Introduction

Microservices architecture is a development approach designed to develop applications through several services that communicate with each other to work as one. Examples are services being developed and run through APIs. This will allow individual services to work independently while also contributing to the overall system. This report will highlight how microservices architecture needs error handling and what different strategies it could implement to catch errors efficiently.

Circuit Breaker Pattern

Circuit breaker pattern is one way to implement error handling into microservices. The method is similar to an electrical circuit breaker where it blocks requests running through from point a to point b. When an operation in a microservices keeps trying to run for execution, the method acts like a breaker and can be used to cut off those attempts to not ruin the entire microservices as a whole. This allows other independent services to run smoothly without repeated attempts of a failed service hindering them. It helps prevent crucial failures and improves system resilience (GeeksforGeeks, 2024).

Retry Pattern

When services are being executed in microservices, there are different types of errors that may pop up due to different circumstances and the Retry pattern helps eliminate one of those circumstances. Retry pattern is used when errors relating to network issues cause errors within microservices. It works by requesting retries in a specific way of delaying each attempt by increasing the delays on each retries to cover network issues etc. Normally, it is set to 3 to 5 retries before it decides that the operation is a failure (GeeksforGeeks, 2024).

Fallback mechanism

Fallback mechanism is when the main operation logic fails to perform, an alternative solution can be made in place of the main one to ensure that the service is at least running to some degree. Main logic normally runs the operation with 100% efficiency but with a “plan b” which is where fallback mechanism comes in, it helps to run the service to ensure no hindering to other services even if it is not optimized to peak performance (Startup House, n.d.).

References

GeeksforGeeks. (2024, October 22). *What is Circuit Breaker Pattern in Microservices?*. https://www.geeksforgeeks.org/what-is-circuit-breaker-pattern-in-microservices/

GeeksforGeeks. (2024, August 5). *Retry Pattern in Microservices*. https://www.geeksforgeeks.org/retry-pattern-in-microservices/

Startup House. (n.d.). *Fallback Strategies: Ensuring Continuity and User Satisfaction Across Domains*. https://startup-house.com/glossary/fallback