

PLANO DE TESTE DA API PROJETO TRIPLEX

Recife

2022

**NOME DOS INTEGRANTES NO README**

**Sumário**

Introdução

Visão geral

A API, Projeto Triplex, é uma API REST que garante a utilização de um software cujo o objetivo é cadastras obras e realizar verificações das mesma com intuito de servir como ferramenta para auxiliar a empresa na gestão de seus projetos. Nela conseguimos utilizar os principais verbos (POST,GET,DELETE E PUT) de diferentes modos mediante a necessidade de sua regra de negócio.

Objetivo

Testar as funcionalidades da API Projeto Triplex, com um o fluxo de requisições necessários para realizar a confirmação das informações presentes na documentação. Testar as rotas apresentadas construindo um fluxo de testes de maneira sequencial simulando caminhos lógicos dos seus usuários com a finalidade de confirmar seu pleno funcionamento.

Escopo

Este documento contém o plano de teste especificando os testes feitos na aplicação. Dentre eles:

* Testes em ambientes diferentes: Produção e Local;
* Criação de suíte de casos de testes para os testes que serão executados;
* Exposição da estratégia de testes e dos tipos de teses aplicados;
* Listagem dos testes que devem ser automatizados.

Terá como objetivo testar o funcionamento da API da Projeto Triplex com testes unitários e também integrados utilizando dos endpoints fornecidos na documentação (<http://localhost:8080/swagger-ui/index.html#/>).

Os endpoints que serão testados:

* **Login - (/usuario-controller)**
  + Realização do Login;
  + Captura de Token.
* **Controle de Obras - (/obra-controller)**
  + Listagem e busca de obras por usuário no sistema, cadastro de obras, edição ou exclusão de obras no sistema;
* **Controle de Obras por localização - (api/obralocal)**
  + Listagem de localização das obras, Busca de localização de uma obra específica, Cadastro, edição e exclusão de obras realizadas pelo usuário.
* **Controle de Inspeções das Obras - (api/obrainspecoes)**
  + Listagem e busca de inspeções das obras por usuário no sistema, cadastro das inspeções, edição ou exclusão de inspeções no sistema;
* **Controle de detalhes Técnicos por obras - (/api/detalhesobra)**
  + Listagem e busca dos detalhes técnicos por obra por usuário no sistema, cadastro dos detalhes técnicos por obra, edição ou exclusão dos detalhes técnicos por obra no sistema;
* **Documentação - (/api/upload)**
  + Importação e exportação de documentações e relatórios.

**Suíte de casos de teste**

| **CT-1** | **Validação de Cadastro de Usuário** |
| --- | --- |

|  |  |
| --- | --- |
| **Objetivo** | Garantir que o usuário consiga se cadastrar no Projeto Triplex |
| **Pré-requisitos** | 1. Acessar servidor local (http://localhost:8080) |
| **Procedimento**  **(Positivo)** | 1. Execute um servidor local para seus testes e utilize sua BaseURL 2. Utilizando o Postman, crie uma requisição com método POST e endpoint /api/usuarios 3. No body declare como raw e substitua o formato Text por JSON 4. Ainda no body, declare o objeto com base no modelo da documentação: 5. (http://localhost:8080) → Usuários – Cadastrar usuário 6. Clique em Send e conclua o cadastro |
| **Procedimento**  **(Negativo)** | 1. Execute um servidor local para seus testes e utilize sua BaseURL 2. Utilizando o Postman, crie uma requisição com método POST e endpoint /api/usuarios 3. No body declare como raw e substitua o formato Text por JSON 4. Ainda no body, declare o objeto de formas diferentes:    1. Usuário já cadastrado:   Utilize os mesmos dados utilizados no procedimento (Positivo)   * 1. Dados vazios: Senha ou Login.   2. Dados Vazios: Ambos os campos senha e login vazios.  1. Clique em Send. |
| **Resultados Esperados** | * **Positivo:** Cadastro concluído com sucesso, adição de ID do usuário, login:nome do usuário, senha: senha do usuário, criação de token e status de adim.   **Negativo:** Mensagem de erro por falta de informações ou invalidez |
| **Pós-Condição** | O usuário conseguir registrar-se na API Projeto Triplex |

| **CT-2** | **Validação de Login** |
| --- | --- |

|  |  |
| --- | --- |
| **Objetivo** | Garantir que o usuário consiga logar em sua conta |
| **Pré-requisitos** | 1. Servidor local 2. Cadastro de usuário |
| **Procedimento**  **(Positivo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **POST** e endpoint **/** /api/usuarios/auth 3. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 4. Ainda no **body**, declare o login e senha do respectivo usuário 5. Clique em **Send** e execute o login. |
| **Procedimento**  **(Negativo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **POST** e endpoint **/** /api/usuarios/auth 3. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 4. Ainda no **body**, declare o objeto de formas diferentes:    1. **Dados incorretos:** 5. Faça dois testes: login não registrado e senha inválida   **II. Dados vazios:**   1. Faça três testes: login vazio, outro com a senha vazia e com os ambos os dados vazios 2. Clique em **Send** |
| **Resultados Esperados** | * **Positivo**: O usuário consiga entrar na conta criada * **Negativo**: Mensagem de erro por falta de dados ou dados incorretos. |
| **Pós-Condição** | Utilizando um login válido, ter sucesso em seu response |

| **CT-3** | **Validação de Cadastro de Obras** |
| --- | --- |

|  |  |
| --- | --- |
| **Objetivo** | Garantir que o usuário consiga adicionar uma obra |
| **Pré-requisitos** | 1. Servidor local 2. Cadastro de usuário 3. Login |
| **Procedimento**  **(Positivo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **POST** e endpoint /api/obras 3. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 4. Ainda no **body**, as informações requisitadas na documentação. 5. Clique em **Send** e execute a busca. |
| **Procedimento**  **(Negativo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **POST** e endpoint /api/obras 3. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 4. Ainda no **body**, declare o objeto de formas diferentes:    1. **Dados incorretos:**   Faça dois testes: e-mail não registrado e senha inválida   * 1. **Dados vazios:**   Faça três testes: login vazio, outro com a senha vazia e com os ambos os dados vazios.   * 1. **Usuário sem está logado:**   **Faça teste:** Cadastrando uma obra sem o token funcionando.   1. Clique em **Send** |
| **Resultados Esperados** | * **Positivo**: Obra Cadastrada com sucesso status code: 201. * **Negativo**: Resposta obras não cadastradas.  1. **Dados Incorretos:** Status code**:** 400 “Bad Request” 2. **Dados Vazios:** Status code**:** 400 “Bad Request” 3. **Usuários Deslogado:** Status code: 403 “Forbidden” |
| **Pós-Condição** | Utilizando um login válido, ter sucesso ao cadastrar uma obra. |

| **CT-4** | **Buscar Obras Cadastradas** |
| --- | --- |

|  |  |
| --- | --- |
| **Objetivo** | Garantir que o usuário visualize todas as obras cadastradas |
| **Pré-requisitos** | 1. Servidor local 2. Cadastro de usuário 3. Login 4. Cadastrar pelo menos uma obra. |
| **Procedimento**  **(Positivo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **GET** e endpoint /api/obras 3. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 4. Em Authorization informar token de usuário. 5. Clique em **Send** e execute a busca. |
| **Resultados Esperados** | * **Positivo**: Visualizar obras cadastradas com sucesso status code: 200 “OK”. |

| **CT-5** | **Visualização da Obra Por ID** |
| --- | --- |

|  |  |
| --- | --- |
| **Objetivo** | Garantir que o usuário visualizar todas as obras cadastradas por apenas um ID específico |
| **Pré-requisitos** | 1. Servidor local 2. Cadastro de usuário 3. Login 4. Id do usuário 5. Cadastrar pelo menos uma obra. |
| **Procedimento**  **(Positivo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **GET** e endpoint /api/obras{id do usuário} 3. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 4. Em Authorization informar token de usuário. 5. Id da obra. 6. Clique em **Send** e execute a busca. |
| **Procedimento**  **(Negativo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **GET** e endpoint /api/obras{id do usuário inesistente} 3. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 4. Em Authorization informar token de usuário.   Faça dois testes ID não registrado.   1. Clique em **Send** |
| **Resultados Esperados** | * **Positivo**: Visualizar obras cadastradas com sucesso status code:200 “OK”. * **Negativo**: status code: 400 e a mensagem:   "errors": [  "Obra não encontrada. Por favor, verifique os campos obrigatorios e tente novamente. "  ] |

| **CT-6** | **Atualização da Obra Por ID da Obra** |
| --- | --- |

|  |  |
| --- | --- |
| **Objetivo** | Garantir que um usuário possa alterar uma obra específica por ID da obra. |
| **Pré-requisitos** | 1. Servidor local 2. Cadastro de usuário 3. Login 4. Cadastrar a uma obra específica. 5. Id da obra |
| **Procedimento**  **(Positivo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **PUT** e endpoint /api/obras{id da obra} 3. Em Authorization informar token de usuário. 4. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 5. Realizar alterações desejadas. 6. Clique em **Send** e execute as alterações. |
| **Procedimento**  **(Negativo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **PUT** e endpoint /api/obras{id da obra} 3. Em Authorization informar token de usuário. 4. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 5. Realizar alterações desejadas: 6. Enviar alterações vazias. 7. Informar alterações erradas. 8. Alterar cadastro sem ID. 9. ID inválido. 10. Clique em **Send** e execute as alterações. |
| **Resultados Esperados** | * **Positivo**: Atualizar obras cadastradas com sucesso status code:204 “No Content”. * **Negativo**:  1. Erro status code: 403 “Forbidden” 2. Erro status code:400 Bad Request 3. Erro status code:400 Bad Request 4. Erro status code: 405 “Method Not Allowed” |

| **CT-7** | **Deletar Obra Por ID da Obra** |
| --- | --- |

|  |  |
| --- | --- |
| **Objetivo** | Garantir que um usuário possa alterar uma obra específica por ID da obra. |
| **Pré-requisitos** | 1. Servidor local 2. Cadastro de usuário 3. Login 4. Cadastrar a uma obra específica. 5. Id da obra |
| **Procedimento**  **(Positivo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **DELETE** e endpoint /api/obras{id da obra} 3. Em Authorization informar token de usuário. 4. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 5. Realizar exclusões desejadas. 6. Clique em **Send** . |
| **Procedimento**  **(Negativo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **DELETE** e endpoint /api/obras{id da obra} 3. Em Authorization informar token de usuário. 4. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 5. Realizar alterações desejadas: 6. ID inexistente. 7. excluir sem ID. 8. Clique em **Send** e execute as alterações. |
| **Resultados Esperados** | * **Positivo**: Visualizar obras cadastradas com sucesso status code:204 “No Content”. * **Negativo**:  1. Erro status code: 400 “Bad Request” 2. Erro status code: 405Method Not Allowed |

| **CT-8** | **Validação de Cadastro de Inspeções de uma Obra** |
| --- | --- |

|  |  |
| --- | --- |
| **Objetivo** | Garantir que o usuário consiga adicionar uma inspeção em determinada obra. |
| **Pré-requisitos** | 1. Servidor local 2. Cadastro de usuário 3. Login 4. Cadastrar a uma obra específica. 5. Id da obra |
| **Procedimento**  **(Positivo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **POST** e endpoint /api/obrainspecoes 3. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 4. Ainda no **body**, as informações requisitadas na documentação. 5. Clique em **Send** para adicionar uma inspeção na obra determinada. |
| **Procedimento**  **(Negativo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **POST** e endpoint /api/obrainspecoes 3. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 4. Ainda no **body**, declare o objeto de formas diferentes:    1. Id de obra inválido    2. Id de obra vazio 5. Clique em **Send** |
| **Resultados Esperados** | * **Positivo**: Obra Cadastrada com sucesso status code: 201 “Created”. * **Negativo**: Resposta obras não cadastradas.  1. Status code: 500”Internal Server Error” 2. Status code**:** 400 “Bad Request” |
| **Pós-Condição** | Inspeção de obra adicionada em Obra específica |

| **CT-9** | **Buscar Inspeções de Obras Cadastradas** |
| --- | --- |

|  |  |
| --- | --- |
| **Objetivo** | Garantir que o usuário visualize todas as inspeções de obras cadastradas |
| **Pré-requisitos** | 1. Servidor local 2. Cadastro de usuário 3. Login 4. Cadastrar pelo menos uma obra. |
| **Procedimento**  **(Positivo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **GET** e endpoint /api/obras 3. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 4. Em Authorization informar token de usuário. 5. Clique em **Send** e execute a busca. |
| **Resultados Esperados** | * **Positivo**: Visualizar inspeções cadastradas com sucesso status code: 200 “OK”. |

| **CT-10** | **Visualização de Inspeção de Obra Por ID** |
| --- | --- |

|  |  |
| --- | --- |
| **Objetivo** | Garantir que o usuário visualizar todas as obras cadastradas por apenas um ID específico |
| **Pré-requisitos** | 1. Servidor local 2. Cadastro de usuário 3. Login 4. Cadastrar pelo menos uma obra. 5. Cadastrar pelo menos uma Inspeção de obra. 6. ID da obra desejada |
| **Procedimento**  **(Positivo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **GET** e endpoint /api/obras{id do usuário} 3. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 4. Em Authorization informar token de usuário. 5. Informar o Id da obra. 6. Clique em **Send** e execute a busca. |
| **Procedimento**  **(Negativo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **GET** e endpoint /api/obras{id do usuário inesistente} 3. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 4. Em Authorization informar token de usuário.   ID não registrado.   1. Clique em **Send** |
| **Resultados Esperados** | * **Positivo**: Visualizar obras cadastradas com sucesso status code:200 “OK”. * **Negativo**: status code: 400 e a mensagem:   {  "errors": [  "Obra inspecao não encontrado. Por favor, verifique os campos obrigatorios e tente novamente. "  ]  } |

| **CT-11** | **Atualização das Inspeções de Obra Por ID da Obra** |
| --- | --- |

|  |  |
| --- | --- |
| **Objetivo** | Garantir que um usuário possa alterar uma obra específica por ID da obra. |
| **Pré-requisitos** | 1. Servidor local 2. Cadastro de usuário 3. Login 4. Cadastrar a uma obra específica. 5. Id da obra 6. Cadastrar uma inspeção |
| **Procedimento**  **(Positivo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **PUT** e endpoint /api/obras{id da obra} 3. Em Authorization informar token de usuário. 4. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 5. Realizar alterações desejadas. 6. Clique em **Send** e execute as alterações. |
| **Procedimento**  **(Negativo)** | 1. Execute um servidor local para seus testes e utilize sua **BaseURL** 2. Utilizando o Postman, crie uma requisição com método **PUT** e endpoint /api/obras{id da obra} 3. Em Authorization informar token de usuário. 4. No **body** declare como **raw** e substitua o formato **Text** por **JSON** 5. Realizar alterações desejadas: 6. Enviar alterações vazias. 7. Informar alterações erradas. 8. Alterar cadastro sem ID. 9. ID inválido. 10. Clique em **Send** e execute as alterações. |
| **Resultados Esperados** | * **Positivo**: Visualizar in obras cadastradas com sucesso status code:204 “No Content”. * **Negativo**:  1. Erro status code: 403 “Forbidden” 2. Erro status code:400 Bad Request 3. Erro status code:400 Bad Request 4. Erro status code: 405 “Method Not Allowed” |