Documento Técnico – Cenários de Teste da API de Cadastro de Usuário

Introdução

Este documento descreve os cenários de teste implementados para a funcionalidade de **cadastro de usuários** via API. Os testes foram escritos em **Cypress**, com foco na verificação de comportamentos esperados e na robustez da camada de validação da API.

Objetivo

O objetivo dos testes é garantir que a API de cadastro:

- Aceita corretamente dados válidos
- Rejeita tentativas de duplicidade
- Valida corretamente os tipos de dados recebidos

A escolha dos cenários foi feita com base em critérios de impacto e cobertura funcional mínima necessária para garantir o comportamento esperado da funcionalidade.

Estrutura dos Testes

Os testes estão localizados em:

cypress/e2e/api/cadastro.api.cy.js

Utilizam arquivos auxiliares para modularização:

- `support/utils/usuarios.js` → geração de dados válidos
- `support/api/usuarios.js` → chamada à API
- `fixtures/api/usuarios_invalidos.js` → corpo de requisição inválido e mensagens esperadas

Cenários de Teste

CT01 – Cadastro de Usuário com Sucesso

Objetivo: Garantir que um novo usuário seja cadastrado com sucesso quando os dados enviados estão corretos.

Justificativa: Este é o fluxo principal da funcionalidade e precisa ser validado como prioritário. A ausência deste teste comprometeria a verificação do objetivo principal da rota.

Dados enviados:

- `nome`: string
- `email`: string válida
- `password`: string
- `administrador`: 'true' ou 'false'

Validações:

- Status `201 Created`
- Presença da mensagem `"Cadastro realizado com sucesso"`
- Campo `_id` gerado na resposta

CT02 – Tentativa de Cadastro com Email Duplicado

Objetivo: Validar que o sistema impede o cadastro de usuários com e-mail já existente.

Justificativa: Este cenário verifica uma regra de negócio fundamental de unicidade de e-mail, essencial para integridade de dados e segurança de login.

Validações:

- Status `400 Bad Request`
- Mensagem: `"Este email já está sendo usado"`

CT03 – Cadastro com Dados de Tipos Inválidos

Objetivo: Verificar se a API realiza a validação correta dos tipos de dados enviados no corpo da requisição.

Justificativa: A validação de tipos protege a aplicação contra dados malformados, falhas internas e possíveis vulnerabilidades. Este teste cobre múltiplos erros de forma combinada.

Dados inválidos enviados:


```
{
  "nome": 125,
  "email": "teste@g",
  "password": 145,
  "administrador": null
}
```


Mensagens esperadas:


```
{
  "nome": "nome deve ser uma string",
  "email": "email deve ser um email válido",
  "password": "password deve ser uma string",
  "administrador": "administrador deve ser 'true' ou 'false'"
}
```


Validações:

- Status `400 Bad Request`
- Presença e correspondência das mensagens esperadas por campo

---

Conclusão

Os três cenários selecionados formam a base mínima essencial para validar a integridade da rota de cadastro. Eles cobrem:

- O caminho ideal (cadastro válido)
- A prevenção de duplicidade
- A robustez contra dados malformados

Essa abordagem garante que a API esteja segura, consistente e preparada para receber dados externos de forma confiável.