

MERN stack powered by MongoDB

Project Documentation

Introduction

Project Title: LearnHub

Team Members:

S.No	Student Name	Register No.	Naan Mudhalvan ID	Role
1.	Dhineshkumar M	412721104012	907993F6A474801BD6448EB4374621E5	Backend
2.	Balachandar S	412721104007	A58E5EEAA518FF4BE3743FF4DEA696D0	Frontend
3.	Dayanithi M	412721104009	2E6CCD61AB2AA0286B044A5C1C459B47	Frontend
4.	Praveenkumar M	412721104035	63281C82F9EBA459648408F0CF6E29E9	Frontend

Project Overview

Purpose:

LearnHub is an online learning platform designed to connect students and instructors in a collaborative learning environment. The platform facilitates course creation, student enrollment, and a discussion forum, providing a user-friendly interface for seamless learning.

Features:

- **Student/User:**
 - Signup with email verification and OTP-based password reset.
 - View all available courses and access enrolled courses.
 - Discussion forum for creating or continuing threads.
 - Editable profile except for email.
- **Instructor:**
 - Create, edit, and delete courses.
 - Upload course sections, including videos and descriptions.
 - View enrolled students for their courses.
 - Editable profile except for email.

- **Admin:**
 - View all registered users, instructors, and courses.
 - Delete courses if necessary.
 - Popup view for instructor-specific courses.
- **Exclusions:**
 - No payment gateway (replaced with a dummy page).
 - Admin login not implemented (accessed via route).

Architecture

Frontend:

- Developed using **React**, styled with **Bootstrap**, and built with **Vite** for optimized performance.
- Responsive design ensures usability across devices.

Backend:

- Built using **Node.js** and **Express.js** to handle APIs and business logic.
- Middleware includes:
 - **bcryptjs** for password hashing.
 - **jsonwebtoken** for authentication.
 - **Multer** for file handling.

Database:

- **MongoDB** as the database solution, implemented with **Mongoose** for schema design and data interaction.

Setup Instructions

Prerequisites:

- Node.js
- MongoDB Atlas Account
- Cloudinary account for image and video storage.

Installation:

1. Clone the repository:

```
git clone https://github.com/dk172923/online-learning-platform  
cd online-learning-platform
```

2. Install dependencies:

○ Frontend:

```
cd client  
npm install
```

○ Backend:

```
cd server  
npm install
```

3. Set up environment variables:

- Create **.env** file in the **server** folder with the following:

```
MONGODB_URI='mongodb+srv://dhineshkumar1729:qwertyuiop@cluster0.cwt4c.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0'
```

```
JWT_SECRET='t9843yt8hg0h8y834th893hy89h'
```

```
EMAIL_USER='dhineshkumar24murugan007@gmail.com'
```

```
EMAIL_PASS='szlv xili ndr w qasg'
```

```
CLOUDINARY_CLOUD_NAME='dgosdgcem'
```

```
CLOUDINARY_API_KEY='695861773226449'
```

```
CLOUDINARY_API_SECRET='H7QShPl9ouKwmBTdrmnFihgaIIU'
```

- Create **.env.local** file in the **client** folder with the following:

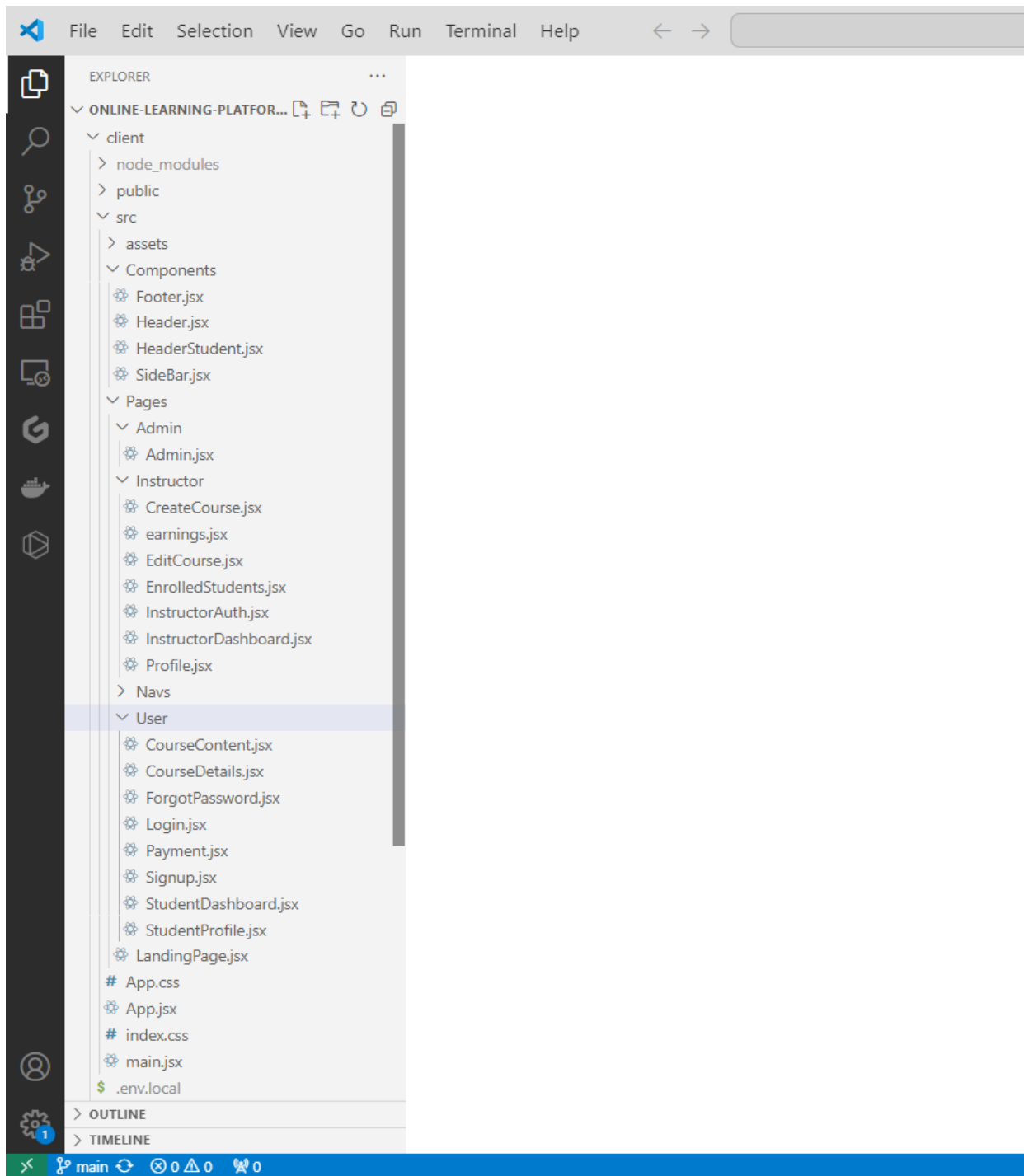
```
VITE_REACT_APP_BACKEND_BASEURL=http://localhost:5000
```

Folder Structure

Client (Frontend):

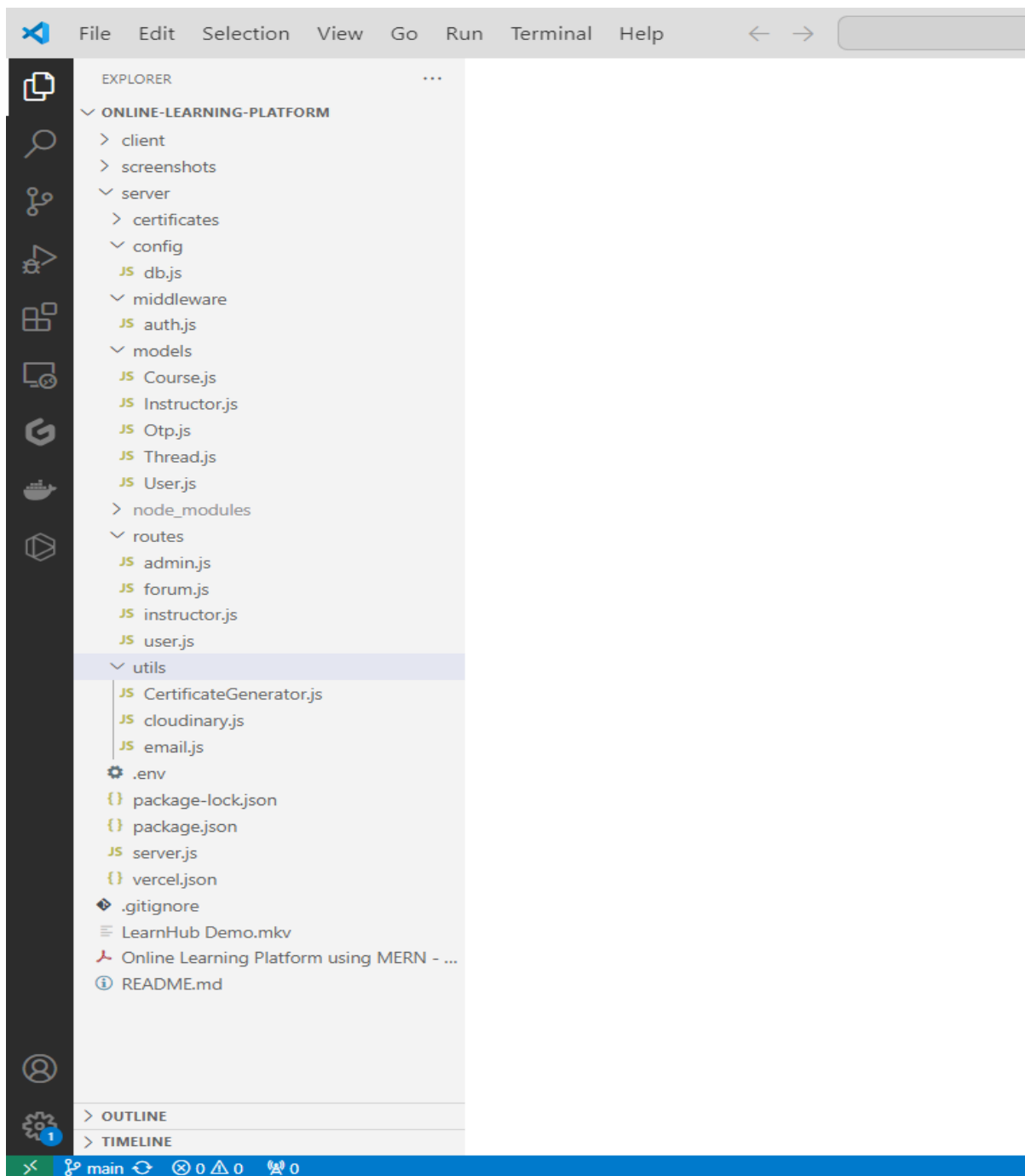
- **src/**: Contains all React components, pages, and assets.
 - **Components/**: Reusable UI components such as Footer, Header, HeaderStudent and SideBar

- **Pages/:** Folders such as Admin, User, Instructor and Navs containing Pages for different aspects of the application.



Server (Backend):

- **routes/:** Defines API routes for users, instructors, forums, and admin.
- **models/:** MongoDB schemas for Users, Courses, Instructor, Thread and OTP.
- **middleware/:** Authentication using jwt token.
- **utils/:** Contains the Cloudinary API and Certificate Generation code.



Running the Application

- **Frontend:**

`cd client`

`npm run dev`

- **Backend:**

`cd server`

`nodemon server.js (or) node server.js`

API Documentation

User Routes:

- **POST /signup:** Register a new user.
- **POST /login:** Login user.
- **GET /verify-email:** To verify the newly registered user.
- **POST /forgot-password:** Send OTP for password reset.
- **POST /verify-otp:** To verify the OTP for changing password.
- **POST /update-password:** Changes the password of the user.
- **PUT /addCourse:** For adding the course to the purchased course list.
- **GET /student/:userEmail:** For fetching the users' profile.
- **PUT /student/:userEmail:** To update the users' profile.
- **POST /upload-profile-image-user:** For updating/uploading the profile image.
- **GET /name:** To fetch the user's name using the email id.
- **GET /student-dashboard/courses:** Gets all the courses of the platform.
- **GET /student-dashboard/my-courses:** Fetch the courses enrolled by the user.
- **GET /course-content:** Get course content by courseId
- **POST /update-progress:** Mark section as completed & update last-viewed section
- **POST /generate-certificate:** To create and download the certificate after course completion.
- **POST /check-course-purchase:** Check whether the user is enrolled or not in a course.
- **GET /courses:** Fetch all courses for header student

- **GET /:email:** Fetch user by email to get purchased courses for header student
- **GET /instructors/:instructorEmail:** To fetch the instructor using email id
- **GET /courses/:courseId:** Fetch the specified course details

Instructor Routes:

- **POST /signup:** Register a new user.
- **POST /login:** Login user.
- **GET /verify-email-instructor:** To verify the newly registered user.
- **POST /forgot-password:** Send OTP for password reset.
- **POST /verify-otp:** To verify the OTP for changing password.
- **POST /update-password:** Changes the password of the user.
- **GET /:instructorEmail:** Fetches the instructor's profile
- **PUT /:instructorEmail:** Updates the instructor's profile
- **POST /upload-profile-image:** Uploads the instructor's profile image
- **GET /courses/:instructorEmail/:courseId/students:** Fetch students enrolled in a specific course, only for the instructor who created the course
- **GET /courses/:instructorEmail:** Fetch all courses taught by a specific instructor
- **POST /courses:** Get courses based on the instructor email id.
- **DELETE /courses/:id:** Delete course Route
- **GET /courses/edit-courses/:id:** Get course by ID Route
- **PUT /courses/:id:** Delete a course by its ID
- **POST /upload-video:** Video upload route using Cloudinary
- **POST /upload-image:** Image upload route using Cloudinary
- **POST /create-course:** To create a new course.

Admin Routes:

- **GET /users:** View all registered users.
- **GET /instructors:** View all registered instructors.
- **GET /courses:** Get all the courses in the platform
- **DELETE /courses/:id:** Delete a specific course.
- **GET /instructors/:email/courses:** To get the courses of any particular instructor.

Forums Routes:

- **GET /:courseId/threads:** Get all messages for a specific course's discussion thread
- **POST /:courseId/threads:** Post a new message to a specific course's discussion thread

Authentication

- **JWT (JSON Web Tokens):**
 - Tokens generated on successful login.
 - Middleware checks token validity for protected routes.
- **Email Verification:**
 - Verification link sent during signup using **Nodemailer**.
- **OTP-based Password Reset:**
 - OTP sent via email for secure password reset.

User Interface

- **Student Dashboard:** Displays enrolled courses, profile management, and discussion forum.
- **Instructor Dashboard:** Course creation tools and enrolled students' view.
- **Admin View:** User, instructor, and course management in a streamlined UI.

Testing

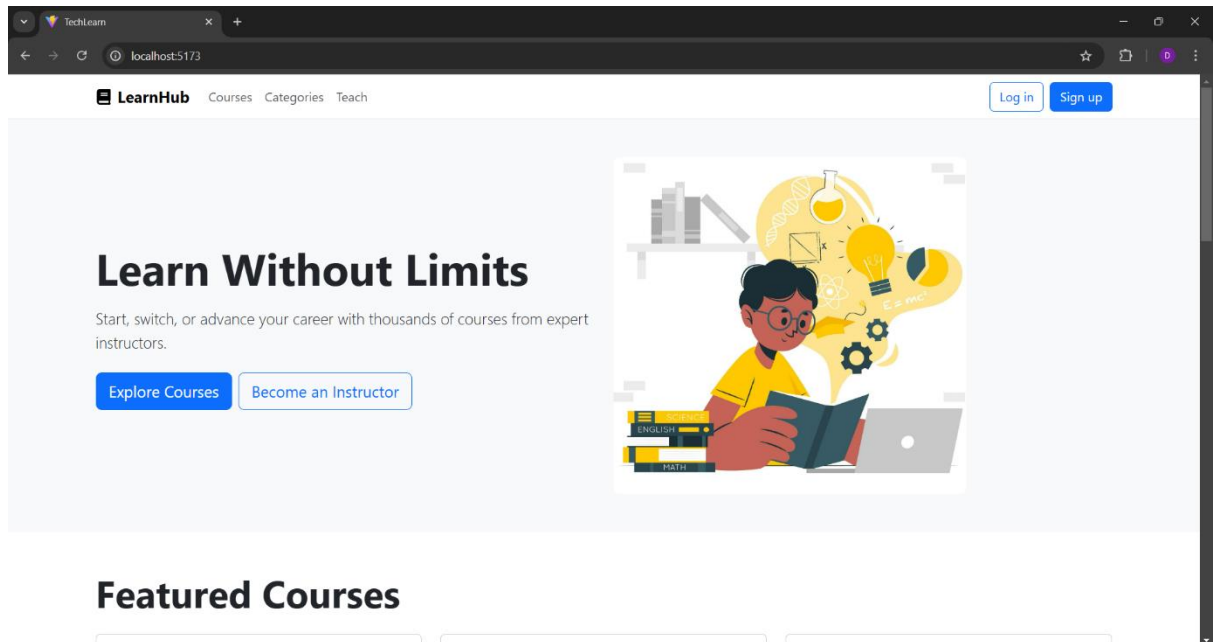
- **Manual Testing:** Verified all major functionalities including user signup, course creation, and discussion threads.
- **Postman:** Used for API endpoint testing.

Screenshots or Demo

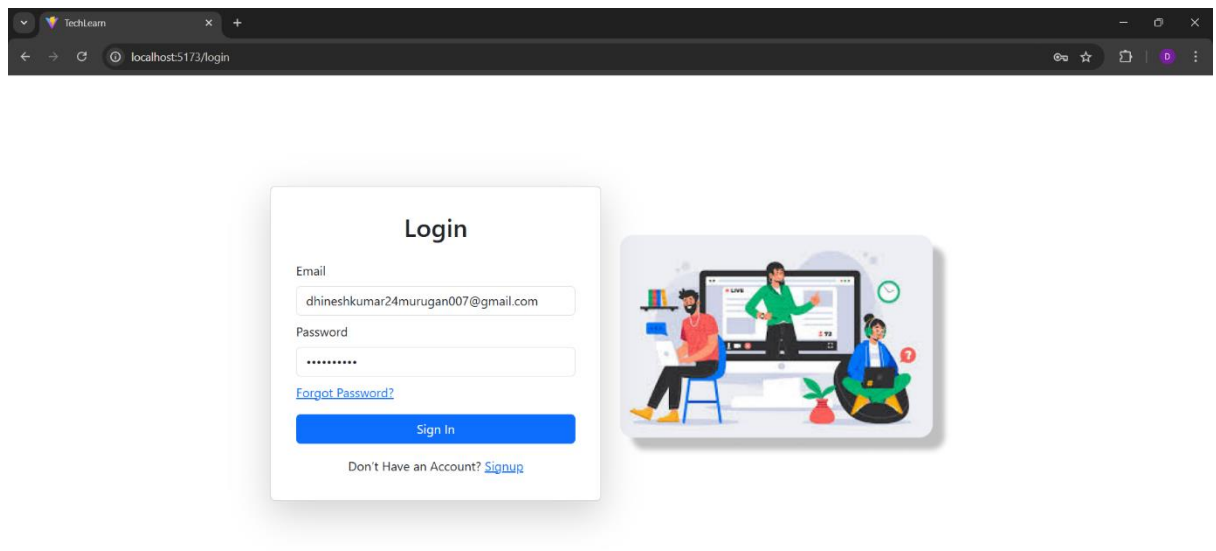
- **Demo Link:** <https://drive.google.com/file/d/1Uxsi-6nVnCJHkIMTg7CFQtv2fTrNtNdR/view?usp=sharing>

- Screenshots:

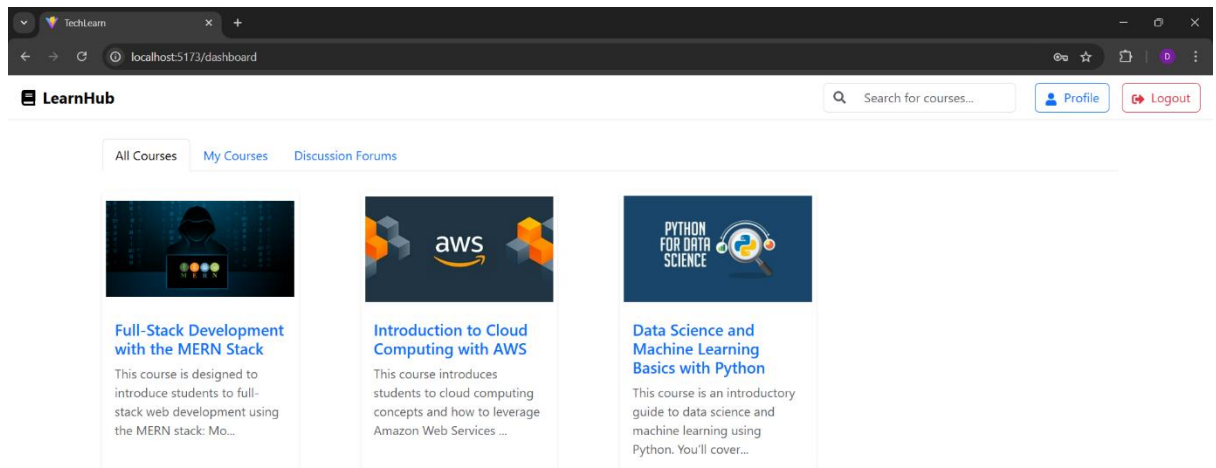
1. Landing Page



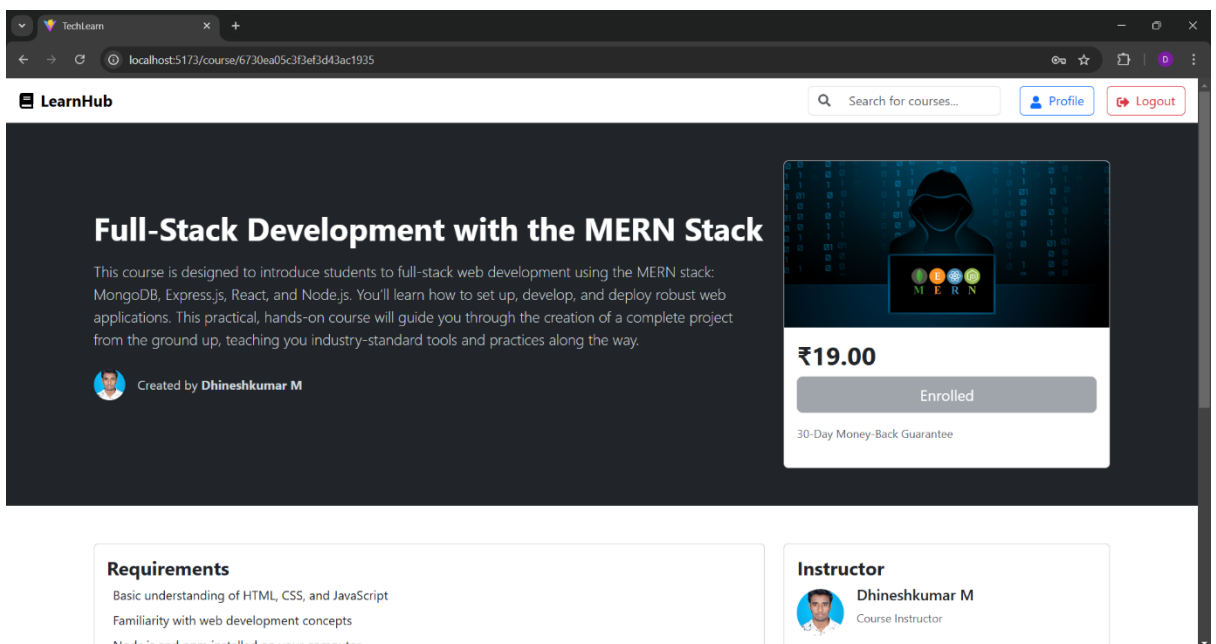
2. User Login



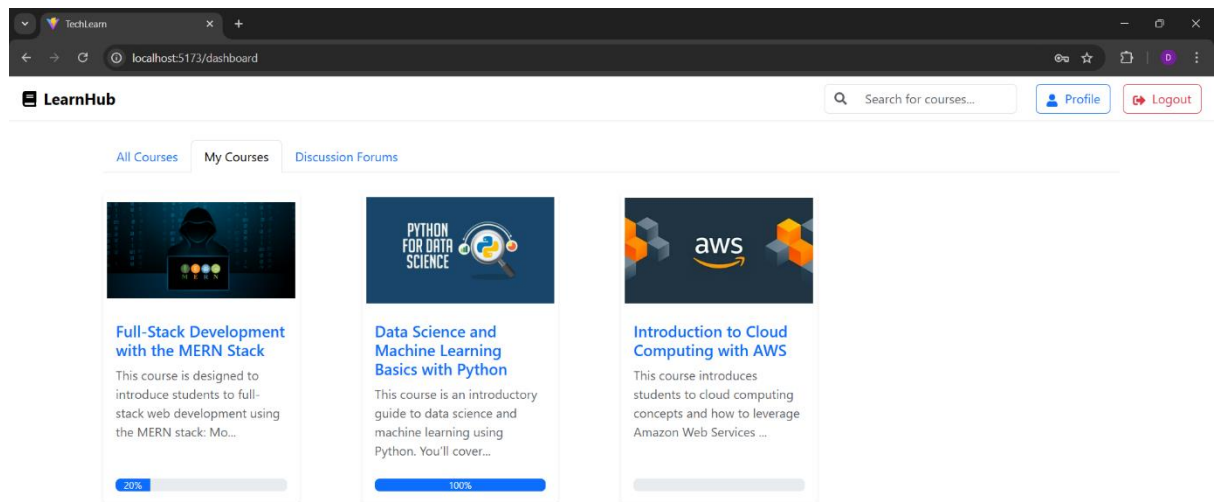
3. User Dashboard



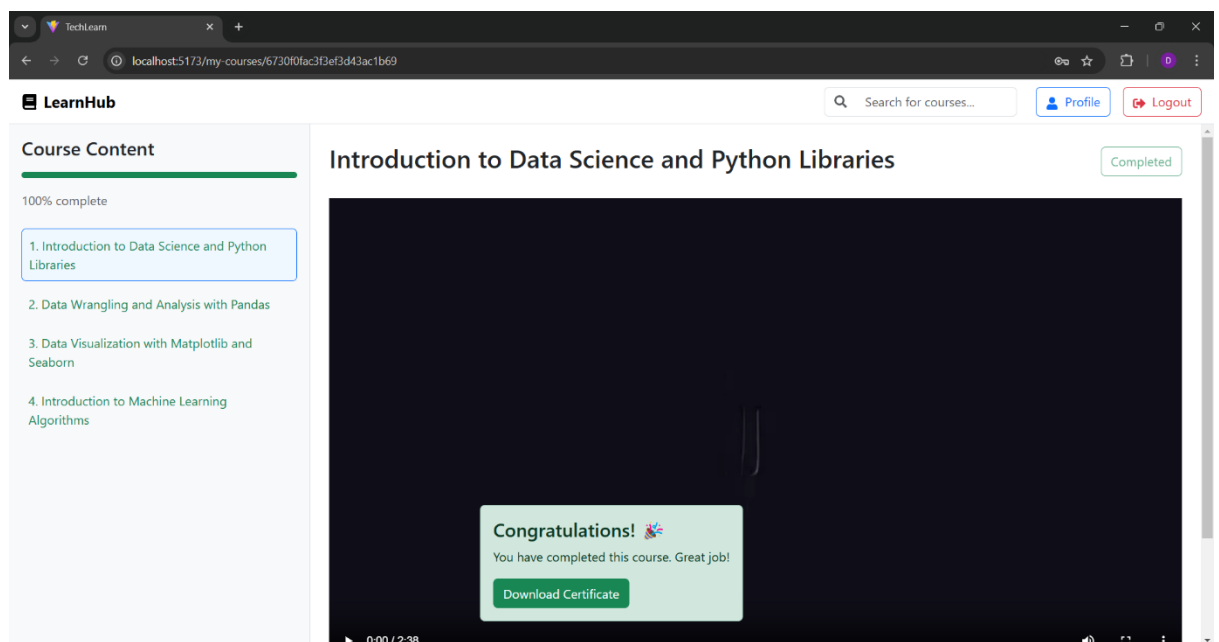
4. User Course Details



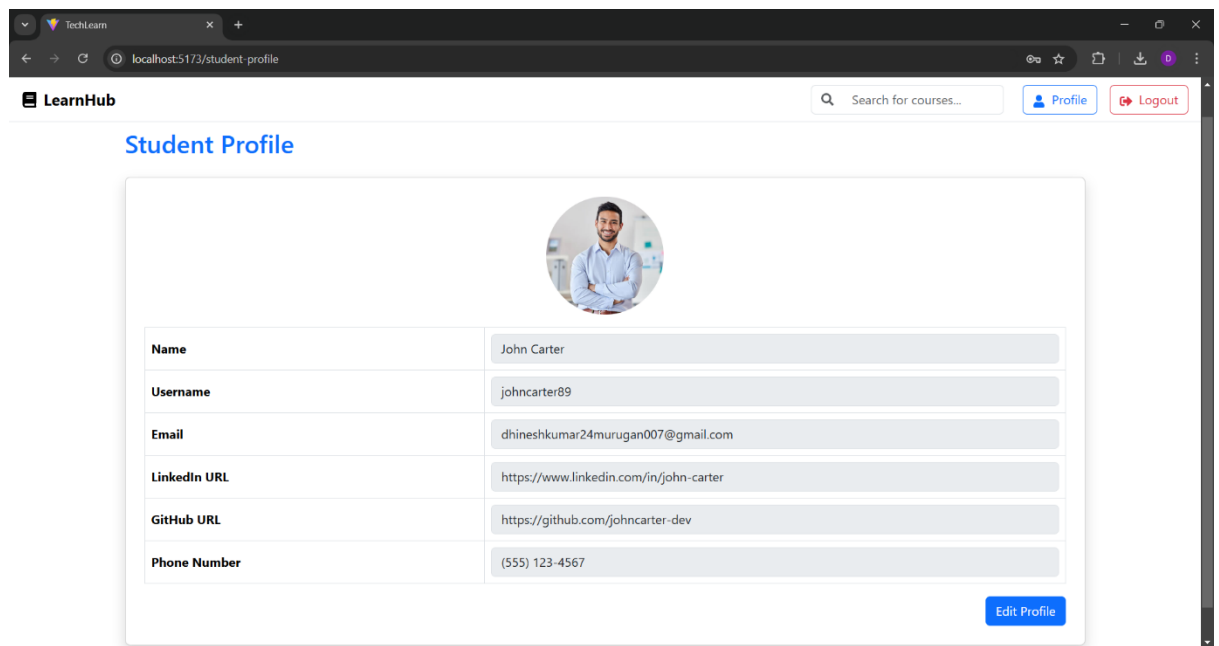
5. User My Courses Page



6. User Course Access



7. User Profile Page

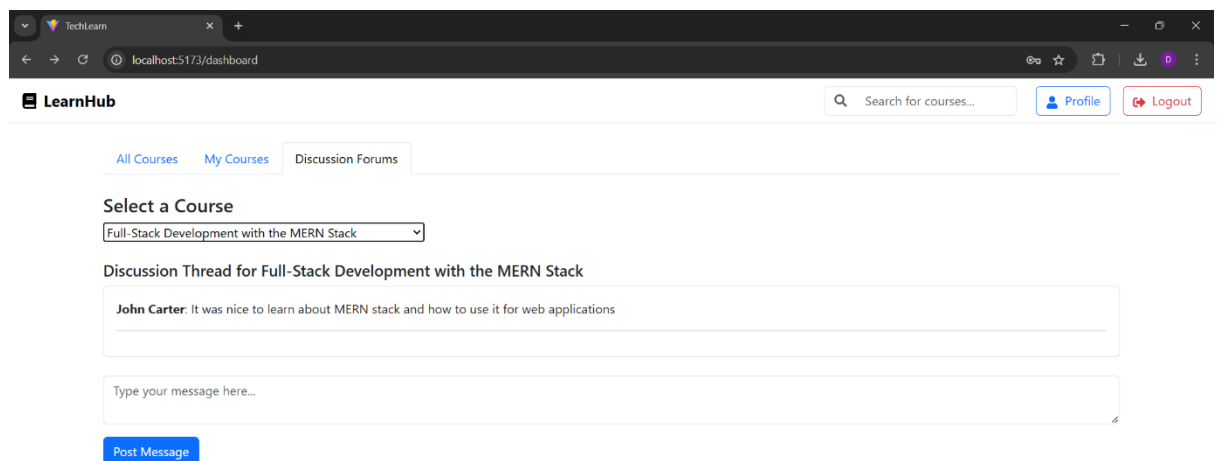


The screenshot shows a web browser window with the URL `localhost:5173/student-profile`. The page is titled "Student Profile" and features a user profile card. The card includes a circular profile picture of a man with arms crossed. Below the picture is a table with the following details:

Name	John Carter
Username	johncarter89
Email	dhineshkumar24murugan007@gmail.com
LinkedIn URL	https://www.linkedin.com/in/john-carter
GitHub URL	https://github.com/johncarter-dev
Phone Number	(555) 123-4567

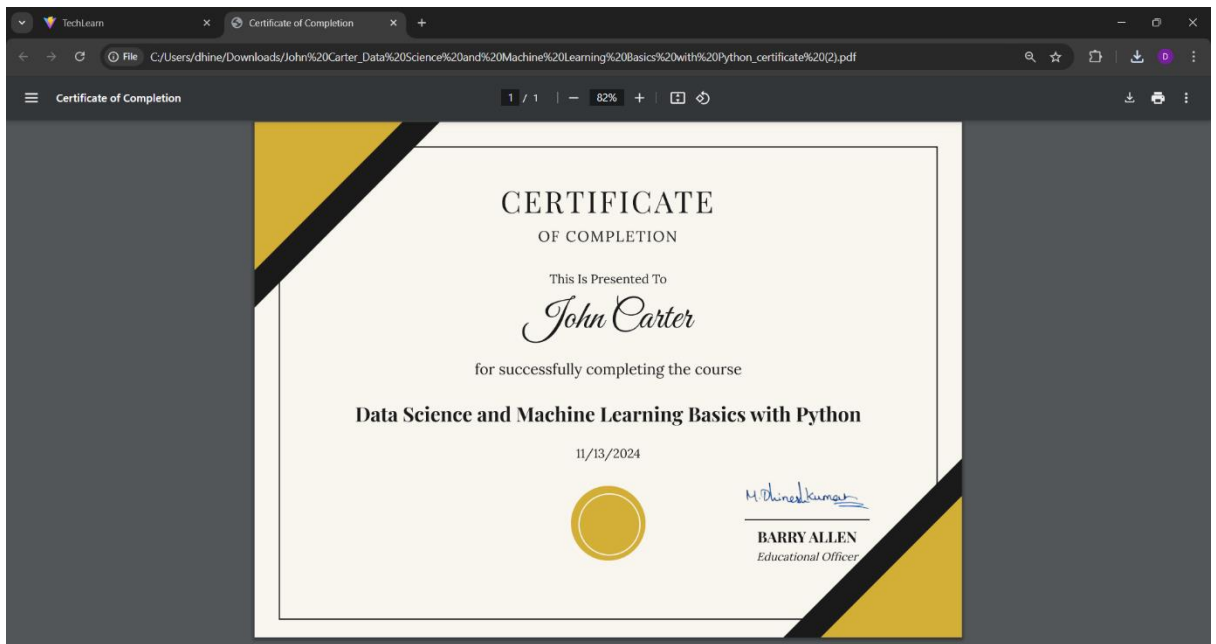
An "Edit Profile" button is located at the bottom right of the profile card. The browser's top bar shows the "LearnHub" logo, a search bar, and "Profile" and "Logout" buttons.

8. User Discussion Forums



The screenshot shows a web browser window with the URL `localhost:5173/dashboard`. The page is titled "Discussion Forums" and features a "Select a Course" dropdown menu. The selected course is "Full-Stack Development with the MERN Stack". Below the dropdown is a "Discussion Thread for Full-Stack Development with the MERN Stack". The thread shows a message from "John Carter" stating: "It was nice to learn about MERN stack and how to use it for web applications". Below the thread is a text input field with the placeholder "Type your message here..." and a "Post Message" button. The browser's top bar shows the "LearnHub" logo, a search bar, and "Profile" and "Logout" buttons.

9. User Certificate



10. Instructor Login



[Login](#) [Sign Up](#)

Login

Email

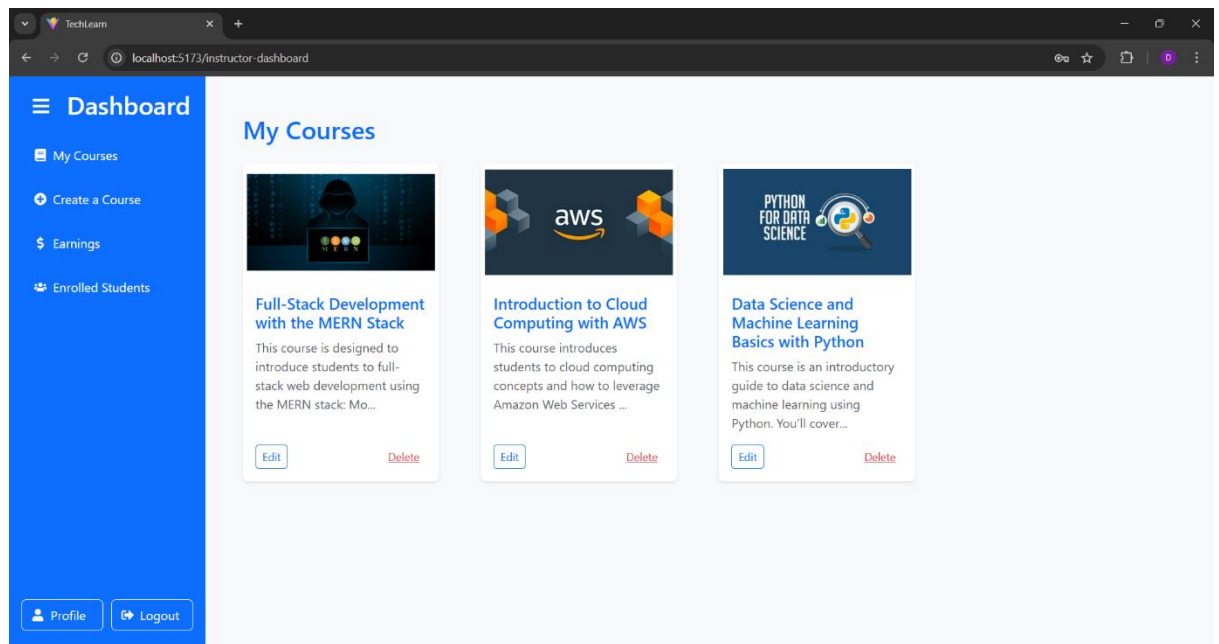
Password

[Sign In](#)

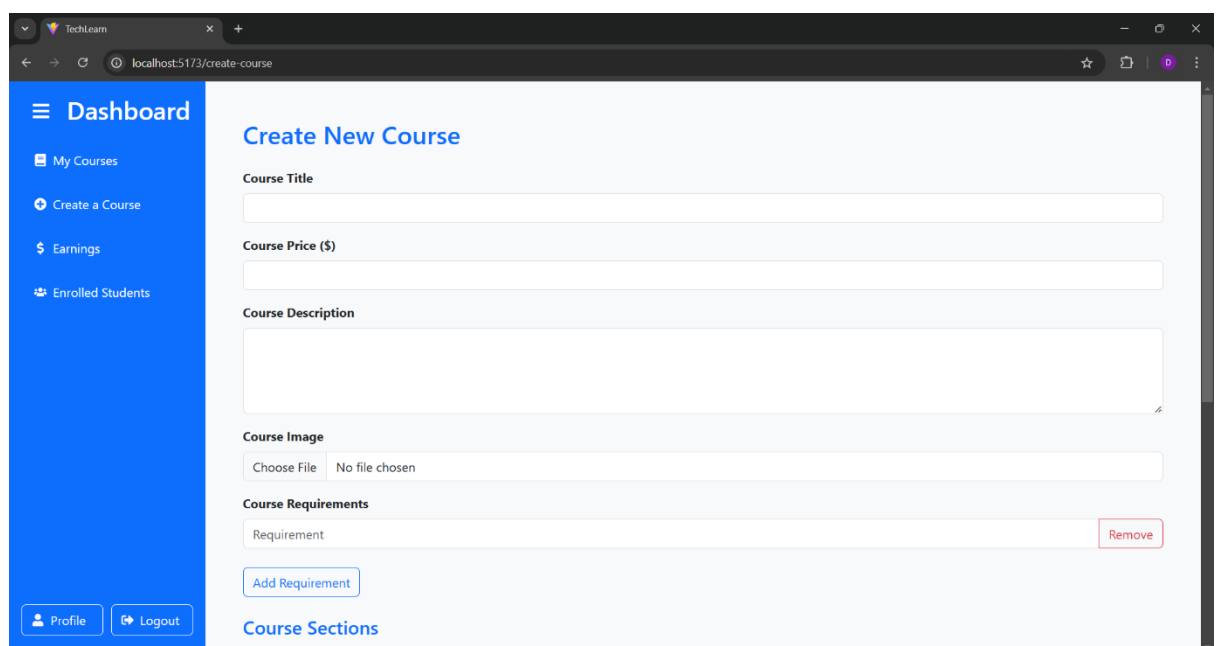
Don't Have an Account? [Signup](#)

[Forgot Password?](#)

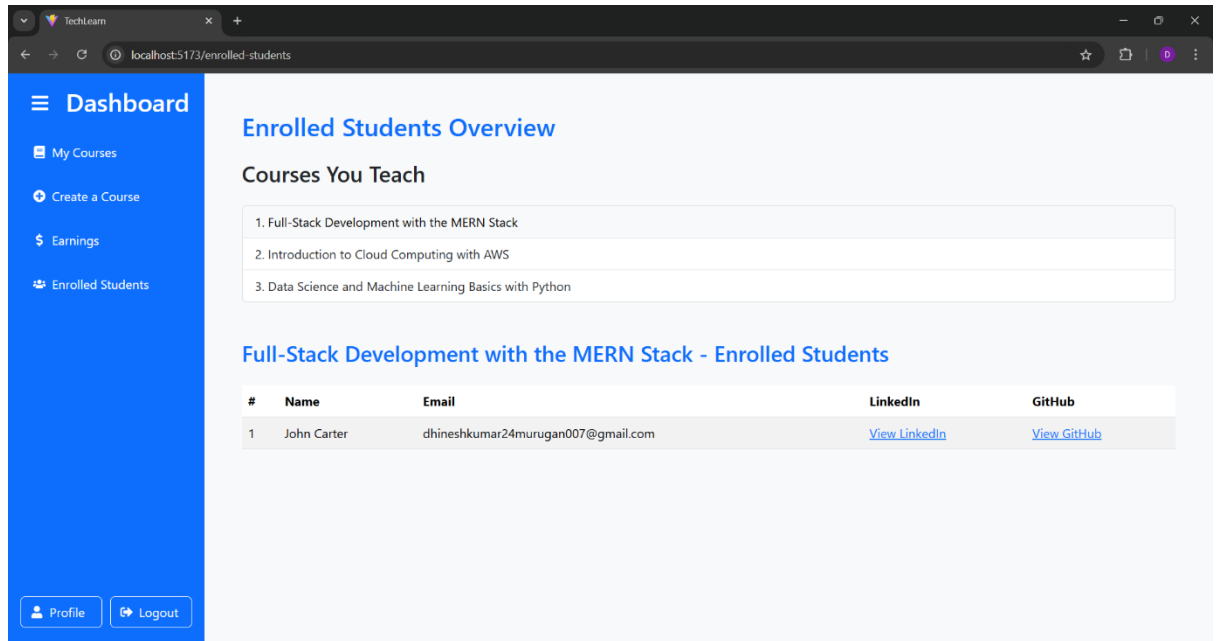
11. Instructor Dashboard



12. Instructor Course Creation



13. Instructor Enrolled Students



TechLearn | Dashboard | My Courses | Create a Course | Earnings | Enrolled Students

Enrolled Students Overview

Courses You Teach

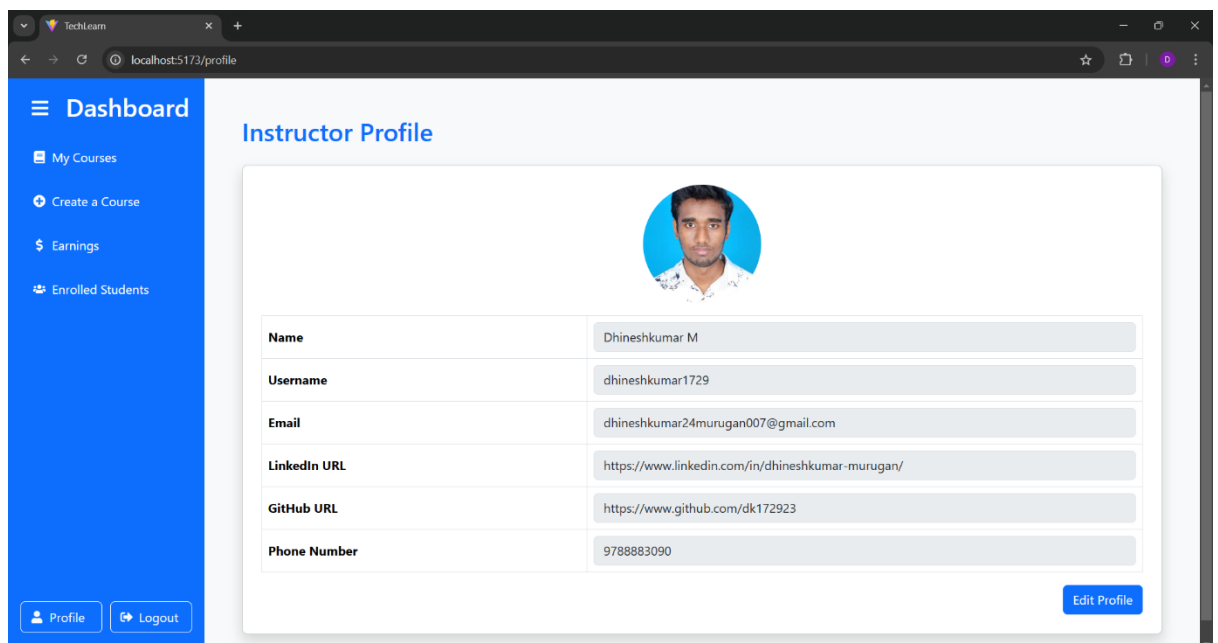
- Full-Stack Development with the MERN Stack
- Introduction to Cloud Computing with AWS
- Data Science and Machine Learning Basics with Python

Full-Stack Development with the MERN Stack - Enrolled Students

#	Name	Email	LinkedIn	GitHub
1	John Carter	dhineshkumar24murugan007@gmail.com	View LinkedIn	View GitHub


[Profile](#) | [Logout](#)

14. Instructor Profile Page



TechLearn | Dashboard | My Courses | Create a Course | Earnings | Enrolled Students

Instructor Profile



Name	Dhineshkumar M
Username	dhineshkumar1729
Email	dhineshkumar24murugan007@gmail.com
LinkedIn URL	https://www.linkedin.com/in/dhineshkumar-murugan/
GitHub URL	https://www.github.com/dk172923
Phone Number	9788883090

[Edit Profile](#)

[Profile](#) | [Logout](#)

15. Admin Dashboard – 1

TechLearn

localhost:5173/admin

Admin Dashboard

User Management

Users List

Name	Email	Progress	Actions
John Carter	dhineshkumar24murugan007@gmail.com	5 Courses	<button>View Details</button>

Instructor Management

Instructors List

Name	Email	Username	Courses
Dhineshkumar M	dhineshkumar24murugan007@gmail.com	dhineshkumar1729	<button>View Courses</button>

Course Management

16. Admin Dashboard – 2

TechLearn

localhost:5173/admin

Admin Dashboard

User Management

Users List

Name	Email	Progress	Actions
John Carter	dhineshkumar24murugan007@gmail.com	5 Courses	<button>View Details</button>

Instructor Management

Instructors List

Name	Email	Username	Courses
Dhineshkumar M	dhineshkumar24murugan007@gmail.com	dhineshkumar1729	<button>View Courses</button>

Course Management

Known Issues

- Admin lacks a dedicated login functionality but only a route.
- No robust error handling for invalid uploads.
- No payment since, the instructor of the course, told us not to include any payment gateway.
- Doesn't fetch the image, video names while editing the profile or courses.

Future Enhancements

- Add payment gateway for course enrollment.
- Implement advanced analytics for admin insights.
- Include a recommendation engine for personalized course suggestions.
- Integrate WebRTC for live webinars