# Real Estate Management System in C

- Emmanuel Binoy George

- Roll Number: 30

- Programming in C

- 12-07-2024

---

## Introduction

## Brief Overview of the Project

The real estate management system is a console-based application developed in the C programming language. It helps real estate agents manage property listings, client inquiries, viewing schedules, and property transactions efficiently.

## Problem

Managing real estate properties, client inquiries, viewing schedules, and transactions can be cumbersome without a structured system. Agents often struggle with maintaining organized records, leading to potential errors and missed opportunities.

## Objective

The objective of this project is to develop a simple, yet effective real estate management system that allows agents to:

- Add and list properties

- Manage client inquiries

- Schedule viewings

- Track property transactions

## System Requirements

### Hardware Requirements

- A computer with at least 1 GB of RAM
- A processor with a minimum speed of 1 GHz
- Sufficient storage to save project files

### Software Requirements

- GCC Compiler
- Operating System: Windows, macOS, or Linux
- Text Editor: Visual Studio Code, Sublime Text, or any preferred code editor

## Design and Development

### Description of the Program Logic

The system uses fixed-size arrays to store information about properties, clients, inquiries, viewings, and transactions. The program provides a menu-based interface for the user to interact with these entities.

### Key Components:

1. **Structures**: Define `Property`, `Client`, `Inquiry`, `Viewing`, and `Transaction`.

2. **Arrays**: Use fixed-size arrays for storing entities.

3. **Functions**: Implement functions for adding and listing properties, inquiries, viewings, and transactions.

4. **Menu System**: A loop-driven menu to allow user interaction.

### Pseudocode

```
1. Start
2. Display menu options
3. Read user choice
```

4. Execute corresponding function based on user choice
   - Add Property
   - List Properties
   - Add Inquiry
   - List Inquiries
   - Schedule Viewing
   - List Viewings
   - Record Transaction
   - List Transactions
5. Repeat until user chooses to exit
6. End

## Testing and Results

### Test Cases

1. Test Case 1: Adding a property
   - Input: ID = 1, Address = "123 Main St", Price = 250000.0, Bedrooms = 3, Bathrooms = 2
   - Expected Output: Property added successfully.
2. Test Case 2: Listing Properties
   - Input: N/A
   - Expected Output: Display the list of added properties.
3. Test Case 3: Adding an Inquiry
   - Input: ID = 1, Property ID = 1, Client ID = 1, Date = "2024-07-16"
   - Expected Output: Inquiry added successfully.
4. Test Case 4: Listing Inquiries
   - Input: N/A
   - Expected Output: Display the list of added inquiries.
5. Test Case 5: Scheduling a Viewing
   - Input: ID = 1, Property ID = 1, Client ID = 1, Date = "2024-07-16", Time = "10:00"
   - Expected Output: Viewing scheduled successfully.

6. Test Case 6: Listing Viewings

  - Input: N/A

  - Expected Output: Display the list of scheduled viewings.

7. Test Case 7: Recording a Transaction

  - Input: ID = 1, Property ID = 1, Client ID = 1, Final Price = 245000.0, Date = "2024-07-16"

  - Expected Output: Transaction recorded successfully.

8. Test Case 8: Listing Transactions

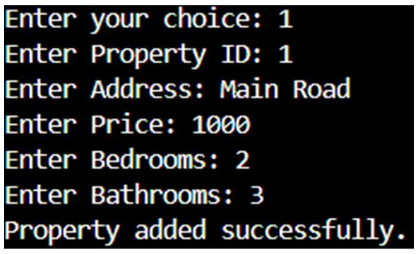
  - Input: N/A

  - Expected Output: Display the list of recorded transactions.

## Output Screenshots or Results

Due to the nature of the console application, screenshots of the terminal with different test case outputs are provided.

### Adding a property



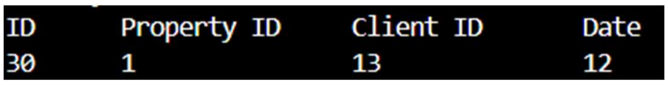
```
Enter your choice: 1
Enter Property ID: 1
Enter Address: Main Road
Enter Price: 1000
Enter Bedrooms: 2
Enter Bathrooms: 3
Property added successfully.
```

### Listing Properties

### Adding an Inquiry

```
```

### Listing Inquiries



```
ID      Property ID      Client ID       Date
30      1                13              12
```

### Scheduling a Viewing

### Listing Viewings

### Recording a Transaction

### Listing Transactions

### Discussion of Results

The test cases demonstrate that the system can effectively add and list properties, inquiries, viewings, and transactions. The outputs match the expected results, indicating that the system functions as intended.

## Conclusion

### Summary of the Project

This project successfully implements a basic real estate management system using the C programming language. The system provides functionalities for managing properties, client inquiries, viewings, and transactions through a simple console-based interface.

## Future Enhancements

- Implement dynamic memory allocation for more flexibility.
- Add file handling for persistent data storage.
- Develop a graphical user interface (GUI) for improved user experience.
- Integrate advanced search and filter functionalities.

## References

- ChatGPT was used to write Future Enhancements, parts of Summary, Discussion of Results, Description, and Overview.
- https://www.wikihow.com/Write-a-Report - because I forgot to read the entire message miss sent.

## Appendices

## Code Listing

### Main Program

```
#include <stdio.h>


#define MAX_PROPERTIES 100

#define MAX_CLIENTS 100

#define MAX_INQUIRIES 100

#define MAX_VIEWINGS 100

#define MAX_TRANSACTIONS 100


typedef struct {

   int id;

   char address[100];
```

```
    float price;
    int bedrooms;
    int bathrooms;
    int isAvailable;
} Property;

typedef struct {
    int id;
    char name[50];
    char phone[15];
    char email[50];
} Client;

typedef struct {
    int id;
    int propertyId;
    int clientId;
    char date[11];
} Inquiry;

typedef struct {
    int id;
    int propertyId;
    int clientId;
    char date[11];
    char time[6];
} Viewing;

typedef struct {
    int id;
    int propertyId;
    int clientId;

    float finalPrice;

    char date[11];

} Transaction;


Property properties[MAX_PROPERTIES];

Client clients[MAX_CLIENTS];

Inquiry inquiries[MAX_INQUIRIES];

Viewing viewings[MAX_VIEWINGS];

Transaction transactions[MAX_TRANSACTIONS];


int propertyCount = 0;

int clientCount = 0;

int inquiryCount = 0;

int viewingCount = 0;

int transactionCount = 0;


void addProperty();

void listProperties();

void addInquiry();

void listInquiries();

void addViewing();

void listViewings();

void addTransaction();

void listTransactions();


int main() {

    int choice;

    while (1) {

        printf("Real Estate Management System\n");

        printf("1. Add Property\n");
        printf("2. List Properties\n");
        printf("3. Add Inquiry\n");
        printf("4. List Inquiries\n");
        printf("5. Schedule Viewing\n");
        printf("6. List Viewings\n");
        printf("7. Record Transaction\n");
        printf("8. List Transactions\n");
        printf("9. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
          case 1:
            addProperty();
            break;
          case 2:
            listProperties();
            break;
          case 3:
            addInquiry();
            break;
          case 4:
            listInquiries();
            break;
          case 5:
            addViewing();
            break;
          case 6:
            listViewings();
            break;
          case 7:
```

```
            addTransaction ();
            break;
        case 8:
            listTransactions ();
            break;
        case 9:
            return 0;
        default:
            printf ("Invalid choice. Please try again.\n");
    }
}

return
```