PURBANCHAL UNIVERSITY

Faculty Of Science & Technology

Biratnagar, Nepal

Project report in C Programming On

"STUDENT MANAGEMENT SYSTEM"

A project report in patrial fulfilment of the requirements for

BACHELOR IN INFORMATION TECHNOLOGY

[BIT] 1ST SEMESTER

OF

PURBANCHAL UNIVERSITY

BIRATNAGAR, NEPAL

SUBMITTED BY

SN	Name	Registration Number
1	Bigyan Shrestha	058-3-2-05271-2022
2	Dinesh Bajagain	058-3-2-05277-2022
3	Prajwal Bhattarai	058-3-2-05292-2022
4	Sandesh Khadka	058-3-2-05299-2022

College Letter Head

TO Whom It May Concern

I am writing to bring to your attention a significant academic project undertaken by students of our esteemed institution. This endeavour, titled "Student Management System," was completed as a part of the curriculum for the bachelor's in information technology (BIT) program during the 1st semester.

The project **"Student Management System"** was successfully developed and executed by the following diligent students:

Mr. Bigyan Shrestha

Mr. Dinesh Bajagain

Mr. Prajwal Bhattarai

Mr. Sandesh Khadka

Under the guidance and supervision in accordance with the guidelines provided by KIST college, the students diligently completed this project. I hereby certify that the work presented in the "Student Management System" project is original and has not been submitted or presented in any form as a part of any other academic work, thereby adhering to the highest standards of academic integrity.

The project was undertaken as part of the subject "Project-I" during the 1st semester of the BIT program. The successful completion of this project is a testament to the students' dedication, technical skills, and their commitment to academic excellence.

I kindly request that you recognize and acknowledge the hard work and dedication of these students in completing this project successfully. Their effort reflects positively on the quality of education provided at **KIST College** and highlights our commitment to producing competent and skilled graduates.

Thank you for your attention to this matter. If you require any additional information or have any questions, please do not hesitate to contact me.

Deepak Khadka

Program Co-ordinator, BIT

KIST College of Informational Technology

Examiner's Certification

This is to certify that the project entitled "Student Management System" has been successfully completed by Mr. Bigyan Shrestha, Mr. Dinesh Bajagain, Mr. Prajwal Bhattarai, and Mr. Sandesh Khadka in partial fulfilment of the Degree of Bachelor of Information Technology (BIT) at KIST College during the academic year 2022.

The project was conducted in accordance with the curriculum of the 1st semester of the BIT program, and it adhered to the academic guidelines set forth by the college. The students completed this project under the expert guidance of Mr. Deepak Khadka, who provided invaluable support and supervision throughout the project's development and execution.

This certification acknowledges the successful completion of the "Student Management System" project, which stands as a testament to the students' dedication and their ability to apply their knowledge and skills in a practical setting. The project's originality and compliance with academic integrity standards have been duly verified.

We commend the students for their outstanding effort and the quality of their work, which reflects positively on both the students and the academic programs offered at KIST College of Information and Technology.

Department of Information and Technology

KIST College of Information and Technology

PURBANCHAL UNIVERSITY

ACKNOWLEDGEMENT

It is with greatest satisfaction and euphoria that we are submitting our project report entitled "**STUDENT MANAGEMENT SYSTEM**". We have completed it as a part of the curriculum of **PURBANCHAL UNIVERSITY.**

We also take this opportunity to express a deep sense of gratefulness to our BIT Coordinator Mr. Deepak Khadka for his amiable support, valuable information and guidance which helped us in completing this task throughout its various stages. We are indebted to all members of KIST College, for the valuable support and suggestion provided by them using their specific fields' knowledge. We are grateful for their cooperation during the period of our project.

Finally, we would also like to express our gratefulness towards **PURBANCHAL UNIVERSITY** for designing such a wonderful course structure. It will help us to get more knowledge in the field of Information Technology & help us to have a bright future in the field of technology.

We hope our university will accept this attempt as a successful project.

Last but not the least, our sincere thanks to our parents, teaching and non-teaching staffs of our college and friends.

ABSTRACT

The Student Management System (SMS) is a software solution designed to enhance the administrative processes of educational institutions. This project aims to create a user-friendly system that simplifies the management of student data including their name, email, contact number, grade, and their total fees and remaining dues.

The SMS project addresses the challenges faced by educational institutions in managing a growing student population, ensuring data accuracy, and improving overall operational efficiency. This system provides a centralized platform where administrators, and students can interact seamlessly, allowing for better communication and data-driven decision-making.

By implementing the Student Management System, educational institutions can significantly improve their administrative efficiency, enhance the educational experience for students, and foster better communication among all stakeholders. This project aims to contribute to the ongoing advancement of educational management practices and ultimately facilitate the pursuit of academic excellence.

Table Of Content

Table of Contents

Table of Figures

1.INTRODUCTION

The Student Management System is a software application developed in C programming language to assist educational institution in efficiently managing student-related information(data). This document provides an overview of the system's purpose, scope, and the technologies used. In the digital age, technology has revolutionized every aspect of our lives, including the way educational institution operates. Therefore, the Student Management System serves as a sophisticated tool that streamlines the management of the student-related information, facilitating administrative tasks and fostering a more efficient educational environment.

2.STATEMENT OF THE PROBLEM

Without too many distinctions, the problem faced by the educational institution without the student management software is inevitable. Some of the problems are:

- Manual Data Entry and Record Keeping
- No backup if the record book is lost
- Time consuming
- Challenges in Timely Communication
- Data Security and Privacy Risks

Student management system is ideal solution for overcoming these complexities.

3.PURPOSE & SCOPE

The primary purpose of Student Management System is to simplify and automate the management of student records and information within educational institutions. By leveraging the power of C programming, the system provides a seamless experience for administrators to manage student data accurately and efficiently.

The system's scope encompasses broad functionalities, including user authentication, student addition, listing student details, deletion of student details, and so on. These features collectively contribute to a comprehensive solution that addresses the need of educational institution to manage, access, and utilizes data effectively.

4.TECHNOLOGY USED

The Student Management System is built using C programming language, a powerful and widely used programming language known for its efficiency and versatility. This language choice allows for precise control over memory management and system resources, crucial aspects when handling sensitive student information. The use of C language also ensures the system's compatibility across various platforms and operating systems, making it accessible to a wider user base.

5.SYSTEM FEATURE

5.1 Features

- User Authentication
- Student Addition
- Student Listing
- Searching Student Details
- Student Details Modification
- Student Deletion

5.2 Functionalities

- Add, view, modify, and delete student's records.
- Generate information like details of the student.
- Secure login system to protect data from unauthorized access.
- Simple and intuitive Command-line interface for the ease of use.

6.INSTALLATION

1. Clone the project repository from GitHub.
2. Navigate to the project directory using the terminal.
3. Compile the main source file using any C compiler.
4. Run the compiled executable.

https://github.com/sandeshkhadka10/Student-Management-System

7.USAGE

7.1 Logging In

1. Launch the software.
2. Enter the login to access the system.

7.2 Dashboard

1. The interface will show the various options in the main menu.

7.3 Adding a student

1. Choose the "Add Student" option.
2. Enter the student details as prompted.
3. Confirm addition.

7.4 Viewing Student Details

1. Select the "Display All Students" option from the main menu.
2. Enter the ID NUMBER of the specific student.
3. This will display the details of the specified student.

7.5 Modifying Student Data

1. Select the "Edit Student Details" option from the main menu.
2. Enter the ID NUMBER of the student.
3. Update the necessary information.

7.6 Deleting an information of Student

1. choose "Delete Student" option from the main menu.
2. Enter the ID NUMBER of the student you want to delete.
3. Confirm the deletion.

8.FUTURE ENHANCEMENTS

- Enhanced authentication mechanisms.
- Support for multiple user roles.
- Graphical User Interface (GUI) implementation.
- Integration of a database system for better performance.

9.BENEFITS OF THE SOFTWARE

- Time Efficient
- Accuracy
- Data Accuracy
- Accessibility
- User-Friendly Interface

10.SYSTEM DESIGN

10.1 Algorithm

Login and Menu page:

Step 1: Start.

Step 2: Enter username and password. If username and password is incorrect, go to step 1.

Step 3: Menu is displayed. Ask user to enter choice. Choices are:

a) Add student.
b) Display all students.
c) Search student.
d) Edit student details.
e) Delete student details.
f) Exit.

Case 1: If the choice is a, or b, or c, or d, or e, the program jumps to algorithm B, or C, or D, or E or F respectively. While entering the option, only lowercase alphabets(a-f) are allowed.

Case 2: If the choice is f or any other character, go to algorithm F.

Step 4: Asks user to press any key to continue. Go to step 3.

Step 5: Stop.

A. Add students:

Step 1: Start.

Step 2: Ask user to enter name, id, course and semester, address, email, phone number, GPA and amount paid.

Step 3: Due amount is calculated as: due amount= total amount – paid amount.

Step 4: All the details are written in a file.

Step 5: Asks user whether to add another record or not. Press Y or y for YES and any other character for NO.

i. If Y or y is entered, go to step 2.

ii. If any other character is entered, go to step 4 of login and menu page.

Step 6: Stop.

B. Display all students:

Step 1: Start.

Step 2: The program reads all the details from the file and display them, if any records are present. But if, there are no records available, "No records found" is displayed.

Step 3: Go to step 4 of login and menu page.

Step 4: Stop.

C. Search students:

Step 1: Start.

Step 2: Enter the id of the student you want to search.

Step 3: The program searches that id and the display the details if such id is found otherwise "record not found" is displays.

Step 4: Go to step 4 of login and menu page.
Step 5: Stop.

D. Edit student details:

Step 1: Start.
Step 2: Ask user to enter the id to edit.
Step 3: If the id is not found, "student with such Id Number not found" is displayed. Go to step 4 of login and menu page.
Step 4: A menu is displayed. Ask user to enter choice. Choices are:

1) Name
2) ID
3) Semester
4) Address
5) Email
6) Phone number
7) GPA
8) Paid amount
9) Exit

Case 1: If the choice is 1 or, 2 or, 3 or, 4 or, 5 or, 6 or, 7 or, 8, the value of the respective choice is overwritten by the recent value. And due amount = due amount – paid amount. Go to step 4.
Case 2: If the choice is 9:

a) If the details are updated, "student details updated successfully" is displayed.
b) If the details are not updated, "No updates were made" is displayed.

Go to step 4 of login and menu page.
Case 3: If any other character is entered, "Invalid choice", "Press any key to retry" are displayed. Go to step 4.
Step 5: Go to step 4 of login and menu page.
Step 6: Stop.

E. Delete student:

Step 1: Start.
Step 2: Ask user to enter id of student whose details is to be deleted.
Step 3: If the id is found, the program copies the details of all the students except the student matching the entered id, in temporary file. The old file is then removed and the temporary file is renamed to the name of that old file otherwise, "student with such id number is not found" is displayed.
Step 4: Go to step 4 of login and menu page.

F. Exit:

Step 1: start.
Step 2: if the user chooses to enter the option f, display a message asking the user to confirm the exit.
Step 3: Accept user input for confirmation.
Step 4: If the user confirms the exit by entering 'Y' or 'y':

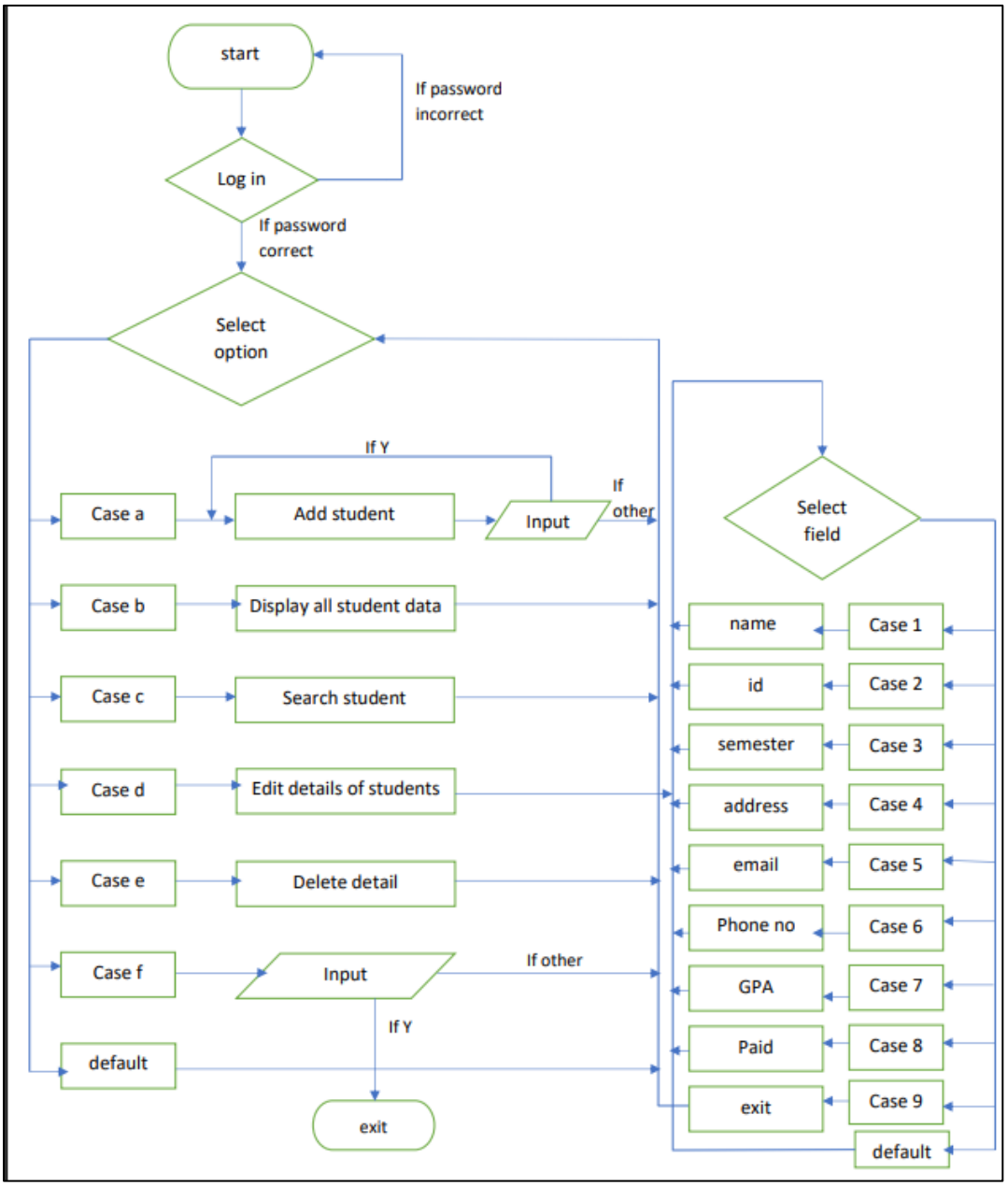
a. Exit the program using the exit (0) function. This function will terminate the program.

Step 5: Else, if the user enters any other character:

a. Return to the main menu, allowing the user to continue using the program.

Step 6: Stop

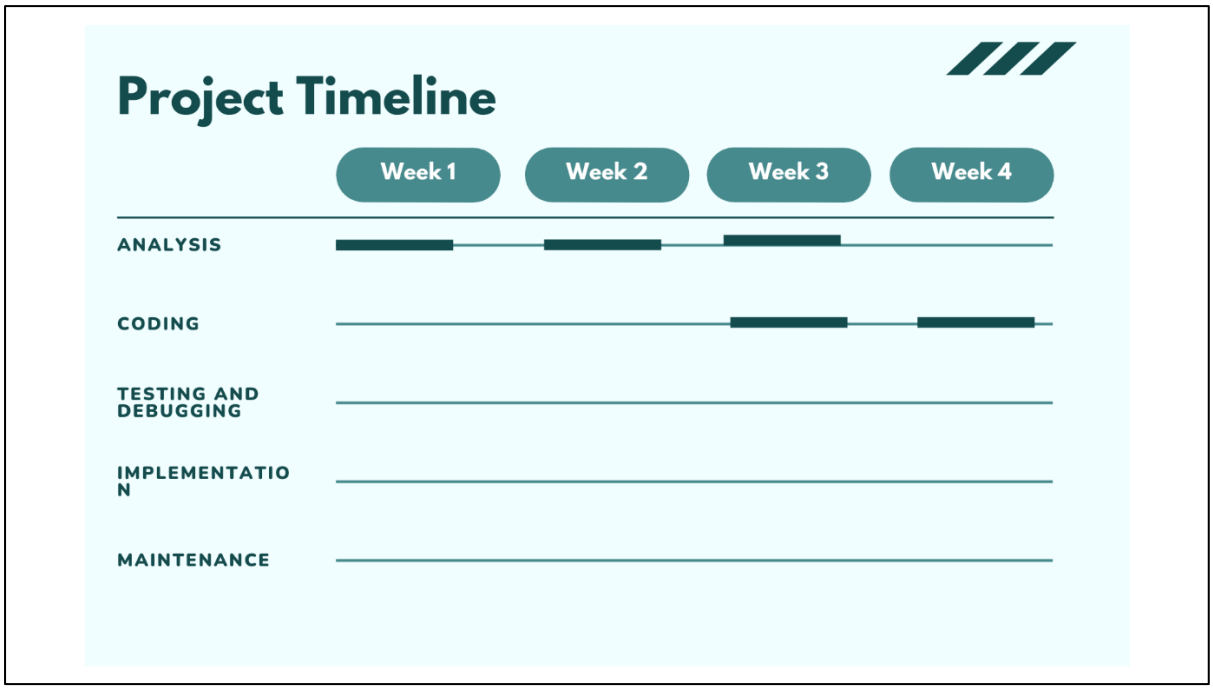
10.2 Flowchart



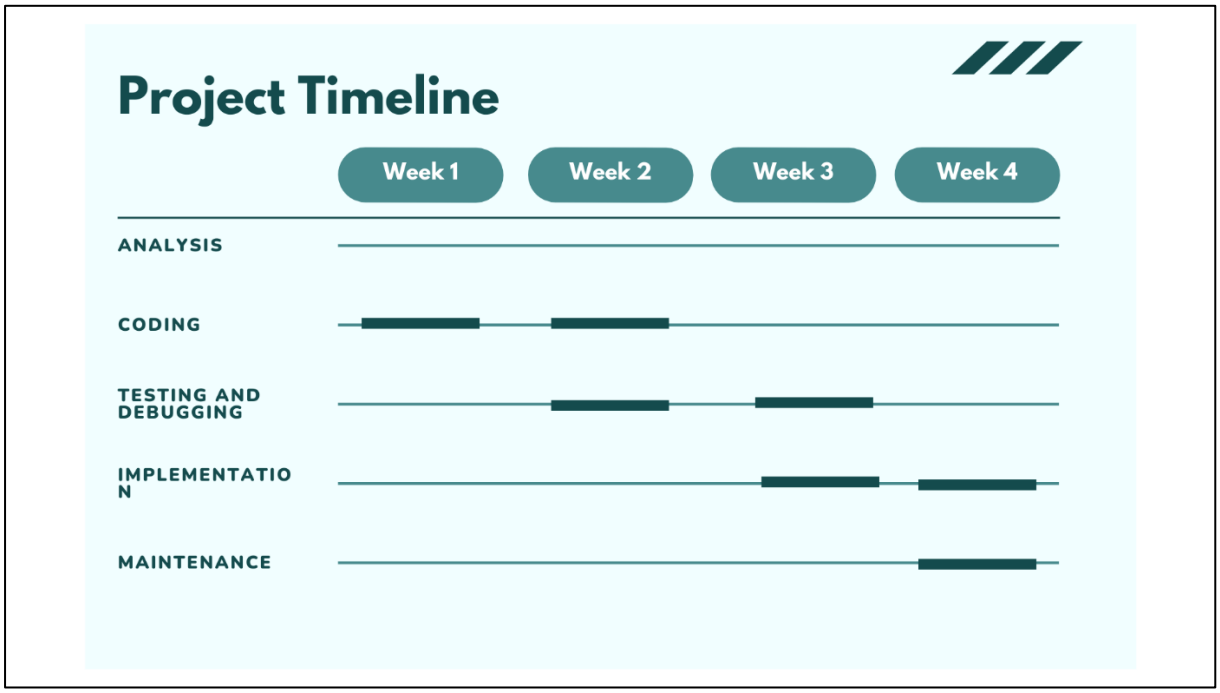
11.GANTT CHART

This project required about 2 months to be completed. All time period required for this project for different task are shown in graph below:

1st month



2nd month



12.FUNCTIONAL ANALYSIS

SN	Function Module	Function Description
1.	**Login ()**	This function is for security purpose so that person other than admin cannot manipulate the system or program.
2.	**Main menu ()**	This function displays all the available features of the software related to students.
3.	**AddStudent()**	This function is for adding the record of the student in the system.
4.	**DisplayAllStudents()**	This function shows the details of all the students that is recorded in the software.
5.	**SearchStudent**()	This function search for the detail of the specific student using the ID number of the student.
6.	**EditStudentDetails()**	This function is to edit the details of the specific student using the ID number of the student.
7.	**DeleteStudent()**	This function completely deletes the record of the specific student with the help of ID number from the system, if found.
8.	**Exit ()**	This function completely terminates and close the program.

13.SYSTEM REQUIREMENTS

Following hardware and software requirement should be met for flawless running of this system:

RAM: Generally, a minimum of 32 MB is recommended for the better function.

Hard Disk: The minimum space capacity should be around 50 MB.

Processor: PC with Pentium II Processor (260 MHz) is recommended.

OS: Any OS that support the SMS. (Windows, Linux).

Applications: Dev C++ or another C compiler.

14.CONCLUSION

In conclusion, the Student Management System built using C programming language represents a significant advancement in the realm of educational technology. By harnessing the power of C, the system efficiently manages student information, empowering educational institutions to operate with greater precision, security, and convenience. As technology continues to evolve, the Student Management System serves as a testament to the profound impact programming languages like C can have an optimizing administrative process within the education sector. In essence, the Student Management System, developed using the C programming language, is a game-changer in the world of education. Think of it as a highly organized digital assistant for schools and colleges. It does an incredible job of handling student data, making things run smoothly and securely for educational institutions. This system is a shining example of how using programming languages like C can revolutionize the way schools manage their operations, making everything simpler and more efficient. It's like upgrading from an old, slow computer to a super-fast, state-of-the-art one.

15.APPENDIX

15.1 SNAPSHOTS:

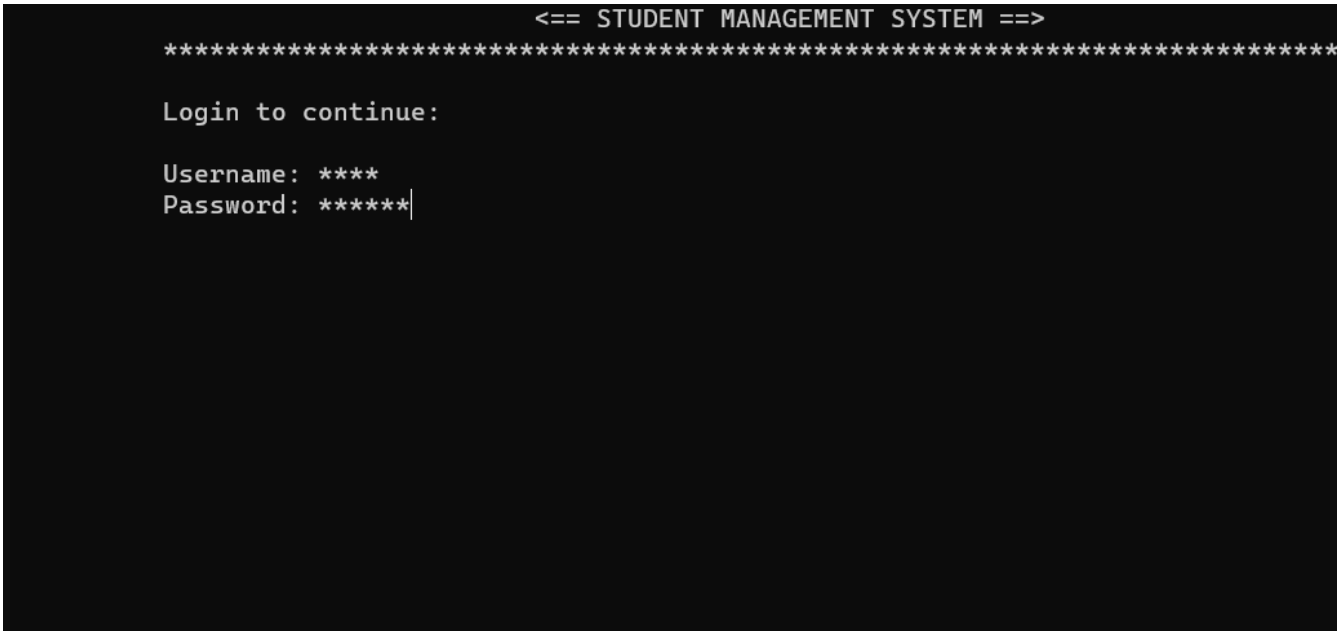


Figure 1 Login Page

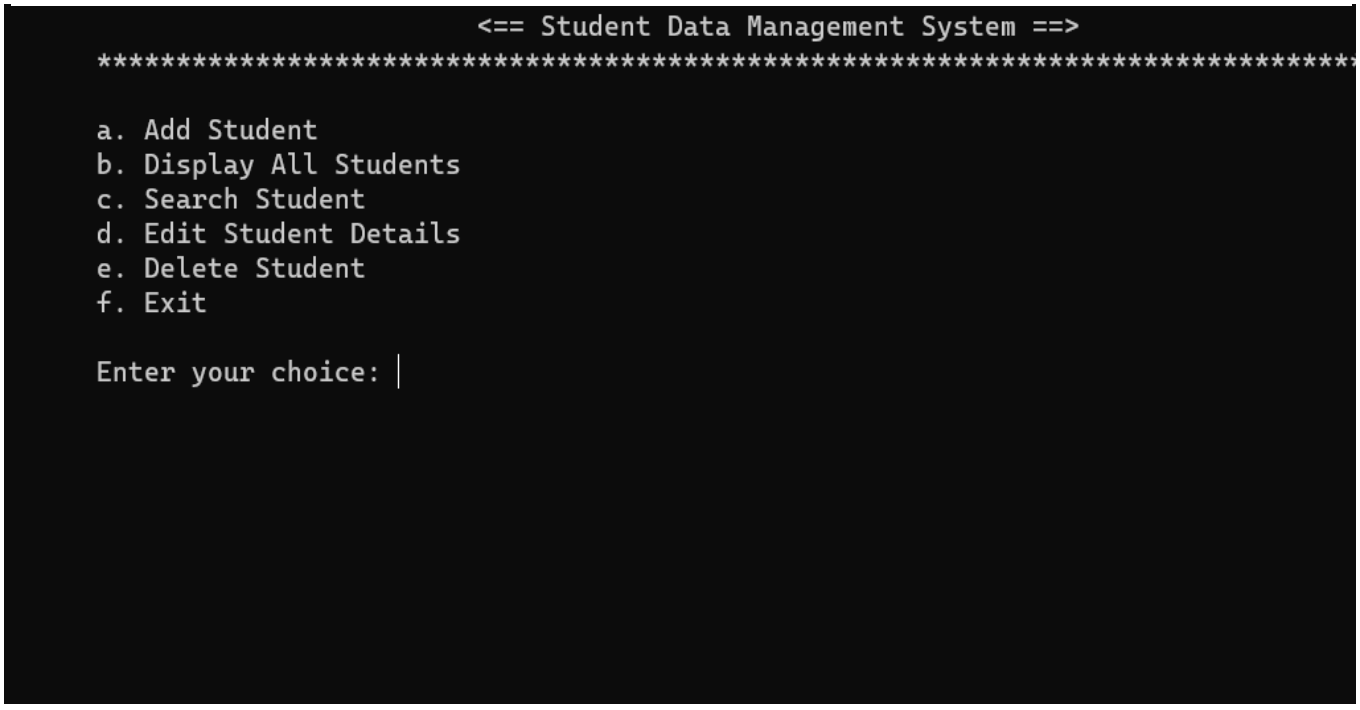


Figure 2 Main Menu

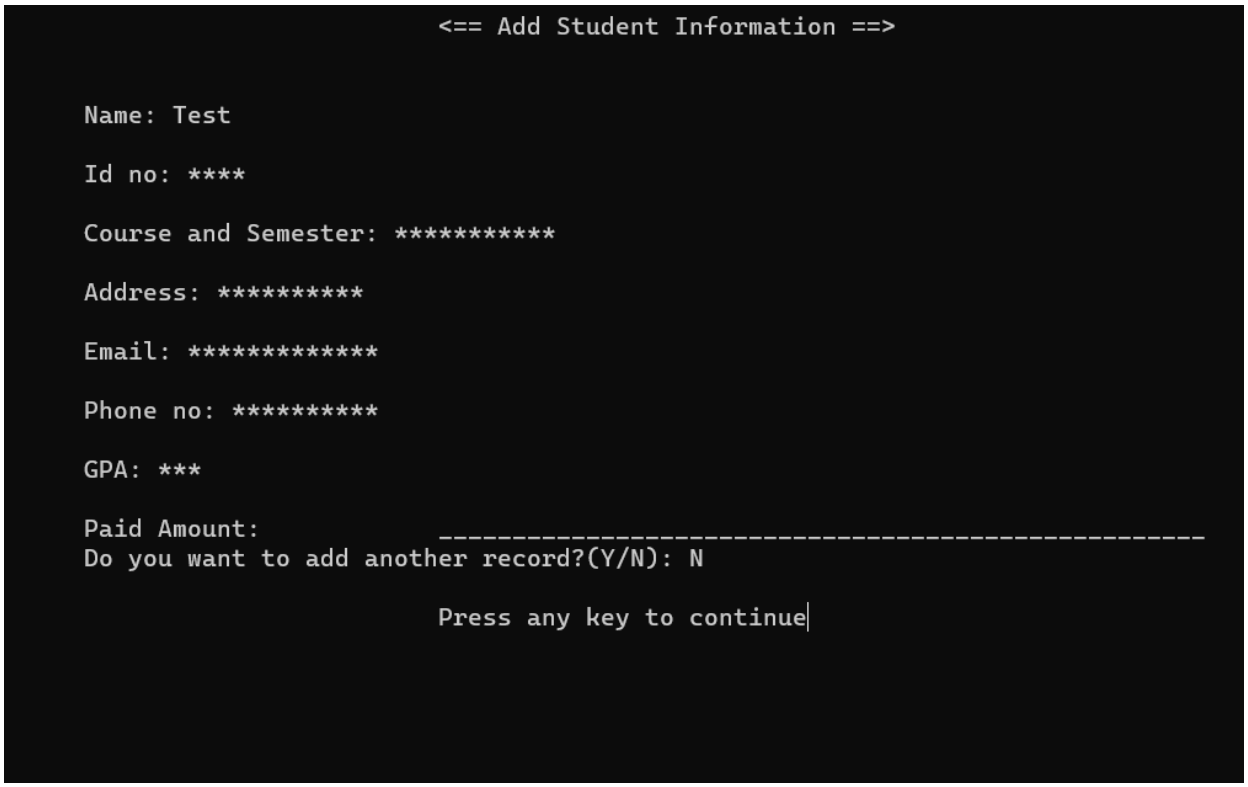


Figure 4 Add Student

Figure 3 Displaying Student Details

```
                        <== Search Student ==>

    Enter the id you want to search: 1111
```

```
                        <== Search Student ==>

     Enter the id you want to search: 1111

     Student Name: sample
     Id: 1111
     Class and Semester: *******
     Address: ******
     Email: sample@gmail.com
     Phone number: *********
     GPA: 0.00
     Total Amount: 790000.00
      Paid Amount: 0.00
     Due Amount: 790000.00
     ----------------------------------
                        Press any key to continue
```

Figure 5 Searching the Students

```
                        <== Edit Student Details ==>

  Enter ID to Edit: 1111
```

```
                        <== Edit Student Details ==>

Select fields to update:
1. Name
2. ID
3. Semester
4. Address
5. Email
6. Phone Number
7. GPA
8. Paid Amount
9. Exit
Enter your choice (1-9): 1

Enter Updated Name: root
```

```
                        <== Search Student ==>


 Enter the ID you want to search: 1111

 Student Name         : root
 ID                   : 1111
 Class and Semester   : *******
 Address              : *******
 Email                : ************
 Phone number         : **********
 GPA                  : 0.00
 Total Amount         : Rs 790000.00
 Paid Amount          : Rs 0.00
 Due Amount           : Rs 790000.00
 ------------------------------------
 Press any key to continue
```

Figure 6 Editing Student Details

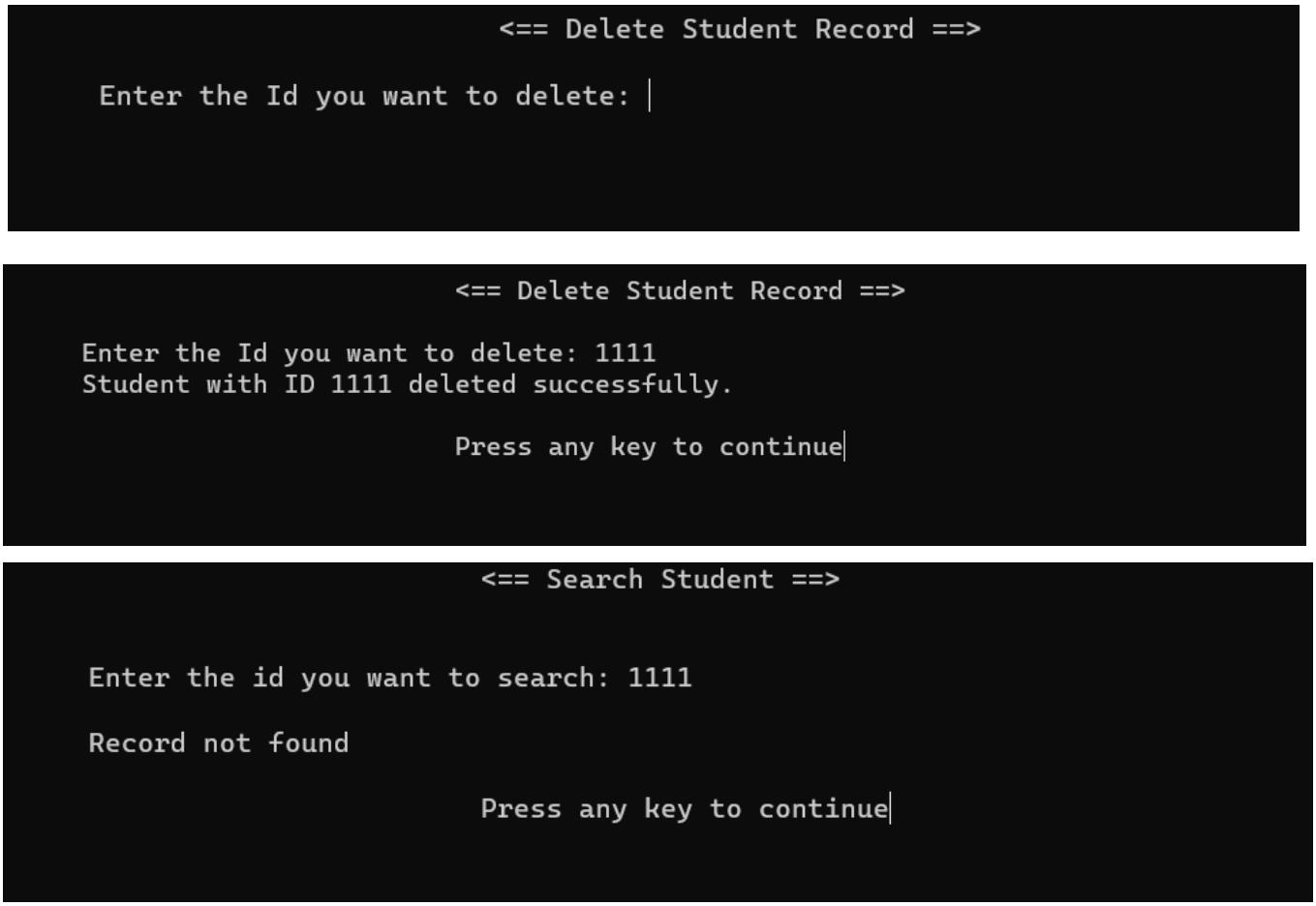


Figure 7 Deleting Student Details

```
                              <== Student Management System ==>
        ********************************************************************************

        a. Add Student
        b. Display All Students
        c. Search Student
        d. Edit Student Details
        e. Delete Student
        f. Exit

        Enter your choice: f
        Press (Y/y) to exit and (N/n) to cancel:
```

```
                              <== Student Management System ==>
      ********************************************************************************

      a. Add Student
      b. Display All Students
      c. Search Student
      d. Edit Student Details
      e. Delete Student
      f. Exit

      Enter your choice: f
      Press (Y/y) to exit and (N/n) to cancel: n
      Please read the option clearly
```

```
                                    <== Student Management System ==>
                ********************************************************************************

                a. Add Student
                b. Display All Students
                c. Search Student
                d. Edit Student Details
                e. Delete Student
                f. Exit

                Enter your choice: f
                Press (Y/y) to exit and (N/n) to cancel: y

--------------------------------
Process exited after 75.6 seconds with return value 0
Press any key to continue . . .
```

Figure 8 Exit

16. SOURCE CODE

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <conio.h> // for getch()
#include <unistd.h> // for sleep()

struct Student
{
    char name[50];
    int id;
    char year[50];
    char address[50];
    char email[50];
    char phonenumber[50];
    float gpa;
    float totalAmount;
    float paidAmount;
    float dueAmount;
};

const char *filename = "students_records.dat";

// Function prototypes
void AddStudent();
void DisplayAllStudents();
```

```
void SearchStudent();
void EditStudentDetails();
void DeleteStudent();
int Login()
{
        char username[50];
    char password[50];

    printf("\n\t\tUsername: ");
    scanf("%s", username);

    printf("\t\tPassword: ");
    scanf("%s", password);

    if (strcmp(username, "admin") == 0 && strcmp(password, "12345") == 0)
    {
        printf("\n\t\tLogin successful\n");
        printf("\n\t\tLogging in . . . .\n");
                sleep(3);
        return 1;
    }
    else
    {
        printf("\n\t\tLogin failed! Invalid username or password.\n");
        return 0;
    }
}
```

```
int main()
{
   char choice, x;
   int loggedIn = 0;

   while (1)
   {
      system("cls");
      printf("\t\t\t\t\t<== STUDENT MANAGEMENT SYSTEM ==>\n");
      printf("\t\t****************************************\n");



      if (!loggedIn)
      {
         printf("\n\t\tLogin to continue:\n");
         loggedIn = Login();
         if (!loggedIn)
         {
            printf("\n\t\tPress any key to retry...\n");
            getch();
            continue;
         }
      }
```

```
while (1)
      {
        flag:
        system("cls");
         printf("\t\t\t\t\t<== Student Management System ==>\n");

printf("\t\t**************************************************\n");
         fflush(stdin);
         printf("\n\t\ta. Add Student\n");
         printf("\t\tb. Display All Students\n");
         printf("\t\tc. Search Student\n");
         printf("\t\td. Edit Student Details\n");
         printf("\t\te. Delete Student\n");
         printf("\t\tf. Exit\n");
         printf("\n\t\tEnter your choice(a-f): ");
         scanf(" %c", &choice);

         switch (choice)
         {
            case 'a':
               AddStudent();
               break;

            case 'b':
               DisplayAllStudents();
               break;
```

```
case 'c':
            SearchStudent();
            break;

        case 'd':
            EditStudentDetails();
            break;

        case 'e':
            DeleteStudent();
            break;

        case 'f':
            printf("\t\tPress Y to exit and any other character to cancel: ");
            fflush(stdin);
            scanf(" %c", &x);
            if (x == 'Y' || x == 'y')
            {
                exit(0);
             }
            else
            {
                goto flag;
            }
            break;
```

```
  default:
                printf("\n\t\tInvalid Choice!\n");
                printf("\t\tPlease Enter from a to f\n");
                printf("\t\tPress any key to retry");
                getch();
                goto flag;
          }

          printf("\n\t\tPress any key to continue");
          getch();
       }
    }

    return 0;
}



void AddStudent()
{
    char another;
    do
    {
       struct Student newstudent;
       system("cls");
       printf("\t\t\t\t<== Add Student Information ==>\n\n");

       printf("\n\t\tName: ");
```

```
fflush(stdin);
gets(newstudent.name);

printf("\n\t\tId no: ");
scanf("%d", &newstudent.id);

printf("\n\t\tCourse and Semester: ");
fflush(stdin);
gets(newstudent.year);

printf("\n\t\tAddress: ");
fflush(stdin);
gets(newstudent.address);

printf("\n\t\tEmail: ");
fflush(stdin);
gets(newstudent.email);

printf("\n\t\tPhone no: ");
fflush(stdin);
gets(newstudent.phonenumber);

printf("\n\t\tGPA: ");
fflush(stdin);
scanf("%f", &newstudent.gpa);

printf("\n\t\tPaid Amount: ");
```

```
        scanf("%f", &newstudent.paidAmount);

        newstudent.totalAmount = 790000; // Fixed total amount
        newstudent.dueAmount = newstudent.totalAmount -
newstudent.paidAmount;

printf("\t\t_____________________________________________________");

        FILE *file = fopen(filename, "ab");
        if (file == NULL)
        {
            printf("\t\tUnable to open file.\n");
            return;
        }

        fwrite(&newstudent, sizeof(struct Student), 1, file);
        fclose(file);

        printf("\n\t\tDo you want to add another record?(Y/any other character): ");
        fflush(stdin);
        scanf(" %c", &another);
    }
    while (another == 'y' || another == 'Y');
}
```

```
void DisplayAllStudents()
{
   system("cls");
   printf("\t\t\t\t\t<== Student Records ==>\n\n");

   FILE *file = fopen(filename, "rb");
   if (file == NULL)
   {
      printf("\t\tUnable to open file.\n");
      return;
   }

   struct Student newstudent;
   int recordsFound = 0; // Initialize a variable to track if any records were found

   while (fread(&newstudent, sizeof(struct Student), 1, file))
   {
      recordsFound = 1; // Records were found
      printf("\n\t\tStudent Name      :   %s", newstudent.name);
      printf("\n\t\tId                :   %d", newstudent.id);
      printf("\n\t\tClass and Semester :   %s", newstudent.year);
      printf("\n\t\tAddress           :   %s", newstudent.address);
      printf("\n\t\tEmail             :   %s", newstudent.email);
      printf("\n\t\tPhone number      :   %s", newstudent.phonenumber);
      printf("\n\t\tGPA               :   %.2f", newstudent.gpa);
      printf("\n\t\tTotal Amount      :   Rs %.2f", newstudent.totalAmount);
      printf("\n\t\tPaid Amount       :   Rs %.2f", newstudent.paidAmount);
```

```
        printf("\n\t\tDue Amount        :    Rs %.2f", newstudent.dueAmount);
        printf("\n\t\t_____________________________________");
    }


    fclose(file);

    // Check if no records were found and display a message
    if (recordsFound==0)
    {
        printf("\n\t\tNo records found.\n");
    }
}

void SearchStudent()
{
    int search = 0, found = 0;
    system("cls");
    printf("\t\t\t\t<== Search Student ==>\n\n");
    FILE *file = fopen(filename, "rb");
    if (file == NULL)
    {
        printf("\t\tUnable to open file.\n");
        return;
    }
    struct Student newstudent;
    printf("\n\t\tEnter the ID you want to search: ");
```

```
    scanf("%d", &search);
    while (fread(&newstudent, sizeof(struct Student), 1, file))
    {
        if (newstudent.id == search)
        {
            found = 1;
            printf("\n\t\tStudent Name        : %s", newstudent.name);
            printf("\n\t\tID                  : %d", newstudent.id);
            printf("\n\t\tClass and Semester  : %s", newstudent.year);
            printf("\n\t\tAddress             : %s", newstudent.address);
            printf("\n\t\tEmail               : %s", newstudent.email);
            printf("\n\t\tPhone number        : %s", newstudent.phonenumber);
            printf("\n\t\tGPA                 : %.2f", newstudent.gpa);
            printf("\n\t\tTotal Amount        : Rs %.2f", newstudent.totalAmount);
            printf("\n\t\tPaid Amount         : Rs %.2f", newstudent.paidAmount);
            printf("\n\t\tDue Amount          : Rs %.2f", newstudent.dueAmount);
            printf("\n\t\t_____________________________________");
            }
    }

    if (found != 1)
    {
        printf("\n\t\tRecord not found\n");
    }
    fclose(file);
}
```

```
void EditStudentDetails()
{
   system("cls");
   printf("\t\t\t\t\t<== Edit Student Details ==>\n\n");

   int id = 0, found = 0, updated = 0;
   printf("\t\tEnter ID to Edit: ");
   scanf("%d", &id);

   struct Student updatedStudent
   FILE *file = fopen(filename, "rb+");
   if (file == NULL)
   {
      printf("\t\tUnable to open file.\n");
      return;
   }
   struct Student newstudent;

   while (fread(&newstudent, sizeof(struct Student), 1, file))
   {
      if (newstudent.id == id)
      {
         updatedStudent = newstudent;
         found = 1;
         break;
      }
}
```

```
    if (found != 1)
    {
        printf("\n\t\tStudent with ID %d not found.\n", id);
        fclose(file);
        return;
    }

    int choice;
    int repeatMenu = 1;
do
    {
        system("cls");
        printf("\t\t\t\t<== Edit Student Details ==>\n\n");
        printf("\t\tSelect fields to update:\n");
        printf("\t\t1. Name\n");
        printf("\t\t2. ID\n");
        printf("\t\t3. Semester\n");
        printf("\t\t4. Address\n");
        printf("\t\t5. Email\n");
        printf("\t\t6. Phone Number\n");
        printf("\t\t7. GPA\n");
        printf("\t\t8. Paid Amount\n");
        printf("\t\t9. Exit\n");
        printf("\t\tEnter your choice (1-9): ");

        scanf("%d", &choice);
```

```
    fflush(stdin);

    switch (choice)
    {
        case 1:
            printf("\n\t\tEnter Updated Name: ");
            fflush(stdin);
            gets(updatedStudent.name);
            updated = 1;
            break;



case 2:
            printf("\n\t\tEnter Updated ID: ");
            scanf("%d", &updatedStudent.id);
            updated = 1;
            break;

        case 3:
            printf("\n\t\tEnter Updated Semester: ");
            fflush(stdin);
            gets(updatedStudent.year);
            updated = 1;
            break;
```

```
    case 4:
            printf("\n\t\tEnter Updated Address: ");
            fflush(stdin);
            gets(updatedStudent.address);
            updated = 1;
            break;

        case 5:
            printf("\n\t\tEnter Updated Email: ");
            fflush(stdin);
            gets(updatedStudent.email);
            updated = 1;
            break;

case 6:
            printf("\n\t\tEnter Updated Phone number: ");
            gets(updatedStudent.phonenumber);
            updated = 1;
            break;

        case 7:
            printf("\n\t\tEnter Updated GPA: ");
            scanf("%f", &updatedStudent.gpa);
            updated = 1;
            break;
```

```
        case 8: // Payment Amount Update
            printf("\n\t\tEnter Updated Payment Amount: ");
            scanf("%f", &updatedStudent.paidAmount);
            updatedStudent.dueAmount = updatedStudent.totalAmount -
updatedStudent.paidAmount;
            updated = 1;
            break;

        case 9:
            if (updated == 1)
            {
               printf("\n\t\tStudent details successfully updated.\n");
            }
            else
            {
               printf("\n\t\tNo updates were made.\n");
            }
            repeatMenu = 0;
            break;


        default:
            printf("\n\t\tInvalid choice!");
            printf("\n\t\tPress any key to retry");
            getch();
            break;
      }
   } while (repeatMenu);
```

```
    fclose(file); // Close the file after editing

    file = fopen(filename, "rb+");
    if (file == NULL)
    {
        printf("\t\tUnable to open file.\n");
        return;
    }
    while (fread(&newstudent, sizeof(struct Student), 1, file))
    {
        if (newstudent.id == id)
        {
            fseek(file, -((long int)sizeof(struct Student)), SEEK_CUR);
            fwrite(&updatedStudent, sizeof(struct Student), 1, file);
            break;
        }
    }
    fclose(file);
}

void DeleteStudent()
{
    int search = 0, found = 0;
    system("cls");
    printf("\t\t\t\t\t<== Delete Student Record ==>\n\n");
```

```
    printf("\t\tEnter the ID you want to delete: ");
    scanf("%d", &search);

    FILE *file = fopen(filename, "rb");
    if (file == NULL)
    {
        printf("\t\tUnable to open file.\n");
        return;
    }

    FILE *tempFile = fopen("temp.dat", "wb");

 if (tempFile == NULL)
    {
        printf("\t\tUnable to create temporary file.\n");
        fclose(file);
        return;
    }
    struct Student newstudent;
    fflush(stdin);

    while (fread(&newstudent, sizeof(struct Student), 1, file))
    {
        if (newstudent.id != search)
        {
            fwrite(&newstudent, sizeof(struct Student), 1, tempFile);
        }
```

```
        else
        {
            found = 1;
        }
    }
    fclose(file);
    fclose(tempFile);
    remove(filename);
    rename("temp.dat", filename);
    if (found) {
        printf("\t\tStudent with ID %d deleted successfully.\n", search);
 }
    else {
        printf("\t\tStudent with ID %d not found.\n", search);
    }
}
```

17.REFERENCES

- Open AI. “ChatGPT.” Chat.openai.com, Open AI, 30 Nov. 2022, chat.openai.com/. Accessed 2 July 2023.
- Open AI. “ChatGPT.” Chat.openai.com, Open AI, 30 Nov. 2022, chat.openai.com/. Accessed 24 July 2023.
- Roberts, Bucky. “Thenewboston - YouTube.” Www.youtube.com, Bucky Roberts April. 2011, youtube.com/user/thenewboston. Accessed 28 July 2023.
- Jain, Sandeep. “GeeksforGeeks | a Computer Science Portal for Geeks.” GeeksforGeeks, Sandeep Jain, 27 Mar. 2023, geeksforgeeks.org. Accessed 3 Aug. 2023.