Smart Inventory Management System

TRAINING PROJECT REPORT

Submitted by

Neeraj Chandel (23BCS14048)

Submitted to

Er. Pravindra Kumar Gole (T2225)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

Chandigarh University

JULY - 2025

TABLE OF CONTENTS

Chapter 1 Introduction..4

1.1 Context and Need Identification..4
1.2 Problem Identification..5
1.3 Scope, Objectives, and Significance..5
1.4 Tasks and Approach..6

Chapter 2 Literature Review / Background Study..7

2.1 Background of Inventory Management..7
2.2 Existing Systems and Solutions..7
2.3 Gaps in Current Practices..8
2.4 Review Summary..9
2.5 Problem Definition and Objectives..9

Chapter 3 System Analysis And Design..10

3.1 Evaluation and Selection of Specifications..10
3.2 Design Constraints..11
3.3 Feature Finalization..11
3.4 System Design Flow..12
3.5 Design Choices and Rationale..12
3.6 Implementation Plan and Methodology..13

Chapter 4 Algorithm Implemtation..14
4.1 Overview of Implementation..14
4.2 Software Modules..14
4.3 Algorithms and Core Functions..14
4.4 Integration and Data Flow..18

Chapter 5 Results,Analysis and Validation...20
5.1 Testing Approach and Methodology **20**
5.2 Output and Performance Analysis **20**
5.3 Limitations and Lessons Learned.. **21**

Chapter 6 Conclusion And Future Scope 22
6.1 Conclusion... **22**
6.2 Future Scope 22

References...23

Chapter 1: Introduction

1.1 Context and Need Identification

In modern times, efficient inventory management is one of the most critical operations for general stores and confectionery shops. Manual stock handling often leads to overstocking, understocking, and wastage. These inefficiencies increase costs and reduce customer satisfaction.

The Smart Inventory Management System is a Java-based console application designed to simplify stock tracking and automate restocking processes. By using demand forecasting algorithms, the system predicts how much stock is needed and when to reorder it, minimizing losses and improving operational efficiency.

This project addresses the growing need for data-driven decision-making in small and medium retail businesses that cannot afford complex ERP solutions.

1.2 Problem Identification

Traditional store management practices rely on human judgment to estimate product demand and reorder times. This leads to:

- Stockouts during high demand periods.
- Overstocking and wastage of perishable goods.
- Lack of insights into product trends and seasonal variations.
- Manual errors in tracking sales and stock levels.
- Inefficient ordering cycles causing financial loss.

Hence, an automated and intelligent system is needed that can forecast demand, monitor inventory levels, and generate reports for better planning.

1.3 Scope, Objectives, and Significance

Scope:

This project focuses on developing a console-based system using Java to manage inventory data, forecast product demand, and automatically generate reorder suggestions. The system uses **ArrayList** and **HashMap** to store, access, and manage inventory efficiently.

Objectives:

- To develop an automated system that maintains product stock details.
- To forecast future demand using statistical methods (SMA & EMA).
- To apply reorder policies such as EOQ and safety stock calculations.
- To reduce manual workload and improve stock management accuracy.
- To generate reports on sales trends, top-demand items, and low-stock alerts.

Significance:

The system provides a low-cost yet effective solution for local businesses to manage stock scientifically. It helps prevent losses due to human errors and supports smart, data-based decisions, increasing both profitability and reliability.

1.4 Tasks and Approach

- Requirement Analysis – Understanding user needs and data flow.
- System Design – Defining modules such as InventoryStore and InventoryReports.
- Algorithm Design – Implementing forecasting and reorder logic.
- Implementation – Using Java Collections for flexible data management.
- Testing – Validating accuracy, usability, and performance.
- Evaluation – Measuring system effectiveness in stock optimization.

CHAPTER 2.
LITERATURE REVIEW/BACKGROUND STUDY

2.1 Background Of Inventory Management

Inventory management systems have evolved from simple ledger-based tracking to automated software systems integrated with analytics. Earlier, manual processes caused inefficiencies and mismanagement. Modern approaches now use software tools for better control over stock and sales data.

2.2 Existing Systems And Solutions

Existing commercial inventory software such as Tally ERP and Zoho Inventory are designed for large-scale operations and require high setup costs. While they provide robust features, small retail shops often find them too complex or expensive.
Simple Excel-based tracking tools are still common but lack automation and forecasting capabilities.

2.3 Gaps In Current Practice

- Lack of automation in demand forecasting.
- High human dependency for restocking decisions.
- No predictive analysis for changing customer demand.
- Poor scalability for small businesses.
- Inability to integrate with modern technologies or cloud systems.

2.4 Review Summary

The literature indicates a strong demand for lightweight and intelligent systems that use algorithmic decision-making. Most existing systems focus on large enterprises, leaving smaller retailers behind. This project bridges that gap through an affordable, algorithm-based inventory tool.

2.5 Problem Definition and Objectives

The problem can be defined as:

“To design a smart, automated inventory management system that forecasts demand and optimizes restocking decisions for small and medium businesses.”

Objectives:

- Automate inventory updates.
- Forecast demand using moving averages.
- Calculate reorder points using EOQ and safety stock formulas.
- Generate low-stock alerts and performance reports

CHAPTER 3.
SYSTEM ANALYSIS AND DESIGN

3.1 Evaluation and Selection of Specifications

Language: Java 17

Environment: Console-based

Frameworks Used: Java Collections (ArrayList, HashMap, List, Map)

Design Pattern: Modular Object-Oriented Design

Hardware Requirements:

- Processor: Intel i5 or above
- RAM: 4GB minimum
- Storage: 200MB

Software Requirements:

- JDK 17 or higher
- Any Java IDE (Eclipse, IntelliJ, or VS Code)

3.2 Design Constraints

- No graphical interface (console-only).
- Limited to small or medium store datasets.
- Requires manual data entry of historical sales.
- No database or cloud integration in current version.

3.3 Feature Finalization

Core Features:

- Add, update, and delete products.
- Maintain stock quantities and reorder levels.
- Demand forecasting using SMA & EMA.
- Automatic reorder suggestions.
- Generate reports for analysis.

Additional Features (Future Scope):

- Integration with databases or cloud.
- Real-time stock monitoring.
- Barcode scanning and billing system.

3.4 Design Flow

- User inputs product details (ID, Name, Price, Stock).
- Sales data is recorded to analyze demand trends.
- Forecast algorithms predict future needs.
- The system compares forecasted demand with current stock.
- Reorder decision is made using ROP and EOQ formulas.
- Reports are generated for store owners.

3.5 Design Choices and Rationale

- ArrayList: Dynamic product listing.
- HashMap: Fast lookup of product details using IDs.
- SMA & EMA: Efficient, simple algorithms suitable for time-series data.
- Modular Classes:
    - InventoryStore – manages products and stock.
    - InventoryReports – generates summaries and alerts.
    - Main – coordinates execution.

3.6 Implementation Plan and Methodology

- Planning & Analysis: Identify store needs.
- Design: Create UML diagrams and data models.
- Implementation: Develop modular code in Java.
- Testing: Unit and integration testing for each feature.
- Evaluation: Validate accuracy of demand forecasting.

CHAPTER 4.
ALGORITHM IMPLEMENTATION

4.1 Overview of Implementation

The project uses forecasting and optimization algorithms to automate inventory control. The core functions are designed for clarity, efficiency, and scalability.

4.2 Software Modules

- **InventoryStore:** Maintains all product data.
- **DemandForecast:** Applies SMA and EMA algorithms.
- **ReorderPolicy:** Implements EOQ and ROP calculations.
- **InventoryReports:** Generates alerts and analytics.

4.3 Sorting Algorithm Implementations

(a) Simple Moving Average (SMA):

Predicts next demand = Average of last N sales.

Complexity: O(N)

(b) Exponential Moving Average (EMA):

Uses weighted average giving more importance to recent data.

Complexity: O(N)

(c) EOQ Formula:

$$EOQ = \sqrt{\frac{2DS}{H}}$$

Where:

D = Demand rate, S = Setup cost, H = Holding cost

(d) Reorder Point (ROP):

$$ROP = (AverageDemand \times LeadTime) + SafetyStock$$

These formulas ensure optimal stock without shortage or overstocking.

4.4 Integration with Visualization

Each product’s data (sales history, reorder level) flows between modules:

Main → InventoryStore → DemandForecast → ReorderPolicy → InventoryReports

CHAPTER 5. Results Analysis and Validation

5.1 Testing Approach and Methodology

- Unit Testing: Checked each function for accuracy.
- Integration Testing: Validated data flow between modules.
- User Testing: Simulated store operations with 20 sample products.

5.2 Output And Performance Analysis

Sample Output:

```
Low Stock Alert:
Product: Sugar, Current Stock: 10, Forecasted Demand: 25
Suggested Order Quantity: 30 units
Expected Lead Time: 3 days
```

5.3 Limitations and Lessons Learned

- Currently console-based; GUI integration needed.
- No database connection for persistent storage.
- Forecast accuracy depends on sales history size.
- Future versions should automate data input.

Chapter 6: Conclusion and Future Scope

6.1 Conclusion

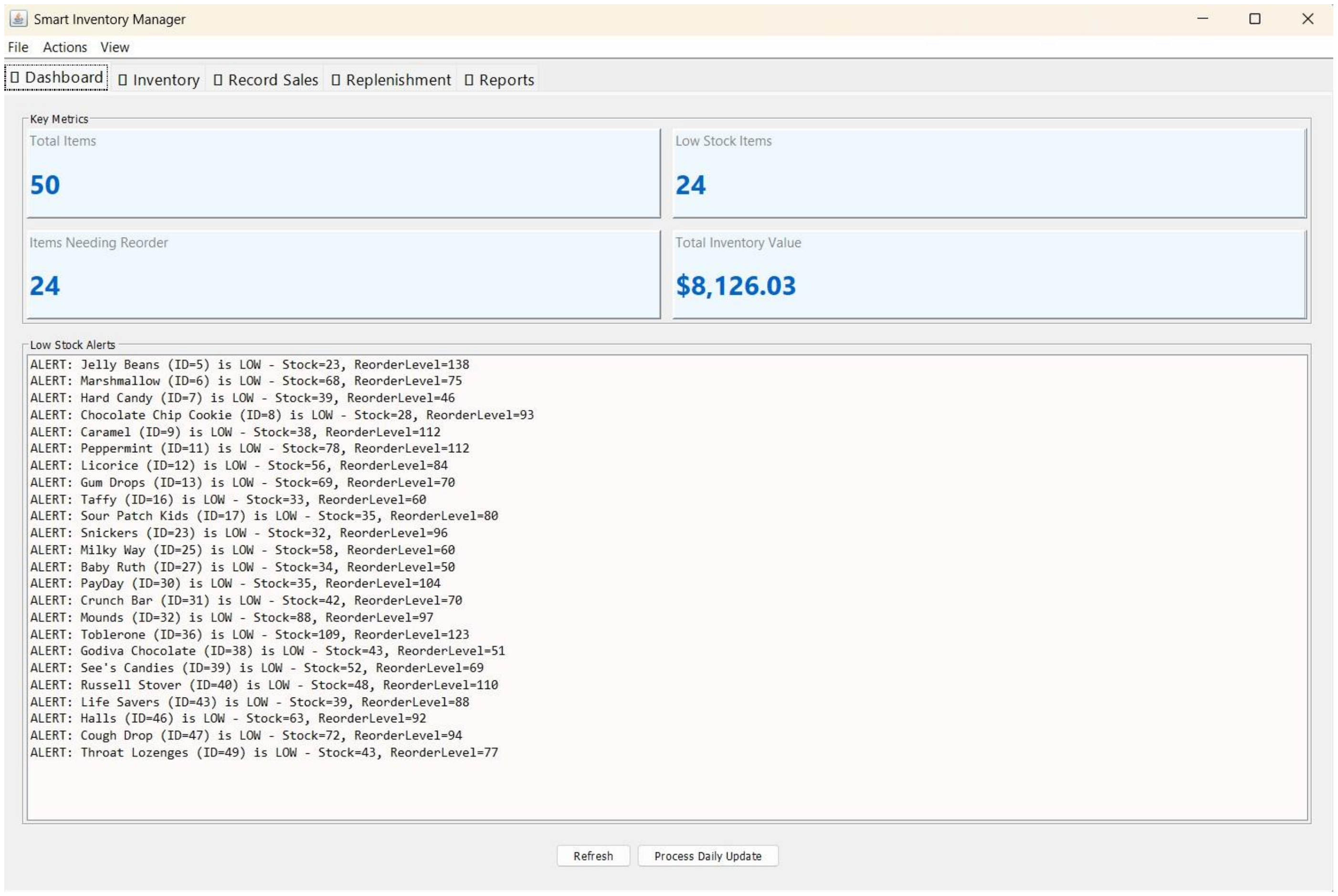
The Smart Inventory Management System successfully automates key inventory operations using simple forecasting and optimization algorithms.
It minimizes human errors, prevents stockouts, and helps store owners make informed decisions.
The modular design ensures scalability for future upgrades such as GUI, cloud, or database connectivity.

6.2 Future Scope

- Integrate a Graphical User Interface (GUI) using JavaFX or Swing.
- Connect to a Database (MySQL or Firebase) for data storage.
- Enable Cloud Synchronization for multi-store management.
- Add Barcode Scanning and Billing modules.
- Implement AI-based predictive analytics for smarter forecasting**.**

References

- Java Documentation – Oracle.
- “Principles of Inventory Management” – Springer Publications.
- GeeksforGeeks Java Collections Tutorials.
- TutorialsPoint: Java Object-Oriented Concepts.
- Research papers on demand forecasting algorithms.
- Tally ERP & Zoho Inventory Documentation (for comparative study).



Key Metrics

Total Items
50

Low Stock Items
24

Items Needing Reorder
24

Total Inventory Value
$8,126.03

Low Stock Alerts

```
ALERT: Jelly Beans (ID=5) is LOW - Stock=23, ReorderLevel=138
ALERT: Marshmallow (ID=6) is LOW - Stock=68, ReorderLevel=75
ALERT: Hard Candy (ID=7) is LOW - Stock=39, ReorderLevel=46
ALERT: Chocolate Chip Cookie (ID=8) is LOW - Stock=28, ReorderLevel=93
ALERT: Caramel (ID=9) is LOW - Stock=38, ReorderLevel=112
ALERT: Peppermint (ID=11) is LOW - Stock=78, ReorderLevel=112
ALERT: Licorice (ID=12) is LOW - Stock=56, ReorderLevel=84
ALERT: Gum Drops (ID=13) is LOW - Stock=69, ReorderLevel=70
ALERT: Taffy (ID=16) is LOW - Stock=33, ReorderLevel=60
ALERT: Sour Patch Kids (ID=17) is LOW - Stock=35, ReorderLevel=80
ALERT: Snickers (ID=23) is LOW - Stock=32, ReorderLevel=96
ALERT: Milky Way (ID=25) is LOW - Stock=58, ReorderLevel=60
ALERT: Baby Ruth (ID=27) is LOW - Stock=34, ReorderLevel=50
ALERT: PayDay (ID=30) is LOW - Stock=35, ReorderLevel=104
ALERT: Crunch Bar (ID=31) is LOW - Stock=42, ReorderLevel=70
ALERT: Mounds (ID=32) is LOW - Stock=88, ReorderLevel=97
ALERT: Toblerone (ID=36) is LOW - Stock=109, ReorderLevel=123
ALERT: Godiva Chocolate (ID=38) is LOW - Stock=43, ReorderLevel=51
ALERT: See's Candies (ID=39) is LOW - Stock=52, ReorderLevel=69
ALERT: Russell Stover (ID=40) is LOW - Stock=48, ReorderLevel=110
ALERT: Life Savers (ID=43) is LOW - Stock=39, ReorderLevel=88
ALERT: Halls (ID=46) is LOW - Stock=63, ReorderLevel=92
ALERT: Cough Drop (ID=47) is LOW - Stock=72, ReorderLevel=94
ALERT: Throat Lozenges (ID=49) is LOW - Stock=43, ReorderLevel=77
```