**OOPS WITH JAVA**

**CS23333**

**MINI PROJECT**

**Hotel Management System**

DONE BY:

NAME: R.NAVEENKARTHIK

ROLL NUMBER: 231401072

CLASS: CSBS-B

**AIM:**  
The purpose of this Hotel Management System is to allow users to manage room availability, book rooms for guests, check guests in and out, and display room statuses. This system demonstrates the use of OOP principles such as encapsulation and collections to organize and manage hotel operations effectively.

**ALGORITHM:**

1. Start.
2. Display the main menu with the following options:
   * Add Room
   * Book Room
   * Check-In
   * Check-Out
   * View Room Status
   * Exit
3. Accept the user’s choice.
4. Perform the corresponding operation:
   * Add Room:
     1. Input room number and room type (e.g., Single, Double, Suite).
     2. Create a new room object.
     3. Add the room to the collection of rooms.
   * Book Room:
     1. Input guest name and desired room type.
     2. Search for an available room of the specified type in the collection.
     3. Assign the guest to the room if available and mark it as booked.
   * Check-In:
     1. Input room number and guest details.
     2. Verify the room has been booked.
     3. Mark the room as checked-in.
   * Check-Out:
     1. Input room number.
     2. Mark the room as available and remove the guest details.
   * View Room Status:
     1. Display the list of all rooms with their status (e.g., Available, Booked, Checked-In).
   * Exit:
     1. Terminate the program.
5. Repeat until the user chooses Exit.
6. End.

**Class Design Sugg**

**PROGRAM:**

**CODE:**

import java.util.ArrayList;

import java.util.Scanner;

class Room {

int roomNumber;

String roomType;

boolean isBooked;

Room(int roomNumber, String roomType) {

this.roomNumber = roomNumber;

this.roomType = roomType;

this.isBooked = false;

}

void bookRoom() {

if (!isBooked) {

isBooked = true;

System.out.println("Room " + roomNumber + " has been successfully booked!");

} else {

System.out.println("Room " + roomNumber + " is already booked!");

}

}

void displayRoomDetails() {

System.out.println("Room Number: " + roomNumber + ", Type: " + roomType + ", Booked: " + isBooked);

}

}

class HotelManagement {

ArrayList<Room> rooms = new ArrayList<>();

void addRoom(int roomNumber, String roomType) {

rooms.add(new Room(roomNumber, roomType));

System.out.println("Room " + roomNumber + " added successfully.");

}

void bookRoom(int roomNumber) {

for (Room room : rooms) {

if (room.roomNumber == roomNumber) {

room.bookRoom();

return;

}

}

System.out.println("Room " + roomNumber + " not found!");

}

void displayRooms() {

if (rooms.isEmpty()) {

System.out.println("No rooms available!");

return;

}

for **(**Room room : rooms) {

room.displayRoomDetails();

}

}

}

public class HotelManagementSystem {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

HotelManagement hotel = new HotelManagement();

while (true) {

System.out.println("\n--- Hotel Management System ---");

System.out.println("1. Add Room");

System.out.println("2. Book Room");

System.out.println("3. Display Rooms");

System.out.println("4. Exit");

System.out.print("Enter your choice: ");

int choice = scanner.nextInt();

switch (choice) {

case 1:

System.out.print("Enter room number: ");

int roomNumber = scanner.nextInt();

System.out.print("Enter room type (Single/Double): ");

String roomType = scanner.next();

hotel.addRoom(roomNumber, roomType);

break;

case 2:

System.out.print("Enter room number to book: ");

int roomToBook = scanner.nextInt();

hotel.bookRoom(roomToBook);

break;

case 3:

hotel.displayRooms();

break;

case 4:

System.out.println("Exiting...");

scanner.close();

System.exit(0);

break;

default:

System.out.println("Invalid choice! Try again.");

}

}

}

}

**OUTPUT:**

--- Hotel Management System ---

1. Add Room

2. Book Room

3. Display Rooms

4. Exit

Enter your choice: 1

Enter room number: 101

Enter room type (Single/Double): Single

Room 101 added successfully.

--- Hotel Management System ---

1. Add Room

2. Book Room

3. Display Rooms

4. Exit

Enter your choice: 1

Enter room number: 102

Enter room type (Single/Double): Double

Room 102 added successfully.

--- Hotel Management System ---

1. Add Room

2. Book Room

3. Display Rooms

4. Exit

Enter your choice: 3

Room Number: 101, Type: Single, Booked: false

Room Number: 102, Type: Double, Booked: false

--- Hotel Management System ---

1. Add Room

2. Book Room

3. Display Rooms

4. Exit

Enter your choice: 2

Enter room number to book: 101

Room 101 has been successfully booked!

--- Hotel Management System ---

1. Add Room

2. Book Room

3. Display Rooms

4. Exit

Enter your choice: 3

Room Number: 101, Type: Single, Booked: true

Room Number: 102, Type: Double, Booked: false

--- Hotel Management System ---

1. Add Room

2. Book Room

3. Display Rooms

4. Exit

Enter your choice: 2

Enter room number to book: 101

Room 101 is already booked!

--- Hotel Management System ---

1. Add Room

2. Book Room

3. Display Rooms

4. Exit

Enter your choice: 4

Exiting...

**CONCLUSION:**

The Hotel Management System successfully demonstrates the practical application of Object-Oriented Programming (OOP) concepts, such as encapsulation, inheritance, and collections, to manage hotel operations effectively. This project highlights the importance of using structured algorithms, clear code organization, and intuitive user interfaces to create a reliable and user-friendly software solution. By addressing real-world requirements like room booking, check-ins, and status management, this system proves the efficiency and scalability of OOP principles in solving complex problems in a maintainable and robust way.