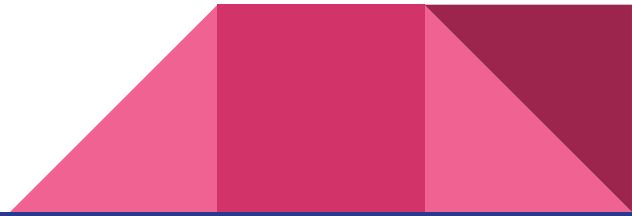
Docker

Huggo Parcelly da Silva	120210155
Igor Tejo Bezerra R. Nogueira	120210131
Marcus Vinicius Norberto Ideao	119210145
Raphael Cavalcante Ramos	120110277

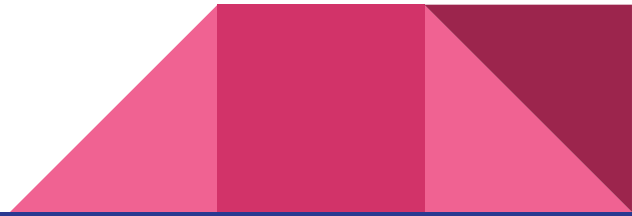
O que é Docker?

- Quem nunca ouviu a frase: "Na minha máquina funciona!"?
- O **Docker é um sistema de virtualização não convencional**. Mas o que isso quer dizer? Em virtualizações convencionais temos um software instalado na máquina *Host* que irá gerenciar as máquinas virtuais (ex.: VirtualBox, VMWare, Parallels e etc...).
-

Como funciona o Docker?



Por que ele é amplamente utilizado?



Casos de Uso do Docker

- **Desenvolvimento**

  O Docker facilita o desenvolvimento de aplicações, permitindo que os desenvolvedores criem ambientes consistentes e replicáveis para testes e desenvolvimento. Ele também facilita a colaboração entre equipes.

- **Implantação**

  O Docker simplifica a implantação de aplicações, garantindo que elas funcionem da mesma maneira em diferentes ambientes. Ele também permite a implantação rápida e fácil de atualizações.

- **Gerenciamento de Infraestrutura**

  O Docker é usado para gerenciar a infraestrutura, incluindo servidores, bancos de dados e outros serviços. Ele permite a otimização de recursos e a automatização de tarefas de administração.

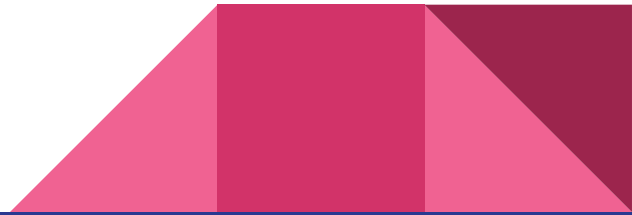
- **Microsserviços**

  O Docker é ideal para a arquitetura de microsserviços, permitindo que as aplicações sejam divididas em serviços independentes e escaláveis. Isso aumenta a flexibilidade e a manutenibilidade.

Imagem Docker

Uma **imagem Docker** é um arquivo leve e portátil que contém tudo o que é necessário para rodar um aplicativo, incluindo:

1. **Código do Aplicativo**: O código-fonte do aplicativo que você deseja executar.
2. **Dependências**: Bibliotecas e pacotes necessários para que o aplicativo funcione.
3. **Configurações**: Arquivos de configuração que definem como o aplicativo deve se comportar em diferentes ambientes.
4. **Sistema de Arquivos**: Uma estrutura de diretórios que contém todos os arquivos necessários para executar o aplicativo.



Explorando Docker Hub

- **Repositório de Imagens**

  O Docker Hub é um repositório público de imagens Docker. Ele oferece milhares de imagens prontas para uso, incluindo sistemas operacionais, linguagens de programação, bancos de dados e ferramentas de desenvolvimento.

- **Pesquisando Imagens**

  Você pode pesquisar por imagens Docker no Docker Hub por nome, descrição ou tag. Isso permite encontrar rapidamente a imagem ideal para seu projeto.

- **Compartilhando Imagens**

  Você pode hospedar suas próprias imagens Docker no Docker Hub, tornando-as disponíveis para outros usuários. Isso facilita a colaboração e o compartilhamento de código.

- **Gerenciamento de Repositórios**

  O Docker Hub oferece ferramentas para gerenciar seus repositórios, incluindo controle de acesso, controle de versão e histórico de alterações.

Como utilizar uma imagem pronta do Docker Hub

- Instalação do Docker (https://docs.docker.com/engine/install/ubuntu/)

  - sudo snap install docker

Fazendo o Pull da Imagem do Nginx

Nginx é um servidor web leve e amplamente utilizado

A principal função do Nginx é entregar arquivos HTML, CSS, JavaScript, imagens e outros conteúdos estáticos para os usuários da web. Quando você acessa um site, o Nginx pode ser o servidor que entrega os arquivos necessários para carregar a página no seu navegador.

Exemplo: Quando você cria uma página HTML simples e a executa com o Nginx, ele responde a solicitações HTTP e entrega os arquivos para o usuário que está acessando o site.

```
igortejo@Igor-Tejo:~$ sudo docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
302e3ee49805: Pull complete
cd986b3703ae: Pull complete
34a52cbc3961: Pull complete
d1875670ac8a: Pull complete
af17adb1bdcc: Pull complete
97182578e5ec: Pull complete
67b9310357e1: Pull complete
Digest: sha256:b5d3f3e104699f0768e5ca8626914c16e52647943c65274d8a9e63072bd015bb
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest
igortejo@Igor-Tejo:~$
```

Faz o download da última versão da imagem do Nginx.

Listar Imagens Baixadas

- Exibirá todas as imagens disponíveis localmente no Docker.

```
igortejo@Igor-Tejo:~$ sudo docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
nginx        latest    9527c0f683c3   6 weeks ago   188MB
igortejo@Igor-Tejo:~$
```

Rodar o Container a partir da Imagem

```
igortejo@Igor-Tejo:~$ sudo docker run -d -p 8080:80 --name meu_nginx nginx
1eabcb0e5aee61e1ac7d4605adbf7b66dee3f9c12dc5a32fcb22440466ece670
igortejo@Igor-Tejo:~$
```

- -d roda o container em segundo plano (detached mode).
- -p 8080:80 mapeia a porta 80 do container para a porta 8080 no host.
- --name meu_nginx nomeia o container para facilitar o uso.
- nginx é a imagem usada.

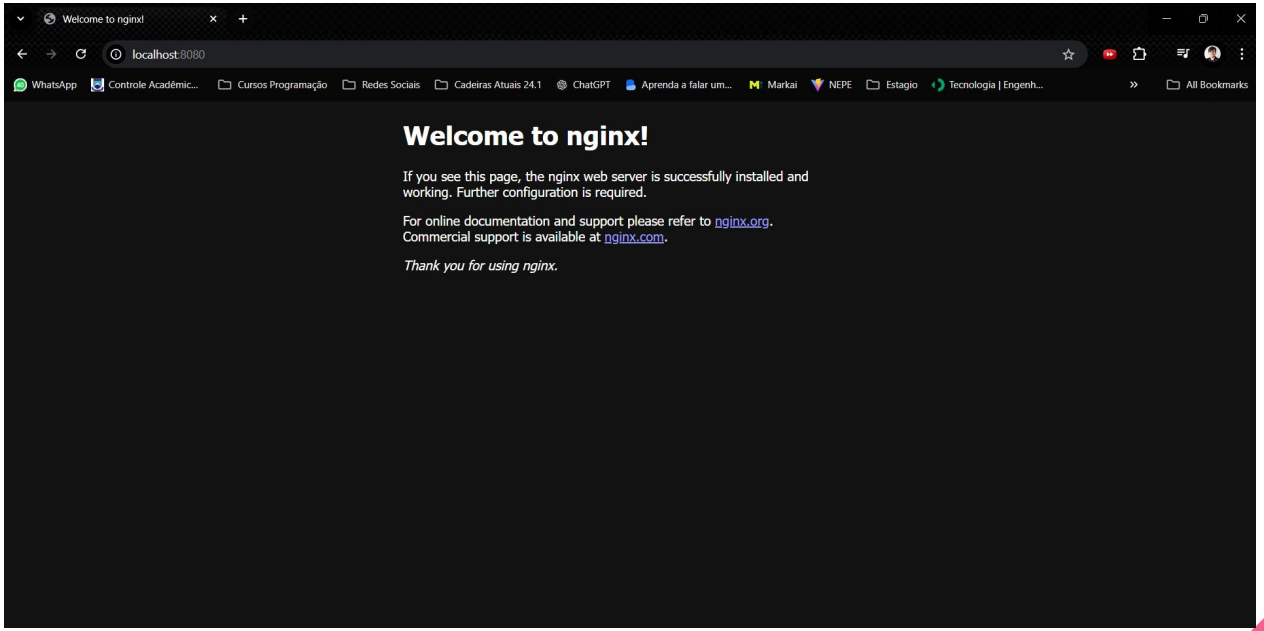
Listar Containers em Execução

```
igortejo@Igor-Tejo:~$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED              STATUS              PORTS                                   NAMES
1eabcb0e5aee   nginx     "/docker-entrypoint.…"   About a minute ago   Up About a minute   0.0.0.0:8080->80/tcp, :::8080->80/tcp   meu_nginx
igortejo@Igor-Tejo:~$
```

- Isso exibirá os containers ativos, incluindo o que acabou de rodar.
- Obs. sudo docker ps -a lista os containers que já foram parados também

Utilizar o Container

`http://localhost:8080`. mostra a página de boas-vindas padrão do Nginx, demonstrando que o container está funcionando.



Apagar um Container

- Primeiro utilizo o comando stop:

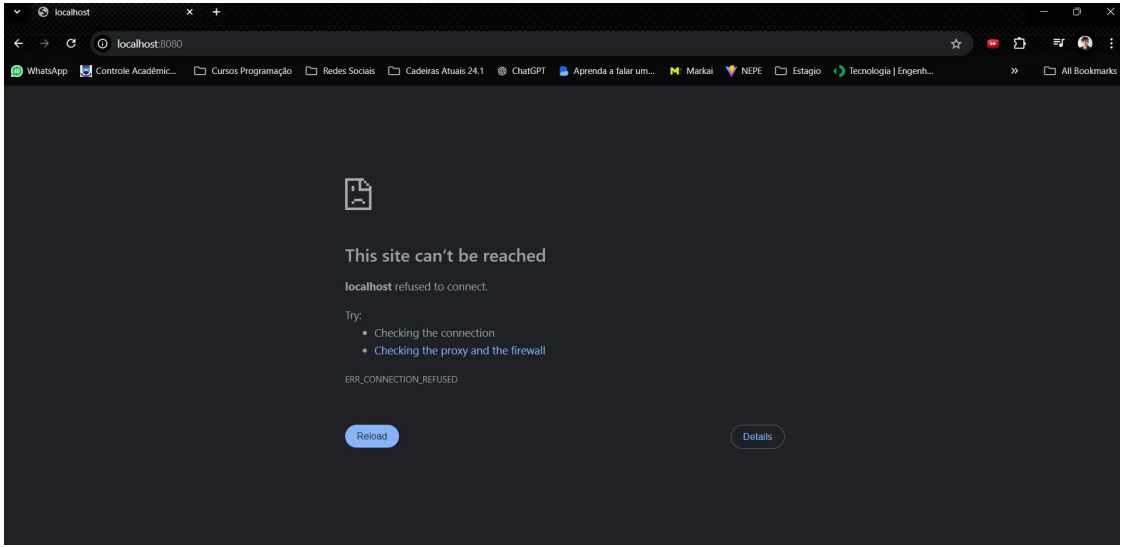
```
igortejo@Igor-Tejo:~$ sudo docker stop meu_nginx
meu_nginx
igortejo@Igor-Tejo:~$
```

- Remover container:

```
igortejo@Igor-Tejo:~$ sudo docker rm meu_nginx
meu_nginx
igortejo@Igor-Tejo:~$
```

Confirmar remoção do container

```
igortejo@Igor-Tejo:~$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
igortejo@Igor-Tejo:~$
```



Apagar uma Imagem

```
igortejo@Igor-Tejo:~$ sudo docker rmi nginx
Untagged: nginx:latest
Untagged: nginx@sha256:b5d3f3e104699f0768e5ca8626914c16e52647943c65274d8a9e63072bd015bb
Deleted: sha256:9527c0f683c3b2f0465019f9f5456f01a0fc0d4d274466831b9910a21d0302cd
Deleted: sha256:c177c57aa7996656fd3126345000236f261e39a0609112a0849f58e9dae0978b
Deleted: sha256:7936e62f79dd6e4850dc94811247b8f94824aad3378e422f86b692542c6d703c
Deleted: sha256:aeda24a2d75e188b2b089b91de16da259ae8e27acbeb7a658fd8a3e020eba6ec
Deleted: sha256:256492ed14f20b1f4028f85760847e8efba76657144306c34cc7f12a6b3004e7
Deleted: sha256:d6ef76047425e4389ddc16af812246a8f1f5cca433b5867cc1d3520394f3dc4a
Deleted: sha256:3d3991af12fbd1d1734d066d4f4e5b81d6ebf6597d9a846befb169ecf7dae930
Deleted: sha256:8d853c8add5d1e7b0aafc4b68a3d9fb8e7a0da27970c2acf831fe63be4a0cd2c
igortejo@Igor-Tejo:~$
```

Confirmar remoção da imagem

```
igortejo@Igor-Tejo:~$ sudo docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
igortejo@Igor-Tejo:~$
```

Criar sua própria imagem

- Criar uma Conta no Docker Hub (https://hub.docker.com/)
- Instalar Docker
- Criar um Dockerfile
- Construir a Imagem Docker
- Enviar a Imagem para o Docker Hub

Criar um Dockerfile

- O Dockerfile é um arquivo de texto que contém todas as instruções necessárias para criar a imagem Docker.
- Crie um diretório no seu sistema para o projeto:

```
igortejo@Igor-Tejo:~$ mkdir primeiro_projeto
igortejo@Igor-Tejo:~$ cd primeiro_projeto/
igortejo@Igor-Tejo:~/primeiro_projeto$
```

Criar um Dockerfile

Crie um arquivo Dockerfile no diretório

```
# Usar uma imagem base do Nginx Alpine para otimizar o tamanho da imagem final
FROM nginx:alpine AS builder

# Definir o diretório de trabalho dentro do container
WORKDIR /usr/share/nginx/html

# Copiar os arquivos de configuração do Nginx (opcional)
# COPY nginx.conf /etc/nginx/nginx.conf

# Copiar os arquivos HTML/CSS/JS (ou qualquer outro conteúdo estático)
COPY ./public/ .

# Expor a porta 80 para acesso HTTP
EXPOSE 80

# Comando para iniciar o servidor Nginx
CMD ["nginx", "-g", "daemon off;"]
~
```

Construir a Imagem Docker

```
igortejo@Igor-Tejo:~/primeiro_projeto$ sudo docker build -t igortejo/primeiro_projeto .
[+] Building 7.5s (8/8) FINISHED                                        docker:defaul
 => [internal] load build definition from Dockerfile                             0.0
 => => transferring dockerfile: 539B                                             0.0
 => [internal] load .dockerignore                                                0.0
 => => transferring context: 2B                                                  0.0
```

- O -t no comando docker build é uma opção que permite você dar um nome e, opcionalmente, uma versão para a sua imagem Docker.
- . : Este ponto indica o diretório onde o Docker deve procurar o Dockerfile. Neste caso, significa que o Dockerfile está no diretório atual.

Testar a Imagem Localmente

```
igortejo@Igor-Tejo:~/primeiro_projeto$ sudo docker run -d -p 8080:80 igortejo/primeiro_projeto
f5a4c867af4c7865d156686f26fee15729a8dcd802ed8762e27ecd29746c538f
```

Fazer Login no Docker Hub

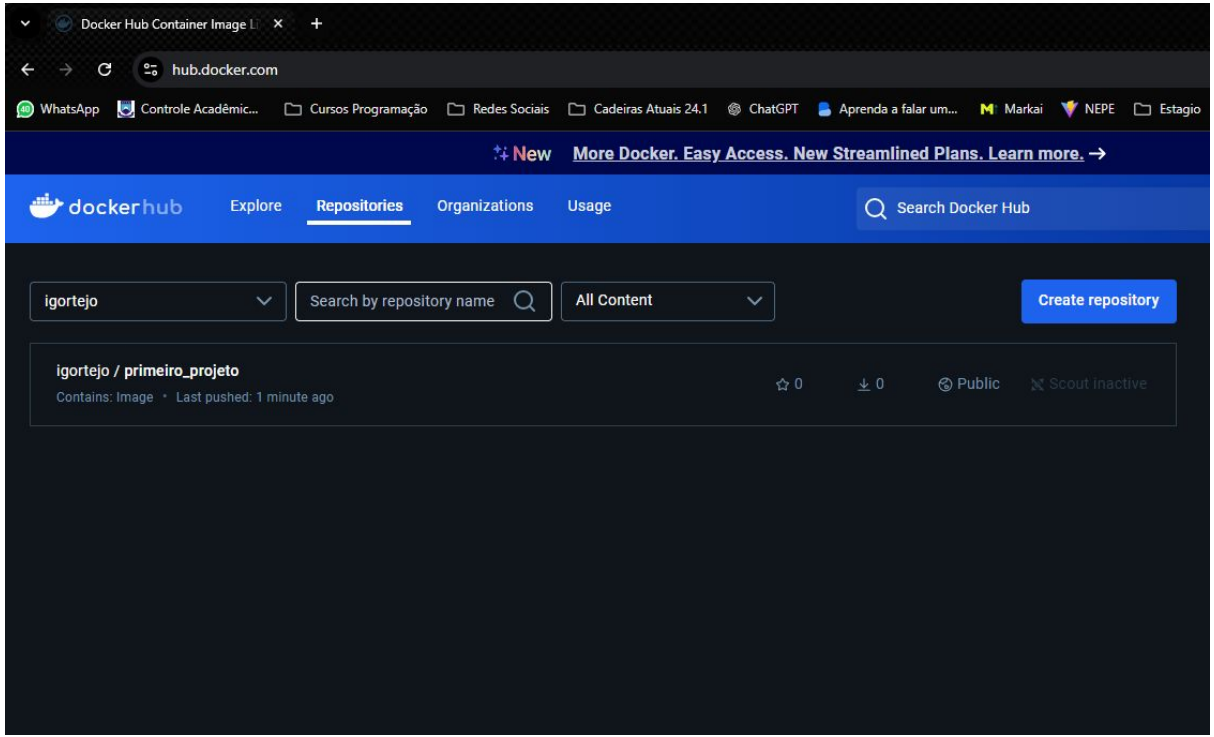
```
igortejo@Igor-Tejo:~/primeiro_projeto$ sudo docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID,
ad over to https://hub.docker.com to create one.
Username: igortejo
Password:
WARNING! Your password will be stored unencrypted in /root/snap/docker/2932/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
igortejo@Igor-Tejo:~/primeiro_projeto$
```

Enviar a Imagem para o Docker Hub

```
igortejo@Igor-Tejo:~/primeiro_projeto$ sudo docker push igortejo/primeiro_projeto
Using default tag: latest
The push refers to repository [docker.io/igortejo/primeiro_projeto]
adba59b4c014: Pushed
5f70bf18a086: Pushed
b0f60355fd52: Mounted from library/nginx
027907faf592: Mounted from library/nginx
11134cc97d7f: Mounted from library/nginx
f7a5847cdca9: Mounted from library/nginx
aec1e8cf14f5: Mounted from library/nginx
717b3a077b07: Mounted from library/nginx
2ff96b2e5450: Mounted from library/nginx
63ca1fbb43ae: Mounted from library/nginx
latest: digest: sha256:62d543faa4cdcea63f04fe3fb9d1e80745c36618465e72b13f50de1cf59f87a3 size: 2402
igortejo@Igor-Tejo:~/primeiro_projeto$
```

Verificar a Imagem no Docker Hub



Utilização da imagem por outros usuários

- Outros usuários agora poderão baixar e usar a imagem com o comando

  - sudo docker pull igortejo/primeiro_projeto

Atualizar a Imagem

Caso você precise atualizar sua imagem no futuro, siga esses passos:

1. Modifique o Dockerfile.
2. Construa a imagem novamente com o comando docker build.
3. Substitua a imagem antiga no Docker Hub com um novo push:

    - **docker push igortejo/primeiro_projeto**

Demonstração de uso do Docker

- **Tecnologias utilizadas**

  NodeJs, Express, TypeScript, Prisma ORM, PostgreSQL, Docker

- **Sumário da demonstração**

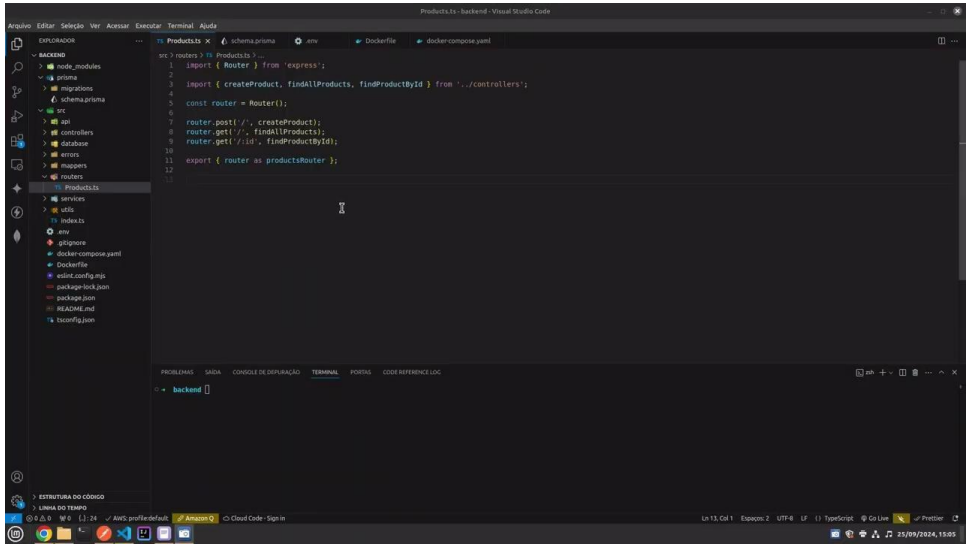
  Aplicação rodando localmente, com banco de dados no container

  Aplicação e banco de dados rodando em container, utilizando os comandos docker manualmente para criar o container

  Aplicação e banco de dados rodando em container, utilizando o docker-compose para criação dos containers

Demonstração de uso do Docker

- **Vídeo demonstrativo**



Links finais

https://github.com/MarcusNoberto/sysadmin_docker/tree/main (Repositorio Git)