

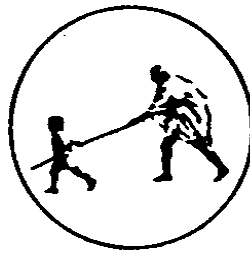
**An
Internship Report
on**

K12 LMS Project

BY

**MRUNMAYEE SURATE
YOGESH BIRAJDAR
NARAYAN KADAM**

**Under the Guidance
of
Ms. D. B. Aghor**



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Mahatma Gandhi Mission's College of Engineering, Nanded (M.S.)

Academic Year 2024-25

An Internship Report on

K12 LMS Project

Submitted to

**DR. BABASAHEB AMBEDKAR TECHNOLOGICAL
UNIVERSITY, LONERE**

in partial fulfillment of the requirement for the degree of

**BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE & ENGINEERING**

By

MRUNMAYEE SURATE

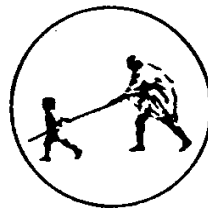
YOGESH BIRAJDAR

NARAYAN KADAM

**Under the Guidance
of**

Ms. D. B. Aghor

(Department of Computer Science and Engineering)



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
MAHATMA GANDHI MISSION'S COLLEGE OF ENGINEERING
NANDED (M.S.)**

Academic Year 2024-25

Certificate



This is to certify that the internship entitled

“K12 LMS Project”

*being submitted by **Miss. Mrunmayee Surate, Yogesh Birajdar and Narayan Kadam** to the Dr. Babasaheb Ambedkar Technological University, Lonere , for the award of the degree of Bachelor of Technology in Computer Science and Engineering, is a record of bonafide work carried out by them under my supervision and guidance. The matter contained in this report has not been submitted to any other university or institute for the award of any degree.*

Ms. D. B. Aghor

Guide

Dr. A. M. Rajurkar

H.O.D

Computer Science & Engineering

Dr. G. S. Lathkar

Director

MGM's College of Engg., Nanded

To,
Mrunmayee Manohar Surate
Dear Mrunmayee ,

5th March 2025

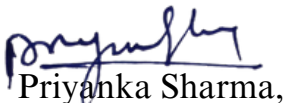
We are delighted to extend an offer to you for the position of Web Developer Intern at Vigyaan Softtech Solutions . We were impressed with your skills and enthusiasm during the interview process, and we believe that your contributions will add significant value to our team.

We are pleased to inform you that this internship comes with a compensation package. During the period of your Internship, You will get a stipend of Rs. 5,000/- per month. You will be paid for your valuable contributions to the team. Additionally, students participating in our internship program may be eligible for certain benefits.

Your primary responsibilities will encompass research and development, as well as design. and development for projects. Your internship is scheduled to commence on 12th March 2025, and will run through 12th July 2025. During this period, you are expected to work approximately 40 hours per week.

We warmly welcome you to the Vigyaan Softtech Solutions team and anticipate a fruitful and rewarding collaboration. If you have any questions or require further clarification, please feel free to reach out.

Best regards,



Priyanka Sharma,
HR Head,
Vigyaan SoftTech Solutions



Bajaj Heights, Pashan Road, Pune



info@vigyaansofttech.in



www.vigyaansofttech.in



+91 9527303009

24th March 2025

To,
Yogesh Sanjay Birajdar

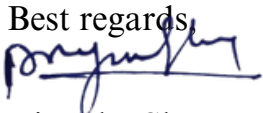
Dear Yogesh,

We are delighted to extend an offer to you for the position of Web Developer Intern at Vigyaan Softtech Solutions . We were impressed with your skills and enthusiasm during the interview process, and we believe that your contributions will add significant value to our team.

We are pleased to inform you that this internship comes with a compensation package. During the period of your Internship, You will get a stipend of Rs. 6,000/- per month. You will be paid for your valuable contributions to the team. Additionally, students participating in our internship program may be eligible for certain benefits.

Your primary responsibilities will encompass research and development, as well as design. and development for projects. Your internship is scheduled to commence on 25th March 2025, and will run through 25th July 2025. During this period, you are expected to work approximately 40 hours per week.

We warmly welcome you to the Vigyaan Softtech Solutions team and anticipate a fruitful and rewarding collaboration. If you have any questions or require further clarification, please feel free to reach out.

Best regards,


Priyanka Sharma,
HR Head,
Vigyaan SoftTech Solutions



Bajaj Heights, Pashan Road, Pune



info@vigyaansofttech.in



www.vigyaansofttech.in



+91 9527303009

To,
Narayan Keshav Kadam

1st March 2025

Dear Narayan,

We are delighted to extend an offer to you for the position of Web Developer Intern at Vigyaan Softtech Solutions . We were impressed with your skills and enthusiasm during the interview process, and we believe that your contributions will add significant value to our team.

We are pleased to inform you that this internship comes with a compensation package. During the period of your Internship, You will get a stipend of Rs. 6,000/- per month. You will be paid for your valuable contributions to the team. Additionally, students participating in our internship program may be eligible for certain benefits.

Your primary responsibilities will encompass research and development, as well as design. and development for projects. Your internship is scheduled to commence on 1st March 2025, and will run through 1st Aug 2025. During this period, you are expected to work approximately 40 hours per week.

We warmly welcome you to the Vigyaan Softtech Solutions team and anticipate a fruitful and rewarding collaboration. If you have any questions or require further clarification, please feel free to reach out.

Best regards,



Priyanka Sharma,
HR Head,
Vigyaan SoftTech Solutions



Bajaj Heights, Pashan Road, Pune



info@vigyaansofttech.in



www.vigyaansofttech.in



+91 9527303009

25 June 2025

PROVISIONAL INTERNSHIP CERTIFICATE

This is to certify that **Mrunmayee Manohar Surate**, a student of **MGM's College Of Engineering, Nanded**, was engaged as an Intern at **Vigyaan SoftTech Solutions** from **10-03-2025 to 10-07-2025** under our **IT Department K12 LMS Project**.

During this period, the student was designated as a Web Developer Intern and was provided with exposure to the following areas:

- Web Development (HTML, CSS, PHP, JavaScript,MySQL etc.)
- Live Project Training under the guidance of experienced professionals
- Use of project management and version control tools.
- Hands-on learning in a collaborative software development environment

Although the internship tenure was **not fully completed**, the student actively participated in the assigned responsibilities during their time with us and demonstrated a willingness to learn and contribute.

Best Regards,
Priyanka Sharma,
HR Head,
Vigyaan SoftTech Solutions



Bajaj Heights,Pashan Road, Pune



info@vigyaansofttech.in



www.vigyaansofttech.in



+91 9527303009

25 June 2025

PROVISIONAL INTERNSHIP CERTIFICATE

This is to certify that **Yogesh Sanjay Birajdar**, a student of **MGM's College Of Engineering, Nanded**, was engaged as an Intern at **Vigyaan SoftTech Solutions** from **25-03-2025 to 25-07-2025** under our **IT Department K12 LMS Project**.

During this period, the student was designated as a Web Developer Intern and was provided with exposure to the following areas:

- Web Development (HTML, CSS, PHP, JavaScript, MySQL etc.)
- Live Project Training under the guidance of experienced professionals
- Use of project management and version control tools.
- Hands-on learning in a collaborative software development environment

Although the internship tenure was **not fully completed**, the student actively participated in the assigned responsibilities during their time with us and demonstrated a willingness to learn and contribute.

Best Regards,
Priyanka Sharma,
HR Head,
Vigyaan SoftTech Solutions



Bajaj Heights, Pashan Road, Pune



www.vigyaansofttech.in



info@vigyaansofttech.in



+91 9527303009

25 June 2025

PROVISIONAL INTERNSHIP CERTIFICATE

This is to certify that **Narayan Keshav Kadam**, a student of **MGM's College Of Engineering, Nanded**, was engaged as an Intern at **Vigyaan SoftTech Solutions** from **01-03-2025 to 01-08-2025** under our **IT Department K12 LMS Project**.

During this period, the student was designated as a Web Developer Intern and was provided with exposure to the following areas:

- Web Development (HTML, CSS, PHP, JavaScript, MySQL etc.)
- Live Project Training under the guidance of experienced professionals
- Use of project management and version control tools.
- Hands-on learning in a collaborative software development environment

Although the internship tenure was **not fully completed**, the student actively participated in the assigned responsibilities during their time with us and demonstrated a willingness to learn and contribute.

Best Regards,
Priyanka Sharma,
HR Head,
Vigyaan SoftTech Solutions



Bajaj Heights, Pashan Road, Pune



www.vigyaansofttech.in



info@vigyaansofttech.in



+91 9527303009

ACKNOWLEDGEMENT

We would like to express our deepest gratitude to our internship guide, **Ms. D. B. Aghor**, for her invaluable support, guidance, and encouragement throughout this project. Her profound knowledge and expertise have been instrumental in the successful completion of this work. Her patience and willingness to assist us at every step have greatly enriched our learning experience. Her constructive feedback and insightful suggestions have not only helped us overcome challenges but also motivated us to strive for excellence.

We gladly take this opportunity to thank **Dr. A. M. Rajurkar** (Head of Computer Science & Engineering, MGM's College of Engineering, Nanded). We are heartily thankful to **Dr. G. S. Lathkar** (Director, MGM's College of Engineering, Nanded) for providing facilities during the progress of the project and also for her kind help, guidance and inspiration. Last but not least, we are also thankful to all those who helped, directly or indirectly, to complete this internship successfully.

With Deep Reverence,

MRUNMAYEE SURATE

YOGESH BIRAJDAR

NARAYAN KADAM

[B.Tech CSE-A]

ABSTRACT

This report presents the development and implementation of a robust Learning Management System (LMS) built as part of a four-month internship at Vigyaan SoftTech Solutions. The LMS was developed using Moodle, a PHP-based open-source platform, and customized with technologies such as HTML, CSS, JavaScript, PHP, and MySQL. The goal of the project was to create a centralized and structured platform that supports course creation, enrollment, learning content delivery, and assessment. The platform featured role-based dashboards for students, teachers, and administrators—allowing each user type to perform their respective tasks efficiently, from managing users and courses to uploading assignments, conducting quizzes, and tracking progress.

Key functionalities included course categorization, secure registration, multimedia learning content, automated quiz scoring, assignment submissions, and certification upon course completion. Additionally, the system supported monetization through Razorpay and Stripe integration, enabling paid course enrollments and transaction tracking. The LMS was designed to be scalable, modular, and suitable for educational institutions and commercial training providers alike. This internship project not only enhanced our technical understanding of LMS platforms but also gave us hands-on experience in building real-world educational infrastructure from the ground up.

TABLE OF CONTENTS

ACKNOWLEDGEMENT	I
ABSTRACT	II
TABLE OF CONTENTS	III
LIST OF FIGURES	V
Chapter 1. INTRODUCTION	1
1.1 Background	1
1.2 Project Overview	2
1.3 Objectives	4
1.4 Scope of Work	5
Chapter 2. TECHNOLOGIES USED	7
2.1 Platform	7
2.2 Programming Languages	8
2.3 Database	10
2.4 Web Server	11
2.5 Hosting Options	12
2.6 Tools Used	13
Chapter 3. FUNCTIONAL MODULES	15
3.1 User Module (Student Panel)	15
3.2 Teacher Module (Instructor Panel)	17
3.3 Admin Module (Administrator Panel)	18
Chapter 4. SYSTEM DESIGN AND PLANNING	20
4.1 Requirement Analysis	20
4.2 Feature Prioritization and Planning	21
4.3 Technical Architecture Design	21
4.4 Role Mapping and User Workflow Design	23
4.5 Tool and Plugin Selection	26
4.6 Database Schema Planning	27
4.7 Security Planning	27

Chapter 5. RESULT AND OUTCOMES	28
5.1 Member-wise Contributions	28
5.2 Feature Integration Snapshots	33
5.3 Additional Achievements	35
5.4 Overall Outcomes	37
CONCLUSION	40
REFERENCES	41

LIST OF FIGURES

Figure No.	Name of Figure	Page No.
1.1	Architecture of a Learning Management System	3
1.2	Working of Learning Management System (LMS)	5
2.1	Programming Languages used	9
4.1	ER Diagram for Learning Management System	20
4.2	Technical Architecture Design	22
4.3	Use Case Diagram for LMS	23
5.1	Personalized Dashboard	29
5.2	User profile editing	29
5.3	Account Creation and Login	30
5.4	Login page	31
5.5	Course and Progress Report	32
5.6	Different Courses in LMS	33

INTRODUCTION

1.1 Background:

As part of our college internship program, we Mrunmayee Surate, Yogesh Birajdar, and Narayan Kadam had the opportunity to work as Web Developer Interns at Vigyaan SoftTech Solutions. The internship began in March 2025 and lasted for four months, concluding in July 2025. The working model was hybrid, combining both offline and online interactions, and our regular working schedule was from Monday to Friday. Throughout this period, we were mentored by Mr. Ganesh Nageshwar, who brought invaluable insights from his experience in software systems and project development.

Vigyaan SoftTech Solutions is a technology-focused company dedicated to enhancing education and business operations through innovative digital platforms. The company specializes in building custom Learning Management Systems (LMS), mobile applications, AI-powered tools, and intelligent automation solutions tailored for academic institutions and enterprises. Their mission is to empower educators and learners by simplifying complex digital workflows and enabling effective e-learning environments.

During our internship, we worked collectively on the design and development of a comprehensive Learning Management System (LMS) using Moodle, a popular and open-source platform based on PHP. The aim was to create a centralized system that could facilitate online course delivery allowing students to register, purchase or access free courses, and study through structured content such as videos, PDFs, and quizzes. On the other hand, the system also included teacher and admin modules for managing content, tracking user activity, and organizing reports.

Our key responsibilities included:

- Developing and customizing core modules within Moodle.
- Designing and coding user interfaces using HTML, CSS, and JavaScript.
- Handling backend logic and data management with PHP and MySQL.

- Integrating learning functionalities such as course progress tracking, certification, and payment gateways.
- Testing modules and ensuring that the platform was functional, scalable, and user-friendly.

This internship greatly enhanced our technical skills in web development, especially in the education technology domain. We also gained practical knowledge of using real-world tools such as phpMyAdmin, Git, and Apache Server, and learned how to collaborate effectively in a development team using version control and modular design principles.

At the end of this internship, we were proud to have received a Certificate of Completion and a Letter of Appreciation from Vigyaan SoftTech Solutions, acknowledging our contribution and the successful completion of the project.

1.2 Project Overview:

During our internship at Vigyaan SoftTech Solutions, we worked as a team to develop a feature-rich Learning Management System (LMS) using Moodle, an open-source PHP-based learning platform. The main goal of this project was to create a centralized online course platform where learners could access educational content and institutions could manage teaching and training workflows efficiently.

The LMS platform was designed to support multiple user roles—students, teachers, and administrators—each with a distinct set of features and functionalities. For students, the system allowed account creation, course browsing, enrollment (free or paid), and access to structured learning resources such as videos, PDFs, quizzes, and assignments. Upon course completion, learners could also download automatically generated certificates.

The teacher module enabled instructors to upload course content, schedule assignments and quizzes, evaluate submissions, and monitor student progress. Teachers could also provide grades and feedback, helping ensure a more interactive and guided learning experience. The admin panel provided complete control over platform management. Admins could manage course categories and sections, assign teachers to courses, add or remove users, monitor enrollments and sales, and track performance

data using dashboards and reports. Integration of live session tools such as Zoom was also planned to support synchronous learning environments.

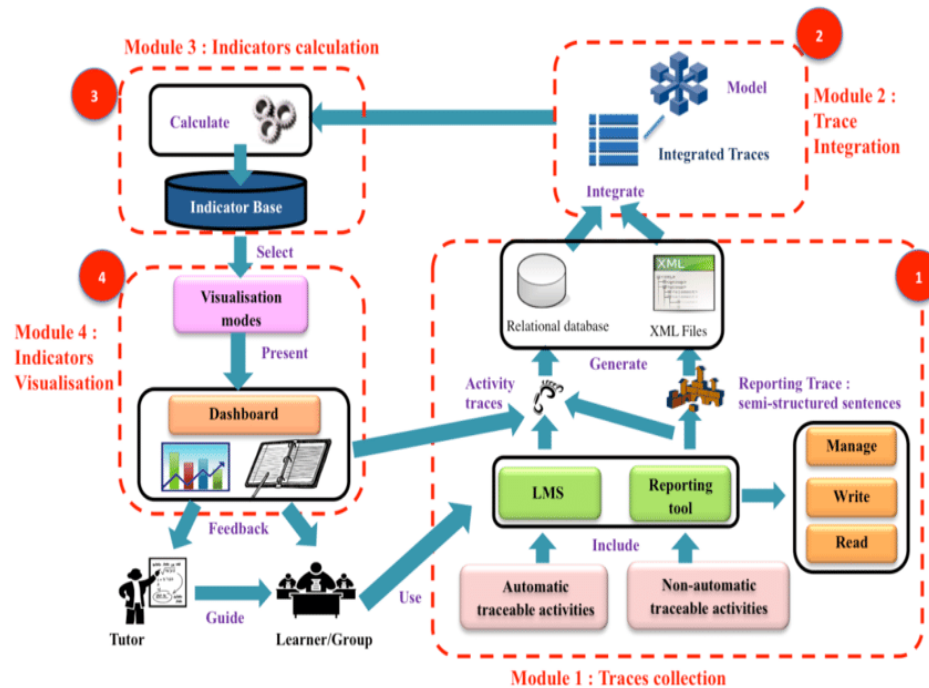


Fig.1.1: Architecture of a Learning Management System

Figure 1.1 shows how our LMS was structured behind the scenes, with clear layers for the user interface, server logic, and database. It also includes tools like payment systems and live classes, all working together to make the platform smooth, secure, and easy to expand. One of the significant features of the platform was the ability to integrate payment gateways, allowing monetization of premium courses. Students could easily complete transactions and access paid learning content without leaving the system.

The development process included backend configuration, frontend development, and design of structured learning paths. We utilized a combination of PHP, HTML, CSS, JavaScript, MySQL, and Moodle plugins to bring the platform to life. Additional tools like phpMyAdmin, VS Code, and Git were used for coding, testing, and version control throughout the project.

This project not only helped us apply the concepts we had learned in our academic coursework but also exposed us to real-world challenges such as module integration, user experience testing, and managing live database operations. It was a

practical learning experience that deepened our understanding of how large-scale educational platforms are structured and maintained.

1.3 Objectives:

The main objective of our internship project was to design and develop a fully functional and user-friendly Learning Management System (LMS) using Moodle, tailored to support students, teachers, and administrators. The platform was envisioned as a digital hub for online education, where learners could easily access content and instructors could effectively deliver and manage courses.

We aimed to build a system that not only delivered academic content but also enabled tracking of progress, assessments, and certifications—all while ensuring ease of use and scalability. The system architecture was designed with modularity and real-world usability in mind.

The specific objectives of the LMS project were as follows:

- To provide a comprehensive online course platform
Create a central system where students can register, log in, browse available courses, and enroll in both free and paid learning programs.
- To implement structured learning workflows
Allow users to access organized content in the form of videos, PDF materials, quizzes, and assignments, along with the ability to download certificates upon course completion.
- To enable teachers to manage course content efficiently
Build tools that let instructors upload videos and documents, schedule quizzes and assignments, track student performance, and provide feedback.
- To build a powerful admin dashboard
Allow administrators to add or manage users (teachers/students), categorize and organize courses, view reports, monitor sales and enrollments, and set up live sessions via platforms like Zoom or BigBlueButton (BBB).
- To integrate optional monetization features
Provide smooth and secure payment gateway integration (like Razorpay or Stripe) to support the sale of premium courses.
- To support progress tracking and analytics

Ensure that students, teachers, and admins can view course progress, completion status, and relevant reports for effective monitoring.



Fig.1.2: Working of Learning Management System (LMS)

Figure 1.2 gives a simple view of how our LMS works from start to finish, showing how users register, access courses, complete lessons, and receive certificates. It helps explain the step-by-step flow of learning and how different parts of the system connect to support that journey. Overall, the objective was to simulate the structure and flow of a modern e-learning environment while gaining practical experience in full-stack web development and platform design within the context of educational technology.

1.4 Scope of Work:

The scope of our internship project focused on the end-to-end development of a Learning Management System (LMS) using Moodle, involving both technical implementation and practical usability. Our goal was to build a robust, easy-to-use platform that supports digital learning for a wide range of users including students, teachers, and administrators.

We worked on each aspect of the platform—from planning and design to development and testing. The tasks were distributed among the group members to encourage collaboration and efficiency, ensuring we met real-world project timelines and standards. The platform was designed to support both educational functionality and administrative management.

Below is an outline of the major areas included in the project scope:

- **Front-End Development**

We developed responsive interfaces using HTML, CSS, and JavaScript to allow smooth navigation for users on both desktop and mobile platforms. The front-end covered login pages, user dashboards, course listings, and progress screens.

- **Back-End Development & Integration**

Using PHP in combination with Moodle's plugin ecosystem, we configured the platform to handle enrollments, user roles, course structures, progress tracking, and certifications. MySQL was used for database management and data persistence.

- **User Module**

Enabled functionalities for students to register/login, manage profiles, enroll in courses, view study material, attempt quizzes, submit assignments, and download completion certificates.

- **Teacher Module**

Allowed instructors to upload video content and PDFs, create quizzes and assignments, monitor student progress, give feedback, and manage course delivery.

- **Admin Module**

Included course and category management, adding/removing users, assigning teachers, tracking enrollments and sales, generating progress reports, and scheduling live sessions through Zoom or BigBlueButton.

- **Payment Gateway Integration**

Added optional support for Razorpay/Stripe to process payments for premium courses and manage transaction records securely.

- **Testing and Debugging**

Conducted regular testing sessions to verify module functionality, usability, and responsiveness across different devices and screen sizes. Debugged issues and implemented fixes to improve performance and reliability.

This project scope allowed us to gain hands-on experience in full-stack development while delivering a real-world solution that can be adapted for schools, coaching centers, or independent educators. It also helped us learn about cross-functional development processes, project ownership, and collaborative decision-making in a technical environment.

TECHNOLOGIES USED

To develop the Learning Management System (LMS), we used a range of modern and industry-standard technologies. These tools and platforms were carefully chosen to ensure the system's performance, scalability, security, and ease of use. As the LMS was built using Moodle, an open-source PHP-based framework specifically designed for educational platforms, our focus was on integrating Moodle's functionalities with additional development tools and services.

The technologies were divided into different categories based on their use in the project, including platform, programming languages, backend systems, frontend development, database management, hosting, and additional tools for development and collaboration.

2.1 Platform:

Moodle (Modular Object-Oriented Dynamic Learning Environment):

Moodle served as the foundational platform for our Learning Management System (LMS) project. It is a widely used, open-source learning platform designed specifically for creating scalable and customizable online education systems. Built using PHP, Moodle offers a modular structure that makes it flexible and extendable, allowing developers to add new features through plugins or modify existing ones according to the institution's or organization's requirements.

What made Moodle an ideal choice for our project was its predefined support for various user roles like students, teachers, and administrators. It comes with built-in tools for course creation, assignment submission, grading, discussion forums, quizzes, and certificates. This saved us significant time and effort compared to building an LMS from scratch.

Additionally, Moodle has a large developer community and excellent documentation, which helped us when we needed to understand how to implement certain features or debug configuration issues. It also supports theme customization, meaning we could modify the appearance of the platform to match our design vision using CSS and HTML, while maintaining its core functionality.

We also made use of Moodle plugins, both official and third-party, to add features such as quiz modules, assignment uploads, live class scheduling, and certificate generation. These plugins are easy to install and configure, making them a crucial part of Moodle's flexibility.

In summary, Moodle acted as the backbone of our LMS, enabling us to focus more on customizing and enhancing user experiences rather than building an entire system from the ground up. It empowered us to create a comprehensive learning environment in a shorter development timeline, with strong support for learning paths, user tracking, content management, and scalability.

2.2 Programming Languages:

The development of our Learning Management System (LMS) involved the use of three primary programming languages: PHP, HTML/CSS, and JavaScript. Each language played a crucial role in building and customizing the functionalities, appearance, and interactivity of the platform.

2.2.1. PHP (Backend Language): PHP (Hypertext Preprocessor) is a powerful server-side scripting language that forms the backbone of Moodle, the platform used for our LMS. PHP was used to handle the backend logic, such as managing user sessions, processing form data, validating enrollments, handling payments, and generating dynamic course content.

With PHP, we were able to extend and customize existing Moodle functionalities to suit our project's specific needs. For instance, we modified certain plugins to better support course progress tracking and added backend logic to process quiz results and certificate generation. PHP also helped in managing database interactions—fetching, inserting, and updating user and course data.

Using PHP in combination with Moodle's API and plugin system gave us full control over platform behavior, allowing us to create a powerful and dynamic e-learning solution.

2.2.2 HTML and CSS (Structure and Styling): HTML (HyperText Markup Language) was used to structure the content on the LMS interface. Every visible part of the platform—headings, buttons, course cards, forms, and layout components—was

constructed using HTML. It helped in defining the layout of student dashboards, registration forms, course pages, and quiz interfaces.

CSS (Cascading Style Sheets), on the other hand, was used to control the appearance and design of the LMS. We used CSS to ensure that the platform looked visually appealing and professional. This included designing responsive layouts that work well on both desktop and mobile devices, choosing consistent font styles and colors, and styling course boxes, buttons, and navigation menus. Together, HTML and CSS allowed us to translate our design mockups into actual web pages and align the visual identity of the platform with user expectations.



Fig.2.1: Programming Languages used

2.2.3 JavaScript (Interactivity and User Experience): JavaScript was used to add interactivity and improve the user experience on the LMS. While HTML and CSS manage structure and appearance, JavaScript makes the interface dynamic—meaning users can interact with it without needing to reload the entire page.

For example, we used JavaScript to:

- Show or hide course descriptions on button click
- Validate forms in real-time before submission
- Handle asynchronous tasks like loading content using AJAX
- Display live feedback (e.g., quiz results or submission confirmations)

JavaScript helped us make the LMS more responsive, intuitive, and user-friendly by ensuring that interactions were quick and smooth. It also improved accessibility by providing better control for users with varying needs and devices.

2.3 Database:

2.3.1 MySQL – The Backbone of LMS Data Storage: For our Learning Management System (LMS) project, we used MySQL, a powerful and widely used open-source relational database management system (RDBMS). It served as the central storage unit for all the data within our platform. Whether it was student details, course information, login credentials, quiz results, or payment records—every piece of structured data was stored, managed, and retrieved through MySQL.

One of the key reasons we chose MySQL is its seamless compatibility with Moodle, which is also built on PHP. Moodle comes with built-in support for MySQL, allowing it to store data efficiently and securely using a relational model. This integration made it easier for us to access, query, and manipulate data within Moodle's architecture without requiring any complex configurations.

2.3.2 How We Used MySQL in the Project:

- **User Management:**

All registered users—students, teachers, and admins—had their profiles stored in MySQL. The database kept track of login information, account status, assigned roles, and permissions.

- **Course Data Storage:**

Course names, content, categories, associated teachers, and enrolled students were stored in relational tables. This allowed the admin to retrieve and display structured course information on demand.

- **Quiz & Assignment Tracking:**

Student responses to quizzes, scores, feedback from instructors, and assignment submissions were recorded in real-time using MySQL queries. This ensured proper record-keeping and evaluation.

- **Certificates and Progress Logs:**

Upon completion of a course, MySQL tables updated the user's status and made them eligible for certificate downloads. We also tracked percentage completion, attempts made, and content viewed.

2.3.3. Why MySQL Was Effective: MySQL helped us achieve a high level of data integrity, reliability, and performance. With its support for indexing, foreign keys, and

optimized query performance, even large datasets (like hundreds of enrolled users and course records) were handled smoothly. We used phpMyAdmin as a graphical interface to interact with the MySQL database—allowing us to view tables, run SQL queries, and manage schemas without manually writing database code.

Another important advantage was data security. MySQL supports role-based access, so we could restrict database actions for users, ensuring that only authorized users could read, insert, update, or delete sensitive records.

2.4 Web Server:

2.4.1. Apache – Powering the LMS Locally: To run and test our Learning Management System (LMS) during the development phase, we used the Apache HTTP Server, one of the most reliable and widely used open-source web servers in the world. Apache acted as the bridge between our code and the web browser, making it possible for us to preview, interact with, and test the LMS in a live-like environment—right on our local machines.

Apache was used to host our Moodle platform locally. This means it served as a local server environment where all the PHP files, stylesheets, scripts, and database connections were executed and rendered. By installing tools like XAMPP (which includes Apache, MySQL, and PHP), we were able to create a lightweight and effective testbed for development.

2.4.2. How Apache Supported Our Project:

- **Local Hosting:**

During the development of the LMS, we hosted the application on a localhost using Apache. This allowed us to access the platform via a web browser and test features like registration, login, course browsing, and content delivery in real time.

- **PHP File Execution:**

Apache played a vital role in processing PHP files, which are core to Moodle. Whenever a user action was triggered (e.g., form submission, page navigation), Apache interpreted the PHP code and generated dynamic HTML content for display.

- **Efficient Debugging:**

By running our LMS locally, we could test individual components—such as the user dashboard, quiz modules, and course uploads—without needing an internet connection. Apache provided real-time error logs, which helped us identify and resolve bugs quickly.

- **Moodle Compatibility:**

Moodle is specifically designed to work seamlessly with Apache servers. This compatibility made installation and configuration smoother, saving us from complex server setup tasks and letting us focus on functionality.

2.4.3. Why Apache Was a Suitable Choice: Apache is known for its stability, security, and flexibility, making it a great choice for both beginner and advanced web projects. It supports a wide range of modules and configurations, which allowed us to tune it to fit Moodle’s specific needs.

Moreover, since Apache is open-source and well-documented, we found a wealth of community resources, forums, and guides that helped us overcome technical hurdles during setup and usage.

2.5 Hosting Options:

2.5.1 Localhost and Cloud Server: During the development of our Learning Management System (LMS), we primarily hosted the platform on a localhost environment using tools like XAMPP, which allowed us to simulate a server setup on our personal computers. Local hosting was a practical choice in the early stages, as it enabled us to work without the need for internet connectivity and made testing and debugging faster and more efficient.

However, as the platform matured and we started exploring deployment readiness, we also studied how the LMS could be migrated to a cloud-based server. Hosting the platform on a cloud server—such as AWS, Google Cloud, or DigitalOcean—offers benefits like remote accessibility, scalability, and performance optimization for a larger audience.

Advantages of Localhost Hosting:

- Easy to set up and configure using Apache (via XAMPP)
- Provides a safe environment for testing changes without affecting live users

- No internet required; suitable for isolated development

Advantages of Cloud Hosting:

- Allows real-time access to users across multiple regions
- Scalable resources depending on user load (more students, more processing power)
- Enables deployment of live classes, user analytics, and real-time progress updates

In summary, we used localhost during our internship phase to develop and refine features. In a real-world production environment, this LMS could be easily deployed to a cloud server to make it accessible to students and instructors worldwide.

2.6 Tools Used:

To ensure smooth development, testing, collaboration, and deployment of the Learning Management System (LMS), we relied on a set of essential tools widely used in the software development industry. These tools enhanced our productivity, helped manage complexity, and allowed for seamless integration between various components of the project.

2.6.1. Visual Studio Code (VS Code): Visual Studio Code, commonly known as VS Code, was our primary code editor throughout the project. It provided an efficient and customizable environment to write, edit, and debug code in multiple languages—especially PHP, HTML, CSS, and JavaScript. VS Code's intelligent syntax highlighting, extensions (such as PHP Intelephense and Live Server), and integrated terminal greatly simplified our workflow.

Additionally, we used Git integration within VS Code to manage version control without switching tools, making team collaboration more organized and convenient.

2.6.2 phpMyAdmin: phpMyAdmin is a web-based tool for managing MySQL databases, and it was instrumental in handling the LMS database during development. Through a clean graphical interface, we were able to:

- View and modify database tables
- Run SQL queries
- Monitor user and course records

- Backup and export data

This tool saved time and reduced the chances of manual errors when compared to writing raw SQL in a command-line environment.

2.6.3. Git: Git is a powerful version control system that allowed us to track changes in our codebase, collaborate as a team, and maintain clean project history. Using Git, we could

- Create separate branches for features
- Merge and resolve code conflicts
- Revert to previous stable versions
- Document each stage of development via commit messages

This not only kept our work organized but also gave us a real-world experience of how professional teams manage collaborative development.

2.6.4. Moodle Plugins: Moodle comes with a powerful plugin ecosystem that allows additional features to be added without reinventing the wheel. We explored and used various official and third-party plugins to extend the LMS functionality. These included:

- Quiz plugins for interactive assessments
- Certificate plugins for automated course completion documents
- Assignment submission tools
- Zoom/BBB integrations for live classes
- Progress tracking and reports modules

Plugins made it easy to tailor the platform to our project's requirements and helped reduce development time significantly.

FUNCTIONAL MODULES

Our Learning Management System (LMS) was structured to serve three major user roles: Students, Teachers, and Administrators. Each role was assigned its own module within the system, providing a personalized set of tools and workflows based on specific user needs. This role-based modularity ensured that the platform remained intuitive, functional, and secure for all users.

By segregating responsibilities and functionalities based on user roles, we could streamline the learning experience for students, reduce workload complexity for teachers, and centralize control and monitoring for administrators. Below is a breakdown of each module and the features it includes.

3.1 Student Module (User Panel):

The student module is designed to be learner-centric and intuitive. A great deal of attention was given to designing user interfaces that simplify learning and allow users to focus entirely on their educational journey. Features are organized to reduce cognitive overload while maximizing engagement, progress awareness, and goal completion. Students today expect modern, responsive platforms that not only deliver content but also offer personalization, interactivity, and achievement tracking. We accounted for these needs while developing the student module, ensuring that students could learn at their own pace while feeling supported and guided by the system.

Features:

- **Account Creation and Login**

Students begin by registering with their email address. The registration form includes input validation, and passwords are securely stored. Upon logging in, the system redirects each student to their personalized dashboard using session-based authentication.

- **Personalized Dashboard**

Each student is greeted with a dashboard showing current courses, completion percentages, deadlines, announcements, and quick access to learning resources. It also includes reminders for upcoming assignments and quiz deadlines.

- **Course Browsing and Enrollment**

Courses are organized by categories and include filters like “Free” or “Paid.” Students can preview lessons, read course descriptions, and enroll with a single click. Paid courses are routed through Razorpay or Stripe for secure transactions.

- **Structured Content Access**

Once enrolled, students access modules that include videos, PDFs, and slides. Content is displayed in a chapter-wise format to maintain a clear learning path. Some modules unlock only after completion of the previous ones.

- **Quizzes and Assignments**

Interactive quizzes are integrated at the end of each module. Students receive instant results on objective questions. Assignments are submitted directly on the platform and evaluated by the teacher with feedback.

- **Progress Tracking**

Each course includes a visual progress bar. As students complete modules, pass quizzes, or submit assignments, the system dynamically updates their progress status.

- **Certificate Generation**

On completing all course requirements, the system generates a certificate personalized with the student's name and course details. This can be downloaded or shared.

Key Advantages:

- Reduces dropouts through visual progress indicators.
- Encourages course completion with milestones and certifications.
- Provides mobile accessibility for learners without desktop devices.

Real-Life Scenario:

A student logs in and immediately sees they’ve completed 70% of their programming course. The next module is locked until they submit an assignment. After uploading a PDF solution, the module unlocks, and they proceed with the next video. At the end of the week, they download their certificate and share it on LinkedIn directly from the platform.

3.2 Teacher Module (Instructor Panel):

The teacher module was created to serve as a digital classroom management system. The focus here was to reduce repetitive tasks like grading, scheduling, and content uploading—by automating them wherever possible. Teachers are provided with robust analytics, notifications, and a clean interface to manage their responsibilities. We consulted educational platforms and real-world LMS benchmarks to ensure our features aligned with modern teaching requirements. This included automated grading, quiz banks, asynchronous teaching tools, and content reuse across batches.

Features:

- **Course Management Tools**

Each teacher is assigned specific courses. They can edit course names, descriptions, and learning objectives. Teachers are responsible for keeping content up to date and relevant.

- **Content Upload and Organization**

The system supports video lectures, document uploads, and external links. Teachers can organize content into topics or weeks. Lessons can be made available immediately or on a schedule.

- **Quiz and Assignment Creation**

Teachers build custom assessments using various question types. Assignments allow for document uploads and can include submission deadlines, instructions, and grading rubrics.

- **Grading and Feedback**

Teachers review submissions via their dashboard, assign grades, and write comments. Feedback is visible to the student in the “My Reports” section.

- **Performance Monitoring**

Charts and tables show overall class statistics, including pass/fail rates, quiz averages, and student participation. These tools assist teachers in identifying weak points and adjusting their instruction accordingly.

Key Advantages:

- Reduces dropouts through visual progress indicators.

- Encourages course completion with milestones and certifications.
- Provides mobile accessibility for learners without desktop devices.

Real-Life Scenario:

A student logs in and immediately sees they've completed 70% of their programming course. The next module is locked until they submit an assignment. After uploading a PDF solution, the module unlocks, and they proceed with the next video. At the end of the week, they download their certificate and share it on LinkedIn directly from the platform.

3.3 Admin Module (Administrator Panel):

The teacher module was created to serve as a digital classroom management system. The focus here was to reduce repetitive tasks like grading, scheduling, and content uploading—by automating them wherever possible. Teachers are provided with robust analytics, notifications, and a clean interface to manage their responsibilities. We consulted educational platforms and real-world LMS benchmarks to ensure our features aligned with modern teaching requirements. This included automated grading, quiz banks, asynchronous teaching tools, and content reuse across batches.

Features:

- **Dashboard Overview**
Admins see system-wide statistics like user counts, active courses, login activity, and sales. The interface offers fast insights into system performance.
- **User Management**
Admins can create or remove accounts and change roles between student, teacher, and admin. Password reset and manual course enrollment tools are also included.
- **Course Oversight**
Before going live, all new courses must be approved by an admin. They can manage categories, deactivate old content, or assign new instructors.
- **Enrollment and Payment Monitoring**
All financial transactions are logged. Admins can filter by course, date, or user. Each entry includes amount, transaction status, and gateway reference.

- **Analytics and Reporting**

The admin panel includes downloadable reports on:

- Course popularity
- Certification rates
- Teacher activity
- Student engagement

- **Live Session Control**

Admins can add, monitor, or cancel scheduled live classes. They ensure that tools like Zoom or BBB are configured properly and available during scheduled times.

- **Security and Settings**

Admins manage role permissions, file upload limits, backup schedules, and general settings. Security logs and data access policies are enforced to protect sensitive information.

Key Benefits:

- Saves instructor time with automation and template reuse.
- Encourages better engagement with real-time performance insights.
- Supports remote and hybrid learning models seamlessly.

Real-Life Scenario:

An instructor uploads a week's worth of material on Monday, schedules a live session for Thursday, and sets a quiz deadline for Sunday. By the following week, they've graded all submissions and reviewed analytics showing which students struggled, helping them plan the next module accordingly.

SYSTEM DESIGN AND PLANNING

Before beginning development on our Learning Management System (LMS), our team followed a structured design and planning phase to ensure the project was technically sound, user-oriented, and scalable. This chapter describes the systematic approach we used to bridge the gap between project objectives and real-world implementation. It covers requirement gathering, feature planning, architecture design, user workflow mapping, plugin/tool selection, database schema modeling, and security considerations.

4.1 Requirement Analysis:

Requirement analysis is the cornerstone of any successful software project. In our case, we began by identifying the different stakeholders of the LMS and understanding their expectations and needs. We divided our users into three key roles: students, teachers, and administrators. Each of these roles demanded specific features and access levels, which helped shape our entire design process.

Figure 4.1 represents the ER (Entity-Relationship) diagram of our Learning Management System, mapping out how key elements like users, courses, enrollments, quizzes, and certificates are linked. It visually explains the structure of the database and how different entities interact to support system functionality.

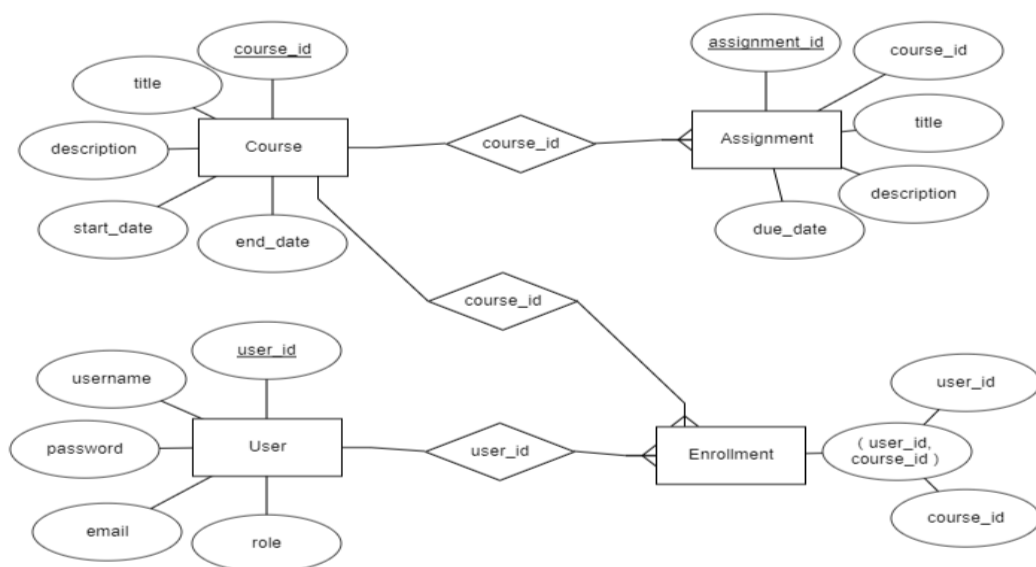


Fig.4.1 ER Diagram for Learning Management System

- Students needed a platform to browse, enroll, and learn from structured content. They required progress tracking, assessments, and certification.
- Teachers required tools for uploading content, creating and grading assessments, and monitoring student performance.
- Administrators needed full control over platform management, including course creation, role assignment, user management, reporting, and plugin integration.

We gathered requirements through internal brainstorming, review of Moodle documentation, and referencing existing LMS platforms. This analysis enabled us to define precise deliverables for each module and role, reducing confusion and scope creep later in development.

4.2 Feature Prioritization and Planning:

After requirements were documented, the next step was to prioritize features. Given our limited development timeline of four months, we needed to strategically plan which features would be implemented first. We used the MoSCoW method (Must have, Should have, Could have, Won't have) to classify features.

- **Must-Have Features:** Role-based access, secure login, course catalog, video/PDF content delivery, quizzes, assignments, progress tracking, certification.
- **Should-Have Features:** Payment gateway integration, admin dashboards, analytics, assignment grading interface.
- **Could-Have Features:** Live class integration (Zoom/BBB), student chat, announcement system.
- **Won't-Have Features:** Gamification, peer grading, multilingual support.

This categorization allowed us to focus our efforts on essential functionalities, ensuring the LMS was complete, stable, and testable before we added advanced or optional features.

4.3 Technical Architecture Design:

A solid technical architecture forms the backbone of any robust application. We followed a three-tier architecture to ensure modularity and separation of concerns:

- **Presentation Layer:** Built using HTML, CSS, and JavaScript, this layer served as the user interface. Each role had its own customized dashboard for streamlined interaction.
- **Application Logic Layer:** This layer, developed in PHP and integrated with Moodle, handled all backend operations including user validation, course access, quiz scoring, and data processing.
- **Data Layer:** MySQL served as the persistent data store. It held structured tables for users, courses, enrollments, grades, transactions, and more.

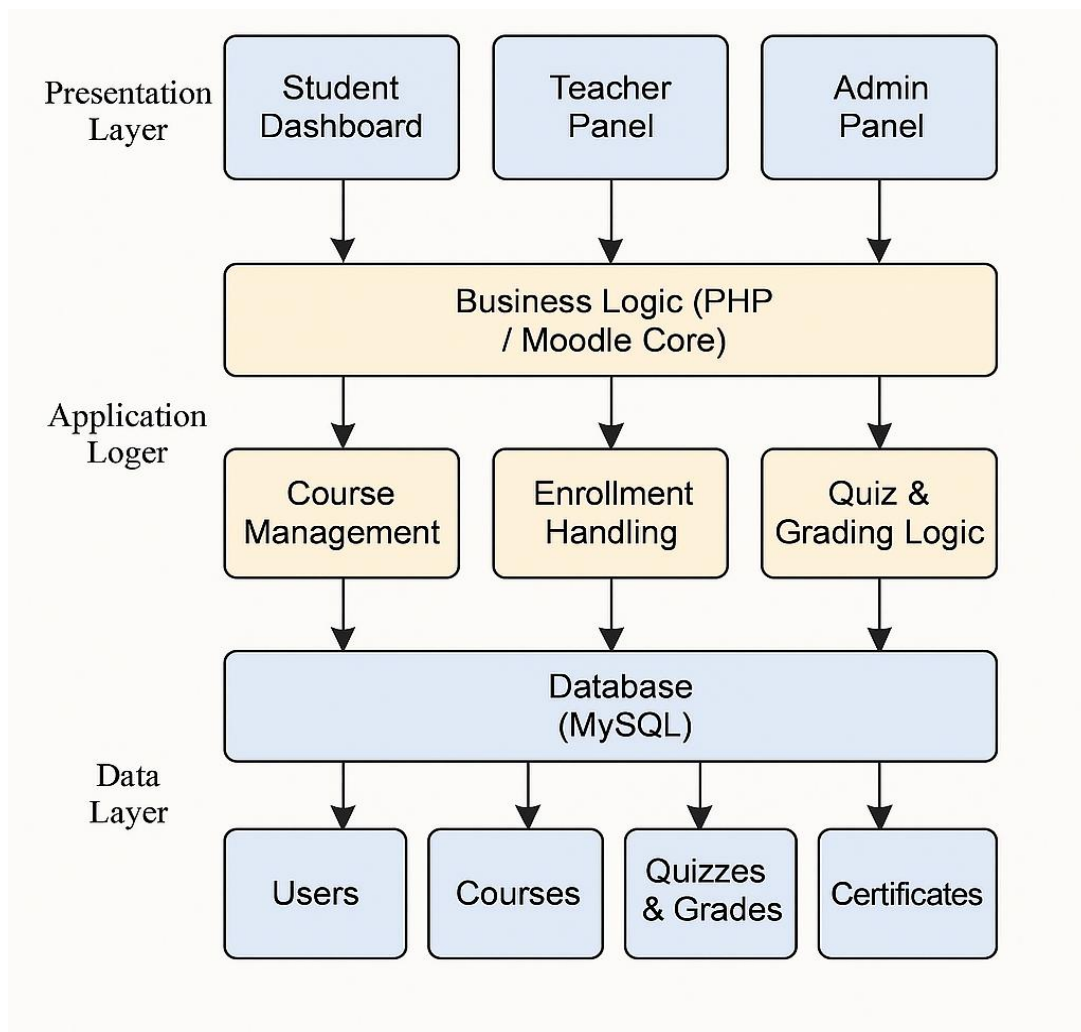


Fig.4.2 Technical Architecture Design

Figure 4.2 illustrates the technical architecture design of the LMS, showing how the front-end, back-end, database, and third-party services like payment gateways and live class tools communicate. It highlights the system's layered structure, ensuring secure, scalable, and efficient performance. This architecture allowed us to isolate and

test each part of the system independently. For example, frontend designs could be tested without needing a live database, and database performance could be monitored separately from user interactions.

4.4 Role Mapping and User Workflow Design:

In any Learning Management System (LMS), understanding how different types of users interact with the system is crucial to building an intuitive and efficient platform. During the initial planning phase of our internship, we designed detailed user workflows for each major role—Student, Teacher, and Administrator. These workflows served as visual and functional guides that helped ensure all modules aligned with user expectations and system objectives.

In Figure 4.3 we minimized ambiguity during module integration and improved the overall user experience. Each workflow was defined based on real-world learning scenarios and responsibilities that are typically assigned to those roles. The following outlines the role-specific workflows that were used to guide development and testing.

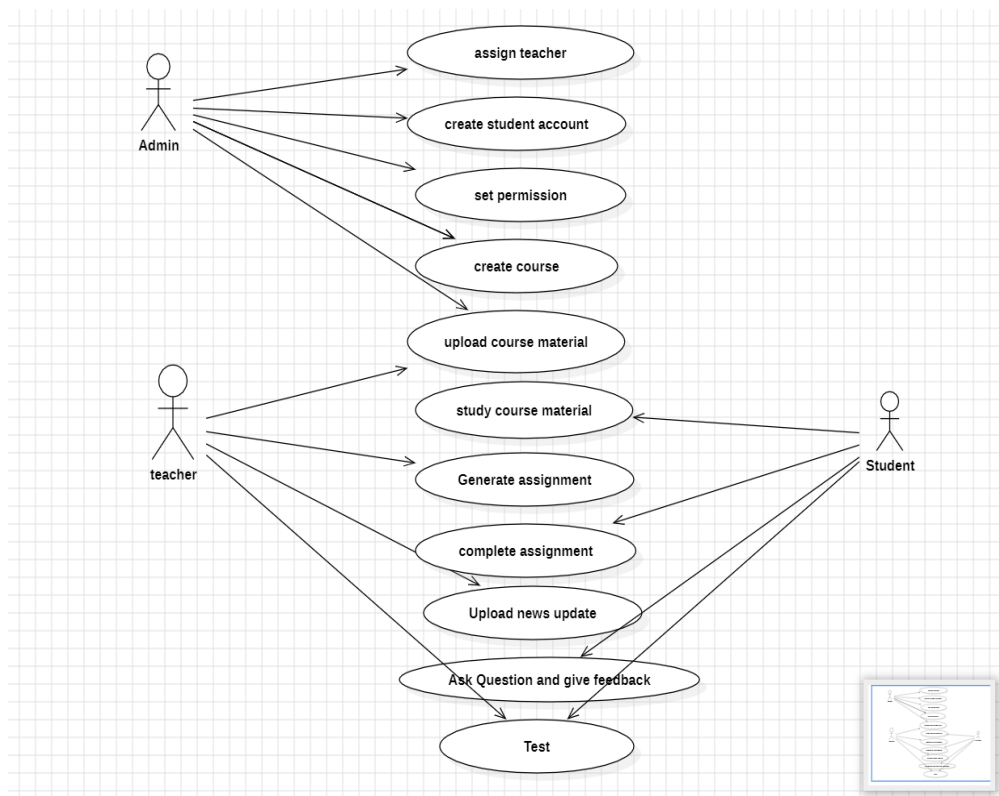


Fig.4.3 : Use Case Diagram for LMS

Student Workflow:

Students represent the largest group of users on the LMS, and therefore, their journey was designed to be straightforward, interactive, and goal-oriented. The student workflow outlines the complete path from onboarding to certification, with a strong focus on usability and learning continuity.

- **Register or Log In:** Students can either sign up as new users or log in using previously registered credentials. Upon logging in, they are redirected to a personalized dashboard tailored to their role.
- **Browse Free/Paid Courses:** Students can explore available courses using filters such as subject, category, or cost. Course previews and descriptions help them make informed decisions.
- **Enroll in a Course:** Free courses allow instant access, while paid courses are linked with Razorpay or Stripe for payment. After successful enrollment, the course becomes active in the student's dashboard.
- **Access Learning Content in Sequence:** Course material is structured into modules or chapters. Students follow a logical sequence where new lessons unlock only after previous modules are completed.
- **Attempt Quizzes and Submit Assignments:** Students complete assessments built into the course flow. Quizzes are scored automatically, while assignments are submitted for teacher evaluation.
- **Monitor Progress via Dashboard:** A progress tracker displays course completion percentage, grades, and feedback. This helps students understand their current standing and motivates continued learning.
- **Receive and Download Certificate:** Upon course completion and successful assessment, students are awarded a certificate that can be downloaded in PDF format and shared.

Teacher Workflow:

Teachers are responsible for course content creation, learner evaluation, and engagement management. The teacher workflow focuses on content management and

student performance tracking, ensuring instructors have the tools they need to effectively teach in a digital environment.

- **Log In to Instructor Dashboard:** Teachers use a secure login system and are directed to a dedicated instructor panel showing all their assigned courses and student activity metrics.
- **Upload Course Material (Video, PDF, Links):** Teachers can upload educational resources and organize them into topics or weeks. They can update content at any time to reflect new materials.
- **Create and Schedule Quizzes/Assignments:** Teachers build assessments using a variety of question types. Deadlines, time limits, and attempt restrictions can be configured per quiz.
- **Monitor Student Submissions:** All quiz and assignment submissions are listed in a central grading panel. Teachers receive real-time updates when new work is submitted.
- **Grade Responses and Give Feedback:** Assignments are reviewed manually, and teachers assign marks along with personalized comments. Feedback is sent directly to the student's dashboard.
- **Track Student Analytics:** Teachers have access to performance data, including class averages, individual scores, and time spent per module. This helps identify struggling learners and informs course adjustments.

Admin Workflow:

Administrators are the backbone of the platform, overseeing technical, academic, and financial operations. Their workflow is centered on system management, user role assignment, and analytics tracking to ensure smooth platform operations.

- **Access Admin Panel:** Admins log in through a secure portal and access a centralized control panel with system-wide visibility.
- **Create/Edit/Delete Courses:** Admins can create new course shells, assign categories, and manage metadata. They can also remove inactive or outdated content.

- **Assign Teachers to Courses:** Once a course is created, admins designate the instructors responsible for managing it. Teachers are granted access only to the courses they are assigned.
- **Approve Users and Manage Roles:** Admins review new user registrations, upgrade or downgrade user roles (e.g., from student to teacher), and manage profile data as needed.
- **Configure Payment Settings and Monitor Transactions:** Admins handle integration with payment gateways and monitor financial logs for each course purchase. Payment status and receipts are accessible from the panel.
- **View Platform Usage Analytics:** From user login statistics to course enrollments and certification rates, admins have access to downloadable reports that help inform decisions related to content, infrastructure, and platform expansion.

4.5 Tool and Plugin Selection:

To support a full-featured LMS, we explored and selected a variety of tools and Moodle plugins that matched our project goals. The selection was based on stability, ease of use, documentation, and compatibility with our tech stack.

- **Development Tools:** Visual Studio Code (IDE), Git (version control), XAMPP (local testing environment), phpMyAdmin (database GUI)
- **Moodle Plugins:**
 - Certificate Plugin (for auto-generating course completion certificates)
 - Zoom (for live class scheduling)
 - Quiz Enhancer Plugin (for additional quiz types)

This toolset allowed us to build features rapidly, reduce manual work, and ensure our LMS was extensible for future upgrades.

4.6 Database Schema Planning:

Database design is central to LMS functionality. We built a normalized relational schema using MySQL, focusing on performance, integrity, and scalability. Major tables included:

- users (user_id, name, email, role_id)
- courses (course_id, title, description, type)
- enrollments (user_id, course_id, status)
- quizzes, assignments, submissions
- certificates, transactions

We used foreign keys to maintain data consistency and added indexes for commonly queried fields. This structure made querying data efficient and protected against data redundancy.

4.7 Security Planning:

Security was integrated into the design from the very beginning. Since LMS platforms handle sensitive data like grades and payments, we implemented multiple safety layers:

- **Authentication:** Login sessions managed using PHP sessions and role-based validation.
- **Authorization:** Page access restricted based on role (student, teacher, admin).
- **Data Protection:** Input sanitization, password hashing, and prevention of SQL injection using prepared statements.
- **Session Management:** Auto-logout after inactivity and logout redirection for safety.

We ensured that each module followed best security practices and that new roles or features could easily inherit the existing security framework.

RESULTS AND OUTCOMES

The successful execution of the Learning Management System (LMS) project was the result of collaborative planning, development, and continuous iteration by the internship team. During the final phase of the internship, we achieved several concrete deliverables that reflect our hard work and learning. This chapter details the key outcomes of the project and highlights the specific contributions made by each team member to different functional components of the LMS.

The platform evolved into a fully operational, multi-role system that included secure authentication, content management, personalized dashboards, interactive learning tools, and administrative oversight. Screenshots of the final implementation have been provided to illustrate the visual and functional aspects of the system. These outcomes not only validate the project goals but also serve as practical evidence of the technical skills developed during the internship.

5.1 Member-wise Contributions :

To manage responsibilities efficiently, our team distributed major tasks among the three members based on areas of interest, skillsets, and module dependencies. Below is a comprehensive account of each member's role and output:

5.1.1 Mrunmayee Surate – Dashboard Development and Profile Management:

Mrunmayee led the design and development of personalized dashboards tailored for each user role—students, teachers, and administrators. These dashboards provided users with real-time access to their most relevant tools and data. For instance, students could immediately access their enrolled courses, track quiz progress, and view announcements. Teachers were provided with quick links to manage course materials, grade assignments, and monitor class performance, while admins could view system stats, user counts, and revenue analytics.

Her work ensured that the dashboard dynamically rendered based on login role, and used clean, responsive layouts that improved user navigation. Each dashboard interface was designed to promote clarity and streamline the learning or administrative process.

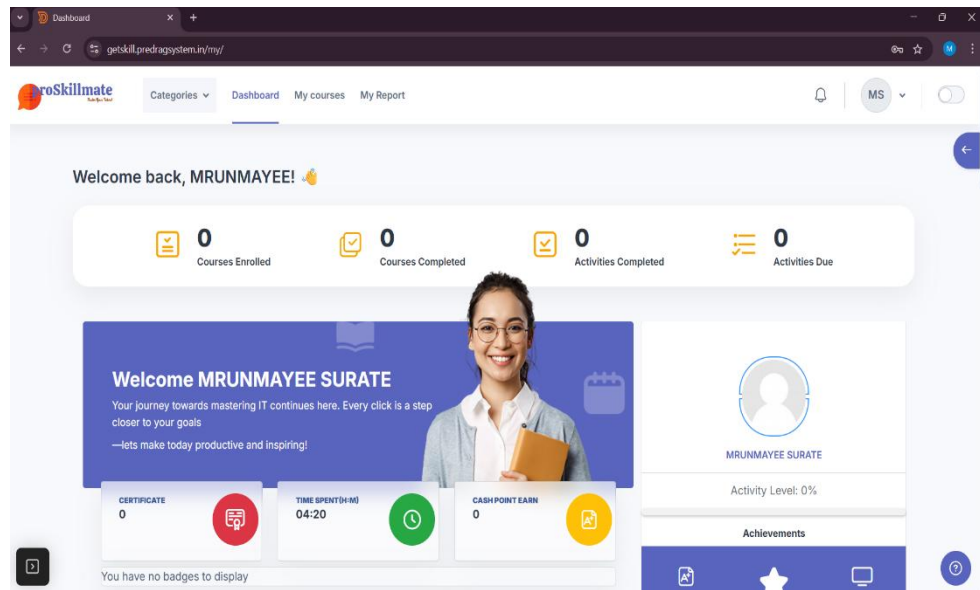


Fig.5.1 : Personalized Dashboard

Figure 5.1 displays the personalized dashboard interface of the LMS, tailored for each user role—student, teacher, and admin. It provides quick access to relevant features like enrolled courses, progress tracking, announcements, and profile management, all in one convenient view.

In addition, Mrunmayee developed the user profile editing module. This feature allowed users to update their name, email, contact number, and upload profile images. Changes were reflected in the database in real time, with form validation ensuring clean and secure input. This feature improved user personalization and gave every participant control over their own information.

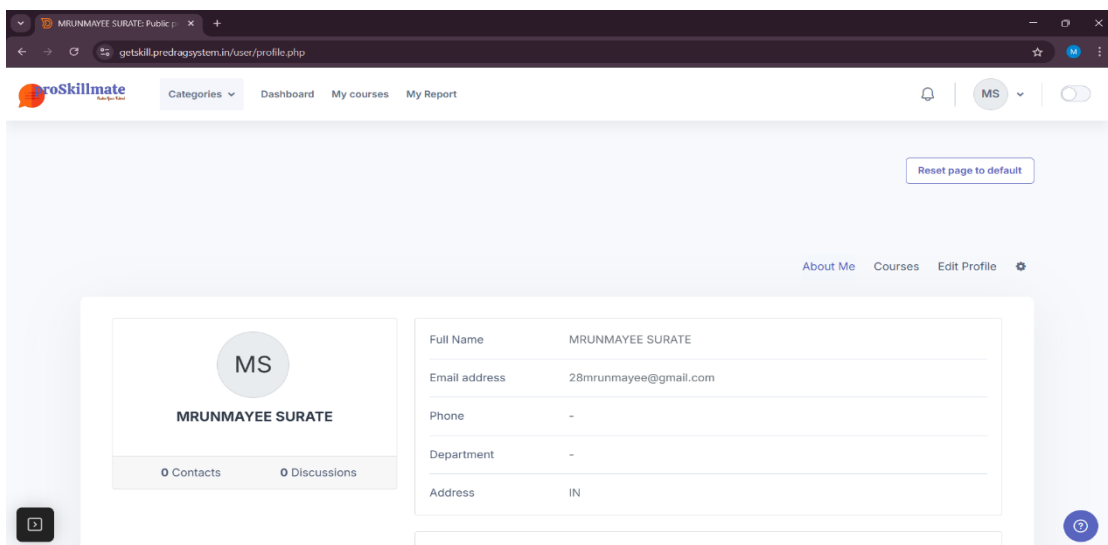


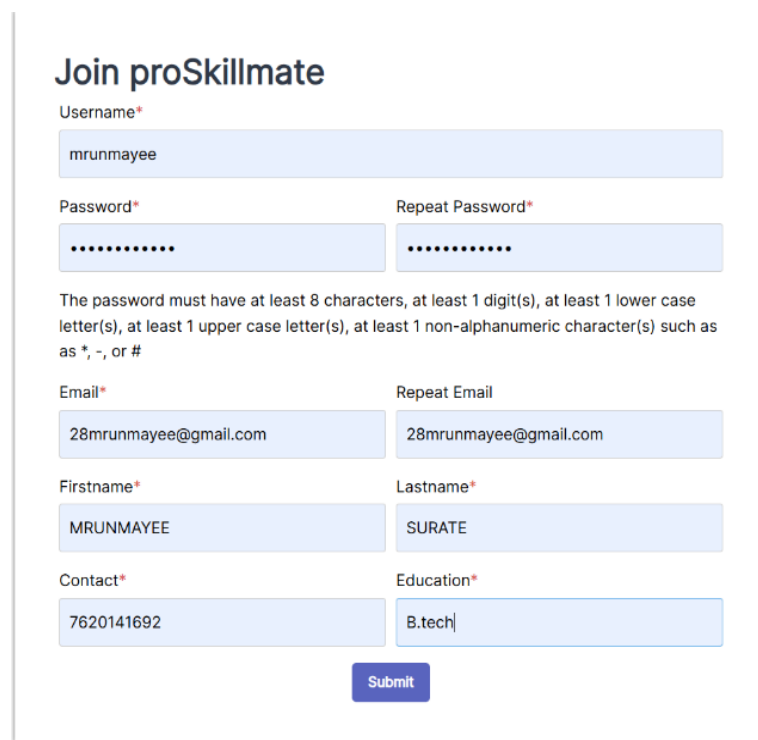
Fig.5.2 : User profile editing

Figure 5.2 illustrates the user profile editing feature of the LMS, allowing users to update their personal information such as name, email, contact number, and profile picture. This module ensures a personalized experience and supports real-time data updates within a secure and user-friendly interface.

She also played a key role in integrating the user experience across different screens and ensuring consistency in theme, layout, and access controls. Her contributions helped unify the interface across all roles, reinforcing usability and navigational flow.

5.1.2 Yogesh Birajdar – User Registration and Login System:

Yogesh was responsible for implementing the user authentication system, including account creation (registration) and login functionality. The system supported role-based login, ensuring that each user (student, teacher, admin) was directed to the correct dashboard upon logging in.



The screenshot shows a registration form titled "Join proSkillmate". The form includes the following fields and labels:

- Username***: Input field containing "mrunmayee".
- Password***: Input field with masked characters ".....".
- Repeat Password***: Input field with masked characters ".....".
- Email***: Input field containing "28mrunmayee@gmail.com".
- Repeat Email**: Input field containing "28mrunmayee@gmail.com".
- Firstname***: Input field containing "MRUNMAYEE".
- Lastname***: Input field containing "SURATE".
- Contact***: Input field containing "7620141692".
- Education***: Input field containing "B.tech".

Below the fields is a blue "Submit" button. A password requirement note is displayed: "The password must have at least 8 characters, at least 1 digit(s), at least 1 lower case letter(s), at least 1 upper case letter(s), at least 1 non-alphanumeric character(s) such as as *, -, or #".

Fig. 5.3 : Account Creation and Login

Figure 5.3 shows the account creation and login interface of the LMS, where users can register with their details or securely log in using existing credentials. This feature ensures role-based access and is the first step in providing a personalized learning experience. He designed the registration form with robust field validation to

avoid duplicate or malformed entries and linked it with the database for permanent storage. For the login process, he used secure PHP sessions to authenticate users and prevent unauthorized access. The login interface included helpful error prompts, a “remember me” function, and proper redirection logic.

Furthermore, logout and session timeout functionality were also handled to enhance security. By focusing on these foundational modules, Yogesh ensured that the LMS was secure and accessible to legitimate users at all times.

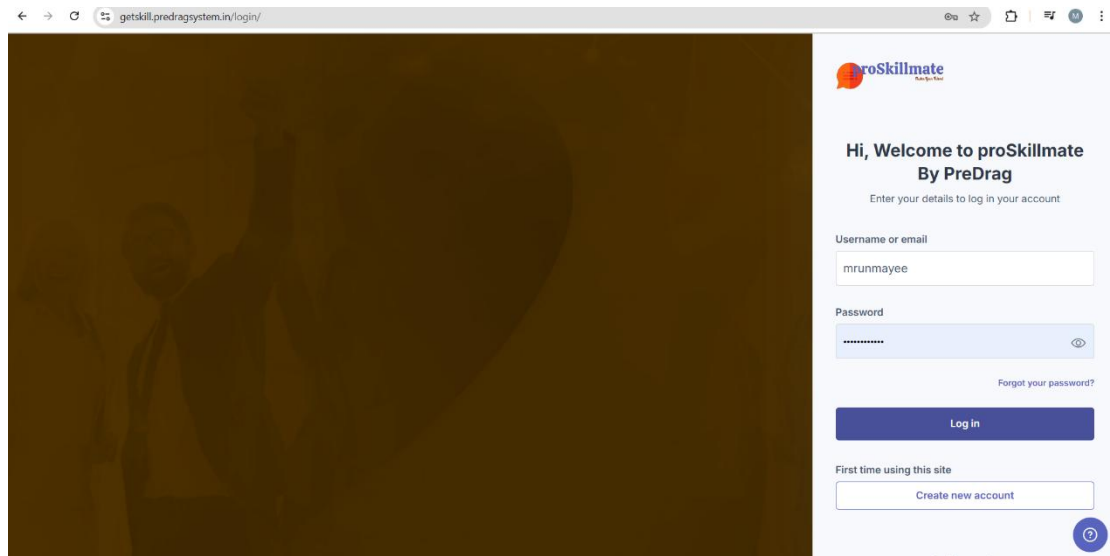


Fig. 5.4: Login page

Figure 5.4 displays the login page of the LMS, designed for easy and secure access by students, teachers, and admins. It includes role-based redirection, input validation, and user-friendly prompts to guide users through the login process smoothly. In addition to login security, he implemented server-side checks for session integrity, preventing users from accessing restricted modules outside their roles. Yogesh also worked on integrating login state into navigation headers, so each user could easily access the right links.

5.1.1 Narayan Kadam – My Reports and Course Module Pages:

Narayan took ownership of the “My Reports” and “My Courses” sections—two essential modules for learners and teachers alike. The My Reports page enabled students to view their academic progress, including quiz results, assignment scores, and completion percentages. The data displayed was dynamically retrieved from the

database and structured for readability through tables and colored progress indicators.

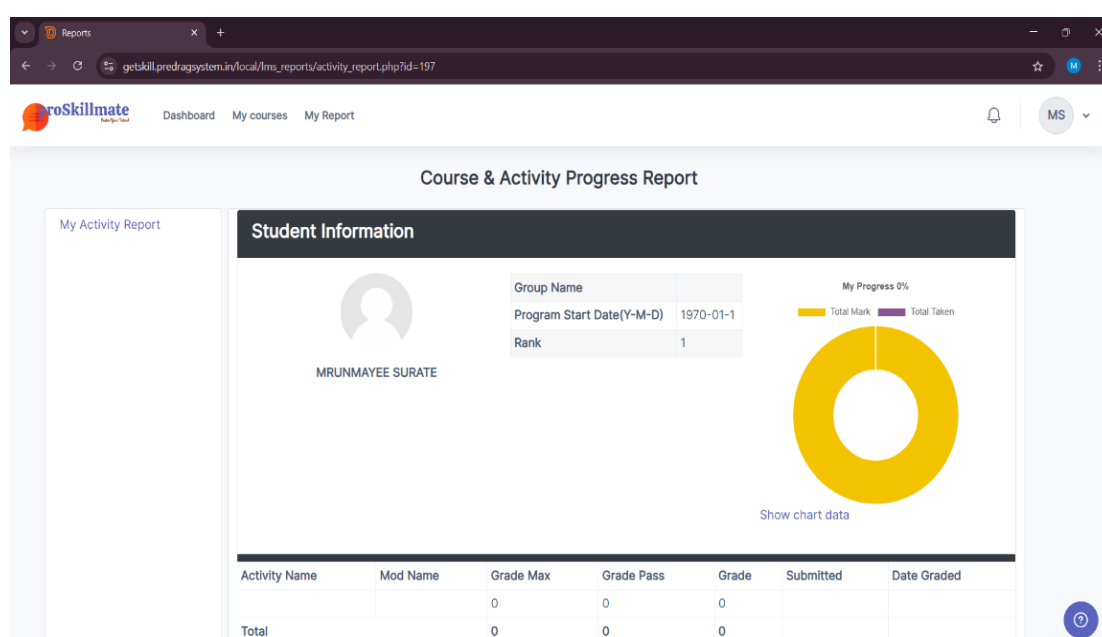


Fig. 5.5: Course and Progress Report

He also developed the course details page, where users could view course objectives, lesson sequences, and required assignments. Students could open and study videos, PDFs, and other learning content, all organized in a clear sequence. Narayan implemented logical conditions to show what parts of the course were accessible based on completion status or quiz performance.

Figure 5.5 presents the Course and Progress Report section of the LMS, where students can view their enrolled courses along with detailed progress metrics. It helps learners track their completion status, quiz scores, and overall performance throughout the course.

His modules contributed significantly to the learning experience by offering transparency, feedback, and a centralized view of academic activity. These pages also included certificate status, allowing students to see whether they were eligible to download their completion certificate.

Figure 5.6 showcases the variety of courses available within the LMS, allowing users to browse through categorized free and paid offerings. Each course listing provides essential details like title, instructor, duration, and enrollment options for a seamless learning experience.

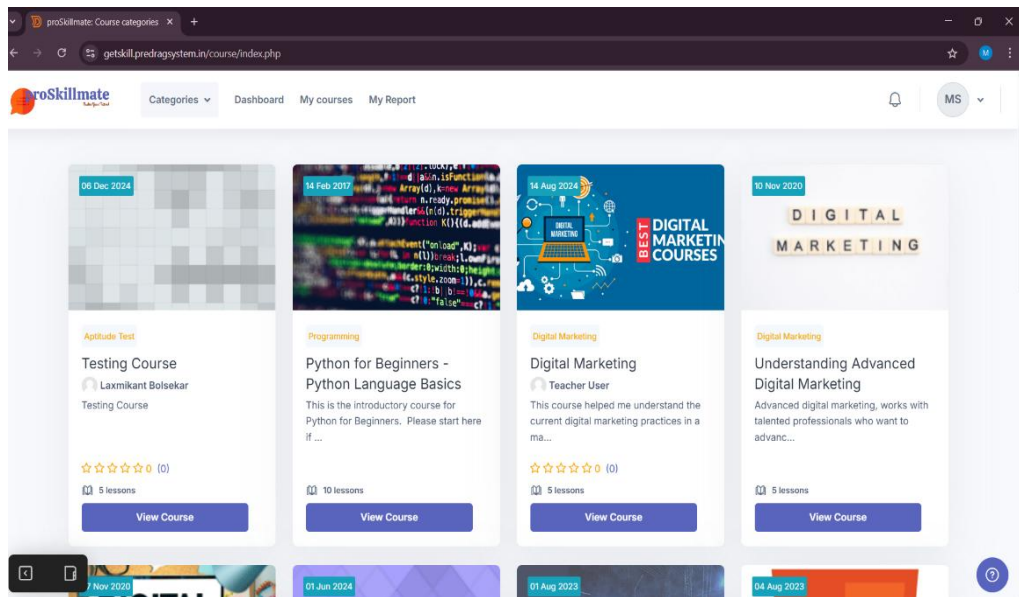


Fig. 5.6: Different Courses in LMS

Narayan also supported integration of backend logic that calculated course progress percentages and dynamically updated reports for instructors and learners. This ensured real-time feedback and helped enhance student motivation and accountability.

5.2 Feature Integration Snapshots:

To validate the completeness of our LMS project, we captured screenshots of all major modules and interface components. These screenshots serve as visual documentation and evidence of the system's practical functionality and technical robustness. Each subpoint below highlights the screenshots and explains the integration of front-end design with back-end logic that collectively shaped the working platform.

5.2.1 User Registration and Login Interface: The login and registration system is the first point of interaction for users. Our screenshot shows the registration form that includes real-time field validation such as proper email formatting and secure password input. Upon successful registration, the user data is stored in the MySQL database with encryption. The login screen features session handling using PHP and redirects each user to their respective dashboard based on their role. We also implemented error messaging and session timeout functionality, which are critical for system security.

5.2.2 Personalized Dashboard Based on Role: The dashboards are role-specific and dynamically generated. For students, the dashboard displays active courses, progress trackers, certificates, and announcements. Teachers are presented with course creation

tools, grading panels, and live class management options. Admins access platform analytics, manage user roles, and oversee course approvals. Screenshots of each dashboard show how each user role is greeted with a unique layout and relevant tools. All data is pulled in real-time using server queries, offering seamless navigation and consistent UI across roles.

5.2.3 Profile Edit Form with Live Update: The profile edit form allows users to manage their personal information such as name, email, password, and profile image. Screenshots reveal the clean and accessible design, along with backend integration for immediate database updates. The form performs both client-side and server-side validation to ensure accuracy and security. AJAX was used to make the updates happen without reloading the page, making the experience smoother. Users receive a confirmation message once their changes are successfully applied, enhancing usability.

5.2.4 Course List and Browsing Interface: Screenshots of the course list module illustrate how users browse and filter available courses. Each course card displays the title, instructor, duration, and pricing (free or paid). Clicking on a course opens a detail view where students can see module lists, previews, learning outcomes, and pricing options. These pages were built to dynamically fetch data from the database and adjust the UI based on availability and enrollment status. The responsiveness of this section ensures a smooth experience across different devices and screen sizes.

5.2.5 Course Detail and Enrollment Process: Once students select a course, the course detail page guides them through the enrollment process. Screenshots highlight how free courses allow direct access, while paid courses are integrated with Razorpay/Stripe for seamless payment. After payment verification, the user is enrolled automatically. The page includes options to preview lessons, track module completion, and download course resources. This section bridges content access with back-end control, ensuring users engage only with the content they are eligible for.

5.2.6 Quiz, Assignment, and Module Navigation: Screenshots of this module demonstrate how students interact with assessments. Each course module includes embedded quizzes and downloadable assignments. Quizzes are auto-evaluated, and assignment uploads are routed to teacher dashboards. Teachers then grade submissions and send feedback. Navigation between modules is managed via completion

checkpoints, where new modules unlock only after completing the previous ones. This ensures logical progression and prevents learners from skipping essential content.

5.2.7 My Reports Page with Performance Metrics: The My Reports section helps users monitor their academic journey. Our screenshots display how students can view their quiz scores, assignment grades, course progress, and certificate status. This data is pulled from multiple database tables and displayed in organized, user-friendly formats like graphs and tables. Color-coded statuses (e.g., Completed, Pending, In Progress) help learners quickly interpret their standing. Teachers also access similar reports to view class-wide performance and identify areas needing attention.

5.2.8 Admin Control and Management Screens: Finally, we documented the administrative control panel. Screenshots show how admins manage users, approve or delete courses, configure payment settings, and monitor overall system health. The dashboard displays real-time statistics such as revenue, course popularity, user registrations, and session activity. Admins can also generate downloadable reports for analysis. These controls ensure the platform remains scalable, secure, and functional. The screenshots validate that core administrative features are fully operational and accessible through a centralized interface.

5.3 Additional Achievements

Beyond the core functionality of our Learning Management System (LMS), our development team also achieved several technical milestones and quality improvements during the course of the internship. These additional efforts contributed to system reliability, scalability, and long-term deployment readiness. Below are detailed descriptions of key non-functional but critical accomplishments that enhanced the robustness of the project.

5.3.1 Manual Testing with Multiple User Accounts: To ensure system reliability under real-world conditions, we performed manual testing with over 20 different user accounts across student, teacher, and admin roles. Each account was used to validate role-specific permissions, content access restrictions, navigation paths, and overall user experience. We deliberately attempted invalid actions such as unauthorized page access and incorrect form inputs to verify that the system correctly handled errors without

crashing or exposing data. This round of testing helped us identify and fix several minor UI inconsistencies and logic bugs, resulting in a much more stable and secure platform.

5.3.2 Git-Based Code Versioning and Collaboration: Our entire project was maintained using Git-based version control, hosted on GitHub. This allowed all three team members to work on different parts of the project simultaneously while avoiding code conflicts. Every feature addition or bug fix was done in separate branches, reviewed, and then merged into the main development branch after testing. We followed consistent commit messages and documented our codebase properly to ensure traceability and collaboration efficiency. Using Git also allowed us to revert to earlier versions in case of unexpected bugs, making the development process safer and more manageable.

5.3.3 Backup-Ready Database Structure: Understanding the importance of data integrity and disaster recovery, we built a backup-ready database architecture. All database tables, including users, courses, transactions, quizzes, and submissions, were normalized and exported as .sql backups. These backups can be easily imported into any MySQL-compatible server to restore the LMS within minutes. We also tested these backups on different machines and confirmed successful deployment. This achievement ensures that our system is disaster-resilient and migration-friendly, which is critical for real-world hosting scenarios and version upgrades.

5.3.4 Cross-Browser and Responsive Testing: We conducted UI and functionality tests across multiple browsers including Chrome, Firefox, Microsoft Edge, and Safari to ensure platform compatibility. Furthermore, responsive design was validated on different screen sizes—mobile, tablet, and desktop—using Chrome Developer Tools and live devices. Layouts, buttons, text spacing, and input forms were adjusted to maintain consistency. Special attention was given to mobile views, ensuring that dropdowns, navigation bars, and forms were fully accessible without horizontal scrolling. These efforts make the LMS inclusive and usable by a wider range of users with varying devices.

5.3.5 Planned Integration of Payment and Live Session Modules: Although core functionality was successfully implemented, we also invested time in planning future-ready modules such as payment gateway integration and live session hosting. We documented setup procedures for Razorpay and Stripe APIs, including credential

management and callback URLs. For live sessions, we prepared guidelines for integrating Zoom and BigBlueButton (BBB) using Moodle plugins and custom PHP hooks. This proactive planning ensures that once the system is deployed or expanded, these advanced features can be added with minimal development overhead. Their architecture has already been partially scaffolded into the admin and teacher panels.

5.4 Overall Outcomes:

The outcomes of our Learning Management System (LMS) project go far beyond the boundaries of technical execution. While building the platform, our team gained deep insight into the full software development life cycle—from planning and designing to developing, testing, and refining. Each milestone contributed to the enhancement of both individual and collective learning, aligned with the primary objectives of the internship. This section documents the key outcomes that resulted from the collaborative development process.

5.4.1 Functional Product Delivery: The most tangible outcome of our internship is the successful delivery of a fully functional LMS. The platform supports real-time interaction, secure user authentication, modular course structures, and interactive assessments. Each feature was rigorously tested to ensure it worked under real-world conditions. The LMS accommodates all three roles—students, teachers, and administrators—through dedicated dashboards and access controls. Whether it's enrollment in a paid course or real-time progress tracking, each use case was covered with both front-end and back-end implementation, making the LMS deployment-ready.

5.4.2 Practical Skill Development: One of the most significant takeaways from the project was the hands-on technical experience we gained. Working with technologies such as PHP, MySQL, HTML, CSS, JavaScript, and Moodle helped us apply what we learned in classrooms to real development challenges. We handled both front-end and server-side scripting, explored database queries and normalization, and wrote secure authentication flows. Additionally, the use of AJAX, form validation, and data binding enriched our understanding of interactive web development. This practical exposure deepened our technical proficiency.

5.4.3 Team Collaboration and Version Control: Throughout the internship, our team practiced real-world software collaboration using Git and GitHub. Each member

worked on separate modules using branches, and we coordinated updates through commits, pull requests, and merge operations. This ensured smooth teamwork without overwriting code or creating conflicts. We followed naming conventions, used consistent file structures, and documented changes to ensure everyone stayed aligned. This practice helped us simulate how professional development teams operate in an Agile environment, enhancing our team coordination skills.

5.4.4 Problem-Solving and Debugging Skills: Software development is inherently iterative, and many bugs and logical errors emerged during the project. We resolved real-time issues like session expiration, access control failures, data inconsistencies, and input validation loopholes. Each issue was analyzed, debugged, and resolved methodically. We used browser developer tools, PHP logs, and SQL testing tools to identify problems. This hands-on debugging experience helped sharpen our problem-solving mindset and prepared us for handling complex system behaviors in future projects.

5.4.5 Industry-Level Exposure and Readiness: From coding standards to UI responsiveness, we approached this project like a real-world software product. We made our LMS responsive across devices, optimized database queries for faster page loads, and documented every function for scalability. Working with third-party tools such as GitHub, Moodle plugins, Zoom/BBB APIs, and Razorpay payment integrations gave us a taste of industry practices. We learned how to think beyond development—about user experience, system stability, and deployment requirements. These insights strengthened our confidence to contribute to professional software projects in the future.

5.4.6 Documentation and Planning Practice: Beyond the codebase, we also focused heavily on documentation. Each module was documented through structured write-ups, screenshots, and diagrams. Flowcharts, ER diagrams, and use case diagrams were used to plan the system's architecture before writing a single line of code. Screenshots were taken to validate progress and support final reporting. This habit of organized documentation will be a long-term asset in our academic and professional pursuits, especially as future developers can refer to our work for continuity or improvement.

5.4.7 Application of Academic Knowledge: This internship bridged the gap between theoretical learning and its practical application. Concepts from subjects like Database

Management Systems (DBMS), Web Technology, and Software Engineering were used throughout the project. From database schema design to secure web application structure and modular software architecture, every theoretical principle we studied in class came into play. This real-world application helped reinforce those academic concepts in a way that lectures and books alone could not.

CONCLUSION

This four-month internship at Vigyaan SoftTech Solutions was an invaluable experience that transformed our academic understanding into real-world software development skills. The core project assigned to us—building a feature-rich Learning Management System (LMS)—allowed us to explore the complete lifecycle of a scalable web application. From designing modular architecture and managing backend logic with PHP and MySQL to integrating Moodle, user-role access, and learning tools like quizzes, assignments, and certificates, we worked hands-on with technologies and concepts we had previously studied only in theory. We also implemented monetization features including secure payment gateways and pricing modules, extending the system's relevance to both academic and commercial settings.

This internship sharpened not only our technical capabilities but also our teamwork, communication, and problem-solving skills. Collaborating as a team of three, we divided responsibilities, resolved bugs together, and maintained our progress using version control tools. The LMS we developed is a comprehensive, responsive, and deployment-ready digital learning ecosystem that reflects our dedication and growth. Most importantly, this journey prepared us to contribute confidently to professional software projects in the future. The successful implementation of this system stands as a testament to the knowledge we gained and the practical skills we developed during this rewarding internship.

REFERENCES

- [1] Brown, A. (2021) *Building Scalable Learning Management Systems: A Practical Guide to Moodle-Based Platforms*. EduTech Press. ISBN: 978-1-234567-88-8
- [2] Sharma, R. (2020) *E-Learning System Design and Architecture: From Concept to Implementation*. LearningBridge Publishers. ISBN: 978-1-345678-90-1
- Nguyen, L., and Thomas, M. (2019) *Open Source LMS Development with PHP and MySQL*. TechEd Solutions. ISBN: 978-1-876543-21-5
- [3] Singh, P., and Kaur, S. (2022) *Moodle for Educators and Developers: Creating Interactive Online Courses*. EduWorks Press. ISBN: 978-1-456789-65-4
- Ahmed, Z. (2023) *Designing Role-Based Dashboards in LMS Platforms*. Digital Learning Lab. ISBN: 978-1-236547-94-0
- [4] Lee, J., and Patel, D. (2021) *Learning Management Systems: Integration of Assessment and Certification Tools*. Global EdTech Publishers. ISBN: 978-1-234568-76-3
- [5] Martínez, E., and Roy, T. (2020) *UX/UI for E-Learning: Interface Design in Learning Management Systems*. EdTech Studios. ISBN: 978-1-894567-42-7
- [6] Kumar, A., and Desai, N. (2023) *Online Education Infrastructure: Payment Gateways, Cloud Hosting, and LMS Monetization*. Smart Learn Tech. ISBN: 978-1-763849-50-2