**Project Report: Task Management**

**1. Introduction**

This project involves creating a simple command-line to-do list application using C++. The application allows users to add, view, and remove tasks organized by categories. The goal is to demonstrate basic data structure manipulation using unordered\_map and to implement a user-friendly interface with ANSI color codes.

**2. Objective**

* To implement a task management system where tasks can be categorized.
* To practice the use of unordered\_map and vector in C++ for managing data.
* To enhance user interaction through color-coded terminal output.

**3. Implementation Details**

**3.1 Key Components**

* **Data Structures**:
  + unordered\_map<string, vector<string>> is used to store tasks. The key is the category name (a string), and the value is a vector of task descriptions.
* **Color Coding**:
  + ANSI escape codes are used to apply colors to terminal output, making the interface more user-friendly. Colors include red, green, yellow, and blue.

**3.2 Main Features**

1. **Add a Task**
   * Users are prompted to enter a task description and a category.
   * The task is added to the specified category in the unordered\_map.
   * Feedback is provided using green text to confirm the task has been added successfully.
2. **View Tasks by Category**
   * Displays all tasks grouped by category.
   * Categories with no tasks are removed from the list.
   * Tasks are displayed in blue to differentiate categories and tasks in default text color for clarity.
3. **Remove a Task**
   * Users can remove a specific task from a category.
   * If the removal of a task leaves the category empty, the category is also removed from the unordered\_map.
   * Feedback is provided to confirm successful task removal or notify if the task was not found.
4. **Exit**
   * Terminates the application with a confirmation message in red.

**3.3 Error Handling**

* **Invalid Input**:
  + If the user inputs non-numeric data when selecting an option, an error message is displayed, and the input is cleared.
* **Task Not Found**:
  + If a task to be removed is not found in the specified category, an error message is shown.
* **Empty Task List**:
  + If there are no tasks available, the application notifies the user with a message in yellow.

**4. Code Explanation**

* **Menu Display**:
  + The menu is printed with options, and user input is handled to navigate between options.
* **Task Management**:
  + unordered\_map is utilized to efficiently manage and access tasks by category. Operations such as adding, viewing, and removing tasks are performed using appropriate C++ STL functions.
* **User Interaction**:
  + The interface is designed to be intuitive, with color-coded messages enhancing the user experience.

**5. Conclusion**

This project successfully demonstrates the use of C++ for building a practical application involving data structures and user interaction. The use of unordered\_map to manage tasks and categories showcases efficient data handling, while ANSI color codes enhance user engagement. The application serves as a foundational example for more complex data management systems and user interfaces in C++.