**Atividade — Introdução ao Docker**

**instruções:**

**Questões de 1 a 10: resposta no documento com o texto tamanho 11 e sem negrito**

**Questões 11 a 15: link para o repositório no GitHub**

**1) O que é uma imagem Docker e qual é a sua principal função?**

Uma imagem Docker é tipo uma “foto” do ambiente que sua aplicação precisa para rodar. Ela já vem com tudo pronto: código, dependências, configurações... Assim, você pode criar vários containers iguais a partir dela, sem dor de cabeça.

### **2) Qual é a diferença entre imagem e container no Docker?**

Uma imagem Docker é tipo uma “foto” do ambiente que sua aplicação precisa para rodar. Ela já vem com tudo pronto: código, dependências, configurações... Assim, você pode criar vários containers iguais a partir dela, sem dor de cabeça.

### **3) O que significa a instrução EXPOSE dentro de um Dockerfile?**

O comando EXPOSE dentro do Dockerfile só serve pra avisar qual porta o container vai usar. Não abre a porta pra fora sozinho, é só uma documentação interna. Pra acessar de fora, tem que usar o -p no docker run.

### **4) Explique com suas palavras o que são volumes no Docker e por que eles são úteis durante o desenvolvimento de uma aplicação.**

Volumes são um jeito de guardar dados do container fora dele, no seu computador. Isso é ótimo pra não perder nada quando o container for apagado e também pra editar arquivos localmente e ver as mudanças na hora, sem precisar reconstruir tudo.

### **5)a) No Docker, o comando para construir uma imagem a partir de um Dockerfile é:**

docker build -t nome-da-imagem .

**5)b) o comando para rodar um container a partir dessa imagem é:**

docker run nome-da-imagem

**6) Você está na pasta de um projeto com um Dockerfile. Escreva o comando para criar uma imagem Docker chamada meu-app.**

docker build -t meu-app .

### **7) Agora, escreva o comando para executar o container dessa imagem, mapeando a porta 5000 do container para a porta 8080 da sua máquina.**

docker run -p 8080:5000 meu-app

### **8) Suponha que você queira editar os arquivos da aplicação localmente e que as alterações reflitam no container imediatamente. Qual opção de volume (bind mount) você adicionaria ao comando docker run?**

docker run -v $(pwd):/app meu-app

**9) Seu container está com o nome app-dev.**

**Qual comando você usaria para:**

1. **Parar o container**

docker stop app-dev

1. **Remover o container**

docker rmi app-dev

**10) Imagine que você terminou o desenvolvimento e agora quer remover a imagem que criou (chamada meu-app).**

**Qual comando você usaria?**

docker rmi meu-app

**Para cada questão a seguir, escreva um Dockerfile que atenda às regras especificadas. Coloque cada Dockerfile em uma pasta com o nome da pasta sendo o número da questão, e coloque a resposta o link para o github.**

### **Questão 11 — Python com Hello World no Terminal**

**Regras:**

* **Use uma imagem oficial do Python.**
* **O container deve rodar um script chamado** [**hello.py**](http://hello.py)**. (crie esse script tambem)**
* **O script imprime "Hello, World from Docker!" no terminal.**
* **Não precisa expor portas (não é aplicação web).**

### **Questão 12 — Aplicação Web em Python (Flask)**

**Regras:**

* **Use uma imagem Python.**
* **Instale a biblioteca Flask.**
* **A aplicação deve estar em um arquivo chamado app.py e rodar na porta 5000.**
* **A porta 5000 deve ser exposta no Dockerfile.**

### 

### **Questão 13 — Imagem com Node.js e Express (via clone de projeto GitHub)**

**Regras:**

* **Use uma imagem oficial do Node.js (ex.: node:18-alpine).**
* **Clone diretamente do GitHub esse projeto:** [**https://github.com/eMahtab/node-express-hello-world**](https://github.com/eMahtab/node-express-hello-world)
* **Instale as dependências com o comando npm install.**
* **A aplicação deve rodar um servidor na porta 3000, respondendo com uma mensagem na localhost .**
* **A porta 3000 deve ser exposta via EXPOSE.**
* **O Dockerfile deve copiar ou usar diretamente o código do clonagem (usando RUN git clone ... ou em conjunto com COPY).**
* **Use CMD para rodar o comando npm start (que inicia o servidor).**

### **Questão 14 — Container para um Script de Processamento**

**Regras:**

* **Crie uma imagem baseada no ubuntu**
* **instale linguagem python do container (usando RUN apt-get install …).**
* **O container executa um script chamado process.py que processa dados (simulado, pode ser um print como "Processing data...").**
* **A imagem deve conter apenas as dependências necessárias (por exemplo, pandas se desejar).**
* **Não é necessário expor portas.**

### **Questão 15 — Servidor NGINX Personalizado**

**Regras:**

* **Use uma imagem oficial do NGINX.**
* **Crie um arquivo index.html personalizado que deve ser servido na raiz /.**
* **Configure o container para servir esse index.html.**
* **A porta 80 deve ser exposta.**