Project Overview

Topic: - Performance Monitor

Group Members

1 Tejas Bogra(211142)

2 Rakshita Kaushik(211153)

3 Chetanya Agarwal(211141)

The provided code presents a performance monitoring application that gathers and displays crucial system information, including CPU and memory usage. The program consists of classes and functions that collect and exhibit relevant performance data in real-time.

The CPU class is responsible for monitoring the CPU's speed and the system's uptime. It initializes by capturing the current time using the high_resolution_clock from the <chrono> library and obtaining the system's uptime using the Windows API's GetTickCount function. The PerformCpuIntensiveOperation method performs a simulated CPU-intensive task by executing nested loops. Once completed, it calculates the CPU speed by evaluating the elapsed time. The GetCpuSpeedInGhz method returns the CPU speed in gigahertz, while the various GetUptime methods compute and provide the system uptime in seconds, minutes, hours, and days, respectively. The PrintCPUInfo method presents the CPU speed and system uptime in a human-readable format.

The MemoryUsage class encompasses functionality to retrieve memory-related information. It employs the Windows API's MEMORYSTATUSEX structure and the GlobalMemoryStatusEx function to retrieve details such as total memory, available memory, and committed memory. The class includes methods to convert memory values from bytes to gigabytes and a printMemoryUsage method that displays the memory information.

The main function serves as the entry point of the program. It instantiates objects of the CPU and MemoryUsage classes. The program then enters a loop where the user can select the type of information they wish to observe: CPU, memory, WiFi, or exit the program. Based on the user's choice, the corresponding method is invoked to retrieve and display the requested information. For CPU information, the PrintCPUInfo method is called, while for memory information, the printMemoryUsage method is utilized. The program continues to prompt the user for their selection until they opt to exit.

In essence, the code provides a rudimentary yet functional performance monitoring application, allowing users to monitor and observe CPU and memory usage. It showcases the retrieval and display of essential performance metrics using the Windows API and the <chrono> library. However, it's important to note that the WiFi information aspect of the program is left unfinished and requires further development to incorporate complete WiFi monitoring functionality.