SE 116

FoodPack

Our project is food ordering system. The aim of the project is to take food order from the user and receive it to the restaurant owner.

Utilies

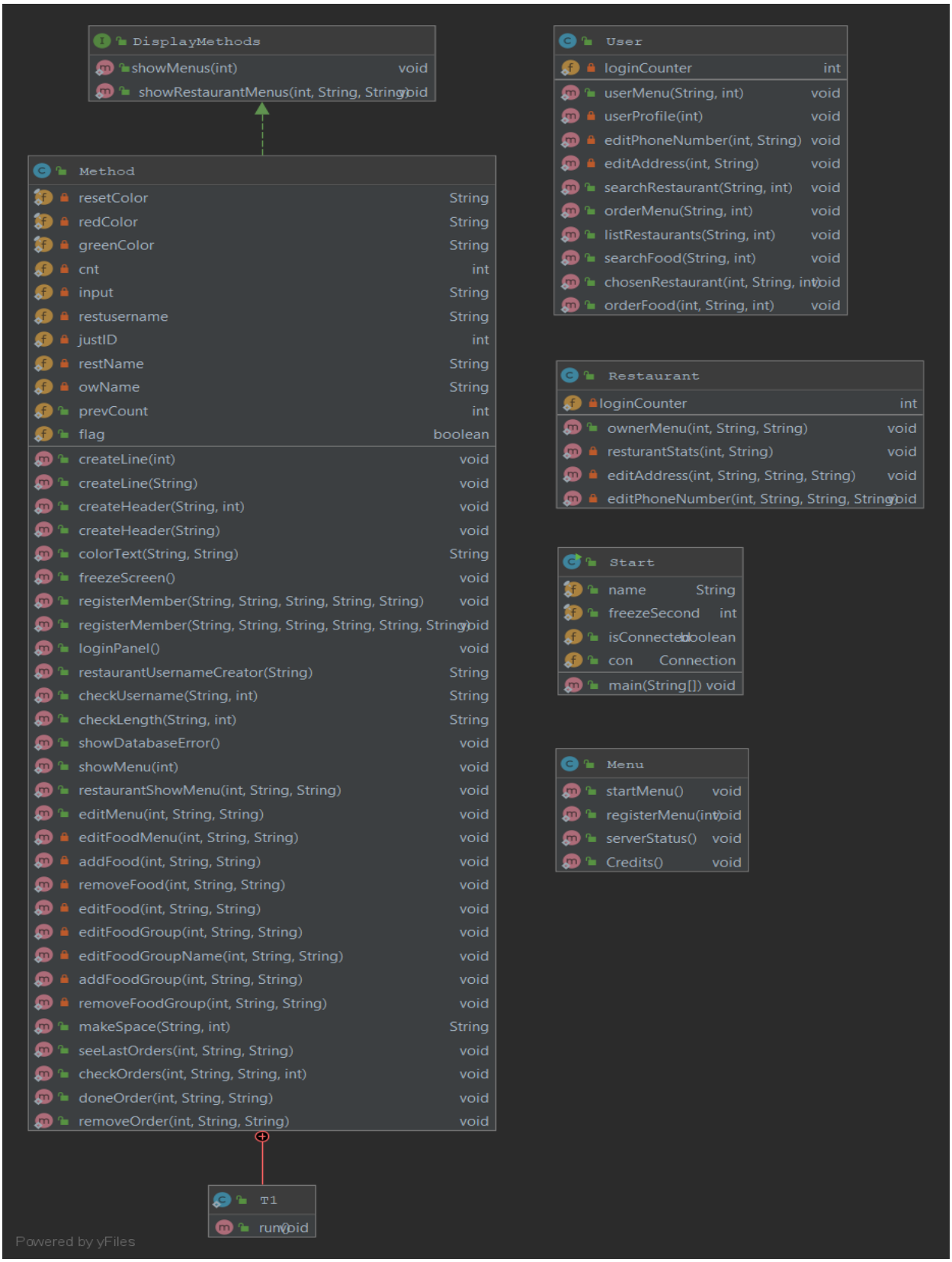
1) Users and Restaurant owners have an account in this program. So they must sign up and log in before using the program.
2) Program can keep the users and restaurant owners data (like number of orders, addresses, phone number est.).
3) Restaurant owners can create a new restaurant and add some new foods on it.
4) Users can search the food by using its name or just search the restaurant name.
5) Program can calculate your order and show how much you pay it and receive your order to the restaurant owner.

We use IntelliJ IDEA Community Edition 2018.2.4 for the development of our project.

This project have 5 classes:

1) Start
2) Menu
3) User
4) Restaurant
5) Method
6) DisplayMethods (interface)

UML Class Diagram



Class Start

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

public class Start {

    public static final String name = "FoodPack";
    public static final int freezeSecond = 0;
    public static boolean isConnected = false;
    public static Connection con;

    public static void main(String[] args){


        try{
            Class.forName("com.mysql.cj.jdbc.Driver");
            con=
DriverManager.getConnection("jdbc:mysql://www.ercom.net:3306/berkdb?useUnic
ode=true&useLegacyDatetimeCode=false&serverTimezone=Turkey&useSSL=false&all
owPublicKeyRetrieval=true","berkmuslu","berk1234");
            isConnected = true;

        }catch(Exception e){
            isConnected = false;
            System.out.println(e);
            }

        Method.createHeader("Welcome To FoodPack");

       Menu.startMenu();

    }
    }
```

Class Menu

```
import java.util.Scanner;
public class Menu {

    public static void startMenu(){

        Scanner input = new Scanner(System.in);

        while(true){
            Method.freezeScreen();
            Method.createHeader("Main Menu", 38);

            System.out.println("\nEnter '1' To Log-In");
            System.out.println("Enter '2' to Register");
            System.out.println("Enter '3' to Check Database Connection");
            System.out.println("Enter '4' to See Credits");
            System.out.println("Enter '-1' to Exit");

            Method.createLine(38);
System.out.print("Your Choice: ");
            int answer = input.nextInt();
```

```
            switch (answer){

                case 1:
                    Method.freezeScreen();

                    Method.loginPanel();
                    break;

                case 2:
                    Method.freezeScreen();
                    Method.createHeader("Register Panel", 41);
                    System.out.println("\nEnter '1' to Register As User");
                    System.out.println("Enter '2' to Register As Restaurant
Owner");
                    System.out.println("Enter '-1' to Return Log-In
Panel");
                    Method.createLine(41);

                    System.out.print("Your Choice: ");
                    int ans2 = input.nextInt();

                    switch (ans2){

                        case 1:
                            registerMenu(1);
                            break;

                        case 2:

                            registerMenu(2);
break;

                        case -1:
                            startMenu();
break;
                        default:
                            continue;

                    }

                case 3:
                    serverStatus();
                    break;

                case 4:
                    Credits();

                case -1:
                    System.out.println("Good Bye!");
                    Method.freezeScreen();

System.exit(1);
                    break;

                    default:
                        continue;


            }
```

```
        }

    }

    public static void registerMenu(int userType){

        Scanner input = new Scanner(System.in);
        Scanner input1 = new Scanner(System.in);

if(userType == 1){

    String fullName;
    String userName;
    String password;
    String address;
    String phone;

    Method.createHeader("Register a New User");
    System.out.println();

    fullName = (Method.checkLength("Full Name",30));


    userName = (Method.checkLength("Username",15));


    password = (Method.checkLength("Password",15));


    while(true){
        System.out.println("Enter Your Phone Number: ");
        phone = input.next();

        if(phone.equals("-1")){
            startMenu();
        }

        if(phone.length() != 10 && !phone.matches("[0-9]+")) {

            System.err.println("Phone Number Must Have 10 Digits!");
            System.err.println("Phone Number Must Only Have Digits!");

        }
        else if(phone.length() != 10){
            System.err.println("Phone Number Must Have 10 Digits!");

        } else if(!phone.matches("[0-9]+")){
            System.err.println("Phone Number Must Only Have Digits!");

        }else{
            break;

        }

    }

    address = (Method.checkLength("Address",255));

    Method.freezeScreen();
```

```
    Method.registerMember(fullName,userName,password,phone,address);
    startMenu();
}

if(userType == 2){

    String fullName;
    String restaurantName;
    String password;
    String address;
    String phone;

    Method.createHeader("Register a New Restaurant Owner");

    System.out.println();
    fullName = (Method.checkLength("Full Name",30));
    restaurantName = (Method.checkLength("Restaurant Name",50));
    password = (Method.checkLength("Password",15));

    while(true){
    System.out.println("Enter Your Phone Number: ");
    phone = input.next();

    if(phone.equals("-1")){
        startMenu();
    }

    if(phone.length() != 10 && !phone.matches("[0-9]+")) {

        System.err.println("Phone Number Must Have 10 Digits!");
        System.err.println("Phone Number Must Only Have Digits!");

    }
    else if(phone.length() != 10){
        System.err.println("Phone Number Must Have 10 Digits!");

    } else if(!phone.matches("[0-9]+")){
        System.err.println("Phone Number Must Only Have Digits!");
    }else{
        break;

    }
}
     address = (Method.checkLength("Address",255));

    String restUsername1 = "restaurant_" +
restaurantName.replaceAll("\\s","");
    String restn1= restUsername1.replaceAll("ş","s");
    String restn2 = restn1.replaceAll("ç","c");
    String restn3 = restn2.replaceAll("ı","i");
    String restn4 = restn3.replaceAll("ü","u");
    String restUsername = restn4.replaceAll("ö","o");
    String restaurantUserName = restUsername.toLowerCase();

    Method.freezeScreen();

Method.registerMember(fullName,restaurantName,restaurantUserName,password,p
hone,address);
    startMenu();
```

```
}

    }

    public static void serverStatus(){

        Method.createHeader("DATABASE STATUS",30);

        if(Start.isConnected){

            System.out.println("\n|Database Connection : " +
Method.colorText("OK!","green") + "    |");

        }else{

            System.out.println("\n|Database Connection : " +
Method.colorText("ERROR!","red") + "|");

        }

        Method.createLine(30);
        Method.freezeScreen();
        startMenu();

    }

    public static void Credits(){
        Method.createHeader("Credits",35);
        System.out.println();
        System.out.println("Project Name: " + Start.name);
        System.out.println("-----------------------------------");
        System.out.println("Lecture: SE-116");
        System.out.println("-----------------------------------");
        System.out.println("Creators: \nBerk Muslu \nÖmer Yakup
Cankurtaran");
        System.out.println("-----------------------------------");
        Method.freezeScreen();
        startMenu();

    }

}
```

Class User

```
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.Calendar;
import java.util.Scanner;

public class User {

    private static int loginCounter = 0;

    public static void userMenu(String userName, int ID) {

if(loginCounter == 0) {
    Method.createHeader("Welcome, " + userName);
loginCounter++;
}
```

```
        Scanner input = new Scanner(System.in);

        while (true) {
            Method.freezeScreen();
            Method.createHeader("User Menu", 32);
            System.out.println("\nEnter '1' To Order Food");
            System.out.println("Enter '2' to See Profile");
            System.out.println("Enter '-1' to Return Log-In Panel");
            Method.createLine(32);
            System.out.print("Your Choice: ");
            int answer = input.nextInt();

            switch (answer) {

                case 1:
                    orderMenu(userName, ID);

                    break;

                case 2:
                    userProfile(ID);
                    break;

                case -1:
                    loginCounter = 0;
                    Menu.startMenu();
                    break;

                default:
                    continue;
            }
        }
    }

    private static void userProfile(int userID) {

        Scanner input = new Scanner(System.in);

        PreparedStatement ps;
        ResultSet rs;

        String fullname, username, password, phoneNumber, address;
        int totalOrder;
        double totalMoney;


        String query = "SELECT * FROM users WHERE `ID` =?";

        try {

            ps = Start.con.prepareStatement(query);

            ps.setString(1, Integer.toString(userID));

            rs = ps.executeQuery();


            if (rs.next()) {

                int ID = rs.getInt("ID");
```

```
                fullname = rs.getString("fullname");
                username = rs.getString("username");
                password = rs.getString("password");
                phoneNumber = rs.getString("phone");
                address = rs.getString("address");
                totalOrder = rs.getInt("totalorder");
                totalMoney = rs.getDouble("totalspending");


                while (true) {
                    Method.freezeScreen();
                    System.out.println();
                    Method.createHeader("User Profile");
                    System.out.println();
                    System.out.println("Full Name: " + fullname);
                    System.out.println("Username: " + username);
                    System.out.println("Password: " + password);
                    System.out.println("Phone Number: " + phoneNumber);
                    System.out.println("Address: " + address);
                    System.out.println("Total Order: " + totalOrder);
                    System.out.println("Money Spent So Far: " + totalMoney
+ " TL");
                    Method.freezeScreen();

                    Method.createLine(30);
                    System.out.println("Enter '1' to Edit Phone Number");
                    System.out.println("Enter '2' to Edit Address");
                    System.out.println("Enter '-1' to Return");
                    Method.createLine(30);
                    System.out.print("Your Choice: ");

                    int answer = input.nextInt();

                    switch (answer) {

                        case 1:
                            editPhoneNumber(userID, phoneNumber);
                            break;
                        case 2:
                            editAddress(userID, address);
                            break;
                        case -1:
                            userMenu(username, userID);
                        default:
                            continue;
                    }

                }


            }


        } catch (Exception e) {
            System.out.println(e);
        }


    }
```

```
    private static void editPhoneNumber(int userID, String oldPhoneNumber)
{


        Scanner input = new Scanner(System.in);

        PreparedStatement ps;
        String newPhoneNumber;


        while (true) {

            System.out.println("Please Enter New Phone Number: ");
            newPhoneNumber = input.next();

            if (newPhoneNumber.equals("-1")) {
                userProfile(userID);
            }


            if (oldPhoneNumber.equals(newPhoneNumber)) {
                System.err.println("The New Phone Number Can't Be The Same
With Old Phone Number!");
            } else if (newPhoneNumber.length() != 10
&& !newPhoneNumber.matches("[0-9]+")) {

                System.err.println("Phone Number Must Have 10 Digits!");
                System.err.println("Phone Number Must Only Have Digits!");

            } else if (newPhoneNumber.length() != 10) {
                System.err.println("Phone Number Must Have 10 Digits!");

            } else if (!newPhoneNumber.matches("[0-9]+")) {
                System.err.println("Phone Number Must Only Have Digits!");

            } else {
                break;

            }

        }

        try {

            String query = "update users set phone = ? where ID = ?";
            ps = Start.con.prepareStatement(query);
            ps.setString(1, newPhoneNumber);
            ps.setString(2, Integer.toString(userID));
            ps.executeUpdate();


            System.out.println(Method.colorText("Phone Number Changed To "
+ newPhoneNumber, "green"));
            Method.freezeScreen();

            userProfile(userID);


        } catch (Exception e) {
            Method.showDatabaseError();
        }
```

```
    }

    private static void editAddress(int userID, String oldAddress) {
        PreparedStatement ps;
        Scanner input1 = new Scanner(System.in);
        String newAddress;
        while (true) {
            System.out.println("Please Enter New Address: ");
            newAddress = input1.nextLine();

            if (newAddress.equals(oldAddress)) {
                System.err.println("New Address Can't Be The Same With Old
Address");
            } else if (newAddress.equals("-1")) {
                userProfile(userID);
            } else {
                break;

            }
        }


        try {

            String query = "update users set address = ? where ID = ?";
            ps = Start.con.prepareStatement(query);
            ps.setString(1, newAddress);
            ps.setString(2, Integer.toString(userID));
            ps.executeUpdate();

            System.out.println(Method.colorText("Address Changed To " +
newAddress, "green"));
            Method.freezeScreen();

            userProfile(userID);


        } catch (Exception e) {
            Method.showDatabaseError();
        }

    }

    public static void searchRestaurant(String username, int userID) {
        Scanner inp = new Scanner(System.in);

        PreparedStatement ps;
        ResultSet rs;
        Method.createHeader("Search Restaurant");
        System.out.println();
        System.out.println("Enter A Property Of The Restaurant");
        String property = inp.nextLine();
property = property.toLowerCase();

        String query = "SELECT * FROM restaurants WHERE
LOWER( restaurants.restaurantname ) LIKE ? OR restaurantID =?";
        String query1 = "SELECT COUNT(*) AS Count FROM restaurants WHERE
LOWER( restaurants.restaurantname ) LIKE ? OR restaurantID =?";

        try {
```

```
            ps = Start.con.prepareStatement(query1);
            ps.setString(1, "%"  + property + "%");
            ps.setString(2, property);
            rs = ps.executeQuery();
            rs.next();

            int count = rs.getInt("Count");

            if(count == 0 || count < 0){
                {
                    Method.createHeader("There Is No Match!");
                    System.out.println();
                    userMenu(username,userID);

                }
            }else if (count == 1) {
                Method.createHeader("There Is 1 Restaurant");
            } else {
                Method.createHeader("There Are " + count + " Restaurans");
            }

            ps = Start.con.prepareStatement(query);
            ps.setString(1, "%"  + property + "%");
            ps.setString(2, property);

            rs = ps.executeQuery();


            System.out.println();
            System.out.printf("%-13s %s %-23s %s %-25s %s %-23s", "ID",
"\t", " Restaurant Name", "\t", "Restaurant Number", "\t\t\t", " Restaurant
Address");
            System.out.println();
            System.out.printf("%-13s %s %-23s %s %-25s %s %-23s", "---",
"\t ", "---------------", "\t", "-----------------", "\t\t\t ", "----------
-----------------");
            System.out.println();

            while (rs.next()) {

                System.out.printf("#%-13s %s %-23s %s %-25s %s %-23s",
rs.getInt("restaurantID"), "\t ", rs.getString("restaurantname"), "\t",
rs.getString("phone"), "\t\t\t ", rs.getString("address"));
                System.out.println();
            }


        } catch (Exception e) {
            System.out.println(e);
        }

}

    public static void orderMenu(String userName,int ID) {

        Scanner input = new Scanner(System.in);

        while (true) {

            Method.freezeScreen();
```

```
            Method.createHeader("Order Menu", 30);
            System.out.println("\nEnter '1' to List Restaurants");
            System.out.println("Enter '2' to Search Restaurant");
            System.out.println("Enter '3' to Search Food");
            System.out.println("Enter '-1' to Return User Menu");
            Method.createLine(30);
            System.out.print("Your Choice: ");
            int answer = input.nextInt();


            switch (answer) {

                case 1:
                    listRestaurants(userName,ID);

                    break;


                case 2:
                    searchRestaurant(userName,ID);

                    break;
                case 3:
                    searchFood(userName,ID);

                    break;
                case -1:
                    userMenu(userName,ID);
                    break;

                default:
                    continue;


            }


        }


    }

    public static void listRestaurants(String username,int ID) {


        PreparedStatement ps;
        ResultSet rs;
        int count;

        String query = "SELECT COUNT(restaurantID) AS Count FROM
restaurants";
        String query1 = "SELECT * FROM restaurants";

        try {
            ps = Start.con.prepareStatement(query);
            rs = ps.executeQuery();
            rs.next();
            count = rs.getInt("Count");
            System.out.println();
```

```
            if(count == 0 || count < 0){
                {
                    Method.createHeader("There Is No Match!");
                    System.out.println();
userMenu(username,ID);
                }
            }else if (count == 1) {
                Method.createHeader("There Is 1 Restaurant");
            } else {
                Method.createHeader("There Are " + count + " Restaurans");
            }


            System.out.println();
            System.out.printf("%-13s %s %-23s %s %-25s %s %-23s", "ID",
"\t", " Restaurant Name", "\t", "Restaurant Number", "\t\t\t", " Restaurant
Address");
            System.out.println();
            System.out.printf("%-13s %s %-23s %s %-25s %s %-23s", "---",
"\t ", "---------------", "\t", "-----------------", "\t\t\t ", "----------
-----------------");
            System.out.println();


        }catch (Exception e){


            System.out.println(e);


        }
;
        try {

            ps = Start.con.prepareStatement(query1);
            rs = ps.executeQuery();

            while (rs.next()){

                System.out.printf("#%-13s %s %-23s %s %-25s %s %-23s",
rs.getInt("restaurantID"), "\t ", rs.getString("restaurantname"), "\t",
rs.getString("phone"), "\t\t\t ", rs.getString("address"));
                System.out.println();
            }


        }catch (Exception e){
            System.out.println(e);
        }


while (true){
    Scanner inp = new Scanner(System.in);
    System.out.println();
    Method.createLine(30);
    System.out.println("Enter '1' To Choose Restaurant");
    System.out.println("Enter '-1' To Return");
    Method.createLine(30);
    System.out.print("Your Choice: ");

    int ans = inp.nextInt();
```

```
    switch (ans) {
        case 1:
try {
    ps = Start.con.prepareStatement(query);
    rs = ps.executeQuery();
    rs.next();
    count = rs.getInt("Count");

    System.out.print("Enter Restaurant ID: #");
    int id = inp.nextInt();

    if (id == -1) {

    }else if(id > count){

        System.err.println("ID Is Out Of Range!");
    }else {

        chosenRestaurant(id,username,ID);
    }

    break;

}catch (Exception e){
    System.out.println(e);
}

            break;
        case -1:

            orderMenu(username,ID);
            break;
            default:


    }

}


    }

    public static void searchFood(String username, int userID) {


        int tempID;
        Scanner input = new Scanner(System.in);
        System.out.println("Enter The Property Of The Food:");
        String property = input.nextLine();
        PreparedStatement ps;
        ResultSet rs;
        property = property.toLowerCase();

        String realquery = "SELECT restaurants.restaurantname AS
restaurantname , menus.* FROM restaurants INNER JOIN menus ON
restaurants.restaurantID = menus.restaurantID AND LOWER(menus.foodname)
LIKE ?";
        try {

            ps = Start.con.prepareStatement(realquery);
```

```
            ps.setString(1, "%"  + property + "%");
            rs = ps.executeQuery();


/*
            if(rs.next()){


            }

            */

tempID = -1;

            while (rs.next()) {

               if(tempID != rs.getInt("restaurantID")){
                    System.out.println();
                    System.out.printf("%-13s",
rs.getString("restaurantname") + "(#" + rs.getInt("restaurantID") + ")");
                    System.out.println();
                    Method.createLine(rs.getString("restaurantname") + "(#"
+ rs.getInt("restaurantID") + ")");

                   System.out.printf("#%-4d %-50s %s %.2f%s",
rs.getInt("foodID"), rs.getString("foodname") + "(" +
rs.getString("foodingredients") + ")" , "\t",
rs.getDouble("foodprice"),"TL");
                   System.out.println();


               }else{


                    System.out.printf("#%-4d %-50s %s %.2f%s",
rs.getInt("foodID"), rs.getString("foodname") + "(" +
rs.getString("foodingredients") + ")" , "\t",
rs.getDouble("foodprice"),"TL");
                    System.out.println();
                }

                tempID = rs.getInt("restaurantID");



            }


        } catch (Exception e) {
            System.out.println(e);
        }



        while (true){
            Scanner inp = new Scanner(System.in);
            System.out.println();
            Method.createLine(30);
            System.out.println("Enter '1' To Choose Restaurant");
            System.out.println("Enter '-1' To Return");
            Method.createLine(30);
```

```
            System.out.print("Your Choice: ");

            int ans = inp.nextInt();

            switch (ans) {
                case 1:
                    try {
String countQuery = "SELECT COUNT(restaurantID) AS restCount FROM
restaurants";
ps = Start.con.prepareStatement(countQuery);
rs = ps.executeQuery();
rs.next();

int count = rs.getInt("restCount");


                        System.out.print("Enter Restaurant ID: #");
                        int id = inp.nextInt();

                        if (id == -1) {

orderMenu(username,userID);
                        }else if(id > count) {
                            System.out.println("Check Restaurant ID!");
                        }
                            else
                         {

                            chosenRestaurant(id,username,userID);
                        }

                        break;

                    }catch (Exception e){
                        System.out.println(e);
                    }

                    break;
                case -1:

                    orderMenu(username,userID);
                    break;
                default:


            }

        }


    }

    public static void chosenRestaurant(int ID,String username,int userID){

        String restname;
        PreparedStatement ps;
        ResultSet rs;
        String query = "SELECT * FROM restaurants WHERE `restaurantID` =?";

        Scanner inp = new Scanner(System.in);
        try {
```

```
            ps = Start.con.prepareStatement(query);
            ps.setInt(1, ID);
            rs = ps.executeQuery();
rs.next();

                restname = rs.getString("restaurantname");

            Method.createLine("Chosen Restaurant:   " +restname);
            System.out.println("|Chosen Restaurant: " +restname + "|");



            while(true){
                Method.createLine("Chosen Restaurant:   " +restname);
                System.out.println("Enter '1' To See Menu");
                System.out.println("Enter '2' To Order");
                System.out.println("Enter '-1' To Return");
                Method.createLine("Chosen Restaurant:   " +restname);
                System.out.print("Your Choice: ");
                int ans = inp.nextInt();

                switch (ans){

                    case 1:
                        Method.showMenu(ID);

                        break;
                    case 2:
                        orderFood(userID,username,ID);

                        break;
                    case -1:
                        userMenu(username,userID);
break;
                    default:


                }

            }




        }catch (Exception e) {
//            System.out.println(e);
        }



    }

    public static void orderFood(int id,String username,int restID){
        java.sql.Date date = new
java.sql.Date(Calendar.getInstance().getTime().getTime());

        PreparedStatement ps;
        ResultSet rs;
        Scanner inp = new Scanner(System.in);
```

```
        System.out.print("Enter Food ID: #");
int foodid = inp.nextInt();
if (foodid == -1){
    userMenu(username,id);
}

String query = "SELECT * FROM menus WHERE foodID =?";
String query1 = "SELECT * FROM users WHERE username =?";
try {
    ps = Start.con.prepareStatement(query);
    ps.setInt(1,foodid);
    rs = ps.executeQuery();
    rs.next();
    String foodname = rs.getString("foodname");


    Method.createHeader("Selected Food: " + foodname);

    System.out.print("\nEnter Amount:");
    int amount = inp.nextInt();

    if(amount < 0){
        orderFood(id,username,restID);
    }else{

        double price;
        String query2 = "INSERT INTO orders
(restaurantID,userID,foodID,amount,price,date) VALUES (?,?,?,?,?,?)";


        price = rs.getDouble("foodprice");

        price *= amount;

        ps = Start.con.prepareStatement(query1);
        ps.setString(1,username);
        rs = ps.executeQuery();
        rs.next();

        int userID = rs.getInt("ID");
        String fullname = rs.getString("fullname");
        String address = rs.getString("address");
        String phone = rs.getString("phone");
        int totalorder = rs.getInt("totalorder");
        double totalspending = rs.getDouble("totalspending");

        Method.createLine(59);
        System.out.println("|                Order Confirmation Screen
|");
        Method.createLine(59);
        System.out.println("|Full Name: " + fullname +
Method.makeSpace("|Full Name: " + fullname,59) + "|");
        System.out.println("|Address: " + address +
Method.makeSpace("|Address: " + address,59) + "|");
        System.out.println("|Phone Number: " + phone +
Method.makeSpace("|Phone Number: " + phone,59) + "|");
        Method.createLine(59);
        System.out.println("|Selected Food: " + foodname + "(" + amount +
")" + Method.makeSpace("|Selected Food: " + foodname + "(" + amount +
")",59) + "|");
        System.out.println("|Price: "  + price + " TL" +
```

```
Method.makeSpace("|Price: "  + price + " TL",59) + "|");
        Method.createLine(59);
        System.out.println("|Enter 1 To Confirm" + Method.makeSpace("Enter
1 To Confirm",58) + "|");
        System.out.println("|Enter -1 To Cancel" + Method.makeSpace("Enter
-1 To Cancel",58) + "|");
        Method.createLine(59);
        System.out.print("Your Choice: ");
int ans = inp.nextInt();

        if(ans == 1){
            totalorder++;
            totalspending+= price;

            ps = Start.con.prepareStatement(query2);
            ps.setInt(1,restID);
            ps.setInt(2,userID);
            ps.setInt(3,foodid);
            ps.setInt(4,amount);
            ps.setDouble(5,price);
            ps.setDate(6,date);
            ps.executeUpdate();

            String query4 = "SELECT * FROM orders WHERE OrderID IN (SELECT
MAX(OrderID) FROM orders WHERE userID = ?);";
            ps = Start.con.prepareStatement(query4);
            ps.setInt(1,userID);
            rs = ps.executeQuery();
            rs.next();
int orderID = rs.getInt("orderID");
            String query3 = "UPDATE users SET totalorder = ? ,
totalspending = ? WHERE ID = ?;";

ps = Start.con.prepareStatement(query3);
ps.setInt(1,totalorder);
ps.setDouble(2,totalspending);
ps.setInt(3,userID);
ps.executeUpdate();

Method.freezeScreen();
            System.out.println(Method.colorText("Order#" + orderID + " Is
Ordered Succesfully!","green"));

            userMenu(username,userID);


       }



    }
}catch (Exception e){
    System.out.println(e);
}}}
```

Class Restaurant

```
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.Scanner;

public class Restaurant {

    private static int loginCounter = 0;

    public static void ownerMenu(int ID,String restaurantName,String
restaurantOwner){

        if(loginCounter == 0) {
            Method.createHeader("Welcome, " + restaurantOwner);
            System.out.println();
            Method.createHeader("Restaurant Name: " + restaurantName);

            loginCounter++;
        }


        Scanner input = new Scanner(System.in);

        while (true) {
            Method.freezeScreen();

            Method.createHeader("Owner Menu", 33);
            System.out.println("\nEnter '1' To See Orders");
            System.out.println("Enter '2' To See Menu");
            System.out.println("Enter '3' to See Last 5 Orders");
            System.out.println("Enter '4' to See Restaurant Stats");
            System.out.println("Enter '-1' to Return Log-In Panel");

            Method.createLine(33);
            System.out.print("Your Choice: ");
            int answer = input.nextInt();

            switch (answer) {

                case 1:


Method.checkOrders(ID,restaurantName,restaurantOwner,0);
                    break;

                case 2:

Method.restaurantShowMenu(ID,restaurantName,restaurantOwner);
                    break;

                case 3:
Method.seeLastOrders(ID,restaurantName,restaurantOwner);
                    break;
                case 4:
                    resturantStats(ID,restaurantName);
                    break;
                case -1:
                    loginCounter = 0;
                    Menu.startMenu();
```

```
                    break;

            }

        }



    }

    private static void resturantStats(int restID,String restName) {
double price = 0;
        PreparedStatement ps;
        ResultSet rs;
String query1 = "SELECT COUNT(*) FROM orders WHERE restaurantID = ?";
        String query = "SELECT * FROM restaurants WHERE restaurantID = ?";
        String query2 = "SELECT price FROM orders WHERE restaurantID = ?";
        try {
            ps = Start.con.prepareStatement(query);
            ps.setInt(1, restID);
            rs = ps.executeQuery();
            rs.next();
           String restowner = rs.getString("restaurantowner");
String phone = rs.getString("phone");
            String address = rs.getString("address");

            ps = Start.con.prepareStatement(query1);
            ps.setInt(1,restID);
            rs = ps.executeQuery();
            rs.next();
            int count = rs.getInt(1);

            ps = Start.con.prepareStatement(query2);
            ps.setInt(1, restID);
rs = ps.executeQuery();
            while (rs.next()){
                price += rs.getDouble("price");
            }

            Method.freezeScreen();
            System.out.println();
            Method.createHeader("Restaurant Stats");
            System.out.println();
            System.out.println("Restaurant Name: " + restName);
            System.out.println("Owner: " + restowner);
            System.out.println("Phone Number: " + phone);
            System.out.println("Address: " + address);
           // System.out.println("Today's Orders: 1");
           // System.out.println("Today's Income: 20.00 TL");
            System.out.println("All Orders So Far: " + count);
            System.out.println("All Income So Far: " + price + " TL");
            Method.freezeScreen();

            Method.createLine(30);
            System.out.println("Enter '1' to Edit Phone Number");
            System.out.println("Enter '2' to Edit Address");
            System.out.println("Enter '-1' to Return");
            Method.createLine(30);
            System.out.print("Your Choice: ");

            Scanner input = new Scanner(System.in);
```

```
            int answer = input.nextInt();

            switch (answer) {

                case 1:
                    editPhoneNumber(restID, restName,restowner,phone);
                    break;
                case 2:
                    editAddress(restID, restName,restowner,address);
                    break;
                case -1:
ownerMenu(restID,restName,restowner);
                    break;
                default:
resturantStats(restID,restName);
            }

        }catch (Exception e){
            System.out.println(e);
        }



    }

    private static void editAddress(int restID, String restName,String
restOwner ,String oldAddress) {
        PreparedStatement ps;
        Scanner input1 = new Scanner(System.in);
        String newAddress;
        while (true) {
            System.out.println("Please Enter New Address: ");
            newAddress = input1.nextLine();

            if (newAddress.equals(oldAddress)) {
                System.err.println("New Address Can't Be The Same With Old
Address");
            } else if (newAddress.equals("-1")) {
                ownerMenu(restID,restName,restOwner);
            } else {
                break;

            }
        }


        try {

            String query = "update restaurants set address = ? where
restaurantID = ?";
            ps = Start.con.prepareStatement(query);
            ps.setString(1, newAddress);
            ps.setString(2, Integer.toString(restID));
            ps.executeUpdate();

            System.out.println(Method.colorText("Address Changed To " +
newAddress, "green"));
            Method.freezeScreen();

            resturantStats(restID,restName);
```

```
        } catch (Exception e) {
            Method.showDatabaseError();
        }

    }

    private static void editPhoneNumber(int restID, String restName,String
restOwner ,String oldPhoneNumber) {


        Scanner input = new Scanner(System.in);

        PreparedStatement ps;
        String newPhoneNumber;


        while (true) {

            System.out.println("Please Enter New Phone Number: ");
            newPhoneNumber = input.next();

            if (newPhoneNumber.equals("-1")) {
                ownerMenu(restID,restName,restOwner);
                            }


            if (oldPhoneNumber.equals(newPhoneNumber)) {
                System.err.println("The New Phone Number Can't Be The Same
With Old Phone Number!");
            } else if (newPhoneNumber.length() != 10
&& !newPhoneNumber.matches("[0-9]+")) {

                System.err.println("Phone Number Must Have 10 Digits!");
                System.err.println("Phone Number Must Only Have Digits!");

            } else if (newPhoneNumber.length() != 10) {
                System.err.println("Phone Number Must Have 10 Digits!");

            } else if (!newPhoneNumber.matches("[0-9]+")) {
                System.err.println("Phone Number Must Only Have Digits!");

            } else {
                break;

            }

        }

        try {

            String query = "update restaurants set phone = ? where
restaurantID = ?";
            ps = Start.con.prepareStatement(query);
            ps.setString(1, newPhoneNumber);
            ps.setString(2, Integer.toString(restID));
            ps.executeUpdate();


            System.out.println(Method.colorText("Phone Number Changed To "
```

```
+ newPhoneNumber, "green"));
            Method.freezeScreen();

            resturantStats(restID,restName);


        } catch (Exception e) {
            Method.showDatabaseError();
        }

    }


}
```

Class Method

```
import java.sql.*;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.Scanner;
import java.util.concurrent.Executors;
import java.util.concurrent.ScheduledExecutorService;
import java.util.concurrent.TimeUnit;

public class Method {

    private static final String resetColor = "\u001B[0m";
    private static final String redColor = "\u001B[31m";
    private static final String greenColor = "\u001B[32m";

    public static void createLine(int Amount) {


        for (int i = 0; i < Amount; i++) {

            System.out.print("-");

        }
        System.out.println();


    }

    public static void createLine(String Text) {


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

            System.out.print("-");

        }
        System.out.println();


    }

    public static void createHeader(String Text, int Bottom) {
        System.out.println();
        for (int i = 0; i < Text.length() + 2; i++) {
            System.out.print("-");
```

```
        }

        System.out.println("\n|" + Text + "|");

        for (int i = 0; i < Bottom; i++) {
            System.out.print("-");
        }


    }

    public static void createHeader(String Text) {

        for (int i = 0; i < Text.length() + 2; i++) {
            System.out.print("-");
        }

        System.out.println("\n|" + Text + "|");

        for (int i = 0; i < Text.length() + 2; i++) {
            System.out.print("-");
        }


    }

    public static String colorText(String Text, String Color) {

        if (Color.toLowerCase().equals("red")) {
            Color = redColor;

        }

        if (Color.toLowerCase().equals("green")) {
            Color = greenColor;

        }

        String newText = (Color + Text + resetColor);
        return newText;

    }

    public static void freezeScreen() {

        try {

            Thread.sleep(Start.freezeSecond * 1000);
        } catch (InterruptedException e) {
            System.out.println(e);
        }

    }

    public static void registerMember(String fullname, String username,
String pass, String phone, String address) {


        try {

            java.sql.Date date = new
```

```
java.sql.Date(Calendar.getInstance().getTime().getTime());

            String query = " insert into users (fullname, username,
password, phone, address, memberdate)"
                    + " values (?, ?, ?, ?, ?, ?)";

            PreparedStatement preparedStmt =
Start.con.prepareStatement(query);
            preparedStmt.setString(1, fullname);
            preparedStmt.setString(2, username);
            preparedStmt.setString(3, pass);
            preparedStmt.setString(4, phone);
            preparedStmt.setString(5, address);
            preparedStmt.setDate(6, date);
            preparedStmt.execute();

            System.out.print(Method.colorText("User " + username + "
Created Successfully!", "green"));
            System.out.println();


        } catch (Exception e) {
            createLine(25);
            showDatabaseError();
            createLine(25);
        }

    }

    public static void registerMember(String fullname, String
restaurantname, String restaurantusername, String pass, String phone,
String address) {

        try {

            java.sql.Date date = new
java.sql.Date(Calendar.getInstance().getTime().getTime());

            String query = " insert into restaurants (restaurantowner,
restaurantname, restaurantusername, restaurantpassword, phone, address,
date)"
                    + " values (?, ?, ?, ?, ?, ?, ?)";

            PreparedStatement preparedStmt =
Start.con.prepareStatement(query);
            preparedStmt.setString(1, fullname);
            preparedStmt.setString(2, restaurantname);
            preparedStmt.setString(3, restaurantusername);
            preparedStmt.setString(4, pass);
            preparedStmt.setString(5, phone);
            preparedStmt.setString(6, address);
            preparedStmt.setDate(7, date);

            preparedStmt.execute();

            System.out.print(Method.colorText("Restaurant " +
restaurantname + " Created Successfully!", "green"));
            System.out.println();
            System.out.println(Method.colorText("Restaurant Username For
Log-In: " + restaurantusername, "green"));
```

```
        } catch (Exception e) {
            createLine(25);
            Method.showDatabaseError();
            createLine(25);
        }

    }

    public static void loginPanel() {

        Scanner input = new Scanner(System.in);

        Method.createHeader("Log-In Panel");


        System.out.println("\nEnter Your Username: ");
        String username = input.next();

        if(username.equals("-1")){
            Menu.startMenu();
        }

        System.out.println("Enter Your Password: ");
        String password = input.next();

        if(password.equals("-1")){
            Menu.startMenu();
        }

        PreparedStatement ps;
        ResultSet rs;

        if (username.contains("restaurant_")) {

            String query = "SELECT * FROM restaurants WHERE
`restaurantusername` =? AND `restaurantpassword` =?";


            try {

                ps = Start.con.prepareStatement(query);
                ps.setString(1, username);
                ps.setString(2, password);


                rs = ps.executeQuery();


                if (rs.next()) {

                    String name = rs.getString("restaurantname");
                    String owner = rs.getString("restaurantowner");
                    int id = rs.getInt("restaurantID");
                    Restaurant.ownerMenu(id, name, owner);


                } else {

                    System.err.println("Check Your Username And
Password!");
```

```
                    Menu.startMenu();

                }

            } catch (Exception e) {

                //System.out.println(e);

            }


        } else {

            String query = "SELECT * FROM users WHERE `username` =? AND
`password` =?";

            try {

                ps = Start.con.prepareStatement(query);

                ps.setString(1, username);
                ps.setString(2, password);

                rs = ps.executeQuery();


                if (rs.next()) {

                    int ID = rs.getInt("ID");
                    User.userMenu(username, ID);

                } else {

                    System.err.println("Check Your Username And
Password!");
                    Menu.startMenu();

                }

            } catch (Exception e) {

                //System.out.println(e);

            }

        }

    }

    public static String checkLength(String type, int length) {

        Scanner checker = new Scanner(System.in);
        String input;


        while (true) {
            System.out.println("Enter Your " + type + ":");
            input = checker.nextLine();

            if (input.equals("-1")) {
                Menu.startMenu();
```

```
            }


            if (input.length() > length) {
                System.err.println(type + " Can't Be More Than " + length +
" Characters!");

            } else if (type.equals("Username") &&
input.contains("restaurant_")) {
                System.err.println("You Can't Have 'restaurant_' In Your
Username!");

            } else {
                break;
            }

        }

        return input;
    }

    public static void showDatabaseError() {
        System.err.println("Error! Please Check Database Connection!");

    }

    public static void showMenu(int ID) {

        String foodname;
        String fooding;
        int foodid;
        double price;
        String query = "SELECT (groupname) AS Types FROM `groups` WHERE
restaurantID = ? AND groupID IN (SELECT menugroup FROM menus WHERE
groups.restaurantID = menus.restaurantID);";
        PreparedStatement ps;
        ResultSet rs;


        try {
            ps = Start.con.prepareStatement(query);
            ps.setInt(1, ID);
            rs = ps.executeQuery();

            System.out.println("----------------------------------------
----------------------------------");
            System.out.println("|                                        MENU
|");
            System.out.println("----------------------------------------
----------------------------------");


            while (rs.next()) {

                String type = rs.getString("Types");

                Method.createHeader(type, 77);
                System.out.println();

                String query2 = "SELECT DISTINCT(foodname) AS
Name,foodingredients AS Ing,foodprice AS Price,foodID as FoodID FROM menus
```

```
WHERE restaurantID=? AND menugroup IN (SELECT groupID FROM `groups` WHERE
groupName = ? )";

                try {
                    PreparedStatement ps1 =
Start.con.prepareStatement(query2);
                    ps1.setInt(1, ID);
                    ps1.setString(2, type);

                    ResultSet rs1 = ps1.executeQuery();

                    while (rs1.next()) {
                        foodname = rs1.getString("Name");
                        fooding = rs1.getString("Ing");
                        price = rs1.getDouble("Price");
                        foodid = rs1.getInt("FoodID");

                        System.out.printf("#%-4d %-50s %s %.2f%s", foodid,
foodname + "(" + fooding + ")", "\t\t\t", price, " TL");
                        System.out.println();

                    }
                    Method.createLine(77);

                } catch (Exception e) {
                    System.out.println(e);
                }

            }
        } catch (Exception e) {
            System.out.println(e);
        }


    }

    public static void restaurantShowMenu(int ID, String restaurantName,
String ownerName) {

        String foodname;
        String fooding;
        int foodid;
        double price;
        String query = "SELECT (groupname) AS Types FROM `groups` WHERE
restaurantID = ? AND groupID IN (SELECT menugroup FROM menus WHERE
groups.restaurantID = menus.restaurantID);";
        PreparedStatement ps;
        ResultSet rs;


        try {
            ps = Start.con.prepareStatement(query);
            ps.setInt(1, ID);
            rs = ps.executeQuery();

            System.out.println("----------------------------------------
----------------------------------");
            System.out.println("|                                        MENU
|");
            System.out.println("----------------------------------------
----------------------------------");
```

```
            while (rs.next()) {

                String type = rs.getString("Types");

                Method.createHeader(type, 77);
                System.out.println();

                String query2 = "SELECT DISTINCT(foodname) AS
Name,foodingredients AS Ing,foodprice AS Price,foodID as FoodID FROM menus
WHERE restaurantID=? AND menugroup IN (SELECT groupID FROM `groups` WHERE
groupName = ? )";

                try {
                    PreparedStatement ps1 =
Start.con.prepareStatement(query2);
                    ps1.setInt(1, ID);
                    ps1.setString(2, type);

                    ResultSet rs1 = ps1.executeQuery();

                    while (rs1.next()) {
                        foodname = rs1.getString("Name");
                        fooding = rs1.getString("Ing");
                        price = rs1.getDouble("Price");
                        foodid = rs1.getInt("FoodID");

                        System.out.printf("#%-4d %-50s %s %.2f%s", foodid,
foodname + "(" + fooding + ")", "\t\t\t", price, " TL");
                        System.out.println();

                    }
                    Method.createLine(77);

                } catch (Exception e) {
                    System.out.println(e);
                }

            }
        } catch (Exception e) {
            System.out.println(e);
        }

        Scanner inp = new Scanner(System.in);

        Method.createLine(22);
        System.out.println("Enter '1' To Edit Menu");
        System.out.println("Enter '-1' To Return");
        Method.createLine(22);
        System.out.println("Your Choice: ");
        int ans = inp.nextInt();

        if (ans == -1) {
            Restaurant.ownerMenu(ID, restaurantName, ownerName);
        } else if (ans == 1) {
            editMenu(ID, restaurantName, ownerName);
        } else {
            Restaurant.ownerMenu(ID, restaurantName, ownerName);

        }
```

```
    }

    public static void editMenu(int ID, String restaurantName, String
ownerName) {

        Scanner inp = new Scanner(System.in);
        Method.createLine("Enter '2' To Edit Food Groups");
        System.out.println("Enter '1' To Edit Food");
        System.out.println("Enter '2' To Edit Food Groups");
        System.out.println("Enter '-1' To Return");
        Method.createLine("Enter '2' To Edit Food Groups");

        System.out.print("Your Choice: ");
        int ans = inp.nextInt();

        switch (ans) {
            case 1:
                editFoodMenu(ID, restaurantName, ownerName);
            case 2:
                editFoodGroup(ID, restaurantName, ownerName);
            case -1:
                restaurantShowMenu(ID, restaurantName, ownerName);

            default:
                editMenu(ID, restaurantName, ownerName);
        }

        System.out.println();


    }

    private static void editFoodMenu(int ID, String restaurantName, String
ownerName) {
        Scanner inp = new Scanner(System.in);
        Method.createHeader("Edit Food", 32);
        System.out.println();
        System.out.println("Enter '1' To Add New Food");
        System.out.println("Enter '2' To Remove Food");
        System.out.println("Enter '3' To Edit Food");
        System.out.println("Enter '-1' To Return");
        createLine("Enter '3' To Change Food's Group");
        System.out.print("Your Choice: ");
        int ans = inp.nextInt();


        switch (ans) {
            case 1:
                addFood(ID, restaurantName, ownerName);
            case 2:
                removeFood(ID, restaurantName, ownerName);

            case 3:
                editFood(ID, restaurantName, ownerName);
            case -1:
                editMenu(ID, restaurantName, ownerName);
            default:
                editMenu(ID, restaurantName, ownerName);
```

```
        }

    }

    private static void addFood(int ID, String restaurantName, String
ownerName) {

        PreparedStatement ps;
        ResultSet rs;
        Scanner inp = new Scanner(System.in);
        Scanner inp1 = new Scanner(System.in);

        String query = "SELECT groupname,groupID FROM `groups` WHERE
restaurantID = ? ";


        try {


            Method.createHeader("Add New Food");
            System.out.println();

            System.out.println("(Enter '0' To See Groups)");
            System.out.print("Enter Food's Group ID: #");
            int groupID = inp.nextInt();

            if (groupID == -1) {
                editMenu(ID, restaurantName, ownerName);
            } else if (groupID == 0) {
                ps = Start.con.prepareStatement(query);
                ps.setInt(1, ID);
                rs = ps.executeQuery();

                Method.createHeader("GRUOPS", 26);
                System.out.println();
                while (rs.next()) {
                    System.out.printf("|%-20s(#%s|\n",
rs.getString("groupname"), rs.getInt("groupID") + ")");
                }
                Method.createLine(26);

                addFood(ID, restaurantName, ownerName);

            }


            System.out.print("Enter Food's Name: ");
            String name = inp1.nextLine();
            if (name.contains("-1")) {
                editFoodMenu(ID, restaurantName, ownerName);
            }

            System.out.print("Enter Food's Ingredients: ");
            String ing = inp1.nextLine();
            if (ing.contains("-1")) {
                editFoodMenu(ID, restaurantName, ownerName);
            }

            System.out.print("Enter Food's Price: ");
            double price = inp1.nextDouble();
            if (price == -1) {
```

```
                editFoodMenu(ID, restaurantName, ownerName);
            }

            String addQuery = "INSERT INTO menus
(restaurantID,foodname,foodingredients,foodprice,menugroup) VALUES
(?,?,?,?,?)";

            ps = Start.con.prepareStatement(addQuery);

            ps.setInt(1, ID);
            ps.setString(2, name);
            ps.setString(3, ing);
            ps.setDouble(4, price);
            ps.setInt(5, groupID);

            ps.executeUpdate();

            System.out.println(colorText("Food Created Successfully!",
"green"));
            editMenu(ID, restaurantName, ownerName);
        } catch (SQLException e) {
            System.out.println(e);
        }


    }

    private static void removeFood(int ID, String restaurantName, String
ownerName) {

        PreparedStatement ps;
        ResultSet rs;
        Scanner inp = new Scanner(System.in);
        String query = "SELECT foodname,foodid FROM menus WHERE
restaurantID = ?";
        try {
            ps = Start.con.prepareStatement(query);
            ps.setInt(1, ID);
            rs = ps.executeQuery();
            ArrayList<Integer> exist = new ArrayList<>();

            Method.createHeader("Remove Food", 26);
            System.out.println();
            while (rs.next()) {
                System.out.printf("|%-20s(#%s|\n",  rs.getString("foodname"),
rs.getInt("foodid") + ")");
                exist.add(rs.getInt("foodid"));
            }


            Method.createLine(26);
            while (true) {
                System.out.print("Select Food To Remove: #");
                int removeFood = inp.nextInt();

                for (int i = 0; i < exist.size(); i++) {
                    if (removeFood == exist.get(i)) {

                        String removeQuery = "DELETE FROM menus WHERE
restaurantID =? AND foodID =?";
                        ps = Start.con.prepareStatement(removeQuery);
```

```
                        ps.setInt(1, ID);
                        ps.setInt(2, removeFood);
                        ps.executeUpdate();
                        System.out.println(Method.colorText("Food Removed
Successfully!", "green"));
                        editMenu(ID, restaurantName, ownerName);
                        break;

                    } else {

                        System.err.println("ERROR!");
                        removeFood(ID, restaurantName, ownerName);

                    }
                }

                for (int i = 0; i < exist.size(); i++) {
                    exist.remove(i);
                }
            }


        } catch (Exception e) {

        }

    }

    private static void editFood(int ID, String restaurantName, String
ownerName) {

        PreparedStatement ps;
        ResultSet rs;
        Scanner liner = new Scanner(System.in);
        Scanner inp = new Scanner(System.in);
        String query = "SELECT foodname,foodID FROM menus WHERE
restaurantID = ?";

        Method.createHeader("Edit Food", 26);
        System.out.println();

        try {

            ps = Start.con.prepareStatement(query);
            ps.setInt(1, ID);
            rs = ps.executeQuery();
            ArrayList<Integer> exist = new ArrayList<>();

            while (rs.next()) {
                System.out.printf("|%-20s(#%s|\n",  rs.getString("foodname"),
rs.getInt("foodID") + ")");
                exist.add(rs.getInt("foodID"));
            }

            System.out.print("Select Food To Edit: #");
            int editFood = inp.nextInt();
            int firstCnt = 0;
            while (true) {

                for (int i = 0; i < exist.size(); i++) {
                    if (editFood == exist.get(i)) {
```

```
                        firstCnt = 0;
                        String selectQuery = "SELECT foodname FROM menus
WHERE foodID =? AND restaurantID =?";
                        ps = Start.con.prepareStatement(selectQuery);
                        ps.setInt(1, editFood);
                        ps.setInt(2, ID);
                        rs = ps.executeQuery();
                        rs.next();
                        String selected = rs.getString("foodname");

                        Method.createHeader("Selected Food: " + selected);
                        System.out.println();
                        Method.createLine("Enter '2' To Change Food's
Ingredients");
                        System.out.println("Enter '1' To Change Food's
Name");
                        System.out.println("Enter '2' To Change Food's
Ingredients");
                        System.out.println("Enter '3' To Change Food's
Price");
                        System.out.println("Enter '4' To Change Food's
Group");
                        System.out.println("Enter '-1' To Return");
                        Method.createLine("Enter '2' To Change Food's
Ingredients");
                        System.out.print("Your Choice: ");
                        int edit = inp.nextInt();

                        switch (edit) {
                            case 1:
                                String updateQuery = "UPDATE menus SET
foodname = ? WHERE restaurantID = ? AND foodID = ?;";
                                System.out.print("Enter Food's New Name:
");
                                String newName = liner.nextLine();

                                ps =
Start.con.prepareStatement(updateQuery);
                                ps.setString(1, newName);
                                ps.setInt(2, ID);
                                ps.setInt(3, editFood);
                                ps.executeUpdate();

                                System.out.println(colorText("Food's Name
Is Changed!", "green"));

                                break;
                            case 2:
                                String updateQuery1 = "UPDATE menus SET
foodingredients = ? WHERE restaurantID = ? AND foodID = ?;";

                                System.out.print("Enter Food's New
Ingredients: ");
                                String newIngredient = liner.nextLine();

                                ps =
Start.con.prepareStatement(updateQuery1);
                                ps.setString(1, newIngredient);
                                ps.setInt(2, ID);
                                ps.setInt(3, editFood);
                                System.out.println(colorText("Food's
```

```
Ingredients Are Changed!", "green"));
                                break;
                            case 3:
                                String updateQuery2 = "UPDATE menus SET
foodprice = ? WHERE restaurantID = ? AND foodID = ?;";

                                System.out.print("Enter Food's New Price:
");
                                double newPrice = inp.nextDouble();
                                ps =
Start.con.prepareStatement(updateQuery2);
                                ps.setDouble(1, newPrice);
                                ps.setInt(2, ID);
                                ps.setInt(3, editFood);

                                System.out.println(colorText("Food's Price
Is Changed!", "green"));
                                break;

                            case 4:
                                String queryGroup = "SELECT
groupname,groupID FROM `groups` WHERE restaurantID = ? ";

                                ps =
Start.con.prepareStatement(queryGroup);
                                ps.setInt(1, ID);
                                rs = ps.executeQuery();

                                ArrayList<Integer> groups = new
ArrayList<>();


                                Method.createHeader("GRUOPS", 26);
                                System.out.println();

                                while (rs.next()) {
                                    System.out.printf("|%-20s(#%s|\n",
rs.getString("groupname"), rs.getInt("groupID") + ")");
                                    groups.add(rs.getInt("groupID"));
                                }

                                Method.createLine(26);
                                Method.freezeScreen();

                                System.out.print("Enter Food's New Group:
#");
                                int newGroup = inp.nextInt();

                                int cnt = 0;

                                for (int x = 0; x < groups.size(); x++) {

                                    if (newGroup == groups.get(x)) {
                                        String updateQuery3 = "UPDATE menus
SET menugroup = ? WHERE restaurantID = ? AND foodID = ?;";
                                        ps =
Start.con.prepareStatement(updateQuery3);
                                        ps.setDouble(1, newGroup);
                                        ps.setInt(2, ID);
                                        ps.setInt(3, editFood);
                                        ps.executeUpdate();
```

```
System.out.println(colorText("Food's Group Is Changed!", "green"));
                                        cnt = 0;
                                        break;
                                    } else {
                                        cnt++;

                                    }
                                }

                                if (cnt == groups.size()) {
                                    System.err.println("ERROR!");
                                    cnt = 0;
                                }
                                for (int j = 0; j < groups.size(); j++) {
                                    groups.remove(j);
                                }

                            case -1:
                                editFoodMenu(ID, restaurantName,
ownerName);
                                break;
                            default:
                        }
                    } else {
                        firstCnt++;
                    }

                    if (firstCnt == exist.size()) {
                        System.err.println("ERROR!");
                        firstCnt = 0;
                    }

                }
            }
        } catch (Exception e) {
            System.out.println(e);
        }


        editFoodMenu(ID, restaurantName, ownerName);


    }

    private static void editFoodGroup(int ID, String restaurantName, String
ownerName) {
        Scanner inp = new Scanner(System.in);
        Method.createHeader("Edit Food", 35);
        System.out.println();
        System.out.println("Enter '1' To Create New Food Group");
        System.out.println("Enter '2' To Remove Food Group");
        System.out.println("Enter '3' To Edit Food Group's Name");
        System.out.println("Enter '-1' To Return");
        createLine("Enter '3' To Edit Food Group's Name");
        System.out.print("Your Choice: ");
        int ans = inp.nextInt();

        switch (ans) {
            case 1:
                addFoodGroup(ID, restaurantName, ownerName);
```

```
                break;
            case 2:
                removeFoodGroup(ID, restaurantName, ownerName);
                break;
            case 3:
                editFoodGroupName(ID, restaurantName, ownerName);
                break;
            case -1:
                editMenu(ID, restaurantName, ownerName);
                break;


        }


    }

    private static void editFoodGroupName(int ID, String restaurantName,
String ownerName) {

        ResultSet rs;
        PreparedStatement ps;
        Scanner inp = new Scanner(System.in);
        String queryGroup = "SELECT groupname,groupID FROM `groups` WHERE
restaurantID = ? ";

        try {

            ps = Start.con.prepareStatement(queryGroup);
            ps.setInt(1, ID);
            rs = ps.executeQuery();
            System.out.println("(Enter '0' To See Groups)");
            System.out.print("Enter Group's ID: #");
            int groupid = inp.nextInt();
            if (groupid == 0) {
                Method.createHeader("GROUPS", 26);
                System.out.println();

                while (rs.next()) {
                    System.out.printf("|%-20s(#%s|\n",
rs.getString("groupname"), rs.getInt("groupID") + ")");
                }

                Method.createLine(26);
                editFoodGroupName(ID, restaurantName, ownerName);


            }
            System.out.println();
            Method.freezeScreen();
            String query = "SELECT groupname FROM `groups` WHERE
restaurantID = ? AND groupID =?";

            ps = Start.con.prepareStatement(query);
            ps.setInt(1, ID);
            ps.setInt(2, groupid);
            rs = ps.executeQuery();
            rs.next();
            String groupname = rs.getString("groupname");

            if (groupid == -1) {
```

```
                editFoodGroup(ID, restaurantName, ownerName);
            }


            createHeader("Selected Group: " + groupname);
            System.out.println();
            Scanner liner = new Scanner(System.in);
            System.out.print("Enter New Name: ");
            String newName = liner.nextLine();

            String updateQuery = "UPDATE `groups` SET groupname = ? WHERE
restaurantID = ? AND groupID = ?";
            ps = Start.con.prepareStatement(updateQuery);
            ps.setString(1, newName);
            ps.setInt(2, ID);
            ps.setInt(3, groupid);

            ps.executeUpdate();

            System.out.println(colorText("Group's Name Changed
Successfully!", "green"));
            editFoodGroup(ID, restaurantName, ownerName);
        } catch (Exception e) {
            System.out.println(e);
            System.err.println("ERROR!");
        }

    }

    private static void addFoodGroup(int ID, String restaurantName, String
ownerName) {

        PreparedStatement ps;

        Scanner inp = new Scanner(System.in);
        createHeader("Create New Food Group");
        System.out.println();

        System.out.print("Enter New Food Group's Name: ");
        String groupName = inp.nextLine();

        if (groupName.equals("-1")) {
            editFoodGroup(ID, restaurantName, ownerName);
        }

        String query = "INSERT INTO `groups` (restaurantID, groupname)
VALUES (?, ?)";
        try {
            ps = Start.con.prepareStatement(query);
            ps.setInt(1, ID);
            ps.setString(2, groupName);
            ps.executeUpdate();

            System.out.println(colorText("Food Group Created Successfully!",
"green"));

            editFoodGroup(ID, restaurantName, ownerName);

        } catch (Exception e) {
            System.out.println(e);
        }
```

```
    }

    private static void removeFoodGroup(int ID, String restaurantName,
String ownerName) {

        PreparedStatement ps;
        ResultSet rs;
        Scanner inp = new Scanner(System.in);
        createHeader("Remove Food Group");
        System.out.println();

        System.out.println("(Enter '0' To See Groups)");
        System.out.print("Enter Food Group's ID: ");
        int groupID = inp.nextInt();

        if (groupID == -1) {
            editFoodGroup(ID, restaurantName, ownerName);
        } else if (groupID == 0) {
            String queryGroup = "SELECT groupname,groupID FROM `groups`
WHERE restaurantID = ? ";

            try {

                ps = Start.con.prepareStatement(queryGroup);
                ps.setInt(1, ID);
                rs = ps.executeQuery();


                Method.createHeader("GROUPS", 26);
                System.out.println();

                while (rs.next()) {
                    System.out.printf("|%-20s(#%s|\n",
rs.getString("groupname"), rs.getInt("groupID") + ")");
                }

                removeFoodGroup(ID, restaurantName, ownerName);
                Method.createLine(26);
                System.out.println();
                Method.freezeScreen();


            } catch (Exception e) {

            }
        }


        String query = "DELETE FROM `groups` WHERE restaurantID = ? AND
groupID = ?";
        String query2 = "DELETE FROM menus WHERE menugroup = ? AND
restaurantID =?";
        try {
            ps = Start.con.prepareStatement(query);
            ps.setInt(1, ID);
            ps.setInt(2, groupID);
            ps.executeUpdate();

            ps = Start.con.prepareStatement(query2);
            ps.setInt(2, ID);
            ps.setInt(1, groupID);
```

```
            ps.executeUpdate();

            System.out.println(colorText("Food Group Removed Successfully!",
"green"));

            editFoodGroup(ID, restaurantName, ownerName);
        } catch (Exception e) {
            System.out.println(e);
        }

    }

    public static String makeSpace(String text, int Total) {

        String length = "";

        Total -= text.length();
        Total--;

        for (int i = 0; i < Total; i++) {
            length += " ";
        }

        return length;

    }

    public static void seeLastOrders(int ID, String restaurantName, String
ownerName) {
        createHeader("Last 5 Orders");
        System.out.println();
        PreparedStatement ps;
        ResultSet rs;
        String query = "SELECT
orders.orderID,orders.userID,orders.foodID,orders.amount,orders.price,
menus.foodname,menus.foodID,menus.restaurantID,
users.fullname,users.phone,users.address,users.ID\n" +
                "FROM orders, menus, users\n" +
                "WHERE menus.restaurantID = ? AND orders.foodID =
menus.foodID AND orders.userID = users.ID AND orders.done = 1 ORDER BY
orderID DESC;\n";

        try {

            ps = Start.con.prepareStatement(query);
            ps.setInt(1, ID);
            rs = ps.executeQuery();

            int orderCounter = 1;



            while (rs.next() && orderCounter < 6) {

                int order = rs.getInt("orderID");
                int amount = rs.getInt("amount");
                double price = rs.getDouble("price");

                String foodname = rs.getString("foodname");
                String customerName = rs.getString("fullname");
                String address = rs.getString("address");
```

```
                String phone = rs.getString("phone");

                System.out.printf("Order ID: %d\nCustomer
Name: %s\nCustomer Address: %s\nCustomer Phone: %s\nCustomer
Order: %s(%d)\nTotal: %.2f\n", order, customerName, address, phone,
foodname, amount, price);
                System.out.println("-----------------------------------");
                orderCounter++;


            }

            freezeScreen();
            Restaurant.ownerMenu(ID, restaurantName, ownerName);

        } catch (Exception e) {
            System.out.println(e);
        }
    }

    private static int justID = 0;
    private static String restName;
    private static String owName;

    public static void checkOrders(int ID, String restaurantName, String
ownerName, int prevCount) {
        justID = ID;
        restName = restaurantName;
        owName = ownerName;

        createHeader("Check Orders");
        System.out.println();



T1 thread = new T1();
T1.flag = true;
thread.run();




    }

    public static class T1 implements Runnable{
public static int prevCount = 0;
        private static boolean flag = true;


        public void run() {

            while (flag) {

                PreparedStatement ps;
                ResultSet rs;

                String query1 = "Select Count(orderID) from orders WHERE
restaurantID = 1 AND done = 0";
                String query2 = "SELECT
orders.orderID,orders.userID,orders.foodID,orders.amount,orders.price,
menus.foodname,menus.foodID,menus.restaurantID,
```

```
users.fullname,users.phone,users.address,users.ID\n" +
                        "FROM orders, menus, users\n" +
                        "WHERE menus.restaurantID = ? AND orders.foodID =
menus.foodID AND orders.userID = users.ID AND orders.done = 0 ORDER BY
orders.orderID ASC;\n";


                try {

                    ps = Start.con.prepareStatement(query1);
                    rs = ps.executeQuery();
                    rs.next();
                    int count = rs.getInt(1);

                    if (count != prevCount) {
                        ps = Start.con.prepareStatement(query2);
                        ps.setInt(1,justID);
                        rs = ps.executeQuery();


                        while (rs.next()) {
                            int order = rs.getInt("orderID");
                            int amount = rs.getInt("amount");
                            double price = rs.getDouble("price");

                            String foodname = rs.getString("foodname");
                            String customerName = rs.getString("fullname");
                            String address = rs.getString("address");
                            String phone = rs.getString("phone");

                            System.out.printf("Order ID: %d\nCustomer
Name: %s\nCustomer Address: %s\nCustomer Phone: %s\nCustomer
Order: %s(%d)\nTotal: %.2f\n", order, customerName, address, phone,
foodname, amount, price);
                            System.out.println("-------------------------
--------");
                        }

                        System.out.println("Enter '1' To Confirm The
Order");
                        System.out.println("Enter '2' To Remove The
Order");
                        System.out.println("Enter '-1' To Return!");
createLine("Enter '1' To Confirm The Order-----");
                        System.out.print("Your Choice: ");


                        Scanner inp = new Scanner(System.in);


                        System.out.println();

                        int answer= inp.nextInt();

                        switch (answer){
                            case 1:
                                flag = false;
                                doneOrder(justID,restName,owName);
                                break;
                            case 2:
                                flag = false;
```

```
                                removeOrder(justID,restName,owName);

                                break;

                            case -1:
                                flag = false;

Restaurant.ownerMenu(justID,restName,owName);

                                break;
                            default:
                                flag = false;
                                checkOrders(justID,restName,owName,
prevCount);

                        }

                    }

                    // System.out.println("Count = " + count + " And
prevCount = " + prevCount);

                    prevCount = count;

                } catch (Exception e) {

                }


            }


        }

    }

    public static void doneOrder(int ID,String restaurantName, String
ownerName){
        PreparedStatement ps;

        String doneQuery = "UPDATE orders SET done = 1 WHERE orderID =? AND
restaurantID =?";
        Scanner inp = new Scanner(System.in);

        System.out.println("Enter Order ID:");
        int orderid = inp.nextInt();

        if(orderid == -1){
            checkOrders(ID,restaurantName,ownerName,0);
        }

        try {



            ps = Start.con.prepareStatement(doneQuery);
            ps.setInt(1,orderid);
            ps.setInt(2,ID);
            ps.executeUpdate();

            System.out.println(colorText("Order is Done!","green"));
```

```
            freezeScreen();

            checkOrders(ID,restaurantName,ownerName,0);


        }catch (Exception e){
            System.out.println(e);
            showDatabaseError();
        }

    }

    public static void removeOrder(int ID, String restaurantName, String
ownerName){

        PreparedStatement ps;
        String removeQuery = "DELETE FROM orders WHERE orderID =? AND
restaurantID =?";

        Scanner inp = new Scanner(System.in);

        System.out.println("Enter Order ID:");
        int orderid = inp.nextInt();

        if(orderid == -1){
            checkOrders(ID,restaurantName,ownerName,0);
        }

        try {

            ps = Start.con.prepareStatement(removeQuery);
            ps.setInt(1,orderid);
            ps.setInt(2,ID);
            ps.executeUpdate();

            System.out.println(colorText("Order is Removed!","green"));

            freezeScreen();

            checkOrders(ID,restaurantName,ownerName,0);


        }catch (Exception e){
            System.out.println(e);
            showDatabaseError();
        }

    }


}
```

interface DisplayMethods

```
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.Scanner;

public interface DisplayMethods {

         static void showMenus(int ID){
            String foodname;
            String fooding;
            int foodid;
            double price;
            String query = "SELECT (groupname) AS Types FROM `groups` WHERE
restaurantID = ? AND groupID IN (SELECT menugroup FROM menus WHERE
groups.restaurantID = menus.restaurantID);";
            PreparedStatement ps;
            ResultSet rs;

            try {
                ps = Start.con.prepareStatement(query);
                ps.setInt(1, ID);
                rs = ps.executeQuery();

                System.out.println("----------------------------------------
----------------------------------------");
                System.out.println("|
MENU                                    |");
                System.out.println("----------------------------------------
----------------------------------------");


                while (rs.next()) {

                    String type = rs.getString("Types");

                    Method.createHeader(type, 77);
                    System.out.println();

                    String query2 = "SELECT DISTINCT(foodname) AS
Name,foodingredients AS Ing,foodprice AS Price,foodID as FoodID FROM menus
WHERE restaurantID=? AND menugroup IN (SELECT groupID FROM `groups` WHERE
groupName = ? )";

                    try {
                        PreparedStatement ps1 =
Start.con.prepareStatement(query2);
                        ps1.setInt(1, ID);
                        ps1.setString(2, type);

                        ResultSet rs1 = ps1.executeQuery();

                        while (rs1.next()) {
                            foodname = rs1.getString("Name");
                            fooding = rs1.getString("Ing");
                            price = rs1.getDouble("Price");
                            foodid = rs1.getInt("FoodID");

                            System.out.printf("#%-4d %-50s %s %.2f%s",
foodid, foodname + "(" + fooding + ")", "\t\t\t", price, " TL");
```

```
                            System.out.println();

                        }
                        Method.createLine(77);

                    } catch (Exception e) {
                        System.out.println(e);
                    }

                }
            } catch (Exception e) {
                System.out.println(e);
            }

        };

         static void showRestaurantMenus(int ID, String restaurantName,
String ownerName) {

            String foodname;
            String fooding;
            int foodid;
            double price;
            String query = "SELECT (groupname) AS Types FROM `groups` WHERE
restaurantID = ? AND groupID IN (SELECT menugroup FROM menus WHERE
groups.restaurantID = menus.restaurantID);";
            PreparedStatement ps;
            ResultSet rs;


            try {
                ps = Start.con.prepareStatement(query);
                ps.setInt(1, ID);
                rs = ps.executeQuery();

                System.out.println("----------------------------------------
----------------------------------------");
                System.out.println("|
MENU                                    |");
                System.out.println("----------------------------------------
----------------------------------------");


                while (rs.next()) {

                    String type = rs.getString("Types");

                    Method.createHeader(type, 77);
                    System.out.println();

                    String query2 = "SELECT DISTINCT(foodname) AS
Name,foodingredients AS Ing,foodprice AS Price,foodID as FoodID FROM menus
WHERE restaurantID=? AND menugroup IN (SELECT groupID FROM `groups` WHERE
groupName = ? )";

                    try {
                        PreparedStatement ps1 =
Start.con.prepareStatement(query2);
                        ps1.setInt(1, ID);
                        ps1.setString(2, type);
```

```
                        ResultSet rs1 = ps1.executeQuery();

                        while (rs1.next()) {
                            foodname = rs1.getString("Name");
                            fooding = rs1.getString("Ing");
                            price = rs1.getDouble("Price");
                            foodid = rs1.getInt("FoodID");

                            System.out.printf("#%-4d %-50s %s %.2f%s",
foodid, foodname + "(" + fooding + ")", "\t\t\t", price, " TL");
                            System.out.println();

                        }
                        Method.createLine(77);

                    } catch (Exception e) {
                        System.out.println(e);
                    }

                }
            } catch (Exception e) {
                System.out.println(e);
            }

            Scanner inp = new Scanner(System.in);

            Method.createLine(22);
            System.out.println("Enter '1' To Edit Menu");
            System.out.println("Enter '-1' To Return");
            Method.createLine(22);
            System.out.println("Your Choice: ");
            int ans = inp.nextInt();

            if (ans == -1) {
                Restaurant.ownerMenu(ID, restaurantName, ownerName);
            } else if (ans == 1) {
                Method.editMenu(ID, restaurantName, ownerName);
            } else {
                Restaurant.ownerMenu(ID, restaurantName, ownerName);

            }



        }


    }
```

Sample Outputs

A)Main Menu

```
--------------------
|Welcome To FoodPack|
--------------------
-----------
|Main Menu|
-------------------------------------
Enter '1' To Log-In
Enter '2' to Register
Enter '3' to Check Database Connection
Enter '4' to See Credits
Enter '-1' to Exit
-------------------------------------
Your Choice:
```

A/1)Log-In Panel (Input ‘1’ In A)

a)To log-in as a user (Just enter username and password)

```
--------------
|Log-In Panel|
--------------
Enter Your Username:
berk70189
Enter Your Password:
alpalp
--------------------
|Welcome, berk70189|
--------------------
```

b)To log-in as a restaurant (Write ‘restaurant_’ before the name of your restaurant)

```
--------------
|Log-In Panel|
--------------
Enter Your Username:
restaurant_berkpide
Enter Your Password:
alpalp
--------------------
|Welcome, Berk Muslu|
--------------------
--------------------------
|Restaurant Name: Berk Pide|
--------------------------
```

A/2)Register Panel (Input ‘2’ In A)

```
---------------
|Register Panel|
----------------------------------------
Enter '1' to Register As User
Enter '2' to Register As Restaurant Owner
Enter '-1' to Return Log-In Panel
----------------------------------------
Your Choice:
```

a)Register as User (Input ‘1’ In A/2)

```
---------------------
|Register a New User|
---------------------
Enter Your Full Name:
Alp Muslu
Enter Your Username:
alpmsl06
Enter Your Password:
alpalp
Enter Your Phone Number:
5425520888
Enter Your Address:
Deneme
User alpmsl06 Created Successfully!
```

b)Register as Restaurant (Input ‘2’ In A/2)

```
--------------------------------
|Register a New Restaurant Owner|
--------------------------------
Enter Your Full Name:
Alp Muslu
Enter Your Restaurant Name:
Kömürde Pizza
Enter Your Password:
alpalp
Enter Your Phone Number:
5242523242
Enter Your Address:
Güzelbahçe/İzmir
Restaurant Kömürde Pizza Created Successfully!
Restaurant Username For Log-In: restaurant_komurdepizza
```

c)Returns To A (Input ‘-1’ In A/2)

A/3)Check Database Connection (Input ‘3’ In A)

```
-----------------
|DATABASE STATUS|
-----------------------------
|Database Connection : OK!   |
-----------------------------
```

A/4)See Credits (Input ‘4’ In A)

```
---------
|Credits|
-----------------------------------
Project Name: FoodPack
-----------------------------------
Lecture: SE-116
-----------------------------------
Creators:
Berk Muslu
Ömer Yakup Cankurtaran
-----------------------------------
```

B)User Menu (After Log-In As User (A-1))

```
-----------
|User Menu|
--------------------------------
Enter '1' To Order Food
Enter '2' to See Profile
Enter '-1' to Return Log-In Panel
--------------------------------
Your Choice:
```

B/1)Food Order Menu (Input ‘1’ In B)

```
-------------
|Order Menu|
-------------------------------
Enter '1' to List Restaurants
Enter '2' to Search Restaurant
Enter '3' to Search Food
Enter '-1' to Return User Menu
-------------------------------
Your Choice:
```

B/1-a)Listing Restaurants (Input ‘1’ In B/1)

```
-----------------------
|There Are 2 Restaurans|
-----------------------
ID              Restaurant Name         Restaurant Number               Restaurant Address
---             ---------------         -----------------               --------------------------
#1              Başak Pizza             5425520000                      Mükemmel Yer
#2              Berk Pide               5425520899                      Berkin Güzel Evi

-----------------------------
Enter '1' To Choose Restaurant
Enter '-1' To Return
-----------------------------
Your Choice:
```

B/1-a.1)Choosing Restaurant (Input ‘1’ In B/1-a)

```
Enter Restaurant ID: #1
--------------------------------
|Chosen Restaurant: Başak Pizza|
--------------------------------
Enter '1' To See Menu
Enter '2' To Order
Enter '-1' To Return
--------------------------------
Your Choice:
```

B/1-b)Searching Restaurant (Input ‘2’ In B/1)

```
------------------
|Search Restaurant|
------------------
Enter A Property Of The Restaurant
pizza
----------------------
|There Is 1 Restaurant|
----------------------
ID              Restaurant Name          Restaurant Number               Restaurant Address
---             ---------------          -----------------               --------------------------
#1              Başak Pizza              5425520000                      Mükemmel Yer
```

B/1-c)Searching Food (Input ‘3’ In B/1)

```
Enter The Property Of The Food:
pizza

Başak Pizza(#1)
---------------
#4    Vegeterian Pizza(Pizza,Brocoly,Tomatoe,Cheese)        16,00TL

Berk Pide(#2)
-------------
#6    Nutella Pizza(Pizza, Nutella)                         22,00TL
#7    Toblerone Pizza(Pizza, Toblerone)                     24,00TL
```

B/2)Showing Profile (Input ‘2’ In B)

```
--------------
|User Profile|
--------------
Full Name: Berk Muslu
Username: berk70189
Password: alpalp
Phone Number: 5425520899
Address: Güzelbahçe İzmir
Total Order: 20
Money Spent So Far: 1544.0 TL
------------------------------
Enter '1' to Edit Phone Number
Enter '2' to Edit Address
Enter '-1' to Return
------------------------------
Your Choice:
```

B/2-a)Editing Phone Number (Input ‘1’ In B/2)

```
Your Choice: 1
Please Enter New Phone Number:
5555555555
Phone Number Changed To 5555555555
```

B/2-b)Editing Address (Input ‘2’ In B/2)

```
Your Choice: 2
Please Enter New Address:
Deneme
Address Changed To Deneme
```

B/1-a.1a)Seeing Menu (Input ‘1’ In B/1-a.1)

```
--------------------------------------------------------------------------------
|                                    MENU                                      |
--------------------------------------------------------------------------------


------------
|Hamburgers|
--------------------------------------------------------------------------------
#3    New Hamburger(Hamburger,Mushroom)                              32,00 TL
#5    Cheeseburger(Hamburger,Mushroom,Cheddar)                       35,00 TL
#8    Lokum Burger((Ham,Meat,Onion,Cheddar))                         42,50 TL
--------------------------------------------------------------------------------


--------
|Pizzas|
--------------------------------------------------------------------------------
#4    Vegeterian Pizza(Pizza,Brocoly,Tomatoe,Cheese)                 16,00 TL
#9    Pizza With Tuna(Pizza,Tuna,Tomatoe Souce)                      25,00 TL
--------------------------------------------------------------------------------
```

B/1-a.1b)Seeing Menu (Input '2' In B/1-a.1)

```
Enter Food ID: #4
--------------------------------
|Selected Food: Vegeterian Pizza|
--------------------------------
Enter Amount:3
------------------------------------------------------------
|                Order Confirmation Screen                 |
------------------------------------------------------------
|Full Name: Berk Muslu                                     |
|Address: Deneme                                           |
|Phone Number: 5555555555                                  |
------------------------------------------------------------
|Selected Food: Vegeterian Pizza(3)                        |
|Price: 48.0 TL                                            |
------------------------------------------------------------
|Enter 1 To Confirm                                        |
|Enter -1 To Cancel                                        |
------------------------------------------------------------
Your Choice: 1
Order#23 Is Ordered Succesfully!
```

C)Owner Menu

C/1)See Orders

```
--------------
|Check Orders|
--------------
Order ID: 14
Customer Name: Berk Muslu
Customer Address: Deneme
Customer Phone: 5555555555
Customer Order: New Hamburger(4)
Total: 128,00
---------------------------------
Order ID: 23
Customer Name: Berk Muslu
Customer Address: Deneme
Customer Phone: 5555555555
Customer Order: Vegeterian Pizza(3)
Total: 48,00
---------------------------------
Enter '1' To Confirm The Order
Enter '2' To Remove The Order
Enter '-1' To Return!
---------------------------------
Your Choice:
```

C/1.a)Confirming Order

```
Your Choice:
1
Enter Order ID:
23
Order is Done!
```

C/1.b)Deleting Order

```
Your Choice:
2
Enter Order ID:
14
Order is Removed!
```

C/2)See Menu

```
Your Choice: 2
--------------------------------------------------------------------------------
|                                    MENU                                      |
--------------------------------------------------------------------------------


------------
|Hamburgers|
--------------------------------------------------------------------------------
#3    New Hamburger(Hamburger,Mushroom)                              32,00 TL
#5    Cheeseburger(Hamburger,Mushroom,Cheddar)                       35,00 TL
#8    Lokum Burger((Ham,Meat,Onion,Cheddar))                         42,50 TL
--------------------------------------------------------------------------------


--------
|Pizzas|
--------------------------------------------------------------------------------
#4    Vegeterian Pizza(Pizza,Brocoly,Tomatoe,Cheese)                 16,00 TL
#9    Pizza With Tuna(Pizza,Tuna,Tomatoe Souce)                      25,00 TL
--------------------------------------------------------------------------------
----------------------
Enter '1' To Edit Menu
Enter '-1' To Return
----------------------
Your Choice:
```

C/2.a)Editing Menu

```
Your Choice:
1
-----------------------------
Enter '1' To Edit Food
Enter '2' To Edit Food Groups
Enter '-1' To Return
-----------------------------
Your Choice:
```

C/2.a)Edit Food

```
Enter '1' To Edit Food
Enter '2' To Edit Food Groups
Enter '-1' To Return
-----------------------------
Your Choice: 1

-----------
|Edit Food|
-------------------------------
Enter '1' To Add New Food
Enter '2' To Remove Food
Enter '3' To Edit Food
Enter '-1' To Return
-------------------------------
Your Choice: 1
--------------
|Add New Food|
--------------
(Enter '0' To See Groups)
Enter Food's Group ID: #4
Enter Food's Name: Mantar Corbasi
Enter Food's Ingredients: Mantar Corbasi, Ekmek, Ayran
Enter Food's Price: 14
Food Created Successfully!
```

C/2.a)Edit Food Group

```
Enter '1' To Edit Food
Enter '2' To Edit Food Groups
Enter '-1' To Return
-----------------------------
Your Choice: 2

-----------
|Edit Food|
-----------------------------------
Enter '1' To Create New Food Group
Enter '2' To Remove Food Group
Enter '3' To Edit Food Group's Name
Enter '-1' To Return
-----------------------------------
Your Choice: 1
------------------------
|Create New Food Group|
------------------------
Enter New Food Group's Name: Ek Lezzetler
Food Group Created Successfully!
```

C/3)See Last 5 Orders

```
|Owner Menu|
---------------------------------
Enter '1' To See Orders
Enter '2' To See Menu
Enter '3' to See Last 5 Orders
Enter '4' to See Restaurant Stats
Enter '-1' to Return Log-In Panel
---------------------------------
Your Choice: 3
---------------
|Last 5 Orders|
---------------
Order ID: 20
Customer Name:  yakup can
Customer Address: koruturk mah. igde sok. 56/5 Balcova/Izmir
Customer Phone: 5075137790
Customer Order: Nutella Pizza(1)
Total: 22,00
----------------------------------
```

C/4)See Restaurant Stats

```
|Owner Menu|
---------------------------------
Enter '1' To See Orders
Enter '2' To See Menu
Enter '3' to See Last 5 Orders
Enter '4' to See Restaurant Stats
Enter '-1' to Return Log-In Panel
---------------------------------
Your Choice: 4

------------------
|Restaurant Stats|
------------------
Restaurant Name: Berk Pide
Owner: Berk Muslu
Phone Number: 5425520899
Address: Berkin Güzel Evi
All Orders So Far: 2
All Income So Far: 44.0 TL
-----------------------------
Enter '1' to Edit Phone Number
Enter '2' to Edit Address
Enter '-1' to Return
-----------------------------
Your Choice:
```

Members of Project Group

Ömer Yakup Cankurtaran
Berk Muslu