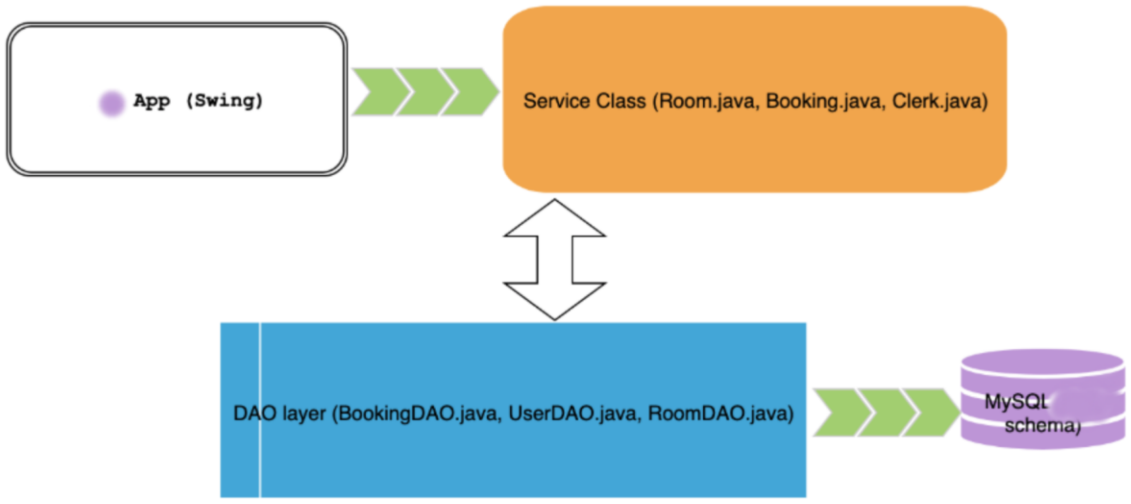
College Room Booking Management

Divya Ramesh Rathod
U2054595

Introduction

This project, I have developed a Room booking application where there will be one manager interface and another clerk(s) interface. Where the manager can add the rooms and schedule the rooms and clerks can book the rooms which are scheduled by the manager. We used Swing, JDBC, MySQL for back end database.

Architecture Diagram


App (Swing)
Service Class (Room.java, Booking.java, Clerk.java)
DAO layer (BookingDAO.java, UserDAO.java, RoomDAO.java)
MySQL schema)

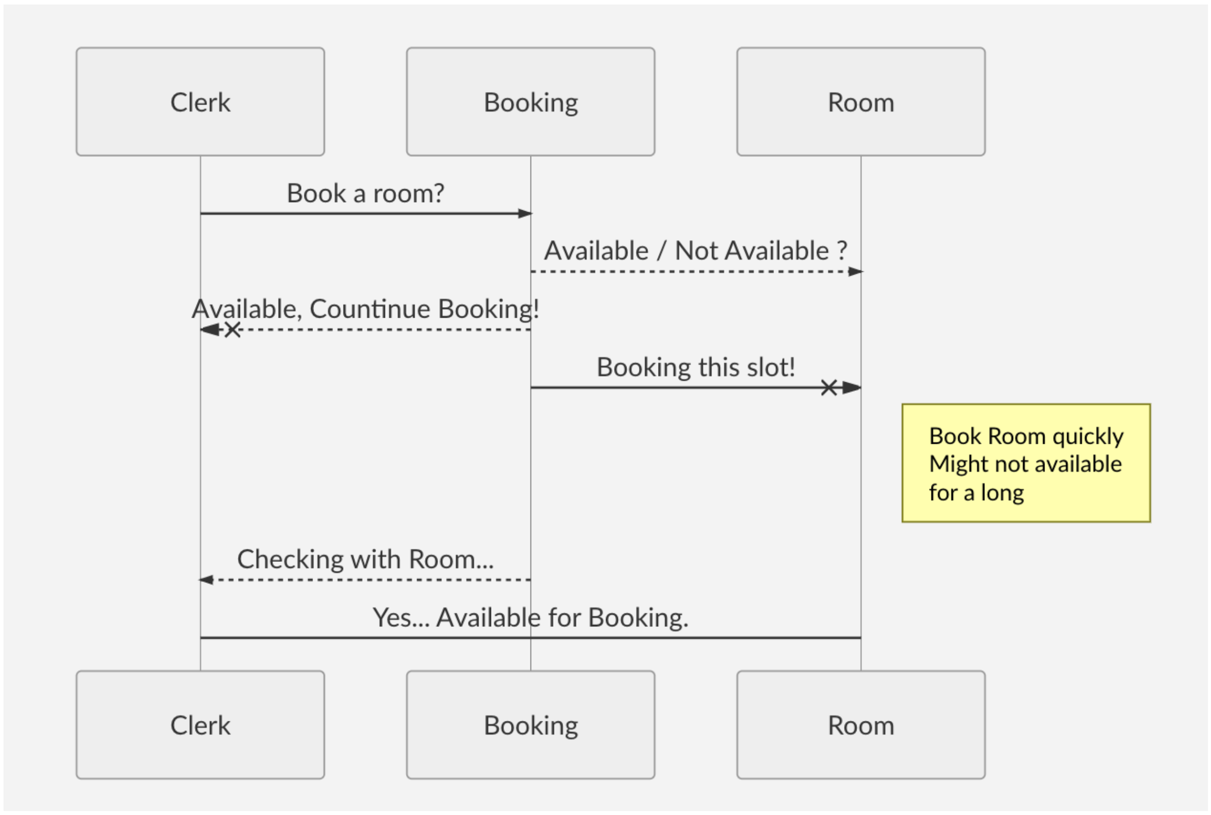
Software / Hardware requirements :

1. Software :
   Java 8 and above
   MySql (Or any Database)
   Eclipse (Any IDE for java)
   Java Swing
   Mac OS (Windows 10)
   Mavan

2. Hardware :
   256 MB RAM
   100 GB Hard Disk
   Any Processor

Class Diagram

Here is one of the Class diagrams between “Clerk - Booking - Room” commutions.

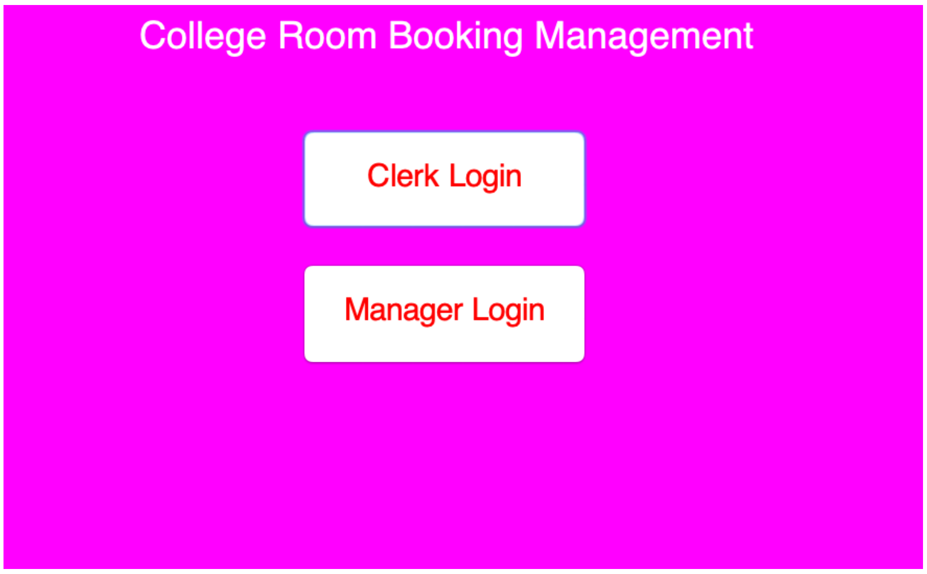

Clerk
Booking
Room
Book a room?
Available / Not Available ?
Available, Countinue Booking!
Booking this slot!
Book Room quickly Might not available for a long
Checking with Room...
Yes... Available for Booking.
Clerk
Booking
Room

GITHUB LINK: https://github.com/Divyarathod779957/Assignment

Use cases

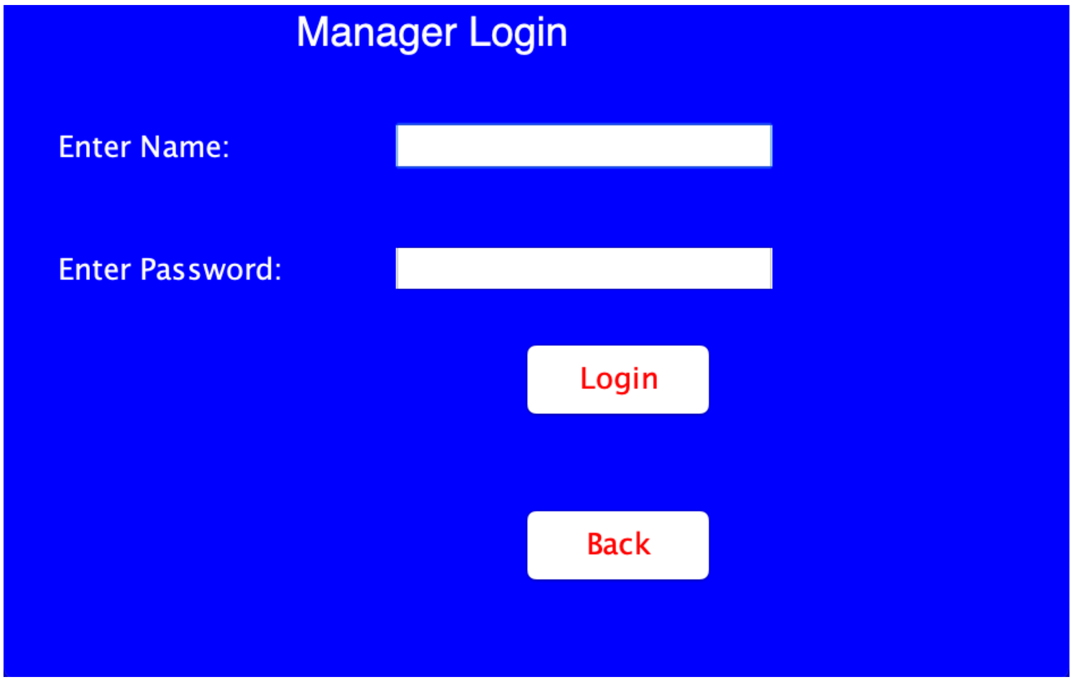
1. Welcome Page

   Common for Manager and Clerk.



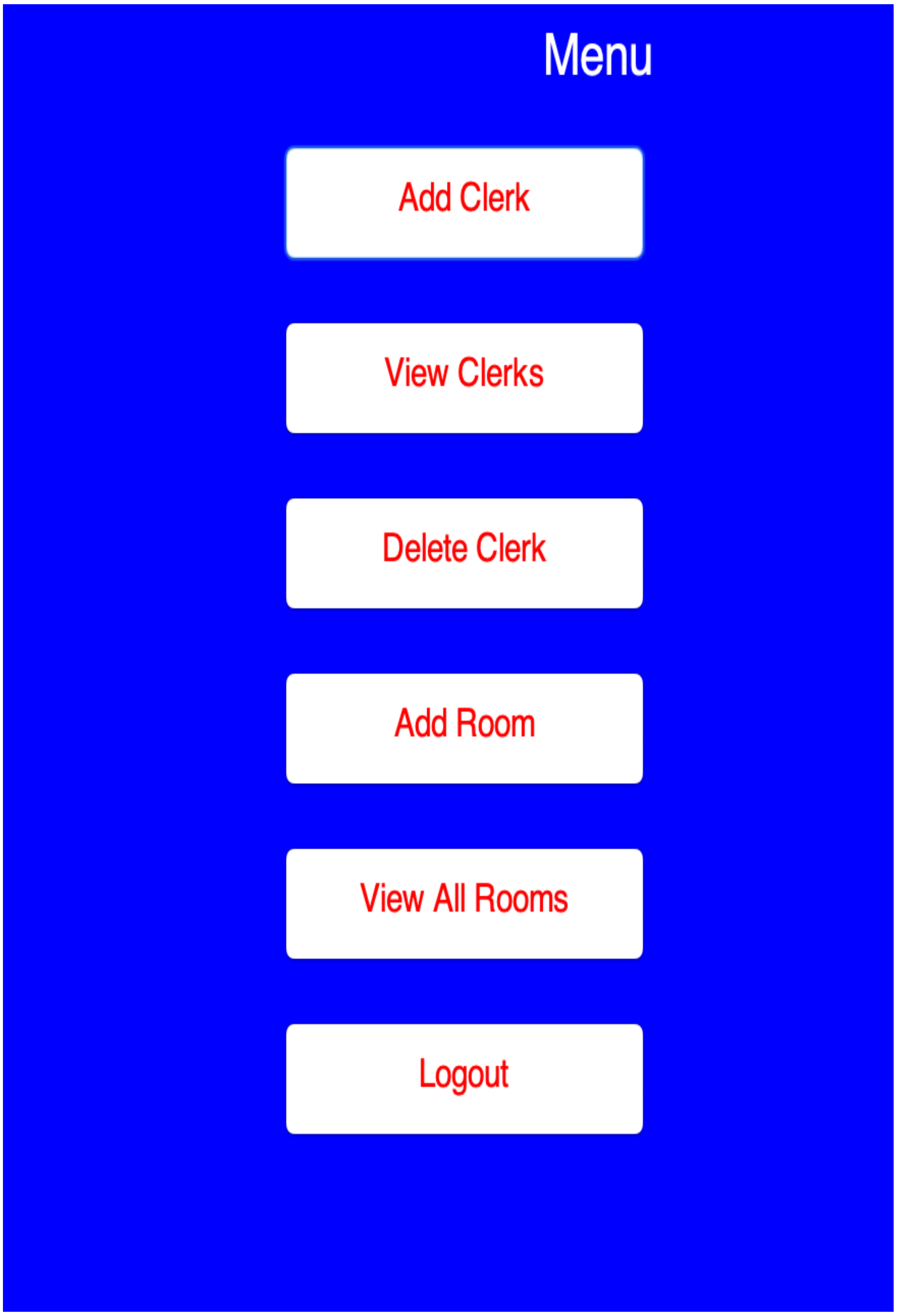
2. Manager Login Page

   Only for Manager Username and Password (01 and 01)



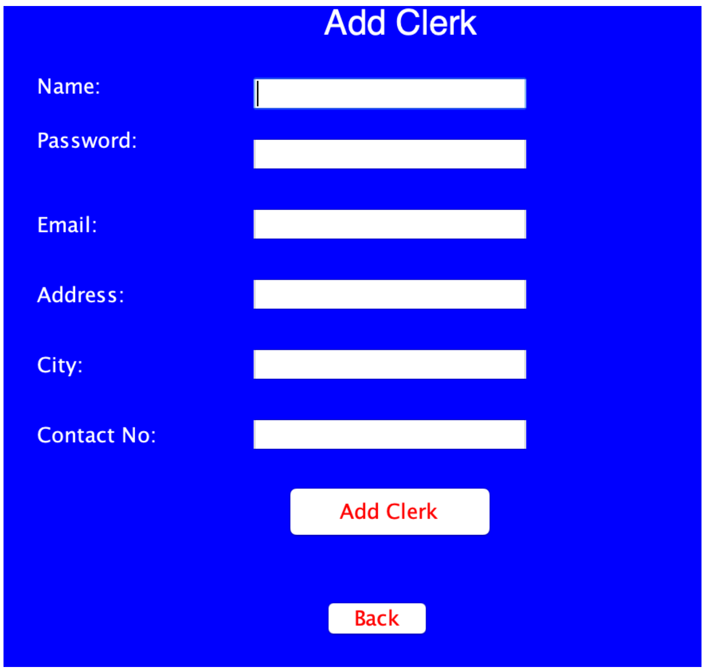
3. Manager Menu

   All action available for Manager


Menu
Add Clerk
View Clerks
Delete Clerk
Add Room
View All Rooms
Logout

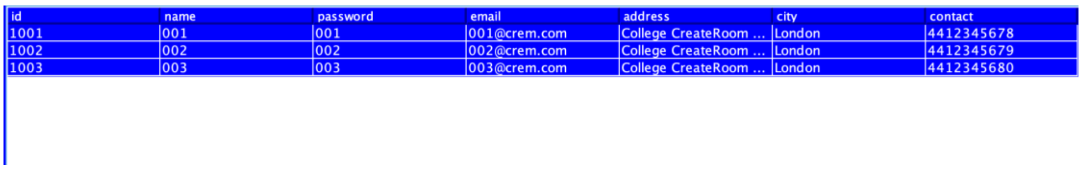
4. Inside Manager Menu
   a. Add Clerk

      Add new Clerk to system



   b. View Clerks

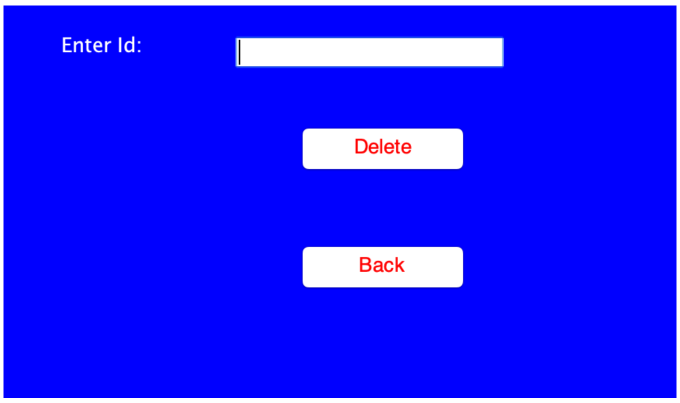
      View all clerks added in the system



id	name	password	email	address	city	contact
1001	001	001	001@crem.com	College CreateRoom ...	London	4412345678
1002	002	002	002@crem.com	College CreateRoom ...	London	4412345679
1003	003	003	003@crem.com	College CreateRoom ...	London	4412345680

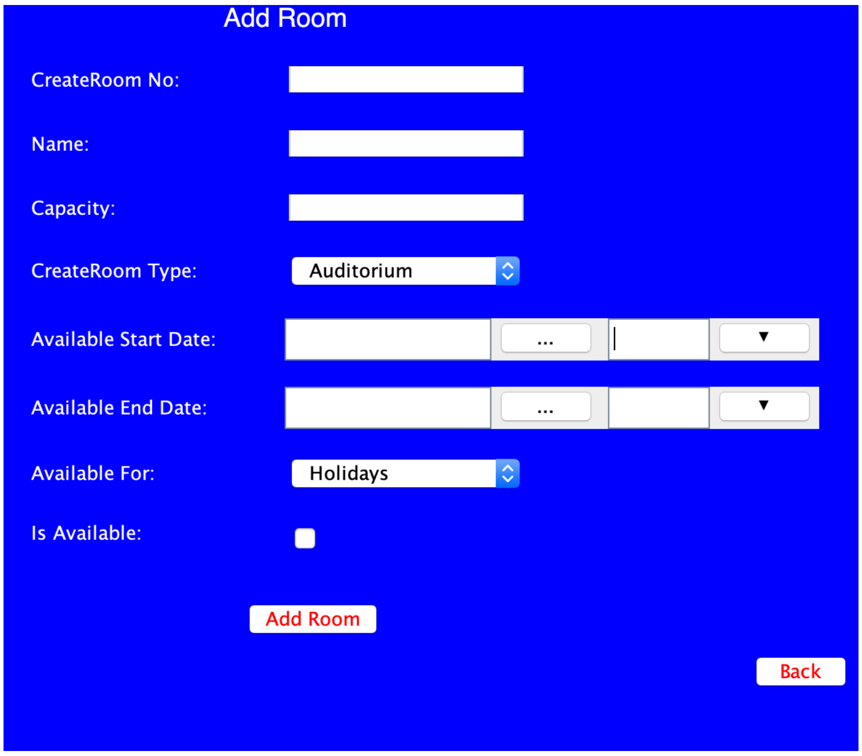
   c. Delete Clerk

      Delete Clerk with Id



   d. Add Room

      Add Rooms which should be available to book

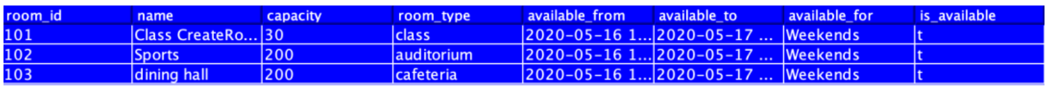


Add Room

CreateRoom No:	
Name:	
Capacity:	
CreateRoom Type:	Auditorium
Available Start Date:	... ▼
Available End Date:	... ▼
Available For:	Holidays
Is Available:	☐

e. View All Rooms

All Rooms that are available for booking



room_id	name	capacity	room_type	available_from	available_to	available_for	is_available
101	Class CreateRo...	30	class	2020-05-16 1...	2020-05-17 ...	Weekends	t
102	Sports	200	auditorium	2020-05-16 1...	2020-05-17 ...	Weekends	t
103	dining hall	200	cafeteria	2020-05-16 1...	2020-05-17 ...	Weekends	t

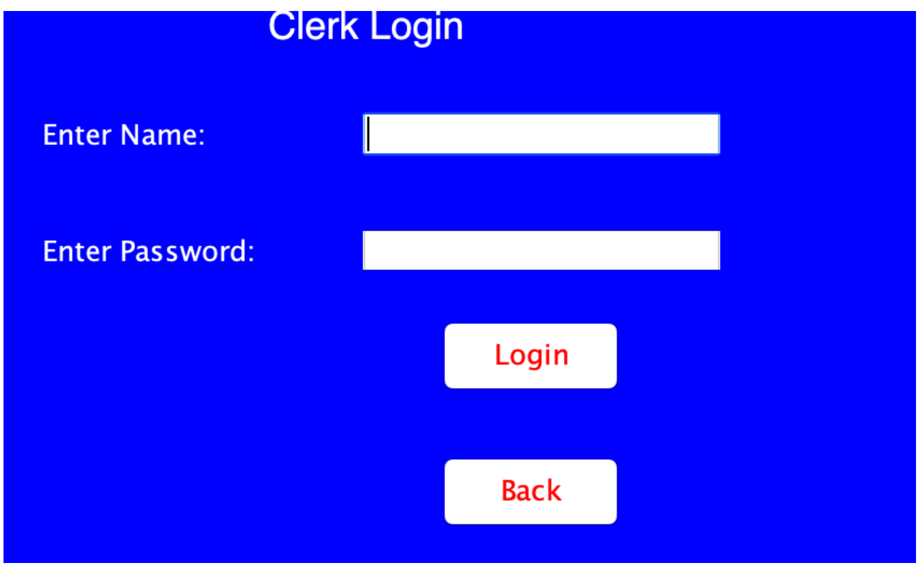
f. Logout

Logout from Manager role.

5. Clerk Login

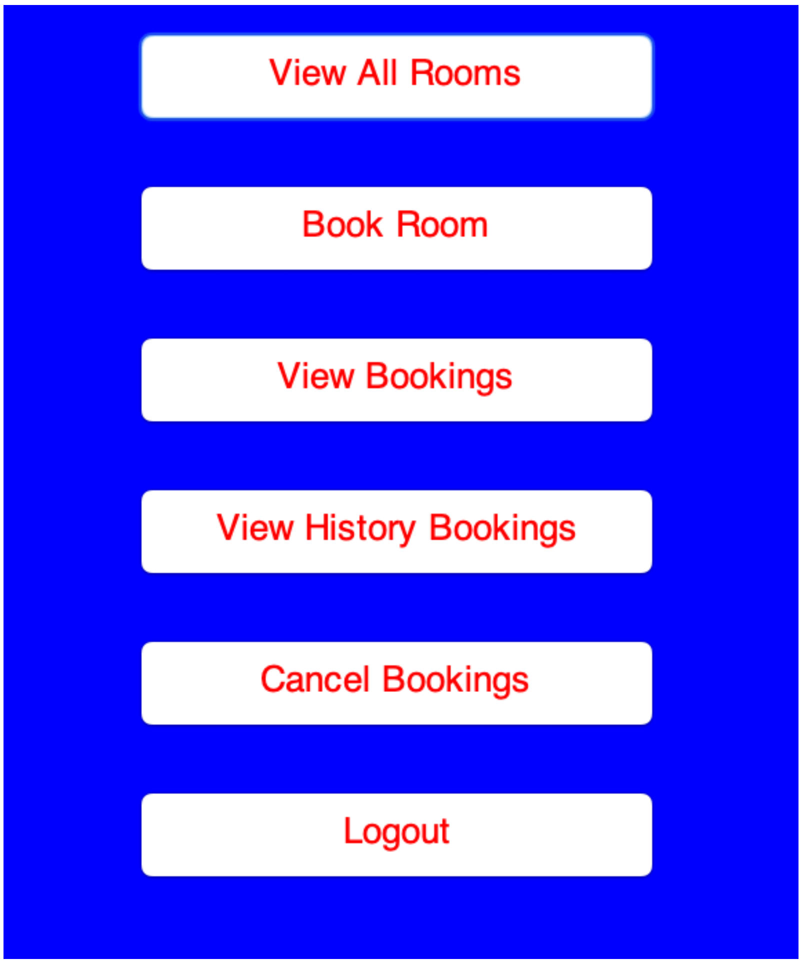
a. Login clerk

Login page for Clerk



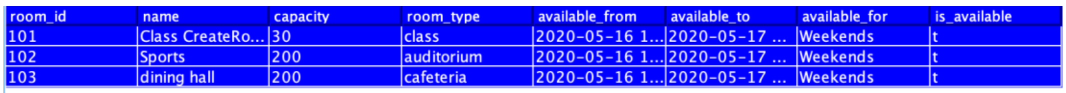
b. Clerk Menu

Menu when user logins



c. View All Rooms

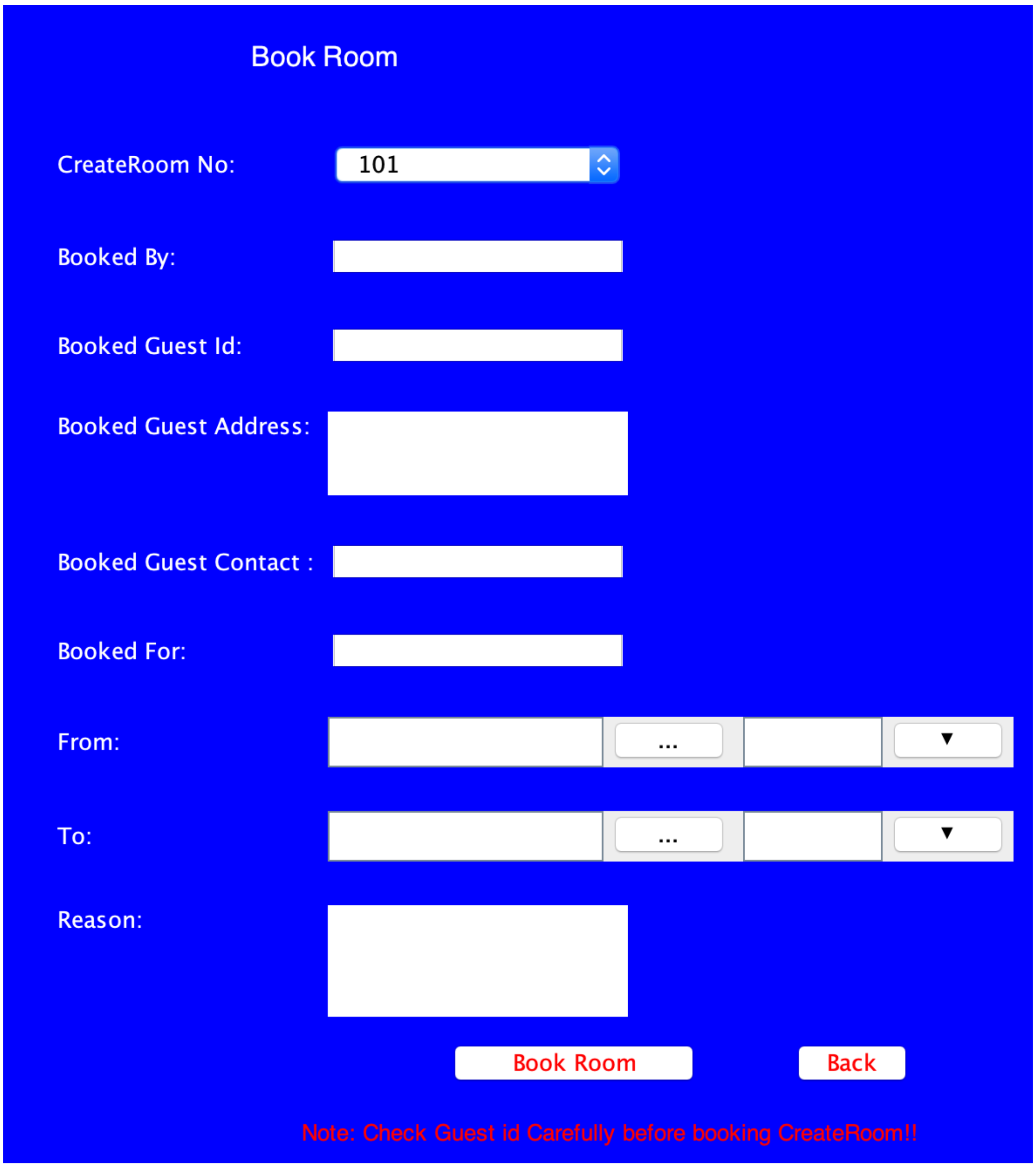
View all available rooms to Book



room_id	name	capacity	room_type	available_from	available_to	available_for	is_available
101	Class CreateRo...	30	class	2020-05-16 1...	2020-05-17 ...	Weekends	t
102	Sports	200	auditorium	2020-05-16 1...	2020-05-17 ...	Weekends	t
103	dining hall	200	cafeteria	2020-05-16 1...	2020-05-17 ...	Weekends	t

d. Book Room

Book available from Dropdown



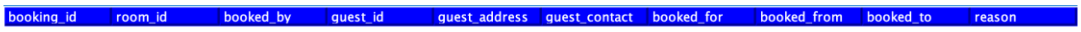
Book Room

CreateRoom No:	101
Booked By:	
Booked Guest Id:	
Booked Guest Address:	
Booked Guest Contact :	
Booked For:	
From:	
To:	
Reason:	

Note: Check Guest id Carefully before booking CreateRoom!!

e. View Bookings

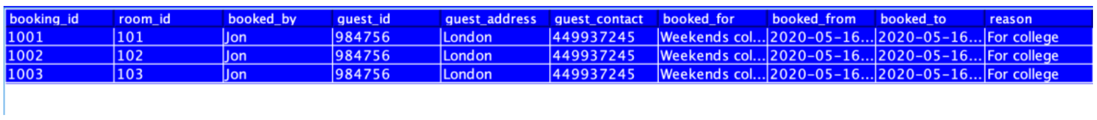
View all bookings done for future dates from now.



booking_id	room_id	booked_by	guest_id	guest_address	guest_contact	booked_for	booked_from	booked_to	reason

f. View History Booking

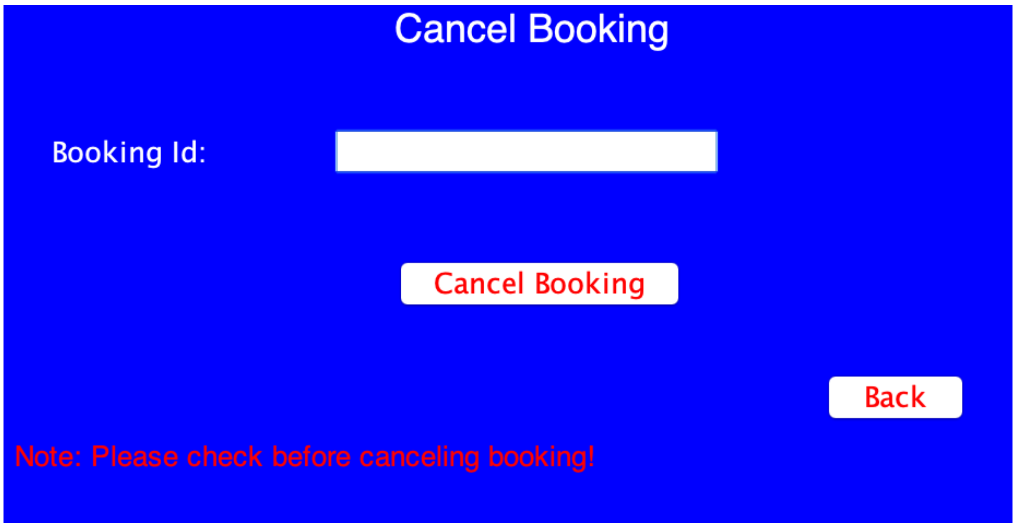
View all bookings done till now.



booking_id	room_id	booked_by	guest_id	guest_address	guest_contact	booked_for	booked_from	booked_to	reason
1001	101	Jon	984756	London	449937245	Weekends col...	2020-05-16...	2020-05-16...	For college
1002	102	Jon	984756	London	449937245	Weekends col...	2020-05-16...	2020-05-16...	For college
1003	103	Jon	984756	London	449937245	Weekends col...	2020-05-16...	2020-05-16...	For college

g. Cancel Booking

Cancel a particular booking



Note: Please check before canceling booking!

h. Logout

DataBase Diagram

1. Schema

```
CREATE DATABASE crbm;
```

2. Users

```
 CREATE TABLE crbm.users (
id int AUTO_INCREMENT PRIMARY KEY ,
name VARCHAR(100) NOT NULL,
password VARCHAR(100) NOT NULL,
email VARCHAR(100) NOT NULL,
address VARCHAR(500) NOT NULL,
city VARCHAR(100) NOT NULL,
contact VARCHAR(20) NOT NULL
);
```

Some dummy data

```
 INSERT INTO crbm.users (name, password, email, address, city, contact) VALUES
('001', '001', '001@crem.com', 'College CreateRoom Booking Management', 'London', '4412345678'),
('002', '002', '002@crem.com', 'College CreateRoom Booking Management', 'London', '4412345679'),
('003', '003', '003@crem.com', 'College CreateRoom Booking Management', 'London', '4412345680');
```

3. Rooms

```
  CREATE TABLE crbm.room (
    room_id varchar(10) PRIMARY KEY,
    name varchar(100) NOT NULL,
    capacity bigint NOT NULL,
    room_type varchar(100) NOT NULL,
    available_from TIMESTAMP NOT NULL,
    available_to TIMESTAMP NOT NULL,
    available_for  varchar(100) NOT NULL,
    is_available boolean DEFAULT true
 );
ALTER TABLE ONLY crbm.room ADD CONSTRAINT x1room UNIQUE (room_id);
```

Some Dummy data

```
INSERT INTO crbm.room (room_id, name, capacity, room_type, available_from, available_to, available_for,
is_available) VALUES
(101, 'Class CreateRoom', 30, 'class', '2020-05-16 10:00:00', '2020-05-17 20:00:00', 'Weekends', true),
(102, 'Sports', 200, 'auditorium','2020-05-16 10:00:00', '2020-05-17 20:00:00', 'Weekends', true),
(103, 'dining hall', 200, 'cafeteria','2020-05-16 10:00:00', '2020-05-17 20:00:00', 'Weekends', true);
```

4. Booking

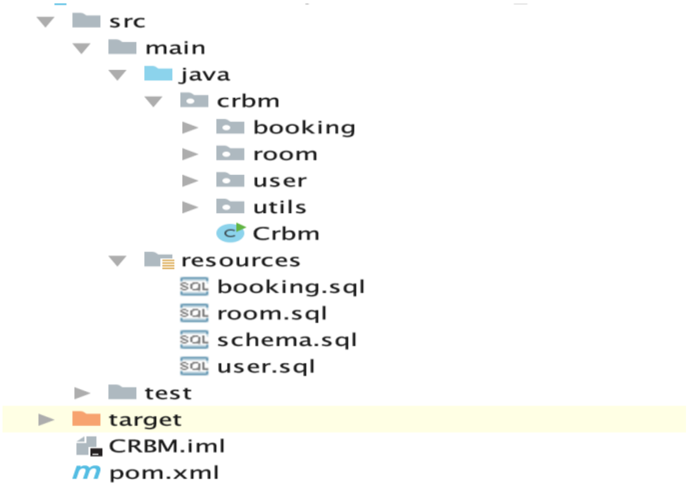
```
CREATE TABLE crbm.booking (
  booking_id int PRIMARY KEY,
  room_id varchar(10) NOT NULL,
  booked_by varchar(100) NOT NULL,
  guest_id varchar(50) NOT NULL,
  guest_address varchar(500) NOT NULL,
  guest_contact varchar(20) NOT NULL,
  booked_for varchar(200) NOT NULL,
  booked_from TIMESTAMP NOT NULL,
  booked_to TIMESTAMP NOT NULL,
  reason varchar(200) NOT NULL,
  FOREIGN KEY(room_id)
    REFERENCES crbm.room(room_id)
);
```

Some dummy data

```
INSERT INTO crbm.booking (booking_id, room_id, booked_by, guest_id, guest_address, guest_contact,
booked_for, booked_from, booked_to, reason) VALUES
(1001, 101, 'Jon', '984756', 'London', '449937245', 'Weekends college', '2020-05-16 10:00:00', '2020-05-16
10:00:00', 'For college'),
(1002, 102, 'Jon', '984756', 'London', '449937245', 'Weekends college', '2020-05-16 10:00:00', '2020-05-16
10:00:00', 'For college'),
(1003, 103, 'Jon', '984756', 'London', '449937245', 'Weekends college', '2020-05-16 10:00:00', '2020-05-16
10:00:00', 'For college');
```

Project Structure

Under Crbm :



```
▼ src
  ▼ main
    ▼ java
      ▼ crbm
        ► booking
        ► room
        ► user
        ► utils
          Crbm
    ▼ resources
        booking.sql
        room.sql
        schema.sql
        user.sql
  ► test
► target
  CRBM.iml
  pom.xml
```

Code Snippet

Utils :

ConnectionUtils.java

```
package crbm.utils;
import java.sql.Connection;
import java.sql.DriverManager;

import static crbm.utils.Constant.*;
```

```
public class ConnectionUtils {
  public static Connection getConnection(){
      Connection connection=null;
      try{
          Class.forName(DRIVER_NAME);
          connection= DriverManager.getConnection(DB_URL,DB_USER_NAME,DB_PASS);
      }catch(Exception e){
          e.printStackTrace();
      }
      return connection;
  }
}
```

Constant.java

```
package crbm.utils;

public interface Constant {
  // Data base
  String DRIVER_NAME = "org.postgresql.Driver";
  String DB_URL = "jdbc:postgresql://localhost:5432/postgres?currentSchema=crbm";
  String DB_USER_NAME = "postgres";
  String DB_PASS = "root";

  String USER_NAME = "01";
  String PASSWORD = "01";

  // FONT
  String FONT = "Helvetica";

  // Generic Action  Labels
  String LOGIN = "Login";
  String LOGOUT = "Logout";
  String BACK = "Back";
  String DELETE = "Delete";

  String TITLE = "College Room Booking Management";
  // Action Labels
  String MANAGER_LOGIN = "Manager Login";
  String MANAGER_MENU = "Menu";
  String ADD_CLERK = "Add Clerk";
  String VIEW_CLERK = "View Clerks";
  String DELETE_CLERK = "Delete Clerk";

  String CLERK_LOGIN = "Clerk Login";
  String ADD_ROOM = "Add Room";
  String VIEW_ALL_ROOMS = "View All Rooms";
  String BOOK_ROME = "Book Room";
  String VIEW_BOOKINGS = "View Bookings";
  String VIEW_HISTORY_BOOKINGS= "View History Bookings";
  String CANCEL_BOOKINGS= "Cancel Bookings";

  // Input labels
  String ENTER_NAME = "Enter Name:";
  String ENTER_PASS = "Enter Password:";
  String ENTER_ID = "Enter Id:";
  String NAME = "Name:";
```

```
    String PASS = "Password:";
    String EMAIL = "Email:";
    String ADDRESS = "Address:";
    String CITY = "City:";
    String CONTACT_NO = "Contact No:";
    String ROOM_NO = "CreateRoom No:";
    String CAPACITY = "Capacity:";
    String ROOM_TYPE = "CreateRoom Type:";
    String AVAILABLE_START_DATE = "Available Start Date:";
    String AVAILABLE_END_DATE= "Available End Date:";
    String AVAILABLE_FOR = "Available For:";
    String IS_AVAILABLE = "Is Available:";
    String CANCEL_BOOKING = "Cancel Booking";
    String BOOKING_ID = "Booking Id:";
    String BOOKED_BY = "Booked By:";
    String BOOKED_GUEST_ID = "Booked Guest Id:";
    String BOOKED_GUEST_ADDRESS = "Booked Guest Address:";
    String BOOKED_GUEST_CONTACT = "Booked Guest Contact :";
    String BOOKED_FOR = "Booked For:";
    String FROM = "From:";
    String TO = "To:";
    String REASON = "Reason:";


    // Alert Msg
    String LOGIN_INVALID = "Sorry, Username or Password Error";
    String ID_NOT_NULL = "Id can't be blank";
    String CLERK_DELETE_SUCCESS = "ClerkLogin deleted successfully!";
    String UNABLE_TO_DELETE = "Unable to delete given id!";
    String CLERK_SUCCESS = "ClerkLogin added successfully!";
    String UNABLE_SAVE = "Sorry, unable to save!";
    String ROOM_ADDED = "Rooms added successfully!";
    String BOOKING_CANCEL_SUCCESS = "Booking Canceled successfully!";
    String SORRY_UNABLE_TO_CANCEL = "Sorry, unable to Cancel booking!";


    // Alert TITLE
    String LOGIN_ERROR = "Login Error!";
    String UNABLE_RO_BOOK_ROOM = "Sorry, unable to Book CreateRoom!" ;
    String ROOM_SUCCESS = "CreateRoom Booked successfully!";
    String OVER_LAP_BOOK_ROOM = "Booking not allowed for selected Start date!" ;
    // NOTE
    String BEFORE_CANCEL = "Note: Please check before canceling booking!";
    String CHECK_GUEST = "Note: Check Guest id Carefully before booking CreateRoom!!";
    String PLS_ADD_ROOMS= "Please add Rooms!";

}
```

ContentPaneUtils

```
        package crbm.utils;

        import javax.swing.*;
        import javax.swing.border.EmptyBorder;

        public class ContentPaneUtils {

            public static JPanel getContentPane(){
                JPanel contentPane = new JPanel();
                contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
                return contentPane;
            }

        }
```

QueryConstant.java

```
    package crbm.utils;

public interface QueryConstant {

  String USER_INSERT = "insert into users(name, password, email, address, city, contact)
values(?,?,?,?,?,?)";
  String USER_DELETE = "delete from users where id=?";
  String USER_LOGIN_SELECT = "select * from users where name=? and password=?";
  String USER_SELECT = "select * from users";

  String ROOM_INSERT = "insert into room(room_id, name, capacity, room_type, available_from,
available_to, available_for, is_available) values(?,?,?,?,?,?,?,?)";
  String ROOM_SELECT_ACTIVE = "select room_id from room where is_available=?";
  String BOOKING_INSERT = "insert into booking(room_id, booked_by, guest_id, guest_address,
guest_contact, booked_for, booked_from, booked_to, reason) values(?,?,?,?,?,?,?,?,?)";
  String BOOKING_DELETE = "delete from booking where booking_id=?";

  String ROOM_BOOKING_VALIDATE1 = "select * from booking a,  room b WHERE ? < b.available_to
AND a.room_id = ? ";
  String ROOM_BOOKING_VALIDATE2 = "select * from booking WHERE ? < booked_to AND room_id =
? ";

  String ROOM_SELECT = "select * from room ";
  String ROOM_PAST_SELECT = "select * from booking where booked_to <= now()";
  String ROOM_CURRENT_SELECT = "select * from booking where booked_to >= now()";

}
```

Main Application:

Crbm.java

```
         package crbm;

        import crbm.user.clerk.ClerkLogin;
        import crbm.user.admin.AdminLogin;
        import crbm.utils.ContentPaneUtils;
        import crbm.utils.GroupLayoutUtils;

        import javax.swing.*;
        import java.awt.*;

        import static crbm.utils.Constant.*;

        public class Crbm extends JFrame {
          private static Crbm crbmFrame;
          private JPanel contentPane;
          /**
           * Create frame.
           */
          public Crbm() {
```

```
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 450, 300);
        contentPane = ContentPaneUtils.getContentPane();
        setContentPane(contentPane);
        addContent();
    }
    /**
     * Launch the CRBM App.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(() -> {
            try {
                crbmFrame = new Crbm();
                crbmFrame.getContentPane().setBackground(Color.MAGENTA);
                crbmFrame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        });
        System.out.println("Application is Launched!!");
    }

    public void addContent(){
        //
        JLabel camManagementLabel = new JLabel(TITLE);
        camManagementLabel.setFont(new Font(FONT, Font.PLAIN, 18));
        camManagementLabel.setForeground(Color.WHITE);
        //
        JButton adminLoginButton = new JButton(MANAGER_LOGIN);
        adminLoginButton.addActionListener(e -> {
            AdminLogin.main(new String[]{});
            crbmFrame.dispose();
        });
        adminLoginButton.setFont(new Font(FONT, Font.PLAIN, 15));
        adminLoginButton.setForeground(Color.RED);

        JButton clerkLoginButton = new JButton(CLERK_LOGIN);
        clerkLoginButton.addActionListener(arg0 -> {
            ClerkLogin.main(new String[]{});
            crbmFrame.dispose();
        });
        clerkLoginButton.setFont(new Font(FONT, Font.PLAIN, 15));
        clerkLoginButton.setForeground(Color.RED);

        // Set group Layout to Pane
        contentPane.setLayout(GroupLayoutUtils.addGrpLayoutForMainApp(contentPane,
camManagementLabel, adminLoginButton, clerkLoginButton));
    }

}
```

BookingDAO.java

```
  package crbm.booking;
import crbm.utils.ConnectionUtils;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Timestamp;
import java.util.ArrayList;
import java.util.List;

import static crbm.utils.QueryConstant.ROOM_SELECT_ACTIVE;
import static crbm.utils.QueryConstant.ROOM_BOOKING_VALIDATE1;
import static crbm.utils.QueryConstant.ROOM_BOOKING_VALIDATE2;
import static crbm.utils.QueryConstant.BOOKING_INSERT;
import static crbm.utils.QueryConstant.BOOKING_DELETE;

public class BookingDAO {

  public static List<String> getAvilableRooms() {
    List<String> roomIds = new ArrayList();
    try {
      Connection connection = ConnectionUtils.getConnection();
      PreparedStatement preparedStatement =
connection.prepareStatement(ROOM_SELECT_ACTIVE);
      preparedStatement.setBoolean(1, true);
      ResultSet resultSet = preparedStatement.executeQuery();
      while (resultSet.next()) {
        roomIds.add(resultSet.getString("room_id"));
      }
      connection.close();
    } catch (Exception e) {
      System.out.println(e);
    }
    return roomIds;
  }

  public static Boolean validateBooking1(Timestamp bookedFrom, String bookingId) {
    Boolean isAllowed = true;
    try {
      Connection connection = ConnectionUtils.getConnection();
      PreparedStatement preparedStatement =
connection.prepareStatement(ROOM_BOOKING_VALIDATE1);
      preparedStatement.setTimestamp(1, bookedFrom);
      preparedStatement.setString(2, bookingId);
      ResultSet resultSet = preparedStatement.executeQuery();
      if(!resultSet.next()){
        isAllowed = false;
      }
      connection.close();
    } catch (Exception e) {
      System.out.println(e);
    }
    return isAllowed;
  }

  public static Boolean validateBooking2(Timestamp bookedFrom, String bookingId) {
    Boolean isAllowed = true;
    try {
      Connection connection = ConnectionUtils.getConnection();
      PreparedStatement preparedStatement =
connection.prepareStatement(ROOM_BOOKING_VALIDATE2);
      preparedStatement.setTimestamp(1, bookedFrom);
```

```
            preparedStatement.setString(2, bookingId);
            ResultSet resultSet = preparedStatement.executeQuery();
            if(!resultSet.next()){
                isAllowed = false;
            }
            connection.close();
        } catch (Exception e) {
            System.out.println(e);
        }
        return isAllowed;
    }

    public static int save(String roomid, String bookedby, String guestid, String guestaddress, String
guestcontact, String bookedfor,
                           Timestamp bookedfrom, Timestamp bookedto, String reason) {
        int status = 0;
        try {
            Connection connection = ConnectionUtils.getConnection();
            PreparedStatement preparedStatement = connection.prepareStatement(BOOKING_INSERT);
            preparedStatement.setString(1, roomid);
            preparedStatement.setString(2, bookedby);
            preparedStatement.setString(3, guestid);
            preparedStatement.setString(4, guestaddress);
            preparedStatement.setString(5, guestcontact);
            preparedStatement.setString(6, bookedfor);
            preparedStatement.setTimestamp(7, bookedfrom);
            preparedStatement.setTimestamp(8, bookedto);
            preparedStatement.setString(9, reason);
            status = preparedStatement.executeUpdate();
            connection.close();
        } catch (Exception e) {
            System.out.println(e);
        }
        return status;
    }

    public static int delete(String bookingId) {
        int status = 0;
        try {
            Connection connection = ConnectionUtils.getConnection();
            if (status > 0) {
                PreparedStatement preparedStatement = connection.prepareStatement(BOOKING_DELETE);
                preparedStatement.setLong(1, Long.parseLong(bookingId));
                status = preparedStatement.executeUpdate();
            }
            connection.close();
        } catch (Exception e) {
            System.out.println(e);
        }
        return status;
    }
}
```

Conclusion

"It was a wonderful learning experience for me while working on this project. This project took me through the various phases of project development and gave me real insight into the world of software engineering. The joy of working and the thrill involved while tackling the various problems and challenges gave me a feel of the developers' industry.

It was due to this project I came to know how professional software is designed."