Project Guideline

This project requires each group to design and develop a complete system using the C++ programming language. The primary goal is to assess students' understanding of core algorithmic concepts and their ability to apply these concepts effectively in a real-world system. Student's system should simulate a realistic business or functional application, integrating key algorithms to perform meaningful tasks.

Project Objectives

- Demonstrate understanding of core data structures and algorithm behavior.
- Learn how to manually implement algorithms within a functioning system.
- Gain practical experience in applying algorithms to real-world scenarios.

Important Guidelines for Algorithm Implementation

- Do not use any built-in functions or libraries for algorithm implementation.
- All code must be original, written manually, and clearly visible in the source file.

Restricted Items (Include but are not limited to):

- #include <algorithm>
- #include <stack>
- #include <queue>
- Standard Library sorting/searching functions like std::sort()
- Use of std::vector

Instead, rely on raw arrays, pointers, and your own logic for implementing data structures and algorithms.

1. Project Title Registration Guidelines

a) The group leader is responsible for registering group members using the link provided by the lecturer in the registration form.
  - ✓ Ensure that the assignment title is clearly written.
  - ✓ Obtain approval from your lecturer before proceeding.
  - ✓ Titles must be unique and not identical to those of other groups.

b) Submission Deadline: **18 April 2025**

c) Group Size: Maximum of 4 members per group.

Only 1 or 2 groups may consist of 3 members, but this is subject to the lecturer's approval.

2. Project Submission Guidelines (Softcopy).

Submission by Group Leader Only:
The following files must be submitted in softcopy format:

1. Project Evaluation Form
    - File Name Format: Project _Evaluation_GR1.xlsx
2. Project_G#_evaluatorname_peer evaluation
    - Each student must complete and submit this peer evaluation form to the group leader.
    - The group leader will compile all peer evaluations and copy them into the assignment Evaluation file.
3. Report File

   - File Name Format: Project_Report_GR1.pdf
4. C++ Source File
   - File Name Format: Project _GR1.cpp
   - File must be executable using Dev C++ only
5. 5 Original Text Files (.txt)
6. README File
   - Instructions for running the program or logging into the system must be clearly included.

Submission Deadline: **Week 14**

3. Project Descriptions:

In this project, you are required to develop a mini-information system that consists of a collection of records. You should provide at least 50 records with a minimum of **2300 code lines**. Exclude empty lines and comments exist in the code. Each group leaders, shall create a GitHub account and add in your lecturer as a collaborator.

1. General Requirements

- Your system must be written in C++ and follow good coding practices.
- Create a GitHub repository for version control.
  - The group leader must create the repository and add group members and the lecturer as collaborators.
- You may include additional features or algorithms.
  - Refer to the Project Evaluation Form for bonus criteria or extra functions.

2. Core System Requirements

Your system must include two main modules:

Module	Description
Customer	Regular user functionalities
Staff/Admin	Administrative and management tasks

- **Required Functionalities (For Both Modules)**

Function	Staff Module	Customer Module
Registration	✅	✅
Login & Logout	✅	✅
Add New Record	✅ *(At least 2 records)*	✅ *(At least 2 record)*
Edit/Update Record	✅ *(2 records)*	✅ *(2 records)*
Display Record	✅ *(2 records)*	✅ *(2 records)*
Search Record	✅ *(2 records)*	✅ *(2 records)*

Sort Records	✅ *(2 sorting criteria)*	✅ *(2 sorting criteria)*
Delete Record	✅	✅
Summary Report	✅ *(to be displayed and saved in a txt file. To be displayed and retrieve from txt file)*	✅ *(to be displayed and saved in a txt file. To be displayed and retrieve from txt file)*

Examples of System Interactions

Operation	Staff/Admin	Customer
Add Record	Add a new product	Make a new booking
Edit Record	Edit product details	Edit booking info
Sort Record	Sort products by price or category	Sort bookings by date or name
Search Record	Search products	Search bookings
Display Record	Show all or filtered data	View past or active bookings
Report	Display the sales report for the Product of the Month – February. Sales data is stored in a .txt file.	Display a summary of bookings and allow it to be saved in a .txt file. The system should also be able to retrieve and display the booking summary from the saved file.

Record Consistency:
All record updates or additions must be visible to both modules. For example, if a customer makes a booking, it must appear in the admin view, and vice versa.

---

3. System Requirements: Mini Information System (C++ Project)

Your project must meet the following technical and structural requirements. These elements are essential for evaluating your system's functionality, code quality, and algorithmic understanding.

---

1. Functional and Code Structure Requirements
   - At least 10 user-defined functions, including:
     - 4 overloaded/overridden functions
     - At least 1 constructor (in each module)
     - At least 1 destructor (in each module)
   - 2 struct data types
   - 2 base classes
   - 3 derived classes (inheritance implementation)
   - 4 friend functions

- 2 dynamic memory access operations (e.g., using new / delete)
- 1 sorting algorithm (manually implemented, no STL)
- 1 searching algorithm (manually implemented, no STL)
- At least 5 text (txt) files used for:
    - Storing inputs, outputs, or system data (e.g., login info, records)
- The system must include proper error handling using try and catch blocks in C++ to manage exceptions effectively. Error handling mechanisms should be implemented in both the Customer and Staff/Admin modules.
- 1 dynamic non-primitive (DNP) data structure, such as:
    - Stack
    - Queue
    - Linked List
    - Hash Table
    - Linked Stack
    - Linked Queue

---

4. Algorithm Integration Requirement

You are required to implement at least one algorithm from each of the following three categories:

i) Dynamic Non-Primitive Data Structure (choose 1)

- Stack
- Queue
- Linked List
- Hash Table
- Linked Stack
- Linked Queue

ii) Sorting Algorithm (choose 1)

- Merge Sort
- Selection Sort
- Insertion Sort
- Bubble Sort
- Quick Sort

iii) Searching Algorithm (choose 1)

- Binary Search
- Hashing

All algorithms must be manually implemented without using C++ STL (e.g., std::sort, std::find, std::vector).

4. Project Report Guidelines:

The report must document your Mini Information System project in a clear, organized, and professional manner. All groups are required to submit the final report in PDF format, following the format and content structure outlined below.

A. Report Format

- File Format: PDF only
- Main Font: Times New Roman, size 12 pt
- Header/Footer & Page Numbers: Times New Roman, size 9 pt

Header:

- Left: Group No – Project Title
- Right: Subject Code

Footer:

- Right-aligned page numbers

Additional Formatting:

- Use bold, *italic*, and underline where appropriate.
- Apply proper numbering to:
    - Section titles and subtitles
    - Tables
    - Figures

B. Report Content Structure

1. Front Matter

- Cover Page
- Table of Contents
- List of Figures & Tables

---

2. Team Members' Contributions

- Clearly state the role and contribution of each group member

---

Part 1: Introduction

- Topic selection rationale
- System overview and description
- Objectives of the project
- Scope and limitations
- Key system features

Part 2: Data Structures & Algorithms Used

- Explain how you store and organize data in memory
- Specify the Dynamic Non-Primitive (DNP) data structure used (e.g., Stack, Queue, Linked List)
- Explain the sorting and searching algorithms applied
- Include diagrams showing relationships between:
    - Files
    - Data structures
    - Algorithms
- Mention any additional algorithms used beyond core requirements

Part 3: System Features & Output (Screenshots)

- System Overview
  - Purpose and functionality
- Screenshots with descriptions
  - Input/output examples
  - All supported user actions and scenarios
- Ensure all screenshots are numbered and labeled

Part 4: C++ Code Implementation

- Include the full C++ source code
- Code should be:
  - Well-indented and organized
  - Contain meaningful comments
  - Include documentation or explanations for each major function

Part 5: Conclusion

- Summarize your project experience
- Reflect on challenges, achievements, and lessons learned
- Emphasize the relevance of the algorithms and data structures applied
- End on a positive and professional note

Part 6: Appendices

- References & Bibliography
  - All URLs, book titles, and external resources used
- AI Tool Usage
  - State clearly if AI tools (e.g., ChatGPT) were used
  - Mention what parts were AI-assisted and how they were implemented
- Git Log
  - Show a list of commits from your repository
  - Include contribution logs from all group members
  - Use git log to display and explain team collaboration history

---

Summary Checklist for Report

Ensure your report includes the following:

1. **Overview:** Current trends & system relevance to data structures
2. **Algorithms:** Clearly explain the logic behind each implemented algorithm
3. **System Output:** Screenshots, text file samples, and explanations
4. References & Bibliography
5. Git Log & Member Contributions

5. Group Presentation:

a) Final Presentation Guidelines

- Presentation Schedule
- Presentations will commence in Week 14.
- Each group will be allocated 45 minutes to present.
- The exact date and time will be decided and communicated by your respective lecturer.

---

b) Presentation Requirements

- Begin with a brief introduction of your project.
- Students must download and present a live demonstration of their system based on the project uploaded or submitted before the due date
- All group members must actively participate in the presentation.
- Formal attire is mandatory for all presenters.
  - Gentlemen: Shirt, slacks, formal shoes (tie recommended)
  - Ladies: Blouse with skirt/pants or formal dress

Important Reminders

- **No late submissions will be accepted.**
- **Cheating, plagiarism, or direct copying will result in 0 marks for the entire group.**
- **Be prepared, confident, and professional.**

Disclosed AI Use

1. **Your project may avoid marks deduction if the following conditions are met:**
   - AI tool usage is clearly and honestly disclosed in your report. *Example: "Used ChatGPT to generate a base function for the sorting algorithm; logic was modified and adapted."*
   - You provide your own explanation, personalization, and improvements within the code, and these changes are clearly described in the report.
   - The AI-generated code is well-integrated and properly commented, demonstrating understanding and thoughtful application.

2. **Marks Deduction Guidelines for AI Chatbot Usage**

Level Copy-Paste System	**Description**	**Marks Deduction**
Minor 10–20% of work copied	Small portions copied without meaningful changes or explanation. Code works but lacks comments and shows limited understanding, appearing mostly blindly copied.	–5 marks
Moderate 30–50% of work copied	Around 30–50% of the work is copied from an AI chatbot. Multiple sections are taken with little to no adaptation or modification. Student shows minimal personalization efforts, with limited understanding reflected.	–15 marks
Severe 60–80% of work copied	Large sections copied (entire functions, classes, or major parts of report) without any logical changes. Clear evidence of dependency on AI without understanding.	–30marks
Full Excessive 90–100% of work copied (No Disclosure).	Almost entire work copied from AI with no meaningful modification, no comments, no personalization, no logic adaptation and no real effort shown. Uses AI-generated code/content but didn't mention it in the report or code comments.	–40marks