CMPD344 – Data Structures and Algorithms

UNIVERSITI TENAGA NASIONAL

GROUP PROJECT: GAYU HOTEL BOOKING SYSTEM

PREPARED BY GROUP: GADIS MELAYU		
No	**Name**	**Student ID**
1	Mas Aisyah Maisarah binti Mas Irhan	DC98869
2	Ainin Sofiya binti Mohd Zarak Zurkanain	DC98871
3	Siti Fatimah Zahra binti Abd Rahman	DC98883
4	Nur Ilyana Batrisyia binti Muhamed Noh	DC98867

SUBMITTED TO:	
Lecturer's Name:	Madam TS. Nur Hanani Azmi

Table of Content

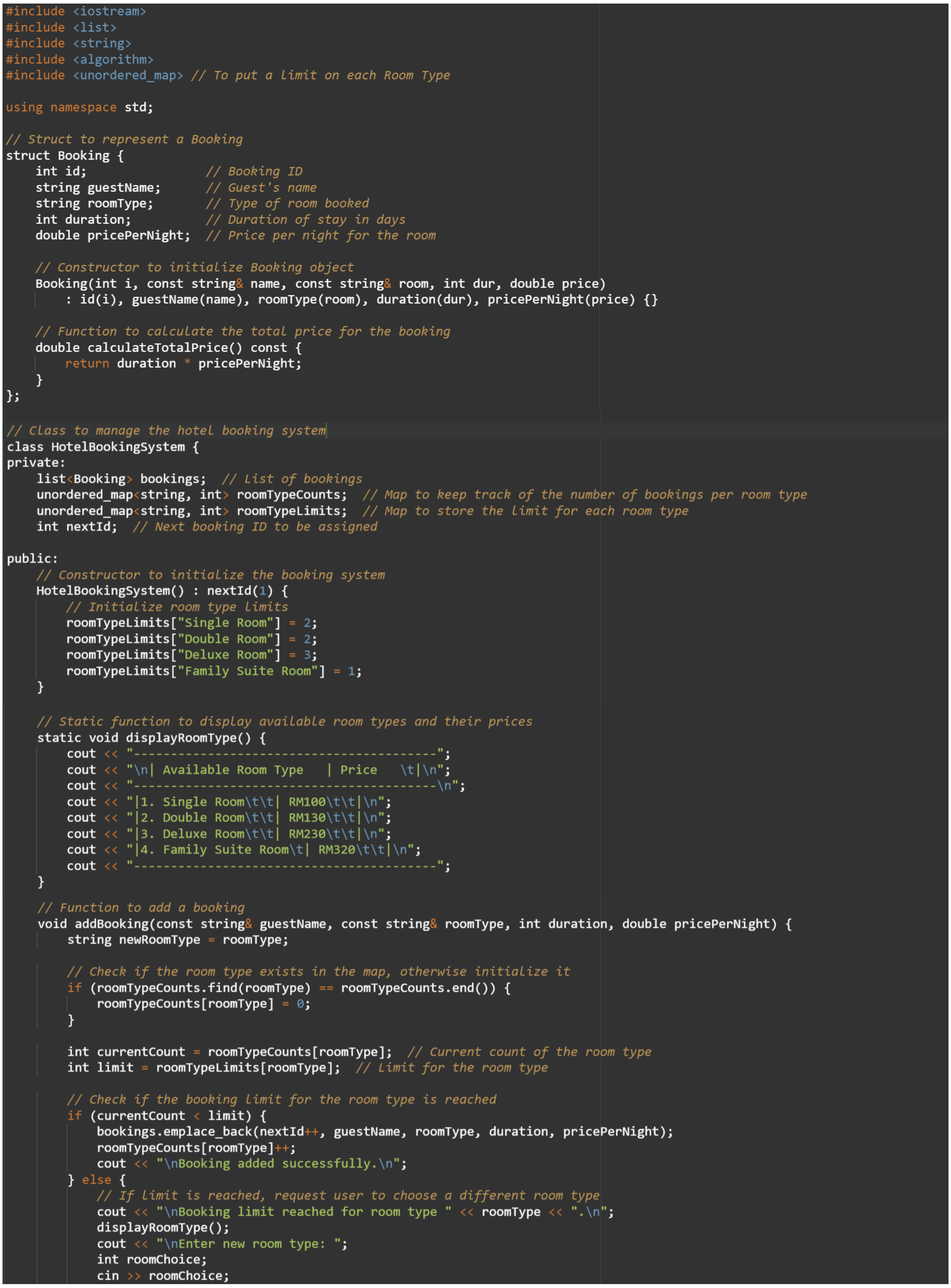
1.0 Introduction

For our group project, we have developed a Hotel Booking System using STL C++ linked list. This system provides key functionalities such as adding new bookings, displaying all existing bookings, searching for specific bookings, updating selected bookings, and canceling bookings.

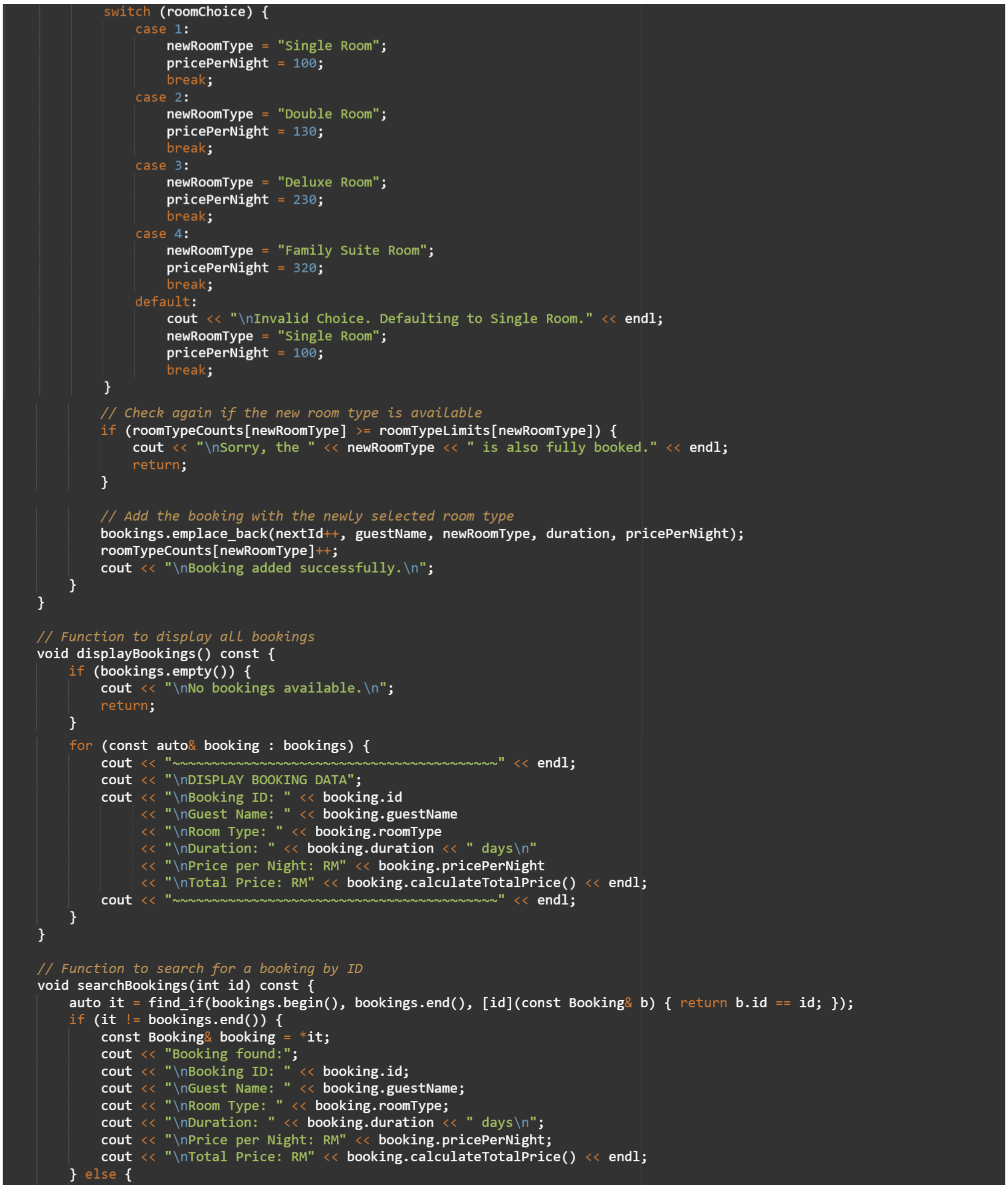
We chose linked lists due to their efficiency in managing dynamic data updates, allowing for easy addition, removal, and searching of bookings. These operations are fundamental for handling hotel reservations effectively. This project underscores our commitment to delivering a reliable solution for hotel reservations, leveraging the performance benefits and object-oriented features of C++ and STL linked lists.

In summary, our Gayu Hotel Booking System aims to provide an efficient solution for managing hotel reservations.

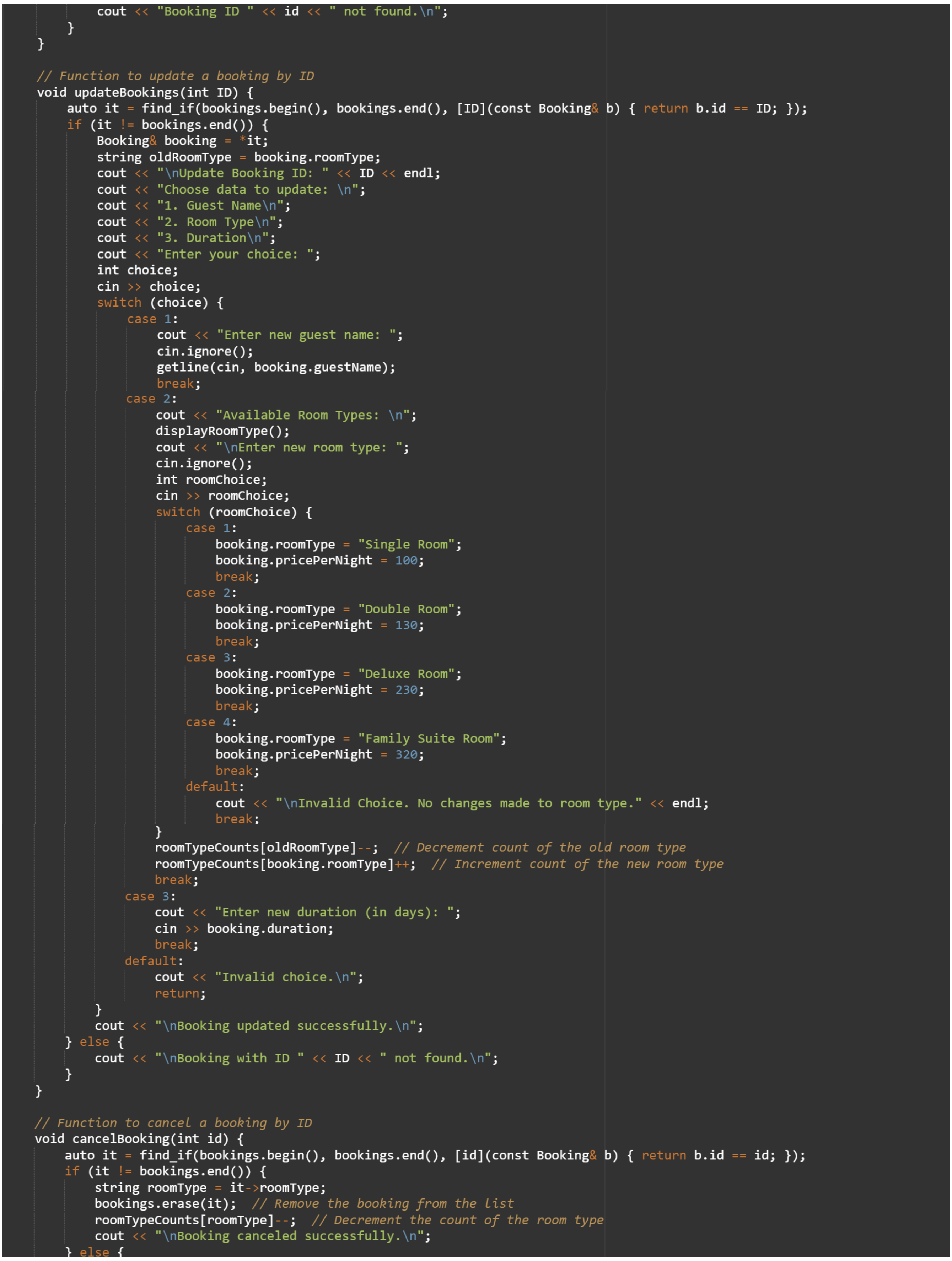
2.0 Source Codes



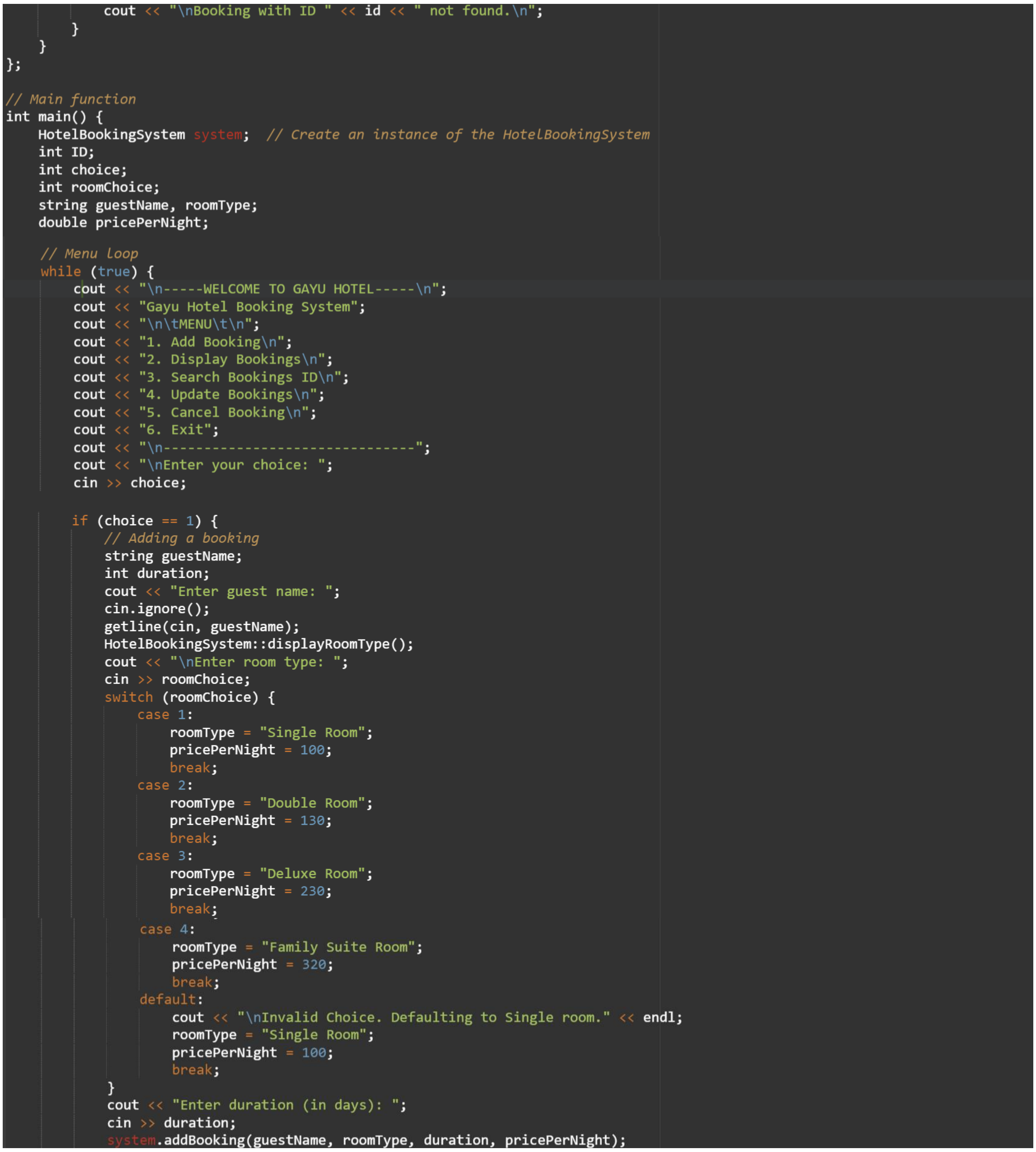
```cpp
#include <iostream>
#include <list>
#include <string>
#include <algorithm>
#include <unordered_map> // To put a limit on each Room Type

using namespace std;

// Struct to represent a Booking
struct Booking {
    int id;                 // Booking ID
    string guestName;       // Guest's name
    string roomType;        // Type of room booked
    int duration;           // Duration of stay in days
    double pricePerNight;   // Price per night for the room

    // Constructor to initialize Booking object
    Booking(int i, const string& name, const string& room, int dur, double price)
        : id(i), guestName(name), roomType(room), duration(dur), pricePerNight(price) {}

    // Function to calculate the total price for the booking
    double calculateTotalPrice() const {
        return duration * pricePerNight;
    }
};

// Class to manage the hotel booking system
class HotelBookingSystem {
private:
    list<Booking> bookings;  // List of bookings
    unordered_map<string, int> roomTypeCounts;  // Map to keep track of the number of bookings per room type
    unordered_map<string, int> roomTypeLimits;  // Map to store the limit for each room type
    int nextId;  // Next booking ID to be assigned

public:
    // Constructor to initialize the booking system
    HotelBookingSystem() : nextId(1) {
        // Initialize room type limits
        roomTypeLimits["Single Room"] = 2;
        roomTypeLimits["Double Room"] = 2;
        roomTypeLimits["Deluxe Room"] = 3;
        roomTypeLimits["Family Suite Room"] = 1;
    }

    // Static function to display available room types and their prices
    static void displayRoomType() {
        cout << "-----------------------------------------";
        cout << "\n| Available Room Type    | Price    \t|\n";
        cout << "-----------------------------------------\n";
        cout << "|1. Single Room\t\t| RM100\t\t|\n";
        cout << "|2. Double Room\t\t| RM130\t\t|\n";
        cout << "|3. Deluxe Room\t\t| RM230\t\t|\n";
        cout << "|4. Family Suite Room\t| RM320\t\t|\n";
        cout << "-----------------------------------------";
    }
    // Function to add a booking
    void addBooking(const string& guestName, const string& roomType, int duration, double pricePerNight) {
        string newRoomType = roomType;

        // Check if the room type exists in the map, otherwise initialize it
        if (roomTypeCounts.find(roomType) == roomTypeCounts.end()) {
            roomTypeCounts[roomType] = 0;
        }

        int currentCount = roomTypeCounts[roomType];  // Current count of the room type
        int limit = roomTypeLimits[roomType];  // Limit for the room type

        // Check if the booking limit for the room type is reached
        if (currentCount < limit) {
            bookings.emplace_back(nextId++, guestName, roomType, duration, pricePerNight);
            roomTypeCounts[roomType]++;
            cout << "\nBooking added successfully.\n";
        } else {
            // If limit is reached, request user to choose a different room type
            cout << "\nBooking limit reached for room type " << roomType << ".\n";
            displayRoomType();
            cout << "\nEnter new room type: ";
            int roomChoice;
            cin >> roomChoice;
```



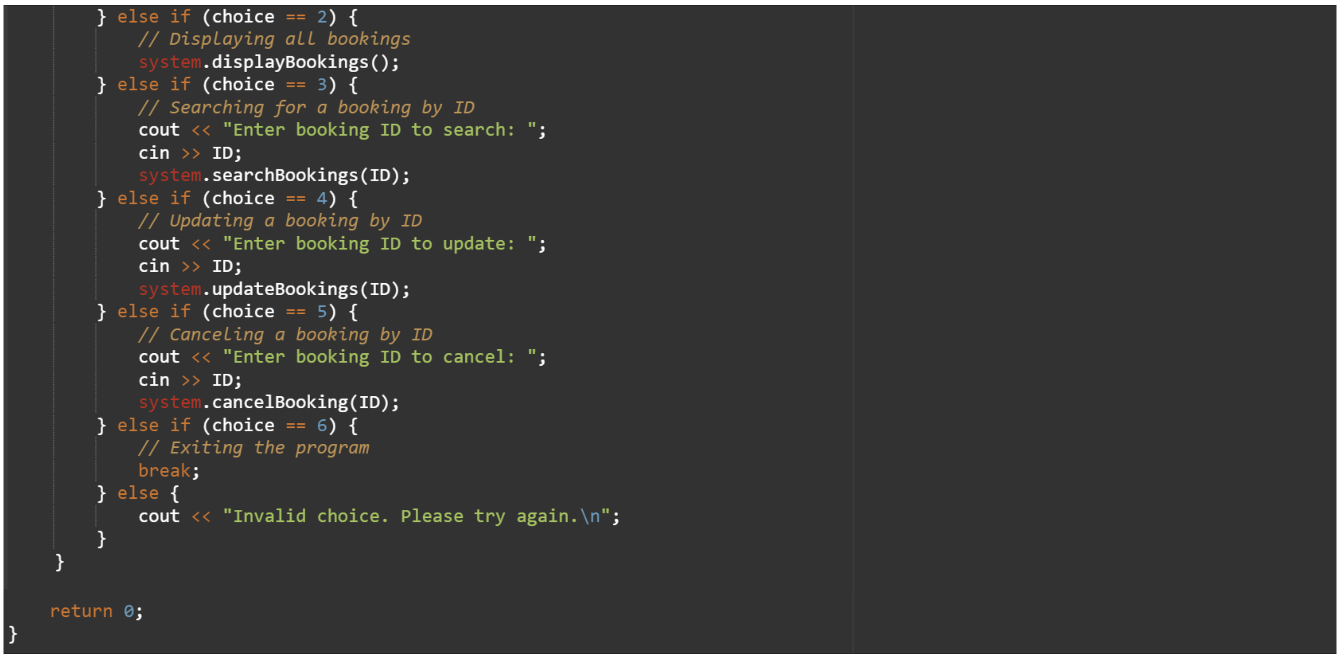
```cpp
        switch (roomChoice) {
            case 1:
                newRoomType = "Single Room";
                pricePerNight = 100;
                break;
            case 2:
                newRoomType = "Double Room";
                pricePerNight = 130;
                break;
            case 3:
                newRoomType = "Deluxe Room";
                pricePerNight = 230;
                break;
            case 4:
                newRoomType = "Family Suite Room";
                pricePerNight = 320;
                break;
            default:
                cout << "\nInvalid Choice. Defaulting to Single Room." << endl;
                newRoomType = "Single Room";
                pricePerNight = 100;
                break;
        }

        // Check again if the new room type is available
        if (roomTypeCounts[newRoomType] >= roomTypeLimits[newRoomType]) {
            cout << "\nSorry, the " << newRoomType << " is also fully booked." << endl;
            return;
        }

        // Add the booking with the newly selected room type
        bookings.emplace_back(nextId++, guestName, newRoomType, duration, pricePerNight);
        roomTypeCounts[newRoomType]++;
        cout << "\nBooking added successfully.\n";
    }
}

// Function to display all bookings
void displayBookings() const {
    if (bookings.empty()) {
        cout << "\nNo bookings available.\n";
        return;
    }
    for (const auto& booking : bookings) {
        cout << "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" << endl;
        cout << "\nDISPLAY BOOKING DATA";
        cout << "\nBooking ID: " << booking.id
             << "\nGuest Name: " << booking.guestName
             << "\nRoom Type: " << booking.roomType
             << "\nDuration: " << booking.duration << " days\n"
             << "\nPrice per Night: RM" << booking.pricePerNight
             << "\nTotal Price: RM" << booking.calculateTotalPrice() << endl;
        cout << "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" << endl;
    }
}

// Function to search for a booking by ID
void searchBookings(int id) const {
    auto it = find_if(bookings.begin(), bookings.end(), [id](const Booking& b) { return b.id == id; });
    if (it != bookings.end()) {
        const Booking& booking = *it;
        cout << "Booking found:";
        cout << "\nBooking ID: " << booking.id;
        cout << "\nGuest Name: " << booking.guestName;
        cout << "\nRoom Type: " << booking.roomType;
        cout << "\nDuration: " << booking.duration << " days\n";
        cout << "\nPrice per Night: RM" << booking.pricePerNight;
        cout << "\nTotal Price: RM" << booking.calculateTotalPrice() << endl;
    } else {
```



```
            cout << "Booking ID " << id << " not found.\n";
        }
    }

    // Function to update a booking by ID
    void updateBookings(int ID) {
        auto it = find_if(bookings.begin(), bookings.end(), [ID](const Booking& b) { return b.id == ID; });
        if (it != bookings.end()) {
            Booking& booking = *it;
            string oldRoomType = booking.roomType;
            cout << "\nUpdate Booking ID: " << ID << endl;
            cout << "Choose data to update: \n";
            cout << "1. Guest Name\n";
            cout << "2. Room Type\n";
            cout << "3. Duration\n";
            cout << "Enter your choice: ";
            int choice;
            cin >> choice;
            switch (choice) {
                case 1:
                    cout << "Enter new guest name: ";
                    cin.ignore();
                    getline(cin, booking.guestName);
                    break;
                case 2:
                    cout << "Available Room Types: \n";
                    displayRoomType();
                    cout << "\nEnter new room type: ";
                    cin.ignore();
                    int roomChoice;
                    cin >> roomChoice;
                    switch (roomChoice) {
                        case 1:
                            booking.roomType = "Single Room";
                            booking.pricePerNight = 100;
                            break;
                        case 2:
                            booking.roomType = "Double Room";
                            booking.pricePerNight = 130;
                            break;
                        case 3:
                            booking.roomType = "Deluxe Room";
                            booking.pricePerNight = 230;
                            break;
                        case 4:
                            booking.roomType = "Family Suite Room";
                            booking.pricePerNight = 320;
                            break;
                        default:
                            cout << "\nInvalid Choice. No changes made to room type." << endl;
                            break;
                    }
                    roomTypeCounts[oldRoomType]--;  // Decrement count of the old room type
                    roomTypeCounts[booking.roomType]++;  // Increment count of the new room type
                    break;
                case 3:
                    cout << "Enter new duration (in days): ";
                    cin >> booking.duration;
                    break;
                default:
                    cout << "Invalid choice.\n";
                    return;
            }
            cout << "\nBooking updated successfully.\n";
        } else {
            cout << "\nBooking with ID " << ID << " not found.\n";
        }
    }

    // Function to cancel a booking by ID
    void cancelBooking(int id) {
        auto it = find_if(bookings.begin(), bookings.end(), [id](const Booking& b) { return b.id == id; });
        if (it != bookings.end()) {
            string roomType = it->roomType;
            bookings.erase(it);  // Remove the booking from the list
            roomTypeCounts[roomType]--;  // Decrement the count of the room type
            cout << "\nBooking canceled successfully.\n";
        } else {
```



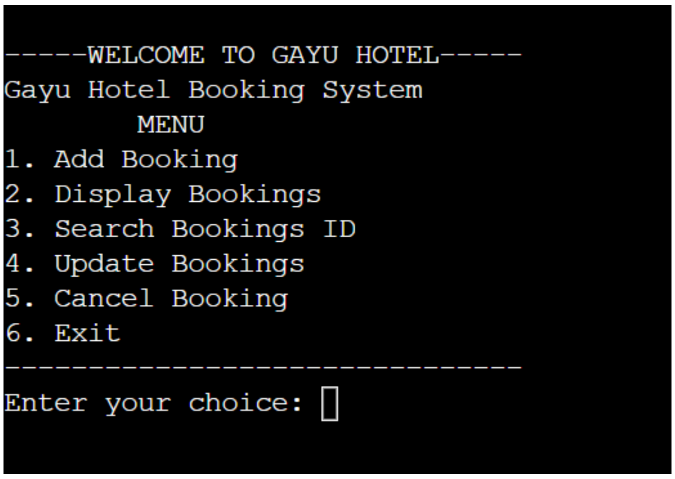
```cpp
            cout << "\nBooking with ID " << id << " not found.\n";
        }
    }
};

// Main function
int main() {
    HotelBookingSystem system;  // Create an instance of the HotelBookingSystem
    int ID;
    int choice;
    int roomChoice;
    string guestName, roomType;
    double pricePerNight;

    // Menu loop
    while (true) {
        cout << "\n-----WELCOME TO GAYU HOTEL-----\n";
        cout << "Gayu Hotel Booking System";
        cout << "\n\tMENU\t\n";
        cout << "1. Add Booking\n";
        cout << "2. Display Bookings\n";
        cout << "3. Search Bookings ID\n";
        cout << "4. Update Bookings\n";
        cout << "5. Cancel Booking\n";
        cout << "6. Exit";
        cout << "\n------------------------------";
        cout << "\nEnter your choice: ";
        cin >> choice;

        if (choice == 1) {
            // Adding a booking
            string guestName;
            int duration;
            cout << "Enter guest name: ";
            cin.ignore();
            getline(cin, guestName);
            HotelBookingSystem::displayRoomType();
            cout << "\nEnter room type: ";
            cin >> roomChoice;
            switch (roomChoice) {
                case 1:
                    roomType = "Single Room";
                    pricePerNight = 100;
                    break;
                case 2:
                    roomType = "Double Room";
                    pricePerNight = 130;
                    break;
                case 3:
                    roomType = "Deluxe Room";
                    pricePerNight = 230;
                    break;
                case 4:
                    roomType = "Family Suite Room";
                    pricePerNight = 320;
                    break;
                default:
                    cout << "\nInvalid Choice. Defaulting to Single room." << endl;
                    roomType = "Single Room";
                    pricePerNight = 100;
                    break;
            }
            cout << "Enter duration (in days): ";
            cin >> duration;
            system.addBooking(guestName, roomType, duration, pricePerNight);
```



```
        } else if (choice == 2) {
            // Displaying all bookings
            system.displayBookings();
        } else if (choice == 3) {
            // Searching for a booking by ID
            cout << "Enter booking ID to search: ";
            cin >> ID;
            system.searchBookings(ID);
        } else if (choice == 4) {
            // Updating a booking by ID
            cout << "Enter booking ID to update: ";
            cin >> ID;
            system.updateBookings(ID);
        } else if (choice == 5) {
            // Canceling a booking by ID
            cout << "Enter booking ID to cancel: ";
            cin >> ID;
            system.cancelBooking(ID);
        } else if (choice == 6) {
            // Exiting the program
            break;
        } else {
            cout << "Invalid choice. Please try again.\n";
        }
    }

    return 0;
}
```

3.0 Output

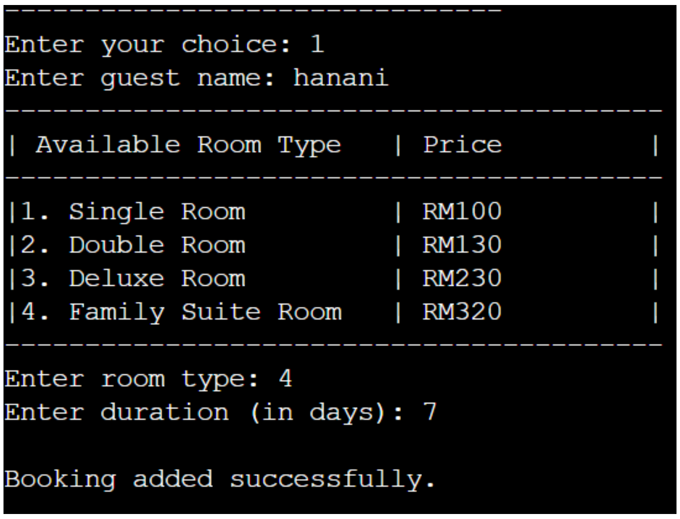
3.1 Menu



```
-----WELCOME TO GAYU HOTEL-----
Gayu Hotel Booking System
        MENU
1. Add Booking
2. Display Bookings
3. Search Bookings ID
4. Update Bookings
5. Cancel Booking
6. Exit
-------------------------------
Enter your choice: 
```

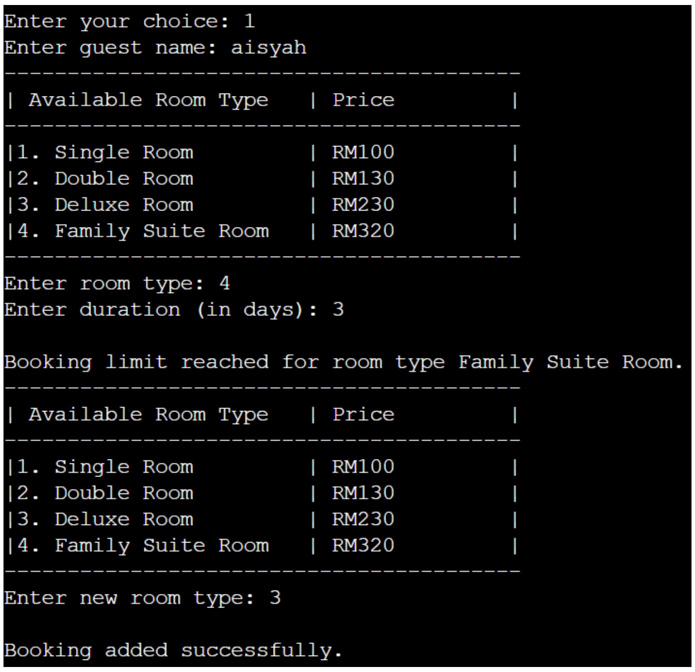
Menu: User chooses which menu to key in input.

3.2 Add Booking



```
------------------------------
Enter your choice: 1
Enter guest name: hanani
----------------------------------------
| Available Room Type    | Price          |
----------------------------------------
|1. Single Room          | RM100          |
|2. Double Room          | RM130          |
|3. Deluxe Room          | RM230          |
|4. Family Suite Room    | RM320          |
----------------------------------------
Enter room type: 4
Enter duration (in days): 7

Booking added successfully.
```

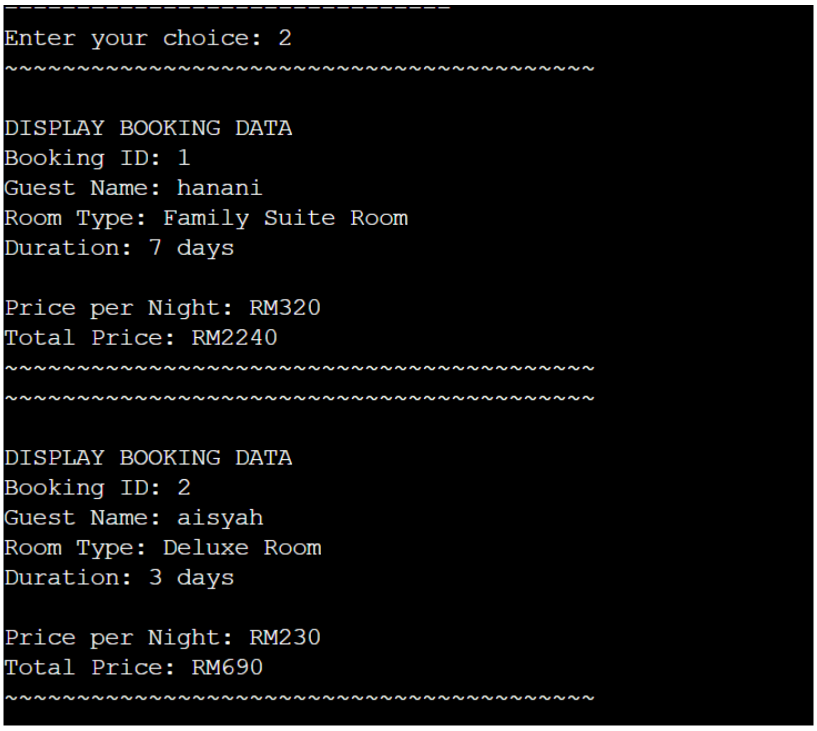
Add Booking: Users choose 1 to add information of the guest, choosing room type and enter duration of stay.



```
Enter your choice: 1
Enter guest name: aisyah
-----------------------------------------
| Available Room Type   | Price          |
-----------------------------------------
|1. Single Room         | RM100          |
|2. Double Room         | RM130          |
|3. Deluxe Room         | RM230          |
|4. Family Suite Room   | RM320          |
-----------------------------------------
Enter room type: 4
Enter duration (in days): 3

Booking limit reached for room type Family Suite Room.
-----------------------------------------
| Available Room Type   | Price          |
-----------------------------------------
|1. Single Room         | RM100          |
|2. Double Room         | RM130          |
|3. Deluxe Room         | RM230          |
|4. Family Suite Room   | RM320          |
-----------------------------------------
Enter new room type: 3

Booking added successfully.
```

A room limit is reached. User needs to choose another option.

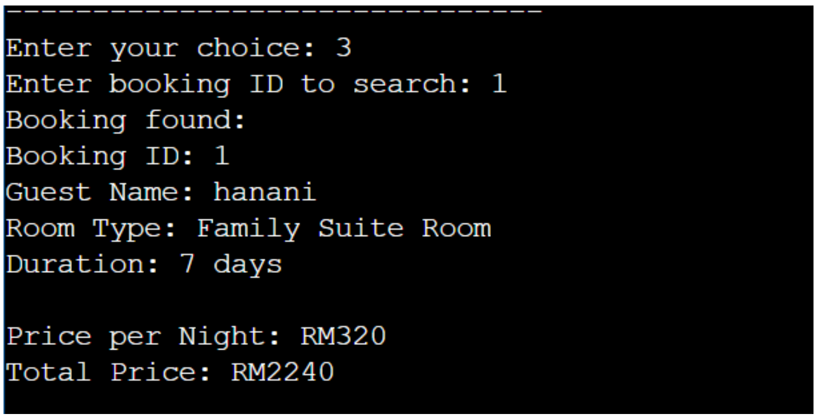
3.3 Display Booking



```
--------------------------------
Enter your choice: 2
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

DISPLAY BOOKING DATA
Booking ID: 1
Guest Name: hanani
Room Type: Family Suite Room
Duration: 7 days

Price per Night: RM320
Total Price: RM2240
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

DISPLAY BOOKING DATA
Booking ID: 2
Guest Name: aisyah
Room Type: Deluxe Room
Duration: 3 days

Price per Night: RM230
Total Price: RM690
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

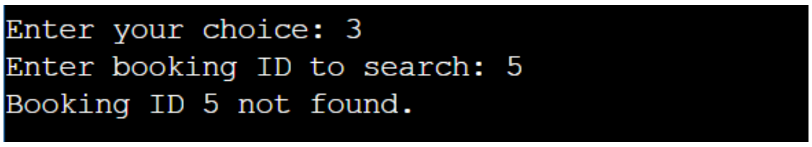
Display Booking: Users choose 2 to display all bookings data that has been key in.

3.4 Search Booking



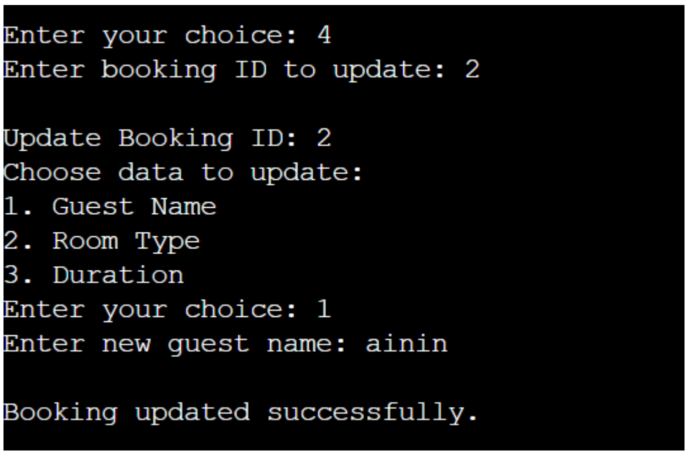
```
--------------------------------
Enter your choice: 3
Enter booking ID to search: 1
Booking found:
Booking ID: 1
Guest Name: hanani
Room Type: Family Suite Room
Duration: 7 days

Price per Night: RM320
Total Price: RM2240
```

Search Booking: Users choose 3 to search for a booking that is already in the data.



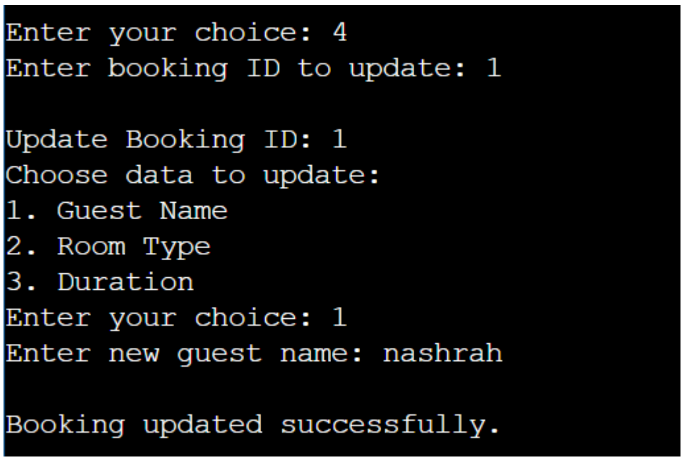
```
Enter your choice: 3
Enter booking ID to search: 5
Booking ID 5 not found.
```

3.5 Update Booking

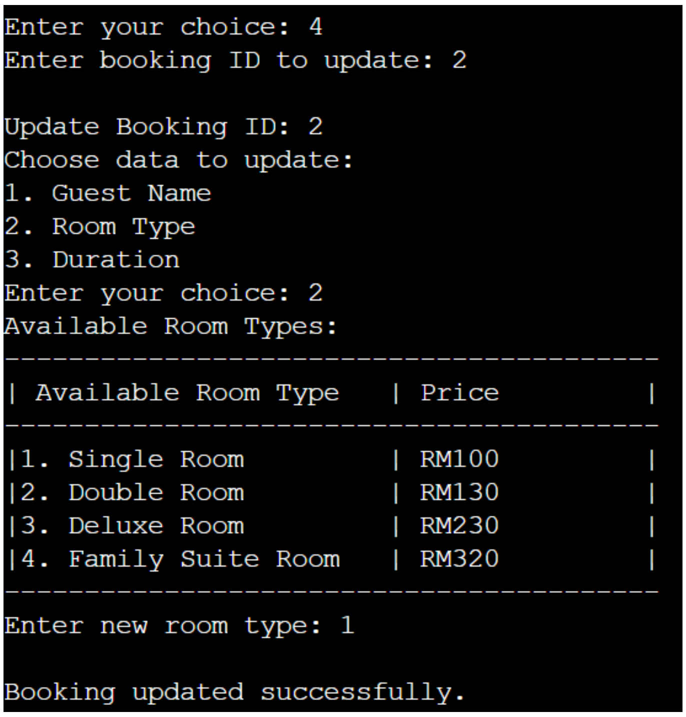


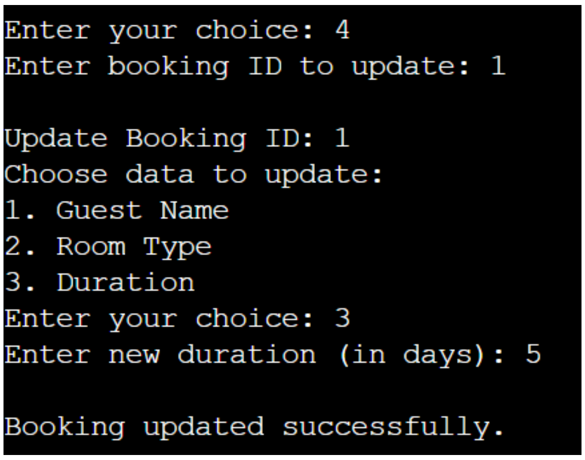
```
Enter your choice: 4
Enter booking ID to update: 2

Update Booking ID: 2
Choose data to update:
1. Guest Name
2. Room Type
3. Duration
Enter your choice: 1
Enter new guest name: ainin

Booking updated successfully.
```

Update Booking: User choose 4 in order to update a booking whether updating the guest name, room type or changing the duration.

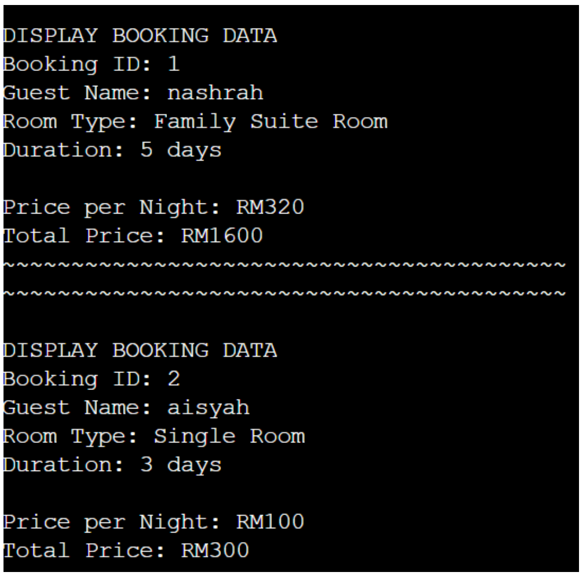


```
Enter your choice: 4
Enter booking ID to update: 1

Update Booking ID: 1
Choose data to update:
1. Guest Name
2. Room Type
3. Duration
Enter your choice: 1
Enter new guest name: nashrah

Booking updated successfully.
```

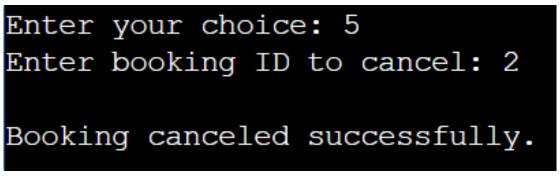

```
Enter your choice: 4
Enter booking ID to update: 2

Update Booking ID: 2
Choose data to update:
1. Guest Name
2. Room Type
3. Duration
Enter your choice: 2
Available Room Types:
-----------------------------------------
| Available Room Type   | Price         |
-----------------------------------------
|1. Single Room         | RM100         |
|2. Double Room         | RM130         |
|3. Deluxe Room         | RM230         |
|4. Family Suite Room   | RM320         |
-----------------------------------------
Enter new room type: 1

Booking updated successfully.
```


```
Enter your choice: 4
Enter booking ID to update: 1

Update Booking ID: 1
Choose data to update:
1. Guest Name
2. Room Type
3. Duration
Enter your choice: 3
Enter new duration (in days): 5

Booking updated successfully.
```

The display after updating data:


```
DISPLAY BOOKING DATA
Booking ID: 1
Guest Name: nashrah
Room Type: Family Suite Room
Duration: 5 days

Price per Night: RM320
Total Price: RM1600
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

DISPLAY BOOKING DATA
Booking ID: 2
Guest Name: aisyah
Room Type: Single Room
Duration: 3 days

Price per Night: RM100
Total Price: RM300
```

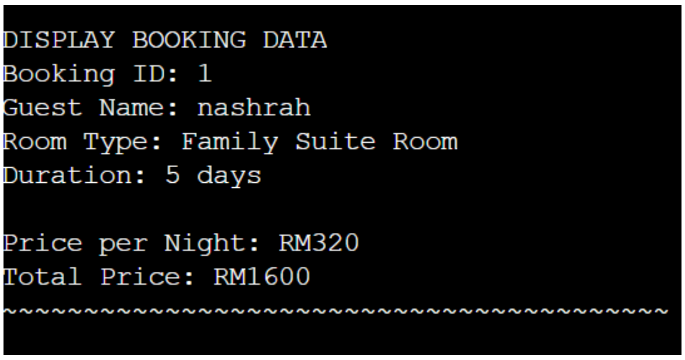
3.6 Cancel Booking



```
Enter your choice: 5
Enter booking ID to cancel: 2

Booking canceled successfully.
```

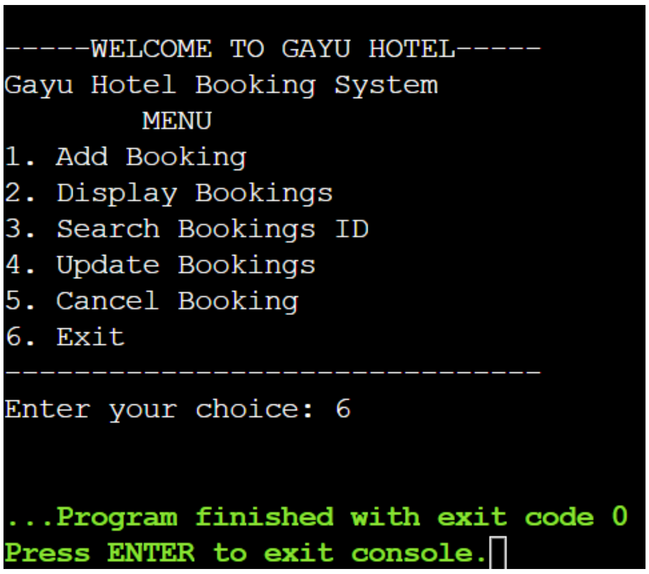
Cancel Booking: User chooses 5 to cancel a booking. The record wouldn't be display anymore after cancelling.

The display after cancellation:



```
DISPLAY BOOKING DATA
Booking ID: 1
Guest Name: nashrah
Room Type: Family Suite Room
Duration: 5 days

Price per Night: RM320
Total Price: RM1600
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

3.7 Exit



```
-----WELCOME TO GAYU HOTEL-----
Gayu Hotel Booking System
        MENU
1. Add Booking
2. Display Bookings
3. Search Bookings ID
4. Update Bookings
5. Cancel Booking
6. Exit
-------------------------------
Enter your choice: 6


...Program finished with exit code 0
Press ENTER to exit console.
```

Exit: Users choose 6 to exit the program.