DevOps

UNIVERSIDADE DO VALE DO SAPUCAÍ – UNIVAS – POUSO ALEGRE/MG
PROFESSOR: RAFFAEL CARVALHO

Protegendo a “main” e Publicando no Registro

Objetivos

- Entender o que é "Proteção de Branch" (Branch Protection) e por que ela é vital.
- Ver a CI atuando como um "status check" obrigatório em um Pull Request.
- Aprender a publicar (fazer push) de um artefato (imagem Docker) em um registro.
- Será usado o mesmo projeto em Node da aula “**Integração Contínua**”.

Parte 1: Proteger a Branch main

1. No seu repositório do GitHub, vá em **Settings > Branches**.
2. Clique em "**Add branch protection rule**".
3. Em "Branch name pattern", digite `main`.
4. Marque a opção "**Require status checks to pass before merging**".
5. Logo abaixo, uma caixa de busca aparecerá. Marque a opção "**Require status checks to be up-to-date before merging**".
6. Na caixa de busca "**Status checks that are required**", digite o nome do seu job. No nosso caso: `test-and-build`. (Basta digitar test e ele deve aparecer).
7. Clique em "**Create**" na parte inferior.

Parte 2: Modificar o Pipeline para Publicar a Imagem

O desafio é: não queremos publicar a imagem em *todo Pull Request*. Queremos apenas testar no PR e, *somente* após o *merge* (ou seja, em um push para a main), publicar a imagem.

Para isso, o aluno deve:

- Criar uma nova *branch* (ex: git checkout -b dev).
- Modificar o arquivo .github/workflows/ci.yml.
- **A Modificação no ci.yml:**
  1. Adicionar 2 novos *steps* **antes** do *step* "**Build e Push da Imagem Docker**" que vai substituir "**Build da Imagem Docker**".
  2. Adicionar um bloco **permissions** logo abaixo do nome do seu *job* (runs-on: ubuntu-latest):

```
12    test-and-build:
13      # A máquina virtual que o job usará
14      runs-on: ubuntu-latest
15      permissions:
16        contents: read # Permissão para fazer checkout do código
17        packages: write # Permissão para FAZER PUSH de pacotes (imagem Docker)
```

A Modificação no ci.yml

```
# ... (depois do step 'Configurar Docker Buildx')
      # 6. Logar no GitHub Container Registry (GHCR)
      # Este step só roda se for um push na branch main
      - name: Logar no GitHub Container Registry
        if: github.event_name == 'push' && github.ref == 'refs/heads/main'
        uses: docker/login-action@v3
        with:
          registry: ghcr.io
          username: ${{ github.actor }}
          password: ${{ secrets.GITHUB_TOKEN }} # Segredo automático do GitHub!
      # 7. Extrair metadados (tags) para o Docker
      # Este step só roda se for um push na branch main
      - name: Extrair Metadados do Docker
        if: github.event_name == 'push' && github.ref == 'refs/heads/main'
        id: meta
        uses: docker/metadata-action@v5
        with:
          images: ghcr.io/${{ github.repository }} # ghcr.io/SEU_USUARIO/SEU_REPO
          tags: |
            type=sha
            latest
    # 8. Build E PUSH da Imagem Docker
    # Note que este step agora tem um 'if' e usa os 'labels' e 'tags' do step anterior
    - name: Build e Push da Imagem Docker
      uses: docker/build-push-action@v5
      with:
        context: .
        file: ./Dockerfile
        # Só faz PUSH se for um push na branch main
        push: ${{ github.event_name == 'push' && github.ref == 'refs/heads/main' }}
        # Usa as tags geradas pelo 'metadata-action'
        tags: ${{ steps.meta.outputs.tags }}
        labels: ${{ steps.meta.outputs.labels }}
```

Parte 3: Executar os passos

1. O aluno deve fazer o git push desta nova branch (dev).
2. Ir ao GitHub e abrir um **Pull Request** (PR) da branch dele para a main.
3. **Observar (1):** O GitHub mostrará o pipeline rodando. Ao final, ele verá que o step "Logar...", "Extrair Metadados..." e o push da imagem foram pulados (skipped), pois não era um push na main.
4. **Observar (2):** O GitHub agora mostrará que o PR está "liberado para merge" (o "check" da CI passou).
5. O aluno deve fazer o merge do PR.
6. **Observar (3):** O merge gera um push na main, o que dispara o pipeline novamente.
7. Agora, o aluno deve ir na aba "Actions" e olhar o log deste novo pipeline (o que rodou na main). Ele verá que os steps "Logar...", "Extrair Metadados..." e o push da imagem foram executados!

Entrega da Atividade

Envio do link do repositório no GitHub.

Bibliografia Básica

- PRESSMAN, Roger S. Engenharia de Software: uma abordagem profissional.8. ed. São Paulo: McCraw Hill – Artmedia, 2016.
- BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. UML
- Guia do Usuário. Rio de Janeiro: Campus, 2012.
- JOHNSON, Ralph; VLISSIDES, John; HELM, Richard; GAMMA, Erich. Padrões de Projeto. São Paulo: Bookman, 2008.