1. Plano de Teste - PGE

1.1 Objetivo

Este plano visa garantir a qualidade do sistema de cadastro e consulta de inscrições protestadas, validando seus requisitos funcionais e de segurança, e se os endpoints da API responsáveis pelo cadastro e pela visualização de inscrições protestadas estão funcionando corretamente e em conformidade com os critérios de aceitação definidos nas histórias de usuário.

1.2 Escopo dos Testes:

1.2.1 Testes funcionais dos endpoints:

1.2.1.1 POST /login – Autenticação

1.2.1.2 POST /contribuintes - Cadastro de contribuintes

1.2.1.3 GET /inscrições/{cpf} - Consulta de inscrições

1.2.2 Validação dos campos obrigatórios

1.2.3 Validação de campos obrigatórios, CPF válido e único

1.2.4 Validar que o acesso aos dados seja restrito exclusivamente aos usuários autorizados, permitindo apenas a visualização e manipulação das informações pertencentes ao próprio contribuinte.

1.3 Critérios de Entrada:

1.3.1 API em funcionamento no endpoint (URL BASE: http://testeqa.pge.ce.gov.br)

1.3.2 Token JWT para realizar autenticação

1.3.3 Dados de acesso:

{
“usuário”: “admin”,
“senha”: “password”,
}

1.4 Critérios de saída:

1.4.1 Todos os cenários de teste validados e executados

1.4.2 Registros de execução e evidencias anexados

1.4.3 Repositório GitHub atualizado com artefatos

1.5 Ferramentas:

1.5.1 Postman

1.5.2 Cypress

1.5.3 Git + GitHub

1.5.4 Capturador de tela

Cenários de Teste

1.6 Cadastro de Contribuinte

1.6.1 **CT 01 - Cadastro com dados válidos**

Dado que o usuário está autenticado com token válido
Quando o usuário envia uma requisição POST para /contribuintes com todos os campos obrigatórios preenchidos corretamente
Então a resposta deve retornar status 201 Created
E o contribuinte deve ser cadastrado com sucesso

1.6.2 **CT 02 - Cadastro com CPF inválido**

Dado que o usuário está autenticado com token válido
Quando o usuário envia uma requisição POST para /contribuintes com um CPF inválido
Então a resposta deve retornar status 400 Bad Request
E deve conter mensagem indicando erro de CPF inválido

1.6.3 **CT 03 - Cadastro com CPF duplicado**

Dado que o usuário está autenticado com token válido
E o CPF já esta cadastrado no sistema
Quando o usuário envia uma requisição POST para /contribuintes com esse mesmo CPF
Então ele deve receber a resposta 400 bad request
E deve conter mensagem indicando duplicidade de CPF

1.6.4 **CT 04 - Cadastro com campos obrigatórios faltando**

Dado que o usuário está autenticado com token válido
Quando o usuário envia uma requisição POST para /contribuintes faltando algum campo obrigatório
Então a resposta deve retornar status 400 Bad Request
E deve conter mensagem informando que todos os campos obrigatórios devem ser preenchidos

1.6.5 **CT 05 - Cadastro sem token de autenticação**

Dado que o usuário não está autenticado
Quando o usuário envia uma requisição POST para /contribuintes com dados válidos
Então a resposta deve retornar status 401 Unauthorized
E a resposta deve conter uma mensagem de erro “Token inválido”

1.7 Visualização de Inscrições Protestadas

1.7.1 **CT 01 - Consulta de inscrições existentes para contribuinte autenticado**

Dado que o usuário está autenticado com token válido
Quando o usuário faz uma requisição GET para /inscricoes/{cpf} onde existem inscrições cadastradas
Então a resposta da API deve ser 200 ok
E a lista deve conter as informações: número da inscrição, descrição, valor, data da inscrição e data do prazo

1.7.2 **CT 02 - Consulta de inscrições inexistentes para contribuinte autenticado**

Dado que o usuário está autenticado com token válido
Quando o usuário faz uma requisição GET para /inscricoes/{cpf} onde não existem inscrições cadastradas
Então a resposta deve retornar status 404 Not Found
E deve retornar a mensagem de Erro “Nenhuma inscrição encontrada para este contribuinte”

1.7.3 **CT 03 - Consulta de inscrições sem autenticação**

Dado que o usuário não está autenticado

Quando o usuário faz uma requisição GET para /inscricoes/{cpf} sem token válido
Então a resposta deve retornar erro de autorização (401 Unauthorized ou similar)

1.8 Validações e Segurança

1.8.1 **CT 01 - Restrição de acesso às inscrições**
Dado que o usuário está autenticado com token válido para um CPF específico
Quando o usuário tenta acessar inscrições de outro CPF
Então o acesso deverá ser negado

1.9 Login

1.9.1 **CT 01 - Login com credenciais válidas**
Dado que informo o usuário "admin" e senha "password"
Quando envio a requisição POST para /login
Então devo receber o status code 200
E a resposta deve conter o campo "token"

1.9.2 **CT 02 - Login com credenciais inválidas**
Dado que informo o usuário "admin" e senha "senha_incorreta"
Quando envio a requisição POST para /login
Então devo receber o status code 401
E a mensagem de erro "Usuário ou senha inválidos"

1.10 Abordagem dos Testes

1.10.1 Testes funcionais serão realizados para validar o comportamento esperado dos endpoints de login, cadastro e consulta de inscrições.
1.10.2 Serão utilizados testes automatizados com Cypress para garantir a repetibilidade e cobertura dos cenários descritos.
1.10.3 Testes manuais podem ser realizados para exploração inicial e validação visual dos resultados.
1.10.4 Será verificada a conformidade dos status HTTP, estrutura e conteúdo das respostas.
1.10.5 Os testes serão executados no ambiente homologação com dados controlados para garantir a integridade dos testes.

1.11 Riscos e Restrições:

1.11.1 O ambiente de testes deve estar disponível e estável durante as execuções
1.11.2 O processo de autenticação deve funcionar corretamente para permitir o acesso aos endpoints protegidos.

1.12 Critérios de Suspensão e Retomada

1.12.1 Suspender os testes caso o ambiente ou endpoints estejam indisponíveis.
1.12.2 Retomar os testes assim que o ambiente estiver novamente estável

1.13 Critérios de Aceitação dos Testes

1.13.1 Todos os testes automatizados devem retornar os status HTTP esperados para cada cenário
1.13.2 A estrutura e o conteúdo das respostas devem estar em conformidade com os requisitos.
1.13.3 O processo de autenticação deve garantir acesso apenas aos usuários autorizados.
1.13.4 Todas as evidências dos testes devem ser registradas e armazenadas.

1.14 Planejamento e Execução

1.14.1 Execução dos testes será realizada após validação do ambiente.
1.14.2 Feedbacks serão documentados para acompanhamento contínuo da qualidade.
1.14.3 Os testes automatizados serão executados automaticamente na pipeline de integração contínua (CI), garantindo a validação dos endpoints antes de qualquer deploy em ambiente de produção.

1.15 Gestão de Evidências

1.15.1 Todas as execuções de testes automatizados terão logs gerados automaticamente.
1.15.2 Serão capturadas capturas de tela (screenshots) em caso de falhas para facilitar a análise.
1.15.3 Vídeos das execuções de testes quando possível serão gravados para demonstração da cobertura dos cenários.
1.15.4 Todos os artefatos (logs, screenshots, vídeos) serão armazenados em um repositório centralizado no GitHub.

1.16 Postman Collections

1.16.1 Será criada uma coleção no Postman contendo as requisições para os endpoints de login, cadastro e consulta.
1.16.2 As variáveis de ambiente do Postman estarão devidamente configuradas para facilitar a execução dos testes.
1.16.3 A collection estará documentada para permitir reutilização por outros membros da equipe.

1.17 Relatório de Melhorias

1.17.1 Durante a execução dos testes, podem ser identificadas oportunidades de melhoria na API, como validações mais rígidas, mensagens de erro padronizadas e melhorias no processo de autenticação e segurança.
1.17.2 Essas melhorias serão reportadas em um documento específico, separado do plano de testes, visando apoiar a evolução contínua da qualidade do sistema.