# AULA 2: APRENDA CI/CD NA PRÁTICA E AUTOMATIZE DO BUILD AO DEPLOY”

---

### 1. Introdução ao Docker Compose

O Docker Compose é uma ferramenta que utiliza um arquivo YAML para orquestrar múltiplos containers Docker, facilitando a execução de aplicações complexas.

### 2. Estrutura do arquivo docker-compose.yml

- ➔ O arquivo define os serviços, volumes e variáveis de ambiente necessários para a aplicação.
- ➔ A importância da indentação correta no YAML para que o Docker entenda as configurações.

### 3. Comando docker compose up

Este comando constrói a imagem e inicia todos os contêineres definidos no arquivo docker-compose.yml em um único passo, simplificando o processo de execução da aplicação.

### 4. Variáveis de Ambiente

As variáveis de ambiente são utilizadas para manter a segurança e a flexibilidade das configurações, permitindo que diferentes ambientes (desenvolvimento, produção) sejam facilmente gerenciados.

### 5. CI/CD (Continuous Integration/Continuous Deployment)

O conceito de CI/CD é introduzido como uma prática para automatizar o processo de desenvolvimento e entrega de software, garantindo que as aplicações sejam testadas e implantadas de forma consistente.

### 6. Conclusão

A aula enfatizou a importância do Docker Compose na automação e simplificação do desenvolvimento, além de preparar o caminho para a integração contínua e entrega contínua no fluxo de trabalho de DevOps.

---

## Questionário da aula com respostas:

**O que é o Docker Compose?**
Uma ferramenta que utiliza um arquivo YAML para orquestrar múltiplos containers Docker.

**Qual é a função do arquivo docker-compose.yml?**
Definir a estrutura e os serviços que serão utilizados em um projeto Docker.

**O que acontece quando você executa o comando docker compose up?**
Ele constrói a imagem e inicia os contêineres definidos no arquivo docker-compose.yml.

**Qual é a importância de usar variáveis de ambiente em um projeto Docker?**
Elas ajudam a manter a segurança e a flexibilidade das configurações.

**O que é CI/CD e qual é seu objetivo principal?**
Continuous Integration/Continuous Deployment; automatizar o processo de desenvolvimento e entrega de software.

---

## Passos da aula 2:

- **Criar o arquivo `docker-compose.yml` na pasta do projeto no VSCode**
- **Pedir ao Copilot ou ao Gemini para criar um docker compose baseado na minha aplicação**
- **Ajustar arquivo conforme necessidade do projeto**
- **Rodar no terminal o comando**

```
docker compose up
```

- **Verificar a API funcionando no `localhost:8000/docs`**
- **Comparar o trabalho feito antes a partir do ReadMe do projeto e como ficou mais simples agora com um comando apenas**
- **Apertar `Ctrl+C` no terminal ocasiona um "gracefully stopping" no run**
- **Criar um repositório novo no Git (dica: escrever `repo.new` no navegador é um atalho para isso), nomear como "ellis"**
- **Subir o projeto para lá**

- **Abrir o GitHub Actions, usar o template do Docker image, verificar e dar commit**
- **Ativar créditos no Google Cloud para o projeto alura-imersao**

---

# Exercício: Criando um Docker Compose para uma Aplicação Web

**Objetivo: Criar um arquivo docker-compose.yml para uma aplicação web simples que utiliza um servidor web (como Nginx) e um banco de dados (como PostgreSQL). Instruções:**

1. **Estrutura do Projeto:**
- Crie uma nova pasta chamada meu-projeto
- Dentro dessa pasta, crie duas subpastas: web e db

2. **Arquivo Dockerfile para o Servidor Web:**
- Na pasta web, crie um arquivo chamado Dockerfile que utiliza a imagem do Nginx
- Adicione um arquivo HTML simples (por exemplo, index.html) que será servido pelo Nginx

3. **Configuração do Docker Compose:**
- Na raiz da pasta meu-projeto, crie um arquivo chamado docker-compose.yml
- Defina dois serviços: um para o Nginx (servidor web) e outro para o PostgreSQL (banco de dados)
- Configure o Nginx para servir o arquivo index.html que você criou
- Configure o PostgreSQL com um nome de usuário e senha padrão

4. **Volumes:** Utilize volumes para persistir os dados do banco de dados, garantindo que os dados não sejam perdidos quando o contêiner for parado.

6. **Executando o Projeto:**
- Após criar o docker-compose.yml, execute o comando docker compose up na raiz do seu projeto.

- Acesse o servidor web pelo navegador em http://localhost e verifique se a página HTML está sendo exibida.

7. **Reflexão:** Após completar o exercício, escreva uma breve reflexão sobre como o uso do Docker Compose facilitou a configuração e execução da sua aplicação.

**O uso do Docker Compose facilitou bastante o processo de subir múltiplos serviços ao mesmo tempo. Com um único comando (`docker compose up`), consegui iniciar um servidor web e um banco de dados PostgreSQL, com os volumes e variáveis de ambiente já configurados. Isso reduz erros manuais e melhora a organização do ambiente, além de ser facilmente replicável por qualquer pessoa da equipe.**