**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

**Date:** 27 June 2025  
**Team ID:** *[Enter Your Team ID]*  
**Project Name:** Medical Inventory Management  
**Maximum Marks:** 4 Marks

**Technical Architecture:**

Below is the architectural diagram and relevant technology stack components used in the Medical Inventory Management project.

*(Insert Architecture Diagram Here – e.g., a 3-tier architecture with Web UI → Application Layer → Database)*

**Table-1: Components & Technologies**

| **S.No** | **Component** | **Description** | **Technology** |
| --- | --- | --- | --- |
| 1. | User Interface | Web application to manage medical stock and orders | HTML, CSS, JavaScript, React.js |
| 2. | Application Logic-1 | Inventory tracking and update logic | Python (Flask or Django) |
| 3. | Application Logic-2 | Automated alert system for low stock | Python Scheduler / Celery |
| 4. | Application Logic-3 | Chatbot for medical staff queries | IBM Watson Assistant or Dialogflow |
| 5. | Database | Stores stock, supplier, and order details | MySQL / PostgreSQL |
| 6. | Cloud Database | Cloud-hosted version of the database | Amazon RDS / Google Cloud SQL |
| 7. | File Storage | Stores invoices and purchase documents | AWS S3 / Google Cloud Storage |
| 8. | External API-1 | Authentication API | OAuth 2.0 / Firebase Auth |
| 9. | External API-2 | Medicine verification or barcode scan API | OpenFDA API / Barcode Lookup API |
| 10. | Machine Learning Model | Forecasting demand for medical items | Time Series Forecasting with scikit-learn |
| 11. | Infrastructure | Cloud hosting and deployment | AWS EC2 / Google Kubernetes Engine (GKE) |

**Table-2: Application Characteristics**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
| 1. | Open-Source Frameworks | Frameworks used to develop backend and frontend | Flask/Django, React.js |
| 2. | Security Implementations | Authentication, data protection, and secure access | SSL/TLS, SHA-256, OAuth 2.0, IAM Controls |
| 3. | Scalable Architecture | Modular architecture using microservices and containerization | Docker, Kubernetes |
| 4. | Availability | High availability with cloud load balancers and replicas | AWS Elastic Load Balancer, Multi-zone RDS |
| 5. | Performance | Caching and performance optimization | Redis Cache, CDN, Asynchronous processing |

**References:**

* <https://c4model.com/>
* <https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/>
* <https://www.ibm.com/cloud/architecture>
* <https://aws.amazon.com/architecture>
* <https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d>

Example:

* The system follows a three-tier architecture with separate layers for user interface, application logic, and database management to enhance modularity and ease of maintenance.
* Cloud deployment ensures the application is accessible from multiple locations and supports scalability for growing inventory and user demands.
* Real-time tracking of medical inventory helps in automatically identifying low-stock items and generating alerts to avoid shortages.
* Secure integration with external APIs like OAuth 2.0 for authentication and OpenFDA for medicine verification enhances both functionality and data safety.

