

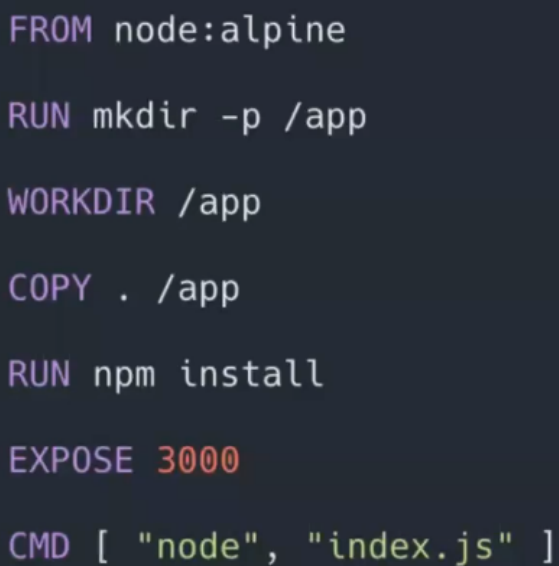
Container: ambiente 'fechado' com programas (node, java, bd ...) e versões específicas, na qual irei desenvolver meu código

O **docker** cria containers

O docker trabalha com **imagens**, elas são como se fossem um .iso de programas como node, mongodb, mysql, python, ubuntu ...

DockerFile: receita de bolo de como a aplicação vai funcionar

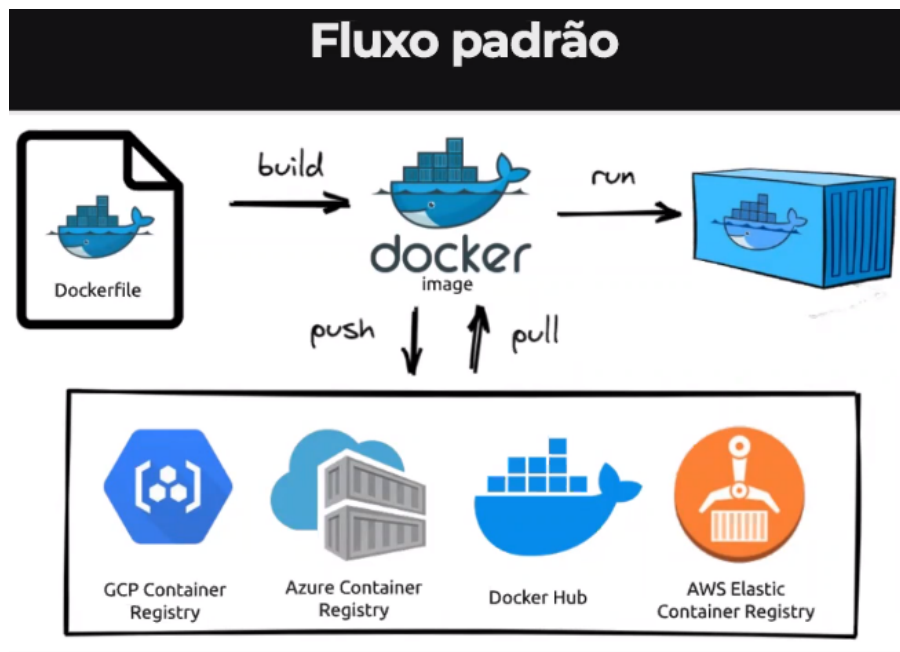
- **from:** sistema que deve ser usado, *nesse caso é o node alpine*
- **run:** comando para rodar o app no sistema
- **workdir:** é a pasta que será usada para rodar a aplicação
- **copy:**
- **run:** comando para rodar a aplicação
- **expose:** porta para rodar a aplicação
- **cmd:** comandos para rodar a aplicação

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top left corner. It displays a Dockerfile with the following commands:

```
FROM node:alpine
RUN mkdir -p /app
WORKDIR /app
COPY . /app
RUN npm install
EXPOSE 3000
CMD [ "node", "index.js" ]
```

Como funciona o docker na nuvem?

- em nível local o docker funciona com build/run
- em nível de nuvem funciona com aws, ms, google ... mas o mais comum é o **docker hub**

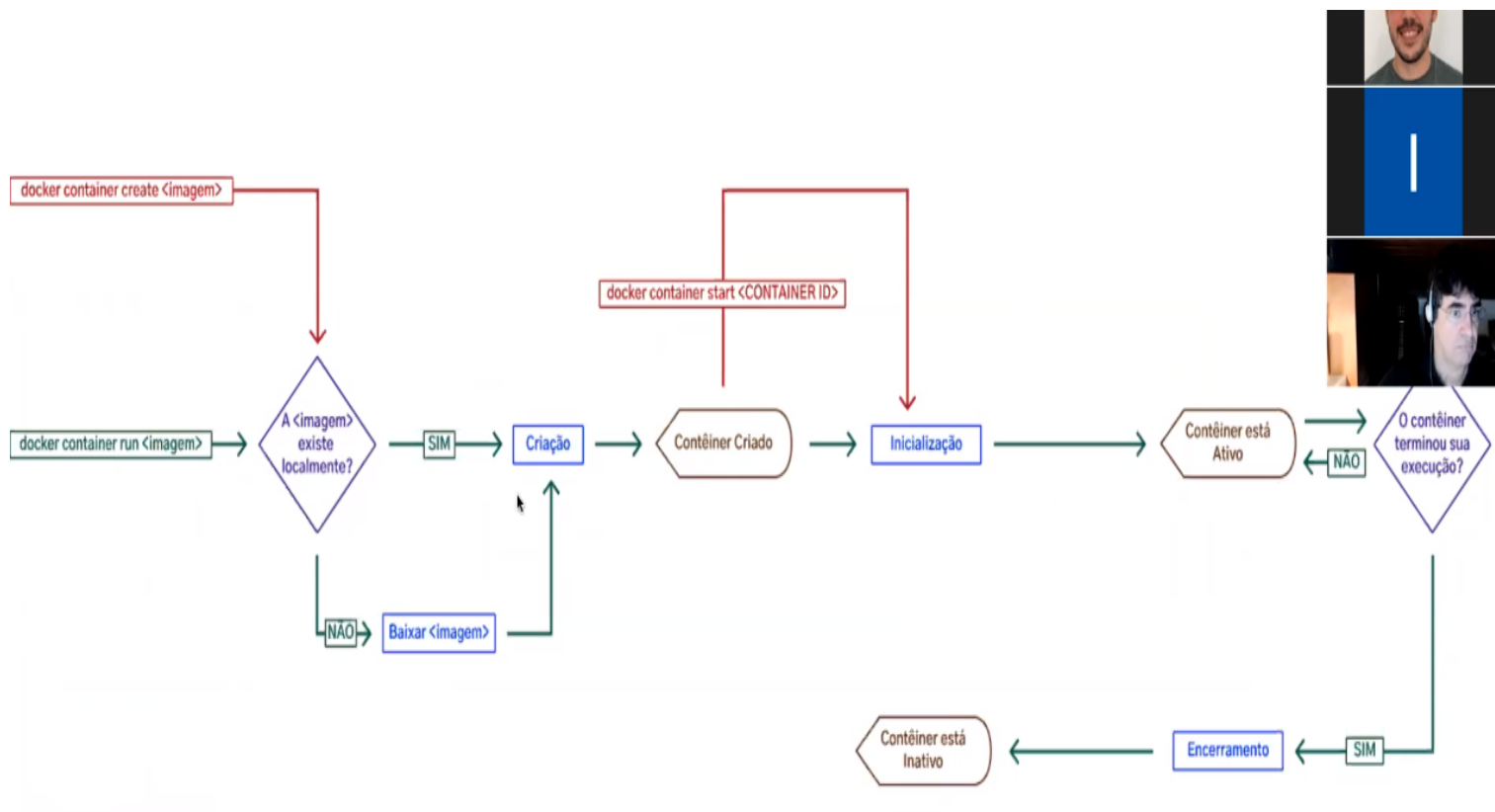


Testando containers na nuvem

- Cadastro: <https://hub.docker.com/>
- Brincar: <https://labs.play-with-docker.com/>

Usando o docker

1. **docker -v**: para ver a versão instalada na máquina
2. **docker container ls -a** ou **docker container ps -a**: visualizar a lista de containers
3. **docker container run [nome-pacote]**: inicia (e se necessário baixar) um container



Usando docker do ubuntu

1. **docker container run -it ubuntu bash**: vai baixar e executar o ubuntu, além de abrir o terminal do ubuntu (bash)
2. **[comandos terminal linux]**: posso escrever qualquer comando de bash

Brincando com containers

docker container run: apenas baixa e cria um container

1. **docker container ls -a**: Exibe os containers ativos e inativos
2. **docker container start [id_container]**: pega o container baixado para ser trabalhado
3. **docker container attach [id_container]**: entra dentro do container, e consigo navegar por dentro dele
4. **docker container run -dit [nome_container]**: este container vai ficar sendo executado em background
5. **docker container stop [id_container]**: para a execução do container
6. **docker container start [id_container]**: inicia a execução do container
7. **docker container restart [id_container]**: re inicia a execução do container
8. **docker container rm [id_container]**: remove o container, mas ele tem que estar pausado
9. **docker container rm -f [id_container]**: remove o container, ele estando em execução ou não
10. **docker container prune**: remove TODOS os containers da máquina

wfe