HOTEL MANAGEMENT

**1.MAIN MENU:**

#include <iostream>

int main() {

int choice;

while (true) {

// Display main menu

std::cout << "Hotel Management System" << std::endl;

std::cout << "1. Check-in" << std::endl;

std::cout << "2. Check-out" << std::endl;

std::cout << "3. View Guest List" << std::endl;

std::cout << "4. Search for a Guest" << std::endl;

std::cout << "5. Exit" << std::endl;

std::cout << "Enter your choice: ";

std::cin >> choice;

switch(choice) {

case 1:

// Call check-in function

break;

case 2:

// Call check-out function

break;

case 3:

// Call view guest list function

break;

case 4:

// Call search function

break;

case 5:

std::cout << "Exiting program..." << std::endl;

return 0;

default:

std::cout << "Invalid choice. Please enter a number between 1 and 5." << std::endl;

}

}

return 0;

}

2.**GUEST CLASS:**

#include <string>

class Guest {

private:

std::string name;

int roomNumber;

public:

Guest(std::string \_name, int \_roomNumber) : name(\_name), roomNumber(\_roomNumber) {}

std::string getName() { return name; }

int getRoomNumber() { return roomNumber; }

};

**3.ROOM CLASS:**

class Room {

private:

int roomNumber;

bool available;

public:

Room(int \_roomNumber) : roomNumber(\_roomNumber), available(true) {}

bool isAvailable() { return available; }

void setAvailability(bool status) { available = status; }

};

**4.HOTEL CLASS:**

#include <vector>

class Hotel {

private:

std::vector<Room> rooms;

std::vector<Guest> guests;

public:

void checkIn(Guest guest, int roomNumber) {

// Implement check-in logic

}

void checkOut(int roomNumber) {

// Implement check-out logic

}

void viewGuestList() {

// Implement logic to display guest list

}

Guest searchGuest(std::string name) {

// Implement search logic

}

};

**CODE FOR THE FUNCTIONS INSIDE THE HOTEL CLASS**

#include <iostream>

#include <vector>

#include <algorithm> // for std::find\_if

class Guest {

private:

std::string name;

int roomNumber;

public:

Guest(std::string \_name, int \_roomNumber) : name(\_name), roomNumber(\_roomNumber) {}

std::string getName() { return name; }

int getRoomNumber() { return roomNumber; }

};

class Room {

private:

int roomNumber;

bool available;

public:

Room(int \_roomNumber) : roomNumber(\_roomNumber), available(true) {}

bool isAvailable() { return available; }

void setAvailability(bool status) { available = status; }

};

class Hotel {

private:

std::vector<Room> rooms;

std::vector<Guest> guests;

public:

// Function to check-in a guest

void checkIn(const std::string& guestName, int roomNumber) {

// Check if the room is available

auto it = std::find\_if(rooms.begin(), rooms.end(), [&](const Room& room) {

return room.getRoomNumber() == roomNumber && room.isAvailable();

});

if (it != rooms.end()) {

it->setAvailability(false); // Set the room as occupied

// Add the guest to the guest list

guests.push\_back(Guest(guestName, roomNumber));

std::cout << "Guest " << guestName << " has been checked in to room " << roomNumber << std::endl;

} else {

std::cout << "Room " << roomNumber << " is not available." << std::endl;

}

}

// Function to check-out a guest

void checkOut(int roomNumber) {

auto it = std::find\_if(guests.begin(), guests.end(), [&](const Guest& guest) {

return guest.getRoomNumber() == roomNumber;

});

if (it != guests.end()) {

// Find the room and mark it as available

auto roomIt = std::find\_if(rooms.begin(), rooms.end(), [&](const Room& room) {

return room.getRoomNumber() == roomNumber;

});

if (roomIt != rooms.end()) {

roomIt->setAvailability(true);

}

guests.erase(it); // Remove the guest from the guest list

std::cout << "Guest checked out from room " << roomNumber << std::endl;

} else {

std::cout << "No guest found in room " << roomNumber << std::endl;

}

}

// Function to view guest list

void viewGuestList() {

std::cout << "Guest List:" << std::endl;

for (const auto& guest : guests) {

std::cout << "Name: " << guest.getName() << ", Room: " << guest.getRoomNumber() << std::endl;

}

}

// Function to search for a guest by name

void searchGuest(const std::string& name) {

auto it = std::find\_if(guests.begin(), guests.end(), [&](const Guest& guest) {

return guest.getName() == name;

});

if (it != guests.end()) {

std::cout << "Guest found - Name: " << it->getName() << ", Room: " << it->getRoomNumber() << std::endl;

} else {

std::cout << "Guest not found." << std::endl;

}

}

};

int main() {

Hotel hotel;

hotel.checkIn("John Doe", 101);

hotel.checkIn("Alice Smith", 102);

hotel.checkIn("Bob Brown", 103);

hotel.viewGuestList();

hotel.checkOut(101);

hotel.viewGuestList();

hotel.searchGuest("Alice Smith");

return 0;

}