Learning Management System

CAPSTONE PROJECT

Submitted in partial fulfillment of the requirements for the degree of

Bachelor of Computer Applications

in

Department of Computer Applications

By

SHAMBHU PANDEY
22BCA0028

Under the guidance of

Vijay Anand R
Professor Grade 1 Score

School of Computer Science Engineering and Information Systems

VIT, Vellore

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

April, 2025

Learning Management System

CAPSTONE PROJECT

Submitted in partial fulfillment of the requirements for the degree of

Bachelor of Computer Applications

in

Department of Computer Applications

By

SHAMBHU PANDEY
22BCA0028

Under the guidance of

Vijay Anand R
Professor Grade 1 Score

School of Computer Science Engineering and Information Systems

VIT, Vellore

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

April, 2025

DECLARATION

I hereby declare that the Capstone Project entitled "Learning Management System" submitted by me, for the award of the degree of **Bachelor of Computer Applications (BCA)** *Department of Computer Applications School of Computer Science, Engineering and Information Systems* VIT is a record of bonafide work carried out by me under the supervision of Vijay Anand R , Professor Grade 1 SCORE , VIT Vellore

I further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree ordiploma in this institute or any other institute or university.

Place: Vellore

Date:

Shaubhee Pandey

Signature of the Candidate

CERTIFICATE

This is to certify that the **Capstone Project** entitled Learning Management System submitted by "SHAMBHU PANDEY 22BCA0028" , SCORE, VIT, for the award of the degree of *Bachelor of Computer Applications in Department of Computer Applications*, is a record of bonafide work carried out by him / her under my supervision during the period, 13. 12. 2024 to 17.04.2025, as per the VIT code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute orany other institute or university. The Capstone project fulfills the requirements and regulations ofthe University and in my opinion meets the necessary standards for submission.

Place: Vellore
Date:

[illegible] 5/04/25

Signature of the VIT-SCORE - Guide

Internal Examiner **External Examiner**

Head of the Department
Department of Computer Application

Learning Management System

CAPSTONE PROJECT

Submitted in partial fulfillment of the requirements for the degree of

Bachelor of Computer Applications

in

Department of Computer Applications

By

SHAMBHU PANDEY
22BCA0028

Under the guidance of

Vijay Anand R
Professor Grade 1 Score

School of Computer Science Engineering and Information Systems

VIT, Vellore

April, 2025

DECLARATION

I hereby declare that the **Capstone Project** entitled “**Learning Management System**" submitted by me, for the award of the degree of **Bachelor of Computer Applications (BCA)** *Department of Computer Applications School of Computer Science, Engineering and Information Systems* VIT is a record of bonafide work carried out by me under the supervision of **Vijay Anand R , Professor Grade 1 SCORE , VIT Vellore**

I further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree ordiploma in this institute or any other institute or university.

Place: Vellore

Date:

Signature of the Candidate

CERTIFICATE

This is to certify that the **Capstone Project** entitled **Learning Management System** submitted by “**SHAMBHU PANDEY 22BCA0028”** , SCORE, VIT, for the award of the degree of ***Bachelor of Computer Applications in Department of Computer Applications***, is a record of bonafide work carried out by him / her under my supervision during the period, 13. 12. 2024 to 17.04.2025, as per the VIT code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute orany other institute or university. The Capstone project fulfills the requirements and regulations ofthe University and in my opinion meets the necessary standards for submission.

Place: Vellore

Date:

Signature of the VIT-SCORE - Guide

Internal Examiner **External Examiner**

Head of the Department
Department of Computer Application

ACKNOWLEDGEMENT

It is my pleasure to express with a deep sense of gratitude to my Capstone project guide **Dr. Vijay Anand R Professor Grade 1 Score ,**School of Computer Science Engineering and Information Systems, Vellore Institute of Technology, Vellore for **his/her** constant guidance, continual encouragement, in my endeavor. My association with **him/her** is not confined to academics only, but it is a great opportunity on my part to work with an intellectual and an expert in the field of **Web development .**

"I would like to express my heartfelt gratitude to Honorable Chancellor **Dr. G Viswanathan**; respected Vice Presidents **Mr. Sankar Viswanathan**, **Dr. Sekar Viswanathan**, Vice Chancellor **Dr. V. S. Kanchana Bhaaskaran**; Pro-Vice Chancellor **Dr. Partha Sharathi Mallick**; and Registrar **Dr. Jayabarathi T**.

My whole-hearted thanks to Dean **Dr. Daphne Lopez**, School of Computer Science Engineering and Information Systems, Head, Department of Computer Applications **Dr. E Vijayan**, MCA Project Coordinator **Dr. Senthil Kumar T**, SCORE School Project Coordinator **Dr. Thandeeswaran R**, all faculty, staff and members working as limbs of our university for their continuous guidance throughout my course of study in unlimited ways.

It is indeed a pleasure to thank my parents and friends who persuaded and encouraged me to take up and complete my capstone project successfully. Last, but not least, I express my gratitude and appreciation to all those who have helped me directly or indirectly towards the successful completion of the capstone project.

Place: Vellore

Date: **SHAMBHU PANDEY**

Executive Summary

The **Learning Management System (LMS)** is a comprehensive full-stack web application designed to deliver a secure, scalable, and engaging digital learning experience for both students and educators. It effectively overcomes the limitations of traditional LMS platforms—such as high operational costs, limited customization, weak communication infrastructure, and inadequate coding tools—by offering a modern, open-source, and highly adaptable solution tailored to current educational needs.

The platform is developed using **React.js** for the frontend, enhanced with **Redux** for efficient state management, and styled with **Material UI** and **Bootstrap** to ensure a sleek, responsive user interface. On the backend, **Node.js** and **Express.js** power the server-side logic, with **RESTful APIs** and **JWT-based authentication** providing secure and seamless communication. **MongoDB** is used for data storage, delivering both high scalability and flexibility.

Educators benefit from features such as intuitive course creation, custom assessments, live video classes, and secure content distribution, while **Students** gain access to rich learning materials, live class participation, assignment submissions, instructional videos, downloadable resources, and profile management tools. The platform also includes live coding and testing environments to enhance practical learning experiences.

To ensure data protection and user privacy, the system implements **role-based access control** and **SSL/TLS encryption**. It is deployed on the cloud, making it accessible from any internet-enabled device. Built following the **Agile development methodology**, the LMS evolves through structured phases of requirement analysis, design, development, testing, and deployment. This project ultimately offers a dynamic, cost-effective, and future-ready solution for educational institutions seeking to modernize their learning infrastructure.

TABLE OF CONTENTS

3. TECHNICAL SPECIFICATION

4. DESIGN APPROACH AND DETAILS (As Applicable)

5. SCHEDULE, TASKS, AND MILESTONES

6. PROJECT DEMONSTRATION

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Full Form
LMS	Learning Management System
UI	User Interface
CRUD	Create, Read, Update, Delete
API	Application Programming Interface
MVC	Model-View-Controller
HTTPS	HyperText Transfer Protocol Secure
JWT	JSON Web Token
REST	Representational State Transfer
CI/CD	Continuous Integration/Continuous Deployment
UX	User Experience
SSO	Single Sign-On
LMS Instructor	Learning Management System Instructor
CMS	Content Management System
SCORM	Sharable Content Object Reference Model
FTP	File Transfer Protocol
S3	Amazon Simple Storage Service
JSON	JavaScript Object Notation

Symbols and Notations

Symbol	Meaning
Δt	Response Time
λ	System Load (Requests per Second)
μ	Server Processing Rate
P	Performance Score
τ	Average Session Duration
σ	Data Processing Latency
C	Number of Concurrent Users
E	Error Rate
η	System Efficiency
ρ	Server Utilization Factor
β	Cache Hit Ratio
φ	Encryption Strength
κ	User Engagement Score
γ	Bandwidth Consumption per User
θ	Session Timeout Interval

CHAPTER 1

Introduction

A Learning Management System (LMS) is a digital platform that facilitates the storage, organization, and distribution of educational materials. It serves as a centralized system where educators manage courses and students access resources, enhancing the overall learning experience. Unlike traditional methods that rely heavily on printed materials such as textbooks and notes, an LMS digitizes the learning process, making it more accessible, structured, and efficient.

Educational institutions often face challenges in distributing course materials, managing assignments, and ensuring student engagement. Instructors need a streamlined way to organize their lessons, track student progress, and facilitate communication. Students, on the other hand, require a platform that provides easy access to materials, simplifies assignment submissions, and allows seamless interaction with instructors. A well-designed LMS addresses these challenges by creating a structured learning environment that enhances the educational process.

With the increasing shift toward digital education, institutions are recognizing the need for an LMS that is adaptable and user-friendly. Existing LMS platforms, while offering essential features, often include unnecessary complexities such as excessive automation and AI-driven analytics, which may not align with the actual needs of students and educators. Some systems require advanced technical knowledge, making them difficult to operate, especially for institutions that lack IT expertise.

Furthermore, the platform will support secure data storage, ensuring that student and faculty information remains protected. Many existing LMS platforms fail to implement proper security measures, leaving data vulnerable to breaches. This LMS will incorporate authentication mechanisms to safeguard user data while maintaining ease of access.

In addition to improving accessibility, the LMS will integrate essential features such as course management, assignment tracking, discussion forums, and assessment tools. Instructors will be able to efficiently create and manage courses, while students can easily navigate through study materials and submit assignments. The communication system will provide a structured way for students to engage with instructors, making the learning process more interactive and collaborative.

Ultimately, the goal of this project is to create a cost-effective, secure, and scalable LMS that enhances the learning experience for both students and instructors. By focusing on simplicity, usability, and efficiency, this system will bridge the gap between traditional education and modern digital learning, ensuring that institutions can adopt technology without unnecessary complications.

1.1 Objective

The primary objective of this project is to develop a structured, reliable, and accessible LMS that simplifies course management, assignment tracking, and student-instructor communication. The system is designed to be scalable, flexible, and user-friendly, ensuring it meets the needs of educational institutions of all sizes.

Key Objectives:

- **Unified Course Management**
    - A well-structured system for instructors to create, manage, and distribute course content efficiently.
    - Eliminates confusion related to course materials and provides students with a clear academic roadmap.
    -
- **Easy Access to Learning Materials**
    - Ensures students can access notes, presentations, and reference materials anytime.
    - Reduces dependency on physical copies and enhances learning flexibility.
    -
- **Secure Assignment Submission System**
    - Students can submit assignments directly through the platform.
    - Prevents lost submissions and allows for systematic evaluation by instructors.
    -
- **User-Friendly Communication**
    - Provides a structured discussion forum for student-instructor interactions.
    - Enables students to ask questions and receive real-time feedback.
    -
- **Simple and Scalable Design**
    - Unlike complex LMS platforms, this system offers an intuitive and easy-to-use interface.
    - Requires minimal technical knowledge for implementation.
    -
- **Efficient Instructors**
    - Instructors can, manage enrollments, and review assignments seamlessly..
    -
- **Cost-Effective Solution**
    - Developed using open-source frameworks, reducing operational costs.
    - Ensures scalability, security, and affordability for institutions.

By incorporating these features, this LMS enhances the educational experience by providing a well-structured, secure, and user-friendly learning environment.

1.2 Motivation

The increasing demand for a structured and efficient digital learning platform has driven the development of this LMS. Traditional education systems face multiple challenges, including inefficient content distribution, low student engagement, and poor academic management. Additionally, many existing LMS platforms have complex technical requirements, making them difficult for users with minimal digital experience.

Challenges in Traditional Learning Systems:

- **Lack of Centralized Learning Resources**
  - Students struggle to access all course materials in one place.
  - Fragmented resources lead to confusion and inefficiency.
- **Rigid Testing Methods**
  - Many LMS platforms do not offer customizable testing options.
  - Instructors have limited flexibility in evaluating student performance.
- **Limited Collaboration Features**
  - Communication is often restricted to emails or external chat applications.
  - Real-time student-instructor interaction is challenging.
- **Unorganized Assignment Tracking**
  - Without a proper submission and review system, managing academic deadlines becomes difficult.
  - Students and instructors lack a structured approach to assignment tracking.
- **Security Concerns**
  - Many LMS platforms lack strong encryption and authentication measures.
  - Student and instructor data may be vulnerable to breaches.

This Project Addresses These Challenges By:

- Centralizing learning resources for easy access.
- Offering flexible assignment submission and evaluation options.
- Providing dedicated discussion forums for seamless communication.
- Implementing robust security features to protect user data.

By improving accessibility, security, and usability, this LMS empowers educational institutions with a practical and efficient digital learning solution.

1.3 Background

The evolution of education has been shaped by technological advancements. Traditional classroom-based learning heavily relied on printed textbooks, handwritten notes, and in-person instruction. However, the rise of digital platforms has transformed education into a more interactive and accessible process.

Key Developments in LMS Platforms:

- **Introduction of Digital Learning**
  - Early 2000s: Basic LMS platforms emerged to upload learning materials and track student progress.
- **Expansion into Higher Education**
  - Universities and colleges adopted LMS platforms to manage courses, assignments, and student engagement.
- **Growing Demand for Customizable LMS**
  - The shift to hybrid and online learning increased the need for flexible, adaptive LMS solutions.

Why This LMS is Different:

- **Prioritizes Simplicity**
  - Many existing LMS platforms are overly complex due to excessive automation.
  - This system focuses on user-friendly functionality rather than technical intricacies.
- **No AI-Based Tracking**
  - Unlike AI-driven LMS solutions that introduce unnecessary automation, this LMS maintains a human-centric approach.
- **Cost-Effective and Secure**
  - Built on an open-source framework to reduce costs while ensuring scalability and security.
- **Designed for Academic Needs**
  - Tailored specifically for educational institutions, providing a structured, practical, and efficient digital learning experience.

By bridging the gap between traditional and digital education, this LMS serves as a complete academic management solution that enhances student engagement, simplifies course distribution, and ensures a secure learning environment.

2. PROJECT DESCRIPTION AND GOALS

2.1 Key Features of the Project

This Learning Management System (LMS) is designed to streamline course management, assignment tracking, student-instructor communication, and overall digital learning processes. It integrates multiple features that enhance the efficiency of learning environments while ensuring ease of access, security, and scalability. Below are the key features of this LMS:

1. Centralized Course Management

- Instructors can create, update, and organize course materials in a structured manner.
- Students can easily access all learning resources, including lecture notes, videos, and reference materials, from a single platform.
- Courses can be categorized into different subjects, semesters, or academic levels to maintain clarity.

2. Secure and Efficient Assignment Submission

- Students can submit assignments directly through the LMS, ensuring a systematic approach to academic tasks.
- The system automatically records submission timestamps, preventing late submission disputes.
- Instructors can track, grade, and provide feedback efficiently, enhancing the evaluation process.

4. User Authentication and Role-Based Access

- The LMS includes a secure login system with role-based access, ensuring that students, instructors, and administrators have the appropriate permissions.
- Multi-layer authentication enhances data security and protects sensitive academic information.
- Admin users can manage user roles, monitor activities, and enforce security policies.

6. Scalability and Customization

- The LMS is designed to be scalable, supporting educational institutions of different sizes, from small coaching centers to large universities.
- Institutions can customize the platform to align with their academic structure and specific requirements.

These features collectively enhance the efficiency, security, and accessibility of digital education, making the LMS an invaluable tool for educational institutions.

2.2 Project Goals

The primary goal of this project is to create an innovative and efficient Learning Management System that overcomes the limitations of traditional learning methods while providing a structured, secure, and user-friendly digital learning experience. The project aims to bridge the gap between instructors and students by ensuring seamless communication, resource sharing, and progress tracking.

Key Goals of the Project:

1. Enhancing Accessibility and Inclusivity

- Ensure 24/7 access to learning materials, allowing students to study at their own pace.
- Provide an intuitive and easy-to-use interface for students and instructors, even for those with minimal technical knowledge.
- Support multiple devices, including mobile phones and tablets, for on-the-go learning.

2. Improving Course Management Efficiency

- Develop a system where instructors can easily create, update, and organize course content.
- Enable students to find all academic resources in one place, reducing confusion and improving productivity.

4. Providing Secure and Organized Assignment Management

- Implement an organized submission and evaluation system that eliminates manual handling of assignments.
- Ensure that students receive confirmation upon submission, reducing concerns about lost assignments.

5. Ensuring Data Security and Privacy

- Implement strong authentication and role-based access to prevent unauthorized users from accessing sensitive information.
- Use encrypted data storage and regular backups to safeguard user data against cyber threats.
- Maintain compliance with data protection regulations to ensure the privacy of students and instructors.

7. Supporting Scalability and Customization

- Design the LMS to be flexible, allowing institutions to customize features according to their specific needs.
- Ensure that the system can scale to accommodate more users as institutions grow.
- Allow integration with third-party tools such as video conferencing

3. TECHNICAL SPECIFICATION

3.1 System Overview

The Learning Management System (LMS) is a web-based platform designed to facilitate digital learning by providing a structured and efficient environment for students, instructors, and administrators. The system ensures seamless access to course materials, assignment submissions, communication tools, and progress tracking.

Key Aspects of the System:

- **User Roles:** The LMS includes multiple roles such as Students, Instructors, and Administrators, each with specific access permissions.
- **Course Management:** Instructors can create, update, and manage courses, while students can enroll, access learning materials, and submit assignments.
- **Security and Authentication:** Role-based access control and secure login mechanisms ensure data protection.
- **Scalability:** The system is designed to accommodate institutions of varying sizes, from small coaching centers to large universities.
- **User Interface:** A responsive and intuitive UI ensures easy navigation and accessibility on both desktops and mobile devices.

The LMS is developed using open-source technologies, making it cost-effective while ensuring high performance, reliability, and security.

3.2 System Architecture

The LMS follows a **three-tier architecture** consisting of the Presentation Layer, Application Layer, and Database Layer.

1. Presentation Layer (Frontend)

- Provides an interactive user interface for students, instructors, and administrators.
- Built using **HTML, CSS, JavaScript, React.js, or Bootstraps** for a responsive design.
- Ensures a seamless experience across devices, including desktops, tablets, and smartphones.

2. Application Layer (Backend)

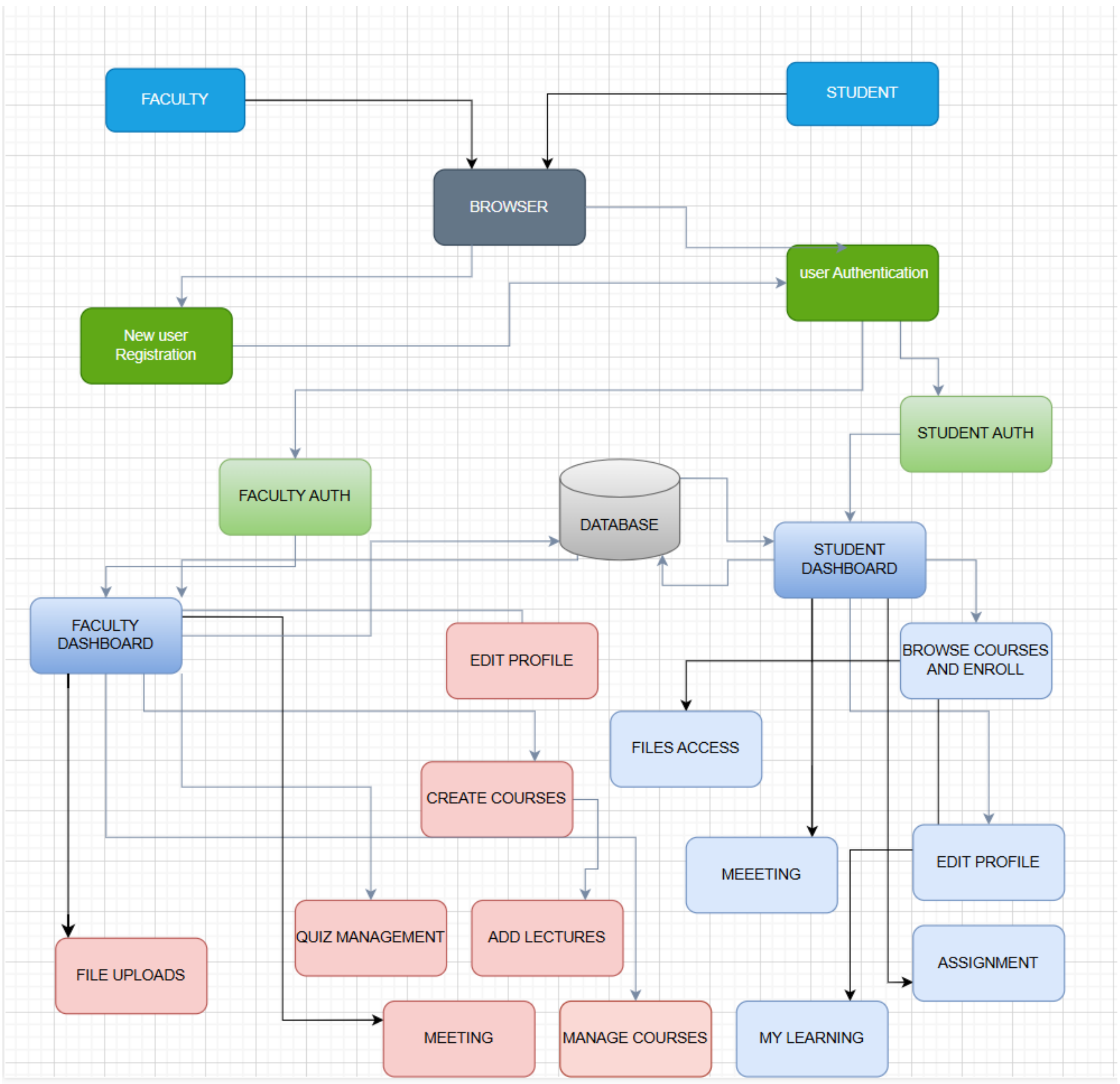
- Handles business logic and processes user requests.
- Developed using **Node.js with Express.js)** for server-side processing.
- Implements authentication, role-based access, and data management functionalities.

3. Database Layer (Storage & Management)

- Stores user data, course materials, assignments.
- Uses **MongoDB** for efficient data handling and retrieval.

System Communication Flow:

1. A user (student or instructor) logs into the LMS.
2. The frontend sends a request to the backend API for authentication.
3. Upon successful authentication, user-specific data is retrieved from the database.
4. The user can access course materials, submit assignments.
5. The system stores all interactions securely in the database and provides real-time updates.



3.3 Hardware Requirements

The infrastructure requirements for the Learning Management System (LMS) vary depending on the scale of deployment. Given the use of **MongoDB, React.js, Node.js, and Express.js**, the system relies on a well-configured backend to manage database operations, API processing, and front-end interactions efficiently.

Small-Scale Deployments

For institutions with a limited number of active users, a moderately equipped environment can ensure smooth operation. A **virtual private server (VPS) or shared hosting** can accommodate essential system functions, while **MongoDB can be hosted locally** or on **MongoDB Atlas Free Tier** for cloud accessibility.

Medium to Large-Scale Deployments

Larger institutions with higher user activity require a more advanced configuration. A **dedicated server or cloud-based hosting** on **AWS, Google Cloud, or DigitalOcean** provides improved scalability and reliability. Utilizing **MongoDB Atlas Dedicated Clusters** ensures optimized performance and automatic scaling to handle increasing data loads.

Network and Performance Optimization

Efficient data transfer and response times are critical for an uninterrupted learning experience. For smaller implementations, **moderate internet bandwidth** suffices, whereas larger deployments benefit from **high-speed connectivity** to support real-time interactions. Additionally, **load balancers like Nginx** distribute incoming requests effectively in high-traffic scenarios. Implementing **caching mechanisms such as Redis or Memcached** further enhances performance by reducing redundant database queries.

This structured approach ensures that the LMS remains responsive, scalable, and capable of handling varying levels of user engagement without compromising efficiency.

3.4 Software Requirements

The LMS is built using modern web technologies that ensure it is flexible, easy to maintain, and scalable for institutions of all sizes.

Operating System:

- The system can run on **Windows Server, Linux (Ubuntu, CentOS), or macOS** for local development.
- For deployment, **Linux-based servers (Ubuntu recommended)** are preferred due to better security and performance.

Frontend (User Interface):

- **React.js** – Used for creating a dynamic and responsive user interface.
- **HTML5, CSS3, JavaScript** – Standard web technologies for structuring and styling the UI.
- **Bootstrap** – A CSS framework to make the platform mobile-friendly and easy to navigate.

Backend (Server-Side Logic):

- **Node.js with Express.js** – Manages user requests, authentication, and database interactions.
- **JWT (JSON Web Token)** – Provides a secure way to manage user authentication and login sessions.
- **Bcrypt.js** – Ensures secure password encryption for user accounts.

Database:

- **MongoDB (NoSQL Database)** – Used for storing user data, courses, assignments, and communication logs.
- **MongoDB Atlas (Cloud Database Option)** – Ensures automatic backups and security for institutions that prefer a cloud solution.

Additional Tools & Security Measures:

- **Git & GitHub** – Version control system to track and manage code updates.
- **OAuth** – Provides a secure method for users to log in without compromising their credentials.

3.5 Functional Requirements

Functional requirements define the essential capabilities of the Learning Management System (LMS) to ensure a seamless and effective digital learning experience. These requirements focus on course management, user interaction, security, and system scalability.

1. User Authentication & Role Management

- Secure user registration and login system with email and password.
- Role-based access control for **Students, Instructors** to manage permissions.
- Profile management with the ability to update personal information.

2. Course Management

- Instructors can create, update, and delete courses.
- Students can browse and enroll in available courses.
- Support for multiple content formats, including PDFs, videos, and presentations.
- Ability to upload and manage **assignments, quizzes, and additional resources**.

3. Assignment & Submission System

- Instructors can create and schedule assignments.
- Students can submit assignments in various formats (PDF, Word, images, etc.).
- Submission deadlines with automated notifications.
- Instructors can review and provide feedback on submitted work.

5. File & Resource Management

- A centralized file storage system for instructors to upload study materials.
- Students can access and download course materials easily.
- Version control for document updates by instructors.

6 .Live Coding Terminal & Development Tools

The LMS includes an integrated live coding environment for programming courses, allowing students and instructors to write, test, and debug code directly within the platform.

- Supports multiple programming languages such as Python, JavaScript, Java.
- Provides automatic code execution, allowing users to run and test programs instantly.
- Features syntax highlighting and error detection to enhance learning efficiency.

6. Security & Data Protection

- Encrypted storage for user credentials and sensitive data.
- Secure file upload system to prevent unauthorized access.
- Two-factor authentication (2FA) for additional security layers.

4. DESIGN APPROACH AND DETAILS

4.1 Design Approach / Materials & Methods

The Learning Management System (LMS) is designed with a modern, scalable, and modular architecture, ensuring flexibility and ease of expansion. The system follows a **client-server model** where the frontend interacts with the backend via API calls, and the backend manages data storage and business logic.

Materials & Methods Used:

- **Frontend:** Built with React.js to ensure a dynamic and responsive user experience. Bootstrap is used for styling to enhance UI consistency across devices.
- **Backend:** Developed using Node.js with Express.js to handle API requests efficiently.
- **Database:** MongoDB is used for data storage, allowing for flexible and scalable data management.
- **API Communication:** RESTful APIs facilitate communication between the frontend and backend, ensuring a smooth flow of data.
- **Authentication:** Implemented using JWT (JSON Web Token) for secure user authentication.
- **Hosting & Deployment:** The system is deployed on cloud platforms like AWS or DigitalOcean, with options for scalability based on user demand.

By following this design approach, the LMS ensures high performance, security, and ease of future enhancements.

4.2 Codes and Standards

To ensure security, efficiency, and compatibility, the LMS adheres to industry standards and best practices, including:

- **Web Standards:** Uses HTML5, CSS3, and ECMAScript (JavaScript) standards for frontend development.
- **Security Standards:** Implements **OWASP security guidelines** to protect against vulnerabilities like SQL injection, XSS, and CSRF.
- **Authentication & Authorization:** Follows OAuth 2.0 and JWT authentication mechanisms to manage user access securely.
- **Data Storage & Privacy:** Complies with **GDPR (General Data Protection**

Regulation) and **ISO 27001** security standards to ensure data protection.

- **REST API Standards:** Adheres to RESTful API design principles for structured and scalable data communication.
- **Accessibility:** Ensures WCAG (Web Content Accessibility Guidelines) compliance for an inclusive user experience.

Following these standards helps in maintaining system reliability, data security, and ease of integration with other platforms.

4.3 Constraints, Alternatives, and Tradeoffs

Constraints

The development and implementation of the LMS face several constraints:

- **Scalability Limitations:** The choice of MongoDB allows flexibility, but without proper indexing and optimization, it may slow down under heavy load.
- **Network Dependency:** Since it is a web-based system, a stable internet connection is necessary for uninterrupted access.
- **Storage Requirements:** Video lectures and large learning materials require significant storage capacity, increasing hosting costs.

Alternatives Considered

Several alternatives were evaluated before finalizing the LMS technology stack:

- **SQL vs. NoSQL:** While relational databases like MySQL or PostgreSQL provide structured data management, MongoDB was chosen for its flexibility in handling dynamic data.
- **Monolithic vs. Microservices:** A monolithic approach was considered for simplicity, but a microservices architecture could offer better scalability in future iterations.
- **Local Hosting vs. Cloud Deployment:** Cloud-based hosting was chosen for scalability, but local hosting can be an alternative for institutions with on-premise infrastructure.

Tradeoffs

Some tradeoffs were made to balance performance, cost, and ease of development:

- **Performance vs. Cost:** Dedicated servers offer better performance but increase operational costs. A VPS (Virtual Private Server) was chosen as a balance between performance and affordability.
- **Security vs. User Convenience:** Implementing strong authentication (e.g., two-factor authentication) improves security but may add extra steps for users during login.

- **Customization vs. Development Time:** Highly customizable features increase complexity, so only essential features were prioritized in the initial version.

By addressing these constraints, considering alternatives, and making informed tradeoffs, the LMS is designed to be efficient, secure, and adaptable to future improvements.

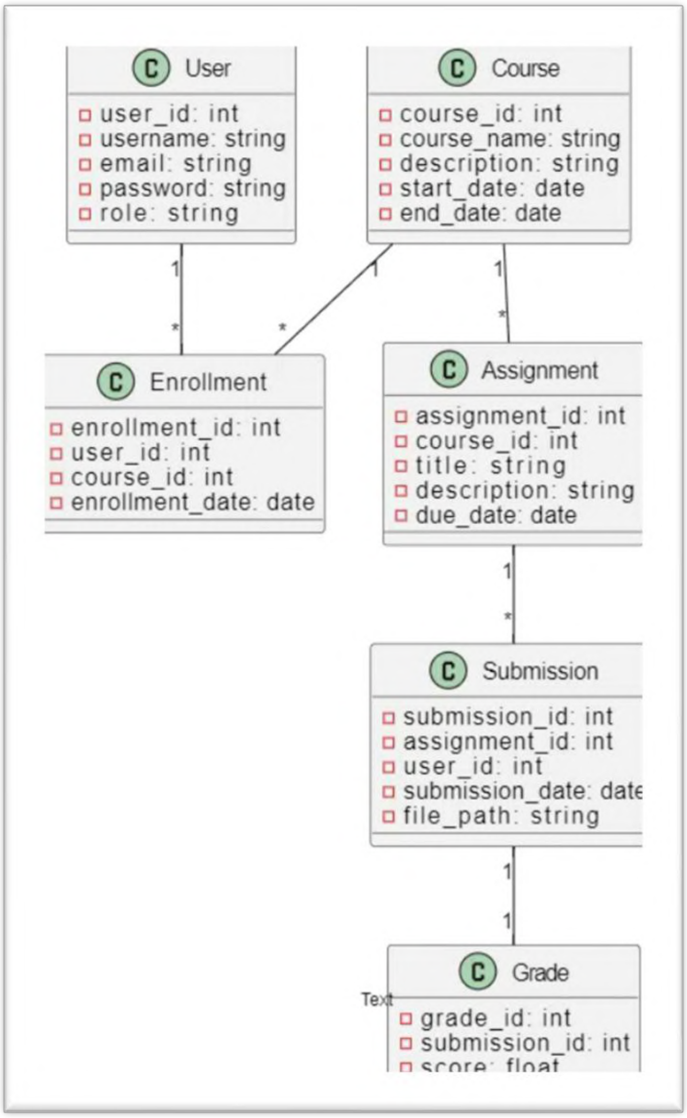
LITERATURE SURVEY:

-

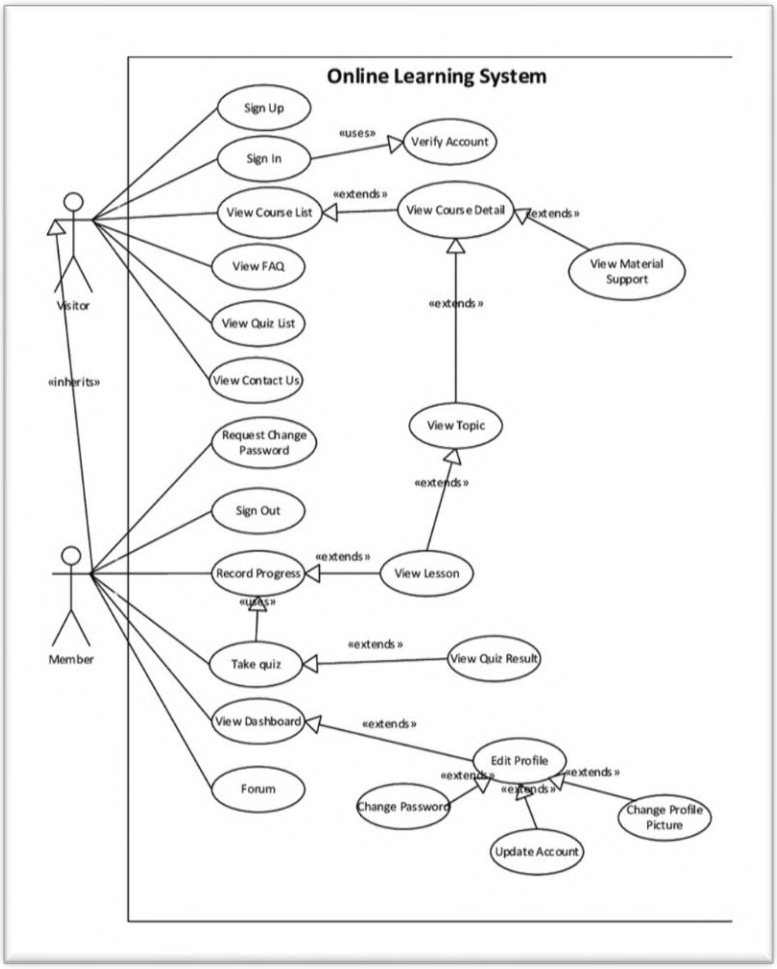
S.NO	TITLE	MERITS	DEMERITS
1	Adaptive Learning in LMS: Enhancing Personalized Education	Provides personalized learning paths, adjusts difficulty based on student performance, improves engagement	Requires high computational power, needs large datasets for accuracy.
2	Security Challenges in E-Learning Platforms: A Comprehensive Review	Identifies common vulnerabilities like data breaches, phishing, and weak authentication, suggests solutions	Lacks real-world implementation details, does not cover evolving cyber threats
3	Role-Based Access Control in LMS for Secure Data Management	Enhances security by limiting access based on user roles (teacher student), reduces unauthorized access risks	Complex implementation, requires frequent role updates and maintenance
4	Artificial Intelligence for Improving Student Engagement in LMS	AI-powered chatbots and recommendation systems help personalize learning, improves student interaction	High implementation costs, potential data privacy issues
5	Cloud-Based LMS Solutions: Scalability and Cost Benefits	Provides scalability, cost-effective compared to on-premise systems, allows remote access	Dependency on stable internet connectivity, potential data security risks
6	Gamification in LMS: Impact on Student Motivation and Learning Outcomes	Enhances motivation through rewards, leaderboards, and challenges, improves retention rates	Can distract from actual learning, may not suit all subjects
7	Mobile Learning in LMS: Accessibility and Performance Analysis	Provides learning on-the-go, supports multiple devices, increases student flexibility	Device compatibility issues, screen size limitations for detailed content
8	Data Analytics in	Provides real-time tracking of	Requires data privacy

	LMS: Enhancing Learning Insights	student progress, identifies weak areas for improvement, helps in decision-making	measures, high processing power needed
9.	Open-Source LMS Platforms: Benefits and Challenges	Cost-effective, customizable, supports integration with third-party tools	Requires technical expertise for setup and maintenance, lacks dedicated customer support
10	Integration of LMS with External Tools: Expanding Functionality	Enhances system capabilities with third-party integrations (Zoom, Google Drive, etc.), improves user experience.	Compatibility issues, may require additional costs for API access
11	Video-Based Learning in LMS: Effectiveness and Limitations	Provides interactive learning, allows self-paced study, supports visual learners	High bandwidth requirement, may not be suitable for text-heavy subjects
12	LMS for Remote Learning: Challenges and Opportunities	Supports online education effectively, allows remote collaboration, reduces geographical barriers	Lack of face-to-face interaction, potential for student isolation.
13	LMS Usability and User Experience: Improving Student Engagement	Intuitive UI/UX improves student engagement, easy navigation enhances learning experience	Requires frequent updates for feature enhancements, can become complex over time
14	Blockchain for Secure LMS: Ensuring Data Integrity and Authentication	Improves data security, prevents unauthorized tampering, ensures transparent record-keeping	Complex implementation, requires high computational resources
15	Discussion Forums in LMS: Improving Collaborative Learning	Encourages peer-to-peer learning, builds a sense of community, supports knowledge sharing	Requires active participation, can be difficult to moderat

CLASS DIAGRAM



USE CASE DIAGRAM



PROBLEM STATEMENT:

Modern education demands an advanced LMS that enhances learning, engagement, and management. Key needs include:

- Personalized Learning – Flexible, adaptive experiences for diverse learners.
- Interactive Engagement – Gamification, live quizzes, and AI-driven support.
- Seamless Integration – Compatibility with third-party tools and systems.
- Scalability & Accessibility – Support for institutions of all sizes across devices.

2. Non-Functional Requirements

These define the system's quality attributes, ensuring efficiency, security, and usability.

- **Scalability:**
  - Ability to support multiple institutions and thousands of users concurrently.
- **Security:**
  - Secure access with role-based permissions.
- **Performance & Reliability:**
  - System response time should be less than **2 seconds** for any request.
  - 99.9% uptime guarantee for uninterrupted learning.
- **Usability:**
  - Intuitive UI/UX for ease of access and navigation.
- **Compliance & Data Privacy:**
  - .
- **Technology Stack:**
- **Front-end:** React, JavaScript , CSS, HTML, JWT(for Auth)
- **Back-end:** Node.js Express
- **Database:** MongoDB
- **Deployment:** Open-source framework for scalability and cost-effectiveness.

5. SCHEDULE, TASKS, AND MILESTONES

Requirement Analysis and Planning

- **Tasks:**
  - Identify project goals, user needs, and core functionalities.
  - Research and analyze existing Learning Management Systems (LMS).
  - Finalize the technology stack (MongoDB, Node.js, Express.js, React.js, Bootstrap).
  - Define project scope, features, and database schema.
- **Milestone:** Completion of project requirement documentation and initial wireframes.

System Design and Database Setup

- **Tasks:**
  - Design the system architecture, including frontend-backend interaction.
  - Create UML diagrams and finalize API endpoints.
  - Set up MongoDB and implement basic authentication (login, registration, role-based access).
  - Develop an initial backend structure with Express.js.
- **Milestone:** Completion of system design and initial backend setup.

Backend and Frontend Development

- **Tasks:**
  - Implement course management, file upload, and live coding features in the backend.
  - Develop core frontend components using React.js and Bootstrap.
  - Connect frontend with backend APIs for smooth interaction.
  - Implement security features like authentication and authorization.
- **Milestone:** Functional prototype with backend and frontend integration.

Feature Implementation and Testing

- **Tasks:**
  - Add user role management (students, instructors, administrators).
  - Implement discussion forums and course material access.
  - Perform unit testing and integration testing for all modules.
  - Optimize system performance and fix critical bugs.
- **Milestone:** Fully functional system ready for deployment testing.

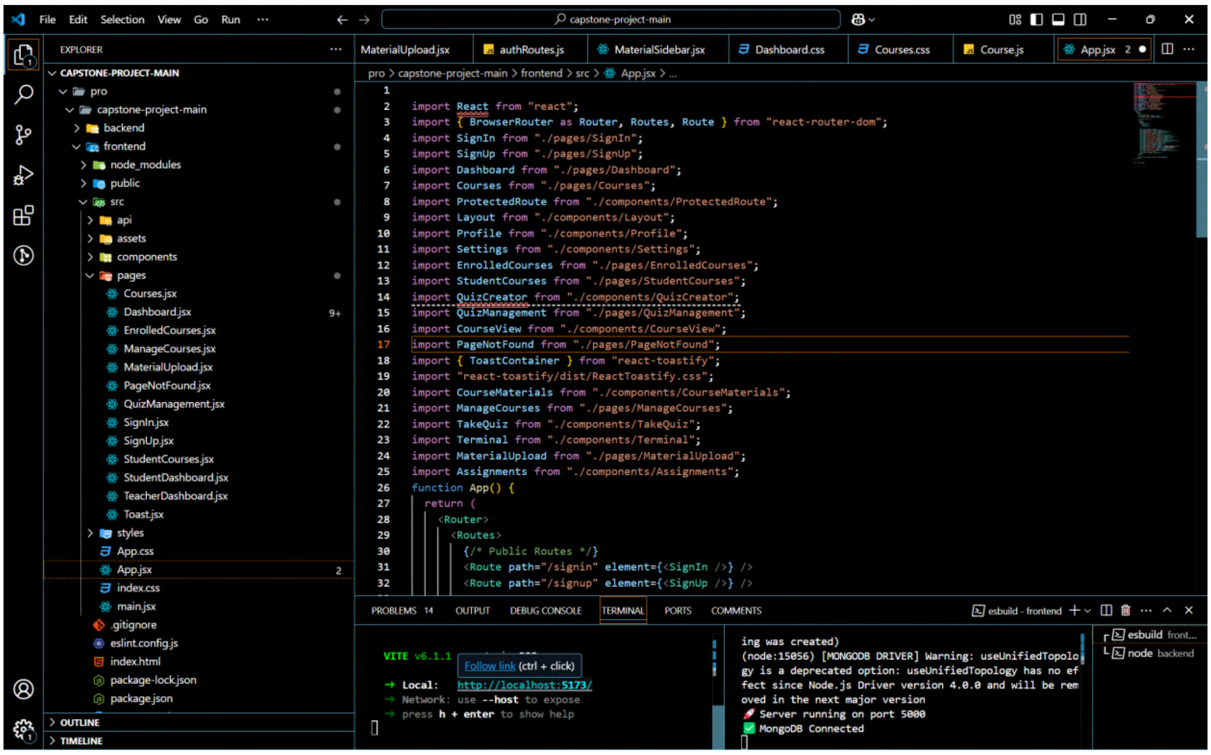
Deployment and Final Review

- **Tasks:**
  - Deploy the LMS on a cloud server (AWS, Google Cloud)
  - Conduct final testing and bug fixes.
  - Document system usage guidelines and create user manuals.
  - Collect feedback and make final improvements.
  -
- **Milestone:** Successfully deployed LMS with final documentation.

1. Project Timeline and Phases

Phase	Tasks	Duration	Deliverables
Phase 1: Requirement Analysis & Planning	Identify system requirements, user needs, and project scope. Finalize technology stack and database schema.	**2 Week**	Requirement Specification Document, Project Plan
Phase 2: System Design & Prototyping	Create UI wireframes, database schema, and system architecture. Define APIs and security protocols.	**3 Week**	System Design Document, UI Mockups, ER Diagrams
Phase 3: Development & Implementation	Develop core functionalities including authentication, course management, assessments, and communication tools.	**5 Week**	Functional MVP (Minimum Viable Product)
Phase 4: Testing & Debugging	Perform unit, integration, and security testing. Identify and resolve major bugs. Optimize system performance.	**3 Week**	Test Cases, Bug Reports, Performance Optimization
Phase 5: Deployment & Final Review	Deploy LMS on a cloud server. Conduct user testing and finalize documentation. Prepare for project submission.	**2 Week**	Live LMS Platform, Final Report, Source Code….

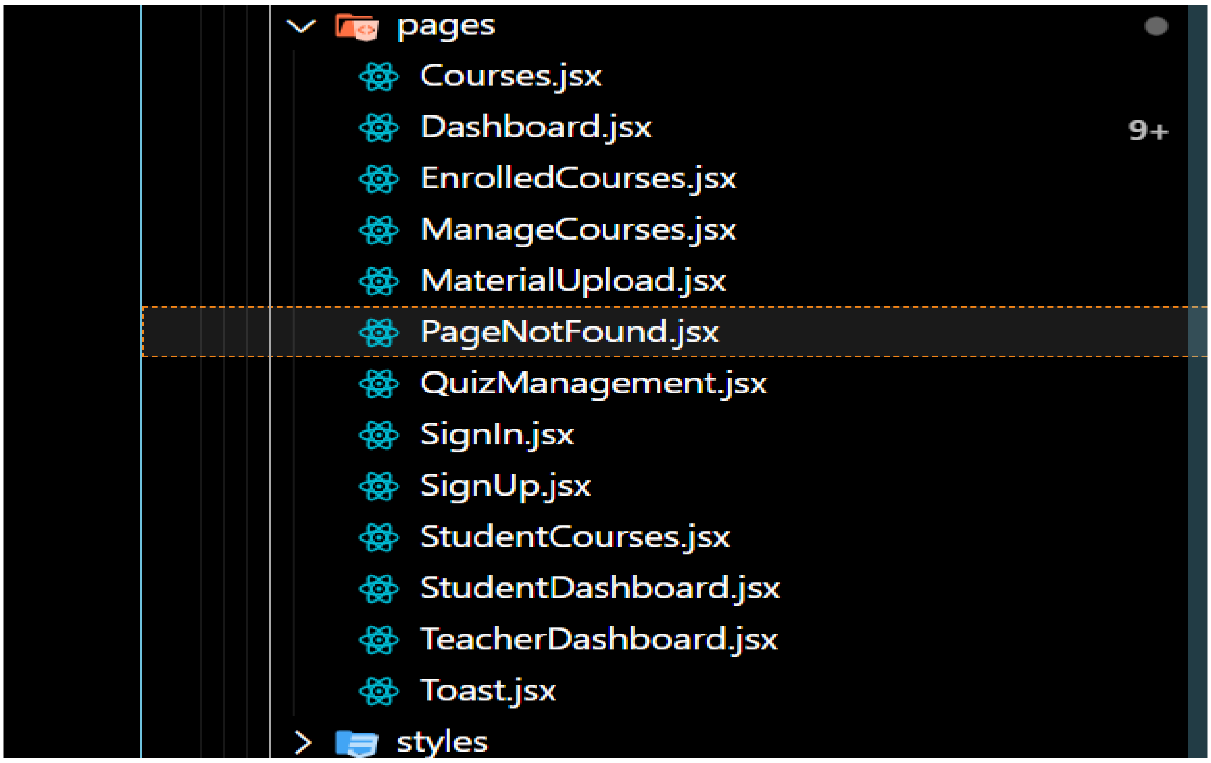
6. PROJECT DEMONSTRATION

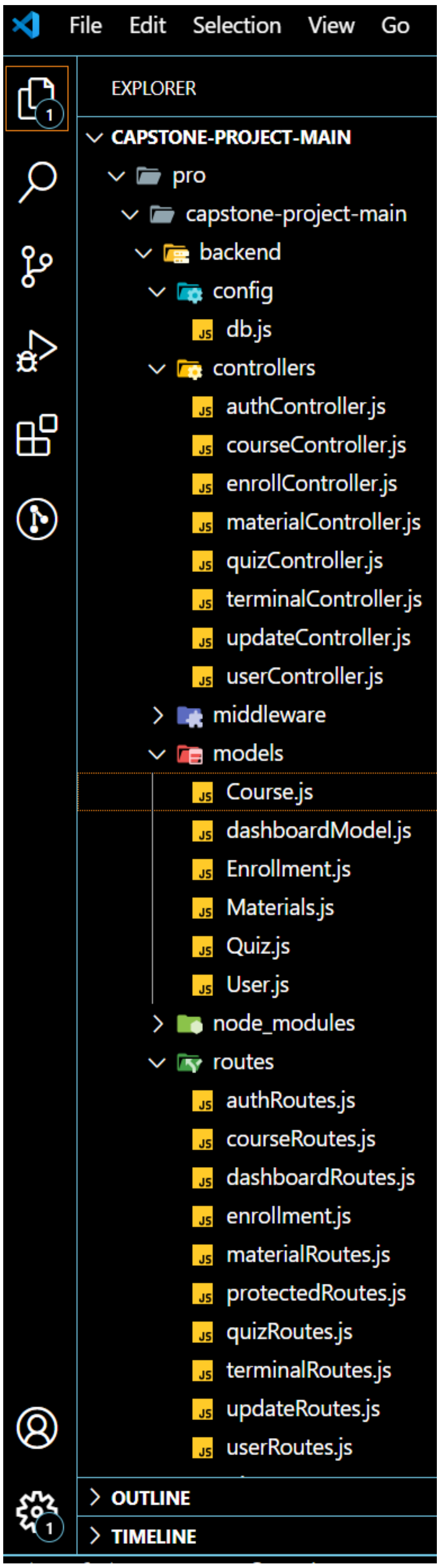


```

import React from "react";
import { BrowserRouter as Router, Routes, Route } from "react-router-dom";
import SignIn from "./pages/SignIn";
import SignUp from "./pages/SignUp";
import Dashboard from "./pages/Dashboard";
import Courses from "./pages/Courses";
import ProtectedRoute from "./components/ProtectedRoute";
import Layout from "./components/Layout";
import Profile from "./components/Profile";
import Settings from "./components/Settings";
import EnrolledCourses from "./pages/EnrolledCourses";
import StudentCourses from "./pages/StudentCourses";
import QuizCreator from "./components/QuizCreator";
import QuizManagement from "./pages/QuizManagement";
import CourseView from "./components/CourseView";
import PageNotFound from "./pages/PageNotFound";
import { ToastContainer } from "react-toastify";
import "react-toastify/dist/ReactToastify.css";
import CourseMaterials from "./components/CourseMaterials";
import ManageCourses from "./pages/ManageCourses";
import TakeQuiz from "./components/TakeQuiz";
import Terminal from "./components/Terminal";
import MaterialUpload from "./pages/MaterialUpload";
import Assignments from "./components/Assignments";
function App() {
  return (
    <Router>
      <Routes>
        {/* Public Routes */}
        <Route path="/signin" element={<SignIn />} />
        <Route path="/signup" element={<SignUp />} />
```

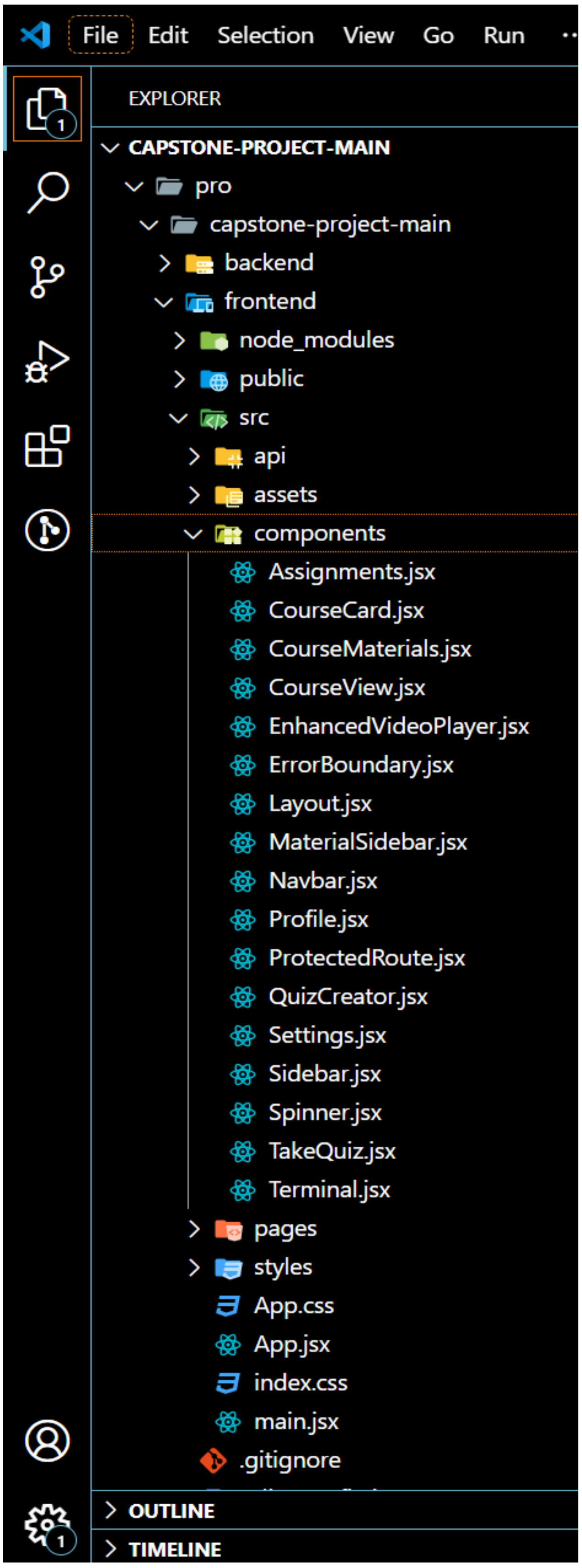
```
VITE v6.1.1

→ Local:   http://localhost:5173/
→ Network: use --host to expose
→ press h + enter to show help
```

```
ing was created)
(node:15056) [MONGODB DRIVER] Warning: useUnifiedTopolo
gy is a deprecated option: useUnifiedTopology has no ef
fect since Node.js Driver version 4.0.0 and will be rem
oved in the next major version
🚀 Server running on port 5000
✅ MongoDB Connected
```





```
File  Edit  Selection  View  Go
EXPLORER
∨ CAPSTONE-PROJECT-MAIN
  ∨ pro
    ∨ capstone-project-main
      ∨ backend
        ∨ config
            db.js
        ∨ controllers
            authController.js
            courseController.js
            enrollController.js
            materialController.js
            quizController.js
            terminalController.js
            updateController.js
            userController.js
        > middleware
        ∨ models
            Course.js
            dashboardModel.js
            Enrollment.js
            Materials.js
            Quiz.js
            User.js
        > node_modules
        ∨ routes
            authRoutes.js
            courseRoutes.js
            dashboardRoutes.js
            enrollment.js
            materialRoutes.js
            protectedRoutes.js
            quizRoutes.js
            terminalRoutes.js
            updateRoutes.js
            userRoutes.js
> OUTLINE
> TIMELINE
```



```
File  Edit  Selection  View  Go  Run  ···
EXPLORER
∨ CAPSTONE-PROJECT-MAIN
  ∨ pro
    ∨ capstone-project-main
      > backend
      ∨ frontend
        > node_modules
        > public
        ∨ src
          > api
          > assets
          ∨ components
              Assignments.jsx
              CourseCard.jsx
              CourseMaterials.jsx
              CourseView.jsx
              EnhancedVideoPlayer.jsx
              ErrorBoundary.jsx
              Layout.jsx
              MaterialSidebar.jsx
              Navbar.jsx
              Profile.jsx
              ProtectedRoute.jsx
              QuizCreator.jsx
              Settings.jsx
              Sidebar.jsx
              Spinner.jsx
              TakeQuiz.jsx
              Terminal.jsx
          > pages
          > styles
            App.css
            App.jsx
            index.css
            main.jsx
        .gitignore
> OUTLINE
> TIMELINE
```

App.jsx

```
import React from "react";
import { BrowserRouter as Router, Routes, Route } from "react-router-dom";
import SignIn from "./pages/SignIn";
import SignUp from "./pages/SignUp";
import Dashboard from "./pages/Dashboard";
import Courses from "./pages/Courses";
import ProtectedRoute from "./components/ProtectedRoute";
import Layout from "./components/Layout";
import Profile from "./components/Profile";
import Settings from "./components/Settings";
import EnrolledCourses from "./pages/EnrolledCourses";
import StudentCourses from "./pages/StudentCourses";
import QuizCreator from "./components/QuizCreator";
import QuizManagement from "./pages/QuizManagement";
import CourseView from "./components/CourseView";
import PageNotFound from "./pages/PageNotFound";
import { ToastContainer } from "react-toastify";
import "react-toastify/dist/ReactToastify.css";
import CourseMaterials from "./components/CourseMaterials";
import ManageCourses from "./pages/ManageCourses";
import TakeQuiz from "./components/TakeQuiz";
import Terminal from "./components/Terminal";
import MaterialUpload from "./pages/MaterialUpload";
import Assignments from "./components/Assignments";
function App() {
  return (
    <Router>
      <Routes>
        {/* Public Routes */}
        <Route path="/signin" element={<SignIn />} />
        <Route path="/signup" element={<SignUp />} />

        {/* Protected Routes */}
        <Route
          path="/"
          element={
            <ProtectedRoute>
              <Layout />
            </ProtectedRoute>
          }
        >
          <Route index element={<Dashboard />} />
          <Route path="dashboard" element={<Dashboard />} />
          <Route path="dashboard/profile" element={<Profile />} />
          <Route path="dashboard/settings" element={<Settings />} />
          <Route path="courses" element={<Courses />} />
          <Route path="student/courses" element={<StudentCourses />} />
          <Route path="enrolled-courses" element={<EnrolledCourses />} />
          <Route path="course/:courseId/quiz/:quizId" element={<TakeQuiz />} />
          <Route path="course/:courseId" element={<CourseView />} />
          <Route path="quiz-management" element={<QuizManagement />} />
          <Route path="/courses/:courseId/materials" element={<CourseMaterials />} />
          <Route path="/upload-materials" element={<MaterialUpload />} />
          <Route path="assignments" element={<Assignments/>} />
          <Route path="exams" element={<div>Exams</div>} />
          <Route path="terminal" element={<Terminal />} />
          <Route path="grade-assignments" element={<div>Grade Assignments</div>} />
          <Route path="student-progress" element={<div>Student Progress</div>} />
          <Route path="create-course" element={<div>Create Course</div>} />
          <Route path="manage-courses" element={<ManageCourses />} />
          <Route path="course/:courseId/materials" element={<CourseMaterials />} />
          {/* Catch all route for 404 */}
          <Route path="*" element={<PageNotFound />} />
        </Route>
      </Routes>

      <ToastContainer position="top-right" autoClose={3000} />
    </Router>
  );
}

export default App;
```

main.jsx

```
import { StrictMode } from 'react'

import { createRoot } from 'react-dom/client'

import './index.css'

import "bootstrap/dist/css/bootstrap.min.css";

import "react-toastify/dist/ReactToastify.css";

import App from './App.jsx'

createRoot(document.getElementById('root')).render(

  <StrictMode>
```

```
    <App />
  </StrictMode>,
)
```

Dashboard.jsx

```
import React, { useEffect, useState } from "react";
import axios from "axios";
import { useNavigate, useOutletContext } from "react-router-dom";
import "../styles/Dashboard.css";
import {
  FaSun,
    FaMoon,
    FaBook,
    FaClipboardList,
    FaChartBar,
    FaUser,
    FaCog,
    FaComments,
    FaSignOutAlt,
    FaChalkboardTeacher,
    FaHome,
} from "react-icons/fa";
```

```
const Dashboard = () => {
  const { user } = useOutletContext();
  const [enrolledCourses, setEnrolledCourses] = useState([]);
  const [loading, setLoading] = useState(true);
  const navigate = useNavigate();

  // Debugging - Check if user is correctly received
  useEffect(() => {
    console.log("User data received:", user);

    if (user?.role === "student") {
      fetchEnrolledCourses();
    }
  }, [user])
  const fetchEnrolledCourses = async () => {
    try {
      const response = await axios.get("/api/courses/enrolled", {
        headers: { Authorization: `Bearer ${localStorage.getItem("token")}` },
      });
      console.log("Enrolled Courses:", response.data); // Debugging API Response
      setEnrolledCourses(response.data || []);
    } catch (error) {
      console.error("Error fetching enrolled courses:", error);
    } finally {
```

```
    setLoading(false);
  }
};

// Ensure user data is loaded before rendering
if (!user || !user.role) {
  return <p>Loading user data...</p>;
}

return (
  <div className="dashboard-container">
    <main className="dashboard-content">
      <h1>Welcome, {user.name}!</h1>

      {/* Student Dashboard Sections */}
      {user.role === "student" && (
        <>
          <section className="dashboard-section">
            <h2>📚 Course Progress</h2>
            <p>Ongoing: Web Development</p>
            <p>Completed: Java Basics</p>
          </section>
          <section className="dashboard-section">
            <h2>📌 Assignments & Exams</h2>
```

```
 <p>Upcoming Assignment: ReactJS Project (Due: 28th March)</p>
 <p>Upcoming Exam: Data Structures (Exam Date: 7th May)</p>
</section>
<section className="dashboard-section">
 <h2>🎥 Google Meet Links</h2>
 {loading ? (
  <p>Loading your courses...</p>
 ) : enrolledCourses.length > 0 ? (
  enrolledCourses.map((course) => (
   <div key={course._id} className="course-meet-link">
    <p>
     <strong>{course.title}:</strong>{" "}
     {course.googleMeetLink ? (
      <a
       href={course.googleMeetLink}
       target="_blank"
       rel="noopener noreferrer"
       className="google-meet-link"
      >
       Join Google Meet
      </a>
     ) : (
      "No Google Meet link available"
     )}
```

```
                </p>
              </div>
            ))
          ) : (
            <p>You are not enrolled in any courses yet.</p>
          )}
        </section>
      </>
    )}
{(user.role?.toLowerCase() === "teacher" || user.role?.toLowerCase() === "instructor")
&& (
  <>
    <section className="dashboard-section">
      <h2>📖 Manage Courses</h2>
      <p>You have 3 courses currently active.</p>
      <button onClick={() => navigate("/manage-courses")}>
        Go to Course Management
      </button>
    </section>
    <section className="dashboard-section">
      <h2>✅ Grade Assignments</h2>
      <p>5 assignments need grading.</p>
      <button onClick={() => navigate("/grade-assignments")}>
        Go to Grading
      </button>
    </section>
  </>
)}
        {/* Debugging: Show user role to confirm it's detected correctly */}
        <p style={{ marginTop: "20px", color: "gray" }}>
          <strong>Debug:</strong> Role Detected - {user.role}
        </p>
      </main>
    </div>
  );
};
export default Dashboard;
```

Navbar.jsx

```
import { useNavigate, Link } from "react-router-dom";
import { FaUser, FaCog, FaSignOutAlt, FaBell } from "react-icons/fa";

export default function Navbar({ user }) {
  const navigate = useNavigate();

  const handleLogout = () => {
    localStorage.removeItem("token");
    localStorage.removeItem("role");
    navigate("/");
```

```
 };

 return (
   <nav className="bg-blue-600 text-white p-4 flex justify-between items-center">
     <div className="flex items-center gap-4">
       <h1 className="text-xl font-semibold">LMS Dashboard</h1>
     </div>
     <div className="flex items-center gap-6">
       {/* Notifications */}
       <div className="relative cursor-pointer">
         <FaBell className="text-xl hover:text-blue-200" />
         <span className="absolute -top-2 -right-2 bg-red-500 text-xs rounded-full w-4 h-
4 flex items-center justify-center">
           2
         </span>
       </div>

       {/* User Profile Dropdown */}
       <div className="relative group">
         <div className="flex items-center gap-3 cursor-pointer">
           <div className="w-10 h-10 rounded-full bg-blue-700 flex items-center justify-
center">
             <FaUser className="text-xl" />
           </div>
           <div className="hidden md:block">
```

```
            <p className="text-sm font-medium">{user?.name}</p>

            <p className="text-xs opacity-75">{user?.email}</p>

          </div>

        </div>



        {/* Dropdown Menu */}

         <div className="absolute right-0 mt-2 w-48 bg-white rounded-lg shadow-lg py-2
invisible group-hover:visible z-50">

          <Link

            to="/dashboard/profile"

            className="flex items-center gap-2 px-4 py-2 text-gray-700 hover:bg-blue-50"

          >

            <FaUser className="text-blue-600" />

            <span>My Profile</span>

          </Link>

          <Link

            to="/dashboard/settings"

            className="flex items-center gap-2 px-4 py-2 text-gray-700 hover:bg-blue-50"

          >

            <FaCog className="text-blue-600" />

            <span>Settings</span>

          </Link>

          <hr className="my-2" />

          <button
```

```
            onClick={handleLogout}
             className="flex items-center gap-2 px-4 py-2 text-red-600 hover:bg-red-50 w-
full"
          >
            <FaSignOutAlt />
            <span>Logout</span>
          </button>
        </div>
      </div>
    </div>
  </nav>
 );
}
```

Sidebar.jsx

```
import { NavLink } from "react-router-dom";
import { FaTerminal } from "react-icons/fa";
import "../styles/Dashboard.css";
import {
   FaUser,FaHome,FaBook,FaChartBar,FaClipboardListFaChalkboardTeacher,FaCom
ments,FaCog,FaSignOutAlt,FaGraduationCap,FaSearch,FaUserGraduate,FaQuestion
Circle,FaUserCircle, FaUpload,
} from "react-icons/fa";


export default function Sidebar({ user, handleLogout }) {
  return (
<aside className="sidebar">
 <div style={{ display: "flex", alignItems: "center", gap: "10px" }}>
  <img src="/icon.png" alt="Learnit Logo" style={{ width: "40px" }} />
  <h2 style={{ marginTop: "15px" }}> LMS Dashboard</h2>
 </div>

 <div className="user-info">
  <FaUser className="user-icon" />
  <div>
   <p>{user?.name}</p>
   <span>{user?.role === "instructor" ? "Instructor" : "Student"}</span>
  </div>
 </div>

 <nav>
  <ul>
   <li>
    <NavLink
     to="/dashboard"
     className={({ isActive }) => (isActive ? "active" : "")}
    >
     <FaHome /> Dashboard
    </NavLink>
   </li>
```

```
{/* Student Navigation */}
{user?.role === "student" && (
  <>
    <li>
      <NavLink
        to="/courses"
        className={({ isActive }) => (isActive ? "active" : "")}
      >
        <FaSearch /> Browse Courses
      </NavLink>
    </li>
    <li>
      <NavLink
        to="/enrolled-courses"
        className={({ isActive }) => (isActive ? "active" : "")}
      >
        <FaGraduationCap /> My Learning
      </NavLink>
    </li>
    <li>
      <NavLink
        to="/terminal"
        className={({ isActive }) => (isActive ? "active" : "")}
  >
    <FaTerminal /> Code Terminal
  </NavLink>
</li>
<li>
  {/* <NavLink
    to="/my-progress"
    className={({ isActive }) => (isActive ? "active" : "")}
  >
    <FaChartBar /> My Progress
  </NavLink> */}
</li>
<li>
  <NavLink
    to="/assignments"
    className={({ isActive }) => (isActive ? "active" : "")}
  >
    <FaClipboardList /> Assignments
  </NavLink>
</li>
<li>
  {/* <NavLink
    to="/exams"
    className={({ isActive }) => (isActive ? "active" : "")}
```

```
      >
        <FaUserGraduate /> Exams
      </NavLink> */}
    </li>
  </>
)}

{/* Instructor Navigation */}
{user?.role === "instructor" && (
  <>
    <li>
      <NavLink
        to="/courses"
        className={({ isActive }) => (isActive ? "active" : "")}
      >
        <FaBook /> Create Course
      </NavLink>
    </li>
    <li>
      <NavLink
        to="/manage-courses"
        className={({ isActive }) => (isActive ? "active" : "")}
      >
        <FaChalkboardTeacher /> Manage Courses
  </NavLink>
</li>
<li>
  {/* <NavLink
    to="/grade-assignments"
    className={({ isActive }) => (isActive ? "active" : "")}
  >
    <FaClipboardList /> Grade Work
  </NavLink> */}
</li>
<li>
  {/* <NavLink
    to="/student-progress"
    className={({ isActive }) => (isActive ? "active" : "")}
  >
    <FaChartBar /> Student Progress
  </NavLink> */}
</li>
<li>
  <NavLink
    to="/quiz-management"
    className={({ isActive }) => (isActive ? "active" : "")}
  >
    <FaQuestionCircle /> Quiz Management
```

```
      </NavLink>
    </li>
    <li>
      <NavLink
        to="/upload-materials"
        className={({ isActive }) => (isActive ? "active" : "")}
      >
        <FaUpload /> Upload Materials
      </NavLink>
    </li>
  </>
)}

{/* Common Navigation */}
<li className="divider"></li>
<li>
  {/* <NavLink
    to="/discussions"
    className={({ isActive }) => (isActive ? "active" : "")}
  >
    <FaComments /> Discussions
  </NavLink> */}
</li>
<li>
```

```
          <NavLink
            to="/dashboard/profile"
            className={({ isActive }) => (isActive ? "active" : "")}
          >
            <FaUserCircle /> My Profile
          </NavLink>
        </li>
        <li>
          <NavLink
            to="/dashboard/settings"
            className={({ isActive }) => (isActive ? "active" : "")}
          >
            <FaCog /> Settings
          </NavLink>
        </li>
        <li className="logout" onClick={handleLogout}>
          <FaSignOutAlt /> Logout
        </li>
        <li className="divider"></li>
      </ul>
    </nav>
  </aside>
 );
}
```

SignIn.jsx

```
import React, { useState } from "react";
import axios from "axios";
import { useNavigate } from "react-router-dom";
import "bootstrap/dist/css/bootstrap.min.css";
const SignIn = () => {
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [message, setMessage] = useState({ text: "", type: "" });
  const navigate = useNavigate();

  const handleLogin = async (e) => {
    e.preventDefault();
    try {
      const res = await axios.post("http://localhost:5000/api/auth/login", {
        email,
        password,
      });
      localStorage.setItem("token", res.data.token);
      setMessage({ text: "Login Successful! Redirecting...", type: "success" });
      setTimeout(() => {
        navigate("/dashboard");
      }, 1500);
  } catch (err) {
    setMessage({
      text: err.response?.data?.message || "Login failed",
      type: "danger",
    });
  }
};

return (
  <div className="d-flex justify-content-center align-items-center vh-100">
    <div
      className="card p-4 shadow-lg"
      style={{ width: "400px", background: "#fff", borderRadius: "15px" }}
    >
      <div className="text-center">
        <img src="./icon.png" alt="LearnIt Logo" style={{ width: "60px" }} />
      </div>
      <h3 className="text-center text-dark mt-3">Login to Learnit.com</h3>
      {message.text && (
        <div className={`alert alert-${message.type} mt-2`}>
          {message.text}
        </div>
      )}
      <form onSubmit={handleLogin} className="mt-3">
        <div className="mb-3">
          <label className="fw-bold mb-1">Email</label>
          <input
            type="email"
            className="form-control"
            value={email}
            onChange={(e) => setEmail(e.target.value)}
            required
          />
        </div>
        <div className="mb-3">
          <label className="fw-bold mb-1">Password</label>
          <input
            type="password"
            className="form-control"
            value={password}
            onChange={(e) => setPassword(e.target.value)}
            required
          />
        </div>
        <button type="submit" className="btn btn-primary w-100 fw-bold">
          Login
        </button>
      </form>
```

```
        <p className="mt-3 text-center">
          Don't have an account?{" "}
          <a href="/signup" className="fw-bold">
            Sign up
          </a>
        </p>
      </div>
    </div>
  );
};
export default SignIn;
```

Signup.jsx

```
import React, { useState } from "react";
import axios from "axios";
import { useNavigate } from "react-router-dom";
import "bootstrap/dist/css/bootstrap.min.css";

const SignUp = () => {
  const [name, setName] = useState("");
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [role, setRole] = useState("student");
  const [message, setMessage] = useState({ text: "", type: "" });
const navigate = useNavigate();

const handleSignup = async (e) => {
  e.preventDefault();
  try {
    await axios.post("http://localhost:5000/api/auth/register", {
      name,
      email,
      password,
      role,
    });
    setMessage({
      text: "Signup Successful! Redirecting to login...",
      type: "success",
    });

    setTimeout(() => {
      navigate("/");
    }, 1500);
  } catch (err) {
    setMessage({
      text: err.response?.data?.message || "Signup failed",
      type: "danger",
    });
  }
};

return (
  <div className="d-flex justify-content-center align-items-center vh-100">
    <div
      className="card p-4 shadow-lg text-center"
      style={{
        width: "400px",
        background: "#fff",
        borderRadius: "12px",
        padding: "30px",
      }}
    >
      <img
        src="/icon.png" // Change this to your logo path
        alt="Learnit Logo"
        style={{ width: "60px", margin: "0 auto 15px" }}
      />
      <h3 className="text-primary mb-3">Create Your Account</h3>
      {message.text && (
        <div className={`alert alert-${message.type} mt-2`}>
          {message.text}
        </div>
)}
<form onSubmit={handleSignup}>
  <div className="mb-3 text-start">
    <label className="fw-bold mb-1">Name</label>
    <input
      type="text"
      className="form-control"
      value={name}
      onChange={(e) => setName(e.target.value)}
      required
    />
  </div>
  <div className="mb-3 text-start">
    <label className="fw-bold mb-1">Email</label>
    <input
      type="email"
      className="form-control"
      value={email}
      onChange={(e) => setEmail(e.target.value)}
      required
    />
  </div>
  <div className="mb-3 text-start">
    <label className="fw-bold mb-1">Password</label>
    <input
      type="password"
      className="form-control"
      value={password}
      onChange={(e) => setPassword(e.target.value)}
      required
    />
  </div>
  <div className="mb-3 text-start">
    <label className="fw-bold mb-1">Role</label>
    <select
      className="form-control"
      value={role}
      onChange={(e) => setRole(e.target.value)}
    >
      <option value="student">Student</option>
      <option value="instructor">Instructor</option>
    </select>
  </div>
  <button type="submit" className="btn btn-primary w-100">
    Sign Up
  </button>
</form>
<p className="mt-3">
```

```
          Already have an account? <a href="/">Login</a>
        </p>
      </div>
    </div>
  );
};

export default SignUp;
```

Backends:

Server.js

```
const express = require("express");
const dotenv = require("dotenv");
const cors = require("cors");
const mongoose = require("mongoose");
const path = require("path");
const fs = require("fs");
dotenv.config();

const app = express();
const uploadDir = path.join(__dirname, 'uploads', 'course-materials');
if (!fs.existsSync(uploadDir)) {
    fs.mkdirSync(uploadDir, { recursive: true });
}
app.use(cors({
  origin: ['http://localhost:5173', 'http://127.0.0.1:5173'],
  credentials: true,
  methods: ['GET', 'POST', 'PUT', 'DELETE', 'OPTIONS'],
  allowedHeaders: ['Content-Type', 'Authorization']
}));
app.use(express.json());
app.use(express.urlencoded({ extended: true }));
app.use('/uploads', express.static(path.join(__dirname, 'uploads')));
mongoose
  .connect(process.env.MONGO_URI, {
    useNewUrlParser: true,
    useUnifiedTopology: true,
  })
  .then(() => console.log("✅ MongoDB Connected"))
  .catch((err) => {
    console.error("❌ MongoDB Connection Error:", err);
    process.exit(1);
  });
const authRoutes = require("./routes/authRoutes");
const courseRoutes = require("./routes/courseRoutes");
const quizRoutes = require("./routes/quizRoutes");
const userRoutes = require("./routes/userRoutes");
const materialRoutes = require('./routes/materialRoutes');
const terminalRoutes = require('./routes/terminalRoutes');
app.use("/api/auth", authRoutes);
app.use("/api/courses", courseRoutes);
app.use("/api/", quizRoutes); // Fixed quiz routes path
app.use("/api/users", userRoutes);
app.use("/api", materialRoutes);
app.use('/uploads', express.static('uploads'));
app.use('/api', require('./routes/materialRoutes'));
app.use('/api/terminal', terminalRoutes);
app.use((err, req, res, next) => {
  console.error('Error:', err);
  res.status(err.status || 500).json({
    success: false,
    message: err.message || "Internal server error",
    error: process.env.NODE_ENV === 'development' ? err : {}
  });
});

app.use('*', (req, res) => {
  res.status(404).json({
    success: false,
    message: `Route ${req.originalUrl} not found`
  });
});


const PORT = process.env.PORT || 5000;
app.listen(PORT, () => console.log(`🚀 Server running on port ${PORT}`));
```

config

db.js

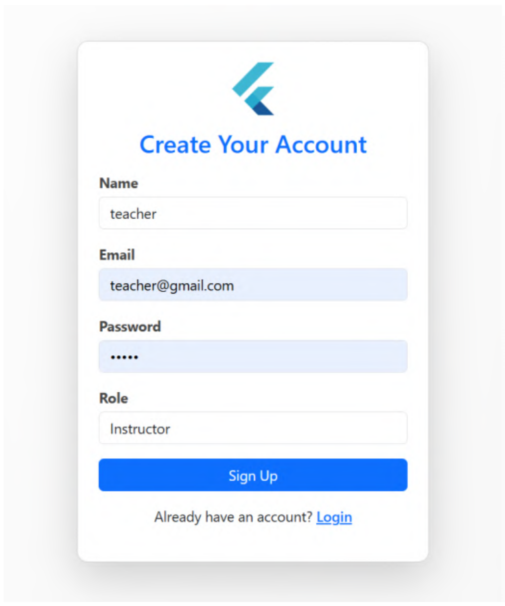
```
const mongoose = require('mongoose');
const connectDB = async () => {
  try {
    const conn = await mongoose.connect(process.env.MONGO_URI, {
      useNewUrlParser: true,
      useUnifiedTopology: true,
      serverSelectionTimeoutMS: 5000,
      socketTimeoutMS: 45000,
    });


    console.log(`✅ MongoDB Connected: ${conn.connection.host}`);


    mongoose.connection.on('error', err => {
      console.error('❌ MongoDB connection error:', err);
    });
    mongoose.connection.on('disconnected', () => {
      console.warn('⚠️ MongoDB disconnected');
    });
    mongoose.connection.on('reconnected', () => {
      console.log('✅ MongoDB reconnected');
    })
  } catch (error) {
    console.error('❌ MongoDB Connection Error:', error.message);
    process.exit(1);
  }
};
module.exports = connectDB;
```

Index.js

```
const express = require("express");
const mongoose = require("mongoose");
const cors = require("cors");
require("dotenv").config();
const authRoutes = require("./routes/authRoutes"); // Ensure correct path
const app = express();
// Middleware
app.use(express.json());
app.use(cors());

// Routes
```

```
app.use("/api/auth", authRoutes); // Prefix API endpoints with /api/auth


// Connect to MongoDB
mongoose
  .connect(process.env.MONGO_URI) // ✅ Removed deprecated options
  .then(() => console.log("✅ MongoDB Connected"))
  .catch((err) => {
    console.error("❌ MongoDB Connection Error:", err.message);
    process.exit(1); // Exit process if DB connection fails
  });


const PORT = process.env.PORT || 5000;
app.listen(PORT, () => console.log(`🚀 Server running on port ${PORT}`));
```

Sample output :



Create Your Account

Name: teacher

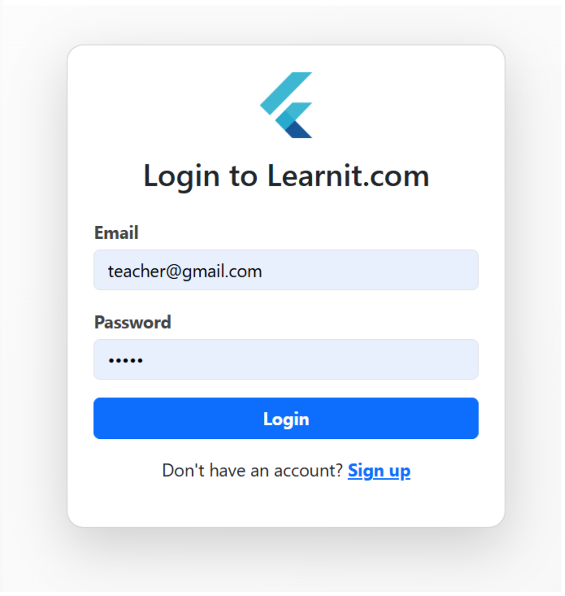
Email: teacher@gmail.com

Password: •••••

Role: Instructor

Sign Up

Already have an account? Login



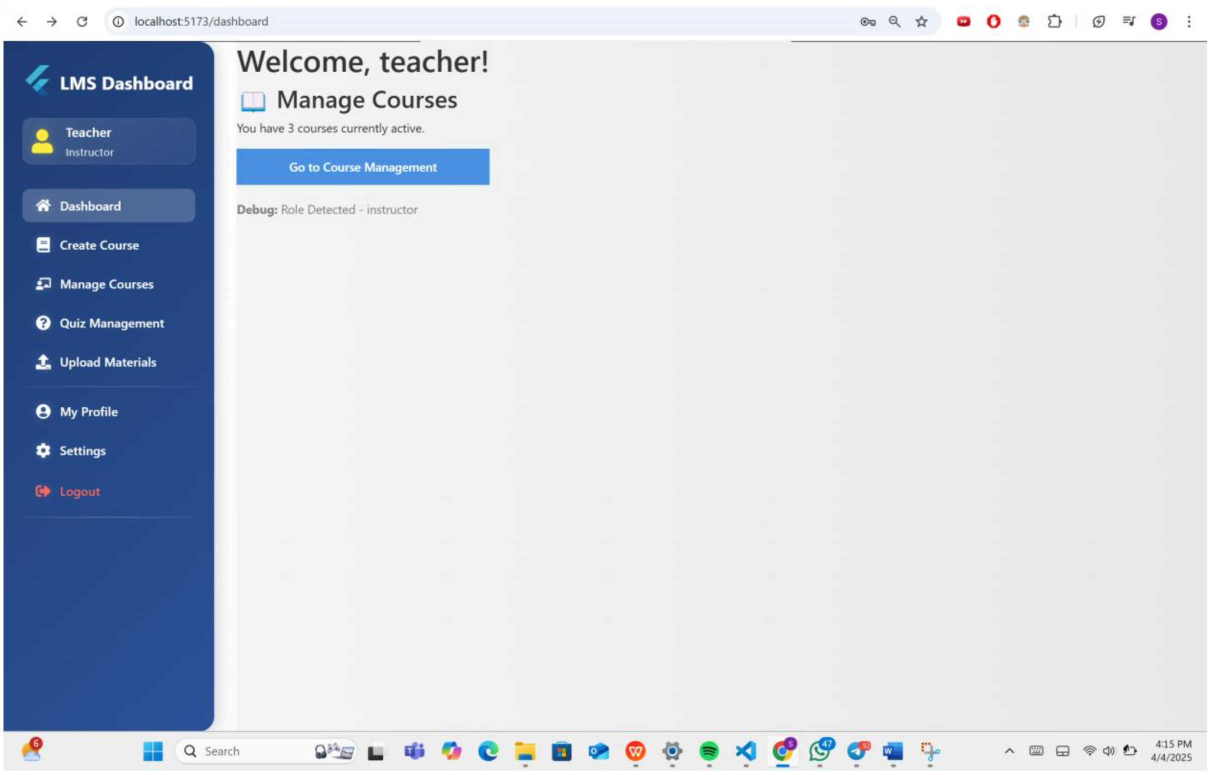
Login to Learnit.com

Email: teacher@gmail.com

Password: •••••

Login

Don't have an account? Sign up



localhost:5173/dashboard

LMS Dashboard

Teacher
Instructor

- Dashboard
- Create Course
- Manage Courses
- Quiz Management
- Upload Materials
- My Profile
- Settings
- Logout

Welcome, teacher!

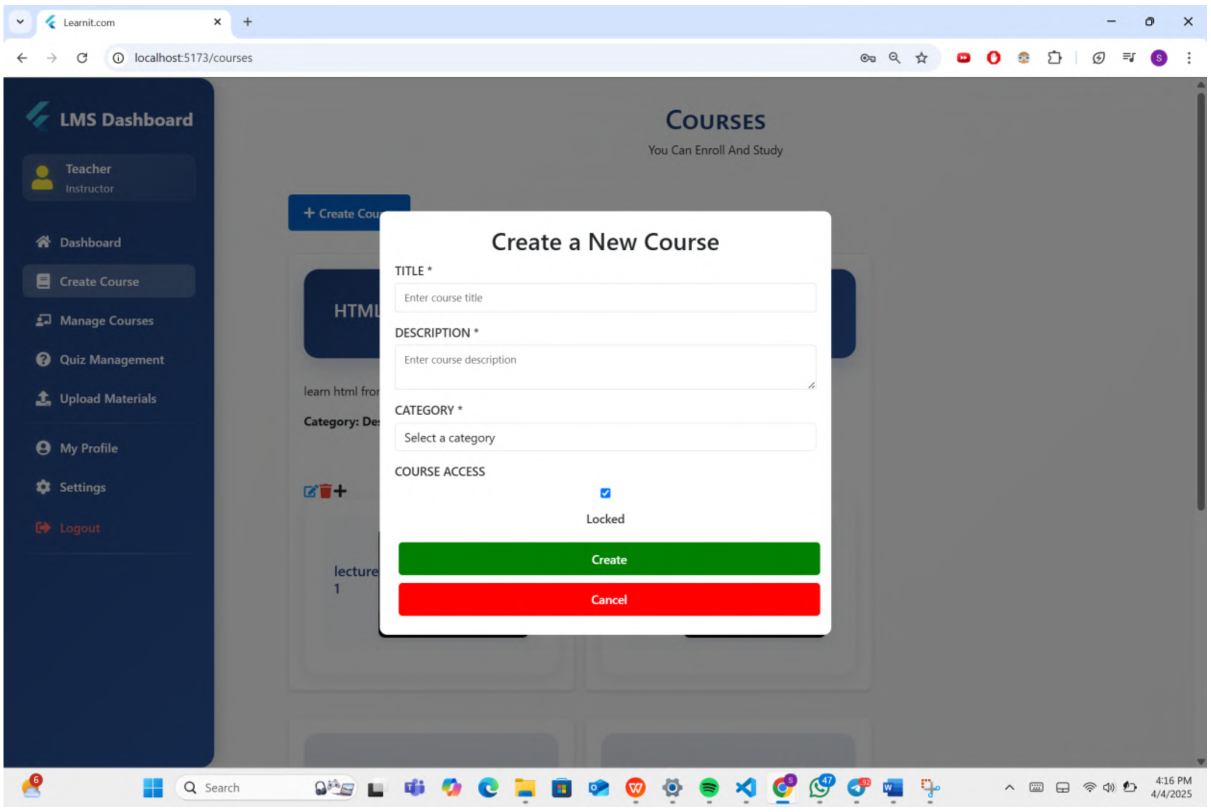
📖 Manage Courses

You have 3 courses currently active.

Go to Course Management

Debug: Role Detected - instructor

4:15 PM
4/4/2025


Learnit.com

localhost:5173/courses

LMS Dashboard

Teacher
Instructor

- Dashboard
- Create Course
- Manage Courses
- Quiz Management
- Upload Materials
- My Profile
- Settings
- Logout

COURSES

You Can Enroll And Study

Create a New Course

TITLE *

Enter course title

DESCRIPTION *

Enter course description

CATEGORY *

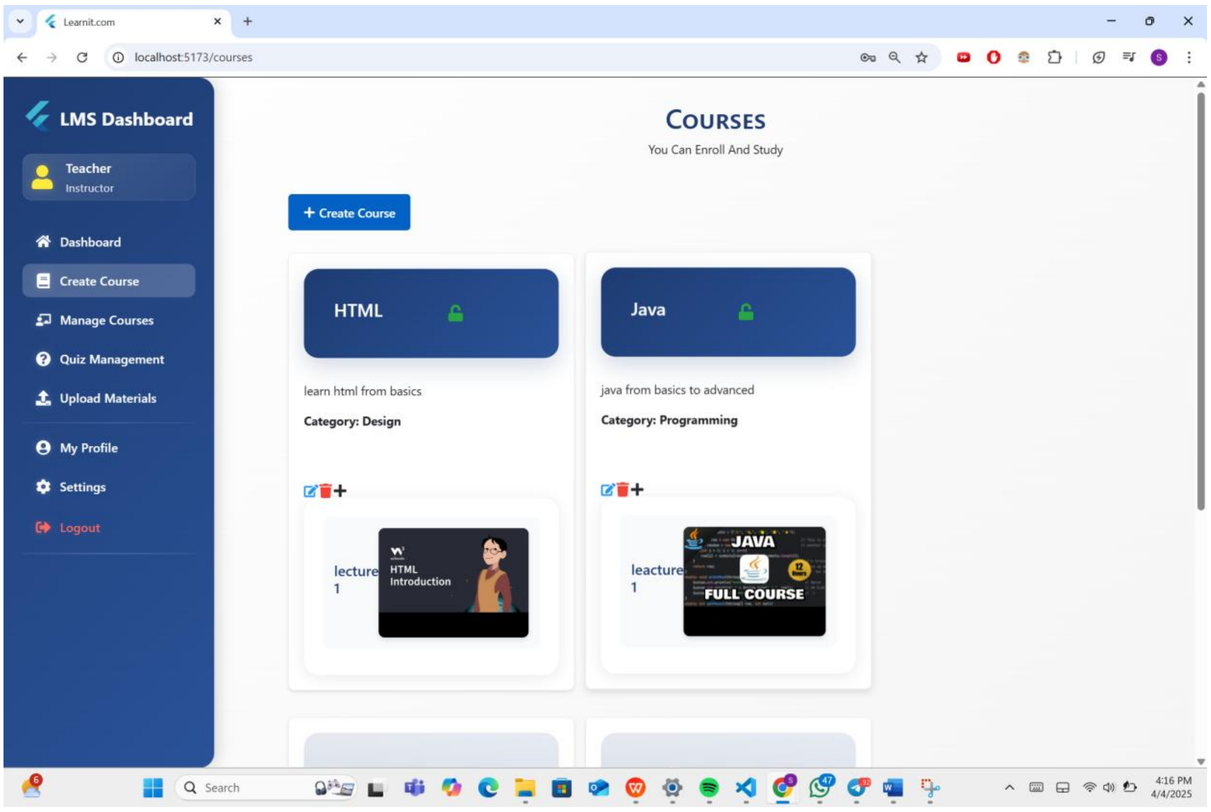
Select a category

COURSE ACCESS

Locked

Create

Cancel


Learnit.com

localhost:5173/courses

LMS Dashboard

Teacher
Instructor

- Dashboard
- Create Course
- Manage Courses
- Quiz Management
- Upload Materials
- My Profile
- Settings
- Logout

COURSES

You Can Enroll And Study

+ Create Course

HTML

learn html from basics

Category: Design

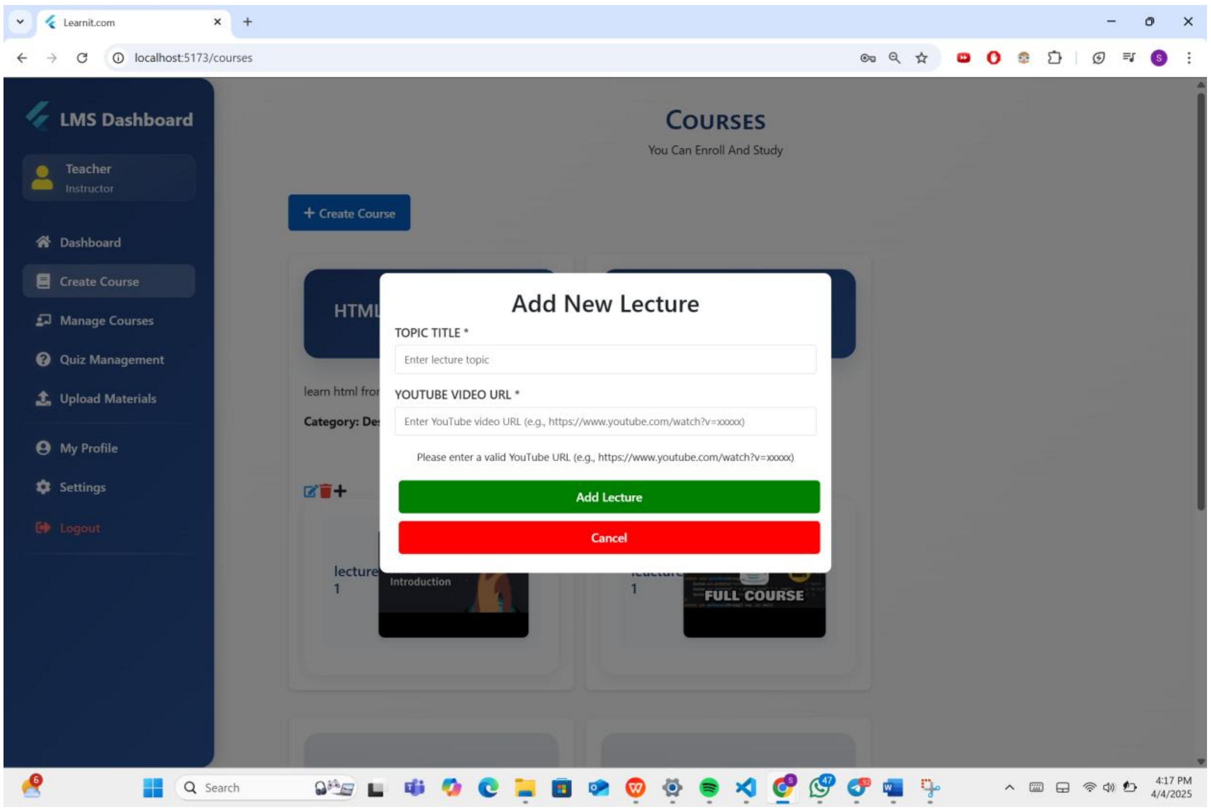
lecture 1

Java

java from basics to advanced

Category: Programming

leacture 1



Add New Lecture

TOPIC TITLE *

Enter lecture topic

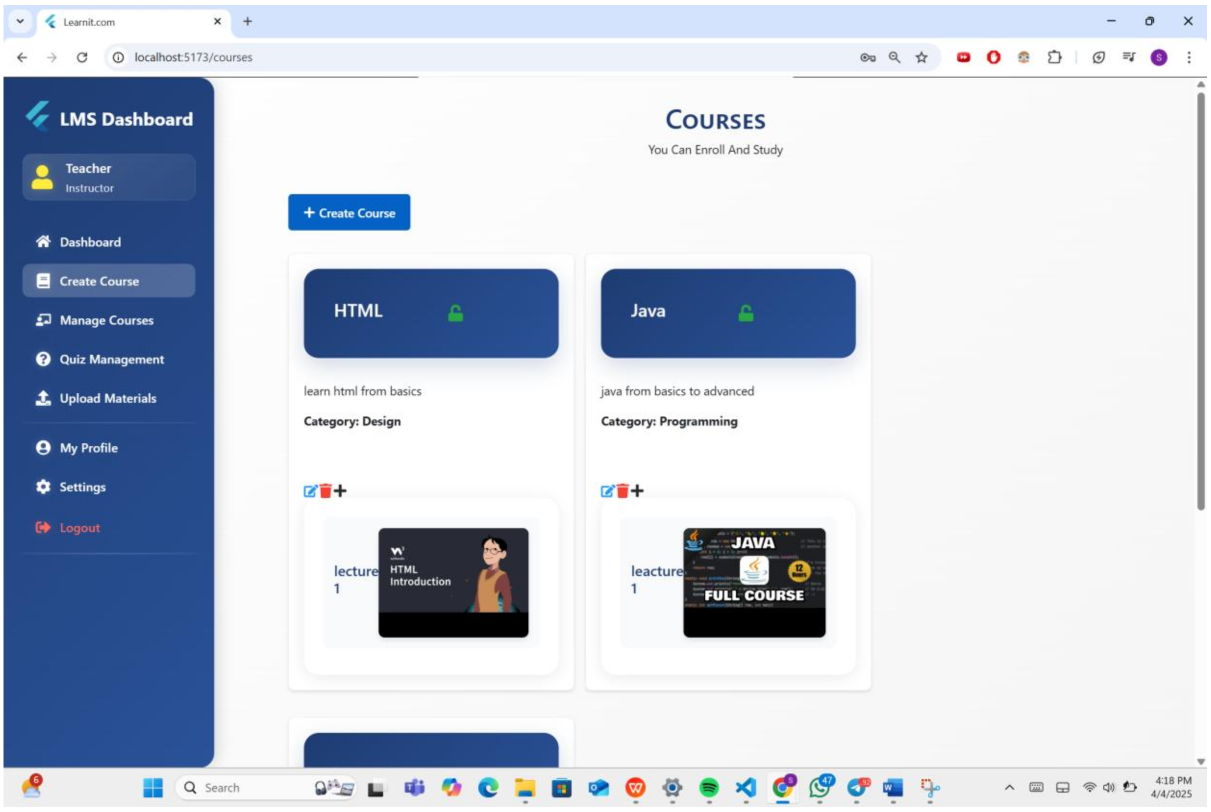
YOUTUBE VIDEO URL *

Enter YouTube video URL (e.g., https://www.youtube.com/watch?v=xxxxx)

Please enter a valid YouTube URL (e.g., https://www.youtube.com/watch?v=xxxxx)

Add Lecture

Cancel



HTML

learn html from basics

Category: Design

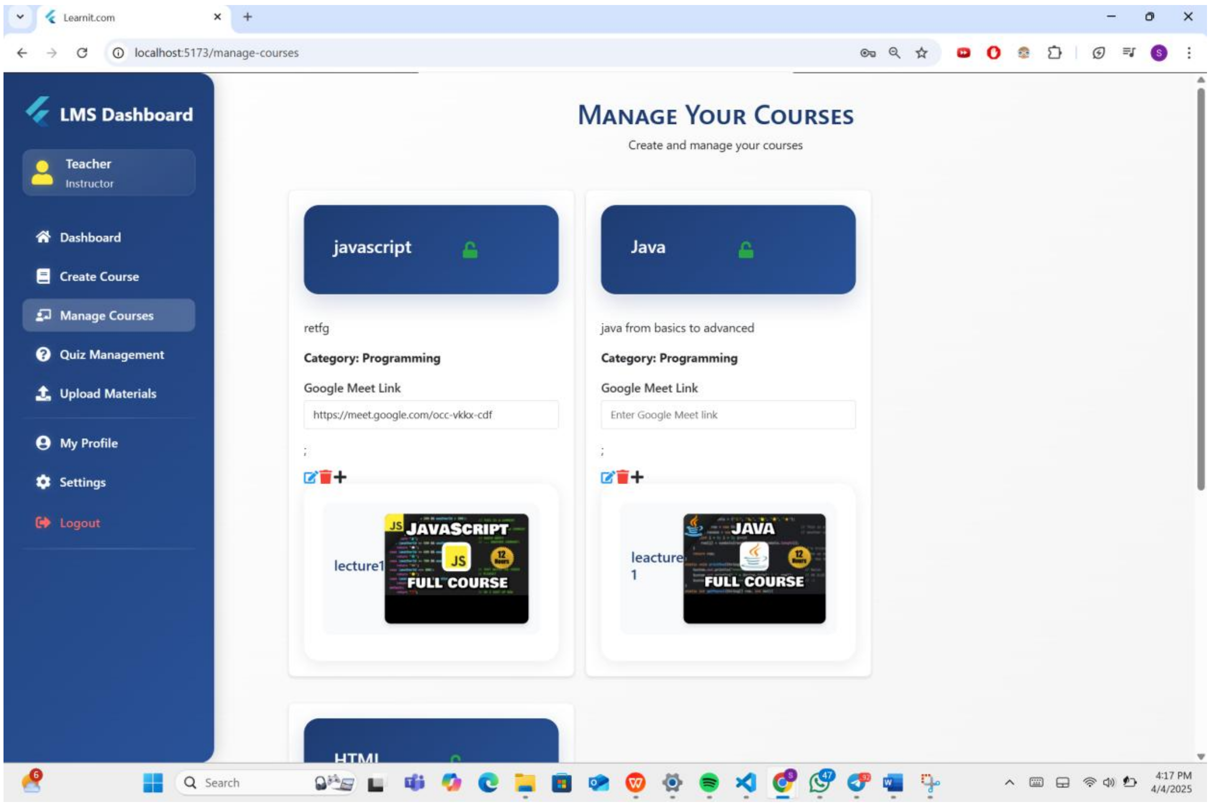
lecture 1

Java

java from basics to advanced

Category: Programming

leacture 1



Learnit.com

localhost:5173/manage-courses

LMS Dashboard

Teacher
Instructor

- Dashboard
- Create Course
- Manage Courses
- Quiz Management
- Upload Materials
- My Profile
- Settings
- Logout

MANAGE YOUR COURSES

Create and manage your courses

javascript

retfg

Category: Programming

Google Meet Link

https://meet.google.com/occ-vkkx-cdf

;

lecture1

Java

java from basics to advanced

Category: Programming

Google Meet Link

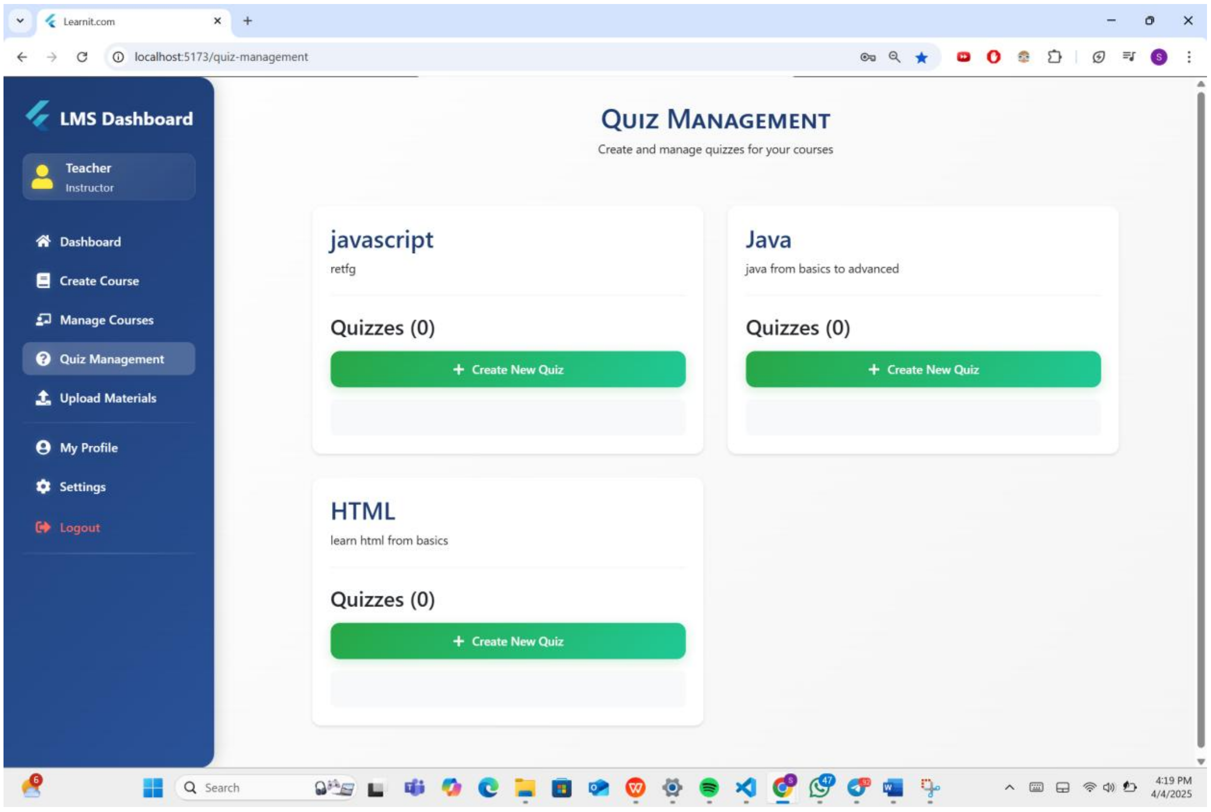
Enter Google Meet link

;

leacture 1

HTML

4:17 PM 4/4/2025



Learnit.com

localhost:5173/quiz-management

LMS Dashboard

Teacher
Instructor

- Dashboard
- Create Course
- Manage Courses
- Quiz Management
- Upload Materials
- My Profile
- Settings
- Logout

QUIZ MANAGEMENT

Create and manage quizzes for your courses

javascript

retfg

Quizzes (0)

+ Create New Quiz

Java

java from basics to advanced

Quizzes (0)

+ Create New Quiz

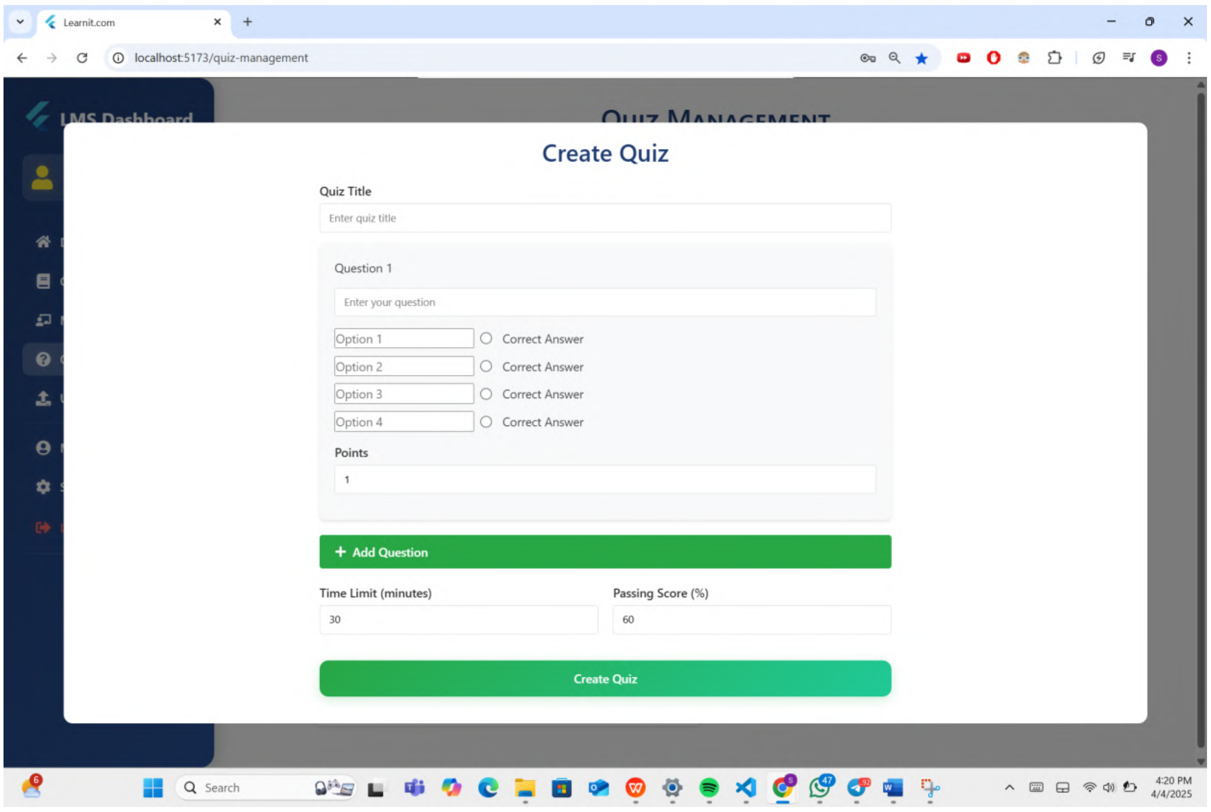
HTML

learn html from basics

Quizzes (0)

+ Create New Quiz

4:19 PM 4/4/2025



Create Quiz

Quiz Title

Enter quiz title

Question 1

Enter your question

Option 1	○ Correct Answer
Option 2	○ Correct Answer
Option 3	○ Correct Answer
Option 4	○ Correct Answer

Points

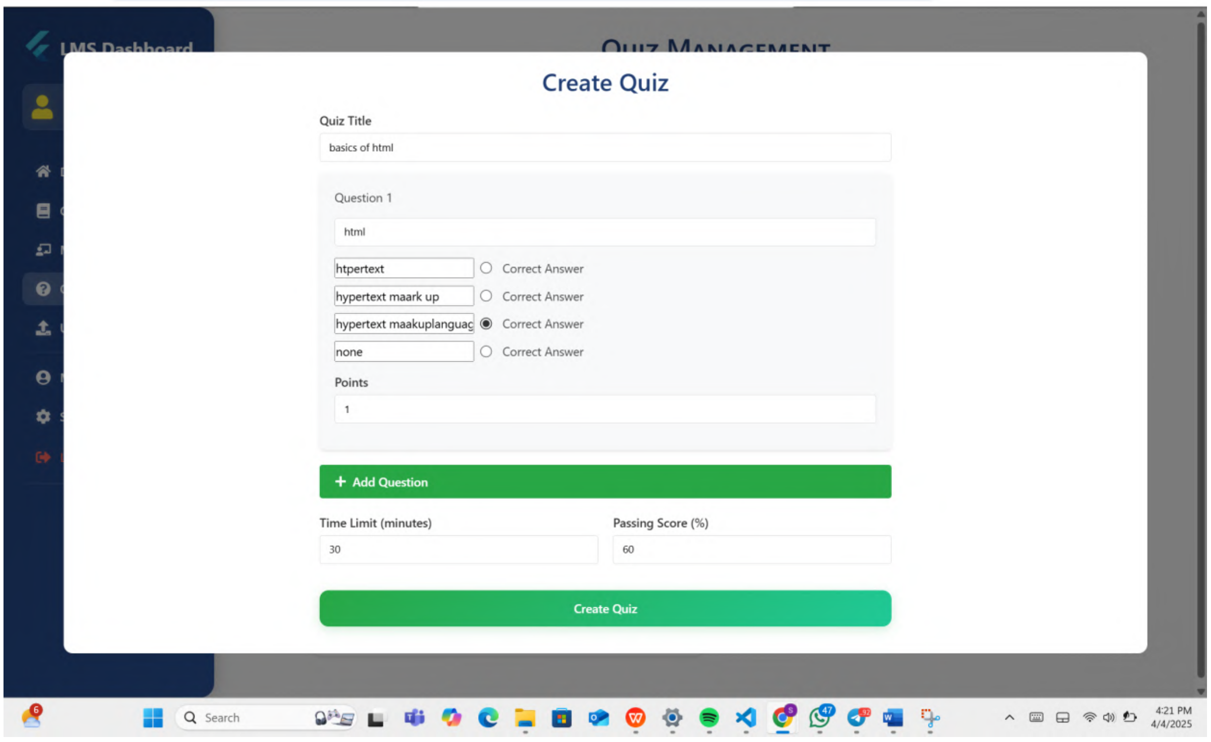
1

Time Limit (minutes)

30

Passing Score (%)

60



Create Quiz

Quiz Title

basics of html

Question 1

html

htpertext	○ Correct Answer
hypertext maark up	○ Correct Answer
hypertext maakuplanguag	● Correct Answer
none	○ Correct Answer

Points

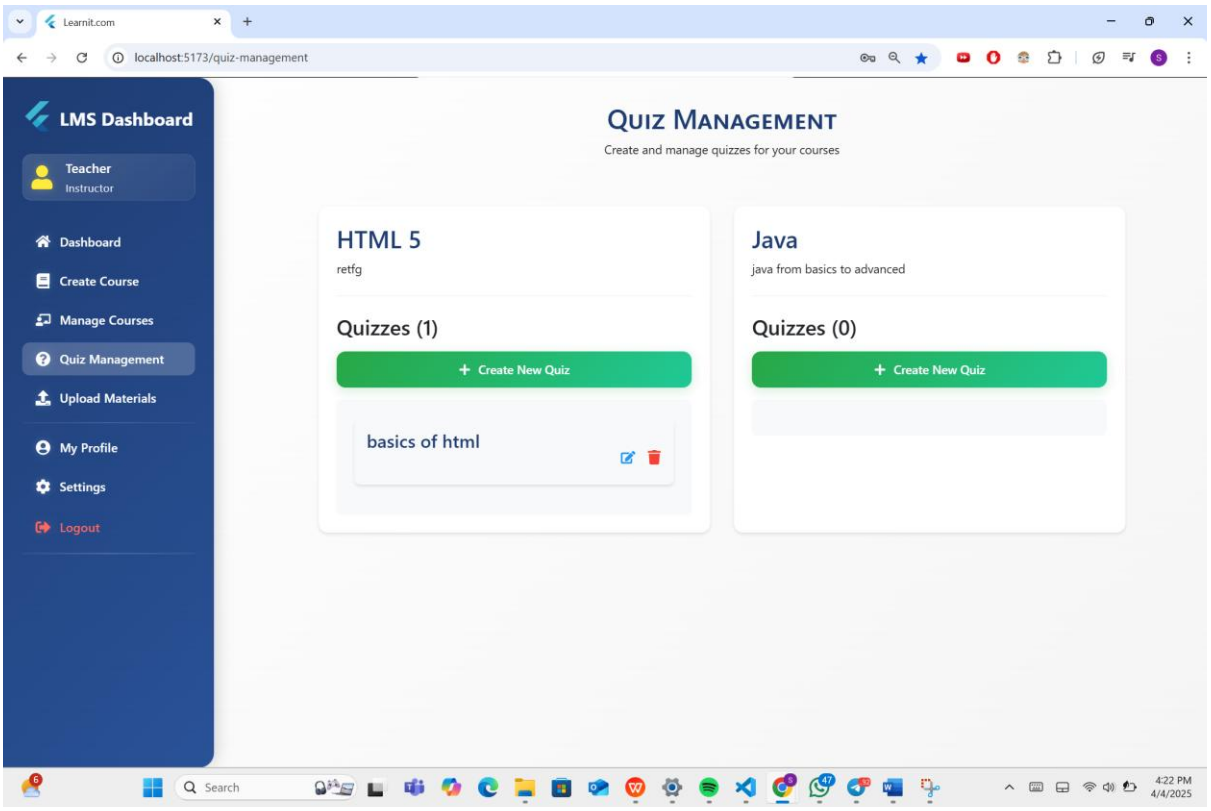
1

Time Limit (minutes)

30

Passing Score (%)

60



Quiz Management

Create and manage quizzes for your courses

HTML 5

retfg

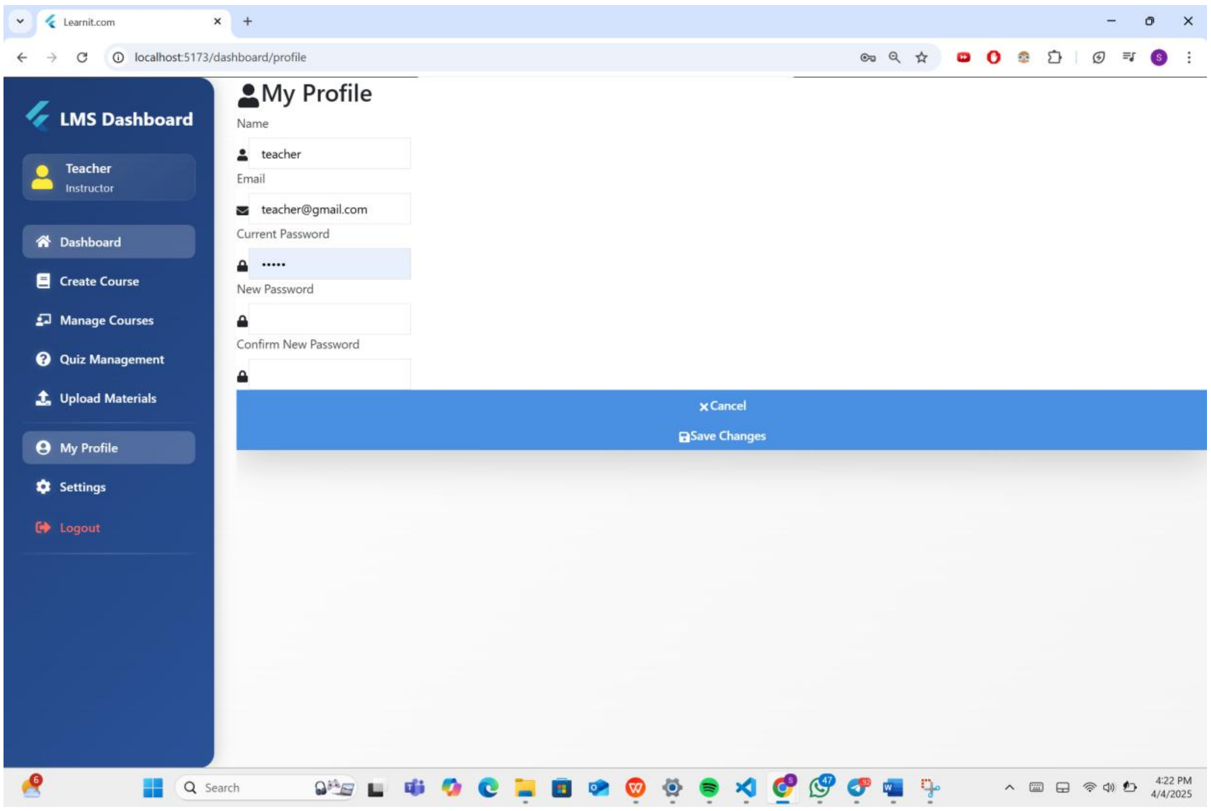
Quizzes (1)

basics of html

Java

java from basics to advanced

Quizzes (0)



My Profile

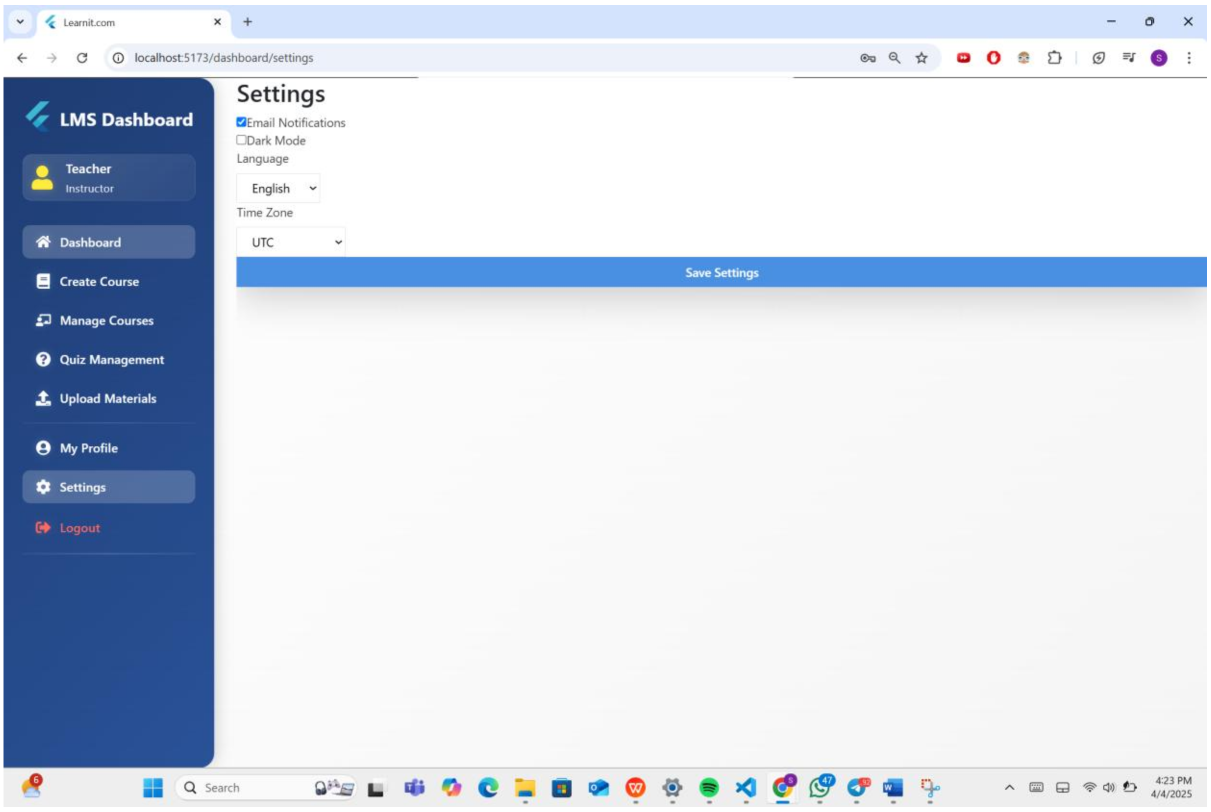
Name: teacher

Email: teacher@gmail.com

Current Password: •••••

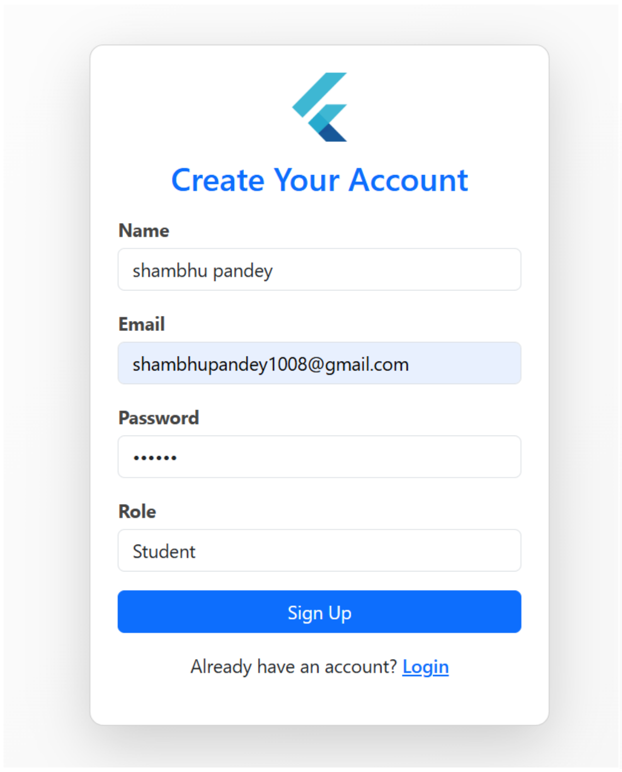
New Password

Confirm New Password



Settings

[✓] Email Notifications
[ ] Dark Mode
Language
English
Time Zone
UTC



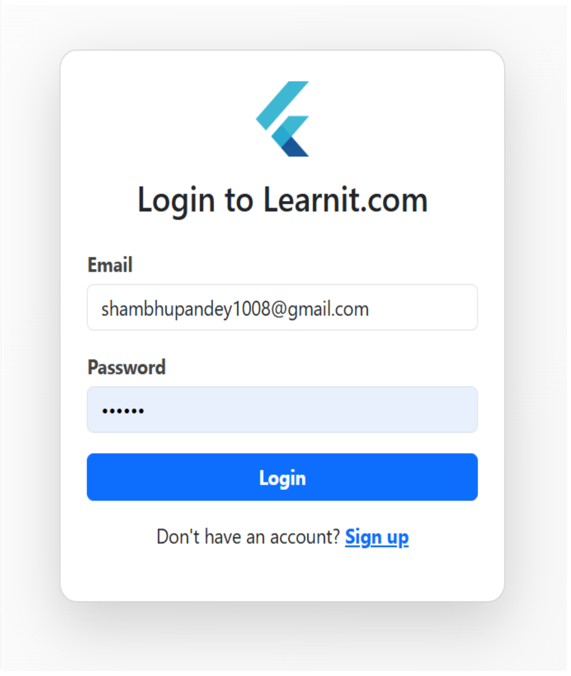
Create Your Account

Name
shambhu pandey

Email
shambhupandey1008@gmail.com

Password
••••••

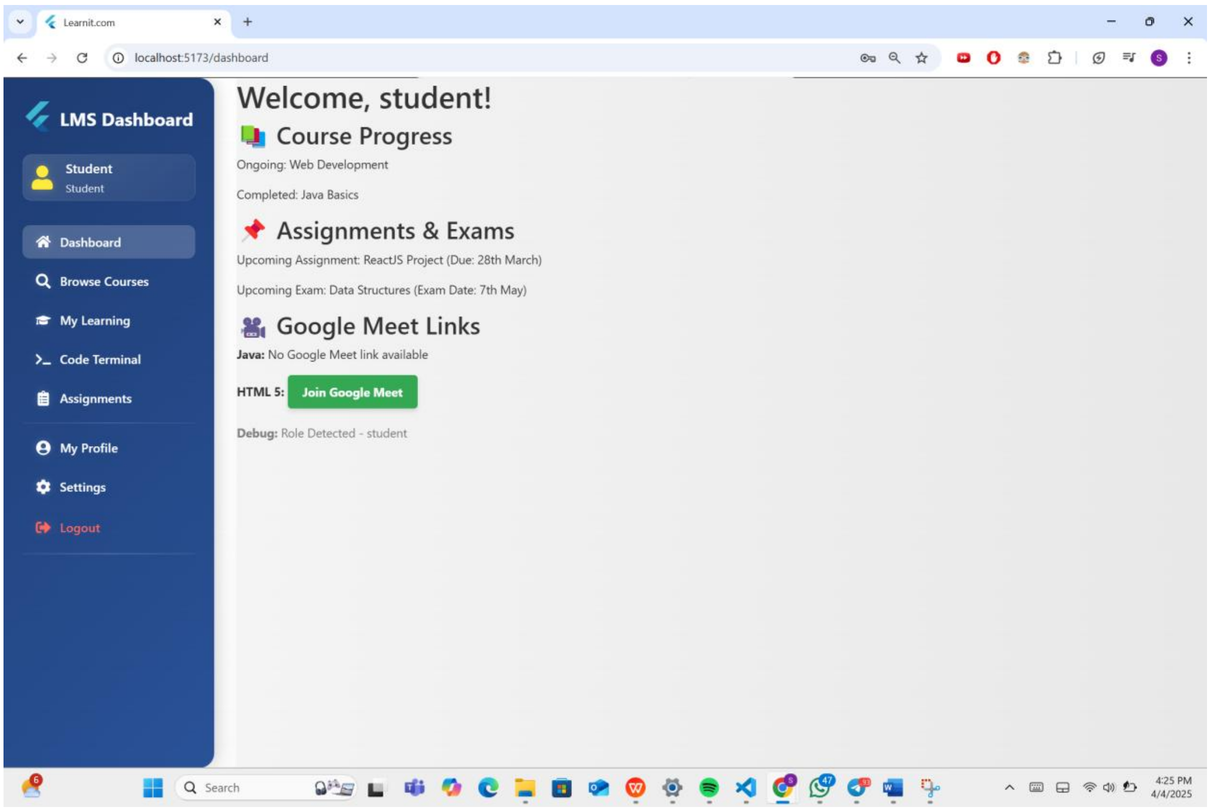
Role
Student



Login to Learnit.com

Email
shambhupandey1008@gmail.com

Password
••••••



localhost:5173/dashboard

LMS Dashboard

Student
Student

Dashboard
Browse Courses
My Learning
Code Terminal
Assignments
My Profile
Settings
Logout

Welcome, student!

Course Progress

Ongoing: Web Development

Completed: Java Basics

Assignments & Exams

Upcoming Assignment: ReactJS Project (Due: 28th March)

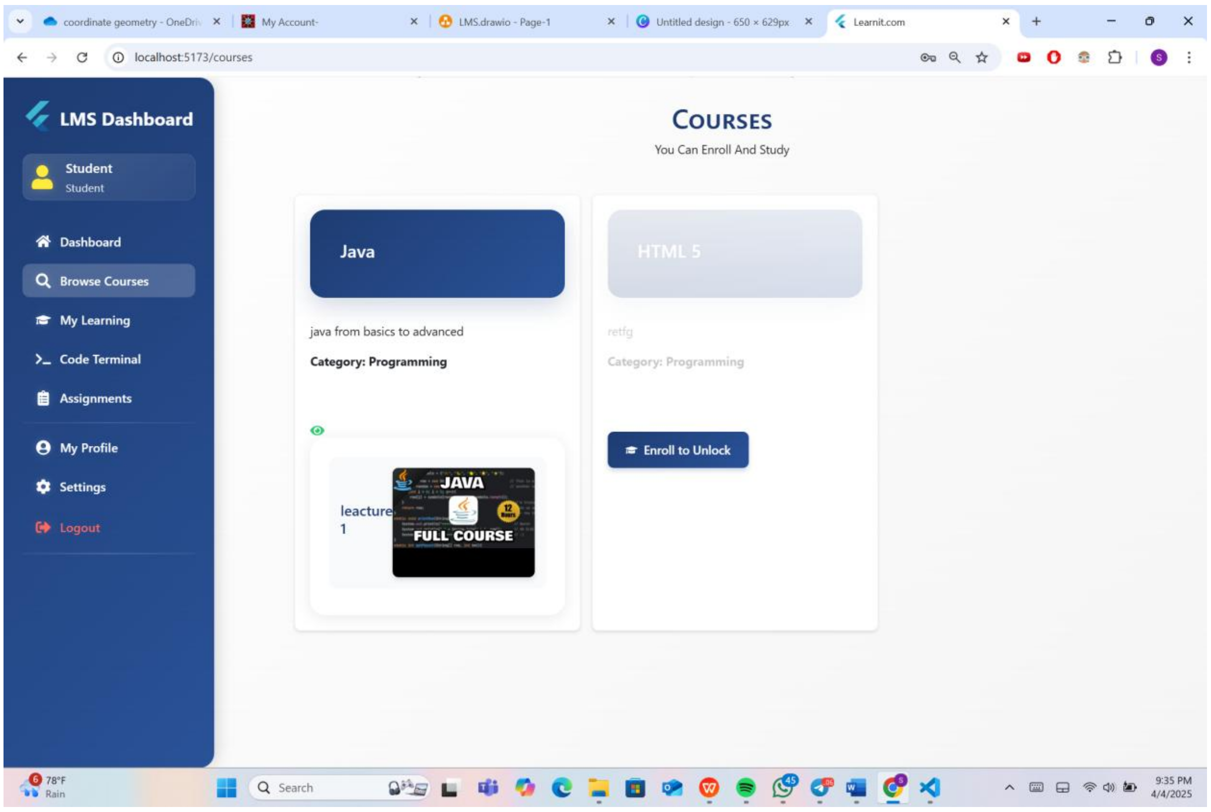
Upcoming Exam: Data Structures (Exam Date: 7th May)

Google Meet Links

Java: No Google Meet link available

HTML 5: Join Google Meet

Debug: Role Detected - student



localhost:5173/courses

LMS Dashboard

Student
Student

Dashboard
Browse Courses
My Learning
Code Terminal
Assignments
My Profile
Settings
Logout

COURSES

You Can Enroll And Study

Java

java from basics to advanced

Category: Programming

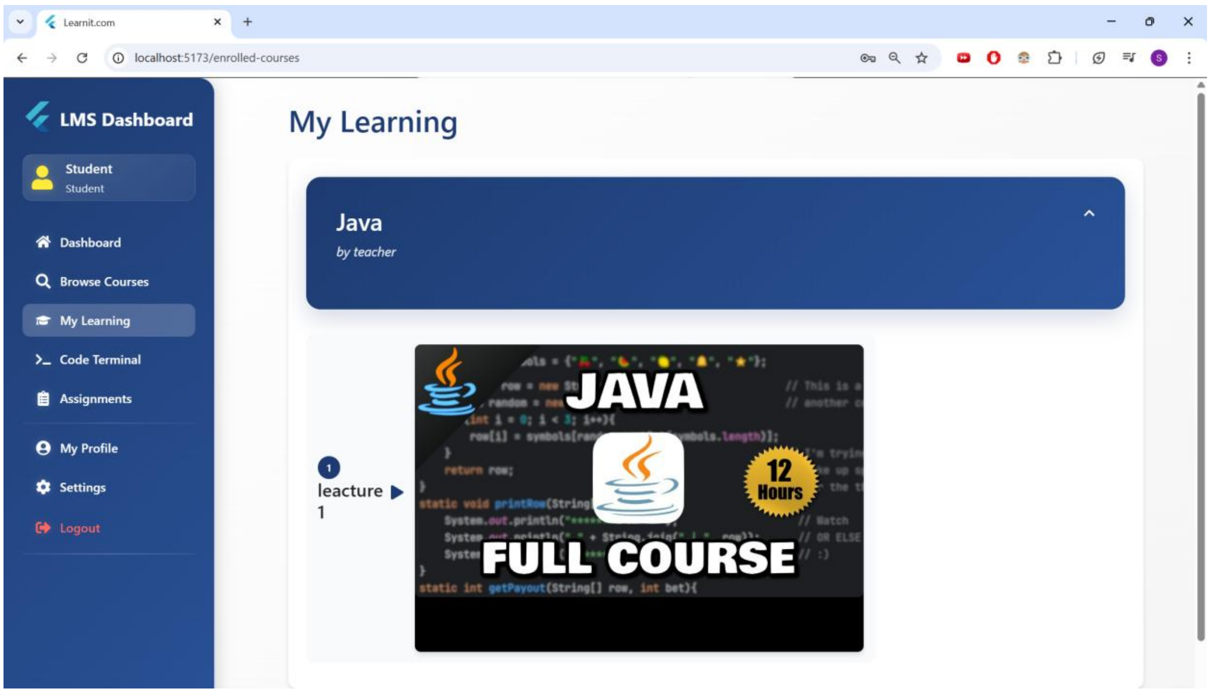
leacture 1

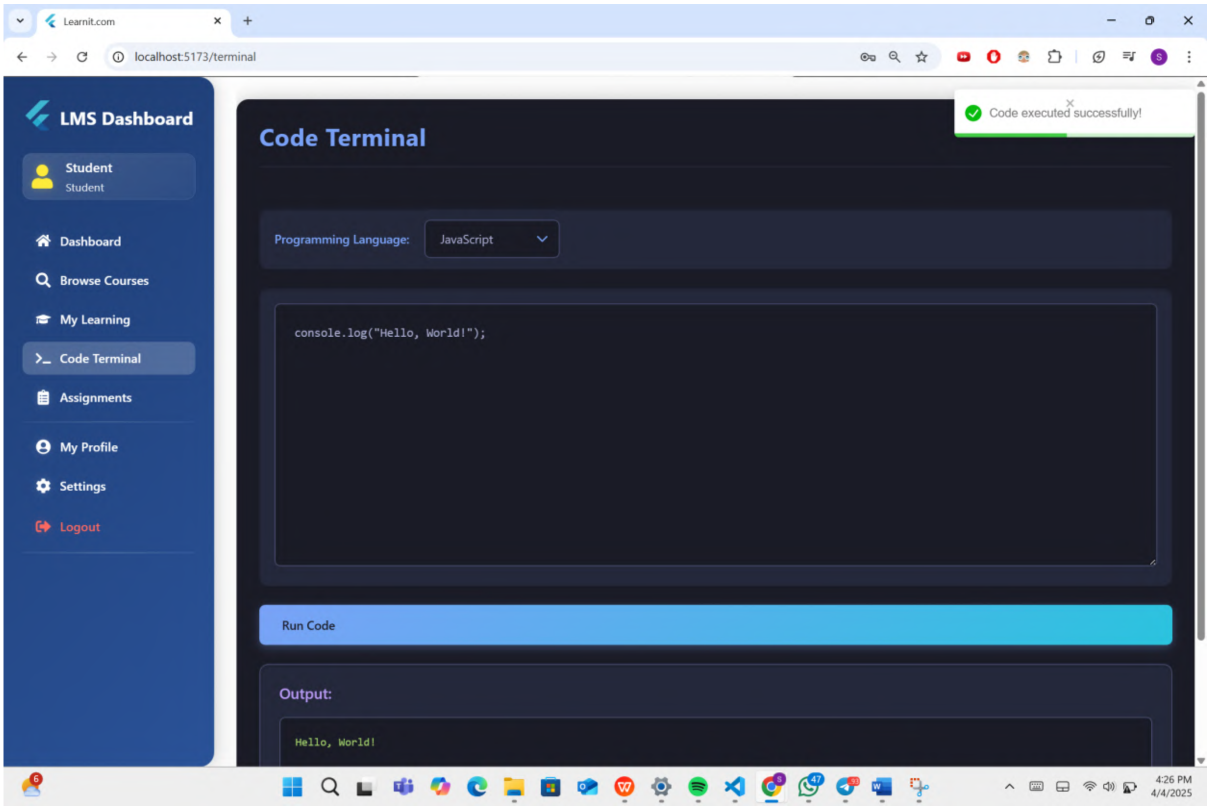
HTML 5

retfg

Category: Programming

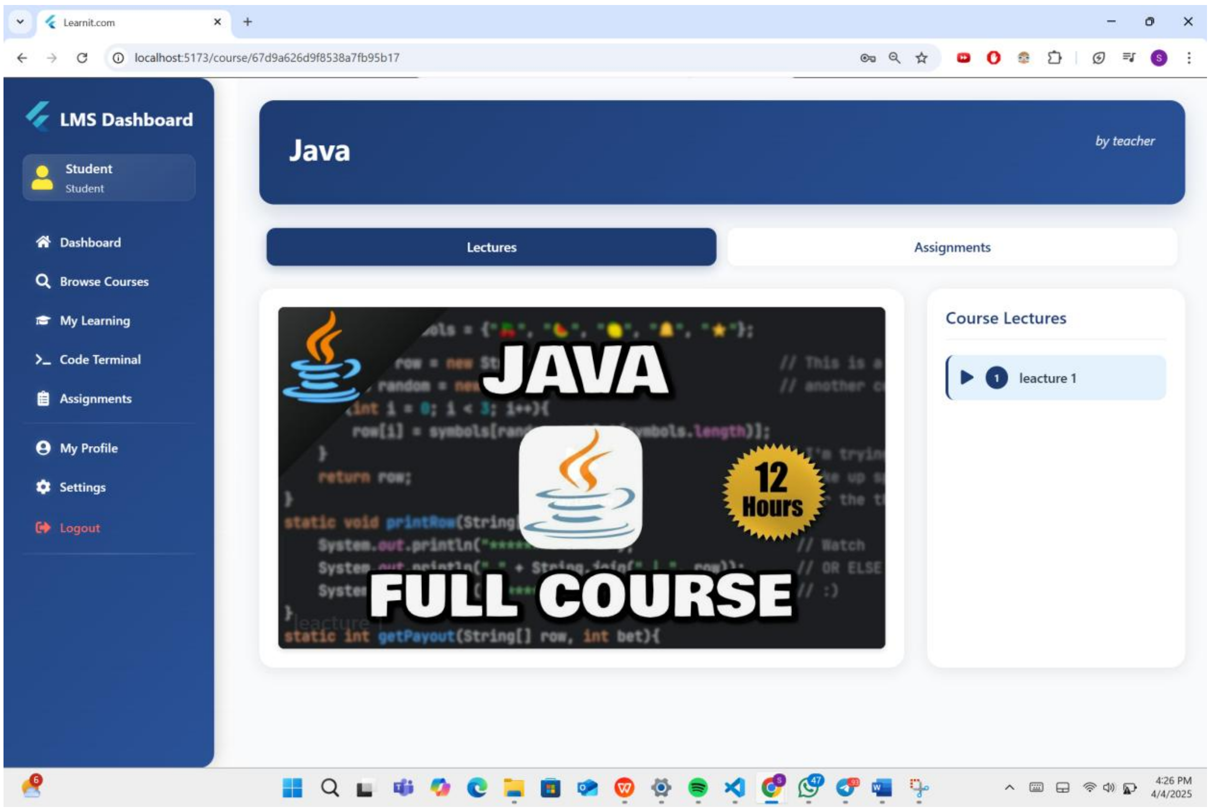
Enroll to Unlock

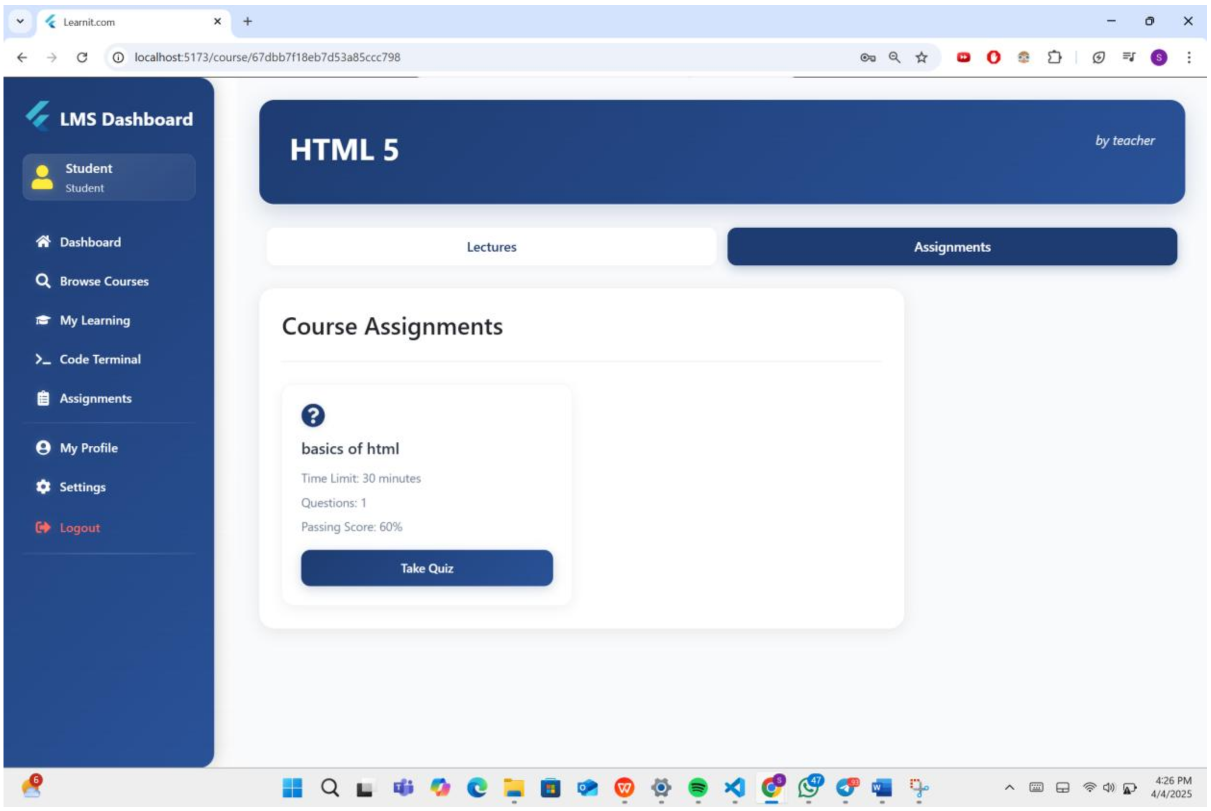


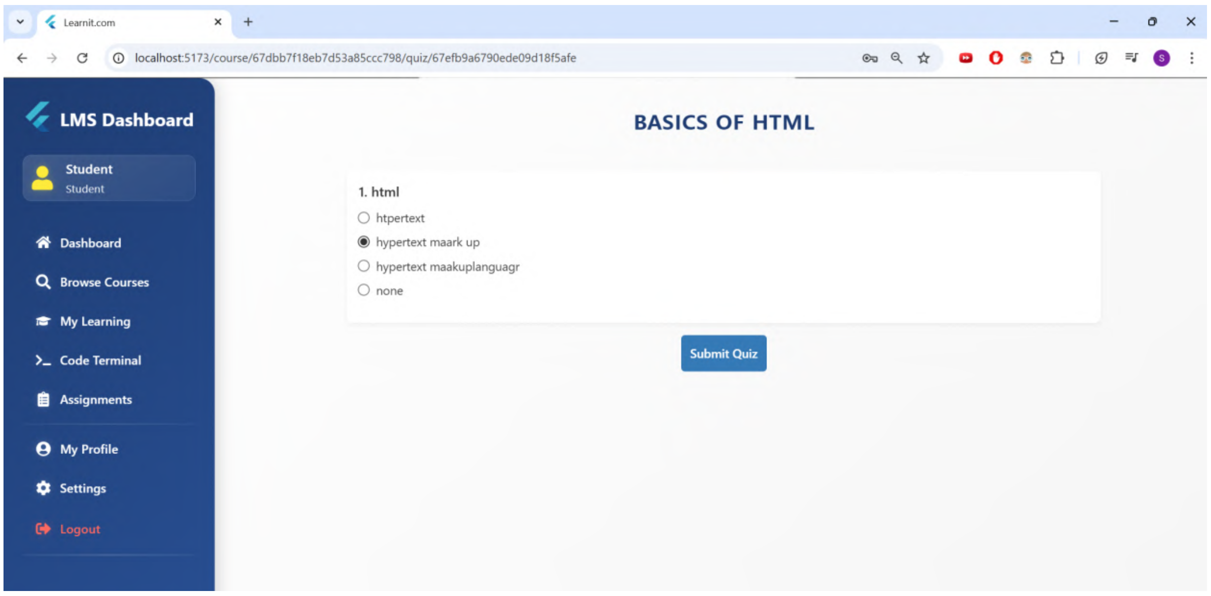


```
console.log("Hello, World!");
```

```
Hello, World!
```


Learnit.com
localhost:5173/course/67d9a626d9f8538a7fb95b17
LMS Dashboard
Student
Student
Dashboard
Browse Courses
My Learning
Code Terminal
Assignments
My Profile
Settings
Logout
Java
by teacher
Lectures
Assignments
JAVA
12 Hours
FULL COURSE
Course Lectures
1 leacture 1
4:26 PM
4/4/2025

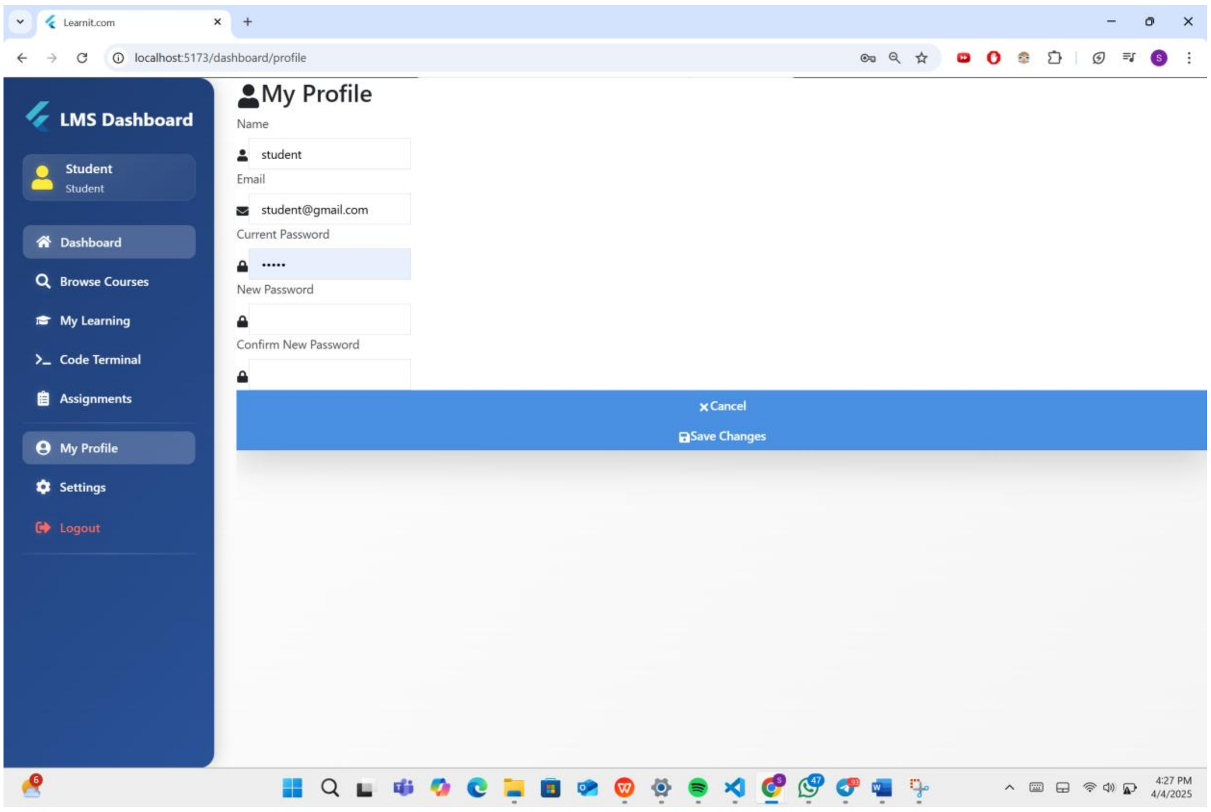

Learnit.com
localhost:5173/course/67dbb7f18eb7d53a85ccc798
LMS Dashboard
Student
Student
Dashboard
Browse Courses
My Learning
Code Terminal
Assignments
My Profile
Settings
Logout
HTML 5
by teacher
Lectures
Assignments
Course Assignments
basics of html
Time Limit: 30 minutes
Questions: 1
Passing Score: 60%
Take Quiz
4:26 PM
4/4/2025



BASICS OF HTML

1. html

- ( ) htpertext
- [✓ hypertext maark up]
- ( ) hypertext maakuplanguagr
- ( ) none



My Profile

Name: student

Email: student@gmail.com

Current Password: •••••

New Password:

Confirm New Password:

7. COST ANALYSIS / RESULT & DISCUSSION

7.1 Cost Analysis

The development of the Learning Management System (LMS) involves various resource requirements, including infrastructure, development tools, hosting, and maintenance. Below are the key components that contribute to the overall project cost:

1. Development Resources

- **Domain & Hosting:** A stable domain and reliable hosting environment are necessary to ensure smooth access and performance.
- **Database Hosting:** Since MongoDB is used as the backend database, a suitable cloud-based or on-premise database management system is required for efficient data storage and retrieval.
- **Software & Development Tools:** Integrated development environments (IDEs), testing frameworks, and deployment tools are required to build and maintain the system.

2. Infrastructure & Maintenance

- **Frontend & Backend Development:** Requires efficient coding practices to ensure a responsive and scalable user interface.
- **Regular Maintenance & Updates:** Ongoing updates are required to fix bugs, enhance security, and introduce new features as needed.

7.2 Results & Discussion

Once the LMS is developed, testing and evaluation are carried out to ensure system reliability and efficiency. Key aspects of discussion include:

1. **Performance & Scalability:**
    - The system should efficiently handle multiple users accessing course content simultaneously.
    - Optimization techniques should be implemented to ensure smooth API responses and database queries.
2. **User Experience & Accessibility:**
    - The platform should be intuitive, user-friendly, and accessible across various devices and screen sizes.
    - Features such as live coding terminals, discussion forums, and multimedia content should enhance the overall learning experience.
3. **Security & Data Protection:**
    - Authentication and authorization mechanisms should ensure user data privacy and prevent unauthorized access.
    - Regular security updates and data backups should be implemented to maintain data integrity and prevent loss.
4. **Deployment & Future Enhancements:**

- The system should be deployed with minimal downtime, ensuring a smooth transition for users.
- Future updates may include additional modules, mobile app integration, and enhanced collaboration features.

This structured cost and result analysis ensures that the LMS remains efficient, scalable, and adaptable for long-term educational use.

8. SUMMARY

8.1 Conclusion

The Learning Management System (LMS) is designed to provide an efficient, scalable, and interactive platform for digital learning. With a robust **tech stack consisting of MongoDB, React.js, Node.js, and Express.js**, the system ensures seamless communication between students, instructors, and administrators. The platform focuses on enhancing the learning experience through **secure authentication, structured course management, real-time collaboration, and multimedia-based learning resources**.

Throughout the development process, various key aspects were prioritized, including:

- **User Authentication & Role Management**: A secure login system with role-based access for students, instructors, and administrators.
- **Course Management**: Instructors can create, update, and manage courses, while students can enroll and access learning materials such as PDFs, videos, and assignments.
- **Live Coding Environment**: Integrated support for multiple programming languages, enabling students to practice coding directly within the LMS.
- **Scalability & Performance Optimization**: The system is built to handle increasing users efficiently by leveraging cloud hosting, load balancing, and caching mechanisms.
- **Security & Data Protection**: Features like encrypted data storage, two-factor authentication, and automated backups ensure the integrity and safety of user data.
- **Collaboration & Communication**: Discussion forums, live chat, and video conferencing integration help in fostering an interactive learning environment.
- **Responsive & User-Friendly Interface**: Designed using **Bootstrap and React.js**, the UI ensures a seamless experience across different devices.
- **Integration with External Tools**: Supports third-party services like **GitHub for project-based assignments, cloud storage for document management, and external APIs for extended functionalities**.

Future Enhancements

As technology evolves, the LMS can be enhanced with additional features such as:

- **Mobile App Development**: Expanding the platform to mobile devices for better accessibility.
- **AI-driven Personalized Learning Paths**: Implementing machine learning algorithms to recommend courses and materials based on user preferences and performance.
- **Multi-language Support**: Enabling students and instructors from different linguistic backgrounds to interact seamlessly.
- **Gamification & Interactive Learning**: Adding quizzes, leaderboards, and badges to enhance student engagement.
- **Advanced Analytics & Reporting**: Providing insights into student progress, course effectiveness, and engagement trends.

9 . REFERENCES:

- **Almarashdeh, I. A., Sahari, N., & Zin, N. A. M. (2011)**
  - *The Success of Learning Management System among Distance Learners in Malaysian Universities*
  - Discusses the effectiveness of LMS in distance learning, focusing on user satisfaction and system success factors.

 Martin, F., & Ndoye, A. (2016)

- *Using Learning Analytics to Assess Student Learning in Online Courses*
- Explores how learning analytics can improve student performance assessment in online environments.

 Graf, S., & List, B. (2005)

- *An Evaluation of Open Source E-Learning Platforms Stressing Adaptation Issues*
- Evaluates open-source e-learning platforms, highlighting their adaptability and customization challenges.

 Wang, Y., & Wu, M. (2009)

- *Comparative Study on Application of Open-Source LMS in Higher Education Institutions of China*
- Examines the use of open-source LMS in Chinese universities, comparing their effectiveness and implementation.

 Al-Busaidi, K. A., & Al-Shihi, H. (2010)

- *Instructors' Acceptance of Learning Management Systems: A Theoretical Framework*

- Presents a framework analyzing instructors' acceptance and usage of LMS.

□ **Pappas, C. (2015)**

- *Top Learning Management Systems of 2023*
- Provides an overview of leading LMS solutions with their features and market trends.

□ **Joo, Y. J., Lim, K. Y., & Kim, J. (2011)**

- *Online University Students' Satisfaction and Persistence*
- Investigates factors like presence, learning outcomes, and satisfaction influencing students' persistence in online learning.

□ **Park, S. Y. (2009)**

- *An Analysis of the Technology Acceptance Model in Understanding University Students' Behavioral Intention to Use e-Learning*
- Explores students' intentions to adopt e-learning using the Technology Acceptance Model (TAM).

□ **Kumar, V., & Kumar, U. (2013)**

- *Cloud Computing in Learning Management Systems: A Study of Emerging Trends*
- Discusses the impact of cloud computing on LMS scalability, accessibility, and cost-effectiveness.

□ **Mohapatra, D., & Biswal, M. (2017)**

- *Role of Learning Analytics in Learning Management Systems*
- Highlights how learning analytics enhances decision-making and personalization in LMS.

□ **Dhawan, S. (2020)**

- *Online Learning: A Panacea in the Time of COVID-19 Crisis*
- Explores the role of online learning platforms during the pandemic, focusing on adaptability and accessibility.

□ **Selim, H. M. (2007)**

- *Critical Success Factors for e-Learning Acceptance: Confirmatory Factor Models*
- Identifies key factors influencing the acceptance of e-learning systems.

□ **Yengin, İ., Karahoca, D., & Karahoca, A. (2010)**

- *E-Learning Success Model for Instructors' Satisfaction in Perspective of Interaction and Usability Outcomes*

- Proposes a model evaluating instructors' satisfaction with LMS in terms of usability and interaction.

 Islam, N., Beer, M., & Slack, F. (2015)

- *E-Learning Challenges Faced by Academics in Higher Education: A Literature Review*
- Reviews challenges in adopting e-learning, including technological and pedagogical issues.

 McGill, T. J., & Klobas, J. E. (2009)

- *Task–Technology Fit View of Learning Management System Impact*
- Examines the impact of LMS on learning outcomes using the Task–Technology Fit (TTF) model.

THANK YOU