

Criando container web

Transcrição

Na etapa anterior, construímos nosso container do banco de dados, e agora falta fazermos as configurações do container da aplicação web. Em `docker-compose.yml` o nomearemos como sendo `web`, tomando cuidado de deixar tudo com a mesma indentação de `db` (o container do banco de dados).

Qual será a imagem base a ser utilizada pelo container web? Precisaremos de um servidor web para rodar esta aplicação, bem como do PHP instalado - já que o projeto da Alura Sports foi feito nesta linguagem - e, por fim, é preciso abrigar os arquivos do projeto nesta imagem customizada.

Como vimos no curso de Docker, é necessário realizarmos uma customização da imagem conforme nossa necessidade. Conseguiremos fazê-lo através do arquivo ***Dockerfile***, que construiremos agora.

Com "Ctrl + N", abriremos um novo documento no Atom, em que informaremos inicialmente qual a imagem base a ser usada, e este será o ponto de origem. Uma das formas como se diz "origem" em inglês é "*from*", queremos que nossa imagem tenha um servidor web da Apache e o PHP seja instalado na versão `5.6`. Sendo assim, digitaremos:

```
FROM php:5.6-apache
```

Lembremos que esta imagem que estamos customizando se refere à parte **web**, que precisa se comunicar com o banco de dados por meio de `mysqli_connect`, que não vem instalado automaticamente.

Faremos isto acrescentando um comando de instalação:

```
FROM php:5.6-apache
RUN docker-php-ext-install mysqli
```

Feito isto, indicaremos que os arquivos do projeto do diretório `loja` irão para `var/www/html`, do servidor da Apache, que é onde todo o conteúdo web fica inserido.

Não podemos esquecer que a comunicação web utiliza a porta de comunicação `80`, ou seja, devemos informar à imagem, que agora contempla nossa parte web, de que é necessário expor esta porta para que possamos acessar estes recursos web:

```
FROM php:5.6-apache
RUN docker-php-ext-install mysqli
COPY /loja /var/www/html
EXPOSE 80
```

Salvaremos este novo arquivo em "Projeto", nomeando-o com "Dockerfile". Faltava criarmos a imagem customizada a partir dele. No terminal, acessaremos o diretório em que salvamos o arquivo recém criado:

```
cd Área\ de\ Trabalho/
cd Projeto/
```

```
ls
docker build -t "aplicacao-loja:v1" .
```

Desta forma, teremos a construção da nossa imagem com base no "Dockerfile". Aguardaremos a finalização do processo de criação da imagem `aplicacao-loja:v1`. Vamos voltar ao arquivo `docker-compose.yaml` para passarmos esta imagem junto aos arquivos da Alura Sports.

Precisaremos acessar o conteúdo web deste container também, portanto iremos mapear sua porta, referente à parte web, para outra, disponível na máquina local. Diremos que a porta `80` deverá ser mapeada para a `8080` da máquina local - sendo que poderia ser qualquer outra porta disponível.

Não podemos esquecer que os containers, `web` e `db`, precisarão se comunicar entre si, e o primeiro depende do segundo para cadastrar os produtos e realizar a persistência dos dados. Isto será indicado da seguinte forma:

```
web:
  image: aplicacao-loja:v1
  ports:
    - 8080:80
  depends_on:
    - db
```

Com isto terminamos a configuração do arquivo do `docker-compose.yaml`. Salvaremos este arquivo e voltaremos ao terminal para criarmos os containers e inicializá-los. Digitemos o seguinte, apertando "Enter" em seguida:

```
docker-compose up -d
```

Assim, teremos os passos para a criação do container `db` com a imagem do MySQL e, depois, o processamento do nosso container da aplicação web. Aguardaremos um pouco e obteremos `projeto_db_1` e `projeto_web_1`, que devem estar sendo executados em modo "plano de fundo", ou "*background*". Para confirmar isto, usaremos o comando `docker ps` no terminal.

Vamos testar nossa aplicação! Para isso, acessaremos e criaremos tabelas em nosso container do banco de dados, `loja`. Pediremos ajuda ao docker para executarmos alguns comandos:

```
docker exec -it projeto_db_1 sh
```

Apertaremos "Enter" e nos encontraremos dentro do container. Para acessarmos o MySQL, vamos digitar:

```
mysql -u root
```

E apertaremos "Enter" novamente, após o qual teremos que utilizar o banco `loja`:

```
use loja
```

Os diretores da Alura Sports já nos passaram os códigos SQL para criarmos estas tabelas, então basta apenas copiar e colá-los no terminal, apertando "Enter" depois.

Agora, verificaremos se de fato a nossa aplicação está funcionando como esperado. Para tal, abriremos uma janela no *browser* e digitaremos `localhost:8080`, pois em nosso arquivo `docker-compose.yml` havíamos feito o mapeamento da porta `80` do container da aplicação web para a porta `8080`, da máquina local.

Ao acessarmos a página, vê-se que a aplicação está disponível! Vamos fazer alguns testes? Cadastraremos alguns produtos preenchendo o campo "Nome" com "Camiseta seleção brasileira", "Preço" com "200", "Descrição" com "Camiseta verde e amarela" na categoria "Futebol". Depois, clicaremos em "Cadastrar". Maravilha!

Clicaremos em "Adiciona Produto" na aba superior e incluiremos "Raqueteira", "300" e "Raqueteira marca Alura", marcaremos o checkbox de "Usado" e cadastraremos na categoria "Tenis".

Na nova página, clicaremos em "alterar" para trocar o valor da camiseta, adicionada anteriormente, para "250". A mensagem abaixo é exibida:

O produto Camiseta seleção brasileira , 250 foi alterado.

Continuando, desta vez removeremos o produto "Raqueteira", com sucesso. Nossa aplicação está funcionando!!