GROUP ASSIGNMENT

(MAX 4 MEMBERS PER GROUP)

Assignment Brief

Title: School Management System — Data Structures in Action
Course: Data Structures and Algorithms
Semester: Sem1-2024/2025
Instructor: Dismas Kitaria
Due Date: 24/10/2025

Objective

You are required to design and implement a **modular School Management System** that solves real-world problems using **at least five different data structures**. Your system should reflect thoughtful design, efficient algorithms, and ethical considerations in handling student and institutional data.

Scenario

Meru University is modernizing its internal systems. As a systems architect, your task is to prototype a School Management System that supports:

- ✅ Student registration and lookup
- ✅ Course allocation and scheduling
- ✅ Fee tracking and reporting
- ✅ Library book management
- ✅ Performance analytics

Each module must use a **specific data structure** that best suits its operational needs. You are expected to justify your choices and demonstrate how they improve system performance and usability.

Required Modules & Data Structures

Module	Suggested Data Structure(s)	Example Task
Student Registry	Hash Table / Linked List	Quickly find a student by ID, add new students, or remove graduates.
Course Scheduling	Queue / Circular Array	Allocate students to courses based on registration order and course capacity.

Module	Suggested Data Structure(s)	Example Task
Fee Tracking	Binary Search Tree / AVL Tree	Maintain sorted records of payments and generate fee clearance reports.
Library System	Stack / Hash Map	Track book borrowing and returns, and check availability by ISBN.
Performance Analytics	Graph / Matrix / Heap	Analyze student performance across subjects and identify top performers.

Deliverables

1. System Design Document

Provide a comprehensive design blueprint:

- **Architecture Overview:** Describe how modules interact and share data.
  Example: Use a central controller to manage requests between modules.
- **Data Structure Justification:** Explain why each structure was chosen.
  Example: A hash table allows constant-time student lookup.
- **Flow Diagrams or Pseudocode:** Visualize logic and data flow.
  Example: Show how a student is added to the registry and enrolled in a course.

---

2. Code Implementation

Develop a working prototype:

- **Modular Code:** Organize code into separate files or classes per module.
- **Comments & Documentation:** Explain logic clearly for each function.
- **Sample Data:** Include mock student records, course lists, fee transactions, etc.
  Tip: Use JSON, CSV, or hardcoded arrays for testing.

---

3. Performance Report

Analyze how well your system performs:

- **Time Complexity:** Use Big-O notation to describe operations.
  Example: Searching a student in a hash table is O(1).
- **Space Complexity:** Estimate memory usage for each structure.
- **Trade-offs:** Discuss alternatives and why your choices are optimal.
  Example: A BST offers sorted output but slower lookup than a hash table.

---

4. Ethical Reflection

Reflect on the ethical dimensions of your system:

- **Fairness:** Does your course allocation system treat all students equally?
- **Privacy:** How is student data protected from unauthorized access?
- **Transparency:** Can users understand how decisions are made?

Submission Checklist

Before submitting, ensure you have:

- [ ] A complete design document
- [ ] Functional code with sample data
- [ ] A performance analysis section
- [ ] A thoughtful ethical reflection