LEARNHUB: YOUR CENTRE FOR SKILL ENHANCEMEN

[Document subtitle]

team id:LTVIP2025TMID45605

Online Learning Platform Using MERN Stack

1. Introduction

TheOnline Learning Platform (OLP) is a comprehensive digital solution created to provide flexible, accessible, and effective education through the internet. It leverages the MERN stack—MongoDB, Express.js, React.js, and Node.js—to build an interactive and responsive learning system that caters to students, teachers, and administrators.

Online learning platforms have become increasingly significant, especially post-pandemic, offering the ability to learn anytime, anywhere. The OLP application simplifies the process of accessing educational content and ensures inclusive learning for users with different technical backgrounds.

2. Key Features of OLP

- User-FriendlyInterface:The platform has an intuitive interface that supports easy navigation, even for users with limited technical skills.
- Efficient Course Management: Teachers can upload, structure, and update course materials. Students can browse courses, enroll, and keep track of their progress.
- Interactive Learning: The platform includes features like chat rooms, discussion forums, and live webinars to enhance student-teacher collaboration.
- Certification: Learners receive certificates upon completing their courses, which can be useful for career advancement or academic purposes.
- Cross-Device Accessibility: The platform is accessible on desktops, tablets, and smartphones.
- Self-Paced Learning: Users can complete course materials according to their own schedules.
- Flexible Pricing: Some courses are available for free, while others require payment or a subscription.

3. Case Study: Swathi Learning Journey

Scenario: Swathi,a college student, is eager to learnweb development. She signs up on the OLP, using her email and a secure password.

1. Course Discovery: Upon logging in, Swathi is welcomed by a well-organized course catalog. She filters the courses and selects "Web Development Fundamentals."
2. Enrollment: After reading the course overview and instructor details, Sarah enrolls in the course.

3. Progress Tracking: Swathi studies the modules at her convenience. The platform automatically saves her progress.
4. Interaction: She attends live webinars and participates in forums to clarify doubts and engage with peers.
5. Certification: After completing all lessons and passing the final exam, she receives a certificate.
6. Advanced Learning: Swathi later purchases a premium course to continue learning advanced topics.

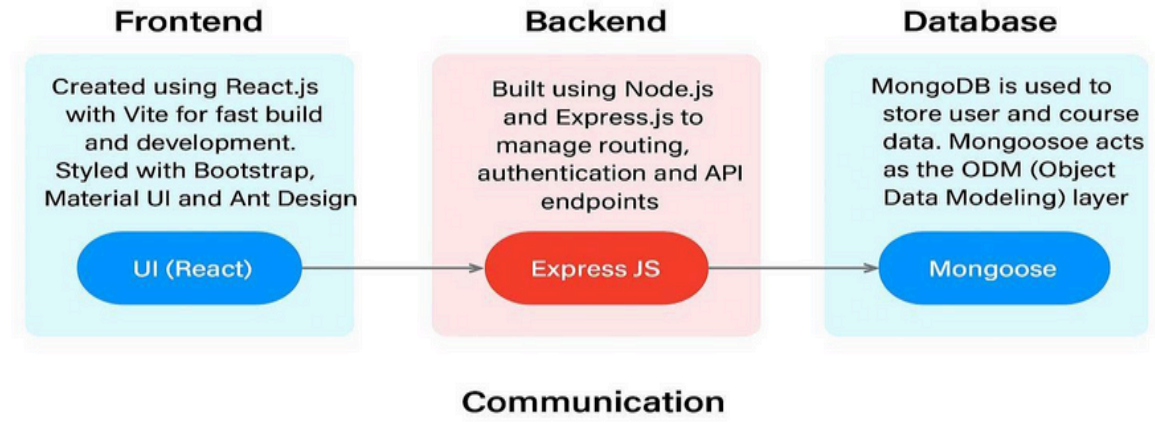
Teacher's Role:Tejaswi , a web development instructor, uploads new course content, edits existing modules, and monitors enrollments.

Admin's Role: The administrator oversees platform operations, manages user accounts, and ensures that all systems run smoothly.

4. Technical Architecture of the Platform

The OLP follows a client-server architecture

- Frontend: Created using React.js with Vite for fast build and development. Styled with Bootstrap, Material UI, and Ant Design.
- Backend: Built using Node.js and Express.js to manage routing, authentication, and API endpoints.
- Database: MongoDB is used to store user and course data. Mongoose acts as the ODM (Object Data Modeling) layer.
- Communication: The frontend and backend communicate via Axios through RESTful APIs.



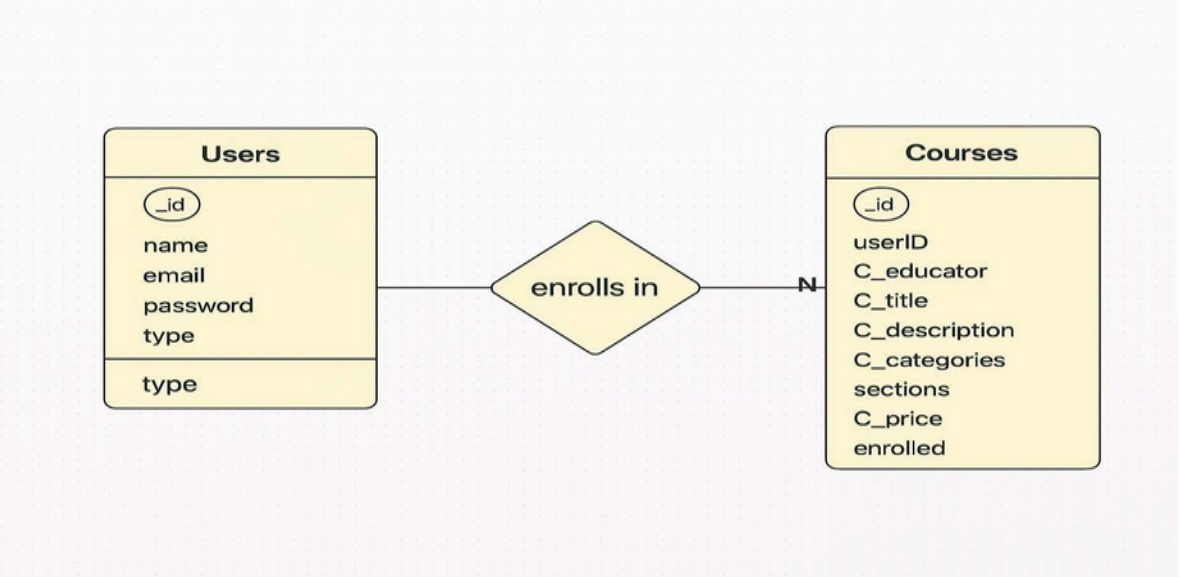
5. ER Diagram - Database Schema

Users Collection:

- _id: Unique identifier (auto-generated by MongoDB)
- name: Name of the user
- email: Email ID
- password: Encrypted password
- type: Role of user (student, teacher, admin)

Courses Collection:

- _id: Unique identifier
- userID: Foreign key referring to the instructor
- C_educator: Instructor's name
- C_title: Title of the course
- C_description: Detailed overview
- C_categories: Course category
- sections: List of lessons/modules
- C_price: Price of the course
- enrolled: Array of enrolled student IDs



6. Pre-Requisites

To run this project, the following tools and skills are required

- Vite: Frontend bundler for React apps. Command: npm create vite@latest
- Node.js & npm: Server-side runtime. Download from nodejs.org
- Express.js: Backend web application framework. Install using npm install express

- MongoDB: NoSQLdatabase. Download from mongodb.com
- Mongoose: ODMfor MongoDB. Install using npm install mongoose
- React.js: Frontendlibrary. Follow guide at React Docs
- Additional Tools: cors, dotenv, bcryptjs, jsonwebtoken, multer, nodemon, axios

7. Setup Instructions

1. Clone the Repository: Navigate to your local folder.
2. Install Dependencies:

   cd frontend

   npm install

   cd ../backend

   npm install
3. Start Development Server:

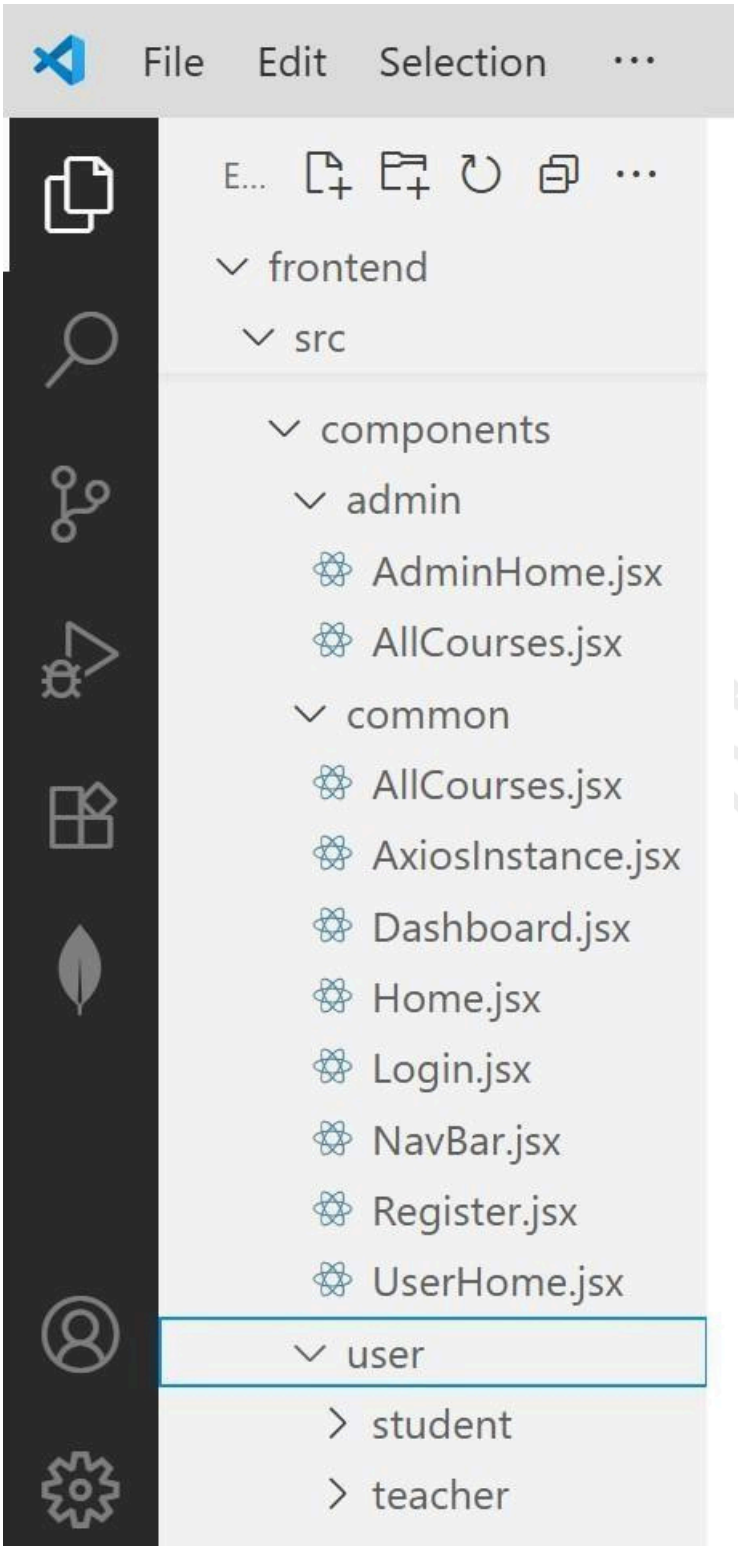
   npm start
4. Access Application: Open browser and visit http://localhost:5173

8. Project Structure

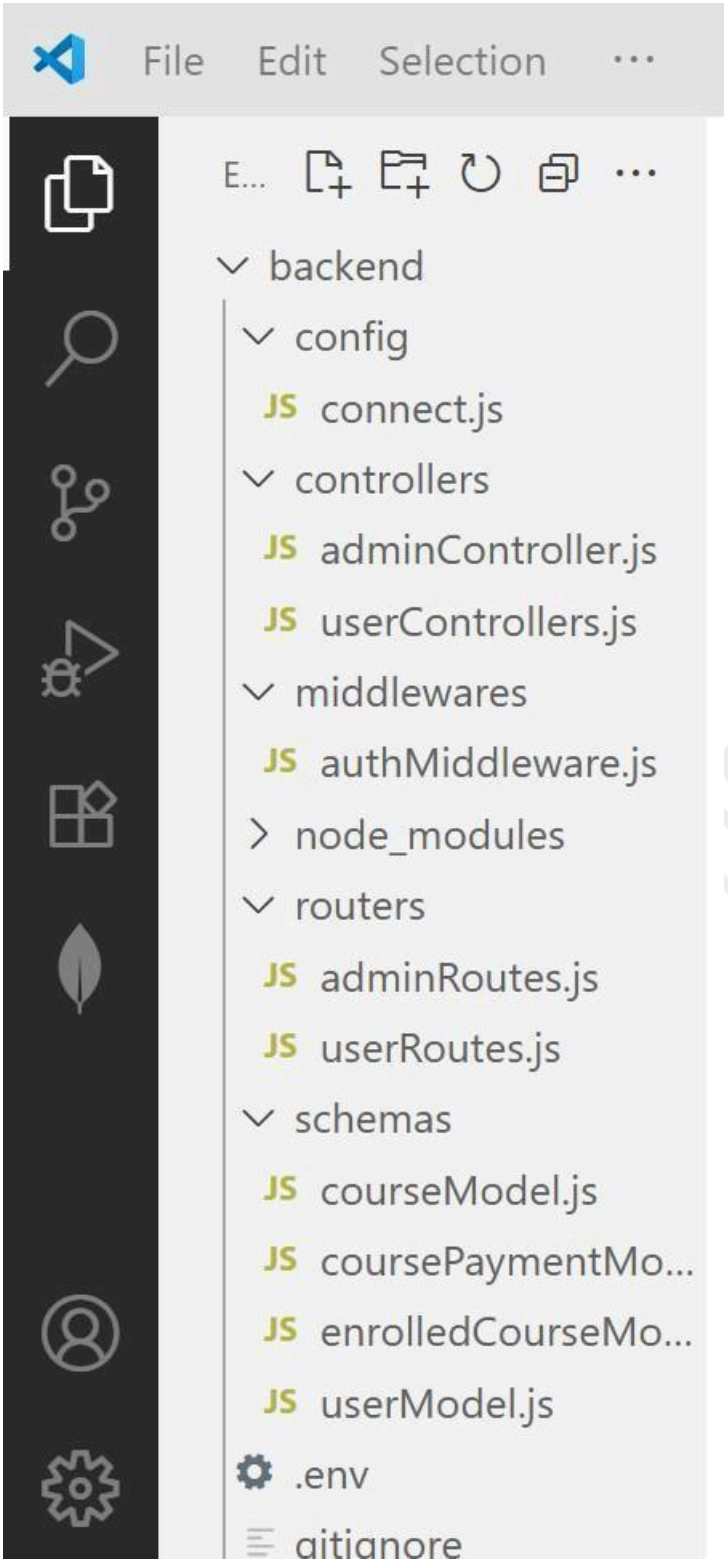
The first image is of the front part which shows all the files and folders that have been used in UI development

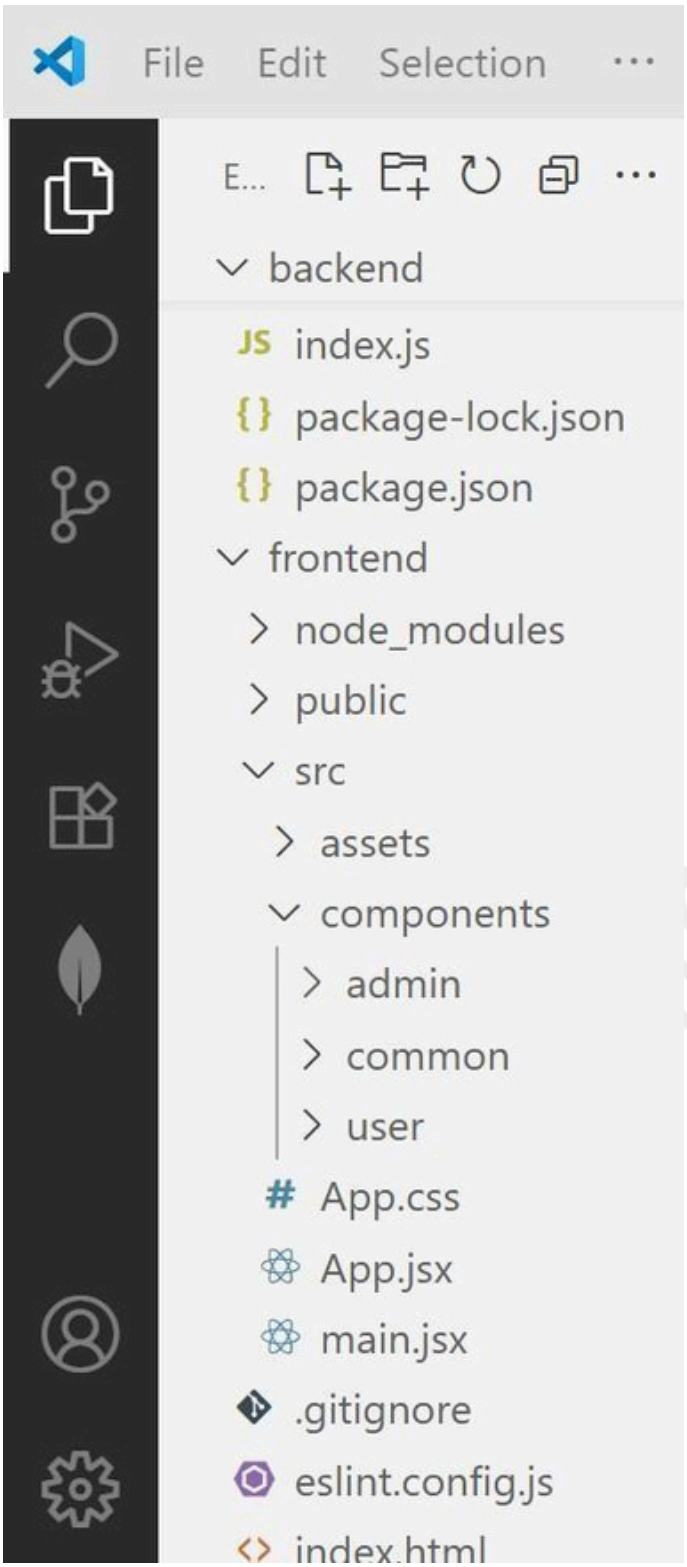
The remaining image is of the Backend part which shows all the files and folders that have been used in the backend development

Frontend Structure



Backend structure





9. Application Flow: User Roles and Activities

Teacher:

- Create, manage, and delete courses.
- Upload video lectures and materials.
- View enrolled student list.

Student:

- Register, log in, and enroll in multiple courses.
- Resume course from where they left off.
- Download certificates after course completion.
- Purchase paid courses securely.
- Search or filter courses by title and category.

Admin:

- Monitor all users and content.
- Modify or delete any course.
- Keep records of student enrollments.

10. Milestone Breakdown

Milestone 1: Initial Setup

- Folder structure creation for frontend and backend.
- Installed backend packages: cors, express, dotenv, mongoose, bcryptjs, multer, nodemon, jsonwebtoken

```
{  "name": "backend",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
   ▷Debug
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "nodemon index" },
  "dependencies": {
    "bcryptjs": "^2.4.3",
    "cors": "^2.8.5",
    "dotenv": "^16.3.1",
    "express": "^4.18.2",
    "jsonwebtoken": "^9.0.2",
    "mongoose": "^7.5.2",
    "multer": "^1.4.5-lts.1",
    "nodemon": "^3.0.1" },
  "keywords": [],
  "author": "",
  "license": "ISC"}
```

Milestone 2: Backend Development

- Created index.js, added CORS, body-parser
- Connected to MongoDB using environment variables
- Created authMiddleware.js for JWT authentication

Milestone 3: Database Configuration

- Defined models in the models/ directory
- Configured database connection in config.js

Milestone 4: Frontend Development

- Setup using React + Vite
- Installed UI libraries (Material UI, Bootstrap)
- Integrated frontend with backend via Axios

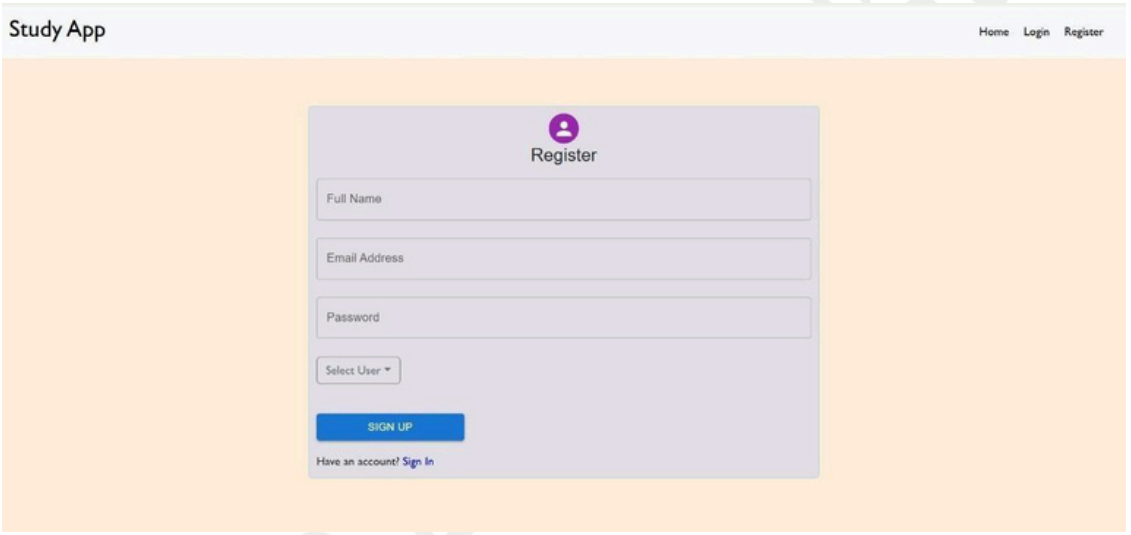
```
{  "name": "frontend",
  "private": true,
  "version": "0.0.0",
  "type": "module",
    ▷Debug
  "scripts": {
    "dev": "vite",
    "build": "vite build",
    "lint": "eslint .",
    "preview": "vite preview" },
  "dependencies": {
    "html2canvas": "^1.4.1",
    "jspdf": "^3.0.1",
    "mdb-react-ui-kit": "^9.0.0",
    "react": "^19.1.0",
    "react-bootstrap": "^2.10.10",
    "react-dom": "^19.1.0",
    "react-player": "^2.16.0"  },
  "devDependencies": {
    "@eslint/js": "^9.25.0",
    "@types/react": "^19.1.2",
```

Milestone 5: Implementation and Testing

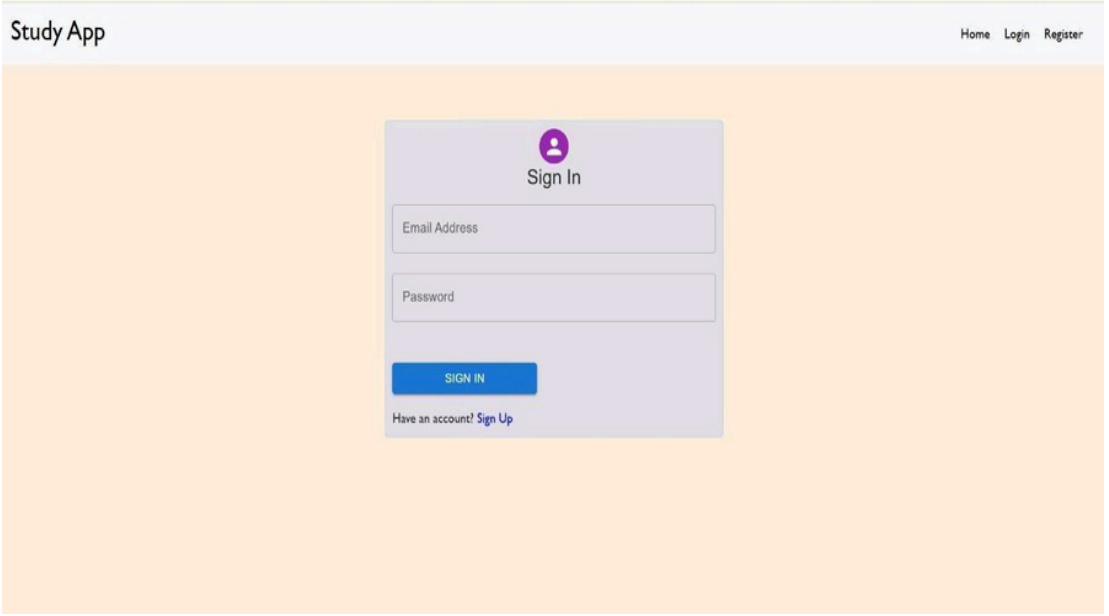
- Validated all dashboards (Admin, Teacher, Student)
- Final testing to verify features like registration, login, enrollments, and certificate

Home Page

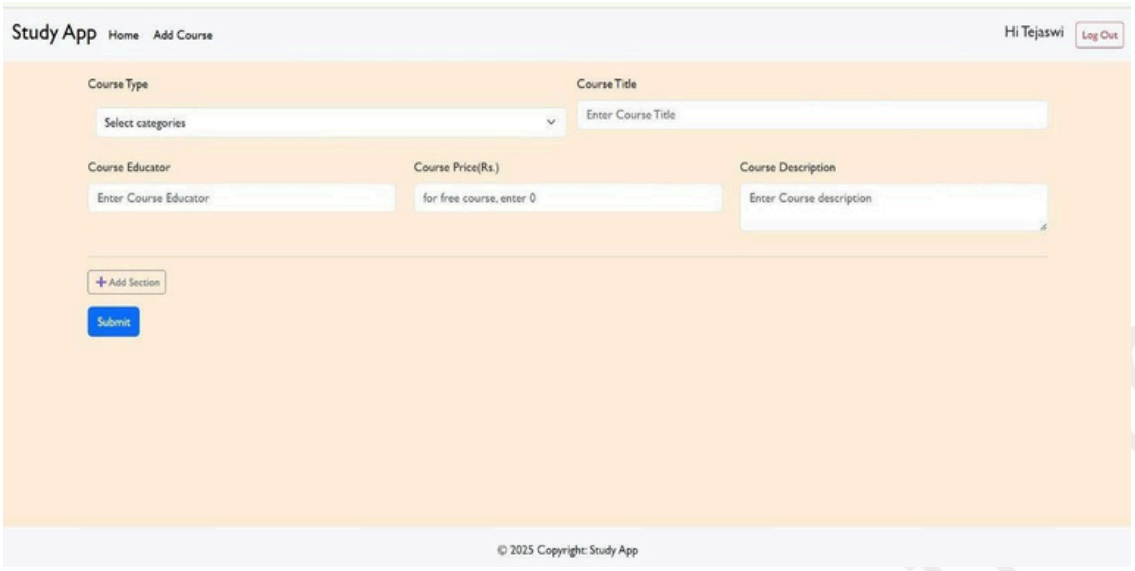
Register Page



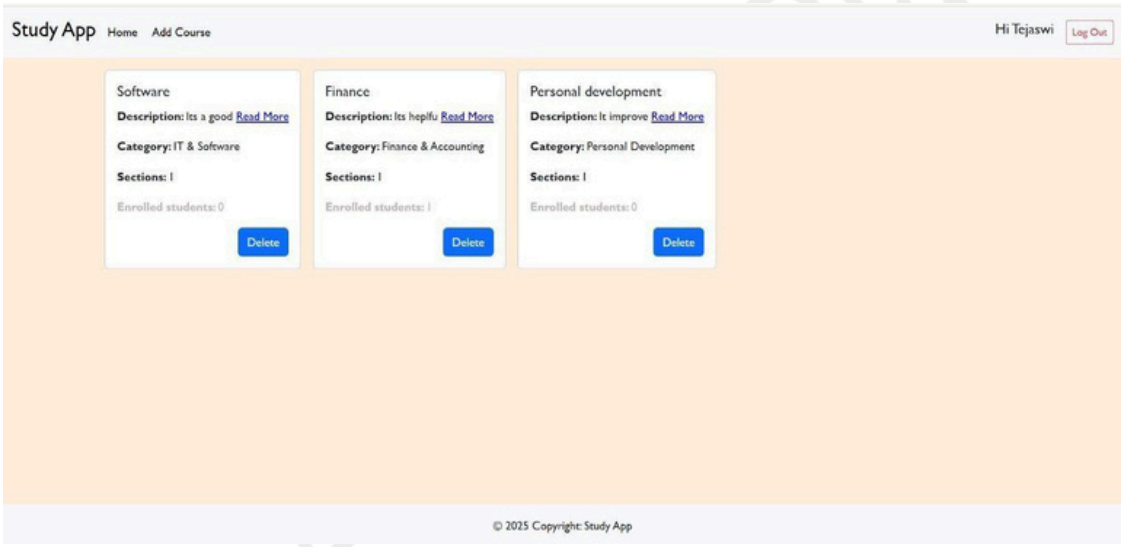
Login Page



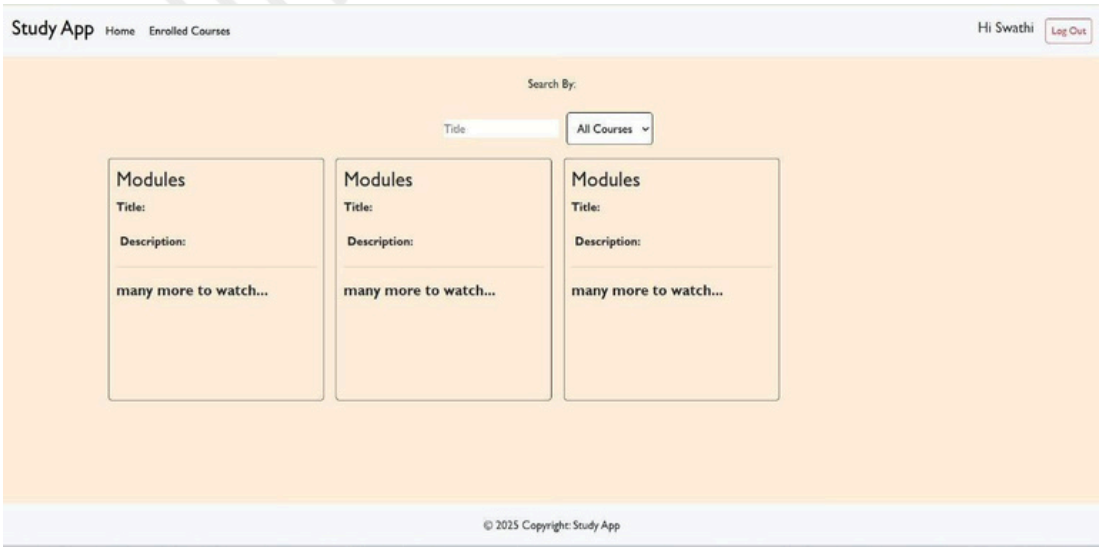
Add Course Page



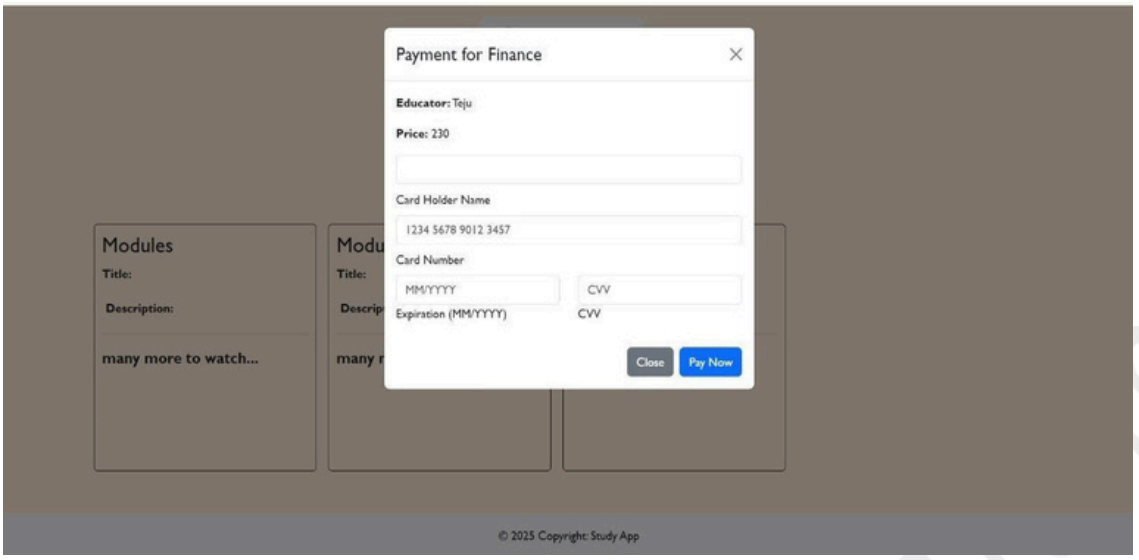
Added Courses Into Teacher Dashboard



Student Dashboard

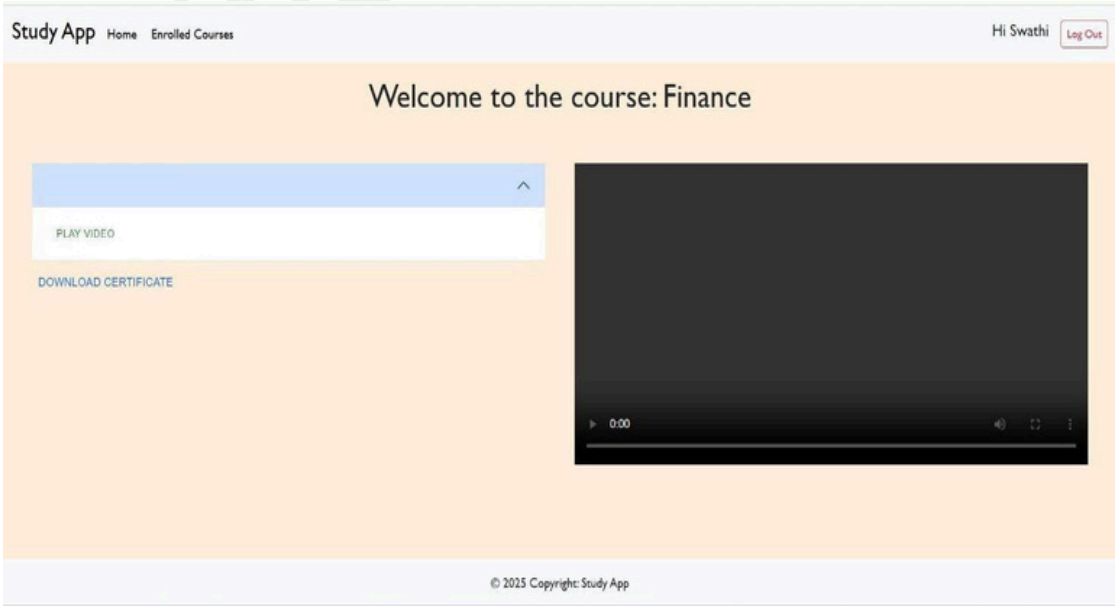


Course Payment



Go Enroll Course Page

Weclome to Enrolled Course Page



Completion Certificate

11. Future Enhancements

- PaymentintegrationwithRazorpay or Stripe
- Add real-time chat for students and instructors
- In-course quizzes and graded assessments
- AI-based personalized course suggestions
- Mobile app version using React Native

12. Conclusion

The OnlineLearning Platform demonstrates how full-stack web technologies can be used to create a responsive and scalable education portal. With clearly defined roles, modular design, and a dynamic interface, the application can be extended with new features like real-time interaction, secure payments, and mobile compatibility. This project serves as a strong foundation for future innovations in digital learning.