**LEARNING MANAGEMENT SYSTEM**

Major Course Output

College of Computing and Information Sciences

NORHTWEST SAMAR STATE UNIVERSITY

Calbayog City

In Partial Fulfillment

of the Requirements for the Subject

APPLICATION DEVELOPMENT AND EMERGING TECHNOLOGY

By

Jayson F. Mancol  
Aaliyah Bianca S. Pallones

March 2025

**CHAPTER 2**

**TECHNICAL BACKGROUND**

Brief introduction ikaw nala didi

**Software Requirements**

Brief introduction ikaw nala didi

Table 1

Software Requirements for the Development of Learning Management System

|  |  |  |
| --- | --- | --- |
| Software | Specifications | Descriptions |
| Markup Language | HTML | Used to structure web content |
| Style Sheet Language | CSS | Used for designing and styling web pages |
| Programming Language | JavaScript | Adds interactivity and logic to web applications |
| Runtime Environment | Node.js | Executes JavaScript on the server-side |
| Web Framework | Express.js | A Node.js framework for building web applications |
| Database Management System | MySQL | Stores and manages application data |

The table outlines key software requirements for developing a **Learning Management System (LMS)**. **HTML** structures the web content, while **CSS** styles and enhances its visual appeal. **JavaScript** adds interactivity, making the system dynamic. **Node.js**, as a runtime environment, allows JavaScript to run on the server side, ensuring efficient request handling. **Express.js** simplifies backend development by managing routes and APIs. Lastly, **MySQL** stores and manages user data, course content, and progress tracking.

**Hardware Requirements**

Brief introduction ikaw nala didi

Table 2

Hardware Requirements for the Development of Learning Management System

|  |  |  |
| --- | --- | --- |
| Hardware | Specifications | Descriptions |
| Laptop/Desktop | Intel i5/i7 or AMD Ryzen 5/7, 64-bit OS | Main device for coding and testing |
| Keyboard | Mechanical or ergonomic keyboard | Enhances typing speed and comfort for coding |
| Mouse | Optical or wireless with high precision | Ensures smooth navigation and better control |
| Monitor | 24” Full HD (1920x1080) or higher | Improves visibility and multitasking |
| Internet Connection | Broadband, 10 Mbps+ | Ensures smooth access to online tools, cloud services and tisting |

The table above outlines the essential **hardware requirements** for developing a **Learning Management System (LMS)**. Each component plays a crucial role in ensuring a **smooth, efficient, and productive** development process. A **laptop or desktop** with a fast **processor**  ensures that development tools and servers run efficiently. **A keyboard** enhances comfort during long coding sessions, while a **mouse** ensures easy navigation. A **monitor** improves visibility and multitasking, allowing developers to work on multiple screens effectively. **Stable internet** ensures seamless access to online resources, cloud services, and testing environments.

**Gantt Chart**

Brief introduction ikaw nala didi

Table 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Phases | Month | | | |
| February | March | April | May |
| Planning & Research |  |  |  |  |
| UI/UX Design |  |  |  |  |
| Frontend Development |  |  |  |  |
| Backend Development |  |  |  |  |
| Testing & Debugging |  |  |  |  |
| Final Deployment |  |  |  |  |

The **table above** represents the actual activities completed during different phases of website development. The development process started in **February** with **planning and research**. Similar systems were studied to understand their features and development methods, helping to create a clear plan for building the system. In **March**, the **UI/UX design** phase began, focusing on making the system easy to use. New features were added to improve the user experience. At the same time, **frontend development** started using **HTML, CSS, and JavaScript** and continued into **April**. By **April**, work shifted to **backend development**, using **Node.js** for server-side functions, **MySQL** for storing data, and **Express.js** for managing the backend. This phase focused on making the system fully functional and allowed time for **testing and fixing errors**. Several tests were done to find and fix **bugs and performance issues** to make sure the system ran smoothly. In **May**, the system was made **ready for launch**. Final checks and tests were done to make sure everything worked correctly before making the system available to users.

**CHAPTER 3**

**METHODOLOGY, RESULTS AND DISCUSSION**

Introductory Paragraph of the chapter and includes discussion of the methodology used. **Ikaw didi**

**Requirements Analysis**

* Fact-findings Methods (Interviews, Survey Questionnaire, Observation, etc.) **ikaw didi**

**Requirements Documentation**

**Functional Requirements**

Table 1: Functional Requirements for Learning Management System

|  |  |
| --- | --- |
| FUNCTIONALITIES | DESCRIPTION |
| User Management | Allows users to register, log in, and manage their profiles. |
| Section Management | The teacher can create and manage multiple sections under their subject. Students can only enroll in the section assigned to them. |
| Student List Management | The teacher can view a list of enrolled students in each section. This includes student number, student name.Teachers can also search, remove or update student information if needed. |
| Quiz Management | The teacher can create quizzes. Students can take quizzes, and receive scores. |
| Activities Management | The teacher can upload lesson materials, PDFs, videos, documents and submit assignments. Only enrolled students can access these materials. |
| Announcements | The teacher can post updates, deadlines, and important notices for all sections through notifacation via email. |
| Authentication & Security | Implements login/logout features, password encryption, and session management to ensure data security. |
| Responsive Design | The LMS is accessible on desktops, tablets, and mobile devices. |

Table above shows key **functional requirements** for the Learning Management System (LMS) to ensure smooth user interactions and system efficiency. **User Management** allows users to register, log in, and manage their profiles, while **Section Management** lets teachers create sections and assign students accordingly. **Student List Management** provides teachers with access to enrolled students' details for updates and modifications. **Activities Management** enables teachers to upload learning materials like PDFs, Assignment and videos for student access. **Quiz Management** supports quiz creation, allowing students to participate and receive scores. **Announcements** ensure teachers can post important updates through notification via email to keep students informed about assignments, quizzes, deadlines, and announcements, while **Authentication & Security** maintain data protection with login/logout features and encryption. **Responsive Design** makes the system accessible across different devices.

**Non-Functional Requirements**

Table 2: Non- Functional Requirements for Learning Management System

|  |  |
| --- | --- |
| FUNCTIONALITIES | DESCRIPTION |
| Performance | The system should respond to user requests within **seconds**, ensuring a smooth experience even under high traffic. |
| Scalability | The LMS should support **thousands of concurrent users** without performance degradation. |
| Security | Implements **encryption, multi-factor authentication (MFA), and secure session management** to protect user data. |
| Availability | Ensures **99.9% uptime**, minimizing system downtime for users. |
| Usability | The interface should be **user-friendly**, accessible to users with different levels of tech proficiency, and follow **UI/UX best practices**. |
| Compatibility | The system should work on **desktops, tablets, and mobile devices** across different browsers. |
| Maintainability | The LMS should be **easy to update and debug**, allowing for quick issue resolution and feature enhancements. |
| Data Backup & Recovery | The system should perform **daily backups** to prevent data loss and support **quick recovery** in case of failure. |

Table above shows the Non-Functional Requirements we ensured the system would be **fast and responsive** by optimizing code, using fast servers, and reducing load times. To handle **thousands of users**, we planned for database optimization, and load balancing. Security measures like **password encryption, multi-factor authentication, and secure data storage** were included to protect user information. To maintain **99.9% uptime**, we set up backup servers, monitoring tools, and quick recovery systems. The interface was designed to be **simple and user-friendly**, making navigation easy for all users. We also ensured compatibility across **desktops, tablets, and mobile devices** using different web browsers. For easy maintenance, we planned for **quick updates, debugging tools, and clear documentation**. Lastly, we implemented **daily backups** and a fast recovery system to prevent data loss.

**Design of Software, Systems, Product, and/or Processes**

**System Architecture**

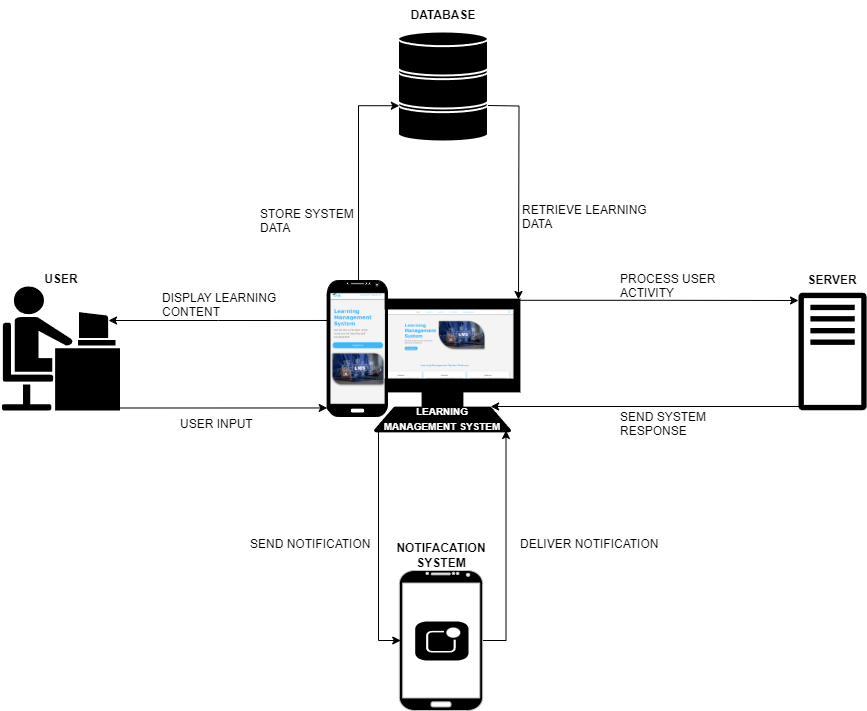
  
Figure 1. System Architecture Diagram for Learning Management System

Figure 1 above shows a Learning Management System (LMS) where users interact with learning content through a web interface. The system processes user input, retrieves and stores learning data in a MySQL database, and manages user activity through a Node.js and Express.js backend. Notifications are sent to users via a notification system. To implement this, you can develop the frontend using HTML, CSS, and JavaScript, while using Node.js and Express.js for backend operations, ensuring smooth data handling and communication with the database.

**Data Flow Diagram**

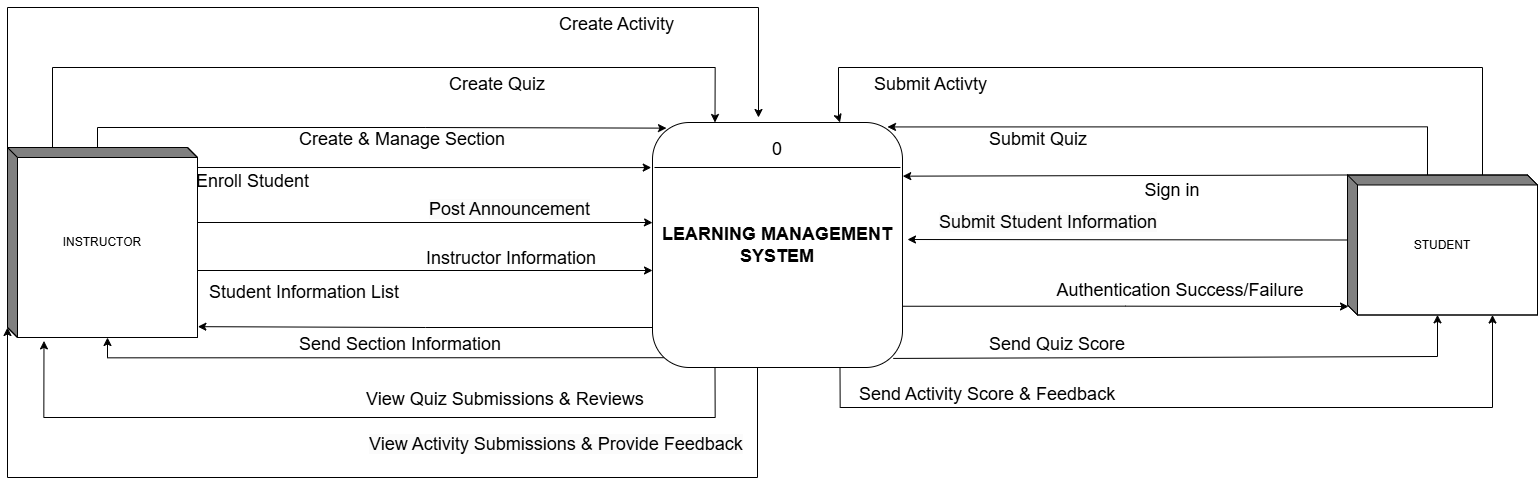
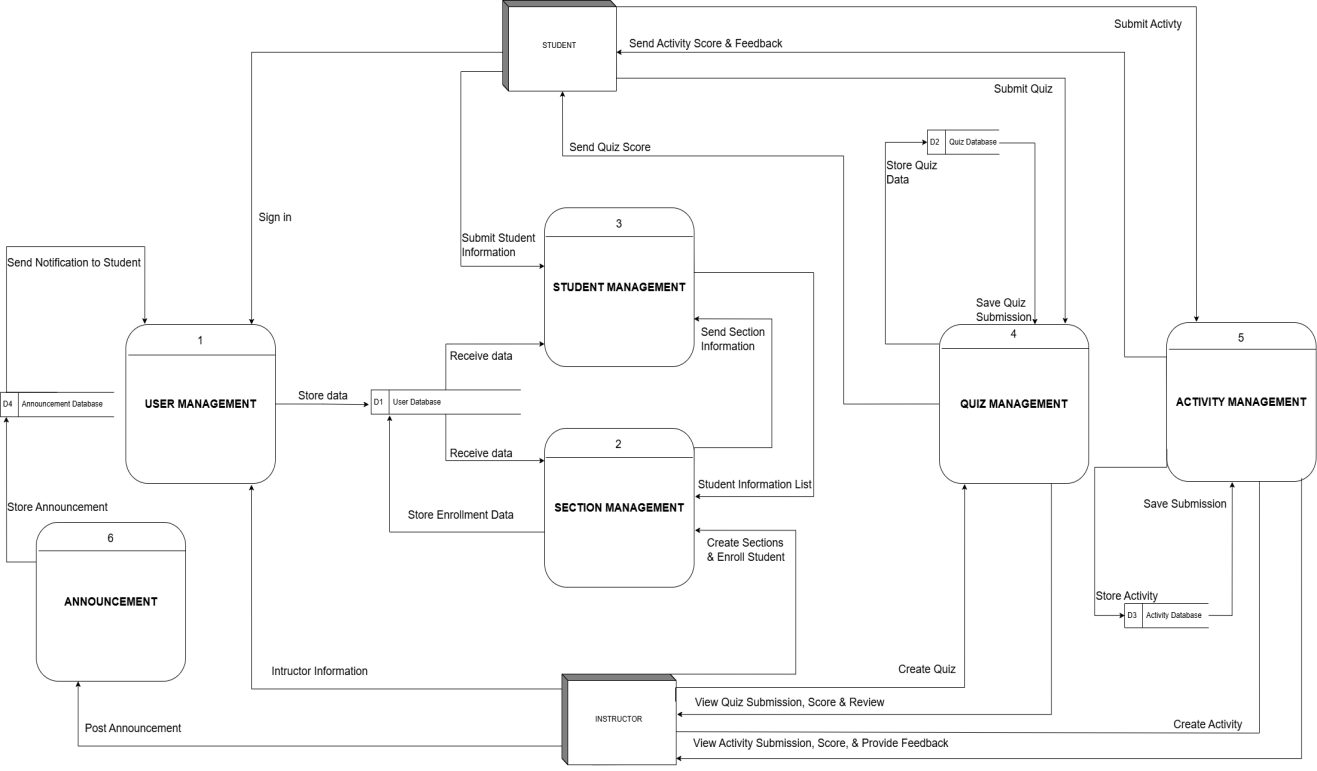
**Context Data Flow Diagram  
**

Figure . Context Data Flow Diagram for Learning Management System

**Detailed Data Flow Diagram**Figure . Detailed Data Flow Diagram for Learning Management System

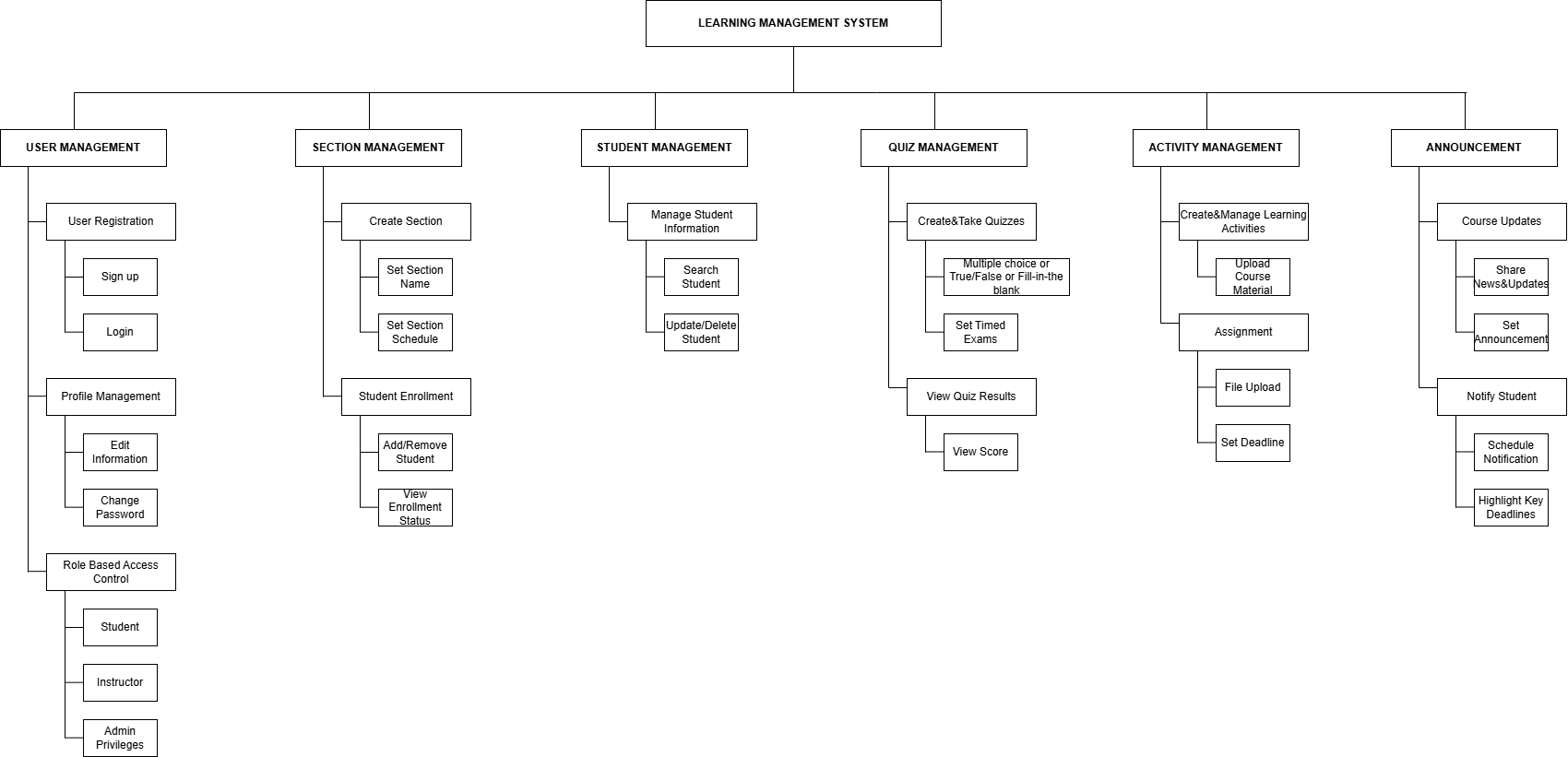
**Functional Decomposition Diagram**

Figure 5. Functional Decomposition Diagram for Learning Management System

Figure 5 above shows a the Functional Decomposition Diagram (FDD) for the Learning Management System (LMS) where the **User Management** module handles user registration, login, profile updates, and role-based access control. **Section Management** enables the creation of course sections, instructor assignments, and student enrollments. **Student List Management** allows tracking and updating student records and progress. **Activities Management** focuses on creating and assigning learning materials. The **Quiz & Assessment** module manages quiz creation, submissions, and grading. **Announcements** provide a way to post updates and notify students about important events. **Authentication & Security** ensures secure access with login authentication and data protection. **Responsive Design** adapts the LMS interface for various devices, and **Notifications** facilitate communication through email, push notifications, and in-system alerts. This structure ensures an efficient, user-friendly, and scalable learning platform.