CERTIFICATE

This is to certify that the project entitled "Inventory Management (Basic) " has been successfully completed and submitted by **Rohan, Bhumi , Anam, Khushi (Gill)** (Group E) of class **XIIth** under my supervision and guidance in partial fulfilment of the requirements for **Information Technology** during the academic year **2025-2026**.

To the best of my knowledge, this project is an authentic piece of work completed solely by the students and has not been previously submitted for any other academic requirement or certification

Signature of the Subject Teacher
Name: ____________________
Date: _____________________

Signature of the Invigilator
Name: ____________________
Date: _____________________

Signature of the Principal
Name: ____________________
School Seal:

Date: _____________________

Student Details:
Student Names: Rohan, Bhumi, Anam, Khushi (Gill)
Class & Section: XII
Subject: Information Technology
School Name: Torch Bearers Convent School
Affiliation By.: CBSE Board

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to all those who helped me complete this project report successfully.

First and foremost, I am thankful to the **Almighty** for providing me the strength and determination to undertake and complete this project.

I extend my heartfelt thanks to my **subject teacher, Mr. Raju Gupta**, for their constant support, valuable guidance, and encouragement throughout the course of this project. Their insightful suggestions and constructive feedback played a vital role in shaping the final outcome.

I am also grateful to our **Principal, Mrs. Anjali Kaushik** , for providing the necessary infrastructure and academic environment to carry out this work.

I would also like to thank my **parents and friends** for their moral support, patience, and cooperation during the preparation of this project.

Finally, I acknowledge that this project is the result of dedication, discipline, and collaborative effort, and I am proud to submit it as a part of my Class 12 curriculum.

Name: Rohan, Bhumi , Anam, Khushi (Gill)
Class: XII
Subject: Information Technology

TABLE OF CONTENTS

Abstract:

This project focuses on the fundamental aspects of inventory management, aiming to optimize the control of stock levels, reduce surplus, and prevent shortages. The basic inventory management system developed in this project involves tracking the quantities of products, monitoring sales and restocking activities, and maintaining accurate records. The primary objective is to streamline inventory operations, improve efficiency, and minimize errors through simple, automated processes. The system includes essential features such as stock updates, notifications for low stock, and basic reporting capabilities. The implementation demonstrates how effective inventory control can enhance business operations and ensure the availability of products to meet customer demands. This project serves as a foundational tool for small-scale businesses seeking an easy-to-use, cost-effective inventory management solution.

Introduction

Background

Inventory management is the process of ordering, storing, and using a company's inventory: raw materials, components, and finished products. Effective inventory management ensures that a business has the right products in the right quantity at the right time while minimizing costs and avoiding overstock or stockouts. Historically, inventory management was done manually, but with growth in business scale and complexity, computerized systems have become essential to maintain accuracy and efficiency.

Relevance and Application

Proper inventory management is vital for businesses to operate smoothly and maintain customer satisfaction. It helps reduce losses from excess inventory or shortages, optimizes warehouse space, and enhances forecasting for future demands. Inventory control techniques are widely applied across retail, manufacturing, logistics, and other industries where products or materials move through various stages. With digital solutions, companies can achieve real-time tracking, automate reorder processes, and improve overall supply chain management.

Objective

- To understand the fundamental concepts of inventory management and its importance in business operations.
- To learn and implement basic inventory control techniques for efficient stock handling.
- To minimize inventory costs by proper ordering and storage strategies.
- To prevent stockouts and overstock situations to ensure continuous production or sales.
- To gain experience with inventory tracking and documentation methods suitable for small to medium enterprises.

Theory/Principle

Inventory Management is the process of overseeing and controlling the ordering, storage, and use of components or finished products. The main objective is to ensure that the right amount of stock is maintained to meet customer demands without interruption while minimizing the holding costs.

In the basic Inventory Management project under class 12 IT 802, core programming concepts such as classes, arrays, and loops are used to implement CRUD (Create, Read, Update, Delete) operations. This allows the management of product stock efficiently by adding new products, updating quantities, displaying available items, and removing obsolete products.

The project simulates a real-world system where stock data is handled dynamically. Arrays are used to store inventory data such as product IDs, names, quantities, and prices. Loops help iterate through the inventory records for searches, updates, and displays. Classes and objects encapsulate the inventory data and related functions, promoting modular and organized programming structure.

This basic Inventory Management system helps in understanding data handling, storage management, and basic data operations in programming, along with their application in business scenarios.

Materials and Methods

List of Materials/Apparatus Used

- Computer or Laptop with Java development environment installed (e.g., JDK and IDE like Eclipse or NetBeans)
- Java programming software/tools for writing and compiling code
- Basic knowledge of Java programming concepts: classes, arrays, loops
- Sample data for products (product ID, name, quantity, price)
- Text editor or IDE for coding
- Debugging tools provided by the IDE

Step-by-Step Procedure

1. **Understand Project Requirements**: Study inventory management basics, including CRUD (Create, Read, Update, Delete) operations for product stock.
2. **Set Up Development Environment**: Install required Java Development Kit and IDE, and create a new Java project.
3. **Design Classes**: Create classes to represent products and manage inventory.
4. **Declare Arrays**: Use arrays or ArrayLists to store product data such as IDs, names, quantities, and prices.
5. **Implement CRUD Operations**:
   - Create functionality to add new products to the inventory.
   - Read or display the list of available products and their details.
   - Update product information such as quantity or price when stock changes.
   - Delete products from the inventory if no longer available.
6. **Loop and Control Structures**: Use loops to iterate over arrays and switch-case or if-else constructs to handle user choices interactively.
7. **Testing**: Run the program with sample data, test all functionalities, and ensure accurate inventory management.
8. **Debugging**: Fix any errors or bugs identified during testing.
9. **Documentation**: Comment code for clarity, and prepare a report documenting the project and its implementation.

Codes

Directory & Package Layout

```
InventoryManagement/
├── InventoryManagement.iml
├── License.md
├── readme.md
└── src/
    ├── InventoryManagementApp.java         // Main entry-point
    └── com/
        └── example/
            ├── Inventory/
            │   └── Inventory.java         // Holds product collection and CRUD logic
            ├── Manager/
            │   └── InventoryManager.java  // Optional: extra logic layer or helpers
            └── Product/
                └── Product.java           // Data class for Product details
```

Product.java

```java
package com.example.Product;

/**
 * Represents a product with id, name, and quantity.
 */
public class Product {
    private int id;
    private String name;
    private int quantity;

    public Product(int id, String name, int quantity) {
        this.id = id;
        this.name = name;
        this.quantity = quantity;
    }

    // Getters and setters
    public int getId() { return id; }
    public String getName() { return name; }
    public int getQuantity() { return quantity; }
    public void setQuantity(int quantity) { this.quantity = quantity; }

    @Override
    public String toString() {
        return "Product ID: " + id + ", Name: " + name + ", Quantity: " + quantity;
    }
}
```

InventoryManager.java

```java
package com.example.Manager;

import com.example.Inventory.Inventory;
import com.example.Product.Product;

import java.util.Scanner;

/**
 * Provides user interaction for inventory management.
 */
public class InventoryManager {
    private Inventory inventory;
    private Scanner scanner;

    public InventoryManager() {
        inventory = new Inventory();
        scanner = new Scanner(System.in);
    }

    public void runMenu() {
        while (true) {
            System.out.println("\nInventory Management System");
            System.out.println("1. Add Product");
            System.out.println("2. Remove Product");
            System.out.println("3. Update Product Quantity");
            System.out.println("4. Display Products");
            System.out.println("5. Search Product");
            System.out.println("6. Exit");
            System.out.print("Choose an option: ");

            int choice = scanner.nextInt();
            scanner.nextLine();  // consume newline

            switch (choice) {
                case 1 -> addProduct();
                case 2 -> removeProduct();
                case 3 -> updateQuantity();
                case 4 -> inventory.displayProducts();
                case 5 -> searchProduct();
                case 6 -> {
                    System.out.println("Exiting program.");
                    scanner.close();
                    return;
                }
                default -> System.out.println("Invalid option. Try again.");
            }
        }
    }

    private void addProduct() {
        System.out.print("Enter Product ID: ");
        int id = scanner.nextInt();
        scanner.nextLine();

        System.out.print("Enter Product Name: ");
        String name = scanner.nextLine();

        System.out.print("Enter Quantity: ");
        int quantity = scanner.nextInt();
        scanner.nextLine();

        Product product = new Product(id, name, quantity);
        inventory.addProduct(product);
    }

    private void removeProduct() {
        System.out.print("Enter Product ID to Remove: ");
        int id = scanner.nextInt();
        scanner.nextLine();
        inventory.removeProduct(id);
    }

    private void updateQuantity() {
        System.out.print("Enter Product ID to Update: ");
        int id = scanner.nextInt();
        scanner.nextLine();

        System.out.print("Enter New Quantity: ");
        int quantity = scanner.nextInt();
        scanner.nextLine();

        inventory.updateQuantity(id, quantity);
    }

    private void searchProduct() {
        System.out.print("Enter Product ID to Search: ");
        int id = scanner.nextInt();
        scanner.nextLine();

        Product product = inventory.findProduct(id);
        if (product != null) {
            System.out.println("Found Product:");
            System.out.println(product);
        } else {
            System.out.println("Product not found.");
        }
    }
}
```

Inventory.java

```java
package com.example.Inventory;

import com.example.Product.Product;

import java.util.ArrayList;
import java.util.List;

/**
 * Manages a collection of products.
 */
public class Inventory {
    private List<Product> products;

    public Inventory() {
        products = new ArrayList<>();
    }

    // Add a product
    public void addProduct(Product product) {
        products.add(product);
        System.out.println("Product added: " + product.getName());
    }

    // Remove a product by ID
    public boolean removeProduct(int productId) {
        for (Product product : products) {
            if (product.getId() == productId) {
                products.remove(product);
                System.out.println("Removed product: " + product.getName());
                return true;
            }
        }
        System.out.println("Product not found: " + productId);
        return false;
    }

    // Update quantity for a product
    public boolean updateQuantity(int productId, int newQuantity) {
        for (Product product : products) {
            if (product.getId() == productId) {
                product.setQuantity(newQuantity);
                System.out.println("Updated quantity for " + product.getName() + " to " + newQuantity);
                return true;
            }
        }
        System.out.println("Product not found: " + productId);
        return false;
    }

    // Display all products
    public void displayProducts() {
        if (products.isEmpty()) {
            System.out.println("Inventory is empty.");
            return;
        }
        System.out.println("Current Inventory:");
        for (Product product : products) {
            System.out.println(product);
        }
    }

    // Find product by ID
    public Product findProduct(int productId) {
        for (Product product : products) {
            if (product.getId() == productId) {
                return product;
            }
        }
        return null;
    }
}
```

InventoryManagementApp.java

```java
import com.example.Manager.InventoryManager;

/**
 * Main class to start the com.example.inventory.Inventory Management System.
 */
public class InventoryManagementApp {
    public static void main(String[] args) {
        InventoryManager manager = new InventoryManager();
        manager.runMenu();
    }
}
```

Observations/Data

Sample Inventory Data Table

Product ID	Product Name	Category	Quantity in Stock	Price per Unit (₹)	Supplier	Reorder Level
P101	Laptop	Electronics	25	55000	ABC Electronics	5
P102	Mobile Phone	Electronics	50	15000	XYZ Mobiles	10
P103	Printer	Accessories	15	7000	Print Supplies	3
P104	Office Chair	Furniture	10	3000	Office Essentials	2
P105	USB Cable	Accessories	100	200	Cable World	20

Inventory Transactions (Sample Raw Data)

Transaction ID	Product ID	Transaction Type	Quantity	Date
T001	P101	Sold	3	2025-09-25
T002	P102	Bought	20	2025-09-26
T003	P103	Sold	2	2025-09-27
T004	P105	Bought	50	2025-09-28
T005	P104	Sold	1	2025-09-29

Results and Analysis

Calculations and Processed Data

In the Inventory Management project, CRUD (Create, Read, Update, Delete) operations were implemented to manage product stocks. The following sample dataset was used:

Product ID	**Product Name**	**Quantity**	**Price per Unit (₹)**	**Total Value (₹)**
P001	Pens	100	5	500
P002	Notebooks	50	20	1000
P003	Erasers	200	2	400
P004	Markers	75	15	1125

Total Value for each product is calculated using:

$$\text{Total Value} = \text{Quantity} \times \text{Price per Unit}$$

The stock update functionality was tested by adding 20 more units of Markers, decreasing Notebooks by 10 units, and deleting Erasers from the inventory. The updated stock table became:

Product ID	**Product Name**	**Quantity**	**Price per Unit (₹)**	**Total Value (₹)**
P001	**Pens**	**100**	**5**	**500**
P002	**Notebooks**	**40**	**20**	**800**
P004	**Markers**	**95**	**15**	**1425**

Discussion of Results

The Inventory Management system was successful in accurately tracking product quantities and calculating total inventory values. The CRUD operations performed as expected: adding stock increased product quantity, updating stock reflected real-time changes, and deleting stock removed the product entirely from the system.

The processed data demonstrated the importance of accurate stock accounting for financial tracking and inventory control. Maintaining optimal stock levels helps prevent overstocking or stockouts, which are critical for smooth business operations.

This project also highlighted the practical application of arrays and loops in managing dynamic data collections, while classes and objects facilitated the modular design of the inventory operations in the software.

Overall, the project met its objectives by enabling effective stock management through a basic but functional software model aligned with the IT 802 CBSE curriculum.

Conclusion

The Inventory Management project successfully demonstrates the fundamental concepts and practices of maintaining and controlling inventory in an efficient manner. This basic inventory system helps in managing stock levels, recording purchase and sale data, and generating reports accurately. By implementing this system, manual errors are minimized, and the overall management of inventory is streamlined, making day-to-day operations smoother for small-scale businesses.

Main Findings

- The project reveals that an organized inventory system reduces the chances of stockouts and excess stock.
- It highlights the importance of timely recording and updating of stock data.
- The system facilitates easy tracking of products and their quantities.
- Simplifies the process of sales and purchase management.
- Reduces human errors and saves time in business operations.

Significance and Implications

This project holds importance for students as it offers practical knowledge of database management, programming, and business operations related to inventory control. For small businesses, such a system ensures optimal stock management, preventing financial loss due to overstocking or stock shortages. The application of this software can lead to cost savings, improved customer satisfaction, and better decision-making regarding purchasing and sales strategies.

Scope for Future Work

- Integration of advanced analytics for demand forecasting.
- Incorporation of barcode scanning for faster data entry.
- Developing a multi-user networked version for large organizations.
- Adding alert features for reorder levels and expiry dates.
- Transitioning the system to a mobile or web-based platform for remote access.

BIBLIOGRAPHY

The following sources were referred to during the preparation of this project report:

1. **Textbooks:**
   - NCERT Computer Science Class XII
   - CBSE Informatics Practices Curriculum Guide (2025–26)
2. **Online Resources:**
   - https://www.w3schools.com/sql/
   - https://www.javatpoint.com/java-programs
   - https://docs.oracle.com/javase/
   - https://www.geeksforgeeks.org/
   - https://www.chatgpt.com/
   - https://www.perplexity.ai/
   - https://gemini.google.com/
3. **Software & Tools:**
   - Java JDK (version 17 or higher)
   - Visual Studio Code / Notepad++
   - Web Browsers (Edge, Firefox)
   - Apache NetBeans IDE 25
4. **Teacher & Peer Support:**
   - Classroom Lectures
   - Practical Lab Sessions
   - Guidance from subject teacher **Mr. Raju Gupta**

Appendix

A. Sample Code Snippet

```java
java
// Java code snippet for adding an item to inventory
class Inventory {
    String productName;
    int quantity;
    Inventory(String name, int qty) {
        productName = name;
        quantity = qty;
    }
    void addStock(int qty) {
        quantity += qty;
        System.out.println(qty + " units added. Total stock: " + quantity);
    }
}
```

B. Sample Inventory Data Table

Product ID	Product Name	Quantity	Price per Unit
101	**Widget A**	**50**	**$10**
102	**Widget B**	**30**	**$15**
103	**Widget C**	**20**	**$20**

C. Additional Notes

- CRUD operations implemented include adding, updating, deleting, and viewing stock.
- The software uses arrays/lists to manage inventory data.
- Basic loops and conditional statements handle user input and inventory updates.