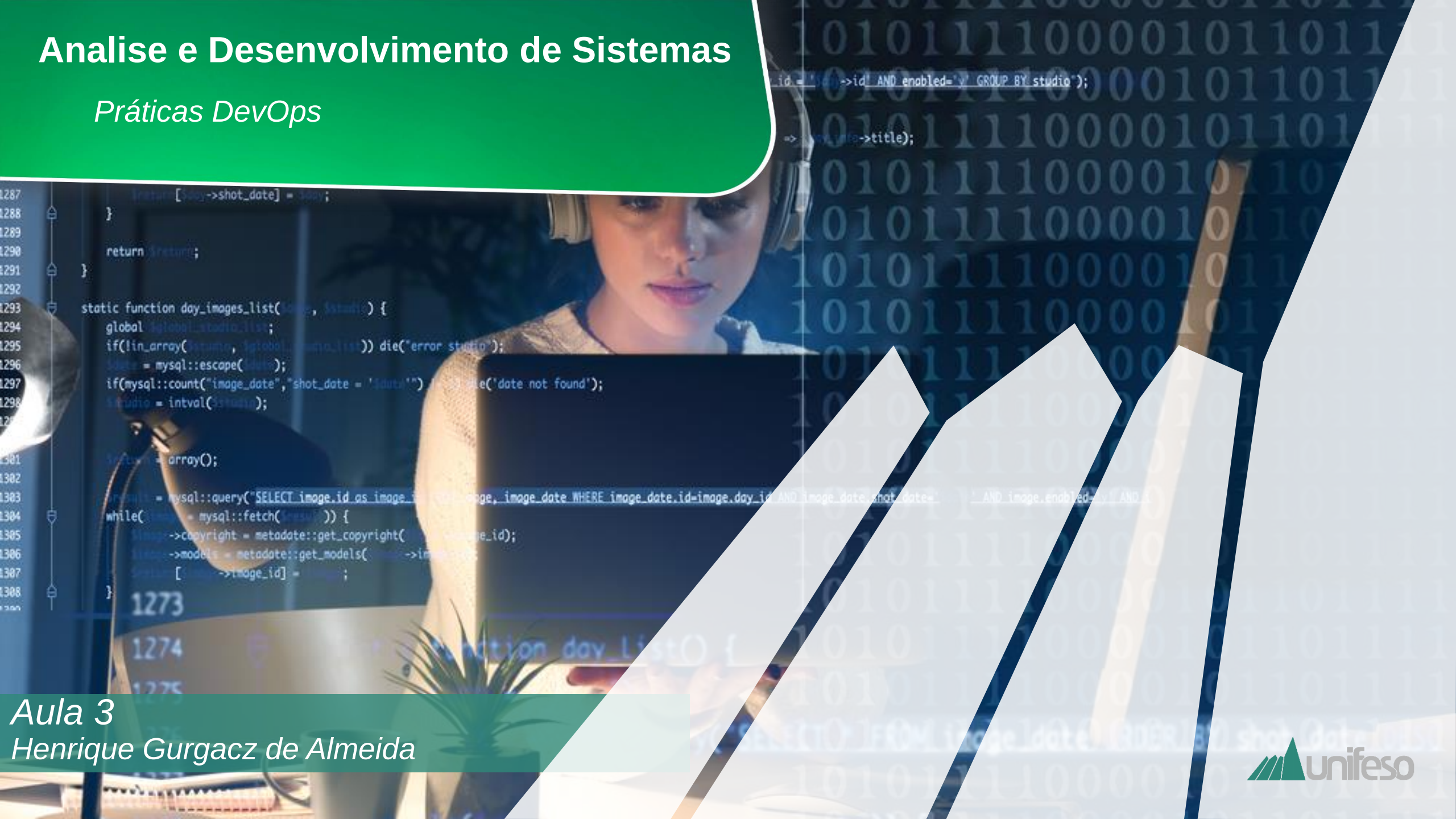
Analise e Desenvolvimento de Sistemas

Práticas DevOps



Aula 3

Henrique Gurgacz de Almeida

Explorando a Ferramenta GitHub

Módulo 3

- Versionamento de Código
- Pipelines no GitHub

Versionamento de Código

Unidade 3.1

O GitHub

- Criado em 2008
- Microsoft adquiriu o GitHub por US$ 7,5milhões
- Possui 36 milhões de usuários
- Possui 100 milhões de projetos
- Possui licença gratuita
- Possui Códigos abertos (públicos) e códigos fechados (privados)

O GitHub

- Armazena, Gere e Versiona inclusive arquivos comuns
- Colaboração e Comunicação
- Documentação e Visibilidade (Wikis, Views, Markdown, etc)
- Páginas de empresas e desenvolvedores (Redes Sociais)
- Integração Com IA Generativa para criar código (CoPilot)
- Actions -> Para CI e CD
- Segurança: Code Scanning

O GitHub

- Projetos famosos hospedados no GitHub:
  - https://github.com/WordPress
  - https://github.com/mysql
  - https://github.com/grafana

- Pessoas Famosas:
  - https://github.com/addyosmani
  - https://github.com/tonsky
  - https://github.com/DanWahlin

O GitHub

- Documentação Oficial GitHub

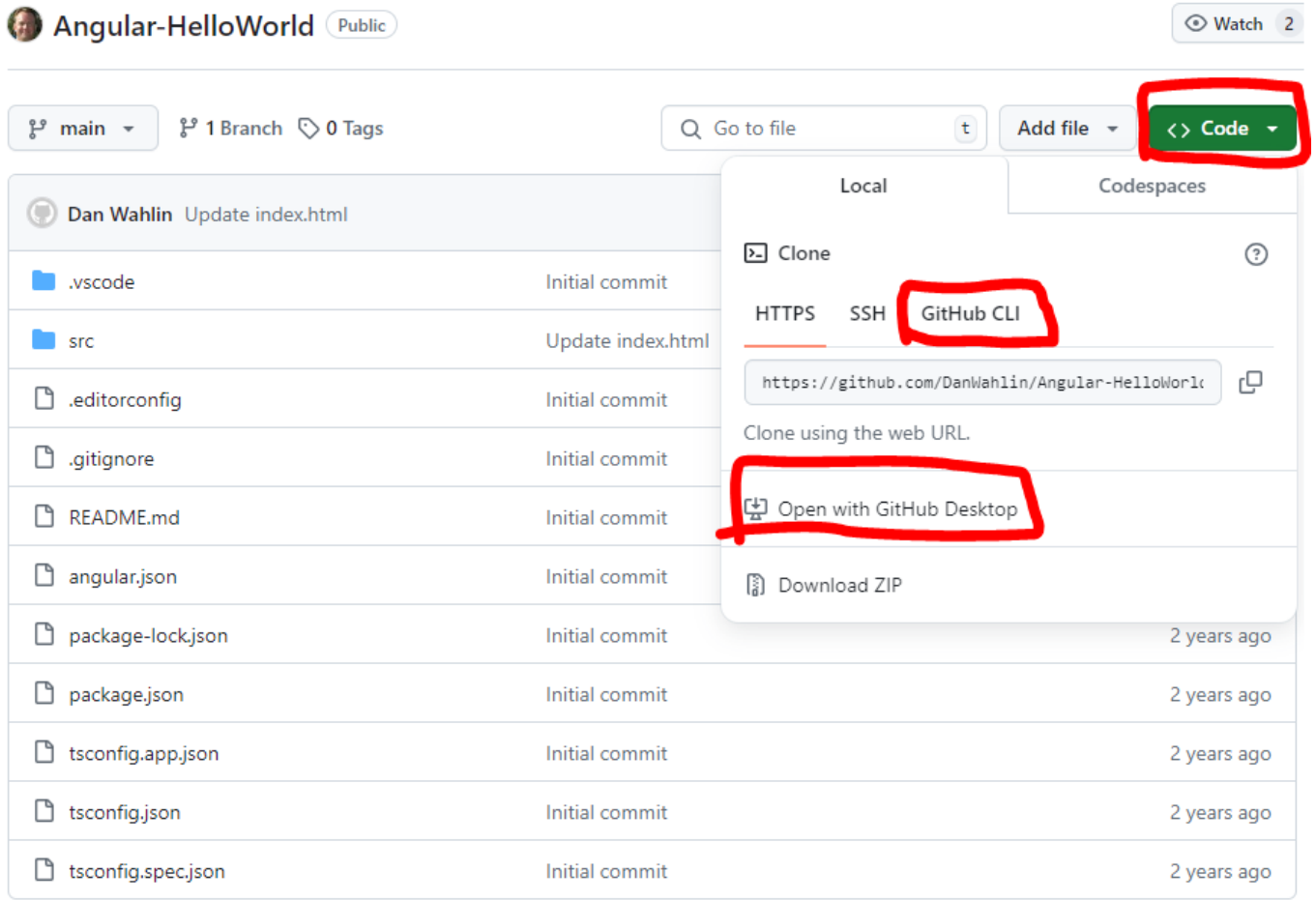
https://docs.github.com/pt

O GitHub – Cli e Desktop

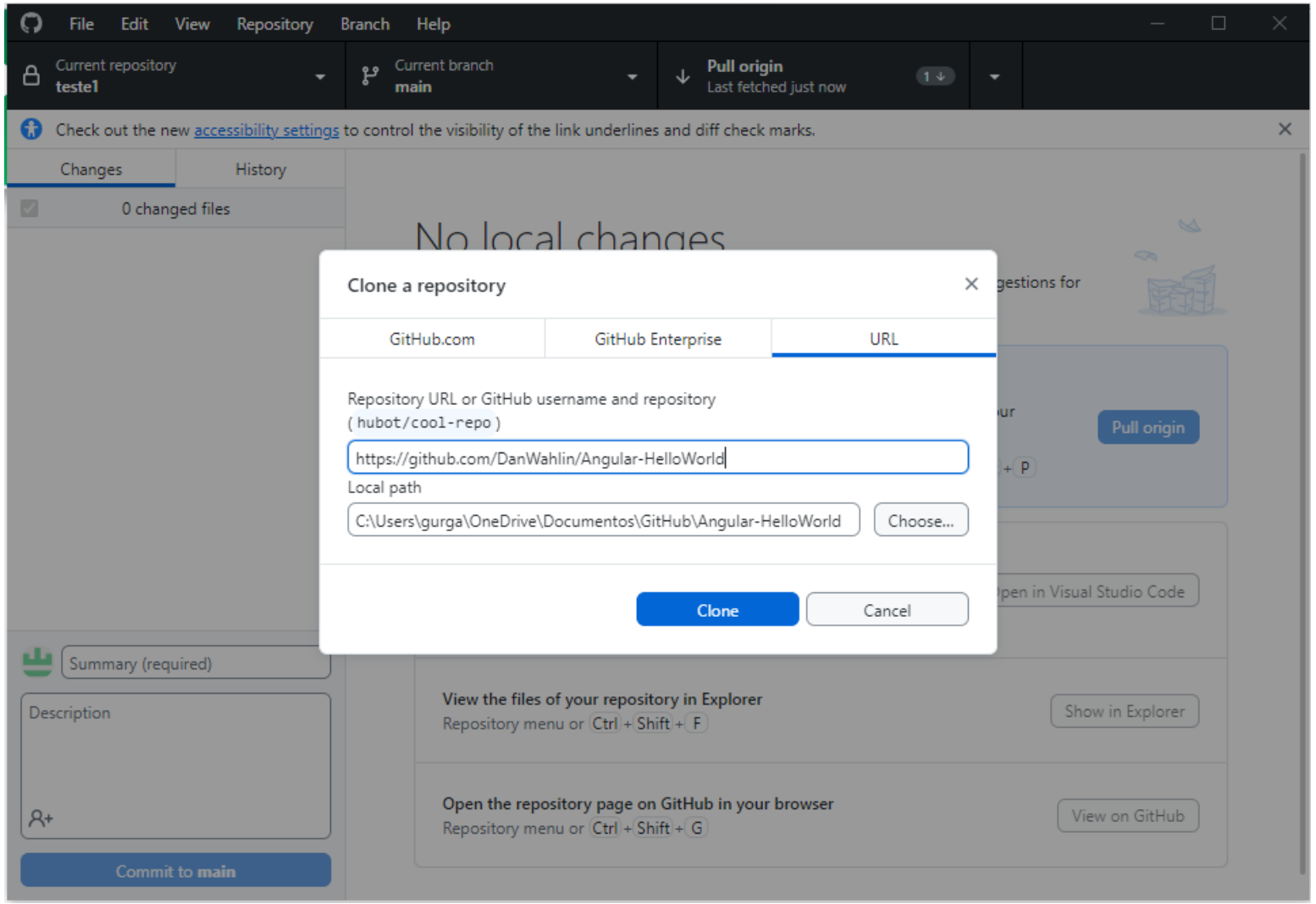
https://docs.github.com/pt/github-cli

https://docs.github.com/pt/desktop

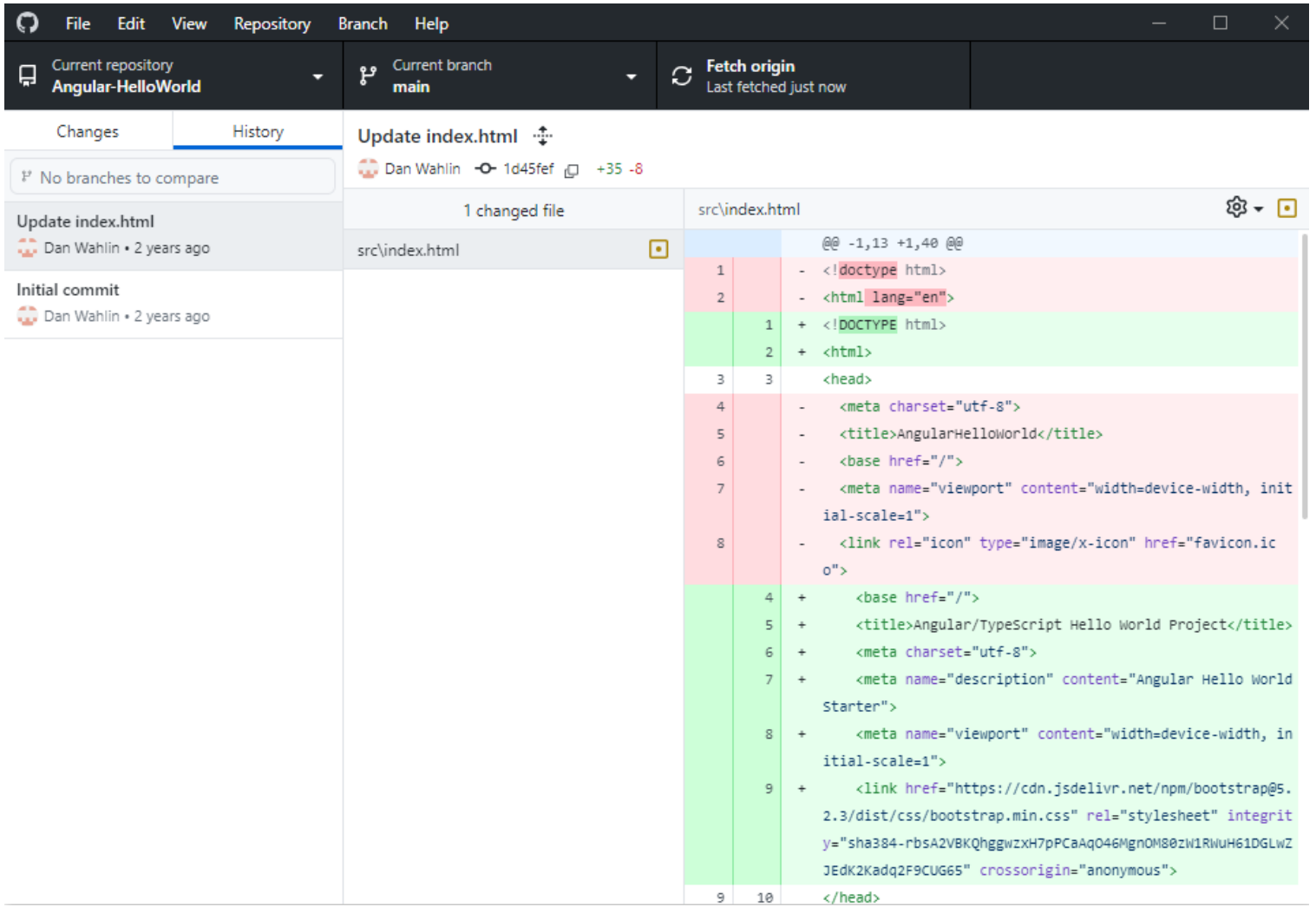
O GitHub – Clonar Repositório Público



O GitHub – Clonar Repositório Público



O GitHub – Clonar Repositório Público

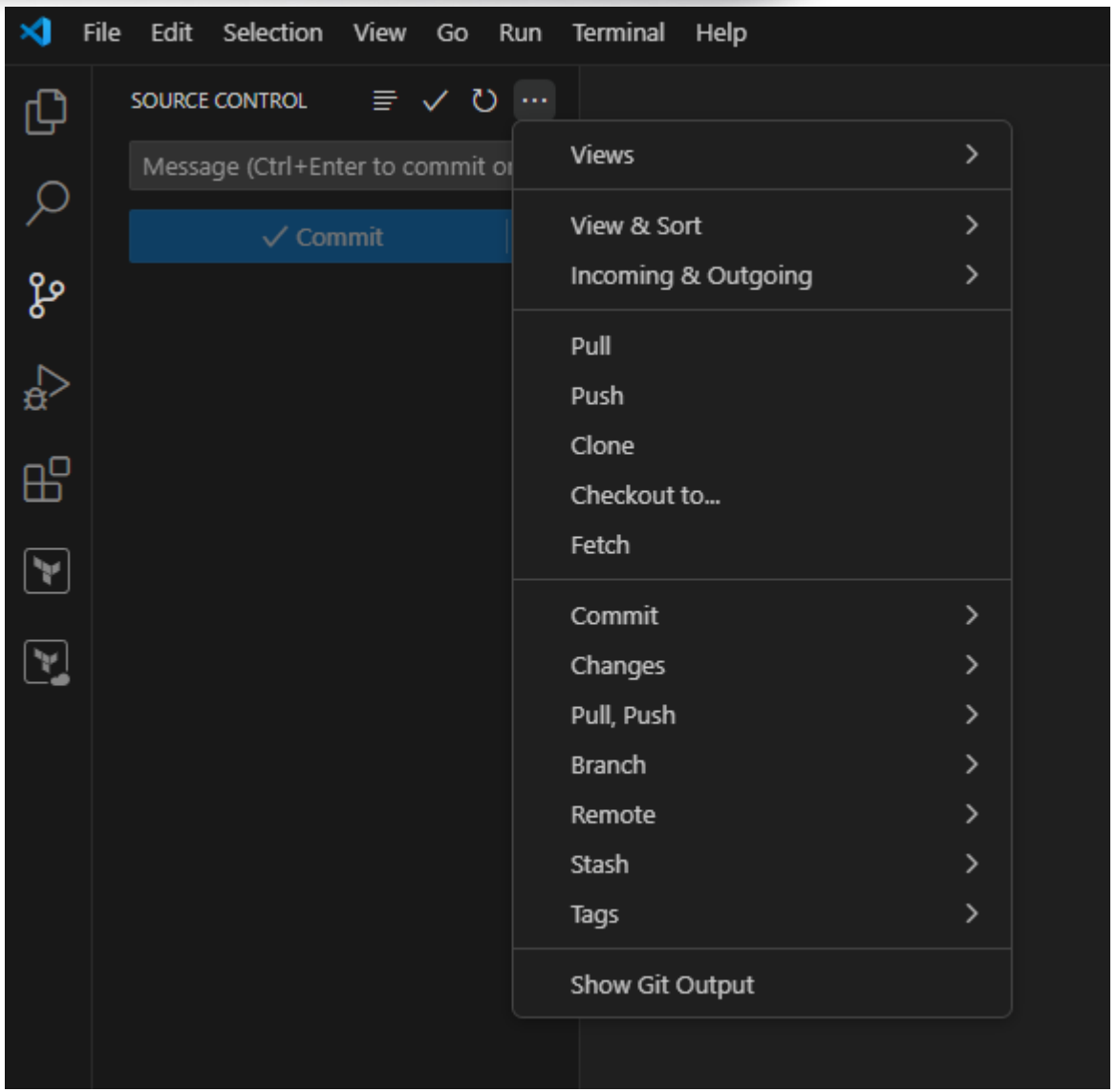


O GitHub – Passo a Passo

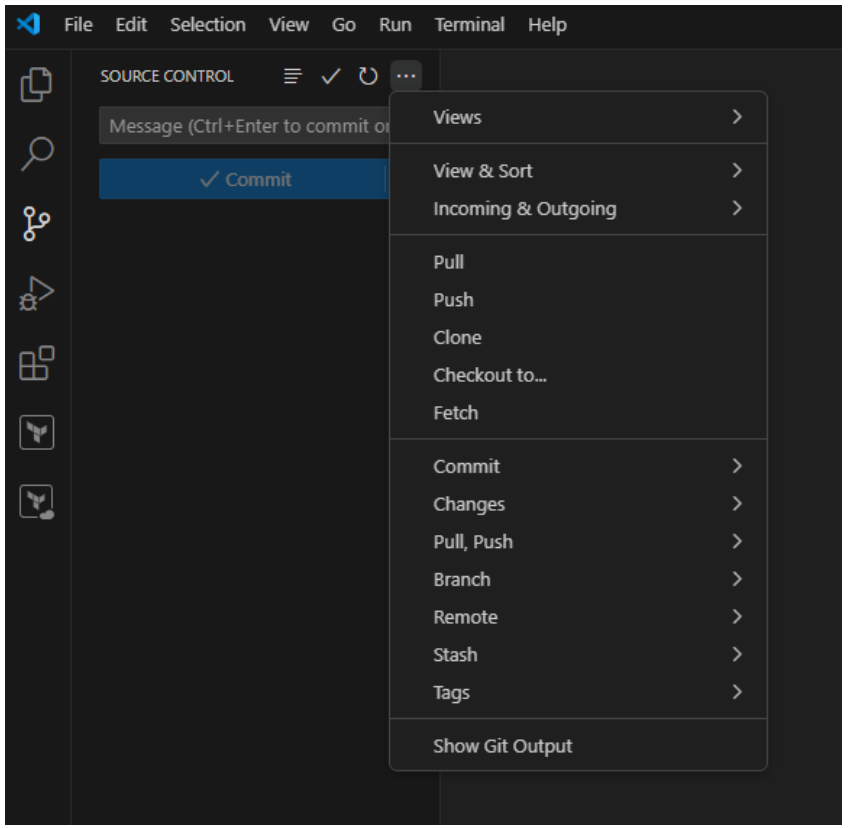
- Criar uma Conta
- Criar um Repositório
- Readme e a Linguagem Markdown

https://www.markdownguide.org/basic-syntax/

O GitHub – No VsCode



O GitHub – No VsCode



- **Clone:** Copia o repositório completo para o computador local
- **Commit:** Salva as alterações até o momento no repositório local. Permite voltar a esse ponto (antes do salvamento) caso necessário
- **Push:** Envia as alterações para a nuvem
- **Pull:** Baixa as atualizações de um repositório

O GitHub – No VsCode

- **Executando Commit, Push e Pull no GitHub Usando VSCode**

O GitHub – No VsCode

- **Resolvendo Conflitos no GitHub com VsCode**

O GitHub – Branches

- **Criar um Branch**
- **Usar um Branch**

Pipelines no GitHub

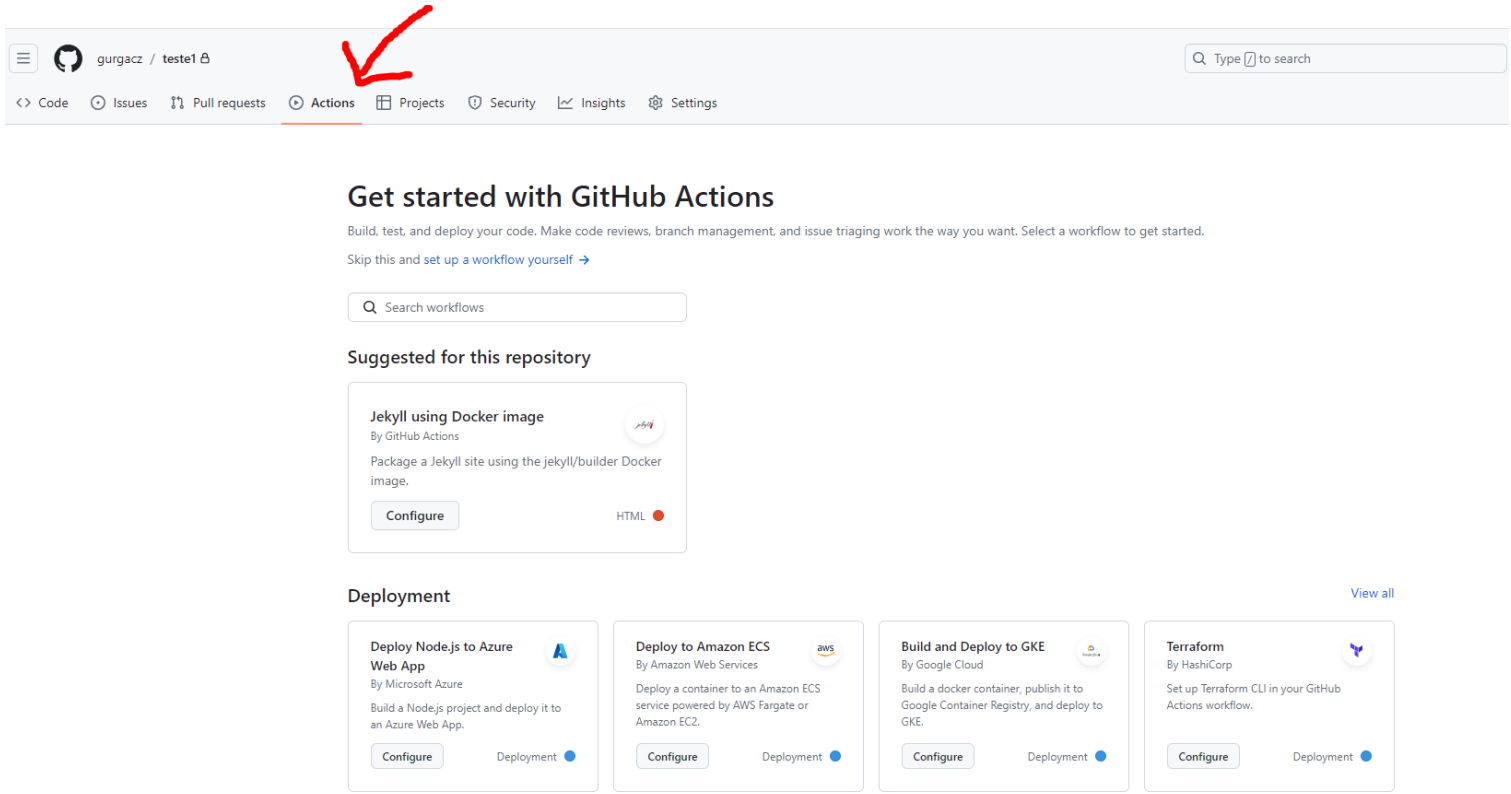
Unidade 3.2

Exemplos de Pipelines

GitHub Actions

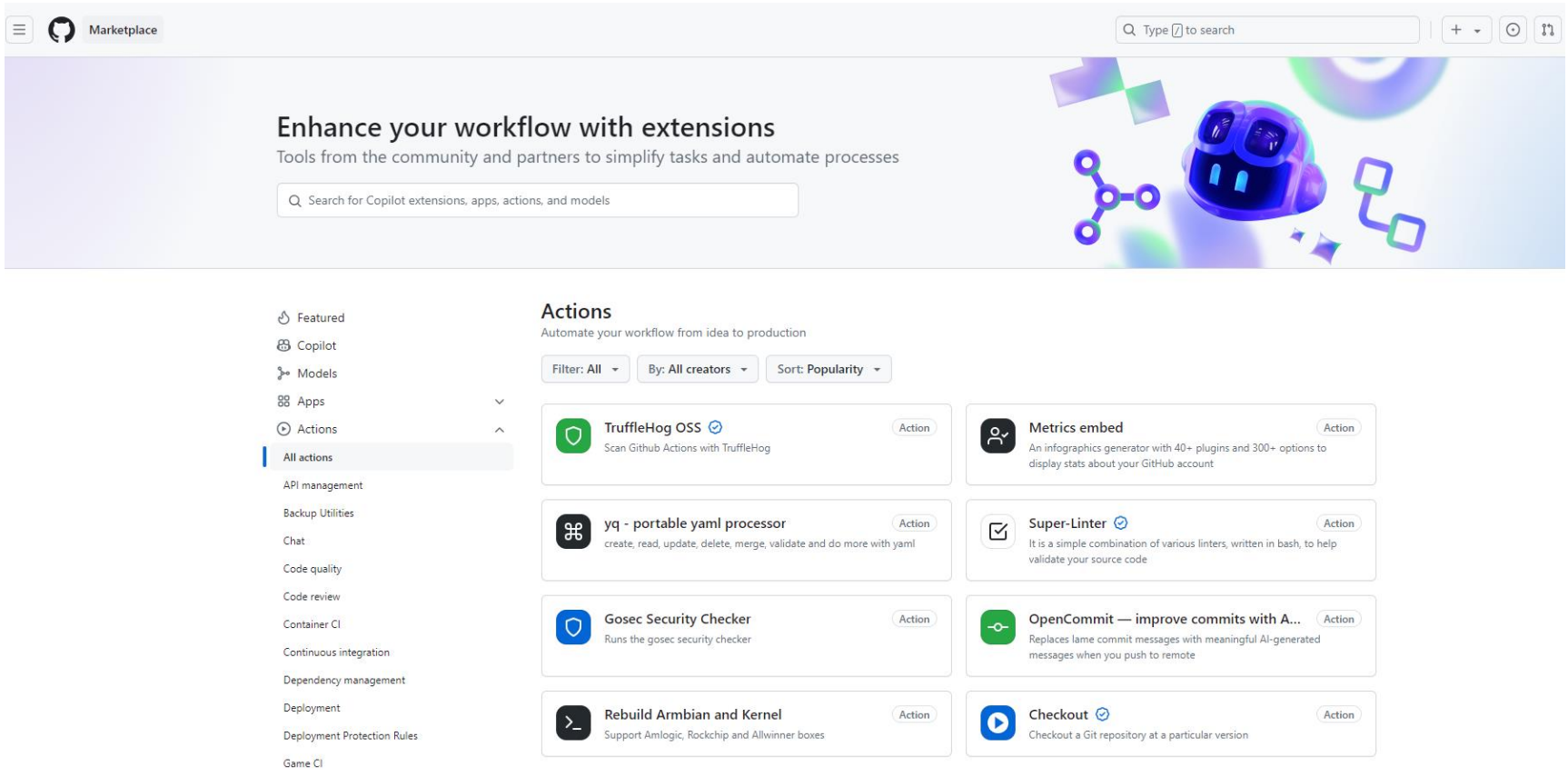
- Realizar o Build ou compilação do código para confirmar que não possui erros de escrita de código;
- Realizar uma bateria de testes unitários para identificar se não possui erros funcionais no sistema ;
- Realizar uma análise de padrões e boas práticas, tais como: nome de variáveis, quantidade de comentários, uso indevido de operações ou comandos;
- Realizar a execução do deploy do sistema em um ambiente qualquer;
- Enviar e-mails e alertas de operações;
- Solicitar aprovação de ações realizadas.

GitHub Actions



GitHub Actions - Marketplace

https://github.com/marketplace?type=actions



GitHub Actions - YAML

- GitHub Actions usa a sintaxe YAML para escrever os Pipelines
- Cada pipeline é armazenado como um arquivo com extensão yml e deve ficar salvo no diretório chamado .github/workflows
- O cabeçalho base de qualquer pipeline escrito em YAML no GitHub é:

```
name: nome_do_pipeline

on: [xxxxx]

jobs:
```

GitHub Actions – Exemplo de um Pipeline em YAML

```yaml
name: GitHub Pipeline de Exemplo
run-name: Teste de GitHub Actions
on: [push]
jobs:
  Explore-GitHub-Actions:
    runs-on: ubuntu-latest
    steps:
      - run: echo "Job iniciado automaticamente"
      - run: echo "Job executando no ${{ runner.os }} server hospedado pelo GitHub!"
      - run: echo "Branch ${{ github.ref }} e repositório ${{ github.repository }}."
      - name: Copiando codigo para o ambiente do GitHub Actions
        uses: actions/checkout@v4
      - run: echo "O repositório ${{ github.repository }} foi clonado no servidor do Github."
      - name: Listando arquivos copiados para o ambiente do GitHub Actions
        run: |
          ls ${{ github.workspace }}
      - run: echo "Situação do Job ${{ job.status }}."
      - run: echo "Pipeline Encerrado"
```

GitHub Actions – Exemplo de um Pipeline em YAML

```yaml
name: GitHub Pipeline de Exemplo
run-name: Teste de GitHub Actions
on: [push]
jobs:
  Explore-GitHub-Actions:
    runs-on: ubuntu-latest
    steps:
      - run: echo "Job iniciado automaticamente"
      - run: echo "Job executando no ${{ runner.os }} server hospedado pelo GitHub!"
      - run: echo "Branch ${{ github.ref }} e repositório ${{ github.repository }}."
      - name: Copiando codigo para o ambiente do GitHub Actions
        uses: actions/checkout@v4
      - run: echo "O repositório ${{ github.repository }} foi clonado no servidor do Github."
      - name: Listando arquivos copiados para o ambiente do GitHub Actions
        run: |
          ls ${{ github.workspace }}
      - run: echo "Situação do Job ${{ job.status }}."
      - run: echo "Pipeline Encerrado"
```

Nome do Pipeline

Quando será executado.
- Pull Request
- Push
- Commit
- Create

Onde o pipeline será executado

GitHub Actions – Exemplo de um Pipeline em YAML

- Criar um exemplo do Pipeline com o código anterior
- Mostrar o resultado após o Push

```
name: GitHub Pipeline de Exemplo
run-name: Teste de GitHub Actions
on: [push]
jobs:
 Explore-GitHub-Actions:
  runs-on: ubuntu-latest
  steps:
   - run: echo "Job iniciado automaticamente"
   - run: echo "Job executando no ${{ runner.os }} server hospedado pelo GitHub!"
   - run: echo "Branch ${{ github.ref }} e repositório ${{ github.repository }}."
   - name: Copiando codigo para o ambiente do GitHub Actions
     uses: actions/checkout@v4
   - run: echo "O repositório ${{ github.repository }} foi clonado no servidor do Github."
   - name: Listando arquivos copiados para o ambiente do GitHub Actions
     run: |
       ls ${{ github.workspace }}
   - run: echo "Situação do Job ${{ job.status }}."
   - run: echo "Pipeline Encerrado"
```

GitHub Actions – Exemplo de um Pipeline Real

```
name: Pipeline Teste
on
 pull_request:
   branches:
    - main
jobs:
  Tarefa1:
   runs-on: ubuntu-latest
   steps:
     - name: Passo1-Puxar última versão do código
          uses: actions/checkout@v2
     - name: Passo2-Instalar Node.JS
          uses: actions/setup-node@v1
          with:
            node-version: "12.x"
     - name: Passo3-Instalar Dependencias
          run: npm ci
     - name: Passo4-Rodar os testes
          run: npm run test:ci
```

Dúvidas???