**Hotel Management System Using Java.**

**Introduction:**

Welcome to Hotel Management System using Java! A Hotel Management System is an application used to manage various activities and operations of a hotel. It helps to automate many of the processes involved in running a hotel, such as reservations, check-ins and check-outs and room assignments. This system works on admin side where admin can check-ins and check-outs the guests based on room availability. Overall, a hotel management system can help to streamline and automate many of the day-to-day operations of a hotel, which can lead to improved efficiency, increased revenue, and better customer service. Main objective of the system is to make the hectic method of pen-paper simple and brings the management of hotel on the fingertips of admin.

**Explanation :**

The Hotel Management System we will build will be a simple console-based application that allows users to check-ins, check-outs and view the availability of rooms. We have used single class in program to make it as easy as possible. Main class consists of main method, switch case and other required methods for check-ins and check-outs. For rooms 2D array has been used to store room number and guest name in that room. A 2D array in Java is a multidimensional array that represents a table or matrix of values. It is an array of arrays, where each element of the array is itself an array. This array has been initialized as room numbers from 1 to 10 and guest name as empty which will get updated as per check-ins of guests. Menu will provide four options as – check-in, check-out, view occupancy and exit. Scanner class used for taking input from user through keyboard. For check-in user needs to provide room number. Given room number will get checked as it is valid or not and if that room is empty then system will ask for guest name. Check-out process is based on room number. Again, the conditions will get checked for room validation. If room number is valid then assign that room as ‘empty’. For viewing the occupancy simple for loop to retrieve data of 2D array has been used. As array starts from index 0, we need to minus one from provided input of room number. Checking conditions properly for each operation is important to make this system transparent. Switch case shows the menu options and each case of switch case, the appropriate method has been called to perform the required operations. Just some basic knowledge of arrays and for loops and you are ready to build your own hotel Management system **Source Code:-**

import java.util.ArrayList; import java.util.Scanner;

class Room {

private int roomNumber; private boolean isBooked;

public Room(int roomNumber) {

this.roomNumber = roomNumber; this.isBooked = false;

}

public int getRoomNumber() {

return roomNumber;

}

public boolean isBooked() {

return isBooked;

}

public void book() { if (!isBooked) {

isBooked = true;

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

} else {

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

}

}

public void checkout() {

if (isBooked) {

isBooked = false;

System.out.println("Room " + roomNumber + " has been checked out.");

} else {

System.out.println("Room " + roomNumber + " is not currently booked.");

}

}

}

class Hotel {

private ArrayList<Room> rooms;

public Hotel() {

rooms = new ArrayList<>();

}

public void addRoom(int roomNumber) {

rooms.add(new Room(roomNumber));

System.out.println("Room " + roomNumber + " added to the hotel.");

}

public void bookRoom(int roomNumber) {

for (Room room : rooms) {

if (room.getRoomNumber() == roomNumber) {

room.book(); return;

}

}

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

}

public void checkoutRoom(int roomNumber) {

for (Room room : rooms) {

if (room.getRoomNumber() == roomNumber) {

room.checkout(); return;

}

}

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

}

public void listRooms() { for (Room room : rooms) {

System.out.println("Room " + room.getRoomNumber() + " - " + (room.isBooked() ? "Booked" : "Available"));

}

}

public static void main(String[] args) {

Hotel hotel = new Hotel();

Scanner scanner = new Scanner(System.in);

while (true) {

System.out.println("Hotel Management System:");

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

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

System.out.println("3. Checkout Room");

System.out.println("4. List Rooms");

System.out.println("5. Exit"); System.out.print("Choose an option: "); int choice = scanner.nextInt();

switch (choice) {

case 1:

System.out.print("Enter room number to add: "); int roomNumberToAdd = scanner.nextInt(); hotel.addRoom(roomNumberToAdd); break;

case 2:

System.out.print("Enter room number to book: "); int roomNumberToBook = scanner.nextInt(); hotel.bookRoom(roomNumberToBook); break;

case 3:

System.out.print("Enter room number to checkout: "); int roomNumberToCheckout = scanner.nextInt(); hotel.checkoutRoom(roomNumberToCheckout); break;

case 4:

hotel.listRooms(); break;

case 5:

System.out.println("Exiting..."); scanner.close(); return;

default:

System.out.println("Invalid option. Please try again.");

}

}

}

}

**Output :-**

Hotel Management System:

1. Add Room
2. Book Room
3. Checkout Room
4. List Rooms
5. Exit

Choose an option: 1 Enter room number to add: 101 Room 101 added to the hotel.

Hotel Management System:

1. Add Room
2. Book Room
3. Checkout Room
4. List Rooms
5. Exit

Choose an option: 1 Enter room number to add: 102 Room 102 added to the hotel.

Hotel Management System:

1. Add Room
2. Book Room
3. Checkout Room
4. List Rooms 5. Exit

Choose an option: 2 Enter room number to book: 101 Room 101 has been booked.

Hotel Management System:

1. Add Room
2. Book Room
3. Checkout Room
4. List Rooms
5. Exit

Choose an option: 4

Room 101 - Booked

Room 102 - Available Hotel Management System:

1. Add Room
2. Book Room
3. Checkout Room
4. List Rooms 5. Exit

Choose an option: 3 Enter room number to checkout: 101 Room 101 has been checked out.

Hotel Management System:

1. Add Room
2. Book Room
3. Checkout Room
4. List Rooms
5. Exit

Choose an option: 4

Room 101 - Available

Room 102 - Available Hotel Management System:

1. Add Room
2. Book Room
3. Checkout Room
4. List Rooms
5. Exit

Choose an option: 5

Exiting...

**Conclusion:-**

This output demonstrates the following interactions:

Adding two rooms (101 and 102) to the hotel.

Booking room 101.

Listing the rooms to show that room 101 is booked and room 102 is available.

Checking out of room 101.

Listing the rooms again to show that both rooms are now available.

Exiting the system.