Raviraj Tushar Wadnerkar

Individual Project

Problem being solved

- The primary purpose of this command-line application is to analyze a log file that includes different kinds of log entries and organize these entries into specific categories such as APM Logs, Application Logs, and Request Logs.
- Each category of logs contains unique data that requires processing to derive valuable insights.
- For instance, APM Logs focus on compiling performance metrics such as CPU usage and memory load, Application Logs sort entries based on severity levels, and Request Logs scrutinize the details and timing of HTTP requests and responses.

Design Pattern Used

- Chain of responsibility

Chain of Responsibility Pattern: This design pattern involves passing requests through a series of handlers. Each handler in the sequence either processes the request or forwards it to the next handler in the chain.

In this application:

- **Handlers**: Individual handlers are set up for each type of log entry, such as APM, Application, and Request logs. Each handler is tasked with identifying if a log entry falls under its category and processing it based on specific criteria.
- **Chain Formation**: The application constructs a chain in which each handler evaluates a log entry to see if it meets certain conditions, such as specific keywords or patterns. If the entry qualifies, the handler processes it; otherwise, it is sent to the next handler in the sequence.

Consequences of Using the Chain of Responsibility

Advantages:

- **Decoupled Code**: Handlers are independent, adhering to principles of single responsibility and the open/closed principle, allowing each handler to focus exclusively on its specific type of log entry.

- **Flexibility**: It's easy to introduce new types of logs or modify existing ones without affecting other handlers. This adaptability is crucial as the formats and types of logs continue to evolve.

SJSU ID: 017458935

- **Dynamic Control**: The sequence of handlers can be easily adjusted, or handlers can be added or removed dynamically, offering flexible control based on runtime decisions.

Disadvantages:

- **Performance Concerns**: Each log entry may pass through several handlers before being processed by the appropriate one, or it might not be processed at all if it reaches the end of the chain. This could lead to inefficiencies, particularly if the chain is long or the suitable handler is often found toward the end.
- **Debugging Complexity**: Debugging can become complex as the process may involve multiple handlers, making it difficult to identify where errors or unexpected behaviors occur.

Class Diagram