“E-Learning – Online Learning Platform Using MERN”



NAAN MUDHALVAN PROJECT

NM1042 MERN STACK POWERED BY MONGO DB

**Submitted by**

# C GANESH KUMAR (412721205011)

# SM PRAVEEN (412721205033)

# C RAJAN (412721205036)

# DENNIS D SOUZA (412721205008)

# R KAMESH (412721205024)

DEPARTMENT OF INFORMATION TECHNOLOGY



**TAGORE ENGINEERING COLLEGE**

**RATHINAMANGALAM, CHENNAI 600 127**

**ANNA UNIVERSITY: CHENNAI 600 025**

**NOVEMBER 2024**

**BONAFIDE CERTIFICATE**

Certified that this project report **“**E-Learning – Online Learning Platform Using MERN**”** is the bonafide work of **Mr. C. Ganesh kumar, Mr. SM. Praveen, Mr. C. Rajan, Mr. Dennis D souza, Mr. R. Kamesh** of final year students of Information Technology, who carried out the Naan Mudhalvan project work in **NM 1042 MERN STACK POWERED BY MONGO DB** under my supervision.

**Signature of the Course coordinator Signature of the HOD**

**INTERNAL EXAMINER EXTERNAL EXAMINER**

## TABLE OF CONTENT

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO** | **TITLE** | **PAGE NO** |
| **1.** | **INTRODUCTION** | 3 |
|  |  |
| 2. | **Project Overview** | 3 |
| 3. | **Architecture** | 3 |
|  |  |
|  |  |
|  |  |
|  |  |
| 4. | **Setup Instructions** | 4 |
|  |  |
|  |  |
|  |  |
| 5. | **Folder Structure** | 4 |
| 6. | **Running the Application** | 4 |
| 7. | **Authentication Interface** | 5 |
| 8. | **Testing** | 6 |
| 9. | **Screenshots or Demo** | 7 |
| 10. | **Known Issues** | 12 |
| 11. | **Future Enhancements** | 12 |

**1.INTRODUCTION:**

 **Project Title**: E-Learning – Online Learning Platform Using MERN

 **Team Members**:

Ganesh kumar C

Praveen SM

Rajan C

Dennis D souza

Kamesh R

The online learning platform aims to transform the way people access education by delivering a seamless, dynamic, and user-friendly environment for students, instructors, and content creators. This platform, built on the MERN stack (MongoDB, Express.js, React.js, and Node.js), combines modern web technologies to provide a scalable and fast online learning solution. The platform has a number of features, including user registration and profile management, course browsing, enrollment, interactive content delivery, real-time progress tracking, and feedback mechanisms. Its sturdy architecture provides dependability, and its modular design makes it simple to maintain and expand.

### **2. Project Overview**

* **Purpose: The online learning platform, built with the MERN stack, seeks to offer users a seamless digital education experience. It provides course enrollment, progress tracking, and interactive content delivery. The platform is built for usability, scalability, and personalized learning. It connects instructors and learners with a user-friendly interface and extensive capabilities.**
* **Features**:
  + User Authentication and Authorization: Secure login and registration system for students, instructors, and administrators.
  + Course Enrollment and Management: Students can browse, enroll in courses, and track their progress. Instructors can create, update, and manage courses.
  + Interactive Content Delivery: Deliver lectures, quizzes, and assignments that engage students in learning.
  + Progress Tracking and Analytics: Allow students to track their learning progress, including completed lessons, quiz scores, and overall grades.
  + Discussion Forums or Community: Students and instructors can discuss course materials, share ideas, and collaborate.
  + Payment Integration: Students can pay for courses through a secure payment system.

### **3. Architecture**

* **Frontend(React.js):** Create a dynamic, user-friendly interface, User authentication pages (login, register),Dashboard to browse and enroll in courses, Course content viewer (videos, documents, quizzes),Progress tracking and badges, Discussion forums or live chat.
* **Backend (Node.js + Express.js):** Provide APIs for frontend and handle server-side logic,RESTful APIs for user management, courses, progress tracking, and messaging, Middleware for authentication (JWT or OAuth), File uploads for course materials.
* **Database** (MongoDB): Store user and platform data.
  + Users: Store user data (name, email, password, enrolled courses, etc.).
  + Courses: Store course details (title, instructor, modules, etc.).
  + Progress: Track user progress (courseID , userID, completed modules).
  + Messages: Handle forum or chat messages.
  + Assignments: Store assignments and submissions.

### **4. Setup Instructions**

* **Prerequisites**:
  + **Node.js** and **npm** (for backend and frontend development).
  + **MongoDB** (for data storage).
  + **Git** (for version control).
* **Installation**:
  + Clone the repository: git clone https://github.com/rajanc144/Online-Learning-platform-using-mern.git
  + Navigate to the server and client directories and install dependencies:
    - Backend: cd server -> npm install
    - Frontend: cd client -> npm install
  + Set up environment variables (.env files) for sensitive information like database URLs and API keys.

### **5. Folder Structure**

* **Client**: The React frontend includes components, services, styles, and utils for organizing the UI logic and styling.
  + **/src**
    - **/components**: Holds reusable UI components (e.g., CourseList.js, EnrolledCourses.js).
    - **/pages**: Contains main pages (e.g., navigation, logo, user menu).
    - **/services**: API service files for interacting with backend endpoints.
* **Server**: The Node.js backend includes routes, controllers, and models for handling HTTP requests and managing data.
  + **/src**
    - **/routes**: Contains API routes Endpoints for authentication (e.g., /login, /register).
    - **/controllers**: Houses functions handling the logic for requests.
    - **/models**: **MongoDB schemas** using Mongoose.

### **6. Running the Application**

* **Frontend**:
  + Navigate to the client directory: cd client
  + Start the frontend server: npm start
* **Backend**:
  + Navigate to the server directory: cd server
  + Start the backend server: npm start

### **7. API Documentation**

* **Endpoints**:
  + **User Management**:
    - POST /api/users/register: Register a new user.
    - POST /api/users/login: Authenticate a user.
  + Teacher:
    - GET /api/teachers: Retrieve a list of all teachers.
    - GET /api/teachers/:id: Retrieve details of a specific teacher.
  + **Student**:
    - POST /api/students: student login course
    - GET /api/students/:Id/enroll: ID of the student.

### **8. Authentication**

* **User Authentication**:
  + **JWT Tokens**: Secure authentication using JSON Web Tokens (JWT) for access control.
  + **Session Management**: User sessions are managed with token-based authentication to prevent unauthorized access.
* **Admin Authorization**:
  + Admins have access to specific routes and features, ensuring data security and restricted access to sensitive operations.

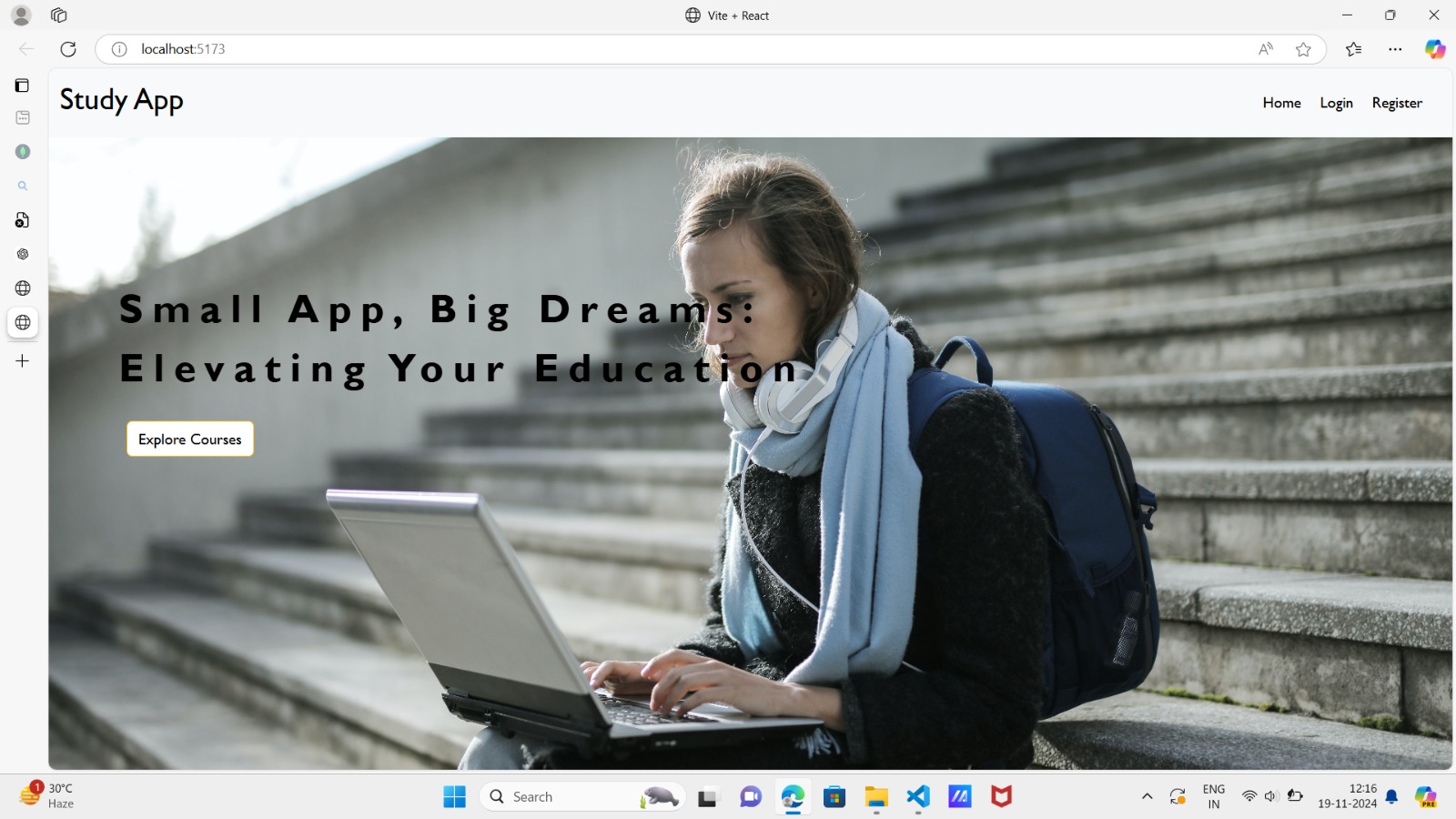
### **9. User Interface**

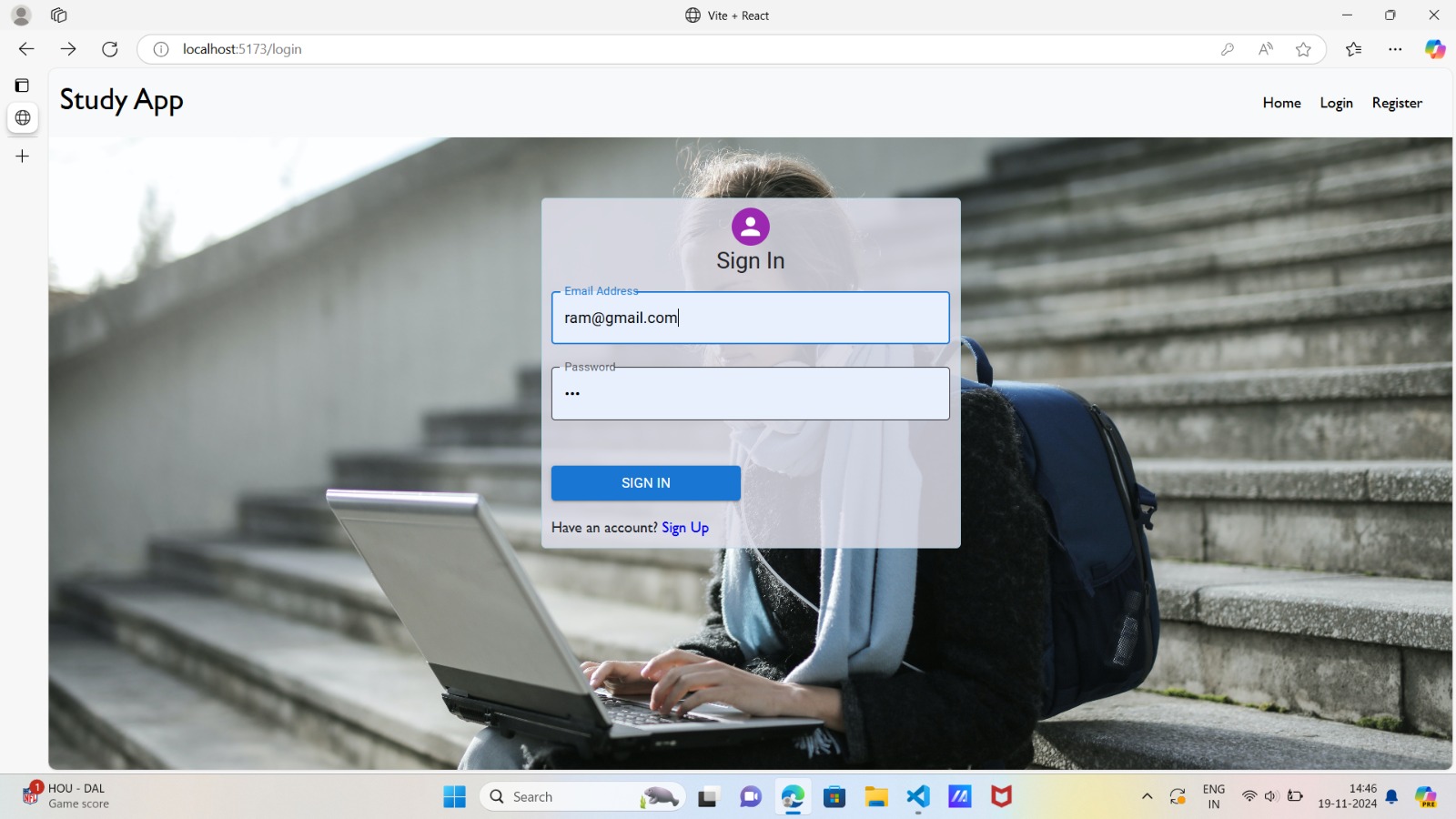
* **UI Elements**:
  + **Homepage**: Displays login and signup page.
  + Course Cards: Image, title, instructor name, price, and rating.
  + **Video Preview Section**: Short introduction video.
  + Course Progress Cards: Thumbnail, title, and progress bar.
  + Quizzes/Assignments: Timed or open-ended, Immediate feedback on answers.
  + Sign Up/Sign In Page: Email/password and social login options.
  + Reports: Analytics dashboard for performance tracking.
* **Screenshots**:
  + Include screenshots or GIFs of key UI features (e.g., browsing course, course lists).

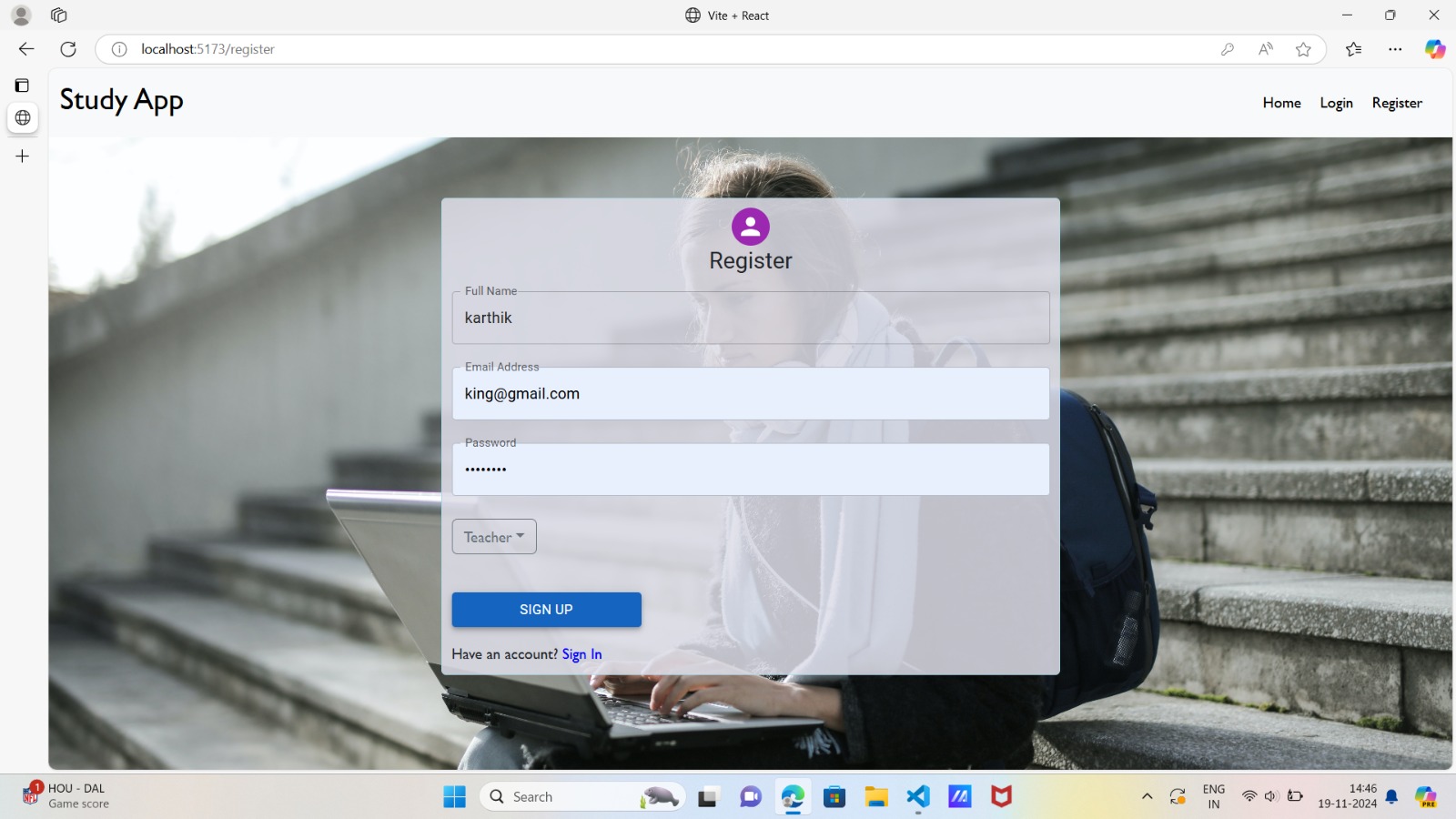
### **10. Testing**

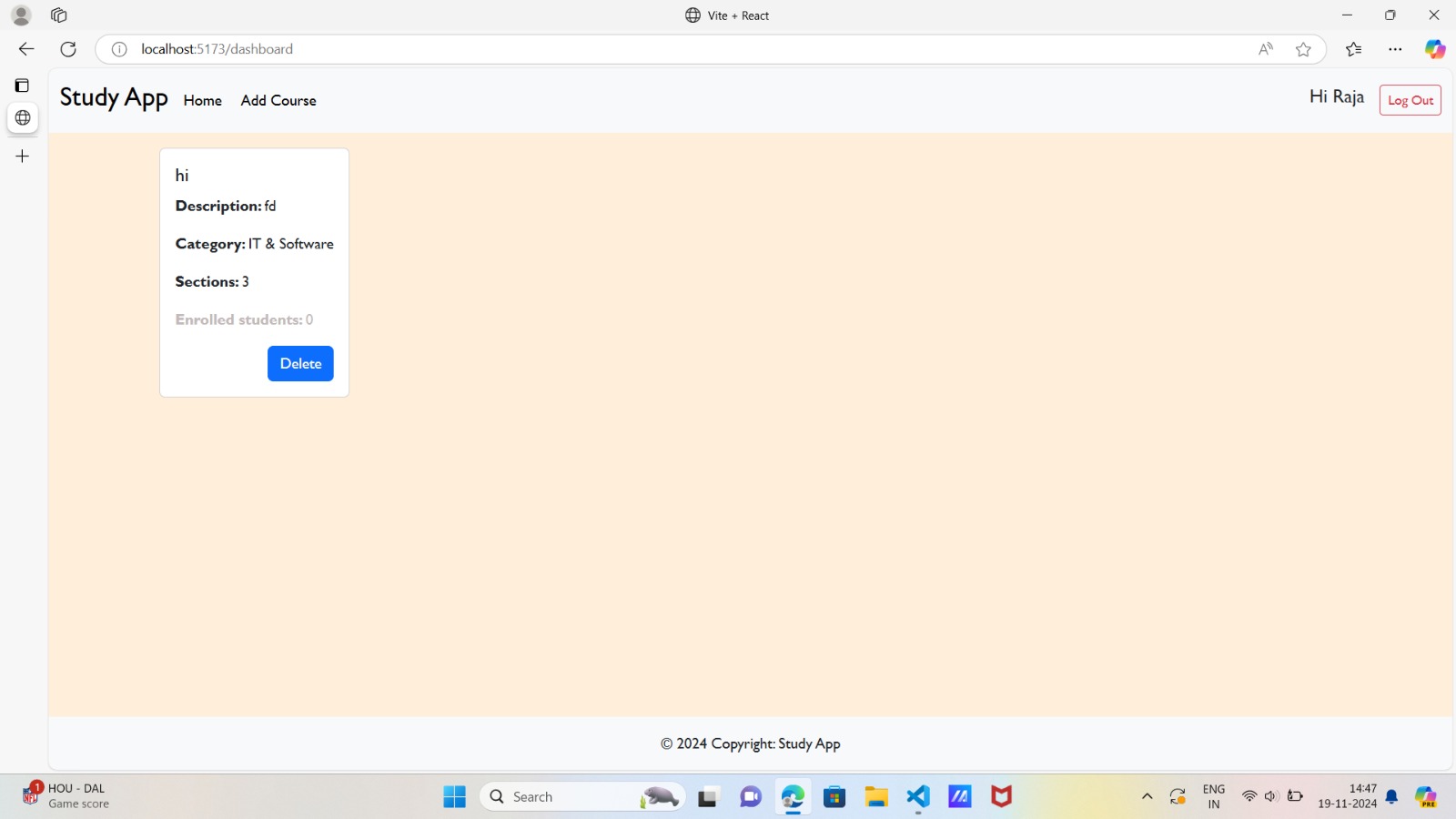
* **Testing Strategy**:
  + **Frontend Testing**: React Testing Library and test for unit testing components.
  + **Backend Testing**: Mocha and Chai for API endpoint testing.
  + **End-to-End Testing**: user interactions, covering critical workflows like user registration, course enrollment, content consumption, and progress tracking, ensuring functionality across all integrated components.

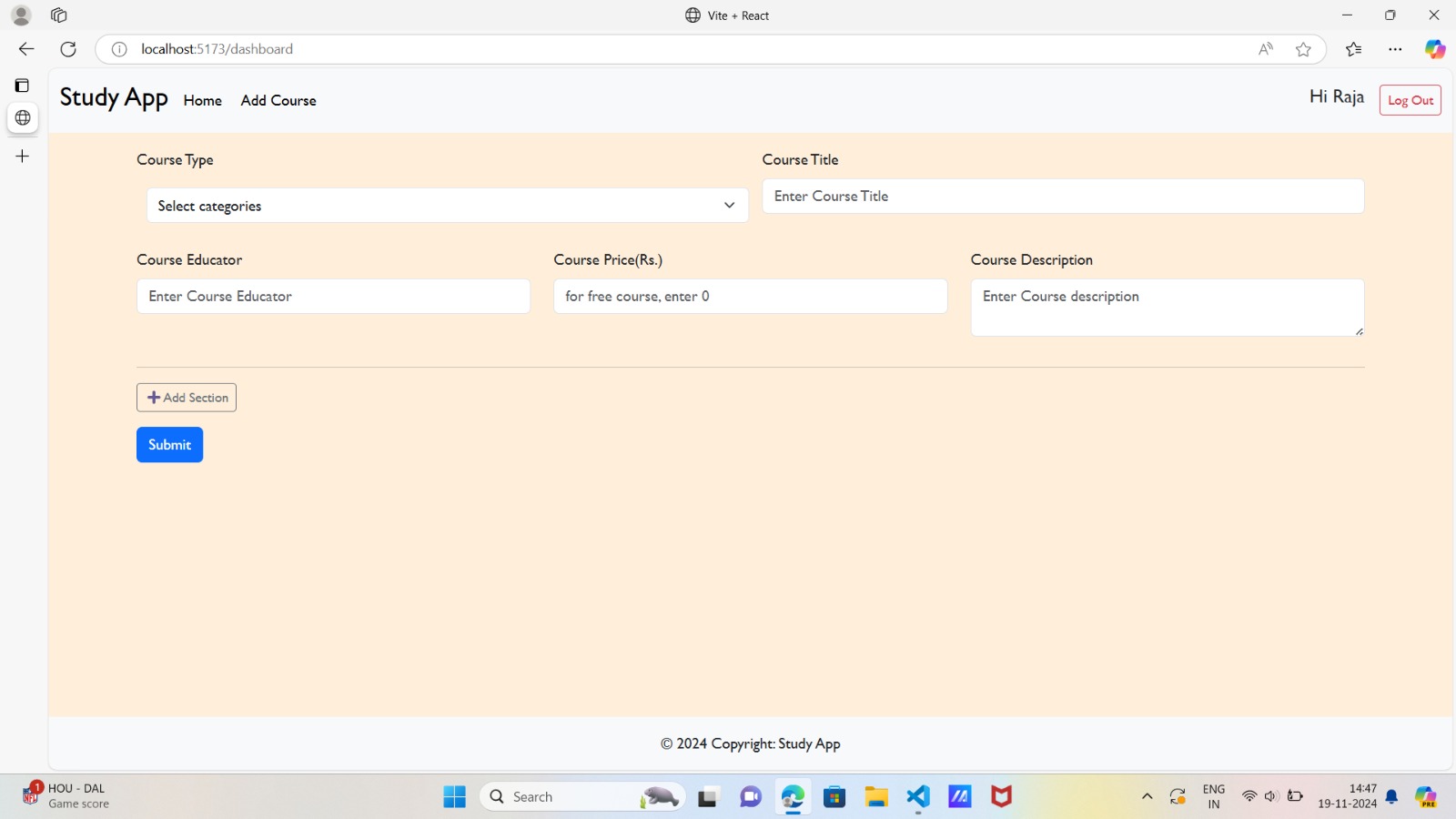
### **11. Screenshots or Demo**

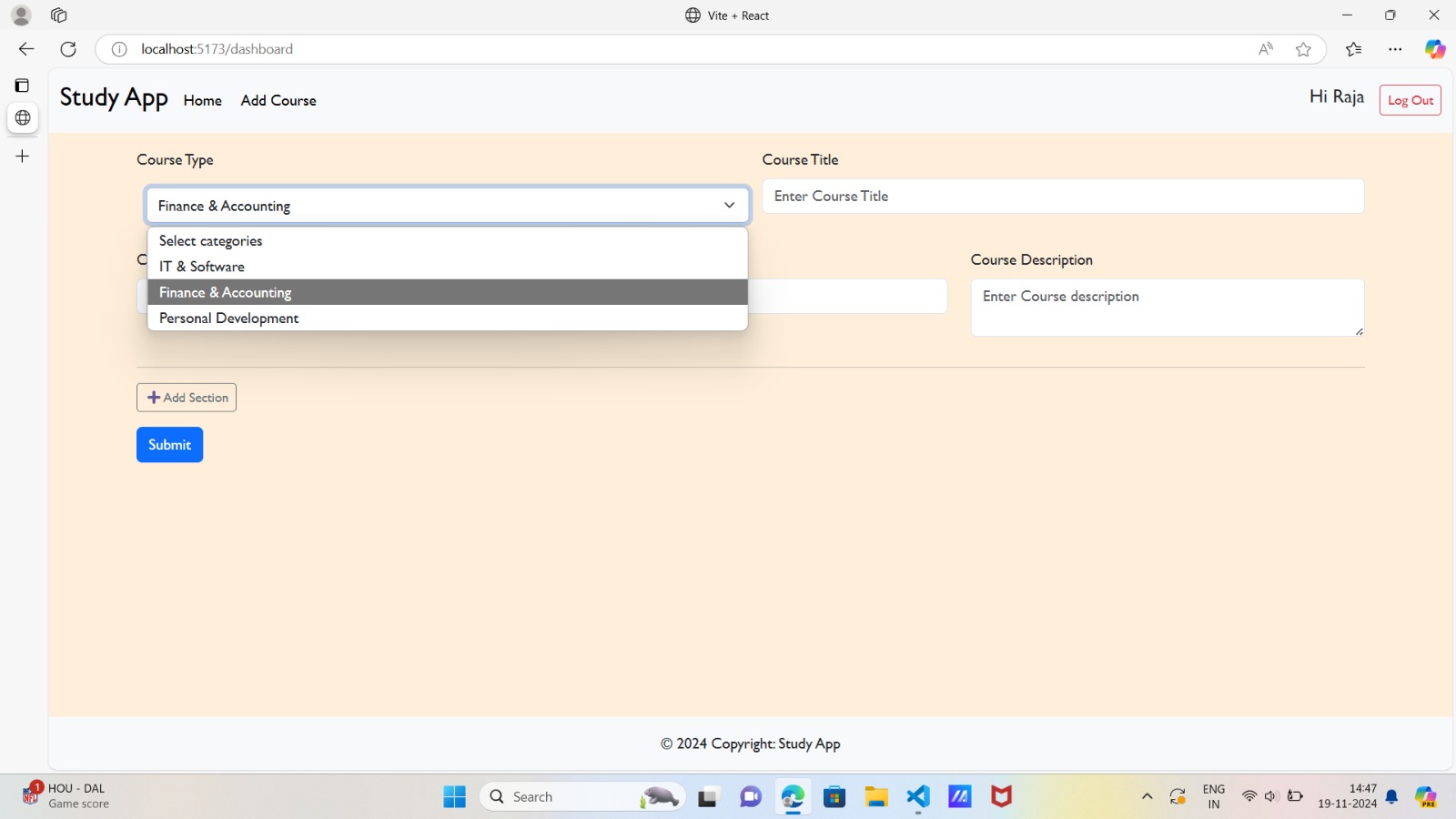


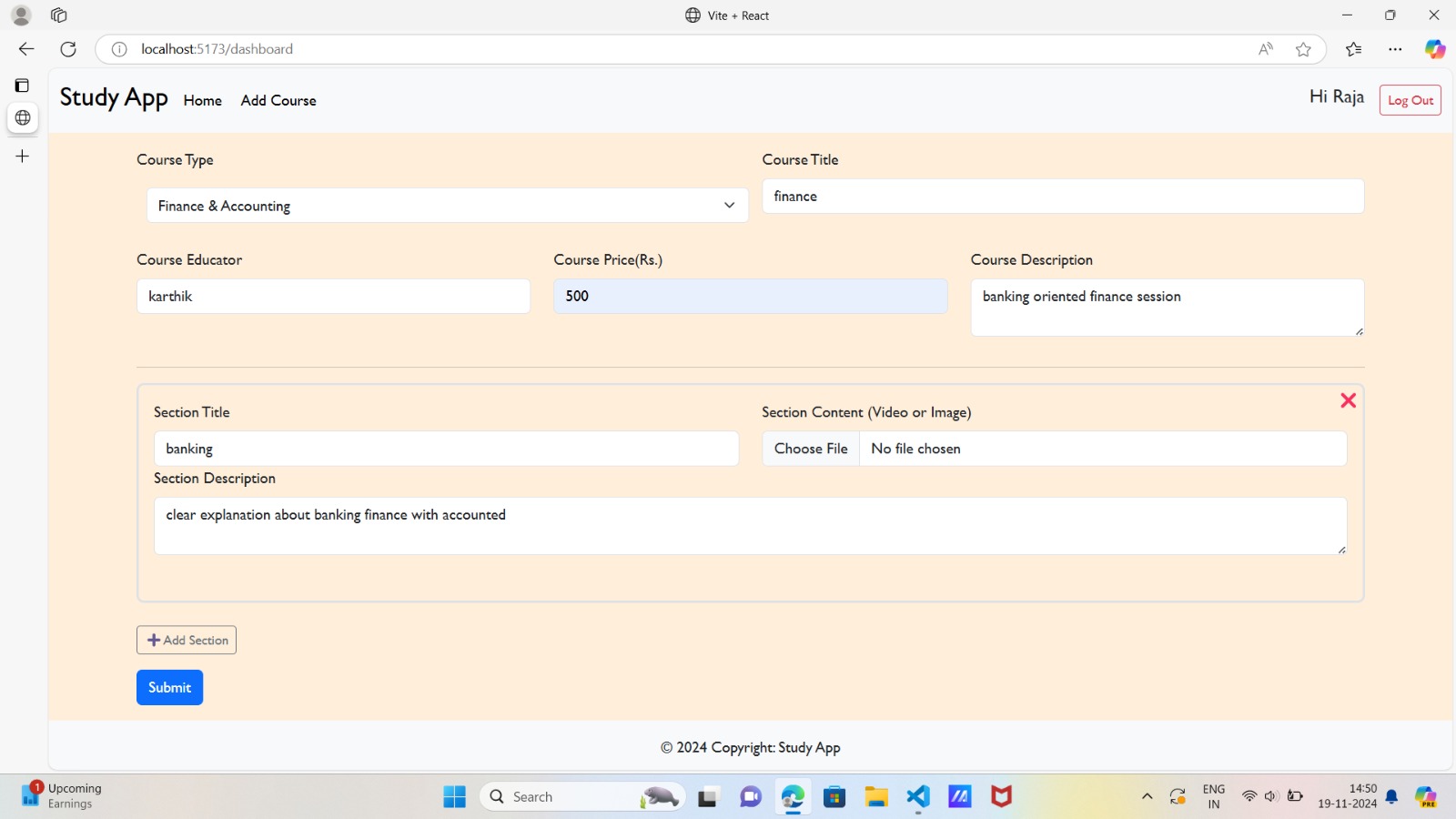


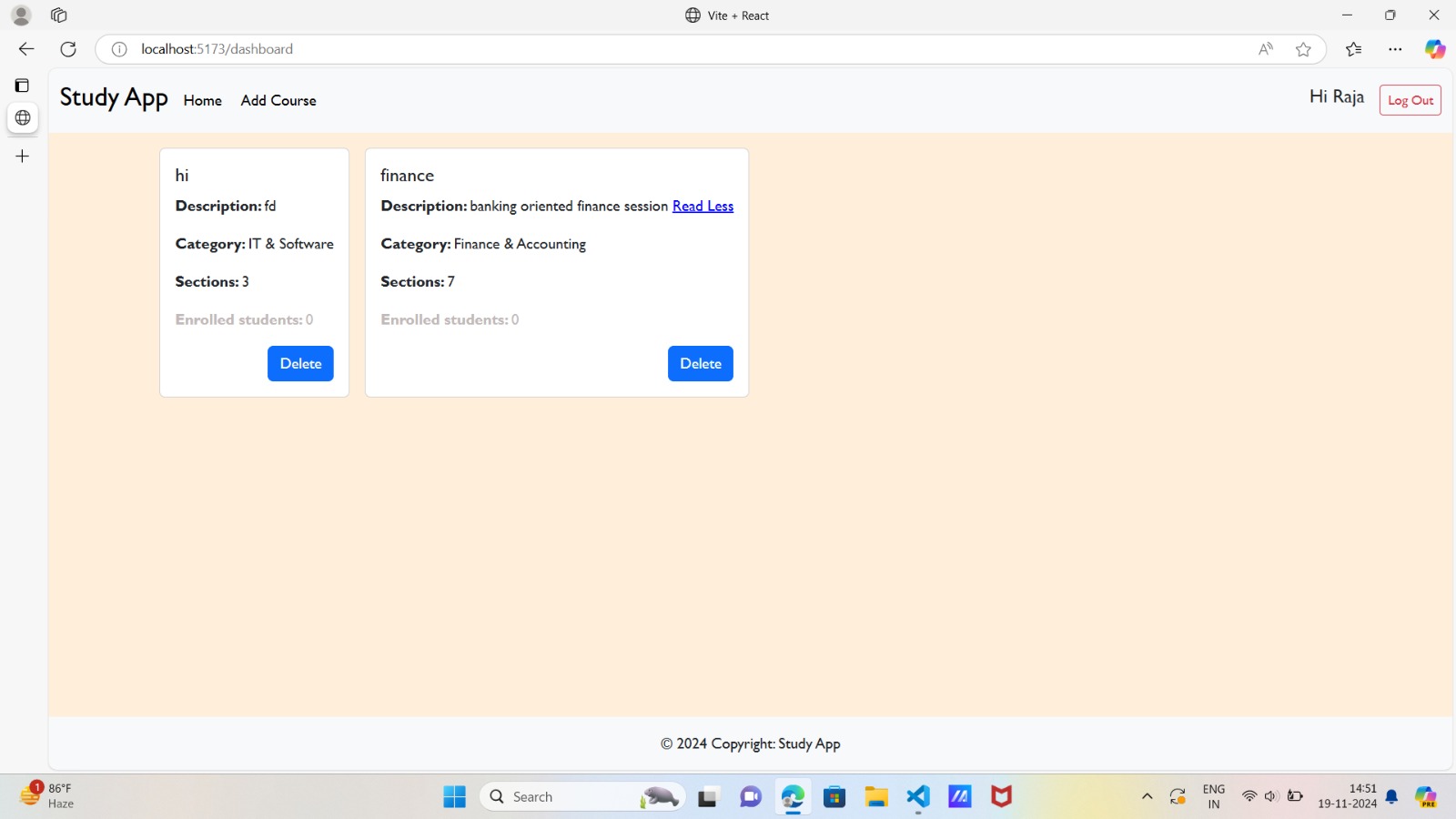


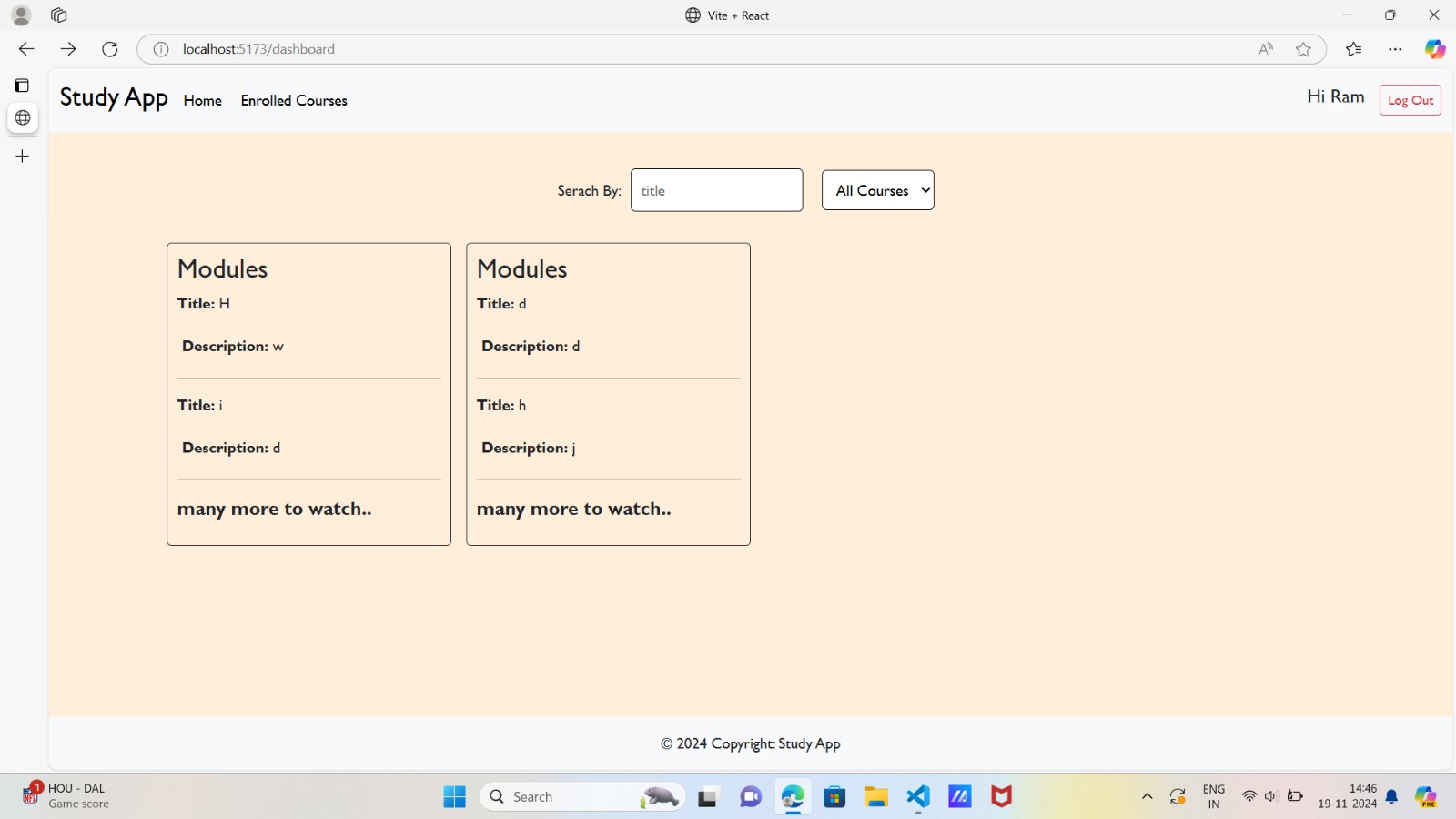


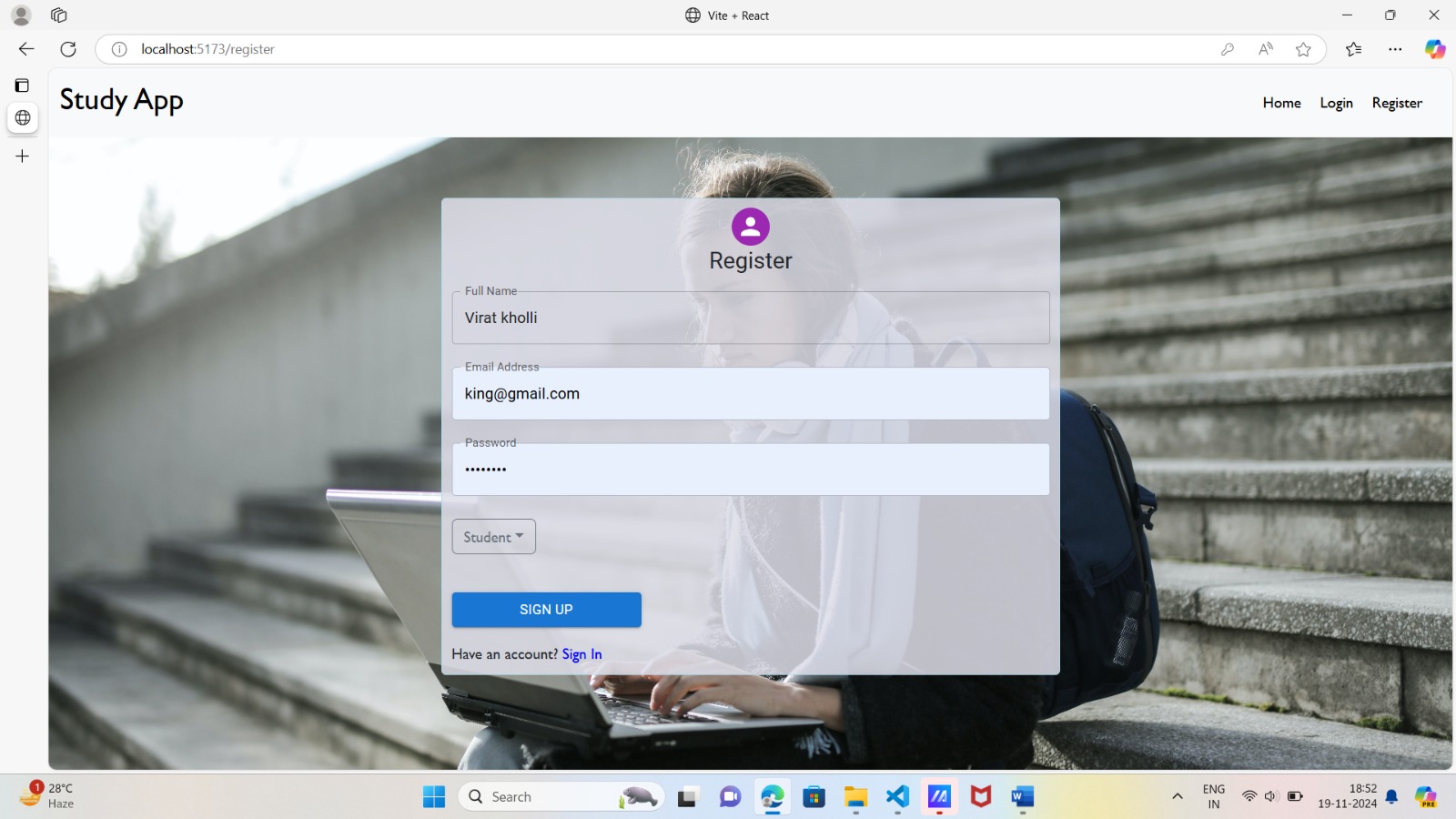


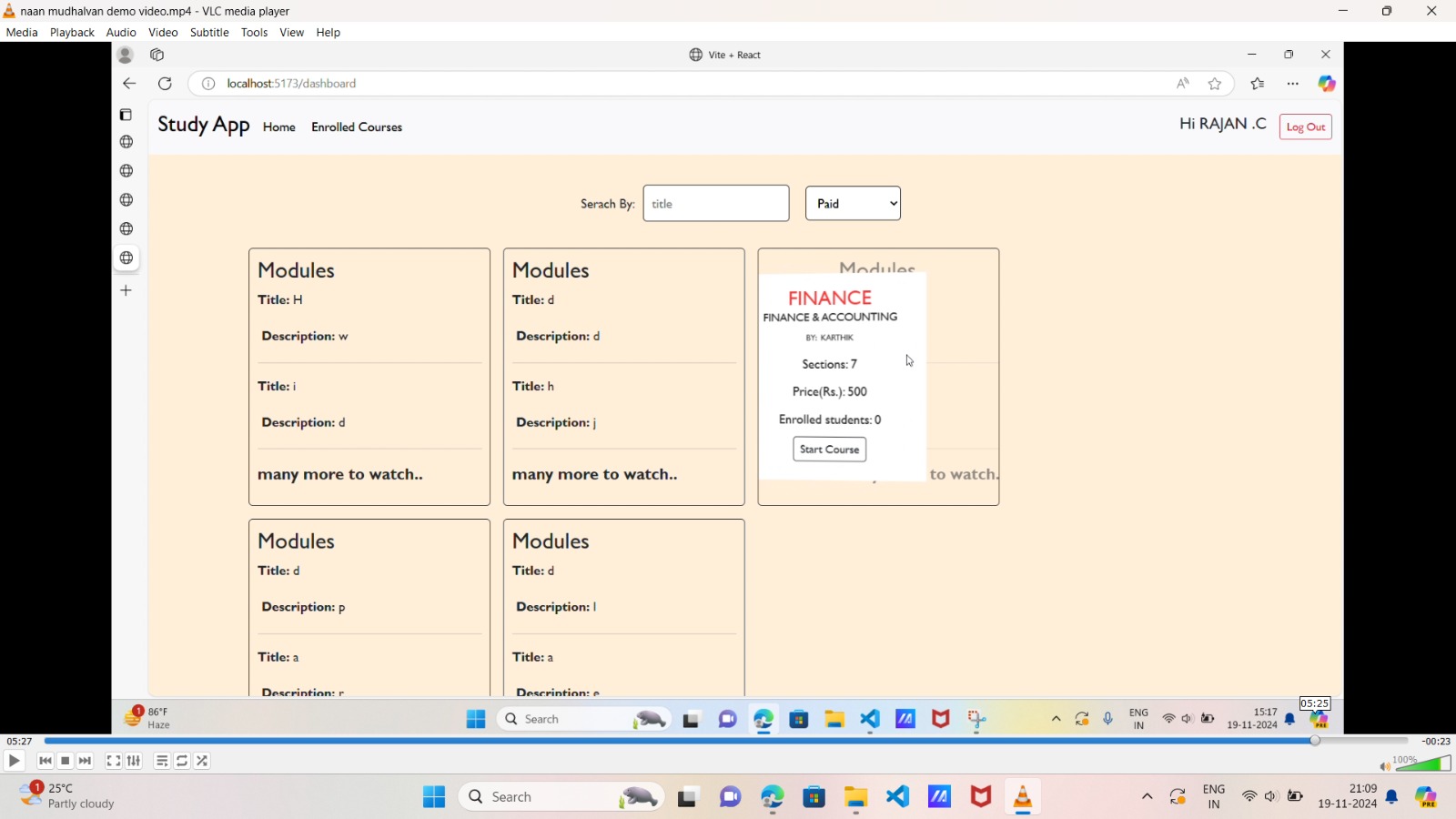












### **12. Known Issues**

* **Authentication and Authorization: Managing user authentication and role-based authorization (e.g., students, teachers, admins) can be complex.**
* Database Design: Structuring the database to handle diverse data (courses, users, grades, feedback) while maintaining scalability and performance.
* Real-Time Features (Live Classes, Chat, etc.): Implementing real-time communication for live sessions or chat between students and instructors.
* **Real-Time Notifications**: Occasionally, notifications may delay under heavy server load.
* **Responsive Design Tweaks**: Some minor design adjustments may be needed on specific screen sizes.

### **13. Future Enhancements**

* **Customizable Dashboards**: Allow users to personalize their learning dashboard with widgets that display upcoming lessons, recommended courses, or recently viewed content.
* **Live Classes & Webinars**: Implement live streaming functionality for instructors to conduct live sessions with real-time chat and Q&A features.
* **Timed Quizzes**: Implement quizzes with timers to simulate real exam conditions and provide instant feedback on performance.
* **Activity Reminders**: Send reminders about incomplete lessons, quizzes, or upcoming course events to keep students on track.
* **Voice Assistants**: Integrate AI-powered voice assistants that can guide users through the platform, answer questions, or help them find relevant courses.