Mathias Hagen
Apimanju Rajan
Ulrik Auflem Eikås
Herman Svae Nauf

Proposed interactive platform for visualizing collected data on road railings for iSi AS

Bachelor’s thesis in Bachelor of Engineering in Computer Science
Supervisor: Di Wu
May 2022

Bachelor’s thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of ICT and Natural Sciences

Mathias Hagen
Apimanju Rajan
Ulrik Auflem Eikås
Herman Svae Nauf

Proposed interactive platform for visualizing collected data on road railings for iSi AS

Bachelor's thesis in Bachelor of Engineering in Computer Science
Supervisor: Di Wu
May 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of ICT and Natural Sciences

NTNU
Kunnskap for en bedre verden

Abstract

After the major uncovering of faults on railings along the roads in 2019, The Norwegian Public Roads Administration acknowledged the lack of inspection routines. iSi started an innovation-project shortly after to streamline and improve the inspection routines.

This thesis provides an insight to the approach and method for the development of the website used to visualize iSi's processed data.

The main objective tied to the problem is developing a web-based interactive map to display the data iSi has gathered. Some of the key features include filtering, searching and an advanced presentation of the actual data. The main users of the website are road owners and The Norwegian Public Roads Administration. Ease of use has been a big priority, this allows for a more efficient process, accessible for everyone.

To create the website we employed the SCRUM project management, meaning bi-weekly meetings with the client, and daily standup-meetings with the group. The website was developed with Angular using the TypeScript language. The library Leaflet was used to display the interactive map.

During bi-weekly meetings, the client provided useful feedback and a better insight as to what the users will need in terms of functionality. This helped in guiding the website in the right direction, and let us engage with the user's actual needs.

Structured tests were planned with multiple participants from The Norwegian Public Roads Administration, as well as another group testing the user experience.

This solution will have positive impact on both the environment and society in many ways. By using our solution, the quality control of railings will be a more cost-effective process. This solution will also benefit the environment, because of the reduced need of cars and human resources. Relevant UN goals are goal 9, goal 8 and goal 17.

The team was able to create an interactive map by utilizing the framework, library and language mentioned previously. One of the main takeaways from this project has been the importance of a structured SCRUM-team. The planned meetings and sprint reviews have been very useful for the progression of the web-site.

Sammendrag

Etter den store avdekkingen av feil på rekkverk langs veiene i 2019, erkjente Statens vegvesen manglende tilsynsrutiner. Kort tid etter startet iSi AS et innovasjonsprosjekt for å effektivisere og forbedre inspeksjonsrutinene.

Denne oppgaven gir et innblikk i tilnærmingen og metoden for utviklingen av nettsiden som brukes for å visualisere iSi sine behandlede data.

Hovedmålet knyttet til oppgaven er å utvikle et nettbasert interaktivt kart for å vise dataene iSi har samlet inn. Noen av nøkkelfunksjonene inkluderer filtrering, søking og en avansert presentasjon av de faktiske dataene. Hovedbrukerne av nettstedet er vegeiere og Statens vegvesen. Brukervennlighet har vært en høy prioritet, dette gir mulighet for en mer effektiv prosess, tilgjengelig for alle.

For å lage nettsiden anvendte vi SCRUM-metodikk, som betyr møter med kunden annenhver uke og daglige standup-møter med gruppen. Nettstedet ble utviklet med Angular ved å bruke TypeScript-språket. Biblioteket Leaflet ble brukt til å vise det interaktive kartet.

Under de bi-ukentlige møtene ga oppdragsgiver nyttige tilbakemeldinger og bedre innsikt i hva brukerne ønsker av funksjonalitet. Dette bidro til å lede utviklingen av nettstedet i riktig retning, og lot oss engasjere oss i brukerens faktiske behov.

Det ble planlagt strukturerte tester med flere deltakere fra Statens vegvesen, samt en annen gruppe som testet brukeropplevelsen.

Denne løsningen vil ha positiv innvirkning på både miljø og samfunn på mange måter. Ved å bruke vår løsning vil kvalitetskontrollen av rekkverk bli en mer kostnadseffektiv prosess. Denne løsningen vil også være til fordel for miljøet, på grunn av redusert behov for biler og menneskelige ressurser. Relevante FN-mål er mål 9, mål 8 og mål 17.

Gruppen var i stand til å lage et interaktivt kart ved å bruke rammeverket, biblioteket og språket nevnt tidligere. Et av hovedtrekkene fra dette prosjektet har vært viktigheten av et strukturert SCRUM-team. De planlagte møtene og sprintgjennomgangene har vært svært nyttige for utviklingen av nettstedet.

Preface

We wanted our bachelor thesis to make a difference, have a positive impact on the society. After reading all the different bachelor thesis one immediately caught our attention, “Map display of road railings”. We liked the idea of streamlining a time and resource consuming process from a labor intense sector to increase the traffic safety. By creating the interactive map, we could be a part of the solution which increase traffic safety and road infrastructure, as well as potentially saving lives in the traffic when an accident occurs.

Acknowledgements

We would like to acknowledge and extend our sincere thanks to our supervisor Di Wu. Her guidance and advice helped us through all stages of development. A big thanks to Ida Kjørholt and Nils Tarjei Hjelme for providing the topic for our bachelor thesis and the basis for the application development. Their feedback and guidance throughout the process has been phenomenal.

We would also like to thank Girts Strazdins for being there for us throughout the bachelors degree. A big thanks for providing us with help and guidance when creating the server that hosts our application.

Lastly we would also like to extend a heartfelt thanks to all the professors involved in the computer science program.

Mathias Hagen	19/05 Ålesund
Hagen, Mathias	Time & Place
Apimanju Rajan	19/05/22 Ålesund
Rajan, Apimanju	Time & Place
Ulrik Auflem Eikås	19.05.22 Ålesund
Eikås, Ulrik Auflem	Time & Place
Herman Svae Nauf	19/05/22, Ålesund
Nauf, Herman Svae	Time & Place

Table of Contents

List of Figures

List of Tables

List of Code examples

Glossary

$\LaTeX$ software for document preparation. Unlike text editors where ”you get what you see” such as Microsoft Word, you write plain text. The writer uses markup tagging to define the structure of the document. 30

backend Backend is usually the part of a system that is not shown to the user, typically stores and calculates data.. 16

cloud computing an on-demand available resource that can serve as cloud storage and computing power, without the user actively managing it. 32

cross-platform software that can run on multiple types of computer systems. 33

ECMAScript is a programming language specification that Ecma International has standardized in the standards ECMA-262 and ISO / IEC 16262. The specification was initially based on JavaScript, but this has now been reversed such that JavaScript now follows the development of the ECMAScript standard. 24

framework framework in programming is a tool that provides already made components or solutions in order to speed up the development. 16, 36

Git software used to track changes in a collection of files. Providing developers with speed, data integrity and support for distributed workflows. 30

IntelliSense is a context-aware code completion tool that helps to speed up the coding process by eliminating typos and other typical errors. 31

latitude an angle which ranges from 0° at the Equator to 90° at the poles. 21

library is a collection of functions and classes which by itself usually is not a complete program, but they can be used to save development time by using existing code. 33, 52

longitude a geographic coordinate that describes a point’s east-west location on the Earth’s surface. 21

MongoDB a source-available dtabase program based on NoSQL. 33

open source software with public source code. It is distributed under a license where the copyright holder gives its users rights to use, change and distribute the software for any purpose. 24, 32, 33, 52

repository A code repository is an archive that stores all the documents of a project, such as code, documentations, notes etc.. 20

single-page application is a web application or website that interacts with the user by dynamically rewriting the current web page with new data from the web server, instead of the defualt method of a web browser loading entire new pages. 33

SSH Secure shell. A program and network protocol to access command line on another computer. . 32

strict a programming language that only allows strict functions. This means the parameters of the given functions must be evaluated completely before they can be called. 24

superset given sets A & B, if A contains all elements of set B, A is a superset of B. TypeScript contains all elements of JavaScript, and is therefore a superset. 24

transcompiled Transcompile is the process of translating source code from a programming language to equivalent source code in the same or a different programming language. 24

web development the process of creating websites. Including design, publishing and maintaining the website. 23

well-known text is used to both construct new instances of the type aswell as to convert the existing instance to more readable text form. 14, 21

wireframe is an illustration of a page's interface and layout to help visualize the concept. 29

Acronyms

API Application Programming Interface. 16, 19, 31, 36, 38–43

BDD Behavior-driven development. 26, 35

CD Continuous Delivery. 20

CD Continuous Deployment. 20, 32

CI Continuous Integration. 20, 32

CSS Cascading Style Sheets. 23, 24

HTML HyperText Markup Language. 19, 23, 24

HTTP Hypertext Transfer Protocol. 19

IaaS Infrastructure as a Service. 32

IDE Integrated development environment. 25

JS JavaScript. 24, 33, 34, 52

JSON JavaScript Object Notation. 33

MVP Minimum Viable Product. 41, 46

REST Representational State Transfer. 19

SCSS Sassy CSS. 24

TCP Transfer Control Protocol. 19

TS TypeScript. 24, 34, 46

WKT Well-known text. 21, 43

1 Introduction

1.1 Background

A well-maintained road is an important factor when trying to improve traffic safety, and it is for that reason required in Norway to control the quality of all railings along all roads once every year. Today's manual practice of controlling railings is both time- and resource-consuming, at the same time as it increases the traffic danger along the road. This is due to there being one vehicle driving slowly with one person inside the car manually inspecting the railing.

In 2019 The Norwegian Public Roads Administration uncovered a series of faults on the railings. This caught both the media's attention and The Office of the Auditor General of Norway's attention. They called for a more effective solution to carry out these quality controls to improve the road safety. In 2020 iSi AS started an innovation-project to streamline and improve this process.

iSi's solution is based on mass-collection of images and tailored machine learning models for image recognition. These models automatically detect what type of railing is depicted and any visible errors, they also provide an accurate position of the railings. Image data is collected using a custom rig with high-tech cameras, infrared flashes and GPS. This rig is mounted behind a car, the data collection can be done in regular driving speed, up to 80 kilometers per hour. This makes the process of collecting data fast and effective, while maintaining road safety.

1.1.1 Objectives

We collaborate with a company called iSi located in Åndalsnes. iSi is an IT company that works with digitalizing of work processes, and specialize on solutions with focus on electronic interaction, documentation and machine learning.

The image-data iSi has collected is stored in a database, and the actual images are stored in a cloud storage-service. Every meter of the railing is photographed with three cameras, one upper and two lower.

iSi has challenged us to present the data in an interactive map, where you can filter based on the criticality, group criticality on a stretch of road and display the images in a way that allows users to get an understanding of the risk tied to different stretches of road.

1.1.2 Limitations

From NTNU

Having a parallel subject in conjunction with the bachelor project has been a big limiting factor. The subject has taken a big portion of our time due to the nature of the subject, as it is an intensive subject that runs over a span of 10 weeks.

The aforementioned subject had its exam date in the end of March, and up to that point it was increasingly time consuming due to exam preparations. After the course finished, the group was able to allocate more hours, and show increased progress and effectiveness.

From project provider

Since the development of the backend was an ongoing process on iSi's side while we worked on the project, this has also been a limiting factor. There has been functionality we had to delay because we did not have the backend data to use. Since the backend was not ready, we got test data to use, but some of them had bugs and missing data which made for example the implementation of the graph hard since it gave the wrong impression of the data displayed in the graph.

1.2 Motivation

The primary motivation of this project has been to be able to create an interactive platform that road owners can utilize to gain a good overview over the railings on the different roads they are in charge of. By doing this we can achieve safer roads all throughout Norway by establishing this webpage with good functionality to filter on different defects and determine the criticality of the state of the road railings. This website will be a tool that can be used to make the roads in Norway safer and can potentially prevent lethal consequences when an accident along the road occur.

1.3 History

As previously stated, iSi began this innovation-project in 2020 and has been working on the mass-collection of images and their machine learning models since then. iSi currently has millions of images of road railings, including a large amount of images with different faults and errors on the railings. Our group was connected to this project in the beginning of 2022 and made the web page to display all this data in a clean and user-friendly way based on the goals below.

1.4 Goals

- Introduction to the data, and the corresponding Application Programming Interfaces (APIs)
- Agree to a framework.
- Develop a web-based map-solution for displaying and filtering of data.
- Show images in the map, with functionality to scroll through images, and choose the upper or lower camera.
- Look at the possibility of classifying sections.

1.5 Thesis structure

Section 2 - Theoretical background: Describes the required theoretical background for this thesis.

Section 3 - Method: Provides an insight to project management, tools and the structure of both frontend and backend.

Section 4 - Results: Presents the results from the project.

Section 5 - Discussion: Discussion of the results presented in the results section.

Section 6 - Societal impact: Describes the societal impact in relation to a system perspective with technology ethics and sustainable development goals in mind.

Section 7 - Conclusion and future work: Contains a conclusion of the thesis along with recommendations for future work.

2 Theoretical background

2.1 Guardrail

Guardrails are installed along the road to give protection from potentially hazardous surroundings. They serve to prevent serious injuries in an accident, and the center-guardrail to prevent serious collisions. When a guardrail is damaged, it must be identified so that it may be replaced or repaired. It is possible that someone has damaged it without notice, a bolt has come loose or that the posts moves with the terrain. By always replacing and repairing damaged guardrails, will the nation be one step closer to achieve the ”nullvisjon” the Norwegian Parliament decided in 2002. [30]

2.2 Code documentation

Code documentation is text that goes with software code to explain what it does, why it’s written in a certain way and/or how to use it. There are two types of documentation: documentation contained inside the code and supporting documentation about the code.

2.2.1 Documentation in code

Notes put in the same file as your code are referred as this form of code documentation. When code is running, your programming language ignores these notes, but they are utilized by both developers viewing the code and some tools that can create information based on the comments.

If you need to comment to explain what your code is doing, it may be wiser to change some of it to make it more apparent. Making modifications to make things clearer, on the other hand, might take time or be difficult. In some circumstances , having a comment is preferable to having nothing. Another reason to include documentation in your code is to provide instructions for third-party tools. [22]

2.2.2 Documentation About code

Documentation about code is non-code documents that are used to supplement people’s knowledge of the code. This category might include a variety of types of documentation. The most popular are README, developer docs and code examples.[9]

2.3 Accessibility

The technique of making your website accessible to as many people as possible is known as accessibility. We typically think about this in terms of people with impairments, but making sites more accessible also benefits other groups such as those who use mobile devices or have sluggish network connections. [20] [31]

You may also consider of accessibility as treating everyone the same and providing equal chances to everyone, regardless of the ability or circumstance. Excluding someone from a physical structure because they are in a wheelchair (most contemporary public buildings include accessible ramps or elevators) is just as unjust as excluding someone from a website because they have a vision handicap. We are all unique, yet we are all human, and as such, we all share the same basic rights. [20] [32]

Accessibility is the right thing to do. In certain countries, providing accessible websites is required by law, which can open up huge markets that would otherwise be unable to utilize your services or purchase your products. [20]

2.4 Client Server Communication

Clients and servers exchange messages in a request response messaging pattern. The client makes a request, and the server responds. To communicate, the computers must use a common language and follow rules to ensure both the client and the server know what to anticipate. A communication protocol defines the language and principles of communication.

2.4.1 HTTP

Hypertext Transfer Protocol (HTTP) is a communications protocol, which is mainly used to transfer HTML documents between the client and server with the help of the transport protocols. The most used transport protocol is Transfer Control Protocol. [12]

2.4.2 REST API

REST API or RESTful API is an Application Programming Interface that uses the Representational State Transfer (REST) architectural style. In essence, REST is a set of guidelines that can make the API faster and more lightweight [15]. An example of this is when we make a request to get information about a specific railing along a road, the API will return the state of that railing, such as height, pole-type, railing-type and any faults.

2.5 CI/CD and Deployment

2.5.1 Continuous Integration

Continuous Integration (CI) is the method of automating the integration of code changes from various contributors into a single software project. This enables developers to merge code changes into a common repository, from which builds and tests may be executed. Before integrating the new code, automated tools are employed to verify its validity.

2.5.2 Continuous Deployment

Continuous Deployment (CD) automates every stage after the code is written, which means the code is built with a continuous integration tool, and then it is deployed to a few test environments and all automated tests are executed. When all of the tests pass, the code is then put into production. This implies that the entire procedure is automated and no permission is required before the code is deployed.

2.5.3 Continuous Delivery

Continuous Delivery (CD) builds on the same principles as Continuous Deployment (CD) but differs in the sense that Continuous Delivery focuses on automating as much of the process as possible, instead of the whole process. Which means that an approval is needed from a developer before the code can be send to production.

2.6 Design patterns

A software design pattern is a general, reusable solution to a commonly occurring problem in software design within a given context. It is not a finished design that can be directly translated into source or machine code. It is, rather, a description or template for how to solve a problem that can be used in a variety of situations. Design patterns are best practices that have been formalized that a programmer can use to solve common problems when designing an application or system. [37][29]

2.6.1 Observer Design Pattern

Observer design pattern is a behavioral design pattern that allows you to define a subscription mechanism to notify multiple objects about events that occur to the object being observed. [23]

2.7 Standards

2.7.1 Standard representation of geographic point location by coordinates

According to ISO 6709 which is the international standard representation for latitude and longitude, the latitude comes before the longitude if there are not any other definitions that says otherwise. North latitude is positive and East longitude is positive [34].

However, geospatial software has a inconsistency of which order they put longitude and latitude in. There are some agreement on longitude and latitude order for geospatial formats, but it is still a mess for libraries and software. It is the developer's responsibility to be aware of this issue, read the necessary documentation, and flip coordinates if necessary to translate between different systems. [18]

2.7.2 Well-known text representation of geometry

Well-known text (WKT) is a text markup language used to represent vector geometry objects. WKT can represent the following geometric objects: points, lines, polygons, multi-geometries (which represents more than one geometry of the same dimensions in a single object), and geometry collections (which store geometries of different dimensions). [14]

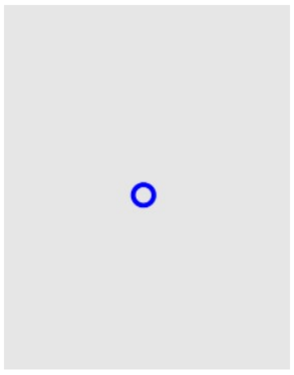
Examples of WKT of geometry are shown in table 1.

Geometry Type	Text Literal Representation	Comment
Point	Point (10 10)	a Point
LineString	LineString (10 10, 20 20, 30 40)	a LineString with 3 points
Multipoint	MultiPoint ((10 10), (20 20))	a MultiPoint with 2 points

Table 1: Example Well-known Text Representation of Geometry [16]

2.7.3 Geospatial data interchange format based on JSON

GeoJSON is an open standard format for representing simple geographical features as well as non-spatial attributes. It is built on the JSON data format. GeoJSON representations of instances of the following geometry types:


(a) GeoJSON Point

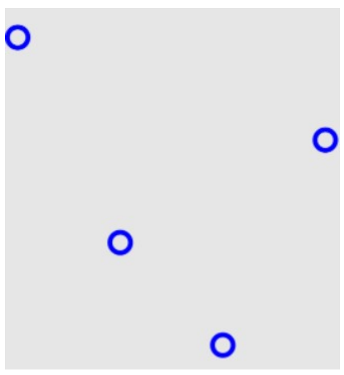
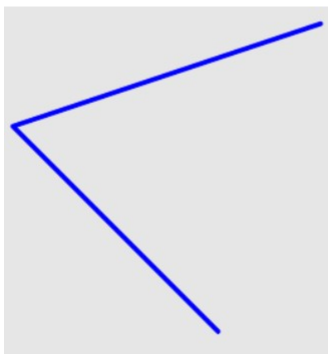

(b) GeoJSON MultiPoint

Figure 1: GeoJSON Point and MultiPoint [11]


(a) GeoJSON LineString

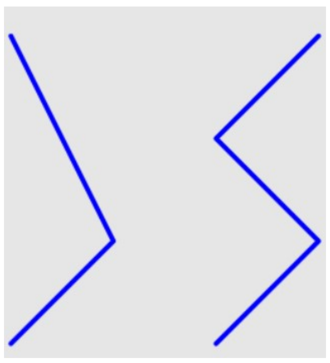
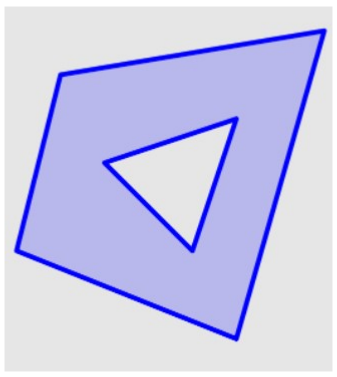

(b) GeoJSON MultiLineS-
tring

Figure 2: GeoJSON LineString and MultiLineString [11]


(a) GeoJSON Polygon

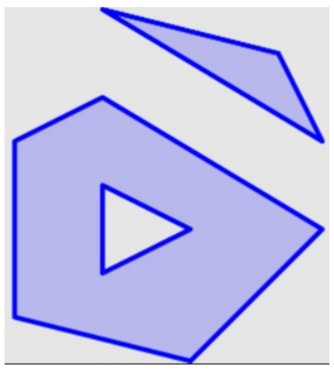

(b) GeoJSON MultiPolygon

Figure 3: GeoJSON Polygon and MultiPolygon [11]

Example of a GeoJSON object:

```
{
  "type": "Feature",
  "geometry": {
    "type": "Point",
    "coordinates": [125.6, 10.1]
  },
  "properties": {
    "name": "Dinagat Islands"
  }
}
```

Code example 1: GeoJSON example

2.8 Technologies

2.8.1 HTML

HyperText Markup Language, or mostly known as HTML and is the most basic building block of Web. Tags are used in the language to indicate the meaning and structure of the content that they enclose. A tag often instructs the text to be read as a headline, part of a numbered list, or a link to another website. Tags can also be used to refer to other resources, such as images.

HTML is only intended to tell the browser what type of content the page's different parts contain. The browser offers an extra effect to the text to make this function obvious, such as increasing the font size of headlines or making links blue and highlighted.

2.8.2 CSS

CSS or Cascading Style Sheets is a stylesheet language used for the presentation of a HTML document. It describes how the elements should appear on a screen. CSS is among the code languages in Web development.

CSS is intended to separate appearance from content, including layout, colors, and fonts. This separation can improve content accessibility; provide more flexibility and control in the specification of presentation characteristics; allow multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, reducing complexity and repetition in the structural content; and allow the .css file to be cached to improve page load speed between the pages that share the file and its formatting. [33]

2.8.3 SCSS

SCSS or Sassy CSS is a preprocessor to CSS and is similar, but allows additional functionality and is an extension of the syntax of CSS. This means that every valid CSS stylesheet is also a valid SCSS file with the same semantics. The .scss extension is used for files that use this syntax.

2.8.4 JavaScript

JavaScript or JS is a high-level programming language. It is one of the pillars of modern web development, along with HTML and CSS, and all modern browsers can run JavaScript applications without the need for extensions. The language can be used for both large web applications and basic scripting. The ECMAScript specification is used to standardize JavaScript.

JavaScript is an interpreted language that allows for prototype-based object orientation as well as functional programming. The language is well-known for its first-class features, including dynamic data types that are implicitly transformed while running.

The language was initially intended for online usage, but from the end of the 2000s, JavaScript has been widely employed as a server-side language, especially through the Node.js platform. [35] [19]

2.8.5 TypeScript

TypeScript or TS is an open source programming language developed and maintained by Microsoft. TS is a strict syntactical superset of JavaScript. TS is intended for large-scale application development and can be transcompiled into JavaScript. Existing JavaScript programs are also valid TS program since TS is a superset of JavaScript. [38] [21]

2.9 Version Control

The method of tracking and managing changes to software code is known as version control, sometimes known as source control. Version control systems are software tools that assist software development teams in managing changes to source code over time. Version control technologies assist software teams in working quicker and smarter as development environments have advanced.

In a specific type of database, version control software maintains track of every change to the code. If a mistake was made, developers may go back in time and compare previous version of the code to assist address the problem while minimizing disturbance to other team members.

2.10 Software Development Tools

Software development tools are computer programs that software developers use to build, debug, maintain, or otherwise assist other programs and applications. The most fundamental tools are a source code editor and a compiler or interpreter, which are utilized on a daily basis. Other tools, such as a debugger or profiler, are utilized in varying degrees based on the language, development technique, and individual engineer. Tools can be standalone programs that are run from the command line, or they can be components of a larger program that is run as a whole.

2.10.1 Source Code Editor

A source code editor is a text editor designed specifically for creating software. A source code editor can be a standalone application or a component of an IDE. They make creating and viewing source code easier by distinguishing between elements and routines, allowing programmers to more readily examine their code.

2.10.2 Code Formatter

A code formatter is a tool used in software development that changes the appearance of source code in a source-code editor. A code formatter is not a requirement for the source code to function properly; rather, it is a tool that makes the source code more readable and consistent.

2.10.3 Linter

Linting is the automatic detection of programmatic and stylistic flaws in the source code. This is accomplished by using a lint tool. Lint tools are simple static code analyzers. Linting is essential for reducing mistakes and improving overall code quality. Lint tools can help you speed up development by detecting problems early.

2.11 Testing

Testing is a process within software development that verifies the correctness, quality, and performance of software. Software testing ensures that systems and product features preform as intended. Testing can be done manually or automatically. A team or individual leads manual software testing, which involves operating a software product to verify it operates as intended. Many different technologies are used in automated testing, with capabilities ranging from isolated code quality checks to emulating a full human-driven manual testing experience. [6]

2.11.1 Unit Testing

A unit test is a method of testing a unit, which is the smallest amount of code in a system that can be logically separated. In most programming languages, this is referred to as a function, a subroutine, a method, or a property. [27]

2.11.2 Behavior-driven Development

Behavior-driven development (BDD) is an Agile software development methodology in which an application is defined and developed around the expected behaviour of the user while interacting with it. BDD helps to minimize bloat, extra coda, unneeded features, and lack of focus by pushing developers to focus exclusively on the intended behaviors of an app or program. [13]

2.12 Software Project Methodologies

To better design and product management, software projects are typically broken into smaller, parallel, or sequential steps or sub-processes. The approach may include the pre-definition of certain deliverables and artifacts that a project team creates and completes in order to develop or maintain an application. SCRUM is a widely used approach for organizing software development projects nowadays. [1]

2.12.1 SCRUM

Scrum encourages teams to learn by doing, to self-organize while working on a challenge, and to reflect on their successes and mistakes in order to continually improve. Although scrum is most commonly utilized by software development teams, its ideas and lessons may be applied to any type of team. One of the reasons scrum is so popular is because of this. Scrum is a combination of meetings, tools, and responsibilities that work together to help teams structure and manage their work. It is sometimes thought of as an agile project management framework. [8] [25] [24]

2.12.2 SCRUM Roles

SCRUM has three roles: product owner, SCRUM master, and members of the development team. Because the essence of scrum is empiricism, self-organization, and continuous improvement, the three roles define a minimum responsibilities and accountability to allow teams to deliver work successfully. This enables teams to take accountability for how they arrange themselves and to continually improve. [5]

Development Team
The development team are the people in the Scrum Team that are committed to creating any aspect of a usable Increment each Sprint. [25]

Product Owner
The Product Owner is responsible for maximizing the value of the product created by the Scrum Team. How this is done varies greatly between organizations, Scrum Teams, and people. [25]
SCRUM Master
The Scrum Master is responsible for implementing Scrum methodology. They accomplish this by assisting everyone, both inside the Scrum Team and throughout the business, in understanding Scrum philosophy and practice. [25]

2.12.3 SCRUM Events

Sprint
A sprint is a short, time-boxed period when a scrum team works to complete a set amount of work. [25]
Stand-up
The daily scrum meeting is not used to solve problems or resolve issues. Each team member answers the following three questions at the daily scrum:

1. What did you do the day before?
2. What are your plans for today?
3. Are there any roadblocks on your path?

The team develops a great knowledge of what work has been completed and what work remains by concentrating on what each individual performed yesterday and will do today. [28]
Sprint Planning
Sprint planning is an event in scrum that kicks off the sprint. The purpose of sprint planning is to define what can be delivered in the sprint and how that work will be achieved. Sprint planning is done in collaboration with the whole scrum team. [7]
Sprint Review
A sprint review is an opportunity to showcase the efforts of the whole team, including designers, developers, and the product owner. [4]
Sprint Retrospective
Retrospectives provide an opportunity for the agile team to evaluate itself and prepare for future areas of improvement. By stepping outside the work cycle to reflect on the past, the retrospective embraces the concept of continual progress while protecting against the traps of complacency. [3]

3 Method

3.1 Research method

Software developers tend to intuitively follow a scientific approach when writing code, usually without realising it. When solving a problem, the first important step is trying to understand what question you are trying to answer. [26]

Then you must express the hypothesis that may provide an answer to the question. There will always be several hypotheses, so use your expertise to select the most plausible one. It doesn't matter if it turns out to be incorrect; you will ultimately uncover the correct solution.

Then predict the outcome after you have developed your hypothesis. This step will tell you if your experiment was successful or not. Once you know what to expect, do an experiment to put your hypothesis to the test. This is the step at which you write code.

Finally, the computer will respond with some results: it will perform what you want, or it may display an error message or nothing at all. Even the lack of a result is an outcome that should be investigated. When an error occurs, it is critical to thoroughly study the error message and pause to consider what caused it. And if you don't receive the expected result, you must consider why and develop a better hypothesis. [26] [39]

3.2 Project management

3.2.1 Group

The group acts as the contractor in this project. The group consists of four computer science students from NTNU in Ålesund: Apimanju Rajan, Herman Nauf, Mathias Hagen and Ulrik Eikås.

3.2.2 Responsibilities

We drafted a collaboration agreement before we started working on the project, in which we organized the roles for each member. The scrum master is Mathias, the document manager is Ulrik, the quality assurance manager is Apimanju, and the communications manager is Herman. These roles were meant for everyone to have an overarching role, without hindering the contribution to the project work.

3.2.3 Supervisor

The supervisor's role in the project is to supervise the progress and assist the group with technical and theoretical knowledge. The supervisor for this project is Di Wu, who is a associate professor at NTNU in Ålesund.

3.2.4 Client

iSi AS is the client of this project, they will receive the final product from the group. Ida Kjørholt and Nils Tarjei Hjelme are the representatives from iSi AS.

3.2.5 Planning

The initial stage of the project was planning. A roadmap was developed and agreed with the client to ensure everyone was on the same page. Before coding, a wireframe of the tool's initial draft was also produced. The wireframe can be found at page 57 The group then decided to hold stand-up meetings every morning at 9:00 to discuss challenges and time management. Following the standup, the group worked until 15:00 before wrapping up the work day by reviewing the project's progress.

3.2.6 Requirement specification

The end-user of our application is primarily the road owners and the maintenance contractors. The most central road owners and users of our application will be The Norwegian Public Roads Administration. The requirement specifications has been created after many workshops and conversations between the end-user and iSi AS, and is as follows.

- To be able to present an overview over types, faults and criticality on the inspected area in a web-based map
- To be able to filter over types and faults in the railway, as well as the criticality on a road stretch
- To be able to look closer at concrete types and faults with the help of images and metadata (Be able to click between different areas on the map, and change image display between upper and lower camera.
- To be able to display different level of detail/data based on zoom and section on the map.

3.2.7 Sprints

The sprints lasted for two weeks. This was decided based on the fact that we were having a parallel subject, which held back progression. The bi-weekly sprint meetings persisted after the course was finished due to the fact that meetings were more profound as a result of having more to discuss.

We held our review and retrospective on the last Friday of the sprint. The group, the client and the supervisor participated in the review, while the retrospective was only for the group. The time frame for the review varied based on the progress that was done in the sprint.

3.2.8 Jira

To organize the sprints we decided to use Jira. Jira helps the group stay organized showing everyone which tasks needs to be done in the sprint, who is working on which tasks at the moment. In addition to these features, Jira also provides a backlog where we store all the issues, a road map for the timeline of the whole project, and lastly many different types of reports such as ”Sprint Burndown-diagram” and ”Burnup diagram” to help us understand and improve our productivity and progress.

3.2.9 Stand-ups

Stand-ups were held on the start of each workday. In the stand-up the team could discuss problems team-members had, or decide if there were some tasks that required more attention. The stand-ups were held in a Discord call or physically at the campus.

3.2.10 Backlog

The backlog contains any features, bugs or other tasks reported as issues on Jira. The group can then move issues from the backlog to a sprint.

3.2.11 Bitbucket

Bitbucket is an Atlassian application which allows for complete code and version control. Bitbucket manages Git repositories and improves code sharing, making life easier for development. It integrates well with Jira, which can link issues to pull requests and branches.

3.3 Tools

3.3.1 Teams

Teams is a platform for communication and collaboration, and provides support for video-meetings, chat between members and file-storage. The group have used this to communicate and arrange meetings with the supervisor and client.

3.3.2 Overleaf

Overleaf is a cloud-based $\LaTeX$-editor for writing and publishing scientific documents. It can provide countless templates, and endless customization of the document-structure. Overleaf is the editor this thesis is written in.

3.3.3 Postman

Postman is a platform for building and using APIs. The platform allows the user to write requests to a given server, and study the response given back in an easy way.

3.3.4 Visual Studio Code

Visual Studio Code is a source-code editor for a range of programming languages. It comes with minimal support for the majority of popular programming languages out of the box. Syntax highlighting, bracket matching, code folding, and customisable snippets are all part of the basic functionality. Visual Studio Code additionally includes IntelliSense for JavaScript, TypeScipt, JSON, CSS, and HTML, as well as Node.js debugging support. Additional language support is accessible through freely downloadable extensions on the VS Code Marketplace. Since it supports IntelliSense and code navigation for Angular projects out of the box and many great extensions, Visual Studio Code is a great choice when working with Angular and other web development projects.

3.3.5 Prettier

Prettier is a code formatter which enforces a consistent style defined by a ruleset. Prettier supports languages like TypeScipt, HTML, SCSS and most of the other languages used in web-development.

3.3.6 ESLint

ESLint is a static code analysis tool that detects problematic patterns in JavaScript and TypeScript code. It may be customized to provide warnings or even compilation failures when linting rule sets are not followed. ESLint can also be set up to automatically correct the majority of problems when you save your work.

3.3.7 Compodoc

Compodoc is a documentation tool for Angular applications. It provides clear and useful documentation for the application in a simple way. This makes it easier for the team to understand the code, as well as future developers understand the features of the application.

3.4 Backend

3.4.1 Deployment

To deploy the web-application we used Bitbucket pipelines. Pipelines is Bitbucket's CI/CD tool and allows for automation of some code. The code for the pipeline itself can be found at the following listing: **Pipeline Code**. The pipeline goes through the following steps:

- NPM Installation
  - Used to be able to build our application on the remote server
- Lint
  - Code analysis tool
- Testing
- Build
  - Builds the application, and uses the build files in the next step
- Deploy
  - Opens an SSH connection to our remote server, and updates the old files with new ones.

To enable the pipeline, we are required to use key pairs to connect using SSH. This is sensitive data that should not be visible to other people. Sensitive information like this is stored in Bitbucket directly, where only administrators of the project have access.

3.4.2 OpenStack

OpenStack is a free cloud computing platform. Mostly used for Infrastructure as a Service (IaaS). OpenStack is the platform used to run the machine our website is hosted on.

3.4.3 NGINX

NGINX is a free open source web server. It can also be used as a reverse proxy, load balancer, mail proxy and HTTP cache. NGINX is simple to set up in order to serve static web content, and is therefore a popular choice of web server.

3.4.4 Server

The group was provided with a server from NTNU on OpenStack running Ubuntu Server 20.04 (Focal).

The server has 2 vCPUS, 4GB RAM and 40GB of disk space. The server is used as a web server for the application.

3.4.5 Amazon S3

Amazon S3 is a service that allows object-storage. An object is any file and metadata that describes the file. It provides scalability, security and performance. iSi uses this service to storage all their images.

3.4.6 MongoDB

MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas [36].

3.5 Frontend

3.5.1 Single Page Application

To achieve the requirements above we have created a Single-page application (SPA). The main purpose with this method is so that the web app only loads a single web document, and then updates the body content of that single document using TypeScript APIs such as HttpRequest when different content is to be shown. By doing it this way we can achieve performance gains and a more dynamic experience.

3.5.2 NodeJS

NodeJS is a open source cross-platform server environment. NodeJS lets the team develop by running scripts server-side to produce the web page content locally.

3.5.3 Leaflet

Leaflet is an open source JS-library for interactive maps. It is the most central library in this project, as almost every feature depends on the map itself. Leaflet provides functions and features to customize the way the map is displayed, and you can for example add geometry-shaped outlines to highlight specific parts of the map.

3.5.4 D3.js

D3.js is a JavaScript library for data-driven document manipulation. D3 allows you to visualize data by utilizing HTML, SVG, and CSS. D3 is pretty fast, with minimum overhead, and it supports large datasets as well as dynamic behaviors for interaction and animation. D3's emphasis on web standards offers you complete access to contemporary browser features without tying you to a proprietary framework, combining sophisticated visualization components with a data-driven approach to DOM manipulation.

3.5.5 Angular

To develop the application in Angular was a requirement from the client. Angular is a platform and framework that allows you to create single-page client applications in HTML and TypeScript. It provides essential and optional functionality as a collection of TypeScript libraries that you can incorporate into your applications.

The code-snippets below showcases the difference between TypeScript and JavaScript, and what the result of each action will be [2].

```
var myFunc = function(userObj){
    console.log(userObj.FirstName); // display first name
}

var user = {
    FirstName: 'Alex'
};
myFunc(user); // FirstName is displayed

var stringName = 'Alex';
myFunc(stringName); // undefined is displayed
```

Code example 2: JavaScript example

```
class User {
    FirstName: string
    constructor (fName: string) {
         this.FirstName = fName;
     }
}

// expecting a variable of type User
var myFunc = function(userObj: User) {
console.log(userObj.FirstName);
}

var user = new User('Alex');
myFunc(user); // FirstName is displayed

var stringName = 'Alex';
myFunc(stringName); // compilation error
```

Code example 3: TypeScript example

3.5.6 Unit testing

Unit tests were written for the components and services of the Angular application. These tests would verify the basic functionality of the unit under test. The tests were run in the local development environment and on main branch commits.

```
it('should call getStretchById and return stretch details', (done: DoneFn) => {
   httpSpy.get.and.returnValue(of(mockStretch));

   backendService.getStretchById('62189309f77014463c4f70d3').subscribe({
     next: (data) => {
       expect(data).withContext('expected stretch').toEqual(mockStretch);
       done();
     },
     error: done.fail,
   });

   expect(httpSpy.get.calls.count()).withContext('one call').toBe(1);
 });
```

Code example 4: Example unit test

3.5.7 Jasmine

Jasmine is a popular JavaScript Behavior-driven development unit testing framework that can test both synchronous and asynchronous JavaScript programs. It attempts to describe tests in a human-readable language so that non-technical people may comprehend what is being tested. Even if you are a technical person, reviewing tests in Jasmine makes it much easier to grasp what is going on. Jasmine is already configured when creating a new Angular project with the Angular CLI, and was therefore an easy choice.

3.5.8 Karma

Manually executing Jasmine by refreshing a browser tab in different browsers every time we change some code might get tedious. Karma is a command-line tool that launch browsers and execute Jasmine tests inside of them. The test results are also presented on the command line. Karma may also monitor your development files for changes and automatically re-run tests. Karma enables us to execute Jasmine tests as part of a development tool chain that requires tests to be runnable and results to be inspected from the command line.

4 Results

4.1 Scientific results

While attempting to build the interactive map, we sometimes ran into issues and bugs. These issues and bugs were not obvious most of the time, and the way we proceeded to solve them could resemble the scientific method. Following the scientific method exactly was not always needed, however there are steps that were more relevant than others. Specifically asking a question as to why a bug happened, then proceed to construct an idea of what could resolve it. Once we had an idea of what could cause the issue, we proceeded to test it. If it failed we would construct another idea, and test it again. If it was successful, the bug was gone.

The scientific method was also employed in the web page design process. It would always begin with the question, ”How can we present this function in an easy-to-understand manner without restricting it?” Then, like in the scientific method example above, continue the procedures.

4.2 Engineering result

4.2.1 Goals

The goals as mentioned previously are

- Introduction to the data, and the corresponding APIs
- Agree to a framework.
- Develop a web-based map-solution for displaying and filtering of data.
- Show images in the map, with functionality to scroll through images, and choose the upper or lower camera.
- Look at the possibility of classifying sections.

Goal 1 - Over time, all group members got increasingly comfortable reading the data provided to us by iSi and executing the API-calls to provide relevant data to the user. We mainly used two APIs, National Roads Database, and iSi’s own API.

Goal 2 - The framework iSi suggested using was Angular. We decided to choose Angular as they had made previous web applications using Angular, and to challenge our self.

Goal 3 - Our web-based map-solution meets the specification, and according to iSi some expectations were exceeded. We were given a very open specification, that allowed us to reflect upon what we saw as a useful and user-friendly web-site while still maintaining important functionality. The website got continuously improved after hearing feedback from iSi and our supervisor.

Goal 4 - Showing the images, scrolling through them, and switching angle is one of the main features of the web-application. This is how the user will be able to take a closer look at the railing and get an overview of the state of the railing. It was time consuming and difficult at times to make these functions quick and reliable.

Goal 5 - Classifying the sections means distinguishing the stretches. This was implemented by giving each stretch on the map a specific color correlated to the severity or criticality of each stretch. In our final solution this is based on the average height of the railing. Creating an advanced calculation of the criticality is hard, and there are many values we could base this calculation on, however for performance's sake we decided to leave it at average height.

4.2.2 User testing results

When conducting user testing, we wanted to mainly focus on two things. We wanted the product to be both user friendly and relevant for the main users of the interactive map. Testing the user-friendliness has been done throughout the entire process, by getting feedback from peers and both iSi and our supervisor.

We have conducted structured tests specifically testing user experience, and the feedback we received from these tests were mostly positive while still containing ideas for improvement.

To specifically test the relevancy of the product we wanted to test it directly with the road owners. Early on there was discussions with iSi regarding conducting tests with The Norwegian Public Roads Administration, and they have received a link to the website.

4.3 Administrative results

While working on the project, the Scrum approach was employed throughout the entire process. A meeting with iSi was held at the beginning of the process. It was agreed to have sprints every two weeks. At the end of each sprint, we met with iSi and our supervisor to discuss the goals of that sprint, what had been accomplished, a quick demo, and finally feedback on what had been done. Following the meeting, the group held a sprint retrospective, during which we reviewed a few aspects regarding how the sprint went. The project manual attachment contains all of the reports, reviews, retrospectives, and meeting notes.

At the start of each sprint, the group met and added as many items from the backlog as they deemed were necessary for the sprint. Then each participant choose the topics they wanted to work on. The time sheet for each sprint, detailing who worked on which tasks, can be found in the project manual attachment.

4.4 Performance

4.4.1 Rendering

Leaflet renders the map in fixed size tiles, tiles that are outside of your current view will not be rendered. This means that even if there are a large amount of polygons or polylines present on the map, the performance will not necessarily go down. While testing the web-application, performance has not been an issue.

4.4.2 API

Currently most of the APIs are fast, the user rarely has to wait for the calls returning information. When waiting for API-responses the user will see a loading-circle to let them know they have to wait for some data to arrive.

4.4.3 Functions

The most time-consuming function we currently have is the function responsible for creating the polyline, and adding event-handlers to it. This process takes around 0.5 milliseconds (based on console.time()). The county with the most polylines is Vestland with 280 stretches, meaning this function takes around 140 milliseconds in total to complete.

5 Discussion

5.1 Scientific results

The main difference between an interactive map and a static map is the possibility of the user to click, pan and zoom. As well as allowing the user to show location-specific data, and the use of layers to display multiple data sets [10].

The use of an interactive map allows for easy demonstration of how an issue affects different geographic areas, and the nature and distribution of a problem is made clearer by using different map layers to give new insights and comparisons [17].

In the approach of designing the web page, the keywords usability and user-friendly was always lingering in the back.

5.2 Engineering result

5.2.1 Goals

Goal 1 - To be able to provide a web-based interactive map solution that presents iSi's data in a user-friendly and easy-to-understand format. The group needed to comprehend iSi's data as well as other APIs that may assist us in displaying it.

Goal 2 - The team has no prior experience with Angular. This is the first time we've used this framework to build a working application. Therefore, it was decided to dedicate the first sprint to learn as much as possible about Angular and Typescript. One sprint was not enough to learn a new framework, so we had to try and fail, and improve along the way. While getting into Angular was quite hard at first, the results started to show after a while, although the team probably did not have enough background to utilize Angular to its full potential.

Goal 3 - As mentioned above, the requirement specifications provided was very open. This required reflection upon designs and functionality to decide how to solve problems and which functionality to add and remove. This was both a blessing and curse. The opportunity to decide upon which functionality to add, how it should work, and how much work should be put into this functionality, made it so much more fun and engaging working with the website. It also allowed for further thinking, rather than how to create that specific functionality. On the other end, it was really demanding to be able to display all the data in a website, since designing a responsive and interactive website is challenging. The designing process required several iterations and feedback from both supervisor and client before finalizing the design.

Goal 4 - The main issue faced when implementing an image view, with all the images in a stretch, was the speed of the API. It could take up to a few seconds before the displayed image was loaded, and all the images from the selected stretch would show. Many different solutions was tested as solutions to the problem, but it was not anything that could be fixed, so a loading wheel to display the delay was added.

Goal 5 - The implementation of criticality was to be able to segment each stretch; the reason for its simplicity is that if it were made more complex with many additional parameters, the solution's speed would suffer significantly as a result of having to make too many API-calls. A good solution to solve this problem is to let the back-end handle calculating the criticality, but since iSi did not provide access to their back-end this was the only possible solution.

5.2.2 Feedback

iSi provided feedback on the end product and the collaboration, which included discussions about some of the main features, the opportunity for us to demonstrate innovation, and our contribution to their final solution.

"The project description is relatively open, which gives students the opportunity to show both creativity, problem-solving and implementation skills. We have had regular sprint meetings and kept in touch using Teams during the assignment period, and we think the collaboration worked excellently." - One paragraph paraphrased from iSi's feedback found on page 75

In many aspects, the cooperation, discussions, and final product have inspired iSi. The graph that displays the height of railing in a specific stretch is an example of a feature in the final product that they found inspiration from and implemented in their solution.

Our approach went through several stages in which solutions were tested and either pursued, altered, or rejected entirely. An example would be the picture view, which first displayed the images vertically before switching to tab display based on client input.

According to iSi's response, they are satisfied with the outcome. They mention that we have been able to focus our main functionality on the needs of the end user, they also provided some instances of how we have done so. These, along with the remainder of iSi's feedback, may be seen on page **75**

5.2.3 User testing

After conducting user testing with focus on user-friendliness we received feedback which was mainly positive, with potential improvements. The general positive impressions of the website includes ease of use, intuitive navigation and no clutter. On the other hand, the users suggested a button in the info-window to be able to minimize or close the window.

We have received feedback regarding functionality. Most of the functionality has been implemented, but due to time constraints and other factors, such as the backend, we prioritized the most important functions.

It was not possible to complete the appropriate amount of user testing. With the support of iSi, a relationship with the Norwegian Public Roads Administration was formed with the purpose of having them user test the solution. Because the product was designed with The Norwegian Public Roads Administration as one of the key end users, the objective was to have them try out and test not only the usability of the product, but also the functionality. We supplied them the IP as soon as we had an MVP so that they could do at least two rounds of user testing before submitting. Unfortunately, their proxy servers prevented them from testing the website since unknown API requests were banned. We attempted to contact them several times to get feedback, but we have not received a response.

5.3 Administrative results

Despite the fact that sprint meetings were only held fortnightly, iSi was always available for a brief team chat or a larger meeting if it was essential for development. This made it simple for us to constantly acquire the help we needed when we got stuck and maintain strong progression throughout the project.

It was possible to add extra issues from the backlog during each sprint if anything was discovered during the sprint, such as a bug or an essential feature, or if all of the previous issues for that sprint were completed. If an issue could not be finished within the given sprint due to a lack of time or data, it was possible to transfer it out of that sprint and back to the backlog, or to carry it over to the next sprint to continue working on the issue.

5.4 Result

5.4.1 Performance

Currently, the maximum number of polylines visible on the Leaflet map is roughly 550. If speed becomes a concern, for example, when the number of polylines displayed dramatically grows, a solution would be to group up clusters of polylines when the user zooms out too far.

There is one API-call that takes some time from the API, the API-call to retrieve all stretches in a county, which is shown in the table **REST API resources**. The backend must query and return a huge quantity of data, which might cause this to be sluggish. This may be remedied by installing a cache on the backend or optimizing the query.

If the function responsible for drawing polylines needs to draw a large number of polylines, it can become sluggish. By combining the polylines into smaller groupings, such as municipalities, the time can be minimized. Another option is to decrease the quantity of data imported by converting GeoJSON to TopoJSON or to render items on the map using alternative ways.

5.4.2 Strength

The strengths of this solution is the ability to compare different types of data and to efficiently analyze the state of a guardrail. iSi AS provided access to a large data set, which we had to figure out how to use and visualize. By using an interactive map the user gets a lot of flexibility to display exactly the information the user is looking for. At first the map is empty so the user don't get a lot of irrelevant information just displayed on the screen, this will also make the web page load almost instantly when visited.

After searching for something in the search bar, the area on the map will be marked and the user will be taken to that area automatically. All the relevant stretches will be displayed with colors that indicate their criticality providing the user with a quick and clear overview. In the top right corner there are three buttons representing the three different criticalities, which now display the actual number of stretches in the different categories for that area. When the user presses this button, a list off all the stretches will be displayed and the user can then select a specific stretch and press go to map. The user will then be taken to that exact stretch and the information window will open in the bottom left. The user can also open the information window by pressing a stretch directly from the map. This will keep the map clean and allow the user to look around on the map until he finds the stretch he wants to inspect before clicking it and see more relevant information about the stretch.

Another strength is that when a stretch is selected, the user will know exactly where he is on the stretch at all times. When pressing a stretch for the first time, a marker will appear at the start of the stretch, in the information window the user will also see the first image on the stretch and the graph will show the exact height for that position with a marker. When the user start scrolling through the list of road references the marker on the map will move, the image will update, the marker on the graph will move and the information in the additional information box will also update. This is important so that all the information which is displayed to the user is relevant and accurate at all times when the user is performing a quality control.

The solution will also make it easy to filter the displayed stretches based on criticality, guardrail and pole type. The user can check and uncheck the different checkboxes that are relevant to the things they want to filter on and the interactive map will be updated immediately. It is also intuitive and designed with usability in focus. The solution displays as much information as possible on the web page without being cluttered.

5.4.3 Weaknesses

The main weakness of our solution was the lacking backend in the beginning. The first iteration of displaying lines on the map was done using local JSON-files extracted from iSi's database. Their API was also limited to supplying the metadata and images. This meant that some of the functions we created probably belongs backend and through their API. For example speed limit, traffic volume and some form of criticality calculation. To get the speed limit we need to make a call for each line on the map, which makes it hard to use in the calculation of the criticality, simply because there would be needed too many API-calls which makes our solution slow. The same applies to traffic volume.

Both speed limit and traffic volume is extracted from NVDB API. This API also provides the list of all counties, and the WKT to draw outlines of each county. The API used for all these functions also supply additional information that is not needed, and to construct an API-call that filters out the less needed information was quite complex. Some of these API-calls are therefore less efficient than they potentially could be, which results in longer waiting times in some cases. The speed of these calls are often negligible, and does not impact the solution as of now.

6 Societal impact

6.1 System perspective

This solution will have positive impact on both the environment and society in many ways. As mentioned earlier, today’s manual practice of controlling the railing is both time and resource consuming, at the same times as it increases the traffic danger along the road. By displaying all the data that iSi has collected using their rig in a good way on our single page application, we can give the road owners a much better overview of the state of the railings along the roads. Using the different functionality like for example the different filters, the user will be able to see which roads are the most critical and can from that information prioritize roads with a lot of faults to make those roads safer and improve the traffic safety. When the railing are inspected on our web page and then improved it will result in a positive societal impact in the way that it makes the roads safer. The health impact of this is significant because it can prevent fatalities in the traffic.

By using our solution, the quality control of railings will be a more cost-effective process. This because the road owners can sit in the office using the web page to carefully control the railings on their roads. Since all the data is on the web page they can look at it whenever they want, unlike todays solution where they have to sit in the car and inspect the railings while another person is driving the car in 10-15 km/h. It is no longer necessary to have two cars and three people with the new solution which will make it much more cost-effective and will save resources that can be used on other things.

This solution will also benefit the environment. It is no longer required to have two cars in the process of inspecting the railings because the one person inspecting the railings will now do it in the office. This means that the car’s emissions have already been cut in half. iSi will also look into the possibility of collecting all the data using an electric vehicle, which would make it possible to eliminate all the emissions of fossil fuels from driving.

6.2 Technology ethics

What if a fatal accident occurs because there was a critical fault that the machine learning models didn’t catch and therefore a wrong impression of the status of the railing is displayed in our web page. There are a lot of different people involved for this whole solution to work, the one collecting the data, the one who works on the machine learning models, the one calculating the criticality, the one displaying all the data and the one who uses the data for inspecting the railing and prioritize which railings need to be improved.

Since today’s manual inspection is done by humans this process won’t uncover 100% off the faults on the railings across the roads. Humans aren’t perfect and all humans makes mistakes so 100% won’t be achievable, the machine learning model won’t uncover 100% of all faults either. The goal is to make the process better than the manual inspection.

6.3 Sustainable development goals

The main benefit of this solution, is to increased traffic safety and better road infrastructure. This is relevant for UN's goal number 9, "Industry, innovation and infrastructure". After creating this web page to display all the data, we are involved to achieve this.

Another important goal in this project is the UN's goal number 8 "Promote sustained, inclusive and sustainable economic growth, full and productive employment and decent work for all". More specific the 8.2 "Achieve higher levels of economic productivity through diversification, technological upgrading and innovation, including through a focus on high-value added and labor-intensive sectors". Road work and infrastructure is an labor intensive sector and by streamlining the time and resource consuming process of quality control of railings by digitizing it, we can put off some workload in this sector with our solution.

UN's sustainable development goal number 17 says "Partnerships for the goals". To make this bachelor possible we have partnership with iSi and the Norwegian public roads administration. iSi as our employer and the Norwegian public roads administration as potential client for feedback and user testing.

7 Conclusion and future work

The primary purpose of this project was to create and deploy a user-friendly front-end web-based interactive map for displaying railings. In this report, we attempted to describe and defend the decisions we made in order to complete the work.

The team was able to create an interactive map by utilizing the Angular framework, TypeScript, Leaflet, and Node as the primary technology stack. The learning curve was high at times, but the learning outcomes were excellent. This is the group's first time using Angular, so the full potential of the framework may not have been realized, but it's essential to note that we attempted looking at, and implementing, the many capabilities the framework offered us.

One of the main takeaways from this project has been the importance of a structured SCRUM-team. The planned meetings and sprint reviews has been very useful for the progression of the web-site. Working organized with specific tasks managed in Jira has been essential to ensure both the quality and speed of which we were able to implement the different features.

7.1 Final product

The project has progressed beyond the Minimum Viable Product (MVP) stage. The project description was open, with not much detailed specifications, although the client still had a goal in mind for the progression. Therefore, recommendations to enhance the project were still included in all sprints. All in all the final project ended up with meeting all the criteria and even exceed some. At the end of the bachelor's program, a fully functional website with a lot of features was produced. Overall, everyone is pleased with the result.

7.2 Future work

Due to the time constraints, the most impactful features were prioritized. These aspects, as well as what they entail, will be outlined in the following sections, along with suggestions as to who could help develop the project further, as well as some pointers on where to begin.

7.2.1 Recommendations

To develop this project further it is recommended to establish a new repository with the most recent version. The following step would be to become acquainted with the technological stack. It's especially crucial to understand Angular, Leaflet, and how to work with geospatial data. It would also be necessary for them to become familiar with the API and datatypes supplied by iSi and NVDB, as they provide all of the data visualized in this project. According to Compodoc, 96% of our source code in this project is extensively documented, making it easy to understand and immediately begin developing on the solution.

To further develop the project without having to start from scratch, contact iSi, who owns the Bitbucket repository. They may grant access to the repository as well as other information needed to continue developing the project. Nils from iSi has been kept up to date on the progress of the website with each sprint, in addition to being knowledgeable with the entirety of the technological stack used in development.

For anyone interested in developing something similar or continuing to work on this project. We strongly suggest adopting SCRUM agile development for development, as well as Git and Bitbucket/GitHub for collaboration.

7.2.2 Potential

Despite the fact that the website is functional and provides the necessary information to provide useful insight to the state of each railing, it’s full potential has naturally not been reached. This is due to the fact that creating a production-ready web-site that is fully functional on all devices is a very time-consuming process, and it is out of the scope for this project. It’s hard to pinpoint exact features that *should* be added, however we have compiled a list of future features that could be useful, some more than others:

- Search for a specific road (e.g. E6)
  - This feature relied on the backend at which point was not ready
- Support for mobile and small tablet
  - The team agreed with the client and supervisor to focus on creating a website mainly for PC use
- Save searches (e.g. ”Recent searches” in the searchfield)
  - This is a suggestion that has not been brought up in any meetings, and was not prioritized due to its low-importance, however it could be a useful feature for frequent users of the website
- A way to display the time and date when the pictures and meta-data was updated
  - The suggestion was brought up very late in the process, and at the time it was agreed upon to just finish what was currently started, and consider the product at that time finished.

Bibliography

[1] Altitude Accelerator. *Agile Software Development Methodologies: Which to Choose?* URL: https://altitudeaccelerator.ca/agile-software-development-methodologies/. (accessed: 11.05.2022).

[2] Apriorit. *TypeScript vs JavaScript: What's the Difference?* URL: https://www.apriorit.com/dev-blog/700-web-typescript-vs-javascript. (accessed: 28.03.2022).

[3] Atlassian. *Agile Retrospectives*. URL: https://www.atlassian.com/agile/scrum/retrospectives. (accessed: 11.05.2022).

[4] Atlassian. *Agile sprint reviews*. URL: https://www.atlassian.com/agile/scrum/sprint-reviews. (accessed: 11.05.2022).

[5] Atlassian. *Scrum roles and the truth about job titles in scrum*. URL: https://www.atlassian.com/agile/scrum/roles. (accessed: 11.05.2022).

[6] Atlassian. *Software testing for continuous delivery*. URL: https://www.atlassian.com/continuous-delivery/software-testing. (accessed: 28.04.2022).

[7] Atlassian. *Sprint planning*. URL: https://www.atlassian.com/agile/scrum/sprint-planning. (accessed: 11.05.2022).

[8] Atlassian. *What is Scrum?* URL: https://www.atlassian.com/agile/scrum. (accessed: 11.05.2022).

[9] Eric Boersma. *Code Documentation: The Complete Beginner's Guide*. URL: https://blog.submain.com/code-documentation-the-complete-beginners-guide/. (accessed: 06.04.2022).

[10] Sam Cranwell. *How interactive maps can improve user experience on your site*. URL: https://www.versantus.co.uk/blog/how-interactive-maps-can-improve-user-experience-your-site. (accessed: 05.05.2022).

[11] Michael Dorman. *Introduction to Web Mapping*. URL: http://132.72.155.230:3838/js/geojson-1.html. (accessed: 09.05.2022).

[12] Henrik Dvergsdal. *HTTP*. URL: https://snl.no/HTTP. (accessed: 06.04.2022).

[13] Laura Fitzgibbons. *behavior-driven development (BDD)*. URL: https://www.techtarget.com/searchsoftwarequality/definition/Behavior-driven-development-BDD. (accessed: 28.04.2022).

[14] TerraExplorer Programmers Guide. *Well-Known Text and Well-Known Binary (WKT and WKB)*. URL: https://www.skylinesoft.com/KB_Resources/TED/Web%20Help/Programmers%20Guide/Well-Known_Text_and_Well-Known_Binary_WKT_and_WKB.htm. (accessed: 11.04.2022).

[15] Red Hat. *What is a REST API?* URL: https://www.redhat.com/en/topics/api/what-is-a-rest-api. (accessed: 06.04.2022).

[16] John R. Herring. 'OpenGIS® Implementation Standard for Geographic information - Simple feature access - Part 1: Common architecture'. In: 2 (2011), pp. 60–61.

[17] Alice Macfarlan. *Interactive Mapping*. URL: https://www.betterevaluation.org/en/evaluation-options/interactive_mapping. (accessed: 05.05.2022).

[18] Tom MacWright. *lon lat lon lat*. URL: https://macwright.com/lonlat/. (accessed: 11.04.2022).

[19] MDN. *JavaScript*. URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript. (accessed: 06.04.2022).

[20] MDN. *What is accessibility?* URL: https://developer.mozilla.org/en-US/docs/Learn/Accessibility/What_is_accessibility. (accessed: 06.04.2022).

[21] Microsoft. *TypeScript*. URL: https://www.typescriptlang.org/. (accessed: 07.04.2022).

[22] Toby Osbourn. *The Long-Term Impact of Great Code Documentation (with Examples)*. URL: https://textexpander.com/blog/code-documentation. (accessed: 06.04.2022).

[23] Refactoring.Guru. *Observer*. URL: https://refactoring.guru/design-patterns/observer. (accessed: 11.04.2022).

[24] Scrum.org. *WHAT IS SCRUM?* URL: https://www.scrum.org/resources/what-is-scrum?gclid=CjwKCAjwve2TBhByEiwAaktM1Nw-ziW0dWXKvY7j6ifWXG07w6eB_dNkiyC-m1wHwygUMspmbRkfuhoC__kQAvD_BwE. (accessed: 11.05.2022).

[25] ScrumGuides.org. *The 2020 Scrum GuideTM*. URL: https://scrumguides.org/scrum-guide.html#scrum-team. (accessed: 11.05.2022).

[26] Evgeny Shadchnevg. *Scientific method in programming*. URL: https://blog.makersacademy.com/scientific-method-in-programming-3b729c0b3fc3. (accessed: 11.05.2022).

[27] Smartbear. *What Is Unit Testing?* URL: https://smartbear.com/learn/automated-testing/what-is-unit-testing/. (accessed: 28.04.2022).

[28] Mountain Goat Software. *Daily Scrum Meeting*. URL: https://www.mountaingoatsoftware.com/agile/scrum/meetings/daily-scrum. (accessed: 11.05.2022).

[29] SourceMaking.com. *Design Patterns*. URL: https://sourcemaking.com/design_patterns. (accessed: 11.04.2022).

[30] Statens vegvesen. *Nullvisjonen*. URL: https://www.vegvesen.no/fag/fokusomrader/trafikksikkerhet/nullvisjonen/. (accessed: 06.05.2022).

[31] W3C. *Accessibility*. URL: https://www.w3.org/standards/webdesign/accessibility. (accessed: 06.04.2022).

[32] Wikipedia. *Accessibility*. URL: https://en.wikipedia.org/wiki/Accessibility. (accessed: 06.04.2022).

[33] Wikipedia. *CSS*. URL: https://en.wikipedia.org/wiki/CSS. (accessed: 06.04.2022).

[34] Wikipedia. *ISO 6709*. URL: https://en.wikipedia.org/wiki/ISO_6709. (accessed: 18.03.2022).

[35] Wikipedia. *JavaScript*. URL: https://no.wikipedia.org/wiki/JavaScript. (accessed: 06.04.2022).

[36] Wikipedia. *MongoDB*. URL: https://en.wikipedia.org/wiki/MongoDB. (accessed: 11.03.2022).

[37] Wikipedia. *Software design pattern*. URL: https://en.wikipedia.org/wiki/Software_design_pattern. (accessed: 11.04.2022).

[38] Wikipedia. *TypeScript*. URL: https://en.wikipedia.org/wiki/TypeScript. (accessed: 07.04.2022).

[39] Jason Yingling. *Using the Scientific Method in Web Development*. URL: https://jasonyingling.me/using-the-scientific-method-in-web-development/. (accessed: 11.05.2022).

Appendix

A Requirements documentation

A.1 Introduction

The purpose of this requirements specification is to elaborate on the scope and content of the bachelor thesis «Map display of road railings», which is a collaboration between iSi AS and four NTNU students in the spring of 2022. iSi AS provided an example of a service that was fairly comparable to the one they want, as you can see on Figure 4.

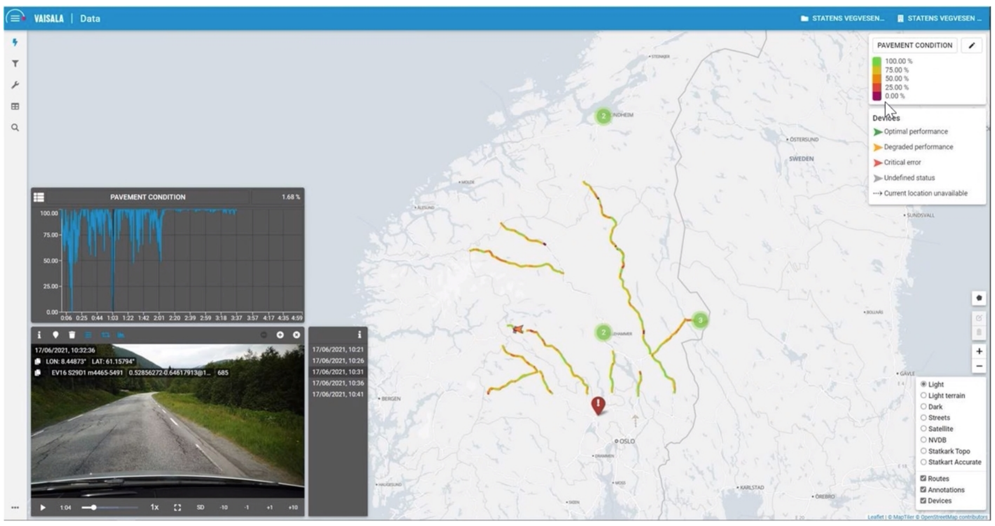


Figure 4: Picture of a similar solution

A.2 Objectives of the project - Users and user needs

The end users of the map display for road railings are mainly road owners and operating contractors. The most central road owner and user of the solution is the Norwegian Public Roads Administration. After pilot projects, workshops and conversations with them, iSi AS has mapped the following user needs and wishes for map view:

- To be presented with an overview of the type, error and criticality of the inspected areas in a online map
- To be able to filter in the map on the type and errors of the road barrier, as well as the criticality of road sections
- To be able to look more closely at specific types and errors using images and metadata (among others, options to click back and forth in the map and switch image display between upper and lower camera)
- To be able to display different detail levels / data based on zoom level and section in map

A.3 General decisions and delimitation

The solution that the Bachelor students will develop is limited to a map and images displaying the railing data. For example, it will not be necessary to develop the feature to generate reports or filters other than type, error and criticality. Any ambiguities in connection with the delimitation can be discussed by the client and the Bachelor students at start-up and during the project.

A.4 Map

To display the group were advised to use Leaflet, an open source JavaScript-library. This is a lightweight library for interactive maps.

A.5 Search and filter

A window for searching and filtering of the data the application displayed was required. It's main filtering options will be *type*, *fault* and *criticality*.

A.6 Graph displaying height levels on a given stretch

Displaying a graph to better visualize some data has proved valuable. It lets the user get a quick look at some of the data for a road in it's entirety.

A.7 Choice of graph-type

The graph-type used in the application is a line graph, which is used to display information that changes continuously over time, or in this case distance.

Within this graph the group have added some colors to display the appropriate height level of the railings. The accepted height of railings is currently between 65 centimetres and 80 centimetres. Anything below or above respectively is considered a potential hazard, and should be remedied.

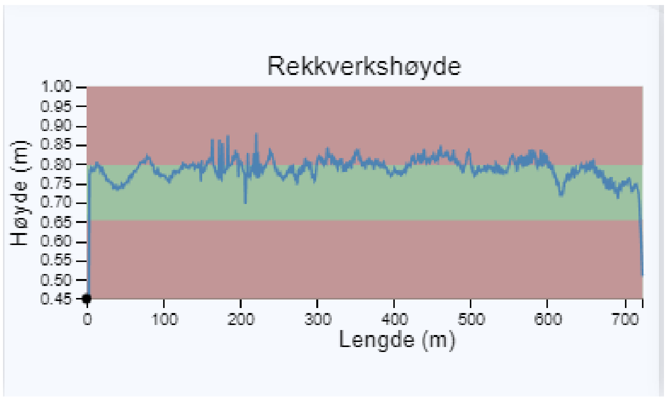


Figure 5: Graph display as of 28/03/2022

A.8 Data

The data the application displays in the graph is the height of the road at a given point, with 1 meter intervals. On a stretch that is 100 meters, there will be 100 points on the graph.

A.9 Image display

When showing the images, it must cover two distinct perspectives. One from the mounted rig's left side and one from the top. Initially, these images were placed on the top of each other, but we realized that it took up far too much space on the screen, so tabs were added to switch between the two images.

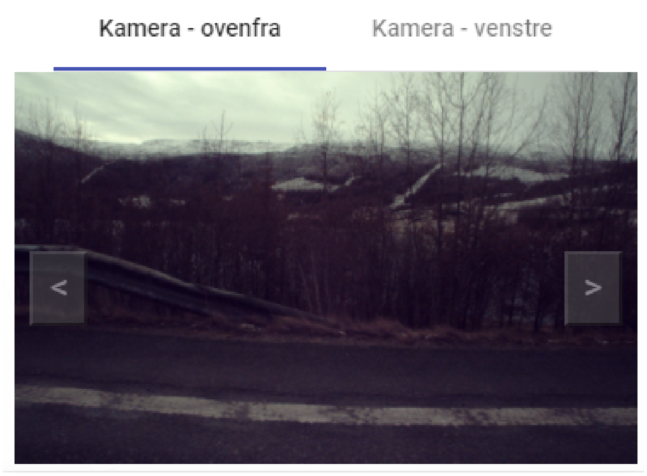


Figure 6: Image display as of 28/03/2022

A.10 Criticality

The total criticality is determined by a variety of factors, such as the average height of the railing, the speed limit, the volume of traffic and much more. It is estimated in this mannner to provide the user with a broad picture of which roads are the most critical and should be prioritized.

A.11 Use Case diagram

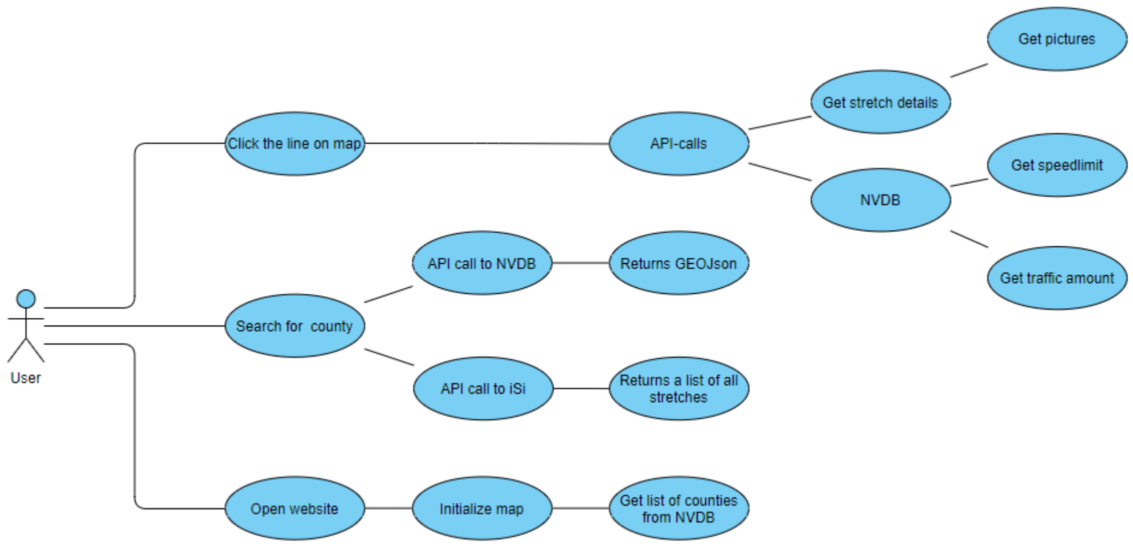


Figure 7: Use case diagram

A.12 User Stories

This is user stories from the client perspective.

As a user, I want to click on a stretch so that I can get more information about the selected stretch.

As a user, I want to be able to search for an specific area like for example county so that I can see all the stretches within that area.

As a user, I want to see the images from all the angels so that I can inspect the railings properly and be able to find visible fault.

As a user, I want to filter the displayed stretches based on criticality, guardrail and pole type so that I can easily find the stretches that I want to inspect.

As a user, I want to see the height of the railing on the entire stretch when it is selected so that I can get a quick overview to see if and where it is above or below accepted height.

A.13 Sequence diagrams

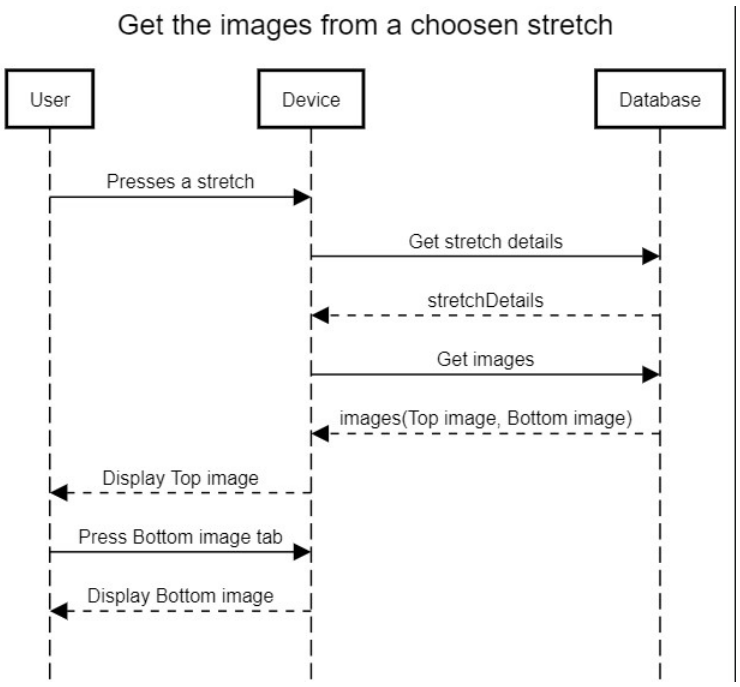
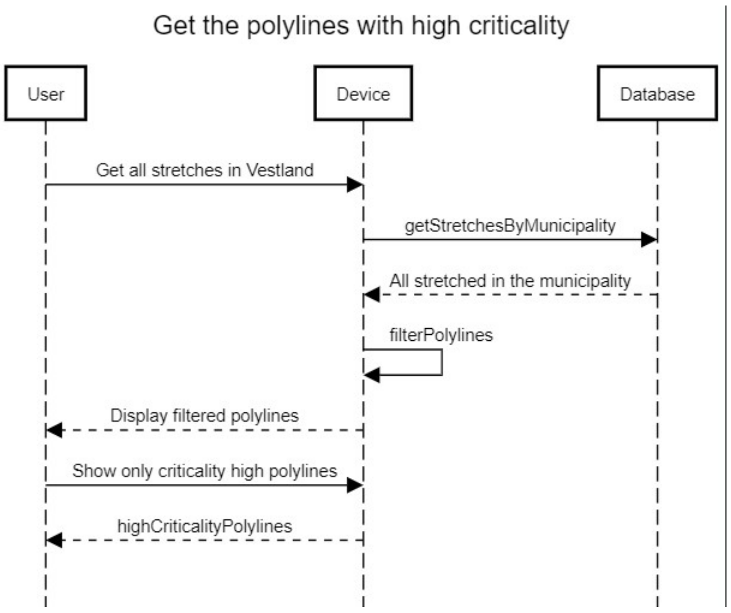


Figure 8: Get images from both angles



Get the polylines with high criticality
User
Device
Database
Get all stretches in Vestland
getStretchesByMunicipality
All stretched in the municipality
filterPolylines
Display filtered polylines
Show only criticality high polylines
highCriticalityPolylines

Figure 9: Filter on criticality high

A.14 Prototypes

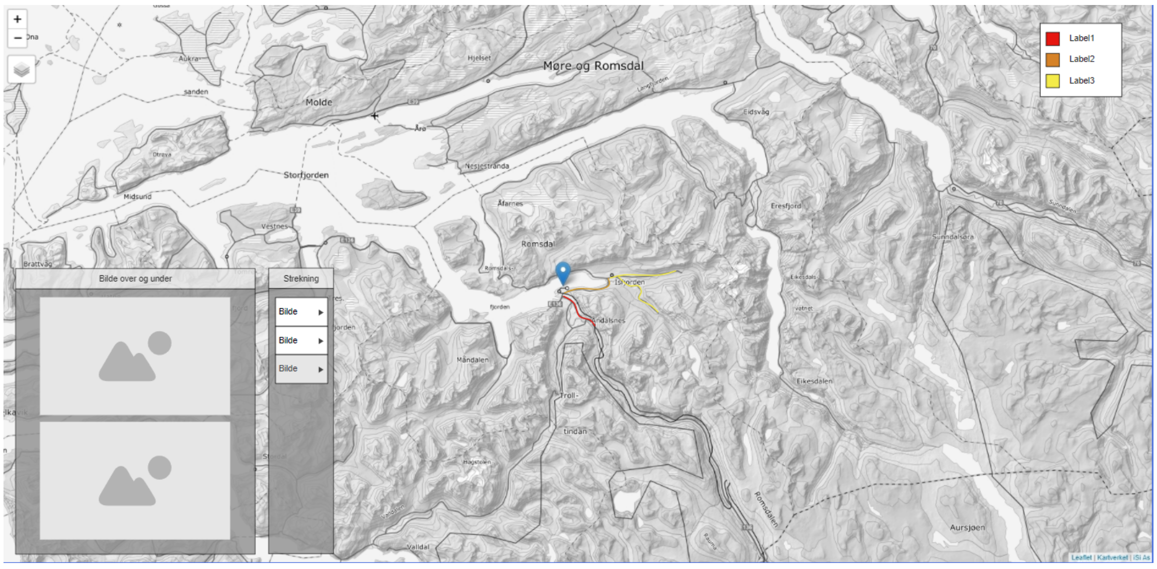


Figure 10: Wireframe

B System documentation

B.1 Introduction

This document will describe the technical aspect of the system. It will also describe the setup and installation steps to get the system up and running and ready to work.

B.2 Architecture

This application's core architecture is divided into three sections. The backend and Amazon S3 storage are provided by iSi AS. With a NodeJS REST API and a MongoDB database serve as the backend. All of the images, which are delivered as links through the REST API, are stored in Amazon S3. The last component is the Angular application's webserver. NTNU hosts this server on OpenStack. Furthermore, the system makes use of some of the resources supplied by the NVDB API, which delivers data from the Norwegian National Road-Database.

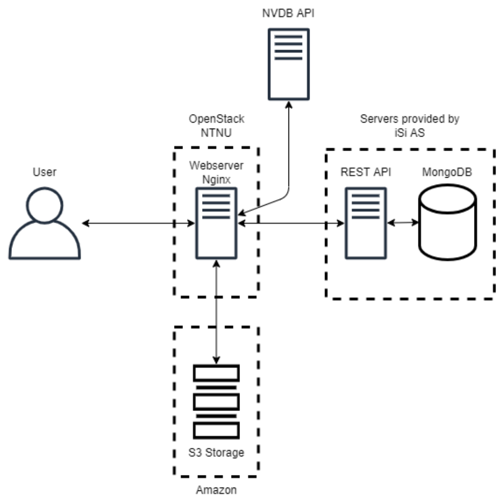


Figure 11: Architecture

B.3 Project structure

The workspace's top level contains workspace-wide configuration files, root-level application configuration files, and subfolders for root-level application source and test files.

Workspace configuration files	Purpose
.editorconfig	Configuration for code editors.
.gitignore	Specifies intentionally untracked files that Git should ignore.
README.md	Introductory documentation for the root application.
angular.json	CLI configuration defaults for all projects in the workspace, including configuration options for build, serve, and test tools that the CLI uses, such as Karma, and Protractor.
package.json	Configures npm package dependencies that are available to all projects in the workspace.
package-lock.json	Provides version information for all packages installed into node_modules by the npm client.
src/	Source files for the root-level application project.
node_modules/	Provides npm packages to the entire workspace. Workspace-wide node_modules dependencies are visible to all projects.
tsconfig.json	The base TypeScript configuration for projects in the workspace. All other configuration files inherit from this base file.

Table 2: Explanation of workspace configuration files

The source files (application logic, data, and assets) for the root application are stored in the workspace's src subfolder. The files at the top level of src/ help you test and run your application. The application source and application-specific configuration are stored in subfolders.

Application support files	Purpose
app/	Contains the component files in which your application logic and data are defined.
assets/	Contains image and other asset files to be copied as-is when you build your application.
environments/	Contains build configuration options for particular target environments. By default there is an unnamed standard development environment and a production (”prod”) environment. You can define additional target environment configurations.
favicon.ico	An icon to use for this application in the bookmark bar.
index.html	The main HTML page that is served when someone visits your site. The CLI automatically adds all JavaScript and CSS files when building your app, so you typically don’t need to add any <script> or <link> tags here manually.
main.ts	The main entry point for your application. Compiles the application with the JIT compiler and bootstraps the application’s root module (AppModule) to run in the browser. You can also use the AOT compiler without changing any code by appending the –aot flag to the CLI build and serve commands.
polyfills.ts	Provides polyfill scripts for browser support.
styles.sass	Lists CSS files that supply styles for a project. The extension reflects the style preprocessor you have configured for the project.
test.ts	The main entry point for your unit tests, with some Angular-specific configuration. You don’t typically need to edit this file.

Table 3: Explanation of application support files

The app folder, which is located within the src folder, contains the logic and data for your project. This is where you’ll find Angular components, templates, and styles.

src/app/ files	Purpose
app/app.component.ts	Defines the logic for the application's root component, named AppComponent. The view associated with this root component becomes the root of the view hierarchy as you add components and services to your application.
app/app.component.html	Defines the HTML template associated with the root AppComponent.
app/app.component.css	Defines the base CSS stylesheet for the root AppComponent.
app/app.component.spec.ts	Defines a unit test for the root AppComponent.
app/app.module.ts	Defines the root module, named AppModule, that tells Angular how to assemble the application. Initially declares only the AppComponent. As you add more components to the app, they must be declared here.

Table 4: Explanation of the Angular components, templates and styling files.

The root application's application-specific configuration files are located at the workspace root level.

Application-specific configuration files	Purpose
.browserslistrc	Configures sharing of target browsers and Node.js versions among various front-end tools.
karma.conf.js	Application-specific Karma configuration.
tsconfig.app.json	Application-specific TypeScript configuration, including TypeScript and Angular template compiler options.
tsconfig.spec.json	TypeScript configuration for the application tests.

Table 5: Explanation of application-specific configuration files

More information about Angular specific workspace and project structure can be found in the **official Angular documentation**.

In Figure 12 you can see a diagram of how the folder structure looks like with the current modules and components.

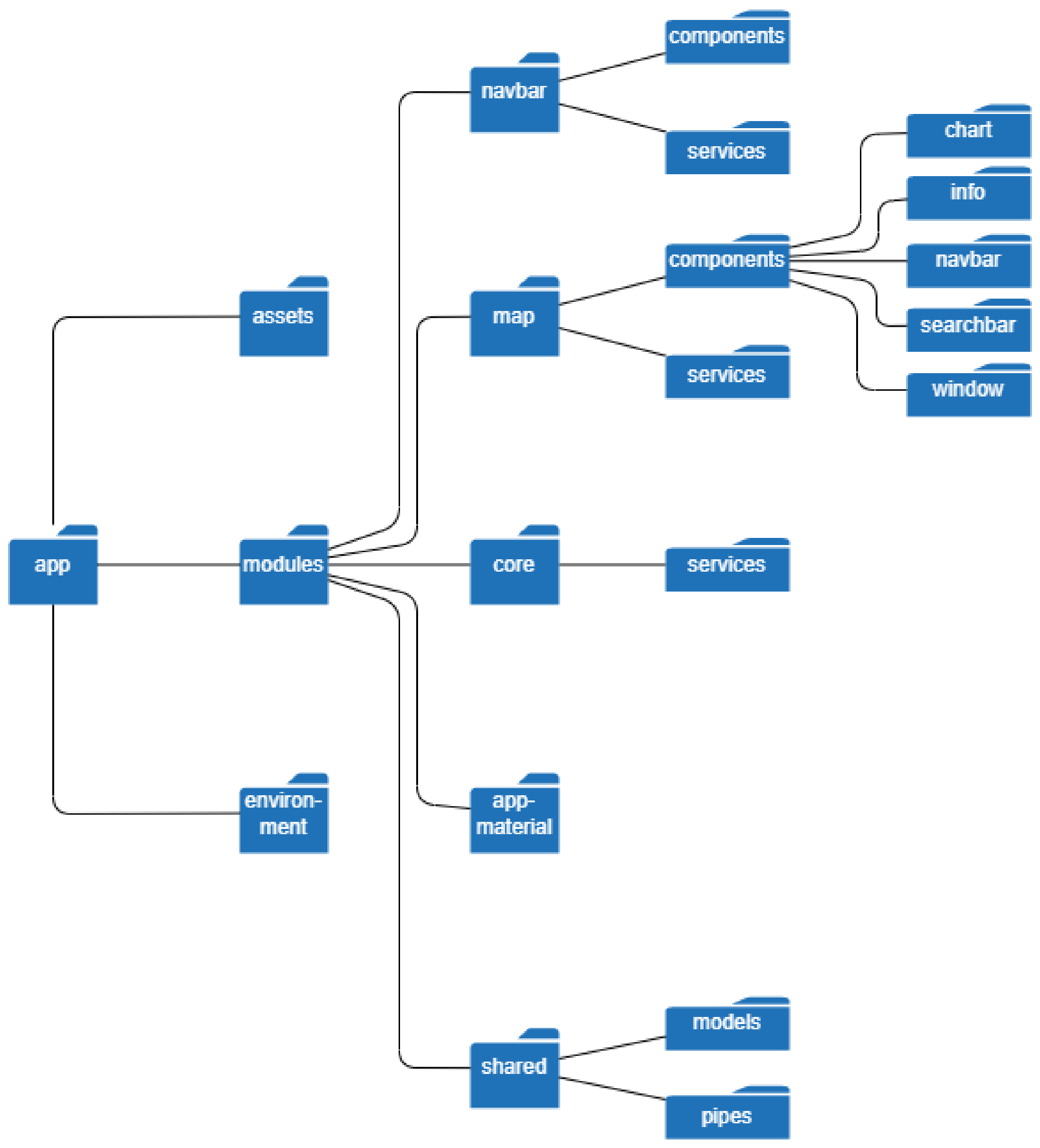


Figure 12: Folder structure

B.4 Module diagrams

The module diagrams diagrams describe how the different modules depend on each other. It also describes what the modules contains. The App component is the root component of the application.

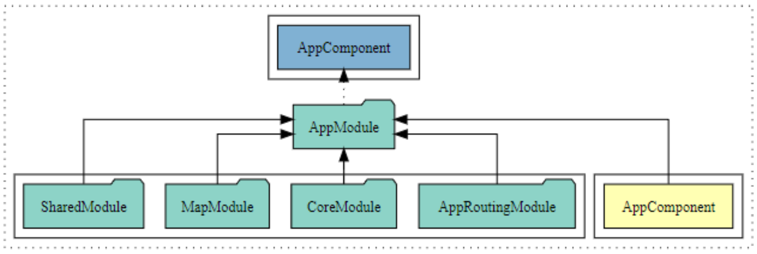


Figure 13: App module diagram

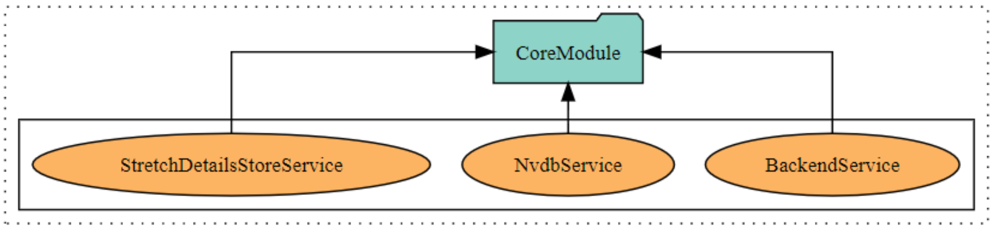


Figure 14: Core module diagram

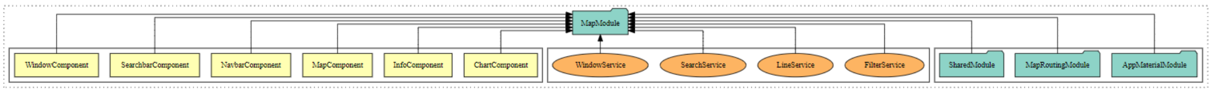


Figure 15: Map module diagram

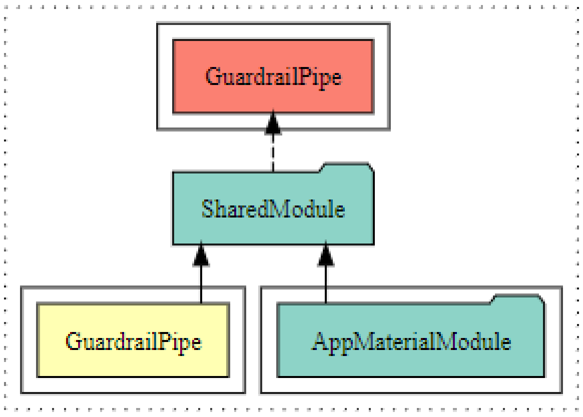


Figure 16: Shared module diagram

B.5 Server services

The REST API resources listed below is provided by iSi AS:

Path	/guardRails/fylke/{countyId}
Description	Get all guardrail stretches in the specified county
Method	GET
Path parameters	
countyId	Unique identifier of county.

Path	api/guardRails/{id}
Description	Get more details of the stretch
Method	GET
Path parameters	
id	Stretch id

Path	s3url/0548c9f8ecff/{name}
Description	Get image URL from the name of the image specified in stretch details
Method	GET
Path parameters	
name	Image name

Path	file?filename={name}
Description	Get 2D detection object from image name
Method	GET
Path parameters	
name	Image name

Table 6: REST API resources

Documentation of the NVDB API used in this project can be found on their website.

B.6 Installation

To run this project, you are required to have installed node, npm and Angular CLI on your system.

In the project folder, run ‘ng serve‘ for a dev server. Navigate to ‘http://localhost:4200/‘. The app will automatically reload if you change any of the source files.

Run ‘ng build‘ to build the project. The build artifacts will be stored in the ‘dist/‘ directory.

List of third party libraries used in the project:

Package name	Details
Leaflet	Leaflet is the leading open-source JavaScript library for mobile-friendly interactive maps.
d3.js	D3.js is a JavaScript library for producing dynamic, interactive data visualizations in web browsers.
wellknown	Wellknown is used to parse and stringify Well-Known Text into GeoJSON.

Table 7: Third party libraries

B.7 Testing

Unit tests are used to validate this project. Unit tests are used to validate the components and services. Jasmine and Karma are the testing frameworks utilized. The commands to run the tests can be found in Table 8.

Command	Purpose
npm test	Runs the tests in a Chrome browser.
npm run test:ci	Runs the tests in a headless Chrome browser. This is used in the deployment pipeline to run tests uninterrupted.

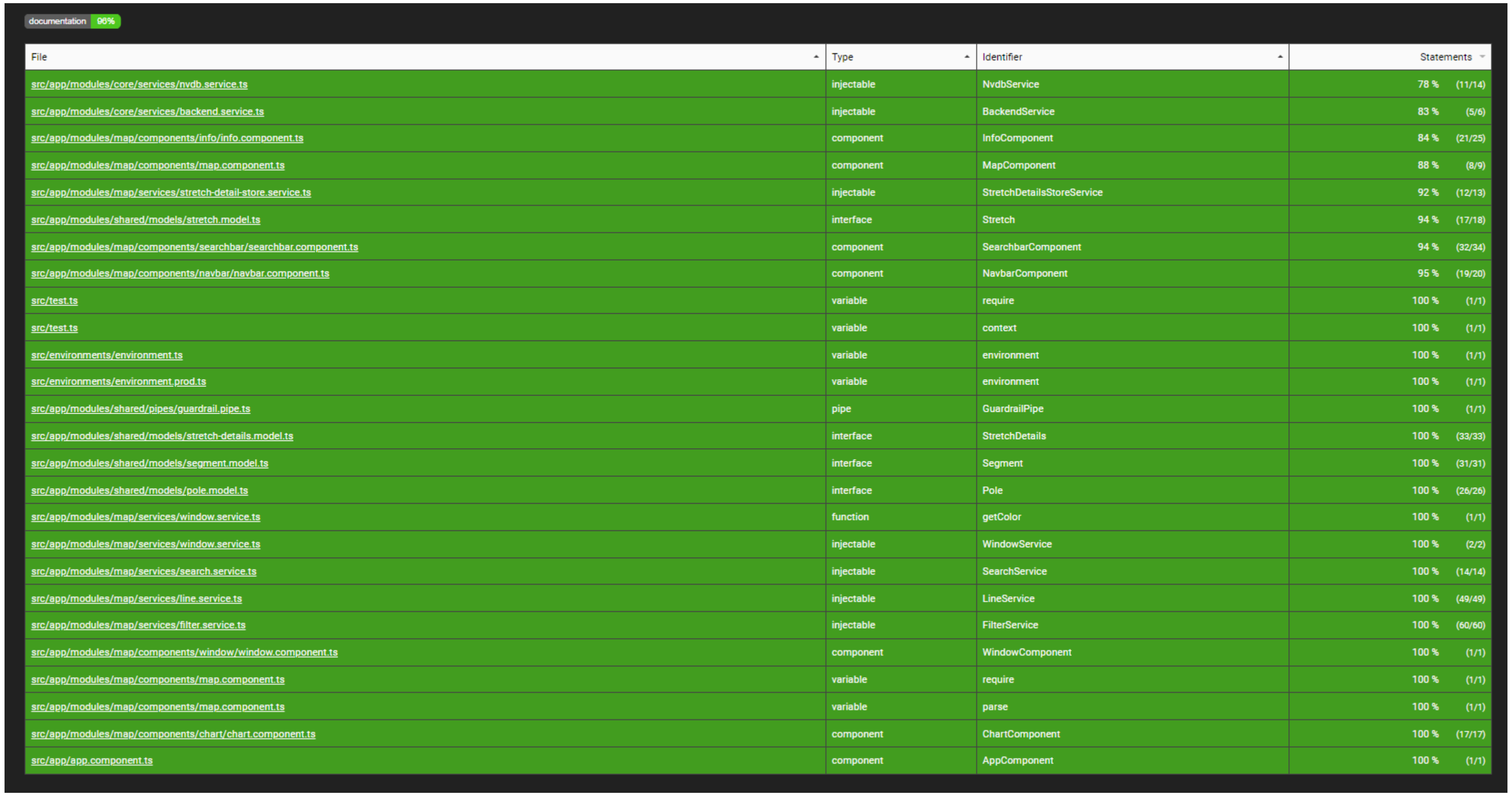
Table 8: Commands to run tests

B.8 Documentation of code

This project is using compodoc to generate docs of the Angular application. The tool generates documentation for all the common APIs of Angular: modules, components, injectables, routes, directives, pipes and classical classes. Compodoc also adds the JSDoc comments from the source-code to classes, methods and etc. in the documentation.

Run 'npm run generate-docs' to generate docs with the compodoc tool. Run 'npm run serve-docs' to serve the generated documentation on 'http://127.0.0.1:8080'.

Compodoc can also generate an overview of how much of the code is documented as you can see on the figure on page **66**.



documentation 96%

File	Type	Identifier	Statements
src/app/modules/core/services/nvdb.service.ts	injectable	NvdbService	78 % (11/14)
src/app/modules/core/services/backend.service.ts	injectable	BackendService	83 % (5/6)
src/app/modules/map/components/info/info.component.ts	component	InfoComponent	84 % (21/25)
src/app/modules/map/components/map.component.ts	component	MapComponent	88 % (8/9)
src/app/modules/map/services/stretch-detail-store.service.ts	injectable	StretchDetailsStoreService	92 % (12/13)
src/app/modules/shared/models/stretch.model.ts	interface	Stretch	94 % (17/18)
src/app/modules/map/components/searchbar/searchbar.component.ts	component	SearchbarComponent	94 % (32/34)
src/app/modules/map/components/navbar/navbar.component.ts	component	NavbarComponent	95 % (19/20)
src/test.ts	variable	require	100 % (1/1)
src/test.ts	variable	context	100 % (1/1)
src/environments/environment.ts	variable	environment	100 % (1/1)
src/environments/environment.prod.ts	variable	environment	100 % (1/1)
src/app/modules/shared/pipes/guardrail.pipe.ts	pipe	GuardrailPipe	100 % (1/1)
src/app/modules/shared/models/stretch-details.model.ts	interface	StretchDetails	100 % (33/33)
src/app/modules/shared/models/segment.model.ts	interface	Segment	100 % (31/31)
src/app/modules/shared/models/pole.model.ts	interface	Pole	100 % (26/26)
src/app/modules/map/services/window.service.ts	function	getColor	100 % (1/1)
src/app/modules/map/services/window.service.ts	injectable	WindowService	100 % (2/2)
src/app/modules/map/services/search.service.ts	injectable	SearchService	100 % (14/14)
src/app/modules/map/services/line.service.ts	injectable	LineService	100 % (49/49)
src/app/modules/map/services/filter.service.ts	injectable	FilterService	100 % (60/60)
src/app/modules/map/components/window/window.component.ts	component	WindowComponent	100 % (1/1)
src/app/modules/map/components/map.component.ts	variable	require	100 % (1/1)
src/app/modules/map/components/map.component.ts	variable	parse	100 % (1/1)
src/app/modules/map/components/chart/chart.component.ts	component	ChartComponent	100 % (17/17)
src/app/app.component.ts	component	AppComponent	100 % (1/1)

Figure 17: Documentation coverage according to Compodoc

B.9 Continuous Deployment

There are five stages in the present pipeline architecture. The first step is to install the node modules that will be used to build the application. Because the modules are cached, this phase can be skipped if there are no new updates to the modules. Linting and testing are the next two phases. ESLint does the linting, while Karma runs the tests in a headless Chrome browser. The final stages are to create and deploy the artifacts. The artifacts are automatically deployed to an OpenStack webserver hosted by NTNU, and the website is publicly accessible at 129.241.152.152. Listing 6 contains the pipline code used in this deployment.

```yaml
image: node:14

pipelines:
  branches:
    main:
      - step:
          name: Installation
          caches:
            - node
          script:
            - npm install
          artifacts:
            - node_modules # Save modules for next steps
      - step:
          name: Lint
          script:
            - npm run lint
      - step:
          name: Test
          script:
            - >
              wget -q -O - https://dl-ssl.google.com/linux/linux_signing_key.pub | apt-key add - && \
                sh -c 'echo "deb [arch=amd64] http://dl.google.com/linux/chrome/deb/ stable main" >> /etc/apt/sources.list.d/google.list' && \
                apt-get update && \
                apt-get install -y google-chrome-stable xvfb procps
            - npm run test:ci
      - step:
          name: Build
          script:
            - npm run build:production
          artifacts:
            - dist # Save build for next steps
      - step:
          name: Deploy
          deployment: production
          script:
            - echo "$(ls -la)"
            - echo "$(ls -la dist)"
            - ssh ubuntu@129.241.152.152 rm -rf /var/www/html/
            - scp -r dist/bachelor-kart ubuntu@129.241.152.152:/var/www
    /html/
```

Code example 5: Pipeline Code

B.10 Security

The SSH key used to deploy the artifacts to the webserver is stored as a secret in the Bitbucket repository. This secret can only be viewed or changed by users with admin access to the repository.

C Pre-project plan

Prosjektnr: 17

Kartvisning av veirekkverk Forprosjektplan

Versjon <1.2>

Prosjektnr: 17

Revisjonshistorie

Dato	Versjon	Beskrivelse	Forfatter
21.01.22	1.0	Første utkast	Gruppe 2
28.01.22	1.1	Rettelser	Gruppe 2
13.05.22	1.2	Fjerna seksjon 6	Gruppe 2

Prosjektnr: 17

Innholdsfortegnelse

Prosjektnr: 17

1. Mål og rammer

1.1 Orientering

Vi fikk tak i denne oppgaven via "Prosjektforslag – Data V22" som var en liste med oppgaver som går gjennom NTNU. Vi valgte denne oppgaven fordi den virket spennende og relevant. Vi har nylig jobbet med kart i ett annet emne og tenkte vi kunne bruke den kunnskapen vi hadde opparbeidet oss der i denne oppgaven. Oppgaven virket givende i den forstand at vi er med på å lage noe som vil forbedre trafikksikkerheten og som i det store bilde kan redde liv. Vi synes også det er motiverende å kunne lage noe som vil gjøre jobben til andre mindre tidkrevende og som kan gjøre at de kan jobbe på en mer effektiv måte.

1.2 Problemstilling / prosjektbeskrivelse og resultatmål

iSi sitter på millioner av bilder av vegrekkverk, deriblant en god del bilder med ulike feil på rekkverket. Dataene for hvert bilde ligger i en egen MongoDB-database, mens bildene ligger på Amazon S3. Hver meter av rekkverket er fotografert med to kamera, ett øvre og ett nedre. Vi ønsker å vise disse dataene i et interaktivt kart, der man kan filtrere på feiltyper, gruppere feil på strekninger og vise bilder på en måte som gjør at brukere får forståelse av risiko på ulike vegstrekninger.

Resultatmål:

- Innføring i datagrunnlag og API for data
- Sette opp nettbasert kartløsning for visning og filtrering av data
- Vise bilder på kart, bla. muligheter fram og tilbake og skifte mellom øvre og nedre kamera
- Se på mulighet for klassifisering av strekninger

1.3 Effektmål

Målene for gruppa:

- Bli gode på å jobbe i Scrum og bruke agile metoder
- Få erfaring i å jobbe med prosjekt med en ekstern oppdragsgiver
- Utvikle en løsning som vil bli brukt av oppdragsgiver

1.4 Rammer

Behov for penger, utstyr og tid. Spesialbehov materialer og rom.

Ikke aktuelt.

2. Organisering

Utviklerteamet: Apimanju Rajan, Herman Svae Nauf, Mathias Hagen, Ulrik Auflem Eikås

Oppdragsgiver: iSi AS
Kontaktperson: Ida Kjørholt

Veileder: Di Wu
Medveileder: Ottar L. Olsen

3. Gjennomføring

3.1. Hovedaktiviteter

Arbeidsoppgaver fordeles i Jira, og vi vil sørge for lik fordeling av oppgaver blant alle gruppemedlemmene. Hvordan det gjøres og hvorfor vil være en vurdering vi tar før oppgaven blir lagt inn i Jira. Når det gjøres vil være avhengig av i hvilken grad fullføring av oppgaven er kritisk for fremtidige gjøremål.

3.2. Milepæler

Opplisting av kritiske datoer.

28. Januar – Innleveringsfrist for forprosjektrapporten

22. April – Muntlig presentasjon på engelsk

6. Mai – Levere rapport til veileder senest 2 uker før innleveringsfrist

18. Mai – Poster

20. Mai – Innleveringsfrist for rapporten

4. Oppfølging og kvalitetssikring

4.1 Kvalitetssikring

En utnevnt student har overordnet ansvar for kvalitetssikring, løsninger gjennomgås i sprint review og sprint retrospective, men alle i gruppa plikter til sikre god kvalitet på arbeidet.

4.2 Rapportering

Rapportering vil skje ved sprint review ukentlig eller bi-ukentlig. Det vil skje i et møte med oppdragsgiver og veileder. I møtet vil det bli rapportert om produktet og prosessen.

5. Risikovurdering

Risikoanalyse som vurderer sårbarheter i prosjektet (hendelse, sannsynlighet, konsekvens og tiltak).

Hendelse	Innvirkning	Sannsynlighet	Tiltak
Corona virus (COVID-19)	Middels	Middels	Smittevern
Sent oppmøte	Lav	Lav	Fraværsrapport
Uenigheter	Lav	Middels	Skikkelig planlegging. Diskusjon ved uenigheter

For stor arbeidsmengde	Høy	Lav	Planlegge tidsbruk og bestemme omfanget av prosjektet.
Dårlig samarbeid	Høy	Lav	Snakke om det, kartlegge løsninger på problemet.
For lite brukertesting	Høy	Lav	Sikre gode og relevante testere f.eks SVV.

D Feedback

Bakgrunn

iSi AS holder på å utvikle en brukervennlig kartvisning for sin digitale rekkverkskontroll, iSi inSight. Dette er et innovativt produkt som bruker maskinlæring for bildegjenkjenning for å inspisere rekkverk. I den forbindelse skriver fire studenter fra NTNU bacheloroppgave i samarbeid med oss våren 2022.

Oppgaveteksten er relativt åpen, noe som gir studentene mulighet til å vise både kreativitet, problemløsnings- og gjennomføringsevner. Vi har hatt jevnlige sprintmøter og holdt kontakt på Teams under oppgaveperioden, og vi synes samarbeidet fungerer utmerket.

Vår tilbakemelding på løsningen iSi er svært tilfreds med løsningen som studentene har kommet frem til. De har klart å sette seg inn i behovet til sluttbruker, noe som er helt essensielt for å lykkes, og har utviklet flere brukervennlige funksjonaliteter i webportalen. Her er noen eksempler vi vil trekke frem:

Kartvisningen gjør det enkelt for brukeren å få oversikt over strekningene som er inspisert. I tillegg fremgår kritikalitet på de ulike vegstrekningene tydelig ved bruk av fargekoder.

Bildevisningen har flere funksjonaliteter vi tror brukeren vil sette pris på, for eksempel zoom-funksjonen og at man enkelt kan bytte mellom to ulike kameravisninger. Det er også nyttig at kartet automatisk forflytter seg slik at kartmarkøren sentraliseres på siden dersom nye vegreferanser velges.

Grafen under bildevisningen er hjelpsom for brukeren ved at den gir god oversikt over den generelle tilstanden på hele vegstrekningen. Holder man musepekeren over grafen, vil man få ytterligere detaljert informasjon, noe som er gunstig.

Studentene har integrert data via API fra Nasjonal vegdatabank om både fartsgrense og trafikkmengde på strekningen. Denne konteksten bidrar til å fastslå i hvilken grad en eventuell feil er kritisk. Slik informasjon vil kunne hjelpe brukeren å prioritere hvilke feil som bør utbedres raskest i en beslutningsprosess.

Løsningens bidrag til iSi

iSi har dratt mye inspirasjon fra både samarbeidet og diskusjoner med studentene og det endelige resultatet de har utviklet. Noe av funksjonaliteten har vi gått videre med å implementere i vårt eget produkt. Et eksempel på dette er grafen under bildevisningen som illustrerer rekkverkshøyden over hele veistrekningen.