Automação de Processos com Docker e Airflow

Carlos Eduardo Rocha Miranda

1. **Aula:** Identifique as principais diferenças com relação ao exemplo que fizemos em sala;
   - **funções utilizadas;**

     Variáveis de Ambiente: Diferentemente do exemplo em sala, o Try Docker trabalhou com variáveis de ambiente para poder rodar o seu programa.

     Expose: No Try Docker utilizamos o EXPOSE para definir a porta que o container vai escutar.

   - **containers, aplicações e dependências;**
     - No primeiro exemplo criamos dois containers: Um chamado test-redis e outro test-webapp. No primeiro, rodamos um simples servidor web feito com nodejs e o framework Express. No segundo, Redis para armazenar dados.
     - No primeiro exemplo criamos dois containers: Um chamado web e outro redis. No primeiro temos uma aplicação web feita com Python utilizando o Flask. No segundo temos o Redis.
     - O segundo exemplo utilizou o bind para que o código dentro do container seja atualizado sem que seja necessário reconstruir a sua imagem.
   - **forma de comunicação entre containers;**
     - No primeiro e no segundo, a comunicação é feita através da porta 6379. A diferença é a ferramenta que realiza essa comunicação, no Node é redis.createClient e no Python é redis.Redis.
   - **comandos e instruções não usados nos exemplos em sala**
     - docker compose up –d
       - Para inicializar no background.
     - docker compose ps
       - Para ver todos os compose.
     - docker compose run web env
       - Para executar apenas um comando específico. (Neste caso, para ver as variáveis disponíveis para o container web.

- docker compose stop
  - Para parar o compose.
- docker compose down –volumes
  - Remove os containers e todos os dados usados pelo container redis.