**Project Documentation for Online Learning Platform**

**1. Introduction**

**Project Title**: Online Learning Platform

**Team Members:**

* + Developer 1: Abilash R
  + Developer 2: Akash S
  + Developer 3: Ashwin R
  + Developer 4: Logesh K

**2. Project Overview**

* **Purpose:**

The purpose of this online learning platform is to provide a user-friendly interface for teachers to offer courses, for students to enroll, track, and complete these courses, and for an admin to manage overall activities within the platform.

* **Features:**
* Teachers can create, manage, and remove courses and add specific sections.
* Students can enroll in courses, resume from where they left off, complete courses, and receive certificates.
* Admins can monitor users, manage course listings, and maintain records of enrolled students.

**3. Architecture**

* Frontend:

The frontend is built using React.js, providing a responsive and dynamic user interface. React components manage the different pages and functions, allowing users to browse, filter, and view courses.

* Backend:

Node.js and Express.js power the backend, handling user requests, authentication, course management, and other business logic. The backend follows a RESTful API approach to communicate with the frontend.

* Database:

MongoDB is used as the database, storing user information, course details, sections, enrollments, and completed course records. Collections include users, courses, sections, and enrollment records.

**4. Setup Instructions**

* Prerequisites:
* [Node.js](https://nodejs.org/): v14+
* [MongoDB](https://www.mongodb.com/): Community Edition or MongoDB Atlas (for cloud setup)
* npm or yarn as a package manager
* Installation:

1. Clone the repository: `git clone [repository link]`

2. Navigate to the project folder.

3. Install dependencies:

* For the backend: `cd server && npm install`
* For the frontend: `cd client && npm install`

**4. Configure environment variables:**

* Create a `.env` file in the server directory with required keys (e.g., database URI, JWT secret).

**5. Folder Structure**

* Client:
* `src/components`: Contains reusable UI components.
* `src/pages`: Houses different pages (Home, Course Detail, Dashboard).
* `src/services`: Manages API calls and handles communication with the backend.
* Server:
* `controllers/`: Holds logic for handling various API requests.
* `models/`: Contains Mongoose schemas and models for MongoDB collections.
* `routes/`: Defines all API routes.
* `middlewares/`: Houses middleware for authentication and error handling.

**6. Running the Application**

* Frontend:
* Navigate to the client directory and run `npm start` to start the React application.
* Backend:
* Navigate to the server directory and run `npm start` to start the Express.js server.

**7. API Documentation**

* Endpoints:

1. Teacher Routes:

* `POST /api/teacher/course`: Add a course.
* `DELETE /api/teacher/course/:id`: Delete a course if no students are enrolled.
* `POST /api/teacher/course/:id/section`: Add a section to a course.

1. Student Routes:

* `POST /api/student/enroll/:courseId`: Enroll in a course. `POST /api/student/course/:id/progress`: Update course progress.
* `GET /api/student/course/:id/certificate`: Download certificate after course completion.
* `GET /api/student/courses`: Filter courses by name, category, or other criteria.

3. Admin Routes:

* `GET /api/admin/users`: View all users.
* `GET /api/admin/courses`: View all courses.
* `POST /api/admin/course/:id/update`: Update course details.

**8. Authentication**

* Authentication Mechanism:

Authentication is managed using JSON Web Tokens (JWT). When a user logs in, a JWT is generated and sent to the client to be stored locally (typically in cookies or localStorage).

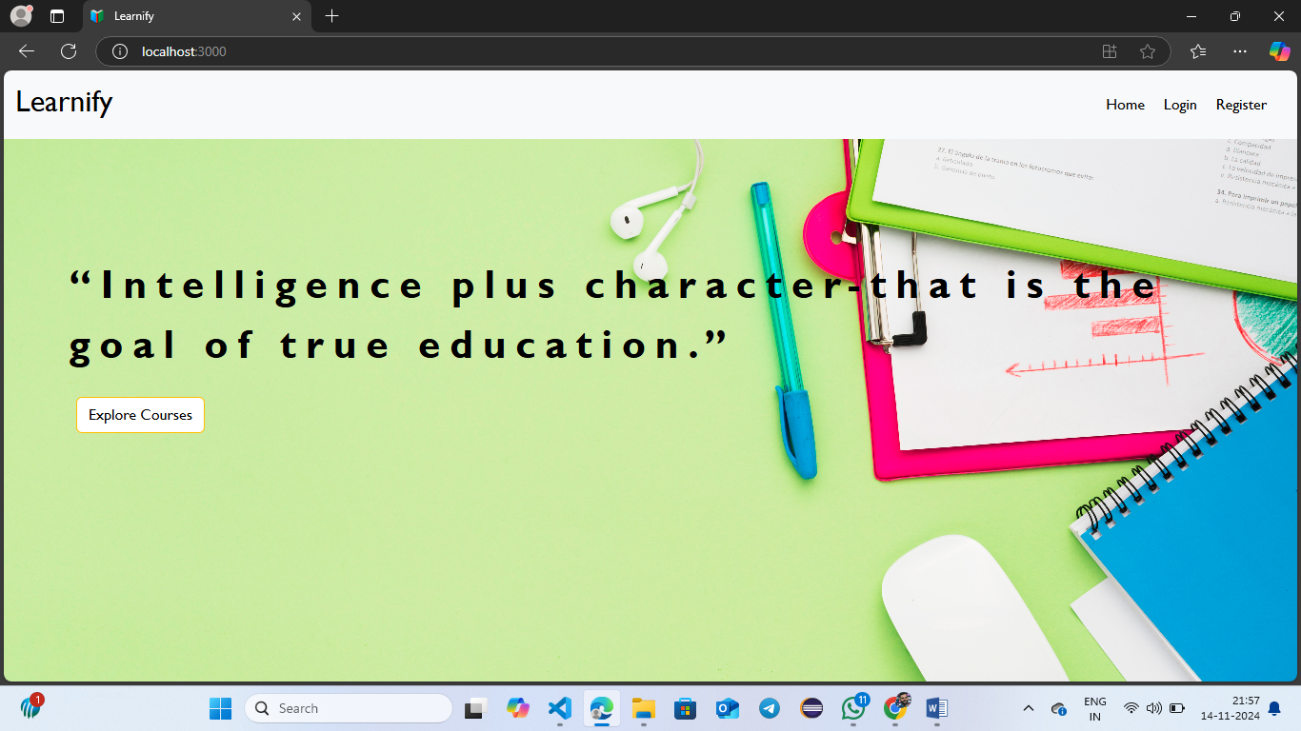
Authorization:

Roles are assigned as `teacher`, `student`, or `admin`, with each role having specific permissions enforced via middleware.

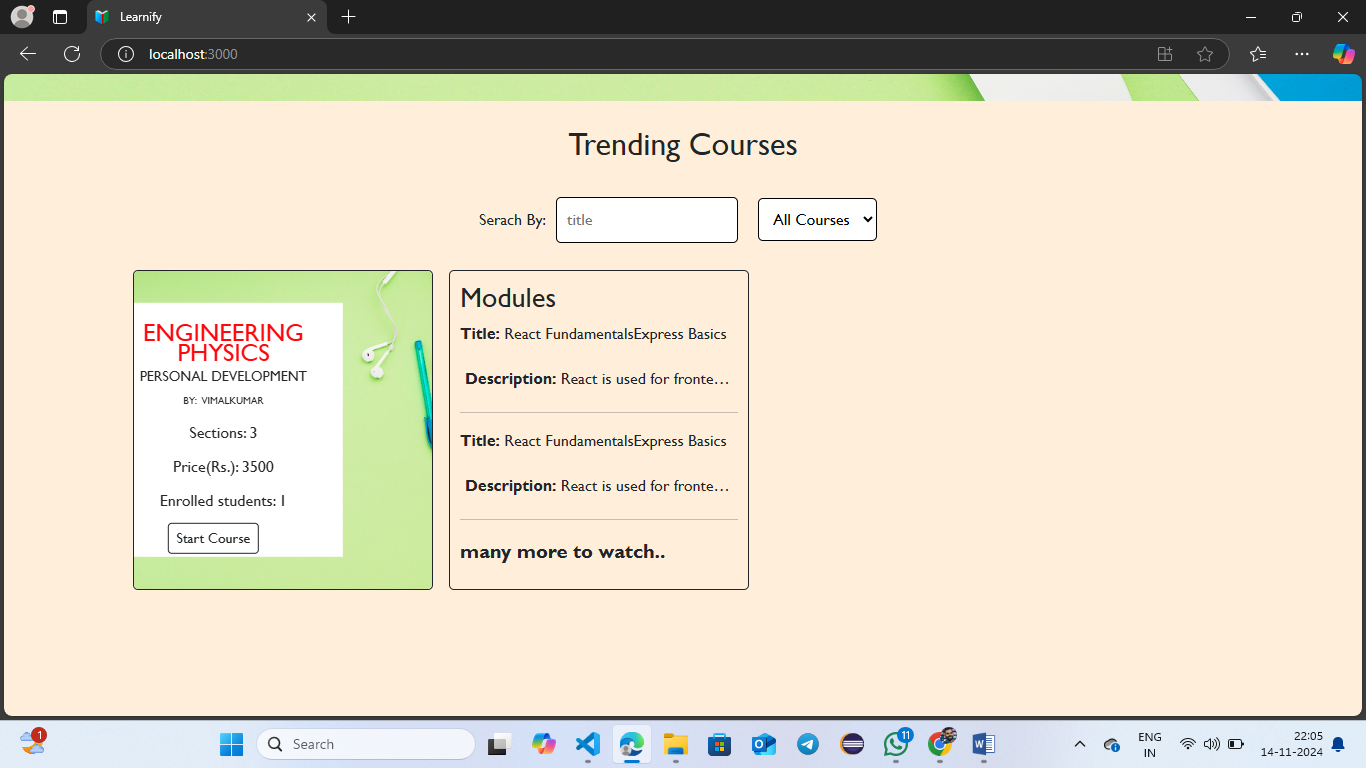
**9. User Interface**

* **Screenshots**:

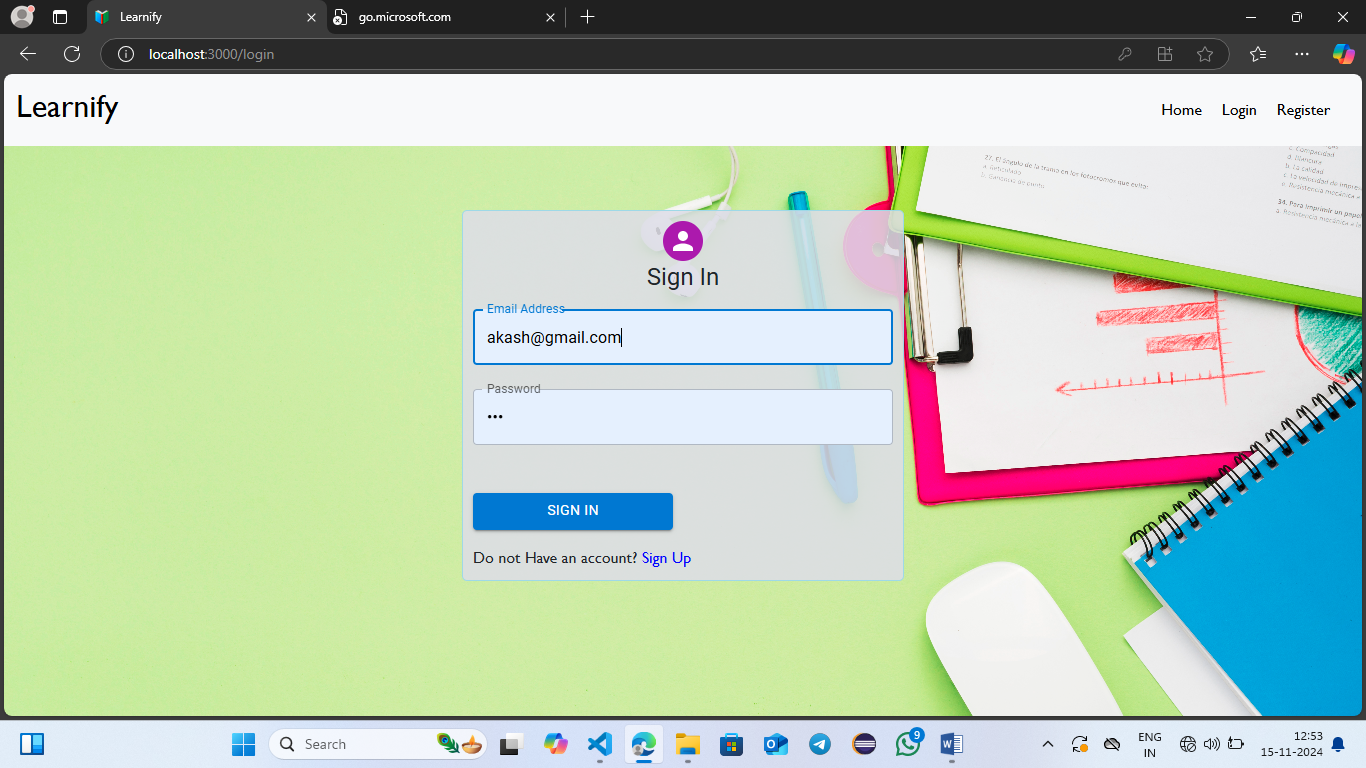
**Home Page**



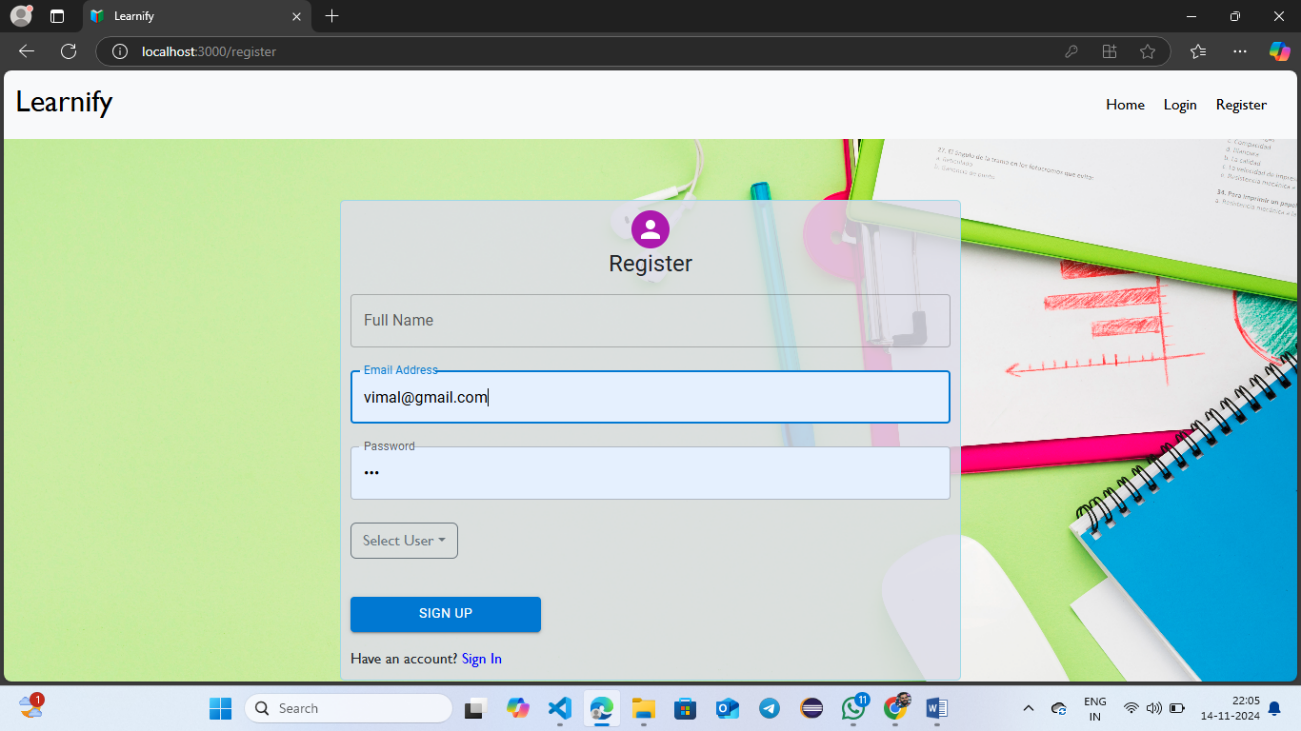
**Trending Courses**



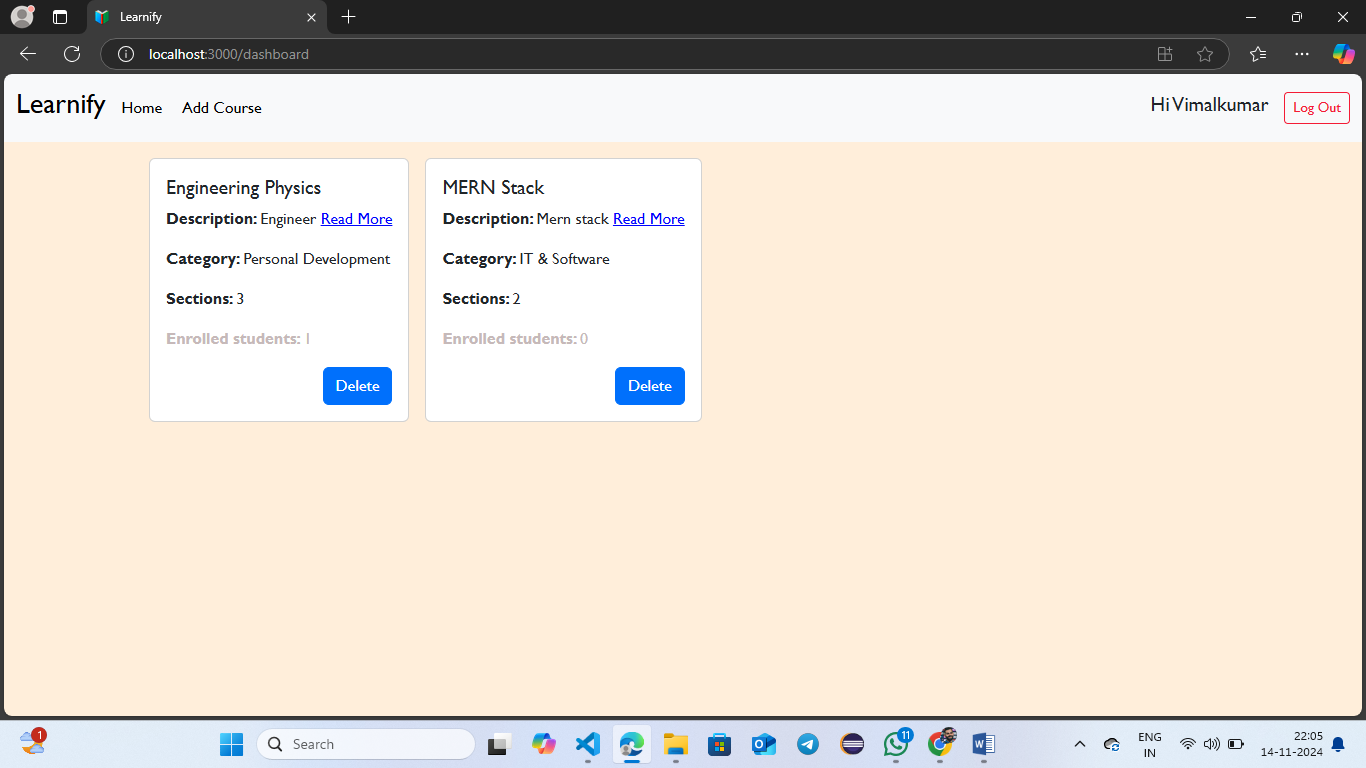
**Login Page**



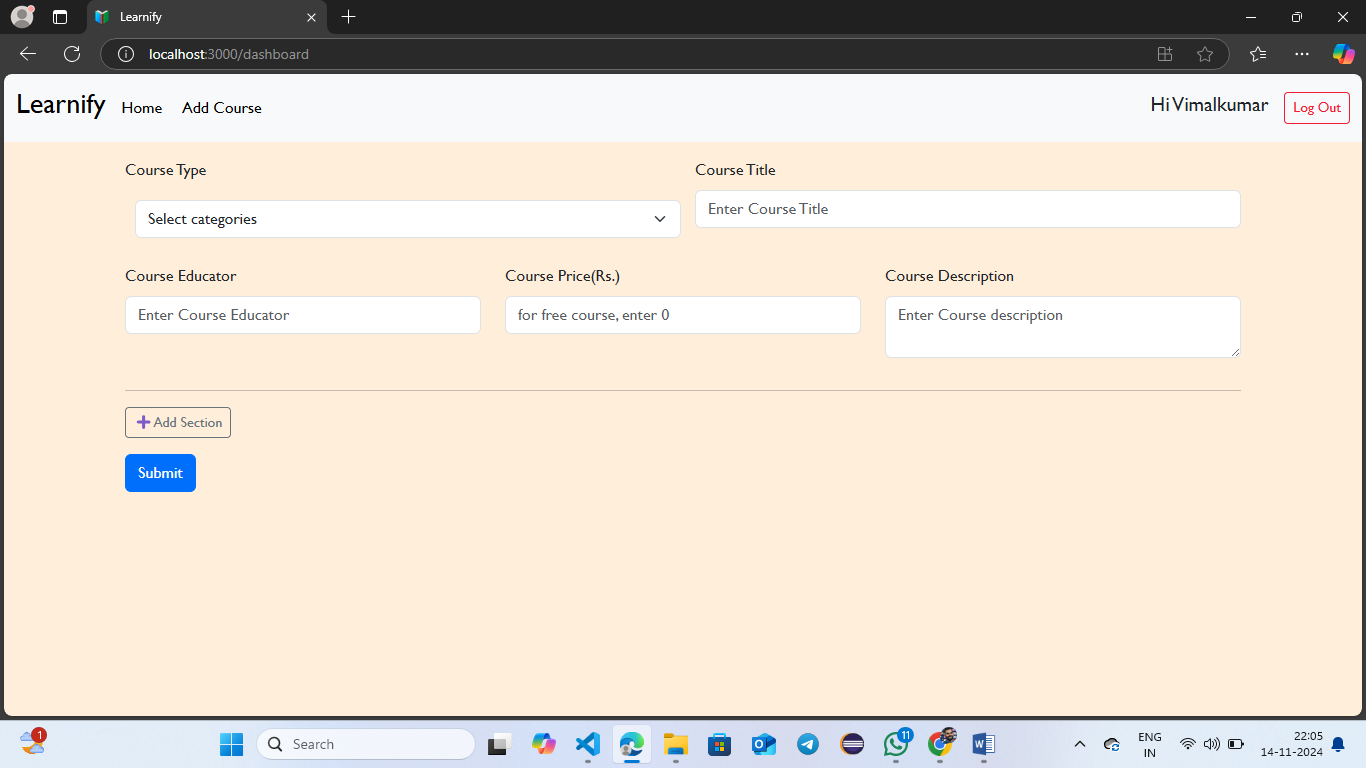
**Register Page**



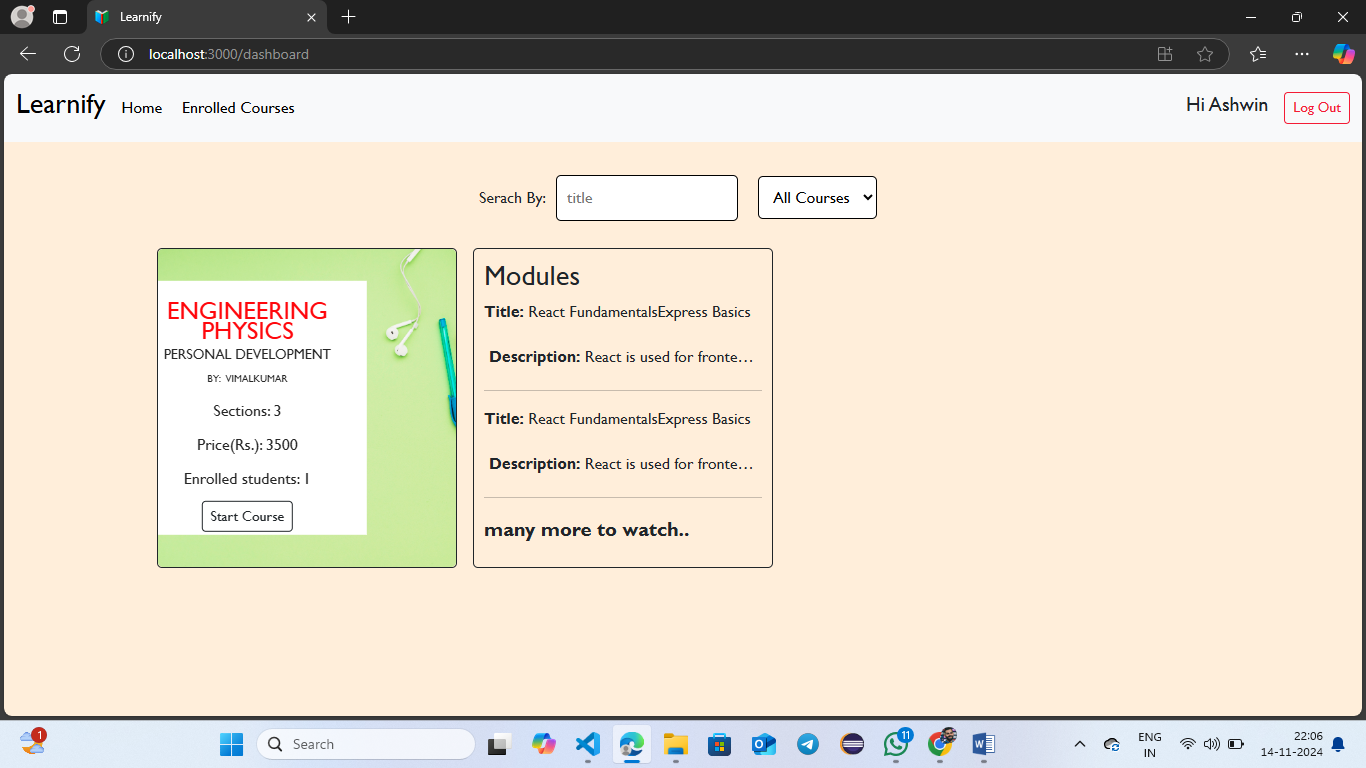
**Teacher dashboard**



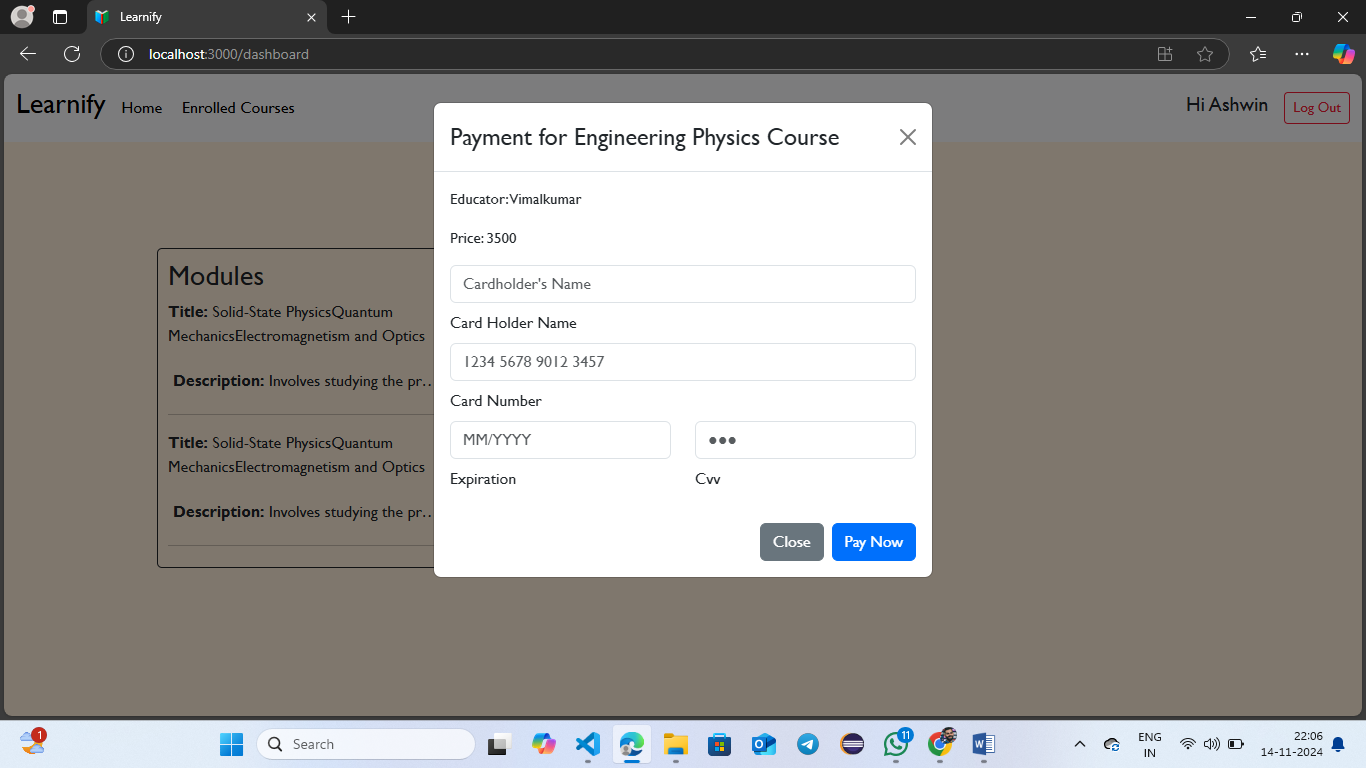
**Add Course**



**Student Dashboard**



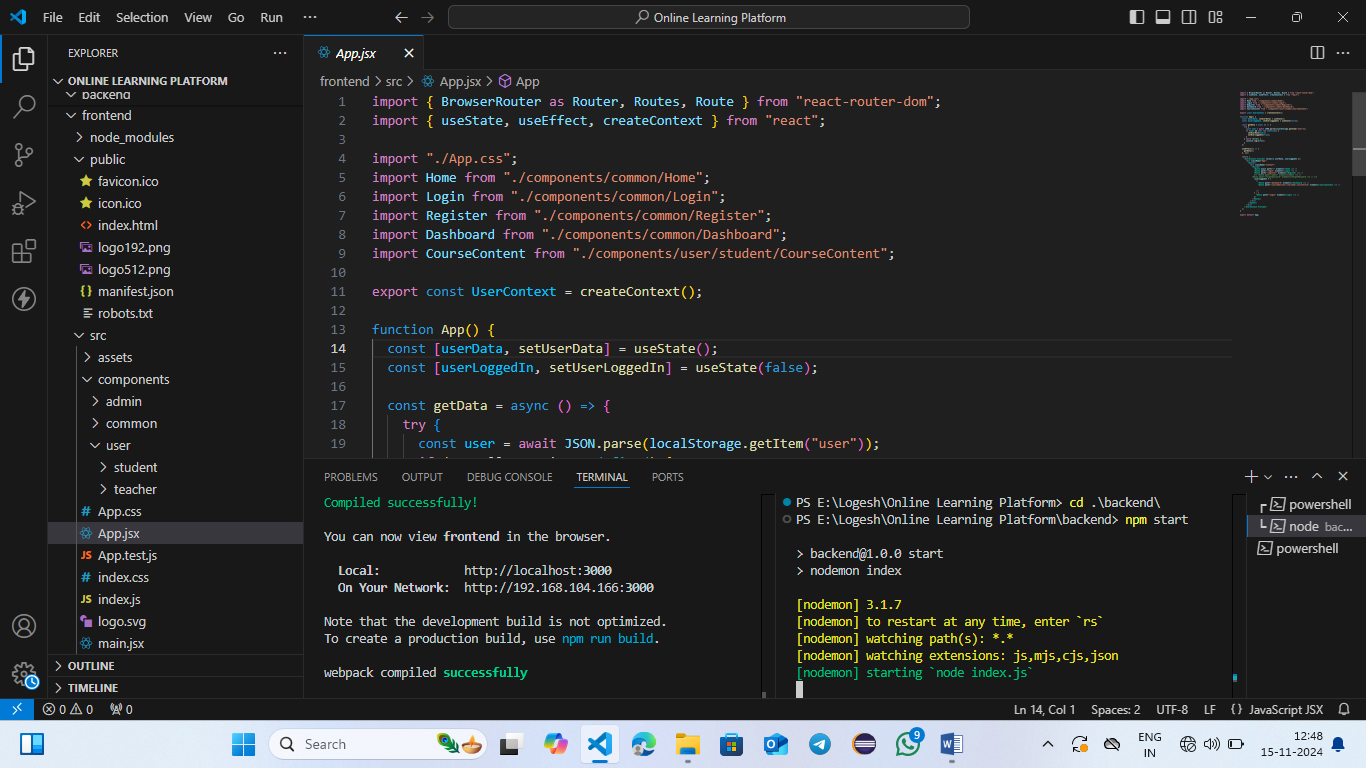
**Payment Page**



**Enrolled Courses Page**



**Project Structure**



**10. Testing**

* **Testing Strategy:**
  + Unit tests for individual components and functions using Jest.
  + Integration tests for API endpoints using Mocha and Chai.
  + UI testing with Cypress to ensure a seamless user experience.

**11. Demo Link**

**https://github.com/Logesh235/Online-Learning-Platform/blob/master/Demo%20Video%20for%20OLP%20Project.mp4**

**12. Known Issues**

**Issues:**

* Some filtering options may cause performance delays for large datasets.
* Mobile layout needs further optimization for smaller screens.

**13. Future Enhancements**

**Planned Features:**

* Addition of a forum or community feature for students to discuss course content.
* Integration of payment gateways for seamless course purchases.
* Implement AI-powered recommendations for courses based on user interests and past enrollments.

This documentation provides an organized overview, technical details, and future improvements to ensure comprehensive coverage of the online learning platform's development and features.