E-Learning Platform

By

Ayush Katoch

A DISSERTATION

Submitted to

The University of Liverpool

in partial fulfilment of the requirements

for the degree of

MASTER OF SCIENCE

21st September 2023

ABSTRACT

E-Learning Platform

By

Ayush Katoch

*Teaching programming to Computer Science and Non-Computer Science students have become a very common practice in universities. Programming is being used in all kinds of fields from Business Analytics to Bioinformatics. Student from non-technical field often face a lot of issues when they are given a programming assignment like being unable to setup a development environment and similarly professors have to go through all the submissions and execute the code one at a time. This website has been developed to fix / improve the teaching and learning process for both the students and the teachers by eliminating the tiring and time-consuming processes.*

*To make this website possible, multiple research papers were read to decide on the features that should be implemented into this website, these features focus on making teaching and learning programming easier for both the parties. Some of those features being Live Code Share, Course Cloning, Plagiarism Detector, Code Executor, Domain Name Filtering and many more.*

*At the end, this project has been a success and all the mentioned features along with the aims have been achieved.*

DECLARATION

I hereby certify that this dissertation constitutes my own product, that where the language of others is set forth, quotation marks so indicate, and that appropriate credit is given where I have used the language, ideas, expressions or writings of another.

I confirm that I have not copied material from another source nor committed plagiarism nor commissioned all or part of the work (including unacceptable proof-reading) nor fabricated, falsified or embellished data when completing the attached piece of work.

I declare that the dissertation describes original work that has not previously been presented for the award of any other degree of any institution.

Signed,



Ayush Katoch

ACKNOWLEDGEMENTS

Firstly, I would like to thank my parents for their continuous support, love and encouragement. This project wouldn’t be possible without their support.

I would also like to thank my supervisor Mr. Henry Forbes. Who gave me this opportunity to work on this project. He has been there to support me from day one. Always going out of his way to offer his support whenever I require. His insightful feedback and mentorship have been invaluable in shaping the course of this project.

I am also grateful to the University of Liverpool, specifically the Computer Science Department for giving me this opportunity to study Master’s in Advanced Computer Science. Also, for providing the essential resources and fostering an environment conducive to my dissertation project.

The university's exceptional faculty, resources, and academic community have significantly contributed to my academic journey. I appreciate the opportunities and support that have been pivotal in shaping this dissertation.

*\*

**TABLE OF CONTENT**

Page

LIST OF TABLES vi

LIST OF FIGURES vii

Chapter 1. Introduction 2

Chapter 2. Aims & Objectives 3

Chapter 3. Background 6

Chapter 4. Ethical use of data 7

Chapter 5. Design 8

Chapter 6. Implementation 16

Chapter 7. Evaluation 21

Chapter 8. Learning Points 23

Chapter 9. Professional Issues 25

Chapter 10. Conclusion 26

Chapter 11. Bibliography 26

Chapter 12. Appendix 28

LIST OF TABLES

Page

Table 1: List of database tables 13

Table 2: List of new tables added to the database 18

Table 3: Project Log 33

LIST OF FIGURES

Page

Figure 1 Code Executor flow diagram 10

Figure 2: Database ER Diagram during design phase 14

Figure 3: Frontend UI during design phase 15

Figure 4: Website's Sitemap Diagram 15

Figure 5: Use-case diagram for different types of users 16

Figure 6: New Database ER Diagram 18

Figure 7: Snippet of generating forms 19

Figure 8: New UI Changes 20

Figure 9: Unit testing in Postman 21

Figure 10: Code for a single Unit Test 21

Figure 11: Creating a DB from PgAdmin 29

Figure 12: Settings.py to enable / disable a database 30

***Intentionally left blank (as per the provided template)***

# Introduction

Computer Science is one of the fastest-growing fields in the industry, generating massive number of jobs along with appealing compensation and attracting large number of students to pursue Computer Science for their Bachelors / Master’s degree. Nowadays, it’s becoming very common for the universities to teach programming, not only to computer science students but to non-computer science students as well. Thus, it has become very important for a software / tool to exist that can help in teaching & learning programming for both the professors and the students respectively.

Old ways of teaching programming involved, large number of class lectures and a few programming assignments. This outdated way of teaching programming is no longer a feasible option. Reasons for the few programming assignments is related to the process of submitting and grading of the assignments. Earlier, for a student to complete their assignments, they had to setup a development environment so that they can start coding. After writing their code, they had to submit their files to the professors. Professors would then download, execute and grade each submission one at a time. Which is a very tedious process and also results in a lot of failures since different students might have been using a different version of a programming language / library.

This project was proposed with the aim to make teaching & learning programming simple for professors and the students respectively. It achieves these objectives by eliminating the need to setup a development environment, granting access to lecture notes and recording in one place, promoting group work through group assignments, avoids issues related to different software / library versions being used, helps in the grading of the assignments by providing features like plagiarism detection and automatic code grading based on predefined criteria setup by the professor when creating the assignment, prevents unauthorized access to a course content by a third party through Domain Name Filtering, makes transitioning from one term to another easier through Course Cloning.

# Aims & Objectives

## Aims

The main aim of this project is to make teaching & learning programming easier for both the professors and the students by understanding the problems faced by the targeted users and attempting to eliminate those problems by introducing new and efficient ways of doing things.

Below is the list of Aims’ that have been achieved.

### Students

* This software provides them a platform where they can write and test their code without having to setup a development environment.
* Provides real time feedback on the submitted code.
* Promotes teamwork through group assignments.
* Access to uploaded video lectures and notes

### Teachers

* Easy way of monitoring class performance through graphs.
* Automatic grading of programming assignments which eliminates the exhausting old ways of grading assignments.
* Plagiarism checks on the submitted assignments.

## Objectives

In order to achieve the above-mentioned Aims, the following objectives had to be accomplished.

### Account Management System:

* Allows creation and management of student and teacher accounts.

### Course Management System:

* Allows creation and management of courses created by the professors.
* Allows students to join a course

### Module System

* Allows uploading of files like lecture notes, ppts and video lectures by the professors
* Allows students to download uploaded files.

### Assignment System

* Allows professors to upload an assignment which can be of report type, programming type or hybrid.
* Students can upload their assignments

### Grading System

* Allows professors to grade student assignments using a predefined rubric cube for marking.
* Students will be able to see their grades

### Plagiarism Checker

* Checks for plagiarism in the submitted reports and the code

### Code Executor

* Runs user code in the backend, maintains the virtual environment and the installed libraries.

Below is the list of objectives that were decided later on in the project

### Course Management System

* Professors can also provide a specific domain name that will allow only those students tojoin the course who have the domain mentioned in their email address.
* Allows teachers to clone their courses which also copies all the lectures and assignments.

### Assignment System

* Professors can now also create group assignments enabling group work.

### Live Code Share

* Allows users to share code with each other when working in a group assignment which will make collaboration very convenient.

# Background

In order to understand the features that should be implemented, research on Computing Pedagogy was read. Pedagogy is a way of teaching a subject. Computing Pedagogy means effective ways to teach computing to people. It involves 12 NCCE’s principles, Lead with concepts, work together, get hands-on, unplug unpack and repack, model everything, foster program comprehension, create projects, add variety, challenge misconceptions, make concrete, structure lessons lastly, read & explore code first *(Teach Computing. (n.d.). Pedagogy)*. This software follows multiple principles of Computing Pedagogy,

* **Lead with concepts:** “graphical tools for organising and representing knowledge” *(Florida Institute for Human and Machine Cognition, 2006).* Concepts maps are used to represent a knowledge visually. It consists of an interconnected network of concepts. Links are used between the concepts to specify the relationships between them. This site will support uploading video lectures and notes that can be used by the students to better understand the concept.
* **Work Together: “**Peer Instruction (PI) leads to significant learning gains for learners: those engaged with PI made up to twice as much progress as other learners.” (Porter, 2016). Pair programming is an approach that can be used in a classroom. It involves students working together on a problem while sharing a computer. Students can work together on a programming assignment using this software through the feature like Group Assignments.
* **Get hands-on:** “There is some evidence that physical computing activities can support a learner’s program comprehension” (Jayathirtha, 2021). Physical computing is a term to describe activities where learners write programs to interact with the real world using specialized hardware. This software could be the best way to practice programming, as it allows students to start coding without having to install / setup any third-party software.
* **Unplug, unpack, repack:** “‘Semantic waves’ describe an ideal conceptual journey for novice learners to follow, shifting between expert and novice understanding, abstract and concrete context, and technical and simple meanings” (Maton, 2013). Programming is a subject that has lots of technical terms having a precise technical meaning. In order for students to understand and master these terminologies is to follow semantic wave (Making semantic waves, 2018). This involves using metaphors, analogies to teach.
* **Model Everything:** When learning students are exposed to the completed program and are not shown the troubleshooting that took place while writing the program. Live coding is when a teacher develops the solution to a problem in front of the class to allow learners to follow.

To Learn ReactJS. ReactJS official documentation was used (react.dev. n.d.)

A research paper by I. and Sumitra *(Indriyanto, I. and Sumitra, I.D, 2019)* was read to understand the working of plagiarism detection.This paper talked about using TF-IDF and Cosine Similarity to detect plagiarism.

Finally, to learn about the multiprocessing, following article was referred (*Multiprocessing in Python, 2021).* Which talked about importance of multiprocessing and how to implement it.

# Ethical use of data

All the data that have been utilized in the website is a dummy data and no real data has been used. Purpose of using mock data was to test the working of the website and to show the working during presentations. Personal information that is being acquired is done through forms and are as follow:

* Email Address: For registration and login
* First & Last Name: For personalized experience and identification
* Profile Picture: For personalization and can be deleted by the user
* Courses: Courses that a user is part of.
* Grades: Grades acquired by a student for visualization.

In order to protect the users and their data, All the confidential information is encrypted and hashed, such as passwords, prior to storage. This ensures that even in the event of a database breach, unauthorized access to user data, including passwords, remains unattainable.

During the account registration process, users will be informed that once an account is created, it cannot be deleted. This is done to maintain the stability and reliability of the database. Given the intricate network of references among tables, with most being linked to the Profile table, the deletion of a user account could compromise the database's integrity and result in inaccuracies in the information presented to the users.

# Design

Website development will be divided into two separate parts. Frontend and the backend. Backend will utilize Python (Django framework) and PostgreSQL whereas the frontend will be using JavaScript (ReactJS) and CSS (Tailwind CSS).

## Backend

In order to develop the RestAPI, Django will be used. Django is one of the Python frameworks for creating web servers. It has its own Object Relational Manager (ORM) that makes performing actions like creating / modifying / querying the database much simpler.

SQLite is a Database Management System (DBMS). It’s small and a reliable database good for small application. Other DBMS were also considered like PostgreSQL which is definitely better than SQLite but considering the scope of this project, for the time being SQLite was chosen.

Backend development will involve creating the database and coding the RestAPI to handle various HTTP requests like creating / updating / modifying / deleting resources example, Courses. Backend consisted of three major key components Code Executor, Plagiarism Detector and Live Code Share (proposed after the initial proposed design report. More about live code share in the section 6.1).

### Code Executor

This module will take user's code as an input and executes it in the backend, providing appropriate error and success messages. It will also be responsible for grading the programming assessment and granting marks to the student based on the code’s performance like code compiles, code runs without error, code passes all the test cases.

Major drawback of the Code Runner module is that it’s blocking. When a user tries to execute his / her code, it gets executed on the first come first serve basis (similar to Queue data structure). Meaning, when one user’s code is being executed other users will have to wait for that execution to end before their code starts to get executed. This problem can be fixed by implementing Django Consumers (WebSocket) and Multi-processing. WebSocket enables two-way communication between the server and the client. This way, even when the code takes a long-time user will receive their code output immediately as soon as it becomes available without having to worry about request timeout issues. Multi-processing will be used for running long operations like executing user code. This way multiple users can run their code simultaneously. One issue with this approach is that it will have a significant impact on the performance of the server specially when multiple users are running their code.

Code Executor will operate by taking the user’s code executing it in the server using subprocess library. It first compiles the user code and depending on the outcome it either awards scores to the users (if they earned any) or stops the execution while returning the error message. Similarly, code is then executed and tested against predefined test cases. based on the outcome, it either grants scores to the user and returns a success message or returns an error message.

Below is the diagram for the Code Runner algorithm.

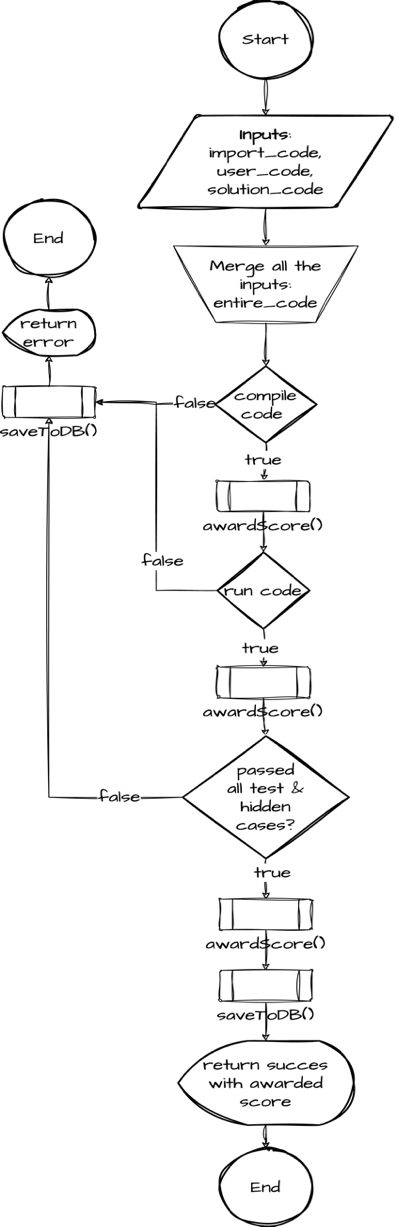
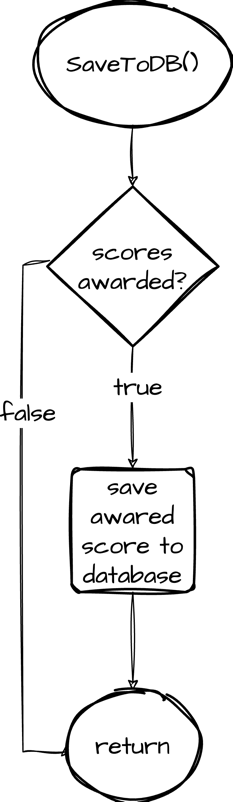


Figure 1 Code Executor flow diagram

### Plagiarism Detector

To implement plagiarism detector, three methods were considered

* Using API
* Vector Space Model (VMS) and Cosine Similarity
* Machine Learning

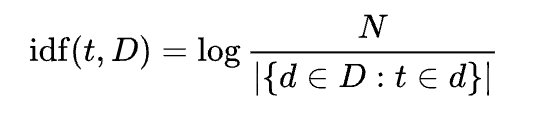
After carefully considering the limitations and advantages of all the three methods, Vector Space Model (VMS) and Cosine Similarity was selected as an algorithm to implement the plagiarism detector. API wasn’t selected because all of them were paid. Machine learning worked similarly to the VMS and Cosine Similarity method but required a large dataset with relevant articles in order to find plagiarism. There wasn’t any dataset available that met the project’s requirement and scrapping data from the required website would have taken a huge amount of time. Because of these limitations, VMS and Cosine Similarity was picked.

Vector Space Model (VMS) and Cosine Similarity compares the strings between the two documents and returns a score between 0 to 1 where higher number signifies how similar the documents are. In order to detect plagiarism, submissions by the students are converted to vectors using TF-IDF (Term Frequency – Inverse Document Frequency) algorithm. These vectors are supplied to the Cosine Similarity formula that returns the similarity scores between the documents. Advantage of this method is that it’s completely free and also doesn’t rely on a dataset to function. Disadvantage of this method is that it only looks for plagiarism between the submissions and doesn’t check the online websites for the plagiarism.

TF-IDF consist of two parts, calculating the TF and the IDF. TF is calculated using the following formula

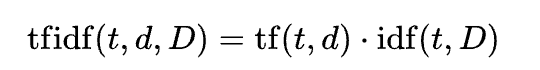
Where *ft*,*d* is the number of times that term *t* occurs in document *d*. denominator is simply the total number of terms in document *d*

IDF is calculated using

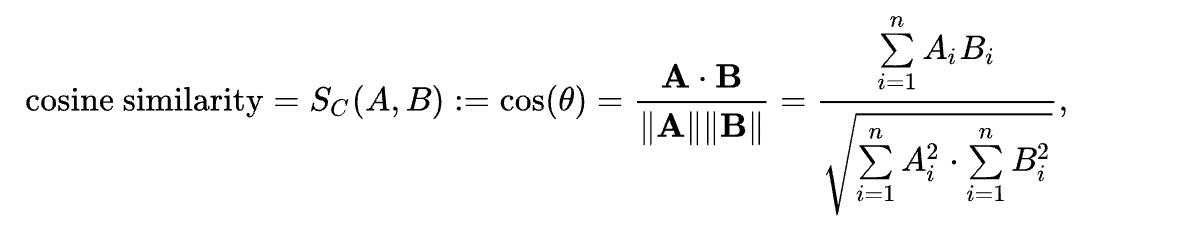


Where *N* is total number of documents in the corpus N = *|D|* and denominator is the number of documents where the term t appears.

And finally, then we can calculate the TF-IDF using



After calculating the TF-IDF, cosine similarity formula is used to find the similarity between two vectors using following formula.



Where, n is the dimensions of the vectors. A­­­i and Bi are the ith components of the vector A and B.

An example of how this algorithm works in the website, say we have 30 submissions for an assignment. All the 30 submissions get converted to vectors using TF-IDF algorithm. Out of these submissions, one submission is selected and its vectors are then compared with the other remaining submissions using Cosine similarity algorithm. Cosine similarity returns the similarity score between 0 and 1. This is repeated for all the submissions.

### Database

The database consists of 10 tables.

|  |  |
| --- | --- |
| **Table Name** | **Description** |
| **User** | Holds information like first & last name, email, username and password |
| **Profile** | Holds users’ personal information like profile picture, user type (teacher/student), a reference to the user table and date & time of account creation. |
| **Course** | Used to store the details of the students who are in any course. |
| **CourseDetail** | Used to store the details of the students who are in any course. |
| **ModuleGroup** | Stores information about different groups for organizing modules like week 1, week 2 and so on. Each group belongs to a Course |
| **Assignment** | Stores information about an assignment (report portion of the assignment). It has a reference to AssignmentCode that stores the ‘programming’ portion of the assignment and to Course which indicate the course this assignment belongs to. |
| **AssignmentCode** | Used for storing programming part of an assignment |
| **AssignmentSubmission** | Whenever an assignment is submitted, the details are stored into this table. It has a reference to Assignment table and Profile Table (user who uploaded the assignment) |
| **AssignmentRemark** | Remarks submitted by the professor is stored in this table. It has a reference to AssignmentSubmission and to Profile table (user who graded the assignment). |

Table 1: List of database tables

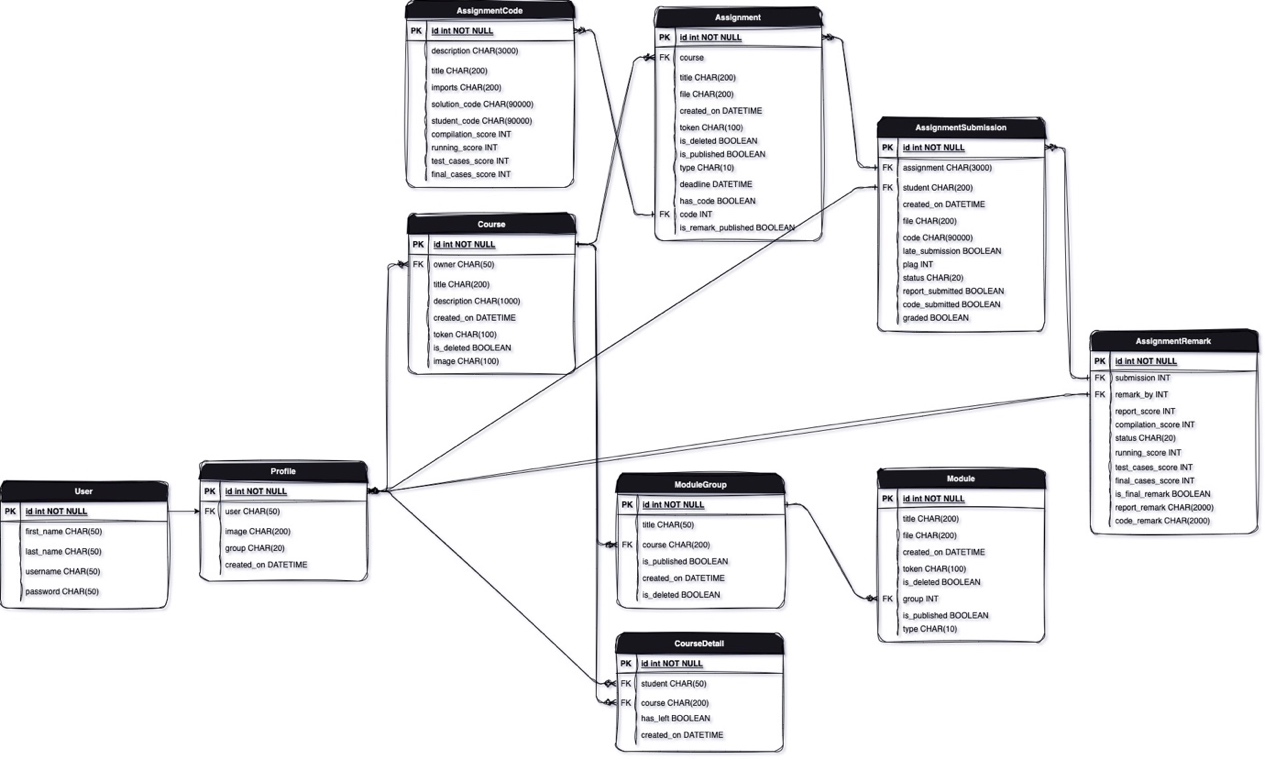


Figure 2: Database ER Diagram during design phase

## Frontend

To implement the frontend, ReactJS, one of the most used JavaScript libraries for creating web apps was utilized. As per the Stack Overflow survey of 2023. Out of 71,802 responses 40.58% respondent preferred / selected ReactJS over another frontend framework like Angular.js or Vue.js (Stack overflow, n.d). It was developed by Facebook and is widely used in websites like Twitter, Instagram and Facebook. One of the main features of ReactJS is reusability which is achieved through Components.

TailwindCSS is being used for styling and to make the website responsive. Alternatively, Bootstrap could have been used but unlike bootstrap, Tailwind CSS default components have a much more professional and modern appearance.

Canvas and CodeWars will be used as references to design the frontend. Below are the screenshots of the UI during the design phase.

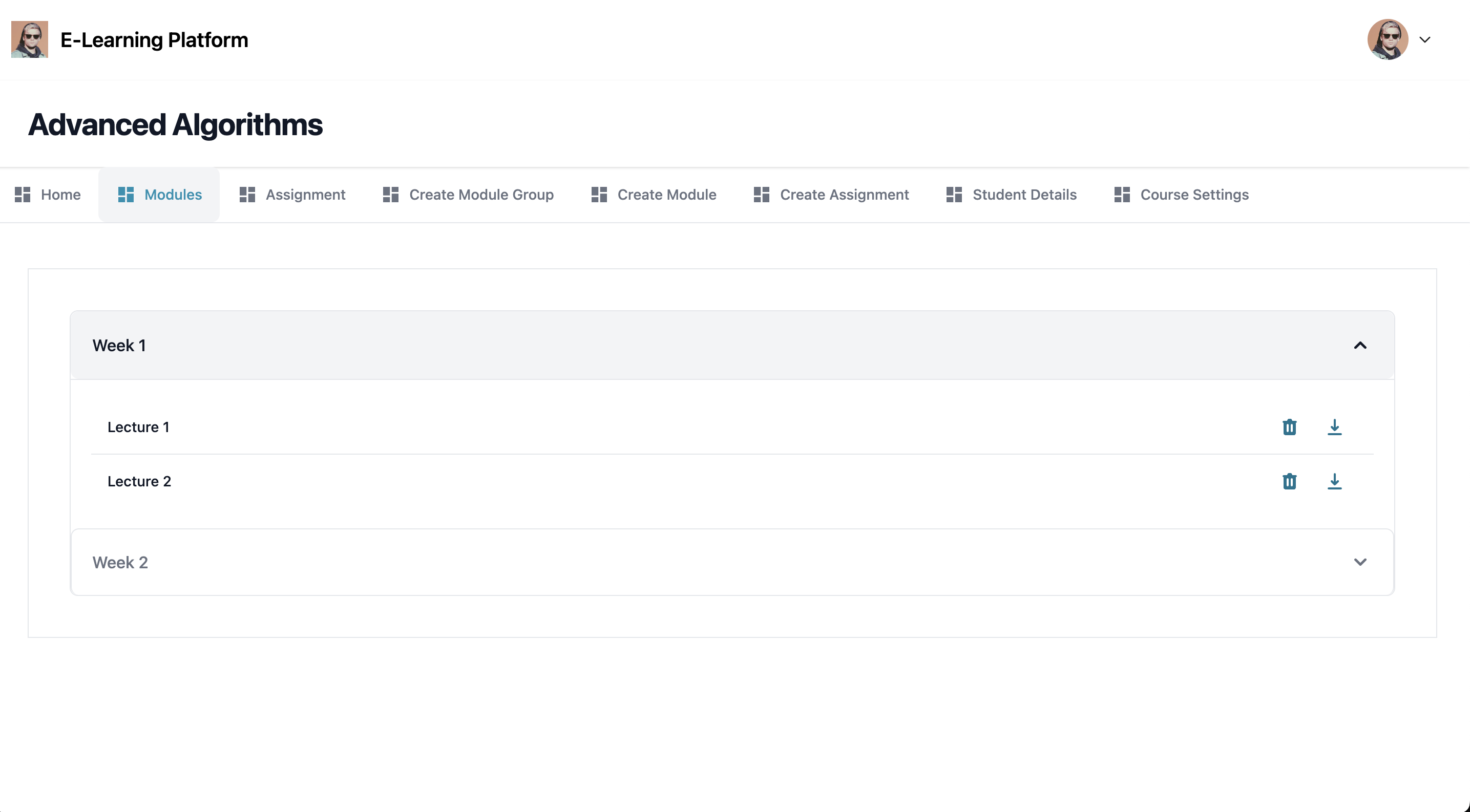
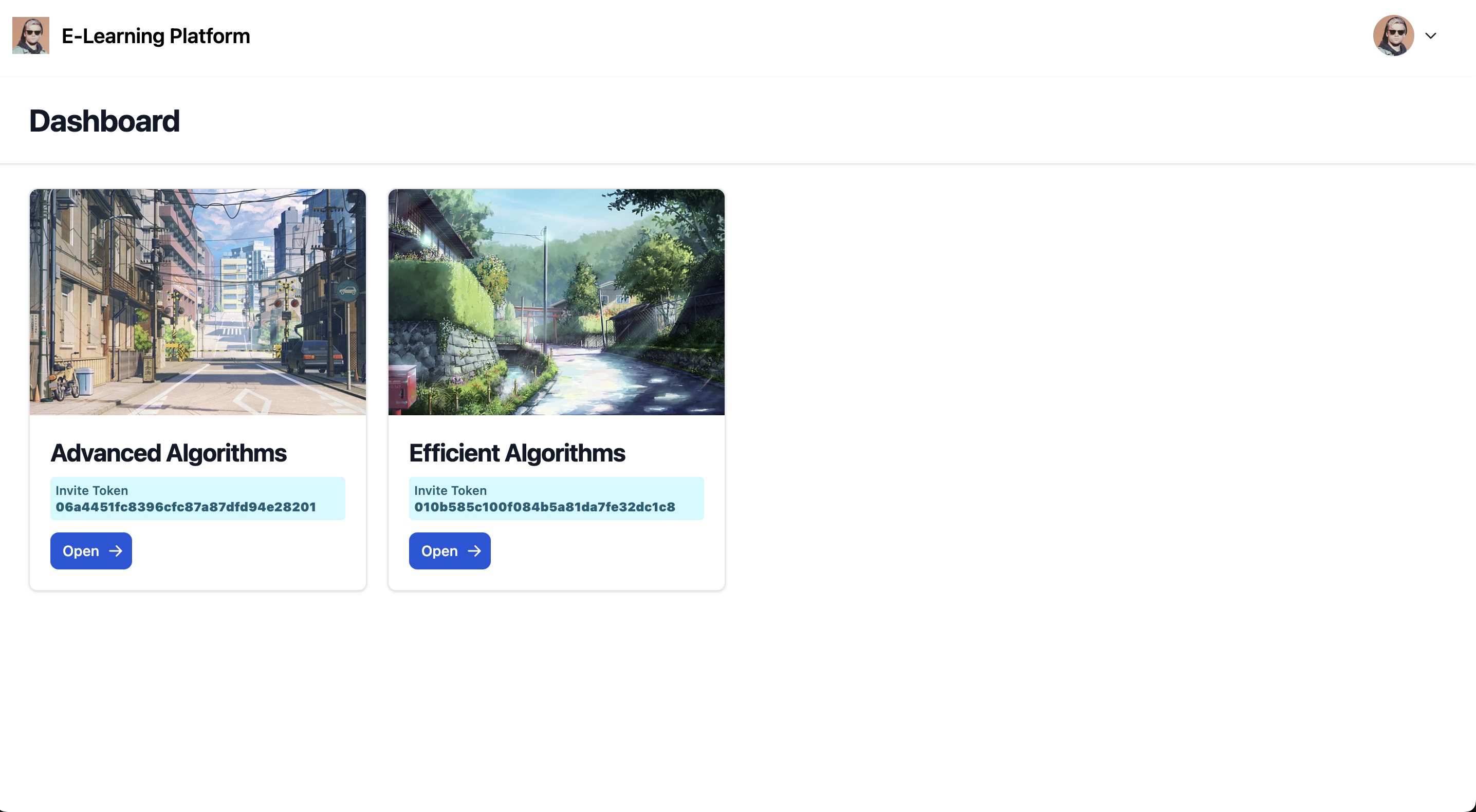


Figure 3: Frontend UI during design phase

Website’s sitemap Diagram

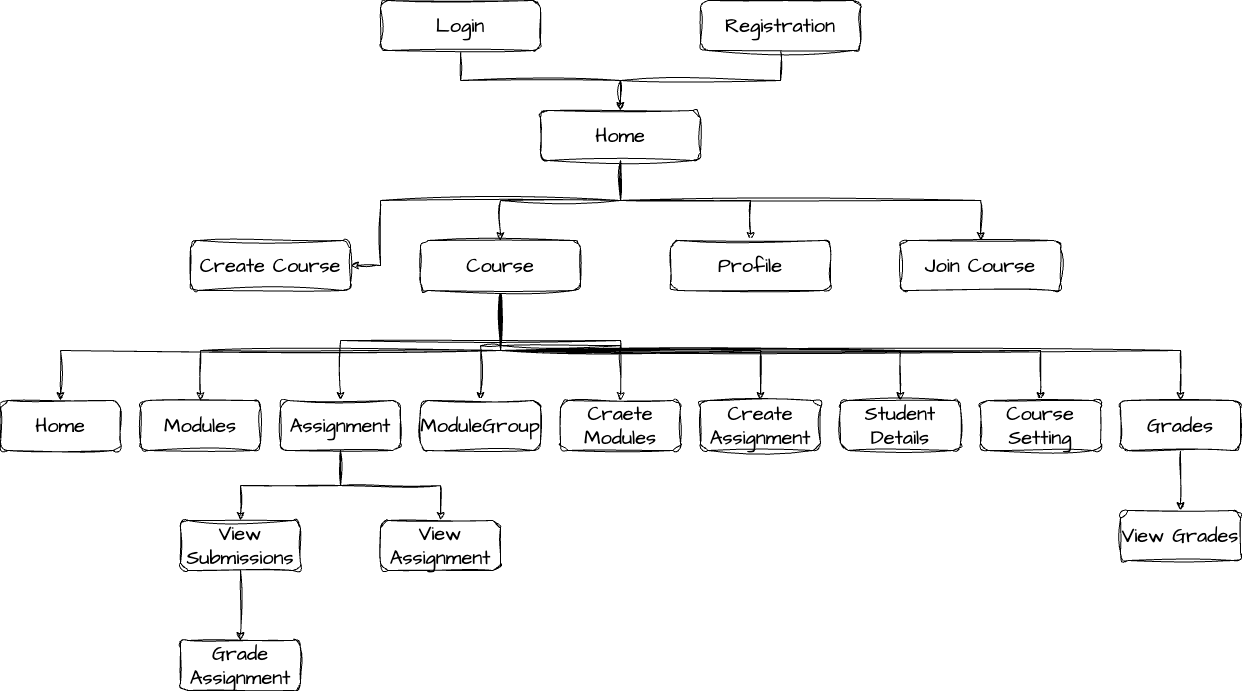


Figure 4: Website's Sitemap Diagram

Use case diagram of the software based on different roles

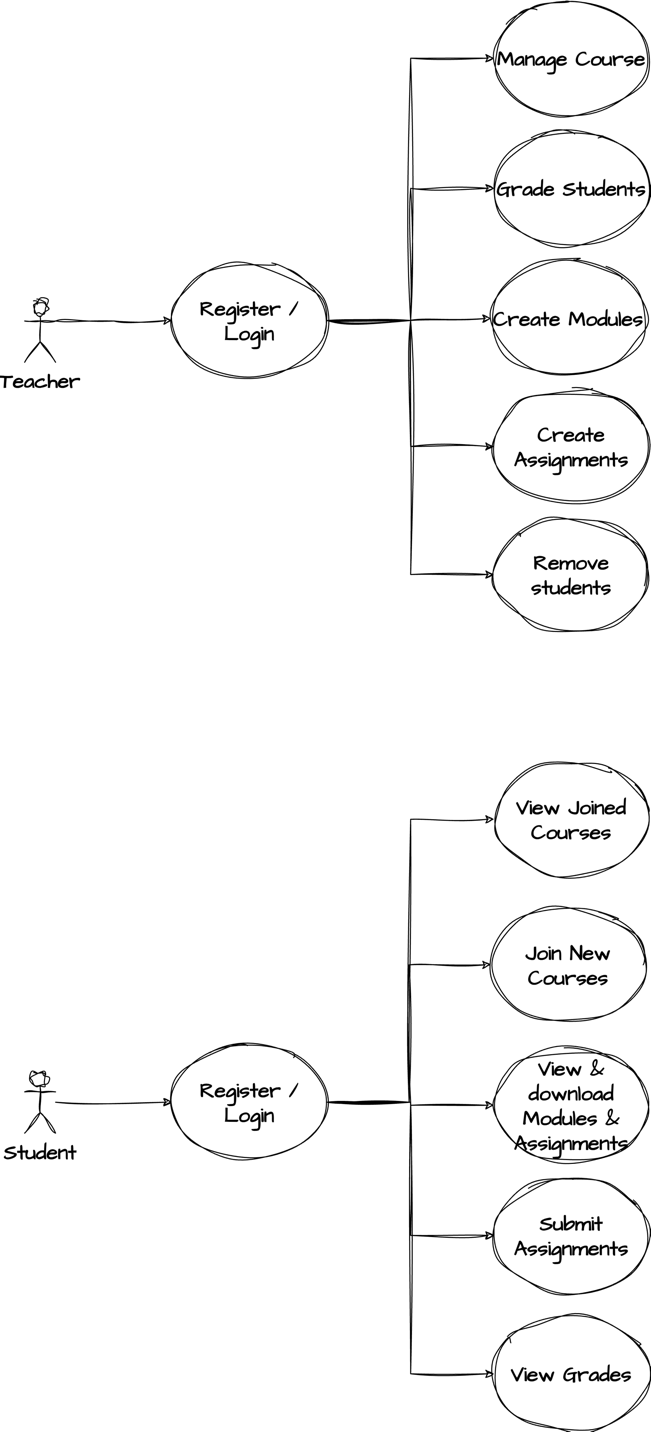
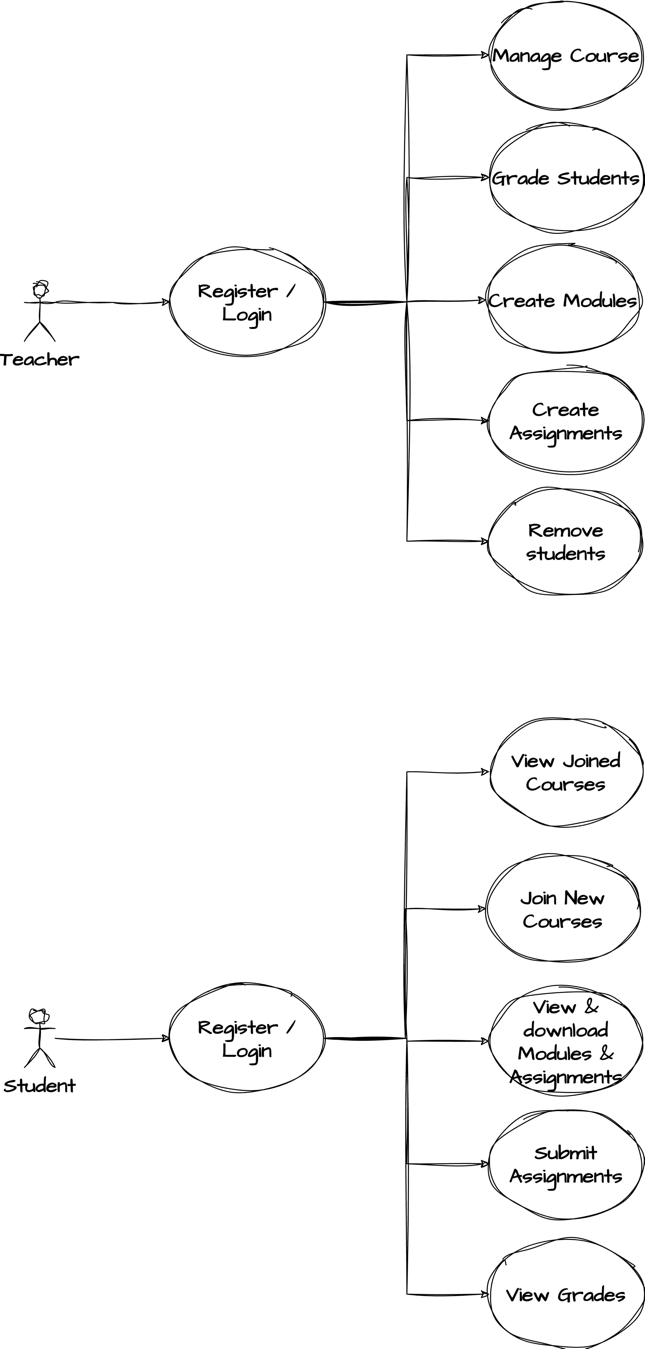


Figure 5: Use-case diagram for different types of users

# Implementation

Website development was divided into two separate parts. Frontend and the backend. Backend utilized Python (Django framework) and PostgreSQL whereas the frontend used JavaScript (ReactJS) and CSS (Tailwind CSS).

## Backend

When developing the backend, Django framework called Django-Rest-Framework was discovered. This framework makes it an easy task to create Rest-API along with setting up the user Registration and Login architecture. It also privdes wide variety of logging in options like token, sessions and much more.

While developing the website, a major issue was found when using SQLite. Since SQLite is a single file-based database, it often throws ‘database locked’ error when multiple users try to read and write from the database. Due to this, SQLite was replaced with PostgreSQL. It is a Relational Database Management System (RDBMS). It’s highly scalable and fast making it one of the best options for both small-scale and large-scale projects. Compared to other databases like SQLite, PostgreSQL can handle huge number of requests and data without ever slowing down.

After deciding with the supervisor, new features were decided to be added to the project. This resulted in changes to be made to the structure of the database. Some fields were either added or removed from the existing tables and a new table called AssignmentGroup was added to support group assignment feature.

Some of the new features that were added are

* **Group Assignments:** Allows professors to create group assignments where multiple students can work together on a coding assignment.
* **Live Code Share:** Live Code Share is used during group assignments, allowing group members to collaborate with each other. Live Code Share employed HTTP long polling to achieve this task. Changes made to a code, are sent to the backend and is saved in the database. For retrieval, frontend sends a HTTP request to the backend every five seconds. If backend detects any changes in the code, those changes are merged with the current code allowing group members to continue working with each other without having to worry about losing their progress.
* **Domain Name Filtering:** It prevents users from joining a course if their email address domain name is different from the required domain name, even if they have the course invite token. Instead, of rejecting their request, a new request is sent to a professor that can either be accepted or declined.
* **Course Cloning:** Professors can clone their existing courses that copies all the modules and the assignments that were uploaded by the professor.

### Database Changes

Below is a list of new tables that was added along with the updated ER Diagram of the database.

|  |  |
| --- | --- |
| **Table Name** | **Description** |
| **AssignmentGroup** | Holds information about the group members for an assignment. |

Table 2: List of new tables added to the database

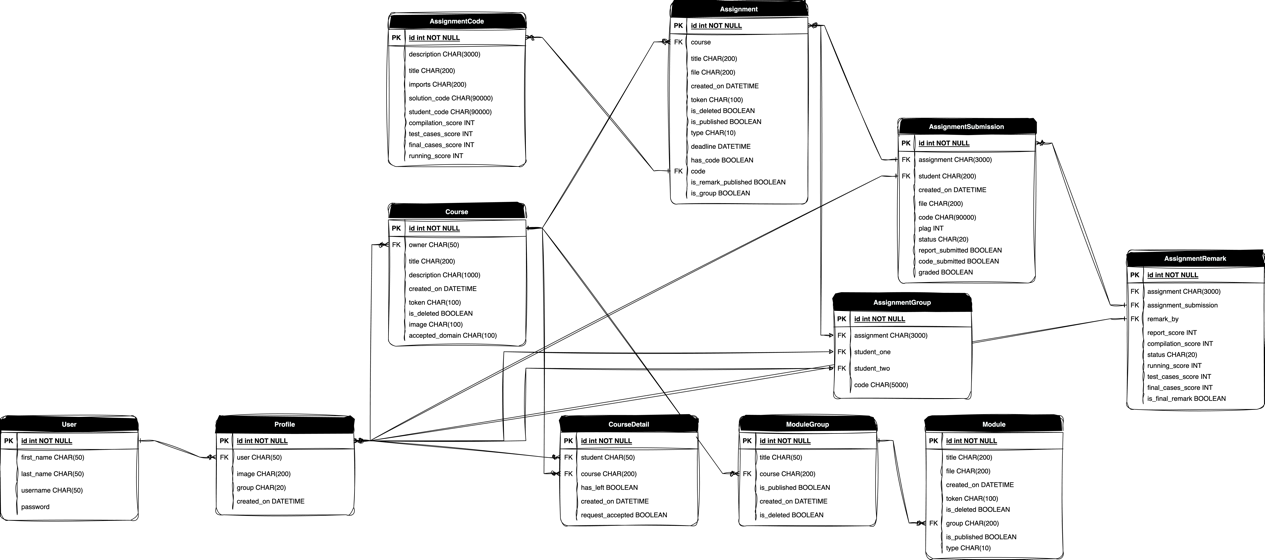


Figure 6: New Database ER Diagram

## Frontend

Frontend development started with the designing of registration and login page, frontend elements that were detected / identified as being repeating were turned into a component to make them reusable throughout the website. This project consists of large number of forms from login and registration to forms for creating / modifying / deleting courses, assignments, modules, grading and much more. In order to make generating forms easier, FormGenerator component was designed.

### FormGenerator

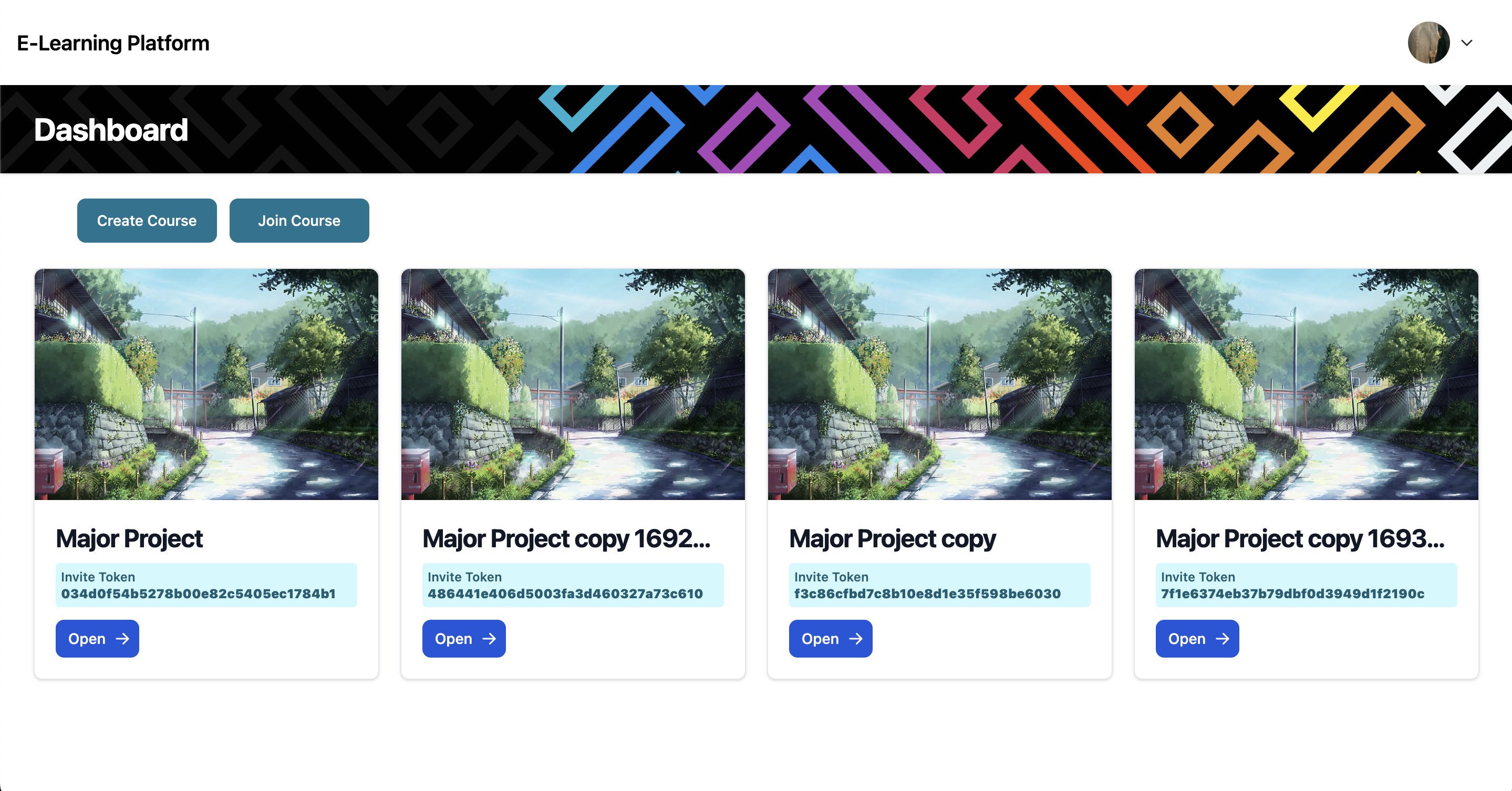
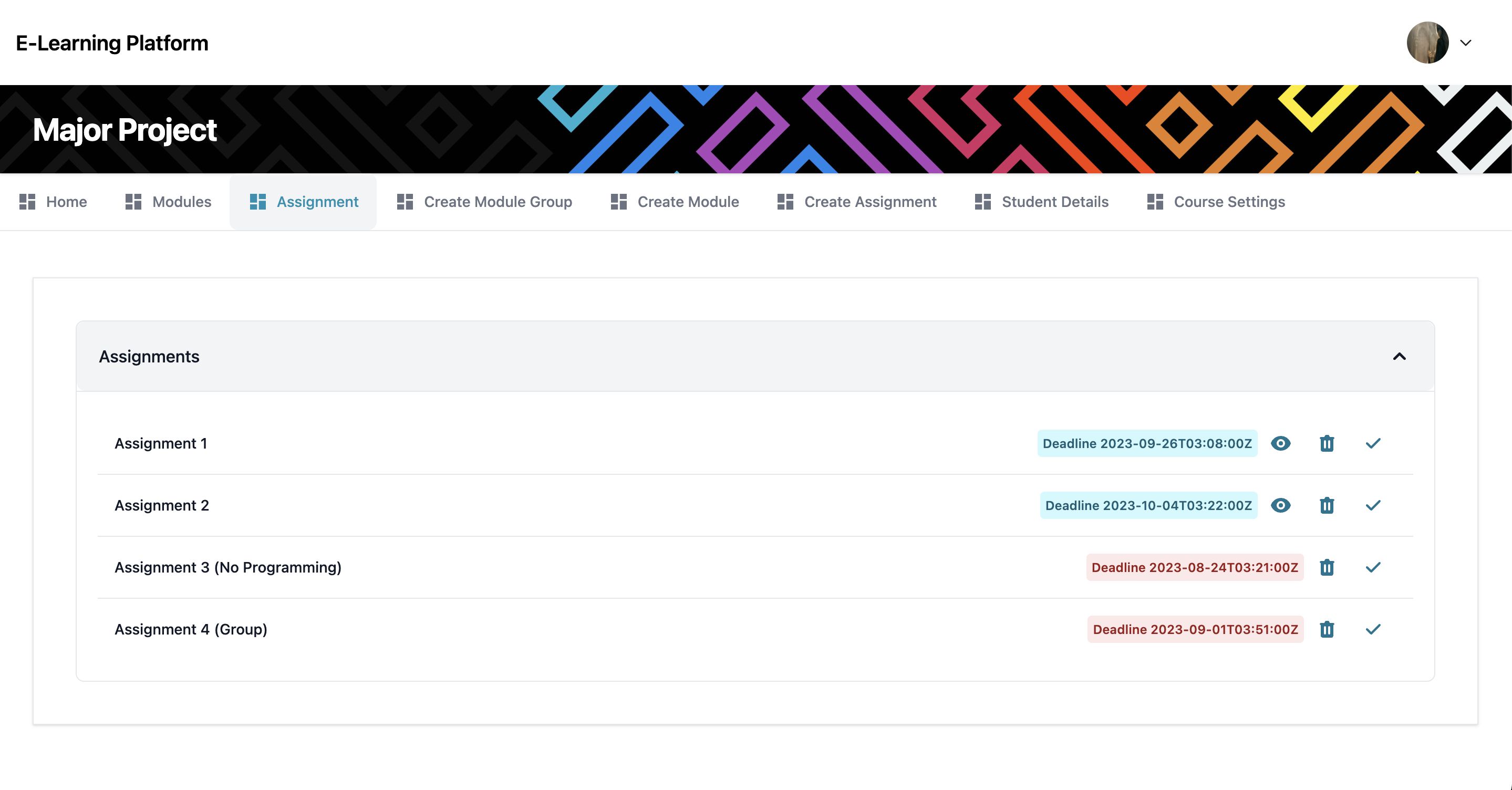
FormGenerator accepts a list of objects. Each object in the list defines an input type like email, text, number, radio box etc. FormGenerator then loops through this list and render all the input type in the webpage. This component significantly made the designing of forms simple and also dramatically helped in reducing the number of lines of code that had to be written.



Figure 7: Snippet of generating forms

### User Interface

Except some beautification changes, not much have been changed in the UI shown in the design phase.

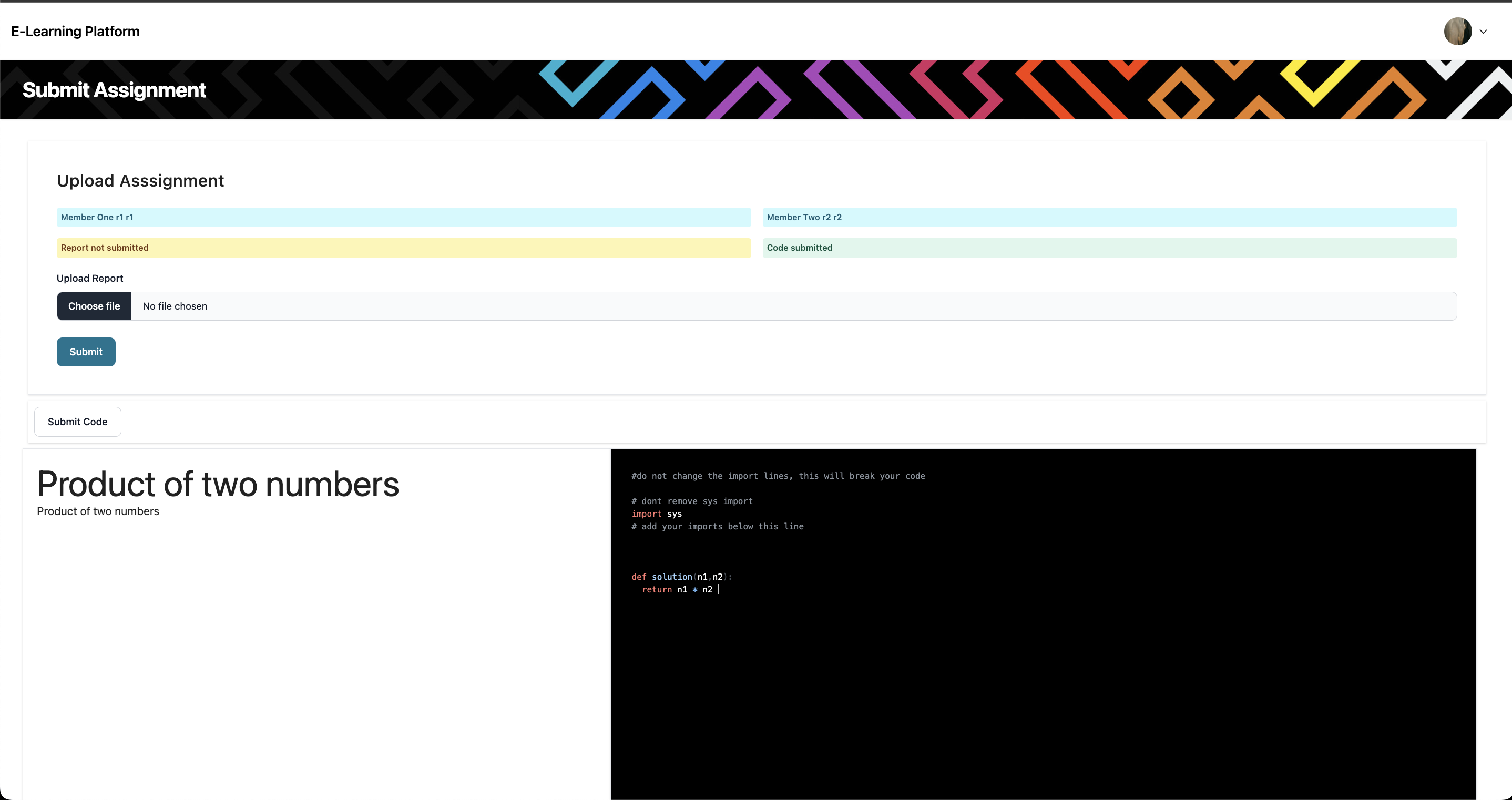


Figure 8: New UI Changes

## Testing

Unit Testing was performed using Postman. Since all the modules in this software are pretty small it makes it easy to implement Unit Testing. Friends were also asked to test the software.

To test a module, through Postman different inputs were supplied to ensure that it can handle all types inputs.

After performing Unit Testing, a friend was asked to use the software. Without any interference from the developer their software usage and feedback were noted and appropriate actions was taken if the software breaks / crashes / unexpected behaviour is observed during the test.

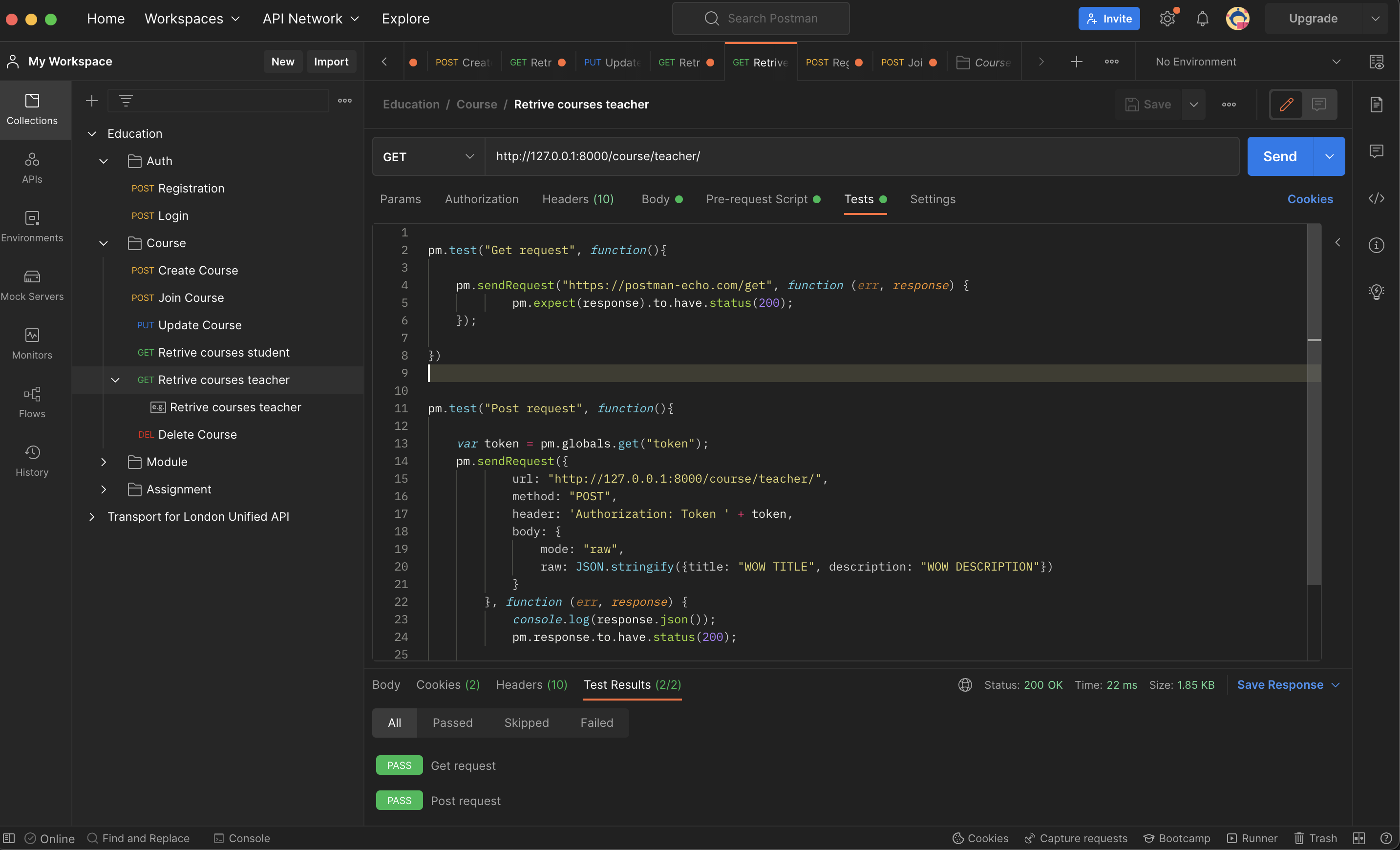


Figure 9: Unit testing in Postman

A snippet of the unit test code. Similar tests were written for all the URLs and for different types of HTTP request like GET, POST, PATCH etc.



Figure 10: Code for a single Unit Test

# Evaluation

All thought the objectives that were stated in the previous reports have been achieved. This software still has some weaknesses that should be delt with in the future.

## Weakness

Canvas and Code Grade websites were the main inspiration behind this project. Comparing the project with the above two websites following features could be observed that this project lacks.

1. Canvas has features like, *Timetable* where you can view your weekly / monthly timetable, built-in support for *messaging* with fellow students and professors, built-in support to conduct online examinations using formats like MCQs, Essays and much more.
2. Code Grade has features like, support for multiple programming languages

Implementation wise, limitations could be found in the Live Code Share module.

1. Live Code Share,as explained in the section 6.1.3, Live Code Share utilized long polling. Though it works fine when dealing with small number of users at a time, this approach could crash the servers when handling large number of active users all at once. One way to handle / fix this is to use WebSocket’s that will allow two-way communication between the server and the client, eliminating the need to send a HTTP request to the server every five seconds and also lowers the load on the server.

## Strengths

1. **Ease of use:** Unlike Code Grade and Canvas, currently this website is relatively small and doesn’t require much time to learn. Whereas canvas providing large numbers of features needs efforts to learn them. As for Code Grade, creating a programming assignment is not a very user-friendly process.
2. **Code Executor:** Code executor works flawlessly. Providing appropriate error and success messages. Currently, it only supports Python but in future more programming support could be added.
3. **Domain Name Filtering** Professors can easily manage every aspect of their courses, from basic information like course description, title and cover picture to advanced features like domain name filtering, which prevents users form joining a course if their email address domain name is different from the required domain name, even if they have the course invite token. Instead, a request is sent to the professor which professor can then either accept / decline.
4. **Course Clone:** Usually whenever a semester / year ends, professors have to recreate a course with (usually) same content as the previous year. Instead of reuploading them every year professors can clone their course which copies the modules and the assignments.
5. **Plagiarism Detector:** Plagiarism detector is effortlessly able to detect plagiarism among the submissions. In the future it could be modified to be able to detect plagiarism from online sources as well.
6. **Two-in-one:** This website combines the features of Code Grade and Canvas into a single website.

* Unlike CodeGrade: This website can store lecture notes and recordings and organized courses
* Unlike Canvas: This website allows users to write code and provide real-time feedback

1. **Fulfilling the objectives & Aims:** As mentioned in Chapter 2. This website have achieved the required Aims and Objectives.

# Learning Points

Throughout the duration of this module COMP702, I got to learn several new things which are not only limited to hard skills but also soft skills.

I believe, these skills are definitely going to be a huge help in my career. Some of the hard and soft skills that I learned are

* **Presentation:** I learned a lot of creating and giving presentations. I had no prior experience with giving presentation and working on this project really helped me in gaining confidence when it comes to giving presentations.
* **Ethics:** Through this project I learned about ethics that one must always follow like referencing the document / article that you have used and Code of Conduct that BCS expects every IT-Professional to follow.
* **ReactJS:** Learnt ReactJS to create the frontend
* **TailwindCSS:** Learnt TailwindCSS for styling frontend.
* **WebSocket:** Some of the features of this project required usage of WebSocket. It helped me a lot to understand what WebSocket is, why and when we use them.
* **Multiprocessing:** In order to learn the best method to implement Code Executor, I had to delve into the field of multi-processing which could dramatically improve the Code Executor performance.
* **Serializers & Django-Rest-Framework:** I have been using Django for some time now, but through this project I found out of Serializers and learned to use Django-Rest-Framework which makes creating RestAPI an easy task.
* **Plagiarism Detection:** Before starting this project, I had no knowledge about the working of plagiarism detection. But I got to learn about plagiarism detection using TF-IDF and Cosine Similarity formulas. I also got to learn about other usages of TF-IDF which are for search query example, searching for a movie in Netflix.

# Professional Issues

British Computer Society (BCS) has laid down some guidelines that every IT professional should practice. BCS Code of Conduct serves as a set of ethical standards for professionals in the fields of computing. BCS Code of Conduct consists of four principles.

## Public Interest

* This website doesn’t discriminate anyone regardless of their race, sex, gender and age.
* It respects the right of everyone and provides equal access to its services

## Professional Competence & Integrity

* Before taking up this project, I knew I was competent and that I had the resources as well the knowledge to complete this project.
* In areas where I lacked the knowledge, I learned about it and implemented the newly learned knowledge into the project. Example, I had to learn about Plagiarism Detection.

## Duty to Relevant Authority

* I have tried to made sure that this project follows all the rules and regulations that have been setup by the University of Liverpool.
* None of the information mentioned in any of the report have been misrepresented nor any information has been hidden.

## Duty to the Profession

* In this project everything that have been done doesn’t break any of the BCS Code of Conduct nor this project involves anything that could harm the reputation of the profession.

# Conclusion

At the end, I can say that the project was a success. Aims like Eliminating the need to setup the development environment, promoting team work through group assignments, Access to uploaded lecture videos and notes, Monitoring class performances using graphs, Automatic grading of assignments eliminating the exhausting old way of grading assignments, Plagiarism detection on the submitted reports were achieved using features like Live Code Share, Plagiarism Detector, Code Executor and Module Management System.

There is definitely more room for improvements and in the future features like Timetable, Support for multiple programming languages, GitHub / File uploading capabilities, Class Exam, A better plagiarism detection method could be implemented. Code Executor could be improved by utilizing WebSocket and Multiprocessing, Implementing Notification module to provide notifications related to new assignments, deadlines, grades could be sent out.

# Bibliography

Stack Overflow. (n.d.). Stack Overflow Developer Survey 2023. [online] Available at: <https://survey.stackoverflow.co/2023/#most-popular-technologies-webframe> [Accessed 3 Jul. 2023].

BCS (2022). *BCS Code of Conduct | BCS*. [online] www.bcs.org. Available at: <https://www.bcs.org/membership-and-registrations/become-a-member/bcs-code-of-conduct/>.

Wikipedia Contributors (2019). *tf–idf*. [online] Wikipedia. Available at: <https://en.wikipedia.org/wiki/Tf%E2%80%93idf>.

Indriyanto, I. and Sumitra, I.D. (2019). Measuring the Level of Plagiarism of Thesis using Vector Space Model and Cosine Similarity Methods. *IOP Conference Series: Materials Science and Engineering*, 662, p.022111. doi: <https://doi.org/10.1088/1757-899x/662/2/022111>

Cheang, B., Kurnia, A., Lim, A. and Oon, W.-C. (2003). On automated grading of programming assignments in an academic institution. Computers & Education, 41(2), pp.121–131. doi: <https://doi.org/10.1016/s0360-1315(03)00030-7>

Porter, L., Bouvier, D., Cutts, Q., Grissom, S., Lee, C., McCartney, R., Zingaro, D. & Simon, B. (2016) A multi-institutional study of peer instruction in introductory computing. Proceedings of the 47th ACM Technical Symposium on Computing Science Education. New York, ACM. pp. 358–363.

Novak, J. & Cañas, A. (2008) The Theory Underlying Concept Maps and How to Construct and Use Them. Florida Institute for Human and Machine Cognition. Technical Report IHMC CmapTools 2006-01 Rev 01-2008.

Maton, K. (2013) Making semantic waves: A key to cumulative knowledge-building. Linguistics and Education. 24(1), 8–22.

Teach Computing. (n.d.). *Pedagogy*. [online] Available at: <https://teachcomputing.org/pedagogy>

Jayathirtha, G., & Kafai, Y. B. (2021, June). Program Comprehension with Physical Computing: A Structure, Function, and Behavior Analysis of Think-Alouds with High School Students. In Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V. 1 (pp. 143-149).

Simplilearn.com. (2021). *Multiprocessing in Python - Running Multiple Processes in Parallel [Updated]*. [online] Available at: <https://www.simplilearn.com/tutorials/python-tutorial/multiprocessing-in-python> [Accessed 20 Sep. 2023].

react.dev. (n.d.). *Quick Start*. [online] Available at: <https://react.dev/learn>.

# Appendix

## Setting up the website

### Software Requirements

* Python 3.10.6
* NPM 9.5.1
* ReactJS 18.2.0
* Pip 23.0.1
* PostgreSQL 14.8 (optional)

### Installation Instructions

After downloading the required software

1. Create a virtual environment
   1. Open up a terminal
   2. Install a virtualenv library using this command ***pip install virtualenv*** if you already don’t have one.
   3. Create a virtualenv using ***virtualenv <your\_virtual\_env\_name>***
   4. Activate the virtual env using ***source*** ***./your\_virtual\_env\_name/bin/activate*** on mac and on windows ***./your\_virtual\_env\_name \Scripts\activate***
   5. You have now activated your virtualenv
2. Install required python libraries
   1. navigate to ***cd project\_files/backend/*** and run the command ***pip install -r requirements.txt***
3. If you have decided to **NOT** use PostgreSQL. You will be using SQLite. Which will work but might be a little slower. If you have decided to use PostgreSQL, go to step 5 otherwise go to step 4.
4. Setting up SQLite
   1. By default, SQLite is enabled from the settings. So, you don’t have to do anything else. Now go to step 6
5. Setting up PostgreSQL
   1. To enable PostgreSQL, open the PostgreSQL pgAdmin and create a database for this project

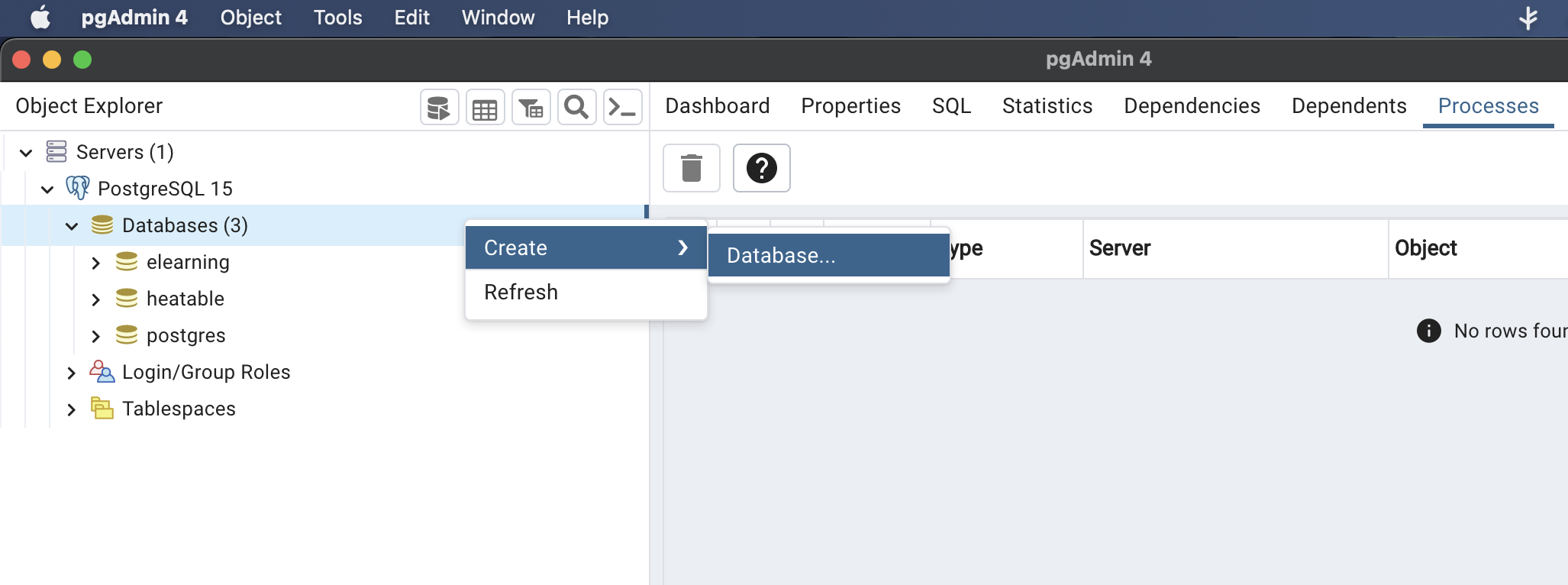


Figure 11: Creating a DB from PgAdmin

* 1. Go to.***/project\_files/backend/backend/settings.py*** and comment out the SQLite database code and uncomment the PostgreSQL code

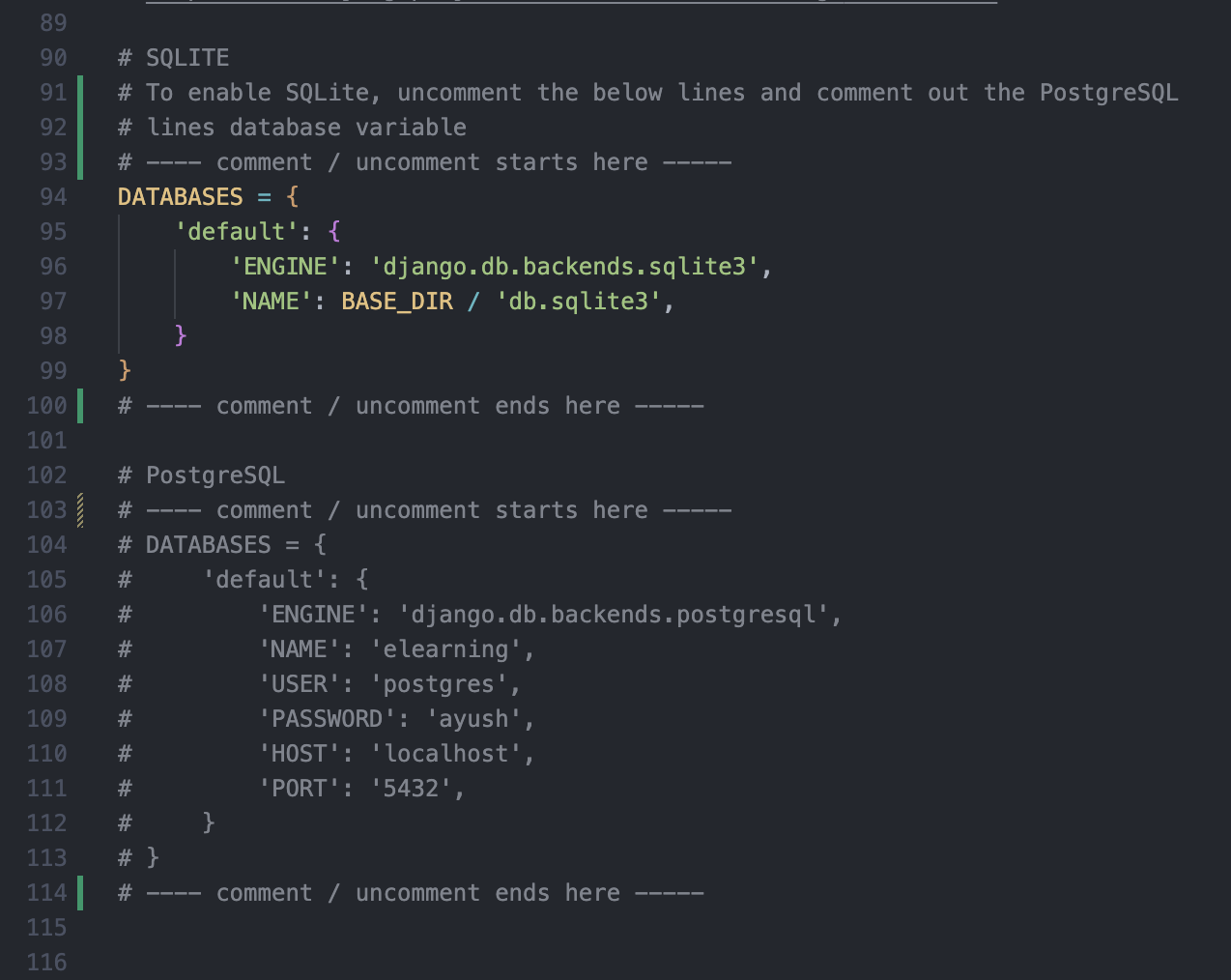
******

Figure 12: Settings.py to enable / disable a database

* 1. Replace the
     1. line 107 ‘elearning’ with the name of your database
     2. line 108 ‘postgres’ with your username
     3. line 109 ‘ayush’ with your PostgreSQL password
     4. line 110 ‘localhost’, you don’t really have to change this unless your database is live in some other server, if so? Replace it with the hostname of your database
     5. *line 111 ‘5432’ by default PostgreSQL uses this port and you too might be using this same port (please do make sure your PostgreSQL is using this port) otherwise change the port number.*

1. Migrating your database
   1. *Run the following commands* **python manage.py makemigrations** *then* ***python manage.py migrate***
2. Creating your first user
   1. *In the terminal, run the following command* ***python manage.py createsuperuser*** and follow the onscreen instructions
3. Start your backend server using this command **python manage.py runserver**
4. Your backend server should be up and running now. let’s move to frontend.
5. Go to ***cd ./project\_files/frontend/***and run the following command ***npm install***
6. Start your frontend using ***npm run dev***

## Project Log

|  |  |
| --- | --- |
| **Date** | **Description** |
| 26th May 2023 | Project allotted |
| 5th June 2023 | Project officially started |
| 6th June 2023 | First meeting with the supervisor |
| 20th June 2023 | Second Meeting with the supervisor |
| 29th June 2023 | Completed RestAPI development and started working on the frontend |
| 4th July 2023 | Third Meeting with the supervisor |
| 13th July 2023 | Fourth Meeting with the supervisor |
| 17th July 2023 | Finished frontend development and started working on Plagiarism Detection, Live Code Share and Code Executor |
| 2nd August 2023 | Fifth Meeting with the supervisor |
| 22nd August 2023 | Sixth meeting with the supervisor |
| 24th August | Finished working on Plagiarism Detection, Live Code Share and Code Executor |
| 28th August | Completed the testing of the entire system |
| 30th August 2023 | Seventh meeting with the supervisor |
| 1st September 2023 | Final presentation submitted |
| 5th September | Q&A with both the supervisors |
| 19th September 2023 | Eighth meeting with the supervisor |
| 21st September 2023 | Final dissertation report submitted |
| 22nd September 2023 | Dissertation official ended |

Table 3: Project Log