**NALLA MALLA REDDY**

**ENGINEERING COLLEGE**

**DOCUMENT**

**ON**

**Inventory management**

**system using Java Swing**

**Name:J.JAIPAL REDDY**

**Roll No:22B61A0586**

**Section:CSE-3B**

**Project Title: Inventory management system using Java Swing for the GUI**

**Creating an inventory management system using Java Swing for the GUI is a great project! Below is an outline of how to approach this, along with some sample code snippets to get you started.**

**Step 1: Project Setup**

**1.Initialize Git Repository**:

•Create a new directory for your project.

•Run git init to initialize a new Git repository.

•Use git add . and git commit -m "Initial commit" to track your changes.

**2.Set Up Java Project**:

•Create a new Java project in your favorite IDE (e.g., IntelliJ IDEA, Eclipse).

•Add the necessary libraries for Java Swing (usually included with the JDK).

**Step 2: Define Requirements**

Before coding, define the basic features your inventory management system will have. Common features might include:

•Add new items

•Edit existing items

•Delete items

•Search items

•View inventory list

**Step 3: Create GUI with Java Swing**

Create a simple GUI with Java Swing. Here’s an example of how to set up a basic inventory management system interface.

**Sample Code for GUI**

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class InventoryManager extends JFrame {

private JTable inventoryTable;

private DefaultTableModel tableModel;

private JTextField nameField, quantityField, priceField;

public InventoryManager() {

// Set up the main window

setTitle("Inventory Management System");

setSize(600, 400);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setLocationRelativeTo(null);

// Set up the table model and table

tableModel = new DefaultTableModel(new String[]{"Name", "Quantity", "Price"}, 0);

inventoryTable = new JTable(tableModel);

// Set up input fields and labels

JPanel inputPanel = new JPanel(new GridLayout(4, 2, 5, 5));

inputPanel.add(new JLabel("Item Name:"));

nameField = new JTextField();

inputPanel.add(nameField);

inputPanel.add(new JLabel("Quantity:"));

quantityField = new JTextField();

inputPanel.add(quantityField);

inputPanel.add(new JLabel("Price:"));

priceField = new JTextField();

inputPanel.add(priceField);

// Set up buttons

JButton addButton = new JButton("Add Item");

JButton deleteButton = new JButton("Delete Selected");

inputPanel.add(addButton);

inputPanel.add(deleteButton);

// Add components to the main layout

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

add(inputPanel, BorderLayout.SOUTH);

// Add action listeners for buttons

addButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

addItem();

}

});

deleteButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

deleteSelectedItem();

}

});

}

private void addItem() {

String name = nameField.getText();

String quantityStr = quantityField.getText();

String priceStr = priceField.getText();

if (name.isEmpty() || quantityStr.isEmpty() || priceStr.isEmpty()) {

JOptionPane.showMessageDialog(this, "Please fill in all fields", "Error", JOptionPane.ERROR\_MESSAGE);

return;

}

try {

int quantity = Integer.parseInt(quantityStr);

double price = Double.parseDouble(priceStr);

tableModel.addRow(new Object[]{name, quantity, price});

clearFields();

} catch (NumberFormatException e) {

JOptionPane.showMessageDialog(this, "Invalid quantity or price", "Error", JOptionPane.ERROR\_MESSAGE);

}

}

private void deleteSelectedItem() {

int selectedRow = inventoryTable.getSelectedRow();

if (selectedRow != -1) {

tableModel.removeRow(selectedRow);

} else {

JOptionPane.showMessageDialog(this, "Please select an item to delete", "Error", JOptionPane.ERROR\_MESSAGE);

}

}

private void clearFields() {

nameField.setText("");

quantityField.setText("");

priceField.setText("");

}

public static void main(String[] args) {

SwingUtilities.invokeLater(() -> {

InventoryManager manager = new InventoryManager();

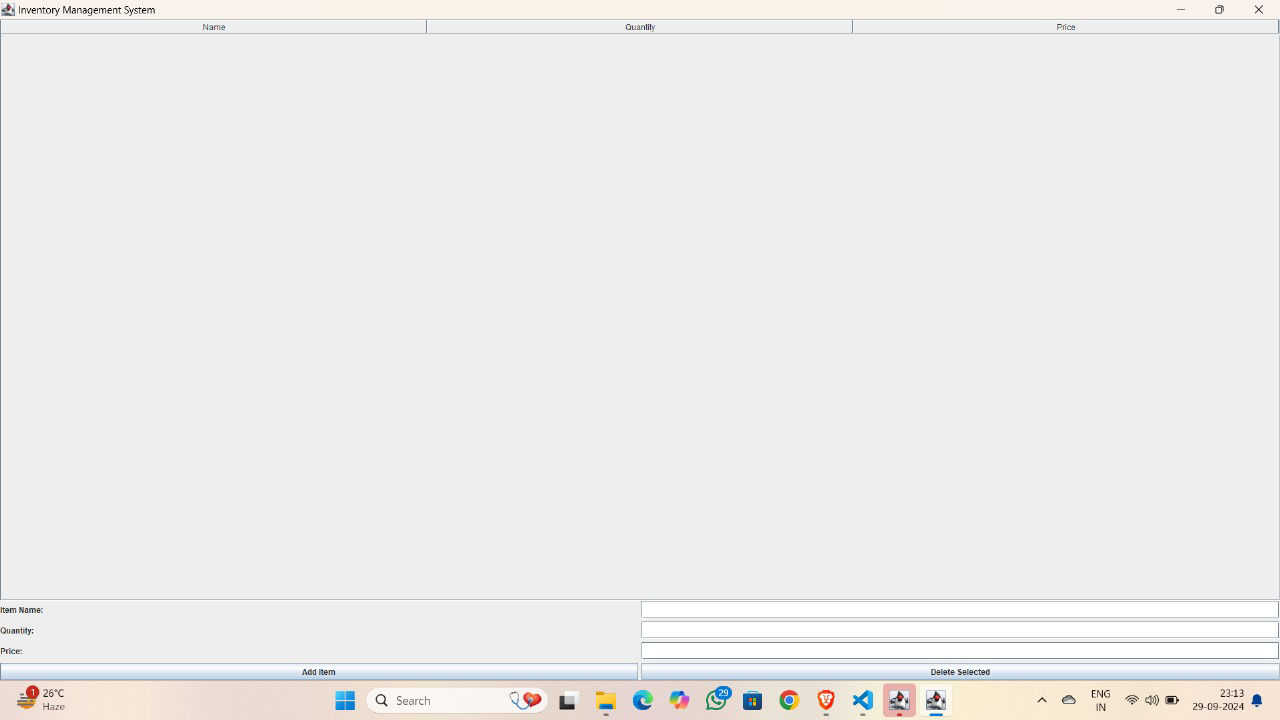
manager.setVisible(true);

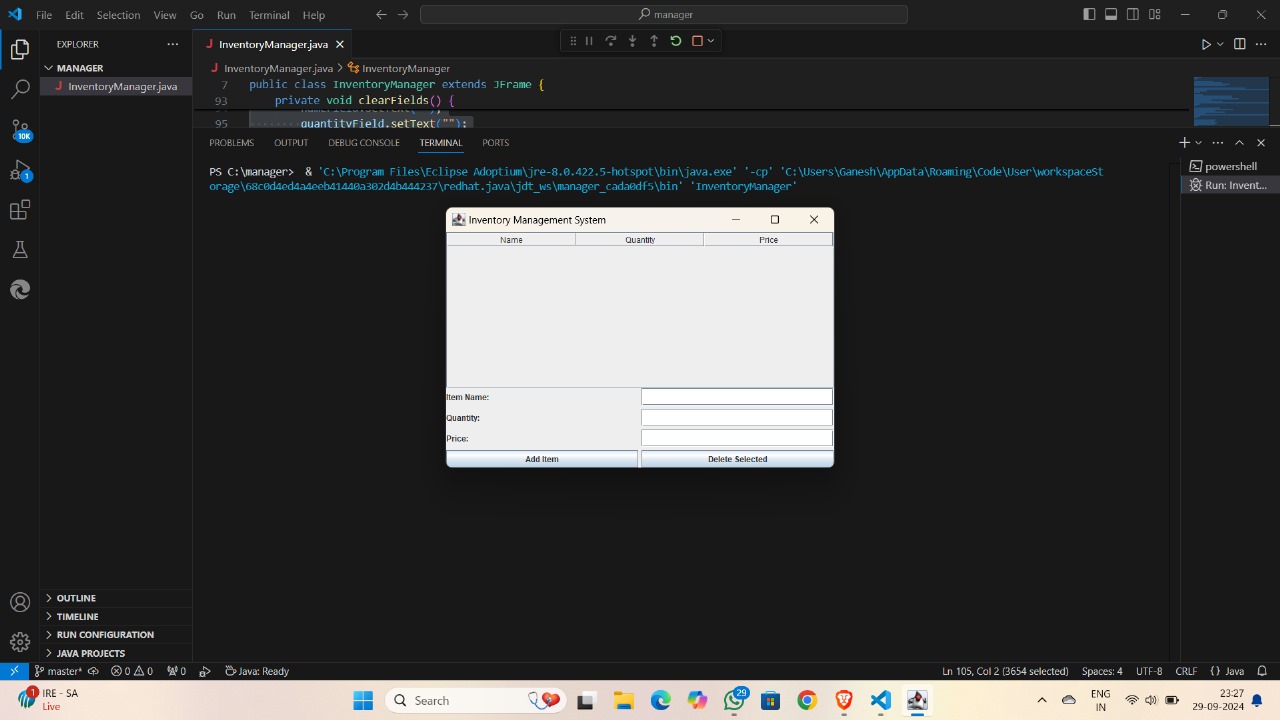
});

}

}

**OUTPUT:**

****

****

**CONCLUSION:**

By following these steps, we have developed a basic Inventory Management System using Java Swing for the GUI and Git for version control. The project demonstrates key programming concepts such as object-oriented design (Product class), data management (ArrayList), event handling (ActionListeners in Swing), and working with a version control system (Git).