Docker Compose – Exercícios - Resolvidos

1 - Você deve criar uma aplicação ASP .NET Core MVC mais simples possível e a seguir executar essa aplicação em um contêiner Docker. **Resposta:** *(O arquivo **projeto_webmvc.zip** é a resolução deste exercício.)*

a-) crie a aplicação ASP .Net Core MVC usando a ferramenta de linha de comando:
- Abra um terminal de comandos e crie o projeto usando os comandos:

mkdir webmvc
cd webmvc
dotnet new mvc
dotnet run

Esses comandos criam e executam uma aplicação ASP .Net Core MVC chamada webmvc.

b-) crie um arquivo **Dockerfile** que permita criar uma imagem para executar a aplicação MVC

Entre na pasta do projeto – **cd webmvc** e digite **Code .** para abrir o projeto no VS Code

Crie o arquivo chamado **Dockerfile** na raiz do projeto com conteúdo abaixo:

```
FROM microsoft/dotnet:2.1-aspnetcore-runtime
COPY dist /app
WORKDIR /app
EXPOSE 80/tcp
ENTRYPOINT ["dotnet", "webmvc.dll"]
```

Para executar a aplicação precisamos usar a imagem base **microsoft/dotnet:2.1-aspnetcore-runtime** *(não precisamos do SDK pois não vamos compilar o projeto no contêiner)*

Vamos ter que publicar a aplicação em uma pasta **dist** no projeto e copiar o conteúdo desta pasta para a pasta **/app** do contêiner

Temos que expor a porta 80 e a seguir executar a aplicação no contêiner : **dotnet webmvc.dll**

Essa será a imagem que iremos criar.

c-) crie um arquivo **Docker-Compose** para criar o contêiner da aplicação a partir da imagem criada no item anterior

Ainda no VS Code crie um arquivo chamado **docker-compose.yml** na raiz do projeto e inclua o código abaixo neste arquivo:

```
version: "3"

networks:
     frontend:

services:

    mvc:
      build:
        context: .
        dockerfile: Dockerfile
      container_name: AppMvc
      networks:
        - frontend
      ports:
        - 5000:80
```

Criamos a rede **frontend** e a seguir definimos um serviço que vai criar o contêiner **AppMvc** usando o **build** no contexto da **pasta atual** usando o **Dockerfile** criado anteriormente

Nosso contêiner vai usar a rede **frontend** e mapear para a porta 5000 do **host** para a porta 80 do contêiner.

Antes de testar você tem que publicar a aplicação na pasta **dist** no projeto:

dotnet publish --configuration Release --output dist

Docker Compose – Exercícios - Resolvidos

Para testar basta digitar: **docker-compose build** e depois **docker-compose up -d** e acessar o navegado em **localhost:5000**.

2-) O que é o **Docker Compose** e para que serve ? **Resposta**

O Docker Compose é uma ferramenta usada para descrever aplicativos complexos e gerenciar os contêineres, redes e volumes que eles exigem.

Dessa forma o Docker Compose simplifica o processo de configuração e execução de aplicativos para que você não precise digitar comandos complexos, o que pode levar a erros de configuração.

A descrição da aplicação e seus requisitos é definida em um arquivo de composição usando o formato YAM que pode usar a extensão **.yaml ou .yml (mais comum).** O nome padrão usado é **docker-compose.yml.** O processamento do arquivo é feito usando o comando **docker-compose**.

3-) Quais os principais comandos usando com o Docker Compose ? **Resposta**

docker-compose build – cria a imagem
docker-compose up - processa o arquivo de composição
docker-compose ps – exibe os contêineres criados
docker-compose logs – exibe o log gerado no processamento
docker-compose start – inicia um contêiner para um serviço
docker-compose stop – para um contêiner para um serviço
docker-compose pause – pausa a execução de contêiner para um serviço
docker-compose unpause – desfaz uma pausa para um contêiner
docker-compose rm – remove contêineres parados
docker-compose down – para e remove contêineres, redes, volumes e imagens criada pelo comando up

Para detalhes dos comandos veja a documentação no link abaixo:
https://docs.docker.com/compose/reference/