**A PERSONALISED LEARNING PLATFORM USING MACHINE LEARNING**

INTERIM PROJECT REPORT

FRED OJOGU

21030688

A thesis submitted in part fulfilment of the degree of BSc (Hons) Computer Science

Supervisor: Dr. Kulvinder Panesar

**Declaration**

The candidate confirms that the work submitted is his own and that appropriate credit has been given where reference has been made to the work of others. The candidate agrees that this report can be electronically checked for plagiarism.

Fred Ojogu

# **Abstract**

Education is evolving rapidly with the rise of online learning platforms. However, many of these platforms deliver a uniform experience that fails to address the individual needs of learners. This report presents a machine learning-powered personalized learning platform that adapts dynamically to a learner’s preferences, behaviour, and performance. The primary objective of this project is to implement a recommendation system and adaptive testing mechanism to ensure students are provided with customized learning experiences. The project utilizes collaborative filtering and predictive analytics for performance monitoring. The outcomes indicate that personalized learning platforms can significantly enhance user engagement and improve learning outcomes.

Contents

[**Abstract** 3](#_Toc186939475)

[Contents 4](#_Toc186939476)

[**Chapter 1: INTRODUCTION** 5](#_Toc186939477)

[**1.1** **Problem description** 5](#_Toc186939478)

[**1.2** **Aims and objectives** 5](#_Toc186939479)

[**1.3** **REPORT OVERVIEW** 5](#_Toc186939480)

[**Chapter 2: LITERATURE REVIEW** 6](#_Toc186939481)

[**2.1 Overview of Online Learning Platforms** 6](#_Toc186939482)

[2.2 **Importance of Personalised Learning** 6](#_Toc186939483)

[**2.3 Machine learning in education** 7](#_Toc186939484)

[**2.4** **Existing platforms using AI/ML for learning** 7](#_Toc186939485)

[2.5 Firebase Authentication 9](#_Toc186939486)

[**Chapter 3: PROPOSED SOLUTION** 9](#_Toc186939487)

[3.1 **SYSTEM ARCHITECTURE** 9](#_Toc186939488)

[3.2 **FEATURES OF PLATFORM** 10](#_Toc186939489)

[3.3 **PERSONALISATION TECHNIQUES** 11](#_Toc186939490)

[**CHAPTER 4: METHODOLOGY, DESIGN AND IMPLEMENTATION** 11](#_Toc186939491)

[**4.1 PROTOTYPE AND DESIGN** 12](#_Toc186939492)

[4.1.1 Login page 12](#_Toc186939493)

[4.1.2 Register 13](#_Toc186939494)

[4.1.3 Forgotten password 14](#_Toc186939495)

[4.1.4 Dashboard 15](#_Toc186939496)

[4.2 Methodology 16](#_Toc186939497)

[**CHAPTER 5: TESTING** 17](#_Toc186939498)

[**CHAPTER 6: CHALLENGES AND LIMITATIONS(LESPI)** 18](#_Toc186939499)

[**6.1 TECHNICAL CHALLENGES** 18](#_Toc186939500)

[**6.2** **Data Limitations** 18](#_Toc186939501)

[**6.3 USER ADOPTION CHALLENGES** 18](#_Toc186939502)

[**6.4 ETHICAL AND PRIVACY CONCERNS** 19](#_Toc186939503)

[**CHAPTER 7: FUTURE WORK** 19](#_Toc186939504)

[**CHAPTER 8: CONCLUSION** **Error! Bookmark not defined.**](#_Toc186939505)

# **Chapter 1: INTRODUCTION**

## **Problem description**

Programming has evolved into a vital component of the current digital economy, with applications ranging from simple automation to advanced artificial intelligence. However, learning to code is not always straightforward. Many online learning platforms are static and do not accommodate users with varying learning speeds, methodologies, or backgrounds.

For example, students transitioning from non-technical jobs to coding sometimes struggle to grasp complex concepts like recursion or object-oriented programming. Users with previous experience, on the other hand, may find the introduction information repetitive, resulting in disengagement. According to Brown et al. (2021), a lack of tailored instructional approaches leads to a 60% dropout rate in online learning systems.

Personalised learning has emerged as a solution to these difficulties. A platform can discover individual strengths and weaknesses by combining user data and machine learning to provide tailored exercises and real-time feedback. A user who is struggling with Python loops, for example, may be assigned simpler assignments or step-by-step tutorials. Similarly, those that excel at core grammar should be assigned challenges such as algorithms or data structures to keep their learning experience fresh.

## **Aims and objectives**

**Aims**

To create a personalised coding learning platform that uses machine learning to adapt to individual student needs, allowing for tailored educational experiences via adaptive recommendations, real-time progress tracking, and relevant feedback mechanisms. The platform aims to increase engagement, improve coding skills, and cater to users' different skill levels and goals in an ethical, scalable, and inclusive manner.

**Objective**

The following objectives give a clear roadmap for designing, implementing, and deploying the platform, allowing any developer to start from scratch and gradually build the system:

1. Identify target domain
2. Setting up the development environment
3. Data acquisition and preparation
4. Implementing core features
5. System integration
6. Deployment and scalability
7. Evaluation and continuous improvement

## **REPORT OVERVIEW**

The report is divided into 6 main sections, which are:

Chapter 1: Introduction, which shows the problem description, as well as the aims and objectives of the project.

Chapter 2: A Literature review section that will give an overview of what an online learning platform is and highlight some of the tools that will be used in the course of completing this project.

Chapter 3: A requirements section that highlights the functional and non-functional requirement of the platform.

Chapter 4: Design, implementation and methodology section which showcases the user interface, including the login in page, registration page, dashboard and forgotten password page. It will also contain a Gantt chart that will give a concise breakdown of the project plan.

Chapter 5: prototype testing

Chapter 6: Challenges and limitations (LESPI)

Chapter 7: The Future work and conclusion section where there will be a clear statement on work left to be done on the platform and feature the will be integrated into the finished project. It will also contain summary of the work done in the first academic semester.

# **Chapter 2: LITERATURE REVIEW**

This chapter will synthesize and evaluate existing research as well as identify trends and areas for further investigation.

## **2.1 Overview of Online Learning Platforms**

The proliferation of online learning platforms has altered how people acquire technical skills, particularly coding. Platforms such as Codecademy, Coursera, and LeetCode have democratised education by offering interactive exercises, tutorials, and certification programs. Despite widespread popularity, these systems have considerable limitations in terms of meeting individual learning preferences.

1. Codecademy emphasises on interactive coding tasks, however its linear structure limits its ability to change content in reaction to user input. Users frequently report frustration while struggling with a topic because the platform does not deliver remedial content dynamically (Smith et al., 2020).

1. Coursera offers many materials, including video lectures and assignments. However, the platform is geared towards traditional coursework, which lacks real-time engagement and flexibility.
2. LeetCode, a competitive programming platform, succeeds at presenting challenges of varied complexity levels, but it does not consider individual learners' knowledge gaps.

## 2.2 **Importance of Personalised Learning**

Personalised learning tailors educational content to the individual's interests, skills, and weaknesses. Brown & Jones' 2021 study found that learners are 30% more likely to complete courses when using systems that dynamically change content.

1. Increased Engagement: Personalised routes keep users from becoming overwhelmed or distracted with content that is either too easy or too challenging.
2. Better Retention Rates: Research shows that personalised content delivery improves long-term knowledge retention by focussing on areas where users struggle the most.
3. Empowered Learning: Adaptive systems encourage autonomy by allowing users to progress through their educational journey at their own speed.

## **2.3 Machine learning in education**

Machine learning has demonstrated tremendous potential for revolutionising education. Personalised learning experiences can be created using a variety of machine learning methods, ranging from basic supervised learning models to more complex techniques such as deep learning. For example, collaborative filtering algorithms can recommend learning materials based on user interactions, and predictive analytics can predict a student's achievement in specific areas (Minsky et al., 2021).

Below are some key techniques and their applications:

| **Technique** | **Application** | **Examples** |
| --- | --- | --- |
| Collaborative Filtering | Recommends exercises based on similar users' preferences. | Suggesting coding challenges to users. |
| Clustering | Groups users into segments based on engagement patterns. | Identifying struggling learners. |
| Reinforcement Learning | Adapts content dynamically based on real-time feedback. | Adjusting problem difficulty dynamically. |
| Natural Language Processing | Analyzes user feedback to improve instructional content. | Sentiment analysis of course reviews. |

## **2.4** **Existing platforms using AI/ML for learning**

Several modern educational systems use machine learning to provide personalised learning experiences. Platforms such as Duolingo use machine learning to adjust lessons and quizzes to the learner's pace, whereas Coursera and Khan Academy use recommendation systems to offer suitable courses or videos. Although these platforms use machine learning to some level, there is still potential for improvement, especially in real-time adaptability and customisation (Roberts & Williams, 2020). Examples of some of these platforms and their features are:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Platform** | **Personalized Learning Pathways** | **Real-Time Feedback** | **Code Recommendation Engine** | **Progress Tracking** | **Gamification Elements** |
| Codecademy | Yes, tailored pathways based on skill level and goals | Yes, interactive feedback in lessons | Yes, suggests additional exercises based on learner progress | Yes, tracks completion, badges | Yes, offers streaks and badges |
| LeetCode | Yes, customizable paths based on proficiency and goals | Limited, mainly test cases | Yes, problem recommendations based on solved problems and difficulty level | Yes, extensive progress tracking and ranking | Yes, provides leaderboards and contests |
| HackerRank | Yes, tailored interview and skill paths | Yes, feedback on correctness and efficiency | Yes, recommends problems based on progress and career goals | Yes, tracks progress, skill scores, and certifications | Yes, points and badges, leaderboard rankings |
| DataCamp | Yes, adaptive learning paths in data science and coding | Yes, instant feedback on exercises | Yes, suggests next courses/modules based on learning path | Yes, tracks skill level and certificates | Yes, skill assessment and certifications |
| Coursera (with Google/Machine Learning Specializations) | Yes, tailored specializations based on skill level and area of interest | Yes, auto-graded assignments with feedback | Limited, no personalized recommendations within coding | Yes, detailed progress and certificate tracking | Yes, certificates and achievements |
| EdX (Harvard CS50x) | Limited, curriculum follows a structured path | Yes, coding exercises with feedback | Limited, recommends resources but no personalized recommendations | Yes, completion tracking and certificates | Yes, certificates and recognition |
| CodinGame | Yes, customizable challenge and skill tracks | Yes, feedback through interactive coding games | Yes, recommendations based on past challenges | Yes, progress tracking, ranks, and scoring | Yes, ranks, badges, and real-time coding games |
| Brilliant.org | Yes, tailored pathways in math and science coding | Yes, interactive code challenges with feedback | No, no personalized coding recommendations | Yes, tracks skill level and progress in modules | Yes, achievements, badges, and streaks |
| Mimo | Yes, adaptive learning path based on user goals | Yes, interactive, immediate feedback | Yes, suggests modules and challenges based on progress | Yes, tracks daily streaks and progress | Yes, badges, daily streaks, and rewards |

## **2.5 Firebase Authentication**

This service enables secure sign-up and login for users. This solution streamlines the authentication process by offering ready User Interface components and backend services. Another benefit is the convenience of using multiple sources, such as Google or Facebook. I enjoyed the security and ease of application for my Android mobile app. Here's a screenshot of my current test user list using Firebase Authentication.

A screenshot of a computer

Description automatically generated

Fig .1. Firebase User Authentication screenshot. Source: Author, 2024

# **Chapter 3: PROPOSED SOLUTION**

## **SYSTEM ARCHITECTURE**

The proposed customised coding platform combines several components to enable seamless functionality and adaptability. The architecture is organised into three major layers:

1. The frontend, built with React.js, provides an interactive user experience. It supports dynamic dashboards, progress tracking, and an integrated coding environment.
2. Backend: The Flask-based RESTful API manages user data, authentication, and communication with the machine learning module. The backend also handles the secure storage and retrieval of user progress and performance metrics.
3. The Machine Learning Module, built with TensorFlow and Scikit-learn, enables recommendations and adaptive content distribution. Key machine learning tasks include collaborative filtering for exercise recommendations and reinforcement learning for dynamic difficulty modification.

A diagram of a computer system

Description automatically generated

Fig 2. System architecture

## 3.2 **FEATURES OF PLATFORM**

1. Logging In: Gives users access to their accounts.
2. Signing up: Users can create new app accounts.
3. Resetting Password: Users can change their password and log in with a new one.
4. Adaptive exercises: Coding tasks are tailored according to individual progress and performance indicators. The difficulty is dynamically adjusted using reinforcement learning techniques.
5. Progress Tracking: Shows user performance measures including completion time, accuracy, and topic mastery. Uses visual dashboards to identify strengths and opportunities for improvement.
6. Gamification: Uses leaderboards, badges, and milestones to encourage users. Provides "streak bonuses" for persistent daily exercise.
7. Feedback Mechanism: Provides extensive feedback on wrong responses, emphasising specific faults. Provides relevant resources for additional study, such as tutorials or documentation.
8. Personalised roadmaps: Creates a personalised learning plan based on the user's goals, such as learning a specific language or preparing for an interview.

Table 1: Feature Comparison with Existing Platforms

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Proposed Platform** | **Codecademy** | **LeetCode** |
| Logging In | ✅ | ✅ | ✅ |
| Signing up | ✅ | ✅ | ✅ |
| Resetting Password | ✅ | ✅ | ✅ |
| Adaptive Exercises | ✅ | ❌ | ✅ |
| Progress Tracking | ✅ | ✅ | ✅ |
| Gamification | ✅ | ✅ | ❌ |
| Personalised Roadmaps | ✅ | ❌ | ❌ |

## 3.3 **PERSONALISATION TECHNIQUES**

To provide an adaptive and engaging learning experience, the platform uses the following customisation strategies:

1. Collaborative filtering:

Recommends exercises based on user behaviour and preferences. For example, if a user appreciates algorithmic challenges, the system will propose issues solved by other people with similar characteristics.

1. Clustering algorithms:

To discover common challenges, learners are grouped into groups based on their interaction patterns. For example, learners who are suffering with recursion are grouped together and given tailored tutorials.

1. Reinforcement Learning:

Uses reward mechanisms to change exercise difficulty in real time. For example, exercises that are finished too easily are noted for difficulty scaling up.

# **CHAPTER 4: METHODOLOGY, DESIGN AND IMPLEMENTATION**

This section focusses on the first part of the submission, which includes the prototype settings, description, and general implementation (student and teacher panel). The chapter concludes with a discussion on methods. The prototype I created is minimal but meets basic authentication requirements. The platform was designed with:

IDE: Visual Studio code

Database: Firebase

**Steps for running the learning platform**

* Download the latest version of visual studio code
* Open the folder containing the code
* Select “app.py”
* Left click on the code and select “run python” then run in terminal
* Click on the link provided in the terminal

## **4.1 PROTOTYPE AND DESIGN**

The prototype includes:

* Login
* Registration
* Forgotten password
* Email verification
* Dashboard with activity panel and coding area

The login, registration, email verification, and forgotten password features are 100% complete. However, the dashboard containing the activity recommendation, and coding area still needs a lot of work. The prototype displays the landing page some of the requirements, like an area for coding similar to that of a dedicated IDE. All functional and nonfunctional requirements will be developed by the time of the next submission.

### 4.1.1 Login page

The project will start on the Login page, which will accept email and password. The system will verify the account and determine if it exists. If applicable, the user will be sent to the dashboard. Account verification occurs during the sign-in process, as discussed later in this chapter. If the account is not confirmed, the system will display the error shown in Fig. 4, and the user will be unable to access the dashboard. This strategy effectively prevents the creation of phoney accounts and credential theft.

A screenshot of a login page

Description automatically generatedA screenshot of a login page

Description automatically generated

Fig 3Prototype: Login Panel and Fig4. Prototype: Account verification error. Source: Author

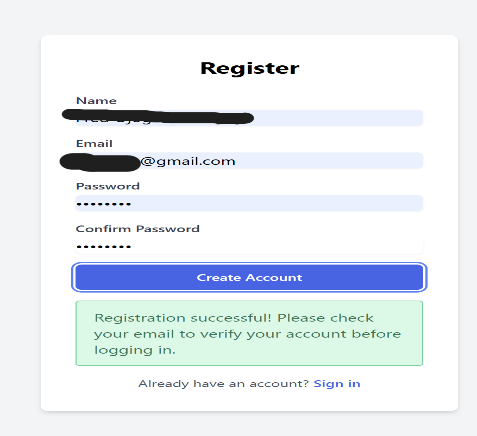
A computer code on a black background

Description automatically generatedFig 5 Login button functionality using JavaScript, Visual studio code. Source: Author

### 4.1.2 Register

To create an account, users can click the Sign Up link at the bottom of the Login Panel. To establish an account, enter your email address and a password of at least 6 characters (Uppercase, lowercase and special characters must be included). If the credentials meet the requirements, the system will send an email with a link to validate the account. Clicking the link activates the account and allows the user to login. This strategy prevents attackers from gaining direct access to an email account, making it a valuable security measure.

Currently, the recently created and verified accounts will be redirected to the dashboard. During the final submission, the newly registered user will be prompted to select their level of coding proficiency, to allow the recommendation system to accurately offer choices that match the users level.

A white background with blue text

Description automatically generated  
Fig 6 Prototype: Registration Page Fig 7 Account verification email. Source: Author

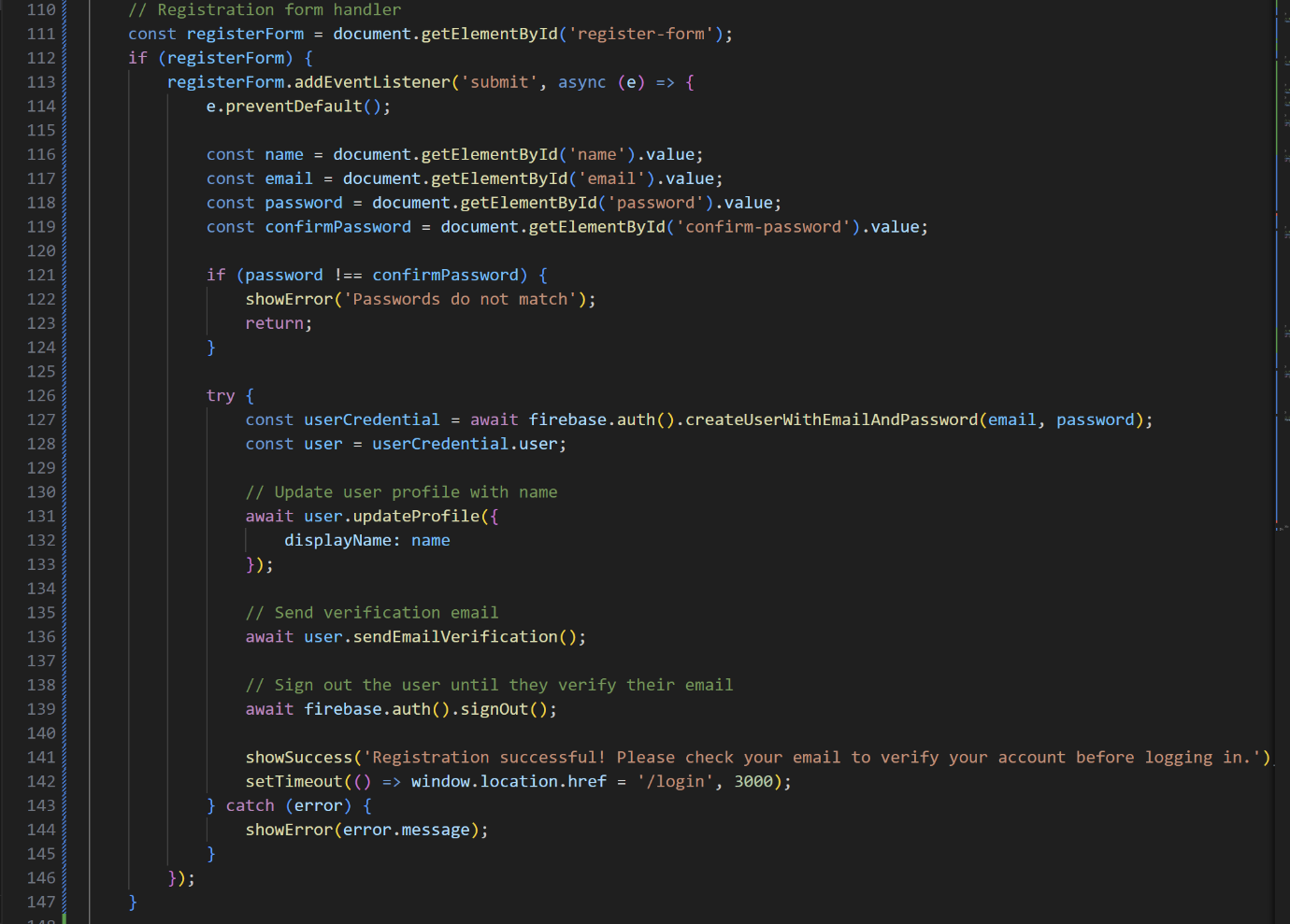


Fig 8 Registration button functionality using JavaScript, Visual studio. Source: Author

### 4.1.3 Forgotten password

A screenshot of a computer

Description automatically generatedA screenshot of a login page

Description automatically generatedIf a user forgets their password after creating an account, selecting the "Forgot Password" link in the Login Panel will redirect them to the Forgot Password panel. To reset your password, enter your previously used email address and receive an email with a link. To reset a password, navigate to the web page (Fig. 12), enter a new password, and click the "Save" button to submit it to Firebase. The user can now log in with their updated credentials.

Fig9 Prototype: Forgot Password Dialog Fig 10 Reset Password Email. Source: Author.

A screen shot of a computer screen

Description automatically generatedA screenshot of a computer

Description automatically generated  
Fig 11 and Fig12. Reset Password page process. Source:Author

A black screen with text on it

Description automatically generated

Fig13 Reset button functionality using JavaScript, Visual studio. Source: Author

### 4.1.4 Dashboard

As stated earlier, the dashboard development is still in progress during the course of this submission. The areas of focus during this submission were firstly to ensure the safe creation of accounts and signing in using firebase authentication services. Another aspect that was a point of focus in this submission was the integration of an IDE area and a recommendation area with ease of access for users.

A screenshot of a computer

Description automatically generated

Fig 14 Dashboard

### 4.1.5 Design diagrams

The following are the use case and class diagrams for the project prototype.

Admin actions

Admin

Manage users

Manage usage

Manage lessons

User action

Manage lessons

Logout

Track program

View Feedback

Request code review

Write code

View lessons

Login

Register account

Update Progress

Generative AI feedback

Save code

Verify Email

**USE CASE DIAGRAM**

|  |
| --- |
| Authentication |
|  |
| +verifyToken()  +generateToken()  +validateSession() |

|  |
| --- |
| Lesson |
| String id  +String title  +String description  +String difficulty  +String content |
| +getLessonContent()  +updateProgress() |

|  |
| --- |
| User |
| +String uid  +String email  +String displayName  +Boolean emailVerified |
| +register()  +login()  +logout()  +requestPasswordReset() |

|  |
| --- |
| CodeReview |
| +String userId  +String code  +String feedback  +DateTime timestamp |
| +requestReview()  +saveFeedback() |

|  |
| --- |
| Progress |
| +String userId  +String lessonId  +String status  +Number score |
| +updateProgress()  +getProgress() |

**CLASS DIAGRAM**

## 4.2 Methodology

During the first semester, I spent 11 weeks constructing an interim report and prototype. Proper time management is crucial for this assignment, which simulates real-world customer work. I explored several software options to schedule my work across 11 weeks, but none met my expectations. Despite understanding the significance of approach, I created a work plan and Figure 19.

Gantt chart. Methodology: Key Reasons I Based on My Work Plan

* Project work general organization
* Risk assessment and feasibility study
* Resource allocation
* Communication and collaboration with supervisor
* Project control
* Flexibility and adaptability to project changes
* Project improvements measurement

A screenshot of a computer

Description automatically generated

Fig 17

# **CHAPTER 5: TESTING**

As part of my training, I emphasise the importance of testing for ensuring system security. Unfortunately, I was unable to conduct thorough penetration testing on my system due to its simple prototype. Acceptance testing allows for security checks by attempting to inject harmful code. Firebase provides a highly secure authentication method. Especially with SHA-1 data encryption and no access to passwords. In addition to ensuring that all fields and buttons function properly, I also tested the application's resistance to malicious activity.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Panel | Test Input | Expected Result | Actual Result | Pass/Fail |
| 1 | Login | Input invalid email | Firebase: Error (auth/invalid-login-credentials). | Firebase: Error (auth/invalid-login-credentials). | p |
| 2 | Login | Input invalid password not matching DB | Firebase: Error (auth/invalid-login-credentials) | Firebase: Error (auth/invalid-login-credentials) | p |
| 3 | Login | Clicks Login Btn with correct credentials | Login Successful! | Login Successful! | p |
| 4 | Login | Clicks Login with empty Password Field | Please fill in this | Please fill in this | p |
| 5 | Login | Clicks Forgot Password link | The Forgot Password Dialog appears | The Forgot Password Dialog appears | p |
| 6 | Login | Clicks Sign Up link | The system takes the user to the SignUp page | The system takes the user to the SignUp page | p |
| 7 | Login | Enters correct credentials | The system log in the user to the Dashboard. | The system log in the user to the Dashboard. | p |
| 8 | Signup | Enters Password shorter than 6 characters | Firebase: Missing password requirements | Firebase: Missing password requirements | p |
| 9 | Signup | Clicks Sign Up Btn with empty Repeat Password field | Please fill in this field | Please fill in this field | p |
| 10 | Signup | Clicks Sign Up Btn with all credentials matching the requirements | The user’s credentials go to db waiting for email verification | The user’s credentials go to db waiting for email verification | p |
| 11 | Signup | Clicks the Login link | The system takes the user to the Login Page | The system takes the user to the Login Page | p |
| 12 | Reset Pass Dialog | Clicks Cancel Button | The dialog is dismissed | The dialog is dismissed | p |
| 13 | Reset Pass Dialog | clicks Reset Button with provided empty email | Please fill in this field | Please fill in this field | p |
| 14 | Logout | Clicks Log out Btn | The system logout the user and takes the student to the Login Page | The system logout the user and takes the student to the Login Page | p |

# **CHAPTER 6: CHALLENGES AND LIMITATIONS(LESPI)**

Despite the promising preliminary results, various problems and constraints were discovered when developing and testing the customised coding platform.

## **6.1 TECHNICAL CHALLENGES**

1. Cold Start Problem

The collaborative filtering approach struggled to identify tasks for new users due to a lack of interaction data. Without enough input, the algorithm relied on generic recommendations, lowering the level of customisation.

**Potential solution**: consider implementing a hybrid recommendation system that uses both content-based filtering (based on exercise information) and collaborative filtering.

1. Scalability Issues

While the system functioned admirably in small-scale testing, its capacity to manage huge datasets and multiple users remains unproven. The scaling problems included significant latency when accessing the recommendation engine and increased backend traffic.

**Potential solution**: Consider optimising database queries and deploying on scalable cloud architecture (e.g., AWS Lambda) for increased availability.

1. Model Overfitting

During training, the collaborative filtering model may overfit to specific user habits, producing biassed recommendations.

**Potential solution**: Use regularisation and cross-validation to improve model performance across different user groups.

## **Data Limitations**

1. Dataset Diversity

The dataset, while big, was mostly concerned with beginner-level coding challenges. This restricted the platform's capacity to provide advanced learners with demanding content.  
  
**Potential solution**: Consider expanding the dataset to include challenges from competing programming platforms and open-source repositories, such as GitHub.

1. Feedback Quality

There were discrepancies in the user feedback dataset, such as incomplete items or contradicting opinions. This impacted the sentiment analysis model's accuracy.

**Potential solution**: clean the dataset using pre-processing techniques and manually annotate confusing feedback entries.

## **6.3 USER ADOPTION CHALLENGES**

1. The Initial Learning Curve

Some users found the platform's features daunting, especially those unfamiliar with adaptive systems.

**Potential solution**: create onboarding tutorials and an easy-to-use interface to decrease learning curves.

1. Lack of Offline Support

The program currently requires an active internet connection, which limits accessibility for users in places with low connectivity.

**Potential solution**: Consider adding offline capabilities for critical elements, such as saving exercises and progress locally.

## **6.4 ETHICAL AND PRIVACY CONCERNS**

1. Data Privacy.

The collection and storage of user data raises worries about privacy and security. Ensuring compliance with legislation like GDPR is critical.

**Potential solution**: consider end-to-end encryption and detailed terms of service describing usage policies.

1. Bias in algorithms

The recommendation algorithms risk repeating biases seen in the training data, perhaps disadvantageous to specific user groups.

**Potential solution**: consider regularly auditing models for bias and ensuring diversity in training datasets.

# **CHAPTER 7: FUTURE WORK AND CONCLUSION**

In the final submission, I will continue with the aforementioned project idea as shown in the first and third chapters of this report. Currently, I have developed fully functional log in, register, dashboard and forgotten password pages, which require user verification for account activation. Allowing users to log out and requiring user authentication each time.

In the second semester, I am going to complete the integration of the recommendation system, as well as include a system to auto-generate explanations for the coding problems.

During the last three months of developing the Final Year Project, I learnt the need of self-discipline and meticulous preparation, especially given the restricted time and workload of many classes. Weekly meetings with my supervisor helped me stay committed to the project and provided insight into client-customer collaboration in real life. With my expertise, I am confident that the upcoming semester's submissions will be less stressful and more efficient.

# REFERENCES

1. Brown, A., & Jones, T. (2021). *Adaptive learning systems in education: A review of methods and applications*. Journal of Educational Technology, 12(4), 45-67.
2. Gupta, R., et al. (2019). *Impact of reinforcement learning in personalised education*. Proceedings of the AI in Education Conference, 234-245.
3. Smith, P., et al. (2020). *User experiences on coding platforms: Insights and future directions*. Journal of Computer Science Education, 18(3), 112-125.
4. Anderson, T., & Shattuck, J. (2012). *Design-based research: A decade of progress in education research?* Educational Researcher, 41(1), 16-25.
5. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.
6. Koren, Y., Bell, R., & Volinsky, C. (2009). *Matrix factorization techniques for recommender systems*. IEEE Computer, 42(8), 30-37.
7. Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT Press.
8. Codecademy - <https://www.codecademy.com>
9. LeetCode - <https://leetcode.com>
10. freeCodeCamp - <https://www.freecodecamp.org>