A logo of a university

Description automatically generated**COMSATS University Islamabad Sahiwal Campus**

**(Department of Computer Sciences)**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Course Title:** | Object Oriented Programing | | | | **Course Code:** | | CSC241 | | **Credit Hours:** | 4(3,1) |
| **Course Instructor:** | Ali Sher Kashif | | | | **Programme Name:** | | | BS (CS) | | |
| **Semester:** | 3rd | **Batch:** | FA23 | **Section:** | E | | | **Date:** | 23-12-2024 | |
| **Time Allowed:** | 180 Minutes | | | | **Maximum Marks:** | | | | 50 | |
| **Student’s Name:** | HABIB UR REHMAN | | | | **Reg. No.** | CUI/ FA23-BCS-251 /SWL | | | | |

**Final Lab Exam**

**Question**

You are required to develop a Generalized Management System that can be applied to various real-world domains such as a Library System, Hotel Management System, University Management System, or Inventory System. The system should be capable of managing the following entities:

* Primary Entity: For example, Books in a Library, Guests in a Hotel, Students in a University, or Products in an Inventory. Each entity should have unique attributes like ID, name, and specific properties related to the chosen domain.
* Secondary Entity: For example, Authors for Books, Rooms for Guests, Courses for Students, or Categories for Products.
* Relationships: Demonstrate relationships between the primary and secondary entities using appropriate OOP principles

**Code Solution:**

**Main Class**

import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ComponentAdapter;  
import java.awt.event.ComponentEvent;  
  
  
class MainPage extends JFrame {  
  
 public MainPage() {  
 setTitle("Main Page");  
 setExtendedState(JFrame.*MAXIMIZED\_BOTH*);  
 setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 setLocationRelativeTo(null);  
 setLayout(new BorderLayout());  
  
 JPanel sidebarPanel = new JPanel();  
 sidebarPanel.setLayout(new BoxLayout(sidebarPanel, BoxLayout.*Y\_AXIS*));  
 sidebarPanel.setBackground(new Color(112,128,144));  
 sidebarPanel.setPreferredSize(new Dimension(200, getHeight()));  
  
  
 // Side Buttons  
 JButton mainDashboardButton = createSidebarButton("Order Form");  
 mainDashboardButton.addActionListener(e -> openMainDashboard());  
  
 JButton addProductButton = createSidebarButton("Add Product");  
 addProductButton.addActionListener(e -> addProduct());  
  
 JButton productsListButton = createSidebarButton("Products List");  
 productsListButton.addActionListener(e -> showProductsList());  
  
  
 JButton ordersButton = createSidebarButton("Orders");  
 ordersButton.addActionListener(e -> showOrders());  
  
  
 sidebarPanel.add(Box.*createVerticalStrut*(10));  
 sidebarPanel.add(mainDashboardButton);  
 sidebarPanel.add(Box.*createVerticalStrut*(10));  
 sidebarPanel.add(addProductButton);  
 sidebarPanel.add(Box.*createVerticalStrut*(10));  
 sidebarPanel.add(productsListButton);  
 sidebarPanel.add(Box.*createVerticalStrut*(10));  
 sidebarPanel.add(ordersButton);  
 sidebarPanel.add(Box.*createVerticalStrut*(10));  
  
  
  
 JPanel contentPanel = new JPanel();  
 contentPanel.setLayout(new BorderLayout());  
 contentPanel.setBackground(new Color(245, 245, 245));  
  
 JPanel northContainer = new JPanel();  
 northContainer.setLayout(new BorderLayout());  
  
 JPanel headerPanel = new JPanel(new BorderLayout());  
 headerPanel.setBackground(new Color(54,69,79));  
 headerPanel.setPreferredSize(new Dimension(getWidth(), 60));  
  
 JPanel coloredLinePanel = new JPanel();  
 coloredLinePanel.setPreferredSize(new Dimension(0, 10));  
 coloredLinePanel.setBackground(new Color(255,165,0));  
 northContainer.add(headerPanel, BorderLayout.*CENTER*);  
 northContainer.add(coloredLinePanel, BorderLayout.*SOUTH*);  
  
 JLabel headerLabel = new JLabel(" Sadar Marble and Granite", SwingConstants.*LEFT*);  
 headerLabel.setForeground(Color.*WHITE*);  
 headerLabel.setFont(new Font("Segoe UI", Font.*BOLD*, 28));  
 headerPanel.add(headerLabel, BorderLayout.*CENTER*)  
 JPanel southContainer = new JPanel(new BorderLayout());  
 southContainer.setBackground(new Color(54, 69, 79));  
 southContainer.setPreferredSize(new Dimension(getWidth(), 70));  
  
 // Orange line above the bottom panel  
 JPanel coloredLineBottom = new JPanel();  
 coloredLineBottom.setPreferredSize(new Dimension(0, 5)); // Height of orange line  
 coloredLineBottom.setBackground(new Color(255, 165, 0));  
  
 // Centered bottom label  
 JLabel bottomLabel = new JLabel("Chiniot Road, Near Mian Filling Station, Jhumra City");  
 bottomLabel.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bottomLabel.setForeground(Color.*WHITE*);  
 bottomLabel.setFont(new Font("Segoe UI", Font.*BOLD*, 22));  
  
 southContainer.add(coloredLineBottom, BorderLayout.*NORTH*); // Orange line  
 southContainer.add(bottomLabel, BorderLayout.*CENTER*);  
  
  
 ImageIcon originalIcon = new ImageIcon("src/img.jpg");  
 JLabel imageLabel = new JLabel();  
 imageLabel.setHorizontalAlignment(SwingConstants.*CENTER*);  
 imageLabel.setVerticalAlignment(SwingConstants.*CENTER*);  
  
 contentPanel.addComponentListener(new ComponentAdapter() {  
 @Override  
 public void componentResized(ComponentEvent e) {  
 // Get the panel's current size  
 Dimension size = contentPanel.getSize();  
  
 // Scale the image to fit the panel  
 Image scaledImage = originalIcon.getImage()  
 .getScaledInstance(size.width, size.height, Image.*SCALE\_SMOOTH*);  
  
 // Update the label with the new scaled image  
 ImageIcon scaledIcon = new ImageIcon(scaledImage);  
 imageLabel.setIcon(scaledIcon);  
 }  
 });  
  
  
 // Add the image label to the center of the BorderLayout  
 contentPanel.add(imageLabel, BorderLayout.*CENTER*);  
 contentPanel.add(northContainer, BorderLayout.*NORTH*);  
 contentPanel.add(southContainer, BorderLayout.*SOUTH*);  
  
  
 // Divider  
 JSplitPane splitPane = new JSplitPane(JSplitPane.*HORIZONTAL\_SPLIT*, sidebarPanel, contentPanel);  
 splitPane.setDividerSize(0);  
 splitPane.setEnabled(false);  
 splitPane.setDividerLocation(200);  
 add(splitPane, BorderLayout.*CENTER*);  
 setVisible(true);  
 }  
 private JButton createSidebarButton(String label) {  
 JButton button = new JButton(label);  
 button.setMaximumSize(new Dimension(200, 40));  
 button.setFocusPainted(false);  
 button.setBorderPainted(false);  
 button.setBackground(new Color(130,148,164));  
 button.setForeground(Color.*WHITE*);  
 Font f = new Font("Arial",Font.*BOLD*,14);  
 button.setFont(f);  
 button.setAlignmentX(Component.*CENTER\_ALIGNMENT*);  
 button.setCursor(new Cursor(Cursor.*HAND\_CURSOR*));  
 return button;  
 }  
  
 private void addProduct() {  
 System.*out*.println("Add Product button clicked");  
 Product p1 = new Product();  
 p1.addProductGUI();  
  
 }  
  
 private void openMainDashboard() {  
 System.*out*.println("Main Dashboard button clicked");  
 Product.*addSampleProducts*(); // Add sample products  
 Order order = new Order();  
 order.showOrderGUI();  
 }  
  
 private void showProductsList() {  
 System.*out*.println("List button clicked");  
 Product p2 = new Product();  
 p2.displayProducts();  
 }  
 private void showOrders() {  
 System.*out*.println("Orders button clicked");  
 Order order = new Order();  
 order.displayOrderDetails();  
  
 }  
  
  
  
}  
public class Main {  
 public static void main(String[] args) {  
 MainPage p1 = new MainPage();  
  
 }  
}

**Product Class:**

import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.KeyAdapter;  
import java.awt.event.KeyEvent;  
import javax.swing.table.DefaultTableModel;  
import java.util.ArrayList;  
  
public class Product {  
 private static int *nextId* = 1;  
 private int id;  
 private String name;  
 private String category;  
 private double price;  
 private static ArrayList<Product> *productList* = new ArrayList<>();  
  
  
 private JTextField productNameField;  
 private JTextField productPriceField;  
 private JList<String> productCategoryList;  
 private JButton addButton;  
  
 public Product() {  
 System.*out*.println("Product object created");  
 }  
  
 public Product(int id, String name, String category, double price) {  
 this.id = id;  
 this.name = name;  
 this.category = category;  
 this.price = price;  
 }  
  
 public Product(String name, double price) {  
 this.name = name;  
 this.price = price;  
 }  
 public static ArrayList<Product> getProductList() {  
 return *productList*;  
 }  
 public static void addSampleProducts() {  
 *productList*.add(new Product(*nextId*++, "White Marble", "Marble", 25.5));  
 *productList*.add(new Product(*nextId*++, "Granite Tile", "Granite", 30.0));  
 *productList*.add(new Product(*nextId*++, "Kitchen Top", "Kitchen", 50.0));  
 *productList*.add(new Product(*nextId*++, "Stair Riser", "Stairs", 15.0));  
 }  
  
 public int getId() {  
 return id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public String getCategory() {  
 return category;  
 }  
  
 public double getPrice() {  
 return price;  
 }  
  
  
 public void saveProduct(String name, double price, String category) {  
 int id = *nextId*++; // Generate unique ID  
 Product newProduct = new Product(id, name, category, price);  
 *productList*.add(newProduct); // Add product to the list  
 System.*out*.println("Product saved: " + newProduct);  
 System.*out*.println("Total products: " + *productList*.size());  
 }  
  
 public void addProductGUI() {  
 JFrame frame = new JFrame("Add Product");  
 frame.setSize(500, 300);  
 frame.setDefaultCloseOperation(JFrame.*DISPOSE\_ON\_CLOSE*);  
 frame.setLocationRelativeTo(null);  
 frame.setLayout(new BorderLayout(10, 10));  
  
 JPanel headerPanel = new JPanel();  
 headerPanel.setBackground(new Color(54, 69, 79));  
 headerPanel.setLayout(new FlowLayout(FlowLayout.*CENTER*, 20, 10));  
  
 JLabel titleLabel = new JLabel("Add New Product");  
 titleLabel.setForeground(Color.*WHITE*);  
 titleLabel.setFont(new Font("Arial", Font.*BOLD*, 24));  
 headerPanel.add(titleLabel);  
  
 JPanel inputPanel = new JPanel(new GridBagLayout());  
 inputPanel.setBackground(new Color(240, 240, 240));  
 GridBagConstraints gbc = new GridBagConstraints();  
 gbc.insets = new Insets(10, 10, 10, 10);  
 gbc.fill = GridBagConstraints.*HORIZONTAL*;  
  
 JLabel nameLabel = new JLabel("Product Name:");  
 nameLabel.setFont(new Font("Arial", Font.*BOLD*, 16));  
 productNameField = new JTextField();  
 productNameField.setFont(new Font("Arial", Font.*PLAIN*, 15));  
 productNameField.setPreferredSize(new Dimension(200, 30));  
  
 JLabel priceLabel = new JLabel("Product Price:");  
 priceLabel.setFont(new Font("Arial", Font.*BOLD*, 16));  
 productPriceField = new JTextField();  
 productPriceField.setFont(new Font("Arial", Font.*PLAIN*, 15));  
 productPriceField.setPreferredSize(new Dimension(200, 30));  
  
 JLabel categoryLabel = new JLabel("Select Category:");  
 categoryLabel.setFont(new Font("Arial", Font.*BOLD*, 16));  
  
 String[] categories = {"Marble", "Granite", "Kitchen", "Stairs", "Skirting", "Tuff Tile", "Border", "Others"};  
 JComboBox<String> categoryDropdown = new JComboBox<>(categories);  
 categoryDropdown.setFont(new Font("Arial", Font.*PLAIN*, 15));  
 categoryDropdown.setBackground(Color.*WHITE*);  
  
 productNameField.addKeyListener(new KeyAdapter() {  
 @Override  
 public void keyPressed(KeyEvent e) {  
 if (e.getKeyCode() == KeyEvent.*VK\_ENTER*) {  
 productPriceField.requestFocus();  
 }  
 }  
 });  
  
 productPriceField.addKeyListener(new KeyAdapter() {  
 @Override  
 public void keyPressed(KeyEvent e) {  
 if (e.getKeyCode() == KeyEvent.*VK\_ENTER*) {  
 categoryDropdown.requestFocus();  
 }  
 }  
 });  
  
 categoryDropdown.addKeyListener(new KeyAdapter() {  
 @Override  
 public void keyPressed(KeyEvent e) {  
 if (e.getKeyCode() == KeyEvent.*VK\_ENTER*) {  
 addButton.requestFocus();  
 }  
 }  
 });  
  
 // Arrangement in Grids by Column & Rows  
 gbc.gridx = 0;  
 gbc.gridy = 0;  
 inputPanel.add(nameLabel, gbc);  
 gbc.gridx = 1;  
 gbc.gridy = 0;  
 inputPanel.add(productNameField, gbc);  
  
 gbc.gridx = 0;  
 gbc.gridy = 1;  
 inputPanel.add(priceLabel, gbc);  
 gbc.gridx = 1;  
 gbc.gridy = 1;  
 inputPanel.add(productPriceField, gbc);  
  
 gbc.gridx = 0;  
 gbc.gridy = 2;  
 inputPanel.add(categoryLabel, gbc);  
 gbc.gridx = 1;  
 gbc.gridy = 2;  
 inputPanel.add(categoryDropdown, gbc);  
  
 JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.*RIGHT*));  
 buttonPanel.setBackground(new Color(240, 240, 240));  
  
 addButton = new JButton("Add Product");  
 addButton.setBackground(new Color(39, 174, 96));  
 addButton.setForeground(Color.*WHITE*);  
 addButton.setFont(new Font("Arial", Font.*BOLD*, 14));  
 addButton.addActionListener(e -> {  
 String name = productNameField.getText();  
 String category = (String) categoryDropdown.getSelectedItem();  
 double price;  
  
 try {  
 price = Double.*parseDouble*(productPriceField.getText());  
 } catch (NumberFormatException ex) {  
 JOptionPane.*showMessageDialog*(frame,  
 "Please enter a valid number for the price",  
 "Input Error",  
 JOptionPane.*ERROR\_MESSAGE*);  
 return;  
 }  
  
 if (name.isEmpty() || category == null) {  
 JOptionPane.*showMessageDialog*(frame,  
 "Please fill in all fields and select a category",  
 "Input Error",  
 JOptionPane.*ERROR\_MESSAGE*);  
 return;  
 }  
  
 saveProduct(name, price, category);  
 frame.dispose();  
 });  
  
 addButton.addKeyListener(new KeyAdapter() {  
 @Override  
 public void keyPressed(KeyEvent e) {  
 if (e.getKeyCode() == KeyEvent.*VK\_ENTER*) {  
 addButton.doClick();  
 }  
 }  
 });  
  
 buttonPanel.add(addButton);  
 frame.add(headerPanel, BorderLayout.*NORTH*);  
 frame.add(inputPanel, BorderLayout.*CENTER*);  
 frame.add(buttonPanel, BorderLayout.*SOUTH*);  
 frame.setVisible(true);  
 }  
 public void displayProducts() {  
 JFrame frame = new JFrame("Product List");  
 frame.setSize(600, 400);  
 frame.setDefaultCloseOperation(JFrame.*DISPOSE\_ON\_CLOSE*);  
 frame.setLocationRelativeTo(null);  
  
 String[] columnNames = {"ID", "Name", "Category", "Price"};  
 DefaultTableModel tableModel = new DefaultTableModel(columnNames, 0);  
  
 for (Product product : Product.*productList*) {  
 Object[] row = {product.id, product.name, product.category, product.price};  
 tableModel.addRow(row);  
 }  
  
 JTable table = new JTable(tableModel);  
 JScrollPane scrollPane = new JScrollPane(table);  
 frame.add(scrollPane);  
  
 frame.setVisible(true);  
 }  
 @Override  
 public String toString() {  
 return "Product {" +  
 "id=" + id +  
 ", name='" + name + '\'' +  
 ", category='" + category + '\'' +  
 ", price=" + price +  
 '}';  
 }  
}

**Order Class:**

import javax.swing.\*;  
import javax.swing.table.DefaultTableModel;  
import java.awt.\*;  
import java.awt.event.\*;  
import java.util.ArrayList;  
  
public class Order {  
 private JTable productTable;  
 private JTable cartTable;  
 private JLabel totalLabel;  
 private DefaultTableModel cartTableModel;  
 private double total = 0.0;  
 private ArrayList<OrderDetail> orders = new ArrayList<>(); // Stores all orders  
  
 public void showOrderGUI() {  
 JFrame frame = new JFrame("Place Order");  
 frame.setSize(800, 600);  
 frame.setDefaultCloseOperation(JFrame.*DISPOSE\_ON\_CLOSE*);  
 frame.setLocationRelativeTo(null);  
 frame.setLayout(new BorderLayout(10, 10));  
  
 // Category Buttons  
 JPanel categoryPanel = new JPanel(new FlowLayout(FlowLayout.*LEFT*, 10, 10));  
 categoryPanel.setBackground(new Color(240, 240, 240));  
 String[] categories = {"Marble", "Granite", "Kitchen", "Stairs"};  
 for (String category : categories) {  
 JButton categoryButton = new JButton(category);  
 categoryButton.setFocusPainted(false);  
 categoryButton.setBorderPainted(false);  
 categoryButton.setBackground(new Color(255,165,0));  
 categoryButton.setForeground(Color.*BLACK*);  
 Font f = new Font("Arial",Font.*BOLD*,14);  
 categoryButton.setFont(f);  
 categoryButton.addActionListener(e -> showProductsByCategory(category));  
 categoryPanel.add(categoryButton);  
 }  
  
 // Product Table  
  
 DefaultTableModel productTableModel = new DefaultTableModel(new String[]{"ID", "Name", "Category", "Price"}, 0);  
 productTable = new JTable(productTableModel);  
 JScrollPane productScrollPane = new JScrollPane(productTable);  
  
 // Cart Table  
  
 cartTableModel = new DefaultTableModel(new String[]{"Name", "Category", "Price", "Quantity"}, 0);  
 cartTable = new JTable(cartTableModel);  
 JScrollPane cartScrollPane = new JScrollPane(cartTable);  
  
  
 // Total Label and Checkout Button  
 totalLabel = new JLabel("Total: $0.0");  
 totalLabel.setFont(new Font("Arial", Font.*BOLD*, 16));  
  
 JButton checkoutButton = new JButton("Checkout");  
 checkoutButton.setBackground(new Color(39, 174, 96));  
 checkoutButton.setForeground(Color.*WHITE*);  
 checkoutButton.setFont(new Font("Arial", Font.*BOLD*, 14));  
 checkoutButton.addActionListener(e -> checkout(frame));  
  
 JPanel bottomPanel = new JPanel(new BorderLayout());  
 bottomPanel.add(cartScrollPane, BorderLayout.*CENTER*);  
  
 JPanel totalPanel = new JPanel(new FlowLayout(FlowLayout.*RIGHT*));  
 totalPanel.add(totalLabel);  
 totalPanel.add(checkoutButton);  
 bottomPanel.add(totalPanel, BorderLayout.*SOUTH*);  
  
 // Product Table Row Selection  
 productTable.addMouseListener(new MouseAdapter() {  
 @Override  
 public void mouseClicked(MouseEvent e) {  
 int selectedRow = productTable.getSelectedRow();  
 if (selectedRow >= 0) {  
 String productName = productTable.getValueAt(selectedRow, 1).toString();  
 String productCategory = productTable.getValueAt(selectedRow, 2).toString();  
 double productPrice = Double.*parseDouble*(productTable.getValueAt(selectedRow, 3).toString());  
  
 String quantityStr = JOptionPane.*showInputDialog*(frame,  
 "Enter quantity for " + productName + ":",  
 "Quantity",  
 JOptionPane.*QUESTION\_MESSAGE*);  
  
 if (quantityStr != null) {  
 try {  
 int quantity = Integer.*parseInt*(quantityStr);  
 addToCart(productName, productCategory, productPrice, quantity);  
 } catch (NumberFormatException ex) {  
 JOptionPane.*showMessageDialog*(frame,  
 "Please enter a valid quantity.",  
 "Input Error",  
 JOptionPane.*ERROR\_MESSAGE*);  
 }  
 }  
 }  
 }  
 });  
  
 frame.add(categoryPanel, BorderLayout.*NORTH*);  
 frame.add(productScrollPane, BorderLayout.*CENTER*);  
 frame.add(bottomPanel, BorderLayout.*SOUTH*);  
 frame.setVisible(true);  
 }  
  
 private void showProductsByCategory(String category) {  
 DefaultTableModel model = (DefaultTableModel) productTable.getModel();  
 model.setRowCount(0); // Clear existing rows  
  
 for (Product product : Product.*getProductList*()) {  
 if (product.getCategory().equalsIgnoreCase(category)) {  
 model.addRow(new Object[]{product.getId(), product.getName(), product.getCategory(), product.getPrice()});  
 }  
 }  
  
 if (model.getRowCount() == 0) {  
 JOptionPane.*showMessageDialog*(null, "No products found for category: " + category,  
 "Info", JOptionPane.*INFORMATION\_MESSAGE*);  
 }  
 }  
  
 private void addToCart(String name, String category, double price, int quantity) {  
 cartTableModel.addRow(new Object[]{name, category, price, quantity});  
 total += price \* quantity;  
 totalLabel.setText("Total: $" + total);  
 }  
  
 private void checkout(JFrame frame) {  
 if (cartTableModel.getRowCount() == 0) {  
 JOptionPane.*showMessageDialog*(frame,  
 "Your cart is empty. Please add products before checking out.",  
 "Checkout Error",  
 JOptionPane.*ERROR\_MESSAGE*);  
 return;  
 }  
  
 ArrayList<OrderDetail> currentOrder = new ArrayList<>();  
 for (int i = 0; i < cartTableModel.getRowCount(); i++) {  
 String name = cartTableModel.getValueAt(i, 0).toString();  
 String category = cartTableModel.getValueAt(i, 1).toString();  
 double price = Double.*parseDouble*(cartTableModel.getValueAt(i, 2).toString());  
 int quantity = Integer.*parseInt*(cartTableModel.getValueAt(i, 3).toString());  
  
 currentOrder.add(new OrderDetail(name, category, price, quantity));  
 }  
  
 orders.add(new OrderDetail("Order #" + (orders.size() + 1), currentOrder, total));  
 JOptionPane.*showMessageDialog*(frame,  
 "Order placed successfully! Total: $" + total,  
 "Order Success",  
 JOptionPane.*INFORMATION\_MESSAGE*);  
  
 // Clear cart  
 cartTableModel.setRowCount(0);  
 total = 0.0;  
 totalLabel.setText("Total: $0.0");  
 }  
  
class OrderDetail {  
 private String orderId; // For full order  
 private ArrayList<OrderDetail> items; // List of items in the order  
 private double total; // Total price of the order  
  
 // Fields for individual product details  
 private String productName;  
 private String productCategory;  
 private double productPrice;  
 private int productQuantity;  
  
 // Constructor for full order  
 public OrderDetail(String orderId, ArrayList<OrderDetail> items, double total) {  
 this.orderId = orderId;  
 this.items = items;  
 this.total = total;  
 }  
  
 // Constructor for individual product details  
 public OrderDetail(String productName, String productCategory, double productPrice, int productQuantity) {  
 this.productName = productName;  
 this.productCategory = productCategory;  
 this.productPrice = productPrice;  
 this.productQuantity = productQuantity;  
 }  
  
 public String getOrderId() {  
 return orderId;  
 }  
  
 public ArrayList<OrderDetail> getItems() {  
 return items;  
 }  
  
 public double getTotal() {  
 return total;  
 }  
  
 public String getProductName() {  
 return productName;  
 }  
  
 public String getProductCategory() {  
 return productCategory;  
 }  
  
 public double getProductPrice() {  
 return productPrice;  
 }  
  
 public int getProductQuantity() {  
 return productQuantity;  
 }  
  
 @Override  
 public String toString() {  
 if (orderId != null) {  
 return "Order ID: " + orderId + ", Total: $" + total;  
 } else {  
 return "Product: " + productName + ", Category: " + productCategory +  
 ", Price: $" + productPrice + ", Quantity: " + productQuantity;  
 }  
 }  
}  
 public void displayOrderDetails() {  
 // Create a new frame to display orders  
 JFrame orderFrame = new JFrame("Order Details");  
 orderFrame.setSize(800, 600);  
 orderFrame.setDefaultCloseOperation(JFrame.*DISPOSE\_ON\_CLOSE*);  
 orderFrame.setLocationRelativeTo(null);  
  
  
 String[] columnNames = {"Order ID", "Product Name", "Category", "Price", "Quantity", "Total"};  
  
  
 DefaultTableModel tableModel = new DefaultTableModel(columnNames, 0);  
  
 // Loop through the orders list and add rows to the table model  
 for (OrderDetail order : orders) {  
 if (order.getItems() != null) {  
 for (OrderDetail item : order.getItems()) {  
  
 tableModel.addRow(new Object[]{  
 order.getOrderId(), item.getProductName(), item.getProductCategory(),  
 item.getProductPrice(), item.getProductQuantity(),  
 item.getProductPrice() \* item.getProductQuantity()  
 });  
 }  
 }  
 }  
 JTable orderTable = new JTable(tableModel);  
 JScrollPane scrollPane = new JScrollPane(orderTable);  
 orderFrame.add(scrollPane);  
 orderFrame.setVisible(true);  
 }  
}