



DEVOPS BÁSICO

Fabrício Veronez - Aula 01

Professores

FABRÍCIO VERONEZ

Professor Convidado

Fundador da Formação KubeDev, é um grande apaixonado por tecnologias. Com mais de 13 anos de experiência no mercado de tecnologia, atuando como desenvolvedor e arquiteto em projetos de pequeno e grande porte, atualmente dedica-se a transformar a carreira de profissionais de TI, compartilhando conhecimento sobre o universo de containers e a cultura DevOps. Em 2018, passou a compartilhar a sua experiência através da criação de conteúdos. Foi assim que surgiu o seu site veronez.dev e, em seguida, o seu canal no YouTube.

MARCO AURÉLIO SOUZA MANGAN

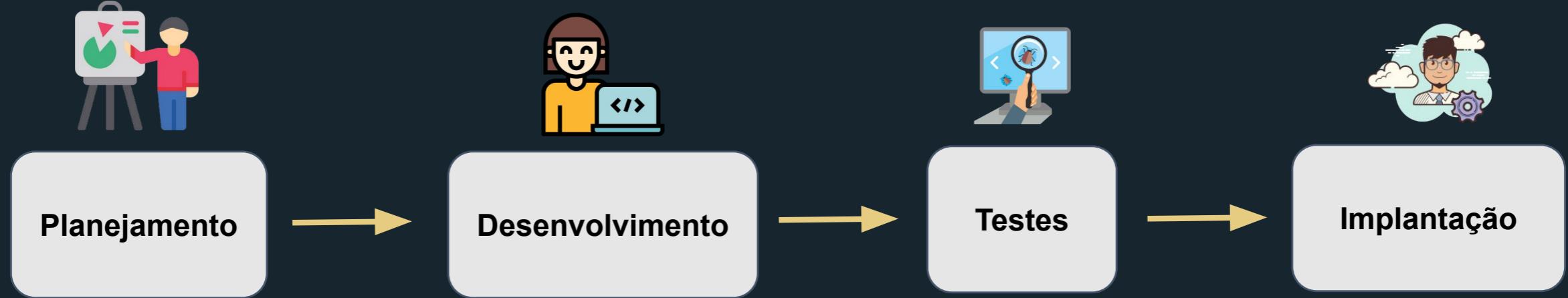
Professor PUCRS

Possui graduação em Ciências da Computação pelo Instituto de Informática/UFRGS, mestrado em Ciências da Computação pelo Instituto de Informática/UFRGS e doutorado em Engenharia de Sistemas e Computação pelo PESC/UFRJ. Atualmente, é professor da Faculdade de Informática/PUCRS e da Faculdade Senacrs Porto Alegre. Tem experiência na área de Ciência da Computação, com ênfase em linguagens de programação, atuando principalmente nos seguintes temas: engenharia de software, CSCW, ambientes de desenvolvimento de software, reutilização de software e mecanismos de cooperação.

Ementa da disciplina

Introdução aos fundamentos de gerência de configuração. Estudo sobre Integração contínua (CI). Utilização de contêineres, ferramentas e ambientes direcionados ao desenvolvimento de software como Git, GitHub, Maven, Gradle, Npm, Yarn, GitHub Actions, Jenkins, Travis e Docker.





Dev



Ops



Dev



Ops



Dev



Ops



Dev



- Conhece profundamente programação
- Tem como objetivo entregar novos recursos
- Conhecimento mínimo ou nulo de infraestrutura
- Não tem contato com o processo de execução do software em um ambiente de produção

Ops



- Conhece profundamente infraestrutura
- Tem como objetivo manter a solução estável
- Conhecimento mínimo ou nulo de programação
- Não tem contato com o processo desenvolvimento e compilação do software

DevOps



- Interesses em comum, foco no produto
- Comunicação ágil e simplificada
- O objetivo é sempre resolver o problema e aprender com ele
- 3 maneiras: Fluxo, Feedback, Aprendizado Contínuo e Experimentação

Primeira maneira: Fluxo

- Análise e otimização dos processos
- Inclusão de testes
- Integração contínua e deploy contínuo
- Entregas de baixo risco

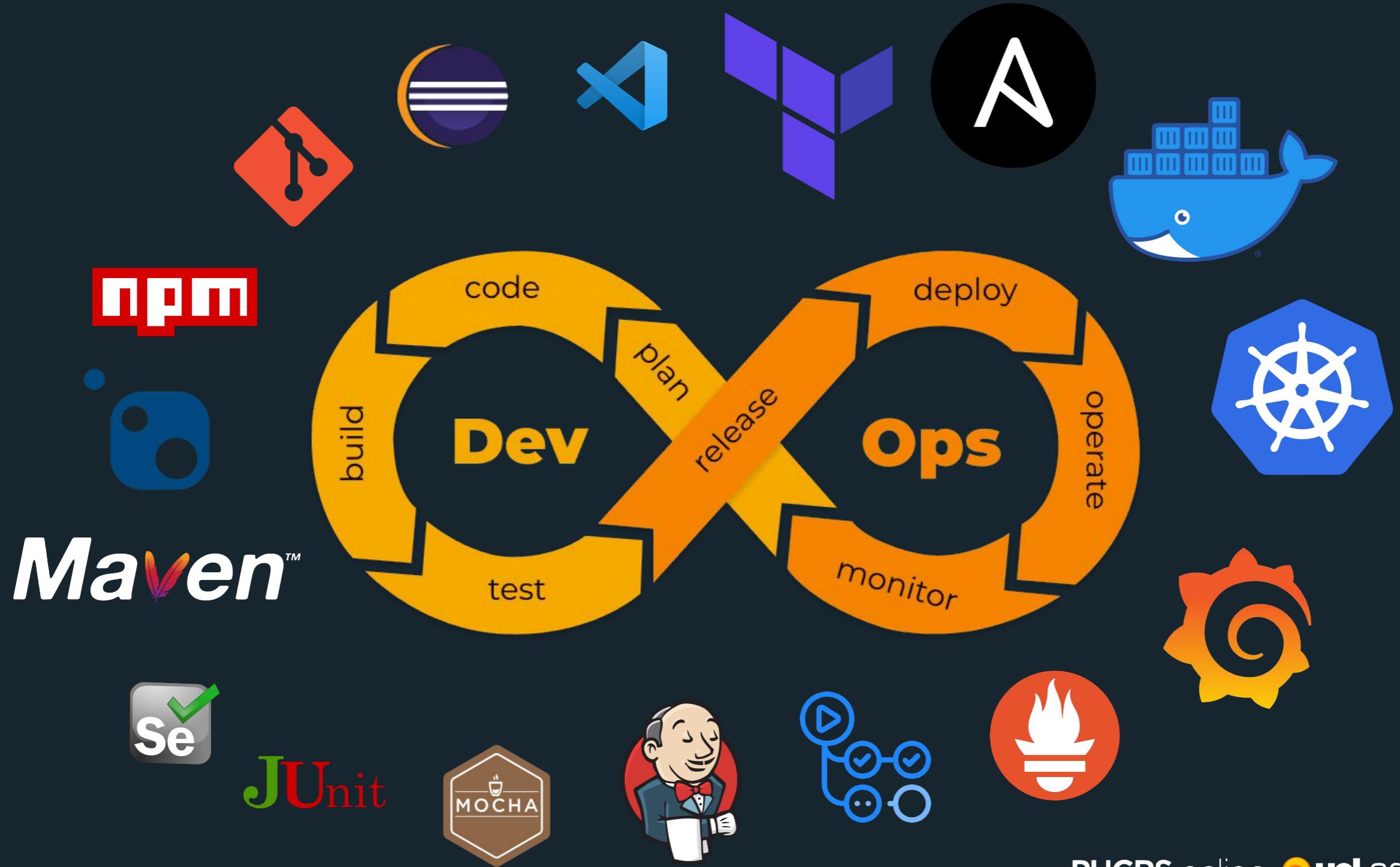
Segunda maneira: Feedback

- **Implementar e coletar métricas**
- **Observabilidade**
- **Teste A/B**
- **Feedback dos resultados para replanejamento**

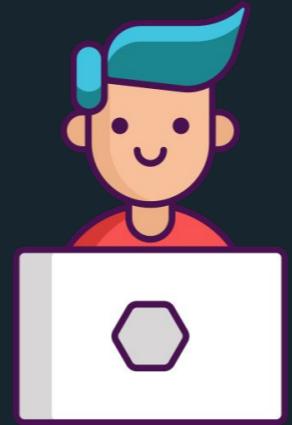
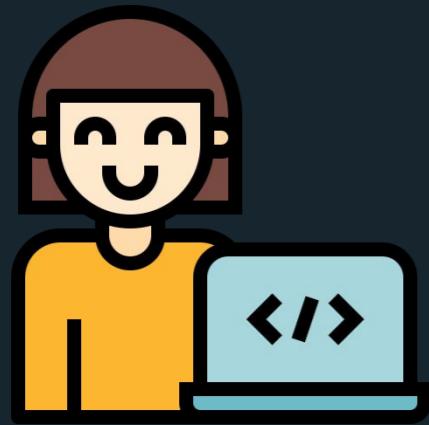
Terceira maneira: Feedback

- **Aprender com os erros**
- **Experimentação controlada**
- **Disseminar o conhecimento e padronizar o que dá certo**





DevOps



Boas práticas de Dev para DevOps

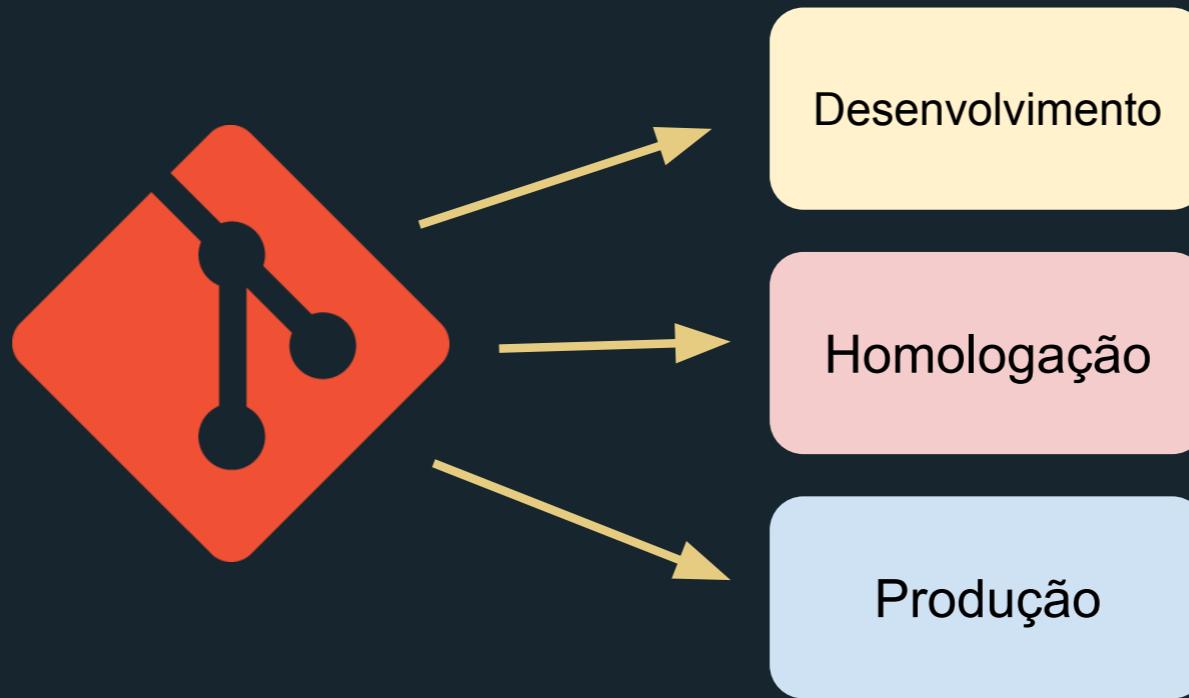
Fabrício Soares Veronez

Twelve Factor Apps

- Utilizar formatos declarativos para configuração de ambientes para facilitar automação
- Garantir a maior portabilidade possível entre ambientes
- Ter compatibilidade com plataformas de nuvem, sem depender de servidores e administração de sistemas
- Minimizar o impacto de migração entre ambientes de desenvolvimento e produção
- Ser escalável sem alterações significativas

01-Base de Código (Codebase)

Cada aplicação possui uma base de código única e centralizada. Todos os deploys são reflexo da base de código



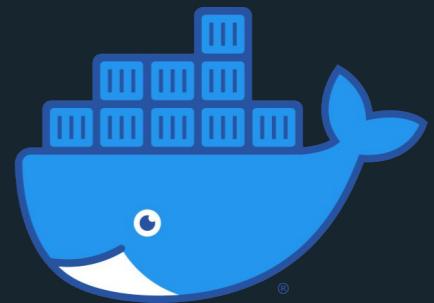
02-Dependências (Dependencies)

Declare as dependências das aplicações de forma explícita com gerenciadores de pacote



03-Configurações (Config)

Armazene as configurações no ambiente



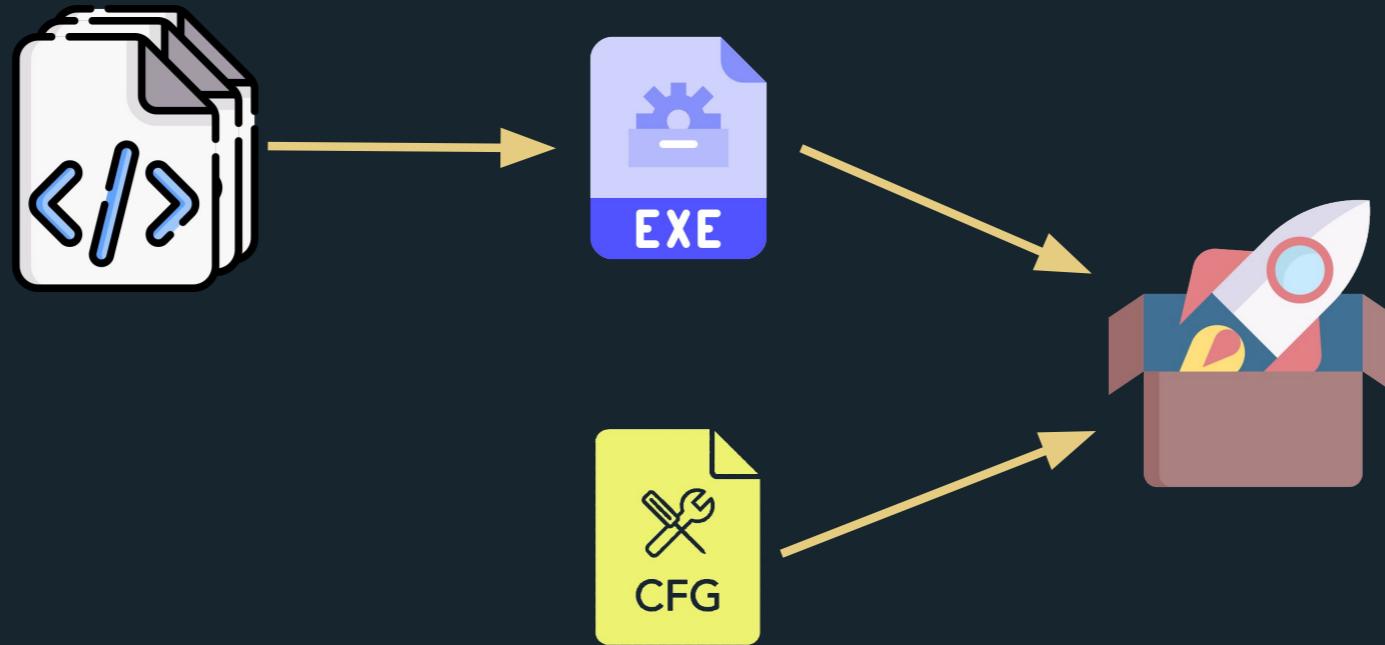
04-Backing services (Serviços de Apoio)

Trate serviços como recursos.



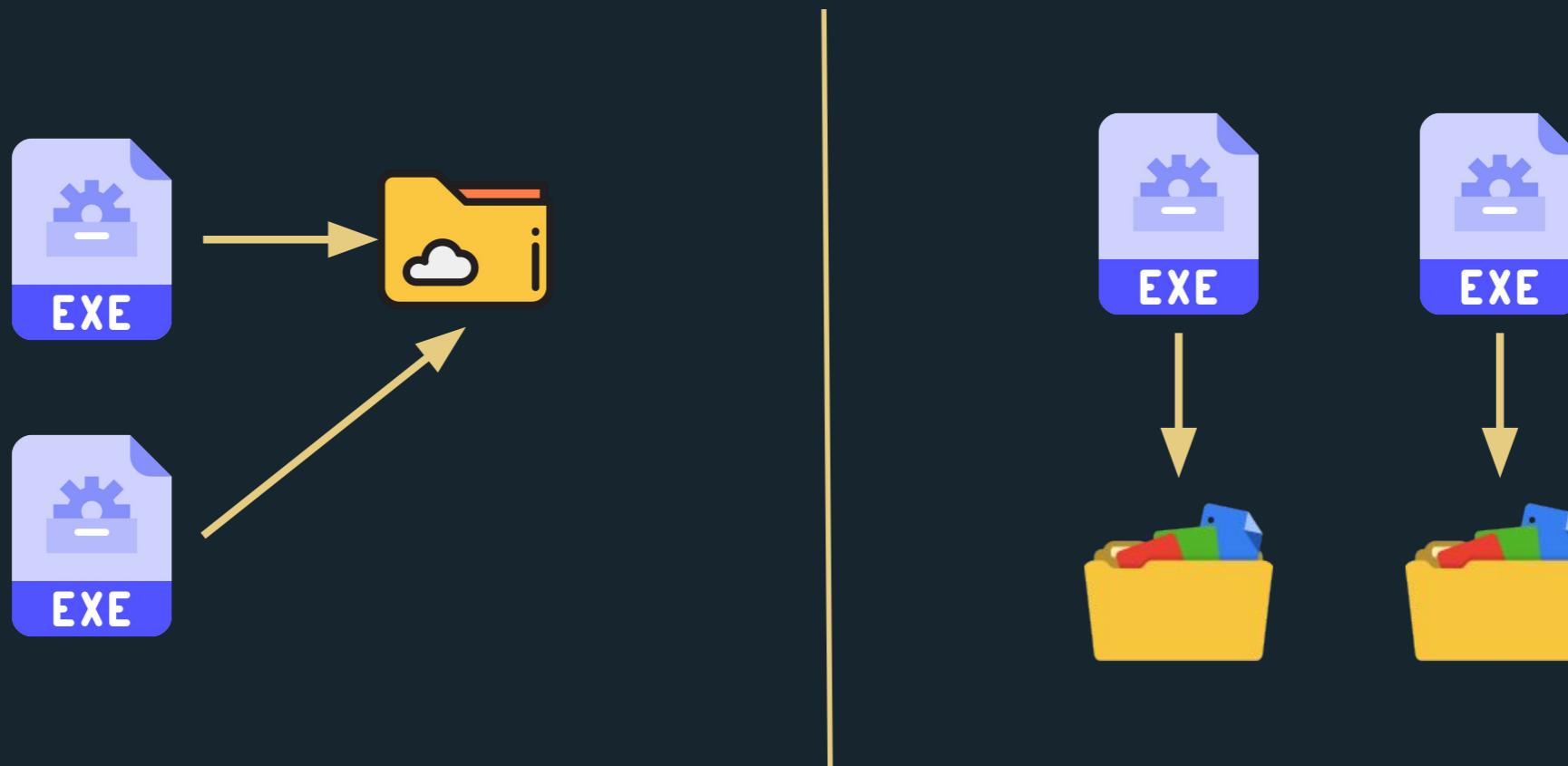
05-Construa, lance, execute (Build, release, run)

Separe os estágios de build e release dos projetos



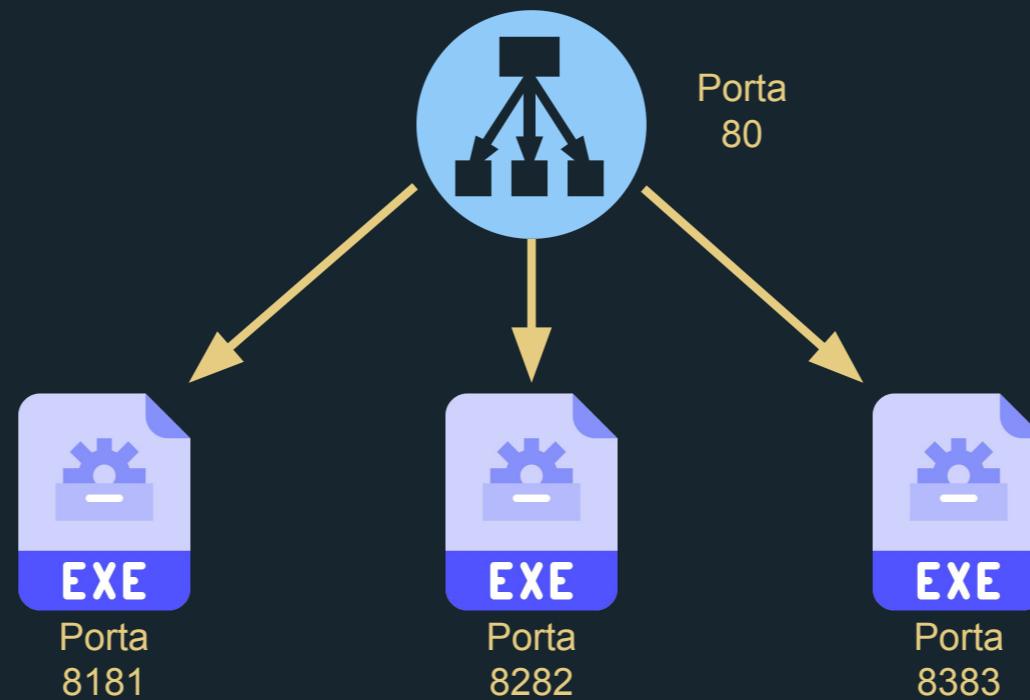
06-Processos (Processes)

Execute a aplicação como um ou mais processos que não armazenam estado



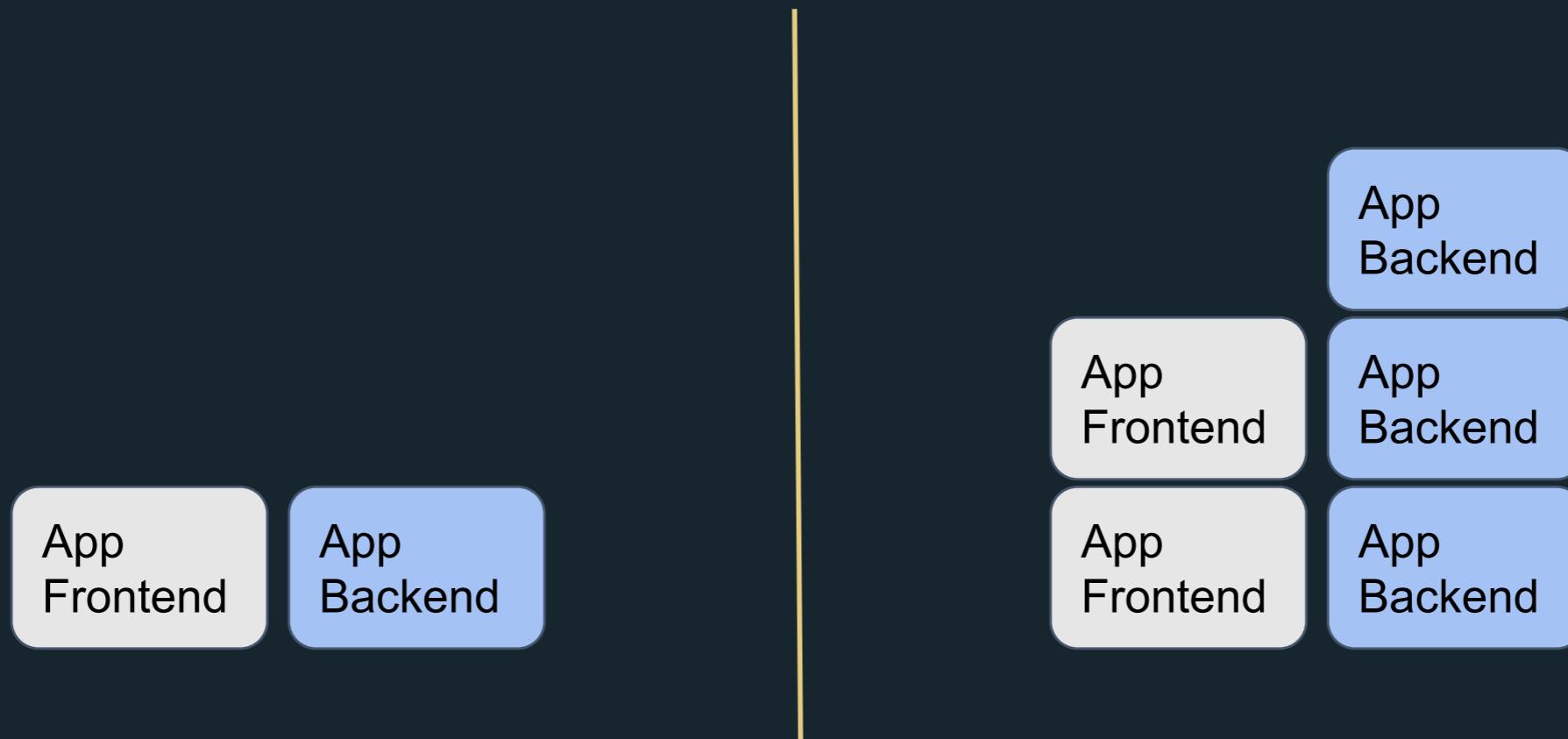
07-Vínculo de Portas (Port binding)

Utilizar port binding para expor serviços



08-Concorrência (Concurrency)

Escale através do processo modelo



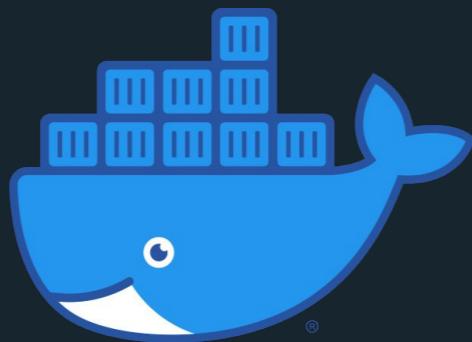
09-Descartabilidade (Disposability)

Processos com rápida inicialização e encerramento gracioso



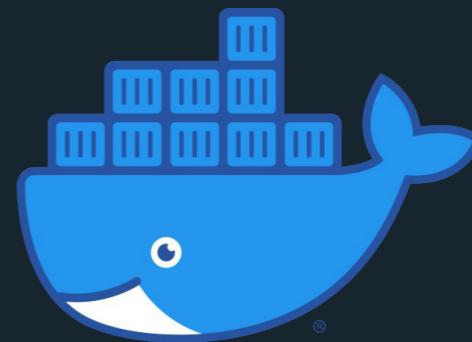
10-Paridade entre desenvolvimento e produção (Dev/prod parity)

Mantenha o desenvolvimento, homologação e produção o mais similares possível



DEV

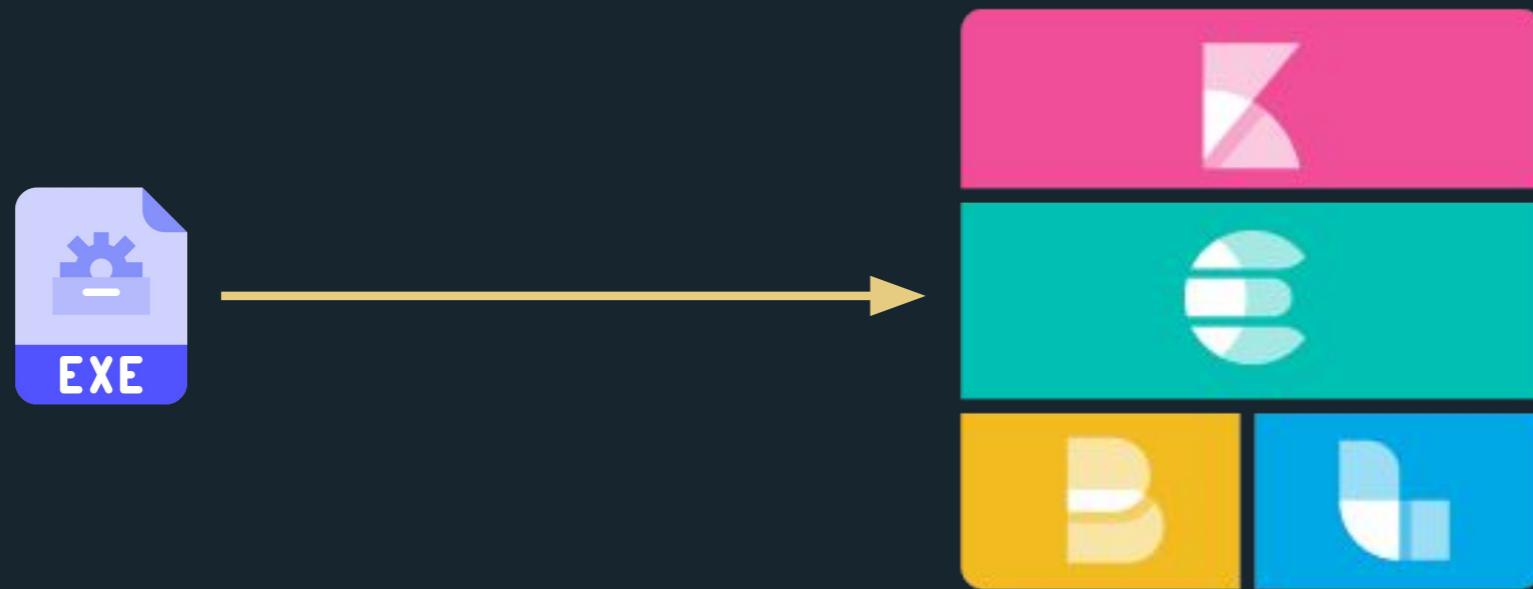
=



PROD

11-Logs (Logs)

Trate logs como fluxos de eventos

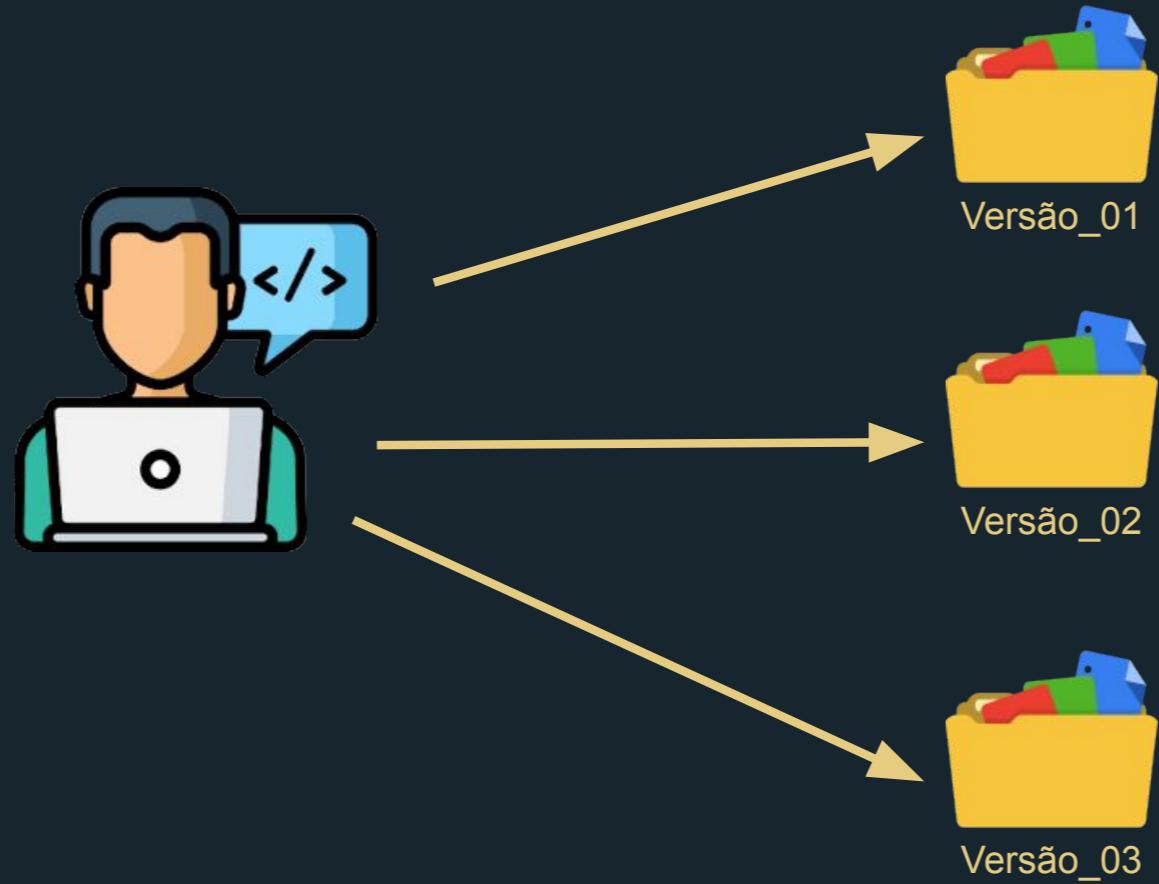


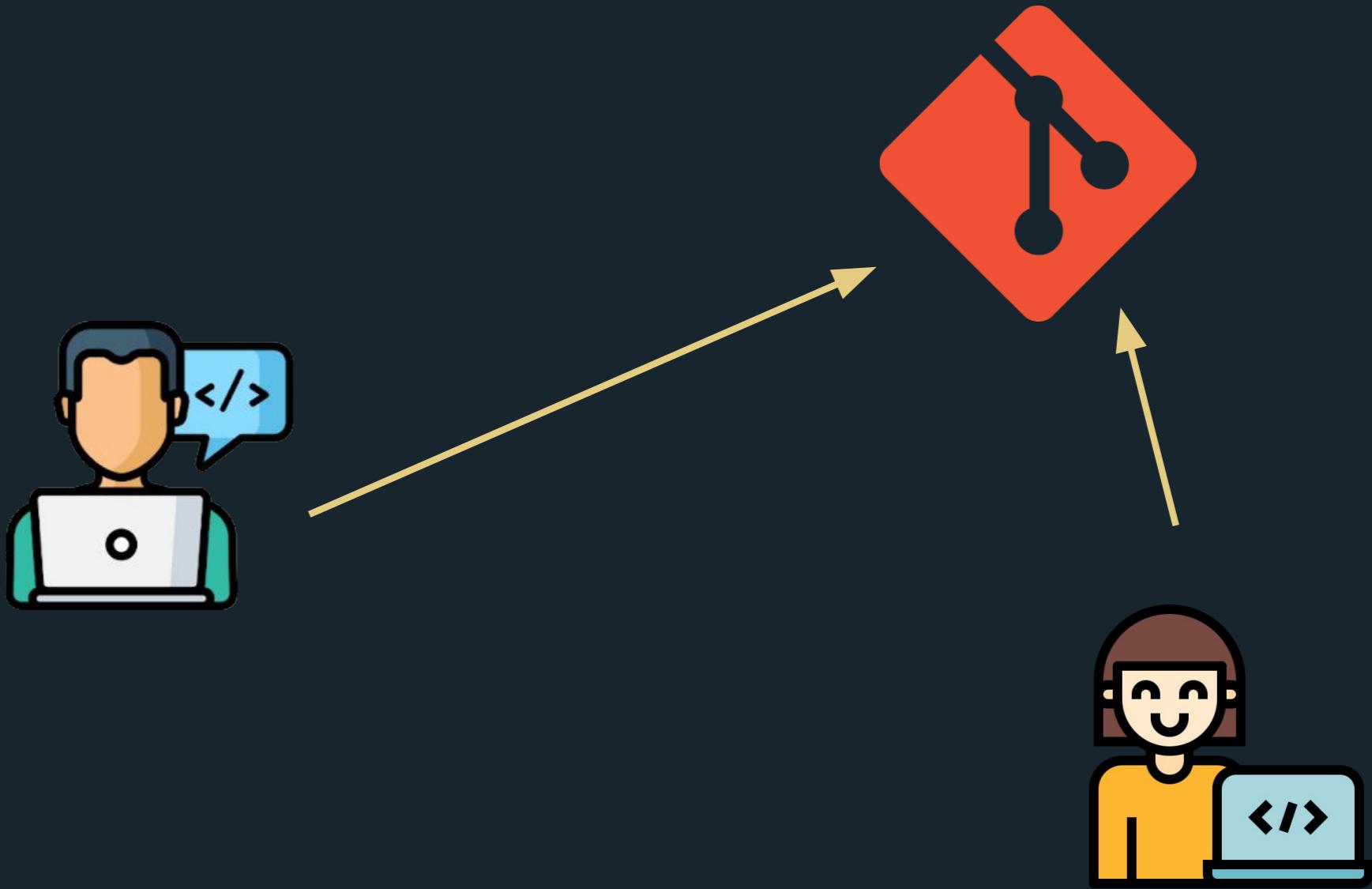
12-Processos administrativos (Admin processes)

Rode tarefas de administração/gestão em processos pontuais



Versionamento de Código





SSH



Introdução à Containers

Fabrício Soares Veronez



MongoDB®

RabbitMQ



MongoDB®

RabbitMQ

“Na minha máquina
funciona...”





O'Connor

O'Connor
A division of the
Stobart
Group

SCANIA

R420

PN57 JZ0

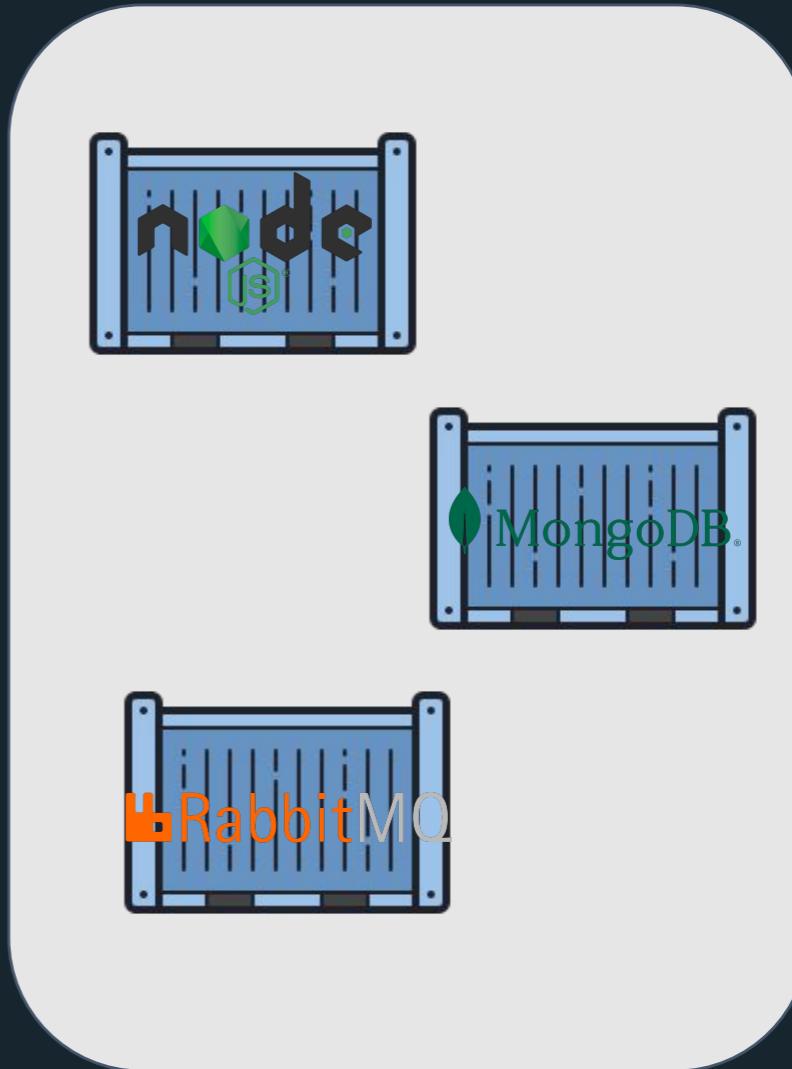
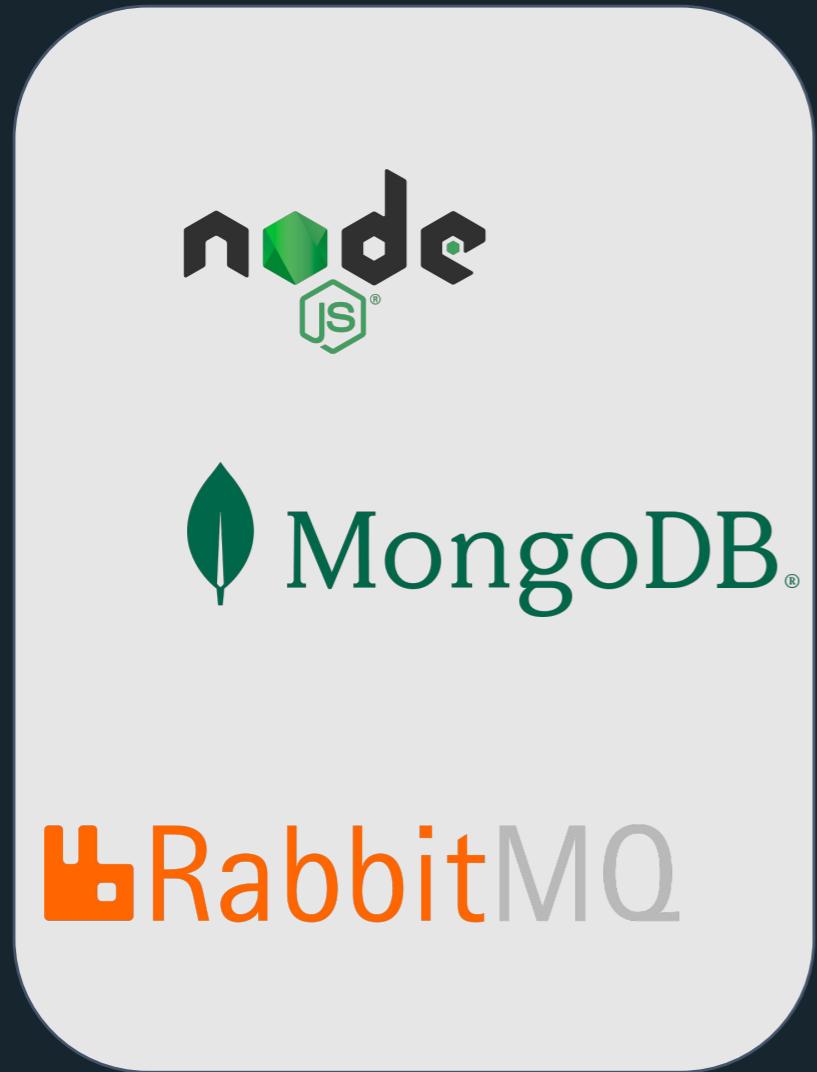
MSU
01 335
451

www.oconnor.co.uk

MAERSK

SEALAND









- Isolamento entre processos
- Controle sobre os processos
- Idempotência
- Portabilidade
- Confiabilidade de comportamento

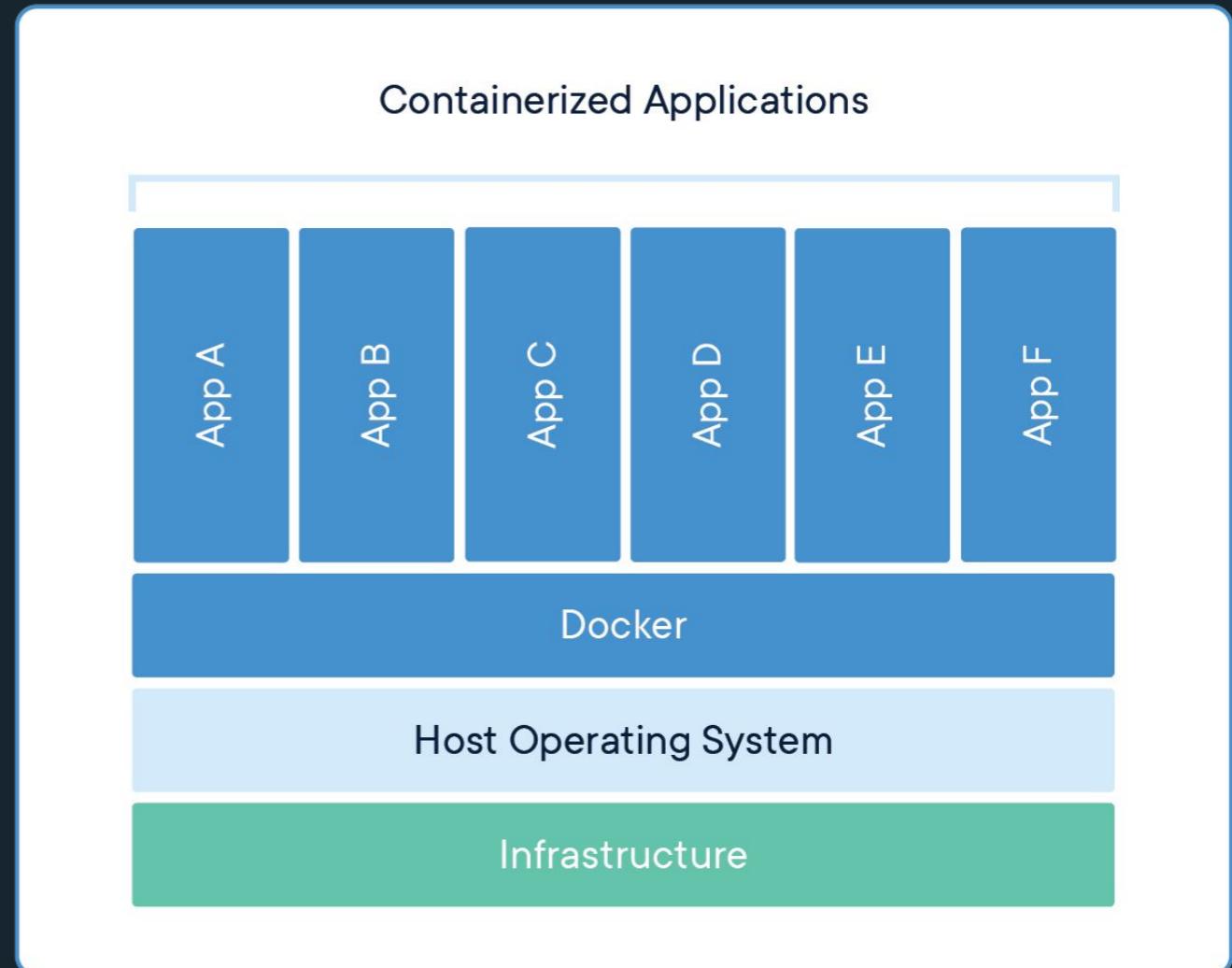
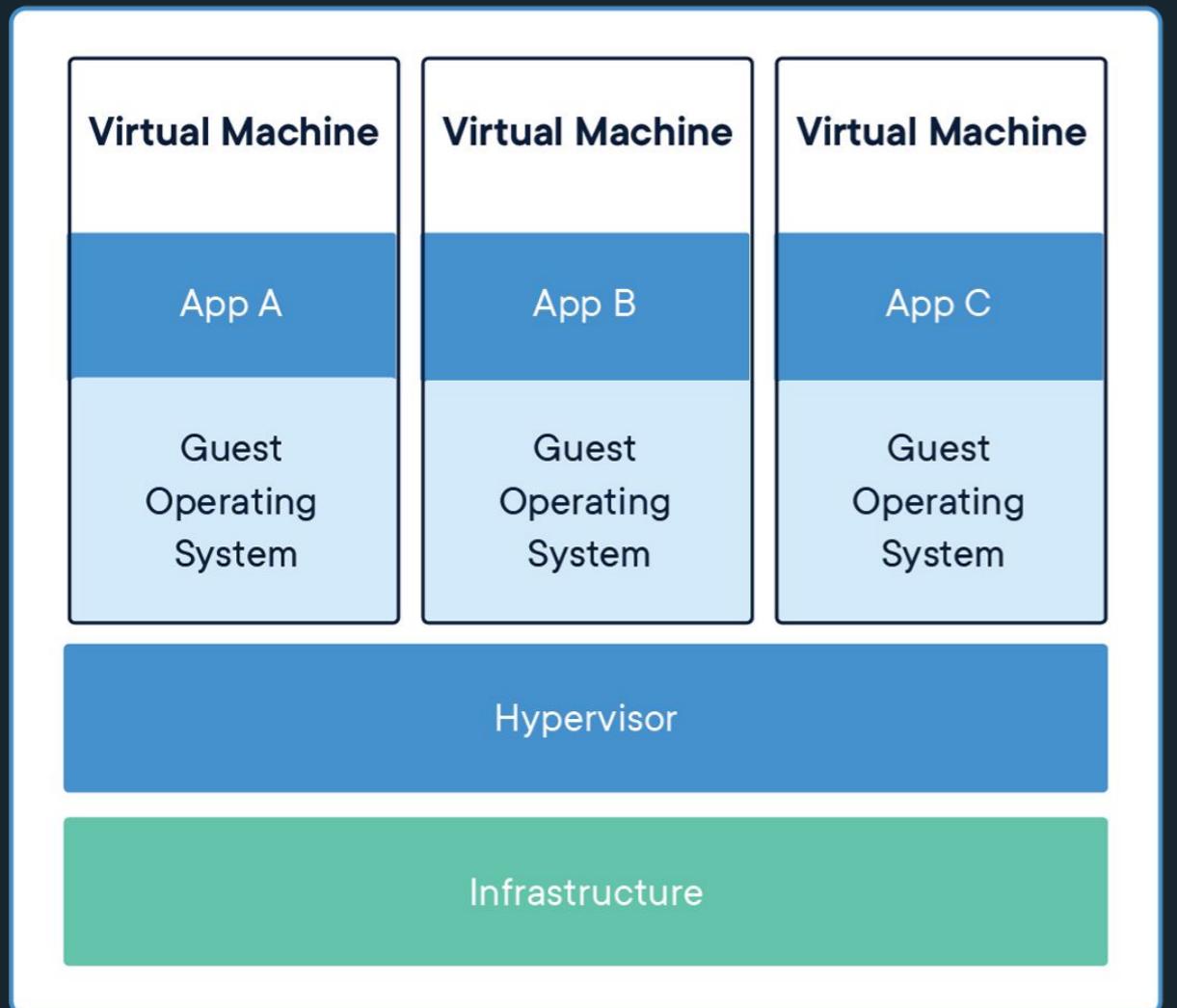
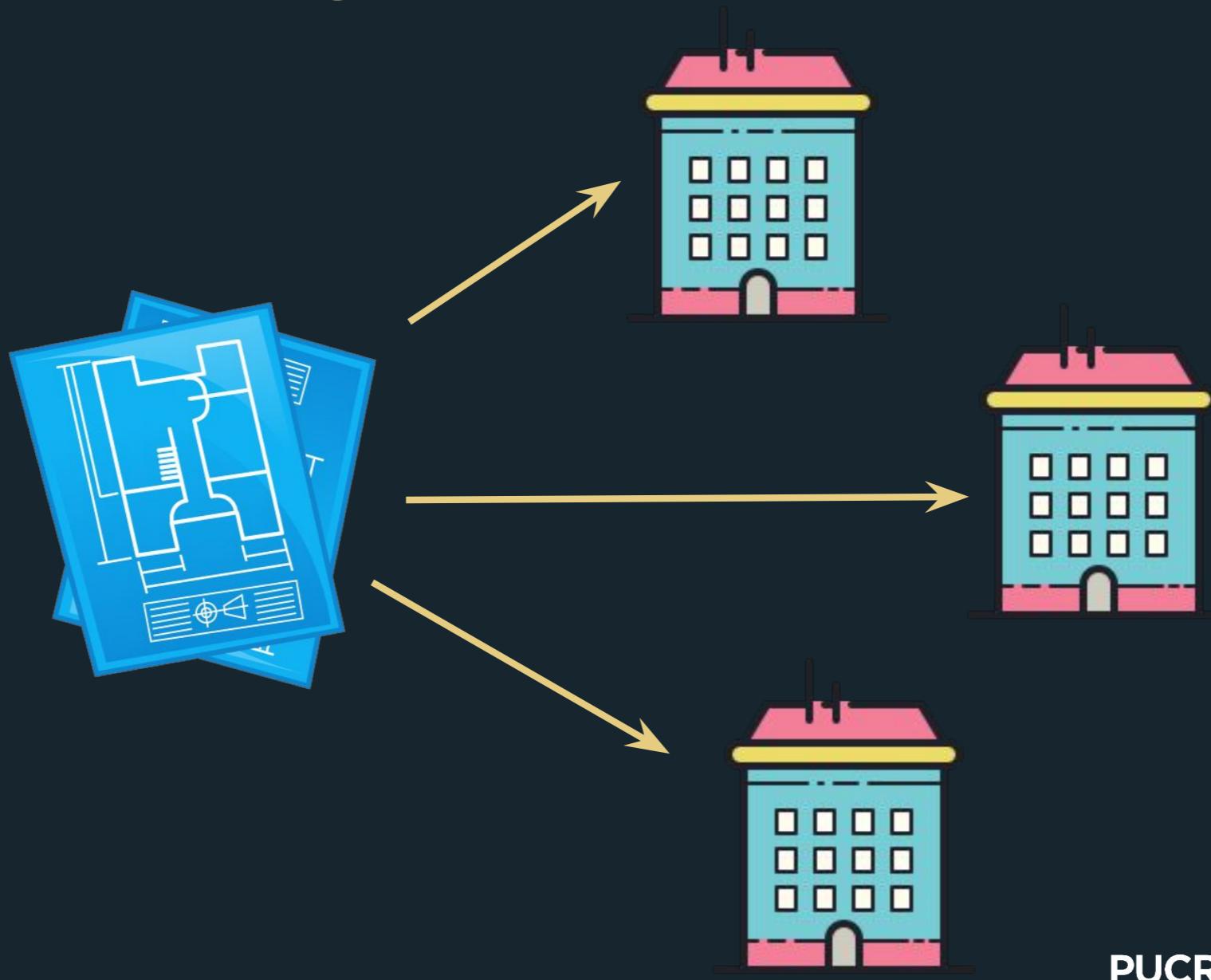
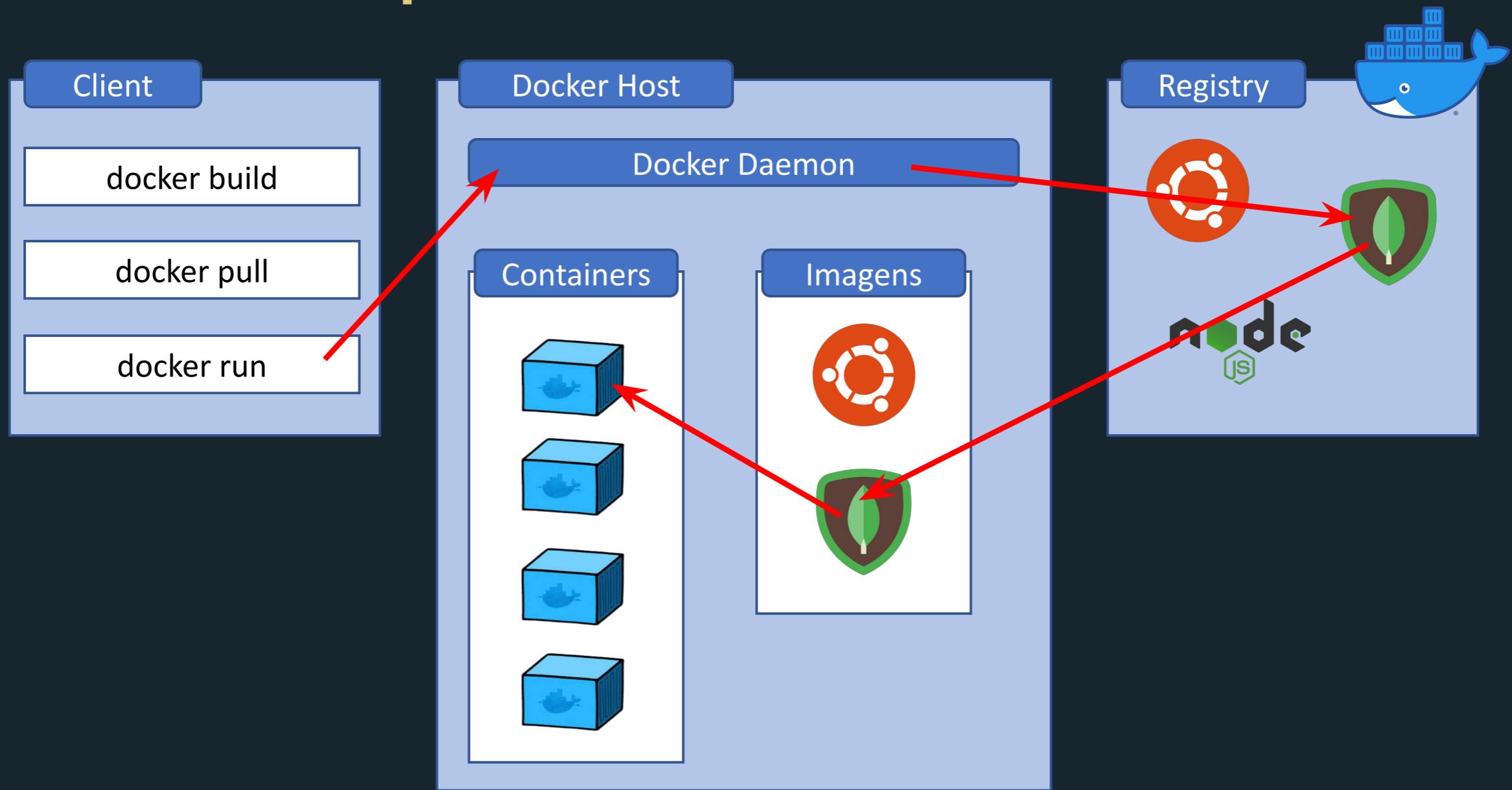




Imagen x Container



Arquitetura do Docker



Opções de uso no Dockerfile

- **FROM =>** Inicializa o build de uma imagem a partir de uma imagem base
- **RUN =>** Executa um comando
- **LABEL =>** Adiciona metadados a imagem
- **CMD =>** Define o comando e/ou os parâmetros padrão
- **EXPOSE =>** Define que o container precisa expor a porta em questão
- **ARG =>** Define um argumento pra ser usado no processo de construção
- **ENV =>** Define variáveis de ambiente
- **ADD =>** Copia arquivos ou diretórios ou arquivos remotos e adiciona ao sistema de arquivos da imagem
- **COPY =>** Copia arquivos ou diretórios e adiciona ao sistema de arquivos da imagem
- **ENTRYPOINT =>** Ajuda você a configurar um contêiner que pode ser executado como um executável
- **VOLUME =>** Define volumes que devem ser definidos
- **WORKDIR =>** Define o seu diretório corrente







Boas práticas



- Imagens oficiais
- Dockerignore
- Não usar tag latest

PUCRS online  uol edtech