Instituto Federal de Educação, Ciência e Tecnológica da Paraíba

Tecnologia em Análise e Desenvolvimento de Sistemas

Desenvolvimento de Aplicações Corporativas

Professora: Asheley Emmy Lacerda Alves

**Alunos**: Diuari Pessoa e Guilherme Costa.

**Atividade Colaborativa 01 – Docker**

# **Objetivos de aprendizagem**

● Saber identificar as características de uma image e container Docker.

● Possuir um overview sobre os principais comandos e customizações.

● Propor estratégias para manutenção de um sistema fazendo uso de containers.

# **Metodologia**

Esta atividade prática está planejada para ser executada em equipes (**duas ou três** pessoas) e em duas etapas. A primeira etapa será um **levantamento teórico**, onde cada equipe deve responder às cinco questões listadas no *Teste de objetivos de aprendizagem*. As respostas devem estar em um arquivo no repositório. Por fim, a segunda etapa consiste em realizar a atividade prática descrita na *Atividade Prática*.

Todos os artefatos devem ser entregues em um repositório do Github e o link **(apenas um link por equipe)** adicionado à atividade no Google Classroom.

# **Teste de objetivos de aprendizagem**

1. Qual a diferença entre **image** e **container**?

A imagem é um template que o container irá utilizar,ou seja, é um arquivo de configuração que servirá de base para os containers podem ser rodados vários containers a partir da mesma imagem ,nela são definidos o que vai rodar ali dentro e quais serão as bibliotecas e configurações que estarão presentes nela e não pode ser executada é apenas um arquivo de leitura.

O container é um ambiente isolado entre o sistema operacional e outros containers, é a estrutura que vai ser executada,ou seja, execução o que está na imagem fazendo com que as alterações que ocorram em tempo de execução não persistam de volta na imagem utilizada de base.

2. Qual a diferença entre os comandos COPY, EXPOSE e ADD?

A instrução ADD assemelha-se à copy fazendo a cópia de arquivos ou diretórios para o destino especificado dentro do container. Esse comando espera argumentos de origem (<src>) e destino (<dest>) do arquivo e também permite a cópia de arquivos externos por meio de uma url informada. Por permitir fazer a extração de arquivos compactados ao mesmo tempo em que copia arquivos compactados, tornou-se bastante comum casos em que apenas copiava quando você esperava por uma extração e extraia quando queria apenas copiar.

A instrução COPY é uma maneira de copiar um arquivo sem nenhuma manipulação automática para o destino, como o ADD geralmente problematiza ao fazer. Ela não é capaz de fazer a extração de arquivos compactados e lida apenas com cópias de arquivos/diretórios presentes no host de build da imagem.

A instrução EXPOSE abre uma porta específica com um protocolo de rede específico dentro de um container docker para realizar a conexão do container com outros containers, com o host ou até mesmo para serviços do mundo externo. Para especificar a porta usa-se a seguinte sintaxe: EXPOSE <port>/<protocolo>. Se não especificado, por padrão o protocolo será o TCP.

3. Qual a diferença entre os comandos RUN, CMD e ENTRYPOINT?

O comando run permite executar comandos dentro da sua imagem docker eles são executados uma vez no tempo de compilação e são gravados na imagem do docker como uma nova camada.

O comando CMD deve ser usado como uma maneira de definir argumentos padrão para um comando ENTRYPOINT ou para executar um comando em um contêiner.

O comando ENTRYPOINT especifica um comando que sempre será executado quando o contêiner for iniciado.

4. Qual a diferença entre as estratégias de shell e exec?

Exec executa um comando em um container em execução. O comando iniciado usando exec apenas é executado enquanto o processo principal do contêiner estiver em execução e não será reiniciado se o contêiner for reiniciado.

Shell executa diretamente comandos em processos filhos que afetarão o processo principal, para isso precisa-se especificar o container que será afetado pelo comando.

5. Qual a diferença entre os comandos docker stop <container\_id> e docker kill <container\_id>?

O comando docker stop interrompe o container em execução,ou seja, tenta disparar um desligamento normal que ocorre num determinado tempo que pode ser configurado enquanto o comando docker kill mata um contêiner em execução sem que haja esse tempo de espera.

# **Atividade prática**

Desenvolva uma aplicação que realize as operações de CRUD para a entidade Livro e Editora. A funcionalidade precisa estar disponível com UI (interface para o usuário) com um template usável. A aplicação desenvolvida precisa estar disponível em contêiner. As demais informações de cadastro podem ser inseridas via script sql.

class Livro{

private int id; private String titulo;

private LocalDate dataDeLancamento;

}

class Editora{

private int codigo; private String localDeOrigem;

private String nomeFantasia;

}

# **Requisitos**

● RF01 - Implementar classes de acesso aos dados (em memória e via JDBC);

● RF02 - Criar as páginas para edição e listagem da entidade Livro;

● RF03 - Criar as páginas para edição e listagem da entidade Editora;

● RF04 - Criar uma página para realizar uma busca de Livro por titulo;

● RF05 - Criar uma página para realizar uma busca de Editora por localDeOrigem;

● RF06 - Realizar o deploy da aplicação no Docker usando uma das *images* do Payara.

● RF07 - Realizar o deploy da aplicação usando o Docker Compose.

# **Script do banco**

O script para criação do banco.

CREATE TABLE livro(

id serial PRIMARY KEY, titulo VARCHAR(80), dataDeLancamento DATE

);

CREATE TABLE editora(

codigo SERIAL PRIMARY KEY, localDeOrigem VARCHAR(100), nomeFantasia VARCHAR(100)

);