Project Design Phase-II

Technology Stack (Architecture & Stack)

Date	03 November 2025
Team ID	NM2025TMID01909
Project Name	Medical Inventory Management
Maximum Marks	4 Marks

Technology Stack & System Architecture

System Architecture Overview

The **Medical Inventory Management System (MIMS)** follows a **Three-Tier Architecture**, separating the application into **Presentation**, **Application (Business Logic)**, and **Data** layers.

Architecture Layers

Layer	Description	Technologies Used
1. Presentation Layer (Front-End)	User interface for Admins, Pharmacists, Nurses, and Suppliers to interact with the system.	HTML5, CSS3, JavaScript, Bootstrap / ReactJS
2. Application Layer (Back-End)	Processes business logic such as inventory updates, expiry tracking, order generation, and reporting.	Node.js / Python (Flask or Django) / Java (Spring Boot)
3. Data Layer (Database)	Stores all medical stock, supplier, transaction, and user data securely.	MySQL / PostgreSQL / MongoDB

Key Architectural Features

- **Modular Design:** Each component (login, inventory, reporting) can be maintained independently.
- **Scalability:** Easily expandable for multiple hospital branches or departments.
- **Interoperability:** Can integrate with Hospital Management Systems (HMS) or billing systems via APIs.
- **Cloud-Ready:** Deployable on cloud servers for real-time multi-user access.
- **Data Security:** Implements encryption and secure authentication methods.

Detailed Technology Stack

Component	Technology Options	Purpose / Functionality
Frontend (UI)	HTML5, CSS3, Bootstrap, ReactJS	To create a responsive, easy-to-use interface for all user roles.
Backend (Server-side)	Node.js with Express / Python Flask / Django	Handles business logic, validation, and communication with the database.
Database	MySQL / PostgreSQL / MongoDB	Stores all information about medicines, users, suppliers, and transactions.
Authentication	JWT (JSON Web Token) / OAuth 2.0	Ensures secure login and role-based access control.
APIs	RESTful APIs	Allows communication between front-end and back-end services.
Notifications	Email / SMS Gateway (e.g., Twilio, SMTP)	Sends alerts for expiry, low stock, or purchase order confirmations.
Hosting / Deployment	AWS / Azure / Google Cloud / Local Server (XAMPP)	To host the web application and database for 24/7 access.
Version Control	Git / GitHub	Source code management and collaboration.
Reports / Visualization	Chart.js / Power BI / Google Charts	For displaying reports on usage, stock levels, and supplier performance.
Security	HTTPS, Data Encryption, Role-based Access Control	Protects sensitive data and ensures authorized access.

Example Technology Stack Combination

Layer	Chosen Stack Example
Frontend	ReactJS + Bootstrap
Backend	Node.js (Express.js Framework)
Database	MySQL
Authentication	JWT (JSON Web Token)
Hosting	AWS EC2 / Render / Localhost (for testing)