**Full Stack Development with MERN**

**Project Documentation**

**1.Introduction:**

**Project title:**Online learning website using mern stack

**Team members:**

1.Gayathri.S-au112621104005

2.Ashna.N-au112621104002

3.Hudha.N-au112621104009

4.Vishwanathan.T-au112621104025

**2.Project overview:**

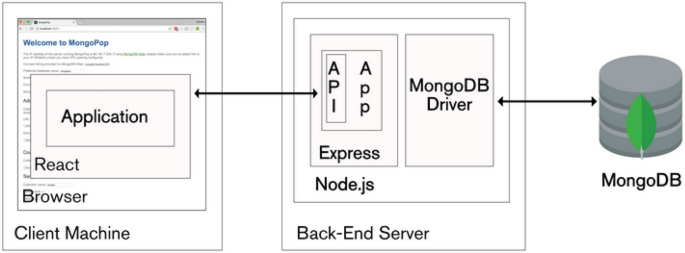
**Purpose:**

* Centralized Learning Hub
* Interactive Learning Environment
* Accessibility and Flexibility
* Scalability and Performance
* Real-time Features
* Personalized Learning
* Instructor Tools
* Community Building
* Modern Web Development Practices
* Cost-Effective and Fast Development

**Features:**

* Student: Register, enroll, access course materials, participate in discussions, and leave reviews.
* Instructor: Create courses, upload content, track student progress, and engage in forums.
* Admin: Approve courses, moderate content, view analytics, and manage notifications.
* General: Search/filter courses, real-time notifications, payments (subscriptions, discounts), and mobile compatibility.
* Security: Secure login (JWT), data privacy compliance.

**3. Architecture**



**4.Setup Instructions**

**1.System Requirements**

* Node.js
* MongoDB
* Code Editor: Visual Studio Code
* Git: To clone the project repository

**2. Installation Steps**

Step 1: Clone the Repository

* Open a terminal and clone the project repository from GitHub:

**git clone <repository-url>**

* Navigate into the project directory:

**cd <project-directory>**

Step 2: Set Up the Backend

* Navigate to the backend folder:

**cd backend**

* Install backend dependencies:

**npm install**

* Create an .env file in the backend directory with the following environment variables:

**MONGODB\_URI=<your-mongodb-uri>**

**JWT\_SECRET=<your-jwt-secret>**

**STRIPE\_API\_KEY=<your-stripe-api-key>**

**5. Folder Structure**

* **backend/**: Contains the server-side code, including configuration files, controllers for handling requests, models for defining database schemas, routes for defining API endpoints, middlewares for authentication and error handling, and utility functions.
* **frontend/**: Contains the client-side code with React, including reusable components, page-specific components, and context files for managing global state.
* **public/**: Houses static files, such as images, icons, and the main HTML file.
* **src/**: Contains the core files of the React application:
  + **components/**: Reusable components shared across multiple pages.
  + **pages/**: Each page represents a view in the app, like the homepage, course page, and profile page.
  + **context/**: Holds context providers, such as AuthContext, to manage state globally.
  + **services/**: Defines API calls and any data-fetching logic.

**6.Running the Application**

**Steps to Run the Application**

Start the Application

Starting the Backend:

* 1. In the backend folder, run:

**npm start**

* 1. The backend server should start on http://localhost:5000 (or as specified).

Starting the Frontend:

* 1. In the frontend folder, run:

**npm run dev**

* 1. The frontend should be accessible at http://localhost:3000.

**7.API Documentation**

**User**

* **POST /register**: Register a new user.
* **POST /login**: Log in a user and get a token

**Course**

* **GET /courses**: Get all available courses.
* **POST /courses/enroll**: Enroll in a course.
* **GET /courses//materials**: Access course materials.

**Admin**

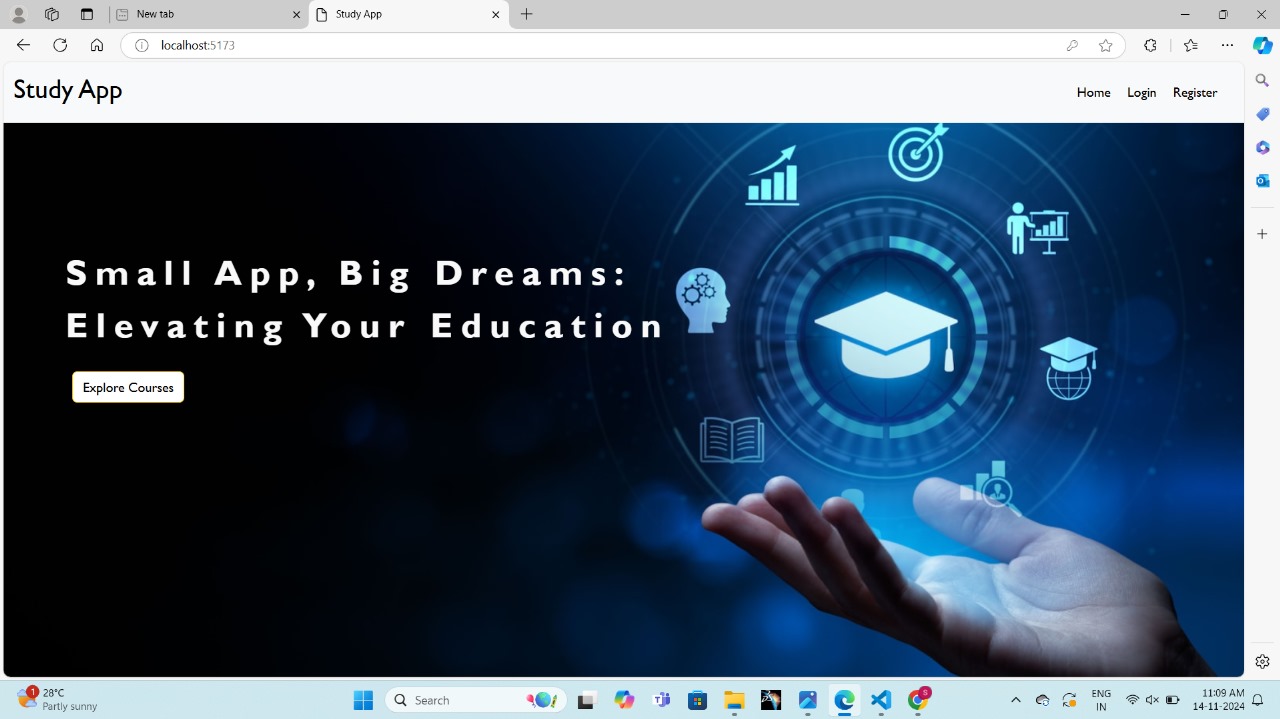
* **GET /admin/users:** List all users.
* **GET /admin/courses:** List all courses.

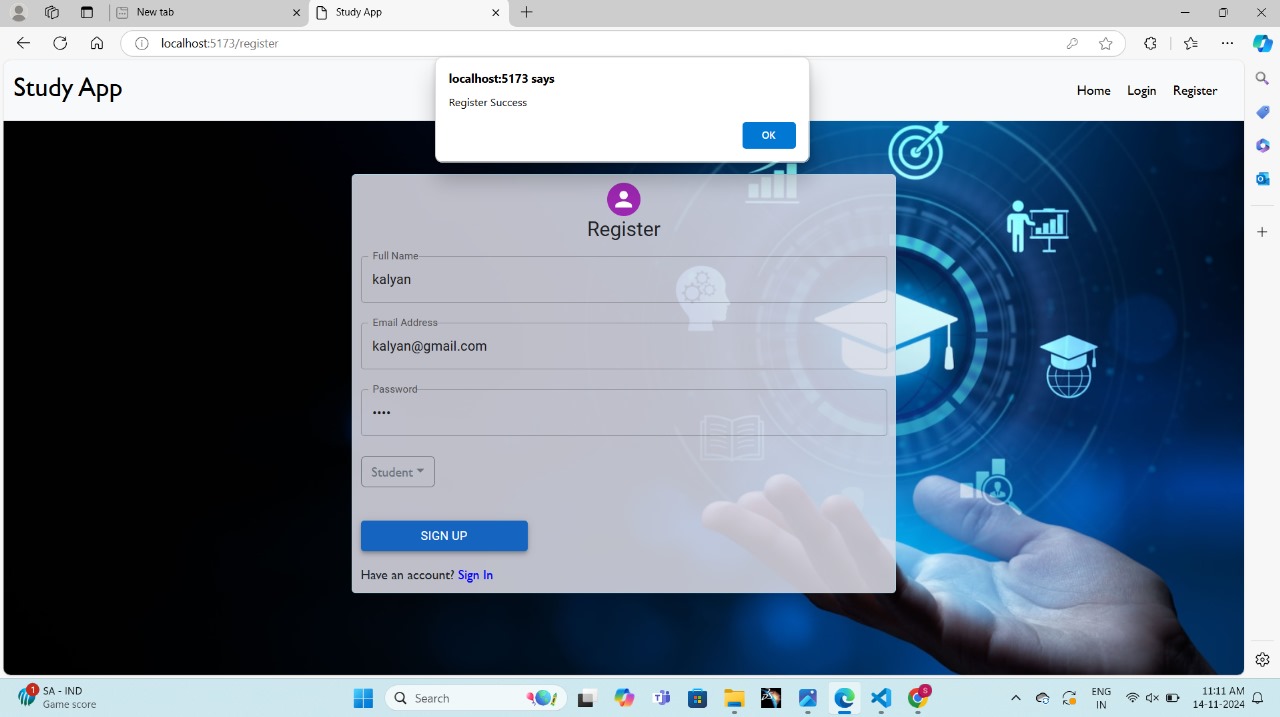
**8.Authentication**

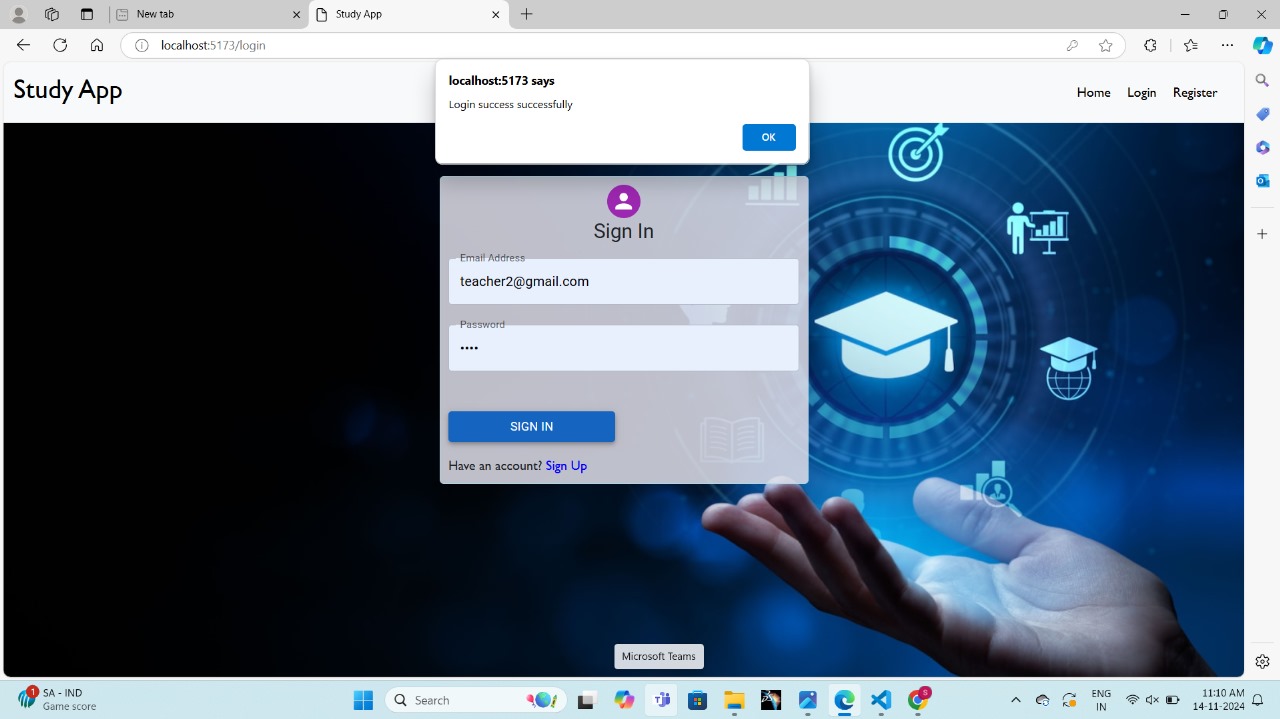
JSON Web Token (JWT) for Authentication and Authorization

1. Token-Based Authentication:
   * Registration and Login: During registration, user credentials are validated and stored securely in MongoDB. When a user logs in, the backend verifies their credentials. If the credentials are valid, the server generates a JWT, which is then sent back to the user as part of the response.
   * Token Structure: The JWT contains a payload with user-specific data (such as user ID and role). It is signed with a secret key, which helps verify the authenticity of the token on subsequent requests.

**9. User Interface**



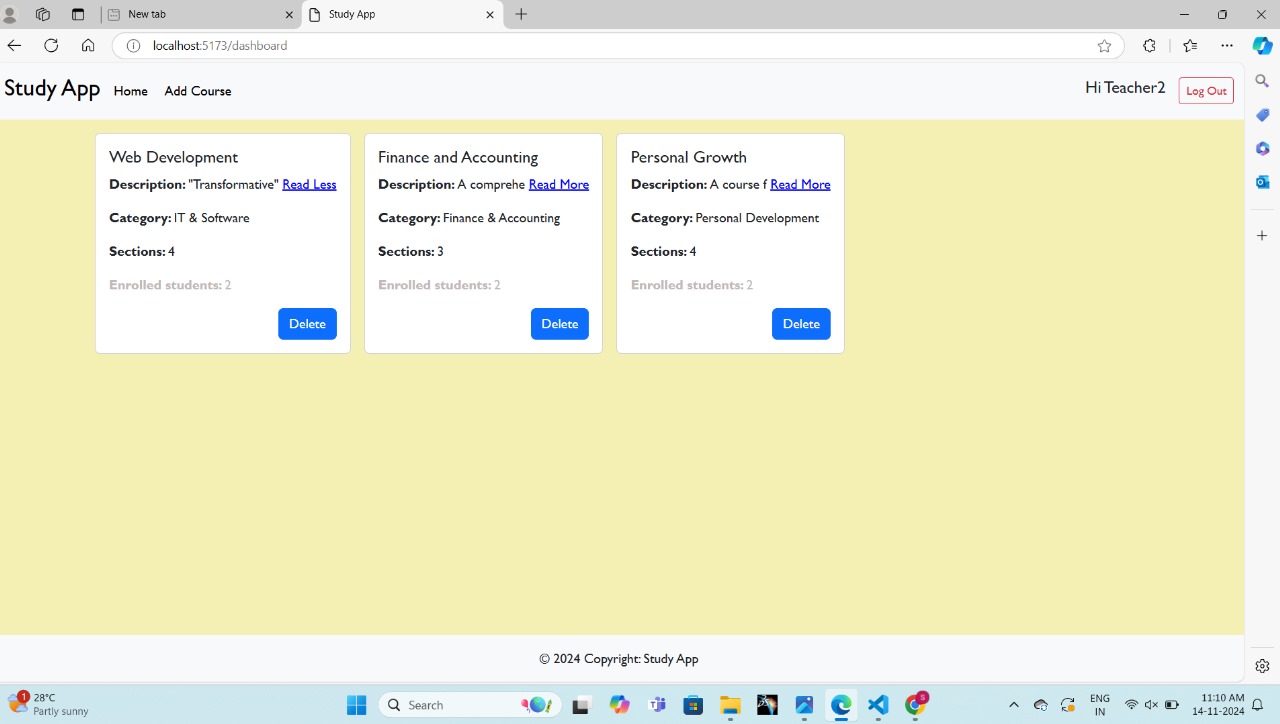
****

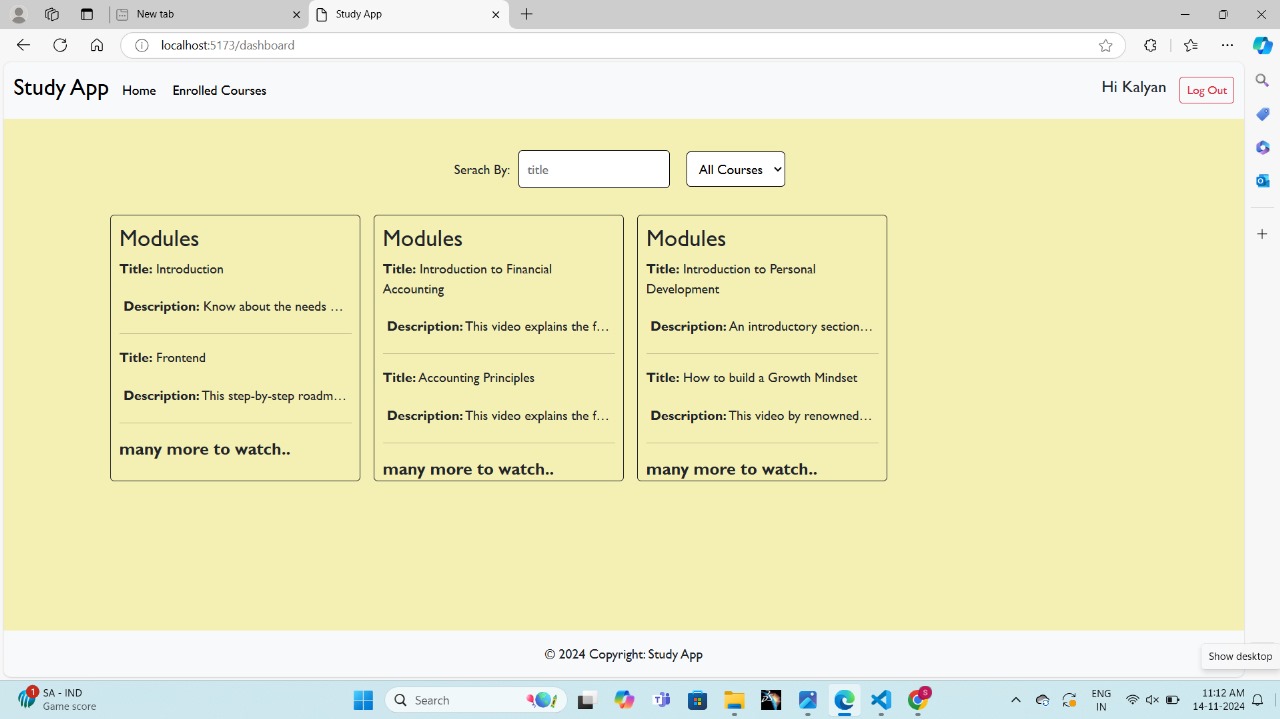
****

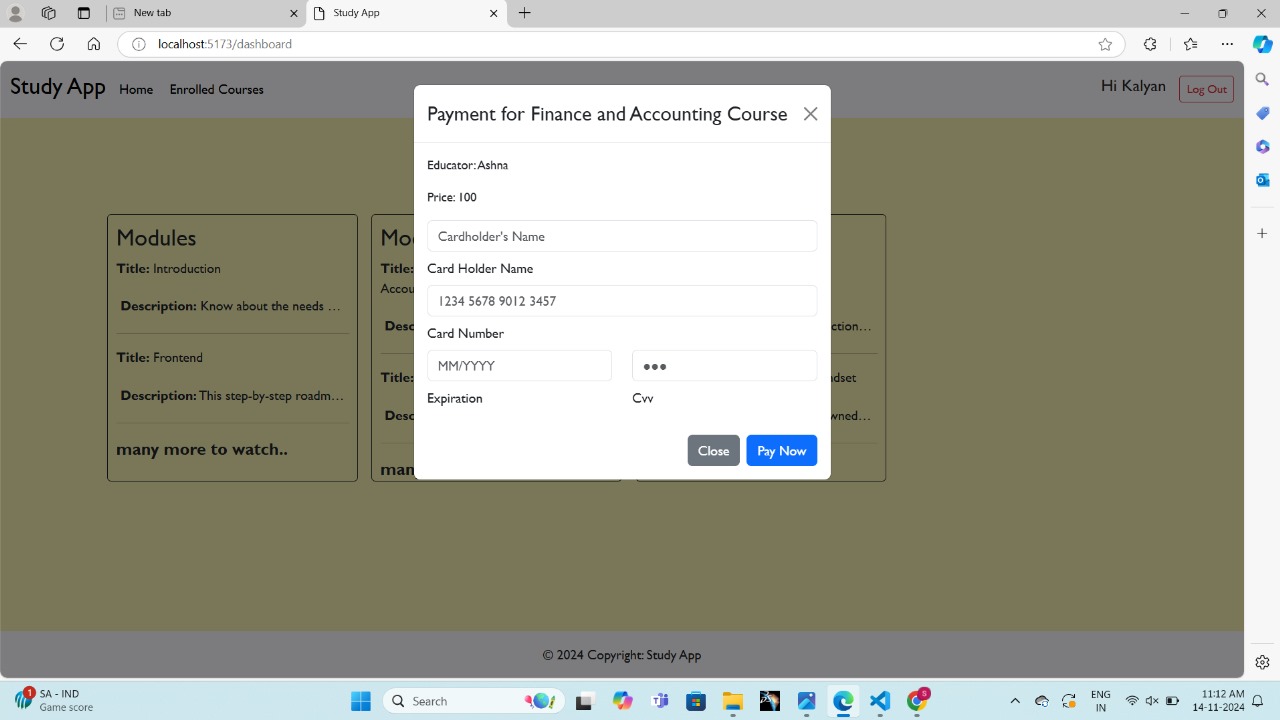
**10. Testing**

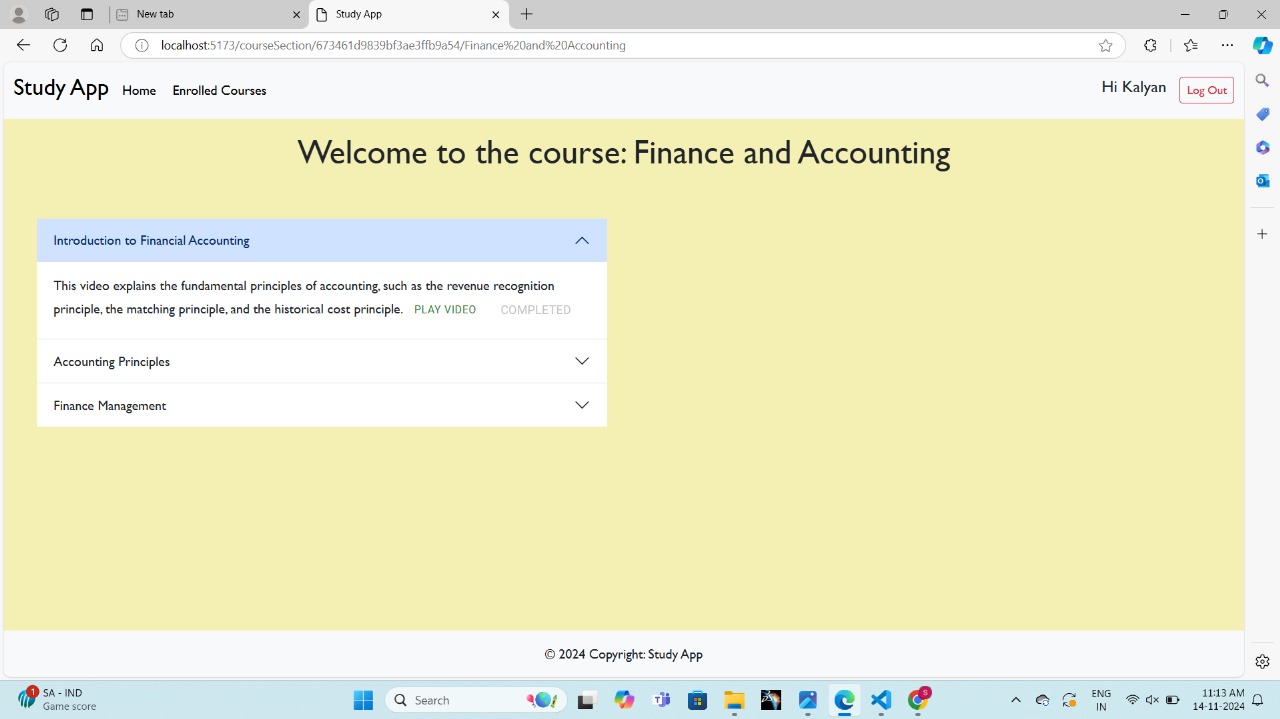
Manual testing of all frontend and backend functionality.

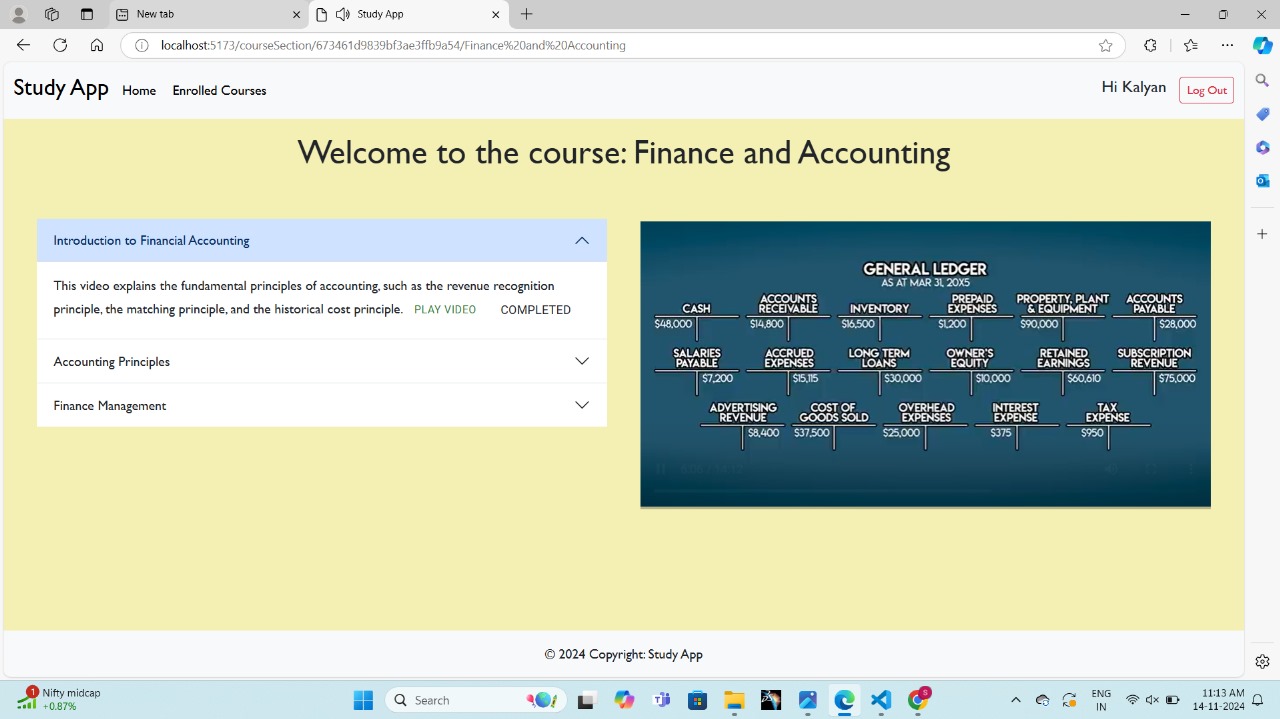
**11. Screenshots or Demo**

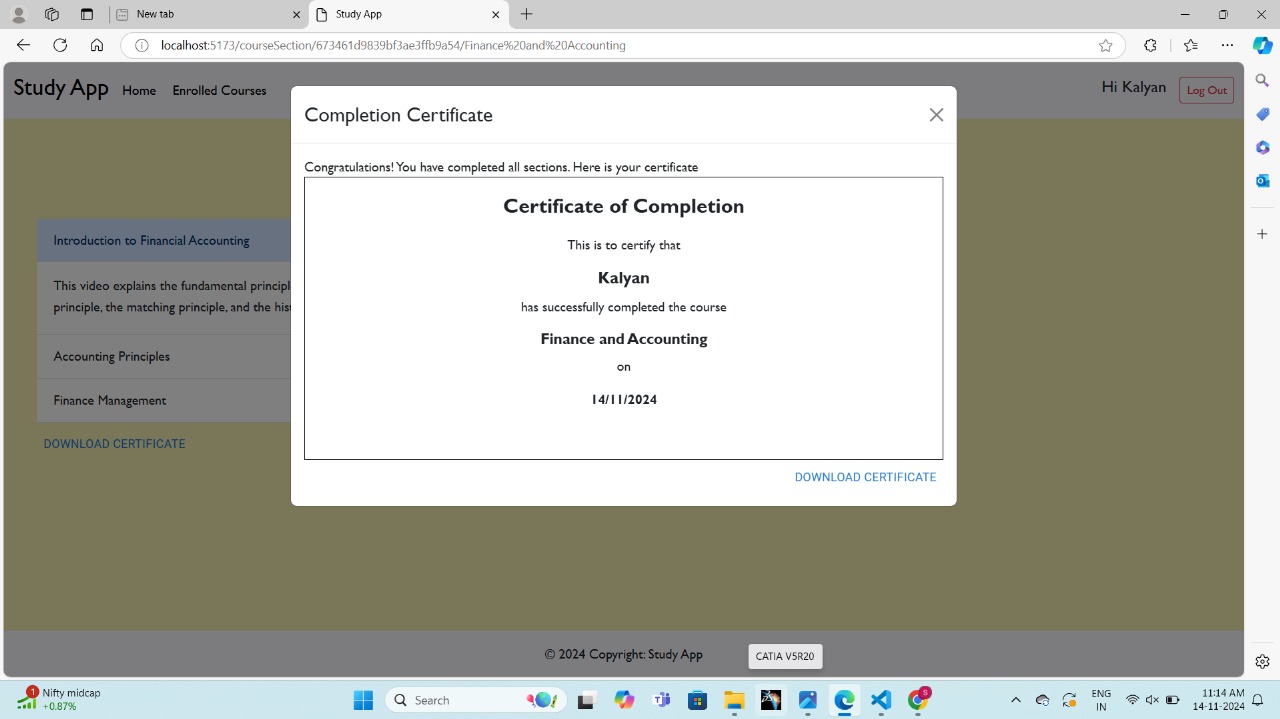
****

****

****

****

****

****

**12. Known Issues**

* MongoDB connection gets expired within limited time.
* Takes time to reload page

**13. Future Enhancements**

1. Enhanced User Profiles:
2. Advanced Search and Filtering:
3. Discussion Forums and Q&A:
4. Live Class Integration:
5. Gamification:
6. Mobile App Development: