IS096-1 – Principles of Programming and Data Structures 2021-2022

Assignment 2 – Hotel Online Customer Booking and Management System

INDIVIDUAL WORK

Student ID: 2146504

Table of Contents

Introduction/Overview

The principles and structure of the Java programming language are required for the Hotel Luton Project. To accomplish the project, information from many sources must be gathered and used to develop, create, and execute a fully functional desktop application.

The Luton Hotel provided the data, scenario, and instructions necessary to construct a fully functional desktop application that allowed customers to reserve a room online using a graphical user interface (GUI). Keeping the registration and payment information of each Guest. The major target is to provide the hotel with a complete online booking service.

Task Description

Hotel Luton is getting ready to introduce a reservation and vacation planning service using an internet desktop application.
Visitors may use the app to reserve a hotel room, change their booking details, and cancel their reservation if the service is not used. Guests must first register on the Luton desktop application in order to reserve a room. Following registration, visitors can select their preferred accommodation type (i.e., single room, twin room, or double room). Each room has its own primary amenities, which are included in the reservation.
The reservation procedure involves a registration form that must be filled out with correct guest information, and visitors must enter their credit card information to ensure the booking. Due to the inability of guests to pay online, their credit card information is only used to validate their registration.

Luton provides auxiliary services such as a bar for refreshment, hotel room service, a swimming pool, a spa, and so on. Because this is an external service, it can be added to room service from the receptionist for an additional fee.

Guests are not expected to pay immediately after each service received. All bills are included to the room charge and can be paid entirely at the time of check-out.

Assumptions

- Admin data are already stored in database so Receptionist never register to the system but have to login in order to reach to receptionist page.
- Without credit card details Receptionist cannot accept the booking of customer.
- The non corporate user are not able to get discount where as corporate will get discount as given by receptionist.
- Bill are not generated until receptionist create bill of booking ID with discount percent if available.
- The payment date for customer is the booking check out date and for non customer payment date is one month after of booking check out date.

Project Plan/Schedule

Week No.	Tasks	Priority
1	Conduct research for Introduction/Overview section and assignment description.	MUST
2	Map the functional, technological, and usability requirements.	MUST
3	Working on project design (database design and application prototype)	MUST

4	Will work with MySQL database management system.	MUST
6	Work on GUI apps using Java programming.	MUST
7	Work on the logical programming section.	MUST
8	Submit Group Report, Project Code, and Individual Reflective Report, Video Recording	MUST
9	Project Presentation	MUST

Overview of Functional, Technical (Non-Functional Requirements) and Usability Requirements

Req. No	Requirement	Priority*
1	The main page containing hotel information must be visible to guest users.	MUST
2	Guest users should not be allowed to reserve rooms without Logging In.	MUST
3	Place the registration and login pages next to one other for quick access.	MUST
4	A profile page should be displayed after logging in.	SHOULD
5	Guests should be able to browse all of the rooms and the service galleries without logging in.	SHOULD
6	During the booking process, a reservation form should be produced for all essential information to be filled out.	MUST

7	A credit card information form should be established in addition to the reservation form.	MUST
8	A user's upcoming reservation data must be generated by their user profile.	SHOULD
9	All reservation data must be preserved as history in the user profile after check out.	COULD

Non-functional Requirements

Req. No	Requirement	Priority*
1	The desktop application process time should be quick.	MUST
2	The desktop program must be device-screen adaptive.	SHOULD
3	The application must be adaptable enough to accommodate future changes.	MUST

Usability Requirements

Req. No	Requirement	Priority*
1	The desktop application should be designed with the user in mind.	MUST
3	Forms and reservations should be minimal and simple to complete.	MUST
4	Application must be simple and straightforward interface.	SHOULD

Design

Uses Case Diagrams(s)

(According to Lucid Software Inc 2022)

In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. To build one, you'll use a set of specialized symbols and connectors. An effective use case diagram can help your team discuss and represent:

- Scenarios in which your system or application interacts with people, organizations, or external systems
- Goals that your system or application helps those entities (known as actors) achieve
- The scope of your system

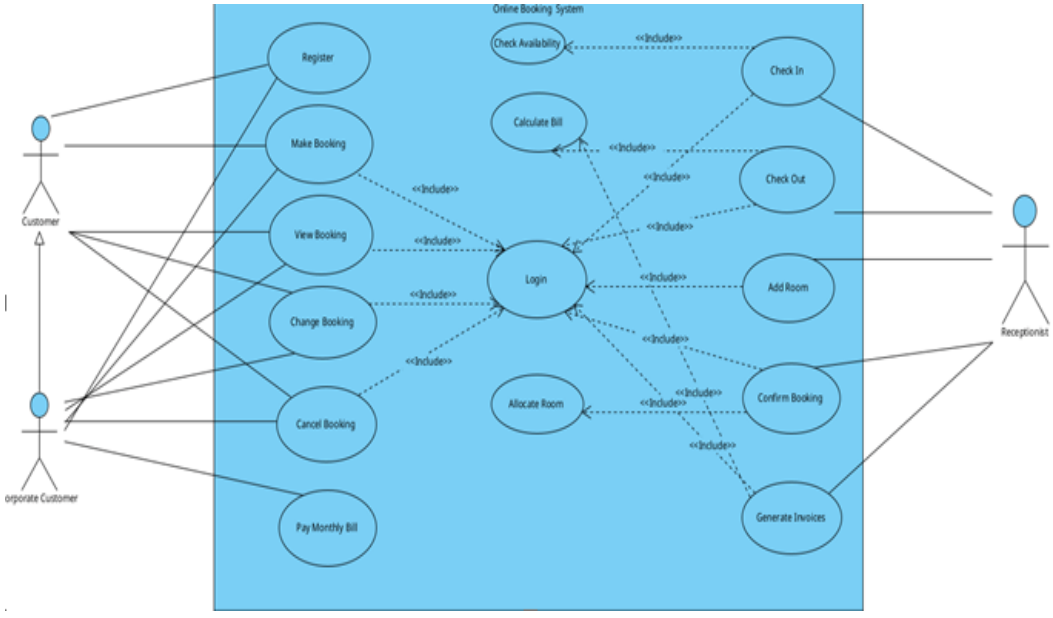


Fig 1: Hotel Online Customer Booking and Management System - Use Case Diagram (Sea Level)

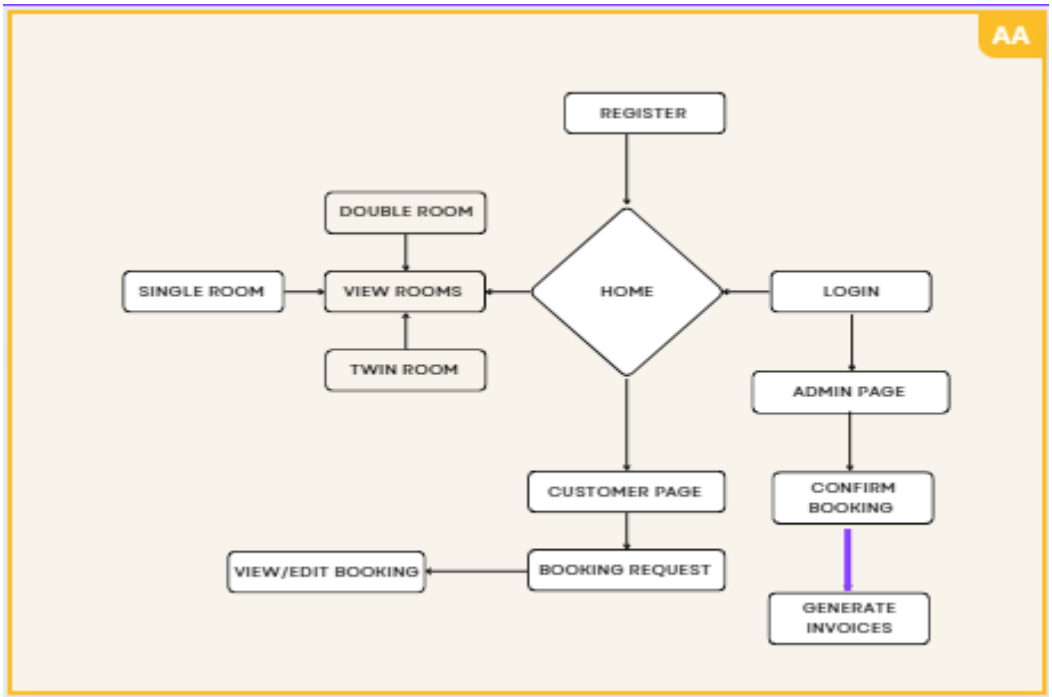
Use Case Specifications/Description

The Non-Corporate Customer, Corporate Customer, and Receptionist user categories are shown in the use case diagram below. In order to use the program, each user must first log in. Both corporate and non corporate customers may make, view, and cancel reservations after logging in. Additionally, receptionists may create invoices for the reservations, add rooms, authorize bookings, and allocate rooms to bookings. The receptionist must verify both the credit card information and the room's availability before allocating a booking to a room.

Activity Diagram(s)

(According to Visual Paradigm 2022)

Activity diagram is another important behavioral diagram in UML diagram to describe dynamic aspects of the system. Activity diagram is essentially an advanced version of flow chart that modeling the flow from one activity to another activity.



In above activity diagram the flow of activity is shown where After register from the home page receptionist can log in to system , go to admin page , confirm booking and generate invoice.

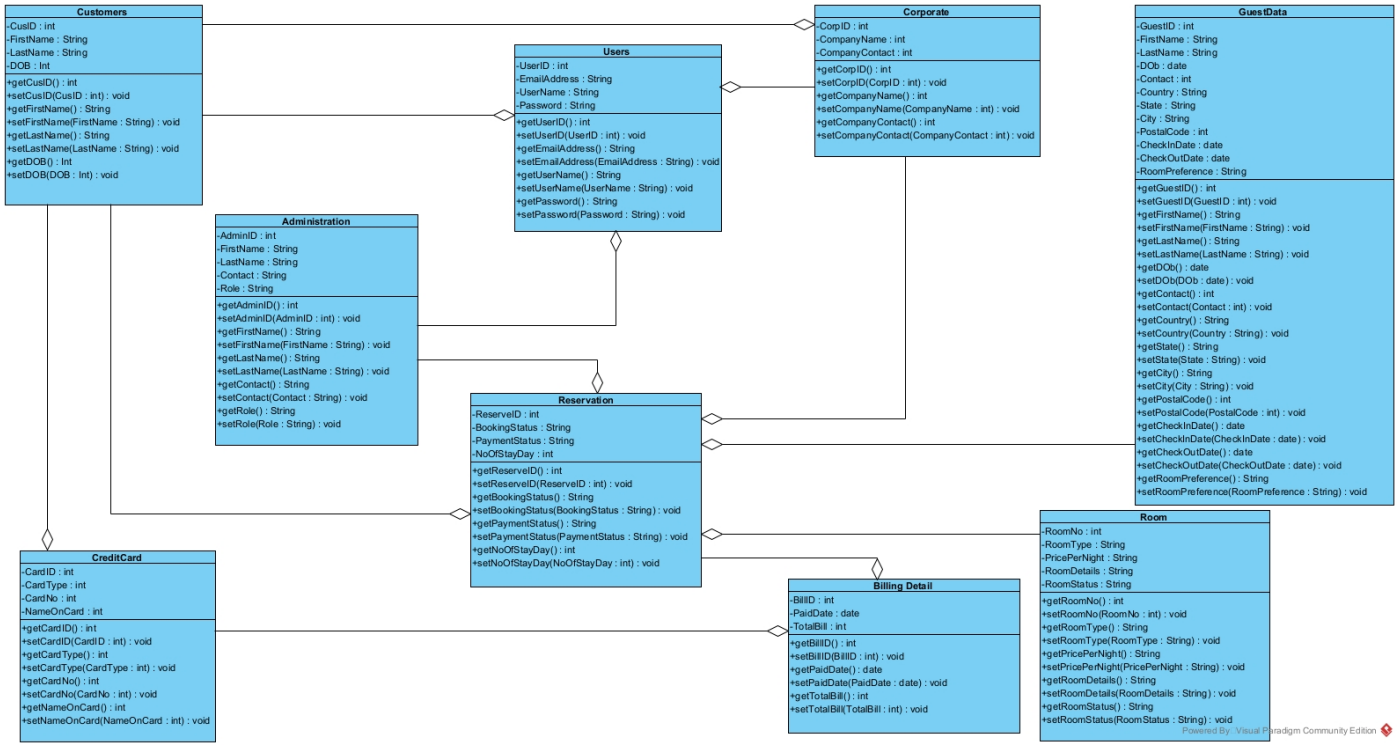
Logging in as customer allow the user to place booking as well as view and edit the booking.

But Without logging in user can view the types of rooms and view more details about the room.

Class Diagram(s)

(According to toutorials point 2022)

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

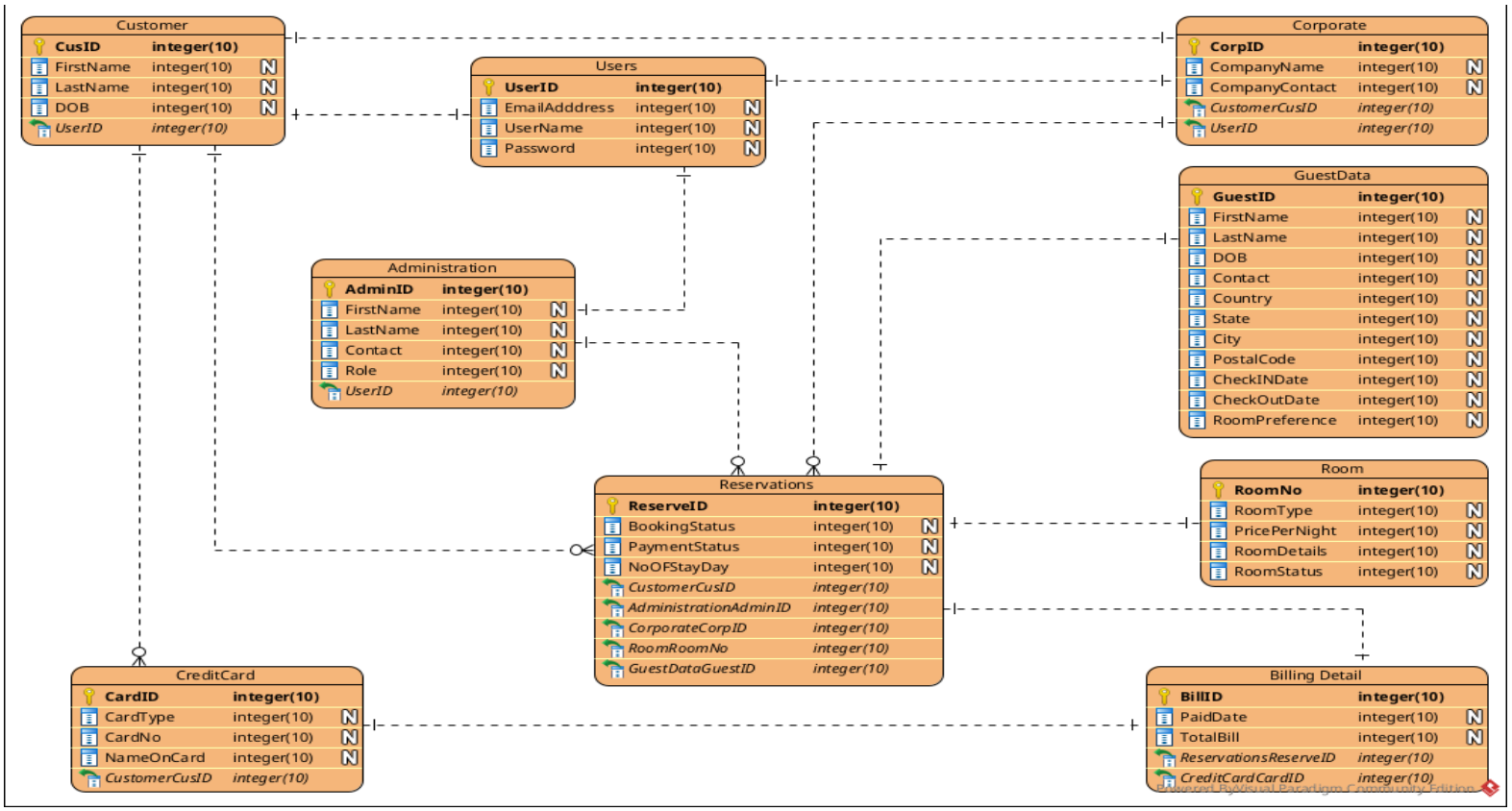


Database Design

Entity Relationship Model (ERM)

- ERM Diagram

Researched From tutorialspoint (2022)



Above ERM diagram shows the relation bettween the entities used in Luton Hotel Application database . Where nine Entities are created. (Users, Customers, Corporate, Administration, GuestData, Room, Reservations, CreditCard, BillingDetails)

Where Users contains all the attributes related to login details. Customer contain details related to the customer information along with the login details so there is one to one relation with Users where all the Users attributes are inheritanced by Customer. Corporate entity contain details about the company the customer is related to so creating one to one relationship with Users and customer where corporate accpet all the attrebutes from Customer and Users Table. Same goes for Administraton It create one to one relation ship between Users entity.

Before reservation GuestData form is completed by the Customer . So there is one to one relation between GuestData and Reservation where reservation holds all the data from the GuestData.

The room that are assigned to the booking ID create a one to one relationship between Room and Reservation Entity where reservation holds all the data from the rooms that are assigned to the booking ID.

Reservation also create ne to many relation with the Customer, Corporate and Administration as Customer and Corporate user can place there reservation and Administration can approve or cancel the booking from the reservation.

To confrim the booking Customer must enter there credit card details so Creating a one to one relationship betwwen the Customer and Creditcard Entity the Credit card holds the information of Customer .

For the Billing, Reservation Entity and CreditCard Entity create a One to One relation with the Billing where billing holds all the information from the reservation and the Customer in order to create the bill.

Physical Database Design

Skeleton Tables

Users(**UserID,** EmailAddress, UserName, Password)

Customer(**CusID,** FirstName, LastName, DOB, UserID*)

Corporate(**CorpID,** CompanyName, CompanyContact, CustomerID*, UserID*)

Administration(**AdminID,** FirstName, LastName, Contact, UserType, UserID*)

CreditCard(**CardID,** CardType, CardNo, NameOn Card, CustomerCusID*)

GuestData(**GuestID,** FullName, DOB, Contact, Country, State, City, CheckInDate, CheckOutDate, RoomPreference)

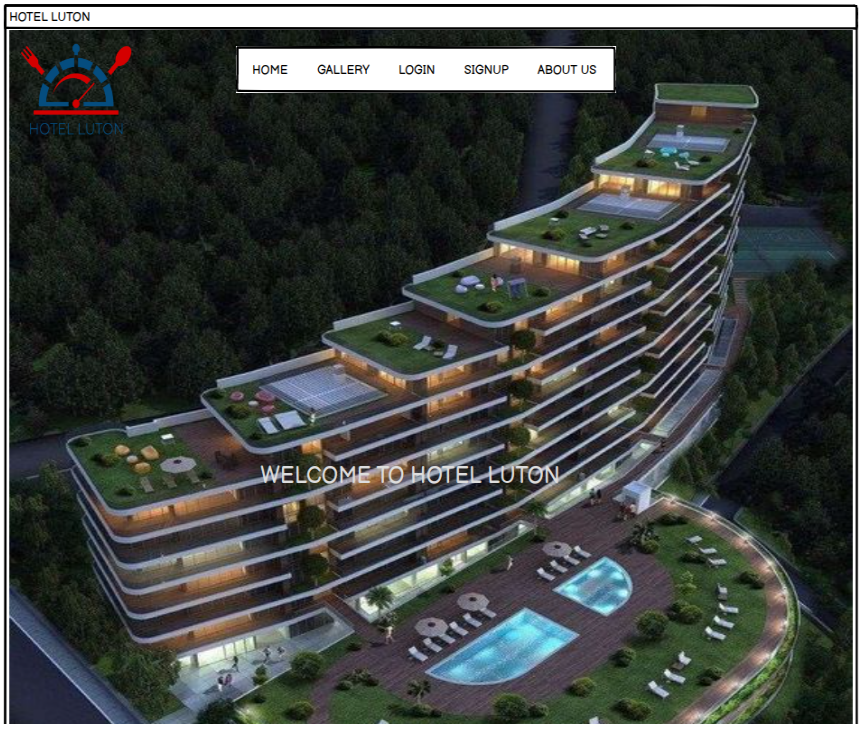
Room(**RoomNo,** RoomType, PricePerNight, RoomDetails, RoomStatus)

Reservations(**ReserveID,** BookingStatus, PaymentStatus, NoOfStayDay, CustomerCusID*, AdminID*, CorpID*, RoomNo*, GuestID*)

BillingDetails(**BillID,** PaidDate, TotalBill, ReserveID*, CardId*)

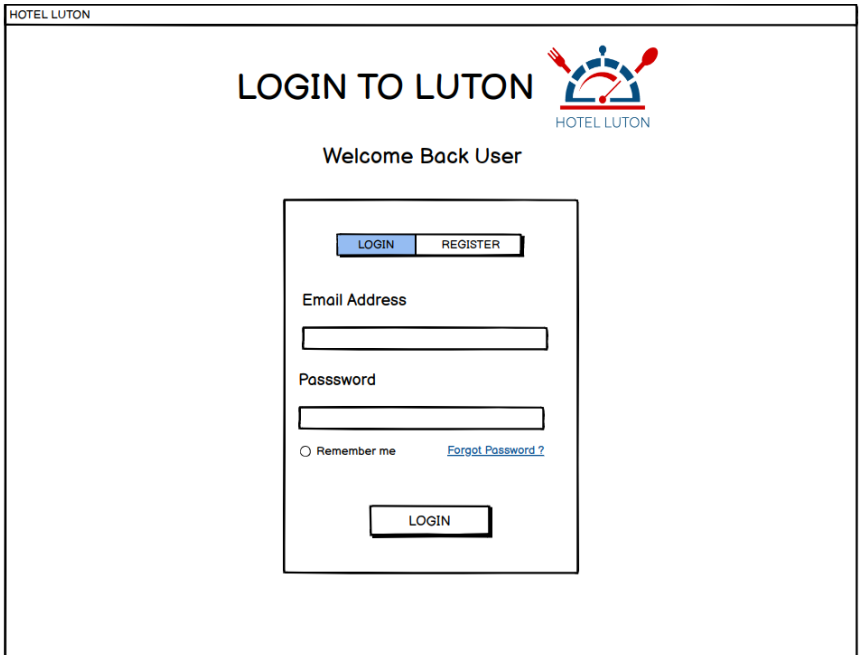
User Interface Design

Luton Welcome Page

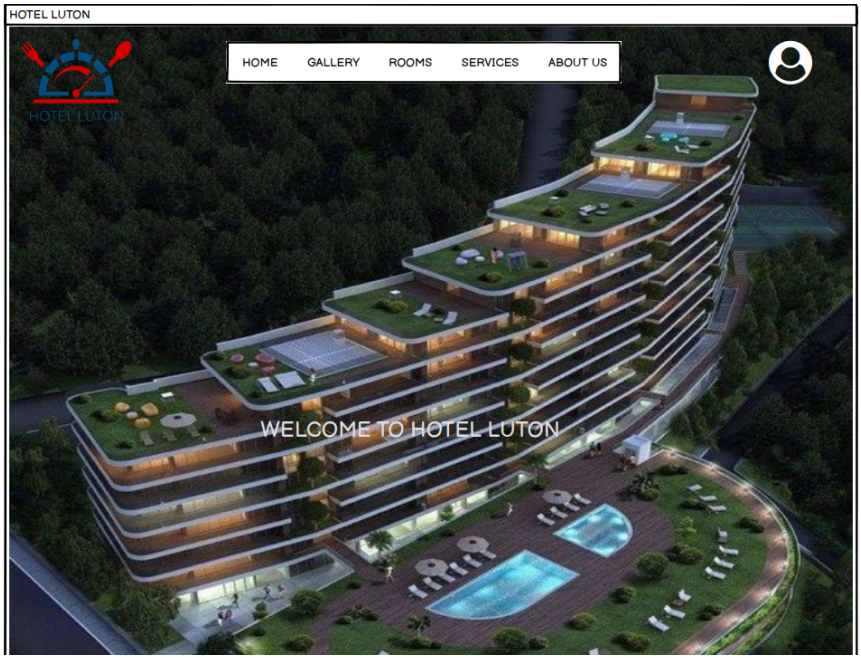


SignUp Page

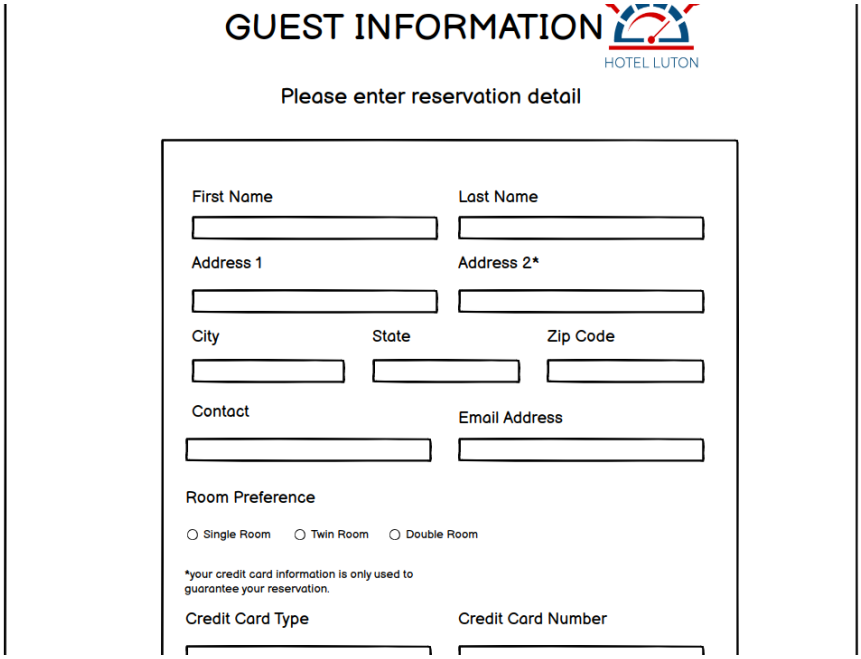
LogIn Page



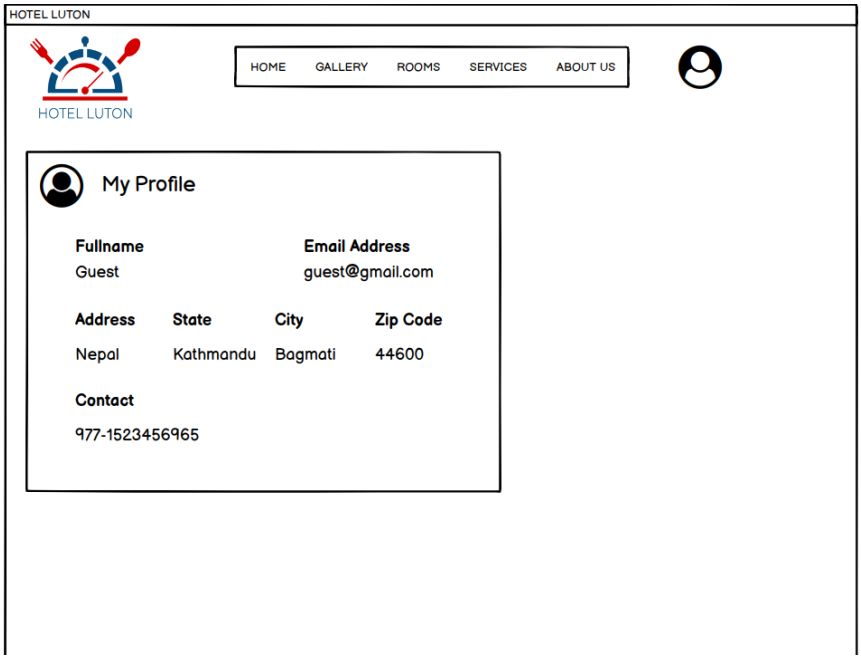
User Home Page



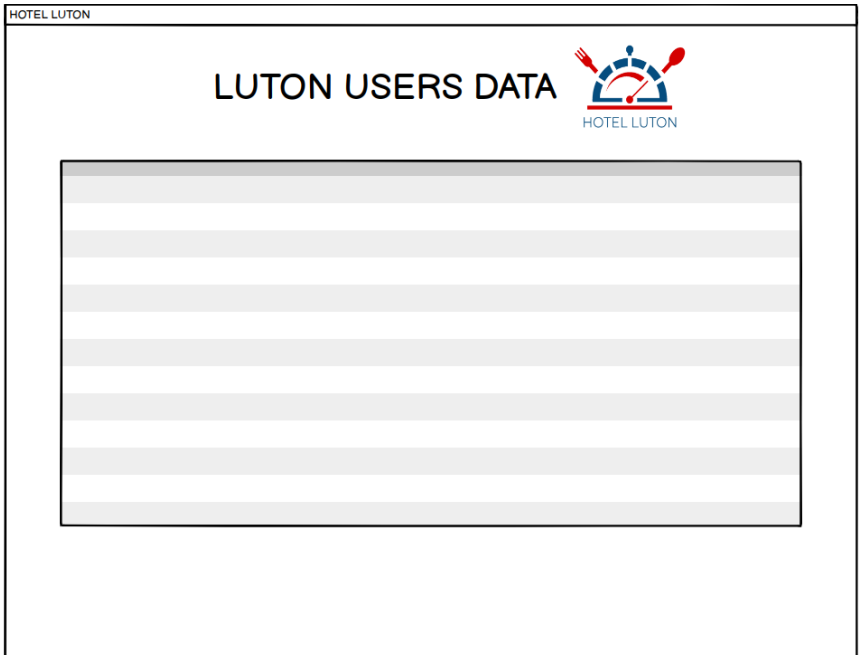
Booking Page With Guest Form



User Profile Page

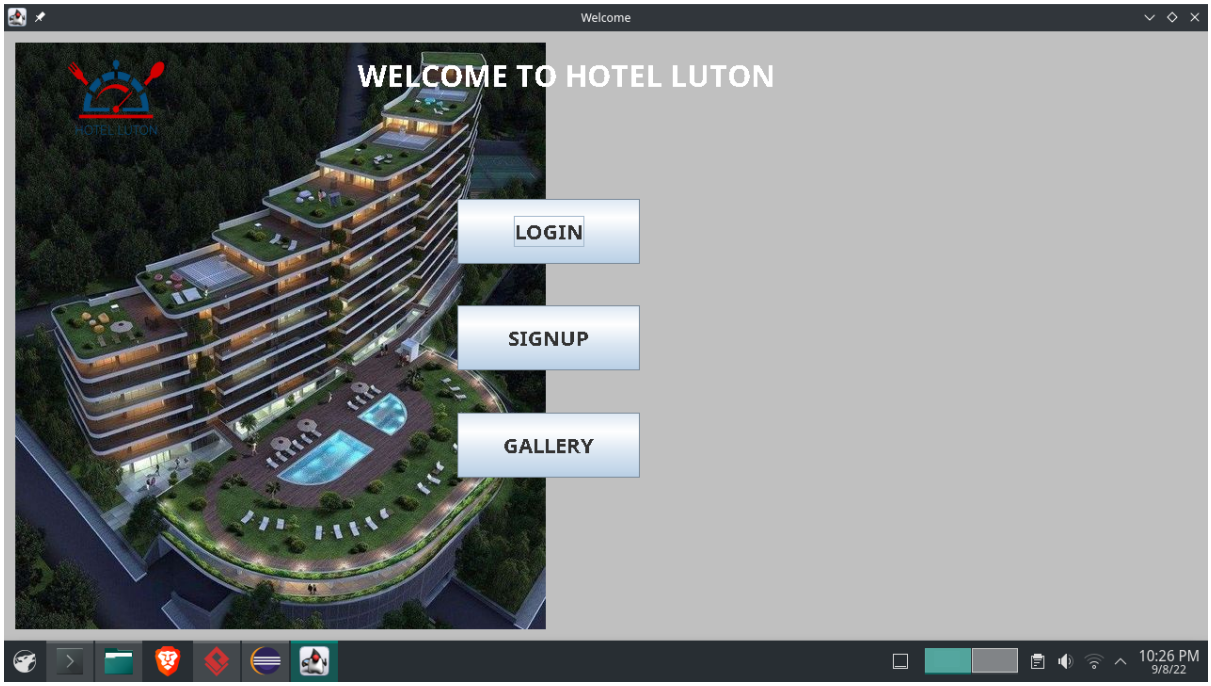


Receptionist Page



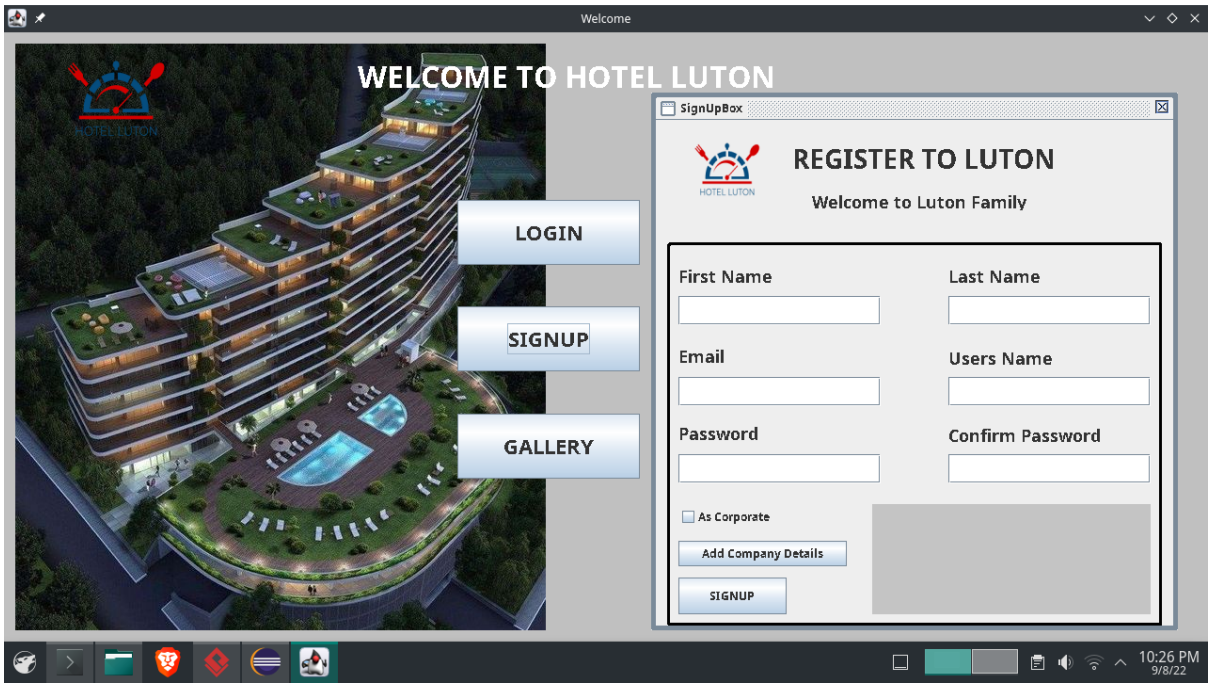
Implementation

Luton Welcome Page



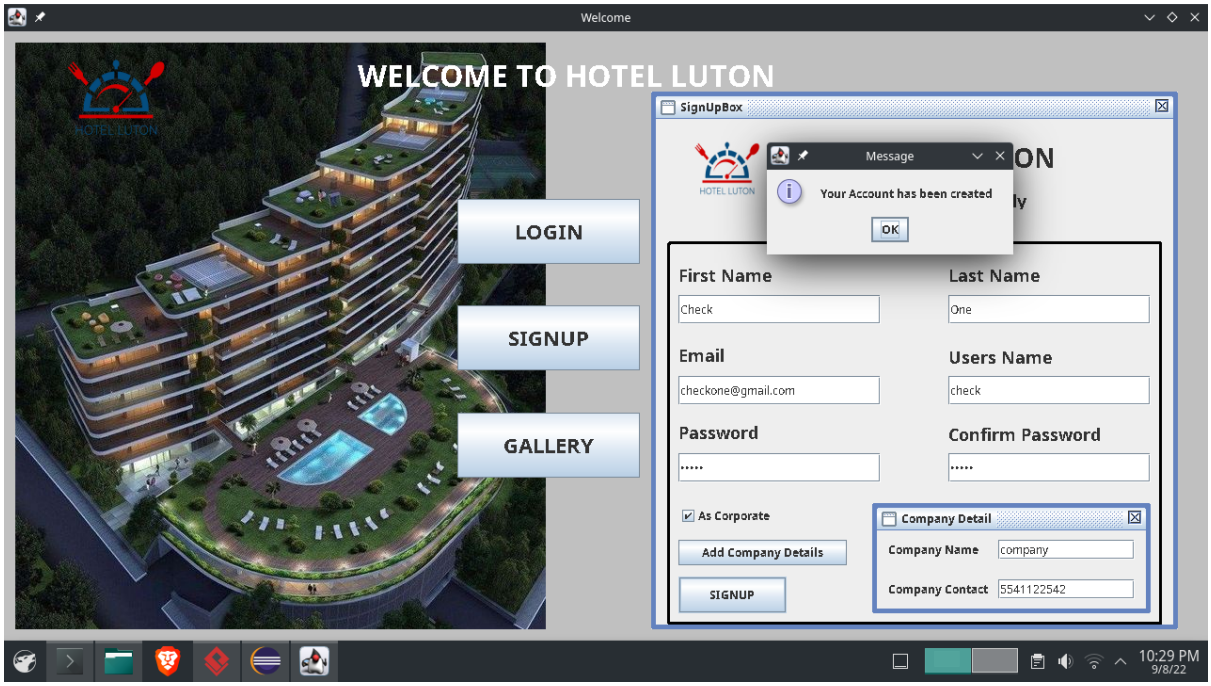
First when app is runned , Luton welcome page is opened.

SignUp Page



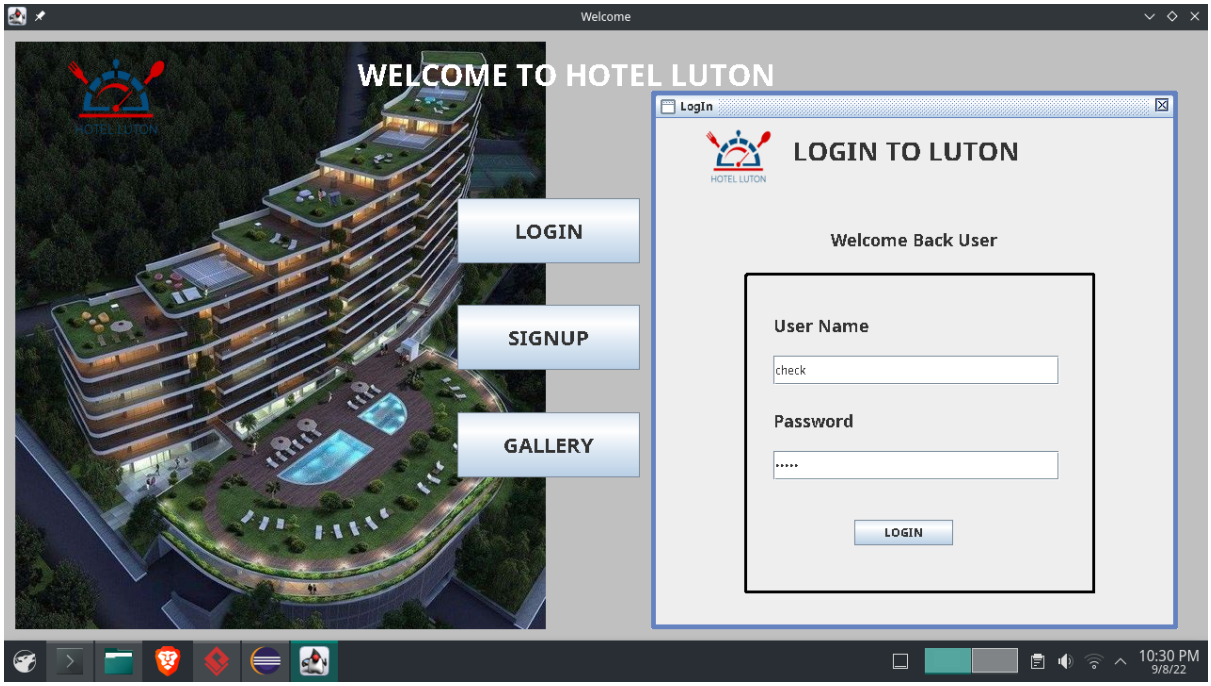
First User Have signup into luton application.

Signning Up as Corporate User

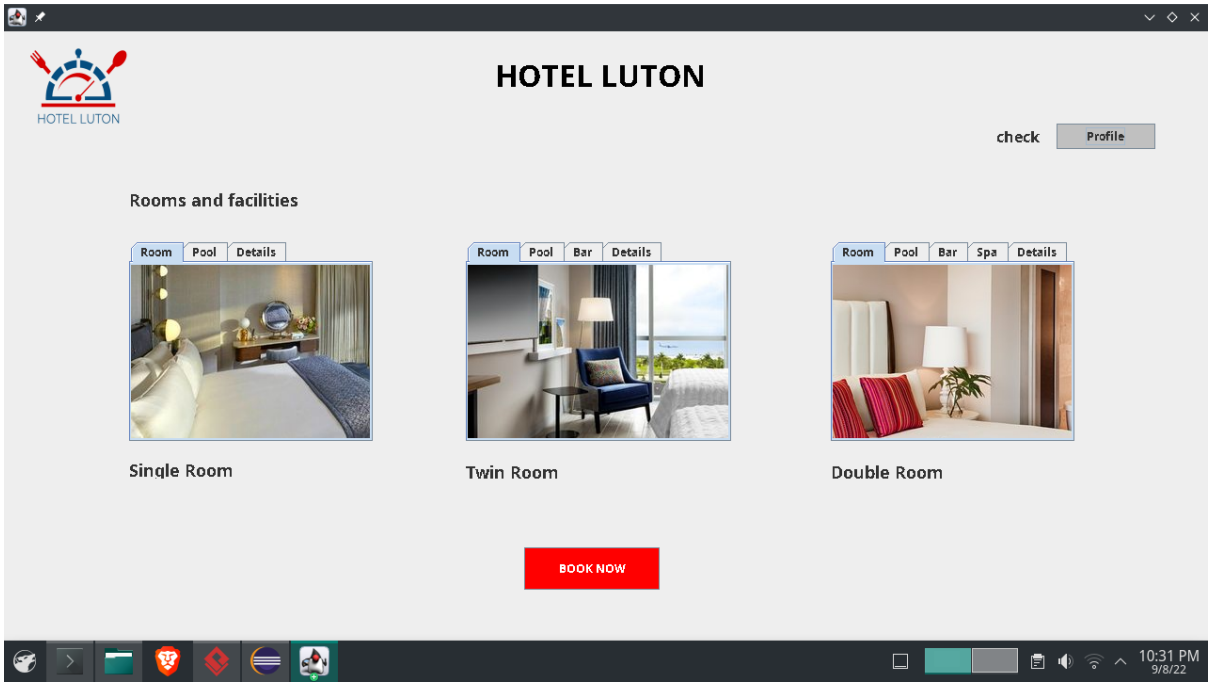


to signup as corporate user Customer must check on As corporate box and the add there company details.

Now Logging In Using Creadted User data

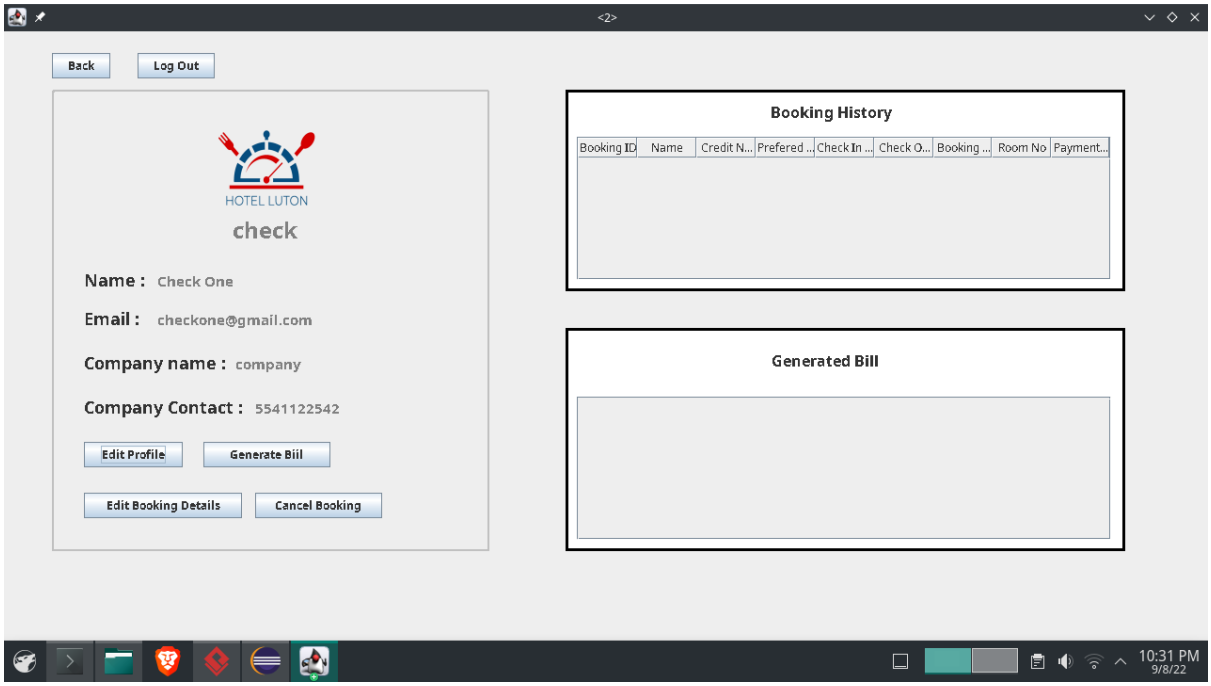


User Home Page



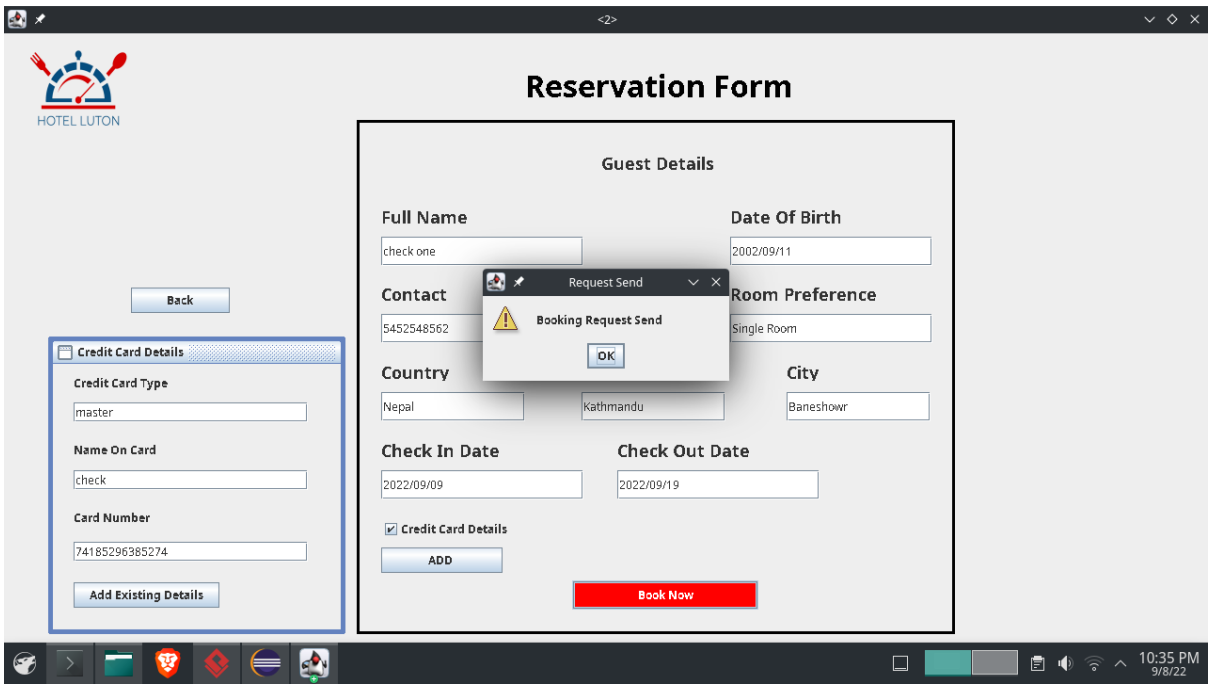
After Logging In User are redirected to user home page where they can see there profile and place there booking.

User Profile Page before making any booking



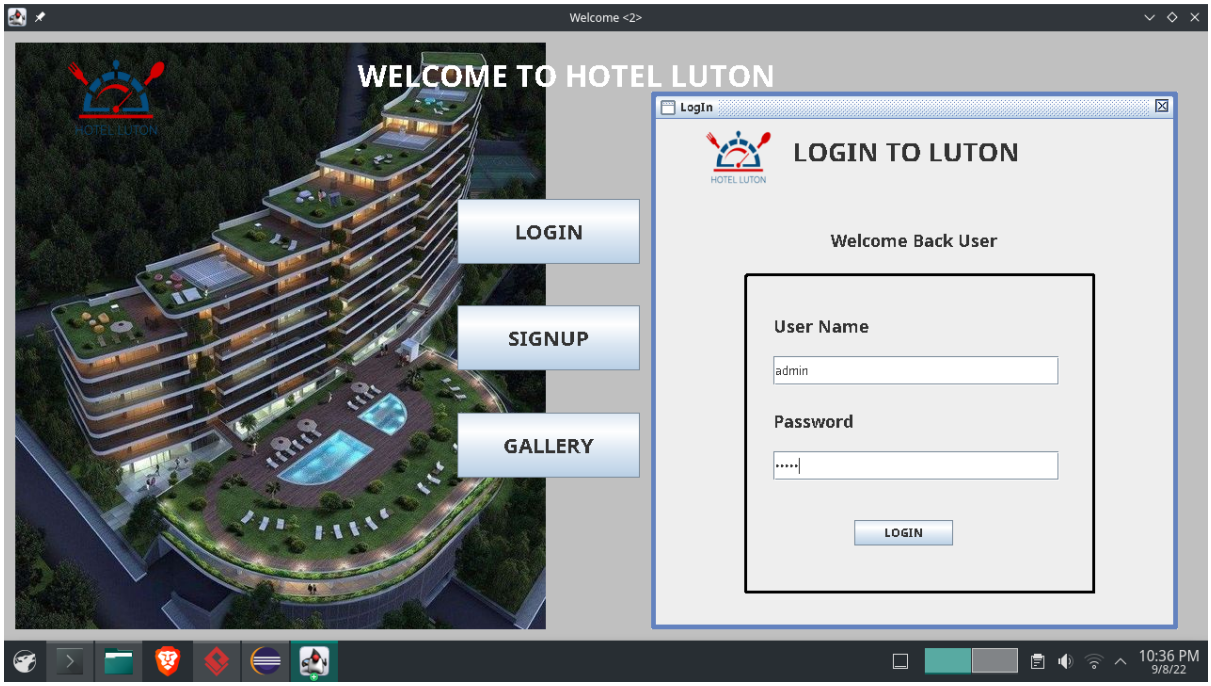
User can see there details.

Adding details to place booking sucessfully



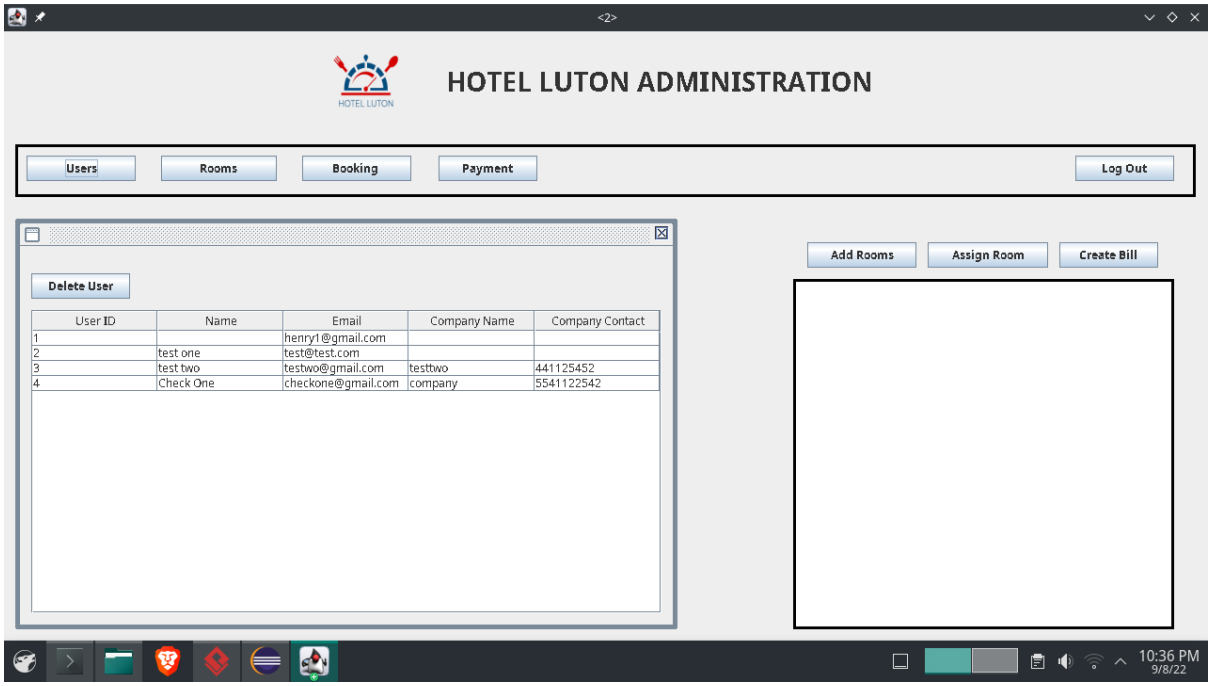
Every detail must be kept wisely and credit card detail should be added through checking the credit card details checkbox.

Logging in As Admin



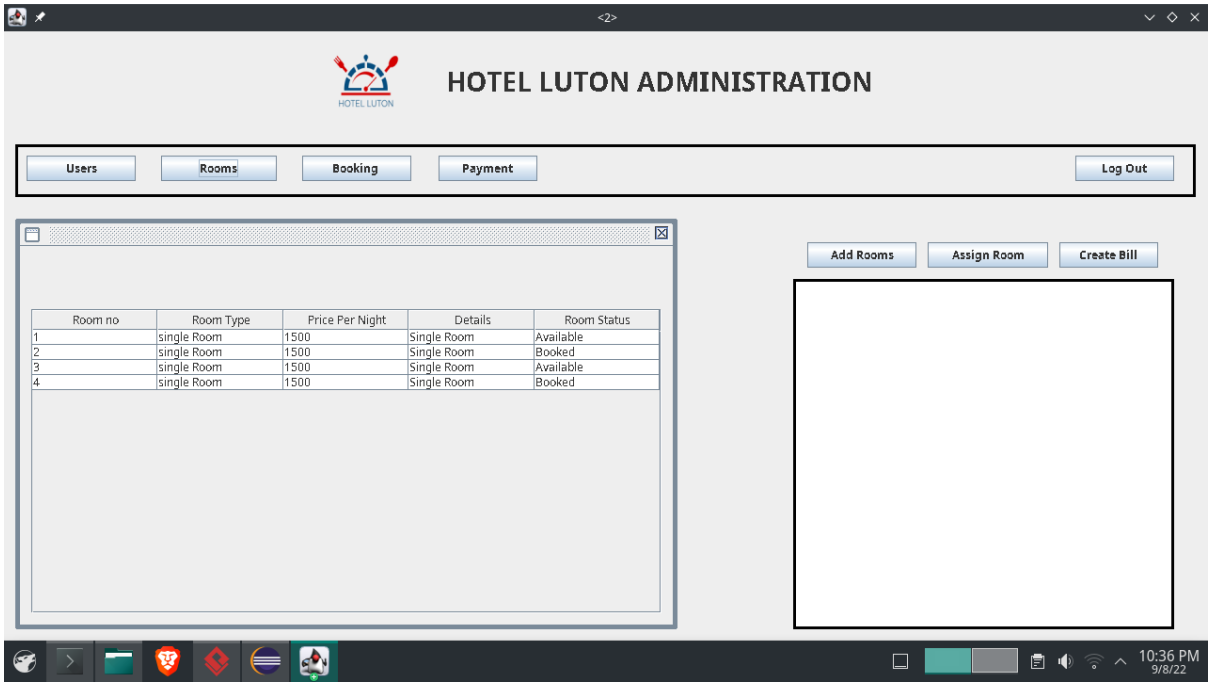
Admin have a special user name and password which are given to only the main receptionist.

Viewing Users details from receptionist page



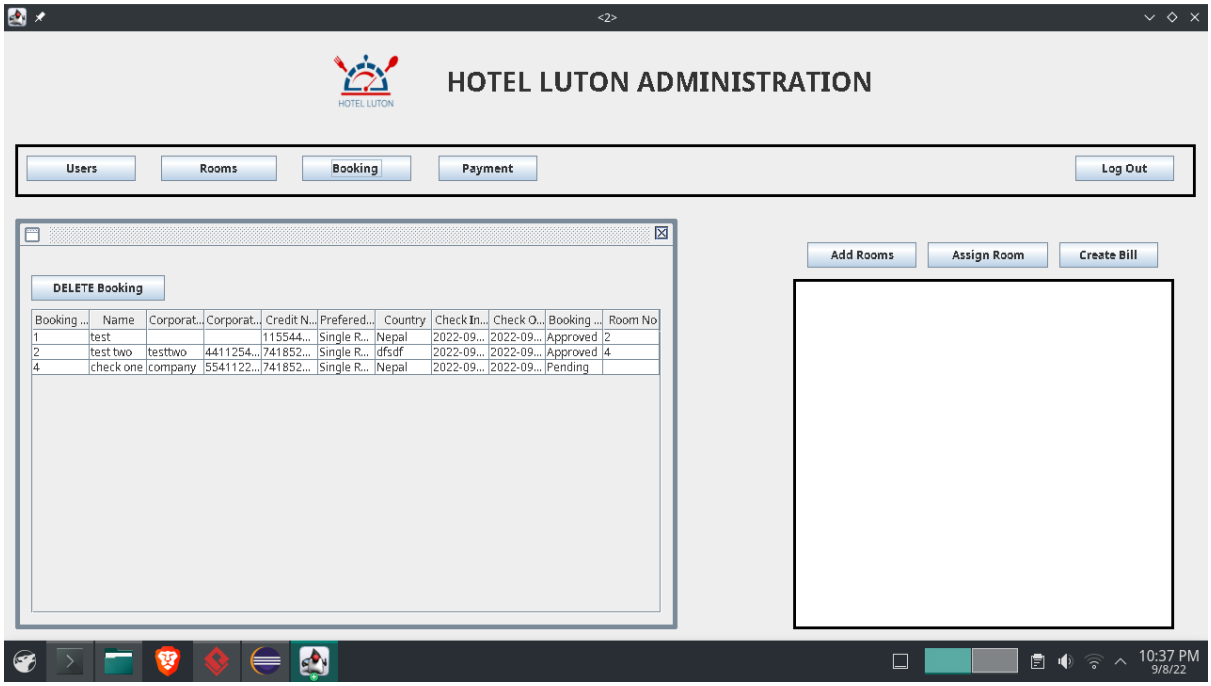
After clicking on users buttoon the data of registered user is shown in below table.

Viewing Room details from receptionist page



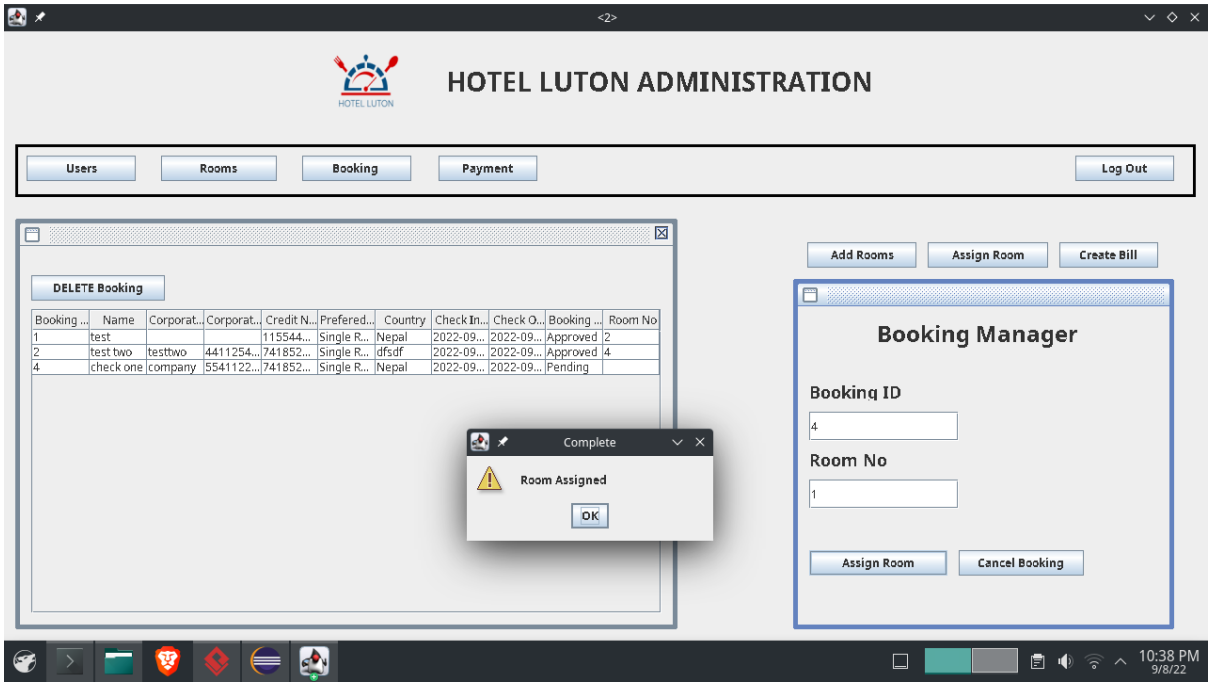
After clicking the room button details of every available and booked room is shown bellow in the table.

Viewing Booking details from receptionist page



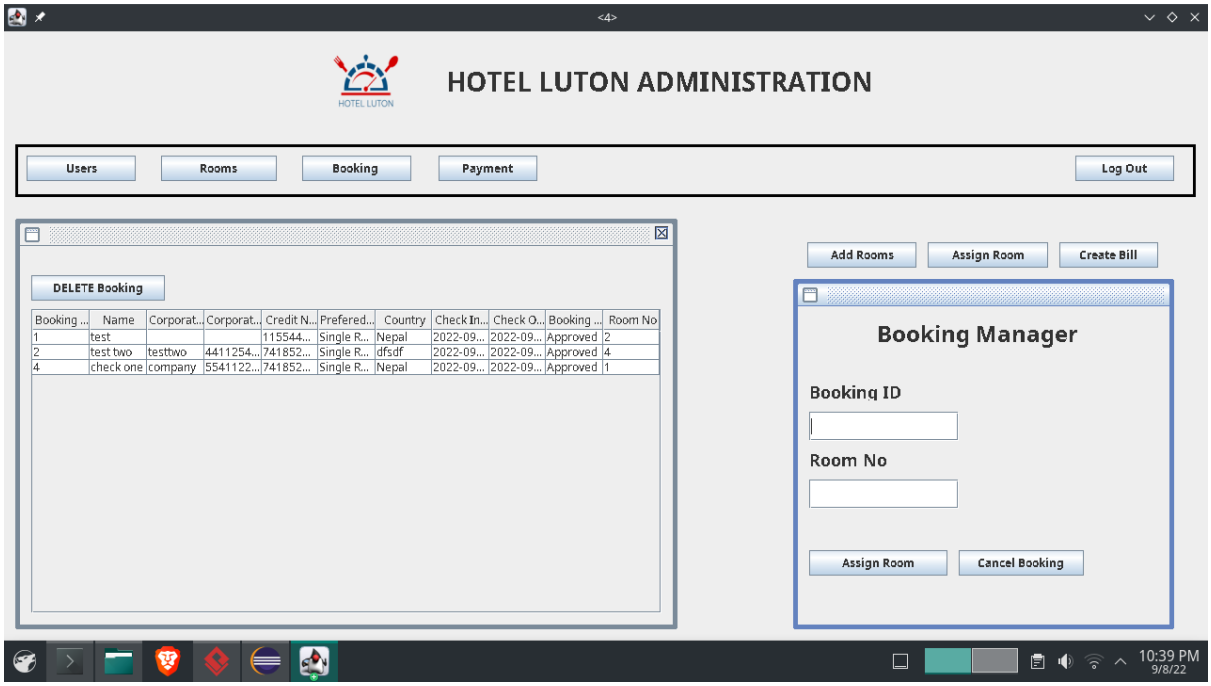
After Clicking on booking button all the details of booking made is shown bellow in the table.

Assigned room to booking id with choosen room number



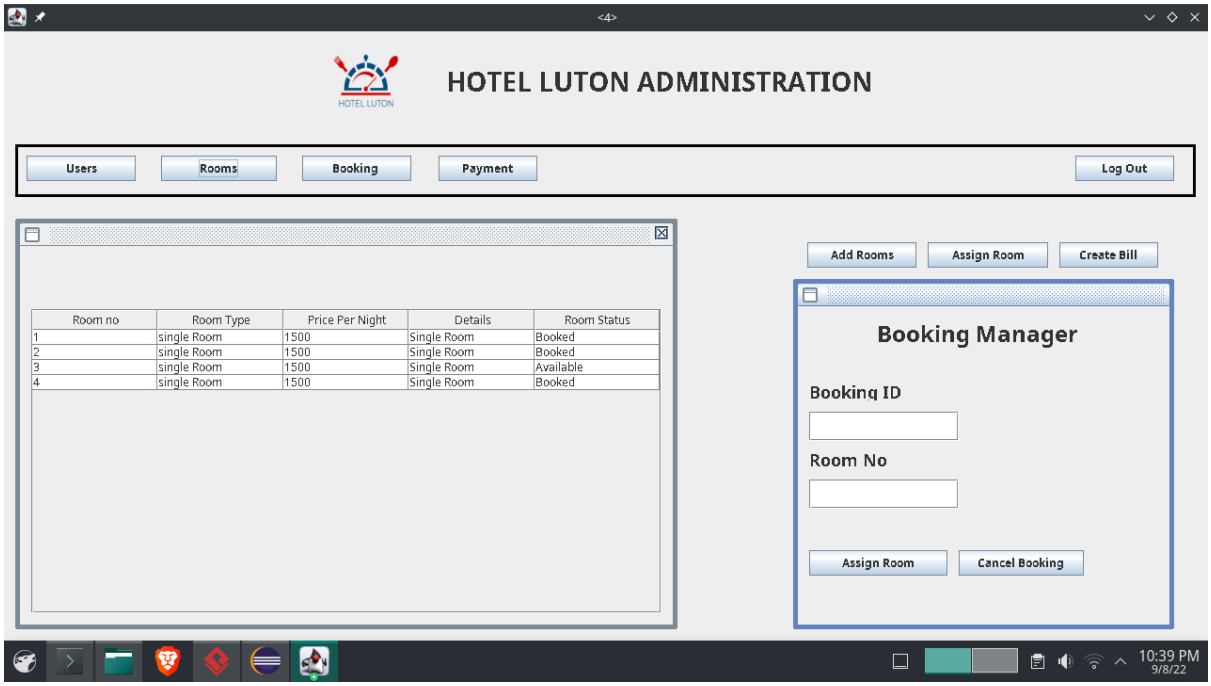
Afte clicking on assign room button at left hand side a box is popped up where receptionist can add the booking id which have to be accepted and the room number to be assigned to the booking.

Viewing Details after room is assigned



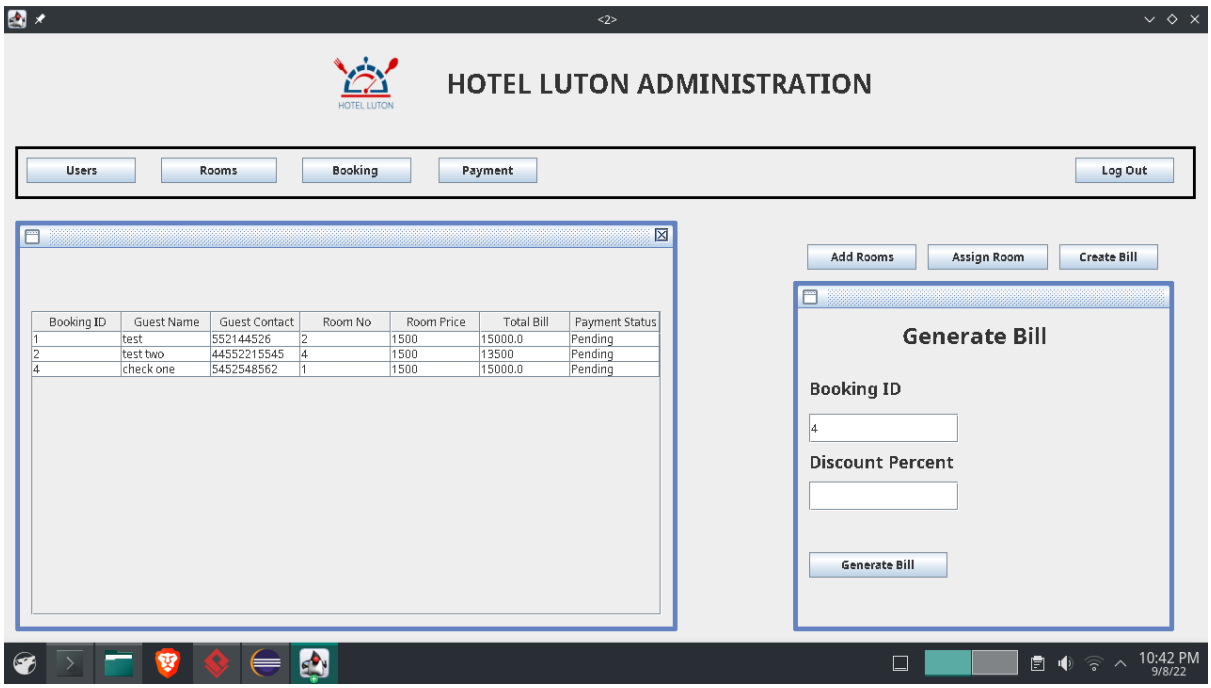
After room is to a booking ID Booking status is changed to Approved from Pending and Assigned room number is add to Room No table.

Checking Room status after room is assigned



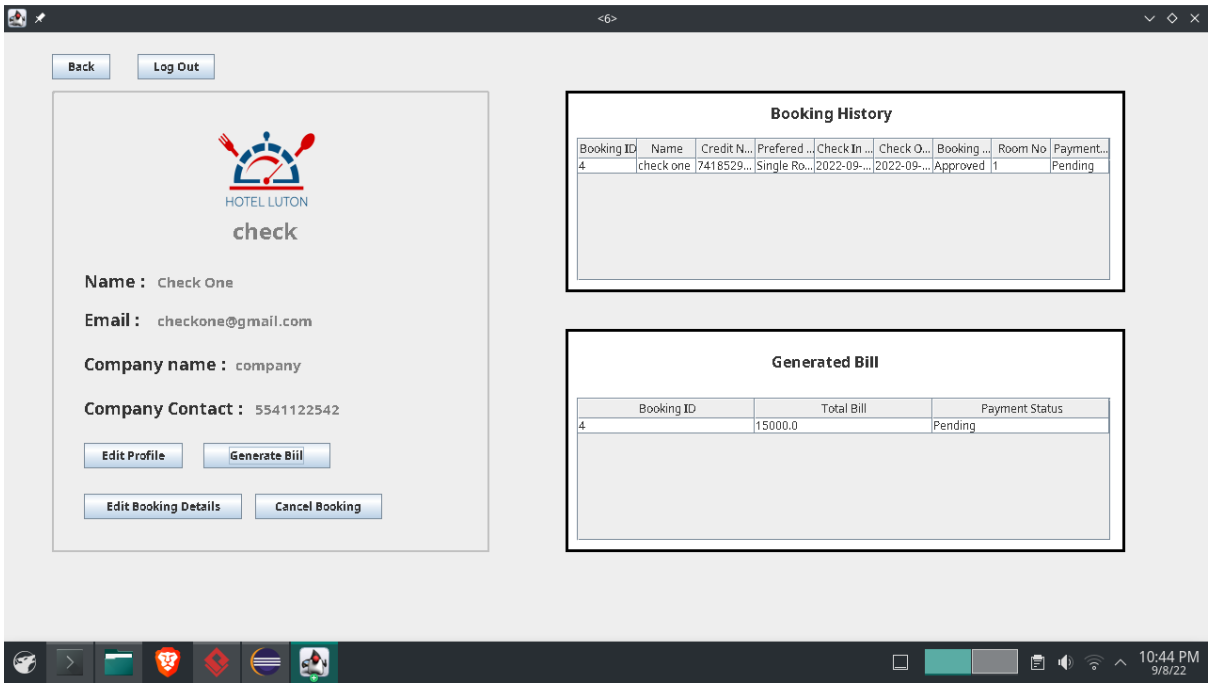
Now the status of assigned room is changed to booked from Available.

Generating bill with and without discount



After clicking on create bill button a box is paopped up where receptionist can add booking ID to generate bill and discount percent. In Abouve figure without discount is created for booking ID 4 but with same data bill is created for booking ID 2 with discount.

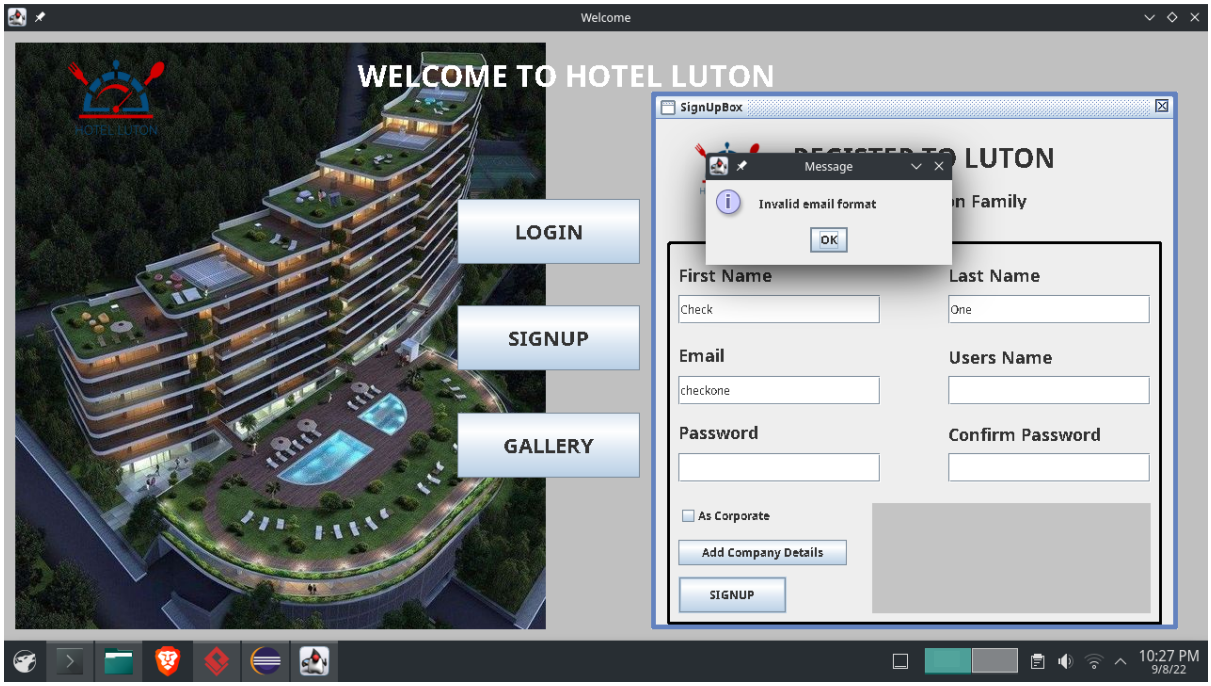
Checking Personal booking and bill detail from personal profile page



After booking is made and approved and bill generated by Receptionist the personal detail for the logged in user is shown in the following two table.

Testing

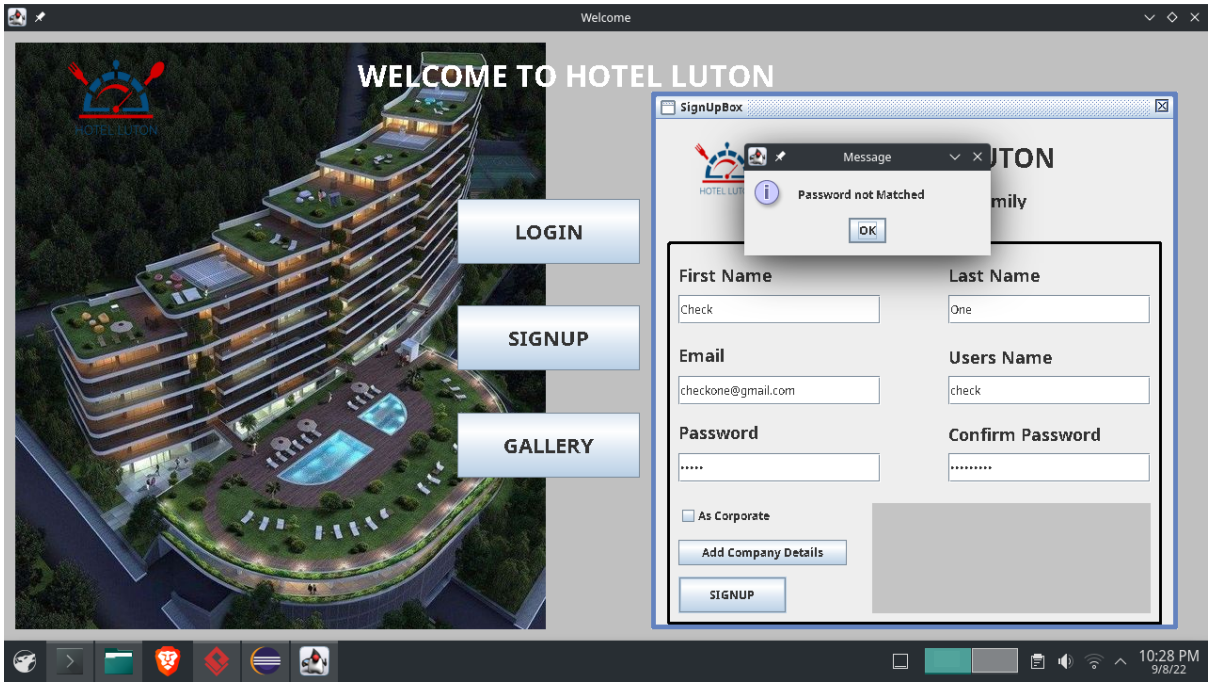
1) Testing the email address format



The following is done to check the email address format is required or not.

After adding the non email format the error must show of erroe format

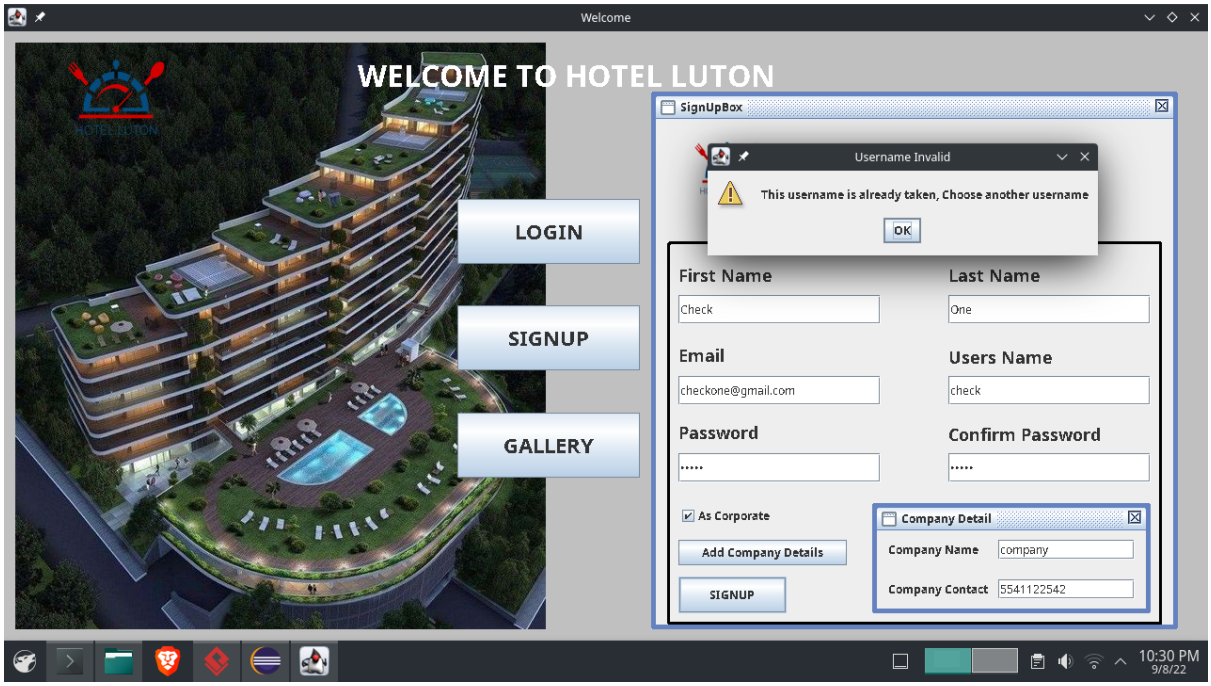
2) **Matching Password and Confirm Password**



The test is done in order to find if the password and confrim password should be same in order to signin or not.

When password and confirm password are inputed different the error message must pop up with password not matched message.

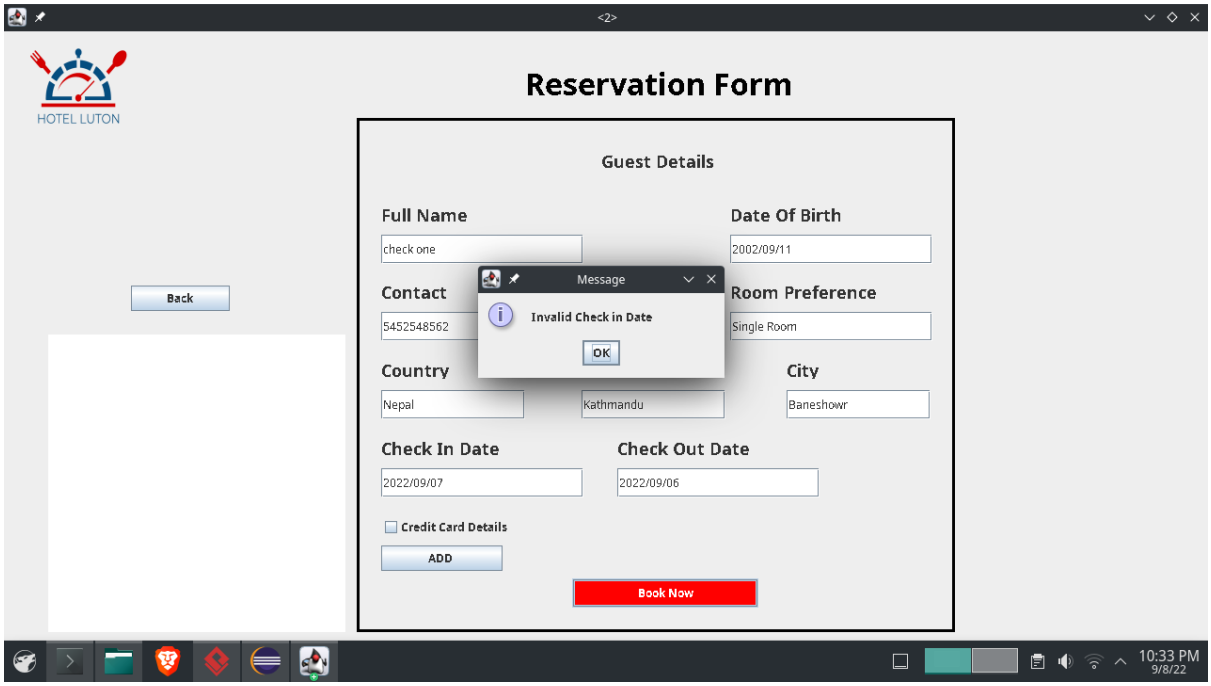
3) **Checking user name is already used or not**



The test is done in order to find weather the input user name is already is used or not as the user ID is the main component to logIn so the user ID of two user cannot be same.

After Useing the same user ID that was used to create the last User the massage must show giving the message of user name is already used.

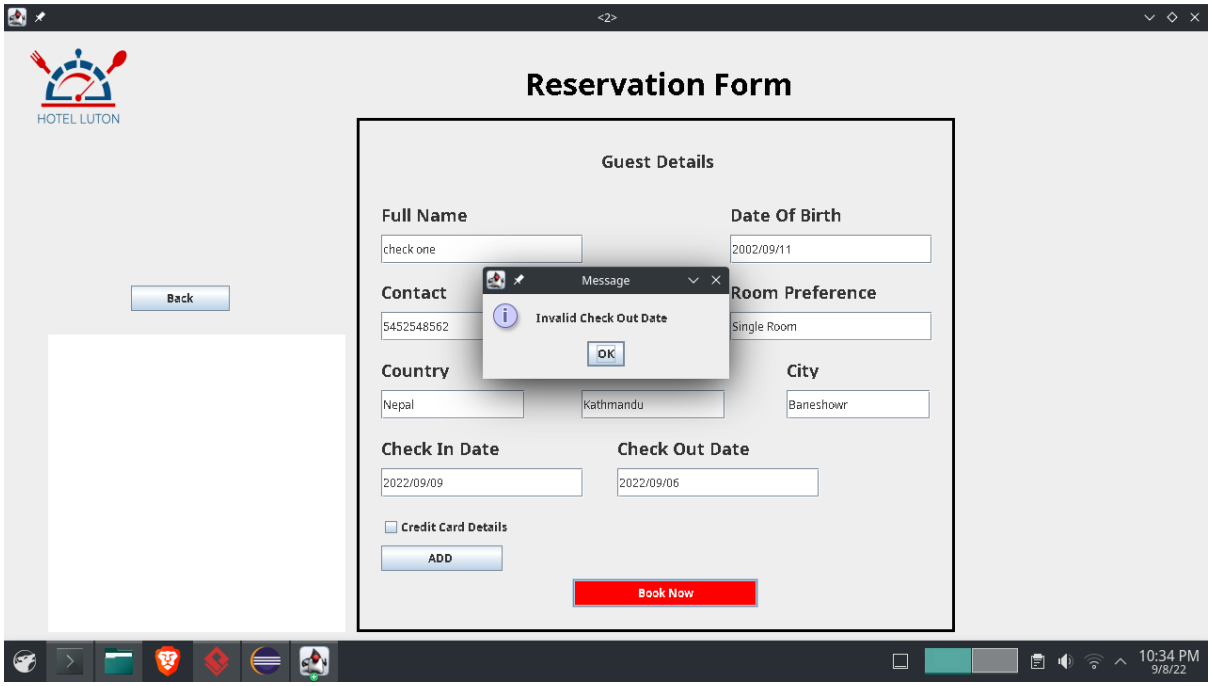
4) **Booking Check In Date checking**



The test is done to check if the check in date can be inputed back than today date or not

After using the passed date from the today date the error message must pop up to inform the useer that input check in date is invalid.

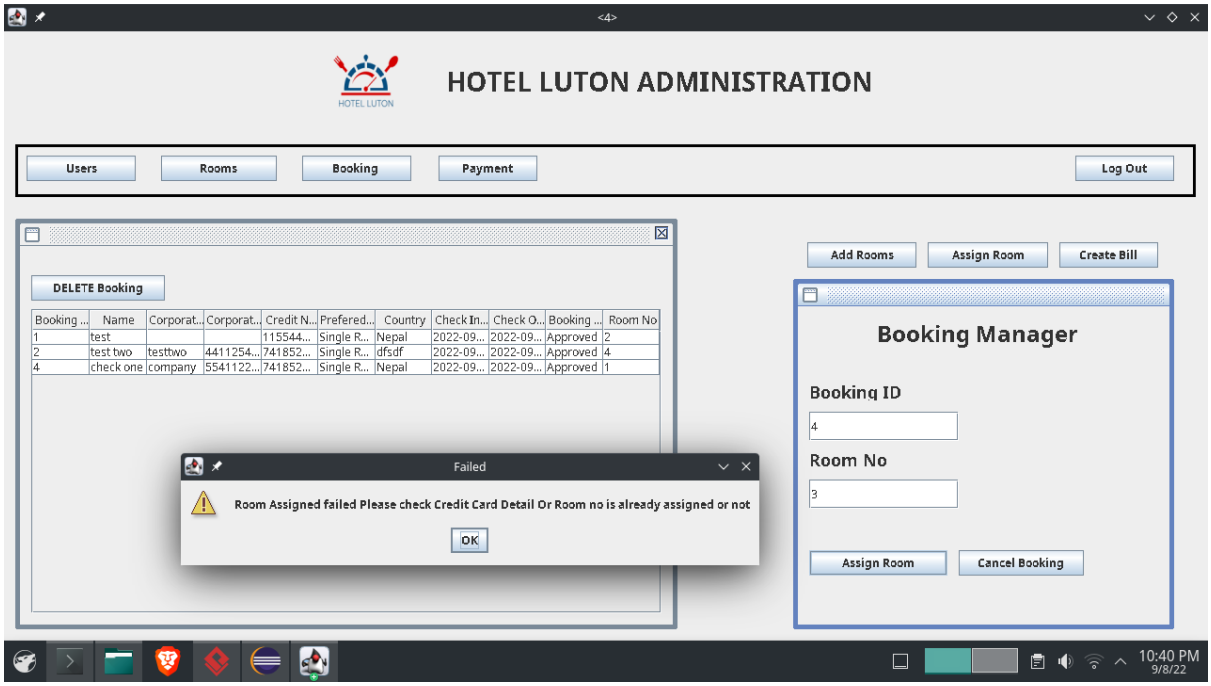
5) **Booking Check Out Date Checking**



After the check in date is corrected the test is done to check weather we can add out check out date before than the check in date.

After inputing the check out date before than the check in date the error mut show to inform the user that inputed data for check out date is invalid.

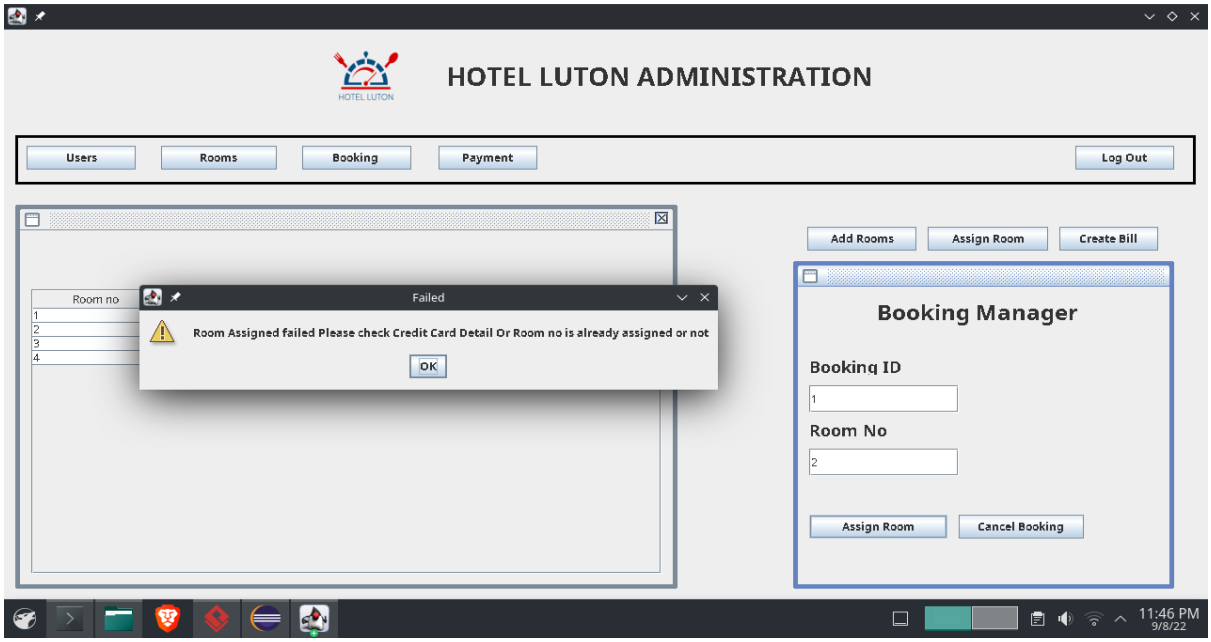
6) **Assign room Booking Check**



The test is to check weather we can add same booking ID to 2 Room number or not

As the Booking ID 4 was aready used by the room number 1 so after assigning the room number 3 again into the Booking ID 4 the error must pop up to notify the reciptionist that the booking is already used.

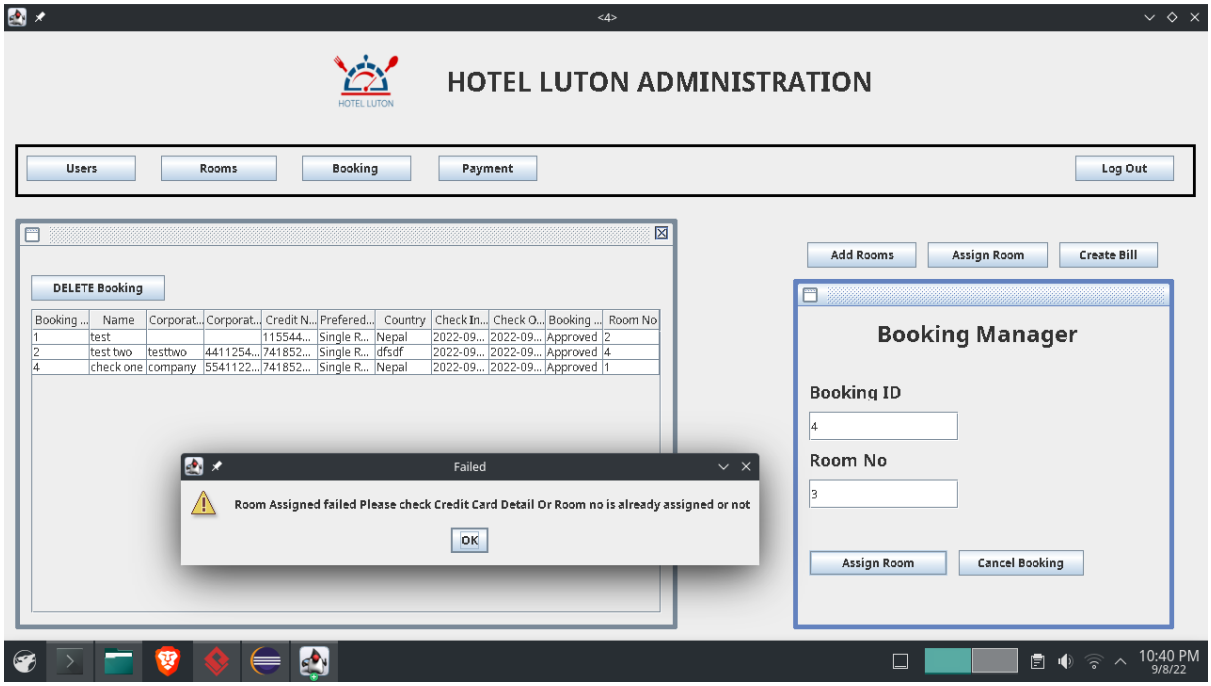
7) **Assign room Room details Check**



The test is to check weather we can add same Room number to different booking ID or not

As the Room no 2 was aready used by the Booking ID 1 so after assigning the room number 2 again into the Booking ID 1 the error must pop up to notify the reciptionist that the Room is already used to another booking .

8) **Assign room Credit Card details Check**



The test is to check if we can assign rooom to the booking request that don't have a credit card detail as credit card detail was the must essential to get the booking request approved.

When receptionist try to approve the booking ID without having credit card details the erroe is pop up informing the receptionist not to approve the booking that don't provide the credit card details.

Discussion / Critical Analysis / Reflection

To complete the project, information must be gathered from many sources and used to design, construct, and deploy a database system. From research database shown below was established.

While researching potential users of hotel luton appplication three entires were created Customer (to store customer data), Corporate (to store company details) and

administration(to store data for receptionist) and following all three entities Users Entity was created in order to store all the login data for all user.
Use case diagram was created in order to show the functionality of a system using actor and use cases. The "actors" are people or entities operating under defined roles within the system.

Representing the ERM diagram a skeleton table was formed containing Entity name along with its attributes.

User Interface design was created using balsamiq showing the application GUI will look like.

When application was completely developed part Implementation was completed whowing the completion of program and it’s features.

Test were conducted to show the required result where errror were pass to generate the right error pop message.

Conclusion

The Overall Task was to provide Hotel Luton with full functioning Online Booking application. And, the task was completed successfully, with all of the requirements met. All of the challenges were solved and all of the assigned tasks were performed via ongoing study and learning. So, the Hotel Ltton application now operational.

References

Visual Paradigm 2022 . What is Use Case Diagram?Accessed on: 9th September, 2022.Retrieved form:https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/

tutorialspoint 2022.UML - Class Diagram. Accessed on: 9th September, 2022. Retrieved from:https://www.tutorialspoint.com/uml/uml_class_diagram.htm

Visual Paradigm 2022 . What is Activity Diagram?.Accessed on: 9th September, 2022.Retrieved form:https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/

tutorialspoint 2022.ER Diagram Representation. Accessed on: 9th September, 2022. Retrieved from:https://www.tutorialspoint.com/dbms/er_diagram_representation.htm#

Appendix

Package FrontendLayer

Main.java

```
package Base;

import FrontendLayer.UserDetailsTable;
import FrontendLayer.WelcomePage;

public class Main {
	public static void main(String[] args) {
		try {
			WelcomePage welcomePage = new WelcomePage();
```

```
                    welcomePage.setVisible(true);
                }catch (Exception e) {
                    e.printStackTrace();
                }
            }
}
```

WelcomePage.java

```
package FrontendLayer;

import java.awt.Image;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import javax.swing.JLabel;
import java.awt.Color;
import javax.swing.SwingConstants;
import java.awt.Font;
import javax.swing.UIManager;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JDesktopPane;

public class WelcomePage extends JFrame {

private static final long serialVersionUID = 7881920390015596332L;
private JPanel contentPane;

/**
```

```
* Launch the application.
*/

/**
* Create the frame.
*/
public WelcomePage() {
setBackground(Color.WHITE);
setTitle("Welcome");
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setBounds(100, 100, 1920,1080);
contentPane = new JPanel();
contentPane.setBackground(Color.LIGHT_GRAY);
contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
setContentPane(contentPane);
contentPane.setLayout(null);
JLabel lblWelcomeToHotel = new JLabel("WELCOME TO HOTEL LUTON");
lblWelcomeToHotel.setForeground(Color.WHITE);
lblWelcomeToHotel.setFont(new Font("Dialog", Font.BOLD, 32));
lblWelcomeToHotel.setHorizontalAlignment(SwingConstants.CENTER);
lblWelcomeToHotel.setBackground(UIManager.getColor("Button.select"));
lblWelcomeToHotel.setBounds(313, 12, 570, 69);
contentPane.add(lblWelcomeToHotel);

JDesktopPane externalDesktopPane = new JDesktopPane();
externalDesktopPane.setBorder(null);
externalDesktopPane.setBackground(Color.LIGHT_GRAY);
externalDesktopPane.setBounds(688, 64, 560, 571);
contentPane.add(externalDesktopPane);
JButton btnNewButton = new JButton("LOGIN");
btnNewButton.setFont(new Font("Dialog", Font.BOLD, 22));
btnNewButton.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
externalDesktopPane.removeAll();
```

```
LogInBox logIn = new LogInBox();
logIn.setVisible(true);
externalDesktopPane.add(logIn);
}
});
btnNewButton.setBounds(482, 178, 194, 69);
contentPane.add(btnNewButton);
JButton btnSignup = new JButton("SIGNUP");
btnSignup.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
externalDesktopPane.removeAll();
SignUpBox signUp = new SignUpBox();
signUp.setVisible(true);
externalDesktopPane.add(signUp);
}
});
btnSignup.setFont(new Font("Dialog", Font.BOLD, 22));
btnSignup.setBounds(482, 291, 194, 69);
contentPane.add(btnSignup);
JButton btnGallery = new JButton("GALLERY");
btnGallery.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
}
});
btnGallery.setFont(new Font("Dialog", Font.BOLD, 22));
btnGallery.setBounds(482, 405, 194, 69);
contentPane.add(btnGallery);
JLabel logoLabel = new JLabel("");
logoLabel.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment and
Feedback/Eclipse/hotelLutonApplication/Img/LOGO.png"));
logoLabel.setBounds(52, 12, 135, 115);
contentPane.add(logoLabel);
JLabel WallpapperLabel = new JLabel("");
```

```
WallpapperLabel.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment                    and
Feedback/Eclipse/hotelLutonApplication/Img/HotelWallpaper.jpg"));
WallpapperLabel.setBackground(Color.WHITE);
WallpapperLabel.setBounds(12, 12, 1256, 623);
contentPane.add(WallpapperLabel);
}
}
```

LogInBox.java

```
package FrontendLayer;

import java.awt.Color;

import java.awt.Font;
import java.awt.Image;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseListener;

import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JInternalFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;
import javax.swing.border.LineBorder;
import javax.swing.plaf.basic.BasicInternalFrameUI;

import Helper.InputException;
```

```
import Models.Users;
import ServiceLayer.UserServiceLayer;

@SuppressWarnings("serial")
public class LogInBox extends JInternalFrame {
public JTextField liUserNameTextField;
public JPasswordField liPasswordPassField;

/**
* Launch the application.
*/
/**
* Create the frame.
*/
public LogInBox() {
setTitle("LogIn");
setClosable(true);
setBounds(0, 0, 560, 571);
getContentPane().setLayout(null);
setVisible(true);
BasicInternalFrameUI basicInternalFrameUI =
((javax.swing.plaf.basic.BasicInternalFrameUI)this.getUI());
for(MouseListener listener:
basicInternalFrameUI.getNorthPane().getMouseListeners()){
basicInternalFrameUI.getNorthPane().removeMouseListener(listener);
}
JLabel logTitle = new JLabel("LOGIN TO LUTON");
logTitle.setFont(new Font("Dialog", Font.BOLD, 28));
logTitle.setBounds(145, 26, 249, 19);
getContentPane().add(logTitle);
JLabel liLogoLabel = new JLabel("");
liLogoLabel.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment and
Feedback/Eclipse/hotelLutonApplication/Img/LOGO2.png"));
```

```
liLogoLabel.setBounds(43, 0, 90, 80);
getContentPane().add(liLogoLabel);
JLabel logInMess = new JLabel("Welcome Back User");
logInMess.setFont(new Font("Dialog", Font.BOLD, 18));
logInMess.setBounds(186, 121, 191, 17);
getContentPane().add(logInMess);
JPanel logInPanel = new JPanel();
logInPanel.setBorder(new LineBorder(new Color(0, 0, 0), 3, true));
logInPanel.setBounds(94, 164, 373, 340);
getContentPane().add(logInPanel);
logInPanel.setLayout(null);
JLabel liUserNameLabel = new JLabel("User Name");
liUserNameLabel.setFont(new Font("Dialog", Font.BOLD, 18));
liUserNameLabel.setBounds(31, 48, 166, 17);
logInPanel.add(liUserNameLabel);
liUserNameTextField = new JTextField();
liUserNameTextField.setBounds(31, 88, 304, 31);
logInPanel.add(liUserNameTextField);
liUserNameTextField.setColumns(10);
JLabel liPasswordLabel = new JLabel("Password");
liPasswordLabel.setFont(new Font("Dialog", Font.BOLD, 18));
liPasswordLabel.setBounds(31, 149, 166, 17);
logInPanel.add(liPasswordLabel);
liPasswordPassField = new JPasswordField();
liPasswordPassField.setColumns(10);
liPasswordPassField.setBounds(31, 189, 304, 31);
logInPanel.add(liPasswordPassField);
JButton btnLogin = new JButton("LOGIN");
btnLogin.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
userLogIn();
}
});
btnLogin.setBounds(117, 262, 105, 27);
```

```
logInPanel.add(btnLogin);
}
public JTextField getLiUserNameTextField() {
return liUserNameTextField;
}

public void setLiUserNameTextField(JTextField liUserNameTextField) {
this.liUserNameTextField = liUserNameTextField;
}

public JPasswordField getLiPasswordPassField() {
return liPasswordPassField;
}

public void setLiPasswordPassField(JPasswordField liPasswordPassField) {
this.liPasswordPassField = liPasswordPassField;
}

private void userLogIn() {
// On click of the log In button
// It fetch the data from the field
// Create an object of Service Layer and pass the model to Service layer
// Perform the required action from the Service layer.
try {
Users user = new Users();
user.setUserName(liUserNameTextField.getText());
user.setPassword(String.valueOf(liPasswordPassField.getPassword()));
UserServiceLayer userSL = new UserServiceLayer();
userSL.ValidateLogIn(user);
userSL.userLogIn(user);
}
```

```
catch(InputException ex) {
JOptionPane.showMessageDialog(null, ex.getMessage());
}
catch(Exception ex) {
JOptionPane.showMessageDialog(null, ex.getMessage());
}
}
}
```

UserHomePage.java

```
package FrontendLayer;

import java.awt.BorderLayout;
import java.awt.EventQueue;

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;

import DatabaseLayer.UserDatabaseLayer;

import javax.swing.JLabel;
import javax.swing.UIManager;
import java.awt.Font;
import java.awt.Color;
import javax.swing.SwingConstants;
import javax.swing.ImageIcon;
import javax.swing.JTabbedPane;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

public class UserHomePage extends JFrame {
```

```
private JPanel contentPane;


/**
* Create the frame.
*/
public UserHomePage() {
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setBounds(100, 100, 1920, 1080);
contentPane = new JPanel();
contentPane.setForeground(Color.WHITE);
contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
setContentPane(contentPane);
contentPane.setLayout(null);
JLabel lblWelcomeToHotel = new JLabel("HOTEL LUTON");
lblWelcomeToHotel.setHorizontalAlignment(SwingConstants.CENTER);
lblWelcomeToHotel.setForeground(Color.BLACK);
lblWelcomeToHotel.setFont(new Font("Dialog", Font.BOLD, 32));
lblWelcomeToHotel.setBackground(UIManager.getColor("Button.select"));
lblWelcomeToHotel.setBounds(456, 12, 356, 69);
contentPane.add(lblWelcomeToHotel);
JTabbedPane singleRoomTabbedPane = new JTabbedPane(JTabbedPane.TOP);
singleRoomTabbedPane.setBounds(133, 222, 259, 213);
contentPane.add(singleRoomTabbedPane);
JLabel singleRoomLabel = new JLabel("");
singleRoomTabbedPane.addTab("Room", null, singleRoomLabel, null);
singleRoomLabel.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment                    and
Feedback/IMAGE/SingleRoom.jpg"));
JLabel sRSwimmingLable = new JLabel("");
singleRoomTabbedPane.addTab("Pool", null, sRSwimmingLable, null);
```

```
sRSwimmingLable.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment and
Feedback/IMAGE/SwimmingPool.jpg"));
JLabel singleRoomDetailsLabel = new JLabel("<html>Room Type: Single
Room<br/>"
+ "Room Price: 1500/-<br>Room Facilities: Swimming Pool</html>");
singleRoomTabbedPane.addTab("Details", null, singleRoomDetailsLabel, null);
singleRoomDetailsLabel.setBackground(Color.CYAN);
singleRoomDetailsLabel.setFont(new Font("Dialog", Font.BOLD, 18));
JTabbedPane deluxRoomTabbedPane = new JTabbedPane(JTabbedPane.TOP);
deluxRoomTabbedPane.setBounds(491, 222, 282, 213);
contentPane.add(deluxRoomTabbedPane);
JLabel deluxRoomLabel = new JLabel("");
deluxRoomTabbedPane.addTab("Room", null, deluxRoomLabel, null);
deluxRoomLabel.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment and
Feedback/IMAGE/TwinRoom.jpg"));
JLabel tRSwimming = new JLabel("");
deluxRoomTabbedPane.addTab("Pool", null, tRSwimming, null);
tRSwimming.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment and
Feedback/IMAGE/SwimmingPool.jpg"));
JLabel tRBarLabel = new JLabel("");
deluxRoomTabbedPane.addTab("Bar", null, tRBarLabel, null);
tRBarLabel.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment and
Feedback/IMAGE/Bar.jpg"));
JLabel deluxRoomDetailsLabel = new JLabel("<html>Room Type: Twin
Room<br/>"
+ "Room Price: 2500/-<br>Room Facilities: Swimming Pool, Bar</html>");
deluxRoomTabbedPane.addTab("Details", null, deluxRoomDetailsLabel, null);
deluxRoomDetailsLabel.setFont(new Font("Dialog", Font.BOLD, 18));
deluxRoomDetailsLabel.setBackground(Color.CYAN);
JTabbedPane doubleRoomTabbedPane = new JTabbedPane(JTabbedPane.TOP);
```

```
doubleRoomTabbedPane.setBounds(879, 222, 259, 213);
contentPane.add(doubleRoomTabbedPane);
JLabel doubleRoomLabel = new JLabel("");
doubleRoomTabbedPane.addTab("Room", null, doubleRoomLabel, null);
doubleRoomLabel.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment and
Feedback/IMAGE/DoubleRoom.jpg"));
JLabel doRSwimmingLabel = new JLabel("");
doubleRoomTabbedPane.addTab("Pool", null, doRSwimmingLabel, null);
doRSwimmingLabel.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment and
Feedback/IMAGE/SwimmingPool.jpg"));
JLabel dorBarLabel = new JLabel("");
doubleRoomTabbedPane.addTab("Bar", null, dorBarLabel, null);
dorBarLabel.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment and
Feedback/IMAGE/Bar.jpg"));
JLabel dorSpaLabel = new JLabel("");
doubleRoomTabbedPane.addTab("Spa", null, dorSpaLabel, null);
dorSpaLabel.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment and
Feedback/Eclipse/hotelLutonApplication/Img/Spa.jpg"));
JLabel doubleRoomDetailsLabel = new JLabel("<html>Room Type: Double
Room<br/>Room Price: 3500/-<br>"
+ "Room Facilities: Swimming Pool, Bar , Spa</html>");
doubleRoomTabbedPane.addTab("Details", null, doubleRoomDetailsLabel, null);
doubleRoomDetailsLabel.setFont(new Font("Dialog", Font.BOLD, 18));
doubleRoomDetailsLabel.setBackground(Color.CYAN);
JLabel logoLabel = new JLabel("");
logoLabel.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment and
Feedback/Eclipse/hotelLutonApplication/Img/LOGO.png"));
logoLabel.setBounds(12, 0, 135, 115);
contentPane.add(logoLabel);
```

```
JLabel lblRooms = new JLabel("Rooms and facilities");
lblRooms.setFont(new Font("Dialog", Font.BOLD, 18));
lblRooms.setBounds(133, 171, 191, 17);
contentPane.add(lblRooms);
JLabel lblSingleRoom = new JLabel("Single Room");
lblSingleRoom.setFont(new Font("Dialog", Font.BOLD, 18));
lblSingleRoom.setBounds(133, 458, 191, 17);
contentPane.add(lblSingleRoom);
JLabel lblTwinRooms = new JLabel("Twin Room");
lblTwinRooms.setFont(new Font("Dialog", Font.BOLD, 18));
lblTwinRooms.setBounds(491, 460, 191, 17);
contentPane.add(lblTwinRooms);
JLabel lblDoubleRooms = new JLabel("Double Room");
lblDoubleRooms.setFont(new Font("Dialog", Font.BOLD, 18));
lblDoubleRooms.setBounds(879, 460, 191, 17);
contentPane.add(lblDoubleRooms);
JButton btnBookNow = new JButton("BOOK NOW");
btnBookNow.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
BookingFormPage form = new BookingFormPage();
form.setVisible(true);
}
});
btnBookNow.setForeground(Color.WHITE);
btnBookNow.setBackground(Color.RED);
btnBookNow.setBounds(553, 548, 144, 45);
contentPane.add(btnBookNow);
JButton btnProfile = new JButton("Profile");
btnProfile.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
ProfilePage profile = new ProfilePage();
profile.setVisible(true);
profile.loadAllUserbooking();
}
```

```
});
btnProfile.setBackground(Color.LIGHT_GRAY);
btnProfile.setBounds(1119, 98, 105, 27);
contentPane.add(btnProfile);
JLabel userNameLabel = new JLabel(UserDatabaseLayer.uName);
userNameLabel.setFont(new Font("Dialog", Font.BOLD, 16));
userNameLabel.setHorizontalAlignment(SwingConstants.RIGHT);
userNameLabel.setBounds(966, 98, 135, 27);
contentPane.add(userNameLabel);
}
}
```

UserDetailsTable.java

```
package FrontendLayer;


import java.util.ArrayList;


import javax.swing.JInternalFrame;
import javax.swing.JOptionPane;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;


import Models.DefultModel;
import Models.Room;
import Models.Users;
import ServiceLayer.UserDetailsServiceLayer;
import ServiceLayer.UserServiceLayer;

import javax.swing.JScrollPane;
import javax.swing.JButton;
```

```
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;



public class UserDetailsTable extends JInternalFrame {
private DefaultTableModel model;
private JTable userTable;


/**
* Create the frame.
*/
public UserDetailsTable() {
setClosable(true);
setBounds(0, 0, 704, 436);
getContentPane().setLayout(null);
JScrollPane scrollPane = new JScrollPane();
scrollPane.setBounds(12, 68, 670, 322);
getContentPane().add(scrollPane);
userTable = new JTable();
userTable.setFillsViewportHeight(true);
scrollPane.setViewportView(userTable);
JButton btnDeleteUser = new JButton("Delete User");
btnDeleteUser.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {

}
});
btnDeleteUser.setBounds(12, 29, 105, 27);
getContentPane().add(btnDeleteUser);
model = new DefaultTableModel();
Object[] columnsName = new Object[5];
columnsName[0] = "User ID";
```

```
columnsName[1] = "Name";
columnsName[2] = "Email";
columnsName[3] = "Company Name";
columnsName[4] = "Company Contact";
model.setColumnCount(0);
model.setColumnIdentifiers(columnsName);
}
public void loadAllUser() {
try {
UserDetailsServiceLayer userDetailsSL = new UserDetailsServiceLayer();
ArrayList<DefultModel> defultModel = userDetailsSL.getAllData();
setTableData(defultModel);
} catch (Exception e1) {
JOptionPane.showMessageDialog(null, e1.getMessage());
}
}
private void setTableData(ArrayList<DefultModel> defultModel) {
// Create the object array from arraylist and add to the table row
Object[] rowData = new Object[5];
// set the number of rows in table model to zero
model.setRowCount(0);
for(int i=0; i<defultModel.size(); i++) {
rowData[0] = defultModel.get(i).getUserID();
rowData[1] = defultModel.get(i).getName();
rowData[2] = defultModel.get(i).getEmail();
rowData[3] = defultModel.get(i).getCorpName();
rowData[4] = defultModel.get(i).getCorpContact();
model.addRow(rowData);
}
userTable.setModel(model);

}
}
```

SignUpBox.java

```
package FrontendLayer;

import java.awt.Color;




import java.awt.Font;
import java.awt.Image;
import java.awt.SystemColor;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseListener;

import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JDesktopPane;
import javax.swing.JInternalFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;
import javax.swing.border.LineBorder;
import javax.swing.plaf.basic.BasicInternalFrameUI;

import Helper.InputException;
import Models.Corporate;
import Models.Customer;
import Models.Users;
import ServiceLayer.CorporateServiceLayer;
```

```
import ServiceLayer.CustomerServiceLayer;
import ServiceLayer.UserServiceLayer;
import javax.swing.JCheckBox;

public class SignUpBox extends JInternalFrame {
private JTextField siFirstTextField;
private JTextField siLastTextField;
private JTextField siEmailTextField;
private JTextField siUsersNameTextField;
private JPasswordField siPasswordPassField;
private JPasswordField siConfirmPasswordPassField;
public JCheckBox corporateCheckBox;
public JDesktopPane coDesktopPane;
/**
* Create the frame.
*/
public SignUpBox() {
setTitle("SignUpBox");
setVisible(true);
setClosable(true);
setBounds(0, 0, 560, 571);
getContentPane().setLayout(null);
BasicInternalFrameUI basicInternalFrameUI =
((javax.swing.plaf.basic.BasicInternalFrameUI)this.getUI());
for(MouseListener listener:
basicInternalFrameUI.getNorthPane().getMouseListeners()){
basicInternalFrameUI.getNorthPane().removeMouseListener(listener);
}
JLabel signTitle = new JLabel("REGISTER TO LUTON");
signTitle.setFont(new Font("Dialog", Font.BOLD, 28));
signTitle.setBounds(145, 26, 292, 31);
getContentPane().add(signTitle);
JLabel siLogoLabel = new JLabel("");
```

```
siLogoLabel.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment                    and
Feedback/Eclipse/hotelLutonApplication/Img/LOGO2.png"));
siLogoLabel.setBounds(31, 12, 90, 80);
getContentPane().add(siLogoLabel);
JLabel signInMess = new JLabel("Welcome to Luton Family");
signInMess.setFont(new Font("Dialog", Font.BOLD, 18));
signInMess.setBounds(166, 79, 228, 17);
getContentPane().add(signInMess);
JPanel signInPanel = new JPanel();
signInPanel.setBorder(new LineBorder(new Color(0, 0, 0), 3, true));
signInPanel.setBounds(12, 130, 526, 407);
getContentPane().add(signInPanel);
signInPanel.setLayout(null);
JLabel siFirstLabel = new JLabel("First Name");
siFirstLabel.setFont(new Font("Dialog", Font.BOLD, 18));
siFirstLabel.setBounds(12, 28, 179, 17);
signInPanel.add(siFirstLabel);
JLabel siLastLabel = new JLabel("Last Name");
siLastLabel.setFont(new Font("Dialog", Font.BOLD, 18));
siLastLabel.setBounds(299, 28, 198, 17);
signInPanel.add(siLastLabel);
siFirstTextField = new JTextField();
siFirstTextField.setColumns(10);
siFirstTextField.setBounds(12, 57, 215, 31);
signInPanel.add(siFirstTextField);
siLastTextField = new JTextField();
siLastTextField.setColumns(10);
siLastTextField.setBounds(299, 57, 215, 31);
signInPanel.add(siLastTextField);
JLabel siEmailAddreessLabel = new JLabel("Email");
siEmailAddreessLabel.setFont(new Font("Dialog", Font.BOLD, 18));
siEmailAddreessLabel.setBounds(12, 113, 179, 17);
signInPanel.add(siEmailAddreessLabel);
```

```
JLabel siUsersNameLabel = new JLabel("Users Name");
siUsersNameLabel.setFont(new Font("Dialog", Font.BOLD, 18));
siUsersNameLabel.setBounds(299, 115, 179, 17);
signInPanel.add(siUsersNameLabel);
siEmailTextField = new JTextField();
siEmailTextField.setColumns(10);
siEmailTextField.setBounds(12, 143, 215, 31);
signInPanel.add(siEmailTextField);
siUsersNameTextField = new JTextField();
siUsersNameTextField.setColumns(10);
siUsersNameTextField.setBounds(299, 143, 215, 31);
signInPanel.add(siUsersNameTextField);
JLabel siPasswordLabel = new JLabel("Password");
siPasswordLabel.setFont(new Font("Dialog", Font.BOLD, 18));
siPasswordLabel.setBounds(12, 195, 179, 17);
signInPanel.add(siPasswordLabel);
siPasswordPassField = new JPasswordField();
siPasswordPassField.setColumns(10);
siPasswordPassField.setBounds(12, 225, 215, 31);
signInPanel.add(siPasswordPassField);
JButton btnSignup = new JButton("SIGNUP");
btnSignup.setBounds(12, 356, 115, 39);
signInPanel.add(btnSignup);
btnSignup.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
register();
}
});
coDesktopPane = new JDesktopPane();
coDesktopPane.setBackground(SystemColor.scrollbar);
coDesktopPane.setBounds(218, 278, 296, 117);
signInPanel.add(coDesktopPane);
corporateCheckBox = new JCheckBox("As Corporate");
corporateCheckBox.setBounds(12, 278, 115, 27);
```

```
signInPanel.add(corporateCheckBox);
JButton btnCorporateButton = new JButton("Add Company Details");
btnCorporateButton.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
if(corporateCheckBox.isSelected()) {
coDesktopPane.removeAll();
CorporateBox corporate = new CorporateBox();
corporate.setVisible(true);
coDesktopPane.add(corporate);
}
}
});
btnCorporateButton.setBounds(12, 317, 179, 27);
signInPanel.add(btnCorporateButton);
JLabel siConfirmPasswordTextField = new JLabel("Confirm Password");
siConfirmPasswordTextField.setFont(new Font("Dialog", Font.BOLD, 18));
siConfirmPasswordTextField.setBounds(299, 197, 179, 17);
signInPanel.add(siConfirmPasswordTextField);
siConfirmPasswordPassField = new JPasswordField();
siConfirmPasswordPassField.setColumns(10);
siConfirmPasswordPassField.setBounds(299, 225, 215, 31);
signInPanel.add(siConfirmPasswordPassField);
}
private void register() {
// On click of the save button
// Read data from the fields and store it in the model
// Create an object of Business Layer and pass the model to business layer
// Perform the required action from the business layer.
try {
Users user = new Users();
user.setEmail(siEmailTextField.getText());
user.setUserName(siUsersNameTextField.getText());
user.setPassword(String.valueOf(siPasswordPassField.getPassword()));
```

```
user.setConfirmPassword(String.valueOf(siConfirmPasswordPassField.getPassword()
));
Customer customer = new Customer();
customer.setFirstName(siFirstTextField.getText());
customer.setLastName(siLastTextField.getText());
UserServiceLayer userSL = new UserServiceLayer();
CustomerServiceLayer customerSL = new CustomerServiceLayer();
CorporateServiceLayer corporateSL = new CorporateServiceLayer();
if(corporateCheckBox.isSelected()) {
Corporate corporate = new Corporate();
corporate.setCompanyName(CorporateBox.CCompanyNameTextField.getText());
corporate.setCompanyContact(CorporateBox.CCompanyContactTextField.getText());
if(userSL.ValidateSignup(user)  &&  customerSL.ValidateCustomer(customer)  &&
corporateSL.ValidateCorporate(corporate)) {
user = userSL.userSave(user);
customer = customerSL.customerSave(customer);
corporate = corporateSL.corporateSave(corporate);
}
}
else {
if(userSL.ValidateSignup(user) && customerSL.ValidateCustomer(customer)) {
user = userSL.userSave(user);
customer = customerSL.customerSave(customer);
}
}
}
catch(InputException ex) {
JOptionPane.showMessageDialog(null, ex.getMessage());
}
catch(Exception ex) {
JOptionPane.showMessageDialog(null,"Complete your information");
}
}
}
```

RoomDetailsTable.java

```
package FrontendLayer;

import java.awt.Color;




import java.awt.Font;
import java.awt.Image;
import java.awt.SystemColor;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseListener;

import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JDesktopPane;
import javax.swing.JInternalFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;
import javax.swing.border.LineBorder;
import javax.swing.plaf.basic.BasicInternalFrameUI;

import Helper.InputException;
import Models.Corporate;
import Models.Customer;
import Models.Users;
```

```
import ServiceLayer.CorporateServiceLayer;
import ServiceLayer.CustomerServiceLayer;
import ServiceLayer.UserServiceLayer;
import javax.swing.JCheckBox;

public class SignUpBox extends JInternalFrame {
private JTextField siFirstTextField;
private JTextField siLastTextField;
private JTextField siEmailTextField;
private JTextField siUsersNameTextField;
private JPasswordField siPasswordPassField;
private JPasswordField siConfirmPasswordPassField;
public JCheckBox corporateCheckBox;
public JDesktopPane coDesktopPane;
/**
* Create the frame.
*/
public SignUpBox() {
setTitle("SignUpBox");
setVisible(true);
setClosable(true);
setBounds(0, 0, 560, 571);
getContentPane().setLayout(null);
BasicInternalFrameUI basicInternalFrameUI =
((javax.swing.plaf.basic.BasicInternalFrameUI)this.getUI());
for(MouseListener listener:
basicInternalFrameUI.getNorthPane().getMouseListeners()){
basicInternalFrameUI.getNorthPane().removeMouseListener(listener);
}
JLabel signTitle = new JLabel("REGISTER TO LUTON");
signTitle.setFont(new Font("Dialog", Font.BOLD, 28));
signTitle.setBounds(145, 26, 292, 31);
getContentPane().add(signTitle);
JLabel siLogoLabel = new JLabel("");
```

```
siLogoLabel.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment                    and
Feedback/Eclipse/hotelLutonApplication/Img/LOGO2.png"));
siLogoLabel.setBounds(31, 12, 90, 80);
getContentPane().add(siLogoLabel);
JLabel signInMess = new JLabel("Welcome to Luton Family");
signInMess.setFont(new Font("Dialog", Font.BOLD, 18));
signInMess.setBounds(166, 79, 228, 17);
getContentPane().add(signInMess);
JPanel signInPanel = new JPanel();
signInPanel.setBorder(new LineBorder(new Color(0, 0, 0), 3, true));
signInPanel.setBounds(12, 130, 526, 407);
getContentPane().add(signInPanel);
signInPanel.setLayout(null);
JLabel siFirstLabel = new JLabel("First Name");
siFirstLabel.setFont(new Font("Dialog", Font.BOLD, 18));
siFirstLabel.setBounds(12, 28, 179, 17);
signInPanel.add(siFirstLabel);
JLabel siLastLabel = new JLabel("Last Name");
siLastLabel.setFont(new Font("Dialog", Font.BOLD, 18));
siLastLabel.setBounds(299, 28, 198, 17);
signInPanel.add(siLastLabel);
siFirstTextField = new JTextField();
siFirstTextField.setColumns(10);
siFirstTextField.setBounds(12, 57, 215, 31);
signInPanel.add(siFirstTextField);
siLastTextField = new JTextField();
siLastTextField.setColumns(10);
siLastTextField.setBounds(299, 57, 215, 31);
signInPanel.add(siLastTextField);
JLabel siEmailAddreessLabel = new JLabel("Email");
siEmailAddreessLabel.setFont(new Font("Dialog", Font.BOLD, 18));
siEmailAddreessLabel.setBounds(12, 113, 179, 17);
signInPanel.add(siEmailAddreessLabel);
```

```
JLabel siUsersNameLabel = new JLabel("Users Name");
siUsersNameLabel.setFont(new Font("Dialog", Font.BOLD, 18));
siUsersNameLabel.setBounds(299, 115, 179, 17);
signInPanel.add(siUsersNameLabel);
siEmailTextField = new JTextField();
siEmailTextField.setColumns(10);
siEmailTextField.setBounds(12, 143, 215, 31);
signInPanel.add(siEmailTextField);
siUsersNameTextField = new JTextField();
siUsersNameTextField.setColumns(10);
siUsersNameTextField.setBounds(299, 143, 215, 31);
signInPanel.add(siUsersNameTextField);
JLabel siPasswordLabel = new JLabel("Password");
siPasswordLabel.setFont(new Font("Dialog", Font.BOLD, 18));
siPasswordLabel.setBounds(12, 195, 179, 17);
signInPanel.add(siPasswordLabel);
siPasswordPassField = new JPasswordField();
siPasswordPassField.setColumns(10);
siPasswordPassField.setBounds(12, 225, 215, 31);
signInPanel.add(siPasswordPassField);
JButton btnSignup = new JButton("SIGNUP");
btnSignup.setBounds(12, 356, 115, 39);
signInPanel.add(btnSignup);
btnSignup.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
register();
}
});
coDesktopPane = new JDesktopPane();
coDesktopPane.setBackground(SystemColor.scrollbar);
coDesktopPane.setBounds(218, 278, 296, 117);
signInPanel.add(coDesktopPane);
corporateCheckBox = new JCheckBox("As Corporate");
corporateCheckBox.setBounds(12, 278, 115, 27);
```

```
signInPanel.add(corporateCheckBox);
JButton btnCorporateButton = new JButton("Add Company Details");
btnCorporateButton.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
if(corporateCheckBox.isSelected()) {
coDesktopPane.removeAll();
CorporateBox corporate = new CorporateBox();
corporate.setVisible(true);
coDesktopPane.add(corporate);
}
}
});
btnCorporateButton.setBounds(12, 317, 179, 27);
signInPanel.add(btnCorporateButton);
JLabel siConfirmPasswordTextField = new JLabel("Confirm Password");
siConfirmPasswordTextField.setFont(new Font("Dialog", Font.BOLD, 18));
siConfirmPasswordTextField.setBounds(299, 197, 179, 17);
signInPanel.add(siConfirmPasswordTextField);
siConfirmPasswordPassField = new JPasswordField();
siConfirmPasswordPassField.setColumns(10);
siConfirmPasswordPassField.setBounds(299, 225, 215, 31);
signInPanel.add(siConfirmPasswordPassField);
}
private void register() {
// On click of the save button
// Read data from the fields and store it in the model
// Create an object of Business Layer and pass the model to business layer
// Perform the required action from the business layer.
try {
Users user = new Users();
user.setEmail(siEmailTextField.getText());
user.setUserName(siUsersNameTextField.getText());
user.setPassword(String.valueOf(siPasswordPassField.getPassword()));
```

```
user.setConfirmPassword(String.valueOf(siConfirmPasswordPassField.getPassword()
));
Customer customer = new Customer();
customer.setFirstName(siFirstTextField.getText());
customer.setLastName(siLastTextField.getText());
UserServiceLayer userSL = new UserServiceLayer();
CustomerServiceLayer customerSL = new CustomerServiceLayer();
CorporateServiceLayer corporateSL = new CorporateServiceLayer();
if(corporateCheckBox.isSelected()) {
Corporate corporate = new Corporate();
corporate.setCompanyName(CorporateBox.CCompanyNameTextField.getText());
corporate.setCompanyContact(CorporateBox.CCompanyContactTextField.getText());
if(userSL.ValidateSignup(user)  &&  customerSL.ValidateCustomer(customer)  &&
corporateSL.ValidateCorporate(corporate)) {
user = userSL.userSave(user);
customer = customerSL.customerSave(customer);
corporate = corporateSL.corporateSave(corporate);
}
}
else {
if(userSL.ValidateSignup(user) && customerSL.ValidateCustomer(customer)) {
user = userSL.userSave(user);
customer = customerSL.customerSave(customer);
}
}
}
catch(InputException ex) {
JOptionPane.showMessageDialog(null, ex.getMessage());
}
catch(Exception ex) {
JOptionPane.showMessageDialog(null,"Complete your information");
}
}
}
```

```
package FrontendLayer;

import java.awt.event.MouseListener;
import java.util.ArrayList;
import javax.swing.JInternalFrame;
import javax.swing.JOptionPane;
import javax.swing.JTable;
import javax.swing.plaf.basic.BasicInternalFrameUI;
import javax.swing.table.DefaultTableModel;
import Models.Room;
import ServiceLayer.RoomServiceLayer;
import javax.swing.JScrollPane;
public class RoomDetailsTable extends JInternalFrame {
private JTable roomtable;
private DefaultTableModel model;
/**
* Create the frame.
*/
public RoomDetailsTable() {
setClosable(true);
setBounds(0, 0, 704, 436);
getContentPane().setLayout(null);
BasicInternalFrameUI basicInternalFrameUI =
((javax.swing.plaf.basic.BasicInternalFrameUI)this.getUI());
for(MouseListener listener:
basicInternalFrameUI.getNorthPane().getMouseListeners()){
basicInternalFrameUI.getNorthPane().removeMouseListener(listener);
}
JScrollPane scrollPane = new JScrollPane();
scrollPane.setBounds(12, 67, 670, 323);
getContentPane().add(scrollPane);
roomtable = new JTable();
scrollPane.setViewportView(roomtable);
model = new DefaultTableModel();
```

```
Object[] columnsName = new Object[5];
columnsName[0] = "Room no";
columnsName[1] = "Room Type";
columnsName[2] = "Price Per Night";
columnsName[3] = "Details";
columnsName[4] = "Room Status";
model.setColumnCount(0);
model.setColumnIdentifiers(columnsName);
}
public void loadAllRoom() {
try {
RoomServiceLayer rSL = new RoomServiceLayer();
ArrayList<Room> room = rSL.getAllRoom();
setTableRoomData(room);
} catch (Exception e1) {
JOptionPane.showMessageDialog(null, e1.getMessage());
}
}
private void setTableRoomData(ArrayList<Room> room) {
// Create the object array from arraylist and add to the table row
Object[] rowData = new Object[5];
// set the number of rows in table model to zero
model.setRowCount(0);
for(int i=0; i<room.size(); i++) {
rowData[0] = room.get(i).getRoomNo();
rowData[1] = room.get(i).getRoomType();
rowData[2] = room.get(i).getPricePerNight();
rowData[3] = room.get(i).getRoomDetails();
rowData[4] = room.get(i).getRoomStatus();
model.addRow(rowData);
}
roomtable.setModel(model);
}
}
```

RoomAssignBox.java

```
package FrontendLayer;


import javax.swing.JInternalFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;

import java.awt.Font;
import java.sql.Connection;
import java.sql.PreparedStatement;

import javax.swing.JTextField;
import javax.swing.plaf.basic.BasicInternalFrameUI;

import Helper.DatabaseConnector;
import Helper.InputException;
import Models.DefultModel;
import Models.Users;
import ServiceLayer.RoomAssignServiceLayer;
import ServiceLayer.RoomServiceLayer;
import ServiceLayer.UserServiceLayer;

import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.MouseListener;
import java.awt.event.ActionEvent;

public class RoomAssignBox extends JInternalFrame {
private JTextField customerTextBox;
```

```
private JTextField roomNoTextField;
private DatabaseConnector db;
private Connection connection;
private DefultModel defultModel;


/**
* Create the frame.
*/
public RoomAssignBox() {
setBounds(0, 0, 405,372);
getContentPane().setLayout(null);
BasicInternalFrameUI basicInternalFrameUI =
((javax.swing.plaf.basic.BasicInternalFrameUI)this.getUI());
for(MouseListener listener:
basicInternalFrameUI.getNorthPane().getMouseListeners()){
basicInternalFrameUI.getNorthPane().removeMouseListener(listener);
}
JLabel addRoomtitle = new JLabel("Booking Manager");
addRoomtitle.setFont(new Font("Dialog", Font.BOLD, 24));
addRoomtitle.setBounds(84, 12, 232, 33);
getContentPane().add(addRoomtitle);
JLabel pricePerNightLabel = new JLabel("Booking ID");
pricePerNightLabel.setFont(new Font("Dialog", Font.BOLD, 18));
pricePerNightLabel.setBounds(12, 83, 159, 17);
getContentPane().add(pricePerNightLabel);
customerTextBox = new JTextField();
customerTextBox.setColumns(10);
customerTextBox.setBounds(12, 112, 159, 31);
getContentPane().add(customerTextBox);
JLabel pricePerNightLabel_1 = new JLabel("Room No");
pricePerNightLabel_1.setFont(new Font("Dialog", Font.BOLD, 18));
pricePerNightLabel_1.setBounds(12, 155, 159, 17);
getContentPane().add(pricePerNightLabel_1);
```

```
roomNoTextField = new JTextField();
roomNoTextField.setColumns(10);
roomNoTextField.setBounds(12, 184, 159, 31);
getContentPane().add(roomNoTextField);
JButton assignButton = new JButton("Assign Room");
assignButton.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
assignRoom();
}
});
assignButton.setBounds(12, 259, 147, 27);
getContentPane().add(assignButton);
JButton btnNewButton_1 = new JButton("Cancel Booking");
btnNewButton_1.setBounds(171, 259, 133, 27);
getContentPane().add(btnNewButton_1);

}
private void assignRoom() {
try {
defultModel = new DefultModel();
defultModel.setBookingID(Integer.parseInt(customerTextBox.getText()));
defultModel.setRoomNo(roomNoTextField.getText());
RoomAssignServiceLayer roomAssignSL = new RoomAssignServiceLayer();
roomAssignSL.roomAssign(defultModel);
}
catch(InputException ex) {
JOptionPane.showMessageDialog(null, ex.getMessage());
}
catch(Exception ex) {
JOptionPane.showMessageDialog(null, ex.getMessage());
}
}
}
```

ProfilePage.java

```
package FrontendLayer;



import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import javax.swing.border.LineBorder;
import javax.swing.table.DefaultTableModel;

import DatabaseLayer.UserDatabaseLayer;
import Models.DefultModel;
import ServiceLayer.BookingDetailsServiceLayer;
import ServiceLayer.PaymentDetailsServiceLayer;

import java.awt.Color;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.ImageIcon;
import javax.swing.SwingConstants;
import java.awt.Font;
import java.util.ArrayList;

import javax.swing.JButton;
import javax.swing.JDesktopPane;
import javax.swing.JTable;
import javax.swing.JScrollPane;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

public class ProfilePage extends JFrame {
```

```
private JPanel contentPane;
private JTable userBookingTable;
private DefaultTableModel model;
private DefaultTableModel paymentmodel;
private JTable paymentTable;

public ProfilePage() {
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setBounds(0, 0, 1920,1080);
contentPane = new JPanel();
contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
setContentPane(contentPane);
contentPane.setLayout(null);
JPanel profilePanel = new JPanel();
profilePanel.setBorder(new LineBorder(new Color(192, 192, 192), 2, true));
profilePanel.setBounds(51, 62, 465, 490);
contentPane.add(profilePanel);
profilePanel.setLayout(null);
JLabel profilePictureLable = new JLabel("");
profilePictureLable.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment                    and
Feedback/Eclipse/hotelLutonApplication/Img/LOGO.png"));
profilePictureLable.setBounds(161, 30, 135, 105);
profilePanel.add(profilePictureLable);
JLabel userNameLabel = new JLabel(UserDatabaseLayer.uName);
userNameLabel.setForeground(Color.GRAY);
userNameLabel.setFont(new Font("Dialog", Font.BOLD, 24));
userNameLabel.setHorizontalAlignment(SwingConstants.CENTER);
userNameLabel.setBounds(132, 130, 189, 37);
profilePanel.add(userNameLabel);
JLabel nameLable = new JLabel("Name :");
nameLable.setFont(new Font("Dialog", Font.BOLD, 18));
nameLable.setBounds(34, 185, 82, 35);
profilePanel.add(nameLable);
```

```
JLabel emailLabel = new JLabel("Email :");
emailLabel.setFont(new Font("Dialog", Font.BOLD, 18));
emailLabel.setBounds(34, 232, 82, 23);
profilePanel.add(emailLabel);
JLabel companyNameLabel = new JLabel("Company name :");
companyNameLabel.setFont(new Font("Dialog", Font.BOLD, 18));
companyNameLabel.setBounds(34, 279, 168, 23);
profilePanel.add(companyNameLabel);
JLabel companyContactLebel = new JLabel("Company Contact :");
companyContactLebel.setFont(new Font("Dialog", Font.BOLD, 18));
companyContactLebel.setBounds(34, 326, 189, 23);
profilePanel.add(companyContactLebel);
JLabel cusNamelabel = new JLabel(UserDatabaseLayer.nameConcat);
cusNamelabel.setFont(new Font("Dialog", Font.BOLD, 15));
cusNamelabel.setForeground(Color.GRAY);
cusNamelabel.setBounds(112, 195, 270, 17);
profilePanel.add(cusNamelabel);
JLabel cusEmailLabel = new JLabel(UserDatabaseLayer.userEmail);
cusEmailLabel.setForeground(Color.GRAY);
cusEmailLabel.setFont(new Font("Dialog", Font.BOLD, 15));
cusEmailLabel.setBounds(112, 236, 270, 17);
profilePanel.add(cusEmailLabel);
JLabel corpNameLabel = new JLabel(UserDatabaseLayer.corpName);
corpNameLabel.setForeground(Color.GRAY);
corpNameLabel.setFont(new Font("Dialog", Font.BOLD, 15));
corpNameLabel.setBounds(195, 283, 258, 17);
profilePanel.add(corpNameLabel);
JLabel corpContactLabel = new JLabel(UserDatabaseLayer.corpcontact);
corpContactLabel.setForeground(Color.GRAY);
corpContactLabel.setFont(new Font("Dialog", Font.BOLD, 15));
corpContactLabel.setBounds(216, 330, 237, 17);
profilePanel.add(corpContactLabel);
JDesktopPane lowerDesktopPane = new JDesktopPane();
lowerDesktopPane.setBorder(new LineBorder(new Color(0, 0, 0), 3));
```

```
lowerDesktopPane.setBounds(597, 315, 595, 237);
contentPane.add(lowerDesktopPane);
lowerDesktopPane.setLayout(null);
JScrollPane lowerScrollPane = new JScrollPane();
lowerScrollPane.setBounds(12, 73, 568, 152);
lowerDesktopPane.add(lowerScrollPane);
paymentTable = new JTable();
lowerScrollPane.setViewportView(paymentTable);
JLabel lblPayment = new JLabel("Generated Bill");
lblPayment.setFont(new Font("Dialog", Font.BOLD, 16));
lblPayment.setBounds(219, 23, 197, 23);
lowerDesktopPane.add(lblPayment);
JButton btnEditProfile = new JButton("Edit Profile");
btnEditProfile.setBounds(34, 374, 105, 27);
profilePanel.add(btnEditProfile);
JButton btnGenerateBiil = new JButton("Generate Biil");
btnGenerateBiil.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
loadAllPayment();
}
});
btnGenerateBiil.setBounds(161, 374, 135, 27);
profilePanel.add(btnGenerateBiil);
JButton btnCancelBooking = new JButton("Cancel Booking");
btnCancelBooking.setBounds(216, 428, 135, 27);
profilePanel.add(btnCancelBooking);
JButton btnEditBookingDetails = new JButton("Edit Booking Details");
btnEditBookingDetails.setBounds(34, 428, 168, 27);
profilePanel.add(btnEditBookingDetails);
JDesktopPane upperDesktopPane = new JDesktopPane();
upperDesktopPane.setBorder(new LineBorder(new Color(0, 0, 0), 3));
upperDesktopPane.setBounds(597, 62, 595, 214);
contentPane.add(upperDesktopPane);
upperDesktopPane.setLayout(null);
```

```
JScrollPane scrollPane = new JScrollPane();
scrollPane.setBounds(12, 50, 568, 152);
upperDesktopPane.add(scrollPane);
userBookingTable = new JTable();
scrollPane.setViewportView(userBookingTable);
JLabel lblBookingHistory = new JLabel("Booking History");
lblBookingHistory.setFont(new Font("Dialog", Font.BOLD, 16));
lblBookingHistory.setBounds(218, 12, 142, 23);
upperDesktopPane.add(lblBookingHistory);
JButton btnBack = new JButton("Back");
btnBack.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
UserHomePage up = new UserHomePage();
up.setVisible(true);
dispose();
}
});
btnBack.setBounds(51, 23, 62, 27);
btnBack.setFocusable(false);
contentPane.add(btnBack);
JButton btnLogOut = new JButton("Log Out");
btnLogOut.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
WelcomePage wel = new WelcomePage();
wel.setVisible(true);
dispose();
}
});
btnLogOut.setFocusable(false);
btnLogOut.setBounds(142, 23, 82, 27);
contentPane.add(btnLogOut);
model = new DefaultTableModel();
Object[] columnsName = new Object[9];
columnsName[0] = "Booking ID";
```

```
columnsName[1] = "Name";
columnsName[2] = "Credit Number";
columnsName[3] = "Prefered Room";
columnsName[4] = "Check In date";
columnsName[5] = "Check Out date";
columnsName[6] = "Booking Status";
columnsName[7] = "Room No";
columnsName[8] = "Payment Status";
model.setColumnCount(0);
model.setColumnIdentifiers(columnsName);
paymentmodel = new DefaultTableModel();
Object[] columnspayment = new Object[3];
columnspayment[0] = "Booking ID";
columnspayment[1] = "Total Bill";
columnspayment[2] = "Payment Status";
paymentmodel.setColumnCount(0);
paymentmodel.setColumnIdentifiers(columnspayment);
}
public void loadAllUserbooking() {
try {
BookingDetailsServiceLayer bookingSL = new BookingDetailsServiceLayer();
ArrayList<DefultModel> defultModel = bookingSL.getAllUserBooking();
setTableBookingData(defultModel);
} catch (Exception e1) {
JOptionPane.showMessageDialog(null, e1.getMessage());
}
}
private void setTableBookingData(ArrayList<DefultModel> defultModel) {
// Create the object array from arraylist and add to the table row
Object[] rowData = new Object[9];
// set the number of rows in table model to zero
model.setRowCount(0);
for(int i=0; i<defultModel.size(); i++) {
rowData[0] = defultModel.get(i).getBookingID();
```

```
rowData[1] = defultModel.get(i).getName();
rowData[2] = defultModel.get(i).getCardNumber();
rowData[3] = defultModel.get(i).getRoomPreference();
rowData[4] = defultModel.get(i).getCheckInDate();
rowData[5] = defultModel.get(i).getCheckOutDate();
rowData[6] = defultModel.get(i).getBookingStatus();
rowData[7] = defultModel.get(i).getRoomNo();
rowData[8] = defultModel.get(i).getPaymentStatus();
model.addRow(rowData);
}
userBookingTable.setModel(model);

}
public void loadAllPayment() {
try {
PaymentDetailsServiceLayer paymentSL = new PaymentDetailsServiceLayer();
ArrayList<DefultModel> defultModel = paymentSL.getAllPayment();
setTablePaymentData(defultModel);
} catch (Exception e1) {
JOptionPane.showMessageDialog(null, e1.getMessage());
}
}
private void setTablePaymentData(ArrayList<DefultModel> defultModel) {
// Create the object array from arraylist and add to the table row
Object[] rowData = new Object[3];
// set the number of rows in table model to zero
paymentmodel.setRowCount(0);
for(int i=0; i<defultModel.size(); i++) {
rowData[0] = defultModel.get(i).getBookingID();
rowData[1] = defultModel.get(i).getTotalBill();
rowData[2] = defultModel.get(i).getPaymentStatus();
paymentmodel.addRow(rowData);
}
paymentTable.setModel(paymentmodel);
```

```
}
}
```

PaymentDetailsBox.java

```
package FrontendLayer;

import java.awt.event.MouseListener;

import java.util.ArrayList;

import javax.swing.JInternalFrame;
import javax.swing.JOptionPane;
import javax.swing.JTable;
import javax.swing.plaf.basic.BasicInternalFrameUI;
import javax.swing.table.DefaultTableModel;

import Models.DefultModel;
import ServiceLayer.BookingDetailsServiceLayer;
import ServiceLayer.PaymentDetailsServiceLayer;
import ServiceLayer.RoomServiceLayer;


import javax.swing.JScrollPane;

public class PaymentDetailsBox extends JInternalFrame {
private JTable paymentTable;
private DefaultTableModel model;
/**
* Create the frame.
*/
public PaymentDetailsBox() {
setClosable(true);
```

```
setBounds(0, 0, 704, 436);
getContentPane().setLayout(null);
BasicInternalFrameUI basicInternalFrameUI =
((javax.swing.plaf.basic.BasicInternalFrameUI)this.getUI());
for(MouseListener listener:
basicInternalFrameUI.getNorthPane().getMouseListeners()){
basicInternalFrameUI.getNorthPane().removeMouseListener(listener);
}
JScrollPane scrollPane = new JScrollPane();
scrollPane.setBounds(12, 67, 670, 323);
getContentPane().add(scrollPane);
paymentTable = new JTable();
scrollPane.setViewportView(paymentTable);
model = new DefaultTableModel();
Object[] columnsName = new Object[7];
columnsName[0] = "Booking ID";
columnsName[1] = "Guest Name";
columnsName[2] = "Guest Contact";
columnsName[3] = "Room No";
columnsName[4] = "Room Price";
columnsName[5] = "Total Bill";
columnsName[6] = "Payment Status";
model.setColumnCount(0);
model.setColumnIdentifiers(columnsName);

}
public void loadAllPayment() {
try {
PaymentDetailsServiceLayer paymentSL = new PaymentDetailsServiceLayer();
ArrayList<DefultModel> defultModel = paymentSL.getAllPayment();
setTableBookingData(defultModel);
} catch (Exception e1) {
JOptionPane.showMessageDialog(null, e1.getMessage());
}
```

```
}
private void setTableBookingData(ArrayList<DefultModel> defultModel) {
// Create the object array from arraylist and add to the table row
Object[] rowData = new Object[8];
// set the number of rows in table model to zero
model.setRowCount(0);
for(int i=0; i<defultModel.size(); i++) {
rowData[0] = defultModel.get(i).getBookingID();
rowData[1] = defultModel.get(i).getGuestName();
rowData[2] = defultModel.get(i).getContact();
rowData[3] = defultModel.get(i).getRoomNo();
rowData[4] = defultModel.get(i).getPricePerNight();
rowData[5] = defultModel.get(i).getTotalBill();
rowData[6] = defultModel.get(i).getPaymentStatus();
model.addRow(rowData);
}
paymentTable.setModel(model);

}
}
```

CreateBillBox.java

```
package FrontendLayer;


import javax.swing.JInternalFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
```

```
import java.awt.Font;
import java.sql.Connection;
import java.sql.PreparedStatement;


import javax.swing.JTextField;
import javax.swing.plaf.basic.BasicInternalFrameUI;

import Helper.DatabaseConnector;
import Helper.InputException;
import Models.DefultModel;
import Models.Users;
import ServiceLayer.BillGenerateServiceLayer;
import ServiceLayer.RoomAssignServiceLayer;
import ServiceLayer.RoomServiceLayer;
import ServiceLayer.UserServiceLayer;

import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.MouseListener;
import java.awt.event.ActionEvent;

public class CreateBillBox extends JInternalFrame {
private JTextField customerTextBox;
private JTextField discountPerTextField;
private DatabaseConnector db;
private Connection connection;
private DefultModel defultModel;


/**
* Create the frame.
*/
```

```
public CreateBillBox() {
setBounds(0, 0, 405,372);
getContentPane().setLayout(null);
BasicInternalFrameUI basicInternalFrameUI =
((javax.swing.plaf.basic.BasicInternalFrameUI)this.getUI());
for(MouseListener listener:
basicInternalFrameUI.getNorthPane().getMouseListeners()){
basicInternalFrameUI.getNorthPane().removeMouseListener(listener);
}
JLabel addRoomtitle = new JLabel("Generate Bill");
addRoomtitle.setFont(new Font("Dialog", Font.BOLD, 24));
addRoomtitle.setBounds(111, 12, 232, 33);
getContentPane().add(addRoomtitle);
JLabel pricePerNightLabel = new JLabel("Booking ID");
pricePerNightLabel.setFont(new Font("Dialog", Font.BOLD, 18));
pricePerNightLabel.setBounds(12, 72, 159, 27);
getContentPane().add(pricePerNightLabel);
customerTextBox = new JTextField();
customerTextBox.setColumns(10);
customerTextBox.setBounds(12, 112, 159, 31);
getContentPane().add(customerTextBox);
JLabel discountPerLabel = new JLabel("Discount Percent");
discountPerLabel.setFont(new Font("Dialog", Font.BOLD, 18));
discountPerLabel.setBounds(12, 155, 159, 17);
getContentPane().add(discountPerLabel);
discountPerTextField = new JTextField();
discountPerTextField.setColumns(10);
discountPerTextField.setBounds(12, 184, 159, 31);
getContentPane().add(discountPerTextField);
JButton generateButton = new JButton("Generate Bill");
generateButton.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
generateBill();
}
```

```
});
generateButton.setBounds(12, 259, 147, 27);
getContentPane().add(generateButton);

}
private void generateBill() {
try {
defultModel = new DefultModel();
defultModel.setBookingID(Integer.parseInt(customerTextBox.getText()));
defultModel.setDiscountPer(discountPerTextField.getText());
BillGenerateServiceLayer generateBillSL = new BillGenerateServiceLayer();
generateBillSL.generateBill(defultModel);
}
catch(InputException ex) {
JOptionPane.showMessageDialog(null, ex.getMessage());
}
catch(Exception ex) {
JOptionPane.showMessageDialog(null, ex.getMessage());
}
}
}
```

CorporateBox.java

```
package FrontendLayer;

import java.awt.event.MouseListener;

import javax.swing.JInternalFrame;



import javax.swing.JLabel;
```

```
import javax.swing.JOptionPane;
import javax.swing.JTextField;
import javax.swing.plaf.basic.BasicInternalFrameUI;

import Helper.InputException;
import Models.Corporate;
import Models.Users;
import ServiceLayer.CorporateServiceLayer;




@SuppressWarnings("serial")
public class CorporateBox extends JInternalFrame {
static JTextField CCompanyNameTextField;
static JTextField CCompanyContactTextField;
/**
* Launch the application.
*/

/**
* Create the frame.
*/
public CorporateBox() {
setTitle("Company Detail");
setClosable(true);
setBounds(0, 0, 296, 117);
getContentPane().setLayout(null);
BasicInternalFrameUI basicInternalFrameUI =
((javax.swing.plaf.basic.BasicInternalFrameUI)this.getUI());
for(MouseListener listener:
basicInternalFrameUI.getNorthPane().getMouseListeners()){
basicInternalFrameUI.getNorthPane().removeMouseListener(listener);
}
```

```
JLabel CCompanyNameLabel = new JLabel("Company Name");
CCompanyNameLabel.setBounds(12, 12, 96, 17);
getContentPane().add(CCompanyNameLabel);
JLabel CCompanyContactLabel = new JLabel("Company Contact");
CCompanyContactLabel.setBounds(12, 54, 116, 17);
getContentPane().add(CCompanyContactLabel);
CCompanyNameTextField = new JTextField();
CCompanyNameTextField.setBounds(129, 10, 145, 21);
getContentPane().add(CCompanyNameTextField);
CCompanyNameTextField.setColumns(10);
CCompanyContactTextField = new JTextField();
CCompanyContactTextField.setColumns(10);
CCompanyContactTextField.setBounds(129, 52, 145, 21);
getContentPane().add(CCompanyContactTextField);

}
}
```

CardDetailsBox.java

```
package FrontendLayer;

import java.awt.EventQueue;
import java.awt.event.MouseListener;

import javax.swing.JInternalFrame;
import javax.swing.JLabel;
import javax.swing.JTextField;
import javax.swing.plaf.basic.BasicInternalFrameUI;
import java.awt.BorderLayout;
import javax.swing.JButton;

public class CardDetailBox extends JInternalFrame {
static JTextField cardTypeTextField;
```

```
static JTextField nameOnCardTextField;
static JTextField cardNumberTextField;

public CardDetailBox() {
setBounds(0, 0, 316, 316);
setTitle("Credit Card Details");
BasicInternalFrameUI basicInternalFrameUI =
((javax.swing.plaf.basic.BasicInternalFrameUI)this.getUI());
for(MouseListener listener:
basicInternalFrameUI.getNorthPane().getMouseListeners()){
basicInternalFrameUI.getNorthPane().removeMouseListener(listener);
}
getContentPane().setLayout(null);
JLabel cardTypeLabel = new JLabel("Credit Card Type");
cardTypeLabel.setBounds(22, 12, 126, 17);
getContentPane().add(cardTypeLabel);
JLabel nameOnCardLabel = new JLabel("Name On Card");
nameOnCardLabel.setBounds(22, 74, 126, 34);
getContentPane().add(nameOnCardLabel);
JLabel cardNumberLabel= new JLabel("Card Number");
cardNumberLabel.setBounds(22, 146, 126, 34);
getContentPane().add(cardNumberLabel);
cardTypeTextField = new JTextField();
cardTypeTextField.setBounds(22, 41, 249, 21);
getContentPane().add(cardTypeTextField);
cardTypeTextField.setColumns(10);
nameOnCardTextField = new JTextField();
nameOnCardTextField.setColumns(10);
nameOnCardTextField.setBounds(22, 113, 249, 21);
getContentPane().add(nameOnCardTextField);
cardNumberTextField = new JTextField();
cardNumberTextField.setColumns(10);
cardNumberTextField.setBounds(22, 189, 249, 21);
getContentPane().add(cardNumberTextField);
```

```
JButton btnAddExistingDetails = new JButton("Add Existing Details");
btnAddExistingDetails.setBounds(22, 232, 155, 27);
getContentPane().add(btnAddExistingDetails);
}
}
```

BookingFormPage.java

```
package FrontendLayer;

import java.awt.BorderLayout;

import java.awt.EventQueue;

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.ImageIcon;
import javax.swing.UIManager;
import java.awt.Font;
import java.awt.Color;
import javax.swing.SwingConstants;
import javax.swing.border.LineBorder;

import DatabaseLayer.UserDatabaseLayer;

import javax.swing.JTextField;
import javax.swing.JButton;
import javax.swing.JDesktopPane;
import java.awt.event.ActionListener;

import java.awt.event.ActionEvent;
```

```
import Helper.InputException;
import Models.CreditCard;
import Models.GuestData;
import Models.Reservation;
import ServiceLayer.CreditServiceLayer;
import ServiceLayer.GuestServiceLayer;
import ServiceLayer.ReserveServiceLayer;

import javax.swing.JCheckBox;

public class BookingFormPage extends JFrame {

private JPanel contentPane;
private JTextField fullNameTextField;
private JTextField contactTextField;
private JTextField roomPreferenceTextField;
private JTextField countryTextField;
private JTextField cityTextField;
private JTextField stateTextField;
private JTextField dOBTextField;
private JTextField checkInDateTextField;
private JTextField checkOutDateTextField;
private JCheckBox creditCardChckbx;

/**
* Launch the application.
*/
public BookingFormPage() {
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setBounds(100, 100, 1920, 1080);
contentPane = new JPanel();
contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
setContentPane(contentPane);
```

```
contentPane.setLayout(null);
JLabel logoLabel = new JLabel("");
logoLabel.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment                    and
Feedback/Eclipse/hotelLutonApplication/Img/LOGO.png"));
logoLabel.setBounds(12, 0, 135, 115);
contentPane.add(logoLabel);
JLabel lblReservationFom = new JLabel("Reservation Form");
lblReservationFom.setHorizontalAlignment(SwingConstants.CENTER);
lblReservationFom.setForeground(Color.BLACK);
lblReservationFom.setFont(new Font("Dialog", Font.BOLD, 32));
lblReservationFom.setBackground(UIManager.getColor("Button.select"));
lblReservationFom.setBounds(411, 21, 570, 69);
contentPane.add(lblReservationFom);
JPanel detailsPanel = new JPanel();
detailsPanel.setBorder(new LineBorder(new Color(0, 0, 0), 3));
detailsPanel.setBounds(375, 92, 636, 546);
contentPane.add(detailsPanel);
detailsPanel.setLayout(null);
JLabel lblGuestDetails = new JLabel("Guest Details");
lblGuestDetails.setBounds(260, 33, 120, 26);
lblGuestDetails.setFont(new Font("Dialog", Font.BOLD, 18));
detailsPanel.add(lblGuestDetails);
JLabel fullnameLabel = new JLabel("Full Name");
fullnameLabel.setFont(new Font("Dialog", Font.BOLD, 18));
fullnameLabel.setBounds(26, 95, 179, 17);
detailsPanel.add(fullnameLabel);
fullNameTextField = new JTextField();
fullNameTextField.setColumns(10);
fullNameTextField.setBounds(26, 124, 215, 31);
detailsPanel.add(fullNameTextField);
JLabel birthDateLabel = new JLabel("Date Of Birth");
birthDateLabel.setFont(new Font("Dialog", Font.BOLD, 18));
birthDateLabel.setBounds(397, 95, 179, 17);
```

```
detailsPanel.add(birthDateLabel);
JLabel contactLabel = new JLabel("Contact");
contactLabel.setFont(new Font("Dialog", Font.BOLD, 18));
contactLabel.setBounds(26, 177, 179, 17);
detailsPanel.add(contactLabel);
contactTextField = new JTextField();
contactTextField.setColumns(10);
contactTextField.setBounds(26, 206, 215, 31);
detailsPanel.add(contactTextField);
JLabel roomPrefLabel = new JLabel("Room Preference");
roomPrefLabel.setFont(new Font("Dialog", Font.BOLD, 18));
roomPrefLabel.setBounds(397, 177, 179, 17);
detailsPanel.add(roomPrefLabel);
roomPreferenceTextField = new JTextField();
roomPreferenceTextField.setColumns(10);
roomPreferenceTextField.setBounds(397, 206, 215, 31);
detailsPanel.add(roomPreferenceTextField);
JLabel countryLabel = new JLabel("Country");
countryLabel.setFont(new Font("Dialog", Font.BOLD, 18));
countryLabel.setBounds(26, 260, 179, 17);
detailsPanel.add(countryLabel);
JLabel stateLabel = new JLabel("State");
stateLabel.setFont(new Font("Dialog", Font.BOLD, 18));
stateLabel.setBounds(250, 260, 179, 17);
detailsPanel.add(stateLabel);
JLabel cityLabel = new JLabel("City");
cityLabel.setFont(new Font("Dialog", Font.BOLD, 18));
cityLabel.setBounds(457, 260, 131, 17);
detailsPanel.add(cityLabel);
countryTextField = new JTextField();
countryTextField.setColumns(10);
countryTextField.setBounds(26, 289, 153, 31);
detailsPanel.add(countryTextField);
cityTextField = new JTextField();
```

```
cityTextField.setColumns(10);
cityTextField.setBounds(457, 289, 153, 31);
detailsPanel.add(cityTextField);
stateTextField = new JTextField();
stateTextField.setColumns(10);
stateTextField.setBounds(239, 289, 153, 31);
detailsPanel.add(stateTextField);
JLabel checkInLabel = new JLabel("Check In Date");
checkInLabel.setFont(new Font("Dialog", Font.BOLD, 18));
checkInLabel.setBounds(26, 343, 179, 17);
detailsPanel.add(checkInLabel);
JLabel checkOutLabel = new JLabel("Check Out Date");
checkOutLabel.setFont(new Font("Dialog", Font.BOLD, 18));
checkOutLabel.setBounds(277, 343, 179, 17);
detailsPanel.add(checkOutLabel);
JButton btnBookNow = new JButton("Book Now");
btnBookNow.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
guestDetails();
}
});
btnBookNow.setForeground(Color.WHITE);
btnBookNow.setBackground(Color.RED);
btnBookNow.setBounds(229, 489, 198, 31);
detailsPanel.add(btnBookNow);
JDesktopPane creditDesktopPane = new JDesktopPane();
creditDesktopPane.setBounds(47, 322, 316, 316);
contentPane.add(creditDesktopPane);
JButton btnAdd = new JButton("ADD ");
btnAdd.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
if(creditCardChckbx.isSelected()) {
creditDesktopPane.removeAll();
CardDetailBox cardDetails = new CardDetailBox();
```

```
cardDetails.setVisible(true);
creditDesktopPane.add(cardDetails);
}
}
});
btnAdd.setBounds(26, 454, 129, 27);
detailsPanel.add(btnAdd);
dOBTextField = new JTextField();
dOBTextField.setColumns(10);
dOBTextField.setBounds(397, 124, 215, 31);
detailsPanel.add(dOBTextField);
checkInDateTextField = new JTextField();
checkInDateTextField.setColumns(10);
checkInDateTextField.setBounds(26, 372, 215, 31);
detailsPanel.add(checkInDateTextField);
checkOutDateTextField = new JTextField();
checkOutDateTextField.setColumns(10);
checkOutDateTextField.setBounds(277, 372, 215, 31);
detailsPanel.add(checkOutDateTextField);
creditCardChckbx = new JCheckBox("Credit Card Details");
creditCardChckbx.setBounds(26, 422, 153, 25);
detailsPanel.add(creditCardChckbx);
JButton btnBack = new JButton("Back");
btnBack.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
UserHomePage pg = new UserHomePage();
pg.setVisible(true);
dispose();
}
});
btnBack.setBounds(135, 270, 105, 27);
contentPane.add(btnBack);

}
```

```
private void guestDetails() {
// On click of the save button
// Read data from the fields and store it in the model
// Create an object of Business Layer and pass the model to business layer
// Perform the required action from the business layer.
try {
String bookingStatus = "Pending";
String paymentStatus = "Pending";
if(creditCardChckbx.isSelected()) {
GuestServiceLayer guestSL = new GuestServiceLayer();
ReserveServiceLayer reserveSL = new ReserveServiceLayer();
CreditServiceLayer creditSL = new CreditServiceLayer();
GuestData guest = new GuestData();
guest.setFullName(fullNameTextField.getText());
guest.setdOB(dOBTextField.getText());
guest.setContact(contactTextField.getText());
guest.setRoomPreference(roomPreferenceTextField.getText());
guest.setCountry(countryTextField.getText());
guest.setState(stateTextField.getText());
guest.setCity(cityTextField.getText());
guest.setCheckInDate(checkInDateTextField.getText());
guest.setCheckOutDate(checkOutDateTextField.getText());
Reservation reserve = new Reservation();
reserve.setBookingStatus(bookingStatus);
reserve.setPaymentStatus(paymentStatus);
CreditCard credit = new CreditCard();
credit.setCardType(CardDetailBox.cardTypeTextField.getText());
credit.setNameOnCard(CardDetailBox.nameOnCardTextField.getText());
credit.setCardNo(CardDetailBox.cardNumberTextField.getText());
if(guestSL.validateGuestDetails(guest) && creditSL.validateCreditDetails(credit)) {
guest = guestSL.guestSave(guest);
reserve = reserveSL.reservationSave(reserve);
credit = creditSL.creditSave(credit);
}
```

```
}
else {
GuestServiceLayer guestSL = new GuestServiceLayer();
ReserveServiceLayer reserveSL = new ReserveServiceLayer();
GuestData guest = new GuestData();
guest.setFullName(fullNameTextField.getText());
guest.setdOB(dOBTextField.getText());
guest.setContact(contactTextField.getText());
guest.setRoomPreference(roomPreferenceTextField.getText());
guest.setCountry(countryTextField.getText());
guest.setState(stateTextField.getText());
guest.setCity(cityTextField.getText());
guest.setCheckInDate(checkInDateTextField.getText());
guest.setCheckOutDate(checkOutDateTextField.getText());
Reservation reserve = new Reservation();
reserve.setBookingStatus(bookingStatus);
reserve.setPaymentStatus(paymentStatus);
if(guestSL.validateGuestDetails(guest)){
guest = guestSL.guestSave(guest);
reserve = reserveSL.reservationSave(reserve);
}
}
}
catch(InputException ex) {
JOptionPane.showMessageDialog(null, ex.getMessage());
}
catch(Exception ex) {
JOptionPane.showMessageDialog(null, ex.getMessage());
}
}
}
```

BookingDetailsTable.java

```
package FrontendLayer;

import java.awt.event.MouseListener;


import java.util.ArrayList;

import javax.swing.JInternalFrame;
import javax.swing.JOptionPane;
import javax.swing.JTable;
import javax.swing.plaf.basic.BasicInternalFrameUI;
import javax.swing.table.DefaultTableModel;

import Models.DefultModel;
import Models.Room;
import ServiceLayer.BookingDetailsServiceLayer;
import ServiceLayer.RoomServiceLayer;
import ServiceLayer.UserDetailsServiceLayer;

import javax.swing.JScrollPane;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

public class BookingDetailsTable extends JInternalFrame {
private JTable bookingtable;
private DefaultTableModel model;
/**
* Create the frame.
*/
public BookingDetailsTable() {
setClosable(true);
setBounds(0, 0, 704, 436);
getContentPane().setLayout(null);
```

```
BasicInternalFrameUI basicInternalFrameUI =
((javax.swing.plaf.basic.BasicInternalFrameUI)this.getUI());
for(MouseListener listener:
basicInternalFrameUI.getNorthPane().getMouseListeners()){
basicInternalFrameUI.getNorthPane().removeMouseListener(listener);
}
JScrollPane scrollPane = new JScrollPane();
scrollPane.setBounds(12, 67, 670, 323);
getContentPane().add(scrollPane);
bookingtable = new JTable();
scrollPane.setViewportView(bookingtable);
JButton btnDeleteBooking = new JButton("DELETE Booking");
btnDeleteBooking.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
if(bookingtable.getSelectionModel().isSelectionEmpty()) {
JOptionPane.showMessageDialog(null,"No Selected Item" );

}
int row = bookingtable.getSelectedRow();
DefaultTableModel model = (DefaultTableModel) bookingtable.getModel();
int reserveID = Integer.parseInt(model.getValueAt(row, 0).toString());

DefultModel defultmodel = new DefultModel();
defultmodel.setBookingID(reserveID);


BookingDetailsServiceLayer bookingDetailsSL = new
BookingDetailsServiceLayer();
try {
bookingDetailsSL.deleteBooking(defultmodel);
} catch (Exception e1) {
// TODO Auto-generated catch block
e1.printStackTrace();
}
```

```
}
});
btnDeleteBooking.setBounds(12, 31, 142, 27);
getContentPane().add(btnDeleteBooking);
model = new DefaultTableModel();
Object[] columnsName = new Object[11];
columnsName[0] = "Booking ID";
columnsName[1] = "Name";
columnsName[2] = "Corporate Name";
columnsName[3] = "Corporate Contact";
columnsName[4] = "Credit Number";
columnsName[5] = "Prefered Room";
columnsName[6] = "Country";
columnsName[7] = "Check In date";
columnsName[8] = "Check Out date";
columnsName[9] = "Booking Status";
columnsName[10] = "Room No";
model.setColumnCount(0);
model.setColumnIdentifiers(columnsName);

}
public void loadAllbooking() {
try {
BookingDetailsServiceLayer bookingSL = new BookingDetailsServiceLayer();
ArrayList<DefultModel> defultModel = bookingSL.getAllBooking();
setTableBookingData(defultModel);
} catch (Exception e1) {
JOptionPane.showMessageDialog(null, e1.getMessage());
}
}
private void setTableBookingData(ArrayList<DefultModel> defultModel) {
// Create the object array from arraylist and add to the table row
Object[] rowData = new Object[11];
// set the number of rows in table model to zero
```

```
model.setRowCount(0);
for(int i=0; i<defultModel.size(); i++) {
rowData[0] = defultModel.get(i).getBookingID();
rowData[1] = defultModel.get(i).getName();
rowData[2] = defultModel.get(i).getCorpName();
rowData[3] = defultModel.get(i).getCorpContact();
rowData[4] = defultModel.get(i).getCardNumber();
rowData[5] = defultModel.get(i).getRoomPreference();
rowData[6] = defultModel.get(i).getCountry();
rowData[7] = defultModel.get(i).getCheckInDate();
rowData[8] = defultModel.get(i).getCheckOutDate();
rowData[9] = defultModel.get(i).getBookingStatus();
rowData[10] = defultModel.get(i).getRoomNo();
model.addRow(rowData);
}
bookingtable.setModel(model);

}
}
```

AdminPage.java

```
package FrontendLayer;

import java.awt.Image;


import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import javax.swing.JLabel;
```

```
import java.awt.Font;
import javax.swing.border.LineBorder;

import DatabaseLayer.BookingDetailsDatabaseLayer;

import java.awt.Color;
import javax.swing.JButton;
import javax.swing.JTextField;
import javax.swing.JTable;
import java.awt.event.ActionListener;

import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JDesktopPane;

@SuppressWarnings("serial")
public class AdminPage extends JFrame {

private JPanel contentPane;


/**
* Create the frame.
*/
public AdminPage() {
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setBounds(100, 100, 1920,1080);
contentPane = new JPanel();
contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
setContentPane(contentPane);
contentPane.setLayout(null);
JLabel apLogoLabel = new JLabel("");
```

```
apLogoLabel.setIcon(new
ImageIcon("/home/vivu/Class_Stuff/Programming/Assignment and
Feedback/Eclipse/hotelLutonApplication/Img/LOGO2.png"));
apLogoLabel.setBounds(340, 12, 90, 80);
contentPane.add(apLogoLabel);
JLabel aplHotelLutonAdministration = new JLabel("HOTEL LUTON
ADMINISTRATION");
aplHotelLutonAdministration.setFont(new Font("Dialog", Font.BOLD, 28));
aplHotelLutonAdministration.setBounds(470, 25, 524, 56);
contentPane.add(aplHotelLutonAdministration);
JPanel upperButtonPanel = new JPanel();
upperButtonPanel.setBorder(new LineBorder(new Color(0, 0, 0), 3));
upperButtonPanel.setBounds(12, 120, 1255, 56);
contentPane.add(upperButtonPanel);
upperButtonPanel.setLayout(null);
JDesktopPane lowerDesktopPane = new JDesktopPane();
lowerDesktopPane.setBorder(new LineBorder(new Color(0, 0, 0), 3));
lowerDesktopPane.setBounds(12, 199, 704, 436);
contentPane.add(lowerDesktopPane);
JButton apBtnUsersDetails = new JButton("Users");
apBtnUsersDetails.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
lowerDesktopPane.removeAll();
UserDetailsTable userTable = new UserDetailsTable();
userTable.setVisible(true);
lowerDesktopPane.add(userTable);
userTable.loadAllUser();
}
});
apBtnUsersDetails.setBounds(12, 12, 116, 27);
upperButtonPanel.add(apBtnUsersDetails);
JButton btnRoomDetails = new JButton("Rooms");
btnRoomDetails.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
```

```
lowerDesktopPane.removeAll();
RoomDetailsTable roomTable = new RoomDetailsTable();
roomTable.setVisible(true);
lowerDesktopPane.add(roomTable);
roomTable.loadAllRoom();
}
});
btnRoomDetails.setBounds(155, 12, 123, 27);
upperButtonPanel.add(btnRoomDetails);
JButton btnBooking = new JButton("Booking ");
btnBooking.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
lowerDesktopPane.removeAll();
BookingDetailsTable bookingTable = new BookingDetailsTable();
bookingTable.setVisible(true);
lowerDesktopPane.add(bookingTable);
bookingTable.loadAllbooking();
}
});
btnBooking.setBounds(305, 12, 116, 27);
upperButtonPanel.add(btnBooking);
JButton btnPayment = new JButton("Payment");
btnPayment.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
PaymentDetailsBox paymentTable = new PaymentDetailsBox();
lowerDesktopPane.add(paymentTable);
paymentTable.setVisible(true);
paymentTable.loadAllPayment();
}
});
btnPayment.setBounds(450, 12, 105, 27);
upperButtonPanel.add(btnPayment);
JButton btnLogOut = new JButton("Log Out");
btnLogOut.addActionListener(new ActionListener() {
```

```
public void actionPerformed(ActionEvent arg0) {
WelcomePage wel = new WelcomePage();
wel.setVisible(true);
dispose();
}
});
btnLogOut.setBounds(1127, 12, 105, 27);
upperButtonPanel.add(btnLogOut);
JDesktopPane sideDesktopPane = new JDesktopPane();
sideDesktopPane.setBorder(new LineBorder(new Color(0, 0, 0), 3));
sideDesktopPane.setBounds(839, 263, 405, 372);
contentPane.add(sideDesktopPane);
JButton addRoomButton = new JButton("Add Rooms");
addRoomButton.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
sideDesktopPane.removeAll();
AddRoomPage addRoom = new AddRoomPage();
addRoom.setVisible(true);
sideDesktopPane.add(addRoom);
}
});
addRoomButton.setBounds(854, 224, 116, 27);
contentPane.add(addRoomButton);
JButton roomAsignButton = new JButton("Assign Room");
roomAsignButton.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
sideDesktopPane.removeAll();
RoomAssignBox assignBox = new RoomAssignBox();
sideDesktopPane.add(assignBox);
assignBox.setVisible(true);
}
});
roomAsignButton.setBounds(982, 224, 128, 27);
contentPane.add(roomAsignButton);
```

```
JButton createBillLable = new JButton("Create Bill");
createBillLable.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
sideDesktopPane.removeAll();
CreateBillBox createBill = new CreateBillBox();
sideDesktopPane.add(createBill);
createBill.setVisible(true);
}
});
createBillLable.setBounds(1122, 224, 105, 27);
contentPane.add(createBillLable);
}
}
```

AddRoomPage.java

```
package FrontendLayer;

import java.awt.EventQueue;


import javax.swing.JInternalFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;

import java.awt.Font;
import javax.swing.JTextField;
import javax.swing.plaf.basic.BasicInternalFrameUI;
import javax.swing.table.DefaultTableModel;
import Helper.InputException;
import Models.Room;
import Models.Users;
import ServiceLayer.RoomServiceLayer;
import ServiceLayer.UserServiceLayer;
```

```
import javax.swing.JComboBox;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.MouseListener;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;
import java.awt.event.ActionEvent;

public class AddRoomPage extends JInternalFrame {
private JTextField pricePerNightTextField;
private JTextField detailsTextField;
private JComboBox statusChooser;
private JComboBox roomTypeChooser;
private DefaultTableModel model;

/**
* Create the frame.
*/
public AddRoomPage() {
setClosable(true);
setBounds(0, 0, 405, 372);
getContentPane().setLayout(null);
BasicInternalFrameUI basicInternalFrameUI =
((javax.swing.plaf.basic.BasicInternalFrameUI)this.getUI());
for(MouseListener listener:
basicInternalFrameUI.getNorthPane().getMouseListeners()){
basicInternalFrameUI.getNorthPane().removeMouseListener(listener);
}
JLabel addRoomtitle = new JLabel("ADD ROOMS");
addRoomtitle.setFont(new Font("Dialog", Font.BOLD, 24));
addRoomtitle.setBounds(117, 12, 187, 25);
getContentPane().add(addRoomtitle);
```

```
JLabel roomTypeLabel = new JLabel("Room Type");
roomTypeLabel.setFont(new Font("Dialog", Font.BOLD, 18));
roomTypeLabel.setBounds(12, 71, 107, 17);
getContentPane().add(roomTypeLabel);
JLabel pricePerNightLabel = new JLabel("Price Per Night");
pricePerNightLabel.setFont(new Font("Dialog", Font.BOLD, 18));
pricePerNightLabel.setBounds(227, 71, 159, 17);
getContentPane().add(pricePerNightLabel);
String[] roomType = { "single Room", "Delux Room", "Double Room" };
roomTypeChooser = new JComboBox(roomType);
roomTypeChooser.setBounds(12, 104, 159, 26);
getContentPane().add(roomTypeChooser);
pricePerNightTextField = new JTextField();
pricePerNightTextField.setColumns(10);
pricePerNightTextField.setBounds(227, 102, 159, 31);
getContentPane().add(pricePerNightTextField);
JLabel roomDetailsLabel = new JLabel("Room Details");
roomDetailsLabel.setFont(new Font("Dialog", Font.BOLD, 18));
roomDetailsLabel.setBounds(12, 155, 144, 17);
getContentPane().add(roomDetailsLabel);
JLabel statusLable = new JLabel("Status");
statusLable.setFont(new Font("Dialog", Font.BOLD, 18));
statusLable.setBounds(12, 234, 107, 17);
getContentPane().add(statusLable);
detailsTextField = new JTextField();
detailsTextField.setColumns(10);
detailsTextField.setBounds(12, 184, 374, 31);
getContentPane().add(detailsTextField);
String[] roomStatus = { "Available", "Reserved", "Booked" };
statusChooser = new JComboBox(roomStatus);
statusChooser.setBounds(12, 263, 208, 26);
getContentPane().add(statusChooser);
JButton btnAddRoom = new JButton("ADD ROOM");
btnAddRoom.addActionListener(new ActionListener() {
```

```
public void actionPerformed(ActionEvent arg0) {
saveRoom();
}
});
btnAddRoom.setBounds(262, 263, 105, 27);
getContentPane().add(btnAddRoom);
}
private void saveRoom() {
// On click of the save button
// Read data from the fields and store it in the model
// Create an object of Business Layer and pass the model to business layer
// Perform the required action from the business layer.
try {
Room room = new Room();
room.setRoomType(roomTypeChooser.getSelectedItem());
room.setPricePerNight(pricePerNightTextField.getText());
room.setRoomDetails(detailsTextField.getText());
room.setRoomStatus(statusChooser.getSelectedItem());
RoomServiceLayer roomSL = new RoomServiceLayer();
room = roomSL.roomSave(room);
}
catch(InputException ex) {
JOptionPane.showMessageDialog(null, ex.getMessage());
}
catch(Exception ex) {
JOptionPane.showMessageDialog(null, ex.getMessage());
}
}
}
```

Package Helper

DatabaseConnector.java

```
package Helper;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

// This is a Singleton class which creates only one instance object
public class DatabaseConnector {
private static DatabaseConnector dbInstance = null;
private String driverClass = null;
private String connectionString = null;
private String hostName = null;
private String portNumber = null;
private String databaseName = null;
private String dbUserName = null;
private String dbPassword = null;
// Database connections
private Connection conn = null;
private void setConnectionString() {
this.driverClass = "com.mysql.cj.jdbc.Driver";
this.hostName = "localhost";
this.portNumber = "3306";
this.databaseName = "luton_data";
this.dbUserName = "incri";
this.dbPassword = "fastrack";
this.connectionString =
"jdbc:mysql://"+this.hostName+":"+this.portNumber+"/"+this.databaseName;
}
private DatabaseConnector() throws Exception {
try {
this.setConnectionString();
Class.forName(this.driverClass);
```

```
this.conn = DriverManager.getConnection(this.connectionString, this.dbUserName,
this.dbPassword);
}catch(SQLException ex) {
Exception e = new Exception(ex.getMessage() + " SQL Error Occured");
throw e;
}catch(ClassNotFoundException ex) {
Exception e = new Exception(ex.getMessage() + " Class not found");
throw e;
}
}
public static DatabaseConnector getInstance() throws Exception {
if(dbInstance == null) {
try {
dbInstance = new DatabaseConnector();
}catch(Exception ex) {
throw ex;
}
}
return dbInstance;
}
public Connection getConnection() {
return this.conn;
}
}
```

InputException.java

Package Model

Administration.java

```
package Models;

public class Administration {
private int adminID;
private String contact;
private String firstName;
private String lastName;
public Administration() {
this.adminID = 0;
this.contact= "";
this.firstName = "";
this.lastName = "";
}
public Administration(int id, String contact, String firstName, String lastName ) {
this.adminID = id;
this.contact = contact;
this.firstName = firstName;
this.lastName = lastName;
}

public int getAdminID() {
return adminID;
}

public void setAdminID(int adminID) {
this.adminID = adminID;
}

public String getContact() {
return contact;
}

public void setContact(String contact) {
this.contact = contact;
```

```
}

public String getFirstName() {
return firstName;
}

public void setFirstName(String firstName) {
this.firstName = firstName;
}

public String getLastName() {
return lastName;
}

public void setLastName(String lastName) {
this.lastName = lastName;
}
}
```

Billing.java

```
package Models;

import java.util.Date;

public class Billing {
private int billingID;
private Date paidDate;
private String totalBill;
public Billing() {
this.billingID = 0;
this.paidDate = null;
this.totalBill = "";
}
```

```
public Billing(int billingID, Date paidDate, String totalBill ) {
this.billingID = billingID;
this.paidDate = paidDate;
this.totalBill = totalBill;
}

public int getBillingID() {
return billingID;
}

public void setBillingID(int billingID) {
this.billingID = billingID;
}

public Date getPaidDate() {
return paidDate;
}

public void setPaidDate(Date paidDate) {
this.paidDate = paidDate;
}

public String getTotalBill() {
return totalBill;
}

public void setTotalBill(String totalBill) {
this.totalBill = totalBill;
}
}
```

Corporate.java

```
package Models;
```

```
public class Corporate {
private int corpID;
private String companyName;
private String companyContact;
public Corporate() {
this.corpID = 0;
this.companyName = "";
this.companyContact = "";
}
public Corporate(int corpID, String companyName, String companyContact ) {
this.corpID = corpID;
this.companyName = companyName;
this.companyContact = companyContact;
}

public int getCorpID() {
return corpID;
}

public void setCorpID(int corpID) {
this.corpID = corpID;
}

public String getCompanyName() {
return companyName;
}

public void setCompanyName(String companyName) {
this.companyName = companyName;
}

public String getCompanyContact() {
return companyContact;
```

```
}

public void setCompanyContact(String companyContact) {
this.companyContact = companyContact;
}
}
```

CreditCard.java

```
package Models;

public class CreditCard {
private int cardID;
private String cardType;
private String cardNo;
private String nameOnCard;
public CreditCard() {
this.cardID = 0;
this.cardType = "";
this.cardNo = "";
this.nameOnCard= "";
}
public CreditCard(int cardID, String cardType, String cardNo, String nameOnCard ) {
this.cardID = cardID;
this.cardType = cardType;
this.cardNo = cardNo;
this.nameOnCard = nameOnCard;
}

public int getCardID() {
return cardID;
}

public void setCardID(int cardID) {
```

```
this.cardID = cardID;
}

public String getCardType() {
return cardType;
}

public void setCardType(String cardType) {
this.cardType = cardType;
}

public String getCardNo() {
return cardNo;
}

public void setCardNo(String cardNo) {
this.cardNo = cardNo;
}

public String getNameOnCard() {
return nameOnCard;
}

public void setNameOnCard(String nameOnCard) {
this.nameOnCard = nameOnCard;
}
}
```

Customer.java

```
package Models;
```

```
public class Customer {
private static int cusID;
private String firstName;
private String lastName;
public Customer() {
Customer.cusID = 0;
this.firstName = "";
this.lastName = "";
}
public Customer(int cusID, String firstName, String lastName) {
Customer.cusID = cusID;
this.firstName = firstName;
this.lastName = lastName;

}

public static int getCusID() {
return cusID;
}

public void setCusID(int cusID) {
Customer.cusID = cusID;
}

public String getFirstName() {
return firstName;
}

public void setFirstName(String firstName) {
this.firstName = firstName;
}

public String getLastName() {
return lastName;
```

```
}

public void setLastName(String lastName) {
this.lastName = lastName;
}

}
```

DefaultModel.java

```
package Models;

public class DefultModel {
private int userID;
private String cusID;
private String roomNo;
private int bookingID;
private String corpID;
private String guestID;
private String reserveID;
private int billID;
private String email;
private String name;
private String corpName;
private String corpContact;
private String guestName;
private String contact;
private String roomPreference;
private String country;
private String checkInDate;
private String checkOutDate;
private String cardType;
private String cardNumber;
private String nameOnCard;
```

```
private String bookingStatus;
private String paymentStatus;
private String totalBill;
private String pricePerNight;
private String discountPer;
public DefultModel() {
this.userID = 0;
this.cusID = "";
this.roomNo = "";
this.bookingID = 0;
this.corpID = "";
this.guestID = "";
this.reserveID = "";
this.billID = 0;
this.email = "";
this.name = "";
this.corpName = "";
this.corpContact = "";
this.guestName = "";
this.contact = "";
this.roomPreference = "";
this.country = "";
this.checkInDate = "";
this.checkOutDate = "";
this.cardType = "";
this.cardNumber = "";
this.nameOnCard = "";
this.bookingStatus = "";
this.paymentStatus = "";
this.totalBill = "";
this.pricePerNight = "";
this.discountPer = "";
}
```

```
public DefultModel(int userID, String cusID,String roomNo,
int bookingID, String guestID, String reserveID, int billID, String corpID,
String email, String name, String corpName, String corpContact, String guestName,
String contact,
String roomPreference, String country, String checkInDate, String checkOutDate,
String cardType,
String cardNumber, String nameOncard, String bookingStatus, String paymentStatus,
String totalBill, String nameOnCard,String pricePerNight, String discountPer ) {
this.userID = userID;
this.cusID = cusID;
this.roomNo = roomNo;
this.bookingID = bookingID;
this.corpID = corpID;
this.guestID = guestID;
this.reserveID = reserveID;
this.billID = billID;
this.email = email;
this.name = name;
this.corpName = corpName;
this.corpContact = corpContact ;
this.guestName = guestName ;
this.contact = contact;
this.roomPreference = roomPreference;
this.country = country;
this.checkInDate = checkInDate;
this.checkOutDate = checkOutDate;
this.cardType = cardType;
this.cardNumber = cardNumber;
this.nameOnCard = nameOnCard;
this.bookingStatus = bookingStatus;
this.paymentStatus = paymentStatus;
this.totalBill = totalBill;
this.pricePerNight = pricePerNight;
this.discountPer = discountPer;
```

```
}


public int getUserID() {
return userID;
}


public void setUserID(int userID) {
this.userID = userID;
}


public String getCusID() {
return cusID;
}


public void setCusID(String cusID) {
this.cusID = cusID;
}


public String getRoomNo() {
return roomNo;
}


public void setRoomNo(String roomNo) {
```

```
this.roomNo = roomNo;
}


public int getBookingID() {
return bookingID;
}


public void setBookingID(int reserveID2) {
this.bookingID = reserveID2;
}


public String getCorpID() {
return corpID;
}


public void setCorpID(String corpID) {
this.corpID = corpID;
}


public String getGuestID() {
return guestID;
}
```

```
public void setGuestID(String guestID) {
this.guestID = guestID;
}


public String getReserveID() {
return reserveID;
}


public void setReserveID(String reserveID) {
this.reserveID = reserveID;
}


public int getBillID() {
return billID;
}


public void setBillID(int reserveID2) {
this.billID = reserveID2;
}


public String getEmail() {
return email;
}
```

```
public void setEmail(String email) {
this.email = email;
}


public String getName() {
return name;
}


public void setName(String name) {
this.name = name;
}


public String getCorpName() {
return corpName;
}


public void setCorpName(String corpName) {
this.corpName = corpName;
}


public String getCorpContact() {
return corpContact;
}
```

```
public void setCorpContact(String corpContact) {
this.corpContact = corpContact;
}


public String getGuestName() {
return guestName;
}


public void setGuestName(String guestName) {
this.guestName = guestName;
}


public String getContact() {
return contact;
}


public void setContact(String contact) {
this.contact = contact;
}


public String getRoomPreference() {
return roomPreference;
}
```

```
public void setRoomPreference(String roomPreference) {
this.roomPreference = roomPreference;
}


public String getCountry() {
return country;
}


public void setCountry(String country) {
this.country = country;
}


public String getCheckInDate() {
return checkInDate;
}


public void setCheckInDate(String checkInDate) {
this.checkInDate = checkInDate;
}


public String getCheckOutDate() {
return checkOutDate;
}
```

```
public void setCheckOutDate(String checkOutDate) {
this.checkOutDate = checkOutDate;
}


public String getCardType() {
return cardType;
}


public void setCardType(String cardType) {
this.cardType = cardType;
}


public String getCardNumber() {
return cardNumber;
}


public void setCardNumber(String cardNumber) {
this.cardNumber = cardNumber;
}


public String getNameOnCard() {
return nameOnCard;
}
```

```
public void setNameOnCard(String nameOnCard) {
this.nameOnCard = nameOnCard;
}

public String getBookingStatus() {
return bookingStatus;
}

public void setBookingStatus(String bookingStatus) {
this.bookingStatus = bookingStatus;
}

public String getPaymentStatus() {
return paymentStatus;
}

public void setPaymentStatus(String paymentStatus) {
this.paymentStatus = paymentStatus;
}

public String getTotalBill() {
return totalBill;
}
```

```
public void setTotalBill(String totalBill) {
this.totalBill = totalBill;
}


public String getPricePerNight() {
return pricePerNight;
}


public void setPricePerNight(String pricePerNight) {
this.pricePerNight = pricePerNight;
}


public String getDiscountPer() {
return discountPer;
}


public void setDiscountPer(String discountPer) {
this.discountPer = discountPer;
}

}
```

GuestData.java

```
package Models;

import java.util.Date;
```

```
public class GuestData {
private int guestID;
private String fullName;
private String dOB;
private String country;
private String state;
private String city;
private String checkInDate;
private String checkOutDate;
private String contact;
private String roomPreference;

public GuestData() {
this.guestID = 0;
this.fullName = "";
this.dOB= "";
this.country = "";
this.state = "";
this.city= "";
this.checkInDate = "";
this.checkOutDate= "";
this.contact = "";
this.roomPreference= "";
}
public GuestData(int guestID, String fullName, String dOB,
String country, String state, String city,
String checkInDate, String checkOutDate, String contact, String roomPreference) {
this.guestID = guestID;
this.fullName= fullName;
this.dOB= dOB;
this.country = country;
this.state = state;
this.city= city;
```

```
this.checkInDate = checkInDate;
this.checkOutDate= checkOutDate;
this.contact = contact;
this.roomPreference= roomPreference;
}

public int getGuestID() {
return guestID;
}

public void setGuestID(int guestID) {
this.guestID = guestID;
}

public String getFullName() {
return fullName;
}

public void setFullName(String fullName) {
this.fullName = fullName;
}


public String getdOB() {
return dOB;
}

public void setdOB(String dOB) {
this.dOB = dOB;
}

public String getCountry() {
return country;
}
```

```
public void setCountry(String country) {
this.country = country;
}

public String getState() {
return state;
}

public void setState(String state) {
this.state = state;
}

public String getCity() {
return city;
}

public void setCity(String city) {
this.city = city;
}


public String getCheckInDate() {
return checkInDate;
}

public void setCheckInDate(String checkInDate) {
this.checkInDate = checkInDate;
}

public String getCheckOutDate() {
return checkOutDate;
}
```

```
public void setCheckOutDate(String checkOutDate) {
this.checkOutDate = checkOutDate;
}

public String getContact() {
return contact;
}

public void setContact(String contact) {
this.contact = contact;
}

public String getRoomPreference() {
return roomPreference;
}

public void setRoomPreference(String roomPreference) {
this.roomPreference = roomPreference;
}
}
```

Reservation.java

```
package Models;

public class Reservation {
private int reserveID;
private String bookingStatus;
private String paymentStatus;
private int noOfStayDay;
public Reservation() {
this.reserveID = 0;
this.bookingStatus = "";
this.paymentStatus= "";
```

```
this.noOfStayDay= 0;
}
public Reservation(int reserveID, String bookingStatus, String paymentStatus, int
noOfStayDay) {
this.reserveID = reserveID;
this.bookingStatus = bookingStatus;
this.paymentStatus= paymentStatus;
this.noOfStayDay= noOfStayDay;
}

public int getReserveID() {
return reserveID;
}

public void setReserveID(int reserveID) {
this.reserveID = reserveID;
}

public String getBookingStatus() {
return bookingStatus;
}

public void setBookingStatus(String bookingStatus) {
this.bookingStatus = bookingStatus;
}

public String getPaymentStatus() {
return paymentStatus;
}

public void setPaymentStatus(String paymentStatus) {
this.paymentStatus = paymentStatus;
}
```

```
public int getNoOfStayDay() {
return noOfStayDay;
}

public void setNoOfStayDay(int noOfStayDay) {
this.noOfStayDay = noOfStayDay;
}
}
```

Room.java

```
package Models;



public class Room {
private int roomNo;
private Object roomType;
private String pricePerNight;
private String roomDetails;
private Object roomStatus;
public Room() {
this.roomNo = 0;
this.roomType = "";
this.pricePerNight = "";
this.roomDetails = "";
this.roomStatus ="";
}
public Room(int roomNo, Object roomType, String pricePerNight, String
roomDetails, Object roomStatus) {
this.roomNo = roomNo;
this.roomType = roomType;
this.pricePerNight = pricePerNight;
this.roomDetails = roomDetails;
```

```
this.roomStatus = roomStatus;
}

public int getRoomNo() {
return roomNo;
}

public void setRoomNo(int roomNo) {
this.roomNo = roomNo;
}

public Object getRoomType() {
return roomType;
}

public void setRoomType(Object object) {
this.roomType = object;
}

public String getPricePerNight() {
return pricePerNight;
}

public void setPricePerNight(String pricePerNight) {
this.pricePerNight = pricePerNight;
}

public String getRoomDetails() {
return roomDetails;
}

public void setRoomDetails(String roomDetails) {
this.roomDetails = roomDetails;
}
```

```
public Object getRoomStatus() {
return roomStatus;
}

public void setRoomStatus(Object object) {
this.roomStatus = object;
}
}
```

Users.java

```
package Models;

public class Users {
private static int userID;
private String email;
private String userName;
private String password;
private String confirmPassword;
private String userType;
public Users() {
Users.userID = 0;
this.email = "";
this.userName = "";
this.password = "";
this.confirmPassword = "";
this.userType = "";
}
public Users(int id, String email, String userName, String password, String
confirmPassword, String userType ) {
Users.userID = id;
this.email = email;
this.userName = userName;
this.password = password;
```

```
this.confirmPassword = confirmPassword;
this.userType = userType;
}

public static int getUserID() {
return userID;
}

public void setUserID(int userID) {
Users.userID = userID;
}

public String getEmail() {
return email;
}

public void setEmail(String email) {
this.email = email;
}

public String getUserName() {
return userName;
}

public void setUserName(String userName) {
this.userName = userName;
}

public String getPassword() {
return password;
}

public void setPassword(String password) {
this.password = password;
```

```
}

public String getConfirmPassword() {
return confirmPassword;
}

public void setConfirmPassword(String confirmPassword) {
this.confirmPassword = confirmPassword;
}

public String getUserType() {
return userType;
}

public void setUserType(String userType) {
this.userType = userType;
}
}
```

Pakage ServiceLayer

BillGenerateServiceLayer.java

```
package ServiceLayer;

import Models.DefultModel;
import DatabaseLayer.BillGenerateDatabaseLayer;
import DatabaseLayer.RoomAssignDatabaseLayer;
import DatabaseLayer.RoomDatabaseLayer;

public class BillGenerateServiceLayer {
```

```
DefultModel defultModel;
public BillGenerateServiceLayer() {
this.defultModel = new DefultModel();
}

public DefultModel getDefultModel() {
return defultModel;
}


public DefultModel generateBill(DefultModel defultModel) throws Exception {
try {
BillGenerateDatabaseLayer generateBillDL = new
BillGenerateDatabaseLayer(defultModel);
return generateBillDL.generateBill();
}catch(Exception e) {
throw e;
}
}

}
```

BookingDetailsServiceLayer.java

```
package ServiceLayer;

import java.util.ArrayList;

import javax.swing.JOptionPane;

import DatabaseLayer.BookingDetailsDatabaseLayer;
import DatabaseLayer.UserDetailsDatabaseLayer;
import Models.DefultModel;
```

```
public class BookingDetailsServiceLayer {
DefultModel defultModel;
public BookingDetailsServiceLayer() {
this.defultModel = new DefultModel();
}

public DefultModel getDefultModel() {
return defultModel;
}
public ArrayList<DefultModel> getAllBooking() throws Exception {
try {
BookingDetailsDatabaseLayer bookingDetailsDL = new
BookingDetailsDatabaseLayer(this.defultModel);
return bookingDetailsDL.loadUserDetails();
}catch(Exception e) {
throw e;
}
}
public ArrayList<DefultModel> getAllUserBooking() throws Exception {
try {
BookingDetailsDatabaseLayer bookingDetailsDL = new
BookingDetailsDatabaseLayer(this.defultModel);
return bookingDetailsDL.getAllUserBooking();
}catch(Exception e) {
throw e;
}
}
public DefultModel deleteBooking(DefultModel defultmodel) throws Exception {
// This function saves the user detail to database and returns the user object after
saving
try {

BookingDetailsDatabaseLayer dlDetailsBooking = new
BookingDetailsDatabaseLayer(defultModel);
```

```
dlDetailsBooking.deleteBooking(defultmodel);
} catch (Exception e) {
throw e;
}
return defultmodel;
}
}
```

CorporateServiceLayer.java

```
package ServiceLayer;
// This class uses the corporate model to receive and send data to the database layer
// This class uses the corporate model to receive data from the frontend layer
import Models.Corporate;
import Models.Customer;

import java.util.ArrayList;

import DatabaseLayer.CorporateDatabaseLayer;
import DatabaseLayer.CustomerDatabaseLayer;
import Helper.InputException;

public class CorporateServiceLayer {
Corporate corporate;
public CorporateServiceLayer() {
this.corporate = new Corporate();
}
public Corporate getCorporate() {
return corporate;
}

public boolean ValidateCorporate(Corporate corporate) throws InputException {
```

```
if(corporate.getCompanyName() == null || corporate.getCompanyName().length() ==
0) {
throw new InputException("Company name cannot be empty.");
}
if(corporate.getCompanyContact() == null || corporate.getCompanyContact().length()
== 0) {
throw new InputException("Company Contact cannot be empty.");
}
if(!(corporate.getCompanyContact()).matches("^[0-9, ]+$") ) {
throw new InputException("cannot input alphabet on contact");
}
return true;
}
public Corporate corporateSave(Corporate corporate) throws Exception {
try {
CorporateDatabaseLayer corporateDatabaseLayer = new
CorporateDatabaseLayer(corporate);
return corporateDatabaseLayer.corporateSave();
}catch(Exception e) {
throw e;
}
}
}
// save the corporate, update the corporate, delete the corporate, getCorporateList
```

CreditServiceLayer.java

```
package ServiceLayer;

import DatabaseLayer.CreditCardDatabaseLayer;
import Helper.InputException;
import Models.CreditCard;
```

```
public class CreditServiceLayer {
CreditCard credit;
public CreditServiceLayer() {
this.credit = new CreditCard();
}

public CreditCard getCreditCard() {
return credit;
}

public boolean validateCreditDetails(CreditCard credit) throws InputException {
if(credit.getCardType() == null || credit.getCardType().length() == 0) {
throw new InputException("Card Type cannot be empty.");
}

if(credit.getNameOnCard() == null || credit.getNameOnCard().length() == 0) {
throw new InputException("Please enter your name used on card.");
}
if(!(credit.getNameOnCard()).matches("^[a-zA-Z\\s]*$") ) {
throw new InputException("Cannot input number on name");
}
if(credit.getCardNo() == null || credit.getCardNo().length() == 0) {
throw new InputException("Credit Card Number canot be empty.");
}
if(!(credit.getCardNo()).matches("^[0-9, ]+$") ) {
throw new InputException("cannot input alphabet on contact");
}


return true;
}
public CreditCard creditSave(CreditCard credit) throws Exception {
try {
```

```
CreditCardDatabaseLayer creditDatabaseLayer = new CreditCardDatabaseLayer(credit);
return creditDatabaseLayer.creditSave();
}catch(Exception e) {
throw e;
}
}

}
```

CustomerServiceLayer.java

```
package ServiceLayer;
// This class uses the customer model to receive and send data to the database layer
// This class uses the customer model to receive data from the frontend layer
import Models.Customer;
import Models.Users;

import java.util.ArrayList;

import DatabaseLayer.CustomerDatabaseLayer;
import Helper.InputException;

public class CustomerServiceLayer {
Customer customer;
public CustomerServiceLayer() {
this.customer = new Customer();
}
public Customer getCustomer() {
return customer;
}
//LogInBox validation
public boolean ValidateCustomer(Customer customer) throws InputException {
if(customer.getFirstName() == null || customer.getFirstName().length() == 0) {
```

```
throw new InputException("First name cannot be empty.");
}
if(!(customer.getFirstName()).matches("[a-zA-Z]+") ) {
throw new InputException("Cannot input number on first name");
}
if(customer.getLastName() == null || customer.getLastName().length() == 0) {
throw new InputException("Last name cannot be empty.");
}
if(!(customer.getLastName()).matches("[a-zA-Z]+") ) {
throw new InputException("Cannot input number on last name");
}
return true;
}
public Customer customerSave(Customer customer) throws Exception {
try {
CustomerDatabaseLayer customerDatabaseLayer = new
CustomerDatabaseLayer(customer);
return customerDatabaseLayer.customerSave();
}catch(Exception e) {
throw e;
}
}
}
```

GuestServiceLAyer.java

```
package ServiceLayer;

import java.sql.Date;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
```

```
import DatabaseLayer.GuestDatabaseLayer;
import DatabaseLayer.ReserveDatabaseLayer;
import Helper.InputException;
import Models.GuestData;
import Models.Reservation;


public class GuestServiceLayer {
GuestData guest;
java.util.Date checkInDate;
java.util.Date checkOutDate;
java.util.Date currentDate;
DateTimeFormatter dtf;
LocalDateTime now;
public GuestServiceLayer() {
this.guest = new GuestData();
}

public GuestData getGuestData() {
return guest;
}

public boolean validateGuestDetails(GuestData guest) throws InputException {
if(guest.getFullName() == null || guest.getFullName().length() == 0) {
throw new InputException("Name cannot be empty.");
}
if(!(guest.getFullName()).matches("^[a-zA-Z\\s]*$") ) {
throw new InputException("Cannot input number on name");
}

if(guest.getdOB() == null || guest.getdOB().length() == 0) {
throw new InputException("Date of Birth cannot be empty.");
}
if(guest.getContact() == null || guest.getContact().length() == 0) {
```

```
throw new InputException("Contact cannot be empty.");
}
if(!(guest.getContact()).matches("^[0-9, ]+$") ) {
throw new InputException("cannot input alphabet on contact");
}

if(guest.getRoomPreference() == null || guest.getRoomPreference().length() == 0) {
throw new InputException("Please enter your room preference");
}

if(guest.getCountry() == null || guest.getCountry().length() == 0) {
throw new InputException("Country Name cannot be empty.");
}
if(!(guest.getCountry()).matches("[a-zA-Z]+") ) {
throw new InputException("Cannot input number on Country");
}
if(guest.getState() == null || guest.getState().length() == 0) {
throw new InputException("State Name cannot be empty.");
}
if(!(guest.getState()).matches("[a-zA-Z]+") ) {
throw new InputException("Cannot input number on State");
}
if(guest.getCity() == null || guest.getCity().length() == 0) {
throw new InputException("City Name cannot be empty.");
}
if(!(guest.getCity()).matches("[a-zA-Z]+") ) {
throw new InputException("Cannot input number on City.");
}
dtf = DateTimeFormatter.ofPattern("yyyy/mm/dd");
now = LocalDateTime.now();

SimpleDateFormat format = new SimpleDateFormat("yyyy/mm/dd");
try {
checkInDate = format.parse(guest.getCheckInDate());
```

```
checkOutDate = format.parse(guest.getCheckOutDate());
currentDate = format.parse(dtf.format(now));
} catch (ParseException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
if(guest.getCheckInDate() == null || guest.getCheckInDate().length() == 0) {
throw new InputException("Must enter Check Out Date");
}
if(checkInDate.before(currentDate)) {
throw new InputException("Invalid Check in Date");
}
if(guest.getCheckOutDate() == null || guest.getCheckOutDate().length() == 0) {
throw new InputException("Must enter Check Out Date");
}
if(checkOutDate.before(checkInDate)) {
throw new InputException("Invalid Check Out Date");
}
return true;
}
public GuestData guestSave(GuestData guest) throws Exception {
try {
GuestDatabaseLayer guestDatabaseLayer = new GuestDatabaseLayer(guest);
return guestDatabaseLayer.guestSave();
}catch(Exception e) {
throw e;
}
}

}
```

PaymentDetailsServiceLAyer.java

```
package ServiceLayer;
```

```
import java.util.ArrayList;

import DatabaseLayer.PaymentDetailsDatabaseLayer;
import DatabaseLayer.UserDatabaseLayer;
import Models.DefultModel;
import Models.Users;


public class PaymentDetailsServiceLayer {
DefultModel defultModel;
public PaymentDetailsServiceLayer() {
this.defultModel = new DefultModel();
}

public DefultModel getDefultModel() {
return defultModel;
}
public ArrayList<DefultModel> getAllPayment() throws Exception {
try {
PaymentDetailsDatabaseLayer paymentDetailsDL = new
PaymentDetailsDatabaseLayer(this.defultModel);
return paymentDetailsDL.loadPaymentDetails();
}catch(Exception e) {
throw e;
}
}
public DefultModel generateBill() throws Exception {
try {
PaymentDetailsDatabaseLayer paymentDatabaseLayer = new
PaymentDetailsDatabaseLayer(defultModel);
return paymentDatabaseLayer.generateBill();
}catch(Exception e) {
throw e;
```

```
}
}
}
```

ReserveServiceLayer.java

```
package ServiceLayer;

import DatabaseLayer.ReserveDatabaseLayer;
import Models.Reservation;


public class ReserveServiceLayer {
Reservation reserve;
public ReserveServiceLayer() {
this.reserve = new Reservation();
}

public Reservation getReservation() {
return reserve;
}
public Reservation reservationSave(Reservation reserve) throws Exception {
try {
ReserveDatabaseLayer reserveDL = new ReserveDatabaseLayer(reserve);
return reserveDL.reserveSave();
}catch(Exception e) {
throw e;
}
}
}
```

RoomAssignServiceLayer.java

```
package ServiceLayer;


import Models.DefultModel;
import DatabaseLayer.RoomAssignDatabaseLayer;
import DatabaseLayer.RoomDatabaseLayer;

public class RoomAssignServiceLayer {
DefultModel defultModel;
public RoomAssignServiceLayer() {
this.defultModel = new DefultModel();
}

public DefultModel getDefultModel() {
return defultModel;
}


public DefultModel roomAssign(DefultModel defultModel) throws Exception {
try {
RoomAssignDatabaseLayer roomAssignDL = new
RoomAssignDatabaseLayer(defultModel);
return roomAssignDL.roomAssign();
}catch(Exception e) {
throw e;
}
}

}
```

RoomServiceLayer.java

```
package ServiceLayer;
```

```
import Models.Room;

// This class uses the room model to receive and send data to the database layer
// This class uses the room model to receive data from the frontend layer
import Models.Users;

import java.util.ArrayList;


import DatabaseLayer.RoomDatabaseLayer;

public class RoomServiceLayer {
Room room;
public RoomServiceLayer() {
this.room = new Room();
}

public Room getRoom() {
return room;
}


public Room roomSave(Room room) throws Exception {
try {
RoomDatabaseLayer roomDatabaseLayer = new RoomDatabaseLayer(room);
return roomDatabaseLayer.roomSave();
}catch(Exception e) {
throw e;
}
}
public ArrayList<Room> getAllRoom() throws Exception {
try {
```

```
RoomDatabaseLayer roomDL = new RoomDatabaseLayer(this.room);
return roomDL.loadRoom();
}catch(Exception e) {
throw e;
}
}
}
```

UserDetailsServiceLayer.java

```
package ServiceLayer;

import java.util.ArrayList;

import javax.swing.JOptionPane;

import DatabaseLayer.UserDatabaseLayer;
import DatabaseLayer.UserDetailsDatabaseLayer;
import Models.DefultModel;


public class UserDetailsServiceLayer {
DefultModel defultModel;
public UserDetailsServiceLayer() {
this.defultModel = new DefultModel();
}

public DefultModel getDefultModel() {
return defultModel;
}
public ArrayList<DefultModel> getAllData() throws Exception {
try {
UserDetailsDatabaseLayer userDetailsDL = new
UserDetailsDatabaseLayer(this.defultModel);
```

```
return userDetailsDL.loadUserDetails();
}catch(Exception e) {
throw e;
}
}
public DefultModel deleteUser(DefultModel defultmodel) throws Exception {
// This function saves the user detail to database and returns the user object after
saving
try {

UserDetailsDatabaseLayer dlDetailsUser = new
UserDetailsDatabaseLayer(defultModel);
dlDetailsUser.deleteUser(defultmodel);
} catch (Exception e) {
throw e;
}
return defultmodel;
}
}
```

UserServiceLayer.java

```
package ServiceLayer;

// This class uses the user model to receive and send data to the database layer
// This class uses the user model to receive data from the frontend layer
import Models.Users;

import DatabaseLayer.UserDatabaseLayer;
import Helper.InputException;

public class UserServiceLayer {
```

```
Users user;
public UserServiceLayer() {
this.user = new Users();
}

public Users getUser() {
return user;
}

//LogInBox validation
public boolean ValidateLogIn(Users user) throws InputException {
if(user.getUserName() == null || user.getUserName().length() == 0) {
throw new InputException("User name cannot be empty.");
}
if(user.getPassword() == null || user.getPassword().length() == 0) {
throw new InputException("Password cannot be empty.");
}
return true;
}
public boolean ValidateSignup(Users user) throws InputException {
if(user.getEmail() == null || user.getEmail().length() == 0) {
throw new InputException("Email name cannot be empty.");
}
if(!(user.getEmail()).matches("^[\\w-_\\.+]*[\\w-_\\.]\\@([\\w]+\\.)+[\\w]+[\\w]$") ) {
throw new InputException("Invalid email format");
}
if(user.getUserName() == null || user.getUserName().length() == 0) {
throw new InputException("User name cannot be empty.");
}
if(user.getPassword() == null || user.getPassword().length() == 0) {
throw new InputException("Password cannot be empty.");
}
if(!(new String(user.getPassword()).equals(new String(user.getConfirmPassword()))))
{
```

```
throw new InputException("Password not Matched");
}
return true;
}
// save the user, update the user, delete the user, getUserList
public Users userLogIn(Users user) throws Exception {
//this fuction get the user detail to log in
try {
UserDatabaseLayer userDatabaseLayer = new UserDatabaseLayer(user);
return userDatabaseLayer.userLogIn();
} catch (Exception e) {
throw e;
}
}
public Users userSave(Users user) throws Exception {
try {
UserDatabaseLayer userDatabaseLayer = new UserDatabaseLayer(user);
return userDatabaseLayer.userSave();
}catch(Exception e) {
throw e;
}
}
}
```

package DatabaseLayer

BillGenerateDatabaseLayer.java

```
package DatabaseLayer;

import java.sql.Connection;
```

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

import javax.swing.JOptionPane;

import FrontendLayer.BookingDetailsTable;
import Helper.DatabaseConnector;
import Models.DefultModel;

public class BillGenerateDatabaseLayer {

private DefultModel defultModel;
private DatabaseConnector db;
private Connection connection;
private String toPayDate;
private String totalBill;
private String cardNo;
private String paymentStatus;
public BillGenerateDatabaseLayer() {
this.defultModel = new DefultModel();
}
public BillGenerateDatabaseLayer(DefultModel defultModel) throws Exception {
this.defultModel = defultModel;
try{
this.db = DatabaseConnector.getInstance();
this.connection = this.db.getConnection();
}catch (Exception ex) {
throw ex;
}
}

public DefultModel getDefultModel() {
return defultModel;
}
```

```
public void setDefultModel(DefultModel defultModel) {
this.defultModel = defultModel;
}

public DefultModel generateBill() throws Exception {
PreparedStatement gettingDataStatement;
PreparedStatement billInserStatement;
PreparedStatement discountUpdatestatement;
PreparedStatement discountNonStatement;

ResultSet grs;
try {
String selectBillQuery = "SELECT g.checkOutDate As toPayDate, "
+ "(DATEDIFF(g.checkOutDate ,g.checkInDate) * r2.pricePerNight) AS TotalBill, "
+ "cc.cardNo , r.paymentStatus FROM Reservation r "
+ "LEFT JOIN Guest g ON r.cusID = g.cusID "
+ "LEFT JOIN Room r2 ON r2.roomNo = r.roomNo "
+ "LEFT JOIN CreditCard cc ON cc.cusID = r.cusID WHERE r.roomNo IS NOT
NULL AND r.reserveID = ? GROUP BY r.reserveID ";
gettingDataStatement = this.connection.prepareStatement(selectBillQuery);
gettingDataStatement.setInt(1, this.defultModel.getBookingID());

grs = gettingDataStatement.executeQuery();
if (grs.next()) {
toPayDate = grs.getString("toPayDate");
totalBill = grs.getString("totalBill");
cardNo = grs.getString("cardNo");
paymentStatus = grs.getString("paymentStatus");

}

String insertBillQuery = "INSERT INTO Billing (toPayDate, totalBill, reserveID,
cardNo, paymentStatus, discountPer) VALUES (?,?,?,?,?,?)";
```

```
billInserStatement = this.connection.prepareStatement(insertBillQuery);
billInserStatement.setString(1, toPayDate);
billInserStatement.setString(2, totalBill);
billInserStatement.setInt(3, this.defultModel.getBookingID());
billInserStatement.setString(4, cardNo);
billInserStatement.setString(5, paymentStatus);
billInserStatement.setString(6, this.defultModel.getDiscountPer());
String updateBillQuery = "UPDATE Billing b LEFT "
+ "JOIN Reservation r ON b.reserveID = r.reserveID "
+ "LEFT JOIN Corporate c ON c.cusID = r.cusID "
+ "SET b.totalBill = (b.totalBill -(b.totalBill * b.discountPer)/100), "
+"b.toPayDate = DATE_ADD(b.toPayDate, INTERVAL 1 MONTH)"
+ "WHERE c.corpID != 0 AND r.reserveID = ? AND r.roomNo IS NOT NULL" ;
discountUpdatestatement = this.connection.prepareStatement(updateBillQuery);
discountUpdatestatement.setInt(1, this.defultModel.getBookingID());
String updatNoneBillQuery = "UPDATE Billing b LEFT "
+ "JOIN Reservation r ON b.reserveID = r.reserveID "
+ "LEFT JOIN Corporate c ON c.cusID = r.cusID "
+ "SET b.totalBill = ? "
+ "WHERE r.reserveID = ? " ;
discountNonStatement = this.connection.prepareStatement(updatNoneBillQuery);
discountNonStatement.setString(1, totalBill);
discountNonStatement.setInt(2, this.defultModel.getBookingID());
try {
if(billInserStatement.executeUpdate() != 0) {
if(discountUpdatestatement.executeUpdate() != 0) {
JOptionPane.showMessageDialog(null, "Bill Generated", "Sucess", 2);

}
else if (discountNonStatement.executeUpdate()!=0) {
JOptionPane.showMessageDialog(null, "No discount Added", "Sucess", 2);

}
else {
```

```
JOptionPane.showMessageDialog(null, "Room not assigned", "Sucess", 2);

}
}
} catch (Exception ex) {
throw ex;
}
} catch (Exception ex) {
throw ex;
}
return defultModel;
}

}
```

BookingDetailsDatabaseLayer.java

```
package DatabaseLayer;

import java.sql.Connection;


import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;

import javax.swing.JOptionPane;

import Helper.DatabaseConnector;
import Models.DefultModel;


public class BookingDetailsDatabaseLayer {
```

```
private DefultModel defultModel;
private DatabaseConnector db;
private Connection connection;
public BookingDetailsDatabaseLayer() {
this.defultModel = new DefultModel();
}
public BookingDetailsDatabaseLayer(DefultModel defultModel) throws Exception {
this.defultModel = defultModel;
try{
this.db = DatabaseConnector.getInstance();
this.connection = this.db.getConnection();
}catch (Exception ex) {
throw ex;
}
}

public DefultModel getDefultModel() {
return defultModel;
}

public void setDefultModel(DefultModel defultModel) {
this.defultModel = defultModel;
}

public ArrayList<DefultModel> loadUserDetails() throws Exception {
try {
ArrayList<DefultModel> defultModel = new ArrayList<DefultModel>();
String query = "SELECT r.reserveID , g.fullName , c.companyName,
c.companyContact, "
+ "cc.cardNo , g.roomPreference , g.country ,g.checkInDate ,"
+ "g.checkOutDate, r.bookingStatus , r.roomNo FROM Reservation r "
+ "LEFT JOIN Customer c2 ON c2.cusID = r.cusID "
+ "LEFT JOIN Corporate c ON r.cusID = c.cusID "
```

```
+ "LEFT JOIN CreditCard cc ON cc.cusID = r.cusID "
+ "LEFT JOIN Guest g ON g.guestID = r.reserveID "
+ "GROUP BY r.reserveID";
Statement statement = this.connection.createStatement();
ResultSet rs = statement.executeQuery(query);
while(rs.next()) {
DefultModel dM = new DefultModel();
dM.setBookingID(rs.getInt("reserveID"));
dM.setName(rs.getString("fullName"));
dM.setCorpName(rs.getString("companyName"));
dM.setCorpContact(rs.getString("companyContact"));
dM.setCardNumber(rs.getString("cardNo"));
dM.setRoomPreference(rs.getString("roomPreference"));
dM.setCountry(rs.getString("country"));
dM.setCheckInDate(rs.getString("checkInDate"));
dM.setCheckOutDate(rs.getString("checkOutDate"));
dM.setBookingStatus(rs.getString("bookingStatus"));
dM.setRoomNo(rs.getString("roomNo"));
defultModel.add(dM);
}
return defultModel;
}
catch(Exception ex) {
throw ex;
}
}
public ArrayList<DefultModel> getAllUserBooking() throws Exception {
try {
ArrayList<DefultModel> defultModel = new ArrayList<DefultModel>();
String query = "SELECT r.reserveID , g.fullName , cc.cardNo , g.roomPreference ,"
+ "g.checkInDate , g.checkOutDate , r.bookingStatus , r.roomNo , b.paymentStatus "
+ "FROM Reservation r LEFT JOIN Guest g ON r.reserveID = g.guestID "
+ "LEFT JOIN CreditCard cc ON cc.cusID = r.cusID "
+ "LEFT JOIN Billing b ON b.reserveID = r.reserveID "
```

```
+ "LEFT JOIN Customer c ON c.cusID = r.cusID WHERE c.cusID = ? GROUP BY
r.reserveID";
PreparedStatement statement = this.connection.prepareStatement(query);
statement.setInt(1, UserDatabaseLayer.cusPrimeKey);
ResultSet rs = statement.executeQuery();
while(rs.next()) {
DefultModel dM = new DefultModel();
dM.setBookingID(rs.getInt("reserveID"));
dM.setName(rs.getString("fullName"));
dM.setCardNumber(rs.getString("cardNo"));
dM.setRoomPreference(rs.getString("roomPreference"));
dM.setCheckInDate(rs.getString("checkInDate"));
dM.setCheckOutDate(rs.getString("checkOutDate"));
dM.setBookingStatus(rs.getString("bookingStatus"));
dM.setRoomNo(rs.getString("roomNo"));
dM.setPaymentStatus(rs.getString("paymentStatus"));
defultModel.add(dM);
}
return defultModel;
}
catch(Exception ex) {
throw ex;
}
}
public DefultModel deleteBooking(DefultModel defultmodel) throws Exception {
try {
// create the statement

String query = "DELETE FROM Reservation WHERE reserveID = ?";
PreparedStatement statement = this.connection.prepareStatement(query);
statement.setInt(1,defultmodel.getBookingID());
// execute the query
if(statement.executeUpdate()!= 0) {
JOptionPane.showInternalMessageDialog(null, "Booking Details Deleted");
```

```
}
else {
JOptionPane.showInternalMessageDialog(null, "Unable to delete");

}
}catch(Exception ex) {
throw ex;
}
return defultModel;
}
}
```

CorporateDatabaseLayer.java

```
package DatabaseLayer;

import java.sql.Connection;



import java.sql.PreparedStatement;
import java.sql.ResultSet;

import Helper.DatabaseConnector;
import Models.Corporate;
import Models.Users;

public class CorporateDatabaseLayer {

private Corporate corporate;
private DatabaseConnector db;
private Connection connection;
```

```
public CorporateDatabaseLayer() {
this.corporate = new Corporate();
}
public CorporateDatabaseLayer(Corporate corporate) throws Exception {
this.corporate = corporate;
try{
this.db = DatabaseConnector.getInstance();
this.connection = this.db.getConnection();
}catch (Exception ex) {
throw ex;
}
}

public Corporate getCorporate() {
return corporate;
}

public void setCorporate(Corporate corporate) {
this.corporate = corporate;
}
public Corporate corporateSave() throws Exception {
PreparedStatement statement;
ResultSet rs;
String registerCorporateQuery = "INSERT INTO Corporate (companyName,
companyContact, userID, cusID) VALUES (?,?,?,?)";
try {
statement = this.connection.prepareStatement(registerCorporateQuery);
statement.setString(1, this.corporate.getCompanyName());
statement.setString(2, this.corporate.getCompanyContact());
statement.setInt(3, UserDatabaseLayer.PrimKey);
statement.setInt(4, CustomerDatabaseLayer.PrimKey);
try {
if (statement.executeUpdate() !=0) {
}
```

```
} catch (Exception ex) {
throw ex;
}
} catch (Exception ex) {
throw ex;
}
return corporate;
}
}
```

CreditCardDatabaseLayer.java

```
package DatabaseLayer;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

import javax.swing.JOptionPane;


import Helper.DatabaseConnector;
import Models.CreditCard;
import Models.Customer;
import Models.Users;

public class CreditCardDatabaseLayer {

public static int creditPrimkey = 0;
private CreditCard credit;
private DatabaseConnector db;
private Connection connection;
public CreditCardDatabaseLayer() {
```

```
this.credit = new CreditCard();
}
public CreditCardDatabaseLayer(CreditCard credit) throws Exception {
this.credit = credit;
try{
this.db = DatabaseConnector.getInstance();
this.connection = this.db.getConnection();
}catch (Exception ex) {
throw ex;
}
}

public CreditCard getCreditCard() {
return credit;
}

public void setCreditCard(CreditCard credit) {
this.credit = credit;
}
//credit login method
public CreditCard creditSave() throws Exception {
PreparedStatement statement;
PreparedStatement updateStatement;

ResultSet rs;
String creditDetailQuery = "INSERT INTO CreditCard (cardType,nameOnCard,
cardNo, cusID) VALUES (?,?,?,?)";
String updateQuery = "Update CreditCard SET cardType = ?,nameOnCard = ?,
cardNo = ? WHERE cusID= ?";

try {
String[] creditID = new String[] { "id" };
statement = this.connection.prepareStatement(creditDetailQuery, creditID);
statement.setString(1, this.credit.getCardType());
```

```
statement.setString(2, this.credit.getNameOnCard());
statement.setString(3, this.credit.getCardNo());
statement.setInt(4, UserDatabaseLayer.cusPrimeKey);
updateStatement = this.connection.prepareStatement(updateQuery);
updateStatement.setString(1, this.credit.getCardType());
updateStatement.setString(2, this.credit.getNameOnCard());
updateStatement.setString(3, this.credit.getCardNo());
updateStatement.setInt(4, UserDatabaseLayer.cusPrimeKey);
try {
if (statement.executeUpdate() !=0) {
updateStatement.executeUpdate();
ResultSet generatedKeys = statement.getGeneratedKeys();
if ( generatedKeys.next() ) {
creditPrimkey = generatedKeys.getInt(1);
}
}
} catch (Exception ex) {
throw ex;
}
} catch (Exception ex) {
throw ex;
}
return credit;
}}
```

CustomerDatabaseLayer.java

```
package DatabaseLayer;

import java.sql.Connection;
```

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

import javax.swing.JOptionPane;

import FrontendLayer.AdminPage;
import FrontendLayer.SignUpBox;
import FrontendLayer.UserHomePage;
import Helper.DatabaseConnector;
import Models.Customer;
import Models.Users;


public class CustomerDatabaseLayer {

private Customer customer;
private DatabaseConnector db;
private Connection connection;
public static int PrimKey;
public CustomerDatabaseLayer() {
this.customer = new Customer();
}
public CustomerDatabaseLayer(Customer customer) throws Exception {
this.customer = customer;
try{
this.db = DatabaseConnector.getInstance();
this.connection = this.db.getConnection();
}catch (Exception ex) {
throw ex;
}
}

public Customer getCustomer() {
return customer;
```

```
}

public void setCustomer(Customer customer) {
this.customer = customer;
}
public Customer customerSave() throws Exception {
String[] cusID = new String[] { "id" };
PreparedStatement statement;
ResultSet rs;
String registerCustomerQuery = "INSERT INTO Customer (firstName, lastName,
userID) VALUES (?,?,?)";
try {
statement = this.connection.prepareStatement(registerCustomerQuery, cusID);
statement.setString(1, this.customer.getFirstName());
statement.setString(2, this.customer.getLastName());
statement.setInt(3, UserDatabaseLayer.PrimKey);
try {
if (statement.executeUpdate() !=0) {
ResultSet generatedKeys = statement.getGeneratedKeys();
if ( generatedKeys.next() ) {
PrimKey = generatedKeys.getInt(1);
}
}
} catch (Exception ex) {
throw ex;
}
} catch (Exception ex) {
throw ex;
}
return customer;
}
}
```

GuestDatabaseLayer.java

```
package DatabaseLayer;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

import javax.swing.JOptionPane;


import Helper.DatabaseConnector;
import Models.GuestData;
import Models.Reservation;

public class GuestDatabaseLayer {

private GuestData guest;
private DatabaseConnector db;
private Connection connection;
public static int guestPrimeKey;
public GuestDatabaseLayer() {
this.guest = new GuestData();
}
public GuestDatabaseLayer(GuestData guest) throws Exception {
this.guest = guest;
try{
this.db = DatabaseConnector.getInstance();
this.connection = this.db.getConnection();
}catch (Exception ex) {
throw ex;
}
}

public GuestData getGuestData() {
```

```
return guest;
}

public void setGuestData(GuestData guest) {
this.guest = guest;
}
//guest login method
public GuestData guestSave() throws Exception {
PreparedStatement statement;
ResultSet rs;
String guestDetailQuery = "INSERT INTO Guest (fullName, dOb, contact, "
+ "roomPreference, country, state, city, "
+ "checkInDate, checkOutDate, cusID) VALUES (?,?,?,?,?,?,?,?,?,?)";
try {
String[] guestID = new String[] { "id" };
statement = this.connection.prepareStatement(guestDetailQuery, guestID);
statement.setString(1, this.guest.getFullName());
statement.setString(2, this.guest.getdOB());
statement.setString(3, this.guest.getContact());
statement.setString(4, this.guest.getRoomPreference());
statement.setString(5, this.guest.getCountry());
statement.setString(6, this.guest.getState());
statement.setString(7, this.guest.getCity());
statement.setString(8, this.guest.getCheckInDate());
statement.setString(9, this.guest.getCheckOutDate());
statement.setInt(10, UserDatabaseLayer.cusPrimeKey);
try {
if (statement.executeUpdate() !=0) {
ResultSet generatedKeys = statement.getGeneratedKeys();
if ( generatedKeys.next() ) {
guestPrimeKey = generatedKeys.getInt(1);
}
}
}
```

```
catch (Exception ex) {
throw ex;
}
}catch (Exception ex) {
throw ex;
}
return guest;
}
}
```

PaymentsDetailsDatabaseLayer.java

```
package DatabaseLayer;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;

import javax.swing.JOptionPane;

import FrontendLayer.AdminPage;

import FrontendLayer.UserHomePage;
import Helper.DatabaseConnector;
import Models.DefultModel;
import Models.Users;

public class PaymentDetailsDatabaseLayer {
```

```
private DefultModel defultModel;
private DatabaseConnector db;
private Connection connection;
public PaymentDetailsDatabaseLayer() {
this.defultModel = new DefultModel();
}
public PaymentDetailsDatabaseLayer(DefultModel defultModel) throws Exception {
this.defultModel = defultModel;
try{
this.db = DatabaseConnector.getInstance();
this.connection = this.db.getConnection();
}catch (Exception ex) {
throw ex;
}
}

public DefultModel getDefultModel() {
return defultModel;
}

public void setDefultModel(DefultModel defultModel) {
this.defultModel = defultModel;
}

public ArrayList<DefultModel> loadPaymentDetails() throws Exception {
try {
ArrayList<DefultModel> defultModel = new ArrayList<DefultModel>();
String query = "SELECT r.reserveID , g.fullName , g.contact , r.roomNo ,"
+ "r2.pricePerNight , b.totalBill , b.paymentStatus "
+ "FROM Reservation r LEFT JOIN Guest g ON r.reserveID = g.guestID "
+ "LEFT JOIN Room r2 ON r2.roomNo = r.roomNo "
+ "LEFT JOIN Billing b ON b.reserveID = r.reserveID GROUP BY r.reserveID ";
Statement statement = this.connection.createStatement();
ResultSet rs = statement.executeQuery(query);
```

```
while(rs.next()) {
DefultModel dM = new DefultModel();
dM.setBookingID(rs.getInt("reserveID"));
dM.setGuestName(rs.getString("fullName"));
dM.setContact(rs.getString("contact"));
dM.setRoomNo(rs.getString("roomNo"));
dM.setPricePerNight(rs.getString("pricePerNight"));
dM.setTotalBill(rs.getString("totalBill"));
dM.setPaymentStatus(rs.getString("paymentStatus"));
defultModel.add(dM);
}
return defultModel;
}
catch(Exception ex) {
throw ex;
}
}
public DefultModel generateBill() throws Exception {
PreparedStatement billStatement;
ResultSet brs;
//getting email and password from the login page
// query to check if the username and password exist in data base or not
String billQuery = "SELECT * FROM Reservation r INNER JOIN Customer c ON
r.cusID = c.cusID WHERE r.reserveID = ? AND c.cusID = ?";
try {
billStatement = this.connection.prepareStatement(billQuery);
billStatement.setInt(1, this.defultModel.getBookingID());
billStatement.setInt(2, UserDatabaseLayer.cusPrimeKey);
brs = billStatement.executeQuery();
//condition if the username and password match
if (brs.next())
{
}
else {
```

```
JOptionPane.showMessageDialog(null, "Please check your booking ID", "Failed", 2);
}
}catch(Exception ex) {
throw ex;
}
return defultModel;
}
}
```

ReserveDatabaseLayer.java

```
package DatabaseLayer;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

import javax.swing.JOptionPane;


import Helper.DatabaseConnector;
import Models.Reservation;

public class ReserveDatabaseLayer {

public static int reservePrimkey = 0;
private Reservation reserve;
private DatabaseConnector db;
private Connection connection;
public ReserveDatabaseLayer() {
this.reserve = new Reservation();
}
public ReserveDatabaseLayer(Reservation reserve) throws Exception {
this.reserve = reserve;
```

```
try{
this.db = DatabaseConnector.getInstance();
this.connection = this.db.getConnection();
}catch (Exception ex) {
throw ex;
}
}

public Reservation getReservation() {
return reserve;
}

public void setRegistration(Reservation reserve) {
this.reserve = reserve;
}
//reserve login method
public Reservation reserveSave() throws Exception {
PreparedStatement statement;
ResultSet rs;
String reserveDetailQuery = "INSERT INTO Reservation (bookingStatus,
paymentStatus, cusID) VALUES (?,?,?)";
try {
String[] reserveID = new String[] { "id" };
statement = connection.prepareStatement(reserveDetailQuery, reserveID);
statement.setString(1, this.reserve.getBookingStatus());
statement.setString(2, this.reserve.getPaymentStatus());
statement.setInt(3, UserDatabaseLayer.cusPrimeKey);
try {
if (statement.executeUpdate() !=0) {
JOptionPane.showMessageDialog(null, "Booking Request Send", "Request Send", 2);
ResultSet generatedKeys = statement.getGeneratedKeys();
if ( generatedKeys.next() ) {
reservePrimkey = generatedKeys.getInt(1);
}
```

```
}
else {
JOptionPane.showMessageDialog(null, "Please Check Your Information", "Request
Error", 2);
}
} catch (Exception ex) {
throw ex;
}
} catch (Exception ex) {
throw ex;
}
return reserve;
}}
```

RoomAssignDatabaseLayer.java

```
package DatabaseLayer;

import java.sql.Connection;


import java.sql.PreparedStatement;
import java.sql.ResultSet;

import javax.swing.JOptionPane;

import Helper.DatabaseConnector;
import Models.DefultModel;

public class RoomAssignDatabaseLayer {

private DefultModel defultModel;
```

```
private DatabaseConnector db;
private Connection connection;
public RoomAssignDatabaseLayer() {
this.defultModel = new DefultModel();
}
public RoomAssignDatabaseLayer(DefultModel defultModel) throws Exception {
this.defultModel = defultModel;
try{
this.db = DatabaseConnector.getInstance();
this.connection = this.db.getConnection();
}catch (Exception ex) {
throw ex;
}
}

public DefultModel getDefultModel() {
return defultModel;
}

public void setDefultModel(DefultModel defultModel) {
this.defultModel = defultModel;
}
//room login method
public DefultModel roomAssign() throws Exception {
PreparedStatement updateRoomStatement;
PreparedStatement updateStatusStatement;
ResultSet irs;
ResultSet urs;
try {
String inserRoomNoQuery = "UPDATE Reservation r INNER JOIN CreditCard c
ON c.cusID = r.cusID SET r.roomNo = ? "
+ "WHERE r.reserveID = ? AND r.bookingStatus = 'Pending' AND c.cardNo IS NOT
NULL AND r.roomNo IS NULL";
updateRoomStatement = this.connection.prepareStatement(inserRoomNoQuery);
```

```
updateRoomStatement.setString(1, this.defultModel.getRoomNo());
updateRoomStatement.setInt(2, this.defultModel.getBookingID());
String updateStatusQuery = "UPDATE Reservation r INNER JOIN Room ro ON
r.roomNo = ro.roomNo SET"
+ " r.bookingStatus = 'Approved', ro.roomStatus = 'Booked' WHERE reserveID = ?";
updateStatusStatement = this.connection.prepareStatement(updateStatusQuery);
updateStatusStatement.setInt(1, this.defultModel.getBookingID());
try {
if (updateRoomStatement.executeUpdate() != 0) {
updateStatusStatement.executeUpdate();
JOptionPane.showMessageDialog(null, "Room Assigned", "Complete", 2);
}else {
JOptionPane.showMessageDialog(null, "Room Assigned failed Please check Credit "
+ "Card Detail Or Room no is already assigned or not", "Failed", 2);
}
} catch (Exception ex) {
throw ex;
}
} catch (Exception ex) {
throw ex;
}
return defultModel;
}

}
```

RoomDatabaseLayer.java

```
package DatabaseLayer;

import java.sql.Connection;

import java.sql.PreparedStatement;
```

```
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;

import javax.swing.JOptionPane;

import Helper.DatabaseConnector;
import Models.Room;
import Models.Users;
import FrontendLayer.AdminPage;
import FrontendLayer.LogInBox;

import FrontendLayer.UserHomePage;

public class RoomDatabaseLayer {

public static int primkey = 0;
private Room room;
private DatabaseConnector db;
private Connection connection;
public RoomDatabaseLayer() {
this.room = new Room();
}
public RoomDatabaseLayer(Room room) throws Exception {
this.room = room;
try{
this.db = DatabaseConnector.getInstance();
this.connection = this.db.getConnection();
}catch (Exception ex) {
throw ex;
}
}

public Room getRoom() {
```

```
return room;
}

public void setRoom(Room room) {
this.room = room;
}
//room login method
public Room roomSave() throws Exception {
PreparedStatement statement;
ResultSet rs;
String registerRoomQuery = "INSERT INTO Room (roomType, pricePerNight,
roomDetails, roomStatus) VALUES (?,?,?,?)";
try {
String[] roomID = new String[] { "id" };
statement = this.connection.prepareStatement(registerRoomQuery, roomID);
statement.setObject(1, this.room.getRoomType());
statement.setString(2, this.room.getPricePerNight());
statement.setString(3, this.room.getRoomDetails());
statement.setObject(4, this.room.getRoomStatus());
try {
if (statement.executeUpdate() !=0) {
ResultSet generatedKeys = statement.getGeneratedKeys();
if ( generatedKeys.next() ) {
primkey = generatedKeys.getInt(1);
}
JOptionPane.showMessageDialog(null, "Room has been added");
}
else {
JOptionPane.showInternalMessageDialog(null, "Please Check Your Information");
}
} catch (Exception ex) {
throw ex;
}
} catch (Exception ex) {
```

```
throw ex;
}
return room;
}
public ArrayList<Room> loadRoom() throws Exception {
try {
ArrayList<Room> room = new ArrayList<Room>();
String query = "SELECT * FROM Room ORDER BY roomNo";
Statement statement = this.connection.createStatement();
ResultSet rs = statement.executeQuery(query);
while(rs.next()) {
Room r = new Room();
r.setRoomNo(rs.getInt("roomNo"));
r.setRoomType(rs.getObject("roomtype"));
r.setPricePerNight(rs.getString("pricePerNight"));
r.setRoomDetails(rs.getString("roomDetails"));
r.setRoomStatus(rs.getObject("roomStatus"));
room.add(r);
}
return room;
}
catch(Exception ex) {
throw ex;
}
}
}
```

UserdatabaseLayer.java

```
package DatabaseLayer;

import java.sql.Connection;
```

```
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;

import javax.swing.JOptionPane;

import Helper.DatabaseConnector;
import Models.Room;
import Models.Users;
import FrontendLayer.AdminPage;
import FrontendLayer.LogInBox;

import FrontendLayer.UserHomePage;

public class UserDatabaseLayer {

private Users user;
private DatabaseConnector db;
private Connection connection;
public static int userPrimeKey;
public static int cusPrimeKey;
public static int adminPrimeKey;
public static int PrimKey;
public static String uName;
public static String nameConcat;
public static String userEmail;
public static String corpName;
public static String corpcontact;
public UserDatabaseLayer() {
this.user = new Users();
}
public UserDatabaseLayer(Users user) throws Exception {
this.user = user;
```

```
try{
this.db = DatabaseConnector.getInstance();
this.connection = this.db.getConnection();
}catch (Exception ex) {
throw ex;
}
}

public Users getUser() {
return user;
}

public void setUser(Users user) {
this.user = user;
}
//user login method
public Users userLogIn() throws Exception {
PreparedStatement customerStatement;
PreparedStatement adminStatement;
ResultSet crs;
ResultSet ars;
//getting email and password from the login page
// query to check if the username and password exist in data base or not
String customerQuery = "SELECT u.userID, c.cusID, u.userName, "
+ "CONCAT(c.firstName,' ',c.lastName) AS name, u.email, "
+ "c2.companyName , c2.companyContact FROM Users u "
+ "INNER JOIN Customer c ON u.userID = c.userID "
+ "LEFT JOIN Corporate c2 ON c.userID = c2.userID "
+ "WHERE u.userName = ? AND u.password = ? AND u.userType is NULL";
String adminQuery = "SELECT u.userID, a.adminID FROM Users u INNER JOIN
Administration a ON u.userID = a.userID WHERE u.userName = ? AND u.password
= ? AND u.userType = 'receptionist'";
try {
customerStatement = this.connection.prepareStatement(customerQuery);
```

```
customerStatement.setString(1, this.user.getUserName());
customerStatement.setString(2, this.user.getPassword());
adminStatement = this.connection.prepareStatement(adminQuery);
adminStatement.setString(1, this.user.getUserName());
adminStatement.setString(2, this.user.getPassword());
crs = customerStatement.executeQuery();
ars = adminStatement.executeQuery();
//condition if the username and password match
if (crs.next())
{
//take user to user home page
userPrimeKey = crs.getInt(2);
cusPrimeKey = crs.getInt(2);
uName = crs.getString("userName");
nameConcat = crs.getString("name");
userEmail = crs.getString("email");
corpName = crs.getString("companyname");
corpcontact = crs.getNString("companyContact");
UserHomePage homePage = new UserHomePage();
homePage.setVisible(true);
homePage.pack();
homePage.setLocationRelativeTo(null);
}
else if (ars.next())
{
//take user to user home page
AdminPage adminPage = new AdminPage();
adminPage.setVisible(true);
adminPage.pack();
adminPage.setLocationRelativeTo(null);
adminPrimeKey = ars.getInt(1);
}
else {
```

```
JOptionPane.showMessageDialog(null, "Invalid User name / password", "Log In Error", 2);
}
}catch(Exception ex) {
throw ex;
}
return user;
}
//to check if the user name is already exists
public boolean checkUsername(String username, String email) throws Exception {
PreparedStatement statement;
ResultSet rs;
boolean username_exist = false;
boolean email_exist = false;
String query = "SELECT * FROM Users WHERE username = ? AND email = ? ";
try {
statement = this.connection.prepareStatement(query);
statement.setString(1, username);
statement.setString(2, email);
rs = statement.executeQuery();
if (rs.next()) {
username_exist = true;
JOptionPane.showMessageDialog(null, "This username is already taken, Choose another username","Username Invalid",2);
}
if (rs.next()) {
email_exist = true;
JOptionPane.showMessageDialog(null, "Account already created , Please LogIn","Email already used",2);
}
}catch(Exception ex) {
throw ex;
}
return username_exist;
```

```
}
public Users userSave() throws Exception {
if((!checkUsername(user.getUserName(), user.getEmail()))) {
PreparedStatement statement;
String registerUserQuery = "INSERT INTO Users (email, userName, password,
userID) VALUES (?,?,?,?)";
try {
String generatedID[] = {"id"};
statement = this.connection.prepareStatement(registerUserQuery, generatedID);
statement.setString(1, this.user.getEmail());
statement.setString(2, this.user.getUserName());
statement.setString(3, this.user.getPassword());
statement.setInt(4,UserDatabaseLayer.userPrimeKey);
try {
if (statement.executeUpdate() !=0) {
ResultSet generatedKeys = statement.getGeneratedKeys();
if ( generatedKeys.next() ) {
PrimKey = generatedKeys.getInt(1);
}
JOptionPane.showMessageDialog(null, "Your Account has been created ");
}
else {
JOptionPane.showInternalMessageDialog(null, "Please Check Your Information");
}
} catch (Exception ex) {
throw ex;
}
} catch (Exception ex) {
throw ex;
}
}
return user;
}
```

```
}
```

UserDetailsDatabaseLayer.java

```
package DatabaseLayer;

import java.sql.Connection;


import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;

import javax.swing.JOptionPane;

import Helper.DatabaseConnector;
import Models.DefultModel;


public class UserDetailsDatabaseLayer {

private DefultModel defultModel;
private DatabaseConnector db;
private Connection connection;
public UserDetailsDatabaseLayer() {
this.defultModel = new DefultModel();
}
public UserDetailsDatabaseLayer(DefultModel defultModel) throws Exception {
this.defultModel = defultModel;
try{
this.db = DatabaseConnector.getInstance();
this.connection = this.db.getConnection();
}catch (Exception ex) {
```

```
throw ex;
}
}

public DefultModel getDefultModel() {
return defultModel;
}

public void setDefultModel(DefultModel defultModel) {
this.defultModel = defultModel;
}

public ArrayList<DefultModel> loadUserDetails() throws Exception {
try {
ArrayList<DefultModel> defultModel = new ArrayList<DefultModel>();
String query = "SELECT u.userID, CONCAT(c.firstName,\" \" ,c.lastName) AS
Name, u.email, c2.companyName, c2.companyContact FROM Users u LEFT JOIN
Customer c ON u.userID = c.userID \n"
+ "LEFT JOIN Corporate c2 ON c2.userID = u.userID GROUP BY u.userID ORDER
BY u.userID ";
Statement statement = this.connection.createStatement();
ResultSet rs = statement.executeQuery(query);
while(rs.next()) {
DefultModel dM = new DefultModel();
dM.setUserID(rs.getInt("UserID"));
dM.setName(rs.getString("name"));
dM.setEmail(rs.getString("email"));
dM.setCorpName(rs.getString("companyName"));
dM.setCorpContact(rs.getString("companyContact"));
defultModel.add(dM);
}
return defultModel;
}
catch(Exception ex) {
```

```
throw ex;
}
}
public DefultModel deleteUser(DefultModel defultmodel) throws Exception {
try {
// create the statement

String query = "DELETE FROM Users INNER JOIN WHERE userID = ?";
PreparedStatement statement = this.connection.prepareStatement(query);
statement.setInt(1,defultmodel.getUserID());
// execute the query
if(statement.executeUpdate()!= 0) {
JOptionPane.showInternalMessageDialog(null, "User Deleted");

}
else {
JOptionPane.showInternalMessageDialog(null, "Unable to delete");

}
}catch(Exception ex) {
throw ex;
}
return defultModel;
}
}
```