BSc in Computer Science

School of Computing, Science and Engineering

Graphical user interface

Description automatically generated

Final Year Project Dissertation - 2023

Project: Codenemy

BSc (Hons) Computer Science (Level 6)

Author: Chaudhary Samir Zafar

Student Id: @00546317

User Id: AEE363

Supervisors: Stefan Pletschacher & Shadab Mashuk

Word count: 9998

Excluding Table of contents, tables of figures, acknowledgements, chapters’ 1,2 & 3, bibliography, and appendix

**Acknowledgements**I would like to take this opportunity to thank Stefan Pletschacher and Shadab Mashuk for the help they have provided throughout the development of this project. The advice and feedback received allowed me to produce a robust and complete project successfully.

**Abstract**

This project attempted to create a web application that allows users to solve coding challenges alone or against other users. The project aimed to provide a fun and collaborative way for developers to prepare for their software engineering technical interviews. Technical interviews consist of a coding challenge and the interviewee needs to solve the challenge within a given time.

Using various full-stack software engineering techniques the project was developed successfully with all of its required requirements implemented. The project consisted of three main stages, specification and design, development and implementation and testing and analysis. The project has successfully been deployed to the web and is available for public use (<https://www.codenemy.com>).

Through usability testing, we can conclude that the web application produced was interactive, fun and easy to use for developers who wanted to improve their technical ability. Test users claimed the application was beneficial and it served its purpose. The results of this study imply that the gamification of coding boosts an individual's learning journey and makes them better candidates in the eyes of an employer.

Overall, the web application produced is a minimum viable product and users were satisfied with the application. The application offers a fun and interactive way to prepare for technical interviews. However, the application has plenty of areas that can be extended. Users who have tested the application gave various suggestions that would extend and improve the website. Due to time restrictions, the website falls short when compared to competitors although it offers unique features.

**Contents**

[Chapter 1: Introduction 9](#_Toc132414890)

[1.1 – Motivation 9](#_Toc132414891)

[1.2 – Project Aim 9](#_Toc132414892)

[1.3 – Objectives 9](#_Toc132414893)

[1.4 – Project Approach 10](#_Toc132414894)

[1.5 – Summary of this dissertation 10](#_Toc132414895)

[Chapter 2: Literature Review 11](#_Toc132414896)

[2.1 - Introduction 11](#_Toc132414897)

[2.2 - Competition & Performance 11](#_Toc132414898)

[2.3 - Competition & Learning 11](#_Toc132414899)

[2.4 - Competitive Coding 12](#_Toc132414900)

[2.5 - Related products 13](#_Toc132414901)

[2.5.1 - Leetcode 13](#_Toc132414902)

[2.5.2 - HackerRank 14](#_Toc132414903)

[2.5.3 - CodeCollab 14](#_Toc132414904)

[2.5.4 - Summary 14](#_Toc132414905)

[Chapter 3: Methodology 16](#_Toc132414906)

[3.1 - Introduction 16](#_Toc132414907)

[3.2 - Research Methods 16](#_Toc132414908)

[3.3 - Development methodology 16](#_Toc132414909)

[3.4 - Programming Languages & Frameworks 17](#_Toc132414910)

[3.4.1 - Front end 17](#_Toc132414911)

[3.4.2 - Back end 17](#_Toc132414912)

[3.5 - Development Techniques 18](#_Toc132414913)

[Chapter 4: Requirements Specification and Design 20](#_Toc132414914)

[4.1 – Introduction 20](#_Toc132414915)

[4.2 - Requirements 20](#_Toc132414916)

[4.2.1 – Non-Functional Requirements 22](#_Toc132414917)

[4.2.2 - Functional Requirements 22](#_Toc132414918)

[4.2.3 – User Stories 25](#_Toc132414919)

[4.3 – Design 27](#_Toc132414920)

[4.3.1 – Introduction 28](#_Toc132414921)

[4.3.2 – System Architecture 28](#_Toc132414922)

[4.3.3 – User Interface/Front Design 28](#_Toc132414923)

[4.3.4 – Backend Design 35](#_Toc132414924)

[4.3.5 – Database design 36](#_Toc132414925)

[4.4 – Summary 37](#_Toc132414926)

[Chapter 5 – Development and Implementation 38](#_Toc132414927)

[5.1 – Introduction 38](#_Toc132414928)

[5.2 – Database 38](#_Toc132414929)

[5.3 – Log In & Register 38](#_Toc132414930)

[5.3.1 – Front end 38](#_Toc132414931)

[5.3.2 – Back end 39](#_Toc132414932)

[5.3.3 – Problems encountered 43](#_Toc132414933)

[5.4 – User profile management 43](#_Toc132414934)

[5.4.1 – Front end 43](#_Toc132414935)

[5.4.2 – Back end 45](#_Toc132414936)

[5.5 – Library of coding challenges 45](#_Toc132414937)

[5.5.1 – Front end 45](#_Toc132414938)

[5.5.2 – Back end 46](#_Toc132414939)

[5.6 – Implementing a coding environment 47](#_Toc132414940)

[5.6.1 – Front end 47](#_Toc132414941)

[5.6.2 – Back end 49](#_Toc132414942)

[5.6.3 – Challenges to complete 49](#_Toc132414943)

[5.7 – Allowing users to solve coding challenges alone. 49](#_Toc132414944)

[5.7.1 – Front end 49](#_Toc132414945)

[5.7.2 – Back end 49](#_Toc132414946)

[5.7.3 – Problems encountered 53](#_Toc132414947)

[5.8 – Allowing users to solve coding challenges against other users. 54](#_Toc132414948)

[5.8.1 – Front end 54](#_Toc132414949)

[5.8.2 – Back end 57](#_Toc132414950)

[5.9 - Summary 57](#_Toc132414951)

[Chapter 6 – Testing and Analysis 58](#_Toc132414952)

[6.1 – Introduction 58](#_Toc132414953)

[6.2 – CI/CD pipeline 58](#_Toc132414954)

[6.3 - Regression testing 60](#_Toc132414955)

[6.4 – Unit Testing 61](#_Toc132414956)

[6.5 – Usability testing 63](#_Toc132414957)

[6.5.1 – Questionnaire responses 64](#_Toc132414958)

[6.5.2 – Analysis and Retrospective 68](#_Toc132414959)

[6.6 – Summary 68](#_Toc132414960)

[Chapter 7 - Critical Evaluation 69](#_Toc132414961)

[7.1 – Review of the project against its objectives 69](#_Toc132414962)

[7.2 – Review of the plan 69](#_Toc132414963)

[7.3 – Review of Feedback and Product 70](#_Toc132414964)

[7.4 – Lessons Learnt 70](#_Toc132414965)

[7.5 – Summary 71](#_Toc132414966)

[Chapter 8 – Conclusion 72](#_Toc132414967)

[8.1 – Project Overview 72](#_Toc132414968)

[8.2 – State-of-the-art comparison 72](#_Toc132414969)

[8.3 – Future work 72](#_Toc132414970)

[8.4 – Legal, social, ethical and professional issues 73](#_Toc132414971)

[8.5 – Summary 73](#_Toc132414972)

[Bibliography 74](#_Toc132414973)

[Appendices 76](#_Toc132414974)

[Appendix A - Project Proposal 76](#_Toc132414975)

[Aim 76](#_Toc132414976)

[Motivation 76](#_Toc132414977)

[Background 76](#_Toc132414978)

[Who will benefit? 76](#_Toc132414979)

[Objectives 77](#_Toc132414980)

[Development Requirements 77](#_Toc132414981)

[Methodology 78](#_Toc132414982)

[Gannt Chart 79](#_Toc132414983)

[Appendix B - Project logbook 80](#_Toc132414984)

[Final Year Project – Logbook 80](#_Toc132414985)

[Trimester 1 80](#_Toc132414986)

[Christmas Holidays 84](#_Toc132414987)

[Trimester 2 87](#_Toc132414988)

[Easter Holidays 94](#_Toc132414989)

[Reflection 94](#_Toc132414990)

Figure 1 – The structure of conducted user studies (Fischer et al., 2021). 12

Figure 2 – Ratio of games won over the desire for competition (Fischer et al., 2021). 12

Figure 3 - Ratio of games won over motivation to solve the programming tasks (Fischer et al., 2021). 12

Figure 4 - Sprint breakdown 17

Figure 5 - Architecture analysis summary 18

Figure 6 - Target architecture 19

Figure 7 – Use case diagram 20

Figure 8 – Activity Diagram 21

Figure 9 – Kanban board in Jira 26

Figure 10 – Subtasks of each user story 27

Figure 11 - Index/home page design ………………………………………………………………………………………………………………….. 26

Figure 12 - Log in/Register page design 27

Figure 13 - Problems page (displaying a library of coding challenges) when logged in design 28

Figure 14 - Problems page (displaying a library of coding challenges) when not logged in design 28

Figure 15 - Coding environment design 29

Figure 16 - Challenge lobby results from design 30

Figure 17 - Challenge lobby design 30

Figure 18 - Challenge page design 30

Figure 19 - User profile management page design 31

Figure 20 – Backend Architecture Design 35

Figure 21 – Database Entity Relationship (ER) diagram 36

Figure 22 – LogIn&Register page (State 1) 38

Figure 23 – LogIn&Register page (State 2) 38

Figure 24 – LogInAndRegisterPage component code 39

Figure 25 – Backend Diagram 39

Figure 26 – Overview of SpringSecurity 40

Figure 27 – AuthenticationFilter code 41

Figure 28 – AuthorizationFilter code 42

Figure 29 – User Profile Management UI. 43

Figure 30 – Input validation 44

Figure 31 – form validation 44

Figure 32 – change user details endpoint in the authentication controller. 45

Figure 33 – The user interface of the problems page 45

Figure 34 – two lists that store a list of challenges. 45

Figure 35 – Mutated problem list is set to the filtered original list. 46

Figure 36 – Rendering each Challenge in 46

Figure 37 – Backend diagram updated (version one). 46

Figure 38 – getAll endpoint in problem service. 47

Figure 39 – Coding environment (State 2) 47

Figure 40 – Coding environment UI (State 1) 47

Figure 41 - Components of coding environment user interface 48

Figure 42 – Backend diagram updated (version two). 49

Figure 43 – Structure of each Service 50

Figure 44 – Compiler Controller 50

Figure 45 – Factory pattern 51

Figure 46 – CompilerService interacting with CompilerServiceFactory. 52

Figure 47 – Abstract interface that is implemented by all language compiler services. 52

Figure 48 – CompilerServiceFactory implementation 52

Figure 49 – Docker script to build the docker image. 53

Figure 50 – Challenge page UI (Starting point of flow). 54

Figure 51 – Challenge lobby UI (second point of flow). 55

Figure 52 – Challenge solving problem UI (third point of flow). 55

Figure 53 – Challenge results (four point of flow). 55

Figure 54 – createLobbyHandler code. 56

Figure 55 – joinLobbyHandler code. 56

Figure 56 – WebSocket Diagram 56

Figure 57 – Backend Diagram updated (Version three). 57

Figure 58 – Message endpoints for web socket server. 57

Figure 59 – Gitlab pipeline configuration script 58

Figure 60 – Gitlab pipeline overview 59

Figure 61 - Pipeline run breakdown 59

Figure 62 – Pipeline status on Gitlab 59

Figure 63 – Pipeline test report 60

Figure 64 – Regression testing overview 60

Figure 65 – Regression testing process 61

Figure 66 – TDD framework 61

Figure 67 – Application test package 62

Figure 68 – Example unit tests 62

Figure 69 – Application Unit Tests 63

Figure 70 - 90%+ code coverage 63

Figure 71 – Question 1 response graph 64

Figure 72 – Question 2 response table 64

Figure 73 – Question 3 response graph 65

Figure 74 – Question 4 response graph 65

Figure 75 – Question 5 response graph 66

Figure 76 – Question 6 response table 66

Figure 77 – Question 7 response graph 67

Figure 78 – Question 8 response table 67

# Chapter 1: Introduction

## 1.1 – Motivation

One motivation point behind this project is the value it will bring to the Computer Science industry, especially for junior developers. It will allow them to develop their problem-solving ability whilst having fun. Junior developers can make use of this skill when applying for jobs.

Although it was very possible to gain a lot of web development experience during the first and second years of university, the choices of learning new frameworks and languages were limited. I believe this project will be complex therefore it will challenge my core computing knowledge and coding skills. This project will also provide opportunities to learn widely used industry-standard front-end frameworks and libraries. Thus, boosting overall career opportunities and helping to improve my skill set.

## 1.2 – Project Aim

This project aims to build a web application that creates a fun and challenging experience for software engineers that are preparing for potential technical interview questions. The engineers will be able to compete against other engineers in solving algorithmic coding challenges. The challenges will consist of a description, some template code, an input, and the desired output. The engineers will need to add to the template code to get the desired output. The engineer that gets the desired output and passes all test cases first will win the competition. The project will also allow users to do this alone, similar to what Leetcode (<https://www.leetcode.com>) offers.

## 1.3 – Objectives

**Objective 1: Conduct background research on how similar web applications work & make a list of required features for my project.**

* Research how to compile code on a website to perform code execution in the safest manner possible.
* Research similar web applications to understand the approach they may be taking.
* Research web sockets to understand how they work and why I will need them.

**Objective 2: Develop a data-driven web application that allows users to practice code and compete against each other.**

* Design a friendly UI while focusing on accessibility & usability.
* Create a quality database for the project that cannot be simplified any further.
* Implement a quality UI that is responsive to changes in screen size.
* Develop a user profile system that allows users to register, log in, edit, and update their details.
* Develop a feature that allows users to compete against each other.
* Create an algorithm that helps match users of the same ability.

**Objective 3: Perform testing on the application covering common use cases, and edge cases and generate a test report.**

* Research and identify suitable testing frameworks for chosen technologies.
* Carry out a wide range of testing on usability and accessibility (front end).
* Perform unit testing (back end).
* Perform usability testing.

**Objective 4: Implement DevOps such as CI/CD for deployment & ensure the pipeline generates test reports and successfully deploys repositories.**

* Create CI/CD pipeline that generates test reports and deploys the repositories.
* Use Docker to create an image that stores the code-compiling API.

## 1.4 – Project Approach

The project will be developed using the Agile methodology. Before the development phase begins, research will be conducted. Similar products should be analysed to set a minimum standard for this project and to identify weaknesses, that this project should aim to fill. Starting, the sprints will consist of research and requirement elicitation, from there on each sprint will be focused on working towards a requirement of the project. Requirements will be converted to user stories and they will have child user stories to break them down. A Kanban board will be created to help track the progress of user stories. Sprints will last a week and retrospectives will be held to reflect on sprints and how future sprints can be improved and what tasks need to be prioritised in future.

## 1.5 – Summary of this dissertation

This dissertation aims to break down the process of building this project. Topics such as requirement specification and design, development and implementation and testing and analysis will be discussed. This dissertation aims to give a comprehensive understanding of the process of development for the project. The approach and processes to build the project will be evaluated to identify any weaknesses that can be improved on in the future.

# Chapter 2: Literature Review

## 2.1 - Introduction

Coding challenges are usually given to a potential employee by an employer during the application process. They allow employers to assess the employee’s coding, analysis, and problem-solving ability. In most cases, the challenge is time constrained and is completed on a specific platform.

The purpose of this paper is to review recent research into elements related to the gamification of coding challenges.

Conducting this research will prove beneficial as it will contribute to building a robust and well-rounded web application. The research will help identify key elements that are needed in the application.

## 2.2 - Competition & Performance

Research has been conducted to identify the influence of competition on physical performance. The method that was used was fourteen participants who completed a 2000-metre familiarization time trial on a Velotron cycle ergometer. They then completed an additional time trial where they competed against a computer-generated image which was their own best performance. “Participants were able to complete the 2000-m cycling time trial faster when they believed that they were competing in a head-to-head competition than when they exercised alone” (CORBETT et al., 2012). The findings allow us to conclude that competition positively affects physical performance.

Another study has also researched the effects of competition alone on task performance. The method of this study was 347 students took part in a brainstorming task. The concept of the task was to come up with uses for a cardboard box. There were two conditions for the brainstorming task. An experimental condition where the student was told they’d be competing against others and the high score was held by a made-up individual called Alex which they had to beat. A controlled condition where the student was instructed to come up with as many ideas as possible within the time limit. The results of this study showed that adding the element of competition improved both quantity and quality (Landers et al., 2019).

From the two studies, we can conclude that competition improves physical and nonphysical performance. We can also assume that competition has a positive effect on an individual’s intrinsic motivation.

## 2.3 - Competition & Learning

A study by Atilim University researches the effects of competition on learning in games. 142 undergraduate students participated in the study. Their age ranged between 20 – 28 years. All students were enrolled on a course that is related to computer engineering. The method that was used for this study was the students would take a test before playing a game. Then the participants would play a game. The game was designed to present conceptual database modelling concepts. The content of the game was the user would roll two dice, then be presented with a true and false question, and then earn points if their response is correct. Half of the students played a competitive version of the game where the student could view all of the current player's scores, and the other half played in isolation with no information about other students’ scores. After the students participated in the game, they sat a post-test. “The results indicated that the competition group performed better in the post-test than the control group” (Cagiltay et al., 2015). The study also conducted an identical test to compare the motivation of participants in the two groups and similarly, the competition group had higher motivation. We can evaluate that competition has a positive effect on learning & motivation.

Near East University in North Cyprus researched the effects of a gamification-supported flipped classroom model on the attitudes and opinions regarding game-coding education. The flipped classroom model is the idea of freeing up lecture/classroom time by giving students information before class, so it frees class time for activities. 35 classroom students went through a period of 14 weeks where the structure was adhering to the supported flipped classroom and gamification of coding. The study made use of quantitative and qualitative methods. Regarding the qualitative methods, pre and post-tests were applied to measure the attitude of students towards coding. The quantitative method used interviews to retrieve in-depth information. “The student’s eagerness to learn coding increased. The results also showed a shift in the teacher’s beliefs. Before the study, the eagerness of the teacher-candidate students regarding the use of computer games in coding training was at a medium level (x=2.83) and after the coding training incorporating the activities in the gamification-supported flipped classroom, their eagerness increased to a high level (x=4.06)” (Ã–zer et al., 2018) The experiment has shown that gamification supported flipped classroom model regarding game-coding education has had a positive effect on both students and teachers.

## 2.4 - Competitive Coding

The Institute of Information in Munich, Germany conducted a study that researched the effects of competitive coding games on novice programmers. The method used by this study was two groups of Computer Science students participated. The only difference between the groups was the programming language they preferred. The students followed a step-by-step process as shown in *Figure 1*.

Table

Description automatically generated

Figure – The structure of conducted user studies (Fischer et al., 2021).

The multiplayer phase consisted of two students going head-to-head to solve a problem in code. The first to solve the problem wins. The single-player phase was the same but in isolation. The results of this study revealed that multiple factors have a positive linear relationship with winning.

Chart, scatter chart

Description automatically generated

Figure – Ratio of games won over desire for competition (Fischer et al., 2021).

Chart, scatter chart

Description automatically generated

Figure - Ratio of games won over motivation to solve the programming tasks (Fischer et al., 2021).

In *Figure 2* in the two right quadrants we can see - “among the participants who won more than 50% of their multiplayer games, there are more participants who like to compete with others than participants who do not.” (Fischer et al., 2021).

From *Figure 2* and *Figure 3,* we can see that both willingness to compete and motivation to solve problems are positively affected by winning competitions. The results of the study also support that winning had a positive impact on calmness and confidence whilst competing. We can also conclude that participants who didn’t win too often preferred the single-player mode.

The study states that it could not come to a definitive conclusion of whether competition has positive or negative effects on a programming student's learning experience. The study has highlighted a few elements that they missed and suggests a future platform includes them. Elements such as:

* Rating system for challenges and solutions
* Possibility to comment on solutions
* Inclusion of more game elements such as:
  + Experience points
  + Badges

A variety of literature sources have been published on the element of gamification, and competition. They dwell on the effects of competition on performance whether that be physical or mental. A specific set of the literature focused on adding the competition element to coding & education related to coding. From the information gathered, we can evaluate the pros and cons of the proposed application.

The benefit of the proposed application has been highlighted. The application will increase the intrinsic motivation of participants who compete against each other. Not only will the participants be more willing to complete coding challenges but the application will also accelerate their learning journey of programming. If we evaluate all of the benefits the application will bring to a participant then we can assume that after using the application, participants will be stronger candidates in the eyes of an employer. Some extra features have also been identified. They can help extend the reach of the proposed application. For example, one study has shown that people who win competitions are more inclined to carry on competing. So, in this project, a feature where users solve problems offline and alone can be offered. Also, some of the studies have identified where they fell short and could have improved. This project can take these features and implement them, for example, a rating system.

## 2.5 - Related products

This section will investigate some of the top-rated applications that are related to this project to help distinguish the market gap that this project will attempt to fill.

The criteria that will be looked at for each application are the features (Pros & Cons) and this will include the user interface.

### 2.5.1 - Leetcode

(<https://leetcode.com>)

Leetcode is a prominent application that is well-known throughout the software engineering industry. It is used for preparation for technical interviews.

#### Pros

* Professional, responsive & functional user interface
* Study plans are offered to enhance specific skills such as Data Structures & Algorithms
* Top Interview Questions section
* 2000 problems available

#### Cons

* Better features are restricted to a paid subscription
  + An autocomplete feature is paid only. This could also make conditions unfair between two users.
* Not beginner friendly

### 2.5.2 - HackerRank

#### (<https://www.hackerrank.com/>)

HackerRank is another well-known application. It serves a few different purposes. It can be used to prepare for technical interviews, and it can be used by companies to identify and hire developers.

#### Pros

* Professional, responsive & functional user interface
* Preparation kits are offered for interviews
* Certification is available by sitting a test
* Problems specific to a variety of programming languages

#### Cons

* Not beginner friendly as it requires a learning curve as the challenges are based on Standard input (stdin)
* Bad reputation for having terrible problem descriptions

### 2.5.3 - CodeCollab

**(**[**https://codecollab.io/**](https://codecollab.io/)**)**

CodeCollab is an application that is dedicated solely to allowing users to collaboratively code in real-time.

#### Pros

* Professional, responsive & functional user interface
* Accurate collaborative feature in real-time
* Live web chat

#### Cons

* The collaborative feature doesn’t lock lines when editing code, so overwriting code is possible.
* Collaborated work expires in 7 days meaning you can lose it

### 2.5.4 - Summary

Below is a comparison of similar products.

|  |  |  |  |
| --- | --- | --- | --- |
| Features | Application | | |
|  | Leetcode | HackerRank | CodeCollab |
| User Interface |  | | |
| Responsive | Yes | Yes | Yes |
| Looks Professional | Yes | Yes | Yes |
| Compatibility |  | | |
| Apple Phone | Yes | Yes | Yes |
| Android Phone | Yes | Yes | Yes |
| Laptop | Yes | Yes | Yes |
| Desktop | Yes | Yes | Yes |
| Different browsers | Yes | Yes | Yes |
| Functionality |  | | |
| Supports multiple programming languages | Yes | Yes | Yes |
| Problem-Solving offline (Single player) | Yes | Yes | No |
| Problem-Solving head-to-head (Multiplayer) | No | No | No |
| Technical Interview Preparation | Yes | Yes | No |
| Collaborative coding | No | No | Yes |

*Table 1 - Summarisation of similar applications*

*Table 1* allows us to identify the gaps that exist within similar applications. For example, problem-solving head-to-head isn’t available on all platforms, similarly, collaborative coding isn’t available on every single platform. These feature gaps can be filled by the proposed application to make sure the application is well-rounded. Leetcode is the closest product to the proposed application. However, Leetcode misses out on the significant feature of multiplayer problem-solving. From *Table 1* we can also outline some minimum expectations for the proposed application. For example, compatibility with all devices, a clean and professional user interface and support for multiple programming languages. If these minimum expectations are not met, then the proposed application may fall short of the competition.

# Chapter 3: Methodology

## 3.1 - Introduction

This project is aiming to build a web application that allows users to prepare for technical interview questions through elements of gamification such as competition to make the process more fun.

To develop this application, specific questions need to be answered before commencing the development phase of the project. Questions such as:

* What’s the best research methodology for this project?
* What’s the best-suited development methodology for this project?
* What frameworks and programming languages can be used for development?
* What development techniques will be used?

## 3.2 - Research Methods

Two research methodologies will be used in a hybrid fashion. Qualitative and Quantitative.

“Quantitative research is the process of collecting and analysing numerical data. It can be used to find patterns and averages, make predictions, test causal relationships, and [generalise](https://www.scribbr.co.uk/bias-in-research/generalisability/) results to wider populations” (Bhandari, 2022).

Quantitative methods will be used to monitor the performance of the web application. Some of the elements of the website that will be measured via these methods are:

* Load time of pages
* Time is taken to perform CRUD (Create, Read, Update & Delete) operations.

Collecting this data will help identify if the application is working as it’s intended to and if it’s performing to a high-quality standard. After the development phase is completed then the process of collecting more in-depth quantitative data can begin.

“Qualitative research involves collecting and analysing non-numerical data (e.g., text, video, or audio) to understand concepts, opinions, or experiences. It can be used to gather in-depth insights into a problem or generate new ideas for research” (Bhandari, 2022).

This project will require a lot of user interaction. Qualitative methods can be used to evaluate the user experience on the website. A questionnaire will be designed to track the user experience on the website. This approach will help identify major & minor bugs within the application before it is completed. It will allow a robust finished application.

## 3.3 - Development methodology

A development methodology will streamline the development phase. For this particular project, Agile is the chosen methodology. Agile will allow the project to be developed in 1-week increments/sprints. This will give the opportunity of using modularisation to focus on specifics in each sprint. Testing is also part of the development cycle so a high-quality, robust application will be produced. Agile is also very flexible allowing changes whenever, so user feedback can be actioned as soon as possible. Agile has been chosen over a waterfall methodology because of the following reasons (Amlani, 2012):

1. Refinement to requirements is not a concept in Waterfall, if requirements change then new requirements must be written meaning everything has to be started over
2. The methodology is static so it’s not suitable where requirements may change

In the previous section of research methods, user testing was introduced as one of the qualitative methods. If user testing is conducted and requirements are changed or introduced then Waterfall would not accommodate that, hence, Agile has been chosen.

Figure - Sprint breakdown

*Figure 4* represents what each sprint/iteration will consist of.

* Design: This stage will consist of designing the architecture/user interface of the feature that is going to be added.
* Develop: This stage will consist of the development of the designs in the previous stage.
* Test: This stage will consist of manual testing and unit testing. If there are tests that are failing, then the development stage will be revisited. This cycle will carry on until all tests pass.
* Deploy: The project will be deployed to the repository.
* Review: A sprint retrospective will be held to review the performance during the sprint.

## 3.4 - Programming Languages & Frameworks

### 3.4.1 - Front end

The JavaScript library React will be used for the front end of the application. React uses HTML to structure the web app and CSS is used for styling. React is designed to split a web page up into many smaller components. If we split a web page into smaller components, we introduce the idea of reusability. The main benefit of React is it introduces the idea of reusing components.

Other benefits include:

* Simplifies the development of a dynamic website
* Easy to test
* NPM (node package manager)
  + NPM is a package manager and contains software packages. Meaning you can retrieve code already written and reuse it, preventing you from rewriting code.
* Maintenance is easier

React is used widely in the industry so as well as the technical benefits, it’s a valued skill to be familiar with the library. React has a variety of benefits and it will add desirable value to a skill set and that’s why React has been chosen as the front-end library.

### 3.4.2 - Back end

As React is being used as a front end for this project but there is unfamiliarity with it, the scope of backend languages was narrowed down to a few familiar programming languages. Java, C#, and PHP.

The backend will consist of an API or several APIs that communicate. So, all of the prospected languages are appropriate as they supply frameworks that allow APIs to be built with ease. The prospected frameworks are:

* Springboot (Java)
* .NET Core (C#)
* Symfony (PHP)

Having previous experience with Java and Springboot is a benefit and for that reason, it has been selected. Springboot/Java hasn’t been selected solely based on familiarity. There are many benefits to the Springboot framework such as:

* Robustness
* Security
* Huge online community
* Offers a microservices module (Talked about more in Development Techniques)

## 3.5 - Development Techniques

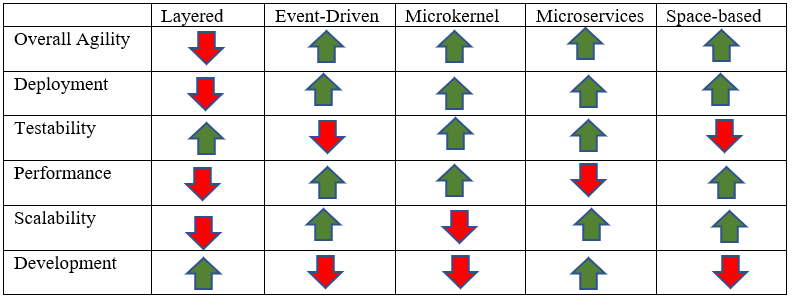


Figure - Architecture analysis summary

*Note.* The figure gives a green upwards arrow if the component has a high rating else if it has a low rating a red downwards arrow is given.Pattern-analysis summary (Richards, 2015, p. 46)

Layered architecture is the most common among engineers that are starting their careers. *Figure 5* shows the downfalls of the layered architecture. This application will be using the microservices architecture as there is previous experience with this architecture and it has an abundance of benefits.

Diagram

Description automatically generated

*Figure 6* represents the target architecture for the application. Each service of the application will be split into an individual module.

Figure - Target architecture

* Code Compiling Service: This service will allow users to send code to it. It will respond with the output of the code.
* Account Service: This service will track the users who have registered on the site. It will also provide information such as authentication keys & refresh keys. This service will connect to a database that stores all relevant information about users.
* Lobby Creation Service: This service will be responsible for allowing users to create and join lobbies using web sockets. This service uses an in-memory database to allow for better performance.

As the services have been separated. We have the freedom of allocating more server resources to a particular service. Performance is most important for the lobby creation service so microservices architecture allows us to focus on a particular service.

# Chapter 4: Requirements Specification and Design

## 4.1 – Introduction

The following chapter will outline the requirements and designs that were essential to complete the project. Since the chosen development methodology is incremental and consists of sprints, the first few sprints were used for requirement elicitation and design creation. The requirements identified have been derived from a literature review, brainstorming and comparison of similar applications.

The overall aim of this project was to create a web application that allows users to participate in coding challenges, one versus one against other users, and live in real time. The meaning of real-time in this context means users must compete at the same time and whoever finishes first is the winner.

## 4.2 - Requirements

A use case diagram is a type of Unified Modelling Language (UML) diagram that allows us to describe the possible usage scenarios that a system is developed for. A use case diagram addresses what the system should do and what specific actions certain users can do. The diagram also highlights the functionality that the relevant stakeholders want (Seidl, 2015).

*Figure 7* helps identify the requirements of the application. *Figure 7* includes two primary actors, an authenticated user, and an unauthenticated user. The secondary actor in this scenario is the system. Primary actors initiate the execution of use cases, secondary actors are involved in the execution of the use case (Seidl, 2015).

Diagram

Description automatically generatedAs *Figure 7* depicts, a user that is logged in (authenticated) will have the option to browse the challenges library, run code, submit code, modify their account, and challenge other users. However, if a user is not logged in (unauthenticated), they only have access to a restricted number of actions such as logging in, registering, or browsing the challenges library.

Figure – Use case diagram

An activity diagram is another type of UML diagram. Activity diagrams aim to show the flow between the actions in an activity (IBM, 2021). Below *figure 8* represents the main two workflows of the application. A user can log on and choose one of two options, to compete against others or to solve problems alone.

Chart

Description automatically generated

Figure – Activity Diagram

### 4.2.1 – Non-Functional Requirements

Non-functional requirements are attributes that are necessary for a well-working system. Attributes such as responsiveness, usability, and security. Non-functional requirements enhance the functionality of a system by specifying capabilities and constraints a system must have, they can also be used as a criterion to evaluate the performance of a system (Tiun et al., 2020). *Table 2* presents the non-functional requirements for this application.

*Table

Description automatically generatedTable 2 – Non-functional requirements*

### 4.2.2 - Functional Requirements

Functional requirements specify what functionality a system must offer. It can be any functionality that the software is expected to perform, for example, this project requires the functionality of code compilation (Dabbagh et al., 2015).

#### 4.2.1 – Requirement 1 – Log in and register.

Creating user accounts and signing in are crucial requirements for the proposed application. Both features enhance security by introducing controlled access via authorisation. The features also allow users to access their personal information, track progress and challenge other users in coding challenges.

Table

Description automatically generated

*Table 3 – Log In use case*   
Table

Description automatically generated

*Table 4 – Register use case*

### 4.2.2 – Requirement 2 – User profile management.

Another vital requirement for the proposed application is to allow users to manage their profiles and view their progress. Giving users the ability to edit their details provides them with a personalized user experience. This opportunity creates a sense of community within the application as it allows users to connect easily.  
Table

Description automatically generated

*Table 5 – User profile management use case*

4.2.3 – Requirement 3 – Library of coding challenges.

Table

Description automatically generatedThe proposed application must have a comprehensive library of coding challenges. The requirement will offer users a wide range of coding challenges to solve and a chance to improve a vast range of their skills. By offering the functionality to search, filter and sort challenges users can target challenges they prefer, improve specific skills, and track their progress. This requirement ensures users can continuously improve.

*Table 6 – Browse through the library of coding challenges use case*

### 4.2.4 – Requirement 4 – A coding environment.

Table

Description automatically generatedA user-friendly coding environment is another vital requirement for the proposed application. The coding environment will allow users to easily write, test and submit solutions to challenges. The necessary tools and resources to write code should be provided for the user. This requirement is vital as it ensures users have a positive experience on the platform and it also ensures users can fully engage with the coding challenges available.

*Table 7 – Access coding environment use case*

### 4.2.5 – Requirement 5 – Participate in a one versus one coding challenge.

Allowing users to challenge other users is the main requirement for the proposed application as it provides users with an additional level of engagement and creates an aspect of friendly competition. This feature would help make the proposed application unique as no similar coding platforms offer a similar feature. The feature would also help build a community on the application. By adding this feature, the application can provide a unique and engaging experience for users.

Table

Description automatically generated

*Table 8 – One versus one coding challenge use case*

### 4.2.6 – Requirement 6 – Solving coding challenges alone.

Table

Description automatically generatedResearch from the literature review highlighted a particular pattern, if a user wasn’t winning many competitions, then they may become discouraged and stop competing. Allowing users to solve challenges alone is another essential requirement for the proposed application. The feature allows users to work at their own pace and focus on challenges that interest them most, particularly new coders who will need time to build their skills. This feature also supports user learning and development ensuring users stay engaged with the application long-term.

*Table 9 – Solving a coding challenge alone use case*

### 4.2.3 – User Stories

In agile methodology, a user story is a form of expressing requirements. User stories consist of short descriptions of a feature or functionality written from a user’s perspective (Atlassian). All the identified requirements were converted to user stories as shown in *Table 10*.

*Table

Description automatically generated*Jira is an issue-tracking software that allows developers to plan, track and work faster. For this project, Jira was used to track the progress of the user stories in *Table 10*. A Kanban board was set up in Jira to track the progress of each user story. “A Kanban board is an agile project management tool designed to help visualize work, limit work-in-progress, and maximized efficiency)” (Atlassian).

*Table 10 – shows the user stories made from the requirements.*

Graphical user interface, application, Teams

Description automatically generated*Figure 9* shows user stories added to a Kanban board in Jira to track the progress of each story. This Kanban board helped with the decision of prioritization. For example, the story “Participate in a coding problem alone” was linked to the story “Create a coding environment” and it was linked as a blocker, meaning the former couldn’t be started until the latter was completed.

Figure – Kanban board in Jira

Graphical user interface, application, Teams

Description automatically generatedAs the user stories were produced from the requirements, they were abstract. So, *Figure 10* shows the subtasks of the “Log in and register system” requirement/story, all subtasks were to be completed for the parent story to be marked as completed. All the user stories presented in *Figure 9* had subtasks.

Figure – Subtasks of each user story

## 4.3 – Design

### 4.3.1 – Introduction

Since the project is a web application, designs were needed for the user interface and backend architecture. Figma was used to create the front-end designs. Draw.io was used to create the back-end architecture designs.

### 4.3.2 – System Architecture

As this application will use React for the client side (front end) and Springboot for the server side (back end) the Model-View-Controller (MVC) design pattern will be used. React will be used as the view layer, which will be responsible for rendering the user interface and handling user interactions. Springboot will serve as the model and controller layer. The controller layer will be responsible for handling the requests and responses between the client and the server. The model layer will be responsible for data access, domain objects and application services.

### 4.3.3 – User Interface/Front Design

Before the user interface was designed, a base colour palette was needed. The colours dark purple, grey, and white were chosen as the three primary colours. The chosen colour palette aims to set a competitive tone for the application.

Graphical user interface

Description automatically generated

Graphical user interface

Description automatically generated

Graphical user interface, text

Description automatically generated

Graphical user interface, website

Description automatically generated

Graphical user interface

Description automatically generated

Graphical user interface, application

Description automatically generated

### 4.3.4 – Backend Design

From the methodology chapter, a decision of using the microservices architecture was made. *Figure 20* presents the target architecture for the backend of the application. *Figure 20* has been extended from the initial design in the methodology. The microservices architecture will allow modularisation which will help prevent tight coupling, resulting in higher-quality code. Each service has its specific responsibilities.

* **Authentication Service** – This service will allow users to log in, register and change their details.
* **Problem Service** – This service will be responsible for the management of the library of problems, meaning it can add, delete and change problems. It will also allow users to retrieve all of the problems.
* **Compiler Service** – This service will be responsible for compiling code and returning its output.
* **Submission Service** – This service will be responsible for allowing users to submit their code and get a score/percentage back based on their code.
* **Lobby Creation** – This service will be responsible for allowing users to create and join lobbies.

Two databases will be used. The first database named ‘Database’ in *Figure 20* will be the main database as it will store user details, scores and details about problems. The ‘In Memory Database’ will be created automatically by model mappings in the lobby creation service, only the lobby creation service will use the ‘In Memory Database’. The application aims to allow users to compete in real-time, so, using an in-memory database will allow for faster data retrieval and less latency. Also, the data that is needed for lobbies is temporary because once a lobby shuts the data can be deleted.

Diagram

Description automatically generated

Figure 20 – Backend Architecture Design

### 4.3.5 – Database design

*Figure 21* presents the entity relationship diagram that will store the data of the application. Each entity has its specific justification :

* **User**
  + This entity is integral as it will store information about users. This will help with authentication and allow the application to track user-specific data.
* **Role & UserRoles**
  + The Role entity will store all of the available roles a user can have.
  + The UserRoles entity will be composed of a composite key allowing users to have multiple roles. This relation will help with authorising any actions a user attempts.
* **Problem**
  + This entity will store the library of coding challenges.
* **TestCase**
  + When a user runs code against a problem, the code that is submitted needs to be run against a criterion, so, this entity specifies one or more test cases for each problem.
* **Tag & ProblemTags**
  + The Tag entity will consist of all the available tags a problem can be associated with. For example, a problem could consist of interacting with arrays, so, ‘arrays’ could be a row within the Tag table.
  + The ProblemTags entity will be composed of a composite key allowing problems to have multiple tags. This relation will eventually help with filtering down problems.
* **Language & ProblemLanguage**
  + The Language entity will consist of all the programming languages the application offers.
  + Graphical user interface

    Description automatically generated with medium confidenceThe ProblemLanguage entity will specify a base template of code for each programming language for each specific problem. This code will be presented to a user when they first visit a problem. This entity will also store ‘testRunOne’ and ‘testRunAll’ this code will be added to the user's code to run all test cases or one test case. This Entity is essential as it gives the user a starting point and it stores data that will help test each solution.

Figure 21 – Database Entity Relationship (ER) diagram

## 4.4 – Summary

The discussion in this chapter concentrated on the primary requirements required for this project to meet its objectives. Six essential requirements have been identified. Designs were also considered in this chapter; a dark colour scheme has been chosen for the theme of the user interface so it symbolizes the characteristics that are related to competition. The target architecture picked for the backend is microservices with 5 specific services.

# Chapter 5 – Development and Implementation

## 5.1 – Introduction

The following chapter is an in-depth discussion of how the designs and requirements were implemented. In each requirement that has been implemented, there will be screenshots, diagrams and code shown, as well as a discussion on any problems faced during development and how they were overcome.

The development phase was completed in sprints. The sprints were based on developing each specific requirement from the previous chapter. There were 6 requirements.

## 5.2 – Database

The development phase commenced with the creation of the Database. The database designed in the previous chapter was created. The database was then connected to an empty Springboot application. The Springboot application was to be the backend API that the client interacts with.

## 5.3 – Log In & Register

The first requirement that was developed was the login and register requirement.

### 5.3.1 – Front end

*Figures 22 and 23* represent the two states the user interface (UI) has. The UI for this requirement has been implemented with both log-in and register forms on the same page. This decision was made to simplify the user experience. Both forms require a username and a password, the registration form requires a few more inputs. The user needs to at least provide an email, username, and password to register, all other details can be left blank, including a blank profile picture. A default profile picture will be allocated to the user. To log in the user will need to use their username and password.

Graphical user interface, application

Description automatically generated

Figure 22 – LogIn&Register page (State 1)

Graphical user interface, application

Description automatically generated

Figure 23 – LogIn&Register page (State 2)

*Figure 24* presents code from the logInAndRegisterPage component. The useState function returns a value and a method to update that value. When the value is updated the component re-renders. This technique was used to dynamically render in the appropriate form. When the Register button was pressed (but the login form was showing) then the value would turn to false, and the registration form would show. The opposite would happen when the log-in button is pressed when the registration form is showing. As shown in *Figure 24*  labels were dependent on this state and they would change appropriately when the state changed. Other things were also dependent on the state such as :

* Which errors are displayed? (If inputs were failing validation)
* Which buttons are displayed?
* Text

  Description automatically generatedWhich endpoint is a request sent to?

Figure 24 – LogInAndRegisterPage component code

### 5.3.2 – Back end

*Figure 25* represents the specific microservice that was developed for this requirement. As this was the first requirement developed, the development included the initial creation of the API and the introduction of its first service. Springboot offers a framework called SpringSecurity. SpringSecurity offers authentication and authorization. SpringSecurity was used to create the Authentication service.

Diagram

Description automatically generated

Figure 25 – Backend Diagram

*Figure 26* presents an overview of how SpringSecurity authentication and authorization work. The authentication filter attempts to authenticate a user, if the attempt was a success then the user receives an access token and a refresh token. The access token is passed to any request the client wants to make in the future. Below, *Figure 27* presents the code for the authentication filter. Two methods are implemented :

* attemptAuthentication
  + This method is executed when a user tries to log in.
* successfulAuthentication
  + If the login attempt is successful then this method is executed.

If the login attempt is not successful then SpringSecurity has a built-in method that sends an error to the client.

Diagram

Description automatically generatedThe user must attach the access token they received to every request they send to the API. The access token is a JSON web token that consists of data such as the roles a user has. When the user sends a request, before their request reaches any microservice they go through the authorization filter. Below, *Figure 28* represents the code that authorizes a user to access an endpoint. This framework helps create protected endpoints. For example, It allows only admins to trigger endpoints that could delete entries from the database.

Figure 26 – Overview of SpringSecurity

Text

Description automatically generated

Figure 27 – AuthenticationFilter code

Text

Description automatically generated

Figure 28 – AuthorizationFilter code

### 5.3.3 – Problems encountered

#### 5.3.3.1 – Storing images

Images are stored as base64 strings. When a user uploads their image, the image is encoded to a base64 string. The image column was a blob in the database and this was causing issues for certain users to sign up as their image was too large when converted to a base64 string. The resolution to this problem was to change the data type to a long blob.

## 5.4 – User profile management

The next few sprints were focused on adding user profile management.

### 5.4.1 – Front end

The front end of the user profile management requirement was simple. It included a page with two sections. *Figure 29* shows the UI that was implemented by following the design in the previous chapter. The section on the left-hand side allows a user to change their current profile picture. The section on the right allows users to change their details.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 29 – User Profile Management UI.

Below, *Figure 30* shows the code that ensures the password, and the username are validated, similar code is also ran on the email input field. If validation fails on any of the inputs, then errors are displayed. Below, *Figure 31* shows there is validation once again when the form is submitted to ensure all inputs are valid, if any input is invalid then a HTTP request is not sent to the backend API.

Text

Description automatically generatedText

Description automatically generated

Figure 30 – Input validation

Figure 31 – form validation

### 5.4.2 – Back end

The backend development for user profile management was extending the authentication service. *Figure 32* shows code that adds a post endpoint to the authentication controller. The endpoint retrieves a user object, compares the user object against the current user details in the database and updates the user entry accordingly.

Text

Description automatically generated

Figure 32 – change user details endpoint in the authentication controller.

## 5.5 – Library of coding challenges

The next few sprints consisted of adding coding challenges to the application.

### 5.5.1 – Front end

The UI for the requirement of having a library of coding challenges was simple. *Figure 33* presents the implemented UI for the requirement. The challenges are listed out whilst the user has a choice to filter the challenges and some additional details are displayed, so the user can track their progress.

Graphical user interface, application, Teams

Description automatically generated

Figure 33 – The user interface of the problems page

The client requests the backend API when a user visits the problems page. The client receives JSON data that contains a list of objects where each object contains data about each challenge. So, on the client side, the original list of challenges is stored and another copy is also made. This is because the second copy that is made is going to be the filtered version of the original copy. So, when a user selects a filter, the original list will be filtered but the filtered version will be applied to the second copy made, so, we always have the original list. This logic helps eliminate the need of sending requests with different filters. *Figures 34 and 35* represent the implementation of this logic.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 34 – two lists that store a list of challenges.

A screenshot of a computer

Description automatically generated

Figure 35 – Mutated problem list is set to the filtered original list.

Each challenge is rendered as ProblemBar as shown in *Figure 36*. A ProblemBar component was created to encapsulate each challenge's data. *Figure 36* shows code that makes use of the ProblemBar component. The code results in each challenge object in the mutated list being rendered on screen. The mutated list is used so the challenges shown are filtered to the users liking. The scenario of having no challenges with certain filters has been considered as *Figure 36* shows.

Text

Description automatically generated

Figure 36 – Rendering each Challenge in

### 5.5.2 – Back end

The backend development for this requirement was straightforward. *Figure 37* highlights the added microservice in red. *Figure 38* shows the added endpoint for this requirement. The main responsibility of the backend for this requirement was to send all the challenges to the client, the client-side was then responsible for filtering and displaying the data appropriately.

Diagram

Description automatically generated

Figure 37 – Backend diagram updated (version one).

Text

Description automatically generated

Figure 38 – getAll endpoint in problem service.

## 5.6 – Implementing a coding environment

### 5.6.1 – Front end

To implement a coding environment, a code editor was needed. “AceEditor” is a library that can be downloaded using the node package manager in React. AceEditor was used as the code editor for the application.  
 *Figure 39* represents the user interface (UI) for the requirement of having a coding environment. *Figures 39 and 40* represent the two different states the UI can have. *Figure 39* represents the state of the UI when neither the “run code” nor “submit code” buttons have been pressed, on the other hand, *Figure 40* represents the state of the UI when either of the buttons has been pressed.

A screenshot of a computer

Description automatically generated with medium confidence

*A screenshot of a computer

Description automatically generated with medium confidence*

Figure 40 – Coding environment (State 2)

Figure 39 – Coding environment UI (State 1)

*A screenshot of a computer

Description automatically generated*React introduces the idea of splitting a UI into many smaller components. *Figure 41* represents the different components of the coding environment UI. The idea of splitting the UI into many smaller components is beneficial regarding performance. When one of the two buttons is pressed “Run Code” or “Submit Code” then an HTTP request is made to the backend API. Instead of re-rendering the whole UI when a response is received, only the section highlighted in Yellow is re-rendered. Overall breaking down the UI into many smaller components helped towards code maintainability, organisation and optimised performance.

Figure 41 - Components of coding environment user interface

### 5.6.2 – Back end

Implementing a coding environment required no backend code. The backend of this requirement was to compile code. Code compilation was added in the development of the next requirement in section 5.7.

### 5.6.3 – Challenges to complete

The coding environment is accessed from the problems page. As mentioned in section 5.5, The problems page implements a UI for users to access a library of coding challenges. Each coding challenge is rendered in as a horizontal card and each individual coding challenge has a button “Solve Problem”. When the user clicks that button, the application navigates to the coding environment and shows a problem description, examples and template code relative to the specific coding challenge the user chose. The user must then add to the template code and run their code to see if they have a working solution. The user can pick from 10 coding challenges.

## 5.7 – Allowing users to solve coding challenges alone.

### 5.7.1 – Front end

The UI of this feature was implemented for the requirement: Implement a coding environment.

### 5.7.2 – Back end

For the coding environment to be fully functional the user would need to see the output of their code and the result of their submissions.

*Figure 42* highlights two microservices in red that were developed and created for this requirement. The submission service is only interacted with if the user has tried to submit their code.

*Chart, diagram, box and whisker chart

Description automatically generated*

Figure 42 – Backend diagram updated (version two).

*Figure 43* represents the structure of each service. A controller is used to create an endpoint that is available to clients. When a specific endpoint is hit then a service class would be called where business logic is executed. Models are used in the process as data transfer objects. Some services are internal to the application, For example, the submission service was only ever used by the Compiler service so in that case a controller isn’t necessary.

Graphical user interface, text, application

Description automatically generated

Figure 43 – Structure of each Service

*Figure 44* is the code of the Compiler service controller. It has two post endpoints which were also shown in *Figure 42*. Each method runs when the relative endpoint is hit by a client. As mentioned previously, the controller creates endpoints that are hit by a client and then the CompilerService is called.  
  
Text

Description automatically generated

Figure 44 – Compiler Controller

The CompilerService class is called and created by the CompilerController class. The CompilerService runs the code and returns the output of the code, which is sent back to the user. The CompilerService needs to know which language the client is coding in as there are different steps for different languages, I.e. Java needs to be compiled before it can be run however Python can just be run. This is why the CompilerService uses the Factory pattern. Factory pattern is a creational design pattern. “The factory method provides specific object types via a route that is not specific to the application” (Drumm, 2020). *Figure 45* presents the factory pattern implemented within the CompilerService. The CompilerService calls a method from the CompilerServiceFactory class and passes it a specific programming language name, The language the client is programming with. Then the compiler service factory correctly returns the appropriate language service i.e. if Java was the language being used then the JavaCompilerService would be returned. Each language has its specific compiler service as they are all unique.Graphical user interface, text, application

Description automatically generated  
  
  
  
F*igures 46, 47 and 48* show the code implementation ofthe factory pattern. *Figure* 46 shows the CompilerService class interacting with the CompilerServiceFactory class. The interaction between them results in the correct language service being returned. *Figure 47* shows the CompilerServiceFactory and how it determines what is the correct language compiler service. *Figure 48* shows the abstract interface that is implemented by each specific language compiler service.

Figure 45 – Factory pattern

Text

Description automatically generatedText

Description automatically generated

Figure 46 – CompilerService interacting with CompilerServiceFactory.

Text

Description automatically generated

Figure 48 – Abstract interface that is implemented by all language compiler services.

Figure 47 – CompilerServiceFactory implementation

### 5.7.3 – Problems encountered

The way each language compiler service would work is similar, the following steps would be executed:

* Append the client's code with extra code that will call the method that the client has implemented
* Write the code to a temporary file
* Compile the file by running a Process (This Step is skipped for some languages)
* Run the file by running a Process
* Store the output in an appropriate data structure
* Return the output of the client's code

#### 5.7.3.1 – Deploying application

When the application was deployed to a server, a problem was introduced as the server didn’t have compilers for all three available languages. The application was erroring for Python and JavaScript. This meant every time a client wanted to see the output of their Python or JavaScript code the application was throwing errors.

The resolution to this problem was to use Docker. Docker packages software into standardized units. Instead of deploying the application directly to the server, deploy a docker image to the server instead. *Figure 49* below shows the docker script to create the image. First, the application was set up by completing two stages, clean and build. Then JDK-17, Python and Node were installed so the server could compile and run all three languages. A port number is exposed and the application is started. This resolution successfully fixed the problem.

Text

Description automatically generated

Figure 49 – Docker script to build the docker image.

#### 5.7.3.2 – Compiling code

Code was being compiled and run without any security measures in place, making the server vulnerable to malicious code and potential memory issues.

To address this issue, two solutions were thought of. Either sandbox the environment the code runs in or use an external API to compile and run code. The latter was chosen as it would save time and resources. Piston API was chosen (<https://github.com/engineer-man/piston>). The API uses Docker to sandbox the code and has restrictions in place, restrictions such as :

* Prevents network interaction
* Caps runtime execution
* Caps max processes

This API offered a robust but very customisable solution, relying on this API successfully fixed the problem however it created an external dependency meaning this application cannot run if Piston API goes down. This solution also meant docker wasn’t needed anymore so it took away some complexity.

## 5.8 – Allowing users to solve coding challenges against other users.

The final few development sprints were based on allowing users to compete against other users.

### 5.8.1 – Front end

Below Figures 50, 51, 52 and 53 present the UI for this requirement. The flow of the UI goes from Figure 50 to 51 to 52 and finally 53.

Figure 50 is the challenge page UI where users can create or join a lobby. When a user creates a lobby then a unique ID is generated. Figure 54 shows the code that is run when the create button is pressed. A unique id is created, and the user is navigated to the challenge lobby which is shown in Figure 51. The challenge lobby route takes an id so a user can simply send their URL to another user, and they can use that URL to join the lobby. The user also has an option of sharing the lobby code with other users. When a user presses the join button then the code in *Figure 55* is executed. The code takes a challenge code and navigates to that lobby. The user who creates the lobby is marked as the lobby leader, while those who join aren’t.

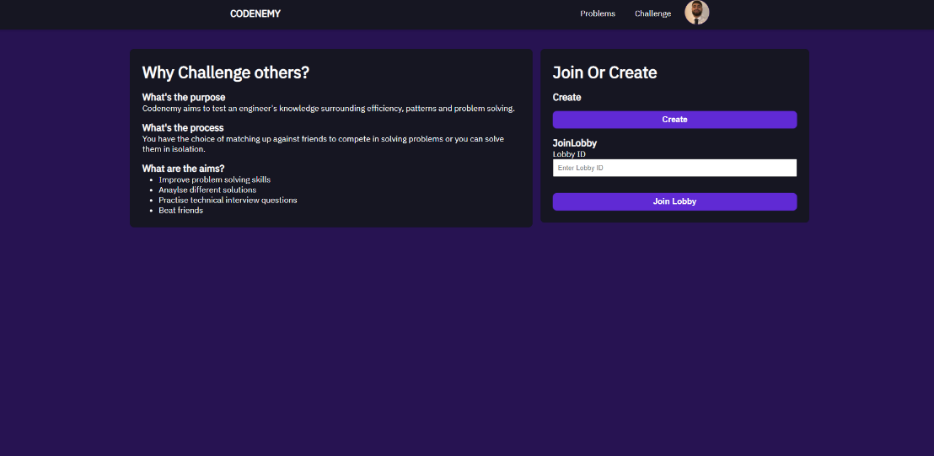


Figure 50 – Challenge page UI (Starting point of flow).

A screenshot of a computer

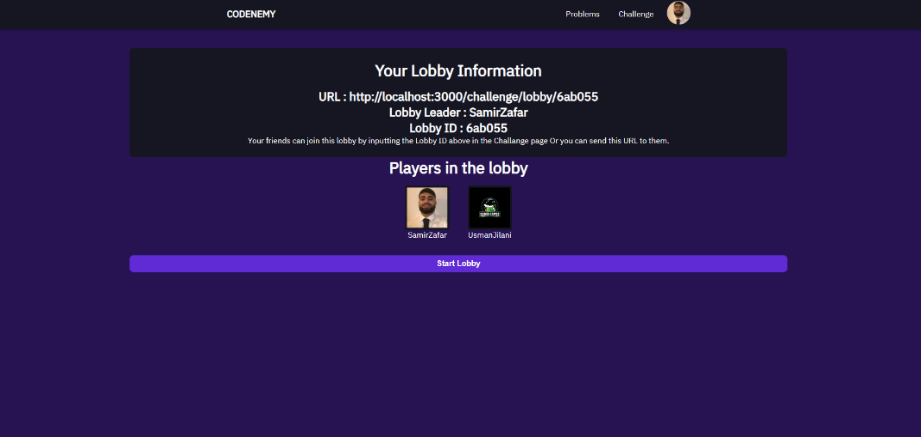
Description automatically generated with medium confidence

Figure 51 – Challenge lobby UI (second point of flow).

Figure 52 – Challenge solving problem UI (third point of flow).

Graphical user interface, application

Description automatically generated

Figure 53 – Challenge results (four point of flow).

Text

Description automatically generated

*Figure 53 – Challenge results (four point of flow).*

Figure 54 – createLobbyHandler code.

*Text

Description automatically generated*

Figure 55 – joinLobbyHandler code.

When a user enters a challenge lobby a web socket connection is initialized between the client and the server, the user also subscribes to a channel specific to that lobby. A web socket connection is a duplex connection seen in Figure 56, it allows for two-way communication. When a new user joins that lobby then the server sends out a message to all users in the lobby with an updated user list. During the challenge, a coding challenge is presented in the coding environment to the user. When a user submits their solution, a message is sent to the server to indicate they’ve finished. The server then sends a message to all users in the lobby with updated user statuses. Finally, the results screen is presented.

Diagram, timeline

Description automatically generated with medium confidence

Figure 56 – WebSocket Diagram

### 5.8.2 – Back end

*Figure* 57 shows the microservice that was developed for this requirement. The lobby creation service was created solely to allow users to join lobbies, create lobbies and track user statuses within lobbies.

Chart, diagram

Description automatically generated

Figure 57 – Backend Diagram updated (Version three).

*Text

Description automatically generatedFigure 58* shows the code of the LobbyController. The code shows how the server sends out a message to lobbies. For example, the client would be subscribed to the endpoint “lobby/public/getAll/dummyLobbyId” and every time the server sends a message to that endpoint the client side will receive it and re-render to show the updated information.

Figure 58 – Message endpoints for web socket server.

## 5.9 - Summary

In this chapter, the development and implementation of the requirements and design of the previous chapter were discussed. The designs and requirements were successfully implemented, and a minimum viable product was produced.

# Chapter 6 – Testing and Analysis

## 6.1 – Introduction

With the requirements developed and implemented, the following chapter gives an overview of how the application was tested. To ensure the code was robust testing was conducted to test the code's efficiency and reliability. Different methods of testing were deployed. The following testing methods were used:

* Regression testing
* Unit testing
* Usability testing

Testing was conducted in each sprint that contained development. For example, if the backend needed to be extended in a sprint then new unit tests would be introduced and existing tests would be run again to ensure new logic hasn’t affected anything else. Test-driven development was used for unit testing. Regression and usability testing required more manual and exploratory testing.

A continuous integration and continuous delivery (CI/CD) pipeline was also created to ensure all additions of code were robust and didn’t cause any other functionality to fail.

## 6.2 – CI/CD pipeline

GitLab is an open-source code repository and collaborative software development platform for DevOps (Contributor, 2020). Gitlab was used to create the CI/CD pipeline. A CI/CD pipeline is the most fundamental component of automated software development. A CI/CD pipeline is a set of stages that run when you push code to a repository. An example of a stage could be to build the project to ensure the project has no compile time errors. Continuous integration automates the building and testing of your software. Continuous Deployment (CD) is an extension of CI and it allows your code to be deployed after every commit (Fosco, 2022). According to (Zampetti et al., 2021), having a CI/CD pipeline can introduce the following benefits:

* Early defect discovery
* Increased Productivity
* Faster release cycles

*Figure 59* shows the file that is used to create the GitLab CI/CD pipeline. At the top of the file, three stages can be seen, two CI stages, build and test and one CD stage, deploy. The build stage is simply ensuring the project builds. The test stage runs all unit tests. The deploy stage deploys the application to Heroku, the deploy stage only runs if the code is committed to the main branch.

Text

Description automatically generated

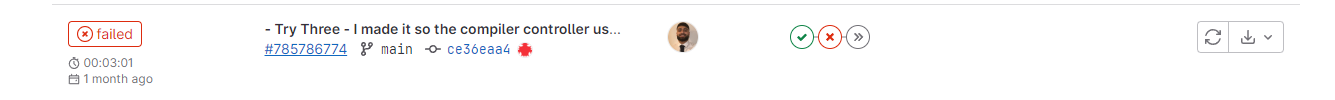
Figure 60 represents the role of the CI/CD pipeline. When code is pushed up to the remote repository, the pipeline is triggered and three stages are completed. Only once the pipeline is green then a pull request (PR) can be approved and finally the changes can be merged.

Figure 59 – Gitlab pipeline configuration script

*Diagram, venn diagram

Description automatically generated*

Figure 60 – Gitlab pipeline overview

*Figure 61* represents the visualisation of each pipeline run in Gitlab. If a pipeline fails, an automated email is sent out. *Figure 61* shows the pipeline status, branch name, commit title and status of each stage in the pipeline. *Figure 62* is a more in-depth view of the same information with the only difference is having the option to click into each stage to view the terminal. Each time the test stage runs it produces a test report, this is shown in *Figure 63*. The test report helps narrow down the exact test that is failing. *Figure 63* highlights the failing test in red.

Graphical user interface, text, application

Description automatically generated

Figure 62 - Pipeline run breakdown

Figure 61 – Pipeline status on Gitlab

Graphical user interface, text, application

Description automatically generated

Figure 63 – Pipeline test report

## 6.3 - Regression testing

Regression testing ensures that new changes made to the software, such as adding new functionality or modifying existing functionality, have not affected existing functionality. The purpose of regression testing is to ensure the existing application is intact with the newly added features. So, Regression testing is used to ensure old code is still working as intended once new code changes have been introduced. Regression testing was conduced manually to ensure existing functionality still worked hence there were no screenshots.

*Figure 64* represents when regression testing was conducted. Once new code was committed to the remote repository, passed the pipeline, had an approved pull request and was finally merged then regression testing would start. Regression testing can be executed completely but in this case, it was executed selectively. Regression testing can be prioritised based on the areas of the application that are most likely going to be impacted, this approach saved time and resources. For example, if the functionality of log in was being added then the register functionality would be prioritized for testing as it’s most likely going to be impacted. *Figure 65* represents the process of regression testing.

Diagram

Description automatically generated

Figure 64 – Regression testing overview

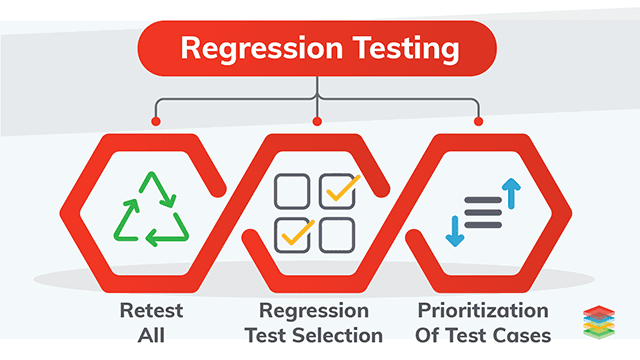


Figure 65 – Regression testing process

## **6.4 – Unit Testing**

A unit test is a unit of code, that unit of code is isolated and the actual result of the unit is compared to the expected. In Java, one unit is usually a class. The unit tests invoke one or more methods from the class to ensure they function and return (if applicable) the expected result (McGregor, 2007). Unit testing was used to test the backend of the application which was coded in Java using the Springboot framework.

Test-driven development (TDD) was used to create unit tests. TDD is a software development practice where unit tests are written before code implementation (George & Williams, 2004). Sometimes when unit tests are written after code has been implemented they can change to pass but by writing unit tests first, developers can ensure new code introduced acts as it originally intended to. *Figure 66* represents the TDD framework. First, a failing unit test is written, then code is implemented until the test passes, the code is then refactored and this cycle is repeated.

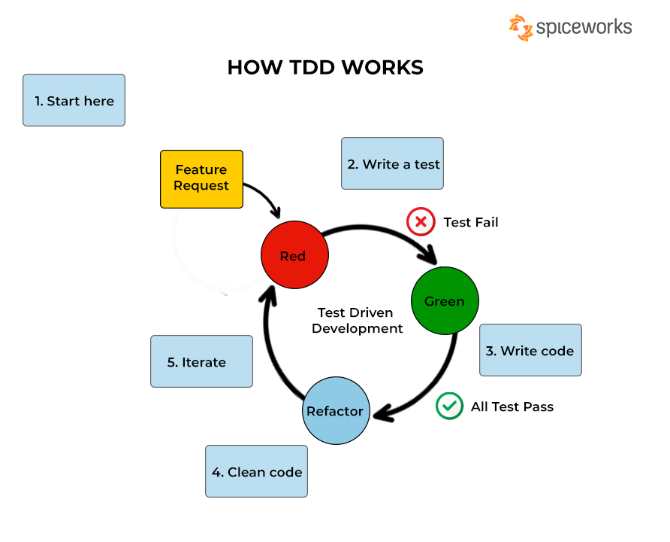


Figure 66 – TDD framework

*Figure 67* shows the test package of the application. Each service was unit tested. The compiler service is expanded to show how each layer of each service was tested.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 67 – Application test package

*Figure 68* presents a unit test from the UserControllerTest file, this file tests whether the user controller is working as intended. The user controller has a method for admins that returns all the users. So, a unit test tests individual layers, so we mock other layers. For example, *figure 68* shows a mock service being used and we tell that service to return an array of users we have defined. We then call the method from the user controller and finally we verify various things such as the HTTP status, the users returned, and we also verify that the user service was interacted with.

Text

Description automatically generatedBelow *figure 69* presents the test suite in an integrated development environment (IDE). 82 tests were created to test the whole backend application.

Figure 68 – Example unit tests

Graphical user interface, text

Description automatically generated *Figure 70* presents the amount of code the unit tests cover. The unit tests cover over 96% of the code. High code coverage ensures that there are few areas untested where bugs could potentially exist so it reduces the likelihood of undiscovered bugs, improves code quality and provides confidence in code.

Figure 69 – Application Unit Tests

Graphical user interface, text, application

Description automatically generated

Figure 70 - 90%+ code coverage

## **6.5 – Usability testing**

Usability testing is the process of an application or service being tested by representative users. The users are typically selected based on the target demographic for the product. The purpose of usability testing is to identify any areas of improvement in the application. The results of the testing can be used to target specific areas for improvement to enhance the application.

Volunteers were instructed to use the application as they normally would. Then the volunteers were asked to provide feedback on their experience. The volunteers filled out a questionnaire. 14 volunteers participated and the following questions were asked:

* Question 1 – How easy or difficult was it to navigate the website?
  + Very Easy - 1, Very Hard - 5
* Question 2 – How was the user interface and design of the website?
* Question 3 – Were the problem statements clear and easy to understand?
  + Very Understandable - 1, Very Confusing - 5
* Question 4 – Were you able to submit/run your code and receive feedback on it?
  + Yes
  + No
  + Yes (But I would like to see which test cases I am failing)
* Question 5 – Were you able to find and challenge other users?
  + Yes
  + No – I didn’t want to challenge anybody
  + No – I couldn’t find anybody to challenge
* Question 6 – How was the overall user experience on the website?
* Question 7 – How was the performance of the website?
  + Very Good - 1, Very Bad - 5
* Question 8 – Are there any features you would like to see added or changed on the website?

### 6.5.1 – Questionnaire responses

The next eight figures represent the responses to each question in either table format or graphical visualisation.Chart

Description automatically generated

Figure 71 – Question 1 response graph

Table

Description automatically generated

Figure 72 – Question 2 response table

Chart, bar chart

Description automatically generated

Figure 73 – Question 3 response graph

Chart, bar chart

Description automatically generated

Figure 74 – Question 4 response graph

Chart, bar chart

Description automatically generated

Figure 75 – Question 5 response graph

Table

Description automatically generated

Figure 76 – Question 6 response table

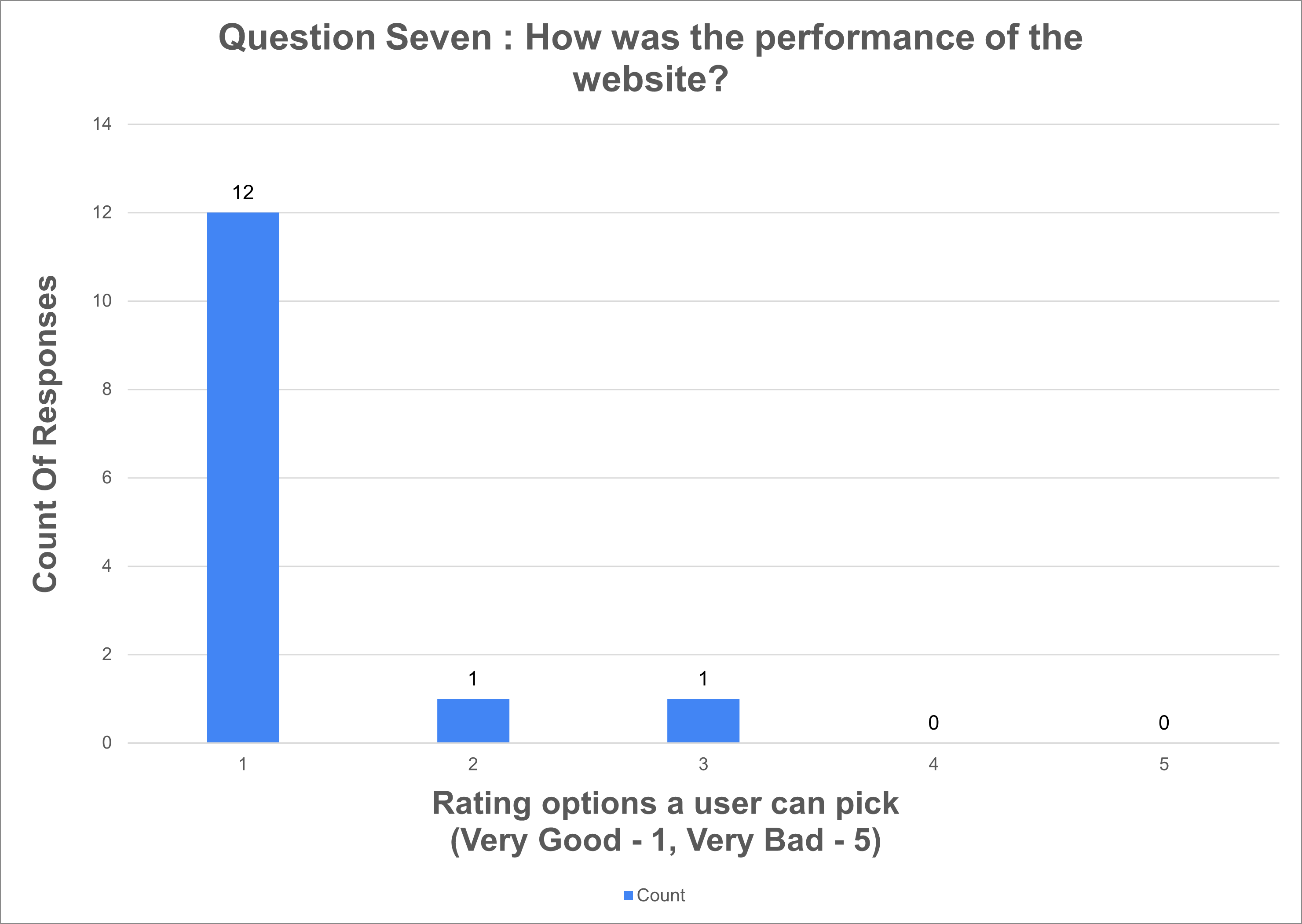


Figure 77 – Question 7 response graph

Table

Description automatically generated

Figure 78 – Question 8 response table

### 6.5.2 – Analysis and Retrospective

Overall the responses recorded from users suggest that the website had very easy navigation as most respondents selected “Very Easy” (1) as their response to question one. The user interface (UI) of the application was very well received as from responses we can conclude that users found the UI clean, intuitive, and visually appealing. However, some users would like to see more features that are related to UI such as a light mode option, scoring for challenges and the ability to see which test cases they are failing.

The responses that were related to the problem statements were mixed, with some users finding them clear and understandable whilst others found them a little more confusing. This suggests that some problem statements need to be revised as some users may benefit from clearer explanations and instructions.

The majority of responses show that most of the users were successfully able to submit their code and receive feedback on it, although some users expressed a desire to see which specific test cases they are failing on. Also, users that desired to challenge other users could successfully challenge other users. Some users did not want to challenge anyone.

The responses that were related to user experience on the website were generally good. Users found the website fun, easy to use and educational. Also, the responses from the users suggest that the performance of the website was good although some users commented on the page load speed and how it can be improved.

In terms of the features that users would like to see added, responses were varied. Some users have suggested enhancing the learning experience of the website by adding TDD and code Katas. Other users also suggested a message feature for better communication whilst users challenge each other.

Overall, the responses to the questionnaire were very positive. The responses allow us to assume that users had a positive experience on the website, with most finding it easy to use and visually pleasing. However, there were a few suggestions for improvements and additional features that could enhance the user experience and provide more educational value.

## **6.6 – Summary**

This chapter focused on how the crucial requirements of the application were tested. Feedback was gathered from users to meet requirements more concisely. The chapter also highlighted how tests were produced to ensure all code was efficient and reliable. Overall, the testing strategies deployed worked very well and allowed for a higher quality, more robust application. However, suggestions from test users were not implemented due to time restrictions.

# Chapter 7 - Critical Evaluation

This chapter will give an overview of whether the project successfully met all requirements that were identified in the project proposal. This chapter will also evaluate the project plan, by explaining and justifying any deviations made during the development phase. Finally, this chapter will highlight any lessons learnt.

## 7.1 – Review of the project against its objectives

The first stage of this project was to identify the aims and objectives. The objectives identified contained core objectives to complete the project. All of the objectives identified at the start stayed the same throughout the whole development phase. No objectives were dropped. *Table 11* contains an evaluation of each objective. The objectives were identified in the project proposal which is attached in Appendix A.

Table

Description automatically generated  
*Table 11 – objective evaluation*

## 7.2 – Review of the plan

Overall, there weren’t many deviations from the project plan. The project was developed and implemented in parallel with the proposed project plan. Each sprint's start and end dates were met. Requirements were completed in time or sometimes even earlier than intended. The project overall was completed earlier than expected which gave more time for comprehensive usability testing to be conducted.

The order of completing the objectives was changed slightly. I originally intended to create a CI/CD pipeline after completing objective 2. I implemented the CI/CD pipeline whilst working on objective 2. Objective 2 consisted of developing a data-driven website that allowed users to practice code and compete against each other. If a CI/CD pipeline was implemented after the implementation of the website, then the CI/CD pipeline would’ve served no benefits. The early implementation of the pipeline was beneficial and helped rectify many mistakes during the development phase.

Under objective 4, a task of using Docker to create an image was made. This task was left incomplete. Originally the task was completed as my application had a custom built-in code compiler in the backend. However, after my supervisors highlighted the security implications of having a compiler on a server, I decided to depend on a third-party API to compile code for me. This change meant a Docker image of the application was no longer needed.

Under objective 2, the task of creating an algorithm that helps match users of similar abilities was made. This task was left incomplete. During the development phase, an alternative approach was given to the users. The users could create lobbies and invite users they want to challenge. This alternative approach gives users more choices and a more personal experience on the website. In the future, the application should be extended to add an option of matching with random users of the same ability in the case of users who are alone.

Overall, the project plan was accurate and it turned out to be a good plan that allocated more than enough time to successfully meet each objective.

## 7.3 – Review of Feedback and Product

Overall, the feedback from Chapter 6 was very positive. The feedback from the test users shows the website served its intended purpose. I am happy with the application. The application offers a great user experience. The website functions as it’s supposed to and provides value to any user that uses it. Feedback from respondents highlights that although the website is good in its current state there are plenty of areas that can be improved. I also believe the website can be extended and improved in certain areas. The website in its current state is a minimum viable product but to compete with competitors it needs to be extended thoroughly.

## 7.4 – Lessons Learnt

Developing this project presented me with an opportunity to demonstrate everything I have learned throughout university, an industrial placement, and an industrial internship. It allowed me to showcase my knowledge of the latest technologies and techniques used in the commercial space. The project also helped me develop a deeper, more comprehensive understanding of various software engineering techniques, frameworks, technologies, the software lifecycle, the importance of testing, and most importantly the gamification of coding.

I learned the importance of time management and staying organized. Both of these skills have developed for me over my final year. Having to complete multiple modules in parallel to developing and documenting a final-year project forced me to focus on both of these skills to ensure I delivered high-quality work consistently across all modules.

Conducting intensive research was one of the most important requirements of this project. This requirement helped me develop deeper research and information-gathering skills. These skills also came to use for my other modules that were report based. I also improved my writing and presentation skills as I consistently had to present information gathered in simpler forms.

I also learnt the importance of not over-engineering an application. In Chapter 5 I discussed how I first approached the code compiling for my website. I tried to implement my code compiler. This approach introduced unnecessary features, processes and components that added little benefit. In the end, I used an external API that compiled code for me. As I initially over-engineered, I was less efficient and used more development time than intended. My supervisor reinforced the importance of not over-engineering by telling me about the saying “reinventing the wheel”.

Before this project, I was not aware of the importance and practicality of using a continuous integration (CI) and continuous deployment (CD) pipeline. Because a CI/CD pipeline was created and set up for the backend of my application I got to experience the benefits of having a pipeline. On some occasions I had changed code for a specific requirement, the change affected other areas that I was unaware of and obliviously I committed the code. I only realised the code had affected other areas once the pipeline failed as unit tests were failing. Without a CI/CD pipeline in place, unintentional changes could’ve gone unnoticed until it was too late. I also learnt how to set up a CI/CD pipeline in GitLab.

Finally, I learned the importance of testing, usability testing in particular. Receiving feedback from test users was very important as it allowed me to gain insight into the perspective of the targeted demographic. The responses from users allowed me to identify the strengths and weaknesses of the website.

Overall, I learnt that developing a website involves much more than just programming. Building a website requires a variety of skills such as time management, research, testing and deployment. I gained a comprehensive understanding of the software development lifestyle, including the importance of requirements gathering, design and maintenance. I also got the opportunity to familiarise myself with more technologies, for example, Docker. Throughout this project, I was able to apply the theoretical and practical knowledge I have gained throughout university and my work experience. I received positive responses from test users and that was a rewarding experience. Overall, this project was great for my learning.

## 7.5 – Summary

The proposed objectives were all successfully met. The project plan was stuck to and minimal deviations were made. A few tasks were dropped for better alternatives. The project process was full of learning opportunities. The challenges I faced helped me become a more knowledgeable engineer. I am very happy with the result but I, as well as the respondents, realise the website can be extended in particular areas.

# Chapter 8 – Conclusion

## 8.1 – Project Overview

This project aimed to build a web application that would allow users to solve coding challenges against others or alone. The primary aim of this project was successfully achieved. The web application produced was of a high standard as it is responsive and can be used on any device, compatible with different web browsers, allows users to compete against other users, and allows users to solve challenges alone. This application can be used on any device and is available to all users. The application provides a unique way of developing skills for technical interviews. Moreover, fourteen test users participated in usability testing to ensure the usability of the software satisfied them. All feedback received was positive, and all feedback showed that users found the website easy to use and functional.

## 8.2 – State-of-the-art comparison

A state-of-the-art comparison with Leetcode and HackerRank was drawn. HackerRank and Leetcode are two very popular platforms that offer users the opportunity to solve coding challenges. Although this project was unique and provided functionality that the named platforms don’t offer, they are very similar. This project had a pretty small library of coding challenges as adding challenges was a time strenuous task, Leetcode and Hackerrank both offer a large library of coding challenges. Both platforms have over 1000 coding challenges so naturally, both platforms allow users to improve larger variations of technical skills. This project offered users the opportunity to solve coding challenges alone or against someone but nothing was tailored to a specific user. Hackerrank and Leetcode offer various study plans. Study plans with varying durations and varying technical areas. For example, a 2-week data structure study plan is offered on Leetcode and a 1-month preparation kit is offered on HackerRank. This project required users to challenge in real time whereas Leetcode and HackerRank offer a mode where you can compete but in your own time. With the above in mind, it can be concluded that this project offered a unique and new feature that has never been seen before however, the two platforms offer a variety of features that cover a larger spectrum.

## 8.3 – Future work

During a final-year project seminar, the time restrictions for this project were highlighted. It was concluded that it was not feasible to produce a complete software due to time restrictions and other variables. When the usability testing was conducted, it was made apparent that there is always room for improvement. There is always a chance to fit in more features if more time is allocated. For example, a live chat feature could’ve been implemented. A leaderboard system could have been implemented so users had intrinsic motivation to get better at solving coding challenges.

As highlighted previously state-of-the-art applications offer a large variety of features, As this is a similar application many of the same features can be considered to be implemented in the future. Below is a list of significant features that would extend the functionality of this application:

* Build a large library of coding challenges so users have a variety of challenges to pick from and they also cannot complete all of the coding challenges within a few days
* Offer various study plans that allow users to focus on particular technical knowledge/skills
* Allow users to chat on a discussions page to create a community on the platform
* Create an Elo rating algorithm that rates the ability of users
* Create an algorithm that matches random users of the same ability for a coding challenge so users can compete against anyone
* Implementing a system where users can earn badges, trophies, or points for participating in coding challenges
* Allowing users to review other users' code. This approach could help users identify areas for improvement and learn techniques and perspectives from fellow users
* Offer certification to build a good reputation amongst employers for the platform

These features in particular would bring a lot of value to the application and users. The addition of these features would allow the platform to compete better against state-of-the-art applications. However, many other features could be considered and prioritised over the stated.

## 8.4 – Legal, social, ethical and professional issues

The website does not infringe on any legal laws. All images and icons published on the website were copyright-free or created. All user details were classed as sensitive so they were protected and stored in a password-protected database. The application also does not make unethical remarks so no users are offended.

## 8.5 – Summary

Overall, the project’s core objectives were fully met by providing a web application that allowed users to compete in challenges alone or against other users. The project is a great starting point but it needs improvement to compete against state-of-the-art competitors.

# Bibliography

[1] CORBETT, J. O., BARWOOD, M. A. R. T. I. N. J., OUZOUNOGLOU, A. L. E. X., THELWELL, R. I. C. H. A. R. D., & DICKS, M. A. T. T. H. E. W. (2012). Influence of competition on performance and pacing during cycling exercise. *Medicine & Science in Sports & Exercise*, *44*(3), 509–515. https://doi.org/10.1249/mss.0b013e31823378b1

[2] Landers, R. N., Collmus, A. B., & Williams, H. (2019). The greatest battle is within us: An experiment on the effects of competition alone on Task Performance. *International Journal of Human-Computer Studies*, *127*, 51–61. https://doi.org/10.1016/j.ijhcs.2018.09.011

[3] Cagiltay, N. E., Ozcelik, E., & Ozcelik, N. S. (2015). The effect of competition on learning in games. *Computers & Education*, *87*, 35–41. https://doi.org/10.1016/j.compedu.2015.04.001

[4] Ã–zer, H. H., Kanbul, S., & Ozdamli, F. (2018). Effects of the gamification-supported flipped classroom model on the attitudes and opinions regarding game-coding education. *International Journal of Emerging Technologies in Learning (IJET)*, *13*(01), 109. https://doi.org/10.3991/ijet.v13i01.7634

[5] Fischer, K., Vaupel, S., Heller, N., Mader, S., & Bry, F. (2021). Effects of competitive coding games on novice programmers. *Educating Engineers for Future Industrial Revolutions*, 464–475.

[6] Bhandari, P. (2022, November 24). *What is quantitative research? Definition uses & methods*. Scribbr. Retrieved December 4, 2022, from https://www.scribbr.com/methodology/quantitative-research/

[7] Bhandari, P. (2022, November 24). *What is qualitative research? Methods & examples*. Scribbr. Retrieved December 4, 2022, from <https://www.scribbr.com/methodology/qualitative-research/>

[8] Amlani, R. D. (2012). Advantages and Limitations of Different SDLC Models. International Journal of Computer Applications & Information Technology Issue III, I(6). http://www.ijcait.com/IJCAIT/13/1334.pdf

[9] Richards, M. (2015) *Software Architecture Patterns (Report).* O’Reilly Media, Inc.

[10] Seidl, M. (2015). *Uml @ classroom: An introduction to object-oriented modelling*. Springer International Publishing Switzerland.

[11] *Activity diagrams* (no date) *UML activity diagrams*. Available at: https://www.ibm.com/docs/en/rsm/7.5.0?topic=diagrams-activity (Accessed: March 4, 2023).

[12] Tiun, S., Mokhtar, U. A., Bakar, S. H., & Saad, S. (2020). Classification of functional and non-functional requirements in software requirements using word2vec and fast text. *Journal of Physics: Conference Series*, *1529*(4), 042077. https://doi.org/10.1088/1742-6596/1529/4/042077

[13] Dabbagh, M., Lee, S. P., & Parizi, R. M. (2015). Functional and non-functional requirements prioritization: Empirical evaluation of IPA, AHP-based, and Ham-based approaches. *Soft Computing*, *20*(11), 4497–4520. https://doi.org/10.1007/s00500-015-1760-z

[14] Atlassian. (n.d.). *User stories: Examples and template*. Atlassian. Retrieved March 2, 2023, from https://www.atlassian.com/agile/project-management/user-stories

[15] Atlassian. (n.d.). *What is a Kanban Board?* Atlassian. Retrieved March 2, 2023, from https://www.atlassian.com/agile/kanban/boards

[16] Drumm, I. (2020, November). Design Patterns 6 - Creational Patterns. Lecture.

[17] Contributor, T.T. (2020) *What is GitLab? Definition from TechTarget*, *WhatIs.com*. TechTarget. Available at: https://www.techtarget.com/whatis/definition/GitLab (Accessed: March 25, 2023).

[18] Fosco, M. (2022) *What is a CI/CD pipeline?* *CircleCI*. CircleCI. Available at: https://circleci.com/blog/what-is-a-ci-cd-pipeline (Accessed: March 25, 2023).

[19] Zampetti, F. *et al.* (2021) “CI/CD pipelines evolution and restructuring: A qualitative and quantitative study,” *2021 IEEE International Conference on Software Maintenance and Evolution (ICSME)* [Preprint]. Available at: <https://doi.org/10.1109/icsme52107.2021.00048>.

[20] Wong, W.E. *et al.* (no date) “A study of effective regression testing in practice,” *Proceedings The Eighth International Symposium on Software Reliability Engineering* [Preprint]. Available at: <https://doi.org/10.1109/issre.1997.630875>.

[21] McGregor, J.D. (2007) “Test early, test often.,” *The Journal of Object Technology*, 6(4), p. 7. Available at: https://doi.org/10.5381/jot.2007.6.4.c1.

[22] George, B. and Williams, L. (2004) “A structured experiment of test-driven development,” *Information and Software Technology*, 46(5), pp. 337–342. Available at: https://doi.org/10.1016/j.infsof.2003.09.011

# Appendices

## Appendix A - Project Proposal

### Aim

This project aims to build a web application that creates a fun and challenging experience for software engineers that are preparing for potential technical interview questions. The engineers will be able to compete against other engineers in solving algorithmic problems. The problems will consist of a problem description, some template code, an input, and the desired output. The engineers will need to add to the template code to get the desired output. The engineer that gets the desired output and passes all test cases first will win the competition. The project will also offer to do this in isolation, similar to what Leetcode offers.

### Motivation

One motivation point behind this project is the value it will bring to the Computer Science industry, especially for junior developers. It will allow them to develop their problem-solving ability whilst having fun. Junior developers can make use of this skill when applying for jobs.

Although it was very possible to gain a lot of web development experience during the first and second years of university, the choices of learning new frameworks and languages were limited. I believe this project will be complex therefore it will challenge my core computing knowledge and coding skills. This project will also provide opportunities to learn widely used industry standard front-end frameworks/libraries such as React, and a backend framework such as Springboot Java. Thus, boosting overall career opportunities and helping to improve my skill set.

### Background

Leetcode was founded in 2015 and it is used by many developers. Its primary focus is to prepare software engineers for potential interview questions. Leetcode consists of 2431 problems that a user can solve. These problems usually are composed of a description of the problem, an input, the desired output, and some template code that you alter to get the desired output. All problems are designed to test an engineer's knowledge surrounding efficiency, patterns and problem-solving.

When developers apply for jobs, they usually use platforms such as Leetcode to prepare for their technical interviews. Leetcode offers the service of solving algorithmic problems but in isolation. This is the gap this project is aiming to fill. Developers should enjoy the process of preparing for interviews. This project will have similar functionality to Leetcode, but it will have an additional feature of allowing users to compete against other users in solving problems. This will increase the chances of improvement for engineers as they will be able to view their competitor’s code and understand different perspectives.

### Who will benefit?

This project is aimed at any developers that are looking to prepare for their technical interviews. When applying for jobs at the current time developers should anticipate that they will receive a challenge of some sort that they need to resolve. Usually, this challenge is some sort of algorithmic question where you receive some sort of input, and you must code a method that produces the desired output. So, the developers will benefit from this project as it is contrary to the typical type of website they would visit to prepare for technical interviews. This project will add an element of friendly competition and fun to the process. It will make the process of preparing for interviews a lot more fun.

Developers who are casually looking to develop their problem-solving ability, learn a new language, or want to have some fun with friends can also benefit from this project. As this project will force them to solve problems, learn new syntax & put them up against their friends to see who solves a problem first.

### Objectives

* **Objective 1: Conduct background research on how similar web applications work & make a list of required features for my project**

***17/10/22 - 20/11/22***

* + Research how to compile code on a website to perform code execution in the safest manner possible.
  + Research similar web applications to understand the approach they may be taking.
  + Research web sockets to understand how they work and why I will need them.
* **Objective 2: Develop a data-driven web application that allows users to practice code and compete against each other**

***21/11/22 - 23/04/23***

* + Design a friendly UI while focusing on accessibility & usability
  + Create a quality database for the project that cannot be simplified any further.
  + Implement a quality UI that is responsive to changes in screen size.
  + Develop a user profile system that allows users to register, log in, edit and update their details.
  + Develop a feature that allows users to compete against each other.
  + Create an algorithm that helps match users of the same ability
* **Objective 3: Perform testing on the application covering common use cases, and edge cases & generate a test report**

***28/11/22 – 23/04/23***

* + Research and identify suitable testing frameworks for chosen technologies
  + Carry out a wide range of testing on usability and accessibility (front end)
  + Perform unit testing (back end)
  + Perform user testing
* **Objective 4: Implement DevOps such as CI/CD for deployment & ensure the pipeline generates test reports and successfully deploys repositories**

***03/04/23 – 23/04/23***

* + Create CI/CD pipelines that generate test reports and deploy the repositories
  + Use Docker to create an image that stores the code-compiling API.
* **Objective 5: Write my dissertation**

***17/10/22 – 23/04/23***

### Development Requirements

There are no specific hardware requirements for this project.

The software requirements are:

* Visual Studio Code – IDE widely used for Web Development. To be used for the development of the front end and to run the front end.
* IntelliJ – IDE widely used for Java Development. To be used for the development of the backend and to run the backend.
* MySQL – relational database management system used for storing and manipulating data & defining relationships between tables. To be used to view & edit the database for this application.
* Node.js – JavaScript runtime environment. To be used to run the front end.
* JDK – Java Development Kit which includes tools for developing and testing programs. To be used to run my backend.
* Docker – open-source containerization platform – To be used to create an image of an API and run the image within a container.
* Heroku – Cloud platform as a service – To be used to deploy my Java repositories.
* React – front-end JavaScript library – To be used to develop the front end of this project
* Springboot – backend Java framework – To be used to develop the back end of this project.

### Methodology

The approach that I desire to take on this project is to deliver the project in small increments and not the whole project in one go. Therefore, I will be using the agile methodology.

The agile methodology is based on an approach that is incremental and iterative so naturally, it is open to more changes during the development period according to customer requirements. Considering I don’t have much experience with React this approach will be beneficial as it will allow me to consistently make improvements as opposed to the Waterfall methodology.

By using Scrum with Agile methodology, I will be presented with the opportunity to divide my work into sprints which will allow me to focus on specific features in each specific sprint and that will result in me producing a high-quality product.

I will also have the opportunity to showcase my progress after each sprint and gather feedback accordingly. I also have the opportunity to evaluate my approach after each sprint where I can think about what worked well, what did not go as planned and what can be done to avoid such things in the future.

Each sprint will last for a week. Here is a breakdown of how many sprints my objectives will cover.

* **Objective 1**
  + Sprint 1 to 5
* **Objective 2**
  + Sprint 6 to 19
    - The contingency plan for this objective will cover sprints 20 to 27.
* **Objective 3** 
  + Sprint 7 to 27
* **Objective 4**
  + Sprint 25 to 27
* **Objective 5**
  + Sprint 1 to 27

### Timeline Description automatically generatedGannt Chart

## Appendix B - Project logbook

## Final Year Project – Logbook

### Trimester 1

#### Week 1 – Wednesday - 21st September (3 hours) – Lecture & Self Study

Dr Murray hosted a lecture surrounding the final year project. It was only an introductory lecture so we were told what materials to familiarize ourselves with, what was expected from us and how our work would be marked. The kickoff lecture was two hours, I spent an hour after the lecture reading the documents that Dr Murray highlighted in the lecture.

#### Week 1 – Thursday - 22nd September (5 hours) – Self Study

I am still in the research phase for my final year project idea. However, I am pretty certain that I would like to develop something that will be hosted on the web. I aspire to take on an application that will challenge me in an abundance. I have already decided that React will be used as a front-end technology & following this decision I have decided to buy a course to further expand my knowledge of the library hence I spent a good few hours going through this (<https://www.udemy.com/course/react-the-complete-guide-incl-redux/> ) course. I went through two sections of the course “Getting Started” & “JavaScript Refresher”. I am already familiar with React so I’ll be using this course as a refresher.

#### Week 2 – Monday - 26th September (2 hours) – Lecture & Seminar

Dr Murray hosted a lecture for an hour, this particular lecture was surrounding our project proposal. He expanded on points such as ‘What makes a good project proposal?’. This lecture will help me as we thoroughly went through the mark scheme and I now understand what a good project proposal is composed of.

The second hour was a seminar. As no one has been given any supervisors yet we were instructed to join our prospected supervisors’ teams call. As I have previously mentioned I aspire to create a web application so I joined Stefan’s and Shadab’s call. I also now had an idea in mind so it was great to receive feedback on this idea. My idea is to create a replicate of Leetcode (<https://leetcode.com/problemset/all/> ) however I will add extra functionality that will allow users to solve problems live at the same time and against each other. I also presented this idea to Stefan and his feedback was positive. I understand the complexity of this project is high & I like that as I enjoy a challenge.

Now I have a concrete idea of what I am going to do, I can advance in the research and learning processes.

#### Week 2 – Wednesday - 28th September (5 hours) – Self Study

I have continued to progress through the course that I have previously purchased. I worked through a large and core segment of the course – “React Basics & Working with Components”. This helped me understand component-based development. It introduced the idea of how certain components on a web page can be reused. The section was a little around 2 hours but I spent some extra time studying the topic so I could ensure my fundamentals are solid.

#### Week 3 – Monday - 03rd October (2 hours) – Lecture & Seminar

Dr Bass hosted a lecture for an hour. The lecture’s theme was Development methods. This lecture mainly covered the Agile methodology. It broke down the stages of each iteration and explained each one in detail. We covered the following stages

* Plan
* Analyse
* Design
* Build
* Test
* Deploy

Not only did we cover these stages and what they consisted of but they were presented in the context of an individual. This was very beneficial as I came to understand that Agile would probably be the best methodology to use for my project. We also mentioned Architecture styles and Service-oriented was mentioned. This has planted a seed in my head and I will research further into the architecture as I believe I can make use of it and I intend to.

The second hour consisted of a seminar where we joined our appointed supervisors. I, fortunately, got my first choice, so as of now my supervisors are Stephan & Shadab. As we are still in the early stages of the year, the workshop was structured very similarly to the previous one. My peers that were unsure about their project would present their ideas and get some clarification on the complexity and suitability of it. I also asked a few questions about being rewarded for implementing CI/CD and using test-driven development (TDD) as a development approach. I now know both will be rewarded and will add value to my project. So I will consider this when developing my project and setting my objectives in my project proposal.

#### Week 3 – Wednesday – 5th October (6 hours) – Self Study

I have continued to progress through the React course that I have been undertaking. I worked through a large and core segment of the course – “React State & Working with Components”. I am beginning to feel a lot more comfortable as I have recapped most of the fundamentals now. I have decided to stop dedicating time to this course now. In the future, I will refer back to this course if there are any issues. As I have recapped most of the fundamentals of React, I believe I can develop applications with the occasional help of online forums.

#### Week 4 – Monday - 10th October (2 hours) – Lecture & Seminar

Dr Chris Hughes hosted a lecture for an hour. The lecture’s theme was Research methods. Dr Chris made a critical point about picking a final-year project that we are interested in as we will be spending a lot of time on it. I took some time to evaluate and I can say that I am happy with my project idea. He also expanded on topics surrounding research and how to conduct it. I will be able to make use of this information in the coming weeks as my first few objectives are based on research.

The second hour consisted of the usual workshop with our groups & supervisors. The main topic in the workshop was how we should be making our decisions as this is an individual project. One of my colleagues offered to share his project proposal whilst Stephan evaluated it. This evaluation helped me identify some improvements that I can implement in my project proposal.

#### Week 4 – Wednesday - 12th October (6 hours) – Self Study

I started to create a mini-project that would test my knowledge of React fundamentals. This exercise reiterated my knowledge.

I also started to read a book called Software Architecture Patterns by Mark Richards. I briefly started this book but I aim to give it a thorough look in future as I believe it will help me decide what particular architectural pattern I will be using.

I also received feedback on my project proposal from my appointed supervisor. So, I worked on improving my project proposal with the feedback in mind, as well as the review that was given to my colleague in the seminar.

#### Week 5 – Monday 17th October (2 hours) – Lecture & Seminar

Lee Griffins hosted a lecture today surrounding the importance of being realistic and how coding is not worth a significant amount. The lecture highlighted the importance of having a balanced final-year project. So, he emphasised the point of prioritising research but also have a well-rounded application that offers a good amount of functionality.

In the seminar, the whole group went through their ideas and our supervisors reflected upon these ideas. As I had already presented my idea and got feedback on it, I just listened to this session.

#### Week 5 – Wednesday 19th October (8 hours) – Self-study

I spent some time researching similar web applications and what features they offer.

I looked at the following sites :

* Leetcode
* HackerRank

I documented the research to help me identify patterns of success. For example, I can conclude that 3 particular characteristics are needed for my site to be successful :

* A professional and responsive user interface.
* Allowing users to submit code in a variety of programming languages.
* Allowing users to solve problems offline.

The research helped me identify the minimum requirements of my application and how my application will be unique.

I also spent some time proofreading my project proposal and making any last-minute changes as the deadline was close.

#### Week 6 – Monday 24th October (2 hours) – Lecture & Seminar

Dr Bass conducted a talk about writing our introduction and literature review. It was a short lecture as it started late. It was short but informative. I will have to revisit the content to revise it.

The second hour consisted of a seminar with my supervisors and the web development group. As some members were absent last week, the supervisors focused on these individuals and ask them about their ideas and gave feedback to them. As I had already got my idea I just listened to this particular session.

#### Week 6 – Wednesday 26th October & Thursday 27th October (8 hours) – Self study

The project research is now well underway. I have started to research topics for my Literature review & Methodology. Some research papers that I have looked into are:

* Web sockets and how they work
* Effects of Competition on an Individual’s Performance
* Influence of Competition on a Performance
* Why do people like competition

I also went back to my previous documentation of similar applications and extended it to fully understand the void that I will be filling with my application.

#### Week 7 – Monday 31st October (2 hours) – Lecture & Seminar

Dr Chris Hughes hosted a lecture for an hour. The lecture’s theme was based on designing our research methodology for our project. This was very informative as I plan to complete my research methodology assignment in the upcoming week.

The second hour consisted of a stand-up with the web development group. This week it was more of an open-ended discussion and I gathered some information to clarify the relevance of the research I’ve previously made. I found research that says people who lose a lot whilst competing, actually don’t like competing. So this suggests I should add a feature of allowing users to complete problems in isolation. This feature was also highlighted as a must-have from the research of similar web applications.

#### Week 7 – Wednesday 2nd November (6 hours) – Self Study

I have continued to research. I have discovered the following articles :

* Game-Based Coding challenges to foster programming practises.
* Coding as a Sport.
* Effects of Gamification supported flipped classroom model on the attitudes and opinions regarding game-coding education.

I aim to build my repository of research papers by next Monday. I then plan to start my literature review.

I will take some time out this week as it is my birthday tomorrow.

#### Week 8 – Monday 7th November (2 hours) – Lecture & Seminar

Dr Ian Drum hosted a lecture for an hour. The lecture was about Software Architecture. This was an interesting lecture and I enjoyed it. It was very useful for me in particular as I could relate and think about what kind of architecture I will be using in my backend. It was also good to reiterate some knowledge.

The second hour consisted of the usual seminar with the web development group. I got the opportunity to ask some questions surrounding the literature review.

#### Week 8 – Wednesday 9th November ( 6 hours) – Self Study

I have started to write up my literature review. I have multiple articles surrounding the topics of:

* How competition affects an individual’s performance
* How competition affects learning
* How competition can be applied to learning coding
* Competitive coding

Research into suitable development methodologies and research methodologies has started. This research will allow me to write a detailed methodology section.

In the following weeks, I aim to complete my literature review & methodology so I can commence designing my application.

#### Week 9 – Monday 14th November ( 1 hour) – Seminar

The final year project lectures for this semester finished last week.

The usual seminar was still running. This week I took the opportunity to clarify the structure of my literature review. I also took the opportunity to discuss some research that I have come across.

#### Week 9 – Wednesday 16th November ( 5 hours) – Self Study

The first draft of my literature review has been completed. I have started to write up my Methodology section. The target is to finish the methodology section by the end of this week. This will allow me to receive feedback and whilst I wait, I can dedicate some time to other modules. The start of December includes various deadlines so I’ve to focus on multiple assignments.

#### Week 10 – Monday 21st (1 hour) – Seminar

A seminar with the web development group was held. At this point, the meetings were kind of unstructured and just Q&As. Most of my questions surrounded the literature review & methodology so I could iron out any major issues before sending my draft off for feedback.

#### Week 10 – Wednesday 23rd November (6 hours) – Self Study

The first draft of my literature review and methodology has been completed. I have sent the document to my supervisors for review. I await a response, in the meantime, I have been conducting research on one of the more complex features in my application, live competitive coding. The research was based on how to use web sockets using Java. During the research, I thought it would be a good idea if my web socket application to use an in-memory database. Using an in-memory database would be beneficial as it will allow for faster response times due to faster data retrieval. Data is not needed permanently for live competitive coding, once a lobby shuts the data can be disposed of.

I have also started to think about my backend service and what its structure of it would be like. I haven’t finalised any thoughts on it just yet.

#### Week 11 – Monday 28th November (30 minutes) – Seminar

A seminar with the web development group was held. As there weren’t a lot of attendants this week the meeting was concluded early. This was the last final-year project seminar for the first semester.

#### Week 11 – Wednesday 30th November (7 hours) – Self Study

I received feedback on my literature review and methodology, so, I decided to dedicate all my time to that as the submission deadline is the 5th of December. I also read a book about software architectures and a comparison of them. I have decided to use the microservice architecture for my application. Here is a proposed architectural design of my application.

Diagram

Description automatically generated

#### Week 12 & 13 – Monday 5th December – 18th December – Self Study (Other modules)

I dedicated all of my time to all of the other modules. All of my module’s deadlines fell after the literature review & methodology so I have decided to temporarily focus on them. I aim to start the designing phase after all of our assignments have been handed in.

### Christmas Holidays

#### Week 14 – Monday 19th – Sunday 25th December (15 hours) – Self Study

Semester one teaching is now over. I will dedicate all of my spare time to the final year project. I will document the work done on the project weekly, this is because I will have a lot more time to spend on the project.

The first stage was requirement elicitation. I needed to produce a list of requirements for my minimum viable product. The requirements that I produced are :

* Allow users to log in/register.
* Allow users to solve problems in isolation.
* Allow users to compete against other users.
* Have a profile management page.
* Give users a coding environment.
* Have a library of coding challenges.

These requirements are abstract so they will have sub-tasks to achieve them.

I also commenced the designing phase. I used Figma to design the front end of the application. I designed pages for all of the requirements that I have produced so far. For the backend of the application I already have a proposed design but I created a use case diagram as well. I also created a system design diagram to get an overview of the application and its architecture.

Graphical user interface, website

Description automatically generated

I also researched how to set up a CI/CD pipeline so I can streamline deployment & integration. This process will also help me identify any introduced bugs from new code as the pipeline will run and will fail if any existing unit tests fail.

#### Week 15 – Monday 26th December – Sunday 1st January (30 Hours) – Self Study

As the designs for the MVP were now completed, I started the development phase.

I created a Jira board and integrated it with my Repositories.

Graphical user interface, application, Teams

Description automatically generated

As mentioned previously the requirements were abstract so these user stories had child user stories. Here is one example :

Graphical user interface

Description automatically generated

Before I began implementing any functionality, I had to focus on implementing the user interface. For example, I needed to create a homepage, navigation bar and footer. I made all of these components and placed them under a ‘Common Components’ folder.

Graphical user interface, text, application

Description automatically generated

This folder structure of this project streamlined the development process as it kept the code tidy. I also took advantage of a core benefit of React, reusing components. I declared & created these components once and only once; however I used them on multiple pages by simply importing them. The Footer & Navigation don’t need to be justified as their name gives away their purpose.

I created the Box component to reuse a container-like box for content on each page. For example, in the screenshot below you can see the 4 boxes contain different content but in actuality, they have a lot of similarities. I used the Box component for each of these components. You can also see from the screenshot that all Logos are of the same size, this is because the Logo component was used to keep them consistent.

Graphical user interface

Description automatically generated

The LoadingSpinner component was made to display a loading spinner on the screen whilst the client side sends an HTTP request to the back end and retrieves and displays the data from the response.

After the core components of the application were created. I moved on to develop the login & Register functionality of the system. To develop the backend, I used the approach of test-driven development (TDD). I used the Spring Security dependency in my backend to manage user authentication and authorization.

#### Week 16 – Monday 2nd – Sunday 8th January (40 hours) – Self Study

After completing the login and register functionality and user interface. I decided to move on to the most challenging features of the application. Participating in coding one vs one and completing problems alone. I thought it would be a promising idea to complete the latter first as it could be re-used in the former.

I faced an abundance of problems. The first problem I faced was compiling code. When a user sent code to the backend, I needed to compile it, run it to see the output the code gives, and compare it against the test cases. The main problem faced was compiling the code but after some research, I ran into the Process class in Java that would allow me to do what I desired.

Process class Java Documentation - <https://docs.oracle.com/javase/8/docs/api/java/lang/Process.html>

After setting up the compilation of code, I set up the comparison which was relatively easy as I controlled the test cases, as they are from the application's database.

At the moment a User can only run code in Java & Python.

I made use of existing code packages within the node package manager. For example, after some research, a code editor called ace editor can be used on websites. This editor supports advanced features such as auto-completing. The development of the feature, where a user could solve a problem alone was easy compared to the development of the feature where users can compete against each other. Upon research, I gathered I would have to make use of web sockets. I was unfamiliar with how web sockets work so I watched a YouTube tutorial for making a sample application in Springboot using web sockets.

WebSocket tutorial - <https://www.youtube.com/watch?v=U4lqTmFmbAM>

I learnt how to work web sockets in spring-boot and implemented the example in the video to further investigate.

By the end of the week, I had completed the feature of allowing users to solve problems in isolation but I had only half completed the competitive coding feature.

#### Week 17 – Monday 9th – Sunday 15th January (30 hours) – Self Study

As I did not finish the competitive coding feature last week, this week I continued to develop that particular feature.

By the end of the week, I could successfully, create a lobby, join a lobby, start a lobby and participate in a coding one vs one. I also successfully developed a profile management page that was fully functional. As Semester Two commences on the 16th of January I have decided to take some time out and wait for some feedback from my supervisors on the application. I will commence development again when I have received feedback.

I re-used some components of the solving problems in isolation feature, I added a boolean parameter to the components called “competition” meaning they act differently when a user is competing against somebody else. This approach helped speed up the process of creating the components for the competitive coding feature.

### Trimester 2

#### Week 18 – Monday 16th January (1 hour) – Seminar

The second semester commenced today. There are only three final-year project lectures for this semester, which are all later on in the semester. A group seminar was held with the web development group. I took this opportunity to present my work from the holidays to the rest of the group. Here is a quick summary of the feedback I received from the supervisors :

* Prioritise highlighting the unique features your platform has
* Make current features robust and high quality instead of focusing on quantity
* Conduct user testing
* Start planning for the final demonstration of the product
  + For example, I may need two computers to showcase my project
* Add more problems
  + Around 10 – with varying difficulty

Overall the feedback received was positive. Stephan and Shadab were extremely impressed with my project. The feedback helped shift my aims. I aim to make this project as robust as possible and solidify the existing features. I also aim to deploy my project within the week, so, I can assess the application with the rest of the group in the next seminar.

#### Week 18 – Saturday 28th & Sunday 29th January (12 hours) – Self Study

I started the process of applying the feedback I had received earlier in the week. I attempted to first deploy the project and then apply any changes needed.

The deployment of the project resulted in many problems. First of all, I was getting an error to do with cross-origins, as I have already dealt with this error before I figured out the root cause and fixed it. Another problem that I was facing was, my backend API compiles and runs Python as well as Java but on my server, I didn’t have the option of downloading Python. This is where I saw a perfect opportunity to use Docker. I created a Dockerfile that would create a docker image of my project and then that image would be hosted on my server rather than my raw project. The Docker image downloads both Java and Python, so, when I uploaded that specific Docker image to my server it was working. Below is a snippet of my Docker configuration file with comments.

Text

Description automatically generated

After creating the docker image, I decided to build a pipeline that would build, test and deploy my project when I pushed code to my repository. My pipeline contains three separate stages build, test and deploy (This stage only runs if the code is pushed to the master branch). First, the pipeline ensures the project can be built, then all of the unit tests are run and finally, the project is deployed on the Server. As you can see below the pipeline also produces a test report which is highlighted in green.

Graphical user interface, text, application, email

Description automatically generated

I also spent time manual testing and mitigating bugs. I have created a Jira ticket that tracks all of my bugs.

Graphical user interface, text, application

Description automatically generated

For the next seminar, I aim to ask some questions:

* What kind of questions should I include in my questionnaire for user testing?
* What is the best/more preferred structure of the dissertation?
* Do I have to use the University server?
* Can we test my application as a group in the seminar (We all try it out as normal users in the seminar)?
* What is the progress with the ethics application?

I have deployed the project but at the moment I am not satisfied with it to conduct user testing so I will have to delay the user testing by a week.

#### Week 19 – Monday 23rd January (30 minutes) – Seminar

A group seminar was held with the web development group. I took this opportunity to ask the questions from my previous logbook entry. All of my questions were answered and I plan to test my website out with the rest of the group next week if everything is okay and I am satisfied that I have covered all of the obvious bugs.

#### Week 19 – Saturday 28th & Sunday 29th January (16 hours) – Self Study

I spent the duration of the weekend bug fixing. I followed the cycle below.

I aim to start my dissertation next week. I have also prepared a Google form ahead of my live test with the group. In total I mitigated around 30 bugs, so my solution is now robust, secure and high quality.

#### Week 20 – Monday 30th January (1 Hour) – Seminar

A group seminar was held with the web development group. One of the supervisors was missing. This seminar consisted of a few of us giving through our progress with our project. I didn’t showcase much as I have already presented my project. The opportunity to test my application with the rest of the group didn’t arise so I will look to do that next week but I am aiming to test it with some friends over the week.

#### Week 20 – Wednesday 1st February (3 Hours) – Self Study

A screenshot of a computer

Description automatically generated with medium confidenceI gathered friends to conduct user testing on my application. A screenshot of a computer

Description automatically generated with medium confidenceGraphical user interface

Description automatically generated

The user testing gave me an idea of how robust my application is. Overall, the performance and feedback were great. Bugs have arisen and I will look to mitigate them in the next week as I will be away from tomorrow.

#### Week 21 – Monday 6th February (1 Hour) – Seminar

A group seminar was held with the web development group. This seminar consisted of a few of us giving through our progress with our project. I did not showcase much as I have already presented my project. As I was away, the opportunity to test my application with the rest of the group didn’t arise so I will look to do that next week.

#### Week 22 – Monday 13th February (1 Hour) – Seminar

As I was unwell for the remainder of the last week, I didn’t make much progress.

As usual, a group seminar was held with the web development group. In this seminar, I had the opportunity to showcase my project again. This time the supervisors wanted to know a little more about the functionality of the application, in particular how code is run on the application. After explaining to the supervisors how the code runs, an obvious and dangerous security implication was highlighted. At the moment, there is nothing in place that will filter out malicious code in my backend. This was a great spot by the supervisors and I aim to mitigate this risk as soon as possible.

#### Week 22 – Wednesday 15th February (5 Hours) – Self Study

As my application development is ending, I decided to start on the main components of the dissertation. I have started with the requirements specification and design document. I managed to complete a draft version of the document. I have sent off the draft document to my supervisor for review and await a response. In the meantime, I will be focusing on mitigating the potential risk that was pointed out to me in the weekly seminar.

Graphical user interface, website

Description automatically generated

Diagram

Description automatically generated

#### Week 23 – Wednesday 22nd & Thursday 23rd February (10 Hours) – Self Study

This week I could not attend the weekly seminar with my group.

I aimed to focus on minimizing the serious security threat of allowing users to run code on my server. potential solutions could be:

* Not allowing users to create/delete files.
* Restrict certain libraries.
* Run the code in a sandbox environment.
* Find an external API that allows me to compile code.

With some research, I found the Piston API - <https://github.com/engineer-man/piston>. I thought finding an external API that allows me to compile code was the best solution as it didn’t require me to reinvent the wheel. The downside of this is it adds an external dependency to my project. The Piston API is secure as it uses docker to sandbox the code. The majority of my time was spent on refactoring my code and fixing existing unit tests.

#### Week 24 – Monday 27th February (1 Hour) – Seminar

A group seminar was held with the web development group. I got the opportunity to display my progress once again. I showed Stefan and Shadab how I migrated the code compilation code out, and now rely on a third-party API that compiles code and returns the output. This migration saved me a lot of time as it prevented me from ‘re-inventing the wheel.’ The third-party API is called Piston. The Piston API has comprehensive documentation of how it covers security issues. The API also can run/compile up to twenty-five programming languages which are a lot more than my previous 2 languages. With my development ending, I am going to dedicate my time to my dissertation from this week onwards.

#### Week 25 – Monday 6th March (1 Hour) - Seminar

A group seminar was held with the web development group. I took the opportunity to educate myself on an appropriate structure for my dissertation. The web development example was presented in increments but I did not work that way so I learnt it was okay to not follow the example in terms of structure.

#### Week 25 – Wednesday 8th March (8 Hours) – Self Study

I dedicated my time to improving the requirements and design document. I had received feedback from Shadab, so, I knew what exactly to improve. The feedback I got was to:

* Split requirements into functional and non-functional
* Add use case diagram
* Add an activity diagram
* Add ER diagram of the database

I successfully implemented all of the feedback.

#### Week 26 – Monday 13th March (1 Hour) – Seminar

Another group seminar was held with the web development group. I didn’t get to ask many questions, as I had already been heavily involved and asked a lot of questions, and the supervisors wanted to hear about some of the others.

#### Week 26 – Wednesday 15th March (5 Hours) – Self Study

I decided to start writing up the development and implementation section of my dissertation. I managed to get half of the section done but I am a little confused about the structure of the section.

I have two questions to ask in the next seminar:

* What’s the best structure for this section
* What would the supervisors like to see

#### Week 27 – Monday 20th March (1 Hour)

Another group seminar was held with the web development group. The supervisors stated the best structure for the development and implementation section was breaking down how each requirement was developed and then showing screenshots of the requirement implemented. They want to see plenty of diagrams. This was the last group seminar for the year.

### Easter Holidays

#### Week 28 – Week 29 - (40 Hours)

I completed the dissertation two weeks prior. The dissertation is in for week 31, April 17th. I have completed the dissertation early, so, I can review the document and keep making continuous improvements.

### Reflection

I thoroughly enjoyed working under the supervision of Shadab and Stefan. The feedback and technical guidance they offered me were second to none. I thoroughly enjoyed the process of developing my final year project. I created a full-stack application that was incredibly challenging. I managed to create an application that allows users to compete against each other in real-time. The users also can solve challenges alone. I learnt about an abundance of things, overall the experience has made me a better engineer. The hard work and time put into the project paid off several times in interviews I had throughout the year. Many times interviewers were shocked I took so much on for a final-year project. The inclusion of Jira to track issues, Jira being attached to my repositories, the creation of a CI/CD pipeline and an array of different testing methods helped bring the value and realism of my project up. These practices are used in the real world. Overall, I am happy with the final product.