PROJECT REPORT

Student Management System using Python

INDEX

1. Introduction

In today's digital world, managing student records manually is time-consuming and error-prone. The Student Management System is a simple Python-based application developed to manage student information digitally. It helps in storing, searching, viewing, and deleting student records efficiently.

This project is designed for BCA students to understand the practical implementation of Python programming and file handling concepts.

---

2. Objectives of the Project

The main objectives of this project are:

- To maintain student records digitally
- To reduce paperwork and manual effort
- To provide fast searching of student data
- To understand Python file handling concepts
- To create a simple and user-friendly system

---

3. Scope of the Project

This project can be used in schools, colleges, and coaching institutes for maintaining basic student information such as roll number, name, and course.

It is a small-scale management system and can be extended in future with:

- Database integration
- Graphical user interface (GUI)
- Attendance management

- Marks and result system

---

4. Algorithm

Algorithm for Student Management System

Step 1

Start the program.

Step 2

Display the main menu:

1. **Add Student**
2. **View Students**
3. **Search Student**
4. **Delete Student**
5. **Exit**

Step 3

Ask the user to enter their choice.

Step 4

If the user selects Add Student:

- **Input Roll Number, Name, and Course.**
- **Store the data in the file.**

Step 5

If the user selects View Students:

- **Read all student records from the file.**

- **Display them on the screen.**

Step 6

If the user selects Search Student:

- **Input Roll Number.**
- **Search the record in the file.**
- **Display student details if found.**

Step 7

If the user selects Delete Student:

- **Input Roll Number.**
- **Remove the record from the file.**

Step 8

If the user selects Exit:

- **Terminate the program.**

Step 9

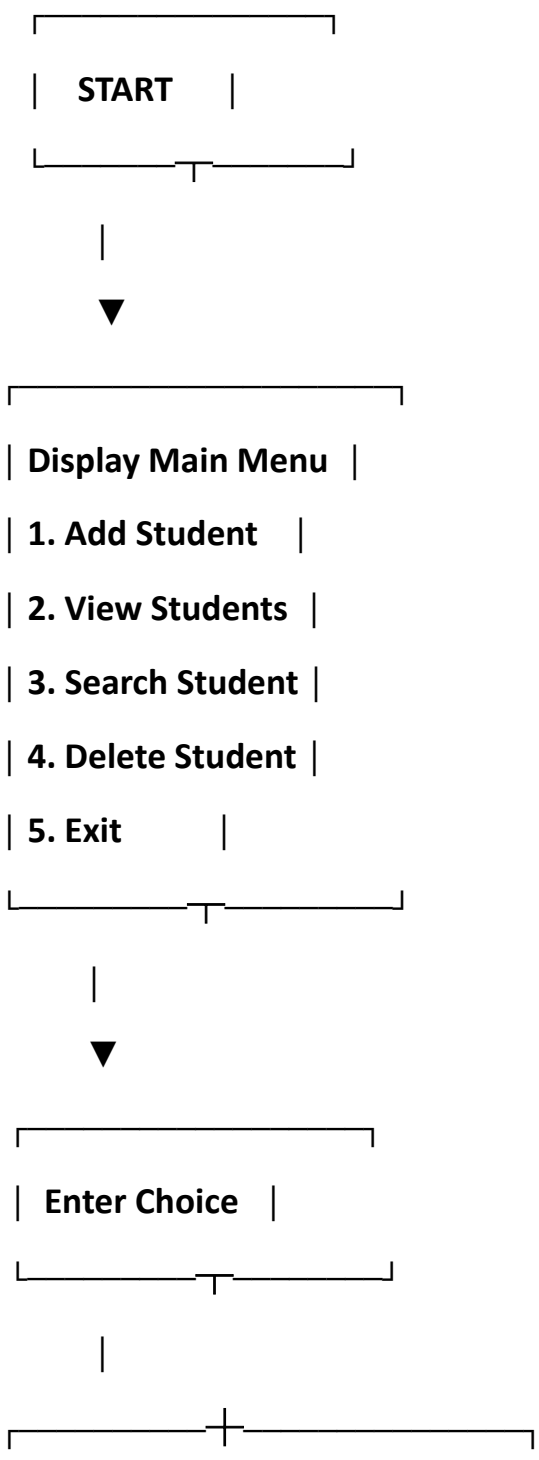
If the user enters an invalid choice:

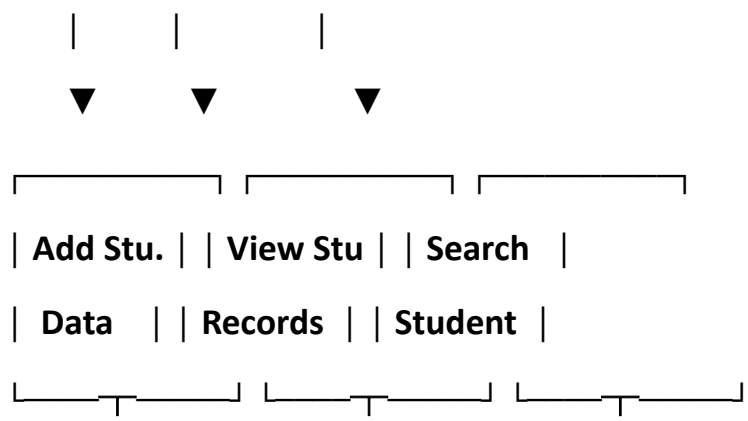
- **Display error message.**
- **Return to the main menu.**

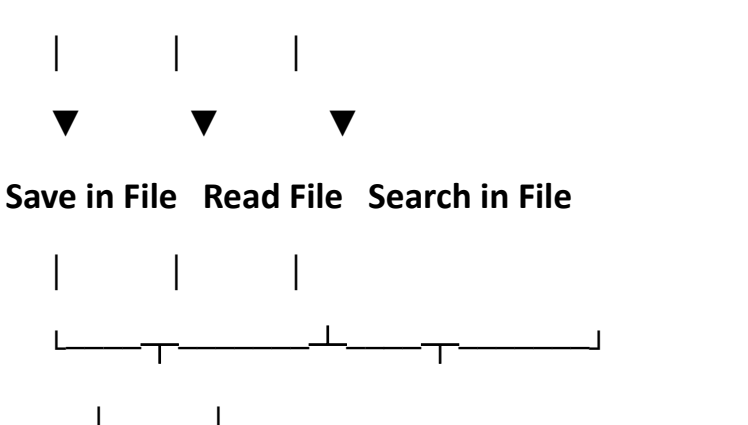
Step 10

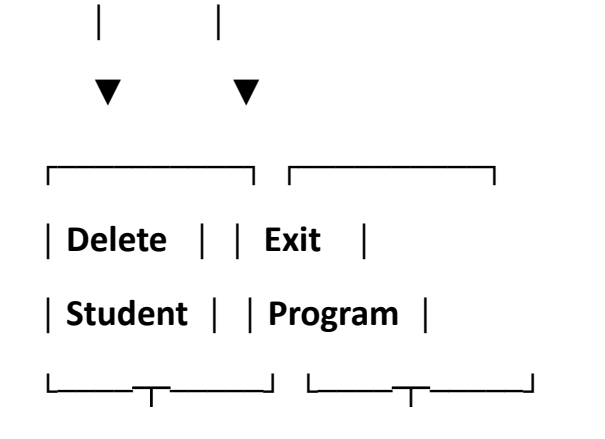
Stop the program.

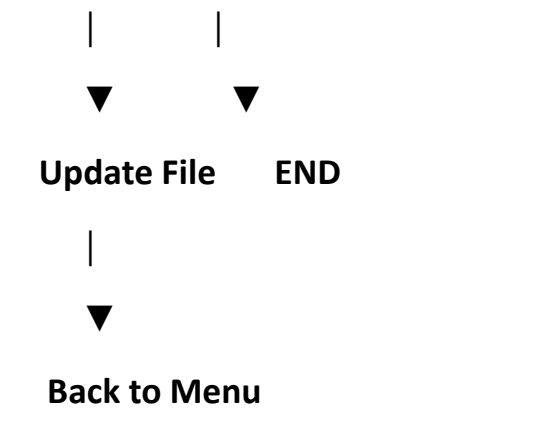
5.FLOWCHART


START
Display Main Menu
1. Add Student
2. View Students
3. Search Student
4. Delete Student
5. Exit
Enter Choice


Add Stu. Data
View Stu Records
Search Student


Save in File
Read File
Search in File


Delete Student
Exit Program


Update File
END

Back to Menu

---

6. Tools and Technologies Used

Tool / Technology	Description
Python	Programming Language
File Handling	Data storage
Text File	Student record storage
VS Code / IDLE	Code Editor

---

7. System Requirements

Hardware Requirements:

- Computer / Laptop
- Minimum 2GB RAM
- Keyboard & Mouse

Software Requirements:

- Windows / Linux
- Python 3.x
- Any Code Editor (VS Code / IDLE)

---

8. Project Description

The Student Management System provides the following functionalities:

1. Add Student

Allows user to add new student record including:

- Roll Number
- Name
- Course

2. View Students

Displays all student records stored in the file.

3. Search Student

Searches a student record using roll number.

4. Delete Student

Deletes a student record from the file.

5. Exit

Closes the application.

---

9. Advantages of the System

- Simple and easy to use
- Fast access to student records
- Reduces manual paperwork
- Secure file storage

- Easy to maintain

---

10. Future Enhancements

In future, this project can be enhanced by:

- Adding login authentication
- Using database instead of text file
- Adding marks and attendance module
- Creating GUI using Tkinter
- Generating reports automatically

---

11. source code

```
def add_student():
    roll = input("Enter Roll No: ")
    name = input("Enter Name: ")
    course = input("Enter Course: ")

    with open("students.txt", "a") as f:
        f.write(f"{roll},{name},{course}\n")

    print("Student Added Successfully!")

def view_students():
    try:
        with open("students.txt", "r") as f:
```

```
            data = f.readlines()
            if not data:
                print("No students found!")
            else:
                print("\n--- Student List ---")
                for line in data:
                    roll, name, course = line.strip().split(",")
                    print(f"Roll: {roll} | Name: {name} | Course: {course}")
    except FileNotFoundError:
        print("No record file found!")

def search_student():
    roll = input("Enter Roll No to search: ")
    found = False

    try:
        with open("students.txt", "r") as f:
            for line in f:
                r, name, course = line.strip().split(",")
                if r == roll:
                    print("\nStudent Found!")
                    print(f"Roll: {r}")
                    print(f"Name: {name}")
                    print(f"Course: {course}")
                    found = True
                    break
```

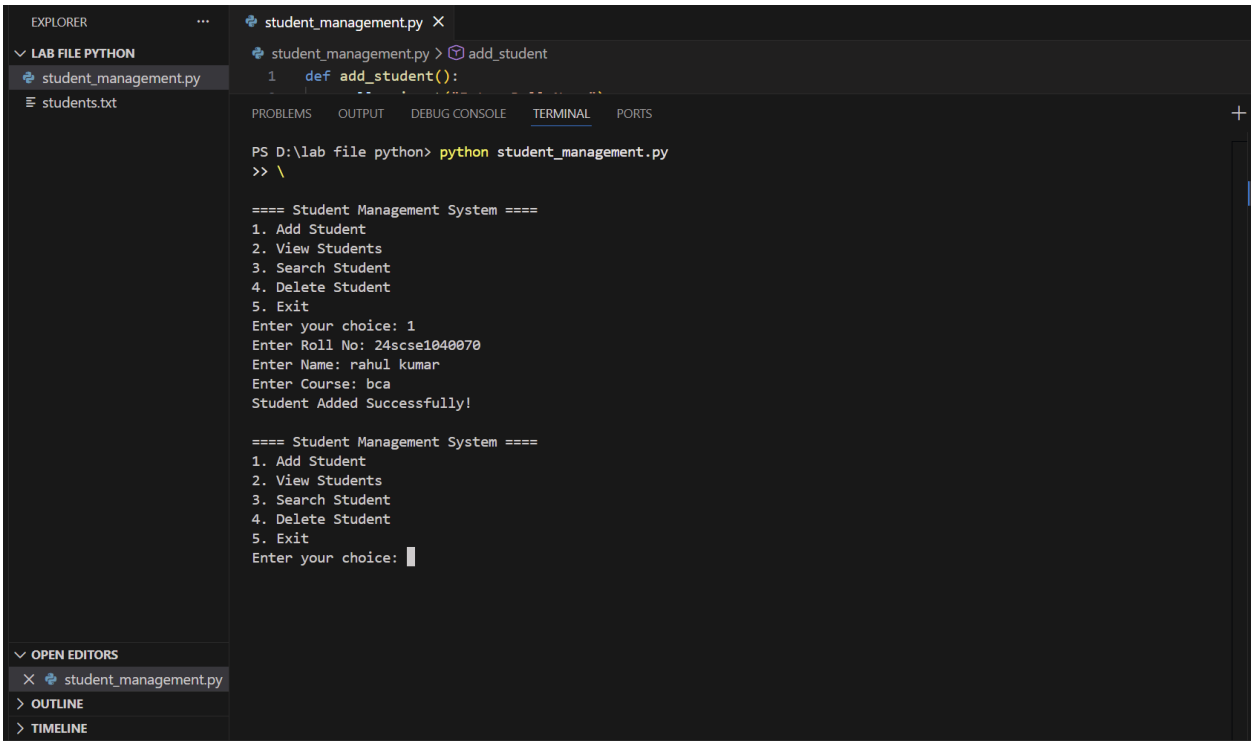
```
        if not found:
            print("Student not found!")
    except FileNotFoundError:
        print("No record file found!")

def delete_student():
    roll = input("Enter Roll No to delete: ")
    found = False
    new_data = []

    try:
        with open("students.txt", "r") as f:
            for line in f:
                r, name, course = line.strip().split(",")
                if r != roll:
                    new_data.append(line)
                else:
                    found = True

        with open("students.txt", "w") as f:
            for line in new_data:
                f.write(line)

        if found:
            print("Student deleted successfully!")
        else:
```

```
            print("Student not found!")

    except FileNotFoundError:
        print("No record file found!")

while True:
    print("\n==== Student Management System ====")
    print("1. Add Student")
    print("2. View Students")
    print("3. Search Student")
    print("4. Delete Student")
    print("5. Exit")

    choice = input("Enter your choice: ")

    if choice == "1":
        add_student()
    elif choice == "2":
        view_students()
    elif choice == "3":
        search_student()
    elif choice == "4":
        delete_student()
    elif choice == "5":
        print("Thank you! Exiting...")
        break
```

```
else:
    print("Invalid choice! Try again.")
```

---

12. Screen short of running project



---

13. Conclusion

The Student Management System is a simple and efficient Python-based application that helps in managing student records digitally. It demonstrates the use of Python programming and file handling in real-world applications. This project is suitable for BCA students to understand basic software development concepts.
