FAF.PAD16.1 Autumn 2021

Lab 1: Web Proxy

Handed out: October 10, 2021
Due: Midterm 1

The Service

Service nodes are the ones that do the work a client is interested in. They receive tasks, process them and send back responses. Processing requests is usually costly so we imply the help of gateways and caches. Each service has a database. For the services, you'll need to implement the following features:

- ■ SQL and NoSQL databases;
- ■ Tasks distributed across multiple requests;
- ■ Status Endpoint;
- ■ Concurrent tasks limit;
- ■ Service Discovery;
- ★ Adapter with a unified interface to build DB calls based on DB type;
- ★ Priority System;
- ★ Task Timeouts;
- ★ RPC;
- ★ Unit Testing.

The Gateway

The gateway is the node that receives and forwards user tasks to the service nodes. Based on some logic, the gateway caches responses and balances the load of service nodes. Finally, it has a service registry and chooses from registered services when load balancing. For the gateway service, you'll need to implement the following features:

- ■ Load Balancing: Round Robin;
- ■ Service Discovery;
- ■ Circuit Breaker: Trip if a call to a service fails;
- ■ Outbound API - REST;
- ★ Load Balancing: Service Load;
- ★ Circuit Breaker: Remove service if threshold is reached.

The Cache

A cache allows your system to temporary store responses given by your services and serve them without bothering the service nodes. Using caches makes your system more responsive. Usually caches use in-memory storage. For the cache service, you'll need to implement the following features:

- ■ Keep-Alive Connection;
- ■ Multiple Simultaneous Connections;
- ★ Authentication / Authorization;
- ★ Query Language.

Reporting

To ensure that no surprises happen at the presentation day, besides the final presentation / repo, you are asked to do weekly status reports on your progress. Every lab you will need to provide a status report in which you discuss your *tangible progress* on the lab. Tangible things include, but are not limited to:

- Commits to your code repo;
- Written research notes;
- Diagrams / drawings.

Make sure that all research materials end up in the project's readme file or in a folder called "docs". Don't forget to upload a link to your public repo of the project on Else, in the "Submit Lab" activity.

Grading

To ensure a better workflow during the whole "lab development" period, this lab will have 3 checkpoints, a presentation and (possibly) a test based on the knowledge gained during the lab. These checkpoints will roughly follow the waterfall model of software development, but are open to discussion, so do discuss with your teacher if you would want to apply another approach. *You can pass at most one checkpoint per day*, so plan accordingly.

Passing the first checkpoint involves designing your system. You need to document the services, their API's and the technologies you plan to use during development (i.e. languages, protocols, frameworks etc.). A system's architecture diagram is also required.

Getting the second checkpoint implies that you have created an MVP that can receive tasks from clients and route them to service nodes via a gateway node, can process tasks on said service nodes, and can cache the responses on a cache node.

Lastly, passing the third checkpoint means that you've developed most of the features that you plan to present at the presentation.

To receive a mark for this laboratory work you need to receive a passed mark on all three checkpoints as well as successfully present your work during the presentation, as well as successfully pass the (possible) test.

Working in Groups

Working in groups is allowed but optional. You can work in groups of 1, 2 or 3 students. The bigger the group, the more features you are expected to implement. *The exact workload should be negotiated with your teacher*, before the first checkpoint.

Presentation

"I believe in honesty.. especially now, in what will be your final moments in this uni"
To be admitted to the lab presentation, you need to:

- Submit your repo link on Else;
- Have passed at least 2 checkpoints.

The presentation will follow a strict schedule. During the presentation, you are required to:

- State the mark you are aiming for;
- State the features you have implemented;
- Demonstrate the implemented functionality;
- Pass the individual code review;
- Discuss Midterm 1 test and final mark.

Each team will have at their disposal 15/22.5/30 minutes to present their work (for teams of 1/2/3 members respectively). If by the end of the specified time slot you didn't finish presenting, the presentation is considered finished and I mark you (or not) based on what I have. Obviously, the individual code review has the most influence on passing / not passing.

Readings

[1] Mark Smallcombe. "SQL vs NoSQL: 5 Critical Differences". https://www.xplenty.com/blog/the-sql-vs-nosql-difference.

[2] Phil Sturgeon. "Understanding RPC Vs REST For HTTP APIs". https://www.smashingmagazine.com/2016/09/understanding-rest-and-rpc-for-http-apis/.

[3] Martin Fowler. "CircuitBreaker". https://martinfowler.com/bliki/CircuitBreaker.html.

[4] MDN Web Docs. "HTTP caching". https://developer.mozilla.org/en-US/docs/Web/HTTP/Caching.

[5] Julien Le Coupanec. "Redis Cheatsheet - Basic Commands You Must Know". https://gist.github.com/LeCoupa/1596b8f359ad8812c7271b5322c30946.

Good Luck!