SRI MUTHUKUMARAN INSTITUTE OF TECHNOLOGY

Chikkarayapuram, Mangadu, Chennai-600069

ONLINE LEARNING PLATFORM USING MERN

A PROJECT REPORT

Submitted by

HARIHARAN M.L - 212621104021
GURU PRASATH N - 212621104018
KAVIYA G - 212621104034
ALLENROYAN A - 212621104301
FRANKLIN A - 212621104305

In a partial fulfillment for the award of the degree

Of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

In association with **Naan Mudhalvan**, **Smart Internz** , **Mongo DB**

BONAFIDE CERTIFICATE

Certified that this project report "**ONLINE LEARNING PLARFORM USING MERN**" is the Bonafide work of **HARIHARAN M.L (212621104021), GURU PRASATH N (212621104018), KAVIYA G (212621104034), ALLENROYAN A (212621104301), FRANKLIN A (212621104305)** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Mrs.S.P.AUDLINE BEENA, M.E,	Dr.D.RAJINIGIRINATH, M.Tech, Ph.D.,
Naan Mudhalvan Mentor,	**Head of the Department,**
Assistant Professor.	Professor.
Department of Computer Science and Engineering	Department of Computer Science and Engineering
Sri Muthukumaran Institute of Technology, Chikkarayapuram, Chennai-600069	Sri Muthukumaran Institute of Technology Chikkarayapuram, Chennai-600069

Submitted for the Anna University Naan Mudhalvan Project Viva Voce held on_____

Internal Examiner **External Examiner**

ACKNOWLEDGEMENT

I express my heartfelt gratitude to the Honorable Chief Minister of Tamil Nadu. **Thiru. M.K. Stalin**, for launching and championing the visionary Naan Mudhalvan initiative. This transformative program has provided countless opportunities for students like me to develop essential skills and pave the way for a brighter future.

I am immensely grateful to our Chairperson & Managing Trustee, **Mrs. Gomathi Radhakrishnan**, for her encouragement and guidance. I would also like to extend my sincere gratitude to **Dr. Gautham Srinivas**, Chairman, and **Dr. Arvind Srinivas,** Managing Director, for their unwavering support and motivation, which played a significant role in the successful completion of my project.

We express our deepest gratitude to our respectable Head of the department, **Dr.D.Rajinigirinath, Professor/CSE**, for his valuable support and guidance, which kept me focused and committed throughout the project.

A special thanks to my Naan Mudhalvan mentor, **Mrs. S.P AUDLINE BEENA, Assistant Professor/CSE**, for her consistent guidance, encouragement, and constructive feedback, which were instrumental in enhancing my learning experience.

Lastly, I would like to thank all the staff members and lab-in-charge for their timely assistance and support, which contributed to the smooth progression of my project.

Thank you to everyone who contributed to making this project work a memorable and valuable learning experience.

INDEX

INTRODUCTION

PROJECT NAME: ONLINE LEARNING PLATFORM USING MERN

TEAM MEMBERS:

MEMBERS	ROLES
HARIHARAN M.L (TL)	BACKEND WORKS
GURU PRASATH N	FRONTEND WORKS
KAVIYA G	DATABASE WORKS
ALLENROYAN A, FRANKLINA	TESTING

PROJECT OVERVIEW

PURPOSE:

The primary objective of this project is to develop a comprehensive and user-friendly online learning platform that seamlessly connects teachers and learners. This platform aims to simplify the process of finding and enrolling in courses while ensuring efficient learning management. By integrating advanced features and an intuitive user interface, the project focuses on enhancing the experience for students, educators, and administrators alike.

FEATURES:

The platform offers a wide range of features:

- **Course Catalog:** Extensive selection of courses across various categories.
- **User Interface:** Intuitive, responsive, and accessible design.
- **Payment Options:** Secure payment gateway supporting multiple options.
- **Support:** Comprehensive support via email, phone, and live chat.

1.User Authentication and Authorization

- Secure login and registration for customers.
- Password encryption and recovery options for enhanced security.
- Rolebased access to ensure secure data management for admins and restaurant owners.

2.Course Enrolment and Progress Tracking

- Real-time progress tracking for students, including milestones and completion status.
- Easy course enrollment process with multiple payment options.

3.Dedicated Teacher Portal

- Exclusive login portal for teachers to manage their courses, schedules, and profiles.
- Access to student feedback, course analytics, and performance insights.

4.Admin Panel for Management

- Comprehensive admin panel to efficiently manage users, courses, and transactions.
- Features to approve new teacher registrations and resolve disputes.
- Tools for creating promotional campaigns and platform-wide announcements.

5.Flexible Learning Options

- Support for self-paced and live courses to cater to different learning styles.
- Night mode option for the platform interface to ensure a comfortable user experience during late hours.

ARCHITECTURE

FRONTEND:

The frontend of the application is designed to offer a visually appealing, responsive, and userfriendly interface. Built with modern technologies, it ensures an engaging user experience and seamless interactivity across various devices. Key components of the frontend include:

1.React.js

- Utilized as the primary library for building the user interface.
- Offers a componentbased architecture that ensures modular, reusable, and maintainable code.
- Enables dynamic updates without reloading the page, enhancing the overall performance and responsiveness of the application.

2.Styling with Tailwind CSS and Bootstrap

- Tailwind CSS: Used for rapid UI development with its utilityfirst design approach. This allows for consistent styling and flexibility in design implementation.
- Bootstrap: Incorporated for its predesigned components and responsive grid system, ensuring crossbrowser compatibility and mobile responsiveness.

3.**Framer Motion for Animations**

- Adds smooth and visually appealing animations to elevate user interaction.

➢ Features animations like fade ins, slide effects, and transitions for buttons, modals, and page elements, making the interface more interactive and dynamic.

BACKEND:

The backend of the platform ensures robust and scalable serverside functionality to handle requests efficiently and manage business logic. Key technologies include:

1.Node.js

➢ A highperformance JavaScript runtime that handles concurrent requests efficiently, making it ideal for realtime applications like food delivery.
➢ Its nonblocking I/O model ensures quick data processing and responsiveness.

2.Express.js Framework

➢ A lightweight and flexible web application framework for Node.js that simplifies backend development.
➢ Used to implement RESTful API architecture for structured and consistent communication between the frontend and the database.
➢ Handles routing, middleware, and error management efficiently.

3.RESTful API Architecture

➢ Enables smooth communication between the clientside and serverside components.
➢ API endpoints are created for user authentication, restaurant management, order processing, and realtime updates, ensuring secure and efficient data handling.

DATABASE

The database layer is designed to store and manage application data securely and efficiently. Using MongoDB and Mongoose ensures flexibility and scalability for handling complex data.

1.MongoDB

- A NoSQL, documentoriented database that stores data in a JSONlike format, offering flexibility in schema design.
- Supports high scalability and performance, making it suitable for applications with dynamic and growing data requirements.
- Ensures efficient indexing and querying for faster data retrieval.

2.Mongoose

- Provides a schemabased solution for modeling application data.
- Simplifies interaction with MongoDB through a welldefined structure, ensuring data consistency and validation.
- Offers advanced features like hooks, middleware, and population, making data management more efficient.

3. DBconnection.js

```
 const mongoose = require('mongoose');

const connectDB = async () => {

  try {

    await mongoose.connect(process.env.MONGODB_URI, {

     useNewUrlParser: true,

     useUnifiedTopology: true,

    });

    console.log('MongoDB connected successfully');

  } catch (err) {

    console.error('MongoDB connection error:', err);

    // Exit process with failure

    process.exit(1);

  }

};


module.exports = connectDB
```

FOLDER ARCHITECTURE

CLIENT:

```
├── src/
│   ├── components/
│   ├── context/
│   ├── pages/
│   ├── styles/
│   ├── App.js
│   └── index.js
└── package.json
```

SERVER:

```
server/
├── api/
├── models/
├── routes/
├── middleware/
├── config/
└── package.json
```

RUNNING THE APPLICATION

To run the project locally, both the frontend and backend need to be set up and executed individually. Follow the steps below to initialize and start each component of the project.

FRONTEND SETUP AND EXECUTION
The frontend is responsible for the user interface and client side interactions.

1.Navigate to the Frontend Directory

- Open your terminal or command prompt.
- Change to the frontend directory by running: **cd src**

2.Install Dependencies

- Ensure all required packages are installed. Run the following command:
- **npm install.**
- This command will download and install all the dependencies listed in the package.json file.

3.Start the Frontend Server

- Once dependencies are installed, start the frontend server using: **npm start**

- This will launch the React development server, typically running on http://localhost:3000.

- Open this URL in your browser to access the application's frontend interface.

4.Frontend Hot Reloading

➢ Any changes made to the source code will automatically reflect in the browser, thanks to React's hot reloading feature.

BACKEND SETUP AND EXECUTION

The backend handles server side logic, database interactions, and API endpoints.

1.Navigate to the Backend Directory

➢ Open a new terminal or command prompt.

➢ Change to the backend directory by running: **cd server**

2.Install Dependencies

➢ Install the required Node.js packages by running: npm install

➢ This will ensure that all backend dependencies specified in the package.json file are available.

3.Start the Backend Server

➢ Start the backend server with the following command:npm start

➢ The server will typically run on http://localhost:5000 or any port specified in the environment configuration.

4.Environment Variable

Ensure that the backend has access to the required environment variables. Create a .env file in the backend directory, including configurations such as the database connection string and API keys.

5.Backend API Testing

Use tools like Postman or URL to test API endpoints during development.

API DOCUMENTATION

The API documentation outlines the key endpoints available in the food delivery platform. These endpoints allow for authentication, restaurant management, order processing, and administrative functions. Each endpoint is designed to interact seamlessly with the platform's frontend and backend, ensuring secure and efficient communication.

Authentication

1. Register a New User

Endpoint: POST /api/auth/register
Description: Registers a new user by creating an account.

Request Body:

```
{
  "name": "John Doe",
  "email": "john@example.com",
  "password": "password123"
}
```

Response:

Success: {
  "message": "User registered successfully",
  "userId": "123456"
}

Error: {
"error": "Email already exists"
}

2. User Login

Endpoint: POST /api/auth/login
Description: Authenticates a user and provides a JWT token for further requests.

Request Body:

{
"email": "john@example.com",
"password": "password123"
}

Response:

Success: {
"token": "jwttoken",
"userId": "123456"
}
Error: {
"error": "Invalid email or password"
}

COURSES

1. Get All Courses

Endpoint: GET /api/courses
Description: Retrieves a list of all courses available on the platform.

Response:

[{ "id": "1",

"title": "Introduction to Python",

"category": "Programming",

"price": 49.99, "enrolled_students": 150 },

{ "id": "2",

"title": "Graphic Design Basics",

"category": "Design",

"price": 0.0,

"enrolled_students": 200 }]

2. Get a Specific Course

Endpoint: GET /api/courses /:id
Description: Fetches detailed information about a specific course.

Response:

{
"id": "1",
"name": "Introduction to Python",
"category": "Programming",
"price": "49.99",
"rating": 4.5
}

Add a Course (Teacher Only)

Endpoint: POST /api/teacher/courses
Description: Allows teachers to add new courses to the platform.

Request Body:

```
{
  "title": "Advanced JavaScript",
  "category": "Programming",
  "price": 99.99,
  "description": "Master JavaScript with advanced concepts."
}
```

Response:

```
{
  "id": "3",
  "message": "Course added successfully."
```

4. Delete a Course (Teacher Only)

Endpoint**:** DELETE /api/teacher/courses/:id
Description**:** Allows teachers to delete a course if no students are enrolled or for other reasons.

Response:


```
{
  "message": "Course deleted successfully."
}
```


5. Add Sections to a Course (Teacher Only)

Endpoint**:** POST /api/teacher/courses/:id/sections
Description**:** Allows teachers to add sections to their courses.

Request Body:


```
{
  "title": "Loops in Python",
  "content": "Learn about loops such as for and while loops."
}
```


Response:


```
{
  "section_id": "201",
  "message": "Section added successfully."
}
```


6. Enroll in a Course (Student Only)

Endpoint**:** POST /api/student/courses/:id/enroll
Description: Enables students to enroll in a course. For paid courses, payment is required before enrollment.

Response:


```
{
  "message": "Enrollment successful.",
  "certificate_available": false
}
```


7. Resume a Course (Student Only)

Endpoint**:** GET /api/student/courses/:id/resume
Description**:** Allows students to resume their course from where they left off.

Response:


```
{
  "id": "1",
  "current_section": "102",
  "title": "Data Types",
  "progress": "30%"
}
```


8. Download Certificate (Student Only)

Endpoint**:** GET /api/student/courses/:id/certificate
Description**:** Enables students to download their certificate of completion after finishing a course.

Response:


```
{
  "certificate_url": "https://platform.com/certificates/123456.pdf"
}
```

9. Filter Courses

Endpoint**:** GET /api/courses?search=&category=
Description**:** Enables students to filter and search for courses by name or category.

Response:


```
[

 {
  "id": "1",
  "title": "Introduction to Python",
  "category": "Programming",
  "price": 49.99
 },
 {
  "id": "3",
  "title": "Advanced JavaScript",
  "category": "Programming",
  "price": 99.99
 }
]
```


10. Admin: Manage All Courses

Endpoint**:** PUT /api/admin/courses/:id
Description**:** Allows admins to update course details.

Request Body:


```
{"title": "Updated Course Title",

"category": "Updated Category",

 "price": 59.99}
```

11. Admin: View All Users

Endpoint**:** GET /api/admin/users
Description**:** Allows admins to view all users (students and teachers) on the platform.

Response:


```
[
 {
  "id": "1",
  "name": "John Doe",
  "role": "Student"
 },
 {
  "id": "2",
  "name": "Jane Smith",
  "role": "Teacher"
 }
]
```


12. Admin: View Enrollments

Endpoint**:** GET /api/admin/enrollments
Description**:** Allows admins to view all student enrollments in courses.

Response:


```
 {
  "student_id": "1",
  "course_id": "1",
  "enrollment_date": "2024-11-28"
 },
 {
  "student_id": "2",
  "course_id": "3",
  "enrollment_date": "2024-11-29"
 }
```

AUTHENTICATION

- **JWT (JSON Web Tokens) for Authentication:**

JSON Web Tokens are used to securely transmit information between the client and server as a JSON object. They are compact, selfcontained, and signed to ensure data integrity. JWTs are primarily used to authenticate users and maintain session states.

- **Tokens Stored in Local Storage on the ClientSide:**

On successful login, the server generates a JWT and sends it to the client. The token is stored in local storage for persistence across page reloads.

- **Protected Routes on Both Frontend and Backend:**

Frontend Protected Routes: React or other frontend frameworks implement route guards to check the presence and validity of the JWT before allowing access to specific pages. Unauthorized users are redirected to the login page.

Backend Protected Routes: Backend APIs validate the JWT included in the `Authorization` header. Middleware is used to decode and verify the token, ensuring only authenticated users can access restricted endpoints.

- **Token Expiry and Refresh Tokens:**

JWTs often include an expiry time (`exp` claim). Once expired, the client must refresh the token using a refresh token mechanism or prompt the user to log in again.

- **Logout Mechanism:**

Logging out involves removing the token from local storage and redirecting the user to the login page. For additional security, the server can invalidate the token if stored in a serverside database.

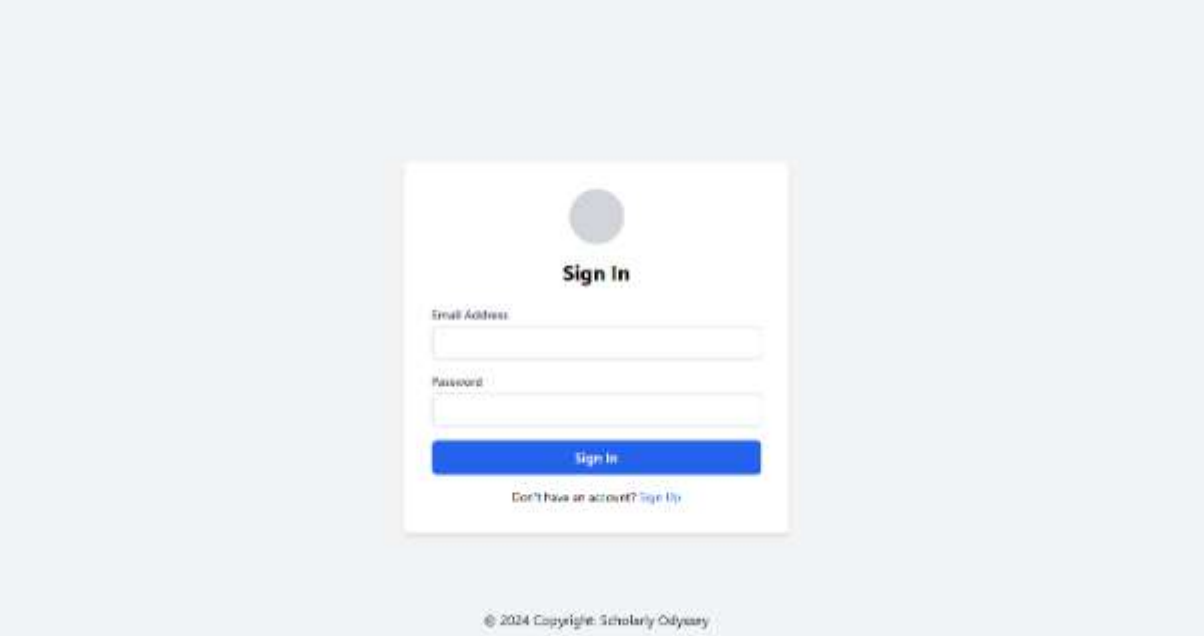
- **RoleBased Access Control (RBAC):**

JWTs can include custom claims (e.g., roles) that determine user permissions.

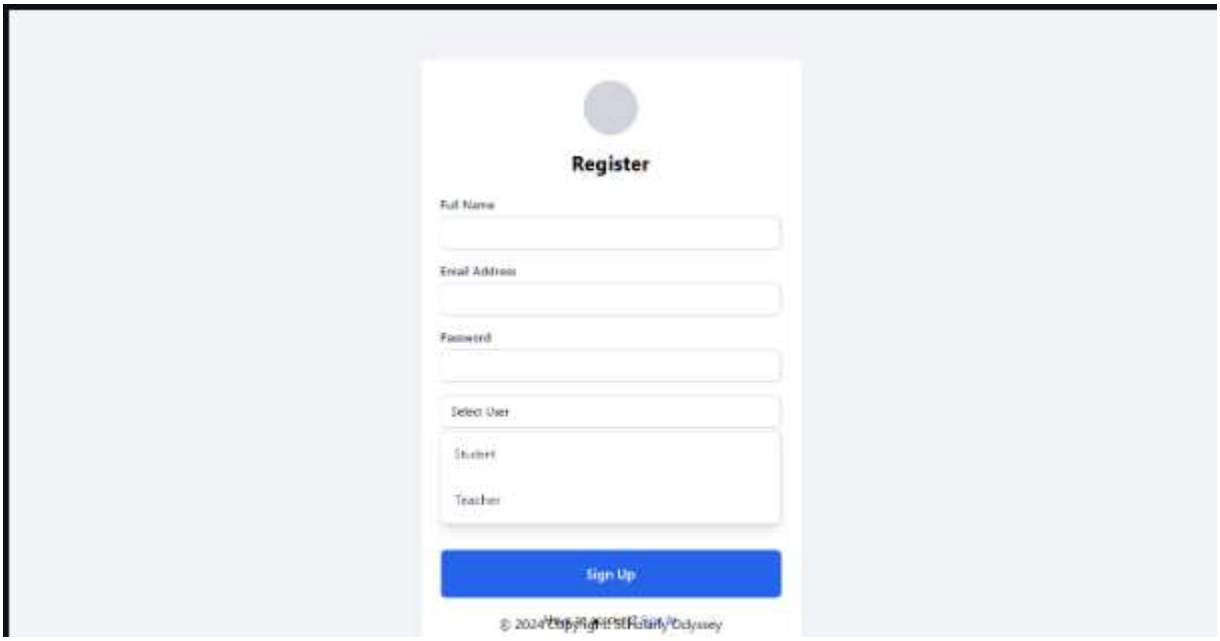
USER INTERFACE

HOME PAGE:

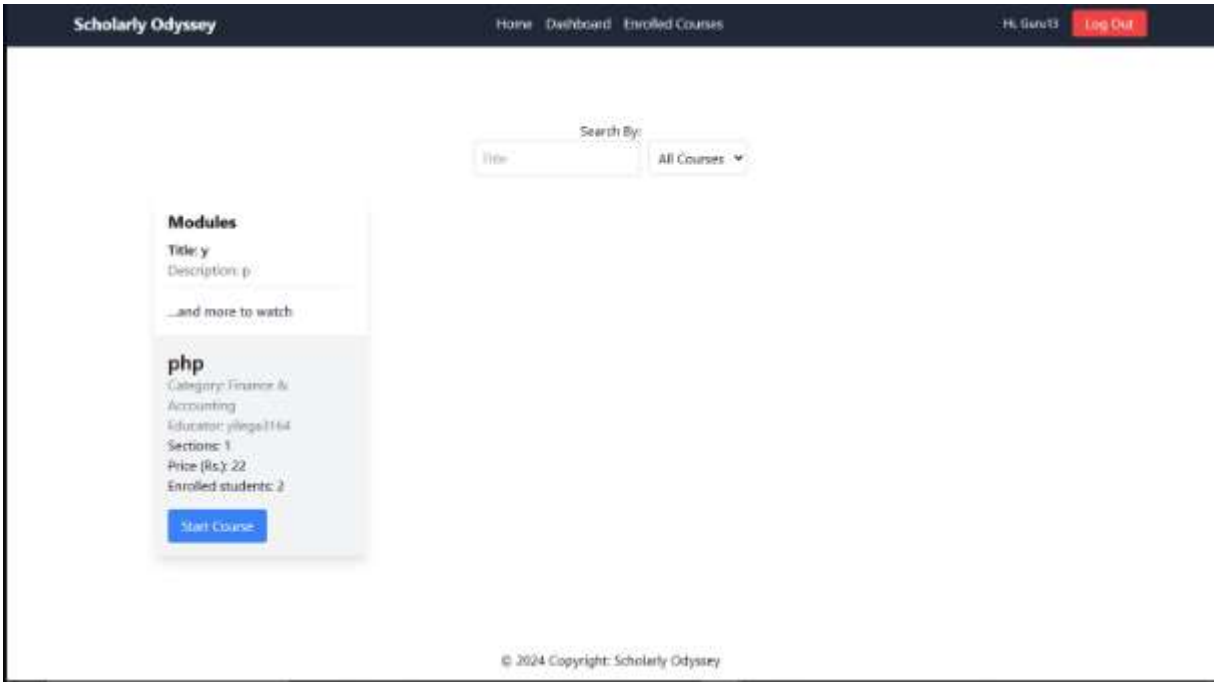
USER LOGIN:



USER REGISTER:

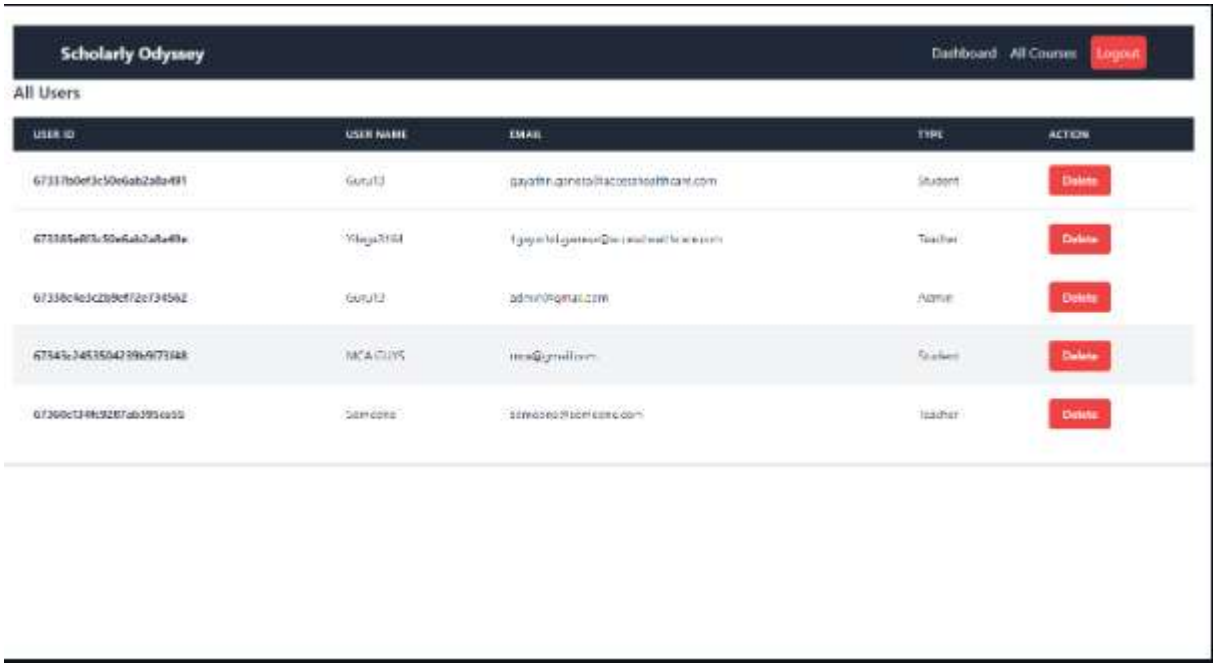


STUDENT DASHBOARD:



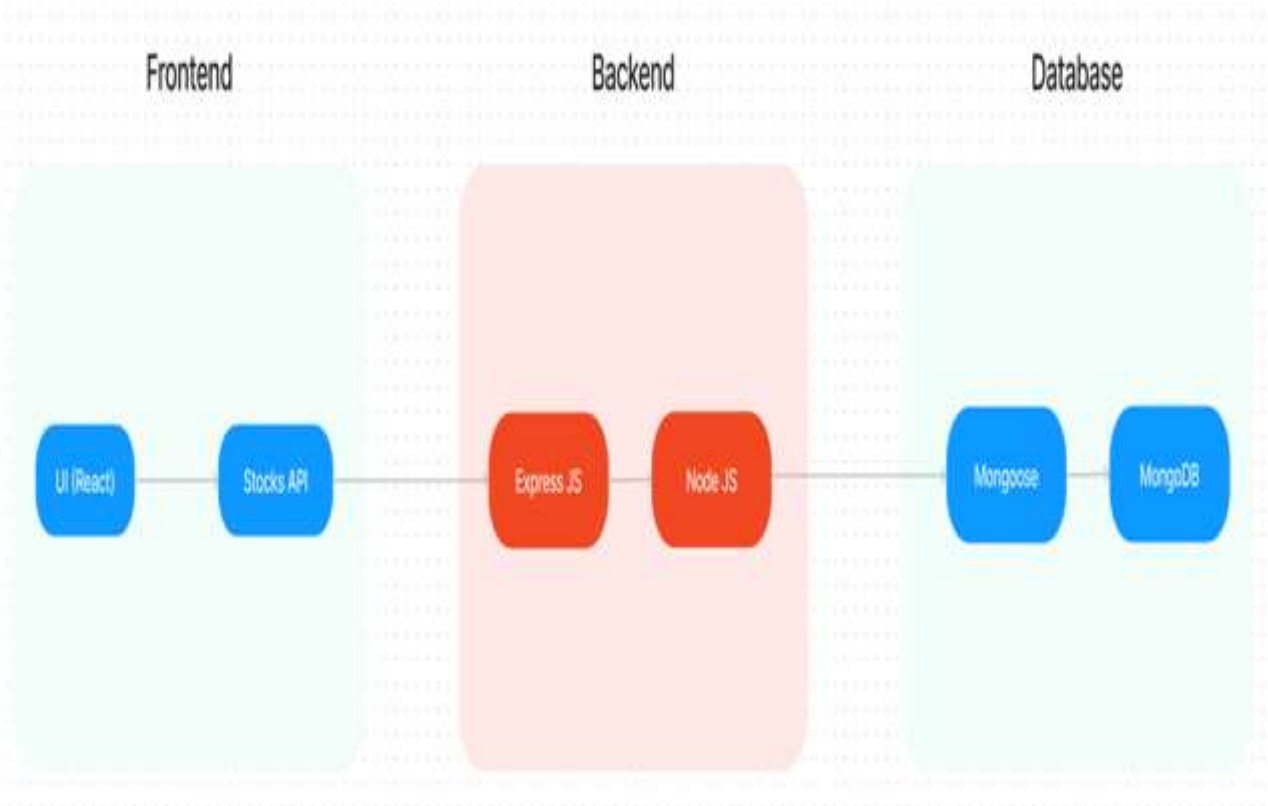
TEACHER DASHBOARD:

ADMIN DASHBOARD:



ALL COURSE

OVERALL STRUCTURE FLOW:



FUTURE ENHANCEMENT

1. Add a Loyalty Program for Frequent Customers

- **Overview**: Reward repeat customers with exclusive offers, discounts, or free items to encourage loyalty.

- **Functionality**:

Points are earned for each purchase based on order value.

Customers can redeem points for discounts, cashback, or rewards.

Tiers (e.g., Silver, Gold, Platinum) provide increasing benefits as loyalty grows.

- **Benefits**:

Encourages repeat visits and builds a loyal customer base.

Increases customer satisfaction and retention.

- **Technical Details**:

Use a centralized system to track points and redemption history.

Develop automated notifications for points updates and redemption opportunities.

2. Integrate with Multiple Payment Gateways

- **Overview**: Support various payment methods (credit/debit cards, digital wallets, UPI, net banking) for seamless transactions.
- **Functionality**:

Integrate popular gateways like Stripe, PayPal, Razorpay, or Google Pay.

Enable customers to save payment methods for quicker checkouts.

Provide multi-currency support for international customers.

- **Benefits**:

Enhances user experience by offering flexibility in payments.

Reduces cart abandonment caused by limited payment options.

- **Technical Details**:

Secure payment processing using SSL encryption.

Use webhooks to handle payment confirmations and refunds.

3. Develop a Mobile App Version Using React Native

- **Overview**: Expand accessibility by offering a mobile application compatible with iOS and Android.
- **Functionality**:

Customers can browse menus, make reservations, and place orders on the app.

Push notifications inform users of exclusive deals and updates.

CONCLUSION

Conclusion

The proposed **online learning platform** is designed to seamlessly connect teachers, students, and administrators, offering a user-friendly system for creating, enrolling in, and managing courses. With features like secure authentication, progress tracking, certificate generation, and robust role-based functionality, it ensures a tailored experience for all users. Teachers can efficiently manage courses and sections, students can easily filter, enroll, and complete courses, and administrators maintain full oversight of users, courses, and enrollments. By integrating intuitive design, advanced features, and scalability, the platform aims to enhance learning outcomes and streamline the digital education experience.

REFERENCES

MONGO DB CLOUD:
https://www.mongodb.com/products/platform/cloud

BOOTSTRAP:
https://getbootstrap.com/

W3 SCHOOLS:
https://www.w3schools.com/

COPILOT AI:
https://copilot.microsoft.com/onboarding

SWIGGY (UI REFERENCE):
https://www.swiggy.com/

CERTIFICATES

Proof of Completion

Congratulations to

Hari N/A

For successfully completing

MongoDB and the Document Model Smartbridge

On 09-24-2024

Sahir Azam
CPO
MongoDB, Inc

MongoDB.

MDBu6dxr4mer5

Proof of Completion

Congratulations to

N. Guru Prasath

For successfully completing

MongoDB Node.js Developer Path for SmartBridge

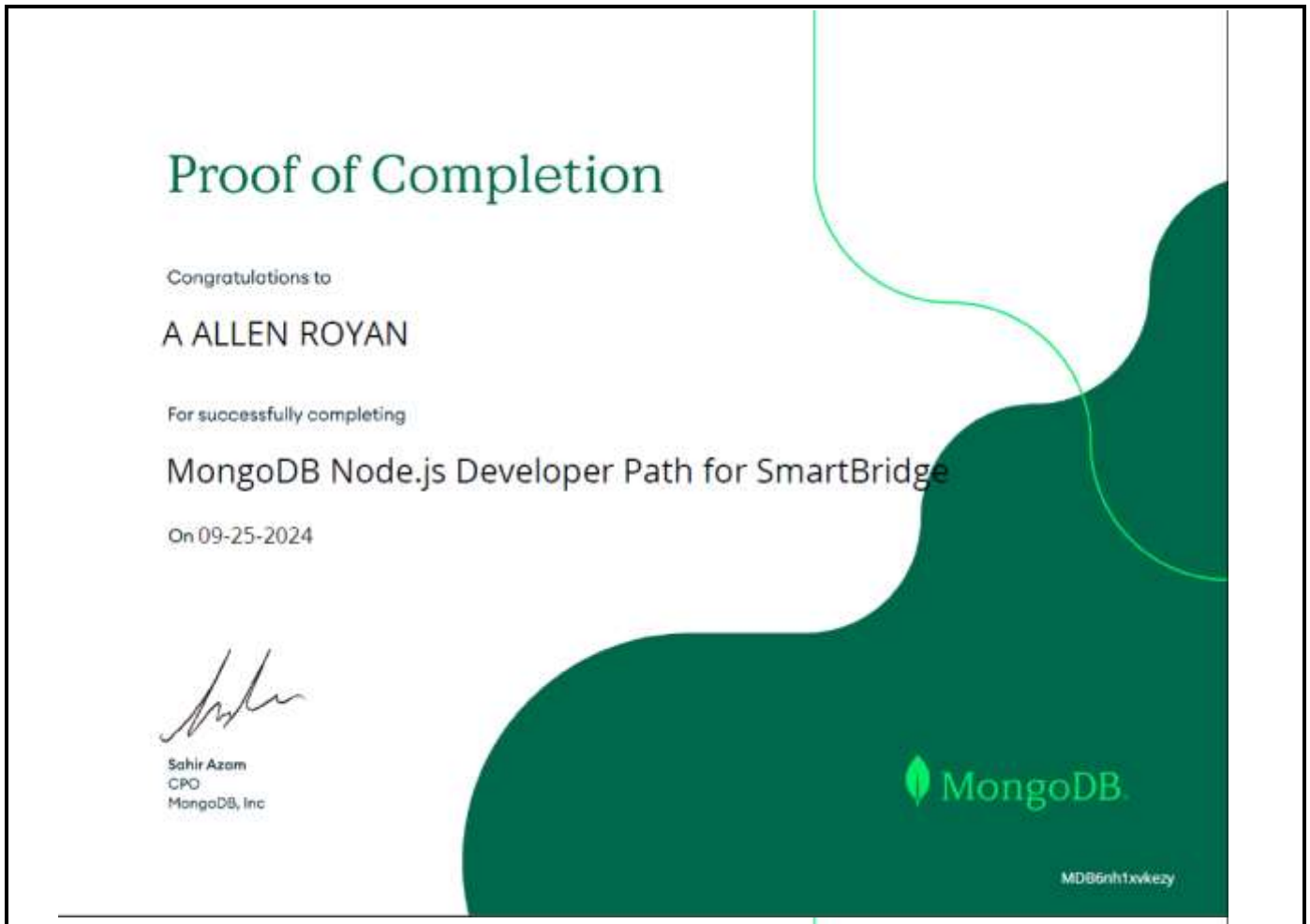
On 09-26-2024

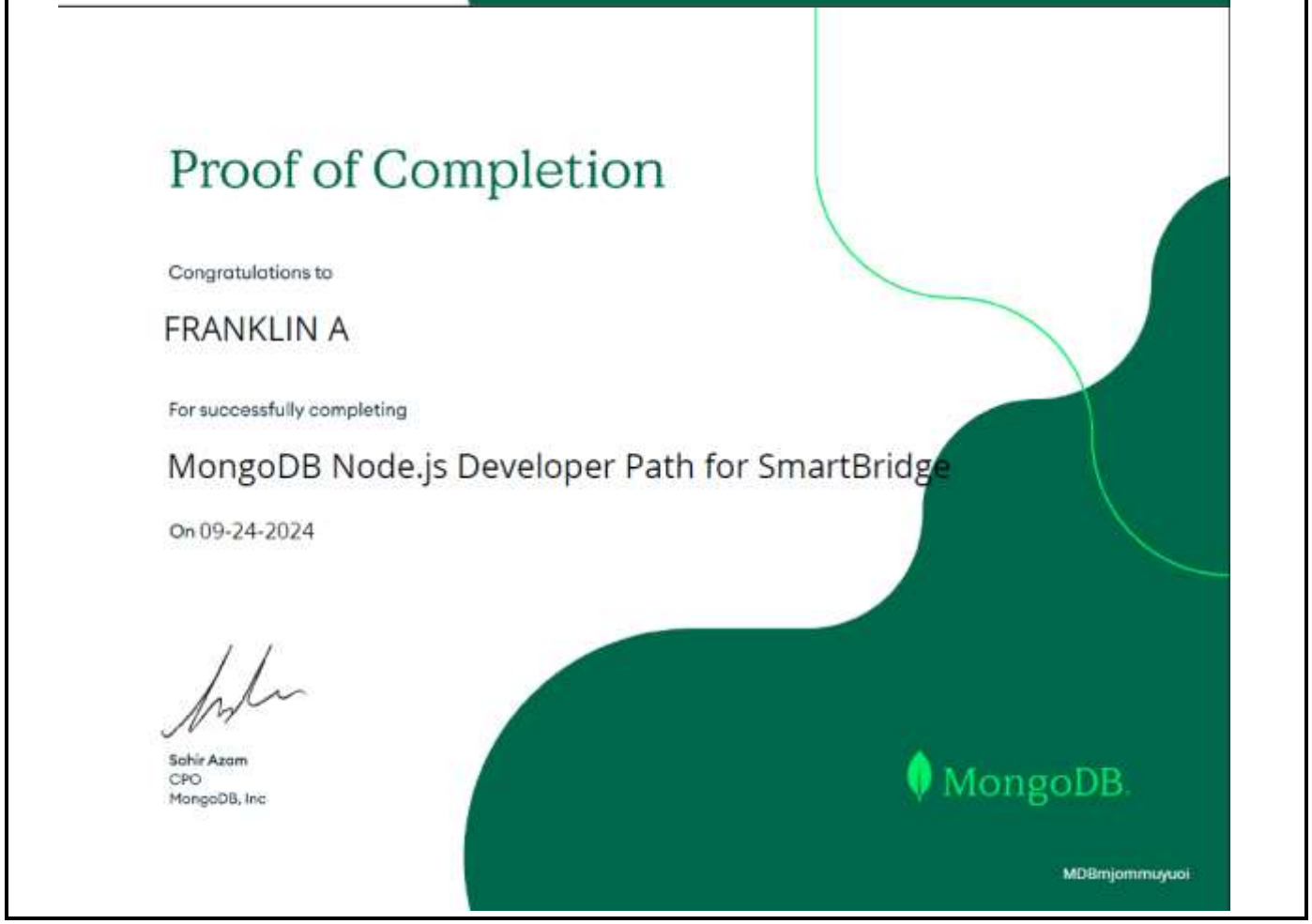
Sahir Azam
CPO
MongoDB, Inc

MongoDB.

MDBl3of4u4l0e


Proof of Completion
Congratulations to
G.Kaviya N/A
For successfully completing
MongoDB Node.js Developer Path for SmartBridge
On 09-25-2024
Sahir Azam
CPO
MongoDB, Inc
MongoDB.
MDBiy14p4phu8


Proof of Completion
Congratulations to
A ALLEN ROYAN
For successfully completing
MongoDB Node.js Developer Path for SmartBridge
On 09-25-2024
Sahir Azam
CPO
MongoDB, Inc
MongoDB.
MDB6nh1xvkezy


Proof of Completion
Congratulations to
FRANKLIN A
For successfully completing
MongoDB Node.js Developer Path for SmartBridge
On 09-24-2024
Sahir Azam
CPO
MongoDB, Inc
MongoDB.
MDBmjommuyuoi