# System Analysis & Design Document

## Retail Store Inventory Forecasting System

## 1. Problem Statement & Objectives

## Problem Statement

Retail businesses often struggle with inaccurate inventory forecasting, leading to overstocking, understocking, and financial losses. This system aims to implement a data-driven forecasting model that optimizes inventory levels, reduces waste, and enhances decision-making.

## Objectives

- Develop an inventory forecasting system using historical sales data.  
- Provide accurate sales predictions to optimize inventory management.  
- Automate the forecasting process for efficiency and scalability.  
- Integrate a user-friendly interface for decision-makers to access insights.

## 2. Use Case Diagram & Descriptions

## Actors

- Store Manager  
- Inventory Analyst  
- System (AI-based Forecasting Model)

## Key Use Cases

- Upload historical sales data.  
- Generate sales forecasts.  
- View reports and analytics.  
- Adjust forecasting parameters.  
- Integrate with existing POS systems.  
\*(Include a Use Case Diagram here)\*

## 3. Functional & Non-Functional Requirements

## Functional Requirements

- Data upload and preprocessing functionality.  
- Sales forecasting using time-series models.  
- Real-time data visualization and reporting.  
- User authentication and role-based access.

## Non-Functional Requirements

- Scalability to handle large datasets.  
- Secure authentication and data encryption.  
- High system availability and performance.  
- Compatibility with multiple platforms.

## 4. Software Architecture

## Architecture Style: Microservices-based system

Machine Learning:Scikit-learn

## (Include an architecture diagram here)

## 5. Database Design & Data Modeling

## ER Diagram

- \*\*Entities:\*\* Products, Sales Data, Forecasts, Users, Reports  
- \*\*Relationships:\*\* One-to-Many (Product to Sales Data), One-to-One (User to Reports)

## Logical & Physical Schema

- \*\*Sales Data Table:\*\* ID, Product\_ID, Date, Sales\_Amount, Promotions, Weather\_Factor  
- \*\*Forecast Table:\*\* ID, Product\_ID, Forecast\_Date, Predicted\_Sales  
- \*\*User Table:\*\* ID, Name, Role, Email, Password

## (Include ER diagram here)

## 6. Data Flow & System Behavior

## DFD Levels

- \*\*Level 0:\*\* User uploads data → System processes & forecasts → User views results.  
- \*\*Level 1:\*\* Data preprocessing → Model training → Forecast generation → Report generation.

## Diagrams

- \*\*Sequence Diagram:\*\* Shows interactions between users, system, and forecasting model.  
- \*\*Activity Diagram:\*\* Represents forecasting workflow.  
- \*\*State Diagram:\*\* Depicts system states during forecasting.  
- \*\*Class Diagram:\*\* Defines system structure.  
\*(Include diagrams here)\*

## (Include wireframe images)

## 7. System Deployment & Integration

## Technology Stack

Streamlit

## Deployment Diagram

- Web UI → API Layer → Forecasting Engine → Database

## Component Diagram

- API Service  
- Data Storage Module  
- Forecasting Module  
- User Authentication

## (Include deployment and component diagrams)

## 9. Additional Deliverables

## API Documentation

- Endpoints for data upload, forecast retrieval, and user authentication.

## Testing & Validation

- Unit tests for model accuracy.  
- Integration tests for data flow.  
- User acceptance testing.

## Deployment Strategy

- streamlit cloud

## Conclusion

This system provides a robust, AI-driven approach to inventory forecasting, reducing inefficiencies and improving decision-making for retail businesses. The architecture ensures scalability, security, and ease of integration into existing retail management workflows.