

Universidade Federal de Sergipe
Engenharia de Software II

Diagnóstico e Auditoria do Projeto LangExtract
Atividade 3 – Etapa 1

Membros:

José Fernando Bispo dos Santos – 202200014210
Valter Fabricio dos Santos – 202000066991
Raphael Ferreira Portella Bacelar - 202100045822

Professor: Dr. Glauco de Figueiredo Carneiro
Janeiro de 2026

SUMÁRIO

1. INTRODUÇÃO.....	4
2. DESCRIÇÃO DO PROJETO AVALIADO.....	4
3. METODOLOGIA DE INVESTIGAÇÃO.....	4
4. FERRAMENTAS DE CI/CD UTILIZADAS.....	5
5. EVIDÊNCIAS TÉCNICAS.....	5
6. FLUXO ATUAL.....	8
7. CONCLUSÃO.....	8

1. INTRODUÇÃO

A evolução contínua de sistemas de software exige mecanismos que garantam a qualidade e a estabilidade do código ao longo do tempo. Nesse contexto, a Integração Contínua (Continuous Integration – CI) surge como uma prática fundamental, permitindo que alterações no código sejam verificadas automaticamente, reduzindo riscos de regressão e facilitando a colaboração entre desenvolvedores.

Esta atividade tem como objetivo analisar a maturidade das práticas de CI de um projeto de software open source real, no qual a equipe atua como engenheiros de DevOps. A partir dessa análise, busca-se compreender como a automação influencia a manutenibilidade do sistema e a entrada de novos contribuidores no projeto.

2. DESCRIÇÃO DO PROJETO AVALIADO

O projeto analisado nesta atividade é o **LangExtract**, um projeto open source mantido pela Google e disponibilizado na plataforma GitHub.

Repositório: <https://github.com/google/langextract>

LangExtract é uma biblioteca Python que utiliza LLMs para extrair informações estruturadas de documentos de texto não estruturados com base em instruções definidas pelo usuário. Ela processa materiais como anotações clínicas ou relatórios, identificando e organizando detalhes-chave, ao mesmo tempo em que garante que os dados extraídos correspondam ao texto de origem.

3. METODOLOGIA DE INVESTIGAÇÃO

A investigação do projeto foi realizada por meio de uma análise exploratória do repositório no GitHub, com foco nas práticas de Integração Contínua adotadas. Foram observados os seguintes aspectos:

- Existência do diretório `.github/workflows/`;
- Análise dos arquivos YAML de workflows;
- Verificação da aba *Actions* do GitHub;

- Observação do histórico de Pull Requests

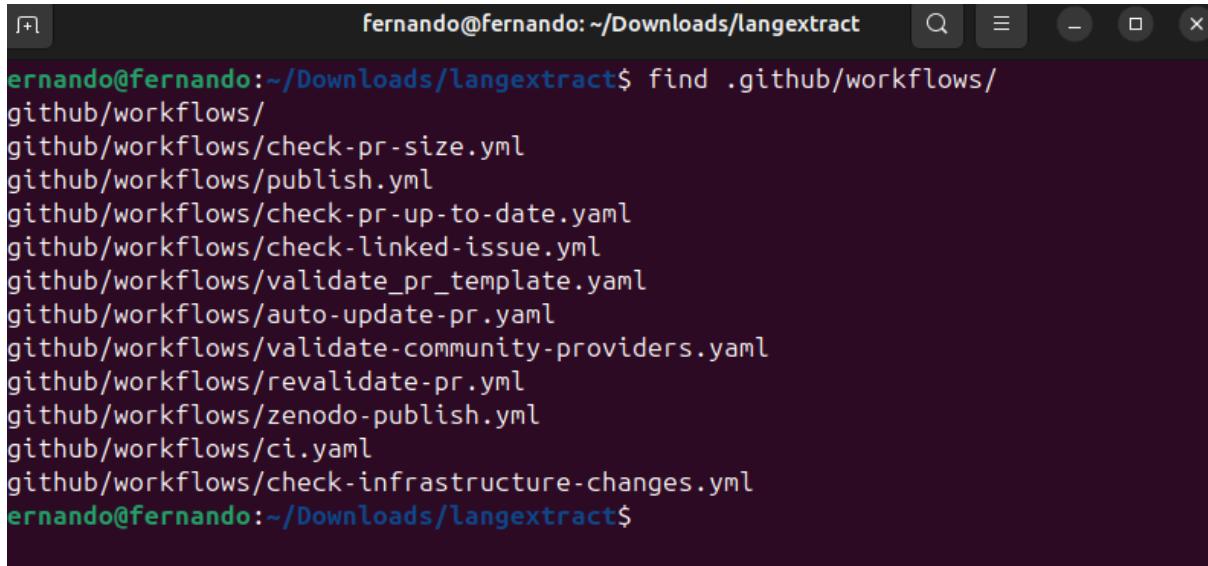
Essa abordagem permitiu mapear o fluxo atual de desenvolvimento e identificar o nível de automação presente no projeto.

4. FERRAMENTAS DE CI/CD UTILIZADAS

O projeto **LangExtract** utiliza o **GitHub Actions** como ferramenta de Integração Contínua (CI). Essa ferramenta é nativa da plataforma GitHub e permite a execução automática de tarefas em resposta a eventos como *push* e *pull request*.

5. EVIDÊNCIAS TÉCNICAS

Como evidência da utilização de CI, foram identificados diversos arquivos de workflow no diretório `.github/workflows/`. Esses arquivos definem ações automatizadas que são executadas durante o ciclo de desenvolvimento do projeto.



```
fernando@fernando: ~/Downloads/langextract$ find .github/workflows/
.github/workflows/
.github/workflows/check-pr-size.yml
.github/workflows/publish.yml
.github/workflows/check-pr-up-to-date.yaml
.github/workflows/check-linked-issue.yml
.github/workflows/validate_pr_template.yaml
.github/workflows/auto-update-pr.yaml
.github/workflows/validate-community-providers.yaml
.github/workflows/revalidate-pr.yaml
.github/workflows/zenodo-publish.yaml
.github/workflows/ci.yaml
.github/workflows/check-infrastructure-changes.yaml
fernando@fernando: ~/Downloads/langextract$
```

Complementarmente, foi realizada uma análise do histórico de commits. Essa verificação permitiu identificar o primeiro commit diretamente relacionado à criação e evolução dos workflows de CI. O commit 6d336c1, no qual foi adicionada uma pipeline de GitHub Actions voltada para a execução de linting e testes automatizados, com suporte a múltiplas versões

do Python (3.10 e 3.11), além da introdução da ferramenta tox para gerenciamento de ambientes de teste.

Commit 6d336c1

 **aksg87** committed on Jul 18, 2025 · ✓ 2/2

```
ci(LangExtract): Add workflow for linting and tests
Adds a GitHub Actions CI workflow to run linting and tests against Python 3.10 and 3.11.

- Introduces `tox` for managing test environments.
- Updates `pyproject.toml` to require Python 3.10+ and add `tox` as a development dependency.
- Excludes docs, tests, and other non-essential files from the distributable package via `pyproject.toml`.

main ·  v1.1.1 · v1.0.0
```

Filter files...

- .github/workflows
 - + ci.yaml
 - + .pylintrc
 - + README.md
 - + pyproject.toml
 - + tox.ini

5 files changed +116 -4 lines changed

... @@ -0,0 +1,47 @@

```
1 + # Copyright 2025 Google LLC.
2 +
3 + # Licensed under the Apache License, Version 2.0 (the "License");
4 + # you may not use this file except in compliance with the
5 + # You may obtain a copy of the License at
6 +
7 + #     http://www.apache.org/licenses/LICENSE-2.0
8 +
9 + # Unless required by applicable law or agreed to in writing,
10 + # distributed under the License is distributed on an "AS
11 + # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either ex-
12 + # See the License for the specific language governing per-
13 + # limitations under the License.
14 +
15 + name: CI
16 +
17 + on:
18 +   push:
19 +     branches: ["main"]
20 +   pull_request:
21 +     branches: ["main"]
22 +
23 + permissions:
24 +   contents: read
25 +
26 + jobs:
27 +   test:
```

Além disso, a aba *Actions* do GitHub apresenta um histórico consistente de execuções dos workflows, indicando que os pipelines estão ativos e em uso contínuo.

The screenshot shows the GitHub Actions interface for the repository "langextract". The left sidebar has "Actions" selected under "Code & Issues". The main area displays "All workflows" with 2,882 workflow runs. A search bar at the top right says "Type / to search". The table lists workflow runs with columns for Event, Status, Branch, Actor, and a timestamp. Most runs are labeled "SCHEDULED" or "PENDING". Some runs are triggered by CI or feature branches like "feat(gemini)". The last run listed is "feat(gemini): add thinking config to handled_keys support Gemini 2.0" from Jan 22, 12:27 AM GMT-3.

	Event	Status	Branch	Actor	Time	...
Auto Update PR	SCHEDULED	PENDING	main	CI	Jan 25, 11:48 PM GMT-3	...
Auto Update PR	SCHEDULED	PENDING	main	CI	Jan 24, 11:48 PM GMT-3	...
Auto Update PR	SCHEDULED	PENDING	main	CI	Jan 23, 11:37 PM GMT-3	...
Auto Update PR	SCHEDULED	PENDING	main	CI	Jan 22, 11:40 PM GMT-3	...
feat(gemini): add thinking config to handled_keys support Gemini 2.0	PULL REQUEST #278 SCHEDULED	PENDING	main	CI #457: Pull request #278 synchronize by Eaton	Jan 22, 12:27 AM GMT-3	...
feat(gemini): add thinking config to handled_keys support Gemini 2.0	PULL REQUEST #279 SCHEDULED	PENDING	main	CI #458: Pull request #279 synchronize by Eaton	Jan 22, 12:27 AM GMT-3	...
feat(gemini): add thinking config to handled_keys support Gemini 2.0	PULL REQUEST #326 SCHEDULED	PENDING	main	Protect Infrastructure Files #326: Pull request #326 synchronize by Eaton	Jan 22, 12:27 AM GMT-3	...
feat(gemini): add thinking config to handled_keys support Gemini 2.0	PULL REQUEST #327 SCHEDULED	PENDING	main	Check PR size-to-date #327: Pull request #327 synchronize by Eaton	Jan 22, 12:27 AM GMT-3	...
feat(gemini): add thinking config to handled_keys support Gemini 2.0	PULL REQUEST #328 SCHEDULED	PENDING	main	Require linked issue with community support #463: Pull request #328 synchronize by Eaton	Jan 22, 12:27 AM GMT-3	...
feat(gemini): add thinking config to handled_keys support Gemini 2.0	PULL REQUEST #329 SCHEDULED	PENDING	main	Validate PR template #552: Pull request #329 synchronize by Eaton	Jan 22, 12:27 AM GMT-3	...
feat(gemini): add thinking config to handled_keys support Gemini 2.0	PULL REQUEST #330 SCHEDULED	PENDING	main	Check PR size #336: Pull request #330 synchronize by Eaton	Jan 22, 12:27 AM GMT-3	...
feat(gemini): add thinking config to handled_keys support Gemini 2.0	PULL REQUEST #331 SCHEDULED	PENDING	main	Check PR size #336: Pull request #331 synchronize by Eaton	Jan 22, 12:27 AM GMT-3	...

No histórico de Pull Requests, é possível observar a presença de *status checks* ✘ ou ✓, que informam aos revisores se as verificações automatizadas foram concluídas com sucesso ou falha antes da integração do código.

Essas evidências confirmam que o projeto possui automação funcional e integrada ao processo de colaboração.

6. FLUXO ATUAL

Com base na análise realizada, o fluxo atual de desenvolvimento do projeto pode ser descrito da seguinte forma:

- Criação de alterações no código-fonte
 - Os contribuidores realizam modificações no código localmente e submetem essas alterações ao repositório por meio de commits.
- Abertura de Pull Requests (PRs)
 - As mudanças são integradas ao fluxo principal do projeto através da abertura de Pull Requests no GitHub, permitindo revisão por outros colaboradores ou mantenedores.
- Execução automática dos workflows de CI
 - Ao ocorrer um evento de *push* ou a abertura/atualização de um Pull Request, os workflows definidos no diretório `.github/workflows/` são acionados automaticamente pelo GitHub Actions.
- Feedback automatizado
 - Os resultados das execuções são exibidos diretamente no Pull Request por meio de *status checks* visuais (sucesso ou falha). Esse feedback imediato informa se o código atende aos critérios mínimos de qualidade antes da revisão humana.
- Revisão e integração do código
 - Apenas após a conclusão bem-sucedida das verificações automatizadas, o Pull Request pode ser revisado e, eventualmente, aprovado e mesclado ao branch principal do projeto.

7. CONCLUSÃO

A análise do projeto LangExtract demonstra um nível consistente de maturidade no uso de práticas de Integração Contínua. A adoção do GitHub Actions como ferramenta de CI, aliada à execução automatizada de linting e testes em múltiplas versões do Python, contribui diretamente para a manutenção da qualidade do código e para a detecção precoce de erros.

A presença de pipelines ativos e integrados ao fluxo de Pull Requests evidencia uma preocupação com a estabilidade do projeto e com a prevenção de regressões, fatores essenciais para a evolução sustentável do software. Além disso, o feedback automatizado fornecido pelas execuções de CI facilita a entrada de novos contribuidores, pois estabelece critérios claros e objetivos para a aceitação de mudanças.

Dessa forma, conclui-se que o LangExtract possui um processo de desenvolvimento bem estruturado no que se refere à Integração Contínua, reduzindo gargalos manuais e fortalecendo a confiabilidade do projeto ao longo de seu ciclo de vida.