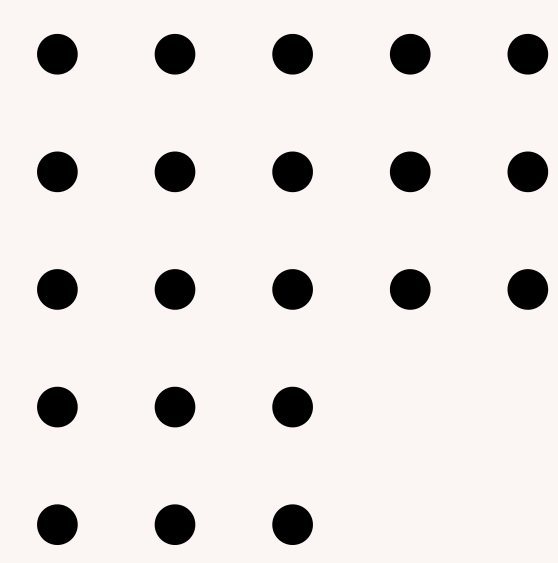
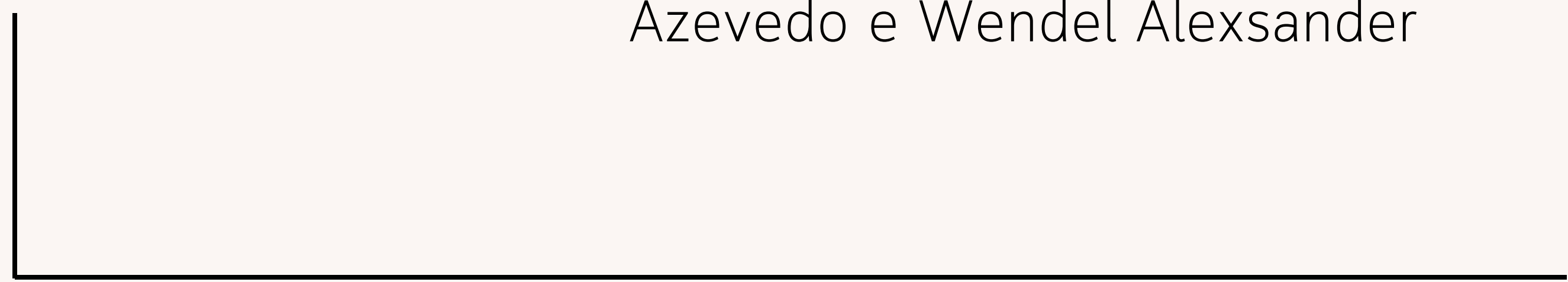


DIAGNÓSTICO E AUDITORIA DE PROJETO

Integrantes: Breno Henrique, Carla Stefany, Fernanda
Karoliny, João Paulo, João Vinícius, José Arthur, Vinícius
Azevedo e Wendel Alexander

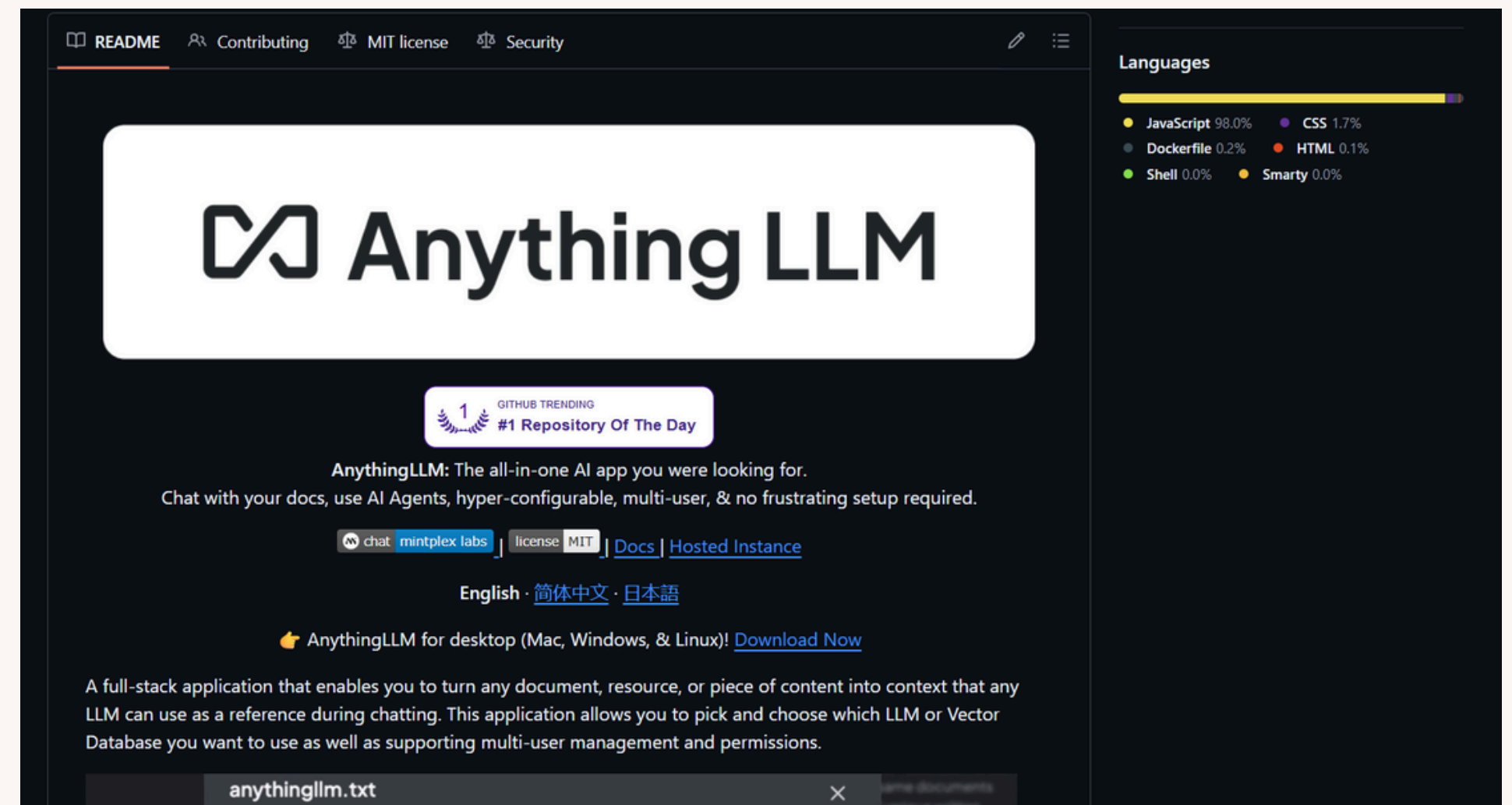


PARTICIPANTES

| Nome | Matrícula | Contribuição |
|------------------------------------|--------------|---------------------------------|
| Breno Henrique Do Carmo Santos | 202200078737 | Pesquisa e edição do documento. |
| Carla Stefany R. Santos | 202400060148 | Pesquisa e edição do documento. |
| Fernanda Karoliny Santos Silva | 202200092431 | Pesquisa e edição do documento. |
| João Paulo Menezes Machado | 202300038743 | Pesquisa e edição do documento. |
| João Vinícius De Almeida Argolo | 202200025573 | Pesquisa e edição do documento. |
| José Arthur Calixto Da Rocha Costa | 202300038770 | Pesquisa e edição do documento. |
| Vinicius Azevedo Peroba | 201900076892 | Pesquisa e edição do documento. |
| Wendel Alexsander Gomes Menezes | 202300027740 | Pesquisa e edição do documento. |

PROJETO

- Projeto escolhido: Anything LLM
- Analisar a maturidade de um projeto real sob a perspectiva de DevOps

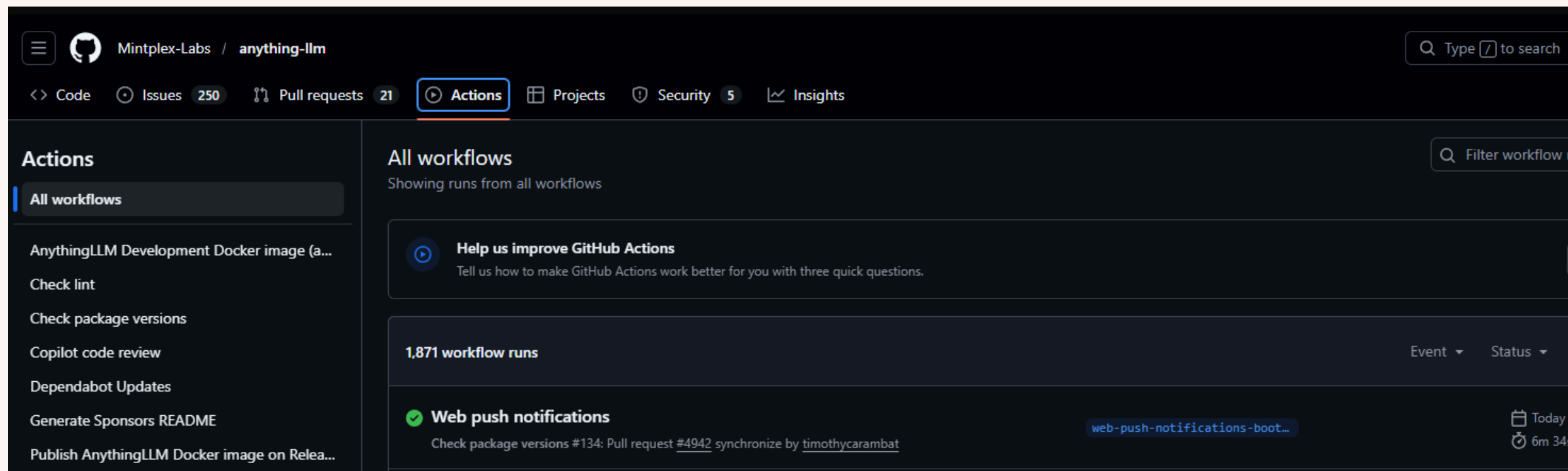


CONCEITOS IMPORTANTES

- Integração Contínua (CI)
- *Automação de Pipeline*
- *Manutenibilidade e Regressão*
- *Ferramentas de Orquestração*

Diagnóstico de CI/CD e Evidências

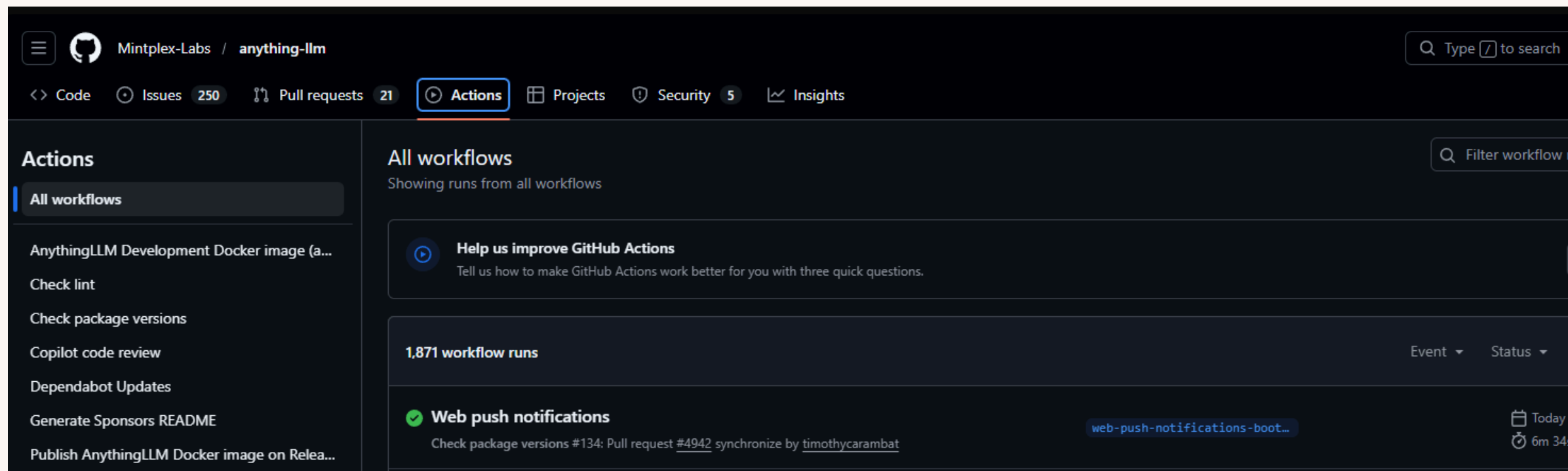
- **Uso de Integração Contínua**



- **Utilização do GitHub Actions**
- **CI/CD integrado ao fluxo de desenvolvimento**

Diagnóstico de CI/CD e Evidências

- **Uso de Entrega Contínua**



- **Integração nativa ao Github**
- **Ausência de ferramentas externas**
- **Adoção de boas práticas recomendadas pelo GitHub**

Diagnóstico de CI/CD e Evidências

- Tipos de Workflow

Actions

All workflows

AnythingLLM Development Docker image ...

Check lint

Check package versions

Copilot code review

Dependabot Updates

Generate Sponsors README

Publish AnythingLLM Docker image on Re...

Publish AnythingLLM Primary Docker ima...

Publish Docker image for PG database de...

Publish Docker image for Render & Railw...

Re-release AnythingLLM Docker image 1....

Re-release AnythingLLM Docker image 1....

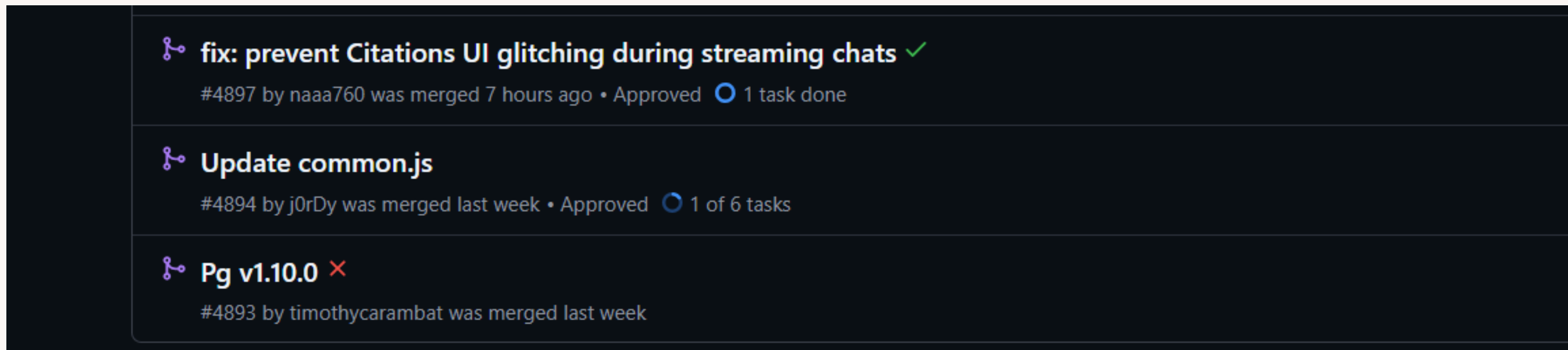
Release AnythingLLM Primary Docker ima...

Run backend tests

Verify translations files

Diagnóstico de CI/CD e Evidências

- Análise de Pull Request



Fluxo Atual de Desenvolvimento

Core Team

1. Abertura de PR
2. Execução de testes automatizados
(com filtragem de caminhos)
3. *Code reviews* opcionais
4. Merge

Fluxo Atual de Desenvolvimento

Desenvolvedores externos

1. Fork (Forking Workflow)
2. Abertura de PR
3. Code review
4. Merge

Fluxo Atual de Desenvolvimento

Filtragem de caminhos

check-translations.yaml

```
11   on:
12     pull_request:
13       types: [opened, synchronize, reopened]
14       paths:
15         - "frontend/src/locales/**/*.js"
16
17   jobs:
18     run-script:
19       runs-on: ubuntu-latest
20
```

run-tests.yaml

```
7     on:
8       pull_request:
9         types: [opened, synchronize, reopened]
10        paths:
11          - "server/**/*.js"
12          - "collector/**/*.js"
13
14   jobs:
15     run-script:
```

Fluxo Atual de Desenvolvimento

Deploy e montagem de imagem Docker

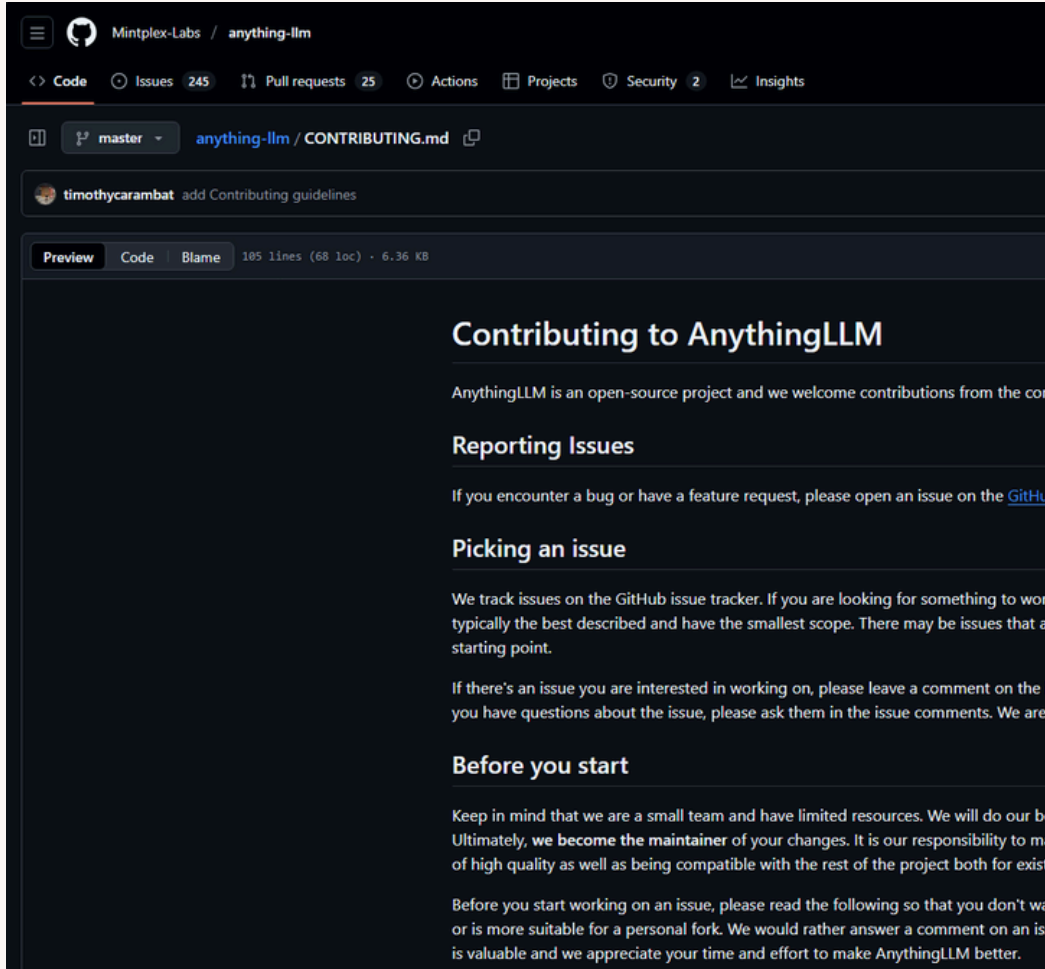
build-and-push-image.yaml

```
13   on:
14     push:
15       branches: ['master'] # master branch
16     paths-ignore:
17       - '**.md'
18       - '.gitmodules'
19       - 'cloud-deployments/**/*'
20       - 'images/**/*'
21       - '.vscode/**/*'
```

```
run: |
  # Check if the secret for USERNAME is set (don't ev
  if [[ -z "${{ secrets.DOCKER_USERNAME }}" ]]; then
    echo "DockerHub build not needed"
    echo "enabled=false" >> $GITHUB_OUTPUT
  else
    echo "DockerHub build needed"
    echo "enabled=true" >> $GITHUB_OUTPUT
  fi
```

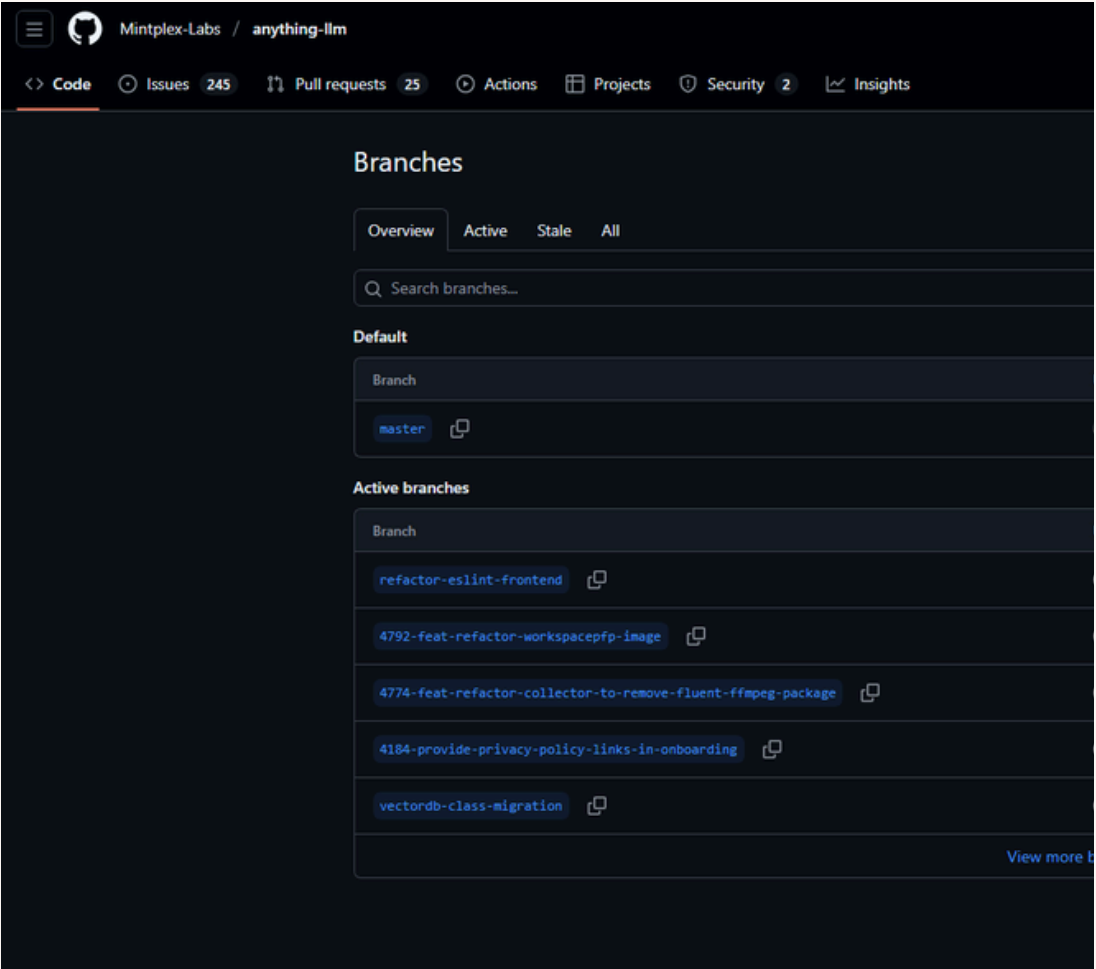
Evidências Encontradas

Contribue.md



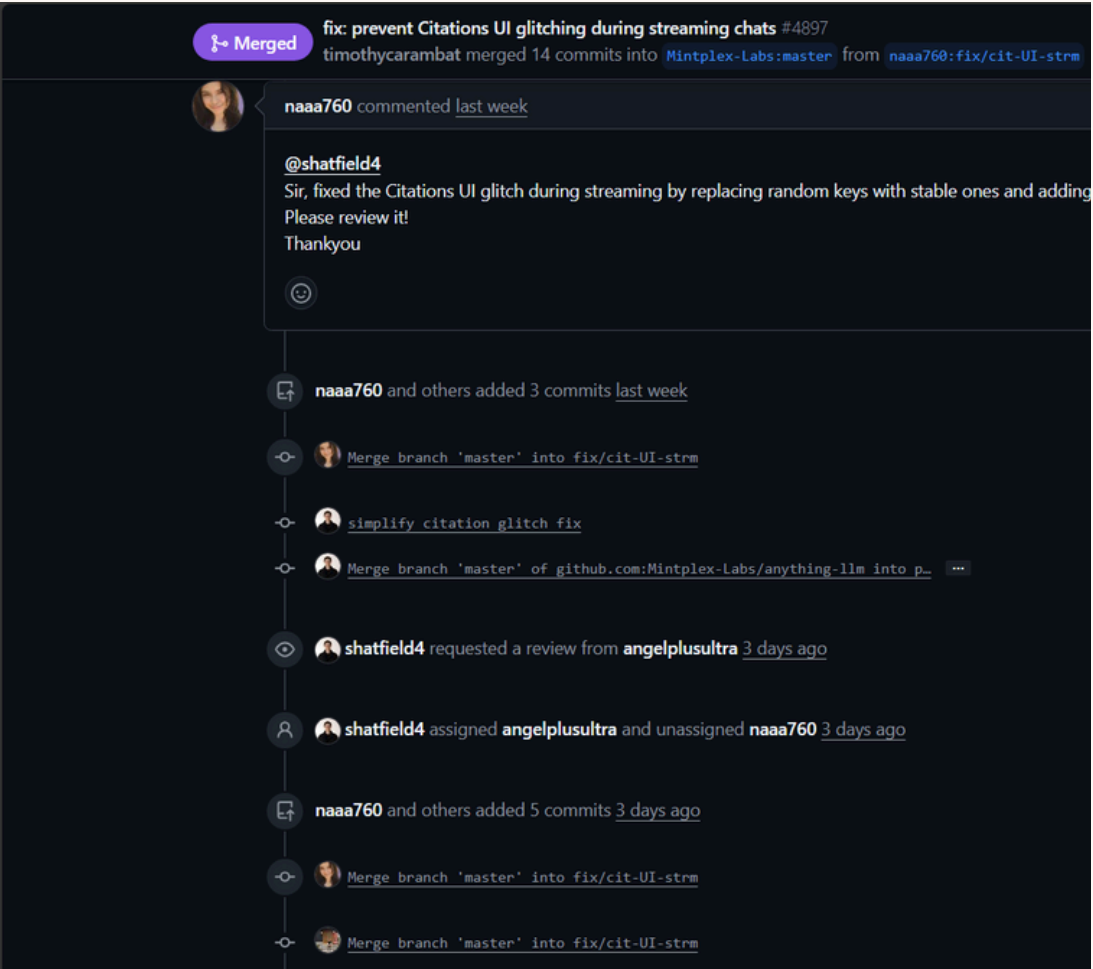
Fonte: GitHub
AnythingLLM

Branches



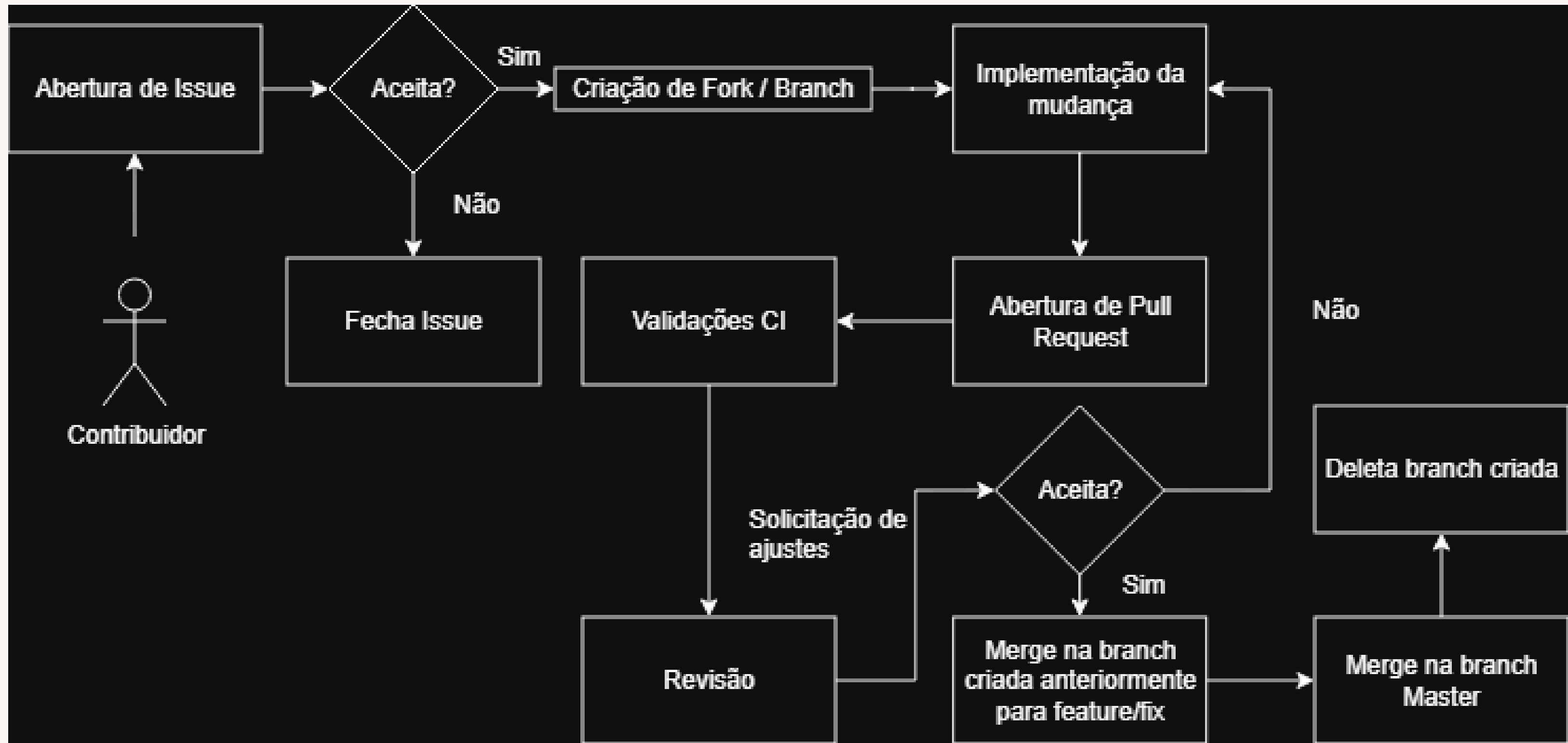
Fonte: GitHub
AnythingLLM

Pull Request



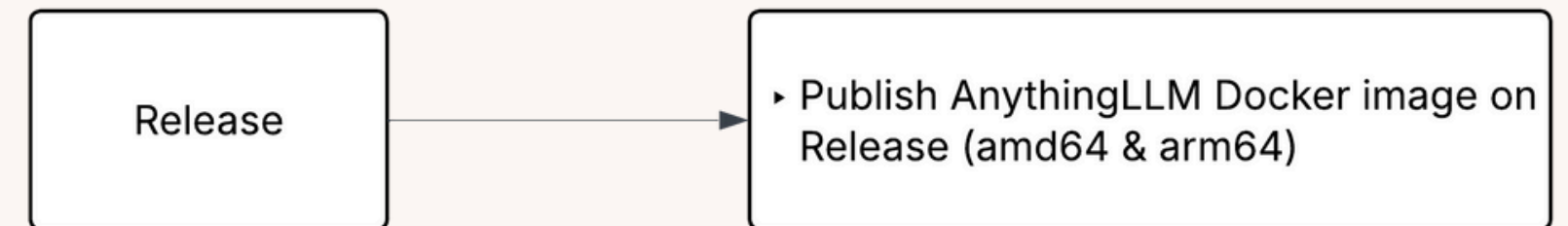
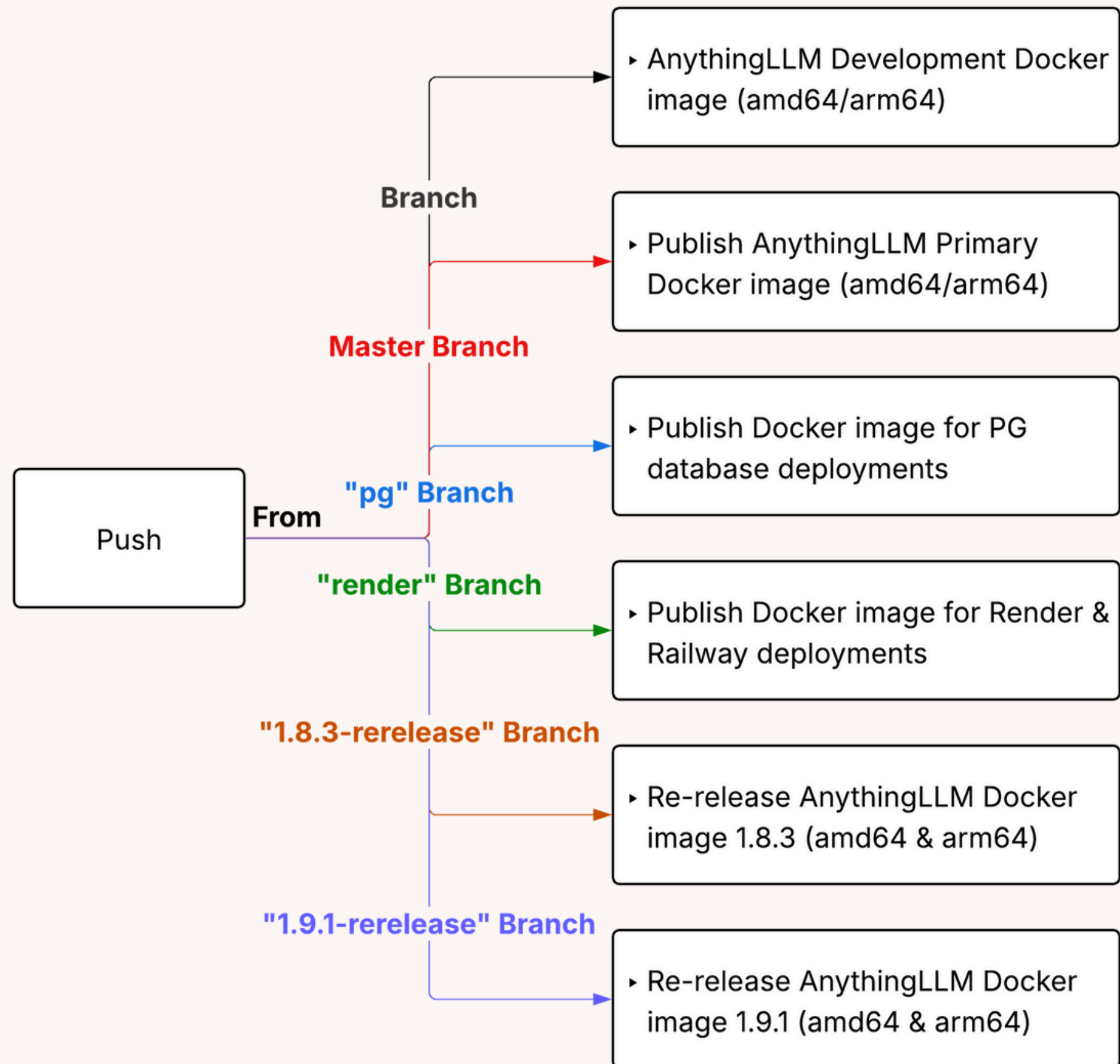
Fonte: GitHub
AnythingLLM

Diagrama de Processos

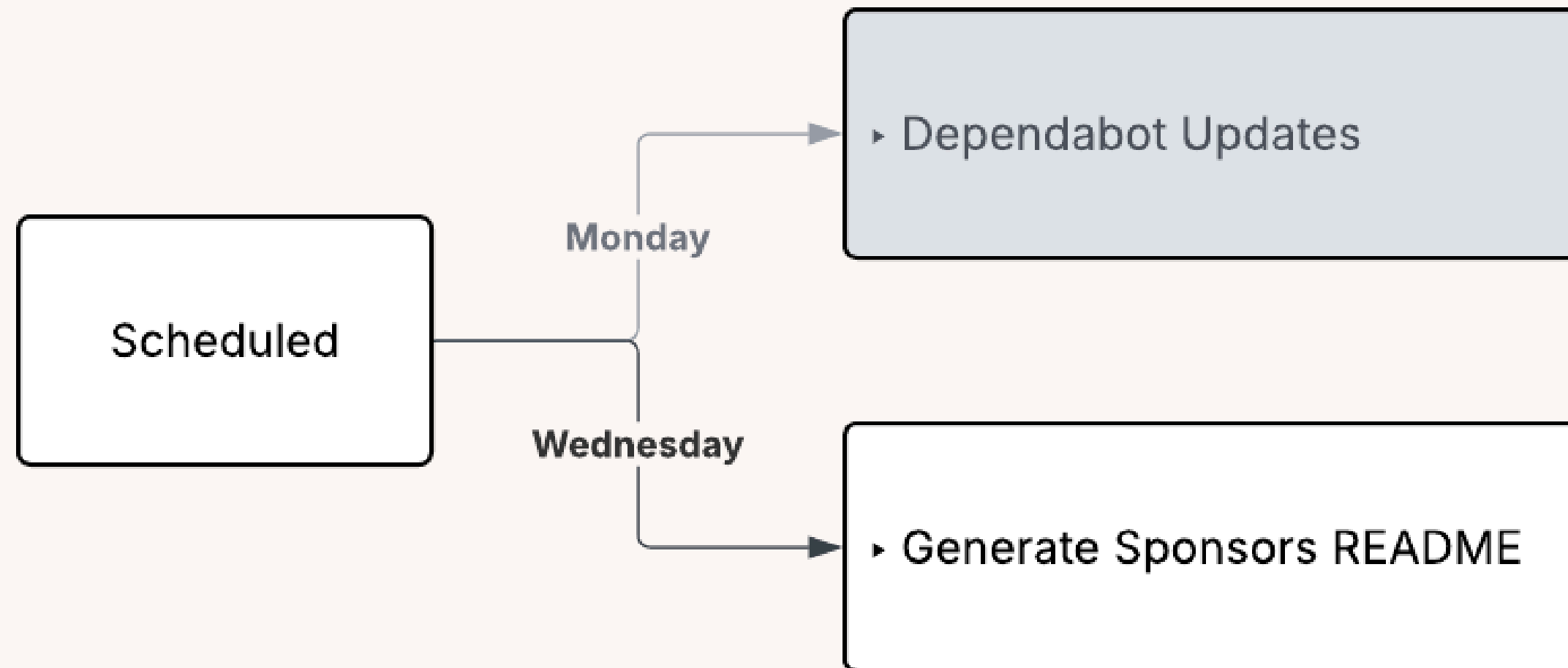


Fonte: Autor

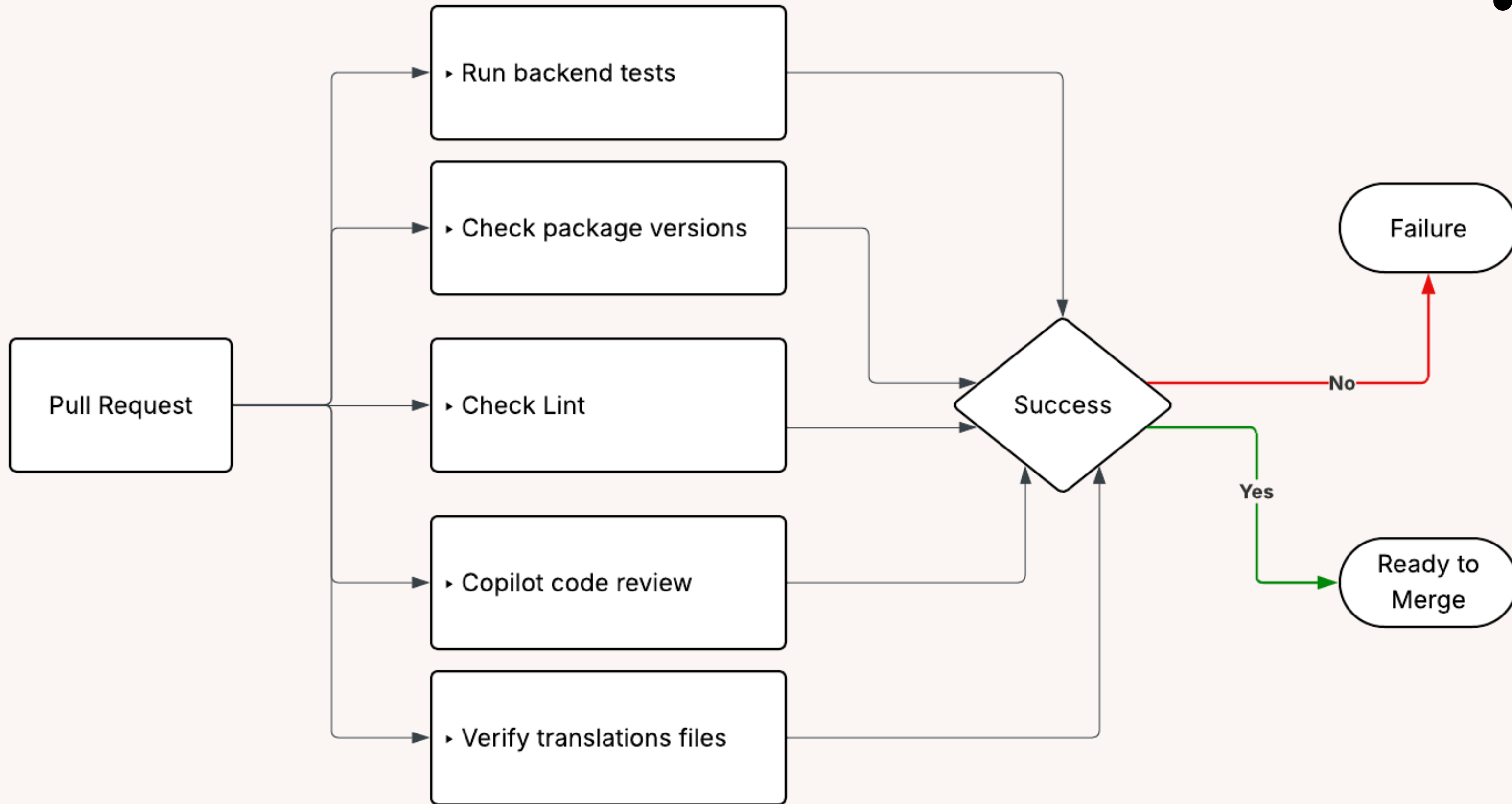
Automação



Automação



Automação

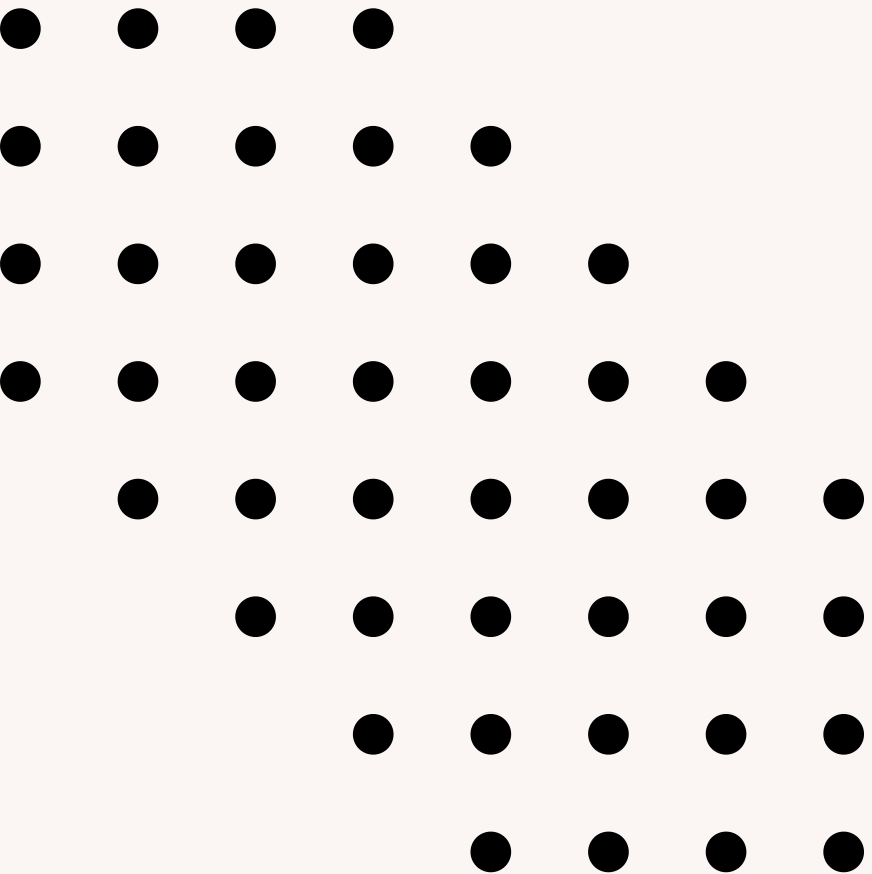


Riscos de Regressão

- **Riscos de falhas chegarem a produção:** A alta frequência de commits na branch principal, seguida rapidamente por builds e publicações de imagens Docker, indica que eventuais falhas não capturadas pelos testes automatizados podem ser propagadas para a versão distribuída. Essa conclusão vem da análise do curto intervalo entre pipelines de teste e publicação.
- **Risco de regressões por dependências externas:** Os workflows de build mostram reconstruções frequentes de imagens Docker e uso de bibliotecas de terceiros sem validação rigorosa de versões, o que pode introduzir mudanças inesperadas entre builds e causar regressões no comportamento do sistema.
- **Risco organizacional por centralização de acessos:** As etapas críticas de publicação dependem de segredos e credenciais acessíveis apenas a poucos mantenedores, aumentando a dependência de pessoas-chave e podendo comprometer a continuidade do processo em caso de indisponibilidade.

Gargalos e Limitações

- **Dependência de validação manual:** Mesmo com a presença de Workflows automatizados, a ausência de quality gates bloqueantes faz com que a efetivação de merges esteja sujeita a revisões humanas (o que tende a gerar gargalos em projetos de grande porte).
- **Centralização de decisões:** A distinção de privilégios e concentração de tomada de decisões a um grupo restrito acabam reduzindo a autonomia do projeto.
- **Assimetria na validação de imagens de Docker:** Os workflows de construção e publicação de Docker não são executados para merges provenientes de forks. Apesar de dirimir riscos de segurança, ela reduz a cobertura de validação e pode atrasar a identificação de falhas.
- **Baixo nível de automação:** A inexistência de automação para builds intermediários e dependência de reconfigurações manuais dos branches comprometem a reprodutibilidade do ambiente e a evolução contínua, gerando gargalos operacionais.



OBRIGADO !