INTEGRAÇÃO CONTÍNUA

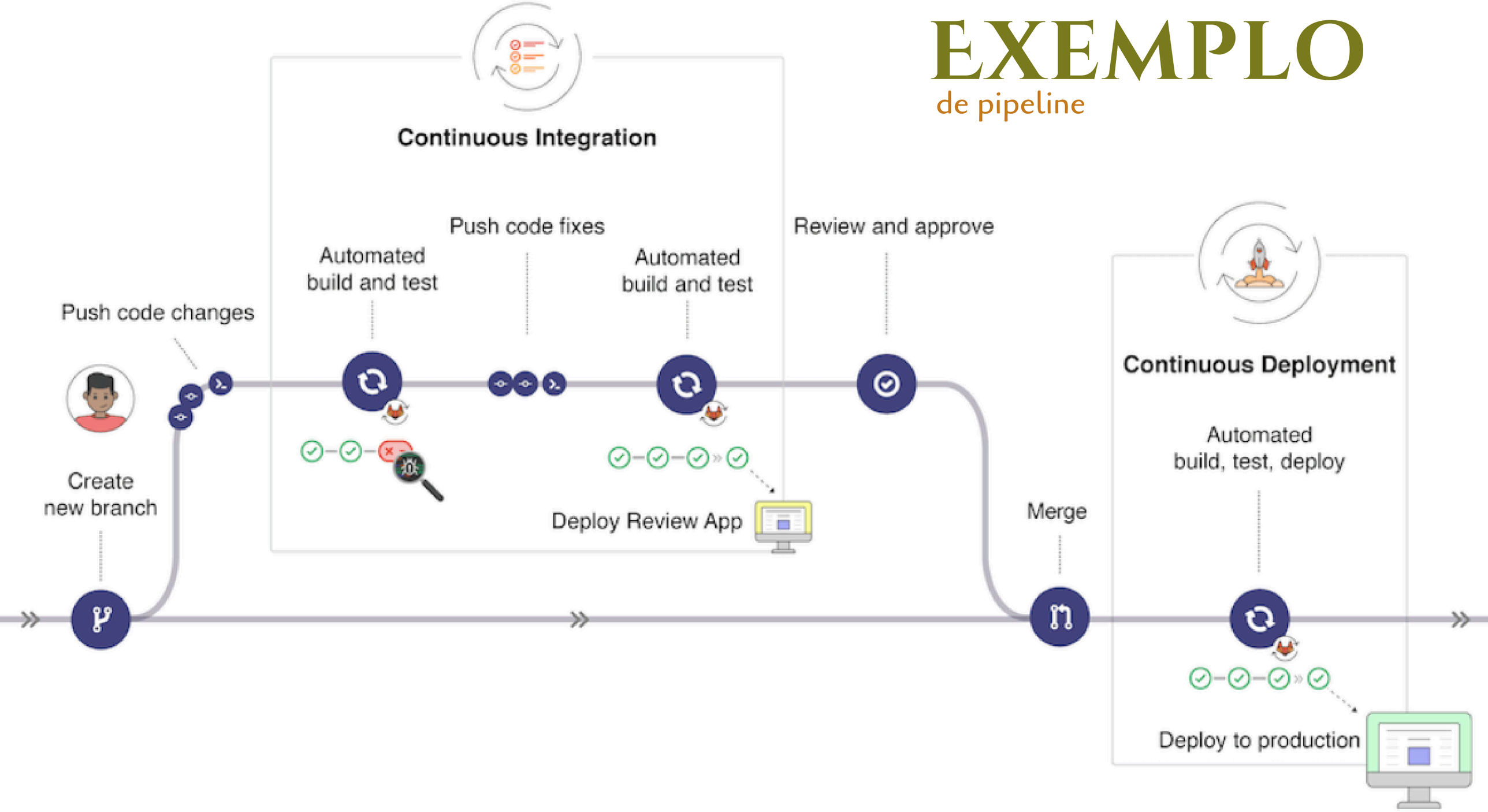
Um dos segredos para muitos times trabalharem em um mesmo aplicativo e fazerem várias entregas por dia.

MERCADO


Web Dev
Mobile Dev
Game Dev
Data Engineer
Cloud Engineer
IoT Engineer

EXEMPLO

de pipeline



ETAPAS

Integração

Diz respeito ao conjunto de quase todas as outras etapas. Etapas bem feitas facilitam muito integrar modificações de código, que podem ser feitas por times distantes.

Análise Estática

Esta etapa diz respeito a analisar o código sem rodar ele. Pode englobar análise da formatação, do estilo, de vulnerabilidades e de diversas métricas de qualidade.

ETAPAS

Análise Dinâmica

Esta etapa diz respeito a analisar o código rodando. Pode englobar diversos tipos de testes, que podem buscar problemas na segurança, na performance e no funcionamento do código.

Revisão Humana

Pode incluir revisões de pull requests e pode incluir gatilhos manuais na pipeline.

ETAPAS

Empacotamento

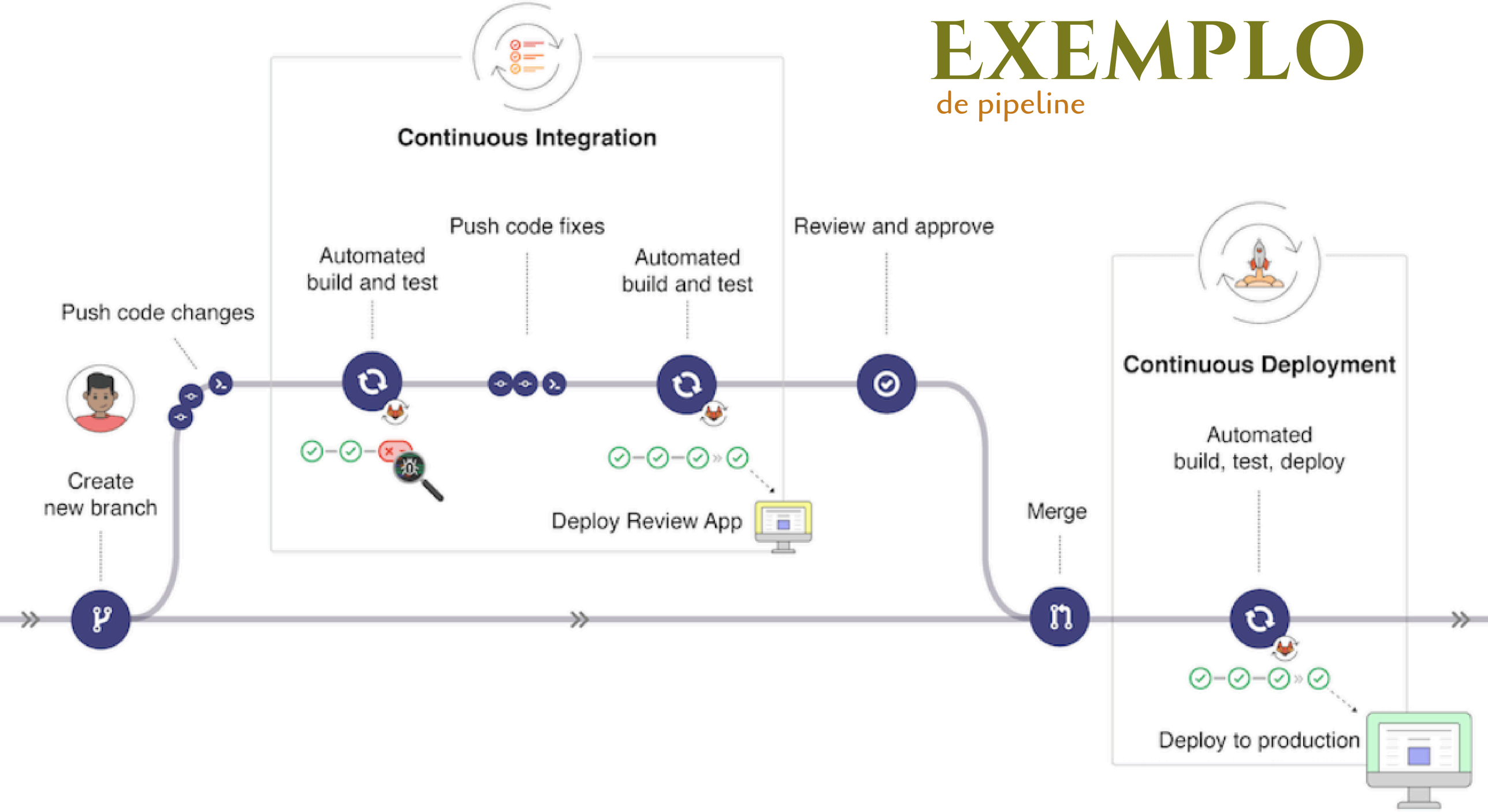
Cada framework dispoẽ de várias formas de fazer pacotes ou artefatos que podem ser utilizados para executar o aplicativo. Com pacotes ou não, também pode ser usada a virtualização.

Entrega

É entregar o software para seus usuários. Seja por meio de um celular, de um carro ou de outro dispositivo eletrônico.

EXEMPLO

de pipeline



ADICIONAL

Monitoramento

Monitorar é essencial para o sucesso de estratégias avançadas de entrega (exemplo do canário) e para responder perguntas dificeis (como “Será que a performance regrediu depois de uma atualização?”).

IaC

Infraestrutura como código ou infrastructure as code. Uma vez que a infraestrutura que vai rodar uma aplicação é definida como código, podemos tratar a entrega da infraestrutura do mesmo jeito que tratamos a entrega do código do app.

FERRAMENTAS

para construir pipelines

JÁ HOSPEDADAS

PRINCIPAIS:

- GitHub Actions;
- GitLab CI;
- Azure Pipelines.

VANTAGENS:

- Fácil configuração;
- Nível gratuito para pequena escala;
- Baixo custo da mão de obra.

PARA HOSPEDAR (SELF-HOSTED)

PRINCIPAIS:

- FluxCD;
- ArgoCD;
- Jenkins.

VANTAGENS:

- Alta flexibilidade;
- Informações seguras;
- Baixo custo do serviço.

Para mais ferramentas, acompanhe landscape.cncf.io

App Definition and Development

Continuous Integration & Delivery

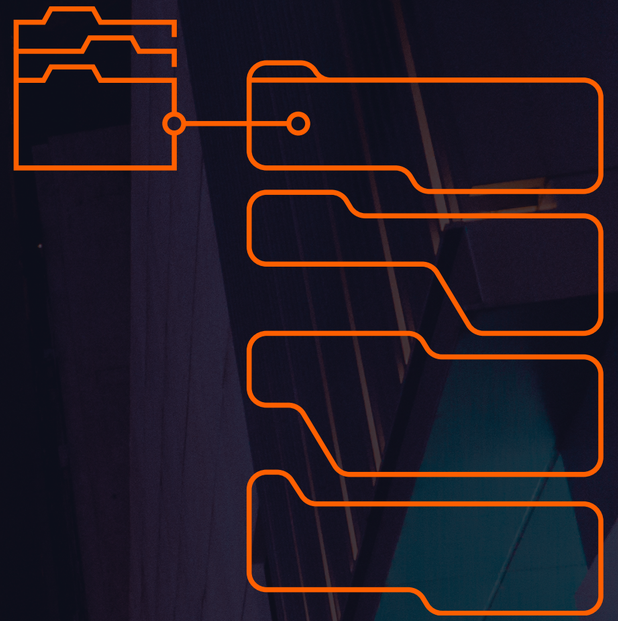
argo — CNCF GRADUATED
flux — CNCF GRADUATED
keptn — CNCF INCUBATING
OpenKruise — CNCF INCUBATING
agola
AKUITY
AppVeyor
AWS CodePipeline
Azure Pipelines
Bamboo
BRIGADE
Buildkite
bunnyshell
Bytebase
CARTOGRAPHER
circleci
Skycap
CloudBees.
codefresh
Concourse
D2 IQ Dispatch
devtron
flagger
GitHub Actions
GitLab
gitness
go
Google Cloud Build
harness
HELMWAVE
hyscale
Jenkins
JENKINS X
k6
Keploy
Liquibase
Mergify
Northflank
Octopus Deploy
OpenGitOps
OpsMx
ORTELIUS
ozone
PipeCD
Razee
Screwdriver.cd
semaphore
spacelift
Spinnaker
TeamCity
TEKTON
terramate
TESTKUBE
Travis CI
unleash
weaveworks
werf
XL DEPLOY
kube-burner

INTEGRAÇÃO CONTÍNUA

ATIVIDADE

PIPELINE COM GITHUB ACTIONS

BUILD -> LINT -> TEST -> PUSH



DETALHES

00: escolher aplicação

Opções: projeto de MDS ou aplicação disponibilizada para o exercício (github.com/leomichalski/exemplo). A ordem das etapas pode variar, desde que a última seja o deploy ou o push.

01: build

Compilar, empacotar ou gerar o artefato da aplicação.

02: lint

Fazer análise estática do código usando algum linter da linguagem de programação da aplicação. Caso a etapa de linting falhe, a pipeline deve parar.

03: test

Rodar testes da aplicação. Caso os testes falhem, a pipeline deve parar.

04: push

Fazer o deploy da aplicação ou armazenar o artefato gerado na etapa “build”.