INVENTORY MANAGEMENT SYSTEM

A MINI-PROJECT REPORT

Submitted by

BASKAR A	**240701074**
PRAGADEESH S	**240701388**
JOBESON J	**240701217**

in partial fulfillment of the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

An Autonomous Institute

CHENNAI

NOVEMBER 2025

BONAFIDE CERTIFICATE

Certified that this project **"INVENTORY MANAGEMENT SYSTEM FOR RETAIL SHOP"** is the Bonafide work of **"BASKAR A ,PRAGADEESH S AND JOBESON J"** who carried out the project work under my supervision.

SIGNATURE

Deepa B

ASSISTANT PROFESSOR SG

Dept. of Computer Science and Engg,
Rajalakshmi Engineering College
Chennai

This mini project report is submitted for the viva voce examination to be held on _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

The Inventory Management System is designed to help retail shop owners manage their stock efficiently and automate the billing process. The system allows the user to log in, view products, generate bills, and automatically update stock levels after each sale. When a product runs out, the system marks it as “Out of Stock.”

Developed using **Java (Swing)** for the front end, **MySQL** for the backend, and **JDBC** for connectivity, this project provides a simple yet powerful solution for small retail shops to maintain accurate stock and sales records.

ACKNOWLEDGEMENT

We express our sincere thanks to our beloved and honorable chairman **MR. S. MEGANATHAN** and the chairperson **DR. M.THANGAM MEGANATHAN** for their timely support and encouragement.

We are greatly indebted to our respected and honorable principal **Dr. S.N. MURUGESAN** for his able support and guidance.

No words of gratitude will suffice for the unquestioning support extended to us by our Head Of The Department **Dr. E.M. MALATHY** and our Deputy Head Of The Department **Dr. J. MANORANJINI** for being ever supporting force during our project work

We also extend our sincere and hearty thanks to our internal guide **Deepa B** for her valuable guidance and motivation during the completion of this project.

Our sincere thanks to our family members, friends and other staff members of computer science engineering.

BASKAR A

PRAGADEESH S

JOBESON J

TABLE OF CONTENTS

✓ CHAPTER NO	✓ TITLE	✓ PAGE NO
✓ ABSTRACT		✓ iv
✓ 1	✓ INTRODUCTION	✓ 1
✓ 1.1	✓ INTRODUCTION	✓ 1
✓ 1.2	✓ SCOPE OF THE WORK	✓ 2
✓ 1.3	✓ PROBLEM STATEMENT	✓ 2
✓ 1.4	✓ AIM AND OBJECTIVES	✓ 2
✓ 2	✓ SYSTEM SPECIFICATIONS	✓ 3
✓ 2.1	✓ HARDWARE SPECIFICATIONS	✓ 3
✓ 2.2	✓ SOFTWARE SPECIFICATIONS	✓ 3
✓ 3	✓ MODULE DESCRIPTION	✓ 4
✓ 4	✓ CODING	✓ 6
✓ 5	✓ SCREENSHOTS	✓ 14
✓ 6	✓ CONCLUSION AND FUTURE ENHANCEMENT	✓ 17
✓	✓ REFERENCES	✓ 18

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

The Inventory Management System for a retail shop simplifies daily sales and product tracking activities. It helps store owners keep accurate records of stock quantities, item prices, and billing. The system automatically updates product availability and flags “Out of Stock” items, improving efficiency and reducing manual errors.

1.2 SCOPE OF THE WORK

This project helps retail shops move from manual stock entry and billing to an automated system. It provides a user-friendly interface for product lookup, billing, and stock maintenance. The system ensures that product information, pricing, and availability are always accurate.

1.3 PROBLEM STATEMENT

Most small retail shops still depend on manual methods to manage inventory and generate bills. This leads to human error, delays, and inaccurate stock counts. A digital system is required to handle real-time updates, billing, and product monitoring efficiently**.**

1.4 AIM AND OBJECTIVES OF THE PROJECT

- To create a user-friendly interface for managing retail inventory.
- To provide secure login access for authorized users.
- To generate bills and update product quantities automatically.
- To alert users when products are out of stock.

CHAPTER 2

SYSTEM SPECIFICATIONS

2.1 HARDWARE SPECIFICATIONS

Component	Specification
Processor	Intel Core i3 / i5
Memory	4 GB RAM (minimum)
Hard Disk	10 GB free space
Display	1024×768 pixels (minimum)

2.2 SOFTWARE SPECIFICATIONS

Component	Specification
Operating system	Windowa 10/11
Front end	Java(Swing GUI)
Back end	MySQL
Connectivity	JDBC(java database connectivity)
IDE	IntelliJ IDEA community edition

CHAPTER 3

MODULE DESCRIPTION

T his application consists of two modules.

1. Login Module

- Allows the user to log in using a username and password.
- On successful login, opens the billing window.
- Prevents unauthorized access.

2. Billing Module

- Accepts product ID and quantity.
- Fetches product name, price, and stock from the database.
- Displays “Out of Stock” when stock = 0.
- Updates quantity after each sale and calculates the total bill.

CHAPTER 4

SAMPLE CODING

Database Setup Example:

CREATE DATABASE inventorydb;

USE inventorydb;

CREATE TABLE products (id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(100) NOT NULL, category VARCHAR(50), quantity INT NOT NULL DEFAULT 0, price DECIMAL(10,2) NOT NULL DEFAULT 0.00, created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP);

USE inventorydb;

DESC products;

INSERT INTO products (name, category, quantity, price)

VALUES

('Milk', 'Dairy', 40, 25.00),

('Bread', 'Bakery', 60, 30.00),

('Eggs', 'Grocery', 100, 6.00),

('Sugar', 'Grocery', 80, 45.00),

('Tea Powder', 'Beverages', 50, 120.00),

('Coffee', 'Beverages', 40, 150.00),

('Biscuits', 'Snacks', 70, 15.00),

('Wheat Flour', 'Grocery', 90, 55.00),

('Cooking Oil', 'Grocery', 60, 160.00),

('Toothpaste', 'Toiletries', 50, 65.00),

('Toothbrush', 'Toiletries', 75, 25.00),

('Detergent', 'Household', 45, 90.00),

('Shampoo', 'Toiletries', 35, 85.00),

('Soap', 'Toiletries', 80, 25.00),

('Rice', 'Grocery', 100, 60.00);

Sample 1: Login page

```
package com.inventory.ui;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class LoginFrame extends JFrame {

    private JTextField txtUsername;
    private JPasswordField txtPassword;
    private JButton btnLogin, btnClear;

    // hardcoded credentials
    private static final String USERNAME = "dbms";
    private static final String PASSWORD = "rec123";

    public LoginFrame() {
        setTitle("🔐 Inventory Management Login");
        setSize(450, 350);
        setLocationRelativeTo(null);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```
        // Colors
        Color primaryColor = new Color(52, 152, 219);
        Color accentColor = new Color(41, 128, 185);
        Color buttonColor = new Color(46, 204, 113);
        Color textColor = Color.WHITE;

        // Background gradient panel
        JPanel bgPanel = new JPanel() {
            @Override
            protected void paintComponent(Graphics g) {
                super.paintComponent(g);
                Graphics2D g2d = (Graphics2D) g;
                GradientPaint gp = new GradientPaint(0, 0, primaryColor,
getWidth(), getHeight(), accentColor);
                g2d.setPaint(gp);
                g2d.fillRect(0, 0, getWidth(), getHeight());
            }
        };
        bgPanel.setLayout(new GridBagLayout());
        add(bgPanel);

        // Card-style panel
        JPanel panel = new JPanel(new GridBagLayout());
        panel.setPreferredSize(new Dimension(320, 220));
        panel.setBackground(Color.WHITE);
        panel.setBorder(BorderFactory.createLineBorder(primaryColor, 2, true));

        GridBagConstraints gbc = new GridBagConstraints();
        gbc.insets = new Insets(10, 10, 10, 10);
        gbc.fill = GridBagConstraints.HORIZONTAL;

        JLabel lblTitle = new JLabel("LOGIN", SwingConstants.CENTER);
        lblTitle.setFont(new Font("Segoe UI", Font.BOLD, 22));
        lblTitle.setForeground(primaryColor);

        gbc.gridx = 0;
        gbc.gridy = 0;
        gbc.gridwidth = 2;
        panel.add(lblTitle, gbc);
```

```
        gbc.gridwidth = 1;

        JLabel lblUsername = new JLabel("Username:");
        lblUsername.setFont(new Font("Segoe UI", Font.PLAIN, 14));
        gbc.gridx = 0;
        gbc.gridy = 1;
        panel.add(lblUsername, gbc);

        txtUsername = new JTextField(15);
        txtUsername.setFont(new Font("Segoe UI", Font.PLAIN, 14));
        gbc.gridx = 1;
        gbc.gridy = 1;
        panel.add(txtUsername, gbc);

        JLabel lblPassword = new JLabel("Password:");
        lblPassword.setFont(new Font("Segoe UI", Font.PLAIN, 14));
        gbc.gridx = 0;
        gbc.gridy = 2;
        panel.add(lblPassword, gbc);

        txtPassword = new JPasswordField(15);
        txtPassword.setFont(new Font("Segoe UI", Font.PLAIN, 14));
        gbc.gridx = 1;
        gbc.gridy = 2;
        panel.add(txtPassword, gbc);

        btnLogin = new JButton("Login");
        btnLogin.setBackground(buttonColor);
        btnLogin.setForeground(textColor);
        btnLogin.setFont(new Font("Segoe UI", Font.BOLD, 14));
        btnLogin.setFocusPainted(false);

        gbc.gridx = 0;
        gbc.gridy = 3;
        gbc.gridwidth = 1;
        panel.add(btnLogin, gbc);

        btnClear = new JButton("Clear");
        btnClear.setBackground(new Color(231, 76, 60));
```

```
        btnClear.setForeground(textColor);
        btnClear.setFont(new Font("Segoe UI", Font.BOLD, 14));
        btnClear.setFocusPainted(false);

        gbc.gridx = 1;
        gbc.gridy = 3;
        panel.add(btnClear, gbc);

        bgPanel.add(panel);

        // Action listeners
        btnLogin.addActionListener(e -> attemptLogin());
        btnClear.addActionListener(e -> clearFields());

        // Enter key triggers login
        txtPassword.addKeyListener(new KeyAdapter() {
            @Override
            public void keyPressed(KeyEvent e) {
                if (e.getKeyCode() == KeyEvent.VK_ENTER) attemptLogin();
            }
        });
    }

    private void attemptLogin() {
        String username = txtUsername.getText().trim();
        String password = new String(txtPassword.getPassword());

        if (username.isEmpty() || password.isEmpty()) {
            JOptionPane.showMessageDialog(this, "Please enter username and
password.", "Missing", JOptionPane.WARNING_MESSAGE);
            return;
        }

        // Hardcoded credential check
        if (USERNAME.equals(username) && PASSWORD.equals(password)) {
            JOptionPane.showMessageDialog(this, "☑ Login successful!");
            // open billing and close login
            SwingUtilities.invokeLater(() -> {
                new BillingFrame().setVisible(true);
```

```
            });
            dispose();
        } else {
            JOptionPane.showMessageDialog(this, "❌ Invalid username or
password.", "Login Failed", JOptionPane.ERROR_MESSAGE);
            txtPassword.setText("");
            txtUsername.requestFocus();
        }
    }

    private void clearFields() {
        txtUsername.setText("");
        txtPassword.setText("");
        txtUsername.requestFocus();
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new LoginFrame().setVisible(true));
    }
}
```

Sample 2

Sample 2 depicts the billing page

```
package com.inventory.ui;

import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
```

```
public class BillingFrame extends JFrame {

    private JTextField txtProductName, txtQuantity;
    private JTable table;
    private DefaultTableModel model;
    private JLabel lblGrandTotal;
    private double grandTotal = 0;

    public BillingFrame() {
        setTitle("🛒 Inventory Billing System");
        setSize(850, 600);
        setLocationRelativeTo(null);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());

        // ---- Colors ----
        Color headerColor = new Color(52, 152, 219);
        Color bgColor = new Color(245, 245, 245);
        Color buttonColor = new Color(46, 204, 113);
        Color textColor = Color.WHITE;

        getContentPane().setBackground(bgColor);

        // ---- Top Panel ----
        JPanel topPanel = new JPanel(new FlowLayout());
        topPanel.setBackground(headerColor);

        JLabel lblProduct = new JLabel("Product Name:");
        lblProduct.setForeground(textColor);
        topPanel.add(lblProduct);

        txtProductName = new JTextField(10);
        topPanel.add(txtProductName);

        JLabel lblQty = new JLabel("Quantity:");
        lblQty.setForeground(textColor);
        topPanel.add(lblQty);
```

```
        txtQuantity = new JTextField(5);
        topPanel.add(txtQuantity);

        JButton btnAdd = new JButton("Add to Bill");
        btnAdd.setBackground(buttonColor);
        btnAdd.setForeground(textColor);
        topPanel.add(btnAdd);

        JButton btnPrint = new JButton("Print Bill");
        btnPrint.setBackground(new Color(231, 76, 60));
        btnPrint.setForeground(textColor);
        topPanel.add(btnPrint);

        add(topPanel, BorderLayout.NORTH);

        // ---- Table ----
        String[] columns = {"S.No", "Product Name", "Quantity", "Price",
"Total"};
        model = new DefaultTableModel(columns, 0);
        table = new JTable(model);
        table.setRowHeight(25);
        table.setBackground(Color.WHITE);
        table.setFont(new Font("Segoe UI", Font.PLAIN, 14));
        table.getTableHeader().setFont(new Font("Segoe UI", Font.BOLD, 15));
        table.getTableHeader().setBackground(headerColor);
        table.getTableHeader().setForeground(Color.WHITE);

        add(new JScrollPane(table), BorderLayout.CENTER);

        // ---- Bottom Panel ----
        JPanel bottomPanel = new JPanel(new FlowLayout(FlowLayout.RIGHT));
        bottomPanel.setBackground(bgColor);

        lblGrandTotal = new JLabel("Grand Total: ₹0.00");
        lblGrandTotal.setFont(new Font("Segoe UI", Font.BOLD, 18));
        lblGrandTotal.setForeground(new Color(39, 174, 96));
        bottomPanel.add(lblGrandTotal);
```

```
        add(bottomPanel, BorderLayout.SOUTH);

        // ---- Button Actions ----
        btnAdd.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                addProductToBill();
            }
        });

        btnPrint.addActionListener(e ->
                JOptionPane.showMessageDialog(this, "🖨 Print functionality
coming soon!")
        );
    }

    private void addProductToBill() {
        String productName = txtProductName.getText().trim();
        String qtyText = txtQuantity.getText().trim();

        if (productName.isEmpty() || qtyText.isEmpty()) {
            JOptionPane.showMessageDialog(this, "Please enter product name and
quantity!");
            return;
        }

        int enteredQty;
        try {
            enteredQty = Integer.parseInt(qtyText);
        } catch (NumberFormatException e) {
            JOptionPane.showMessageDialog(this, "Quantity must be a number!");
            return;
        }

        try (Connection conn = DBConnection.getConnection()) {
            String query = "SELECT quantity, price FROM products WHERE name =
?";
            PreparedStatement ps = conn.prepareStatement(query);
            ps.setString(1, productName);
```

```
            ResultSet rs = ps.executeQuery();

            if (rs.next()) {
                int availableQty = rs.getInt("quantity");
                double price = rs.getDouble("price");

                if (availableQty >= enteredQty) {
                    double total = price * enteredQty;
                    grandTotal += total;

                    model.addRow(new Object[]{
                            model.getRowCount() + 1,
                            productName,
                            enteredQty,
                            price,
                            total
                    });

                    lblGrandTotal.setText("Grand Total: ₹" +
String.format("%.2f", grandTotal));

                    // Update remaining quantity in DB
                    String update = "UPDATE products SET quantity = quantity -
? WHERE name = ?";
                    PreparedStatement psUpdate = conn.prepareStatement(update);
                    psUpdate.setInt(1, enteredQty);
                    psUpdate.setString(2, productName);
                    psUpdate.executeUpdate();

                    // Clear input fields
                    txtProductName.setText("");
                    txtQuantity.setText("");
                    txtProductName.requestFocus();

                } else {
                    JOptionPane.showMessageDialog(this,
                            "Only " + availableQty + " items available. Out of
stock!");
                }
```

```
            } else {
                JOptionPane.showMessageDialog(this, "Product not found!");
            }

        } catch (SQLException ex) {
            JOptionPane.showMessageDialog(this, "Database error: " +
ex.getMessage());
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new BillingFrame().setVisible(true));
    }
}
```

CHAPTER 5

SCREEN SHOTS

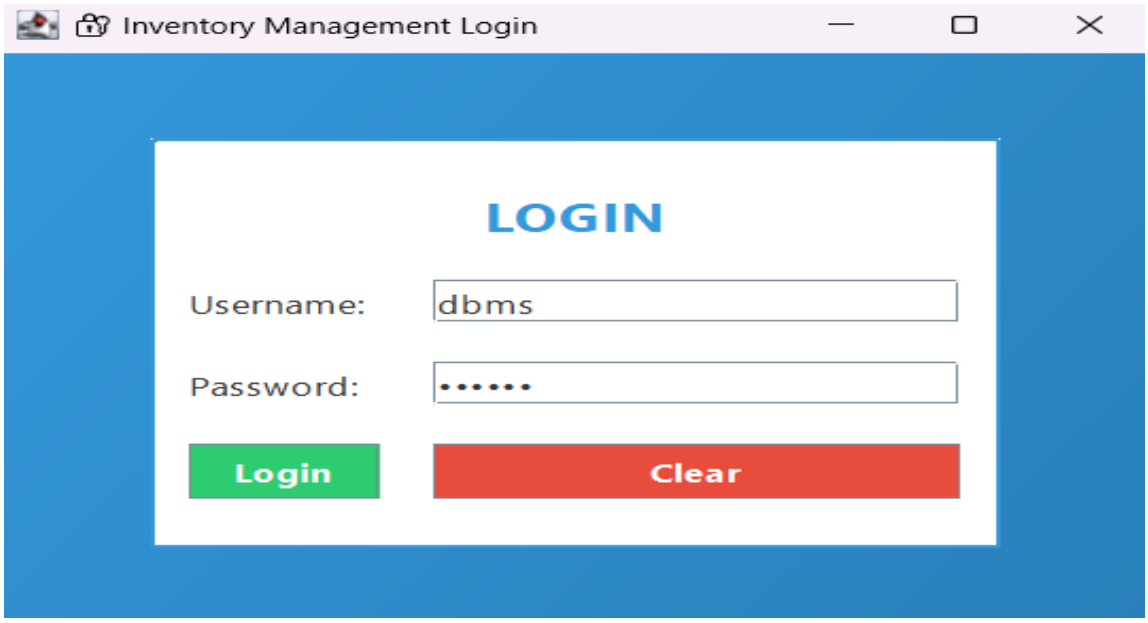

Fig 5.1 Login page

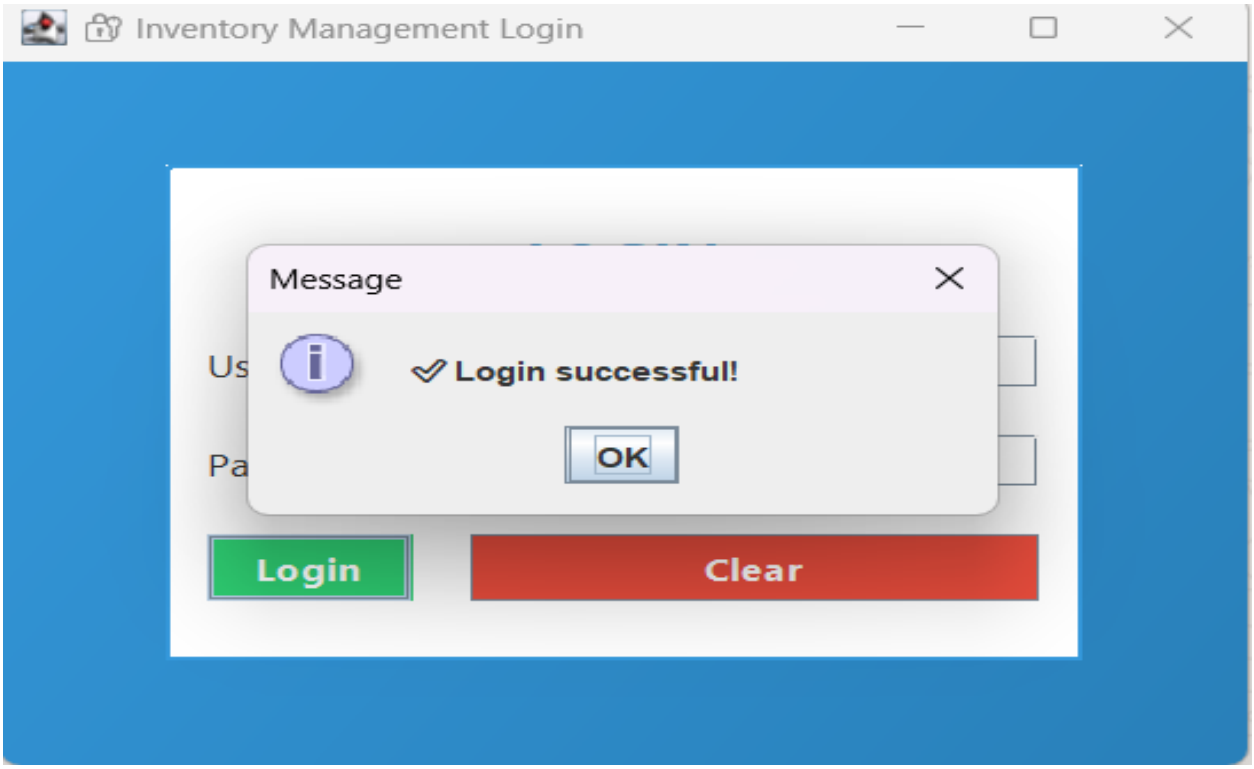

Fig 5.2 Login page

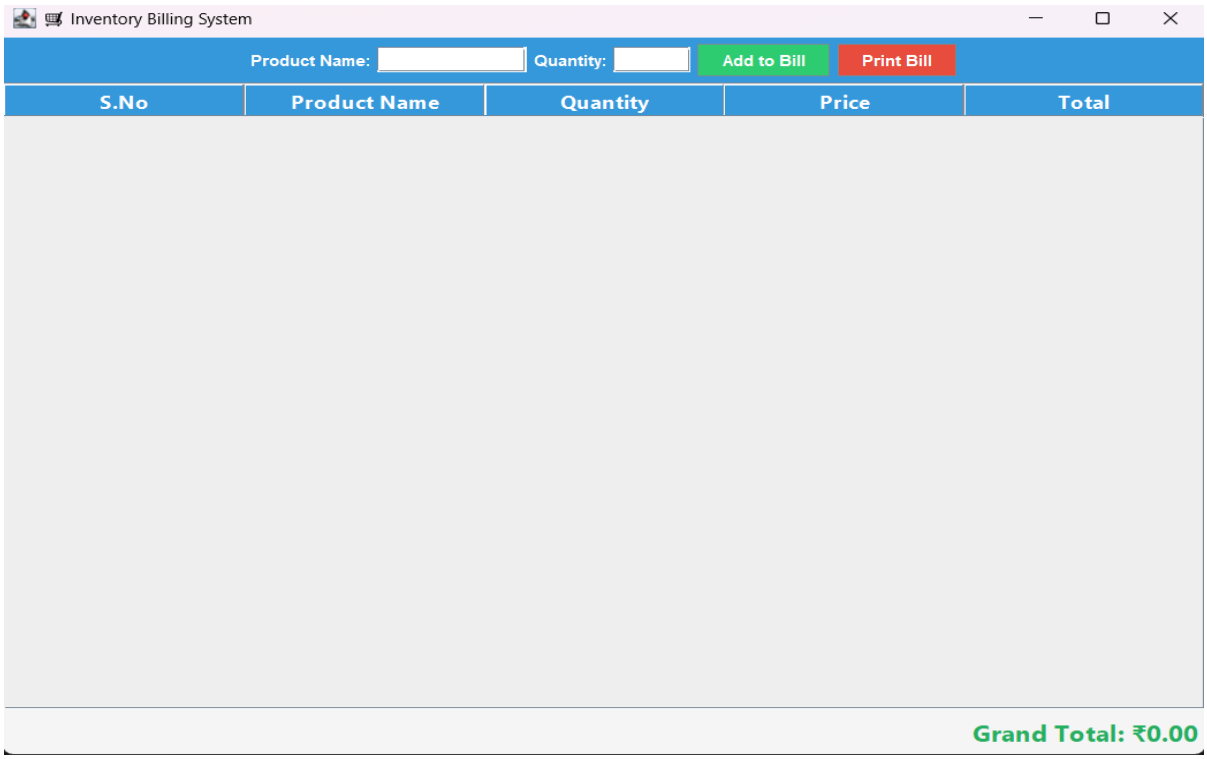

Fig 5.3 Billing page

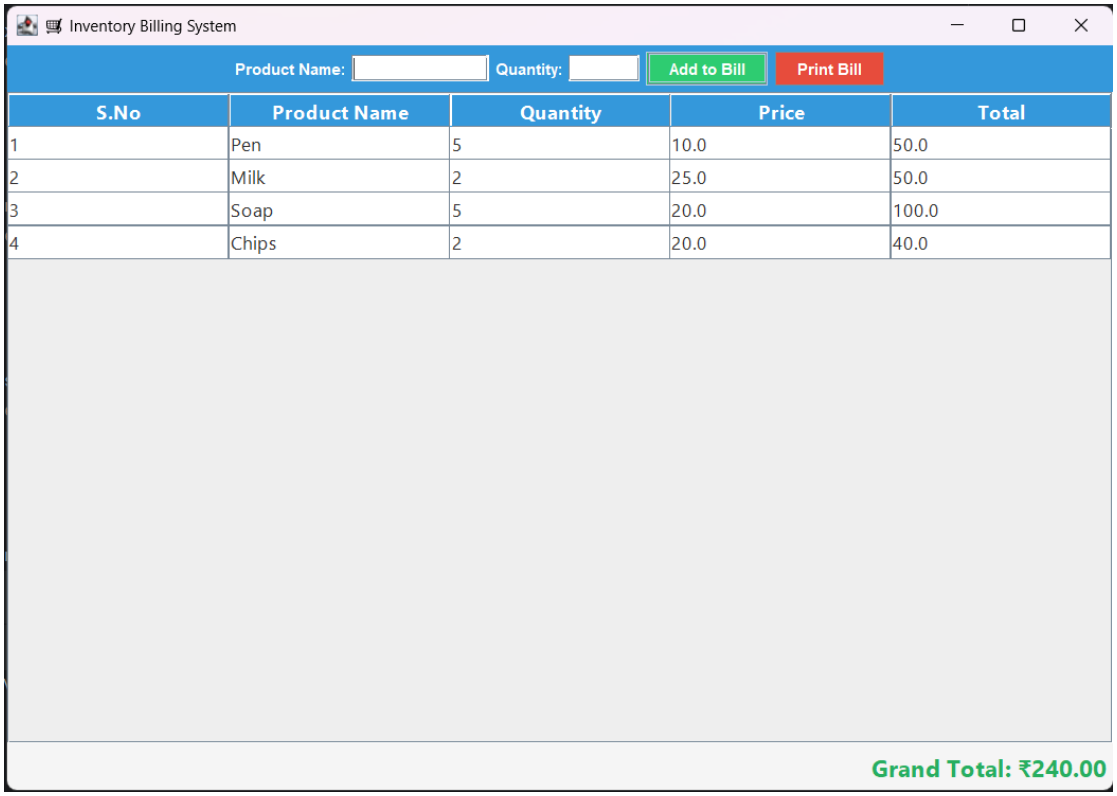

Fig 5.4 Billing page

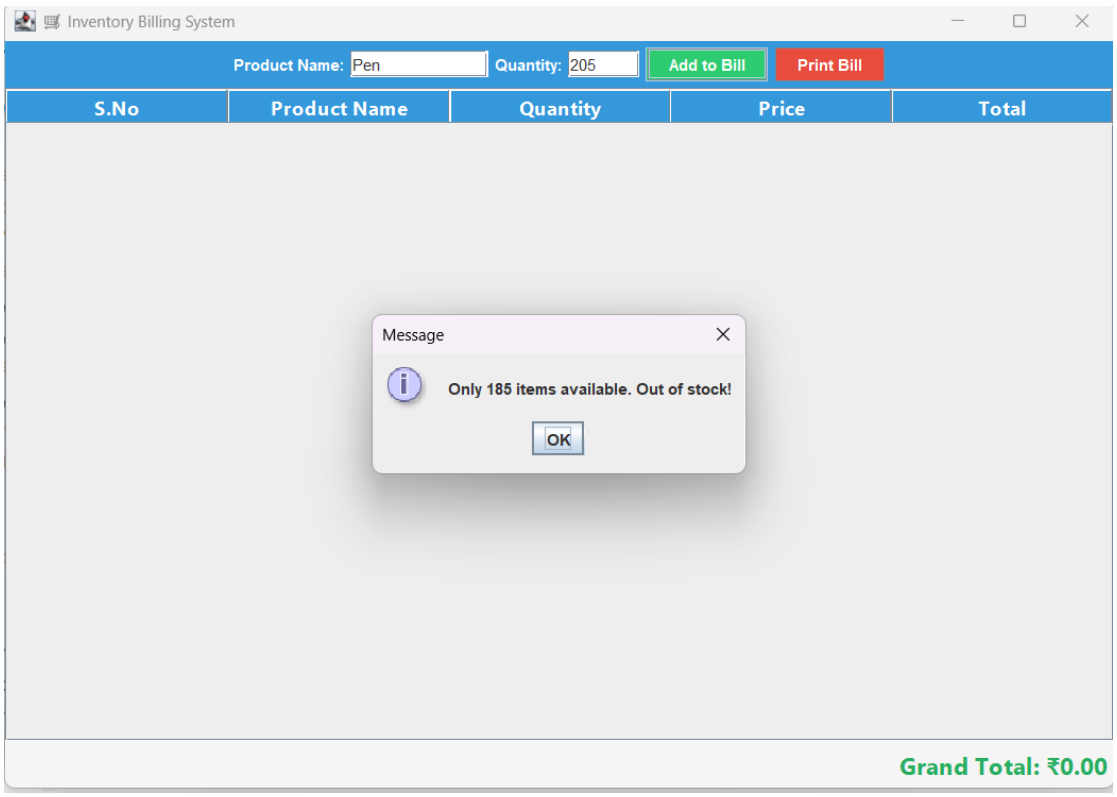

Fig 5.5 Out of stock checking

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

This Inventory Management System provides a reliable and efficient solution for retail shops to manage their stock and billing processes. It reduces manual work and errors, providing real-time product availability updates.

Future Enhancements:

- Integration of barcode scanning for faster billing.
- Generating sales reports and analytics.
- Adding multi-user roles (admin, cashier).
- Cloud-based database for remote shop access.

REFERENCES

1. https://www.w3schools.com/java/
2. https://www.mysql.com/
3. https://www.geeksforgeeks.org/jdbc/
4. https://docs.oracle.com/javase/tutorial/