

# **EDUOKUL: AN E-LEARNING PLATFORM**

**A PROJECT REPORT  
for  
Mini Project (KCA353)  
Session (2024-25)**

**Submitted by**

**ISHIKA GARG  
(2300290140078)  
KM. ANJALI SAGAR  
(2300290140091)  
HIMANSHI CHOPRA  
(2300290140075)**

**Submitted in partial fulfilment of the  
Requirements for the Degree of**

## **MASTER OF COMPUTER APPLICATION**

**Under the Supervision of  
Mr. Prashant Agrawal  
Associate Professor**



**Submitted to**

**DEPARTMENT OF COMPUTER APPLICATIONS  
KIET Group of Institutions, Ghaziabad  
Uttar Pradesh-201206**

**(DECEMBER 2024)**

## **CERTIFICATE**

Certified that **Ishika Garg (2300290140078), Km. Anjali Sagar (2300290140091), Himanshi Chopra (2300290140075)** have carried out the project work having “**EduOkul: an e-learning platform**” (**Mini-Project-KCA353**) for **Master of Computer Application** from Dr. A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself/herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

**Mr. Prashant Agrawal**  
**Associate Professor**  
**Department of Computer Applications**  
**KIET Group of Institutions, Ghaziabad**

**Dr. Arun Kumar Tripathi**  
**Dean and Professor**  
**Department of Computer Applications**  
**KIET Group of Institutions, Ghaziabad**

## **EduOkul: an e-learning platform**

**Ishika Garg**

**Km. Anjali Sagar**

**Himanshi Chopra**

### **ABSTRACT**

EduOkul is a comprehensive e-learning platform built using the MERN stack (MongoDB, Express.js, React.js, Node.js) to cater to both students and administrators in an online educational ecosystem. The platform provides a user-friendly interface and a secure authentication system that enables students to register, log in, and access various courses. Admins, on the other hand, have the ability to manage users, courses, and course content efficiently, offering a seamless user experience on both ends.

The platform's core functionality includes a dual registration and login system for both students and admins, with an OTP-based verification process during registration for added security. Upon successful registration, users can log in to access a personalized dashboard and begin their course journey. Once logged in, students can explore and subscribe to courses, view course details such as price, duration, and instructor information, and access video-based lectures associated with each course. Each course is accompanied by an image and a detailed description, ensuring an engaging user experience. Students can also track their enrolled courses from their account dashboard.

The admin panel of EduOkul provides comprehensive control over the platform. Admins can create, edit, and delete courses, upload course-related images, and manage lecture content by adding or deleting video lectures. Admins can also manage user roles, such as upgrading users to admin roles or vice versa. The platform includes a secure user authentication system, leveraging JSON Web Tokens (JWT) for maintaining user sessions.

File uploads for course images and video lectures are managed using Multer, ensuring efficient handling of media files. The platform is hosted on Vercel, which provides fast and scalable hosting for both the backend and frontend, ensuring that the platform remains highly available and responsive to user requests.

EduOkul's overall design focuses on scalability, maintainability, and security, providing a flexible solution for online learning. With its modular architecture, the platform can be easily extended to include more advanced features in the future, such as password recovery, payment integration for course purchases, and interactive assessments. By leveraging modern web technologies, EduOkul aims to provide a robust and reliable e-learning solution for both students and educators, helping to bridge the gap in online education with an intuitive, efficient, and secure platform.

## **ACKNOWLEDGEMENTS**

Success in life is never attained single-handedly. My deepest gratitude goes to my project supervisor, **Mr. Prashant Agrawal** for his guidance, help, and encouragement throughout my project work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to Dr. Arun Kumar Tripathi, Professor and Head, Department of Computer Applications, for his insightful comments and administrative help on various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me with moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

**Ishika Garg**

**Km. Anjali Sagar**

**Himanshi Chopra**

# TABLE OF CONTENTS

Certificate	ii
Abstract	iii
Acknowledgements	iv
Table of Contents	v
List of Tables	viii
List of Figures	ix
1 Introduction	10-12
1.1 Overview	10
1.2 Features of the Platform	10
1.2.1 User Registration and Login	10
1.2.2 Homepage and Navigation	10
1.2.3 Courses	11
1.2.4 User Account and Profile	11
1.2.5 Admin Dashboard	11
1.3 Technologies Used	12
2 Feasibility Study	13-18
2.1 Technical Feasibility	13
2.1.1 Technologies Used	13
2.1.2 Deployment on Vercel	14
2.1.3 Scalability	15
2.1.4 Security Considerations	15
2.2 Operational Feasibility	15
2.2.1 User Interaction Flow	15
2.2.2 Course Management	16
2.2.3 User Role and Management	16
2.2.4 Content Delivery and Accessibility	16
2.3 Financial Feasibility	17
2.3.1 Development Cost	17
2.3.2 Operational Cost	17
2.3.3 Revenue Generation	18
2.3.4 Long Term Sustainability	18
2.4 Conclusion	18
3 Design	19-25
3.1 Introduction	19
3.2 Front-End Design	19
3.2.1 Layout and Navigation	19

3.2.2	Home Page Design	20
3.2.3	Courses Page Design	20
3.2.4	Courses Detail Page	20
3.2.5	Account and Profile Page	21
3.2.6	Admin Dashboard	21
3.3	Back-End Design	21
3.3.1	System Architecture	21
3.2.2	Database Design	22-24
3.4	Security Considerations	24
3.5	Data Flow Diagram (DFD)	25
4	Testing	26-29
4.1	Introduction	26
4.2	Types of Testing	26
4.2.1	Manual Testing	26
4.2.2	Unit Testing	27
4.2.3	Integration Testing	27
4.2.4	User Acceptance Testing	28
4.3	Testing Tools	28
4.4	Bug Reporting and Fixes	29
4.5	Conclusion	29
5	Project Screenshot	30-37
5.1	Home Page	30
5.1.1	Header	30
5.1.2	Review Section	30
5.2	Courses Page	31
5.3	Login Page	32
5.4	About Us Page	32
5.5	Account Page	33
5.6	Admin Dashboard (Home)	33
5.7	Admin Dashboard (Courses)	34-35
5.8	Admin Dashboard (Users)	36
5.9	User Account Page	36
5.10	User Dashboard Page	37
6	Future Scope	38-41
6.1	Advanced User Authentication and Authorization	38
6.2	Course Recommendations and Personalization	38
6.3	Interactive Learning Feature	39
6.4	Gamification of Learning	39
6.5	Mobile Application	39
6.6	Payment Gateway Integration	39
6.7	Collaboration and Certification	40

6.8	AI-Powered Tutor Assistance	40
6.9	Integration with Other Educational Platforms	40
6.10	Scalability and Performance	40-41
7	Conclusion	42-43
8	References	44

## LIST OF TABLES

<b>Table No.</b>	<b>Name of Table</b>	<b>Page</b>
3.1	Users Table	22
3.2	Courses Table	23
3.3	Lectures Table	24



## LIST OF FIGURES

<b>Figure No.</b>	<b>Name of Figure</b>	<b>Page No.</b>
3.1	Data Flow Diagram	25
5.1.1	Home Page	30
5.1.2	Review Section	31
5.2	Courses Page	31
5.3	Login Page	32
5.4	About Us Page	33
5.5	Account Page	33
5.6	Admin Dashboard Home Page	34
5.7.1	Admin Dashboard Courses Page	35
5.7.2	Admin Dashboard Courses Page (Adding Lecture)	35
5.8	Admin Dashboard Users Page	36
5.9	User Account Page	37
5.10	User Dashboard Page	37

# CHAPTER 1

## INTRODUCTION

### 1.1 OVERVIEW

EduOkul is an e-learning platform built using the MERN stack (MongoDB, Express.js, React.js, Node.js) that provides an interactive and engaging online learning experience. The platform is designed to offer a range of courses, including video lectures, for both students and administrators. The website provides user registration, login, and account management for students and admins, along with functionality to view, enrol in, and study courses.

### 1.2 Features of the Platform

The EduOkul platform has been developed with a user-friendly interface that supports both student and admin roles. The key features of the platform are as follows:

#### 1.2.1 User Registration and Login

- **Student and Admin Registration:**
  - The registration process includes three fields: **Name**, **Email**, and **Password**.
  - Upon successful registration, users are redirected to the login page.
  - Users must confirm their email address via OTP (One Time Password) sent to the registered email.
- **Login Process:**
  - Users can log in with their **Email** and **Password**.
  - There is no **Forgot Password** option implemented at the moment.
  - Upon successful login, the user is redirected to the homepage, and the **Login** option on the navigation bar is replaced with the **Account** option.

#### 1.2.2 Homepage & Navigation

- **Navigation Bar:**
  - The navigation bar contains the **EduOkul** logo and options such as **Home**, **About Us**, **Courses**, and **Login**.
  - When logged in, the **Login** option changes to **Account**.
  - The homepage includes a **Get Started** button that redirects the user to the **Courses** page.
- **About Us Section:**
  - The **About Us** page contains a brief description of the platform in one or two paragraphs.

### 1.2.3 Courses

- **Course Access:**
  - Users can view the course catalog, but access to detailed courses is restricted to logged-in users only.
  - The course section includes images for each course, and users can see the course name, description, price, and duration.
  - Clicking on **Study** redirects the user to a course details page.
- **Course Details Page:**
  - The course description, price, and other relevant details are displayed.
  - Users can watch video lectures listed on the right side of the page.
  - Only **videos** are allowed for lectures, with each course having its own image.
  - Admins can **Add**, **Delete**, or **Access** lectures.

### 1.2.4 User Account & Profile

- **User Account Page:**
  - After logging in, the **Account** section shows **My Profile**, which displays the user's name and email, along with options for **Dashboard** and **Logout**.
  - The **Dashboard** button takes the user to a page where they can view all the courses they are enrolled in.

### 1.2.5 Admin Dashboard

- **Admin-Specific Features:**
  - Admins have additional privileges such as managing users and courses.
  - The **Admin Dashboard** includes options like **Users**, **Courses**, **Logout**, and **Home** (a special admin home page).
- **Managing Courses:**

- Admins can view, add, or delete courses. Each course has fields like **Title**, **Description**, **Category**, **Price**, **Duration**, and **Created By**.
- The **Add Course** form includes a file upload button for adding images to represent the course.
- **Managing Lectures:**
  - Admins can view lectures associated with each course and can add or delete lectures. The lecture addition form includes **Title**, **Description**, and a file upload button for video lectures.
- **User Management:**
  - The **Users** section allows the admin to view a list of all users, with their names, emails, and roles (User/Admin).
  - Admins can update the roles of users through an **Update Role** button.
- **Admin Logout:**
  - Admins can log out by clicking the **Logout** button, which redirects to the login page.

### 1.3 Technologies Used

- **Frontend:**
  - **React.js:** For building the user interface, handling state management, and rendering components dynamically.
  - **React Router:** For navigating between different pages, including the homepage, courses, and account pages.
- **Backend:**
  - **Node.js & Express.js:** Used to handle server-side logic, API routing, and user authentication.
- **Database:**
  - **MongoDB:** A NoSQL database used to store user information, courses, and other dynamic content. Mongoose is used for schema definition and database interaction.
- **Email Service:**
  - OTP emails are sent using a third-party email service like **Nodemailer**, which facilitates email functionality in the platform.
- **Authentication:**
  - **JWT (JSON Web Tokens)** is used for authentication to secure routes and ensure that only authenticated users can access certain pages and resources.

## CHAPTER 2

### FEASIBILITY STUDY

A feasibility study analyses the viability of a project to determine whether the project or venture is likely to succeed. The study is also designed to identify potential issues and problems that could arise while pursuing the project.

A feasibility study is essential to determine whether the EduOkul project is viable in terms of its technical, operational, and financial aspects. Below is a detailed feasibility analysis of the project, updated to reflect deployment on **Vercel**:

#### 2.1 Technical Feasibility

Technical feasibility assesses the ability of the proposed technologies and tools to support the development and operational requirements of the EduOkul platform. Here, we evaluate the core components of the project:

##### 2.1.1 Technologies Used

The EduOkul platform is built using the **MERN Stack** (MongoDB, Express.js, React.js, Node.js). Each of these technologies has been chosen based on their robustness, scalability, and wide usage in the development community.

- **MongoDB:** A NoSQL database is used for storing user data, course information, and other dynamic content. MongoDB is highly scalable and flexible, making it a great choice for applications that handle varying data formats. It allows for horizontal scaling, which is beneficial as the platform grows.
- **Node.js:** A powerful server-side JavaScript framework that uses event-driven architecture to handle numerous requests concurrently. This will be used to develop the backend of the platform. Its asynchronous nature is perfect for handling I/O operations like API requests, course management, and email processing.

- **Express.js:** A web application framework for Node.js that simplifies routing, middleware management, and API integration. It helps to structure the backend code and manage HTTP requests efficiently.
- **React.js:** A widely used frontend JavaScript library for building interactive and dynamic user interfaces. React's component-based architecture ensures that the user interface is maintainable and scalable. The use of **React Router** allows for smooth navigation between pages like the login page, course catalog, and user dashboard.
- **Authentication: JWT (JSON Web Tokens)** will be used for user authentication and secure session management. This allows users to log in and retain their authentication token across multiple sessions while ensuring that only authorized users can access certain resources.
- **File Handling:** The platform will use **Multer** middleware for handling file uploads (course images and lecture videos). Multer is efficient in managing file storage and enables image and video uploads on the backend.
- **Email Service: Nodemailer** will be used to send emails, including OTPs for email validation during user registration.

### 2.1.2 Deployment on Vercel

EduOkul will be deployed using **Vercel**, a cloud platform for static and serverless deployments. Vercel is particularly well-suited for React-based projects because of its seamless integration with frontend frameworks like React.js.

- **Frontend Deployment:** Vercel provides simple deployment for frontend React apps. It automatically handles **build** and **deploy** processes, integrates well with GitHub, and provides continuous deployment.
- **Backend Deployment:** While Vercel is optimized for static sites, it also supports serverless functions, which will be used for the backend of EduOkul. **API routes** (Express.js routes) can be deployed as serverless functions, allowing for scalability and a streamlined deployment process. The use of serverless functions ensures that the platform's backend is responsive and resource-efficient.
- **Database: MongoDB Atlas** will be used for cloud database management. It integrates well with Vercel, and Vercel provides optimizations for connecting to remote databases, ensuring low-latency data access.

### 2.1.3 Scalability

The **MERN Stack** is highly scalable, and this is complemented by the **serverless** deployment model provided by **Vercel**. As EduOkul grows in terms of users, courses, and data, the backend and frontend can scale independently:

- **Frontend Scaling:** Vercel automatically scales the frontend app based on traffic, ensuring that users receive quick page loads even during peak times.
- **Backend Scaling:** Serverless functions on Vercel scale automatically depending on the number of incoming requests. This allows EduOkul to handle fluctuating traffic efficiently without needing to manage dedicated servers.

Additionally, **MongoDB Atlas** will ensure that the platform's database can scale horizontally, adapting to the increasing amount of data generated by users.

### 2.1.4 Security Considerations

Ensuring security is a critical aspect of the project, particularly with sensitive data like user passwords and course materials. The following security measures will be implemented:

- **JWT Authentication:** Secure tokens are used to authenticate users and protect sensitive routes.
- **Data Encryption:** User passwords will be hashed using **bcrypt** before storing them in the database.
- **Role-Based Access Control (RBAC):** Admin and user roles will be clearly defined, with restricted access to certain sections of the platform (e.g., course management is limited to admins).
- **SSL/TLS Encryption:** HTTPS will be used across the platform to secure data transmission between the client and server.

## 2.2 Operational Feasibility

Operational feasibility examines whether the organization can implement the platform successfully from a managerial and operational standpoint. This section covers user interaction, management processes, and expected user activity.

### 2.2.1 User Interaction Flow

EduOkul provides two main user roles: **Students** and **Admins**. The user flow is straightforward:

- **Students** register by providing their name, email, and password, followed by OTP verification. After logging in, they can access the course catalog, enroll in courses, and watch video lectures.
- **Admins** follow a similar registration process but with elevated privileges. After logging in, they have access to a dedicated admin dashboard where they can manage courses, lectures, and users.

### 2.2.2 Course Management

The admin can manage courses effectively:

- **Add/Edit/Delete Courses:** Admins can add new courses by providing a title, description, price, and image. They can also edit existing courses or delete courses.
- **Lecture Management:** Admins can add video lectures to courses and delete or edit existing ones.

As more courses are added, admins will need tools for managing the growing catalog efficiently. Features such as search, filters, and sorting by category, price, and duration will be implemented to help admins and users easily navigate the platform.

### 2.2.3 User and Role Management

User management is critical, particularly for maintaining the correct roles and permissions. Admins can manage user roles through a dedicated dashboard, allowing them to upgrade or downgrade users' roles (e.g., from user to admin). The system will also allow for tracking and viewing user activity, such as course enrollments and completed courses.

### 2.2.4 Content Delivery and Accessibility

The platform will focus on accessibility by providing:

- **Responsive Design:** The platform will be responsive across various devices such as desktops, tablets, and mobile phones, ensuring that users can access content anywhere.



- **Video Streaming:** Lecture videos will be delivered in a format that ensures smooth playback and compatibility across all devices.

## 2.3. Financial Feasibility

Financial feasibility looks at the costs involved in the development, deployment, and maintenance of EduOkul and whether the platform will be financially sustainable. Here's a detailed overview of the financial aspects:

### 2.3.1 Development Costs

- **Personnel:** The development process will require a team of developers skilled in React.js, Node.js, Express.js, and MongoDB. The cost of hiring or contracting developers varies based on location and skill level, but the open-source nature of the technologies used minimizes additional software costs.
- **Development Tools:** Tools for development and testing are mostly open source, reducing upfront costs. Some paid services (e.g., for cloud storage, email services) will incur monthly fees.

### 2.3.2 Operational Costs

- **Cloud Infrastructure:** EduOkul will be hosted on **Vercel**, a cloud platform with **serverless functions** for handling the backend. Vercel offers a free tier, but as the platform scales, the need for paid tiers may arise. Vercel's serverless model is cost-efficient, with pricing based on usage, so the operational cost grows with the platform's traffic.
- **MongoDB Atlas:** MongoDB Atlas provides a flexible pricing model based on the resources used, including storage and the number of requests. It is highly scalable and integrates seamlessly with Vercel, making it a suitable choice for cloud database management.
- **Third-Party Services:**
  - **Email Service: Nodemailer** will be used for email functionality, and this may have a cost associated with higher volume usage, depending on the number of registrations and OTP verifications.
  - **Storage:** Storing course images and videos will require cloud storage services like **AWS S3** or **Cloudinary**. These services generally offer pay-

as-you-go pricing, which will grow as more courses and lectures are uploaded.

### 2.3.3 Revenue Generation

- **Subscription Model:** The platform could generate revenue through a subscription-based model, charging users a monthly or yearly fee to access premium courses. Admins could add paid courses, and users would pay to enroll in these courses.
- **Freemium Model:** Some courses could be offered for free, with additional advanced courses available for a fee.

### 2.3.4 Long-Term Sustainability

The financial sustainability of EduOkul depends on its ability to grow its user base and generate consistent revenue. The costs of operation are manageable, especially with scalable cloud hosting solutions and open-source technologies. The platform's success will largely depend on its ability to attract and retain users and offer high-quality courses that provide value.

## 2.4 Conclusion

The feasibility study concludes that the **EduOkul** project is technically, operationally, and financially viable. The MERN stack provides a solid foundation for the platform, with scalable technologies that will support growth. The operational flow is efficient, and user management features will help ensure smooth interaction between students and admins. Financially, the platform can be sustained through a subscription or freemium model, and costs remain manageable.

Given these factors, EduOkul has strong potential to succeed as an e-learning platform, offering both educational value and a solid business model. The use of **Vercel** for deployment ensures a seamless and cost-effective deployment process, making the platform highly scalable and ready for future growth.

## CHAPTER 3

### DESIGN

This section will cover the overall design considerations, user interface (UI) design, system architecture, database schema, and API design.

#### 3.1 Introduction

The design of the EduOkul platform focuses on creating an efficient, user-friendly environment for both students and administrators. The system is built using the MERN stack (MongoDB, Express, React, Node.js), providing a seamless user experience while ensuring flexibility, scalability, and security. The design encompasses both front-end (UI/UX) and back-end (database and API) components.

#### 3.2. Front-End Design (User Interface)

The front-end of EduOkul is designed to be simple, intuitive, and responsive. It ensures that users can easily navigate through various pages, courses, and functionalities. The design is built using **React** to provide a dynamic and interactive user experience.

##### 3.2.1 Layout and Navigation

The platform's navigation system includes a consistent navigation bar (navbar) available on all major pages. This navbar includes the following components:

- **Home:** Links to the homepage with general information and call-to-action buttons like "Get Started."
- **About Us:** Provides basic information about the platform.
- **Courses:** Displays a list of available courses, but users must log in to access course details and enrol.

- **Login / Account:** Allows users to log in or access their account after logging in.

For users who are logged in, the navbar changes to display the **Account** option, and the login button is replaced with a logout button. This allows the user to access their profile and manage their subscriptions.

### 3.2.2 Home Page Design

The homepage serves as an introduction to EduOkul. It includes the following elements:

- **Welcome Section:** A brief welcome message with a call-to-action button, "Get Started," which redirects users to the Courses page.
- **Course Previews:** A brief preview of some available courses with images and titles.
- **Footer:** Contains additional links, such as contact information or terms of use.

### 3.2.3 Courses Page Design

The courses page displays all available courses. Each course is represented by an image, title, category, price, and duration. Unauthenticated users can view the course listings but cannot access course details or enroll. The page includes a "Subscribe" button, which becomes active only once the user logs in.

After logging in, the user can click on a course to view more details, including:

- **Course Description:** A detailed overview of what the course offers.
- **Instructor Details:** Information about the course creator.
- **Enroll / Subscribe Option:** To enroll in the course.

### 3.2.4 Course Detail Page

When a user clicks on a course, they are redirected to a page with detailed information, such as:

- **Course Overview:** Detailed description, price, duration, and instructor.

- **Lectures Section:** A list of available lectures in the course. Each lecture contains a title, description, and a "Study" button.
- **Video Playback:** When the user clicks on the "Study" button, they are redirected to a new page where they can watch the video lecture.

### 3.2.5 Account and Profile Page

The account page displays the following:

- **Profile Information:** User's name, email, and other personal details.
- **Dashboard Button:** Redirects users to their personal dashboard, showing the courses, they are enrolled in.
- **Logout Button:** Logs the user out and redirects them to the login page.

For the **admin** user, additional sections are displayed, such as the **Admin Dashboard**.

### 3.2.6 Admin Dashboard

The **Admin Dashboard** is a unique page only accessible to users with admin privileges. It provides the following functionalities:

- **Sidebar Navigation:** Links to manage courses, users, and system settings.
- **Courses Management:** Admin can add, delete, and modify courses. There is also an option to add lectures to each course.
- **Users Management:** Admin can update roles of users (student/admin).
- **Overview Section:** Displays key statistics, such as the total number of courses, users, and lectures.

## 3.3. Back-End Design (Server, API, and Database)

### 3.3.1 System Architecture

EduOkul follows a **client-server architecture**, where:

- The **client side** is built using React, providing the front-end for users to interact with.
- The **server side** is built using Node.js and Express, which handle API requests, interact with the database, and serve responses to the front-end.
- **MongoDB** serves as the database, storing data for courses, users, and lectures. It allows for flexibility, scalability, and easy querying.

### 3.3.2 Database Design

The database schema is designed to handle the core functionalities of EduOkul. The following collections are used:

#### 1. Users Collection

This collection stores information about the users, including both students and admins.

Field	Data Type	Description
id	ObjectId	Unique identifier (automatically generated by MongoDB)
name	String	Full name of the user
email	String	Email address (used for login)
password	String	Hashed password for security
role	String	Role of the user (user or admin)
subscription	Array	List of course IDs the user is enrolled in
createdAt	Date	Timestamp when the user account was created

updatedAt	Date	Timestamp when the user profile was last updated
-----------	------	--

Table 3.1: Users Table

## 2. Courses Collection

This collection stores the course information.

Field	Data Type	Description
id	ObjectId	Unique identifier (automatically generated by MongoDB)
title	String	Title of the course
description	String	Brief description of the course
image	String	Path to the course image
price	Decimal	Price of the course
duration	String	Duration of the course
category	String	Category of the course
createdBy	ObjectId	Reference to the <b>Users</b> collection (ID of the admin/user who created the course)
createdAt	Date	Timestamp when the course was created

Table 3.2: Courses Table

### 3. Lectures Collection

This collection stores the lectures related to each course.

Field	Data Type	Description
id	ObjectId	Unique identifier (automatically generated by MongoDB)
title	String	Title of the lecture
description	String	Description of the lecture
video	String	Path to the video
course	ObjectId	Reference to the <b>Courses</b> collection (ID of the associated course)
createdAt	Date	Timestamp when the lecture was created

Table 3.3: Lectures Table

### 3.4 Security Considerations

Security is a critical aspect of the platform:

- **User Authentication:** Passwords are securely hashed using bcrypt to ensure the safety of user data.
- **Role-based Access Control:** Different routes and actions are protected based on user roles (admin or user).
- **JWT Tokens:** JSON Web Tokens (JWT) are used for maintaining user sessions and ensuring secure access to restricted areas of the platform.



### 3.5 Data Flow Diagram (DFD)

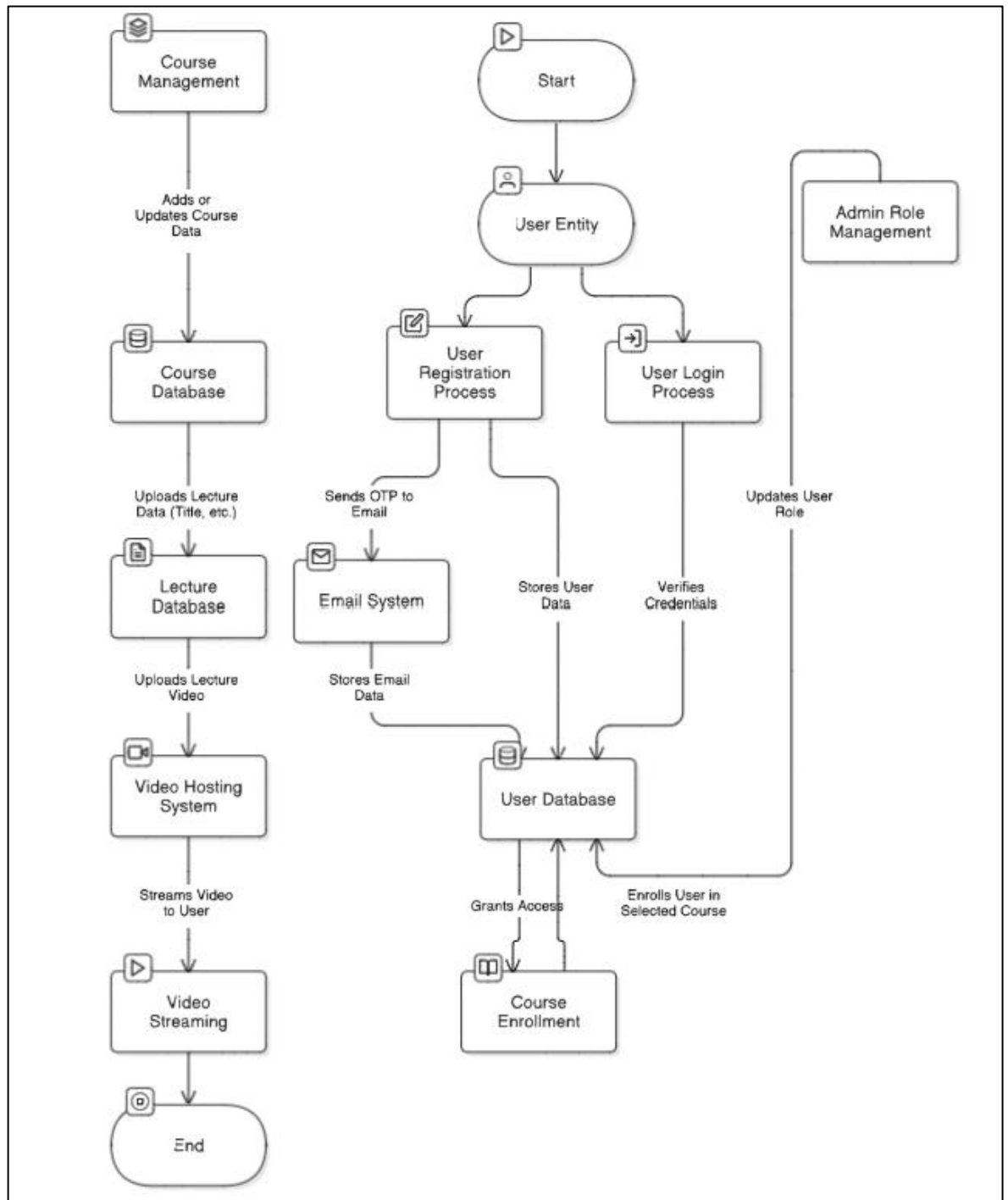


Figure 3.1: Data Flow Diagram

## CHAPTER 4

### TESTING

Testing is a critical part of ensuring that the EduOkul platform is functional, user-friendly, and stable. This section outlines the different types of testing performed, the tools used, and the approach taken to validate both front-end and back-end components.

#### 4.1 Introduction to Testing

For EduOkul, testing was focused on ensuring the following:

- **Correct functionality** of core features such as registration, login, course enrolment, and video playback.
- **Seamless integration** between the front-end (React), back-end (Node.js/Express), and the database (MongoDB).
- **Responsive design** and ease of use across various devices.
- **API correctness** for all interactions between the client and server.

To achieve this, both automated and manual testing were performed across several phases, covering all aspects of the platform.

#### 4.2 Types of Testing

Various types of testing were carried out during the development of EduOkul to ensure its robustness:

##### 4.2.1 Manual Testing

Manual testing was conducted by a group of testers who interacted with the EduOkul platform directly, performing test cases on the user interface and user experience. This included ensuring that all workflows, such as registration, course browsing, and video playback, functioned as expected on different devices. Testers followed predefined steps to validate whether the system meets the requirements, including actions like:

- Signing up as a new user and logging in.
- Testing the process of browsing and enrolling in courses.
- Ensuring the responsiveness of the platform across multiple screen sizes and devices.

#### **4.2.2 Unit Testing**

Unit testing was carried out for individual JavaScript functions and utilities. These tests focused on ensuring the correctness of the internal logic of the platform. For example, tests were written to check the following:

- Whether form validation logic correctly verifies inputs during registration and login.
- If the course data processing logic accurately handles course creation and modifications.
- Ensuring that user authentication processes (like password hashing) function as expected.

Though Jest, Mocha, and Chai weren't used in this case, basic unit tests were written in JavaScript to test core functions and utilities.

#### **4.2.3 Integration Testing**

Integration testing ensured that the front-end, back-end, and database layers communicated correctly. API endpoints were tested to verify that data flowed smoothly from the client side to the server, and responses were accurate. For example:

- Ensuring that after a user registers, their data is properly saved in the database.
- Verifying that course details are correctly fetched and displayed from the database to the front-end after the user selects a course.

- Confirming that enrolling in a course updates the user's subscription in the database.

Each of these integrations was tested to ensure that all components worked as expected in combination.

#### **4.2.4 User Acceptance Testing (UAT)**

User Acceptance Testing (UAT) was conducted with real users to ensure the platform met their expectations for functionality and usability. This phase was crucial for validating that the platform was user-friendly, and that all workflows were intuitive. Key UAT scenarios included:

- Testing how new users interact with the registration and login pages.
- Confirming that users can easily find and enroll in courses.
- Gathering feedback on the general usability of the interface and design.

Feedback gathered during this phase helped refine the platform's design, improve the course browsing experience, and enhance the overall user flow.

### **4.3 Testing Tools**

To support the different types of testing, the following tools were employed:

#### **1. Postman:**

- Postman was used for API testing, especially for testing the server-side routes and ensuring correct data flow between the client and the database.
- Various API endpoints, such as user registration, course fetching, and course enrollment, were tested using Postman by sending requests to the backend and verifying responses.

#### **2. MongoDB Compass:**

- MongoDB Compass was used to manually inspect the database and verify that the correct data was being stored after performing certain actions (such as creating a user or enrolling in a course).

- It allowed the testing team to check whether the backend was correctly handling data insertion, updates, and deletions in collections such as users, courses, and lectures.

### 3. **Manual Testing:**

- Manual testing was essential to identify potential usability issues, especially regarding the user interface. Testers focused on ensuring that the website was intuitive and user-friendly by performing real-user actions, interacting with the UI, and ensuring responsiveness on different devices.

## 4.4 Bug Reporting and Fixes

Throughout the testing process, several bugs were identified and resolved. The following are some of the key issues that were addressed:

1. **UI Misalignment on Mobile:** On smaller screen sizes, certain elements were misaligned. This issue was fixed by updating the CSS styles, ensuring the platform remains responsive and user-friendly across all devices.
2. **Course Enrolment Failure:** An issue occurred where users could not enroll in courses due to an incorrect database update. This was resolved by ensuring that the subscription data was correctly updated in the database once a user enrolled.
3. **Admin Dashboard Access:** Some admin users encountered errors while trying to access the Admin Dashboard due to role-based access control (RBAC) issues. These access issues were resolved by correctly implementing role checks before allowing access to certain pages.

## 4.5 Conclusion

The testing phase for the EduOkul platform was thorough, covering all critical areas such as functionality, performance, and usability. The platform passed all major test cases, and the feedback received from UAT allowed for the refinement of the user interface. The use of Cypress, Postman, MongoDB Compass, and manual testing tools ensured that both the front-end and back-end of the system were robust and ready for production.

With all tests successfully passed, EduOkul is now ready for deployment, and further optimizations will be made based on user feedback and future load testing.

## CHAPTER 5

### PROJECT SCREENSHOT

#### 5.1.1 Home Page:

The homepage of EduOkul features a clean and user-friendly layout with a navigation bar, welcome message, and a prominent "Get Started" button that redirects users to the courses page.

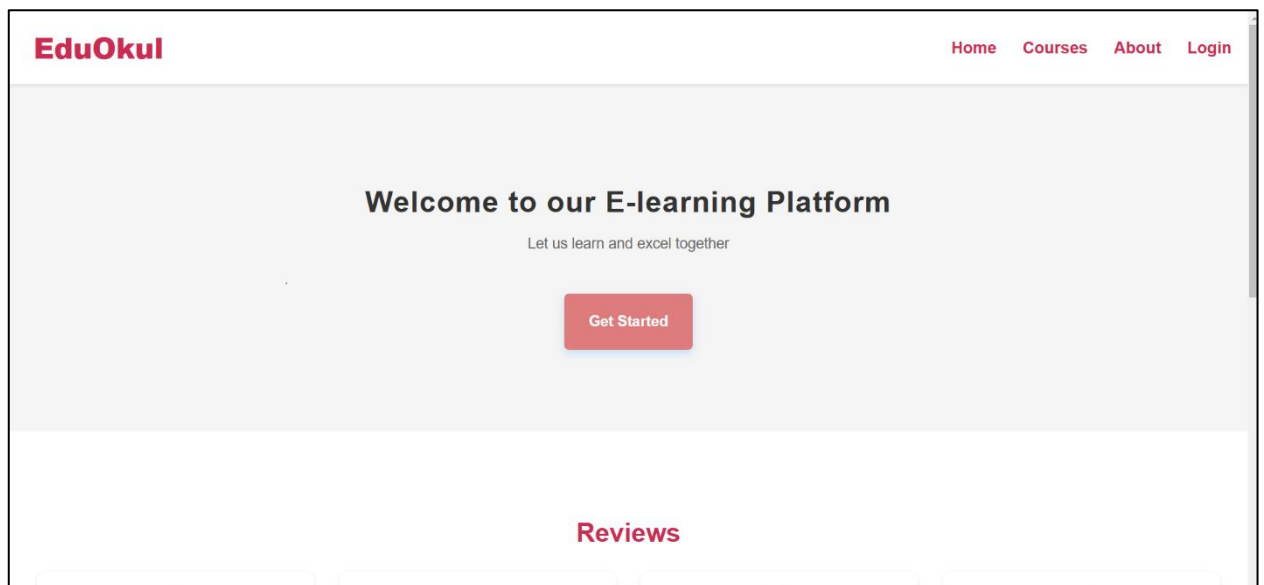


Figure 5.1.1: Home Page

#### 5.1.2 Home Page (Review Section):

The review section on the homepage displays manually curated testimonials from users, providing insights into the effectiveness and quality of EduOkul's courses and platform.

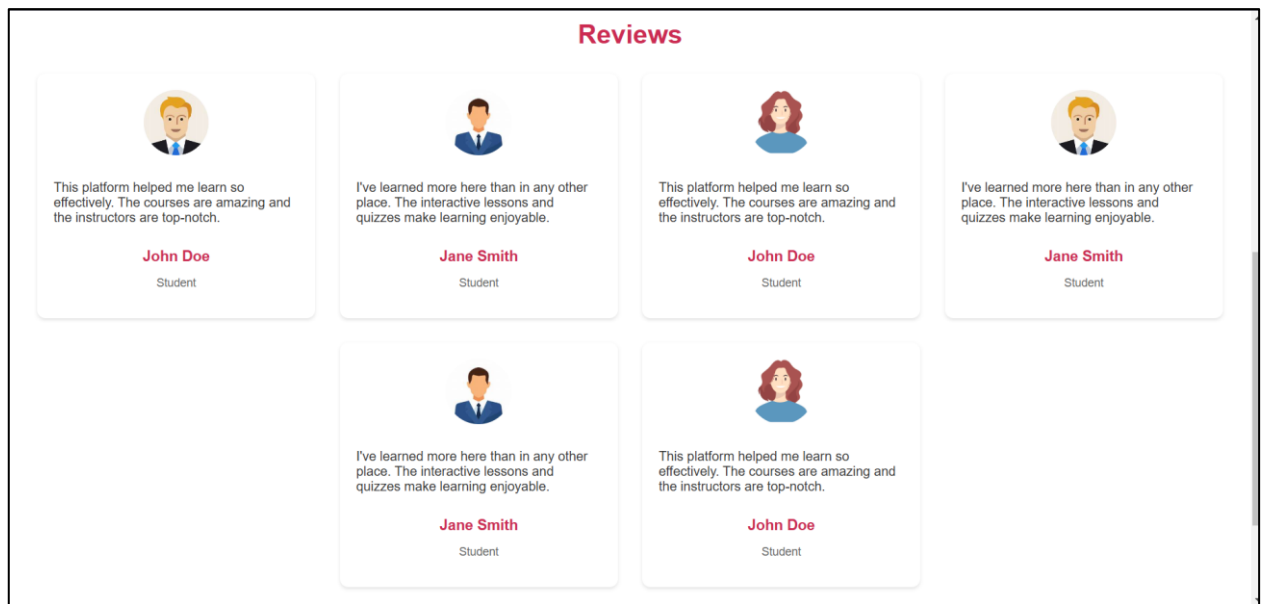


Figure 5.1.2: Review Section

## 5.2 Courses Page:

The courses page showcases a list of available courses with brief descriptions, images, and a "Study" button. Users must log in to access detailed course information and enrol.

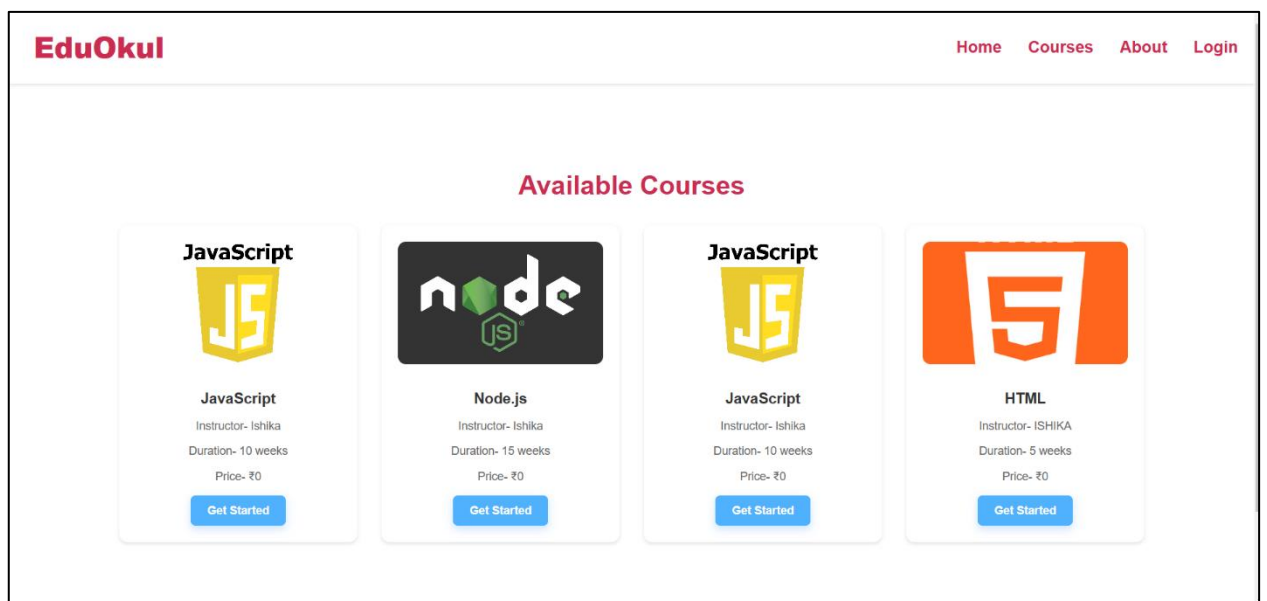
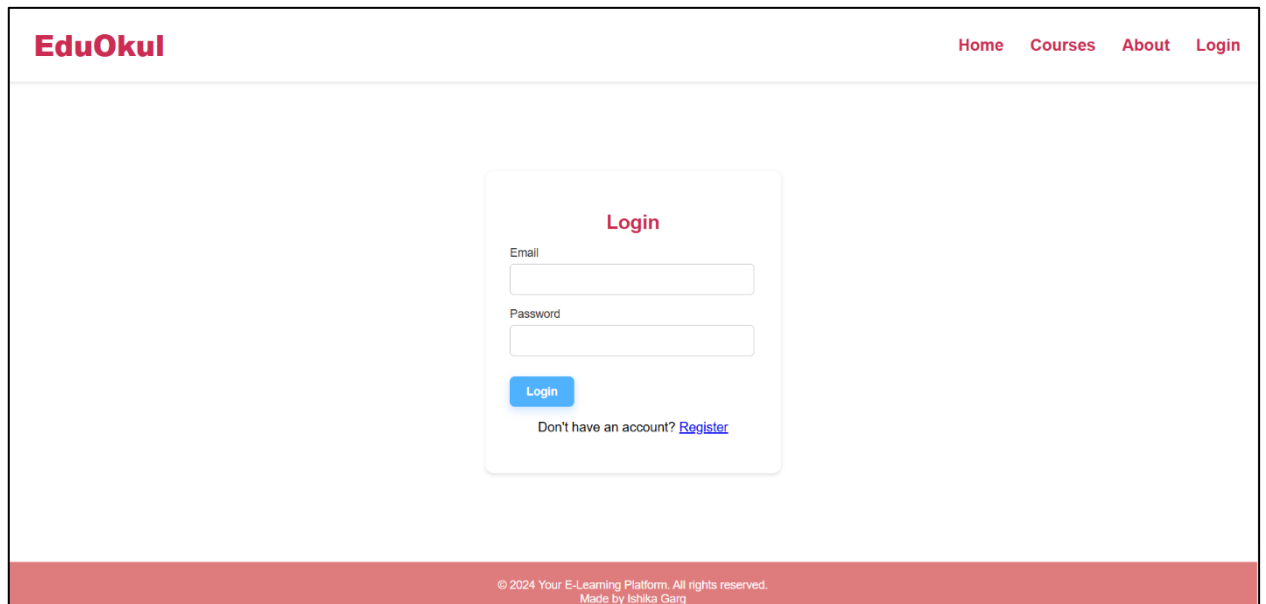


Figure 5.2: Courses Page

### 5.3 Login Page:

The login page allows users to enter their email and password to access their account. Once logged in, users are redirected to the homepage with a personalized welcome message.



The screenshot displays the login interface of the EduOkul platform. At the top left, the 'EduOkul' logo is visible. The top right navigation bar includes links for 'Home', 'Courses', 'About', and 'Login'. The central focus is a white login card with a red 'Login' title. It contains two input fields: 'Email' and 'Password'. Below these fields is a blue 'Login' button. A link for 'Register' is provided for users who do not have an account. The footer of the page contains the copyright notice: '© 2024 Your E-Learning Platform. All rights reserved. Made by Ishika Gang'.

Figure 5.3: Login Page

### 5.4 About Us Page:

The "About Us" page contains a brief overview of EduOkul, highlighting the platform's mission, vision, and the services it offers to learners and educators.



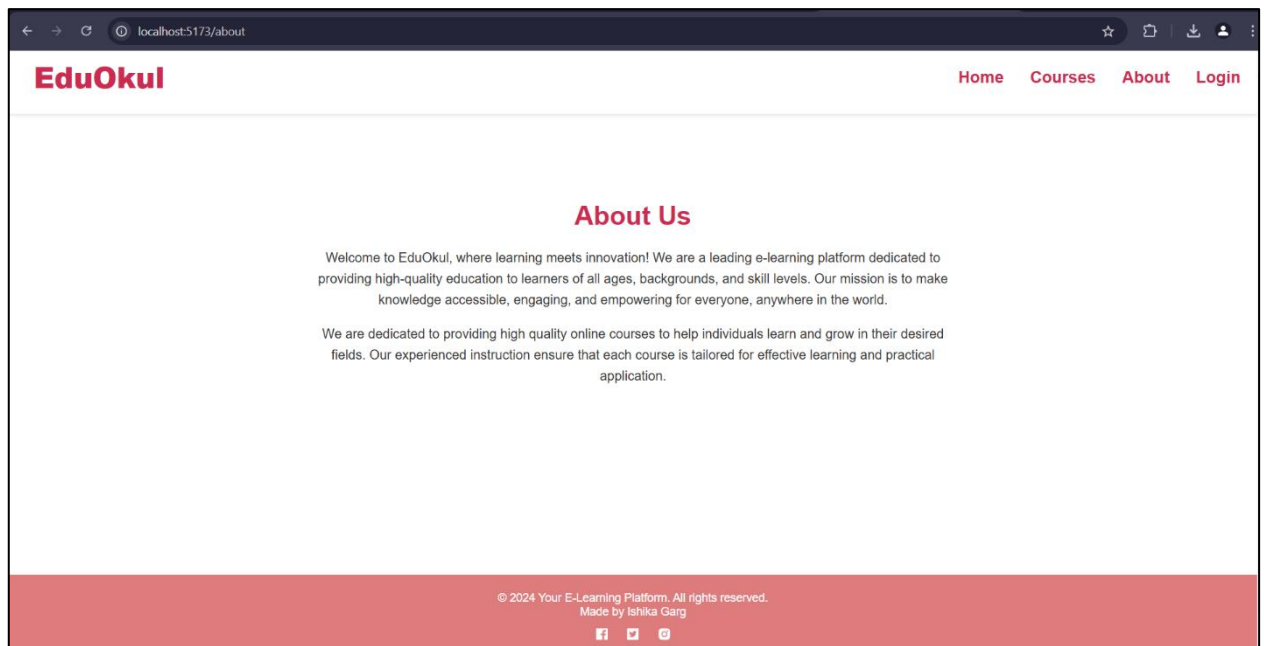


Figure 5.4: About Us Page

## 5.5 Account Page:

The account page shows the user's profile details, including name and email, along with options to view the dashboard or log out.

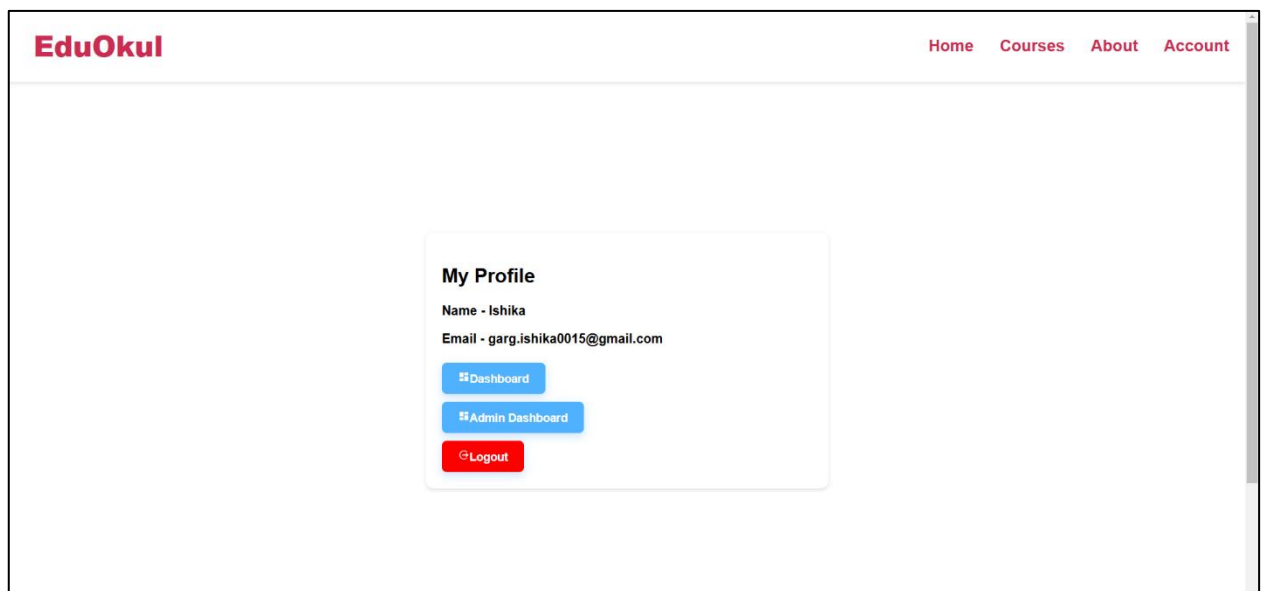


Figure 5.5: Account Page

## 5.6 Admin Dashboard Page (Home Page):

The admin's homepage includes a side menu with links to various sections, such as Users, Courses, and Logout, as well as a metrics section displaying total users, courses, and lectures.

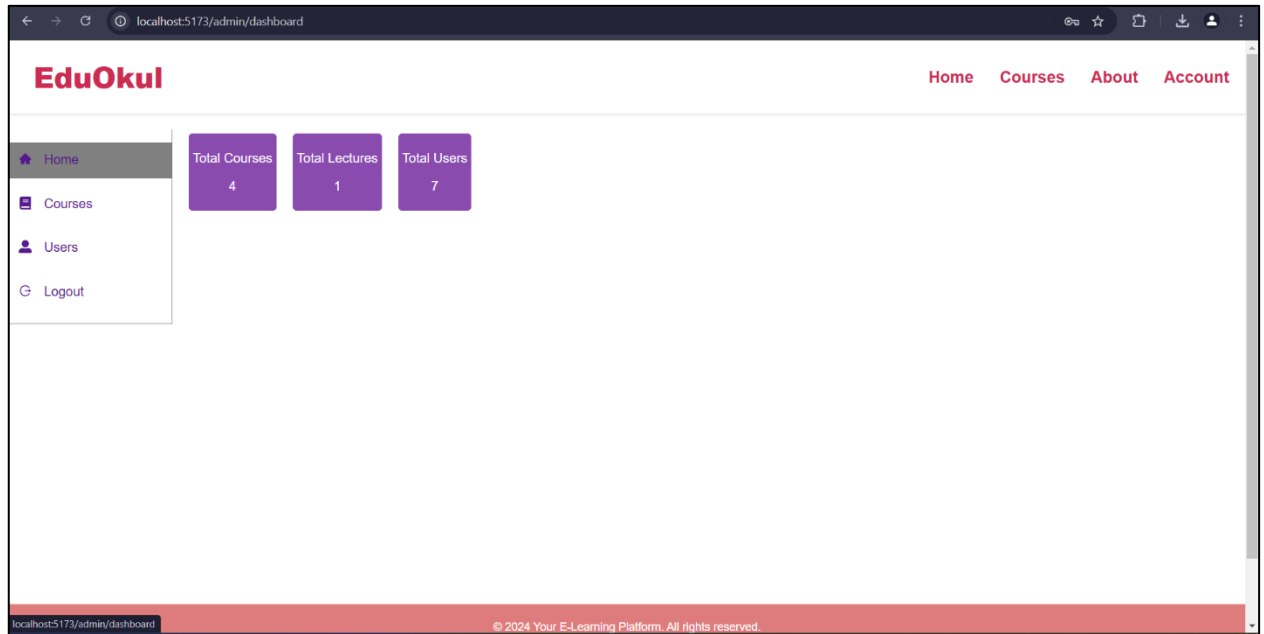


Figure 5.6: Admin Dashboard Home Page

### 5.7.1 Admin Dashboard Page (Courses Page):

The courses page within the admin dashboard lists all courses, displaying key details like course name, price, and duration, along with options to manage or delete them.

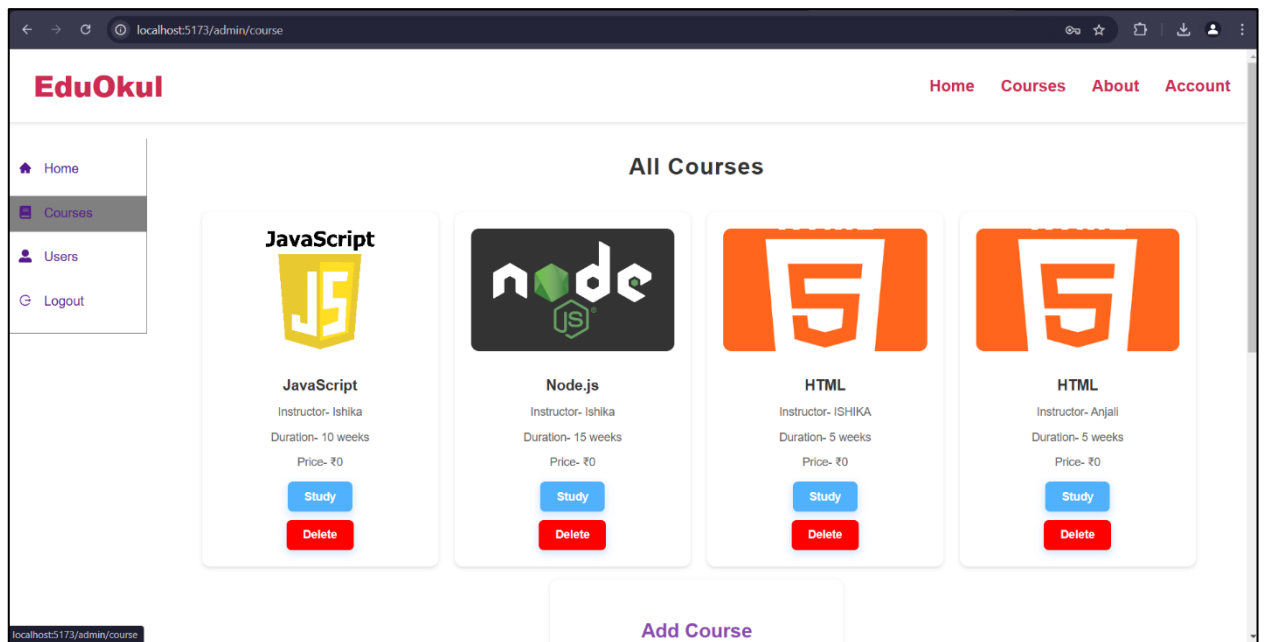


Figure 5.7.1: Admin Dashboard Courses Page

### 5.7.2 Admin Dashboard Courses Page (Adding new Course):

In the admin dashboard, the "Add New Course" page includes fields for the course title, description, price, duration, category, and an image upload button, enabling admins to create new courses.

The screenshot shows the 'Add Course' form in the admin dashboard. The form includes the following fields and controls:

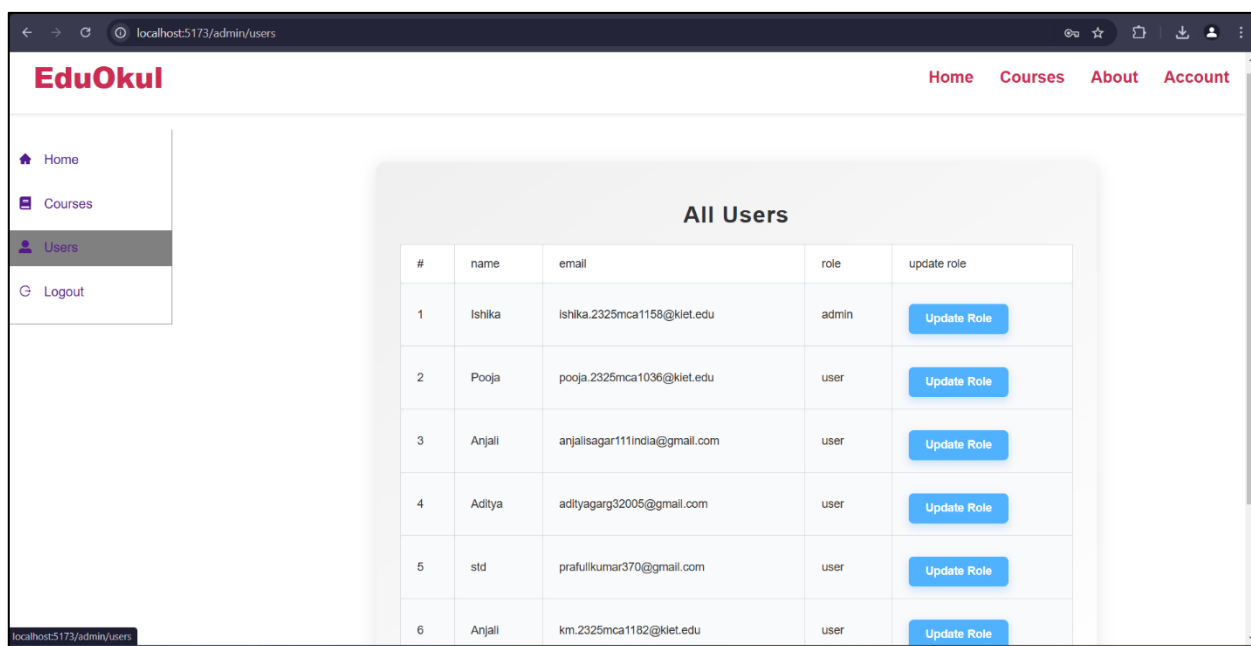
- Title**: Text input field.
- Description**: Text input field.
- Price**: Text input field.
- createdBy**: Text input field.
- Select Category**: Dropdown menu.
- Duration**: Text input field.
- Image Upload**: A button labeled 'Choose File' followed by the filename 'HTML.png'.
- Add**: A blue button to submit the form.

The footer of the page contains copyright information: '© 2024 Your E-Learning Platform. All rights reserved. Made by Ishika Garg' and social media icons for Facebook, Twitter, and Instagram.

Figure 5.7.2: Admin Dashboard Courses Page (Adding Lecture)

## 5.8 Admin Dashboard Users Page:

The users page allows admins to view a table of users, showing their names, roles, and emails, with an option to update each user's role from user to admin or vice versa.



#	name	email	role	update role
1	Ishika	ishika.2325mca1158@kiet.edu	admin	<button>Update Role</button>
2	Pooja	pooja.2325mca1036@kiet.edu	user	<button>Update Role</button>
3	Anjali	anjalisagar111india@gmail.com	user	<button>Update Role</button>
4	Aditya	adityagarg32005@gmail.com	user	<button>Update Role</button>
5	std	prafullkumar370@gmail.com	user	<button>Update Role</button>
6	Anjali	km.2325mca1182@kiet.edu	user	<button>Update Role</button>

Figure 5.8: Admin Dashboard Users Page

## 5.9 User Account Page:

The user account page displays the user's profile information, including their name and email, with links to view their dashboard and log out.

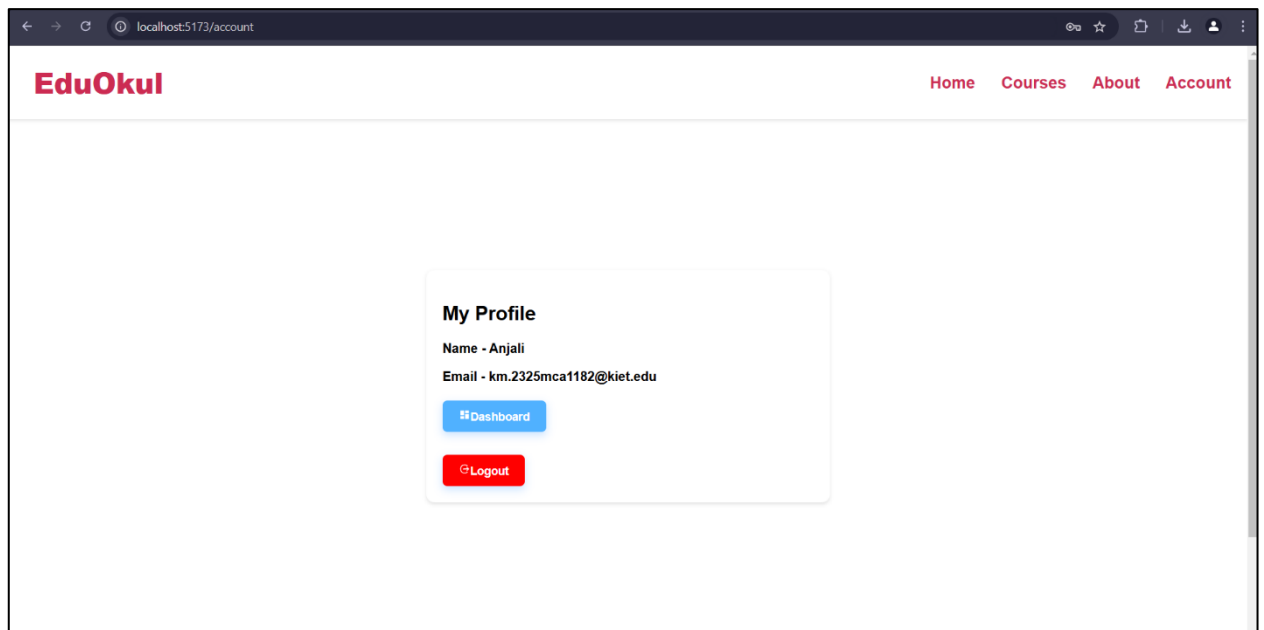


Figure 5.9: User Account Page

### 5.10 User Dashboard Page:

The user dashboard provides an overview of the courses in which the user is enrolled, allowing them to track their progress and access course materials.

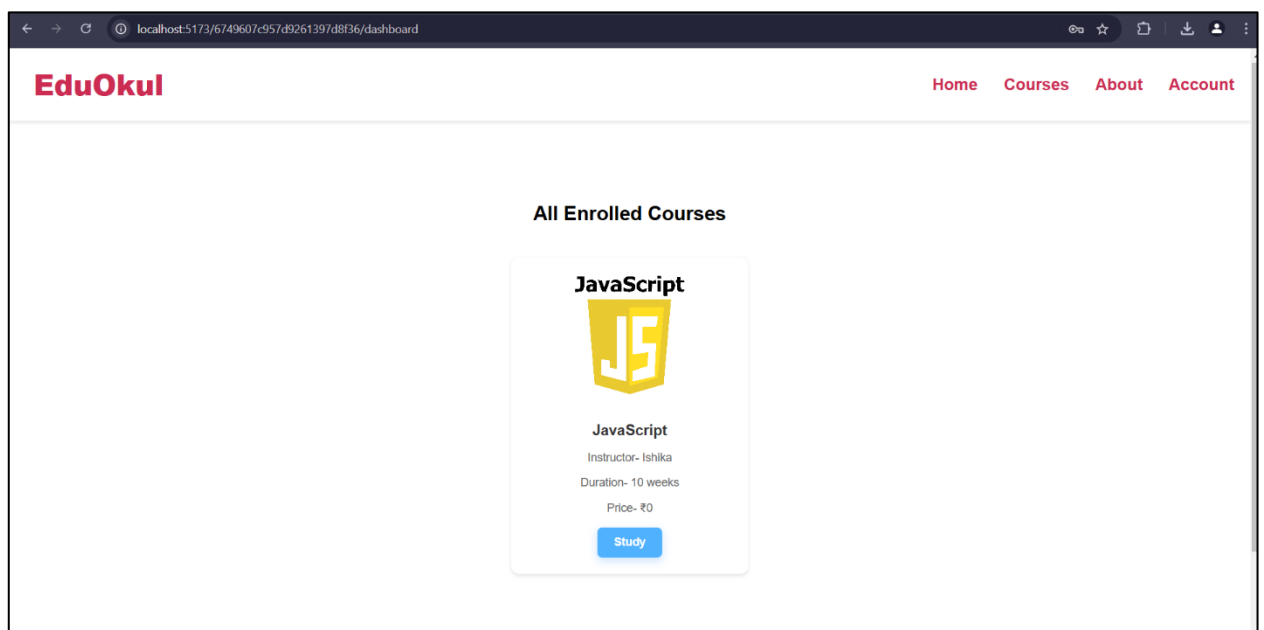


Figure 5.10: User Dashboard Page

## CHAPTER 6

### FUTURE SCOPE

EduOkul, as an evolving e-learning platform, has the potential to grow and incorporate numerous features that can enhance both user experience and administrative capabilities. Below are some key areas where EduOkul can be expanded and improved in the future:

#### 6.1 Advanced User Authentication and Authorization

- **Social Media Logins:** Implement social login features (such as Google, Facebook, or LinkedIn) for faster registration and login processes.
- **Multi-Factor Authentication (MFA):** Enhance security by adding an extra layer of authentication (e.g., OTPs or authentication apps).
- **Password Recovery (Forgot Password):** Currently, the platform lacks a password recovery mechanism. Implementing a "Forgot Password" feature would allow users to reset their passwords securely via email, improving the overall user experience.
- **Two-Factor Authentication (2FA):** Adding two-factor authentication will significantly enhance the security of user accounts. This could involve sending a verification code via SMS or email when a user logs in or performs sensitive actions.

#### 6.2 Course Recommendations and Personalization

- **AI-Based Course Recommendations:** Introduce an algorithm to recommend courses to users based on their past enrollments, searches, or interests.
- **User Progress Tracking:** Allow users to track their learning progress, showing which modules or lectures they have completed and what they still need to watch.

- **Custom Learning Paths:** Provide users with the ability to create personalized learning paths, selecting multiple courses that align with their specific goals.

### 6.3 Interactive Learning Features

- **Live Classes and Webinars:** Integrate live video streaming for instructors to host real-time classes and Q&A sessions. This could also include live assessments and peer interactions.
- **Interactive Quizzes and Assessments:** Implement quizzes and interactive tests after each course or lecture to help students evaluate their understanding.
- **Discussion Forums:** Integrate discussion boards where students can interact with peers and instructors, asking questions or discussing course material.
- **Peer Reviews:** Allow students to rate and review courses, giving feedback that could help others decide whether to enroll.

### 6.4 Gamification of Learning

- **Badges and Achievements:** Reward students with badges or certificates when they complete courses, modules, or quizzes, encouraging them to continue their learning journey.
- **Leaderboard:** Introduce a leaderboard that ranks students based on course completion, quiz scores, or learning milestones.

### 6.5 Mobile Application

- **EduOkul Mobile App:** Develop a mobile application for both Android and iOS to allow students to access courses, watch videos, and interact with the platform on-the-go. The app can include offline functionality to watch videos without an internet connection.
- **Push Notifications:** Implement push notifications to alert users about new courses, upcoming webinars, special promotions, or course updates.

### 6.6 Payment Gateway Integration

- **Multiple Payment Options:** Integrate more payment gateways (e.g., PayPal, Stripe, Razorpay) to facilitate international payments and offer various payment methods.

- **Subscription Models:** Implement subscription-based models where users can subscribe to a bundle of courses, rather than paying for individual courses. This could include monthly, quarterly, or yearly memberships.
- **Discount Coupons and Promotions:** Add the ability for admins to create discount codes or promotional offers to encourage course enrollments during sales events.

## 6.7 Collaborations and Certifications

- **Collaborate with Institutions:** Partner with universities, educational institutions, and industry experts to offer accredited certifications and degrees.
- **Certification Exams:** Allow students to take certification exams after completing a course, with an official certificate issued upon passing.
- **Corporate Training Programs:** Expand to offer specialized courses for corporate training, allowing businesses to purchase group licenses for their employees.

## 6.8 AI-Powered Tutor Assistance

- **AI Tutor for Students:** Use artificial intelligence to create a virtual assistant or tutor that helps students navigate through courses, answer questions, or clarify concepts when needed.
- **Automated Grading System:** Implement an AI-powered system that automatically grades assignments, quizzes, or exams based on predefined criteria, providing quick feedback to students.

## 6.9 Integration with Other Educational Platforms

- **Third-Party Course Integration:** Integrate with other e-learning platforms like Coursera, Udemy, or edX to offer a broader range of courses to EduOkul users.
- **LMS Integration:** Integrate EduOkul with existing Learning Management Systems (LMS) used by schools and universities, allowing them to manage their courses and students on one platform.

## 6.10 Scalability and Performance



- **Cloud Deployment:** As the user base grows, migrating the platform to a cloud-based architecture such as AWS, Google Cloud, or Azure will ensure that the system can handle more traffic and scale effortlessly.
- **Video Streaming Optimization:** Implementing a Content Delivery Network (CDN) to serve video lectures will speed up loading times and improve the user experience for students in different regions.

## **Conclusion**

EduOkul has a strong foundation and offers essential e-learning features, but there is significant potential for expansion. By implementing advanced features such as AI-based recommendations, mobile applications, gamification, and more personalized learning experiences, EduOkul can further enhance its position in the competitive e-learning market. The future scope of EduOkul focuses on providing a holistic and scalable learning ecosystem that supports students, instructors, and institutions worldwide.

## CHAPTER 7

### CONCLUSION

The **EduOkul** platform represents a comprehensive solution for online learning, designed with a focus on user experience, flexibility, and scalability. This project effectively demonstrates the integration of various technologies such as the **MERN stack (MongoDB, Express, React, Node.js)** to build an efficient, user-friendly system that caters to the needs of both students and administrators.

Throughout the development of the platform, several core features have been implemented, including a user-friendly registration and login system, course management, content delivery through video lectures, and a dedicated admin dashboard. These features have been designed with scalability in mind, ensuring the platform can evolve in response to future needs and demands.

Key highlights of the project include:

- A seamless **user authentication** system with login, registration, and OTP verification for enhanced security.
- A **dynamic courses page** with comprehensive course information, subscription features, and access to video lectures.
- An intuitive **admin dashboard** allowing the management of users, courses, and lectures, along with real-time statistics to monitor platform engagement.
- A **responsive user interface** that ensures compatibility across different devices and provides a smooth browsing experience for both students and administrators.

The **EduOkul** platform has the potential for further expansion with numerous opportunities for additional features, such as gamification, AI-based recommendations, social media integration, and mobile app development. These enhancements will help

improve the platform's capabilities and provide a more personalized and engaging learning experience for users.

In conclusion, **EduOkul** is a robust, scalable, and future-proof e-learning platform. It successfully addresses the core needs of online education, from easy course management to secure user authentication and a rich learning experience. With its foundation in the MERN stack, **EduOkul** is well-positioned to adapt to future technological trends and expand into new markets, making it a valuable tool for both educational institutions and learners. As the online education landscape continues to evolve, EduOkul can play a key role in shaping the future of digital learning.

## CHAPTER 8

### REFERENCES

#### WEB RESOURCES:

1. Express.js Documentation. (n.d.). *Express.js Docs*. Retrieved from <https://expressjs.com/>
  - a. Express.js helped in building the back-end RESTful APIs for user authentication, course management, and session management on EduOkul.
2. MongoDB Documentation. (n.d.). *MongoDB Manual*. Retrieved from <https://www.mongodb.com/docs/>
  - a. The MongoDB documentation was critical for understanding how to set up the database and structure data collections such as users, courses, and lectures within the EduOkul platform.
3. Node.js Documentation. (n.d.). *Node.js Docs*. Retrieved from <https://nodejs.org/api/all.html>
  - a. The Node.js documentation guided the server-side development of EduOkul, including building APIs, handling authentication, and serving the React app.
4. React Documentation. (n.d.). *ReactJS Docs*. Retrieved from <https://legacy.reactjs.org/docs/getting-started.html>
  - a. The official React documentation was used for creating dynamic user interfaces, managing state with hooks, and implementing routing within the front-end.