**SIR SYED C@SE INSTITUTE OF TECHNOLOGY ISLAMABAD**



**PROJECT NAME**

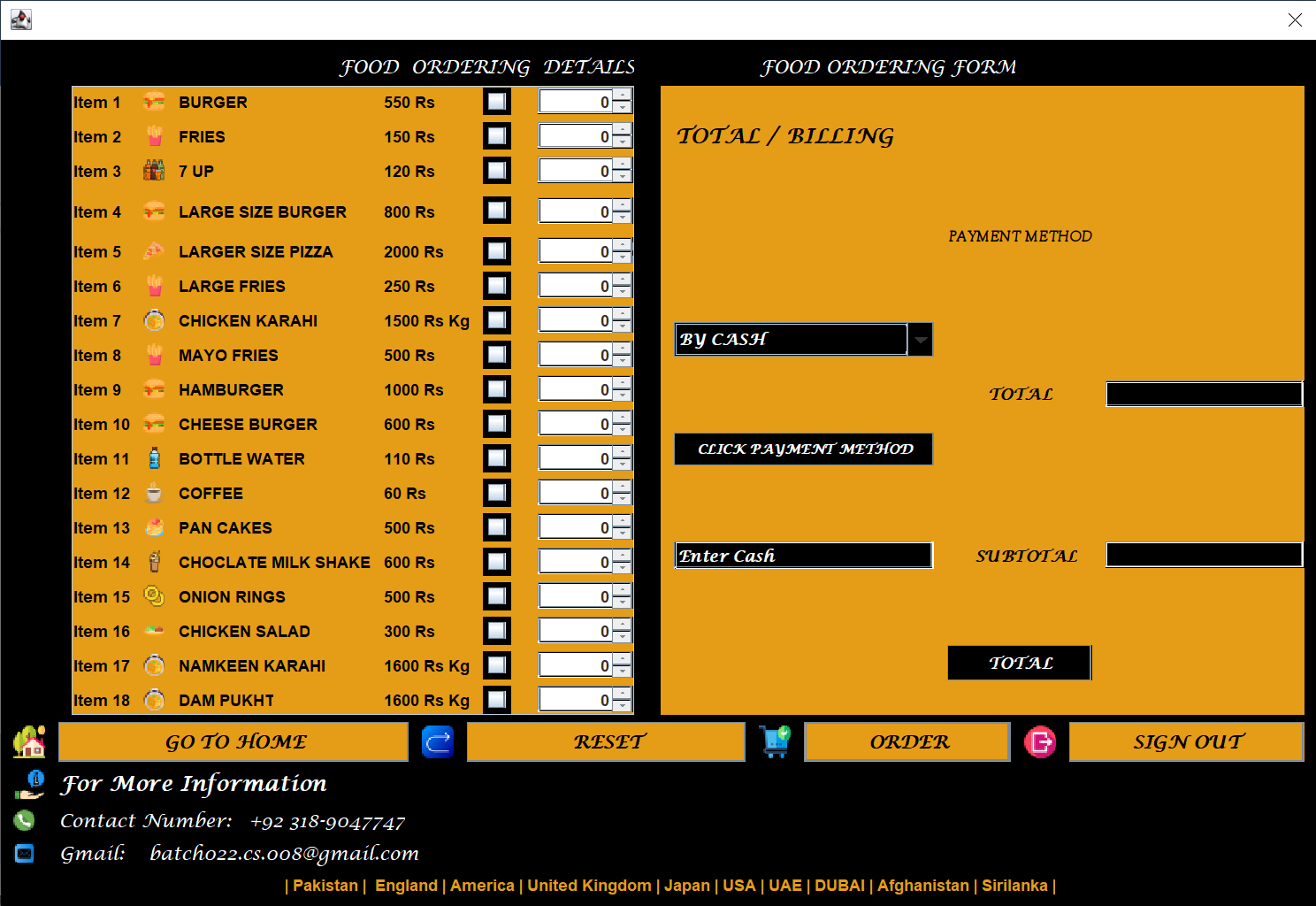
**YUM YUM CAMPUS BITES FOOD APPLICATION**

**(FOOD DELIVERY FORM)**

**CODE:**

package FOOD\_PANDA\_APP;  
import javax.swing.\*;  
import javax.swing.table.DefaultTableModel;  
import java.awt.\*;  
import java.awt.event.FocusAdapter;  
import java.awt.event.FocusEvent;  
import java.io.BufferedWriter;  
import java.io.FileWriter;  
import java.io.IOException;  
import java.net.URI;  
import java.text.SimpleDateFormat;  
import java.util.Date;  
  
public class order extends JDialog  
{  
 private JPanel contentPane;  
 private JButton buttonOK;  
 private JButton buttonCancel;  
 private JCheckBox checkBox2;  
 private JCheckBox checkBox3;  
 private JCheckBox checkBox4;  
 private JCheckBox checkBox5;  
 private JCheckBox checkBox6;  
 private JCheckBox checkBox7;  
 private JSpinner spinner1;  
 private JSpinner spinner2;  
 private JSpinner spinner3;  
 private JSpinner spinner4;  
 private JSpinner spinner5;  
 private JSpinner spinner6;  
 private JSpinner spinner7;  
 private JSpinner spinner8;  
 private JSpinner spinner9;  
 private JSpinner spinner11;  
 private JSpinner spinner12;  
 private JSpinner spinner13;  
 private JSpinner spinner14;  
 private JSpinner spinner15;  
 private JSpinner spinner16;  
 private JSpinner spinner17;  
 private JSpinner spinner18;  
 private JSpinner spinner19;  
 private JSpinner spinner20;  
 private JCheckBox checkBox8;  
 private JCheckBox checkBox9;  
 private JCheckBox checkBox10;  
 private JCheckBox checkBox11;  
 private JCheckBox checkBox12;  
 private JCheckBox checkBox13;  
 private JCheckBox checkBox14;  
 private JCheckBox checkBox15;  
 private JCheckBox checkBox16;  
 private JCheckBox checkBox17;  
 private JCheckBox checkBox18;  
 private JCheckBox checkBox19;  
 private JButton CANCELORDERButton;  
 private JComboBox comboBox1;  
 private JTextField enterCashTextField;  
 private JTextField textField1;  
 private JTextField textField2;  
 private JButton RESETButton;  
 private JButton clickPAYMENTMETHODButton;  
 private JButton EXITButton;  
  
 public order()  
 {  
 setContentPane(contentPane);  
 setModal(true);  
 getRootPane().setDefaultButton(buttonOK);  
 CANCELORDERButton.addActionListener(e -> {  
 Home home=new Home();  
 dispose();  
 home.Home1();  
 });  
  
 buttonCancel.addActionListener(e -> calculateTotal());  
  
 enterCashTextField.addFocusListener(new FocusAdapter() {  
 @Override  
 public void focusGained(FocusEvent e) {  
 super.focusGained(e);  
 enterCashTextField.setText(""); // Clear the text when gaining focus  
 }  
 });  
  
  
 RESETButton.addActionListener(e -> resetFields());  
  
 buttonOK.addActionListener(e -> showOrderDetails());  
  
 clickPAYMENTMETHODButton.addActionListener(e -> handlePaymentMethod());  
 EXITButton.addActionListener(e -> {  
 SignIn2 s2=new SignIn2();  
 dispose();  
 s2.SignIn2();  
 });  
 }  
  
 // New method to handle the payment method button click  
 private void handlePaymentMethod() {  
 // Check the selected payment method (Assuming you have a JComboBox named comboBox1)  
 String selectedPaymentMethod = (String) comboBox1.getSelectedItem();  
  
 if ("BY EASYPAISA".equals(selectedPaymentMethod)) {  
 // Open the EasyPaisa website in the default web  
 openEasyPaisaWebsite();  
 }  
 else if("BY PTM".equals(selectedPaymentMethod))  
 {  
 openPTMWebsite();  
 }  
 else if("BY JAZCASH".equals(selectedPaymentMethod))  
 {  
 openJAZCASHWebsite();  
 }  
 else if("BY ZINDAGI".equals(selectedPaymentMethod))  
 {  
 openZindagiWebsite();  
 }  
 else if("BY NAYAPAY".equals(selectedPaymentMethod))  
 {  
 openNayaWebsite();  
 }  
 else if("BY PAYPAL".equals(selectedPaymentMethod))  
 {  
 openPayPalWebsite();  
 }  
 else if ("BY SADA PAY".equals(selectedPaymentMethod))  
 {  
 openSadaPayWebsite();  
  
 }  
 }  
  
 // Method to open the EasyPaisa website  
 private void openEasyPaisaWebsite() {  
 try {  
 // Replace "https://easypaisa.com.pk/" with the actual URL of the EasyPaisa website  
 URI easypaisaUri = new URI("https://easypaisa.com.pk/");  
  
 // Open the default web browser with the EasyPaisa website  
 Desktop.*getDesktop*().browse(easypaisaUri);  
 } catch (Exception ex) {  
 // Handle any exceptions that may occur while opening the website  
 ex.printStackTrace();  
 }  
 }  
  
 private void openSadaPayWebsite() {  
 try {  
 URI easypaisaUri = new URI("https://sadapay.pk/");  
  
 Desktop.*getDesktop*().browse(easypaisaUri);  
 } catch (Exception ex) {  
 // Handle any exceptions that may occur while opening the website  
 ex.printStackTrace();  
 }  
 }  
 private void openJAZCASHWebsite() {  
 try {  
 URI easypaisaUri = new URI("https://www.jazzcash.com.pk/");  
  
 Desktop.*getDesktop*().browse(easypaisaUri);  
 } catch (Exception ex) {  
 // Handle any exceptions that may occur while opening the website  
 ex.printStackTrace();  
 }  
 }  
  
 private void openZindagiWebsite() {  
 try {  
  
 URI easypaisaUri = new URI("https://zindigi.pk/");  
  
  
 Desktop.*getDesktop*().browse(easypaisaUri);  
 } catch (Exception ex) {  
 // Handle any exceptions that may occur while opening the website  
 ex.printStackTrace();  
 }  
 }  
  
 private void openPTMWebsite() {  
 try {  
  
 URI easypaisaUri = new URI("https://www.bankalfalah.com/");  
  
  
 Desktop.*getDesktop*().browse(easypaisaUri);  
 } catch (Exception ex) {  
 // Handle any exceptions that may occur while opening the website  
 ex.printStackTrace();  
 }  
 }  
  
 private void openNayaWebsite() {  
 try {  
  
 URI easypaisaUri = new URI("https://www.nayapay.com/");  
  
 Desktop.*getDesktop*().browse(easypaisaUri);  
 } catch (Exception ex) {  
 // Handle any exceptions that may occur while opening the website  
 ex.printStackTrace();  
 }  
 }  
  
 private void openPayPalWebsite() {  
 try {  
 URI easypaisaUri = new URI("https://www.paypal.com/pk/home/");  
  
 Desktop.*getDesktop*().browse(easypaisaUri);  
 } catch (Exception ex) {  
 // Handle any exceptions that may occur while opening the website  
 ex.printStackTrace();  
 }  
 }  
  
 private void showOrderDetails() {  
 // Create a table model  
 DefaultTableModel tableModel = new DefaultTableModel();  
 tableModel.addColumn("Item");  
 tableModel.addColumn("Quantity");  
 // Add selected items to the table model  
 /\*addSelectedItemsToTableModel(tableModel, checkBox1, "Item 1");\*/  
 addSelectedItemsToTableModel(tableModel, checkBox2, "Item 1");  
 addSelectedItemsToTableModel(tableModel, checkBox3, "Item 2");  
 addSelectedItemsToTableModel(tableModel, checkBox4, "Item 3");  
 addSelectedItemsToTableModel(tableModel, checkBox5, "Item 4");  
 addSelectedItemsToTableModel(tableModel, checkBox6, "Item 5");  
 addSelectedItemsToTableModel(tableModel, checkBox7, "Item 6");  
 addSelectedItemsToTableModel(tableModel, checkBox8, "Item 7");  
 addSelectedItemsToTableModel(tableModel, checkBox9, "Item 8");  
 addSelectedItemsToTableModel(tableModel, checkBox10, "Item 9");  
 addSelectedItemsToTableModel(tableModel, checkBox11, "Item 10");  
 addSelectedItemsToTableModel(tableModel, checkBox12, "Item 11");  
 addSelectedItemsToTableModel(tableModel, checkBox13, "Item 12");  
 addSelectedItemsToTableModel(tableModel, checkBox14, "Item 13");  
 addSelectedItemsToTableModel(tableModel, checkBox15, "Item 14");  
 addSelectedItemsToTableModel(tableModel, checkBox16, "Item 15");  
 addSelectedItemsToTableModel(tableModel, checkBox17, "Item 16");  
 addSelectedItemsToTableModel(tableModel, checkBox18, "Item 17");  
 addSelectedItemsToTableModel(tableModel, checkBox19, "Item 18");  
  
 // Generate and display the bill  
 generateBill(tableModel);  
 }  
  
 private void generateBill(DefaultTableModel tableModel) {  
 StringBuilder bill = new StringBuilder();  
 bill.append("\*\*\*\*\*\*\*\*\*\*\*\* YumYum Campus Bites Food Application \*\*\*\*\*\*\*\*\*\*\*\*\n");  
  
 // Add selected items and quantities to the bill  
 for (int row = 0; row < tableModel.getRowCount(); row++) {  
 String item = (String) tableModel.getValueAt(row, 0);  
 int quantity = (int) tableModel.getValueAt(row, 1);  
 bill.append(String.*format*("%-20s %d\n", item, quantity));  
 }  
  
 // Add total to the bill  
 double total = Double.*parseDouble*(textField2.getText());  
 bill.append(String.*format*("\nTotal: %.2f PKR\n", total));  
  
 // Display the bill  
 JOptionPane.*showMessageDialog*(null, bill.toString(), "Bill", JOptionPane.*INFORMATION\_MESSAGE*);  
  
 // Store the bill in a file with date and time  
 saveBillToFile(bill.toString());  
 }  
  
 private void saveBillToFile(String billContent) {  
 try {  
 SimpleDateFormat dateFormat = new SimpleDateFormat("yyyyMMdd\_HHmmss");  
 String fileName = "bill\_" + dateFormat.format(new Date()) + ".txt";  
  
 try (BufferedWriter writer = new BufferedWriter(new FileWriter(fileName))) {  
 writer.write(billContent);  
 }  
  
 JOptionPane.*showMessageDialog*(null, "Bill saved to file: " + fileName, "File Saved", JOptionPane.*INFORMATION\_MESSAGE*);  
 } catch (IOException e) {  
 e.printStackTrace();  
 JOptionPane.*showMessageDialog*(null, "Error saving bill to file", "Error", JOptionPane.*ERROR\_MESSAGE*);  
 }  
 }  
 private void writeOrderDetailsToFile(DefaultTableModel tableModel) {  
 try (BufferedWriter writer = new BufferedWriter(new FileWriter("order\_details.txt"))) {  
 // Write header  
 writer.write("Item\tQuantity");  
 writer.newLine();  
  
 // Write each row  
 for (int row = 0; row < tableModel.getRowCount(); row++) {  
 String item = (String) tableModel.getValueAt(row, 0);  
 int quantity = (int) tableModel.getValueAt(row, 1);  
  
 // Write item and quantity to the file  
 writer.write(item + "\t" + quantity);  
 writer.newLine();  
 }  
  
 // Display a message indicating that the order details are saved  
 JOptionPane.*showMessageDialog*(null, "Order details saved to file: order\_details.txt");  
 } catch (IOException e) {  
 e.printStackTrace();  
 JOptionPane.*showMessageDialog*(null, "Error saving order details to file");  
 }  
 }  
 private void addSelectedItemsToTableModel(DefaultTableModel tableModel, JCheckBox checkBox, String itemName) {  
 if (checkBox.isSelected()) {  
 int quantity = getQuantityForCheckBox(checkBox);  
 tableModel.addRow(new Object[]{itemName, quantity});  
 }  
 }  
 private int getQuantityForCheckBox(JCheckBox checkBox) {  
 int quantity = 0;  
  
 // Determine the corresponding spinner based on the checkbox  
 JSpinner spinner = null;  
 switch (checkBox.getText()) {  
 case "Item 1":  
 spinner = spinner2;  
 break;  
 case "Item 2":  
 spinner = spinner3;  
 break;  
 case "Item 3":  
 spinner = spinner4;  
 break;  
 case "Item 4":  
 spinner = spinner5;  
 break;  
 case "Item 5":  
 spinner = spinner6;  
 break;  
 case "Item 6":  
 spinner = spinner7;  
 break;  
 case "Item 7":  
 spinner = spinner8;  
 break;  
 case "Item 8":  
 spinner = spinner9;  
 break;  
 case "Item 9":  
 spinner = spinner11;  
 break;  
 case "Item 10":  
 spinner = spinner12;  
 break;  
 case "Item 11":  
 spinner = spinner13;  
 break;  
 case "Item 12":  
 spinner = spinner14;  
 break;  
 case "Item 13":  
 spinner = spinner15;  
 break;  
 case "Item 14":  
 spinner = spinner16;  
 break;  
 case "Item 15":  
 spinner = spinner17;  
 break;  
 case "Item 16":  
 spinner = spinner18;  
 break;  
 case "Item 17":  
 spinner = spinner19;  
 break;  
 case "Item 18":  
 spinner = spinner20;  
 break;  
 // ... repeat for other items  
 }  
  
 if (spinner != null) {  
 quantity = (int) spinner.getValue();  
 }  
  
 return quantity;  
 }  
  
 private void resetFields()  
 {  
 // Reset checkboxes  
 /\*checkBox1.setSelected(false);\*/  
 checkBox2.setSelected(false);  
 checkBox3.setSelected(false);  
 checkBox4.setSelected(false);  
 checkBox5.setSelected(false);  
 checkBox6.setSelected(false);  
 checkBox7.setSelected(false);  
 checkBox8.setSelected(false);  
 checkBox9.setSelected(false);  
 checkBox10.setSelected(false);  
 checkBox11.setSelected(false);  
 checkBox12.setSelected(false);  
 checkBox13.setSelected(false);  
 checkBox14.setSelected(false);  
 checkBox15.setSelected(false);  
 checkBox16.setSelected(false);  
 checkBox17.setSelected(false);  
 checkBox18.setSelected(false);  
 checkBox19.setSelected(false);  
 // ... repeat for other checkboxes  
  
  
 // Reset spinners  
 /\*spinner1.setValue(0);\*/  
 spinner2.setValue(0);  
 spinner3.setValue(0);  
 spinner4.setValue(0);  
 spinner5.setValue(0);  
 spinner6.setValue(0);  
 spinner7.setValue(0);  
 spinner8.setValue(0);  
 spinner9.setValue(0);  
 spinner11.setValue(0);  
 spinner12.setValue(0);  
 spinner13.setValue(0);  
 spinner14.setValue(0);  
 spinner15.setValue(0);  
 spinner16.setValue(0);  
 spinner17.setValue(0);  
 spinner18.setValue(0);  
 spinner19.setValue(0);  
 spinner20.setValue(0);  
 // ... repeat for other spinners  
  
 // Reset text fields  
 enterCashTextField.setText("");  
 textField1.setText("");  
 textField2.setText("");  
 // ... repeat for other text fields  
  
 // Reset other components as needed  
 }  
  
 private void calculateTotal() {  
 // Define prices for each food item  
 /\*double price1 = 1000; // Adjust the prices accordingly\*/  
 double price2 = 550;  
 double price3 = 150;  
 double price4 = 120; // Adjust the prices accordingly  
 double price5 = 800;  
 double price6 = 2000;  
 double price7 = 250; // Adjust the prices accordingly  
 double price8 = 1500;  
 double price9 = 500;  
 double price10 = 1000; // Adjust the prices accordingly  
 double price11 = 600;  
 double price12 = 110;  
 double price13= 60; // Adjust the prices accordingly  
 double price14= 500;  
 double price15= 600;  
 double price16= 500; // Adjust the prices accordingly  
 double price17= 300;  
 double price18= 1600;  
 double price19= 1600; // Adjust the prices accordingly  
  
 // ... add more prices if needed  
  
 // Get the quantities from spinners  
 /\*int quantity1 = (int) spinner1.getValue();\*/  
 int quantity2 = (int) spinner2.getValue();  
 int quantity3 = (int) spinner3.getValue();  
 int quantity4 = (int) spinner4.getValue();  
 int quantity5 = (int) spinner5.getValue();  
 int quantity6 = (int) spinner6.getValue();  
 int quantity7 = (int) spinner7.getValue();  
 int quantity8 = (int) spinner8.getValue();  
 int quantity9 = (int) spinner9.getValue();  
 int quantity10 = (int) spinner11.getValue();  
 int quantity11 = (int) spinner12.getValue();  
 int quantity12 = (int) spinner13.getValue();  
 int quantity13 = (int) spinner14.getValue();  
 int quantity14 = (int) spinner15.getValue();  
 int quantity15 = (int) spinner16.getValue();  
 int quantity16 = (int) spinner17.getValue();  
 int quantity17 = (int) spinner18.getValue();  
 int quantity18 = (int) spinner19.getValue();  
 int quantity19 = (int) spinner20.getValue();  
  
  
 // Calculate the total  
 double total = (price2 \* quantity2) + (price3 \* quantity3)+  
 (price4 \* quantity4) + (price5 \* quantity5) + (price6 \* quantity6)+  
 (price7 \* quantity7) + (price8 \* quantity8) + (price9 \* quantity9)+  
 (price10 \* quantity10) + (price11 \* quantity11) + (price12 \* quantity12)+  
 (price13 \* quantity13) + (price14 \* quantity14) + (price16 \* quantity16)+  
 (price15 \* quantity15) + (price17 \* quantity17) + (price18 \* quantity18)+  
 (price19 \* quantity19);  
  
 // ... add more items to the total if needed  
  
 // Display the total in the totalTextField  
 textField1.setText(String.*valueOf*(total));  
 textField2.setText(String.*valueOf*(total));  
 }  
 public static void main(String[] args) {  
 order dialog = new order();  
 dialog.pack();  
 dialog.setResizable(false);  
 dialog.setVisible(true);  
 System.*exit*(0);  
 }  
}

**OUTPUT:**



**In this it is showing us following buttons**

1. **Go To Home:**

It will open Home Page.

1. **Reset:**

It will Reset all the Page.

1. **Order:**

It will order the items which were added by the user to order and it will be save with date and time and also be stored in file.

1. **Sign Out:**

It will exit the program and will shift to sign in page.

Other are the thing which can be showing in pictures there are spinners, and etc.