### VISVESVARAYA TECHNOLOGICAL UNIVERSITY

**Jnana Sangama, Belagavi - 590018**

## TECHNICAL TRAINING

**Mini project On**

## “ REAL ESTATE ADVISOR”

**By**

**KARTHIK U SHETTIGAR 4MT21CS064**

**DEPARTMENT OF COMPUTER SCIENCE& ENGINEERING**

**(*Accredited by NBA*)**

## MANGALORE INSTITUTE OF TECHNOLOGY & ENGINEERING

*Accredited by NAAC with A+ Grade, An ISO 9001: 2015 Certified Institution* (*A Unit of Rajalaxmi Education Trust®, Mangalore - 575001*) Affiliated to VTU, Belagavi, Approved by AICTE, New Delhi,**Badaga Mijar, Moodabidri-574225, Karnataka**

**2022-23**

**1. Abstract:**

The Real Estate Advisor project is a command-line application designed to manage and provide information about properties. It allows users to perform various operations, including displaying property details, inserting new properties, deleting existing properties, searching for properties in specific locations, and updating property information. This report provides an in-depth overview of the project's development, design, features, and functionality.

**2. Introduction:**

**2.1 Background:**

The Real Estate Advisor project was developed to address the need for an efficient and user-friendly system for managing real estate properties. Real estate professionals and property seekers often require a tool to organize and access property information conveniently.

**2.2 Objectives:**

The primary objectives of this project are as follows:

* To create a user-friendly interface for managing property information.
* To allow users to add new properties to the system.
* To enable users to search for properties based on their location.
* To provide functionality for updating property details.
* To facilitate the deletion of properties from the system.

**3. Technologies Used:**

The technologies and tools used in the development of the Real Estate Advisor project include:

* C programming language
* File I/O for data storage

**4. System Architecture:**

The Real Estate Advisor project follows a simple architecture consisting of the following components:

**4.1 Front-End:**

The front-end of the application is a command-line interface (CLI) that interacts with users. It provides a menu-driven interface for users to choose various operations.

**4.2 Back-End:**

The back-end handles the core logic of the application, including property management, searching, updating, and deleting. It uses C programming to implement these functionalities.

**4.3 Database:**

The application stores property information in a text file named "techpro.txt." Each property record is stored as a structure within the file.

**5. Project Modules:**

The Real Estate Advisor project is divided into several modules to manage different aspects of property information:

**5.1 Module 1: Property Management:**

This module handles property-related actions such as displaying, buying, selling, inserting, searching, deleting, and updating property information.

**5.2 Module 2: User Management:**

This module manages user authentication, distinguishing between admin and user modes.

**5.3 Module 2: Property Search:**

Users can search for properties based on their location using this module.

**5.4 Module 3: Property Update:**

This module allows users to modify existing property details.

**5.5 Module 4: Property Deletion:**

Properties can be deleted from the system using this module.

**6. Design and Implementation:**

**6.1 Front-End Design:**

The front-end design involves creating a menu-driven interface that presents users with options to perform various operations.

**6.2 Back-End Design:**

The back-end design includes implementing the logic for property management, searching, updating, and deletion.

**6.3 Database Design:**

The database design consists of defining the structure of property records within the "techpro.txt" file.

**7. Features and Functionality:**

**7.1 Property Listing Display:**

Users can view available property listings.

**7.2 Property Purchase:**

Users can buy properties, provided they have sufficient funds.

**7.3 Property Selling:**

Users can sell properties, adding funds to their purse.

**7.4 Property Insertion:**

Admins can insert new property listings.

**7.5 Property Search:**

Users can search for properties by location.

**7.6 Property Deletion:**

Admins can delete property listings.

**7.7 Property Update:**

Admins can update property details.

**8. Testing:**

The Real Estate Advisor project underwent the following testing phases:

**8.1 Unit Testing:**

Individual functions and modules were tested to ensure they perform as expected.

**8.2 Integration Testing:**

The integration of various modules and their interactions were tested to verify the entire system's functionality.

**8.3 User Acceptance Testing:**

The application was tested by end-users to ensure that it meets their requirements and functions as intended.

**9. Challenges Faced:**

During the development of the Real Estate Advisor project, several challenges were encountered, including:

* File I/O error handling
* Input validation and sanitization
* User interface design in a command-line environment

These challenges were addressed through careful coding and testing.

**10. Future Enhancements:**

To further improve the Real Estate Advisor project, future enhancements can be considered, such as:

* Implementing a graphical user interface (GUI) for a more user-friendly experience.
* Adding advanced search and filter options.
* Integrating with external data sources for property listings.

**11. Conclusion:**

The Real Estate Advisor project provides a basic yet functional solution for property management. It offers essential features for adding, searching, updating, and deleting property information for admin and additional buy and sell property option for the user. While it serves its core purpose, there is room for expansion and enhancement to make it more versatile and user-friendly.

**12)References:**

<https://youtu.be/JNO3NIwNL30?si=UWF833tw6xRGgxvj>

<https://www.kashipara.com/project/idea/c-c-/real-estate-advisor_1454.html>

**13)Appendices:**

**13.2)Code Snippets:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

double purse=100000;

struct property {

    int id;

    char types[50];

    char location[100];

    double price;

};

struct property p;

void display() {

    FILE \*fp;

    fp = fopen("techpro.txt", "rb");

    if (fp == NULL) {

        printf("Error: File not found or cannot be opened\n");

        return;

    }

    printf("\n---Available property details are as follows:---\n");

    printf("ID\tTYPE\t\tLOCATION\tPRICE\n");

    while (fread(&p, sizeof(p), 1, fp) == 1) {

        printf("%-6d\t%-10s\t%-8s\t%-6lf\n", p.id, p.types, p.location, p.price);

    }

    fclose(fp);

}

void display1() {

    FILE \*fp;

    fp = fopen("user.txt", "rb");

    if (fp == NULL) {

        printf("Error: File not found or cannot be opened\n");

        return;

    }

    printf("\n---Owning property details are as follows:---\n");

    printf("ID\tTYPE\t\tLOCATION\tPRICE\n");

    while (fread(&p, sizeof(p), 1, fp) == 1) {

        printf("%-6d\t%-10s\t%-8s\t%-6lf\n", p.id, p.types, p.location, p.price);

    }

    fclose(fp);

}

void buyProperty() {

    FILE \*fp, \*ft, \*userFile;

    int pid;

    unsigned flag = 0;

     int purchased = 0;

    fp = fopen("techpro.txt", "rb");

    if (fp == NULL) {

        printf("Error: File not found or cannot be opened\n");

        return;

    }

    display();

    printf("Enter property id you want to buy: ");

    scanf("%d", &pid);

    ft = fopen("temp.txt", "wb");

    userFile = fopen("user.txt", "ab+");

    while (fread(&p, sizeof(p), 1, fp) == 1) {

        if (pid == p.id) {

            if (purse >= p.price) {

                flag = 1;

                purse -= p.price;

                fwrite(&p, sizeof(p), 1, userFile);

                purchased=1;

            } else {

                flag=2;

                printf("Not enough money in your purse to buy this property.\n");

            }

        } else {

            fwrite(&p, sizeof(p), 1, ft);

        }

    }

    if (flag == 0) {

        printf("Error: No such record found\n");

    } else if (flag==1){

        printf("Successfully bought the property\n");

    }

        fclose(fp);

    fclose(ft);

    fclose(userFile);

        if (purchased==1) {

        remove("techpro.txt");

        rename("temp.txt", "techpro.txt");

    }

}

void sellProperty() {

    FILE \*userFile, \*fp, \*tempFile;

    int pid;

    unsigned flag = 0;

    userFile = fopen("user.txt", "rb");

    if (userFile == NULL) {

        printf("Error: User file not found or cannot be opened\n");

        return;

    }

    display1();

    printf("Enter property id you want to sell: ");

    scanf("%d", &pid);

    fp = fopen("techpro.txt", "ab+");

    tempFile = fopen("temp.txt", "wb");

    while (fread(&p, sizeof(p), 1, userFile) == 1) {

        if (pid == p.id) {

            flag = 1;

            purse += p.price;

            fwrite(&p, sizeof(p), 1, fp);

        } else {

            fwrite(&p, sizeof(p), 1, tempFile);

        }

    }

    if (flag == 0) {

        printf("Error: Property not found in your list\n");

    } else {

        printf("Successfully sold the property\n");

    }

    fclose(userFile);

    fclose(tempFile);

    fclose(fp);

    remove("user.txt");

    rename("temp.txt", "user.txt");

}

void insert() {

    FILE \*fp;

    fp = fopen("techpro.txt", "ab+");

    if (fp == NULL) {

        printf("Error: File not found or cannot be opened\n");

        return;

    }

    printf("Enter new property details:\n");

    printf("Enter property ID: ");

    scanf("%d", &p.id);

    fflush(stdin);

    printf("Enter property type: ");

    scanf("%s", p.types);

    printf("Enter property location: ");

    scanf("%s", p.location);

    printf("Enter property price: ");

    scanf("%lf", &p.price);

    fwrite(&p, sizeof(p), 1, fp);

    printf("Property inserted successfully\n");

    fclose(fp);

}

void search() {

    FILE \*fp;

    char loc[100];

    int flag = 0;

    fp = fopen("techpro.txt", "rb");

    if (fp == NULL) {

        printf("Error: File not found or cannot be opened\n");

        return;

    }

    printf("Enter location where you want to search for properties: ");

    scanf("%s", loc);

    printf("\n---Available property details in %s are as follows:---\n", loc);

    printf("ID\tTYPE\t\tLOCATION\tPRICE\n");

    while (fread(&p, sizeof(p), 1, fp) > 0) {

        if (strcasecmp(p.location, loc) == 0) {

            flag = 1;

            printf("%-6d\t%-10s\t%-8s\t%-6lf\n", p.id, p.types, p.location, p.price);

        }

    }

    if (flag == 0) {

        printf("\nNo properties available in %s\n", loc);

    }

    fclose(fp);

}

void deleteProperty() {

    FILE \*fp, \*ft;

    int pid;

    unsigned flag = 0;

    fp = fopen("techpro.txt", "rb");

    if (fp == NULL) {

        printf("Error: File not found or cannot be opened\n");

        return;

    }

    display();

    printf("Enter property id you want to delete: ");

    scanf("%d", &pid);

    ft = fopen("temp.txt", "wb");

    while (fread(&p, sizeof(p), 1, fp) == 1) {

        if (pid == p.id) {

            flag=1;

        } else {

            fwrite(&p, sizeof(p), 1, ft);

        }

    }

    if (flag == 0) {

        printf("Error: No such record found\n");

    }

    else{

        printf("Successfully deleted the property\n");

    }

    fclose(fp);

    fclose(ft);

    remove("techpro.txt");

    rename("temp.txt", "techpro.txt");

}

void update() {

    FILE \*fp;

    int pid;

    int flag = 0;

    fp = fopen("techpro.txt", "r+b");

    if (fp == NULL) {

        printf("Error: File not found or cannot be opened\n");

        return;

    }

    printf("Enter property id which you want to update: ");

    scanf("%d", &pid);

    while (fread(&p, sizeof(p), 1, fp) > 0 && flag == 0) {

        if (p.id == pid) {

            flag = 1;

            printf("Enter updated property details:\n");

            printf("Enter updated property ID: ");

            scanf("%d", &p.id);

            fflush(stdin);

            printf("Enter updated property type: ");

            scanf("%s", p.types);

            printf("Enter updated property location: ");

            scanf("%s", p.location);

            printf("Enter updated property price: ");

            scanf("%lf", &p.price);

            fseek(fp, -sizeof(p), SEEK\_CUR);

            fwrite(&p, sizeof(p), 1, fp);

            printf("Property updated successfully\n");

        }

    }

    if (flag == 0) {

        printf("Error: Property not found\n");

    }

    fclose(fp);

}

int main() {

    int ch,log,pwd;

    printf("Enter 0 for admin login 1 for user: \n");

    scanf("%d",&log);

    if(log==0){

        printf("Enter admin password:  \n");

        fflush(stdin);

        scanf("%d",&pwd);

        if(pwd!=1234){

            printf("Wrong password!!!!!\n");

            main();

        }

        else{

            printf("login successfull\_-\_-\_\n");

            while (ch != 7) {

        printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

        printf("\t\tReal Estate Advisor\n");

        printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

        printf("----------------------ADMIN MENU------------------------\n");

        printf("\t\t1. Display property\n");

        printf("\t\t2. Insert property\n");

        printf("\t\t3. Delete property\n");

        printf("\t\t4. Search property\n");

        printf("\t\t5. Modify property\n");

        printf("\t\t6. Switch to user page\n");

        printf("\t\t7. Exit\n");

        printf("Enter your choice: ");

        scanf("%d", &ch);

        switch (ch) {

            case 1:

                display();

                break;

            case 2:

                insert();

                break;

            case 3:

                deleteProperty();

                break;

            case 4:

                search();

                break;

            case 5:

                update();

                break;

            case 6: main();

                break;

            case 7:

                printf("Quitting!!!!!\n");

                exit(0);

                break;

            default:

                printf("Invalid choice\n");

        }

    }

        }

    }

    else if(log==1){

      while (ch != 7) {

        printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

        printf("\t\tReal Estate Advisor\n");

        printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

        printf("-----------------------USER MENU------------------------\n");

        printf("\t\t1. Display all properties\n");

        printf("\t\t2. Search property\n");

        printf("\t\t3. Buy property\n");

        printf("\t\t4. Sell Porperty\n");

        printf("\t\t5. Display owning properties\n");

        printf("\t\t6. Switch to admin page\n");

        printf("\t\t7. Exit\n");

        printf("Enter your choice: ");

        scanf("%d", &ch);

        switch (ch) {

            case 1:

                display();

                break;

            case 2:

                search();

                break;

            case 3:buyProperty();

                break;

            case 4:sellProperty();

                 break;

            case 5:display1();

                break;

            case 6:main();

                break;

            case 7:

                printf("Quitting!!!!!\n");

                exit(0);

                break;

            default:

                printf("Invalid choice\n");

        }

    }

    }

    else{

        printf("Invalid input!!!!!\n");

        main();

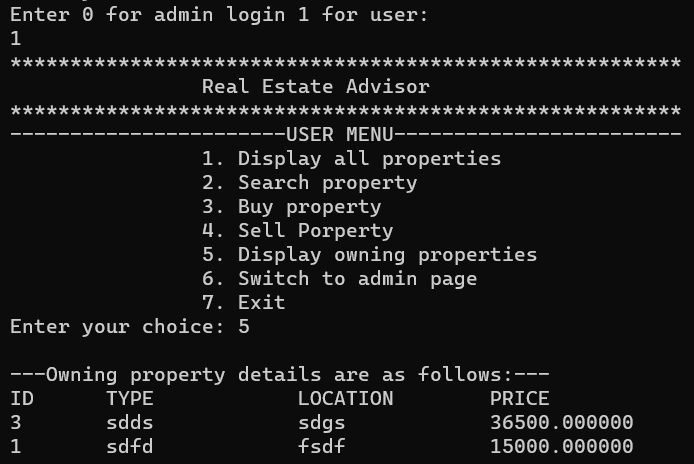
    }

    return 0;

}

(Screenshots):

****

****