DOCUMENTATION:

**PROJECT TITLE: *Van Management System***

**ABOUT:**

Our project is designed to make van management easier. The code addresses the challenges of overwhelming information and complex functions that a manager cannot perform efficiently when done manually.

**PERFORMANCE:**

* Efficiently stores data for students and drivers.
* Manages van information effectively.
* Enables managers to perform a wide range of functions on this data, improving organization and decision-making.

**ITS STRUCTURE:**

The code is structured according to OOP principles, including:

* Composition
* Classes
* Enums
* Abstraction
* Inheritance
* Encapsulation
* File Handling

It is professionally designed to allow future expansion. It's not gibberish—it's well-structured. Each class has its own header and CPP file, and finally, the main file combines them all. It is well structured, with seven classes that have proper setters, getters, constructors, and functions. We ensured that every class follows a blueprint instead of a bulky, messy approach. The BaseUser class is the parent, and Student, Manager, and Driver inherit from it. It utilizes loops, switches, vectors, and more to ensure efficiency. . This is not AI-generated—it is original 😊.

**HOW TO RUN IT:**

1. Forget Dev-C++, it’s garbage. Use Code::Blocks.

2. Make sure you have the GNU GCC Compiler installed. Go to Settings → Compiler and check if it's set to GNU GCC Compiler.

3. Open the project file (.cbp) in Code::Blocks. If you see random missing files, you probably moved something you shouldn’t have—fix your folder structure.

4. Make sure complain.txt is open in the project.

5. Hit Build & Run (or press F9).

**LIMITATIONS:**

* Currently, the system is structured around CRUD operations.
* No access control – Everyone can see everything, no restrictions, no security.
* Incomplete TimeSlot enum – Was supposed to handle scheduling better, but had limited time so I have to left it unfinished.
* Vectors are a placeholder – A more realistic data structure could be used instead.
* Van capacity tracking is missing – No system for monitoring how many students are in a van.
* No real-time setup – It's static, not aligned with actual real-world time mechanics.

**FUTURE IMPROVEMENTS:**

* Implement access control – Add authentication so not just anyone can mess with the data.
* Complete TimeSlot enum – Actually make it functional instead of half-baked due to time constraints.
* Replace vectors with real-world structures – Use a more appropriate data handling approach for better efficiency.
* Van capacity tracking – Create a system that monitors how many students are assigned to a van in real time.
* Integrate real-time operations – Make scheduling, status updates, and data logging work dynamically instead of static inputs.

***NOTE: I HAVE ADDED PROJECT FILE OF CODE AND THE (I/O) OF THE VMS.***