**Criando um Ambiente PHP com Docker (Nginx + PHP-FPM)**

1. O que são contêineres e por que usamos Docker para desenvolvimento?

Contêineres são unidades leves e portáteis que encapsulam uma aplicação e todas as suas dependências, garantindo que ela funcione de maneira consistente em diferentes ambientes. O Docker é uma plataforma que facilita a criação, o gerenciamento e a execução de contêineres, permitindo que desenvolvedores construam, testem e implantem aplicações de forma rápida e confiável.

1. Qual a função do PHP-FPM e por que ele é utilizado junto ao Nginx?

O PHP-FPM (FastCGI Process Manager) é uma implementação alternativa do PHP que gerencia processos para interpretar scripts PHP de forma eficiente. O Nginx é um servidor web que, por si só, não processa arquivos PHP. Portanto, ele delega essa tarefa ao PHP-FPM, que executa os scripts PHP e retorna o resultado ao Nginx para ser servido ao cliente.

1. Qual a diferença entre um servidor Apache e um servidor Nginx?

* **Arquitetura:** O Apache utiliza uma abordagem baseada em processos ou threads, enquanto o Nginx utiliza uma arquitetura assíncrona e baseada em eventos, permitindo melhor desempenho sob alta carga.
* **Módulos:** O Apache possui uma vasta gama de módulos integrados, enquanto o Nginx é mais enxuto e focado em desempenho e escalabilidade.
* **Configuração:** O Nginx é conhecido por sua configuração simples e eficiente para servir conteúdo estático e atuar como proxy reverso.

1. Como o Docker organiza imagens e contêineres? Explique o conceito de multi-stage builds.

O Docker usa imagens como modelos imutáveis que contêm tudo o que é necessário para executar uma aplicação. A partir das imagens, são criados os contêineres, que são instâncias na execução das imagens.

O conceito de multi-stage builds permite criar imagens Docker em etapas distintas, onde cada estágio pode utilizar uma imagem base diferente. Isso é útil para construir aplicações de forma eficiente, incluindo apenas o necessário na imagem final, resultando em imagens menores e mais seguras.

**Desafíos Extras (Opcional)**

1. Modifique o **Dockerfile** para instalar **extensões do PHP**, como pdo\_mysql e mbstring.

Para que o PHP possa interagir com bancos de dados MySQL e manipular strings multibyte, é necessário instalar as extensões **pdo\_mysql** e **mbstring**. No entanto, a instalação da extensão **mbstring** requer a biblioteca **libonig-dev**. Portanto, o **Dockerfile** foi ajustado para incluir a instalação dessa dependência:

# Usa a imagem oficial do PHP com suporte a FPM

FROM php:8.2-fpm

# Define o diretório de trabalho no contêiner

WORKDIR /var/www/html

# Instala dependências necessárias, incluindo extensões PHP

RUN apt-get update && apt-get install -y libonig-dev \

&& docker-php-ext-install pdo\_mysql mbstring \

&& rm -rf /var/lib/apt/lists/\*

# Copia os arquivos da aplicação para dentro do contêiner

COPY . /var/www/html

# Define permissões adequadas para os arquivos

RUN chown -R www-data:www-data /var/www/html && chmod -R 755 /var/www/html

# Expõe a porta 9000 para comunicação com o Nginx

EXPOSE 9000

# Comando para iniciar o PHP-FPM

CMD ["php-fpm"]

1. Configure um volume Docker para persistir os arquivos da aplicação.

Para garantir que os dados da aplicação sejam preservados mesmo após a interrupção ou remoção dos contêineres, configurei um volume no Docker Compose. Isso permite que as alterações feitas nos arquivos da aplicação no host sejam refletidas no contêiner em tempo real. No arquivo **docker-compose.yml**, adicionei a seguinte configuração de volume para o serviço **php**:

services:

php:

volumes:

- .:/var/www/html

Essa linha mapeia o diretório atual do host para o diretório /var/www/html dentro do contêiner, garantindo a persistência dos dados.

1. Utilize o **Docker Compose** para separar os serviços **PHP-FPM** e **Nginx** em contêineres distintos.

Para uma arquitetura mais modular e escalável, utilizei o Docker Compose para definir dois serviços distintos:

* **Serviço php**: Responsável pelo processamento do PHP utilizando o PHP-FPM.
* **Serviço nginx**: Servidor web que encaminha as requisições para o serviço **php**.

O arquivo docker-compose.yml foi configurado da seguinte maneira:

version: '3.8'

services:

php:

build: .

container\_name: php-container

volumes:

- .:/var/www/html

expose:

- "9000"

nginx:

image: nginx:latest

container\_name: nginx-container

volumes:

- .:/var/www/html

- ./nginx/nginx.conf:/etc/nginx/nginx.conf

ports:

- "8080:80"

depends\_on:

- php

Essa configuração faz com que o **Nginx** e o **PHP-FPM** sejam executados em contêineres separados, permitindo uma melhor gestão e escalabilidade dos serviços.

Aluno: Lucas de Oliveira dos Prazeres

RGM: 28797710