Handling partial failure

In this lesson we will learn about partial failures in microservices and how to deal with them using timeouts and the exponential backoff technique.

Partial failures will happen

Even after placing our best efforts to ensure we have a healthy system it is a matter of fact that in distributed systems partial failures will happen. This could be due to multiple reasons including network outages, hardware failures, dependency failures and even routine things like having a deployment in-progress.

Regardless of the cause of the partial failure when calling a dependent service, it could certainly cause our microservice to fail, which will end up in a bad experience for our clients.

So, whenever a service makes a synchronous request to another service, there is an ever-present risk of partial failures and so you must design your service to be resilient to them.

Setting timeouts

One of the first things to consider when making requests to a dependent service is setting appropriate timeouts. A service client should be designed not to block indefinitely and use timeouts.

Think about the experience of our client when the Catalog service takes a long time to come back to our Inventory service, if it ever comes back. Now the Inventory service is also taking forever to respond to the client, leaving our users with a bad experience. And, not only that, at least one of the threads of Inventory is now busy not able to serve any other requests, which reduces the amount of available resources in the service.

Instead of this you can set a timeout of, say, no more than 1 second so that if Catalog service takes more than that to respond the request immediately fails and the Inventory service can in turn return the appropriate request to the client, even if it's a failure.

This enables a more responsive experience and also ensures that resources are never tied up indefinitely.

Retries with exponential backoff

Like we mentioned, it is not uncommon for transient failures to occur in a distributed environment. Therefore, you usually want to give the dependent service one more chance to come back with a successful reply. However, you don't want to keep retrying at a constant rate, since that could overwhelm the dependency.

A good strategy that you can use for retries is the one called "retries with exponential backoff". This strategy performs call retries a certain number of times with a longer wait between each retry.

Here's how it works.

As usual the client will make a request to our service and this one in turn will make a request to its dependent service. If this second request fails, instead of failing right away, our service will wait some time and then it will try again. If the request fails again, we will now wait a longer amount of time before trying the request. If it keeps failing, we will wait a yet longer amount of time before one more try. And eventually, if we have tried enough times with no successful response, we let the call fail.

As you see, this strategy lets the failing dependency have an increasing amount of time to recover. It also avoids overwhelming the dependency.

Let’s see how to implement timeouts and retries with exponential backoff in our Inventory microservice.