

Java Project Description

In this project, we have 2 different panels which are Customer and Manager Panels.

Customer Panel

The screenshot shows the 'Customer' panel of a Java application. At the top, there are two tabs: 'Customer' (selected) and 'Management'. The panel contains a 'Menu' button at the top center. Below it, there are three input fields labeled 'Name', 'Number of People', and 'Age'. To the right of these fields are five radio buttons for table selection: 'Table 1(1-2 people)', 'Table 2(1-2 people)', 'Table 3(3-4 people)', 'Table 4(3-4 people)', and 'Table 5(5-6 people)'. Below the input fields is a 'Complete reservation' button. At the bottom left, there is a label 'Your Order'.

There are 3 different blanks that we are getting inputs from the user. Also, we expect the user to select the proper table which are provided as JRadioButtons.

Errors

When user does not select a table;

This screenshot shows the same 'Customer' panel as before, but with an error message dialog box displayed in the center. The dialog box has a green header with the title 'Message' and a close button (X). It contains an information icon (i) and the text 'Please choose a table!'. Below the text is an 'OK' button. In the background, the input fields are filled with 'Esra', '4', and '20'. The 'Complete reservation' button is visible at the bottom.

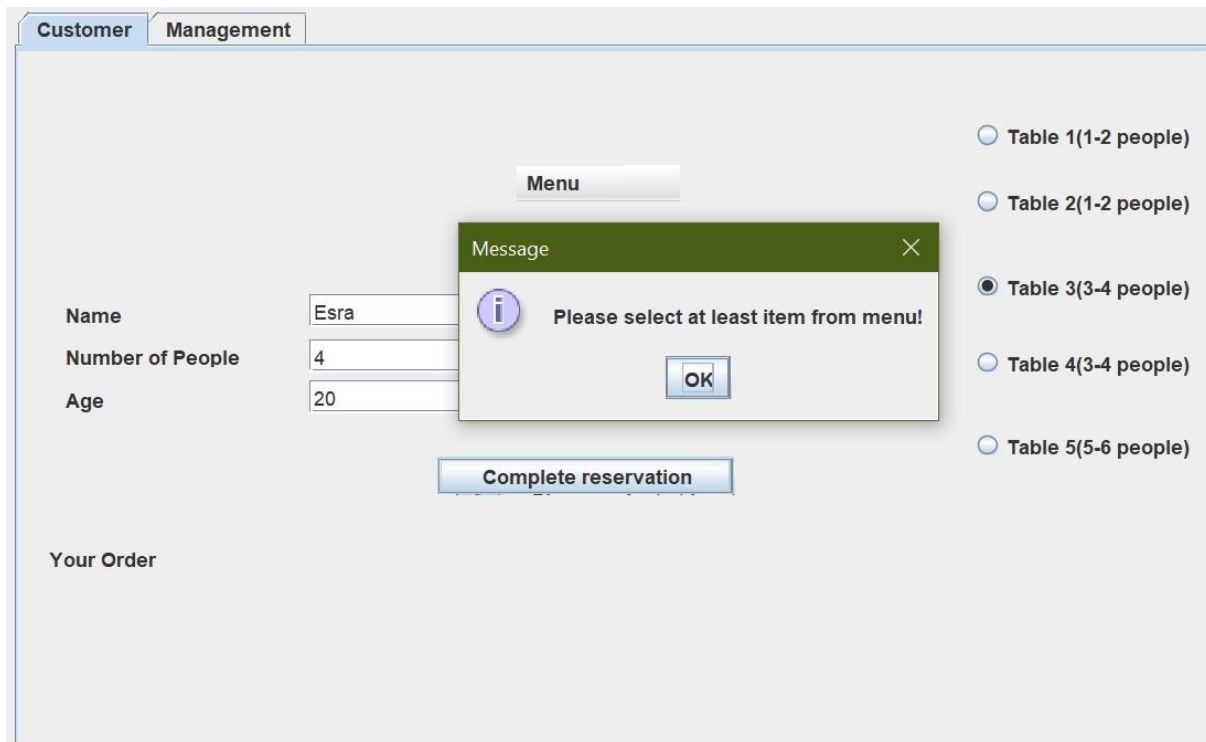
There are 5 tables in the restaurant. In order to select table1 or 2, the customer should enter the number of people as 1 or 2. Similarly, Table 5 only accepts the customers who are 5 or 6 people.

The screenshot shows a web application interface with two tabs: "Customer" and "Management". The "Customer" tab is active. On the left, there are three input fields: "Name" (containing "Esra"), "Number of People" (containing "4"), and "Age" (containing "20"). Below these fields is a "Complete reservation" button. On the right, there are five radio button options for table selection: "Table 1(1-2 people)", "Table 2(1-2 people)" (which is selected), "Table 3(3-4 people)", "Table 4(3-4 people)", and "Table 5(5-6 people)". A message dialog box is overlaid in the center, titled "Message", with an information icon and the text "This table is not suitable for you!". The dialog has an "OK" button.

When user does not provide at least one information;.

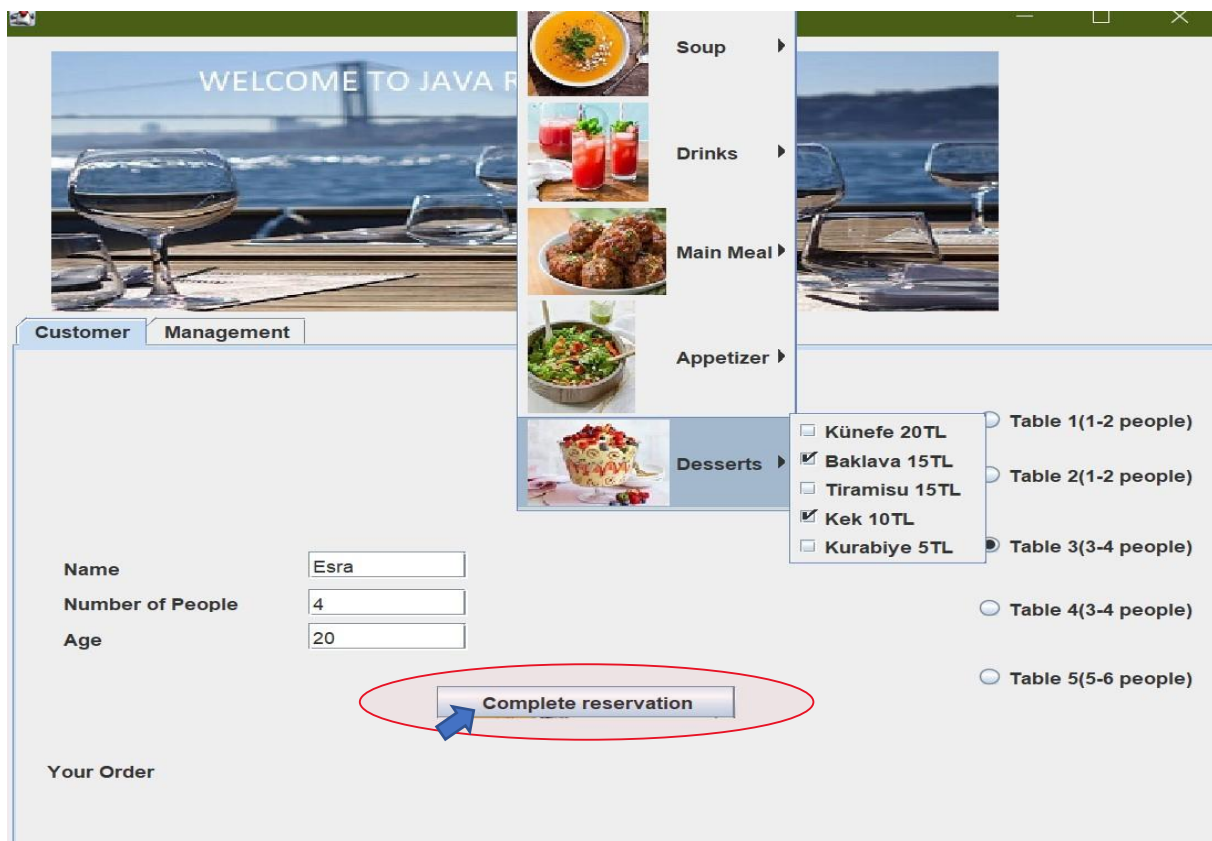
The screenshot shows the same web application interface as the previous one, but with a different state. The "Name" field contains "Esra", the "Number of People" field contains "4", and the "Age" field is empty. The "Complete reservation" button is still present. The table selection options on the right are the same, but now "Table 3(3-4 people)" is selected. A message dialog box is overlaid in the center, titled "Message", with an information icon and the text "Please fill all the blanks!". The dialog has an "OK" button.

After we completed filling the blanks and selecting our table, we need to select item from the menu. If we press the “Complete reservation” without selecting any item we get that error;



Selecting Item

When we click the “Menu”, we check the JCheckBoxMenuItems and then press the “Complete Reservation” button.



As seen below, when there is no stock left, the selections would be gray out and customers cannot select them.

Customer Management

Menu

Soup

Drinks

Main Meal

Appetizer

Table 1(1-2 people)

Table 2(1-2 people)

Table 3(1-2 people)

Table 4(3-4 people)

Table 5(5-6 people)

Grayed Out!

Ezogelin 15TL

Tarhana 15TL

Yayla 15TL

Mercimek 15TL

Tavuksuyu 15TL

Name

Number of People

Age

You chose Table 2. Order: [Mercimek] Total Price: 15 TL

Also, the selected tables would be disabled after the order is completed,

At the bottom, the customer can see their order and total prices.

Customer Management

Appetizer

Desserts

Table 1(1-2 people)

Table 2(1-2 people)

Table 3(3-4 people)

Table 4(3-4 people)

Table 5(5-6 people)

Grayed Out!

Name

Number of People

Age

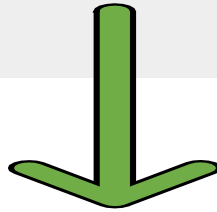
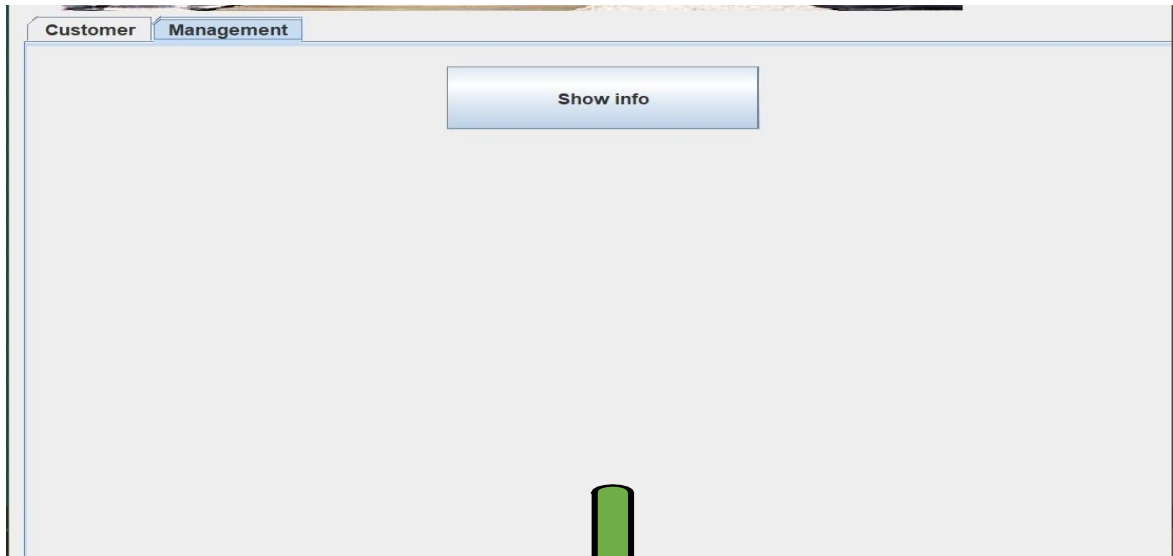
Complete reservation

You chose Table 3. Order: [Baklava, Kek] Total Price: 25 TL

After creation of customer, the blanks and all the selections will be empty in order to register new customer.

Manager Panel

The manager panel would be empty at first, when we click the “Show info” button, there appears a table with orders and also a table with stock information of items.

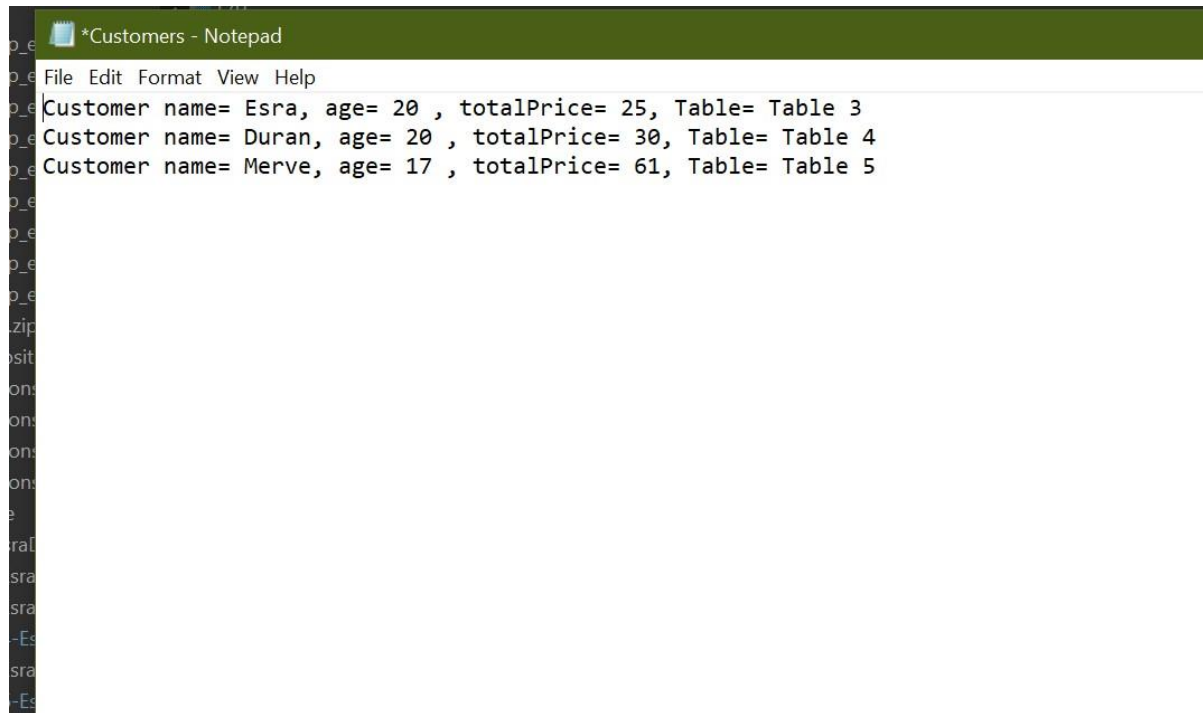


A screenshot of the same web application interface after clicking the 'Show info' button. The 'Show info' button is still present. Below it, two tables are displayed side-by-side. The left table shows customer orders, and the right table shows item stock information.

Age	Name	Total Price	Table
17	Merve	61	Table 5
20	Duran	30	Table 4
20	Esra	25	Table 3

Item	Stock	Ordered
Ezogelin	10	0
Tarhana	4	1
Yayla	8	0
Mercimek	3	1
Tavuksuyu	6	0
Cola	49	1
Water	40	0
Coffee	4	2
Tea	100	0
Ayran	6	0
Meatball	10	0
Kuru Fasulye	19	1
Pasta	10	0
Steak	6	0
Chicken Finger	6	0
Salad	6	0
Sandwich	5	1
Hamburger	30	0
Pizza	6	0
Cheese	100	0

Note: Also, our codes creates a txt file that consists of each orders named "Customers.txt".



```
*Customers - Notepad
File Edit Format View Help
Customer name= Esra, age= 20 , totalPrice= 25, Table= Table 3
Customer name= Duran, age= 20 , totalPrice= 30, Table= Table 4
Customer name= Merve, age= 17 , totalPrice= 61, Table= Table 5
```

PART 2

In this project there exists 3 classes.

Class Customer

```
1 package src;
2
3 public class Customer {
4     public String name;
5     public int age;
6     public int totalPrice;
7     public String table;
8     public Customer(String name, int age, int totalPrice, String table) {
9         this.name = name;
10        this.age = age;
11        this.totalPrice = totalPrice;
12        this.table=table;
13    }
14 }
```

The customer class creates customer objects, however they are not be storage in the memory since each time we are creating the customer with the same name (Number of customers are not known).

We storage that customers information in an ArrayList in the Main class with the help of the Customer methods.

Class MenuItem

```
package src;

import java.util.List;

public class MenuItem {
    public int stock;
    public int price;
    public String name;
    public int decreasedstock;
    public MenuItem(int stock, int price,String name) {

        this.stock = stock;
        this.price = price;
        this.name=name;
        decreasedstock=stock;
    }

    public void decreaseStock() {
```


Each 25 items are objects of MenuItem and this class stores their attributes. There are stock and decreasedstock attributes that we will be using while identifying the number of order of each type of items.

Main Class

In main class, we create our project. First we start with creating objects from MenuItem class.

```
public void Project() throws Exception {  
  
    List<String> orders= new ArrayList<>();  
  
    ArrayList<MenuItem > menuItemList= new ArrayList<>();  
    MenuItem Ezogelin = new MenuItem(10,15,"Ezogelin");  
    menuItemList.add(Ezogelin);  
    MenuItem Tarhana = new MenuItem(5,15,"Tarhana");  
    menuItemList.add(Tarhana);  
    MenuItem Yayla = new MenuItem(8,15,"Yayla");  
    menuItemList.add(Yayla);  
    MenuItem Mercimek= new MenuItem(4,15,"Mercimek");  
    menuItemList.add(Mercimek);  
    MenuItem Tavuksuyu = new MenuItem(6,15,"Tavuksuyu");  
    menuItemList.add(Tavuksuyu);  
}
```

Then, we created panels, menu and menuitems. As well as we inserted some images.

```
setResizable(true);  
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
setBounds(100, 100, 1243, 726);  
contentPane = new JPanel();  
contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));  
setContentPane(contentPane);  
contentPane.setLayout(null);  
  
JTabbedPane tabbedPane = new JTabbedPane(JTabbedPane.TOP);  
tabbedPane.setBounds(10, 203, 812, 486);  
contentPane.add(tabbedPane);
```

```
Image resim=new ImageIcon(Project.class.getResource("/imageRes.jpg")).getImage();  
Image resim1=new ImageIcon(Project.class.getResource("/resim1.jpg")).getImage();
```


We set the labels of CheckBoxMenuItems

```
JCheckBoxMenuItem soup1 = new JCheckBoxMenuItem("Ezogelin 15TL");  
JCheckBoxMenuItem soup2 = new JCheckBoxMenuItem("Tarhana 15TL");  
JCheckBoxMenuItem soup3 = new JCheckBoxMenuItem("Yayla 15TL");  
JCheckBoxMenuItem soup4 = new JCheckBoxMenuItem("Mercimek 15TL");  
JCheckBoxMenuItem soup5 = new JCheckBoxMenuItem("Tavuksuyu 15TL");
```

Then, we add ActionListener to the button "Complete Reservation". We check the conditions and display corresponding warnings.

```
Button.addActionListener(new ActionListener() {  
    @Override  
    public void actionPerformed(ActionEvent e) {  
        ArrayList<String> orderedmeals= new ArrayList<>();  
        int totalprice=0;  
  
        if (bg.getSelection()!=null) {  
            JOptionPane.showMessageDialog(null, "Please choose a table! ");  
        }  
        else if (Nametext.getText().isEmpty() || numtext.getText().isEmpty() || Agetext.getText().isEmpty()) {  
            JOptionPane.showMessageDialog(null, "Please fill all the blanks! ");  
        }  
    }  
}
```

If all necessities are provided, we add the customer into list and set the selected table disabled.

```
if (selected!=0) {  
  
    customerLabel.setText("You chose "+bg.getSelection().getActionCommand()  
        + ". Order: "+orderedmeals.toString() + " Total Price: "+totalprice + " TL"  
        );  
  
    Customer customer =new Customer(Nametext.getText(),Integer.parseInt(Agetext.getText())  
        ,totalprice,bg.getSelection().getActionCommand());  
  
    bg.getSelection().setEnabled(false);  
    bg.clearSelection();  
}
```

We create a txt output file and append the information of new customer.

```
FileWriter fileWriter;  
try {  
    fileWriter = new FileWriter(file, true);  
    BufferedWriter bWriter = new BufferedWriter(fileWriter);  
    bWriter.write(customer.toString()+"\n");  
  
    bWriter.close();  
} catch (IOException e2) {  
    e2.printStackTrace();  
}
```

Then, we create new button named "Show info" in Management panel. We also call the method ActionListener for that button and creates ordered table for customers according to their ages (if same, descending order of total prices). Then, we generate the second table which shows the stocks and the number of ordered each type of foods.

```
orderButton.addActionListener(new ActionListener() {  
    @Override  
    public void actionPerformed(ActionEvent e){  
        Collections.sort(orders);  
  
        for (int i=0;i<orders.size()-1;i++) {  
            System.out.println();  
            if (Integer.parseInt(orders.get(i).split("-")[0])!=Integer.parseInt(orders.get(i+1).split("-")[0]))  
                System.out.println("jer");  
            if (Integer.parseInt(orders.get(i).split("-")[2]) < Integer.parseInt(orders.get(i+1).split("-")[2]))  
                String a=orders.get(i);  
                orders.set(i, orders.get(i+1));  
                orders.set(i+1, a);  
            }  
        }  
  
        String[][] data= new String[orders.size()][4];  
        String[] colNames= {"Age", "Name", "Total Price", "Table"};  
        for(int i=0;i<orders.size();i++) {  
            String[] a=orders.get(i).split("-");  
            data[i]=a;  
        }  
        JTable jTable = new JTable(data,colNames);
```