****

Bilkent University CS 319 Object-Oriented

Software Engineering

Summer 2017

Design Report Project Name: Hotel Reservation System

Group #1

Mehmet Ali Altuntaş 21401004

Damla Eda Bıçakcı 21402130

Murat Süerdem 21401107

İrem Yurdakul 21400299

**Index**

**1.Introduction**

1.1.  Purpose of The System

1.2. Design Goals

1.3 Definitions, Acronyms and Abbreviations

1.4. References

1.5. Overview

**2.Software Architecture**

2.1.Overview

2.2. Subsystem Decompositon

2.3. Hardware/ Software Mapping

2.4. Persistent Data Management

2.5. Access Control and Security

2.6. Boundry Conditions

**3. Subsystem Services**

3.1. Model Subsystems

3.2. User Interface Subsystems

3.3. Control Subsystems

## 1. INTRODUCTION

## 1.1 Purpose of the system

The Hotel Reservation System is a reservation system which aim to offer easy and well design to customer to make reservation and offer detailed information about rooms and reservations to manager to check system. Program is easy to understand, customers make their booking with pleasure and easy way, and manager can Access room details practically with their Access code.

Main advantages of thesystem is choosing features of the room while making reservation. If customer choose just to stay in the room with basic form of it, customer pay less and customer does not have tochoose fixed features, customer can decide which feature is important for him like laundry, breakfastortv.

## 1.2 Design Goals

### Performance:

For the system, it is important to how quickly the system reacts to use r input because this situation can cause to distract customers’ interest andcomfort. Our system can response quickly at each step (1 second), because the main purpose is to make reservation quickly and efficiently.

### Adaptability:

The hotel reservation system is coded with Java. Java provide user to use program with all JRE installed platforms.

### Extensibility:

In software engineering, reusability and extendibility are two importance concepts. Hotel Reservation System suitable to be extended and re-used for future works because separation of controller, view and model classes offer to change or add new features without modifying anything in other classes.

### Usability:

Easy use of program is important functionality because it makes system more charming. Our program provideuserwitheasyandclearinterfacetounderstandandmaketheirbookingswiftly. Especially focus on the plainness of the program, sousers can use program without priorty knowledge.

### Reliability:

The system should not demonstrate with unexpected crashes due to software bugs, so separate tests will be practiced ,and exceptios and boundary conditions will be performed so carefully in order to avoid unconsidered conditions.

## 1.3 Definitions, acronyms, and abbreviations

Abbreviations:

MVC: Model View Controller

2. **SOFTWARE ARCHITECTURE**

## 2.1 Overview

Hotel Reservation system consists of severa lsubsystems. We use MVC structure and our system is decomposed based on this structere type. By dividing the system to subsystems, our goal is to manage the system more easily. And also it enables us code reusability. We can make changes easily.

## 2.2 Subsystem Decomposition

For a good organization, it is essentialtoseperatesytemintosubsystems. Therefore, ur subsystemsarebasicallybased on models, viewsandcontrollers.

## 2.3 Hardware / Software Mapping

Our Hotel Reservation Systemwill be coded in Java programming language and use java run time environment. Mouse and keyboard will be used to interact with the program, users will determine reservation details by using keyboard and mouse. A standard computer with operating system and javacompiler will be needed to run program. Hotel Reservation Program does not require internet connection, it is desktop program. The program will keep data by using .txtbased structures which compose of details of the every separate reservation.

## 2.4 Persistent Data Management

In our system, we use text file to store the data. Our data is about basically customer, manager and reservations. We keep customer’s info which are name, surname and for the manageral so we keeps name, surname and ID for the system enterance.

## 2.5. Access Control and Security

In our system, there is no need to connectany network, our program works on the desktop. Therefore, Hotel Reservation System performs withou tany contro lfor access. However, we keep information about our customers, also managers. Therefore, security is an significant issue to consider it.

## 2.6 BoundaryConditions

**Initialization**

Our program does not require any installization. It can be opned by a simple .jar file.

**Termination**

User can exit the program simply by clicking exit button and in the program there are several buttons to return the backpage.

**Error**

If an error ocur during the information enterence or before thestorage, all information kept by the sytem, can disappered unfortunately.Other than that. some errors happens during the program works, so user must be close the program and he or she can start the program again.

3. SUBSYSTEM SERVICES

3.1. Model Subsystems

**Hotel Class**

**Attributes:**

**private int floors**

This attribute defines the number of floors the hotel has.

**private int numberOfRooms**

This attribute defines the number of rooms in each floor the hotel has

**private Room[][] rooms**

This attribute defines the rooms that the hotel has. The order of the room is such that:

Each line represents a floor and each room object in a line represents a room in that floor.

And the room numbers are composed of floor number and the Ith room in that floor.

For example, the room with room number 503 is in the 5th floor and in the 3rd order in that

Floor and it is represented as rooms[floor-1][order-1], i.e. room[4][2] .

**Constructors:**

**public Hotel(int floors, int numOfRooms)**

This constructor has two parameter : floors and numOfRooms. When this constructor is

Is called, it assigns these two parameter values two Hotel object’s floor and numberOfRooms values. Then it creates new Room[][] double array with given parameters and assign rooms attribute to this new double array.

**Methods:**

**public Room[] findFreeRoom(Date enter, Date out, Room[] searchRoom)**

This method is to find available room(s) between the given enter and out dates. It takes three parameters enter, out and searchRoom. Enter and out are the given Date objects. searchRoom is a room array initialized when the user selects the number of rooms to reserve in User Interface with the size user enters through buttons. The objects in the searchRoom array has capacity values which are also input from the user. Then this method iteratively searches available rooms for the given time interval. If it finds available room(s) then it creates a new array from that room(s) and returns it. Else it returns NULL.

**public boolean availability (Room[] rooms)**

This method is to check whether the Room array returned from findFreeRoom is empty or not.

**public String noAvailableRoomWarning()**

This method is called when the availability method returns false. It gives an error message specifying that there is no available room(s) between the given time interval.

**public Room[] showAvailableRoom(Room[] rooms)**

This method is called when the availability method returns true. It returns the Room[] array given as parameter.

**public Reservation createReservation(Date enter, Date out, Room[] rooms)**

This method is created when the user sees the available rooms and continues to make booking. It creates a temporary reservation with the given time interval and the available Room array. This method is called to make changes on an imaginary made reservation.

**public Reservation selectRoomFeatures(Reservation r, room aRoom, int[] features)**

This method is called when the user selects features that he wants in the rooms he choosed. It calls the setSelections() method of the room and updates the features and the price. Then, it returns the reservation r given as parameters as modified.

**public String showReservationDetails(Reservation r)**

This method is to show the details of a given reservation with all attributes which are not NULL or empty.

**public Reservation makeBooking(String name, String last, int id, int phoneNumber, Reservation r)**

This method is called when the user clicks on “Make Booking” button on the reservation details page when making booking. It updates the given reservation r with the given name, last, id and phoneNumber parameters and adds the reservation r to the reservation lists in of the rooms that the reservation r has. Then it returns the modified reservation r.

**public Reservation makePayment( String name, long cardNumber, int month, int year, int cvc, Reservation r)**

This method is called after the user clicks on “Make Payment” button on the reservation details page when making booking. Before this method, makeBooking() method is called and reservation r is updated. Then this method is called to take credit card information and make payment. According to the result of payment it updates the isPayed attribute of the reservation r and returns the reservation r.

**public boolean verifyPayment(Reservation r)**

This method called after the makePayment() method. It is to check whether the given reservation r’s fee is payed or not.

**public String paymentNotAcceptedWarning()**

This method is called if the verifyPayment() method returns false. It gives an error message specifying that the payment is not accepted.

**public Reservation findReservation(String code)**

This method is called when the user wants to find a reservation made before. It takes a String parameter code and it iteratively searches the rooms in the hotel class for whether there is reservation with the given code or not. If there is an existing reservation with the given code the method returns it, else it returns NULL.

**public boolean reservationExists(Reservation r)**

This method is to check whether the Reservation returned from the findReservation is existing or not. If the return value of the findReservation is NULL, this method returns false. Else it returns true.

**public String noExistingReservationWarning()**

This method is called is the return value of reservationExists() is false. It gives an error specifying that there is no reservation with the given code.

**public boolean removeReservation(Reservation r)**

This method is to remove a given reservation from the reservation lists of the rooms that the parameter reservation r has. It returns boolean value representing the condition of remove.

**public String removedVerificationMessage()**

This method is called after removeReservation() method. It gives a message specifying that the reservation is removed.

**Room Class**

**Attributes:**

**private int roomNumber**

This attribute keeps the room number of the room.

**private int roomCapacity**

This attribute keeps the capacity of the room.

**private int defaultPrice**

This attribute keeps the default price value, input from the receptionist, of the room.

**private ReservationNode reservations**

This attribute keeps the reservation list of the room.

**Constructor:**

**public Room(int roomNo, int capacity, int price)**

This constructor takes three inputs roomNo, capacity and price. Then it assigns these values to Room object’s roomNumber, capacity and defaultPrice attributes. Then it creates an empty reservation list.

**Methods:**

**public Reservation setSelections(int[] features, Reservation r)**

This method is called after the selectRoomFeatures() method of the hotel. This method sets the features of the room kept in the reservation r, to the given features parameter values. Then it returns the modified parameter value reservation r.

**public boolean removeReservation(String code)**

This method removes the reservation with the given reservation code. If the remove is successful it returns true, else it returns false.

**public Reservation addReservation(String name, String last, int id, int phoneNumber, Reservation r)**

This method is called after makeBooking() method of the hotel. It updates the given parameter reservation r with other given parameters. Then it returns the modified reservation r.

**public ReservationNode getReservations()**

This method returns the reservation list of the room.

**public int getRoomNo()**

This method returns the room number.

**public int getCapacity()**

This method returns the capacity of the room.

**public int getDefaultPrice()**

This method returns the default price of the room.

**public void setDefaultPrice(int price)**

This method sets the default price to the given parameter price.

**ReservationNode Class**

**Attributes:**

**public Reservation item**

This attribute keeps the Reservation item.

**public ReservationNode next**

This attribute keeps another ReservationNode item.

**Constructors:**

**ReservationNode()**

This method creates a ReservationNodewith NULL values.

**Reservation Class**

**Attributes:**

**private Date enter**

This attribute keeps enter date of the customer to the hotel.

**private Date out**

This attribute keeps leaving date of the customer from the hotel.

**private Room[] rooms**

This attribute keeps the rooms reserved by this reservation.

**private Customer c**

This attribute keeps the customer object who made this reservation.

**private String code**

This attribute keeps the reservation’s code.

**private boolean isPayed**

This attribute keeps the boolean value whether the reservation’s fee is payed or not.

**private String[][] roomFeatures**

This attribute keeps the rooms’ features that the rooms reserved by this reservation has. First line of this attribute keeps the room number in String forms. First column of this attribute keeps the features names in String forms. Last line of this attribute keeps the prices of the modified rooms in string forms. Other values keeps “true”/”false” strings representing that whether the feature is included to the room or not.

**Constructor:**

**public Reservation()**

This constructor creates a Reservation with NULL/empty values.

**Methods:**

**public void createReservation(Date enter, Date out, Room[] rooms)**

This method is called after the showAvailableRooms() method of the hotel. It creates a temporary reservation with the given time intervals and given Room[] array. This method is to keep the room features without making a real change in the rooms of the hotel.

**public boolean updateReservation(Room aRoom, int[] features)**

This method is called after the setSelections() method of the room class. It takes two parameters Room object and an integer array. Room object is the desired room to modify and the integer array keeps 0 and 1 values representing that whether the feature is included or not. It updates the roomFeatures attribute of the reservation and return true if successful.

**public void verifyPayment(boolean payed)**

This method is called after the makePayment() method of hotel. It is to set the isPayed value to the given parameter payed.

**public void formCode()**

This method is to form a reservation code from the attributes of this class.

**public Date getEnterDate()**

This method returns the enter date.

**public Date getOutDate()**

This method returns the leaving date.

**public Customer getCustomer()**

This method returns the customer who made this reservation.

**public String getCode()**

This method returns the reservation code of this reservation.

**public int getTotalPrice()**

This method calculates the total price according to the roomFeatures attribute and returns it.

**public String showDetails()**

This method returns a String value describing the details of the reservation.

**public boolean getPayed()**

This method returns the isPayed value of the reservation.

**public void setCustomer(Customer c)**

This method sets the Customer attribute of this class to the given Customer c parameter.

**User Class**

**Attributes:**

**private String name**

This attribute keeps the name of the user.

**private String lastName**

This attribute keeps the last name of the user.

**private long id**

This attribute keeps the id of the user.

**Constructors:**

**public User(String name, String lastName, long id)**

This constructor takes three values and assigns them into the user object’s attributes.

**Methods:**

**public void setName(String name)**

This method sets the name attribute of the user object to the given parameter name.

**public void setLastName(String lastName)**

This method sets the lastName attribute of the user object to the given parameter lastName.

**public void setId(long id)**

This method sets the id attribute of the user object to the given parameter id.

**public String getName()**

This method returns the name attribute of the user object.

**public String getLastName()**

This method returns the lastName attribute of the user object.

**public long getId()**

This method returns the id attribute of the user object.

**Receptionist Class implements User Class**

**Attributes:**

**private String password**

This attribute keeps the password of the Receptionist object.

**Constructors:**

**public Receptionist(String name, String lastName, long id, String password)**

This constructor creates a Receptionist object with the given four parameters. It assigns the password parameter to the password attribute and then calls the super’s constructor with the given other parameters.

**Methods:**

**public void setPassword(String pw)**

This method sets the password attribute to the given String parameter pw.

**public String getPassword()**

This method returns the password attribute of Receptionist.

**public boolean checkPassword(String pw)**

This method checks whether the given String parameter pw is equal with the password attribute of Receptionist and returns ture if they are equal. Else it returns false.

**Customer Class implements User Class**

**Attributes:**

**private long phoneNumber**

This attribute keeps the contacting phone number of the Customer.

**Constructors:**

**public Customer(String name, String lastName, long id, long phoneNumber)**

This constructor creates a Customer object with the given four parameters. It assigns the phoneNumber parameter to the phoneNumber attribute and then calls the super’s constructor with the other given parameters.

**Methods:**

**public void setPhoneNumber(long phoneNumber)**

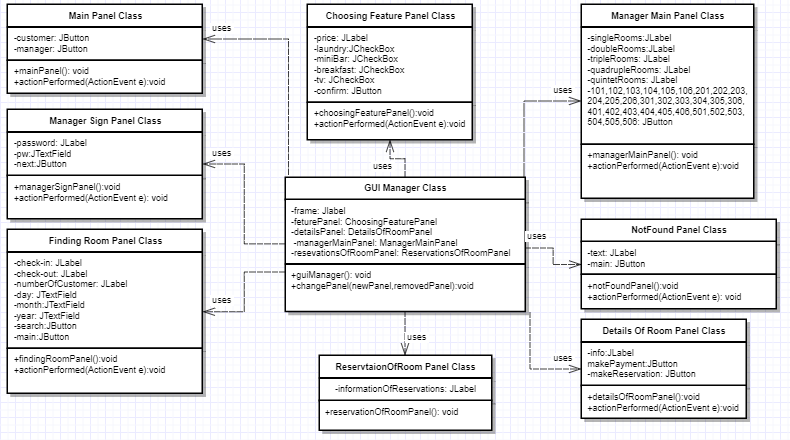
This method sets the phoneNumber attribute of the Customer object to the given phoneNumber parameter.

**public long getPhoneNumber()**

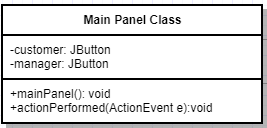
This method returns the phoneNumber attribute of Customer object.

3.2. User Interface Subsystems

User interface subsystem provides our software system with graphical system components.



Main Panel Class



Attributes:

**private JButton customer**:When the user first enter the system by the help of this button user can indicate that he is a customer.

**private JButton manager:** When the user first enter the system by the help of this button user can indicate that he is a manager.

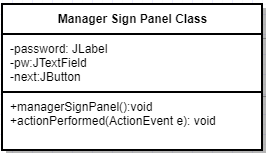
Constructor:

**public mainPanel():** This method initializes the first interface sttaing two buttons in the system which indicates the user is a customer or a manager.

Methods:

**public void actionPerformed(ActionEvent e):** This method determines what is going to happen when customer selects “customer” button or “manager” button.

Manager Sign Panel Class



Attributes:

**private JLabel password:** This label shows just a text that in the next JTextField user should enter his password.

**private JTextField pw:** This text field helps manager to enter his password to take action in the system.

**private JButton next**: This button evaluate the trueness of the password, and if its true pass the manager to the next page.

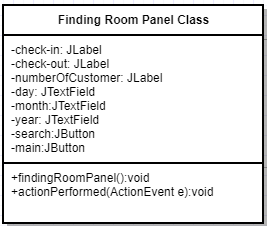
Constructor:

public managerSignPanel(): This panel shows the password text place and the informative text inside.

Methods:

public void actionPerformed(ActionEvent e): This method determines what is going to happen when customer selects “next” button after writiring the password correctly.

Finding Room Panel Class



Attributes:

**private JLabel checkin:** This label shows just a text that in the next JTextField user should enter his check-in day, month and year.

**private JLabel checkout:** This label shows just a text that in the next JTextField user should enter his check-out day, month and year.

**private JLabel numberOfCustomer:** This label shows just a text that in the next JTextField user should enter his number of customer that is going to accommodate in the room.

**private JTextField day:** This text field offer to the user to sekect the day that isgoing to check-in/out in the hotel.

**private JTextField month:** This text field offer to the user to select the month that isgoing to check-in/out in the hotel.

**private JTextField year:** This text field offer to the user to select the year that isgoing to check-in/out in the hotel.

**private JButton search:** When the user complete all the check-in/out and number of customer information this button starts the search process.

**private JButton main:**When the user complete or does not complete the seacrh criterias he can turn back to main menu by the help of this buton.

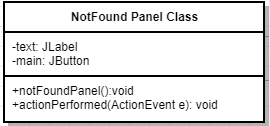
Consturctor:

**public findingRoomPanel():** This panel shows the criterias and the selection options to the user to ask.

Methods:

**public void actionPerformed(ActionEvent e):** This method determines what is going to happen when customer selects “next” button after writiring the password correctly.

NotFound Panel Class



Attributes:

**private JLabel text**: This label includes necessary information about the the search process.

**private JButton main:** This button helps user the turn bak to main menu after the search process competed unsuccesfully.

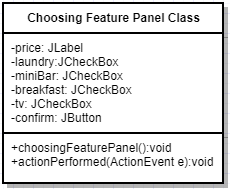
**Consturctor:**

**public notFoundPanel():** This panel holds the information in the text to inform the user and holds a button insace the user wants to turn back to menu.

Methods:

**public void actionPerformed(ActionEvent e):** This method determines what is going to happen when customer selects “main” buttonthe seacrh process completed unsuccesfully.

### ChoosingFeturePanel Class



#### Attributes:

**private JLabel price:** This label shows the specified room price and while customer is determining features of the room, price change.

**private JCheckBox laundry :** This check box is to get the user input to determine feature of the room whether there is laundry service for customer.

**private JCheckBox mini bar :** This check box is to get the user input to determine feature of the room whether there will be tv in the room or not.

**private JCheckBox breakfast :** This check box is to get the user input to determine feature of the room whether hotel will offer breakfast for the customer or not.

**private JCheckBox tv :** This check box is to get the user input to determine feature of the room whether there will be tv in the room or not.

**private JButton confirm:** This button confirms features and provide customer to pass the other page.

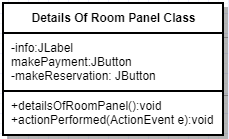
#### Constructors:

**public choosingFeaturePanel:**This is to initialize and add attributes to panel.

#### Methods:

**public void actionPerformed(ActionEvent e):**  This method determine what will happen when customer selects “confirm” button.

### DetailsOfRoomPanel Class



#### Attributes:

**private JLabel ınfo:** This label shows the specified room features before make reservation.

**private JButton makePayment:** If customer wants to make payment of the reservation, customer selects this button, his or her reservation is registered and payment page is opened.

**private JButton makeReservation:** If customer wants to make reservation without making payment of the reservation, customer selects this button, his or her reservation is registered and system shows the reservation code.

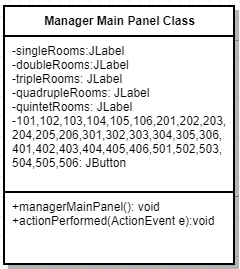
#### Constructors:

**public detailsOfRoomPanel:**This is to initialize and add attributes to panel.

#### Methods:

**public void actionPerformed(ActionEvent e):**  This method determine which button is selected and what will happen.

### ManagerMainPanel Class



#### Attributes:

**private JLabel 1st Floor(single rooms) :** This label give information about floor of the room to make clear interface.

**private JLabel 2nd Floor(double rooms) :** This label give information about floor of the room to make clear interface.

**private JLabel 3rd Floor(triple rooms) :** This label give information about floor of the room to make clear interface.

**private JLabel 4th Floor(quadruple rooms) :** This label give information about floor of the room to make clear interface.

**private JLabel 5th Floor(quintet rooms) :** This label give information about floor of the room to make clear interface.

**private JButton 101,102,103,104,105,106,201,202,203,204,205,206,301,302,303,304,305,306,401,402,403,404,405,406,501,502,503,504,505,506 :** If manager selects this button, the other page is opened where all reservations of the specified room with its all details are showed..

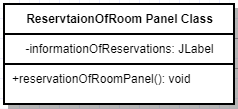
#### Constructors:

**public managerMainPanel :**This is to initialize and add attributes to panel.

#### Methods:

**public void actionPerformed(ActionEvent e):**  This method determine which button is selected and what will happen.

### ReservationsOfRoomPanel Class



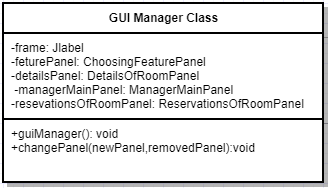
#### Attributes:

**private JLabel informationOfReservations:** This label shows all reservations with its all details which are belong to specified room.

#### Constructors:

**public reservationOfRoomPanel:**This is to initialize and add attributes to panel.

### GUİManager Class



#### Attributes:

**private JFrame frame:** This is the main frame where we display specified panel

**private ChoosingFeaturePanel featurePanel:** This JPanel is to show choosing feature menu on the screen .

**private DetailsOfRoomPanel detailsPanel:** This JPanel is to show details of specified room details menu on the screen .

**private ManagerMainPanel managerMainPanel:** This JPanel is to show rooms menu on the screen .

**private ReservationsOfRoomPanel reservationsOfRoomPanel:** This JPanel is to show rooms’ reservations on the screen .

#### Constructors:

**public guiManager:**This is to initialize and add attributes to panel.

#### Methods:

**public void changePanel(newPanel,removedPanel):** It changes the panel on frame according to commands of controllers.