### **Refactoring and Optimization Overview for CmManager**

**This document outlines the recent changes and enhancements made to the CmManager module, focusing on replacing the old structure, improving performance, and introducing a new class for efficient data retrieval.**

#### **1. Transition to a New Structure**

* **The earlier structure primarily relied on attributes such as site\_name and sector\_name.**
* **The objective of this update was to replace the old structure across all existing flows with a new, more reliable format.**
* **All occurrences of the old flow have been thoroughly reviewed and replaced with the new structure, ensuring consistency across the codebase.**

#### **2. Enhancing Efficiency and Reliability**

* **Challenge Identified:**
  + **The previous process required fetching data directly from the SDK API for every request. This approach was not only slow but also less efficient.**
* **Solution Implemented:**
  + **To address this, a caching mechanism has been introduced to minimize repetitive API calls.**
  + **This enhancement ensures faster response times and reduces the dependency on continuous SDK API requests.**

#### **3. Introduction of the CellDetails Class**

* **To further optimize data retrieval, a new class named CellDetails has been implemented.**
* **Purpose of the Class:**
  + **The CellDetails class acts as a bridge between the new structure and efficient data caching.**
  + **It retrieves data in the new format while storing frequently accessed data in a cache for faster subsequent access.**
* **Key Benefits:**
  + **Eliminates repeated API calls for identical data.**
  + **Accelerates the data retrieval process by leveraging in-memory caching.**
  + **Ensures a seamless transition to the new structure while maintaining performance.**

#### **Conclusion**

**With these updates, the CmManager module now supports a more robust, efficient, and reliable workflow. The introduction of the CellDetails class addresses long-standing issues related to repetitive API calls and slow data retrieval, making the system significantly faster and more adaptable to future requirements.**

**These changes not only improve the overall performance but also align the module with modern development practices, ensuring scalability and maintainability.**