CSE101 FINAL REPORT

TOPIC: CA EVALUATOR SYSTEM

DESCRIPTION: The CA evaluation system is a computer program designed to automate the process of evaluating the performance of students in a course over a period of time. This system is commonly used in educational institutions to measure the progress of students throughout a semester or academic year. The main aim of this project is to develop a CA evaluation system using the C programming language.
The primary objective of this project is to create a CA evaluation system that can automate the process of evaluating student performance in a course. The system will be designed to handle the following tasks:

- Create Record
- Search Record
- Display Evaluation Criteria
- Modify Record
- Delete Record
- Display Record
- Display The Details

MODULES EXPLANATION:

- Create Record: The Create Record module enables users to create a new student record in the system. This module allows users to input all the necessary information related to a student, such as their name, ID number, course, and other relevant details. Users can also input additional information such as the student's contact information, enrolment status, and any other relevant notes.

- Search Record: The Search Record module allows users to search for specific student records based on their name, ID number, or other relevant information. This module provides a search bar where users can enter the search criteria and find the desired student record. The search function is designed to be fast and efficient, enabling users to find the relevant records quickly.

- Display Evaluation Criteria: The Display Evaluation Criteria module enables users to view the evaluation criteria for each course in the system. Users can view the weightage of each component of the evaluation, such as assignments, quizzes, and final exams. This module provides users with a comprehensive overview of the evaluation process, enabling them to make informed decisions while evaluating student performance.

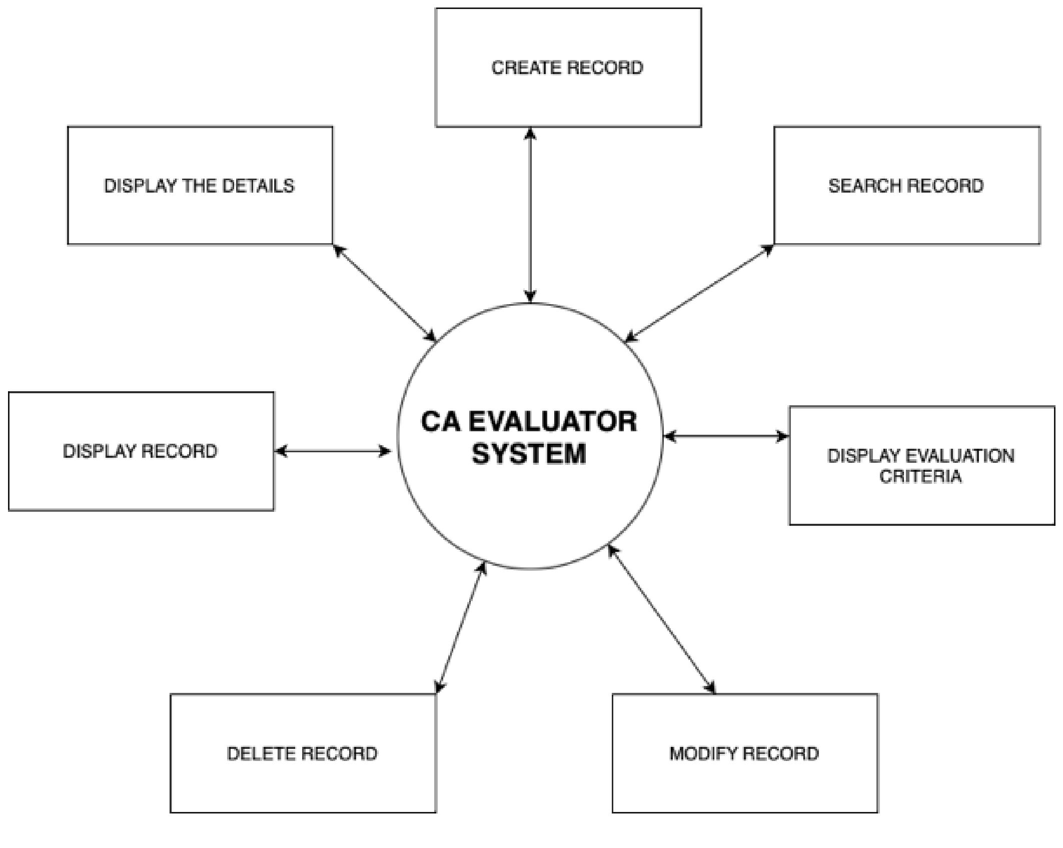
- Modify Record: The Modify Record module allows users to modify an existing student record in the system. Users can update the student details, change the evaluation scores, or add any other relevant information to the record. This module enables users to make any necessary changes to the student record without having to create a new record from scratch.

- Delete Record: The Delete Record module enables users to delete an existing student record from the system. This module is designed to be user-friendly and secure, ensuring that users can delete records without accidentally deleting the wrong record. Users can select the record they wish to delete and confirm the action before proceeding.

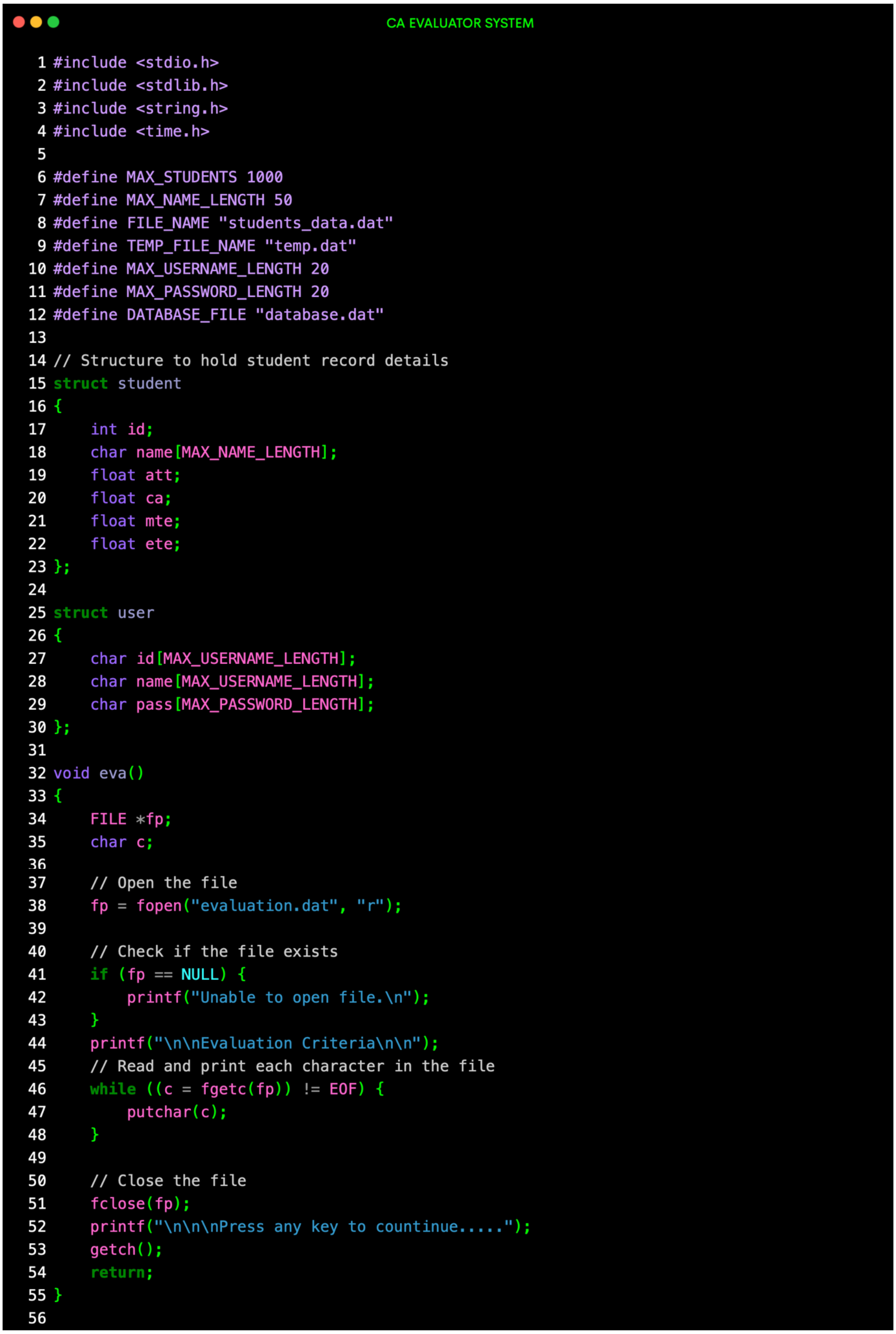
- Display Record: The Display Record module allows users to view all the student records in the system. Users can view the student details and their evaluation scores in a tabular format. This module provides users with an overview of all the student records, enabling them to identify any trends or patterns in student performance.

- Display The Details: The Display The Details module provides detailed information about each student record. Users can view the evaluation scores for each component of the evaluation, such as assignments, quizzes, and final exams. Additionally, users can view any notes or comments made by the evaluator. This module enables users to view all the relevant information related to a student record in one place.

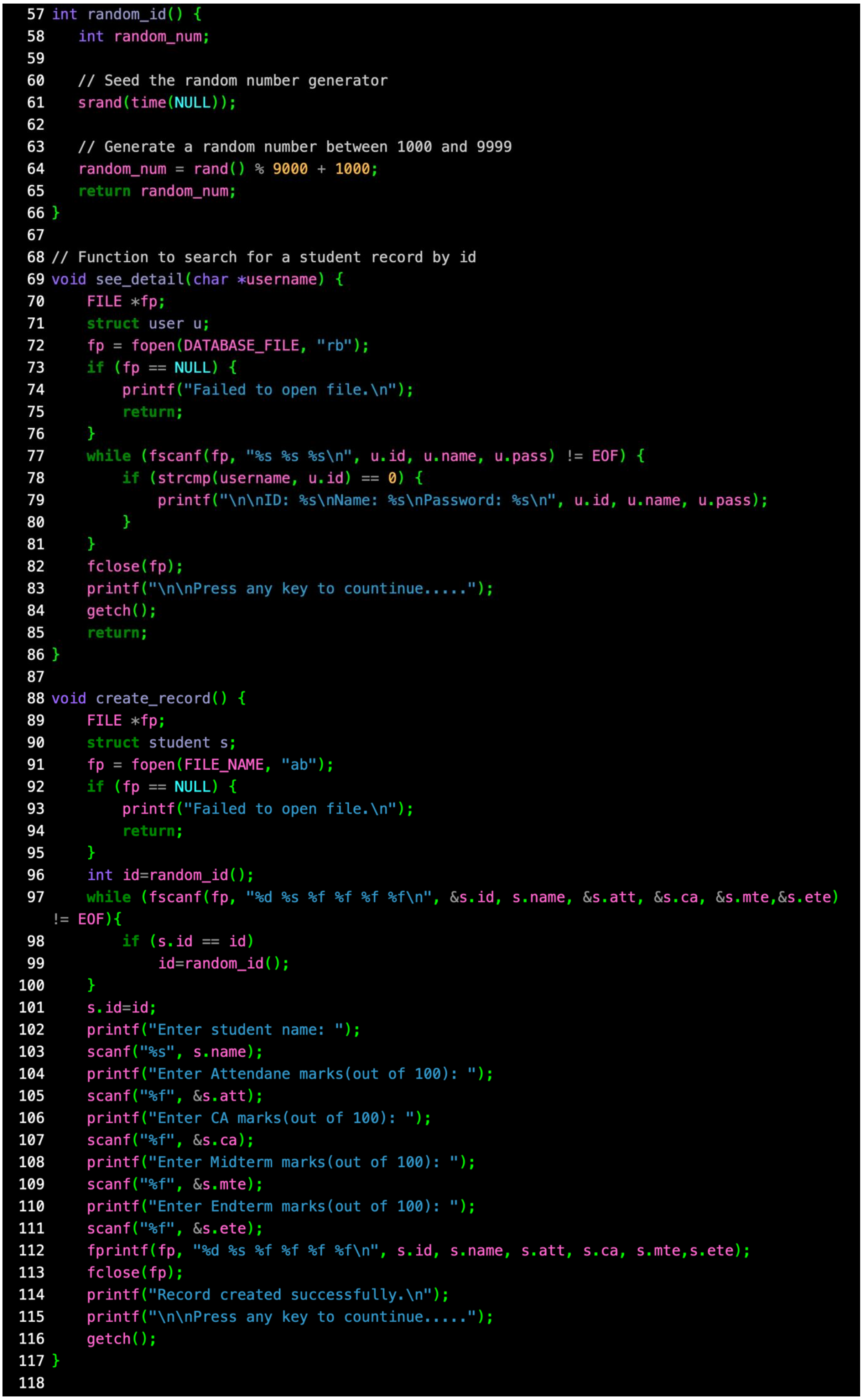
DATA FLOW DIAGRAM (LEVEL 0):



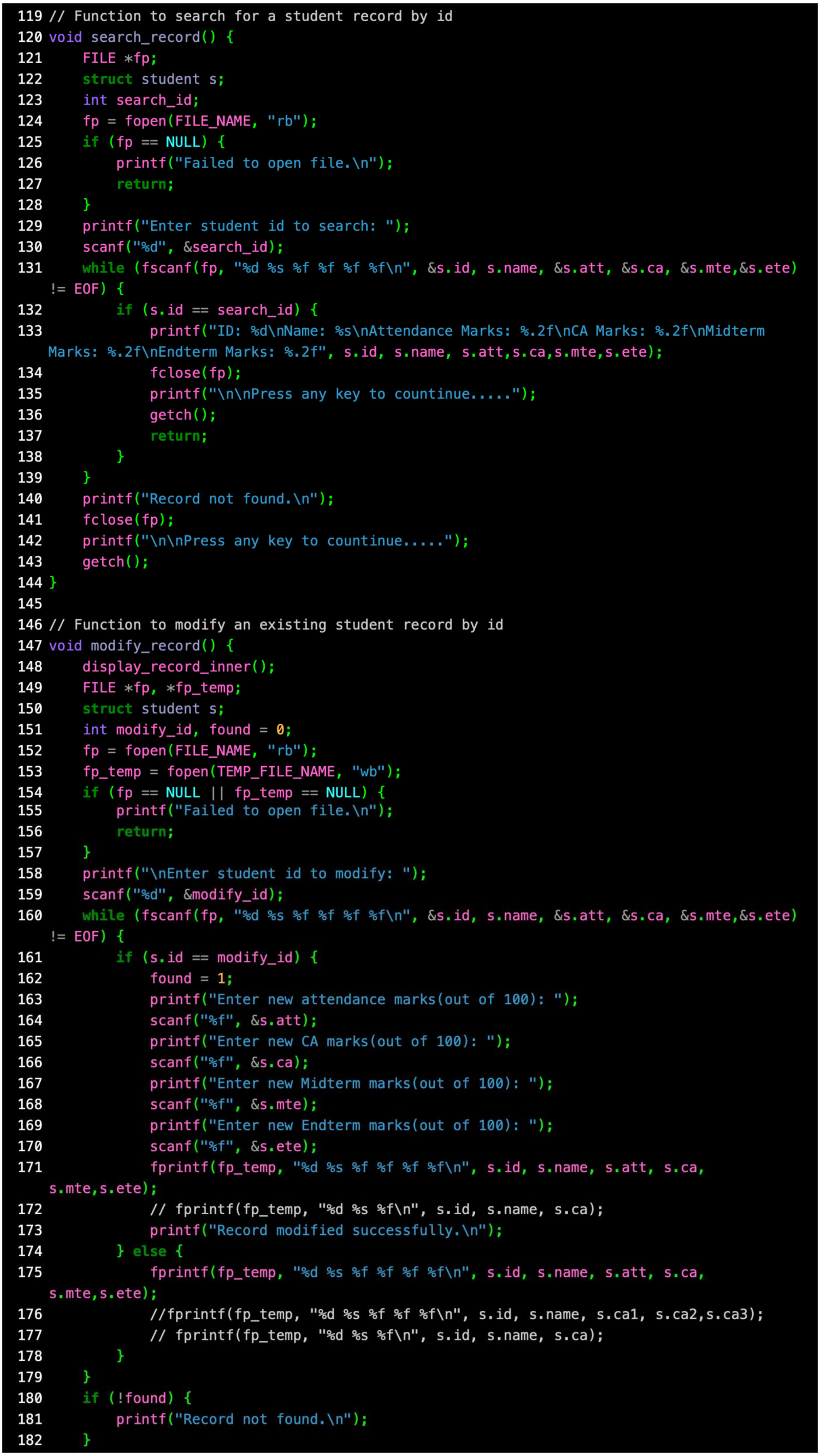
PROGRAMMING CODE:



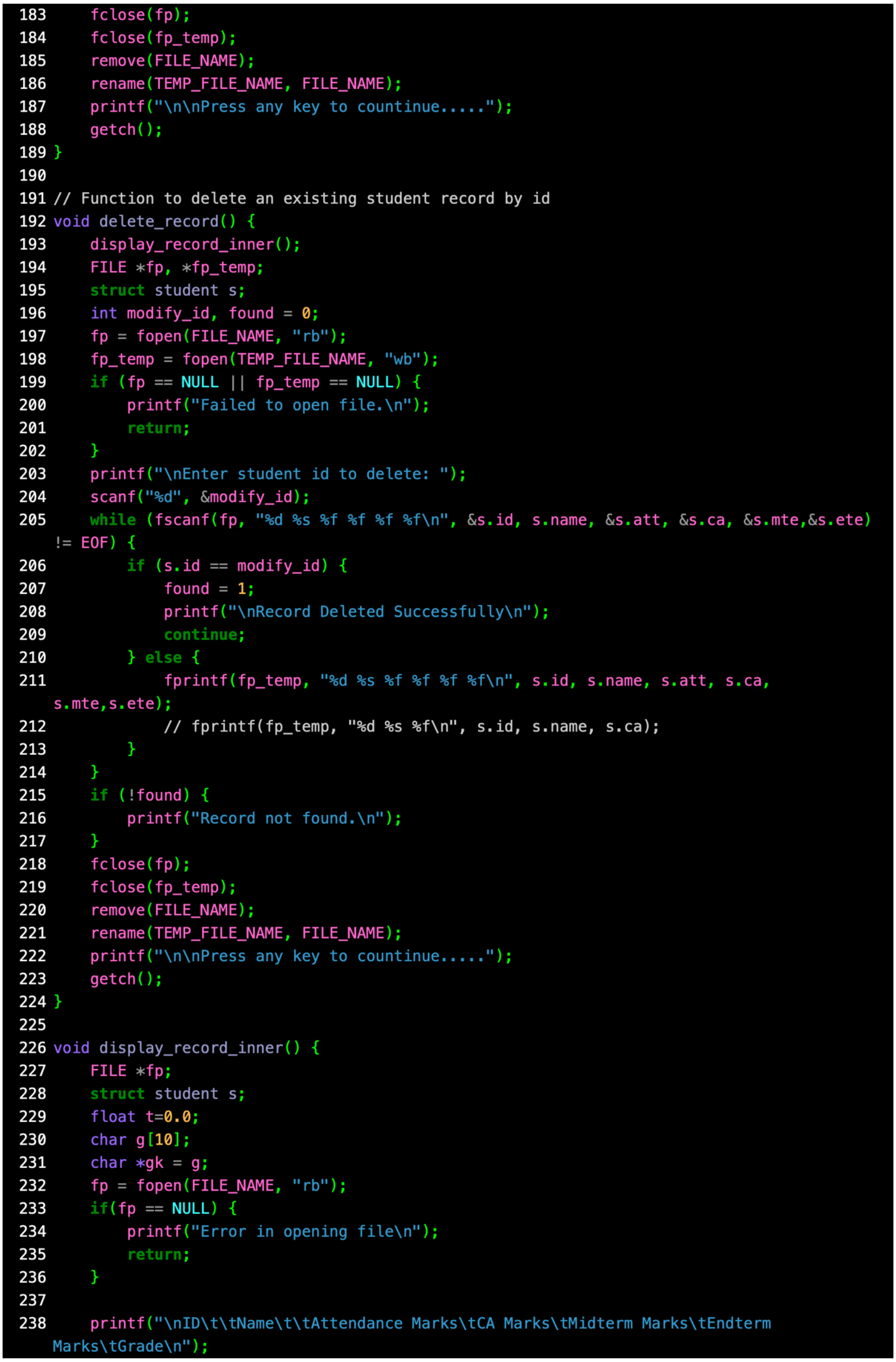
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

#define MAX_STUDENTS 1000
#define MAX_NAME_LENGTH 50
#define FILE_NAME "students_data.dat"
#define TEMP_FILE_NAME "temp.dat"
#define MAX_USERNAME_LENGTH 20
#define MAX_PASSWORD_LENGTH 20
#define DATABASE_FILE "database.dat"

// Structure to hold student record details
struct student
{
    int id;
    char name[MAX_NAME_LENGTH];
    float att;
    float ca;
    float mte;
    float ete;
};

struct user
{
    char id[MAX_USERNAME_LENGTH];
    char name[MAX_USERNAME_LENGTH];
    char pass[MAX_PASSWORD_LENGTH];
};

void eva()
{
    FILE *fp;
    char c;

    // Open the file
    fp = fopen("evaluation.dat", "r");

    // Check if the file exists
    if (fp == NULL) {
        printf("Unable to open file.\n");
    }
    printf("\n\nEvaluation Criteria\n\n");
    // Read and print each character in the file
    while ((c = fgetc(fp)) != EOF) {
        putchar(c);
    }

    // Close the file
    fclose(fp);
    printf("\n\n\nPress any key to countinue.....");
    getch();
    return;
}

```



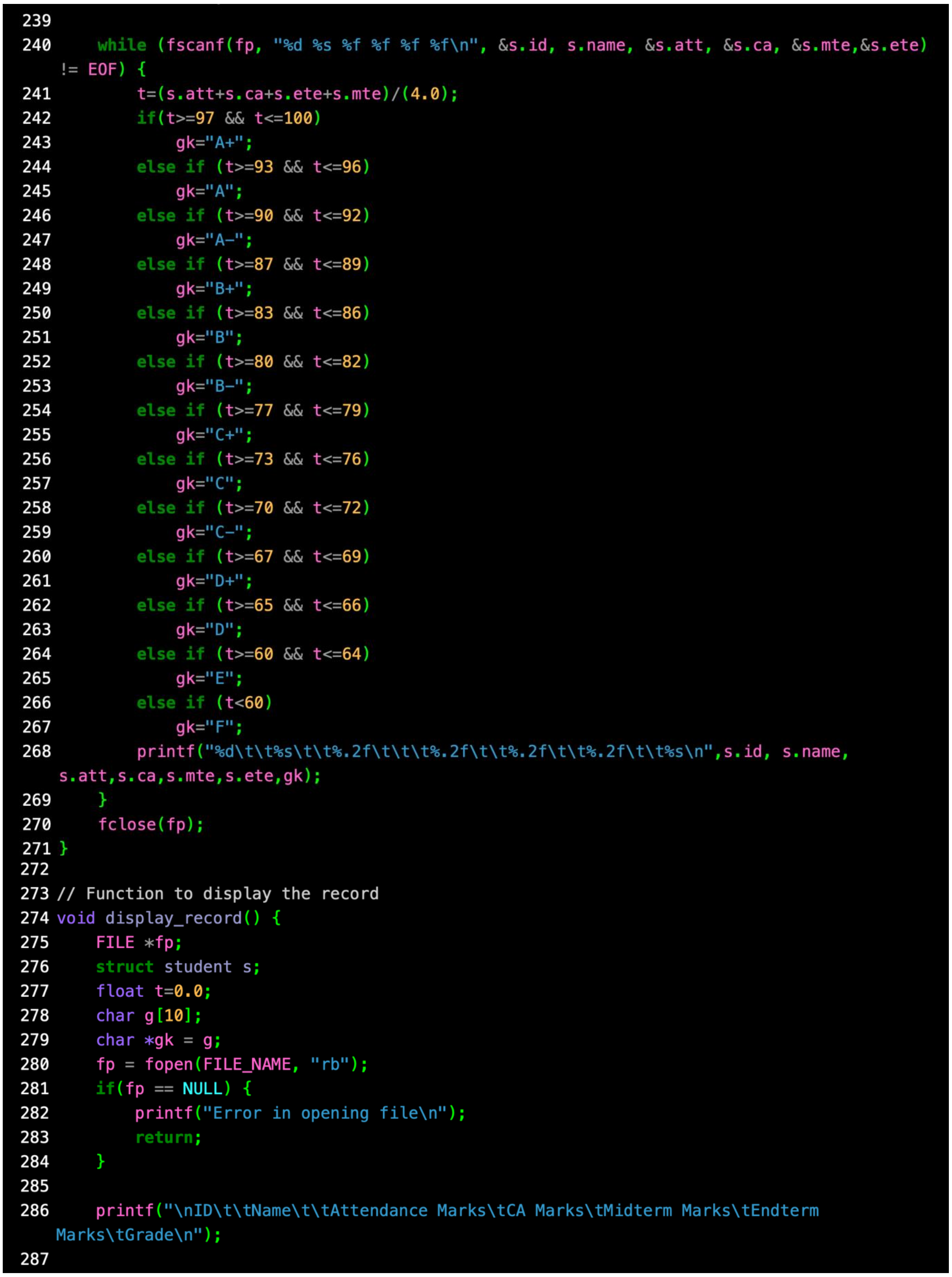
```c
int random_id() {
    int random_num;

    // Seed the random number generator
    srand(time(NULL));

    // Generate a random number between 1000 and 9999
    random_num = rand() % 9000 + 1000;
    return random_num;
}

// Function to search for a student record by id
void see_detail(char *username) {
    FILE *fp;
    struct user u;
    fp = fopen(DATABASE_FILE, "rb");
    if (fp == NULL) {
        printf("Failed to open file.\n");
        return;
    }
    while (fscanf(fp, "%s %s %s\n", u.id, u.name, u.pass) != EOF) {
        if (strcmp(username, u.id) == 0) {
            printf("\n\nID: %s\nName: %s\nPassword: %s\n", u.id, u.name, u.pass);
        }
    }
    fclose(fp);
    printf("\n\nPress any key to countinue.....");
    getch();
    return;
}

void create_record() {
    FILE *fp;
    struct student s;
    fp = fopen(FILE_NAME, "ab");
    if (fp == NULL) {
        printf("Failed to open file.\n");
        return;
    }
    int id=random_id();
    while (fscanf(fp, "%d %s %f %f %f %f\n", &s.id, s.name, &s.att, &s.ca, &s.mte,&s.ete) != EOF){
        if (s.id == id)
            id=random_id();
    }
    s.id=id;
    printf("Enter student name: ");
    scanf("%s", s.name);
    printf("Enter Attendane marks(out of 100): ");
    scanf("%f", &s.att);
    printf("Enter CA marks(out of 100): ");
    scanf("%f", &s.ca);
    printf("Enter Midterm marks(out of 100): ");
    scanf("%f", &s.mte);
    printf("Enter Endterm marks(out of 100): ");
    scanf("%f", &s.ete);
    fprintf(fp, "%d %s %f %f %f %f\n", s.id, s.name, s.att, s.ca, s.mte,s.ete);
    fclose(fp);
    printf("Record created successfully.\n");
    printf("\n\nPress any key to countinue.....");
    getch();
}

```



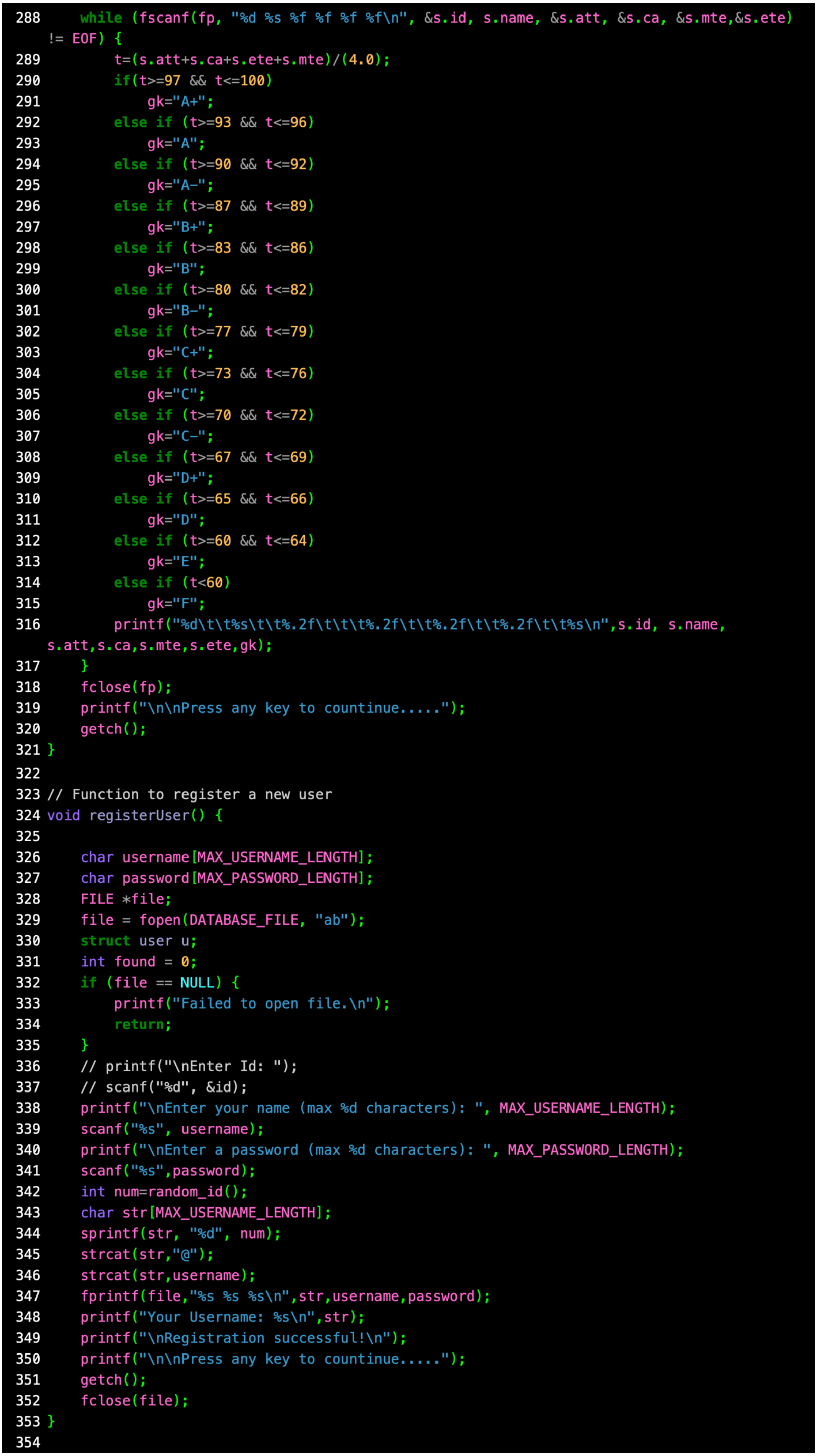
```c
// Function to search for a student record by id
void search_record() {
    FILE *fp;
    struct student s;
    int search_id;
    fp = fopen(FILE_NAME, "rb");
    if (fp == NULL) {
        printf("Failed to open file.\n");
        return;
    }
    printf("Enter student id to search: ");
    scanf("%d", &search_id);
    while (fscanf(fp, "%d %s %f %f %f %f\n", &s.id, s.name, &s.att, &s.ca, &s.mte,&s.ete) != EOF) {
        if (s.id == search_id) {
            printf("ID: %d\nName: %s\nAttendance Marks: %.2f\nCA Marks: %.2f\nMidterm Marks: %.2f\nEndterm Marks: %.2f", s.id, s.name, s.att,s.ca,s.mte,s.ete);
            fclose(fp);
            printf("\n\nPress any key to countinue.....");
            getch();
            return;
        }
    }
    printf("Record not found.\n");
    fclose(fp);
    printf("\n\nPress any key to countinue.....");
    getch();
}

// Function to modify an existing student record by id
void modify_record() {
    display_record_inner();
    FILE *fp, *fp_temp;
    struct student s;
    int modify_id, found = 0;
    fp = fopen(FILE_NAME, "rb");
    fp_temp = fopen(TEMP_FILE_NAME, "wb");
    if (fp == NULL || fp_temp == NULL) {
        printf("Failed to open file.\n");
        return;
    }
    printf("\nEnter student id to modify: ");
    scanf("%d", &modify_id);
    while (fscanf(fp, "%d %s %f %f %f %f\n", &s.id, s.name, &s.att, &s.ca, &s.mte,&s.ete) != EOF) {
        if (s.id == modify_id) {
            found = 1;
            printf("Enter new attendance marks(out of 100): ");
            scanf("%f", &s.att);
            printf("Enter new CA marks(out of 100): ");
            scanf("%f", &s.ca);
            printf("Enter new Midterm marks(out of 100): ");
            scanf("%f", &s.mte);
            printf("Enter new Endterm marks(out of 100): ");
            scanf("%f", &s.ete);
            fprintf(fp_temp, "%d %s %f %f %f %f\n", s.id, s.name, s.att, s.ca, s.mte,s.ete);
            // fprintf(fp_temp, "%d %s %f\n", s.id, s.name, s.ca);
            printf("Record modified successfully.\n");
        } else {
            fprintf(fp_temp, "%d %s %f %f %f %f\n", s.id, s.name, s.att, s.ca, s.mte,s.ete);
            //fprintf(fp_temp, "%d %s %f %f %f\n", s.id, s.name, s.ca1, s.ca2,s.ca3);
            // fprintf(fp_temp, "%d %s %f\n", s.id, s.name, s.ca);
        }
    }
    if (!found) {
        printf("Record not found.\n");
    }
```



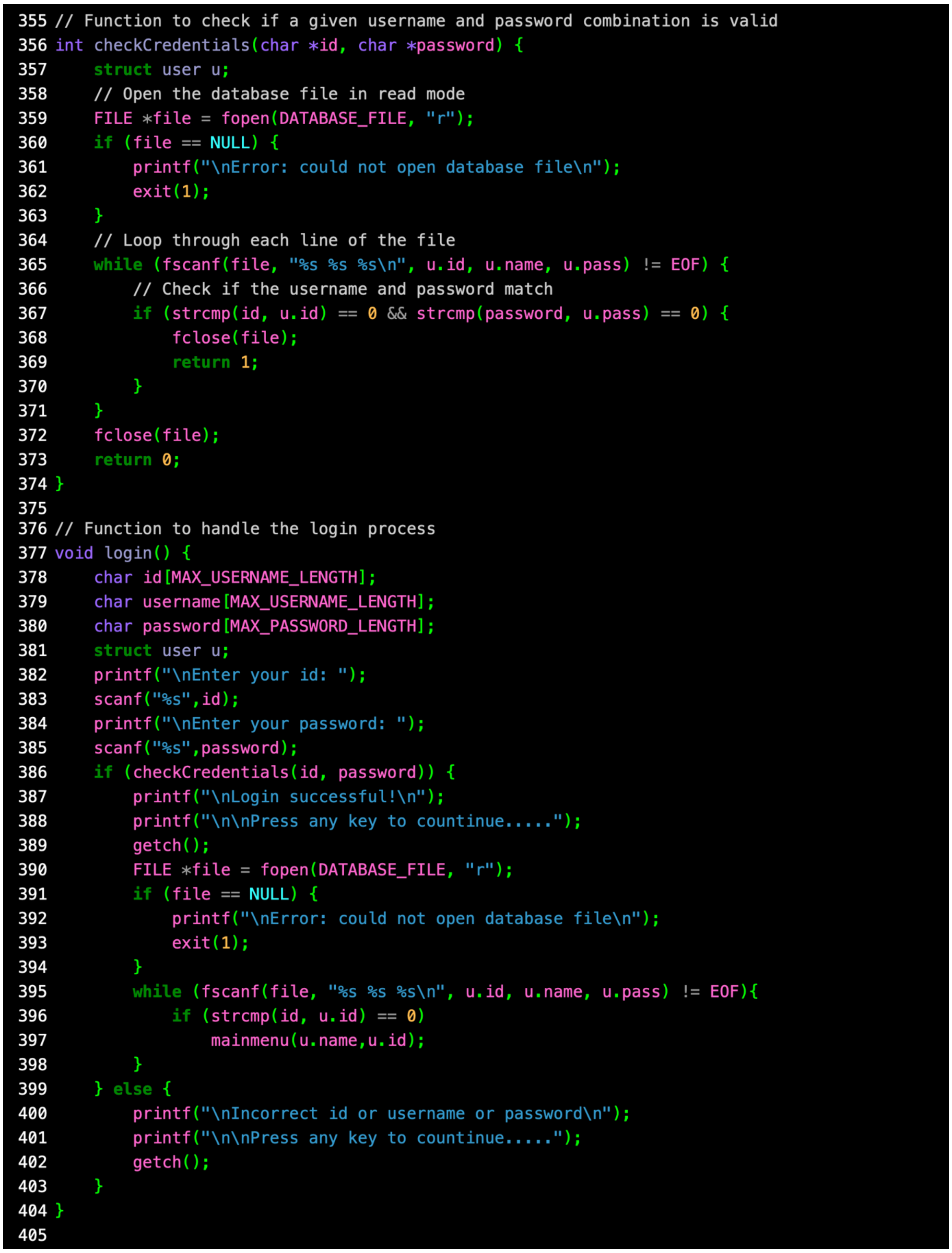
```c
    fclose(fp);
    fclose(fp_temp);
    remove(FILE_NAME);
    rename(TEMP_FILE_NAME, FILE_NAME);
    printf("\n\nPress any key to countinue.....");
    getch();
}

// Function to delete an existing student record by id
void delete_record() {
    display_record_inner();
    FILE *fp, *fp_temp;
    struct student s;
    int modify_id, found = 0;
    fp = fopen(FILE_NAME, "rb");
    fp_temp = fopen(TEMP_FILE_NAME, "wb");
    if (fp == NULL || fp_temp == NULL) {
        printf("Failed to open file.\n");
        return;
    }
    printf("\nEnter student id to delete: ");
    scanf("%d", &modify_id);
    while (fscanf(fp, "%d %s %f %f %f %f\n", &s.id, s.name, &s.att, &s.ca, &s.mte,&s.ete) != EOF) {
        if (s.id == modify_id) {
            found = 1;
            printf("\nRecord Deleted Successfully\n");
            continue;
        } else {
            fprintf(fp_temp, "%d %s %f %f %f %f\n", s.id, s.name, s.att, s.ca, s.mte,s.ete);
            // fprintf(fp_temp, "%d %s %f\n", s.id, s.name, s.ca);
        }
    }
    if (!found) {
        printf("Record not found.\n");
    }
    fclose(fp);
    fclose(fp_temp);
    remove(FILE_NAME);
    rename(TEMP_FILE_NAME, FILE_NAME);
    printf("\n\nPress any key to countinue.....");
    getch();
}

void display_record_inner() {
    FILE *fp;
    struct student s;
    float t=0.0;
    char g[10];
    char *gk = g;
    fp = fopen(FILE_NAME, "rb");
    if(fp == NULL) {
        printf("Error in opening file\n");
        return;
    }

    printf("\nID\t\tName\t\tAttendance Marks\tCA Marks\tMidterm Marks\tEndterm Marks\tGrade\n");
```



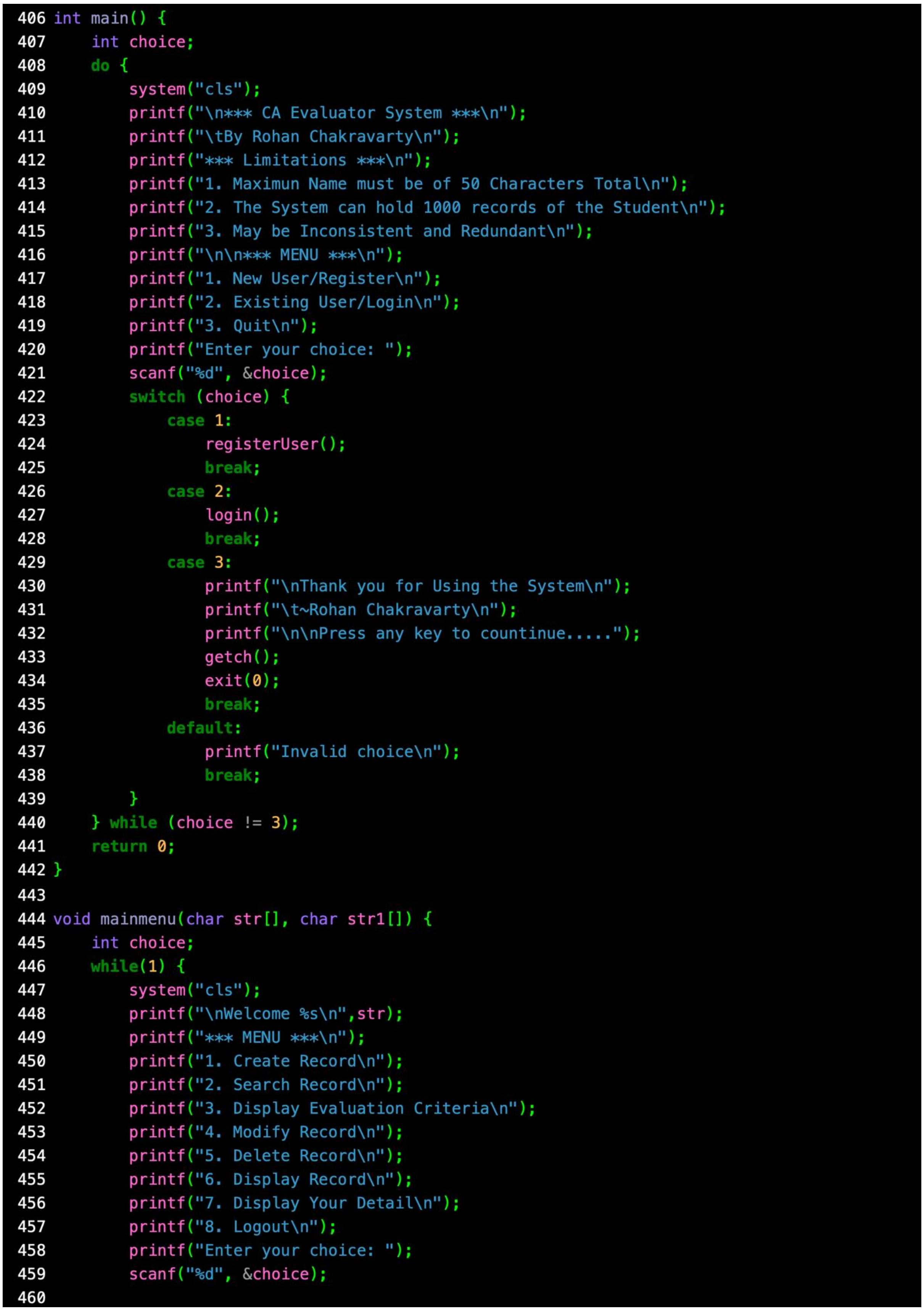
```c

    while (fscanf(fp, "%d %s %f %f %f %f\n", &s.id, s.name, &s.att, &s.ca, &s.mte,&s.ete) != EOF) {
        t=(s.att+s.ca+s.ete+s.mte)/(4.0);
        if(t>=97 && t<=100)
            gk="A+";
        else if (t>=93 && t<=96)
            gk="A";
        else if (t>=90 && t<=92)
            gk="A-";
        else if (t>=87 && t<=89)
            gk="B+";
        else if (t>=83 && t<=86)
            gk="B";
        else if (t>=80 && t<=82)
            gk="B-";
        else if (t>=77 && t<=79)
            gk="C+";
        else if (t>=73 && t<=76)
            gk="C";
        else if (t>=70 && t<=72)
            gk="C-";
        else if (t>=67 && t<=69)
            gk="D+";
        else if (t>=65 && t<=66)
            gk="D";
        else if (t>=60 && t<=64)
            gk="E";
        else if (t<60)
            gk="F";
        printf("%d\t\t%s\t\t%.2f\t\t\t%.2f\t\t%.2f\t\t%.2f\t\t%s\n",s.id, s.name, s.att,s.ca,s.mte,s.ete,gk);
    }
    fclose(fp);
}

// Function to display the record
void display_record() {
    FILE *fp;
    struct student s;
    float t=0.0;
    char g[10];
    char *gk = g;
    fp = fopen(FILE_NAME, "rb");
    if(fp == NULL) {
        printf("Error in opening file\n");
        return;
    }

    printf("\nID\t\tName\t\tAttendance Marks\tCA Marks\tMidterm Marks\tEndterm Marks\tGrade\n");

```



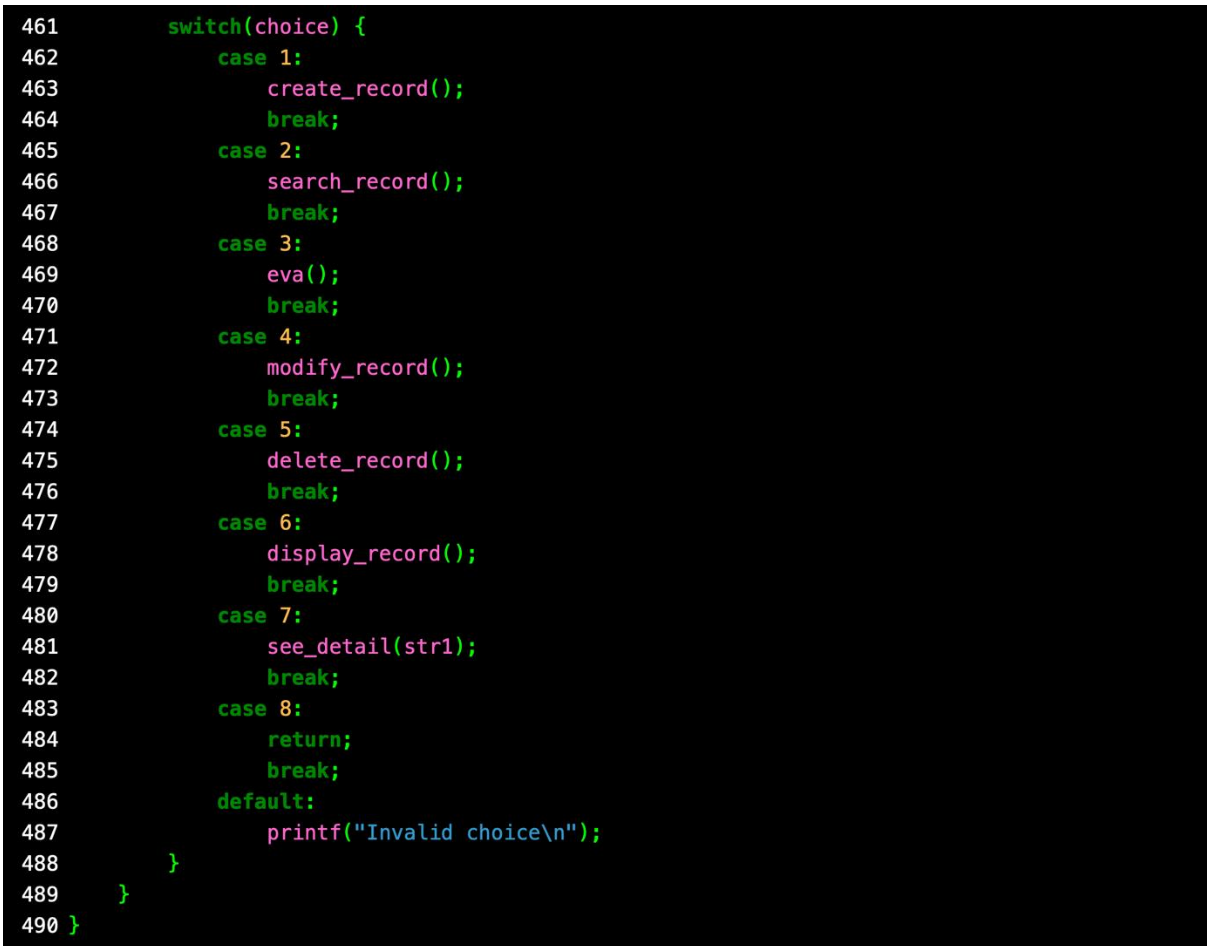
```c
    while (fscanf(fp, "%d %s %f %f %f %f\n", &s.id, s.name, &s.att, &s.ca, &s.mte,&s.ete) != EOF) {
        t=(s.att+s.ca+s.ete+s.mte)/(4.0);
        if(t>=97 && t<=100)
            gk="A+";
        else if (t>=93 && t<=96)
            gk="A";
        else if (t>=90 && t<=92)
            gk="A-";
        else if (t>=87 && t<=89)
            gk="B+";
        else if (t>=83 && t<=86)
            gk="B";
        else if (t>=80 && t<=82)
            gk="B-";
        else if (t>=77 && t<=79)
            gk="C+";
        else if (t>=73 && t<=76)
            gk="C";
        else if (t>=70 && t<=72)
            gk="C-";
        else if (t>=67 && t<=69)
            gk="D+";
        else if (t>=65 && t<=66)
            gk="D";
        else if (t>=60 && t<=64)
            gk="E";
        else if (t<60)
            gk="F";
        printf("%d\t\t%s\t\t%.2f\t\t\t%.2f\t\t%.2f\t\t%.2f\t\t%s\n",s.id, s.name, s.att,s.ca,s.mte,s.ete,gk);
    }
    fclose(fp);
    printf("\n\nPress any key to countinue.....");
    getch();
}

// Function to register a new user
void registerUser() {

    char username[MAX_USERNAME_LENGTH];
    char password[MAX_PASSWORD_LENGTH];
    FILE *file;
    file = fopen(DATABASE_FILE, "ab");
    struct user u;
    int found = 0;
    if (file == NULL) {
        printf("Failed to open file.\n");
        return;
    }
    // printf("\nEnter Id: ");
    // scanf("%d", &id);
    printf("\nEnter your name (max %d characters): ", MAX_USERNAME_LENGTH);
    scanf("%s", username);
    printf("\nEnter a password (max %d characters): ", MAX_PASSWORD_LENGTH);
    scanf("%s",password);
    int num=random_id();
    char str[MAX_USERNAME_LENGTH];
    sprintf(str, "%d", num);
    strcat(str,"@");
    strcat(str,username);
    fprintf(file,"%s %s %s\n",str,username,password);
    printf("Your Username: %s\n",str);
    printf("\nRegistration successful!\n");
    printf("\n\nPress any key to countinue.....");
    getch();
    fclose(file);
}

```



```c
// Function to check if a given username and password combination is valid
int checkCredentials(char *id, char *password) {
    struct user u;
    // Open the database file in read mode
    FILE *file = fopen(DATABASE_FILE, "r");
    if (file == NULL) {
        printf("\nError: could not open database file\n");
        exit(1);
    }
    // Loop through each line of the file
    while (fscanf(file, "%s %s %s\n", u.id, u.name, u.pass) != EOF) {
        // Check if the username and password match
        if (strcmp(id, u.id) == 0 && strcmp(password, u.pass) == 0) {
            fclose(file);
            return 1;
        }
    }
    fclose(file);
    return 0;
}

// Function to handle the login process
void login() {
    char id[MAX_USERNAME_LENGTH];
    char username[MAX_USERNAME_LENGTH];
    char password[MAX_PASSWORD_LENGTH];
    struct user u;
    printf("\nEnter your id: ");
    scanf("%s",id);
    printf("\nEnter your password: ");
    scanf("%s",password);
    if (checkCredentials(id, password)) {
        printf("\nLogin successful!\n");
        printf("\n\nPress any key to countinue.....");
        getch();
        FILE *file = fopen(DATABASE_FILE, "r");
        if (file == NULL) {
            printf("\nError: could not open database file\n");
            exit(1);
        }
        while (fscanf(file, "%s %s %s\n", u.id, u.name, u.pass) != EOF){
            if (strcmp(id, u.id) == 0)
                mainmenu(u.name,u.id);
        }
    } else {
        printf("\nIncorrect id or username or password\n");
        printf("\n\nPress any key to countinue.....");
        getch();
    }
}
```

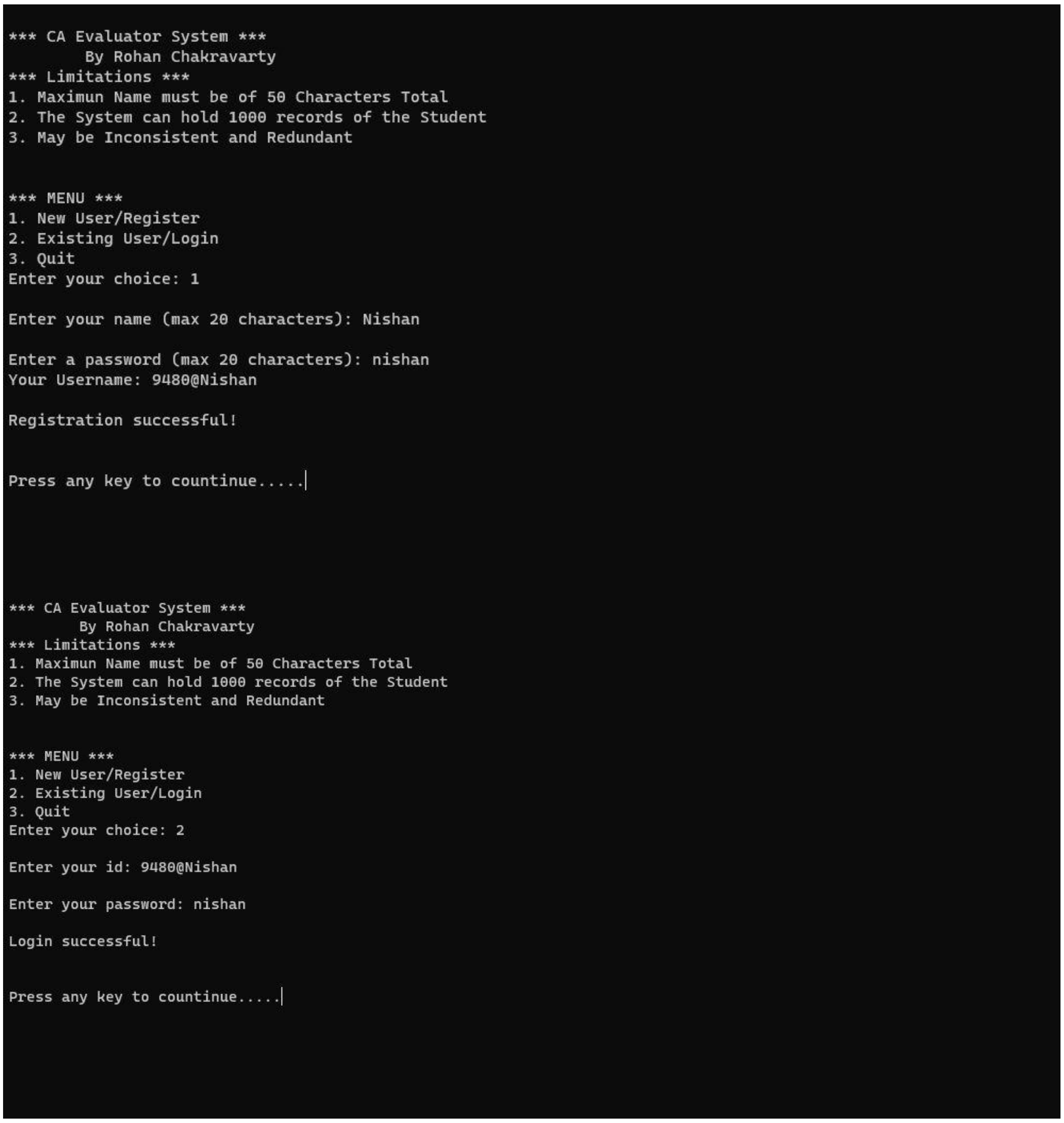


```c
int main() {
    int choice;
    do {
        system("cls");
        printf("\n*** CA Evaluator System ***\n");
        printf("\tBy Rohan Chakravarty\n");
        printf("*** Limitations ***\n");
        printf("1. Maximun Name must be of 50 Characters Total\n");
        printf("2. The System can hold 1000 records of the Student\n");
        printf("3. May be Inconsistent and Redundant\n");
        printf("\n\n*** MENU ***\n");
        printf("1. New User/Register\n");
        printf("2. Existing User/Login\n");
        printf("3. Quit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                registerUser();
                break;
            case 2:
                login();
                break;
            case 3:
                printf("\nThank you for Using the System\n");
                printf("\t~Rohan Chakravarty\n");
                printf("\n\nPress any key to countinue.....");
                getch();
                exit(0);
                break;
            default:
                printf("Invalid choice\n");
                break;
        }
    } while (choice != 3);
    return 0;
}

void mainmenu(char str[], char str1[]) {
    int choice;
    while(1) {
        system("cls");
        printf("\nWelcome %s\n",str);
        printf("*** MENU ***\n");
        printf("1. Create Record\n");
        printf("2. Search Record\n");
        printf("3. Display Evaluation Criteria\n");
        printf("4. Modify Record\n");
        printf("5. Delete Record\n");
        printf("6. Display Record\n");
        printf("7. Display Your Detail\n");
        printf("8. Logout\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

```



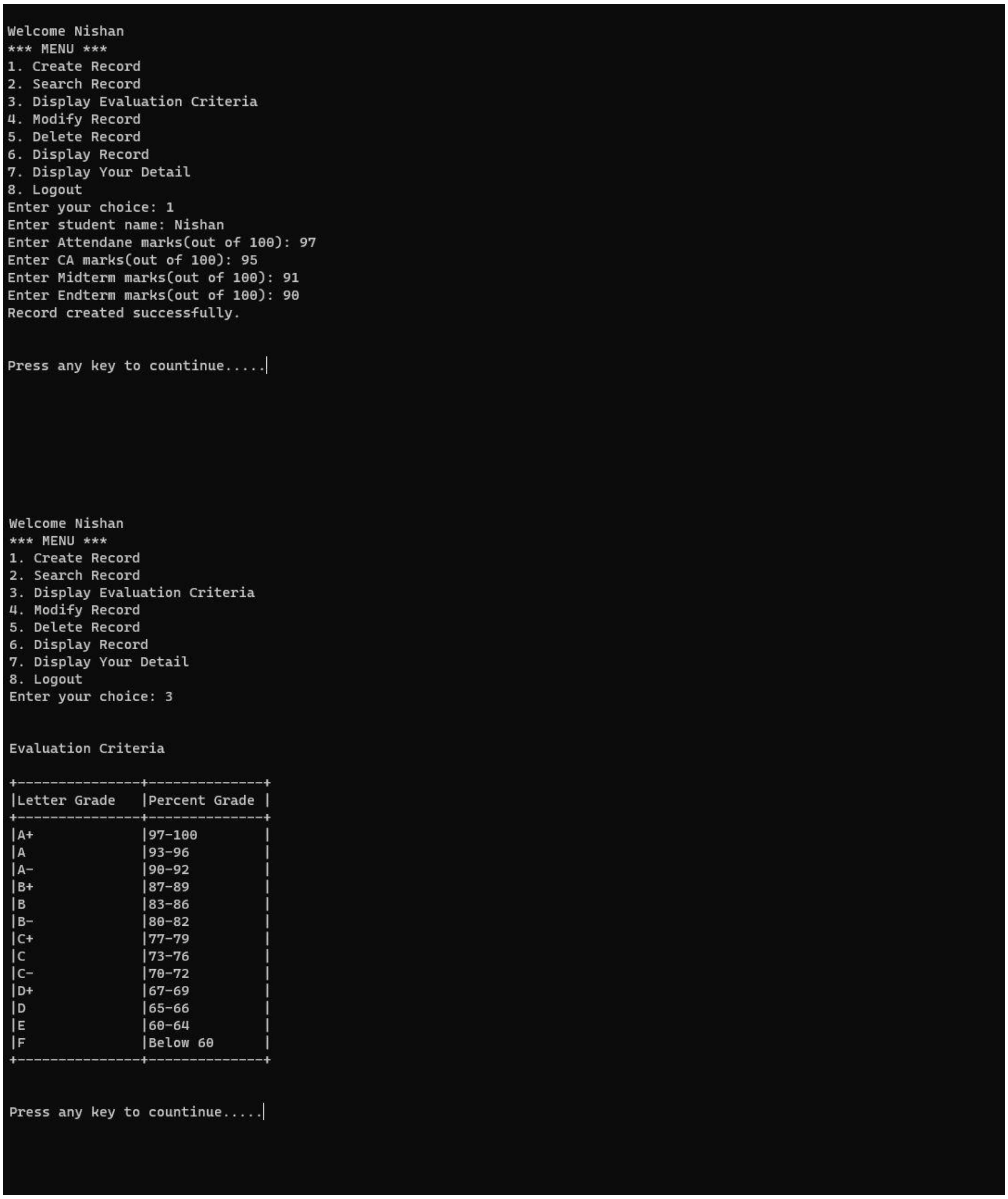
```c
        switch(choice) {
            case 1:
                create_record();
                break;
            case 2:
                search_record();
                break;
            case 3:
                eva();
                break;
            case 4:
                modify_record();
                break;
            case 5:
                delete_record();
                break;
            case 6:
                display_record();
                break;
            case 7:
                see_detail(str1);
                break;
            case 8:
                return;
                break;
            default:
                printf("Invalid choice\n");
        }
    }
}
```

OUTPUT SNAPSHOT:



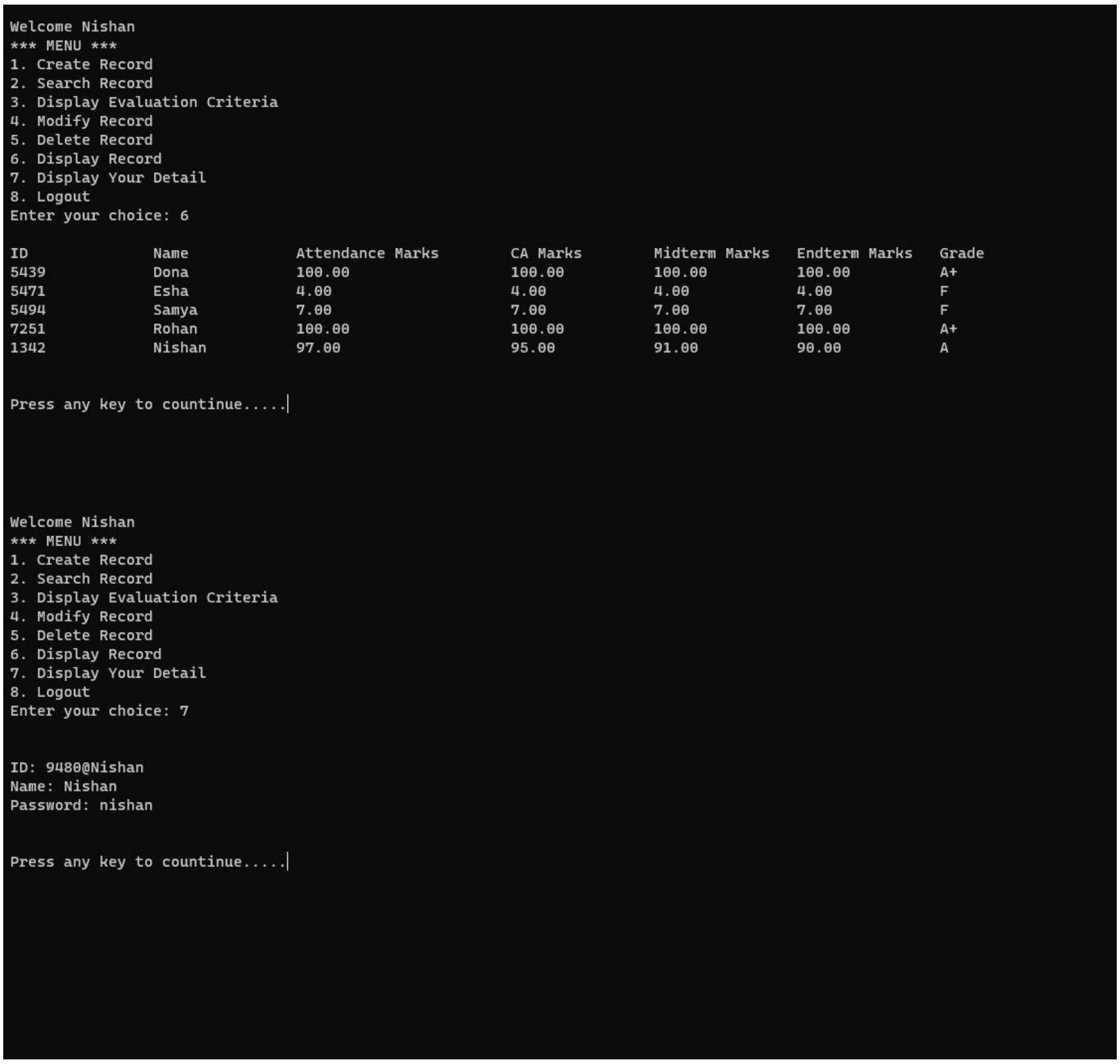
```
*** CA Evaluator System ***
        By Rohan Chakravarty
*** Limitations ***
1. Maximun Name must be of 50 Characters Total
2. The System can hold 1000 records of the Student
3. May be Inconsistent and Redundant


*** MENU ***
1. New User/Register
2. Existing User/Login
3. Quit
Enter your choice: 1

Enter your name (max 20 characters): Nishan

Enter a password (max 20 characters): nishan
Your Username: 9480@Nishan

Registration successful!


Press any key to countinue.....
```

```
*** CA Evaluator System ***
        By Rohan Chakravarty
*** Limitations ***
1. Maximun Name must be of 50 Characters Total
2. The System can hold 1000 records of the Student
3. May be Inconsistent and Redundant


*** MENU ***
1. New User/Register
2. Existing User/Login
3. Quit
Enter your choice: 2

Enter your id: 9480@Nishan

Enter your password: nishan

Login successful!


Press any key to countinue.....
```



```
Welcome Nishan
*** MENU ***
1. Create Record
2. Search Record
3. Display Evaluation Criteria
4. Modify Record
5. Delete Record
6. Display Record
7. Display Your Detail
8. Logout
Enter your choice: 1
Enter student name: Nishan
Enter Attendane marks(out of 100): 97
Enter CA marks(out of 100): 95
Enter Midterm marks(out of 100): 91
Enter Endterm marks(out of 100): 90
Record created successfully.


Press any key to countinue.....
```

```
Welcome Nishan
*** MENU ***
1. Create Record
2. Search Record
3. Display Evaluation Criteria
4. Modify Record
5. Delete Record
6. Display Record
7. Display Your Detail
8. Logout
Enter your choice: 3


Evaluation Criteria

+---------------+--------------+
|Letter Grade   |Percent Grade |
+---------------+--------------+
|A+             |97-100        |
|A              |93-96         |
|A-             |90-92         |
|B+             |87-89         |
|B              |83-86         |
|B-             |80-82         |
|C+             |77-79         |
|C              |73-76         |
|C-             |70-72         |
|D+             |67-69         |
|D              |65-66         |
|E              |60-64         |
|F              |Below 60      |
+---------------+--------------+


Press any key to countinue.....
```



```
Welcome Nishan
*** MENU ***
1. Create Record
2. Search Record
3. Display Evaluation Criteria
4. Modify Record
5. Delete Record
6. Display Record
7. Display Your Detail
8. Logout
Enter your choice: 6

ID              Name            Attendance Marks        CA Marks        Midterm Marks   Endterm Marks   Grade
5439            Dona            100.00                  100.00          100.00          100.00          A+
5471            Esha            4.00                    4.00            4.00            4.00            F
5494            Samya           7.00                    7.00            7.00            7.00            F
7251            Rohan           100.00                  100.00          100.00          100.00          A+
1342            Nishan          97.00                   95.00           91.00           90.00           A


Press any key to countinue.....






Welcome Nishan
*** MENU ***
1. Create Record
2. Search Record
3. Display Evaluation Criteria
4. Modify Record
5. Delete Record
6. Display Record
7. Display Your Detail
8. Logout
Enter your choice: 7


ID: 9480@Nishan
Name: Nishan
Password: nishan


Press any key to countinue.....
```