STUDENT MANAGEMENT SYSTEM

Submitted by - Mannat Arora

SAP id - 590028068

Batch - 60

Submitted to - Mr. Prashant Trivedi

Abstract

The Student Management System in C is a console-based program designed to store, manage, and process student records efficiently. It allows adding, viewing, searching, updating, and deleting student information using structures, pointers, and file handling. The system reduces manual errors and ensures organized student data management.

Problem Definition

Manual handling of student records is slow and error-prone. There is a need for a system that can efficiently store, search, update, and manage student information. The Student Management System in C provides an organized and fast solution for handling student data.

Algorithms and Flowchart

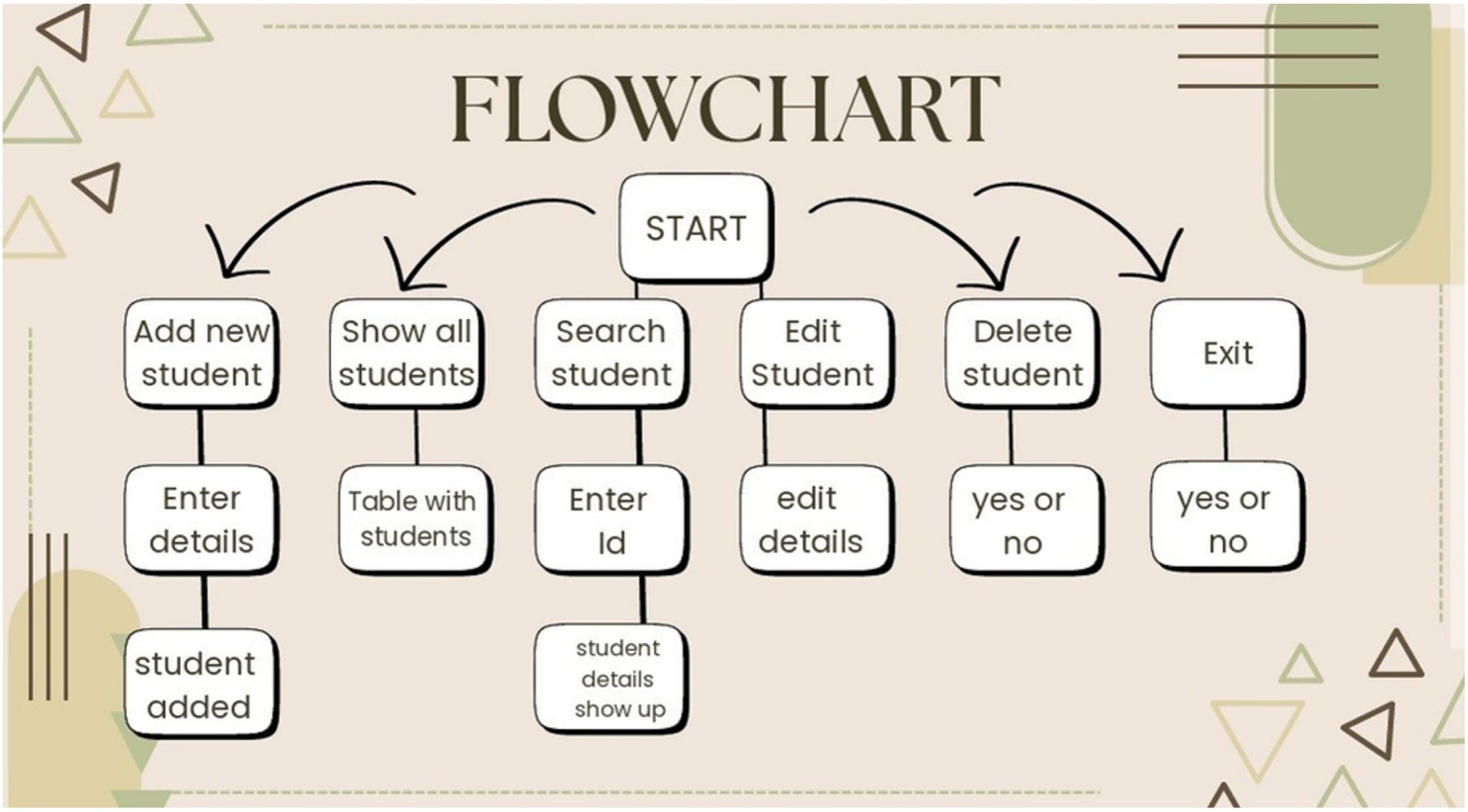
1. Add Student: Input details → store in file → confirm addition.

2. View Students: Read file → display all records.

3. Search Student: Input roll number → find record → display or “Not Found”.

4. Update Student: Input roll → search → modify → overwrite record.

5. Delete Student: Input roll → copy all except target → replace file.



IMPLEMENTATION

1.Menu with all the functions that can initiated and completed are shown as options.

```
void Menu()
{
printf("\n\n\t**Student Man
agement System Using C\n\n"
);
printf("\t\t\tMAIN MENU\n");
printf("\t\t======================\n");
printf("\t\t[1] Add A New Student.\n");
printf("\t\t[2] Show All Students.\n");
printf("\t\t[3] Search A student.\n");
printf("\t\t[4] Edit Student.\n");
printf("\t\t[5] Delete Student.\n");
printf("\t\t[6] Delete All Students.\n");
printf("\t\t[7] User Guideline.\n");
printf("\t\t[0] Exit the Program.\n");
printf("\t\t======================\n");
printf("\t\tEnter Your Choice: ");
printf("\t\tEnter The Choice: ");
}   // end menu
```

2.Some of the implementations are as follows -

2.a Adding a new student

```
void AddNewStudent()
{
    char StudentID[300];
    char Name[300];
    char Phone[300];
    char Email[300];
    int NumberOfCourses;
    char CourseCode[300];
    char CourseName[300];
```

2.b Show all students (table form)

```
void ShowAllStudents()
{
    printf("|==========|
====================|
================================|
====================|=============|\n");
    printf("|    ID    |
        Name         |
            Email              |
       Phone         |  NO.Course  |\n");
    printf("|==========|
====================|
================================|
====================|=============|\n");
```

2.c Edit student details

```
void EditStudent(int StudentFoundIndex)
{
    printf("\n\t\t **** Upda
te The New Student ****\n\n"
);

    char NewName[300];
    char NewPhone[300];
    char NewEmail[300];
    int NewNumberOfCourses;
    char StudentID[300];
    strcpy(StudentID, Students[
StudentFoundIndex].ID);
    int OldTotalNumberOfCourse = Students[
StudentFoundIndex].NumberOfCourse;

    int IsValidName = 0;
    while(!IsValidName)
    {
        printf(
" Enter The New Name(0 for skip): ");
        scanf(" %[^\n]s",&NewName);
        if(strlen(NewName) > 20)
        {
            printf(" Error: Name can
not be more than 20 characters.\n\n"
);
            IsValidName = 0;
        }
        else if(strlen(NewName) <= 0)
```

2.d delete Student details

```
void DeleteStudent(int StudentIndex)
{
    int d;
    int FirstCourseIndexs;
    struct StudentInfo ThisStudents;
    ThisStudents = Students[StudentIndex];
    for(d=0; d<TotalCourse; d++)
    {
        if(strcmp(ThisStudents.ID,Courses[d
].StudentID) == 0)
        {
            FirstCourseIndexs = d;
            break;
        }
    }
    for(d=1; d<=ThisStudents.NumberOfCourse
; d++)
    {
        DeleteCourseByIndex(
FirstCourseIndexs);
    }
    DeleteStudentByIndex(StudentIndex);
    printf(" Student Deleted Successfully.
\n\n");
    GoBackOrExit();
}
```

2.e Go back from one implementation to menu or exit the program

```
void GoBackOrExit()
{
    getchar();
    char Option;
    printf(" Go back(b)? or Exit(0)?: ");
    scanf("%c",&Option);
    if(Option == '0')
    {
        ExitProject();
    }
    else
    {
        system("cls");
    }
}

void ExitProject()
{
    system("cls");
    int i;
    char ThankYou[100]     =
" ========= Thank You =========\n";
    char SeeYouSoon[100]   =
" ========= See You Soon ======\n";
    for(i=0; i<strlen(ThankYou); i++){
    printf("%c",ThankYou[i]);
        Sleep(40);
        }
    for(i=0; i<strlen(SeeYouSoon); i++){
            printf("%c",SeeYouSoon[i]);
    sleep(40);
    }
```

Conclusion & Future Work

The system provides an organized, fast, and error-free method to manage student records. It demonstrates file handling, structures, and basic C programming concepts effectively.

1.Implement GUI

2.Add login and role-based access

3.Include grades, attendance, and reports

4.Convert into a database-driven application for scalability

References

1.The study material provided by the teacher.

2. Online C documentation.

(www.programiz.com, online C tutorials)