Full Stack Development with MERN

1. INTRODUCTION:

Project Title: Skill Plus - An E learning WebApp
Team Members:

Noorul Ameena. S - UI/UX Designer
Priyadharshini. P - Frontend Developer
Mohammed Thashreef. S - Frontend Developer
Muhammad Firas. S - Backend Developer
Sameena Mumtaz. F - Backend Developer

---

2. PROJECT OVERVIEW:

In an era where digital education has become an integral part of the learning landscape, our platform aims to provide a holistic and interactive environment for individuals seeking knowledge and those looking to share their expertise. The platform will feature a range of functionalities, ensuring a smooth user experience and fostering an engaging and collaborative learning community. This platform is designed to cater to the needs of both learners and instructors, offering a seamless and enriching educational experience. The primary goal of this project is to create a robust and feature-rich e-learning solution that addresses the distinct requirements of students, instructors, and administrators alike.

- **PURPOSE:**

  The purpose of the proposed online learning platform is to create a dynamic and user-centric environment that facilitates effective teaching and learning experiences. This platform aims to bridge the gap between learners and instructors, providing a comprehensive suite of features for education seekers, content creators, and administrators. Key objectives include:

  ❖ Enhanced Learning Experience
  ❖ Empower Instructors
  ❖ Administrative Efficiency
  ❖ Community Building:
  ❖ Scalability

- **FEATURES:**

The e-learning platform offers a range of functionalities to facilitate learning and collaboration within enterprises:

- ❖ User Authentication and Authorization: Users can register, login, and manage their accounts securely. Role-based access control ensures appropriate access levels for learners, instructors, and administrators.
- ❖ Course Management: Instructors can create and manage courses, including adding modules, assignments, quizzes, and learning materials. Learners can enroll in courses and track their progress.
- ❖ Interactive Learning Environment: The platform provides interactive features such as discussion forums, live chat, and virtual classrooms to foster collaboration and engagement among learners and instructors.
- ❖ Analytics and Reporting: Analytics capabilities are integrated to track user activity, monitor learning progress, and generate reports for administrators and instructors. Insights derived from analytics data help optimize learning outcomes and identify areas for improvement.

---

3. ARCHITECTURE:

The architecture of the e-learning platform is based on the MERN stack, comprising MongoDB, Express.js, React.js, and Node.js. MongoDB serves as the database solution, providing flexibility and scalability for storing and managing data related to users, courses, modules, and learning materials. Express.js facilitates the development of robust web servers, handling routing, middleware, and HTTP requests. React.js powers the frontend of the platform, enabling the creation of dynamic and interactive user interfaces. Node.js serves as the runtime environment for server-side applications, allowing for the implementation of business logic and integration with external systems.

- **FRONTEND:**

  The frontend architecture using React is built on a component-based structure, where reusable, independent components form the core of the UI. Data flows unidirectionally through props, with state managed locally using hooks like `useState` or globally using tools like Redux or Context API. A clear folder structure organizes components, pages, hooks, services, and styles for maintainability. React Router enables seamless navigation, while CSS Modules, styled-components, or frameworks like Tailwind CSS handle styling. API integrations are abstracted into service files, keeping components clean. Performance is optimized through techniques like code splitting and memoization, while testing at various levels ensures robustness. Tools like Vite or Webpack bundle the application, which can be deployed using platforms like Vercel or Netlify. This architecture emphasizes scalability, maintainability, and a great user experience.

- **BACKEND:**

  The backend architecture using Node.js and Express.js is modular and scalable, featuring a structured folder system for controllers, models, routes, middleware, and utilities. Express handles routing, while middleware manages tasks like authentication, validation, and error handling. Database integration is achieved using tools like Mongoose for MongoDB database. RESTful APIs or GraphQL facilitate client communication, secured with practices like JWT-based authentication, input validation, and headers protection via Helmet. Performance is optimized with caching (Redis) and clustering. Testing is handled with tools like Jest or Mocha, and deployment is streamlined with PM2 and cloud platforms like AWS or Heroku.

- **DATABASE:**

  In MongoDB, schema design revolves around collections and documents, often managed using Mongoose for schema definition and interactions. For example, in an e-learning platform, collections like Users, Courses, and Assignments store data with defined schemas, incorporating relationships through references (e.g., a course referencing multiple students). CRUD operations such as creating users, fetching courses, updating enrollments, and deleting data are handled efficiently using Mongoose methods. Indexing, like on email fields, improves query performance, while transactions ensure atomicity for multi-collection operations, such as enrolling a student in a course. Validation, such as enforcing email formats or required fields, maintains data integrity. This approach enables flexibility and scalability for applications leveraging MongoDB's NoSQL capabilities.

---

4. SETUP INSTRUCTION:

- **PREREQUISITE:**
  Before setting up the application, ensure the following software dependencies are installed:

  1. **Node.js:** Install the latest LTS version (recommended) from [Node.js official website](https://nodejs.org/).
     - Verify installation:
       ```bash
       node -v
       npm -v
       ```

  2. **MongoDB:** Install MongoDB Community Edition or use a cloud-hosted service like [MongoDB Atlas](https://www.mongodb.com/atlas).
     - Verify installation:

```bash
mongodb --version
```

3. **Package Manager:**
   - **NPM**: Comes with Node.js installation.

4. **Git:** Install Git for version control from [Git official website](https://git-scm.com/).
   - Verify installation:
     ```bash
     git --version
     ```

5. **Code Editor:** Install a code editor like [Visual Studio Code](https://code.visualstudio.com/) for development.

6. **Environment Variables Management:** Use a tool or library like `dotenv` for managing environment variables.

- **Additional Tools**

  Node Package Dependencies:
  Install required dependencies for the project after cloning the repository:
  ```bash
  npm install
  ```
  Common dependencies include:
  - express: Web framework.
  - mongoose: MongoDB object modeling.
  - dotenv: Environment variable management.
  - cors, helmet: Middleware for security.
  - bcrypt, jsonwebtoken: For authentication.

  With these prerequisites in place, you can proceed to set up the project locally or deploy it on your desired platform.

- **INSTALLATION GUIDE:**

  Follow these steps to set up the SkillPlus project on your local machine:

  1. **Clone the Repository**
  Download the project repository to your local machine:
  ```bash
  ```

```
git clone https://github.com/TSF-Solutions-24/Naan_Mudhalvan
cd Naan_Mudhalvan
```

2. **Install Dependencies**
Install the required Node.js dependencies on both frontend and backend:
```bash
npm install
```

3. **Set Up Environment Variables**
Create a `.env` file in the project root directory and add the following variables:
```plaintext
MONGO_URI=<your-mongodb-connection-string>
JWT_SECRET=<your-secret-key>
PORT=<desired-port-number, default: 3000>
```
- Replace `<your-mongodb-connection-string>` with your MongoDB connection URI.
- Replace `<your-secret-key>` with a strong secret key for JWT authentication.
- Optionally set a custom `<desired-port-number>` for the server.

---

4. **Start the Server**
Run the development server:
```bash
npm run dev
```
The server will start, and the application will be accessible at `http://localhost:<PORT>` (default: `3000`).

5. Test the Application
Open a browser or API testing tool like Postman and interact with the application to ensure everything is working correctly.

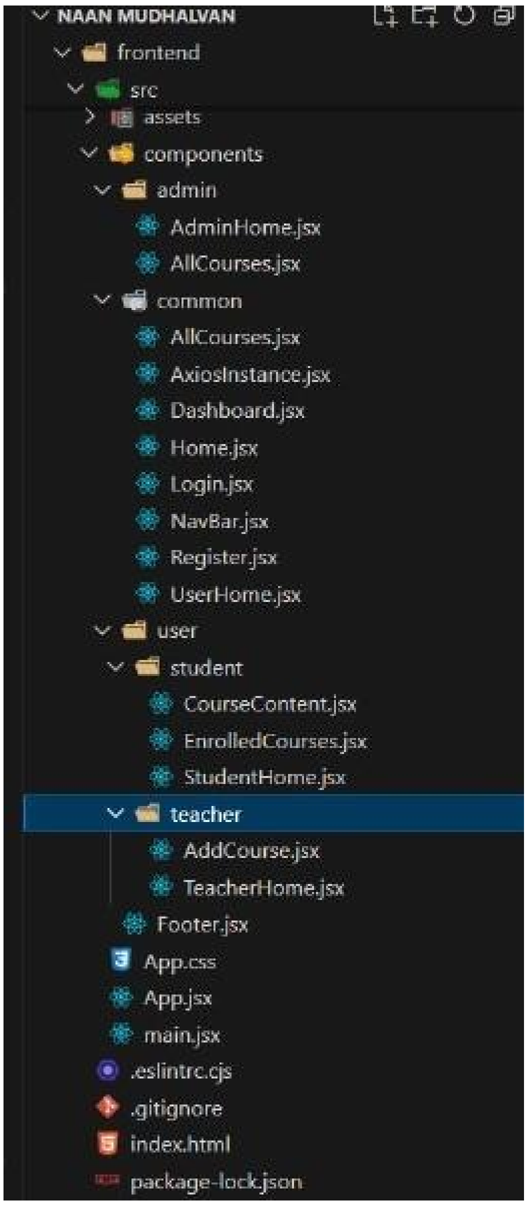
You're now ready to use SkillPlus!

---

5. FOLDER STRUCTURE:

- **Client-side: React Frontend Structure**

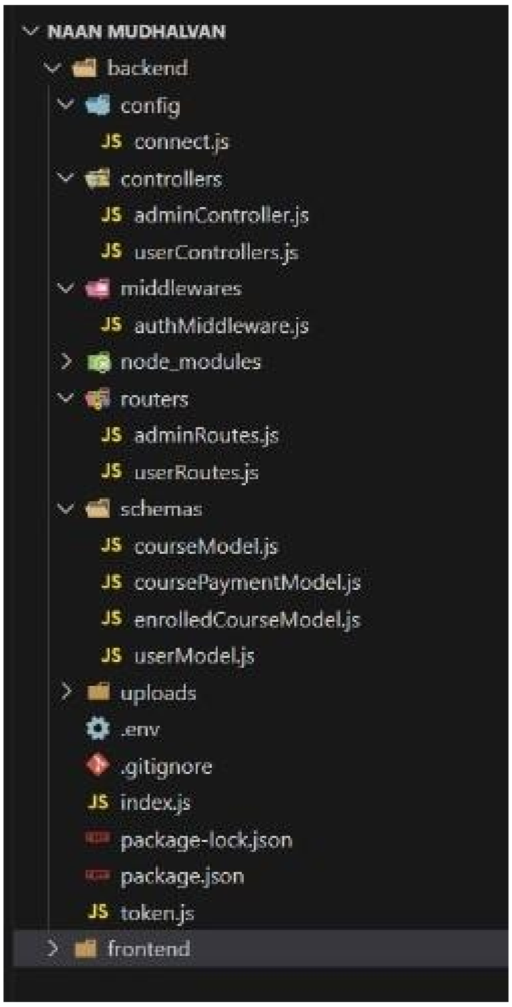
  The React frontend for SkillPlus is designed with a modular and scalable architecture to ensure maintainability and efficiency. A typical folder structure for the frontend might look like this:



```
NAAN MUDHALVAN
  frontend
    src
      assets
      components
        admin
          AdminHome.jsx
          AllCourses.jsx
        common
          AllCourses.jsx
          AxiosInstance.jsx
          Dashboard.jsx
          Home.jsx
          Login.jsx
          NavBar.jsx
          Register.jsx
          UserHome.jsx
        user
          student
            CourseContent.jsx
            EnrolledCourses.jsx
            StudentHome.jsx
          teacher
            AddCourse.jsx
            TeacherHome.jsx
        Footer.jsx
      App.css
      App.jsx
      main.jsx
    .eslintrc.cjs
    .gitignore
    index.html
    package-lock.json
```

- **Server-side: Node.js Backend Organization**

  The Node.js backend for SkillPlus follows a structured and modular architecture to ensure scalability, maintainability, and ease of development. The backend is organized into the following directories and files:



```
NAAN MUDHALVAN
  backend
    config
      connect.js
    controllers
      adminController.js
      userControllers.js
    middlewares
      authMiddleware.js
    node_modules
    routers
      adminRoutes.js
      userRoutes.js
    schemas
      courseModel.js
      coursePaymentModel.js
      enrolledCourseModel.js
      userModel.js
    uploads
    .env
    .gitignore
    index.js
    package-lock.json
    package.json
    token.js
  frontend
```

---

6. RUNNING THE APPLICATION:

Follow these commands to start the frontend and backend servers locally:

1. Frontend:
Navigate to the frontend directory (Frontend) and start the React development server:
```bash
cd frontend
npm start
```
- The frontend server will run on `http://localhost:3000` by default.

2. Backend
Navigate to the backend directory (backend) and start the Node.js server:
```bash
```

cd backend
npm start
```
- The backend server will run on `http://localhost:5000` by default (or as configured in the `.env` file).

- **Ensure Both Servers are Running**
  - Frontend interacts with the backend through API endpoints.
  - Confirm both servers are up to experience the full functionality of the application.

---

## 7. API DOCUMENTATION:

The backend of **SkillPlus** exposes several endpoints to handle authentication, course management, and user interactions. Below is the detailed documentation of the available endpoints:

**1. Authentication Endpoints**

1.1 Register User

❖ Method:`POST`
❖ Endpoint: `/api/auth/register`
❖ Description:Creates a new user account with role-based access.
❖ Parameters:
  Body:
  - `name` (string) - Name of the user.
  - `email` (string) - Email address.
  - `password` (string) - Password for the account.
  - `role` (string) - Role of the user (`student`, `instructor`, `admin`).

1.2 Login User

❖ Method: `POST`
❖ Endpoint: `/api/auth/login`
❖ Description:Authenticates user and issues a JWT token.
❖ Parameters:
  Body:
  - `email` (string) - User's email.
  - `password` (string) - User's password.

1.3 Refresh Token
```

- Method: `POST`
- Endpoint: `/api/auth/refresh`
- Description: Provides a new JWT access token using a refresh token.
- Parameters:

  Body:

  - refreshToken` (string) - Refresh token for session renewal.

2. User Management Endpoints

2.1 Get User Profile

- - Method:`GET`
- - Endpoint: `/api/users/profile`
- - Description: Retrieves the profile of the logged-in user.
- - Authorization: Bearer Token required.

2.2 Update User Profile

- - Method: `PUT`
- - Endpoint: `/api/users/profile`
- - Description: Updates the profile information of the logged-in user.
- - Parameters:

  Body:

  - `name` (string) - Updated name of the user.
  - `email` (string) - Updated email address.

3. Course Management Endpoints

3.1 Get All Courses

- Method: `GET`
- Endpoint: `/api/courses`
- Description: Retrieves a list of all available courses.
- Query Parameters:
  - `page` (number) - Page number for pagination.
  - `limit` (number) - Number of courses per page.

3.2 Create a Course

- Method: `POST`
- Endpoint: `/api/courses`
- Description: Allows instructors to create a new course.
- Authorization: Bearer Token required (Instructor role).
- Parameters:

  Body:

  - `title` (string) - Title of the course.
  - `description` (string) - Detailed description.

- `modules` (array) - List of course modules.

3.3 Enroll in a Course

- ❖ Method: `POST`
- ❖ Endpoint: `/api/courses/enroll`
- ❖ Description: Allows students to enroll in a specific course.
- ❖ Authorization: Bearer Token required.
- ❖ Parameters:
  Body:
  - `courseId` (string) - ID of the course to enroll in.

3.4 Track Progress

- ❖ Method: `GET`
- ❖ Endpoint: `/api/courses/progress`
- ❖ Description: Retrieves progress data for the enrolled courses of the logged-in user.
- ❖ Authorization: Bearer Token required.

4. Interaction Endpoints

4.1 Get Forum Threads

- ❖ Method: `GET`
- ❖ Endpoint: `/api/forum`
- ❖ Description: Retrieves all discussion threads for a course.
- ❖ Query Parameters:
  - `courseId` (string) - ID of the course.
  - `page` (number) - Page number for pagination.
  - `limit` (number) - Number of threads per page.

4.2 Post a Comment

- ❖ Method: `POST`
- ❖ Endpoint: `/api/forum/comment`
- ❖ Description: Allows users to post a comment in a discussion thread.
- ❖ Authorization: Bearer Token required.
- ❖ Parameters:
  Body:
  - `threadId` (string) - ID of the thread to comment on.
  - `content` (string) - Comment text.

5. Analytics Endpoints

5.1 Get Course Analytics

- ❖ Method: `GET`
- ❖ Endpoint: `/api/analytics/course`

- ❖ Description: Provides engagement metrics for a specific course.
- ❖ Authorization: Bearer Token required (Instructor/Admin role).
- ❖ Query Parameters:
  - `courseId` (string) - ID of the course.

5.2 Get User Activity Report

- ❖ Method: `GET`
- ❖ Endpoint:`/api/analytics/user`
- ❖ Description: Retrieves detailed activity logs for a user.
- ❖ Authorization: Bearer Token required (Admin role).
- ❖ Query Parameters:
  - `userId` (string) - ID of the user.

General Notes

- **Authorization:** Most endpoints require a valid JWT token in the `Authorization` header as `Bearer <token>`.
- **Response Format:** JSON objects with fields for `data`, `message`, and `status` are used for consistency.
- **Error Handling:** Returns appropriate status codes (`400`, `401`, `404`, etc.) with descriptive error messages.

---

8. AUTHENTICATION AND AUTHORIZATION IN SKILLPLUS

The SkillPlus platform uses **JWT (JSON Web Token)** for handling **authentication** and **authorization**. This method ensures secure access to user-specific resources by verifying the identity of users and managing their access rights. Below is a detailed explanation of how authentication and authorization are managed:

1. Authentication

JWT Authentication Flow

❖ **User Registration**:
When a user registers, their details (e.g., name, email, and password) are stored in the database. The password is hashed using bcrypt to ensure security.

❖ **Login**:
Upon login, users provide their credentials (email and password). The system verifies the email and password. If valid, a **JWT token** is generated and sent back to the client. The token is signed using a secret key stored in the server environment (`JWT_SECRET`).

❖ **JWT Token**:
The generated token contains encoded information (such as the user ID) and is used to identify and authenticate the user in subsequent requests. The client stores the token, typically in **localStorage** or **sessionStorage** on the frontend.

2. Authorization

Protected Routes

To restrict access to certain resources (e.g., user profiles, course data), the backend uses middleware that verifies the JWT in the request headers. If the token is valid, the request proceeds; if not, the user is denied access.

❖ **Authorization Middleware**:
A custom middleware (`authMiddleware.js`) is used to check if the incoming request contains a valid token. The token is passed in the HTTP request headers under `Authorization: Bearer <token>`.

3. Token Expiry and Refresh

JWT tokens have an expiration time (`exp`) encoded in them. When the token expires, the user is required to log in again. To manage session persistence and avoid frequent logins, the system can implement a refresh token mechanism. The refresh token is used to obtain a new access token after the previous one expires. This ensures that users don't have to reauthenticate every time a token expires.

Refresh Token Flow:

➢ When the user logs in, two tokens are sent: the access token (JWT) and the refresh token.
➢ When the access token expires, the client sends the refresh token to the server to obtain a new access token.
➢ The refresh token is stored securely, and it has a longer expiry time compared to the access token.

4. Session Management

While JWT does not maintain server-side sessions (since the token is stateless and stored on the client), the application can optionally store the refresh token in a secure HTTP-only cookie for enhanced security, preventing cross-site scripting (XSS) attacks.

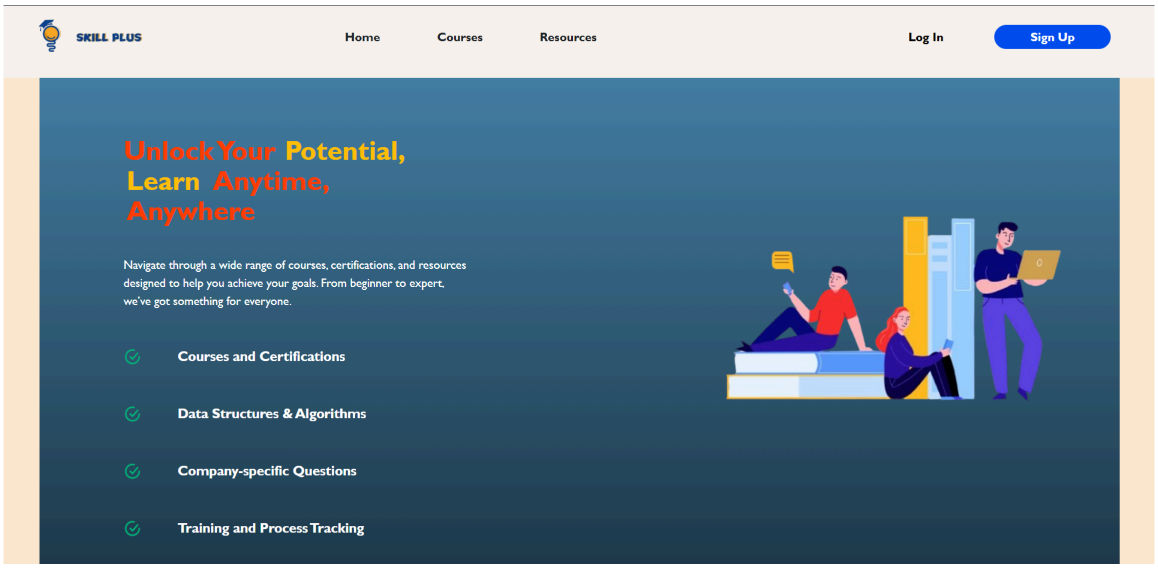
Summary of Authentication Flow:

❖ User registers, password is hashed, and data is stored.
❖ User logs in with email and password, and receives a JWT token.
❖ The JWT token is sent in subsequent requests to access protected routes.
❖ If the token expires, the user must log in again or use a refresh token to get a new JWT.

This approach with JWT ensures secure, scalable, and efficient authentication and authorization for the SkillPlus platform.

---

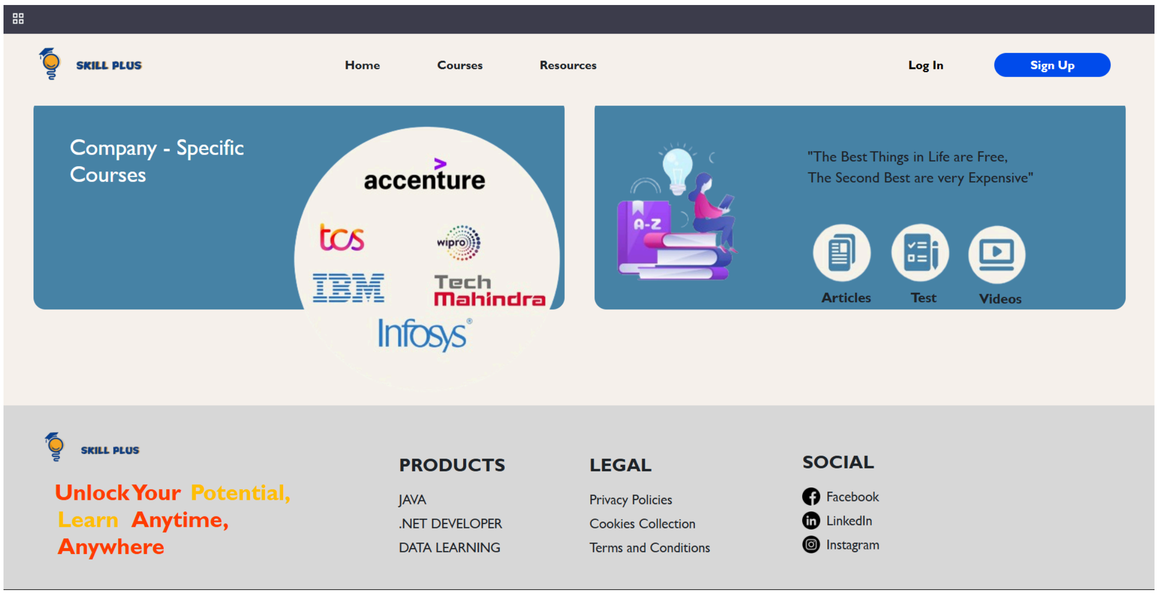
9. USER INTERFACE:



Unlock Your Potential, Learn Anytime, Anywhere

Navigate through a wide range of courses, certifications, and resources designed to help you achieve your goals. From beginner to expert, we've got something for everyone.

- Courses and Certifications
- Data Structures & Algorithms
- Company-specific Questions
- Training and Process Tracking

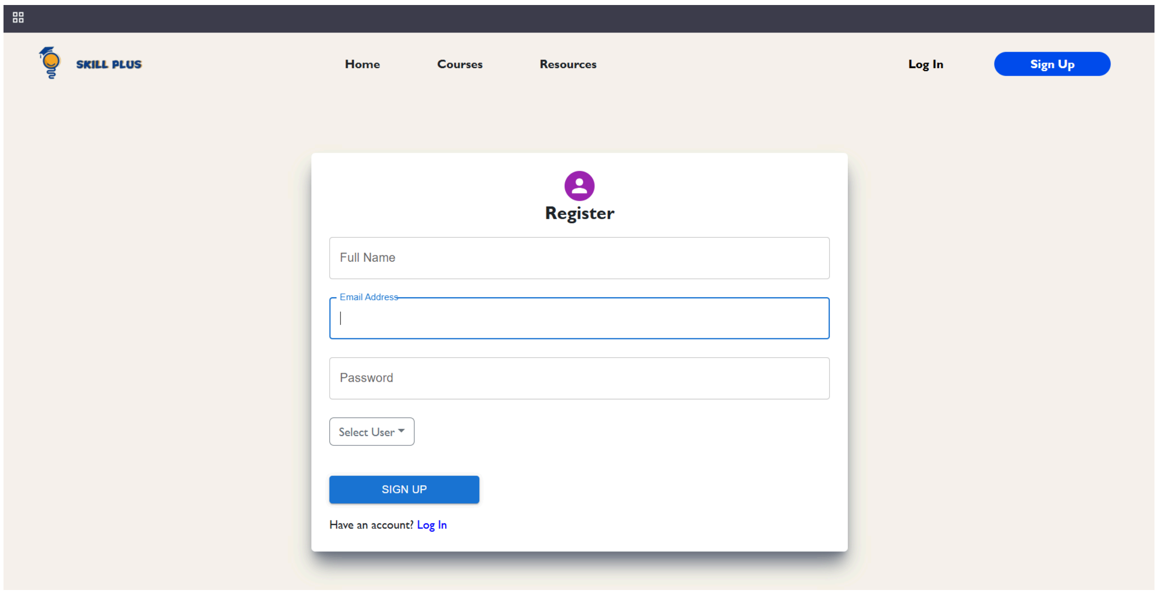


"The Best Things in Life are Free, The Second Best are very Expensive"

SKILL PLUS

Unlock Your Potential, Learn Anytime, Anywhere

PRODUCTS	LEGAL	SOCIAL
JAVA	Privacy Policies	Facebook
.NET DEVELOPER	Cookies Collection	LinkedIn
DATA LEARNING	Terms and Conditions	Instagram



Register

Full Name

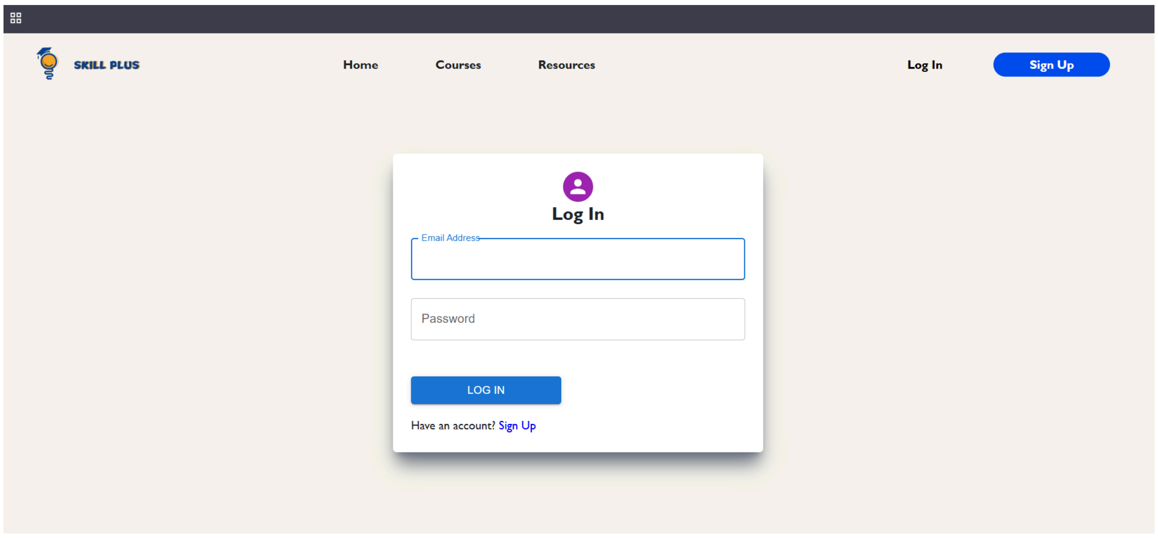
Email Address

Password

Select User

SIGN UP

Have an account? Log In



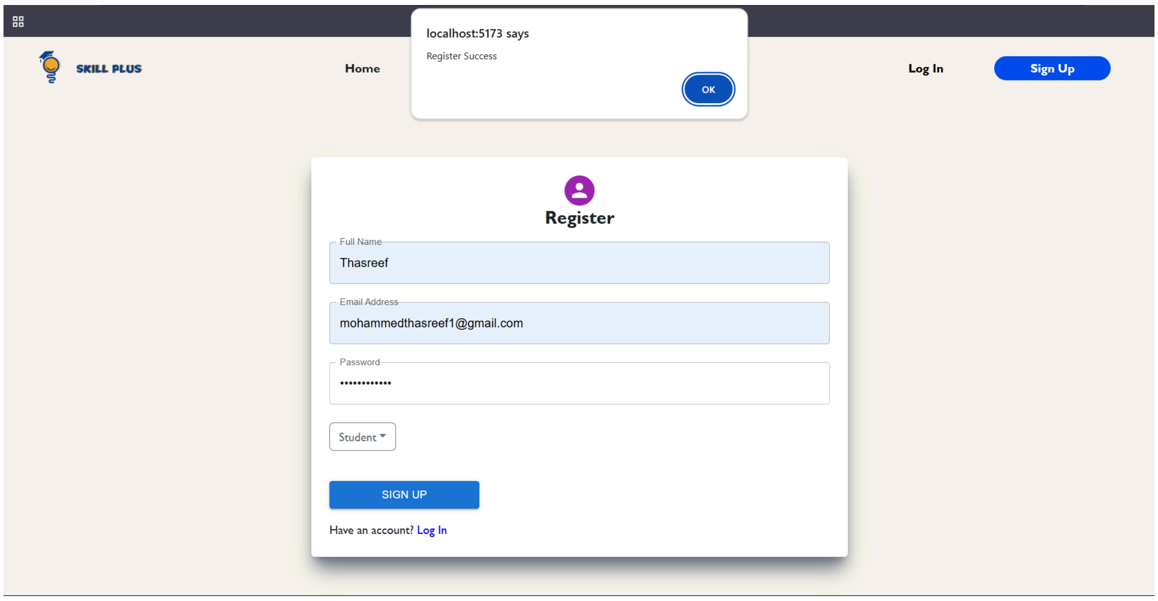
Log In

Email Address

Password

LOG IN

Have an account? Sign Up



localhost:5173 says

Register Success

OK

Register

Full Name: Thasreef

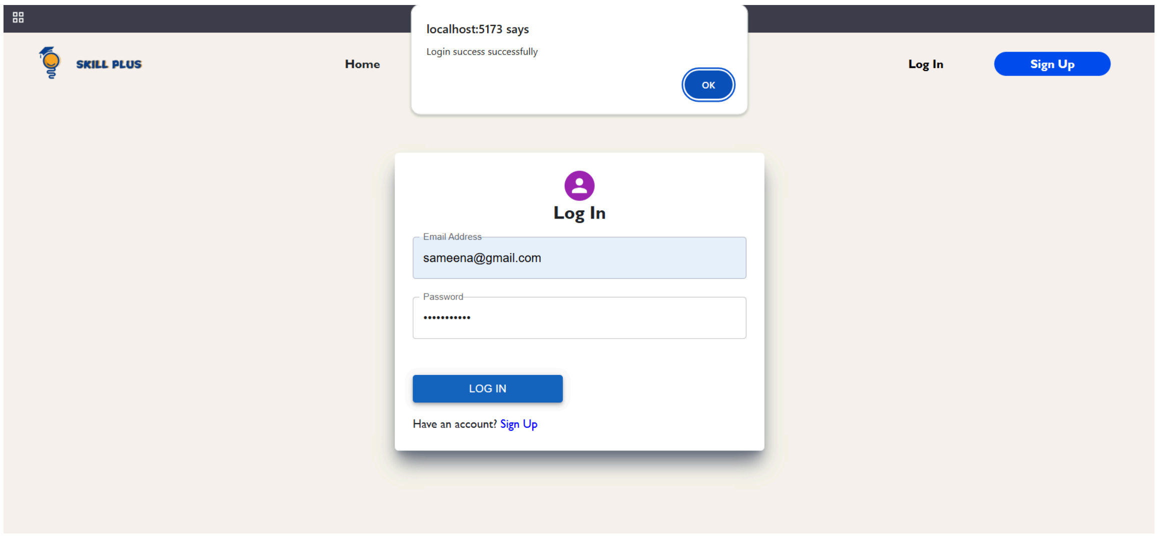
Email Address: mohammedthasreef1@gmail.com

Password: ••••••••••••

Student

SIGN UP

Have an account? Log In



SKILL PLUS   Home   Log In   Sign Up

localhost:5173 says

Login success successfully

OK

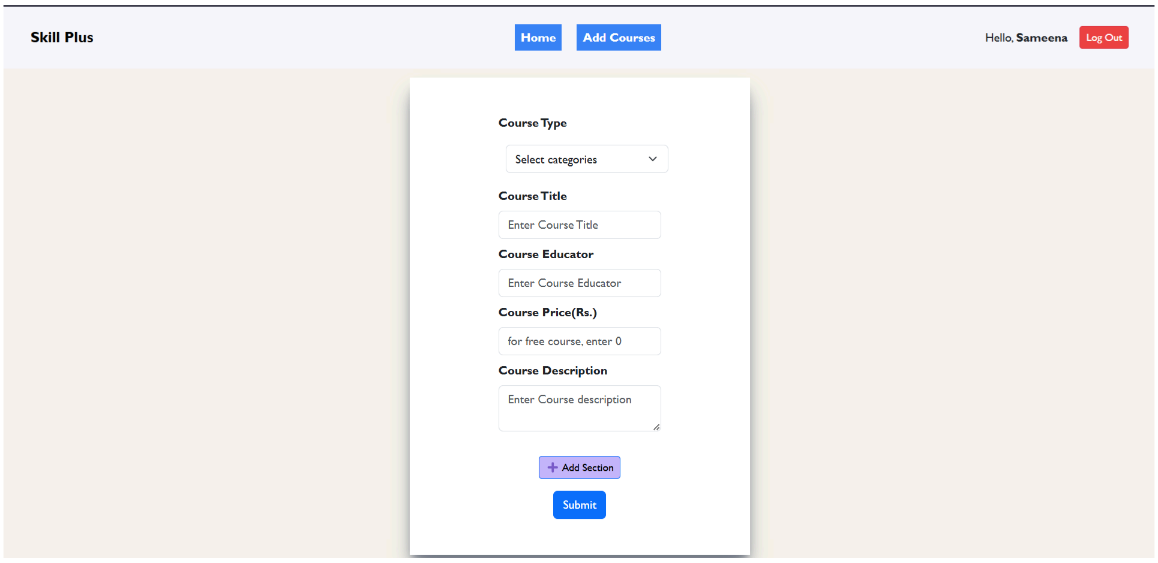
Log In

Email Address: sameena@gmail.com

Password: •••••••••••

LOG IN

Have an account? Sign Up



Skill Plus   Home   Add Courses   Hello, Sameena   Log Out

Course Type: Select categories

Course Title: Enter Course Title

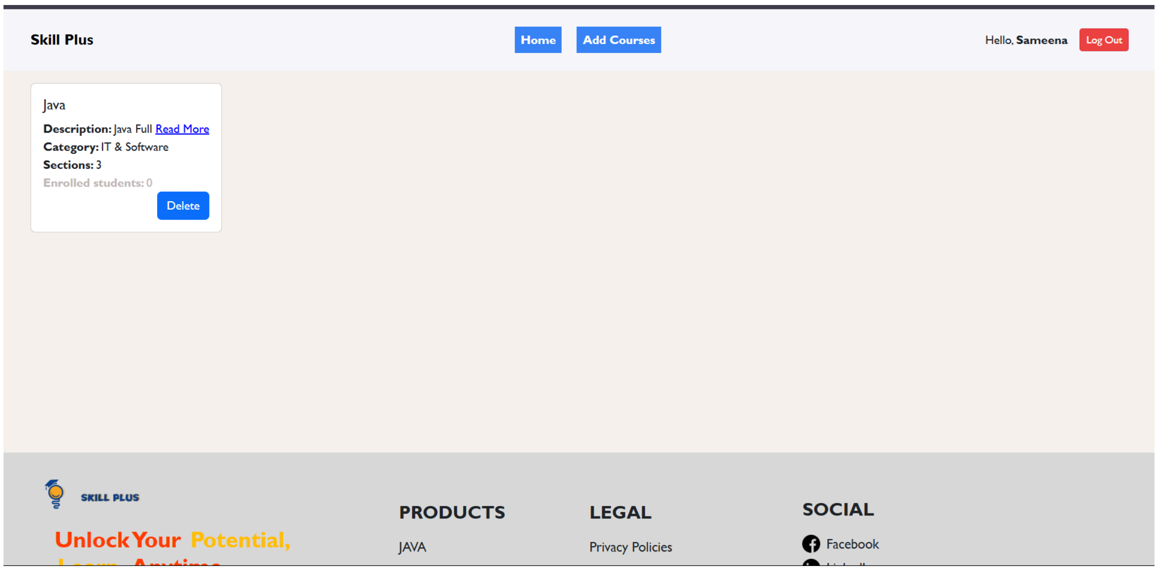
Course Educator: Enter Course Educator

Course Price(Rs.): for free course, enter 0

Course Description: Enter Course description

Add Section

Submit



Skill Plus   Home   Add Courses   Hello, Sameena   Log Out

Java

Description: Java Full Read More

Category: IT & Software

Sections: 3

Enrolled students: 0

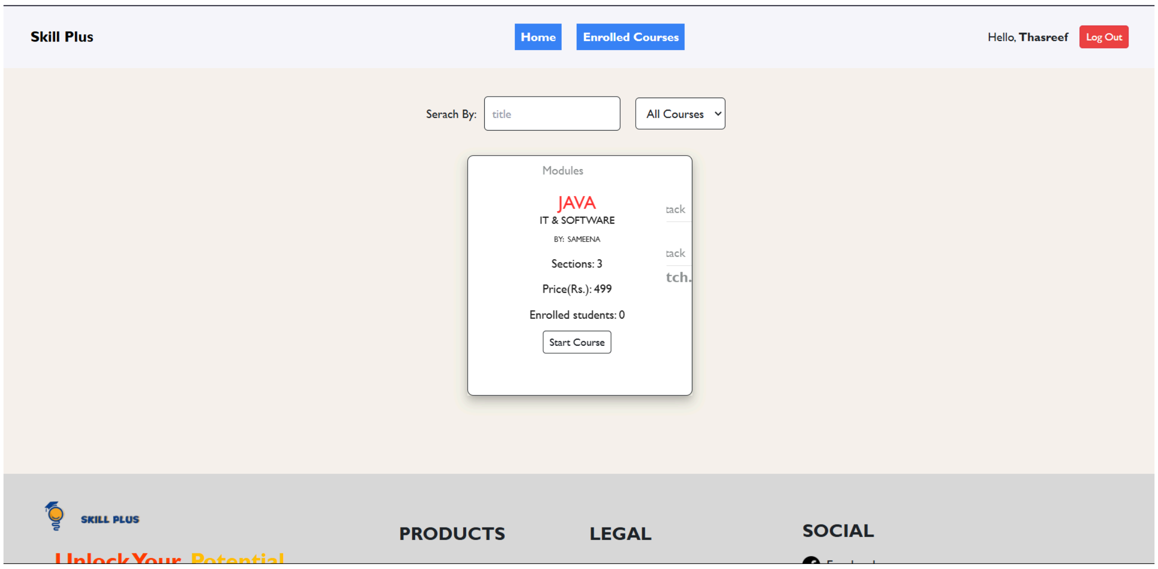
Delete

SKILL PLUS

Unlock Your Potential,

PRODUCTS   LEGAL   SOCIAL

JAVA   Privacy Policies   Facebook


Skill Plus

Home | Enrolled Courses

Hello, Thasreef | Log Out

Serach By: title | All Courses

Modules

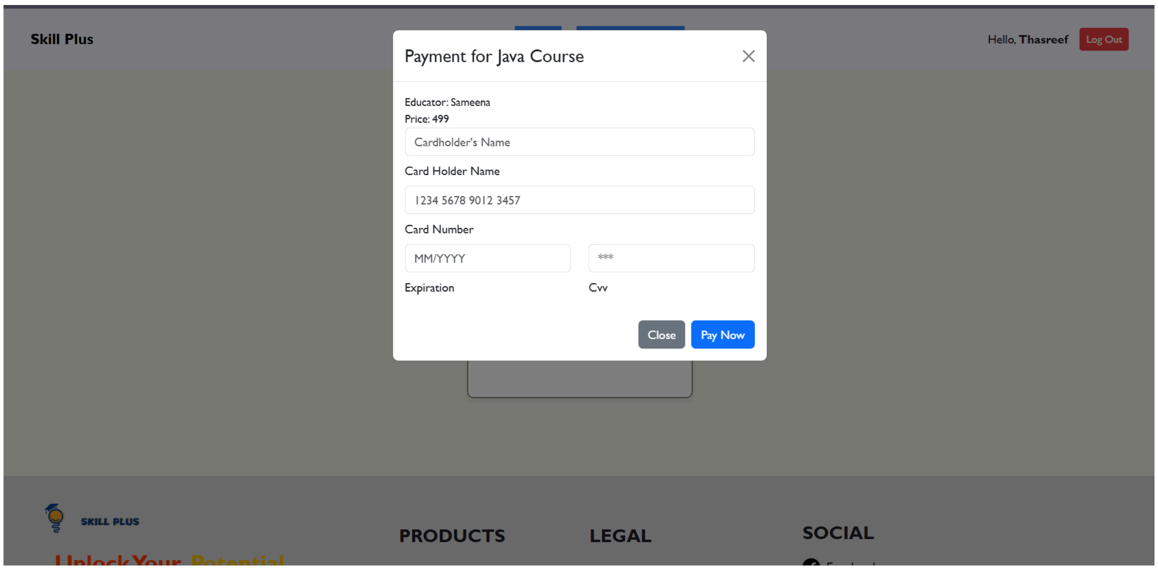
JAVA
IT & SOFTWARE
BY: SAMEENA

Sections: 3

Price(Rs.): 499

Enrolled students: 0

Start Course


Payment for Java Course

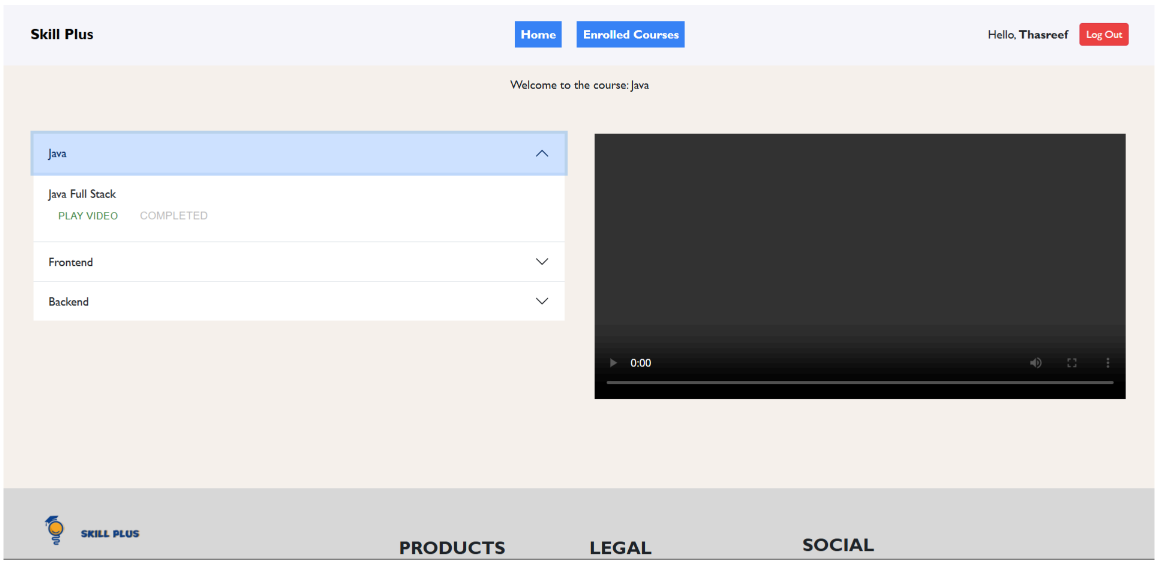
Educator: Sameena
Price: 499

Cardholder's Name
Card Holder Name

1234 5678 9012 3457
Card Number

MM/YYYY | ***
Expiration | Cvv

Close | Pay Now


Skill Plus

Home | Enrolled Courses

Hello, Thasreef | Log Out

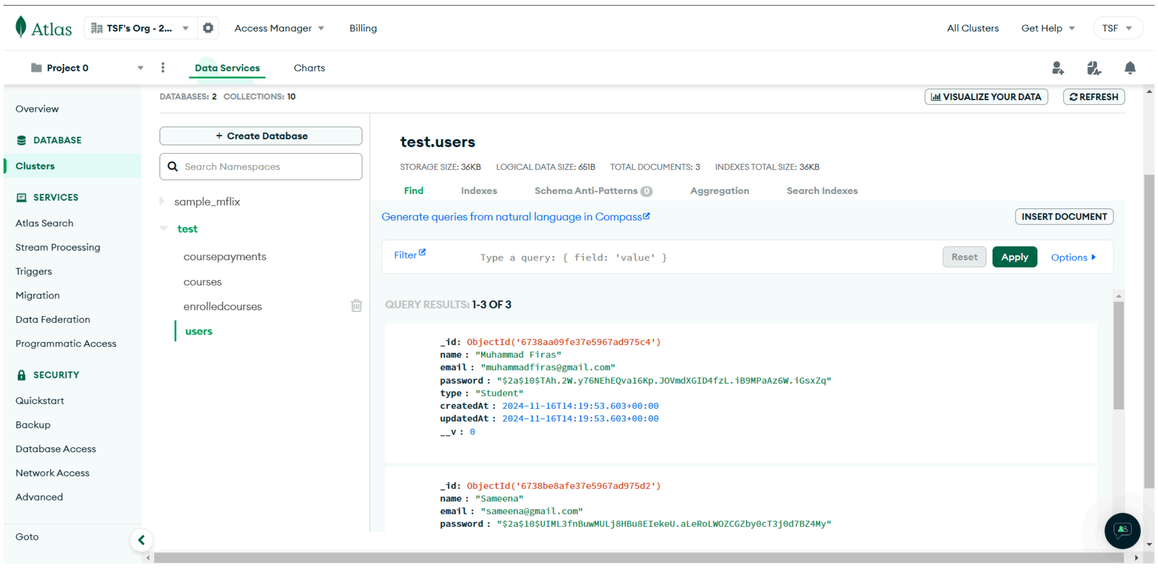
Welcome to the course: Java

Java

Java Full Stack
PLAY VIDEO   COMPLETED

Frontend

Backend



```
_id: ObjectId('6738aa09fe37e5967ad975c4')
name : "Muhammad Firas"
email : "muhammadfiras@gmail.com"
password : "$2a$10$TAh.2W.y76NEhEQva16Kp.JOVmdXGID4fzL.iB9MPaAz6W.iGsxZq"
type : "Student"
createdAt : 2024-11-16T14:19:53.603+00:00
updatedAt : 2024-11-16T14:19:53.603+00:00
__v : 0

_id: ObjectId('6738be8afe37e5967ad975d2')
name : "Sameena"
email : "sameena@gmail.com"
password : "$2a$10$UIML3fnBuwMULj8HBu8EIekeU.aLeRoLWOZCGZby0cT3j0d7BZ4My"
```

10. TESTING:

The testing strategy for the SkillPlus project aims to ensure the robustness, reliability, and performance of both the frontend and backend. The approach includes unit testing, integration testing, and end-to-end testing, with a focus on both functionality and security.

1. Frontend Testing (React)

A. Testing Strategy

- **Unit Testing**:
  Unit tests are written for individual React components to ensure that they work as expected in isolation. This includes verifying that components render correctly based on props and state, and that user interactions trigger the expected behavior.
- **Integration Testing**:
  Integration tests focus on testing how components interact with each other. This includes testing form submissions, API calls, and managing state across components.
- **End-to-End Testing**:
  End-to-end tests ensure that the entire user flow works correctly, from user input to backend communication and frontend rendering. These tests simulate real-world scenarios and verify the overall functionality of the app.

B. Tools Used:

- **React Testing Library**:
  React Testing Library is used in combination with Jest to test React components. It

encourages writing tests that simulate real user interactions rather than testing implementation details.

2. Continuous Integration and Deployment (CI/CD)

The testing process is integrated into the CI/CD pipeline to ensure that tests run automatically with every code change, maintaining high code quality.

- **GitHub Actions**:
  GitHub Actions is used to run automated tests on every push and pull request. This ensures that tests are always executed in a consistent environment, and only passing code gets merged into the main branch.

---

11. DEMO VIDEO LINK:

https://drive.google.com/file/d/1G55NP-UPYWu7FxySv1JOQUuwyhOlzLks/view?usp=sharing

---

12. KNOWN ISSUES

1. Course Title and Description Alignment Issue

- **Description**:
  When a student logs in and registers for a course, the **course title** and **description** are not properly aligned on the course registration page. This misalignment may cause the text to appear in an inconsistent or visually unappealing way, impacting the overall user experience.
- **Steps to Reproduce**:
  1. Log in as a student.
  2. Navigate to the course registration page.
  3. Select and register for a course.
  4. Observe the misalignment of the course title and description on the page.
- **Expected Behavior**:
  The course title and description should be aligned correctly, with proper spacing and formatting, for a neat and organized display.
- **Current Workaround**:
  No specific workaround has been identified at this moment, but the issue will be addressed in an upcoming update.

- **Status**:
  The issue has been acknowledged and is in progress to be resolved in the next release.

---

13. FUTURE ENHANCEMENT:

- **Personalized Learning Experience:** Implementing machine learning algorithms to analyze user behaviour and preferences, allowing for personalized course recommendations and adaptive learning paths. Incorporating features such as progress tracking, quizzes, and assessments to tailor the learning experience to individual needs.
- **Accessibility Features**: Implementing accessibility features such as screen reader compatibility, keyboard navigation, and alternative text for multimedia content to ensure inclusivity and compliance with accessibility standards.
