**Describe what problem you’re solving.**

My project tackles the challenge of efficiently organizing and analyzing diverse types of log data. By categorizing logs into specific groups such as APM, Application, and Request logs, it facilitates precise monitoring of system performance, quick identification of errors, and thorough assessment of system health. This enables targeted diagnostics and performance optimization, crucial for maintaining and enhancing the functionality of complex software systems.

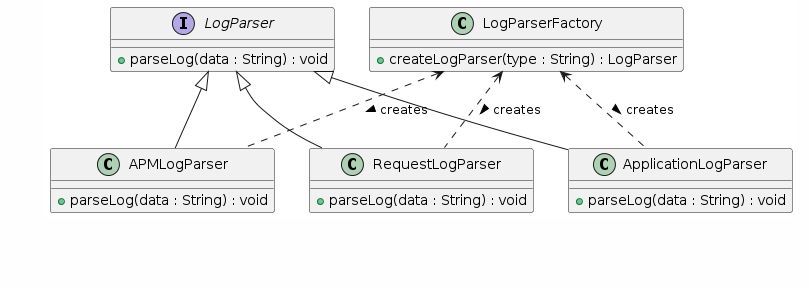
**What design pattern(s) will be used to solve this?**

In my project, the Factory Method design pattern is utilized to manage the creation of various log parser objects. This pattern enables to decide at runtime which parser class to instantiate based on the type of log data provided. By encapsulating the instantiation process in a separate factory class,system remains flexible and scalable, easily accommodating new types of log parsers as needs evolve. This approach simplifies the maintenance of the codebase and enhances its modularity. The Factory Method pattern not only promotes a clear organizational structure but also supports the open/closed principle by allowing new parser types to be added without altering existing code.

**Describe the consequences of using this/these pattern(s).**

Firstly, it enhances modularity and scalability, allowing easy integration of new log parser types without modifying existing code. This adherence to the open/closed principle promotes robust and maintainable code. However, it introduces additional complexity and potentially more classes, which could complicate the system's architecture. While this might increase the initial development effort, it pays off by making the application more flexible and easier to extend in the future. Overall, the Factory Method pattern provides a structured approach to object creation that is particularly beneficial for systems requiring scalability and flexibility in their components.

**Create a class diagram - showing your classes and the Chosen design pattern**

****