Bachelor's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Mathias Iversen
Mikael Røv Mathiassen
Oda Katrine Steinsholt

Software for Virtual Puzzle Reassembly

Bachelor's thesis in Computer Engineering
Supervisor: Jon Yngve Hardeberg
Co-supervisor: Davit Gigilashvili
May 2024

NTNU
Norwegian University of
Science and Technology

Mathias Iversen
Mikael Røv Mathiassen
Oda Katrine Steinsholt

Software for Virtual Puzzle Reassembly

Bachelor's thesis in Computer Engineering
Supervisor: Jon Yngve Hardeberg
Co-supervisor: Davit Gigilashvili
May 2024

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

ABSTRACT

Title: Software for Virtual Puzzle Reassembly		Date: 21.05.2024
Participants:	Mathias Iversen	
	Mikael Røv Mathiassen	
	Oda Katrine Steinsholt	
Supervisors:	Jon Yngve Hardeberg	
	Davit Gigilashvili	
Employer:	NTNU Colourlab	
Keywords:	software development, react, usability, archaeology, computer vision	
Number of pages/words: 107 pages/29887 words	Number of appendices: 12	Availability: Publish

In 1903, a Viking ship, buried in 834, was discovered right outside Tønsberg in Norway. It contained a fragmented tapestry, possibly several of them. The fragments are approximately 1200 years old, and the original colours of what is depicted there have bleached into being almost unrecognisable. Today, the TexRec Project, with international participation, is trying to understand what was once depicted. A part of this is to reconstruct the original(s), but the fragments are very degraded and fragile, requiring minimal movement or handling. Therefore, they will need to be reconstructed virtually.
In this project, a software application called Artifact Assembly was continued to be developed for this purpose. The objective was to add as many features to the software as possible to increase its usability and usefulness. The development process employed the Scrum methodology with short sprints and regular meetings with the product owner. The software was written in React using the Konva.js library and Tauri to make an executable application. The main functionality added to the software was to be able to save a project, improve the filters that can enhance motifs of the fragments, and use a similarity metric to suggest fragments that can be adjacent to each other, etc. Usability testing with an archaeologist and acquaintances, along with an expert UX evaluation, ensures that the software will be useful and usable, advancing how archaeologists work with precious artefacts.

SAMMENDRAG

Tittel: Software for Virtual Puzzle Reassembly		Dato: 21.05.2024
Deltakere:	Mathias Iversen	
	Mikael Røv Mathiassen	
	Oda Katrine Steinsholt	
Veiledere:	Jon Yngve Hardeberg	
	Davit Gigilashvili	
Oppdragsgiver:	NTNU Fargelabb	
Nøkkelord:	programvareutvikling, react, brukervennlighet, arkeologi, datasyn	
Antall sider/ord: 107 sider/29887 ord	Antall vedlegg: 12	Publiseringsavtale inngått: Publiser

I 1903 ble et vikingskip fra 834 funnet rett utenfor Tønsberg i Norge. Det inneholdt et fragmentert veggteppe, muligens flere. Fragmentene er ca. 1200 år gamle, og de opprinnelige fargene på det som var avbildet, er bleket til det nesten ugjenkjennelige. I dag forsøker TexRec-prosjektet, med internasjonal deltakelse, å forstå hva som en gang var avbildet. En del av dette er å rekonstruere originalen(e), men fragmentene er svært nedbrutte og skjøre, så de krever minimalt med berøring og håndtering. Derfor må de rekonstrueres virtuelt.
I dette prosjektet ble en programvare kalt Artifact Assembly videreutviklet for dette formålet. Målet var å legge til så mange funksjoner som mulig i programvaren for å øke brukervennligheten og nytteverdien. Utviklingsprosessen ble gjennomført ved hjelp av Scrum-metodikken, med korte sprinter og regelmessige møter med produkteieren. Programvaren ble skrevet i React ved hjelp av Konva.js-biblioteket og Tauri for å lage en kjørbar applikasjon. Den viktigste funksjonaliteten som ble lagt til i programvaren, var å kunne lagre et prosjekt, forbedre filtrene som kan utheve motivene til fragmentene, og en likhetsberegning for å foreslå fragmenter som kan ligge ved siden av hverandre, osv. Brukertesting med en arkeolog og bekjente, sammen med en UX-evaluering utført av en ekspert, sikrer at programvaren vil være nyttig og anvendelig og bidra til å fremme arkeologers arbeid med gamle gjenstander.

PREFACE

This bachelor's thesis, "Virtual Puzzle Reassembly", is written by Mathias Iversen, Mikael Røv Mathiassen, and Oda Katrine Steinsholt. The thesis was written as the last work for our bachelor's degree, "Bachelor in Computer Engineering", under the Department of Computer Science at NTNU in Gjøvik.

We want to thank Davit Gigilashvili, our supervisor and employer representative from the NTNU Colourlab in Gjøvik and a part of the TexRec project. You have inspired us greatly as you showed your dedication to the project, representing the archaeologists' interests in it. We are grateful for the information you have given us on the TexRec project and all the work you spent commenting on where we could improve our thesis. Thank you, Jon Yngve Hardeberg, our supervisor from NTNU and also a part of the TexRec project, for all the discussions about everything and inputs on the software's features. Without them, the software would not have been the same. Thank you both for this semester of effort.

Thank you, Margrethe Havgar, the archaeologist from Kulturhistorisk Museum/-Museum of Cultural History at UiO. We appreciate your time and effort in testing the software with us digitally and during our visit to Oslo. We look forward to seeing the results you can get with the software.

We want to acknowledge the help from our fellow students and family, including Davit, who has also tested the software, and Ruth Eline Iversen, a UX designer, for evaluating the software using heuristics.

Lastly, we want to thank our fellow group members for all the work we have put into this project this last semester together.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF CODES

GLOSSARY

Adobe Fresco is a software used in image editing, layering and painting. One can edit the colours of an image based on the HSV colour space. 65, 100

Agile is a development methodology emphasising collaboration, customer feedback, and rapid, frequent deliveries. Key principles are adapting to change, continuous improvement, and delivering high-quality results in short, manageable phases called sprints. 29, 30

Artifact Assembly is the software the group members have worked on during this bachelor project. It is an application that will help archaeologists at the Museum of Cultural History discover more about the fragmented tapestries found inside the Oseberg Viking ship. 1, 3, 19, 39, 83, 87, 101

Component is a reusable block of code in the React library that represents a part of the user interface. They can be simple functions representing a button or complex classes representing entire applications. 4, 11, 12, 22–24, 26, 37, 39, 40, 42, 43, 45–47, 49, 50, 57, 96, 98

Computer Vision is a field in computer science that focuses on enabling computers to identify and understand objects in images and videos. 4–6, 12, 16, 24, 65, 84, 86–89, 94, 102

CSS Cascading Style Sheets offers a flexible way to style web content, with styles originating from browser defaults, user preferences, or web designers. 4, 11, 37

Daily Scrum is a daily meeting taking place during a sprint in Scrum. During it, the development team will discuss what they have done since the last daily Scrum and what they will do until the next one. 32, 34, 90

Discord is a digital communication platform designed primarily for creating communities ranging from gamers to educational groups. It allows users to com-

municate through messages, voice calls, and video calls in private chats or as part of communities known as 'servers'. 34, 36

Fragment is a textile object that could come from one of the several theorised tapestries that originally hung inside the Oseberg Viking ship. 1–5, 9, 10, 16, 60, 64, 65, 72, 74, 83, 87, 88, 94, 95, 100, 102

Git is a distributed version control system. Its branching and merging capabilities support non-linear development, making it a powerful tool for collaborative software development. One uses commands like "commit", "push", "update". 35, 37, 47, 96

GitHub is an online platform for software development and version control using Git. It allows developers to store their code in repositories, collaborate with others, track and manage changes to their code over time, and resolve conflicts if necessary. liii, 31, 35, 79, 83, 87, 90

JavaScript is a programming language commonly used to create interactive effects within web browsers, enhancing user experience and web page functionality. 4, 11, 22, 37, 51, 52, 87, 95, 99

JSON JavaScript Object Notation is a language-independent text format that uses conventions that are familiar to programmers and is easy for humans to read. JSON is often used for lightweight data interchange. 37, 52, 98

Kanban is a visual workflow management model. It involves a board with cards representing tasks and columns representing the stages of the process a task can be in. 29–31, 90

Konva.js is an HTML5 Canvas JavaScript framework that extends the 2D context by enabling canvas interactivity for desktop and mobile applications. 22, 25, 37, 45, 47, 87, 93, 95, 96, 100, 101

LaTeX is a markup language especially suited for writing scientific documents. 36

Modal is a pop-up window that appears on top of a GUI. It requires users to interact with it before it can close and usually tells important information, asks for user input, or presents options that need confirmation. 15, 25, 42, 51, 55, 60, 61, 99

Product Backlog is a document used in Scrum that lists items to be done in a project. The list is ordered by importance. 6, 23, 32, 34–36, 63, 89, 90, 93

React is a library for web and native user interfaces. It builds user interfaces out of individual pieces called components written in JavaScript. 4, 6, 11, 12, 22, 37, 46, 87, 90, 91, 101

Rust is a versatile, suitable programming language for systems, web applications, and concurrent tasks. 4, 22, 51, 80, 87

Scrum is a widely used agile team collaboration framework common to software development for managing complex projects. The process emphasises collaboration, regular feedback, continuous improvement facilitated by the Scrum Master and Product Owner, and events like daily meetings, sprint reviews, and retrospectives. 4–6, 19, 23, 29–32, 34, 90, 101

Scrum Master is someone responsible for instituting Scrum by helping everyone in the Scrum team understand it. They are also responsible for the efficiency of the teamwork. 5

Sprint is a period in the Scrum where the development team creates value for the product by doing the items on the Sprint Backlog. 23–26, 29–32, 34, 35, 46, 47, 89–91, 101

Sprint Backlog is a document used in Scrum that defines the goals of a sprint, containing a subset of the items from the Product Backlog. 23, 32, 34–36, 89, 90

Tauri is a framework written in Rust for building tiny and fast executables for all major desktop platforms. With Tauri, developers can integrate any front-end framework that compiles to HTML, JavaScript and CSS for building their user interface. 4, 21, 22, 24, 25, 37, 80, 87, 93, 94, 99, 101

Teams is a communication and teamwork platform with features like chat, video conference, file sharing and application integration. 34, 36, 63, 67

Think-aloud protocol is a term that describes how we want a user to act during a usability test. When following the protocol, the user tries to say what they think as much as possible, for the test host to gain as much information as possible. 14, 67, 69, 71, 88

Waterfall is a sequential development methodology that moves steadily through distinct phases. It is a linear and non-iterative model where every phase must be completed before a new one can start. 29, 30

ABBREVIATIONS

CPU Central Processing Unit 93

ctrl control 15, 22, 23, 69, 72, 75, 100

DOM Document Object Model 11, 12

GPU Graphical Processing Unit 93

GUI Graphical User Interface 4, 11, 19, 37, 52, 53, 97

HSL Hue, Saturation, Luminance 16, 17, 20, 56

HSV Hue, Saturation, Value 16, 17, 25, 56, 57, 65

HTML Hypertext Markup Language 4, 11, 37

IDE Integrated Development Environment 37

KHM Kulturhistorisk Museum / Museum of Cultural History 1, 3, 4, 6, 7, 19, 63, 72, 86, 88, 89, 94, 102

NPM Node Package Manager 80

NTNU Norwegian University of Science and Technology 1, 3, 4, 6, 36, 75, 86, 90, 94

PNG Portable Network Graphics 17

RAM Random Access Memory 52, 93, 94

RGB Red, Green, Blue 16, 17, 95

RGBA Red, Green, Blue, Alpha 16, 17

SDK Sofware Development Kit 80

SDLC Software Development Life Cycle 4, 31, 67

SVG Scalable Vector Graphics 57

UiO University of Oslo 1

UML Unified Modeling Language 26

UN United Nations 88, 89

URL Uniform Resource Locator 22

UX User Experience 4, 12, 16, 54, 71

WIP Work In Progress 30

CHAPTER 1

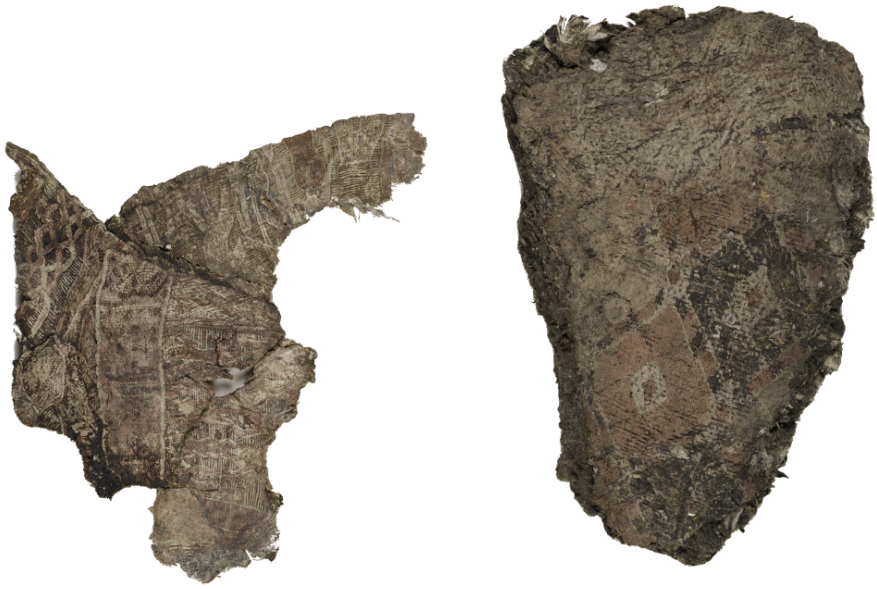
INTRODUCTION

This project was commissioned by the Norwegian University of Science and Technology (NTNU) Colourlab in Gjøvik in collaboration with archaeologists at the Museum of Cultural History (KHM), a part of the University of Oslo (UiO). It is under the supervision of Davit Gigilashvili, our contact at the NTNU Colourlab, and Jon Yngve Hardeberg, the supervisor from NTNU.

The archaeologists at KHM are currently working on tapestries from the Oseberg ship, a Viking ship buried in 834 CE, found in 1903 near Tønsberg, and excavated in 1904 [1]. The TexRec Project [2], which the archaeologists and the NTNU Colourlab are part of, studies the numerous textile objects found inside the ship, ranging from simple wool fabrics to precious silk embroideries. The project focuses on the over 80 fragments found belonging to one or several different tapestries. They are researching how these all fit together. Since the fabrics are very fragile, unnecessary moving should be avoided. Therefore, Ultra-high resolution colour images of the fragments have been taken with a photo camera, and the images will be loaded into a software application to create a virtual reconstruction of the appearance of the tapestries. The TexRec project is funded by the Research Council of Norway [3, 4].

In 2023, Casper Fabian Guldbrandsen was working on his Master's thesis [5]. This thesis was made in collaboration with the NTNU Colourlab together with the archaeologists from KHM. As a part of his thesis, Casper made a beta version of an application called "Artifact Assembly" that could load images onto a canvas to move or rotate the images beside each other (Section 3.1: Initial Software).

However, the software was not completed. It is missing some core functionalities essential to making it practical and usable in the archaeologists' research process. Therefore, this bachelor project's objective is to continue developing the software that Casper started and add the crucial features, or start from scratch and make another based on Casper's software.



(a) Tapestry fragment 1 **(b)** Tapestry fragment 2

Figure 1.1: This figure illustrates two of the many fragments found inside the Oseberg ship, shared by the archaeologists. Pictures taken by: George Alexis Pantos.

1.1 Project Description

The project involves developing a software application with several fundamental features outlined in the task description (Appendix B). First, it should allow loading several images onto a virtual canvas and be able to move them around to try different arrangements. Second, it should allow different colour enhancements and contrast manipulation operations to make patterns on the fragments more visible. Third, it should measure different colour and texture statistics of the images and provide a similarity score or a suggestion to the user about which fragments are similar and, hence, likely to be adjacent to each other in the final arrangement and come from the same original object before segmentation.

Adding to this, after an interview with one of the archaeologists (Section 6.1: Interview 1), we made a list of features they would like to see in the software (Appendix C). A big part of the features listed on the list was to make the software more usable and user-friendly. To get as much progress on the software as possible, The group decided to continue developing Casper's software [5] instead of starting from scratch and not progressing as far.

1.1.1 Delimitations

The goal of this bachelor thesis is not to create an application that contains all the features discussed with the client and intended end-users but to focus on adding the most important features to make the software more useful, usable, and reliable. As written in the task description (Appendix B), we will not create a

fully automatic solver of a puzzle as it seems infeasible at this stage due to them being unlikely to emerge in the near future [6]. And, as previously mentioned, we will not create a new solution from the ground up, but rather continue working on the preexisting software.

1.1.2 Target audience

This Bachelor's thesis is intended for the researchers at the Colourlab working with the archaeologists at KHM, the archaeologists themselves, and the possible future students who may take over and continue working on the Artifact Assembly software.

1.2 Project Goals

The goals of this project are divided into result, impact, and learning goals. The result goals are the specific targets we aim to achieve by the end of the project. The impact goals are the impact that we hope this bachelor thesis will have after we are finished. The learning goals are the knowledge we want to gain during the project and what we hope to retain after the project is over.

1.2.1 Result Goals

Our main objective in the project is to make an application that can be useful and easy to use for archaeologists at KHM and other similar institutions. The software should fill a gap in their work by implementing as many of the features the archaeologists want and outline for us in interviews, usability tests, and other ways of feedback. The resulting software will provide visualisation tools and similarity metrics to help the archaeologists identify matching fragments. Ultimately, we want the software to be something the researchers at the NTNU Colourlab can use to continue their work with the archaeologists at KHM.

1.2.2 Impact Goals

For this thesis, our impact goals include increasing the knowledge and information that archaeologists can obtain from the fragments. We hope the software will enable the archaeologists to try different arrangements to virtually reconstruct the potential multiple tapestries to better understand how the motives tell their scenes and stories. We hope this will allow them to gain further knowledge and then share it further so that everyone can learn more about the Viking age. We also hope archaeologists can use this software in other projects to gain more information there. An impact of using the virtual environment is the improved conservation of the fragments as it minimises the need for physical interaction.

A potential long-term impact could be a gamification of the software so it can be used in an educational setting to teach visitors at the KHM how the archaeologists work to assemble the fragments together.

Another impact goal for this project is to make the project easy to understand and expandable for other programmers so that they can continue working on it,

maintaining it, and adding new features.

1.2.3 Learning Goals

During this thesis, we hope to gain more experience working as a team on a longer project and improve our teamwork skills by getting better knowledge about the Software Development Life Cycle (SDLC) and the Scrum methodology. A part of this is that we hope to gain experience in inheriting partially developed software and continuing development on it.

Since we are inheriting software, we must continue using the same programming languages that have been used until now. Therefore, a learning goal is to get more knowledge and experience with JavaScript and use the React framework to create the application components. We want to learn how to make the software work as a native application. Therefore, we will need a better understanding of the Tauri framework using the programming language, Rust.

We want to learn more about the development of a user-centred Graphical User Interface (GUI) using CSS through user communications and usability testing to give the best User Experience (UX) as possible.

A goal is to learn more about computer vision to extract information from fragments and use the information to compare them. We also want to learn more about using computer vision to make colour enhancements and apply filters to the fragment images.

Our last learning goal is to learn more about interdisciplinary teamwork. As we have to work with two researchers from the NTNU Colourlab and an archaeologist from KHM who have little to no knowledge about software development and computer science in general, we need to learn how to adapt to speaking in terms and language that they can understand, or we can easily explain to them.

1.3 Group Background and Motivation

1.3.1 Group background

The group members of this bachelor's project are all students in the Bachelor in Computer Engineering program at NTNU in Gjøvik. Together, we have much of the same academic knowledge from the courses in our study, with some differences in relevant chosen courses. One took the Web Technologies course, and the other two took the User-Centred Design course.

Web Technologies introduced Hypertext Markup Language (HTML) elements, CSS styling, and JavaScript programming, which is relevant to the development of the software. User-centred design equips us with the knowledge of gathering information from the end users through interviews, usability tests and other processes. It allows us to design the best software based on the information gathered, which is relevant to the development process and design of the software. Furthermore, programming courses have taught us how to learn new languages and coding principles, and the Computer vision course taught us introductory methods

in computer vision, which combined gives us a great knowledge base to complete the bachelor's thesis.

1.3.2 Motivation

When we looked at all the different bachelor thesis options presented to us, we decided to score them all individually to find the ones we all wanted the most on average. The task description outlined in Appendix B was one we all scored high and placed first on the list of our wishes. It stood out as a project that would cater to our preference for a development project, and the result of the project is an application with intended end users. We were also interested because the project would require different fields in programming to finish, such as front-end design, back-end development, and computer vision.

Although we have some knowledge in each field required for the project, we look forward to furthering our knowledge and practical experience with each of them. We are also very motivated to do the thesis since the software will be used by the archaeologists in their work. This project will fill a gap in their workflow, hopefully increasing the information they can get from the fragments, getting to know more about the Vikings, and sharing it with the world.

1.4 Roles

1.4.1 Team member roles

- Scrum Master (Mathias Iversen)
  - Follows up with every member and ensures that everyone contributes to the project and follows the project plan.
  - Responsible for solving disputes within the group and mediating between members if the need arises.
  - Summons all members to Scrum meetings.
  - Leads the Scrum meetings and follows up that everything on the agenda is brought up.
  - Responsible for teaching and implementing the Scrum framework according to the Scrum Guide [7]
- Secretary (Oda Katrine Steinsholt)
  - Writes meeting minutes during meetings.
  - Makes sure that meetings take place and that the meeting schedule is followed.
- Documentation and Communications (Mikael Røv Mathiassen)
  - Ensures all necessary documents are made, placed correctly, and structured.
  - Responsible for communication outside of the group.

- Sources and quality assurance (Mikael Røv Mathiassen, Oda Katrine Steinsholt)
  - Make sure that everything that needs sources has a source and that it is written and structured properly.
  - Goes over the written materials, fixes typographical errors, and ensures that the text is structured correctly.
- Developers (Mathias Iversen, Mikael Røv Mathiassen, Oda Katrine Steinsholt)
  - Will do the work assigned to them and follow code standards.

1.4.2 Non-team member roles

- Supervisor (Jon Yngve Hardeberg)
  - Will supervise the group and help us do the project to the best of our abilities.
- Product owner (NTNU Colourlab in Gjøvik with Davit Gigilashvili)
  - Represents the owner of the program after the project is finished.
  - Responsible for effective management of the Product Backlog in the Scrum. Creating and defining new backlog items. Ordering backlog items.
- User (KHM Archaeologist, Margrethe Havgar)
  - Will be the subject of the software's usability tests and will give feedback on their experience and thoughts.

1.5 Thesis Structure

This section will briefly introduce each chapter and the structure of the thesis.

The thesis starts with an English and a Norwegian abstract and a preface. Then, it lists all the chapters and sections of the thesis, along with a list of figures, tables, code, glossaries, and abbreviations, before the first chapter starts.

1. **Introduction**

   This chapter introduces the project for this thesis, our goals, the group's academic background and motivation, as well as the roles of everyone involved.

2. **Theory**

   Here, some of the theoretical background will be explained, including the TexRec project, React, User-Centred Design and Computer vision.

3. **Requirements**

After describing the initial software, this chapter will list the requirements for the project's software.

4. **Development Process**

   This chapter will discuss the project's development methodology and how it was used.

5. **Implementation, Coding, Production**

   This chapter will describe the technical implementation of new features to the software, focusing on major discussion points.

6. **Interviews**

   This chapter tells the reader about the interviews done before and in the middle of the development process. The interviews were held with an archaeologist at KHM.

7. **Evaluation**

   This chapter will describe the evaluation of the software, based on usability tests and heuristic evaluation.

8. **Installation**

   This chapter contains a detailed description of how to install the software made in this thesis.

9. **Discussion**

   In the discussion chapter, we will discuss our results towards the goals, the usefulness of our software and some research results. We will also discuss the sustainability perspective in our thesis.

10. **Future Work**

    This chapter will describe a lot of opportunities for working further with the project building on what is implemented.

11. **Conclusion**

    In this chapter, we will present a short conclusion of the whole thesis, including some words about the product, the development process and some final words.

12. **Appendices:**

    The appendices are all the documents found essential to describe the project work.

CHAPTER 2

THEORY

This chapter describes the background theory needed to get a greater understanding of this project.

2.1 TexRec Project

The TexRec project consists of 5 areas of research. [2]

1. Analyses of the fabrics.
2. **The puzzle program and virtual reconstruction.**

   Our work to develop a puzzle program to assist in virtual reconstruction falls under this research area.
3. Interpretation.
4. Conservation strategies.
5. Dissemination.

The work done in the TexRec project is time-consuming, and therefore results take time. From a dataset they have, we got to see that there are 25 of the over 80 fragments that they have currently theorised are adjacent to another piece. Figure 2.1 shows 2 of these 25. These and 8 other fragments are part of the biggest group they believe is adjacent to each other.

They use different techniques, tools, and study areas to research and find information on which belong together. Research area 1 focuses on identifying original dyestuffs despite significant bleaching. They use well-established techniques such as HPLC [8] and adapted SERS [9]. The decomposition products will help infer the original colours of the fragment. Mock-ups they create will serve as a reference for studying dyeing techniques, conducting ageing and decomposition studies, and

Figure 2.1: TexRec Item Nr. 53 (left), 55.1 (right), and 55.2 (top). Theorised to be adjacent to each other

investigating interactions with conservation agents. Advanced photography, high-resolution electron microscopy, and biochemistry will explore fibre degradation at the molecular level.

Research area 2 focuses on research for virtually reconstructing the tapestries. The fragments are fragile, and unnecessary moving must be avoided. The Ultra-high-resolution colour photos will be used to determine the thickness of the threads, weaving characteristics, and others. They will be put into a Puzzle Program to assist the archaeologists in creating a virtual reconstruction of the tapestries.

Research area 3 focuses on interpreting the motifs which are depicted in high detail on the tapestries. Most seem to be dedicated to warfare and religious ceremonies: male and female warriors, a warrior on a wagon without a horse, a procession, a grove, and a tree with a hanged man. If they manage to piece the fragments together, they can better understand the stories depicted on the tapestries. Do they match already known stories or new ones?

Research area 4 focuses on developing proper conservation strategies for the fabrics. Current methods suffer from several drawbacks. Therefore, they aim to explore new conservation agents focusing on bio-inspired materials. Using the motto "Learning from nature", they evaluate different biopolymers, aiming for tailor-made and biomimetic consolidants. The envisioned material will help preserve the Oseberg textiles and, hopefully, textiles in other collections.

Research area 5 focuses on the dissemination of the research they are doing.

Holding academic presentations and publications, as well as outreach to the public. They will plan small courses and demonstrations, including experiments on historical dying techniques.

2.2 React

React is a framework library that helps build graphical user interfaces for web and native applications [10]. The core of React lies in making reusable components representing parts of the GUI defined by its unique logic or appearance. This could be anything from a whole web page to a button.

2.2.1 Component

A component is written in JavaScript either as a class that inherits from the React Component class or, more commonly used, as a function using React hooks, as we have done in this project. Functional components are easier to use and open up for the use of hooks. Therefore, this is the most used type of components today.

The components can be nested, meaning that a component can consist of other components. Like a component describing a web page having button components.

The component is commonly defined using a markup syntax called JSX, even though it is not required. JSX looks similar to HTML, but can dynamically use JavaScript to get data or call functions using curly braces "{...}", to return to using JavaScript. The markup is also stricter than HTML since you have to close all tags, and you can not return multiple tags and therefore have to put the tags into an empty tag "<>...</>" called a "fragment" to do so. You could also add CSS styling to the component in JSX using the reserved word "className" instead of "class" in HTML that will reference the style class in a separate CSS file. You could also display a list using the map function to return a list of components.

2.2.2 DOM

Simply put, the Document Object Model (DOM) is a way to represent HTML elements structurally as a tree. This is being used in several programming languages. That means the whole user interface is represented with the DOM, with each element as a node. The DOM needs to be updated each time the tree is rendered. React uses a virtual representation of DOM. In that way, React will update the value of the virtual representation of the DOM before replacing the original DOM with the virtual one. React has a reconciliation process that will calculate the most efficient way to update the virtual DOM. Therefore the whole tree will not be changed, but only the necessary parts. [11]

2.2.3 Hooks

There are different types of functions that one could use in React. These functions are called hooks. React has many built-in hooks, but it is also possible to create your own hooks by combining existing ones. Hooks can be divided into state management, performance, context, and reference hooks.

State management hooks help you keep the components data between render calls. The most common state hook is useState. This hook takes an initial state in as input and returns a list containing the last state constant and a setter function for updating the state.

Performance hooks are also an important type of hook. This includes useCallback and useMemo, which cache functions or function results between renders. Both hooks need two inputs: a function and an array of the attributes this function depends on.

Context hooks subscribe to a context to get data or functions provided by the context. A context hook will get the provided values from the context higher up in the component hierarchy. A normal hook here is useContext. Hence, useContext is a hook that is preferred when the programmer needs to pass information deep into a component tree. The context is made using the React function createContext.

Reference hooks are a fourth kind of hook, used when the programmer wants to refer to the DOM object. The useRef is a hook that will hold information that will not be used for rendering, this can be a DOM node. Therefore, the component will not have to re-render each time the useRef hook is updated. [12]

2.3 User-Centred Design

User-centred design is an important principle for the development of this software. To make an application useful for users, a developer should involve users in the development process as much as possible from an early stage. Focusing on the user experience can make a solution more enjoyable and prevent frustrations for users interacting with it.

A bad design usually results in a cost of time and money. In this bachelor's thesis context, the risk can be that the resulting software would not be used.

In UX design, one usually takes on an iterative process. In this project, the development team's process started with an understanding phase, followed by changes based on these insights, evaluation of the resulting design with users, and repeat.

2.3.1 Understanding Phase

This is a period during which the goal is to understand and explore the users' needs and problems and gain insights into their actions and attitudes. The design team questions the assumptions they may have and empathises with the target users. At the end of the phase, the team establishes the requirements needed to make the product more valuable to the users.

Usually, in this phase, there should be some desk research. The team read what had already been done in research using computer vision to aid the archaeologist in reconstructing the tapestries [13]. The software of the bachelor project was also continued from the master thesis of Casper Gulbrandsen [5], so this was an important start for the research.

User research is the primary way to obtain insights from users. The bachelor team focused mostly on qualitative research tailored to the intended users. Therefore, the team held in-depth interviews rather than breadth-focused questionnaires.

2.3.1.1 Interviews

In an interview, one should avoid poorly structured questions, double-barrelled questions that touch on multiple issues, and vague or leading questions. One should ask open-ended questions.

During the interview, one has to take an active listening approach, making room for silence and staying neutral to the best of one's ability, never correcting the interviewee.

2.3.2 Evaluation Phase

After designing a solution, an evaluation should be performed. This involves reviewing the solution by testing whether it meets some criteria. The goal is to assess the usability of aspects of the solution, validate whether the design solved the problems, and compare the solution to other designs. Also, to learn what works or not, not to prove that the design is finished. It is a way for the design team to test their assumptions about the design. The last favour the evaluation can do is find bugs that have not been discovered before.

In the development process of this software, the team chose to do three iterations of usability tests to evaluate it. These tests were summative, meaning they tested the high-fidelity product against a set of metrics. The first was remote, and the second was on-site, both testing the intended users. The third iteration of tests were guerrilla usability tests, meaning testing the software on random people the team got hold of. These people consisted mainly of students and acquaintances. By testing, they got invaluable direct feedback from the users.

2.3.2.1 Usability Test

This test evaluates how easy it is for a user to use an interface. I.e. testing the ease of use. Usability is defined by 5 components of quality [14]:

- Learnability - How easy is it for a user to accomplish a simple task the first time they encounter the design?
- Efficiency - How fast can a user perform a task after learning it?
- Memorability - How easy is it for a user to return to proficiency after a period away from the design?
- Errors - How many errors do the user make, what is the severity of the errors, and how easy can the user recover from the errors?
- Satisfaction - How pleasant is it for the user to use the design?

A design is useful for users if it has both utility and usability. Utility means that the design solves a real problem or is needed or wanted by the users. A

usability test measures usability, not utility. Utility is usually evaluated in the understanding phase.

In a usability test, the design team watches users interact with the team's design. They asked the user to complete concrete tasks and gather feedback for further improvements on the design's usability. The Think-aloud protocol is a way to get more user feedback. This involves making the users speak out what they are currently thinking during the task. This will be things like what they are doing, looking at or expecting rather than their opinion about the design or what they think other people would think about the design [15].

2.3.2.2 Heuristics

Heuristic evaluation is a type of expert evaluation. This means that one uses person that is an expert in design, usability or interaction design and is an expert in the relevant field, instead of an end user to evaluate the solution. Expert evaluation should not be used as a substitute for testing with the target end users, but can still be an effective way to find the main usability issues or other issues with the solution. The arguments for using of this type of evaluation are that it is usually quicker than a typical usability test and can also be less expensive if one has access to an expert. The negative is that it does not uncover all the issues with the solution. The reason often is that the experts are not the target users, so they may have biases that do not reflect the end users.

Heuristic evaluation involves having experts evaluate an interface and then judge it based on a set of heuristics. A heuristic is a design principle recognised as a guideline or a rule of thumb. There are different sets of heuristics. In most cases, these are developed from many years of experience with evaluation and seeing reoccurring patterns.

In the field of usability, one of the most used heuristics in the industry and academia is Nielsen and Molich's 10 User Interface Design Heuristics [16]. These were created after the Nielsen Norman group conducted hundreds of usability reviews and grouped the frequently occurring usability issues they discovered. Then, they created guidelines with descriptions and examples of how to solve or prevent those issues. These 10 heuristics are:

1. **Visibility of system status**

   An interface should inform the users about what is happening in the system as soon as possible. An example in our software shows "loading in images..." at the right side of the navigation while the images load in.

2. **Match between the system and the real world**

   An interface should use the language familiar to the users rather than technical jargon. An example from our software is that we changed "Similarity Score" to "Similarity Metrics" since this was closer to the user's language.

3. **User control and freedom**

   An interface needs a clear and easy way to undo an action if users make it by mistake. Undo and redo is a way this has been accomplished in our

software.

4. **Consistency and standards**

   An interface should follow platform and industry conventions, meeting users' expectations when using words, situations or actions so the users do not have to wonder if it means the same thing. We implemented control (ctrl) + Z to undo and both ctrl + Y and ctrl + shift + Z to redo shortcuts since both are used in other applications.

5. **Error prevention**

   Mitigate error-prone conditions to prevent users from making errors. From our software, an example would be the confirm modal that pops up when returning to the landing page, asking if the user remembered to save. Or the button that will return the user to the work area, should they lose it.

6. **Recognition rather than recall**

   Minimise the elements, actions or options the users need to memorise between parts of an interface. They should be visible or retrieved easily. An example is our use of icons on the undo and redo buttons.

7. **Flexibility and efficiency of use**

   Speed up the interaction with an interface for expert users by lending them more options hidden from novice users. Examples are the key shortcuts in our software: undo, redo, save and delete.

8. **Aesthetic and minimalist design**

   An interface should not contain irrelevant or rarely needed information. In our software, we implemented the most valuable features and removed unnecessary features like the moving grid or the resize feature.

9. **Help users recognise, diagnose, and recover from errors**

   An interface should give good error messages in plain language, indicating the problem and suggesting solutions. An example from our software is the alert when the software can not generate an image of the canvas because it is too large, prompting the user to lower the setting.

10. **Help and documentation**

    The best interface does not need additional explanations, but in some cases, an interface should provide documentation to help users. An example is the information pop-up in the modal when exporting an image of the canvas in our software.

To conduct a heuristic evaluation, one needs to know what to test and what the objective of the test is and clearly define this to the evaluators. The evaluators need to know the users and clearly define the users' goals, context, and so on. Here, one could use user personas and user profiles representing the wants and needs of some of your target users. This can help the evaluators see the interface from the users' perspective. According to Nielsen, one should have 3 to 5 evaluators

who are both experts on usability and the relevant industry. The heuristics also need to be defined. One could adapt the heuristics to reflect better the interface or solution, which needs to be communicated with the evaluators. One should conduct the evaluation individually before accumulating the insights to reduce group confirmation bias.

In this project, we conducted something similar to a heuristic evaluation by asking one UX designer to give some feedback on the software (Section 7.2: Heuristic Evaluation). This was done casually and did not follow the heuristic practices very thoroughly. For example, the UX designer had little to no insight into the archaeological industry and our target users. The team still think they got some feedback that revealed some of the most major usability issues with the design at the time.

2.4 Computer Vision

Computer vision is a subfield of computer science which addresses how a computer understands images or video. The goal is to make the computer understand the information behind the literal values of the image. This refers to real-world properties, such as shapes, objects, distance, and texture. Without any algorithmic processing of an image, the computer only understands the image as a matrix of values, where each matrix element is called a pixel. The computer vision algorithms will use the matrix as input, then try to emulate the human vision or better [17]. For example, by mimicking how the human brain works through machine learning. In this project, the goal is to make an application that can aid archaeologists in understanding and retrieving information from the fragments. This aligns with the goal of computer vision. Computer vision involves capturing or acquiring information, such as an image, by a sensor, processing this image, and analysing the image into a model. It is often done using a pipeline consisting of image acquisition, pre-processing, feature extraction, detection and segmentation, high-level processing, and application. Filtering is a part of this. So is segmentation and image enhancement. [18]

The main problem with this project is the deterioration of the tapestry fragments. The fragments are captured in high-quality photos taken under controlled light conditions to be as well-lit as possible, but the fragments themselves are deteriorated. The fragments in the images needs to be segmented from the background. The team did not implement this feature since the images were pre-segmented. However, the team proposes implementing segmentation as a feature for future application development (Section 10.15: Segmentation).

2.4.1 Colour Spaces

A colour space or model is a virtual coordinate system to describe the colour of a pixel on an image through values. Some colour spaces important to this project are the Red, Green, Blue, Alpha (RGBA), the Hue, Saturation, Luminance (HSL) and the Hue, Saturation, Value (HSV) colour spaces.

The RGBA is the three-valued colour space of Red, Green, Blue (RGB) aug-

mented with the alpha channel that expresses the opacity of the pixel [19]. The RGB is an additive colour model [20], adding the three channels representing the primary colours red, green and blue together to represent the pixel's colour.

Colour spaces that are more intuitive for a human to understand than RGB are the HSL or the HSV colour models (Figure 2.2). These are both cylinders that represent every colour value. Instead of "mixing" three colours to make the final colour as RGB does, one rather builds it up by other values that are understandable from a perceptual point of view.

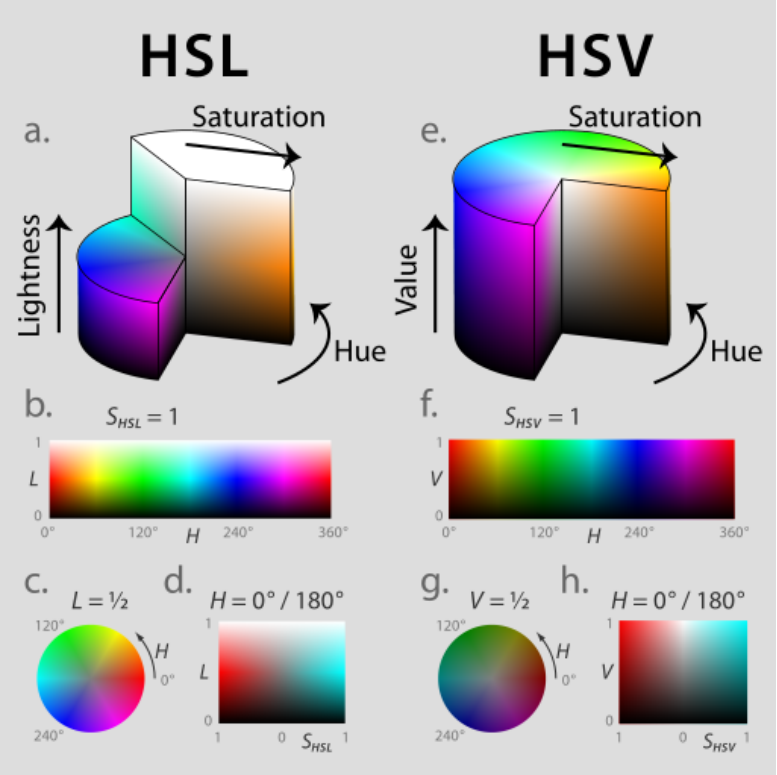


Figure 2.2: HSL and HSV Colour spaces [21]

HSL stands for hue, saturation and lightness, and HSV stands for hue, saturation and value. The hue is represented as a 360-degree circle along the central vertical axis in both colour spaces, where the colour red is both the values 1 and 360. The saturation is represented by the axis going from the middle of the cylinder out to the edge, the lowest value of 0 being a version of neutral grey and the highest value of 1 being the hue. The central vertical axis represents the lightness in HSL or value in HSV. In the HSL colour space, the lightness starts at black at 0, makes its way to the pure colour at 0.5, and ends up as white at 1. In the HSV colour space, the value starts at black at 0 and makes its way to the pure colour at 1 [22].

The image files used in this application use images of file type Portable Network Graphics (PNG) that represent the colours through the RGBA colour space. The hue value used in the histogram (Task 4c p.24) and the hue metrics (Task 4b p.24) were derived from these values. During the development, we tried to use different colour space filters on the images, both HSL and HSV. Through user feedback, the team ended up using only the HSV colour space filter.

2.4.2 Image Histogram

A histogram is the statistics of how many pixels in an image have a certain value. These values could be any discrete value describing the image, such as the grey-scale intensity or one of the colour space values. Although one can not reconstruct an image from its histogram, as it does not say anything about the spatial distribution of the pixels, the histogram can help us detect acquisition issues like brightness, contrast, or dynamic range, such as overexposure and underexposure. Based on the values in the image, one can then select some threshold for the image. One can also enhance images with histogram equalisation, stretching, or matching.

A normalised histogram contains the probability of the value occurrence instead of the number of occurrences. This is calculated by dividing the histogram values by the number of pixels in the image.

CHAPTER 3

REQUIREMENTS

Here are described the initial version of Artifact Assembly and the project requirements. The requirements were formed during the Scrum development to fulfil the result goal of making a useful and easy-to-use application for the archaeologists at KHM.

3.1 Initial Software

During his master's thesis, Casper F. Guldbrandsen made an initial software, to help users reconstruct images of artefacts from images of damaged pieces as a virtual puzzle on a canvas. As part of this bachelor project, we were to make a similar software. We could, therefore, decide if we wanted to continue developing from Casper's work or start from scratch and recreate the software's functionalities. We chose to continue the development of Casper's Artifact Assembly. Here is a more detailed description of the state of the software as it was when we started.

3.2 GUI

This was the state of the GUI in this version.

3.2.1 Landing Page

When starting the software, the user was greeted with the landing page (Figure 3.1). The user was welcomed to Artifact Assembly and could either open the Canvas or quit. On the right side of the screen was a tab named Projects. Here Casper intended for projects that were saved to appear in a list, but this was never implemented.

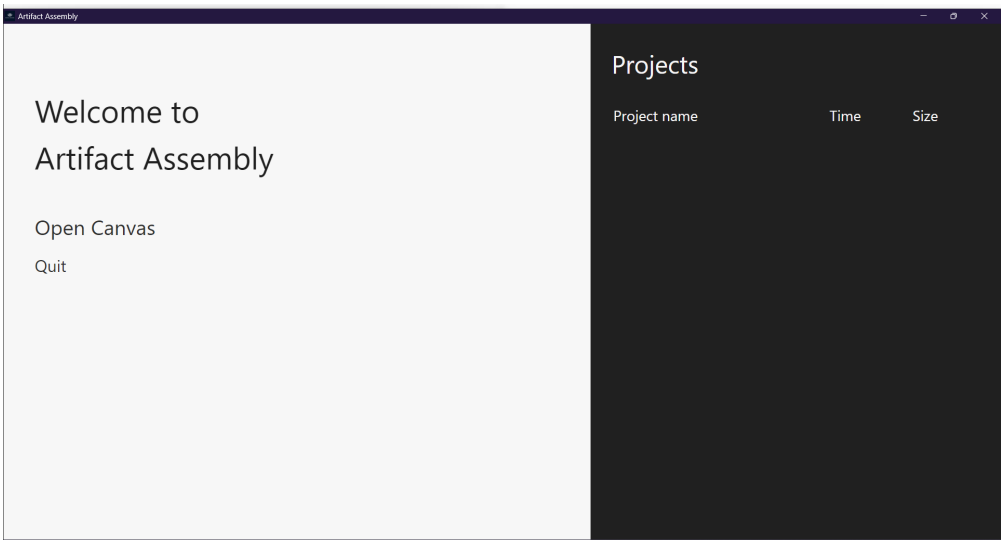


Figure 3.1: The landing page of the software.

3.2.2 Canvas

Selecting to open the canvas would open up a new canvas (Figure 3.2). There, the user could load images onto the canvas. The user could select an image and move or rotate it on the canvas. All other options the users could interact with were in the navigation bar at the top of the screen. Here, the user could choose to go "Home" to the landing page, "Load Image" onto the canvas, "Lock Canvas", which would lock all the images in their position so none could be moved away from each other, and to "Enable Filter".

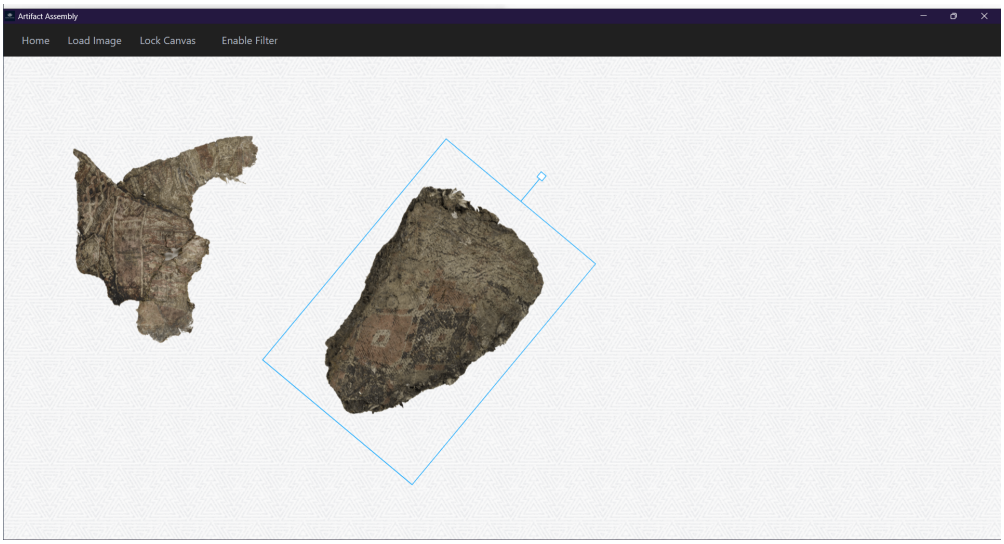


Figure 3.2: The canvas that opens. A couple of example images of fragments have been loaded in. One is selected and has been rotated.

3.2.3 Filter

Enabling the filter would open four number inputs, one each for hue, saturation, and luminance, representing the HSL colourspace, and one for contrast. Changing

values in the number input fields would change the look of all the images (Figure 3.3). The hue filter was not gradual but would rather rotate between red, green, blue, and magenta with slightly varying differences.

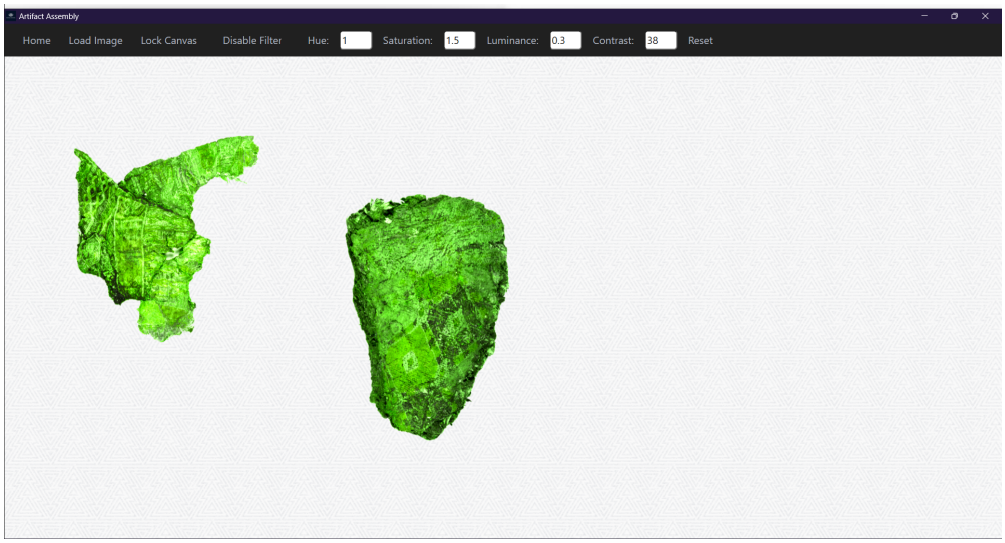


Figure 3.3: Here, the filter has been enabled, and some values for the hue, saturation, luminance, and contrast filters have been set.

3.2.4 Context Menu

The user could also right-click the canvas to open a context menu (Figure 3.4). This menu was provided by Tauri and had the option for saving or copying an image of the canvas or taking a cropped screenshot.

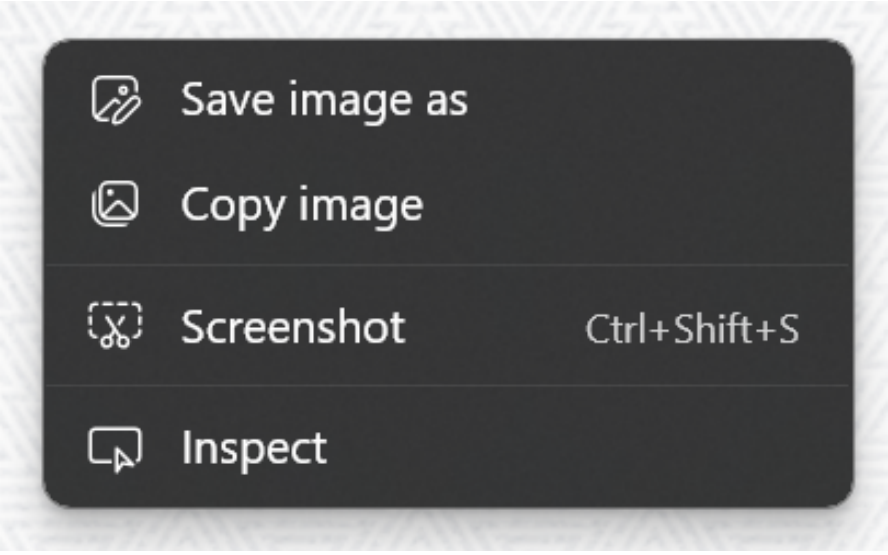


Figure 3.4: The Tauri context menu

3.3 Technical Design

3.3.1 Language and Framework

The original software was a native application written in JavaScript using the framework React, and Tauri to deploy it using Rust as the back-end. The main library used in the software is the Konva.js library, used for the canvas, the images and the image filters.

3.3.2 Image saving

The user could save the project to the Tauri browser's local storage using ctrl + S. This was buggy, and it was hard to keep track of the images saved in the system. The saved data was also inefficient by saving the Uniform Resource Locator (URL) containing the whole image encoded as base64. Base64 encoding is another way to store an image as plain text instead of as a file.

3.3.3 Components

The application's root component is App (Figure 3.5). It contains the whole application. Under it are the two Route pages that are endpoints of the application, LandingPage at the "/" endpoint and Canvas at the "/canvas" endpoint.

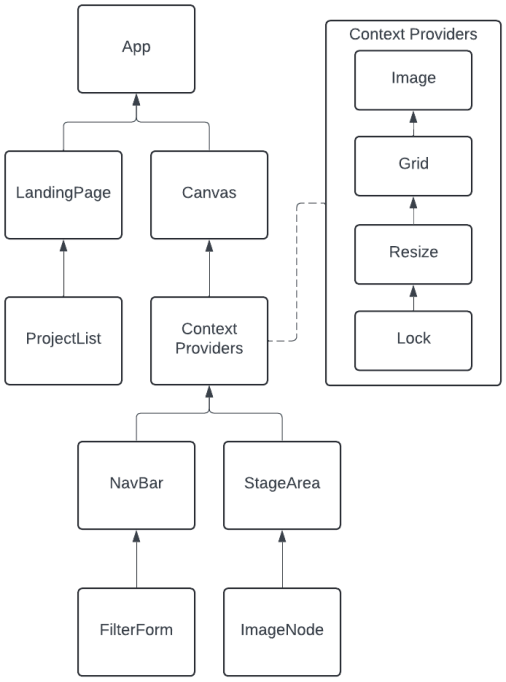


Figure 3.5: The React component tree of the initial application

The software has components representing the landing page, the canvas, the navigation bar, the filter form, the stage area where the user can move the images, the filter form for the number input for the filters, and the image node representing an image on the canvas.

3.3.4 Contexts

This version utilised the useContext hook for sharing data and functions between components. The contexts were all at the Canvas component level. The data that the context shared were placed in useStates in the Canvas component. The contexts were the ImageContext, GridContext, ResizeContext and LockContext. The contexts all held the same provider values, making all but one redundant. These values were: “grid”, “resize”, “lock”, “images”, the filter values of “filter”; “saturation”, “hue”, “contrast”, and “luminance”, and the setter functions for all of these values.

One of the code files contained a block of commented-out code for snapping the images on the canvas to a grid position when moving them. The “grid” value was then used to determine whether this functionality was enabled. The “resize” value was whether the resizing images were enabled. This was connected to a commented-out button on the navigation bar. When the “lock” value is true, it locks the images from moving on the canvas. The "images" value was an array of all the image state prop objects used by the ImageNode component to make the images on the canvas.

3.3.5 Undo

The user had the option to undo using the ctrl + Z key shortcut, but it had some bugs. The code for this was also mixed with code that has other responsibilities in the StageArea component.

3.4 Product Backlog

The project’s Product Backlog (Appendix E) was initialised according to normal Scrum practice. This was an excellent tool to use in the development process. The team could add more items to the backlog as more user feedback was obtained. For example, items were added to this backlog after every usability test or interview. Some of the items were also split up into smaller sub-items or redefined when the tasks of the item were too big or as it was needed.

At the beginning of every Sprint, the team could then see what tasks that was next in the prioritised order and add them to the Sprint Backlog (Appendix F) as requirements for that Sprint.

This is the list of the functionality requirements for the application the team decided at the end:

1. Sprint 1
    (a) **Select multiple elements:** The ability to select more elements on the canvas by clicking and holding a selecting key like the ctrl or shift key. These elements can then be moved, rotated and deleted together.
    (b) **Project file:** The project could be represented through a file that contains information about the project. This will not contain direct image information to reduce file size. This file will include:

- Project name
- Position on the canvas as coordinates
- The zoom scale of the canvas
- The info on all the elements on the canvas:
  - The type of element
  - The position of the element
  - The rotation of the element
  - The identification of the element
  - If the element is an image, it will contain:
    - The file name of the image file
    - The file path to the image file
    - If the image has filters, the filter values

(c) **Saving and opening projects:** The feature of making a new project, saving a project and opening an existing project using the Tauri dialogue windows for saving and opening the project files. This makes it possible to have multiple projects.

(d) **Export image of the canvas:** Give the users an intuitive way to save an image of the current canvas view in different resolution scales.

2. Sprint 2

(a) **Similarity metrics window:** Create a window that can display in a meaningful way the different computer vision comparison scores between images.

3. Sprint 3

(a) **Grouping images:** The ability to lock or combine the selected canvas elements into a group that will be utilised as a unified canvas element.

4. Sprint 4

(a) **Filter slider and toggle:** A component that contains a slider, spin box and reset button for changing the filter values on an image. Also, a component to turn on or off filters that either can only be in those states.

(b) **Hue Histogram metrics:** Implement different metrics to compare the hue histograms of the different images.

(c) **Hue histogram:** In the similarity metrics window, display a bar graph for each image. This graph represents the number of pixels in the image with a hue value between two integer hue degrees per bar.

(d) **Image Filters:** Implement the ability to add multiple different filters to an image. These filters are in the Konva.js library. This includes the following filters:

- HSV colour filter
- Contrast and luminance filter
- Mask threshold filter
- Greyscale filter
- Invert colour filter

(e) **Improve undo and redo:** Improve the implementation of undo and redo by making the redo function and separating the logic for undo and redo away from unrelated code. Additionally, implement common key shortcuts used in other software and buttons in the navigation bar for undo and redo.

(f) **Confirm save modal on return:** Make a pop-up modal to ask the user to save the project when returning to the starting page. This is to improve the software usability by preventing the error of losing the project work by not saving before returning to the starting page.

(g) **Find work area function:** A function to move the canvas automatically so that the closest image is in the centre and the zoom scale is 1. This will be done in a smooth animation.

(h) **Filter window:** Create a window where users can add, change or remove filters on the selected images.

(i) **Improve similarity metrics window:** Improve the design of the similarity metrics window and make the similarity show information on the selected and compare scores based on the selected.

(j) **Export Image modal:** When exporting an image of the canvas, make it more intuitive for the users by making a pop-up modal. This will also prompt a Tauri save dialogue window to allow the users to save the image with a specified name in a specified place.

5. Sprint 5

(a) **Improve the filter window:** Based on the feedback from the second usability test, make changes to the filter window. This includes:

- make an open filter button to the navigation bar instead of right-clicking on an image to open the filter window.
- removing the luminance filter
- changing the name of the value in the HSV filter to brightness
- changing the name of the reset filters button
- changing the name of the enable/disable button

- other design changes

(b) **Make filters apply to the selected images:** Apply the filters not only to one image but to all the selected images by overwriting the previous filter for all of them. Also, add a button to the navigation bar to open the filter window.

(c) **Undo and redo arrows:** Make the undo and redo buttons in the navigation bar use intuitive arrow icons instead of text.

6. Sprint 6

(a) **Score table component:** Refactor the score table into a component rather than a function. Also, make the table sort when the users click the table headers.

(b) **Change the structure of the data:** Change the structure to make it feasible to implement the functionality of locking images together into groups.

Figure 3.6 is a Unified Modeling Language (UML) use-case diagram showing the intended use of the software. The intended users, represented by the stick figure, can make actions represented by ovals. An action can "extend" another action represented by dotted lines. This means the action can be performed after performing the action it extends, for example, after making a project the user can load an image and then select it and move it or rotate it. The "Open Project" action inherits from the "Make Project" action using the white triangle arrow. This means that all actions extending "Make project" also extend "Open project".

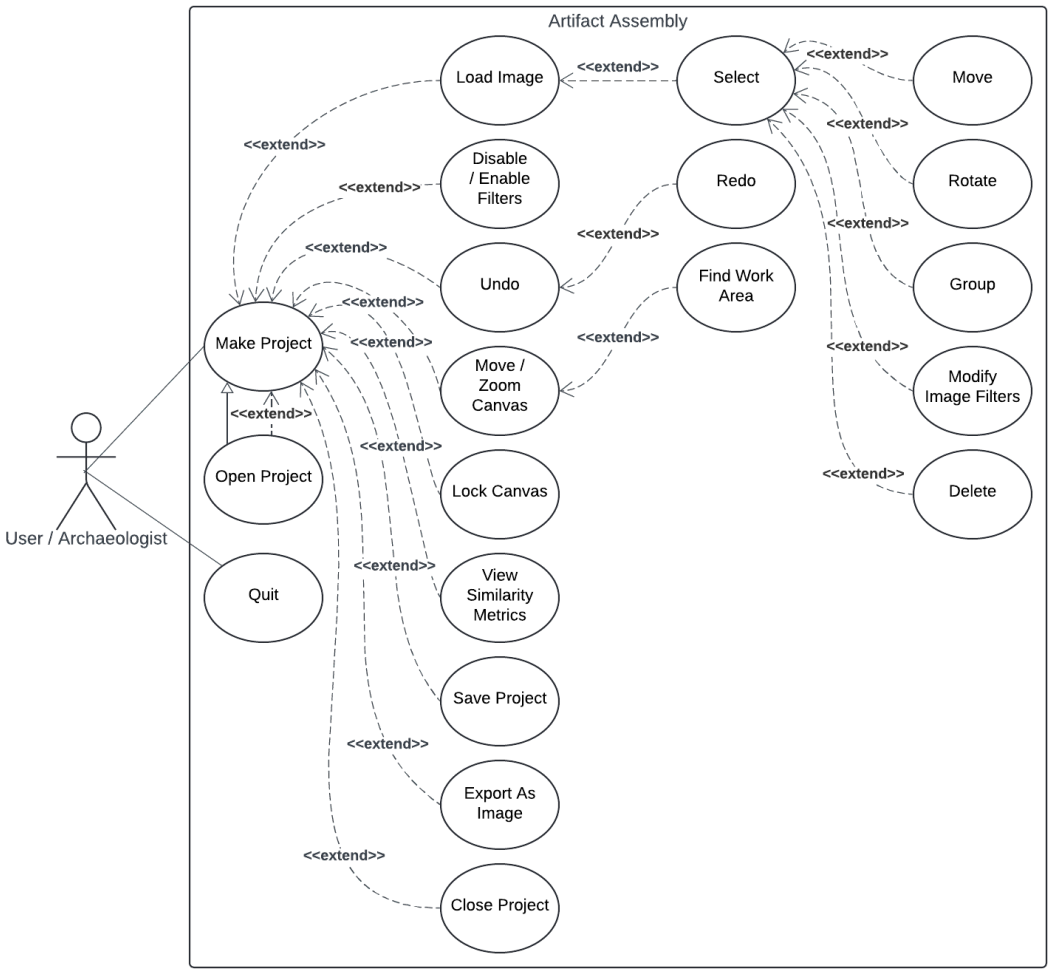


Figure 3.6: Use Case Diagram of the intended finished software

CHAPTER 4

DEVELOPMENT PROCESS

This chapter describes the development process of this project. This includes a discussion of which development methodology we used, how we used it, and the tools used during the development.

4.1 Development Method

In the software development industry, many development methodologies exist, such as Scrum, Waterfall, Kanban, Agile, etc. Each offers advantages and is tailored to different types of projects and teams. Choosing a suitable development methodology is crucial as it can influence the project workflow, our response to new challenges, our communication patterns, and our requirements.

4.1.1 Discussing different methodologies

The Scrum methodology allows teams to get work done in small pieces at a time. It is an empirical process that emphasises collaboration and a feedback loop to learn and improve as time goes on. Its advantages lie in its short sprints, which enable constant feedback so changes and challenges are easier to handle for smaller, fast-moving projects. Daily meetings between the developers to follow and measure each developer's productivity and progress on each task make each Sprint highly informative and adaptable. A disadvantage with Scrum is scope creep, which means it's easy to be tempted to keep wanting to add new functionality to the software. [23, 24]

The Waterfall methodology is a simple idealistic model. It is useful for large-scale projects with well-defined requirements and long timelines. It has a sequential approach where each part of the project is completed before moving on to the next, emphasising testing at each part of the project. This requires rigorous project planning so that the project team is working towards clear and well-understood goals. A downside to the clear goals is the challenge of accommodating changes in

the project and the limited feedback since it assumes there will be no error during any development phase. Going back after the testing stage may cause problems. [24, 25]

The Kanban is a development methodology that uses a Kanban Board to visualise the workflow of the entire project. The board is divided into columns, and these columns represent the phases of development: To Do, In Progress, Validating, and Completed. It keeps track of Work In Progress (WIP) items and limits their amount. It is best used in scenarios where there are frequent changing requirements and continuous incoming work. With a limited amount of WIP items, new priority items can be moved to the next stage before other items. It will also allow for frequent releases, with user feedback available after each release. The disadvantage of Kanban is that the items are difficult to track over a long period, which is not suited for projects where items are too dependent on each other. [26, 27]

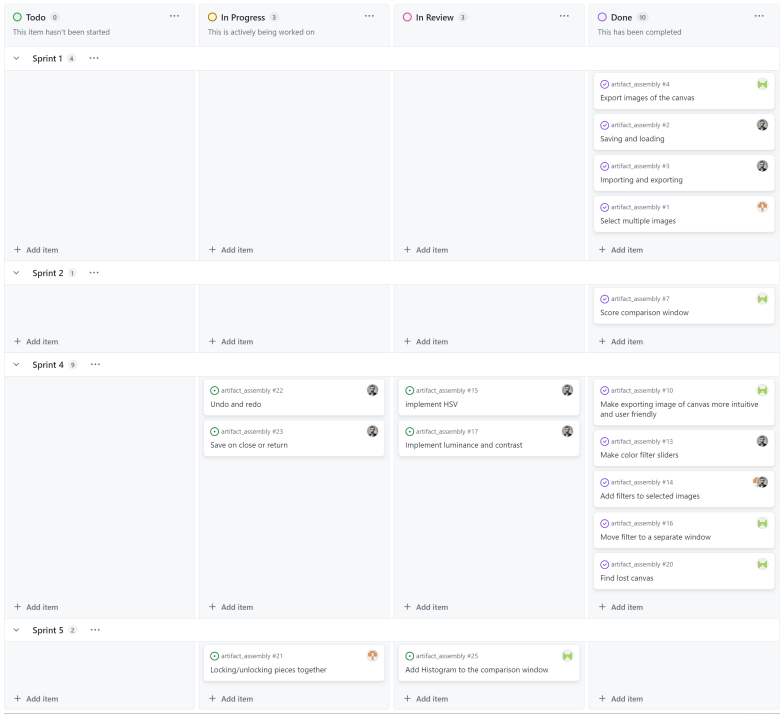
The Agile methodology values flexibility, collaboration, and customer satisfaction. It is an iterative and incremental approach emphasising delivering a working product quickly and frequently. It starts with little planning and executes an idea, then adjusts accordingly based on feedback. The development team collaborates closely with the customer to ensure the product meets their needs. It has a flexible and adaptable approach, which is able to respond to changing customer requirements, but it can easily get taken off track if the customer is not clear about the outcome they want. Since it prioritises working software over documentation, it can lead to further misunderstandings and difficulties later in the project. Agile can be intense and fast-paced, with frequent Sprints and deadlines, which can easily lead to burnout. With less planning initially, hard decisions are required in the early stage of the project, which can be difficult to adapt to for programmers new to Agile without proper resources. [24, 28, 29]

4.1.2 Development model decision

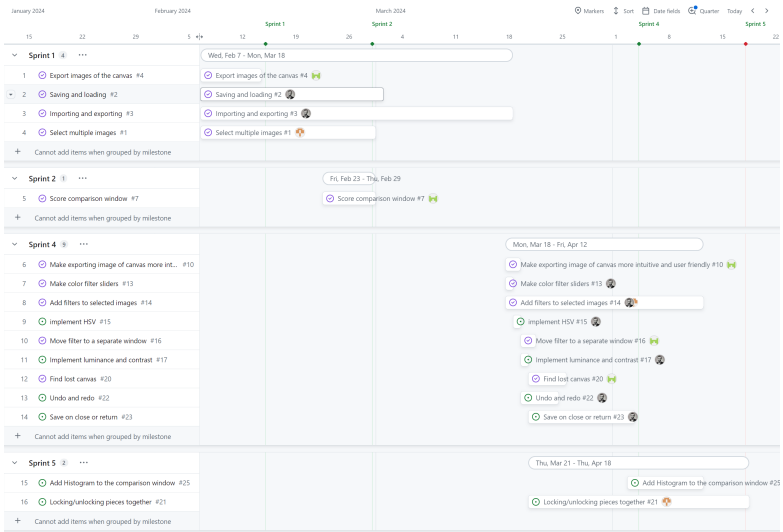
After the first interview with the archaeologist, we decided on the development method we would use during the project. The base software was already made, and we were adding features to it and making it more user-friendly. With many ideas of features to add, we decided on using Scrum based on its agility, which would allow us to adjust the requirements during the development to better align with the needs and preferences of the archaeologists. Additionally, Scrum is a methodology we have been using previously in a casual setting, so we thought it would be useful to learn how to implement it in a stricter sense. The method also lends itself to learning by having an iterative cycle of Sprints with a retrospective at the end where the team could reflect on the development process and their implementation of Scrum.

The decision was also based on the open-endedness of the task description (Appendix B). We felt like the Waterfall model would be too rigid to fit this project since it would be challenging to plan ahead for everything. With Agile, we felt it would be too open-ended. We had some concrete goals for the project that were outlined in the task description. Although we did not choose Kanban, we did want to incorporate some part of it into our workflow. We liked the visualisation

we could get with a Kanban board, so we used GitHubs built-in functionality to create a Kanban board for our issues in the GitHub repository (Appendix G). When we had already created the board, we also created a Roadmap using the same issues to get another view of our progress through the Sprints (Figure 4.1).



(a) Kanban board near the end of the project



(b) Roadmap near the end of the project

Figure 4.1: Visualisation of the project

4.1.3 How we used Scrum during the project

Throughout the SDLC, we tried to implement the Scrum methodology to the best of our abilities according to the Scrum Guide [7]. Following the Scrum Guide, we used it in an iterative process that goes as follows:

1. The product owner orders some problems to be solved by inserting them into a list called Product Backlog (Appendix E). This list is arranged by the level of importance-based value for the users and the difficulty of implementing it (Table 4.1), where a lower level means higher priority.

Table 4.1: Priority matrix of the Scrum Backlog items

		Difficulty for the developers		
		Low	Medium	High
Value for the users	High	Level 1	Level 2	Level 3
	Medium	Level 2	Level 3	Level 4
	Low	Level 3	Level 4	Level 5

2. The Scrum team plans a development period called a Sprint by picking a subset of items form the Product Backlog and defines a goal. The resulting document is the Sprint Backlog (Appendix F). The developers then complete the items in the Sprint Backlog during the Sprint.

3. During the Sprint, the developers have a short meeting of 10 minutes each day called Daily Scrum. Here, they each discuss what they have done since the last Daily Scrum and what they will do until next Daily Scrum so that the team may come closer to and complete the items of the Sprint Backlog.

4. When the Sprint is over, the Scrum team and other stakeholders inspect the results of the Sprint and improve and plan for the next Sprint.

5. Repeat the process.

This project's development process used a Sprint length of 2 weeks. This was chosen since we did not have too much time, and anything smaller would cause too much overhead. We had 7 Sprints where the first 6 Sprints was mostly development, and the last was a pure report writing Sprint (Figure 4.2). We had two iterations of interviews and usability tests with the intended end user and one iteration of usability tests with random users. The last day of development was the 30th of April.

4.2 Meetings

Creating a meeting schedule (Table 4.2) for Scrum was simple. During a Sprint, we had four different types of meetings. The meetings were our Daily Scrum meetings when everyone had shown up at the start of the day. At the end of every Sprint, we had our Sprint meeting part A on Thursdays at 14.00 with only the group members. Right after, we met up with our supervisor and product owner at 15.15 for the Sprint meeting part B. The other weeks that were not the end of a Sprint, we had a regular supervisor meeting with our supervisor and product owner on Thursdays at 15.15.

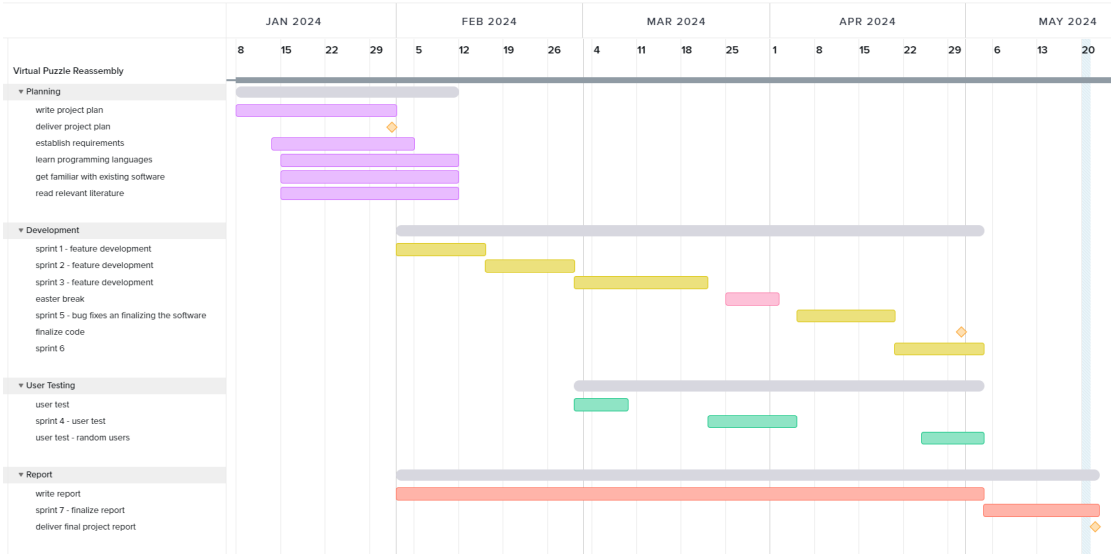


Figure 4.2: Gantt chart of the development process

Table 4.2: The meeting schedule that has been used during the project

Week	Monday	Tuesday	Wednesday	Thursday	Friday
Regular	Daily Scrum	Daily Scrum	Daily Scrum	Daily Scrum Supervisor Meeting	Daily Scrum
Sprint end	Daily Scrum	Daily Scrum	Daily Scrum	Daily Scrum Sprint Part A Sprint Part B	Daily Scrum

4.2.1 Daily scrum

The Daily Scrum is usually at the start of the day when all group members are present, usually between 08.00 and 09.00, and lasts about 10 to 15 minutes on average. The meeting location is where we usually would be working or another place that was decided beforehand, but if some or all group members cannot meet in person, we will have a meeting in a Discord call. The meeting starts with asking each member how they are feeling about their progress and in general. Then, the meeting transitions into the technical part of the meeting. First, each member goes through what they did the day before to catch the other members up on their progress, and if they managed to do what they planned. Then, we all discuss what we plan to work on for the day, such as which tasks we will do.

4.2.2 Sprint meeting

Since our Sprints are so much shorter and with fewer developers than usual in a typical project that uses Scrum, we decided to merge the three typical Scrum events, Sprint review, Sprint retrospective, and Sprint planning [7] into one Sprint meeting. Our supervisor and product owner do not have much technical programming knowledge, so we also split the Sprint meeting into two parts.

In the first part, the team members do the three merged meetings of technical discussion. Then, in the second part, the supervisor and product owner join us, and we discuss with them what we have discussed and decided on in the first part and get their input on each subject, whether they agree or not, and whether they might want to change something.

Sprint meeting part A

Sprint meeting A is only with the team members, and here we go over the three parts of a Sprint. First, we do a ten-minute Sprint Review to review the tasks and work we have done the last Sprint. Then we go to the next part, which is a 20-minute Sprint Retrospective. Here, we discuss how we felt the last Sprint went for each team member, like what went well, what problems were encountered, and how we may solve those. We discussed what we did in the last Sprint to improve our workflow and how the other team members could implement it into their workflow to increase their effectiveness. At the end, we have a 30-minute Sprint Planning. In the first ten minutes of the planning, we discuss why this next Sprint is valuable for the project and define a Sprint Goal that communicates that value. Then we discuss for 15 minutes what can be done in this next Sprint. We select items from the Product Backlog that we will work on next and transfer over any items from the last Sprint that we did not manage to finish in the allotted time and put them into the Sprint Backlog. Lastly, we have a five-minute discussion on how we will get the work done. This entails choosing which members will work on which item in the Sprint Backlog. [7]

Sprint meeting part B

The second part of the Sprint meeting takes place in person. Someone who cannot attend physically can attend over Teams. The first 25 minutes of the meeting begin

with us going over our work in the last Sprint and explaining it in detail. Here, we get feedback on the work from the product owner and our supervisor regarding whether or not something is up to par or could be changed for the better. We also discuss whether the proposed change is possible or not. When we are finished reviewing the last Sprint, we start discussing the plan for the next one in the last 20 minutes of the meeting. We use three minutes discussing why this next Sprint is valuable, to seven minutes of which Product Backlog items we are doing in this Sprint. We explain why we chose these items and can get feedback from the product owner and supervisor if they differ on which items we should prioritise. Then we spend five minutes going over how we will get the work done in the next Sprint and lastly, spend 5 minutes going over the changes to the Product Backlog.

4.2.3 Supervisor/Product owner meetings

This meeting takes place in the same place as Sprint meeting part B and happens when we are not at the end of a Sprint, which is most often every second week, except the once when we had a three week Sprint over the Easter break. During this meeting, we update our supervisor and the product owner on our progress in the Sprint. We take feedback on each item and discuss different options for creating and doing an item on the Sprint Backlog. We also got an opportunity to ask more general questions about the bachelor's thesis and get an opinion from the product owner and our supervisor.

4.3 Tools

4.3.1 Collaboration, Communication, and Documentation

- **GitHub** [30] is an internet hosting platform where the code for the project was pushed to using Git. Git is a version control tool that helps developers keep track of and manage changes to code files in a repository, by committing. This way, the developer can easily look back on the code versions if needed. A repository can have branches that keep track of parallel versions of the code that are not linearly connected to each other. The developer can then merge one branch into another to add the changes done in the code of that branch into the other. GitHub allows a developer to upload a local repository to a remote repository on the internet. Another can then download or "fork" a version of the remote repository, make changes, and push the changes up again. This allows developers to collaborate on a project.

  The GitHub repository for our project had a branch called "main", and a branch from "main" called "dev", and we created new branches for each new feature we developed (Appendix G), from "dev". These branches allowed a member to work on a feature from start to finish and use Git to merge the changes into the "dev" branch. This means there would be no interference between each member's work, and only small merge conflicts must be dealt with. We then merged the "dev" branch into "main" when we felt like there was a new testable version of the software. The branches that had nothing new compared to the main branch were deleted at the end of the project.

- **Discord** [31] is the main communication tool we use inside the group. We created our own server and a few text channels we would use. We had four channels, each with its own separate purpose. We had a channel for resource-sharing on relevant topics, a channel for notifying the other members about being sick, a channel for sending the time and place for a booked room we would use, and a general channel for everything else. We also had a voice and video channel where we would call each other to talk if we were not in the same place during a group meeting or in general.

- **Teams** [32] is the communication tool we used during meetings with our supervisor and product owner if one or more could not attend a meeting in person. It was also what we used to call the archaeologists for an interview or usability test.

- **Outlook** [33] is the mailing tool we use to communicate with our supervisor and product owner outside of meetings.

- **Overleaf** [34] is the online tool we used to write the project plan (Appendix C) and this report. It is a LaTeX editing environment using its typesetting system to create documents. It made collaborating on the writing easy with real-time updates between browsers. Overleaf has a template browser where we found the NTNU template used to write this report. Lastly, it has a great versioning control that we use to revert changes if needed or to review the history of a ".tex" document.

- **Excel** [35] is the software we used to document the work hours we put in during the Bachelor's thesis (Appendix H). We created an Excel sheet for each of the members where we could put in what we did during each week. For each week, we could select an activity and put in the amount of hours and minutes we worked on it. We created custom functions inside Excel so we could calculate the total time we spent on each activity. We used different methods to track the time spent on each activity during the day. We used a stopwatch or noted each activity's start and end time to calculate the time in hours and minutes later using Excel. We also used Excel for keeping our Product Backlog (Appendix E).

- **Word** [36] is what we used to write meeting minutes for each meeting (Appendix I), interviews (Appendix D), Sprint Backlog (Appendix F), usability tests template and tests (Appendices J K). We decided to use Word over Overleaf since it would be inefficient to create a new .tex document, format it correctly, and input it to the main file so it could be rendered.

- **OneDrive** [37] is the tool we used to store the documents generated through the project. With it, we could all edit the same document simultaneously, and it has file versioning for each file we could use to revert to if needed. With folders and sub-folders, keeping track of where each document is saved or needs to be saved was simple.

- **Lucidchart** [38] is a web-based application that allows users to collaborate, making diagrams visually. It is made by Lucid Software Inc. Lucidchart was used when making diagrams in this project. (Figures 3.5, 3.6, 5.1, 5.2, and

5.3)

- **TeamGantt** [39] is an online work planner and manager platform centred around a visual timeline. This tool was used to make the Gantt charts for the project. It was made for planning the project (Appendix C) and was modified during the process (Figure 4.2).

4.3.2 Development

- **Webstorm** [40] is the Integrated Development Environment (IDE) we primarily used during the project for developing the software. The IDE is where we edited the code, ran the software, and since it has built-in Git functionality, committed, pushed, and updated branches. It has an extensive extension library, which makes development easier, such as a tool to format JSON, a tool that makes parentheses ( ), square [ ], and curly { } brackets appear coloured to make it easier to see in which block you are typing in, a tool to help with using React, and a tool to help find and fix coding issues.
- **React** [41] is a front-end JavaScript library we used to build the software's GUI. It is what Casper [5, 42] used to start the development of the software, and we continued using it since it is a fast, declarative, component-based library that works well with other tools and libraries. We needed knowledge in HTML, CSS, and JavaScript to use it. Since it is component-based, creating new components to add to the software during development was easy [43].
- **Konva.js** [44] is the JavaScript library used to display and manipulate the fragments on a canvas. It has native integration with React and built-in filters for images on the canvas, which made it easier to develop features for the software.
- **Tauri** [45] is the tool we used to make the software into an executable application that the user can use. React is originally used to create websites, so Tauri makes the website into an executable software that starts a local web server to run the website on. The web server is contained in its own window and acts independently from other browsers on the computer.

CHAPTER 5

IMPLEMENTATION, CODING, PRODUCTION

The section describes the implementation of the tasks listed in the project's Product Backlog in Requirements. We will focus on the major decision and discussion points during the development.

5.1 Commenting, Renaming and Linting

One of the first things we discovered when starting the further development of the code of Artifact Assembly made by Casper was that there was not a lot of documentation of the code. Just one of the functions was documented, and none of the components. This was something that we improved and had a focus on since one of our goals was to make software that could be further developed (Section 1.2: Project Goals). To better explain their usage, we also improved some of the names of variables, functions and contexts.

Functional components, are a type of functions, and can therefore be written in the same ways as other functions. They can also be written in different ways, if one finds it necessary to differentiate them more. We did that by using const lambda declaration for functional components, and using the function declaration for other functions.

Example of a functional component:

Code 5.1: Functional Component example

```
const StageArea = () => {...}
```

Example of a function:

Code 5.2: Function example

```
function handleFilters() {...}
```

5.2 Technical Design

When we decided to continue developing the software, we also continued using the original technology stack (Section 3.3.1: Language and Framework).

5.2.1 Components

We kept all of the components in the software we started with (Figure 3.5), but all of them were changed in some way. Some components were also moved or renamed (Figure 5.1).

5.2.2 Contexts

Each of the contexts was changed from the initial version so that the providers for the contexts were all encapsulated into their respective component holding the data and the functions the context provides. This improves the code's adherence to the single responsibility principle, making it easier to develop and maintain. A simple example of such component is the LockedContextProvider that encapsulates the provider for the locked context and holds the provided value, isLocked and its setter function.

Code 5.3: ContextProvider example

```
1  /**
2   * Provider for the locked context that allows for getting
       and setting locking the stage.
3   * @param children {JSX.Element} the components that can
       use the context.
4   * @return {JSX.Element} the context provider.
5   * @constructor
6   */
7  export const LockedContextProvider = ({children}) => {
8      const [isLocked, setIsLocked] = useState(false);
9
10     return (
11         <LockedContext.Provider value={{isLocked,
               setIsLocked}}>
12             {children}
13         </LockedContext.Provider>
14     )
15 }
```

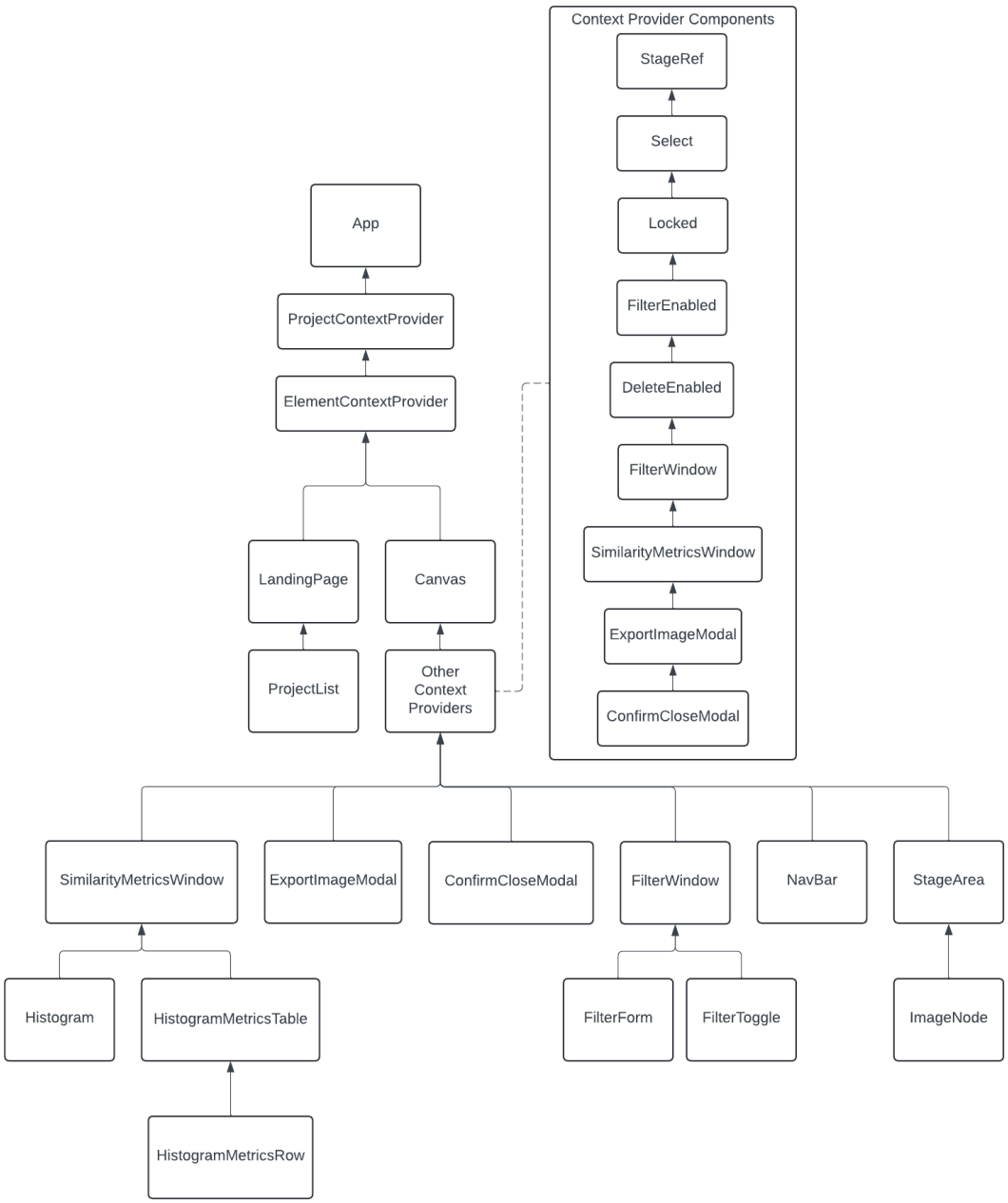


Figure 5.1: The React component tree of the final application

What happened to the old context provider values

The original version had contexts that provided the following values with their setter functions: “grid”, “resize”, “lock”, “filter”, “images”, and filter values of “saturation”, “hue”, “contrast” and “luminance” (Section 3.3.4: Contexts). The "grid" value and the code connected to it were removed after feedback from the user. In the first interview with the user (Section 6.1: Interview 1), she said that the ability to change the size of the images was unwanted since they were already in the correct scale. Adding this feature would only cause mistakes. After discussing it with the product owner, we removed "resize" completely. The "lock" and "filter", renamed to isLocked and filterEnabled, were both kept in the final software. The "images" array was renamed to "elements" and put in the ElementContext to represent an array of all possible elements on the canvas by describing their props. However, the code currently only contains one type of element, images. We wanted to prepare the software for implementing other types, like the group type that could represent a group of grouped elements, as this was the only feature we took on but could not finish, which we discuss more in future work (Section 10.4: Refactor and Grouping Images Together). One of the features that the user wanted was the ability to have different filters on different images (Section 6.1: Interview 1). To achieve this, we needed to move the filter values from a common scope to an individual scope. Therefore, the values were set as attributes of the individual image state prop objects in "elements".

Context Providers

When making windows and modals, they each needed a boolean value and a setter function to tell if it is open and showing. This needed to be provided through a context. First, we made one context to hold all of them and equip the possibility to open them from anywhere on the canvas page. But then we separated them into single responsibility contexts for each window or modal and placed them in the same JSX file as the component for the window or modal itself. The argument for this was that it would be easier to keep track of what value and setter function were connected to which component.

This idea of separating context based on responsibility rather than scope is what we ended up with for all our contexts. Resulting in many nested context providers like this in Canvas.jsx:

Code 5.4: Canvas.jsx nested ContextProviders

```
1  /**
2   * Component that represents the canvas page.
3   * @returns {JSX.Element} the canvas page.
4   * @constructor
5   */
6  const Canvas = () => {
7    return (
8      <StageRefContextProvider>
9        <SelectContextProvider>
10         <LockedContextProvider>
11           <FilterEnabledContextProvider>
```

```
12                <DeleteEnabledContextProvider>
13                  <FilterWindowContextProvider>
14                    <SimilarityMetricsWindowContextProvider>
15                      <ExportImageModalContextProvider>
16                        <ConfirmCloseModalContextProvider>
17                          <div className="stage-container">
18                            <NavBar/>
19                            <StageArea/>
20                            <SimilarityMetricsWindow/>
21                            <ExportImageModal/>
22                            <FilterWindow/>
23                            <ConfirmCloseModal/>
24                          </div>
25                        </ConfirmCloseModalContextProvider>
26                      </ExportImageModalContextProvider>
27                    </SimilarityMetricsWindowContextProvider>
28                  </FilterWindowContextProvider>
29                </DeleteEnabledContextProvider>
30              </FilterEnabledContextProvider>
31            </LockedContextProvider>
32          </SelectContextProvider>
33        </StageRefContextProvider>
34      );
35  };
```

We discussed the option of only having two context providers. Having one at the App component level and the other at the Canvas component level would be enough, but we believe this would result in huge files, making the code harder to maintain and develop.

5.2.3 useHistory

We made a custom hook to abstract away the logic used for undo and redo (Task 4e p.25). This hook wraps the elements list into another list representing the history of the canvas states. By default, using the setter function of the elements makes a new commit and adds a new item to the history. One could also use the setter function for elements and overwrite the previous commit item in the history. The hook also has a useState that holds an index pointing to the current element item in this list. This index is then decremented or incremented when undo or redo is called.

After making the useHistory hook, we wanted functions for undo and redo to even work with the changes done when changing the filter values, specifically when using the sliders. The implementation of the undo/redo history was that one could either save the images as a new state commit in the history or overwrite the last commit. But in the case of the filter sliders, we wanted the changes to be committed when the mouse last released the slider but still used the setter function and updated every time every slider changed to update the visual of the filtered image. For example, moving the slider from value 0 to 5 would result in five new

commits to the history, one for each new value increment. This would result in an impractical undo and redo. We experimented with having an explicit commit function in useHistory to make a new state in history. While implementing this we uncovered this would result in a lot of refactoring and even undermine the idea behind the useHistory hook to seamlessly update the history upon using its setter function. The solution we ended up with was making a new commit when first clicking down with the mouse on the slider and then overwriting this commit on every change until the mouse releases. This way, there will be only one commit for every slider click and not every slider change. For example, when the slider moved from value 0 to 5, there would be one new commit to the history with value 0 that would then be overwritten for each value increment, resulting in the commit with the value being 5.

5.2.4 Select Multiple

A feature the user wanted was the ability to select multiple images to move, rotate, delete, or modify all their filters together (Task 1a p.23). Since the logic around selection was a little complex, a flowchart was made to sort out how to implement it (Figure 5.2).

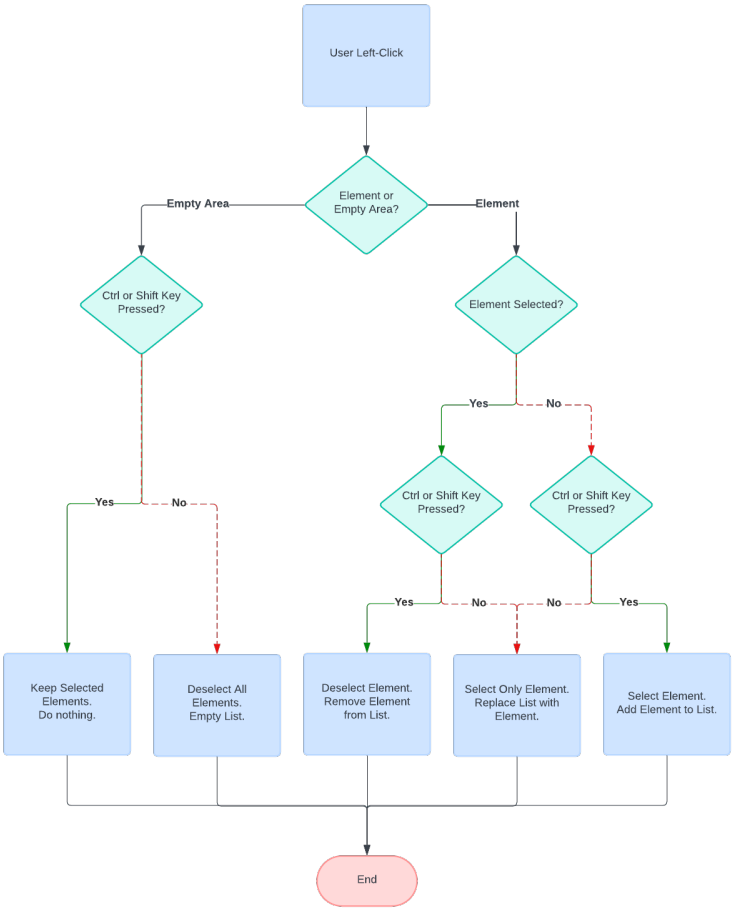


Figure 5.2: Flowchart explaining the selection logic

We then used the flowchart to make two on-click event listeners, handleDeselect

and handleElementClick, that implement the logic for if a vacant area on the canvas is clicked and if an element is clicked, respectively.

The Konva.js transformer box used for the selected image was moved out of the ImageNode to the StageArea component when adding the feature of selecting multiple images. The SelectContext were also made to enable this.

Code 5.5: handleDeselect function

```
/**
 * Deselects when the mouse left-clicks on an empty area
 * on the canvas and ctrl key is not pressed.
 * @param e{KonvaEventObject<MouseEvent>} the event.
 */
function handleDeselect (e){
    if (e.target === e.currentTarget &&
    e.evt.button !== 2 && !ctrlPressed && !shiftPressed) {
        deselectAll();
    }
};
```

Code 5.6: handleElementClick function

```
/**
 * Event handler for element clicking. This will check the
    selection of the element.
 * @param e {KonvaEventObject<MouseEvent>} click event.
 * @param index {number} of the element clicked on.
 */
function handleElementClick(e, index) {
    if (e.evt.button === 2) return;
    const element = e.target;

    if (ctrlPressed || shiftPressed) {
        if (isSelected(index)) {
            // already selected
            deselect(index);
        } else {
            // not already selected
            select(element, index);
        }
    } else {
        selectOnly(element, index);
    }
}
```

Filter Selected

After implementing the functionality to select multiple images, we wanted the ability to change the filters on all the selected images and not only on the one

that was chosen with right-click (Task 5b p.26). When we started to implement this, the code was made to iterate through all the selected images and add the new filter value from user input to the filter value each image had before it was selected. The selected images could therefore end up having different filter values from each other, based on what their filter values originally were. If the value went out of our defined range, the value was set to a minimum or maximum valid value.

After implementing it, we discovered that the feature was not intuitive, especially when the value went out of range. We therefore went back to only applying filters to the right-clicked image.

Later on, another idea emerged: to display the value 0 or false when the filter values of the selected images were different and overwrite the previous value completely when changing the values. This made it possible to implement what we wanted earlier: to be able to filter multiple images at once. We gathered that this was more intuitive, as the images filtered at the same time, would have the same values for the filters that had been changed.

5.2.5 Grouping Images Together

We took on the task of grouping images (Task 3a p.24) in Sprint 2 that we began working on in Sprint 3 but did not manage to finish. The task consisted of combining images into a unit that could be manipulated on the canvas as one element. We needed to make structural changes to the code to support the implementation of this task. But these were larger changes than we first thought, and we did not manage to complete them. Consequently, it was left as a feature for future development (Section 10.4: Refactor and Grouping Images Together).

Abstract Component

In Sprint 3, we tried to restructure the code to allow for future features like having other elements than images on the canvas, such as text notes (Section 10.7: Text Boxes or Notes) or groups (Task 3a p.24). To do this, we tried making a polymorphic abstract type of functional component that could represent a generic element on the canvas. Other components like image, note or group could then inherit from this generic component. These ideas are found in object-oriented programming, which is what we are familiar with from before. But this idea is not supported by the React way of thinking. Components do not support inheritance and rather use composition. We still tried to make a polymorphic ElementNode to wrap the different components into. While working on it, we discovered that the only thing common for these components was the ability to be selected, i.e. placed into a transformer and made draggable, and to be grouped. We thought this functionality did not make sense to refactor into a generic component. We, therefore, went back and just moved the common functionality out of ImageNode into StageArea.

GroupNode

We then began creating a component to represent a group of elements. However, this approach was abandoned due to loss of manageability over the code, increased complexity, and bugs. Despite these issues, we believe it could be re-implemented in the future to support the feature of grouping images together (Section 10.4: Refactor and Grouping Images Together).

Refactoring

Near the end of the development, in Sprint 6, we still had not implemented the grouping images task. Therefore an opinion of refactoring the code emerged (Task 6b p.26).

The goal of the refactoring was to initialise the Konva.js stage as early as possible so that we also could initialise Konva.js image objects when loading in images. We could then use the Konva.js objects instead of the element prop objects that we are using in the original version of the software. Using more of the Konva.js library and its functions, we would have more control over the stage, its layers and all the objects in the stage. Having this direct control over them would allow us to finally implement the task, grouping elements (task 3a p.24) using a Konva.js group.

We tried to do this by expanding the StageRefContext, later renamed to StageContext. We added functions to fetch specific children of the Konva.js stage and to include the functionality of several contexts, including ElementContext, LockedContext and SelectContext. The resulting component tree is Figure 5.3.

Some examples of the functions for fetching specific children are getting all elements, groups or all the selected elements. Later, these functions were changed to variables called consts.

The functionality of the ElementContext was keeping track of the state history. We attempted to implement this seamlessly into the StageContext, but the implementation was not bug-free.

The functionality of SelectContext was moved to StageContext because StageContext could now define the selected elements as children of a layer in the Konva.js stage rather than a separate list of indices pointing to elements in another list. By having two layers, we also optimised the code by separating the elements that changed from the non-changing ones into the different layers. This resulted in not redrawing all the static, non-changing elements every time changes happened.

This was a bigger job than expected. Given the limited time allocated to this thesis work, we had to set strict deadlines to manage the time. If the feature wasn't finished within the allocated time, we had to leave it for future work. After hard working days, we still could not finish by the deadline, so the refactoring was not implemented in the final version. We still wanted to pass on the work we did with this refactoring, so we made a separate Git branch that future developers could use as a reference or continue working on and finish the refactoring (Section 10.4: Refactor and Grouping Images Together).

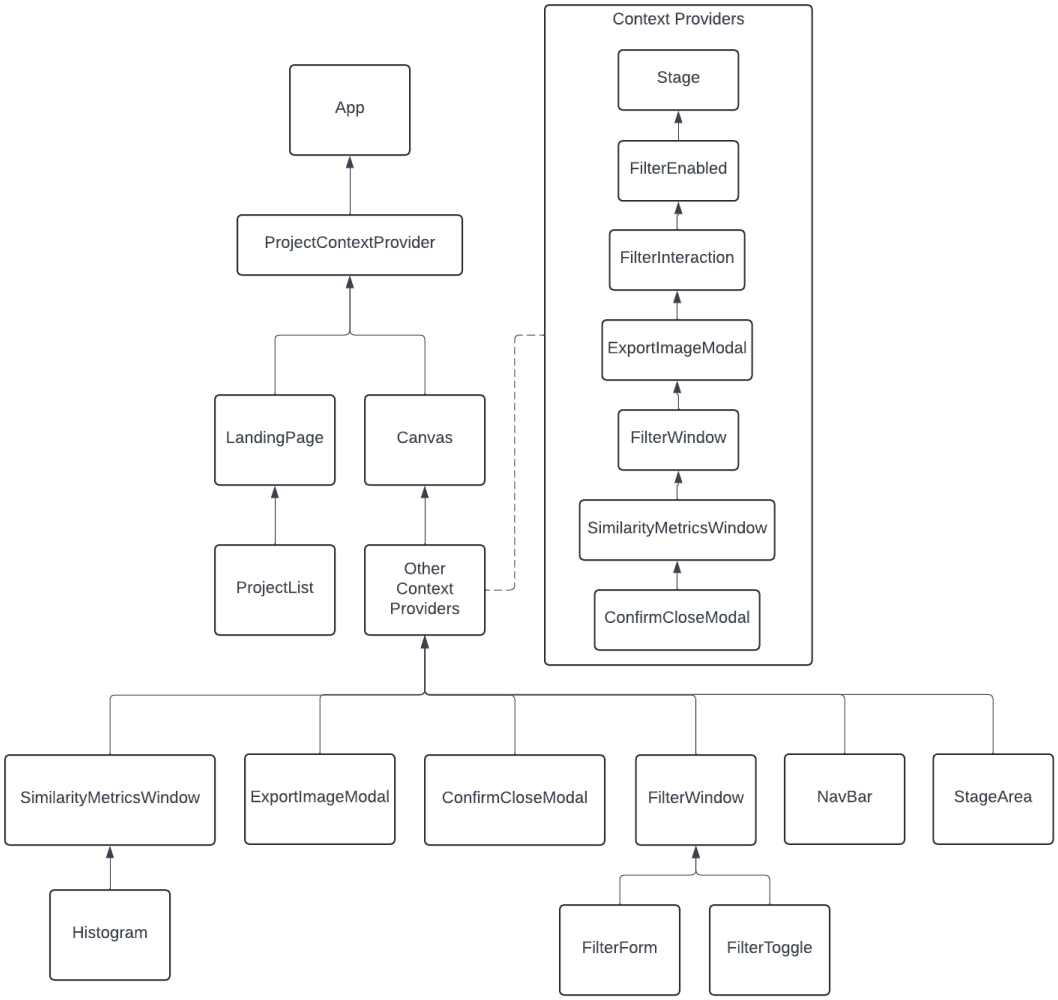


Figure 5.3: The React component tree of the refactored version

Still, we implemented some of the ideas that emerged from the refactoring. Mainly the filtration of the images from elements and if there are any checks, both in the ElementContext and the SelectedContext.

The refactoring was not completed, hence it was left for future work to focus on improving other parts of the code. This left time to incorporate concepts and ideas explored in the refactoring into the final version of the software. Therefore, we concluded that this was the right decision.

5.2.6 Windows

We created two windows positioned on top of the canvas page: the SimilarityMetricsWindow (Task 2a p.24) and the FilterWindow (Task 4h p.25). We could not have the windows be static on the screen as it could cover what is on the canvas, so we made a function so the windows could be moved by dragging from the top. This works by setting the position of the window div to a new position relative to the change in position of the mouse when it has clicked and holds the top of the window. To prevent them from being dragged off-screen, it checks if the boundary of the window is about to be moved off-screen and then sets its position at the limit so it does not disappear. Both windows have this functionality, but the SimilarityMetricsWindow also needed to be able to resize. This function works similarly but instead resizes the window when the mouse drags on the edges of the window.

SimilarityMetricsWindow

The window displays a histogram and a similarity score table for each selected image (Task 2a and 4i pp.24-25). Each selected image's table will contain similarity scores between the image and every other image on the screen. The Histogram component represents the image's histogram bar graph (Task 4c p.24), while the HistogramMetricsTable shows the hue comparison metrics in a table format, with each comparison in a HistogramMetricsRow (Task 6a p.26).

The similarity metrics (Task 4b p.24) we chose, which we will show later in the chapter, are Euclidean Distance, Bhattacharyya Distance, and Histogram Intersection [46–48]. They are calculated based on the histogram values. First, we do some calculations per histogram bar. Then, we finalise the value by doing a last calculation.

For every value in histogram 1 (x) and every value in histogram 2 (y), we do the calculations where n is the size of the histogram:

$$\textbf{euclidean} = \sum_{i=1}^{n}(x_i - y_i)^2$$

$$\textbf{bhattacharyya} = \sum_{i=1}^{n}\sqrt{x_i \cdot y_i}$$

$$\textbf{intersection} = \sum_{i=1}^{n}\min(x_i, y_i)$$

Then, the final calculations:

$$\textbf{Euclidean Distance} = \sqrt{euclidean}$$

$$\textbf{Bhattacharyya Distance} = -\log(bhattacharyya)$$

$$\textbf{Histogram Intersection} = (1 - intersection)$$

To combine the scores, we add them all together and divide them by 3. This makes it possible to add weight to each score if necessary. We chose these metrics because, among the different metrics we found, these three all approach 0, the more similar the histograms are.

FilterWindow

This is the window for changing the filters on an image (Task 4h p.25). The FilterForm component is moved from the NavBar component to this window. FilterForm is changed to include two more parts to improve usability (Task 4a p.24). It has a slider to improve how intuitive changing the filter is and a button to reset it back to its default value. In addition to changing the FilterForm, we also made a FilterToggle component for filters with only an on-and-off state.

5.2.7 Reduction of Complexity

When the similarity metrics window was first created, the function for getting the hue value array used for the histogram looked like this:

Code 5.7: Complex updateHistograms function

```
1  async function updateHistograms() {
2      const imageNodes = stageRef.current.getChildren()[0].
           getChildren().filter((child) => child.
3      getClassName() === 'Image')
4      for (const index of selectedElementsIndex) {
5          for (const imageNode of imageNodes) {
6              if (elements[index].id === imageNode.
7              attrs.id) {
8                  const newHues = await getHueData(imageNode.
                       toDataURL());
9                  elements[index] = {
10                     ...elements[index],
11                     hueValues: newHues,
12                 }
13             }
14         }
15     }
16     ...
17 }
```

By changing what object holds the hue value array, the code was reduced in complexity to this:

Code 5.8: updateHistograms function reduced complexity

```
1  async function updateHistograms() {
2      const imageNodes = stageRef.current.find((node) =>
```

```
3       node.getClassName() === "Image");
4       for (const imageNode of imageNodes) {
5           imageNode.attrs.hueValues = await getHueData(
                imageNode.toDataURL());
6       }
7       ...
8   }
```

This is one example of how we developed. We first created a functional version that implemented the feature we wanted. We then refactored the code to improve its quality and make it more readable and robust, facilitating further development in the future.

5.2.8 Confirm Close modal

To improve error prevention in the software, we implemented a modal to confirm if users wanted to save the project before leaving or closing it (Task 4f p.25). This would prevent the user from losing unsaved work. When we implemented the modal, we wanted it to pop up when returning to the landing page and closing the software from the canvas page. However, we could only display it when returning to the landing page since we did not find a way to call JavaScript functions from the Rust back-end. We will discuss this more in future work (Section 10.17: Confirm Save on close).

5.2.9 Find Work Area

A problem with the canvas when zooming out and moving is that it makes losing the images easy, as they are small. To counter this problem, we made a function to find the closest image on the canvas and move to it (Task 4g p.25). It works by calculating the position coordinates of the centre of the current stage. Then, it goes through all the images on the canvas to find the one closest to the current position using the Pythagorean Theorem. First, calculating the difference in x and y coordinates between the current position and the image,

$$dx = elementPosX - currentStageCenterX$$

$$dy = elementPosY - currentStageCenterY$$

Then, it calculates the hypotenuse for the image.

$$distance = \sqrt{dx^2 + dy^2}$$

When the distance of every image has been calculated, it moves the current stage to the position of the image with the lowest distance.

5.2.10 Filter Enabled

In the version we got from Casper, a button in the navigation bar on the canvas page switched on or off the filters on the images (Section 3.2.3: Filter). This feature was kept, but the button was moved to the filter window when the window was created. From the heuristic evaluation, we got the feedback that the filters should be enabled by default (Section: 7.2: Heuristic Evaluation). We discovered later that having the filter enabled by default made the loading of the images slow due to Random Access Memory (RAM) limitations. We will discuss a solution to this in future work (Section 10.1: Memory Problem Solutions:
Rewrite or Change Environment). So, in the end, we ended up having the filters disabled by default and adding the button back to the navigation bar, this time under the tools drop-down menu.

5.2.11 Project File

The most important feature discussed in the first interview with the user was the ability to save a project (Section 6.1: Interview 1). To achieve this, we needed to define a file that could contain the needs defined by the requirements (Task 1b p.23). We decided to utilise the file type JSON (JavaScript Object Notation). We chose it because it is widely used, human-readable, lightweight to represent data structures and lists, and well integrated with JavaScript.

5.3 GUI

The software's design was discussed early on when we started working on it. We could either keep the general design and look of the software as it was when we inherited it or change it.

5.3.1 Landing Page Changes

Based on the inherited GUI, we made the GUI more user-friendly (Section 3.2: GUI). During development, we encountered no situation that would require us to drastically change the look of the GUI, so we kept the original look of the software so that the user would still feel familiar with the software.

In the new landing page version, the user will have more options to choose from (Figure 5.4). Instead of only having the option to open the canvas and quit, the user can choose between starting a new project, opening an existing project, or quitting.

Opening a project will open a file explorer dialogue window provided by Tauri where the user can choose a project file (Task 1c p.24).

We discussed whether or not to keep the Projects tab on the landing page since it did not serve a purpose. However, since a part of the bachelor's was to develop and implement a way to save projects to a file, we decided to keep the tab. We did not have time to incorporate using the projects tab, so we made no changes there. We discuss this more in future work (Section 10.11: Saved Project List).

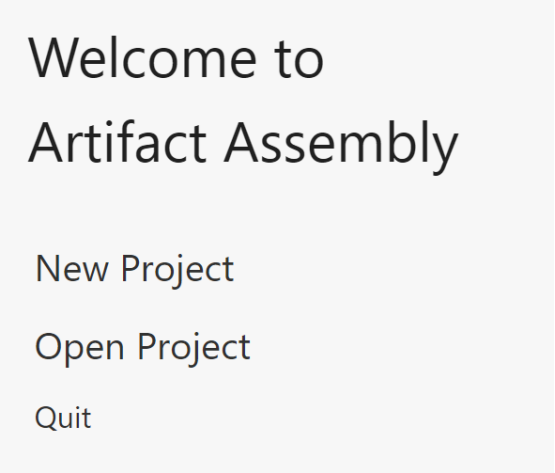


Figure 5.4: Landing page with the new option to "Open Project"

5.3.2 Navigation Bar

In the new GUI, the same canvas as in the old version is opened when starting a new project or opening one (Section 3.2: GUI). However, the user will notice the main difference is the new navigation bar at the top of the screen (Figure 5.5).

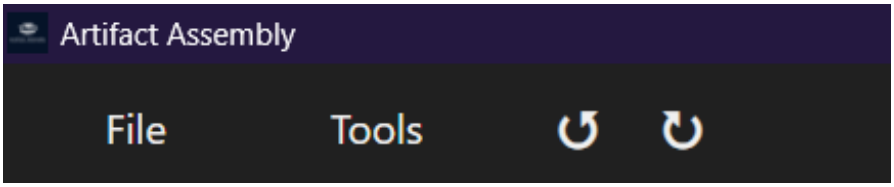
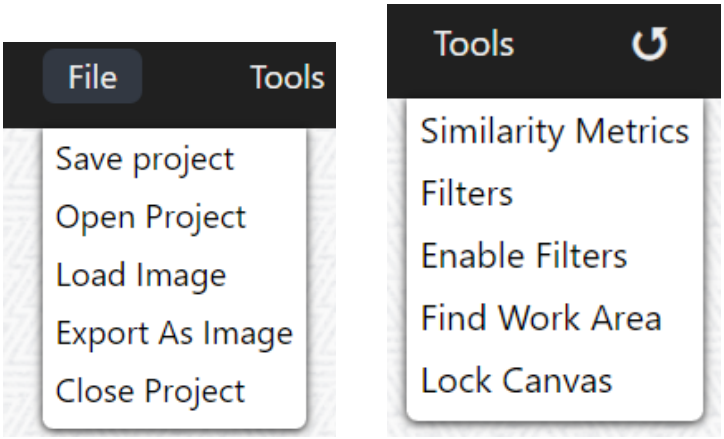


Figure 5.5: The new layout of the navigation bar

The software's new version has many more features than the original, so the navigation bar could not contain everything we wanted to put there. So, to future-proof, we added drop-down menus to the software that would make the navigation bar less cluttered (Figure 5.6). We moved the Home button from the original navigation bar to the File drop-down menu (Figure 5.6a) and renamed it to "Close Project" because it was not used often, and we did not want it to be pressed accidentally as pointed out during the heuristic evaluation (Section 7.2: Heuristic Evaluation).

The menus are designed to be dynamic and easy to add new items to, so for a new menu item with longer text, the menu will expand to fit the new text. Initially, the menus were centred under their respective button, but we encountered a problem with how the file menu expanded horizontally off the left side of the screen, so we changed it so the menus were aligned with the left side of their respective buttons.

The last thing on the navigation bar is the undo and redo buttons. The undo feature existed as a keyboard shortcut before we took over. Taking inspiration from other software, such as Microsoft Office, we added an undo button and a redo button when that was implemented. The keyboard shortcuts for both are still working, but since not everyone might know about common keyboard shortcuts,



(a) File drop-down menu **(b)** Tools drop-down menu

Figure 5.6: New navigation bar drop-down menus

we decided to add buttons for those who do not know (Task 4e p.25).

The undo and redo buttons have undergone a few changes. (Figure 5.7). When initially implemented, they started as text on the buttons (Figure 5.7a). We felt that the text represented the feature well enough at first. But then we got some feedback from a UX-designer that this conflicted with the heuristic of "Recognition rather than recall" and should rather use arrow icons to represent than by text (Section 7.2: Heuristic Evaluation). This is also seen in other software with undo and redo buttons. We prepared to ask what the user would prefer during the second usability test (Section 7.1.2: Usability test 2). Before we could ask, the archaeologist commented on it herself, so we later sent her some options of arrows they could choose between (Figure 5.7b) since we had differing opinions in the group. They preferred the first one of the examples, so that is what we went with (Figure 5.7c)(Task 5c p.26).

(a) Initial Undo/Redo **(b)** Undo/Redo options

(c) Final Undo/Redo decision

Figure 5.7: The design progress of the undo/redo buttons

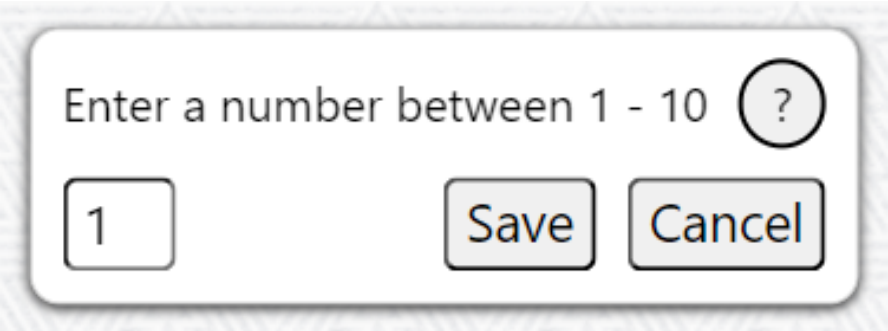
5.3.3 Export as Image

A new feature with a few revisions is the Export as Image located in the File drop-down menu (Figure 5.6a)(Task 1d p.24). It started as a button with a number input attached (Figure 5.8). The number input had two different looks during this

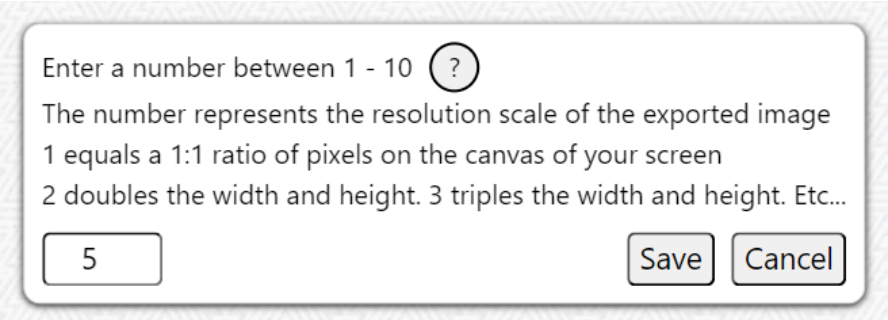
version: the version with a percentage and the version with a multiplier from 1-10. The number indicated the scaling of the resulting image that was downloaded when pressed, but we got feedback about the button during the first usability test (Section 7.1.1). The archaeologist said that the percentage was not easy to understand. The starting value was 100%, so without knowing that it went from 100% to 1000%, she tried to go lower than 100%. The exported image was also directly downloaded to the downloads folder, and the user could not set the file's name themselves, which she said would be preferable.

Figure 5.8: The first design of the export as image button that took in a percentage on the button

Therefore, after the usability test, we changed how the button works. We removed the number input from the button, and when the button is pressed, a modal is opened in the middle of the screen (Figure 5.9)(Task 4j p.25). The modal works similarly, but we indicate that the user should put in a number between one and ten (Figure 5.9a). To inform the user what the number means, they can press the (?) button to display or hide text that explains what the number does. Pressing the save button will open a file explorer window so the user can indicate where they want to save the image and what to name the file.



(a) Export modal - normal



(b) Export modal - informative

Figure 5.9: Export modal window that is opened after pressing the Export as Image button in the File menu

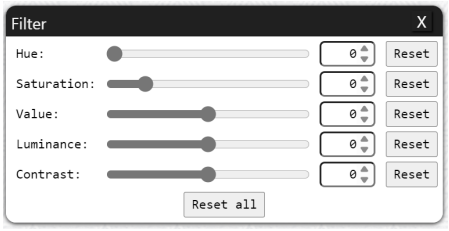
5.3.4 Filters

The already existing feature that we changed the most is the filters. During the first interview with one of the archaeologists, we learned that only the number input was not intuitive for them (Section 6.1: Interview 1). So, to make it more intuitive, we added sliders to each filter (Figure 5.10)(Task 4a p.24). The hue value would represent the degree of hue from the HSV colour space and not rotate between hues. The sliders update the number-input fields and vice versa. We thought the slider would be a better visual representation of the number ranges and more interactive than pressing a button to increase/decrease the number or to delete and write in numbers manually.

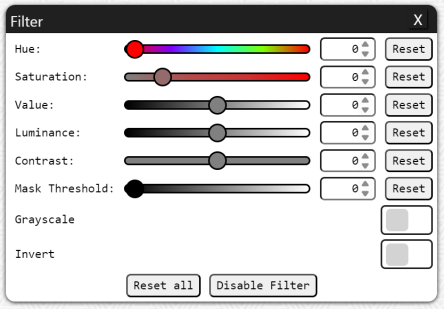
Figure 5.10: First revision of filters. Filters also have a slider

Problems arose when we wanted to add more filters (Task 4d p.25). When enabling the filters, there was insufficient horizontal space to fit all the filters while keeping the sliders at a usable length and not too small. We discussed an option to make the navigation bar taller to fit more filters under the others but decided not to since it would take vertical space away from the canvas. Therefore, we decided to create a window to hold the filters (Figure 5.11a)(Task 4h p.25). While moving the filters, we also added a reset button for each filter to give the user more control and not have to restart every time a user pressed the button.

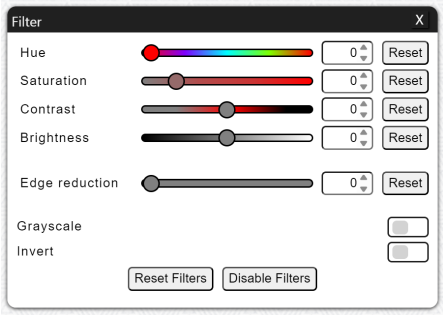
Before the second usability test, we wanted to add design changes to make the filter window more intuitive and user-friendly. We added colours to the sliders to make the slider thumb and background colour match the changes made to the image (Figure 5.11b). Both HSV and HSL colour spaces were used, but they had different max values for hue, so switching between Value and Luminance changed the colour of the fragments. Therefore, we asked during the usability test which they preferred. She preferred Value, so Luminance was removed, but she did not like the word Value. The word's meaning was unclear to users unfamiliar with the HSV colour space. She would much rather it be named Brightness so that she could understand what it did better. During the usability test, we also got other feedback from the archaeologist about some of the sliders that, for her, were not as she expected. We discussed Contrast since we could not find a way to represent it in a good way before the test, and we came up with an idea together. The last discussion was about the Mask Threshold, which did not make sense to her. It was removing the darkest parts of the image and turning it transparent. She understood that is what the filter did but not the correlation between the label and the filter, so we discussed what would make more sense and decided on calling it Edge reduction (Section 7.1.2: Usability test 2). After the usability test, we made the changes to the filter windows as discussed, swapped the position of the Contrast and Brightness filters, and added separation between some of the filters to indicate which filtered similarly to each other to make the finished filter window (Figure 5.11c)(Task 5a p.25).



(a) Filters in its own window



(b) Slider colours and new options



(c) Finished filter window

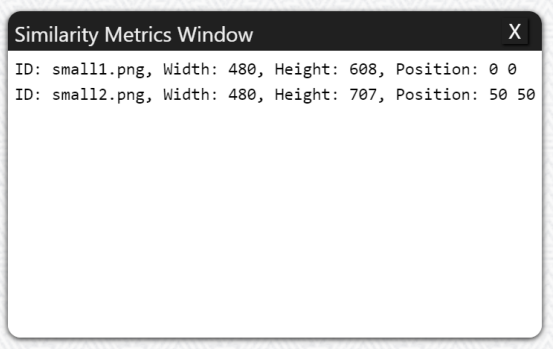
Figure 5.11: The two main updates to the filter window. When we first moved it, and when we added colours and new options

5.3.5 Similarity Metrics

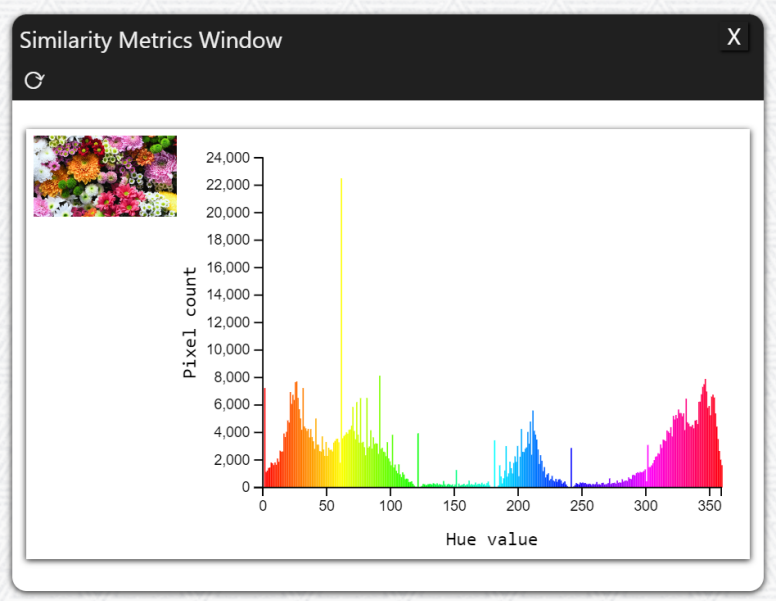
The Similarity Metrics underwent multiple testing iterations during development (Task 2a p.24). At first, it was just a window with some text (Figure 5.12a). That is because setting data to an image prop when the image is loaded into the software and retrieving it after it was set was challenging to figure out. When we figured out the data extraction, we needed to find the best way to show the similarity between the images. We had several discussions but decided on using the Hue channel in the HSV colour space. The reason we chose Hue is because it represents the colour of an image. It groups meaningful regions on an image better than saturation and value (brightness) and can better indicate similarity between images.

Multiple ways existed to create a histogram (Task 4c p.24). We tried building our own histogram component and using the plotly.js library, but we used the d3.js library as it best fit our needs. It takes in the hue values from the image and displays them in the histogram (Figure 5.12b). The window is not limited to one image at a time but is linked to the images the user selects.

The histogram is an Scalable Vector Graphics (SVG) image that is generated when an image is selected by taking in the hue values. We wanted to give the user the ability to view the hue histogram of a filtered image, but we did not want it to make a new SVG image every time a filter is changed, so we created the update button to give the user the choice of when to recalculate the hue values and update the histogram of the filtered image. It would also change the look of the displayed image on the left so the user can recognise it more easily if there are



(a) Information extraction test



(b) Hue histogram display test

Figure 5.12: Testing information gathering and display

duplicates on the canvas.

(a) Non-filtered image of a fragment **(b)** Filtered image of a fragment

Figure 5.13: Histogram changes when applying filter to an image of a fragment

The issue with displaying the whole range of hue values that range from 0 to 360 is that the hue value range is small for some images. This shows up as a spike on the histogram, which does not relay much information to the user. During a meeting with our supervisor and the product owner, we discussed whether we should keep it as is or implement a way to reduce the range of values the histogram shows. We decided to have a way for the user to manually limit the range, which would be the best option to "Zoom in" onto specific ranges of the 0-360 range (Figure 5.14).

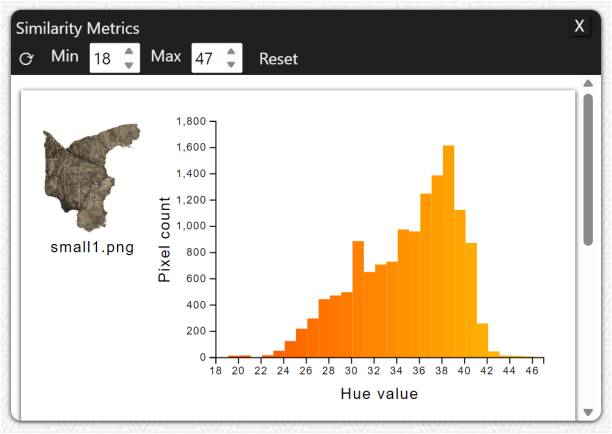


Figure 5.14: The histogram's min and max values have been manually reduced to 18-47 to show the relevant hue range values better than the original (Figure 5.13a)

The last thing needed for the window was to create a similarity metric and calculate the similarity scores between an image and every other image (Task 4b p.24). We found the best way to display these scores was in a table (Figure 5.15)(Task 6a p.26). More images on the canvas will create a table with more rows. Then, the image with the lowest combined score will be displayed as the most similar to the selected image above the table. It was originally on the bottom, but after the puzzle usability test with many table rows and lots of time spent scrolling, we moved it to be above it to minimise scrolling time. The scores are automatically sorted in ascending order by the combined score but can be sorted in

ascending order by pressing the name of each metric. We display all the different scores to give the user more options because one metric might be better than the others in specific situations. These displayed images will also change colour when their corresponding image is filtered. This will help the archaeologists find the most similar fragment and solve the puzzle.

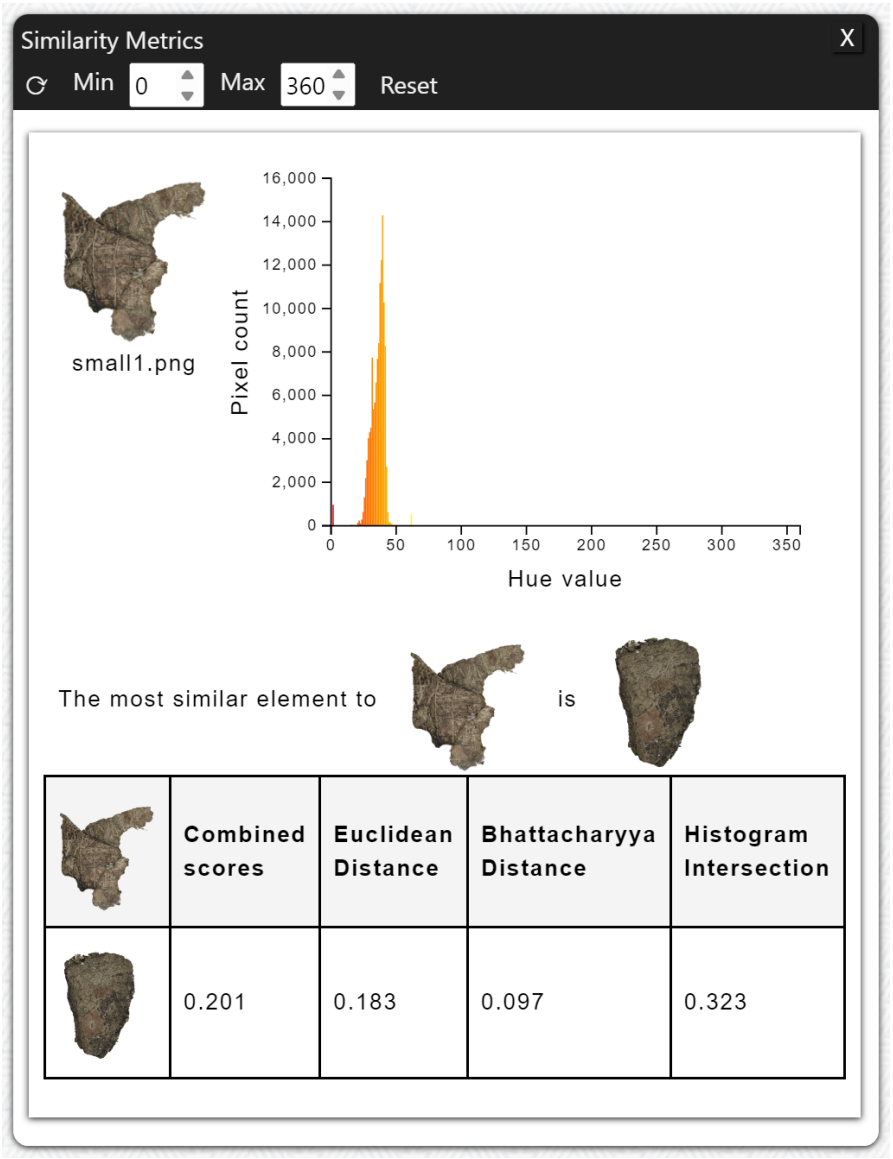


Figure 5.15: Similarity metrics window with a table that calculates the similarity between two images. Lower is better.

5.3.6 Save on close

Under the File drop-down menu (Figure 5.6a) is the option to "Close Project". When pressed, it would originally close the project and return to the landing page, and it was moved there to minimise accidental clicks, but we wanted to further minimise the risks of deleting unsaved work. To do that, we made another modal for when the user clicks the button, either on purpose or not (Figure 5.16)(4f p.25). It will double-check if the user wants to save their changes or cancel the

close operation. The save option will open a file explorer save window, just as when pressing the "Save Project" button or exporting an image of the canvas.

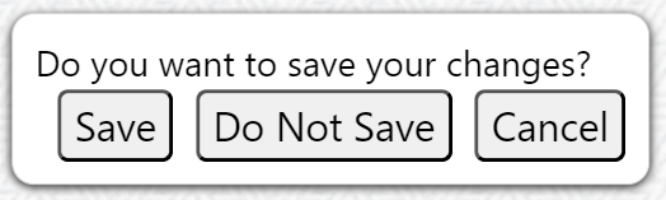


Figure 5.16: The modal that shows up when pressing the close project button

5.3.7 Multiple selected

A functional change from the original version is that the user can select multiple images instead of just one (Figure 5.17)(Task 1a p.23). This allowed us to add more functionality to the software, such as filtering multiple images simultaneously. It will create a Transformer box around all the selected images to visualise which are selected.

Figure 5.17: Multiple images selected at the same time

5.3.8 Loading in multiple images

A last feature implemented is that now when the user loads several images at the same time in a project, they will not be stacked on top of each other. A little offset, measured by the location where the last image was put, was implemented to decide the spawning location for the next image. This way, it is easier for the user to see and control all the images loaded in.

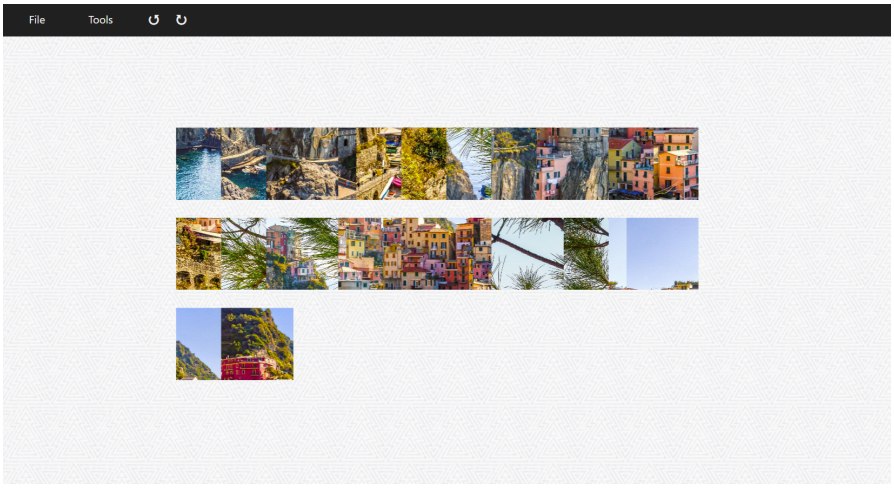


Figure 5.18: Multiple images loaded in with an offset

CHAPTER 6

INTERVIEWS

The understanding phase is the first phase of the design process [15]. In this phase, one wants to develop an understanding of what the needs of the users are. In this case, there was a very specific group that was intended to be the end users, the archaeologists at KHM, and therefore it was natural to interview them. The results from the interviews we hosted gave us many ideas about what could be done to the software (Appendix D). This became the base of how we structured the bachelor's thesis.

The team conducted two semi-structured interviews (Appendix D), following an incomplete script with discussion topics. This left room for follow-up questions for deeper understanding. The loss of consistency was not an issue since the interviews were only done once each. The first interview was remote over Teams for practical reasons, and the other was done on-site with the archaeologists in Oslo, where they worked.

6.1 Interview 1

At the first meeting with the archaeologist, the group conducted an interview, asking about what features she wanted as a user. They also asked a few questions about her workflow to understand what was important to her as a user of the software.

This interview gave the group a lot of ideas for tasks to do. It was therefore made a Product Backlog, prioritising the tasks after a matrix containing importance for the user and degree of difficulty for the group (Appendix E).

6.1.1 Result

The first interview gave the most answers. In this interview, the group learned about what the archaeologists needed in the software and what it had before. This

was mostly about functional additions that would make the software easier to use.

One of the things that came up in the interview was that scaling the fragments was not something they wanted to be possible. The pictures of the fragments have been taken from the same distance for every fragment, so their size would be correct compared to each other. Having the feature of segmenting the images would be a good idea so that one does not need to do any preprocessing on the images before loading them into the application (Section 10.15: Segmentation). That would mean removing the blank space around the fragment, that are a part of the image. Cropping the images could also be a good idea, as some parts of the fragments are impossible to find patterns in due to degradation and fading over time. They could, therefore, be cropped out if the user felt for it.

A feature that already existed in the software was the ability to change the hue, saturation, luminance and contrast values of the fragment images. However, the way to do this was not very convenient, as all values just had numbers that the user needed to put in. It was very hard for the user to understand which numbers meant what and how they would affect the images. An idea that came up during the interview was that this could be represented as sliders (Task 4a p.24).

There are a lot of small functions that were also mentioned by the interview subject. She said the filters should apply to the selected images individually (Task 5b p.26). The ability to take a screenshot of the project (Task 1d p. 24) also came up during the interview.

The capability to save a project was a fundamental topic during the interview. That way, the user could have the ability to work on multiple projects at the same time without worrying about losing the work on some of them (Task 1c p. 1c), which came out as something essential.

Undo and redo functionality was also mentioned as something important (Task 4e p.25) and an opportunity to add a text box connected to an image on the screen. Giving credit to the photographer of the fragment images on screen was desirable.

One bigger feature that the user wanted was, among several, the ability to draw on top of the canvas as an own layer (Section 10.19: Drawing). When interpreting archaeological textiles, sketching is used a lot. [49] Another one was to have a ruler on the canvas that would be updated as the user zoomed in or out (Section 10.14: Size Indicator). In that case, the user could count the number of threads per centimetre or measure other technical characteristics.

Features that need to be kept from the original versions were the ability to zoom in and out, moving the workspace, and basic features like loading an image onto the canvas and moving it around.

The group learned much about the archaeologist's work process in the interview. One thing was that looking at the motifs on the fragments was more important than looking at the edges for identifying matching fragments, but that could also vary from person to person.

6.2 Interview 2

Right after the first usability test, we hosted the second interview. This marked the end of the first development cycle and the beginning of the second. We were then back at phase 1: the understanding phase. In this cycle, we wanted to have more focus on the computer vision part of our bachelor's thesis.

6.2.1 Result

The second interview focused more on the computer vision functionality that could be useful in the software.

From this interview, the group learned that using the HSV colour space for filtering the image is very convenient, as it is what they use in other software today, such as Adobe Fresco. We learned that the brightness function might be the most important feature, but combining several filters still gives more value to the user. Some of the functions from Adobe Fresco are also desired to be in our software. That includes adding filters that will increase the differences within the image (Task 4d p.25).

Noise removal and contrast filers are some filters that would be very useful (Section 10.3: Better Filters). Filters that could be helpful to have but might not be the most useful are converting to greyscale, inverting each pixel's value, and masking by removing every pixel darker than a set threshold and line- and edge-detection. A histogram with scores for the hue values of an image would also be insightful for the user.

We asked some questions about the work process and goals for the archaeologists while working with the fragments. The usual approach was adding some filters to the fragment that would enhance its differences. By doing that, figures and patterns could become visible in the fragment. Different filter combinations could work for each fragment, so trying out various combinations was crucial.

CHAPTER 7

EVALUATION

The evaluation phase is a phase in the design process [15]. This is the phase where one wants to get insight into how easy the software is to use, i.e. its usability.

7.1 Usability tests

The group hosted three rounds of usability tests on our software while developing it. The usability tests are a good example of summative evaluations as they are done in the Evaluation Phase towards the end of the SDLC [15]. All of the usability tests are done on the actual product, not a prototype. The usability tests answered questions about the functionality that had been implemented by us, and a few items from the original program. More specifically, they showed the usability of the features. Thus, they gave the group smaller and bigger tasks to fix or change to improve usability.

A template was made before the two first usability tests (Appendix J), with the structure for filling in rules, tasks, and questions about each task. Then, we used the template for each of these usability tests to create a usability test form (Appendix K).

7.1.1 Usability test 1

The first usability test was held online via Teams and marked the end of the first cycle in the usability process. Therefore, this was our first evaluation phase.

We chose to utilise the Think-aloud protocol [50], which means that the test subject should say aloud what they are thinking about during the test. This will help us understand how users think and how they would move forward, such as finding a specific button. Otherwise, we used the "concurrent probing" [51] method, which required the test subject to say something about the usability after every task, as we had a few pre-made questions. We asked questions about

how easy it was to find and use the feature if the number of clicks was decent, and how the design was.

The tasks for the first usability test were:

1. Saving a new project.
2. Opening a saved project.
3. Exporting an image of the canvas.
4. Take a screenshot when right-clicking the canvas.
5. Opening the score window (Similarity Metrics Window).
6. Move and rotate multiple images simultaneously.
7. Delete multiple images simultaneously.

Results

The first test gave us a lot of great feedback based on the tasks.

Saving and opening a project went smoothly.

When exporting an image of the canvas, the user did not understand what scaling the exported image meant. She assumed it could go over 100% but thought it was meant for cropping. It would be better if there were options for low, medium, high, and advanced to choose between (Section 10.5: Automatic Sizes on Export Image of Canvas). She also wanted the option to select where to save the image and what to name it since it was just put in the download folder with the name "image.png" (Task 4j p.25).

Right-clicking to select the built-in screenshot function was fine, and she liked the feature.

She did not find the button to open the score first, hesitating and wondering where it was. With just the name and size of the images displayed in the window, she did not see the point of the window as it did not include helpful information. The last thing she noted was the button's name since the term "score" did not tell her anything. Naming it "Similarity Metrics" would make more sense. This was changed in the software.

When she tried selecting multiple images, she first tried holding the shift key to select more images. She was familiar with using this in software like PowerPoint. We added the possibility of also using the shift key to select images.

She also tried to use the backspace button when trying to delete an image, which we had not considered. We also changed the software to make it possible to use the backspace to delete the selected.

Summing up what we learned:

- It is hard to understand the scaling functionality when using the export image of the canvas. It would be easier with just three options to choose from (Section10.5: Automatic Sizes on Export Image of Canvas).

- The user would like to choose where to save the image on her computer and the option to give the file a name (Task 4j p. 25).
- Hard to find the button for "score".
- The “score” name was unintuitive. It should be named to something like "Similarity Metrics".
- More intuitive for the test subject to use the shift key than the ctrl key for selecting multiple.
- More intuitive for the test subject to use “backspace” than “delete” to delete the selected.

Depending on how you look at it, the usability test could have gone better. We asked questions that came to mind during the test instead of asking them after. Asking the pre-made questions after each task also "broke up" the flow a little bit. The testing also took longer than expected.

7.1.2 Usability test 2

The second usability test was done in person. In this session, we chose to continue using the Think-aloud protocol [50], but this time, we tried to utilise "retrospective probing" [52] to ask the user questions. We did all the tasks, expecting the user to say what was on her mind, but asked the main questions afterwards, looking through every task again. During it, we took a screen recording that we could look through afterwards. Before the second test started, we went through a list with the test subject, explaining in more detail the test process of Think-aloud protocol [50], retrospective probing [52], and how we wanted her to act during the test accordingly

The tasks for the second usability test were:

1. Make a project and load multiple images onto the canvas.
2. Lose the workspace and find it again with the tool.
3. Open the filter window.
4. Add a filter to all the images on the canvas.
5. Reset the filters on one image.
6. Undo the last change.
7. Redo the last undo.
8. Explore the filter options.
9. Disable the filters and compare.

Results

Creating a new project and loading images onto the canvas was as easy as last time since the buttons were located in the same place. This time, however, she used the trackpad instead of a mouse and noted that the zoom function works

a bit quirky since she has to drag two fingers up and down to zoom instead of pinching and spreading two fingers on the trackpad (Section 10.20: Pinch and Stretch Zoom).

When we told her to lose the work area and find it again, we did not specify how. So, she overthought the task and took some time to look for a button to do it for her. When she first looked under the right drop-down menu, she found the button called “Find work area” and returned to the work area without a problem.

Then, we started testing the new filter window. Opening it took some time since there were not any obvious buttons that would do it. She eventually found it could be opened with right-clicking, which made sense when no button existed. She did, however, expect that when left-clicking another image, she would target it for the filter window, but it only worked when right-clicking the new image. When asked to add a filter to all images, she first tried to select all and apply a filter, but it only targeted the one she had right-clicked (Task 5b p.26).

She noted that she did not understand what the "value" slider meant and that the three value, luminance and mask sliders all being greyscale could make them difficult to differentiate when tired. Some rearrangement of the filters and look changes would be nice. Other things that could be done were changing the contrast slider from just grey to grey $\rightarrow$ hue $\rightarrow$ black, moving the contrast slider to be under saturation, and renaming "Mask threshold" to something like "Edge reduction" (Task 5a p.25).

When resetting the filters on an image, she wondered if the "reset all" button would reset the filters on all the images instead of just the one. She suggested naming the button “reset filters”. The “enable”/”disable”-button was suggested to have the name “filters”. The option to reset a single filter was great. Here, text on the reset button, instead of a symbol, worked great.

The undo and redo buttons and keyboard shortcuts worked great, but she noted that the buttons would be better with symbols instead of text (Task 5c p.26). When exploring the filters more, she noted that using the contrast and invert filters simultaneously worked as she thought they would.

She also noted that the sensitivity of the saturation slider was high, and how the "mask threshold" ate up the image like acid. The last task was to disable and enable the filter, which was not a problem since the button was easy to locate.

Summing up what we learned:

- The zoom function is unintuitive when using a trackpad. It should use the pinch zoom, which we will discuss in future work (Section 10.20: Pinch and Stretch Zoom).
- A little hard to find the “find my workspace”-button.
- Hard to find out how to open the filter window.
- The user might think that another image’s filter window is chosen when left-clicking the image, when one needs to right-click it.
- Could be slow if you want to add the same filters to multiple images, which

we discuss more in future work (Section 10.1: Memory Problem Solutions: Rewrite or Change Environment).

- Name “Value” should be changed to "Brightness".
- Some of the colours on the sliders could be changed.
- The overall look of the filter window can be improved, and some things can change places.
- The “Mask threshold” name is not easy to understand for the user.
- Symbols would look better than text on the “undo” and ”redo” buttons. (Task 5c p.26)
- Saturation slider is very sensitive.

One thing that became an issue was the fact that the user felt that following the Think-aloud protocol made her overthink. She felt that she used more time on the tasks because she needed to say a lot about her thoughts. That prevented her from acting faster and trying out more options, as she would normally do. Doing retrospective probing instead of concurrent probing worked well, as all the pre-made questions were asked at the end of the test. That gave a better flow in the work during the second usability test.

7.2 Heuristic Evaluation

We got feedback from a UX designer in the second evaluation phase right before the second usability test. This was a heuristic evaluation but was done in a more casual setting. We only got feedback from one expert and she did not have expertise in archaeology. Hence, she did not have insight into an archaeologist’s specific needs and wants. We still used this to get experience in using different evaluation methods on the software. The feedback we got from the UX designer was, therefore, very general and had to be confirmed with the intended users. The feedback was as follows:

- The "Home" button is in a “scary” position. It’s not a common position to have it, and there is no error prevention when a user clicks on it. This violates Error prevention and Consistency and standards. (Task 4f p.25)
- The Undo and Redo buttons in the navigation bar could be represented as arrow icons instead of text. This is to make the software comply with Recognition rather than recall. (Task 5c p.26)
- The navigation bar should maybe only have drop-down menus or separate the undo and redo from the drop-down menu buttons. These drop-down menus could be “file” and “tools”. This is connected to the heuristic of Consistency and standards.
- When loading in multiple images, the images need to be offset more to separate them more from each other to increase the Visibility of the system status.

- Users should be unable to rotate an image when the canvas is locked to prevent errors.
- The disable and enable filter should be enabled by default, and the button should be in the filter window and could be represented as a toggle button. This will make it comply with the Visibility of system status.
- Fix the bug that occurs when closing from the landing page.
- Deleting images when clicking on backspace. A delete button in a context menu could also appear when right-clicking on a piece. This would only delete that image. This is to give the users Flexibility and efficiency of use (Section 10.21: Context Menu).
- Redo should also be on ctrl + shift + Z to better comply with Consistency and standards.

All the feedback that was confirmed by the intended end user was implemented and some of the rest too. We saw that making the undo and redo button arrow icons separated them enough from the drop-down menus, so we did not do more than that. We changed the filters to be enabled by default, which caused latency in the software, so we later changed it back. We did not prioritise changing the visual of the enable filter button. Making a context menu button for deleting an image was left for future work.

7.3 Puzzle Usability Tests

During the first interview with an archaeologist, she mentioned to us that one of the potential uses for the software was to be used on a public computer with the software loaded. Then visitors to the KHM could try loading in fragments themselves to find fragments that could be adjacent to each other. In that sense, the software could be developed into a crowdsourcing data collection tool using people from the general public. This also means there is potential to make the puzzling process gamified.

To test the feasibility of this specific use case of the software, we wanted to test the software's ability to help the user solve a puzzle. We wanted to use regular people as test subjects, as that would reflect the use case better, but also to get a broader perspective than just using one test subject. We also chose to use images other than the usual fragments, as we wanted to test the puzzling functionality more wholesomely. To do this, we created a custom puzzle using an image we found online [53], that we thought was a good candidate for the test, as it has large homogeneous areas with the same hue, as long as many details and a big diversity in hue values. (Figure 7.1).

We wanted to create individual puzzle pieces that have interlocking interjambs (protruding part) and blanks (indented part), but no tool that created a puzzle had the option to download individual pieces. Therefore, we found a tool called ImageSplitter [54] that allowed us to cut the image into individual squares. Using it, we created a 4×6 puzzle we could use for the tests (Figure 7.2).

Figure 7.1: Original seaside picture used for creating a custom puzzle [53]

Figure 7.2: The split image when assembled with a little spacing

However, since this image with many different vibrant colours is not a very good indicator of how the software will work on the archaeological textile fragments that are often highly faded and degraded, we also wanted a test where the pieces were more bland and not so easy to differentiate. For this test, we used pictures we got from the product owner of a fabric that had been cut into several pieces and further digitally separated (Figure 7.3). These textile pieces had been used as subjects of automatic puzzle-solving research in the puzzle assembly master thesis. [5]. They were well preserved in comparison to the fragments, but their colour distributions were still closer to the fragments than the other images we found on the internet.

Figure 7.3: Assembled flower picture with a little spacing

After three tests, however, we noticed that there was not as much of a difference in the helpfulness of the similarity metrics as we wanted between using the seaside and flower images. So, for the next tests, we got another image (Figure 7.4) to use that would create the difference we wanted. When we tested it ourselves, the similarity metric pointed to an orthogonal or diagonal neighbouring piece as the best match for every piece except three. The top right piece pointed to the piece two rows down and vice versa, and the bottom right piece pointed to the piece on the second row, second to the last column.

The task's purpose during the test was to test the whole process, from loading images to solving the puzzle for both the test images. We also wanted to test each user's preference between using just their eyes and the similarity metric.

Tasks:

1. Start a new project and load the puzzle pieces onto the canvas.
2. Separate each piece from each other randomly so they are not stacked on top of each other. Do not try to solve it yet.
3. Find 4-5 pieces that fit together and assemble them.

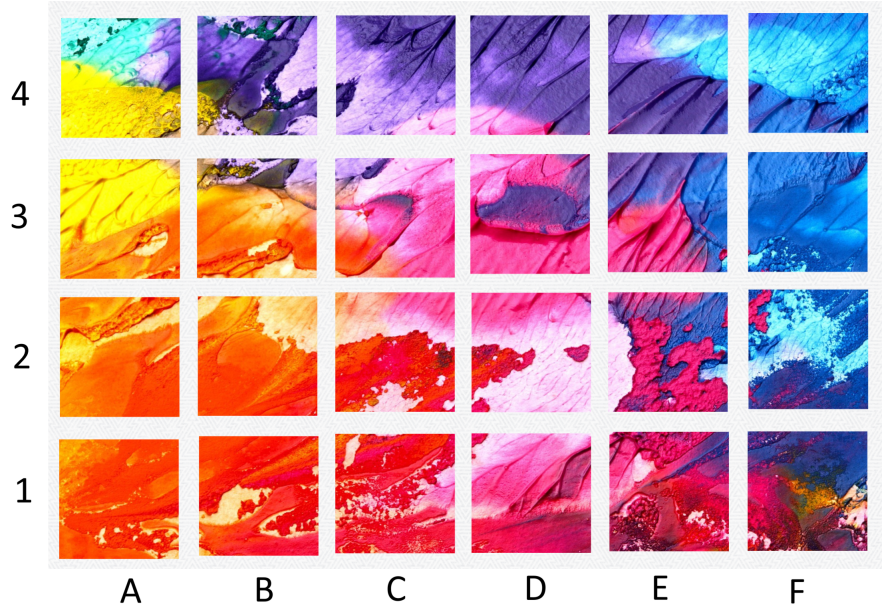


Figure 7.4: The colourful image used in the second round of usability tests [55]

4. Use the Similarity metrics tool on a selected piece. Do the same as the last task again with the tool.
5. How did you find using the similarity metrics compared to just your eyes?
6. Additional comments on the test process?

7.4 Usability Test Result

These tests were run after the software was finished, and for each subject, we ran the tests twice. First, it was run on the seaside or the colourful picture to let the test subject learn the controls and try to find 4-5 pieces with and without using the similarity metrics. Then, the same was done on the flower textile picture. The seaside and colourful pictures are made to test the software when there are many different colours and details to make it easier for the user to use both their eyes and the similarity metrics. In contrast, the flower picture is meant to better represent the challenges the archaeologists face when finding matching fragments that will fit together. We did the test on fellow students from NTNU Gjøvik, among others, and this is what we learned. The document for the usability tests is in Appendix L.

7.4.1 Getting familiar with the controls

The test's start focused on familiarising the test subjects with the software's controls. They had no problems finding the button to load images onto the canvas. As soon as the task had been given, they moved to the file button to find it. One test subject tried the "tools" scroll-down menu first but quickly found out it was wrong and then found "file". A problem for the first subject was that they did not like having to ctrl + A to select all images in the folder. They felt selecting a folder and loading all its images would be easier.

Some test subjects quickly got used to moving the pieces around and separating them from being stacked on top of each other, although they had multiple different ways of approaching it. Others struggled with it during the whole test, even though they had learned how to do it correctly. One subject found it frustrating that many of the elements spawned outside the screen when loading in many at the same time. He also found it hard to understand how to zoom out. One other used a while to discover that it was possible to zoom out and did not understand that more elements had been loaded in than only the ones visible on the screen. However, she did find out after a little while.

Several of the subjects tried to click and drag immediately instead of clicking to select and then clicking to drag again, which a few of them did not like. However, another subject noted that it meant that every movement of a piece had to be intentional. The same user also tried moving pieces and the canvas with their touch-screen, which worked fine but not zooming. Another found the left click to select and move a piece and thought it was only meant for that purpose, so they tried using the right click to move the canvas.

7.4.2 Not using vs. using the similarity metrics

Seaside image

The first three test subjects started with the seaside picture. When they tried to find similar pieces using their eyes, two test subjects started with the ocean in the bottom right and expanded to the left towards the cliffside, and the third used the branches on the top right to complete four to five pieces. The time to complete the task was not taken since the objective of the task was not how fast it could be done, but they all completed the task relatively fast.

All test subjects found it challenging to familiarise themselves with the similarity metrics. Scrolling down to the bottom to look at the most similar piece every time they clicked on the canvas to move was an annoyance. A user had problems understanding the table because they thought higher was better, and it was never said otherwise. Finding pieces that would fit together took longer since the tool only gave a neighbouring piece half of the time. They all eventually found some pieces that fit together from the houses or other infrastructure.

Colourful image

When the test subjects tried to match pieces together from the colourful image, they found it a little bit hard, as the image depicted a more abstract scene, and they did not have any concrete figures to look at. They quickly found out, however, that using the colours would help them a lot, as the colour was changing gradually.

The three test subjects that used the colourful image all managed to get results from the similarity metrics that could help them connect the puzzle pieces. One of the test subjects tried to look at more than just the final score and tried to get some useful information from the other scores than just the combined scores. He did not completely understand what they meant, but he found the piece with the highest score from one of the categories that was not the same piece as the

"most similar piece", and this piece would also fit together with the chosen puzzle piece. Another subject found it frustrating that the metrics would give the user the most "equal" image without saying what that meant. She would rather have a metric giving the image with a matching edge. Several test subjects said they would want the metrics to give the user more than just one result when calculating the similarity.

Flower image

When the test subjects had to do the test again on the flower image, the users took longer to find 4 to 5 pieces than the first image since there were a lot fewer details in the images for them to base their judgement on. However, some of the test subjects did find out that they could use the type of flower or leaf to find matching pieces. Therefore, most of them eventually found the upper left red flower on pieces 3A and 4A, and some found other combinations, even if it took longer. One user relied on the shape of the pieces to connect them, as that was possible in this puzzle.

The problem arose when they had to rely on the similarity metrics to find pieces. They struggled with the same problems using the tool as when they did it on the first image, but this time, they also struggled more with finding matches that would fit together. The test subjects had to click on almost every piece to find some that would fit with each other. Still, it was a struggle, as it looked like it could fit, but when put around each other, it was evident that they did not. It took several times longer to complete this task than with the first picture. Two of the test subjects had to give up because they felt like they had tried every piece and could not find a match, even when looking at the number scores.

7.4.3 Proof of concept

The similarity metrics tool works well on very specific images, like the colourful image 7.4 and the seaside image 7.2, and does a worse job the more bland and less vibrant a complete image is, like the flower picture 7.3. With this test, we wanted to show that the tool can be useful in some situations and is, therefore, a proof of concept.

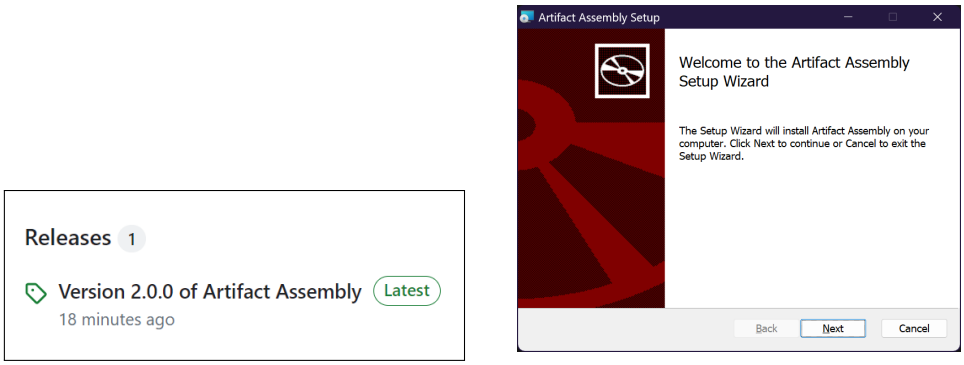
CHAPTER 8

INSTALLATION

The installation is the same as found in the README.md file in the GitHub repository (Appendix G).

8.1 Latest Release

To install the latest version of the software, head to the GitHub page (Appendix G) and download either the .exe or .msi file from the latest release (Figure 8.1a. When downloaded, run the installation wizard (Figure 8.1b).



(a) Latest release

(b) Installation Wizard

Figure 8.1: Github release and install wizard

8.2 Start Development

To start development on the software, you need the source code, which can be downloaded or forked from our GitHub (Appendix G).

8.2.1 Tauri Prerequisites

The software is dependent on Tauri to make it a standalone runnable. Tauri has some prerequisites to be able to run [56]. For Windows, which we used, it will need the three things listed on the website.

1. Microsoft Visual Studio C++ Build Tools.
2. WebView2.
3. Rust.

After downloading and installing build tools [57], it is important to select "Windows development with c++" and the latest version of the MSVC package and the latest version of the Sofware Development Kit (SDK) for either Windows 11 or 10 depending on what is on the computer (Figure 8.2).

Figure 8.2: Build tools packages

Webview2 comes built-in with Windows 11 or can be downloaded [58] for Windows 10.

The easiest way to install Rust is using the rustup installer [59] and following the instructions, or it is also possible to do it through the command line.

Code 8.1: Install Rust command

```
C:\>winget install --id Rustlang.Rustup
```

When Rust is installed, it is important to make sure the correct toolchain is selected by running this in the command line.

Code 8.2: Confirm correct Rust toolchain command

```
C:\>rustup default stable-msvc
```

With this, Rust and Tauri are set up correctly.

The last prerequisite is the Node Package Manager (NPM) tool, which comes with the NodeJS [60] installer. When the installer has been downloaded and run, the installation can be confirmed in the command line by running.

Code 8.3: Confirm node and npm install command

```
C:\>node -v
C:\>npm -v
```

8.2.2 Testing and Building

Before running tests or building installation files, the software's node packages must be installed using the command line inside the project's root folder. This will create and fill the node_modules folder with the packages.

Code 8.4: Install npm packages command

```
C:\>npm install
```

Then, to run a local test development version of the software, run this command:

Code 8.5: Run Tauri test environment command

```
C:\>npm run tauri dev
```

To create installation files, run this command:

Code 8.6: Run Tauri build command

```
C:\>npm run tauri build.
```

The installation files can be found inside the src-tauri folder.

```
/src-tauri/
└── target/
    └── release/
        └── bundle/
            ├── msi/
            │   └── install_file.msi
            └── nsis/
                └── install_file.exe
```

8.2.3 Starting and Running

The development test will start after the command is run and will run as long as the command is not stopped or the software is exited. Then, the command will need to be run again to start the development test. After the software is installed, it can be found by searching for it in the menu (Figure 8.3), and will only stop when the software is exited.

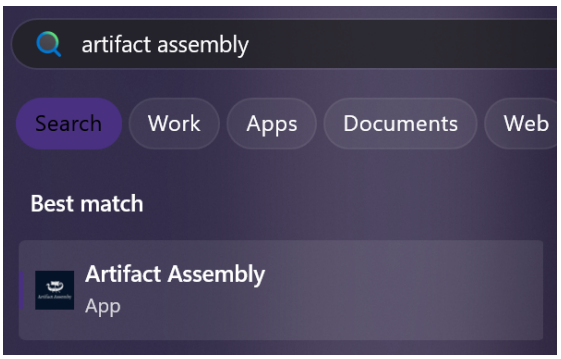


Figure 8.3: Start menu search for Artifact Assembly

CHAPTER 9

DISCUSSION

9.1 Usefulness Results of the Software

Looking at the results of our work with Artifact Assembly, we can discuss its usefulness.

With the new features added or improved, the archaeologists now have more options than they previously had in the software. The new filters and the image selection improvement will give them more fine control to filter either one or several fragments at a time. The finer controls will allow them to highlight the subtle differences in colour to study the patterns that emerge. The Similarity Metrics can indicate which fragments could be adjacent to each other, and in combination with the new filters, it might perform better than expected. It will take practice and experience to get familiar with using the similarity metric. Most importantly, after finishing their work for the day, they can now save the progress and continue on it later. Other helpful features, such as the export image feature, will allow them to combine two fragments into a single image instead of selecting the two each time or sharing the results with others, and the find workspace will ensure they never lose their work by accident.

Reapplying how the software is used, the archaeologists can let visitors to the museum try different combinations of fragments. With this method, one visitor could be lucky when looking for adjacent fragments and find something the archaeologists have not found yet. Given that the program is open-source on GitHub, it might be possible to set up a way for enthusiasts to download the program and fragment images in their own time and keep trying at home, reporting any findings to the museum through proper channels. They could also use the software to solve their puzzle problem if they have digital puzzle pieces.

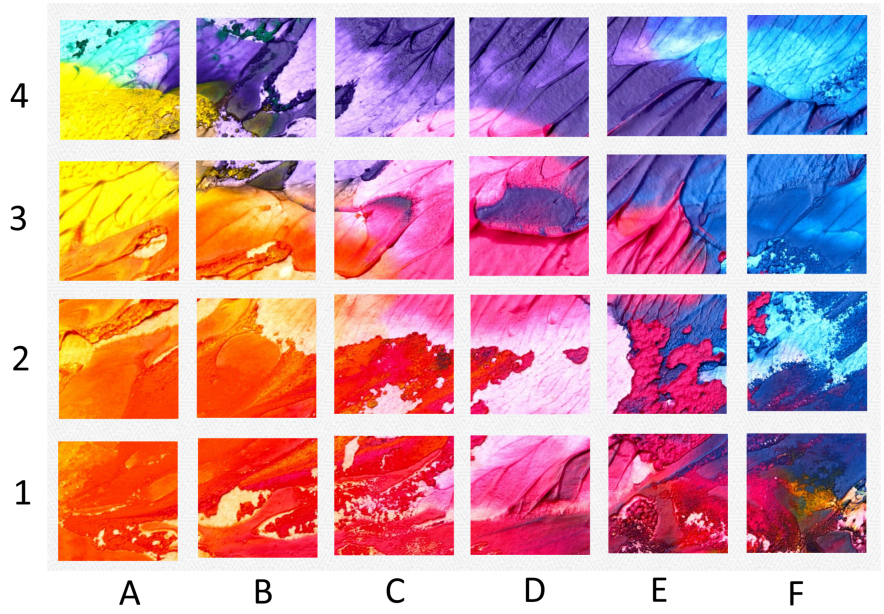


Figure 9.1: The assembled version of the Colourful image

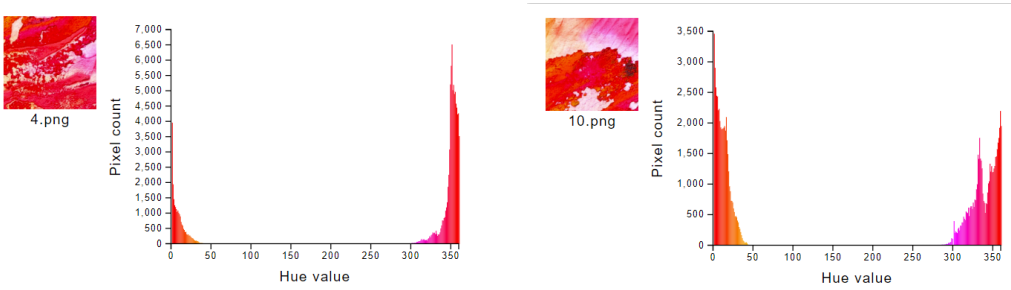
9.2 Similarity Metrics Results

Say we want to use the similarity metrics to solve the puzzle of the Colourful image (Figure 9.1). We select piece 1C (Figure 9.2a) and open the similarity metrics window. The piece 2C (Figure 9.2b) is evaluated as the most similar piece to 1C (Figure 9.2c) because they have the lowest values on the combined metric. Their hue histograms are alike, having almost all their pixels at hue values between 0 and 50 and between 300 and 360. That means they have similar colour hues. Since they are neighbouring pieces in the puzzle (Figure 9.1), this is one example of how the chosen metrics work well on the image "colourful" and is proof of success.

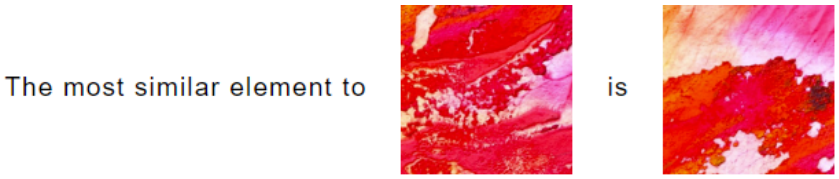
Say we then want to do the same to solve the Flower image (Figure 9.3). We open the similarity metrics window for piece 2E (Figure 9.4a). The element most similar to 2E is piece 2A (Figure 9.4b). That is because they are similar in hue shown by two their matching hue histograms as they both have a lot of pixels around 0, between 25 and 125, and between 200 and 250. These two pieces, however, do not belong together, as seen in the solved puzzle (Figure 9.3). This is thus an example of a situation where the chosen similarity metrics would not work.

Summing up the similarity metrics worked much better on the colourful image (Figure 9.2c) than on the flower image (Figure 9.3). This is because the metrics gave us images with similar hue values. For example, in the colourful image, one yellow piece would be placed close to another yellow piece because the image was gradually changing colours. In the flower image, all pieces would have the same background colour, with similar patterns all over the image in terms of hue values and an image could be alike in hue on the opposite side of the puzzle.

Even though there is a limited benefit of using the similarity metrics as it is today, our focus was to develop some basic computer vision features to make



(a) Piece 1C histogram **(b)** Piece 2C histogram



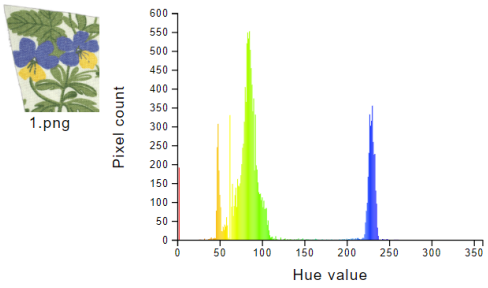
(c) Piece 2C is chosen as most similar to piece 1C

Figure 9.2: Comparing 2 pieces of the Colourful image that the Similarity Metrics indicate as 2C being closest to 1C

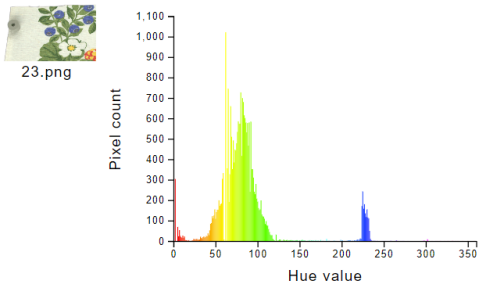
Figure 9.3: The assembled version of the Flower image

the software usable. Therefore, a small part of our time was utilised to work on this. We did not expect the chosen metrics to solve the whole puzzle, as it was only supposed to be a helping tool for the users. Therefore, we understand its limitations in solving the puzzle. This is therefore a first step towards introducing some intelligence into the software. After all, we created a platform for developing and researching more computer vision metrics and algorithms that were beyond the scope of our thesis.

To find the right metrics that will help the archaeologists in the best possible way, much more research needs to be done on the computer vision field. [5]



(a) Piece 2E histogram



(b) Piece 2A histogram

(c) Piece 2E and 2A similarity

Figure 9.4: Comparing 2 pieces of the Flower image that the Similarity Metrics indicate as 2A being closest to 2E

9.3 Results of the Project Goals

Here, we will discuss how the results of our work satisfy the goals we set at the start of the project (Section 1.2: Project Goals).

9.3.1 Result Goals Results

In the Result Goals, we outlined our desire to create software that the NTNU Colourlab can use to continue their work to help the archaeologists at KHM. Implementing as many features as possible during the project's time frame while making the software as user-friendly as possible.

At the end of development, we accomplished many of the original goals that met the expectations of the intended end users. We implemented the main features of the project: a similarity metric to suggest to the user which fragments are similar and hence belong together, improving the filters, saving a project to a file and loading from that file. This is a pivotal feature to make the program usable in practice since without being able to save and load, users would have to complete all

tasks in one go, which is infeasible and makes the program unusable. We consider many other features important, but these were the most important to make the software usable.

There are more features that we could have implemented that were of value to the user but were out of the scope of this thesis due to time limitations. These features we will discuss as possible future work (Chapter 10: Future Work). We are however pleased with the software as it is. The usefulness of Artifact Assembly to the archaeologists' work was confirmed by the archaeologist in the last interview (Section: 6.2: Interview 2). The software does not replace their workflow and may never do so, given the limitations and current state-of-the-art in computer vision and machine learning [6]. But our work is a step in the right direction providing direction for future work. It addresses gaps in their workflow, providing a way to compare two or more fragments without physically manipulating highly fragile historical artefacts.

9.3.2 Impact Goals Results

We improved the program we inherited, and now it is ready to be used and offers the archaeologists a good opportunity to try different arrangements of the fragments. Even though evaluating the program's impact at this stage is difficult because it has not been used for practical problems yet, we foresee that it has a big potential to contribute to archaeology research and pave the way for further interdisciplinary research between archaeologists and computer scientists.

9.3.3 Learning Goals Results

Taking on a project as big as a bachelor's thesis was new to us, and we all gained new knowledge and experiences during it. Inheriting a program and continuing work on it were both easier and harder than we had thought. It was easier because we did not have to spend time researching and deciding on a programming language that would fit the needs of our problem. The base software and functionality were already created, so we did not need to create it from scratch. It was harder because of our limited experience with inheriting software. We had to learn a new programming language, which was not a big problem, but we did not realise how challenging it would be to get used to a code and file structure that we did not create ourselves when there were little to no comments in the code.

Learning JavaScript, React with JSX, and Konva.js was a great experience. Based on comments from the product owner and supervisor in our first meetings, we originally thought we were writing the code in Rust when going into the project. So we started learning Rust and were surprised when we got access to the GitHub repository and saw that only 1% of the repository that was written in Rust account for the Tauri code. We switched direction to focus on which programming language we needed to learn and felt we gained valuable project planning experience by failing to ask more probing questions about the existing code.

We learned a lot about user-centred design during the work with the software. By implementing three rounds of usability testing, we gained experience with what we had previously only learned about in theory. We also feel that we improved our

skills after each usability test. One of the hardest things to remember during the test was not to help the user when they were stuck unless the problem concerned something other than our software. This could be very tempting, but the test aimed to determine how the user would react in these situations. In the first usability test, we failed on this a few times, but it was drastically improved by the second usability test. Another important thing in usability testing is using the Think-aloud protocol. We explained this under every usability test, but in the second one, we also showed a small example of how it worked. This made the user understand it even more.

The learning goal we could have accomplished better is learning more about using computer vision to extract information from the fragments and comparing them using the information. We researched and learned about some methods to create a proof of concept with the hue histogram, but we could most likely have learned more and created something better if we had more time, which we will discuss in Future Work (Section 10.2: Better Similarity Metric). However, we are satisfied with what we learned about using computer vision to make colour enhancements and applying filters to the fragments. We applied what we learned to improve the existing filters and add new ones.

Working in an interdisciplinary team during the project taught us much. Communicating with people who did not have a computer science background was different than only working with people in the same industry. There are a lot of words one uses that are typically only in use within the computer science industry, for example, that people from the outside might not understand. During the usability tests and the e-mail communication with the archaeologists at KHM, we ensured that we either avoided using technical words from computer science or explained it to them extra well. That did not mean we avoided telling them technical details about the software that could interest them. Instead, we tried to use more general words, and explain it in a way everyone could understand. Therefore, working on a project across different industries was a great learning tool for improving our ability to explain terms and concepts and, therefore, our understanding of them.

9.4 Sustainability

Our project can help make a more sustainable society in a few ways. We provide a way for the archaeologists to work with these fragments in a digital workspace. Since the fragments they are working with are very fragile, there is no option for them to work with the actual fragments without damaging them. Another solution the product owner said they had thought about is using 3D-printed models of the fragments to study different arrangements. In addition, they also thought about printing the fragments on paper or textiles. In that case, they would have to use much material to produce the 3D-printed, paper and textile models. Using a digital version of the fragments will reduce the resources needed to make these.

This aligns with the United Nations (UN)'s 12th sustainability goal [61] (Figure 9.5c) to "Ensure sustainable consumption and production patterns". The target relevant to our project is 12.5. It says the following: "By 2030, substantially

reduce waste generation through prevention, reduction, recycling and reuse". Our project will contribute to "prevention" and "reduction" as we will prevent the resources needed to 3D-print from being used in the first place.

Since the archaeologists are part of the KHM, their findings will eventually be published and may be displayed at the museum. We believe this aligns with UN's 4th goal [62] (Figure 9.5a) to "Ensure inclusive and equitable quality education and promote lifelong learning opportunities for all". The museum is open to everyone and will provide quality education and the opportunity to learn more about the Vikings to everyone who visits, ranging from children on a field trip to tourists and pensioners.

The project aligns with the UN's 9th goal [63] (Figure 9.5b) to "Build resilient infrastructure, promote inclusive and sustainable industrialization and foster innovation". Specifically fostering innovation is relevant. The project, if taken toward improving the Similarity Metric, can foster innovation in the computer vision field to create algorithms custom-made for deteriorated textile fragments. It also aligns with the 9.5 target to "Enhance scientific research, upgrade the technological capabilities of industrial sectors in all countries [...]" as the software will exchange the research the archaeologists can do.

(a) Goal 4 **(b)** Goal 9 **(c)** Goal 12

Figure 9.5: Relevant United Nations Sustainability Goals

9.5 Project Process

Here, we will evaluate and discuss the group's experience of the project process.

9.5.1 Planning

After the first interview with the archaeologist, the list of items in the Product Backlog was a great starting point for planning what we would do in the Sprints. Using the Product Backlog was great to plan each Sprint Backlog. A problem we had in planning both the Product Backlog and the Sprint Backlogs was the scope creep we experienced. New items were steadily added to the Product Backlog, and sometimes, we felt it was too overwhelming to overcome and plan when there were so many different items. This was anticipated and communicated clearly to the product owner and the users that we would not implement all the features.

9.5.2 Meetings

We had regular meetings with our supervisor and the product owner. Since both of them are part of the NTNU Colourlab and working on the TexRec project with the archaeologists, we did not have separate meetings with them. These meetings were great for developing the software, as we got great feedback from them in place of the archaeologists. This meant, however, that it was sometimes challenging to differentiate between having a meeting focused on development and one focused on the thesis. It meant that we had a late start on writing the project plan early on in the project since we were focused on discussing the software we would develop, and later in the project where we discussed the software more and less about how we should set aside a bit of time to write on the thesis during the project not to have to catch up later.

9.5.3 Collaboration

The collaboration went mostly smoothly using GitHub. We could all work on different tasks independently of each other. Some tasks were dependent on another one, and if one of us wanted to work on a dependent item, they had to postpone starting work on it and work on something until that item was finished. Sometimes, we had merge conflicts, but they were quickly fixed. The Daily Scrum meetings allowed us to get up-to-date on each member's progress. This optimised the collaboration and allowed communication to flow freely.

9.5.4 Development Process

We view the decision to use Scrum for our project as a success. The iterative model allowed us to modify our approach to develop the software as time passed and improve our process per Sprint. The Scrum Guide [7] defines the Sprint review and retrospective to be a maximum of four and three hours, respectively, but usually lower for shorter sprints. We fit our modified version of the meetings inside a 2 to 2.5-hour window, which perfectly suited our two-week Sprint time. The two-week Sprint allowed us a long enough time to complete tasks while also being short enough so that we could regularly review the items steadily, which we are satisfied with.

Something we would have done differently is how we made the Product Backlog. It was written in Excel (Appendix E), and it worked for our project, but we felt it was not an optimal solution. If we knew from the start how many more items would be added, we would have liked a way to keep a version history of the items. Short of creating a copy of the whole Product Backlog Excel file every time it was updated, using a Kanban board would have been better. Therefore, having a Scrum-Kanban fusion as a development model for the whole Product Backlog instead of only using Kanban for the items we had started on would have been better.

Some of the items added to a Sprint Backlog were too big for that Sprint and were worked on over multiple Sprints because they were either too broad or we did not have enough experience with React to complete it during a Sprint. Through this experience, we learned to split items into more manageable parts to finish

them during a Sprint and have a better overview of our progress.

9.5.5 Time Management

For such a big project as a bachelor's thesis, keeping time of the work done during it is important. Finishing the other course we had alongside the project, we realised we had not put in enough time as we could have during the days assigned for the bachelor thesis work. We, therefore, started trying to work the same as a full working day as much as we could from then on. This worked great; we did not feel pressured at the end of the thesis with too much to do. The last week was the only exception where we did more work to finalise and proofread the thesis.

To log the time, we used Excel. With a list of activities to choose between and custom Excel functions, we got a great overview of our work on each activity and how much each member had worked on each activity. The distribution of the total hours (Figure 9.6) each member had worked was around equal. Some differences appeared slowly over time as we had slightly different working schedules, and one member was sick for a little while and had some other absences, which is why they had a lower percentage of total time put into the project. This time difference proved not a problem for the project.

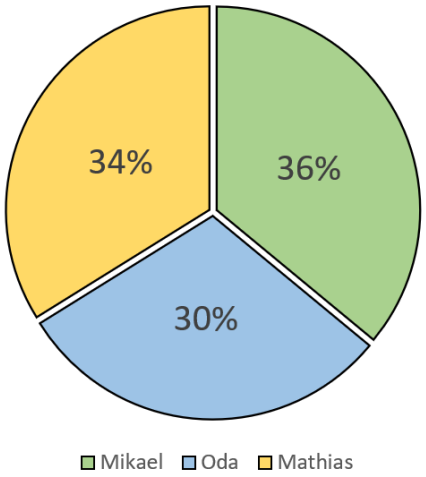


Figure 9.6: Total hour distribution between members

Most of the time put into the project was either in Programming or Report writing (Figure 9.7) activities. Understandably, these two are the most used activities since they are the most important for the thesis. We discussed whether we should split them into subcategories of the items we worked on and the chapters we were writing on, but we decided not to since it was unnecessary and would look messy in the end. We could have merged some of the meeting activities, but we wanted the separation of what were Sprint meetings and not. We are satisfied with the amount put into each activity and when we put time into them. We focused the early part of the project on Programming with some Report writing, which we are happy to have done as we did not have to write everything in the last month of the project. We have tried to be as accurate as possible, but if ten minutes were spent on how certain React functions worked, it was put into

Programming and not on Self-learning. Appendix H provides a full overview of the timekeeping.

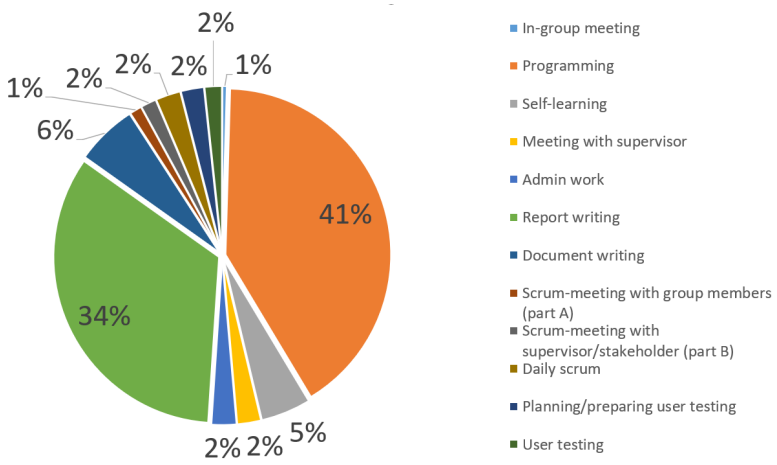


Figure 9.7: Activity distribution of all members

9.6 AI tools

During the project, we have used some AI tools. For development, we have used ChatGPT [64]. We used ChatGPT for assistance on small problems. If we were stuck on something, we would provide ChatGPT with a mock example of the code snippet we needed assistance on. Using this to explain our problem, ChatGPT would give a solution or, in most cases, point us in the right direction to look for a solution ourselves. We used it to save time looking for a solution on the internet that maybe would answer our problem, to get an answer in a relatively shorter time. We did not use ChatGPT to generate whole sections of code.

We also used a built-in plugin with Webstorm called "Full Line Code Completion," which suggests what can be written on a line after we start writing it. Most of the time, it suggested what we would write, so it was a good timesaver.

To write the report, we used Grammarly [65] as a browser extension to help with spelling and sentence structure. It would automatically change misspelt words if it were close enough, underline words that could be changed such as "were/was" mistakes, and if a sentence could be restructured to be more clear and concise. We did not use Grammarly to generate text for us but only used it as explained above to correct the meaning of what we had already written.

CHAPTER 10

FUTURE WORK

We are aware of the software's limitations. During the development, we discovered many features that, if developed, can improve this software and help the intended end users. Due to the limited time, we had to prioritise the most significant features requested by users and product owners. The software development can be continued by another person or group in a thesis next year, and these tasks can be used as a starting point for planning this development. For example, continue using it as a Product Backlog.

10.1 Memory Problem Solutions: Rewrite or Change Environment

During development, we faced a memory problem when using the software. Since it hosts a local web server to run the software, it uses a lot of RAM to function. This causes problems when loading in larger images or when trying to apply filters when there are many individual smaller images or fewer larger images on the canvas. The RAM issue causes larger images not to load in or the filtering to be very slow and stuttering.

We have discussed a couple of solutions to this problem. To try to resolve it, we have looked into better memory management and optimisation in both Konva.js and Tauri but could not find any concrete solution. If we continued working on the project with no time limit, we could put more time into looking for such a solution. Another solution we discussed is finding another programming language to rewrite the whole software. This would hopefully help with the memory issue since the web server would not take up a lot of space in the RAM. It would also help if a side effect of this were the ability to offload work from the Central Processing Unit (CPU) to the Graphical Processing Unit (GPU) when applying filters to the images.

If rewriting the whole software in another language is not feasible, we discussed

another option when keeping the software as is. There is always the option to have more RAM on the computer running the software, but the operating system and other software will always also use up a portion of it. Therefore, we then discussed a solution: to rewrite the software to not depend on Tauri to run as a desktop application but rather to have the web application uploaded to run on a dedicated server. The dedicated server could have as much RAM as needed and be accessible from any modern browser. The server should require the user to log in to use the software, and all images needed in a project would be stored locally on the server so there would not be a need to upload from the storage of a computer every time. The server could be bought and set up locally at either KHM or NTNU with the application hosted there. Or to outsource the application's hosting to services such as Amazon Web Services, Microsoft Azure, or Google Cloud Platform. These can offer more easily accessible scalability on resources the application needs rather than being fixed to the specifications of the local server.

10.2 Better Similarity Metric

There is only so much information that is possible to extract from just the histogram of the hue channel of an image. Especially when the fragments used in the images are largely monochromatic. Therefore, a metric that works better in these types of situations is something we would have liked to do.

A solution we have discussed to solve this is to integrate using either MATLAB or Python to do more complex computer vision algorithms on the fragments to extract information. Many different algorithms can be used to varying degrees of success. An algorithm performed on the fragments could look at their shapes or outlines to determine if some might fit together. It can look for patterns or symbols on a fragment and see if there are matches across different pieces. The patterns or symbols can be everything from a straight line with the same colour and thickness or depictions of humans that appear on different fragments. We do not think this would have been easy to do and would have likely taken up most of our time to get it working at a decent rate of success as automatic puzzle solving is an open problem that merits an entire PhD research [66].

Using these MATLAB or Python algorithms would most likely take a little longer to run and calculate than the current metric. Therefore, it might be necessary to reevaluate which images to use when determining the similarity. Instead of comparing the selected image with every other image, it might be better to compare just the first selected with the other selected. Determine if it should start calculating automatically when selecting images or if manual input should trigger the calculation since it can take more time.

Machine learning is an option for determining the similarity we have discussed. A problem with machine learning is the limited dataset of just over 80 fragments, which can result in low precision. Casper did his Master's thesis on this problem, and he concluded with:

> "[...] the methods are possible to be applied in very simple scenarios where the conditions of the data are top quality, however, the results

> quickly fail when the complexity increases compared to the simpler datasets. This means that the methods [...] are not ready yet for the very complex task of working with RGB images of highly degraded and damaged textiles as the information available for feature extraction is too ambiguous. Therefore, there is a knowledge gap in the machine learning field, and algorithms that will be specifically tailored to textiles are needed." [5]

Using machine learning could be possible when the tailored algorithms are made. The algorithms would need to be able to work on both the fragments in the archaeologists' current project and the next ones for it to warrant being done.

10.3 Better Filters

As mentioned, applying filters to many or large images makes the software stutter and slow. The filters used are built-in with Konva.js, but Konva.js also allows for creating custom filters. It could be worthwhile to see if creating custom filters, either with JavaScript or if MATLAB or Python could be used, would increase the speed the filters are applied and reduce the slowness. This would also give more control over the filters and allow for making more custom filters.

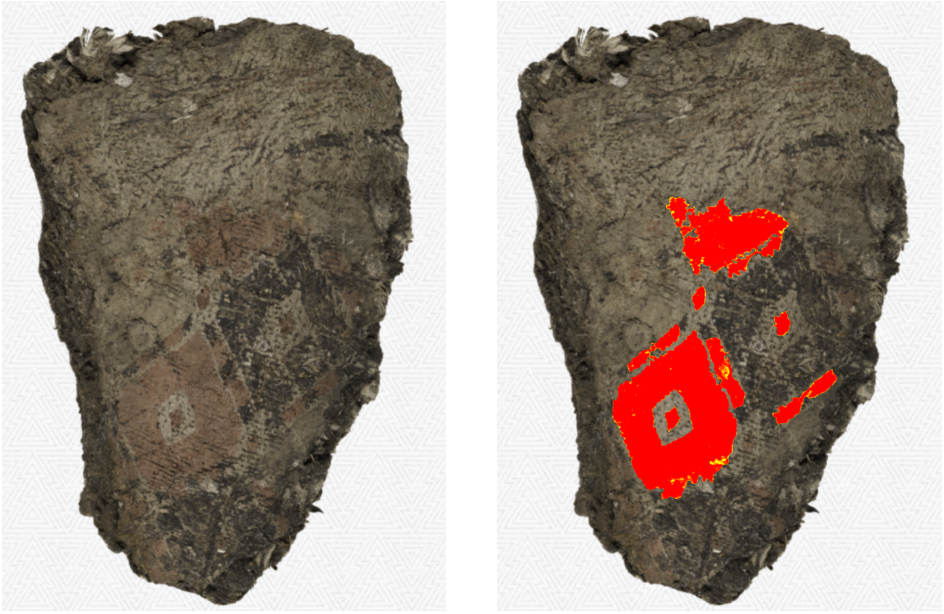
A filter feature the archaeologist told us she missed is the option to enhance and highlight a specific colour or colour range. This filter would need to be made with a custom filter inside Konva.js as it is not already built-in. The filter would look for subtle hue differences in the images and enhance only that specific part of the image whilst the rest of the image stays the same (Figure 10.1). This would allow them to easier look for shapes and figures depicted on the fragments. An example could be choosing the colour yellow, choosing the hue values between 50 and 75. Then every pixel with values in that range would be highlighted, and change colour after the user's choice. This method could also be used by the similarity metric to make the features more visible before finding similar features on other images copying the same filter settings.

Other filters that could be applied are edge detection filters [68], like Canny edge detection. This filter will enhance the edges between contrasting areas in the image. This could make perceiving the motifs on the fragments easier. One could also add some noise reduction filters [69] to enhance the motifs on the fragments. An example of a filter is the medial noise reduction filter.

One could also add algorithms to automatically enhance the colour of pieces like STRESS or contrast stretching [70]. Then the focus would be on algorithms to enhance differences in the image.

10.4 Refactor and Grouping Images Together

During this project, we tried to implement the feature of locking images into a group that could be modified as a single unit (Task 3a p.24). We spent much time trying different ways to implement this feature (Section 5.2.5: Grouping Images Together). The last method resulted in a huge refactoring of the initialisation



(a) Original **(b)** Enhanced Features

Figure 10.1: Example of a fragment with a feature enhanced to make it easier to see. Made manually with GIMP [67]

of Konva.js objects in the code. We were not capable of finishing it in the time frame we had. Some of the ideas from this version, like filtering out the images from the elements, were implemented in the final version. We will, however, keep this version of the code in a separate Git branch even when it has a lot of bugs. We hope that if someone were to take over the software development, they could finish this refactoring or use it as a reference to complete the feature of locking images together. If future developers do not use this refactoring, we suggest re-implementing a component that could represent a group of elements.

10.5 Automatic Sizes on Export Image of Canvas

During a usability test with the archaeologist and meetings with the product owner and our supervisor, there were a suggestion and discussions about how the resolution of the exported image is calculated. The suggestion is that numbers that represent the number of times the screen's pixel size is multiplied by are too advanced for regular users to understand intuitively. Therefore, options that represent low, medium, and high should be added, as stated in the usability test results. We felt using 1080p, 4K, and 8K as the final result of the images would be the best option since anything lower than 1080p would be unnecessarily low. Replacing the current numbers only requires calculating it based on the screen's size in pixels and the size of the resulting image.

1080p: $scalingNumber = \frac{1920}{screenWidth}$

4K: $scalingNumber = \frac{3840}{screenWidth}$

8K: $scalingNumber = \frac{7680}{screenWidth}$

This only accouns for screens with a size ratio of 16:9 and uses width, but to account for ultra-wide screens or screens in portrait mode, it might be better to use the lower number of either width or height. Then, use the corresponding direction when calculating the scaling number. This should ensure that the resulting resolution is not heavily affected by different types of screens.

- $scalingNumber = \frac{lowestDirectionNew}{lowestDirectionCurrent}$

The names of the resolution options do not need to be low, medium, and high but can be other names based on what feels best. The current method with numbers can be kept as an advanced option.

10.6 Metrics Table Sort

Very late in the development, we implemented a simple feature of sorting the table in increasing order of a metric when clicking on the metric header in the histogram table. One could also add the ability for the users to sort the table in decreasing order. The visuals of the headers could be improved, utilising icons common in the industry to indicate which column the table is sorted by and by which order.

10.7 Text Boxes or Notes

In the first interview, the user expressed a wish for a feature to add text boxes or comments to images or groups on the canvas. We thought this could be implemented as a separate free-floating element on the canvas that could then be locked together with other images or text boxes. Another feature could be adding text saved as some kind of metadata to each image.

10.8 Unit Tests

We did not add unit tests to the program since it has such a high percentage of GUI-centred code compared to logical code. If more logical back-end code is added in the future, it should be tested with unit tests.

10.9 Display Name and Credit of Image Files in Image Export of the Canvas

Use the filename of the image files and credit the photographers of the pictures fetched from metadata. When exporting the canvas as an image this data could be edited and be displayed in a text box or saved in the metadata. The goal is to make it easier for the user to show references in a report or similar.

10.10 Sign the Software Build

When installing the program onto a computer, Windows Defender sometimes shows a warning before it can be installed properly. This is because it can not recognise the publisher. To prevent this from happening, the code should be signed [71]. Signing the code will indicate that the program comes from a trusted source and that it has not been tampered with.

10.11 Saved Project List

The software version we started with had a component representing a list of saved projects on the landing page. This did not have any functionality but promoted a feature where the users could click on one project and open it. We did not prioritise implementing this feature, but we theorised some solutions:

Solution 1:

The first solution we discussed was to remove the option to save a project wherever a user wants. Instead, all projects would be saved into a single folder in the user's root folder, or inside the Appdata/Local folder. The user will be prompted for a name for the project. When the Landing Page is loaded, the program will look through all the JSON files there and display the names of the project files in the Projects tab.

Solution 2:

The second solution works similarly to the current solution, where users can save the projects wherever they want. The location of the file is then stored in a list. The list can be stored in a file at the same location as in the first solution or, since the program is a web application, as a cookie or a key-value pair in the browser's local storage. The downside of this solution is that the file can be moved to another folder and now the program does not know where it is. In this situation, an indicator that the file could not be found should be shown in the Projects tab.

In both these solutions, selecting a project should highlight it and keep it highlighted until another is highlighted or a project is opened. Clicking Open Project should then open the file corresponding to the selected project in the tab. A check to see if the file's name matches another that is already saved should also be implemented, and that situation should be properly handled if it occurs. Other file handling can be implemented, such as renaming or deleting files inside the program.

10.12 Dark Mode

We discussed the option of implementing a dark mode for the canvas with the archaeologist. This could be useful when there are lighter images on the canvas, and the user wants to see the contours of an image better against the background. It should be easily togglable from the canvas. The dark mode can be an inverted version of the current background.

10.13 Project Information Window

When starting a new project, a window could pop up and ask the user for information on the project, such as the project name and description. The user should also be able to open the window to edit this information from the canvas navigation bar.

10.14 Size Indicator

Showing a size indicator on the canvas to depict the real-world size of the images. Inspired by map size indicators. This will be useful to know the sizes of both an image and what is shown on it. However, this can only work if every image on the canvas is taken on the same scale and the ruler can be adjusted to match the scale.

10.15 Segmentation

Images loaded onto the canvas can be different. Some have transparent backgrounds, while some do not, or the image border is much larger than what is depicted on it. In these situations, pre-processing the images to remove the background and crop the images to fit would be great. Implementing a function to do this in an automatic or semi-automatic way would make using the images on the canvas much easier. It can be done when an image is loaded onto the canvas or another process that only does this function. The images created with this method should be downloadable to be reused later.

10.16 Image Separator

Some images consist of multiple fragments that might not belong together, or some images would benefit from having different filters on different parts of the image. So, a tool to separate an image into multiple pieces would be useful. This could be done using a free-form select, allowing users to cut the image where they want. This could even replace or enhance segmenting the image.

10.17 Confirm Save on close

When a project is closed via the file menu, a confirm close modal pops up to remind users whether they saved the project before closing. The plan was for this modal to also open when closing the whole software by pressing the exit button, which would prevent errors when accidentally pressing it. Since the exit button is a part of the application created by Tauri, the program would most likely have to interrupt the exit action and then find a way to run the JavaScript function to open the modal from the Tauri part of the code. Unlike when the modal returns to the landing page when the modal is confirmed in this situation, it should fully close the program when finished saving.

10.18 Drag select

Many software similar to ours have a drag select function in addition to the already implemented shift or ctrl select. This would improve efficiency and flexibility of use by giving users a quick way to select multiple elements on the canvas.

10.19 Drawing

To preserve the details and motifs spotted on a fragment, drawing on top of the fragment to trace what is under is a solution the archaeologists already do with Adobe Fresco using layers. This could maybe be implemented using different layers in Konva.js. The drawings would need to follow the fragment it is over when it is moved or rotated.

10.20 Pinch and Stretch Zoom

Zooming in or out on the canvas with a scroll wheel on a mouse works intuitively, but when zooming on a trackpad, the user has to put two fingers and move them up and down the same way as scrolling a news article. Adding the option to zoom with pinch and stretch on the trackpad would improve the software's intuitive usability.

10.21 Context Menu

Someone might not know about using delete or backspace to delete a selected image. Therefore, a context menu when right-clicking an image should show an option to delete the image. This context menu could also have an option to open the filter window, etc.

CHAPTER 11

CONCLUSION

In this project, a software application, Artifact Assembly, was developed. It will help archaeologists solve a puzzle of fragmented textile objects. This was accomplished by continuing development on existing software using React, Konva.js, and Tauri. It was assessed through feedback from an archaeologist to make it more useful for their work compared to the previous version. It is ready for others to continue developing the software in the future.

The results of our work met both our and the intended users' expectations. The product is a complete software usable for both archaeologists and people from the general public. With a lot of focus on usability throughout the design- and development process, the software is easy to understand and use. In the original version of the Artefact Assembly program, the user could load images, select them and move them around. After working on the software, it is now more user-friendly, has more features and gives the user many more opportunities. It contains features such as filtering the images with many different filters and getting quantitative metrics on how similar they are to the other images on the screen. Other functionalities, such as being able to undo and redo, export screenshots, and select multiple images, further enhance the user experience. The possibility to save and load a project is a feature that actually makes the software usable. Without this feature, the users had to do all the work in one go, which was extremely impractical. Now users can save, and continue later from where they stopped.

The group work was a great experience and went smoothly. Scrum provided a great structure to develop the software, and we have expanded our knowledge and experience of working in a team. It helped us improve our efficiency during the project by reviewing our workflow in the retrospective after each Sprint. Communication during the project was good, and discussions flowed easily between us as we spent most days working in the same room and could turn around to speak with each other without delay. This also meant we could work with or help one another by moving next to them. Communicating in an interdisciplinary team was also a great learning experience for us, as we had to find a common language

with archaeologists from the archaeology field and us from the computer science field.

In the work with the software, we focused mostly on increasing the existing software's usability and adding critical functions. If the software were to be worked on further, we believe that the focus should mostly be on the computer vision part and increasing the value of the similarity metrics. This would be done by making the similarity metrics more robust and providing more assistance to human users when solving a puzzle. There should also be a focus on memory optimising the software when using high-resolution images and filters.

Working on a project that will help archaeologists at KHM and potentially other institutions has been very exciting and rewarding. It can open new doors in how archaeologists work with artefacts and study them. It will greatly help with the conservation of the fragments because physical interaction will not be needed anymore to solve the puzzle. In addition to this, it can also be a starting point for more collaboration between archaeologists and computer engineers.

In conclusion, thanks to the group's hard work throughout the project, we have arrived at a product and thesis we are very proud of.

REFERENCES

[1] Ostberg R. *Oseberg ship. Encyclopedia Britannica.* June 2023. URL: https://www.britannica.com/topic/Oseberg-ship.

[2] Kulturhistorisk Museum. *TexRec – Virtual Reconstruction, Interpretation and Preservation of the Textile Artefacts from the Oseberg Find.* Jan. 2024. URL: https://www.khm.uio.no/english/research/projects/texrec/index.html.

[3] Prosjekbanken Forskningsrådet. *Virtual Reconstruction, Interpretation and Preservation of the Textile Artifacts from the Oseberg Find.* URL: https://prosjektbanken.forskningsradet.no/project/FORISS/316268 (visited on 2024-05-10).

[4] Current Research Information System In Norway. *Virtual Reconstruction, Interpretation and Preservation of the Textile Artifacts from the Oseberg Find (TexRec).* URL: https://app.cristin.no/projects/show.jsf?id=2522202 (visited on 2024-05-10).

[5] Gulbrandsen C. F. "Reconstructing the Original: Machine Learning Puzzle Assembly for Matching Archaeological Textile Fragments". MA thesis. Norwegian University of Science and Technology (NTNU), 2023.

[6] Davit Gigilashvili, Hana Lukesova, Casper Fabian Gulbrandsen, Akash Harijan, and Jon Yngve Hardeberg. *Computational techniques for virtual reconstruction of fragmented archaeological textiles.* URL: https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/3116321 (visited on 2024-05-10).

[7] Schwaber K. and Schwaber J. *The 2020 Scrum Guide.* 2020. URL: https://scrumguides.org/scrum-guide.html (visited on 2024-01-26).

[8] ThermoFisher. *What is HPLC?* URL: https://www.thermofisher.com/no/en/home/industrial/chromatography/chromatography-learning-center/liquid-chromatography-information/hplc-basics.html (visited on 2024-05-18).

[9] Edinburgh Instruments. *What is Surfaced Enhanced Raman Scattering (SERS)?* URL: https://www.edinst.com/blog/surfaced-enhanced-raman-scattering-sers/ (visited on 2024-05-18).

[10] React. URL: https://react.dev/learn (visited on 2024-05-03).

[11] Geeks For Geeks. *ReactJS Virtual DOM*. URL: https://www.geeksforgeeks.org/reactjs-virtual-dom/ (visited on 2024-05-07).

[12] React. *Built-in React Hooks*. URL: https://react.dev/reference/react/hooks (visited on 2024-05-07).

[13] Gigilashvili D., Nguyen H. T., Gulbrandsen C. F., Havgar M., Vedeler M., and Hardeberg J. Y. "Texture-based Clustering of Archaeological Textile Images". In: *Archiving Conference* (2023). URL: https://library.imaging.org/archiving/articles/20/1/29.

[14] Jakob Nielsen. *Usability 101: introduction to Usability*. Jan. 2012. URL: https://www.nngroup.com/articles/usability-101-introduction-to-usability/.

[15] Baxter K., Courage C., and Caine K. *A practical guide to user research methods: Understanding your users*. ELSEVIER, 2005.

[16] Jakob Nielsen. *10 Usability Heuristics for User Interface Design*. URL: https://www.nngroup.com/articles/ten-usability-heuristics/ (visited on 2024-05-06).

[17] Vesna Vučković. "IMAGE AND ITS MATRIX, MATRIX AND ITS IMAGE". In: *Academia.edu* (2008).

[18] Rafael C. Gonzalez and Richard E. Wood. *Digital Image Processing*. Pearson, 2018.

[19] Wikipedia. *RGBA color model*. 2023. URL: https://en.wikipedia.org/wiki/RGBA_color_model (visited on 2024-05-18).

[20] Wikipedia. *RGB Color Model*. 2024. URL: https://en.wikipedia.org/wiki/RGB_color_model (visited on 2024-05-18).

[21] Jacob Rus. *Hsl-hsv models.svg*. 2010. URL: https://commons.wikimedia.org/wiki/File:Hsl-hsv_models.svg (visited on 2024-05-16).

[22] Wikipedia. *HSL and HSV*. URL: https://en.wikipedia.org/wiki/HSL_and_HSV (visited on 2024-05-17).

[23] Scrum.org. *What is scrum?* 2024. URL: https://www.scrum.org/resources/what-scrum-module (visited on 2024-04-17).

[24] Kamal Sharma. *Top 12 Software Development Methodologies*. 2020. URL: https://www.tatvasoft.com/blog/top-12-software-development-methodologies-and-its-advantages-disadvantages/ (visited on 2024-04-17).

[25] Sayan Kumar Pal. *Waterfall Model – Software Engineering*. 2024. URL: https://www.geeksforgeeks.org/waterfall-model/ (visited on 2024-04-17).

[26] Oxygen. *What is Kanban?* 2023. URL: https://www.geeksforgeeks.org/what-is-kanban (visited on 2024-04-17).

[27] shubhammodi1403. *Kanban – Agile Methodology*. 2022. URL: https://www.geeksforgeeks.org/kanban-agile-methodology (visited on 2024-04-17).

[28] Naveen Naidu. *Agile Software Development – Software Engineering*. 2024. URL: https://www.geeksforgeeks.org/software-engineering-agile-software-development (visited on 2024-04-17).

[29] prachipaliwal1207. *What is Agile Methodology?* 2024. URL: https://www.geeksforgeeks.org/what-is-agile-methodology (visited on 2024-04-17).

[30] Github. URL: https://github.com/ (visited on 2024-04-23).

[31] Discord. URL: https://discord.com/ (visited on 2024-04-23).

[32] Microsoft. *Teams*. URL: https://www.microsoft.com/en-gb/microsoft-teams/group-chat-software (visited on 2024-04-23).

[33] Microsoft. *Outlook*. URL: https://www.microsoft.com/en-gb/microsoft-365/outlook/email-and-calendar-software-microsoft-outlook (visited on 2024-04-23).

[34] Overleaf. URL: https://www.overleaf.com/ (visited on 2024-04-23).

[35] Microsoft. *Excel*. URL: https://www.microsoft.com/en-gb/microsoft-365/excel (visited on 2024-04-23).

[36] Microsoft. *Word*. URL: https://www.microsoft.com/en-gb/microsoft-365/word (visited on 2024-04-23).

[37] Microsoft. *OneDrive*. URL: https://www.microsoft.com/en-gb/microsoft-365/onedrive/online-cloud-storage (visited on 2024-04-23).

[38] Wikipedia. *Lucidchart*. URL: https://en.wikipedia.org/wiki/Lucidchart (visited on 2024-05-17).

[39] Wikipedia. *TeamGantt*. URL: https://www.teamgantt.com/ (visited on 2024-05-17).

[40] Jetbrains. *Webstorm*. URL: https://www.jetbrains.com/webstorm/ (visited on 2024-04-23).

[41] React. URL: https://react.dev/ (visited on 2024-04-23).

[42] Gulbrandsen C. F. *Artifact Assembly Beta*. URL: https://github.com/casperfg/artifact_assembly (visited on 2024-05-13).

[43] GeeksforGeeks. *React Introduction*. 2024. URL: https://www.geeksforgeeks.org/reactjs-introduction/ (visited on 2024-04-23).

[44] KonvaJs. URL: https://konvajs.org/ (visited on 2024-04-23).

[45] Tauri. URL: https://tauri.app/ (visited on 2024-04-23).

[46] KelvinS. *Properly comparing two histograms*. URL: https://math.stackexchange.com/questions/2383969/properly-comparing-two-histograms (visited on 2024-05-13).

[47] OpenCV. *Tutorial Histogram Comparison*. URL: https://math.stackexchange.com/questions/2383969/properly-comparing-two-histograms (visited on 2024-05-13).

[48] YoavAvneon. *Bhattacharyya distance: From statistics to application in data science*. URL: https://medium.com/@yoavyeledteva/bhattacharyya-distance-from-statistics-to-application-in-data-science-8eb5ccdbba62 (visited on 2024-05-13).

[49] Vedeler M. *Oseberg - de gåtefulle billedvevene*. Scandinavian Academic Press, 2019.

[50] Jakob Nielsen. *Thinking Aloud: The #1 Usability Tool*. 2012. URL: https://www.nngroup.com/articles/thinking-aloud-the-1-usability-tool/ (visited on 2024-04-29).

[51] think.design. *Concurrent Probing*. URL: https://think.design/user-design-research/concurrent-probing/ (visited on 2024-04-29).

[52] think.design. *Retrospective Probing*. URL: https://think.design/user-design-research/retrospective-probing/ (visited on 2024-04-29).

[53] Pixabay. *Houses Cliff Sea*. 2019. URL: https://pixabay.com/photos/houses-cliff-sea-italy-4093227/ (visited on 2024-04-25).

[54] Postccron. *ImageSplitter*. URL: https://postcron.com/image-splitter/en/ (visited on 2024-04-24).

[55] Pixabay. *Colourful Painting Background*. 2017. URL: https://pixabay.com/photos/colorful-painting-background-2468874/ (visited on 2024-04-25).

[56] Tauri. URL: https://tauri.app/v1/guides/getting-started/prerequisites/ (visited on 2024-05-03).

[57] Microsoft. *Microsoft C++ Build Tools*. URL: https://visualstudio.microsoft.com/visual-cpp-build-tools/ (visited on 2024-05-03).

[58] Microsoft. *Microsoft Edge WebView2*. URL: https://developer.microsoft.com/en-us/microsoft-edge/webview2?form=MA13LH (visited on 2024-05-03).

[59] Rust.lang. *Install Rust*. URL: https://www.rust-lang.org/tools/install (visited on 2024-05-03).

[60] NodeJS. *Node.js - Run JavaScript Everywhere*. URL: https://nodejs.org/en (visited on 2024-05-03).

[61] United Nations. *Goal 12 - Ensure sustainable consumption and production patterns*. URL: https://sdgs.un.org/goals/goal12#targets_and_indicators (visited on 2024-05-15).

[62] United Nations. *Goal 4 - Ensure inclusive and equitable quality education and promote lifelong learning opportunities for all*. URL: https://sdgs.un.org/goals/goal4#targets_and_indicators (visited on 2024-05-15).

[63] United Nations. *Goal 9 - Build resilient infrastructure, promote inclusive and sustainable industrialization and foster innovation*. URL: https://sdgs.un.org/goals/goal9#targets_and_indicators (visited on 2024-05-15).

[64] OpenAI. *ChatGPT*. URL: https://chatgpt.com/ (visited on 2024-05-16).

[65] Grammarly. *Responsible AI that ensures your writing and reputation shine*. URL: https://www.grammarly.com/ (visited on 2024-05-16).

[66] Marie-Morgane Paumard. “Solving Jigsaw Puzzles with Deep Learning for Heritage”. PhD thesis. cy cergy paris université, 2020.

[67] GIMP. *GIMP - Gnu Image Manipulation Program.* URL: https://www.gimp.org/ (visited on 2024-05-16).

[68] Wikipedia. *Edge detection.* URL: https://en.wikipedia.org/wiki/Edge_detection (visited on 2024-05-16).

[69] Wikipedia. *Noise reduction.* URL: https://en.wikipedia.org/wiki/Noise_reduction (visited on 2024-05-16).

[70] Øyvind Kolås, Ivar Farup, and Alessandro Rizzi. “Spatio-Temporal Retinex-Inspired Envelope with Stochastic Sampling: A Framework for Spatial Color Algorithms”. In: *NTNU Open* (2011), pp. 1–10.

[71] Venafi.com. *What Is Code Signing?* URL: https://venafi.com/machine-identity-basics/what-is-code-signing/#item-0 (visited on 2024-05-10).

APPENDICES:

APPENDIX A

STANDARD AGREEMENT

NTNU

Norges teknisk-naturvitenskapelige universitet

Fastsatt av prorektor for utdanning 10.12.2020

STANDARDAVTALE

om utføring av studentoppgave i samarbeid med ekstern virksomhet

Avtalen er ufravikelig for studentoppgaver (heretter oppgave) ved NTNU som utføres i samarbeid med ekstern virksomhet.

Forklaring av begrep

Opphavsrett
Er den rett som den som skaper et åndsverk har til å fremstille eksemplar av åndsverket og gjøre det tilgjengelig for allmennheten. Et åndsverk kan være et litterært, vitenskapelig eller kunstnerisk verk. En studentoppgave vil være et åndsverk.

Eiendomsrett til resultater
Betyr at den som eier resultatene bestemmer over disse. Utgangspunktet er at studenten eier resultatene fra sitt studentarbeid. Studenten kan også overføre eiendomsretten til den eksterne virksomheten.

Bruksrett til resultater
Den som eier resultatene kan gi andre en rett til å bruke resultatene, f.eks. at studenten gir NTNU og den eksterne virksomheten rett til å bruke resultatene fra studentoppgaven i deres virksomhet.

Prosjektbakgrunn
Det partene i avtalen har med seg inn i prosjektet, dvs. som vedkommende eier eller har rettigheter til fra før og som brukes i det videre arbeidet med studentoppgaven. Dette kan også være materiale som tredjepersoner (som ikke er part i avtalen) har rettigheter til.

Utsatt offentliggjøring
Betyr at oppgaven ikke blir tilgjengelig for allmennheten før etter en viss tid, f.eks. før etter tre år. Da vil det kun være veileder ved NTNU, sensorene og den eksterne virksomheten som har tilgang til studentarbeidet de tre første årene etter at studentarbeidet er innlevert.

1. Avtaleparter

Norges teknisk-naturvitenskapelige universitet (NTNU) Institutt: Institutt for datateknologi og informatikk (IDI)
Veileder ved NTNU: Jon Yngve Hardeberg e-post og tlf.: jon.hardeberg@ntnu.no, +47 98216899
Ekstern virksomhet: Ekstern virksomhet sin kontaktperson, e-post og tlf.: Davit Gigilashvili, davit.gigilashvili@ntnu.no, +47 40748687
Student: Mikael Røv Mathiassen Fødselsdato: 06.09.1999
Student: Oda Katrine Steinsholt Fødselsdato: 12.03.2000
Student: Mathias Iversen Fødselsdato: 03.03.1995

Partene har ansvar for å klarere eventuelle immaterielle rettigheter som studenten, NTNU, den eksterne eller tredjeperson (som ikke er part i avtalen) har til prosjektbakgrunn før bruk i forbindelse med utførelse av oppgaven. Eierskap til prosjektbakgrunn skal fremgå av eget vedlegg til avtalen der dette kan ha betydning for utførelse av oppgaven.

2. Utførelse av oppgave

Studenten skal utføre: (sett kryss)

Masteroppgave	
Bacheloroppgave	x
Prosjektoppgave	
Annen oppgave	

Startdato: 10.01.2024
Sluttdato: 21.05.2024

Oppgavens arbeidstittel er: Software for Virtual Puzzle Reassembly

Ansvarlig veileder ved NTNU har det overordnede faglige ansvaret for utforming og godkjenning av prosjektbeskrivelse og studentens læring.

3. Ekstern virksomhet sine plikter

Ekstern virksomhet skal stille med en kontaktperson som har nødvendig faglig kompetanse til å gi studenten tilstrekkelig veiledning i samarbeid med veileder ved NTNU. Ekstern kontaktperson fremgår i punkt 1.

Formålet med oppgaven er studentarbeid. Oppgaven utføres som ledd i studiet. Studenten skal ikke motta lønn eller lignende godtgjørelse fra den eksterne for studentarbeidet. Utgifter knyttet til gjennomføring av oppgaven skal dekkes av den eksterne. Aktuelle utgifter kan for eksempel være reiser, materialer for bygging av prototyp, innkjøp av prøver, tester på lab, kjemikalier. Studenten skal klarere dekning av utgifter med ekstern virksomhet på forhånd.

Ekstern virksomhet skal dekke følgende utgifter til utførelse av oppgaven:

Dekning av utgifter til annet enn det som er oppført her avgjøres av den eksterne underveis i arbeidet.

4. Studentens rettigheter

Studenten har opphavsrett til oppgaven[1]. Alle resultater av oppgaven, skapt av studenten alene gjennom arbeidet med oppgaven, eies av studenten med de begrensninger som følger av punkt 5, 6 og 7 nedenfor. Eiendomsretten til resultatene overføres til ekstern virksomhet hvis punkt 5 b er avkrysset eller for tilfelle som i punkt 6 (overføring ved patenterbare oppfinnelser).

I henhold til lov om opphavsrett til åndsverk beholder alltid studenten de ideelle rettigheter til eget åndsverk, dvs. retten til navngivelse og vern mot krenkende bruk.

Studenten har rett til å inngå egen avtale med NTNU om publisering av sin oppgave i NTNUs institusjonelle arkiv på Internett (NTNU Open). Studenten har også rett til å publisere oppgaven eller deler av den i andre sammenhenger dersom det ikke i denne avtalen er avtalt begrensninger i adgangen til å publisere, jf. punkt 8.

5. Den eksterne virksomheten sine rettigheter

Der oppgaven bygger på, eller videreutvikler materiale og/eller metoder (prosjektbakgrunn) som eies av den eksterne, eies prosjektbakgrunnen fortsatt av den eksterne. Hvis studenten skal utnytte resultater som inkluderer den eksterne sin prosjektbakgrunn, forutsetter dette at det er inngått egen avtale om dette mellom studenten og den eksterne virksomheten.

Alternativ a) (sett kryss) Hovedregel

[1] Jf. Lov om opphavsrett til åndsverk mv. av 15.06.2018 § 1

	Ekstern virksomhet skal ha bruksrett til resultatene av oppgaven

Dette innebærer at ekstern virksomhet skal ha rett til å benytte resultatene av oppgaven i egen virksomhet. Retten er ikke-eksklusiv.

Alternativ b) (sett kryss) Unntak

x	Ekstern virksomhet skal ha eiendomsretten til resultatene av oppgaven og studentens bidrag i ekstern virksomhet sitt prosjekt

Begrunnelse for at ekstern virksomhet har behov for å få overført eiendomsrett til resultatene: NTNU Colour Labben skal fortsette og arbeide på programmet som arkeologene på UiO skal bruke etter bachelor oppgaven er fullført. Det vil være enklere å fortsette dette arbiedet for dem hvis det er de som har eierskap av programmet.

6. Godtgjøring ved patenterbare oppfinnelser

Dersom studenten i forbindelse med utførelsen av oppgaven har nådd frem til en patenterbar oppfinnelse, enten alene eller sammen med andre, kan den eksterne kreve retten til oppfinnelsen overført til seg. Dette forutsetter at utnyttelsen av oppfinnelsen faller inn under den eksterne sitt virksomhetsområde. I så fall har studenten krav på rimelig godtgjøring. Godtgjøringen skal fastsettes i samsvar med arbeidstakeroppfinnelsesloven § 7. Fristbestemmelsene i § 7 gis tilsvarende anvendelse.

7. NTNU sine rettigheter

De innleverte filer av oppgaven med vedlegg, som er nødvendig for sensur og arkivering ved NTNU, tilhører NTNU. NTNU får en vederlagsfri bruksrett til resultatene av oppgaven, inkludert vedlegg til denne, og kan benytte dette til undervisnings- og forskningsformål med de eventuelle begrensninger som fremgår i punkt 8.

8. Utsatt offentliggjøring

Hovedregelen er at studentoppgaver skal være offentlige.

Sett kryss

x	Oppgaven skal være offentlig

I særlige tilfeller kan partene bli enige om at hele eller deler av oppgaven skal være undergitt utsatt offentliggjøring i maksimalt tre år. Hvis oppgaven unntas fra offentliggjøring, vil den kun være tilgjengelig for student, ekstern virksomhet og veileder i denne perioden. Sensurkomiteen vil ha tilgang til oppgaven i forbindelse med sensur. Student, veileder og sensorer har taushetsplikt om innhold som er unntatt offentliggjøring.

Oppgaven skal være underlagt utsatt offentliggjøring i (sett kryss hvis dette er aktuelt):

Sett kryss		Sett dato
	ett år	
	to år	
	tre år	

Behovet for utsatt offentliggjøring er begrunnet ut fra følgende:

Dersom partene, etter at oppgaven er ferdig, blir enig om at det ikke er behov for utsatt offentliggjøring, kan dette endres. I så fall skal dette avtales skriftlig.

Vedlegg til oppgaven kan unntas ut over tre år etter forespørsel fra ekstern virksomhet. NTNU (ved instituttet) og student skal godta dette hvis den eksterne har saklig grunn for å be om at et eller flere vedlegg unntas. Ekstern virksomhet må sende forespørsel før oppgaven leveres.

De delene av oppgaven som ikke er undergitt utsatt offentliggjøring, kan publiseres i NTNUs institusjonelle arkiv, jf. punkt 4, siste avsnitt. Selv om oppgaven er undergitt utsatt offentliggjøring, skal ekstern virksomhet legge til rette for at studenten kan benytte hele eller deler av oppgaven i forbindelse med jobbsøknader samt videreføring i et master- eller doktorgradsarbeid.

9. Generelt

Denne avtalen skal ha gyldighet foran andre avtaler som er eller blir opprettet mellom to av partene som er nevnt ovenfor. Dersom student og ekstern virksomhet skal inngå avtale om konfidensialitet om det som studenten får kjennskap til i eller gjennom den eksterne virksomheten, kan NTNUs standardmal for konfidensialitetsavtale benyttes.

Den eksterne sin egen konfidensialitetsavtale, eventuell konfidensialitetsavtale den eksterne har inngått i samarbeidprosjekter, kan også brukes forutsatt at den ikke inneholder punkter i motstrid med denne avtalen (om rettigheter, offentliggjøring mm). Dersom det likevel viser seg at det er motstrid, skal NTNUs standardavtale om utføring av studentoppgave gå foran. Eventuell avtale om konfidensialitet skal vedlegges denne avtalen.

Eventuell uenighet som følge av denne avtalen skal søkes løst ved forhandlinger. Hvis dette ikke fører frem, er partene enige om at tvisten avgjøres ved voldgift i henhold til norsk lov. Tvisten avgjøres av sorenskriveren ved Sør-Trøndelag tingrett eller den han/hun oppnevner.

Denne avtale er signert i fire eksemplarer hvor partene skal ha hvert sitt eksemplar. Avtalen er gyldig når den er underskrevet av NTNU v/instituttleder.

Signaturer:

Instituttleder: v Emneansvarlig Dato: 5/2-24
Veileder ved NTNU: Dato: 25.1.2024
Ekstern virksomhet: Dato: 25.01.2024
Student: Mikael Røv Mathiassen Dato: 23.01.2024
Student: Oda K. F. Stenshorne Dato: 23.01.2024
Student: Mathias Wenzer Dato: 24.01.2024

APPENDIX B

TASK DESCRIPTION

Oppgave 42 BPROG, BIDATA, DIGSEC 2-4 stk

Oppgavetittel: Software for Virtual Puzzle Reassembly

Bedrift:	NTNU Colourlab
Kontaktperson:	Davit Gigilashvili
E-post:	Davit.gigilashvili@ntnu.no
Telefon:	40748687
Lokasjon:	Gjøvik

Beskrivelse av oppgaven

Solving jigsaw puzzles is a fun pastime activity. In addition to this, it has a very significant practical implications in archaeology and cultural heritage. Archaeologists need to solve a puzzle of fragmented historical artifacts to reconstruct their original appearance. However, some artifacts are extremely fragile, and we cannot actively interact with them. Therefore, virtual puzzle solvers are needed to make puzzle solving possible without damaging the actual fragments.

This project entails a development of a software that will have several fundamental features: first, it should allow loading several images and manually moving them around on a virtual canvas to try different arrangements; second, it should allow different color enhancement and contrast manipulation operations to make some interesting patterns more visible; and third, it should measure different color and texture statistics of the images and provide a similarity score or a suggestion to the user about which fragments are similar and hence, likely to belong together. We do not expect a fully automatic solution of a puzzle. The puzzle should be solved by humans, but the software should assist humans in this process by giving relevant suggestions.

The project is in collaboration with archaeologists from the University of Oslo. As the intended end users, they will continuously provide the feedback on software's functionalities and user-friendliness. As a case study, we will test the software with the actual open unsolved puzzle problem of Oseberg Tapestry from the Viking Age.

The project can be supervised by Jon Yngve Hardeberg.

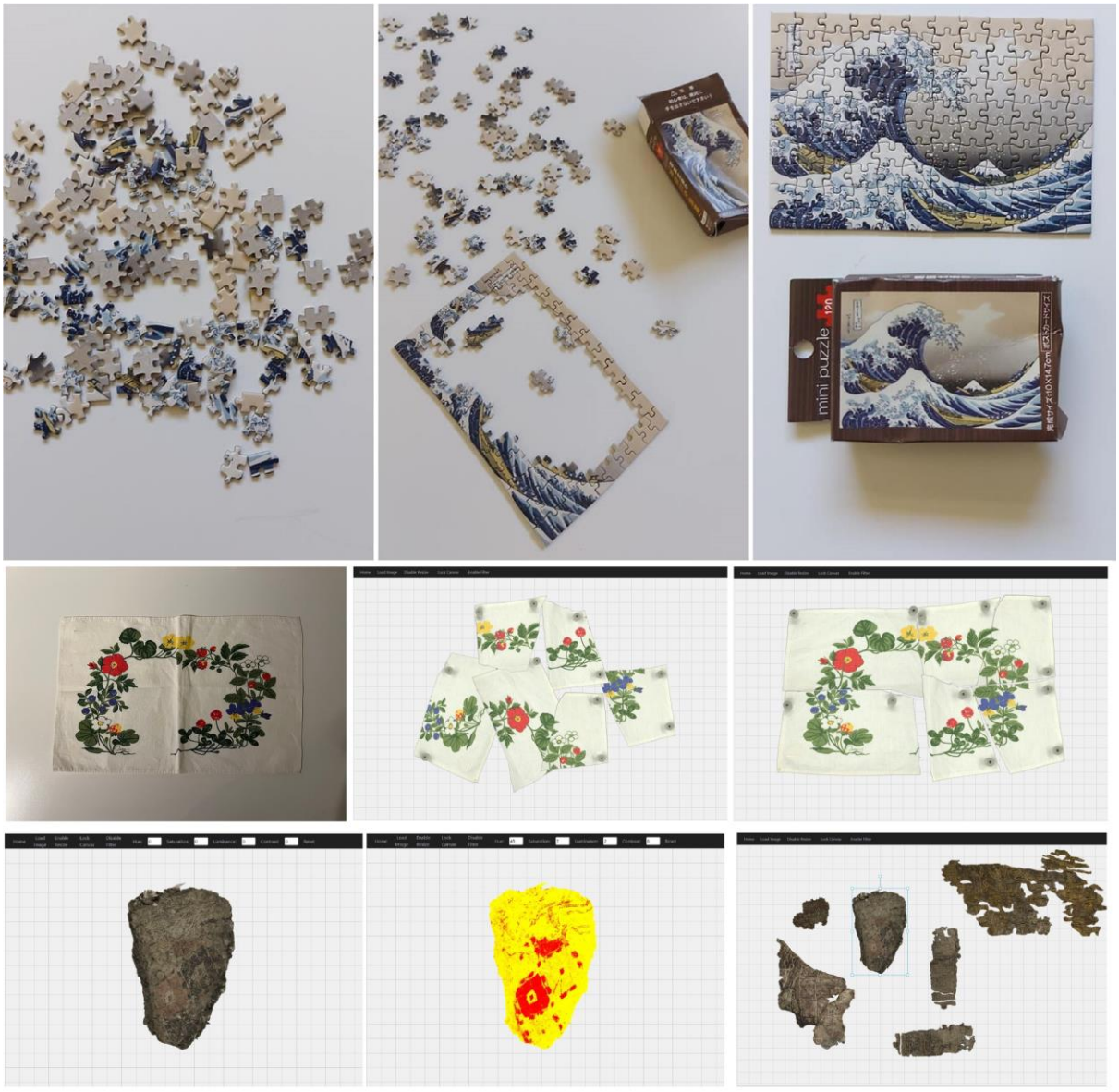
We have a preliminary version of the software with limited capabilities. The new solution can be developed either from scratch or be based on the existing program.

References/Possible literature:

[1] Gulbrandsen CF. Reconstructing the Original: Machine Learning Puzzle Assembly for Matching Archaeological Textile Fragments. Master Thesis. Norwegian University of Science and Technology; 2023 https://hdl.handle.net/11250/3079162

[2] Gigilashvili D, Nguyen HT, Gulbrandsen CF, Havgar M, Vedeler M, Hardeberg JY. Texture-based clustering of archaeological textile images. The Proceedings of Archiving 2023 Conference. vol. 20. IS&T; 2023. p. 139–142.

[3] Vedeler M. The Oseberg Tapestries. Scandinavian Academic Press Oslo; 2019.



APPENDIX C

PROJECT PLAN

NTNU
Kunnskap for en bedre verden

Department of Computer Science

IDATG2900 - Bacheloroppgave

Virtual Puzzle Reassembly Project Plan

Authors:
Oda Katrine Steinsholt
Mathias Iversen
Mikael Røv Mathiassen

Date: 31.01.2024

Table of Contents

List of Figures

List of Tables

1 Introduction

This is the project plan for the bachelor thesis, Virtual Puzzle Reassembly that was made by Davit Gigilashvili at the Colour Lab at NTNU (Norwegian University of Science and Technology) in Gjøvik in collaboration with archaeologists at UiO (University of Oslo), and will be done by the computer science students at NTNU in Gjøvik; Oda Steinsholt, Mikael Røv Mathiassen, and Mathias Iversen.

1.1 Inspiration

This project plan has taken inspiration from the project plans of these groups:

- "Viten i senter" by Steinar Opphus, Ole A. Slettum, and Steffen Granberg [1]
- "Securing the Software Development Life Cycle" by Anniken Arildset, Celina H. Brynhildsen, Sebastian Hestsveen, and Thea Urne [2]
- "Fish Detection in Underwater Video" by Benjamin L. Bjerken, Lars B. Holter, Daniel H. Huynh, and Lillian A. Wangerud [3]

2 Goals and Restrictions

2.1 Background

Osebergskipet is a Viking Ship that was found in 1904 in the Norwegian town, Tønsberg. It is 22 meters long, and one of the most famous Viking ships found. Archaeologists believe that the ship was built between the year 815 AD and 820 AD. It was buried in a large pile of dirt in 834 AD, along with two women.

The Oseberg Tapestries are fragments of textile found inside the Oseberg ship. These fragments are believed to belong to several different textile objects, with a big variety of patterns and fine granular details.

At UiO, archaeologists are now working with these tapestries from the Oseberg ship.[4] They are trying to recover and interpret the motifs, to uncover the stories that these tapestry objects are telling to know more about the life of the people that lived back then. The tapestries are very fragmented and they need to find matching pieces to recover the original tapestries, which resembles puzzle solving. Doing this work manually is time consuming, and since the pieces are old and fragile they can easily fall apart if being handled incorrectly. It is also hard to see all the patterns on the fragments with the human eye.

Therefore the Colour Lab at NTNU in Gjøvik together with the archaeologists at UiO want to make a software that can aid the archaeologists in this process. With this software one can load in images of the fragments and use them in a virtual puzzle solver canvas to rotate, move them around, and examine the composition of the tapestry pieces. The software will also have functionalities to aid with color enhancement and also give a similarity score based on different computer vision algorithms. Last year the master student Casper Fabian Guldbrandsen started on the development of such a software in his master thesis.[5] The group will continue his work.

2.2 Project goals

The goals of this project is divided into performance, result, and learning goals. The performance goals are the performance targets that we have set for ourselves to reach the desired end-result or outcome of the final product. The result goals are the specific targets that we aim to achieve by the end of the project. The learning goals are the knowledge we want to gain during the project and what we hope to retain after the project is over.

2.2.1 Performance goals

We will:

- Work hard and efficiently to achieve the best possible result for the NTNU Colour Lab, so that they can continue the work they are doing to help the archaeologists at UiO.
- Make sure that the project is finished in time.
- As a group, work towards enhancing our cooperation and communication, to have the best experience for our members throughout the project and streamline the workflow.

2.2.2 Result goals

We aim to:

- Continue development on the Artifact Assembly program that was started by Casper Fabian Gulbrandsen for his Master thesis last year.[5] And make a software that the NTNU Colour Lab can continue to develop further.

- Create and implement as much of the features that the archaeologists want and outlines for us in meetings, user tests and other ways of feedback. In this way make a software that is useful and easy to use for the archaeologists. The software will help with the conservation of archaeological artifacts like the fragments of the Oseberg tapestry, by being a substitution that will minimize the physical interaction on the artifacts.
- Achieve a grade that is satisfactory for the group. The group will aim for the character A or B.

2.2.3 Learning goals

We hope to:

- Improve our teamwork skills in the context of a bigger project, by getting a better knowledge about the Software Development Life Cycle (SDLC) and the SCRUM method.
- Learn more about using HTML, CSS, and JavaScript to create a program instead of a website, using Python in a computer vision setting and using RUST and the Tauri dependency.
- Learn how to continue development on software that is already partially developed.
- Learn more about development of a user centered graphical user interface through user communications and user testing. During the project also learn a little about archaeology and their process when trying to piece together the tapestries from the Oseberg ship.
- Learn more about computer vision, basic machine learning and different algorithms and pipelines like clustering to best compare to rgb-images of pieces of fabric. [6]
- Learn more about color enhancements of rgb-images, and general image processing.

2.3 Schema

The software will need to be a standalone program that can run on a Windows system as this was given as a requirement by the users, the archaeologists. The program is written in the programming languages HTML, CSS, JavaScript, and a little Rust using the Tauri framework, so we will continue development with this as a foundation.

2.3.1 Time frame

- This project plan has to be finished, signed, and delivered by the end of the 31st of January 2024.
- The bachelor thesis has to be finished, and delivered by the 21st of May 2024.
- The bachelor thesis will be presented the 5th or the 6th of June 2024.

3 Scope

3.1 Subject field

Our client, the NTNU Colour Lab in Gjøvik is working with archaeologists at UiO. At UiO they are working on the Oseberg tapestry from the Viking Age. As the tapestry is approximately 1200 years old, it is extremely fragile. When found, it was already fragmented into many pieces. From talking to one of the archaeologists we got to know that from the Oseberg ship, their hypothesis is that it was three different tapestries that were originally hung up. For now it is difficult to test this though, since when the ship was dug up, the documentation of which piece belong to which tapestry was bad or non-existent.
Their process right now is to take really high quality pictures of each piece and look at them manually. Using process of elimination and comparison to try to find pieces that can belong to each other and which tapestry. To identify matching pieces, they rely on several technical features of the weave, such as:

- What is the thread count per centimeter, to see if they are woven at the same time or loom.
- How is the thread spun compared to other pieces.

Besides, they also rely on high level motifs and identification of similar patterns, like how the motif is structured, and if there are objects or a repeating pattern that can belong together.
This is work that has been done on individual pieces, but as the pieces are so old and fragile, they can not be manipulated or moved around much. The program we are continuing the work on is to help the archaeologists compare the pieces side by side and to incorporate some features to help them do so.

3.2 Delimitation

To make sure that the bachelor thesis is doable in a timely manner, we have determined some elements that will not be implemented to narrow it down:

- As written in the task description that it is not expected, we will not create a fully automatic solution of a puzzle.
- We will not create a new solution from scratch, and rather continue working on the preexisting program.
- The goal of the project is not to create a program that incorporates all features discussed with the users, but rather to add as many features as possible to make program more useful, usable and reliable.

3.3 Task description

Our project will be to continue development on the Artifact Assembly program so the work for the archaeologists will be easier, and to have another platform to gather useful data about the pieces. The program should continue to be easy to understand and use for the users. Features that one of the archaeologists have indicated they would like are:

- Saving and loading of the canvas and the possibility to have multiple canvas projects. The program has currently no way to save and when closing the program everything is lost. This is a high priority feature, because puzzle solving may take days, weeks, or months. Without saving and loading everything needs to be done in one go, which make the software very inconvenient to use.

- Importing and exporting. The project could be saved to a file that can be shared. This file will contain references to the images that are on the canvas their transformation: position, rotation, scaling, their image enhancements, and if they are locked together to other pieces and other info about the state of the canvas. The file will not contain the images themselves to limit the size of the exported files.
- Export images of the canvas.
- Adding a text-box/note to a piece image.
- Using computer vision and basic machine learning to create scores/numbers for each piece to indicate what might fit together or not. Using histograms, colours, shape of the piece, shape of the motifs on the piece, etc...
- Replace colour filter that currently uses numbers, with a colour wheel or spectrum to make it more intuitive and user friendly. This should be enabled or disabled with a button.
- An algorithm to enhance the colours of the pieces automatically.
- Button to convert the pieces to binary (black and white) or grey-scale images.
- Select individual or multiple pieces and only affect those with the colour filter, using a drag and select function, or Ctrl clicking.
- Undo and redo buttons in the graphical user interface. Only works with Ctrl+z or Ctrl+y for now.
- Crediting the photographer of the pictures. This is not a high priority.
- Drawing on top of the fragments on another layer in multiple colours. Sketching is an important step for the archaeologists to interpret the motifs.
- Separating a piece into multiple pieces. Some pictures are of multiple pieces, but their location might not be correct and some needs to have different enhancements on different parts of the piece.
- Combining/locking two or more pieces together.
- Automatic or semi-automatic segmentation of piece images when loading them in.
- Size indicator, like a ruler on the canvas to compare the pieces to real world size.
- Ability to use high resolution images that would scale so that they can be used with ease. The image files can be 1GB large, so some memory optimization could be needed.

This feature list is the starting point for our product back log that we will use in the Scrum process, and will be added to and changed according to users wants.

4 Project Organization

4.1 Responsibility areas and roles

- Group leader/SCRUM master (Mathias Iversen)
  - Follows up with every member and ensures that everyone contributes to the project, and that the project plan is followed.
  - Responsible for solving disputes within the group, and mediate between members if the need arises.
  - Summons all members to scrum meetings.
  - Leads the scrum meetings and follows up that everything on the agenda is brought up.
  - Responsible for teaching and implementing the Scrum framework according to the Scrum Guide [7]
- Secretary and Communications (Oda Katrine Steinsholt)
  - Writes meeting minutes during meetings.
  - Follows up on communication in and out of the group.
- Documentation (Mikael Røv Mathiassen)
  - Makes sure that all necessary documents are made, placed correctly and structured.
- Sources and quality assurance (Mikael Røv Mathiassen, Oda Katrine Steinsholt)
  - Makes sure that everything that need sources have a source and that it is written and structured properly.
  - Goes over the written materials and fixes typographical errors, and that the text is structured correctly.
- Developers (Mathias Iversen, Mikael Røv Mathiassen, Oda Katrine Steinsholt)
  - Will do the work assigned to them and follow code standards.
- Supervisor (Jon Yngve Hardeberg)
  - Will supervise the group and help us do the best to our abilities.
- Product owner (Davit Gigilashvili)
  - Represents the owner of the program after the project is finished.
  - Responsible for effective management of product backlog i the Scrum. Creating and defining new backlog items. Ordering backlog items.
- Users (UiO Archaeologists)
  - Will do the user tests of the program and give feedback of what they think.

4.2 Routines

- Every group member should have a target to work 30 hours on average per week.
- There will be a sprint meeting at 14.00 on Thursdays every other week. The other weeks, this will be a summary meeting of what happened last week, and planning the next.
- There will be a meeting at 15:15 every Thursday with the supervisor and the external contact. This will be a continuation of the sprint meeting on the weeks they are occurring.

- The time worked will be logged in the excel spreadsheet along with the activity the member did right after the activity is done. At the end of the week, each group member will review their time sheet to see if they missed something.
- For every project- or status meeting, meeting minutes will be written.
- Discord and Outlook will be the communication channels that the group members will use.
- As much as possible, the group members will try to work in the same space together.

4.3 Group Rules

- If a group member is late to a meeting or group working sessions, they will need to notify the other members as soon as they notice they will be late, so that the other members can plan accordingly.
  If a member shows up late and fails to notify the others, they will have to buy one 4-pack of Aunt Mabels Milk Chocolate Cookies for the next time the group meets.
- Longer leaves of absence will have to be notified about with a 24h notice prior to the leave of absence. With exceptions for sudden sickness, but proof of said sickness must be shared internally on discord.

5 Planning, Follow-up and Reporting

5.1 Scrum

In this project there will be implemented an agile Scrum method defined by the Scrum Guide to the best of our ability.[7] This development method was chosen because it allows for the flexibility that we need. Changes may occur to the tasks to fulfill the requirements described in the product backlog, due to the user centered focus of the software.

The sprints will have a length of 2 weeks resulting in seven sprints during the project. This length is because of the short time frame we have on this project. The short sprints will also allow for several adjustments of the product back log by the product owner, and allow the scrum team to learn quicker from the sprint retrospective. Any shorter sprints would result in more overhead in sprint meetings than would be beneficial.

At the beginning of every day the developers along with the Scrum master, will have a daily Scrum meeting where they will discuss what they have done since the last meeting, possible problems they have, and what they plan to do until the next meeting. They will have an transparent communication that allows for inspection of the work they do. They will then adapt according to each others feedback.

5.2 Plan for status meetings

There is planned to be weekly meetings on Thursdays with the supervisor, Jon Yngve Hardeberg and the product owner, Davit Gigilashvili. These meetings may change time and frequency.

Every other of these meeting would be a sprint meeting that would mark the end of the sprint and the start of a new sprint. The sprint meeting will contain two parts, one with just the developers (part a), and the next including the stakeholder and the supervisor (part b).

Part a will start with the sprint retrospective. This contains talking about what worked well and what did not. The team will also talk about changes that can improve the work. Part a will also contain some initial planning for the next sprint, by choosing tasks from the product backlog and make a sprint backlog for the sprint.

Part b will contain the sprint review, which means that the meeting attendants will talk about the results of the work that has been done in the previous sprint. The main part of the sprint planning - for the next sprint - will also be done in this part of the meeting. Here the product backlog changes may be discussed. These events are described in the scrum guide. [7]

6 Organization of Quality Assurance

6.1 Documentation

We will use the tools that are mentioned below. Documents will be shared using OneDrive. Smaller documents will be written in Microsoft Word and Microsoft Excel. This can be meeting minutes, notes, and other documents that will be written during development. Bigger documents or deliverables will be written in LaTeX using Overleaf. The code that we write will be well documented using comments in a uniform style to make the code more readable and make it easier for later developers to continue the development of the software.

6.2 Tools

Name	Type	Usage
Discord	Communication platform that offers voice, video, and text chat, primarily used by gamers but also widely adopted for various communities and collaboration	Communication primarily inside the group
GitHub	Web-based platform for version control and collaborative software development, enabling developers to host, review, and manage code repositories	Version control of the code
Microsoft Excel	Spreadsheet software that allows users to organize, analyze, and visualize data using tabular grids, formulas, and charts	Time table
Microsoft Word	Word processing application that allows users to create, edit, and format text documents	Writing smaller documents, meeting minutes
Node.js	Open-source, cross-platform JavaScript runtime environment that allows developers to execute server-side JavaScript code, enabling the development of scalable and high-performance network applications	Running JavaScript code
OneDrive	Cloud-based file hosting service by Microsoft that enables users to store and share files, documents, and photos across devices	Sharing documents and files
Outlook	Personal information manager software developed by Microsoft, commonly used for email communication, calendar organization, task management, and contact tracking	Communication primarily out of the group
Overleaf	Online collaborative platform for writing and editing LaTeX documents	Writing the report and project plan
Tauri	Open-source framework that facilitates the creation of highly customizable and lightweight desktop applications using web technologies such as HTML, CSS, and JavaScript, while providing native integration and performance	In the software of our project
Team Gantt	Project management tool that provides collaborative Gantt chart software for planning, scheduling, and managing projects with teams	Making Gantt chart

Visual Studio Code	Lightweight and versatile source code editor developed by Microsoft, supporting various programming languages and featuring powerful extensions for enhanced development workflows	Coding

Table 1: Tools that we will use during the project.

6.3 Plan for inspections and testing

Our code will be uniform and follow a single coding style. We will be using a linting analyzer in our editor and in this way use less time and energy on making sure all code follows the same coding style. This will involve a uniform use of line space, number of tabs, placement of parentheses, and variable naming style. This will make the code more uniform and readable for future developers. During development we will write unit test wherever possible to quality check our code. We will also test the graphical user interface when implementing a new feature.

6.4 Risk analysis

Underneath, we have outlined descriptions of the risks that we have had to consider in this project. The risks have been identified, analyzed and the measures needed to be taken have been described. After, follows a matrix where the risks are categorized into low, medium and high risk, after green, yellow and red colours. To be ranked, it has been used two factors - consequence and probability. Both factors are separated into five levels, ranging from low to high on consequence, and very unlikely to most certain on probability. See: Table 2

1: Loss of data. There is a chance that data needed for the project, or data created as a part of the project get lost, damaged or corrupted.
This risk is categorised as unlikely because while there is a low probability of it happening, accidents can occur which bumps it up to **unlikely**.
In the worst possible scenario where everything is lost, the consequence is **high**.
This makes the overall risk: **High**.

Measures:
For the code files, we will use remote version control and the code will be cloned locally on each developers computer. This results in a type of redundancy that would prevent the loss of these files. The text documents on the other hand are more prone to loss. Therefore we are writing the report online using overleaf, and every other document is saved in OneDrive.
If a document is lost or damaged, the document should then be recoverable from a previous version.

2: We use more time than planned for specific tasks. Not completing tasks before they are due, is always a risk, and in worst case, it can delay the whole project.
The probability of this is set to **possible**, because it is hard to know what might happen months in advance.
Since the expectations of the clients are very open, the consequences is **medium low**.
Therefore the overall risk is: **Medium**.

Measures:
Having a lot of focus on the planning will help preventing this from happening. The nature of scrum is that the tasks not finished from one sprint, will be carried over to the next and we will learn and get better at predicting the time of a task. In the worst case, the group will have to skip tasks that was planned, in able to focus on other tasks.

3: A group member gets sick. A group member can end up getting sick both short term and long term.
The probability for a short term illness is **possible**, and for long term illness like hospitalisation is (**very unlikely**).
The consequence for a short term illness is **low**, and a long term hospitalisation is (**medium high**).
The overall risk for a short term illness is **Low**, and for long term it is (**Medium**).

Measures:
Measures against getting a cold or other illnesses is not something we will take, more than what we usually do. If someone does get sick though, the group will have a talk, and find out if the person still can do some work. In the worst case, the group will have to reduce the number of tasks to implement.

4: Conflict in the group. The group might end up not agreeing on something important for the thesis. We can also end up fighting over something else, and thereby loosing our good communication.
The group has know each other for almost three years, so the probability is **unlikely**.
The consequence of a bigger fight where some members will not talk to each other afterwards is **medium**.
Therefore the risk is: **Medium**.

Measures:
There are not many measures to be taken as we are already good friends. If we have a fight, we will talk about it. In the very worst case, we will get help from the supervisor to communicate.

5: A group member missed a meeting and is unresponsive for the day. One of the group members ended up not attending a meeting, and is not answering messages and calls.
The possibility is **unlikely**, but a group member might oversleep.
The consequence of losing contact with a group member for a day is **medium low**.
Which makes the risk **Low**.

Measures:
It can be hard to prevent someone forgetting to notify the group of their absence, but if they do, the rest of the group would try different channels of communications to contact the missing person to see what is happening.

6: The archaeologists does not have time for user testing One of the risks we are considering, are the fact that our stakeholders - the archaeologists at UIO does not have time for completing the user testing that we need them to do.
The archaeologists have an interest in this product becoming as good as possible, and both of them being unavailable for a longer period makes the possibility of this happening **unlikely**.
The consequence of not having a user test with the end user that will know what they want best, is **medium high**.
The overall risk is then: **Medium**.

Measures:
The group will plan with the archaeologists to save the dates for the user tests. If somethings comes up and they do not have the time, the group will try to reschedule the user tests to another date. If they never have time, and the group needs to hold user tests to continue, they will hold user tests using random users and get outside feedback from them.

7: The group finds out that an implementation they planned is not possible to make, or would take way too much time. One risk to consider is that we might simply find out that a task we planned to do, is not implementable considering the time and resources we have. This can be a big or small implementations.
The group does not know how long a feature will take to implement beforehand, so the probability

of a feature taking longer time than expected is **possible**.
The list of features that was requested was long, and the archaeologists were understanding of not getting all of them, so the consequence of a feature taking longer than expected or not being implemented is **medium low**.
That makes the risk: **Medium**.

Measures:
The group will sort the wishes of features and implementations given from the archaeologists in the product backlog. That will happen based on the two factors, importance, and how hard they are to implement. These wishes will be implemented in order through the sprints. This will prevent using time on hard features that is not of value to the users. The group may still find that an implementation they have taken on in a sprint is out of the realm of possibility to implement. It will then be discussed during the review of the sprint, and it may be moved down the product backlog or modified.

8: We don't have a program that can, run or the program contains bugs at the end.
One thing that can happen, is that the whole program will simply not run or parts of the program do not work at the time we planned that we would be done. This can be smaller or bigger bugs causing the problem.
Since the program that we are continuing on from is already running, the probability that bugs will still be present in the final program is **unlikely**.
If bugs cause parts of the program to be unusable by the archaeologists and we do not deliver a clean product, the consequences would be **medium high**.
That altogether will make the risk: **Medium**.

Measures:
To prevent this, the group will have a rule in GitHub, that everything to be pushed to the main branch, should be running and bug-free. The program will be frequently ran and tested by the developers, also in between the user tests.

9: The supervisor or stakeholder becomes hard to get in touch with. The supervisor and the stakeholder might have a busy period working or travelling. This can affect their communication with us.
We have already planned weekly meetings with both the supervisor and the stakeholder. They both work at NTNU, and will continue working here throughout the period of our project. What can be a problem, is that if one of them is travelling, and technical issues appear, getting in contact could be hard. Another thing to think about is possible health issues. Summing up, the probability is set to **medium low**.
Talking about the consequence, we could still get in contact with the archaeologists without using the stakeholder. If the stakeholder or the supervisor misses one meeting, we still have the other one. We can also get in touch at a later point of time. The consequence is therefore set to **medium low**.
Overall, the risk will therefore end up at the level: **Low**.

Measures:
Putting effort into keeping in touch with the stakeholder is what can be done from our side. Scheduling meetings at a fixed interval, as mentioned earlier, will make this easier.

		Consequence				
		Low	Medium low	Medium	Medium high	High
Probability	Very unlikely				**(3)**	
	Unlikely		**5, 9**	**4**	**6, 8**	**1**
	Possible	**3**	**2, 7**			
	Likely					
	Most certain					

Table 2: This table shows the placement of the risks after measuring, based on probability and consequence levels. The colours indicates the overall risk, ranging from green - low, through yellow - medium, to red - high. Each numbers in the table corresponds to the number of a risk in the list above.

Risk summary. The risk analysis shows us that there are a few low, a few high risks, and that most of the risks are at a medium level shown in Figure 1. Having a big project is never risk free, so this is just natural. If the project had a lot of high risks though, it would be a sign that the project would be out of scope, with our capabilities.

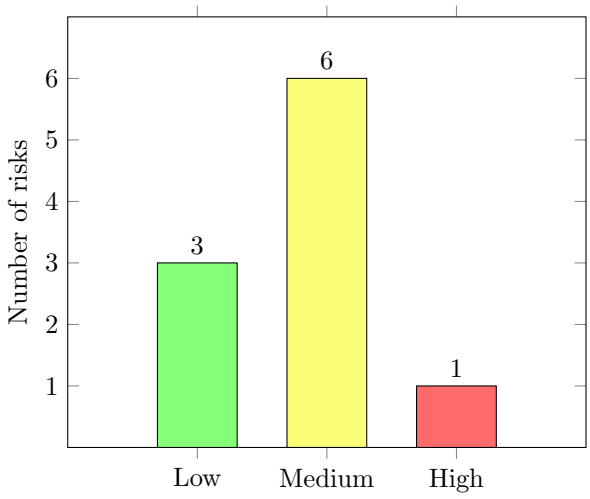


Figure 1: Histogram showing the number of risks in the different risk levels.

7 Plan for Execution

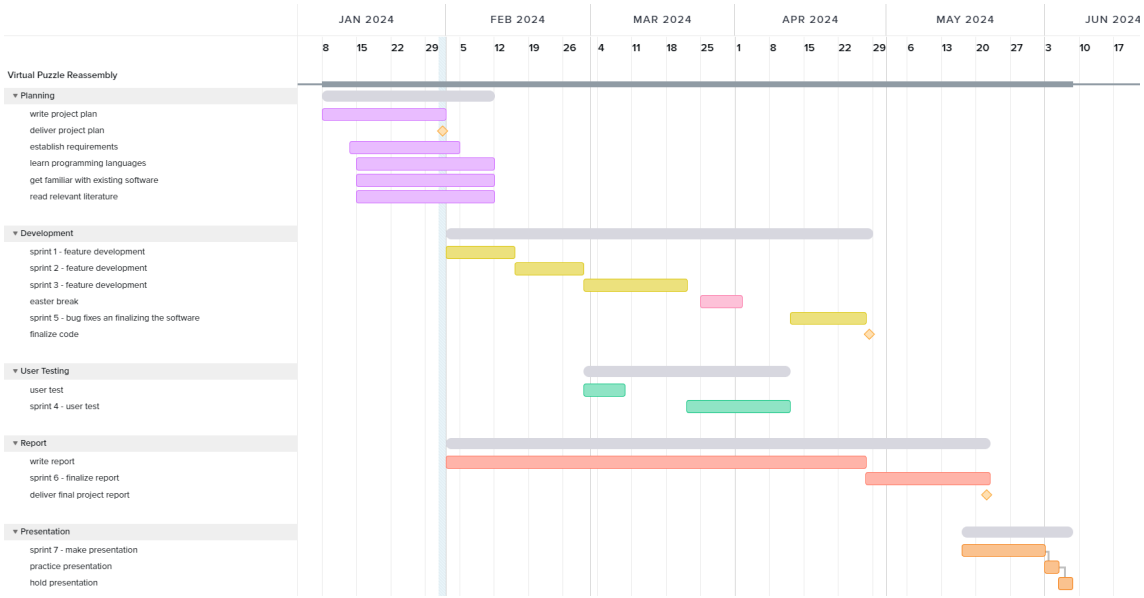


Figure 2: Gantt chart of the project plan

In this project our activities are: developing the software, user testing, writing the report of the bachelor thesis, and holding a presentation of the final project. We have segmented the process into seven sprints. Sprint 1, 2, and 3 will be focused on feature development while sprint 4 will focus on user testing. Sprint 5 will focus on bug fixes and finalizing the software based on the feedback of the user tests. In sprint 6, we will write the report complete. The final sprint, sprint 7, will involve making and presenting a presentation of the result of our bachelor thesis. We have planed to conduct two user tests with the archaeologists at UiO. One user test in the beginning of March and one at the beginning of April. Before the first sprint we will also have a planning phase where we write the initial project plan, establish the requirements of the software, and research the state of the already existing software.

Milestones are dates where we will check if we are on schedule with the plan. We have the following milestones:

1. The 31st of January the project plan will be delivered to the supervisor.
2. The 26th of April we have set the deadline for the software development.
3. The 21st of May is the delivery date of the report.
4. The 5th or the 6th of June the presentation of the project will be held.

Bibliography

[1] Granberg S., Opphus S. and Slettum O. A. 'Viten i senter'. BA thesis. Norwegian University of Science and Technology (NTNU), 2017, pp. 85–103.

[2] Arildset A., Brynhildsen C. H., Hestveen S. and Urne T. 'Securing the Software Development Life Cycle'. BA thesis. Norwegian University of Science and Technology (NTNU), 2023, pp. 143–157.

[3] Bjerken B. L., Holter L. B., Huynh D. H. and Wangerud L. A. 'Fish Detection in Underwater Video'. BA thesis. Norwegian University of Science and Technology (NTNU), 2023, pp. 145–176.

[4] Vedeler M. *Oseberg - de gåtefulle billedvevene*. Scandinavian Academic Press, 2019.

[5] Gulbrandsen C. F. 'Reconstructing the Original: Machine Learning Puzzle Assembly for Matching Archaeological Textile Fragments'. MA thesis. Norwegian University of Science and Technology (NTNU), 2023.

[6] Gigilashvili D., Nguyen H. T., Gulbrandsen C. F., Havgar M., Vedeler M. and Hardeberg J. Y. 'Texture-based Clustering of Archaeological Textile Images'. In: *Archiving Conference* (2023). URL: https://library.imaging.org/archiving/articles/20/1/29.

[7] Schwaber K. and Schwaber J. *The 2020 Scrum Guide*. 2020. URL: https://scrumguides.org/scrum-guide.html (visited on 26th Jan. 2024).

APPENDIX D

INTERVIEWS

The first interview

Date: 12.01.2024
Time: 11.00 - 12.00
Place: Teams
Present: Mathias Iversen, Mikael R. Mathiassen, Oda K. F. Steinsholt, Margrethe Havgar, Davit Gigilashvili

1. What kind of meeting the report is about

Interview with the archaeologists from the University of Oslo.

2. Which cases were discussed, and which decision was made?

At first, the archaeologist talked freely about her work:

- She uses the Adobe Fresco program for her work with the fragments.
- Colour filter could be set using a colour wheel. Sliders are good, but numbers are not intuitive for these users.
- She emphasized that the filters should only apply to selected pieces because different measures are performed on different fragments.
- She also expressed a desire for a feature that allows users to drag and select multiple fragments.
- She stated that scaling the images not a good function for the application.
- When asked if the software should have cropping and automatic or semi-automatic segmentation features, she suggested it could be a good idea.
- She also indicated that a screen grab/shot feature would be beneficial.
- The ability to add a text box feature connected to the fragment would be nice.
- Save function is a necessary feature. This could give the possibility to have multiple projects and export and share projects.
- Undo and redo functionalities were noted as desirable.
- Zoom is important since there are big fragments in reality.
- We asked about having an algorithm enhancing contrast algorithm. She wants the software to help the user to some degree.
- She often uses a drawing feature similar to paint, which would be great to have on a separate layer within the software.
- When asked if having a binary image filter be useful, she answered maybe.
- She suggested having the ability to apply different color settings to various parts of a single image simultaneously by separating an image into parts.

- She recommended including credits for the images within the canvas, possibly on their own layer.
- We discussed having a ruler/measurement indicator in the application.
- When discussing using machine learning for helping the user fit images together, she said that it also needs to explain why these two images fit together for example.

Then we asked her some questions:

1. Tell us more about the process:
    a. Illumination and comparison
    b. Placement where they were found (compared to each other)
    c. Try to find threads per centimetre, which means that it could have been woven at the same time/loom. How are the threads spinning compared to each other.
    d. Colouring and sketching in layers.
2. Are there any problems software-wise today?
    a. High-resolution images that need to be downscaled because the file size is too small. The resolutions end up at only 50% because of this.
    b. Solution. Vector-image? Lower resolution when zooming out and high again when zooming in.
3. What is the hardest part of the process?
    a. Figuring out the motives.
4. What can our software help with?
    a. Showing the results to people, easier to explain.
    b. Gamified version for the museum.
    c. Much easier to create figures, quicker.
5. When comparing two pieces, what do you focus on: outlines or motive?
    a. Start with motive, and then look at edges, to narrow it down.
    b. People do different stuff.
    c. Supervisor Margrethe do more technical details.
6. Would you like to have info in the canvas?
    a. A lot of data, that can make it harder.
7. What platform/OS do you use?
    a. Windows first of all. Everyone has access to windows. (10 and 11). PC.
8. Is there other software that you use today?
    a. RTI something. Different usage from our software.
9. Do you already have metrics?
    a. The size of the bands of the fragments.

The second interview

Date: 05.03.2024
Time: 11.30 - 12.00
Place: Teams
Present: Mathias Iversen, Mikael R. Mathiassen, Oda K. F. Steinsholt, Margrethe Havgar, Davit Gigilashvili

1. **What kind of meeting the report is about**

Interview with the archaeologists from the University of Oslo.

2. **Which cases were discussed, and which decision was made?**

Frequency/usage

- Do you use the system on a regular basis today?
  - Haven't used it all that much.
- Do you think that you would like to use this system frequently?
  - 100% yes.
- How frequently would you use the system?
  - If focusing on the fragments for a period, it would be used every second day or so
- What could make you use it more frequently?
  - Having the option to display the file name as a text box (and possibly edit them). Would make it easier to reference each fragment in a report or similar.
- Do you like moving the fragments on a grid or free move?
  - Really like the way it works right now, as free move.

Colour wheel

- Are the colour filters in today's version (HSV + C) useful in your work?
  - Using the HSV colour space is useful.
- How would you describe the use in your work process?
  - Look at how it looks in adobe fresco. Choose the colour range and only affect the colours in that range.

- Are there any other colour spaces that are useful to you?
    - HSV could be more useful. It could be difficult for people that has not used HSL for 1.5 years to get used to it.
- How is the process in Adobe Fresco when you want to change the contrast of an image? Can you show us how you use their colour wheel.
    - Contrast is useful, but it is only really useful if the brightness function is also present.

Filters

- Which filters are you using in Adobe Fresco today?
    - Sometimes using blend mode in fresco
    - A filter for sort of inverting the image. Like exclusion in blend mode
- What is the goal when you apply a filter? What do you look for?
    - What is your approach or method when applying filters?
        - First: To do anything she can to increase differences in the fragment. Can she see different objects and features. Can she use that to interpret what she is looking at.
        - Alternating the hues to quickly get information about the fragment.
        - Changing specific fragments in our program would be useful
    - Trial and error or specific method?
    - Do you use any automatic filters (algorithms) that do the job for you?
- Examples from Konva filters https://konvajs.org/docs/filters/Contrast.html

  There are a few filters we are thinking about applying. We want to quickly show them to you, and then you can answer how useful the filter would be in your work on this scale: (very useful/somewhat useful/a little bit useful/redundant), (3/2/1/0).

    - How useful are these filters?
        - Edge detection (Of patterns on the fragment)
            - Could absolutely be useful.
        - Line detection
            - A great idea but would not be super useful for the work on these fragments. Could be really useful for some other fragments, but for the detail on these fragments it could be hard to get it to a good enough level.
        - Noise removal
            - could be great. There are some noise on the fragments, and it could be good to see what the fragments will recognise as noise.
        - Contrast
            - Absolutely useful.

- Emboss
  - Very cool. A feature that she is do not have access to now.
- Grayscale
  - Think it could be useful. Because there is fragment that they think belong, but they degraded differently, so on a greyscale it could be easier to see the similarities.
- Invert
  - Invert could be cool. Curios on how it would affect the images.
- Mask
  - Cool, but unsure how useful it would for the fragments.
- Merge filters
  - Useful. Would be great.

Histogram

- One of the features in our that we have taken on, is making a histogram score of the images, based on colours. Do you understand what that means/does?
  - yes
- Will this be useful for you?
  - Using histogram on the hues of the fragment would be useful.

APPENDIX E

SCRUM BACKLOG

Item number	Name of the item	Description	Dependent on	Difficulty	Value	Priority	Sprint	Status
1	Select multiple elements	The ability to select more elements on the canvas by clicking and holding a selecting key like the ctrl or shift key. These elements can then be moved, rotated, and deleted together.		Medium	High	level 2	1, 2, 3	Done
1.1 (32.1.1)	Apply filters to the selected images	Make filters apply to the selected images.		Medium	High	level 2	4	Done
2	Similarity metrics window	Create a window that can display the different Computer Vision comparison scores between images in a meaningful way.		Low	High	level 1	2	Done
2.1	Improve similarity metric window	Improve the design of the similarity metrics window.		Low	Medium	level 2	4	Done
2.2	Make similarity metrics window show information on selected	Make the similarity show information on the selected and compare scores based on the selected.		Medium	High	level 2	4, 5	Done
2.3	Score table component	Refactor the score table into a component rather than a function.	2	Low	Medium	level 2	6	Done
2.4	Make score table sortable	Make the table sort when the users click the table headers.	2	Low	High	level 1	6	Done
3	Saving and opening projects	The feature of making a new project, saving a project and opening an existing project using the Tauri dialogue windows for saving and opening the project files. This makes it possible to have multiple projects.	4	Medium	High	level 2	1, 2, 3	Done
3.1	Confirm save modal on return	Make a pop-up modal to ask the user to save the project when returning to the starting page. This is to improve the software usability by preventing the error of losing the project work by not saving before returning to the starting page.		Low	High	level 1	4	Done
4	Project file	The project could be represented through a file that contains information about the project. This will not contain direct image information to reduce file size. This file will include: * Project name * Position of the canvas as coordinates * The zoom scale of the canvas * The info on all the elements on the canvas: · The type of element · The position of the element · The rotation of the element · The identification of the element · If the element is an image contain 1. The file name of the image file 2. The file path to the image file 3. If the image has filters, the filter values		Medium	High	level 2	1, 2, 3	Done
4.1	Make Project file save image file path	The project file sould not direct image information like the url to reduce the file size. Instead save the file path of the images in the file.		Medium	Medium	level 3	4	Done
4.2	Save rotation of the images	Save the rotation of the elements on the canvas in the project file.		Medium	High	level 2	4	Done

6	Filter slider and toggle	A component that contains a slider, spin box and reset button for changing the filter values on an image. Also, a component to turn on or off filters that either can be on or off.	33	Medium	High	level 2	3?, 4	Done
6.1	HSV image filter	Implement the Hue, Saturation and Value image filter from Konva.js.		Low	High	level 1	4	Done
6.2	Contrast and brightness image filters	Implement the Contrast and Luminance image filters from Konva.js		Low	High	level 1	4	Done
7	Export images of the canvas	Give the users an intuitive way to save an image of the current canvas view in different resolution scales.		Low	Medium	level 2	1	Done
7.1	Export Image Modal	When exporting an image of the canvas, make it more intuitive for the users by making a pop-up modal.		Low	High	level 1	4	Done
7.2	Export Image with specified name and place	When exporting an image of the canvas, prompt a Tauri save dialogue window to allow the users to save the image with a specified name in a specified place.		Medium	High	level 2	4	Done
9	Improve undo and redo	Improve the implementation of undo and redo by making the redo function and separating the logic for undo and redo away from unrelated code. Additionally, implement common key shortcuts used in other software and buttons in the navigation bar for undo and redo.		Medium	Medium	level 3	4	Done
19	Hue Histogram metrics	Implement different metrics to compare the hue histograms of the different images.	2	Low	Medium	level 2	3?, 4, 5	Done
19.1	Hue histogram	Display a bar graph for each image in the similarity metrics window. This graph represents the number of pixels in the image having a hue value between two integer hue degrees.		Low	Medium	level 2	4, 5	Done
21	Find work area function	A function to move the canvas automatically so that the closest image is in the centre and the zoom scale is 1. This will be done in a smooth animation.		Medium	High	level 2	4	Done
24	Remove the commandline	When the program is running there is a commandline running in the background. We want to remove this		Low	Medium	level 2	4	Done
28	Image Filters	Implement the ability to add multiple different filters to an image. These filters are in the Konva.js library.	1.1	Medium	High	level 2	4	Done
28.1	Filter window	Create a window where users can add, change or remove filters on specified images.		Medium	High	level 2	4	Done
32	Usability test 2 changes	Make changes based on the feedback from second usability test.	The 2. usability test	Medium	High	level 2	5	Done
32.1	Nav bar filter button	Make an open filter button to the navigation bar instead of right-clicking on an image to open the filter window.		Medium	High	level 2	5	Done

32.2	Filter window changes	Make changes to the filter window. This includes: * removing the luminance filter * change the name of the value in the HSV filter to brightness * change the name of the reset filters button * change the name of the enable / disable button * other design changes		Low	High	level 1	5	Done
32.3	Undo and redo arrows	Make the undo and redo buttons in the navigation bar use intuitive arrow icons instead of text.		Low	Medium	level 2	5	Done
33	Invert image filter	Implement invert color image filter from Konva.js.	28	Low	Medium	level 2	4	Done
35	Mask threshold image filter	Implement the mask treasholding image filter form Konva.js.	28	Low	Low	level 3	4	Done
36	Grayscale image filter	Implement grayscale image filter from konva.js.	28	Low	Medium	level 2	4	Done
26	Display name of the fragment file	Display the filename of the fragment file as a text box and edit them. This would make it easier for the user to refernce them in a report or similar.	8	Medium	High	level 2		
27	Colour range	Selecting a colour range and only affecting the colour enhancement to that range.	6	Medium	High	level 2		
3.1.1	Save on close	Make the modal for confirm save pop up when closing from the canvas		Medium	High	level 2	4?	
32.4	Make exporting easier	The user requested a "default" value for the resolution when they now have to change the number. An alternative could be 3 default values to change between (low, medium and high). The solution as it is now could be an advanced setting, that the user could find if they want to.		Medium	High	level 2		
23	Sign the program build	When downloading the program a warning comes up since it is not signed.		Low	Medium	level 2		
2.4.1	Improve score table sort	Add the functionality to sort in descending order and add a visual indicator to show which column it is sorted by and which direction	2.4	Low	Medium	level 2		
5	Grouping images	The ability to lock or combine the selected canvas elements into a group that will be utilised as a unified canvas element.	1	High	High	level 3	2?, 3, 4, 5	In development
8	Adding a text-box/note to a piece image	Add a canvas object that could holds text. In this way it could be locked to other canvas objects or free floating on the canvas. There could also be added a feature to add text as metadata to the image.		High	High	level 3		
10	High resolution images	Ability to use high resolution images that would scale down so that they can be used with ease. The image files can be 1GB large, so some memory optimization could be needed.		High	High	level 3		
11	Motif score	Use basic machine learining or klustering to compare the motif of two images	2	High	High	level 3		
12	Color enhancement algorithms	Add algorithms to enhance the colour of a pieces automatically. Like stress or contrast stretching. Focusing on algorithms to enhance differences in the image.		High	High	level 3		

29	Canny edge detection filter	Filter for detecting edges in the patterns in the image	28	High	High	level 3		
1.2	Drag select	the functionality to click and drag select	1	High	High	level 3		
3.1	List of projects	a list in the landing page, where you chould click on one to open it. When you save a project or open a new project it would be added to the list.	4	Medium	Medium	level 3		
25	Dark mode	have a mode where the background of the canvas is dark		Medium	Medium	level 3		
34	Crop images	Crop the loaded images so that there are less invisable space around the image		Medium	Medium	level 3	4?	
5.1	Unlocking connected pieces	Make the ability to unlock groups that has been locked together		Medium	Medium	level 3	5?	
37	Image context menu	Create a context menu that shows up when right-clicking on an image. Containing a button to delete the image.		Medium	Medium	level 3		
22	Project information window	when making a new project a window will pop up and ask for infromation like project name and description. In the canvas page you can also open the window to edit the information		Low	Low	level 3		
31	Emboss	implement the different emboss filters from konva	28	Low	Low	level 3		
18	Size indicator	Size indicator on the canvas to compare the pieces to real world size. Inspired by map size indicator.		High	Medium	level 4		
30	Noise removal filter	A median filter to reduse the noise in an image	28	High	Medium	level 4		
6.3	Specific saturation change	Make functionality that can a find specific hue value (like red or yellow), and let the user change the saturation on only those parts of the image		High	Medium	level 4		
13	Drawing	Drawing on top of the fragments on another layer in multiple colours. Sketching is an important step for the archaeologists to interpret the motifs.		Medium	Low	level 4		
16	Crediting	Crediting the photographer of the pictures.		Medium	Low	level 4		
20	Shape score	Use basic machne learining or klustering to compare the shape outline of two images	2	High	Low	level 5		
14	Segmentation	Automatic or semi-automatic segmentation of piece images when loading them in.		High	Low	level 5		
17	Image seperation	Separating a piece into multiple pieces. Some pictures are of multiple pieces, but their location might not be correct and some needs to have different enhancements on different parts of the piece.	14	High	Low	level 5		

APPENDIX F

SPRINT BACKLOGS

Sprint 1 Backlog

Sprint goal (why):

To learn about and understand the process of a sprint in Scrum by having a “jump start”.

The tasks selected for sprint 1 (what):

ID	Name	Description	Dependent on	Difficulty	Value	Priority
1	Select multiple images	Select individual or multiple pieces and only affect those with for example the colour filter. Using a drag and select function, or Ctrl clicking.		Low	High	Level 1
3	Saving and loading	Saving and loading of the canvas and the possibility to have multiple canvas projects. The program has currently no way to save and when closing the program everything is lost. This is a high priority feature, because puzzle solving may take days, weeks, or months.	4	Medium	High	Level 2
4	Project file	The project could be saved to a file that can be shared. This file will contain references to the images that are on the canvas their transformation: position, rotation, scaling, their image enhancements, and if they are locked together to other pieces and other info about the state of the canvas. The file will not contain the images themselves to limit the size of the exported files.		Medium	High	Level 2
7	Export images of the canvas	The canvas should be able to be exported to different image file types at different resolutions. Using the highest resolution of images for the image of the canvas may be impossible.		Low	Medium	Level 2

The plan to get the work done (how):

- In the beginning of this sprint, we will work with the development together
- First day: develop/find file type
- Second day: save-functions for the selected file type
- Third say: load-functions for the selected file type
- Fourth day: use files to export image of canvas
- After that: Select multiple images

Sprint 2 Backlog

Sprint goal (why):

Make our product ready for user testing

The tasks selected for sprint 2 (what):

ID	Name	Description	Dependent on	Difficulty	Value	Priority
1	Select multiple images	Select individual or multiple pieces and only affect those with for example the colour filter. Using a drag and select function, or Ctrl clicking.		Low	High	Level 1
3	Saving and loading	Saving and loading of the canvas and the possibility to have multiple canvas projects. The program has currently no way to save and when closing the program everything is lost. This is a high priority feature, because puzzle solving may take days, weeks, or months.	4	Medium	High	Level 2
4	Project file (importing and exporting)	The project could be saved to a file that can be shared. This file will contain references to the images that are on the canvas their transformation: position, rotation, scaling, their image enhancements, and if they are locked together to other pieces and other info about the state of the canvas. The file will not contain the images themselves to limit the size of the exported files.		Medium	High	Level 2
2	Score comparison window	Make a window to show the comparison scores between two images on the canvas in a meaningful way.		Low	High	Level 1
5	Locking / unlocking pieces together	Combining/locking two or more canvas objects together.	1	Medium	High	Level 2

The plan to get the work done (how):

- Mikael will do number 2
- Oda will finish number 1 and 5
- Mathias will finish 3 and 4

Sprint 3 Backlog

Sprint goal (why):

“To get a good understanding of the user’s needs in our program”

The tasks selected for sprint 3 (what):

ID	Name	Description	Dependent on	Difficulty	Value	Priority
1	Select multiple images	Select individual or multiple pieces and only affect those with for example the colour filter. Using Ctrl clicking.		Medium	High	Level 2
3	Saving and loading	Saving and loading of the canvas and the possibility to have multiple canvas projects. The program has currently no way to save and when closing the program everything is lost. This is a high priority feature, because puzzle solving may take days, weeks, or months. Without saving and loading, everything needs to be done in one go, which make the software very inconvenient to use.	4	Medium	High	Level 2
4	Project file (importing and exporting)	The project could be saved to a file that can be shared. This file will contain references to the images that are on the canvas their transformation: position, rotation, scaling, their image enhancements, and if they are locked together to other pieces and other info about the state of the canvas. The file will not contain the images themselves to limit the size of the exported files.		Medium	High	Level 2
5	Locking / unlocking pieces together	Combining/locking two or more canvas objects together.	1	Medium	High	Level 2

	User testing	Planning and implementation of user tests with the archaeologists. Writing about the results in the report.				
6	Colour wheel	The colour filter that currently only uses numbers. Add a colour wheel or spectrum to make it more intuitive and user friendly. This should be enabled or disabled with a button.		Medium	High	level 2
19	Histogram score	compare colour histogram of two images		Low	Medium	level 2

The plan to get the work done (how):

- Mikael will start planning the user test, with input from Oda and Mathias.
- Oda will work on finishing the task 1.
- Mathias will work on 3 and 4.
- Then we will host the user tests
- After the user tests we will write about the results
- Parallelly we will finish task 5.
- After that, we will start on task 6 and 19.

Sprint 4 Backlog

Sprint goal (why):

"Develop the software based on the user feedback, making changes that are valuable to the users, that can be tested in the next sprint."

The tasks selected for sprint 4 (what):

ID	Name	Description	Dependent on	Difficulty	Value	Priority
5	Locking / unlocking pieces together	Combining/locking two or more canvas objects together.	1	Medium	High	level 2
6	Colour weel	The colour filter that currently only uses numbers. Add a colour wheel or sliders to make it more intuitive and user friendly. This should be enabled or disabled with a button.		Medium	High	level 2
19	Histogram score	compare colour histogram of two images based on hue	2	Low	Medium	level 2
19.1	Show histogram			Low	Medium	level 2
1.1	Filter only selected	Make filters only apply to selected images.		Medium	High	level 2
6.1	Implement HSV	implement the hue, saturation and value colour space.		Low	High	level 1
6.2	Contrast and brightness	implementing the contrast and brightness		Low	High	level 1
7.1	Make the scaling more intuitive	Make the scaling more intuitive based on the feedback from the users. (low, medium, high + advanced)		Low	High	level 1
7.2	Save dialog window	Saving the export image with specified name and place		Medium	High	level 2
28	Merge filters	make the ability to have multiple filters on an image		Medium	High	level 2
2.1	Improve design on similarity matric window					

2.2	Show only information on selected					
4.1	Make project file use file path instead of url					
4.2	Save the rotation of the image					
	Write on the introduction					
	Write on the theory					
9	Undo/redo	Undo and redo buttons in the graphical user interface. Only works with Ctrl+z or Ctrl+y for now.		Low	Medium	level 2
34	Crop images	Crop the loaded images so that there is less invisible space around the image		Medium	Medium	level 3
3.1	Save on close or return	Make a pop up window that ask to save on close or return		Low	High	level 1
28.1	Make filter window	Make a pop up window where user can change filter values of a specific image		Medium	Medium	level 3
21	Find my canvas	Move and zoom in the canvas automatically so that all elements are in view		Medium	High	level 2
3.1.1	Save on close	Make the modal for confirm save pop up when closing from the canvas				
24	Remove the command line	When the program is running there is a command line running in the background. We want to remove this		Low	Medium	level 2
36	Grayscale	Implement grayscale filter from Konva	28	Low	Medium	level 2
33	Invert	Implement the invert filter from Konva	28	Low	Medium	level 2
35	Mask	Implement the mask thresholding form Konva	28	Low	Low	level 3

After this we will add more items as we finish them.

The plan to get the work done (how):

With undertasks:

- Mikael: 2, 6, 7, 19, 21, 24, 28,
- Mathias: 3, 4, 6, 9, 35, 36, 33,
- Oda: 1, 5,

Sprint 5 Backlog

Sprint goal (why):

Finish the programming in a great and presentable way.

The tasks selected for sprint 5 (what):

ID	Name	Description	Dependent on	Difficulty	Value	Priority
5	Locking / unlocking pieces together	Combining/locking two or more canvas objects together.	1	Medium	High	level 2
5.1	Unlocking connected pieces	Make the ability to unlock groups that has been locked together		Medium	High	level 2
19	Histogram score	Compare colour histogram of two images based on hue	2	Low	Medium	level 2
19.1	Show histogram			Low	Medium	level 2
2.2	Show only information on selected					
32.1	Nav bar filter button (we need to discuss this task!)	Put the "open filter" button in the nav bar, and make it apply to all selected elements		Medium	High	level 2
32.1.1	Make filters apply to all selected	Now the filters only apply to the right clicked, but the filters should also apply to multiple (the selected) images	1	Medium	High	level 2
32.2	Filter window changes	Remove "luminance", change the name of "value" and do some design changes to the sliders. Change the names of the buttons; reset filters, and enable/disable		Low	High	level 1
32.3	Arrows	Make undo/redo arrows instead of text		Low	Medium	level 2

- Other programming tasks will be either user test feedback tasks or cleaning up the code such as refactoring and linting.

- Write a lot on the report:
    - Finish “Theory”
    - Finish “Kravspesifikasjon”
    - Write on results: about the user test

The plan to get the work done (how):

- Oda will work with task 5.
- Mikael will work on task 19 and 2.2.
- Mathias will make changes based on the user test feedback.
- Everyone will write in the report.

Sprint 6 Backlog

Sprint goal (why):

Finishing and evaluating the program, then switching focus to the report writing.

The tasks selected for sprint 6 (what):

ID	Name	Description	Dependent on	Difficulty	Value	Priority
5	Locking / unlocking pieces together	Combining/locking two or more canvas objects together.	1	Medium	High	level 2
	Report writing					
	Usability testing					
	Update the README.md file					

The plan to get the work done (how):

- Oda and Mathias will continue refactoring. If it takes way much time, they will give up.
- Then they will join Mikael on the report writing.
- Mikael will make a usability test form, and they all will contribute to usability testing the program with people on the school.

APPENDIX G

GITHUB REPOSITORY

All code used in this document is included in the GitHub repository linked below. The README.md file provides further explanations.

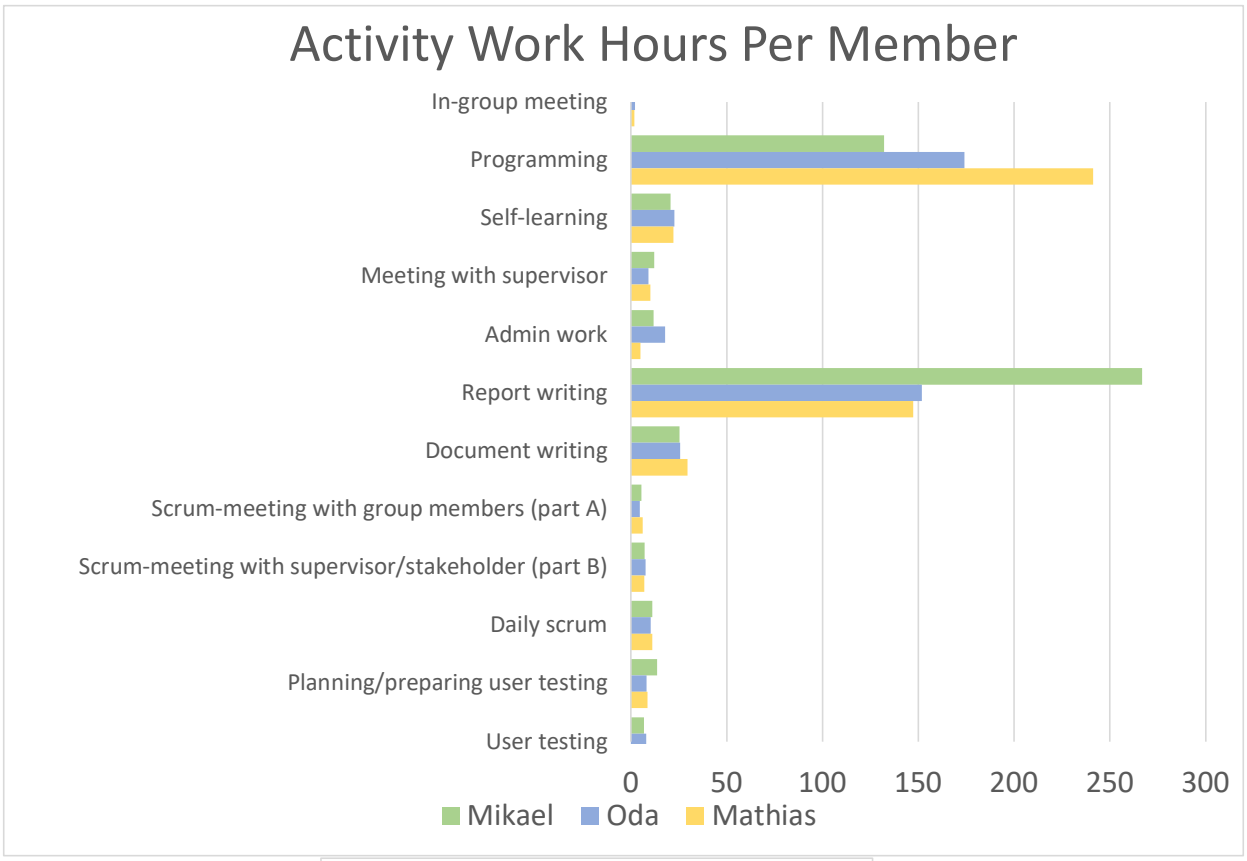
Github Repository Link

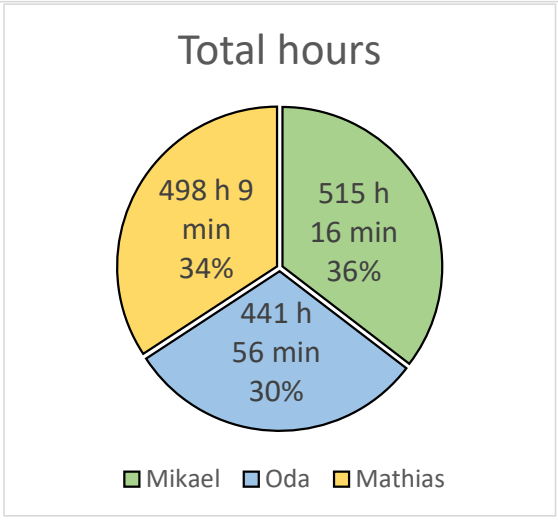
- `https://github.com/MikaelRoev/artifact_assembly`

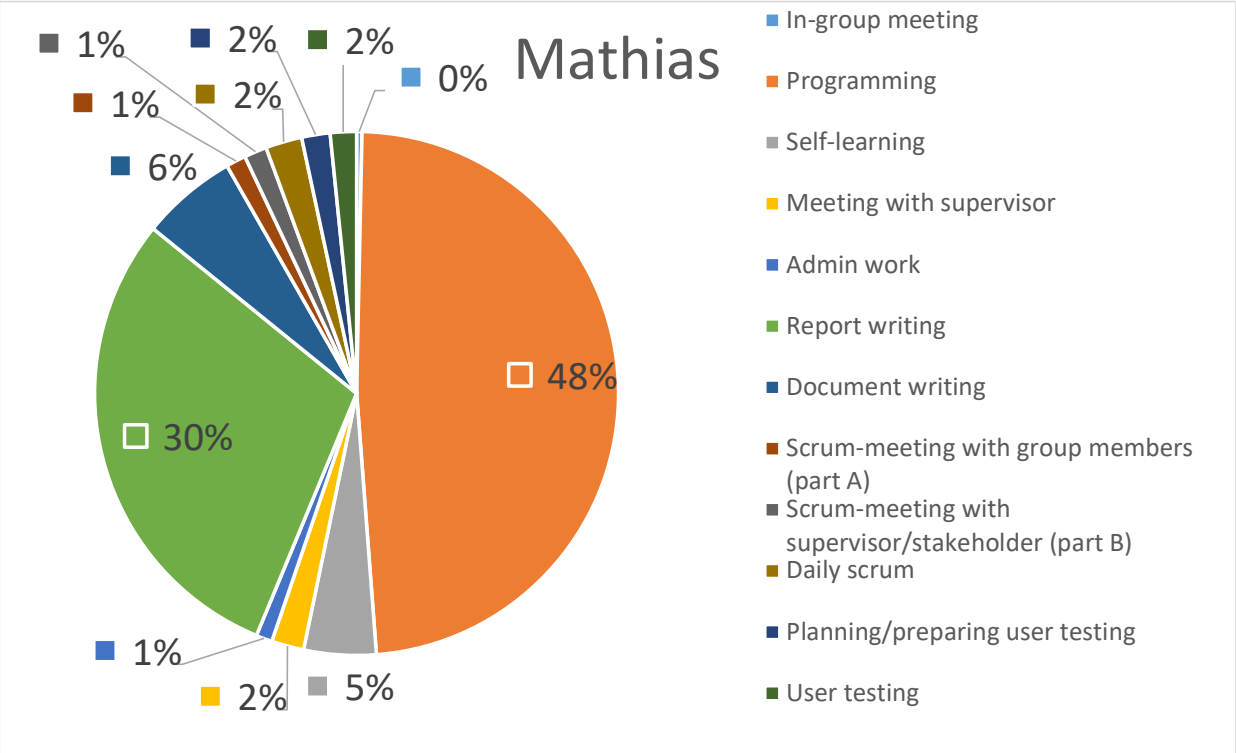
APPENDIX H

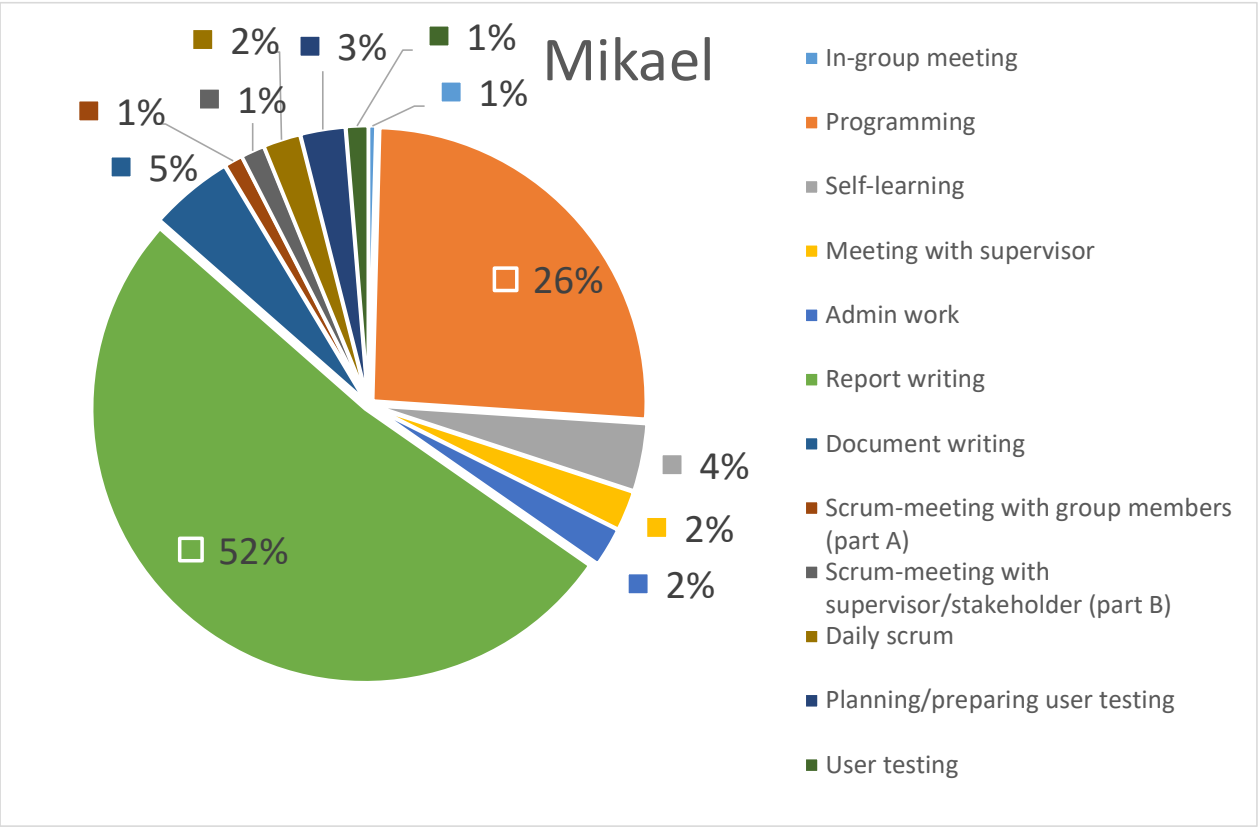
TIME KEEPING

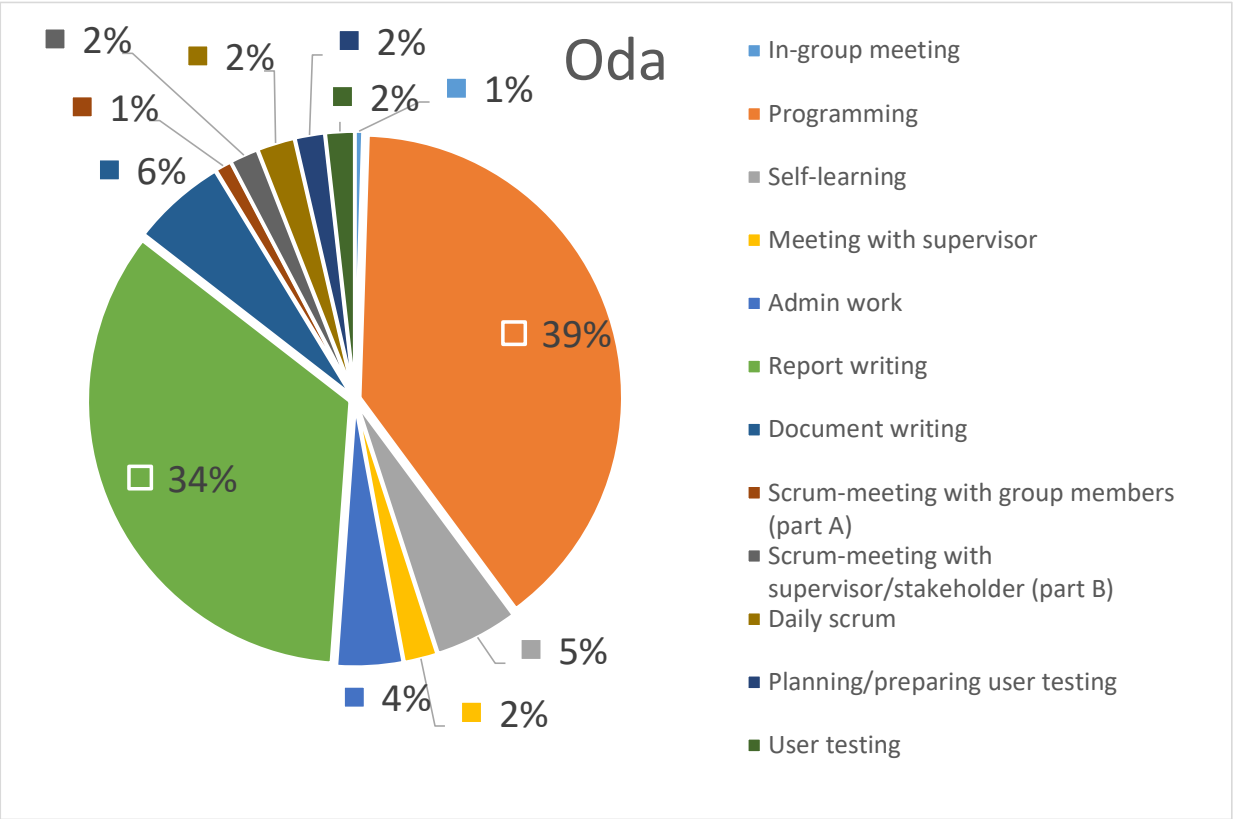
	Mikael	Oda	Mathias
In-group meeting	2 h 10 min	2 h 10 min	1 h 40 min
Programming	132 h 3 min	174 h 0 min	241 h 21 min
Self-learning	20 h 34 min	22 h 45 min	22 h 12 min
Meeting with supervisor	12 h 6 min	9 h 6 min	10 h 1 min
Admin work	11 h 48 min	17 h 57 min	5 h 0 min
Report writing	266 h 53 min	151 h 50 min	147 h 20 min
Document writing	25 h 22 min	25 h 43 min	29 h 30 min
Scrum-meeting with group members (part A)	5 h 35 min	4 h 30 min	6 h 10 min
Scrum-meeting with supervisor/stakeholder (part B)	7 h 7 min	7 h 40 min	6 h 55 min
Daily scrum	11 h 15 min	10 h 15 min	11 h 15 min
Planning/preparing user testing	13 h 37 min	8 h 5 min	8 h 40 min
User testing	6 h 46 min	7 h 55 min	8 h 5 min
Total hours	515 h 16 min	441 h 56 min	498 h 9 min
Average hours	30 h 19 min	27 h 37 min	29 h 18 min

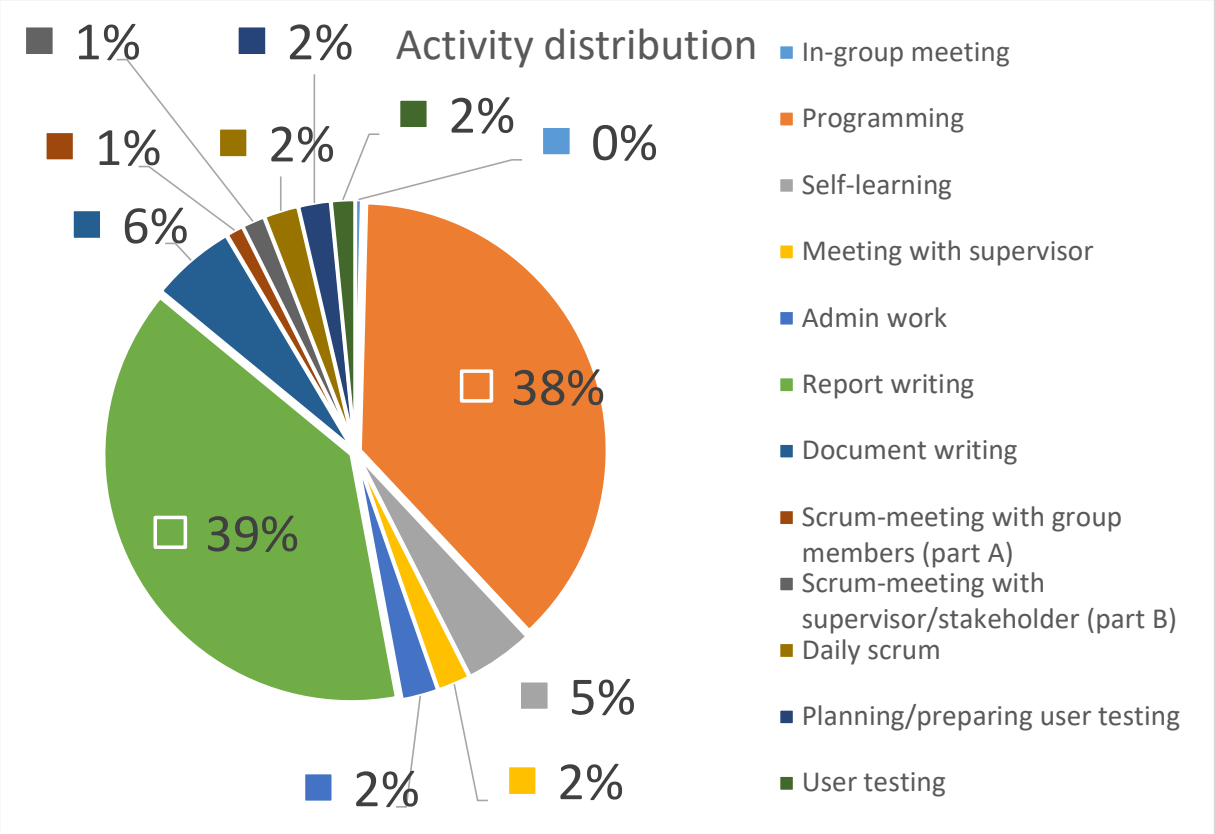





Mathias
0%
48%
30%
1%
2%
5%
6%
1%
1%
2%
2%
In-group meeting
Programming
Self-learning
Meeting with supervisor
Admin work
Report writing
Document writing
Scrum-meeting with group members (part A)
Scrum-meeting with supervisor/stakeholder (part B)
Daily scrum
Planning/preparing user testing
User testing


Mikael
1%
26%
4%
2%
2%
52%
5%
1%
1%
3%
2%
In-group meeting
Programming
Self-learning
Meeting with supervisor
Admin work
Report writing
Document writing
Scrum-meeting with group members (part A)
Scrum-meeting with supervisor/stakeholder (part B)
Daily scrum
Planning/preparing user testing
User testing


Oda
2%
2%
2%
1%
2%
1%
6%
39%
34%
5%
4%
2%
In-group meeting
Programming
Self-learning
Meeting with supervisor
Admin work
Report writing
Document writing
Scrum-meeting with group members (part A)
Scrum-meeting with supervisor/stakeholder (part B)
Daily scrum
Planning/preparing user testing
User testing


Activity distribution
1%
2%
2%
0%
1%
2%
6%
38%
39%
5%
2%
2%
In-group meeting
Programming
Self-learning
Meeting with supervisor
Admin work
Report writing
Document writing
Scrum-meeting with group members (part A)
Scrum-meeting with supervisor/stakeholder (part B)
Daily scrum
Planning/preparing user testing
User testing

Oda's time worked

Week 2, 3, 4				
Activity	Hours	Minutes	Decimal time	Comment
Meeting with supervisor	4	0	4	Week 2+3
Self-learning	1	0	1	Looking at the existing software
Document writing	17	3	17.05	Project plan
Meeting with supervisor		51	0.85	Talked about project plan
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
		Total	22.9	22 h 54 min

Week 5				
Activity	Hours	Minutes	Decimal time	Comment
Document writing	8	40	8.666666667	Finishing project plan
Meeting with supervisor		50	0.833333333	
Self-learning	5	10	5.166666667	thu: JS, CSS & HTML fri: konva.js
Admin work	2	32	2.533333333	
Scrum-meeting with group members (part A)		55	0.916666667	thu: sprint meeting part A
Scrum-meeting with supervisor/stakeholder (part B)		50	0.833333333	fri: sprint meeting part B
In-group meeting		30	0.5	fri: discussing
Daily scrum		15	0.25	
			0	
			0	
			0	
			0	
		Total	19.7	19 h 42 min

Week 6				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		40	0.666666667	
Scrum-meeting with supervisor/stakeholder (part B)		50	0.833333333	Thu
Self-learning	13		13	Wed: Konva Thu: Looking up how to mark objekts Fri: Marking several objects
Programming	3	45	3.75	Thu: Making list of marked objects Fri: Moving the list of marked objects
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
		Total	18.25	18 h 15 min

Week 7				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		10	0.166666667	
Scrum-meeting with group members (part A)	1	10	1.166666667	Thu: prepared for sprint 2
Scrum-meeting with supervisor/stakeholder (part B)		50	0.833333333	Thu: prepared for sprint 2
Programming	5	20	5.333333333	Wed: task 1 Fri: task 1 Sat: task 1
Report writing	5	20	5.333333333	Wed: Fri: Sustainability
			0	
			0	
			0	
			0	

			0	
			0	
			0	
		Total	12.83333333	12 h 50 min

Week 8				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		20	0.333333333	
Report writing	2		2	wed: sustainability
Scrum-meeting with supervisor/stakeholder (part B)		45	0.75	thu:
Programming	16	15	16.25	wed: thu: fri: sat:
			0	f
			0	
			0	
			0	
			0	
			0	
			0	
			0	
		Total	19.33333333	19 h 20 min

Week 9				
Activity	Hours	Minutes	Decimal time	Comment
Programming	16	25	16.41666667	Wed: task 1 Thu: finished task 1 Fri: task 5
Daily scrum		25	0.416666667	
Scrum-meeting with group members (part A)		20	0.333333333	Thu: prepared for sprint 3
Scrum-meeting with supervisor/stakeholder (part B)		45	0.75	Thu: prepared for sprint 3
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
		Total	17.91666667	17 h 55 min

Week 10				
Activity	Hours	Minutes	Decimal time	Comment
Planning/preparing user testing	4		4	
User testing	2	55	2.916666667	Tue: user test and interview with the archeaologist Margrethe
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
Sickness: from wednesday and the rest of the week		Total	6.916666667	6 h 55 min

Week 11				
Activity	Hours	Minutes	Decimal time	Comment
			0	
			0	
			0	
			0	
			0	
			0	

			0	
			0	
			0	
			0	
			0	
			0	
Other absence: from thursday to saturday		Total	0	0 h 0 min

Week 12				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		40	0.666666667	
Programming	21	45	21.75	Mon: task 1.1/28 Tue: task 1.1/28 Thu: started task 5 Fri: task 5
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
Sickness: from wednesday and the rest of the week		Total	22.41666667	22 h 25 min

Week 13				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		25	0.416666667	
Programming	12		12	Wed: task 5 Thu: task 5 Sun: task 5
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
Sickness: from the beginning of the week until tuesday		Total	12.41666667	12 h 25 min

Week 14				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		40	0.666666667	
Planning/preparing user testing	2	30	2.5	Mon
In-group meeting		30	0.5	Mon: about the user test
User testing	2		2	Tue: in person user testing in Oslo
Self-learning	3		3	Wed: about task 5
Programming	3	40	3.666666667	Wed: task 5
Scrum-meeting with group members (part A)		50	0.833333333	Thu: sprint 4 restrospective and sprint 5 planning
Scrum-meeting with supervisor/stakeholder (part B)	1		1	Thu: sprint 5 planning
Report writing	4	25	4.416666667	Thu: writing about the usability tests
Admin work	2	10	2.166666667	Fri: user test document and backlog
Report writing	5	10	5.166666667	Fri: user test, interview
			0	
		Total	25.91666667	25 h 55 min

Week 15				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum	1	5	1.083333333	
Admin work	3		3	Mon: sprint backlog document cleaning Fri:

Programming	22	20	22.33333333	Mon: task 5, tried to solve problem with drawing group Tue: merging branch 5, and CSS design Wed: Design changes, bug fixing and commenting Fri: bugs in task 5
Report writing	9	10	9.166666667	Tue: Making some system in "method" Thu: "results" chapter, mainly user test and interview Fri: "results"
Self-learning		35	0.583333333	Wed: CSS
Meeting with supervisor	1	10	1.166666667	Meeting with Jon and Davit
			0	
			0	
			0	
			0	
			0	
			0	
		Total	37.33333333	37 h 20 min

Week 16				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum	1		1	
Programming	36		36	Mon: refactoring Tue: more refactoring Wed: even more refactoring Thu: even even more refactoring + evening Fri: even even even more refactoring
Scrum-meeting with group members (part A)		50	0.833333333	Thu: sprint 5 retrospective
Scrum-meeting with supervisor/stakeholder (part B)		55	0.916666667	Thu: sprint 6 planning
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
		Total	38.75	38 h 45 min

Week 17				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum	1	20	1.333333333	
Programming	36	20	36.33333333	Mon: refactoring and fixed error Tue: refactoring. Made isAnyImages and isAnySelectedImages Wed: refactored setTable and similarityMetrixsWindow Thu: refactored handleFilters() and eventhandlers Fri: emitter
Meeting with supervisor	1	5	1.083333333	Thu: meeting with Davit
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
		Total	38.75	38 h 45 min

Week 18				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum	1	20	1.333333333	
Planning/preparing user testing	1	35	1.583333333	Mon: prepare to user test
User testing	3		3	Mon: user testing Davit Wed: two more iuser tests
Admin work	8	45	8.75	Mon: cleaning up in the user test document Wed: meeting documents cleaning

Report writing	20	45	20.75	Mon: writing about interview 2 tue: thu: useability tests fri: same as thursday sun: same
Programming		10	0.166666667	tue: commenting
Scrum-meeting with group members (part A)		15	0.25	Thu: short part A meeting
Scrum-meeting with supervisor/stakeholder (part B)		45	0.75	Thu: online meeting with John Yngve
			0	
			0	
			0	
			0	
		Total	36.58333333	36 h 35 min

Week 19				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum	1		1	
Report writing	35		35	Mon: finished user testing. helped Mathias with formualtions Tue: theory on hooks and DOM wed: theory; components thu: thery; hooks fri: results; similarity metrics on colourful and flowers sat: results; similarity and fixing comments
Meeting with supervisor	1	10	1.166666667	Thu: about the comments on the report
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
		Total	37.16666667	37 h 10 min

Week 20 + Mon and Tue in week 21				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		55	0.916666667	
Report writing	70		70	Mon: comments, images in results, introduction to evaluation and interviews Tue: Wed: comments Thu: colour spaces, sim. met, comments fri: comments, solour spaces sat: many comments and TODOS. sun: usefulness results and comments
In-group meeting	1	10	1.166666667	Tue: discussing the set up of the thesis
Scrum-meeting with group members (part A)		10	0.166666667	Thu: very short as just writing
Scrum-meeting with supervisor/stakeholder (part B)	1		1	Thu: nice feedback from supervisors
Admin work	1	30	1.5	sat: prettifying documents
			0	
			0	
			0	
			0	
			0	
			0	
		Total	74.75	74 h 45 min

Mathias' time worked

Week 2, 3, 4				
Activity	Hours	Minutes	Decimal time	Comment
Meeting with supervisor	4		4	includes meeting in week 2 and 3
Self-learning	4	22	4.366666667	Learning rust and JavaScript
Document writing	18	30	18.5	writing on the project plan, making the gantt chart
Meeting with supervisor		51	0.85	got feedback from Jon Yngve on project plan
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
		Total	27.71666667	27 h 43 min

Week 5				
Activity	Hours	Minutes	Decimal time	Comment
Document writing	1	40	1.666666667	wed
Meeting with supervisor		50	0.833333333	wed
Document writing	7		7	wed
Admin work	2		2	thu
Self-learning	1		1	thu
Scrum-meeting with group members (part A)		55	0.916666667	thu: sprint meeting a
Scrum-meeting with supervisor/stakeholder (part B)		50	0.833333333	fri: sprint meeting b
Daily scrum		15	0.25	fri: daily scrum
Admin work	1		1	fri: edit product backlog
Self-learning	4	40	4.666666667	fri: learn about konva.js
			0	
			0	
		Total	20.16666667	20 h 10 min

Week 6				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		40	0.666666667	daily scrum
Self-learning	5	30	5.5	wed
Self-learning	5	40	5.666666667	thu: going through commit history and commenting
Meeting with supervisor		50	0.833333333	thu
Programming		50	0.833333333	thu: some coding
Programming	5	25	5.416666667	fri: make save and read functions
			0	
			0	
			0	
			0	
			0	
			0	
		Total	18.91666667	18 h 55 min

Week 7				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		10	0.166666667	daily scrum
Programming	5	35	5.583333333	wed: writing to file and selecting grou
Self-learning	1		1	wed: react
Scrum-meeting with group members (part A)	1	10	1.166666667	th: sprint meeting a
Scrum-meeting with supervisor/stakeholder (part B)		45	0.75	th: sprint meeting b
Programming	7	35	7.583333333	fr: project saving, reading
Programming	1	27	1.45	sat: polymorfic canvas element

Programming	5		5	sun
			0	
			0	
			0	
			0	
		Total	22.7	22 h 42 min

Week 8				
Activity	Hours	Minutes	Decimal time	Comment
Scrum-meeting with group members (part A)		30	0.5	daily scrum
Report writing	2	30	2.5	wed: theory
Programming	3		3	wed: helping oda
Programming	5	4	5.066666667	thu: helping oda
Programming	4	20	4.333333333	fre
			0	
			0	
			0	
			0	
			0	
			0	
			0	
		Total	15.4	15 h 24 min

Week 9				
Activity	Hours	Minutes	Decimal time	Comment
Programming	7		7	wed
Daily scrum		20	0.333333333	
Programming	2	30	2.5	thu
Scrum-meeting with group members (part A)		40	0.666666667	thu
Scrum-meeting with supervisor/stakeholder (part B)		50	0.833333333	thu
Programming	2	50	2.833333333	thu
Programming	2	25	2.416666667	fri
Programming	2	5	2.083333333	
Programming	2		2	sat
			0	
			0	
			0	
		Total	20.66666667	20 h 40 min

Week 10				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		10	0.166666667	
Planning/preparing user testing	4		4	Mon: preparing for user test
User testing	2	55	2.916666667	tu: 1st usertest
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
		Total	7.083333333	7 h 5 min

Week 11				
Activity	Hours	Minutes	Decimal time	Comment
Programming	2		2	mon: based on user tests
Daily scrum		25	0.416666667	
Admin work	1	25	1.416666667	th: make backlog items based on the user interview
Programming		55	0.916666667	th: undo redo experimanting
Report writing	2	10	2.166666667	th: writing on the theory scrum and UCD
Scrum-meeting with group members (part A)		50	0.833333333	th
Scrum-meeting with supervisor/stakeholder (part B)		45	0.75	th
Report writing	4	50	4.833333333	fr
Programming	1		1	fr
Programming	1		1	sa
			0	
			0	
		Total	15.33333333	15 h 20 min

Week 12				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		85	1.416666667	
Programming	7	40	7.666666667	mo: project file and the image representaiton in the file
Programming	7	5	7.083333333	ti: filter selected
Programming	8	15	8.25	we: luminance and contrast, save rotation, redo, filter window
Programming	6	10	6.166666667	tu: redo, save on close
Meeting with supervisor	1	20	1.333333333	
Programming	8	55	8.916666667	fr: redo/undo bugs
Programming	1	40	1.666666667	su: finished redo/undo
			0	
			0	
			0	
			0	
		Total	42.5	42 h 30 min

Week 13				
Activity	Hours	Minutes	Decimal time	Comment
Programming	1	5	1.083333333	mo: add mask filter
Programming	6		6	we: add filters
Programming	4		4	su: making changes based on feedback from UX designer
Daily scrum		25	0.416666667	tu
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
		Total	11.5	11 h 30 min

Week 14				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		40	0.666666667	
Planning/preparing user testing	3	30	3.5	
Programming		30	0.5	
In-group meeting		30	0.5	
User testing	2		2	tu: in person user test
Programming	4	20	4.333333333	we: make confirm close modal

Scrum-meeting with group members (part A)		50	0.833333333	
Scrum-meeting with supervisor/stakeholder (part B)	1		1	
Admin work		35	0.583333333	th: update backlogs before sprint meetings
Report writing	3	45	3.75	th: theory/computervision notes
Programming	7	30	7.5	fr: comment and linting
			0	
			0	
		Total	25.16666667	25 h 10 min

Week 15				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		60	1	
Report writing		30	0.5	mo: teory/cv
Programming	7	10	7.166666667	mo: sort imports and refactor windows and modals
Programming	5		5	tu: merging 5, dev and refactor
Document writing	2	20	2.333333333	tu: making diagrams
Programming		30	0.5	we: merging 5 into dev
Report writing	6	45	6.75	we: use case diagram, requirement
Programming	2	15	2.25	tu: refactor stageRef into context
Report writing	4		4	tu: reqirements
Meeting with supervisor	1	10	1.166666667	tu
Programming	4	20	4.333333333	fr: filter work in multiple selected
Report writing	2	25	2.416666667	
		Total	37.41666667	37 h 25 min

Week 16				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		60	1	
Programming	6	35	6.583333333	mo:
Programming	7	5	7.083333333	tu:
Programming	8	30	8.5	we:
Programming	4	50	4.833333333	th:
Scrum-meeting with group members (part A)		50	0.833333333	
Scrum-meeting with supervisor/stakeholder (part B)		55	0.916666667	
Programming	8	5	8.083333333	
			0	
			0	
			0	
			0	
		Total	37.83333333	37 h 50 min

Week 17				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		80	1.333333333	
Programming	7	15	7.25	mo: Made setElements for opening project files
Programming	6	90	7.5	
Programming	9	25	9.416666667	
Programming	6	0	6	
Programming	7	40	7.666666667	
Programming	4	35	4.583333333	sa
Programming	4	20	4.333333333	su
			0	
			0	
			0	
			0	
		Total	48.08333333	48 h 5 min

Week 18				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		80	1.333333333	
Planning/preparing user testing	1	10	1.166666667	mo: testing and preparing the user test
User testing	1	10	1.166666667	mo: doing one user test
Programming	3	5	3.083333333	mo
Programming	12		12	tu: last programming
Report writing	5	35	5.583333333	we: getting a overview / profwriting
User testing	2		2	we
Report writing	4	30	4.5	th:
Scrum-meeting with supervisor/stakeholder (part B)		45	0.75	th:
Scrum-meeting with group members (part A)		15	0.25	th:
Report writing	5	5	5.083333333	fr
Report writing	1	45	1.75	su
		Total	38.66666667	38 h 40 min

Week 19				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		70	1.166666667	
Report writing	7	30	7.5	mo
Report writing	7	40	7.666666667	tu
Report writing	7	30	7.5	we
Report writing	5	25	5.416666667	th
Report writing	4	30	4.5	fr
Meeting with supervisor	1		1	fr
Report writing	1	30	1.5	sa
Report writing	1	40	1.666666667	su
			0	
			0	
			0	
		Total	37.91666667	37 h 55 min

Week 20 + Mon and Tue in week 21				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		55	0.916666667	
Report writing	6	20	6.333333333	mo
Report writing	8	5	8.083333333	tu
In-group meeting	1	10	1.166666667	tu
Report writing	9	10	9.166666667	we
Report writing	8	10	8.166666667	th
Scrum-meeting with group members (part A)		10	0.166666667	th:
Scrum-meeting with supervisor/stakeholder (part B)	1	5	1.083333333	th
Report writing	5	15	5.25	fr
Report writing	6	55	6.916666667	sa
Report writing	9	25	9.416666667	su
Report writing	14	25	14.41666667	mo
			0	
			0	
			0	
		Total	71.08333333	71 h 5 min

Mikael's time worked

Week 2, 3, 4				
Activity	Hours	Minutes	Decimal time	Comment
Meeting with supervisor	4	0	4	Meeting time for week 2 and 3. Was so little, to make sense having it on its own
Admin work		47	0.783333333	Creating this timesheet
Document writing	16	40	16.66666667	Working on the Project plan. Working on the goals and scope parts.
Meeting with supervisor		51	0.85	Discussed the current state of the project plan. We will do some changes by the end of Monday next week and get some additional feedback.
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
		Total	22.3	22 h 18 min

Week 5				
Activity	Hours	Minutes	Decimal time	Comment
Document writing	8	42	8.7	Mon: Doing edits on the project plan based on the feedback from last week. Wed: Finishing the project plan, based on the last round of feedback.
Meeting with supervisor		50	0.833333333	Wed: Going over some last feedback on the project plan from both Davit and Jon.
Admin work	2	56	2.933333333	Thu: Setting up the github repository and inviting the other group members. Applying colour rules to the Product backlog for a more intuitive viewing experience. Setting up Visual studio and all the prerequisites to run tauri and open up the program from the code. Startet on documenting how the code base looked before we start adding to it. It will be displayed as a tree-view that can be inserted into the report. Fri: Booked rooms for the next week. Creating a kanban board and roadmap in github projects
Self-learning	5	38	5.633333333	Thu: Reviewing the code and learning what the different files do and how they interact. Looking at how react works. Fri: Continued study of the code. Learning how Github projects work, to make a kanban and roadmap view. Learning about KonvaJS, the framework used to create the canvas in the program.
Scrum-meeting with group members (part A)		55	0.916666667	Thu: Part A of the first sprint meeting.
Daily scrum		15	0.25	Fri: Daily Scrum
In-group meeting		30	0.5	Fri: Discussion
Scrum-meeting with supervisor/stakeholder (part B)		50	0.833333333	Fri: Scrum meeting with Davit and Jon
			0	
			0	
			0	
			0	
		Total	20.6	20 h 36 min

Week 6				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		30	0.5	Wed: Daily scrum meeting. Thu: Daily scrum meeting.
Self-learning	4	41	4.683333333	Wed: Learning more about KonvaJS Thu: Looking at how to videos and documentation on how to create a custom context menu using React and Konva. Fri: Learning more about how to create menu options for the navbar

Programming	11	12	11.2	Wed: Started working on the feature of exporting an image of the canvas. However that got sidetracked by the app preloading the images, and I cant get rid of them. Thu: Working on a custom context menu for taking a screenshot. Fri: Scrapping creating a custom context menu after the meeting with Davit and Jon. Created the function to grab a high resolution screenshot and linked it to a button on the navbar.
Meeting with supervisor		50	0.833333333	Wed: Update meeting with Davit and Jon. Got great feedback on questions we had
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
		Total	17.21666667	17 h 13 min

Week 7				
Activity	Hours	Minutes	Decimal time	Comment
Scrum-meeting with group members (part A)	1	20	1.333333333	Wed: Daily Scrum meeting. Thu: Sprint retrospective. Sprint planning
Programming	5	42	5.7	Wed: Finished moving the function of taking a screenshot of the canvas and loading an image to the canvas to under a file button.
Scrum-meeting with supervisor/stakeholder (part B)		55	0.916666667	Thu: Sprint 2 meeting
Report writing	6	49	6.816666667	Fri: Started writing the introduction of the bachelor report. Started slow because it took some time to get used to the structure of the overleaf template and how to use the features to make a great report from the start
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
		Total	14.76666667	14 h 46 min

Week 8				
Activity	Hours	Minutes	Decimal time	Comment
Report writing	4	40	4.666666667	Wed: Continued work on the introduction
Daily scrum		20	0.333333333	Wed: Daily Scrum Thu: Daily Scrum
Meeting with supervisor		45	0.75	Thur: Sprint update meeting
Programming	5	22	5.366666667	Fri: Workin on the score comparison window. Using tauri dialog for now to learn, but will change it out later
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
		Total	11.11666667	11 h 7 min

Week 9				
Activity	Hours	Minutes	Decimal time	Comment
Programming	6	11	6.183333333	Wed: Changed the score window to be a custom window on top of the program instead of being a tauri dialog.
Daily scrum		25	0.416666667	Thu: Daily scrum Fri: Daily scrum
Report writing	4	36	4.6	Thu: Continuing work on the introduction. Restructuring the introduction to follow the recommended structure provided by Tom on BB

Scrum-meeting with group members (part A)		40	0.666666667	Thu: Sprint retrospective and planning the next
Scrum-meeting with supervisor/stakeholder (part B)		52	0.866666667	Thu: Sprint 3 meeting with Jon and Davit
Planning/preparing user testing	4	30	4.5	Thu: Planning the user testing next week.
			0	
			0	
			0	
			0	
			0	
			0	
		Total	17.23333333	17 h 14 min

Week 10				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		10	0.166666667	Mon: Daily Scrum
Planning/preparing user testing	3	43	3.716666667	Mon: Continued planning of the user test for tomorrow
User testing	2	55	2.916666667	Tue: Holding the user test with Margrethe and Davit.
Report writing	1		1	Wed: Wrote a little bit on the report introduction
			0	
			0	
			0	
			0	
			0	
			0	
			0	
			0	
Sickness: From mid day Wednesday until the weekend		Total	7.8	7 h 48 min

Week 11				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum		25	0.416666667	Thu: Daily Scrum Fri: Daily Scrum
Report writing	14	28	14.46666667	Thu: Writing on the introduction Fri: Finished the introduction for now
Scrum-meeting with group members (part A)		50	0.833333333	Thu: Going over the last sprint and planning the next.
Scrum-meeting with supervisor/stakeholder (part B)		45	0.75	Thu: Meeting with Davit and Jon
Admin work	1		1	Fri: Working on the timesheet to calculate times
			0	
			0	
			0	
			0	
			0	
			0	
		Total	17.46666667	17 h 28 min

Week 12				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum	1	20	1.333333333	Mon: Daily Scrum Tue: Daily Scrum Wed: Daily scrum Thu: Daily Scrum Fri: Daily Scrum
Admin work	2	13	2.216666667	Mon: Finished setting up the summary worksheet which will summarize the work done during the project. How many hours spent on each task, total, and average.

Programming	36	21	36.35	Mon: Made the way to export the canvas as an image more intuitive. With text explaining the scaling and a Save As dialog to save the image wherever the user wants. Tue: Changed the modal to hide the info and show it when pressing a button. Added an alert when fs.WriteBinaryFile will not work because of a too big file. Did some work on the frontend Wed: Changed the look af the similarity metrics window to better match the programme. Made a new window for where the filters will be moved to. Together with Mathias, we moved the form to the new window and made sure it worked as intended. Thu: Fixed the layout of the contents inside the filter window. Fixed an error the caused the canvas to crash and reset when when trying to reset a filter. Changed how opening a modal/window works by changing it to use context. Started on how the find workspace button. Fri: Finished the find workspace button and functionality. Had to bugfix for a long time to get the width and height of the loaded in images to that the nearest image was centered on the screen instead of the top left corner being in the center
Meeting with supervisor	1	25	1.416666667	Thu: Update meeting before easter on what we have done since last week and overall progress. Discussing how things have and could have been done, and later features will be done.
Self-learning	5	30	5.5	Fri: Spent a long time to understand how the refactored image upload works and how to use it. Now it should be easier to work with for me.
			0	
			0	
			0	
			0	
			0	
			0	
			0	
		Total	46.81666667	46 h 49 min

Week 13				
Activity	Hours	Minutes	Decimal time	Comment
Programming	15	51	15.85	Mon: Actually finished the find workplace function. Npw it will zoom to the original level of the canvas so that even if it is extremely zoomed out, when pressing the button, the nearest fragment will show up on the screen instead of being a small dot on the screen. Fixed the error when opening the built version where a cmd window would also show up. Started to try to add power to tauri so that it could load in bigger images of fragments (with no success). Tue: Using what I learned about css variables, I changed the look of the sliders inside the filter window so that the background of the hue slider is of all the hues, the saturation takes in the hue and the slider changes based on that, etc... Thu: Added colour to the new filter slider and fixed look of new toggles. Black and white sliders now invert colour when the invert button is pressed. Added functionality to update the filter window sliders and toggles with the data of the fragment when selecting a new fragment to filter.
Self-learning	4	45	4.75	Mon: Looked at how tauri works and to understand how to manage the application to give it more power to work with. Found something on optimization, but it did not affect the images that could be loaded in. Tue: Learned more about CSS and variables to use inside CSS and set with JS. Looked more into optimizing tauri to load in images. Found a way to load in bigger images(in MB), but at the cost of image resolution(in pixels) when loaded in.
Daily scrum		25	0.416666667	Thu: Few daily scrums since the easter break started and everyone have differenct schedules
			0	
			0	
			0	
			0	
			0	
			0	

			0	
			0	
			0	
		Total	21.01666667	21 h 1 min

Week 14				
Activity	Hours	Minutes	Decimal time	Comment
Programming	26	5	26.08333333	Mon: Fixed layout and css of new changes from Mathias. Wed: Made changes to the filter window after feedback from the user test. Fixed a bug that deleted the fragment when deleting a number in the filter window. Started on the histogram feature in the score comparison window Thu: Continued work on the histogram feature. Fixed a bug where moving a window outside the stage caused it to be lost forever. Fri: Created the histogram component for the comparison window. Only selected images will be displayed in the window.
Planning/preparing user testing	1	14	1.233333333	Mon: Preparing for the usertest tomorrow
In-group meeting		30	0.5	Mon: Discussion about the user test and how we maybe should proceed with the bachelor, such as what we need to do and what we should focus on. Will be discussed more about with Davit and Jon on Thursday probably.
Daily scrum		40	0.666666667	Mon: Daily Scrum Wed: Daily Scrum Fri: Daily Scrum
User testing	2		2	Tue: Doing the second user test in-person in Oslo with Margrethe.
Scrum-meeting with group members (part A)		50	0.833333333	Thu: Going over the last sprint and planning the next.
Scrum-meeting with supervisor/stakeholder (part B)	1		1	Thu: Meeting with Davit and Jon
			0	
			0	
			0	
			0	
			0	
		Total	32.31666667	32 h 19 min

Week 15				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum	1		1	Mon: Daily Scrum Tue: Daily Scrum Wed: Daily Scrum Thu: Daily Scrum Fri: Daily Scrum
Programming	23	26	23.43333333	Mon: Started working on only using the original image in the hue histogram. Turned out to be difficult to get working. Scrapped it for now and worked on a button to refresh the histograms after a filter has been changed Tue: Created a similarity metric score calculation between the selected images. Started with using the hueValues array as is, but encountered faults since the arrays can be different lengths. Changed to counting the number of occurences between integers and normalizing the values. Calculating was now better since the arrays were the same length. Wed: Merged the histogram to dev so that it is ready for discussion on thursday and handled merge conflicts Fri: Finished the current version of the similarity metrics. Created a table for the similarity for each selected with every other element on the canvas. Then informs the user of which piece is the most similar
Report writing	16	30	16.5	Wed: Restructured some section to become chapters. Wrote about how the GUI of the program we inherited looked. Got screenshots of old and new components to use in the report Thu: Started writing on the GUI design for the current program Fri: Continued writing on the current GUI
Meeting with supervisor	1	10	1.166666667	Thu: Meeting with Davit and Jon. Discussing some last edits for the similarity metric and the report.
			0	
			0	
			0	

			0	
			0	
			0	
			0	
			0	
		Total	42.1	42 h 6 min

Week 16				
Activity	Hours	Minutes	Decimal time	Comment
Daily scrum	1		1	Mon: Daily Scrum. Tue: Daily Scrum. Wed: Daily Scrum. Thu: Daily Scrum. Fri: Daily Scrum.
Report writing	38	30	38.5	Mon: Continued writing about the new GUI. Almost finished. Tue: Finished writing about the new GUI. Researched what we can include in the other parts of the report and wrote down ideas for each blank chapter. Wed: Started writing on the development process. Started writing about how we chose Scrum as our development methodology. Thu: Finished writing on the development methology, and started writing about the meeting schedule Fri: Finished writing about the meeting schedule. Restructured the chapter some, and fixed the spelling and sentence structure in the theory chapter
Admin work		45	0.75	Wed: Had a reference group meeting for the bachelor course.
Scrum-meeting with group members (part A)		40	0.666666667	Thu: Going over the last sprint and planning the next.
Scrum-meeting with supervisor/stakeholder (part B)		55	0.916666667	Thu: Meeting with Davit and Jon
Admin work		30	0.5	Sun: Figured out why the pie chart under Summary sheet would not work, and created it.
			0	
			0	
			0	
			0	
			0	
			0	
		Total	42.33333333	42 h 20 min

Week 17				
Activity	Hours	Minutes	Decimal time	Comment
Report writing	31	19	31.31666667	Mon: Started writing about the tools we have used during the project. Wrote about collaboration, communication, and documentation. And a little bit about development tools. Tue: Finished writing about the development tools. Started writing about the general user testing background we will do. Wed: Wrote more about the preparation of the user tests Thu: Started writing about the results of the user tests. Fri: Almost got finished writing about the first 3 user tests.
Daily scrum	1	20	1.333333333	Mon: Daily Scrum Tue: Daily Scrum Wed: Daily Scrum Thu: Daily Scrum Fri: Daily Scrum
Admin work		27	0.45	Mon: Added more graphs to the summary page. A pie chart for each members activity work, and a collected bar chart for every activity by all members.
Planning/preparing user testing	4	10	4.166666667	Tue: Made the form for the general user testing, and made sure that the test procedure would work smoothly. Wed: Changed the user tests some to account for testing on two different images. Fri: Changed the first image to be used in the user tests to create bigger contrast between the first and second image. Keeping the result of the first 3
Programming		30	0.5	Tue: Found a "bug" that made the program slow when many fragments are loaded in. Fixed the bug, and now the user tests should work better.
User testing	1	51	1.85	Wed: Tested 3 users and got very similar results on each test.

Meeting with supervisor	1	5	1.083333333	Thu: Meeting with just Davit since Jon is in Hong Kong and could not attend the meeting
			0	
			0	
			0	
			0	
			0	
		Total	40.7	40 h 42 min

Week 18				
Activity	Hours	Minutes	Decimal time	Comment
Report writing	33	34	33.56666667	Mon: Finished writing about the user tests. Moved the usability testing part from the interview chapter to the testing chapter and proofread it. Prepared to write more about what happened during the tests. Tue: Started writing about the usability tests. Wed: Finished writing about the usability tests. Added changes to the design chapter. Thu: Started writing about the texrec project in theory. Is not needed probably so started writing in the installation chapter. Fri: Finished writing in the install chapter.
Daily scrum	1	20	1.333333333	Mon: Daily Scrum. Tue: Daily Scrum. Wed: Daily Scrum. Thu: Daily Scrum. Fri: Daily Scrum.
Programming		48	0.8	Mon: Identified and fixed a bug that caused the program to not load in images when loading from a project when the file path was invalid.
Admin work	2	40	2.666666667	Mon: Proofread word documents and turned them into pdfs to be used as appendices and added them to the report.
Scrum-meeting with group members (part A)		10	0.166666667	Thu: Going over the last sprint and planning the next.
Scrum-meeting with supervisor/stakeholder (part B)		45	0.75	Thu: Meeting with Jon about the previous and next sprint
			0	
			0	
			0	
			0	
			0	
			0	
		Total	39.28333333	39 h 17 min

Week 19				
Activity	Hours	Minutes	Decimal time	Comment
Report writing	35	54	35.9	Mon: Started writing about the results in the results chapter/discussion. Tue: I finished writing about the results for now and have started writing about rewriting the project or changing the running environment in future work. Wed: Continued writing about better similarity metric in future work. Thu: Wrote about better filters and export image sizes in future work. Fri: Wrote some in future work and started editing the introduction.
Daily scrum	1	10	1.166666667	Mon: Daily Scrum. Tue: Daily Scrum. Wed: Daily Scrum. Thu: Daily Scrum. Fri: Daily Scrum.
Programming		35	0.583333333	Thu: Added styling to sorting buttons on the similarity table.
Meeting with supervisor	1	10	1.166666667	Fri: Meeting with Davit and Jon Yngve discussing the report
			0	
			0	
			0	
			0	
			0	
			0	
			0	
		Total	38.81666667	38 h 49 min

Week 20 + Mon and Tue in week 21				
Activity	Hours	Minutes	Decimal time	Comment
Report writing	79	33	79.55	Mon: Edited the introduction chapter to reflect betterwhat we know now. Added the structure at the end which will need to be filled out. Filled out the abbreviations and glossary descriptions. Wrote about the similarity metric in the implementation chapter. Tue: Finished writing in future work. Fixed the figure system in results. Did changes based on comments from supervisor and product owner. Restructured the report based on the group meeting. Wed: Added more to sustainability in the discussion chapter. Read through the implementation chapter to make sure everything was fine after the restructure. Added captions to every code and command to make a list of them after the other lists of figures and tables. Fri: Wrote about project process in discussion. Happy 17th of May! Sat: Finished project process in discussion, about TexRec in the theory chapter. Sun: Wrote the abstract, sammendrag, and preface. Rewrote the usefulness results in the discussion chapter. Mon: Finalised the thesis and read through it carefully and fixed small mistakes.
Daily scrum		55	0.916666667	Mon: Daily Scrum Tue: Daily Scrum Wed: Daily Scrum Thu: Daily Scrum
In-group meeting	1	10	1.166666667	Tue: Discussing the report structure and how we could restructure it to have better flow and look better.
Scrum-meeting with group members (part A)		10	0.166666667	Thu: Sprint review of the last sprint.
Scrum-meeting with supervisor/stakeholder (part B)	1	5	1.083333333	Thu: Sprint review with Davit and Jon. Also discussing the report.
Admin work	3	30	0.5	Sat: Read through half of the meeting minutes to proofread before converting to pdf for the appendix Sun: Read through the last half of the meeting minutes and merged them all into a single pdf file for the appendix. Then I did the same for other missing appendix items.
			0	
			0	
			0	
			0	
			0	
			0	
		Total	83.38333333	83 h 23 min

APPENDIX I

MEETING MINUTES

Meeting minutes

Date: 18.01.2024
Time: 08.00 - 09.00
Place: Teams
Present: Mathias Iversen, Mikael R. Mathiassen, Oda K. F. Steinsholt, Davit Gigilashvili and Jon Yngve Hardeberg

1. What kind of meeting the report is about

Weekly status meeting with the supervisors

2. Which cases were discussed, and which decision was made?

- Not much time to work since the last meeting (2 min)
- Summary of the last meeting (15 min)
- Talked about the helping algorithm (20 min)
- Will range features next week and start making the plan (15 min)
- Next meeting next Thursday at 15.00 (8 min)

Meeting minutes

Date: 26.01.2024
Time: 15.30 – 16.20
Place: A132
Present: Mathias Iversen, Mikael R. Mathiassen, Oda K. F. Steinsholt, Jon Yngve Hardeberg

1. What kind of meeting the report is about

Meeting with the supervisor and discussing our project plan

2. Which cases were discussed, and which decision was made?

- We spoke about reference lists and styles. (8 min)
    - It is important to use the right style so that it says that the source is a bachelor thesis, for example.
    - The technical details on Overleaf decide that, so it is important to choose the right "style" that chooses the right fields in the list.
- Scrum: Talked about how the sprints should be organised. (5 min)
    - Planning of the sprint Is a part of the sprint.
    - The sprints are agile.
    - We will add more details about scrum in the plan, like the definition of a sprint.
    - Meeting with the product owner is also part of the sprint.
- Meetings in general (6 min)
    - We have meetings planned every week, so every second meeting would be a sprint meeting. We might consider not having the other meetings.
    - The sprint meetings should be partly alone with the group and partly with the supervisor/product owner. We should define later how these meetings should be organised.
- Davit's role (10 min)
    - In the performance goals, we did mention the archaeologists at UIO. We also discussed whether Davit should be mentioned, as he is our main stakeholder.
    - Both groups have an interest in a good project. We can think of Davit as the "salesperson" and the archaeologists as the "users that will buy the product".
- We should discuss our character goals. This can allow our supervisor to give better feedback (5 min)
- Coding languages (6 min)
    - The languages used in the Master thesis we base our work on are JavaScript, CSS, HTML and Rust. We first heard that Rust was the main language, so we were a little surprised when we saw that only 1% of the code was in Rust.

- Which language will the edge detection be based on? (5 min)
    - We have chosen to try JavaScript, but this was not discussed enough.
    - We talked earlier about using Python.
- Using the word “framework” (in chapter 2.3 in the project plan) might not work, as it sounds like something technological. (3 min)
- The process forward (2 min)
    - We will send the project plan to Davit and Jon on Monday. They will look at it on Tuesday and give feedback.
    - We will make the last changes and deliver it on Wednesday.

Meeting minutes

Date: 31.01.2024
Time: 08.00 – 08.50
Place: Teams
Present: Mathias Iversen, Mikael R. Mathiassen, Oda K. F. Steinsholt, Jon Yngve Hardeberg and Davit Gigilashvili

1. What kind of meeting the report is about

Meeting with the supervisor and stakeholder to discuss our project plan.

2. Which cases were discussed, and which decision was made?

This was discussed further from the comments on the project plan: (50 min)

- We should mention that we will read about the literature and choose what to do.
- Discussing using AI tools, write about it. For programming?
    - Will mention this in the final report.
- Do we want to use cut textiles that are not old? Describe the process. Nice for visualisation in the report. Add to process.
    - We do not see the need to write about this in the project plan but will write about it in the final report.
- Write more about why they need to visualise the fragments, conservation, etc.
- Mention Python, machine learning, and image processing in learning goals.
- 3.1 is a little bit chaotic. Make clear what it is pictures of.
- Discussed saving of the files. Want the saved file to be small. Only keep the “changes” in the file.
- Memory usage optimisation: load the high resolution first when they zoom in.
    - Don’t feel like mentioning this now, as we don’t even know if it will be possible.
- Might need a summary for the risk section and a caption for the table (closer to the figure); refer to the text. Refer to the explicit risks and colours. Histogram of overall risk. How many of each colour? Can also write about it in the summary.

Meeting minutes

Date: 01.02.2024
Time: 12.05 – 13.00
Place: K108
Present: Mathias Iversen, Mikael R. Mathiassen and Oda K. F. Steinsholt

1. What kind of meeting the report is about

Sprint meeting part A

2. Which cases were discussed, and which decision was made?

Planning of Sprint 1

- **Why** is this sprint valuable? (20 min)
    - As this is the first sprint, everything is new, and we will learn a lot from this sprint. We will bring this knowledge to the next sprints.
    - Because we want to start with features that we know have a high value to the users.
    - Sprint goal: To learn about and understand the sprint process in the scrum by having a “jump start”.
    - We will learn about the existing program.
- **What** can be done in this sprint? (25 min)
    - Saving and loading.
    - Importing and exporting.
    - Export images of the canvas (“screenshot”).
    - Select multiple images.
    - See backlog for more info about each task.
- **How** to get the work done? (10 min)
    - At the beginning of this sprint, we will work with the development together.
    - First day: develop/find file type.
    - Second day: save functions for the selected file type.
    - Third say: load-functions for the selected file type.
    - Fourth day: use files to export an image of canvas.
    - After that: Select multiple images.

Meeting minutes

Date: 02.02.2024
Time: 09.15 – 10.05
Place: A221 / Teams
Present: Mathias Iversen, Mikael R. Mathiassen, Oda K. F. Steinsholt, Davit Gigilashvili and Jon Yngve Hardeberg

1. What kind of meeting the report is about

Sprint meeting part B

2. Which cases were discussed, and which decision was made?

- We discussed the backlog and introduced it to the supervisor and the stakeholder. (10 min)

- We informed them of which tasks we decided to do in sprint 1: (10 min)
    - Saving and loading
    - Importing and exporting.
    - Export images of the canvas (“screenshot”).
    - Select multiple images.
- We talked about some of the features on the list. (30 min)
    - Export images:
        - Can be done without using a lot of storage space. Only save the changes and links to the images.
    - Textbox adding:
        - Metadata might be used.
        - The text would contain annotations that the archaeologists do today.
        - They also wanted free notes anywhere on the canvas/independent of the images. Like a post-it note, that can also be grouped into an image/cluster of images.
    - High-resolution images.
        - Very hard, as the size is up to 1GB per image.
    - Motif score.
        - Machine learning/computer vision. Can build on something made before.
        - Stress/contrast stretching.

Meeting minutes

Date: 08.02.2024
Time: 15.15 – 16.00
Place: A212
Present: Mathias Iversen, Mikael R. Mathiassen, Oda K. F. Steinsholt, Jon Yngve Hardeberg and Davit Gigilashvili

1. What kind of meeting the report is about

Meeting with the supervisor and stakeholder

2. Which cases were discussed, which decision was made?

- Feedback on backlog from Davit. Will be given within 2 days. (2 min)
- Screenshot function: button or from the menu? (10 min)
    - Depends on how often this feature will be used.
    - Also depends on how much time it will take to make it.
    - Natural to look for this in the "File" drop-down menu.
    - If we remove the "disable default" image capture and screenshot feature, it will be on the right-clicking the canvas.
        - This feature keeps the blue line if an image is "selected".
    - This is enough for our project, according to Davit.
- We still need a feature to save a higher-resolution image/screenshot/file of the canvas. (10 min)
    - We discussed how to catch/save that info. Can use a matrix.
- Need to make file button. Should be easy. (3 min)
- Canvas to file; could be exported to a JSON file. (10 min)
    - Will be fine as long as it fulfils the objective.
    - The files should be small in size.
    - Images should be shared in the same structure as the project files for it to work.
    - Vector file or image file.
    - Can handle drawing and images the same way.
- We have used a lot of the time until now trying to understand the code. This is harder than we thought, as the code is not commented. (5 min)
    - Some problems were solved by changing the "identifier" value. Could be something about the metadata.
- For more guidance on JavaScript and React, we can ask Mariusz Nowostawski. Email: mariusz.nowostawski@ntnu.no (3 min)
- We can also contact Casper Guldbrandsen. (2 min)

Meeting minutes

Date: 15.02.2024
Time: 14.00 – 15.10
Place: K210
Present: Mathias Iversen, Mikael R. Mathiassen and Oda K. F. Steinsholt

1. What kind of meeting the report is about

Sprint meeting part A

2. Which cases were discussed, and which decision was made?

Looking back at Sprint 1

- What have we done: (10 min)
    - 1. Halfway done.
    - 3. Halfway done.
    - 4. Halfway done, closely related to 3.
    - 7. Done.
- Sprint retrospective: (25 min)
    - We talked through the last sprint's plan.
    - We ended up working more alone than what was planned, but we thought that was a nice solution.
    - We still worked a little bit together, and it was nice when you didn't understand something.
    - We want to have a low threshold for asking to work together for a little time if that is necessary.
    - We want to get in contact with someone who knows JS and React. We are not able not able to get so much help from our supervisors on that exact area.
    - We put on a little bit too much work compared to the time and knowledge.
    - We need to define the meeting minutes in the sprint meetings, what to discuss, when, and for how long.

Planning of Sprint 2

- **Why** is this sprint valuable? (10 min)
    - We now have a feeling of how much work we have time to do. Therefore, the planning will be easier.
    - Goal: Make our product ready for usability testing
- **What** can be done in this sprint? (15 min)
    - Finish 1 and 3 from the last sprint

    - Task 4 from the last sprint
    - Will do number 5, as most of it will be done after completing number 1.
    - 2
    - We discussed doing number 19, but we will wait with it.
- **How** to get the work done (10 min)
    - Mikael will do number 2.
    - Oda will finish numbers 1 and 5.
    - Mathias will finish 3 and 4.

Meeting minutes

Date: 15.02.2024
Time: 15.15 – 16.05
Place: A221
Present: Mathias Iversen, Mikael R. Mathiassen, Oda K. F. Steinsholt and Davit Gigilashvili

1. What kind of meeting the report is about

Sprint meeting part B

2. Which cases were discussed, and which decision was made?

- We discussed and showed what was done in sprint 1. (25 min)
    - Task 7 (“Take screenshot”) is done. We showed how it works to Davit. We discussed that we might need some better names and scale values.
    - Task 1 (select multiple images) is about halfway done. We have implemented a “groupNode” JSX file in addition to the already existing “imageNode” file. This will handle the grouping of elements, and the selected images will be one type of group.
    - Task 3 (saving and loading) is halfway done. We decided to use JSON file format to save a project. We can now save and read from a file. We have not decided on the attributes representing a project in the file. (Basically, how the file would look like).
    - Task 4 (project file) is closely related to task 3, and therefore being worked on together.
    - We have not done much coding compared to what was planned, but we have used a lot of time understanding and teaching ourselves about the existing code and the languages. Most of the coding we did was done in the last 3 working days of the sprint.
- The group added a feature to the list: “Find my canvas”. We talked about why it could be important. (5 min)
- We informed the stakeholder which tasks we decided to do in sprint 1 (20 min).
    - 1, 3, and 4 from last sprint
    - 2 and 5 are new.

Meeting minutes

Date: 22.02.2024
Time: 16.00
Place: A221 / Teams
Present: Mathias Iversen, Mikael R. Mathiassen, Oda K. F. Steinsholt, Davit Gigilashvili and Jon Yngve Hardeberg

1. What kind of meeting the report is about

Meeting with stakeholder and the supervisor.

2. Which cases were discussed, and which decision was made?

- We talked about meeting for the usability testing. The 5th of March at 10-12 is a nice time. The 2nd of April works the best for us all at 10. (5 min)
- We have worked with saving/loading files and found out that we need to restructure the code. Hard to know what the best way is. A lot of back and forth. (15 min)
    - Old way: classic object-oriented. Simple for us.
    - New way: functional component. Harder for us to understand.
- Where are we looking for solutions? Konva.js library online, google (5 min)
    - Maybe other places to look?
- Should contact someone with more knowledge about React and JavaScript (5 min)
- We talked more about the sprint backlog. Most of the tasks are related except the score comparison window. Maybe we could add some simple computer vision instead of using dummy data. (5 min)
- It is hard to know how much to work when no one is expecting anything concrete from us. There will always be room for improvement. Must be happy if we did our best within a given time and given resources. (8 min)
- Better with fewer good functions than many halfway working. (2 min)

Meeting minutes

Date: 29.02.2024
Time: 12.00 – 12.40
Place: The hall outside Databasen
Present: Mathias Iversen, Mikael R. Mathiassen and Oda K. F. Steinsholt

1. What kind of meeting the report is about

Sprint meeting part A

2. Which cases were discussed, and which decision was made?

Looking back at Sprint 2

- What have we done: (5 min)
    - 1. Almost done
    - 2. Done
    - 3. Saving and loading to a JSON file is done, but not loading it into a new project
    - 4. Depends on 3, but will be done at the same time
    - 5. Not started on it, but there will not be much to do after completing 1.
- Sprint retrospective: (15 min)
    - Mikael's part has worked well.
    - We feel like we are slightly behind on the report writing. We had a slow start, but it is progressing steadily.
    - This sprint has been mentally tough because we have struggled a lot with task 1. We initially thought the difficulty would be low, but there were a lot of hard and complex tasks that needed to be done because we needed to refactor a lot.
    - It is hard to find someone with experience in React, that we can ask for help.
    - We have learned a lot, though, and we think things will be easier from now on.
    - It was smart to have two people working on the refactoring.

Planning of Sprint 3

- **Why** is this sprint valuable? (5 min)
    - We will get user input through usability testing.
    - We will be able to catch up on features that have been harder to finish than expected and the thesis report.
    - The sprint goal will be: "To get a good understanding of the user's needs in our program".

- **What** can be done in this sprint? (7 min)
    - Usability testing
        - Planning
        - Doing
        - Writing about and discussing the results
    - Finish the unfinished tasks from the last sprint.
    - Teach us selves about computer vision.
    - After finishing the usability testing, we will decide if we will start these tasks:
        - 6, colour wheel.
        - 19, histogram score.

    See backlog for more info about each task.

- **How** to get the work done? (3 min)
    - Mikael will start planning the usability test with input from Oda and Mathias.
    - Oda will work on finishing the task 1.
    - Mathias will work on 3 and 4.
    - Then, we will host the usability tests.
    - After the usability tests, we will write about the results.
    - Parallelly, we will finish task 5.
    - After that, we will start on tasks 6 and 19.

Meeting minutes

Date: 29.02.2024
Time: 13.15 – 14.05
Place: A262
Present: Mathias Iversen, Mikael R. Mathiassen, Oda K. F. Steinsholt and Davit Gigilashvili

1. What kind of meeting the report is about

Sprint meeting part B

2. Which cases were discussed, and which decision was made?

- We discussed and showed what was done in sprint 2. (25 min)
    - Score comparison window is done.
    - 1 is almost done. We can now select the right way with our hard-coded rectangles, but not with the images.
    - 3 and 4 depend on the refactoring.
    - Not started on 5.
    - We wrote more on the report. The introduction is more than halfway done.
    - We discussed seeking help and different ways to do it. Nothing wrong with using several methods.
- We showed our backlog for sprint 3 (25 min).
    - We talked about the sprint goal.
    - We discussed the tasks, including the usability test and the two new tasks.
        - We discussed how the “colour wheel” should be implemented, and what the user should be able to do. In that process, we took a look at the Konva.js documentation.
        - We discussed whether to use Python or the Konva.js framework in the computer vision part.
        - In the histogram, RGB, or other colour spaces. HSV, for instance.

Meeting minutes

Date: 14.03.2024
Time: 14.05 – 14.50
Place: Hall outside Databasen
Present: Mathias Iversen and Mikael R. Mathiassen

1. What kind of meeting the report is about

Sprint meeting part A

2. Which cases were discussed, and which decision was made?

Looking back at Sprint 3

- What have we done: (5 min)
  - We have successfully planned and executed the first usability test with Margrethe over teams.
  - Selecting multiple fragments is now finished and works.
  - Saving and loading a project is finished.
- Sprint retrospective: (10 min)
  - After last week's usability test, all group members got sick through the weekend. Mathias was on a game expo, whilst sick. And we had our exam on Wednesday this week. This means that we did not have the opportunity to do much work on the bachelor.
  - Despite this, we managed to do the tasks we had to do, but we could not start the tasks we were unsure of from the beginning if we had time to do them anyway.
  - Depending on how you look at it, the usability test could have gone better. We asked questions that came to mind instead of asking them afterwards. The testing also took longer than expected.
  - On the task of selecting multiple fragments, taking a step back and starting from the beginning was a huge help. Bringing the knowledge gained previously and doing it differently from the beginning worked.

Planning of Sprint 4

- **Why** is this sprint valuable? (5 min)
  - This sprint is longer, even though it is during easter. We are also finished with our other subject, so we can now focus 100% on the bachelor's.
  - We have just got a lot of great feedback from a user.
  - This sprint leads up to the last usability test.

- **What** can be done in this sprint? (12 min)
    - Locking pieces together.
    - Change the colour filter to be more intuitive.
        - Using HSV, contrast and brightness
        - Filter only selected fragments.
    - Make a histogram score for the images.
    - Make the scaling for taking a screenshot more intuitive. And selecting where it should be saved. Instead of just in downloads.
- **How** to get the work done? (13 min)
    - Mikael will work on the more intuitive image export.
    - Mathias will start editing the colour filter.
    - Oda will start working on the histogram score for the fragments.
    - Mikael will improve the way the fragment scores are displayed.
    - We will all try to catch up on some report writing.

  After, we will add more score functionality.

Meeting minutes

Date: 14.02.2024
Time: 15.15 – 16.00
Place: A221
Present: Mathias Iversen, Mikael R. Mathiassen, Jon Yngve Hardeberg and Davit Gigilashvili

1. What kind of meeting the report is about

Sprint meeting part B

2. Which cases were discussed, and which decision was made?

- We went over the updated backlog (10min).
    - Went over the new items that were added after the usability test.
- Discussed what we have done in sprint 3. (5min)
    - Added multiple selection using the shift button.
    - Changed name of the image export button and similarity score window.
    - Added offset on multiple imported images so that they are not right on top of each other.
- We showed our backlog for sprint 4 (5 min)
    - Discussed the next sprint.
    - Going over the tasks for this sprint from the backlog.
- Discussing solutions for how to show similarities and indicating to the similar fragments (25min)
    - Select 1 and compare everything else.
    - Select multiple and compare all selected.
    - How the similarity metric window should be structured.
    - Choosing the metrics you want to search/compare with.
- Discussing the next meeting (5min)

Meeting minutes

Date: 21.03.2024
Time: 15.15-16.35
Place: A221 / Teams
Present: Mathias Iversen, Mikael R. Mathiassen, Oda K. F. Steinsholt, Davit Gigilashvili and Jon Yngve Hardeberg

1. What kind of meeting the report is about

Meeting with the supervisor.

2. Which cases were discussed, and which decision was made?

- Going over the new filter feature. (15 min)
    - How easy is it to change how the window shows up.
    - Discussing the change from filtering all images selected to only the right-clicked one.
    - We should maybe add the ability to filter all selected images but acknowledge that previous filtering in specific images will be lost to match the filtering of all the images. Maybe save the filtering configuration on a fragment and then load the configuration to all selected with a keystroke like Ctrl + v.
    - Having no way to apply a filter to multiple images is not good, and we should find a way to do that.
      Is slow on bigger images.
- Discussing exporting an image of the canvas. (10 min)
    - How the resulting image is generated. Probably with a virtual screen. We should save/edit the ppt/ppi of the image so that it matches based on the zoom factor on the screen and the size of the resulting image.
- When zooming in it smooths the image out, would like to see the individual pixels. (5 min)
- Saving the project uses file path now. (5 min)
- How will we solve exporting the project when we now don't save the whole image in the project? Need to find or think of a solution. (5 min)
- Image loading is limited by size. The Tauri setting could maybe be adjusted to increase the memory/CPU power allocated to the program. (10 min)
- Discussing undo redo. (15 min)
    - How it works by saving the state of the canvas.
    - Bugs: Saves every step on the slider of the filter.
- Discussing the bounding box. (20 min)
    - Should we recalculate it on load or should Davit do it in preprocessing.

- Konva.js has a cropping feature that could work.
- Cropping of the images by Davit will reduce the memory needed to load the images onto the canvas.
- Davit can add a step in his preprocessing to also crop an image.

Meeting minutes

Date: 04.04.2024
Time: 10.05 – 10.55
Place: Databasen
Present: Oda K. F. Steinsholt, Mathias Iversen and Mikael R. Mathiassen

1. **What kind of meeting the report is about**

Sprint meeting part A

2. **Which cases were discussed, and which decision was made?**

Looking back at Sprint 4

- What have we done: (15 min)
  - We went through the sprint 4 backlog. We have finished most of the tasks, but the remaining are 5, 19, 19.1, 2.2, and 3.1.1.

- Sprint retrospective: (10 min)
  - We have gotten a lot of tasks done this sprint, even with sickness and other absences in the group during it. We are, therefore, well on our way to having done as much as possible with our time on the program.
  - Comparing this sprint to the earlier ones, we see that we should have been stricter about putting in 8-hour workdays on the days we were not working on the other course. If we did, we might not feel as pressed for time as we are.

Planning of Sprint 5

- **Why** is this sprint valuable? (5 min)
  - This is the last sprint containing programming. We, therefore, need to finish and "close" the programming tasks.
  - Once we will finish the programming tasks, we will focus more on writing the report.
  - We also have a usability test behind us, with new and fresh feedback, to build on.

- **What** can be done in this sprint? (10 min)
    - We only have two big and “new” items.
        - Finishing task 5, locking and unlocking pieces together.
        - Histogram score only showing scores from selected fragments, 19, 19.1, and 2.2.
    - Other programming tasks will be either usability test feedback tasks or cleaning up the code, such as refactoring and linting.
    - Write a lot on the report:
        - Finish “Theory”.
        - Finish “Kravspesifikasjon”.
        - Write in the results chapter about the usability test.

- **How** to get the work done? (10 min)
    - We will continue working together on campus to be more efficient and discuss what we are working on more easily.
    - Mathias will write on the report and focus on refactoring the code to be more readable and efficient.
    - Oda will write on the report and work on task 5, locking and unlocking pieces together.
    - Mikael will write on the report and continue work on tasks 19, 19.1, and 2.2, histogram, and only comparing selected fragments.

Meeting minutes

Date: 04.04.2024
Time: 11.10 – 12.10
Place: A232
Present: Oda K. F. Steinsholt, Mathias Iversen, Mikael R. Mathiassen, Jon Yngve Hardeberg and Davit Gigilashvili

1. **What kind of meeting the report is about**

Sprint meeting part B

2. **Which cases were discussed, and which decision was made?**

- We went through the backlog for sprint 4 and quickly talked about the tasks we had finished. (13 min)
- We talked a little bit about the report writing. (10 min)
    - It is helpful to have the meeting minutes and the backlog.
    - Making diagrams will be a challenge.
    - We can share the report with the supervisors after each chapter is done to receive feedback. This is nice, as we haven't received much feedback on it until now.
- We talked about GitHub (5 min)
- Catching Jon up on what we have discussed since he joined late (5 min).
- We discussed the backlog of the next sprint (2 min)
    - Commenting and documenting code is important.
- The plan for the bachelor thesis was also discussed (10)
    - We started sprint 5 earlier, as a week was added to the usability test period.
    - Therefore, we thought about adding sprint 6, which would eat a little bit of the last “writing sprint”.
    - We implemented this, and sprint 6 will be a sprint where we can do more bug fixes and maybe some coding if necessary. We will still do a lot of writing, though.
- We talked about the last presentation and evaluation (10 min)

Meeting minutes

Date: 11.04.2024
Time: 15.15 - 16.25
Place: A221
Present: Mathias Iversen, Mikael R. Mathiassen, Oda K. F. Steinsholt, Davit Gigilashvili and Jon Yngve Hardeberg

1. What kind of meeting the report is about

Meeting with stakeholder/supervisor

2. Which cases were discussed, and which decision was made?

- We started to talk about the report writing. (5 min)
    - We are making a nice progress.
- We discussed using cut textiles to show how the program works in the report. (8 min)
- Usability testing with puzzle pieces and on other people would be smart. (2 min)
    - We have only tested with one person. She is the actual end user, so it makes sense, but having a few more testers is still normal.
- We talked about the similarity metrics window. (45 min)
    - How much info do we want to show the user? If we want to show all the information, we could use a table.
    - By now it is easy to implement other similarity metric types, which might not be the “right” ones.
- Mikael talked for a long time about SVG images and how the histogram function updates. (10 min)
    - Every update creates a new SVG image, so there was a discussion of the update happening every value change or when the user wanted to do it manually.

Meeting minutes

Date: 18.04.2024
Time: 14.10 – 15.00
Place: Hallway outside Databasen
Present: Mathias Iversen, Mikael R. Mathiassen and Oda K. F. Steinsholt

1. **What kind of meeting the report is about**

Sprint meeting part A

2. **Which cases were discussed, and which decision was made?**

Looking back at Sprint 5

- What have we done: (10 min)
    - Coded: 19 and 19.1, histogram score, 2.2, 32 with several sub-tasks and changes from the usability test.
    - Wrote on the report: theory, requirements, result, design (old and new), development process.
    - A lot of refactoring was needed in order to do task 5 in the coding. The task is, therefore, under development.
- Sprint retrospective: (10 min)
    - We now feel that we are "done" with the program. We do have one task left we hope to finish, but we are proud to present the program as it is now if that is the final result.
    - We, therefore, feel like we reached the sprint goal: "Finish the programming in a great and presentable way."

Planning of Sprint 6

- **Why** is this sprint valuable? (5 min)
    - Finishing and evaluating the program then switching focus to the report writing.

- **What** can be done in this sprint? (7 min)

    - We have talked about testing. We will do some simple tests with “non-archaeologists” and “home textiles”.
    - Continue writing on the report.
    - Decide if there is enough time to finish the refactoring, if yes, do it in a timely manner.
    - Update the README.md file
- **How** to get the work done? (8 min)
    - Oda and Mathias will continue refactoring. If it takes way too much time, they will give up.
    - Then they will join Mikael on the report writing.
    - Mikael will make a usability test form, and they all will contribute to usability testing the program with people from the school.
- **Other** discussion (10 min)
    - We discussed what we wanted to discuss with the supervisor and stakeholder.

Meeting minutes

Date: 18.04.2024
Time: 15.20 – 16.15
Place: A221
Present: Oda K. F. Steinsholt, Mathias Iversen, Mikael R. Mathiassen, Jon Yngve Hardeberg and Davit Gigilashvili

1. What kind of meeting the report is about

Sprint meeting part B

2. Which cases were discussed, and which decision was made?

- We presented how the histogram score is shown now. (20 min)
- We talked about how, in a future version, we could have integrated MATLAB or Python. (5 min)
- The filter implementation is now a little bit changed; the button is in a different place, and the user can enable/disable the filters from the tool dropdown menu (10 min)
- We have worked on locking pieces together. We discovered that a lot of refactoring needs to be done and started on that. (10 min)
- For the next sprint, we have a few tasks: one coding task (task 5), writing the report, some usability testing, and updating the README.md file (5 min).
- Is it wrong to say “we” in the report? Have a look at other theses. (5 min).
- We talked about the reviewing of the report and the presentation. (10 min).

Meeting minutes

Date: 25.04.2024
Time: 15.15 - 16.20
Place: A232
Present: Mathias Iversen, Mikael R. Mathiassen, Oda K. F. Steinsholt, Davit Gigilashvili and Jon Yngve Hardeberg

1. What kind of meeting the report is about

Meeting with stakeholder/supervisor

2. Which cases were discussed, and which decision was made?

- We discussed the end presentation and the last days before submitting the thesis. (10 min)
    - We will try to have a test presentation, for example, on the 31st of May.
    - We will try to be "done" with everything on the 16th.
- We also talked about the fact that we did continue to code on an already-made project. (10 min)
    - We have learned a lot from it, as all other school projects have been made from scratch.
    - It is what people normally do in work situations.
- Copyrights for an image we used (5 min)
    - Try to use Pixabay. We can use many those images for free.
- The similarity metrics (5 min)
    - They worked nicely for a colourful image that we found and segmented and linked some of the neighbouring pieces together.
    - They did not work so well for the textile photos we have.
    - In most cases, the human eye still works better than our chosen similarity metrics.
- The usability tests we hosted this week (5 min)
    - The thought was that the program would stay on a computer in the museum and that guests could try to match pieces.
- The program is slow when loading images while filters are enabled (5 min)
- Tauri (10 min)
- Writing is going nice (10 min)
    - Davit and Jon Yngve will start reading and commenting around the 6th of May.
- The science results from the archaeologists (5 min)

- Can be used for testing our program.

Meeting minutes

Date: 02.05.2024
Time: 14.55 – 15.05 + 16.05 - 16.15
Place: At home on Discord
Present: Mathias Iversen, Mikael R. Mathiassen and Oda K. F. Steinsholt

1. What kind of meeting the report is about

Sprint meeting part A

2. Which cases were discussed, and which decision was made?

Looking back at Sprint 6

- What have we done: (5 min)
  - Finished the code.
  - Gave up on the refactoring but adopted some of the work into the program.
  - We have done some other refactoring in the score table and other small details.
  - We have modified the component tree diagram to match the program as it is today.
  - We have also written a lot on the report.
  - We hosted a few usability tests.
- Sprint retrospective: (5 min)
  - It was sad to give up on the refactoring, and a hard decision to make. We are happy now that we are on the other side. It was the right decision to make.
  - All in all, we are happy with the end code; even though many things could be better, there will always be things that can be fixed regarding programming.

Planning of Sprint 7

- **Why** is this sprint valuable (5 min)
  - This sprint is valuable since it is the last sprint of the bachelor thesis. When this sprint is over at the end of the 16th of May, there are only 20 working hours left until the delivery deadline.
- **What** can be done in this sprint? (2 min)
  - In this sprint, we will all focus on the report writing.
  - On the 6th of May, Davit and Jon will start reading through the thesis to give us their first feedback, so we will need to have most ready by then.
- **How** to get the work done? (3 min)
  - We will all write on the report every day.
  - We still need to finish writing the implementation, installation, results, discussion, conclusion chapters, and abstract.

Meeting minutes

Date: 02.05.2024
Time: 15.15 – 16.00
Place: Teams
Present: Oda K. F. Steinsholt, Mathias Iversen, Mikael R. Mathiassen, and Jon Yngve Hardeberg

1. What kind of meeting the report is about

Sprint meeting part B

2. Which cases were discussed, and which decision was made?

- We discussed the refactoring (11 min).
    - We did not manage to finish the refactoring.
    - We moved as much from the refactoring to the main branch as possible without breaking the program.
    - Discussing what the refactoring was supposed to do.
    - Since the refactoring did not work, we did not finish the locking pieces together item.
    - It is now possible to select multiple images and apply the same filter to all the selected images.
- Showing the current state of the program (13 min).
    - The new way images are loaded into the program.
    - Images inside the similarity metrics show the filter applied to them.
    - Showing an example of the usability test with the flower and colourful images.
    - Talking about the similarity metrics table and how it is structured.
- Talking about the report (21 min).
    - Talking about the readthrough of the report by Jon and Davit on Monday.
    - Discussing mentioning the TexRec project in the theory part. Will write more about it in the introduction.
    - Talking about the design chapter. How it is just about the GUI now and if we should add a part about the technical design here, or in the implementation chapter.
    - Can “introduce” what info will come next/later and describe why it is placed at that location in the report.
    - Be careful of writing about the results too early (except for in the abstract)

Meeting minutes

Date: 10.05.2024
Time: 10.05 - 11.05
Place: A221
Present: Mathias Iversen, Mikael R. Mathiassen, Oda K. F. Steinsholt, Davit Gigilashvili and Jon Yngve Hardeberg

1. What kind of meeting the report is about

Meeting with stakeholder and supervisor. Talk about the report

2. Which cases were discussed, and which decision was made?

- Introduction (5 min)
  - Make it clear what features were there before and what we made!
  - More formal language. For example, we used the word “cool” to explain something.
- We exploit our weaknesses too much. (5 min)
  - Formulate it in a more beneficial way for us. NO REGRETS!
  - When we fail something, focus on what we learned, which is a good thing.
- Argue that we relied on the end users' wishes.
  - That was a big deal for what we chose!
  - The users don’t want an automatic solution but something that could help them.
- Computer vision (5 min)
  - Research paper (PhD). It needs more than a PhD to solve the whole problem with the fragments. We can talk a little bit about that.
  - Reference more PhDs or other research papers in the introduction to strengthen the arguments. IMPORTANT!
- Usability tests: (10 min)
  - More bullet points of what we learned; sort the information.
  - Have a discussion about what we implemented from everything we learned in the usability tests.
  - It is important to “sum it up” at the end/discussion and not just mention it along the way.
  - Explain why we don’t use the fragments in the last user test round. Why did we use normal people? Make some good arguments here. Say that we wanted a broader perspective here.

- Future work (5 min)
    - In a research paper, one would not have too many details about future work.
    - Hard to know how much value it has to us (with many details).
    - But useful to the next group
- Results (10 min)
    - Explain the **usefulness** of our results. What can we do now that we have this feature?
    - At this point, mostly of our discussion is towards the goals.
    - Discuss the metrics more in detail, where they work well and badly. Strengths and weaknesses of the existing metrics. Explain why it works and does not work.
    - Can talk about working with archaeologists, as they are from a different field than us. How is the communication? Good for a later job.
    - Make a "coordinate system" on the puzzle image to refer to a single image.
- Presentation of the thesis for the archaeologists (5 min)
    - They could join the pre-presentation, but it might be too technical.
    - We can have a shorter presentation (e.g. 30 min) for them, more directed towards their use.
- The release (5 min)
    - Include a dataset of cut textiles.
- Order of the chapters (10 min)
    - Design might be too early as it describes our results on the design.
    - Maybe it should be early, but we talk about how it was vs. how we want it to be (instead of how it became) and then talk about how it actually became in the result section.
    - Agile scrum makes this hard. **We will look at other theses that also use agile scrum to see how they do it.**

Meeting minutes

Date: 16.05.2024
Time: 14.35 - 14.45
Place: Databasen
Present: Mathias Iversen, Mikael R. Mathiassen and Oda K. F. Steinsholt

1. **What kind of meeting the report is about**

Sprint retrospective meeting part A

2. **Which cases were discussed, and which decision was made?**

Looking back at Sprint 7

- What have we done: (5 min)
  - Written a lot on the report.
  - A few very small code changes
  - Edited details in the report based on comments from Jon Yngve and Davit
- Sprint retrospective: (5 min)
  - We have made big steps in the report. Written 11000 words on the report, added 30+ pages, and now close to finishing it.
  - We have had good routines, working together on campus for around 8 hours daily to progress.

Meeting minutes

Date: 16.05.2024
Time: 15.15 – 16.20
Place: A221 / Teams
Present: Oda K. F. Steinsholt, Mathias Iversen, Mikael R. Mathiassen, and Jon Yngve Hardeberg

1. **What kind of meeting the report is about**

Sprint retrospective meeting part B

2. **Which cases were discussed, and which decision was made?**

- What do we have left; (3 min)
  - A few paragraphs.
  - Reading through the whole report a few times.
- Something called “project critique” (2 min)
  - Discuss that the project is only a small part of a bigger problem and that we were not supposed to solve it all.
- Sustainability (5 min)
  - Our project is opposed to making 3D models, printing on paper and printing on textiles.
- We did not do unit testing in the code (5 min)
  - Should we write about it, or will it only show our weaknesses more?
  - It is more important in backend programming.
  - We have mentioned it in future work, in a situation where there is more backend code, and explained why we don’t have it now. That is good.
- Not normal to have a lot of subsections in future work (5 min)
  - Could be a more overall text, with reflections.
  - Very valuable here.
  - Can put a general overview at the beginning of the chapter, where we write about weaknesses, possibilities, etc.
- Placement of images (10 min)
  - Not use the [H] for here.
  - Rather have it “misplaced” than force it to be where you want.
  - Could use [t] as the top of the page or [b] as the bottom of the page. A little better.
  - [p]: page with only figures.

    - It is a reason that we refer to figures with numbers. The user will find the right figure.
    - Leave it up to LaTeX!
- Product backlog in “Requirements” (3 min)
    - Might enumerate it for better references when it is referred to.
- Daily scrum minutes (5 min)
    - Should not deliver.
    - Maybe as a zip-file appendix.
    - Can wait for requests from examiners.
- Code snippet/commands (5 min)
    - What to call it? Figure?
    - Let it float?
    - Make the caption before
    - Scripts or code
- Delivery (5 min)
    - Can deliver several times before the deadline.
    - Will deliver Monday evening and then have a final look on Tuesday.
- Trial presentation (5 min)
    - We must make a poster and deliver 31st.
    - 31st at 8.00 - 9.00 fits best for everyone.
    - 15-30 minutes of presentation, gives us around 40 minutes to discuss.
- Demo/presentation for the archaeologists (5 min)
    - Could invite them to Gjøvik.
    - 28th can work, as it does not need to be as long as the main presentation.
- Conclusion was written together today (2 min)
    - Davit and Jon Yngve will read through it soon.
- Submitting code (5 min)
    - We have a few lines in the report.

APPENDIX J

USABILITY TEST TEMPLATE

Usability Test Nr. X

Date:
Test host(s):
Test subject(s):

Purpose of the test:

Tasks:

Title:	(Brief title describing the task)
Task description:	(Write the task how it should be said out loud to the test subject for them to do)

Comments:

- Comments

Scores (1-10):

Finding the feature	
Using the feature	
Number of clicks	
Design	

Title:	(Brief title describing the task)
Task description:	(Write the task how it should be said out loud to the test subject for them to do)

Comments:

- Comments

Scores (1-10):

Finding the feature	
Using the feature	
Number of clicks	
Design	

APPENDIX K

USABILITY TEST 1 AND 2

Usability Test Nr. 1

Date: 05.03.2024
Test host(s): Mathias Iversen, Mikael Røv Mathiassen, Oda K. F. Steinsholt
Test subject(s): Margrethe Havgar

Purpose of the test:
To go over the functionality that we have introduced to the program with the archaeologists and get their feedback on how it is implemented.

Downloading was not a problem.

warning when downloading

A command line is open when the program is running.

Tasks:

Title:	Save new project
Task description:	Open a new project on the home screen, load in a couple of pictures, and then save the project to a folder wherever you want. Before finishing, move or rotate one or more of the images without saving.

Comments:

- Confused when opening the images (on top of each other)
- Curser went to “Home” but corrected to “File”

Scores (1-10):

Finding the feature	8
Using the feature	easy
Number of clicks	Expected amount
Design	

Title:	Open saved project
Task description:	From the main page, open the project you just saved.

Comments:

- Quickly found the feature

Scores (1-10):

Finding the feature	10
Using the feature	Easy
Number of clicks	Expected
Design	

Title:	Export Image
Task description:	Export the canvas as an image a couple of times using different resolution scales.

Comments:

- Did not understand what scaling the exported image was
- Did not get the option where to save it and name it.
- Did assume can go over 100% (thought it was the cropping)
- “Export as image” name of the button
- Multiple resolutions, low, medium, high and an advanced feature

Scores (1-10):

Finding the feature	10
Using the feature	5
Number of clicks	Expected, but could be different. See comment
Design	

Title:	Right-click screenshot
Task description:	Right-click on the canvas and explore the different ways to export an image of the canvas.

Comments:

- Liked the built-in feature of canvas capturing.
- Discussing a feature of a darker background on the canvas.

Scores (1-10):

Finding the feature	7, used to right-clicking
Using the feature	Easy
Number of clicks	
Design	yes

Title:	Score window
Task description:	Open the score window and move it around before closing it.

Comments:

- Did not see the point of the score window as it did not include helpful information.
- A hesitation when looking for the button, wondering where it was.
- The term “score” does not tell anything. “Similarity metrics” is better

Scores (1-10):

Finding the feature	9
Using the feature	7
Number of clicks	
Design	Did not match the overall look of the programme

Title:	Move and rotate multiple fragments.
Task description:	(If you have less than three fragments on the canvas, load in some more). Select all fragments and move or rotate them. Unselect 1 fragment and move or rotate the remaining fragments. Try to explore what you could do.

Comments:

- Expected to use “shift” to select more images. Could have both keys. Like in PowerPoint.
- Moving the canvas using the right key is fine.

Scores (1-10):

Finding the feature	
Using the feature	8
Number of clicks	
Design	

Title:	Delete multiple fragments
Task description:	Delete all the fragments at the same time.

Comments:

- Used backshift first to delete.
- Delete when right-clicking on a selected image.

Scores (1-10):

Finding the feature	9
Using the feature	great
Number of clicks	
Design	

Usability Test Nr. 2

Date: 02.04.2024
Test host(s): Oda K. F. Steinsholt, Mikael Røv Mathiassen, Mathias Iversen
Test subject(s): Margrethe Havgar

Purpose of the test:

Test the program as it stands now, focusing on the features implemented or changed since the last user test.

Schedule:

1. Mathias will go over the tasks with the user whilst Mikael writes notes per task.
2. When finished, Oda will go over each task and ask questions and comments about each task.
3. Then Mikael will go over the final questions that we have

Before the test:

1. Ask if we can record screen and audio. We will use our machine for screen recording (which records screen clicks and keystrokes).
2. Explain the aloud protocol: "During the test, we want you to express your thoughts as you perform the tasks verbally. That is, describe what you see and what you expect to see. We want to hear what you are thinking, not what you think others might think about the program."
3. Example of think-aloud: "I'm going to draw a stick figure in Paint. First, I'll look for a way to create a circle for the head... I click on this icon... Then I look for a pencil icon but can't find it. So now I'm a bit confused..."

4. Have a test round with think aloud protocol where they test a program we haven't made (for example, PowerPoint: Create a slide with a title, some text, and an image). The goal is to become comfortable speaking aloud and not actually testing the program.
5. "If you have any questions, feel free to ask them, but we most likely won't answer them until after the test." Oda will take notes.
6. "There are 10 tasks. The expected test time is 30 minutes. We can take breaks if needed."
7. Start recording!

Tips for test hosts:

- Remember not to give them hints or comments, answer questions, or provide feedback on their work. Instead, write it down and provide feedback after the test. If they get frustrated, it's important that we let them be frustrated to see what they do.
- Remember, silence is golden! Don't break the silence with empty talk. Instead, try to maintain the think-aloud protocol by asking questions like: "What are you thinking now?" "What are you trying to do now?" "What are you looking for?" "What did you just do?"

Tasks:

Do not say the tile out loud, only the task descriptions.

Title:	1. Make a project
Task description:	Make a project with at least three images. Move them around a bit so they do not overlap.

Comments:

- Making a new project was not a struggle, and finding where to import fragments was not difficult since it was where it had been the previous time.
- How the zoom function works is a bit strange on the trackpad. Not what is expected.

Scores (1-10) (After the whole test):

Finding the feature	Easy, 10
Using the feature	10
Number of clicks	(calculated by watching the screen recording)
Design	

Title:	2. Find my workspace
Task description:	Try to lose your workspace in the canvas by zooming out as far as you can and moving the canvas. Then, try to find your way back to the images.

Comments:

- Got back to the work area when the button was found.

Scores (1-10) (After the whole test):

Finding the feature	8-10, overthought
Using the feature	10
Number of clicks	(calculated by watching the screen recording)
Design	

Title:	3. Open the filter window
Task description:	Find the window where you can add filters to an image.

Comments:

- Tried to find the filter window under the file and tool dropdown menu.
- No obvious way to open the filter window, but found the right click eventually.
- Made sense it would be right click after not finding a specific button.
- When selecting another fragment with a left-click, expected the filter to target that fragment.

Scores (1-10):

Finding the feature	7, overthinking
Using the feature	
Number of clicks	
Design	

Title:	4. Add a filter to all the images
Task description:	Make all the images have a shade of pink.

Comments:

- Tried to select all and right-click to change the hue for all images simultaneously but found out that did not work.
- Was wondering what “value” means.
- The numbers help compare the settings of different fragments.
- The value, luminance, and mask sliders being the same greyscale could make it difficult to differentiate after a whole day of work.
-

Scores (1-10) (After the whole test):

Finding the feature	8
Using the feature	easy
Number of clicks	(calculated by watching the screen recording)
Design	See comments

Title:	5. Reset all on one image
Task description:	Remove all filters on an image, but keep the filters on the others.

Comments:

- Was wondering if “reset all” would reset the filter on all the fragments instead of just the one.
- Reset all makes sense, but we could change the button text to “Reset Filters”. Same with enable disable to say “Filters”.

Scores (1-10) (After the whole test):

Finding the feature	10
Using the feature	10
Number of clicks	(calculated by watching the screen recording)
Design	Some changes could be

Title:	6. Reset the filter
Task description:	Remove some more of the filters on the other images.

Comments:

- The reset all button works wonders.
- Resetting a single filter with a button with text instead of a symbol was great.

Scores (1-10) (After the whole test):

Finding the feature	10
Using the feature	10
Number of clicks	(calculated by watching the screen recording)
Design	

Title:	7. Undo
Task description:	You regret the reset of the filters. Try to get the filters back in a quick way.

Comments:

- Used the undo feature without a problem.

Scores (1-10) (After the whole test):

Finding the feature	10
Using the feature	10
Number of clicks	(calculated by watching the screen recording)

Design	Arrows would make sense.

Title:	8. Redo
Task description:	You regret the undo you did. Try to get the reset back in a quick way.

Comments:

- Used the redo feature without a problem.
- Commented that she also found the keyboard shortcuts to work as well.

Scores (1-10) (After the whole test):

Finding the feature	10
Using the feature	10
Number of clicks	(calculated by watching the screen recording)
Design	

Title:	9. Explore the filters on an image.
Task description:	Explore the filter options on an image. Change the filters like you would if you were working on a project.

Comments:

- Tried to use contrast and invert simultaneously and was pleased it worked as she thought it would.
- Saturation is sensitive when increasing.
- Mask threshold seems to eat up the fragment like acid.
- Reset individual filters.
- Was wondering about different hue settings.
    - Change the saturation for what the program interprets as a specific hue only.

Scores (1-10) (After the whole test):

Finding the feature	10
Using the feature	10
Number of clicks	(calculated by watching the screen recording)
Design	

Title:	10. Disable filters
Task description:	Compare the filtered images with the originals without removing or changing the filters.

Comments:

- Dragged the fragment to the other copy of the fragment and compared them.
- Then used the disable/enable filter button.
- Could also have a keyboard shortcut. In the event that the filter window is closed.

Scores (1-10) (After the whole test):

Finding the feature	10
Using the feature	10
Number of clicks	(calculated by watching the screen recording)
Design	Well placed, centred is good.

Questions after the test:

- Either luminance or value in the filter window must be removed. Which one do you prefer? Luminance should be removed, and Value should be renamed to Brightness.
- Undo is on Ctrl + z, but there are differing opinions on how to redo Ctrl + y or Ctrl + Shift + z. Which one do you prefer to use? Was used to Ctrl + y, but keeping both is excellent.

- Right now, the undo and redo buttons are just text. Would you like it better if they were arrows? Yes.

Other comments:

- Contrast slider right underneath the saturation slider.
- Change the contrast slider to be grey to hue to black.
- Threshold not being gradient, and contrast being gradient
- The name “Mask threshold” was hard to understand at first. Examples: edge reduction,
- Export image should have a standard resolution such as low, medium, high, and with an advanced function.
    - They should be based on regular standards such as 1080p, 4K, 8K, or different DPI.

APPENDIX L

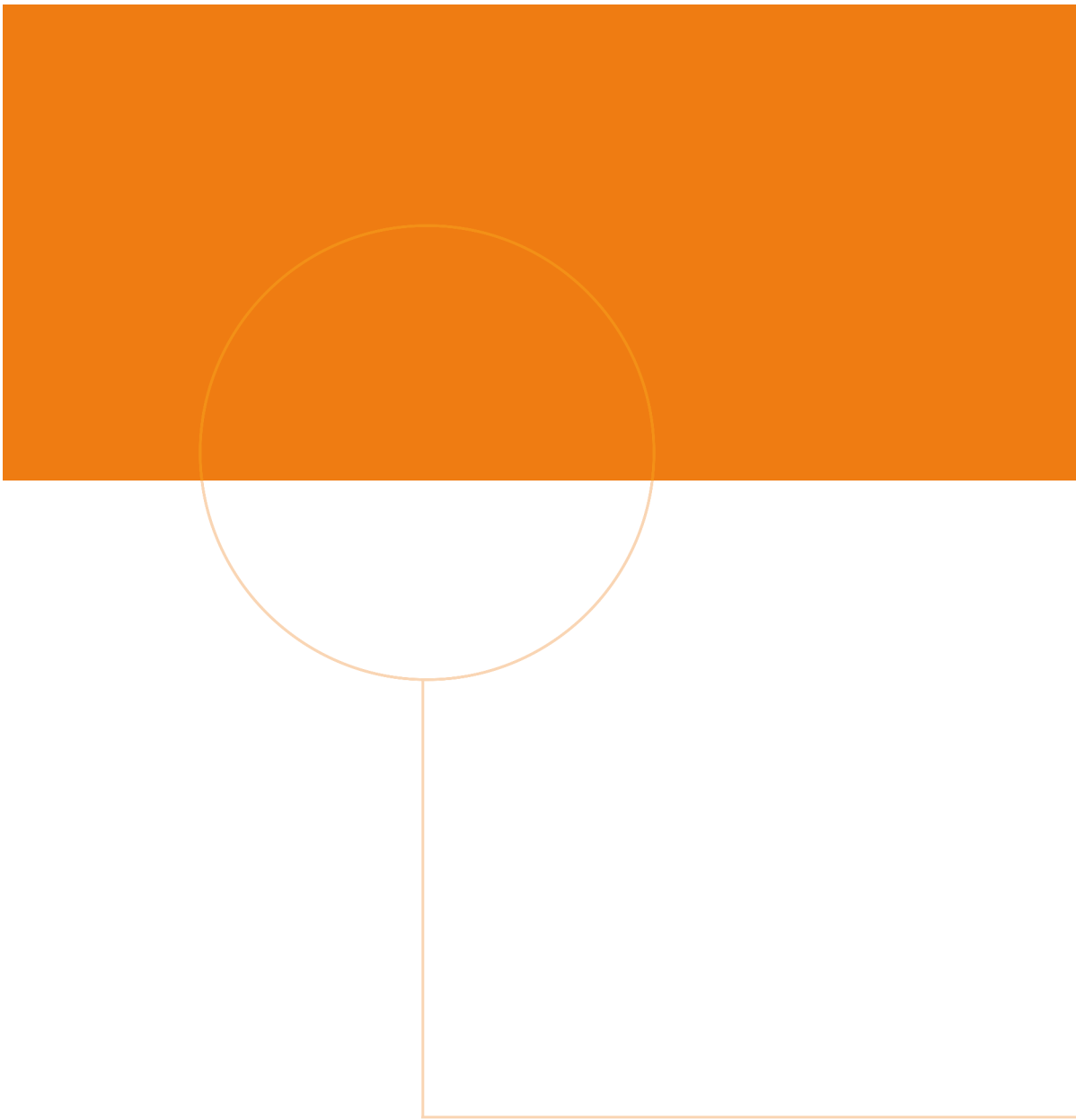
PUZZLE USER TESTS

Task:	Start a new project and load the puzzle pieces onto the canvas.	Separate each piece from each other randomly so they are not stacked on top of each other. Do not try to solve the puzzle yet.	Find 4-5 pieces that fit together and assemble them.	Use the Similarity metrics tool on a selected piece. Do the same as the last task again with the tool.	How did you find using the similarity metrics compared to just your eyes?	Additional comments on the test process?
User 1	It was not a problem on both tests. Did not like having to Ctrl+A to select all images. Would like to select the folder and load in all images in it, but that is specific to this use case	First click to select, and then click again to move is annoying.	Seaside: Went for the pieces with the ocean first and the cliffside Flower: Significantly more difficult than seaside, but found the upper left red flower.	Seaside: Took time to find the info on the most similar, but eventually found some pieces that fit Flower: Did not find any useful information.	Seaside: Worked on some, but wasn't 100% accurate. Flower: Did not help with finding pieces compared to not using it. Harder to do it on the flower than seaside.	
User 2	Loading in images was not a problem.	Took time to get used to moving an image by first selecting it but noted it meant that every movement had to be on purpose, which is a positive thing. Was made more difficult by using the trackpad instead of a mouse	Seaside: Finding pieces was ok. The upper right corner had similar branches. Flower: Found 4 pieces of the top left flower right away and put them together.	Seaside: Took time to scroll down all the time. Not much useful info on a lot of pieces. Flower: Took a long time to find any similar pieces using the tool. Could not find a complete 2x2.	Seaside: Preferred just using eyes. Flower: Very hard to find matching pieces with the tool.	Frustrating to only rely on the tool to find pieces.

User 3	Loading in was not a problem, but could not find the folder at first (not test subjects fault).	Found left clicking to select, but tried to use right click to drag the canvas around.	Seaside: Went fine. Found the sea and cliffside fast. Flower: Like everyone else, found the top left red flower relatively fast.	Seaside: Scrolling problem again. When looking at the numbers. Expected the higher number to be better. Flower: Eventually found some that matched, but noted it was not accurate, since the pieces were so similar.	Seaside: Using just eyes was more reliable to look at smaller details. Flower: The tool alone took to long to find similar pieces. Had to individually click multiple to find the one occurence of matching pieces	Would be easier if there was more explanation on how to use the program.
User 4	Finds file easy. Finds "load images" button. loads everyone quick. Nice that they are not on top of each other, but they go out of screen when loading in many. Zoom out very hard, and unintuitive.	He tried to move the image without selecting it first and found fast out how it worked. It takes time to master it dough, as he struggled with this several times throughout the test. If you miss the click, it makes it harder	colours: difficult to solve by eyes, as no figures. uses the colours. tries to rotate the images and likes that you can do that. flower: you can rely more on contours here.	colours: He clicked at the image before clicking the similarity button. Scrolled to the bottom first. He looked at the different metrics, and not just the final score. He tried to change min/max values and clicked enter, but it did not work. He found the arrow after a while. flowers: already guesses that similarity will be	colours: helpful for some cases. Nice to verify what he already saw before. He used the table a lot to get more than one result. wants full screen oportunity. flowers: harder, but gave some valuable results if looking in the right places. still a lot of misleading results. It will be very useful if we have images from	similarity metrics only give one result, even though the top 3-4 could be relevant. could be useful with colour codes on the table based on a similarity value. Easy to find feature. average hard to use. It's annoying to change the size of the window each time. number of clicks was decent, axcept for the rezising of the window

				similarity will be less helpful before starting. chooses a colourful piece, as it might give better results. he looks more at the histogram this round. manages to make one match!	images from several puzzles on screen.	window. confusing when selecting several images, to see that we have results for both. used to just scroll to the bottom
User 5	did some exploring of the nav bar then loaded in images	after trying, found that needed to double-click to be able to move an image. made more than once moving the canvas by mistake by not double clicking on the images. did not know how to zoom, but found out by accident	color: using color tones to see who match flower: easily found the big red flower and put them together	color: hard to know what this is for. found the metrics names not helpful. tries to click the images in the table. finds out that resizing is possible. flower: found that the numbers where smaller and that means that they are similar. wanted to click on image to mark it in the canvas. was confused when deselecting when the similarity window was open. was hard to find the images the tool suggested. used mostly the closest sugesstion.	colour: It is a sometimes useful tool; maybe the tool would be more useful if the images were more different. flower: found it easier to use the eyes.	easy to find the window hard to understand the table many mouse clicks to move images around fine design.

User 6	finds out of zooming fast	it took a short time to understand that the image needed to be clicked before being able to move them	colours: works quickly with matching pieces, a little bit irritating that it is hard to match them 100% correctly flowers: seems complicated at first, and actually wants to use the metrics. uses types of flowers/leafs to match.	colours: right-clicks at first, to try to find the window. complicated when first seeing the metrics. reads through it but does not understand so much. thinks that the first image in the table is the most similar. they are now in the same order as they were loaded in, and is therefore "cheating" on putting them together. flowers: does not make any progress with the metrics, and gets confused and frustrated. thinks that an image needs to be rotates.	colours: did not have any idea what the similarity metrics did (did not scroll to the bottom). but plays a lot around with it and then understood more. think it is misleading that the metrics say who is "equal", as that does not mean they match at the edge. likes to do things manually in other situations, so don't feel like using the metrics if not needed to. flowers: gets confused about what the metric does as they did not help anything.	kinda easy to find, but thought that if was going to be on right-click. confusing that the window is at the same place when changing the image. hard to use as she did not understand what it did. perfect amount of clicks. especially when changing the image. The design is nice but a long distance to the bottom, with important info.



NTNU

Norwegian University of
Science and Technology