

# **SIM - STUDENT INTERNSHIP MATCHMAKING PLATFORM**

**BACHELOR OF COMPUTER SCIENCE**

BY

GURJOT SINGH AULAKH

MORTAZA BAQERI

OKBAMICHAEL MUSSIE ELIAS

FAESAL AL SHHADAT

ARMAN YEDICAM

**OSLOMET**

UNDER THE SUPERVISION OF

ASSOC. PROF. TULPESH PATEL  
FACULTY OF TECHNOLOGY, ART AND DESIGN  
OSLO METROPOLITAN UNIVERSITY

SPRING 2023



## Department of Information technology

Postal address: PO box 4 St. Olavs plass, 0130 Oslo  
Visiting address: Holbergs plass, Oslo

PROJECT NO.  
2023 - 57

AVAILABILITY  
Open

# BACHELOR'S PROJECT

MAIN PROJECT TITLE	DATO
Student Internship Matchmaking (SIM) Platform	26.05.2022
	NO. OF PAGES/APPENDICES
	203/14

CLIENT	INTERNAL SUPERVISOR
Oslo Metropolitan University	Tulpesh Patel

SUMMARY
In 2021, the government stated in a public message an encouragement for educational institutions to focus on better university-industry collaborations by including more and better practical internships (Kunnskapsdepartementet, 2021, p. 5). OsloMet implemented the "IT project in practice" course in 2015, which aimed to encourage collaboration between students and industry partners through nine different project types. As the number of interested students increased over the years, the coordinator identified the need for a platform to simplify project management and the student internship matchmaking process. Due to the government's statement and the course coordinator's situation, our group felt motivated to implement a solution for this problem. Last semester, our group created an SRS and an MVP for the platform. In the current semester, we reconstructed our previous MVP using Laravel, incorporating improved security measures and user-friendly features. The resulting platform simplifies the management and coordination of student-industry collaborative projects.

3 STIKKORD
Webapplication
Intership Matchmaking Platform
Skill Development

# Acronyms

**API** Application Programming Interface. 6, 7, 10, 11, 35–37, 59, 75, 76, 98, 99, 154

**CRUD** Create, read, update and delete. 66

**CSS** Cascading Style Sheets. 36, 64

**DSL** Domain Specific Language. 36

**GDPR** General Data Protection Regulation. 54

**HTML** HyperText Markup Language. 60, 62, 64

**HTTP** Hypertext Transfer Protocol. 35, 69, 75, 76, 119

**IDE** Integrated Development Environment. 35

**MFA** Multi-Factor Authentication. 97

**MVC** Model View Controller. 60, 64

**MVP** Minimum Viable Product. 5, 18, 31, 32, 48, 58, 156

**REST** Representational State Transfer. 37, 98

**SIM** Student Internship Matchmaking. 4, 9, 11, 22, 51, 52, 118–120

**SPA** Single Page Application. 59

**SRS** Software Requirements Specification. 18

**SSO** Single Sigh On. 53

**TDD** Test Driven Development. 156

**XSS** Cross-Site Scripting. 87, 93

# Preface

This report provides a comprehensive overview of our bachelor thesis completed at Oslo Metropolitan University. The bachelor project was initiated by participating in the “[DATA3710, IT project in practice](#)” course during the previous semester, where we collaborated with OsloMet and one of their project course coordinators.

Our bachelor thesis revolves around creating a complete web application that streamlines the administrative tasks for the course coordinator. Our primary objective was to utilize the knowledge we had accumulated over the past three academic years to develop a solution that would contribute value to the university while meeting Oslomet’s needs and requirements.

We express our heartfelt gratitude to Tulpesh Patel for entrusting us with the responsibility of solving the administrative challenges faced by the “[IT project in practice](#)” course. We are thankful for his guidance and support throughout the project, which enabled us to grow and develop our skills. We also appreciate the course coordinator’s valuable insights and input throughout the platform’s development. With his suggestions and feedback, this project was possible. Lastly, we thank the Department of Computer Science at OsloMet for providing us with the resources and facilities necessary to complete this project.

This report is designed for digital reading as a PDF and includes clickable links to facilitate navigation to references, figures, appendices, and external resources.

# Contents

<b>Preface</b>	<b>3</b>
<b>List of Figures</b>	<b>13</b>
<b>List of Tables</b>	<b>14</b>
<b>1 Introduction</b>	<b>15</b>
1.1 Project Background . . . . .	15
1.2 Challenges of Course Management Process . . . . .	16
1.3 Project Objectives . . . . .	18
1.4 Previous Work and Methodology . . . . .	18
1.5 The project group . . . . .	19
1.6 Supervisor . . . . .	20
1.7 Project Provider . . . . .	21
<b>2 Initial Product Specifications</b>	<b>22</b>
2.1 Stakeholders Impacted by Student Internship Matchmaking (SIM) . . . . .	22
2.2 Functional Requirements . . . . .	23
2.3 Non-functional Requirements . . . . .	26
<b>3 Process documentation</b>	<b>27</b>
3.1 Planning . . . . .	27
3.1.1 At start-up . . . . .	27
3.1.1.1 Requirements Review Workshop and Priority List Creation . . . . .	27
3.1.1.2 Addressing Availability Challenges and Digital Meetings . . . . .	28

3.1.1.3	Introduction to Git Flow and Implementation Approach . . . . .	28
3.1.1.4	Progress Gantt Chart for Task Overview and Timeline . . . . .	29
3.1.1.5	Exploring PHP Laravel as the Framework Choice . . . . .	31
3.1.2	Under development . . . . .	31
3.1.2.1	Sprint Planning and Focus Areas . . . . .	31
3.1.2.2	Deprioritizing some Requirements . . . . .	31
3.1.2.3	Balancing Functional Requirements and Design Focus . . . . .	32
3.1.3	Areas of Responsibility . . . . .	32
3.2	Tools used in the process . . . . .	34
3.3	Framework/library used . . . . .	36
3.4	Working method . . . . .	39
3.4.1	Agile development . . . . .	39
3.4.2	Scrumban . . . . .	39
3.4.3	Scrumban in practice . . . . .	39
<b>4</b>	<b>The development process</b>	<b>41</b>
4.1	Start-up phase . . . . .	41
4.1.1	Design and Prototype Sprint . . . . .	41
4.1.2	Implementation planning and set up of the software environment . . . . .	42
4.2	Development phase . . . . .	46
4.2.1	Development of MVP . . . . .	48
4.2.1.1	Docker Container Hosting . . . . .	48
4.2.1.2	Hosting . . . . .	50
4.2.1.3	MySQL Database . . . . .	51
4.2.1.4	Single Sign On . . . . .	53
4.2.1.5	Development of the front end with the blade . . . . .	53
4.2.2	Documentation . . . . .	53
4.2.3	Correspondence between the project plan and actual implementation . . . . .	54
4.2.4	Quality assurance . . . . .	54
4.3	Reflection and Discussion . . . . .	55
4.3.1	Challenges around basic setup of coding environment . . . . .	55
4.3.2	Challenges regarding changes in software requirement . . . . .	56
4.3.3	Challenges in development of MVP . . . . .	58

4.3.4	Challenges and reflections about the frontend . . . . .	59
4.3.5	Challenges in deprioritizing some Requirements . . . . .	59
<b>5</b>	<b>Product Documentation</b>	<b>60</b>
5.1	Description of solution . . . . .	60
5.1.1	Architecture . . . . .	60
5.1.2	Frontend . . . . .	61
5.1.2.1	Architecture . . . . .	63
5.1.3	Backend . . . . .	64
5.1.3.1	Models . . . . .	66
5.1.3.2	Controllers . . . . .	69
5.1.3.3	Middleware . . . . .	70
5.1.3.4	Policies . . . . .	73
5.1.3.5	API Resource . . . . .	75
5.1.3.6	Routes . . . . .	76
5.1.4	Sign-on . . . . .	76
5.1.5	Design and accessibility . . . . .	78
5.2	Correspondence between requirement specification and product . . . . .	79
5.3	Central data structures in the solution . . . . .	83
5.3.1	Principles for code development . . . . .	83
5.3.1.1	Modularity . . . . .	83
5.3.1.2	Object-Oriented Programming (OOP) . . . . .	83
5.3.1.3	Separation of Concerns . . . . .	84
5.3.1.4	Don't Repeat Yourself (DRY) . . . . .	84
5.4	Database . . . . .	84
5.4.1	Factories . . . . .	84
5.4.2	Migrations . . . . .	85
5.4.3	Seeders . . . . .	86
5.5	Security . . . . .	87
5.5.1	Secure Development and safety measures . . . . .	87
5.5.2	Risk Assessment and Threat Modeling . . . . .	87
5.5.3	OWASP . . . . .	93
5.5.4	Security solutions for the application . . . . .	94

5.5.5	Terms & Conditions . . . . .	95
5.5.6	Secure development in practice . . . . .	96
5.5.7	Reflections on login and authentication method . . . . .	96
5.6	Relationship to machines/databases/OS . . . . .	97
5.6.1	API documentation . . . . .	98
5.6.2	Database Integration . . . . .	100
5.6.3	Main parts of the program . . . . .	100
<b>6</b>	<b>Testing</b>	<b>102</b>
6.1	Introduction . . . . .	102
6.2	User Testing . . . . .	103
6.2.1	Testing under development phase . . . . .	103
6.2.1.1	User Tests done by course coordinator . . . . .	103
6.2.1.2	Feedback/result . . . . .	103
6.2.1.3	Response to the Feedback . . . . .	106
6.2.1.4	Student-led User Testing . . . . .	107
6.2.2	Limitations and Challenges in User Testing . . . . .	109
6.3	Technical Testing . . . . .	109
6.3.1	PHPUnit and Laravel's built-in testing library . . . . .	109
6.3.2	Unit Test . . . . .	110
6.3.2.1	Basics of Defect Tracking . . . . .	111
6.4	Feature test . . . . .	113
6.5	Smoke test . . . . .	114
6.6	Challenges in technical testing . . . . .	116
6.7	Correspondence between accurate test coverage and ideal test coverage . . . . .	117
<b>7</b>	<b>User Guide</b>	<b>120</b>
7.1	Registration and Login . . . . .	120
7.1.1	Common Login Page For All Actors . . . . .	120
7.1.2	Common Password Reset . . . . .	121
7.1.3	Create an Account . . . . .	123
7.2	The Core Flow Of The Application . . . . .	124
7.3	Common Functionalities for all actors . . . . .	144

<b>8 Conclusion and discussion</b>	<b>153</b>
8.1 Learning Outcome . . . . .	153
8.2 What is the product's utility value? . . . . .	154
8.3 What would we have done differently? . . . . .	154
8.4 Feedback from the Course Coordinator . . . . .	155
8.5 Status of further development and production setting of the product . . . . .	156
8.6 Summary and Conclusion . . . . .	156
<b>A Appendix</b>	<b>164</b>
A.1 Previous Minimum Viable Product (MVP) Version 1.0 . . . . .	165
A.2 Final version of the prototype . . . . .	173
A.3 User Test - Template from IT project in practice . . . . .	184
A.4 User Test - Template . . . . .	185
A.5 Interview guide . . . . .	187
A.6 User Guidelines . . . . .	188
A.7 Notification form for the processing of personal data . . . . .	190
A.8 First version of the database . . . . .	193
A.9 Second version of the database . . . . .	194
A.10 Hosting throw OsloMet VM . . . . .	195
A.11 Hosting throw Microsoft Azure . . . . .	198
A.12 Modified docker-compose.yml file . . . . .	200
A.13 Poster . . . . .	202
A.14 Certificate from the course coordinator . . . . .	203

# List of Figures

1.1	Course enrollment and project stats . . . . .	16
1.2	Course management phases for the course coordinator . . . . .	17
2.1	Course coordinator's functional requirements . . . . .	24
2.2	Student's functional requirements . . . . .	25
2.3	Requirements for project provider functionality . . . . .	25
2.4	Non-functional requirements . . . . .	26
3.1	Git workflow . . . . .	28
3.2	A progress Gantt chart of the project . . . . .	30
4.1	"Kanban" project board created in Notion. . . . .	43
4.2	A task from Notion . . . . .	43
4.3	Table of sub-tasks as viewed in the Task section of Notion. . . . .	44
4.4	Priority View from Notion. . . . .	45
4.5	Docker Containers . . . . .	49
4.6	Deploying error from OsloMet VM . . . . .	50
4.7	Application error from Microsoft Azure . . . . .	50
4.8	Final version of SIM's database . . . . .	52
4.9	New functional requirements for course coordinator . . . . .	57
4.10	New functional requirements for Student's functional requirements . . . . .	57
4.11	New functional requirements for project provider . . . . .	58
5.1	Shows how model view controller works . . . . .	61
5.2	Traditional Content Management System . . . . .	62

5.3	Headless Content Management System . . . . .	62
5.4	Resources folder . . . . .	63
5.5	Public folder . . . . .	63
5.6	The layering of the backend . . . . .	65
5.7	Model's folder . . . . .	66
5.8	Project's model . . . . .	67
5.9	Database fields filled with fake data . . . . .	68
5.10	Controller's folder . . . . .	70
5.11	Middleware's folder . . . . .	71
5.12	Kernel class . . . . .	72
5.13	Authenticate.php . . . . .	73
5.14	Policie's folder . . . . .	74
5.15	ProjectPolicy defines the policy for actions on the Profile model . . . . .	74
5.16	Put request route . . . . .	75
5.17	API resources' folder . . . . .	75
5.18	Route's folder . . . . .	76
5.19	Shows the network requests during the login and register . . . . .	77
5.20	Implementation status of Course coordinator's functional requirements . . . . .	80
5.21	Implementation status of Student's functional requirements . . . . .	81
5.22	Implementation status of Project Provider's functional requirements . . . . .	81
5.23	Non-functional requirements . . . . .	82
5.24	Project factory from SIM application . . . . .	85
5.25	Showing the migration code for the user's model . . . . .	86
5.26	Example of Risk Matrix. Retrieved from [smand] . . . . .	88
5.27	Definition of Stride. Retrieved from [Larndj] . . . . .	89
5.28	DREAD in Risk Matrix. Retrieved from: [SSB13] . . . . .	90
5.29	Risk Matrix Ranking . . . . .	91
5.30	Risk matrix with STRIDE and DREAD (first edition) . . . . .	91
5.31	The extended risk matrix (final version) . . . . .	92
5.32	This is the OWASP Top Ten 2021. Retrieved from [OWAnd] . . . . .	93
5.33	Input validation in the registration page . . . . .	95
5.34	Checkbox validation in the registration page . . . . .	95

5.35 Terms & Conditions link . . . . .	95
5.36 Swagger API documentation . . . . .	99
5.37 MySQL database configuration file . . . . .	100
6.1 Example code of Laravel’s built-in unit test . . . . .	111
6.2 Defect tracking rules . . . . .	112
6.3 Defect tracking overview . . . . .	113
6.4 Integration Test - course coordinator project update scenario . . . . .	114
6.5 Testing Company View Retrieval . . . . .	115
6.6 Testing Project View Under High Request Traffic . . . . .	116
6.7 Code Coverage Analysis: Examining Tested and Untested Code . . . . .	118
6.8 Early Stage Code Coverage Analysis for SIM Application . . . . .	118
6.9 Code Coverage Analysis for SIM Controllers . . . . .	119
7.1 The login page . . . . .	121
7.2 The password reset page . . . . .	122
7.3 The password reset mail . . . . .	122
7.4 Link to the account creation form . . . . .	123
7.5 Here, you can specify the type of account you wish to create . . . . .	123
7.6 The verification email . . . . .	124
7.7 The company’s dashboard . . . . .	125
7.8 ‘Add Project’ option from the side navigation . . . . .	125
7.9 Create a company first . . . . .	126
7.10 Example data needed to create a company . . . . .	126
7.11 Company created successfully . . . . .	127
7.12 Page 1, General project information . . . . .	128
7.13 Page 2, Project details . . . . .	129
7.14 Page 3, Project agreements page . . . . .	130
7.15 Page 4, final step to submit the project proposal . . . . .	131
7.16 Back to page 1, validation error message . . . . .	131
7.17 Project proposal created successfully . . . . .	132
7.18 Project proposal appears in “All Projects” table with status “NEW” . . . . .	132
7.19 Course coordinator dashboard where there only projects with status “new” . . . . .	133

7.20	Overview of various statistics . . . . .	133
7.21	Overview over all projects with status “NEW” . . . . .	134
7.22	Currently there are no projects with status “UPDATED” . . . . .	134
7.23	Toolbar . . . . .	134
7.24	Course Coordinator approves a project . . . . .	135
7.25	Course coordinator requesting changes . . . . .	136
7.26	Course coordinator addressing changes needed to the project proposal . . . . .	136
7.27	Change request sent successfully . . . . .	137
7.28	“All projects” link in the side navigation . . . . .	137
7.29	Overview over all projects . . . . .	138
7.30	Company dashboard after receiving a change request on a project proposal . . . . .	138
7.31	More details about the project in the dashboard . . . . .	139
7.32	The changes have been done by the company . . . . .	140
7.33	Company representative modify an existing project proposal . . . . .	140
7.34	Company representative modify tags of an existing project proposal . . . . .	141
7.35	The company representative submits changes to existing project proposal . . . . .	141
7.36	Project updated successfully . . . . .	141
7.37	Company-representative views all projects owned by them . . . . .	142
7.38	Overview of the updated projects on the course coordinators dashboard . . . . .	142
7.39	Success message, project proposal submitted . . . . .	143
7.40	Overview of the projects on the student dashboard . . . . .	143
7.41	Overview of one of the project’s proposals . . . . .	144
7.42	Overview of the profile page . . . . .	145
7.43	Overview of the profile details . . . . .	145
7.44	Overview of profile page after changes . . . . .	146
7.45	Overview of the input field . . . . .	147
7.46	Overview of the error message . . . . .	147
7.47	Showing that the student has been deleted . . . . .	148
7.48	Change password page . . . . .	149
7.49	The “All Users” page . . . . .	150
7.50	The “Admins” page . . . . .	150
7.51	The “Companies” page . . . . .	151

7.52 The “Students” page . . . . .	151
7.53 The result page after searching . . . . .	152

# List of Tables

3.1	Individuals in Group - Names, Roles, Responsibilities. . . . .	33
4.1	Sprints overview . . . . .	47
4.2	Details about docker containers . . . . .	49
6.1	Company requirements . . . . .	104
6.2	Course Coordinator's requirements . . . . .	105
6.3	Student requirements . . . . .	106
6.4	Tasks implemented after user test . . . . .	107
6.5	Feedback of student test . . . . .	108

# Chapter 1

## Introduction

### 1.1 Project Background

In 2021, the Norwegian government emphasized the importance of university-industry collaborations and practical internships for students. OsloMet established the course “[DATA3710, IT project in practice,](#)” which aims to provide students with practical IT experience.[\[Aul+22\]](#)

The course started around August 2015 and over the years, it quickly gained popularity among students, resulting in a total of approximately 492 students and 194 projects spread across 145 organizations in 9 different project types (numbers are from the last 9 semesters). As seen in figure [1.1](#), the trend of students, projects, and project providers has steadily increased over the past six years.

During the spring of 2023, the initial enrollment for the “[IT project in practice](#)” subject was 100 students, but 40 of them eventually dropped the subject, resulting in a total of 60 students remaining, as shown in [1.1](#). The exact reasons for the high dropout rate are unclear, but from the course evaluation we know that it is the number, type and variety of the projects there are not enough projects to satisfy all 100 students. Another potential factor could be the challenging nature of the internship matchmaking process at the beginning of the subject. Many of our fellow students have expressed similar concerns, stating that finding a suitable project and contacting the project provider can be difficult, ultimately leading to fewer students engaging in internships.

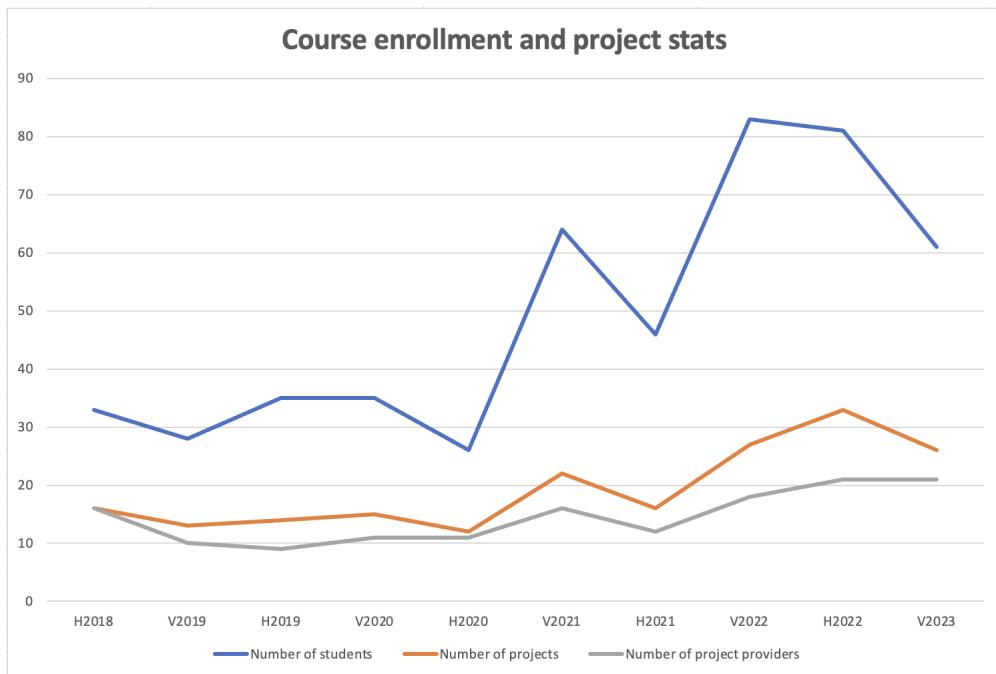


Figure 1.1: Course enrollment and project stats

## 1.2 Challenges of Course Management Process

Two key challenges arise in managing this course. Firstly, the course is overseen by a single course coordinator, which presents limitations regarding resource availability and workload management. Secondly, the current task management approach must be updated and more efficient. Figure 1.2 visually represents the primary administrative tasks associated with the course management process. These tasks include collecting projects via email, conducting quality assurance of projects using online forms, creating project descriptions by rewriting and copying/pasting from the online forms, uploading the descriptions on Canvas for students, and manually verifying project assignments by crosschecking Canvas, communicating with project providers via email, and reaching out to students.

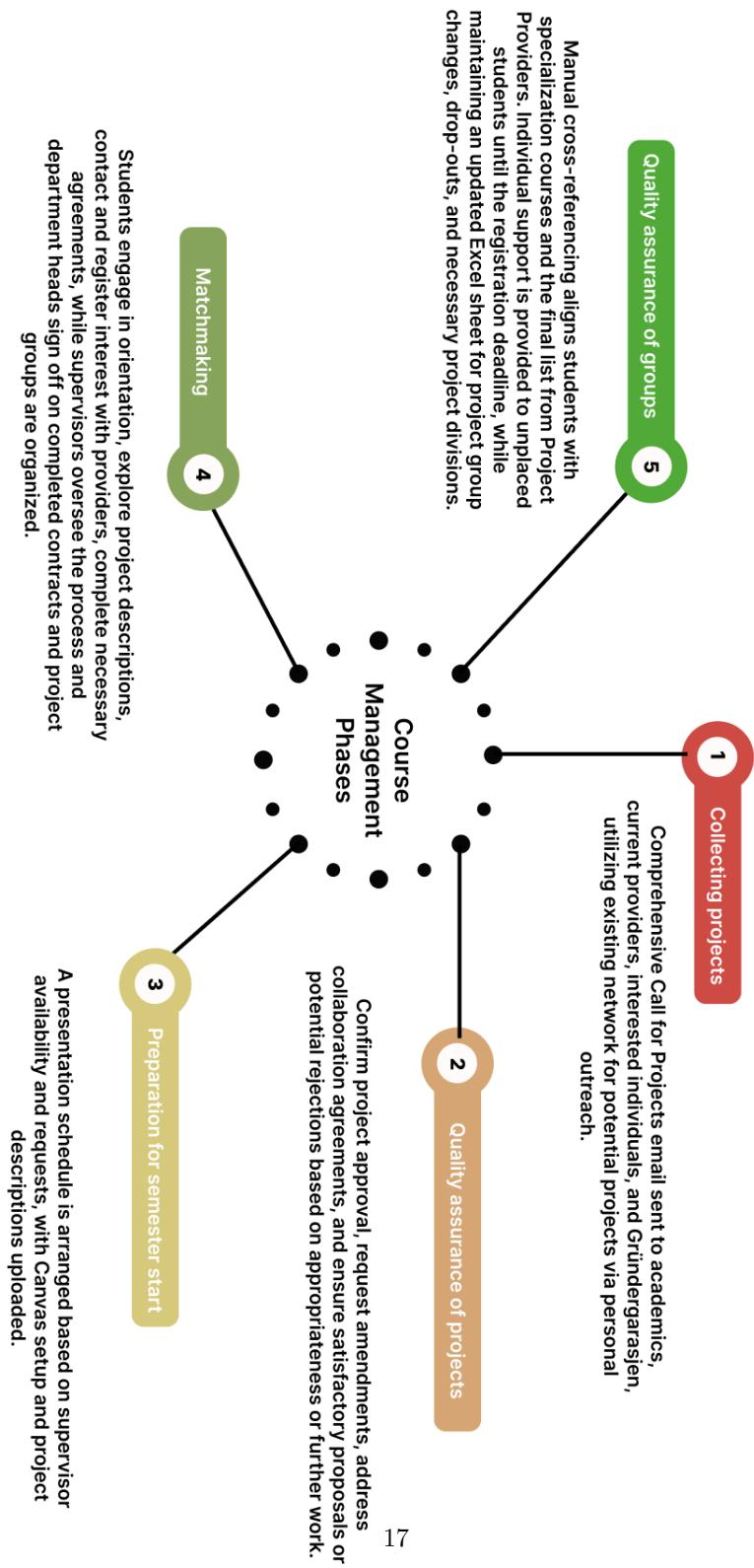


Figure 1.2: Course management phases for the course coordinator

With the increasing number of students taking the courses, as seen in [1.1](#). There is a need for a more effective solution to manage the courses. Addressing this problem is one of the main objectives of our Bachelor Thesis.

### 1.3 Project Objectives

Our solution focuses on streamlining and automating tasks related to managing the course, such as emailing companies, using online forms, exporting data, rewriting documents, and posting project descriptions on Canvas.

The main functionalities of the application include companies creating project proposals, the course coordinator reviewing and approving projects, and students applying for projects that match their interests and skills. This ensures a better matchmaking process and easier coordination in the beginning phase of the course “[IT project in practice](#)”.

### 1.4 Previous Work and Methodology

In the previous semester (Autumn 2022), we undertook a project as part of the “[IT project in practice](#)” course. The objective of the project was to design and implement a platform that simplifies the administrative processes associated with course management. We conducted a thorough analysis of the requirements and documented them in a Software Requirements Specification (SRS), see chapter [2](#) for more about our SRS. Using pure PHP programming, we developed a Minimum Viable Product (MVP) to address this challenge. While the previous Minimum Viable Product (MVP) was simplistic in nature, it did incorporate the essential functionalities, see appendix [A.1](#) The functional requirements for the application mainly remained the same, which was helpful during the development of the current solution.

Building upon our previous work, our goal for the bachelor’s thesis is to redevelop a more efficient and secure platform using the Laravel PHP framework.

## 1.5 The project group

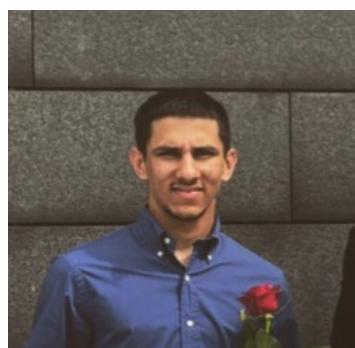
Our project group comprises five computer engineering students from OsloMet who have previously collaborated on various project tasks during our studies. We are a tightly knit group and are well acquainted with each other's strengths and weaknesses, which allows us to complement each other well. Together, we possess a diverse range of expertise and experience that we bring to the table for this bachelor thesis. Given our strong track record of successful collaboration, it was easy for us to join forces for this project.



(a) Gurjot Singh Aulakh  
Full-Stack Developer



(b) Mortaza Baqeri  
Full-Stack Developer



(c) Arman Yedicam  
IT-Security Analyst



(d) Okbamichael Mussie  
Technical tester



(e) Faesal al shhadat  
UI/UX Designer

## 1.6 Supervisor

**OsloMet:**

*Tulpesh Patel*

*Associate Professor,*

*Fakultet for teknologi, kunst og design,*

*OsloMet*

As we solved the problem of Tulpesh Patel, it was natural that he became both our internal and external supervisor. We contacted the supervisor early in the project and agreed to hold weekly meetings.

## **1.7 Project Provider**



Oslo Metropolitan University -  
Storbyuniversitetet  
Postboks 4, St. Olavs plass, 0130 Oslo

OsloMet is an abbreviation for the international name Oslo Metropolitan University. OsloMet offers various degree programs in technology, healthcare, and pedagogy. Institut IT, a department within OsloMet, provides the relevant course(s) related to information technology education and training.

## Chapter 2

# Initial Product Specifications

## 2.1 Stakeholders Impacted by Student Internship Matchmaking (SIM)

Those who will primarily benefit, and be affected by the “SIM”:

- The Course Coordinator
- Students taking the course
- Project Providers

The main tasks for the course coordinator are:

1. Receive projects proposals
2. Evaluate project proposals
3. Make projects available
4. Easy overview of students and projects

The main tasks for the student are:

1. Go through project descriptions
2. Apply for projects
3. Contact companies

The main tasks for the project provider are:

1. Create projects
2. Edit projects
3. Track project status
4. Go through student applications
5. Accept/decline student applications

## 2.2 Functional Requirements

The functional requirements define the specific features and capabilities that the system should possess to address the needs of its users. These requirements are often expressed as user stories in an agile development approach.[\[Prond\]](#) Figure 2.1, 2.2, 2.3, provides examples of functional requirements we made in the last semester prior to the bachelor project.

## **Functional requirements - Course Coordinator**

<b>Id</b>	<b>Description</b>	<b>Priority</b>
FRCC -1	Have an overview of students registered for the project courses	HIGH
FRCC-2	Have an overview of Project Proposals sent in by project providers	HIGH
FRCC-3	Quality-check the Project Proposals; request changes and updates to Project Proposal	HIGH
FRCC-4	Edit status of Project Proposals (accepted, under revision, rejected)	HIGH
FRCC-5	See overview of status of projects	HIGH
FRCC-6	Create Project Descriptions based on key information in accepted Project Proposals	HIGH
FRCC-7	Publish Project Descriptions for students	HIGH
FRCC-8	Have an overview of which students are registered to which project(s)	MIDDLE

Figure 2.1: Course coordinator's functional requirements

### **Functional requirements - Student**

<b>Id</b>	<b>Description</b>	<b>Priority</b>
FRS-1	Create and edit profile of themselves	MIDDLE
FRS-2	See all Project Descriptions	HIGH
FRS-3	Filter projects according to availability	HIGH
FRS-4	Can apply for the published projects	HIGH

Figure 2.2: Student's functional requirements

### **Functional requirements – Project Provider**

<b>Id</b>	<b>Description</b>	<b>Priority</b>
FRPP-1	Create and edit profile of themselves	MIDDLE
FRPP-2	Create project proposals	HIGH
FRPP-3	Edit project proposals based on feedback from the Course Coordinator	HIGH
FRPP-4	Access student profiles	HIGH
FRPP-5	Can accept/reject students	LOW

Figure 2.3: Requirements for project provider functionality

## 2.3 Non-functional Requirements

Non-functional requirements encompass the system's performance, security, reliability, and maintainability. These requirements are not directly tied to the system's functional goals but are crucial for effectiveness. Figure 2.4 presents examples of non-functional requirements.

<b>Id</b>	<b>Description</b>	<b>Priority</b>
NFRS-1	The system is protected from unauthorized access to the system and its stored data.	HIGH
NFRS-2	The system considers different levels of authorization and authentication across different user roles.	HIGH
NFRS-3	Meet and follow GDPR regulation	HIGH
NFRS-4	Follow OsloMet's guidelines and requirements for data security and privacy	MIDDLE
NFRS-5	The system shall provide a web page that explains how to navigate the site. This page should be customized based on what pages that user is allowed to access.	MIDDLE
NFRS-6	This help page should be accessible from all other pages.	MIDDLE
NFRS-7	Adheres to the (seven) principles for Universal Design	LOW
NFRS-8	Has a consistent "look and feel"	LOW
NFRS-9	Translation into another language (Norwegian)	LOW
NFRS-10	Scalability: the system should be able to accommodate additional course types and an increase in the number of students and projects without major reengineering.	MIDDLE

Figure 2.4: Non-functional requirements

# Chapter 3

## Process documentation

### 3.1 Planning

The process documentation will provide detailed information about our project planning and the steps taken to reach the final product. The planning phase of this project has been an essential part of our work, as we have dedicated efforts to documentation, front-end, and back-end development.

#### 3.1.1 At start-up

##### 3.1.1.1 Requirements Review Workshop and Priority List Creation

To ensure a clear understanding of the project requirements and specifications, the group initiated a workshop. This workshop allowed each member to individually review the requirements and specifications. Following the individual review, a joint session was conducted where the group discussed and consolidated their findings. As a result, a priority list of the requirements and specifications was created, as we have shown above in chapter 2 of the project documentation.

### 3.1.1.2 Addressing Availability Challenges and Digital Meetings

Recognizing that group members had other commitments that limited their availability throughout the week, the group decided to address this challenge by conducting digital meetings at the beginning of the project. Although these digital meetings allowed members to participate remotely, it was noted that it hindered the group's ability to work closely together. However, despite this constraint, all members could complete their assigned tasks successfully.

### 3.1.1.3 Introduction to Git Flow and Implementation Approach

In the first sprint of the project, a team member organized a concise workshop to introduce Git flow, a popular version control workflow. This workshop provided an overview of the Git flow process and outlined the team's approach to implementing it in the future, as shown in 3.1. By adopting Git flow, the team aimed to streamline their collaboration, ensure proper version control, and facilitate efficient code management throughout the project.

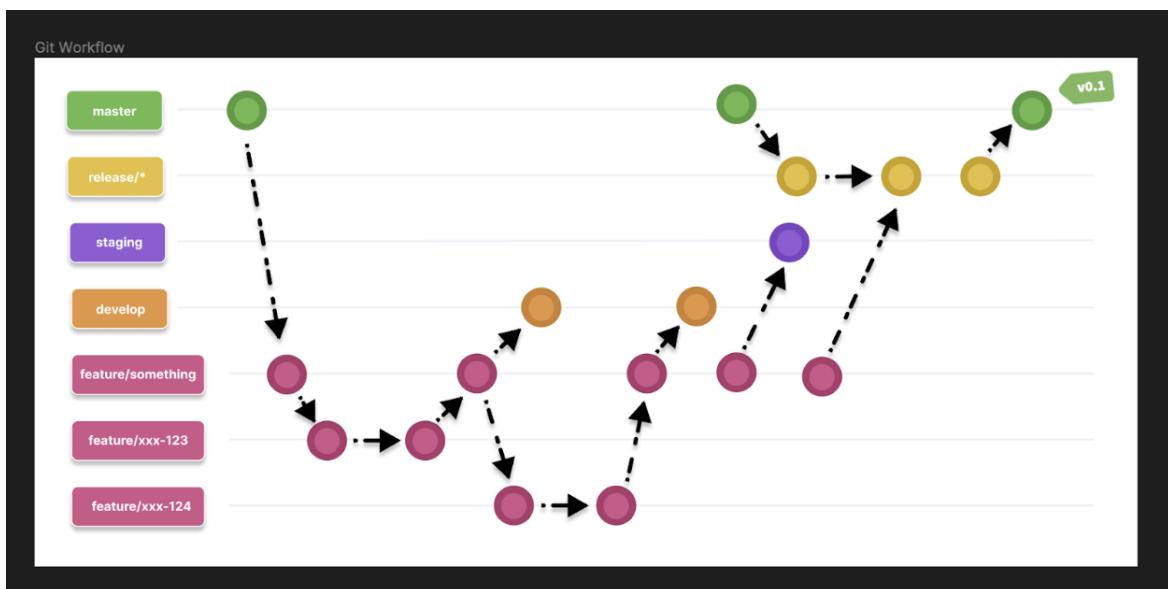


Figure 3.1: Git workflow

### **3.1.1.4 Progress Gantt Chart for Task Overview and Timeline**

To visualize the project tasks and their respective timelines, the group utilized a progress Gantt chart, as shown in [3.2](#). The chart divided the project into four phases: startup, development, testing, and documentation. Within each phase, specific milestones were identified to mark key progress points. The project timeline was depicted on the right side of the chart, providing an overview of the expected duration for each phase and milestone. The Gantt chart served as a helpful tool to monitor and manage task progress throughout the project lifecycle.

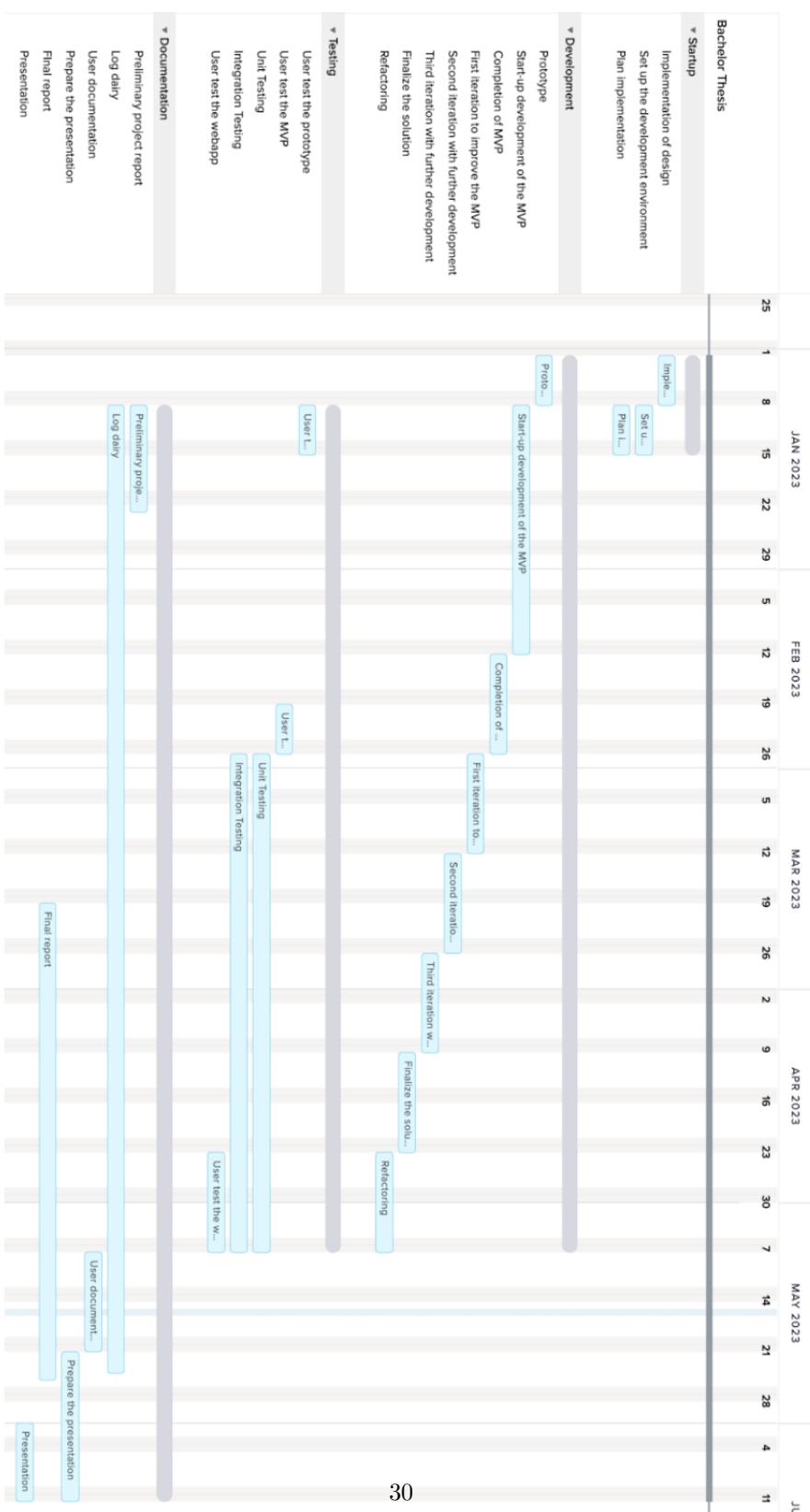


Figure 3.2: A progress Gantt chart of the project

### **3.1.1.5 Exploring PHP Laravel as the Framework Choice**

Our previous MVP was developed using PHP, so it was natural to take the next step and use a framework to improve the application. We learned that Laravel is one of the leading frameworks for PHP and most importantly has built-in security features. [Patnd] Another reason for choosing Laravel as a framework is that two of our team members had experience using Laravel at work. Therefore, we chose to use Laravel.

To prepare for the project, we decided to enroll in a PHP Laravel course before starting the planning phase. [Ari22] With the course, we aimed to expand our understanding of this programming language and framework before using it to create the application. We also wanted to help the rest of the group familiarize themselves with the framework more easily.

## **3.1.2 Under development**

### **3.1.2.1 Sprint Planning and Focus Areas**

In this subchapter, we discuss the formulation of our project plan, as depicted in 3.2. We structured our work into two-week sprints and identified the specific focus areas for each sprint. This approach allowed us to prioritize and allocate our resources efficiently. However, adhering strictly to the plan presented challenges, as we encountered unforeseen obstacles and shifting priorities throughout the project.

### **3.1.2.2 Deprioritizing some Requirements**

Due to many factors described in chapter ??, we had to prioritize and deprioritize requirements in the specification. This was done in collaboration with the course coordinator. We discuss the factors and things we took into account when making these decisions, emphasizing that the changes we made had a minimal effect on the overall project results.

### **3.1.2.3 Balancing Functional Requirements and Design Focus**

This subchapter delves into the importance of striking a balance between functional requirements and design considerations throughout the project. Initially, our emphasis was primarily on functional implementation, keeping in mind the overall/universal development process and our initial MVP 1.0 concept. However, as the project progressed, we recognized the need to allocate more attention to design aspects. We explore the rationale behind this shift and discuss how it was communicated and aligned with the course coordinator. Further details on this topic can be found in chapter ??.

### **3.1.3 Areas of Responsibility**

At the beginning of the project, our goal was that each member of the group could implement parts of the application. Therefore, as mentioned earlier, we purchased a course in Laravel so everyone had knowledge and experience using Laravel. Sadly not everyone could complete the course. After the project started, we quickly realized that this was not the best approach as there was a big gap in technical knowledge when it comes to coding and areas of interest. We solved this issue by allowing each team member to take responsibility for areas in which they were most interested and had the best knowledge. Therefore, early in the project, we thoroughly discussed each individual's preferences, areas of focus, strengths, and weaknesses. After reaching a consensus, we allocated tasks and responsibilities:

Name	Role	Responsibility
Gurjot Singh Aulakh	Scrum master and full-stack developer	Responsible for software development across the entire stack and leading the agile processes and practices within the team.
Mortaza Baqeri	Full-stack developer	Responsible for software development across the entire stack
Okbamichael Mussie	Technical Tester	Responsible for unit testing, feature testing, and API-testing
Arman Yedicam	Security Analyst	Responsible for the security of the product
Faesal al shhadat	UI/UX Designer	Responsible for the design of the product

Table 3.1: Individuals in Group - Names, Roles, Responsibilities.

## 3.2 Tools used in the process

This section provides a concise overview of the tools utilized in the project, along with a general description of each tool. Subsequently, we will delve into a detailed discussion on how these tools were effectively employed throughout the report.

### **Notion:**

*Notion* is a versatile platform that offers a unified space for thinking, writing, and planning. It allows for capturing thoughts, managing projects, and even running an entire company, all while offering complete customization to match one's preferences. [Labnd] Notion was an obvious choice for our team due to their familiarity with the platform, requiring minimal training.

### **Docker:**

*Docker* is a platform-as-a-service application that enables container creation, operation, and management. Virtual machines form the foundation of Docker, allowing for easy creation, distribution, and interaction. Any operating system or software architecture can seamlessly function with programs enabled by it. [Docnd]

### **Facebook Messenger:**

*Facebook Messenger* is a chat tool that operates through a web-based platform. [Metnd] The tool enables group members to share information and engage in informal conversations quickly.

### **Figma:**

*Figma* is a user-friendly tool that enables the rapid creation of visual prototypes for digital interfaces such as webpages. [Fignd] We utilized Figma to construct and test our prototype and refine it for further development. The resulting prototype was the foundation for our service's graphical representation and subsequent progress.

### **Github:**

Developers can work collaboratively on software projects using *GitHub*, a web-based platform that enables version control. [Gitnd] GitHub is built on Git and offers features such as bug tracking, task management, and documentation. Since our team is already familiar with GitHub, it is an ideal option for us.

**Microsoft OneDrive:**

*Microsoft OneDrive* provides the option for online storage. [Micnda] We saved all project-related files and materials on OneDrive and shared them among the group members. This feature enabled us to collaborate and work together on these documents.

**Microsoft Word:**

*Microsoft Word* served as our computerized writing tool. [Micndb] We used it to write all the reports and supporting documents for the project, enabling seamless collaboration and co-authorship on these materials.

**MySQL Workbench:**

*MySQL Workbench* is a comprehensive visual tool that unifies database builders, coders, and DBAs. [Orand] The report creators utilized it to create specific diagrams, such as database diagrams.

**PhpStorm (IDE):**

The group members were already familiar with editors such as IntelliJ IDEA, which made it a sensible choice for editing both the client and backend code. IntelliJ IDEA is widely used by software professionals worldwide, especially those who work with Java and Kotlin in enterprise, online, mobile backend, and full-stack applications. On the other hand, *PhpStorm* is specifically designed for developers who create online applications or command-line tools using PHP or PHP-based solutions. [Capnd] [Jetnd] As a result, no training was required since the group members were already familiar with these editors.

**Postman:**

The tool, *Postman*, is designed for evaluating APIs. This utility provides a convenient way to assess APIs. With Postman, users can transmit API endpoint inquiries with the desired parameters in the "header" and "body" by the HTTP protocol.[Posnd]

**Zoom.us:**

*Zoom.us*, a video conferencing tool, was utilized for guidance meetings with an internal supervisor at OsloMet and all group meetings. [Comndb]

### 3.3 Framework/library used

In this section, we will provide a brief and general overview of the frameworks and libraries that were utilized in the project. This serves as an introduction before we delve into a more detailed explanation of how we employed them and the connections between them later in the report.

#### **Blade:**

The *Blade* is the scripting engine that comes bundled with Laravel and is simple and powerful. We opted to use the blade as our frontend framework since it is integral to Laravel. [[Larnda](#)]

#### **Bootstrap:**

*Bootstrap* is the go-to CSS framework for developers who want to build mobile-friendly, responsive web pages that are flexible and adaptable to different screen sizes. [[W3Snd](#)]

#### **PHP – Laravel:**

*Laravel* is a web application platform that boasts an elegant and expressive syntax, which enables developers to focus on the creative aspects of building applications rather than getting bogged down in details. By providing a solid foundation, Laravel makes it easier to focus on creating rather than worrying about the finer development points. [[Otwnd](#)]

#### **JavaScript:**

*JavaScript (JS)* is a computer language with first-class functions that can be interpreted or, just-in-time, compiled. JavaScript originated as a web scripting language and has evolved into a universal language that finds application in various contexts beyond web development. Various non-browser environments such as Node.js, Apache CouchDB, and Adobe Acrobat utilize it, although it is most well-known as the scripting language for websites. [[nd](#)]

#### **Mockery:**

The *Mockery* is a flexible PHP fake object system that enables easy utilization for unit testing with PHPUnit or other testing frameworks. Its main goal is to offer a concise API that uses a Domain Specific Language (DSL) that can precisely define all possible object operations and interactions in a human-readable manner. [[Bcnd](#)]

### **PHPUnit:**

*PHPUnit* provides an efficient and effective way to perform unit tests on PHP code, allowing developers to identify and fix bugs quickly. It offers a range of features, such as mock objects and code coverage analysis, that help ensure the quality and reliability of PHP applications. [Bernd]

### **Swagger:**

The *Swagger* framework is responsible for automatically generating descriptions of a REST API. Swagger is beneficial for maintaining and developing the solution as it provides a clear and structured overview of API interaction without requiring familiarity with the underlying code or structure. Incorporating Swagger declarations into the API enables the quick creation of a website with visual representations of all endpoints and their specifications. Facilitation of automated documentation generation and rapid assessment of API communication via the website is enabled. [Sofnd]

### **MySQL:**

Popular relational database management system *MySQL* is open source. TSQL, or Structured Query Language, manages and manipulates table data. MySQL is a well-liked option for web-based applications and data-driven websites because of its reputation for dependability, usability, and performance. We use MySQL because it can easily be coupled with other programming languages like PHP and Python and can handle significant volume and high throughput data. [Aul+23]

### **Laravel Sail:**

Simplifies working with Laravel's default Docker development environment by providing a lightweight command-line interface. It includes a docker-compose.yml file and a sail script at the root of your project, allowing you to interact with Docker containers conveniently. Sail is suitable for building Laravel applications using PHP, MySQL, and Redis, even without Docker experience. It is compatible with macOS, Linux, and Windows (via WSL2). [Larndf]

### **xDebug:**

xDebug is an effective PHP debugging tool that aids developers in efficiently identifying and resolving code errors. By incorporating xDebug into PHPUnit and Laravel, developers can enhance the debugging capabilities of their unit tests. This integration enables them to better track test runs, examine variables, and identify code errors. Particularly for Laravel applications, xDebug facilitates step-by-step debugging, the establishment of breakpoints, and analysis of the application's behavior

during testing. [\[Xdend\]](#)

**Livewire:**

Livewire is a comprehensive framework designed for Laravel, which simplifies the process of creating dynamic interfaces while staying within the familiar Laravel environment. It enables developers to construct contemporary web applications using Vue and React, minus the additional intricacies that could complicate their workflow. [\[Livnd\]](#)

## **3.4 Working method**

We used a flexible approach for our project and opted for the agile methodology. This helped us quickly create and test a prototype to confirm or refute our assumptions about the solution. We have used various methods within agile methodology in our project, which we will discuss individually and provide reasoning for their selection.

### **3.4.1 Agile development**

The group decided to work according to an agile framework called Scrumban, combining the two methodologies, Scrum and Kanban. We reached this conclusion because both frameworks contribute value in different areas, and we required qualities from both. [Atlnd]

### **3.4.2 Scrumban**

Scrumban is a hybrid approach combining elements of both Scrum and Kanban. We decided to use Scrumban due to its high relevance in the IT industry. [San22]

The group acknowledged the inherent uncertainty and potential for unforeseen changes in requirements. For example, while planning for some of the sprints, the group remained flexible to accommodate any alterations in the requirements specifications along the way. The agile methodology allowed the team to respond proactively to changes while maintaining a structured project plan.

### **3.4.3 Scrumban in practice**

The project was structured into six sprints lasting for two weeks and seven sprints lasting for one week. Tulpesh conducted sprint review meetings weekly to present our progress and contributions. The sprint review meetings duration was one hour. We also held retrospective meetings to ensure that the team could reflect on the process and make necessary adjustments. These meetings allowed

all team members to share their opinions on what had gone well, what had gone poorly, and what could be improved.

During the planning phase of the first sprint, we created a backlog to keep track of all the tasks in the project. We continuously updated this backlog with new tasks as the project progressed. Initially, these tasks were based on the user stories of the application.

We implemented a Kanban board in Notion. Subchapter [4.1.2](#) discusses the rationale behind selecting Notion and its utilization.

To ensure alignment among team members, we conducted daily 15-minute stand-up meetings at 10 am on Mondays, Wednesdays, and Fridays. These meetings involved brief reviews of individual accomplishments since the previous meeting, discussing intended goals, and identifying obstacles.

Gurjot, with his prior experience in agile development methodologies, assumed the role of the project's scrum master.

# Chapter 4

## The development process

The project has progressed through different phases with well-defined timelines, while some procedures have remained constant throughout the entire project duration. The project's duration spans 20 weeks, and to streamline the workflow, we divided it into nine sprints.

### 4.1 Start-up phase

We will now provide an overview of the distinct sprints and phases consistently present throughout the project's timeline.

#### 4.1.1 Design and Prototype Sprint

Design sprints are a structured process that rapidly develop and test prototypes to validate ideas and solve challenges. They foster collaboration and innovative problem-solving, with key objectives including idea validation, user feedback, and practical solution generation. Design sprints offer benefits such as rapid iteration, risk reduction, teamwork, and vision alignment. [two22]

Our design sprint team consisted of five individuals with varied skills and expertise, driving our achievements. Through collaboration and leveraging our diverse skill sets, we generated well-rounded solutions that addressed project aspects. We prioritized functional requirements over visual design ideas, given the project requirements.

During the first week of the design sprint, we split into two groups, one focusing on backend implementation and the other on design ideas. This approach allowed for independent thinking and a wide range of suggestions. In the second week, we shifted to prototype development, consolidating the best elements from individual proposals and making collective decisions on design direction. Technical considerations were also addressed, and we chose Figma as the tool for implementing the prototype.

We conducted website research on matchmaking internship solutions to gain insights and inspiration, analyzing three specific websites for user flows and interfaces. This research influenced the development of our prototype.

At the start of the project's 3rd sprint, we tested the prototype with our course coordinator and gathered feedback on functionality, user interface, user flow, and the overall concept. The feedback was predominantly positive, with some minor issues related to language usage and images. We iterated on the prototype to align with specifications and the coordinator's feedback. To see the final version of the prototype, check Appendix [A.2](#).

#### **4.1.2 Implementation planning and set up of the software environment**

We utilized Notion to create a dedicated project board to manage our project effectively. This board served as a centralized hub for distributing tasks and responsibilities among team members. It allowed us to track progress on specific functions, complete administrative tasks and projects, and monitor individual responsibilities. We utilized the project board to review our progress during our stand-up meetings. Although this approach was new to the group, we chose Notion due to its familiarity among team members and prior experience using the application. Implementing this approach helped us structure our work process and enhance overall productivity.

To effectively manage the task backlog, shown in figure 4.1. This board provided a complete overview of all the project tasks. We divided each task into sub-tasks to simplify problem-solving and closely monitor task progress. Figures 4.2 and 4.3 offer visual representations of this approach.

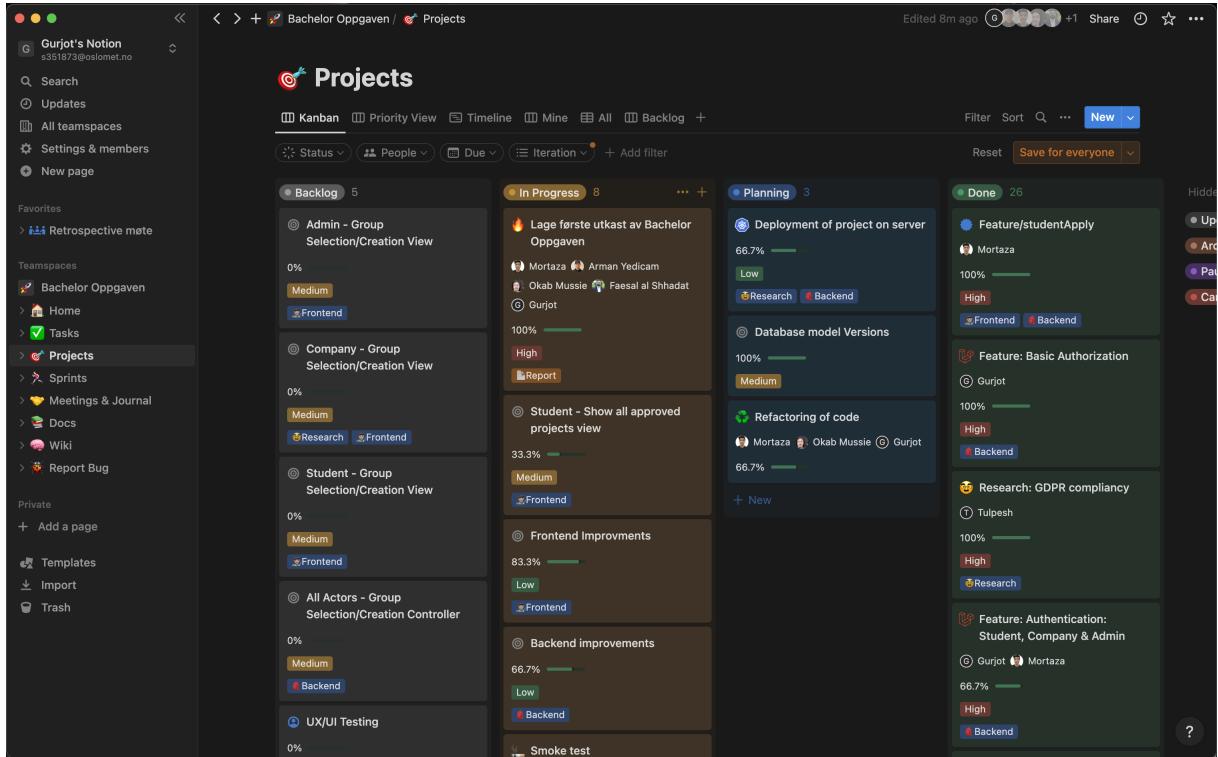


Figure 4.1: "Kanban" project board created in Notion.

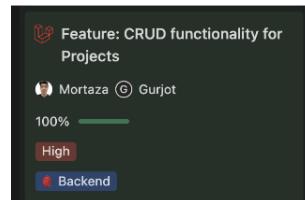


Figure 4.2: A task from Notion

Task Name	Assign	Status	Priority
Create a basic db migration	Mortaza, Gurjot	Done	Medium
Create Listing Model	Mortaza, Gurjot	Done	Medium
Create ListingFactory	Mortaza	Done	Medium
Create db seeding for Listing	Gurjot	Done	Low
Create routes	Mortaza, Gurjot	Done	Medium
CRUD - functionality	Mortaza, Gurjot	Done	Medium
Admin have to approve project proposal	Gurjot, Mortaza	Done	Medium

Figure 4.3: Table of sub-tasks as viewed in the Task section of Notion.

To categorize tasks, we used tags. Each task was represented by a card on the Kanban board, allowing us to assign priority, deadline, responsible person, and iteration for completion. We utilized different board views to enhance task management:

1. Priority View: This view listed tasks based on their priority, difficulty level, and iteration association. It helped us prioritize critical tasks and allocate our time and resources effectively. See

figure 4.4 for an illustration.

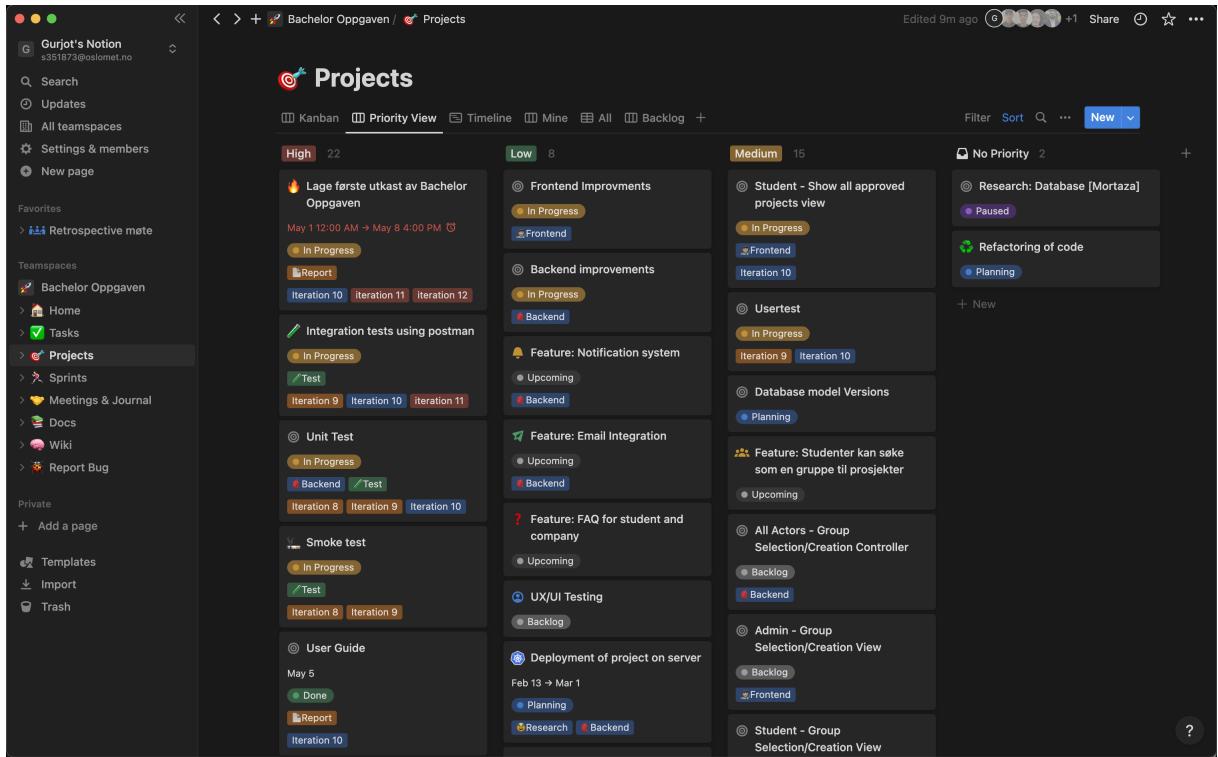


Figure 4.4: Priority View from Notion.

2. Timeline View: We utilized a Timeline view to visualize the sprints and understand the task sequence and inter-dependencies.
3. Mine Board: Each team member had a Mine board displaying tasks assigned to them.
4. All Board: Each team member had an All board that showed all the tasks. This approach facilitated task visibility and effective collaboration within the team.
5. Backlog Board: The Backlog board displays all the tasks currently in the backlog.

These board views and categorizations allowed us to manage our tasks efficiently, ensuring clear visibility, effective collaboration, and streamlined progress throughout the project.

## **4.2 Development phase**

The development phase consists of 13 sprints, each with its specific focus areas, as indicated in the table below. The first six sprints lasted two weeks each. After this period, we reduced the sprint duration from two weeks to one week. The reason we halved the sprint duration was because we understood that gaining feedback from the course coordinator every week was much more efficient for improvement than using two weeks per sprint. In other words, the group's efficiency increased when switching from two to one week sprints. It is also important to note that the front- and backend implementation as well as the testing was done every week until sprint 11.

Sprints	Week Number	Description
Sprint 1	2 & 3	Planning and creation of preliminary project report
Sprint 2	4 & 5	Design and prototyping of frontend
Sprint 3	6 & 7	Security analysis, backend development and smoke testing
Sprint 4	8 & 9	Implementation of frontend and survey planning
Sprint 5	10 & 11	Developing T&Cs, implementing tests, and discussing survey methods
Sprint 6	12 & 13	Usertesting and UI/UX improvements
Sprint 7	14	Code refactoring and restructuring
Sprint 8	15	Building a complete user guide for all actors
Sprint 9	16	Creating project poster to present at IT-Expo
Sprint 10	17	Planning the execution of the Bachelor report and distributing the work based on our roles
Sprint 11	18	Creating a template and an early draft of the Bachelor report
Sprint 12	19	Writing the Bachelor report
Sprint 13	20	Finishing the Bachelor report

Table 4.1: Sprints overview

Throughout the entire development process, the project team maintained a flexible workflow, resulting in a highly positive group dynamic. In situations where individual motivation waned, the group managed to sustain the dynamics by being open about personal challenges. The group was also willing to support and help each other to contribute to the project in the best possible way. This created a conducive collaborative atmosphere and provided significant support to each team.

member.

In the upcoming chapters, the various phases will be discussed in greater detail, providing more insights and elaboration.

During the implementation of the MVP, we experienced that it took longer than expected, and it turned out that the application was more demanding to implement than anticipated. As a result, it was impossible to develop in several iterations as we had planned in the Gantt chart. Additionally, we wanted to add extra functionality, which included the ability for "digital signing of contracts," where students could sign work and confidentiality agreements through the application. We also planned to conduct user tests with companies, but this was postponed due to lack of approval in time from [SIKT](#) (Appendix [A.7](#)).

Along the way, there were also time-consuming activities such as working on the project report and creating a poster, see appendix [A.13](#)

### **4.2.1 Development of MVP**

We began by implementing the main functionality of the application and established database models based on the requirements specifications, appendixes [A.8](#) and [A.9](#).

The team focused heavily on backend development and creating activity and user models. The application's appearance could have been more emphasized initially, as the design was simple and intuitive. Most of the focus was on functional requirements over non-functional ones, seamlessly integrated with the implemented functionality. The goal was to ensure users could use the features effectively and efficiently.

#### **4.2.1.1 Docker Container Hosting**

We started setting up the Docker container by creating a repository on GitHub called "BachelorOppgaven2023". We used PhpStorm as our development environment for the project. We started with a

predefined docker file from Sail and made modifications to create five Docker containers, see appendix A.12.

Docker Containers	What were they used for?
Selenium-1	To run automated tests
Redis-1	To handle caching of data and to accelerate database processing
MySQL-1	Contained the database and was used to store and manage data of the application
Mailhog-1	To capture and display email messages sent from other containers in the system
Sim-1	The container where the entire application was located and could be run.

Table 4.2: Details about docker containers

*The entire application (front- and backend) was placed in the Docker container called "sim-1".*

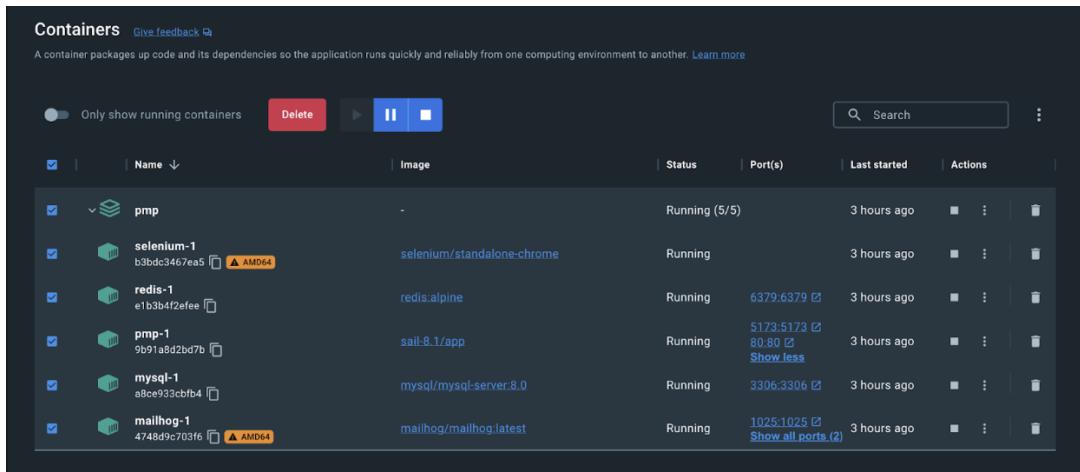


Figure 4.5: Docker Containers

#### 4.2.1.2 Hosting

Early in the process, we decided to host our web application so that we could easily perform user tests. Therefore, we researched how to deploy a Docker-containerized Laravel web application using online resources.

We contacted the IT department at OsloMet to get help deploying our web application. After talking to them, we were given access to a Linux VM hosted on OsloMet servers to deploy our web application. However, some errors prevented us from completing the deployment anyway. We discovered the problem in the container "pmp-1" (which was later renamed to "sim-1"), as shown in figure 4.6, but finding the error and fixing it was tough. Details about this error can be found in appendix A.10.

```
pmp_1 | 2023-04-26 10:11:47,850 INFO spawned: 'php' with pid 8

```

Figure 4.6: Deploying error from OsloMet VM

After not being able to solve the previous issue with the Linux VM, we tried to deploy the web application on Microsoft Azure. We needed more expertise in Microsoft Azure, therefore, we contacted many of our acquaintances and asked for help to deploy. Unfortunately, none had enough knowledge and time to assist us. Nevertheless, we did not give up and tried again. Eventually, we managed to deploy the web application, but we experienced an error that prevented us from getting it to work correctly, as shown in figure 4.7.

## : ( Application Error

If you are the application administrator, you can access the [diagnostic resources](#).

Figure 4.7: Application error from Microsoft Azure

Details about this error can be found in appendix [A.11](#). We found out that the problem was in the docker file.

Since hosting was not a part of the requirements specification, we chose not to spend any more time on this and would consider it later if we had enough time remaining.

#### **4.2.1.3 MySQL Database**

In this project we have used MySQL as our database client. We use MySQL because it can easily be coupled with PHP and Laravel framework and also because it can handle significant volume and high throughput data. MySQL database is run through docker, as a docker container. Figure [4.8](#) illustrates our final version of SIM application.

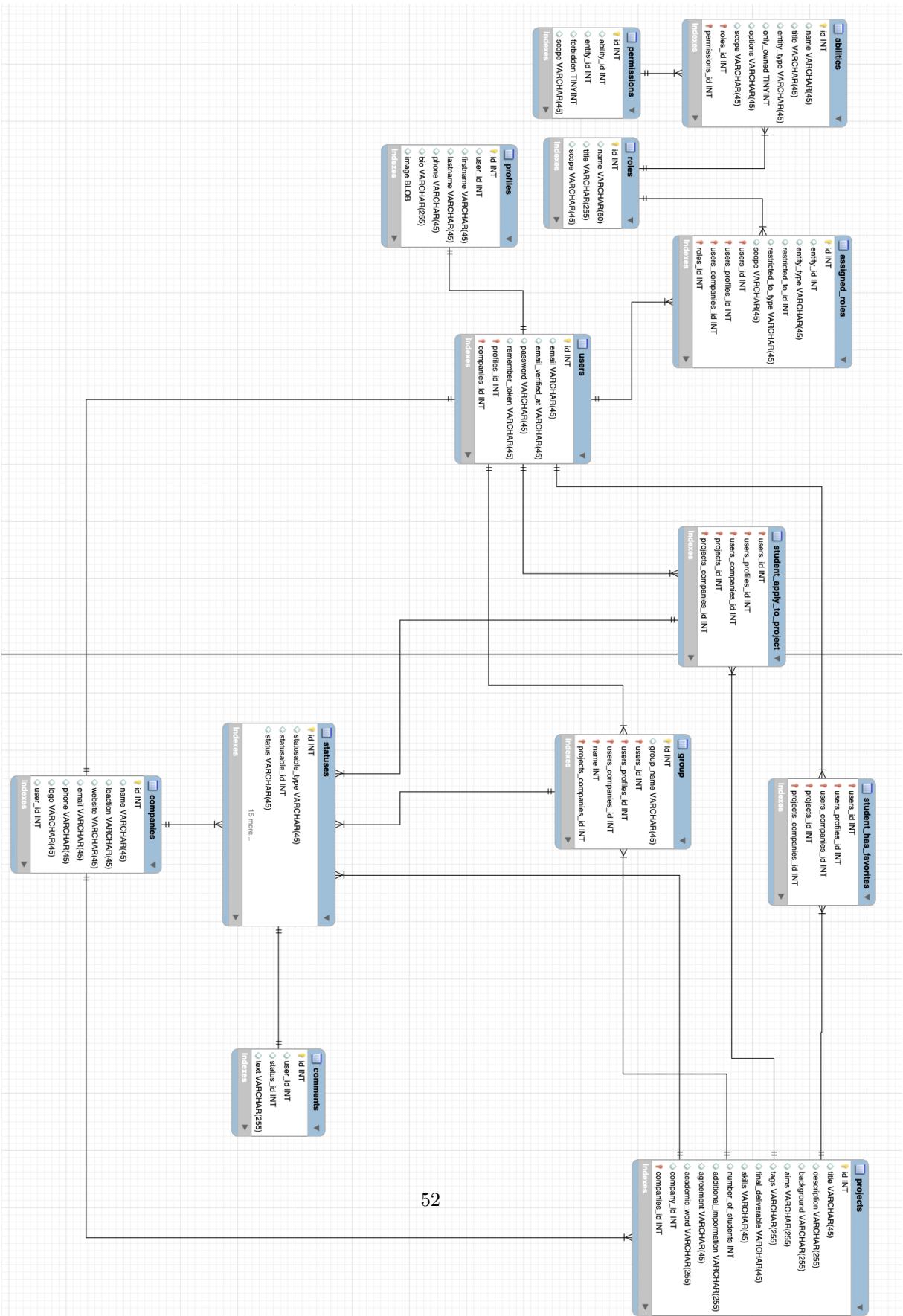


Figure 4.8: Final version of SIM's database

#### **4.2.1.4 Single Sign On**

SSO, or “Single Sigh On (SSO),” is an authentication method that allows users to log into multiple applications with a single login. This means users do not have to create separate accounts or log in multiple times to access different systems. Instead, they can use one login to access all the other systems they need to use, making it easier and more efficient to work across other platforms and applications. [App23] We considered various options and decided to use Feide login, a popular SSO solution in Norway.

In the meantime, we encountered several strict requirements to use Feide login, including the need for at least one person on the team to be a master’s student and the team to have a company with an organization number. Unfortunately, we did not meet these requirements and could not use Feide login. Read more about this in section [5.1.4](#).

#### **4.2.1.5 Development of the front end with the blade**

During the implementation of Blade, most things went smoothly. The blade had some limitations that made it difficult to implement certain parts of the application, especially parts that needed to be dynamic, such as favoriting projects. We used Blade syntax to implement the frontend. For more information, see section [5.1.2](#).

### **4.2.2 Documentation**

Here, we present the documentation methods used for the entire project.

#### **Project Diary**

We have chosen Notion as our project diary and documentation tool. Notion has served as a log for what has been done when it was done, and who worked on what in different stages of the project. We have also used Notion to document the entire project, the process, and the required tasks.

#### **User guide for a solution**

We have created a detailed platform description that can be used as a user guide. The user manual will clearly explain how our solution can be used. See section 7, “User Guide.”

#### **Final report**

Our bachelor thesis summarizes the project, including its challenges and successes. The report will also reflect on what we could have done differently and what we have learned from the project.

#### **4.2.3 Correspondence between the project plan and actual implementation**

The goal of the entire project plan was to establish an operational plan for the project to approach the development of the application with efficiency. The implementation of the product was a crucial part of this plan. The group developed the entire plan collaboratively, based on requirement specifications and project analyses, the plan aimed to manage our work with time as a critical factor. Roles were assigned between the group during the project phase to ensure efficient execution of the tasks.

The group discussed important points such as the requirements specification and GDPR updates. The team had to be flexible and adapt to changes, ensuring alignment with the course coordinator’s needs. Taking an iterative approach to development allowed continuous testing and validation, saving time in the long run.

The implementation was similar to the project plan, but some things needed to go according to the plan. This was because the implementation process sometimes depended on responses from people outside the group. Therefore some tasks had to be put on hold, resulting in a different approach than the project plan.

#### **4.2.4 Quality assurance**

During the development phase, Notion and GitHub were used as tools to gain easier access to the entire project in a structured and easy way. The applications provided value to our product by giving us free comprehensive platforms that made collaboration and communication easier in the software

development process. The applications made it possible for the members to edit and view the same documents. Therefore, if something was wrong or unclear, any of the members had the possibility to comment or edit. When met with challenges requiring a collective review, the responsible team members would conduct a review. Additionally, commenting on these challenges in the Kanban board on Notion was possible. All members had access to the Kanban board to comment and share their thoughts on the different tasks. If anyone had any incidents or questions, they were taken forward and discussed during the stand-up meetings.

## 4.3 Reflection and Discussion

### 4.3.1 Challenges around basic setup of coding environment

The initial setup of the code environment was a significant challenge. We were determined to avoid the common issue of “It works on my machine, but not on yours” and decided to utilize Docker. However, our prior knowledge of Docker was purely theoretical, and we had no practical experience in creating and running a Docker container for a web application, which made the task daunting. We encountered numerous hurdles during the setup and execution of Docker, but thankfully, Laravel came to our rescue with Sail.

Sail simplified the process of configuring the code environment with Docker, however it introduced additional complexity to other aspects of the setup. For example, difficulty arose when trying to configure phpStorm to utilize the docker containers for running the application and the automated tests. Furthermore, Sail made it difficult to configure xDebug.

Lastly, the inability to deploy or host the application meant that we could not set up the development, staging, and production environments as intended. This hindered our ability to follow the planned Git workflow effectively, furthermore it made it more challenging for us to perform user test of new features, because we had to wait for our meeting with our course coordinator to show what we had made and get feedback. Further details on this topic have already been discussed previously in [4.2.1.2](#).

We encountered many challenges and it was incredibly time-consuming to set up the development environment. We spent multiple weeks in the beginning of the project to solve these issues, but ultimately solving the issues made the development process much easier.

#### 4.3.2 Challenges regarding changes in software requirement

At the start of bachelors, we had discussions about the key points of the project with the entire group and course coordinator and created product requirements. Initially, the focus was on the functionality and security aspect of the application. Although we had initially envisioned the application to be visually appealing and accessible to all users, we did not prioritize this aspect significantly as it was not specified in the product requirements provided by the course coordinator.

Later in the project we got many changes in requirements from the course coordinator. It was both functional and nonfunctional changes, but the biggest change was that UI/UX was now heavily prioritized. We performed many small feature tests with our course coordinator, and it was apparent that the UI/UX design was important as we got a lot of feedback on how to improve the UI/UX from these tests.

This meant that we could no longer follow our initial plan of focusing mainly on backend and security and we spent a lot of time refining and creating the best possible frontend.

We had to address these new requirements and try to tackle them as quickly as possible as they came up to stay on track with the project plan. This resulted in delays from the original plan. This also rendered many of our initial designs useless as it was working but not up to standards in terms of UI/UX. We ended up relying on custom bootstrap for most of the development as we knew it was created by professionals.

Throughout the course of the project, we received several updates to the functional requirements from our course coordinator. These updates encompassed additional functionalities as well as improvements to existing ones. The list of functional changes can be found in figures [4.9](#), [4.10](#), and [4.11](#). While some of these modifications may appear minor, they did contribute to project delays due to the time required for implementation. We have made significant progress in developing most

of these new features, with some nearing completion. However, we ultimately made the decision to prioritize the completion of the bachelor's report before resuming work on the application. To see which new requirement we have started working on, refer to chapter [5.2](#).

FRCC-8	Have an overview of which students are registered to which project(s)	MIDDLE
FRCC-9	See the status of students in the sign-up phase (i.e., confirmed member of a project, under consideration, not signed up for a project)	MIDDLE
FRCC-10	Edit project groups	LOW
FRCC-11	Register submission of formal documents (Collaboration Agreement and NDA for each)	LOW
FRCC-12	Archive Project Proposals and Descriptions	MIDDLE
FRCC-13	Remove/Archive company profiles	MIDDLE

Figure 4.9: New functional requirements for course coordinator

FRS-5	Register their interest for a project (individually or as a pre-defined group)	HIGH
FRS-6	See who is in the final group	LOW
FRS-7	Can contact the other groups members	MIDDLE

Figure 4.10: New functional requirements for Student's functional requirements

<b>FRPP-6</b>	Have an overview of students that have registered interest in their project(s)	MIDDLE
<b>FRPP-7</b>	Can accept/reject students	LOW
<b>FRPP-8</b>	See who is in the final group	LOW
<b>FRPP-9</b>	Have the name and contact details of the Internal Supervisor attached to the project	LOW

Figure 4.11: New functional requirements for project provider

#### 4.3.3 Challenges in development of MVP

In the development of the current MVP, we chose not to continue developing the MVP we had from the previous semester. Though most of the core functional requirements for the application remained the same, there are multiple reasons for starting on scratch rather than work on the previous code.

One of the primary reasons for not continuing with the previous MVP was because the previous MVP was implemented in pure PHP and lacked security features. Recognizing the importance of enhanced security, we made the decision to adopt the PHP framework, Laravel, for the new MVP. Laravel offers integrated security features, as outlined in section 5.5. Instead of attempting to retrofit the previous MVP to utilize the framework, it was more efficient to start anew with a fresh Laravel instance.

Finally, the code for the previous MVP lacked proper structure and the database was not well-designed as it did not have proper relationships. This could lead to potential scalability issues for the application in the future, furthermore the code could be difficult to work on especially for new developers who would take over the future development. Our decision was easily made due to the combination of these factors and the chance to expand our knowledge and experience by utilizing a new framework.

#### **4.3.4 Challenges and reflections about the frontend**

As mentioned before we have used Laravel Blade for the frontend. Though Laravel Blade made it easier to create the frontend because it is easy to understand and allows us to write plain php code in our template, there are some drawbacks of using Blade as well. For example, creating dynamic and responsive user interfaces in Laravel Blade is difficult alone, and requires the use of a tool like Livewire.

Late in the development process we realized that using a frontend framework like Angular would have made the development process and setup more difficult, but it would also provide us with several benefits. It would for example allow for a clear separation between the frontend and backend of the application, while Laravel Blade tightly couples server-side rendering with the backend logic. Angular's separation would make it easier to maintain and scale the project as we could separate the backend and frontend groups fully and work independently on our respective parts.

Furthermore, Angular is a big framework and offers a robust set of tools and components for building dynamic and interactive UI. It is also well-suited for building SPA's, which would provide a smoother user experience and could also reduce load time for rendering pages. In the future, it would be possible to change the application to use a frontend framework like Angular. Considering this, we incorporated a Swagger API documentation, which is covered in detail in Chapter [5.6.1](#).

#### **4.3.5 Challenges in deprioritizing some Requirements**

Due to time constraints and changes in requirements, we had to make a prioritization decision and deprioritize some requirements in order to focus on the most important ones. This allowed us to successfully complete the key and high-priority requirements as described in chapter [2.2](#). By adopting this approach, we were able to ensure that we met the core objectives and delivered the product on time.

## Chapter 5

# Product Documentation

The product documentation will comprehensively explain the application's structure and features.

## 5.1 Description of solution

We have a fully functional application to offer. This chapter will provide details on how we have implemented the application. We have utilized support libraries for both front and backend development, which has been referred to in the source code. These libraries are defined in a 'composer.json' file.

### 5.1.1 Architecture

The Laravel application with Blade syntax on the frontend follows a traditional server-side architecture, where the backend handles requests, processes data, and generates responses. This structure employs the Model View Controller (MVC) pattern, with models representing data structures and database tables, controllers managing requests, and views rendering HTML output. The Laravel

framework includes built-in features and tools to create scalable applications while ensuring code uniformity and maintainability.

### MVC (Model View Controller)

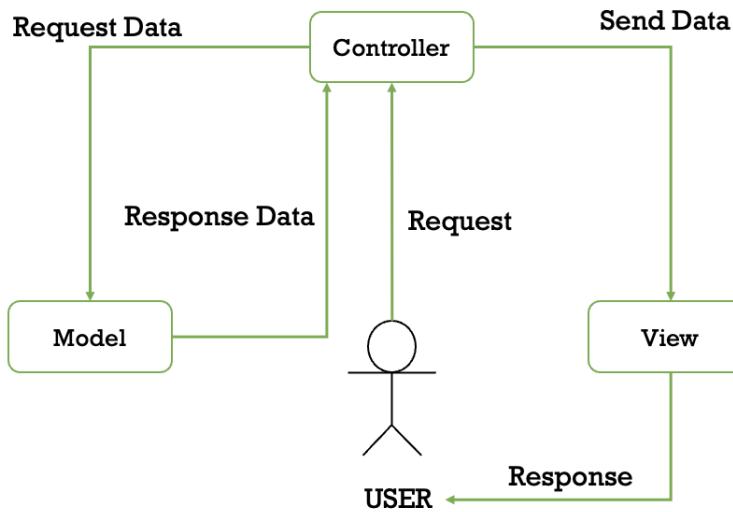


Figure 5.1: Shows how model view controller works

#### 5.1.2 Frontend

Laravel Blade is a templating engine with the Laravel PHP framework. It is simple yet powerful and lets you use plain PHP code in your templates. All Blade templates are compiled into plain PHP code and cached until modified, meaning Blade adds zero overhead to your application. Blade template files use the ‘.blade.php’ file extension and are typically stored in the ‘resources/views’ directory. [\[Larnda\]](#)

Laravel Blade is used for implementing the frontend. We opted for Laravel Blade because it can incorporate plain PHP into our codebase, simplifying the development process. Additionally, regular

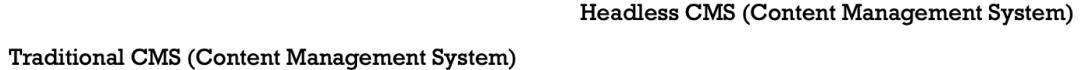


Figure 5.2: Traditional Content Management System

Figure 5.3: Headless Content Management System

HTML in the code makes it easy to understand and work with.

However, it's worth noting that unlike websites built using React or Angular, the website we created using Laravel Blade is static. This poses a challenge when implementing specific features, such as favoriting a project. Dynamic changes to the frontend can only be achieved by using JavaScript or frontend frameworks like React or Angular.

Despite this limitation, the benefits and ease of use of the Laravel blade outweigh its drawbacks. Moreover, our group only has a little experience with frontend frameworks, and learning them would require a significant amount of time, which wasn't feasible given our time constraints.

The application's frontend uses custom Bootstrap and custom JS files taken from a course. [Ari22]

### 5.1.2.1 Architecture

The source code in the front end is separated into two folders, "resources" and "public."

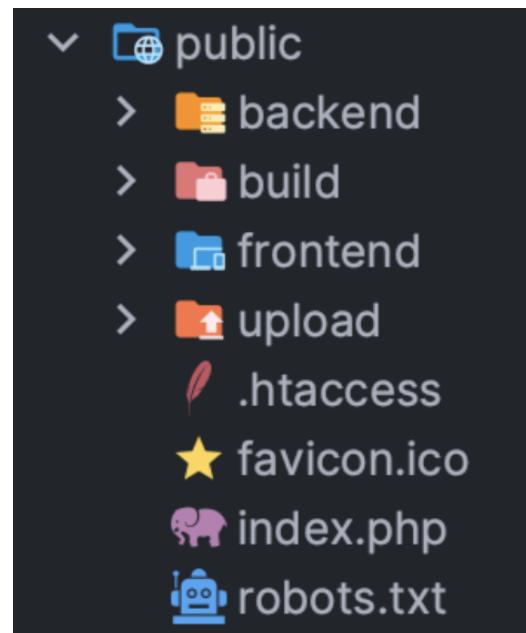
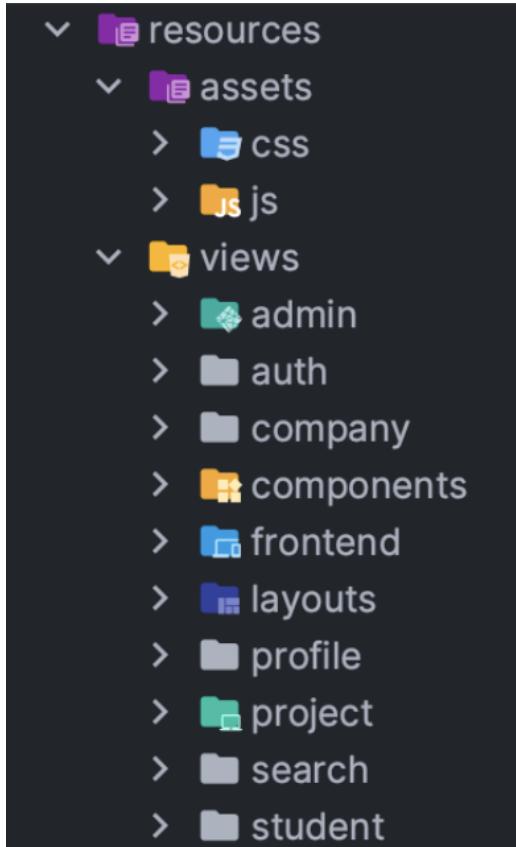


Figure 5.4: Resources folder

Figure 5.5: Public folder

## Resources

Resources help to manage and organize assets such as views, images, style sheets, and scripts. Laravel offers a flexible and robust approach to resource management, making it easy to keep everything organized and accessible.

One of the most significant advantages of using resources in Laravel is the ability to separate the presentation layer from the application logic. This makes it simpler to maintain and modify the codebase as it grows. It also allows for the reuse of components across different application parts,

reducing development time and enhancing code quality.

Resources in Laravel are generally organized into directories based on their purpose. For instance, views are typically stored in the resources/view's directory, while assets such as style sheets and images are kept in the resources/assets directory. Additionally, Laravel provides a way to organize resources based on the context in which they are used through resource routing.

## Public

The public folder contains files and code that is accessible to the web. This includes CSS, Javascript, and other assets needed for the website's frontend.

### 5.1.3 Backend

The backend is part of a software application that runs on the server and handles data processing and server-side logic. The backend is essential to maintaining application performance and security and directly impacts the user experience. Typical tasks in the backend include creating and maintaining databases, communicating with third-party APIs, implementing security features, managing server configurations, and running critical processes that cannot be run on the client.

The organizational structure of Laravel's backend comprises several levels, each with distinct responsibilities. Its design follows the Model View Controller (MVC) pattern, as shown in figure 5.1, with the Model layer responsible for data management, the View layer for displaying data to users, and the Controller layer for managing user requests and facilitating communication between the Model and View levels.

The Model layer is responsible for data modeling and manipulation, while the View layer consists of HTML, CSS, and JavaScript files comprising the user interface. The Controller layer mediates between the Model and View levels, handling user requests and directing data flow.

This layered architecture promotes the separation of concerns, allowing each layer to be modified or updated without affecting the others. It also enables individual layer testing, which is essential for developing robust and sustainable software.

We have divided our backend into multiple layers. These layers consist of routing, controller, model, view, and database, as shown in figure 5.6.

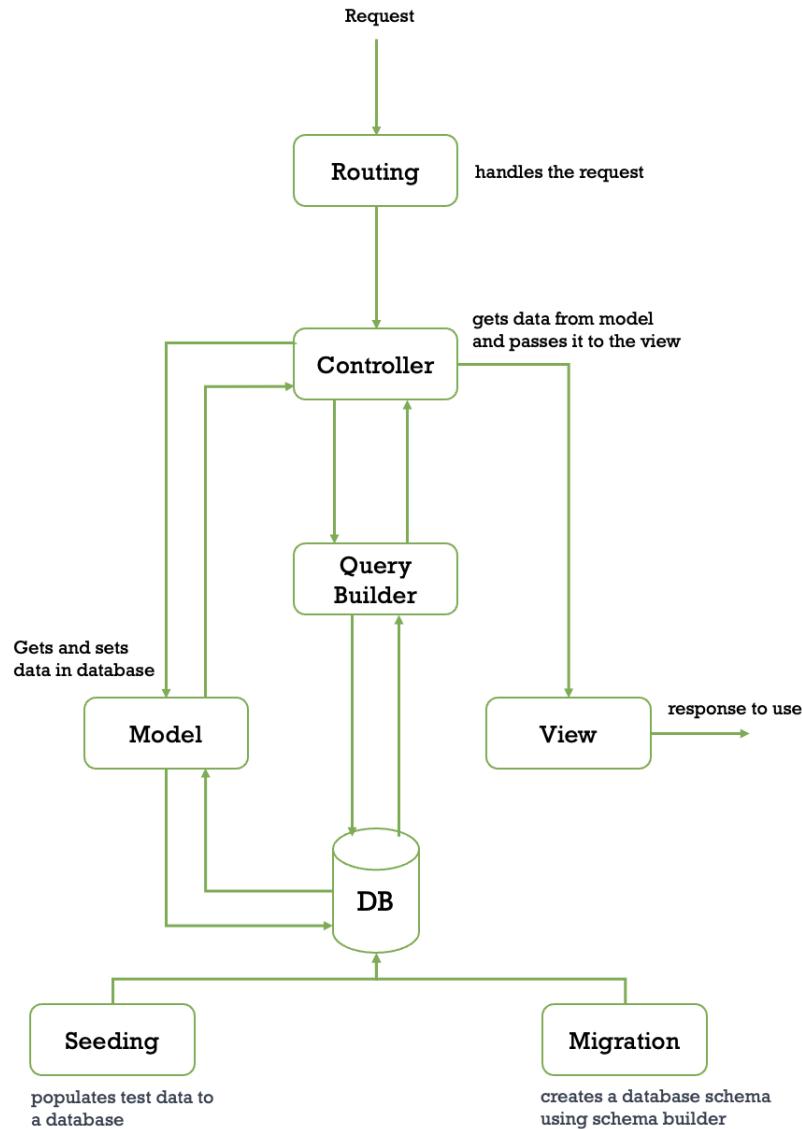


Figure 5.6: The layering of the backend

### 5.1.3.1 Models

The Models folder is in charge of data logic, which includes data access, manipulation, and validation. The models serve as a bridge between the database and the application code, representing the data units in the application.

Laravel uses Eloquent Object-Relational Mapping (ORM), allowing developers to communicate with the database using PHP classes and objects instead of writing complex SQL queries. It is much easier to perform CRUD operations by associating database tables with model classes. [[Larndc](#)]

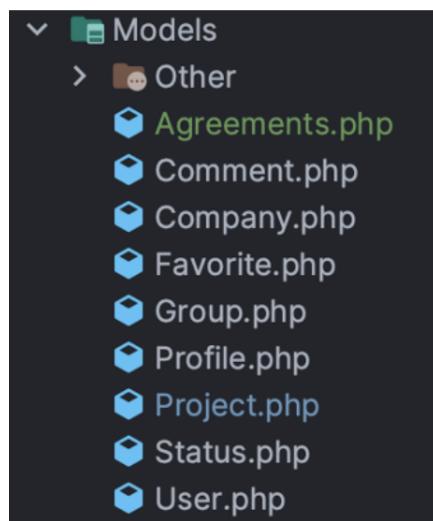


Figure 5.7: Model's folder

One of the core Models in our application is the Project model. The HasFactory tells us that this model has a factory class. Factory classes are used to create fake instances of a model. The naming convention is to combine the model name with the Factory suffix. In this example, the factory class for the Project model is ProjectFactory.

```
<?php  
|  
namespace App\Models;  
  
use ...  
  
class Project extends Model implements IStatusable  
{  
    use HasFactory;  
  
    /** The relationships that should always be loaded. ...*/  
    protected $with = ['company', 'latestStatus'];  
  
    public const STATUS = Status::class;  
    public const COMPANY = Company::class;  
    public const GROUP = Group::class;  
    public const USER = User::class;  
  
    /** The database table used by the model ...*/  
    protected $table = 'projects';  
  
    protected $fillable = [  
        'title',  
        'description',  
        'background',  
        'aims',  
        'tags',  
        'final_deliverable',  
        'skills',  
        'number_of_students',  
        'additional_information',  
        'agreement',  
        'academic_work',  
        'company_id',  
    ];
```

Figure 5.8: Project's model

```

<?php

namespace Database\Factories;

use ...

/**
 * @extends \Illuminate\Database\Eloquent\Factories\Factory<App\Models\Project>
 */
class ProjectFactory extends Factory
{
    /**
     * Define the model's default state.
     *
     * Returns: array<string, mixed>
     */
    public function definition()
    {
        return [
            'title' => $this->faker->sentence(),
            'description' => $this->faker->paragraph( nbSentences: 2), // n of sentences
            'background' => $this->faker->paragraph( nbSentences: 2), // n of sentences
            'aims' => $this->faker->paragraph( nbSentences: 2), // n of sentences
            'tags' => 'laravel, api, backend',
            'final_deliverable' => $this->faker->paragraph( nbSentences: 2), // n of sentences
            'skills' => $this->faker->paragraph( nbSentences: 2), // n of sentences
            'number_of_students' => $this->faker->numberBetween( int1: 1, int2: 5),
            'additional_information'=> $this->faker->paragraph( nbSentences: 2), // n of sentences
            // TODO: add a new table with agreements
            'agreement' => $this->faker->sentence(),
            'academic_work' => $this->faker->sentence(),
        ];
    }
}

```

Figure 5.9: Database fields filled with fake data

In Eloquent, relationships are defined as methods on the Eloquent model classes. Eloquent supports many types of relationships. More about it can be read in the [official Laravel Doc](#).

The relationships are straightforward; for instance, the relationship between the project and a company is defined as “Project **belongs to** (a) company.” And vice versa, in the Company model, we have defined a method, projects(), which defines the relationship to project class. The relationship is defined as: “Company **has many** projects.”

Since relationships are influential query builders, defining relationships as methods provides powerful method-chaining and querying capabilities. [[Larndc](#)]

We have defined scopes that can help filter unwanted data when fetching data from the database; the scopes can easily be chained to the methods.

#### 5.1.3.2 Controllers

The Controllers are essential to every Laravel project since it handles user requests and returns suitable replies. This category usually contains classes that handle different HTTP requests, and each class has many methods that correlate to different actions being performed on the data. [[Larndg](#)]

We have utilized this interface to construct and maintain controller classes, which are used to process HTTP requests submitted to the server.

When a user submits a request to the server, it is directed to the appropriate controller method based on the URL and HTTP verb. The controller then processes the data, interacts with the relevant model, and delivers a response to the user as a view or JSON data.

One of the most significant advantages of using controllers in a Laravel project is that they assist in keeping an application's logic structured and manageable. Developers may guarantee that their code is modular, reusable, and easy to maintain by splitting request handling into various controller classes. Furthermore, controllers promote the separation of concerns, making testing and debugging the application's various components easier.

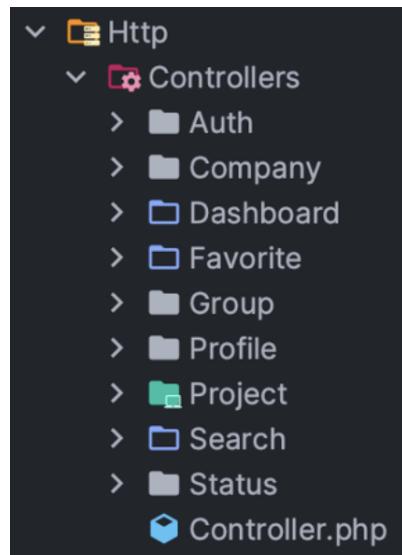


Figure 5.10: Controller's folder

#### 5.1.3.3 Middleware

Middleware is the software between two components and conducts actions depending on established rules. It bridges the client and server sides, enabling developers to specify which actions should be done before or after a request is delivered to the server. [Lar21b]

It includes classes that describe particular actions that must be completed during a request's lifetime. We have used middleware for authentication, authorization, data validation, and other activities. These responsibilities are critical for ensuring an application's security and stability.

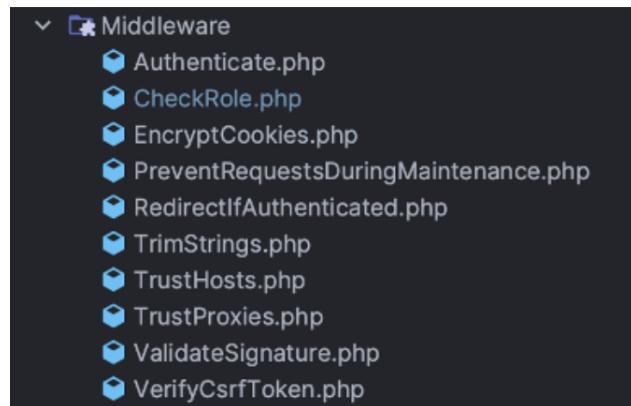


Figure 5.11: Middleware's folder

Our HTTP routes are defined in the web.php file, more in subchapter 5.1.3.6. Every HTTP request is passed through a set of predefined middleware provided by Laravel, \$middleware. An HTTP request to the web.php class is also passed through the ‘web’ middleware group, which Laravel provides.

```

3  namespace App\Http;
4
5  use Illuminate\Foundation\Http\Kernel as HttpKernel;
6
7  class Kernel extends HttpKernel
8  {
9      /**
10      * The application's global HTTP middleware stack.
11      *
12      * These middleware are run during every request to your application.
13      *
14      * @var array<int, class-string|string>
15      */
16      protected $middleware = [
17          // \App\Http\Middleware\TrustHosts::class,
18          \App\Http\Middleware\TrustProxies::class,
19          \Illuminate\Http\Middleware\HandleCors::class,
20          \App\Http\Middleware\PreventRequestsDuringMaintenance::class,
21          \Illuminate\Foundation\Http\Middleware\ValidatePostSize::class,
22          \App\Http\Middleware\TrimStrings::class,
23          \Illuminate\Foundation\Http\Middleware\ConvertEmptyStringsToNull::class,
24      ];
25
26      /**
27      * The application's route middleware groups.
28      *
29      * @var array<string, array<int, class-string|string>>
30      */
31      protected $middlewareGroups = [
32          'web' => [
33              \App\Http\Middleware\EncryptCookies::class,
34              \Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse::class,
35              \Illuminate\Session\Middleware\StartSession::class,
36              \Illuminate\View\Middleware\ShareErrorsFromSession::class,
37              \App\Http\Middleware\VerifyCsrfToken::class,
38              \Illuminate\Routing\Middleware\SubstituteBindings::class,
39          ],
40      ];
41
42 }

```

\App\Http > Kernel

Figure 5.12: Kernel class

In Laravel, it is also possible to create our middleware, but we did not need it for our project. Instead, we can also use middleware created by other developers; for example, we used the authentication middleware.

```

1 <?php
2
3 namespace App\Http\Middleware;
4
5 use App\Models\Other\Messages;
6 use Illuminate\Auth\Middleware\Authenticate as Middleware;
7 use Illuminate\Http\Request;
8
9 /**
10  * Get the path the user should be redirected to when they are not authenticated.
11 *
12 * @param Request $request
13 * @return string|null
14 */
15
16 /**
17  * @param Request $request
18 */
19 protected function redirectTo($request): ?string
20 {
21     if (! $request->expectsJson()) {
22         return route('login');
23     }
24 }

```

Figure 5.13: Authenticate.php

#### 5.1.3.4 Policies

Laravel's policies are a helpful tool for managing authorization logic in applications. Policies are PHP classes that establish authorization rules for specific resources or models. Policies enable us to regulate resource access based on user roles, permissions, and other relevant factors. [Lar21a]

When a user is not authorized to perform a particular action, the policy will respond by returning a status code 403.

These are the policies that we have created.

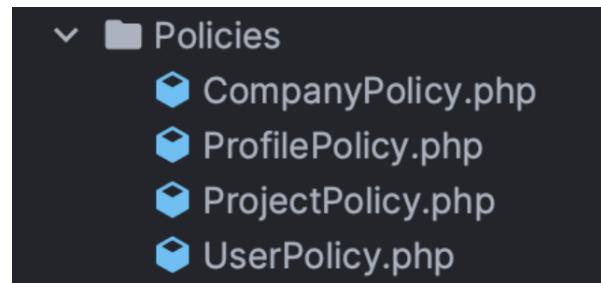


Figure 5.14: Policies's folder

```
ProjectController.php ProfilePolicy.php x
1 <?php
2
3 namespace App\Policies;
4
5 use ...
6
7 class ProfilePolicy
8 {
9     use HandlesAuthorization;
10
11     /**
12      * Determine whether the user can view the model. ...
13     */
14     public function view(User $user): bool
15     {
16         return $user->isAn(...roles: RoleType::ADMIN, RoleType::COMPANY, RoleType::STUDENT);
17     }
18
19     /**
20      * Determine whether the user can update the model. ...
21     */
22     public function update(User $user, Profile $profile): bool
23     {
24         return $user->id == $profile->user_id;
25     }
26
27     /**
28      * Determine whether the user can delete the model. ...
29     */
30     public function delete(User $user, Profile $profile)
31     {
32         return $user->id == $profile->user_id || $user->isAn(...roles: RoleType::ADMIN);
33     }
34
35     /**
36      * Determine whether the user can restore the model. ...
37     */
38     public function restore(User $user, Profile $profile){...}
39
40     /**
41      * Determine whether the user can permanently delete the model. ...
42     */
43     public function forceDelete(User $user, Profile $profile){...}
44 }
```

Figure 5.15: ProjectPolicy defines the policy for actions on the Profile model

We created the profile policy, which contains methods we utilize for authorization. To use these methods, we simply call them with → can() within the route:

```
66 |     Route::put('url: '/{profile}', 'action: 'update')->name('name: 'profiles.update')
67 |     |     ->can('ability: 'update', 'models: 'profile');
68 |
```

Figure 5.16: Put request route

#### 5.1.3.5 API Resource

The API Resource class is a middleman between the database and the API response. API Resource classes allow us to transform the Eloquent models into custom JSON format for API responses. The Resource class can be instantiated for a single model or a collection of models. [Larndh]

We have used a lot of API Resource classes, but we need to understand how the API Resource classes work. The API Resource only transforms data when directly returned from a controller method after an HTTP call. Still, it does not transform data if the API Resource is passed as a parameter to a view or other methods. But we tried to use the API Resources to transform data passed to the views, which is improper use. As a result, API Resources did not transform the data because it was not returned directly from the controller. In the future, if we use other frontend frameworks like React or Angular, we can use the API routes and return JSON responses using the API Resource.

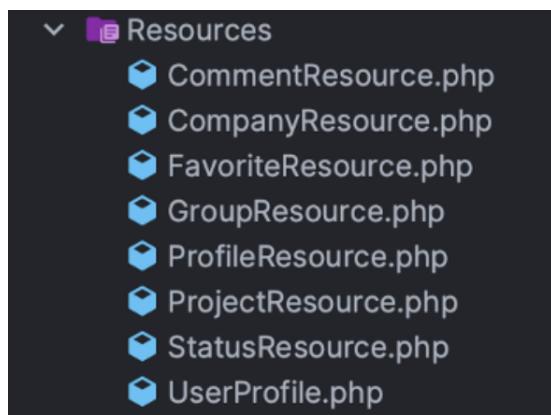


Figure 5.17: API resources' folder

### 5.1.3.6 Routes

Routes determine how the application responds to incoming requests. They are responsible for mapping requests to specific actions in the application, making it easier for developers to manage and maintain logical units of application logic. [\[Larndi\]](#)

Laravel boasts an intuitive and straightforward routing system that enables developers to define routes using a fluid and expressive syntax. These routes are defined in the routes/web.php file, the default location for web application routes. Additionally, these routes can handle various HTTP requests, including GET, POST, PUT, and DELETE.

For API routes, Laravel provides a separate file, routes/api.php, which developers can use to define routes that handle requests from external clients. These routes are typically secured using middleware like API tokens or OAuth authentication.

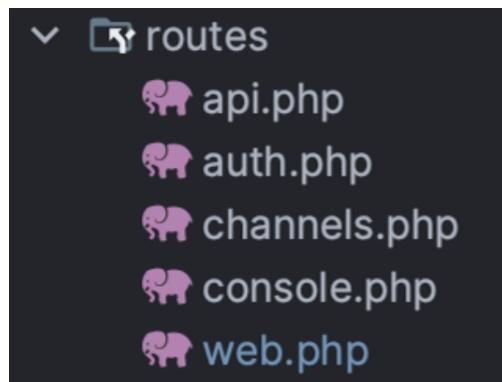


Figure 5.18: Route's folder

### 5.1.4 Sign-on

After we could not implement a federated login in our system, we decided to use another simple login method mentioned in section [4.2.1.4](#). This will still provide users with a secure and easy login process.

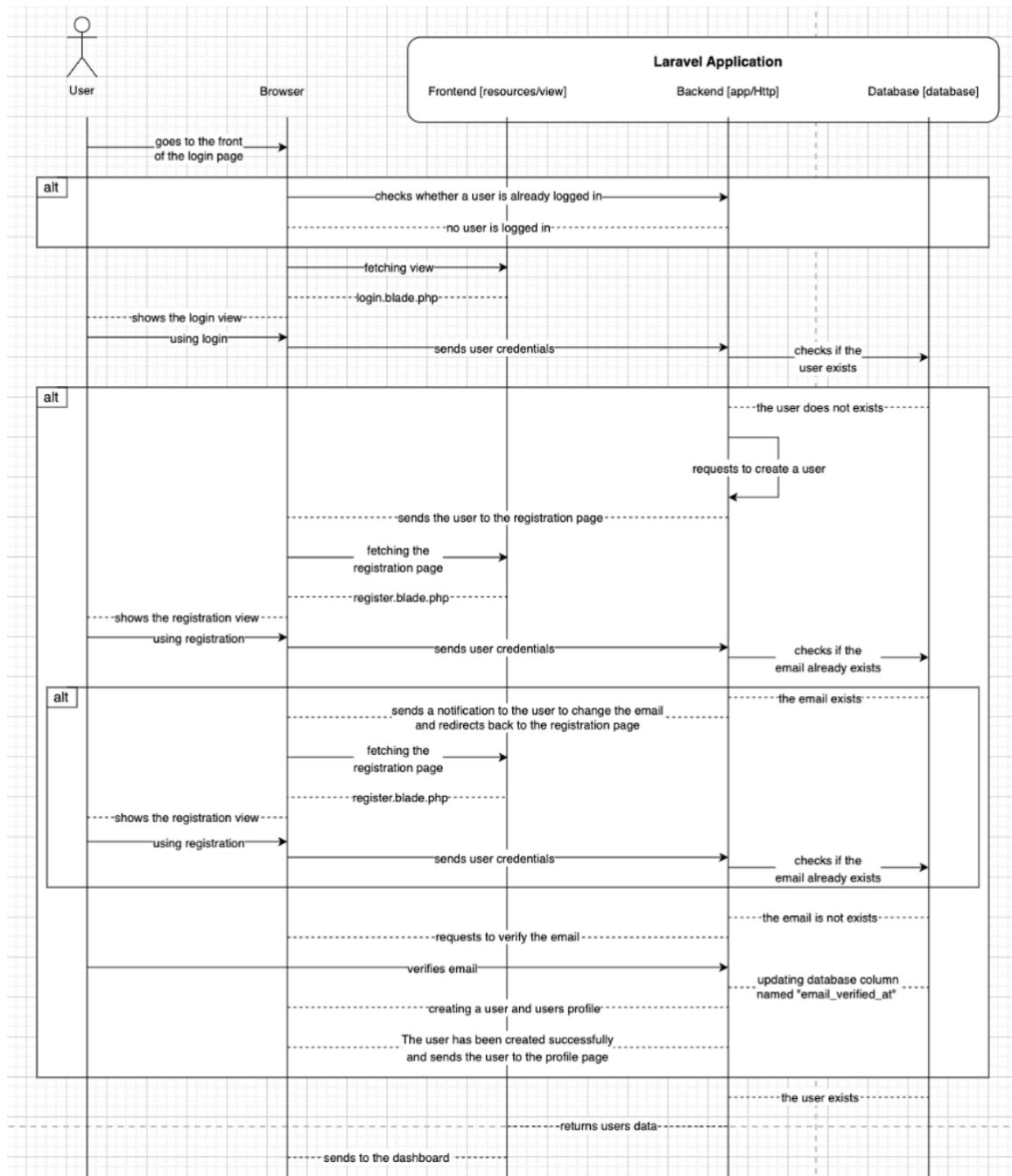


Figure 5.19: Shows the network requests during the login and register

### 5.1.5 Design and accessibility

Design and accessibility haven't been a focus, even though it's crucially important. The principle behind design and accessibility is to create products, services, or solutions accessible to everyone, regardless of their physical and cognitive abilities. This involves considering how design can affect usability, readability, and navigation for people with various disabilities. [Matnd] This includes four main categories:

**Responsive design** refers to a design approach where a website is developed to adapt to different devices and screen sizes, ensuring optimal user experience regardless of the device. [Blind] This made the application more user-friendly and compatible with multiple devices such as mobile phones, tablets, and computers.

**Universal Design** considers all users, regardless of their physical or mental abilities, to ensure that websites and apps are accessible, requiring technological solutions that work for all users. [Blind] Universal design ensures inclusively and equal access for a broad range of users, regardless of their abilities or limitations.

**User-friendliness** refers to how easy and intuitive a website or application is for its users. [Blind] Improving user-friendliness will enable the application to reduce frustration and increase user satisfaction. This can lead to increased user loyalty and positive recommendations.

**User experience** refers to a user's overall experience when using a product or service. [Blind] It will help to emphasize creating a positive user experience. It can also improve the application by forming a favorable perception among users, increasing their engagement and encouraging long-term user loyalty.

In the project's development, we used a predefined Bootstrap template with a specific design and usability. This could be an efficient way to start since we believed it could save us time in the design process. However, we did not have time to conduct user testing on the design's usability for anyone other than the course coordinator. This could be a disadvantage, as it may result in the design needing to be optimally adapted to all users, leading to a poor user experience for some.

The results from user testing of the course coordinator section of the website were positive in terms of design and accessibility. This was a good indication that the chosen design works well for users and is accessible to everyone, regardless of ability. We could not conduct continuous testing throughout the entire development process, which made it challenging to maintain usability at all times. We have learned from this that we need to focus more on the frontend when we continue developing the application or participate in future projects. Therefore, this should be carried forward into the further development phase.

## 5.2 Correspondence between requirement specification and product

The conformity between the requirement specification and the final product was concluded as good after an acceptance test conducted by the course coordinator. An acceptance test is a formal test to determine if the acceptance criteria have been met and allow the system owner to accept the system. Acceptance criteria are defined based on requirements within the scope. [Sjo16]

The course coordinator was pleased with the application submission. This can be confirmed by our recent user testing session with the course coordinator. Additionally, we observed during testing that the course coordinator, acting as a user, validated and approved the application's requirements but requested a few changes. Please refer to chapter 6.2.1.1 for more details. It is recommended for both businesses and students to test the application to ensure their specific needs are met. However, since group and subject coordinators have developed different requirement specifications for students and businesses and have tested it themselves, we can also consider these applications as having undergone acceptance testing for students and businesses. To ensure this is accurate, it should be carried forward into the next development phase after completing the entire project.

#### Functional requirements - Course Coordinator

<b>Id</b>	<b>Description</b>	<b>Priority</b>	<b>Implemented</b>
FRCC-1	Have an overview of students registered for the project courses	HIGH	YES
FRCC-2	Have an overview of Project Proposals sent in by project providers	HIGH	YES
FRCC-3	Quality-check the Project Proposals; request changes and updates to Project Proposal	HIGH	YES
FRCC-4	Edit status of Project Proposals (accepted, under revision, rejected)	HIGH	YES
FRCC-5	See overview of status of projects	HIGH	YES
FRCC-6	Create Project Descriptions based on key information in accepted Project Proposals	HIGH	YES
FRCC-7	Publish Project Descriptions for students	HIGH	YES
FRCC-8	Have an overview of which students are registered to which project(s)	MIDDLE	PARTLY
FRCC-9	See the status of students in the sign-up phase (i.e., confirmed member of a project, under consideration, not signed up for a project)	MIDDLE	YES
FRCC-10	Edit project groups	LOW	NO
FRCC-11	Register submission of formal documents (Collaboration Agreement and NDA for each)	LOW	NO
FRCC-12	Archive Project Proposals and Descriptions	MIDDLE	PARTLY
FRCC-13	Remove/Archive company profiles	MIDDLE	PARTLY

Figure 5.20: Implementation status of Course coordinator's functional requirements

**Functional requirements - Student**

<b>Id</b>	<b>Description</b>	<b>Priority</b>	<b>Implemented</b>
FRS-1	Create and edit profile of themselves	MIDDLE	YES
FRS-2	See all Project Descriptions	HIGH	YES
FRS-3	Filter projects according to availability	HIGH	YES
FRS-4	Can apply for the published projects	HIGH	YES
FRS-5	Filter relevant projects based on their interests	LOW	YES
FRS-6	Register their interest for a project (individually or as a pre-defined group)	HIGH	YES
FRS-7	See who is in the final group	LOW	NO
FRS-8	Can contact the other groups members	MIDDLE	PARTLY

Figure 5.21: Implementation status of Student's functional requirements

**Functional requirements – Project Provider**

<b>Id</b>	<b>Description</b>	<b>Priority</b>	<b>Implemented</b>
FRPP-1	Create and edit profile of themselves	MIDDLE	YES
FRPP-2	Create project proposals	HIGH	YES
FRPP-3	Edit project proposals based on feedback from the Course Coordinator	HIGH	YES
FRPP-4	Access student profiles	HIGH	YES
FRPP-5	Filter student profiles-based on their interests	LOW	NO
FRPP-6	Have an overview of students that have registered interest in their project(s)	MIDDLE	YES
FRPP-7	Can accept/reject students	LOW	YES
FRPP-8	See who is in the final group	LOW	NO
FRPP-9	Have the name and contact details of the Internal Supervisor attached to the project	LOW	NO

Figure 5.22: Implementation status of Project Provider's functional requirements

### **Non-Functional requirements**

<b>Id</b>	<b>Description</b>	<b>Priority</b>	<b>Implemented</b>
NFRS-1	The system is protected from unauthorized access to the system and its stored data.	HIGH	YES
NFRS-2	The system considers different levels of authorization and authentication across different user roles.	HIGH	YES
NFRS-3	Meet and follow GDPR regulation	HIGH	PARTLY
NFRS-4	Follow OsloMet's guidelines and requirements for data security and privacy	MIDDLE	YES
NFRS-5	The system shall provide a web page that explains how to navigate the site. This page should be customized based on what pages that user is allowed to access.	MIDDEL	YES
NFRS-6	This help page should be accessible from all other pages.	MIDDEL	YES
NFRS-7	Adheres to the (seven) principles for Universal Design	LOW	PARTLY
NFRS-8	Has a consistent "look and feel"	LOW	YES
NFRS-9	Translation into another language (Norwegian)	LOW	NO
NFRS-10	Scalability: the system should be able to accommodate additional course types and an increase in the number of students and projects without major reengineering.	MIDDEL	YES

Figure 5.23: Non-functional requirements

## 5.3 Central data structures in the solution

In this section, we will introduce the fundamental data structures crucial to both the frontend and backend of the solution.

### 5.3.1 Principles for code development

We have applied to the best of our abilities the principles of code development while creating our web application. Our goal was to build a maintainable application that can be used and improved by other developers in the future.

#### 5.3.1.1 Modularity

Modularity is a software development principle that involves breaking down a codebase into independent and self-contained modules or components that can be developed, tested, and maintained separately. This approach improves the quality of the code, makes it easier to maintain, and enables developers to work on different parts of the codebase concurrently. [\[WG12\]](#)

Modularity was a primary focus during the development of our application. By incorporating Git branching and embracing a modular approach, we were able to concurrently collaborate on various parts of the codebase without interfering with each other's work.

#### 5.3.1.2 Object-Oriented Programming (OOP)

OOP is a software development approach that uses reusable and modular units called "objects" to represent real-world entities. Objects encapsulate both data and behavior. Using OOP results in cleaner and more maintainable code. We have many model objects that represent real-world entities such as Project and User model. These models are also used to talk with the database through Laravel Eloquent model binding. Using models made the development process much easier.

### **5.3.1.3 Separation of Concerns**

Separation of concerns means breaking code into parts, each responsible for a specific task. For example, separating UI from business logic. Following this concept made it easier to modify and maintain the software without affecting other parts. Also it enabled us to code concurrently, where one group could focus on creating the best UI/UX, while the other worked on backend and providing data for the frontend. We have reflected over what could have been done differently to better achieve “Separation of concerns” in chapter [4.3.4](#).

### **5.3.1.4 Don’t Repeat Yourself (DRY)**

The concept of DRY means avoiding unnecessary repetition or duplication of code or logic. DRY encourages developers to develop reusable and modular code. This is a fundamental knowledge among all developers, but following it was not always easy for us because many times we did not know where to put a certain method or logic to make reusable in the correct way.

## **5.4 Database**

### **5.4.1 Factories**

We have used Laravel Faker, which is a useful tool for easily creating realistic and diverse datasets. It offers various types of fake data and simplifies the process of generating test data or populating databases. By using Laravel Faker, you can improve the quality of test data, simulate real-world scenarios, and save time and effort with just a few lines of code. We did this to test our database model and verify that it functions as intended.

The ‘Factory’ class in Laravel allows developers to create model instances and modify their attributes. [Larndb] For example, in our ‘ProjectFactory’, we generates fake data for the ‘Project’ model, as shown in figure [5.24](#).

```

/**
 * @extends \Illuminate\Database\Eloquent\Factories\Factory<\App\Models\Project>
 */

class ProjectFactory extends Factory
{
    /**
     * Define the model's default state. ...
     */

    public function definition()
    {
        return [
            'title' => $this->faker->sentence(),
            'description' => $this->faker->paragraph( nbSentences: 2), // n of sentences
            'background' => $this->faker->paragraph( nbSentences: 2), // n of sentences
            'aims' => $this->faker->paragraph( nbSentences: 2), // n of sentences
            'tags' => 'laravel, api, backend',
            'final_deliverable' => $this->faker->paragraph( nbSentences: 2), // n of sentences
            'skills' => $this->faker->paragraph( nbSentences: 2), // n of sentences
            'number_of_students' => $this->faker->numberBetween( int1: 1, int2: 5),
            'additional_information'=> $this->faker->paragraph( nbSentences: 2), // n of sentences
            'agreement' => $this->faker->sentence(),
            'academic_work' => $this->faker->sentence(),
        ];
    }
}

```

Figure 5.24: Project factory from SIM application

#### 5.4.2 Migrations

We used Laravel’s migration to build and manage the database in development on our application. Migrations simplified the process of defining and modifying the database schema, including creating tables and defining columns as we can see in figure 5.25. In addition, they functioned as a version control system for the database, which could enable easy tracking and use of changes. Laravel migrations have made database setup and maintenance in Laravel much easier and more efficient.

```

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up()
    {
        Schema::create('users', function (Blueprint $table) {
            $table->id();
            $table->string('email')->unique();
            $table->timestamp('email_verified_at')->nullable();
            $table->string('password');
            $table->rememberToken();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down()
    {
        Schema::dropIfExists('users');
    }
};

```

Figure 5.25: Showing the migration code for the user's model

### 5.4.3 Seeders

Seeders serve as useful helpers to incorporate the initial dataset into a database. They are particularly valuable during testing or at the beginning of a new project. Utilizing seeders can be a time-saving strategy, especially when dealing with extensive databases that require significant data entry.

We used seeder classes to insert the fake data into their respective and corresponding tables. This aided us in both testing and verifying the adequacy of the database structure.

## 5.5 Security

### 5.5.1 Secure Development and safety measures

Information Security is one of the most critical parts of IT. The lack of knowledge in the security field can become critical for each company that is taking advantage of a website. [Moo16] Therefore, security will be in the focus of the application development throughout the entire process. To strengthen the security of our system/application, we first need to find potential threats and weaknesses. The weaknesses may appear insignificant initially, yet failing to take action could result in losing everything. Detecting these weaknesses and providing secure solutions is therefore essential for our application.

We are using the programming language PHP together with the framework Laravel. PHP alone does not contribute to any security for the user. The framework Laravel, however, provides many built-in safety features one can receive numerous benefits from. Therefore Laravel is a good option because it includes many safety features. Some safety features come "out of the box" and are already used. Examples of the features available in Laravel are Password Hashing, Authentication, SQL Injection protection, XSS protection (Cross-Site Scripting), and more. We have taken advantage of all the safety features mentioned using PHP Laravel. Password Hashing and Authentication were done manually, Injection protection was done both manually and automatically (Input validation was done manually, while SQL injection protection is automatically configured by Laravel). XSS protection was also done automatically by Laravel.

### 5.5.2 Risk Assessment and Threat Modeling

Risk assessment searches for potential threats and risks in a system is very typical amongst businesses to have a risk assessment performed by security analysts. Threat modeling is a common practice to achieve this. We developed a threat model using STRIDE and DREAD methodology to perform a risk analysis. These methodologies make the assessment more detailed and precise because they can list what safety measures and security mitigation's to prioritize. In this risk assessment, the score numbers will range from 1-10. There is no "correct" range; some use 1-3 or 1-5. Due to its

contribution to a more detailed analysis, we selected 1-10.



Figure 5.26: Example of Risk Matrix. Retrieved from [[smand](#)]

On the x-axis, we have severity; on the y-axis, we have probability. This matrix will be used as inspiration when creating our own personalized risk matrix to rank the risks and threats. In order to create a risk matrix, we one must dive deeper into the STRIDE and DREAD methodologies to create a risk matrix.

Type	Description	Security Control
Spoofing	Threat action aimed at accessing and use of another user's credentials, such as username/password	Authentication
Tampering	Threat action intending to maliciously change or modify persistent data, such as records in a database, and the alteration of data in transit between two computers over an open network, such as the internet.	Integrity
Repudiation	Threat action aimed at performing prohibited operations in a system that lacks the ability to trace operations.	Non-Repudiation
Information Disclosure	Threat action intending to read a file that one was not granted access to, or to read data in transit. Retrieving information from an ML model through e-g model inversion.	Confidentiality
Denial of service	Threat action attempting to deny access to valid users, such as by making a web server temporarily unavailable or unusable.	Availability
Elevation of Privilege	Threat action intending to gain privileged access to resources in order to gain unauthorized access to information or to compromise a system.	Authorization

Figure 5.27: Definition of Stride. Retrieved from [\[Larnd\]](#)

Figure 5.27 above explains what each of the letters in the word STRIDE stands for, along with the information security pillar it belongs to and the definition of the threat.

Threat	D	R	E	A	D	Total	Average	Rating	Fuzzy Risk value	Risk level	Fuzzy Risk level
Blind SQL Injection	9	6	8	9	6	38	7.60	High	39	High (6 <sup>th</sup> category)	
Login Page SQL Injection	9	6	8	9	6	38	7.60	High	39	High (6 <sup>th</sup> category)	
Unencrypted login request	6	4	6	5	5	26	5.2	Medium	32.5	Somewhat High (5 <sup>th</sup> category)	
Application Error	2	1	3	2	3	11	2.2	Low	19	Somewhat Low (3 <sup>rd</sup> category)	
Inadequate account lockout	2	1	3	2	3	11	2.2	Low	19	Somewhat Low (3 <sup>rd</sup> category)	
Permanent cookie contains sensitive session information	2	1	3	2	3	11	2.2	Low	19	Somewhat Low (3 <sup>rd</sup> category)	
Session information not updated	2	1	3	2	3	11	2.2	Low	19	Somewhat Low (3 <sup>rd</sup> category)	
Unencrypted password Parameter	2	1	3	2	3	11	2.2	Low	19	Somewhat Low (3 <sup>rd</sup> category)	
Unencrypted viewstate Parameter	2	1	3	2	3	11	2.2	Low	19	Somewhat Low (3 <sup>rd</sup> category)	

Figure 5.28: DREAD in Risk Matrix. Retrieved from: [\[SSB13\]](#)

Figure 5.28 is an example of a Risk Matrix using a DREAD methodology. Instead of STRIDE on the Y-axis, this matrix includes specific threats. Creating the risk matrix relies on the nature of the development process and the tools utilized. In our case, we are using the highly popular STRIDE. STRIDE was created by two engineers at Microsoft. [\[Don21\]](#) We decided to use STRIDE because one of our group members personally has used STRIDE to perform a risk assessment for the company Finterai. This was very efficient and made the risk assessment both comprehensive and detailed.

We need to create a risk matrix to rank the different threats the platform could face. The risk matrix combines the consequences alongside the X-axis and the probability of occurrence along the Y-axis. Nevertheless, we used STRIDE and DREAD to create our first version of the risk matrix to provide a more detailed risk assessment for the web application. DREAD will be along the X-axis, and STRIDE will be along the Y-axis.

The ranking we will use for the Risk Matrix will be as follows:

Risk Matrix Ranking	
Low	1-10
Medium	11-24
High	25-39
Critical	40-50

Figure 5.29: Risk Matrix Ranking

RISK MATRIX	D	R	E	A	D	Total	Rating
Spoofing threat	4	7	2	4	1	18	Medium
Tampering threat	9	4	9	10	8	40	Critical
Repudiation threat	6	7	2	2	5	22	Medium
Information Disclosure threat	4	3	7	10	6	30	High
Denial of Service threat	8	3	6	9	5	32	High
Elevation of Privilege threat	8	5	6	8	4	31	High

Figure 5.30: Risk matrix with STRIDE and DREAD (first edition)

After further discussion, we concluded that extending the matrix was necessary. Now that we have the total rating of each type of threat, we can further develop the matrix. To do this extension, we can use the total rating from our first version of the risk matrix as the Y-axis and probability as X-axis. We decided to do this extension because the matrix did not consider probability. The tampering threat is an excellent example of inaccuracy because of the lack of probability. It was the only threat that resulted in a critical rating. It is improbable for a hacker to tamper with the data, making the situation unrealistic. After all, it would have zero benefits for the hacker performing the

attack. So even though a potential tampering attack would be harmful, the chance of it happening is minimal.

To make the extended version of the risk matrix, we needed to round off the ratings from the risk matrix and divide it by 10. Because we want to match the rating scales and consider probability, we will use a scale of 1-4, consisting of low, medium, high, and very high, to create a risk matrix. The final score is Rating + Probability divided by two.

	A	B	C	D
1	EXTENDED RISK MATRIX	Rating	Probability	Total
2	Spoofing threat	2	1	1.5 (low/medium)
3	Tampering threat	4	1	2.5 (medium/high)
4	Repudiation threat	2	2	2 (medium)
5	Information Disclosure threat	3	2	2.5 (medium/high)
6	Denial of Service threat	3	1	2 (medium)
7	Elevation of Privilege threat	3	2	2.5 (medium/high)

Figure 5.31: The extended risk matrix (final version)

Now that we have a complete Risk Matrix with the final ratings, we can analyze what threats and vulnerabilities to prioritize and create solutions for throughout this project. As seen in the Extended Risk Matrix, “Tampering”, “Information Disclosure” and “Elevation of Privilege” were the highest-scoring threats. We are mainly focusing on “Information Disclosure” and “Elevation of privilege” because it is very realistic that this could happen. The chance of a tampering attack is improbable since the hacker will not benefit. Let’s have a further look into OWASP before implementing solutions.

### 5.5.3 OWASP

The Open Web Application Security Project (OWASP) consists of 32,000 volunteers from all across the globe who perform security assessments and research to help website owners and security experts protect web applications from cyber attacks. [Par22] It is also essential to note that OWASP is considered a non-profit organization. We have used OWASP's Top Ten, a standard awareness document comprising web applications' top ten most critical security risks. This document is updated every three or four years.

As of 2021, the [OWASP Top 10](#) includes the following web application security risks:



Figure 5.32: This is the OWASP Top Ten 2021. Retrieved from [\[OWAnd\]](#)

Figure 5.32 gives us information about the top ten security threats to focus on during the application development. Laravel protects us from the third and tenth security risks, Injection and XSS, respectively. It is also possible to manually use other in-built safety features in Laravel to further protect the web application. Although, one has to consider that the Top Ten from OWASP is more general, while our Risk Matrix is more specific toward our application. The OWASP Top Ten and our extended Risk Matrix combined give us a great indication of what to focus on. OWASP A01 correlates to “Elevation of Privilege” from the Risk Matrix, and OWASP A02 correlates to “Information Disclosure” from the Risk Matrix. Therefore providing secure solutions for these two fields

is necessary.

#### 5.5.4 Security solutions for the application

##### *Protection from “OWASP A01 - Broken Access Control”*

One solution that can protect broken access control is to deny all access by default and adequately specify the degree of access in the code. We have used Laravel Breeze to create a fully working login and registration page with authentication control. The authentication protocol is made using Middleware. See [4.2.1.4 - Single Sign On](#) and [5.1.3.3 - Middleware](#) for more about the authentication process. Our authentication protocol is secure and protects against OWASP A01.

##### *Protection from “OWASP A02 - Cryptographic Failures”*

In our application, we have used Bcrypt, a password-hashing algorithm. Niels Provos and David Mazières developed this algorithm based on the Blowfish Cipher, a symmetric-key block cipher requiring only a single key to encrypt and decrypt data. How it is perceived is considerably faster than asymmetric encryption. [Tecnd] Blowfish Cipher also has no practical cryptanalysis to this date, meaning that no one has been able to crack the Blowfish Cipher. So because of the speed and reliability of the Blowfish Cipher, we decided to use Bcrypt to encrypt the sensitive data that the users of our application will provide. We are protected from “Cryptographic Failures,” the second-ranked security risk from OWASP’s Top Ten 2021. Although, it is essential to note that our collected sensitive data gets vulnerable if the Blowfish Cipher algorithm is cracked. In this case, we need to change the password-hashing algorithm as fast as possible to avoid potential data leakage.

##### *Validation for input fields and checkbox*

We have implemented input validation and checkbox validation to secure our application. For example, we display error messages when a user tries to register with an existing email or a password shorter than eight characters, see figure [5.33](#) This validation also provides protection against injection attacks, addressing OWASP A03. Additionally, we use checkbox validation for agreeing to our Terms & Conditions during registration. This way, the user needs to agree before creating an account, see figure [5.34](#)

The registration page displays two validation errors:

- Email:** The input "john.doe@example.com" is highlighted in red, and the error message "The email has already been taken." is shown below it.
- Password:** The input field is redacted with three dots, and the error message "The password must be at least 8 characters." is shown below it.

Figure 5.33: Input validation in the registration page

The registration page shows a validation error for the checkbox field:

- A blue-bordered checkbox labeled "I accept the Terms and Conditions" is shown.
- A purple callout box with white text states: "Please check this box if you want to proceed."

Figure 5.34: Checkbox validation in the registration page



Figure 5.35: Terms & Conditions link

### 5.5.5 Terms & Conditions

Creating Terms & Conditions can be difficult, especially when no members have a specialization or deep knowledge of the law. Usually, lawyers and others with political knowledge create the Terms & Conditions. Therefore, our Terms & Conditions are less advanced than, e.g., Google's. Nevertheless, we have included the essential parts to inform the users of the web application and protect us from

potential exploitation and abuse of the web application. In order for a user to create an account, they have to agree with our Terms & Conditions, which they can read in the registration page (see Figure 10).

Essential things to include in the Terms & Conditions are:

- User Guidelines
- Safety and protection of data
- Termination clause
- Disclaimer for third-party applications/services
- Agree button (checkbox)

See appendix [A.6](#): User Guidelines

### 5.5.6 Secure development in practice

Creating an unreachable system or application with zero vulnerabilities is impossible. Therefore, it is essential to maintain the application's security by constantly updating the security protocols, adding new safety features and mitigating potential attacks. We have taken the web application's security very seriously by analyzing and implementing secure solutions for our application's most critical and relevant security risks. Although we have put in a lot of effort to strengthen and maintain the security of our platform, no system or application is 100% secure. No company can be secure because humans write most code. Humans make mistakes and simply can't write perfect code.

### 5.5.7 Reflections on login and authentication method

We were initially going to use Feide as our login method. We wanted to use Feide because it would make the registration process much more efficient. Their effectiveness is because the students would

not need to create a new user but instead log in with their already created Feide account. This method also ensures that the students are honest. In addition to this, Feide is also very secure because they are taking advantage of MFA, which stands for . MFA means that you can provide more than one form of authentication. Today, there are several ways to bypass login methods that only use one-factor authentication, such as performing a brute-force attack. Therefore, we needed another alternative to protect us from attacks like this.

Even though we wanted to use Feide, we found out that to use this login method; we had to be registered as an official organization to request Feide as a login and authentication method. In addition to this, we had to create another form of login method as well if we wanted to use Feide, because the external companies do not have Feide. Therefore, we tried a second alternative, which was AzureAD's authentication method. This login method lets the users choose an additional form of authentication. This provides considerably more user security and protects them from brute-force attacks. Sadly, we did not manage to set up AzureAD, as it was a very time-consuming and complex process. Therefore, we decided to pause the AzureAD authentication implementation temporarily and instead focus on the main functionalities of the application.

The third alternative was to use Laravel Breeze, a pre-made login implementation. We chose this alternative because of its effectiveness. The purpose of Breeze is to provide a fast and straightforward login page. Something we should have taken into consideration when choosing Breeze was the need for more security features. We later discovered that Laravel had another template called Laravel Jetstream, which had built-in security features like e.g. MFA. In other words, Breeze is like a lite version to provide effectiveness, while Jetstream is a complete version with more features. Getting the same safety features for Laravel Breeze is possible but very time-consuming because they are already set up and ready to use in Laravel Jetstream.

## 5.6 Relationship to machines/databases/OS

In this subsection, we present the functional interfaces of the system. We have chosen to use API connections as a crucial link between the frontend and backend. This application runs on containers, making the operating system less critical for its execution. We also have expectations that the system

will function seamlessly on all types of operating systems and both older and newer web browsers.

### 5.6.1 API documentation

The API documentation provides valuable information to clients and developers about the available endpoints and how to interact with them. Clients can access and use the server's resources by following the instructions provided in the API documentation. Developers can also extend or modify the server's functionality by referring to the documentation.

For a REST API, the documentation typically includes an overview of the available endpoints and the expected input and output formats for each endpoint. A tool like Swagger UI often presents this information through an interactive interface, making exploring the API documentation and testing API requests easier.

Figure 5.36 is a visual representation of the available endpoints in the REST API, which can be a helpful reference for both clients and developers to understand the structure and organization of the API.

**Auth** Auth part of application.

<b>POST</b>	/login	Login user	✓
<b>GET</b>	/logout	Logout user	✓ 🔒
<b>POST</b>	/reset-password/{token}	Chang password by email	✓ 🔒
<b>POST</b>	/forgot-password	Reset password by email	✓ 🔒
<b>POST</b>	/register	Create a new user	✓ 🔒
<b>GET</b>	verify-email/{id}/{hash}	Verify User Email	✓ 🔒
<b>GET</b>	/profiles/change/password	Change users password	✓ 🔒
<b>POST</b>	/profiles/update/password	Update user password	✓ 🔒

**Profile** Profile part of application.

<b>GET</b>	/profiles	Get a list of users with profiles	✓ 🔒
<b>GET</b>	/profiles/admin	Get a list of admins	✓ 🔒
<b>GET</b>	/profiles/company	Get a list of companies	✓ 🔒
<b>GET</b>	/profiles/student	Get a list of students	✓ 🔒
<b>GET</b>	/profiles/{profile}	Show a profile	✓ 🔒
<b>PUT</b>	/profiles/{profile}	Update a profile	✓ 🔒
<b>GET</b>	/profiles/{profile}/edit	Edit a profile	✓ 🔒
<b>DELETE</b>	/profiles/{profile}/destroy	Delete a company profile	✓ 🔒

**Search** Search part of application.

<b>POST</b>	/search	Search for projects	✓ 🔒
-------------	---------	---------------------	-----

**Schemas**

- change\_request >
- forgotten\_request >
- login\_request >
- register\_user >
- create\_request >
- company >
- favorite >
- profile >
- project >
- update\_request >
- create\_project\_request >

**Dashboard** Dashboard part of application.

<b>GET</b>	/index	Retrieve dashboard data	✓ 🔒
------------	--------	-------------------------	-----

**Company** Company part of application.

<b>GET</b>	/company/create	Create a new company	✓ 🔒
<b>POST</b>	/company/store	Create a new company	✓ 🔒

**Project** Projects part of application.

<b>GET</b>	/project	Get projects based on user's role	✓ 🔒
<b>GET</b>	/project/show/{id}	Get project details	✓ 🔒
<b>GET</b>	/project/create	View Create project form	✓ 🔒
<b>POST</b>	/project/store	Add a project	✓ 🔒
<b>GET</b>	/project/edit/{id}	Edit a project	✓ 🔒
<b>PUT</b>	/project/update	Update a project	✓ 🔒
<b>DELETE</b>	/project/destroy/{id}	Delete a project	✓ 🔒
<b>POST</b>	/project/publish	Publish a project	✓ 🔒
<b>POST</b>	/request-change	Request a change for a project	✓ 🔒

Figure 5.36: Swagger API documentation

### 5.6.2 Database Integration

The database's interface and backend communicated via localhost server that required a login consisting of a username and password. MySQL was used to set up a publicly accessible database resource. The database configuration file is shown in the figure below, with the username and password.

```
12 DB_CONNECTION=mysql
13 DB_HOST=mysql
14 DB_PORT=3306
15 DB_DATABASE=pmp
16 DB_USERNAME=sail
17 DB_PASSWORD=password
18
```

Figure 5.37: MySQL database configuration file

### 5.6.3 Main parts of the program

#### **Company - Adds project proposal**

For companies who wish to post projects, the application can offer a simple and user-friendly interface. The company can create an account and then post a project description with the necessary details required to attract students. This may include details about the project's duration, requirements for the participating students, and expected prior experience and competencies.

#### **Course Coordinator - Quality assurance of the project proposals**

The course coordinator's task is to ensure the quality of project proposals by carefully reviewing each proposal. They must verify that the proposals are relevant to the subject and within the students' area of expertise. By approving project proposals before they are posted on the system, the course coordinator can ensure that the projects meet a certain standard and are of high quality. This is a crucial function to ensure that students have access to quality projects and that the companies

posting projects are serious.

#### **Student - Applying for the published projects**

Students can then apply to participate in projects that match their interests and skills. They can apply through a simple interface that allows them to find and filter projects based on various criteria such as subject area. Once they have found a project they want to participate in, they can submit a request that the company can review and approve.

# **Chapter 6**

## **Testing**

### **6.1 Introduction**

We conducted tests to ensure the quality and reliability of our application, which we created using the PHPUnit framework and the Laravel built-in testing library. To accomplish this, we utilized a range of tests, such as unit tests, integration tests, and smoke tests, to uncover and correct any flaws or problems in the application.

Our testing had several specific aims, including identifying errors or bugs in our code, enhancing code quality and maintainability with automated testing, confirming that the application met user expectations and requirements and boosting confidence in the application's reliability and stability.

Using the framework PHPUnit, we run 138 test cases for all test types during the testing process. Our team of five developers and with the help of our course coordinator also performed usertesting of the application over four to five weeks. We used Notion to report and monitor the results.

## 6.2 User Testing

### 6.2.1 Testing under development phase

#### 6.2.1.1 User Tests done by course coordinator

During the development phase, we conducted user tests as an integral part of our application development process. To introduce a user test with the course coordinator, we built upon a template (see appendix A.3) from a practical IT project in the previous semester and further refined it for our bachelor thesis (see appendix A.4). The primary objectives of these tests were to prioritize and identify the essential functions of the application, aligning with the assumptions made during the design sprint and gather targeted user insights.

#### 6.2.1.2 Feedback/result

The course coordinator provided feedback on the functionality and design for all actors within the application. Overall, the coordinator expressed high satisfaction, stating that the application precisely met their preconceived expectations and desires. The project course coordinator wanted us to change the following. See Table 6.1, 6.2 and 6.4.

We organized tasks based on the priority, with 1 being the highest priority. This means we focus on completing priority 1 tasks first before moving on to priority 2 tasks.

Functionality and design for company overview	
Changes needed	Priority
Receipt or confirmation page which comes before finally submitting	1
Status options with descriptions of the different statuses when hovering or clicking, same goes for the other columns	1
Hyperlink in agreement-page	1
All projects, current projects, archived projects	1
Toolbar at bottom instead of at the top	2
Column visibility standards should be very simple, Title and Status	2
Export button, copy to excel, copy to PDF, column visibility should be removed	3
Menu icon needs to change	2
Minimum amount of data, maximum amount of information	3
Swap My Projects and Need Changes. If they have no projects published or no projects needing changes should be removed	2
Overview of deadlines on dashboard	3
All projects get archived at the end of the semester, archived projects get placed in the archives by status “ARCHIVED”	3

Table 6.1: Company requirements

Functionality and design for course coordinator	
Changes needed	Priority
Add project needs some type of connection to the supervisor who is responsible for that specific project	1
Updated Projects: split into Changes Made and Need Changes	1
Projects Overview should include: New, Under Review, Archived	1
Only need Company Name and Project Title	1
A project should include Company Name, Project Title, primary supervisor, company email	2
Can be up to three supervisors per company	3

Table 6.2: Course Coordinator's requirements

Functionality and design for Student	
Changes needed	Priority
Students status : APPLIED, NOT APPLIED, UNDER REVIEW and CONFIRMED	1
Student registration: number of students (dynamic display), status (confirmed/not confirmed) which is linked to a specific project	1
Dashboard for students: should include the status of the projects the student has applied for	2

Table 6.3: Student requirements

#### **6.2.1.3 Response to the Feedback**

The user testing we conducted with the course coordinator regarding functionality and design took place in early May. Unfortunately, we did not have enough time to implement the new requirement based on the feedback received, unless the following conditions are met.

Tasks implemented after user-test		
Tasks	Priority	Actors
Hyperlink on agreement page	1	company
Swap My Projects and Need Changes. If they have no projects published or no projects needing changes should be removed	2	company
Students status : APPLIED, NOT AP- PLIED, UNDER REVIEW and CONFIRMED.	1	student

Table 6.4: Tasks implemented after user test

#### 6.2.1.4 Student-led User Testing

Due to time constraints that prevented us from carrying out the planned user survey, we opted for an alternative approach. We conducted a test involving five participants from OsloMet focusing solely on the student role. These participants were granted access to the student roles and given the ability to create their own user accounts. A specific/predefined flow was assigned to guide their actions during the test. The user test was conducted using an interview guide, which can be found in the appendix [A.5](#).

Afterward, we asked them to navigate through the course coordinator page and test the functionality themselves. Some gave us feedback on what worked well and what could be improved on the website.

Finally, we asked them to test the entire website and give us their thoughts randomly. They gave us multiple feedback points, summarized in the table below. Some of them, also checked the security aspects, such as input validation, to ensure proper notifications. They also tested for SQL injection vulnerabilities in input fields, console logs, and web application inspection.

From this feedback, we gained insights that the users understood our search function, for example, and what was behind it. However, the users, for example, expressed dissatisfaction with the function

on the column visibility dropdown. We also noticed that some users were not entirely sure what using “all” meant for all actors. These insights were noted based on their significance on the table list, as shown in Table 6.5. These insights were then used to prioritize and improve the application’s features to meet our users’ needs better.

Strengths of the Application	Areas for Improvement in the Application
Navigation	Specify Actors: Avoid using '(All)' for actors and be more specific. Use 'Only students' instead of 'All students.'
Searching and filtering	Logo Link: Remove or add a logo
UI - user friendly and responsive, nice combination of colors	course coordinator profile view: Include profile image URL in course coordinator profile view
Layout	Password Criteria: Avoid using numbers only and set a maximum length for passwords
SQL injection was approved	Dropdown Placement: Place the dropdown under the 'Column Visibility' section
Website Status: almost complete with minor functionalities	Remove Options: Eliminate 'Approve' and 'Need Changes' options from the company's interface
Course coordinator Interface: Well-Structured and User-Friendly	Profile Image Display: Fix profile image display issue on Google Chrome
	The area code needs to be fixed
	Larger Headers: Increase header size for better readability

Table 6.5: Feedback of student test

### 6.2.2 Limitations and Challenges in User Testing

This point is essential to highlight. Collecting personal data without proper authorization is illegal in many countries and can lead to severe consequences for the data collector. Therefore, following proper procedures and obtaining permission before collecting personal data is always advisable. [Comnda]

Our initial plan was to conduct in-depth user-tests and interviews, which required the collection of personal data and information (such as key demographic data) . We applied for permission from SIKT to collect personal data (see Appendix A.7). However, as we did not get a timely response, we had to resort to our plan B and conduct the user testing anonymously without registering personally identifiable information.

As a group, we planned to distribute a digital survey to the participants for the user survey. The survey was designed to include scaling questions and yes or no answers. We also intended for each participant to receive a digital form or sheet to provide any helpful input or quotes from the tests. Unfortunately, we could not send the survey digitally as intended initially due to time limitations.

## 6.3 Technical Testing

### 6.3.1 PHPUnit and Laravel's built-in testing library

**PHPUnit** is a programmer-oriented testing framework for PHP that allows us to use PHP code to test our application. It follows the xUnit architecture for the unit testing framework. PHPUnit supports various types of testing, including unit testing, integration testing, and smoke testing. At the same time, Laravel's built-in testing library provides tools and functions to test Laravel applications. It simplifies testing by offering test case classes and assertions to verify application behavior and functionality. [Joh23]

### 6.3.2 Unit Test

*“Unit testing involves testing a single method of a class, focusing on a specific contract aspect with stubbed or mocked dependencies.”* [Fre09]

During PHP Laravel backend development, developers utilized unit tests with PHPUnit to validate the functionality of isolated functions or classes. They employed various assertions, such as assertEquals(), assertTrue(), and assertFalse(), to test code outputs and ensure accurate error messages. [Larnde] [Bernd]

We utilized the PHPUnit framework to test the functionality of the “updateProject” method, which involves retrieving a project’s object from the database. First, we prepared a “mock” project data and an expected return value. Then, we evaluated the method to ensure that it returned accurate information and triggered an error message if it failed before testing it with actual data.



The screenshot shows a code editor with a dark theme. The code is written in PHP and uses the Mockery library for testing. It includes annotations like @unit, @scenario, @case, and @expectation. The code defines a test function `company_can_update_project()` that arranges a request object, acts by calling `update` on a mocked controller, and asserts that the updated project matches the original one.

```
@unit      company projectController
@scenario  update project
@case      COMPANY is trying to update project
@expectation company is updated project successfully
@test

@okabmoss *
public function company_can_update_project()
{
    // ARRANGE
    $created_project = $this->createDataCompany();
    $request = new Request([
        'project_id' => $created_project->id,
        'status' => ProjectStatuses::NEW,
        'title' => $created_project->title,
        'description' => 'backend',
        'tags' => 'new, tags',
    ]);

    // MOCK AND ACT
    $mock_controller = Mockery::mock( ...args: ProjectController::class );
    $mock_controller->shouldReceive( ...methodNames: 'update' )->andReturn(32);
    $response = $mock_controller->update($request);
    $this->assertEquals( expected: 32, $response );
    $updatedProject = Project::find($created_project->id);

    // ASSERT
    $this->assertEquals($created_project->title, $updatedProject->title);
    $this->assertEquals($created_project->description, $updatedProject->description);
    $this->assertEquals($created_project->tags, $updatedProject->tags);
    $latestStatus = $updatedProject->latestStatus;
    $this->assertEquals( expected: ProjectStatuses::NEW, $latestStatus->status );
}
```

Figure 6.1: Example code of Laravel's built-in unit test

Using mock project data allowed us to avoid using actual data from the database in the test, which would have made it more time-consuming to set up and execute. Overall, using the framework PHPUnit and the library Laravel built-in tests for testing during development helped us catch errors early and ensure that individual code units worked as expected, see figure 6.1.

#### 6.3.2.1 Basics of Defect Tracking

We prioritize the sequence in which we address defects and indicate the severity of a bug's impact on the software's functionality. Within software development, priority pertains to the degree of significance of a defect that necessitates resolution before moving on to the next one. In contrast, severity concerns the effect of a bug on the system. Due to its increased criticality, a bug with higher

severity requires prompt resolution regardless of its priority level.

As figures 6.2 and 6.3 illustrates, we employed the notion of tracking bugs. We effectively managed and prioritized bugs according to their severity and priority levels, ensuring that we promptly addressed critical issues.

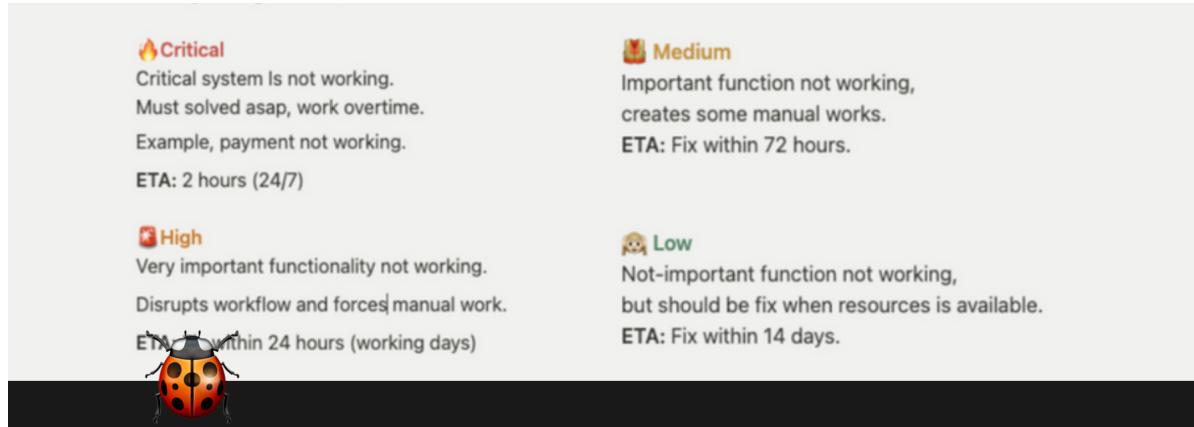


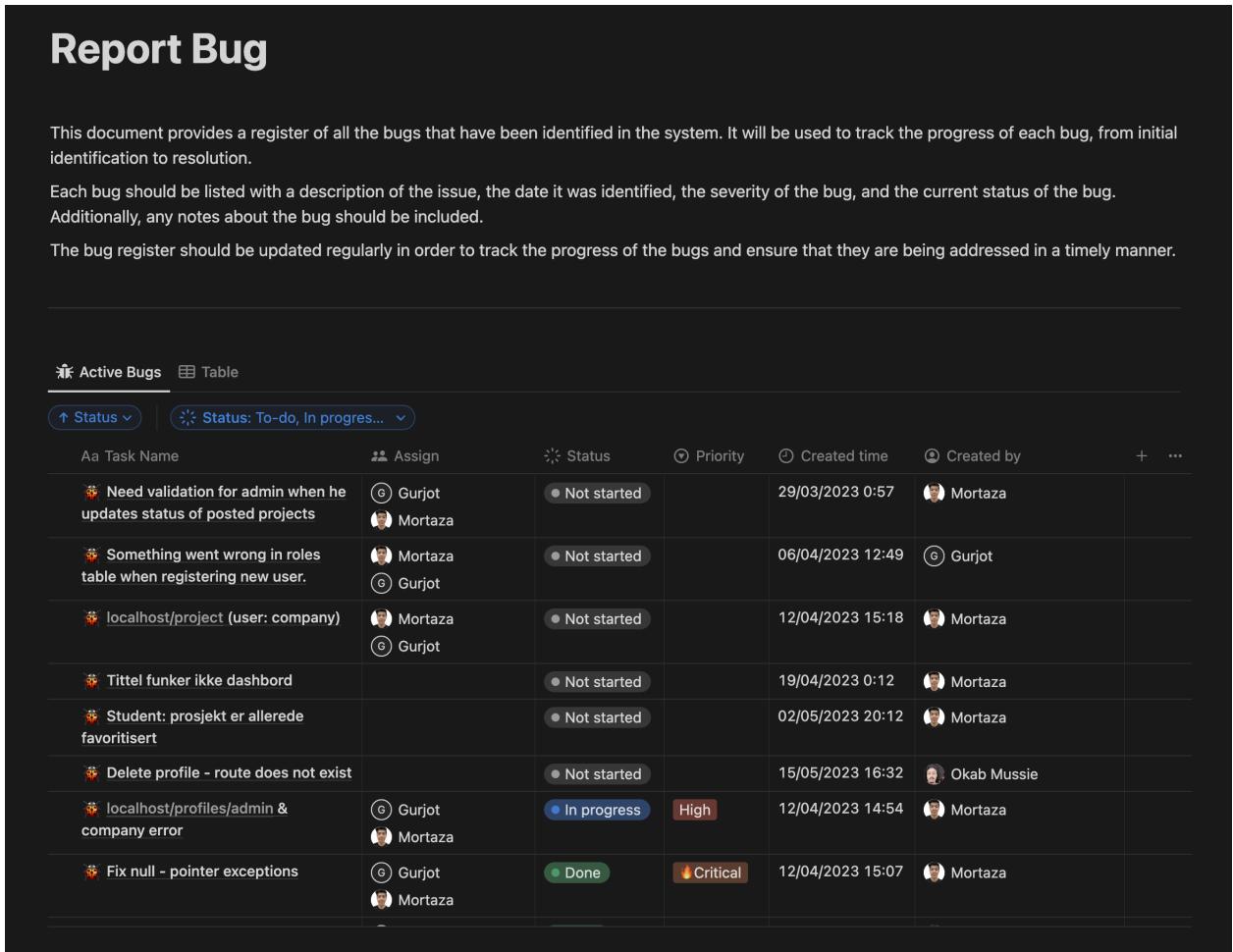
Figure 6.2: Defect tracking rules

# Report Bug

This document provides a register of all the bugs that have been identified in the system. It will be used to track the progress of each bug, from initial identification to resolution.

Each bug should be listed with a description of the issue, the date it was identified, the severity of the bug, and the current status of the bug. Additionally, any notes about the bug should be included.

The bug register should be updated regularly in order to track the progress of the bugs and ensure that they are being addressed in a timely manner.



A screenshot of a web-based defect tracking application. The interface has a dark header with the title 'Report Bug'. Below the header is a text block with instructions. The main area features a table titled 'Active Bugs' with a 'Table' icon. The table has columns for Task Name, Assign, Status, Priority, Created time, and Created by. There are dropdown menus for Status and Priority. The table lists several bugs, each with a small icon, a brief description, and user details. The last row shows a bug with a status of 'Done' and a priority of 'Critical'.

Aa Task Name	Assign	Status	Priority	Created time	Created by	...
🐞 Need validation for admin when he updates status of posted projects	⌚ Gurjot 🕒 Mortaza	● Not started		29/03/2023 0:57	🕒 Mortaza	
🐞 Something went wrong in roles table when registering new user.	🕒 Mortaza ⌚ Gurjot	● Not started		06/04/2023 12:49	⌚ Gurjot	
🐞 localhost/project (user: company)	⌚ Gurjot 🕒 Mortaza	● Not started		12/04/2023 15:18	🕒 Mortaza	
🐞 Tittel funker ikke dashboard		● Not started		19/04/2023 0:12	🕒 Mortaza	
🐞 Student: prosjekt er allerede favoritiseret		● Not started		02/05/2023 20:12	🕒 Mortaza	
🐞 Delete profile - route does not exist		● Not started		15/05/2023 16:32	⌚ Okab Mussie	
🐞 localhost/profiles/admin & company error	⌚ Gurjot 🕒 Mortaza	● In progress	High	12/04/2023 14:54	🕒 Mortaza	
🐞 Fix null - pointer exceptions	⌚ Gurjot 🕒 Mortaza	● Done	Critical	12/04/2023 15:07	🕒 Mortaza	

Figure 6.3: Defect tracking overview

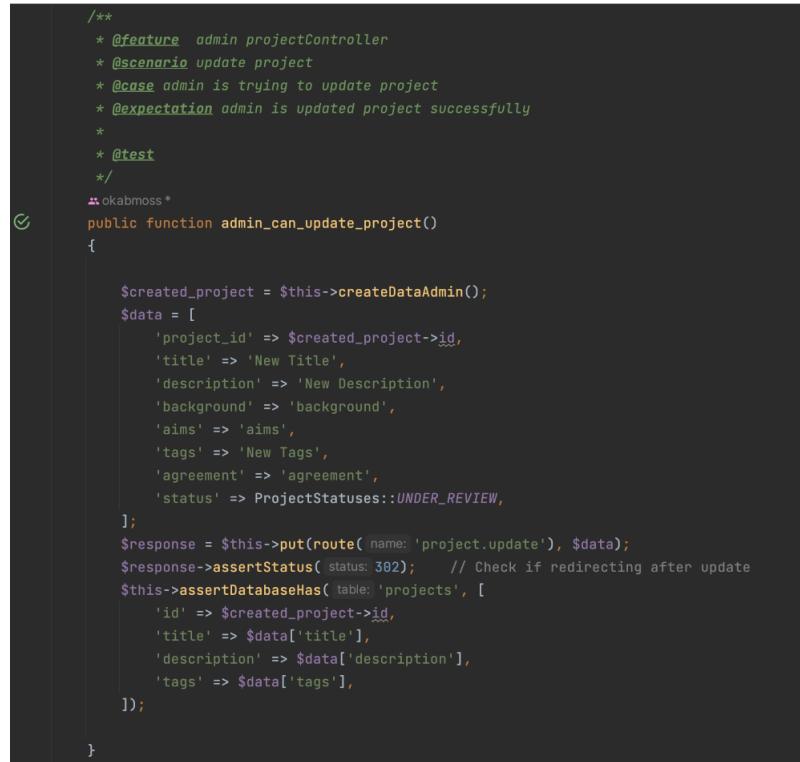
## 6.4 Feature test

*“Integration testing ensures proper interoperation between subsystems, ranging from class integration to production environment integration.”* [Fre09]

Laravel’s Eloquent ORM and PHPUnit’s integration testing framework enable developers to assess the application’s acceptance comprehensively. This includes testing database interactions and other

components. [Ind22] Laravel provides an in-memory SQLite database for evaluating application interactions without impacting the live database. [Larndd]

In Laravel, tests in the "Tests/Feature" directory verify route responses. For "updateProject," a PUT request is made to the "project.update" route. Using assertStatus(), we confirm a successful update (code 302) that redirects correctly. assertDatabaseHas() ensures the updated project data is stored in the database, as shown in figure 6.4.



The screenshot shows a code editor with a dark theme. The code is a PHP test for a 'admin\_can\_update\_project' method. It includes a multi-line docblock at the top, followed by a class definition with a single method. The method creates a new project, updates its details (title, description, background, aims, tags, agreement, status), sends a PUT request to the 'project.update' route, and then uses assertStatus(302) to check if it redirected correctly. Finally, it uses assertDatabaseHas('projects') to verify that the updated project data is stored in the database.

```
/*
 * @feature admin projectController
 * @scenario update project
 * @case admin is trying to update project
 * @expectation admin is updated project successfully
 *
 * @test
 */
@okabmoss*
public function admin_can_update_project()
{
    $created_project = $this->createDataAdmin();
    $data = [
        'project_id' => $created_project->id,
        'title' => 'New Title',
        'description' => 'New Description',
        'background' => 'background',
        'aims' => 'aims',
        'tags' => 'New Tags',
        'agreement' => 'agreement',
        'status' => ProjectStatuses::UNDER_REVIEW,
    ];
    $response = $this->put(route('project.update'), $data);
    $response->assertStatus(302); // Check if redirecting after update
    $this->assertDatabaseHas('projects', [
        'id' => $created_project->id,
        'title' => $data['title'],
        'description' => $data['description'],
        'tags' => $data['tags'],
    ]);
}
```

Figure 6.4: Integration Test - course coordinator project update scenario

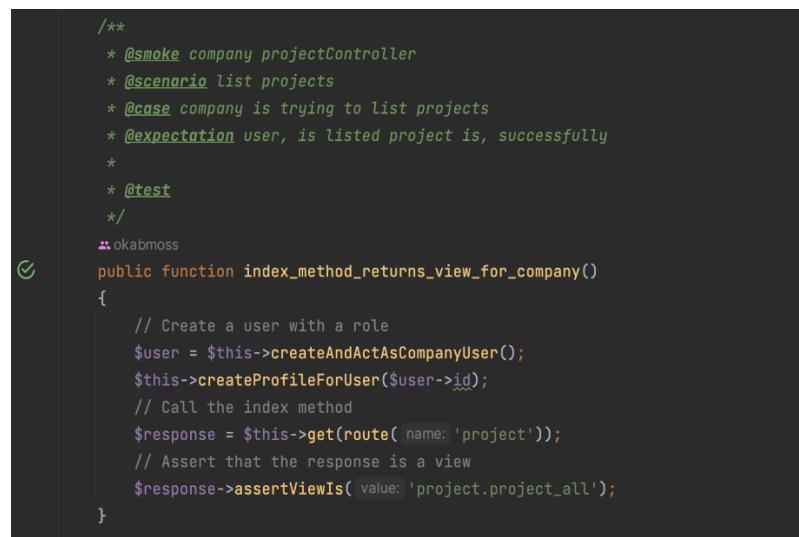
## 6.5 Smoke test

*"In the software development life cycle, developers perform smoke testing to verify code compilation and core functionality. Only after passing this initial test different levels of testing are conducted."*

[Kumnd]

We developed test cases for various situations, including high traffic and adverse conditions, to ensure the application can manage expected traffic volumes and provide a satisfactory user experience.

Figure 6.5 illustrates that the tests involve requesting the application's routes and verifying the correct responses. For instance, a "companyView" smoke test might use the GET request method to the "project" route name and then utilize the assertViewIs() method to ensure that the response value of "project.project.all" corresponds to the appropriate view page.



The screenshot shows a code editor with a dark theme. A file named 'index\_method\_returns\_view\_for\_company.php' is open. The code is a PHPUnit test case for a 'companyController'. It includes a docblock with annotations like '@smoke', '@scenario', '@case', '@expectation', and '@test'. The test itself creates a user, calls the index method, and asserts that the response is a view with the value 'project.project\_all'.

```
/*
 * @smoke company projectController
 * @scenario list projects
 * @case company is trying to list projects
 * @expectation user, is listed project is, successfully
 *
 * @test
 */
okabmoss
public function index_method_returns_view_for_company()
{
    // Create a user with a role
    $user = $this->createAndActAsCompanyUser();
    $this->createProfileForUser($user->id);
    // Call the index method
    $response = $this->get(route( name: 'project'));
    // Assert that the response is a view
    $response->assertViewIs( value: 'project.project_all');
}
```

Figure 6.5: Testing Company View Retrieval

```

/**
 * @smoke projectController
 * @scenario high traffic request to route project
 * @case 1000 users trying to access into project view at the same time
 * @expectation requests is successfully accessed
 *
 * @test void
 */
okabmoss
public function high_traffic_request_to_project_view()
{
    $numberRequests = 1000;
    $route = '/project';
    // Sends a multiple requests in a loop
    for ($i = 0; $i < $numberRequests; $i++) {
        $response = $this->get($route);
        $response->assertStatus(status: 302);
    }
}

```

Figure 6.6: Testing Project View Under High Request Traffic

As depicted in figures 6.5 and 6.6, the ongoing smoke testing aims to determine whether the application performs accurately under high usage and unfavorable circumstances. This testing helps detect potential issues such as compatibility problems and performance inadequacies.

## 6.6 Challenges in technical testing

Based on our experience with JUnit testing, it was challenging to set up xdebug (see: 3.3) and start using phpunit testing early in the development phase. This made us uncertain about the differences between smoke and unit tests, as both types could be done at the same time during early implementation. To get started, we had to spend time going through Laravel's documentation. Later, we discovered PHP traits, which could make the testing process easier. Traits are like reusable bundles of code that add functionality and reduce repetition. Our lack of knowledge about traits caused confusion with smoke and unit tests, so we had to rely on the documentation, which slowed down our progress.

We faced additional challenges in our development process, including the need to modify tests due to changing requirements, potentially slowing down our progress. Furthermore, the implementation

of the application took longer than expected, extending until mid-May. As a result, we were unable to implement the planned integration tests, which aimed to test the interaction between different components of the system.

## 6.7 Correspondence between accurate test coverage and ideal test coverage

Although there is no universally mandated minimum percentage of test coverage for software development projects in organizations, certain industries may have specific standards that dictate a certain level of test coverage.

Like many organizations, the Norwegian insurance company requires that its systems have a test coverage of at least 80%. It is worth noting, however, that this is the minimum requirement, and organizations often aim for a higher test coverage. The solution should have implemented considerably higher test coverage for the PHP Laravel application. [Heu21] [Zur21]

Code coverage is a way to measure how well our tests cover the code we write. It tells us the percentage of our code that is actually tested by automated tests. This helps us see which parts of our code are being tested and which are not. Code coverage also helps us find functions in our code that are not being used but are still counted in the coverage report. By aiming for higher code coverage, we can have more confidence in the quality and reliability of our application. Figures 6.7 and 6.8 provide visual examples of code coverage.

```

42     public function index()
43     {
44         $projects = Project::latest()->get();
45
46         if(Auth::user()->isAn(RoleType::STUDENT)){
47             $projects = ProjectResource::collection(Project::WithStatus(ProjectStatuses::PUBLISHED)->get());
48             return view('project.project_all', compact('projects'));
49         }
50
51         elseif (Auth::user()->isAn(RoleType::COMPANY))
52         {
53             $projects = $this->getUsersProjects();
54             return view('project.project_all', compact('projects'));
55         }
56
57         elseif (Auth::user()->isAn(RoleType::ADMIN))
58         {
59             $projects = Project::latest()->get();
60             return view('project.project_all', compact('projects'));
61         }
62
63         else {
64             return redirect()->back()->with(Messages::SOMETHING_WENT_WRONG);
65         }
66     }
R7

```

Figure 6.7: Code Coverage Analysis: Examining Tested and Untested Code

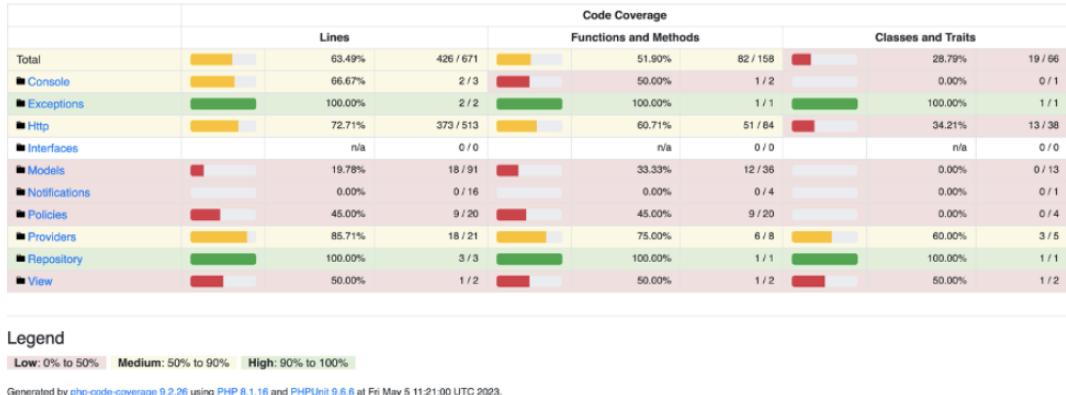


Figure 6.8: Early Stage Code Coverage Analysis for SIM Application

This report, shown in figure 6.8 , analyze our application's code coverage (SIM) using the PHPUnit vendor and offers suggestions for enhancing coverage in the future. Our test coverage is 63% for lines, functions, and methods and 28% for classes and properties. This coverage metric typically includes both our application code and the code from the vendor dependencies, which are external

components used by our application. While this is a positive starting point, there is room for improvement, particularly in covering models, views, and controllers within the HTTP context, which currently need more coverage. Additionally, it is essential to ensure that our tests include edge cases and handle potential errors effectively.

Looking at the coverage distribution by functions, methods, and lines, we can see that some areas of our code are well-tested early while others are not. Our controllers (path:http/controllers) have 72% coverage, while our models only have 19%, and the View has 50% coverage. This suggests we write more tests for everyone, especially for views and models, to ensure they behave as expected.

Based on our analysis of the coverage distribution, we have prioritized testing our controllers during the development phase to ensure steady progress. Regarding real test coverage for PHP, our backend controllers currently have a medium coverage of 81.07%. This percentage represents the line of code covered by structured unit testing. For a visual representation of the test coverage, refer to figure 6.9.

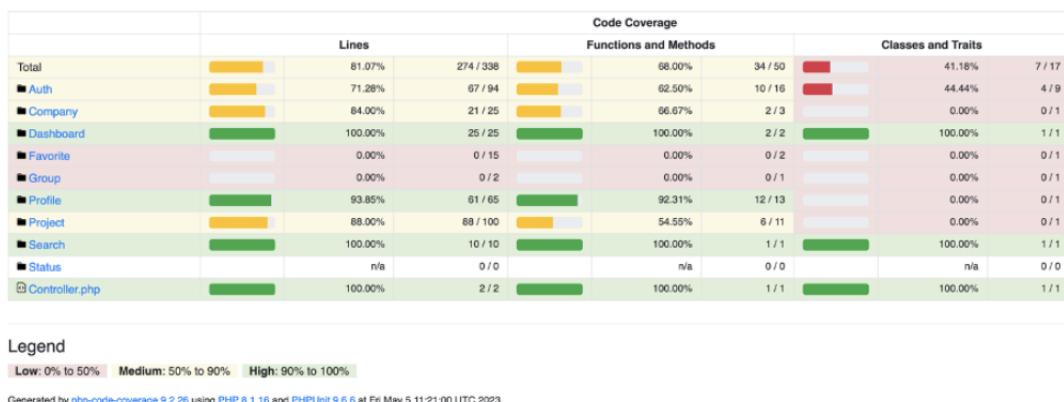


Figure 6.9: Code Coverage Analysis for SIM Controllers

# Chapter 7

## User Guide

### 7.1 Registration and Login

#### 7.1.1 Common Login Page For All Actors

The login page is the entry point to the Student Internship Matchmaking (SIM) web application. It allows users to log in, reset their password, or create a new account. All three actors have the same login process, while the registration process is only common for the project providers and students. In other words, it is not possible to create a user with the role “course coordinator” for security reasons. Figure 7.1 illustrates the login page layout.

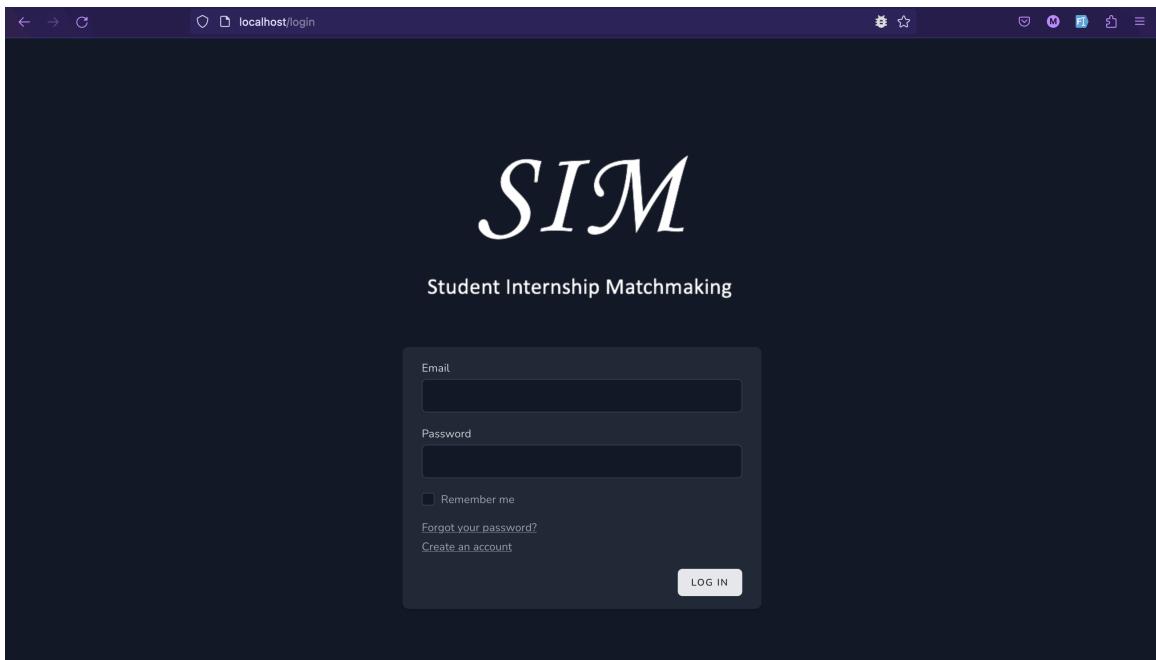


Figure 7.1: The login page

### 7.1.2 Common Password Reset

If users forget their password, they can initiate a password reset process by clicking “Forgot your password?” on the landing page. The system will prompt them to provide their email address to receive an email for a password reset. Figure 7.2 shows the password reset page.

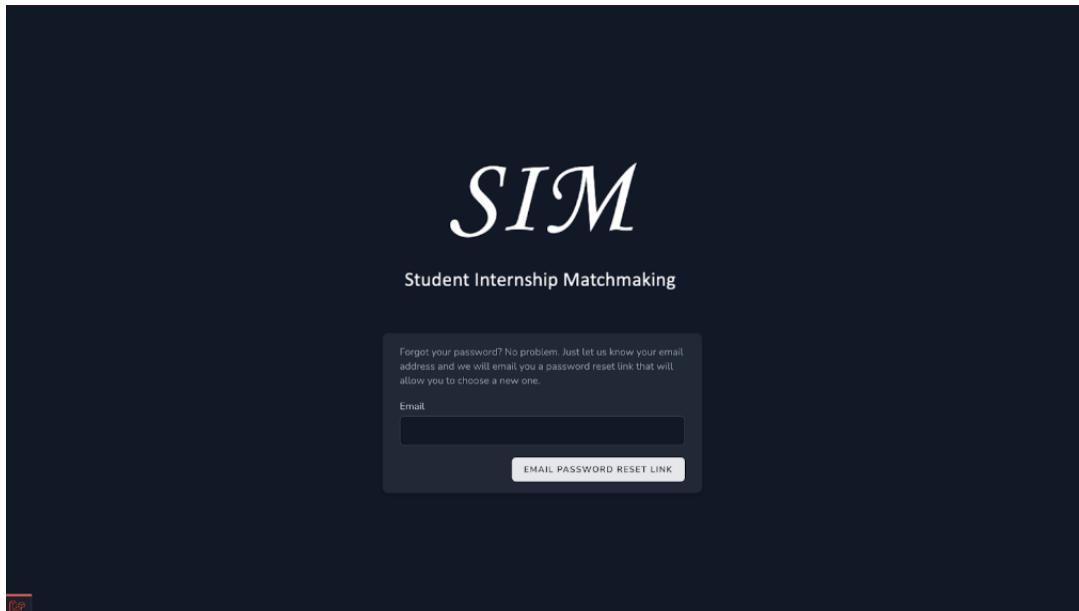


Figure 7.2: The password reset page

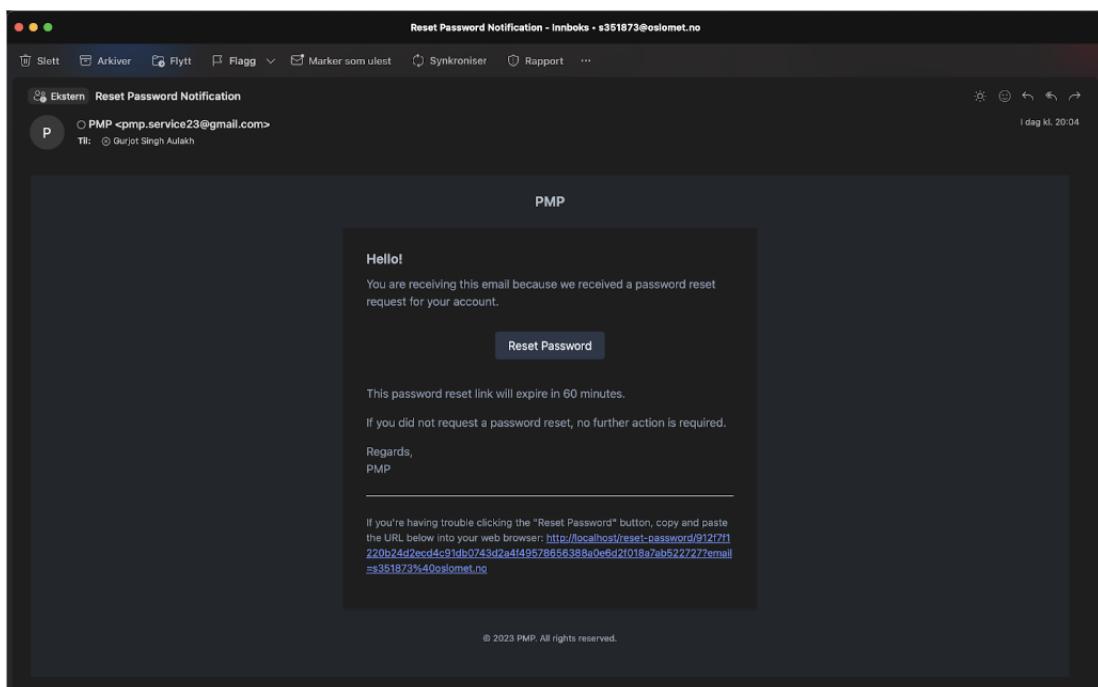


Figure 7.3: The password reset mail

### 7.1.3 Create an Account

Users who do not have an account can easily create one by selecting “Create an account” on the login page.

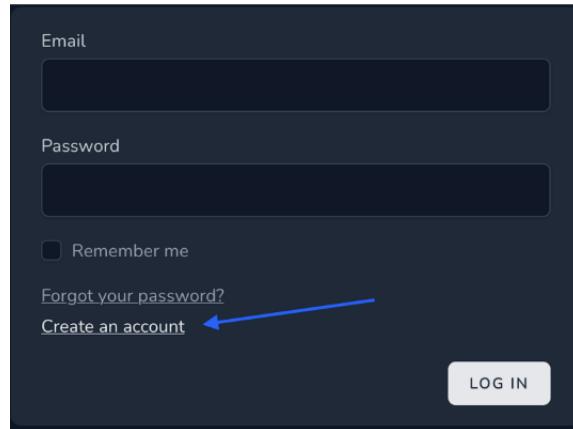


Figure 7.4: Link to the account creation form

In order to proceed, individuals are required to fill in the form and complete the registration process. They can either register as a company or as a student.

A screenshot of a registration form. It includes fields for 'Firstname', 'Lastname', 'Email', 'Password', and 'Confirm Password'. Below these is a section for 'Account Type' with two options: 'Company' (selected) and 'Student'. At the bottom left is a checkbox for 'I accept the Terms and Conditions', and at the bottom right is a 'REGISTER' button.

Figure 7.5: Here, you can specify the type of account you wish to create

To complete the registration process, individuals must adhere to the validations rules of the application which we already discussed in chapter [5.5.4](#).

After registration, users receive an email to verify their email address. Figure [7.5](#) displays the registration page, and Figure [7.6](#) shows the verification email.

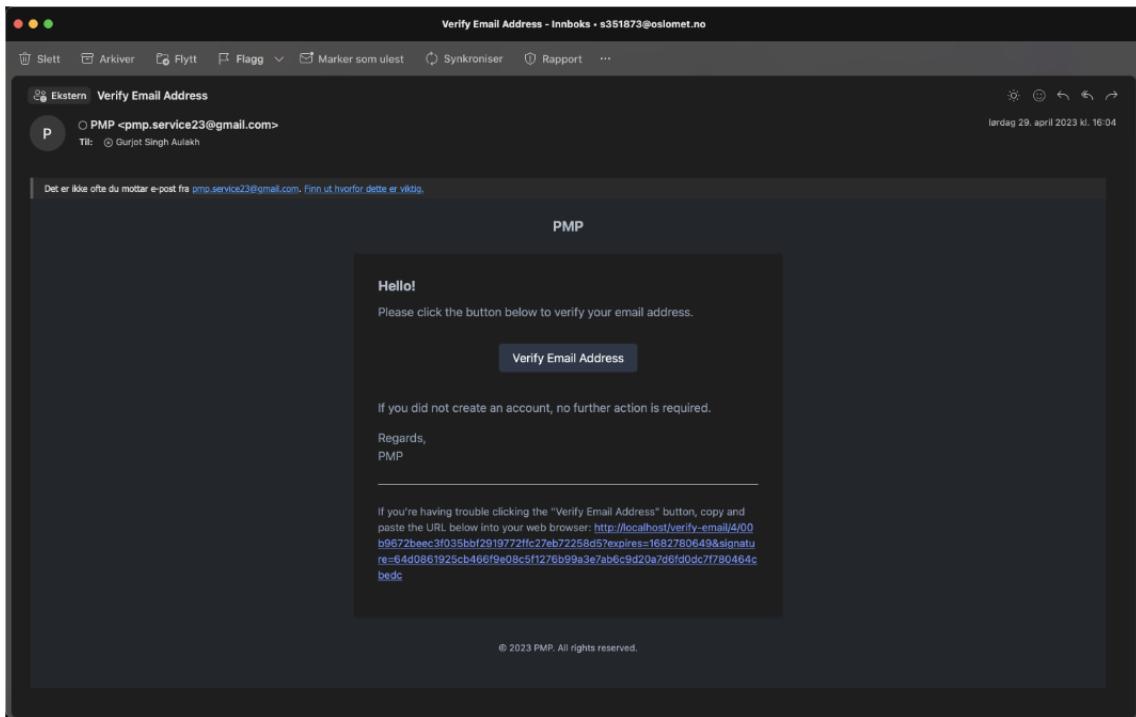


Figure 7.6: The verification email

## 7.2 The Core Flow Of The Application

### Company Flow

After freshly creating a company-user, also referred as company-representative, and logging in the company-user will be directed to the dashboard. Currently, you won't see any content as no projects have been created yet.

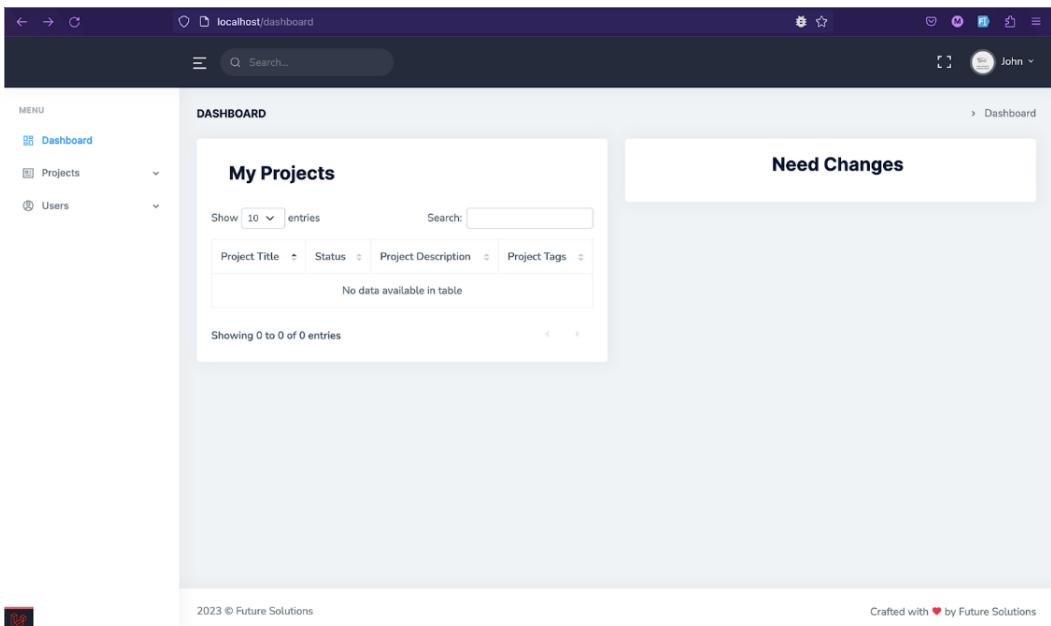


Figure 7.7: The company's dashboard

To create a project proposal, the user can select the 'Add Project' option from the side navigation.

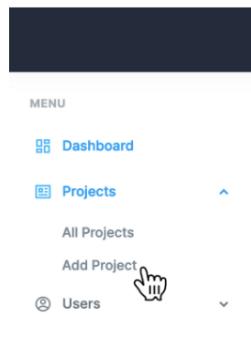


Figure 7.8: 'Add Project' option from the side navigation

To create a project, the user must first create a company as the newly created account does not have a company yet.

**CREATE COMPANY PAGE**

Company > Create company user

MENU

- Dashboard
- Projects
- Users

Name of Institution / Company / Organisation

Location

Website

Email

Phone

Company Logo

Browse... No file selected.

Submit

NO IMAGE AVAILABLE

Please fill in the company information.

Figure 7.9: Create a company first

**CREATE COMPANY PAGE**

Company > Create company user

Name of Institution / Company / Organisation

Location

Website

Email

Phone

Company Logo

Browse... VS logo.png

veridian solutions  
veridian solutions

Submit

Figure 7.10: Example data needed to create a company

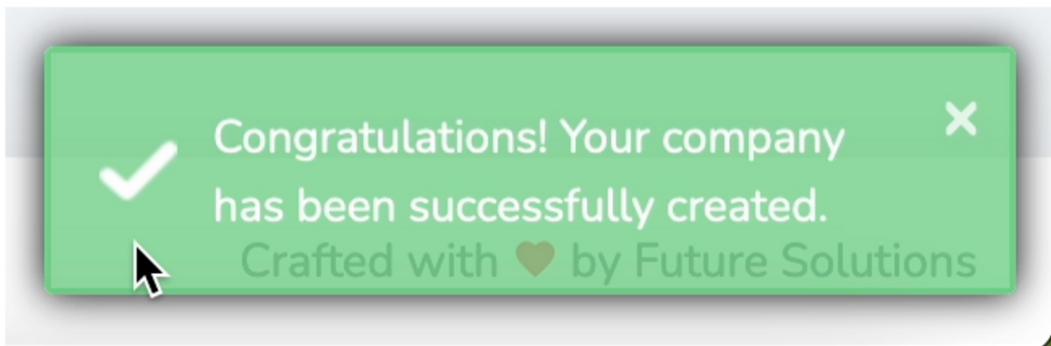


Figure 7.11: Company created successfully

Having a company allows the user to create projects, ensuring validation and ownership by the company rather than the user. This approach accommodates future changes in personnel or representation within the company while maintaining project continuity. Thus, it is more advantageous to link projects to companies rather than directly to users.

#### Create project proposal

The user must complete three sections to create a project proposal. Some inputs are required, while others have validation rules that must be followed in order to successfully create a project description. The company-user will now create a project proposal.

*Note “Aims” input field is empty, which will cause a validation error. However, this error occurs after we have been on page 4 and clicked on the submit button.*

**ADD PROJECT**

Projects > Add project

Mandatory fields are marked with a star \*

01 Project description      02 Project Details      03 Contract Details      04 Confirm Detail

Project Title \*

This can be a tentative or working title.

PHP Laravel Developer - Storage Management Application

Project description \*

Please provide a general description of the project and the key activities

We are seeking PHP Laravel Developer to join our dynamic team and spearhead the development of a cutting-edge storage management application. As a PHP Laravel Developer, you will play a pivotal role in designing, implementing, and maintaining an innovative solution that revolutionizes the way storage facilities handle their inventory.

Responsibilities:

- Design and develop a robust storage management application using PHP and Laravel framework, ensuring scalability, security, and performance.
- Develop intuitive user interfaces that provide storage workers with a user-friendly experience, enhancing productivity and ease of use.

181 WORDS POWERED BY TINY

Background for the project \*

Please provide some general information about the project, and the context in which it takes place (for example in a start-up, an established company or organisation, or as part of larger existing research project)

management application using PHP and Laravel. This user-friendly application streamlines operations, enables real-time inventory tracking, automates stock replenishment, and integrates seamlessly with existing systems. With a focus on scalability and security, our application revolutionizes the way storage facilities handle their inventory, enhancing efficiency and optimizing the customer experience.

Project aims \*

What would you like to achieve with this project?

Tags \*

Laravel x PHP x JQuery x HTML x CSS x JavaScript x

Previous      Next

Figure 7.12: Page 1, General project information

**ADD PROJECT**

Projects > Add project

Mandatory fields are marked with a star \*

01 Project description      02 Project Details      03 Contract Details      04 Confirm Detail

**Final deliverable**  
Please specify what you expect the students to deliver at the end of the project. This can be, for example, a prototype product or service, a case report, or scientific article.

The final deliverable for this project is a powerful storage management application built with PHP and Laravel. It provides a user-friendly interface for streamlined operations, efficient inventory management, real-time tracking, and automated stock replenishment. The application seamlessly integrates with existing systems, ensuring a cohesive workflow. With scalability and robust security measures in place, the final deliverable empowers storage facilities with an optimized solution for enhanced productivity and performance.

**Desired skills or competencies**  
Please specify any experience or skills that would be particularly advantageous for this project (e.g. familiarity with particular concepts, tools, software, techniques or methodologies).

- Experience in PHP and Laravel development.
- Understanding of web application development.
- Knowledge of HTML, CSS, JavaScript, and jQuery.
- Familiarity with SQL and efficient database design.
- Experience in integrating third-party APIs.
- Attention to detail and clean coding practices.
- Familiarity with Git and collaborative development.
- Strong communication and teamwork skills.

**Ideal number of students**  
(Groups are capped at a maximum of five.)

5

**Additional information**  
Please specify any additional information you feel is important to the successful fulfillment of this project.

It is not necessary that the students can PHP and Laravel, but very important that they are willing to explore and learn. This project will be a learning place and a place to experiment.

**Previous** **Next**

Figure 7.13: Page 2, Project details

**ADD PROJECT**

Projects > Add project

Mandatory fields are marked with a star \*

01 Project description      02 Project Details      03 Contract Details      04 Confirm Detail

Intellectual property, commercialization and confidentiality.

A Cooperation Agreement form must be signed at the start of the semester by all parties involved in the project.

There are a variety of standard agreements, tailored to different types of projections and collaborations. The agreements and information about them can be found here: <https://student.oslomet.no/en/student-agreements>

Please select the agreement you feel is most appropriate for the project.

\* If you are affiliated with the Grundergarasje at OsloMet you will receive a separate, tailored cooperation agreement.

If you have any questions or comments about the Cooperation Agreement please include them in the Additional Information box below

Cooperation Agreement

Standard agreement for the implementation of academic work ▾

Will any part of the academic work for this project:  
(please check each box as appropriate).

If any of the elements below are applicable to this project, please contact Tulpesh Patel ([tulpesh@oslomet.no](mailto:tulpesh@oslomet.no)) for more information.

- Be subject to a patent application
- Require protection of design
- Require registration of a brand
- Be commercialized
- Require a specific confidentiality agreement
- Require a postponement of publication

[Previous](#) [Next](#)

Figure 7.14: Page 3, Project agreements page

**ADD PROJECT**

Projects > Add project

Mandatory fields are marked with a star \*

01 Project description      02 Project Details      03 Contract Details      04 Confirm Detail



**Confirm Detail**  
Check if the information you have given is correct



**Previous**

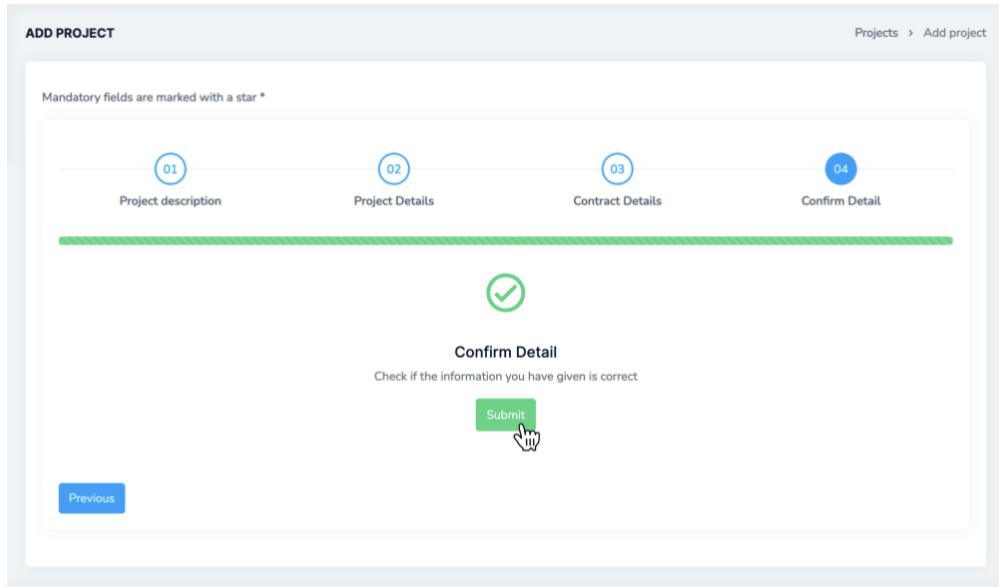


Figure 7.15: Page 4, final step to submit the project proposal

As mentioned above, missing a required field will prevent project submission, resulting in a validation error. The existing input will remain unchanged and unaffected.

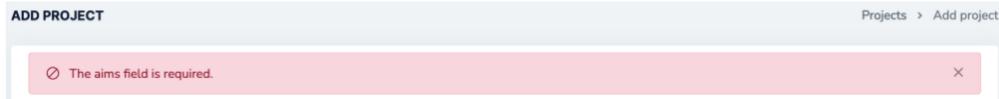


Figure 7.16: Back to page 1, validation error message

After creating the project, it appears under “All projects.”

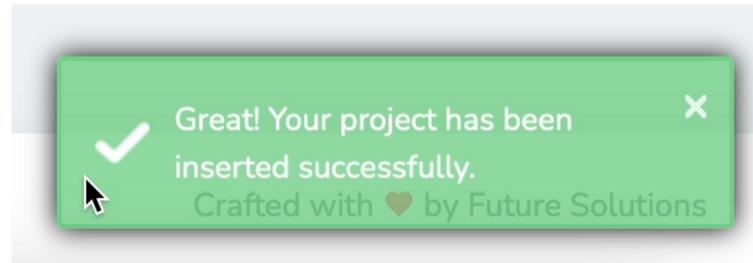


Figure 7.17: Project proposal created successfully

Upon creating a project, it is displayed under the section labeled “All projects.” However, for company users, we have renamed it to ”My projects” as we believe it is more meaningful in that context.

A screenshot of a web application interface. The top navigation bar includes a back arrow, forward arrow, refresh button, and a search bar with placeholder text "localhost/project". On the right side of the bar are user profile icons and a dropdown menu for "John". The main content area has a dark header with the text "ALL PROJECTS" and a sub-header "All Projects". Below this is a table with four columns: "Title", "Status", "Description", and "Tags". The first row in the table is for a project titled "PHP Laravel Developer - Storage Management Application" with a status of "NEW", a description of "We are seeking PHP L...", and tags "Laravel,PHP,JQuery,HTML,CSS,JavaScript". To the right of the table are "Copy", "Excel", and "PDF" buttons, and a "Column visibility" dropdown. A search bar with placeholder "Search:" is also present. At the bottom of the table, it says "Showing 1 to 1 of 1 entries". A small blue circular page number "1" is in the center. The footer contains the text "2023 © Future Solutions". A green success message box with a checkmark and the text "Great! Your project has been inserted successfully. Crafted with ❤ by Future Solutions" is overlaid on the right side of the table.

Figure 7.18: Project proposal appears in “All Projects” table with status “NEW”

## Course Coordinator Flow

After logging in, the course coordinator is directed to the Dashboard, which offers a comprehensive overview of vital information. This version of the dashboard was developed through multiple reviews and discussions to identify its most essential components. It presents various statistics and provides an overview of new and updated projects.

The screenshot shows the Course Coordinator Dashboard. At the top, there's a header bar with a search input and a user profile icon. Below it is a navigation menu with options like Dashboard, Projects, and Users. The main area is titled "DASHBOARD" and contains three summary cards: "Total Students" (0), "Total Companies" (2), and "Total Projects" (4). A large section below is titled "New Projects". It includes a table with columns: Company Name, Project Title, Project Description, and Project Tags. The table has four entries:

Company Name	Project Title	Project Description	Project Tags
Company X	ipsum cumque repudiandae sequi...	Placeat omnis iusto minus nemo...	laravel api backend
Company X	Mollitia natus vitae cum nulla...	Neque facere quisquam non sint...	laravel api backend
Company X	lure pariatur illo sint quia.	Non qui iste sed qui. Vلت of...	laravel api backend
Veridian Solutions	PHP Laravel Developer - Stora...	We are seeking PHP Laravel...	Laravel PHP JQuery HTML CSS JavaScript

Below this is a section titled "Updated Projects" with a table showing "No data available in table".

At the bottom of the page, there's a footer with copyright information: "2023 © Future Solutions" and "Crafted with ❤️ by Future Solutions".

Figure 7.19: Course coordinator dashboard where there only projects with status “new”

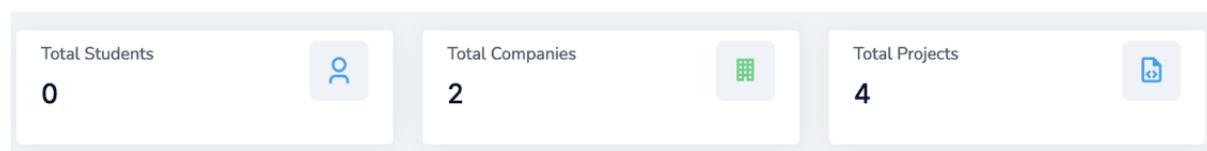


Figure 7.20: Overview of various statistics

New Projects							
The new projects table is a database table that stores information about projects, including their title, descriptions and tags. In addition to contact information for the company. It enables effective project management and coordination.							
<table border="1"> <tr> <td><a href="#">Copy</a></td><td><a href="#">Excel</a></td><td><a href="#">PDF</a></td><td><a href="#">Column visibility</a></td></tr> </table> <span style="float: right;">Search: <input type="text"/></span>			<a href="#">Copy</a>	<a href="#">Excel</a>	<a href="#">PDF</a>	<a href="#">Column visibility</a>	
<a href="#">Copy</a>	<a href="#">Excel</a>	<a href="#">PDF</a>	<a href="#">Column visibility</a>				
Company Name	Project Title	Project Description	Project Tags				
Company X	Ipsum cumque repudiandae sequi...	Placeat omnis iusto minus nemo...	<a href="#">laravel</a> <a href="#">api</a> <a href="#">backend</a>				
Company X	Mollitia natus vitae cum nulla...	Neque facere quisquam non sint...	<a href="#">laravel</a> <a href="#">api</a> <a href="#">backend</a>				
Company X	lure pariatur illo sint quia.	Non qui iste sed qui. Velit of...	<a href="#">laravel</a> <a href="#">api</a> <a href="#">backend</a>				
Veridian Solutions	PHP Laravel Developer - Storag...	We are seeking PHP Laravel...	<a href="#">Laravel</a> <a href="#">PHP</a> <a href="#">JQuery</a> <a href="#">HTML</a> <a href="#">CSS</a> <a href="#">JavaScript</a>				

Figure 7.21: Overview over all projects with status “NEW”

Updated Projects										
The projects table is a database table that stores information about ongoing or completed projects, including project title, descriptions, tags and project status. It is used to track progress, manage resources, and facilitate communication among project stakeholders.										
<table border="1"> <tr> <td><a href="#">Copy</a></td><td><a href="#">Excel</a></td><td><a href="#">PDF</a></td><td><a href="#">Column visibility</a></td><td><a href="#">Project Tags</a></td></tr> </table> <span style="float: right;">Search: <input type="text"/></span>					<a href="#">Copy</a>	<a href="#">Excel</a>	<a href="#">PDF</a>	<a href="#">Column visibility</a>	<a href="#">Project Tags</a>	
<a href="#">Copy</a>	<a href="#">Excel</a>	<a href="#">PDF</a>	<a href="#">Column visibility</a>	<a href="#">Project Tags</a>						
Company Name										
Project Title										
Project Description										
Project Tags										
Company Email										
No data available in table										
Showing 0 to 0 of 0 entries										

Figure 7.22: Currently there are no projects with status “UPDATED”

*Extra functionality: The dashboard for the course coordinator also contains a toolbar giving the opportunity to edit the table view (column visibility), filter projects, copy projects to excel or save it as a PDF’s.*

New Projects							
The new projects table is a database table that stores information about projects, including their title, descriptions and tags. In addition to contact information for the company. It enables effective project management and coordination.							
<table border="1"> <tr> <td><a href="#">Copy</a></td><td><a href="#">Excel</a></td><td><a href="#">PDF</a></td><td><a href="#">Column visibility</a></td></tr> </table> <span style="float: right;">Search: <input type="text"/></span>			<a href="#">Copy</a>	<a href="#">Excel</a>	<a href="#">PDF</a>	<a href="#">Column visibility</a>	
<a href="#">Copy</a>	<a href="#">Excel</a>	<a href="#">PDF</a>	<a href="#">Column visibility</a>				

Figure 7.23: Toolbar

## Quality ensuring project proposals

Creating project descriptions from project proposals used to be a slow and manual process for our course coordinator. It involved copying and pasting from Nettskjema to Microsoft Word and then exporting as PDF for students. Our application automates this entire process, providing a highly efficient solution. This view shows the project description, which will be visible for students once the project is published.

This view shows a project proposal. The course coordinator can choose to "Approve," "Edit," or request changes by clicking on the "Need Changes" button.

If the course coordinator chooses to "Approve" a project, the project will become visible and published to students.

If the course coordinator chooses to "Edit" a project, he can add the necessary information they believe is required before it is published to students.

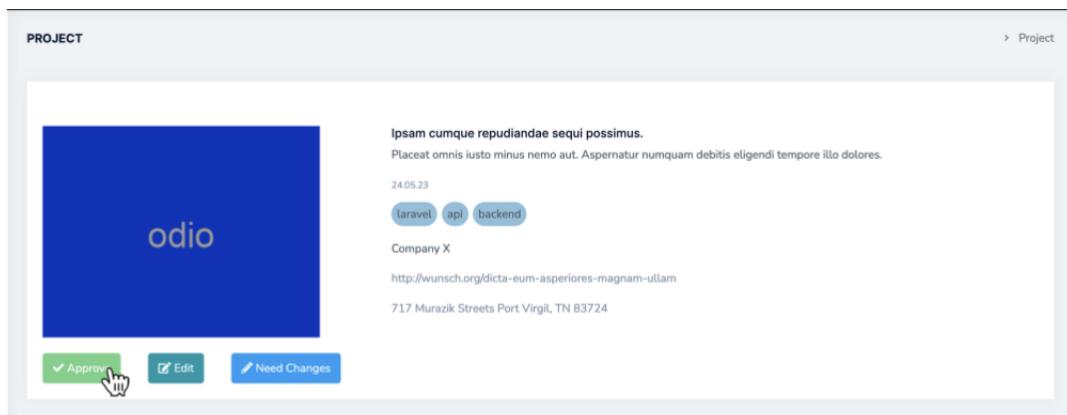


Figure 7.24: Course Coordinator approves a project

The course coordinator can request changes to the project by clicking on the "Need Changes" button. This opens a modal form where a message to the company can be created which addresses what changes are needed. Once the modal is submitted, the project will get status "NEED CHANGES" and the message attached as a comment, which the company user can view.

**PHP Laravel Developer - Storage Management Application**

We are seeking PHP Laravel Developer to join our dynamic team and spearhead the development of a cutting-edge storage management application. As a PHP Laravel Developer, you will play a pivotal role in designing, implementing, and maintaining an innovative solution that revolutionizes the way storage facilities handle their inventory.

**Responsibilities:**

- Design and develop a robust storage management application using PHP and Laravel framework, ensuring scalability, security, and performance.
- Develop intuitive user interfaces that provide storage workers with a user-friendly experience, enhancing productivity and ease of use.
- Integrate third-party APIs and services to enable seamless data synchronization and extend the application's functionalities.
- Write clean, maintainable, and well-documented code.
- Conduct testing, debugging, and troubleshooting.

**Qualities:**

- Proficient in front-end technologies such as HTML, CSS, JavaScript, and jQuery for seamless user interface development.
- Experience working with databases, including SQL, and designing efficient database structures.
- Familiarity with version control systems (e.g., Git) and collaborative development workflows.

Join our team and contribute to create an exceptional storage management application using PHP and Laravel. Be part of a dynamic and collaborative environment and expand your skills and creativity.

24.05.23

Laravel PHP JQuery HTML CSS JavaScript

Veridian Solutions  
<https://www.veridiansolutions.com>  
Oslo

Figure 7.25: Course coordinator requesting changes

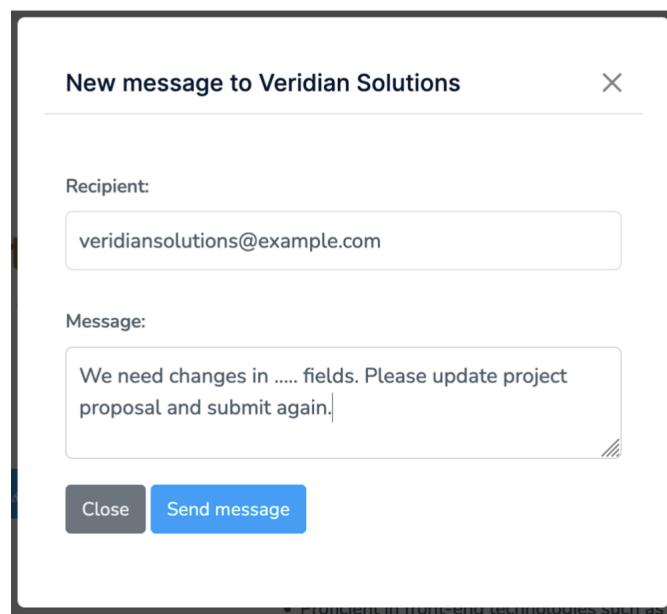


Figure 7.26: Course coordinator addressing changes needed to the project proposal

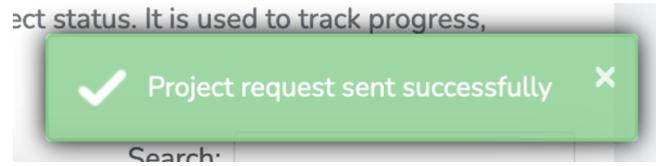


Figure 7.27: Change request sent successfully

The course coordinator can conveniently access an overview of all projects and their current status by simply clicking on "All projects" in the side navigation.

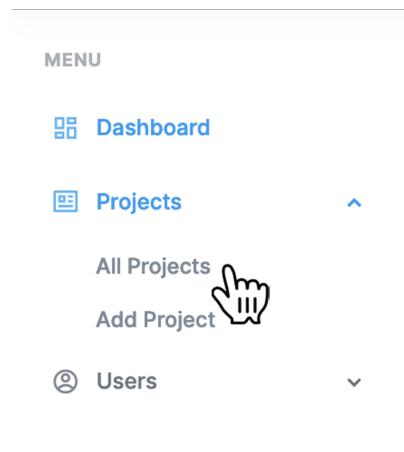


Figure 7.28: "All projects" link in the side navigation

If it says "Published," it means the project has been processed and quality checked by the course coordinator and is now visible to students.

If it says "New" under the status, it means the project is new and requires quality assurance/approval from the course coordinator.

If it says "Needs Changes," it means the course coordinator has requested changes to the project proposal and sent it back to the corresponding company for modifications.

All Projects				
The table contains four columns: Title, Status, Description, and Tags. The Title column likely contains the name of a project, the Status column indicates the current stage or progress of the project, the Description column likely provides a brief overview of the project, and the Tags column may contain keywords or categories associated with the project.				
<a href="#">Copy</a>	<a href="#">Excel</a>	<a href="#">PDF</a>	<a href="#">Column visibility</a>	Search:
Title	Status	Description	Tags	
Ipsum cumque repudianda sequi possimus.	PUBLISHED	Placeat omnis iusto...	laravel, api, backend	<a href="#"></a> <a href="#"></a>
Iure pariatur illo sint quia.	NEW	Non qui iste sed qui...	laravel, api, backend	<a href="#"></a> <a href="#"></a>
Mollitia natus vitae cum nulla debitis nemo dolorum.	NEW	Neque facere quisqua...	laravel, api, backend	<a href="#"></a> <a href="#"></a>
PHP Laravel Developer - Storage Management Application	NEED CHANGES	We are seeking PHP L...	Laravel,PHP,jQuery,HTML,CSS,JavaScript	<a href="#"></a> <a href="#"></a>

Showing 1 to 4 of 4 entries

Figure 7.29: Overview over all projects

## Back To Company Flow

After a change has been requested from the course coordinator, the company user can view the request in the dashboard. On the left side, all the projects owned by this company appear, while on the right side, a window specifically for the projects that need changes is displayed.

The screenshot shows a web-based company dashboard. On the left, there's a sidebar with a 'MENU' section containing 'Dashboard', 'Projects', and 'Users'. The main area is titled 'DASHBOARD' and contains two sections: 'My Projects' and 'Need Changes'. The 'My Projects' section shows a single entry: 'PHP Laravel Developer - Storage Management Application' with a status of 'NEED CHANGES'. The 'Need Changes' section displays a message from a 'Course Coordinator [Course Coordinator]' stating: 'We are seeking PHP Laravel Developer to join ou...' with a list of tags: Laravel, PHP, jQuery, HTML, CSS, JavaScript. Below the message is a comment field with the placeholder 'Comment from Course Coordinator [Course Coordinator]' and a timestamp '24.05.23'. At the bottom of the page, there's a footer with the text '2023 © Future Solutions' and 'Crafted with ❤ by Future Solutions'.

Figure 7.30: Company dashboard after receiving a change request on a project proposal

My Projects	
Show	entries
Project Title	Status
● PHP Laravel Developer - Storage Management Application	NEED CHANGES
Project Description	
We are seeking PHP Laravel...	
Project Tags	
Laravel,PHP,JQuery,HTML,CSS,JavaScript	
Company Email	
<a href="mailto:veridiansolutions@example.com">veridiansolutions@example.com</a>	

Showing 1 to 1 of 1 entries

1

Figure 7.31: More details about the project in the dashboard

## Need Changes

**PHP Laravel Developer - Storage Management Application**

We are seeking PHP Laravel Developer to join our dynamic team and spearhead the development of a cutting-edge storage management application. As a PHP Laravel Developer, you will play a pivotal role in designing, implementing, and maintaining an innovative solution that revolutionizes the way storage facilities handle their inventory.

**Comment from Course Coordinator [Course Coordinator]**

We need changes in .... fields. Please update project proposal and submit again.

24.05.23

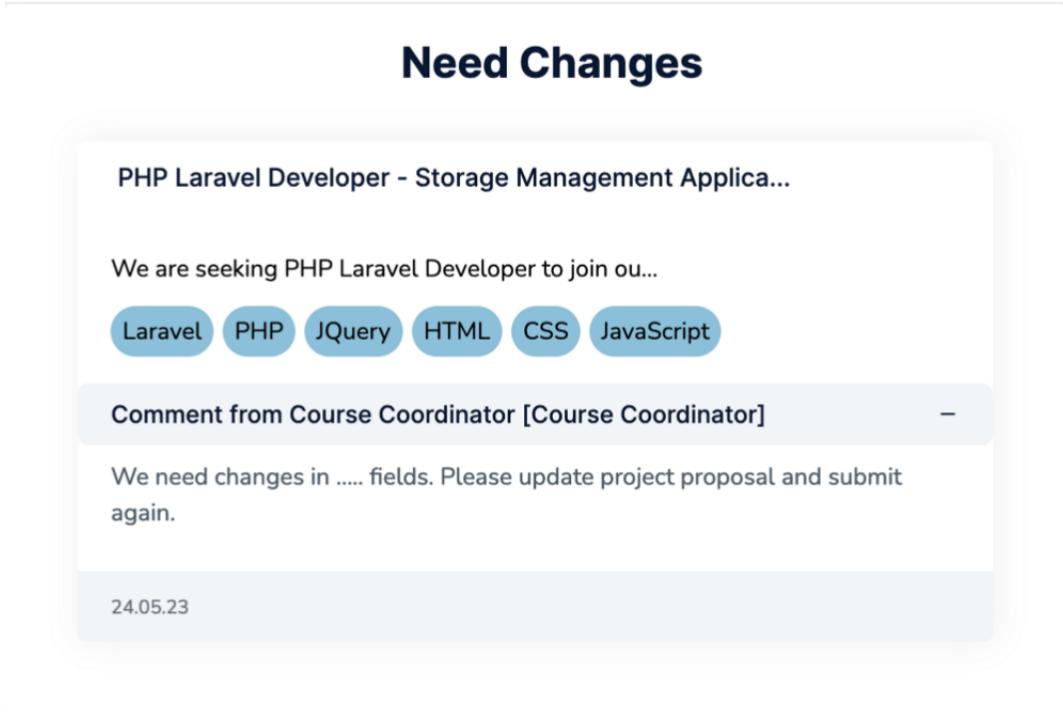


Figure 7.32: The changes have been done by the company

**veridian solutions**  
veridian solutions

**PHP Laravel Developer - Storage Management Application**

We are seeking PHP Laravel Developer to join our dynamic team and spearhead the development of a cutting-edge storage management application. As a PHP Laravel Developer, you will play a pivotal role in designing, implementing, and maintaining an innovative solution that revolutionizes the way storage facilities handle their inventory.

**Responsibilities:**

- Design and develop a robust storage management application using PHP and Laravel framework, ensuring scalability, security, and performance.
- Develop intuitive user interfaces that provide storage workers with a user-friendly experience, enhancing productivity and ease of use.
- Integrate third-party APIs and services to enable seamless data synchronization and extend the application's functionalities.
- Write clean, maintainable, and well-documented code.
- Conduct testing, debugging, and troubleshooting.

**Qualities:**

- Proficient in front-end technologies such as HTML, CSS, JavaScript, and jQuery for seamless user interface development.
- Experience working with databases, including SQL, and designing efficient database structures.
- Familiarity with version control systems (e.g., Git) and collaborative development workflows.

Join our team and contribute to create an exceptional storage management application using PHP and Laravel. Be part of a dynamic and collaborative environment and expand your skills and creativity.

24.05.23

Laravel PHP JQuery HTML CSS JavaScript

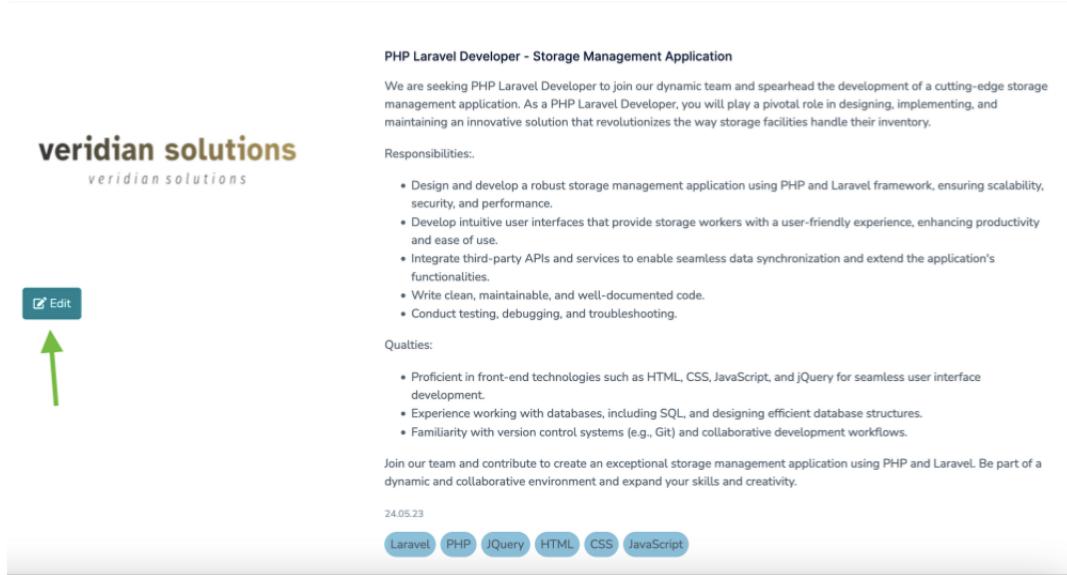


Figure 7.33: Company representative modify an existing project proposal

Tags \*

[Previous](#) [Next](#)

Figure 7.34: Company representative modify tags of an existing project proposal

Mandatory fields are marked with a star \*

01 Project description    02 Project Details    03 Contract Details    04 Confirm Detail

**Confirm Detail**  
If several languages coalesce, the grammar of the resulting

[Previous](#) [Submit](#) [Next](#)

Figure 7.35: The company representative submits changes to existing project proposal



Figure 7.36: Project updated successfully

Company representatives have multiple means to verify the successful update of a project. Firstly, they receive a clear indication of success through a success message following the submission of changes. Additionally, they can gain an overview of all projects owned by them by navigating to the "All projects" section in the side navigation. They can confirm the update by observing the changed status of the project proposal, which should now reflect as "UPDATED." Furthermore, they have

the option to double-check by accessing the edit view and thoroughly reviewing the updated project proposal.

The screenshot shows a web interface titled "MY PROJECTS". At the top, there are buttons for "Copy", "Excel", "PDF", and "Column visibility". A search bar is also present. The main area displays a table with columns: Title, Status, Description, and Tags. One row is visible, showing "PHP Laravel Developer - Storage Management Application" with status "UPDATED", description "We are seeking PHP L...", and tags "Laravel, PHP". Below the table, it says "Showing 1 to 1 of 1 entries". A green curved arrow points upwards from the bottom of the table towards the top of the page.

Figure 7.37: Company-representative views all projects owned by them

## Back To Course Coordinator Flow

After changes have been made to an existing project proposal by the company, it appears under the “Updated projects”-table on the course coordinator’s dashboard.

The screenshot shows a dashboard for a "Course Coordinator". On the left, there is a sidebar with "MENU" and options for "Dashboard", "Projects", and "Users". The main area has a "DASHBOARD" header with three summary boxes: "Total Students" (0), "Total Companies" (2), and "Total Projects" (4). Below this is a section titled "Updated Projects". It contains a table with columns: Company Name, Project Title, Project Description, Project Tags, and Company Email. One entry is shown: "Veridian Solutions" with "PHP Laravel Developer - Storag..." as the project title, "We are seeking PHP Laravel...", tags "Laravel, PHP", and email "veridiansolutions@example.com". A green curved arrow points upwards from the bottom of the table towards the top of the page.

Figure 7.38: Overview of the updated projects on the course coordinators dashboard

The course coordinator approved the project and published it. It will now be visible to the students

as well.

project status. It is used to track progress,

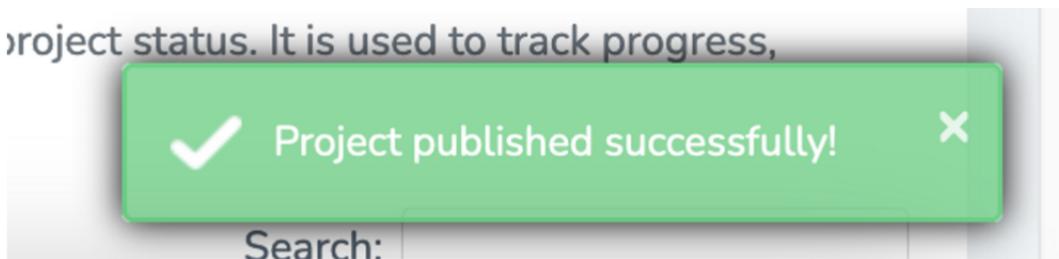


Figure 7.39: Success message, project proposal submitted

## Student Flow

After that the project was approved by the course coordinator, it appears on the students' dashboard.

That students can apply for one or multiple project proposals. After a student applies for a project, they will be added as an applicant to the project and become visible to the company. The company can then either confirm, approve, or reject the student's application.

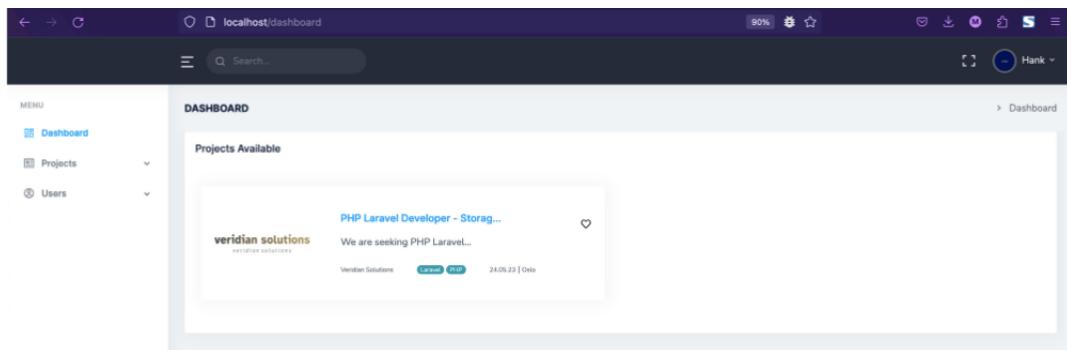


Figure 7.40: Overview of the projects on the student dashboard

That students can apply for one or multiple project proposals. After a student applies for a project, they will be added as an applicant to the project and become visible to the company. The company can then either confirm, approve, or reject the student's application.

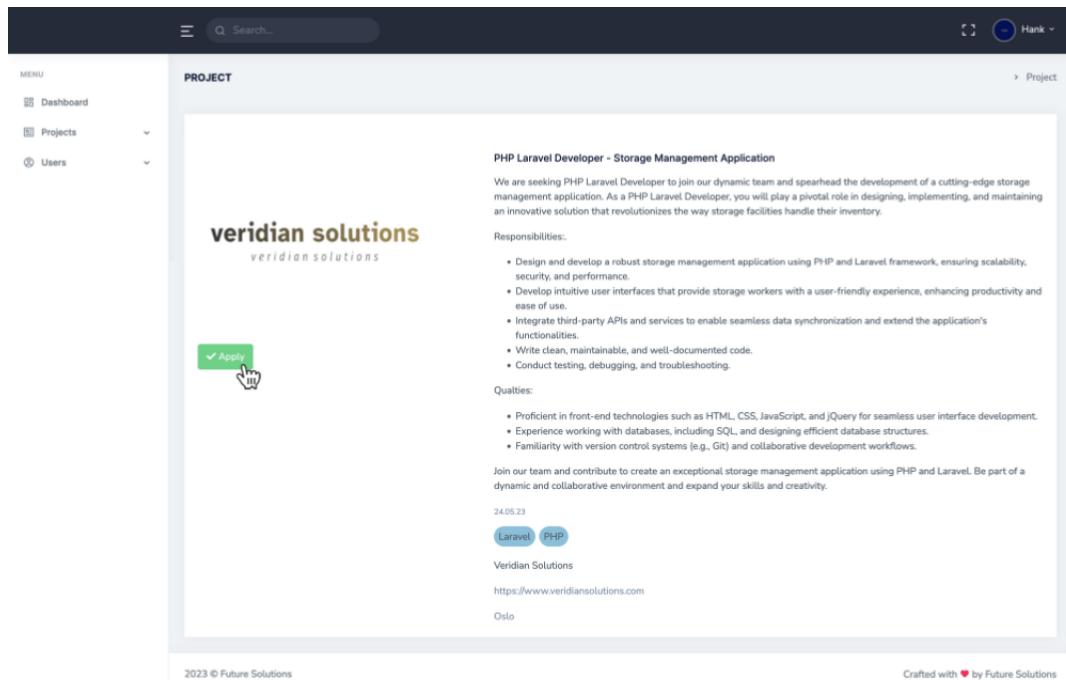


Figure 7.41: Overview of one of the project's proposals

## 7.3 Common Functionalities for all actors

### Profile functionality

Before:

The profile can be seen here. Changes can be made to it by simply clicking on the "Edit Profile" button.

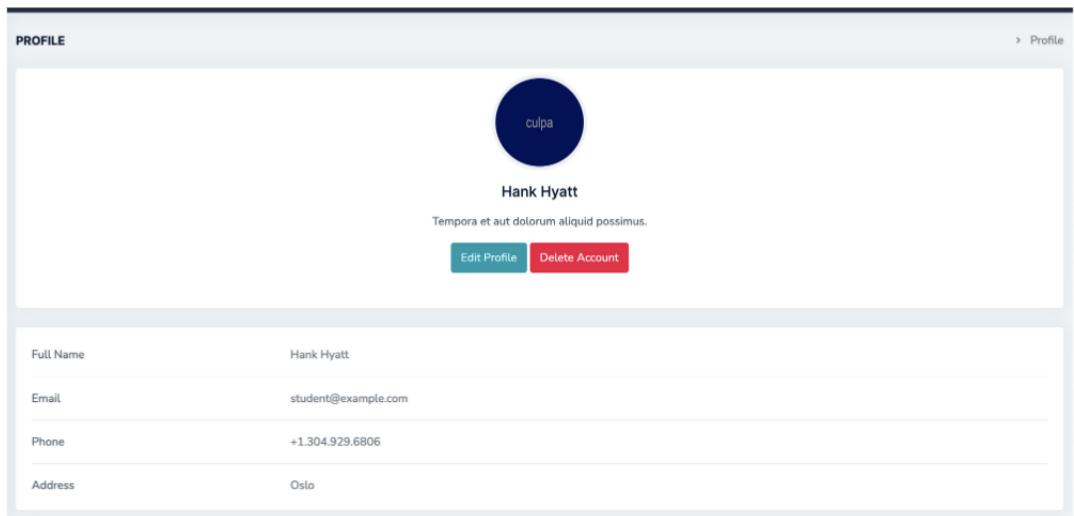


Figure 7.42: Overview of the profile page

A new window will be opened with profile details that can be changed.

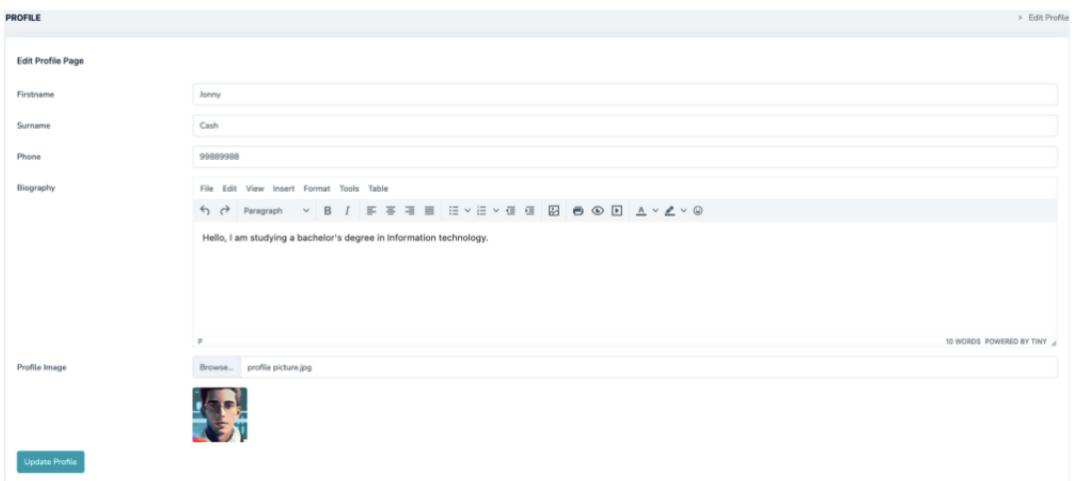


Figure 7.43: Overview of the profile details

The profile after the changes have been made.

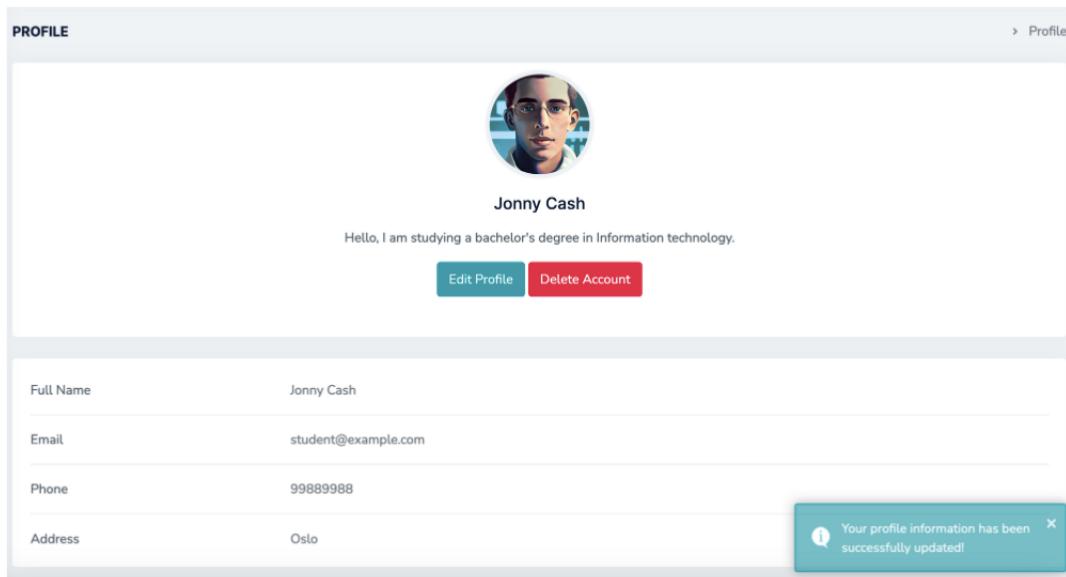


Figure 7.44: Overview of profile page after changes

If the wrong password is used, an error message will appear.

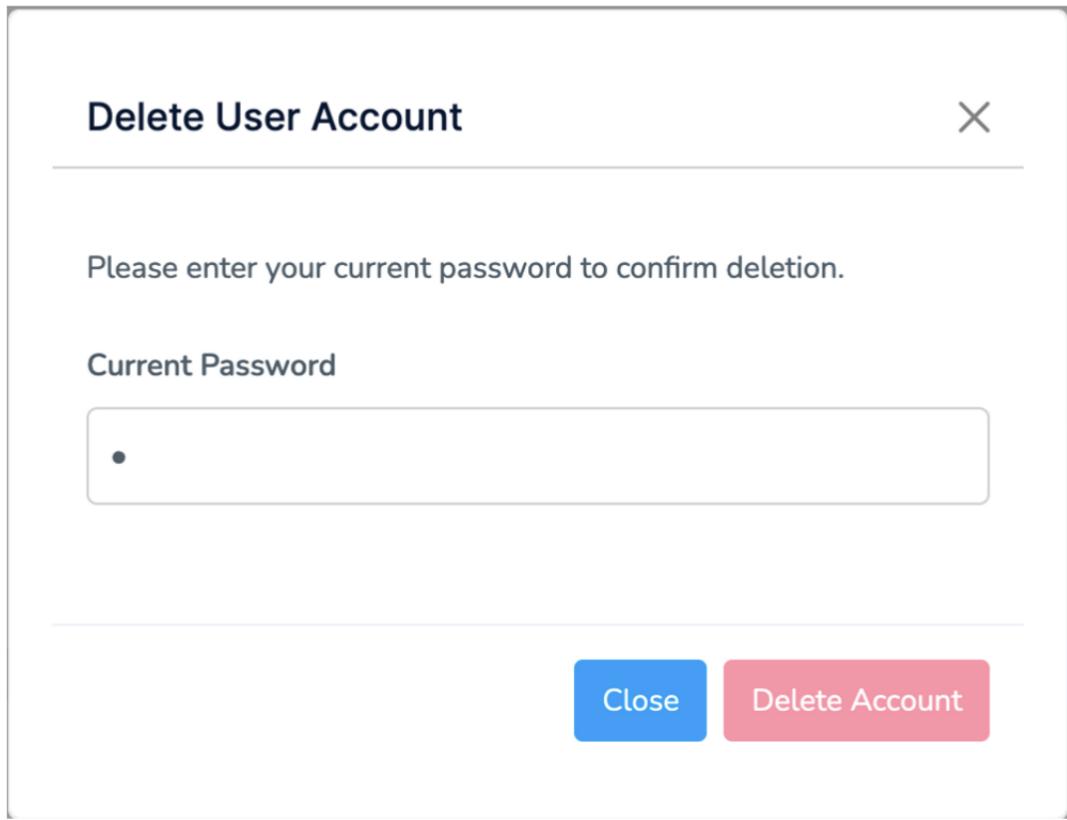


Figure 7.45: Overview of the input field

Here is the error message: "Incorrect password. Please try again."

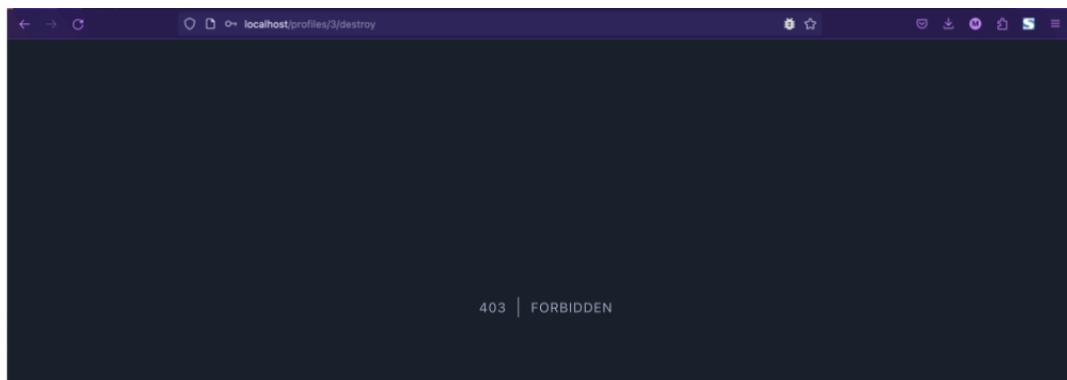


Figure 7.46: Overview of the error message

This shows that the student is no longer registered.

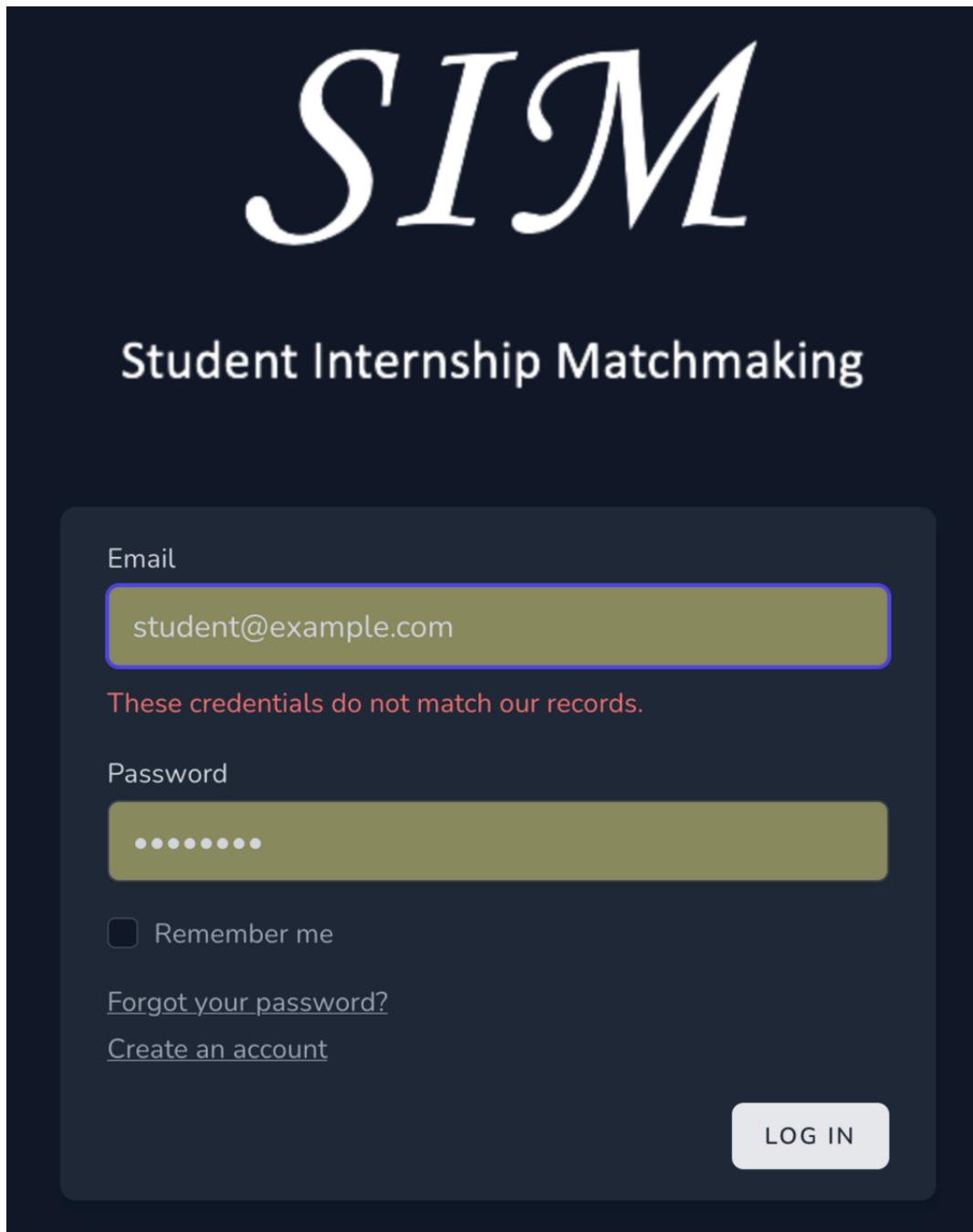


Figure 7.47: Showing that the student has been deleted

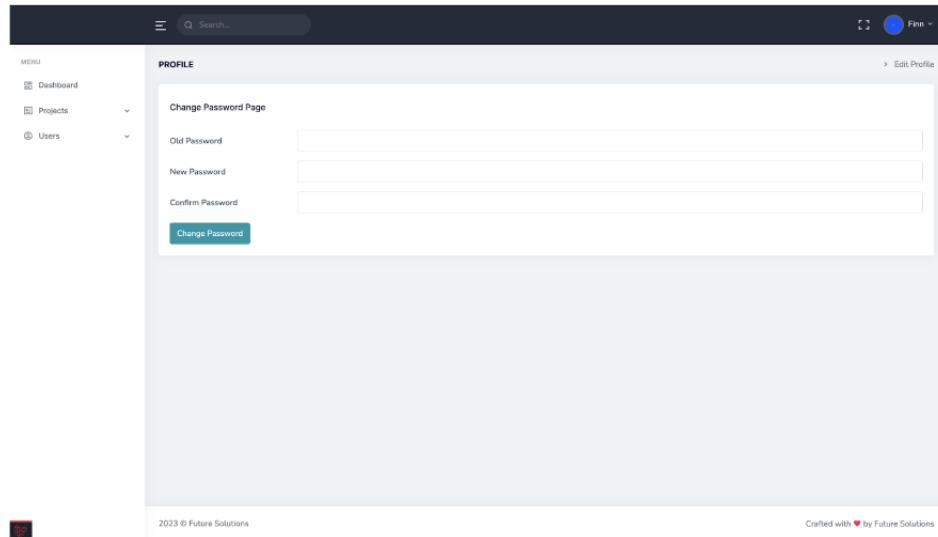


Figure 7.48: Change password page

## User functionality

The "Users" section overviews registered users, categorized by role (course coordinator, companies, and students). All roles can view contact information, while administrators have additional privileges. Figure 25 through Figure 28 display the user pages of the web application.

The screenshot shows a web application interface. On the left is a sidebar with a 'MENU' section containing 'Dashboard', 'Projects', and a 'Users' section with 'All Users', 'All Admins', 'All Companies', and 'All Students'. The main content area is titled 'ALL USERS' and contains a table with three entries:

Name	Email
Finn D'Amore	company@example.com
Moshe Ledner	student@example.com
Thelma West	admin@example.com

Below the table, it says 'Showing 1 to 3 of 3 entries'. At the bottom right of the content area is a blue circular button with a white number '1'. The footer of the page includes a '2023 © Future Solutions' logo, a small red icon, and the text 'Crafted with ❤️ by Future Solutions'.

Figure 7.49: The “All Users” page

This screenshot shows the 'All Admins' page of the same application. The sidebar on the left is identical to Figure 7.49. The main content area is titled 'ALL ADMINS' and contains a table with one entry:

Name	Email
Thelma West	admin@example.com

Below the table, it says 'Showing 1 to 1 of 1 entries'. At the bottom right of the content area is a blue circular button with a white number '1'. The footer of the page includes a '2023 © Future Solutions' logo, a small red icon, and the text 'Crafted with ❤️ by Future Solutions'.

Figure 7.50: The “Admins” page

The screenshot shows a web application interface. On the left is a sidebar titled "MENU" with options: Dashboard, Projects, Users (which is expanded), All Users, All Admins, All Companies (which is selected and highlighted in blue), and All Students. The main content area has a title "ALL COMPANIES". A descriptive text states: "The 'all companies' table contains contact information for various companies, including the name of the contact person, the company name, and their email address. It serves as a directory for easy access and reference to these details." Below this is a table with three columns: "Contact persons name", "Company", and "Email". A single row is present: "Finn D'Amore", "Company X", and "company@example.com". At the bottom of the table are buttons for "Copy", "Excel", "PDF", and "Column visibility", and a search bar labeled "Search: [ ]". The footer of the page includes the text "2023 © Future Solutions", a small logo, and "Crafted with ❤ by Future Solutions".

Contact persons name	Company	Email
Finn D'Amore	Company X	company@example.com

Figure 7.51: The “Companies” page

This screenshot shows the "Students" page of the application. The sidebar is identical to Figure 7.51, with "All Students" selected. The main content area has a title "ALL STUDENTS". A descriptive text states: "The 'all students' table contains records of students with their first name, last name, email, and status. This table provides a comprehensive list of all students and their current enrollment status." Below this is a table with four columns: "Firstname", "Lastname", "Status", and "Email". A single row is present: "Moshe", "Ledner", "NOT APPLIED", and "student@example.com". At the bottom of the table are buttons for "Copy", "Excel", "PDF", and "Column visibility", and a search bar labeled "Search: [ ]". The footer of the page includes the text "2023 © Future Solutions", a small logo, and "Crafted with ❤ by Future Solutions".

Firstname	Lastname	Status	Email
Moshe	Ledner	NOT APPLIED	student@example.com

Figure 7.52: The “Students” page

## The search bar functionality

The search bar enables users to search for projects, project descriptions, or tags within the web application. Search results are displayed on a separate result page. Figure 7.53 demonstrates the result page after performing a search.

A screenshot of a web application interface. At the top, there is a dark header bar with a search bar containing the text "laravel". Below the header is a navigation menu on the left with options: Dashboard, Projects (selected), and Users. The main content area is titled "RESULTS" and contains a heading "Results" with a sub-instruction: "The results table through search bar contains project title, description, and tags. It displays relevant projects matching the search query, making it easier for users to find the desired projects." Below this is a table with three columns: "Project Title", "Project Description", and "Project Tags". There are three rows of results, each with a truncated title and description, and a set of tags including "laravel", "api", and "backend". A green rectangular box highlights the entire results section. At the bottom of the page, there is footer text: "2023 © Future Solutions" and "Crafted with ❤ by Future Solutions".

Project Title	Project Description	Project Tags
Voluptatem aperiam velit nulla...	Facilis labore facil...	laravel api backend
Fugit iure et ut voluptates.	Repellat eum vel ea...	laravel api backend
Sit accusantium sit voluptatem...	Est et quos architect...	laravel api backend

Figure 7.53: The result page after searching

# **Chapter 8**

## **Conclusion and discussion**

### **8.1 Learning Outcome**

Working on this project has greatly benefited our team. It has given us a real sense of working in a team and a taste of working on real-life projects in the workplace. Although we did not have direct contact with a company throughout the project, our client acted as a supervisor, making us feel like we were collaborating with a business.

Another advantage is that we always had to come up with solutions when we encountered challenges, which taught us a lot about confidence and strengthened our skills and experiences. Each team member contributed to the project well, and the whole team functioned as a unit, even though it was sometimes challenging to keep communication at the same level.

Despite being unable to fulfill all the requirements 100%, especially the non-functional ones, we are satisfied with the effort we put in. We are now excited to see our work being used in practice and the positive impact the application can have.

## **8.2 What is the product's utility value?**

This application has shown more advantages and usefulness than we initially anticipated. Although it was primarily developed to make administrative tasks more manageable for administrators, it has the potential to benefit businesses and students greatly. The main benefit of the platform is that course coordinators can scale up the number of projects, companies and students, without increasing the resources needed to manage things.

Communication is improved with this application. This saves time and gives students a more excellent overview of available opportunities. It can also help to find projects more aligned with the student's interests and skills. Overall, this application will be of great benefit to all parties involved.

## **8.3 What would we have done differently?**

If we were to do this project again or work on a similar type of project, we would initially use the experience we had learned through this project and make some changes. First of all, we would use a different type of planning for implementation and documentation. Although we did well with this project and planning, we would have saved much more time if we had executed this project with, for example, the waterfall method, where the requirements are determined from the start. Changes in requirements caused the actual implementation not always to follow the project plan we had created.

Some choices we made regarding implementing the application could also have been different. A specific example is if we had dropped using Blade and instead used a different frontend framework like Angular or React. We could have done several things differently, such as implementing the application. Dropping Blade for the frontend and using a frontend framework like Angular/React/Vue would have been better because:

- API application testing would have been much more accessible.
- The frontend would have been more dynamic. It would have been much easier to implement, for example, the favorite function. (here, we need a reference to it)

- It would have enabled better separation of concern.
- We would have had more choices regarding the frontend because we could have used several libraries for components in frameworks like React.

Regarding user-testing the application, we would have planned it much earlier than we had done. This is because we did not know it would take time to get the application approved by SIKT and TSD. Instead, we could have also spent more time testing the application with actual companies and having more tests with employers to avoid as many change requests as possible.

## 8.4 Feedback from the Course Coordinator

The course coordinator had only one day per week available to work on the project due to limited time in the project schedule. This could be challenging for our team, so we needed to adapt to the course coordinator's schedule. One of the solutions discussed was to schedule meetings of different lengths depending on the necessity, allowing the team to work more efficiently and use time better. Another solution suggested was to find external mentors who could assist us with the project, providing extra support, expertise, and an outside perspective.

It was decided that questions requiring immediate answers should be communicated via Slack to ensure good communication within the team. In contrast, questions that could wait a day should be sent via email. This helped reduce disruptions and maintain focus on the work being done.

Most of the feedback from the course coordinator was upbeat throughout the project. The course coordinator was quite satisfied with creating the requirement specification and how we collaborated with him to realize his requirements in practice. During the project, course coordinator identified some challenges related to project planning and the need for faster feedback and answers to questions. Course coordinator's feedback was positively received, and the group worked together to overcome the challenges, which helped strengthen teamwork and ensure the project's successful completion.

The course coordinator has issued a certificate for the work we carried out during the project period. This certificate can be found in its entirety in Appendix [A.14](#).

## **8.5 Status of further development and production setting of the product**

The course coordinator plans to have the product ready for production and testing by July, following the final changes. Our next step is to attempt to deploy it on Microsoft Azure or AWS to achieve this goal. Additionally, during the development process, we were contacted by a representative from the Department of Behavioral Psychology at OsloMet, as they are working on a similar solution for their master's program, enabling students to find external collaborators for research projects. We discussed the possibility of collaboration in the fall of 2023, based on the MVP we have developed and are currently in the planning phase for. Furthermore, we aim to further enhance the product after the summer and make it available to more schools and programs. Another significant aspect of future development is the implementation of Feide authentication, which can be utilized more efficiently in Norway since Feide access is available to everyone in the country.

We can also continue to work on developing the frontend and separate it from the backend so that we can focus on frontend tasks separately from backend tasks.

Regarding quality assurance of the product, we need to work more on this in the other development phase. The extra focus should be on testing, including Test Driven Development (TDD), system testing, integration testing, and automated testing. We also need to structure the folder in a more organized and clear way according to "Clean structure."

## **8.6 Summary and Conclusion**

Our idea turned into reality in the form of a web application. This was achieved through hard work, collaboration, and consistency. All group members contributed to the project, working in parallel and together. We did this with agile development, working in sprints, and having retrospective meetings. All the members agreed that the project was insightful, instructive, and exciting. We have gained much new knowledge and experience through the software development process in practice.

We are very proud of our solution, even though we did not manage to host the application on OsloMet's server within the project's time frame. However, the project course coordinator was pleased with the final result and looks forward to potentially using the application the following semester. The application will contribute to the "IT project in practice" course being available for more students since the internship matchmaking process is faster and administration is more accessible. Our group is looking forward to continuing this application's development after the bachelor project.

# Bibliography

- [Fre09] Maxim Freidgeim. *What are the differences between unit tests, integration tests, smoke tests, and regression tests?* <https://stackoverflow.com/questions/520064/what-are-the-differences-between-unit-tests-integration-tests-smoke-tests-and>. [Online; accessed May 15, 2023]. Feb. 2009.
- [WG12] W. Eric Wong and Ian Gorton. “Software Modularity: A Review and New Research Directions”. In: *ACM Computing Surveys (CSUR)* 45.1 (2012). [Online; accessed May 9, 2023], pp. 1–36. DOI: [10.1145/2379776.2379778](https://doi.org/10.1145/2379776.2379778).
- [Moo16] Christopher Moody. “Work Wanted: Cybersecurity jobs a priority for government”. In: *The Florida Times-Union* (July 2016). [Online; accessed April 29, 2023].
- [Sjo16] L. A. Sjo. *Overordnet Testplan - MUSIT Ny IT-arkitektur, Pilot og Hovedprosjekt*. [https://wiki.uio.no/usit/musit/images/0/0a/Overordnet\\_testplan\\_MUSIT\\_Ny\\_IT-arkitektur%2C\\_Pilot\\_og\\_Hovedprosjekt\\_v1.0.pdf](https://wiki.uio.no/usit/musit/images/0/0a/Overordnet_testplan_MUSIT_Ny_IT-arkitektur%2C_Pilot_og_Hovedprosjekt_v1.0.pdf). [Online; accessed May 15, 2023]. June 2016.
- [Don21] Fred Donovan. “What Is STRIDE Threat Modeling: Anticipate Cyberattacks”. In: *Security Intelligence* (Jan. 2021). [Online; accessed May 24, 2023]. URL: <https://securityintelligence.com/articles/what-is-stride-threat-modeling-anticipate-cyberattacks/>.
- [Heu21] Matthew Heusser. “What unit test coverage percentage should teams aim for?” In: *TechTarget* (Dec. 2021). [Online; accessed May 16, 2023].
- [Lar21a] Laravel Documentation. *Laravel Authorization*. <https://laravel.com/docs/9.x/authorization>. [Online; accessed May 12, 2023]. 2021.

- [Lar21b] Laravel Documentation. *Laravel Middleware*. <https://laravel.com/docs/9.x/middleware>. [Online; accessed May 12, 2023]. 2021.
- [Żur21] M. Żurawiecki. *Why requirements test coverage is key to success*. <https://deviniti.com/blog/application-lifecycle-management/why-requirements-test-coverage-is-key-to-success/>. [Online; accessed May 5, 2023]. June 2021.
- [Aul+22] G. S. Aulakh et al. “Project course organisational tool”. In: (2022).
- [Ind22] Indeed Editorial Team. “Differences Between Acceptance and Integration Testing”. In: (Aug. 2022). [Online; accessed May 5, 2023].
- [San22] Daniel E. Santo. *Top 5 main Agile methodologies: advantages and disadvantages*. <https://www.xpand-it.com/blog/top-5-agile-methodologies/>. [Online; accessed April 18, 2023]. Mar. 2022.
- [Som+22] M. N. Sommervold et al. “Bachelorprojeckt”. In: *JUVET* 138 (May 2022). Online; accessed January 1, 2023.
- [two22] twoday. *Hva er egentlig en Design Sprint*. <https://www.twoday.no/blogg/teknologi/hva-er-egentlig-en-design-sprint>. [Online; accessed May 9, 2023]. Dec. 2022.
- [App23] Apple. *Introduksjon til Single Sign On med Apple-enheter*. <https://support.apple.com/no-no/guide/deployment/depfdbf18f55/web>. [Online; accessed May 12, 2023]. May 2023.
- [Aul+23] G. S. Aulakh et al. *Preliminary Project Report - Project Matchmaking and Coordinating Platform*. Unpublished report. Jan. 2023.
- [Joh23] Eric Van Johnson. *Automated Testing Using PHPUnit*. [Online; accessed May 24, 2023]. Mar. 2023. URL: <https://www.phparch.com/2023/03/automated-testing-using-phpunit/>.
- [Ari22] Kazi Ariyan. *Laravel 9 - Build Complete Inventory Management System A-Z*. <https://www.udemy.com/course/laravel-build-complete-inventory-management-system/>. Online; accessed January 1, 2023. October 5, 2022.
- [Atlnd] Atlassian. *What is Agile? — Atlassian*. <https://www.atlassian.com/agile>. [Online; accessed April 18, 2023]. n.d.
- [Bernd] Sebastian Bergmann. *Welcome to PHPUnit!* <https://phpunit.de/>. [Online; accessed April 6, 2023]. n.d.

- [Blind] Blinkmarked. *Universell utforming*. <https://blinkmarked.no/universell-utforming/>. [Online; accessed May 24, 2023]. n.d.
- [Bcnd] Dave Marshall Brady Pádraic and contributors. *Mockery Docs*. <http://docs.mockery.io/en/latest/>. [Online; accessed April 6, 2023]. n.d.
- [Capnd] Capterra. *IntelliJ IDEA vs PhpStorm – Capterra*. [https://www.capterra.com/integrated-development-environment-\(ide\)-software/compare/136010-186624/IntelliJ-IDEA-vs-PhpStorm](https://www.capterra.com/integrated-development-environment-(ide)-software/compare/136010-186624/IntelliJ-IDEA-vs-PhpStorm). [Online; accessed March 24, 2023]. n.d.
- [Comnda] European Commission. *What information must be given to individuals whose data is collected?* [https://commission.europa.eu/law/law-topic/data-protection/reform/rules-business-and-organisations/principles-gdpr/what-information-must-be-given-individuals-whose-data-collected\\_en](https://commission.europa.eu/law/law-topic/data-protection/reform/rules-business-and-organisations/principles-gdpr/what-information-must-be-given-individuals-whose-data-collected_en). [Online; accessed May 5, 2023]. n.d.
- [Comndb] Zoom Video Communications. *Zoom: Video Conferencing, Web Conferencing, Webinars, Screen Sharing*. <https://zoom.us/>. Online; accessed March 24, 2023. n.d.
- [Docnd] Docker. *Docker: empowering app development for developers*. <https://www.docker.com/>. Online; accessed March 24, 2023. n.d.
- [Fignd] Figma. *Figma: The collaborative interface design tool*. <https://www.figma.com/>. Online; accessed March 24, 2023. n.d.
- [Gitnd] GitHub. *GitHub: Where the world builds software*. <https://github.com/>. Online; accessed March 24, 2023. n.d.
- [nd] JavaScript. <https://developer.mozilla.org/en-US/docs/Web/JavaScript>. Online; accessed March 24, 2023. n.d.
- [Jetnd] Jetbrains. *The Lightning-Smart PHP IDE – PhpStorm*. <https://www.jetbrains.com/phpstorm/>. [Online; accessed March 24, 2023]. n.d.
- [Kumnd] V. Kumar. *Smoke Testing vs Sanity Testing vs Regression Testing Explained*. <https://testsigma.com/blog/smoke-testing-vs-sanity-testing-vs-regression-testing-explained/>. [Online; accessed May 15, 2023]. n.d.
- [Labnd] Notion Labs. *Notion: Your wiki, docs projects. Together*. <https://www.notion.so/product>. Online; accessed March 24, 2023. n.d.

- [Larnda] Laravel. *Blade templates*. <https://laravel.com/docs/10.x/blade>. [Online; accessed March 24, 2023]. n.d.
- [Larndb] Laravel. *Eloquent Factories*. <https://laravel.com/docs/9.x/eloquent-factories#main-content>. Online; accessed May 25, 2023. n.d.
- [Larndc] Laravel. *Eloquent: Relationships*. <https://laravel.com/docs/9.x/eloquent-relationships#querying-morph-to-relationships>. [Online; accessed April 22, 2023]. n.d.
- [Larndd] Laravel. *Laravel - Database: Laravel 9.x Documentation*. [Online; accessed May 5, 2023]. n.d. URL: <https://laravel.com/docs/9.x/database>.
- [Larnde] Laravel. *Laravel Database Testing*. <https://laravel.com/docs/9.x/database-testing>. [Online; accessed May 15, 2023]. n.d.
- [Larndf] Laravel. *Laravel Sail*. <https://laravel.com/docs/9.x/sail>. Online; accessed April 16, 2023. n.d.
- [Larndg] Laravel Documentation. *Laravel Controllers Documentation*. <https://laravel.com/docs/9.x/controllers>. [Online; accessed April 22, 2023]. n.d.
- [Larndh] Laravel Documentation. *Laravel Eloquent Resources*. <https://laravel.com/docs/9.x/eloquent-resources>. [Online; accessed April 22, 2023]. n.d.
- [Larndi] Laravel Documentation. *Laravel Routing*. <https://laravel.com/docs/9.x/routing>. [Online; accessed April 22, 2023]. n.d.
- [Larndj] Larry Conklin, Victoria Drake, Sven strittmatter. *Threat Modeling Process*. [https://owasp.org/www-community/Threat\\_Modeling\\_Process](https://owasp.org/www-community/Threat_Modeling_Process). [Online; accessed May 23, 2023]. n.d.
- [Livnd] Livewire. *Laravel Livewire*. <https://laravel-livewire.com/>. [Online; accessed May 16, 2023]. n.d.
- [Matnd] Material Design. *Accessibility – Material Design 3*. <https://m3.material.io/foundations/accessible-design/overview>. [Online; accessed May 12, 2023]. n.d.
- [Metnd] Meta. *Messenger: More ways to stay in touch*. <https://www.messenger.com/features>. Online; accessed March 24, 2023. n.d.

- [Micnda] Microsoft. *Microsoft: OneDrive Personal Cloud Storage*. <https://www.microsoft.com/en/microsoft-365/onedrive/online-cloud-storage>. Online; accessed March 24, 2023. n.d.
- [Micndb] Microsoft. *Office is now Microsoft 365 - Microsoft 365*. <https://www.microsoft.com/en-us/microsoft-365?rtc=1>. [Online; accessed March 24, 2023]. n.d.
- [Orand] Oracle. *MySQL Workbench*. <https://www.mysql.com/products/workbench/>. Online; accessed March 24, 2023. n.d.
- [Otwnd] Taylor Otwell. *Laravel - The PHP Framework For Web Artisans*. <https://laravel.com/>. Online; accessed March 24, 2023. n.d.
- [OWAnd] OWASP Foundation. *OWASP Top Ten*. <https://owasp.org/www-project-top-ten/#>. [Online; accessed May 5, 2023]. n.d.
- [Par22] Linda Park. *owasp top 10 vulnerabilities*. [https://www.zscaler.com/blogs/product-insights/what-owasp-top-10?\\_bt=652979179132&\\_bk=&\\_bm=&\\_bn=g&\\_bg=146314240486&utm\\_source=google&utm\\_medium=cpc&utm\\_campaign=google-ads-na&gclid=CjwKCAjw\\_\\_ihBhADEiwAXEazJjUuY6pZcP367yGClpXB7TP5\\_Ha4svX27oEXgoTKFF2NivtkA8G](https://www.zscaler.com/blogs/product-insights/what-owasp-top-10?_bt=652979179132&_bk=&_bm=&_bn=g&_bg=146314240486&utm_source=google&utm_medium=cpc&utm_campaign=google-ads-na&gclid=CjwKCAjw__ihBhADEiwAXEazJjUuY6pZcP367yGClpXB7TP5_Ha4svX27oEXgoTKFF2NivtkA8G). [Online; accessed May 5, 2023]. March 11, 2022.
- [Patnd] A. Patil. *10 Reasons Why Laravel Is the Best PHP Framework For 2023*. Clarion Technologies. Online; accessed May 8, 2023. n.d. URL: <https://www.clariontech.com/blog/10-reasons-why-laravel-is-the-best-php-framework-for-2019>.
- [Posnd] Postman. *Postman: What is Postman?* <https://www.postman.com/>. [Online; accessed March 24, 2023]. n.d.
- [Prond] ProductPlan. *Product Specs*. <https://www.productplan.com/glossary/product-specs/>. Online; accessed April 16, 2023. n.d.
- [smand] smartsheet. *All Risk Assessment Matrix Templates You Need*. <https://www.smartsheet.com/all-risk-assessment-matrix-templates-you-need>. [Online; accessed May 23, 2023]. n.d.
- [Sofnd] SmartBear Software. *What is Swagger? (OpenAPI specification)*. <https://swagger.io/docs/specification/2-0/what-is-swagger/>. [Online; accessed April 6, 2023]. n.d.

- [SSB13] Sonia, A Singhal, and H. Banati. *Fuzzy Logic Approach for Threat Prioritization in Agile Security Framework using DREAD Model*. <https://www.semanticscholar.org/paper/Fuzzy-Logic-Approach-for-Threat-Prioritization-in-Sonia-Singhal/2dc16b8d859402a20180cc7018f7e2f9f090cebd/figure/2>. [Online; accessed April 20, 2023]. December 24, 2013.
- [Tecnd] TechTarget. *Blowfish*. <https://www.techtarget.com/searchsecurity/definition/Blowfish>. [Online; accessed May 5, 2023]. n.d.
- [W3Snd] W3Schools. *What is Bootstrap?* [https://www.w3schools.com/whatis/whatis\\_bootstrap.asp](https://www.w3schools.com/whatis/whatis_bootstrap.asp). [Online; accessed March 24, 2023]. n.d.
- [Xdend] Xdebug. *Xdebug - Step Debug*. [https://xdebug.org/docs/step\\_debug](https://xdebug.org/docs/step_debug). [Online; accessed May 16, 2023]. n.d.

## **Appendix A**

## **Appendix**

## A.1 Previous Minimum Viable Product (MVP) Version 1.0

Web 1920 – 5

The screenshot shows a web browser window with a light gray header bar. On the left, it says "Web 1920 – 5". The main content area displays the StudentHub login page for Feide. At the top, there is a logo with orange and yellow feathers and the text "STUDENTHUB" in bold, with "Our goal: ensuring your future." below it. To the right of the logo are links for "ABOUT STUDENTHUB" and "LOGG INN". The main form is titled "Logg inn med Feide". It includes fields for "Brukernavn" (username) and "Passord" (password), both with placeholder text "Glemt brukernavn eller passord?". Below these is a blue "Logg inn" button. Underneath the password field is a link "Eller velg en ekstern metode:" (Or choose an external method:). It lists two options: "Bruk arbeids- eller skolekonto" (use work or school account) and "Logg inn med ID-porten" (log in with ID-porten). At the bottom of the form, there is a link "Trenger du hjelp?" (Need help?) followed by a plus sign (+) and a link "Personvern og informasjonskapsler" (Privacy and information notices) at the very bottom.



POSITION TYPE

FEILD OF EMPLOYMENT

 Information Technology

TITLE \*

COMPANY NAME \*

STATE/COUNTY \*

CITY

COUNTRY \*

#### TAGS

TAGS \* (Maximum 5)

#### POSITION AVAILABLE

ABOUT THE COMPANY \*

OFFER DESCRIPTION \*

WHAT WE OFFER \*

AVAILABLE PLACES \*

## ATTACHMENT

### ATTACHMENT

Max 5 Mb

### LOGO \*

Add your company logo (jpeg/gif/png) here. Recommended size between 300x150 and 800x400px. Best result with logo in 2:1.

## QUALIFICATIONS

### QUALIFICATIONS NEEDED \*

## HOW TO APPLY

### APPLICATION EMAIL \*

### APPLICATION DESCRIPTION \*

### DEADLINE \*

## CONTACT PERSON

### CONTACT PERSON \*

### CONTACT PERSON EMAIL \*

### CONTACT PERSON PHONE \*



## Forgot Password

*NB! You will receive an email with a temporary password. That password will be available for 24 hours.  
By clicking on the link, a new page will appear where you can create a new password.*

Enter your username and a new password will be sent to your email address

Username :

**SEND**

### What is username?

Students use their assigned student number when logging in.

Companies use e-mail when logging in.

Questions about the service? Contact [contactservice@studenthub.no](mailto:contactservice@studenthub.no) or phone 123 45 678



## Register new company user

Questions about the service? Contact [contactservice@studenthub.no](mailto:contactservice@studenthub.no) or phone 123 45 678

COMPANY NAME \*

ORGANISATION NUMBER \*

Please add your Organisations reg. number here Search in [Bronnøysundregisteret](#)

COUNTRY \*

 Norway ▼

Contact person

FIRST NAME \*

LAST NAME \*

EMAIL \*

OFFICE PHONE \*

### Disclaimer conditions

I accept the disclaimer conditions.

Send

Cancel

Web 1920 – 8



**STUDENTHUB**  
Our goal: ensuring your future.

ABOUT STUDENTHUB   LOGG INN

### STUDENTHUB - LOGG INN

**Student login**

**Administration login**

— OR —

**Company login**

[New password?](#)

Companies can fill in the [registration](#) form for access.

[Do you need help?](#)

Web 1920 – 9

The screenshot shows the STUDENTHUB platform. At the top, there is a logo with a stylized orange bird and the text "STUDENTHUB" with the tagline "Our goals exceed your future." Below the logo, there are links for "ABOUT STUDENTHUB" and "LOGG UT". On the left side, there is a sidebar with a user profile picture of Tulipesh Patel, a "Review Applications" section, a "Confirmed Applications" section, a "Student Overview" section, and a yellow "OsloMet" button. The main content area has a title "Practical IT-project" and a message "22 projects left to review:". There are three cards, each describing a job posting for front-end developers and providing "Approve" and "Dismiss" buttons.

We are looking for front-end Developers with skills in CSS, HTML and React	<input checked="" type="button"/> Approve
In Our Company, FinterAI, we are looking for people that can work on the front-end front. You will be asked to make the front-page for Our Company, which will involve CSS3, HTML5, and React.js.	<input type="button"/> Dismiss

We are looking for front-end Developers with skills in CSS, HTML and React	<input checked="" type="button"/> Approve
In Our Company, FinterAI, we are looking for people that can work on the front-end front. You will be asked to make the front-page for Our Company, which will involve CSS3, HTML5, and React.js.	<input type="button"/> Dismiss

We are looking for front-end Developers with skills in CSS, HTML and React	<input checked="" type="button"/> Approve
In Our Company, FinterAI, we are looking for people that can work on the front-end front. You will be asked to make the front-page for Our Company, which will involve CSS3, HTML5, and React.js.	<input type="button"/> Dismiss

The screenshot shows the "Members" page of the STUDENTHUB platform. On the left, there is a sidebar with a user profile picture of Tulipesh Patel, a "Log out" button, a "Published Internships" section, an "Unpublished Internships" section, a "Members" section, a "View Profile" section, and a yellow "OsloMet" button. The main content area has a title "Members" and a filter section "Filter by member type: Admin, Organization, Student". Below the filter is a table listing members:

Username	Role	Status
Tulipesh Patel ✖	Admin	
Per Hansen ✖	Student	Has Not Applied Yet
Line Jensen ✖	Student	Has Applied
Oslomet ✖	Organization	

**Practical IT-project**

1 internships are not published:

**Customer Support Associate - Part Time**

We believe in a world where all cars are shared. Car sharing enables people to get around in a smarter and easier way, while having a positive impact on the environment and making cities more livable. It is this vision that drives us forward and inspires us to think even bigger. Since November 2021, Nabobil has been part of Getaround. Together we are the world's leading car sharing platform with a community of more than 5 million users who share over 11,000 connected cars in 7 countries. Our team is collaborative, positive, curious and committed. We think fast, work smart, laugh often and are looking for like-minded people to join us in our mission to disrupt car ownership and make cities better.

**CSS (HTML)**  
Co.: OsloMet

0 have applied

[Publish this internship](#) [Decline this internship](#)

**Tulipesh Patel**  
Change profile info  
[Log out](#)

Published Internships  
Unpublished Internships  
Members  
[View Profile](#)

**OsloMet**

**Available internships**

Filter by areas of work:  
 PHP    Java    CSS    HTML    Python    C#    JavaScript    React    NodeJS    Deno    MongoDB    MySQL

1 internships are published:

**Customer Support Associate - Part Time**

We believe in a world where all cars are shared. Car sharing enables people to get around in a smarter and easier way, while having a positive impact on the environment and making cities more livable. It is this vision that drives us forward and inspires us to think even bigger. Since November 2021, Nabobil has been part of Getaround. Together we are the world's leading car sharing platform with a community of more than 5 million users who share over 11,000 connected cars in 7 countries. Our team is collaborative, positive, curious and committed. We think fast, work smart, laugh often and are looking for like-minded people to join us in our mission to disrupt car ownership and make cities better.

**CSS (HTML)**  
Co.: OsloMet

2 have applied

Students interested in this internship:	
Per Hansen	Applied
Line Jensen	Invited

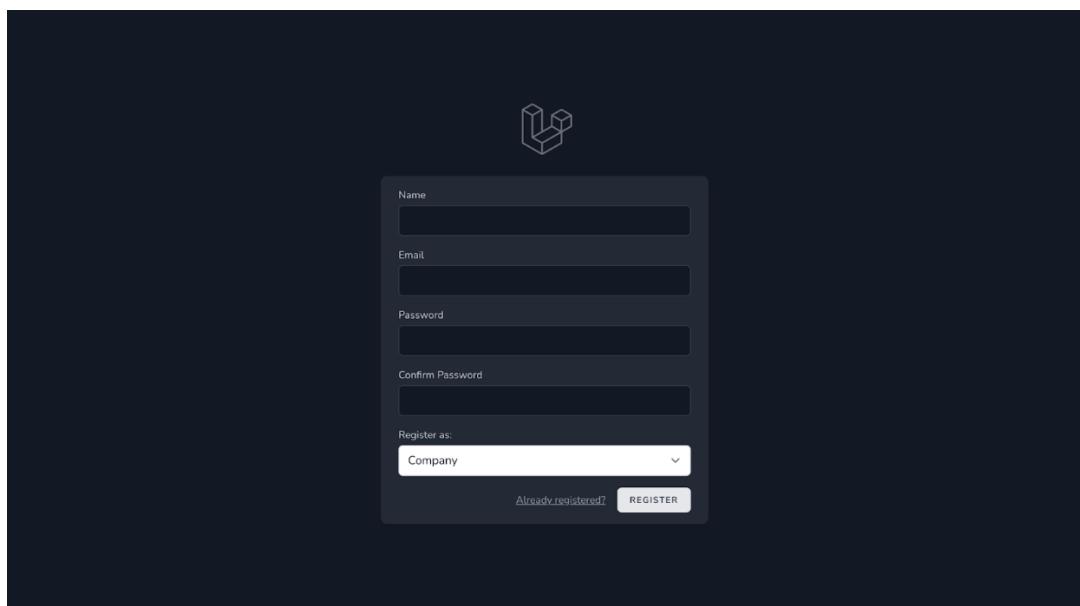
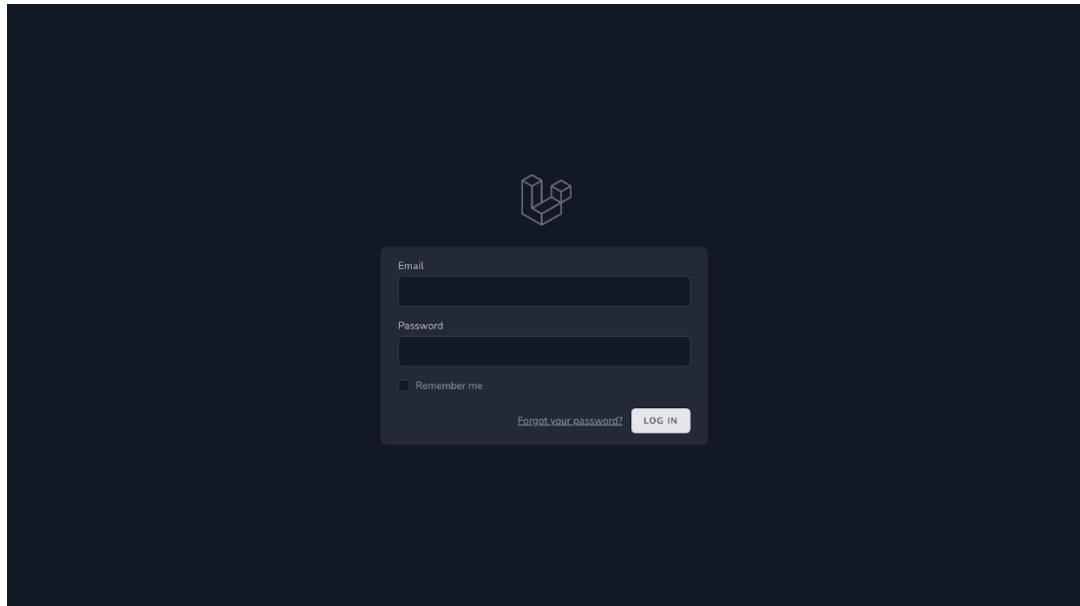
**Tulipesh Patel**  
Change profile info  
[Log out](#)

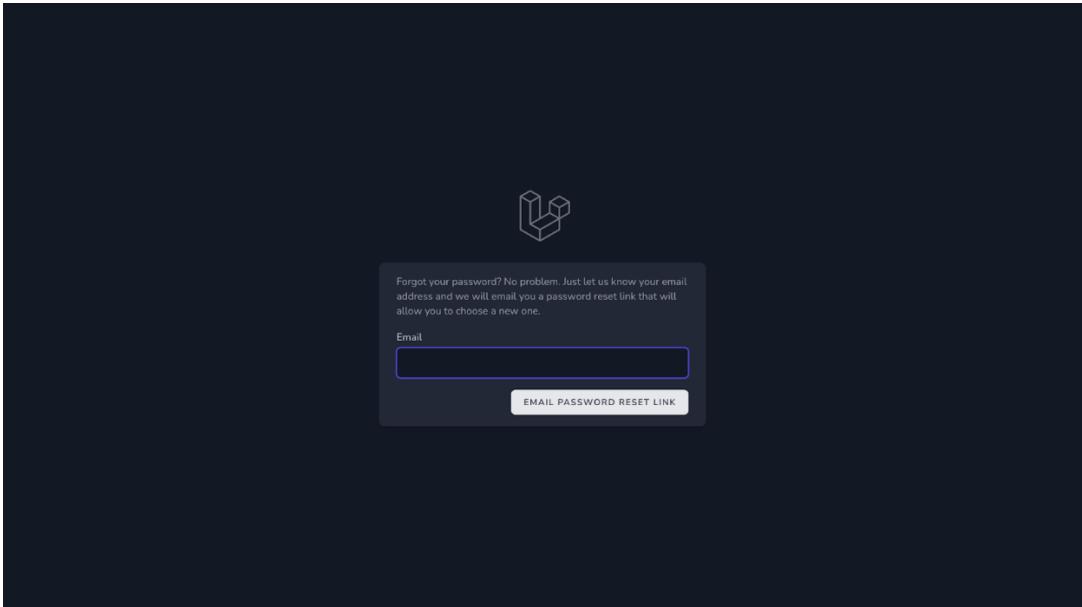
Published Internships  
Unpublished Internships  
Members  
[View Profile](#)

**OsloMet**

## A.2 Final version of the prototype

Prototype - Authentication





## Prototype - Admin

A screenshot of an admin dashboard prototype. The left sidebar has a "MENU" section with "Dashboard", "Projects" (selected), "All Projects", "Add Project", and "Status Categories". The main area shows a "projects" section with two project cards. Each card has a title, a description starting with "Lorem ipsum is simply dummy text of the printing and typesetting industry.", a timestamp ("02.02.23 12:00"), a status indicator ("Maestros"), and a unique ID ("DATA3710"). To the right of the projects is a "Project Name" form with a "Description" section containing placeholder text. Below this is an "About company" section with the same placeholder text. A "Contact information" section is partially visible. At the bottom, there's an "Approved projects" section showing one item: "Junior php utvikler for web app - 1" with a green checkmark icon, a timestamp ("02.02.23 12:00"), a status indicator ("sopra steria"), and a note ("Antall studenter ønsket: 5"). To the right of this are buttons for "Approve", "Reject", "Need Modification", "Edit", and "Submit".

The screenshot shows a web-based project management application. On the left, a sidebar menu includes 'Dashboard', 'Projects' (which is selected), 'All Projects', 'Add Project', and 'Status Categories'. The main area displays a list of projects under the heading 'projects'. Each project card contains a title, a brief description, a timestamp ('02.02.23 12:00'), a team name ('Maestros'), and a unique identifier ('DATA3710'). A blue arrow icon is on the right of each card. Below this is a section titled 'Approved projects' containing three cards with titles like 'Junior php utvikler for web app - 1', descriptions, and green checkmark icons. To the right of these cards is a panel for managing approvals: 'Project Name' (placeholder), 'Description' (placeholder), 'About company' (placeholder), 'Contact information' (placeholder), and buttons for 'Approve' (blue), 'Reject' (blue), 'Need Modification' (blue), 'Edit' (blue), and 'Submit' (green).

This screenshot shows the same application interface, but the 'Approved projects' section is more prominent. It lists three approved projects with green checkmark icons. To the right of the projects is a panel with two checkboxes: 'select all projects' and 'Publish Project Descriptions for students'. At the bottom right of this panel is a large green button labeled 'Lagre/send'.

The screenshot shows a web application interface for managing project status categories. The left sidebar has a 'MENU' section with 'Dashboard', 'Projects' (expanded), 'Status Categories' (selected), 'All Status Categories' (highlighted in blue), and 'Create New Status'. The main content area is titled 'ALL STATUS CATEGORIES' and shows a table with one entry:

ID	Status	Action
1	NEW	

Below the table, it says 'Showing 1 to 1 of 1 entries'. The top right shows 'Project Status > All Project Status' and a search bar. The bottom right corner says 'Crafted with ❤ by Future Solutions'.

The screenshot shows a 'Create New Status' form. The left sidebar is identical to the previous screenshot. The main content area is titled 'Create New Status' and contains a 'Status Name' input field and a 'Insert New Status' button.

The screenshot shows a web application interface with a dark header bar containing a search bar and user information. A sidebar on the left lists 'Dashboard' and 'Projects'. The main area displays two project cards:

- Junior php utvikler for web app - 1**  
Lorem Ipsum is simply dummy text of the printing and typesetting industry.  
Lorem Ipsum has been the industr....  
xxxxxx Antall studenter 5 Endre grupper
- Junior php utvikler for web app - 2**  
Lorem Ipsum is simply dummy text of the printing and typesetting industry.  
Lorem Ipsum has been the industr....  
sopra steria Antall studenter : 4 Endre grupper

On the right, there is a section titled 'Junior php utvikler for web app - 1' with a list of students and a green checkmark icon. At the bottom are two buttons: 'make description' and 'Lagre/send'.

The screenshot shows a web application interface with a dark header bar containing a search bar and user information. A sidebar on the left lists 'Dashboard' and 'Projects'. The main area displays a table of student status checkboxes:

students:	taushetsplikt:	samarbeidsavtalet:	
<input type="checkbox"/> s123445	<input type="checkbox"/>	<input type="checkbox"/>	All students <input checked="" type="checkbox"/>
<input type="checkbox"/> s446688	<input type="checkbox"/>	<input type="checkbox"/>	signert avtale
<input type="checkbox"/> s995411	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	/like signert
<input type="checkbox"/> s774466	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
<input type="checkbox"/> s119922	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> s119922	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
<input type="checkbox"/> s119922	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> s119922	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> s119922	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
<input type="checkbox"/> s119922	<input type="checkbox"/>	<input checked="" type="checkbox"/>	

At the bottom is a 'send/mail' button.

## Prototype - Company

The screenshot shows a web application interface for a company to submit a new project. The top navigation bar includes a search bar and a user profile icon labeled "company". On the left, a sidebar menu lists "Dashboard", "Projects" (which is selected), "All Projects", and "Add Project". The main content area is titled "projects" and contains a "New project" section. This section includes fields for "Project Name" and "Description", with placeholder text about dummy printing and typesetting industry text. Below this, there's an "About company" section with placeholder text about the company's history. A "Contact information" section lists an email address (soprasteria@gmail.com) and a phone number (123456778). At the bottom right of the main form is a green "send request" button.

The screenshot shows a web application interface displaying a list of projects. The top navigation bar includes a search bar and a user profile icon labeled "company". On the left, a sidebar menu lists "Dashboard", "Projects" (selected), "All Projects", and "Add Project". The main content area is titled "projects" and contains a "My projects" section. It lists three projects: "Junior php utvikler for web app - 1", "Junior php utvikler for web app - 2", and "Junior php utvikler for web app - 3". Each project card includes a description of the job requirements (e.g., "Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industr....") and a status indicator (a green checkmark for the second project). To the right of the project cards is a sidebar titled "projects status" with options: "accepted", "not projects", and "All projects".

The screenshot shows a web application interface. On the left, a sidebar menu includes 'Dashboard', 'Projects' (which is selected), 'All Projects', and 'Add Project'. The main content area is titled 'projects' and lists three items:

- Junior php utvikler for web app - 1**  
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industr....  
sopra steria    Antall studenter ønsket: 4
- Junior php utvikler for web app - 2**  
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industr....  
xxxxxxxxxxxxx    Antall studenter ønsket: 5
- Junior php utvikler for web app - 3**  
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industr....  
xxxxxxxxxx .    Antall studenter ønsket: 3

To the right, a sidebar for 'company' details includes fields for 'Project Name' and 'Description', both containing placeholder text. It also shows 'About company' information and contact details (mail: soprasteria@gmail.com, tel: 12345677). A green button at the bottom right says 'Edit project proposals'.

This screenshot shows the same application after publishing the projects. The sidebar menu remains the same. The main content area is titled 'published' and lists the same three projects as before, each with a blue arrow icon to its right.

For the first project, 'Junior php utvikler for web app - 1', there is a list of students who have applied (søkt):

- s123445
- s446688
- s995411
- s774466
- s119922
- s119922
- s119922
- s119922
- s119922

Below this list are two buttons: 'accept student' and 'reject student', each with a blue square icon. A green 'send' button is located at the bottom right of the sidebar.

MENU

- Dashboard
- Projects
  - All Projects
  - Add Project

Search...

company

## projects

### publeshed

**Junior php utvikler for web app - 1**

Lorem ipsum is simply dummy text of the printing and typesetting industry.  
Lorem ipsum has been the industr....

xxxxxxxxx Antall studenter 5

**Junior php utvikler for web app - 2**

Lorem ipsum is simply dummy text of the printing and typesetting industry.  
Lorem ipsum has been the industr....

xxxxxxxxx Antall studenter : 4

MENU

- Dashboard
- Projects
  - 3

Search...

company

## DASHBOARD

**Junior php utvikler for web app - 2**

Lorem ipsum is simply dummy text of the printing and typesetting industry.  
Lorem ipsum has been the industr....

xxxxxxxxx Antall studenter ønsket: 3

Contact information

Admin:  
mail: Admin@gmail.com  
tlf: 123456789

oslomet:  
mail: sentral@oslomet.no  
tlf: 67 23 50 00

## Prototype - Student

The screenshot shows a web application interface for a student. The top navigation bar includes a search bar and user icons. On the left, a sidebar menu lists 'Dashboard' and 'Projects'. Under 'Projects', there is a dropdown menu with 'All projects' selected. The main content area displays three project cards:

- Junior php utvikler for web app - 1**  
Lorem Ipsum is simply dummy text of the printing and typesetting industry.  
Lorem Ipsum has been the industr....  
xxxxxxxxxx . Antall studenter ønsket: 6
- Junior php utvikler for web app - 2**  
Lorem Ipsum is simply dummy text of the printing and typesetting industry.  
Lorem Ipsum has been the industr....  
xxxxxxxxxx . Antall studenter ønsket: 3
- Junior php utvikler for web app - 3**  
Lorem Ipsum is simply dummy text of the printing and typesetting industry.  
Lorem Ipsum has been the industr....  
xxxxxxxxxx . Antall studenter ønsket: 2

A modal window titled 'All projects' is open on the right, showing a list of 'All projects' and 'My projects'.

The screenshot shows a detailed view of a project card for 'Junior php utvikler for web app - 1'. The card includes the project title, a brief description, and a note about the number of students interested. To the right of the card, there is a sidebar with project details and a 'send request' button.

**Project Name**  
**Description**  
Lorem Ipsum is simply dummy text of the printing and typesetting industry.  
Lorem Ipsum has been the industr....

**About company**  
Lorem Ipsum is simply dummy text of the printing and typesetting industry.  
Lorem Ipsum has been the industr....

Contact information:  
mail: soprasteria@gmail.com  
tilf: 12345677

**send request**

The screenshot shows a web application interface. At the top, there is a dark header bar with a search bar containing "Search...". On the right side of the header, there are icons for notifications and user authentication, with the text "student" and a dropdown arrow.

The main content area has a left sidebar labeled "MENU" with options: "Dashboard", "Projects" (which is selected and highlighted in blue), and "All Projects".

The main panel is titled "projects" and contains a section titled "My projects". It displays a card for a project named "Junior php utvikler for web app - 1". The card includes a description: "Lorem ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industr....", and two small status indicators: "sopra steria" and "Antall studenter ønsket: 5". To the right of the card is a blue right-pointing arrow icon.

To the right of the project card, there is a list titled "Junior php utvikler for web app - 1" under the heading "students:". The list contains five entries, each with a small blue square icon and a student ID: "s123445", "s446688", "s995411", "s774466", and "s119922".

At the bottom right of the main panel, there is a green button labeled "contact group".

This screenshot shows a different view of the same web application. The header and sidebar are identical to the previous screenshot.

The main panel is titled "DASHBOARD" and contains a section titled "my projects". It displays the same project card for "Junior php utvikler for web app - 1" with its description, status indicators, and blue right-pointing arrow icon.

**ALL PROJECTS**

ID	Title	Company	Status	Tags
1	Necessitatibus provident sit laborum omnis quia ut et ut.		NEW	laravel, api, backend
2	Dolorum sit dicta omnis voluptas sed iure molestiae.		NEW	laravel, api, backend
3	Et et ipsum porro aut quidem laborum.		NEW	laravel, api, backend

Showing 1 to 3 of 3 entries

**Project Page**

0 WORDS POWERED BY TINY

## A.3 User Test - Template from IT project in practice

### User Test - Template

#### Companies' overview

##### **Execution:**

1. Create an internship.

##### **Understand use-case:**

2. How do you know you created an internship?
3. Is the internship published to students yet?

##### **Execution :**

4. Look through students and invite one of them to an internship.

#### Course Coordinators' overview

##### **Execution:**

1. How can you publish internships to students? Publish one.

##### **Understand use-case:**

2. Verify that the internship got published.

#### Students' overview

##### **Understand use-case :**

1. Where do you find internships that you can apply to?

##### **Execution :**

2. Apply for an internship.
3. Either accept or decline an invitation for a job.
4. Change your profile information.

##### **Understand use-case :**

1. How do you find out if you've been accepted for an internship?

##### **Execution :**

2. Roleplay “Let's say, you changed your mind or made a mistake where you sent an application for a job you do not want. How do you unsend the application?”

## A.4 User Test - Template

### User Test – Template

**Use Case 1:** Project proposal is accepted and published

Companies' overview

**Execution:**

1. Create a project proposal.

**Understand use-case:**

- How do you know that your project proposal was created successfully?
- How do you check the status of your project proposal?
- How do you check if your project published for the students?

Admin's overview

**Execution:**

1. Find project proposals
2. Accept and publish a project

**Understand use-case:**

- How can you verify that the project is published

**Post requisite (to be implemented):**

- Company user gets email from the application that their project has been accepted.

Students' overview

**Execution:**

1. Find all projects available
2. Favorite a project

**Understand use-case:**

- Find the favorited project.

**Use Case 2:** Project proposal is not accepted by admin and changes to the project proposal is made

**Prerequisite:**

A project proposal is made.

[Admin's overview](#)

**Execution:**

1. Find project proposals
2. Request changes to a project

**Understand use-case:**

- How can you make sure that the project change has been requested

[Companies' overview](#)

**Execution:**

1. Find your project proposal that have got
2. Request changes to a project

**Use Case 3:** Admin creates a project and publishes it

[Admin's overview](#)

**Execution:**

1. Create a company (only for first time)
2. Create a project proposal
3. Publish your project

**Understand use-case:**

- Double check that the project is published

## A.5 Interview guide

### Introduction

- Introduction of the interviewing team
- Outline of basic information about the project and the aims of the user-test and interview
- Confirmation that the participant understands what they have consented to, that their information and data will be stored securely and that they can withdraw their participation at any time.

### Interview guide

#### Warm-up questions |

1. Can you tell me a bit about your background and experience related to the platform that you just used?
2. Have you used similar products or services before? If so, could you briefly describe your experience with them?

#### Task related questions

1. What are you thinking as you view the navigation bar?
2. If you were looking for [information], where would you expect to find it?
3. How was the experience of
  - hiding the navigation bar?
  - editing your phone number
  - listing all the projects offered by [company X]
  - filtering out two of the projects
  - finding the latest available projects
  - adding a project to your favorites
  - applying for a project you found interesting.

#### Probe questions

- Tell me more... How so?
- What is this? What is this for?
- What did you notice? What do you think of that?
- What would you expect that to do?

#### General impressions

1. How would you describe your overall experience with the platform?
2. What did you like the most about using the platform?
3. What did you like the least?
4. What, if anything, surprised you about the experience?
5. What, if anything, caused your frustration?

## A.6 User Guidelines

To help ensure a positive user experience, please read, and follow these guidelines:

**1. Create a Strong Password**

When creating an account, using a strong password that combines a minimum of eight letters, at least one number, an uppercase character, and a unique character is essential. The use of such a password will help establish a secure account.

**2. Keep your account information up to date**

It is important to keep account information such as email address and password up to date to receive important notifications and maintain access to the account.

**3. Protect your privacy**

One should be mindful of their privacy and refrain from sharing sensitive data such as email addresses or passwords with anyone.

**4. Report inappropriate behavior**

Encountering inappropriate behavior while using the application should be reported immediately to one of the staff members. Doing this will create a safe and welcoming space for all users.

**5. Use the application responsibly**

One should use this application responsibly and comply with all relevant laws and regulations. This application has a specific purpose and using it for illegal or malicious activities is unacceptable.

**6. Contact support**

If any questions or concerns arise regarding the application, contacting the application support team for assistance is recommended. The support team aims to provide a positive experience for application users. Please follow the provided guidelines to ensure a safe and enjoyable experience using this application.

Thank you for choosing to use the Student Match-Making application.

**Safety and Protection of Data**

The data provided through the application will be stored safely at OsloMet's servers. The security and privacy of the provided information are taken thoughtfully, and reasonable safety measures have been implemented to safeguard this information from unauthorized access and disclosure. It is crucial to note that no Internet or electronic storage transmission method is 100% secure. As such, the absolute security of your information cannot be guaranteed. Recognizing and accepting these inherent risks are necessary when utilizing our web application. It is agreed that any damages resulting from a security breach will not be held against us.

**Termination clause**

Either party may terminate these terms and conditions at any time, for any reason, by providing written notice to the other party. It will be possible if any user wants to delete their account permanently. All the data that OsloMet's database has collected from that person will be deleted.

**Disclaimer for third-party applications/services**

We are using third-party services to assist us in operating our web application. OsloMet's servers have access to the information you provided to us to store the data in their database. They are obliged not to disclose or use this data for any purpose other than storage. We do not warrant, guarantee, or assume responsibility for the reliability or legality of any third-party service. We disclaim all liability for any harm or damages from data storage at OsloMet's servers.

## A.7 Notification form for the processing of personal data

5/8/23, 1:51 PM Meldeskjema for behandling av personopplysninger

 **Sikt**

[Meldeskjema](#) / [Student internship match-making platform](#) / Eksport

**Meldeskjema**

**Referansenummer**  
729026

**Hvilke personopplysninger skal du behandle?**

- Navn (også ved signatur/samtykke)
- Adresse eller telefonnummer
- E-postadresse, IP-adresse eller annen nettidentifikator
- Lydoppak av personer

**Prosjektinformasjon**

**Prosjekttittel**  
Student internship match-making platform

**Prosjektbeskrivelse**  
A bachelor project where students will be developing a prototype platform that will facilitate the management of student-company internships as part of a portfolio of project-based bachelor courses. User-tests of the prototype platform will be conducted as part of the development process.

**Begrunn hvorfor det er nødvendig å behandle personopplysningene**  
Some basic demographic data (including age, gender, professional or educational background) would be useful when analysing user test data (though user testing can be performed anonymously).

**Prosjektbeskrivelse**  
[Prosjektskisse-BachelorOppgave.docx](#)

**Ekstern finansiering**  
Ikke utfyllt

**Type prosjekt**  
Studentprosjekt, bachelorstudium

**Gjelder innmeldingen for flere studentprosjekter (felles vurdering)?**  
Ja

**Oppgi antall studenter**  
5

**Behandlingsansvar**

**Behandlingsansvarlig institusjon**  
OsloMet – storbyuniversitetet / Fakultet for teknologi, kunst og design / Institutt for informasjonsteknologi

**Prosjektansvarlig (vitenskapelig ansatt/veileder eller stipendiat)**  
Tulipesh Patel, tulipesh@oslomet.no, tlf: 4796045890

**Skal behandlingsansvaret deles med andre institusjoner (felles behandlingsansvarlige)?**  
Nei

**Utvalg 1**

**Beskriv utvalget**

<https://meldeskjema.sikt.no/63e0c944-50c2-4c46-bde4-f969bf78cfb/eksport>

Bachelor students at OsloMet

**Beskriv hvordan rekruttering eller trekking av utvalget skjer**

The group will recruit students from their own network (fellow co-students and acquaintances)

**Alder**

18 - 50

**Personopplysninger for utvalg 1**

**Hvordan samler du inn data fra utvalg 1?**

**Personlig intervju**

**Vedlegg**

[Interview guide - Student internship match-making project.docx](#)

**Grunnlag for å behandle alminnelige kategorier av personopplysninger**

Samtykke (Personvernforordningen art. 6 nr. 1 bokstav a)

**Informasjon for utvalg 1**

**Informerer du utvalget om behandlingen av personopplysningene?**

Ja

**Hvordan?**

Skriftlig informasjon (papir eller elektronisk)

**Informasjonsskriv**

[INFORM-2.DOC](#)

---

**Tredjepersoner**

**Skal du behandle personopplysninger om tredjepersoner?**

Nei

---

**Dokumentasjon**

**Hvordan dokumenteres samtykkene?**

- Elektronisk (e-post, e-skjema, digital signatur)

**Hvordan kan samtykket trekkes tilbake?**

Participants can send an email or give verbal withdrawal at any time.

**Hvordan kan de registrerte få innsyn, rettet eller slettet personopplysninger om seg selv?**

Participants can send an email or verbally request access to delete or edit the information. Information in the platforms database that the students and provide can be editing without request.

**Totalt antall registrerte i prosjektet**

100-999

---

**Tillatelser**

**Skal du innhente følgende godkjenninger eller tillatelser for prosjektet?**

Ikke utfyllt

---

**Behandling**

**Hvor behandles personopplysningene?**

- Maskinvare tilhørende behandlingsansvarlig institusjon

**Hvem behandler/har tilgang til personopplysningene?**

- Prosjektsvarlig
- Student (studentprosjekt)

**Tilgjengeliggjøres personopplysningene utenfor EU/EØS til en tredjestaat eller internasjonal organisasjon?**

Nei

**Sikkerhet****Oppbevares personopplysningene atskilt fra øvrige data (koblingsnøkkelen)?**

Ja

**Hvilke tekniske og fysiske tiltak sikrer personopplysningene?**

- Flerfaktoraутentisering
- Adgangsbegrensning
- Adgangslogg
- Personopplysningene anonymiseres fortløpende

**Varighet****Prosjektpериode**

01.02.2023 - 01.08.2023

**Hva skjer med dataene ved prosjektslutt?**

Data med personopplysninger oppbevares midlertidig til: 01.08.2023

**Hva er formålet med den videre oppbevaringen av dataene?**

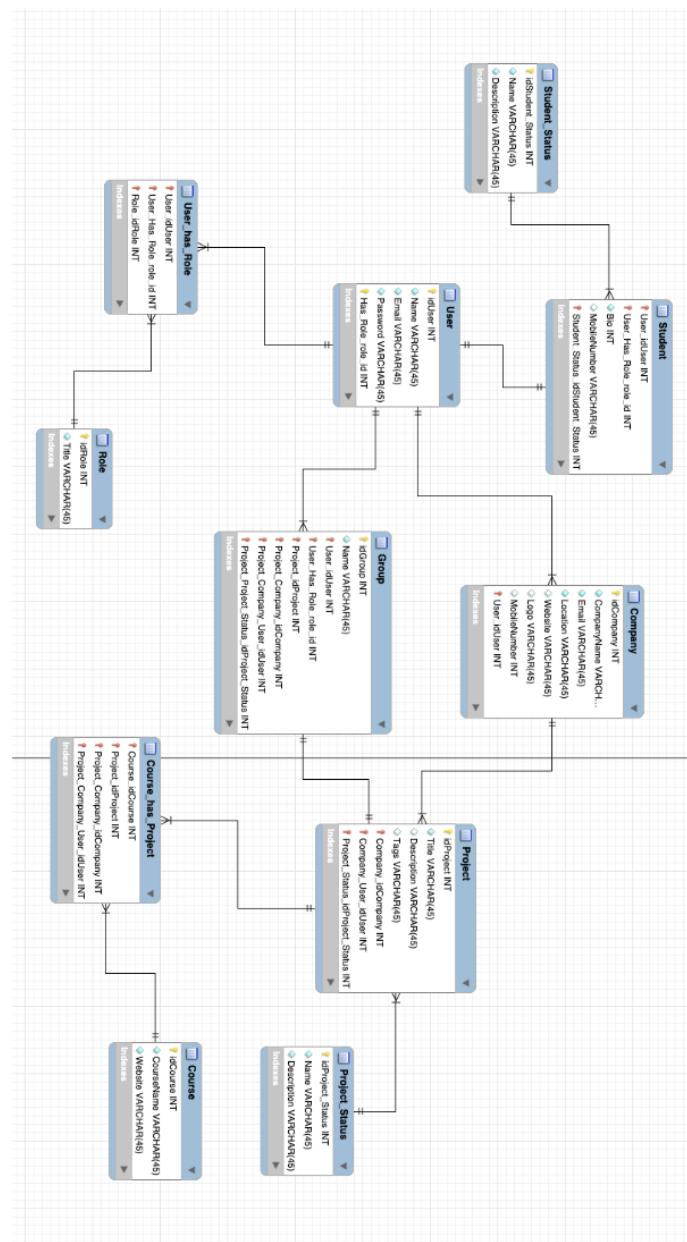
Langtidslagring og/eller arkivering for deling av data

**Vil de registrerte kunne identifiseres (direkte eller indirekte) i oppgave/avhandling/øvrige publikasjoner fra prosjektet?**

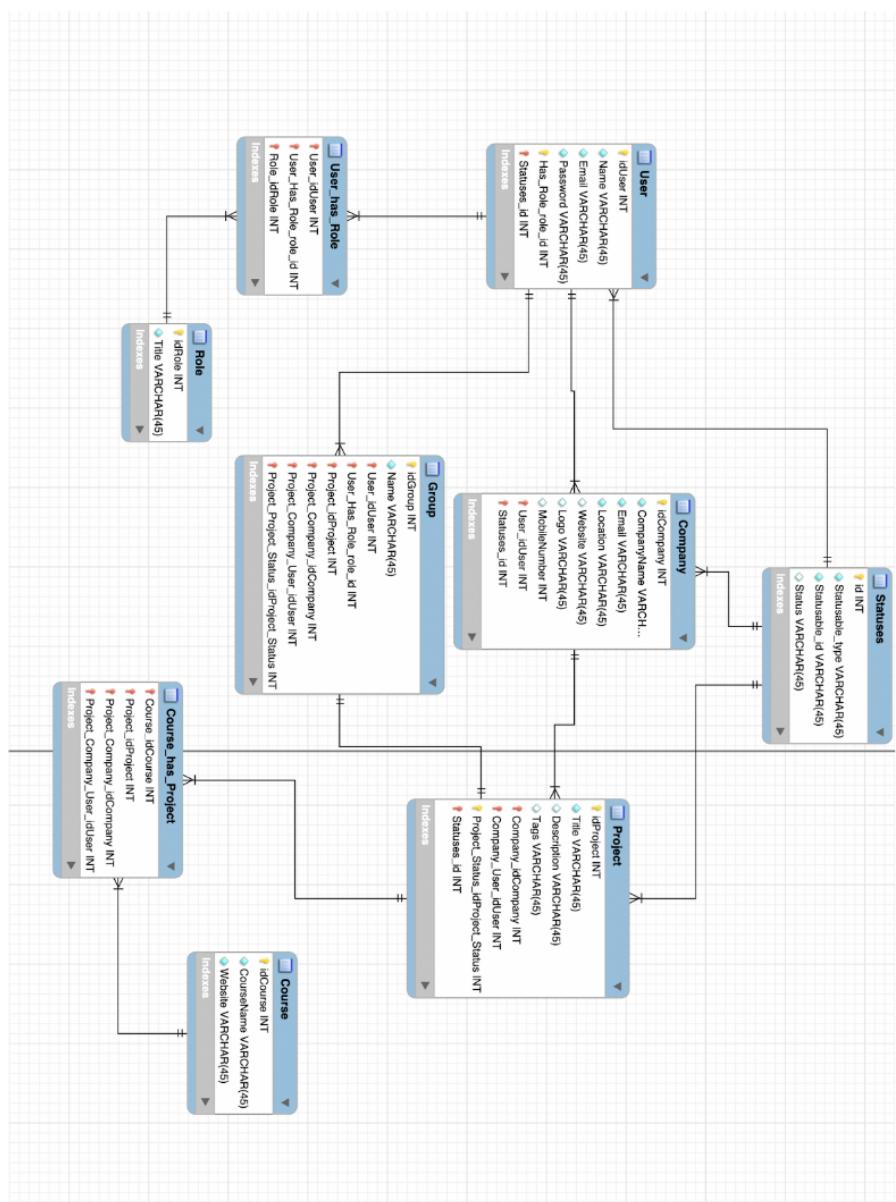
Nei

**Tilleggsopplysninger**

## A.8 First version of the database



## A.9 Second version of the database



## A.10 Hosting throw OsloMet VM

```
[root@bachelor-v2023:/home/ubuntu# ls
docker-compose.yml  docker.sh  vendor
[root@bachelor-v2023:/home/ubuntu# cd vendor/
[root@bachelor-v2023:/home/ubuntu/vendor# ls -l
total 232
```

---

```
Step 1/19 : FROM ubuntu:22.04
22.04: Pulling from library/ubuntu
2ab09b027e7f: Pull complete
Digest: sha256:67211c14fa74f070d27cc59d69a7fa9aeff8e28ea118ef3babc295a0428a6d21
Status: Downloaded newer image for ubuntu:22.04
---> 08d22c0ceb15
Step 2/19 : LABEL maintainer="Taylor Otwell"
---> Running in 604ba7eda3fb
Removing intermediate container 604ba7eda3fb
---> 1a9d1721d8de
Step 3/19 : ARG WWWGROUP
---> Running in e88ceaa0eb8c
Removing intermediate container e88ceaa0eb8c
---> c83b0bf85e59
Step 4/19 : ARG NODE_VERSION=18
---> Running in b87d677b6371
Removing intermediate container b87d677b6371
---> da0592a2662f
Step 5/19 : ARG POSTGRES_VERSION=14
---> Running in b5b9118b702d
Removing intermediate container b5b9118b702d
---> 21c4a0a91edb
Step 6/19 : WORKDIR /var/www/html
---> Running in ee550c5a4869
Removing intermediate container ee550c5a4869
---> 828c08a08368
Step 7/19 : ENV DEBIAN_FRONTEND noninteractive
---> Running in a9924136df1
Removing intermediate container aa9924136df1
---> 233b17174b9a
Step 8/19 : ENV TZ=UTC
---> Running in d2030058affd
Removing intermediate container d2030058affd
---> 7db34adce1d8
Step 9/19 : RUN ln -snf /usr/share/zoneinfo/$TZ /etc/localtime && echo $TZ > /etc/timezone
---> Running in 93fcfd9bcd871
Removing intermediate container 93fcfd9bcd871
---> 249209c3ac68
```

---

```
[root@bachelor-v2023:/home/ubuntu# docker-compose up
WARNING: The WWWGROUP variable is not set. Defaulting to a blank string.
WARNING: The WWWUSER variable is not set. Defaulting to a blank string.
Creating network "ubuntu_sail" with driver "bridge"
Creating volume "ubuntu_sail-mysql" with local driver
Creating volume "ubuntu_sail-redis" with local driver
Creating volume "ubuntu_sail-meilisearch" with local driver
Pulling mysql (mysql/mysql-server:8.0)...
8.0: Pulling from mysql/mysql-server
6a4a3ef82cdc: Pull complete
5518b09b1089: Pull complete
b6b576315b62: Pull complete
349b52643cc3: Pull complete
abe8d2406c31: Pull complete
c7668948e14a: Pull complete
c7e93886e496: Pull complete
Digest: sha256:d6c8301b7834c5b9c2b733b10b7e630f441af7bc917c74dba379f24eeeb6a313
Status: Downloaded newer image for mysql/mysql-server:8.0
Pulling redis (redis:alpine)...
alpine: Pulling from library/redis
f56be85fc22e: Pull complete
10889393d00d: Pull complete
4bfd915173f8: Pull complete
4902c1f811fb: Pull complete
d284f724869a: Pull complete
c1ac59df4c6d: Pull complete
Digest: sha256:da0cc759968a286f9fa8e3a0d8faca70e4dcf8ffc25fd290a041c59a9eb725c7
Status: Downloaded newer image for redis:alpine
Pulling mailhog (mailhog/mailhog:latest)...
latest: Pulling from mailhog/mailhog
df20fa9351a1: Pull complete
ed8968b2872e: Pull complete
a92cc7c5fd73: Pull complete
f17c8f1adafb: Pull complete
03954754c53a: Pull complete
60493946972a: Pull complete
368ee3bc1dbb: Pull complete
Digest: sha256:8d76a3d4ffa32a3661311944007a415332c4bb855657f4f6c57996405c009bea
Status: Downloaded newer image for mailhog/mailhog:latest
Pulling selenium (selenium/standalone-chrome:)... .
latest: Pulling from selenium/standalone-chrome
47c764472391: Pull complete
13aa5d2dc1c2: Pull complete
d98249c08033: Pull complete
606d4140028b: Pull complete
e63617ec364c: Pull complete
b36fe5097318: Pull complete
9cf4c0ef4fb: Pull complete
7d48d8e8d3ea: Pull complete
77bf5577ec48: Pull complete
848670200eba: Pull complete
f97d7fbff663: Pull complete
5a3f0e5473db: Pull complete
8e3228f7f0c3: Pull complete
ef8e95b03d6f: Pull complete
d3bf98c23296: Pull complete
efd261030b68: Pull complete
07a028b52e68: Pull complete
```

```

Creating ubuntu_redis_1    ... done
Creating ubuntu_mysql_1    ... done
Creating ubuntu_mailhog_1  ... done
Creating ubuntu_selenium_1 ... done
Creating ubuntu_pmp_1      ... done

mailhog_1 | 2023/04/26 10:11:44 Using in-memory storage
mailhog_1 | 2023/04/26 10:11:44 Serving under http://0.0.0.0:8025/
mailhog_1 | [HTTP] Binding to address: 0.0.0.0:8025
mailhog_1 | 2023/04/26 10:11:44 [SMTP] Binding to address: 0.0.0.0:1025
mailhog_1 | Creating API v1 with WebPath:
mailhog_1 | Creating API v1 with WebPath:
mysql_1  | [Entrypoint] MySQL Docker Image 8.0.32-1.2.11-server
mysql_1  | [Entrypoint] Initializing database
mysql_1  | 2023-04-26T10:11:44.786320Z 0 [Warning] [MY-011068] [Server] The syntax '--skip-host-cache' is deprecated and will be removed in a future release. Please use SET GLOBAL host_cache_size=0 instead.
mysql_1  | 2023-04-26T10:11:44.788220Z 0 [System] [MY-013169] [Server] /usr/sbin/mysql (mysqld 8.0.32) initializing of server in progress as process 16
mysql_1  | 2023-04-26T10:11:44.952115Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
redis_1   | 1:C 26 Apr 2023 10:11:44.111 * o000o000o000 Redis is starting o000o000o000
redis_1   | 1:C 26 Apr 2023 10:11:44.112 # Redis version=7.0.11, bits=64, commit=00000000, modified=0, pid=1, just started
redis_1   | 1:C 26 Apr 2023 10:11:44.112 # Warning: no config file specified, using the default config. In order to specify a config file use redis-server /path/to/redis.conf
redis_1   | 1:M 26 Apr 2023 10:11:44.113 * monotonic clock: POSIX clock_gettime
redis_1   | 1:M 26 Apr 2023 10:11:44.123 * Running mode=standalone, port=6379.
redis_1   | 1:M 26 Apr 2023 10:11:44.123 # Server initialized
redis_1   | 1:M 26 Apr 2023 10:11:44.123 # WARNING Memory overcommit must be enabled! Without it, a background save or replication may fail under low memory condition. Being disabled, it can also cause failures without low memory condition, see https://github.com/jemalloc/jemalloc/issues/1328. To fix this issue add 'vm.overcommit_memory = 1' to /etc/sysctl.conf and then reboot or run the command 'sysctl vm.overcommit_memory=1' for this to take effect.
redis_1   | 1:M 26 Apr 2023 10:11:44.127 * Ready to accept connections
selenium_1 | 2023-04-26 10:11:45,848 INFO Included extra file "/etc/supervisor/conf.d/selenium.conf" during parsing
selenium_1 | 2023-04-26 10:11:45,873 INFO RPC interface 'supervisor' initialized
selenium_1 | 2023-04-26 10:11:45,873 CRIT Server 'unix_http_server' running without any HTTP authentication checking
selenium_1 | 2023-04-26 10:11:45,874 INFO supervisor started with pid 8
pmp_1     | 2023-04-26 10:11:46,840 INFO Set uid to user 0 succeeded
pmp_1     | 2023-04-26 10:11:46,844 INFO supervisor started with pid 1
selenium_1 | 2023-04-26 10:11:46,895 INFO spawned: 'xvfb' with pid 10
selenium_1 | 2023-04-26 10:11:46,899 INFO spawned: 'vnc' with pid 11
selenium_1 | 2023-04-26 10:11:46,903 INFO spawned: 'novnc' with pid 12
selenium_1 | 2023-04-26 10:11:46,908 INFO spawned: 'selenium-standalone' with pid 13
selenium_1 | Setting up SE_NODE_GRID_URL...
selenium_1 | 2023-04-26 10:11:46,938 INFO success: xvfb entered RUNNING state, process has stayed up for > than 0 seconds (startsecs)
selenium_1 | 2023-04-26 10:11:46,939 INFO success: vnc entered RUNNING state, process has stayed up for > than 0 seconds (startsecs)
selenium_1 | 2023-04-26 10:11:46,939 INFO success: novnc entered RUNNING state, process has stayed up for > than 0 seconds (startsecs)
)
selenium_1 | 2023-04-26 10:11:46,939 INFO success: selenium-standalone entered RUNNING state, process has stayed up for > than 0 seconds (startsecs)
pmp_1     | 2023-04-26 10:11:47,858 INFO spawned: 'php' with pid 8
pmp_1     | Could not open input file: /var/www/html/artisan
pmp_1     | 2023-04-26 10:11:48,988 INFO success: php entered RUNNING state, process has stayed up for > than 1 seconds (startsecs)
pmp_1     | 2023-04-26 10:11:48,989 INFO exited: php (exit status 1; not expected)
pmp_1     | 2023-04-26 10:11:49,994 INFO spawned: 'php' with pid 9

```

## A.11 Hosting throw Microsoft Azure

### Resources

Recent   Favorite

Name	Type	Last Viewed
pmp-webapp	App Service	2 weeks ago
pmp-test	App Service	2 weeks ago
pmpApp	Container registry	2 weeks ago
pmpApp	Resource group	a month ago
app	Resource group	a month ago
pmp-mysql-1	SQL database	a month ago
pmp-mysql-1	SQL server	a month ago
ASP-pmpApp-a456	App Service plan	a month ago

The screenshot shows the Azure portal interface for the 'pmpApp' container registry. The top navigation bar includes the Microsoft Azure logo, a search bar, and user information. Below the header, the breadcrumb navigation shows 'Home > pmpApp'. The main content area displays the 'Overview' tab of the 'pmpApp' container registry. On the left, there is a sidebar with navigation links for Overview, Activity log, Access control (IAM), Tags, Quick start, Events, Settings (Access keys, Encryption, Identity, Networking, Microsoft Defender for Cloud, Locks), Services (Repositories, Webhooks, Replications, Tasks, Connected registries (Preview), Cache (Preview)), and Registry metrics. The central panel shows the 'Essentials' section with resource group (pmpApp), location (North Europe), subscription (Azure subscription 1), and provisioning state (Succeeded). It also shows storage usage: 500 GB included in SKU, 0.40 GB used, and 0.00 GB additional storage. A 'ACR Tasks' section is present, along with 'Container security integrations' and 'Microsoft Defender for Cloud' information. At the bottom right, there is a 'JSON View' link.

## : ( Application Error

If you are the application administrator, you can access the [diagnostic resources](#).

## A.12 Modified docker-compose.yml file

```
version: '3'
services:
  pmp:
    build:
      context: ./vendor/laravel/sail/runtimes/8.1
      dockerfile: Dockerfile
      args:
        WWWGROUP: '${WWWGROUP}'
    image: sail-8.1/app
    extra_hosts:
      - 'host.docker.internal:host-gateway'
    ports:
      - '${APP_PORT:-80}:80'
      - '${VITE_PORT:-5173}:${VITE_PORT:-5173}'
    environment:
      WWWUSER: '${WWWUSER}'
      LARAVEL_SAIL: 1
      XDEBUG_MODE: '${SAIL_XDEBUG_MODE:-develop,debug,coverage}'
      XDEBUG_CONFIG:
        '${SAIL_XDEBUG_CONFIG:-client_host=host.docker.internal}'
    volumes:
      - './var/www/html'
    networks:
      - sail
    depends_on:
      - mysql
      - redis
      - mailhog
      - selenium
  mysql:
    image: 'mysql/mysql-server:8.0'
    ports:
      - '${FORWARD_DB_PORT:-3306}:3306'
    environment:
      MYSQL_ROOT_PASSWORD: '${DB_PASSWORD}'
      MYSQL_ROOT_HOST: "%"
      MYSQL_DATABASE: '${DB_DATABASE}'
      MYSQL_USER: '${DB_USERNAME}'
      MYSQL_PASSWORD: '${DB_PASSWORD}'
      MYSQL_ALLOW_EMPTY_PASSWORD: 1
    volumes:
      - 'sail-mysql:/var/lib/mysql'|
```

```
- ./vendor/laravel/sail/database/mysql/create-testing-database.sh:/dock
er-entrypoint-initdb.d/10-create-testing-database.sh'
networks:
  - sail
healthcheck:
  test: ["CMD", "mysqladmin", "ping", "-p${DB_PASSWORD}"]
  retries: 3
  timeout: 5s
redis:
  image: 'redis:alpine'
  ports:
    - '${FORWARD_REDIS_PORT:-6379}:6379'
  volumes:
    - 'sail-redis:/data'
  networks:
    - sail
  healthcheck:
    test: ["CMD", "redis-cli", "ping"]
    retries: 3
    timeout: 5s
mailhog:
  platform: linux/amd64
  image: 'mailhog/mailhog:latest'
  ports:
    - '${FORWARD_MAILHOG_PORT:-1025}:1025'
    - '${FORWARD_MAILHOG_DASHBOARD_PORT:-8025}:8025'
  networks:
    - sail
selenium:
  platform: linux/amd64
  image: 'selenium/standalone-chrome'
  extra_hosts:
    - 'host.docker.internal:host-gateway'
  volumes:
    - '/dev/shm:/dev/shm'
  networks:
    - sail
networks:
  sail:
    driver: bridge
volumes:
  sail-mysql:
    driver: local
  sail-redis:
```

```
  driver: local
sail-meilisearch:
  driver: local
```

## A.13 Poster

**OsloMet**  
Oslo Metropolitan University  
Department of Computer Science  
Oslo, Norway

### STUDENT INTERNSHIP MATCH-MAKING

**Bachelor Team 57**  
Gurjot Singh Aulakh - s351873  
Mortaza Baqeri - s351899  
Okbamichael Mussie Elias - s331690  
Faesal al shhadat - s339410  
Arman Yedicam - s351926

**1. Background**

- Norwegian government encouraged educational institutions to focus on better university-industry collaborations by including more and better practical internships.
- OsloMet is contributing to this objective by providing nine practical IT project courses per semester.

**4. Solution**

We have developed a web application that streamlines the course management process, eliminating the need for multiple applications and providing a convenient overview of student progress and project status. With our application, managing the course has never been easier or more efficient.

The application serves three actors, course coordinator, companies and students

**2. Today's situation**

In 2018-2023, 492 students completed 194 projects across 145 organizations as a project course.

**3. Problem & Consequence**

**Problems**

- Time consuming:
- Many applications are used to manage the course (Word, Excel, Nettskjema, and Canvas)

**Manual work**

- Keeping track of students and projects are done in a manual and tedious way

**Consequence**

- Limited capacity
- Course coordinator may have to set a limit how many students and project can be accepted each semester.

**Working Method**

Agile Scrum Methodology

*Fig. 1. Scrum*

**Main Technologies**

*Fig. 2. MySQL*

*Fig. 3. GitHub*

*Fig. 4. Laravel*

*Fig. 5. Docker*

**5. Impact**

Creating a unified platform will improve course management for our coordinator, allowing us to accommodate more projects and students than in previous years. Additionally, we have created a shared meeting space for students, employers, and the school.

Sources:

- <https://appneta.com/what-is-scrum/>
- <https://www.pngwing.com/en/free-png-powd>
- [https://www.flaticon.com/free-icon/github-logo-png-flaticon-free-vector-image\\_1384512](https://www.flaticon.com/free-icon/github-logo-png-flaticon-free-vector-image_1384512)
- <https://www.flaticon.com/free-icon-dhad>
- <https://www.flaticon.com/free-icon-yenck>

Sign up to receive notification when the application goes live

202

## A.14 Certificate from the course coordinator



### Attest

Gurjot Singh Aulakh, Mortaza Baqeri, Okbamichael Mussie Elias, Faesal Al Shhadat and Arman Yedicam completed their bachelor thesis *SIM – student internship match-making platform* in Spring 2023, with myself as both the collaborative partner and internal supervision.

The bachelor thesis project built on the work four of the students (Gurjot Singh Aulakh, Mortaza Baqeri, Okbamichael Mussie Elias, Faesal Al Shhadat, and Krystian Karol Mazurkiewicz who did not continue the project) completed as part of a project-based course (DATA3710) in Autumn 2022.

The aim of the project was to build a prototype platform from the ground up that could make it easier to manage and coordinate student-industry collaborative projects and the students have achieved this. The MVP the students have produced will be further developed and refined and become an integral part of how The Department of Computer Science at OsloMet will organise its industry project offerings for over 80 students a semester in the future.

Throughout their time working on the project, Aulakh, Baqeri, Elias, Al Shhadat and Yedicam have shown a great deal of initiative and willingness to suggest and implement their own ideas, to the betterment of the final solution. The students have produced an impressive amount of work on a complex and demanding project. They have shown both a willingness and ability to receive and act on input and feedback, test themselves with unfamiliar tools and methods, and work independently, both as a group and on an individual basis. Challenges that arose during the project, both technical, logistical, and collaborative, were handled with focus and maturity.

I would like to thank the group for their efforts and for a fruitful collaboration.

A handwritten signature in black ink, appearing to read "Tulpesh".

Tulpesh Patel, Associate Professor, Department of Computer Science, OsloMet.