**TU Dublin LOGO to be inserted..**

**Title…**

**Names, Student Numbers**

**Date When Submitted**

**Group Project**

Submitted in part fulfilment for the degree of

**Programme Name and Code**

School of Engineering and Informatics,

TU Dublin – Blanchardstown Campus,

Dublin, Ireland

**Abstract**

The present project ……………this should be written last.

**Contents**

Chapter 1: Introduction 5

1.1 Background 5

1.2 Motivation and Problem Statement 5

1.3 Project Aim(s)/Research Question(s) 5

1.4 Project Objectives 6

1.5 Methodology Overview 6

1.5.1 Systems Development Life Cycle Models 6

1.5.2 Agile 7

1.5.3 Scrum 9

1.5.4 The Scrum Team 9

1.5.5 Scrum Artefacts 9

1.5.6 Scrum Events 9

1.6 Thesis Overview 10

1.7 Summary 10

Chapter 2: Literature Review 11

2.1 Overview 11

2.2 Area 1 11

2.3 Area 2 11

2.4 Area 3 11

2.5 Summary 11

Chapter 3: System Requirements and Specification 12

3.1 Overview 12

3.2 Use Cases 13

3.2.1 Login System 14

3.2.2 Sign-up System 15

3.2.3 Forum System and Private Group Forum System 17

3.2.4 Profile Management System 19

3.2.5 Private Collaboration System 22

3.2.6 Group Collaboration System 24

3.3 Software and Hardware Requirements 26

3.3.1 Overview 26

3.3.2 Front-end 26

3.3.3 Back-end 27

3.3.4 Persistent Storage 28

3.3.5 PDF Text Extraction 28

3.4 Testing 29

3.5 Summary 29

Chapter 4: System Design 30

4.1 Overview of System Architecture 30

4.2 Design Patterns (if any) 30

4.3 Class/File Organisation and Packages/Folders 30

4.4 Graphical User Interface 30

4.5 Data Permanency Mechanism 30

4.6 Optimisation (if any) 30

4.7 Summary 30

Chapter 5: Implementation 31

5.1 Overview 31

5.2 Front-End/View/Presentation Layer Implementation 31

5.3 Middle/Controller/Application Layer Implementation 31

5.4 Back-End/Model/Data Layer Implementation 31

5.5 Summary 31

Chapter 6: Testing and evaluation 32

6.1 Overview 32

6.1 Unit Testing 32

6.2 Integration testing 32

6.3 Debugging 32

6.4 Exception Handling 32

6.5 Evaluation (if you have time) 32

6.6 Summary 32

Chapter 7: Conclusions and Further Work 33

References 34

Appendix A: Application Link(s) 35

Appendix B: Any other extras… 36

# Introduction

## Background

Online collaboration has become a hot topic in recent years, with many companies stressing the importance of teamwork and the ability to be a team player. Any large-scale project involves many team members, some located in separate buildings or countries, and the need to effectively coordinate the communication and collaboration has led to the development of some sophisticated online tools. In academia, this has led to a greater focus on group-work and group assignments, to promote the benefits of learning from each other. While studying, and especially when preparing for exams, the lack of an effective tool aimed at student collaboration became apparent. With many students communicating using social media and instant messaging services, and others using shared documents, the need for a common resource that housed all the tools necessary to facilitate efficient collaboration arose. Research into the area of Online Collaboration has shown that the benefits of working together, and of building knowledge within the context of a group, has significant achievement and satisfaction implications for learners. With this in mind, this project proposes to develop a suitable system for students to use, while working on group assignments or preparing for exams, that will incorporate the convenience of instant messaging, the knowledge sharing capabilities of a public forum, and the practicalities offered by a shared document system.

## Motivation and Problem Statement

This project aims to address the lack of student focused Online Collaboration tools and outline the features and functionality that such a tool should provide to better aid students while working on group assignments, exam preparation, and learning in the course of their studies. The core functionalities when implemented are intended to increase student participation in group work by providing a user-friendly experience with all the tools necessary for effective collaboration in one convenient web application.

## Project Aim(s)/Research Question(s)

To achieve the aims, the team will implement the following features to meet the required objectives:

* A public forum will be implemented as a central repository of information on certain user created topics of discussion or questions. This forum will be publicly available to all users, however, to contribute to or create a topic a user must be registered.
* Users will be able to register an account to avail of the additional features outside of searching and browsing the public forum.
* Registered Users can upload a PDF document, such as a past exam paper, which will be processed by the application to extract questions and create discussion topics automatically, to simplify the process.
* Registered Users can create private groups by inviting other users to collaborate on questions or topics in real time.
* Private groups will have access to a shared document, a place to upload resources such as text or links to relevant information, and a text chat feature to allow for communication between group members.
* Registered Users will be able to rate answers in the public forum based on how well they answer the question, allowing for a consensus to be reached on how helpful an answer is.
* Registered Users will be able to tag questions to assign them to the most appropriate topic.
* Registered Users will be able to rank questions according to the level of difficulty.
* Registered User will be able to take a quiz on a topic, with questions generated from a random selection from the public forum, to test their knowledge.

## Project Objectives

State the exact objectives: they should be related to the overall aim and they should give an indication of the steps you need to take to complete the project. For example:

* To review and critically analyse existing approaches (related to the area of investigation).
* To identify the features/functionalities that are necessary to address the problem.
* To identify…
* …

## Methodology Overview

### Systems Development Life Cycle Models

The development of a software system and the processes that are involved in creating it are referred to as the Systems Development Life Cycle (SDLC). Many SDLC’s exist and each have their own strengths and weaknesses depending on what type of system is under development. However, they all share some common stages and the differences lie in how each of these stages are approached and how they relate to each other. No matter the size or complexity of the system the essential stages are:

* Planning – A feasibility study, estimation of resources, and project planning are carried out.
* Analysis – Requirements Analysis to gather information on how best to solve the problem at hand.
* Design – A detailed solution proposal based on the requirements, is used to guide the developers.
* Development – The system is developed and tested to ensure it meets the requirements.
* Deployment – The system is released for users to work with.
* Maintenance – The system is supported and/or improved over its’ lifetime.

Many SDLC’s lend themselves more suitably to different systems depending on a number of key factors, such as the size and complexity of the project, the size and experience of the development team, the understanding of the technologies involved, the flexibility of the customers’ requirements and the time allocated for the system development. Classic Models, such as the Waterfall, Spiral, and Incremental Models are particularly suited to systems where the requirements are fully known and do not change over the course of development. Others, such as the Rapid Application Development (RAD), Extreme Programming (XP), and Agile Models can be particularly well suited when the requirements are not fully known or are prone to change and provide more feedback to stakeholders with frequent working demonstrations. Many comparisons have been made of different SDLC Models and while some are more rigid, the advantages of flexibility and scalability afforded by Agile make it one of the most popular methodologies in today’s fast paced world (Ally & Ning, 2015).

### Agile

For the project and workflow management (System Development Life-Cycle or SDLC) the Agile method will be used. As reflected in Agile Manifesto (Agile Alliance, 2019), Agile method is, in essence, an iterative approach, that focuses on user’s needs and feedback, with frequent communication between user and the development team. The teams itself are self-organizing, bound by a small number of rules and necessary formalities, a fact that eases and speeds up the development process. In addition to the Agile method itself, Scrum framework and Kanban board will be used for task management. Although not a methodology, Scrum encompasses aspects of development and tackles issues of communication and provides an easier way to manage projects by decomposing complexity into manageable parts, with enforced team a cooperation being a major benefit (Scrum.org, 2019). While Scrum is concerned by how is it done, a way of assigning the actual task is needed, e.g. what is done and by whom is it done (Atlassian, 2019); for this purpose, a Kanban board shall be used – a complex problem discussed during a Scrum stand-up, with a plan of action devised, will be actually split into the tasks to be done by individual team members for the current sprint, all under aegis of Agile method.

The reason Agile has been chosen as the methodology is the fact that the project proposed lends itself to it:

* Primarily, the project can be separated into three different modules – front-end, back-end and persistent data management; furthermore, these modules can themselves be split into even simpler parts, for example, a login subsystem or PDF processing subsystem. It is one of Agile method’s main objectives to deliver working software at the end of each sprint, and with problem decomposition the team will be able to continuously deliver a working piece of software. Moreover, if, for any reason, the team will be unable to finish some of the features in a given timeframe, those features can be pushed onto the next sprint. As a result, there will be less pressure on the individual team members, and the development process will not go astray.
* The aim of Agile is to avoid unnecessary documentation and over-modelling to enable iterative development. As development process goes on, the team will decide whether certain features should be added, for example, whether part of the architecture should be migrated onto a cloud platform, such as Azure or AWS. Since not bound by constraints of the existing modules, it will be easier to revise the architecture when using Agile.
* Using Scrum, a proper communication between team members will be ensured. Also, since Scrum separates development team member’s into separate roles, it will be possible for one or more team members assume managerial positions, while avoiding unnecessary formalities
* Since the project will be split into different tasks and sub-tasks to be done by team members, it is essential for the team to use a tool to organize these tasks; Kanban board will serve this purpose, enabling to track tasks that are finished and that are in progress. Moreover, with the tasks laid out in an orderly manner and difficulty assigned to them, it will be possible to:
  + Track team’s average progress per month, based on the number of tasks completed or based on the scoring system assigned to the task (e.g. a Fibonacci sequence, with easiest task having a score of one, a more complicated task having a number three and up to number thirteen). This will assist in overall planning and will help to determine what additional features are feasible to implement
  + Assign priority to the tasks, with less-important tasks being pushed onto next sprint
* Agile focuses on user experience, thus, the team is in a unique position to implement this project, because the proposed project is a collaboration tool designed to help student and the team members themselves are students. This will give an opportunity to test the project both from developer and user perspective, as well as survey team’s fellow students about features they would possibly want, thus enriching the user stories for the project.

### Scrum

Of the many frameworks that use the Agile methodology, Scrum is probably the best known. Scrum has three main components; The Scrum Team, Scrum Events and Scrum Artefacts.

Figure 1. Scrum Framework Diagram

### The Scrum Team

The Scrum Team is made up of the Product Owner, who is responsible for maximizing the value of the product, The Development Team who develop the product, and the Scrum Master, who is responsible for helping everyone comprehend the theory, practices, rules and values of Scrum.

### Scrum Artefacts

**Product Backlog**

The Product Backlog is a list of everything that the team has identified as being needed in the product. It is the definitive source of requirements and any changes that need to be made to the product. It changes dynamically as the teams understanding of the product and its’ features evolve.

**Sprint Backlog**

The Sprint Backlog is a subset of the product backlog items that are to be implemented in a sprint. It identifies all the work that is needed to achieve the sprint goal.

**Increment**

The Increment is a sum of completed items from the product backlog of the current sprint and all previous Increments. Each item must meet the teams’ definition of done and must be in a useable condition.

### Scrum Events

There are five Scrum events used to create regularity and avoid unnecessary meetings. Each event is an opportunity to analyse and improve the process.

**The Sprint**

The Scrum framework divides the development process into a set of time periods in which useable and potentially shippable products are delivered, known as sprints. The products’ features are broken down into separate tasks and added to the Product Backlog. The team decide what tasks are to be added to each sprint.

**Sprint Planning**

A sprint begins by planning the work to be done and setting up goals. A detailed list of tasks is created and added to the Sprint Backlog.

**Daily Scrum**

Daily Scrum meetings are held to assess the progress of the team towards the sprint goals and to increase collaboration and performance.

**Sprint Review**

At the end of each sprint, a Sprint Review is held to gain feedback on what was achieved and to revise the Product Backlog.

**Sprint Retrospective**

A Sprint Retrospective is also held to enable the team to analyse their progress and suggest improvements that can be applied to the next sprint.

## Thesis Overview

Briefly describe what each chapter deals with (should be done at the end).

## Summary

Summary of the present chapter.

# Literature Review

## Overview

Indicate the areas that you are investigating and relate these to the aim. Note that the number of areas (and, thus, corresponding subsections) will vary, depending on what you are going to research.

## Area 1

…

## Area 2

…

## Area 3

…

## Summary

Summarise your findings.

# System Requirements and Specification

## Overview

The aim of this project is to develop a Web-based, user-friendly application that provides students with a platform on which they can collaborate in order to prepare for their assignments or to further their understanding of a certain module, enquire other students and/or collaborators for assistance and to provide assistance for those in need of it. The application has features closely resembling those of a social network to avail of the boons those networks have that enable efficient and fast communication – such as chats and private collaborations created by students themselves. However, to alleviate the drawbacks of social networks, such as sharing unnecessary information, certain barriers are imposed by the application itself, for example, restricting chats to those users who are either part of a same collaboration group or have added each other as collaborators.

Based on the literature review and preliminary survey conducted, a certain set has been agreed upon. Some of those features are available for both non-signed-up users and signed-up users, while others are restricted for signed-up users only. The features available for both non-signed-up and signed-up users are:

Public forum with automatic tagging system –will be able to browse the forum, search for specific questions by area as denoted by tags and, if a user is a signed-up user who is also logged in, will be able to create a new question and/or reply to other questions, as well as either upvoting or downvoting other questions and replies.

The features available for signed-up users exclusively are:

Profile management – the users may review their profile, change their profile settings, add or remove information, as well as see their most recent questions, responses and most upvoted responses.

Add new collaborators to work together –the users may send collaboration requests to other users, which may be found through public forum using their username or by searching for a user using either their username or the optional details a user might have provided, such as institution name.

Instant messaging between collaborators – the users may send messages to other users, provided that they have added each other as collaborators.

Private collaborations – the users may create private collaboration groups with one or more users, provided that the user who created a group and the user who is invited to the group have added each other as collaborators. Private collaboration groups provide all members of the group with private group chat, private forum and exam preparation tools (discussed in the next bullet point).

Exam preparation tools – the users may upload a PDF document with the exam questions they wish to prepare for; the questions will be extracted from the PDF, and a private real-time workspace will be available for use.

## Use Cases

Each feature described above has been warranted a separate use case, with some minor adjustments:

Sign-up and login were merged into a single use case, as their functionality is closely related, while being too slim to be given separate use case.

Instant messaging is present in use cases for both personal and group collaboration systems and has not been given a separate use case due to being too small to warrant one.

Certain features are available to all users, irrespective of whether they signed-up or not; for features that are available for both types of users, a use case diagram will have two user actors, an Unregistered User/Non-signed-up user and a Signed-up User. Signed-up user extends from an Unregistered user, with implied meaning that all features available for Unregistered users are implicitly available for Signed-up users. For use cases where no Unregistered users are present, the implied meaning is that the feature is available for Signed-up users only. In addition to the abovementioned, signed-up users who are part of a group collaboration might take on the role of a group administrator, which has extended control over the group collaboration.

### Login System

Figure 2. Login System

This use case describes a sequence of steps to be performed when user wishes to login into existing account, with user proceeding as follows:

1. Click on ‘username’ field and enter their username;
2. Click on ‘password’ field and enter their password;
3. Click on ‘login’ button, which will trigger a login request to the server.

The alternative scenarios would arise from a combination of username or password being invalid or an internal server error, and can be summarized as follows:

1. Alternative scenario point 1: The user with the submitted username either does not exist:
2. Display error message and go back to step 1.
3. Alternative scenario point 2: The combination of username and password does not match:
4. Display error message and go back to step 1.
5. Alternative scenario point 3: The server has encountered an internal error that prevented it from fulfilling the request:
6. Display error message and prompt user to try again later, which would return user to step 1.

### Sign-up System

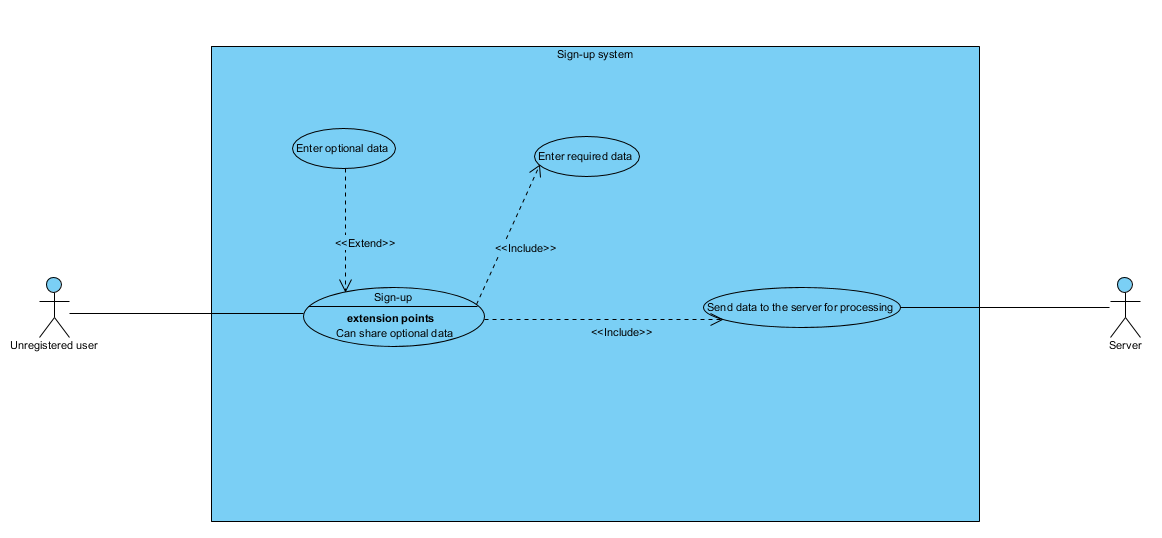
This use case describes a sequence of steps to be performed when user wishes to sign up for a new account, with user proceeding as follows:

Figure 3. Sign-up System

1. Click on ‘sign-up’ button;
2. Click on ‘username’ field and enter the username that user wants to be identified with;
3. Click on ‘password’ field and enter the password user wants to have for their account;
4. Click on ‘repeat password’ field and repeat the password entered in step 3.;
5. Click on ‘email’ password field and enter the email user wants to be associated with their account;
6. Click on ‘submit’ button, which will trigger a sign-up request to the server;
7. The user will be prompted to either share optional information about themselves and proceed to the steps outlined in 7a or to proceed to step 7b:
   1. If the user decides to share additional information:
      1. Click on ‘name’ and enter their name;
      2. Click on ‘surname’ and enter their surname;
      3. Click on ‘gender’ and enter their gender;
      4. Click on ‘institution’ and enter their institution;
      5. Click on ‘finish registration’ button, which triggers a request to the server;
   2. If the user decides not to share additional information:
8. Click on ‘finish registration’ button, which triggers a request to the server.

The alternative scenarios would arise from mandatory user data being invalid or due to an internal server error and can be summarized as follows:

1. Alternative scenario point 1: The server has encountered an internal server error that prevented it from redirecting user to the sign-up page:
   1. Display error message and prompt user to try again later, which would return them to step 1.
2. Alternative scenario point 2: The username field is invalid:
   1. If the username field contains invalid characters:
      1. Display user the error message showing invalid input characters;
      2. Go back to step 2.
   2. If the username field is too short or too long:
      1. Display user the error message showing maximum and minimum acceptable username length;
      2. Go back to step 2.
   3. If the username already exists:
      1. Display user the error message and prompt to choose another username;
      2. Go back to step 2.
3. Alternative scenario point 3: The password field is invalid:
   1. If the password field contains invalid characters:
      1. Display user the error message showing invalid input characters;
      2. Go back to step 3.
   2. If the password field is too short or too long:
      1. Display user the error message showing maximum and minimum acceptable password length;
      2. Go back to step 3
4. Alternative scenario point 4: The password entered in the repeat password field does not match the password entered in the password field:
   1. Display user the error message and go back to step 4.
5. Alternative scenario point 5: The email entered is not a valid email address:
   1. Display user the error message and go back to step 5.
6. Alternative scenario point 6: The server has encountered an internal server error that prevented it from fulfilling sign-up request:
   1. Display user the error message and prompt user to try again later, which would return them to step 1.
7. Alternative scenario point 5.A.V and 5.B.I: The server has encountered an internal server error that prevented it from fulfilling the request:
   1. Display user the error message and prompt user to try again later, which would return them to step 7.

### A close up of a map Description automatically generatedForum System and Private Group Forum System

Figure 4. Forum System

This use case describes a sequence of steps to be performed when user wishes to avail of forum functionality; to browse for questions, the user would proceed as follows:

1. Click ‘forum’ button;
2. Browse through the list of displayed questions;
3. Click on a specific question to view it.

The alternative scenarios would arise if the data retrieval from the server fails due to internal server error and can be summarized as follows:

1. The server failed to retrieve list of questions due to internal server error:
   1. Display user the error message and prompt to try again later;
   2. Return to step 1.
2. The server failed to retrieve a specific question data due to internal server error:
   1. Display user the error message and prompt to try again later;
   2. Return to step 1.

To create a new question, the user would proceed as follows:

1. Click ‘forum’ button;
2. Click ‘create a question’ button;
3. Enter question title in the ‘title’ field;
4. Enter question subtitle in the ‘subtitle’ field;
5. Enter question description in the ‘description’ field;
6. Enter question tags in the ‘tags’ field;
7. Click ‘submit’ button.

The alternative scenarios would arise if either the user input is invalid or if the server fails due to internal server error and can be summarized as follows:

1. Alternative scenario point 1: The server failed to retrieve list of questions due to internal server error:
   1. Display user the error message and prompt to try again later;
   2. Return to step 1
2. Alternative scenario point 2: The redirection to the question creation form failed due to internal server error:
   1. Display user the error message and prompt to try again later;
   2. Return to step 1.
3. Alternative scenario point 3: The title field contains invalid characters:
   1. Display user the error message showing the list of invalid input characters;
   2. Return to step 3.
4. Alternative scenario point 4: The subtitle field contains invalid characters:
   1. Display user the error message showing the list of invalid input characters;
   2. Return to step 4.
5. Alternative scenario point 5: The description field contains invalid characters:
   1. Display user the error message showing the list of invalid input characters;
   2. Return to step 5.
6. Alternative scenario point 6: The tags field contains invalid characters:
   1. Display user the error message showing the list of invalid input characters;
   2. Return to step 6.
7. Alternative scenario point 7: The question submission failed due to internal server error:
   1. Display user the error message and prompt to try again later;
   2. Return to step 1.

To reply to a question, the user would proceed as follows:

1. Click ‘forum’ button;
2. Browse the list of questions;
3. Select a question to reply to;
4. Click on ‘reply’ button;
5. Enter the reply in the ‘reply’ field;
6. Click ‘submit reply’ button.

The alternative scenarios would arise if either the user input is invalid or if the server failed to process the request due to internal server error and can be summarized as follows:

1. Alternative scenario point 1: The server failed to retrieve the list of questions:
   1. Display user the error message and prompt to try again later;
   2. Return to step 1.
2. Alternative scenario point 3: The server failed to retrieve a question user has selected:
   1. Display user the error message and prompt to try again later;
   2. Return to step 1.
3. Alternative scenario point 5: The user is invalid:
   1. If the user input contains invalid characters:
      1. Display user the error message showing the list of invalid characters;
      2. Return to step 5.
   2. If the user input is too short or too long:
      1. Display user the error message showing the minimum and maximum acceptable reply length;
      2. Return to step 5.
4. Alternative scenario point 6: The server failed to process the request due to internal server error:
   1. Display user the error message and prompt to try again later;
   2. Return to step 4.

To upvote or a downvote a question or a reply, the user would proceed as follows:

1. Click ‘forum’ button;
2. Browse the list of questions;
3. Select a question;
4. Click on a ‘upvote’ or ‘downvote’ button on a question or an appropriate reply.

The alternative scenarios would arise if server was unable to fulfil the request due to internal server error:

1. Alternative scenario point 1: The server failed to retrieve the list of questions:
   1. Display user the error message and prompt to try again later;
   2. Return to step 1
2. Alternative scenario point 3: The server failed to retrieve a question
   1. Display user the error message and prompt to try again later;
   2. Return to step 1.
3. Alternative scenario point 4: The server failed to update rating due to internal server error:
   1. Display user the error message and prompt to try again later;
   2. Return to step 4.

### Profile Management System

A close up of a map

Description automatically generated

Figure 5. Profile Management System

This use case describes a sequence of steps to be performed when user wishes to avail of profile management functionality; to update their password, the user would proceed as follows:

1. Click the ‘settings’ button
2. Select the ‘account management’ pane
3. Enter the new password into the ‘password’ field
4. Repeat new password in the ‘repeat password’ field
5. Click the ‘submit’ button

The alternative scenarios would arise from mandatory user data being invalid or due to an internal server error and can be summarized as follows:

1. Alternative scenario point 1: The server has encountered an internal server error that prevented it from redirecting user to the settings page:
   1. Display error message and prompt user to try again later, which would return them to step 1.
2. Alternative scenario point 3: The password field is invalid:
   1. If the password field contains invalid characters:
      1. Display user the error message showing invalid input characters;
      2. Go back to step 3.
   2. If the password field is too short or too long:
      1. Display user the error message showing maximum and minimum acceptable password length;
      2. Go back to step 3
3. Alternative scenario point 4: The password entered in the repeat password field does not match the password entered in the password field:
   1. Display user the error message and go back to step 4.
4. Alternative scenario point 5: The server has encountered an internal server error that prevented it from fulfilling update request:
   1. Display user the error message and prompt user to try again later, which would return them to step 1.

To update their email, the user would proceed as follows:

1. Click the ‘settings’ button
2. Select the ‘account management’ pane
3. Enter the new email into the ‘email’ field
4. Click the ‘submit’ button

The alternative scenarios would arise from mandatory user data being invalid or due to an internal server error and can be summarized as follows:

1. Alternative scenario point 1: The server has encountered an internal server error that prevented it from redirecting user to the settings page:
   1. Display error message and prompt user to try again later, which would return them to step 1.
2. Alternative scenario point 3: The email entered is not a valid email address:
   1. Display user the error message and go back to step 3.
3. Alternative scenario point 4: The server has encountered an internal server error that prevented it from fulfilling update request:
   1. Display user the error message and prompt user to try again later, which would return them to step 1.

To update their private settings, the user would proceed as follows:

1. Click the ‘settings’ button
2. Select the ‘privacy settings’ pane
3. Select whether they want their profile to be visible to public
4. Select whether they want their page to be publicly visible
5. Select whether they want their forum activity to be publicly visible
6. Click the ‘submit’ button

The alternative scenarios would arise due to an internal server errors and can be summarized as follows:

1. Alternative scenario point 1: The server has encountered an internal server error that prevented it from redirecting user to the settings page:
   1. Display error message and prompt user to try again later, which would return them to step 1.
2. Alternative scenario point 6: The server has encountered an internal server error that prevented it from fulfilling update request:
   1. Display user the error message and prompt user to try again later, which would return them to step 1.

To update their personal information, the user would proceed as follows:

1. Click the ‘settings’ button
2. Select the ‘profile settings’ pane
3. Enter their desired name in the ‘name’ field, or leave it as blank
4. Enter their desired surname in the ‘surname’ field, or leave it as a blank
5. Enter their institution in the ‘institution’ field, or leave it as a blank
6. Enter their gender in the ‘gender’ field, or leave it as a blank
7. Click the ‘submit’ button

The alternative scenarios would arise due to an internal server errors and can be summarized as follows:

1. Alternative scenario point 1: The server has encountered an internal server error that prevented it from redirecting user to the settings page:
   1. Display error message and prompt user to try again later, which would return them to step 1.
2. Alternative scenario point 7: The server has encountered an internal server error that prevented it from fulfilling update request:
   1. Display user the error message and prompt user to try again later, which would return them to step 1.

### Private Collaboration System

A close up of a map

Description automatically generated

Figure 6. Private Collaboration System

This use case describes a sequence of steps to be performed when user wishes to avail of private collaboration-related functionality. Since private collaboration is restricted to users who added each other as collaborators, before engaging into the collaboration, the users have to find each other’s profiles and add each other as collaborators. To find other users, the user would perform the following sequence of steps would have to be performed:

1. Click on ‘private collaborations’ button
2. Select ‘search for users’ pane
3. Search can be performed using user’s username or known user details (provided they are in public access as per user’s privacy settings:
   1. If searching by username, click ‘search by username’ button
      1. Enter user’s username
      2. Click on ‘search’ button
      3. Click on ‘add as collaborator’ button
   2. If searching by profile details, click ‘search by profile details’ button
      1. Select the filter
      2. Enter name and/or surname and/or institution
      3. Click on ‘search button’
      4. Find the user in the list of displayed users
      5. Click on ‘add as collaborator’ button

The alternative scenarios would arise from incorrect/non-existent username entered, from incorrect or not matching any search criteria profile details or due to an internal server error and can be summarized as follows:

1. Alternative scenario point 1: The server has encountered an internal server error that prevented it from redirecting user to the private collaborations page:
   1. Display error message and prompt user to try again later, which would return them to step 1.
2. Alternative scenario point 3.A.I and 3.B.III: The search has no results due to invalid information provided or due to internal server error that prevented it from fulfilling the request:
   1. If the search has displayed no results due to invalid search criteria:
      1. Inform user that no other users were found according to the search criteria and prompt them to try again using different search criteria
      2. Return to step 3.A or 3.B.
   2. If server has failed to process the request due to internal server error:
      1. Display user an error message and return to step 2.
3. Alternative scenario point 3.A.III and 3.B.V: The server failed to send a collaboration request due to internal server error:
   1. Display user an error message and return to step 2.

To view collaboration requests, as well as either accept or reject them, the following sequence of steps would have to be performed:

1. Click on ‘private collaborations’ button
2. Select ‘collaboration requests’ pane
3. View the users that sent the collaboration request and select user whose request to accept or reject
4. Click ‘accept’ button if accepting collaboration request or click ‘reject’ button if rejecting collaboration request

The alternative scenarios would arise due to internal server error that prevented it from fulfilling the request and can be summarized as follows:

1. Alternative scenario point 1: The server has encountered an internal server error that prevented it from redirecting user to the private collaborations page:
   1. Display user an error message and return to step 1
2. Alternative scenario point 3: The server has encountered an internal server error that prevented it from retrieving the list of collaboration requests for the specific user:
   1. Display user an error message and return to step 1
3. Alternative scenario point 4: The server has encountered an internal server error that prevented it from processing acceptance or rejection of collaboration request:
   1. Display an error message and return to step 3

To chat with other users as part of private collaboration, the following sequence of steps would have to be performed:

1. Click on the ‘private collaborations’ button
2. Select ‘chat’ pane
3. The user can either send messages in the existing chat thread or create a new one:
   1. If sending messages in an existing chat thread:
      1. Select the thread from the list of existing threads
      2. Enter the message in the ‘message’ field
      3. Click on ‘send’ button or press enter
   2. If sending messages in a new thread:
      1. Click on the ‘new chat’ button
      2. Select the user with whom a new thread will be created
      3. Enter the message in the ‘message’ field
      4. Click on ‘send’ button

The alternative scenarios would arise due to internal server error that prevented it from fulfilling the request and can be summarized as follows:

1. Alternative scenario point 1: The server has encountered an internal server error that prevented it from redirecting to the private collaborations page:
   1. Display user an error message and return to step 1
2. Alternative scenario point 3.A.I and 3.B.II: The server has encountered an internal server error that either prevented it from retrieving the list of existing threads or list of users with whom a new thread can be started:
   1. Display user an error message and return to step 1
3. Alternative scenario point 3.A.III and 3.B.IV: The server has encountered an internal server error that prevented it from sending a message:
   1. Display user an error message and return to step 1

### Group Collaboration System

A close up of a map

Description automatically generated

Figure 7. Group Collaboration System

This use case describes the sequence of steps to be performed to create, administrate and participate in group collaborations. To create a group collaboration, the following sequence of steps would have to be performed:

1. Click on ‘group collaborations’ button
2. Select ‘create a new collaboration’ pane
3. Enter the group collaboration name in the ‘name’ field
4. Click on ‘create’ button

The alternative scenarios would arise from incorrect user input or due to internal server error and can be summarized as follows:

1. Alternative scenario point 1: The server has encountered an internal server error that prevented it from redirecting to the group collaborations page:
   1. Display user an error message and return to step 1.
2. Alternative scenario point 3: The group collaboration name is invalid:
   1. The group collaboration name is either too short or too long:
      1. Show user an error message displaying the minimum and maximum length of group collaboration name
      2. Return to step 3
   2. The group collaboration name contains invalid characters:
      1. Show user an error message displaying the list of invalid characters
      2. Return to step 3
3. Alternative scenario point 4: The server has encountered an internal server error that prevented it from creating a new group collaboration:
   1. Show user an error message and return to step 1

To add new users to the group collaboration, the following sequence of steps would have to be performed:

1. Click on ‘group collaborations’ button
2. Select the group to add users to from the list of group collaborations that user is a member of
3. Click on ‘add members’ button
4. Find the user to add to group collaboration in the list of users the user has a private collaboration with
5. Click on ‘add to the group’ button

The alternative scenarios would arise due to internal server error and can be summarized as follows:

1. Alternative scenario point 1: The server has encountered an internal server error that prevented it from redirecting to the group collaborations page:
   1. Display user an error message and return to step 1.
2. Alternative scenario point 2: The server has encountered an internal server error that prevented it from retrieving the list of groups the user is participating in:
   1. Display user an error message and return to step 1.
3. Alternative scenario point 4: The server has encountered an internal server error that prevented it from retrieving the list of user’s private collaborators:
   1. Display user an error message and return to step 1.
4. Alternative scenario point 5: The server has encountered an internal server error that prevented it from sending the invitation to join the group:
   1. Display user an error message and return to step 1.

To add a PDF Document as part of group exam preparation, the following sequence of steps would have to be performed:

1. Click on ‘group collaborations’ button
2. Select the group to upload a PDF document to
3. Select ‘add PDF’ pane
4. Click on ‘upload file’ button
5. Select the PDF file from the file system
6. Click ‘add’ button
7. Wait until PDF processing is complete
8. Review the extracted exam questions:
   1. If the extracted exam questions are correct:
      1. Click on ‘confirm’ button
   2. If the extracted exam questions are incorrect:
      1. Correct the exam questions in the corresponding fields
      2. Click ‘confirm’ button

The alternative scenarios would arise due to incorrect file uploaded or due to an internal server error that prevented it from fulfilling the request and can be summarized as follows:

1. Alternative scenario point 1: The server has encountered an internal server error that prevented it from redirecting to the group collaborations page:
   1. Display user an error message and return to step 1.
2. Alternative scenario point 2: The server has encountered an internal server error that prevented it from retrieving the list of groups the user is participating in:
   1. Display user an error message and return to step 1.
3. Alternative scenario point 6: The server has encountered an internal server error that prevented it from uploading the file or the uploaded file is not a PDF file:
   1. If the server encountered an internal server error that prevented it from fulfilling the request:
      1. Display user an error message and return to step 4.
   2. If the uploaded file is not a PDF file:
      1. Display user an error message and notify them that the only accepted file format is PDF
      2. Return to step 4
4. Alternative scenario point 7: The server has encountered an internal server error that prevented it from processing the PDF file:
   1. Display user an error message and return to step 4
5. Alternative scenario point 8.A.I and 8.B.II: The server has encountered an internal server error that prevented it from finishing the PDF processing:
   1. Display user an error message and return to step 4

## Software and Hardware Requirements

### Overview

The application itself will be split into front-end, back-end and persistent data management. The front-end is concerned with user interaction and will be implemented using Angular framework. The back-end is concerned with user requests that cannot be implemented on the front-end or are unfeasible to implement on the front-end, such as database queries, and will be implemented using the Spring Boot framework. A MySQL Database and the native file system will manage the persistent data. As the development progresses, based on the time constraints, team might migrate part or the whole architecture to cloud – either Azure or AWS.

### Front-end

To develop the Front-end the team has chosen to use the Angular Framework. Angular is implemented using TypeScript, a superset of JavaScript. It uses a component-based architecture, with each element of the UI and its’ functionality encapsulated into reusable components. This allows for greater reuse of elements and makes the code more testable and maintainable. The components are written in HTML and TypeScript and then compiled to JavaScript before being interpreted by the browser. Although Angular is a relatively new framework with a steep learning curve, and despite the teams’ unfamiliarity with it, it was chosen because it allows the development of a standalone front-end, which provides more flexibility on the back-end.

Development in Angular is done using TypeScript, a superset of JavaScript, with program logic separated in modules and bound to HTML templates and CSS, thus enabling great functionality and flexibility, since both HTML and TypeScript are compiled together. Since Angular is a relatively new framework to the team, a certain timeframe will be dedicated so the team members can get familiar with it. When choosing a front-end framework, the team has faced several options – either using a stand-alone front-end framework, like Angular, or using a back-end framework that incorporates front-end capabilities with template processing, such as Laravel with its Blade Templates or Spring Boot with JSP. A stand-alone framework has been chosen as a preferable option since it is much easier to update the architecture with a stand-alone front-end framework. For example, if the team decides to migrate part of the architecture to AWS, it will be possible to host a webpage straight from the S3, without any changes to the server or to the front-end. On the other hand, it would be not possible to do the same were front-end part of a templatized back-end, since front-end would be essentially a part of the back end. Finally, a stand-alone framework, and Angular in particular, encourage decoupling and are usually more powerful than templatized back-end front-end, because it is possible to embed TypeScript (that is compiled into JavaScript) directly into the page. As a conclusion, the following technologies will be used for front-end development:

• Angular Framework:

* HTML
* CSS or SASS
* TypeScript
* Bootstrap
* Angular Material

### Back-end

To develop the Back end the team has chosen to use the Spring Boot Framework. Spring Boot is a micro-framework that allows developers to quickly create a Spring Application. It builds upon the underlying Spring framework by automatically configuring some of the dependencies needed. This auto-configuration can reduce the amount of code that needs to be written to get an application up and running. Spring Boot uses Java and provides a comprehensive set of libraries and modules in an easily configurable manner. Spring Boot was chosen because of the team’s familiarity with Java and although some initial effort will be required at the beginning, the benefits provided by the auto-configuration will allow for the core of the system to be developed quite quickly.

In addition to that, Spring Boot eases database management with automatic entity creation and update using Hibernate. Since Spring Boot is a relatively new framework to the team, a certain timeframe will be dedicated so the team members can get familiar with it. For the back end, the team faced a choice of which language to use – there are numerous for back-end development, such as Java, C#, PHP or Python. It was decided that Java-based framework would be preferential: all of team members are proficient in Java, and Java remains the predominant language for the server-side programming language (TIOBE Software BV, 2019) While C# and Python gain pace, they were not chosen due to team member’s unfamiliarity with them. PHP has been ruled out, since it has been on a nearly constant decline is popularity for the last fifteen years. When decided what language to use, there was still a choice between the various Java frameworks for back-end development. Spring Boot has been chosen due to its relatively mild learning curve (unlike its predecessor, Spring). The technologies used for back end are:

* Spring Boot:
  + Java
  + Spring Initializr
  + Spring Security and Encryption
* Hibernate with MySQL Connector

### Persistent Storage

MySQL will be used for persistent storage. It was chosen because of the simple to use interface and uncomplicated setup and for the teams’ knowledge and experience of working with it. The user information, such as login credentials and profile data will be encrypted and stored in the database.

Files that must be stored, such as PDFs and user images, will be stored in either Azure Blobs and Azure Files or AWS S3. MySQL database has been chosen due to the fact that it is easy to setup and has a simple user interface, as well as team familiarity with MySQL. Even though MySQL database will be used, little or no SQL scripting will be written, because Spring Boot Server will be managing it without the need for the team to interfere. The team has yet to review its options regarding Azure and AWS – while some team members have limited experience with AWS, pricing might be an issue; Azure, on the other hand, provides better options in term of price, while the services themselves are nearly identical to what team might need (storage being of main interest, however, additional services, such as hosting, databases and request processing are being considered as well). Also, it is worth pointing out, that native file system is likely to be used initially for testing purposes. The full list of technologies is:

* MySQL Database Server
* Azure Blobs and Azure Files OR AWS S3

### PDF Text Extraction

Text contained in a Portable Document Format (PDF) file is represented by stream of character glyphs denoted by a String Object and the associated information used to display the text correctly on the page. This information, such as the font and location, is then used to paint the glyphs in the correct size and shape in the specified location (Adobe 2006).

To extract text from uploaded PDF files the team will use the Apache PDFBox open-source library for java. PDFBox provides many features for working with PDF documents including:

* Extracting Unicode text from PDF files
* Splitting and merging PDF files
* Saving PDF files as images

The PDF specification also allows for images to be displayed on the page. In a case where a PDF file is created by simply scanning an existing document, the document may be represented by a single image, and therefore contain no text Strings. As there is no actual text data in such a document, PDFBox will not be able to extract the text. Therefore, another approach will be needed.

Extracting text from a PDF Image

To extract text from an image, the team will use Optical Character Recognition (OCR). OCR is a widespread technology used to recognise text in an image, with many uses ranging from converting large volumes of printed material such as old newspaper articles into machine readable text, to vehicle number plate recognition.

Tess4J will be used to extract text form images in a machine-readable form. Tess4J provides a Java Native Access (JNA) wrapper for the Tesseract OCR API, which was first developed by Hewlett-Packard, and since being open sourced in 2005, is now developed by Google. The images are extracted from the PDF and then processed using OCR to output a String of machine-readable text. Tess4J can be tested using JUnit and provides support for PDF documents using PDFBox.

## Testing

For back-end testing, JUnit will be used for unit testing; for front-end, Jasmine will be used for unit testing and Angular Protractor will be used for E2E testing. It is team’s intention to use test-driven development approach, so as to spot flaws in the system and ease debugging and tracing, thus, the application will be well-defined and tested at the end of every iteration. These technologies ship with the frameworks the team will be using. The full list of testing libraries used is:

* Back-end:
  + JUnit
* Front-end:
  + Jasmine (Unit Testing)
  + Angular Protractor (E2E Testing)

## Summary

Summarise the chapter.

# System Design

## Overview of System Architecture

The development approach the team has chosen is a three-tier architecture in the context of Model-View-Controller design pattern. As seen in the Figure 8, the application is divided into the presentation layer which is developed using the Angular Framework; the logic layer which is developed using Spring Boot Framework; and the persistence layer, which is the MySQL database.

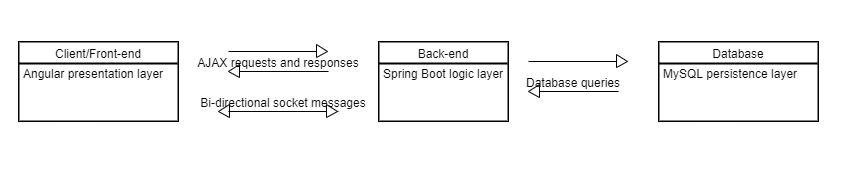


Figure 8. High-level architecture diagram

As seen on the diagram, the client, which is the Angular application, communicates with server via HTTP AJAX requests, as well as via duplex web socket channels.

## Design Patterns

There are several reasons the team has chosen the three-tiered design, as well as Model-View-Controller-Pattern. To some extent, the three-tier approach has been enforced by the technology itself, namely, Angular, which is a front-end framework only, and, as such, has to be in a separate layer. However, the team strongly believes that the multi-layered approach offers strong advantages by itself, such as:

* Complete separation of concerns - the application layer is concerned with presentation and control over the way user interacts with the application, while the logic layer is concerned with how the data is retrieved, processed and stored.
* Modularization – the application logic and the layer logic are independent of each other, thus the changes in rules or inner representation will have either little or no impact on the other side.
* Data representation – the same user data might be represented differently, using either TypeScript interfaces or Spring Boot entity classes, thus, the data that might be relevant for the logic layer purposes can be omitted completely on the application layer and vice versa.

The team strived to design the application in such a way to adhere to the Model-View-Controller pattern as much as feasible to do so; regarding the application layer, the MVC pattern adherence is absolute, with Angular HTML templates being the views, Angular TypeScript files being the controllers and TypeScript interfaces being the data. The MVC pattern in the context of Spring Boot in this particular project is slightly more abstract – since there are no views served from the logic layer per se, one might get an impression that there is no view layer on the back-end. However, when defining view as not what is seen, but what is retrieved, represented and manipulated upon, the Spring Boot controllers would act as views, since they expose the inner logic for application layer to query, retrieve data and act upon it; in addition to that, the Spring Boot service classes act as controllers, and Spring Boot entity classes act as models.

## Project Structure

Since the application consists of the multiple layers that are completely separate from each other, the layers – application layer and logic layer – will be discussed separately.

### Application Layer Structure

The Angular applications are organized into a set of one or more *modules*, with single mandatory module being the app.module.ts file, from which the application is bootstrapped. Modules reside in folders, usually with the same name as the module itself, and contain *components* which are a collection of four files – a HTML template, a TypeScript file, a CSS file and a unit test file. In addition to components, the Angular application has *services* which are injected into the components using *Dependency Injection*. Service classes are used to provide a common functionality for a group of components. Finally, the TypeScript provides *interfaces* as a language feature, whereby interfaces are a collection of data fields used to enforce adherence to the strict typing.

When organizing the application layer components into modules, the team was considering the feature set/targeted user experience the component is, however, it is not a strict rule and exceptions were made, explicitly mentioned further on. In general, the module would consist of the following:

* Module file – a file containing module definitions, declaration of components that are part of the module, imports of other modules and service provides for services that cannot be injected using the normal dependency injection.
* Routing module – a special module that contains routes to components to rout to.
* Components – module components that are routed to and that are not meant to be children of other components.
* Partial Components – module components that are meant to be included as part of other components (note that partial components may be a part of other partial components).
* Services – module services used by the components of the module (note that components from different modules can use each other’s services).
* Interfaces – module interfaces used to describe data.
* Resolvers – specific kind of module services used to pre-fetch the data for the component for the specific route. Components can have any number of resolvers; resolvers are guaranteed to have required data fetched before the component is rendered.

The modules of application layer, listed in alphabetical order, are as follows:

* **Group Collaboration Module** – contains components and services that are related to the group collaboration functionality, such as components displaying the user-groups they are part of, group creation components, etc., as well as the routing module with routing paths and resolvers.
* **Paper Module** – contains components and services that are related to the exam paper upload facilities. This module is an exception, since the paper related functionality is part of the group collaboration exclusively; however, the paper upload, processing and text extraction resulted in quite a large number of related components and services, and the team decided to move it to the separate module. It does not contain the usual components directory, but rather an export components one, since the components are meant to be part of another module components. There is no routing for this module, however, there are Paper-related resolvers, which are contained within the Group Collaboration Module.
* **Private Collaboration Module** – contains components and services that are related to the group collaboration functionality, such as display the user collaborators, the existing collaboration requests, etc., as well as the routing module with routing paths and resolvers.
* **Query Module** – contains components and services that are related to the query and forum functionality, such as the forum itself, query creation, retrieval, etc., as well as the routing module with routing paths and resolvers.
* **User Module** – contains components and services that are related to the User entity, such as user home page, user settings, etc., as well as the routing paths and resolvers.
* **Utility module** – contains components, services and routing paths that are either meant to be included in the root of the application, or do not fall into any other specific functionality category.
* **App Module** – the root module, which bootstraps the application, reads in the routing information from other modules and it contains the root routing path.

In addition to the aforementioned modules, the project contains configuration files for the framework itself, as well as the testing configuration, the linting configuration, the package manager configuration and the environments.

### Logic Layer Structure

Spring boot is a Java-based framework, and, as such, it organises an application using packages. The Spring Boot framework separates groups of classes according to their intended purpose which are:

* Entities, annotated with the @Entity annotation, are the data classes that are persisted in the database.
* Repositories, annotated with the @Repository annotation, expose the data layer functionality and are injected via dependency injection. Repositories are assigned to a specific Entity class and its primary key. Normally, each Entity requires its own repository, however, in the case of class inheritance, a tree of classes can have a single base class repository, being subject to the inheritance persistence strategy .
* Services, annotated with the @Service annotation, are classes that are injected via dependency injection and are meant to provide a common functionality to the controllers and/or other services.
* Controllers, annotated with the @Controller or @RestController annotation, are the classes that expose the logic layer API to the application layer.

Whereas in the application layer the module division criteria was based on the user experience components , in the logic layer, the classes are grouped in packages according to the data they operate upon, as well as the overall functionality, being further divide into sub-packages based on their intended purpose; in general, a package would consist of the following sub-packages:

* Controllers – controllers related to the functionality.
* Entities – entities that are operated upon.
* Services – operate upon entities.
* Repositories – used to for the persistence layer communication.
* Exceptions – exceptions that are related to the particular error condition when working upon the functionality or entities.

The root package is **project.webcollaborationtool.** The packages of the logic layer, listed in alphabetical order, are as follows:

* **Collaboration package** is subdivided into the following sub-packages:
  + **Group collaboration package** – a package containing classes that represent an abstraction of the user group interactions.
  + **Paper** **package**– a package containing classes used for exam paper upload, processing, text extraction.
  + **Private collaboration** – a package containing classes that represent an abstraction of the interaction between two users.
  + **Request package** – a package containing classes that are used to process user requests for private or group collaboration.
  + **Thread package** – a package containing classes used for the chat functionality.
* **Notification package** – a package containing classes that are used for receipt, processing and delivery of the user notifications for using the web sockets.
* **Query package** – a package containing classes that represent an abstraction of the query functionality.
* **User package** – a package containing classes that represent an abstraction of sign-up, login and user-representation functionalities.
* **Utility package** is subdivided into:
  + **Composite Keys package** – a package containing composite keys classes for complex entity relationships.
  + **Configurations package** – a package containing either configurations that have to be provided for advanced functionalities, such as Web Socket configuration, or configurations that override the default functionality.

## Graphical User Interface

Provide an overall description of the user interface (you should include wireframe(s)). Identify the graphical elements you use and what is their purpose.

## Data Permanency Mechanism

Describe what entities are included in your database; use entity diagram to support this.

## Optimisation (if any)

Describe the approach taken to optimise the performance of the application; include here concurrency/parallelisation frameworks.

## Summary

Summarise the chapter.

# Implementation

## Overview

Overview of the implemented components. The title and order of the next sections should be customised, depending on the architecture chosen.

## Front-End/View/Presentation Layer Implementation

How you implemented the logic of the front end: describe the algorithms (you can also include code screenshots) to implement the functionalities.

## Middle/Controller/Application Layer Implementation

How you implemented the logic of the middle component: describe the algorithms (you can also include code screenshots) to implement the functionalities.

## Back-End/Model/Data Layer Implementation

How you implemented the back end: describe the algorithms (you can also include code screenshots) to implement the functionalities.

## Summary

Summarise the chapter.

# Testing and evaluation

## Overview

Describe the approaches you took to testing the application: unit testing, integration testing, etc.

## Unit Testing

What classes/scripts/functionalities (I hope all) you have tested individually and how.

## Integration testing

How was the application tested as a whole; include here how the classes/scripts/functionalities interact, e.g. how do they pass data to other one another, etc.

## Debugging

Identify possible logical errors and how did you deal with these cases; for example, number formatting associated with wrong user input or wrong password or null pointers associated with the inability to create appropriate objects.

## Exception Handling

How did you deal with situations that your application cannot control; for example, IO exceptions associated with file reading/writing.

## Evaluation (if you have time)

How did you evaluate the application? Did you ask people to test it? Did they provide feedback/identify further bugs? Etc.

## Summary

Summarise the chapter.

# Conclusions and Further Work

Restate the aim of the project. Did you answer the research questions? Did you achieve the aim and all the objectives? What are the strengths? What are the weaknesses? Can you suggest how to address the weaknesses?

# References

# Appendix A: Application Link(s)

Provide a link to the github repository and, if deployed online, the link to the website.

# Appendix B: Any other extras…