



SÃO
PAULO
TECH
SCHOOL

Sistemas Operacionais

LAB

Docker

Dockerfile + Web-Data-Viz

Monitor Matheus Matos

`matheus.matos@sptech.school`

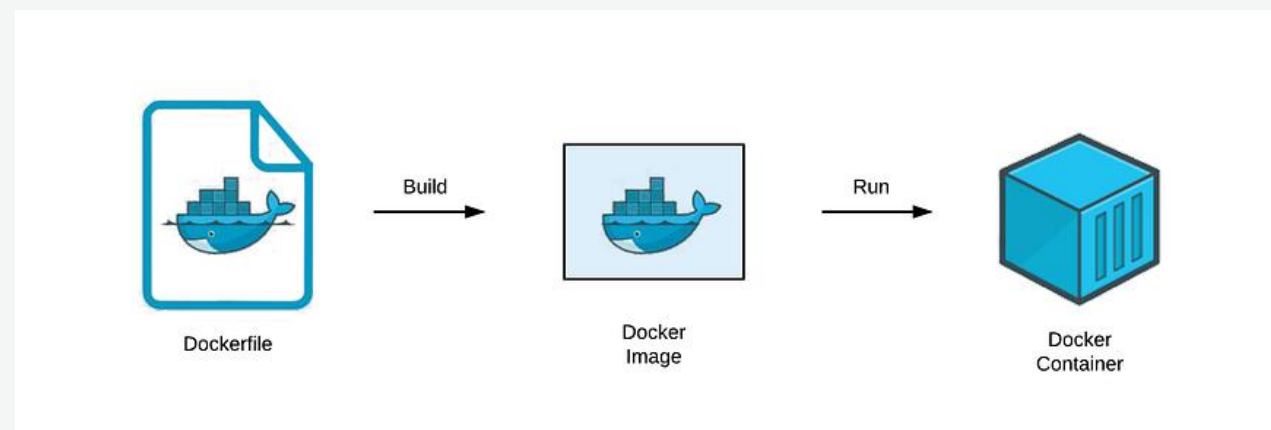
Tópicos da Aula

1. Gerando imagem com Dockerfile
2. Gerando container da imagem



Dockerfile e seus mistérios

- Arquivo de texto;
- Lista de instruções/comandos que descreve como a imagem Docker será construída.



Fonte: Chen, 2020

Criando o Dockerfile



1. Acesse o terminal da EC2, pode ser via protocolo SSH.
2. Crie e edite um arquivo:

`sudo nano Dockerfile`

```
GNU nano 6.2 Dockerfile
```

3. Adicione as seguintes instruções:

`FROM node:17`

`WORKDIR /app`

`COPY ..`

`RUN npm install`

`CMD ["npm", "start"]`

***IMPORTANTE:** Verifique que o Docker já está instalado na VM.



Preparando o Ambiente

3. Faça o clone da aplicação que pretende gerar uma imagem Docker:

`sudo git clone https://github.com/BandTec/web-data-viz.git`

```
ubuntu@ip-172-31-58-0:~$ sudo git clone https://github.com/BandTec/web-data-viz.git
Cloning into 'web-data-viz'...
remote: Enumerating objects: 411, done.
remote: Counting objects: 100% (83/83), done.
remote: Compressing objects: 100% (52/52), done.
remote: Total 411 (delta 54), reused 36 (delta 31), pack-reused 328
Receiving objects: 100% (411/411), 2.72 MiB | 31.66 MiB/s, done.
Resolving deltas: 100% (201/201), done.
```

4. Acesse o **diretório raiz** da aplicação:

`cd web-data-viz/site/`


```
ubuntu@ip-172-31-58-0:~$ cd web-data-viz/site/
ubuntu@ip-172-31-58-0:~/web-data-viz/site$ ls
app.js  package.json  public  src
```

***IMPORTANTE:** O diretório raiz do **projeto** é diferente do diretório raiz da **aplicação**!

Entendendo o Dockerfile



5. Estrutura das instruções do arquivo:

```
 Dockerfile
1  FROM node:17
2  WORKDIR /app
3  COPY . .
4  RUN npm install
5  CMD ["npm", "start"]
```

FROM – Tomar como base outra imagem.

WORKDIR – Define o diretório de trabalho.

COPY – Cópia do diretório raiz para raiz.

RUN – Executa o comando.

CMD – Executa o comando ao iniciar o container.

***IMPORTANTE:** A instrução **COPY** está referenciando o diretório raiz onde o Dockerfile se encontra e o diretório raiz de trabalho do container.



Executando o Dockerfile

6. Lembre-se de mover o Dockerfile para o diretório raiz da aplicação:

```
sudo mv ~/Dockerfile ~/web-data-viz/site/
```

```
ubuntu@ip-172-31-58-0:~$ sudo mv ~/Dockerfile ~/web-data-viz/site/  
ubuntu@ip-172-31-58-0:~$ ls ~/web-data-viz/site/  
Dockerfile  app.js  package.json  public  src
```

7. Para executar o Dockerfile entre no diretório raiz da aplicação:

```
cd web-data-viz/site/
```

```
ubuntu@ip-172-31-58-0:~$ cd web-data-viz/site/  
ubuntu@ip-172-31-58-0:~/web-data-viz/site$
```

8. Build a imagem com o comando:

```
sudo docker build -t app-web-data-viz:1.0 .
```

```
ubuntu@ip-172-31-58-0:~/web-data-viz/site$ sudo docker build -t app-web-data-viz:1.0 .  
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.  
Install the buildx component to build images with BuildKit:  
https://docs.docker.com/go/buildx/
```

- **-t** é usado para adicionar uma tag à imagem Docker, que é um identificador amigável indicando geralmente a versão ou propósito da imagem. No exemplo dado, a tag é **1.0**.



Verificando a Criação da Imagem

9. Verifique se a imagem foi criada:

`sudo docker images`

```
ubuntu@ip-172-31-58-0:~/web-data-viz/site$ sudo docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
app-web-data-viz    1.0                9cee0c9b2d21       36 minutes ago     1.06GB
node                17                 48be0030338e       17 months ago      992MB
```

10. Vamos criar um container a partir da imagem gerada:

`sudo docker run -d -p 8080:3333 --name api-web app-web-data-viz:1.0`

```
ubuntu@ip-172-31-58-0:~/web-data-viz/site$ sudo docker run -d -p 8080:3333 --name api-web app-web-data-viz:1.0
f1c32a2d62070d3a1a9c90695720232bdf3af2e54ef5d7cc72b887afcd0efbbb
```

11. Verifique se o container está no status UP:

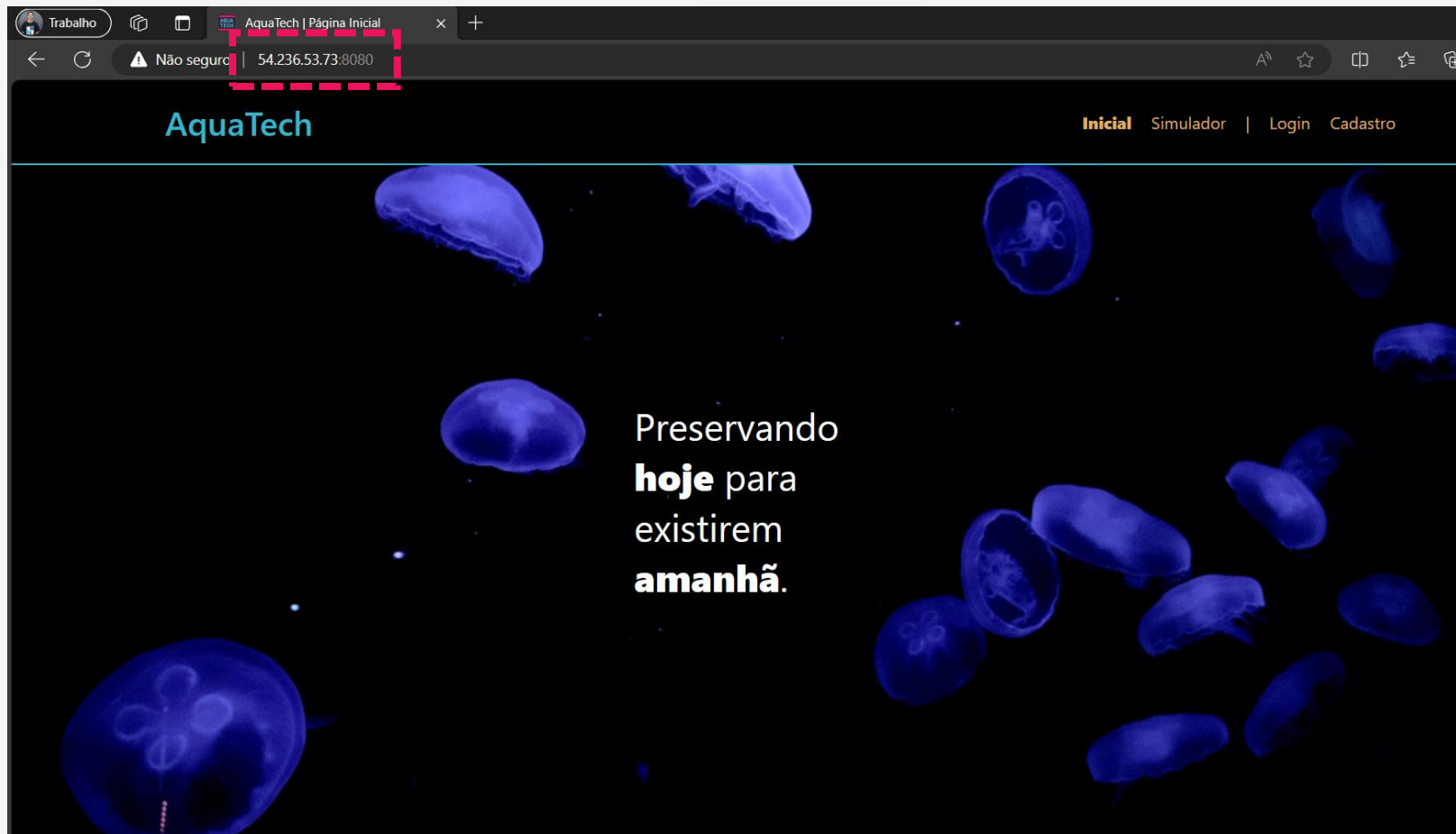
`sudo docker ps`

```
ubuntu@ip-172-31-58-0:~/web-data-viz/site$ sudo docker ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED        STATUS              PORTS                               NAMES
f1c32a2d6207   app-web-data-viz:1.0   "docker-entrypoint.s..." About a minute ago Up About a minute   0.0.0.0:8080->3333/tcp, :::8080->3333/tcp   api-web
```



Testando o Container


12. Abra o navegador e digite o IP Público da VM referenciando a porta 8080:




Atenção!

- Importante lembrar de adicionar as regras de entradas no grupo de segurança na EC2, das portas que estamos utilizando.

Regras de entrada (2)

 [Gerenciar tags](#) [Editar regras de entrada](#)

< 1 >



Name ▾	ID da regra do grup... ▾	Versão do IP ▾	Tipo ▾	Protocolo ▾	Intervalo de portas ▾
acesso-ssh	sgr-03e157efa13ccadd2	IPv4	SSH	TCP	22
acesso-web	sgr-0baa1e373d144d2...	IPv4	TCP personalizado	TCP	8080 - 8081

Conseguimos!!!

- Criamos o Dockerfile;
- Criamos uma imagem;
- Criamos um container;
- Fizemos o deploy da API.



Agradeço
a sua atenção!

Monitor Matheus Matos

matheus.matos@sptech.school

SÃO
PAULO
TECH
SCHOOL