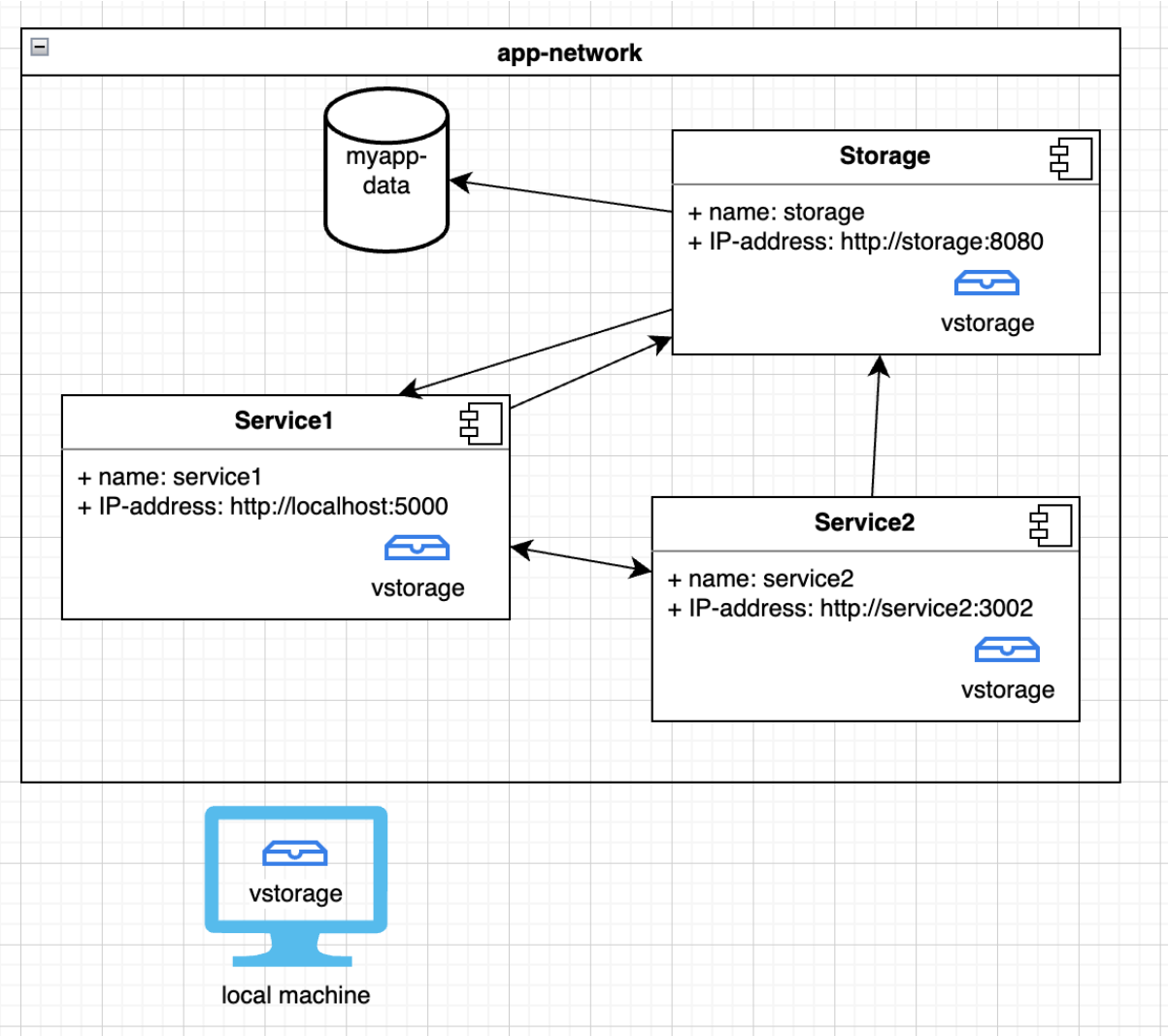
Comp.se.140 Basics of containers and microservices

Basic Information

The development was done on a hardware machine with macOS. Docker used was Docker version 28.0.1 and docker compose used was Docker Compose version v2.33.1.

Diagram



Content of the status records

The disc space and uptime measured was the disc space and uptime of the container the service is running on. The measurements can be used to investigate how different services use disk space and if the containers have been down for some reason. If the containers use a common disk space with each other then it would make more sense to have knowledge of the container platforms general amount of free diskspace and how much is used all together.

Persistent storage solutions

The app had two different types of persistent storage solutions, one was a bindmount that was mounted to each container and the second one was a volume mount that was separated from the containers.

The pros of having a volume mounted directly to a container is that they are relatively easy to set up and inspect so they work well for development. They are also mounted only to one container so it is easy to manage who has access to the mount. One negative side to bind mounts is that they are bound to the host machines file structure.

The pros of having a separate mount running on docker is that it is independent from the containers and different services can easily access the data. The cons for having a separate volume is that they might increase the complexity of the applications as they are separate from the containers.

Cleaning of the persistent storage

The persistent storage can be cleaned using the command:

docker volume rm <volume_name>

What was difficult and the main problems

It was difficult to get the containers to talk to each other until I discovered to create the network for the docker compose file. It was also difficult to create an app with a whole new programing language as I have not used much clojure before. The main problems creating the application was creating the apps to use only the docker network and not the local network and not to expose their endpoints outside of the network.