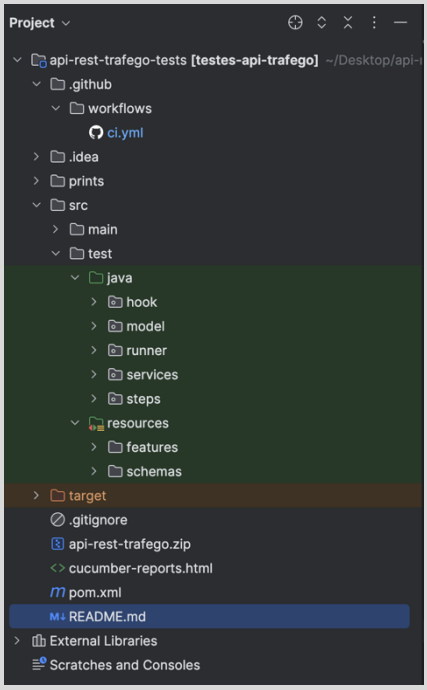
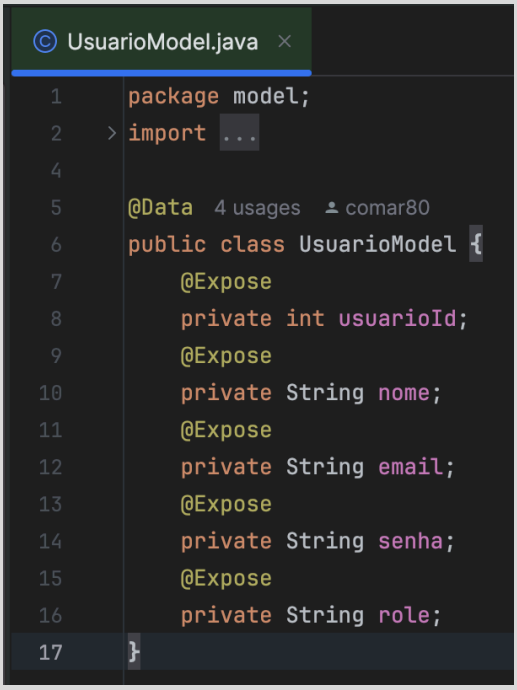
DOCUMENTAÇÃO – API TESTES

TESTES AUTOMATIZADOS COM BEHAVIOR-DRIVEN DEVELOPMENT UTILIZANDO GHERKIN

Organização de pastas

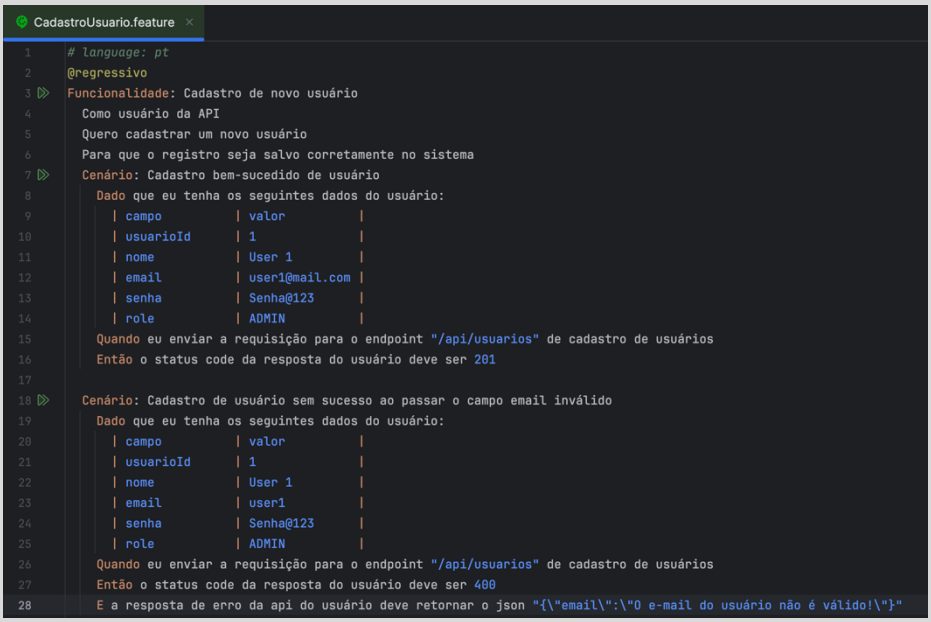


Classe Modelo



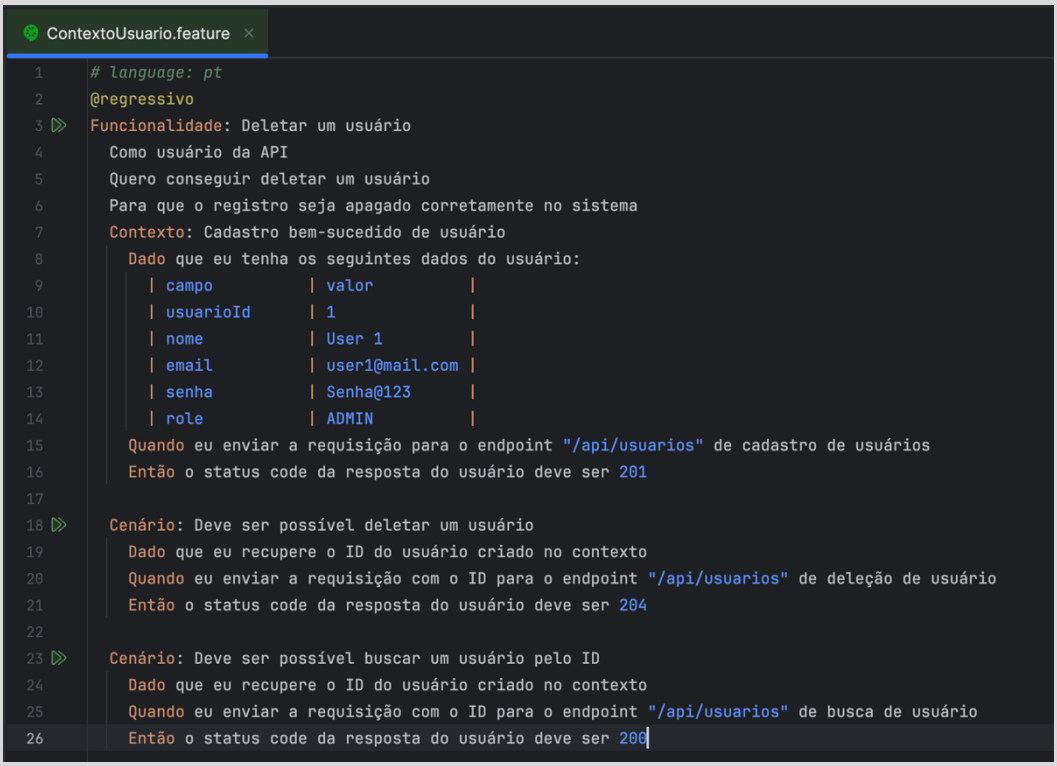
- Com Gherkin, utilizando linguagem natural é possível escrever testes nas classes *features* como os feitos a seguir:

***Feature* de Cadastro com e sem sucesso**



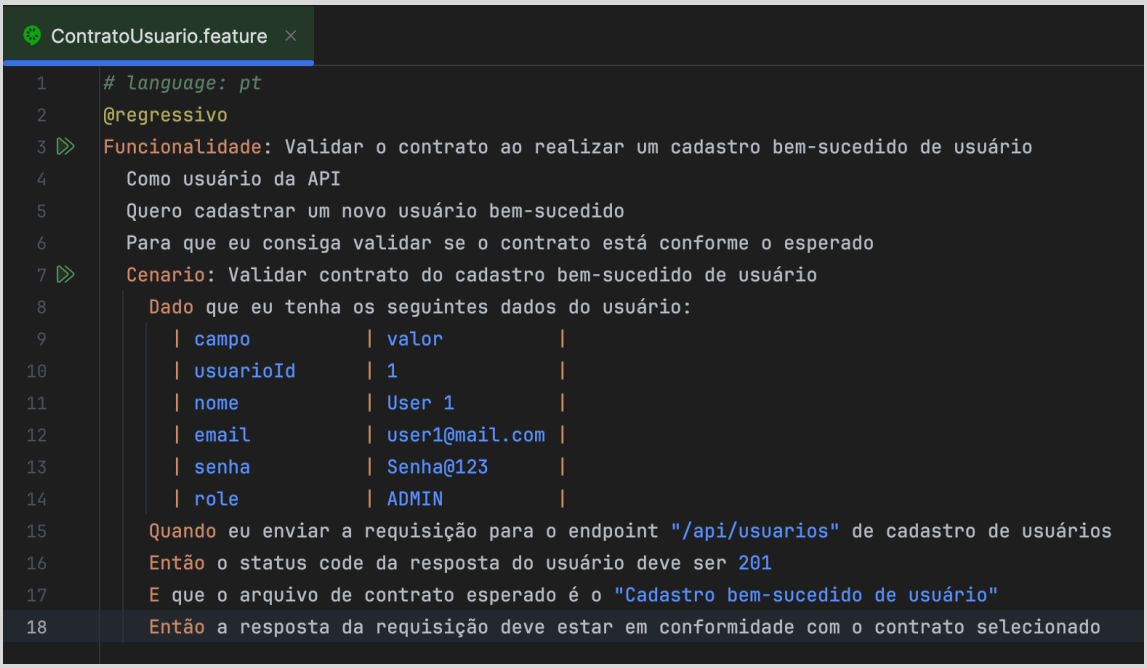
Feature de deleção e leitura usando contexto

- Essa *feature* utiliza o contexto de um cadastro bem-sucedido para buscar por um ID e deletar o usuário criado, ou apenas exibir o usuário.



Feature de validação de contrato

- Essa *feature* valida através de um arquivo json-schema se a resposta da requisição da API principal está de acordo com o conteúdo desse arquivo.



- Essas classes de *features* dependem da lógica das classes *Service* e *Steps* para acessar a API principal e automatizar os testes

Classe Service

CadastroUsuarioSteps.java | CadastroUsuarioService.java

```java
package services;

import ...

public class CadastroUsuarioService {  3 usages  comar80

    String idUsuario;  3 usages
    String schemasPath = "src/test/resources/schemas/";  1 usage
    JSONObject jsonSchema;  2 usages
    private final ObjectMapper mapper = new ObjectMapper();  1 usage

    final UsuarioModel usuarioModel = new UsuarioModel();  6 usages
    public final Gson gson = new GsonBuilder()
            .excludeFieldsWithoutExposeAnnotation()
            .create();
    public Response response;
    String baseUrl = "https://api-trafego-dev-dda3bhdng8gtaqhk.eastus2-01.azurewebsites.net";  3 usages

    public void setFieldsUsuario(String field, String value) {  1 usage  comar80
        switch (field) {
            case "usuarioId" -> usuarioModel.setUsuarioId(Integer.parseInt(value));
            case "nome" -> usuarioModel.setNome(value);
            case "email" -> usuarioModel.setEmail(value);
            case "senha" -> usuarioModel.setSenha(value);
            case "role" -> usuarioModel.setRole(value);
            default -> throw new IllegalStateException("Unexpected field" + field);
        }
    }

    public void createUsuario(String endPoint) {  1 usage  comar80
        String url = baseUrl + endPoint;
        String bodyToSend = gson.toJson(usuarioModel);
        response = given() RequestSpecification
                .contentType(ContentType.JSON)
                .accept(ContentType.JSON)
                .body(bodyToSend)
                .when()
                .post(url) Response
```

Classe Steps

CadastroUsuarioSteps.java

```java
package steps;

import ...

public class CadastroUsuarioSteps {  comar80
        CadastroUsuarioService cadastroUsuarioService = new CadastroUsuarioService();  9 usages

        @Dado("que eu tenha os seguintes dados do usuário:")  comar80
        public void queEuTenhaOsSeguintesDadosDoUsuário(List<Map<String, String>> rows) {
                for (Map<String, String> columns : rows) {
                        cadastroUsuarioService.setFieldsUsuario(columns.get("campo"), columns.get("valor"));
                }
        }

        @Quando("eu enviar a requisição para o endpoint {string} de cadastro de usuários")  comar80
        public void euEnviarARequisiçãoParaOEndpointDeCadastroDeUsuários(String endPoint) {
                cadastroUsuarioService.createUsuario(endPoint);
        }

        @Então("o status code da resposta do usuário deve ser {int}")  comar80
        public void oStatusCodeDaRespostaDoUsuárioDeveSer(int statusCode) {
                Assert.assertEquals(statusCode, cadastroUsuarioService.response.statusCode());
        }

        @E("a resposta de erro da api do usuário deve retornar o json {string}")  comar80
        public void aRespostaDeErroDaApidodoUsuárioDeveRetornarOJson(String message) {
                String errorMessageModel = cadastroUsuarioService.response.getBody().asString();
                Assert.assertEquals(message, errorMessageModel);
        }

        @Dado("que eu recupere o ID do usuário criado no contexto")  comar80
        public void queEuRecupereOIDDoUsuárioCriadoNoContexto() { cadastroUsuarioService.retrieveIdUsuario(); }

        @Quando("eu enviar a requisição com o ID para o endpoint {string} de deleção de usuário")  comar80
        public void euEnviarARequisiçãoComOIDParaOEndpointDeDeleçãoDeUsuário(String endpoint) {
                cadastroUsuarioService.deleteUsuario(endpoint);
        }
```

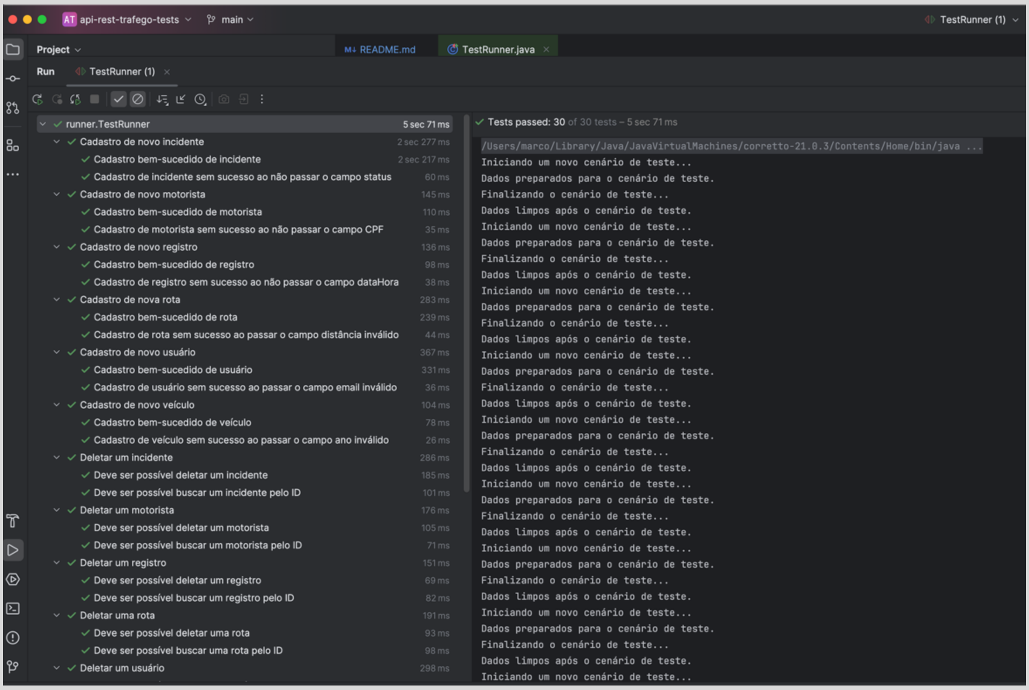
Json-Schema

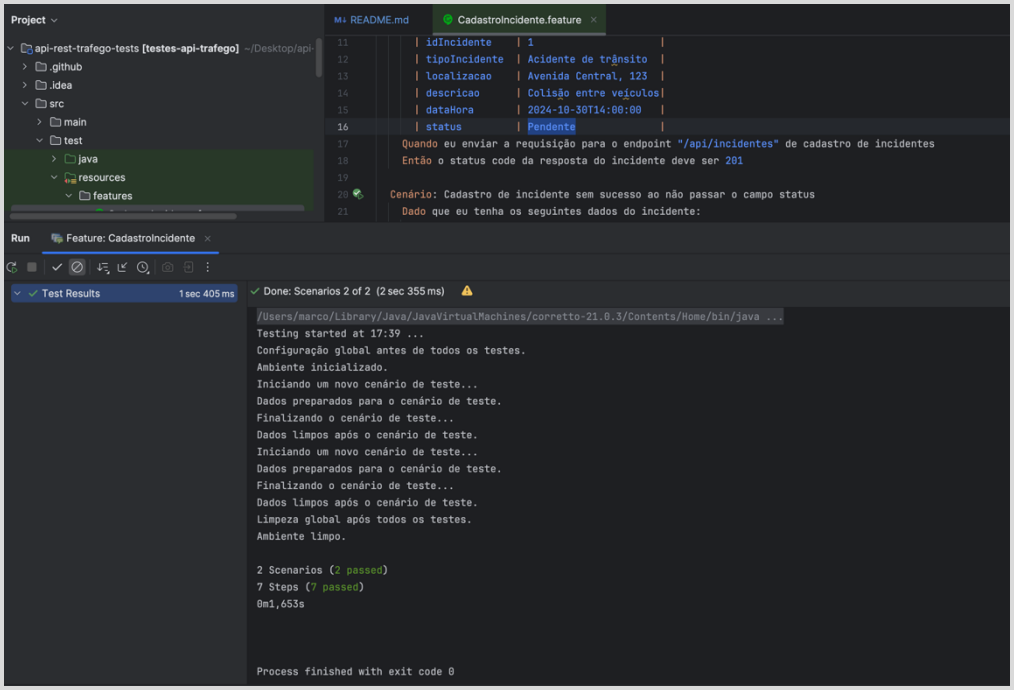
- Utilizado para validar o contrato

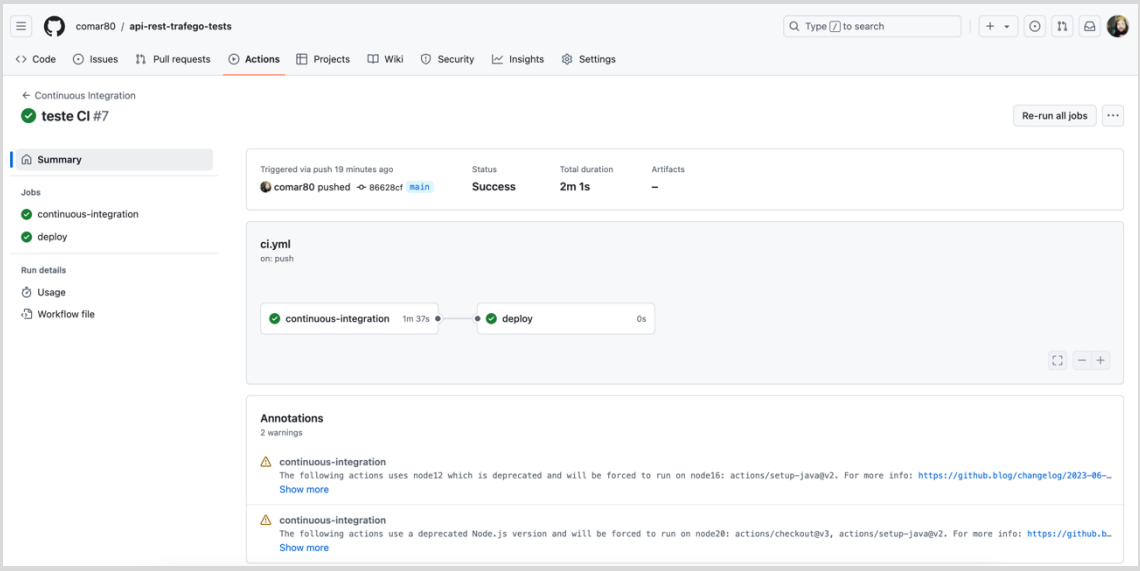
Continuous Integration

- Configuração para *Continuous Integration* via Github Actions

Evidências de Execução




Project
api-rest-trafego-tests [testes-api-trafego] ~/Desktop/api-
.github
.idea
src
main
test
java
resources
features
README.md
CadastroIncidente.feature
idIncidente	1
tipoIncidente	Acidente de trânsito
localizacao	Avenida Central, 123
descricao	Colisão entre veículos
dataHora	2024-10-30T14:00:00
status	Pendente
Quando eu enviar a requisição para o endpoint "/api/incidentes" de cadastro de incidentes
Então o status code da resposta do incidente deve ser 201
Cenário: Cadastro de incidente sem sucesso ao não passar o campo status
Dado que eu tenha os seguintes dados do incidente:
Run
Feature: CadastroIncidente
Test Results
1 sec 405 ms
Done: Scenarios 2 of 2 (2 sec 355 ms)
/Users/marco/Library/Java/JavaVirtualMachines/corretto-21.0.3/Contents/Home/bin/java ...
Testing started at 17:39 ...
Configuração global antes de todos os testes.
Ambiente inicializado.
Iniciando um novo cenário de teste...
Dados preparados para o cenário de teste.
Finalizando o cenário de teste...
Dados limpos após o cenário de teste.
Iniciando um novo cenário de teste...
Dados preparados para o cenário de teste.
Finalizando o cenário de teste...
Dados limpos após o cenário de teste.
Limpeza global após todos os testes.
Ambiente limpo.
2 Scenarios (2 passed)
7 Steps (7 passed)
0m1,653s
Process finished with exit code 0


comar80 / api-rest-trafego-tests
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Continuous Integration
teste CI #7
Re-run all jobs
Summary
Jobs
continuous-integration
deploy
Run details
Usage
Workflow file
Triggered via push 19 minutes ago
comar80 pushed 86628cf main
Status
Success
Total duration
2m 1s
Artifacts
–
ci.yml
on: push
continuous-integration 1m 37s
deploy 0s
Annotations
2 warnings
continuous-integration
The following actions uses node12 which is deprecated and will be forced to run on node16: actions/setup-java@v2. For more info: https://github.blog/changelog/2023-06-…
Show more
continuous-integration
The following actions use a deprecated Node.js version and will be forced to run on node20: actions/checkout@v3, actions/setup-java@v2. For more info: https://github.b…
Show more