

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia

Departamento de Computação

DevOps GRAD_1001544_A_SAO_CARLOS_2024_1

github.com/LucasMBalheiro/DevOps-Pratica

Prática com docker-compose

Prof. Delano Medeiros Beder

Aluno:

Lucas Maciel Balieiro, 800534, BCC

17 de Julho de 2024 - São Carlos/SP

Aplicação escolhida

A primeira aplicação na qual tive em mente de usar era o app que estou desenvolvendo no estágio, sendo ele uma aplicação mobile utilizando ReactNative e MySQL para o IsF (Idiomas sem Fronteiras). Porém, por motivos de privacidade da empresa acabei optando por fazer um site básico usando só HTML, NodeJS e MongoDB.

A containerização feita neste caso seria de 3 containers principais:

- A aplicação web em si
- Uma versão do MongoDB para instanciar dados
- Uma versão do MongoExpress para inserirmos dados pela API

Para executar a aplicação, as versões utilizadas do Docker foram:

```
PS C:\Users\Balieiro> docker -v
Docker version 27.0.3, build 7d4bcd8
PS C:\Users\Balieiro> docker-compose -v
Docker Compose version v2.28.1-desktop.1
PS C:\Users\Balieiro> 
```

As versões do Node, Mongo e Express foram as imagens mais recentes publicadas.



mongo-express
Updated 9 days ago
Web-based MongoDB admin interface, written with Node.js and express
DATABASES & STORAGE

100M+ · 1.5K
Pulls: 316,903
Last week


Learn more



mongo
Updated 2 days ago
MongoDB document databases provide high availability and easy scalability.
DATABASES & STORAGE

1B+ · 10K+
Pulls: 5,569,007
Last week

Learn more



node
Updated 9 days ago
Node.js is a JavaScript-based platform for server-side and networking applications.
LANGUAGES & FRAMEWORKS

1B+ · 10K+
Pulls: 9,290,221
Last week

Learn more

Overview do app

A página é uma simples aplicação em Node que faz uma requisição para o banco de dados por uma database específica (com uma query muito complexa) e retorna essa informação na tela do usuário.

```
let mongoUrlDockerCompose = `mongodb://${DB_USER}:${DB_SENHA}@mongodb`;

let mongoClient = { useNewUrlParser: true, useUnifiedTopology: true };

let database = "devopsDB";
let collection = "devopsCol";

app.get('/fetch-data', function (req, res) {
  let resposta = {};
  MongoClient.connect(mongoUrlDockerCompose, mongoClient, function (erro, client) {
    if (erro) throw erro;

    let db = client.db(database);

    let queryComplexa = { id: 1 };

    db.collection(collection).findOne(queryComplexa, function (err, result) {
      if (err) throw err;
      resposta = result;
      client.close();

      res.send(resposta ? resposta : {});
    });
  });
});
```

Todo o banco será controlado pelo Mongo Express, que é uma interface para gerenciamento de tabelas do MongoDB.

As informações na páginas são requisitadas no seu carregamento, então podemos atualizar o banco de dados que as informações serão atualizadas na página também.

Dockerfile

O arquivo dockerfile instala todas as dependências necessárias do Node que estão presentes no diretório e depois executa o web.

```
1 FROM node:alpine
2
3 #Criando os diretórios com as informações do app
4 RUN mkdir -p /home/app
5 COPY ./app /home/app
6
7 #Setando qual diretório ele vai usar para rodar os próximos comandos
8 WORKDIR /home/app
9
10 #Instala dependencias
11 RUN npm install
12
13 #Sobe a página
14 CMD ["node", "server.js"]
```

O app está todo containerizado dentro desse docker, o docker-compose em seguida gera a docker-network com o Mongo e Express.

```
docker-compose.yml
1 version: "3.8"
2
3 services:
4
5   mongodb:
6     image: mongo
7     ports:
8       - 27017:27017
9     environment:
10      - MONGO_INITDB_ROOT_USERNAME=admin
11      - MONGO_INITDB_ROOT_PASSWORD=senha
12
13   mongo-express:
14     image: mongo-express
15     restart: always
16     ports:
17       - 8081:8081
18     environment:
19      - ME_CONFIG_MONGODB_ADMINUSERNAME=admin
20      - ME_CONFIG_MONGODB_ADMINPASSWORD=senha
21      - ME_CONFIG_MONGODB_SERVER=mongodb
22     depends_on:
23       - "mongodb"
24
25   app:
26     build: .
27     ports:
28       - 3000:3000
29     environment:
30      - MONGO_USERNAME=admin
31      - MONGO_SENHA=senha
32     depends_on:
33       - "mongo-express" #não precisa, porque a conexão só é feita quando a pagina é aberta, mas não custa nada
34
```

O app funciona independente da existência ou não do banco, porém é boa prática manter a dependência visto que, em desenvolvimento, você nunca trabalharia com a aplicação sem a base de dados rodando.

Execução

```
PS D:\github\DevOps-Pratica-DockerCompose> docker ps -a
CONTAINER ID   IMAGE                                COMMAND      CREATED        STATUS        PORTS        NAMES
PS D:\github\DevOps-Pratica-DockerCompose> docker-compose -f docker-compose.yaml up -d
time="2024-07-18T11:00:14-03:00" level=warning msg="D:\\github\\DevOps-Pratica-DockerCompose\\docker-compose.yaml: 'version' is obsolete"
[+] Building 10/13
  █ mongodb Pulled                                                    29.5s
    █ 3713021b0277 Pull complete                                       11.4s
    █ 39bdcacccd97 Pull complete                                       11.5s
    █ d6b691142508 Pull complete                                       12.1s
    █ bcc1924dee6d Pull complete                                       12.4s
    █ 091a7990873d Pull complete                                       12.5s
    █ 77e5254f6ae8 Pull complete                                       12.4s
    █ 403f753f5920 Pull complete                                       26.2s
    █ 88cd53ea307c Pull complete                                       26.2s
  █ mongo-express Pulled                                              20.8s
    █ 619be1103602 Pull complete                                       1.4s
    █ 7e9a007eb24b Pull complete                                       8.3s
    █ 5189255e31c8 Pull complete                                       8.0s
    █ 88f4f8a6bc8d Pull complete                                       8.7s
    █ d8305ae32c95 Pull complete                                       8.2s
    █ 45b24ec126f9 Pull complete                                       8.3s
    █ 9cf7f59574f7d Pull complete                                       17.3s
    █ 0bf3571b6cd7 Pull complete                                       17.0s
[+] Building 34.4s (11/11) FINISHED                                docker:desktop-linux
```

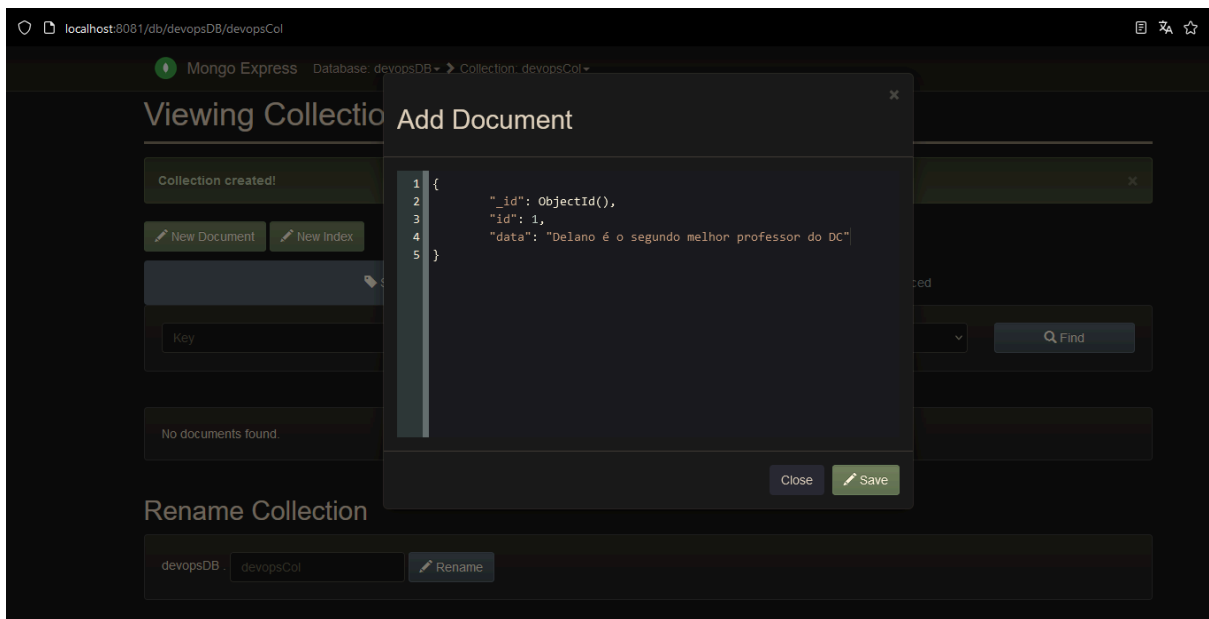
```
[+] Building 34.4s (11/11) FINISHED
=> [app internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 340B 0.0s
=> [app internal] load metadata for docker.io/library/node:alpine 2.5s
=> [app auth] library/node:pull token for registry-1.docker.io 0.0s
=> [app internal] load .dockerignore 0.1s
=> => transferring context: 2B 0.0s
[app 1/5] FROM docker.io/library/node:alpine@sha256:98e7b6b154a3b8d780c618b9622df881f5589a3f097bc7a800ad1418d9d5f72d 17.3s
=> => resolve docker.io/library/node:alpine@sha256:98e7b6b154a3b8d780c618b9622df881f5589a3f097bc7a800ad1418d9d5f72d 0.0s
=> => sha256:c590ed4d62b763538c9b4dfad8460e15dd3bdac2c8c3f3a69dd24f8e7ee551 1.72kB / 1.72kB 0.0s
=> => sha256:c1a8d284bbb626bf5774fecb9475f11b41788ca6044771925b36123f188b9f4 6.38kB / 6.38kB 0.0s
=> => sha256:04aed2589a65eca532f9472df4530273d6b0abd0c9b59b5659d6e90b28e28ee5 47.30kB / 47.30kB 3.4s
=> => sha256:e12fa6d0ccba6512842306425174e3c180fb8460457dc4a6ff34921f3cac1fab 1.39kB / 1.39kB 0.0s
=> => sha256:d524c6361847d1be4eada9f58c30dd63e352a99ebdceaadb27b1ffc682a2abb 448B / 448B 0.0s
=> => sha256:98e7b6b154a3b8d780c618b9622df881f5589a3f097bc7a800ad1418d9d5f72d 6.62kB / 6.62kB 0.0s
=> => extracting sha256:04aed2589a65eca532f9472df4530273d6b0abd0c9b59b5659d6e90b28e28ee5 13.9s
=> => extracting sha256:a12fa640ccba6512842306425174e3c180fb8460457dc4a6ff34921f3cac1fab 0.1s
=> => extracting sha256:d524c6361847d1be4eada9f58c30dd63e352a99ebdceaadb27b1ffc682a2abb 0.0s
[app internal] load build context 0.0s
=> => transferring context: 170B 0.0s
[app 2/5] RUN mkdir -p /home/app 1.5s
[app 3/5] COPY ./app /home/app 0.0s
[app 4/5] WORKDIR /home/app 0.1s
[app 5/5] RUN npm install 11.1s
[app] exporting to image 0.0s
=> => exporting layers 0.0s
=> => writing image sha256:b4f48c90ea6411ad965510ad698b7abda8a8a54b38420a373a89f55fe373bcb75 0.0s
=> => naming to docker.io/library/devops-pratica-dockercompose-app 0.0s
[+] Running 3/4
  Network devops-pratica-dockercompose_default Created 0.1s
  Container devops-pratica-dockercompose-mongodb-1 Started 0.0s
  Container devops-pratica-dockercompose-mongo-express-1 Started 1.2s
  Container devops-pratica-dockercompose-app-1 Started 1.4s
PS D:\github\DevOps-Pratica-DockerCompose>
```

O melhor site apresentado na materia de DevOps

Dados estáticos falsos: **Delano é um PÉSSIMO professor**

Dados factuais da api do Mongo:

Como podemos ver, a aplicação está rodando na porta definida no compose. Agora podemos inserir dados no banco a partir do Expose.



O melhor site apresentado na materia de DevOps

Dados estáticos falsos: **Delano é um PÉSSIMO professor**

Dados factuais da api do Mongo: **Delano é o segundo melhor professor do DC**

Como podemos ver a aplicação em Node tem conexão com a database criada a partir da imagem, porém as informações no site estão erradas, podemos settar novos dados e a página se atualizará.



O melhor site apresentado na materia de DevOps

Dados estáticos falsos: **Delano é um PÉSSIMO professor**

Dados factuais da api do Mongo: **Delano é o MELHOR professor de todos os tempos**

Agora o site apresenta informações atualizadas e corretas.

Depois de testarmos a aplicação podemos remover os containers com docker-compose down.

```
[+] Running 4/4vOps-Pratica-DockerCompose> docker-compose -f docker-compose.yaml down -d
[+] Container devops-pratica-dockercompose-app-1 Removed
[+] Container devops-pratica-dockercompose-mongo-express-1 Removed
[+] Container devops-pratica-dockercompose-mongodb-1 Removed
[+] Network devops-pratica-dockercompose_default Removed
PS D:\github\DevOps-Pratica-DockerCompose> docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
PS D:\github\DevOps-Pratica-DockerCompose> docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
PS D:\github\DevOps-Pratica-DockerCompose>
```