JEPPIAAR
ENGINEERING COLLEGE

JEPPIAAR NAGAR, RAJIVGANDHI SALAI

CHENNAI – 600119.

DEPARTMENT OF INFORMATION TECHNOLOGY

IV YEAR B.TECH – VII SEM

ACADEMIC YEAR 2024 - 25 (ODD SEM)

NM1042 - MERN Stack Powered by MongoDB

Online Learning Platform Report

(Naan Mudhalvan Project)

TEAM MEMBERS

1.Tharanika R -310821205102

2.Pravina Thiruvethi V -310821205066

3.Vanitha S -310821205104

4.Uma Maheswari P -310821205103

JEPPIAAR
ENGINEERING COLLEGE

JEPPIAAR NAGAR, RAJIVGANDHI SALAI, CHENNAI – 600119.

DEPARTMENT OF INFORMATION TECHNOLOGY

This is a Bonafide Record Work of _____

Register No. _____ submitted for the Anna University Practical Examination held on _____ in **NM1042 - MERN Stack Powered by MongoDB** during the year _____.

Signature of the Faculty-In-Charge **Signature of the HOD**

Date: _____ **Examiners Internal:** _____

External: _____

COLLEGE VISION & MISSION

Vision

To build Jeppiaar Engineering College as an institution of academic excellence in technological and management education to become a world class university.

Mission

- To excel in teaching and learning, research and innovation by promoting the principles of scientific analysis and creative thinking.
- To participate in the production, development and dissemination of knowledge and interact with national and international communities.
- To equip students with values, ethics and life skills needed to enrich their lives and enable them to contribute for the progress of society.
- To prepare students for higher studies and lifelong learning, enrich them with the practical skills necessary to excel as future professionals and entrepreneurs for the benefit of Nation's economy.

Program Outcomes

PO1	**Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO2	**Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	**Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO4	**Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO5	**Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
PO6	**The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO7	**Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
PO8	**Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO9	**Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO10	**Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO11	**Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO12	**Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

DEPARTMENT OF INFORMATION TECHNOLOGY

Vision

To produce engineers with excellent knowledge in the field of Information Technology through scientific and practical education to succeed in an increasingly complex world.

Mission

- To demonstrate technical and operational excellence through creative and critical thinking for the effective use of emerging technologies.
- To involve in a constructive, team-oriented environment and transfer knowledge to enable global interaction.
- To enrich students with professional integrity and ethical standards that will make them deal social challenges successfully in their life.
- To devise students for higher studies and perpetual learning, upgrade them as competent engineers and entrepreneurs for country's development.

Program Educational Objectives (PEOs)

PEO 1	To support students with substantial knowledge for developing and resolving mathematical, scientific and engineering problems
PEO 2	To provide students with adequate training and opportunities to work as a collaborator with informative and administrative qualities
PEO 3	To shape students with principled values to follow the code of ethics in social and professional life
PEO 4	To motivate students for extensive learning to prepare them for graduate studies, R&D and competitive exams
PEO 5	To cater the students with industrial exposure in an endeavor to succeed in the emerging cutting-edge technologies

Program Specific Outcomes

PSO1	Students are able to analyze, design, implement and test any software with the programming and testing skills they have acquired.
PSO2	Students are able to design algorithms, data management to meet desired needs, for real time problems through analytical, logical and problem solving skills.
PSO3	Students are able to provide security solutions for network components and data storage & management which will enable them to work in the industry ethically.

Course Outcomes (COs)

C407.1	Configure various virtualization tools such as Virtual Box, VMware workstation
C407.2	Design and deploy a web application in a PaaS environment
C407.3	Learn how to simulate a cloud environment to implement new schedulers.
C407.4	Install and use a generic cloud environment that can be used as a private cloud.
C407.5	Install and use Hadoop

Table of Contents

1. **Introduction** :

Project Title : Online Learning Platform.

The "Online Learning Platform" project focuses on developing a comprehensive e-learning solution using the MERN stack (MongoDB, Express.js, React.js, and Node.js).The platform is designed to provide a flexible, user-friendly, and scalable environment for students and educators alike, to interact, share knowledge, and facilitate the learning process efficiently. The platform supports features such as course creation, quizzes, student progress tracking.

Team Members and Roles:

- **Tharanika R [310821205102]: Team Lead and Backend Developer**

  Responsible for API development, server-side logic using Node.js, and database integration. Ensured the seamless interaction between frontend and backend components.

- **Pravina Thiruvethi V [310821205066]: Backend Developer**

  Worked on API development, implemented backend logic using

Node.js and Express.js, and collaborated on database management tasks.

- **Vanitha S [310821205104]: Database Manager and Frontend Developer**

  Designed and managed MongoDB schemas, ensured data consistency, optimized database queries, and contributed to frontend development using React.js.

- **Uma Maheswari P [3108212103]: Frontend Developer and Quality Assurance**

- Developed the user interface, implemented React components, ensured responsive design, conducted application testing, fixed bugs, and prepared project documentation.

Key Objectives:

1. **Landing Page**:

   Offers an engaging and interactive interface, presenting the platform's features and benefits.

2. **Registration and Login**:

   Secure user onboarding with role-specific access (student, teacher, admin).

3. **Student Features**:

   - Attend quizzes to assess knowledge.

- View trending courses and enroll in them.

4. **Teacher Features**:
    - Create courses, set pricing, and specify course types.
    - Introduce themselves as educators.
5. **Admin Dashboard**:

   Manage platform users, courses, and analyse system performance.

Scope:

- The platform targets both students and educators:
- **For Students**: Access to a variety of courses, quizzes, progress tracking, and a dashboard for course management.
- **For Educators**: Ability to create and manage courses, track student progress.
- Future possibilities include adding features such as live classes, forums, and AI-driven course recommendations.

2. Project Overview :

Purpose:

- The "Online Learning Platform" project aspires to democratize education by providing an accessible and inclusive online space that bridges the gap between students and educators. The

platform provides a way for students to access high-quality educational content remotely while offering educators the tools to create and manage courses effectively. The goal is to create an environment where learning is personalized, self-paced, and accessible to people from different backgrounds and geographical locations.

Features Overview

1) Landing Page:

Presents the platform's vision, mission, and key benefits.

Highlights trending courses and top educators.

Includes quick navigation to registration and login pages.

2)Registration and Login:

Registration: Role-based onboarding for students, teachers, and admins.

Login: Utilizes JWT for secure authentication and session management.

3)Student Dashboard:

View and enroll in trending courses based on demand.

Access purchased course materials and monitor learning progress.

Attempt quizzes to evaluate knowledge and track quiz scores.

4)Teacher Dashboard:

Create courses with details like course name, type, price, and educator information.

Upload resources such as videos, PDFs, and assessments.
View and manage student enrollment in their courses.

5)Admin Dashboard:

Monitor users and courses across the platform.
Review statistics, including course popularity and student participation.

Benefits:

- **Accessibility**: Students from remote areas or those unable to attend physical classes can access a wealth of knowledge online.
- **Personalization**: The platform recommends courses based on the user's interests and previous courses.

- **Scalability**: Built using the MERN stack, the platform is scalable to accommodate increasing numbers of users without performance issues.
- **Student Engagement**: Interactive quizzes, assignments, and a user-friendly interface engage students in a deeper learning process.

3. Architecture:

Overview:

- The architecture of the platform is built using the **MERN stack**—MongoDB for the database, Express.js for backend logic, React.js for the frontend user interface, and Node.js for server-side logic. This stack was chosen for its ability to handle dynamic, data-driven applications and provide a seamless user experience.

Frontend (React.js)

- **Landing Page**: A responsive and engaging UI showcasing trending courses.
- **Registration and Login Forms**: Role-specific validation for teachers, students, and admins.
- **Role-Based Dashboards**:

- Student: View courses, enroll, and attempt quizzes.
- Teacher: Create/manage courses and pricing.
- Admin: Manage users and system insights.

Backend (Node.js with Express.js)

- **API Design**:
  - Registration and Login APIs with role management.
  - Course APIs for adding, editing, and fetching course details.
  - Quiz APIs for creating and retrieving quiz data.
- **Middleware**: Authentication (JWT) and validation checks.

Database (MongoDB)

- **Schemas**:
  - **User Schema**: Role, name, email, password, courses enrolled/created.
  - **Course Schema**: Course name, type, price, educator, and description.
  - **Quiz Schema**: Questions, options, correct answers, and linked courses.

- **Progress Schema**: Student progress tracking for quizzes and course completion.

Flow of Functionality

1. **Student Flow**: Log in → View trending courses → Enroll → Access materials → Attempt quizzes.
2. **Teacher Flow**: Log in → Create a course → Add resources → Set pricing → Manage enrollments.

4. Setup Instructions :

System Requirements

- **Software**: Node.js (v14+), MongoDB (v4.4+), React.js.
- **Hardware**: Minimum 4GB RAM, 2 CPU cores, 10GB free storage.

Installation Process:

1. Clone the repository:

   git clone https://github.com/UmaMaheswari01/NM-Online-Learning-Platform.git

2. Backend setup:

   cd backend

   npm install

   npm start

3.Frontend setup:

cd frontend

npm install

npm start

4.Start MongoDB:

mongod

Access the application on http://localhost:3000.

Directory Structure

- **Frontend**: React.js components for dynamic views.
- **Backend**: Express.js routes, controllers, and authentication.
- **Database**: MongoDB collections for users, courses, and quizzes.

5. Folder Structure:

The project follows a modular architecture, separating the frontend, backend, and database-related configurations. The structure ensures clarity and maintainability for developers.

Project Root:

online-learning-platform/

├── backend/

├── frontend/

├── .env

├── package.json

├── README.md

Backend Folder

Contains all server-side logic, API endpoints, database configurations, and middleware.

backend/

├── config/ # Configuration files (e.g., database connection, JWT secret).

│ └── db.js

├── controllers/ # API request handlers.

│ ├── authController.js # Handles login and registration.

│ ├── courseController.js # Manages course-related operations.

│ ├── quizController.js # Handles quiz operations.

│ └── userController.js # Manages user operations.

├── middleware/ # Middleware for authentication and validation.

│ └── authMiddleware.js

├── models/ # MongoDB schemas for collections.

│ ├── Course.js

│ ├── Quiz.js

│ ├── User.js

```
│   └── Progress.js
├── routes/                 # API routes for different functionalities.
│   ├── authRoutes.js
│   ├── courseRoutes.js
│   ├── quizRoutes.js
│   └── userRoutes.js
├── .env                    # Environment variables (e.g., database URI, secret keys).
├── server.js               # Entry point for the backend server.
├── package.json            # Backend dependencies and scripts.
└── README.md               # Documentation for backend setup.
```

6. Running the Application

Frontend:

To run the frontend locally,

1. Navigate to the client directory in the terminal:

   cd client

2. Install the necessary dependencies:

   npm install

3. Start the frontend application:

   npm start

This will start the application locally on http://localhost:3000

Backend:

To run the backend locally,

1.Navigate to the server directory:

cd server

2.Install the required dependencies:

npm install

3.Start the backend server:

npm start

The backend will be available on http://localhost:5000.

7. API Documentation:

Endpoint	Method	Parameters	Description
/login	POST	username, password	Authenticates a user and returns a JWT.
/register	POST	username, email, password	Registers a new user
/courses	GET	None	Fetches a list of available courses.
/courses/:id	GET	id (path parameter)	Fetches course details by ID.

/profile	GET	None	Fetches the profile of the authenticated user.
/profile	PUT	email, password	Updates user profile details.

8. Authentication:

Authentication for the Online Learning Platform

Authentication is a critical component of any application, ensuring that users can securely log in and interact with the system based on their roles (student, teacher, admin). In the context of the Online Learning Platform, authentication will allow users to create accounts, log in, and access protected resources such as course materials, quizzes, and user profiles.

1. Authentication Process

The Online Learning Platform uses JWT (JSON Web Token) for authentication. JWT is a compact and self-contained way to securely transmit information between the client and server.

Steps Involved in the Authentication Process:

1. **User Registration:**
    - A new user (whether student or teacher) will first register on the platform by providing their username, email, and password.
    - The registration request is sent via a POST request to the /register endpoint of the backend.
    - Upon successful registration, the user is added to the system's database, and they can now log in using their credentials**.**

2. **User Login:**
    - The user logs in by entering their username (or email) and password on the login page.
    - A POST request is sent to the /login endpoint of the backend with the login credentials.
    - The server verifies the credentials against the stored values in the database.
    - If the credentials are valid, the server generates a JWT and sends it back to the client in the response.
    - If the credentials are invalid, an error message is sent to the client (e.g., "Invalid username or password").
3. **JWT Token Generation:**
    - The JWT token is generated using a secret key known only to the server. It consists of three parts:
        - Header: Contains metadata about the token, such as the signing algorithm.
        - Payload: Contains the user information (e.g., user ID, username, role). This is the part that is verified by the server.
        - Signature: Ensures the token has not been tampered with.
    - The server sends the generated JWT back to the client.
4. **Token Storage on Client:**
    - The client (browser) stores the JWT token in the localStorage or sessionStorage. This ensures the token persists even after page reloads.
    - The JWT is then attached to the Authorization header of all subsequent API requests to access protected routes on the server.
    - Example header format:

        Authorization: Bearer <JWT_TOKEN>

5. **Accessing Protected Resources:**
    - For every request made by the user to access protected resources (e.g., their profile, enrolled courses, course content), the client includes the JWT token in the Authorization header.

- The backend checks the token in the header to verify the user's identity and determine whether they have the necessary permissions to access the requested resource.
- If the token is valid, the server responds with the requested resource. If invalid, an error response is returned (e.g., "Unauthorized").

6. **Token Expiry:**
   - JWT tokens have an expiry time set by the server when they are generated. Once expired, the token is no longer valid, and the user will be logged out.
   - The user will need to log in again to obtain a new token.
   - Token expiry is usually a short duration (e.g., 1 hour) to enhance security.

2. Role-Based Authorization

In addition to authentication, the platform also employs role-based authorization. This means that depending on the user's role (student, teacher, or admin), different resources and features are accessible.

Roles and Permissions:

1. **Student:**
   - Can enroll in courses, track progress, attend quizzes, and view course materials.
   - Cannot create or modify courses.
2. **Teacher:**
   - Can create courses, set prices, and introduce new content to students.
   - Can also track student progress and provide assessments.
   - Cannot access admin-specific resources.
3. **Admin:**
   - Can manage users (students and teachers), courses, and monitor the overall platform activity.
   - Has full access to all features of the platform.

Role Verification:

- The user's role is stored in the JWT token's payload.

- Upon receiving a request, the backend decodes the JWT token and verifies the role field to ensure the user has the necessary permissions for the action they're trying to perform.

3. Handling Unauthorized Access

The platform must ensure that users who do not have the correct authentication or authorization are blocked from accessing sensitive data. For instance:

- If a student tries to access an endpoint meant for teachers (e.g., creating a course), the backend will return an HTTP 403 Forbidden status with an appropriate error message.
- Similarly, if a user tries to access a resource without a valid token, the server will return an HTTP 401 Unauthorized status.

4. Security Considerations

1. **Password Hashing:**
    - Passwords are stored in the database in a hashed form using algorithms like bcrypt. This ensures that even if the database is compromised, passwords are not exposed.
    - When the user logs in, the backend hashes the provided password and compares it to the stored hash in the database.
2. **Secure Communication:**
    - The platform uses HTTPS for secure communication between the client and server, ensuring that the JWT token and sensitive data are encrypted during transmission.
3. **Token Revocation:**
    - If a user logs out, the JWT token can be removed from the client's storage. However, because JWTs are stateless, tokens cannot be manually revoked from the server side unless using a more complex solution like blacklisting tokens in a database.

5. Logout Process

When a user decides to log out, the following steps occur:

1. **Client-Side:**
    - The JWT token is removed from localStorage or sessionStorage.
    - The user is redirected to the login page.
2. **Server-Side:**
    - The server does not need to explicitly "log out" the user since JWT tokens are stateless. The server only verifies the token during requests.

Authentication Flow:

1. Registration: User signs up with their username, email, and password.
2. Login: User enters credentials, and the server generates a JWT token.
3. Token Storage: The JWT token is stored in the browser's storage.
4. Access Protected Resources: User sends token with API requests to access protected routes.
5. Authorization: Backend verifies the user's role and permissions.
6. Logout: User clears the token, and the session ends.

9. User Interface

Key UI Pages:

- **Homepage**: Highlights featured courses and categories.
- **Course Listing**: Displays all courses, with filtering options.
- **Course Detail**: Detailed information about a specific course, including lessons and an enroll option.
- **User Profile**: Displays personal information and enrolled courses, with an option to update details.

10. Testing

Unit Testing:

- Frontend: Tested using Jest and React Testing Library.

Example: Test form validations during user login or registration.

- Backend: API endpoints tested with Mocha and Chai.

Example: Ensure /login returns the correct JWT for valid credentials.

Integration Testing:

- Tools like Postman and Supertest verify the flow between components.

    1. Use **Postman** to send requests to the backend and inspect responses, checking whether the user is successfully registered in the database.
    2. **Supertest** can be used for writing test scripts that simulate requests from the frontend to the backend.

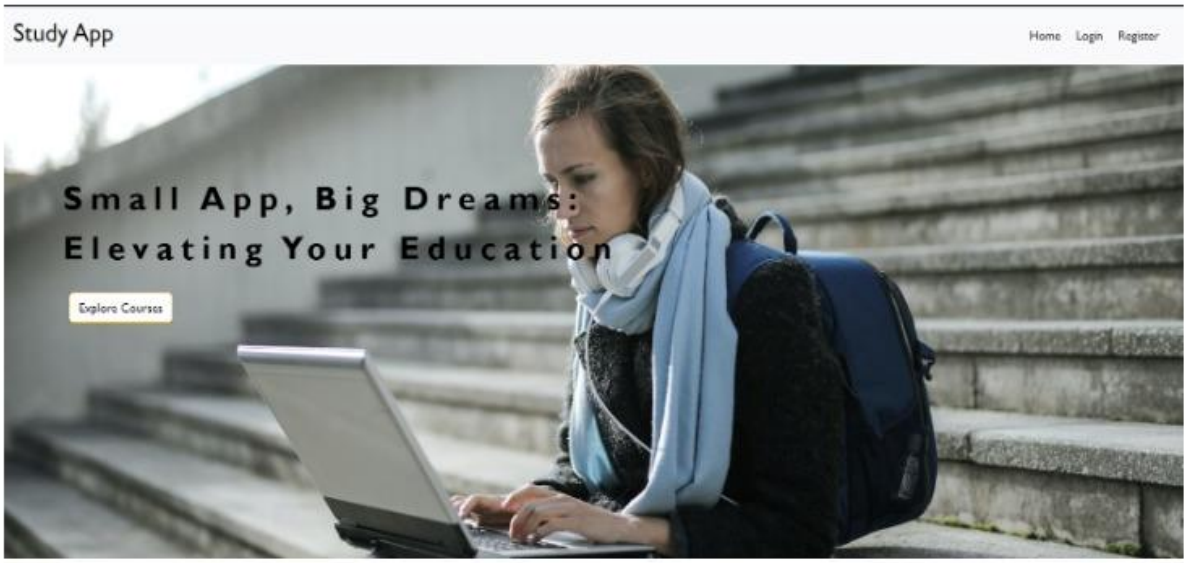
Example: Test user registration flow from frontend input to backend database updates.

End-to-End Testing:

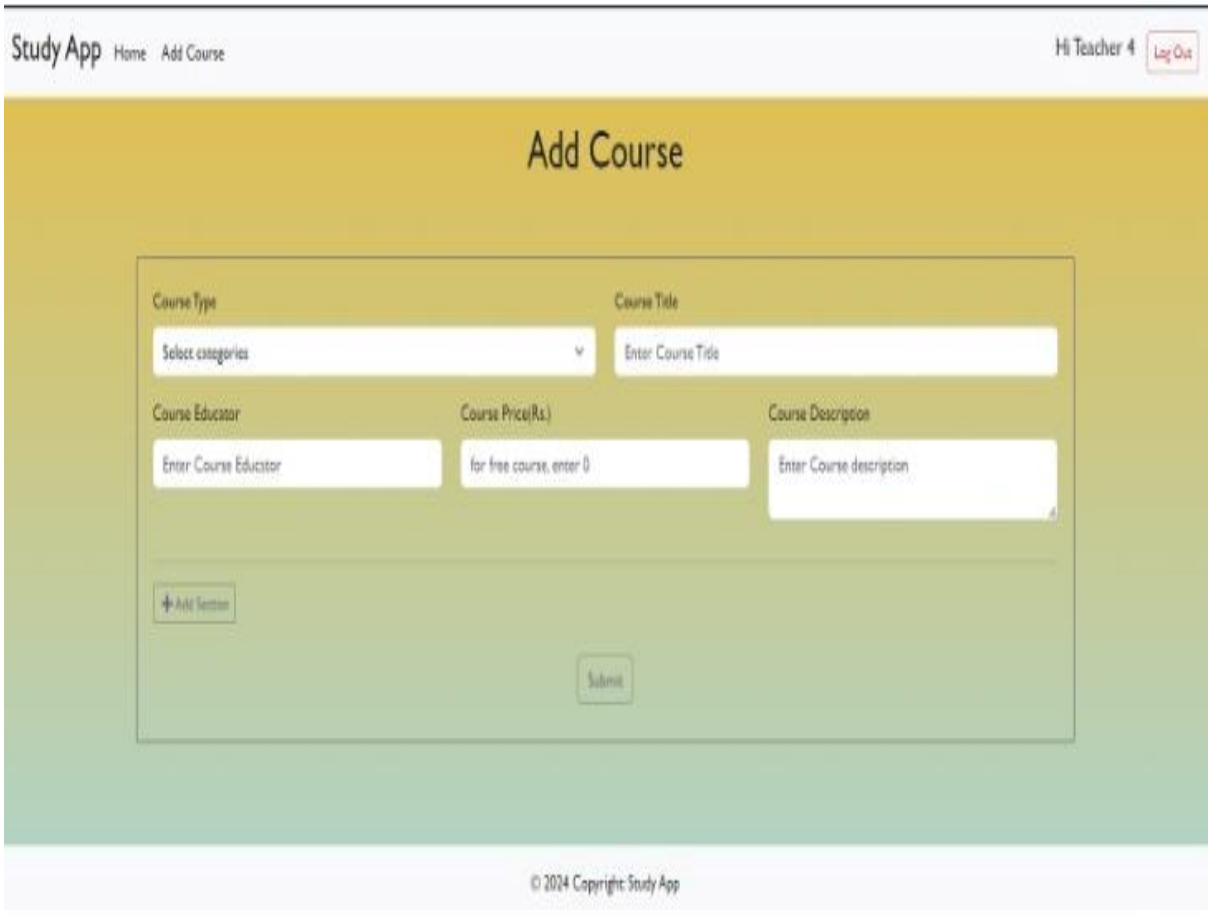
- Cypress is used to simulate real user workflows and ensure full-system reliability.

Example: Validate the enrollment process from course listing to confirmation.

11. Screenshots or Demo:


Study App
Home Login Register
Small App, Big Dreams:
Elevating Your Education
Explore Courses


Trending Courses
Search By: title
All Courses
Modules
Title: l
Description: t
Title: n
Description: h
many more to watch..
© 2024 Copyright: Study App


Study App
Home
Add Course
Hi Teacher 4
Log Out
Add Course
Course Type
Select categories
Course Title
Enter Course Title
Course Educator
Enter Course Educator
Course Price(Rs.)
for free course, enter 0
Course Description
Enter Course description
Add Section
Submit
© 2024 Copyright: Study App


Study App
Home
Enrolled Courses
Hi Student 1
Log Out
Welcome to the course: Frontend Course
PLAY VIDEO
DOWNLOAD CERTIFICATE
QuizApp
QUIZ
iStock
Credit: akuzone
0:00 / 4:45
© 2024 Copyright: Study App

12. Known Issues:

While the Online Learning platform is functional, there are a few known issues that users should be aware of: -

- **Login page** may not load correctly on slow internet connections.
- **The course progress** bar does not always update in real time.
- Some **notifications** may not appear properly on mobile devices.
- **Progress tracking** delays under certain conditions.
- Minor **UI** inconsistencies on smaller screens.
- **Quiz Timer Synchronization Issue:**
  During quizzes, the timer occasionally fails to sync correctly with server time, leading to inaccurate countdowns.
- **Duplicate Notifications:**
  Users might receive duplicate notifications when an event (like course enrollment) is triggered multiple times.
- **Course Content Loading Delay:**
  Course videos or resources might take longer to load on slower networks or during high traffic periods.
- **Search Bar Performance:**
  The search functionality may not return results accurately for partially typed queries.
- **Incomplete Logout:**
  In rare cases, users remain logged in even after clicking the logout button due to session token persistence.
- **Payment Integration Bugs:**
  Some payment transactions fail to process properly when using certain payment gateways.
- **Error Messages Lack Specificity:**
  Error messages displayed to users are sometimes generic and lack details about the actual issue.

- These additional points can further highlight areas for improvement in the application while maintaining a professional tone in the report.

13. Future Enhancements:

The following features are planned for future releases of the Online Learning platform:

1.Recommendation System:

- Implement a system to suggest courses to users based on their interests and past activity.

2.Multi-Language Support: -

- Introduce support for multiple languages to cater to a broader user base.

3.Real-Time Interaction:

- Add a real-time chat feature allowing students to interact with instructors and peers during courses.

4.Mobile Application:

- Develop a mobile version of the platform to enhance user experience on smartphones and tablets.

5.Gamification Features:

- Implement badges, achievements, and leaderboards to increase user engagement, particularly for students completing courses and quizzes.

6.Advanced Search and Filters:

- Add more refined search options, such as filtering by course duration, difficulty level, instructor, or student reviews.

7.Discussion Forums:

- Introduce forums or discussion boards where students and teachers can interact, ask questions, and discuss course material.

8.Offline Mode:

- Enable users to download course materials or videos to access them offline, which is especially useful for users with limited internet access.

9.Multimedia Support for Courses:

- Allow for the integration of different types of media (e.g., quizzes with audio/video components, interactive simulations) to make courses more engaging.

10.Collaborative Learning:

- Create group learning or collaborative project features where students can work together on assignments or projects.

11.Enhanced Payment Gateway Integration:

- Add more payment methods (e.g., crypto, more local payment options) to cater to users from diverse geographical locations.