

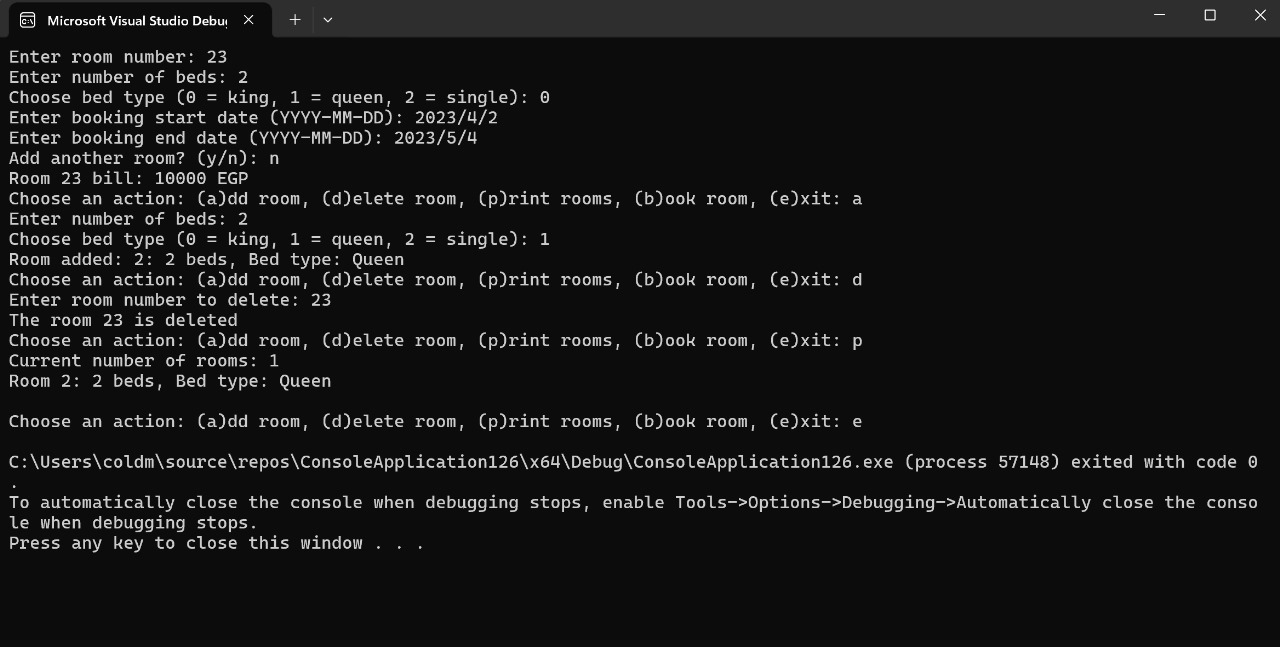
2024

PRESENTED BY: FARIDA HESHAM 2023/00935 MARIAM MOHAMED FARAG 2023/04412 MAYAR AHMED KAMAL 2023/05693 JULIE GAMAL 2023/00892

JMMF HOTEL

**INTRODUCTION**

OUR PROJECT PERFORMS THE HOTEL MANAGEMENT SYSTEM. THE SYSTEM IS BASICALLY A BUNCH OF CODES ALIGNS TOGETHER TO RUN THAT PROGRAM.



THE FIRST STEP:

The system defines three types of beds: Single, Queen, and King, using an enumeration. It utilizes a Booking struct to store booking details, including a unique booking ID, bed type, start date, and end date. The program maintains an array to keep track of all bookings and ensures that the number of bookings does not exceed a defined maximum.

SECOND STEP:

To book a room, the system first checks availability using the is “Room Available” function, which iterates through existing bookings to see if the requested bed type is available for the specified dates. If the room is available, the “book Room” function generates a unique booking ID, creates a new booking, and adds it to the array, displaying the booking details to the user. If the room is not available, it informs the user accordingly. The main function drives the program by prompting the user to select a bed type, enter booking dates, and then attempts to book the room, ensuring all interactions and checks are handled appropriately.

THE BED SELECTION:

THEBOOKING PROGRESS:

Input: 1(Single)

Output: Enter the start date for booking(YEAR-MONTH-DATE):

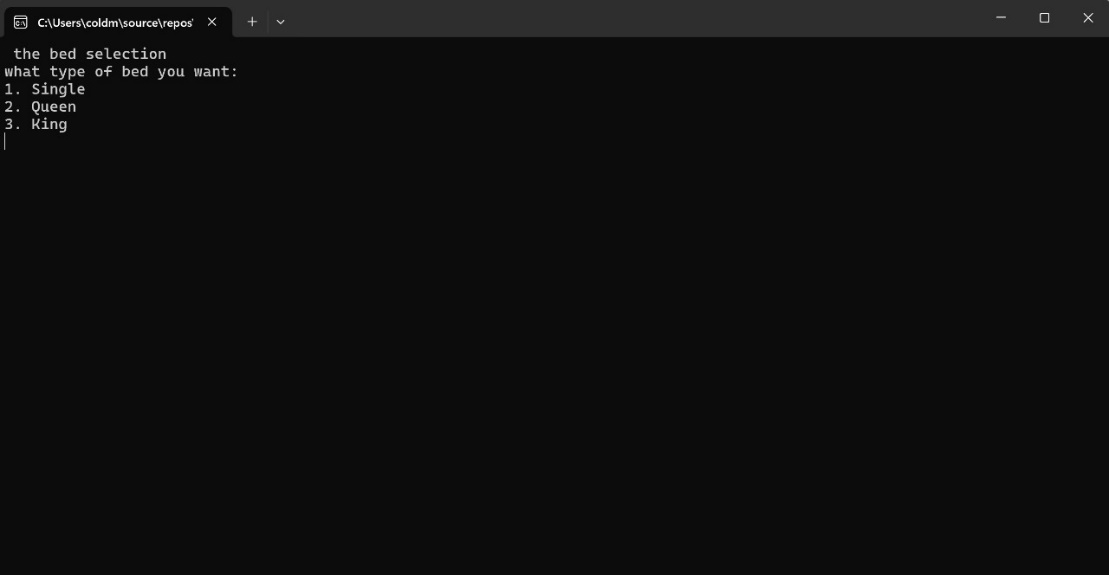
Input: 1/1/2023

Enter the end date for booking(YEAR-MONTH-DATE):

Input: 2/2/2024

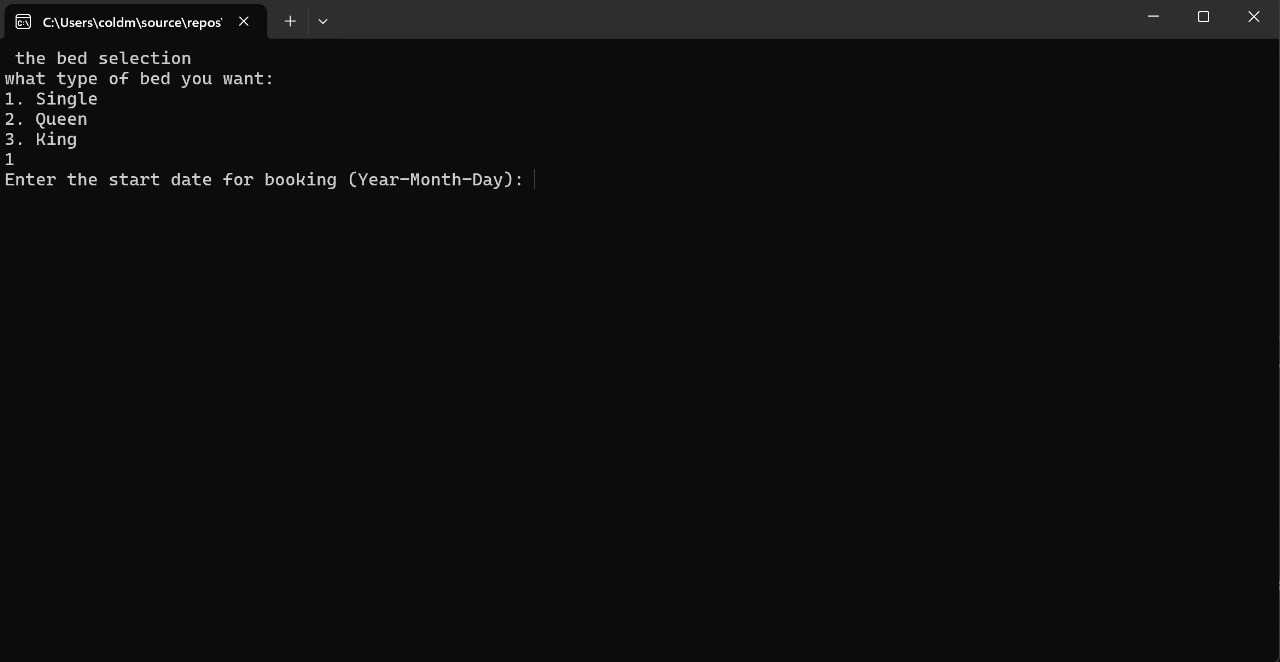
Output: Booking ID: 81

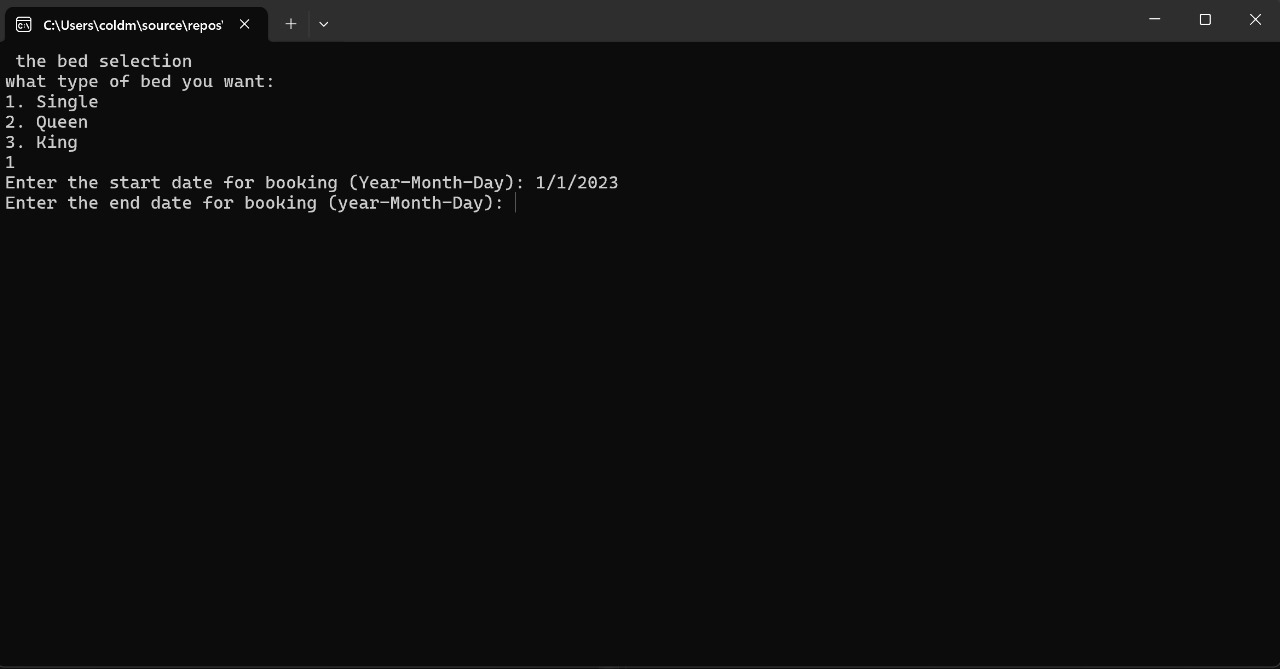
Final output is: Booking a room from 1/1/2023 with a single bed. Room booked Successfully!



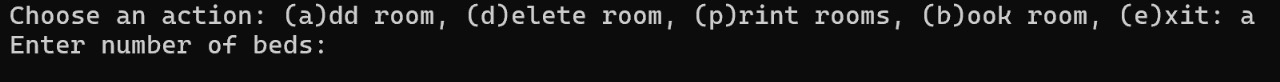
The program before inserting the inputs.

The Bed selection:

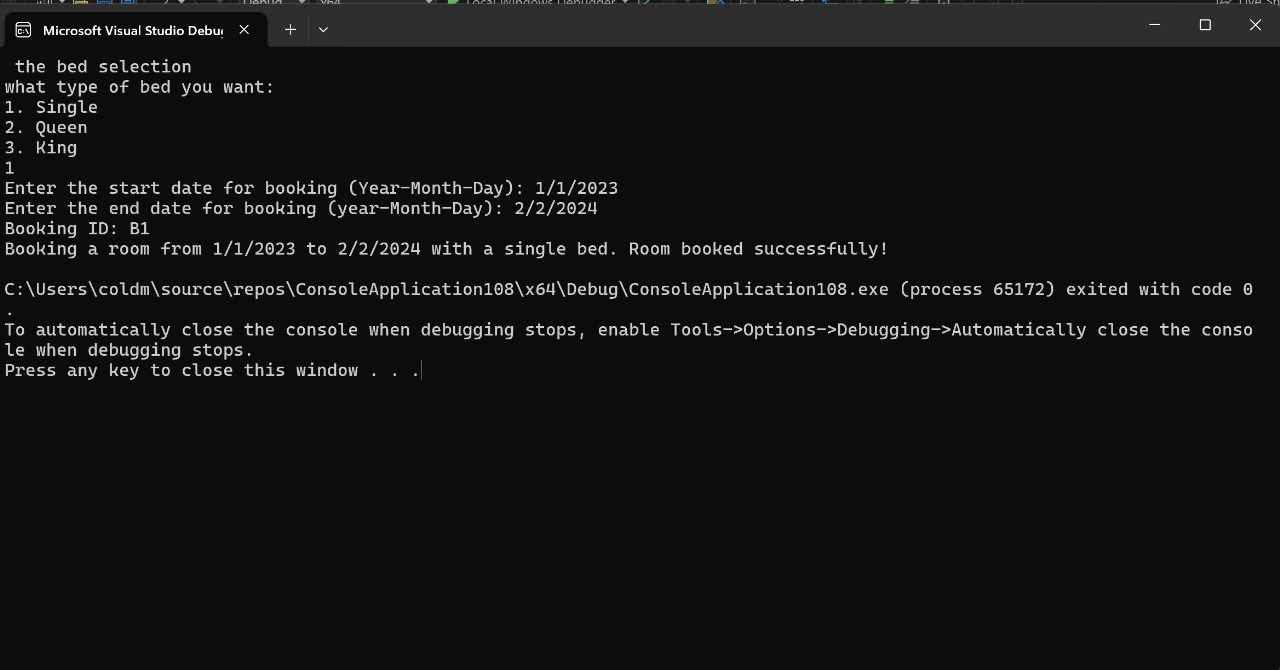


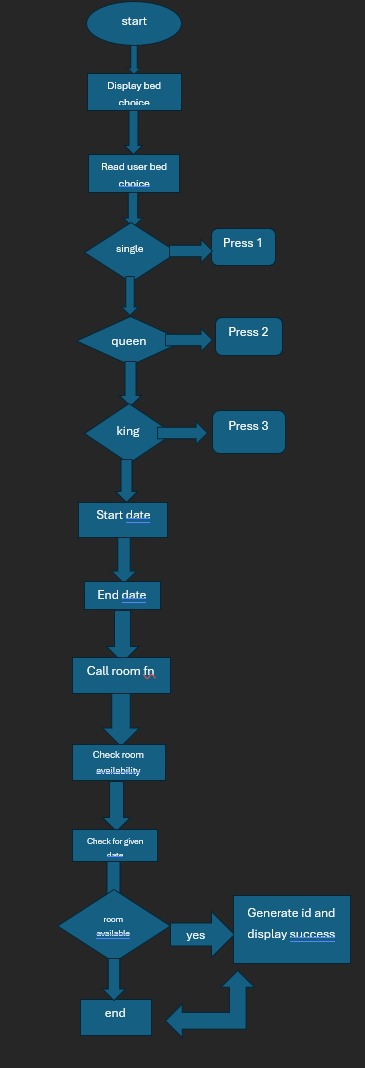
The booking Progress: 

THE ADDING AND DELETING ROOMS:



The final output with booking progress:





THIS A FLOWCART PRESENTING THE SECOND STEP PROCESS.

THIRD STEP:

It begins by including necessary header files and defining a struct named “Hotel Guest “to hold guest information such as name and room number. Global variables are declared to manage guest data and room allocation. Two functions,” check in ()” and “checkout ()”, handle the process of guest check-in and check-out respectively. The “main ()” function presents a menu to the user, prompting them to choose between check-in, check-out, or exiting the program. Depending on the user's choice, the corresponding function is called. The program runs within a loop until the user decides to exit.

First step the check in ()

Input:1

Output: Enter guest name: maro

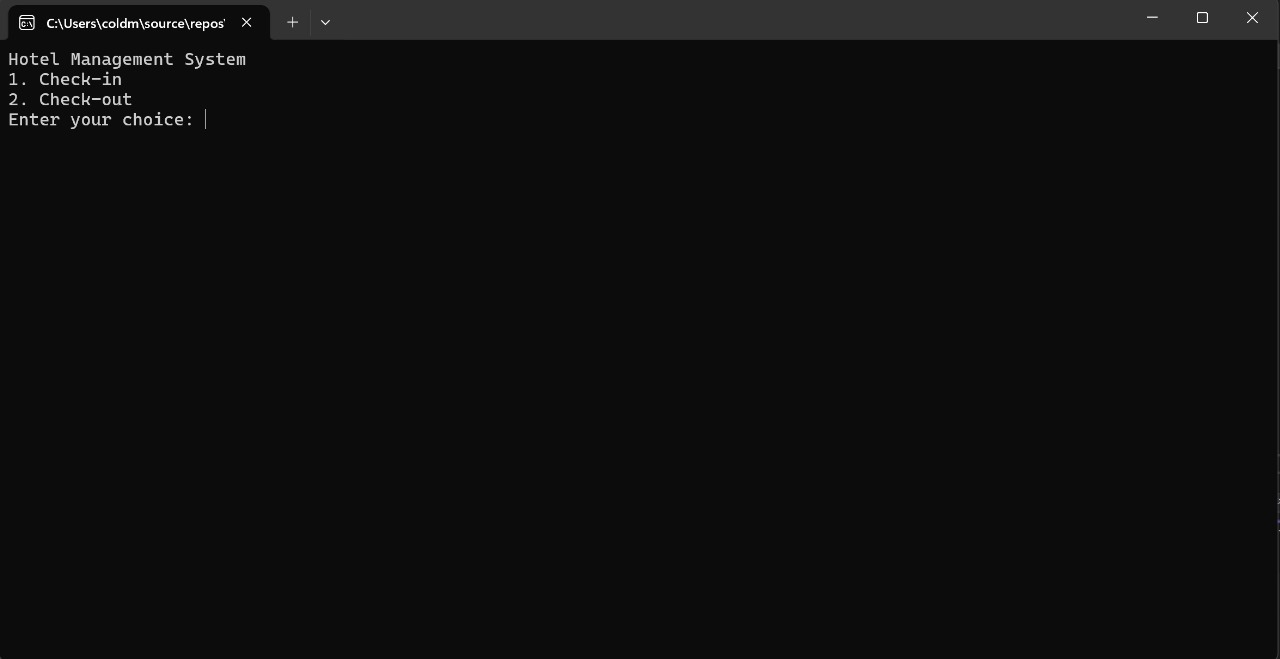
Check-in successful for room number:1

Second Step the checkout ():

Input: 2

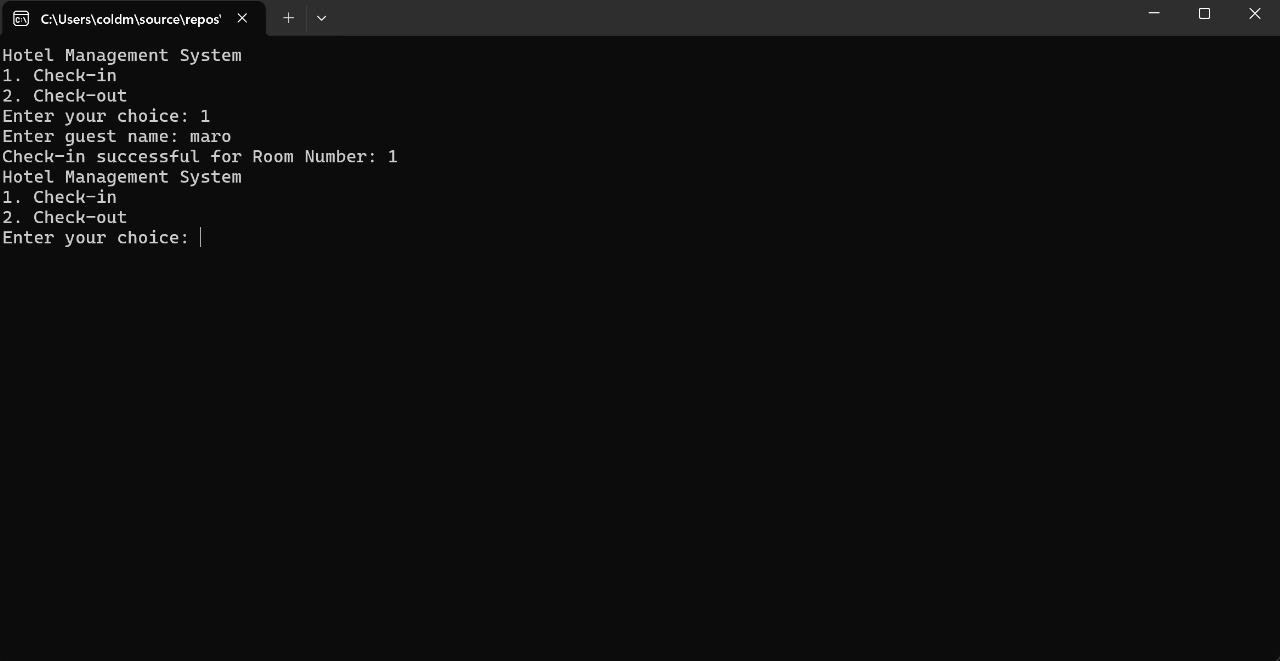
Output: Enter the room number to checkout: 1

Check-Out Successful.

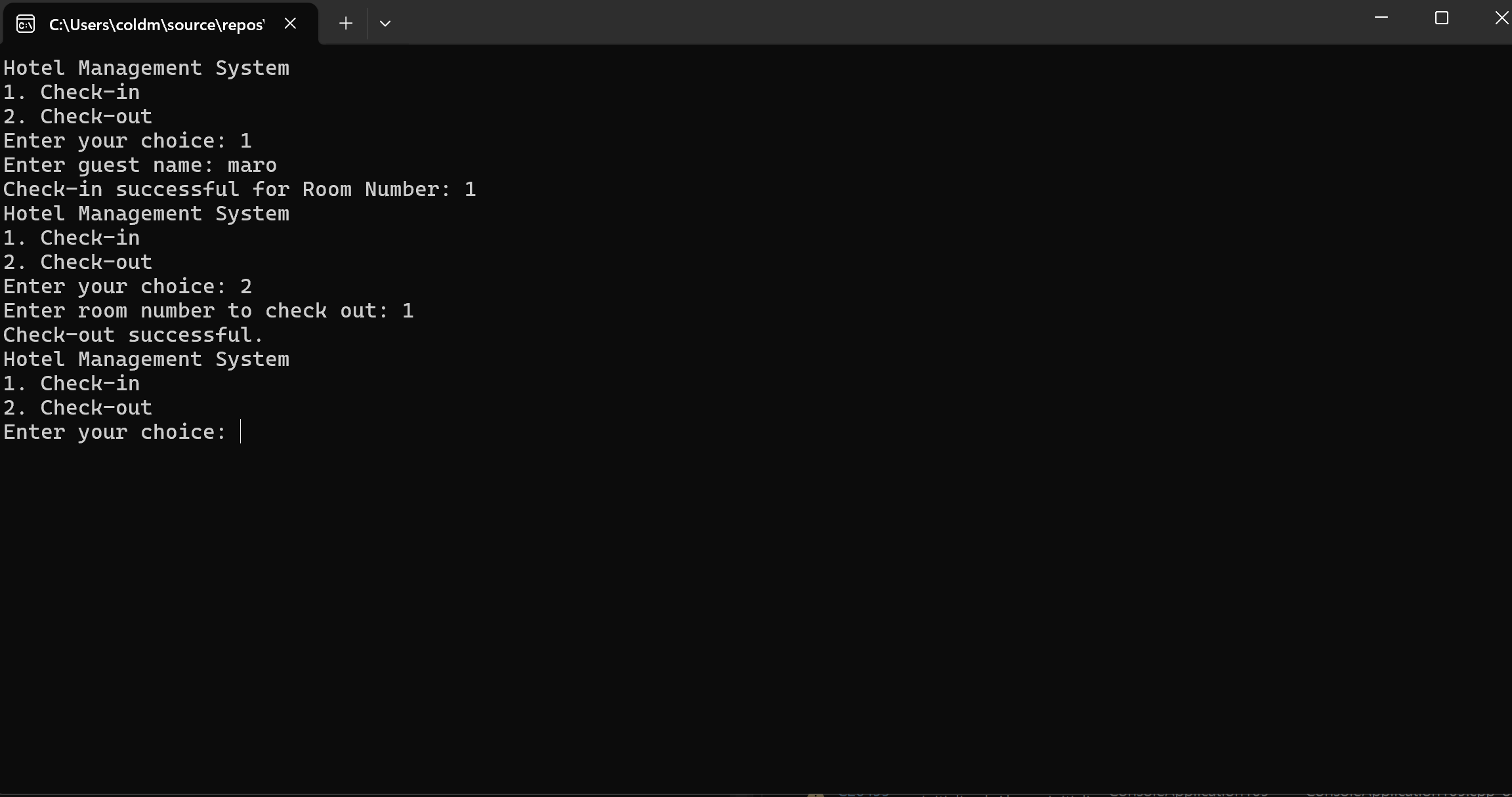


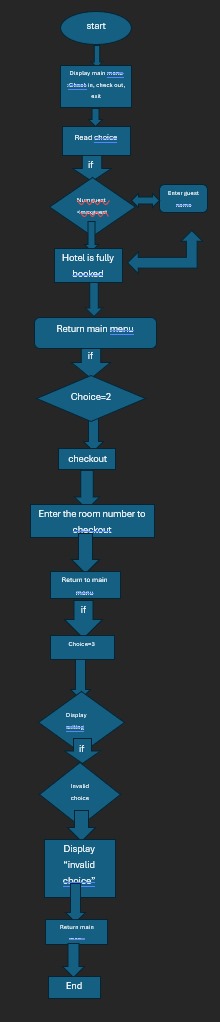
The program before entering the inputs.

The check In step:



The check Out step:





THIS IS A FLOWCHART PRESENTING THE THIRD STEP PROCESS.

FOURTH STEP:

It consists of functions to save and load room inventory from a file, calculate the bill for each room based on booking dates and price per night, and get the type of bed from the user.

The “savetoinventory” function writes the room inventory details, including booking start and end dates, room number, number of beds, bed type, and price per night, to a file named "hotel.txt". The” loadinventoryfromfile” function reads the room inventory from the "hotel.txt" file, populating the “rooms” array with the stored data.

The “calculate bill” function calculates the total bill for a room based on the difference between booking start and end dates, adding an extra day to include the last day, and multiplying the number of nights by the price per night.

The “get bed type from user” function prompts the user to choose a bed type (king, queen, or single) and returns the corresponding enum value. It ensures the user's input is within the valid range (0 to 2) before returning the bed type.

Overall, this code provides a foundation for managing room inventory, calculating bills, and interacting with users to input bed types in a hotel management system.

THE CODE REPRESENTING THE UPWARD EXPLANATION:

// Function to save inventory to file

void saveInventoryToFile(int roomnumber, const room rooms[]) {

ofstream output("hotel.txt");

if (!output) {

cout << "Error opening file for writing!" << endl;

return;

}

for (int i = 0; i < roomnumber; ++i) {

output << setw(10) << rooms[i].bookingstartdate

<< setw(10) << rooms[i].bookingenddate

<< setw(5) << rooms[i].roomnumber

<< setw(5) << rooms[i].numofbeds

<< setw(5) << static\_cast<int>(rooms[i].typeofbed)

<< setw(10) << rooms[i].pricepernight << endl;

}

output.close();

}

// Function to load inventory from file

void loadInventoryFromFile(int& roomnumber, room rooms[]) {

ifstream input("hotel.txt");

if (!input) {

cout << "Error opening file for reading!" << endl;

return;

}

roomnumber = 0;

int bedtypeInt;

while (input >> rooms[roomnumber].bookingstartdate >> rooms[roomnumber].bookingenddate >> rooms[roomnumber].roomnumber >> rooms[roomnumber].numofbeds >> bedtypeInt >> rooms[roomnumber].pricepernight) {

rooms[roomnumber].typeofbed = static\_cast<bedtype>(bedtypeInt);

roomnumber++;

if (roomnumber >= max\_rooms) {

cout << "Maximum room capacity reached." << endl;

break;

}

}

input.close();

}

int calculateBill(const string& startdate, const string& enddate, double pricepernight) {

struct tm starttm = {};

struct tm endtm = {};

istringstream start(startdate);

istringstream end(enddate);

start >> get\_time(&starttm, "%Y-%m-%d");

end >> get\_time(&endtm, "%Y-%m-%d");

time\_t start\_time = mktime(&starttm);

time\_t end\_time = mktime(&endtm);

if (start\_time == -1 || end\_time == -1) {

cerr << "Error parsing dates!" << endl;

return 0;

}

int number\_of\_nights = (difftime(end\_time, start\_time) + 86400) / (60 \* 60 \* 24); // Adding 1 day to include the last day

double totalfees = number\_of\_nights \* pricepernight;

return totalfees;

}

// Function to get bed type from user

bedtype getBedtypeFromUser() {

int choice;

cout << "Choose bed type (0 = king, 1 = queen, 2 = single): ";

cin >> choice;

while (choice < 0 || choice > 2) {

cout << "Invalid choice. Choose bed type (0 = king, 1 = queen, 2 = single): ";

cin >> choice;

}

return static\_cast<bedtype>(choice);

}