```
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.HashMap;
import java.util.Map;
// Product Class
class Product {
 private String name;
 private double price;
 public Product(String name, double price) {
 this.name = name;
 this.price = price;
 }
 public String getName() {
 return name;
 }
 public double getPrice() {
 return price;
 }
 @Override
 public String toString() {
 return name + " - $" + price;
 }
@Override
 public boolean equals(Object obj) {
 if (this == obj) return true;
 if (obj == null || getClass() != obj.getClass()) return false;
 Product product = (Product) obj;
 return name.equals(product.name);
 }
 @Override
 public int hashCode() {
 return name.hashCode();
 }
}
// ShoppingCart Class
class ShoppingCart {
 private Map<Product, Integer> cartItems = new HashMap<>();
 public void addProduct(Product product, int quantity) {
 cartItems.put(product, cartItems.getOrDefault(product, 0) + quantity);
 }
 public void removeProduct(Product product) {
```

```
 cartItems.remove(product);
 }
 public Map<Product, Integer> getCartItems() {
 return cartItems;
 }
 public double calculateTotal() {
 double total = 0;
 for (Map.Entry<Product, Integer> entry : cartItems.entrySet()) {
 total += entry.getKey().getPrice() * entry.getValue();
 }
return total;
 }
}
public class SupermarketBillingSwing {
 private JFrame frame;
 private JTable cartTable;
 private DefaultTableModel tableModel;
 private JLabel totalLabel;
 private ShoppingCart cart;
 private Product[] products;
 public SupermarketBillingSwing() {
 cart = new ShoppingCart();
 products = new Product[] {
 new Product("Apple", 0.5),
 new Product("Bread", 1.2),
 new Product("Milk", 1.5)
 };
 frame = new JFrame("Supermarket Billing System");
 frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
 frame.setSize(600, 400);
 frame.setLayout(new BorderLayout());
 JPanel productPanel = new JPanel();
 productPanel.setLayout(new GridLayout(products.length + 1, 3, 10, 10));
 productPanel.setBorder(BorderFactory.createTitledBorder("Products"));
 JLabel[] productLabels = new JLabel[products.length];
 JButton[] addButtons = new JButton[products.length];
 JTextField[] quantityFields = new JTextField[products.length];
 for (int i = 0; i < products.length; i++) {
 productLabels[i] = new JLabel(products[i].toString());
 quantityFields[i] = new JTextField("1", 5);
 addButtons[i] = new JButton("Add");
 int productIndex = i;
 addButtons[i].addActionListener(e -> {
 try {
```

```
 int quantity = Integer.parseInt(quantityFields[productIndex].getText());
if (quantity > 0) {
 cart.addProduct(products[productIndex], quantity);
 updateCartTable();
 } else {
 JOptionPane.showMessageDialog(frame, "Enter a valid quantity.", "Error",
JOptionPane.ERROR_MESSAGE);
 }
 } catch (NumberFormatException ex) {
 JOptionPane.showMessageDialog(frame, "Invalid quantity.", "Error",
JOptionPane.ERROR_MESSAGE);
 }
 });
 productPanel.add(productLabels[i]);
 productPanel.add(quantityFields[i]);
 productPanel.add(addButtons[i]);
 }
 JPanel cartPanel = new JPanel(new BorderLayout());
 cartPanel.setBorder(BorderFactory.createTitledBorder("Shopping Cart"));
 tableModel = new DefaultTableModel(new String[]{"Product", "Quantity", "Price"}, 0);
 cartTable = new JTable(tableModel);
 JScrollPane scrollPane = new JScrollPane(cartTable);
 JPanel cartButtonPanel = new JPanel();
 JButton removeButton = new JButton("Remove Selected");
 removeButton.addActionListener(e -> {
 int selectedRow = cartTable.getSelectedRow();
 if (selectedRow >= 0) {
 String productName = (String) tableModel.getValueAt(selectedRow, 0);
 Product productToRemove = null;
 for (Product product : products) {
 if (product.getName().equals(productName)) {
 productToRemove = product;
 break;
 }
 }
 if (productToRemove != null) {
cart.removeProduct(productToRemove);
updateCartTable();
 }
 } else {
 JOptionPane.showMessageDialog(frame, "Select a product to remove.", "Error",
JOptionPane.ERROR_MESSAGE)
}
 });
```

```
 JButton generateBillButton = new JButton("Generate Bill");
 generateBillButton.addActionListener(e -> generateBill());
 cartButtonPanel.add(removeButton);
 cartButtonPanel.add(generateBillButton);
 cartPanel.add(scrollPane, BorderLayout.CENTER);
 cartPanel.add(cartButtonPanel, BorderLayout.SOUTH);
 JPanel totalPanel = new JPanel(new FlowLayout(FlowLayout.RIGHT));
 totalLabel = new JLabel("Total: $0.00");
 totalPanel.add(totalLabel);
 frame.add(productPanel, BorderLayout.NORTH);
 frame.add(cartPanel, BorderLayout.CENTER);
 frame.add(totalPanel, BorderLayout.SOUTH);
 frame.setVisible(true);
 }
 private void updateCartTable() {
 tableModel.setRowCount(0); // Clear the table
 for (Map.Entry<Product, Integer> entry : cart.getCartItems().entrySet()) {
 Product product = entry.getKey();
 int quantity = entry.getValue();
 tableModel.addRow(new Object[]{product.getName(), quantity, product.getPrice() *
quantity});
 }
 totalLabel.setText(String.format("Total: $%.2f", cart.calculateTotal()));
 }
 private void generateBill() {
double total = cart.calculateTotal();
 if (total == 0) {
 JOptionPane.showMessageDialog(frame, "Your cart is empty.", "Bill",
JOptionPane.INFORMATION_MESSAGE);
 } else {
 double tax = total * 0.10; // 10% tax
 double finalTotal = total + tax;
 JOptionPane.showMessageDialog(frame,
 String.format("Subtotal: $%.2f\nTax (10%%): $%.2f\nTotal: $%.2f", total, tax,
finalTotal),
 "Bill", JOptionPane.INFORMATION_MESSAGE);
 }
 }
 public static void main(String[] args) {
 new SupermarketBillingSwing();
 }
}
```