**Online Learning Platform using MERN**

**INTRODUCTION :**

An Online Learning Platform (OLP) is a comprehensive digital space that enables the delivery of educational content and the facilitation of learning via the internet. These platforms have gained immense popularity in recent years due to their ability to provide accessible, flexible, and cost-effective education. OLPs cater to learners of all ages, backgrounds, and geographical locations, making them an essential tool for both formal and informal education.

Key Features of an Online Learning Platform:

1. User-Friendly Interface: One of the defining features of a successful OLP is its intuitive, easy-to-navigate interface. Whether the learner is a beginner with limited technical skills or an expert user, the platform ensures smooth interaction. A clean, well-organized design with clear instructions helps users quickly find and access the learning materials, courses, and additional resources.
2. Course Management: OLPs offer powerful course management tools that allow instructors or educational institutions to easily upload, organize, and distribute course materials. Course content may include videos, readings, quizzes, assignments, and other learning resources. The platform often allows instructors to monitor learners’ progress, offer feedback, and update course materials as needed.
3. Interactivity: Modern online learning platforms place a strong emphasis on creating an interactive learning experience. Many platforms integrate discussion forums, chat rooms, group projects, and live webinars to encourage communication and collaboration between learners and instructors. These interactive elements help create a more engaging and dynamic learning environment, allowing learners to ask questions, participate in debates, and clarify doubts in real time.
4. Certification: Upon completing a course or program, learners often receive certificates or badges that serve as official recognition of their skills and achievements. These credentials can be valuable additions to a learner's professional profile, enhancing their qualifications for future employment or higher education. Some platforms also offer certifications that are recognized by accredited institutions or employers, adding credibility to the learner's qualifications.
5. Accessibility: One of the greatest advantages of OLPs is their accessibility across multiple devices. Learners can access content from their computers, tablets, or smartphones, allowing them to learn from virtually anywhere with an internet connection. This device compatibility makes it easier for people to fit learning into their busy schedules, whether they are at home, at work, or on the go.
6. Self-Paced Learning: Many OLPs offer the flexibility of self-paced learning, allowing learners to study at their own speed. This approach benefits those who may have varying levels of prior knowledge or who need to balance their education with work or family commitments. Self-paced learning helps learners take control of their educational journey by enabling them to revisit difficult content and skip over material they already understand.
7. Payment and Subscription Options: While many online learning platforms provide free access to certain courses, they may also offer paid content or subscription-based services for premium courses, certifications, or specialized programs. This flexible pricing structure caters to a diverse range of learners, from those seeking free resources to those willing to invest in more advanced or professional courses. Subscription models may include monthly or yearly plans, with discounts for long-term subscriptions.
8. Engagement and Gamification: To increase learner engagement, some OLPs incorporate elements of gamification, such as quizzes, leaderboards, and achievement badges. These features make learning more fun and competitive, motivating students to complete tasks and engage with course content regularly. Gamified features can encourage learners to achieve milestones and foster a sense of accomplishment as they progress.

**DESCRIPTION :**

An **Online Learning Platform (OLP)** is a digital platform that provides various tools and resources for education over the internet, offering flexibility and accessibility to learners of all ages. These platforms typically feature a **user-friendly interface** for easy navigation, **course management** tools for instructors to upload and organize materials, and **interactive elements** like forums and live webinars to enhance collaboration. Learners can earn **certificates** or **badges** upon course completion, and the content is accessible across **multiple devices**, allowing learning from anywhere. With **self-paced learning**, students can progress at their own speed, while **payment and subscription options** offer both free and paid courses.

**Scenario :** Learning a New Skill

**User Registration:** Sarah, a student interested in learning web development, visits the Online Learning Platform and creates an account. She provides her email and chooses a password.

**Browsing Courses:** Upon logging in, Sarah is greeted with a user-friendly interface displaying various courses categorized by topic, difficulty level, and popularity.

She navigates through the course catalog, filtering courses by name and category until she finds a "Web Development Fundamentals" course that interests her.

**Enrolling in a Course:** Sarah clicks on the course and reads the course description, instructor details, and syllabus. Impressed, she decides to enroll in the course.

After enrolling, Sarah can access the course materials, including video lectures, reading materials, and assignments.

**Learning Progress:** Sarah starts the course and proceeds through the modules at her own pace. The platform remembers her progress, allowing her to pick up where she left off if she needs to take a break.

**Interaction and Support:** Throughout the course, Sarah engages with interactive elements such as discussion forums and live webinars where she can ask questions and interact with the instructor and other learners.

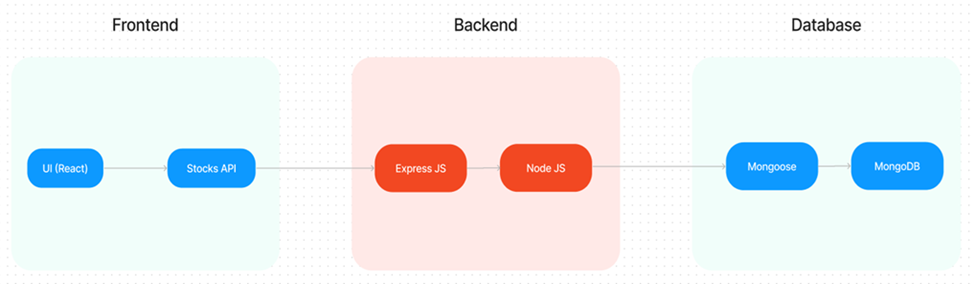
**Course Completion and Certification:** After completing all the modules and assignments, Sarah takes the final exam. Upon passing, she receives a digital certificate of completion, which she can download and add to her portfolio.

**Paid Courses:** Sarah discovers an advanced web development course that requires payment. She purchases the course using the platform's payment system and gains access to premium content.

**Teacher's Role:** Meanwhile, John, an experienced web developer, serves as a teacher on the platform. He creates and uploads new courses on advanced web development topics, adds sections to existing courses, and monitors course enrollments.

**Admin Oversight:** The admin oversees the entire platform, monitoring user activity, managing course listings, and ensuring smooth operation. They keep track of enrolled students, handle any issues that arise, and maintain the integrity of the platform.

**TECHNICAL ARCHITECTURE**



The technical architecture of OLPapp follows a client-server model, where the frontend serves as the client and the backend acts as the server. The frontend encompasses not only the user interface and presentation but also incorporates the axios library to connect with backend easily by using RESTful Apis.

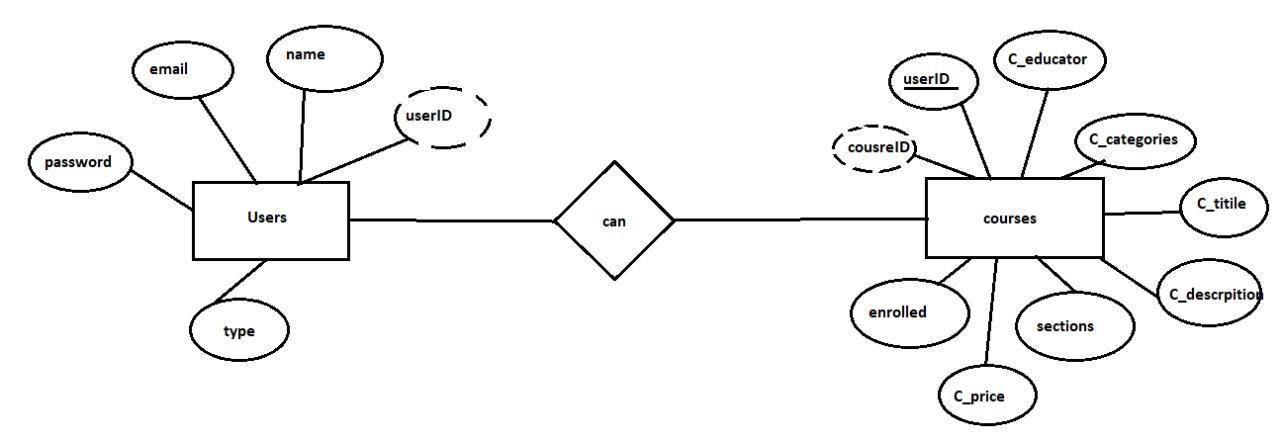
The frontend utilizes the bootstrap and material UI library to establish real-time and better UI experience for any user.

On the backend side, we employ Express.js frameworks to handle the server-side logic and communication.

For data storage and retrieval, our backend relies on MongoDB. MongoDB allows for efficient and scalable storage of user data and necessary information about the place.

Together, the frontend and backend components, along with Express.js, and MongoDB, form a comprehensive technical architecture for our OLPapp. This architecture enables real-time communication, efficient data exchange, and seamless integration, ensuring a smooth and immersive blogging experience for all users.

**ER DIAGRAM**



This is the **ER diagram** of the online learning platform, which shows the relationship between **Users** and **Courses**. It highlights how users, with fields like email, name, password, and type, can interact with courses by enrolling or creating them.

The diagram also represents how courses are structured with fields like C\_title, C\_categories, C\_description, C\_educator, C\_price, and sections. The relationship ensures that a user (e.g., an educator or a learner) can either create or enroll in courses, depending on their role (type).

This ER diagram provides a clear view of the database structure and the flow of information within the platform, helping us understand how users interact with the courses and how entities relate to each other.

**Pre-requisites**

Here are the key prerequisites for developing a full-stack application using Node.js, Express.js, MongoDB, React.js:

**Vite:**

Vite is a new frontend build tool that aims to improve the developer experience for development with the local machine, and for the build of optimized assets for production (go live). Vite (or ViteJS) includes: a development server with ES \_native\_ support and Hot Module Replacement; a build command based on rollup.

**npm create vite@latest**

**Node.js and npm:**

Node.js is a powerful JavaScript runtime environment that allows you to run JavaScript code on the server-side. It provides a scalable and efficient platform for building network applications.

Install Node.js and npm on your development machine, as they are required to run JavaScript on the server-side.

Download: <https://nodejs.org/en/download/>

Installation instructions: <https://nodejs.org/en/download/package-manager/>

**npm init**

**Express.js:**

Express.js is a fast and minimalist web application framework for Node.js. It simplifies the process of creating robust APIs and web applications, offering features like routing, middleware support, and modular architecture.

Install Express.js, a web application framework for Node.js, which handles server-side routing, middleware, and API development.

Installation: Open your command prompt or terminal and run the following command:

**npm install express**

**MongoDB:**

MongoDB is a flexible and scalable NoSQL database that stores data in a JSON-like format. It provides high performance, horizontal scalability, and seamless integration with Node.js, making it ideal for handling large amounts of structured and unstructured data.

Set up a MongoDB database to store your application's data.

Download: <https://www.mongodb.com/try/download/community>

Installation instructions: <https://docs.mongodb.com/manual/installation/>

**React.js:**

React.js is a popular JavaScript library for building user interfaces. It enables developers to create interactive and reusable UI components, making it easier to build dynamic and responsive web applications.

Install React.js, a JavaScript library for building user interfaces.

Follow the installation guide: <https://reactjs.org/docs/create-a-new-react-app.html>

**HTML, CSS, and JavaScript**:

Basic knowledge of HTML for creating the structure of your app, CSS for styling, and JavaScript for client-side interactivity is essential.

**Database Connectivity**:

Use a MongoDB driver or an Object-Document Mapping (ODM) library like Mongoose to connect your Node.js server with the MongoDB database and perform CRUD (Create, Read, Update, Delete) operations. To Connect the Database with Node JS go through the below provided link:

[https://www.section.io/engineering-education/nodejs- mongoosejs-mongodb/](https://www.section.io/engineering-education/nodejs-%20mongoosejs-mongodb/)

**Front-end Framework**:

Utilize Reactjs to build the user-facing part of the application, including entering booking room, status of the booking, and user interfaces for the admin dashboard.

For making better UI we have also used some libraries like material UI and boostrap.

Install Dependencies:

• Navigate into the cloned repository directory:

cd containment-zone

• Install the required dependencies by running the following commands:

cd frontend

npm install

cd ../backend

npm install

Start the Development Server:

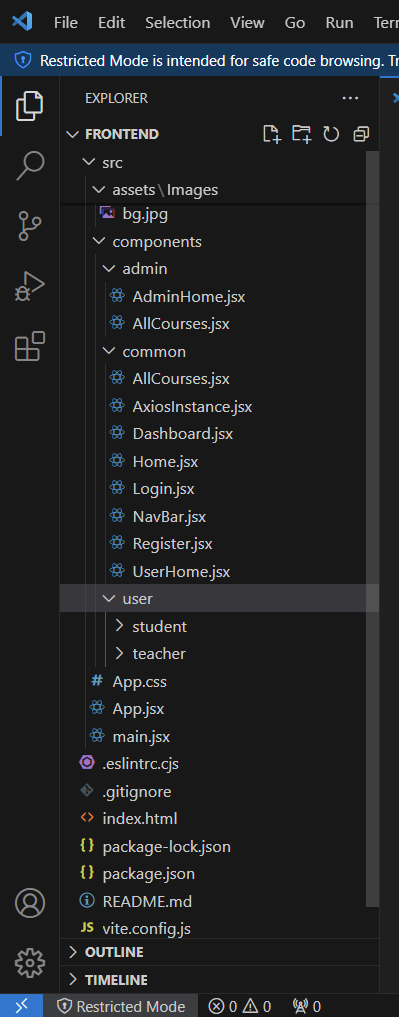
• To start the development server, execute the following command:

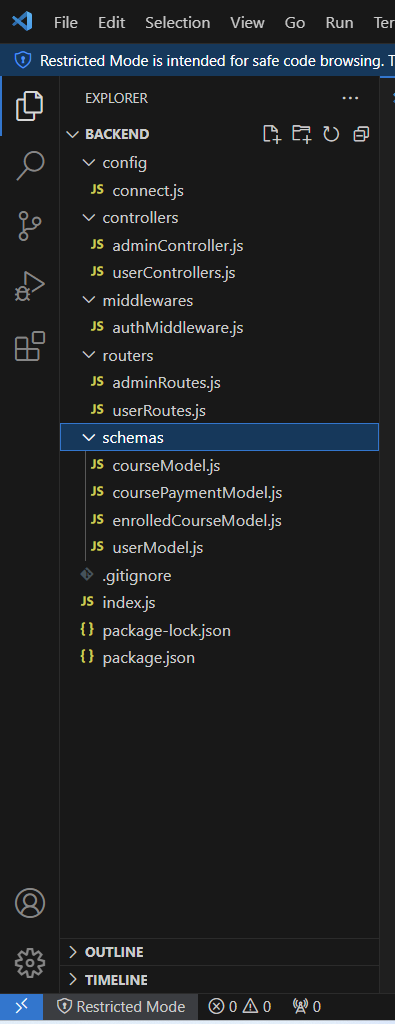
npm start

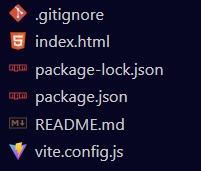
• The OLPapp will be accessible at [http://localhost:5172](http://localhost:5172/)

You have successfully installed and set up the Online leraning app on your local machine. You can now proceed with further customization, development, and testing as needed.

**PROJECT STRUCTURE:**







The first image is of frontend part which is showing all the files and folders that have been used in UI development.

The second image is of Backend part which is showing all the files and folders that have been used in backend development.

**Application Flow**

The project has a user called– teacher and student and other will be Admin which takes care of all the user. The roles and responsibilities of these users can be inferred from the API endpoints defined in the code. Here is a summary:

**Teacher:**

1. Can add courses for the student.
2. Also delete the course if no student enrolled in it or any other reasons.
3. Also add sections to courses.

**Student:**

1. Can enroll in an individual or multiple course.
2. Can start the course where it has stopped.
3. Once the course is completed, they can download their certificate of completeion of the **course**.
4. For paid course, they need to purchase it and then they can start the course.
5. They can filter out the course by searching by name, category, etc

**Admin:**

1. They can alter all the course that are present in the app.
2. Watch out all kind of users in app.
3. Record all the enrolled all the student that are enrolled in course.

# **Project Flow :**

Before starting to work on this project, let’s see the demo.

**Project demo:**

Use the code in:

or follow the videos below for better understanding

**STEP 1**

**Project Setup And Configuration**

Duration: 2 Hrs

Skill Tags:

* **Folder setup:**
  + Create frontend and
  + Backend folders

 Open the backend folder to install necessary tools

For backend, we use:

* + cors
  + bcryptjs
  + express
  + dotenv
  + mongoose
  + Multer
  + Nodemon
  + Jsonwebtoken



REFERENCE VIDEO : <https://drive.google.com/file/d/1OvIQ_Dr6w3QyYe5WxsqL2lIN57XTwkB0/view?usp=drive_link>

STEP 2 :

**Backend DevelopmenT**

Skill Tags:

* **Setup express server**
  + Create index.js file in the server (backend folder).
  + define port number, mongodb connection string and JWT key in env file to access it.
  + Configure the server by adding cors, body-parser.
* **Add authentication:** for this,
  + You need to make middleware folder and in that make authMiddleware.js file for the authentication of the projects and can use in.

STEP 3 :

**Database Development**

Skill Tags:

* **Configure MongoDB**
  + Import mongoose.
  + Add database connection from config.js file present in config folder.
  + Create a model folder to store all the DB schemas.

REFERENCE VIDEO [: https://drive.google.com/file/d/1NGINp2z436ll-nDR91yhfGkkObKbZZ6C/view?usp=drive\_link](:%20https:/drive.google.com/file/d/1NGINp2z436ll-nDR91yhfGkkObKbZZ6C/view?usp=drive_link)

STEP 4 :

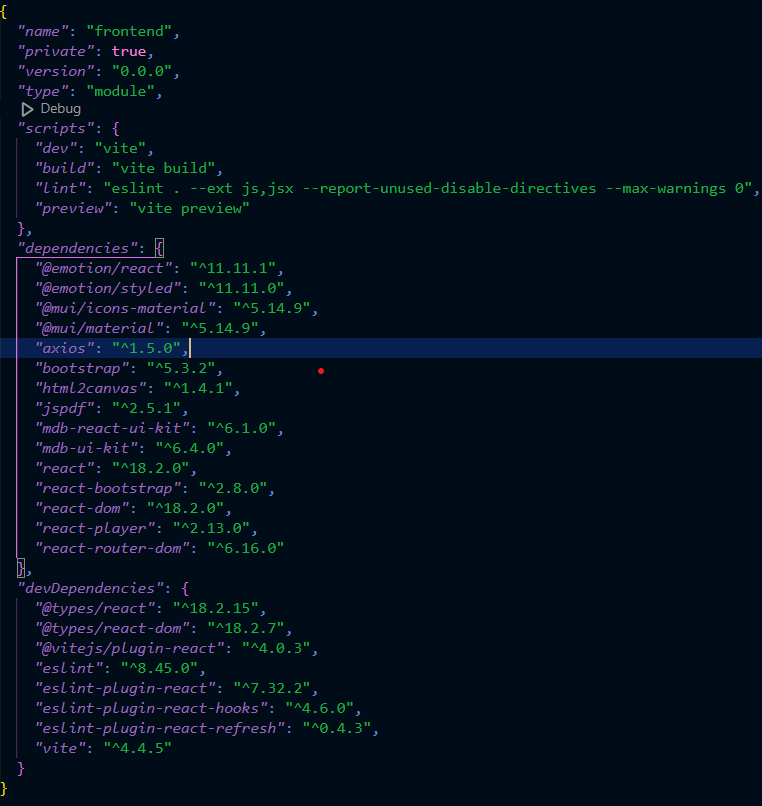
**Frontend development**

Skill Tags:

* **Installation of required tools:**

For frontend, we use:

* + React
  + Bootstrap
  + Material UI
  + Axios
  + Antd
  + mdb-react-ui-kit
  + react-bootstrap



REFERENCE VIDEO : <https://drive.google.com/file/d/1QQqWIizOfxnYJ6UxoRgT1lCu4QOVnaoY/view?usp=drive_link>

STEP 5:

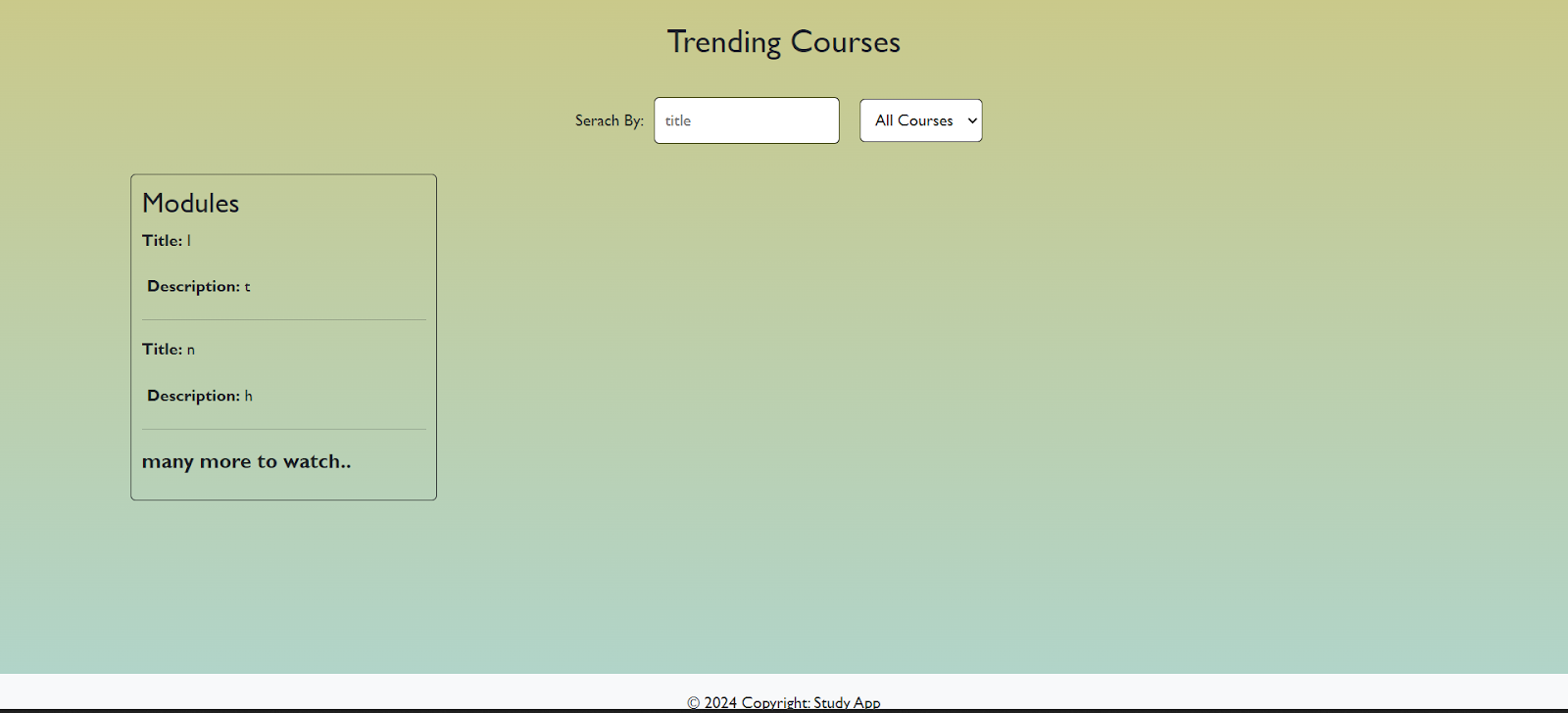
**Project Implementation & Execution**

Skill Tags:

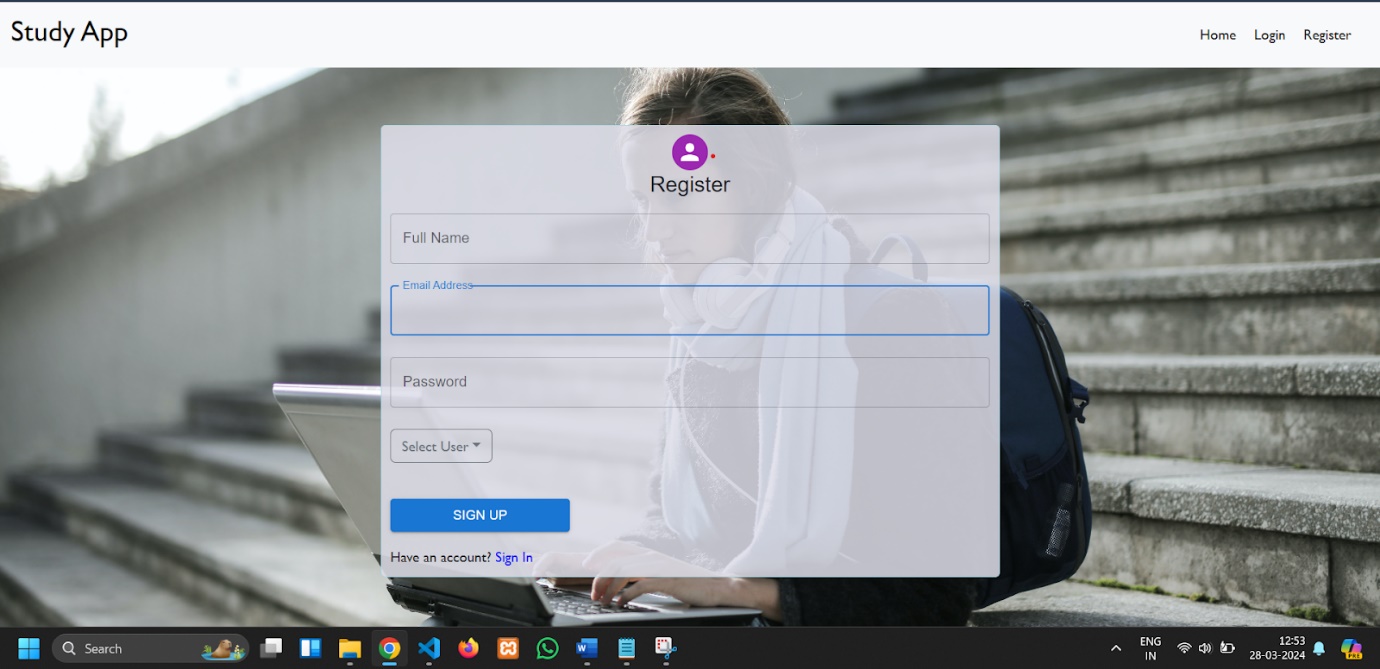
On completing the development part, we then run the application one last time to verify all the functionalities and look for any bugs in it. The user interface of the application looks a bit like the one’s provided below.

**Landing page**

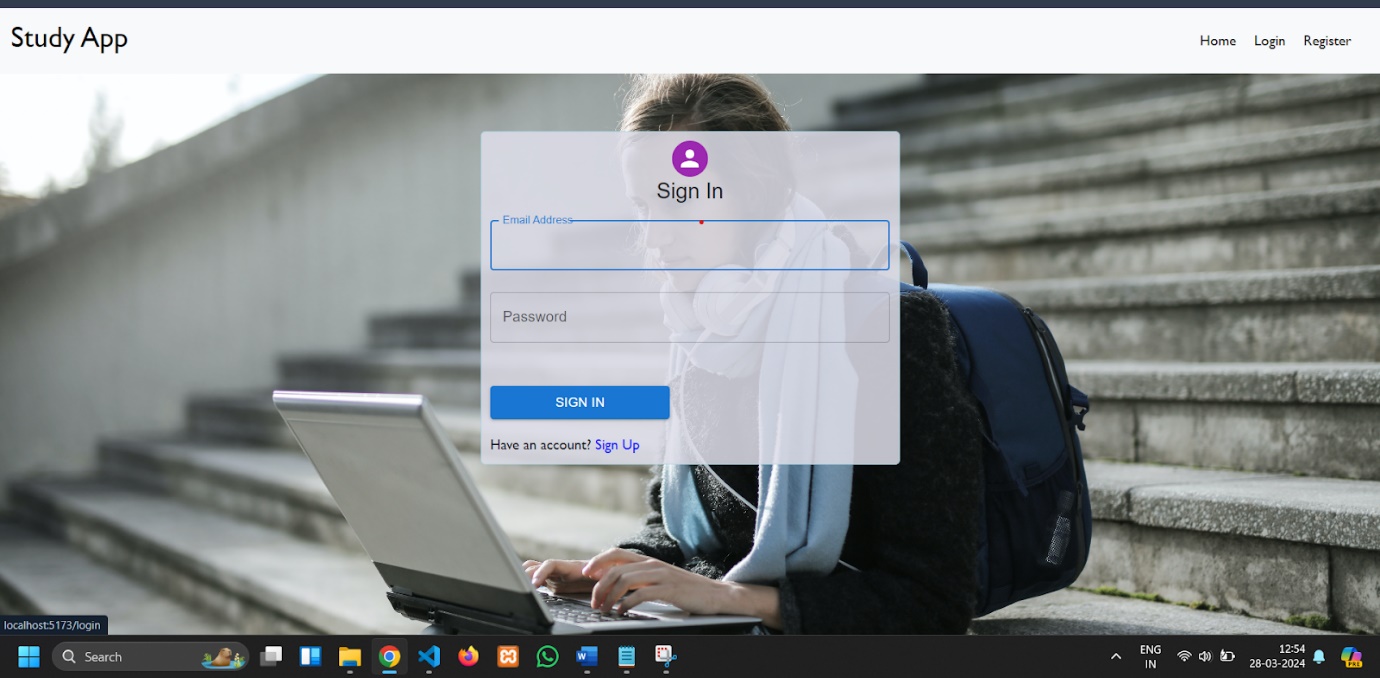




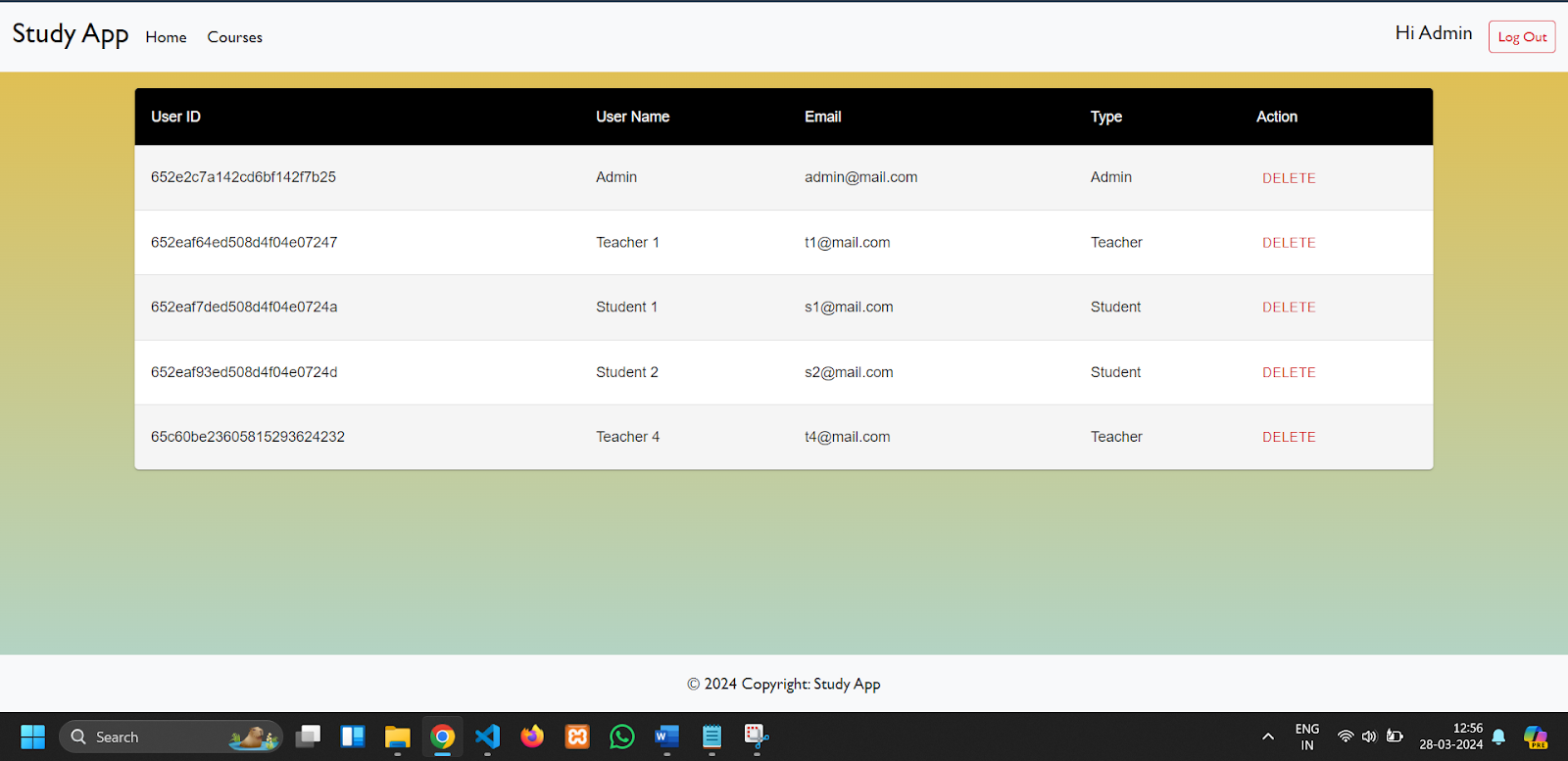
**Register page:**



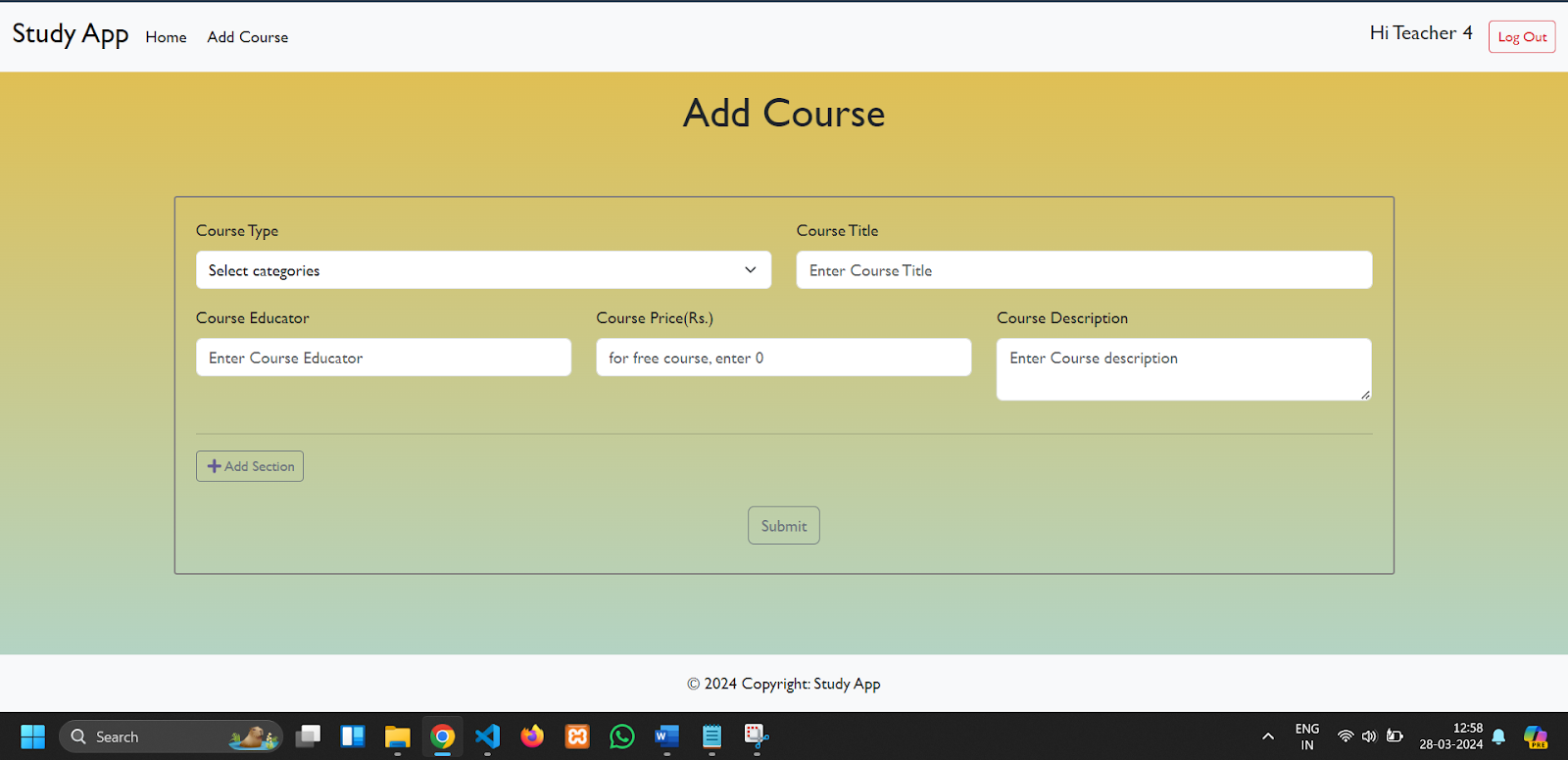
**Login page:**



**Admin Dashboard:**



**Teacher Dashboard:**



**Student Dashboard:**

