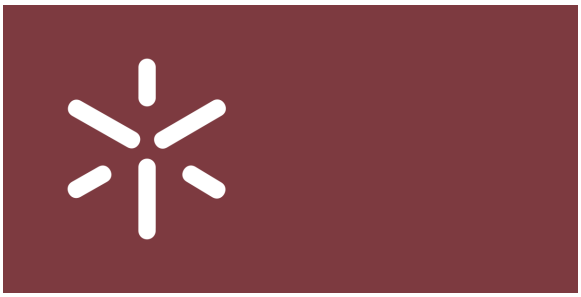

Universidade do Minho

Docker Microservices

Virtualiazação de Redes

Mestrado Integrado em Engenharia Informática

João Amorim A74806

24 de Junho de 2021

1 Introdução

No âmbito da Unidade Curricular de Virtualização de Redes, foi-nos proposta, para o Trabalho Prático nº 2, a implementação de vários micro-serviços com recurso a Docker Containers, seguindo um conjunto de regras que definem a interação entre estes serviços.

De acordo com o enunciado, o trabalho deverá passar por quatro fases, sendo que cada um delas tem como objetivo o aumento da qualidade e complexidade de todo o Sistema.

No presente relatório, será inicialmente apresentada a Arquitetura Base do Sistema que irá ser implementado, com os seus respetivos micro-serviços, seguindo-se uma explicação detalhada da implementação de cada um dos serviços e o respetivo papel que possúi no Sistema.

2 Arquitetura do Projeto

2.1 Planeamento da Arquitetura

Para o planeamento daquilo que será a Arquitetura Base do Sistema, recorri ao material disponibilizado na cadeira do primeiro semestre de Desenvolvimento de Aplicações Web. Aqui, aprendemos a desenvolver determinados serviços, que necessitam uns dos outros, mas que são independentes em termos de implementação. Isto faz com que estes serviços sejam candidatos perfeitos para a sua implementação em Docker Containers.

Ao longo da cadeira referida anteriormente é-nos dito que para o desenvolvimento de uma aplicação web, o ciclo de vida da mesma, comece e acabe no Cliente, ou seja, é no Cliente que são enviados pedidos para o serviço aplicacional, de onde recebe por exemplo chunks com HTML+CSS+JS em que estes representam as Views, ou para o serviço de autenticação, de onde recebe a token que é gerada e que irá ser utilizada para posteriormente enviar pedidos para um determinado serviço. Apesar disto, e tendo em conta o tempo que resta para a conclusão e apresentação do projeto, o Cliente e o Serviço Aplicacional serão o mesmo, o que significa que a estrutura da comunicação entre serviços pode ser demonstrada com a seguinte imagem:

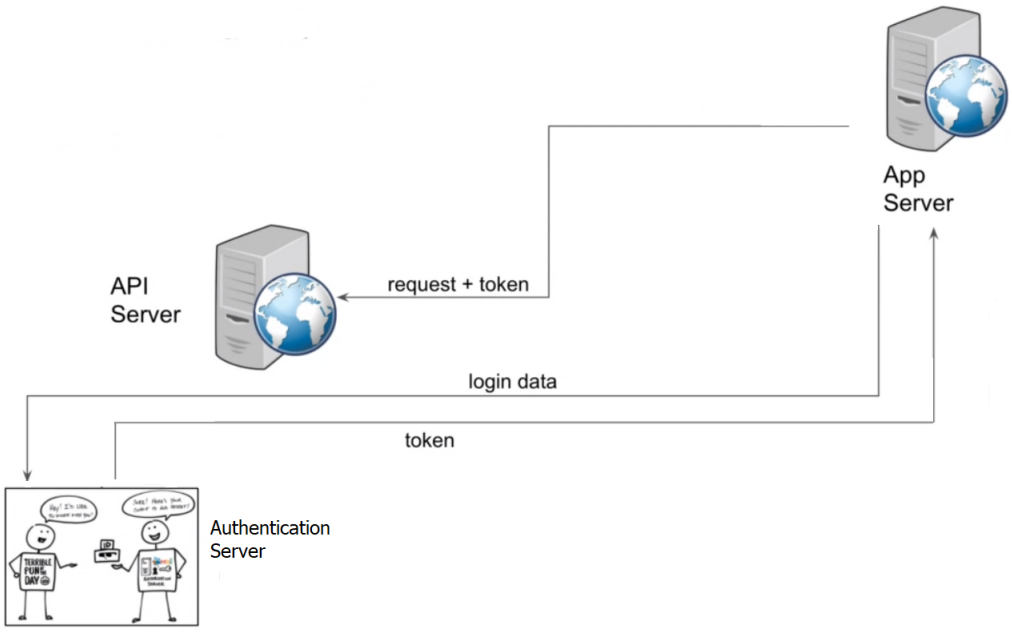

Figure 1: Estruturação da comunicação entre serviços

Isto significa que é através deste Application Service que irá ser feito o pedido de Autenticação ao Authentication Service, que irá devolver a respetiva token, caso os dados do Utilizador se encontrem na Base de Dados, e que irá ser usada, novamente pelo App Service para enviar pedidos ao File Service, que terá obviamente as suas rotas protegidas.

2.2 Arquitetura Desenvolvida

Com base no que foi mostrado até agora, é apresentada aquela que será a Arquitetura Base do Sistema que será desenvolvido neste Projeto:

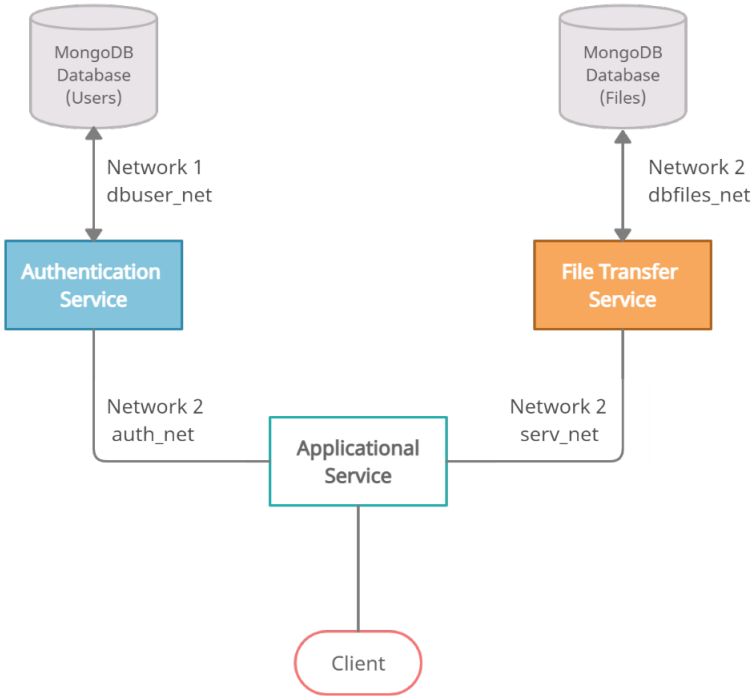

Figure 2: Arquitetura do Sistema

De acordo com a imagem, é possível enumerar os serviços que serão então implementados:

- Application Service
- Authentication Service
- File Transfer Service

Nas próximas secções, irão ser apresentados cada um destes Serviços, o processo aplicado para a transformação destes Serviços em Containers, assim como a Comunicação e Persistências de Dados entre/dos Containers, respetivamente.

3 Serviços

Cada um dos serviços que serão apresentados, foi implementado de raíz, o que significa que terão que ser transformados em imagens, para que os mesmos possam ser usados como Docker Containers.

O processo para criação destas imagens, começa com a criação de um projeto no github, efetuar o push do dockerfile de cada um dos serviços respetivos para dentro do projeto, criar um projeto no Docker Hub e fazer a ligação entre ambos os projetos.

Depois de efetuada a criação das imagens dos vários serviços, podemos passar para o funcionamento dos mesmos.

3.1 Applicational Service

Neste serviço, será usada a imagem criada com o Applicational Service implementado, utilizado Express, que executará num servidor Node.js. Este serviço é responsável por enviar para o Client aquilo que serão as Views da aplicação, nomeadamente as páginas com formulários para Registo e Login de um Client, assim como a página onde será possível fazer o Upload e Download de ficheiros.

Esta útlima, terá a sua rota protegida, pelo que será necessário a existência de um mecanismo de Autenticação. Esta Autenticação, será feita em conjunto com o Authentication Service, sendo que o Applicational Service tem um papel importante a cumprir.

No caso do Registo de um novo utilizador, será enviado um pedido para o Authentication Service com os dados do utilizador a registar, para que este serviço possa executar uma operação de inserção de um novo utilizador na Base de Dados.

Para o Login, o Applicational Service deve recolher os dados fornecidos no formulário e enviá-los para o Authentication Service, de onde deverá receber um token, caso o utilizador exista na Base de Dados. Caso exista, recebe como resposta a token de sessão e guarda a mesma num Cookie no browser. Desta maneira, sempre que enviarmos um pedido para o File Transfer Service, que possúi as suas rotas protegigas, enviaremos este token na query string, para que esta seja validada e passemos a ter acesso ao conteúdo do File Transfer Service.

3.2 Authentication Service

Tal como foi explicado, é necessária a existência de um serviço de autenticação que permita não só efetuar o Registo de novos utilizadores, mas também da autenticação dos mesmos.

Novamente, este serviço vai usar a imagem criada com o Authentication Service implementado, utilizado Express, que executará num servidor Node.js.

Para o correto funcionamento deste serviço, uma vez que iremos recorrer à criação e uso de tokens, são usados os packages "jsonwebtoken" e o "passport", com estratégia "Local", sendo o último um middleware de autenticação utilizado em aplicações que utilizem Express.

Este serviço comunicará com um Container com uma Base de Dados,com uma imagem do MongoDB, que será responsável por guardar os dados de cada um dos Utilizadores, para que o processo de Autenticação possa ser efetuado corretamente.

Esta Autenticação e respetivo processo de geração de tokens, já foi explicado no Applicational Service.

Adicionalmente, será criado neste serviço um Volume para permanência de dados, como um ficheiro de log.

3.3 File Transfer Service

Finalmente, o último serviço irá tornar possível a validação do serviço de autenticação, permitindo o Download e Upload de ficheiros. Tal como os restantes, vai utilizar a imagem criada com o File Transfer Service, que utiliza mais uma vez Express, executando num servidor Node.js.

Este serviço, tal como o anterior, comunicará com um Container com uma Base de Dados,com uma imagem do MongoDB, que será responsável por guardar os ficheiros que foram descarregados pelo Utilizador, para que estes possam ser disponibilizados para Download.

Como já foi referido, os pedidos enviados para este serviço, têm que ser feitos obrigatoriamente com a token que está guardada em cookie no browser, caso a autenticação tenha sido feito com sucesso, visto que as rotas se encontram protegidas.

Adicionalmente, será criado neste serviço um Volume para permanência de dados, como um ficheiro de log.

4 Persistência de Dados

Neste projeto, serão então usados dois Containers com imagens do MongoDB, que irão guardar os dados dos Utilizadores e dos Ficheiros.

Adicionalmente, serão criados quatros volumes, um para cada Container com Mongo, um para o Authentication Service e um para o File Transfer Service.

5 Comunicação entre Containers

Para o correto funcionamento do projeto, é necessário que os Containers criados comuniquem de maneira correta. Esta comunicação é efetuada através da indicação nomes dos containers que pretendemos que comuniquem uns com os outros, sendo o resto do processo é tratado pelo Docker.

Temos apenas que garantir que definimos as redes corretamente no docker compose, de acordo com aquilo que é apresentado na Arquitetura, nas primeiras secções.

6 Conclusão

Neste trabalho, foi abordado o processo de criação de Docker Containers para micro-serviços, sendo que posso dizer que foi um projeto que me alargou horizontes. Antes deste semestre, o próprio conceito do Docker era algo sobre o qual não tinha conhecimento e este trabalho, em conjunto com o primeiro, vieram sem dúvida fornecer-me o conhecimento necessário para dar os primeiros passos.

Em relação ao trabalho produzido, foram criados os Docker Containers para cada um dos Serviços implementados, com respetivos Dockerfiles e Docker Compose, mas bastantes objetivos propostos no enunciado ficaram por cumprir, fazendo com que o trabalho ficasse algo incompleto.

Mesmo assim, faço um balanço positivo uma vez que o projeto despertou, sem dúvida nenhuma, interesse sobre aquilo que poderei a vir um dia fazer com a plataforma.

7 Anexos

De seguida, apresentam-se comandos para testar e observar os Containers criados, assim como os Serviços implementados em funcionamento:

```
joao@joao-VirtualBox:~/Desktop/VR-TP2$ docker-compose up -d
Starting auth ...
db_file is up-to-date
db_user is up-to-date
Starting auth ... done
Starting app  ... done
```

Figure 3: Comando $ docker-compose up -d

```
joao@joao-VirtualBox:~/Desktop/VR-TP2$ docker ps -a
CONTAINER ID   IMAGE                COMMAND                  CREATED
139d7757a8dd   jlpamorim/jwt-auth   "npm start"              23 hours ago
6d6ee5b14a0e   mongo:latest         "docker-entrypoint.s…"   23 hours ago
3c6bc900a49a   mongo:latest         "docker-entrypoint.s…"   23 hours ago
30a26a08f1c3   jlpamorim/http-app   "npm start"              23 hours ago
73a823472437   jlpamorim/http-files "npm start"              23 hours ago
```

Figure 4: Comando $ docker ps -a

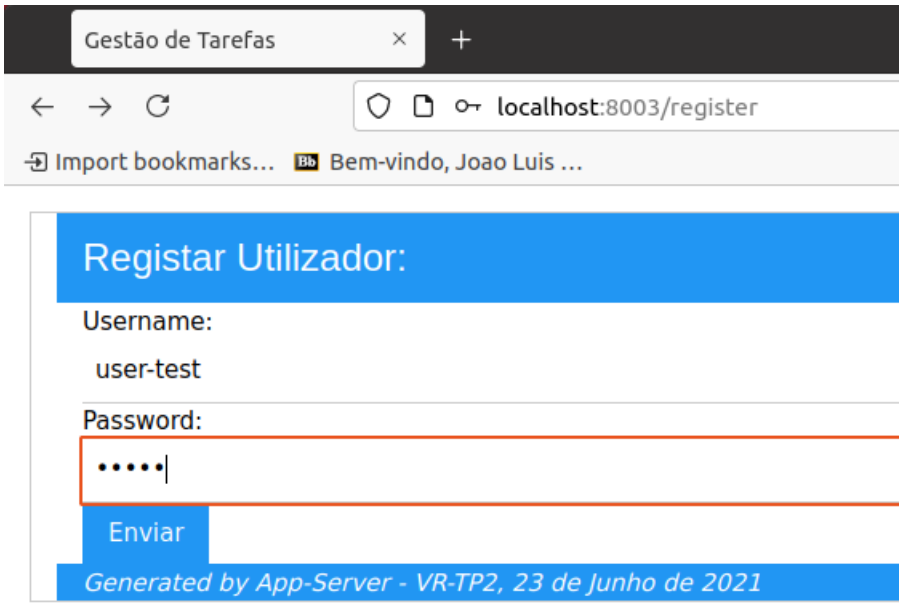

Figure 5: Registar um Utilizador na Base de Dados

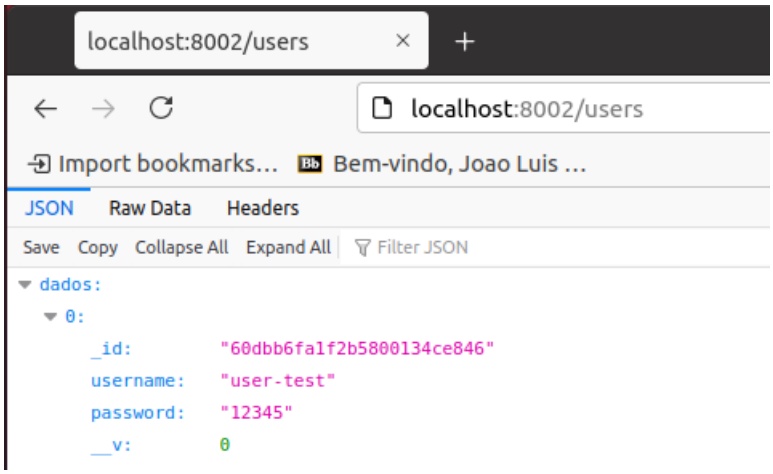

Figure 6: /users no serviço de Autenticação para observar os Utilizadores existentes

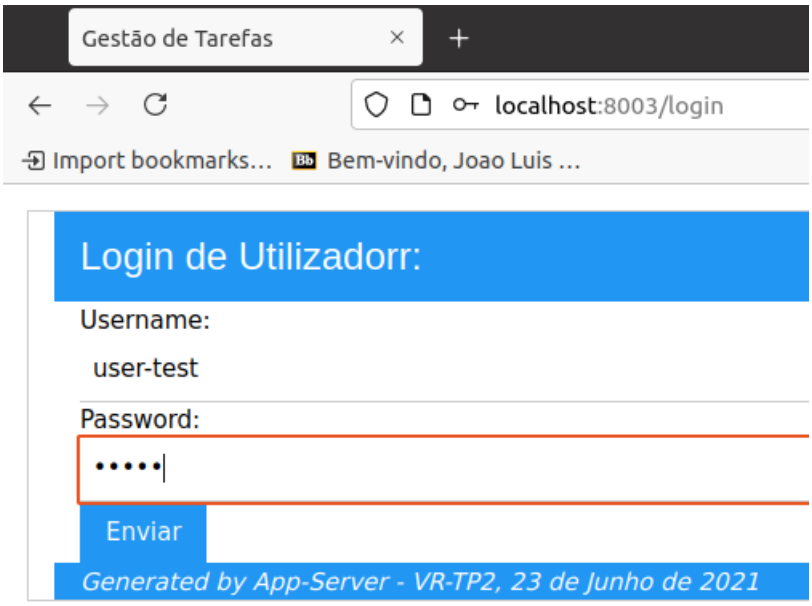

Figure 7: Login de um Utilizador

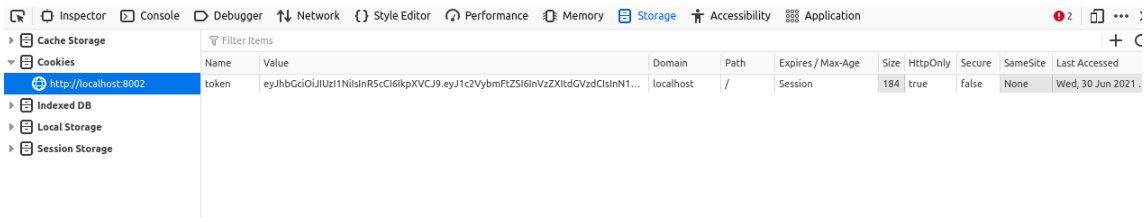

Figure 8: Cookie com a token que foi obtida e posteriormente guardada

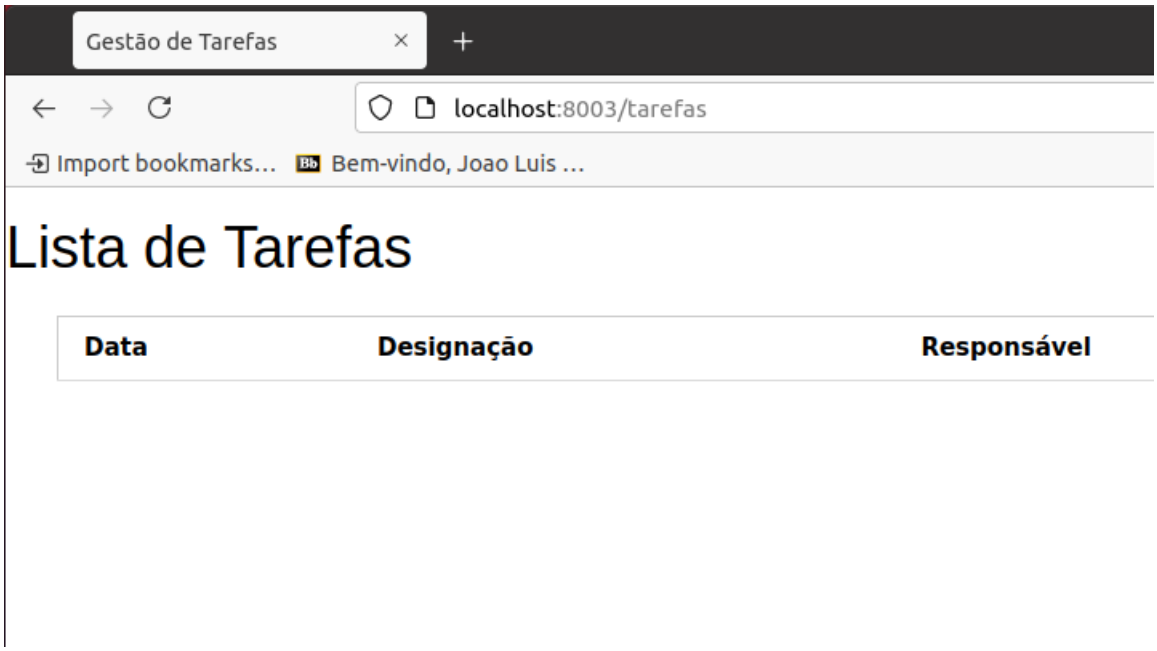

Figure 9: Interface obtida após envio da token para o Serviço Files