**Online Learning Platform using MERN**

Team Members:

TEAM ID : LTVIP2025TMID59463

|  |  |
| --- | --- |
| **MEMBER** | **RESPONSIBILITY** |
| 1. Kotha Naga Sai Krishna | Front end , Testing &Documentation |
| 2. Kojja Rajesh | Documentation & Devops |
| 3. Kollimarla Bhanu | Resources gathering |
| 4. Kommanaboyina Venkata Sai Krishna | Backend , Database & Implementation |

**1. Introduction**

Project Title:

Online Learning Platform using MERN

**2. Project Overview**

Purpose:  
An online learning platform(OLP) is a digital platform that provides a variety of tools and resources to facilitate learning and education over the internet. These platforms have become increasingly popular, especially in recent years, as they offer flexibility and accessibility for learners of all ages and backgrounds.

Features:

* User-Friendly Interface: Easy navigation and intuitive design for all user types.
* Course Management: Instructors can upload, organize, and manage content; learners can enroll, view, and track their progress.
* Interactive Learning: Tools like discussion forums, chat, and webinars promote engagement.
* Certification: Learners receive downloadable certificates after successful course completion.
* Cross-Device Accessibility: Compatible with desktops, tablets, and smartphones.
* Self-Paced Learning: Courses can be started, paused, and resumed at any time.
* Flexible Pricing Models: Includes both free and paid/subscription-based courses.

**3. Architecture**

Frontend:

Built using React.js, the frontend follows a modern component-based architecture to ensure modularity and scalability.

Component Structure:

Utilizes functional components with React Hooks such as useState , useEffect , and useContext for state and side-effect management.

Routing:

React Router v6 is used for client-side routing, enabling seamless navigation between pages like Home, Courses, Dashboard, and Profile.

State Management:

Uses Context API for managing global state, including user authentication, course progress, and cart/enrollment details.

UI Library:

Integrated with Material-UI (MUI) for building responsive, accessible, and modern UI components.

HTTP Requests:

Axios is used for sending and handling asynchronous API calls to the backend.

Backend:

Built using Node.js and Express.js, following a modular and RESTful architecture to support scalability and clean code separation.

API Structure:

Implements RESTful API endpoints for user authentication, course management, enrollment, progress tracking, and reviews.

Middleware:

cors for handling cross-origin requests

dotenv for environment variable configuration

morgan for request logging

express-validator for input validation

multer for file uploads (e.g., certificates, profile images)

Authentication & Authorization:

Uses JWT (JSON Web Tokens) for secure login and role-based access (Student, Teacher, Admin).

Project Structure:

Organized using the MVC pattern:

Routes – API endpoints

Controllers – Business logic

Models – Mongoose schemas

Database:

MongoDB is used as the NoSQL database, offering flexible document-based data storage.

Managed using Mongoose ORM, with schemas for Users, Courses, Enrollments, Reviews, and Certifications.

**4. Setup Instructions**

Prerequisites:

Ensure you have the following installed:

Node.js (v16.x or higher)

npm or yarn

MongoDB Atlas account or local MongoDB instance

Git

**Installation Steps:**

* Clone the repository

git clone https://github.com/Nandhuarumalla/Online\_Learning\_Platform

cd olp

* Set up environment variables:

Create .env file in the server directory

PORT=5000

MONGO\_URI=mongodb://localhost:27017/olp

JWT\_SECRET=mysecretpassword

* Install dependencies :

Backend

cd server

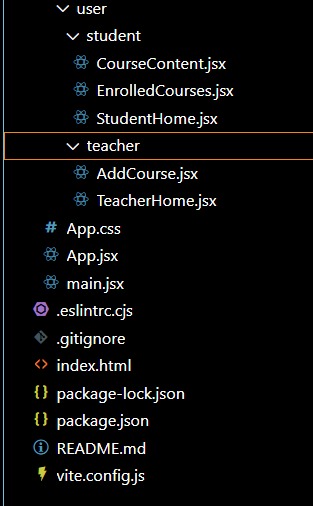
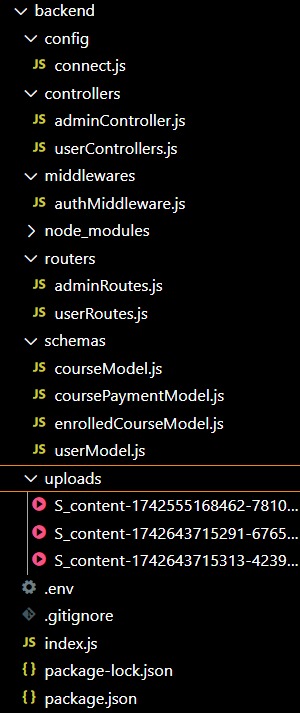
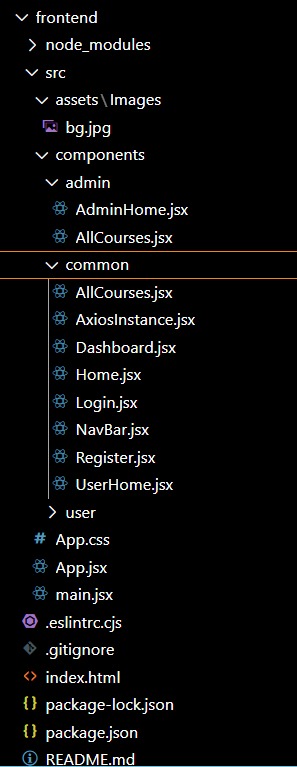
npm install

#Frontend

cd ../client

npm install

**5.Folder Structure**



**6. Running the Application**

To start both servers:

Frontend:

Navigate to the client directory:

cd client

Start the development server:

npm start

The frontend application runs on: http://localhost:3000

Backend:

Navigate to the server directory:

cd server

Start the backend server:

npm start

The backend application runs on: http://localhost:5000

**7. API Documentation**

Base URL:

http://localhost:5000/api

Auth Endpoints

POST /auth/register

Registers a new user (student, teacher, or admin) with name, email, password, and role.

POST /auth/login

Logs in a registered user and returns a JWT token for authentication.

Course Endpoints

GET /courses (Protected)

Retrieves all available courses for browsing.

POST /courses (Protected — Teacher/Admin Only)

Allows a teacher or admin to create a new course.

GET /courses/enrolled (Protected — Student Only)

Retrieves all courses the logged-in student is currently enrolled in.

POST /courses/enroll (Protected — Student Only)

Enrolls a student into a selected course.

GET /courses/:id/content (Protected — Student Only)

Retrieves learning materials (videos, notes, quizzes) for a specific course.

POST /courses/:id/progress (Protected — Student Only)

Updates the student’s progress within the course.

Teacher Endpoints

GET /teacher/courses (Protected — Teacher Only)

Retrieves all courses created by the logged-in teacher.

PUT /courses/:id (Protected — Teacher Only)

Updates an existing course’s details.

DELETE /courses/:id (Protected — Teacher/Admin Only)

Deletes a course from the system.

User Endpoints

GET /users/me (Protected)

Returns the profile information of the currently logged-in user.

GET /users/teachers (Protected — Admin Only)

Returns a list of all teachers in the system.

GET /users/students (Protected — Admin Only)

Returns a list of all registered students.

**8. Authentication**

Authentication is handled using JWT (JSON Web Tokens) :

Upon successful login, the system generates a JWT token.

This token is stored in the browser's localStorage.

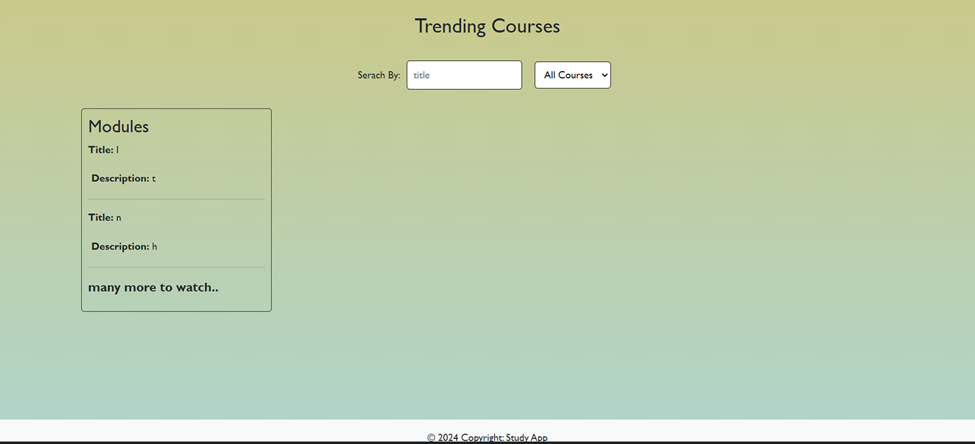
All protected routes require a valid token for access.

Middleware verifies the token and authorizes access based on user roles: student, teacher, or admin.

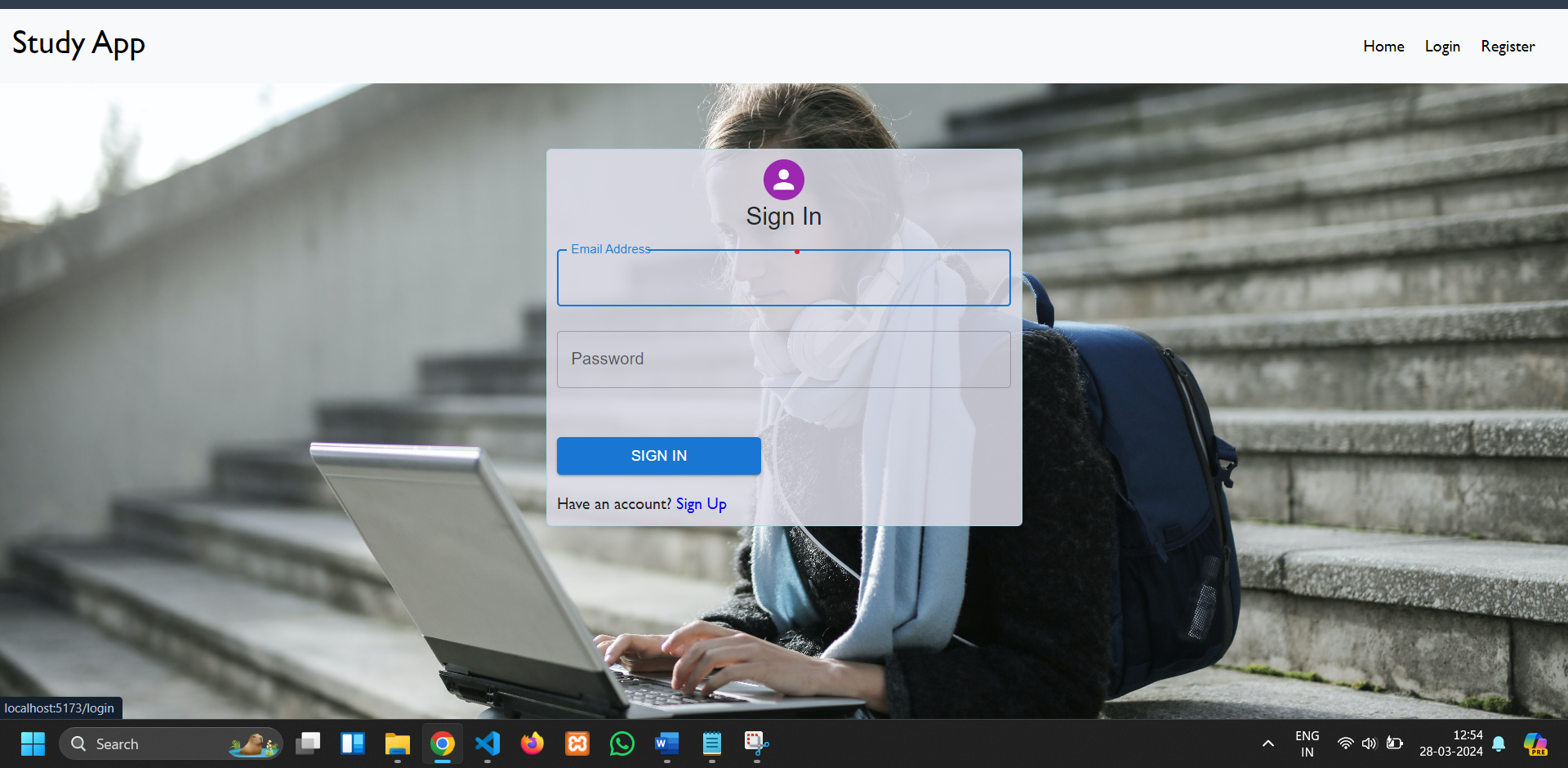
**9. Screenshots or Demo**

HOME PAGE :

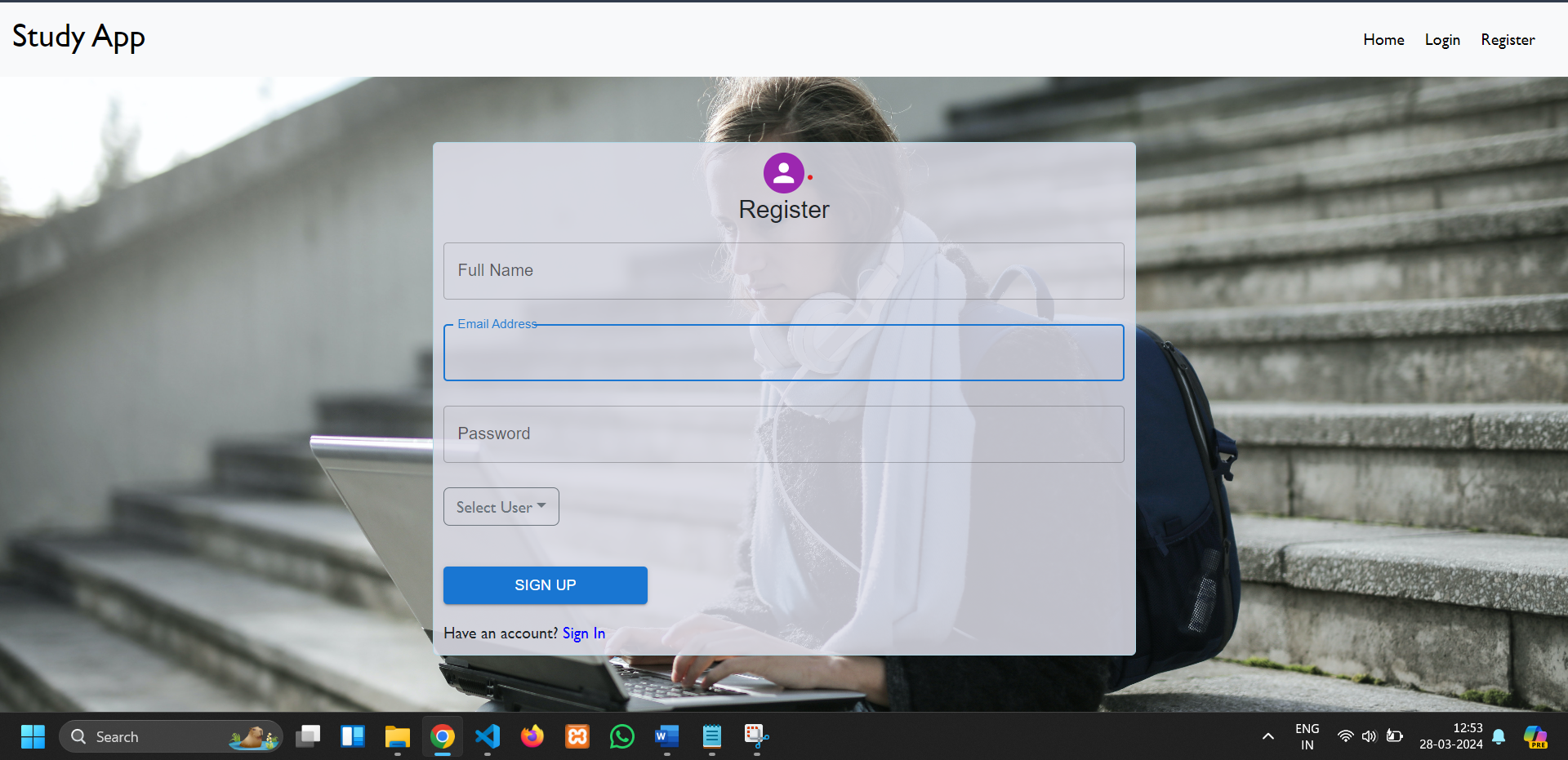


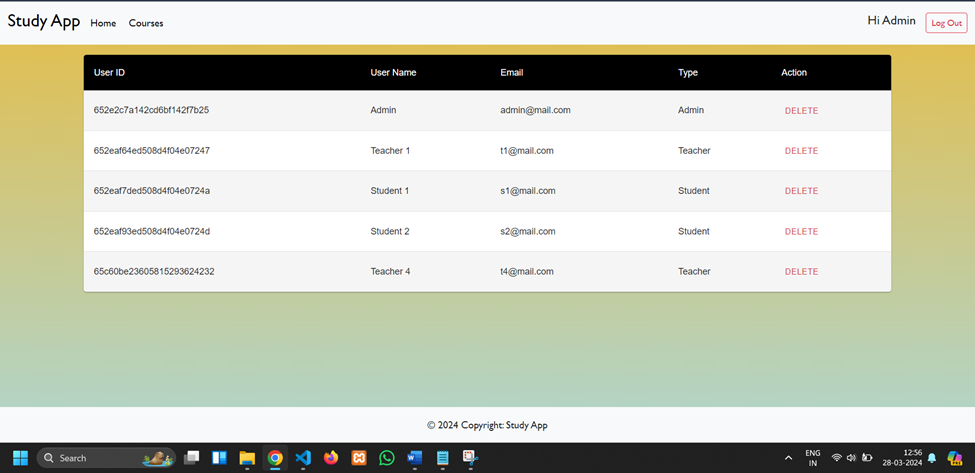


LOGIN PAGE:

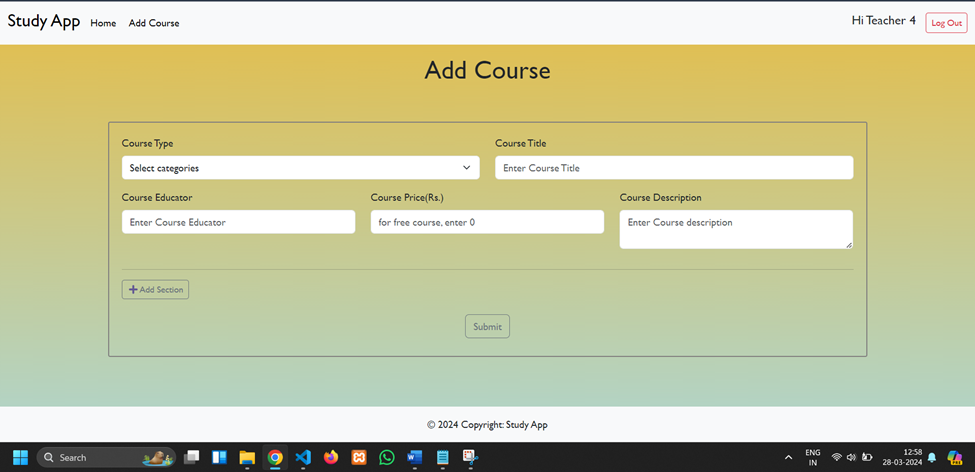


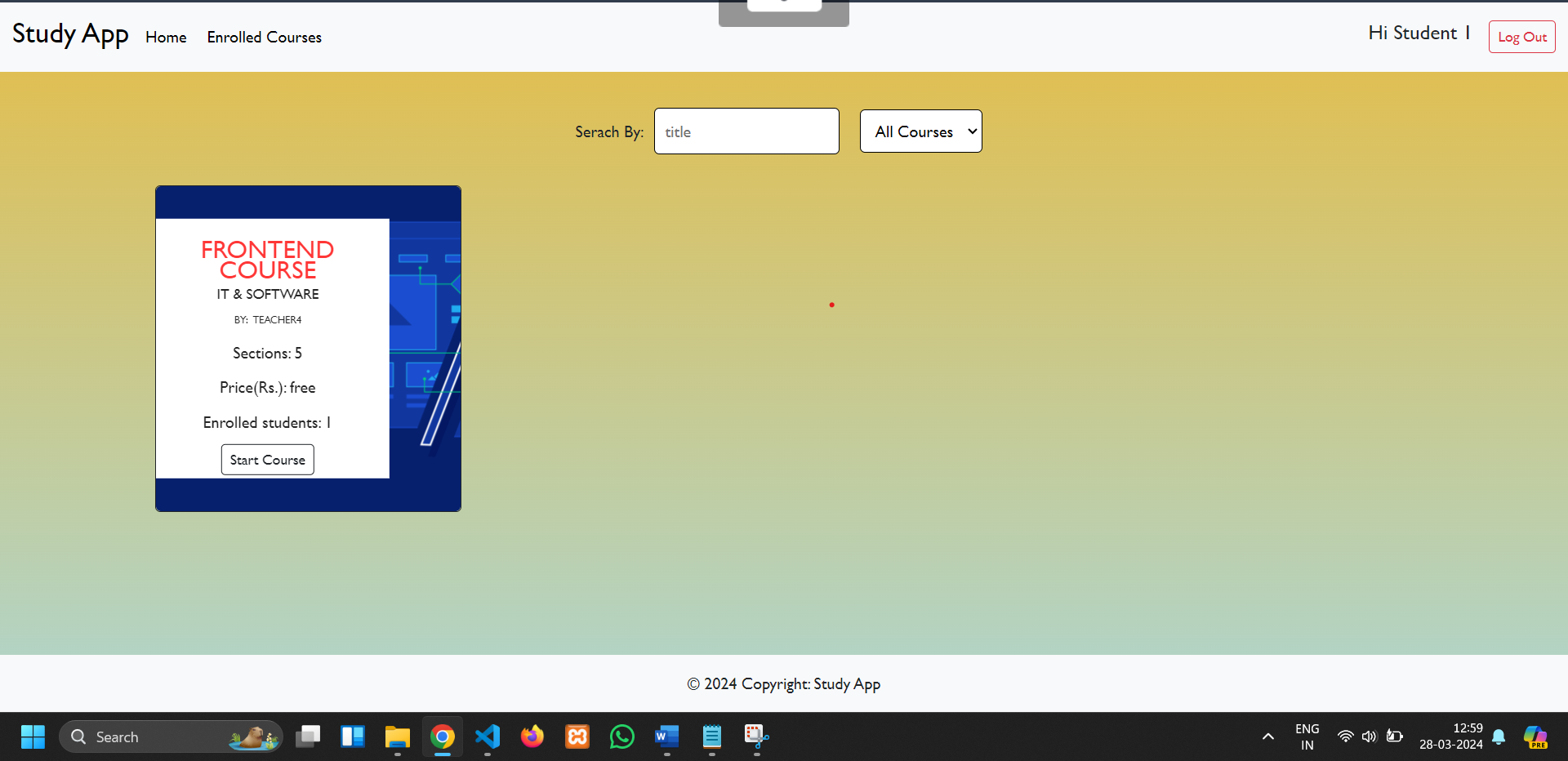
REGISTER PAGE :

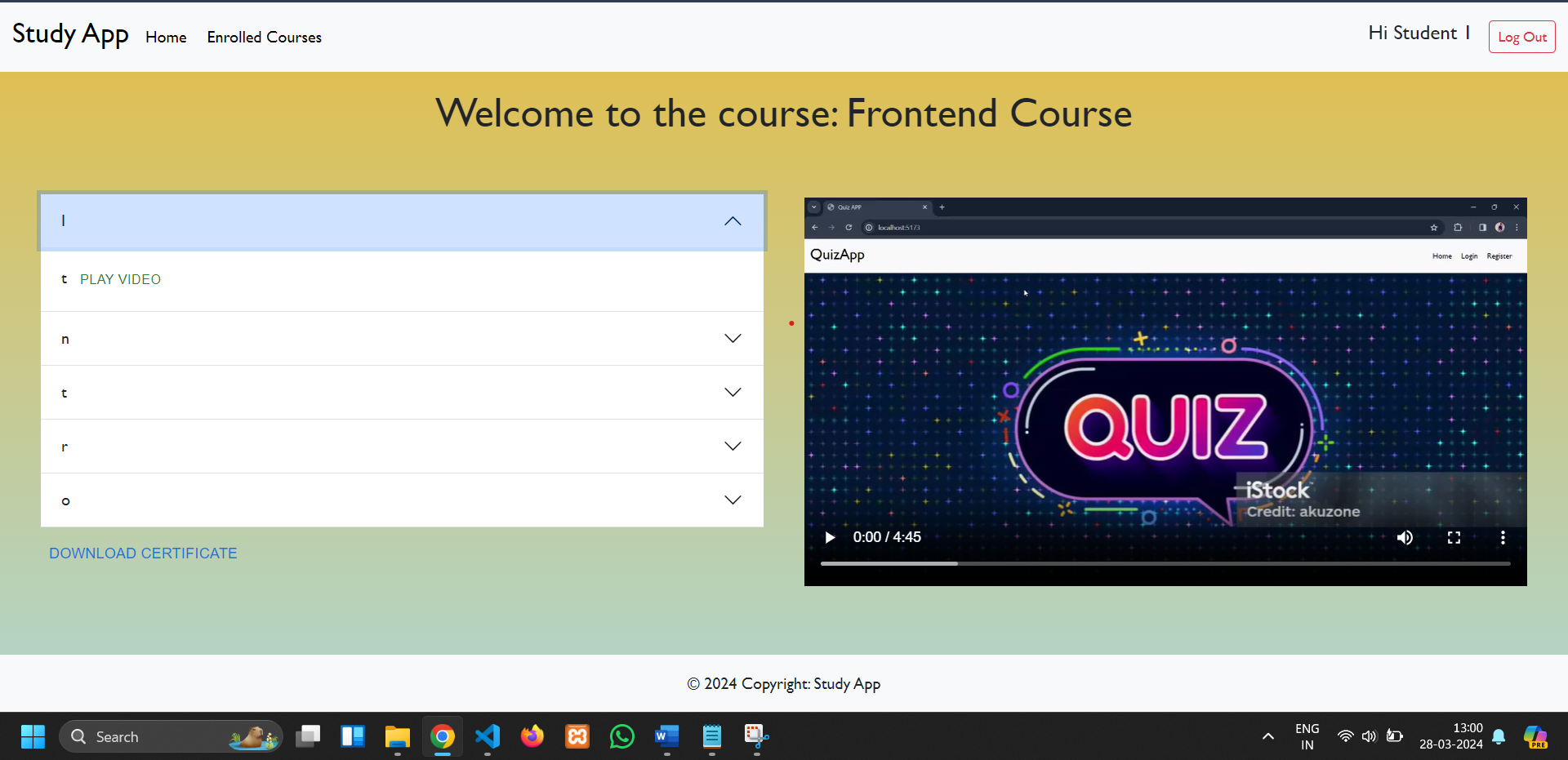


ADMIN DASHBOARD: 

TEACHER DASHBOARD:



STUDENT DASHBOARD: 



**10. Testing**

Testing Strategy:

Frontend components are tested using Jest and React Testing Library.

Backend APIs are tested using Mocha and Chai for validation of responses and error handling.

End-to-end testing is performed using Cypress to simulate real-user interactions.

**11. Known Issues**

Course enrollment status may not update instantly without manually refreshing the page after a new enrollment.

File upload (e.g., certificates or course materials) currently supports only PDF formats. Other file types like images or videos need to be added in future updates.

Admin actions such as course approval or user role updates may require manual triggering in certain deployment environments.

Dashboard tables and layout (especially for admin and teacher panels) have limited responsiveness on smaller mobile devices, which may affect usability.

**12. Future Enhancements**

Integrate live video sessions for interactive learning experiences using tools like Zoom or WebRTC.

Allow students to rate and review courses and instructors after completion to improve content quality and transparency.

Implement an AI-based course recommendation system based on student interests, progress, and skill level.

Sync course schedules and deadlines with external calendars like Google Calendar to help students stay organized.

Add multi-language support to make the platform accessible to users from diverse linguistic backgrounds.

Enable downloadable certificates and LinkedIn integration for showcasing course completion and achievements.

Introduce in-app support and chatbots for real-time doubt clarification and assistance.

Offer subscription-based premium content and membership plans with exclusive access.

Allow offline access to downloaded content (videos, PDFs) for uninterrupted learning.

Enable parental access or monitoring dashboards for student learning progress (for school-aged users).