1. Problem Statement & Objectives

Problem Statement

QuickMart, a local retail store, faces significant challenges in managing its inventory manually. This has resulted in frequent stock shortages, overstock situations, and financial losses. The current process lacks real-time inventory visibility, making it difficult to track stock levels accurately and reorder products on time.

Objectives

- Develop an automated Inventory Management System (IMS) to monitor stock levels.
- Implement an AI-driven predictive restocking feature to avoid shortages and overstocking.
- Automate the generation of purchase orders.
- Provide real-time analytics and reports for better decision-making.

2. Proposed Solution

The proposed IMS will streamline inventory management with the following core components:

- **Real-Time Stock Management:** Monitor inventory levels and update data instantly.
- **Automated Purchase Orders:** Trigger purchase orders when stock reaches a defined threshold.
- **Predictive Restocking:** AI algorithms analyze sales data to forecast future demand.
- **User Management:** Provide different access levels for store staff and management.
- **Reporting and Analytics:** Generate reports to evaluate sales trends and predict restocking needs.

The system will be designed as a web-based application, accessible from any device, providing seamless integration with QuickMart's existing operations.

3. Justification

Business Justification

- Prevent stockouts and overstocking, reducing losses.
- Improve operational efficiency and reduce manual labor.
- Provide data-driven insights for more informed decision-making.

Technical Justification

- The system will use Django and PostgreSQL for robust backend management.
- React will be used for the responsive frontend interface.
- AI algorithms will apply predictive analysis using Python libraries like NumPy and Pandas.
- Real-time inventory updates will ensure accurate tracking.

4. System Components

Prototype Designs

- Initial wireframes will include an inventory dashboard, stock management interface, purchase order management, and analytics reports.

Database Schema

- **Products:** ProductID, Name, Category, Supplier, StockLevel, ReorderPoint, UnitPrice
- **Orders:** OrderID, ProductID, SupplierID, OrderDate, Quantity, Status
- **Users:** UserID, Role, Name, Email, Password
- **Sales:** SaleID, ProductID, Quantity, SaleDate, TotalPrice

Web Forms

- **Login Page:** Secure user authentication.
- **Dashboard:** Overview of stock levels and notifications for low-stock items.
- **Stock Management:** Add, update, and delete product information.
- **Order Management:** View and manage purchase orders.
- **Reports:** Generate sales and inventory reports.

Website Details

The IMS will be hosted on a cloud platform, ensuring accessibility and scalability. The front-end UI will be designed for ease of use with interactive charts and data visualizations.

Deliverables

1. **Prototype**
- Wireframes and mockups of the interface were created using tools like Figma or Canva.
- Screenshots of the developed UI.
2. **Database Fields**
- Detailed schema for Products, Orders, Users, and Sales tables.
3. **Web Forms & Website**
- Design and functionality overview of login, dashboard, stock management, order management, and reporting pages.

4. **Development Process**

- Tools: Django, React, PostgreSQL, Python (AI model)
- Process Models: Data Flow Diagram (DFD) showing system interactions.
- Collaboration Tools: Slack, Notion, and Zoom for communication and task management.

Conclusion

The proposed Inventory Management System will provide QuickMart with a comprehensive solution to manage its inventory efficiently. By automating stock management, predicting restocking needs, and providing actionable insights, the IMS will enhance QuickMart's operational performance and profitability.

Roles

Project Manager - Saugat

- Oversees the entire project, sets deadlines, ensures smooth communication, and coordinates tasks.
- Manages collaboration tools like Slack, Notion, and Zoom.

Database & System Administrator - Shaikh

- Designs and manages the PostgreSQL database schema.
- Ensures database security, performance, and backup strategies.

Backend Developer - Lucas

- Develops the Django-based backend and integrates PostgreSQL for data management.
- Implements API endpoints and ensures secure authentication.

Frontend Developer - Ethan

- Designs and implements the React-based UI for the inventory management system.
- Works on interactive charts and data visualization features.

AI & Predictive Analytics Engineer - Leena

- Develops AI-driven predictive restocking algorithms using Python, NumPy, and Pandas.
- Integrates AI models with backend functionalities.

UI/UX Designer - Gracy

- Creates wireframes and mockups for the web application using Figma or Canva.
- Ensures an intuitive and user-friendly interface.

Quality Assurance & Documentation Lead - Abdul

- Conducts testing (unit, integration, and user testing) to ensure the system runs smoothly.
- Maintains project documentation, including system components, deliverables, and reports.