Análise de falhas em pipelines

Pontifícia Universidade Católica de Minas Gerais Engenharia de Software

Carlos Roberto - Cláudio Jansen - Nathan - Rafael Ferreira - Vitor Xavier

Goals

- **Objeto:** Pipelines
- **Finalidade:** Identificar etapas com falhas, impacto de tempo e recurso para correção de uma falha em pipeline
- **Foco:** Medição e experimentação de uma pipeline
- **Viewpoint:** Desenvolvedores DevOps
- **Ambiente:** Aplicação que tem o fim automatizar testes em aplicações diversas

O que é uma pipeline?

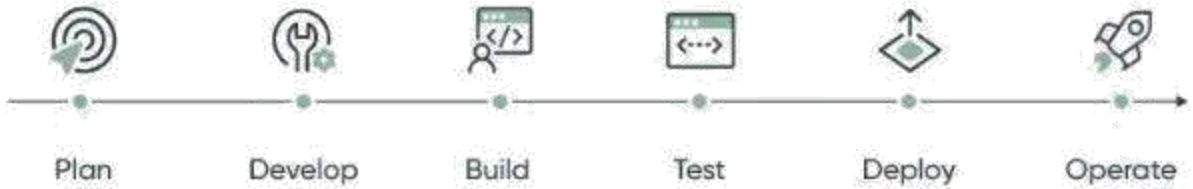
Introdução: As pipelines são configurações automatizadas usadas para compilar, testar e implantar códigos em diferentes ambientes.

Objetivo: Tem o objetivo de automatizar o processo de desenvolvimento de software, garantindo que o código seja testado, construído e implantado, facilitando detecções e correções de bugs rapidamente.

Funcionamento: Os desenvolvedores fazem push do código para o repositório do GitHub, que inicia o processo da pipeline de forma automática, executando Jobs que podem incluir a compilação do código, execução de testes e implantação dos serviços nos ambientes.

Vantagens: As vantagens de utilizar essa ferramenta inclui a rápida identificação de bugs, redução do tempo de entrega das features, melhor qualidade do produto, garantindo uma maior segurança no desenvolvimento, dentre outas.

CI/CD pipeline



Questions e Métricas

Q1: Quais são as etapas mais frequentemente falhadas na pipeline?

R: O sucesso de um pipeline de desenvolvimento de software pode ser afetado por várias etapas, e falhas podem ocorrer em diferentes pontos do processo.

M: Número de pipelines com falha em builds por projeto.

Número de pipelines com falha em testes por projeto.

Número de pipelines com falha em deploys por projeto.

Q2: Quanto tempo é perdido devido a falhas n pipeline?

R: O tempo perdido devido a falhas no pipeline de desenvolvimento de software varia de acordo com a natureza e gravidade das falhas, e depende da eficiência da equipe em identificar, diagnosticar e corrigir os problemas.

M: Tempo total gasto no intervalo em que uma pipeline falhou

até o momento em que subiu uma correção.

Questions e Métricas

Q3: Quais são os recursos mais comuns para correção de uma falha?

R: As falhas no desenvolvimento de software podem ter impactos significativos em uma ampla gama de recursos, incluindo a quantidade de alterações necessárias para corrigir a pipeline… Portanto, é fundamental adotar boas práticas de desenvolvimento, testes rigorosos e processos de garantia de qualidade para minimizar esses impactos.

M: Número de linhas adicionadas no commit em que a pipeline foi corrigida.

Número de linhas removidas no commit em que a pipeline foi corrigida.

Número de linhas comentadas no commit em que a pipeline foi corrigida.

Q4: Quão frequentemente as falhas no pipeline ocorrem e como elas afetam a continuidade do processo?

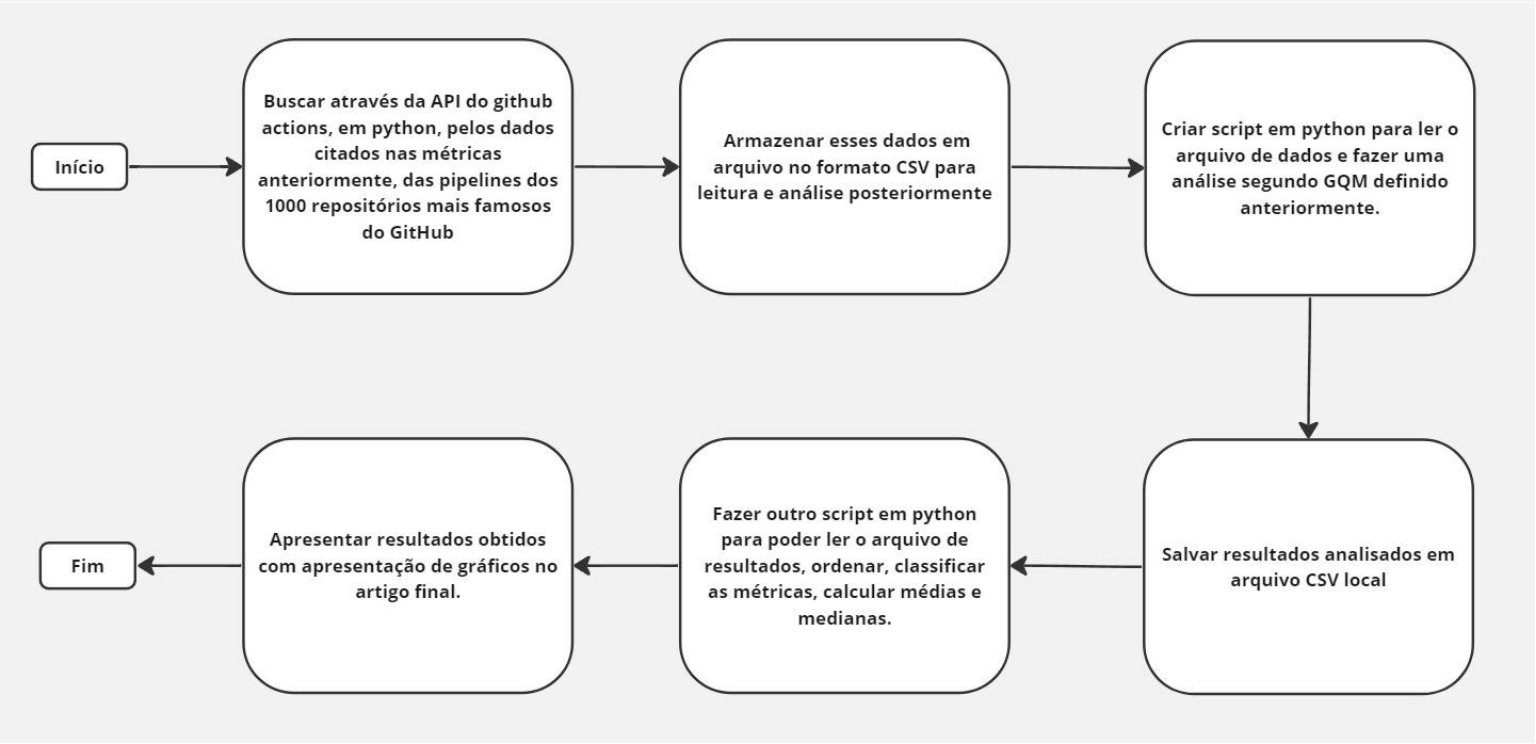
R: Para manter a continuidade do processo de desenvolvimento de software e minimizar o impacto das falhas, é essencial investir em práticas sólidas de engenharia de software, automação de testes, revisões de código, monitoramento contínuo e aprendizado com falhas para identificar e corrigir problemas de forma eficiente e eficaz.

M: Número total de falhas na pipeline por semana/mês.

Número médio de falhas de pipelines.

Percentual de pipeline que são interrompidas devido a falhas.

Planejamento do experimento que será realizado



Scripts, Dados e Resultados

Os resultados obtidos, os dados utilizados para análise da pesquisa, e os scripts necessários para sua coleta se encontram nesse repositório:

https://github.com/ClaudioJansen/GitHub-Script/tree/main/Tis%2006

Referências Bibliográfica