“DSA Final Project”

Submitted to: Mr. Shakeel Ahmad

BSSE F-23 Section B

TEAM:-

Members	Reg No
Faizan Ul Hadi	4823-FOC/BSSE/F23
Sahal Bin Amin	4804-FOC/BSSE/F23
Naqeeb Ur Rehman	4786-FOC/BSSE/F23
Kashan Haider	4788-FOC/BSSE/F23

Date: 29 May, 2025

📘 Student Management System

“ Project Documentation ”

1)Introduction & Objectives:-

The *Student Management System* is a console-based C++ application designed to handle student records effectively. It supports core operations such as adding, deleting, updating, searching, sorting, displaying, undoing actions, and saving data persistently.

The objective was to develop a modular, efficient, and user-friendly system that demonstrates the practical application of data structures and algorithms.

2)Data Structures & Algorithms Used:-

- Binary Search Tree (BST):
  - Used for managing roll number-based search and sorting.
  - Enables efficient O(log n) average-case search, insert, and delete operations.
  - Inorder traversal provides sorted output by roll number.
- Doubly Linked List (DLL):
  - Used for CGPA-related operations (search and sorting).
  - Allows forward and backward traversal and easier insertion/deletion.
- Stack (Custom Implementation):
  - Enables an *undo* feature by storing previous states.
  - Implemented via a DLL-style structure to manage history.
- Sorting Algorithms:
  - Inorder BST Traversal (for sorting by roll number).
  - Bubble Sort (for sorting by CGPA).
- Search Algorithms:
  - Binary Search (implicitly through BST).
  - Linear Search (for CGPA using DLL).

3)Implementation Overview:-

➤ **Main Menu Interface (Console):**

```
===== STUDENT MANAGEMENT SYSTEM =====
1. Add Student
2. Delete Student
3. Update Student
4. Search Student
5. Sort Students
6. Display All Students
7. Undo Last Operation
8. clear file
9. Exit
```

➤ **Add Student (BST + DLL Insertion):**

```
Enter your choice: 1
Enter Roll Number: 4786
Enter Name: naqeeb
Enter CGPA: 3.5
Enter Department: BSSE
Enter Gender (male/female): male
Enter Contact Number: 0326
Enter Semester: 4
Enter Email: naqeeb@gmail.com
```

➤ **Search by Roll Number (BST) And Search by CGPA (DLL):**

```
--- Search Menu ---
1. Search by Roll Number
2. Search by CGPA
3. Return to Main Menu
Enter choice:
```

➤ **Update Feature:**

```
Enter the roll number to update: 4786

--- Student Details ---
Roll Number: 4786
Name: naqeeb
CGPA: 3.5
Department: BSSE
Gender: male
Contact No: 0326
Semester: 4
Email: naqeeb@gmail.com
Enter new details:
Enter Roll Number:
```

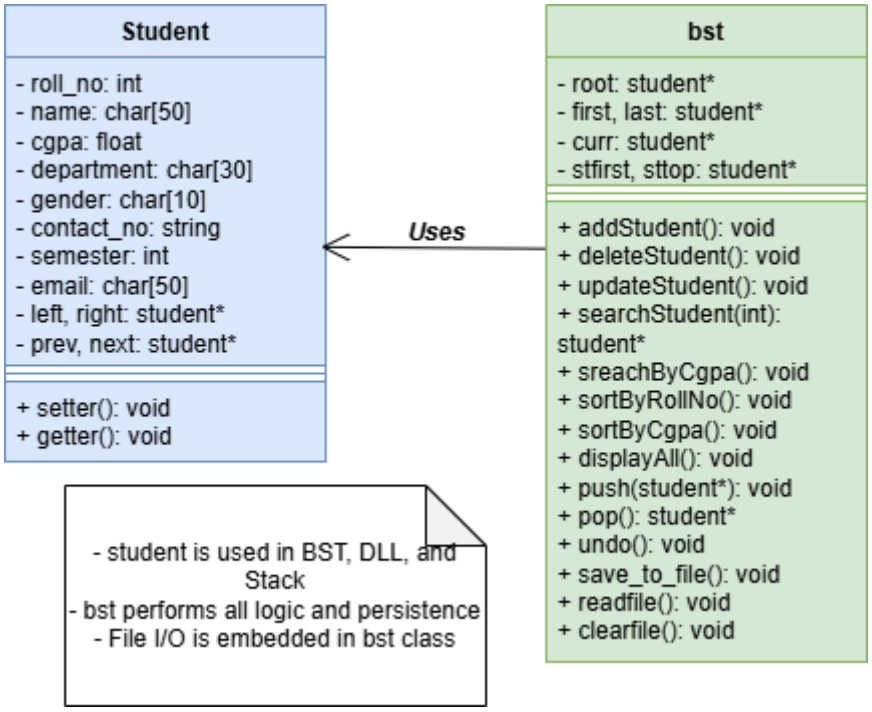
➤ **Undo Feature (Custom Stack):**

```
--- Student Details ---
Roll Number: 21
Name: sa
CGPA: 2
Department: s
Gender: male
Contact No: 1
Semester: 2
Email: f
Undo successful.
Profile saved to file.
```

➤ **Delete Feature :**

```
Enter roll number to delete:
```

3)UML Class Diagram:-



Explaination:

The UML class diagram above represents the core structure of the Student Management System.

- The Student structure models the data for each student. It contains fields like roll number, name, CGPA, department, and contact details. Additionally, it includes pointers that allow each student to be linked into different data structures:
    - left and right pointers for organizing students in a Binary Search Tree (BST) for roll number-based operations.
    - prev and next pointers for organizing students in a Doubly Linked List (DLL) for CGPA-related tasks.
- The bst class encapsulates all the functionality and logic of the system, including:
    - Core operations such as adding, updating, deleting, and searching student records.
    - Sorting based on roll number (using BST inorder traversal) and CGPA (using Bubble Sort on DLL).
    - An undo feature implemented using a custom stack based on a linked list.
    - Persistent file handling, allowing student data to be saved and loaded across sessions.
- The relationship between these classes is represented as a "Uses" dependency: the bst class heavily uses the Student structure to build its data structures (BST, DLL, and Stack).

This modular design makes the system flexible, efficient, and maintainable, clearly separating data representation (Student) from operational logic (bst).

5)Performance Analysis & Optimization:-

Feature	Approach	Time Complexity	Remarks
Add/Search/Delete	BST	O(log n) avg	Efficient for large datasets
Sort by Roll No	Inorder Traversal	O(n)	Linear for already built BST
Sort by CGPA	Bubble Sort	$O(n^2)$	Acceptable for small datasets
Search by CGPA	Linear Search	O(n)	Simpler logic, flexible traversal
Undo Feature	Custom Stack	O(1) push/pop	Efficient and modular

6) Member-wise Contributions:-

Members	Reg No	Contribution
Faizan Ul Hadi	4823-FOC/BSSE/F23	ADD and UPDATE Module
Sahal Bin Amin	4804-FOC/BSSE/F23	DELETE Module
Naqeeb Ur Rehman	4786-FOC/BSSE/F23	SEARCH and Sort Module
Kashan Haider	4788-FOC/BSSE/F23	DISPLAY, FILE I/O, UNDO Module

7)LINKS :-

➤ **Git hub Repository** [https://github.com/Naqeeb-00/Student-Management-System]

➤ **LinkedIn Post** [https://www.linkedin.com/posts/naqeeb-ur-rehman-27b357351_cplusplus-datastructures-studentproject-activity-7333218073346240512-Cro3?utm_source=social_share_send&utm_medium=member_desktop_web&rcm=ACoAAFfLeb8BpsqpEalUKiXW4feGu1JCR_-Dcvo]

8)Conclusion:-

This project demonstrates a practical and educational use of key data structures in managing complex datasets. By integrating a Binary Search Tree for roll number-based operations and a Doubly Linked List for CGPA-based actions, the system achieves a balance between speed and functionality. Additionally, features like undo, file persistence, and sorting enhance the system's robustness.

Through this project, we not only applied our knowledge of data structures and algorithms but also strengthened our understanding of system design, file I/O, and modular programming. It serves as a solid foundation for more advanced management systems and real-world applications.