Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre

A Project Report
on

“Rank Master”

[Code No : COMP 202]

(For partial fulfillment of II/I Year/Semester in Computer Science/Engineering)

Submitted by

Aayam Pokharel(37)(037992-24)

Submitted to

Mr.Sagar Aacharya

Department of Computer Science and Engineering

Submission Date:2/20/2026

Abstract

This mini project is developed in an attempt to overcome the challenges of academic performance evaluation and ranking management in educational environments. This project presents Rank Master, a high-performance Student Ranking System that leverages the core concepts of Data Structures and Algorithms (DSA) to enhance the efficiency of student grading and data management. The system employs the Max-Heap data structure (implemented via a priority queue) primarily to calculate and maintain real-time rankings based on a weighted multi-variable formula involving GPA, Marks, and Attendance. By integrating a standalone C++ web server using the WinSock2 API, the project provides a seamless user interface without the need for external dependencies or traditional web servers. The system demonstrates how theoretical DSA principles, such as heap-based priority management and O(log n) insertion complexity, can be applied to solve real-world organizational challenges while providing a modern, responsive user experience through an embedded glass morphism UI.

Keywords: Data Structures and Algorithms, Max-Heap, Priority Queue, WinSock2, Web Server Development, Academic Ranking, Performance Optimization, C++, Real-time Data Processing, Weighted Scoring Systems.

Acknowledgement

I would like to express my sincere gratitude to Mr.Sagar Aacharya and the teaching staff of the Department Of Computer Science and Engineering for their invaluable guidance, encouragement, and constructive feedback throughout the development of this mini project on Data Structures and Algorithms. I am also thankful to my institution for providing the resources and a supportive environment that enabled me to explore and apply theoretical concepts in practical implementation. My heartfelt appreciation goes to my peers and friends for their collaborative discussions and motivation, which enriched my learning experience. Finally, I am deeply grateful to my family for their constant support and encouragement, which kept me motivated to successfully complete this project.

Table of Contents

List of Figures

Acronyms/Abbreviations

DSA	Data Structures and Algorithms
RAM	Random Access Memory

Chapter 1 **Introduction**

Rank Master is a lightweight, standalone web-based application designed to streamline academic performance evaluation and real-time student ranking. Efficient assessment is essential in modern education, and accurate data processing ensures transparency, fairness, and timely feedback. This mini project applies core Data Structures and Algorithms (DSA) concepts to build a priority-based ranking system that enhances both efficiency and reliability in managing student performance data.

1.1 Background

In today's competitive academic environment, evaluating and ranking students often involves manual calculations, spreadsheet dependency, or slow legacy systems. These approaches are prone to errors, delays, and limited scalability. By leveraging DSA principles, particularly efficient sorting and priority handling, this system minimizes computational complexity and ensures that top-performing records are always quickly accessible. The project demonstrates how structured data management can replace traditional, error-prone methods with a dynamic and optimized solution.

1.2 Objectives

- Demonstrate the practical application of DSA concepts in solving real-world ranking problems.
- Design and implement a priority-based student evaluation model using a Max Heap data structure.
- Develop a weighted scoring algorithm based on GPA, marks, and attendance.
- Utilize a Priority Queue for dynamic storage and automated sorting of student records.

1.3 Motivation and Significance

The project was motivated by the need to eliminate delays, inconsistencies, and manual errors in academic result processing. Traditional ranking systems often lack transparency and real-time capabilities. By integrating a web-based interface with a Max-Heap driven ranking mechanism and weighted evaluation logic, Rank Master provides a fair, efficient, and scalable solution.

Chapter 2 **Related Works**

- **Student Management System**

Traditional SMS platforms utilize the LAMP stack or Java-based frameworks to manage large-scale institutional data through centralized relational databases like MySQL. The architecture focuses on CRUD operations to maintain extensive student profiles, grades, and attendance records, replacing manual ledgers with automated digital storage. These designs are effective for long-term data integrity and generating comprehensive semester reports across multiple departments(Al-Mahmood & Ahmed, 2021; Sharma & Kumar, 2020).

Chapter 3 Design and Implementation

The design and implementation of this project was planned meticulously and was initiated so that every step was precise and meaningful to the proper development of the system.

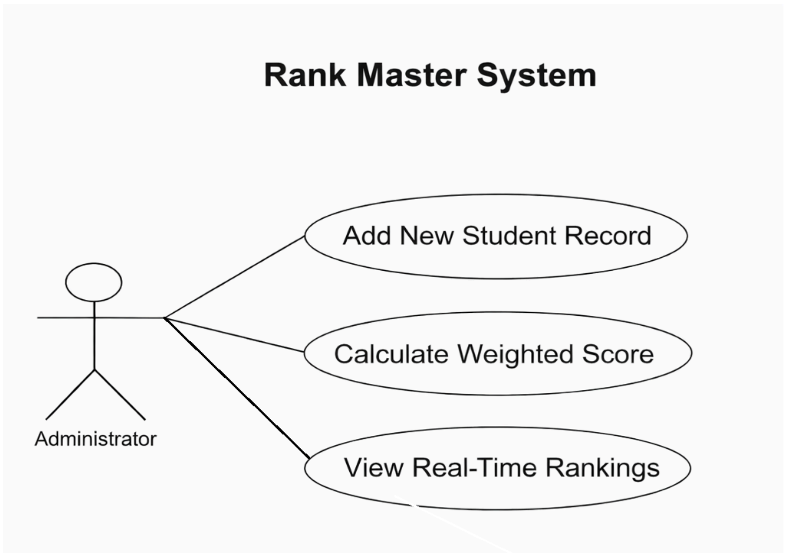


Fig 3.1:Use case diagram

3.1 System Requirement Specifications

This system was developed to solve the real-world problem of academic performance evaluation and ranking using C++ and the WinSock2 networking framework. The system provides an efficient, web-based platform to calculate real-time student standings using a weighted priority model and ensures data safety through persistent file logging.

3.1.1 Software Specifications

The following points depict the software requirements of the proposed system.

- **Functional Requirements:**

a. **Ranking Engine:** The system shall implement a Max-Heap data structure to maintain a real-time leaderboard of student performance.
b. **Weighted Calculation:** The system shall compute a final score for each student based on a weighted formula of GPA (70%), Marks (20%), and Attendance (10%).
c. **Web Interface:** The system shall serve a responsive HTML/CSS interface via a standalone C++ web server for user interaction.
d. **Data Persistence:** The system shall automatically append and store all student records into a local text file (students_records.txt) using file I/O.

- **Non-Functional Requirements:**
  a. **Performance**: The system shall provide O(logn) time complexity for student insertion and O(1) for accessing the top-ranked student.
  b. **Reliability**: The system shall ensure data integrity by committing records to non-volatile storage immediately upon entry.
  c. **Usability**:The system shall offer a modern Glassmorphism-style UI that is intuitive for administrators without technical expertise.
  d. **Portability**:The system shall run on any standard Windows environment that supports the WinSock2 library.

3.1.2 Hardware Specifications

The following hardware components are necessary for the system to function.

- **Processor:**The system shall require a minimum of a dual-core processor (Intel i3, AMD Ryzen 3, or equivalent) to handle concurrent socket connections and heap sorting.
- **Memory (RAM):** The system shall require at least 2 GB of RAM to efficiently manage the in-memory priority queue and string-based HTML generation.
- **Storage:** The system shall require at least 100 MB of free disk space for the executable and the growing persistent text database.
- **Input Devices:**The system shall require a standard keyboard and mouse for entering student metrics and navigating the web interface.

Chapter 4 Discussion on the achievements

The Rank Master system was developed to improve the administrative efficiency of academic performance evaluation. While Max-Heap modeling made real-time ranking computation efficient, scaling to massive institutional datasets could introduce memory overhead. Future iterations could explore distributed data structures or persistent database integration like SQLite. The project shows how algorithmic sorting can support fair and transparent academic management, reducing manual errors and processing time. Rank Master achieved its core objectives: delivering high-speed ranking computation, a modern web-based user interface, and measurable improvements in data organization and persistence.

4.1 Features:

1. Efficient Ranking Optimization

The system uses Max-Heap data structure modeling (via priority_queue) to compute student standings instantly. This ensures $O(\log n)$ insertion time and $O(1)$ access to the top-performing student. As a result, administrators enjoy real-time updates and students receive immediate feedback on their academic standing.

2. Multi-Variable Weighted Evaluation

Rankings are determined by a sophisticated weighted formula rather than simple averages. By factoring in GPA (70%), Marks (20%), and Attendance (10%), the system provides a holistic view of student performance. This balance ensures that long-term consistency is rewarded alongside test results.

3. Persistent Data Storage

The system includes a dedicated data persistence module that ensures all student records are preserved. By utilizing C++ file streams, every entry is automatically logged into a permanent text file (students_records.txt). This mechanism prevents data loss during system restarts, power interruptions, or unexpected application closures, ensuring long-term data integrity.

4. Scalability and Future Extensions

The backend is powered by a high-performance WinSock2 server, making the application a portable, single-executable tool. It requires no external web servers or database installations. Future upgrades may include multi-classroom support, automated report generation in PDF format, and historical trend analytics for individual student growth.

Chapter 5 Conclusion and Recommendation

The system ‘Rank Master’ was developed as a mini project to apply the knowledge of Data Structures and Algorithms (DSA). Its core was able to achieve the primary objective of designing and implementing a priority-based model of student performance. The system successfully utilizes a Max-Heap algorithm (via priority_queue) to compute academic rankings instantly based on a weighted multi-variable formula. Furthermore, the system employs WinSock2 for establishing a standalone web server and utilizes C++ File Streams (fstream) for the dynamic storage and permanent management of student records. This project demonstrates how complex sorting logic and networking can be integrated into a single-executable tool for educational administration.

5.1 Limitations

- The system may experience high memory usage when processing exceptionally large student datasets in real-time.
- Using a plain text file for storage lacks the advanced data searching and indexing features of a professional database.
- All student data must be entered manually into the web form as there is currently no support for bulk file uploads.

5.2 Future Enhancement

- Integration of SQLite or MySQL to replace text files, allowing for more efficient data searching and management of larger datasets.
- Automate module to generate and download professional PDF report cards and ranking certificates.
- Implementation of performance analytics to track student progress over time and predict future academic trends using historical data.

References

1. GeeksforGeeks. (2024, May 24). C program to implement priority queue. https://www.geeksforgeeks.org/c/c-program-to-implement-priority-queue/

2. Pimpalkar, A. (2025, October 10). Student data management systems: Transforming campus administration with vmedulife. vmedulife. https://vmedulife.com/blog/academic-planning/student-data-management-systems-transforming-campus-administration-with-vmedulife/

APPENDIX

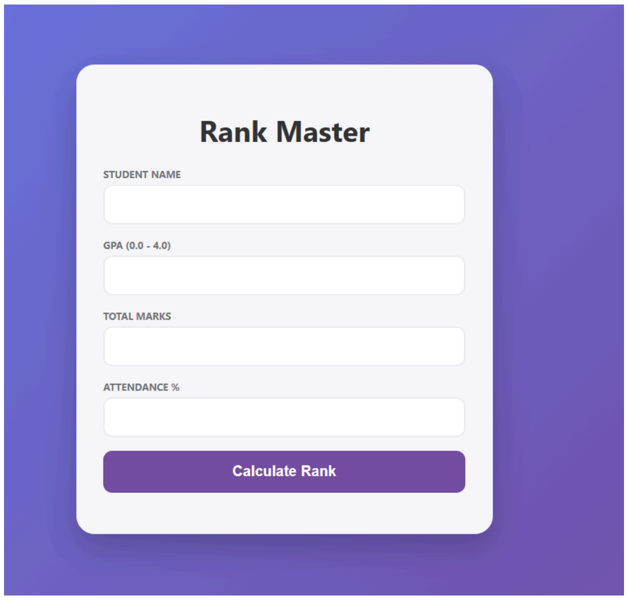
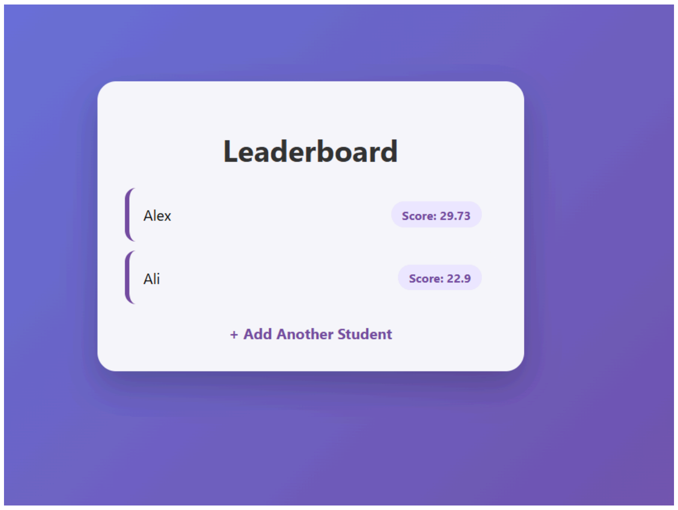


Fig 5.1: Page for Student's Detail



Alex	Score: 29.73
Ali	Score: 22.9

Fig 5.2:Page for showing Rank