# Preliminary Project Report

Web application development for an image accessibility workflow

OsloMet Bachelor thesis – Spring 2024

**Group members:**

Noor Ali Dibo – s374186

Sander Wiik Hjermstad - s362119

Hekuran Ismajli - s364761

Thanh Nguyen - s354587

**Supervisor:**

Raju Shrestha

# Table of contents

## 1.0 - Introduction

Due to the increasing need for a faster and more efficient web development workflow and the speedy evolution of AI, a consistent tool for meeting image accessibility standards could help web developers and others to quickly and consistently describe the images they are using without much effort. We will be attempting to make such a tool in the form of a web application where the user can upload an image and get a generated text description as an output.

## 2.0 - Summary

This bachelor's project is undertaken by four students, Hekuran Ismajli, Noor Ali Dibo Thanh Nguyen and Sander Wiik Hjermstad under the supervision of Raju Shrestha. Our project is sponsored by OsloMet and focuses on developing a web application for providing tool which generates subjective descriptions of an image for web developers and individuals with visual impairments.

Our team consists of students two different study programmes, Applied Computer Science and Software engineering.  This combination gives us the advantage of being an interdisciplinary group, while still having multiple shared courses that give us a common ground to build the project on.

We have dedicated ourselves to developing a web application that not only meets the needs of web developers and the challenges that individuals with visual impairments face. We believe that this project will bring a positive learning outcome. We are looking forward to turning our plans into an excellent product.
In essence, this bachelor's project entails an image description web application with the possibility to extend the project by also implementing it into a web browser extension in the future as an extended part of the project. The web application will be made with .NET (C#) framework.

Furthermore, in the process of making the web application, we have developed a weekly and monthly plan for the project which consists of meetings, web design, coding and report writing. Through the process, we will have meetings with our supervisor for further discussion and improvements on the project.

To enable the progression of the project we have developed a well thought out and detailed plan using Microsoft Excel for the upcoming weeks and months in order to help us with the process of making a finished product for our client.

During the first two weeks, we have discussed collectively assessing our individual skill set and what relevant subjects we are taking to contribute to the plan. Additionally, we had meetings with our supervisor where we discussed the minimal requirements and asked the supervisor questions we needed answered. Within this timeframe, a detailed plan has been established to guide us in the future weeks.

As a team we have engaged in multiple meetings dedicated to brainstorming ideas and working together on the tasks. These meetings have given us the opportunity to share insights and exchanged perspectives and generate concepts. We have managed to declare goals and requirements that include Minimum Viable Product, using .NET framework, selection of an ML model, and incorporation of Docker. By declaring these goals and requirements we are able to establish a structured guidelines to develop the project.

## 3.0 - Goals and requirements

We have differentiated the requirements of the project into two groups: functional requirements and non-functional requirements. The minimal functional requirements are as follows:

- Being able to upload (insert) an image from your local files on your computer to the web application.
- Using the uploaded image to generate a subjective AI description and display it.
- Using the generated image description to get an evaluation of the description and display it.
- Download the image to the computer containing metadata including the generated description.
- Create a user account and be able to login and logout. The user can also make changes to their account (basic profile configuration).
- Upon successful login, users will have the ability to save images with the metadata into a dedicated, private folder. The folder is located within their personal user area on the website.
- The dedicated private folder will have sorting, filtering and search functionalities to enhance user-friendliness.
- Users should also be able to download, delete and modify previous images saved in their private folder.
- If not satisfied with the generated description of the image, or if just want to modify it, it allows manual editing in order to improve the description quality. The modified description will get re-evaluated.

We've also decided to expand the project further with some other functionalities which was not intended in the first place. The following functionalities is something we want to achieve only if we have some time left over – reasoning why we called the functionalities above minimal for an MVP (minimum viable product).

- Allowing the user to modify the website, for example: change in font, font size, contrast/colour, dark/light-mode.
- A web browser extension: allowing the user to use the tool on other websites he/she visits. The extension will gather images that are presented on the website. If the image(s) has an alt-text already coded, then output it. Otherwise, generate a description and evaluate it on selected images.

Now for the non-functional requirements:

- Having a quick performance on the description generator and evaluator is an important factor we have to measure. If it takes too long time relative to just manually writing the caption it might become an unattractive tool to use. At this stage we find it difficult to precisely estimate the time it takes to generate a caption/description of an image. However, we've done some tests on how fast ChatGPT-4 (on date, 25.01.2024) could generate captions to random images and we found the average time being 7-8 seconds. There are a lot of factors to consider here. The most variable part is the complexity of the AI-model, the hardware it is running, current load on the system and the size of the image. However, we decided on using the ChatGPT performance as an idea of a goal plus some margins on the requests from/to the web application. We believe it will take the same amount of time to evaluate the description plus margins on requests. So, the performance goal is that it should not take longer than 30 seconds after uploading the image and starting the generation.
- The web application should be supported by the major web browsers for example: Google Chrome, Firefox, Opera, Safari and Microsoft Edge.
- The web application should be responsive and compatible, meaning accessible with different screen-sizes from a wide-screen to mobile.
- The project should be future proof by being able to easily switch the ML-models with other ones with minimal coding effort.
- Accessibility features should be implemented for user with disabilities.
- The web application should be available and reliable to use.
- User data and images should be stored securely.
- The design should be user-friendly. We will test this by performing user testing.

The tools we have decided to use in the project may change throughout the development process. However, what we have decided to use at this stage is the following:

- *.NET (C#)* and JavaScript for web-development. As the web application itself does not contain many different pages and features, we believe using JS libraries such as REACT or frameworks such as Angular will just be counterproductive. We therefore believe in using Native JavaScript. Might add jQuery for more efficient coding. We will also use *Python* to a degree on setting up the ML models.
- *Figma* is a web design tool we have used before and have good experience with. It allows us to work together as a group in the design process.
- We have not yet decided on the database management software, but all of us have experience with relational databases from *MySQL*, *PostgreSQL* and *MongoDB*. All three mentioned are good alternatives for this project.
- We are going to use *Docker* to build containers and run the web application along with the DB and the machine learning models.
- For the planning phase we are using *Microsoft Excel* on the weekly Gantt project planner and *Miro.com* for the Kanban Board.

The set final deliverable given out by the supervisor is the following:

- A running web application, also containing the source code.
- A couple of docker builds that contain the ML-models which will work together with the Web server.
- A database, also in a docker, containing the data and the images that are uploaded through the web application.
- At last, the project report containing detailed description of the whole project including the groups process and thoughts throughout the whole project.

## 4.0 - Today´s situation

The situation for this project, as we believe is focused on how artificial intelligence in technology such as web applications could be used for as a tool for developers and individuals that generally need a generated description of an image. Artificial intelligence has

drastically changed throughout the years, which we think can have a huge advantage in the future, also a helpful tool for humans. Our client Oslomet may use an application such as this as a tool to generate descriptions or for education.

As for today´s situation, our team have discussed together in a meeting how the project can be applicable to our client. For example, if the application is used either for educational purposes or personal use. Not only will this application not be restricted only for our client, but it can also be a public application for everyone who needs and wants to use it.

So far, in the meetings with our supervisor we have discussed that situations where developers have the need a quick description of an image, they will have the opportunity to use our web application. We have also talked about making a web browser extension which we think is a great idea for a future expansion of the project. Its main goal as an extension is for users to quickly use the extension and drag and drop or upload an image and get a generated description. Furthermore, we aim to make it, - so it also reads aloud for individuals with impaired vision.

Our supervisor mentioned that the daily situation, especially for individuals with impaired vision may have a problem seeing what’s in the picture, therefore our main goal is to make the application and extension quick and easy, which we think would be more accessible for everyone at the same time.

For our client's situation, they might have employees and students that may be visually impaired, and using our application may give them a better understanding what’s in the images they are looking at for example, when professors are reading a text with images or when a student is writing a text and wants to add multiple images.

## 5.0 - Solutions

Our proposed solution will be a functioning web application which consists of several distinct features, some of them with multiple ways of being integrated. We have chosen to focus on accessibility and universal design. This will make for a web application with a simple and understandable interface for a quick and easy workflow.

These are the solutions we think are the best for an accessible web application for our purposes, and some of their alternatives.

## 5.1 - Back End:

The application itself will run inside multiple docker containers containing the website, a database with two tables (one for users and one for images), and the ML-model. We are unsure if the ML-model will run in a single container or two for generating the image descriptions separately and then evaluating the descriptions. We will have to explore this further in the practical part of the project. They will be connected through API endpoints that we will be creating.

### *5.1.1 - ML-model:*

The machine learning model we will be using will be created and trained by ourselves using *Tensorflow* and *Keras* in Python. For training the model we will be using MS COCO (Microsoft Common Objects in Context), which is a free dataset for training image detection machine learning models. We haven't tested what works best yet, so what model and dataset we will be using might change.

#### Alternatives:

The model could be made using multiple ML-platforms, like *PyTorch* and Scikit-learn, which in practice wouldn't make any difference for our purposes other than a possible difference in performance. We also have the option to use a pre-trained model, like *Cobanov*'s "image captioning", which is a free MIT license software for image captioning. This might make for an easier and more efficient way to generate image descriptions but might make it harder to debug and test. This might also make the description evaluations harder to do, as it might be something we have to add to the pre-existing model.

### *5.1.2 - Database:*

The best solution for a database is for it to run inside a Docker container with two tables. One for the images, and one for accounts.

## 5.2 - Front end:

### *5.2.1 - Interface:*

The interface should be simple and readable, with most of the main functions being accessible from the main page: uploading an image, generating a description, creating a user, logging in and out, and editing the description of an uploaded image. In addition to this, we will have a button linked to a different view for browsing your saved images if you are logged in to the page. If you are not logged in, you will have the option to create a user or log in to an existing one instead.

An alternative would be to have everything in the same view. That includes the users´ image gallery automatically appearing on the bottom of the page, with an ability to scroll down and view all your saved images at once. This, however, creates a cluttered interface for every user that's logged in, which is unnecessary for users who work with one image at a time and don't need to review previous image descriptions very often.

### *5.2.2 - Account features:*

An account can be created using an e-mail address and a password. The user will have its saved images tied to it, with the ability to view previously described images and save new ones. This will be implemented in an ordinary manner.

If the user is logged in, they will have the ability to save a generated image to the database.

### *5.2.3 - Upload image:*

The user should be able to upload an image and generate a description without being logged in to an account. They can upload an image by either dragging and dropping the image in a specified area on the web page, or by choosing a local file path. When the image has been chosen, they have to click a button to confirm the upload. When confirmed, the image description is generated and displayed, along with an evaluation score.

They will then have the option to save the image if they are signed in. If they are not, they have to create an account or sign in and regenerate a description.

### *5.2.4 - Create account:*

Creating an account will be implemented in an ordinary manner. On the main page; there will be a button for creating an account. The "create account" form will be shown, and the user can enter the e-mail address and password they want to tie to their account. No other information is to be saved. The e-mail address and password will be encrypted and saved in the database.

### *5.2.5 - Browse gallery:*

If a user has signed into their account, they will be able to browse an image gallery of their saved images. On the main page, there will be a button that directs you to the gallery. When clicked, a new view will appear with the saved images appearing in a grid. The user will be able to sort the images by date, or search by keyword.

#### Alternative:

An alternative could be to have the image gallery browsable in the main view, as mentioned above.

### *5.2.6 - Log in to account:*

Will be implemented in an ordinary fashion with a “log in” button on the main page.

### *5.2.7 - Edit description:*

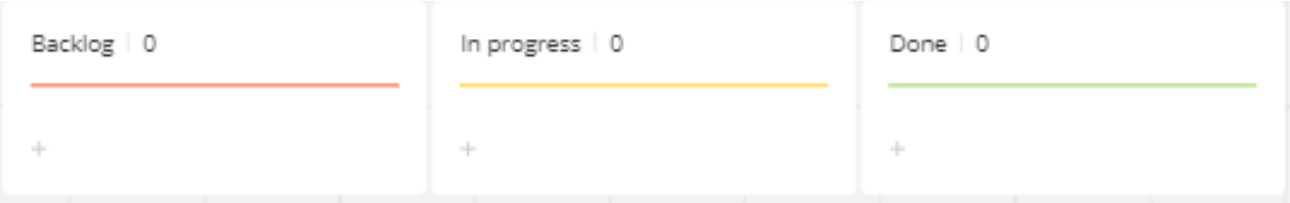
After an image description has been generated, the user will be able to manually edit the description if they are logged in. They will also be able to do this through the “browse gallery” view.

## 6 - The planning

After meeting with the supervisor for the first time we were advised to set up a weekly plan for the whole project. Even though we had 21 weeks to finish the project, we knew it would pass by quickly. One of the first things we did was to set up a Weekly Gantt project planner in Excel. We wrote the main tasks we had to cover and added details later on. We split the project into 5 different phases: Planning, Development, Testing, Final Development, Deployment, and documentation. The weekly planner is attached below. We decided to leave out the main task of “Writing report” since it is an activity we will be doing consistently throughout the project. We’ve already made a type of diary where we write down tasks we are working on, current difficulties we are facing, and our solutions on a daily basis.

| | ACTIVITY | PLAN START | PLAN DURATION |
|---|---|---|---|
| PLANNING | Project kickoff and organization | 2 | 5 |
| | Requirement gathering and analysis | 3 | 2 |
| | System design specification | 4 | 1 |
| | Finalizing tools and technologies | 4 | 1 |
| DEVELOPMENT | Core backend development | 5 | 4 |
| | Frontend Development | 5 | 4 |
| | Integration and Basic Testing | 6 | 2 |
| | Development of additional features | 9 | 3 |
| TESTING | User testing and feedback | 11 | 1 |
| | Security and accessability enhancement | 11 | 2 |
| Final dev | Additional Feature development | 13 | 3 |
| | Extensive testing and bug fixing | 16 | 2 |
| Depl. & Doc. | Deployment preperation | 16 | 3 |
| | Final launch | 18 | 2 |

Although the Gantt project planner provided a good overview of the whole project we need to cover, it is still difficult to understand which specific tasks needs to be accomplished. The way we suggested to solve this is using a Kanban board with three rows: backlog, in progress, and done. In the Kanban Board we could split the bigger task into smaller tasks that can be divided within the group. Each group member was assigned a specific colour so that we were able to see who is currently working on what task.




We decided on a physical group meeting at least once a week, but we were open to increase the frequency if needed. We knew the importance of having an open dialogue/discussion about everything and it should be easy to ask questions, raise concerns, and suggest ideas so we created a *Discord* channel as a place we could freely discuss outside the meetings.

In addition to the group meetings, we set a biweekly group meeting with the supervisor, Raju. The supervisor meeting is somewhere we could ask for advice and get feedback on the project status. His guidance would be crucial when encountering significant decisions about the project's direction. He suggested creating a type of form where we could write down our questions before the meeting, note his answer during the meeting and the action we took after the meeting.

| Date | Question | Answer | Action |
|---|---|---|---|
| 08.12.2024 | Should the images be uploaded and saved? | The assignment is very open, it is up to us as a group to decide. However, we should always keep in mind the usefullness of the webapp. It might be a better idea to have the images stored on the user privatly for later use instead of just disappearing. | We agree on being able to upload the images. Be able to download the images to private computer (local files), but also have a designated private folder in the webapplication where the user could save the images. |
| 08.12.2024 | Should we have a login/user profile? | Again, the assignment is open, however it sound like a good idea. | Yes, we will create a user profile. |
| 08.12.2024 | Where to run the docker containers? | Locally in the beginning, later on - maybe on a server providd by OsloMet. | |
| 08.12.2024 | Idea; expanding the webapp to a browser extension | Good idea, do-able. However try to focus on the MVP (minimum viable product) in the beginning. | Our goal is to acheive this, but first be done with the MVP. |
| 08.12.2024 | How should the image evaluate itself? | After the ML image description generator has sent the response back to the webapp. The description gets fetched and evaluated on the ML image description evaluator. | Updated the sequence diagrams. |

So far, after we have declared our goals and requirements, we have successfully completed following tasks:

- Developed activity diagrams: activity diagrams are crafted to visually represent the interactions and workflow of the system.
- Conclude a textual description of the activity diagrams: this is documentation that provides a detailed narrative description of each step and interactions.
- Developed sequence Diagrams: sequence diagrams illustrate the chronological order of interactions between different components.
- Outlined functional and non-functional requirements: these are specifications for a framework for the development process.
- Produced design sketches using Figma: we have used Figma, a design tool, to make sketches to represent the user interface and overall aesthetics.