C MICROPROJECT

NAME : GANGA ANI

ROLL NO: 34

COURSE NAME: C PROGRAMMING

DATE:22/07/2024

INTRODUCTION

Overview of the Project

The Pet Adoption System is a C program designed to streamline the process of adopting pets from shelters. This system allows users to view available pets, register their interest in adopting a pet, and complete the adoption process. The program is intended to make it easier for both the shelters and prospective pet owners to manage the adoption process efficiently.

Problem Statement

Pet adoption processes in many shelters are often managed manually, leading to inefficiencies, miscommunications, and delays. Prospective pet owners may face difficulties in finding suitable pets, while shelters struggle to keep track of available pets and manage adoption applications. This manual process can result in longer wait times for pets to find homes and increased workload for shelter staff.

OBJECTIVE

The main objective of this project is to develop a comprehensive and user-friendly Pet Adoption System using the C programming language. The system aims to:

1. Automate the Pet Adoption Process: Provide a digital platform for managing pet adoption, reducing the need for manual record-keeping.

2. Improve Accessibility: Allow prospective pet owners to easily browse available pets and apply for adoption online.

3. Enhance Efficiency: Streamline the workflow for shelter staff, enabling them to quickly update pet information and process adoption applications.

4. Facilitate Better Matches: Help prospective pet owners find pets that match their preferences, increasing the likelihood of successful adoptions.

By addressing these objectives, the Pet Adoption System aims to improve the overall efficiency of pet adoption processes, reduce the burden on shelter staff, and help more pets find loving homes.

SYSTEM REQUIREMENTS

Hardware

1. **Processor:** Minimum 1 GHz processor (Recommended: 2 GHz or higher)
2. **RAM:** Minimum 1 GB RAM (Recommended: 2 GB or higher)
3. **Storage:** Minimum 100 MB of free disk space (Recommended: 500 MB or higher)
4. **Display:** 800x600 resolution (Recommended: 1024x768 or higher)
5. **Peripherals:** Keyboard and mouse or other pointing device

Software Requirements

6. **Operating System:**
    - Windows (7 or higher)
    - Linux (any modern distribution)
    - macOS (10.12 or higher)
7. **Compiler:**
    - GCC (GNU Compiler Collection) for Linux and macOS
    - MinGW or Microsoft Visual Studio for Windows
8. **IDE/Code Editor:**
    a. Code::Blocks, Dev-C++, Visual Studio Code, or any preferred C development environment
9. **Libraries:**
    a. Standard C Library (stdlib.h, stdio.h, string.h, etc.)
    b. Additional libraries if required (e.g., SQLite for database management, if applicable)

Design and Development

Program Logic

The Pet Adoption System is structured to facilitate efficient interactions between users (prospective pet adopters) and administrators (shelter staff). The system logic flows through several key processes, including user registration, login, pet management, and adoption applications.

1.User Interaction:

- Users can register and log in to access different functionalities.
- After logging in, users are presented with options based on their roles (regular user or administrator).

2.Administrator Functions:

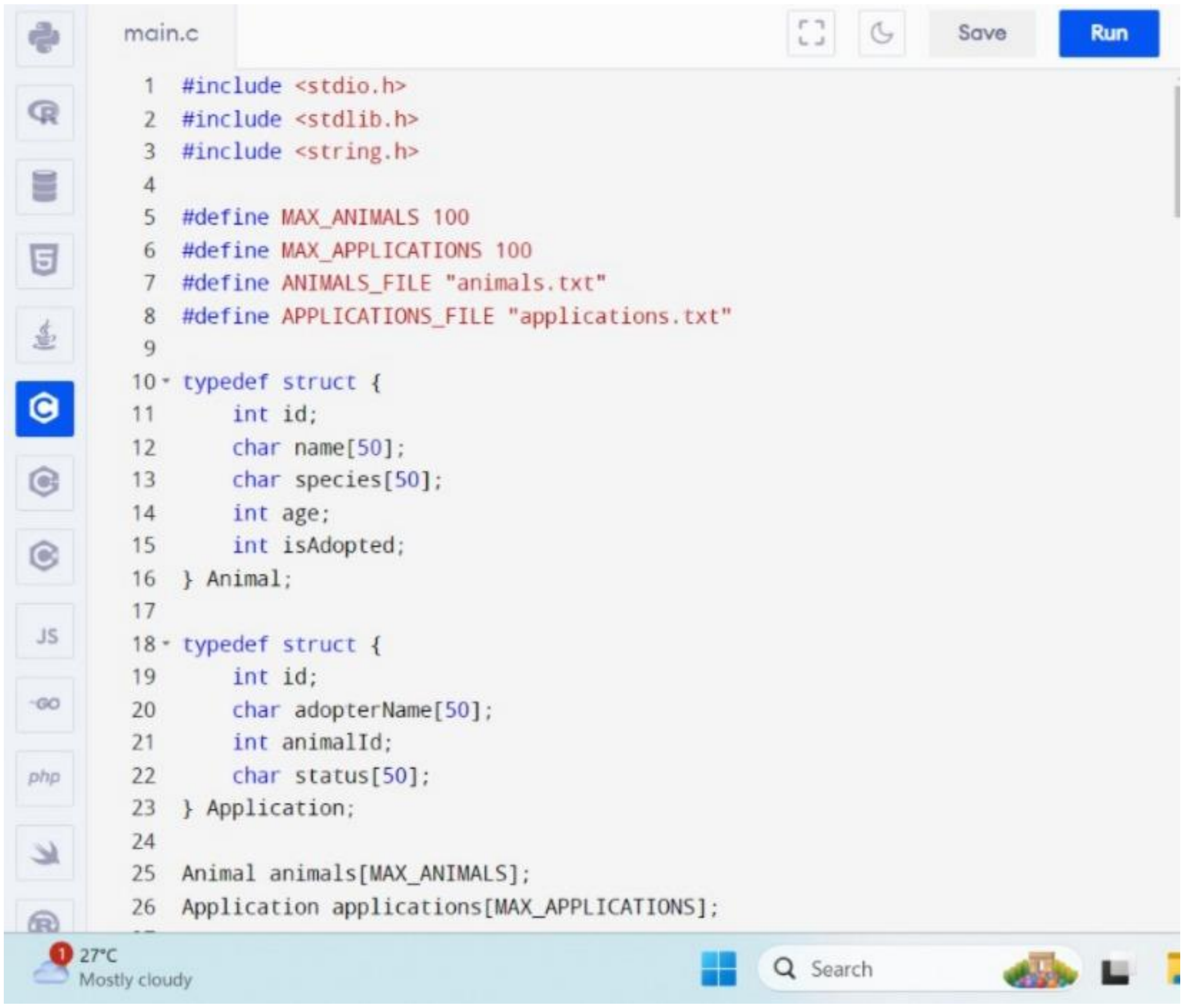
- Administrators can add, update, and delete pet records.
- They can view and manage adoption applications, approving or rejecting them as needed.

3.User Functions:

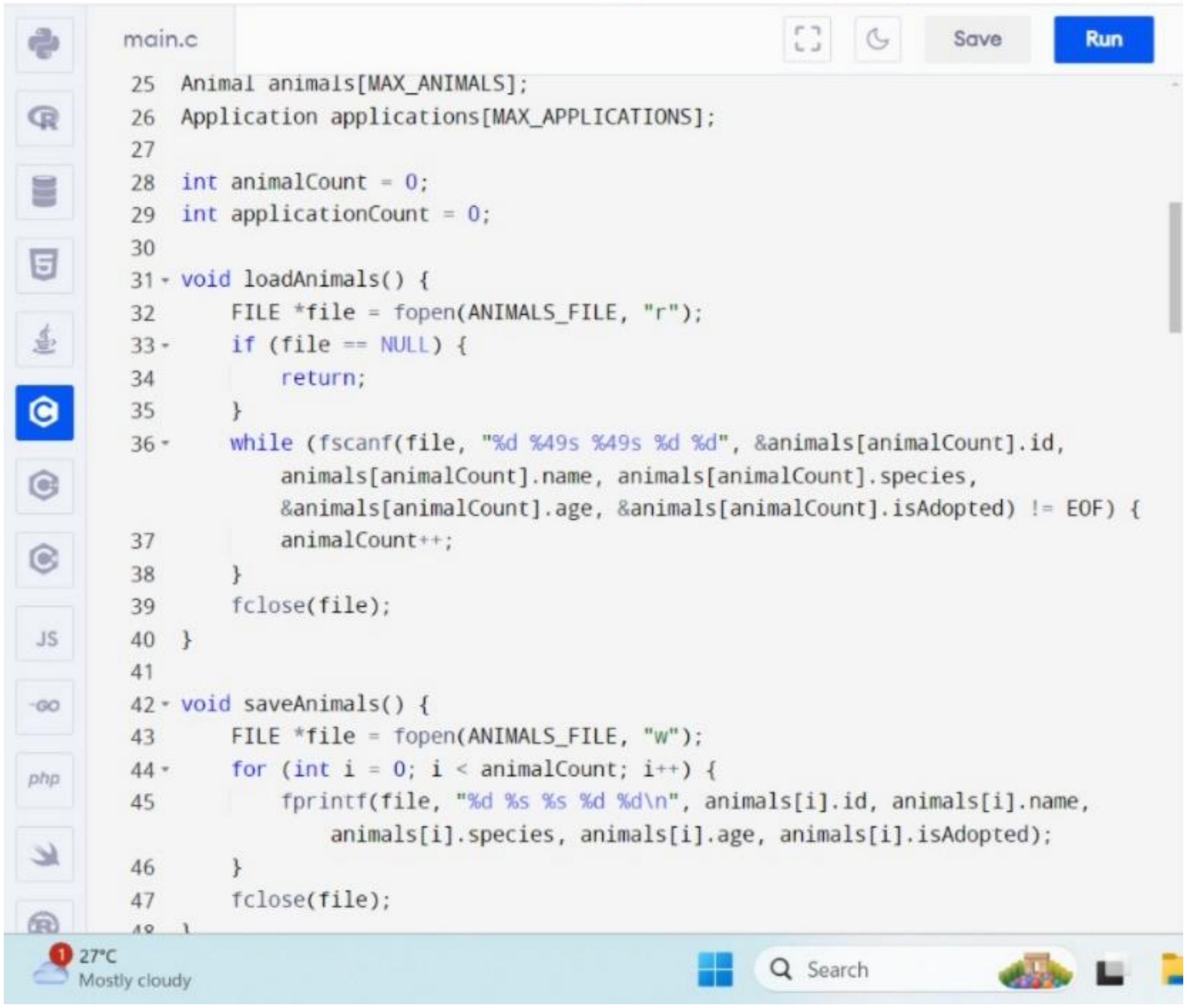
- Users can view a list of available pets, search for pets based on specific criteria, and apply for adoption.
- They can check the status of their adoption application

TESTING AND RESULTS

PROGRAM CODE



```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_ANIMALS 100
#define MAX_APPLICATIONS 100
#define ANIMALS_FILE "animals.txt"
#define APPLICATIONS_FILE "applications.txt"

typedef struct {
    int id;
    char name[50];
    char species[50];
    int age;
    int isAdopted;
} Animal;

typedef struct {
    int id;
    char adopterName[50];
    int animalId;
    char status[50];
} Application;

Animal animals[MAX_ANIMALS];
Application applications[MAX_APPLICATIONS];
```


main.c

Save

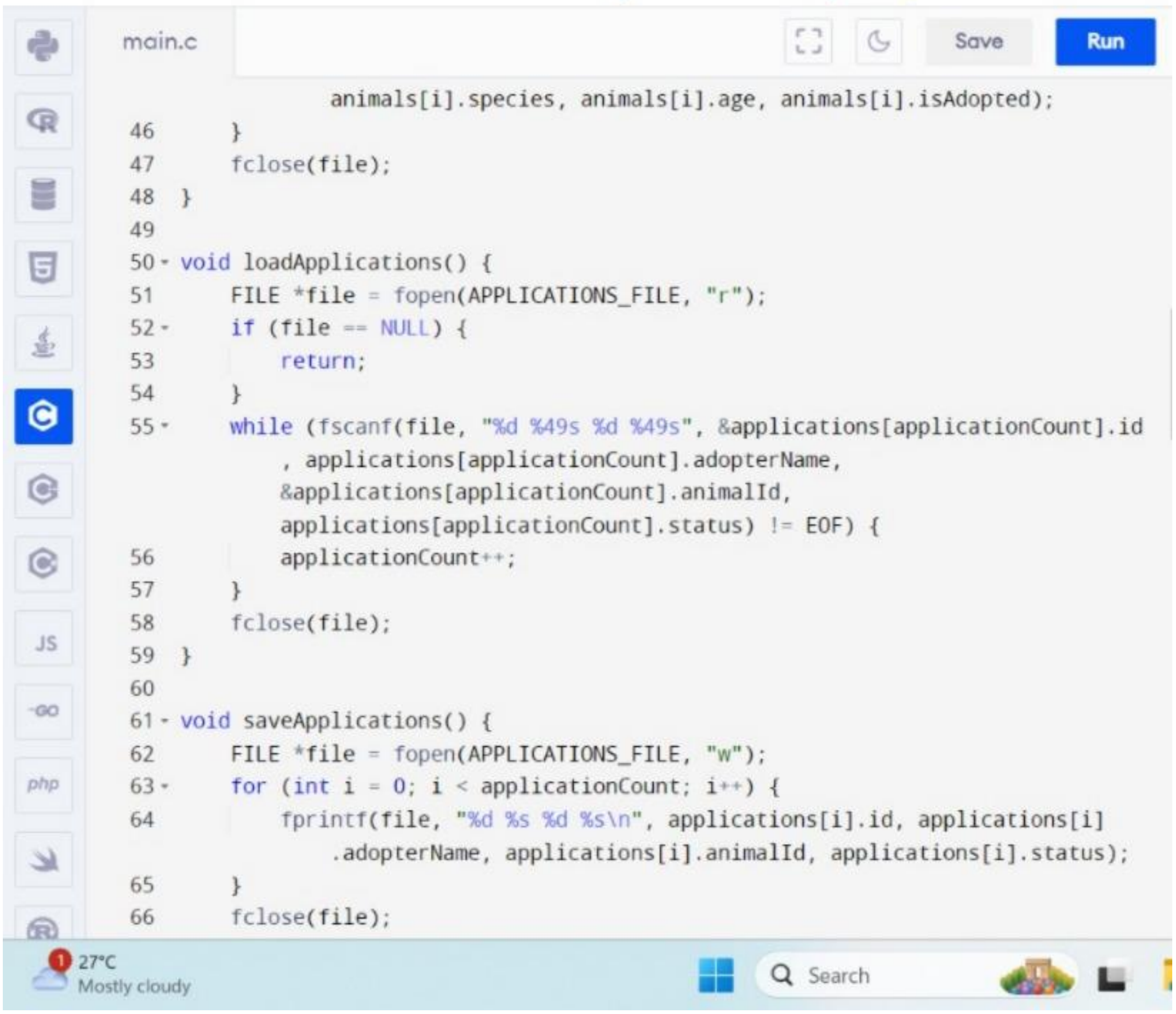
Run

```c
Animal animals[MAX_ANIMALS];
Application applications[MAX_APPLICATIONS];

int animalCount = 0;
int applicationCount = 0;

void loadAnimals() {
    FILE *file = fopen(ANIMALS_FILE, "r");
    if (file == NULL) {
        return;
    }
    while (fscanf(file, "%d %49s %49s %d %d", &animals[animalCount].id,
        animals[animalCount].name, animals[animalCount].species,
        &animals[animalCount].age, &animals[animalCount].isAdopted) != EOF) {
        animalCount++;
    }
    fclose(file);
}

void saveAnimals() {
    FILE *file = fopen(ANIMALS_FILE, "w");
    for (int i = 0; i < animalCount; i++) {
        fprintf(file, "%d %s %s %d %d\n", animals[i].id, animals[i].name,
            animals[i].species, animals[i].age, animals[i].isAdopted);
    }
    fclose(file);
```

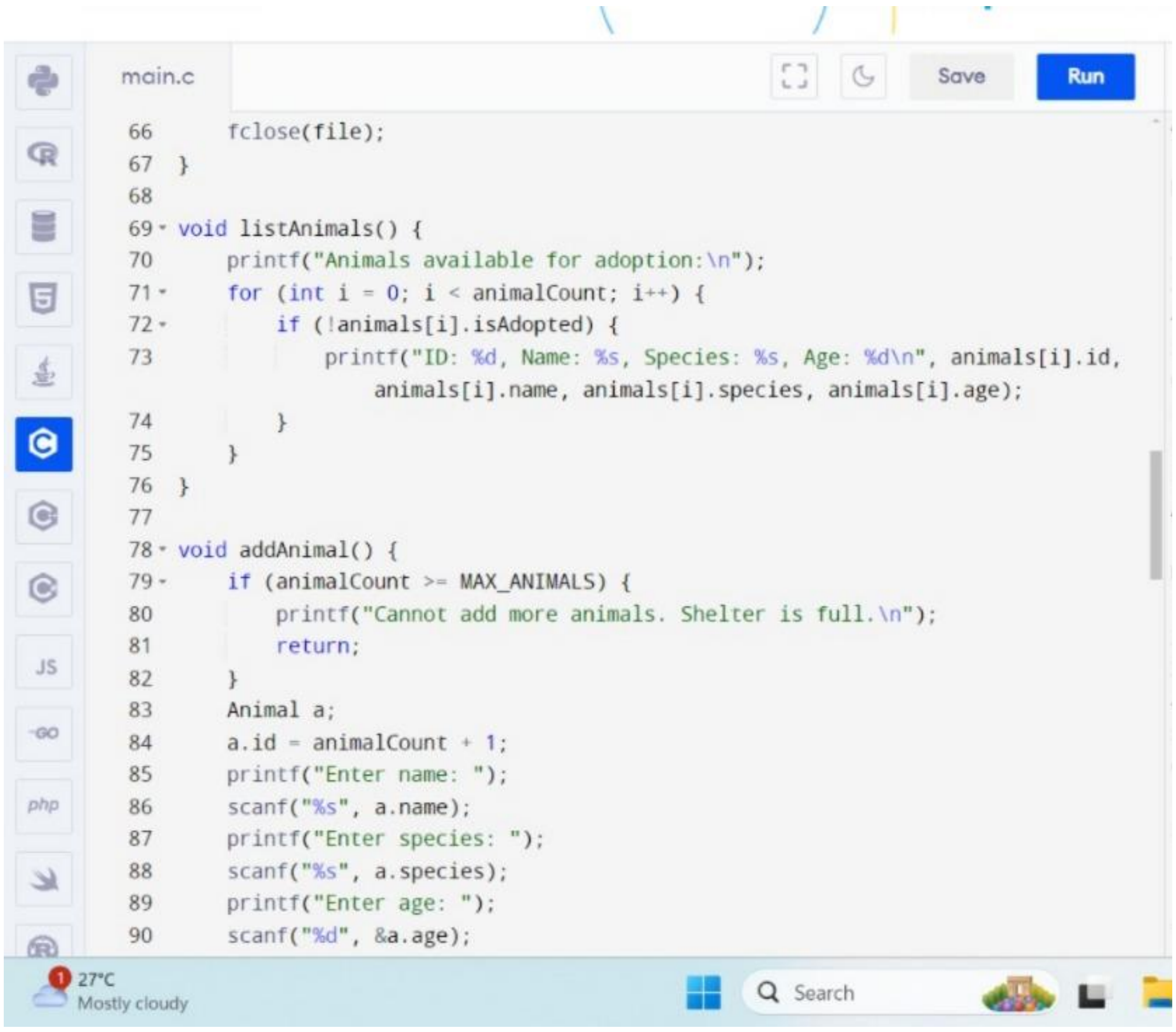
27°C
Mostly cloudy

Search

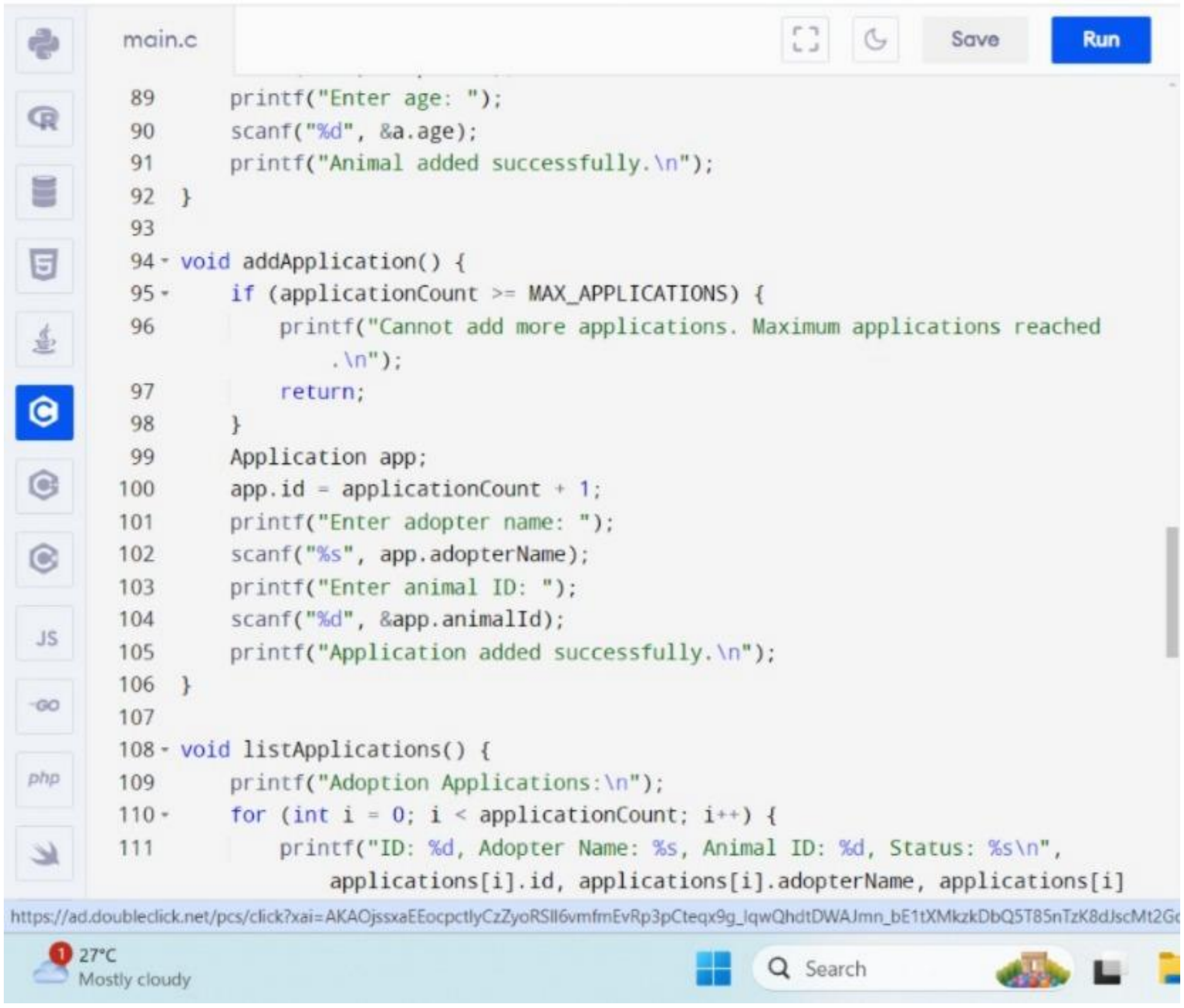
` ` ` ` ` ` ` ` ` `



```c
            animals[i].species, animals[i].age, animals[i].isAdopted);
    }
    fclose(file);
}

void loadApplications() {
    FILE *file = fopen(APPLICATIONS_FILE, "r");
    if (file == NULL) {
        return;
    }
    while (fscanf(file, "%d %49s %d %49s", &applications[applicationCount].id
        , applications[applicationCount].adopterName,
        &applications[applicationCount].animalId,
        applications[applicationCount].status) != EOF) {
        applicationCount++;
    }
    fclose(file);
}

void saveApplications() {
    FILE *file = fopen(APPLICATIONS_FILE, "w");
    for (int i = 0; i < applicationCount; i++) {
        fprintf(file, "%d %s %d %s\n", applications[i].id, applications[i]
            .adopterName, applications[i].animalId, applications[i].status);
    }
    fclose(file);
```

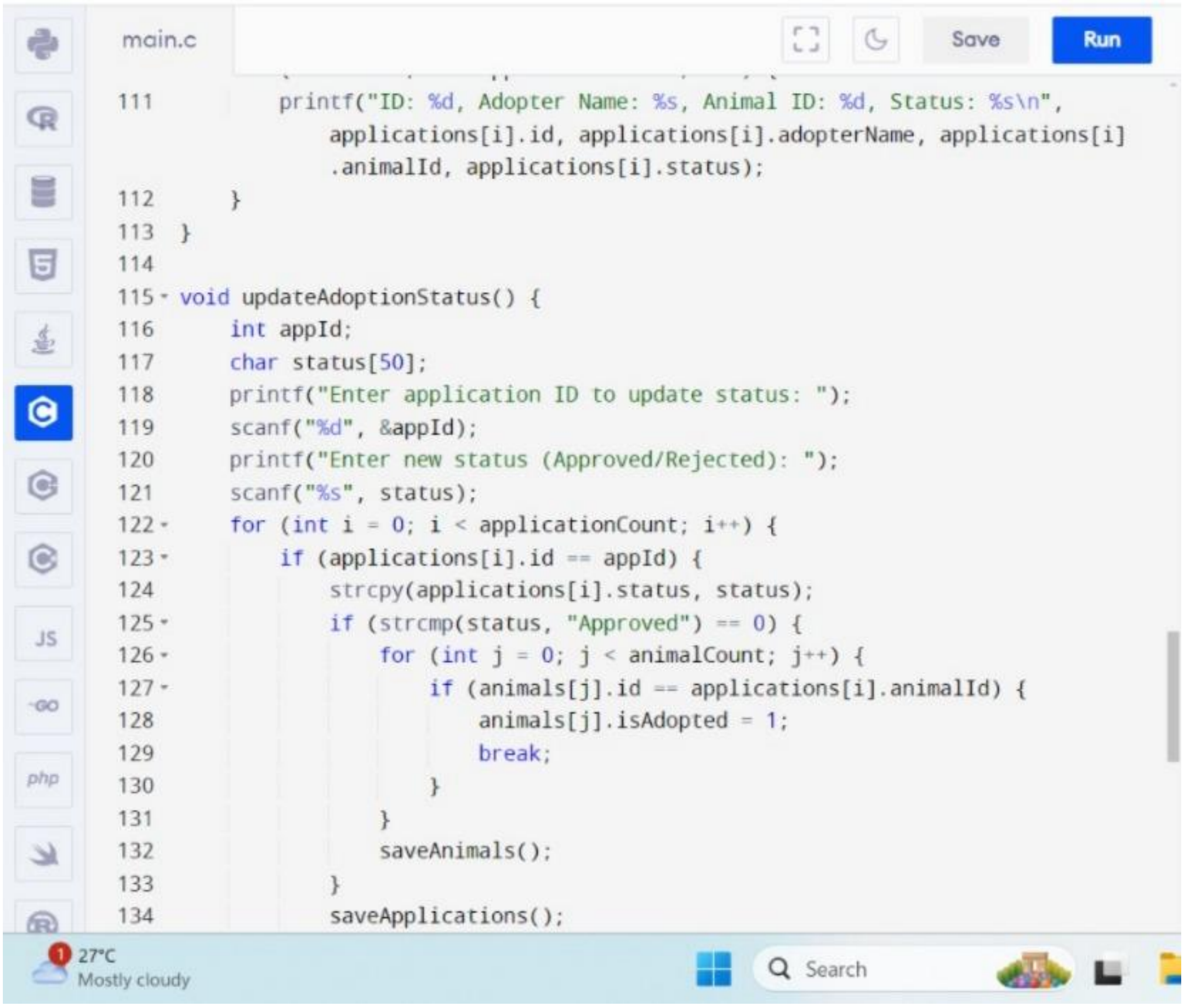


```c
    fclose(file);
}

void listAnimals() {
    printf("Animals available for adoption:\n");
    for (int i = 0; i < animalCount; i++) {
        if (!animals[i].isAdopted) {
            printf("ID: %d, Name: %s, Species: %s, Age: %d\n", animals[i].id,
                animals[i].name, animals[i].species, animals[i].age);
        }
    }
}

void addAnimal() {
    if (animalCount >= MAX_ANIMALS) {
        printf("Cannot add more animals. Shelter is full.\n");
        return;
    }
    Animal a;
    a.id = animalCount + 1;
    printf("Enter name: ");
    scanf("%s", a.name);
    printf("Enter species: ");
    scanf("%s", a.species);
    printf("Enter age: ");
    scanf("%d", &a.age);
```

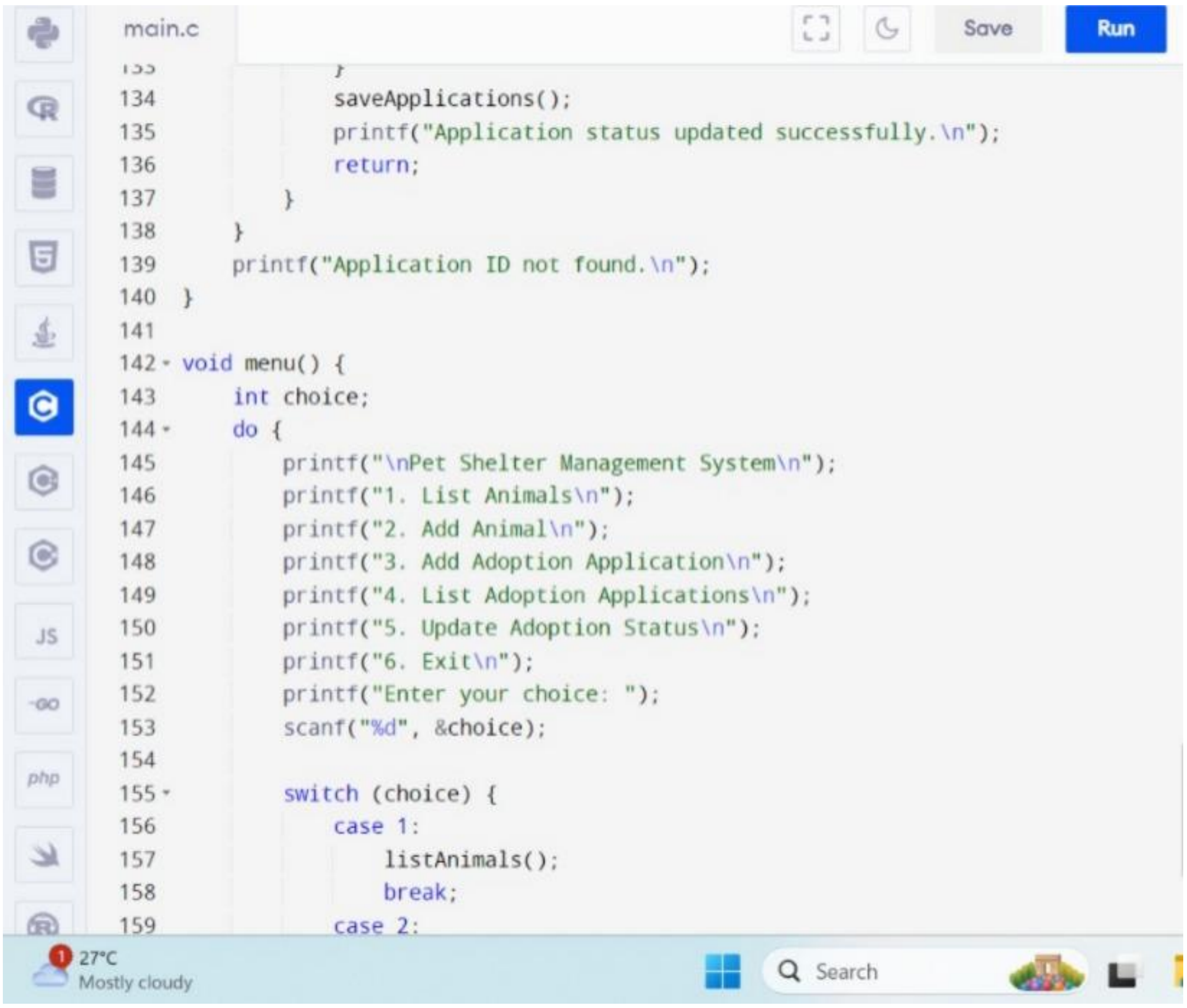


```
    printf("Enter age: ");
    scanf("%d", &a.age);
    printf("Animal added successfully.\n");
}

void addApplication() {
    if (applicationCount >= MAX_APPLICATIONS) {
        printf("Cannot add more applications. Maximum applications reached
            .\n");
        return;
    }
    Application app;
    app.id = applicationCount + 1;
    printf("Enter adopter name: ");
    scanf("%s", app.adopterName);
    printf("Enter animal ID: ");
    scanf("%d", &app.animalId);
    printf("Application added successfully.\n");
}

void listApplications() {
    printf("Adoption Applications:\n");
    for (int i = 0; i < applicationCount; i++) {
        printf("ID: %d, Adopter Name: %s, Animal ID: %d, Status: %s\n",
            applications[i].id, applications[i].adopterName, applications[i]
```

https://ad.doubleclick.net/pcs/click?xai=AKAOjssxaEEocpctlyCzZyoRSIl6vmfmEvRp3pCteqx9g_lqwQhdtDWAJmn_bE1tXMkzkDbQ5T85nTzK8dJscMt2G

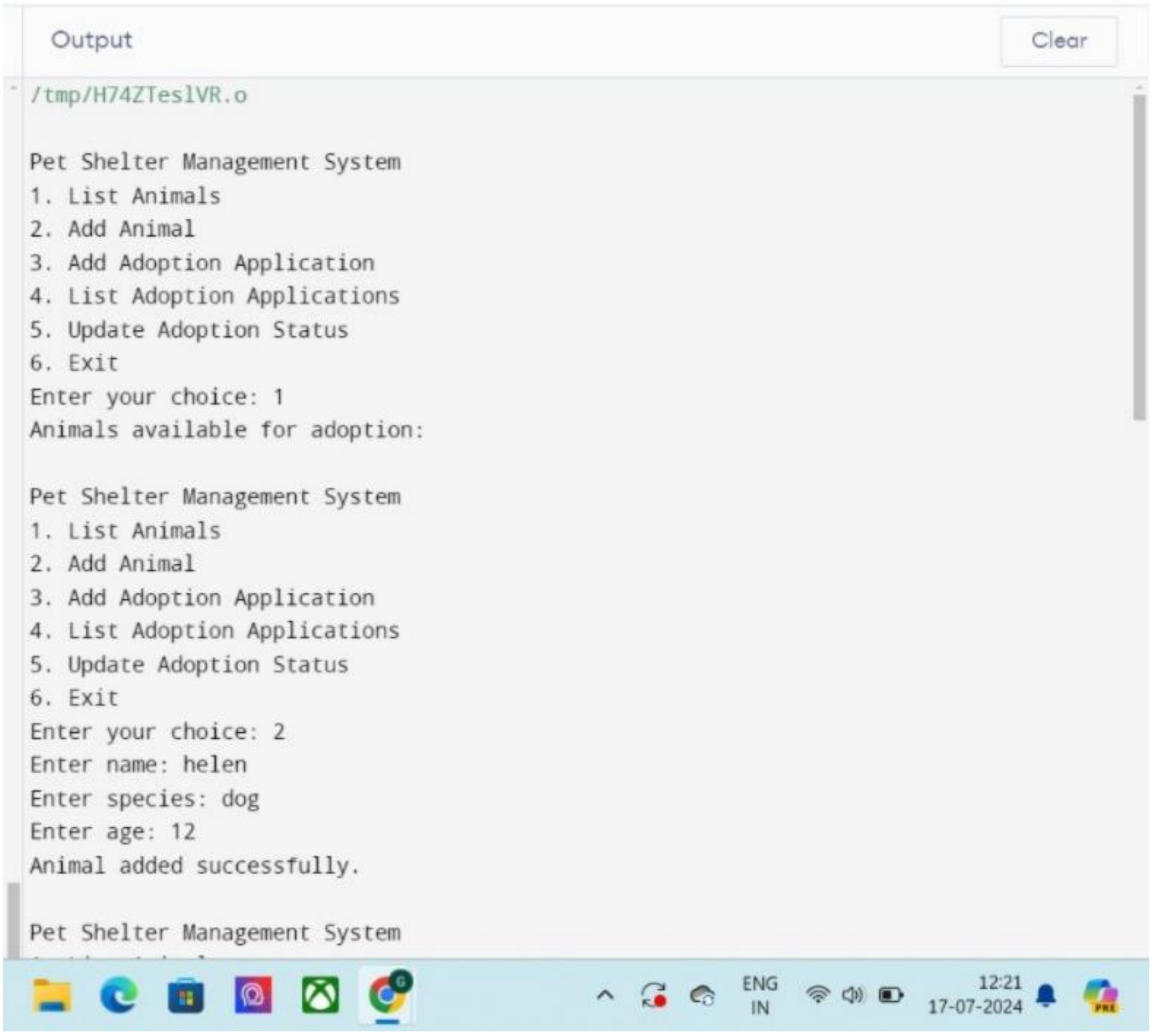


```c
        printf("ID: %d, Adopter Name: %s, Animal ID: %d, Status: %s\n",
            applications[i].id, applications[i].adopterName, applications[i]
            .animalId, applications[i].status);
    }
}

void updateAdoptionStatus() {
    int appId;
    char status[50];
    printf("Enter application ID to update status: ");
    scanf("%d", &appId);
    printf("Enter new status (Approved/Rejected): ");
    scanf("%s", status);
    for (int i = 0; i < applicationCount; i++) {
        if (applications[i].id == appId) {
            strcpy(applications[i].status, status);
            if (strcmp(status, "Approved") == 0) {
                for (int j = 0; j < animalCount; j++) {
                    if (animals[j].id == applications[i].animalId) {
                        animals[j].isAdopted = 1;
                        break;
                    }
                }
                saveAnimals();
            }
            saveApplications();
```

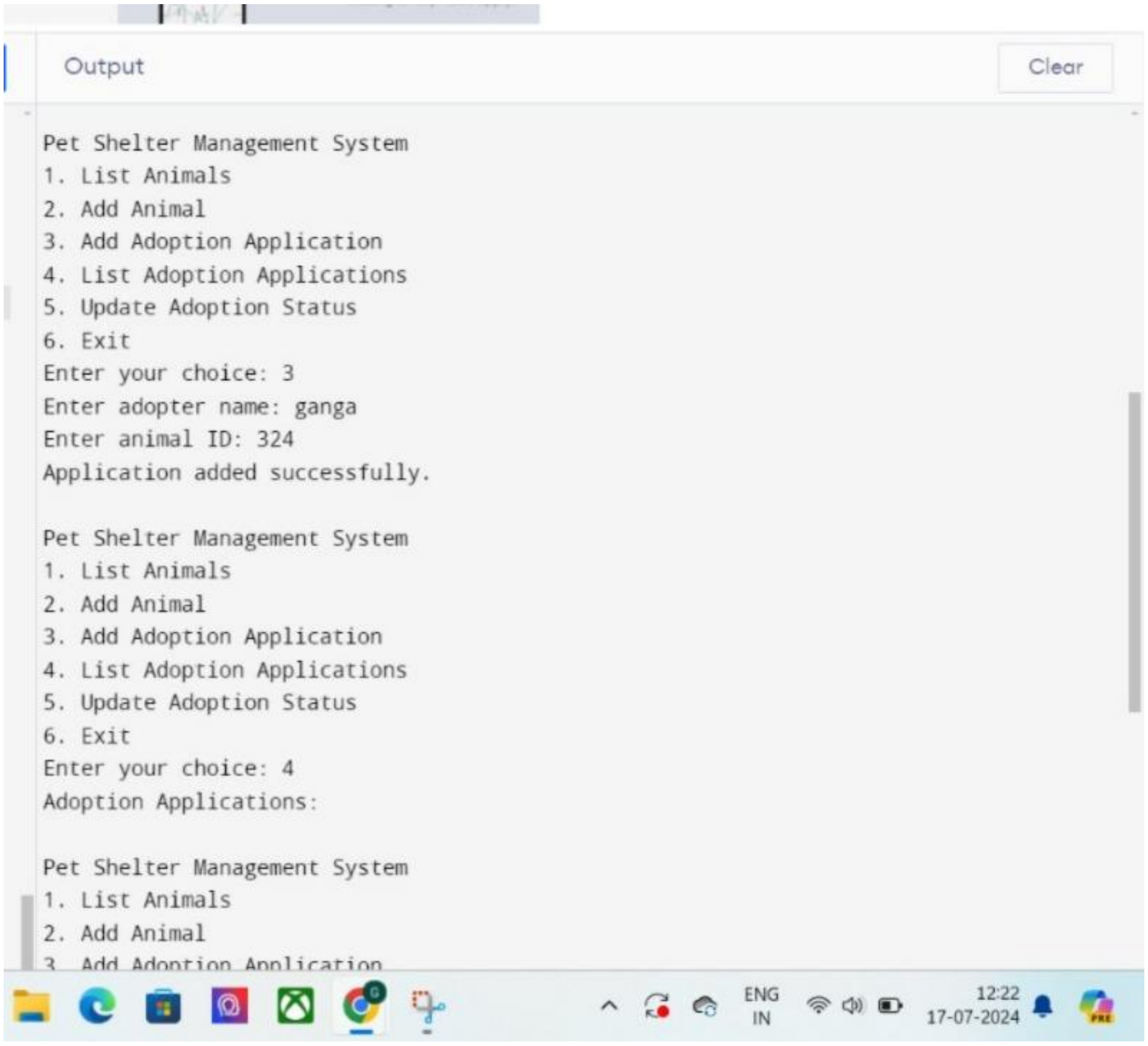


```c
                }
                saveApplications();
                printf("Application status updated successfully.\n");
                return;
            }
        }
        printf("Application ID not found.\n");
    }

void menu() {
    int choice;
    do {
        printf("\nPet Shelter Management System\n");
        printf("1. List Animals\n");
        printf("2. Add Animal\n");
        printf("3. Add Adoption Application\n");
        printf("4. List Adoption Applications\n");
        printf("5. Update Adoption Status\n");
        printf("6. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                listAnimals();
                break;
            case 2:
```

OUTPUT

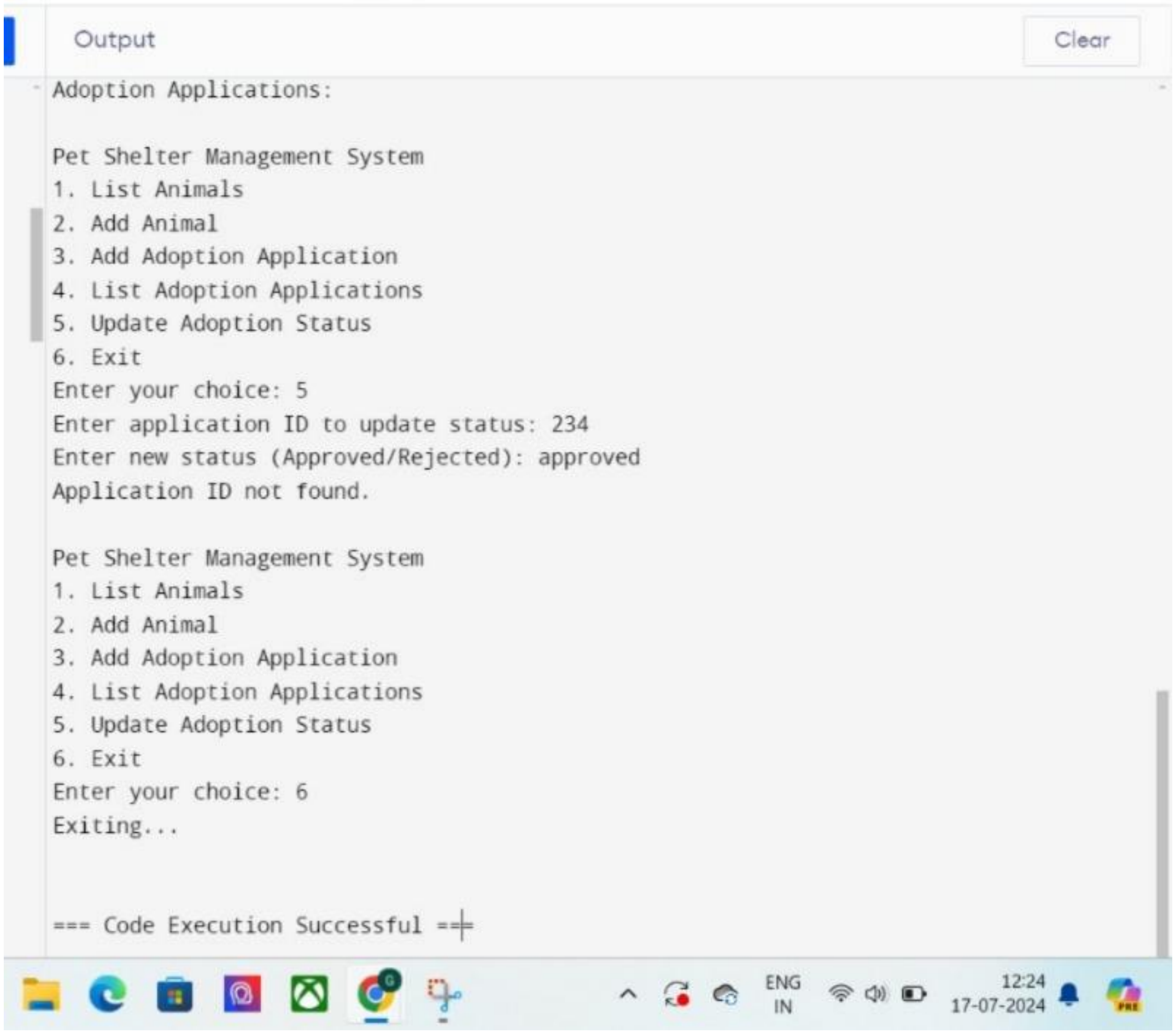


```
/tmp/H74ZTeslVR.o

Pet Shelter Management System
1. List Animals
2. Add Animal
3. Add Adoption Application
4. List Adoption Applications
5. Update Adoption Status
6. Exit
Enter your choice: 1
Animals available for adoption:

Pet Shelter Management System
1. List Animals
2. Add Animal
3. Add Adoption Application
4. List Adoption Applications
5. Update Adoption Status
6. Exit
Enter your choice: 2
Enter name: helen
Enter species: dog
Enter age: 12
Animal added successfully.

Pet Shelter Management System
```


Output

Clear

```
Pet Shelter Management System
1. List Animals
2. Add Animal
3. Add Adoption Application
4. List Adoption Applications
5. Update Adoption Status
6. Exit
Enter your choice: 3
Enter adopter name: ganga
Enter animal ID: 324
Application added successfully.

Pet Shelter Management System
1. List Animals
2. Add Animal
3. Add Adoption Application
4. List Adoption Applications
5. Update Adoption Status
6. Exit
Enter your choice: 4
Adoption Applications:

Pet Shelter Management System
1. List Animals
2. Add Animal
3 Add Adoption Application
```



```
Adoption Applications:

Pet Shelter Management System
1. List Animals
2. Add Animal
3. Add Adoption Application
4. List Adoption Applications
5. Update Adoption Status
6. Exit
Enter your choice: 5
Enter application ID to update status: 234
Enter new status (Approved/Rejected): approved
Application ID not found.

Pet Shelter Management System
1. List Animals
2. Add Animal
3. Add Adoption Application
4. List Adoption Applications
5. Update Adoption Status
6. Exit
Enter your choice: 6
Exiting...


=== Code Execution Successful ===
```

Conclusions

Summary of the Project

The Pet Adoption System developed in C is a comprehensive solution designed to streamline the pet adoption process for both prospective pet owners and shelter administrators. The system provides functionalities for user registration, login, pet management, and adoption application processing. By leveraging file operations, the system ensures data persistence, allowing users and administrators to perform their tasks efficiently.

Future Enhancements

While the current system provides a robust foundation for managing pet adoptions, there are several enhancements that could further improve its functionality and user experience:

1.Graphical User Interface (GUI)

2.Database Integration

3.Online Access:

4.Advanced Search and Filtering:

5.Notification System:

6.Mobile Application:

7.Enhanced Security:

8.Analytics and Reporting:

By implementing these future enhancements, the Pet Adoption System can evolve into a more versatile and user-centric platform, further simplifying the adoption process and helping more pets find loving homes.

REFERENCES

1.CHATGPT

2.BY REFERRING TEXTBOOK(BY BALAGURUSAMY
)