**** **OOPJ (CSL304) PROJECT REPORT**

Rajiv Gandhi Institute of Technology

Department of

Computer Engineering

Academic Year: 2020-2021

**OOPJ (CSL304) PROJECT REPORT**

**ON**

**“RESTAURANT ORDER MANAGEMENT SYSTEM”**

**Submitted by:**

**(B-470) Aayush Shah**

**(B-472) Sanit Patil**

**(B-475) Mohammed Anas Shaikh**

**Submitted to:**

**Mrs.KaajalSharma**



Juhu-Versova Link Road Versova, Andheri(W), Mumbai-53

# Department of Computer Engineering

**C E R T I F I C A T E**

This is to certify that

1. Aayush Shah (SE-B 470)
2. Sanit Patil (SE-B 472)
3. Mohammed Anas Shaikh (SE-B 475 )

Have satisfactorily completed this project entitled

## **“Restaurant Management System**”

Towards the partial fulfilment of the

**SECOND YEAR BACHELOR OF ENGINEERING IN**

**(COMPUTER ENGINEERING)**

**as laid by University of Mumbai**.

**Guide H.O.D.**

**Prof. Kaajal Sharma Prof. Sunil Khachane**

**Principal**

**Dr Sanjay Bokade**

**Contents**

|  |  |
| --- | --- |
| **Figure No.** | **Name** |
| 1 | Abstract |
| 2 | Source Code |
| 3 | Output Screen |
| 4 | Conclusion |
| 5 | Reference |

**Abstract**

The project titled “Restaurant Management Syatem” is a software for monitoring and controlling transactions in a Restaurant system. A “Restaurant Management System” is developed in JAVA.

This project is a Windows application designed to help users maintain and organize library. A system where user can access different functions for fulfilling various tasks related Restaurant. This software is easy to use for both beginners and advanced users. A virtual application for cost effective and user-friendly system.

As mentioned above, we made use of JAVA as our core programming language. Though we can use any IDE to develop the project like Eclipse/NetBeans, here we have used NetBeans. We have MySQL at the back end.

**Source code**

**AdminPassword.java**

public class AdminPassword extends javax.swing.JFrame {

/\*\*

\* Creates new form AdminPassword

\*/

public AdminPassword() {

initComponents();

lblPassword.setVisible(false);

}

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

btnLogin = new javax.swing.JButton();

lblPassword = new javax.swing.JLabel();

txtPassword = new javax.swing.JPasswordField();

btnBack = new javax.swing.JButton();

jLabel1 = new javax.swing.JLabel();

jLabel3 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

getContentPane().setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

btnLogin.setBackground(new java.awt.Color(255, 255, 255));

btnLogin.setText("Login");

btnLogin.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnLoginActionPerformed(evt);

}

});

getContentPane().add(btnLogin, new org.netbeans.lib.awtextra.AbsoluteConstraints(240, 160, 100, 25));

lblPassword.setBackground(new java.awt.Color(255, 255, 255));

lblPassword.setFont(new java.awt.Font("Tunga", 0, 18)); // NOI18N

lblPassword.setForeground(new java.awt.Color(255, 171, 31));

lblPassword.setText("Incorrect Password Please Try Again");

getContentPane().add(lblPassword, new org.netbeans.lib.awtextra.AbsoluteConstraints(120, 130, 250, 20));

getContentPane().add(txtPassword, new org.netbeans.lib.awtextra.AbsoluteConstraints(120, 100, 179, -1));

btnBack.setBackground(new java.awt.Color(255, 255, 255));

btnBack.setText("Back");

btnBack.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnBackActionPerformed(evt);

}

});

getContentPane().add(btnBack, new org.netbeans.lib.awtextra.AbsoluteConstraints(30, 160, -1, 25));

jLabel1.setFont(new java.awt.Font("Microsoft Sans Serif", 1, 18)); // NOI18N

jLabel1.setForeground(new java.awt.Color(255, 255, 255));

jLabel1.setText("Admin Login");

getContentPane().add(jLabel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(120, 50, 120, 20));

jLabel3.setFont(new java.awt.Font("Microsoft Sans Serif", 1, 14)); // NOI18N

jLabel3.setForeground(new java.awt.Color(255, 255, 255));

jLabel3.setText("Password");

getContentPane().add(jLabel3, new org.netbeans.lib.awtextra.AbsoluteConstraints(30, 110, 70, 10));

jLabel2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Grey-Background-GREY.jpg"))); // NOI18N

getContentPane().add(jLabel2, new org.netbeans.lib.awtextra.AbsoluteConstraints(0, 0, 380, 210));

pack();

}// </editor-fold>

private void btnLoginActionPerformed(java.awt.event.ActionEvent evt) {

String password = txtPassword.getText();

//if statement to validate user input for admin password

if(password.equals("password")){

//if entered string mathces the password, it will display the admin home screen, and make it visible

Frame f = new Frame();

this.setVisible(false);

f.setVisible(true);

}

else{

//empties the password field and informs user they are wrong

txtPassword.setText(null);

lblPassword.setVisible(true);

}

}

private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {

Homepage h = new Homepage();

h.setVisible(true);

this.setVisible(false);

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(AdminPassword.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(AdminPassword.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(AdminPassword.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(AdminPassword.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new AdminPassword().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton btnBack;

private javax.swing.JButton btnLogin;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel lblPassword;

private javax.swing.JPasswordField txtPassword;

// End of variables declaration

}

**Dessert.java**

class Dessert {

public Dessert()//constructor

{

}

private String Dessert, DessertPrice;//variable declaration

//getters and setters

public String getDessert()

{

return Dessert;

}

public String getDessertPrice()

{

return DessertPrice;

}

protected void setDessert(String iDessert)

{

this.Dessert = iDessert;

}

protected void setDessertPrice(String iDessertPrice)

{

this.DessertPrice = iDessertPrice;

}

}

**Drink.java**

class Drink {

public Drink()//constructor

{

}

private String Drink, DrinkPrice;//variable declaration

//getters and setters

public String getDrink()

{

return Drink;

}

public String getDrinkPrice()

{

return DrinkPrice;

}

protected void setDrink(String iDrink)

{

this.Drink = iDrink;

}

protected void setDrinkPrice(String iDrinkPrice)

{

this.DrinkPrice = iDrinkPrice;

}

}

**Employee.java**

public class Employee {

//varaible declarations

private String firstName, lastName, PPSN, position, wage;

private int employeeID;

public Employee()//constructor

{

}

//getters and setters

public int getEmployeeID()

{

return employeeID;

}

public String getFirstName()

{

return firstName;

}

public String getLastName()

{

return lastName;

}

public String getPPSN()

{

return PPSN;

}

public String getWage()

{

return wage;

}

public String getPosition()

{

return position;

}

protected void setEmployeeID(int iEmployeeID)

{

this.employeeID = iEmployeeID;

}

protected void setFirstName(String iFirstName)

{

this.firstName = iFirstName;

}

protected void setLastName(String iLastName)

{

this.lastName = iLastName;

}

protected void setPPSN(String iPPSN)

{

this.PPSN = iPPSN;

}

protected void setWage(String iWage)

{

this.wage = iWage;

}

protected void setPosition(String iPosition)

{

this.position = iPosition;

}

}

**Frame.java**

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.ResultSetMetaData;

import java.sql.SQLException;

import java.sql.Statement;

import javax.swing.JOptionPane;

import javax.swing.table.DefaultTableModel;

import java.lang.NullPointerException;

import java.util.Random;

import javax.swing.event.DocumentEvent;

import javax.swing.event.DocumentListener;

/\*\*

\*

\* @author IS2209 Group

\*/

public class Frame extends javax.swing.JFrame {

/\*\*

\* Creates new form Frame

\*/

//declaring a new instance of our database class

database a = new database();

//Declaring a new instance of classes

Employee e = new Employee();

//Creating a random number to be used as employee number

Random rand = new Random();

int x = rand.nextInt(10000);

DefaultTableModel model;

public Frame() {

initComponents();

//a.dropTable();

//a.createTable();

//assigning the default table model to our table

model = (DefaultTableModel) tblEmployees.getModel();

//calling the method which will populate tables upon initialization

populateTable();

btnAddR.setEnabled(false);

btnDelete.setEnabled(false);

//Document Listeners for validation, buttons become enabled when certain conditions are met

//Code manipulated from http://stackoverflow.com/questions/10848335/how-to-implement-documentlistener

txtPPSN.getDocument().addDocumentListener(new DocumentListener() {

@Override

public void changedUpdate(DocumentEvent e){

lengthCheck();

}

@Override

public void insertUpdate(DocumentEvent e) {

lengthCheck();

}

@Override

public void removeUpdate(DocumentEvent e) {

lengthCheck();

}

});

txtFirstName.getDocument().addDocumentListener(new DocumentListener() {

@Override

public void changedUpdate(DocumentEvent e){

lengthCheck();

}

@Override

public void insertUpdate(DocumentEvent e) {

lengthCheck();

}

@Override

public void removeUpdate(DocumentEvent e) {

lengthCheck();

}

});

txtLastName.getDocument().addDocumentListener(new DocumentListener() {

@Override

public void changedUpdate(DocumentEvent e){

lengthCheck();

}

@Override

public void insertUpdate(DocumentEvent e) {

lengthCheck();

}

@Override

public void removeUpdate(DocumentEvent e) {

lengthCheck();

}

});

txtWage.getDocument().addDocumentListener(new DocumentListener() {

@Override

public void changedUpdate(DocumentEvent e){

lengthCheck();

}

@Override

public void insertUpdate(DocumentEvent e) {

lengthCheck();

}

@Override

public void removeUpdate(DocumentEvent e) {

lengthCheck();

}

});

txtPosition.getDocument().addDocumentListener(new DocumentListener() {

@Override

public void changedUpdate(DocumentEvent e){

lengthCheck();

}

@Override

public void insertUpdate(DocumentEvent e) {

lengthCheck();

}

@Override

public void removeUpdate(DocumentEvent e) {

lengthCheck();

}

});

}

//populate the table

public void populateTable()

{

//decalring variables

int EmployeeID;

String FirstName;

String LastName;

String PPSN;

String Wages;

String Position;

int r =0;

try

{

a.createConnection();//creates connection

a.stmt = a.conn.createStatement();//creates and executes sql statement

ResultSet results = a.stmt.executeQuery("select \* from " + a.tableName);

ResultSetMetaData rsmd = results.getMetaData();

while(results.next())

{

//Fills table until it reaches the end of the database

EmployeeID = results.getInt(1);

FirstName = results.getString(2);

LastName = results.getString(3);

PPSN = results.getString(4);

Wages = results.getString(5);

Position = results.getString(6);

model.insertRow(model.getRowCount(), new Object[]{EmployeeID, FirstName, LastName, PPSN, Wages, Position});

r++;

}//end of while loop

//closes and 'shuts down any open streams and connections

results.close();

a.stmt.close();

a.shutdown();

}

catch (SQLException sqlExcept )

{

sqlExcept.printStackTrace();

}

}

//method to insert employee into the database

private void insertEmployee(Employee e)

{

a.createConnection();//creates connection

try

{

a.stmt = a.conn.createStatement();//creates and executes statement to fill database table

a.stmt.execute("insert into " + a.tableName + " values (" +

x + ",'" + e.getFirstName() + "','" + e.getLastName() + "','" + e.getPPSN() + "','" + e.getWage() + "','" + e.getPosition() + "')");

a.stmt.close();

}

catch (SQLException sqlExcept)

{

sqlExcept.printStackTrace();

}

a.shutdown();

}

//mehtod used in document listener for validation purposes

public void lengthCheck(){

if(txtFirstName.getText().equals("") || txtLastName.getText().equals("") || txtPPSN.getText().equals("")|| txtWage.getText().equals("")||txtPosition.getText().equals("")){

btnAddR.setEnabled(false);

btnDelete.setEnabled(false);

}

else if(txtFirstName.getText().length()> 50 || txtLastName.getText().length()>50 || txtPPSN.getText().length()>10|| txtWage.getText().length()>10||txtPosition.getText().length()>30){

btnAddR.setEnabled(false);

}

else{

btnAddR.setEnabled(true);

}

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

txtFirstName = new javax.swing.JTextField();

txtLastName = new javax.swing.JTextField();

txtPPSN = new javax.swing.JTextField();

txtWage = new javax.swing.JTextField();

txtPosition = new javax.swing.JTextField();

jScrollPane1 = new javax.swing.JScrollPane();

tblEmployees = new javax.swing.JTable();

btnAddR = new javax.swing.JButton();

jLabel1 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

jLabel3 = new javax.swing.JLabel();

jLabel4 = new javax.swing.JLabel();

jLabel5 = new javax.swing.JLabel();

btnDelete = new javax.swing.JButton();

btnUnSelect = new javax.swing.JButton();

btnBack = new javax.swing.JButton();

btnMenu = new javax.swing.JButton();

jLabel6 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

getContentPane().setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

txtFirstName.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

txtFirstNameActionPerformed(evt);

}

});

getContentPane().add(txtFirstName, new org.netbeans.lib.awtextra.AbsoluteConstraints(102, 22, 239, -1));

getContentPane().add(txtLastName, new org.netbeans.lib.awtextra.AbsoluteConstraints(102, 60, 239, -1));

getContentPane().add(txtPPSN, new org.netbeans.lib.awtextra.AbsoluteConstraints(411, 22, 130, -1));

getContentPane().add(txtWage, new org.netbeans.lib.awtextra.AbsoluteConstraints(411, 60, 130, -1));

getContentPane().add(txtPosition, new org.netbeans.lib.awtextra.AbsoluteConstraints(102, 98, 439, -1));

tblEmployees.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"EmployeeID", "First Name", "Last Name", "PPSN", "Weekly Wage", "Position"

}

) {

Class[] types = new Class [] {

java.lang.Integer.class, java.lang.Object.class, java.lang.Object.class, java.lang.Object.class, java.lang.Object.class, java.lang.Object.class

};

boolean[] canEdit = new boolean [] {

false, false, false, false, true, true

};

public Class getColumnClass(int columnIndex) {

return types [columnIndex];

}

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

tblEmployees.getTableHeader().setReorderingAllowed(false);

tblEmployees.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

tblEmployeesMouseClicked(evt);

}

});

jScrollPane1.setViewportView(tblEmployees);

getContentPane().add(jScrollPane1, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 136, 592, 160));

btnAddR.setBackground(new java.awt.Color(255, 255, 255));

btnAddR.setText("Add");

btnAddR.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnAddRActionPerformed(evt);

}

});

getContentPane().add(btnAddR, new org.netbeans.lib.awtextra.AbsoluteConstraints(500, 310, 100, 25));

jLabel1.setFont(new java.awt.Font("Microsoft Sans Serif", 0, 14)); // NOI18N

jLabel1.setForeground(new java.awt.Color(255, 255, 255));

jLabel1.setText("Wage");

getContentPane().add(jLabel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(359, 63, -1, -1));

jLabel2.setFont(new java.awt.Font("Microsoft Sans Serif", 0, 14)); // NOI18N

jLabel2.setForeground(new java.awt.Color(255, 255, 255));

jLabel2.setText("PPSN");

getContentPane().add(jLabel2, new org.netbeans.lib.awtextra.AbsoluteConstraints(362, 25, -1, -1));

jLabel3.setFont(new java.awt.Font("Microsoft Sans Serif", 0, 14)); // NOI18N

jLabel3.setForeground(new java.awt.Color(255, 255, 255));

jLabel3.setText("Position");

getContentPane().add(jLabel3, new org.netbeans.lib.awtextra.AbsoluteConstraints(29, 101, -1, -1));

jLabel4.setFont(new java.awt.Font("Microsoft Sans Serif", 0, 14)); // NOI18N

jLabel4.setForeground(new java.awt.Color(255, 255, 255));

jLabel4.setText("Last Name");

getContentPane().add(jLabel4, new org.netbeans.lib.awtextra.AbsoluteConstraints(29, 63, -1, -1));

jLabel5.setFont(new java.awt.Font("Microsoft Sans Serif", 0, 14)); // NOI18N

jLabel5.setForeground(new java.awt.Color(255, 255, 255));

jLabel5.setText("First Name");

getContentPane().add(jLabel5, new org.netbeans.lib.awtextra.AbsoluteConstraints(29, 25, -1, -1));

btnDelete.setBackground(new java.awt.Color(255, 255, 255));

btnDelete.setText("Delete");

btnDelete.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnDeleteActionPerformed(evt);

}

});

getContentPane().add(btnDelete, new org.netbeans.lib.awtextra.AbsoluteConstraints(390, 310, 100, 25));

btnUnSelect.setBackground(new java.awt.Color(255, 255, 255));

btnUnSelect.setText("Deselect");

btnUnSelect.setToolTipText("Use this to unselect \na row from the table");

btnUnSelect.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnUnSelectActionPerformed(evt);

}

});

getContentPane().add(btnUnSelect, new org.netbeans.lib.awtextra.AbsoluteConstraints(290, 310, 90, 25));

btnBack.setBackground(new java.awt.Color(255, 255, 255));

btnBack.setText("Back to Home");

btnBack.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnBackActionPerformed(evt);

}

});

getContentPane().add(btnBack, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 310, 110, 25));

btnMenu.setBackground(new java.awt.Color(255, 255, 255));

btnMenu.setText("Edit Menu");

btnMenu.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnMenuActionPerformed(evt);

}

});

getContentPane().add(btnMenu, new org.netbeans.lib.awtextra.AbsoluteConstraints(130, 310, 90, 25));

jLabel6.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Grey-Background-GREY.jpg"))); // NOI18N

getContentPane().add(jLabel6, new org.netbeans.lib.awtextra.AbsoluteConstraints(0, 0, 620, 350));

pack();

}// </editor-fold>

//button to add employee to table

private void btnAddRActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

//populates table with textfield values

model.insertRow(model.getRowCount(), new Object[]{x, txtFirstName.getText(), txtLastName.getText(), txtPPSN.getText(), txtWage.getText(), txtPosition.getText()});

//Gets info from textfields to be used to populate database table

e.setFirstName(txtFirstName.getText());

e.setLastName(txtLastName.getText());

e.setPPSN(txtPPSN.getText());

e.setWage(txtWage.getText());

e.setPosition(txtPosition.getText());

//clears text fields

txtFirstName.setText(null);

txtLastName.setText(null);

txtPPSN.setText(null);

txtWage.setText(null);

txtPosition.setText(null);

//calls method to insert values to database

insertEmployee(e);

x++;

}

private void txtFirstNameActionPerformed(java.awt.event.ActionEvent evt) {

}

//delete employee from database and table

private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {

//setting the selected row from the table

int row = tblEmployees.getSelectedRow();

DefaultTableModel model= (DefaultTableModel)tblEmployees.getModel();

//Sets the variable selected equal to firstvalue in selected row and sets it to a string

String selected = tblEmployees.getValueAt(row, 3).toString();

if (row >= 0) {

//remove row from table

model.removeRow(row);

try{

a.createConnection();//creates connection

//prepares statement to delete row from the database

PreparedStatement ps = a.conn.prepareStatement("delete from Employee\_Table where PPSN='"+selected+"' ");

ps.executeUpdate();

//creates text fields

txtFirstName.setText(null);

txtLastName.setText(null);

txtPPSN.setText(null);

txtWage.setText(null);

txtPosition.setText(null);

JOptionPane.showMessageDialog(null, "Deleted");

}

catch(SQLException sqlExcept){

sqlExcept.printStackTrace();

}

}

}

private void tblEmployeesMouseClicked(java.awt.event.MouseEvent evt) {

//Setting row equal to selected row

int row = tblEmployees.getSelectedRow();

//setting selected equal to the ppsn and converts to string

String selected = tblEmployees.getValueAt(row, 3).toString();

DefaultTableModel model= (DefaultTableModel)tblEmployees.getModel();

try{

if(selected !=null ){

//displays info of selected row into text fields

txtFirstName.setText(String.valueOf(model.getValueAt(tblEmployees.getSelectedRow(), 1)));

txtLastName.setText(String.valueOf(model.getValueAt(tblEmployees.getSelectedRow(), 2)));

txtPPSN.setText(String.valueOf(model.getValueAt(tblEmployees.getSelectedRow(), 3)));

txtWage.setText(String.valueOf(model.getValueAt(tblEmployees.getSelectedRow(), 4)));

txtPosition.setText(String.valueOf(model.getValueAt(tblEmployees.getSelectedRow(), 5)));

btnDelete.setEnabled(true);

btnAddR.setEnabled(false);

}

} catch(NullPointerException a){

JOptionPane.showMessageDialog(null, "Deleted");

}

}

private void btnUnSelectActionPerformed(java.awt.event.ActionEvent evt) {

//unselects row

tblEmployees.getSelectionModel().clearSelection();

txtFirstName.setText(null);

txtLastName.setText(null);

txtPPSN.setText(null);

txtWage.setText(null);

txtPosition.setText(null);

}

private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {

Homepage h = new Homepage();

h.setVisible(true);

this.setVisible(false);

}

private void btnMenuActionPerformed(java.awt.event.ActionEvent evt) {

MenuEditor me = new MenuEditor();

me.setVisible(true);

this.setVisible(false);

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Frame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Frame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Frame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Frame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new Frame().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton btnAddR;

private javax.swing.JButton btnBack;

private javax.swing.JButton btnDelete;

private javax.swing.JButton btnMenu;

private javax.swing.JButton btnUnSelect;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel jLabel4;

private javax.swing.JLabel jLabel5;

private javax.swing.JLabel jLabel6;

private javax.swing.JScrollPane jScrollPane1;

public javax.swing.JTable tblEmployees;

private javax.swing.JTextField txtFirstName;

private javax.swing.JTextField txtLastName;

private javax.swing.JTextField txtPPSN;

private javax.swing.JTextField txtPosition;

private javax.swing.JTextField txtWage;

// End of variables declaration

}

**Homepage.java**

//imports used in this frame

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.ResultSetMetaData;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import javax.swing.JOptionPane;

import javax.swing.RowSorter;

import javax.swing.SortOrder;

import javax.swing.table.DefaultTableModel;

import javax.swing.table.TableModel;

import javax.swing.table.TableRowSorter;

public class Homepage extends javax.swing.JFrame {

//declaring our table models which pertain to table structures

DefaultTableModel model;

DefaultTableModel model1;

//declaring a new instance of our database class

database a =new database();

public Homepage() { //initial contructor (will automatically call initialize components method)

initComponents();

btnRemoveOrder.setEnabled(false);

//setting our tooltips

btnRemoveOrder.setToolTipText("Select a row from the Bill table to remove, "

+ "it will also remove the corresponding orders from the orders table");

btnNewOrder.setToolTipText("This will create a new order");

btnAdmin.setToolTipText("This will take you to the admin view (where credentials are necessary");

//assigning the default table model to our two tables

model = (DefaultTableModel) tblOrders.getModel();

model1 = (DefaultTableModel) tblBill.getModel();

//calling the method which will populate tables upon initialization

populateTable();

populateTable1();

TableRowSorter<TableModel> sorter = new TableRowSorter<>(tblOrders.getModel());

tblOrders.setRowSorter(sorter);

List<RowSorter.SortKey> sortKeys = new ArrayList<>();

int columnIndexToSort = 0;

sortKeys.add(new RowSorter.SortKey(columnIndexToSort, SortOrder.ASCENDING));

sorter.setSortKeys(sortKeys);

sorter.sort();

TableRowSorter<TableModel> sorter1 = new TableRowSorter<>(tblBill.getModel());

tblBill.setRowSorter(sorter1);

List<RowSorter.SortKey> sortKeys1 = new ArrayList<>();

int columnIndexToSort1 = 0;

sortKeys1.add(new RowSorter.SortKey(columnIndexToSort1, SortOrder.ASCENDING));

sorter1.setSortKeys(sortKeys1);

sorter1.sort();

}

private void populateTable(){

//variable declarations

int OrderID;

String Starter;

String Main;

String Dessert;

String Drink;

String Price;

int r =0;

try

{

a.createConnection();//creates connection to the database

//creates and runs an sql statemtnt to display all info from the 'tablename2' variable

a.stmt = a.conn.createStatement();

ResultSet res = a.stmt.executeQuery("select \* from " + a.tableName1);

ResultSetMetaData rsmd = res.getMetaData();

int numberCols = rsmd.getColumnCount();

//loop will run as long as there is more data to read from the DB, assigning corresponding values accordingly

while(res.next())

{

OrderID = res.getInt(1);

Starter = res.getString(2);

Main = res.getString(3);

Dessert = res.getString(4);

Drink = res.getString(5);

Price = res.getString(6);

//inserts a new row into the table containing an order object with the following properties

model.insertRow(model.getRowCount(), new Object[]{OrderID, Starter, Main, Dessert, Drink, Price});

r++;

}//end while loop

//closes and 'shuts down any open streams and connections

res.close();

a.stmt.close();

a.shutdown();

}//end try

//will display apporpriate catch clause if required

catch (SQLException sqlExcept )

{

sqlExcept.printStackTrace();

}

}

private void populateTable1(){

//variable declarations

int OrderID;

String Table;

String Employee;

String TotalPrice;

int r =0;

try

{

a.createConnection();//creates connection to the database

//creates and runs an sql statemtnt to display all info from the 'tablename2' variable

a.stmt = a.conn.createStatement();

ResultSet res = a.stmt.executeQuery("select \* from " + a.tableName2);

ResultSetMetaData rsmd = res.getMetaData();

int numberCols = rsmd.getColumnCount();

//loop will run as long as there is more data to read from the DB, assigning corresponding values accordingly

while(res.next())

{

OrderID = res.getInt(1);

Table = res.getString(2);

Employee = res.getString(3);

TotalPrice = res.getString(4);

//inserts a new row into the table containing an order object with the following properties

model1.insertRow(model1.getRowCount(), new Object[]{OrderID, Table, Employee, TotalPrice});

r++;

}//end while loop

//closes and 'shuts down any open streams and connections

res.close();

a.stmt.close();

a.shutdown();

}//end try

//will display apporpriate catch clause if required

catch (SQLException sqlExcept )

{

sqlExcept.printStackTrace();

} catch(IllegalArgumentException a){

JOptionPane.showMessageDialog(this, "You have entered an illegal character", "Error", JOptionPane.ERROR\_MESSAGE);

}

}

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jScrollPane1 = new javax.swing.JScrollPane();

tblBill = new javax.swing.JTable();

btnNewOrder = new javax.swing.JButton();

jScrollPane2 = new javax.swing.JScrollPane();

tblOrders = new javax.swing.JTable();

btnRemoveOrder = new javax.swing.JButton();

btnAdmin = new javax.swing.JButton();

jLabel3 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

jLabel1 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

setTitle("Home");

getContentPane().setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

tblBill.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Order Number", "Table No.", "Employee Name", "Total Price"

}

) {

boolean[] canEdit = new boolean [] {

false, false, false, false

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

tblBill.getTableHeader().setReorderingAllowed(false);

tblBill.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

tblBillMouseClicked(evt);

}

});

jScrollPane1.setViewportView(tblBill);

if (tblBill.getColumnModel().getColumnCount() > 0) {

tblBill.getColumnModel().getColumn(0).setResizable(false);

tblBill.getColumnModel().getColumn(1).setResizable(false);

tblBill.getColumnModel().getColumn(2).setResizable(false);

tblBill.getColumnModel().getColumn(3).setResizable(false);

}

getContentPane().add(jScrollPane1, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 30, 560, 130));

btnNewOrder.setBackground(new java.awt.Color(255, 255, 255));

btnNewOrder.setText("New Order");

btnNewOrder.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnNewOrderActionPerformed(evt);

}

});

getContentPane().add(btnNewOrder, new org.netbeans.lib.awtextra.AbsoluteConstraints(470, 380, 100, 25));

tblOrders.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

" Order Number", "Starter", "Main", "Dessert", "Drink", "Price"

}

) {

boolean[] canEdit = new boolean [] {

false, false, false, false, false, false

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

tblOrders.setGridColor(new java.awt.Color(255, 255, 255));

tblOrders.setRowSelectionAllowed(false);

tblOrders.getTableHeader().setReorderingAllowed(false);

tblOrders.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

tblOrdersMouseClicked(evt);

}

});

jScrollPane2.setViewportView(tblOrders);

if (tblOrders.getColumnModel().getColumnCount() > 0) {

tblOrders.getColumnModel().getColumn(0).setResizable(false);

tblOrders.getColumnModel().getColumn(1).setResizable(false);

tblOrders.getColumnModel().getColumn(2).setResizable(false);

tblOrders.getColumnModel().getColumn(3).setResizable(false);

tblOrders.getColumnModel().getColumn(4).setResizable(false);

tblOrders.getColumnModel().getColumn(4).setPreferredWidth(30);

}

getContentPane().add(jScrollPane2, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 190, 560, 183));

btnRemoveOrder.setBackground(new java.awt.Color(255, 255, 255));

btnRemoveOrder.setText("Remove Order");

btnRemoveOrder.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnRemoveOrderActionPerformed(evt);

}

});

getContentPane().add(btnRemoveOrder, new org.netbeans.lib.awtextra.AbsoluteConstraints(350, 380, 110, 25));

btnAdmin.setBackground(new java.awt.Color(255, 255, 255));

btnAdmin.setText("Admin");

btnAdmin.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnAdminActionPerformed(evt);

}

});

getContentPane().add(btnAdmin, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 380, 100, 25));

jLabel3.setFont(new java.awt.Font("Microsoft Sans Serif", 1, 14)); // NOI18N

jLabel3.setForeground(new java.awt.Color(255, 255, 255));

jLabel3.setText("Individual Orders");

getContentPane().add(jLabel3, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 170, -1, -1));

jLabel2.setFont(new java.awt.Font("Microsoft Sans Serif", 1, 14)); // NOI18N

jLabel2.setForeground(new java.awt.Color(255, 255, 255));

jLabel2.setText("Order Summary");

getContentPane().add(jLabel2, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 10, -1, -1));

jLabel1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Grey-Background-GREY.jpg"))); // NOI18N

jLabel1.setName(""); // NOI18N

getContentPane().add(jLabel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(-7, -10, 590, 420));

pack();

}// </editor-fold>

private void btnNewOrderActionPerformed(java.awt.event.ActionEvent evt) {

Order o = new Order();//creates 'o', a new object based off the Order class

//controlling visibiiilty

o.setVisible(true);

this.setVisible(false);

}

private void btnRemoveOrderActionPerformed(java.awt.event.ActionEvent evt) {

int row = tblBill.getSelectedRow();//creates row variable, assigns it to current row

String selected = tblBill.getValueAt(row, 0).toString();//assigns selected variable to the selected order id for deletion purposes

try{

//creates a connection, creates and runs sql queries to delete the selected order from both tables

a.createConnection();

PreparedStatement ps = a.conn.prepareStatement("delete from Price\_Table where OrderID='"+selected+"' ");

PreparedStatement ps1 = a.conn.prepareStatement("delete from ORDER\_Table where OrderID='"+selected+"' ");

ps.executeUpdate();

ps1.executeUpdate();

//creates two variables to delete the current row through for loops

int rowCount = model.getRowCount();

int rowCount1 = model1.getRowCount();

for (int i = rowCount - 1; i >= 0; i--) {

model.removeRow(i);

}

for (int ii = rowCount1 - 1; ii >= 0; ii--) {

model1.removeRow(ii);

}

//repopulates the tables by re calling the methods

populateTable();

populateTable1();

//restricts the possibility of deleting an order which isn't selected

btnRemoveOrder.setEnabled(false);

//confirms deletion

JOptionPane.showMessageDialog(null, "Deleted");

}

//cathes if thrown

catch(SQLException sqlExcept){

sqlExcept.printStackTrace();

}

}

private void tblBillMouseClicked(java.awt.event.MouseEvent evt) {

//allows for the possibility of deletion by enabling the button

btnRemoveOrder.setEnabled(true);

}

private void btnAdminActionPerformed(java.awt.event.ActionEvent evt) {

//creates a new instnace of the admin password frame and sets it to visible

AdminPassword ap = new AdminPassword();

ap.setVisible(true);

this.setVisible(false);

}

private void tblOrdersMouseClicked(java.awt.event.MouseEvent evt) {

}

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Homepage.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Homepage.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Homepage.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Homepage.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new Homepage().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton btnAdmin;

private javax.swing.JButton btnNewOrder;

private javax.swing.JButton btnRemoveOrder;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JScrollPane jScrollPane2;

private javax.swing.JTable tblBill;

private javax.swing.JTable tblOrders;

// End of variables declaration

}

**Main.java**

class Main {

public Main()//constructor

{

}

//variable declaration

private String Main, MainPrice;

//getters and setters

public String getMain()

{

return Main;

}

public String getMainPrice()

{

return MainPrice;

}

protected void setMain(String iMain)

{

this.Main = iMain;

}

protected void setMainPrice(String iMainPrice)

{

this.MainPrice = iMainPrice;

}

}

**Menuclass.java**

public class MenuClass {

//varaibl declaration

private String Main, MainPrice, Dessert, DessertPrice, Drink, DrinkPrice;

public MenuClass(){//constructor

}

//getters and setters

public String getMain()

{

return Main;

}

public String getMainPrice()

{

return MainPrice;

}

public String getDessert()

{

return Dessert;

}

public String getDessertPrice()

{

return DessertPrice;

}

public String getDrink()

{

return Drink;

}

public String getDrinkPrice()

{

return DrinkPrice;

}

protected void setMain(String iMain)

{

this.Main = iMain;

}

protected void setMainPrice(String iMainPrice)

{

this.MainPrice = iMainPrice;

}

protected void setDessert(String iDessert)

{

this.Dessert = iDessert;

}

protected void setDessertPrice(String iDessertPrice)

{

this.DessertPrice = iDessertPrice;

}

protected void setDrink(String iDrink)

{

this.Drink = iDrink;

}

protected void setDrinkPrice(String iDrinkPrice)

{

this.DrinkPrice = iDrinkPrice;

}

}

**Menueditor.java**

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.ResultSetMetaData;

import java.sql.SQLException;

import javax.swing.JOptionPane;

import javax.swing.event.DocumentEvent;

import javax.swing.event.DocumentListener;

import javax.swing.table.DefaultTableModel;

public class MenuEditor extends javax.swing.JFrame {

/\*\*

\* Creates new form MenuEditor

\*/

//declaring a new instance of our database class

database a = new database();

//Declaring a new instance of classes

Starter s = new Starter();

Main m = new Main();

Dessert ds = new Dessert();

Drink dr = new Drink();

//declaring our table models which pertain to table structures

DefaultTableModel Startermodel;

DefaultTableModel Mainmodel;

DefaultTableModel Dessertmodel;

DefaultTableModel Drinkmodel;

public MenuEditor() {

initComponents();

//a.dropTable();

//a.createStarterTable();

//a.createMainTable();

//a.createDessertTable();

//a.createDrinkTable();

//assigning the default table model to our tables

Startermodel = (DefaultTableModel) tblStarters.getModel();

Mainmodel = (DefaultTableModel) tblMains.getModel();

Dessertmodel = (DefaultTableModel) tblDesserts.getModel();

Drinkmodel = (DefaultTableModel) tblDrinks.getModel();

//calling the method which will populate tables upon initialization

populateStarterTable();

populateMainTable();

populateDessertTable();

populateDrinkTable();

//Disabling buttons for validation purposes

btnAdd.setEnabled(false);

btnAddMain.setEnabled(false);

btnAddDessert.setEnabled(false);

btnAddDrink.setEnabled(false);

btnDelete.setEnabled(false);

btnDeleteMain.setEnabled(false);

btnDeleteDessert.setEnabled(false);

btnDeleteDrink.setEnabled(false);

//Document Listeners for validation, buttons become enabled when certain conditions are met

//Code manipulated from http://stackoverflow.com/questions/10848335/how-to-implement-documentlistener

txtStarter.getDocument().addDocumentListener(new DocumentListener()

{

@Override

public void changedUpdate(DocumentEvent e){

StarterlengthCheck();

}

@Override

public void insertUpdate(DocumentEvent e) {

StarterlengthCheck();

}

@Override

public void removeUpdate(DocumentEvent e) {

StarterlengthCheck();

}

});

txtStarterPrice.getDocument().addDocumentListener(new DocumentListener()

{

@Override

public void changedUpdate(DocumentEvent e){

StarterlengthCheck();

}

@Override

public void insertUpdate(DocumentEvent e) {

StarterlengthCheck();

}

@Override

public void removeUpdate(DocumentEvent e) {

StarterlengthCheck();

}

});

txtMain.getDocument().addDocumentListener(new DocumentListener() {

@Override

public void changedUpdate(DocumentEvent e){

MainlengthCheck();

}

@Override

public void insertUpdate(DocumentEvent e) {

MainlengthCheck();

}

@Override

public void removeUpdate(DocumentEvent e) {

MainlengthCheck();

}

});

txtMainPrice.getDocument().addDocumentListener(new DocumentListener() {

@Override

public void changedUpdate(DocumentEvent e){

MainlengthCheck();

}

@Override

public void insertUpdate(DocumentEvent e) {

MainlengthCheck();

}

@Override

public void removeUpdate(DocumentEvent e) {

MainlengthCheck();

}

});

txtDessert.getDocument().addDocumentListener(new DocumentListener()

{

@Override

public void changedUpdate(DocumentEvent e){

DessertlengthCheck();

}

@Override

public void insertUpdate(DocumentEvent e) {

DessertlengthCheck();

}

@Override

public void removeUpdate(DocumentEvent e) {

DessertlengthCheck();

}

});

txtDessertPrice.getDocument().addDocumentListener(new DocumentListener()

{

@Override

public void changedUpdate(DocumentEvent e){

DessertlengthCheck();

}

@Override

public void insertUpdate(DocumentEvent e) {

DessertlengthCheck();

}

@Override

public void removeUpdate(DocumentEvent e) {

DessertlengthCheck();

}

});

txtDrink.getDocument().addDocumentListener(new DocumentListener() {

@Override

public void changedUpdate(DocumentEvent e){

DrinklengthCheck();

}

@Override

public void insertUpdate(DocumentEvent e) {

DrinklengthCheck();

}

@Override

public void removeUpdate(DocumentEvent e) {

DrinklengthCheck();

}

});

txtDrinkPrice.getDocument().addDocumentListener(new DocumentListener() {

@Override

public void changedUpdate(DocumentEvent e){

DrinklengthCheck();

}

@Override

public void insertUpdate(DocumentEvent e) {

DrinklengthCheck();

}

@Override

public void removeUpdate(DocumentEvent e) {

DrinklengthCheck();

}

});

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

//populate the table for the starters

public void populateStarterTable()

{

String Starter;

String StarterPrice;

int r =0;

try

{

a.createConnection();//creates connection to the database

//creates and runs an sql statemtnt to display all info from the 'StarterTable' variable

a.stmt = a.conn.createStatement();

ResultSet results = a.stmt.executeQuery("select \* from " + a.StarterTable);

//loop will run as long as there is more data to read from the DB, assigning corresponding values accordingly

while(results.next())

{

Starter = results.getString(1);

StarterPrice = results.getString(2);

//inserts a new row into the table containing an order object with the following properties

Startermodel.insertRow(Startermodel.getRowCount(), new Object[]{Starter, StarterPrice});

r++;

}//end while loop

//closes and 'shuts down any open streams and connections

results.close();

a.stmt.close();

a.shutdown();

}

catch (SQLException sqlExcept )

{

sqlExcept.printStackTrace();

}

}

private void insertStarter(Starter s)

{

a.createConnection();//Creates a connection to the database

try

{

a.stmt = a.conn.createStatement();//creates and runs an sql statemtnt to insert all info from the textfields

a.stmt.execute("insert into " + a.StarterTable + " values ('" +

s.getStarter() + "','" + s.getStarterPrice()+"')");

a.stmt.close();

}

catch (SQLException sqlExcept)

{

sqlExcept.printStackTrace();

}

a.shutdown();//closes Connection

}

//method used in the document Listener for validation purposes

public void StarterlengthCheck(){

if(txtStarter.getText().equals("") || txtStarterPrice.getText().equals("")){

btnAdd.setEnabled(false);

btnDelete.setEnabled(false);

}

else if(txtStarter.getText().length()> 100 || txtStarter.getText().length()>100){

btnAdd.setEnabled(false);

}

else{

btnAdd.setEnabled(true);

}

}

@SuppressWarnings("unchecked")

public void populateMainTable()

{

String Main;

String MainPrice;

int r =0;

try

{

a.createConnection();//creates a connection to the database

//creates and runs an sql statemtnt to display all info from the 'StarterTable' variable

a.stmt = a.conn.createStatement();

ResultSet results = a.stmt.executeQuery("select \* from " + a.MainTable);

ResultSetMetaData rsmd = results.getMetaData();

//Fills the Table until it reaches the end of the database

while(results.next())

{

Main = results.getString(1);

MainPrice = results.getString(2);

//inserts a new row into the table containing an order object with the following properties

Mainmodel.insertRow(Mainmodel.getRowCount(), new Object[]{Main, MainPrice});

r++;

}//end of loop

//closes and 'shuts down any open streams and connections

results.close();

a.stmt.close();

a.shutdown();

}

catch (SQLException sqlExcept )

{

sqlExcept.printStackTrace();

}

}

private void insertMain(Main m)

{

a.createConnection();//creates a connection to the database

try

{

a.stmt = a.conn.createStatement();//creates and exectutes an sql statement to insert data to the database

a.stmt.execute("insert into " + a.MainTable + " values ('" +

m.getMain() + "','" + m.getMainPrice()+"')");

a.stmt.close();

}

catch (SQLException sqlExcept)

{

sqlExcept.printStackTrace();

}

a.shutdown();

}

//Creates method used in document listener

public void MainlengthCheck(){

if(txtMain.getText().equals("") || txtMainPrice.getText().equals("")){

btnAddMain.setEnabled(false);

btnDeleteMain.setEnabled(false);

}

else if(txtMain.getText().length()> 100 || txtMain.getText().length()>100){

btnAddMain.setEnabled(false);

}

else{

btnAddMain.setEnabled(true);

}

}

@SuppressWarnings("unchecked")

//Doing same thing as other populate methods

public void populateDessertTable()

{

String Dessert;

String DessertPrice;

int r =0;

try

{

a.createConnection();

a.stmt = a.conn.createStatement();

ResultSet results = a.stmt.executeQuery("select \* from " + a.DessertTable);

ResultSetMetaData rsmd = results.getMetaData();

while(results.next())

{

Dessert = results.getString(1);

DessertPrice = results.getString(2);

Dessertmodel.insertRow(Dessertmodel.getRowCount(), new Object[]{Dessert, DessertPrice});

r++;

}//end of loop

results.close();

a.stmt.close();

a.shutdown();

}

catch (SQLException sqlExcept )

{

sqlExcept.printStackTrace();

} catch(IllegalArgumentException a){

JOptionPane.showMessageDialog(this, "You have entered an illegal character", "Error", JOptionPane.ERROR\_MESSAGE);

}

}

//inserts data into the Dessert table in the same way as the other methods

private void insertDessert(Dessert ds)

{

a.createConnection();

try

{

a.stmt = a.conn.createStatement();

a.stmt.execute("insert into " + a.DessertTable + " values ('" +

ds.getDessert() + "','" + ds.getDessertPrice()+"')");

a.stmt.close();

}

catch (SQLException sqlExcept)

{

sqlExcept.printStackTrace();

}

a.shutdown();

}

//method used in document listener

public void DessertlengthCheck(){

if(txtDessert.getText().equals("") || txtDessertPrice.getText().equals("")){

btnAddDessert.setEnabled(false);

btnDeleteDessert.setEnabled(false);

}

else if(txtDessert.getText().length()> 100 || txtDessert.getText().length()>100){

btnAddDessert.setEnabled(false);

}

else{

btnAddDessert.setEnabled(true);

}

}

@SuppressWarnings("unchecked")

//Populates Drink table the same way as previous methods

public void populateDrinkTable()

{

String Drink;

String DrinkPrice;

int r =0;

try

{

a.createConnection();

a.stmt = a.conn.createStatement();

ResultSet results = a.stmt.executeQuery("select \* from " + a.DrinkTable);

ResultSetMetaData rsmd = results.getMetaData();

while(results.next())

{

Drink = results.getString(1);

DrinkPrice = results.getString(2);

Drinkmodel.insertRow(Drinkmodel.getRowCount(), new Object[]{Drink, DrinkPrice});

r++;

}

results.close();

a.stmt.close();

a.shutdown();

}

catch (SQLException sqlExcept )

{

sqlExcept.printStackTrace();

}

}

//inserts drink info into drink table the same way as previous methods

private void insertDrink(Drink dr)

{

a.createConnection();

try

{

a.stmt = a.conn.createStatement();

a.stmt.execute("insert into " + a.DrinkTable + " values ('" +

dr.getDrink() + "','" + dr.getDrinkPrice()+"')");

a.stmt.close();

}

catch (SQLException sqlExcept)

{

sqlExcept.printStackTrace();

}

a.shutdown();

}

//Method used in document listener

public void DrinklengthCheck(){

if(txtDrink.getText().equals("") || txtDrinkPrice.getText().equals("")){

btnAddDrink.setEnabled(false);

btnDeleteDrink.setEnabled(false);

}

else if(txtDrink.getText().length()> 100 || txtDrink.getText().length()>100){

btnAddDrink.setEnabled(false);

}

else{

btnAddDrink.setEnabled(true);

}

}

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

btnBack = new javax.swing.JButton();

jScrollPane1 = new javax.swing.JScrollPane();

tblStarters = new javax.swing.JTable();

jScrollPane2 = new javax.swing.JScrollPane();

tblMains = new javax.swing.JTable();

jScrollPane3 = new javax.swing.JScrollPane();

tblDesserts = new javax.swing.JTable();

jScrollPane4 = new javax.swing.JScrollPane();

tblDrinks = new javax.swing.JTable();

txtStarter = new javax.swing.JTextField();

txtStarterPrice = new javax.swing.JTextField();

btnAdd = new javax.swing.JButton();

btnDelete = new javax.swing.JButton();

txtMainPrice = new javax.swing.JTextField();

btnAddMain = new javax.swing.JButton();

btnDeleteMain = new javax.swing.JButton();

txtMain = new javax.swing.JTextField();

txtDessertPrice = new javax.swing.JTextField();

btnAddDessert = new javax.swing.JButton();

btnDeleteDessert = new javax.swing.JButton();

txtDessert = new javax.swing.JTextField();

btnAddDrink = new javax.swing.JButton();

txtDrinkPrice = new javax.swing.JTextField();

btnDeleteDrink = new javax.swing.JButton();

txtDrink = new javax.swing.JTextField();

btnDeselectS = new javax.swing.JButton();

btnDeselectM = new javax.swing.JButton();

btnDeselectDes = new javax.swing.JButton();

btnDeselectDr = new javax.swing.JButton();

jLabel1 = new javax.swing.JLabel();

jLabel9 = new javax.swing.JLabel();

jLabel10 = new javax.swing.JLabel();

jLabel11 = new javax.swing.JLabel();

jLabel16 = new javax.swing.JLabel();

jLabel17 = new javax.swing.JLabel();

jLabel18 = new javax.swing.JLabel();

jLabel19 = new javax.swing.JLabel();

btnBack1 = new javax.swing.JButton();

jLabel2 = new javax.swing.JLabel();

btnBack.setText("Back to Home");

btnBack.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnBackActionPerformed(evt);

}

});

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

getContentPane().setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

tblStarters.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Starters", "Price"

}

) {

boolean[] canEdit = new boolean [] {

false, false

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

tblStarters.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

tblStartersMouseClicked(evt);

}

});

jScrollPane1.setViewportView(tblStarters);

if (tblStarters.getColumnModel().getColumnCount() > 0) {

tblStarters.getColumnModel().getColumn(0).setResizable(false);

tblStarters.getColumnModel().getColumn(1).setMinWidth(70);

tblStarters.getColumnModel().getColumn(1).setPreferredWidth(70);

tblStarters.getColumnModel().getColumn(1).setMaxWidth(70);

}

getContentPane().add(jScrollPane1, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 11, 242, 138));

tblMains.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Main Courses", "Price"

}

) {

boolean[] canEdit = new boolean [] {

false, false

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

tblMains.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

tblMainsMouseClicked(evt);

}

});

jScrollPane2.setViewportView(tblMains);

if (tblMains.getColumnModel().getColumnCount() > 0) {

tblMains.getColumnModel().getColumn(0).setResizable(false);

tblMains.getColumnModel().getColumn(1).setMinWidth(70);

tblMains.getColumnModel().getColumn(1).setPreferredWidth(70);

tblMains.getColumnModel().getColumn(1).setMaxWidth(70);

}

getContentPane().add(jScrollPane2, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 160, 242, 138));

tblDesserts.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Desserts", "Price"

}

) {

boolean[] canEdit = new boolean [] {

false, false

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

tblDesserts.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

tblDessertsMouseClicked(evt);

}

});

jScrollPane3.setViewportView(tblDesserts);

if (tblDesserts.getColumnModel().getColumnCount() > 0) {

tblDesserts.getColumnModel().getColumn(0).setResizable(false);

tblDesserts.getColumnModel().getColumn(1).setMinWidth(70);

tblDesserts.getColumnModel().getColumn(1).setPreferredWidth(70);

tblDesserts.getColumnModel().getColumn(1).setMaxWidth(70);

}

getContentPane().add(jScrollPane3, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 309, 242, 138));

tblDrinks.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Drinks", "Price"

}

) {

boolean[] canEdit = new boolean [] {

false, false

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

tblDrinks.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

tblDrinksMouseClicked(evt);

}

});

jScrollPane4.setViewportView(tblDrinks);

if (tblDrinks.getColumnModel().getColumnCount() > 0) {

tblDrinks.getColumnModel().getColumn(0).setResizable(false);

tblDrinks.getColumnModel().getColumn(1).setMinWidth(70);

tblDrinks.getColumnModel().getColumn(1).setPreferredWidth(70);

tblDrinks.getColumnModel().getColumn(1).setMaxWidth(70);

}

getContentPane().add(jScrollPane4, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 458, 242, 138));

getContentPane().add(txtStarter, new org.netbeans.lib.awtextra.AbsoluteConstraints(381, 57, 150, -1));

getContentPane().add(txtStarterPrice, new org.netbeans.lib.awtextra.AbsoluteConstraints(381, 88, 150, -1));

btnAdd.setBackground(new java.awt.Color(255, 255, 255));

btnAdd.setText("Add Starter");

btnAdd.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnAddActionPerformed(evt);

}

});

getContentPane().add(btnAdd, new org.netbeans.lib.awtextra.AbsoluteConstraints(381, 120, 120, -1));

btnDelete.setBackground(new java.awt.Color(255, 255, 255));

btnDelete.setText("Delete Starter");

btnDelete.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnDeleteActionPerformed(evt);

}

});

getContentPane().add(btnDelete, new org.netbeans.lib.awtextra.AbsoluteConstraints(507, 120, 120, -1));

getContentPane().add(txtMainPrice, new org.netbeans.lib.awtextra.AbsoluteConstraints(381, 244, 150, -1));

btnAddMain.setBackground(new java.awt.Color(255, 255, 255));

btnAddMain.setText("Add Main");

btnAddMain.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnAddMainActionPerformed(evt);

}

});

getContentPane().add(btnAddMain, new org.netbeans.lib.awtextra.AbsoluteConstraints(381, 275, 120, -1));

btnDeleteMain.setBackground(new java.awt.Color(255, 255, 255));

btnDeleteMain.setText("Delete Order");

btnDeleteMain.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnDeleteMainActionPerformed(evt);

}

});

getContentPane().add(btnDeleteMain, new org.netbeans.lib.awtextra.AbsoluteConstraints(507, 275, 120, -1));

getContentPane().add(txtMain, new org.netbeans.lib.awtextra.AbsoluteConstraints(381, 213, 150, -1));

getContentPane().add(txtDessertPrice, new org.netbeans.lib.awtextra.AbsoluteConstraints(381, 377, 150, -1));

btnAddDessert.setBackground(new java.awt.Color(255, 255, 255));

btnAddDessert.setText("Add Dessert");

btnAddDessert.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnAddDessertActionPerformed(evt);

}

});

getContentPane().add(btnAddDessert, new org.netbeans.lib.awtextra.AbsoluteConstraints(381, 424, 120, -1));

btnDeleteDessert.setBackground(new java.awt.Color(255, 255, 255));

btnDeleteDessert.setText("Delete Dessert");

btnDeleteDessert.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnDeleteDessertActionPerformed(evt);

}

});

getContentPane().add(btnDeleteDessert, new org.netbeans.lib.awtextra.AbsoluteConstraints(507, 424, 120, -1));

getContentPane().add(txtDessert, new org.netbeans.lib.awtextra.AbsoluteConstraints(381, 346, 150, -1));

btnAddDrink.setBackground(new java.awt.Color(255, 255, 255));

btnAddDrink.setText("Add Drink");

btnAddDrink.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnAddDrinkActionPerformed(evt);

}

});

getContentPane().add(btnAddDrink, new org.netbeans.lib.awtextra.AbsoluteConstraints(381, 573, 120, -1));

getContentPane().add(txtDrinkPrice, new org.netbeans.lib.awtextra.AbsoluteConstraints(381, 538, 150, -1));

btnDeleteDrink.setBackground(new java.awt.Color(255, 255, 255));

btnDeleteDrink.setText("Delete Drink");

btnDeleteDrink.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnDeleteDrinkActionPerformed(evt);

}

});

getContentPane().add(btnDeleteDrink, new org.netbeans.lib.awtextra.AbsoluteConstraints(507, 573, 120, -1));

txtDrink.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

txtDrinkActionPerformed(evt);

}

});

getContentPane().add(txtDrink, new org.netbeans.lib.awtextra.AbsoluteConstraints(381, 507, 150, -1));

btnDeselectS.setBackground(new java.awt.Color(255, 255, 255));

btnDeselectS.setText("Deselect");

btnDeselectS.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnDeselectSActionPerformed(evt);

}

});

getContentPane().add(btnDeselectS, new org.netbeans.lib.awtextra.AbsoluteConstraints(537, 70, 86, -1));

btnDeselectM.setBackground(new java.awt.Color(255, 255, 255));

btnDeselectM.setText("Deselect");

btnDeselectM.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnDeselectMActionPerformed(evt);

}

});

getContentPane().add(btnDeselectM, new org.netbeans.lib.awtextra.AbsoluteConstraints(537, 225, 86, -1));

btnDeselectDes.setBackground(new java.awt.Color(255, 255, 255));

btnDeselectDes.setText("Deselect");

btnDeselectDes.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnDeselectDesActionPerformed(evt);

}

});

getContentPane().add(btnDeselectDes, new org.netbeans.lib.awtextra.AbsoluteConstraints(537, 359, 86, -1));

btnDeselectDr.setBackground(new java.awt.Color(255, 255, 255));

btnDeselectDr.setText("Deselect");

btnDeselectDr.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnDeselectDrActionPerformed(evt);

}

});

getContentPane().add(btnDeselectDr, new org.netbeans.lib.awtextra.AbsoluteConstraints(537, 522, 86, -1));

jLabel1.setBackground(new java.awt.Color(255, 255, 255));

jLabel1.setFont(new java.awt.Font("Microsoft Sans Serif", 0, 14)); // NOI18N

jLabel1.setForeground(new java.awt.Color(255, 255, 255));

jLabel1.setText("Starter");

getContentPane().add(jLabel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(334, 58, -1, -1));

jLabel9.setFont(new java.awt.Font("Microsoft Sans Serif", 0, 14)); // NOI18N

jLabel9.setForeground(new java.awt.Color(255, 255, 255));

jLabel9.setText("Starter Price");

getContentPane().add(jLabel9, new org.netbeans.lib.awtextra.AbsoluteConstraints(298, 89, -1, -1));

jLabel10.setFont(new java.awt.Font("Microsoft Sans Serif", 0, 14)); // NOI18N

jLabel10.setForeground(new java.awt.Color(255, 255, 255));

jLabel10.setText("Main Course");

getContentPane().add(jLabel10, new org.netbeans.lib.awtextra.AbsoluteConstraints(298, 214, -1, -1));

jLabel11.setFont(new java.awt.Font("Microsoft Sans Serif", 0, 14)); // NOI18N

jLabel11.setForeground(new java.awt.Color(255, 255, 255));

jLabel11.setText("Dessert");

getContentPane().add(jLabel11, new org.netbeans.lib.awtextra.AbsoluteConstraints(328, 347, -1, -1));

jLabel16.setFont(new java.awt.Font("Microsoft Sans Serif", 0, 14)); // NOI18N

jLabel16.setForeground(new java.awt.Color(255, 255, 255));

jLabel16.setText("Main Course Price");

getContentPane().add(jLabel16, new org.netbeans.lib.awtextra.AbsoluteConstraints(262, 245, -1, -1));

jLabel17.setFont(new java.awt.Font("Microsoft Sans Serif", 0, 14)); // NOI18N

jLabel17.setForeground(new java.awt.Color(255, 255, 255));

jLabel17.setText("Dessert Price");

getContentPane().add(jLabel17, new org.netbeans.lib.awtextra.AbsoluteConstraints(292, 378, -1, -1));

jLabel18.setFont(new java.awt.Font("Microsoft Sans Serif", 0, 14)); // NOI18N

jLabel18.setForeground(new java.awt.Color(255, 255, 255));

jLabel18.setText("Drink");

getContentPane().add(jLabel18, new org.netbeans.lib.awtextra.AbsoluteConstraints(344, 508, -1, -1));

jLabel19.setFont(new java.awt.Font("Microsoft Sans Serif", 0, 14)); // NOI18N

jLabel19.setForeground(new java.awt.Color(255, 255, 255));

jLabel19.setText("Drink Price");

getContentPane().add(jLabel19, new org.netbeans.lib.awtextra.AbsoluteConstraints(308, 539, -1, -1));

btnBack1.setBackground(new java.awt.Color(255, 255, 255));

btnBack1.setText("Back to Home");

btnBack1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnBack1ActionPerformed(evt);

}

});

getContentPane().add(btnBack1, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 607, 110, 25));

jLabel2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Grey-Background-GREY.jpg"))); // NOI18N

getContentPane().add(jLabel2, new org.netbeans.lib.awtextra.AbsoluteConstraints(0, 0, 650, 640));

pack();

}// </editor-fold>

//adding data to the Starter table

private void btnAddActionPerformed(java.awt.event.ActionEvent evt) {

try{

Double.parseDouble(txtStarterPrice.getText());//Checking if the text is double in the price text field

//Inserts data into table

Startermodel.insertRow(Startermodel.getRowCount(), new Object[]{txtStarter.getText(), txtStarterPrice.getText()});

//Gets info from textfields to be used to populate database table

s.setStarter(txtStarter.getText());

s.setStarterPrice(txtStarterPrice.getText());

//Clears text fields

txtStarter.setText(null);

txtStarterPrice.setText(null);

//Calls method insert starter and passes the textfield values

insertStarter(s);

}

catch(Exception c){

//catched the error if a number isn't entered in the price text field

JOptionPane.showMessageDialog(null, "You can only enter numbers into the price textfield");

txtStarter.setText(null);

txtStarterPrice.setText(null);

btnAdd.setEnabled(false);

}

}

private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {

//Sets a variable equal to the selected row

int row = tblStarters.getSelectedRow();

//Sets the variable selected equal to firstvalue in selected row and sets it to a string

String selected = tblStarters.getValueAt(row, 0).toString();

if (row >= 0) {

//removes row from table

Startermodel.removeRow(row);

try{

a.createConnection();//creates connection

//prepares and executes sql statement to delete from the database

PreparedStatement ps = a.conn.prepareStatement("delete from Starter\_Table where Starter='"+selected+"' ");

ps.executeUpdate();

txtStarter.setText(null);

txtStarterPrice.setText(null);

JOptionPane.showMessageDialog(null, "Deleted");

}

catch(SQLException sqlExcept){

sqlExcept.printStackTrace();

}

}

}

private void tblStartersMouseClicked(java.awt.event.MouseEvent evt) {

//sets the selected row equal to the variable

int row = tblStarters.getSelectedRow();

//Sets the variable 'selected' equal to firstvalue in selected row and sets it to a string

String selected = tblStarters.getValueAt(row, 0).toString();

DefaultTableModel model= (DefaultTableModel)tblStarters.getModel();

try{

if(selected !=null ){

//Puts the selected row text into text fields

txtStarter.setText(String.valueOf(model.getValueAt(tblStarters.getSelectedRow(), 0)));

txtStarterPrice.setText(String.valueOf(model.getValueAt(tblStarters.getSelectedRow(), 1)));

//Disable buttons for validation purposes

btnDelete.setEnabled(true);

btnAdd.setEnabled(false);

}

} catch(NullPointerException a){

JOptionPane.showMessageDialog(null, "Deleted");

}

}

//does the same as previous add button

private void btnAddMainActionPerformed(java.awt.event.ActionEvent evt) {

try{

Double.parseDouble(txtMainPrice.getText());

Mainmodel.insertRow(Mainmodel.getRowCount(), new Object[]{txtMain.getText(), txtMainPrice.getText()});

m.setMain(txtMain.getText());

m.setMainPrice(txtMainPrice.getText());

txtMain.setText(null);

txtMainPrice.setText(null);

insertMain(m);

}

catch(Exception c){

JOptionPane.showMessageDialog(null, "You can only enter numbers into the price textfield");

txtMain.setText(null);

txtMainPrice.setText(null);

btnAddMain.setEnabled(false);

}

}

//does the same as previous delete buttons

private void btnDeleteMainActionPerformed(java.awt.event.ActionEvent evt) {

int row = tblMains.getSelectedRow();

String selected = tblMains.getValueAt(row, 0).toString();

if (row >= 0) {

Mainmodel.removeRow(row);

try{

a.createConnection();

PreparedStatement ps = a.conn.prepareStatement("delete from Mains\_Table where Main='"+selected+"' ");

ps.executeUpdate();

txtMain.setText(null);

txtMainPrice.setText(null);

JOptionPane.showMessageDialog(null, "Deleted");

}

catch(SQLException sqlExcept){

sqlExcept.printStackTrace();

}

}

}

//does the same as previous add buttons

private void btnAddDessertActionPerformed(java.awt.event.ActionEvent evt) {

try{

Double.parseDouble(txtDessertPrice.getText());

Dessertmodel.insertRow(Dessertmodel.getRowCount(), new Object[]{txtDessert.getText(), txtDessertPrice.getText()});

ds.setDessert(txtDessert.getText());

ds.setDessertPrice(txtDessertPrice.getText());

txtDessert.setText(null);

txtDessertPrice.setText(null);

insertDessert(ds);

}

catch(Exception c){

JOptionPane.showMessageDialog(null, "You can only enter numbers into the price textfield");

txtDessert.setText(null);

txtDessertPrice.setText(null);

btnAddDessert.setEnabled(false);

}

}

//does the same as previous delete buttons

private void btnDeleteDessertActionPerformed(java.awt.event.ActionEvent evt) {

int row = tblDesserts.getSelectedRow();

String selected = tblDesserts.getValueAt(row, 0).toString();

if (row >= 0) {

Dessertmodel.removeRow(row);

try{

a.createConnection();

PreparedStatement ps = a.conn.prepareStatement("delete from Dessert\_Table where Dessert='"+selected+"' ");

ps.executeUpdate();

txtDessert.setText(null);

txtDessertPrice.setText(null);

JOptionPane.showMessageDialog(null, "Deleted");

}

catch(SQLException sqlExcept){

sqlExcept.printStackTrace();

}

}

}

//does the same as previous add buttons

private void btnAddDrinkActionPerformed(java.awt.event.ActionEvent evt) {

try{

Double.parseDouble(txtDrinkPrice.getText());

Drinkmodel.insertRow(Drinkmodel.getRowCount(), new Object[]{txtDrink.getText(), txtDrinkPrice.getText()});

dr.setDrink(txtDrink.getText());

dr.setDrinkPrice(txtDrinkPrice.getText());

txtDrink.setText(null);

txtDrinkPrice.setText(null);

insertDrink(dr);

}

catch(Exception c){

JOptionPane.showMessageDialog(null, "You can only enter numbers into the price textfield");

txtDrink.setText(null);

txtDrinkPrice.setText(null);

btnAddDrink.setEnabled(false);

}

}

//does the same as previous delete buttons

private void btnDeleteDrinkActionPerformed(java.awt.event.ActionEvent evt) {

int row = tblDrinks.getSelectedRow();

String selected = tblDrinks.getValueAt(row, 0).toString();

if (row >= 0) {

Drinkmodel.removeRow(row);

try{

a.createConnection();

PreparedStatement ps = a.conn.prepareStatement("delete from Drink\_Table where Drink='"+selected+"' ");

ps.executeUpdate();

txtDrink.setText(null);

txtDrinkPrice.setText(null);

JOptionPane.showMessageDialog(null, "Deleted");

}

catch(SQLException sqlExcept){

sqlExcept.printStackTrace();

}

}

}

//does the same as previous table click events

private void tblMainsMouseClicked(java.awt.event.MouseEvent evt) {

int row = tblMains.getSelectedRow();

String selected = tblMains.getValueAt(row, 0).toString();

DefaultTableModel model= (DefaultTableModel)tblMains.getModel();

try{

if(selected !=null ){

txtMain.setText(String.valueOf(model.getValueAt(tblMains.getSelectedRow(), 0)));

txtMainPrice.setText(String.valueOf(model.getValueAt(tblMains.getSelectedRow(), 1)));

btnDeleteMain.setEnabled(true);

btnAddMain.setEnabled(false);

}

} catch(NullPointerException a){

JOptionPane.showMessageDialog(null, "Deleted");

}

}

//does the same as previous table click events

private void tblDessertsMouseClicked(java.awt.event.MouseEvent evt) {

int row = tblDesserts.getSelectedRow();

String selected = tblDesserts.getValueAt(row, 0).toString();

DefaultTableModel model= (DefaultTableModel)tblDesserts.getModel();

try{

if(selected !=null ){

txtDessert.setText(String.valueOf(model.getValueAt(tblDesserts.getSelectedRow(), 0)));

txtDessertPrice.setText(String.valueOf(model.getValueAt(tblDesserts.getSelectedRow(), 1)));

btnDeleteDessert.setEnabled(true);

btnAddDessert.setEnabled(false);

}

} catch(NullPointerException a){

JOptionPane.showMessageDialog(null, "Deleted");

}

}

//does the same as previous table click events

private void tblDrinksMouseClicked(java.awt.event.MouseEvent evt) {

int row = tblDrinks.getSelectedRow();

String selected = tblDrinks.getValueAt(row, 0).toString();

DefaultTableModel model= (DefaultTableModel)tblDrinks.getModel();

try{

if(selected !=null ){

txtDrink.setText(String.valueOf(model.getValueAt(tblDrinks.getSelectedRow(), 0)));

txtDrinkPrice.setText(String.valueOf(model.getValueAt(tblDrinks.getSelectedRow(), 1)));

btnDeleteDrink.setEnabled(true);

btnAddDrink.setEnabled(false);

}

} catch(NullPointerException a){

JOptionPane.showMessageDialog(null, "Deleted");

}

}

private void txtDrinkActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

//deselects the selected row from the table

private void btnDeselectSActionPerformed(java.awt.event.ActionEvent evt) {

tblStarters.getSelectionModel().clearSelection();

txtStarter.setText(null);

txtStarterPrice.setText(null);

}

//deselects the selected row from the table

private void btnDeselectMActionPerformed(java.awt.event.ActionEvent evt) {

tblMains.getSelectionModel().clearSelection();

txtMain.setText(null);

txtMainPrice.setText(null);

}

//deselects the selected row from the table

private void btnDeselectDesActionPerformed(java.awt.event.ActionEvent evt) {

tblDesserts.getSelectionModel().clearSelection();

txtDessert.setText(null);

txtDessertPrice.setText(null);

}

//deselects the selected row from the table

private void btnDeselectDrActionPerformed(java.awt.event.ActionEvent evt) {

tblDrinks.getSelectionModel().clearSelection();

txtDrink.setText(null);

txtDrinkPrice.setText(null);

}

//Navigates back to previous form

private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {

Homepage h = new Homepage();

h.setVisible(true);

this.setVisible(false);

}

private void btnBack1ActionPerformed(java.awt.event.ActionEvent evt) {

Frame h = new Frame();

h.setVisible(true);

this.setVisible(false);

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(MenuEditor.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(MenuEditor.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(MenuEditor.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(MenuEditor.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new MenuEditor().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton btnAdd;

private javax.swing.JButton btnAddDessert;

private javax.swing.JButton btnAddDrink;

private javax.swing.JButton btnAddMain;

private javax.swing.JButton btnBack;

private javax.swing.JButton btnBack1;

private javax.swing.JButton btnDelete;

private javax.swing.JButton btnDeleteDessert;

private javax.swing.JButton btnDeleteDrink;

private javax.swing.JButton btnDeleteMain;

private javax.swing.JButton btnDeselectDes;

private javax.swing.JButton btnDeselectDr;

private javax.swing.JButton btnDeselectM;

private javax.swing.JButton btnDeselectS;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel10;

private javax.swing.JLabel jLabel11;

private javax.swing.JLabel jLabel16;

private javax.swing.JLabel jLabel17;

private javax.swing.JLabel jLabel18;

private javax.swing.JLabel jLabel19;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel9;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JScrollPane jScrollPane2;

private javax.swing.JScrollPane jScrollPane3;

private javax.swing.JScrollPane jScrollPane4;

private javax.swing.JTable tblDesserts;

private javax.swing.JTable tblDrinks;

private javax.swing.JTable tblMains;

private javax.swing.JTable tblStarters;

private javax.swing.JTextField txtDessert;

private javax.swing.JTextField txtDessertPrice;

private javax.swing.JTextField txtDrink;

private javax.swing.JTextField txtDrinkPrice;

private javax.swing.JTextField txtMain;

private javax.swing.JTextField txtMainPrice;

private javax.swing.JTextField txtStarter;

private javax.swing.JTextField txtStarterPrice;

// End of variables declaration

}

**Order.java**

import javax.swing.\*;

import java.io.\*;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.ResultSetMetaData;

import java.sql.SQLException;

import java.util.Random;

import javax.swing.event.DocumentEvent;

import javax.swing.event.DocumentListener;

import javax.swing.table.DefaultTableModel;

public class Order extends javax.swing.JFrame {

public double runningCost;

public double rowCost;

//Declaring a new instance of classes

Starter s = new Starter();

Main m = new Main();

Dessert ds = new Dessert();

Drink dr = new Drink();

TotalPrice p = new TotalPrice();

Person\_Order o = new Person\_Order();

//Declaring a new instance of database class

database a = new database();

//Declaring a new instance of random class to use for order number

Random rand = new Random();

int x = rand.nextInt(10000);

public String xx = Integer.toString(x);

//declaring our table models which pertain to table structures

DefaultTableModel model;

DefaultTableModel Startermodel;

DefaultTableModel Mainmodel;

DefaultTableModel Dessertmodel;

DefaultTableModel Drinkmodel;

public Order(){

initComponents();

//a.dropTable1();

//a.createTable1();

//a.dropTable2();

//a.createTable2();

//This will stop the user exiting the program mid order as this will cause an inconsistency in the database tables

setDefaultCloseOperation(WindowConstants.DO\_NOTHING\_ON\_CLOSE);

txtRowCost.setVisible(false);

Startermodel = (DefaultTableModel) tblStarters.getModel();

Mainmodel = (DefaultTableModel) tblMains.getModel();

Dessertmodel = (DefaultTableModel) tblDesserts.getModel();

Drinkmodel = (DefaultTableModel) tblDrinks.getModel();

txtFinish.setEnabled(false);

btnDelete.setEnabled(false);

model = (DefaultTableModel) tblOrder.getModel();

//populate combobox with employee names

fill();

//populate tables with database info

populateStarter();

populateMain();

populateDessert();

populateDrink();

//Document Listeners for validation, buttons become enabled when certain conditions are met

//Code manipulated from http://stackoverflow.com/questions/10848335/how-to-implement-documentlistener

txtTable.getDocument().addDocumentListener(new DocumentListener() {

@Override

public void changedUpdate(DocumentEvent e){

lengthCheck();

}

@Override

public void insertUpdate(DocumentEvent e) {

lengthCheck();

}

@Override

public void removeUpdate(DocumentEvent e) {

lengthCheck();

}

});

txtCost.getDocument().addDocumentListener(new DocumentListener() {

@Override

public void changedUpdate(DocumentEvent e){

lengthCheck();

}

@Override

public void insertUpdate(DocumentEvent e) {

lengthCheck();

}

@Override

public void removeUpdate(DocumentEvent e) {

lengthCheck();

}

});

}

//method to check text field for validation

public void lengthCheck(){

if(txtCost.getText().equals("")||txtTable.getText().equals("")||txtCost.getText().equals("0.0")){

txtFinish.setEnabled(false);

}

else{

btnBack.setEnabled(false);

txtFinish.setEnabled(true);

}

}

//method to fill combobox with employee names

private void fill(){

try{

a.createConnection();//creates connection

cbxEmployees.removeAllItems();//removes items from combobox

a.stmt = a.conn.createStatement();//creates statement for execution

ResultSet results = a.stmt.executeQuery("SELECT \* from " + a.tableName);

while(results.next()){

//populates combobox

cbxEmployees.addItem(results.getString("First\_Name"));

}

results.close();

a.stmt.close();

a.shutdown();

}

catch(Exception c){

c.printStackTrace();

}

}

//inserts order to tabel the same way as previous inserts

private void insertOrder(Person\_Order o)

{

a.createConnection();

try

{

a.stmt = a.conn.createStatement();

a.stmt.execute("insert into ORDER\_TABLE values ('" +

o.getOrderID() + "','" + o.getStarter() + "','" + o.getMain() + "','" + o.getDessert() + "','" + o.getDrink() + "','" + o.getPrice() + "')");

a.stmt.close();

}

catch (SQLException sqlExcept)

{

sqlExcept.printStackTrace();

}

a.shutdown();

}

//inserts order to tabel the same way as previous inserts

private void insertFinalOrder(TotalPrice p)

{

a.createConnection();

try

{

a.stmt = a.conn.createStatement();

a.stmt.execute("insert into Price\_TABLE values ('" +

p.getOrderID() + "','" + p.getTableNum() + "','" + p.getEmployee() + "','" + p.getTotalPrice() + "')");

a.stmt.close();

}

catch (SQLException sqlExcept)

{

sqlExcept.printStackTrace();

}

a.shutdown();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

txtStarterValue = new javax.swing.JTextField();

txtMainValue = new javax.swing.JTextField();

txtDessertValue = new javax.swing.JTextField();

txtCost = new javax.swing.JTextField();

btnAdd = new javax.swing.JButton();

txtDrinkValue = new javax.swing.JTextField();

jLabel1 = new javax.swing.JLabel();

jScrollPane4 = new javax.swing.JScrollPane();

tblOrder = new javax.swing.JTable();

cbxEmployees = new javax.swing.JComboBox();

txtRowCost = new javax.swing.JTextField();

txtFinish = new javax.swing.JButton();

txtTable = new javax.swing.JTextField();

jScrollPane5 = new javax.swing.JScrollPane();

tblDrinks = new javax.swing.JTable();

jScrollPane1 = new javax.swing.JScrollPane();

tblStarters = new javax.swing.JTable();

jScrollPane2 = new javax.swing.JScrollPane();

tblMains = new javax.swing.JTable();

jScrollPane3 = new javax.swing.JScrollPane();

tblDesserts = new javax.swing.JTable();

txtStarterPrice = new javax.swing.JTextField();

txtMainPrice = new javax.swing.JTextField();

txtDessertPrice = new javax.swing.JTextField();

txtDrinkPrice = new javax.swing.JTextField();

jLabel2 = new javax.swing.JLabel();

jLabel3 = new javax.swing.JLabel();

btnDelete = new javax.swing.JButton();

btnBack = new javax.swing.JButton();

jLabel4 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

getContentPane().setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

txtStarterValue.setEditable(false);

txtStarterValue.setText("-");

getContentPane().add(txtStarterValue, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 369, 188, -1));

txtMainValue.setEditable(false);

txtMainValue.setText("-");

txtMainValue.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

txtMainValueActionPerformed(evt);

}

});

getContentPane().add(txtMainValue, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 400, 188, -1));

txtDessertValue.setEditable(false);

txtDessertValue.setText("-");

getContentPane().add(txtDessertValue, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 431, 188, -1));

txtCost.setEditable(false);

txtCost.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

txtCostActionPerformed(evt);

}

});

getContentPane().add(txtCost, new org.netbeans.lib.awtextra.AbsoluteConstraints(382, 463, 134, -1));

btnAdd.setBackground(new java.awt.Color(255, 255, 255));

btnAdd.setText("Add Order");

btnAdd.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnAddActionPerformed(evt);

}

});

getContentPane().add(btnAdd, new org.netbeans.lib.awtextra.AbsoluteConstraints(276, 462, -1, 25));

txtDrinkValue.setEditable(false);

txtDrinkValue.setText("-");

getContentPane().add(txtDrinkValue, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 463, 188, -1));

jLabel1.setFont(new java.awt.Font("Tahoma", 1, 12)); // NOI18N

jLabel1.setForeground(new java.awt.Color(255, 255, 255));

jLabel1.setText("Rs");

getContentPane().add(jLabel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(363, 465, 15, -1));

tblOrder.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Order ID", "Starters", "Main Course", "Desserts", "Drinks", "Price"

}

) {

boolean[] canEdit = new boolean [] {

false, false, false, false, false, false

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

tblOrder.getTableHeader().setReorderingAllowed(false);

tblOrder.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

tblOrderMouseClicked(evt);

}

});

jScrollPane4.setViewportView(tblOrder);

if (tblOrder.getColumnModel().getColumnCount() > 0) {

tblOrder.getColumnModel().getColumn(0).setMinWidth(50);

tblOrder.getColumnModel().getColumn(0).setPreferredWidth(50);

tblOrder.getColumnModel().getColumn(0).setMaxWidth(50);

tblOrder.getColumnModel().getColumn(1).setMinWidth(120);

tblOrder.getColumnModel().getColumn(1).setPreferredWidth(120);

tblOrder.getColumnModel().getColumn(1).setMaxWidth(120);

tblOrder.getColumnModel().getColumn(2).setMinWidth(120);

tblOrder.getColumnModel().getColumn(2).setPreferredWidth(120);

tblOrder.getColumnModel().getColumn(2).setMaxWidth(120);

tblOrder.getColumnModel().getColumn(3).setMinWidth(120);

tblOrder.getColumnModel().getColumn(3).setPreferredWidth(120);

tblOrder.getColumnModel().getColumn(3).setMaxWidth(120);

tblOrder.getColumnModel().getColumn(4).setMinWidth(120);

tblOrder.getColumnModel().getColumn(4).setPreferredWidth(120);

tblOrder.getColumnModel().getColumn(4).setMaxWidth(120);

tblOrder.getColumnModel().getColumn(5).setMinWidth(70);

tblOrder.getColumnModel().getColumn(5).setPreferredWidth(70);

tblOrder.getColumnModel().getColumn(5).setMaxWidth(70);

}

getContentPane().add(jScrollPane4, new org.netbeans.lib.awtextra.AbsoluteConstraints(534, 17, 597, 402));

cbxEmployees.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "Item 1", "Item 2", "Item 3", "Item 4" }));

cbxEmployees.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

cbxEmployeesActionPerformed(evt);

}

});

getContentPane().add(cbxEmployees, new org.netbeans.lib.awtextra.AbsoluteConstraints(383, 369, 133, -1));

getContentPane().add(txtRowCost, new org.netbeans.lib.awtextra.AbsoluteConstraints(530, 430, 82, -1));

txtFinish.setText("Finish Order");

txtFinish.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

txtFinishActionPerformed(evt);

}

});

getContentPane().add(txtFinish, new org.netbeans.lib.awtextra.AbsoluteConstraints(1020, 490, 110, 25));

getContentPane().add(txtTable, new org.netbeans.lib.awtextra.AbsoluteConstraints(383, 400, 133, -1));

tblDrinks.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Drinks", "Price"

}

) {

boolean[] canEdit = new boolean [] {

false, false

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

tblDrinks.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

tblDrinksMouseClicked(evt);

}

});

jScrollPane5.setViewportView(tblDrinks);

if (tblDrinks.getColumnModel().getColumnCount() > 0) {

tblDrinks.getColumnModel().getColumn(1).setMinWidth(70);

tblDrinks.getColumnModel().getColumn(1).setPreferredWidth(70);

tblDrinks.getColumnModel().getColumn(1).setMaxWidth(70);

}

getContentPane().add(jScrollPane5, new org.netbeans.lib.awtextra.AbsoluteConstraints(266, 189, 250, 154));

tblStarters.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Starters", "Price"

}

) {

boolean[] canEdit = new boolean [] {

false, false

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

tblStarters.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

tblStartersMouseClicked(evt);

}

});

jScrollPane1.setViewportView(tblStarters);

if (tblStarters.getColumnModel().getColumnCount() > 0) {

tblStarters.getColumnModel().getColumn(1).setMinWidth(70);

tblStarters.getColumnModel().getColumn(1).setPreferredWidth(70);

tblStarters.getColumnModel().getColumn(1).setMaxWidth(70);

}

getContentPane().add(jScrollPane1, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 17, 250, 154));

tblMains.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Main Courses", "Price"

}

) {

boolean[] canEdit = new boolean [] {

false, false

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

tblMains.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

tblMainsMouseClicked(evt);

}

});

jScrollPane2.setViewportView(tblMains);

if (tblMains.getColumnModel().getColumnCount() > 0) {

tblMains.getColumnModel().getColumn(1).setMinWidth(70);

tblMains.getColumnModel().getColumn(1).setPreferredWidth(70);

tblMains.getColumnModel().getColumn(1).setMaxWidth(70);

}

getContentPane().add(jScrollPane2, new org.netbeans.lib.awtextra.AbsoluteConstraints(266, 17, 250, 154));

tblDesserts.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Desserts", "Price"

}

) {

boolean[] canEdit = new boolean [] {

false, false

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

tblDesserts.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

tblDessertsMouseClicked(evt);

}

});

jScrollPane3.setViewportView(tblDesserts);

if (tblDesserts.getColumnModel().getColumnCount() > 0) {

tblDesserts.getColumnModel().getColumn(1).setMinWidth(70);

tblDesserts.getColumnModel().getColumn(1).setPreferredWidth(70);

tblDesserts.getColumnModel().getColumn(1).setMaxWidth(70);

}

getContentPane().add(jScrollPane3, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 189, 250, 154));

txtStarterPrice.setEditable(false);

txtStarterPrice.setText("0.0");

getContentPane().add(txtStarterPrice, new org.netbeans.lib.awtextra.AbsoluteConstraints(206, 369, 54, -1));

txtMainPrice.setEditable(false);

txtMainPrice.setText("0.0");

getContentPane().add(txtMainPrice, new org.netbeans.lib.awtextra.AbsoluteConstraints(206, 400, 54, -1));

txtDessertPrice.setEditable(false);

txtDessertPrice.setText("0.0");

getContentPane().add(txtDessertPrice, new org.netbeans.lib.awtextra.AbsoluteConstraints(204, 431, 54, -1));

txtDrinkPrice.setEditable(false);

txtDrinkPrice.setText("0.0");

getContentPane().add(txtDrinkPrice, new org.netbeans.lib.awtextra.AbsoluteConstraints(204, 463, 54, -1));

jLabel2.setFont(new java.awt.Font("Microsoft Sans Serif", 0, 14)); // NOI18N

jLabel2.setForeground(new java.awt.Color(255, 255, 255));

jLabel2.setText("Employee");

getContentPane().add(jLabel2, new org.netbeans.lib.awtextra.AbsoluteConstraints(311, 370, -1, -1));

jLabel3.setBackground(new java.awt.Color(255, 255, 255));

jLabel3.setFont(new java.awt.Font("Microsoft Sans Serif", 0, 14)); // NOI18N

jLabel3.setForeground(new java.awt.Color(255, 255, 255));

jLabel3.setText("Table Number");

getContentPane().add(jLabel3, new org.netbeans.lib.awtextra.AbsoluteConstraints(283, 401, -1, -1));

btnDelete.setText("Delete Order");

btnDelete.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnDeleteActionPerformed(evt);

}

});

getContentPane().add(btnDelete, new org.netbeans.lib.awtextra.AbsoluteConstraints(900, 490, 110, 25));

btnBack.setBackground(new java.awt.Color(255, 255, 255));

btnBack.setText("Back");

btnBack.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnBackActionPerformed(evt);

}

});

getContentPane().add(btnBack, new org.netbeans.lib.awtextra.AbsoluteConstraints(10, 490, 90, 25));

jLabel4.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Grey-Background-GREY.jpg"))); // NOI18N

getContentPane().add(jLabel4, new org.netbeans.lib.awtextra.AbsoluteConstraints(-3, -2, 1150, 530));

pack();

}// </editor-fold>

private void btnAddActionPerformed(java.awt.event.ActionEvent evt) {

//This is used to add up all the the costs and get a total

String starter = txtStarterPrice.getText();

runningCost += Double.parseDouble(starter);

rowCost += Double.parseDouble(starter);

txtCost.setText(String.valueOf(runningCost));

txtRowCost.setText(String.valueOf(rowCost));

String main = txtMainPrice.getText();

runningCost += Double.parseDouble(main);

rowCost += Double.parseDouble(main);

txtCost.setText(String.valueOf(runningCost));

txtRowCost.setText(String.valueOf(rowCost));

String d = txtDessertPrice.getText();

runningCost += Double.parseDouble(d);

rowCost += Double.parseDouble(d);

txtCost.setText(String.valueOf(runningCost));

txtRowCost.setText(String.valueOf(rowCost));

String drink = txtDrinkPrice.getText();

runningCost += Double.parseDouble(drink);

rowCost += Double.parseDouble(drink);

txtCost.setText(String.valueOf(runningCost));

txtRowCost.setText(String.valueOf(rowCost));

btnBack.setEnabled(false);

String rowCost1 = Double.toString(rowCost);

//

o.setOrderId(xx);

o.setStarter(txtStarterValue.getText());

o.setMain(txtMainValue.getText());

o.setDessert(txtDessertValue.getText());

o.setDrink(txtDrinkValue.getText());

o.setPrice(rowCost1);

insertOrder(o);

//This creates the table to which the food orders and their price are placed.

model.insertRow(model.getRowCount(), new Object[]{xx, txtStarterValue.getText(), txtMainValue.getText(), txtDessertValue.getText(), txtDrinkValue.getText(), rowCost1});

txtStarterValue.setText("-");

txtMainValue.setText("-");

txtDessertValue.setText("-");

txtDrinkValue.setText("-");

txtRowCost.setText("-");

txtStarterPrice.setText("0.0");

txtMainPrice.setText("0.0");

txtDessertPrice.setText("0.0");

txtDrinkPrice.setText("0.0");

rowCost = 0;

}

private void cbxEmployeesActionPerformed(java.awt.event.ActionEvent evt) {

}

private void txtFinishActionPerformed(java.awt.event.ActionEvent evt) {

//Sends summary of the order to total price table

try{

Integer.parseInt(txtTable.getText());

p.setOrderId(xx);

p.setTableNum(txtTable.getText());

p.setEmployee((String) cbxEmployees.getSelectedItem());

p.setTotalPrice(txtCost.getText());

insertFinalOrder(p);

//opens hompage form

Homepage h = new Homepage();

h.setVisible(true);

this.setVisible(false);

}//error handling if the user inputs a non numeric character

catch(Exception c){

JOptionPane.showMessageDialog(null, "You can only enter numbers into the Table Number textfield");

txtTable.setText(null);

}

}

private void tblDrinksMouseClicked(java.awt.event.MouseEvent evt) {

//adds selected table row to textfields

int row = tblDrinks.getSelectedRow();

String selected = tblDrinks.getValueAt(row, 0).toString();

DefaultTableModel model= (DefaultTableModel)tblDrinks.getModel();

try{

if(selected !=null ){

txtDrinkValue.setText(String.valueOf(model.getValueAt(tblDrinks.getSelectedRow(), 0)));

txtDrinkPrice.setText(String.valueOf(model.getValueAt(tblDrinks.getSelectedRow(), 1)));

}

} catch(NullPointerException a){

JOptionPane.showMessageDialog(null, "Deleted");

}

}

private void tblStartersMouseClicked(java.awt.event.MouseEvent evt) {

//adds selected table row to textfields

int row = tblStarters.getSelectedRow();

String selected = tblStarters.getValueAt(row, 0).toString();

DefaultTableModel model= (DefaultTableModel)tblStarters.getModel();

try{

if(selected !=null ){

txtStarterValue.setText(String.valueOf(model.getValueAt(tblStarters.getSelectedRow(), 0)));

txtStarterPrice.setText(String.valueOf(model.getValueAt(tblStarters.getSelectedRow(), 1)));

}

} catch(NullPointerException a){

JOptionPane.showMessageDialog(null, "Deleted");

}

}

private void tblMainsMouseClicked(java.awt.event.MouseEvent evt) {

//adds selected table row to textfields

int row = tblMains.getSelectedRow();

String selected = tblMains.getValueAt(row, 0).toString();

DefaultTableModel model= (DefaultTableModel)tblMains.getModel();

try{

if(selected !=null ){

txtMainValue.setText(String.valueOf(model.getValueAt(tblMains.getSelectedRow(), 0)));

txtMainPrice.setText(String.valueOf(model.getValueAt(tblMains.getSelectedRow(), 1)));

}

} catch(NullPointerException a){

JOptionPane.showMessageDialog(null, "Deleted");

}

}

private void tblDessertsMouseClicked(java.awt.event.MouseEvent evt) {

//adds selected table row to textfields

int row = tblDesserts.getSelectedRow();

String selected = tblDesserts.getValueAt(row, 0).toString();

DefaultTableModel model= (DefaultTableModel)tblDesserts.getModel();

try{

if(selected !=null ){

txtDessertValue.setText(String.valueOf(model.getValueAt(tblDesserts.getSelectedRow(), 0)));

txtDessertPrice.setText(String.valueOf(model.getValueAt(tblDesserts.getSelectedRow(), 1)));

}

} catch(NullPointerException a){

JOptionPane.showMessageDialog(null, "Deleted");

}

}

private void txtMainValueActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {

//adds selected table row to textfields

int row = tblOrder.getSelectedRow();

DefaultTableModel model= (DefaultTableModel)tblOrder.getModel();

String selected = tblOrder.getValueAt(row, 5).toString();//assigns selected variable to the selected order id for deletion purposes

String Price = tblOrder.getValueAt(row, 5).toString();

if (row >= 0) {

runningCost -= Double.parseDouble(Price);

txtCost.setText(String.valueOf(runningCost));

model.removeRow(row);

try{

a.createConnection();

PreparedStatement ps = a.conn.prepareStatement("delete from ORDER\_Table where Price='"+selected+"' ");

ps.executeUpdate();

JOptionPane.showMessageDialog(null, "Deleted");

btnDelete.setEnabled(false);

}

catch(SQLException sqlExcept){

sqlExcept.printStackTrace();

}

}

}

private void txtCostActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void tblOrderMouseClicked(java.awt.event.MouseEvent evt) {

//enabled delete button

btnDelete.setEnabled(true);

}

private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {

//goes back to the prevous page

Homepage h = new Homepage();

h.setVisible(true);

this.setVisible(false);

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Order.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Order.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Order.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Order.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new Order().setVisible(true);

}

});

}

//populates starter table

public void populateStarter(){

String Starter;

String StarterPrice;

int r =0;

try

{

a.createConnection();

a.stmt = a.conn.createStatement();

ResultSet results = a.stmt.executeQuery("select \* from " + a.StarterTable);

ResultSetMetaData rsmd = results.getMetaData();

while(results.next())

{

Starter = results.getString(1);

StarterPrice = results.getString(2);

Startermodel.insertRow(Startermodel.getRowCount(), new Object[]{Starter, StarterPrice});

r++;

}

results.close();

a.stmt.close();

a.shutdown();

}

catch (SQLException sqlExcept )

{

sqlExcept.printStackTrace();

} catch(IllegalArgumentException a){

JOptionPane.showMessageDialog(this, "You have entered an illegal character", "Error", JOptionPane.ERROR\_MESSAGE);

}

}

//populates main table

public void populateMain(){

String Main;

String MainPrice;

int r =0;

try

{

a.createConnection();

a.stmt = a.conn.createStatement();

ResultSet results = a.stmt.executeQuery("select \* from " + a.MainTable);

ResultSetMetaData rsmd = results.getMetaData();

int numberCols = rsmd.getColumnCount();

for (int i=1; i<=numberCols; i++)

{

//print Column Names

//System.out.print(rsmd.getColumnLabel(i)+"\t");

}

//System.out.println();

while(results.next())

{

Main = results.getString(1);

MainPrice = results.getString(2);

Mainmodel.insertRow(Mainmodel.getRowCount(), new Object[]{Main, MainPrice});

r++;

}

results.close();

a.stmt.close();

a.shutdown();

}

catch (SQLException sqlExcept )

{

sqlExcept.printStackTrace();

} catch(IllegalArgumentException a){

JOptionPane.showMessageDialog(this, "You have entered an illegal character", "Error", JOptionPane.ERROR\_MESSAGE);

}

}

//populate dessert table

public void populateDessert(){

String Dessert;

String DessertPrice;

int r =0;

try

{

a.createConnection();

a.stmt = a.conn.createStatement();

ResultSet results = a.stmt.executeQuery("select \* from " + a.DessertTable);

ResultSetMetaData rsmd = results.getMetaData();

int numberCols = rsmd.getColumnCount();

for (int i=1; i<=numberCols; i++)

{

//print Column Names

//System.out.print(rsmd.getColumnLabel(i)+"\t");

}

//System.out.println();

while(results.next())

{

Dessert = results.getString(1);

DessertPrice = results.getString(2);

Dessertmodel.insertRow(Dessertmodel.getRowCount(), new Object[]{Dessert, DessertPrice});

r++;

}

results.close();

a.stmt.close();

a.shutdown();

}

catch (SQLException sqlExcept )

{

sqlExcept.printStackTrace();

} catch(IllegalArgumentException a){

JOptionPane.showMessageDialog(this, "You have entered an illegal character", "Error", JOptionPane.ERROR\_MESSAGE);

}

}

public void populateDrink(){

String Drink;

String DrinkPrice;

int r =0;

try

{

a.createConnection();

a.stmt = a.conn.createStatement();

ResultSet results = a.stmt.executeQuery("select \* from " + a.DrinkTable);

ResultSetMetaData rsmd = results.getMetaData();

while(results.next())

{

Drink = results.getString(1);

DrinkPrice = results.getString(2);

Drinkmodel.insertRow(Drinkmodel.getRowCount(), new Object[]{Drink, DrinkPrice});

r++;

}

results.close();

a.stmt.close();

a.shutdown();

}

catch (SQLException sqlExcept )

{

sqlExcept.printStackTrace();

} catch(IllegalArgumentException a){

JOptionPane.showMessageDialog(this, "You have entered an illegal character", "Error", JOptionPane.ERROR\_MESSAGE);

}

}

public String extractDigits(String src)

{

StringBuilder builder = new StringBuilder();

for (int i = 0; i < src.length(); i++) {

char c = src.charAt(i);

if (Character.isDigit(c)) {

builder.append(c);

}

}

return builder.toString();

}

public void openStream(){

//try{

//outputOrder = new FileOutputStream("orders.txt");

//objSaveOrder = new ObjectOutputStream(outputOrder);

//}

//catch(IOException e){

//JOptionPane.showMessageDialog(null, "There was a problem locating the Orders.");

//}

}

// Variables declaration - do not modify

private javax.swing.JButton btnAdd;

private javax.swing.JButton btnBack;

private javax.swing.JButton btnDelete;

private javax.swing.JComboBox cbxEmployees;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel jLabel4;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JScrollPane jScrollPane2;

private javax.swing.JScrollPane jScrollPane3;

private javax.swing.JScrollPane jScrollPane4;

private javax.swing.JScrollPane jScrollPane5;

private javax.swing.JTable tblDesserts;

private javax.swing.JTable tblDrinks;

private javax.swing.JTable tblMains;

private javax.swing.JTable tblOrder;

private javax.swing.JTable tblStarters;

private javax.swing.JTextField txtCost;

private javax.swing.JTextField txtDessertPrice;

private javax.swing.JTextField txtDessertValue;

private javax.swing.JTextField txtDrinkPrice;

private javax.swing.JTextField txtDrinkValue;

private javax.swing.JButton txtFinish;

private javax.swing.JTextField txtMainPrice;

private javax.swing.JTextField txtMainValue;

private javax.swing.JTextField txtRowCost;

private javax.swing.JTextField txtStarterPrice;

private javax.swing.JTextField txtStarterValue;

private javax.swing.JTextField txtTable;

// End of variables declaration

}

**Person\_order.java**

class Person\_Order{

//varaible declaration

String OrderID;

String starter;

String main;

String dessert;

String drink;

String price;

public Person\_Order(){///constructor

}

//getters and setters

public String getOrderID(){

return OrderID;

}

public String getStarter(){

return starter;

}

public String getMain(){

return main;

}

public String getDessert(){

return dessert;

}

public String getDrink(){

return drink;

}

public String getPrice(){

return price;

}

public void setOrderId(String iOrderID){

this.OrderID = iOrderID;

}

public void setStarter(String iStarter)

{

this.starter = iStarter;

}

public void setMain(String iMain){

this.main = iMain;

}

public void setDessert(String iDessert){

this.dessert = iDessert;

}

public void setDrink(String iDrink){

this.drink = iDrink;

}

public void setPrice(String iPrice){

this.price = iPrice;

}

}

**Starter.java**

public class Starter {

public Starter(){//constructor

}

//variable declarations

private String Starter, StarterPrice;

//getters and setters

public String getStarter()

{

return Starter;

}

public String getStarterPrice()

{

return StarterPrice;

}

protected void setStarter(String iStarter)

{

this.Starter = iStarter;

}

protected void setStarterPrice(String iStarterPrice)

{

this.StarterPrice = iStarterPrice;

}

}

**TotalPrice.java**

public class TotalPrice {

//variable declarations

String OrderID;

String TableNum;

String Employee;

String TotalPrice;

//constructor

public TotalPrice(){

}

//getters and setters

public String getOrderID(){

return OrderID;

}

public String getTableNum(){

return TableNum;

}

public String getEmployee(){

return Employee;

}

public String getTotalPrice(){

return TotalPrice;

}

public void setOrderId(String iOrderID){

this.OrderID = iOrderID;

}

public void setTableNum(String iTableNum){

this.TableNum = iTableNum;

}

public void setEmployee(String iEmployee){

this.Employee = iEmployee;

}

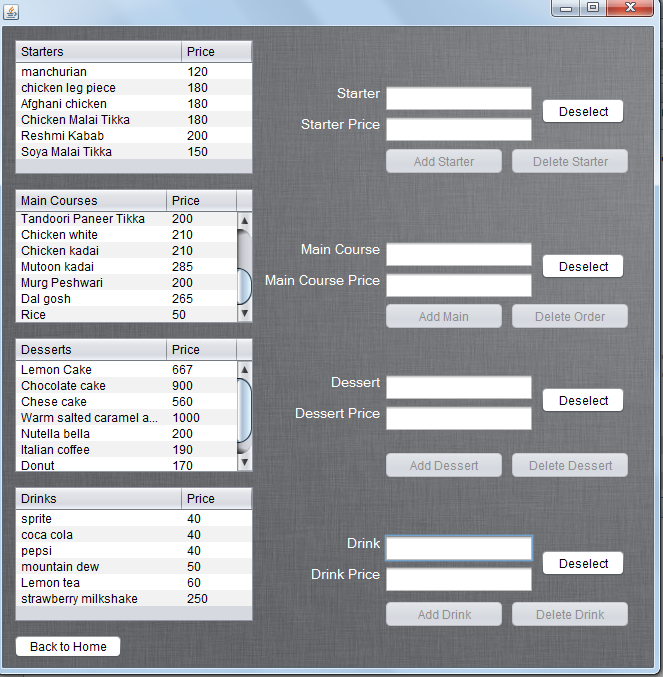
public void setTotalPrice(String iTotalPrice){

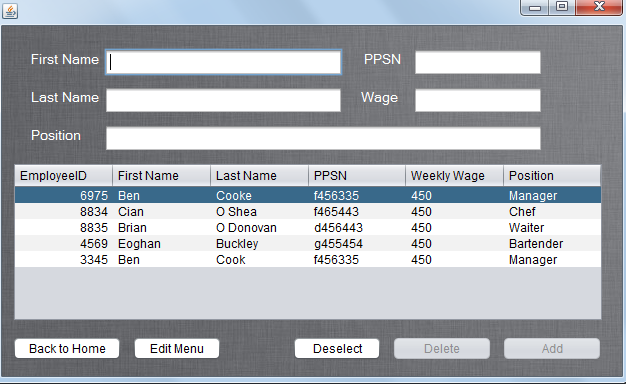
this.TotalPrice = iTotalPrice;

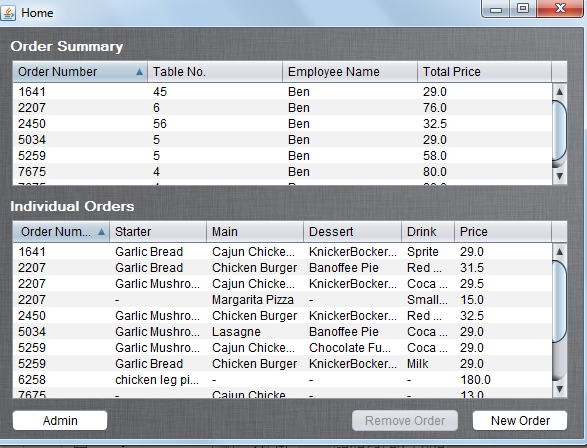
}

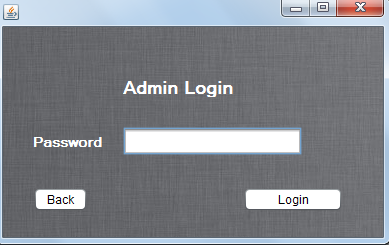
}

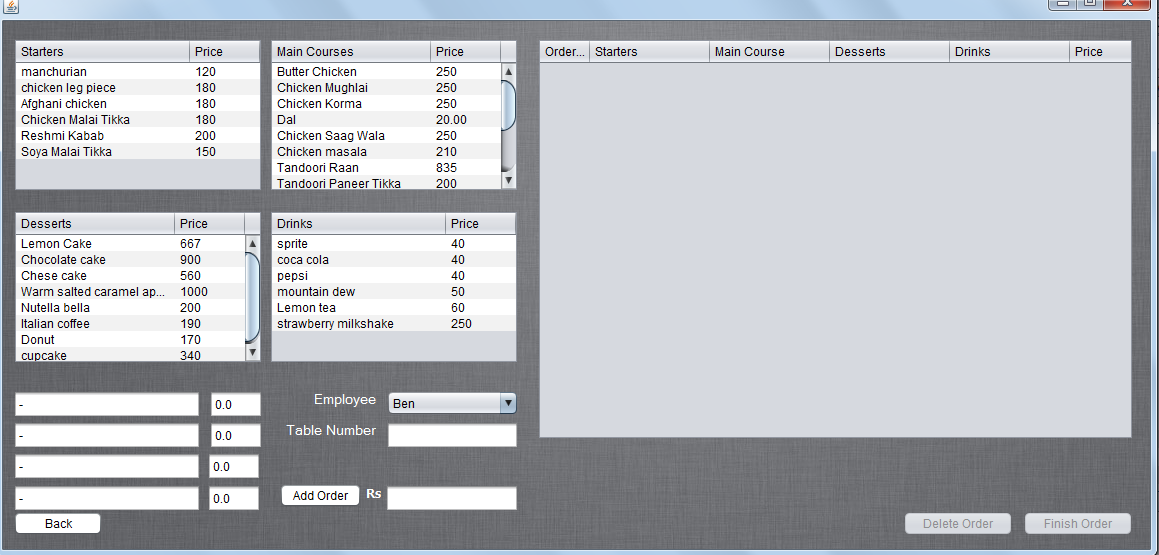
**Output screenshot**

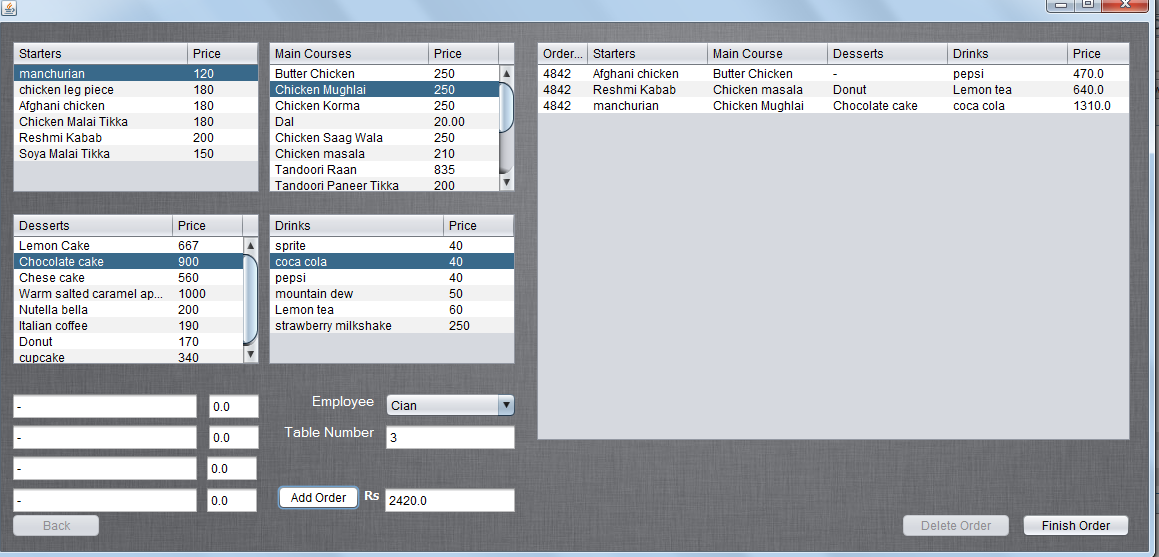
****

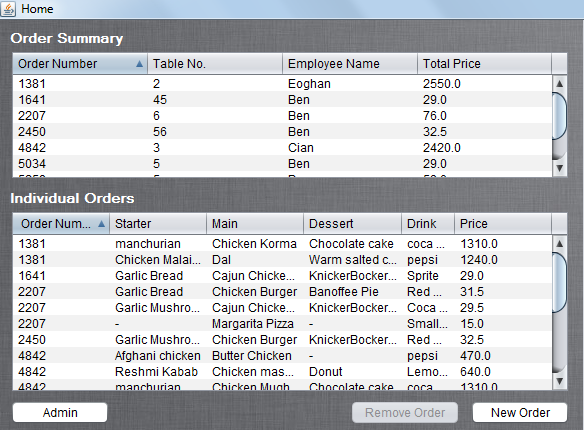
****

****

****

****

****

****

**CONCLUSION**

In this project Restaurent Management System that we are presenting, we have eased the process of Restaurent related stuff. We have used JAVA as our core programming language for the development of our project and implemented it using Netbeans. The system that we have designed is easy to handle and user friendly. Expenses in this project are almost negligible.

**REFERENCES**

1. https:[//www.oracle.com/java/technologies/javase-jre8-downloads.html](http://www.oracle.com/java/technologies/javase-jre8-downloads.html)
2. https:[//www.mysql.com/downloads/](http://www.mysql.com/downloads/)
3. https://[www.javatpoint.com](http://www.javatpoint.com/)