

Steven Magnus Emil Berglund Francis and  
Frithjof Thorvik

## Game of Git

Developing and testing an educational game for  
learning Git version control system

Master's thesis in Computer Science  
Supervisor: George Adrian Stoica  
May 2023



Steven Magnus Emil Berglund Francis and Frithjof  
Thorvik

## **Game of Git**

Developing and testing an educational game for  
learning Git version control system

Master's thesis in Computer Science  
Supervisor: George Adrian Stoica  
May 2023

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





# Abstract

In this thesis we continue developing a game-based prototype aimed at facilitating learning of the Git version control system, that was developed during the specialization project 'Game of Git' by Francis & Thorvik (2023). This prototype is also evaluated and compared to a traditional learning method, to assess the effectiveness of the game for learning Git. The prototype focused on linking learning objectives within the gameplay elements, combined with familiar gameplay in a fantasy context. The goal of the thesis is to answer the question:

- *How effective is game-based learning for learning Git version control system compared to a traditional method?*

The development process followed alpha and beta testing to fix major usability issues and bugs, with the goal of making the prototype ready for gamma testing - which aimed to compare the learning gain of learning through the game or through a traditional approach.

The gamma testing involved 17 participants who were split into an experimental group and a control group, with a pre-test/post-test experiment design. The participants' proficiency in Git was assessed in the pre-test and the post-test, and statistical analysis with ANCOVA was performed to control for the variance in pre-test results. Additionally, the participants answered questions regarding their learning experience and offered feedback on the game.

The analysis found there to be no significant difference in learning gain between the experimental group and control group in the post-test. However, a significant difference in motivation was found in favour of the game group. We also found that the game was successful in linking learning objectives with the gameplay elements, and participants reported that the familiar gameplay aided their focus on learning. However, the game was found to be lacking in storytelling, role-play and a clear overarching goal, resulting in reduced immersion for players. Throughout this process it was also found that following the *playtest* method by Davis et al. (2005) was lacking during beta testing for discovering issues with the educational content of the game.

# Sammen drag

I denne masteroppgaven fortsetter vi å utvikle en spillbasert prototype rettet mot læring av Git versjonskontrollsystem, som ble utviklet under spesialiseringssprosjektet 'Game of Git' av Francis & Thorvik (2023). Denne prototypen blir også evaluert og sammenlignet med en tradisjonell læringsmetode, for å vurdere effektiviteten til spillet for å lære Git. Prototypen fokuserte på å integrere læringsmål inn i spillelementene, kombinert med bruk av velkjent spillmekanikk i en fantasi-kontekst. Oppgaven forsøker å svare på spørsmålet:

- *Hvor effektiv er spillbasert læring for å lære Git versjonskontrollsystem sammenlignet med en tradisjonell metode?*

Utviklingsprosessen fulgte alfa- og beta-testing for å fikse store brukervennlighetsproblemer, med mål om å gjøre prototypen klar for gammatesting – som hadde som mål å sammenligne læringsgevinsten ved læring gjennom spillet eller gjennom en tradisjonell metode.

Gammatestingen involverte 17 deltakere som ble delt inn i en eksperimentgruppe og en kontrollgruppe, og fulgte et pretest–posttest-design. Deltakernes ferdigheter i Git ble vurdert i både pre-testen og post-testen, og statistisk analyse med ANCOVA ble utført for å kontrollere for variansen i pre-testresultatene. I tillegg svarte deltakerne på spørsmål angående deres læringserfaring og ga tilbakemelding på spillet.

Analysen fant at det ikke var noen signifikant forskjell i læringsgevinst mellom gruppene i post-testen. Det ble imidlertid funnet en signifikant forskjell i motivasjon til fordel for spillgruppen. Vi fant også at spillet var vellykket med å integrere læringsmål inn i spillelementene, og deltakerne rapporterte at de velkjente spillmekanikkene hjalp dem med å fokusere på læringen. Det ble også funnet at spillet manglet historiefortelling, rollespill og et klart overordnet mål, noe som resulterte i redusert innlevelse for spillerne. Gjennom denne prosessen ble det også funnet at å følge *playtest*-metoden av Davis et al. (2005) under betatesting var noe mangelfullt for å oppdage problemer med det pedagogiske innholdet i spillet.

# Preface

This paper presents the work for our master's thesis at the Department of Computer and Information Science, Norwegian University of Science and Technology. Our research was conducted from August to December 2022 during our specialization project and from January to June 2023 for our thesis. The primary objective of our project was to design and develop an educational game that facilitates the learning of core concepts in Git version control system. Our goal was to create an interactive and engaging learning experience for students that would enhance their understanding of Git. Additionally, we aimed to evaluate the effectiveness of the game as a teaching tool and determine its potential as a means of learning Git compared to more traditional approaches. We extend our sincere gratitude to the students who participated in our experiment and evaluated the game, and to our supervisor, George Adrian Stoica, for making this project possible.

# Contents

<b>Abstract</b>	<b>ii</b>
<b>Sammendrag</b>	<b>iii</b>
<b>Preface</b>	<b>iv</b>
<b>Table of Contents</b>	<b>viii</b>
<b>List of Tables</b>	<b>ix</b>
<b>List of Figures</b>	<b>x</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Problem Description . . . . .	2
1.2 Personal Motivation . . . . .	2
1.3 Project Scope . . . . .	2
1.4 Research Goal . . . . .	3
1.5 Thesis Structure . . . . .	3
<b>2 Methodology</b>	<b>6</b>
2.1 Research Design . . . . .	7
2.1.1 Literature Study . . . . .	7
2.1.2 Design and Creation . . . . .	7
2.1.3 Survey . . . . .	8
2.1.4 Experiment . . . . .	9
2.1.5 Data Collection Methods . . . . .	9



---

2.2	Statistical analysis . . . . .	11
2.2.1	ANCOVA . . . . .	12
2.2.2	T-test . . . . .	13
2.2.3	Mann-Whitney U test . . . . .	13
2.3	Goal Question Metric . . . . .	14
2.3.1	Research Goals and Questions . . . . .	15
2.3.2	GQM Tables . . . . .	16
<b>I</b>	<b>Prototype</b>	<b>17</b>
<b>3</b>	<b>Research from the Specialization Project</b>	<b>18</b>
3.1	Git . . . . .	19
3.2	Educational games . . . . .	19
3.3	Game design . . . . .	20
3.4	Game development . . . . .	20
3.5	Existing solutions . . . . .	21
<b>4</b>	<b>Implementation</b>	<b>22</b>
4.1	Planning & Research . . . . .	23
4.2	Development . . . . .	24
4.2.1	Methodology . . . . .	24
<b>5</b>	<b>The Game Artefact</b>	<b>26</b>
5.1	Basis in Research . . . . .	27
5.2	Game Concept . . . . .	27
5.3	Features . . . . .	27
5.3.1	Supported Git commands . . . . .	28
5.3.2	Git flows . . . . .	29
5.3.3	Technical components . . . . .	29
5.4	Gameplay . . . . .	31
5.4.1	Fetch screen . . . . .	31
5.4.2	Work screen . . . . .	32

---

5.4.3	Summary screen . . . . .	33
5.4.4	Store screen . . . . .	34
5.4.5	Tutorials . . . . .	35
<b>II</b>	<b>Prestudy</b>	<b>36</b>
<b>6</b>	<b>Testing a serious game</b>	<b>37</b>
6.1	Introduction . . . . .	38
6.2	Evaluation . . . . .	38
6.2.1	Alpha & beta testing . . . . .	38
6.2.2	Gamma testing . . . . .	42
6.3	Standardized Questionnaires . . . . .	44
<b>III</b>	<b>Finalizing The Game</b>	<b>47</b>
<b>7</b>	<b>Testing</b>	<b>48</b>
7.1	Alpha & beta testing . . . . .	49
7.1.1	Alpha testing . . . . .	49
7.1.2	Beta testing . . . . .	51
<b>8</b>	<b>Improvements</b>	<b>52</b>
8.1	Alpha test results . . . . .	53
8.1.1	Internal testing . . . . .	53
8.1.2	External testing . . . . .	58
8.2	Beta test results . . . . .	61
<b>IV</b>	<b>Experiment, Results, and Discussion</b>	<b>62</b>
<b>9</b>	<b>Experiment</b>	<b>63</b>
9.1	Structure . . . . .	64
9.2	Setup . . . . .	64
9.2.1	Distributing Groups . . . . .	65

---

9.2.2	Creating the Git competence test . . . . .	66
9.2.3	Creating the traditional learning document . . . . .	66
9.2.4	Standardized and ad hoc questions . . . . .	67
9.3	Execution . . . . .	67
9.3.1	Recruitment . . . . .	68
9.3.2	Data analysis . . . . .	68
9.3.3	Interviews . . . . .	69
<b>10</b>	<b>Results</b>	<b>70</b>
10.1	Demographic . . . . .	71
10.2	Experiment Results . . . . .	79
10.2.1	Assumptions . . . . .	79
10.2.2	Results . . . . .	81
10.3	Learning Experience . . . . .	82
10.4	Gameplay Feedback . . . . .	86
10.4.1	EGameFlow Questions . . . . .	86
10.4.2	Ad hoc Questions . . . . .	89
10.5	Interviews . . . . .	91
<b>11</b>	<b>Discussion</b>	<b>93</b>
11.1	Testing effectiveness of game-based learning . . . . .	94
11.1.1	Alpha and beta testing was valuable . . . . .	94
11.1.2	Playtest method was lacking . . . . .	94
11.1.3	Gamma testing worked well . . . . .	95
11.1.4	Experiment length was too long . . . . .	96
11.2	Differences in Learning Outcome . . . . .	97
11.2.1	Inconclusive increase in understanding . . . . .	97
11.3	Differences in Learning Experience . . . . .	98
11.3.1	Increased time spent learning . . . . .	98
11.3.2	Increased learning motivation and enjoyment . . . . .	99
11.4	Gameplay and learning integration . . . . .	100
11.4.1	High level of EGameFlow knowledge improvement . . . . .	100

---

11.4.2	Medium to high level of EGameFlow concentration . . .	100
11.4.3	Low to medium level of immersion . . . . .	101
11.5	Limitations of the study . . . . .	103
11.5.1	Samples . . . . .	103
11.5.2	Testing method . . . . .	104
11.5.3	Digital survey . . . . .	105
<b>V</b>	<b>Conclusion and Further Work</b>	<b>106</b>
<b>12</b>	<b>Conclusion</b>	<b>107</b>
12.1	RQ1 What is the process for testing effectiveness of game-based learning? . . . . .	108
12.2	RQ2 Does game-based learning of Git provide a better understanding of basic concepts compared to traditional learning? 109	
12.3	RQ3: Does game-based learning provide a better learning experience compared to traditional methods for learning Git? 109	
12.4	RQ4 Does the game successfully integrate Git elements into its design? . . . . .	110
12.5	Overall . . . . .	110
<b>13</b>	<b>Further Work</b>	<b>112</b>
	<b>Bibliography</b>	<b>114</b>
	<b>Appendix</b>	<b>117</b>
A1	NSD Assessment . . . . .	118
A2	Alpha testing - Questionnaire and results . . . . .	121
A3	Beta testing - Questionnaire and results . . . . .	128
A4	Information Letter . . . . .	136
A5	Experiment - Test versions . . . . .	139
A6	Experiment - Traditional learning document . . . . .	144

# List of Tables

2.1	Research goals . . . . .	16
2.2	Research metrics . . . . .	16
2.3	Goal Question Metric (GQM) table . . . . .	16
10.1	<b>Number of students per year from given study program:</b> MTDT ( <i>Computer Science</i> ), MTKOM ( <i>Communication Technology</i> ), MTTK ( <i>Cybernetics and Robotics</i> ), MTELSYS ( <i>Electronic Systemdesign</i> ), MTING ( <i>Engineering &amp; ICT</i> ), MSIT ( <i>Informatics</i> ), Other (Folkehøyskole and Graphic Design). . .	72
10.2	<b>EGameFlow Knowledge Improvement:</b> showing how many participants answered for each option. The mean is calculated using the scoring shown at the bottom of the table. . . . .	86
10.3	<b>EGameFlow Immersion:</b> showing how many participants answered for each option. The mean is calculated using the scoring shown at the bottom of the table. . . . .	87
10.4	<b>EGameFlow Concentration:</b> showing how many participants answered for each option. The mean is calculated using the scoring shown at the bottom of the table. . . . .	88
10.5	Linking learning objectives with game objectives. . . . .	89
10.6	Familiar Gameplay (Yes/No). . . . .	89
10.7	Familiar Gameplay and learning. . . . .	90
10.8	Bug reports. . . . .	90

# List of Figures

5.1	<b>Fetch screen:</b> displays available branches with respective information on missing ingredients, earnings, etc. . . . .	31
5.2	<b>Work screen:</b> interface for completing orders for each day. .	32
5.3	<b>Summary screen:</b> provides a summary of the player's performance for the day. . . . .	33
5.4	<b>Store screen:</b> the player can buy upgrades, git commands, and new ingredients. . . . .	34
5.5	<b>Tutorial screen:</b> guides the player throughout the game. . .	35
5.6	<b>Help screen:</b> allows players to revisit previous tutorials. . .	35
8.1	<b>Merge screen:</b> merges all branches into the main branch. .	54
8.2	<b>Clone screen:</b> provides repository URL that can be cloned.	55
8.3	<b>New difficulties:</b> increase earnings but decrease daytime, and represent a new repository. . . . .	56
8.4	<b>Improved help screen:</b> includes re-playable tutorials, and information about available Git commands and Git concepts.	57
8.5	<b>Updated terminal:</b> includes color-coded responses, and improved response information. . . . .	59
8.6	<b>Updated info-box:</b> removed unused commit history, and display current branch information. . . . .	60
10.1	<b>Bar chart on participants' field of studies:</b> MTDT ( <i>Computer Science</i> ), MTKOM ( <i>Comunication Technology</i> ), MTTK ( <i>Cybernetics and Robotics</i> ), MTELSYS ( <i>Electronic System-design</i> ), MTING ( <i>Engineering &amp; ICT</i> ), FHS ( <i>Folkehøyskole</i> ), Graphic Design, and MSIT ( <i>Informatics</i> ). . . . .	71

---

10.2	<b>Bar chart on participants' git experience.</b> A Git competence of 0 in the figure represents those answering 'no' to having any experience using Git. . . . .	73
10.3	Bar chart on participants' gender in each method of learning.	74
10.4	Bar chart on participants' code editor experience. . . . .	75
10.5	Bar chart on participants' preferred learning method. (Multiple choice) . . . . .	76
10.6	Bar chart on participants' interest in learning and becoming better at Git. (0 = Strongly Disagree, 4 = Strongly Agree) . . . . .	77
10.7	Bar chart on participants' gaming experience. (0 = Strongly Disagree, 4 = Strongly Agree) . . . . .	78
10.8	Plot of the dependent variable (post-test) against the covariate (pre-test) showing linear relationship. . . . .	80
10.9	ANCOVA results. . . . .	81
10.10	<b>Bar chart on participants' experience learning through each method</b> (0 = Strongly Disagree, 4 = Strongly Agree). Learning Enjoyment= <i>'I enjoyed the learning process for learning Git'</i> , Learning Motivation= <i>'I felt motivated while learning Git'</i> , Learning Perception= <i>'I felt like I learned something'</i> . . . . .	83
10.11	Bar chart on participants' average time (in minutes) spent learning through each method. . . . .	84
10.12	Bar chart on participants' reason for ending learning process. (multiple choice) . . . . .	85

# Chapter 1

## Introduction

The popularity and utility of Git in the software development industry cannot be overstated. However, with the increasing relevance of coding in various fields, there is a growing need for individuals to have both coding skills and an understanding of version control using Git. Unfortunately, learning Git can be challenging for beginners, making it difficult to get started and intimidating to learn. To overcome this challenge, a game-based learning approach for Git can be an effective and engaging way for beginners to learn Git more easily and accessibly.

This thesis aims to continue development of an educational game, which started development during the specialization project 'Game of Git' by Francis & Thorvik (2023), and teaches core Git concepts. Continuing, the thesis evaluates the effectiveness of the game-based learning approach for teaching Git to students. By creating a game-based learning experience, this research will explore the potential benefits of using game-based learning for technical and challenging topics like Git, with the goal of contributing to the existing body of knowledge on the effectiveness of game-based learning for technical topics and improving the teaching and learning of Git.



---

## 1.1 Problem Description

The goal of this project is to design and develop an educational game that aims at facilitating learning of core concepts in Git version control system. The student(s) will work to design, implement and evaluate an application (could be web based) that can be used by students learning version control.

The candidate(s) should contribute with novel and original ways to deal with the various concepts within Git. The student(s) will have to work with the main stakeholders for eliciting requirements.

The project involves a study of research and relevant literature on similar software, design and implementation of a functional prototype and evaluation of the developed prototype at different levels, including user testing.

## 1.2 Personal Motivation

Our personal motivation for taking on this project stems from a few different areas. We both have a shared passion for gaming, which we have enjoyed for many years. In addition to playing games, we have personally experienced the challenges of learning Git and were interested in developing a solution that would facilitate the learning process for other students. We were also intrigued by the potential of game-based learning as an effective teaching tool and a relatively new area of research. Leveraging our expertise as web developers, we were enthusiastic about the opportunity to create an engaging and informative educational game.

## 1.3 Project Scope

The project started in August 2022 with the aim of creating an educational game that effectively teaches core concepts in Git version control. Given the limited development timeline, it was crucial to establish effective ideation, design, and development strategies to generate a feasible concept for the prototype as early as possible. The first phase of the project was carried out between August and December in 2022, which is detailed in 'Game of Git' by Francis & Thorvik (2023) and serves as the foundation for this thesis. The primary focus of this thesis is to evaluate the effectiveness of the developed prototype in terms of its game quality and its ability to teach essential Git concepts.

---

## 1.4 Research Goal

While there has been a considerable amount of research conducted on the effectiveness of game-based learning and methods for evaluating the quality of such games, it is important to note that this area of research is still relatively new and constantly evolving. In this thesis, we aim to address the research goal

1. *How effective is game-based learning for learning Git version control system compared to a traditional method?*

We will provide a detailed description of the research goal and questions in chapter 2.

## 1.5 Thesis Structure

This section gives an overview of the thesis structure in order to gain a quick understanding of each of the parts and chapters contained within this paper.

### Part I: Prototype

This part of the thesis is dedicated to presenting the prototype that was developed in the specialization project created between August 2022 and December 2022, and it serves as a summary of the work presented in the specialization project 'Game of Git' by Francis & Thorvik (2023).

- **Chapter 3 - Research from the Specialization Project**  
Presents a summary of the research performed in the previous specialization project.
- **Chapter 4 - Implementation**  
Presents how the prototype was implemented and which methodologies were used during development.
- **Chapter 5 - The Game Artefact**  
Presents the game concept and the state it was in before it was finalized and improved.

### Part II: Prestudy

This part denotes the start of the work conducted after the end of the specialization project, meaning that this part and the consequent parts was

---

conducted in the period of January 2023 to June 2023 and is unique to this master thesis.

The prestudy was conducted to gather the essential methodologies and information required to establish the foundation of our research.

- **Chapter 6 - Testing a serious game**  
Presents research pertaining how to test the effectiveness of a game based learning artefact

### **Part III: Finalizing The Game**

This part of the thesis presents how the game was finalized and includes how the game was tested and improved as a result.

- **Chapter 7 - Testing**  
Presents how the game was tested using the alpha & beta testing method.
- **Chapter 8 - Improvements**  
Presents which improvements was made in the different stages of testing.

### **Part IV: Experiment, Results, and Discussion**

In this part of the thesis, the main experiment is presented, including the methodology employed, the results obtained, and the ensuing discussion.

- **Chapter 9 - Experiment**  
Presents the main experiment and how it was implemented.
- **Chapter 10 - Results**  
Presents the findings and results of the experiment.
- **Chapter 11 - Discussion**  
Examines and interprets the results.

### **Part V: Conclusion and Further Work**

This section serves as the concluding chapter of the thesis, summarizing the entirety of the research and proposing potential areas for future work to further advance this research.

- 
- **Chapter 12 - Conclusion**  
Summarizes the thesis and results.
  - **Chapter 13 - Further Work**  
Suggests potential future research that can build upon the results presented.

## Chapter 2

# Methodology

To conduct our research, we adopted a variety of methodologies to guide our study. First, we used the Goal Question Metric (GQM) method to identify high-level research goals and questions, which helped us to define the scope of our study. In order to find answers to these questions we used different research strategies, along with different data collection methods. The strategies employed include: literature and game studies to find relevant research already conducted within the field, design and creation for producing a game artefact to be used for our research, surveys to gain a better understanding of the areas for improvement in our prototype, an experiment to investigate the effect of introducing the game artefact for learning Git version control system, and interviews to gain qualitative data after the experiment. The data collection methods used are questionnaires for obtaining quantitative data and interviews for qualitative data. By using these strategies and data collection methods, we were able to approach our research questions with a structured and informed approach, ultimately contributing to the overall quality of our study.

---

## 2.1 Research Design

This section outlines the methodologies that were used to conduct the research, which includes literature study, design and creation, surveys, and experiments. The surveys and experiments collected data with approval from NSD - Norsk senter for forskningsdata (see Appendix A1). The following sections will provide more details on the methodologies used.

### 2.1.1 Literature Study

A literature study is a research method used to collect and analyze existing literature, such as academic papers, books, and reports, on a particular topic. It is an important method for gaining a general understanding and overview of the already existing research within a field. In our thesis, we will be performing a literature study to identify and analyze relevant research on assessing the effectiveness of game-based learning. By conducting a literature study, we can gain insights into the strengths and weaknesses of existing research, identify gaps in knowledge, and build upon existing theories and findings. This will help us to develop a strong foundation for our research and ensure that we are building on existing knowledge in the field. The literature study was carried out using Google Scholar, as advised by Briony J. Oates (2022), with keywords related to "Game Based Learning", "Digital Game Based Learning", "Serious Games", and "Learning Assessment AND Game Based Learning". The strategy for reading articles was influenced by the method outlined in Briony J. Oates (2022), which involves first reviewing the abstract, followed by the introduction and conclusion to determine relevance before deciding to read the entire article.

### 2.1.2 Design and Creation

The goal of design and creation is to create an artefact that can contribute to knowledge in some way (Briony J. Oates 2022). The goal of the artefact is therefore not just to demonstrate technical skills, but also to demonstrate some academic qualities by creating new knowledge. According to Briony J. Oates (2022), the design and creation process is iterative and begins with identifying and defining a problem, which can be prompted by reviewing other solutions or conducting a literature review. Following this is a creative process where an idea is developed. This idea is then implemented and evaluated, and conclusions are drawn from the process to assess what was learned. The produced artefact can contribute to knowledge in various ways. It can serve as the primary focus of the research, such as demonstrating how

---

a theory can be applied in practice. Alternatively, it can be used as a means to an end, such as analyzing the impact of a mobile application in a learning environment. The research can also focus on the development process itself, such as examining the suitability of an agile method for creating an embedded system. In our thesis, the created artefact will serve as a means to an end by helping us answer the question of how effective a game-based learning application is compared to traditional learning methods for learning Git version control system.

### 2.1.3 Survey

The survey strategy is designed to systematically collect data from a large population, allowing for patterns to be identified and generalized to the broader population. Surveys typically involve questionnaires, but may also rely on observations or interviews. Before selecting participants for a survey, a sampling frame must be defined, which identifies the population that fits the target group of the survey. Various sampling techniques can then be used to select participants from within this frame. Random sampling involves selecting participants at random, while self-selection sampling involves advertising for participants and allowing people to decide whether to participate. Convenience sampling involves selecting participants who are readily available and easily accessed. It is important to note that self-selection and convenience sampling may not accurately represent the opinions of the broader population. Participants who actively choose to take part in a survey may hold stronger opinions on the subject matter than the general public. Similarly, if a convenient sample is used, the selected subjects may not be representative of the broader population (Briony J. Oates 2022).

The survey strategy is advantageous because it can generate a large amount of data quickly and at low cost. However, it may not provide the depth of information that other research strategies can offer. For our thesis, we will use surveys to evaluate our game prototype. The surveys will provide insights into usability issues, bugs, and the overall experience for the players. By using surveys, we can gain a deeper understanding of the game's strengths and weaknesses, which can help to improve the final product.

---

### 2.1.4 Experiment

The experiment strategy is used to analyze cause and effect relationships between certain factors. Typically, the experiment begins with a hypothesis that proposes that factor A causes outcome B. An experiment is then designed to exclude all other factors that might affect the results, leaving only the observation of whether factor A causes outcome B. It is crucial to control all variables in the experiment, which can be achieved through the use of a control group. The variable that is being measured is varied between the two groups, while all other variables are held constant (Briony J. Oates 2022).

Experiments usually involve a pre-test and post-test to measure changes. The validity of an experiment can be evaluated based on internal and external factors. Internal validity refers to the extent to which the experiment accurately measures the impact of the independent variable on the dependent variable, meaning the cause-and-effect relationship between them. This includes minimizing the differences between the control and experimental groups, as any observed effects could be due to these differences. Additionally, you should ensure consistency of the timing of the pre-test and post-test, as unseen events occurring between the tests can interfere with the result. Continuing you should be addressing the effects of maturation, as participants can change between tests, for example due to fatigue or boredom from completing the first test. External validity, on the other hand, refers to the extent to which the findings of the study can be generalized to other populations beyond the sample used in the study. To achieve external validity, it is important to have a sufficiently large and representative sample that accurately reflects the population of interest (Briony J. Oates 2022).

### 2.1.5 Data Collection Methods

To conduct the surveys effectively, different methods of data collection were considered, based on which would fit our research objectives the best. Our primary data collection method was through questionnaires, but we also incorporated qualitative interviews to gather additional insights beyond what the questionnaires could provide. Questionnaires allowed us to gather structured and quantitative data from a larger sample size, while qualitative interviews allowed us to explore specific issues in-depth and gain a deeper understanding of the participants' experiences. By using a combination of data collection methods, we aimed to gather a comprehensive and nuanced understanding of our game prototype's usability and user experience.



---

## Questionnaire

Questionnaires are a popular data collection method in research studies and are often used to gather quantitative data from a large number of participants. They typically consist of a set of standardized questions that are designed to measure specific variables of interest. One of the main benefits of using questionnaires is that they allow for efficient and systematic data collection, as well as providing a structured approach to data analysis. Additionally, questionnaires can be easily replicated and administered to different populations, allowing for comparisons and generalizations to be made across groups. However, one disadvantage is that participants may misunderstand or misinterpret the questions, leading to inaccurate or incomplete responses. Additionally, participants may not always be truthful in their responses, which can bias the results. Another potential issue is acquiescence bias, where participants tend to agree with the posed question, regardless of their actual beliefs or experiences (Briony J. Oates 2022). Questionnaires were used in multiple phases of the project, with two main objectives.

1. Measure the prototype's usability, how enjoyable it is, and finding bugs and issues
2. Measure and compare the learning outcome between learning Git through the game and a traditional method

## Interviews

Conducting interviews is a data collection method that is useful for gaining detailed information through open-ended questions. Briony J. Oates (2022) distinguished between three different types of interviews that can be conducted:

- **Structured:** Structured interviews involve identical questions for each participant. The questions are usually read out to participants, and the answers are often noted using pre-coded answers. This type of interviews resemble questionnaires, but the interviewer takes notes of the participants' responses instead of them filling out a form.
- **Semi-structured:** In semi-structured interviews, there is usually a predefined list of themes and questions to discuss. However, the format is more free-formed, allowing the interviewer to change the order of questions or even add new follow up questions throughout the interview.
- **Unstructured:** Unstructured interviews give the researcher even less control, as they involve introducing topics to talk about, however the

---

participants are encouraged to talk freely about their ideas and opinions without too much restriction and interruption from the interviewer.

Both semi-structured and unstructured interviews are suited for allowing the interviewees to express their thoughts and feelings about certain topics. However, they are not suited for generalizing to the whole population (Briony J. Oates 2022).

Semi-structured interviews were used in this thesis to answer questions we had after conducting the main experiment. This allowed us to gain qualitative feedback on issues not answered through the questionnaire.

## 2.2 Statistical analysis

Statistical analysis was conducted as a part of this thesis to assess the result of the experiment. Specifically, analysis of co-variance (ANCOVA) was used to see if there was a statistically significant difference in post-test scores based on the learning method employed. Independent sample t-test was used to see if there was a statistically significant difference in time spent learning between the two groups. Additionally, Mann-Whitney U test was used to see if there was any significant difference in learning motivation, learning enjoyment or learning perception between the two groups. The following sections will provide some details regarding these tests.

---

### 2.2.1 ANCOVA

ANCOVA, analysis of co-variance, is generally recognized as the preferred method for pre-test and post-test statistical analysis. When using ANCOVA in pre-test and post-test analysis, the pre-test score is treated as a co-variate to control for the initial differences between the control group and experimental group. As a result, ANCOVA reduces the influence of pre-test scores on post-test scores, enabling us to measure the effect of the experimental method while controlling for individual variations in pre-test scores. In other words, it answers the question: What is the effect of the experimental method given that each participant scored the same on the pre-test? (Bonate 2000) When operating with ANCOVA, Rutherford (2011) mentions some assumption that underlie the method:

1. **independent groups:** each group contains a random sample from the population, and people should be randomly assigned to groups,
2. **normality:** the test results in each group is normally distributed, which can be tested with Kolmogorov-Smirnov test or Shapiro-Wilk test,
3. **independent observation:** the test results in each group is independent of each other,
4. **homogeneity of variance:** the variance of errors in the test results for each group are equivalent/homogeneous, which can be tested with Levene's test,
5. **linearity:** for each group there is a linear relationship between the co-variate and the dependent variable, which can be tested by visual inspection when plotting the dependent variable against the co-variate,
6. **independent co-variate:** the co-variate is independent of the groups; good design practice is therefore to measure the co-variate (pre-test scores) before executing the experiment,
7. **homogeneity of variance:** the co-variate and independent variable do not interact, meaning there is a homogeneity of regression slopes. This can be assessed by visual inspection of regression slopes (Rutherford 2011, p. 242-259).

According to Rutherford (2011), ANCOVA is regarded as being robust to violations of the normality assumption, especially if there are equal sample size in the two groups and if the sample size is greater than 12. This robustness is a matter of degree, where deviations from normality has less effect on the F-test when the sample size is larger (Rutherford 2011, p. 242-259).

---

## 2.2.2 T-test

The t-test is a parametric test used to compare differences between two groups. Often two different version of the t-test is considered: the independent sample t-test is used to compare the differences in means between the two samples when they are not dependent on each other, whilst the paired sample t-test is used when the observation is collected in pairs, meaning that the samples are dependent (Gerald 2018). In this thesis, the independent sample t-test will be used to measure the difference in time spent learning between an experimental group and a control group.

There are some assumptions that need to be taken into account before executing an independent sample t-test. First is the assumption of normality and the assumption of independent observation; meaning that the samples must come from a normally distributed population, where the samples in the different groups do not interact. Additionally, you must take into account sample size, where the t-test is only valid for sample sizes smaller than 30 (Gerald 2018).

## 2.2.3 Mann-Whitney U test

The Mann-Whitney U test is an example of an non-parametric test used to measure the difference between two groups. Being a non-parametric test, the Mann-Whitney U test does not require the dependent variable to follow a know distribution, like the normal distribution. The Mann-Whitney U test is often regarded as the non-parametric equal to the t-test, where the Mann-Whitney U test is applied to ordinal data and the t-test to interval data (MacFarland & Yates 2016). Nachar (2008), recognizes three assumptions that underlie the Mann-Whitney U test:

1. **independent groups:** the groups should contain a random sample of the population, where people are randomly assigned to the groups,
2. **independent observation:** the measured data should be independent from each other; meaning each observation is an unique participant,
3. **ordinal or continuous:** the measured data is ordinal or continuous; ordinal meaning categorical data with unknown distance between the categories (ex. Likert scale).

Nachar (2008) recommends a sample size of between 5-20 participants.

---

## 2.3 Goal Question Metric

Goal Question Metric (GQM) is a structured approach to define, measure, and improve the effectiveness of a software development process, project, or product. It provides a framework to define and evaluate goals, questions, and metrics that are relevant to a software development project (Basili et al. 1994).

As outlined in Basili et al. (1994), the GQM approach consists of three main steps:

1. **Define the Goals:** The first step is to define the goals for the software development project. These goals should be specific, measurable, and relevant to the project.
2. **Define the Questions:** Once the goals are defined, the next step is to define the questions that will help to evaluate the project's progress towards achieving these goals. These questions should be specific, measurable, and answerable.
3. **Define the Metrics:** The final step is to define the metrics that will be used to measure the progress towards achieving the goals and answering the questions. These metrics should be objective, reliable, and relevant to the questions.

---

### 2.3.1 Research Goals and Questions

In order to properly evaluate a serious game and its effectiveness for teaching Git compared to traditional methods, it is important that the game meet specific quality standards in its usability and playability, to properly evaluate the effectiveness of a game-based approach for learning Git. With these constraints in mind, one research goal was defined.

#### Research Goal

*How effective is game-based learning for learning Git version control system compared to a traditional method?*

This was the main research goal of this thesis, and was furtherly decomposed into four research questions:

#### RQ1

*What is the process for testing effectiveness of game-based learning?*

This question aims to answer and explore what processes that are needed to test the effectiveness of learning through a game. Is it necessary for the game to meet specific standards before undergoing testing? What methodologies should we use to design these assessments, and should the quality of the tests be monitored?

#### RQ2

*Does game-based learning of Git provide a better understanding of basic concepts compared to traditional learning?*

This question aims to answer whether a game-based approach is able to provide better results in terms of learning basic Git concepts compared to a traditional approach.

#### RQ3

*Does game-based learning provide a better learning experience compared to traditional methods for learning Git?*

This question aims to answer if the game-based approach can improve the learning experience of the player compared to a traditional approach in areas such as enjoyment, motivation, and perceived learning.

#### RQ4

*Does the game successfully integrate Git elements into its design?*

This question aims to answer if the game is able to integrate Git concepts successfully into the gameplay. Does Git fit within the context of the game? Is the player able to immerse themselves and feel like they are enjoying a game instead of just learning?

---

### 2.3.2 GQM Tables

The GQM was used to define the research question and which metrics to use for each question. The results of the process can be seen in Table 2.2. The research goal is summarized in Table 2.1, and the metrics used are summarized in Table 2.2.

ID	Research Goal
RG1	How effective is game-based learning for learning Git compared to traditional methods?

Table 2.1: Research goals

ID	Research Metric
M1	Literature study
M2	Questionnaire
M3	Interview

Table 2.2: Research metrics

ID	Research Question	Goal	Metrics
RQ1	What is the process for testing effectiveness of game-based learning?	RG1	M1, M3
RQ2	Does game-based learning of Git provide a better understanding of basic concepts compared to traditional learning?	RG1	M2
RQ3	Does game-based learning provide a better learning experience compared to traditional methods?	RG1	M2, M3
RQ4	Does the game successfully integrate Git elements into its design?	RG1	M2, M3

Table 2.3: Goal Question Metric (GQM) table

# Part I

## Prototype

In this part, we will present an overview of the prototype developed during the specialization project, which took place from August to December 2022. As the prototype is the foundation of the master thesis, we will provide a summary of the research that led to the creation of the prototype, as well as an overview of the prototype itself. This information is essential as it provides the necessary context for the master thesis. Since the content provided in this part is only a summary of ‘Game of git’ by Francis & Thorvik (2023), we do advise reading Francis & Thorvik (2023) for a more detailed description.

The scope of the master thesis is to enhance the prototype’s features, finalize its functionalities, and evaluate its effectiveness in facilitating the learning of Git; this will be covered in Part II, Part III, and Part IV.



## Chapter 3

# Research from the Specialization Project

The specialization project tried to:

*”Investigate which principles of game based learning and game design that makes learning engaging, and map these to creating a educational game for learning Git version control system”*  
Francis & Thorvik (2023)

A substantial part of the specialization project went into researching Git version control system, educational games, game design, game development, and existing solutions; as well as developing the prototype, which will be discussed in chapter 4 and chapter 5. This chapter will provide an overview the research that the prototype is based on.

Read the specialization project here: **Game of Git**

---

## 3.1 Git

From the specialization project we found that: *"Git is a free distributed version control system. A version control system is a system responsible for tracking changes to computer programs, documents or other collections of information, so that specific versions can be retrieved later. With a distributed system, every clone of a server acts as a backup of the data, making it easy to restore data if the server is corrupted"* (Francis & Thorvik 2023).

Furthermore, the main workflow in Git consist of three parts: *the working directory* - where you can modify files, *the staging area* - where the current state of a modified file can be temporarily stored, and the *git repository* - where your modified files are stored as a part of the Git version control system.

Continuing, core concepts of Git where identified as committing, branching, merging and pushing to the remote. Committing is the act saving your changes from the staging area to the Git repository, where the changes are stored as a series of snapshots, called commits; where each commit points to it's previous commit. Branching is a way of organizing your changes, in so called branches, so that you can work on different features of your application separately. A branch is essentially a pointer to a commit, where different branches can point to different commits, allowing you to add changes in one branch without affecting the other branches. Merging is the act of integrating the changes from different branches with each other. When you merge one branch into another, the changes from the former branch are integrated into the latter branch. This allows for changes made to the separate branches to be integrated into a unified code base. Pushing to the remote is the act of sending your local changes to a Git repository that is hosted on the internet. These features enables multiple people to collaborate and work on the same project in parallel. Lastly, to interact with Git you use Git commands, which are usually executed in the command-line of your computer (Francis & Thorvik 2023).

## 3.2 Educational games

In the specialization project an important distinction between traditional games and educational games was found to be the need to take into account pedagogical strategies. In an educational game it is therefore important to integrate the learning objectives into the game without interrupting the flow of the game. Furthermore, it was found that adequate challenge, collaboration, storytelling and providing learning in the correct context, were important attributes to consider when creating an educational game. The challenge

---

should be repeatable to allow the player to try out different strategies, and it should also be varied, adjustable and without too much time pressure. Additionally, the story-telling should be relatable, set in an fantasy context and humorous - all to encourage learning. Lastly, the context of the game should also be grounded with familiar activities already proven successful in the gaming industry (Francis & Thorvik 2023).

### **3.3 Game design**

In addition to attributes important for serious games, Francis & Thorvik (2023) also highlighted useful game mechanics found in literature about game design. When it comes to challenge, the main takeaway is that the game should provide challenge that is broken down into achievable steps, with appropriate feedback about progression towards the goal, at a varied pace that matches the skill level of the player and with a reward that is valuable to the player. The challenge should neither be too easy nor too hard to keep the player engaged. For story-telling the goal is to make players less aware of themselves, which can be achieved with role-playing and a good narrative that is balanced between familiarity and novelty. Furthermore, the game should provide a control scheme that allows the user to feel in control of their actions, where their intentions are accurately translated to in-game actions. The game should also capture the full attention of the player, without too many distracting tasks. It is also important to provide accurate feedback to allow the player to see progress towards the goals of the game, along with appropriate instructions and tutorials to teach the player how to play (Francis & Thorvik 2023).

### **3.4 Game development**

The specialization project also covered research about the development process for game creation. It was found that it is important to create a game design document that captures the vision of the game, including the concept, story, gameplay and aesthetics. From here the literature suggested brainstorming, prototyping, testing the prototype, and iterative development as a guide for creating a game. Continuing, it was found to be crucial during development to manage the transition from the game design document to the actual game development, where one should keep a separate requirement document for the game artifact and the game design (Francis & Thorvik 2023).

---

## 3.5 Existing solutions

The specialization project identified six existing solutions, which also try to teach Git with game based learning, those were: Github Minesweeper<sup>1</sup>, git-game<sup>2</sup>, githug<sup>3</sup>, Twilio<sup>4</sup>, OhMyGit!<sup>5</sup> and LearnGitBranching<sup>6</sup>. A selection of these were analyzed with the LEAGUÊ analysis instrument created by Tahir & Wang (2019), with the goal of discovering the weaknesses and strengths of the solutions. In summary, the strengths of the existing solutions were colourful visualization, intuitive and simple controls, good tutorials, multiple goals, good story telling, wide coverage of git concepts, and gradual difficulty. However, there were clear weaknesses when it came to the difficulty of challenges, a lack of creative problem solving, and too little replayability. The prototype developed in the specialization project took into account the weaknesses and strengths of these existing solutions. The focus was on providing good aesthetics, intuitive and simple controls, and comprehensive tutorials, along with implementing a gradual difficulty curve. Additionally, the prototype aimed to address the shortcomings of other solutions by offering more difficult challenges, better replayability, and a stronger emphasis on creative problem-solving (Francis & Thorvik 2023).

---

<sup>1</sup><https://profy.dev/project/github-minesweeper>

<sup>2</sup><https://github.com/git-game/git-game>

<sup>3</sup><https://github.com/Gazler/githug>

<sup>4</sup><https://www.twilio.com/quest/learn/open-source>

<sup>5</sup><https://ohmygit.org/>

<sup>6</sup><https://learngitbranching.js.org/>

## Chapter 4

# Implementation

In this chapter, we will briefly introduce the implementation of the prototype, which can be split into two phases. The initial planning & research phase and the development phase. We will summarize each of the phases, to give an insight into some of the choices that were made and methods used in each of the phases.

---

## 4.1 Planning & Research

The initial phase of the project was easily the most challenging, mainly because we had to figure out the idea for a game that could both create engagement from a game perspective, but also enable sufficient learning for the player. This required a game concept that could support both of these which proved harder than expected. Research was made on both game-based learning and existing solutions in order to help us generate an idea that could support the requirements that we had elicited. In addition, the design process was inspired by Design Thinking as taught in the course TMM4220 - Innovation by Design Thinking<sup>1</sup> at NTNU, where there is a focus on rapid prototyping with user testing and needs-finding through empathy. In accordance with this, interviews were conducted with relevant participants in order to identify their needs regarding a game based approach for learning Git version control system. Inspired by the feedback from the interviews, three visual prototypes were created in Figma<sup>2</sup> to represent three different game concepts, each covering a need found in the interviews. The concepts were developed through methods from Design Thinking such as brainstorming and Crazy 8s. The three concepts developed were:

1. A highly practical approach with story, scenarios and gameplay as close to reality as possible.
2. A highly abstract approach, that is focused on the game-play aspect, which base game mechanics loosely on core Git concepts with the goal of the player learning Git as a by-product of playing the game.
3. A mix between the two concepts above, with a combination of practical interaction with Git, set in a fictional setting with the goal of creating engagement.

After the visual prototypes of the three concepts were completed, interviews were again conducted, now to gain feedback on the developed concepts and it's corresponding visual prototype. The feedback from this round of interview was considered, and a choice was made to develop concept 3 further (Francis & Thorvik 2023).

---

<sup>1</sup><https://www.ntnu.edu/studies/courses/TMM4220#tab=omEmnet>

<sup>2</sup><https://www.figma.com/>

---

## 4.2 Development

The development phase began after establishing the final game concept. The first step was to choose suitable technologies for the project and the developers, considering efficient development due to limited time. After evaluating popular game engines like Unity, Unreal Engine, and Godot, as well as web development tools like React, we considered several factors such as performance, compatibility, cost, alignment with the game's concept and goals, and our familiarity with each technology (Francis & Thorvik 2023).

Although Unity, Unreal Engine, and Godot were powerful game engines, we lacked experience and we were unsure if they would be the best fit for our game concept. However, we had extensive experience with React and web development. Therefore, we decided to use React as our game did not require advanced graphics, physics simulation, or scripting. Using React allowed us to leverage our existing knowledge and skills to create a web application, bypassing the time and effort it would require to learn a new and complex game engine.

In the end, our decision to use React was based on our game's requirements, our team's expertise, and our goal to create an acceptable quality prototype within minimal time and cost.

### 4.2.1 Methodology

Due to our small development team of only two people, we decided not to adopt a formal development methodology such as Scrum, which typically requires a team of five to nine people (Kniberg 2015). However, we still followed the principles of agile software development, which emphasizes incremental development and continuous delivery of working software (Sommerville 2016). Specifically, we aimed to make small, incremental changes to the software and focused on having a functional prototype after each change.

As Sommerville (2016) explains, agile development relies on informal communication rather than formal meetings and written documentation. We also adopted this approach, preferring to use informal communication rather than extensive documentation. To keep track of our progress, we utilized a Kanban Board, which is a visual tool commonly used in agile development to categorize and visualize progress on tasks. The Kanban Board helped us to minimize overlapping work, categorize new necessary tasks, and track tasks in development, remaining tasks, and completed tasks. By adopting these agile principles and tools, we were able to develop the software iteratively and having a working prototype on a regular basis.

Additionally, we used code-reviews, pair programming and refactoring to

---

ensure high code quality. Code-reviews were performed before accepting any changes to the code base, both to check for quality of code and for quality of the game-play. Likewise, pair programming was often utilized, where code was co-written to check each other's work and to ensure good code quality. Furthermore, refactoring was frequently used to avoid degrading of the software structure over time, which tend to happen in incremental development (Sommerville 2016).



## Chapter 5

# The Game Artefact

In this chapter, we will provide an overview of the game artefact, including its educational features for teaching Git as well as its gameplay elements. This information reflects the state of the game as it existed at the completion of the specialization project in December 2022. However, any updates or improvements to the game will be discussed in detail later (see Part III, chapter 8).

---

## 5.1 Basis in Research

The game artefact produced in the specialization project was based on the conducted research, which is summarized in chapter 3. The game primarily focused on linking learning objectives with game objectives by infusing Git concepts into the game concepts. Secondly, the game focused on providing challenge that is neither too challenging nor too easy, whilst integrating the challenge with creative problem solving. Thirdly, the game focused on providing immersion through storytelling, which focused on intrinsic fantasy and providing familiar gameplay that is proven to do well in the industry. Lastly, the game focused on providing good feedback, with a focus on progression towards goals (Francis & Thorvik 2023).

## 5.2 Game Concept

*”Git Cooking is placed in a world where restaurants have digitized and streamlined many processes. Food and orders are created with machines controlled digitally by the restaurant’s staff. These machines are built around Git to create and provide customers’ orders. Upon entering the game, the player is a new employee in one of these restaurants and will have to learn how to use this system to help the owner grow their restaurant and business.*

*During the day, when the restaurant is open, the player must fulfill the orders placed throughout the day. This process requires the player to create the orders and register these changes in the terminal with the proper Git commands, for the machines to create the orders. After each day, the player will receive a certain amount of money depending on the amount and quality of fulfilled orders. At the end of each day, the player can purchase different upgrades to generate more revenue and unlock new features to increase productivity within the game” (Francis & Thorvik 2023).*

## 5.3 Features

The game provides various features, but this section will focus on the features related to Git and technical components meant for teaching the player how to use Git in a more practical manner.

---

### 5.3.1 Supported Git commands

This is a list of the supported Git commands in the game and their similarity to Git.

- `git add`: Within the game this command work similarly to Git by adding modified items from the working directory to the staging area. The game supports two variants `git add .` or `git add [path]`, which works as in Git by adding all modified items or the specified item.
- `git commit -m [message]`: Similarly to Git, changes in the staging area are committed to the git repository with this command. In the game this is reflected by updating the percentage completed for each order once a commit is made, to reflect the items saved in the commit.
- `git status`: This commands provides information about modified and staged files, as-well as the currently active branch (similar to Git).
- `git checkout [name]`: In the same way as in Git, this command can be used to checkout remote or local branches. However in the game, the command does not support checking out commits.
- `git branch`: As in Git, this command list up the local branches. The game also support the version `git branch -r`, which list up remote branches.
- `git restore`: This command restore modified items to match the last commit or the last staged version of the item, which is similar to the functionality in Git. The game supports two variants `git restore .` or `git restore [path]`, which works as in Git by restoring all modified items or the specified item.
- `git restore --staged`: This command works as in Git and removes items from the staging area. The game supports two variants `git restore --staged .` or `git restore --staged [path]`, which works as in Git by removing all staged items or the specified item.
- `git fetch`: Similar to Git this command fetches remote branches.
- `git push`: Similar to Git this command pushes changes in the currently active branch to the remote tracking branches. The game supports an additional version `git push origin [branch name]` to specify which branch to push.

---

### 5.3.2 Git flows

As the main target group of this game is students that has little to no experience with Git, our main focus was to introduce the most used and important Git concepts and flows to enable its players to successfully use Git after playing the game. For this reason, the game introduces the following Git flows:

- Fetching changes from a repository remote
- Adding, committing, and pushing changes to branches
- General helpful commands such as `git status`, `git branch`, and `git restore`

### 5.3.3 Technical components

The game's approach to teaching Git goes beyond just introducing Git features. In addition to general Git features, the game also provides users with other technical components that are crucial for the practical use of Git such as interacting with a terminal and navigating a directory with folders and files. By offering a comprehensive learning experience that encompasses various technical aspects, users can develop a well-rounded skill set where users not only gain an understanding of Git but also acquire the practical skills necessary to use it effectively.

#### Terminal

The game's approach to teaching Git commands involves interactive terminal use, which is a more practical method that closely aligns with how Git is used in the real world. However, the introduction of the terminal can be daunting for beginners with no prior experience. The rationale behind this approach is to assist players in overcoming this initial hurdle, thereby easing the transition from learning Git to using it in practical situations. By utilizing this approach, players will gain valuable skills that will prepare them for real-world Git usage.

#### Directory & files

The introduction of a feature that includes a directory with folders and files was based on the same rationale as introducing the terminal. Git directly tracks changes made to files, making it possible to display the current state of

---

files for users. By introducing this feature in a structured and clear manner, users can gain confidence in using Git and develop the necessary skills to work with files in the context of version control.

---

## 5.4 Gameplay

### 5.4.1 Fetch screen

The game begins on Day 1. Before each day starts, the player is required to fetch orders from the remote repository, which presents three remote branches to choose from, each providing different orders and rewards, displayed in Figure 5.1. To start the game, the player checks out a remote branch using the Git command `git checkout [branch name]` and is then taken to the work screen.

The following git commands can be used and learned on the fetch screen:

- `git fetch` (Required)
- `git checkout [branch name]` (Required)

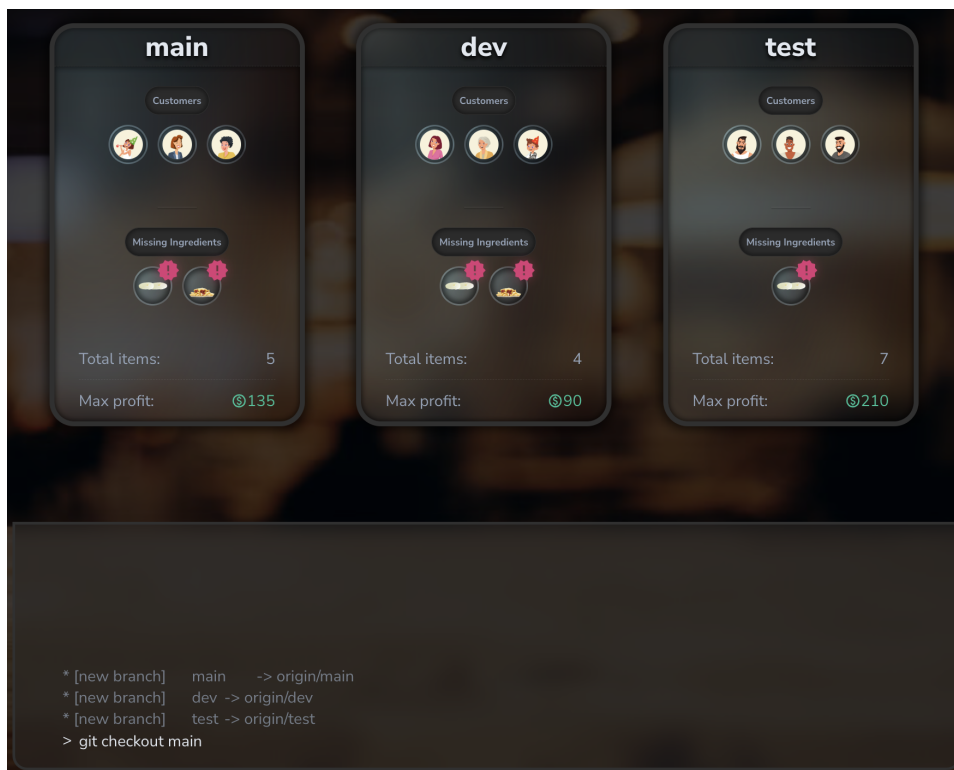


Figure 5.1: **Fetch screen:** displays available branches with respective information on missing ingredients, earnings, etc.

---

## 5.4.2 Work screen

In the work screen (see Figure 5.2), new customers arrive throughout the day, keeping the player busy with new orders to fulfill. The goal of the game is to create items that match the customer's order, add those items to the staging area, and then commit them. At the end of the day, the player is required to push their changes back to the remote repository. If the player has worked on multiple branches, they must remember to push all the branches back up to the remote before ending the day.

The following git commands can be used and learned on the work screen:

- **git add [path]** (Required)
- **git commit -m [msg]** (Required)
- **git push** (Required)
- **git status** (Optional)
- **git branch** (Optional)
- **git restore [path]** (Optional)

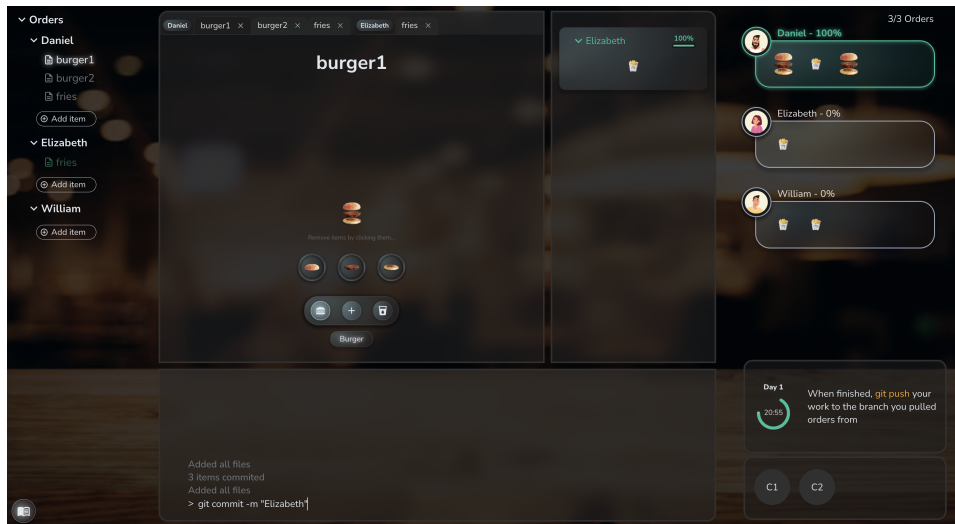


Figure 5.2: **Work screen**: interface for completing orders for each day.

---

### 5.4.3 Summary screen

After ending the day, the player will be directed to the summary screen. Here, the player will receive in-game currency based on how accurately they fulfilled the orders for each branch. Each branch card, displayed in Figure 5.3, presents a summary of the different types of earnings that were made, as well as the maximum potential earnings that could have been achieved.

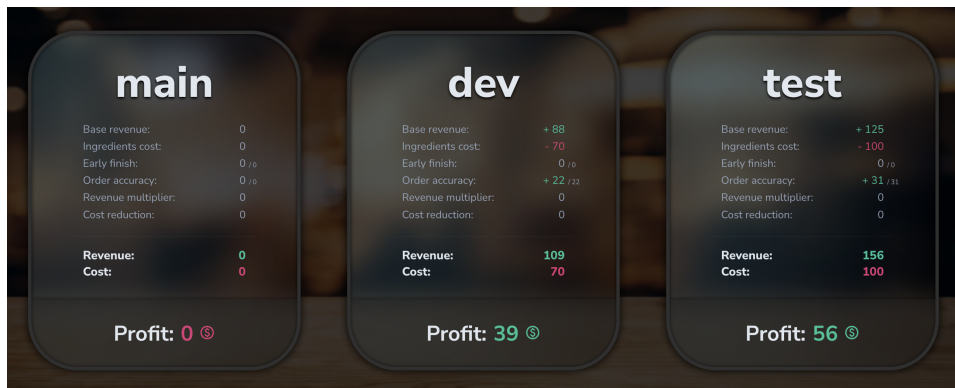


Figure 5.3: **Summary screen**: provides a summary of the player's performance for the day.



---

### 5.4.4 Store screen

After reviewing the summary screen, players can use the in-game currency earned to purchase upgrades, unlock new ingredients, and acquire new Git commands in the store screen (see Figure 5.4). After completing their shopping, players will be redirected to the fetch screen again before starting another day, and the game cycle continues. This game loop provides a repetitive and challenging learning experience, which was identified as crucial for game-based learning in Part II.

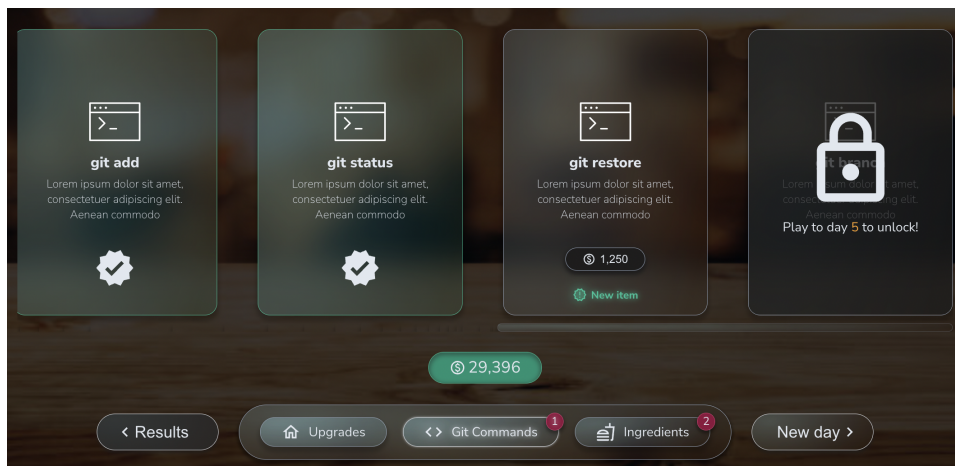


Figure 5.4: **Store screen:** the player can buy upgrades, git commands, and new ingredients.

---

## 5.4.5 Tutorials

To guide players through different stages of the game, tutorials are displayed at various points throughout gameplay to provide the necessary information for understanding the game and progressing in the game. These tutorials take the form of dialogues with the chef and may include images to illustrate concepts or tasks, as shown in Figure 5.5.

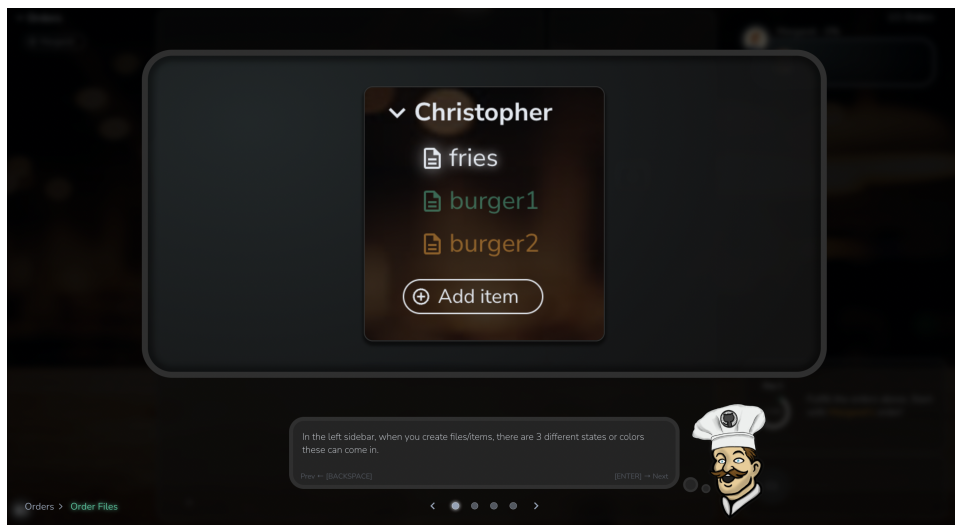


Figure 5.5: **Tutorial screen:** guides the player throughout the game.

In addition to the tutorials that appear automatically at different stages of the game, all tutorials can be accessed and replayed in the help screen, as shown in Figure 5.6. This feature enables players to review and reinforce their understanding of game concepts and mechanics, at any time.

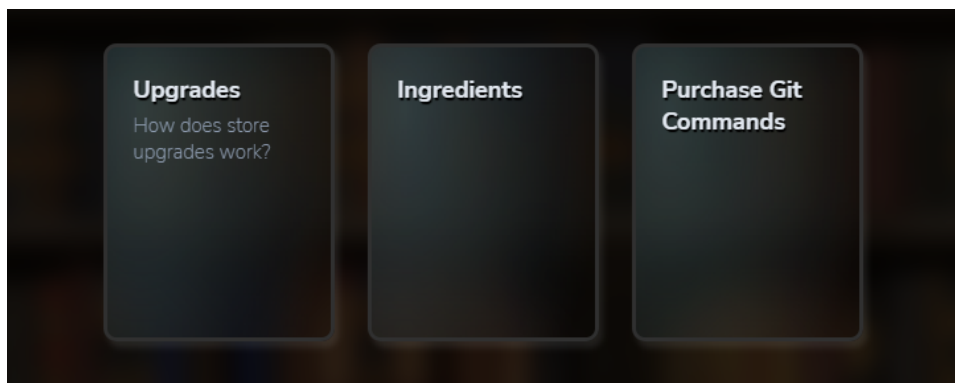


Figure 5.6: **Help screen:** allows players to revisit previous tutorials.

# Part II

## Prestudy

The work conducted for the master thesis begins here, signifying that all the material beyond this point was conducted after the conclusion of the specialization project's, and was performed in the period January 2023 to June 2023.

The prestudy serves as a comprehensive overview of the literature that has been studied in the context of this thesis, with a specific focus on research related to testing the effectiveness of serious games. As the primary focus of this thesis, the research presented in this section is particularly relevant and informs the methodology and analysis utilized in the subsequent chapters.

## Chapter 6

# Testing a serious game

The objective of this chapter is to provide the foundation for testing the effectiveness of an educational game. It aims to address the question of how to design an experiment that can accurately evaluate the learning outcomes of an educational game.

---

## 6.1 Introduction

Over the years, there has been an increase in research regarding the effectiveness of game-based learning. Despite this increase, the results are inconclusive or difficult to interpret. This is the result of a lack of overarching methodology for effective research on game-based learning, which has led to questionable validity and reliability of existing research (All et al. 2016, 2014). Another systematic literature review by Gris & Bengtson (2021) found that most studies provided incomplete information about the methods employed to evaluate educational games' impact on learning. There is a need for a scientific basis to conduct effectiveness studies on game-based learning (All et al. 2016). For that reason, the goal for this chapter is to review literature regarding best practices when conducting effectiveness studies on game-based learning.

## 6.2 Evaluation

All et al. (2016) differentiates between two forms of evaluation of educational interventions: formative evaluation and summative evaluation. The purpose of formative evaluation is to assess the intervention itself, identifying strengths and weaknesses to inform improvements. In contrast, the aim of summative evaluation is to determine the success of the intervention by evaluating its impact on learning outcomes. When evaluating a game-based learning intervention one has to do both summative and formative evaluation, in the sense that you need to test the learning outcomes, but also improve upon the game itself. de Carvalho (2012) recognizes three parts of evaluating game-based learning: alpha testing, beta testing and gamma testing. Alpha testing is related to formative evaluation, where the goal is to provide improvements to the game itself. Beta testing is also related to formative evaluation to look for improvements to the game, but should also address improvements for the educational content of the game. Gamma testing is related to summative evaluation, where the goal is to test the efficiency of the game measured against the learning outcomes.

### 6.2.1 Alpha & beta testing

An important part of the game development process is usability testing, especially when creating a serious game, as a failure in overall usability causes the player's effort to be spent mastering controls, and not on the actual learning content of the game (Olsen et al. 2011). This is also highlighted by Moreno-Ger et al. (2012) who say that *"usability issues alone can hinder the*

---

*gameplay process negatively affecting the learning experience*”, where they point out that serious games target a broader audience, among people that may not play regularly; meaning that usability issues are more likely to arise. Furthermore, Olszewski & Wolbrink (2017) claims that assessment of educational gains are often conducted after release, where release is defined as the product resulting from alpha and beta testing. It becomes apparent then that alpha and beta testing are critical phases in the serious game development process, as the goal of such testing is to improve the game, identify issues and fix bugs. After this is done, one can move on to the summative evaluation of the game.

Although alpha and beta testing are executed similarly, their primary difference lies in the user groups involved. Specifically, alpha testing involves in-house testing, while beta testing involves external testing by users from the target group of the game (Gold & Wolfe 2012). During alpha testing the tests can be executed by the development team, as well as participants from an easily accessible population, which does not have to fit the target group. The goal of such testing is to eliminate glaring usability issues, so that future testing can focus on target group specific issues (Olsen et al. 2011). Gold & Wolfe (2012) also claims that during alpha testing it might even be useful to test on friends of the authors, though this has to be avoided in beta testing to reduce bias. Apart from the difference in test subjects, beta testing in serious games also differs from alpha testing in its focus on testing learning outcomes. The goal of beta testing is therefore not to simply find defects, but also to verify that the application meets the purpose of providing learning within a certain topic (Gold & Wolfe 2012).

The goal of both alpha and beta testing is to perform tests to inform improvements to the game. de Carvalho (2012) explains that the main goal of testing is to improve the game at the user interface level, improve gameplay and improve game dynamics. To improve the usability of the game, Yáñez-Gómez et al. (2017) suggests three possible approaches: applying ”off the shelf” evaluation methods (which are applied to productivity software), updating existing evaluation methods to overcome differences between a game and productivity software, or disregard traditional evaluation methods and develop a new evaluation technique for games. However, there is no consensus on the right approach to choose. It’s worth noting that many conventional usability evaluation methods may not be directly applicable to games since they are typically designed with detailed step-by-step tasks that are not necessarily executable within a gaming context. Still, because of the similarity of productivity software and games, which includes menus, status interfaces, and input/output devices, standard ”off the shelf” methods are still applicable to games (Davis et al. 2005). However, Yáñez-Gómez et al. (2017) suggests replacing step-by-step task with in-game win/lose goals to better fit the gaming context. Similarly, Davis et al. (2005) proposes that rather than

---

evaluating a task sequence in the game, a more effective approach would be to test a specific scenario within the game and assess whether the user plays the game as intended by the developer. Additionally on the difference between productivity software and games, both Moreno-Ger et al. (2012) and Yáñez-Gómez et al. (2017) recognize the significance of distinguishing between in-game frustration and at-game frustration. This differentiation is especially important because games are distinct from productivity software in that they may intentionally include frustrating gameplay elements to enhance engagement, whereas productivity software should strive to minimize user frustration altogether. Consequently, in serious games, it is essential to distinguish between errors arising from usability issues versus errors stemming from the natural trial and error approach to playing a video game.

Regardless of which approach one chooses, the common methods for testing usability in serious games include focus groups, Think Aloud Protocol testing, standardized questionnaires, ad hoc questionnaires, interviews, direct observation and recorded session (Olszewski & Wolbrink 2017, Yáñez-Gómez et al. 2017, Moreno-Ger et al. 2012). According to the systematic review by Yáñez-Gómez et al. (2017), System Usability Scale (SUS) is the most frequently used standardized questionnaire. SUS is a simple ten item Likert scale, where the result is score between 0-100 indicating the usability of the system (Brooke 1996). Despite the statistical validity of such standardized questionnaires, ad hoc questionnaires are more prevalent in the literature (Yáñez-Gómez et al. 2017). One possible reason for this is that relying solely on standardized questionnaires may not yield sufficient feedback on specific aspects of the game being tested. So to address this issue, researchers often supplement with personalized questions that cover the specific aspects of the game that require feedback (Yáñez-Gómez et al. 2017). Continuing, Yáñez-Gómez et al. (2017) found that the most common sets of users in the literature seems to be around 10-20 users. Additionally, as noted by Yáñez-Gómez et al. (2017), while interruptions may be acceptable in traditional usability testing, they should be avoided in games-based usability testing to prevent unnecessary distractions for the participants. Instead it is more common to provide interviews and questionnaires after the play session has ended.

When it comes to how the tests are carried out there doesn't seem to be a common methodology. Some carry out observational studies as seen in Moreno-Ger et al. (2012) and Olszewski & Wolbrink (2017), where the tests are carried out with an observer watching the test subject play. Moreno-Ger et al. (2012) suggests recording sessions to discover the nuanced user interactions, which in games are often subtle, nonverbal and unpredictable. Continuing, Moreno-Ger et al. (2012) suggest having more than one evaluator to analyze each play session, and carrying out the play session with a script indicating which tasks the tester should perform. However, the review

---

by Yáñez-Gómez et al. (2017) suggest that a task-driven test is not among the majority in the literature. An alternative approach to observational studies are survey based methods, such as the *playtest* method by Davis et al. (2005). The *playtest* method combines traditional surveys with a controlled lab environment to gain quantitative feedback on user perceptions. In the *playtest* method the tester is instructed to play as they would at home. Once the play session is complete, participants are asked to provide targeted feedback on their experience. This includes sharing their thoughts on what aspects of the game they found enjoyable, their opinions on the graphics, controls, story, and other relevant features. The goal of the *playtest* method is to assess the player's initial experience and gather valuable information about their perception of the game. This is accomplished through a combination of standardized questions, game-specific questions, and open-ended questions, administered through a digitized survey. In the *playtest* method, the play sessions are not recorded and the moderators generally do not interact with the test subjects, because of this Davis et al. (2005) argues that in order to obtain valuable information about user perceptions, a larger sample size of users (typically 25-35 people) is required. Continuing, Davis et al. (2005) argues that while a small population of users can be useful for discovering usability errors, generalizing from this group may not provide sufficient insight into improving gameplay.

There is both positives and negatives to observational user tests and the survey based *playtest* method. As Davis et al. (2005) argues, the main benefit of the *playtest* method is it's ability to gain knowledge of user perceptions. However, when it comes to identifying pure usability issues, both Davis et al. (2005) and Moreno-Ger et al. (2012) agree that observational methods are superior. There is also an economical difference between the two methods, where Davis et al. (2005) argues that the *playtest* method require less time and cost compared to observational usability tests. This can be a major benefit, as according to Gold & Wolfe (2012), the standard economical rule for testing to halt the process once the expected costs of continued testing outweigh the expected benefits.

Lastly, regarding when to execute these tests, Moreno-Ger et al. (2012) recommends that the prototype should closely resemble the final product before undergoing user testing, as an incomplete prototype may not accurately reflect the usability of the finished product. This is because an incomplete prototype may not accurately capture the essential features and functionality of the final product, leading to inaccurate or unreliable user feedback. Therefore, it is important to ensure that the prototype is as comprehensive and polished as possible before initiating user testing to obtain reliable and informative user feedback.



---

## 6.2.2 Gamma testing

The objective of gamma testing is to conduct a summative evaluation, which involves assessing the effectiveness of a game as a learning tool measured against specific learning goals. de Carvalho (2012) proposes a protocol for conducting this evaluation, which consists of the following steps: (1) identification of participants, (2) a pre-test to assess knowledge and motivation for learning, (3) gameplay sessions, and (4) a post-test to evaluate the acquired knowledge and satisfaction with the game. The subsequent section will elaborate on each of these steps and look into general guidelines for assessing the effectiveness of the game.

The goal of a pre-test is to assess the current level of the test subjects (de Carvalho 2012). According to All et al. (2016), a pre-test is considered essential in the literature for three main reasons. Firstly, it helps to control for any initial differences between the experimental and control groups before they play the game. Secondly, it provides a baseline for measuring relative learning gains, which is necessary for comparing the post-test results. Thirdly, it enables control for dropout characteristics, which tend to be more common among participants who perform poorly in the pre-test. Conducting a pre-test is also important because differences in learning outcomes could be attributed to individual or group differences at the beginning of the intervention (All et al. 2014). de Carvalho (2012) recommends that the pre-test should consist of both a knowledge test and a motivation test. The knowledge test should be closely aligned with the expected learning outcomes, while the motivation test should evaluate motivation and competence related to the relevant topic, as well as general motivation for games.

The post-test serves the dual purpose of evaluating the increase in knowledge, using a knowledge test similar to the pre-test, and obtaining feedback on the participants' perception of the game. According to de Carvalho (2012), the post-test should however include a different set of questions compared to the pre-test to prevent a practice effect. However, as noted by All et al. (2016), researchers should be cautious of introducing differences in the type and difficulty of questions between the two tests. Instead, a better option may be to maintain similarity between the two tests while altering the questions' content. Then, to control for practice effects, All et al. (2014) suggests running parallel versions of the tests, with half of the participants receiving version A in the pre-test and version B in the post-test, and vice versa for the other half. Regarding assessing the participants' perception of the game, de Carvalho (2012) recommends querying about enjoyment, user-experience, and perceived competence. However, as noted by All et al. (2014), the reliability of student opinion on motivation has been found to be inconsistent with direct measures. Still, All et al. (2014) agrees that assessing motivation is an important aspect of evaluating the effectiveness of

---

game-based learning.

When conducting an effectiveness study, it is important to be aware of certain confounding elements that can affect the outcome. These factors can make it difficult to distinguish whether the observed results are due to the intervention or the confounding variables. According to All et al. (2014), there are three possible confounding effects to consider. The first is the addition of extra elements to the game, such as an introduction, debriefing, or supplementary reading material, which makes isolating the effect of the game impossible. The second is the presence of an instructor, where guidance from an instructor can lead to overestimation of the intervention's effect. The third is the practice effect, as mentioned earlier, which can make it challenging to determine whether any observed improvements are due to the intervention or simply due to the participants' familiarity with the testing format.

According to All et al. (2016), having a control group is generally advisable when conducting an effectiveness study. However, it is important to ensure that the conditions between the experimental group and the control group are as similar as possible. All et al. (2016) recommends that the control group should engage in another educational activity, as comparing with a control group with no educational activity will not provide an accurate assessment of the game's effectiveness. Furthermore, the two groups should have similar conditions in terms of the time exposed to the intervention, the learning content, the instructor conditions, received guidance, difficulty level of the content, interaction with other people, day of the week, environment, briefing, and reward for participation. The goal is for the game to be the only variable between the two groups. Regarding the instructor conditions, All et al. (2016) suggests that it would be ideal to have an absence of an instructor to properly isolate the game, although this may not always be possible. Providing guidance can also lead to problems in comparability, as the users playing the game might need more help than the users in the control group. However, the lack of guidance could also be problematic in certain contexts such as a classroom, as guidance is a natural part of such types of environments, and removing it would be unnatural. If the game contains a training session, according to All et al. (2016), this should also be provided to the control group. However it is recommended to exclude substantive information regarding the game completely.

To ensure that the results of the study are accurate and reliable, All et al. (2014) provides some additional guidelines for executing the gamma testing. Firstly, they suggest that a detailed description of the procedure should be provided to allow for replication and falsification of the results. Additionally, a follow-up study is recommended to assess the long-term effects of the educational intervention. Random assignment of participants to groups is

---

ideal, as it ensures that the groups are similar in terms of age, gender, motivation, and other relevant factors. However, All et al. (2016) notes that randomization may not be feasible as it requires a large sample size, not always achievable in this type of research. Therefore randomization at the classroom level may be the only viable option in certain contexts, although a limitation of this type of randomization is that it could result in a biased sample. All et al. (2016) recommends a minimum sample size of 20 participants per condition, with a minimum of 30 participants per condition for more robust statistical analysis. ANCOVA with pre-test scores as a co-variate is recommended for non-randomized designs, as it reduces error variance and adjusts for differences in pre-test scores between groups. The paired-samples t-test can also be used to compare gain scores from pre-test to post-test (All et al. 2014).

## 6.3 Standardized Questionnaires

There exist multiple standardized questionnaires that provide feedback regarding game enjoyment, gameplay and game immersion. These questionnaire's provide statistical validity for measuring the player's opinion and perceptions about a game. As it is recommended by Yáñez-Gómez et al. (2017) to use standardized questionnaires in the kind of research associated with this master thesis, we have identified four different questionnaires that seek to measure user perception of a game. These are: Game Experience Questionnaire, The Situational Motivation Scale, Game Immersion Questionnaire, and EGameFlow.

### Game Experience Questionnaire

The Game Experience Questionnaire, by IJsselsteijn et al. (2013), consists of three parts, each measuring different aspects of the game experience. Part one and two are in-game questions, which prompts the player about their feelings while playing the game. Part three is used post-game, and asks the player how they feel after they have stopped playing. Part one assesses the game experiences on seven components: Immersion, Flow, Competence, Positive and Negative Affect, Tension, and Challenge. Part two assess social presence, meaning the psychological and behavioural involvement with other players or NPC's. Part three assess the players emotions after having stopped playing (IJsselsteijn et al. 2013).

---

## The Situational Motivation Scale

The Situational Motivation Scale (SIMS), by Guay et al. (2000), aim to measure situational intrinsic and extrinsic motivation. Here situational motivation is the motivation an individual experiences while engaging in an activity. Intrinsic motivation refers to engaging in an activity for the activity itself. And extrinsic motivation refers to performing an activity for some goals beyond those inherent in the activity itself (Guay et al. 2000). This could be used to measure what kind of motivation the students are feeling whilst participating in the experiment, either when playing the game or when learning Git through traditional methods.

## Game Immersion Questionnaire

The Game Immersion Questionnaire, by Jennett et al. (2008), includes questions for measuring immersion in a game. The level of immersion is split into three categories: engagement, engrossment and total immersion. For each of levels of immersion, Jennett et al. (2008), identified a number of barriers that limit the immersion. Engagement is the first level of immersion. The barrier associated with this level is that of gamer preference; the player needs to invest time and effort too learn how to play the game - so if the game doesn't look initially engaging to the individual player they might not even try the game. After achieving engagement, the player can move to the next level of immersion: engrossment. In order for the player to achieve engrossment the game needs to contain game features that make the player feel emotion directly by the game, and the player should enter a state where they become less aware of their surroundings. From engrossment the player can feel "total immersion", which is when the player feels completely cut off from reality and the only thing that matters is the game. To achieve "total immersion" the game needs to contain empathy and atmosphere. The Game Immersion Questionnaire can be used to measure the players level of immersion, which is a marker of a good gaming experience (Jennett et al. 2008).

---

## EGameFlow

EGameFlow, by Fu et al. (2009), is a scale to measure enjoyment in e-learning games. The scale can help a developer understand the strengths and weaknesses of the game from the learner's point of view. The EGameFlow scale consist of 42 questions sorted in eight different dimensions:

- *"Concentration (6 items): games must provide activities that encourage the player's concentration while minimizing stress from learning overload, which may lower the player's concentration on the game."* (Fu et al. 2009)
- *"Clear Goal (4 items): tasks in the game should be clearly explained at the beginning."* (Fu et al. 2009)
- *"Feedback (5 items): feedback allows a player to determine the gap between the current stage of knowledge and the knowledge required for ultimate completion of the game's task."* (Fu et al. 2009)
- *"Challenge (6 items): the game should offer challenges that fit the player's level of skills; the difficulty of these challenges should change in accordance with the increase in the player's skill level."* (Fu et al. 2009)
- *"Autonomy (3 items): the learner should enjoy taking the initiative in game-playing and asserting total control over his or her choices in the game."* (Fu et al. 2009)
- *"Immersion (7 items): the game should lead the player into a state of immersion."* (Fu et al. 2009)
- *"Social Interaction (6 items): tasks in the game should become a means for players to interact socially."* (Fu et al. 2009)
- *"Knowledge Improvement (5 items): the game should increase the player's level of knowledge and skills while meeting the goal of the curriculum."* (Fu et al. 2009)

The EGameFlow scale builds upon Sweetser & Wyeth (2005) GameFlow framework to create a more comprehensive and reliable tool for evaluating user enjoyment of e-learning games. In EGameFlow the player skills from GameFlow is restructured as Knowledge Improvement (Fu et al. 2009).

## Part III

# Finalizing The Game

Prior to conducting the main experiment, which aimed to test the effectiveness of game-based learning compared to more traditional methods for learning Git, the game was still in development and had not yet undergone formative evaluation. Due to time constraints during the specialization project, it was necessary to carry out a testing and improvement phase in this master thesis to finalize the game and ensure it was ready for the main experiment. This phase was essential to accurately compare the learning outcomes between the two methods.

The finished game can be found here:

Code link: **Gitlab** or **GitHub**

Game link: **Git Cooking Game**

A downloadable video demo of the game can be accessed here:

Game demo: **Git Cooking Demo Video Download**

## Chapter 7

# Testing

This chapter focuses on the practical implementation of the alpha and beta testing stages that were conducted on the prototype. We will provide a detailed account of how these testing methods were executed. The theoretical framework for these testing methods was previously explored in the prestudy (see Part II, Chapter 6).

---

## 7.1 Alpha & beta testing

Testing and evaluating the effectiveness of the game was one of the primary goals of this thesis, making it crucial for the game to meet specific quality standards regarding both its ability to incorporate various Git concepts and its usability for players. To achieve this, we conducted alpha and beta testing to efficiently identify and resolve any bugs and issues.

### 7.1.1 Alpha testing

The testing process began with alpha testing, which was done in two phases: first internally within the team to identify any major bugs and issues with the game, then with external participants with no prior experience with the game. Complying with the insights gained in subsection 6.2.1, the alpha testing was done in-house - by the authors, or with help of friends of the authors; where the goal was to address major usability issues, discover bugs, and inform improvements to the game.

#### Internal testing

Based on the research in subsection 6.2.1, it is recommended that the prototype closely resembles the final product prior to user testing in order to obtain more accurate feedback. Therefore, the goal of internal testing was to bridge this gap and refine the prototype until it resembled a final product. Being aware of some missing pieces and issues with the game, a significant portion of the internal testing was dedicated to improving the game's playability. This involved addressing incomplete implementation of Git concepts, identifying and resolving bugs, and balancing various aspects of the gameplay. To track bugs during this phase, a Kanban board was utilized, with bugs continuously added to the board and prioritized based on their potential impact on user playability and likelihood of occurrence. In addition, suggestions for gameplay improvements were added to the board and discussed before being implemented in the game. To discover the bugs, each developer was tasked with playing through the game normally to discover gameplay and playability issues. Additionally, edge cases for various gameplay elements were tested to discover game-breaking bugs and logical errors. When adding changes during the internal testing phase, we continued to follow the development methodology described in subsection 4.2.1.

The improvements that were implemented based on the internal testing results are detailed in subsection 8.1.1. Conducting this testing enabled us to address most of the bugs and gameplay issues, preparing the game for



---

valuable feedback and new insights from external participants in the next phase.

## External testing

The second phase of the alpha test involved testing the game on external participants who had no prior experience with the game. The main focus during this phase was to evaluate the game’s usability and playability, as well as finding bugs not addressed in the internal testing phase. To achieve this, we recruited a small group of five friends of the authors, who were willing to participate in the test; all of which had prior Git experience. Although using friends of the author in testing can introduce some bias, it is justified in alpha testing, as noted in subsection 6.2.1, since the main goal is to resolve significant issues, while target group-specific issues will be addressed in beta testing. The test design was inspired by the *playtest* method described in subsection 6.2.1, meaning that the participants were instructed to play without restrictions, and without interruption from moderators. Since we are following the *playtest* method, we must also be aware of its limitations. Davis et al. (2005) recommends a larger sample size of 25-35 people for sufficient insight for improving gameplay, however for finding usability errors, Davis et al. (2005) agrees with smaller sample sizes, where usually five people are needed to find most bugs and errors (Nielsen et al. 2006). Therefore, when interpreting the result of the alpha test, we were careful to draw conclusions on feedback regarding gameplay, and focused mostly on bug reports and usability issues. Additionally, because the game is relatively short, taking approximately 20 minutes to play, we did not test specific scenarios as recommended by Davis et al. (2005). Instead, the entire game was treated as a scenario and assessed as a whole.

For the test, the participants were sent an informational letter outlining the test’s purpose and the steps they needed to follow. The participants were instructed to play as long as they desired, and on completion they were asked to fill out a digital survey. The survey contained a questionnaire with both standardized and ad hoc questions, which included a System Usability Score (SUS) questionnaire, general gameplay questions, and feedback on the game’s bugs and issues. Specifically the participants were asked about reporting game-breaking bugs, and bugs that reduced playability of the game. The questionnaire and results are available in Appendix A2. The results of the test and the improvements made after performing this external testing phase is presented in subsection 8.1.2.

---

### 7.1.2 Beta testing

The second phase of testing the game was through beta testing. The beta testing was performed after improvements, based on feedback from alpha testing, were made to the game. The beta testing was executed similarly to the alpha testing, with the exception that participants were exclusively selected from the game's target audience. The goal was to recruit testers with little to no prior experience with Git to evaluate the effectiveness of the game's tutorials and assess how easy it was to understand the game-play without prior knowledge of Git. Conducting beta testing with people unrelated to the authors was crucial for obtaining unbiased feedback and ensuring that the game met the expected quality standards for a positive user experience. The participants were recruited by hanging up flyers around campus at NTNU, with a QR code directing them to the test.

As discussed in subsection 6.2.1, beta testing serves as a means of verifying whether the application effectively achieves its objective of providing learning on a particular topic. Accordingly, the beta testing phase incorporated additional questions regarding perceived learning that were not included in the alpha testing. The purpose of including these questions was to evaluate whether the prototype effectively conveyed a sense of learning to the player. The actual learning outcomes of the game will be examined later, in the experiment detailed in Part IV. Nonetheless, assessing participants' perception of learning during beta testing can provide meaningful feedback, which can inform improvements to the educational aspects of the game.

The questionnaire and results from the beta test are available in Appendix A3. The results of the test and the improvements made after performing this testing phase are presented in section 8.2.

## Chapter 8

# Improvements

This chapter builds upon the previous chapter, which covered the execution of the testing phase of the prototype. Here, we will present the results of these tests and the subsequent improvements that were made based on them. We will provide a detailed account of the outcomes of each testing stage, and explain how they influenced the final version of the game. By examining the practical implementation of these testing methods, we can better understand of the strengths and weaknesses of the prototype, as well as the impact of each improvement on the final product.

---

## 8.1 Alpha test results

As previously mentioned in subsection 7.1.1, the alpha test was conducted in two phases, where the first phase was conducted internally, and the second externally.

### 8.1.1 Internal testing

During the internal testing phase, the game underwent significant updates and improvements, as it was found to be lacking in certain areas. Below are some of the most significant improvements that were implemented during the internal testing phase of the first alpha testing:

- **git merge functionality:** was added to introduce the concept to players at the end of each day. Merging is an essential concept within Git and was a concept we wanted to introduce to the player. (See Figure 8.1)
- **git push origin [branch] functionality:** was added in order to be able to push changes from another branch that the player is currently on.
- **git clone functionality:** was added to introduce the cloning phase of Git, as this is an essential concept and usually one of the first things one learns in Git. (See Figure 8.2)
- **difficulty modes functionality:** was added to reward players with efficient use of Git commands, by increasing revenue and reducing day length. In addition, it was implemented to introduce the `git clone` command more than once. (See Figure 8.3)
- **updated help/tutorial screen:** in addition to listing all tutorials, we added a list and use cases for each of the available Git commands in the game, in addition to a list of the main git concepts for those interested. (See Figure 8.4)
- **general balancing:** was added to create a better feeling of progress for the player in terms of upgrades, unlocks, and revenue for each day.
- **general bugs:** various bugs were discovered and fixed.

---

## Git merge functionality

As merging in Git is a fundamental and essential concept, we found it important to include it in our game to cover the most important Git flows. Since merging usually takes place through Git platforms like GitHub<sup>1</sup>, GitLab<sup>2</sup>, etc, we opted to create an interface for merging rather than having players execute Git commands. The merge screen, shown in Figure 8.1, provides a user-friendly interface for merging branches into the main branch.

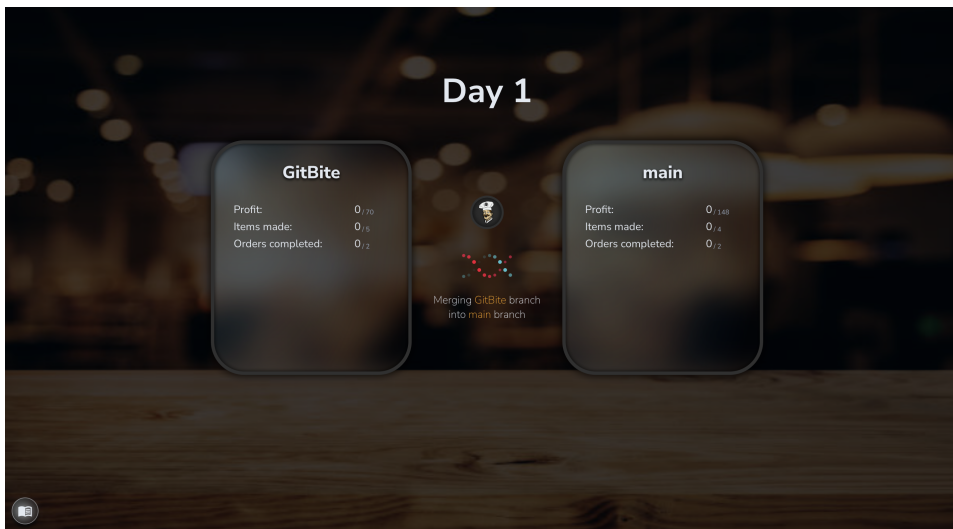


Figure 8.1: **Merge screen:** merges all branches into the main branch.

---

<sup>1</sup><https://github.com/>

<sup>2</sup><https://about.gitlab.com/>

---

## Git clone functionality

In addition to merging, cloning is another crucial concept in Git that we incorporated into the game's Git flow. If the repository has not yet been cloned, players are given a repository URL and prompted to use the `git clone [URL]` command to access the project, shown in Figure 8.2. Once the repository has been cloned, players can use `git fetch` to retrieve updates from the remote, following the standard Git workflow. New repositories are unlocked as the player progresses in the game.

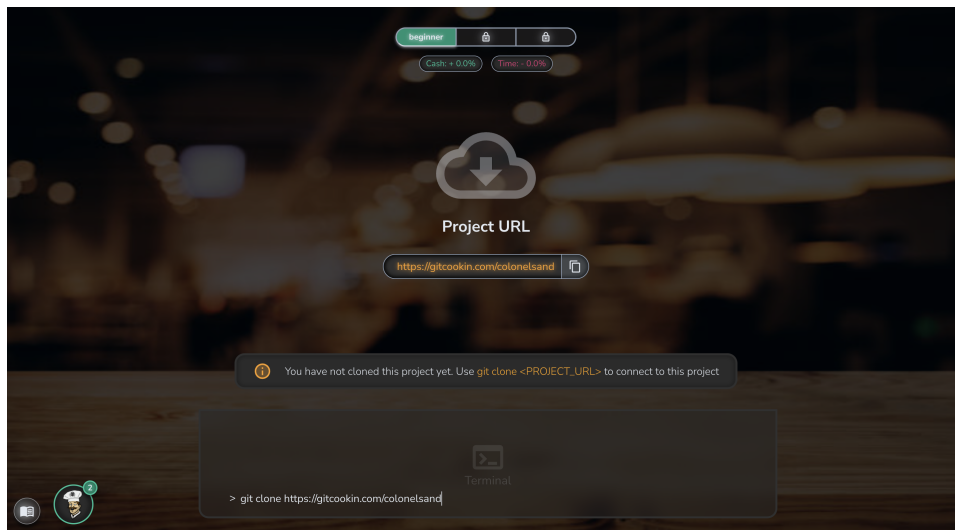


Figure 8.2: **Clone screen:** provides repository URL that can be cloned.

---

## Difficulty modes functionality

To enhance the sense of progression and reward, we introduced a new feature where difficulties are unlocked on specific days, allowing you to earn more and face greater challenges. Each difficulty represents a distinct repository and can be accessed by cloning it. By completing tasks in each difficulty, you will receive an earnings multiplier, but the time allotted for each day will be reduced, as shown in Figure 8.3.

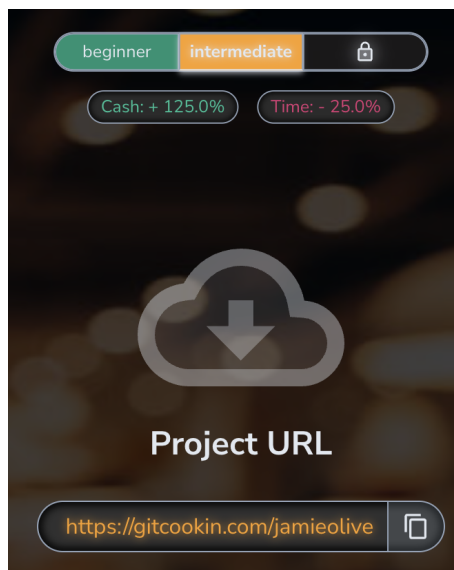


Figure 8.3: **New difficulties:** increase earnings but decrease daytime, and represent a new repository.

---

## Updated help screen

If players want to gain a deeper understanding of Git commands and concepts beyond what is covered in the tutorials, we included a dedicated section for this purpose. While tutorials are designed to be concise and engaging to avoid overwhelming players during the initial phase of the game, they cannot cover everything in detail. To address this, we shortened the tutorials and included more comprehensive information about Git commands and concepts in the help screen. This way, players who are interested can access additional information by navigating through the help screen, as seen in Figure 8.4.

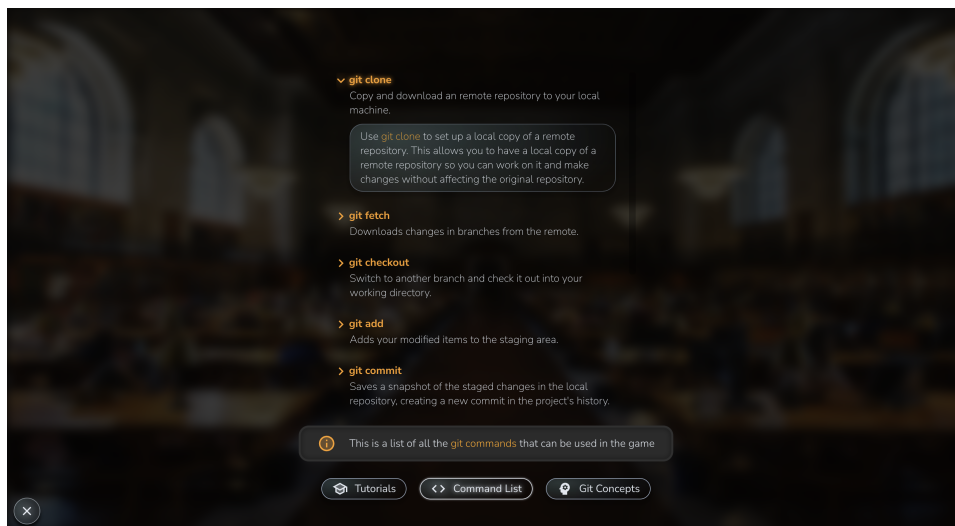


Figure 8.4: **Improved help screen:** includes re-playable tutorials, and information about available Git commands and Git concepts.



---

## 8.1.2 External testing

The external alpha testing phase generated a wealth of valuable insights and feedback. Since the testers were not involved in the game's development, they were able to identify areas of the game that were difficult for new players to understand, as well as issues that were undiscovered in development. The questionnaire and results of the external testing can be found in Appendix A2.

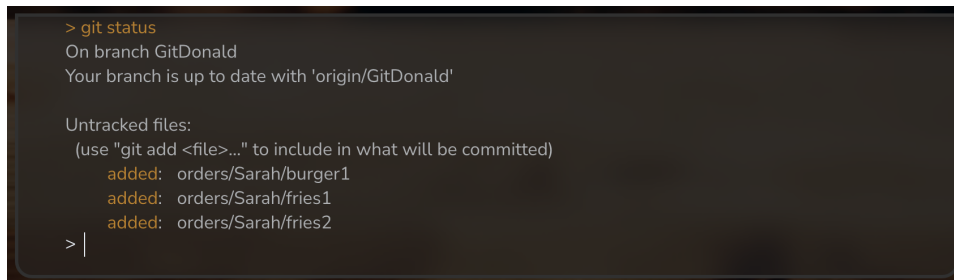
In accordance with the recommendation outlined in subsection 6.2.1, we made a conscious effort to differentiate between at-game frustration and in-game frustration while assessing user feedback; where the former needs to be addressed and fixed, whilst the latter might benefit gameplay. For instance, one user expressed frustration about not being able to use the "git branch" command immediately. This frustration was categorized as in-game frustration, as it was a deliberate part of the game, designed to encourage players to purchase the command in the game's store. As a result we concluded that it did not require any modifications. When it comes to at-game frustration, it is important to make necessary improvements. Following is a list of the improvements that were made based on the feedback we received during the second phase of alpha testing with external participants. This feedback was collected through questionnaires and include issues that were identified as at-game frustration:

- **Tutorial text speed:** was increased and made possible to fast forward, as it was noted to be too slow by users and distracting for reading the tutorial texts.
- **Unknown use case for commit list:** removed the commit list, as it had no functionality within the game, and participants found it more confusing.  
(See Figure 8.6)
- **Updated terminal responses:** added colors, more information, and hints in terminal responses when executing git commands. As users noted a lack of feedback within the game.  
(See Figure 8.5)
- **Updated info-box:** updated information at info-box located at the bottom right in the work screen to improve feedback within the game.  
(See Figure 8.6)

---

## Updated terminal responses

During our alpha testing phase, some external testers reported difficulties and misunderstandings when executing invalid Git commands and interpreting the corresponding error responses. To address these issues, we implemented several improvements. Firstly, we added color coding to the terminal responses to highlight the most critical parts of each response, as shown in Figure 8.5. This makes it easier for users to quickly identify errors and relevant information. Additionally, we replaced some of the generic error messages with more helpful hints and tips for proper command usage. This change should further reduce confusion and provide users with the guidance they need to execute Git commands correctly.



```
> git status
On branch GitDonald
Your branch is up to date with 'origin/GitDonald'

Untracked files:
(use "git add <file>..." to include in what will be committed)
  added: orders/Sarah/burger1
  added: orders/Sarah/fries1
  added: orders/Sarah/fries2
> |
```

Figure 8.5: **Updated terminal**: includes color-coded responses, and improved response information.

---

## Updated info-box

Several external testers reported confusion about the list of commits displayed in the game, as they were unsure of its purpose. Upon investigation, we discovered that the list was included merely to indicate the number of commits made and did not provide any other meaningful information. Additionally, testers reported a lack of clarity regarding the active branch, which created further confusion. To address these issues, we made several changes. Firstly, we removed the list of commits and replaced it with information on the total number of commits registered on the active branch. This change helps to simplify the user interface and reduce unnecessary clutter. Secondly, we added information regarding the active branch, including the name of the branch, whether it is synced with the remote repository, and whether changes need to be pushed. These changes will provide users with clearer and more useful information, as shown in Figure 8.6.

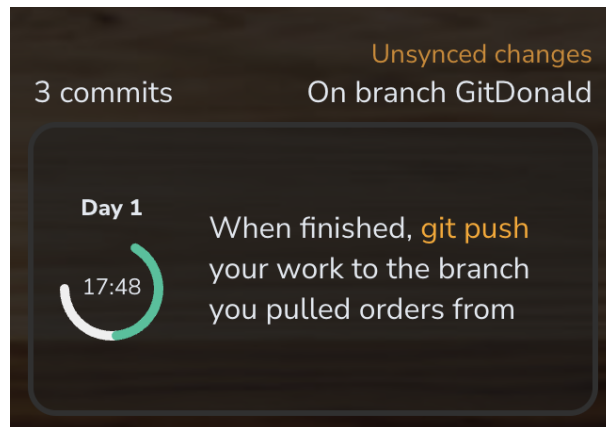


Figure 8.6: **Updated info-box**: removed unused commit history, and display current branch information.

---

## 8.2 Beta test results

The results from the beta test showed that many of the game's most glaring usability issues and bugs had been fixed after the alpha testing. This can be seen from the fact that 3 out of 5 alpha testers found bugs that prevented them from playing the game, whilst zero beta testers reported any game breaking bugs. Additionally, 4 out of 5 alpha testers reported bugs that reduced the quality of the game, whilst the one issue reported in the beta testing was related to spelling errors. Continuing, the calculated System Usability Score for the prototype used in the alpha test was 75, whilst the System Usability Score for the improved prototype used in the beta test was 82. This indicates that the usability of the game has improved as a result of the changes made after alpha testing. The questionnaire and results from the beta tests can be found in section A3.

As a result of the few negative remarks received during the beta testing, there weren't any significant game improvements or bug fixes following the test. However, we still gained valuable feedback, as the test enabled us to evaluate the game's playability among our target audience, who have limited to no experience with Git. This was a crucial aspect that we needed to test, and the beta test provided us with valuable insights into this area. In addition, the perceived learning reported from the beta testers showed that our game has the potential to provide learning. Therefore, it appears that no major changes are required before conducting the main experiment for assessing the prototype's effect on learning.

However, there were some small improvements made after the beta testing, which includes:

- **Bug where a day did not end:** Fixed a bug where the day was reset and started anew when the day should have ended.
- **Small design fixes and spelling errors:** Fixed some small design issues and spelling errors that were reported.

# Part IV

## Experiment, Results, and Discussion

After successfully testing and evaluating the game prototype in the previous stage, the prototype became ready to be utilized in the main experiment. The objective of the main experiment was to compare the learning achieved through a game-based learning approach for learning Git to the learning achieved through a traditional learning approach. This comparative analysis allowed us to evaluate the effectiveness of each approach, as well as the learners' experience and enjoyment with each approach. This part will present the main experiment that was conducted, the results obtained, and the ensuing discussion.

## Chapter 9

# Experiment

This chapter outlines the main experiment of this thesis, which aimed to compare the effectiveness of a game-based learning approach for learning Git (experimental group), compared to a traditional learning approach (control group). In addition to learning outcomes, this experiment also considered factors such as learning experience and enjoyment for a comprehensive evaluation.

---

## 9.1 Structure

The structure of the experiment is based on the theory behind gamma testing previously discussed in the pre-study (See part Part II, chapter 6). The experiment follows the following structure:

1. **Identify the participant:** This part is useful for identifying demographic information for the test, such as the participant's current study program, which year of study, and previous experience with Git.
2. **Pre-test:** The goal of the pre-test is to assess the current Git competence level of the participants. As previously discussed in subsection 6.2.2, this is a crucial step to establish a baseline for comparing relative learning gains and to control for any initial differences between the experimental and control groups.
3. **Learning Git:** This is the learning phase of the experiment, where the experimental group will learn Git through the game, while the control group will learn through a traditional approach. As outlined in subsection 6.2.2, it is essential to control for any differences between the experimental and control groups, which will be further elaborated in section 9.2.
4. **Post-test:** The goal of the post-test is to evaluate the increase in knowledge by providing a competence test similar to the pre-test. The post-test should however consist of different questions than the pre-test to prevent practice effect, as discussed in subsection 6.2.2.
5. **Survey:** The last part of the experiment is to obtain feedback on the participant's perception and experience with their given method of learning.

## 9.2 Setup

The experiment was conducted entirely online through four digital surveys hosted on Nettskjema<sup>1</sup>, which all adhere to the structure outlined in section 9.1. The four surveys can be categorized as follows:

- Control group - Version A
- Control group - Version B

---

<sup>1</sup><https://nettskjema.no/>

- 
- Experimental group - Version A
  - Experimental group - Version B.

The only internal difference between Version A and Version B for both the experimental and control groups is the order of the pre-test and post-test Git competence questions. In Version A, question set 1 is administered before participants learn Git, followed by question set 2 after learning Git. In contrast, Version B follows the opposite sequence. As explained in subsection 6.2.2, this approach helps mitigate practice effects that may occur when using similar pre-tests and post-tests, and also enables control of test scores between the two question sets.

The control group and experimental group share certain similarities, such as identical pre-test and post-test questions, identical demographic questions, identical post-learning survey questions regarding opinions on the learning process, and the same reward for participation. However, there are two differences between the surveys for the two groups. Firstly, the experimental group is redirected to the game for learning Git at the halfway point of the survey, while the control group is redirected to a text document for learning Git. Secondly, the experimental group answer game-specific questions at the end of the survey to provide feedback on the game. Besides these differences, the setup for both groups is identical.

Prior to answering the digital surveys, all participants are directed to an information letter (see Appendix A4) providing details on the purpose and procedures of the experiment. They are then randomly assigned to one of the four versions of the online questionnaire without knowledge of the different versions. This approach ensures similar conditions between the control and experimental groups, which is essential for minimizing confounding effects (see subsection 6.2.2). Furthermore, participants are not informed about their group and version assignment, nor the existence of the other groups and versions, to prevent bias from affecting their responses.

### 9.2.1 Distributing Groups

To ensure an even distribution into into experimental and control group, as-well as even distribution into Version A and Version B within the groups, a software called Linkly<sup>2</sup> was utilized. This software redirects traffic from a single link evenly across multiple links. As a result, after reading the information letter, all participants click the same link and are then randomly assigned to one of the four surveys through Linkly, ensuring an even distribution across all groups and versions.

---

<sup>2</sup><https://linklyhq.com/>



---

## 9.2.2 Creating the Git competence test

To ensure that the Git competence test closely aligns with the expected learning outcome of the game, it was created from scratch. The test is designed to measure both theoretical and practical knowledge of Git. For practical knowledge, the test presents a scenario and asks which Git command should be used, and the answer is accepted as a raw text input to simulate entering a command in the terminal. For theoretical knowledge, multiple-choice questions prompt the user to choose between four different options, testing their understanding of Git concepts. For example, a question may ask what a repository is in Git.

Following the research described in subsection 6.2.2, two similar versions of the test were created to reduce practice effect bias. The versions were designed to be similar in type and difficulty, but different in terms of content. To ensure similarity, the second version is based on the first version, with slight alterations. For the practical questions, the presented scenario is different between the two versions, but we are looking for the same Git commands as the correct answer. For the theoretical questions, the phrasing of both the questions and the correct answers are slightly different between versions. Additionally the incorrect answer options are entirely different between the two versions. The two test versions can be found in Appendix A5.

To avoid confounding effects from guessing, each question includes an *"I don't know"* option, and participants are encouraged to choose the *"I don't know"* option if they genuinely do not know the answer. Additionally, participants are explained that their answers are not used to pass judgment on their skills, and they are asked to refrain from using any aids or additional materials when answering the questions.

## 9.2.3 Creating the traditional learning document

In order to isolate the effect of the game on learning, it is essential to ensure similar conditions between the experimental and control groups, as discussed in subsection 6.2.2. To achieve this, it is recommended that the control group participates in another educational activity. To address this, we faced the decision of either using existing online resources for learning Git or creating our own resource. As outlined in subsection 6.2.2 it is important for the control group to receive similar content, guidance, and level of difficulty as the experimental group. For that reason, we chose to create our own educational resource for learning Git, designed specifically with these variables in mind. The resource takes the form of a text document, which introduces various topics related to the Git version control system.

---

The document presented for the traditional learning approach can be found in Appendix A6.

To ensure consistency between the learning experiences of the experimental and control groups, it was essential to provide similar content to both. As the game includes an in-game tutorial that introduces basic Git concepts and commands, we aimed to replicate this content as closely as possible for the control group. To achieve this, we copied and reformatted all tutorials presented in the game into a text-based format that follows the same logical sequence for each concept introduced. Additionally, we included information from the game’s help screen, which explains various Git concepts and commands, directly into the traditional learning document. As a result, the only significant difference between the learning content of the experimental and control groups is the context of learning, the gameplay of the game, and the format of the tutorial content. In the game, the tutorial content is presented visually and is embedded in a fantasy context, while in the traditional learning document, Git concepts are introduced in a text-based format based in reality. Nonetheless, we aimed to make the content of the traditional learning document as similar as possible to the game’s educational content.

#### **9.2.4 Standardized and ad hoc questions**

For the experimental group, a questionnaire for gathering feedback about the game is also included after they have played the game. This questionnaire includes standardized questions from the EGameFlow framework discussed in section 6.3. Specifically, questions from the dimensions Immersion, Concentration and Knowledge Improvement, was taken from the EGameFlow framework in order to gain feedback in these dimensions about the game. These dimensions were picked out to allow us to gain feedback about the game’s ability to engage the users and capture their attention, in addition to measuring their perceived learning improvement from playing the game. Additionally, some ad hoc questions were added to gain specific feedback on our game - these questions were related to specific goals we had with the game such as linking learning objectives with game objectives and familiar gameplay.

### **9.3 Execution**

Once the experimental design and procedures had been established, the experiment was ready for execution. However, prior to initiating the experiment, a recruitment phase was essential to ensure a sufficient number

---

of participants were enrolled. And following the experiment, the collected data had to be appropriately analyzed to derive meaningful conclusions.

### 9.3.1 Recruitment

In terms of recruiting participants, we wanted to get a decently sized pool in order to get enough data to draw patterns and conclusions. As the experiment support participants with previous experience with Git on a basic level, we wanted to enable such participants to join as well as people with no experience. In order to recruit enough participants, a few different approaches were used.

Firstly, we reached out to professors in charge of subjects that would indicate either that their students had learned some Git, or were going to learn Git at a later stage. By emailing these professors, we addressed the situation and asked if they would be interested to share the test with their students. In addition, a few of the professors had suggestions in terms of what other professors would be relevant to share this experiment with. These professors were also contacted and requested to share the test with their students.

Secondly, we reached out to students enrolled in study programs at NTNU who we knew would learn Git at a later stage or had already learned some Git. We specifically targeted students in the early years of study programs that teach Git at a later stage and also those in study programs currently learning Git. The students were contacted through private Facebook<sup>3</sup> groups associated with their respective study programs, where they were asked to participate.

### 9.3.2 Data analysis

After the experiment was completed, the results from the surveys were transferred over to Excel in order to organize and extract the relevant data for analysis. Participants were identified by a unique ID in the spreadsheet, where their email was not included to uphold NSD regulations. The pre-tests and post-tests were scored, and data such as previous Git experience, current study year, test scores, method, and test orders were collected in one table to perform different statistical analyses and understand the data that had been collected.

In terms of the statistical analysis, the SPSS<sup>4</sup> statistics tool from IBM was used to perform the different analyses effectively.

---

<sup>3</sup><https://www.facebook.com/>

<sup>4</sup><https://www.ibm.com/products/spss-statistics>

---

### **9.3.3 Interviews**

Following the data analysis, interviews were conducted to better interpret the results. These interviews followed a semi-structured approach, where questions and themes were prepared beforehand, while allowing for freedom to ask additional questions when they were relevant throughout the interview.

## Chapter 10

# Results

In this chapter the results of the experiment will be presented. The results are separated into different sections of the survey including demographic, experiment results with statistical analysis, learning experience, and gameplay feedback.

The online survey received 19 answers, where 2 of the answers were excluded from the study. One participant spent only 4 minutes in total for both answering the survey and learning Git, and was therefore excluded from the study for lack of sincerity. Another participant scored full points on the pre-test of Git competence and was excluded as they did not fit our target group of having no or some experience with Git. Additionally their answer could not provide meaningful data in analysis as there was no way to measure improved learning when one scores full points on the pre-test. So, the results presented in this section is based on the answers of the remaining 17 participants.

---

## 10.1 Demographic

This section will present data on demographics to provide insights into the participation pool of this study.

### Study program

To be able to track where our 17 participants came from, and potentially track how they fit our target group, the participants were asked about their study program and current study year. The majority of participants in the study come from Cybernetics and Robotics (*MTTK*) with 7 participants, followed by Computer Science (*MTDT*) with 3 participants, Electronic System Design (*MTELSYS*) with 2 participants, and then Informatics (*MSIT*), Engineering & ICT (*MTING*), Communication Technology (*MTKOM*), Graphic Design and FHS (*Folkehøyskole*) with 1 participant each, as shown in Figure 10.1.

All of the fields of study represented in the experiment have a connection to Git through their coursework, except for Graphic Design and FHS. Participants from these two fields either expressed an interest in learning Git or were planning to pursue a technical field of study in the near future.

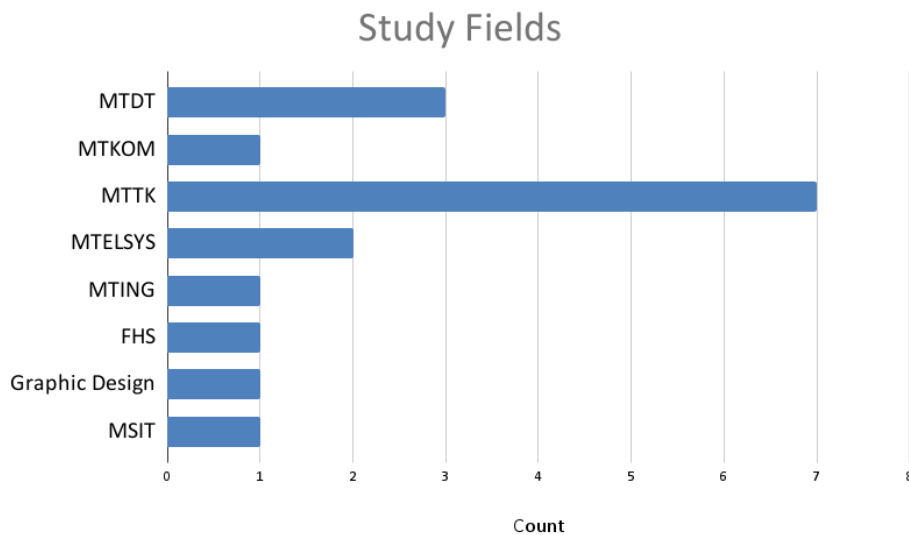


Figure 10.1: **Bar chart on participants' field of studies:** *MTDT* (*Computer Science*), *MTKOM* (*Communication Technology*), *MTTK* (*Cybernetics and Robotics*), *MTELSYS* (*Electronic Systemdesign*), *MTING* (*Engineering & ICT*), *FHS* (*Folkehøyskole*), *Graphic Design*, and *MSIT* (*Informatics*).

---

The participants' study year for each of the unique study programs reported by the participants can be seen in in Table 10.1. From this table we can extract that some of the participants are expected to have learned Git at NTNU before participating in our experiment. For example, we know that students in 5th year of MTDT, and higher than 3th year of MTTK have learned Git from courses at NTNU - so at least 8 of the 17 participants are expected to have learned some Git at NTNU beforehand.

Year	MTDT	MTKOM	MTTK	MTELSYS	MTING	MSIT	Other
1	-	1	1	-	-	-	1
2	1	-	-	-	-	-	-
3	-	-	3	2	1	-	1
4	-	-	2	-	-	-	-
5	2	-	1	-	-	1	-

Table 10.1: **Number of students per year from given study program:** MTDT (*Computer Science*), MTKOM (*Comunication Technology*), MTTK (*Cybernetics and Robotics*), MTELSYS (*Electronic Systemdesign*), MTING (*Engineering & ICT*), MSIT (*Informatics*), Other (Folkehøyskole and Graphic Design).

---

## Git Experience

Even though 8 of the 17 participants are expected to have learned some Git at NTNU before the experiment, we cannot conclude that these students are not fit for our study, as we do not know their level of competence with Git. To gather this we asked the participants to answer a yes or no question: 'Do you have any experience using Git?'. If participants answered 'yes', they would get the follow up question: 'How would you rate your competence with Git?' on a scale from 1 to 5. Here, the level of Git competence seem to be evenly distributed across the various levels of experience, as depicted in Figure 10.2.

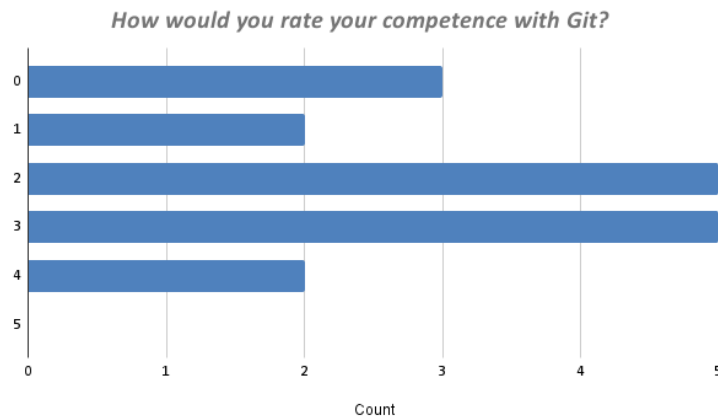


Figure 10.2: **Bar chart on participants' git experience.** A Git competence of 0 in the figure represents those answering 'no' to having any experience using Git.

We see in Figure 10.2 that 3 (18%) of the participants have no experience with Git. Additionally, 12 (70%) of the participants rate themselves between 1 and 3 in Git competence. Assuming people accurately rate their own experience with Git, we can say that approximately 88% (15 out of 17) of the respondents fit our target group of having some or no experience with Git. The remaining 2 participants (12%) rate themselves as 4 on Git competence, so we can assume that some of the participants does not fit our target group exactly as they are quite experienced with Git. Still none of the participants rated themselves with a full score of 5 on Git competence.



---

## Gender

The gender of each participant was recorded to provide extra data points and insights into our participant pool and to possibly help contextualize our findings. The gender distribution of the participants is presented in Figure 10.3, which reveals a notable imbalance in the representation of males (12 out of 17) and females (5 out of 17), particularly in the theory method where a majority of the participants were male (7 out of 8).

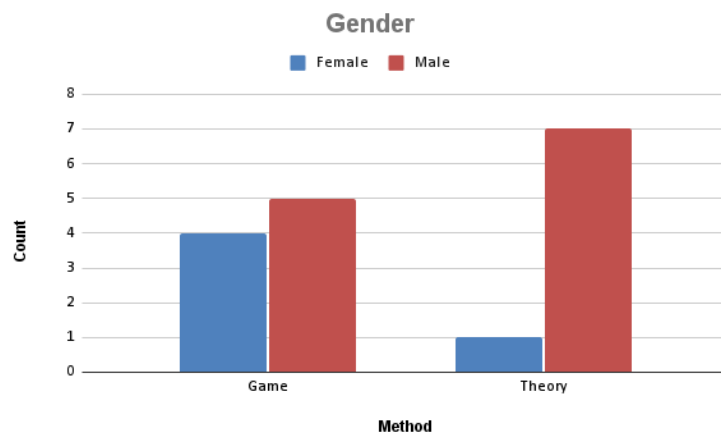


Figure 10.3: Bar chart on participants' gender in each method of learning.

---

## Experience with Code Editors

Since our game features a terminal interface and the layout of the game is reminiscent of code editors used by software developers, we wanted to know if people were familiar with code editors before playing the game. The majority of participants (14 out of 17), reported agreeing or strongly agreeing to having prior experience with code editors as can be seen in Figure 10.4. Only 2 (12%) of the participants were inexperienced (answering 'Strongly Disagree' or 'Disagree') with a code editor, however none of these participants were in the game group. Lastly, 1 participant was neutral to the statement.

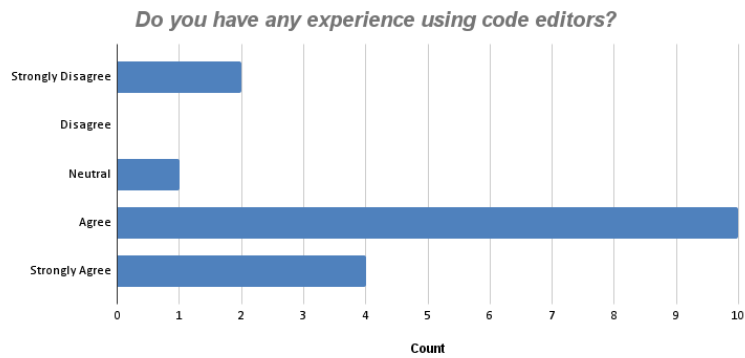


Figure 10.4: Bar chart on participants' code editor experience.

---

## Preferred learning method

We were interested in controlling for differences in participants' preferred method for learning Git between the groups. The participants were therefore asked 'Which method would you prefer for learning Git?', where they could choose multiple options from: 'Watch video tutorials', 'Read documentation on the internet', 'Practical activities', 'Playing a game', and 'Other'. The survey recorded no significant difference between the participants within the the methods 'Read documentation on the internet' and 'Playing a game', as can be seen in Figure 10.5. We also see that "Practical activities" (39% and 25%), "Playing a game" (both 30%), and "Watch video tutorials" (17% and 25%), are the preferred methods of learning for the participants ranked respectively. Additionally, there are some differences between the groups for the options: 'Watch video tutorials' and 'Practical activities', where the game group is more inclined towards practical activities and the theory group towards watching video tutorials. Lastly, the one participant who answered 'other' specified that they preferred 'Experimenting with it for myself'.

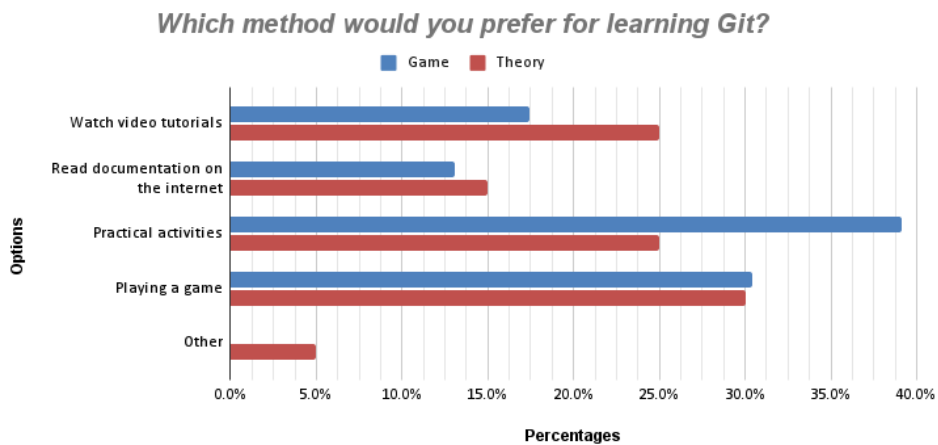


Figure 10.5: Bar chart on participants' preferred learning method. (Multiple choice)

---

## Interest in learning Git

Even though participants might be within our target group and has the potential for learning Git, an important factor that might come to play is the participants' motivation and interest for learning it. In order to get this insight, we asked the participants *'I am interested in learning and becoming better at Git'*, and they could choose between 5 options ranging from *'Strongly Disagree'* to *'Strongly Agree'*. Figure 10.6 show the average score of 3.5 for the game group and 3.25 for the theory group, when *'Strongly Disagree'*=0 and *'Strongly Agree'*=4. So, in total the participant averagely agree to being interested in learning and becoming better at git in both participation pools. Specifically, 0 participants answered *'Disagree'* or *'Strongly Disagree'*, 1 participant answered *'Neutral'*, 9 participants *'Agree'*, and 7 participants answered *'Strongly Agree'*.

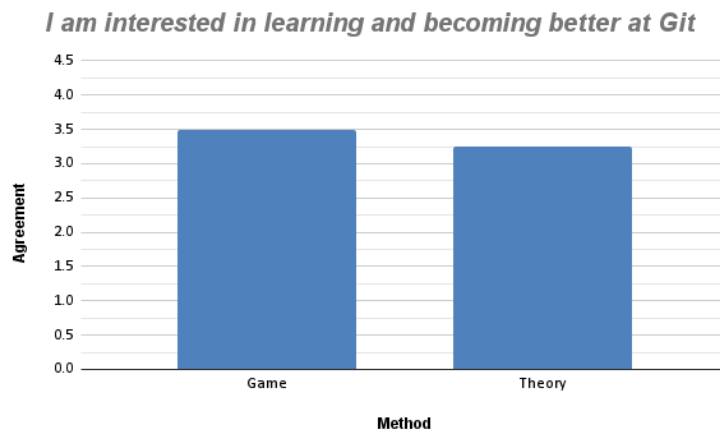


Figure 10.6: Bar chart on participants' interest in learning and becoming better at Git. (0 = Strongly Disagree, 4 = Strongly Agree)

---

## Experience with gaming

Someone who has a lot of experience with playing games, might have an easier time playing and learning through the game than someone with less experience. This factor might play a large role on the results of the study, and was therefore recorded to help contextualize our findings. The participants were asked to choose between 5 options ranging from 'Strongly Disagree' to 'Strongly Agree' for the statement 'I am an experienced gamer'. The game group recorded on average being neutral (avg.=2) to the statement, while participants within the theory group on average agreed (avg.=2.75). This can be seen in Figure 10.7 where the options are scored with 'Strongly Disagree'=0 and 'Strongly Agree'=4.



Figure 10.7: Bar chart on participants' gaming experience. (0 = Strongly Disagree, 4 = Strongly Agree)

More specifically, within the game group: 1 strongly agreed, 3 agreed, 1 was neutral, 3 disagreed, and 1 strongly disagreed with the statement. Meaning that there was an equal number of experienced and inexperienced gamers in the game group.

---

## 10.2 Experiment Results

This section will present the result of the main experiment which aimed to test the effect on learning from playing the game (experimental group) vs. learning by reading a document (control group); the effects were measured through a Git competence pre-test and Git competence post-test to assess learning gain. ANCOVA, previously discussed in section 2.2, was used to perform a statistical analysis on the collected data; it utilizes the pre-test scores as a co-variate to control for the initial differences between the control group and the experimental group. The goal of the analysis is to test if the learning method has an impact on the post-test scores, which measures the learning gain of participants.

### 10.2.1 Assumptions

To perform the ANCOVA we need to account for its underlying assumptions of: independent groups, normality, independent observation, homogeneity of regressions slopes, linearity, independent co-variate, and homogeneity of variance (see section 2.2).

**Independent groups:** Participants were randomly assigned to different groups by using the Linkly software tool, where one link can distribute traffic to multiple destinations, where each of the four surveys had a 25% chance of being served to the participant. The assumption of independent groups is therefore met.

**Normality:** Normality was checked with a Shapiro-Wilk test, which indicate that the post-test scores are normally distributed in the control group,  $W(8)=.933$ ,  $p=.540$ . A Shapiro-Wilk test indicated that the scores are not normally distributed in the game group,  $W(9)=.790$ ,  $p=0.015$ .

**Independent observation:** The assumption of independent test results is met since none of the participants completed more than one of the four survey versions, indicating that there are no interrelationships between the groups or the survey responses.

**Homogeneity of regressions slopes:** By factoring the independent variables: method and test version, with the dependent variable: pre-test, through an ANCOVA test, we can inspect if there are interaction between the combinations of the three variables. The result of the test show that there is not a significant interaction between the combination of these variables:

- $Method*Version*PreTest$  -  $F(1,9)=.463$ ,  $p=.513$ ,

- $Method * Version$  -  $F(3,9)=1.570$ ,  $p=.263$ ,
- $Version * PreTest$  -  $F(1,9)=.3.981$ ,  $p=.077$ ,
- and  $Method * PreTest$  -  $F(1,9)=.979$ ,  $p=.348$ .

Since there no interaction between the co-variate and independent variables, the assumption of homogeneity of regression slopes is met.

**Linearity:** By plotting the dependent variable (post-test) against the co-variate (pre-test), the relationship should appear linear in the plots. The plots can be seen in Figure 10.8, and they possess linear characteristics.

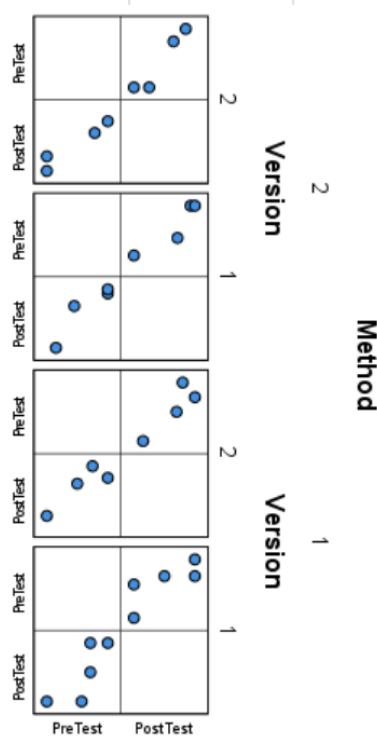


Figure 10.8: Plot of the dependent variable (post-test) against the co-variate (pre-test) showing linear relationship.

**Independent co-variate:** This essentially means that the pre-test scores are not affected by the grouping of the participants. Since the survey performs the pre-test before the experiment, the co-variate is therefore independent from the learning method grouping.

**Homogeneity of variance:** Levene's Test for Equality of Variances indicated that the assumption of homogeneity of variance has been met,  $F(3, 13)=.531$ ,  $p=.669$ .

---

## 10.2.2 Results

The results of the ANCOVA for measuring the effectiveness of the game based approach for learning Git, when controlling for pre-test scores, show that there is not a statistically significant difference between the experimental and control group for the post-test scores,  $F(1, 12) = 1.819$ ,  $p = .202$ ,  $\eta p^2 < .132$ . The estimated marginal means were approximately 1 point higher for the game group ( $M = 10.046$ ,  $E = .516$ ) vs. the control group ( $M = 9.024$ ,  $E = .548$ ). So, even though the result were better for the game group, this result is not statistically significant, and we can conclude that both methods performed equally well for teaching Git. The full ANCOVA test results are shown in Figure 10.9.

**Tests of Between-Subjects Effects**

Dependent Variable: PostTest

Source	Type III Sum of Squares	df	Mean Square	F	Sig.
Corrected Model	168.785 <sup>a</sup>	4	42.196	17.984	<.001
Intercept	66.093	1	66.093	28.168	<.001
PreTest	90.406	1	90.406	38.531	<.001
Method	4.268	1	4.268	1.819	.202
Version	.002	1	.002	.001	.977
Method * Version	.136	1	.136	.058	.814
Error	28.156	12	2.346		
Total	1750.250	17			
Corrected Total	196.941	16			

a. R Squared = .857 (Adjusted R Squared = .809)

Figure 10.9: ANCOVA results.

Upon reviewing the assumptions required for conducting the ANCOVA test, we have observed that normality is the only assumption that has not been met. As mentioned in subsection 2.2.1, the ANCOVA test is robust to violations of the normality assumption. However, upon closer inspection of the data, it was discovered that one single outlier was the cause for failure of the Shapiro-Wilk test. Removing the outlier gives a dataset which is normally distributed by the Shapiro-Wilk test,  $W(8)=.934$ ,  $p=.557$ . A new ANCOVA was performed with the removed outlier, which also yielded non-statistically significant results,  $F(1, 11) = 3.447$ ,  $p = .090$ ,  $\eta p^2 < .239$ , with estimated marginal means higher for the game group ( $M = 10.693$ ,  $E = .541$ ) vs. the control group ( $M = 9.248$ ,  $E = .536$ ). For this test all the other assumptions were also tested, and were all met.



---

## 10.3 Learning Experience

The following section will present data relevant to the learning phase (where the learning phase refers to either playing the game or reading the document) of the survey such as the participants' learning experience, learning outcome, and time spent learning.

### Enjoyment, motivation, and learning perception

After completing the learning phase, we wanted to get the participants opinion on their enjoyment, motivation and perceived learning given their learning method. Participants answered the following questions, between 5 options ranging from '*Strongly Disagree*' to '*Strongly Agree*', regarding their individual experience with the method of learning they were presented:

- I enjoyed the learning process for learning Git (Learning Enjoyment)
- I felt motivated while learning Git (Learning Motivation)
- I felt like I learned something (Learning Perception)

As can be seen in Figure 10.10, participants that were provided the game as a learning method recorded slightly higher enjoyment (2.75 vs. 2.125), motivation (3.0 vs. 2.125) and perception of learning (3.0 vs. 2.63) compared to the participants that learned through theory. The scores were calculated with '*Strongly Disagree*'=0 and '*Strongly Agree*'=4. Combining the three dimension, the game group has an average of 2.93, and the theory group an average of 2.13.

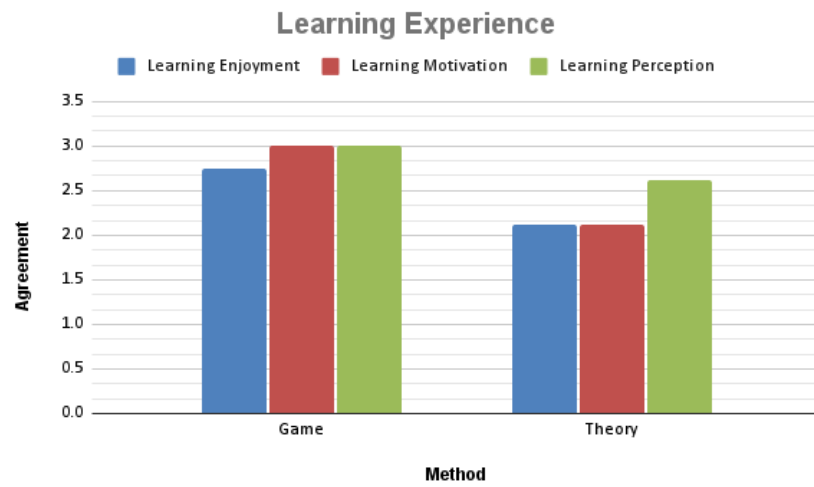


Figure 10.10: **Bar chart on participants' experience learning through each method** (0 = Strongly Disagree, 4 = Strongly Agree). Learning Enjoyment=*'I enjoyed the learning process for learning Git'*, Learning Motivation=*'I felt motivated while learning Git'*, Learning Perception=*'I felt like I learned something'*.

As participants answered on an ordinal scale, the Mann–Whitney U test (see subsection 2.2.3) was executed to determine if there was a statistical difference in these results. For learning enjoyment the result indicated that the difference between the groups was not statistically significant,  $U=18.50$ ,  $p=.080$ . For learning motivation there is a statistically significant difference,  $U=16.50$ ,  $p=.045$ , where the game group has a higher learning motivation (mean rank of 11.17) compared to the control group (mean rank of 6.56). For learning perception, the test indicates that there is no statistically significant difference between the groups,  $U=28.00$ ,  $p=.401$ .

---

## Engagement

To gather information about the participants engagement with their learning method, the participants were asked how long they spent in the learning phase and why they stopped when they stopped. As can be seen in Figure 10.11, participants that played the game averaged approximately 29 minutes of time spent learning, while the participants that learned through theory averaged approximately 14 minutes.

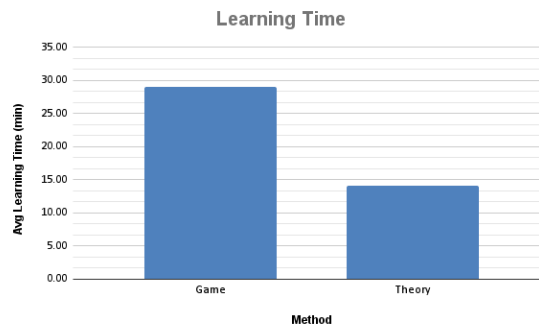


Figure 10.11: Bar chart on participants' average time (in minutes) spent learning through each method.

To check if this result is statistically significant, an independent sample t-test (see subsection 2.2.2) was conducted. The assumption of independent observation is met through the design of the experiment, where no participant is present in both groups. Levene's Test for Equality of Variances indicated that the assumption of homogeneity of variance has been met,  $F(1, 15)=1.195$ ,  $p=.292$ . The normality assumption is also met for both groups, which was tested with a Shapiro-Wilk test, for the game group:  $W(9)=.916$ ,  $p=.363$ , and for the control group:  $W(8)=0.924$ ,  $p=.462$ .

The result of the t-test shows that there is a statistically significant difference in time spent learning between the control group and the experimental group,  $t(15)=2.413$ ,  $p=.029$ . The means for the game group for time spent learning were almost 15 minutes longer ( $M=29.11$ ,  $SD=15.60$ ) vs. the control group ( $M=14.13$ ,  $E=8.48$ )

## Reason for stopping

When asked why they stopped the learning phase, the participants could choose multiple options out of 5 options: 'Boredom', 'Finished', 'Disinterested', 'Satisfied with learning outcome', and 'Other'; their answers are shown in Figure 10.12. The percentages are connected to each group respectively. Compared to the game group, the theory group recorded a significant

portion choosing the "finished" (40% vs. 13%) and "satisfied with learning outcome" (30% vs. 13%) options. Both groups chose 'boredom'(20% vs 20%) equally as a reason for stopping, and the game group was the only group which chose 'disinterest'(13%) as a reason for stopping.

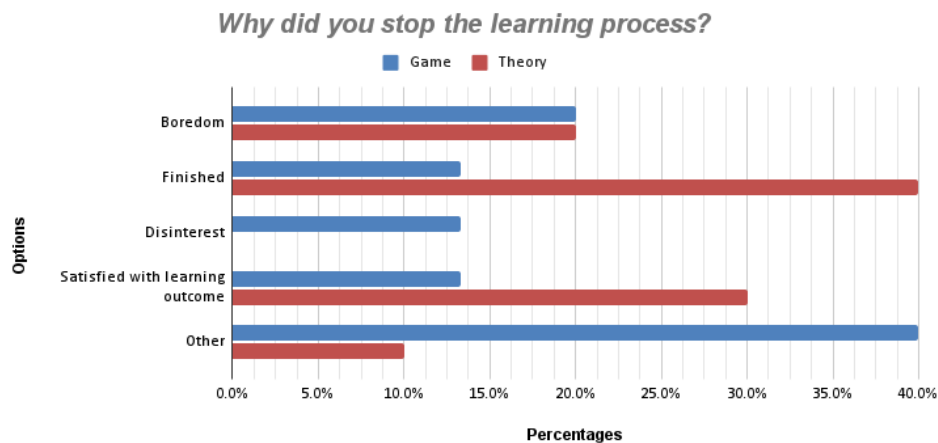


Figure 10.12: Bar chart on participants' reason for ending learning process. (multiple choice)

We also see a large portion of the participants playing the game choosing the "other" (40%) option compared to the theory group. One of the participants in the game group clarified that they had a meeting and therefore had to finish early. Others clarified that they had difficulty understanding the game and became stuck, leading to them giving up on the game. Here one reported that "It was difficult to understand how to git add, and no tutorial seemed to show how it was done. So I was stuck, cause i couldn't finish any orders, and gave up.". Another player reported that they couldn't understand the game mechanics and that they encountered a bug which made the unable to play any further. Additionally, a player reported that they didn't learn enough to move forward in the game. Lastly, one player lacked motivation to play, stating: "Based on the upgrades I saw I felt there wasn't all that much for me to work towards that I didn't know, given that I have some experience with git from before."

The one participant who choose "other" in the theory group had problems understanding the learning material, stating that "I did not understand much. It was a bit complicated because I don't have any knowledge from earlier."

---

## 10.4 Gameplay Feedback

This section will report the results of feedback regarding the game. This feedback uses half the sample size as the other half was provided theory as their method of learning Git. The following data is based on a sample size of 9.

### 10.4.1 EGameFlow Questions

#### Knowledge Improvement

In Table 10.2 the questions from EGameFlow regarding knowledge improvements are presented. 6 (66.6%) of the players recorded to either agree or strongly agree to the game increasing their knowledge. 9 players (100%) recorded either agreeing or strongly agreeing to catch the basic ideas of the knowledge taught. 8 players (88.9%) either agreed or strongly agreed to try to apply the knowledge in the game. 5 players (55.5%) either agreed or strongly agreed that the game motivates to integrate the knowledge taught. 8 players (88.9%) either agreed or strongly agreed wanting to know more about the knowledge taught. The knowledge improvement category for EGameFlow recorded a mean of 3.000.

Statement	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
The game increases my knowledge.	0 (0%)	1 (11.1%)	2 (22.2%)	4 (44.4%)	2 (22.2%)
I catch the basic ideas of the knowledge taught.	0 (0%)	0 (0%)	0 (0%)	7 (77.8%)	2 (22.2%)
I try to apply the knowledge in the game.	0 (0%)	0 (0%)	1 (11.1%)	6 (66.7%)	2 (22.2%)
The game motivates the player to integrate the knowledge taught.	0 (0%)	2 (22.2%)	2 (22.2%)	2 (22.2%)	3 (33.3%)
I want to know more about the knowledge taught.	0 (0%)	0 (0%)	1 (11.1%)	5 (55.6%)	3 (33.3%)
<b>Mean: 3.000</b>					
<i>Strongly Disagree (0), Disagree (1) Neutral (2), Agree (3), Strongly Agree (4)</i>					

Table 10.2: **EGameFlow Knowledge Improvement:** showing how many participants answered for each option. The mean is calculated using the scoring shown at the bottom of the table.

---

## Immersion

In Table 10.3 the questions from EGameFlow regarding immersion in the game are presented. 5 players (55.5%) either disagreed or strongly disagreed to becoming unaware of their surroundings while playing the game, and only 3 players (33.3%) agreed or strongly agreed. 4 players (44.4%) agreed or strongly agreed to temporarily forget worries of everyday life while playing the game, and the same percentage disagreed or strongly disagreed. 4 players (44.4%) were neutral about experiencing an altered sense of time. 4 players (44.4%) disagreed to becoming involved in the game. 6 players (66.6%) either disagreed or strongly disagreed to feeling emotionally involved in the game. Lastly, 4 players (44.4%) either disagreed or strongly disagreed to feeling viscerally involved in the game. The immersion category for EGameFlow recorded a mean of 1.778.

Statement	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
I become unaware of my surroundings while playing the game.	1 (11.1%)	4 (44.4%)	1 (11.1%)	2 (22.2%)	1 (11.1%)
I temporarily forget worries about everyday life while playing the game.	1 (11.1%)	3 (33.3%)	1 (11.1%)	2 (22.2%)	2 (22.2%)
I experience an altered sense of time.	1 (11.1%)	2 (22.2%)	4 (44.4%)	1 (11.1%)	1 (11.1%)
I can become involved in the game.	0 (0%)	4 (44.4%)	2 (22.2%)	1 (11.1%)	2 (22.2%)
I feel emotionally involved in the game.	2 (22.2%)	4 (44.4%)	1 (11.1%)	2 (22.2%)	0 (0%)
I feel viscerally involved in the game.	3 (33.3%)	1 (11.1%)	3 (33.3%)	2 (22.2%)	0 (0%)
<b>Mean: 1.778</b>					
<i>Strongly Disagree (0), Disagree (1) Neutral (2), Agree (3), Strongly Agree (4)</i>					

Table 10.3: **EGameFlow Immersion:** showing how many participants answered for each option. The mean is calculated using the scoring shown at the bottom of the table.

---

## Concentration

In Table 10.4 the questions from EGameFlow regarding concentration are presented. 8 players (88.9%) either agreed or strongly agreed to the game grabbing their attention. 5 players (55.5%) either agreed or strongly agreed to the game providing content that stimulated their attention. 8 players (88.9%) agreed to the gaming activities relating to the learning task. 5 players (55.6%) were neutral to there being no distraction from the task highlighted. 7 players (77.8%) either agreed or strongly agreed to generally being concentrated in the game. 5 players (55.5%) either agreed or strongly agreed to not being distracted from tasks that they should concentrate on while playing the game. 6 players (66.7%) either agreed or strongly agreed to not being burdened with tasks that seemed unrelated. 7 players (77.8%) either agreed or strongly agreed that the workload in the game was adequate. The concentration category for EGameFlow recorded a mean of 2.708.

Statement	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
The game grabs my attention.	0 (0%)	0 (0%)	1 (11.1%)	6 (66.7%)	2 (22.2%)
The game provides content that stimulates my attention.	0 (0%)	0 (0%)	4 (44.4%)	3 (33.3%)	2 (22.2%)
Most of the gaming activities are related to the learning task.	0 (0%)	0 (0%)	0 (0%)	8 (88.9%)	1 (11.1%)
No distraction from the task is highlighted.	0 (0%)	1 (11.1%)	5 (55.6%)	2 (22.2%)	1 (11.1%)
Generally speaking, I can remain concentrated in the game.	0 (0%)	2 (22.2%)	0 (0%)	6 (66.7%)	1 (11.1%)
I am not distracted from tasks that the player should concentrate on.	0 (0%)	3 (33.3%)	1 (11.1%)	4 (44.4%)	1 (11.1%)
I am not burdened with tasks that seem unrelated.	1 (11.1%)	1 (11.1%)	1 (11.1%)	5 (55.6%)	1 (11.1%)
Workload in the game is adequate.	0 (0%)	1 (11.1%)	1 (11.1%)	5 (55.6%)	2 (22.2%)
<b>Mean: 2.708</b>					
<i>Strongly Disagree (0), Disagree (1) Neutral (2), Agree (3), Strongly Agree (4)</i>					

Table 10.4: **EGameFlow Concentration:** showing how many participants answered for each option. The mean is calculated using the scoring shown at the bottom of the table.

---

## 10.4.2 Ad hoc Questions

In addition to the included questions from the EGameFlow standardized questionnaire presented in section 6.3, some personalized questions were added to get insights into how the educational aspects linked with the game and how the familiar gameplay to cooking games affected the players.

The game focuses largely on linking gameplay actions to educational elements. For this reason, the survey included two questions regarding this topic which can be seen in Table 10.5. The survey recorded that 7 (77.7%) of the players either strongly disagreed or disagreed with the educational elements of the game interrupting their enjoyment of the game. 9 (100%) of the players also recorded either agreeing or strongly agreeing that the educational contents aligned well with the overall context of the game.

Statement	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
The educational elements in the game interrupted me from my enjoyment of playing the game.	3 (33.3%)	4 (44.4%)	1 (11.1%)	1 (11.1%)	0 (0%)
The educational content within the game aligned well with the overall context and theme of the game.	0 (0%)	0 (0%)	0 (0%)	6 (66.7%)	3 (33.3%)

Table 10.5: Linking learning objectives with game objectives.

As the game borrows gameplay mechanics from the cooking game genre, the survey included some questions regarding familiarity with such cooking games and how they affected the player, which can be seen in Table 10.6 and Table 10.7. 8 (88.89%) of the players recorded that the game reminded them of cooking games they had played before.

Statement	Yes	No
The gameplay reminded me of cooking games I have played before.	8 (88.89%)	1 (11.1%)

Table 10.6: Familiar Gameplay (Yes/No).



---

5 players (55.5%) either agreed or strongly agreed that this familiarity helped them learn faster, while 2 players (22.2%) felt neutral about the statement, and 1 player (11.1%) disagreed with the statement. 5 players (55.5%) also recorded either agreeing or strongly agreeing to be able to focus on learning Git because of the similar gameplay, while the remaining 3 players (33.3%) felt neutral about the statement.

Statement	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
The similar gameplay helped me learn the game faster.	0 (0%)	1 (11.1%)	2 (22.2%)	3 (33.3%)	2 (22.2%)
The similar gameplay allowed me to focus on learning Git.	0 (0%)	0 (0%)	3 (33.3%)	4 (44.4%)	1 (11.1%)

Table 10.7: Familiar Gameplay and learning.

Even though the game had undergone extensive testing during Alpha and Beta phase of development, the survey still recorded 2 players (22.22%) unable to play the game properly due to bugs. Here one reported that they *"Could not earn ingame currency"*, and the other reported *"Probably not a bug, but it felt like the tutorial stage should have shown you at least how to complete one order of burger"*, which they elaborate made them unable to understand the game and thus progress in the game. The player who could not earn in-game currency also reported that the respective bugs reduced the game quality as can be seen in Table 10.8.

Statement	Yes	No
Did any bugs prevent you from playing?	22.22%)	6 (66.67%)
Did any bugs reduce the game quality?	1 (11.11%)	7 (77.78%)

Table 10.8: Bug reports.

The survey also recorded additional feedback if participants wanted to provide it. The feedback from this was mostly positive, but also included some constructive feedback to the game. For the positive responses some players reported: *"Overall, great game"*, *"I had so much fun!!"*, and *"Great concept"*. Another player also reported that the concept of restaurants worked well for learning Git, as the repetition of actions made a good way for learning.

Other players gave more constructive feedback, such as: wanting to increase the timer length for each day, wanting more in-depth description of tasks, and wanting an info button at all times. Another player said that *"I didn't*

---

*get very far because I knew all the stuff up to that point.*”, where they went on to explain that they would be more interested if there was more advanced Git concepts. This player also suggested that if more advanced concepts were included the game should tell the players at the start to help motivate them to progress to further levels. One player reported that they didn’t understand that the *”git add .”* command included the *”.”*, so they couldn’t get it to work. Lastly, a player reported that they *”Would like to see merging and detached head being included”*.

## 10.5 Interviews

After interpreting the data, we wanted to gain more feedback on certain topics of interest. To do this, interviews were planned to cover the three topics: immersion, experience with the experiment format, and insight in the difference in experience from playing the game and reading the documentation. The interviews were conducted individually with 2 participants, in Norwegian, and later translated to English. One of the interviewees had both read the documentation and played the game before participating in the interview; this interviewee also had zero experience with Git before participating in the experiment. The other interviewee had some prior experience with Git.

### Immersion

The interviewees were asked the question: *’Did you feel immersed whilst playing the game?’*. One interviewee reported: *”I felt like I managed to get quite into the game, but I think it would really helped with more help throughout the game.”*, where they went on to explain that this added help would especially be important if a player get stuck at some point during the gameplay. This interviewee also suggested that hints with easier terminology might have helped to get more immersed in the game. Another interviewee reported: *”I might not have gotten very immersed, but I felt I got quite into the game. It was motivating moving up through the levels.”* where they also elaborated on the fact that the game got quite repetitive after a while and it was no definitive ending which made the game a bit boring after playing for some time. This interviewee also pointed out that it might have helped if some specific goals were added to the game such as winning or achieving something and or preventing the restaurant from going bankrupt.

Furthermore, one of the interviewees was asked: *”What do you think about the story in the game in terms of immersion?”*. The interviewee responded that the story was okay, and that the current story felt more like a tutorial,

---

however the interviewee went on to say that this is what they expected when starting the game. This interviewee also went on to say: *"I feel like it's hard to create a good story with the goal of the game, which is learning"*.

Continuing, the interviewees were asked if a larger focus on story would help increase the immersion. Which one of the interviewees responded with: *"I feel like a story always increases immersion! Regardless what form of media or entertainment."* The other interviewee also reported that the game would most likely benefit from more focus on the story, and connecting the story to certain goals would have made the game more interesting and help with becoming more immersed.

## Experiment experience

The interviewees were asked the question: *'What did you think about the format of the experiment?'*

One interviewee responded that the questionnaire was a bit long, however it was noted that this wasn't a problem for the interviewee as it takes some time to get into new content anyways. The interviewee continued with: *"Splitting up [the experiment] into two parts might have helped, but I also feel this could lead to the opposite effect."*, where they elaborated that splitting the experiment up over a time frame of multiple weeks could be too long, causing people to drop out. Another interviewee also reported the experiment length to be acceptable, but a bit long. This interviewee also pointed out the possibility of splitting up the experiment to possibly be beneficial to see if the participants actually remember what is taught a few days later and that it would be nice to be able to play over a few days to possibly be able to remember the commands better.

## Game vs. traditional approach

The interviewee that had both played the game and read the document was asked the question: *'How did you experience the difference between learning through the game and learning through the document?'* The interviewee reported: *"Personally I learn better by observing physical examples on what I learn, and having the ability to learn in practice is a huge plus."*, where the interviewee clarified that the game made it easier to learn the commands when connecting it to something familiar such as fulfilling orders. The interviewee explained that the game made it easier to understand why the commands were used to perform different tasks.

# Chapter 11

## Discussion

In this chapter, we aim to discuss the thesis question: *"How effective is game-based learning for learning Git version control system compared to a traditional method?"*, using the results from chapter 10. This will be done through answering the four research questions:

- RQ1: What is the process for testing effectiveness of game-based learning? (see section 11.1)
- RQ2: Does game-based learning of Git provide a better understanding of basic concepts compared to traditional learning? (see section 11.2)
- RQ3: Does game-based learning provide a better learning experience compared to traditional methods for learning Git? (see section 11.3)
- RQ4: Does the game successfully integrate Git elements into its design? (see section 11.4)

which are covered in the next sections. The key findings that will be discussed for RQ1 revolve around what we learned regarding how to test the effectiveness of game-based learning. It was also found that a game-based approach for learning Git does not provide a statistically significant difference in learning gain compared to a traditional approach, which relates to RQ2. For RQ3 we found that the game group spent on average double the amount of time learning (29.11 min. vs. 14.125 min.), whilst also reporting greater enjoyment (2.75 vs. 2.125) and motivation (3.0 vs. 2.125) compared to the control group. Regarding RQ4, we found that the Git concepts are well integrated into the design of the game, with all players agreeing to the educational content aligning well with the game context and theme.

---

## 11.1 Testing effectiveness of game-based learning

This section will regard: *RQ1: "What is the process for testing effectiveness of game-based learning?"*

### 11.1.1 Alpha and beta testing was valuable

This research question was mostly answered through the literature study done in Part II, which resulted in the experiment structure that was used in this thesis (see section 9.1). In the literature study it was found that in order to test the effectiveness of a game-based learning approach it is important to do formative evaluation, through both alpha testing and beta testing to reduce the effect of usability issues on learning (see subsection 6.2.1). Our experience also agrees with this finding, as there was discovered 3 game breaking bugs in alpha testing, which went down to 1 bug report in beta testing. Additionally, the usability of the game increased based on the System Usability Scale (SUS) after alpha testing. There were however 2 bugs in the final experiment, but these were less severe than those discovered in alpha testing. Still, even though these last bugs were not discovered in the alpha and beta testing, the quality of the experiment would be greatly reduced if the bugs found in alpha and beta testing were not addressed before the main experiment.

### 11.1.2 Playtest method was lacking

We did find that some people reported difficulty understanding the game mechanics during the main experiment; which was not discovered or reported in the alpha and beta testing. This suggest that our approach to alpha and beta testing was lacking in some ways. We think that this might have been caused by only following the *playtest* method, since compared to an observational method it is harder to discover usability issues, as discussed in subsection 6.2.1. We especially think that using only the *playtest* method in beta testing may not have been the most effective approach for discovering issues related to the educational content. It is difficult to gain valuable feedback regarding difficulties with the educational content solely through a questionnaire. This is suggested by the fact that the beta testing showed high perceived learning, with no reports of difficulties understanding the game, whilst in the main experiment, difficulties understanding the game was one of the most reported issues by the participants. An improvement could therefore be to do either observational studies or interviews along side

---

the *playtest* method. This would introduce both qualitative and quantitative data on the game's usability and educational content, which could help uncover problems with the game that were not discovered by relying solely on the *playtest* method.

That being said, the *playtest* method was effective for discovering game breaking bugs; especially in the alpha testing with internal testers, which discovered both logical errors and usability errors in the game. We also found the method to be a time effective way of gathering feedback on the game, making it a viable option, especially when there are time constraints that prevent the inclusion of interviews and observational studies.

### 11.1.3 Gamma testing worked well

When it comes to gamma testing, the main experiment for testing the effectiveness of game-based learning, we found the structure outlined in section 9.1 to work well. Since the participants recorded a wide variation in previous Git experience, it was crucial to measure relative learning with a pre-test and post-test to account for the initial difference in skill between the participants. It was also important to account for potential differences in difficulty between the two tests, where a difference could cause skewed results in terms of measuring knowledge gain. The method of creating parallel versions of the pre-test and post-test appears to have been effective, as there is no statistically significant difference in the results between the pre-test and post-test, when controlling for the different versions,  $p=.977$  (see Figure 10.9, section 10.2). We would therefore recommend basing one version on the other, only making slight alterations in the questions content, which ensures similar difficulty, as is also recommended by the research in subsection 6.2.2.

We also found that ad hoc questionnaires are needed, despite them not being statistically valid. This agrees with the research from subsection 6.2.2, which stated that standardized questionnaires might not yield sufficient feedback for specific aspects of the game. If we had only relied on standardized questionnaires we would not have been able to answer game-specific questions, such as questions regarding game context and familiar gameplay, which were important aspects of our game design.

From subsection 6.2.2 it was found to be important to control for confounding variables between the experimental group and the control group. We found that utilizing an online survey, combined with software for randomly assigning participants to groups were useful tools to provide the same conditions for both groups, as all participants are introduced to the experiment in an identical manner. However, subsection 6.2.2 also mentions that the absence of an instructor is ideal to properly isolate the game, but this might

---

be unnatural in certain contexts where guidance is a natural part of the environment. We discovered that some people quit the game because they were stuck or couldn't understand what to do (see section 10.3). It can be argued that this is a problem with the game, especially with the tutorial and how the game introduces beginners to Git. However, it might also be the case that the people who quit due to a lack of understanding would have been able to continue if they received some guidance. This was also the case for one participant in the control group, who quit due to a lack of understanding. The exclusion of an instructor providing guidance might therefore have been unnatural in the context of learning Git, since in a natural environment, guidance could be provided to help people get started, especially for those with no experience to Git. However, we do believe that this is mainly a fault with how the game introduces Git to absolute beginners. Still, it might be useful too see how the two methods compare in a more natural environment.

#### **11.1.4 Experiment length was too long**

Instead of conducting the experiment in multiple phases where the pre-test, learning phase, and post-test could be sent out a few days apart, participants had to do the entire experiment in one sitting. This could have introduced many unknown factors affecting the quality of the results.

This was confirmed by the interviews, where some participants felt that the experiment length was too long. They also suggested that splitting up the experiment into multiple phases, with the pre-test, learning phase, and post-test sent out a few days apart, could have been more beneficial. One participant noted that this could help determine if participants could remember what they learned a few days later. However, there was also concern that such an approach might lead to people dropping out of the experiment. Overall, it seems that the length of the experiment was a point of discussion among the participants and may have impacted the quality of the results.

---

## 11.2 Differences in Learning Outcome

This section will regard *RQ2*: *"Does game-based learning of Git provide a better understanding of basic concepts compared to traditional learning?"*

### 11.2.1 Inconclusive increase in understanding

Using ANCOVA to control for differences in pre-test scores, the analysis show that there is not a statistically significant difference between the game group and the traditional method group for learning Git. However, the results indicate that the game-based approach performed at-least as well as the traditional method; where a 1 point higher mean on the post-test score was measured in the game group compared to the control group, although this difference was not statically significant. It is worth noting that a larger sample size would be beneficial to confirm this difference. Increasing the sample size could also lead to a more normally distributed data set, which was not observed in the post-test results. Since ANOCVA is robust to violations of the normality assumption (see subsection 2.2.1), the results from the initial test is still valid, though not ideal. Additionally, to address the non-normal distribution in the post-test results, another ANCOVA was conducted after removing an outlier. The adjustment resulted in the data conforming to a normal distribution, however, even with this adjustment, the results still indicated that the difference was not statistically significant.

One aspect to point out regarding the learning methods in the experiment, is that the participants were exclusively subjected to either the game-based approach or the traditional method. This was done to properly isolate the effects of each of the methods. However, in a more realistic context, a student learning Git would not follow such a one-dimensional approach. For instance, a student could attend a lecture introducing them to Git, and then receive some supplementary material to read, as well as engaging in practical exercises. An interesting question is therefore if using the game, not as a sole method for learning Git, but rather as supplementary material to enhance the learning process, would yield different results. It was discovered in section 10.3 that the people learning Git through the game were more motivated and spent more time learning, and thus providing the game as supplementary material could increase the likelihood of students spending additional time learning Git. This approach could be studied further.



---

## 11.3 Differences in Learning Experience

This section will regard *RQ3*: "*Does game-based learning provide a better learning experience compared to traditional methods?*".

While learning outcome is undoubtedly the primary concern when it comes to choosing a learning method, the learning experience itself, including factors such as motivation, enjoyment, and time spent learning, should also be taken into account, as sustainable learning requires engagement from the learner.

### 11.3.1 Increased time spent learning

From Figure 10.11 (see section 10.3), we found that participants in the game group spent approximately double the time learning compared to participants learning through the theory method on average. This difference was also found to be statistically significant, meaning that people playing the game were more likely to spend more time learning. However, this is not necessarily due to higher engagement in the game group compared to the traditional learning group. It might be the case that the game simply took longer to finish. This is supported by the fact that 40% of the traditional group stated that they quit due to being finished, whilst only 13% of the game group reported the same. The same is true for the fact that the traditional group reported quitting more often due to being satisfied with learning outcome compared to the game group.

As previously discussed, the length of the experiment was quite long and was performed in one sitting. This might have had an effect on the reason for participants quitting, raising the question on what difference it would make if the learning phase of the experiment was conducted over a given period of time. The results might have been different if the participants only needed to focus on the learning aspect, without the pressure of needing to complete the rest of the experiment. It would also be interesting too see if there would be a significant difference between how often the two groups would return to the learning phase if they were given a week time-span. This was also mentioned in the interview by one participant, who stated that they would be more inclined to play the game multiple times and learn the commands better if they had more time.

---

### 11.3.2 Increased learning motivation and enjoyment

We found evidence that the game group experienced more enjoyment and motivation compared to the traditional learning group. From Figure 10.10 (see section 10.3) we found that the game group on average agreed to statements regarding enjoyment, motivation, and learning perception, whilst the theory group were more neutral to the same statements. From the specialization project (see Francis & Thorvik (2023)) it was found that providing learning in the right context in educational games can provide higher motivation. Our results agrees with this, as most participants agreed with the educational context aligning with the context of the game, whilst also reporting statistically significant higher motivation ( $p=.045$ ) compared to the traditional group.

Even though the learning gains were not found to be significantly different between the two groups, the fact that motivation is higher in the game group has positive effects on memory and personal development (see Francis & Thorvik (2023)). So, in this sense, the game group has achieved some advantages compared to the traditional group.

The reason why enjoyment was not significantly higher for the game group might be due to shortcomings in the game design. It was discovered in Francis & Thorvik (2023) that appropriate feedback is important for enjoyment in games, and some players (see subsection 10.4.2) noted that they would like more in-depth description of task and an info button. This was also mentioned in the interviews, where it was stated that more hints with easier terminology would be beneficial. Improving these aspects of the game, would provide more feedback, and could help improve the enjoyment in the game.

---

## 11.4 Gameplay and learning integration

This section will regard *RQ4*: “Does the game successfully integrate Git elements into its design?”

In order to answer this question, we will look at the gameplay feedback presented in section 10.4 which revolved around concentration, immersion, knowledge improvement, educational context and familiar gameplay.

### 11.4.1 High level of EGameFlow knowledge improvement

Our findings revealed that participants, on average, agreed with the statements related to knowledge improvement in EGameFlow, as indicated by a mean score of 3.000 on a scale ranging from 0 to 4. This is a good result, and supports the game’s ability to integrate educational content in the gameplay in a positive way making players feel like they are learning.

In addition to these standardized questions, some custom questions regarding the players’ experience with the educational context and the familiar cooking game elements were also added to support the knowledge improvement questions in the EGameFlow model. Means were not calculated, as these questions are not standardized and were only meant to provide further insight into this category. Questions regarding how educational elements aligned with gameplay received highly positive results, indicating that the educational content neither interfered with enjoyment or the context and theme of the game. This indicates that the integration of Git within the game was successful. In addition, participants reported that their familiarity with popular cooking games aided them in terms of learning the game faster and helping them focus on learning Git. This agrees with the findings from the specialization project (see section 3.2), which emphasized the importances of familiar gameplay when designing educational games.

### 11.4.2 Medium to high level of EGameFlow concentration

In the concentration category of EGameFlow, we recorded a mean of 2.708 (see Table 10.4), indicating that participants generally agreed to the statements presented in this category. This is an acceptable result, and indicates that the game enabled players to stay concentrated during gameplay. One possible explanation for the high concentration level might be that the game has successfully integrated the learning objectives without interrupting the

---

flow, which was found to be important (see section 3.2). This is supported by most players agreeing to the learning objectives being integrated into the game, and that the gaming activities are related to the learning task.

Inspecting the individual question in the concentration category, it seems like the statements regarding distractions and unrelated tasks are the reason for some of the participants recording less concentration. This indicates that there could be some improvements to the game in this regard. We think this could relate to the numerous dynamic elements during gameplay such as the deadline for each day, the updating of orders, and the sequence of Git commands that must be performed in the correct order. All these elements increase the sense of urgency and demands high levels of concentration from the player. However, it can be difficult to stay on top of these task unless a sufficient skill level has been achieved by the player. As a result, the player can become overwhelmed and unable to focus on a single task. This is important to address because minimizing distractions increases concentration (see section 3.3).

Another impact on concentration is the challenge level of the game, which should not be too easy or too hard to keep the player in the zone (see section 3.3). This is supported by the feedback that was provided by players on why they stopped playing. Certain players reported that some elements were too difficult to understand, stating that the game was too complicated without any previous experience with Git, causing them to quit. On the other hand, some players stated that the challenge level was too low and found it too easy because they already had previous experience with Git. The issue appears to be that the game fails to match the skill level of the player, resulting in it being too difficult for some and too easy for others. It is therefore apparent that the game would benefit from including a more adaptive challenge level that would better adjust to the skill level of the player. One way to address this could be to allow the player to choose a difficulty mode at the start of the game based on their Git competence, and then having the the game adjust the difficulty accordingly.

### **11.4.3 Low to medium level of immersion**

Compared to the results for knowledge improvement and concentration, the immersion category for EGameFlow recorded a low mean of 1.778 (see Table 10.3), indicating that on average participants were mostly neutral or disagreed with statements regarding immersion in the game. This is a poor result, and indicates that the players are struggling to become immersed in the gameplay. Identifying the specific factors driving this result is difficult as there are no specific statements that stands out from each other within the category. One possible reason for the low immersion might be related

---

to storytelling and role-play in the game. It was found that a good narration is important for immersion, inspiring curiosity and facilitating deep involvement (see section 3.3). From Table 10.3 it can be seen that players disagree to being emotionally and viscerally involved in the game, which could be the result of a lack of immersive storytelling. The story in the game is integrated as a part of the tutorial, but the game could benefit from including story elements that are not a part of the tutorial, which we think would increase the immersion. From the specialization project (see Francis & Thorvik (2023)) it was found that in order for players to feel a part of the story, it is important that they know both what is happening and who the characters are. The game is lacking in this regard, as the character in the game lacks a clear identity. The game could benefit from creating a clearer identity for the player by introducing more role-play elements. Perhaps the player could have an icon which is customized as they progress, and then this player icon could interact with the story of the game, strengthening the player's identity within the game. It would also be useful to include more narrative surrounding the background of the player and the characters, to let the player know who they are and why things are happening.

The interviews conducted regarding immersion in the game also showed that the level of immersion was a problem within the game. One of the interviewees suggested that more help throughout the game and hints with easier terminology could have increased immersion. The repetitive nature of the game and the lack of specific goals such as winning or achieving something were also reported to have decreased immersion. The story in the game was reported to be okay, but all the interviewees agreed that a larger focus on story would have most likely helped with increasing their immersion within the game.

Another reason why players may lose their immersion while playing the game is that most of the Git knowledge is taught within the first day. This can be overwhelming for players who are new to the game, particularly if they have little or no experience with Git, making it hard to become immersed. To address this issue, limiting the number of required actions in the initial phase of the game might be a good idea. Instead, the game can gradually introduce new features and actions in different stages of the game to help players adjust and avoid overwhelming them.

---

## 11.5 Limitations of the study

This section will highlight the limitations of this study to help understand which parts of the study that is lacking and needs improving in order to get even more valuable results.

### 11.5.1 Samples

#### Small sample size

The research suffers from a limitation in terms of sample size. This issue is particularly pronounced when considering that the sample is divided into two distinct groups that employ different learning methods. Moreover, the sample is further subdivided into two subgroups with varying test orders in order to find flaws or inconsistencies with either of the two tests. These factors collectively contribute to a potential lack of statistical power.

#### Variations in Git experience

In addition to the limiting sample size, the participation pool exhibited diverse experience in Git. This was most likely the result of the large range of study fields and study years. The diversity in experience posed challenges in drawing statistically significant conclusions, as we don't know the effect that different levels of proficiency with Git will have on the learning efficiency for each of the learning methods.

Looking at our target group, which includes newcomers or beginners at Git, participants that recorded higher proficiency with Git might have an extra advantage. They already have previous experience that serve as a foundation to connect the new information to. On the contrary, someone without any experience with Git might be overwhelmed with all the new information and as a result learn less. Both groups falls within our target group, but might be in need of different things. The survey recorded only 3 people without any experience in Git, one of which played the game.

We also encountered participants falling outside of our target group with too much Git experience, where they ended up with less or no room for improvement. One participant scored a full score on the pre-tests, meaning there was no room for improvement.

---

## Self-selection bias

Another limitation of this study is the potential for self-selection bias, which may have been introduced as a result of the recruitment method used. Participants were recruited primarily through relevant Facebook groups, where members were asked to participate voluntarily. This recruitment method may have attracted participants who were particularly interested or motivated to learn and improve their skills in Git, potentially skewing the results towards a more positive outcome for both learning methods.

This bias is supported by the fact that a large portion of participants expressed interest in learning and becoming better at Git. The results might have been different if an alternative recruitment method had been employed, such as recruiting a larger and more diverse sample, e.g., through a professor tasking their students with completing the survey. This could contribute to a more varied and representative participant pool, potentially leading to more realistic and significant differences between the game and theory methods.

### 11.5.2 Testing method

In terms of the method of testing, there are some potential limitations that might have affected both the results and the lack of participators in this research. First and foremost, the survey is quite long and demanding. The length of the survey is around 45 minutes and includes two tests in addition to a learning phase. These steps require a lot of concentration and patience, and those who completed the surveys containing theory as the learning method spent the entire survey either reading or answering questions. The reasoning behind the long survey was to eliminate the possibility for participants of learning anything outside the provided learning methods provided in the survey. In addition, we feared participants would drop out in the middle of the process if the experiment lasted for multiple days or weeks, which was also confirmed as a concern by participants in the interviews. Instead of sending the pre-test, learning phase, and post-test separately after a given time, participants would need to complete the entire survey in one sitting, which can cause fatigue.

Another limitation of performing the survey in one sitting is how effective the different methods are for long-term learning effects. As the pre-test, learning, and post-test are completed in succession, the short-term effect is tested, while the results may be completely different if the post-test was performed later.

---

### 11.5.3 Digital survey

The experiment was conducted digitally without any supervision from anyone, meaning most of the results are based on the recorded answers in the survey which can become one-dimensional and limiting in many ways.

#### **Potential for cheating**

Both the pre-test and post-test were conducted on the participants' computers without any software or environment for disabling participants from looking up answers to the tests on the web or moving back to the provided learning methods. The survey explicitly mentions and encourages the participants to restrain themselves from cheating, but this information might have been overlooked or ignored.

#### **Inaccurate answers**

Another limitation with survey was the fact that some of the questions depended on the participants' own perception of experience in, for example, experience with Git, code editors, gaming, etc, in addition to recording the time they spent learning. The accuracy of the results might vary a lot depending on the person.

#### **Environmental variables**

With digital participation, each participant's environment for conducting the survey can vary and affect the results. One of the participants recorded needing to end the learning phase quicker due to an upcoming meeting. Some participants might be in an environment where they get interrupted while others might be in a stable and quiet environment which could differentiate in concentration levels and results.



# Part V

## Conclusion and Further Work

The primary objective of this thesis was to continue developing a game from our specialization project (see Francis & Thorvik (2023)), and evaluate the effectiveness the game as a means of teaching Git compared to a more traditional approach. First, we conducted alpha and beta testing to identify any critical bugs or usability issues within the game, utilizing 5 and 4 participants respectively. Following this, the main experiment (gamma testing) was conducted to evaluate the differences in learning gain and motivation between learning through the game or through the traditional approach. We created a pre-test and post-test to compare relative learning between an experimental group (consisting of 10 participants who played our game) and a control group (consisting of 9 participants who learned through a theoretical document covering the same content presented in the game). We removed two outlier results from our data before conducting a statistical analysis to identify significant findings.

This part serves to bring the thesis together by presenting a concise and comprehensive conclusion that summarizes the main findings in conjunction with our research questions. We will also explore the implications of our research and provide suggestions for future work. Specifically, we will identify potential avenues for further research in the field and explain how these may build upon our current findings.

## Chapter 12

# Conclusion

In this chapter we will use the research questions that was introduced in the beginning of this thesis to guide and conclude our findings in this study. We will also discuss the implications of our findings for game-based learning and highlight strengths and weaknesses of the study. The four research questions for the thesis were:

**RQ1:** *What is the process for testing effectiveness of game-based learning?*

**RQ2:** *Does game-based learning provide a better learning experience compared to traditional methods for learning Git?*

**RQ3:** *Does game-based learning of Git provide a better understanding of basic concepts compared to traditional learning?*

**RQ4:** *Does the game successfully integrate Git elements into its design?*

---

## 12.1 RQ1 What is the process for testing effectiveness of game-based learning?

Based on the research on testing the effectiveness of game-based learning conducted in chapter 6, we found the need to do both formative evaluation, through alpha and beta testing, and summative evaluation, through gamma testing. From our experience with this testing process, we found that alpha and beta testing methods were highly effective in identifying and fixing usability issues before assessing the game's effectiveness for learning Git. The progressive approach we followed for testing allowed us to address larger, general issues before the final experiment. As a result, the final experiment was mostly unobstructed by bugs and usability issues, allowing it to focus on assessing the effects of the game on learning and motivation. However, we found that following the *playtest* method, which involved no observation or guidance, had some limitation, especially for feedback regarding educational content within the game. We would therefore recommend incorporating observational studies or interviews along side the *playtest* method during beta testing, to improve feedback regarding the educational content of a game.

To test the learning effectiveness of the game, we followed a testing structure based on the research in subsection 6.2.2 (see also section 9.1), which consisted of (1) Identify the participant, (2) Pre-test, (3) Learning git, (4) Post-test, (5) Survey. We found it crucial to control for the variation in participants' previous experience with Git by using both pre-test and post-test, as the participants reported a variety of previous Git experience. We agree that parallel versions of the pre-test and post-test were needed to control for any inconsistencies, and we found no significant effect was when controlling for our test versions, which strengthens the analysis. Standardized questions from the EGameFlow model was included to gather feedback on the game, still we agree with the findings in subsection 6.2.2 that ad hoc questions are needed for game-specific feedback. However, due to the extensive amount of data collected, the experiment ended up being quite long, which might have affected the results.

In summary, our experience agrees with the findings in subsection 6.2.2, that alpha and beta testing methods are effective in identifying usability issues. However, we found that incorporating observational studies and interviews alongside the *playtest* method should be considered for beta testing. Additionally, our experience agrees with the findings in subsection 6.2.2, stating a need for controlling for existing difference with a pre-test, post-test design.

---

## **12.2 RQ2 Does game-based learning of Git provide a better understanding of basic concepts compared to traditional learning?**

Based on the analysis conducted, it can be concluded that the game-based approach for learning Git did not provide a statistically significant improvement in understanding of basic concepts compared to the traditional learning method, when controlling for pre-test scores. However, the game-based approach performed at-least as well as the traditional method, with a slightly higher mean in the post-test score for the game group. Although a larger sample size would be beneficial to confirm this difference, the results from the initial test are still valid.

It is important to note that the experiment only subjected participants to either the game-based approach or the traditional method, which is not a realistic approach in a classroom setting. Using the game as supplementary material to enhance the learning process could be a more effective approach. It was also found that participants who learned Git through the game were more motivated and spent more time learning, indicating the potential usefulness of incorporating game-based learning in the classroom. Further research could be conducted to explore this approach in greater detail.

## **12.3 RQ3: Does game-based learning provide a better learning experience compared to traditional methods for learning Git?**

The results of the study indicate that game-based learning method may provide a better learning experience compared to traditional methods for learning Git. Specifically, participants in the game group spent approximately double the time learning compared to participants in the traditional learning group. This difference was found to be statistically significant. Additionally, the game group experienced higher levels of motivation and enjoyment compared to the traditional learning group, where motivation proved to be statistically significant ( $p=.045$ ) while enjoyment was not ( $p=.080$ ).

Overall, while the study did not find significant differences in learning gains between the game and traditional learning groups, the results suggest that

---

game-based learning can provide a more enjoyable and motivating learning experience. Further research could explore how game-based learning could be used as supplementary material to enhance the traditional learning process, and how different game designs could impact learning gains and motivation.

## 12.4 RQ4 Does the game successfully integrate Git elements into its design?

The findings suggest that the game has successfully integrated educational content in the gameplay in a positive way. The way Git has been integrated within the game was also reported as successful, which supports the claims from Francis & Thorvik (2023) stating the need to integrate learning content into the gameplay elements. Furthermore, familiarity with popular cooking games was reported to be useful in terms of learning the game faster and focusing on learning Git, which agrees with Francis & Thorvik (2023). The concentration of players during gameplay was moderately high, indicating that the game enables players to stay concentrated during gameplay. This might be due to how the game has successfully integrated the learning objectives without interrupting the flow. However, there are room for improvements, especially concerning distraction in the game, to increase concentration. For the immersion category, however, the result were poor. This indicates that players were struggling to become immersed in the gameplay. One possible reason for losing immersion while playing the game is that the game requires the player to perform the same actions repeatedly every day, which can be overwhelming for players who are new to the game. Additionally, the game can be improved when it comes to storytelling, role-playing elements, and the inclusion of an overarching goal connected to the story.

## 12.5 Overall

To answer the research goal: *"How effective is game-based learning for learning Git version control system compared to a traditional method?"*, we found that difference in learning gain between the game-based approach and traditional approach to be insignificant. Though the game-based approach did score 1 point higher on average, when controlling for pre-test scores. Regardless, the game-based approach provided statistically higher motivation and time-spent learning compared to the traditional approach, which can advantageous for personal development and memory (see Francis & Thor-

---

vik (2023)). Additionally, we agree with research from subsection 6.2.1 that alpha and beta testing were needed to increase the quality of the gamma testing, as it reduced the effect of bugs and usability issues from construing the results. However, an improvement might be to use observational studies and interviews in beta testing. Lastly, the game prototype was found to successfully integrate Git concepts into the game's context, and familiar gameplay was reported as useful for learning. Still, the game needs improvement regarding immersion, which could be increased by adding more story-telling, a clearer overarching goal and reducing the complexity of tasks.

## Chapter 13

# Further Work

In this chapter we will provide suggestions and potential avenues for further work within the field.

---

This thesis focused on testing the effectiveness of game-based learning for learning Git version control system compared to a traditional method. We have attempted to follow guidelines from the literature regarding the process for testing, as well as which factors to consider when creating a game-based learning application. From this we gained valuable experience in designing an educational game and testing its effectiveness, at the same time there are certain elements that were discovered and learned throughout the process, which we think could be useful to study further.

First and foremost, the experiment in this thesis involved a small sample size, with people of varying Git competence level. To further confirm the insight gained in this thesis it would be beneficial to redo the experiment on a larger scale with more participants.

Our thesis tested the game in an experimental environment, which might not represent how the average student would learn Git. A point to explore further is therefore if the game could be used in a more natural environment, for example as supplementary material to how Git is usually taught to students. This could be done by introducing the game to some students as they learn Git through a course, and compare the result with a control group which does not have access to the game. It would be interesting to both study if there are differences in learning gain using this method, and also if there is a difference in motivation and time spent learning in a more natural environment.

Regarding the process for testing a game-based learning application, it is recommended for further work to test if introducing observational studies and interviews, along side the *playtest* method, would aid in discovering issues with educational content during development. Additionally, it would be beneficial to split the experiment over a longer time frame. This could be done by first doing a pre-test, then give the participant a week for learning using either the game or a traditional approach, followed by a post-test. The question to answer could be if given a longer time-frame, would the game group spend even more time learning, and would this increase the difference in learning gain and motivation between the two groups.

Lastly, the game developed in this thesis is still a prototype and not a finished product. It would therefore be interesting to develop the prototype further to answer the question if improving storytelling and role-play elements would increase the immersion in the game. It could also be useful to test if creating a difficulty mode that is chosen at the start of the game could better match the skill level to the player, allowing both beginners and more experienced Git users to be more engaged in the game.



# Bibliography

- All, A., Castellar, E. P. N. & Looy, J. V. (2014), 'Measuring effectiveness in digital game-based learning: A methodological review', *International Journal of Serious Games* **1**.  
**URL:** [https://www.researchgate.net/publication/275889436\\_Measuring\\_Effectiveness\\_in\\_Digital\\_Game-Based\\_Learning\\_A\\_Methodological\\_Review](https://www.researchgate.net/publication/275889436_Measuring_Effectiveness_in_Digital_Game-Based_Learning_A_Methodological_Review)
- All, A., Castellar, E. P. N. & Looy, J. V. (2016), 'Assessing the effectiveness of digital game-based learning: Best practices', *Computers & Education* **92-93**, 90–103.  
**URL:** <https://www.sciencedirect.com/science/article/pii/S0360131515300567>
- Basili, V. R., Caldiera, G. & Dieter, R. H. (1994), 'The goal question metric approach', *Encyclopedia of software engineering* pp. 528–532.
- Bonate, P. L. (2000), *Analysis of Pretest-Posttest Designs*, Chapman & Hall/CRC, chapter 5.
- Briony J. Oates, Marie Griffiths, R. M. (2022), *Researching Information Systems and Computing*, SAGE Publications.
- Brooke, J. (1996), 'Sus-a quick and dirty usability scale', *Usability Evaluation in Industry*, **189**, 4-7 .
- Davis, J., Steury, K. & Pagulayan, R. (2005), 'A survey method for assessing perceptions of a game: The consumer playtest in game design', *Game Studies* **5**.
- de Carvalho, C. V. (2012), Is game-based learning suitable for engineering education?, in 'Proceedings of the 2012 IEEE Global Engineering Education Conference (EDUCON)', pp. 1–8.
- Francis, S. & Thorvik, F. (2023), 'Game of git'.  
**URL:** <https://frithjofthorvik.github.io/git-cooking/docs/Specialization%20Project.pdf>

- 
- Fu, F.-L., Su, R.-C. & Yu, S.-C. (2009), 'Egameflow: A scale to measure learners' enjoyment of e-learning games', *Computers & Education* **52**(1), 101–112.  
**URL:** <https://www.sciencedirect.com/science/article/pii/S0360131508001024>
- Gerald, B. (2018), 'A brief review of independent, dependent and one sample t-test', *International Journal of Applied Mathematics and Theoretical Physics* **4**, 50.
- Gold, S. C. & Wolfe, J. (2012), 'The validity and effectiveness of a business game beta test', *Simulation & Gaming* **43**(4), 481–505.  
**URL:** <https://doi.org/10.1177/1046878111431868>
- Gris, G. & Bengtson, C. (2021), 'Assessment measures in game-based learning research: A systematic review', *International Journal of Serious Games* **8**.  
**URL:** <https://journal.seriousgamessociety.org/index.php/IJSG/article/view/383/409>
- Guay, F., Vallerand, R. J. & Blanchard, C. (2000), 'On the assessment of situational intrinsic and extrinsic motivation: The situational motivation scale (sims)', *Motivation and Emotion* **24**.
- IJsselsteijn, W., de Kort, Y. & Poels, K. (2013), 'The game experience questionnaire'.
- Jennett, C., Cox, A. L., Cairns, P., Dhoparee, S., Epps, A., Tijs, T. & Walton, A. (2008), 'Measuring and defining the experience of immersion in games', *International Journal of Human-Computer Studies* **66**(9), 641–661.  
**URL:** <https://www.sciencedirect.com/science/article/pii/S1071581908000499>
- Kniberg, H. (2015), *SCRUM AND XP FROM THE TRENCHES - 2nd edition*, C4Media.
- MacFarland, T. W. & Yates, J. M. (2016), *Mann-Whitney U Test*, Springer International Publishing, Cham, pp. 103–132.  
**URL:** [https://doi.org/10.1007/978-3-319-30634-6\\_4](https://doi.org/10.1007/978-3-319-30634-6_4)
- Moreno-Ger, P., Torrente, J., Hsieh, Y. G. & Lester, W. T. (2012), 'Usability testing for serious games: Making informed design decisions with user data', *Advances in Human-Computer Interaction* **2012**.  
**URL:** <https://dl.acm.org/doi/pdf/10.1155/2012/369637>
-

- 
- Nachar, N. (2008), ‘The mann-whitney u: A test for assessing whether two independent samples come from the same distribution’, *Tutorials in Quantitative Methods for Psychology* **4**.
- Nielsen, J., Turner, C. W. & Lewis, J. R. (2006), ‘Determining usability test sample size’, *International Encyclopedia of Ergonomics and Human Factors*, **3**.
- Olsen, T., Procci, K. & Bowers, C. (2011), Serious games usability testing: How to ensure proper usability, playability, and effectiveness, in A. Marcus, ed., ‘Design, User Experience, and Usability. Theory, Methods, Tools and Practice’, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 625–634.
- Olszewski, A. E. & Wolbrink, T. A. (2017), ‘Serious gaming in medical education: A proposed structured framework for game development’, *Simulation in Healthcare: The Journal of the Society for Simulation in Healthcare* **12**.  
**URL:** [https://journals.lww.com/simulationinhealthcare/FullText/2017/08000/Serious\\_Gaming\\_in\\_Medical\\_Education\\_A\\_Proposed.6.aspx](https://journals.lww.com/simulationinhealthcare/FullText/2017/08000/Serious_Gaming_in_Medical_Education_A_Proposed.6.aspx)
- Rutherford, A. (2011), *ANOVA AND ANCOVA A GLM Approach*, John Wiley & Sons, Inc.
- Sommerville, I. (2016), *Software Engineering - 10th GLOBAL Edition*, Pearson.
- Sweetser, P. & Wyeth, P. (2005), ‘Gameflow: A model for evaluating player enjoyment in games’, *Comput. Entertain.* **3**(3), 3.  
**URL:** <https://doi.org/10.1145/1077246.1077253>
- Tahir, R. & Wang, A. I. (2019), ‘Codifying game-based learning: Development and application of league framework for learning games’, *The Electronic Journal of e-Learning* **18**, 69–87. available online at [www.ejel.org](http://www.ejel.org).
- Yáñez-Gómez, R., Cascado-Caballero, D. & Sevillano, J.-L. (2017), ‘Academic methods for usability evaluation of serious games: a systematic review’, *Multimedia Tools and Applications* **76**.  
**URL:** <https://dl.acm.org/doi/pdf/10.1155/2012/369637>

# Appendix

---

## A1 NSD Assessment



[Notification form](#) / [Game of Git](#) / Assessment

# Assessment of processing of personal data

**Reference number**

744740

**Assessment type**

Automatic ⓘ

**Date**

09.03.2023

**Project title**

Game of Git

**Data controller (institution responsible for the project)**

Norges teknisk-naturvitenskapelige universitet / Fakultet for informasjonsteknologi og elektroteknikk (IE) / Institutt for datateknologi og informatikk

**Project leader**

George Adrian Stoica

**Student**

Steven Magnus Emil Berglund Francis

**Project period**

09.12.2022 - 30.06.2023

**Categories of personal data**

General

**Legal basis**

Consent (General Data Protection Regulation art. 6 nr. 1 a)

The processing of personal data is lawful, so long as it is carried out as stated in the notification form. The legal basis is valid until 30.06.2023.

[Notification Form](#) ↗

**Basis for automatic assessment**

The notification form has received an automatic assessment. This means that the assessment has been automatically generated based on the information registered in the notification form. Only processing of personal data with low risk for data subjects receive an automatic assessment. Key criteria are:

- Data subjects are over the age of 15
- Processing does not include special categories of personal data;
  - Racial or ethnic origin
  - Political, religious or philosophical beliefs
  - Trade union membership
  - Genetic data
  - Biometric data to uniquely identify an individual
  - Health data
  - Sex life or sexual orientation
- Processing does not include personal data about criminal convictions and offences
- Personal data shall not be processed outside the EU/EEA, and no one located outside the EU/EEA shall have access to the personal data
- Data subjects will receive information in advance about the processing of their personal data.

**Information provided to data subjects (samples) must include**

- The identity and contact details of the data controller
- Contact details of the data protection officer (if relevant)
- The purpose for processing personal data
- The scientific purpose of the project
- The legal basis for processing personal data
- What type of personal data will be processed and how it will be collected, or from where it will be obtained
- Who will have access to the personal data (categories of recipients)

- How long the personal data will be processed
- The right to withdraw consent and other rights

We recommend using our [template for the information letter](#).

**Information security**

You must process the personal data in accordance with the storage guide and information security guidelines of the data controller. The institution is responsible for ensuring that the conditions of Article 5(1)(d) accuracy and 5(1)(f) integrity and confidentiality, as well as Article 32 security, are met.

---

## **A2 Alpha testing - Questionnaire and results**



# Game of Git - Alpha

Oppdatert: 23. mars 2023 kl. 9:31

---

## Consent

### Participation is voluntary

Participation in the project is voluntary. If you chose to participate, you can withdraw your consent at any time without giving a reason. All information about you will then be made anonymous. There will be no negative consequences for you if you choose not to participate or later decide to withdraw.

### Your personal privacy – how we will store and use your personal data

We will only use your personal data for the purpose(s) specified here and we will process your personal data in accordance with data protection legislation (the GDPR). Your responses in the survey will be kept anonymous.

### What will happen to your personal data at the end of the research project?

The project is scheduled to end on 05.06.2023, at which point all personal information collected during the project will be securely and permanently deleted.

### Your rights

So long as you can be identified in the collected data, you have the right to:

- access the personal data that is being processed about you
- request that your personal data is deleted
- request that incorrect personal data about you is corrected/rectified
- receive a copy of your personal data (data portability), and
- send a complaint to the Norwegian Data Protection Authority regarding the processing of your personal data

### What gives us the right to process your personal data?

We will process your personal data based on your consent.

Based on an agreement with NTNU, The Data Protection Services of Sikt – Norwegian Agency for Shared Services in Education and Research has assessed that the processing of personal data in this project meets requirements in data protection legislation.

### Where can I find out more?

If you have questions about the project, or want to exercise your rights, contact:

**Steven Magnus Emil Berglund Francis:** smfranci@stud.ntnu.no

**Frithjof Thorvik:** frithjot@stud.ntnu.no

If you have questions about how data protection has been assessed in this project by Sikt, contact:

**email:** personverntjenester@sikt.no

**telephone:** +47 73 98 40 40

Yours sincerely,

Steven Magnus Emil Berglund Francis & Frithjof Thorvik

## I give consent to participate in the online survey, and for my data to be processed until the end of the project.

Antall svar: 5

Svar	Antall	% av svar	
No	0	0%	0%
Yes	5	100%	100%

We regret that you have decided to opt out of the survey. Please close this window to complete the opt-out process. If you choose not to submit this questionnaire, none of your data will be collected or used for the purposes of this survey.

## Misc questions

## Do you have any experience using Git?

Antall svar: 5

Svar	Antall	% av svar	
No	0	0%	0%
Yes	5	100%	100%

## How would you rate your competence with Git?

Antall svar: 5

Snitt: 3.80

Median: 4

Svar	Antall	% av svar	
5	0	0%	0%
4	4	80%	80%
3	1	20%	20%
2	0	0%	0%
1	0	0%	0%

## I am an experienced gamer

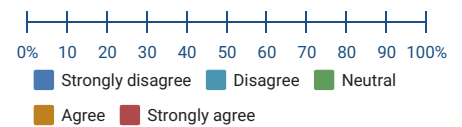
Antall svar: 5

Svar	Antall	% av svar	
Strongly agree	0	0%	0%
Agree	2	40%	40%
Neutral	2	40%	40%
Disagree	1	20%	20%
Strongly disagree	0	0%	0%

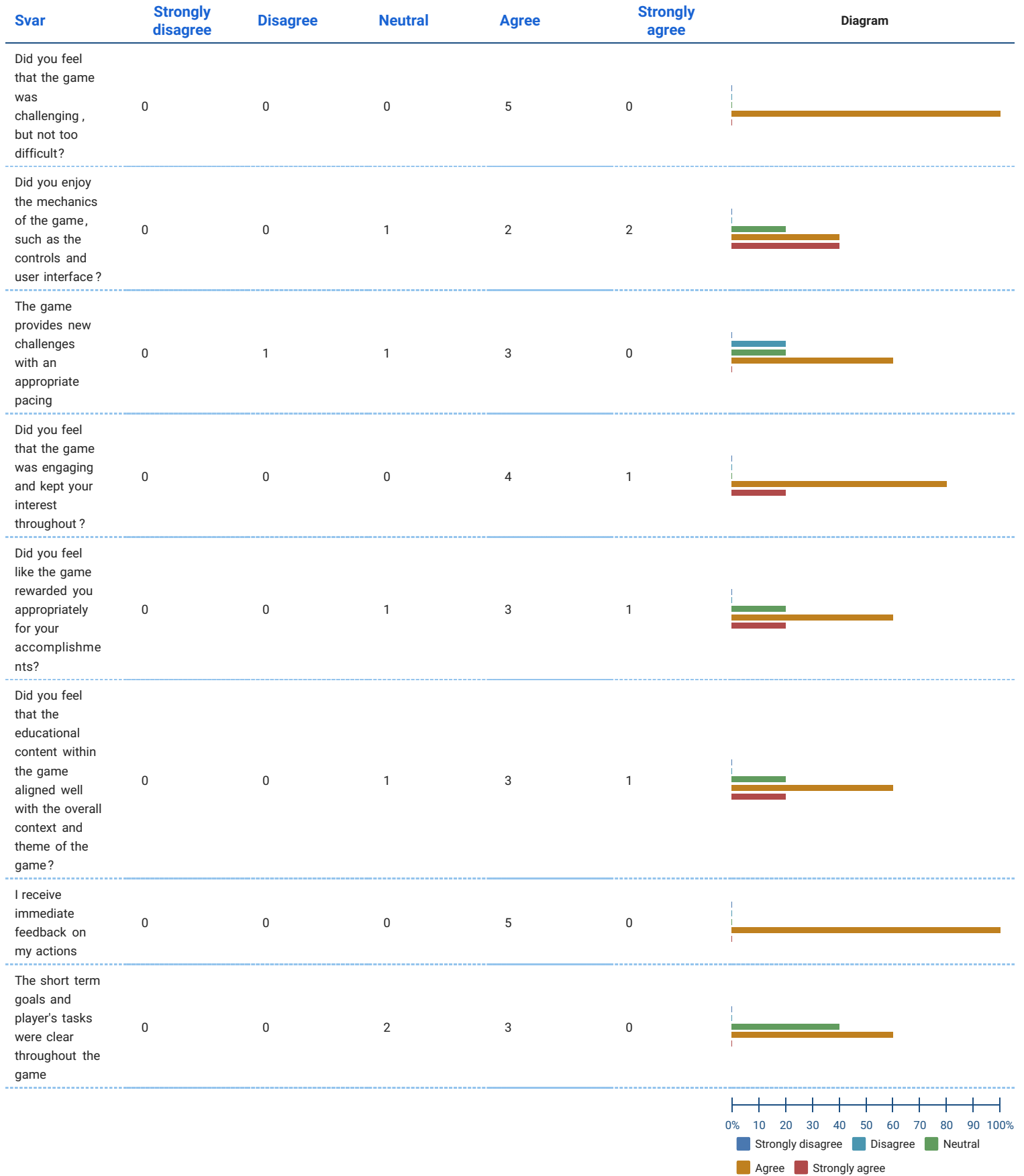
## Game Usability and Gameplay

# System Usability Scale

Svar	Strongly disagree	Disagree	Neutral	Agree	Strongly agree	Diagram
I think that I would like to use this game frequently	0	0	1	4	0	
I found the game unnecessarily complex	1	2	2	0	0	
I thought the game was easy to use	0	0	1	4	0	
I think that I would need the support of a technical person to be able to play this game	2	3	0	0	0	
I found the various functions in the game were well integrated	0	0	0	4	1	
I thought there was too much inconsistency in the game	2	3	0	0	0	
I would imagine that most people would learn to play this game very quickly	0	0	0	4	1	
I found the game very cumbersome to use	1	2	1	1	0	
I felt very confident playing the game	0	0	1	3	1	
I needed to learn a lot of things before I could get going with the game	0	4	1	0	0	





# Gameplay



# Bugs & Feedback

## Did any bugs prevent you from playing?

Antall svar: 5



Svar	Antall	% av svar	
No	2	40%	 40%
Yes	3	60%	 60%

## Please provide a description of the bugs

- When buying a git command upgrade (git branch), it is not possible to exit that screen without going through the tutorial screen.
- I gave the item name Burger. When I later wanted to do git add Burger, nothing happened and I could not go further in the game. Maybe I should have only used git add ., but as I have some experience with git from before then I thought I could use all comands.
- Some minor error spelling. When hover the mouse over the customer bubbles (C1, C2, ...), a text box displayed "root xx", ...

## Did any bugs reduce the game quality?

Antall svar: 5

Svar	Antall	% av svar	
No	4	80%	 80%
Yes	1	20%	 20%

## Please provide a description of the bugs

- The example text for a commit message, e.g., "Patricia's order," is not accepted. This could be very confusing for someone without experience with git, and it is a potential way to get stuck in the game. There should be an example of naming the files for the "orders" already in the first round so that someone without experience can more easily understand that step. Some of the example images are difficult to read since the images are very stretched out. There are no hints when switching between restaurants, and it is impossible to complete all three restaurants in one workday with the current time constraints.

## Do you have any additional feedback that you would like to share?

---

- Alt i alt veldig imponerende! Kun små-ting som kan være bedre men ikke nødvendigvis har så stor effekt for brukeropplavelsen. Litt småting de første dagene når alt var nytt, men kom godt inn i det etterhvert så det var bra! Tror tutorials kan være enda mer detaljerte, f.eks, hvordan bruke de forskjellige delene av "Work screen" og ikke bare hva de er. Slet litt med å forstå at man måtte lage en ny fil for å starte på bestillingen, og at man så måtte lage ny fil for ny del av bestillingen. Ville vært fint om det at man kan se tutorials igjen ved å trykke på knappen i venstre hjørne kommer ganske tidlig i spillet. Kom litt brått på nye tutorials mens man spilte når de plutselig dukka opp på skjermen. Mulig å illustrere at man snart ikke har tid igjen på en litt tydeligere måte? Hatt problemer i et prosjekt der flere samarbeider at det blir problemer om man bruker git add . fordi "skjulte" filer kan lure seg med fordi absolutt alt blir adda, og man blander da inn ting i committen som kan skape problemer om noen andre puller den. Kanskje ikke så viktig på nybegynner-nivå, men evt. noe å tenke på.

---

- I really enjoyed the game, and I think this could be a very engaging and fun way to learn git! You found a good analogy between restaurants and git, which was easy to understand and follow. The bite-sized tutorials were nice, and I didn't feel overwhelmed with information when playing. On day 1, the pacing of the tutorials in between the gameplay is a bit off. I would prefer if there was a bit more time between hitting enter/completing a task before the tutorial chef shows up, just so I have a chance to see what I have done before being presented with the next step. It's good that the help button is easily available, and I like that you separate tutorials, the command list, and git concepts into different categories so that the player can "how technical" they want the information to be. I think having an additional motivation to maximise "profit" could be valuable and more relatable for more players. A suggestion is "to profit and help our customers" or "to grow our restaurants and customer satisfaction". The shop is a bit too expensive, there should be some small upgrades available already after day 1 or 2, maybe a freebie like an ingredient that is needed for the next round to get the player started with upgrades. Overall, a really good game. I also liked that it remembered my progress so I could continue later. I had fun!

---

- In the intro at first page then I am informed that I can go back with backslash, but there is nothing to go back at. This should appear as an option from page two

---

- Til tider bevegde teksten seg litt treigt, og det ble ekstra irriterende når man ikke klarte å "spole". Alternativet var å ikke lese hele teksten eller vente til at hele teksten ble vist. Burde være mulig å adde enkeltfiler også, og ikke bare "git add ." Skjønnte heller ikke hva C1, C2, ... var på bunnen av skjermen, spesielt når (hjelp)tekstboksen med dukket når man hovret musen over sirklene var. Angående terminalen burde man kunne trykke på hele bredden. Litt kjipt å miste tekstpekeren hvis man bommer i terminalen. Bra at man kan bytte branch, men irriterende at man da må memorisere hva de heter før man evt får mulighet til å kjøre "git branch". ELLERS VELDIG KULT SPILL!

---

- Teksten var litt tider litt treg, og når man da trykket [Enter] to ganger så mistet du teksten. Burde heller vært sånn at den fullførte setningen. Itillegg var det litt irriterende at man ikke kunne trykke overalt innenfor terminalboksen for å begynne skrivingen. Synes også at det var litt vanskelig å skjønne alt på første dagen, kanskje en idé å legge til litt piler så man får litt hjelp til hvor alt er første dagen? Skjønnte heller ikke at det var ulik vanskelighetsgrad på de ulike restaurantene før dag 2 hehe. Syntes også "git add"-delen var litt forvirrende fordi jeg trodde at man kunne adde enkelte ordre, men man skal skrive "git add ." Så det var litt nedtur å tjene null penger:( Og så burde det komme enda tydeligere at det er en fil per order/bestilling. Og også skrive at ting er case-sensitivt. Beklager for veldig mye pirk fra meg, men dere skal vite at spillet var skikkelig morsom og gøy (utenom den litt lange introen som man må gjennom hver gang man starter spillet...). OOOOOG kanskje ha consent-delen før man svarer på masse spm, i tilfelle man er uendig ;D

---

---

## A3 Beta testing - Questionnaire and results

# Game of Git - Beta

Oppdatert: 23. mars 2023 kl. 9:41

## Consent

### Participation is voluntary

Participation in the project is voluntary. If you chose to participate, you can withdraw your consent at any time without giving a reason. All information about you will then be made anonymous. There will be no negative consequences for you if you choose not to participate or later decide to withdraw.

### Your personal privacy – how we will store and use your personal data

We will only use your personal data for the purpose(s) specified here and we will process your personal data in accordance with data protection legislation (the GDPR).

### What will happen to your personal data at the end of the research project?

The project is scheduled to end on 05.06.2023, at which point all personal information collected during the project will be securely and permanently deleted.

### Your rights

So long as you can be identified in the collected data, you have the right to:

- access the personal data that is being processed about you
- request that your personal data is deleted
- request that incorrect personal data about you is corrected/rectified
- receive a copy of your personal data (data portability), and
- send a complaint to the Norwegian Data Protection Authority regarding the processing of your personal data

### What gives us the right to process your personal data?

We will process your personal data based on your consent.

Based on an agreement with NTNU, The Data Protection Services of Sikt – Norwegian Agency for Shared Services in Education and Research has assessed that the processing of personal data in this project meets requirements in data protection legislation.

### Where can I find out more?

If you have questions about the project, or want to exercise your rights, contact:

**Steven Magnus Emil Berglund Francis:** smfranci@stud.ntnu.no

**Frithjof Thorvik:** frithjot@stud.ntnu.no

If you have questions about how data protection has been assessed in this project by Sikt, contact:

**email:** personverntjenester@sikt.no

**telephone:** +47 73 98 40 40

Yours sincerely,

Steven Magnus Emil Berglund Francis & Frithjof Thorvik

## I give consent to participate in the online survey, and for my data to be processed until the end of the project.

Antall svar: 4

Svar	Antall	% av svar	
No	0	0%	0%
Yes	4	100%	100%



We regret that you have decided to opt out of the survey. Please close this window to complete the opt-out process. If you choose not to submit this questionnaire, none of your data will be collected or used for the purposes of this survey.

## Misc questions



## Do you have any experience using Git?

Antall svar: 4






Svar	Antall	% av svar	
No	1	25%	 25%
Yes	3	75%	 75%

## How would you rate your competence with Git?

Antall svar: 3






Snitt: 2.00

Median: 2

Svar	Antall	% av svar	
5	0	0%	 0%
4	0	0%	 0%
3	1	33.3%	 33.3%
2	1	33.3%	 33.3%
1	1	33.3%	 33.3%

## I am an experienced gamer

Antall svar: 4

Svar	Antall	% av svar	
Strongly agree	0	0%	 0%
Agree	0	0%	 0%
Neutral	0	0%	 0%
Disagree	3	75%	 75%
Strongly disagree	1	25%	 25%

## What are you studying?

• Computer Science

• Energy and Environment

• Datateknologi

• Datateknologi

## Which grade are you in?

---

• 6

---

---

• 5

---

---

• 1

---

---



• 1

---

---

## I have experience using code editors

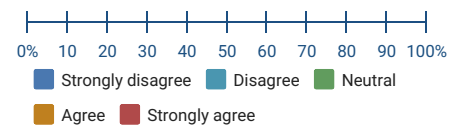
Antall svar: 4

Svar	Antall	% av svar	
Strongly agree	0	0%	0%
Agree	3	75%	 75%
Neutral	1	25%	 25%
Disagree	0	0%	0%
Strongly disagree	0	0%	0%

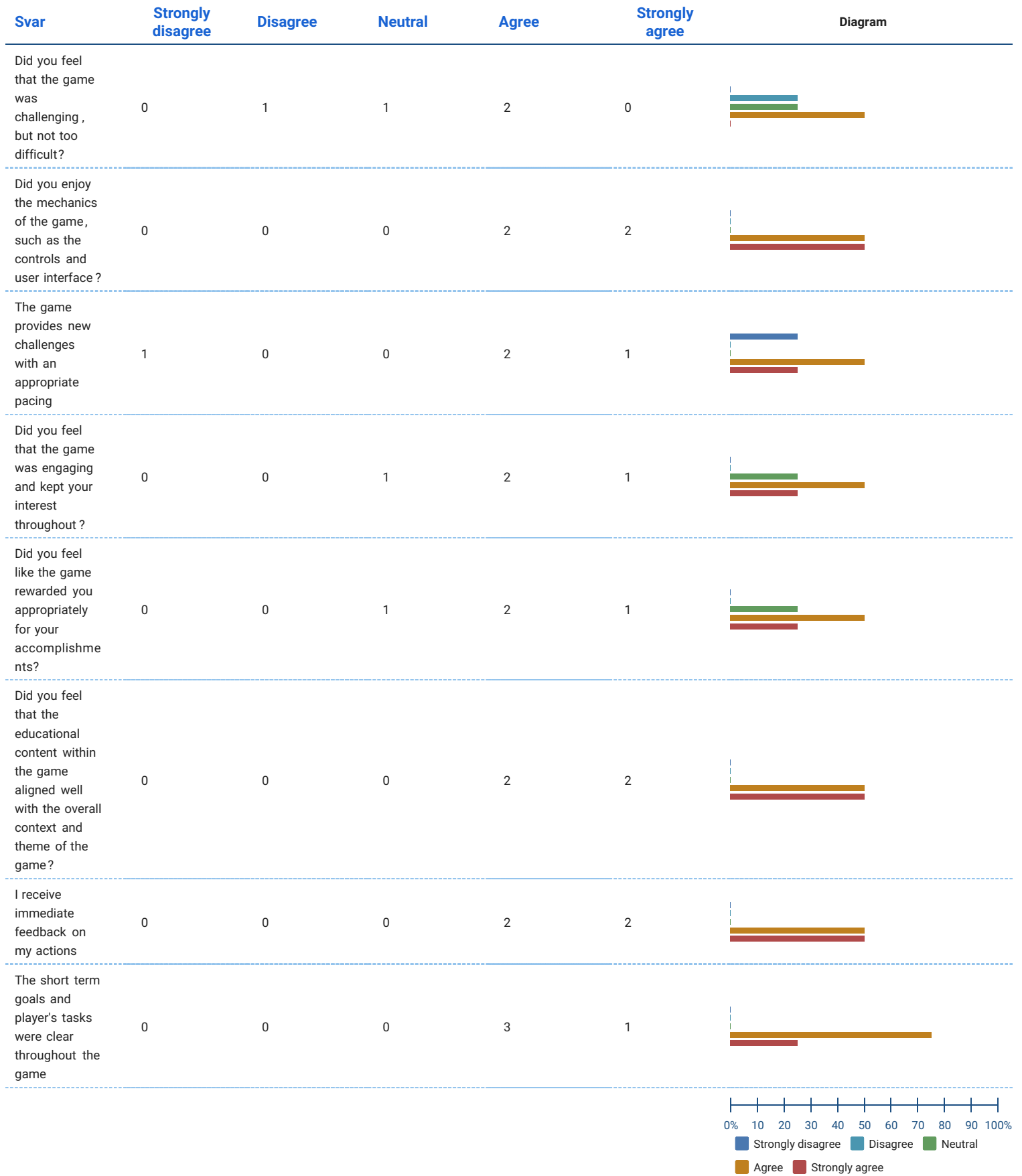
## Game Usability and Gameplay

# System Usability Scale

Svar	Strongly disagree	Disagree	Neutral	Agree	Strongly agree	Diagram
I think that I would like to use this game frequently	0	1	1	1	1	
I found the game unnecessarily complex	3	0	0	1	0	
I thought the game was easy to use	0	0	0	3	1	
I think that I would need the support of a technical person to be able to play this game	4	0	0	0	0	
I found the various functions in the game were well integrated	0	0	0	2	2	
I thought there was too much inconsistency in the game	3	1	0	0	0	
I would imagine that most people would learn to play this game very quickly	0	0	0	2	2	
I found the game very cumbersome to use	0	2	0	1	1	
I felt very confident playing the game	0	0	0	1	3	
I needed to learn a lot of things before I could get going with the game	2	2	0	0	0	



# Gameplay



# Learning

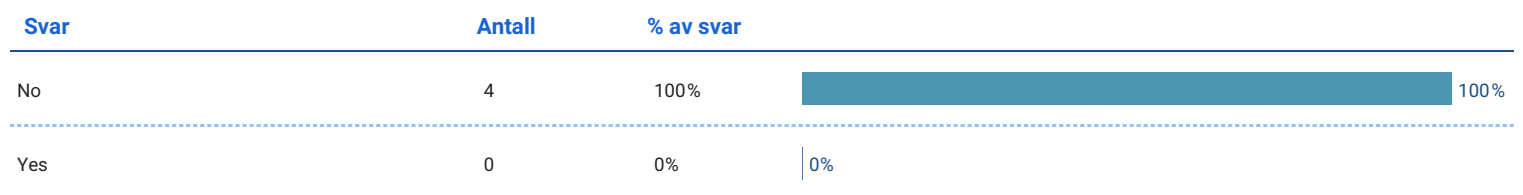
## Spørsmål uten tekst



## Bugs & Feedback

### Did any bugs prevent you from playing?

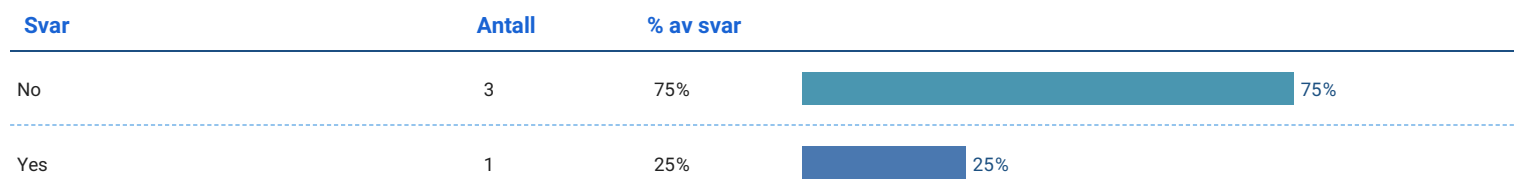
Antall svar: 4



### Please provide a description of the bugs

### Did any bugs reduce the game quality?

Antall svar: 4



### Please provide a description of the bugs

- Many spelling errors in the text.

---

## Do you have any additional feedback that you would like to share?

---

- A couple of UI things that bothered me: 1. When I purchased a new git command (git branch), for a long time I couldn't figure out how to dismiss the popup modal display. This is because the close button was in the top right corner, but on my large desktop monitor it was so far away from the content that I didn't see it. Also, it was inconsistent with the rest of the game controls which were either terminal commands or in the middle bottom part of the display. 2. Adding orders was a very cumbersome process. I appreciate the way this maps to creating files, but adding a separate file for each order of fries was tedious and made it hard to want to keep playing. A few other things: I thought the game was incredibly made overall. Very slick, a wonderful approach to teaching git. I came in with some (not much) experience in Git (I know, not your target audience), and still found the game enjoyable. Well done! Last thought: I would prefer the game progression to be a bit quicker. It is a lot of work (read, little actions) to complete a Day, but then the new options to purchase (namely one new ingredient or a single new git command) felt underwhelming for the work I'd put in.
- 

- I really liked the game! :)
- 

- No
- 

- I did find it difficult to find the names of the other branches. Other than that, I thought it was engaging and educational
-

---

## A4 Information Letter

✔ 500kr for 4 winners

✔ Digital participation

✔ ETC ~ 25-45 min

# Evaluating Different Methods for Learning Git

Are you interested in taking part in a research project for learning Git? Complete all the steps listed below and you will enter the competition for winning a **500kr** gift card!

## Purpose of the project:

You are invited to participate in a research project where the main purpose is to evaluate a method of facilitating the learning of core concepts in the Git version control system. By joining this project, you will have a chance to win a **500kr** gift card where 4 winners will be drawn.

You have been selected to participate in this study as you are in a field of study that indicates that you will require having skills in Git. **To participate, fulfill the three stages below before the end of April (30.04).**

Your contact information was obtained through NTNU.

## What to do:

### Stage 1 - Git Pre-test

ETC: ~ 5 min

The first test is to find your baseline knowledge of some core Git concepts. Do not worry if you can't answer any of the questions! You will get the opportunity to learn after this test.

### Stage 2 - Learning Git

ETC: ~ 15-30 min

You will be presented with a method of learning Git.

### Stage 3 - Git Post Test & Survey

ETC: ~ 5-10 min

The next step is to complete another test to see if you have improved with the provided method of learning. This will also include a short survey to get insight into your experience with this form of learning.

**START**

<https://l.linklyhq.com//1iDto>

(Click the button or the link to **start the survey**. All information regarding your **privacy** and the necessary details to complete the stages above is located in the survey)

**NOTE:** FAQs are listed on the next page of this document



## **FAQs**

### **What is Git?**

*Git is a version control system for tracking changes in files, particularly code. It's beneficial for software development and any field that involves document changes. It enables multiple people to work on a project simultaneously and keeps track of their alterations. Think of it as a history book for your documents, tracking changes made by who and when.*

### **How will I know if I won or not?**

*We will send you an email when the study is finished, and when we have randomly drawn from the participants of this study. You will receive an email about whether you have won or lost.*

**Note:** *We will only collect your email in order to send the information about whether you have won or not. If you don't want to participate in the drawing, do not provide your email in the survey.*

### **What are my chances of winning?**

*We don't know how many will participate in the survey, but we are aiming for between 30-50 participants. Meaning your winning chances will approximately be around 8-13%.*

---

## A5 Experiment - Test versions

# Git Tests

This test is designed to evaluate your understanding of some core Git commands.

**We kindly request you avoid using any aids or additional materials while answering the questions.** The purpose of this test is to establish a baseline for evaluating the effectiveness of the learning method presented later. We would like to emphasize that this test is not intended to pass any judgment on your skills

## 1. `git clone <URL>`

<INPUT>

- a. You have a Git project/repository that you wish to add to your computer with the following **URL**: <https://github.com/User48372/git-cooking.git>. Which command would you use?
- b. What command would you use to download a copy of a Git repository located at <https://github.com/User48372/git-cooking.git> onto your computer?

## 2. `git checkout <BRANCH>`

<INPUT>

- a. You are in a Git project that has two branches (**main** and **test2**). You are currently on the **main** branch but want to move to the **test2** branch. Which command would you use?
- b. You are currently on the **main** branch of a project, but you want to work on the **dev** branch. Which command would you use?

## 3. `git add <PATH>`

<INPUT>

- a. You have modified two files, and with a single command, you want to register/stage those changes in your Git repository. Which command would you use?
- b. After creating one file and modifying another you want to register/stage these changes with one command. Which command would you use?

4. **git commit -m <MSG>**

<INPUT>

- a. You want to save changes you have registered/staged with an “**add new feature**” message attached. Which command would you use?
- b. You have registered/staged some changes, and want to save these changes with the message “**made some changes**” to your Git repository. Which command would you use?

5. **git push**

<INPUT>

- a. You are on the **dev** branch and the remote is named **origin**. You want to update the remote Git repository with the changes you have saved in the **dev** branch on your local project. Which command would you use?
- b. After making some changes in your local project, you wish to send them to the remote Git repository. You are on the **test** branch and want to save those changes to the **origin** remote. Which command would you use?

6. **git fetch**

<INPUT>

- a. You received a notice that there are new updates added to the remote Git repository. You want to receive those changes without merging them into your local repository. Which command would you use?
- b. Your friend has pushed a new branch to the remote Git repository. Now you want to receive those changes without merging them into your local repository. Which command would you use?

7. **git branch**

<INPUT>

- a. You don't remember all the available branch names in the Git project on your computer. Which command would you use?
- b. You want to list all the available **local** branches of your Git repository. Which command would you use?

8. **git merge**

<INPUT>

- a. You are on the **main** branch and wish to integrate the changes from the **dev** branch into the **main** branch. Which command would you use?
- b. Assuming you are on the **dev** branch, you intend to combine the modifications made in the **test** branch into the **dev** branch. Which command would you use?

9. **git status**

<INPUT>

- a. You want to get information on the current state of your project (If you have created, deleted, or modified any files). Which command would you use?
- b. You want to see which files you have modified and staged, in addition to information about the state of your Git repository. Which command would you use?

10. **Git repository**

<CHOICE>

- a. What is a Git repository?
  - A workspace that is keeping track of changes in a project
  - A text-based interface that allows you to interact with your computer system
  - A ZIP archive containing code files
  - An interface for developers to write, modify and debug code in a specific programming language, such as Python, JavaScript, or HTML.
- b. What is a Git repository?
  - A system that tracks and saves the history of all changes made to the files in a project
  - An editor used to write source code
  - A server that manages software updates and patches
  - A web application for creating and editing files in a project

11. **Git branch**

<CHOICE>

- a. What is a Git branch?
  - A physical branch used to measure the progress of a Git project
  - A copy of the main project that is created for backup purposes
  - A separate line of development that allows for experimentation and isolation of changes
  - A virtual environment in which Git stores files for development and testing
- b. What is a Git branch?
  - A named pointer to a commit that enables the developer to make separate and independent changes to the codebase
  - A copy of a Git repository located in Git cloud storage, allowing parallel changes
  - A file that contains the changes made to a Git project

- A version of a project that has been merged into the main branch.

## 12. Git remote

<CHOICE>

- a. What is a Git remote repository?
  - A repository that contains only the most recent changes
  - A location where Git stores the project files locally on a developer's computer
  - A copy of the main project that is created for backup purposes
  - A shared Git repository that is hosted on an online server
- b. What is a Git remote repository?
  - A separate repository that is used to experiment with code changes
  - A virtual environment in which Git stores files for development and testing
  - A system containing information about which changes will be stored in the next commit
  - A Git repository hosted on the internet

## 13. Git staging area

<CHOICE>

- a. What does the Git staging area contain?
  - A clone of the Git repository
  - Information about which files are saved in the Git repository
  - A copy of all the branches in the Git repository
  - Information about which changes will be stored in the next commit
- b. What does the Git staging area contain?
  - A copy of the entire repository
  - Changes that have been committed to the repository
  - A snapshot of the repository at a specific point in time
  - Changes that are ready to be committed to the repository

---

**A6 Experiment - Traditional learning document**

# Learn Git

Hello! This is a document for introducing you to some core concepts within Git, to give you the foundational understanding in order to use it. The first section **Walkthrough** will give you the main points you need to be presented in a logical order that is similar to how you normally would use it. The next section **Theory & Concepts** will give you some additional information about the theory and concepts within Git but is not necessary to read through.

## Table of contents

### Walkthrough

1. [Terminal](#)
2. [Git concepts](#)
3. [Git Flows & Commands](#)

### Theory

1. [Introduction](#)
2. [Working Directory](#)
3. [Staging Area](#)
4. [Commit](#)
5. [Branch](#)
6. [Merge](#)
7. [Remote](#)
8. [Terminal](#)
9. [Git commands](#)



## Walkthrough

This walkthrough will give you a logical order and most important information about what you need in order to use Git. This will only walk you through the basics of Git, which will give you the knowledge to use it in a job or in your studies.

## Terminal

In order to use Git, you need to learn the basics of a computer terminal, which is the most common way of using Git today.

- A terminal is a text-based interface that allows you to interact with your computer system.
- By chaining specific words in your terminal input, you navigate to specific programs that you want the computer to execute for you.
- An example can be **git push**.
  - **git** is the first word, which tells your computer to use **git** (a program you have downloaded on your computer) to perform the command.
  - **push** is the second word, which tells your computer to search for a script that can be run. The computer will either run the program if everything is ok or give you feedback about any issues with the command.
- By knowing the different chains of commands, you are able to interact and utilize the benefits that Git can provide you.
- You can view the structure of a command as if you are commanding someone to do a task, ex: **sarah eat salad**, where you first identify the person you want to command (**sarah**), followed by the action she should do (**eat**), and finally what she should eat (**salad**).

## **Git Concepts**

In order to understand where and how Git is used, there are some concepts that you need to know about.

### **Git Repository**

The git repository is a system that tracks and saves the history of all changes made to the files and folders in a project. A git repository can both exist remotely and locally. The local instance of a git repository can be connected to a remote repository, which is hosted online. This enables multiple participants to interact with the same git repository.

### **Working Directory**

The working directory is where you will be able to create, modify, or delete code. Within this directory, Git will be able to track all the files and folders present, giving it access to monitor any changes made to them. Your working directory is where you are constantly making changes before you save and store the finished results.

### **Staging Area**

The staging area allows you to temporarily save and organize your changes, and decide which changes to include in the next commit. Where a commit is an action within Git that will save the changes you have prepared in the staging area into your Git repository.

## Git Flows & Commands

Now we will go through an example scenario of using Git, in order to introduce and make some core Git commands familiar to you.

### Step 1 - git clone

Imagine that you have been invited to a git repository to help out with development and you received the URL for the remote repository. You need to get the git repository on your local computer to start working. To copy a remote git repository to your local computer, you can use:

```
> git clone <repository-url>
```

```
> git clone https://github.com/User1234/project1.git
```

### Step 2 - git checkout

Imagine now that you now have the repository on your local computer. You were told to start working on the **fix-function** branch. A branch is sort of like a working space for you alone or a smaller team, where you work on a separate line of development inherited from another branch, like an extension. To start working on a specific branch in the project, you can use:

```
> git checkout <branch-name>
```

```
> git checkout fix-function
```

### Step 3 - git status

Imagine now that you have modified a file called **func.js** and fixed an issue, and are ready to save those changes. Before you save any changes to your git repository history, you need to add and prepare the changes in the staging area. You have only modified one file, but you want to make sure that this is the only change made. To get the current status of the working directory, you can use:

```
> git status
```

This will give you a response where you see that the **func.js** file, that you modified in the **src** folder, is returned in red, where **src/func.js** is the path to that file:

```
On branch fix-function
```

```
Changes not staged for commit:
```

```
(use "git add <file>..." to update what will be committed)
```

```
(use "git restore <file>..." to discard changes in working directory)
```

```
modified: src/func.js
```

```
no changes added to commit (use "git add" and/or "git commit -a")
```

## Step 4 - git add

Now that you know the status of your working directory, you can add and prepare the changes you made in the staging area. To add files to the staging area, you can use:

```
> git add <path>
```

```
> git add src/func.js    or    > git add .
```

You can add a single file at a time by providing the path to that file, or you can add all the changes you made by providing a "." which symbolizes the path to the entire directory folder containing all your files.

## Step 5 - git commit

You have added your changes to the staging area, and now you are ready to commit and save those changes into your git repository's history. To commit staged files, you can use:

```
> git commit -m <message>
```

```
> git commit -m "fixed function"
```

We attach a message to the commit to make it easier to locate the exact point in the repository's history where a certain change was made.

## Step 6 - git push

Imagine that you are finished with your task, and the commit(s) you have registered in your local git repository are now ready to be sent to the remote git repository to allow your collaborators to get access to the changes. To push new commits, you can use:

```
> git push <remote> <branch>
```

```
> git push    or    > git push origin fix-function
```

When using only **git push**, you push the commits that have been made on the branch you are currently on. With **git push <remote> <branch>** you can also specify which remote to push to, and from which branch.

## Step 7 - git merge

Because the issue has been fixed in the **fix-function** branch, you want to make those changes apply in another branch, called the **dev** branch, (which is the branch that the **fix-function** branch was created/inherited from). In order to merge a branch into another branch in the terminal, we need to be located in the branch that we want to merge into (in this case the **dev** branch). To do so, we first move to the **dev** branch and then merge with the **fix-function** branch:

```
> git checkout <branch>
> git merge <branch>

> git checkout dev
> git merge fix-function
```

Usually, the merging of branches happens within tools like GitHub and GitLab (where remote git repositories can be hosted online), which enables collaborators to look over your code through an interface, in addition to more features to help protect and make sure that everything is merged as you want it.

## Step 8 - git fetch

Let's say you were notified by a friend of a new branch, called **fix-color-bug**, that they added to the remote repository. You try to use **git checkout** to move to the branch, but it does not work. The reason is that only the remote repository has been updated, but your local instance of the repository does not get updated automatically. To get the updated state, you can use:

```
> git fetch

d6be7d6f..003a32c  dev          -> origin/dev
* [new branch]    fix-color-bug -> origin/fix-color-bug
```

By using **git fetch** you now have access to the new branch that was added. This command does not update the code for branches that you already have locally. If you want to get the updated code as well, **git pull** must be used from the branch you want to pull the code from. When you execute **git pull**, Git first performs **git fetch**, and then proceeds with **git merge**. This essentially retrieves the most recent changes from the remote repository, and integrates them with your local copy.

## Step 9 - git branch

Imagine that you don't remember the branches that are available in the git repository but want a quick way to list all the branch names. To get a list of branches you can use:

```
> git branch    or    git branch -r
```

The **git branch** command provides a list of the branches that are stored in your local git repository. While **git branch -r** lists the branches that exist on the remote git repository.

# Theory

## Introduction

Git is a tool that helps you manage the changes you make to your files. It's like a time machine for your code! You can think of it as a way to keep track of all the changes you make to your code over time so that you can always go back to an earlier version if you need to.

With Git, you can make changes to your code without worrying about losing anything, because Git keeps a record of all the changes you make. You can also work on your code with other people, and Git will help you merge everyone's changes together.

Git has a few basic concepts you'll need to understand. The first is a repository, which is just a fancy word for a workspace that Git is keeping track of. You can create a new repository or clone an existing one.

Once you have a repository, you can make changes to your code and commit those changes. A commit is like a snapshot of your code at a particular point in time. You can add a message to each commit to describe what you changed.

You can also create branches in Git, which are like alternate versions of your code that you can experiment with. You can switch between branches and merge them together when you're ready.

Finally, when you're ready to share your code with others, you can push your changes to a remote repository, like one hosted on GitHub.

That's the basic idea of Git! It can be a bit confusing at first, but once you get the hang of it, it's a very powerful tool for managing your code.

In the following sections, you can get more in-depth information about git.

## Working Directory

The working directory acts as the platform where you can modify files and make changes to your project. It can also be viewed as a checkout of a version/snapshot of your repository, where you can make changes. These changes are not saved in Git before you commit them or stage them. Within the working directory, files can be in one of two states, tracked or untracked. Tracked files are files that existed in the previous snapshot or in the staging area, whilst untracked files are new files not yet tracked in the Git repository.

## Staging Area

A modified file from the working directory can be added to the staging area. The staging area contains information about which changes will be stored in the next commit. Meaning that the staging area contains information about the modified files in the state it was in when it was added to the staging area.

## Commit

A commit takes all the changes added to the staging area and saves them to the Git repository. Git stores its data in a series of snapshots, so whenever you commit, you save a snapshot of the current state of your files. Every commit contains a pointer to its previous commit, resulting in a stream of snapshots - called the commit history.

## Branch

A branch in Git is essentially a pointer that points to a commit. The default branch is called the main branch or master branch. Whenever you commit, this branch points to the last commit you made, and for each commit, the main branch pointer automatically moves forward. Whenever you make a new branch you create a new pointer that points to the commit you branched from. This new branch pointer can move independently from other branches. Git also has a special pointer called HEAD which can be used to know which branch you are currently on. In this way, you can switch between branches. This allows you to experiment with changes by making new branches, doing some changes in that branch, and switching to another branch to make other changes. For these changes to be integrated with each other, Git supports a feature called merging, which integrates the changes with each other.

## Merge

Merging is how Git manages to integrate changes from different branches together. Based on the changes made, the merge can happen in two different ways: fast-forward merge or three-way merge. A fast-forward merge occurs whenever the changes you are trying to merge can be reached by following the commit history. When this happens, the merge can be simplified by moving the pointer forward, as there are no conflicting changes to merge together. A three-way merge occurs when the commit history has diverged down the line. This happens when the commit of the branch you are trying to merge is not a direct ancestor of the branch you are merging in. In this case, Git creates a new commit, called a merge-commit, which takes the changes from the two branches and combines them. This commit is special compared to a normal commit as it has more than one parent. In the case that the changes between the two branches are in conflict, the three-way merge cannot proceed. This is called a merge conflict and it happens when the same part of a file is changed in different ways in the two branches you want to merge. When this happens, Git pauses the merge and cannot proceed until you have resolved the conflicting changes. Once the conflicting changes have been resolved, the merge can proceed in the same way as before.



## **Remote**

Git enables you to collaborate on Git projects through remote repositories. A remote repository is similar to a normal Git repository, except that it is hosted on the Internet. This remote repository can then be accessed by multiple people, allowing collaboration through pushing and pulling in Git. Pushing and pulling is the act of sending data to the remote (pushing) and getting data from the remote (pulling). This enables you to work on a directory make some changes, commit those changes and push those changes to the remote. The person you are collaborating with can then pull those changes, gaining access to the changes you made. If you want to start working on an existing remote repository, you can clone the project, which pulls down every file and its history from the remote project.

## **Terminal**

A terminal is a text-based interface that allows you to interact with your computer system. By chaining specific words in your terminal input, you navigate to specific programs that you wish the computer to execute for you. If you want to use git commands, you need to start by writing git and chain and keep chaining the correct words to reach the thing you need to run. All the git commands use the terminal

# Git Commands

## git clone

*Copy and download a remote repository to your local machine.*

- Use **git clone** to set up a local copy of a remote repository. This allows you to have a local copy of a remote repository so you can work on it and make changes without affecting the original repository.

## git fetch

*Downloads changes in branches from the remote.*

- Use the **git fetch** command to retrieve the latest changes made by other contributors in the shared remote repository.

## git checkout

*Switch to another branch and check it out in your working directory.*

- Use **git checkout <BRANCH>** to switch to a specific branch in your local repository. Changes made in one branch do not affect other branches, meaning that you can work on new features in isolation.
- Use **git checkout -b <BRANCH>** to create a new branch based on the current branch.

## git add

*Adds your modified items to the staging area.*

- Use **git add <FILE\_PATH>** to add and prepare a modified file to be committed later.
- Use **git add .** to add all modified files.

## git commit

*Saves a snapshot of the staged changes in the local repository, creating a new commit in the project's history.*

- Use **git commit -m <message>** to capture and preserve the changes made to the files in the repository. This creates a new commit in the commit history, allowing you to keep track of the evolution of the codebase and easily revert to previous versions if necessary.

## git push

*Transfer committed changes from the local repository to the remote repository.*

- Use **git push** or **git push origin <BRANCH>** to transfer your local changes in a branch to a branch in the remote repository. This allows other people to work with you and see the changes you have made. If no branch name is provided in the command, the branch you are currently in will be used.

## git status

*Shows modified items in the working directory and staged items in the staging area.*

- Use **git status** to display the current state of the working directory and the staging area, indicating which files have been modified, staged for commit, or remain unmodified.

## git branch

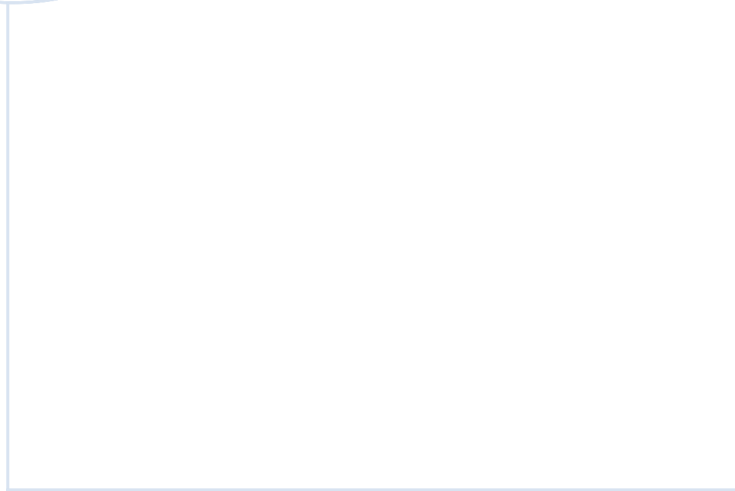
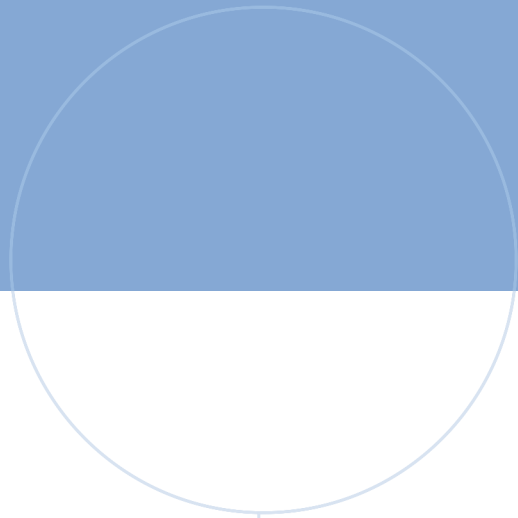
*List up existing branches.*

- Use **git branch** to list existing branches; the current branch will be marked with an asterisk.
- Use **git branch -r** to show remote branches.

## git restore

*The restore command is used to discard uncommitted changes.*

- Use **git restore <FILE\_PATH>** to recover changes in the working directory to their previous state, or recover files that were deleted from the repository.
- Use **git restore .** to restore all files.
- Use **git restore —staged <FILE\_PATH>** to unstage changes in the staging area.
- Use **git restore —staged .** to unstage all changes in the staging area.



 **NTNU**

Norwegian University of  
Science and Technology