Project Proposal

**Project Title**  
Smart Reminder Application

**Group Members**

* Harsh (24K‑0558)
* Khubaib ur Rehman (24K‑0558)
* Aaraiz (24K‑0620)

**1. Introduction**

**Background**  
In today’s fast‑paced world, staying organized and on‑track with tasks is crucial. Many users rely on paper lists or generic calendar apps that lack customization or automated notifications. Our project focuses on building a console‑based reminder system that leverages Object‑Oriented Programming to manage tasks efficiently.

**Problem Statement**  
Existing CLI tools for reminders often lack core OOP principles such as inheritance and polymorphism, leading to rigid code that’s hard to extend for features like recurring events, snoozing, or user management. We aim to address this by designing a modular, maintainable reminder application that cleanly separates concerns and can be extended with minimal effort.

**Objectives**

* Develop a fully functional C++ CLI (partial GUI using Windows API) application for task reminders.
* Demonstrate key OOP concepts: encapsulation, inheritance, polymorphism etc.
* Implement real‑time notifications, snooze/reschedule options, and recurring tasks.
* Provide file‑based persistence and user login/registration.

**2. Scope of the Project**

**Inclusions**

* User registration and login with password protection.
* Adding, viewing, modifying, deleting, and searching reminders.
* One‑time and recurring reminders (daily, weekly, monthly, yearly).
* Snooze functionality (5‑minute delay).
* Popup notifications with sound alerts (Windows API).
* File I/O for saving/loading reminders.
* Multithreaded background check for due reminders.

**Exclusions**

* Cross‑platform GUI interface (Windows API only).
* Advanced scheduling features (e.g., customized recurrence patterns beyond daily/weekly/monthly/yearly).
* Advanced snoozing features (only limited to 5 mins).
* Database integration (only plain text files will be used for data storage).

**3. Project Description**

**Overview**  
The Smart Reminder Application is a C++ console (partial GUI using Windows API) program that allows multiple users to register, log in, and manage their personal reminders. Core OOP principles ensure the codebase is modular: a base **Reminder** class is extended by **RecurringReminder** class, and polymorphic display() methods render entries uniformly. A separate **NotificationManager** thread polls reminders in real time and triggers Windows popup alerts with beeps.

**Technical Requirements**

* **IDE/Editor:** Visual Studio Code
* **Compiler:** MinGW GCC 11 or later (with C++17 support)
* **Platform:** Windows 10 or later (for Windows API calls)
* **Libraries/APIs:** <thread>, <chrono>, Windows API
* **Project Phases**

1. **Research phase:** In this phase we conducted intensive research to finalize the features that were feasible for us to develop. The libraries we needed the technology we need to learn and all the things wee needed to implement the finalised features.
2. **Task distribution:** In this phase we conducted a group meeting and discussed how the task allotment should be done, and in what tasks a group member felt the most confident in building. In this way we coordinated and evenly distributed the tasks.
3. **Implementation:** Most of our time we spent in implementation, coming up with out own file saving format, learning Windows API, researching how multithreading works were some of the tasks we faced the most difficulty in. But through well coordination and helping each other constantly we got through all the problems.
4. **Testing & Debugging:** This phase is yet to come as our project still isn’t completed therefore testing could not be done now.

**4. Methodology**

**Approach**  
We will follow an agile‑inspired, iterative workflow. Each week concludes with a working build that incorporates that week’s features. We’ll hold brief daily check‑ins (in person or via chat) to discuss progress and blockers.

**Team Responsibilities**

* **Harsh:** File management, Multithreading and User management.
* **Khubaib ur Rehman:** Notification manager and Windows API integration.
* **Aaraiz:** Management of reminders (Add, Delete, View, Modify and Search).

**5. Expected Outcomes**

**Deliverables**

* Fully functional C++ console (partial GUI using Windows API) application (.cpp source files).
* A sample user data file demonstrating saved reminders.
* A short report documenting the features of the program and instructions on how to use.

**Relevance**  
This project reinforces core ICT topics—object‑oriented design, file organization, multithreading, and basic human–computer interaction through notifications—preparing us for more advanced software development tasks.

**6. Resources Needed**

**Software**

* Visual Studio Code
* MinGW GCC 11+
* Windows SDK (for API headers)

**Other Resources**

* Online C++ tutorials for advanced <chrono>/<thread> usage
* Official libraries sites to learn more about them.
* Online forums for doubt solving and guidance.