OOPS WITH JAVA

CS23333

# MINIPROJECT-GROCERY MANAGEMENT SYSTEM

NAME:K.K.SWETHA

ROLLNO:231401115
CLASS:CSBS-‘B’

# ABSTRACT

This project develops a Grocery Management System using Java Swing to streamline product management for grocery store owners. The system provides a user-friendly interface for managing inventory details such as product name, price, discount, stock, and expiry date. It consists of three key components: a secure login for owner authentication, a dynamic product display panel, and an add product panel for database management.

The system uses Java Swing for the GUI and JDBC for database connectivity, storing product data in a MySQL database. Features include real-time inventory updates, input validation, and seamless navigation between different sections of the application. This project aims to reduce manual errors and improve operational efficiency for grocery store owners.

By applying object-oriented programming principles and database integration, the system addresses real-world inventory management challenges and provides an intuitive solution for retail operations.

# TABLE OF CONTENTS

# INTRODUCTION

## 1.1 INTRODUCTION

The Grocery Management System is a Java Swing-based application designed to simplify inventory management for grocery stores. It allows store owners to manage product details like stock, pricing, and expiry dates, with data stored in a MySQL database. The system provides a user-friendly interface to streamline daily operations and enhance efficiency.

## 1.2 IMPLEMENTATION

The **GROCERY MANAGEMENT SYSTEM** project discussed here is implemented using the concepts of **JAVA SWINGS** and **MYSQL**.

## 1.3 SCOPE OF THE PROJECT

The website is designed in a way where students will have to register on the website by creating an account for themselves in order for the students to get all their data in one place, where everything is organized for them.Thus saving them time and giving a sense of professionalism.

## 1.4 WEBSITE FEATURES

- Login Page
- Product Dashboard
- Add New Product
- Edit Product Details
- Database Integration

# SYSTEM SPECIFICATIONS

## 2.1 HARDWARE SPECIFICATIONS:

| | | |
|---|---|---|
| PROCESSOR | : | Intel i5 |
| MEMORY SIZE | : | 4GB(Minimum) |
| HARD DISK | : | 500 GB of free space |

## 2.2 SOFTWARE SPECIFICATIONS:

| | | |
|---|---|---|
| PROGRAMMING LANGUAGE | : | Java, MySQL |
| FRONT-END | : | Java |
| BACK-END | : | MySQL |
| OPERATING SYSTEM | : | Windows 10 |

## SAMPLE CODE

### PROGRAM CODE

```
public class GroceryManagementSystem extends JFrame {
    private Connection connection;
    private CardLayout cardLayout;
    private JPanel mainPanel;
    private JTextField nameField, mrpField, discountField, stockField, expiryField;

    public GroceryManagementSystem() {
        try {
            connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/grocerydb",
"root", "password");

            cardLayout = new CardLayout();
            mainPanel = new JPanel(cardLayout);
            add(mainPanel);
            mainPanel.add(createLoginPanel(), "Login");
            mainPanel.add(createProductDisplayPanel(), "ProductDisplay");
            mainPanel.add(createAddProductPanel(), "AddProduct");
        } catch (SQLException e) {
            JOptionPane.showMessageDialog(this, "Database connection failed: " +
e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
            e.printStackTrace();
        }
    }

    private JPanel createLoginPanel() {
        JPanel loginPanel = new JPanel();
        loginPanel.setLayout(new GridLayout(3, 2));
        loginPanel.add(new JLabel("Username:"));
        JTextField usernameField = new JTextField();
        loginPanel.add(usernameField);
        loginPanel.add(new JLabel("Password:"));
        JPasswordField passwordField = new JPasswordField();
        loginPanel.add(passwordField);

        JButton loginButton = new JButton("Login");
        loginButton.addActionListener(e -> {
            String username = usernameField.getText();
            String password = new String(passwordField.getPassword());
            if (username.equals("owner") && password.equals("password")) {
                cardLayout.show(mainPanel, "ProductDisplay");
            } else {
                JOptionPane.showMessageDialog(this, "Invalid credentials", "Error",
```

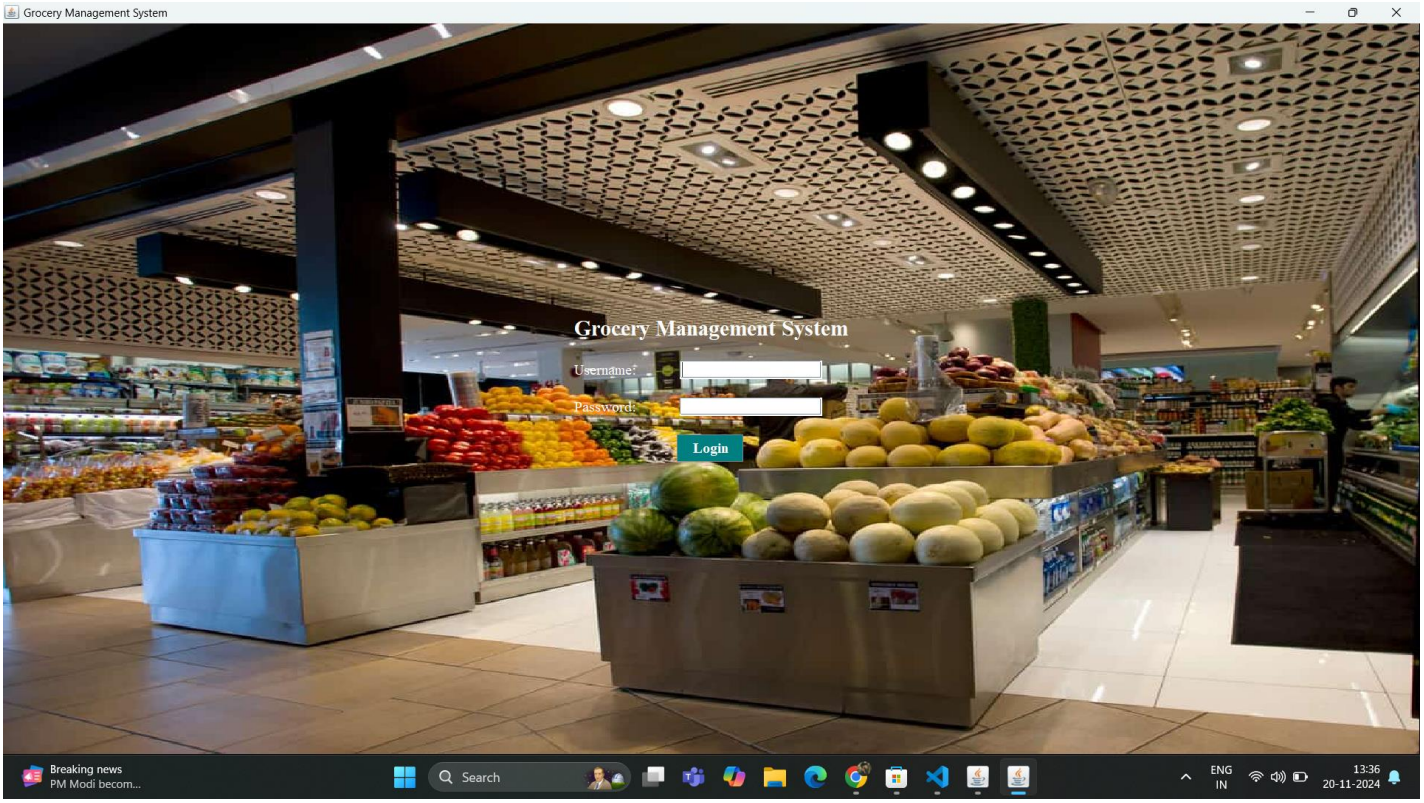
```
JOptionPane.ERROR_MESSAGE);
            }
        });
        loginPanel.add(loginButton);
        return loginPanel;
    }

    private JPanel createProductDisplayPanel() {
        JPanel productDisplayPanel = new JPanel();
        productDisplayPanel.setLayout(new BorderLayout());
        JTable productTable = new JTable(new DefaultTableModel(
            new Object[]{"ID", "Name", "MRP", "Discount", "Stock", "Expiry Date"}, 0
        ));

        JScrollPane scrollPane = new JScrollPane(productTable);
        productDisplayPanel.add(scrollPane, BorderLayout.CENTER);

        JButton addProductButton = new JButton("Add Product");
        addProductButton.addActionListener(e -> cardLayout.show(mainPanel, "AddProduct"));
        productDisplayPanel.add(addProductButton, BorderLayout.SOUTH);

        refreshProductTable(productTable);
        return productDisplayPanel;
    }

    private JPanel createAddProductPanel() {
        JPanel addProductPanel = new JPanel();
        addProductPanel.setLayout(new GridLayout(6, 2));
        addProductPanel.add(new JLabel("Product Name:"));
        nameField = new JTextField();
        addProductPanel.add(nameField);
        addProductPanel.add(new JLabel("MRP:"));
        mrpField = new JTextField();
        addProductPanel.add(mrpField);
        addProductPanel.add(new JLabel("Discount:"));
        discountField = new JTextField();
        addProductPanel.add(discountField);
        addProductPanel.add(new JLabel("Stock:"));
        stockField = new JTextField();
        addProductPanel.add(stockField);
        addProductPanel.add(new JLabel("Expiry Date (yyyy-mm-dd):"));
        expiryField = new JTextField();
        addProductPanel.add(expiryField);

        JButton addButton = new JButton("Add Product");
        addButton.addActionListener(e -> {
            try {
```

```
            String name = nameField.getText();
            double mrp = Double.parseDouble(mrpField.getText());
            double discount = Double.parseDouble(discountField.getText());
            int stock = Integer.parseInt(stockField.getText());
            String expiryDate = expiryField.getText();

            String insertQuery = "INSERT INTO products (name, mrp, discount, stock,
expiry_date) VALUES (?, ?, ?, ?, ?)";
            PreparedStatement ps = connection.prepareStatement(insertQuery);
            ps.setString(1, name);
            ps.setDouble(2, mrp);
            ps.setDouble(3, discount);
            ps.setInt(4, stock);
            ps.setString(5, expiryDate);
            ps.executeUpdate();

            JOptionPane.showMessageDialog(addProductPanel, "Product added successfully!",
"Success", JOptionPane.INFORMATION_MESSAGE);
            refreshProductTable(productTable);
            cardLayout.show(mainPane, "ProductDisplay");
        } catch (SQLException ex) {
            ex.printStackTrace();
            JOptionPane.showMessageDialog(addProductPanel, "Error adding product!",
"Error", JOptionPane.ERROR_MESSAGE);
        } catch (NumberFormatException ex) {
            JOptionPane.showMessageDialog(addProductPanel, "Invalid input format", "Error",
JOptionPane.ERROR_MESSAGE);
        }
      });
      addProductPanel.add(addButton);
      return addProductPanel;
   }

   private void refreshProductTable(JTable table) {
      try {
         String query = "SELECT * FROM products";
         PreparedStatement ps = connection.prepareStatement(query);
         ResultSet rs = ps.executeQuery();
         DefaultTableModel tableModel = (DefaultTableModel) table.getModel();
         tableModel.setRowCount(0);

         while (rs.next()) {
            Object[] row = {
               rs.getInt("id"),
               rs.getString("name"),
               rs.getDouble("mrp"),
               rs.getDouble("discount"),
```

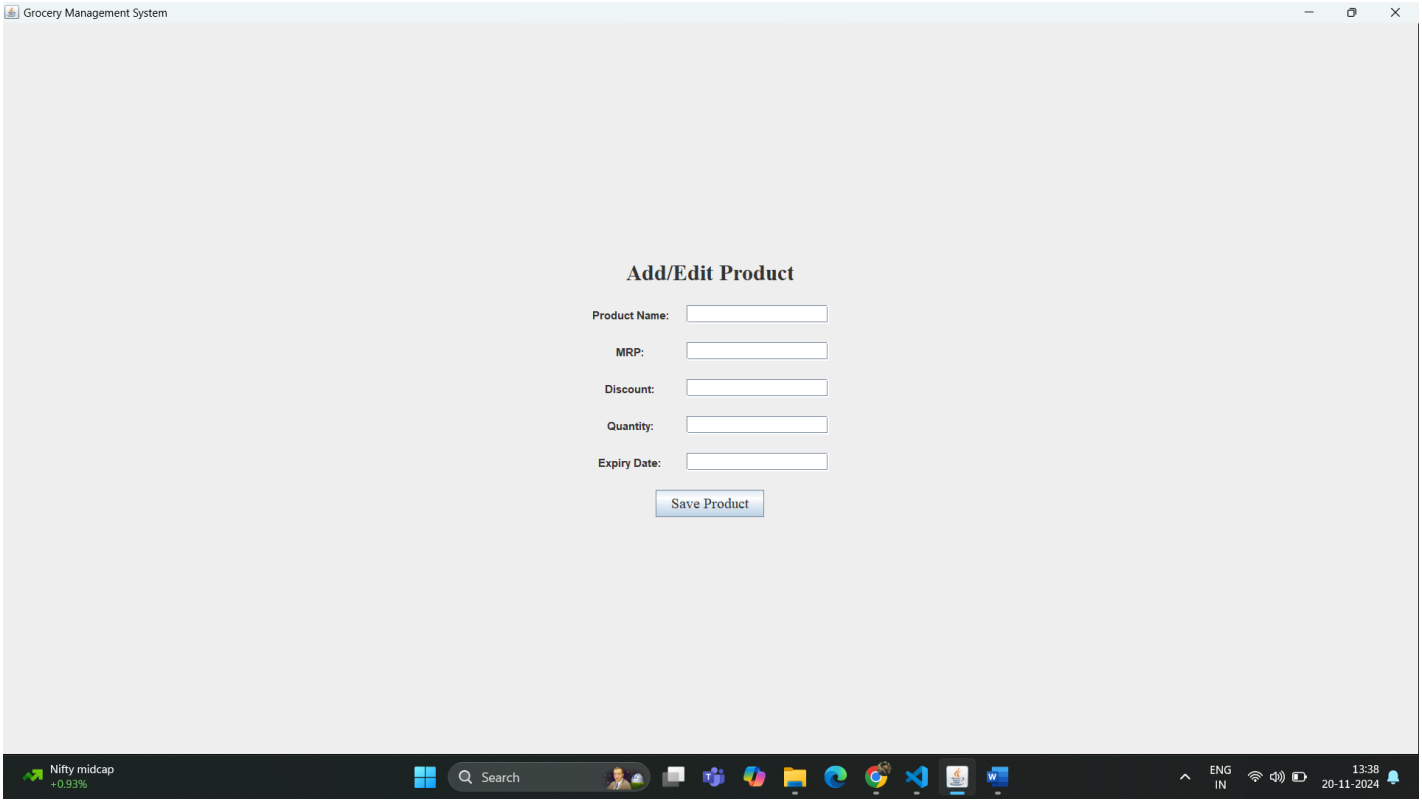
```
                rs.getInt("stock"),
                rs.getDate("expiry_date")
            };
            tableModel.addRow(row);
        }
    } catch (SQLException e) {
        JOptionPane.showMessageDialog(this, "Error loading data from database: " +
e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
        e.printStackTrace();
    }
  }

  public static void main(String[] args) {
    SwingUtilities.invokeLater(() -> {
        GroceryManagementSystem app = new GroceryManagementSystem();
        app.setTitle("Grocery Management System");
        app.setSize(800, 600);
        app.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        app.setVisible(true);
    });
  }
}
```

# SNAPSHOTS

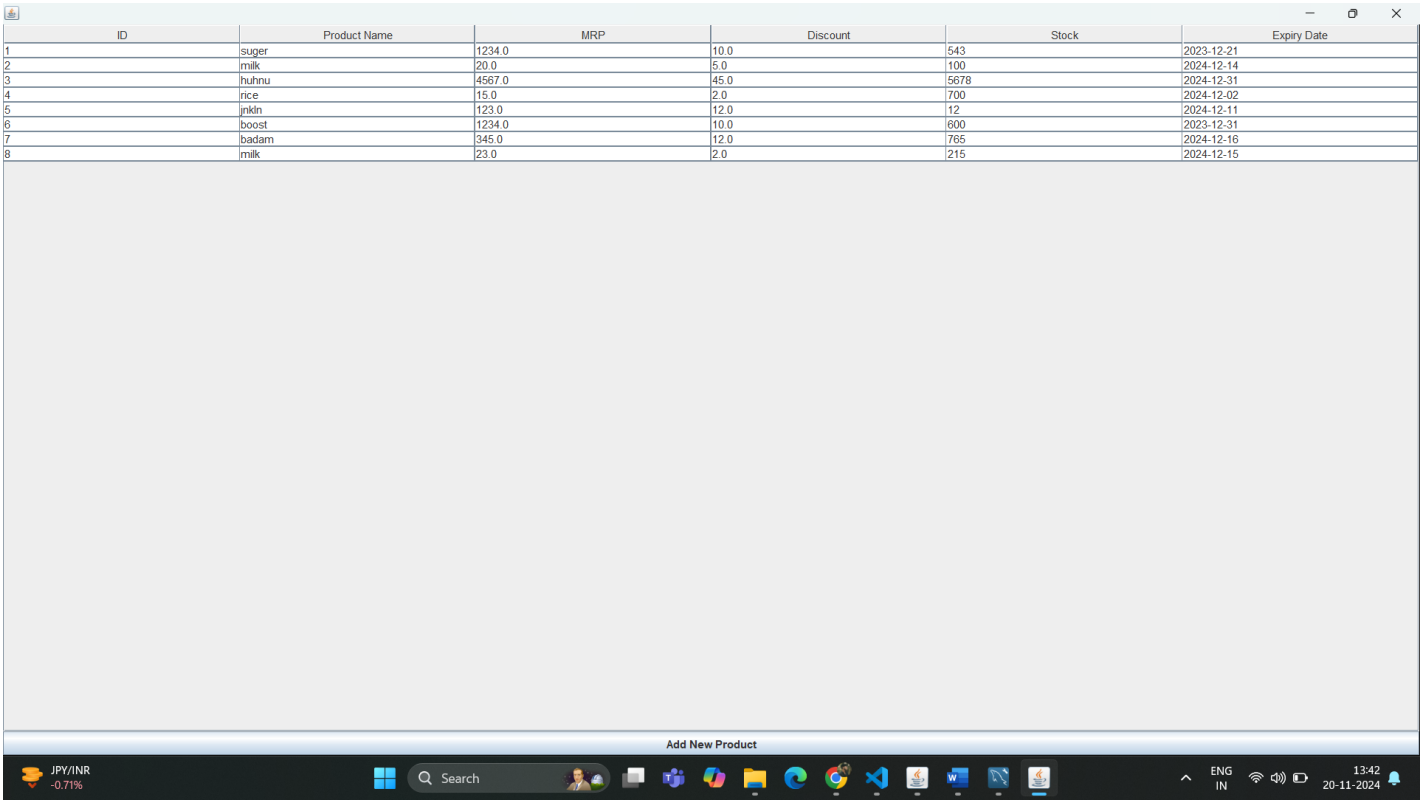
## 4.1 LOGIN PAGE




## UPDATING PRODUCTS



# 4.2 PRODUCTS LIST



## CONCLUSION

the design and backend implementation of the grocery management system using Java Swing and JDBC provide a solid foundation for building a user-friendly and functional application. The registration and login pages ensure secure user authentication, while the dashboard offers easy navigation for managing products and other key features. By integrating frontend components with a MySQL database, the system becomes scalable and efficient. Overall, this system serves as a starting point for a fully-fledged grocery management application that can be expanded with additional functionalities as needed.

## REFERENCES


1. **https://www.javatpoint.com/java-tutorial**
2. https://www.wikipedia.org/
3. https://www.w3schools.com/sql/