# MeruLite School Management System

Course: Data Structures and Algorithms
Instructor: Mr. Dismas Kitaria
Semester: 2024/2025

## Group Project Report

The MeruLite School Management System (SMS) is a modular C++ prototype designed to demonstrate the application of various data structures in solving real-world academic administration problems. The system handles student registration, course scheduling, fee tracking, library management, and performance analytics. Each module is implemented using an appropriate data structure to ensure efficient operations.

**1. Architecture Overview**
The MeruLite SMS is designed with five main modules, each interacting through a central controller. Data is stored in-memory for this prototype. Each module encapsulates its own data structure and provides well-defined interfaces for CRUD (Create, Read, Update, Delete) operations.

• Student Registry – manages student data using a Hash Table (unordered_map).
• Course Scheduler – handles course enrollments using a Circular Queue.
• Fee Tracker – maintains fee records using an AVL Tree.
• Library System – tracks book borrowing and availability using a Stack and Hash Map.
• Performance Analytics – stores and analyzes marks using a Matrix and Max Heap.

**2. Data Structure Justification**
Each data structure was selected to optimize time and space complexity for its module:

• **Hash Table**: Enables O(1) average lookup for students by ID.
• **Circular Queue**: Ensures fair, first-come-first-served course registration.
• **AVL Tree**: Keeps fee records balanced and sorted with O(log n) operations.
• **Stack + Hash Map**: Allows quick borrow/return operations and availability checks.
• **Matrix + Heap**: Efficiently stores and ranks student marks for analytics.

**3. Performance Analysis**
The table below summarizes time and space complexity of core operations:

Student lookup (Hash Table): O(1) time, O(n) space.
Course queue operations (Circular Array): O(1) time, O(n) space.
Fee insertion/search (AVL Tree): O(log n) time, O(n) space.
Book borrow/return (Stack/Hash Map): O(1) time, O(n) space.
Top performers (Heap): O(n log n) build, O(k log n) for top-k extraction.

**4. Ethical Reflection**

• **Fairness:** Course allocation follows a queue model ensuring fairness in registration order.
• **Privacy:** Prototype data is stored in-memory; production systems must encrypt data and use access control.
• **Transparency:** All operations are logged and explainable to stakeholders.
• **Data Minimization:** Only essential data fields are collected for system functionality.
• **Security:** Prototype assumes trusted input. Future enhancements should include authentication and secure file I/O.

**5. Conclusion**

The MeruLite School Management System demonstrates the practical integration of five major data structures within a modular design. Each structure contributes to the system's efficiency, scalability, and reliability. This project showcases the importance of algorithmic thinking and ethical responsibility in software engineering.