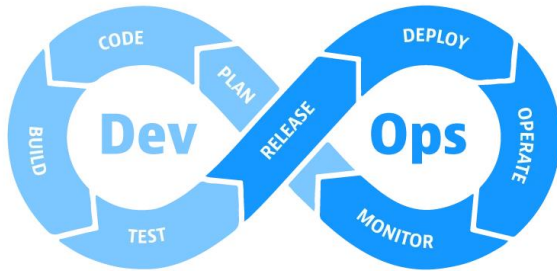


DevOps

Ferramenta Jenkins



DevOps

Um composto de Dev (desenvolvimento) e Ops (operações).

O DevOps é a união de pessoas, processos e tecnologias para fornecer continuamente valor aos clientes.

O que o DevOps significa para as equipes?

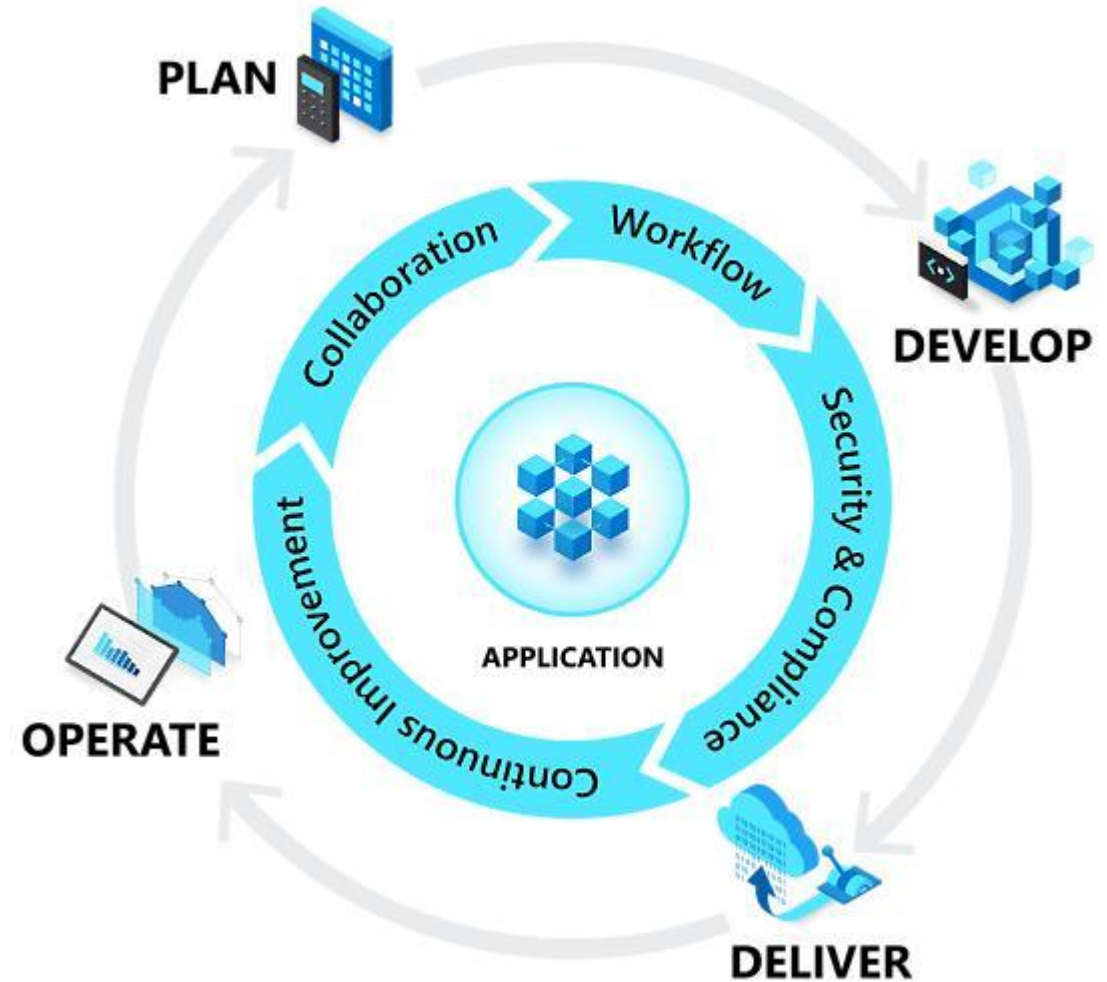
- O DevOps permite que funções anteriormente isoladas – desenvolvimento, operações de TI, engenharia da qualidade e segurança – atuem de forma coordenada e colaborativa para gerar produtos melhores e mais confiáveis.
- Ao adotar uma cultura de DevOps em conjunto com as práticas e ferramentas de DevOps, as equipes ganham a capacidade de responder melhor às necessidades dos clientes, aumentar a confiança nos aplicativos que constroem e cumprir as metas empresariais mais rapidamente.

Benefícios do DevOps

- Equipes que adotam a cultura, as práticas e as ferramentas de DevOps apresentam alto desempenho, criando produtos melhores, com mais rapidez, para maior satisfação do cliente.
- Esse aumento na colaboração e na produtividade também é essencial para cumprir metas empresariais como as seguintes:
 - Acelerar a colocação no mercado;
 - Adaptar ao mercado e à concorrência;
 - Manter a estabilidade e a confiabilidade do sistema;
 - Melhorar o tempo médio de recuperação

DevOps e o ciclo de vida do aplicativo

- O DevOps influencia o ciclo de vida do aplicativo em todas as fases do planejamento, do desenvolvimento, da entrega e da operação.
- Cada fase depende das demais e elas não são específicas da função.
- Em uma verdadeira cultura de DevOps, cada função está envolvida de alguma forma em cada fase.



Planejamento

- Na fase de planejamento, as equipes de DevOps idealizam, definem e descrevem os recursos e as funcionalidades dos aplicativos e sistemas que estão construindo.
- Elas acompanham o progresso em níveis altos e baixos de granularidade, desde tarefas de produto único até tarefas que abrangem portfólios de vários produtos.
- Criar listas de pendências, acompanhar bugs, gerenciar o desenvolvimento de software Agile com o Scrum, usar quadros Kanban e visualizar o progresso com painéis são algumas das maneiras pelas quais as equipes de DevOps planejam com agilidade e visibilidade.

Desenvolvimento

- A fase de desenvolvimento inclui todos os aspectos da codificação – gravação, teste, revisão e integração do código pelos membros da equipe – bem como a compilação do código em artefatos de compilação, que podem ser implementados em vários ambientes.
- As equipes de DevOps buscam inovar rapidamente sem sacrificar a qualidade, a estabilidade e a produtividade.
- Para fazer isso, elas usam ferramentas extremamente produtivas, automatizam etapas elementares e manuais e iteram em pequenos incrementos por meio de testes automatizados e integração contínua.

Entrega

- A entrega é o processo de implantação de aplicativos nos ambientes de produção de maneira consistente e confiável.
- A fase de entrega também inclui a implantação e a configuração da infraestrutura fundamental totalmente governada que compõe esses ambientes.

Entrega

- Na fase de entrega, as equipes definem um processo de gerenciamento de versão com estágios claros de aprovação manual.
- Elas também definem portões automatizados que movem os aplicativos entre os estágios, até que sejam disponibilizados aos clientes.
- A automação desses processos os torna escalonáveis, repetíveis e controlados.
- Dessa forma, as equipes que praticam o DevOps podem frequentemente atuar e entregar com facilidade, confiança e tranquilidade.

Operação

- A fase de operação envolve manter, monitorar e solucionar problemas de aplicativos em ambientes de produção.
- Ao adotar as práticas de DevOps, as equipes trabalham para garantir a confiabilidade do sistema, a alta disponibilidade e o objetivo de tempo de inatividade igual a zero, reforçando a segurança e a governança.
- As equipes de DevOps buscam identificar os problemas antes que eles afetem a experiência do cliente e mitigar os problemas rapidamente quando ocorrem.
- Manter esse nível de vigilância requer telemetria avançada, alertas acionáveis e visibilidade total sobre os aplicativos e o sistema subjacente.

Práticas do DevOps

- Além de estabelecer uma cultura de DevOps, as equipes dão vida ao DevOps implementando certas práticas ao longo do ciclo de vida do aplicativo.
- Algumas dessas práticas ajudam a acelerar, automatizar e melhorar uma fase específica.
- Outras abrangem várias fases, ajudando as equipes a criar processos contínuos que melhoram a produtividade.

CI/CD (Integração Contínua e Entrega Contínua)

- O gerenciamento de configuração refere-se ao gerenciamento do estado dos recursos em um sistema, incluindo servidores, máquinas virtuais e bancos de dados.
- Usando ferramentas de gerenciamento de configuração, as equipes podem implementar alterações de maneira controlada e sistemática, reduzindo os riscos de modificar a configuração do sistema.
- As equipes usam ferramentas de gerenciamento de configuração para acompanhar o estado do sistema e ajudar a evitar desvios de configuração, já que é assim que a configuração de um recurso de sistema se desvia do estado desejado ao longo do tempo.

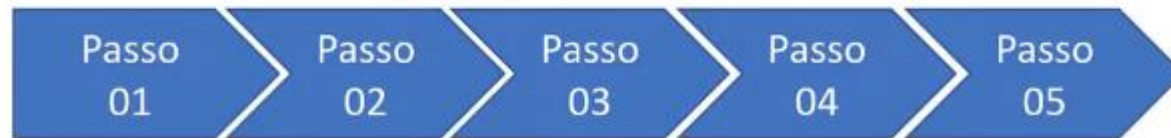
CI/CD (Integração Contínua e Entrega Contínua)

- Praticadas em conjunto com a infraestrutura como código, a definição e a configuração do sistema são fáceis de padronizar e automatizar, o que ajuda as equipes a operar ambientes complexos em escala.

O que é um pipeline em TI?

- Conjunto de tarefas para automatizar os fluxos de automação entre os profissionais e equipes, por exemplo desenvolvimento e infraestrutura;

Pipeline



Integração contínua(CI)/Entrega Contínua(CD)

Integração Contínua



Deploy Contínuo



Jenkins

- Ferramenta de automação de diversos tipos de tarefas;
- É uma das ferramentas mais utilizadas para executar os processos de CI/CD;
- Criado em 2005 por um desenvolvedor da Sun (Kohsuke Kawaguchi);
- Possui plugins com os mais diversos tipos de sistemas para a execução de tarefas;
- Pode ser utilizado em qualquer ambiente cloud;

Jenkins

- Desenvolvido em Java (WAR)
- Possui uma interface de gerenciamento web e disponibiliza APIs;
- Permite gerenciamento de usuários e permissões;
- Permite escalar outros servidores de Jenkins de forma fácil;
- Suporte a pipelines e arquivo declarativo (jenkinsfile);

Ferramenta de Integração Contínua (CI/CD)

- Permite identificar bugs de alteração de código que entra no pipeline de produção;
- É possível identificar bugs quando é feito um merge de códigos desenvolvidos por diversos usuários;
- Acompanhamento de métricas das etapas do pipeline até o código entrar em produção;
- Permite fazer o checkout da aplicação em um repositório de produção imediatamente logo após um commit;

Ferramenta de Integração Contínua (CI/CD)

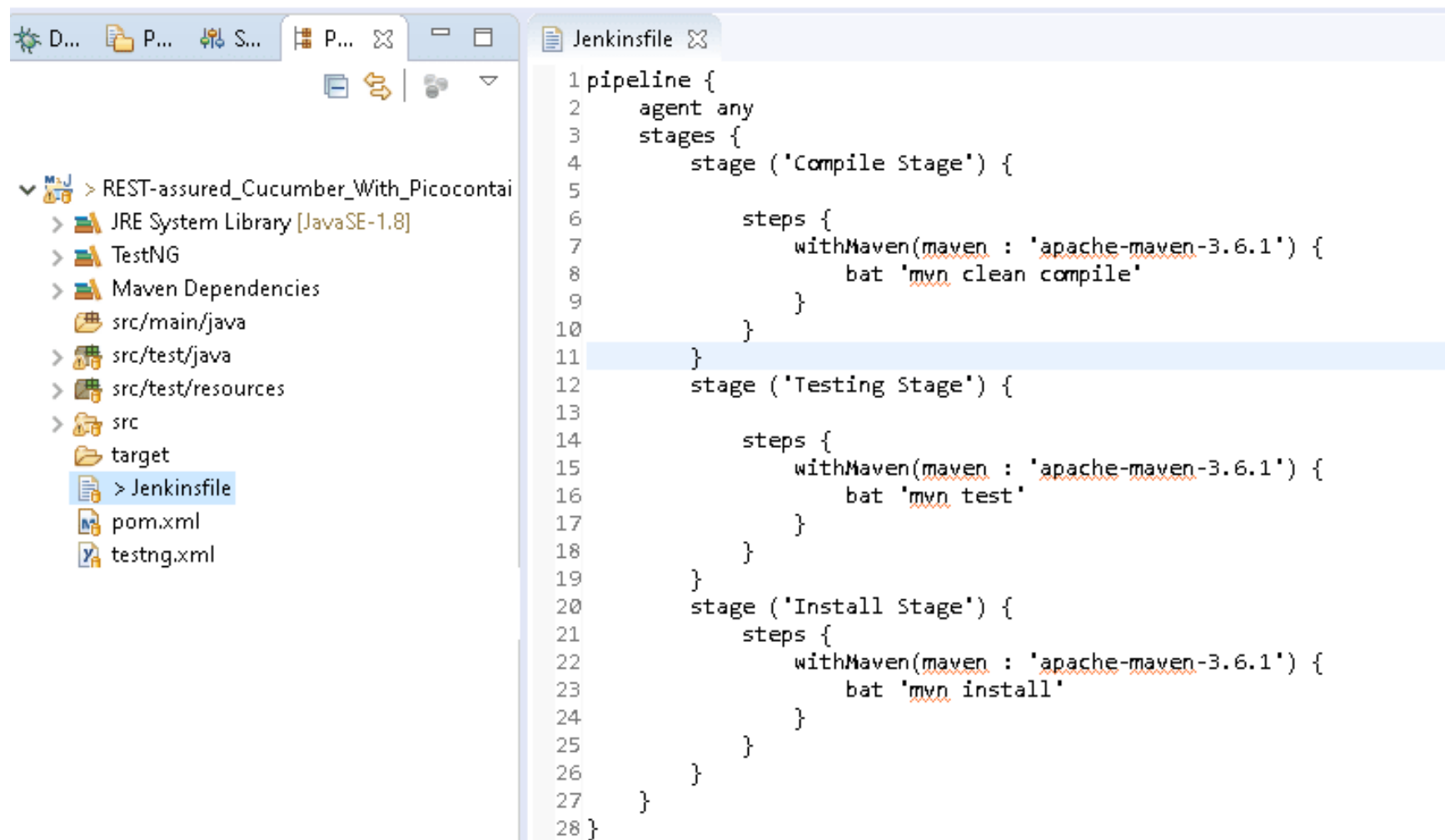
- Executa os processos de integração contínua (build, test, performance, tarefas necessárias para a verificação do bom funcionamento do software);
- Gera um relatório apresentando o status dos passos configurados executados;
- Realiza o processo de deploy de forma automatizada;
- Permite o processamento estilo CRON, para a programação de tarefas de forma automatizada;

Pipeline CI/CD

- Trilha ou passo a passo com cada etapa de execução de uma tarefa;
- Criada facilmente através de um Jenkinsfile;
- Apresentada de forma visual



Pipeline CI/CD



The screenshot shows an IDE with two main panels. The left panel displays a project structure for 'REST-assured_Cucumber_With_Picocontai'. The right panel shows the content of the 'Jenkinsfile'.

Project Structure (Left Panel):

- REST-assured_Cucumber_With_Picocontai
 - JRE System Library [JavaSE-1.8]
 - TestNG
 - Maven Dependencies
 - src/main/java
 - src/test/java
 - src/test/resources
 - src
 - target
 - Jenkinsfile
 - pom.xml
 - testng.xml

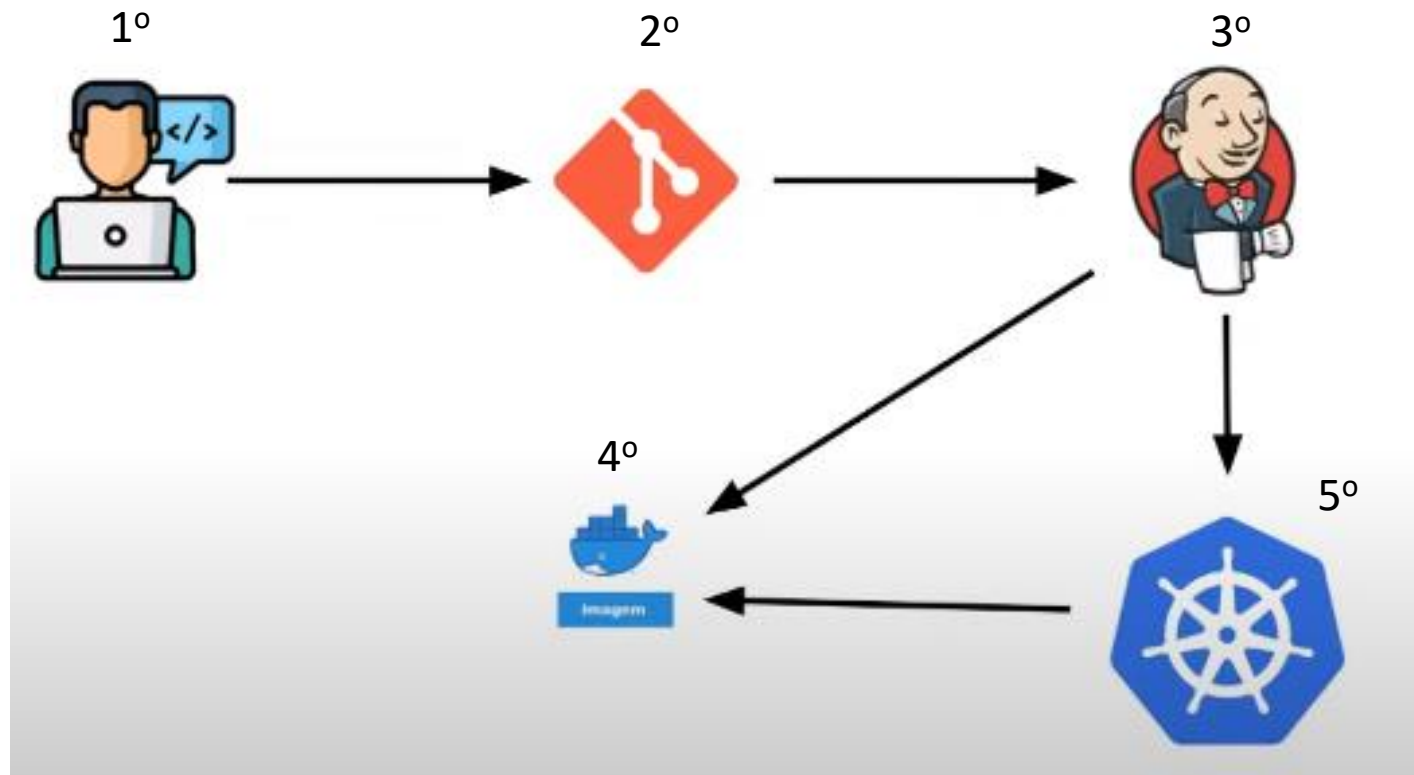
Jenkinsfile (Right Panel):

```
1 pipeline {
2   agent any
3   stages {
4     stage ('Compile Stage') {
5
6       steps {
7         withMaven(maven : 'apache-maven-3.6.1') {
8           bat 'mvn clean compile'
9         }
10      }
11    }
12    stage ('Testing Stage') {
13
14      steps {
15        withMaven(maven : 'apache-maven-3.6.1') {
16          bat 'mvn test'
17        }
18      }
19    }
20    stage ('Install Stage') {
21      steps {
22        withMaven(maven : 'apache-maven-3.6.1') {
23          bat 'mvn install'
24        }
25      }
26    }
27  }
28 }
```

Jenkinsfile

```
node {  
  stage('Git checkout'){  
    git Branch: 'master', url: 'https://github.com/adssenacgit/aprendizagemFrontendAngular'  
  }  
  stage('Instalação'){  
    npm i  
  }  
  stage('Startup server'){  
    npm start  
  }  
}
```

Passos para implementação de um sistema automatizado



Glossário:

- **CRON:** é um comando do Linux que permite agendar tarefas a serem realizadas futuramente, podendo ser horas depois ou em outro dia. Com essa configuração, o servidor vai executar qualquer processo no dia e hora informados em um site ou sistema, respeitando a devida fila de trabalhos que foi projetada.