Project plan
Face quality metrics application

Walid Demloj
Kjetil Grosberghaugen
Julian Nyland Skattum
Hans Petter Fauchald Taralrud

January 2021

Contents

1 Goals and Frameworks

1.1 Background

Mobai is a technology firm and system that works primarily with facial recognition. They are located at Norsk Biometrilab in Gjøvik and delivers Systems for facial recognition, attack detection against biometrical systems and face morph detection.

An important tool for Mobai is artificial intelligence and machine learning, which they use to create models for detection of biometric attributes like faces, and detect attacks against these solutions. Good models for these solutions are dependent on relevant datasets and training of the models. In order to train good models, it is important to have good quality on the data. This is a process that Mobai would like to have automatized.

Currently Mobai is using different Image Quality Metrics (IQMs) to determine the quality of the images in a dataset. In order to make this process go smoother, Mobai now wishes to create an application that will take a dataset, and display the quality of the images as a report using two IQMs Mobai already has to determine the quality. Mobai wants to include human assessments to evaluate the accuracy of the IQMs. This will make it easier for Mobai to determine whether a specific dataset is usable in training good models, or give a quick assessment of a customers dataset.

1.2 Project goals

1.2.1 Main goal

The main goal is separated into three essential parts:

- Provide Mobai a working web application that showcases face image quality scores for images in a dataset, using two IQMs given by the company.
- Create and conduct a subjective experiment to obtain data from human appraisals.
- Evaluate the IQMs accuracy by correlating the subjective and objective face image quality scores.

1.2.2 Long term goals

Building reliable models are an essential part of artificial intelligence and machine learning. Therefore knowing the quality of images plays an important part. An application would automate the process of training models, building satisfying and clean datasets, and evaluating customers datasets.
The main goals Mobai wishes to fulfill in the long term are:

- Speed up the whole process of evaluating customers datasets
- Achieve better models for training

- Create new datasets for further research

1.3 Framework

- The application will be packed and delivered as a container solution or as a set of containers.
- The client side of the application will have a graphical user interface.

2 Scope

2.1 Subject Area

For Mobai, artificial intelligence and machine learning are essential subjects for their work. For this report and creation of the application, it is important to understand what is a good image, and how different IQMs work in separate ways to generate a quality score.

Our task will cover different technologies and subjects within programming and image quality assessment, these include:

- Different face and image quality assessment papers, like [1] and [2]
- Face image quality metrics in python
- Docker desktop and containers for ease of deployment
- Subjective assessment survey to collect data
- Unit testing
- Front-end web-developed user interface using HTML and JavaScript
- Back-end development using mainly python

2.2 Delimitation

We as developers are not responsible for creation of the IQMs or relevant data sets. All IQMs for face image quality and relevant data sets are given to us from Mobai.

2.3 Task Description

The task is separated into two parts, one objective and one subjective Face Image Quality evaluation. The objective part consists of creating an application that uses two IQMs from Mobai that generates a quality score on each individual image within a dataset. The application will create a report on the whole dataset and output relevant data about the quality of the dataset, including face image quality score on the images.

The subjective assessment consists of having subjects evaluate facial images from a dataset based on what they think is good or bad quality. They will be shown different images of varying quality and asked to rate the quality of the images ranging from very low to very high. This way, we get multiple quality scores on the images in the dataset. The subjective results will then be used to compare with the objective results and presented in the report. The higher the correlation, the more accurate the IQM will be.

3 Project Organization

3.1 Responsibilities and roles

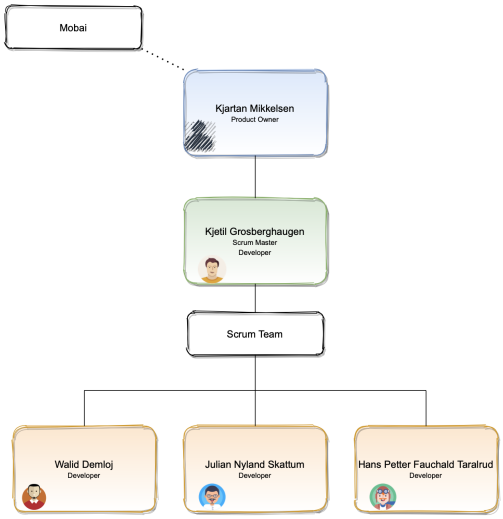


Figure 1: Organization chart

By using a Scrum development model to structure the project, the group members have different responsibilities according to their roles within this agile methodology.

Kjartan Mikkelsen is the Product Owner and our main contact from Mobai. He will participate in all sprint planning meetings/sprint review meetings. Mikkelsen is accompanied by two scientist that work for Mobai. They are specialized in our topic and will join our scheduled meetings when intricate questions occur.

Kjetil Grosberghaugen is the Scrum Master. Beside being a developer, his main tasks are to ensure that the development process flows evenly, arrange necessary meetings as well as acting like a connector between the developers and the Product Owner.

Walid Demloj, Julian Nyland Skattum and Hans Petter Fauchald Taralrud are the project developers. Throughout the project lifetime they will be working

on different tasks. However, their main tasks are to perform assigned work and to ensure that they have something to do.

3.2 Routines and rules in the group

Some essential points from our group rules scheme:

- It is expected that each group member works daily with this project, and the workload should be about 30-40 hours per week.
- All group members are expected to be available to work together as a group for at least 4 hours every working day, unless an acceptable reason has been given in advance.
- All group members must be active in the project. If there are things you don't understand, it's better to ask the other members for help than do nothing at all.
- All working hours shall be logged for each group member and be presented at status meetings.
- Status meetings will be held once a week.

4 Planning, Follow-Up and Reporting

4.1 Main division of the project

The project will consist of a web application with both an objective assessment and a subjective assessment for face image quality.

4.1.1 Choice of software development method

This project is characterised by uncertainty, a limited amount of labor, several meetings, a survey that may differ in complexity and a development phase with requirements that can change. These characteristics paved the way for us to pick the agile software development method Scrum, as was suggested by Mobai.

Our group consists of four inexperienced students with little to none experience with comprehensive projects of this size. Our inexperience alone causes some uncertainty in regards to deadlines, survey work and development. Having regular meetings with both Mobai and our project supervisor as well as receiving regular feedback, will decrease the chance to drift of track. This also ensures that our final product is as close to Mobai's vision as possible. The agile software development methods involves the client to a far more degree than the plan-driven approaches, which was an important point to take into consideration.

Since our team is rather small in size with no experts, the agile development methods seemed natural. Should we have decided to use one of the plan-driven methods, like the Waterfall-method, the chance to not succeed seemed greater.

A key principle in the Waterfall-method is its strict way of dividing a project into phases. The model emphasises finishing one phase before the next one starts. This would not be suitable for our project because of the very reason that it´s time consuming. Should the development phase run into trouble that was not taken into consideration in the planning/design phase, the design phase would have to start over. It´s likely that problems will occur in our project and having the flexibility provided by Scrum assures that progress is achieved as much as possible.

Should we finish Mobai's desired application quickly it was suggested by Mobai that additional functionality could be added to the solution if our time schedule allowed for it. Changes in the requirements were something Mobai were open to discuss and Scrum handles this well, which was another reason for our choice.

Within the agile development methods both Kanban and eXtreme Programming (XP) were taken into consideration. Both of these are reliable methods that provide solid structuring and flexibility [3]. However Kanban usually does not incorporate the element of predefined roles and therefore it suits our project to a lesser degree. The chapter "Responsibilities and roles" (3.1) describes what each group member is in charge of. Although all of us are expected to participate in all the tasks, some delegation of responsibility were decided to achieve greater structure of the tasks. In addition to this the end dates for all the tasks are rather tough to determine, which is why a sprint with several tasks will be more reliable.

With that said, Kanban is excellent for getting an overview of the tasks needed to be done, which is why we also decided to incorporate this method to a lesser degree. *Trello*[4] is used for tracking the tasks and the tasks are divided into three phases: *To-Do*, *Doing* and *Done*. Since Kanban is not our main method, no specific rules are set considering the amount of tasks in each phase.

XP was briefly considered, but we quickly realised that it was difficult to follow all 12 practices all the time. However we decided to implement the *Pair Programming* practice with a little tweak: Due to the uncertainty that the pandemic causes, the pair programming will be done by screen-sharing on laptop etc. Pair programming will improve the code quality which is the reason behind our choice [5].

4.1.2 Scrum layout

We will have sprints with a two-week duration. One-week sprints are rather short and leads to excessive meetings which affects our time management. On the other hand longer sprints tends to involve the client to a lesser degree, which is not ideal in our case. We are dependant on input from Mobai to satisfy their requirements.

The start of the sprints takes place on Tuesdays, starting 2. February. These are expected to finish before 11:00 every other Monday. The weekly scheduled meetings with our bachelor supervisor takes place every Monday at 12:00, which

gives us at least one hour to prepare both the next sprint and what we should discuss during the meeting. The following Tuesday a meeting with Mobai will be held, where we will go through what the group achieved during the sprint.

We use the Scrum-pattern *Definition of done* to collectively define what development-tasks are considered done [6]. In partnership with Mobai we will come to an agreement upon different criteria that form the definition of done. It is important to set different criteria to different work tasks (it should be a difference between coding-criteria and report-criteria). This pattern gives us a familiar understanding of work quality and absoluteness. We also obtain good habits in our workflow using the definition of done as a checklist to correlate with the user stories. In that way we prevent possible delays occurring in the development process. Here is an example of our report-criteria:

1. The section is completed according to the member
2. The member has analyzed the section's contents
3. The section is checked for typos
4. The whole group has read and approved the section

4.2 Plan for status meetings and decision points

The group will have weekly status meetings on Mondays at 11:00. On these meetings we will discuss and evaluate our current progress regarding the project, and discuss other subjects of importance.

We will have regular meetings with our NTNU project supervisor every Monday at 12:00. If we think an extraordinary meeting is needed we can contact our supervisor and try to arrange an extra meeting that week, preferably on Thursday or early on Friday.

During the initial phase of the project we will have regular status meetings with Mobai every Tuesday at 12:00. The frequency of these meetings can be adjusted as needed during the project.

5 Organization of Quality Assurance

5.1 Documentation, standard usage and source code

Documents and storage

- The bachelor-thesis is written in LaTeXusing Overleaf online compiler.
- LaTeXsource code is synchronized between Overleaf and our Github-repository to maintain backup protection.
- Summaries from meetings with Mobai and our supervisor are stored in a shared folder in Google disk.

Testing and code quality For coding in python, we will be using the pep–8 standard for project structure and style. Pylint[7] will be used to confirm that the standard is used correctly. We will use unit tests for testing the code, and SonarQube[8] for a whole analysis of the code quality.

5.2 Configuration Management

When multiple developers are working on this project at the same time, it is important to have a management system. This is to ensure that all developers have the latest version of the program, and also that the developers don't interfere with each others work. The commits to the project will have to be coordinated, to ensure that no developer is doing something that already has been done.

We will choose to use Github as our configuration management system. Commits will be done evenly to ensure that all developers always have access to the latest version of the program, with descriptive commit messages that shortly explains what has been done. The commit history can be used if we need to rollback to a previous version of the code. If two developers are working on the same component, a merge conflict will occur and have to be fixed before it gets pushed out to Github. This will keep the developers from overwriting each others code. Bugs, features and improvements can be tracked with the Github issue tracker.

5.2.1 Development workflow

Issues tracking For tracking issues during coding we will use Github's integrated Issues tracker. Discovered bugs, new features, suggested improvements and other potential issues will be added to Issues.

Smart commits We will use smart commits during the development to track which issues the commits are connected to. Git commits and merge requests will contain a keyword for what the commit is doing with the issue (e.g. fix, close, implement), the issue number the developer is trying to solve and a clear commit description.

5.3 Tools

Table 1: Table of tools

Name	Description	Application
draw.io	Online flowchart and diagram software	Creating charts and diagrams
Docker	Software containerization platform	Containers for final product
GanttProject	Project management software	Creating Gantt charts
Git	Distributed version-control system for tracking software changes	Version control
GitHub	Internet hosting for software development and version control	Version control
GitLab	Internet hosting for software development and version control	Version control
Google Drive	File storage and synchronization service	File storage and synchronization
Overleaf	Cloud-based editor used for writing and editing LaTeX documents	Project report
Planning Poker	Online sprint planning game	Sprint planning
Pylint	Source-code, bug and quality checker for Python	Code quality inspection
SonarQube	Platform for code quality inspection	Code quality inspection
Toggl	Application and online tool for time tracking	Time tracking
Trello	Online KanBan board for task tracking	Project management
Visual Studio Code	Development IDE	Application development

5.4 Risk analysis

5.4.1 Identifying the risks

A project with this range of scope is clearly exposed for several risks. It is important to identify these hazards, and to initiate countermeasures if needed. The list below shows different risks that possibly could occur during the project. The risks are numbered independently of their severity.

1. Group members leaving the project
2. Deadlines are not met
3. Bachelor thesis is not delivered in time
4. Mobai cancels the project
5. A similar project launches
6. The software does not fulfill the requirements
7. Inadequate planning and execution of subjective experiment
8. Loss of documents and source code
9. Sickness among group members, project supervisor or Mobai
10. Application breaking bugs

5.4.2 Risk analysis

The figure (2) analyzes the different risks that could happen during the project. These risks are split into two different factors: the probability for an action to occur and the consequences of that action. The colors indicates levels of risks where green illustrates low risk, yellow illustrates moderate risk and red illustrates high risk. Each number in the table are adopted from the risk identification described in the previous section.

Consequence / Probability	Little consequence	Moderate consequence	Serious consequence	Very serious consequence
Unlikely	5		1, 6	3, 4, 8
Likely			2	10
Very likely			7, 9	
Vastly likely				

Figure 2: Overview of the consequences and probability

5.4.3 Risk management and countermeasures

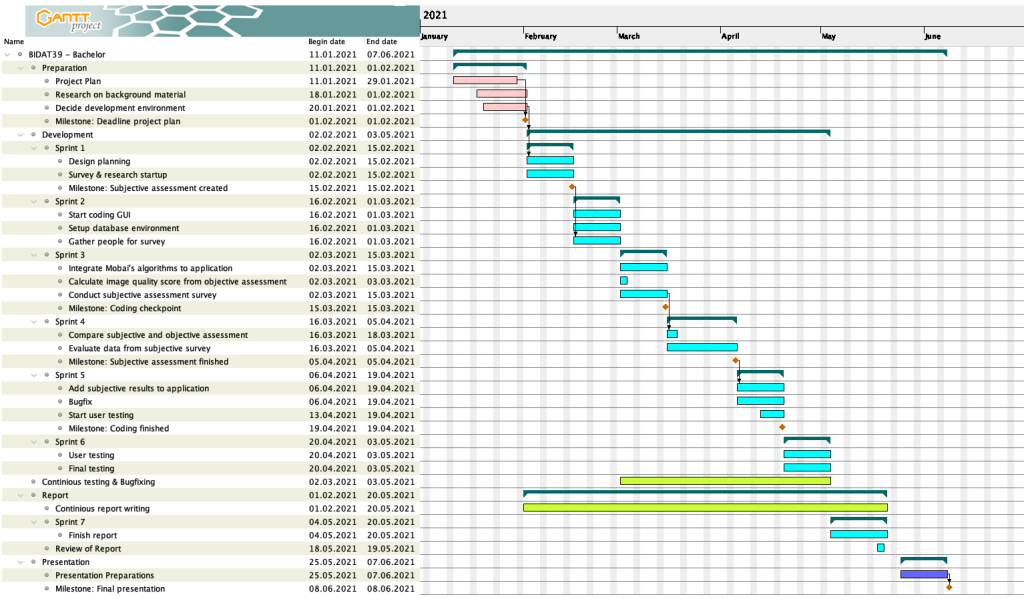
There are discussed measures to avoid risk events to occur. The countermeasures lower the negativity of the consequences or could reduce the probability of an action to take place at all.

Event	Severity	Countermeasure
Group members leaving the project	1	A structured group with good planning prevents members leaving the project. If a member is forced to leave by lack of participation, a new project orchestration will be necessary.
Deadlines are not met	2	Having a consistent workflow and a regular meeting schedule makes it easier to meet the deadlines. If there are too many tasks in the upcoming sprint, a rescheduling may occur and Product Owner will be informed.
The software does not fulfill the requirements	6	Performing research at an early stage in the development process clarifies what is possible to develop technologically. Being able to communicate with the Product Owner during the software development process enables regular requirements updates.
Inadequate planning and execution of subjective experiment	7	Creating the survey at an early development stage and ensuring enough participants to perform the survey.
Loss of documents and source code	8	Using backup-technologies when writing the report and coding the software prevents loss of important content.
Sickness among group members or Mobai	9	Following the Covid-19 restrictions and arranging online-meetings baffle sickness to the greatest extent.
Application breaking bugs	10	At an early stage in the coding process analyze what bugs that can result being application breaking. Applying regular testing to our workflow enable the bugs to be fixed quickly.

Figure 3: Risk management

6 Plan of Execution

6.1 Gantt diagram



Name	Begin date	End date
BIDAT39 – Bachelor	11.01.2021	07.06.2021
Preparation	11.01.2021	01.02.2021
Project Plan	11.01.2021	29.01.2021
Research on background material	18.01.2021	01.02.2021
Decide development environment	20.01.2021	01.02.2021
Milestone: Deadline project plan	01.02.2021	01.02.2021
Development	02.02.2021	03.05.2021
Sprint 1	02.02.2021	15.02.2021
Design planning	02.02.2021	15.02.2021
Survey & research startup	02.02.2021	15.02.2021
Milestone: Subjective assessment created	15.02.2021	15.02.2021
Sprint 2	16.02.2021	01.03.2021
Start coding GUI	16.02.2021	01.03.2021
Setup database environment	16.02.2021	01.03.2021
Gather people for survey	16.02.2021	01.03.2021
Sprint 3	02.03.2021	15.03.2021
Integrate Mobai's algorithms to application	02.03.2021	15.03.2021
Calculate image quality score from objective assessment	02.03.2021	03.03.2021
Conduct subjective assessment survey	02.03.2021	15.03.2021
Milestone: Coding checkpoint	15.03.2021	15.03.2021
Sprint 4	16.03.2021	05.04.2021
Compare subjective and objective assessment	16.03.2021	18.03.2021
Evaluate data from subjective survey	16.03.2021	05.04.2021
Milestone: Subjective assessment finished	05.04.2021	05.04.2021
Sprint 5	06.04.2021	19.04.2021
Add subjective results to application	06.04.2021	19.04.2021
Bugfix	06.04.2021	19.04.2021
Start user testing	13.04.2021	19.04.2021
Milestone: Coding finished	19.04.2021	19.04.2021
Sprint 6	20.04.2021	03.05.2021
User testing	20.04.2021	03.05.2021
Final testing	20.04.2021	03.05.2021
Continious testing & Bugfixing	02.03.2021	03.05.2021
Report	01.02.2021	20.05.2021
Continious report writing	01.02.2021	20.05.2021
Sprint 7	04.05.2021	20.05.2021
Finish report	04.05.2021	20.05.2021
Review of Report	18.05.2021	19.05.2021
Presentation	25.05.2021	07.06.2021
Presentation Preparations	25.05.2021	07.06.2021
Milestone: Final presentation	08.06.2021	08.06.2021

Figure 4: Gantt diagram

BIDAT39 - Bachelor	11.01.2021	07.06.2021
Preparation	11.01.2021	01.02.2021
Project Plan	11.01.2021	29.01.2021
Research on background material	18.01.2021	01.02.2021
Decide development environment	20.01.2021	01.02.2021
Milestone: Deadline project plan	01.02.2021	01.02.2021
Development	02.02.2021	03.05.2021
Sprint 1	02.02.2021	15.02.2021
Design planning	02.02.2021	15.02.2021
Survey & research startup	02.02.2021	15.02.2021
Milestone: Subjective assessment created	15.02.2021	15.02.2021
Sprint 2	16.02.2021	01.03.2021
Start coding GUI	16.02.2021	01.03.2021
Setup database environment	16.02.2021	01.03.2021
Gather people for survey	16.02.2021	01.03.2021
Sprint 3	02.03.2021	15.03.2021
Integrate Mobai's algorithms to application	02.03.2021	15.03.2021
Calculate image quality score from objective assessment	02.03.2021	03.03.2021
Conduct subjective assessment survey	02.03.2021	15.03.2021
Milestone: Coding checkpoint	15.03.2021	15.03.2021
Sprint 4	16.03.2021	05.04.2021
Compare subjective and objective assessment	16.03.2021	18.03.2021
Evaluate data from subjective survey	16.03.2021	05.04.2021
Milestone: Subjective assessment finished	05.04.2021	05.04.2021
Sprint 5	06.04.2021	19.04.2021
Add subjective results to application	06.04.2021	19.04.2021
Bugfix	06.04.2021	19.04.2021
Start user testing	13.04.2021	19.04.2021
Milestone: Coding finished	19.04.2021	19.04.2021
Sprint 6	20.04.2021	03.05.2021
User testing	20.04.2021	03.05.2021
Final testing	20.04.2021	03.05.2021
Continious testing & Bugfixing	02.03.2021	03.05.2021
Report	01.02.2021	20.05.2021
Continious report writing	01.02.2021	20.05.2021
Sprint 7	04.05.2021	20.05.2021
Finish report	04.05.2021	20.05.2021
Review of Report	18.05.2021	19.05.2021
Presentation	25.05.2021	07.06.2021
Presentation Preparations	25.05.2021	07.06.2021
Milestone: Final presentation	08.06.2021	08.06.2021

Figure 5: Tasks

6.2 Milestones and decision points

In order to keep up with the time schedule, we have made some general milestones for when different activities should be finished. This is to have a greater assurance that we are in route with the project.

- **Milestone 1, 1.February:** Deadline project plan, start coding
- **Milestone 2, 15.February:** Subjective experiment created.
- **Milestone 3, 15.March:** Coding checkpoint.
  - Database should be working.
  - Data sets should be read from database and run through IQMs. Initialized docker desktop.
- **Milestone 4, 30.March:** Subjective experiment finished.
  - Gathered data from the subjective experiment and ready to implement it into the objective results and solution.
- **Milestone 5, 19.April:** Coding finished.

References

[1] Torsten Shlett, Christian Rathgeb, Olaf Henniger, Javier Galbally, Julian Fierrez, and Christoph Busch. Face image quality assessment: A literature survey. 22(1):1–17, 2020.

[2] Javier Hernandez-Ortega, Javier Galbally, Julian Fierrez, Rudolf Haraksim, and Laurent Beslay. Faceqnet: Quality assessment for face recognition based on deep learning. 22(2):1–9, 2020.

[3] Rachaelle Lynn. *Kanban vs. Scrum: What are the Differences? [Internett].* Planview; 2021 [hentet 21.01.2021]. Available from: https://www.planview.com/no/resources/guide/introduction-to-kanban/kanban-vs-scrum/.

[4] Trello. https://trello.com/.

[5] Agile Alliance. *Extreme Programming [Internett].* Agile Alliance; 2016 [hentet 21.01.2021]. Available from: https://www.agilealliance.org/glossary/xp/.

[6] The Scrum Patterns Group. *Definition of done [Internett].* Raleigh: The Scrum Patterns Group; 2019 [hentet 25.01.2021]. Available from: http://scrumbook.org/.

[7] Pylint. https://www.pylint.org/.

[8] Sonarqube. https://www.sonarqube.org/.