بسم الله الرحمن الرحيم

Capital University of Science & Technology
Pakistan

Data Structure

Group Project

TOPIC: <u>Hotel Management System</u>

SUBMITTED BY:

Muhammad Huzaifa Jawed	**(BSE223016)**
Muhammad Ahmed Chorahi	**(BSE223121)**
Talat Hussain	**(BSE223009)**
Rauf Ahmed	**(BSE223013)**
Adnan Ali	**(BSE223036)**

SUBMITTED TO: **Sir Taimoor Riaz**

Table of Contents

1. Introduction 3
   1.1 Background 3
   1.2 Objectives 3
2. System Design 3
   2.1 Data Structure Choice 3
   2.2 Functions 3
      2.2.1 Insert() 3
      2.2.2 Update() 5
      2.2.3 Search() 8
      2.2.4 Delete() 9
      2.2.5 DisplayAll() 10
3. Implementation 11
   3.1 Data Structures and Algorithms 11
      3.1.1 Code Snippet 11
4. Results 12
   4.1 Test Cases 15
5. Conclusion 15
6. Future Enhancements 15

Project Proposal: Hotel Management System

1. Introduction

1.1 Background

This project involves the development of a Hotel Management System using linked lists in C++. The system is designed to manage hotel records efficiently, allowing users to insert, update, search, delete, and display records.

1.2 Objectives

The main objectives of the project include:

- Implementing a hotel management system using linked lists.
- Providing functionalities to insert, update, search, delete, and display records.
- Sorting and displaying records to facilitate easy management.

2. System Design

2.1 Data Structure Choice

The chosen data structure for this project is a linked list. This allows for dynamic memory allocation and efficient insertion, deletion, and traversal operations.

2.2 Functions

The system includes the following functions:

2.2.1 Insert()

- Accepts user input for name, id, phone number, room type, email, check-in, and check-out.
- Inserts a new record into the linked list.

```
void Hotel::insert()
{
	system("cls");
```

```
cout << "\n\t______Hotel Management System______\n\n";
Data data;
//input
cout << "Enter the following details\n";
cout << "ID: ";
cin >> data.id;
cin.ignore();
cout << "Name: ";
getline(cin, data.name);
cout << "Room Type: ";
getline(cin, data.roomType);
cout << "Number: ";
getline(cin, data.number);
cout << "Address: ";
getline(cin, data.address);
cout << "Email: ";
getline(cin, data.email);
cout << "Check In: ";
getline(cin, data.checkIn);
cout << "Check Out: ";
getline(cin, data.checkOut);

//create new node
Node* newNode = new Node(data);

//if list is empty
if (head == nullptr)
{
      head = newNode;
}
else
{
      Node* temp = head;
      while (temp->next != nullptr)
```

```
            {
                temp = temp->next;
            }
            temp->next = newNode;
        }
        system("cls");
}
```

2.2.2 Update()

- Allows users to update specific fields (name, id, phone number, room type, email, check-in, or check-out) of a record.
- Takes user input for the record to be updated and the field to be modified.

```
void Hotel::update()
{
        system("cls");
        cout << "\n\t______Hotel Management System______\n\n";
        int id;
        cout << "Enter the ID to update: ";
        cin >> id;
        cout << endl;

        //search the record
        Node* temp = head;
        while (temp != nullptr)
        {
            if (temp->info.id == id)
            {
                cout << "ID: " << temp->info.id << endl;
                cout << "Name: " << temp->info.name << endl;
                cout << "Room Type: " << temp->info.roomType << endl;
                cout << "Number: " << temp->info.number << endl;
                cout << "Address: " << temp->info.address << endl;
                cout << "Email: " << temp->info.email << endl;
                cout << "Check In: " << temp->info.checkIn << endl;
```

```cpp
                cout << "Check Out: " << temp->info.checkOut << endl;
                break;
            }
            temp = temp->next;
        }
        if (temp == nullptr)
        {
            cout << "Record not found\n";
        }
        else
        {
            //update the record
            int choice;
            cout << endl;
            cout << "\t1. Name\n";
            cout << "\t2. Room Type\n";
            cout << "\t3. Number\n";
            cout << "\t4. Address\n";
            cout << "\t5. Email\n";
            cout << "\t6. Check In\n";
            cout << "\t7. Check Out\n";

            cout << "\nEnter the field to update: ";
            cin >> choice;

            switch (choice)
            {
            case 1:
                cin.ignore();
                cout << "Enter the new name: ";
                getline(cin, temp->info.name);
                break;
            case 2:
                cin.ignore();
                cout << "Enter the new room type: ";
                getline(cin, temp->info.roomType);
                break;
            case 3:
                cin.ignore();
                cout << "Enter the new number: ";
                getline(cin, temp->info.number);
                break;
            case 4:
                cin.ignore();
                cout << "Enter the new address: ";
                getline(cin, temp->info.address);
                break;
            case 5:
                cin.ignore();
                cout << "Enter the new email: ";
                getline(cin, temp->info.email);
                break;
            case 6:
                cin.ignore();
                cout << "Enter the new check in: ";
                getline(cin, temp->info.checkIn);
                break;
            case 7:
                cin.ignore();
                cout << "Enter the new check out: ";
                getline(cin, temp->info.checkOut);
                break;
            default:
                cout << "Invalid choice\n";
                break;
            }
            cout << "\n\n\t Record updated successfully\n";
        }
```

```
        system("pause");
        system("cls");
}
```

2.2.3 Search()

- Accepts an id as input and searches for the corresponding record in the linked list.
- Displays the data of the found record.

```
void Hotel::search()
{
        system("cls");
        cout << "\n\t______Hotel Management System______\n\n";
        int id;
        cout << "Enter the ID to search: ";
        cin >> id;
        cout << endl;

        Node* temp = head;
        while (temp != nullptr)
        {
                if (temp->info.id == id)
                {
                        cout << "ID: " << temp->info.id << endl;
                        cout << "Name: " << temp->info.name << endl;
                        cout << "Room Type: " << temp->info.roomType << endl;
                        cout << "Number: " << temp->info.number << endl;
                        cout << "Address: " << temp->info.address << endl;
                        cout << "Email: " << temp->info.email << endl;
                        cout << "Check In: " << temp->info.checkIn << endl;
                        cout << "Check Out: " << temp->info.checkOut << endl;
                        break;
                }
                temp = temp->next;
```

```
    }
    if (temp == nullptr)
    {
        cout << "Record not found\n";
    }
    system("pause");
    system("cls");
}
```

2.2.4 Delete()

- Accepts an id as input and deletes the corresponding record from the linked list.

```
void Hotel::del()
{
    system("cls");
    cout << "\n\t______Hotel Management System______\n\n";
    int id;
    cout << "Enter the ID to delete: ";
    cin >> id;
    cout << endl;

    //search the record
    Node* temp = head;
    Node* prev = nullptr;
    while (temp != nullptr)
    {
        if (temp->info.id == id)
        {
            cout << "ID: " << temp->info.id << endl;
            cout << "Name: " << temp->info.name << endl;
            cout << "Room Type: " << temp->info.roomType << endl;
            cout << "Number: " << temp->info.number << endl;
            cout << "Address: " << temp->info.address << endl;
            cout << "Email: " << temp->info.email << endl;
            cout << "Check In: " << temp->info.checkIn << endl;
```

```
                    cout << "Check Out: " << temp->info.checkOut << endl;
                    break;
                }
                prev = temp;
                temp = temp->next;
        }
        if (temp == nullptr)
        {
                cout << "Record not found\n";
        }
        else
        {
                //delete the record
                if (prev == nullptr)
                {
                        head = temp->next;
                }
                else
                {
                        prev->next = temp->next;
                }
                delete temp;
                cout << "\n\n\t Record deleted successfully\n";
        }
        system("pause");
        system("cls");
}
```

2.2.5 DisplayAll()

- Sorts the records and displays the entire dataset one by one.

```
void Hotel::display()
{
        system("cls");
        cout << "\n\t______Hotel Management System______\n\n";
        Node* temp = head;
        while (temp != nullptr)
        {
                cout << "ID: " << temp->info.id << endl;
                cout << "Name: " << temp->info.name << endl;
                cout << "Room Type: " << temp->info.roomType << endl;
                cout << "Number: " << temp->info.number << endl;
                cout << "Address: " << temp->info.address << endl;
                cout << "Email: " << temp->info.email << endl;
                cout << "Check In: " << temp->info.checkIn << endl;
                cout << "Check Out: " << temp->info.checkOut << endl;
                cout << endl;
                temp = temp->next;
        }
        system("pause");
        system("cls");
}
```

3. Implementation

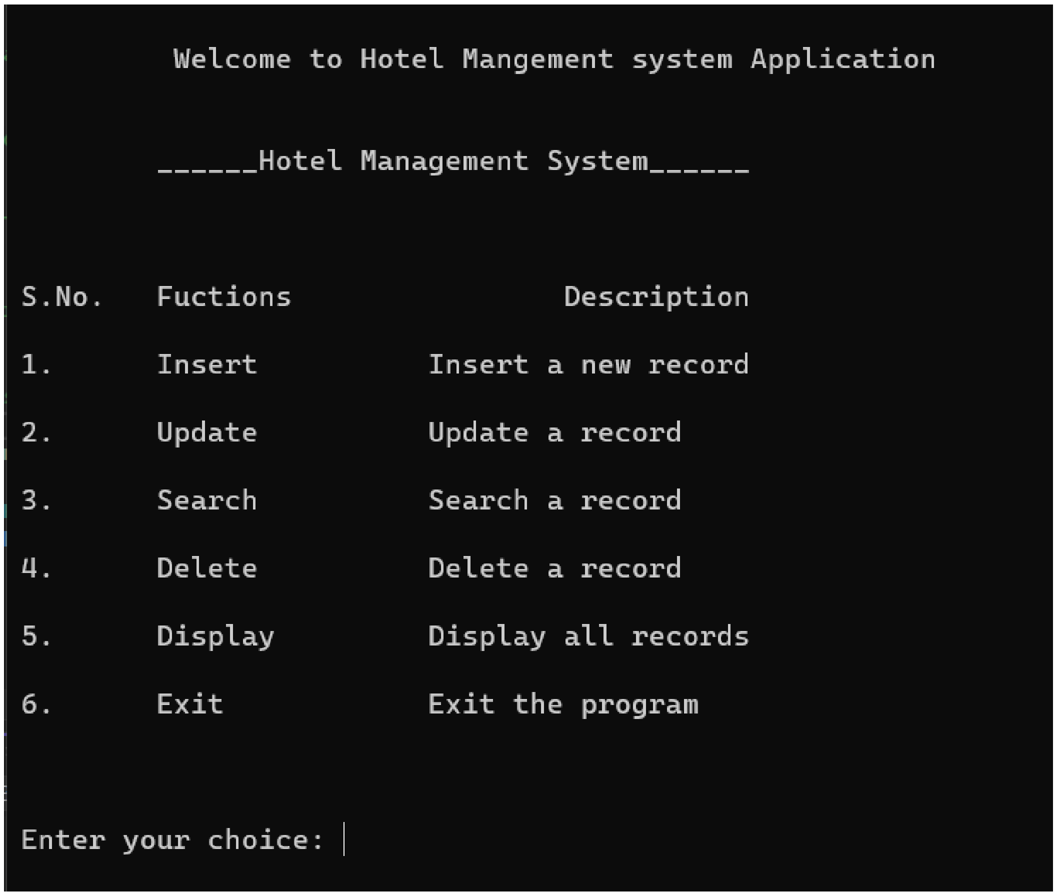
3.1 Data Structures and Algorithms

The linked list is implemented using dynamic memory allocation for each node. Sorting is achieved using a suitable sorting algorithm bubble sort.

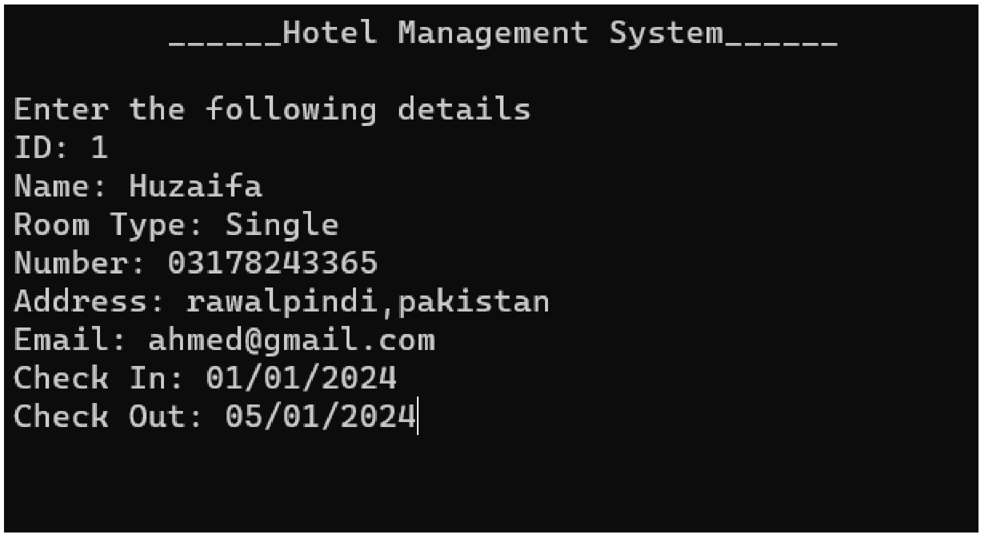
3.1.1 Code Snippet

```
void Hotel::sort()
{
        cout << "\n\t______Hotel Management System______\n\n";
        Node* temp = head;
        while (temp != nullptr)
        {
                Node* temp2 = temp->next;
                while (temp2 != nullptr)
                {
                        if (temp->info.id > temp2->info.id)
                        {
                                Data data = temp->info;
                                temp->info = temp2->info;
                                temp2->info = data;
                        }
                        temp2 = temp2->next;
                }
                temp = temp->next;
        }
}
```

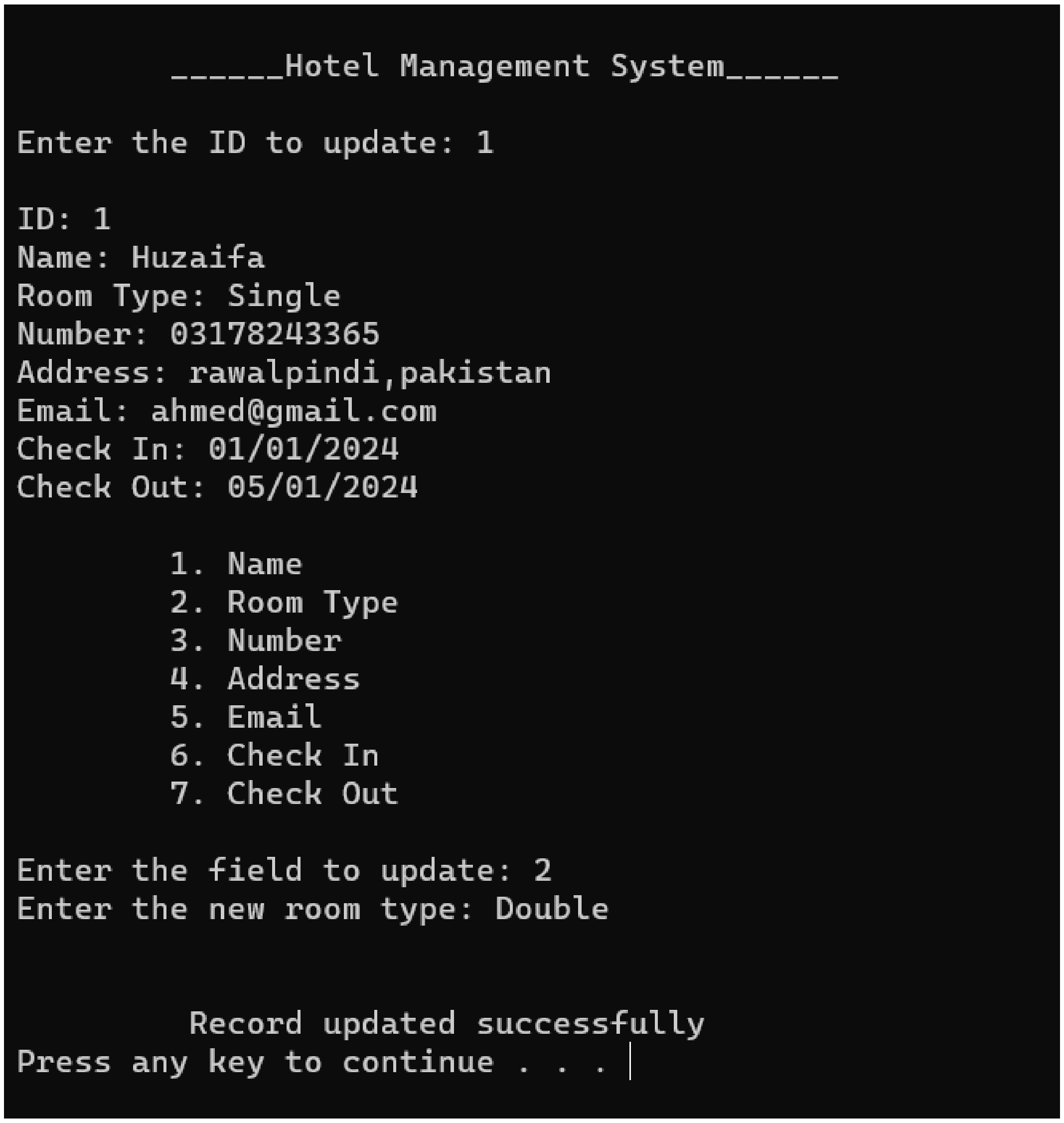
4. Results



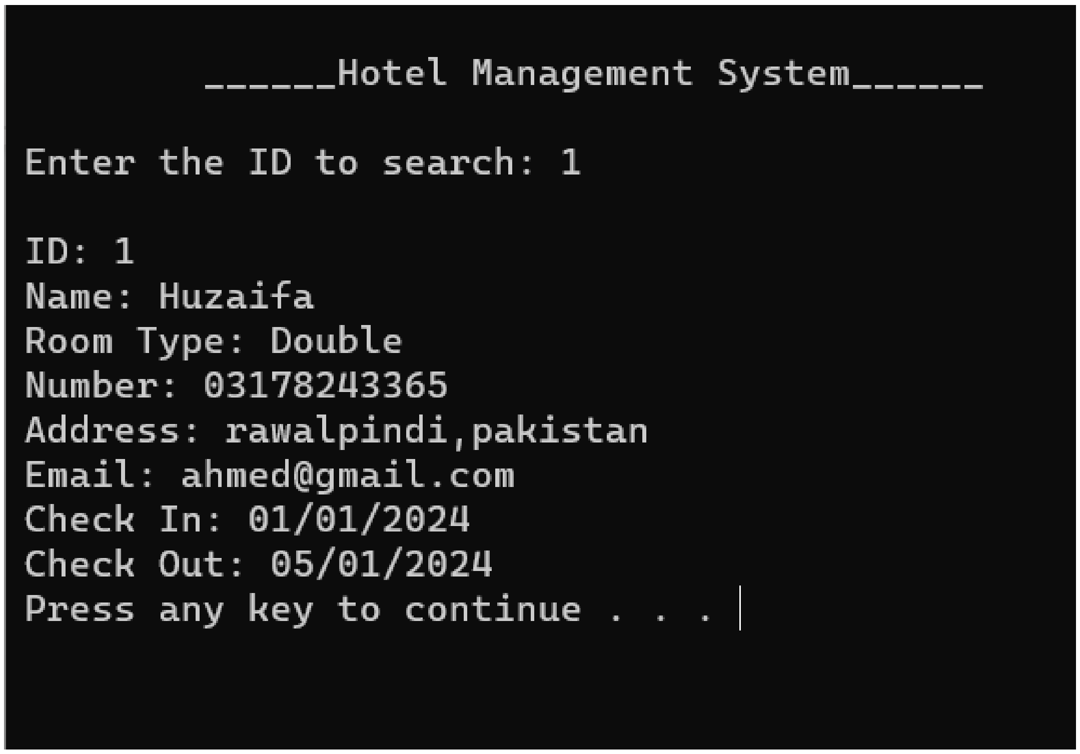
```
          Welcome to Hotel Mangement system Application


         ______Hotel Management System______


S.No.   Fuctions                Description
1.      Insert          Insert a new record
2.      Update          Update a record
3.      Search          Search a record
4.      Delete          Delete a record
5.      Display         Display all records
6.      Exit            Exit the program


Enter your choice: 
```



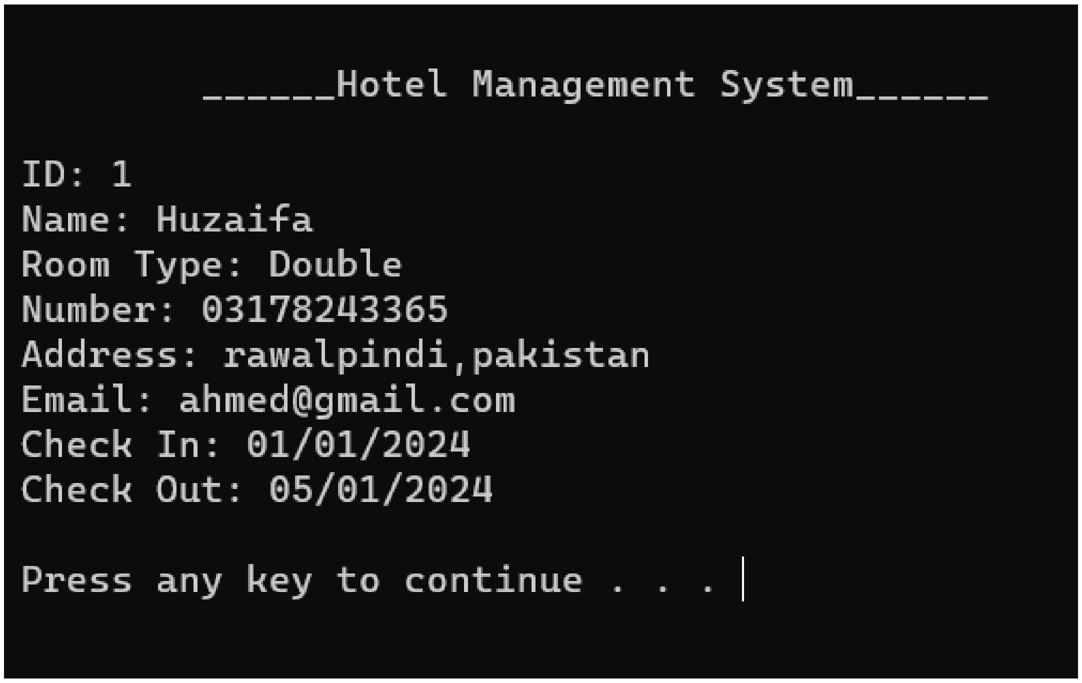
```
        ______Hotel Management System______

Enter the following details
ID: 1
Name: Huzaifa
Room Type: Single
Number: 03178243365
Address: rawalpindi,pakistan
Email: ahmed@gmail.com
Check In: 01/01/2024
Check Out: 05/01/2024
```



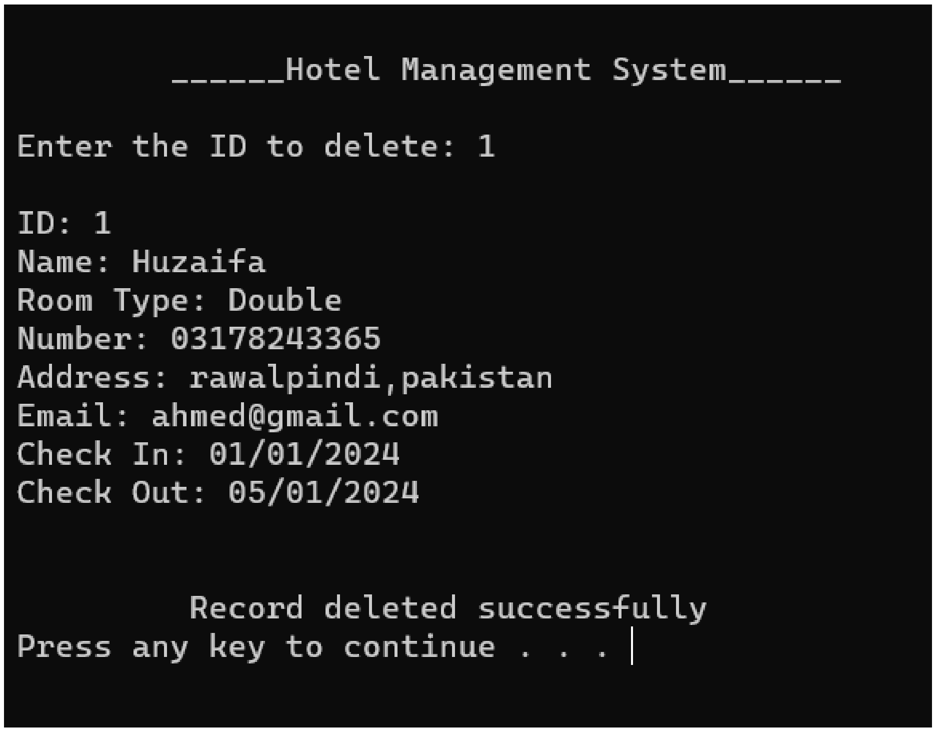
```
        ______Hotel Management System______

Enter the ID to update: 1

ID: 1
Name: Huzaifa
Room Type: Single
Number: 03178243365
Address: rawalpindi,pakistan
Email: ahmed@gmail.com
Check In: 01/01/2024
Check Out: 05/01/2024

        1. Name
        2. Room Type
        3. Number
        4. Address
        5. Email
        6. Check In
        7. Check Out

Enter the field to update: 2
Enter the new room type: Double


         Record updated successfully
Press any key to continue . . .
```



```
        ______Hotel Management System______

Enter the ID to search: 1

ID: 1
Name: Huzaifa
Room Type: Double
Number: 03178243365
Address: rawalpindi,pakistan
Email: ahmed@gmail.com
Check In: 01/01/2024
Check Out: 05/01/2024
Press any key to continue . . .
```



```
        ______Hotel Management System______

ID: 1
Name: Huzaifa
Room Type: Double
Number: 03178243365
Address: rawalpindi,pakistan
Email: ahmed@gmail.com
Check In: 01/01/2024
Check Out: 05/01/2024

Press any key to continue . . .
```



```
        ______Hotel Management System______

Enter the ID to delete: 1

ID: 1
Name: Huzaifa
Room Type: Double
Number: 03178243365
Address: rawalpindi,pakistan
Email: ahmed@gmail.com
Check In: 01/01/2024
Check Out: 05/01/2024


         Record deleted successfully
Press any key to continue . . .
```

4.1 Test Cases

The system was tested with various scenarios to ensure correctness and efficiency. Test cases included inserting records, updating fields, searching for records, deleting records, and displaying sorted data.

5. Conclusion

The Hotel Management System using linked lists in C++ has been successfully implemented, providing a reliable solution for managing hotel records. The linked list structure allows for dynamic and efficient data management.

6. Future Enhancements

- Integration of additional features (e.g., billing, room availability checking).
- Improving the sorting algorithm for larger datasets.
- Enhancing the user interface and experience.