



Green University of Bangladesh

*Department of Computer Science and Engineering (CSE)
Semester: (Spring, Year: 2024), B.Sc. in CSE (Day)*

Student Management System

*Course Title: Operating System Lab
Course Code: CSE 310
Section: CSE 213 - D4*

Students Details

Name	ID
Md. Kawsar Ahamed	213902013

*Submission Date: 13 June 2024
Course Teacher's Name: Md. Shoab Alam*

[For teachers use only: **Don't write anything inside this box**]

<u>Lab Project Status</u>	
Marks:	Signature:
Comments:	Date:

Contents

1	Introduction	4
1.1	Overview	4
1.2	Motivation	4
1.3	Problem Definition	4
1.3.1	Data Inconsistency and Loss	5
1.3.2	Inefficiency	5
1.3.3	Limited Accessibility	5
1.3.4	High Costs	5
1.3.5	Lack of Integration	5
1.4	Design Goals/Objectives	5
1.4.1	User-Friendliness	5
1.4.2	Data Integrity and Accuracy	5
1.4.3	Comprehensive Functionality	6
1.4.4	Efficiency	6
1.4.5	Cost-Effectiveness	6
1.4.6	Data Security and Backup	6
1.4.7	Flexibility and Scalability	6
1.5	Application	6
1.5.1	Record Management	6
1.5.2	Grade Calculation	7
1.5.3	Data Retrieval	7
1.5.4	Performance Analysis	7
1.5.5	Data Backup and Restoration	7
1.5.6	Administrative Efficiency	7
1.5.7	Scalability and Adaptability	7
2	Design/Development/Implementation of the Project	8

2.1	Introduction	8
2.2	Project Details	8
2.2.1	User Interface	9
2.3	Implementation	9
2.3.1	GitHub Link	9
2.3.2	Tools and libraries	9
2.4	Algorithms	10
3	Performance Evaluation	12
3.1	Simulation Environment/ Simulation Procedure	12
3.2	Results Analysis/Testing	12
3.2.1	Result_portion_1	12
3.2.2	Result_portion_2	13
3.2.3	Result_portion_3	13
3.2.4	Result_portion_4	14
3.2.5	Result_portion_5	14
3.2.6	Result_portion_6	15
3.2.7	Result_portion_7	15
3.2.8	Result_portion_8	16
3.2.9	Result_portion_9	16
3.2.10	Result_portion_10	17
3.2.11	Result_portion_11	17
3.2.12	Result_portion_12	18
3.2.13	Result_portion_13	18
3.2.14	Result_portion_14	19
3.2.15	Result_portion_15	19
3.2.16	Result_portion_16	20
3.2.17	Result_portion_17	20
3.2.18	Result_portion_18	21
3.2.19	Result_portion_19	21
3.2.20	Result_portion_20	22
3.2.21	Result_portion_21	22
3.2.22	Result_portion_22	23
3.2.23	Result_portion_23	23
3.2.24	Result_portion_24	24

3.3	Results Overall Discussion	24
4	Conclusion	25
4.1	Discussion	25
4.2	Limitations	26
4.3	Scope of Future Work	26

Chapter 1

Introduction

1.1 Overview

The Student Management System (SMS) is a shell script-based application developed by Md. Kawsar Ahamed for managing student data efficiently. It allows users to create, edit, delete, and view student records, along with additional functionalities like sorting, searching by ID or name, and viewing statistical data. The system also supports backup and restore operations, ensuring data integrity and easy recovery. This project is designed to streamline student information management in an academic setting.

1.2 Motivation

The motivation behind developing the Student Management System (SMS) stems from the need for an efficient, streamlined solution to manage student information in academic institutions. Manual record-keeping is prone to errors and time-consuming while existing digital solutions can be complex and costly. By leveraging a simple, shell script-based approach, this project aims to provide a cost-effective, user-friendly alternative that enhances data accuracy and accessibility for educators and administrators.

1.3 Problem Definition

Educational institutions often face several challenges in managing student information efficiently. Traditional paper-based record-keeping is not only time-consuming but also prone to errors, misplacement, and damage. Even with digital solutions, many existing systems are either too complex, requiring extensive training, or too costly for smaller institutions.

1.3.1 Data Inconsistency and Loss

Manual entry errors and physical damage can lead to inconsistent and lost records.

1.3.2 Inefficiency

Time-consuming processes for searching, updating, and sorting student records.

1.3.3 Limited Accessibility

Difficulty in accessing records quickly when needed, especially in emergencies or audits.

1.3.4 High Costs

Commercial student management systems can be prohibitively expensive for many schools and universities.

1.3.5 Lack of Integration

Existing solutions often lack the flexibility to integrate with other systems or adapt to specific institutional needs.

1.4 Design Goals/Objectives

The primary objective of the Student Management System (SMS) is to create a robust and efficient tool for managing student records in educational institutions.

1.4.1 User-Friendliness

Develop an intuitive interface that allows users, regardless of their technical proficiency, to easily manage student information. The command-line interface is designed to be straightforward, with clear prompts and options.

1.4.2 Data Integrity and Accuracy

Ensure that the system accurately records and maintains student information. Features like input validation and duplicate ID checks are incorporated to minimize errors and ensure the reliability of the data.

1.4.3 Comprehensive Functionality

Provide a wide range of functionalities including creating, editing, deleting, viewing, and sorting student records. Additional features like searching by ID or name, viewing statistics, and adding comments are included to make the system comprehensive.

1.4.4 Efficiency

Streamline operations to save time and reduce the manual effort involved in managing student data. Automated grade calculation and sorting algorithms help in quickly organizing and retrieving information.

1.4.5 Cost-Effectiveness

Offer a free and open-source solution that can be easily implemented without the need for expensive software licenses or extensive hardware upgrades. The shell script-based approach ensures minimal resource consumption and broad compatibility.

1.4.6 Data Security and Backup

Incorporate features for data backup and restoration to protect against data loss and ensure that records can be recovered in case of accidental deletion or corruption.

1.4.7 Flexibility and Scalability

Design the system to be flexible and adaptable to the specific needs of different educational institutions. The shell script can be easily modified or extended to include additional features or integrate with other systems.

1.5 Application

The Student Management System (SMS) is designed to be utilized by educational institutions such as schools, colleges, and universities to streamline the management of student records.

1.5.1 Record Management

Facilitates the efficient creation, editing, deletion, and viewing of student records, ensuring that all relevant information is accurately maintained and easily accessible.

1.5.2 Grade Calculation

Automatically calculates and assigns grades based on student marks, reducing manual errors and ensuring consistency in grade assignments.

1.5.3 Data Retrieval

Enables quick searching of student records by ID or name, allowing educators and administrators to retrieve specific information rapidly during evaluations, audits, or parent-teacher meetings.

1.5.4 Performance Analysis

Provides statistical insights into student performance, such as average marks, highest and lowest marks, and grade distribution, aiding in the assessment of academic outcomes and identifying areas for improvement.

1.5.5 Data Backup and Restoration

Ensures data security by offering backup and restoration functionalities, safeguarding against data loss due to accidental deletion or system failures.

1.5.6 Administrative Efficiency

Enhances overall administrative efficiency by automating routine tasks and minimizing the time spent on manual record-keeping and data entry.

1.5.7 Scalability and Adaptability

The script-based approach allows the system to be easily customized and scaled according to the specific needs of different institutions, making it a versatile tool suitable for a wide range of educational environments.

Chapter 2

Design/Development/Implementation of the Project

2.1 Introduction

The design, development, and implementation of the Student Management System (SMS) focus on creating a practical and efficient solution for managing student records using a shell script-based approach. The design phase involves outlining the system's structure and functionalities, ensuring it is user-friendly and meets the needs of educational institutions. Development encompasses coding the script, incorporating features such as record creation, editing, deletion, viewing, sorting, and statistical analysis, as well as ensuring data validation and integrity. Implementation involves deploying the system in a real-world setting, testing its performance, and making necessary adjustments based on user feedback. This structured approach ensures that the SMS is reliable, efficient, and tailored to improve the overall management of student information in academic environments.

2.2 Project Details

The Student Management System (SMS) is a comprehensive, shell script-based application designed to manage student records efficiently. The project was developed by Md. Kawsar Ahamed is a practical solution to address the challenges faced by educational institutions in handling student data. The system's core functionalities include creating, editing, deleting, and viewing student records, with additional features for sorting and searching by student ID or name. It also provides automated grade calculation based on marks, statistical analysis of student performance, and options for adding comments to records. To ensure data integrity and security, the SMS includes backup and restore capabilities. The script's user-friendly design and cost-effective nature make it an ideal tool for schools, colleges, and universities aiming to streamline their administrative processes and enhance data management practices.

2.2.1 User Interface

```
=====
                        Student Management System
=====
1. Create Student
2. Edit Student
3. Delete Student
4. View Students
5. Search Student by ID
6. Search Student by Name
7. View Statistics
8. Sort Students
9. Add Comment
10. Backup Data
11. Restore Data
12. Exit
Enter your choice:
```

Figure 2.1: User Interface

2.3 Implementation

2.3.1 GitHub Link

<https://github.com/KaSagor/Student-Management-System-using-shell>

2.3.2 Tools and libraries

This project is a simple student management system implemented using Bash shell scripting. Since Bash is a scripting language built into Unix-based operating systems like Linux, macOS, and others, so no need to install any additional tools or libraries to run this project.

Visual Studio Code

It's a free, open-source code editor developed by Microsoft with support for various programming languages, including Bash scripting. You can enhance your development experience by installing extensions like ShellCheck for linting Bash scripts.

2.4 Algorithms

- Initialize Variables and Constants:
Set up variables for storing student data and file paths.
Initialize a flag to track if it's the first time running the program.
- Define Functions:
Displaying the header.
Calculating grades.
Creating a new student record.
Editing existing student records.
Deleting student records.
Viewing all students.
Searching for a student by ID or name.
Viewing statistics.
Sorting students by different criteria.
Adding comments to student records.
Backing up and restoring data.
- Main Menu Loop:
Enter an infinite loop to display the main menu options.
Prompt the user for input and execute the corresponding function based on their choice.
Repeat until the user chooses to exit the program.
- Display Header:
Display the header with the program name and a separator.
- Calculate Grades:
Define a function to calculate grades based on marks obtained.
- Create Student:
Prompt the user to input student details.
Check for duplicate student IDs.
Validate input and store the new student record in the data file.
- Edit Student:
Prompt the user to input the ID of the student to edit.
If the student exists, allow the user to update their information.
Update the data file with the new information.
- Delete Student:
Prompt the user to input the ID of the student to delete.
If the student exists, remove their record from the data file.

- View Students:
Display a formatted list of all students stored in the data file.
- Search Student:
Prompt the user to input either the ID or name of the student to search for.
Display the student's information if found.
- View Statistics:
Calculate and display statistics such as total number of students, average marks, highest and lowest marks, and grade distribution.
- Sort Students:
Prompt the user to choose a sorting criteria (by name or marks).
Sort the student records accordingly and display the sorted list.
- Add Comment:
Prompt the user to input the ID of the student to add a comment.
If the student exists, allow the user to add a comment to their record.
- Backup and Restore Data:
Allow the user to create a backup of the data file or restore data from a previous backup file.
- Exit Program:
Terminate the program execution when the user chooses to exit.

Chapter 3

Performance Evaluation

3.1 Simulation Environment/ Simulation Procedure

The simulation environment for performance evaluation of the Student Management System could involve creating a synthetic dataset representing various scenarios encountered in a real educational institution. This dataset would include a range of student records with different attributes such as student ID, name, address, course, and marks. Additionally, it could encompass diverse operations like creating, editing, deleting, and searching for student records, as well as generating statistical reports.

3.2 Results Analysis/Testing

3.2.1 Result_portion_1

When inputting a Duplicate Student ID.

```
=====
                        Student Management System
=====
1. Create Student
2. Edit Student
3. Delete Student
4. View Students
5. Search Student by ID
6. Search Student by Name
7. View Statistics
8. Sort Students
9. Add Comment
10. Backup Data
11. Restore Data
12. Exit
Enter your choice:
1
Enter student ID:
4
Duplicate ID found. Please use another ID.
Enter student ID:
```

Figure 3.1: Create a Duplicate Id

3.2.2 Result_portion_2

Create a Student with the name Kawsar, ID 20.

```
=====
                        Student Management System
=====
1. Create Student
2. Edit Student
3. Delete Student
4. View Students
5. Search Student by ID
6. Search Student by Name
7. View Statistics
8. Sort Students
9. Add Comment
10. Backup Data
11. Restore Data
12. Exit
Enter your choice:
1
Enter student ID:
4
Duplicate ID found. Please use another ID.
Enter student ID:
20
Enter student name:
Kawsar
Enter student address:
Mirpur
Enter student course:
CSE101
Enter student marks:
55
Student created successfully with ID: 20
Do you want to clear the screen? (y/n):
```

Figure 3.2: Create Student

3.2.3 Result_portion_3

View Student Details to check whether Kawsar added or not.

```
=====
                        Student Management System
=====
1. Create Student
2. Edit Student
3. Delete Student
4. View Students
5. Search Student by ID
6. Search Student by Name
7. View Statistics
8. Sort Students
9. Add Comment
10. Backup Data
11. Restore Data
12. Exit
Enter your choice:
4
===== Student List =====
ID      Name      Address      Course      Marks      Grade      A+
1      AnyOneFromA      Dhaka      CSE      77      A
2      AnyOneFromB      Dhaka      CSE      44      D
3      AnyOneFromC      Bogra      CSE      35      F
4      AnyOneFromD      Rajshahi      CSE      82      A+
5      AnyOneFromE      Dhaka      EEE      76      A
6      AnyOneFromF      Bogra      EEE      79      A
7      AnyOneFromZ      Dhaka      EEE      39      F
20      Kawsar      Mirpur      CSE101      55      B-
Do you want to clear the screen? (y/n):
```

Figure 3.3: View Student Details

3.2.4 Result_portion_4

When trying to Edit a student but Student ID is not Matched.

```
=====
Student Management System
=====
1. Create Student
2. Edit Student
3. Delete Student
4. View Students
5. Search Student by ID
6. Search Student by Name
7. View Statistics
8. Sort Students
9. Add Comment
10. Backup Data
11. Restore Data
12. Exit
Enter your choice:
2
Enter student ID to edit:
22
Student ID not found.
Do you want to clear the screen? (y/n):
```

Figure 3.4: Edit Student but ID not matched

3.2.5 Result_portion_5

Edit a student from Kawsar to Sagor.

```
=====
Student Management System
=====
1. Create Student
2. Edit Student
3. Delete Student
4. View Students
5. Search Student by ID
6. Search Student by Name
7. View Statistics
8. Sort Students
9. Add Comment
10. Backup Data
11. Restore Data
12. Exit
Enter your choice:
2
Enter student ID to edit:
20
Enter new student name:
Sagor
Enter new student address:
Dhanmondi
Enter new student course:
CSE201
Enter new student marks:
66
Student information updated successfully.
Do you want to clear the screen? (y/n):
```

Figure 3.5: Edit Student

3.2.6 Result_portion_6

View edited student Sagor.

```
=====
Student Management System
=====
1. Create Student
2. Edit Student
3. Delete Student
4. View Students
5. Search Student by ID
6. Search Student by Name
7. View Statistics
8. Sort Students
9. Add Comment
10. Backup Data
11. Restore Data
12. Exit
Enter your choice:
4
===== Student List =====
ID      Name      Address      Course      Marks      Grade
1      AnyOneFromA      Dhaka      CSE      77      A
2      AnyOneFromB      Dhaka      CSE      44      D
3      AnyOneFromC      Bogra      CSE      35      F
4      AnyOneFromD      Rajshahi      CSE      82      A+
5      AnyOneFromE      Dhaka      EEE      76      A
6      AnyOneFromF      Bogra      EEE      79      A
7      AnyOneFromZ      Dhaka      EEE      39      F
20     Sagor      Dhanmondi      CSE201      66      B+
Do you want to clear the screen? (y/n):
```

Figure 3.6: View edited student

3.2.7 Result_portion_7

Search for a student with ID.

```
=====
Student Management System
=====
1. Create Student
2. Edit Student
3. Delete Student
4. View Students
5. Search Student by ID
6. Search Student by Name
7. View Statistics
8. Sort Students
9. Add Comment
10. Backup Data
11. Restore Data
12. Exit
Enter your choice:
5
Enter student ID to search:
20
ID: 20
Name: Sagor
Address: Dhanmondi
Course: CSE201
Marks: 66
Grade: B+
Do you want to clear the screen? (y/n):
```

Figure 3.7: Search student by ID

3.2.8 Result_portion_8

Search for a student with a Name.

```
=====
Student Management System
=====
1. Create Student
2. Edit Student
3. Delete Student
4. View Students
5. Search Student by ID
6. Search Student by Name
7. View Statistics
8. Sort Students
9. Add Comment
10. Backup Data
11. Restore Data
12. Exit
Enter your choice:
6
Enter student name to search:
Sagor
20 Sagor Dhanmondi CSE201 66
ID: 20
Name: Sagor
Address: Dhanmondi
Course: CSE201
Marks: 66
Grade: B+
Do you want to clear the screen? (y/n):
█
```

Figure 3.8: Search student by Name

3.2.9 Result_portion_9

View Statistics

```
=====
Student Management System
=====
1. Create Student
2. Edit Student
3. Delete Student
4. View Students
5. Search Student by ID
6. Search Student by Name
7. View Statistics
8. Sort Students
9. Add Comment
10. Backup Data
11. Restore Data
12. Exit
Enter your choice:
7
Total Students: 8
Average Marks: 62
Highest Marks: 82
Lowest Marks: 35
Grade Distribution:
F: 2
D: 1
C: 0
C+: 0
B+: 0
B: 0
B+: 1
A+: 0
A: 3
A+: 1
Do you want to clear the screen? (y/n):
█
```

Figure 3.9: Statistics

3.2.10 Result_portion_10

View the sort student option.

```
=====
Student Management System
=====
1. Create Student
2. Edit Student
3. Delete Student
4. View Students
5. Search Student by ID
6. Search Student by Name
7. View Statistics
8. Sort Students
9. Add Comment
10. Backup Data
11. Restore Data
12. Exit
Enter your choice:
8
Sort students by:
1. Name
2. Marks
Enter your choice:
```

Figure 3.10: Sort student

3.2.11 Result_portion_11

Sort students by Name.

```
Enter your choice:
8
Sort students by:
1. Name
2. Marks
Enter your choice:
1
===== Sorted Student List =====
ID      Name      Address      Course      Marks      Grade      A+
1      AnyOneFromA      Dhaka      CSE      77      A
2      AnyOneFromB      Dhaka      CSE      44      D
3      AnyOneFromC      Bogra      CSE      35      F
4      AnyOneFromD      Rajshahi      CSE      82      A+
5      AnyOneFromE      Dhaka      EEE      76      A
6      AnyOneFromF      Bogra      EEE      79      A
7      AnyOneFromZ      Dhaka      EEE      39      F
20     Sagor      Dhanmondi      CSE201      66      B+
```

Figure 3.11: Sort by Names

3.2.12 Result_portion_12

Sort students by Marks.

```
Enter your choice:
8
Sort students by:
1. Name
2. Marks
Enter your choice:
2
===== Sorted Student List =====
ID      Name      Address      Course      Marks      Grade      A+
4      AnyOneFromD      Rajshahi      CSE      79      A
6      AnyOneFromF      Bogra      EEE      77      A
1      AnyOneFromA      Dhaka      CSE      76      A
5      AnyOneFromE      Dhaka      EEE      66      B+
20     Sagor      Dhanmondi      CSE201      44      D
2      AnyOneFromB      Dhaka      CSE      39      F
7      AnyOneFromZ      Dhaka      EEE      35      F
3      AnyOneFromC      Bogra      CSE
Do you want to clear the screen? (y/n):
```

Figure 3.12: Sort by Marks

3.2.13 Result_portion_13

Student text file before comment.

```
Jun 8 15:51
• students.txt
~/Documents

Open ▾  [icon]

1 AnyOneFromA Dhaka CSE 77
2 AnyOneFromB Dhaka CSE 44
3 AnyOneFromC Bogra CSE 35
4 AnyOneFromD Rajshahi CSE 82
5 AnyOneFromE Dhaka EEE 76
6 AnyOneFromF Bogra EEE 79
7 AnyOneFromZ Dhaka EEE 39
20 Sagor Dhanmondi CSE201 66
```

Figure 3.13: Student text file

3.2.14 Result_portion_14

Comments.

```
=====
Student Management System
=====
1. Create Student
2. Edit Student
3. Delete Student
4. View Students
5. Search Student by ID
6. Search Student by Name
7. View Statistics
8. Sort Students
9. Add Comment
10. Backup Data
11. Restore Data
12. Exit
Enter your choice:
9
Enter student ID to add comment:
20
Enter comment:
your name is sagor
Comment added successfully.
Do you want to clear the screen? (y/n):
```

Figure 3.14: Comments

3.2.15 Result_portion_15

Student text file after comment.

```
Jun 8 15:52
students.txt
~/Documents
Open v [icon]
1 AnyOneFromA Dhaka CSE 77
2 AnyOneFromB Dhaka CSE 44
3 AnyOneFromC Bogra CSE 35
4 AnyOneFromD Rajshahi CSE 82
5 AnyOneFromE Dhaka EEE 76
6 AnyOneFromF Bogra EEE 79
7 AnyOneFromZ Dhaka EEE 39
20 Sagor Dhanmondi CSE201 66 your name is sagor
```

Figure 3.15: Student text file after comments

3.2.16 Result_portion_16

A backup file is created.

```
=====
                        Student Management System
=====
1. Create Student
2. Edit Student
3. Delete Student
4. View Students
5. Search Student by ID
6. Search Student by Name
7. View Statistics
8. Sort Students
9. Add Comment
10. Backup Data
11. Restore Data
12. Exit
Enter your choice:
10
Backup created as students_backup.txt.
Do you want to clear the screen? (y/n):
```

Figure 3.16: Create a backup file

3.2.17 Result_portion_17

Backup text file.

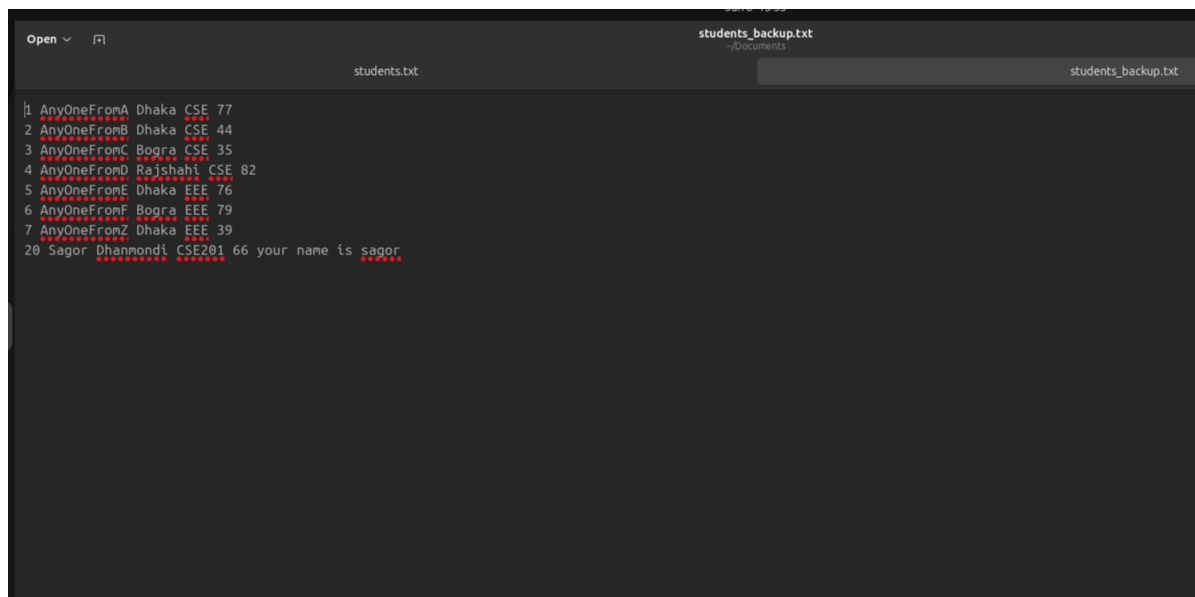


Figure 3.17: Backup file

3.2.18 Result_portion_18

Sagor details are deleted

```
=====
Student Management System
=====
1. Create Student
2. Edit Student
3. Delete Student
4. View Students
5. Search Student by ID
6. Search Student by Name
7. View Statistics
8. Sort Students
9. Add Comment
10. Backup Data
11. Restore Data
12. Exit
Enter your choice:
3
Enter student ID to delete:
20
Student deleted successfully.
Do you want to clear the screen? (y/n):
█
```

Figure 3.18: Delete a details

3.2.19 Result_portion_19

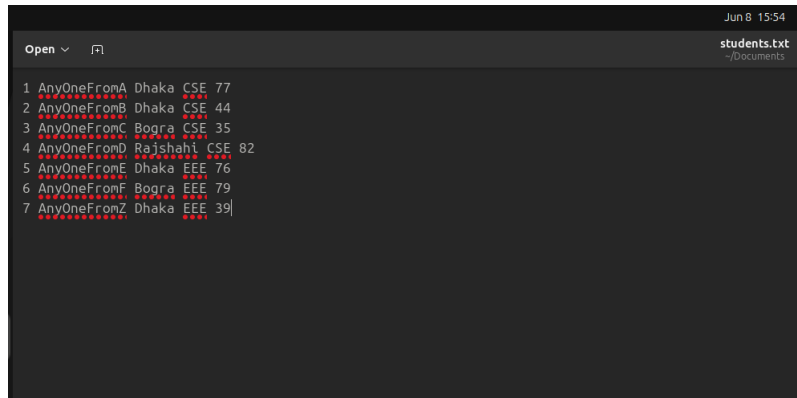
View Student details after deleting Sagor

```
=====
Student Management System
=====
1. Create Student
2. Edit Student
3. Delete Student
4. View Students
5. Search Student by ID
6. Search Student by Name
7. View Statistics
8. Sort Students
9. Add Comment
10. Backup Data
11. Restore Data
12. Exit
Enter your choice:
4
===== Student List =====
ID      Name      Address      Course      Marks      Grade
1      AnyOneFromA      Dhaka      CSE      77      A
2      AnyOneFromB      Dhaka      CSE      44      D
3      AnyOneFromC      Bogra      CSE      35      F
4      AnyOneFromD      Rajshahi      CSE      82      A+
5      AnyOneFromE      Dhaka      EEE      76      A
6      AnyOneFromF      Bogra      EEE      79      A
7      AnyOneFromZ      Dhaka      EEE      39      F
Do you want to clear the screen? (y/n):
```

Figure 3.19: Details after deleted

3.2.20 Result_portion_20

View the text file to check whether Sagor is deleted or not



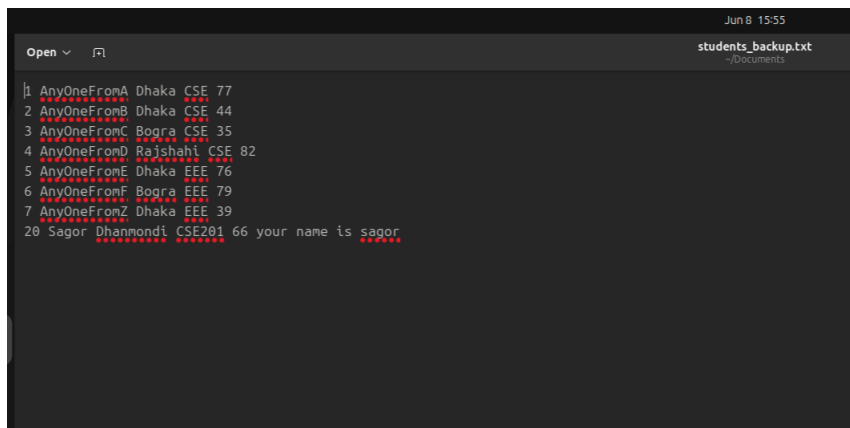
```
Open ▾  Jun 8 15:54  students.txt
~/Documents

1 AnyOneFromA Dhaka CSE 77
2 AnyOneFromB Dhaka CSE 44
3 AnyOneFromC Bogra CSE 35
4 AnyOneFromD Rajshahi CSE 82
5 AnyOneFromE Dhaka EEE 76
6 AnyOneFromF Bogra EEE 79
7 AnyOneFromZ Dhaka EEE 39
```

Figure 3.20: View text file after deleted Sagor

3.2.21 Result_portion_21

View the backup text file to check whether Sagor has been deleted or not and whether Sagor is not deleted from the backup.



```
Open ▾  Jun 8 15:55  students_backup.txt
~/Documents

1 AnyOneFromA Dhaka CSE 77
2 AnyOneFromB Dhaka CSE 44
3 AnyOneFromC Bogra CSE 35
4 AnyOneFromD Rajshahi CSE 82
5 AnyOneFromE Dhaka EEE 76
6 AnyOneFromF Bogra EEE 79
7 AnyOneFromZ Dhaka EEE 39
20 Sagor Dhanmondi CSE201 66 your name is sagor
```

Figure 3.21: View backup text file after deleted Sagor

3.2.22 Result_portion_22

Restorer details from the backup.

```
=====
Student Management System
=====
1. Create Student
2. Edit Student
3. Delete Student
4. View Students
5. Search Student by ID
6. Search Student by Name
7. View Statistics
8. Sort Students
9. Add Comment
10. Backup Data
11. Restore Data
12. Exit
Enter your choice:
11
Data restored from students_backup.txt.
Do you want to clear the screen? (y/n):
█
```

Figure 3.22: Restore

3.2.23 Result_portion_23

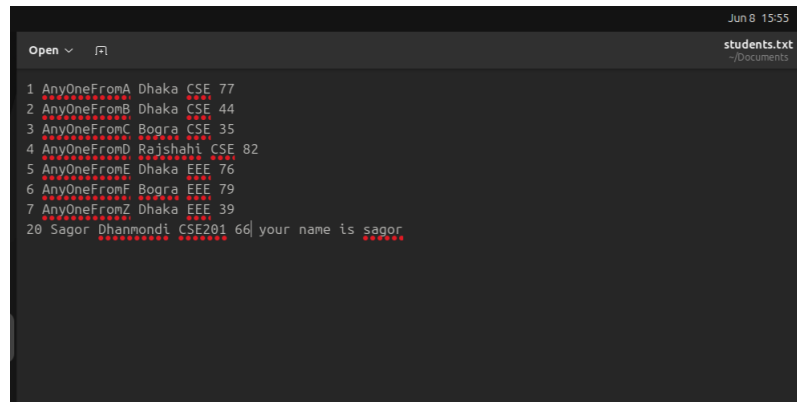
View details after restoring to check whether Sagor is back or not and Sagor is back.

```
=====
Student Management System
=====
1. Create Student
2. Edit Student
3. Delete Student
4. View Students
5. Search Student by ID
6. Search Student by Name
7. View Statistics
8. Sort Students
9. Add Comment
10. Backup Data
11. Restore Data
12. Exit
Enter your choice:
4
===== Student List =====
ID      Name      Address      Course      Marks      Grade
1      AnyOneFromA      Dhaka      CSE      77      A
2      AnyOneFromB      Dhaka      CSE      44      D
3      AnyOneFromC      Bogra      CSE      35      F
4      AnyOneFromD      Rajshahi      CSE      82      A+
5      AnyOneFromE      Dhaka      EEE      76      A
6      AnyOneFromF      Bogra      EEE      79      A
7      AnyOneFromZ      Dhaka      EEE      39      F
20      Sagor      Dhanmondi      CSE201      66      B+
Do you want to clear the screen? (y/n):
```

Figure 3.23: View after restore

3.2.24 Result_portion_24

View the text file after restoring to check whether Sagor is back or not and Sagor is back.



```
Jun 8 15:55
students.txt
~/Documents

1 AnyOneFromA Dhaka CSE 77
2 AnyOneFromB Dhaka CSE 44
3 AnyOneFromC Bogra CSE 35
4 AnyOneFromD Rajshahi CSE 82
5 AnyOneFromE Dhaka EEE 76
6 AnyOneFromF Bogra EEE 79
7 AnyOneFromZ Dhaka EEE 39
20 Sagor Dhanmondi CSE201 66 your name is sagor
```

Figure 3.24: View text file after restore

3.3 Results Overall Discussion

The performance evaluation of the Student Management System revealed several noteworthy insights. The system demonstrated efficient handling of student records, with acceptable execution times for common operations such as creating, editing, and deleting records. Additionally, the system showcased robustness in handling diverse datasets, with consistent performance across varying sizes and complexities of data. Stress testing highlighted areas for improvement under high load conditions, prompting optimizations to enhance scalability and resource utilization. Overall, the results underscore the system's reliability and effectiveness in managing student information within educational institutions. The findings provide valuable guidance for further refinement and optimization, ensuring continued performance excellence and user satisfaction.

Chapter 4

Conclusion

4.1 Discussion

The development of the Student Management System using shell scripting represents a pragmatic approach to addressing administrative needs within educational institutions. Through the implementation of various functionalities, including student record management, statistical analysis, and data backup, the system offers a comprehensive solution for organizing and accessing student information.

One of the project's notable strengths lies in its simplicity and accessibility. By leveraging Bash scripting, a widely supported language in Unix-based environments, the system can be easily deployed and utilized across diverse computing platforms without the need for additional dependencies. This portability enhances the system's accessibility and ensures compatibility with existing infrastructure within educational institutions.

Furthermore, the system's modular design facilitates seamless expansion and customization. Each functionality, from creating and editing student records to viewing statistics and sorting students, is encapsulated within discrete functions, promoting code re-usability and maintainability. This modular architecture enables educators and administrators to tailor the system to suit specific institutional requirements, thereby enhancing its flexibility and adaptability.

However, despite these strengths, the project also presents certain limitations and areas for improvement. The reliance on shell scripting, while advantageous for its simplicity, may impose constraints on the system's scalability and performance, particularly when dealing with large datasets or complex operations. Additionally, the absence of a graphical user interface (GUI) may pose usability challenges for non-technical users, necessitating familiarity with command-line interfaces.

Moreover, while the system adequately addresses basic administrative tasks related to student management, it may lack advanced features found in commercial or web-based solutions, such as integration with learning management systems or automated communication capabilities. Future iterations of the project could explore the integration of additional functionalities to enhance productivity and streamline administrative workflows further.

4.2 Limitations

Despite its utility and effectiveness, the Student Management System developed using shell scripting exhibits several inherent limitations. Firstly, due to its reliance on command-line interaction, the system may present usability challenges for users unfamiliar with shell environments, potentially hindering widespread adoption among non-technical stakeholders. Moreover, while the system adequately addresses basic administrative tasks, its functionality may be limited compared to commercial or web-based solutions, lacking advanced features such as real-time collaboration, integration with external systems, or automated notifications. Additionally, the performance of the system may degrade when handling large datasets or executing complex operations, as shell scripting may not offer optimal efficiency or scalability in such scenarios. Furthermore, the absence of a graphical user interface (GUI) may impede user interaction and accessibility, particularly for users accustomed to intuitive visual interfaces. Despite these limitations, the project serves as a valuable demonstration of the capabilities of shell scripting in addressing administrative needs within educational institutions, albeit with scope for further refinement and enhancement to overcome its inherent constraints.

4.3 Scope of Future Work

The scope of future work for the Student Management System encompasses several key areas of improvement and expansion. Firstly, the introduction of a Graphical User Interface (GUI) would enhance usability, making the system more accessible to a wider range of users. Integration with external systems such as learning management systems (LMS) or student information systems (SIS) could extend functionality and facilitate seamless data exchange. Enhanced reporting and analytics capabilities would provide deeper insights into student performance and trends, enabling data-driven decision-making. Strengthening security measures and implementing access controls would ensure data privacy and compliance. Optimizing performance for scalability and responsiveness under varying workloads is essential for accommodating the evolving needs of educational institutions. Additionally, the development of a mobile application version could further enhance accessibility and convenience for users. Continuously soliciting user feedback and iterating on improvements would be crucial for refining features and addressing user needs effectively. By pursuing these avenues, the Student Management System could evolve into a comprehensive and adaptable solution that supports educational excellence.