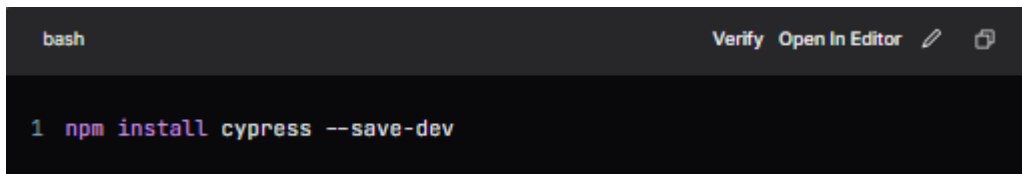


Antes de tudo baixem a dependência do cypress

A screenshot of a terminal window with a dark background. The top bar shows 'bash' on the left and 'Verify Open In Editor' with icons on the right. The main area contains a single line of code: '1 npm install cypress --save-dev'.

Testes de Controles Financeiros RPV

Introdução

Este documento descreve os testes de controles financeiros RPV, realizados utilizando o framework de testes Cypress. Os testes são divididos em diferentes cenários, cada um verificando uma funcionalidade específica do sistema.

Cenários de Teste

1. Cadastrar Entradas

- Descrição: Verifica se é possível cadastrar uma entrada no sistema.
- Pré-requisitos: Nenhum.
- Passos:

1. Visitar a página de controles financeiros.
 2. Clicar no botão de transação.
 3. Preencher os campos de descrição, valor e data.
 4. Clicar no botão de salvar.
- Resultado Esperado: A entrada deve ser cadastrada com sucesso e exibida na tabela de transações.

2. Remover Lançamento

- Descrição: Verifica se é possível remover um lançamento do sistema.
- Pré-requisitos: Ter uma entrada cadastrada.
- Passos:
 1. Visitar a página de controles financeiros.
 2. Clicar no botão de transação.
 3. Preencher os campos de descrição, valor e data.
 4. Clicar no botão de salvar.
 5. Clicar no botão de remover da entrada cadastrada.
- Resultado Esperado: A entrada deve ser removida com sucesso e não exibida na tabela de transações.

3. Remover Entrada e Saída

- Descrição: Verifica se é possível remover uma entrada e uma saída do sistema.
- Pré-requisitos: Ter uma entrada e uma saída cadastradas.
- Passos:
 1. Visitar a página de controles financeiros.
 2. Clicar no botão de transação.
 3. Preencher os campos de descrição, valor e data para a entrada.
 4. Clicar no botão de salvar.
 5. Clicar no botão de transação.

6. Preencher os campos de descrição, valor e data para a saída.
 7. Clicar no botão de salvar.
 8. Clicar no botão de remover da saída cadastrada.
- Resultado Esperado: A saída deve ser removida com sucesso e não exibida na tabela de transações.

Executando os Testes

Para executar os testes, utilize os seguintes comandos:

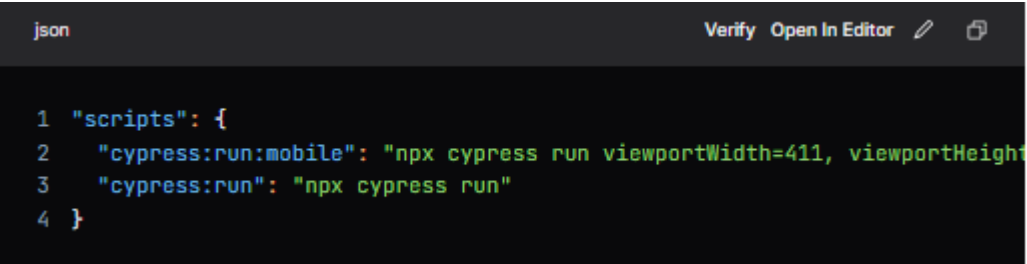
- `npm run cypress:run`: Executa os testes em resolução padrão.
- `npm run cypress:run:mobile`: Executa os testes em resolução móvel (411x823).

Observação: Ao executar o comando `npm run cypress:run:mobile`, os testes serão executados com uma resolução de 411x823, simulando um dispositivo móvel.

Já o comando `npm run cypress:run` executa os testes em resolução padrão.

Configuração

A configuração para as resoluções está definida no arquivo `package.json`. O comando `cypress:run:mobile` utiliza as flags `viewportWidth` e `viewportHeight` para definir a resolução móvel.



```
1 "scripts": {  
2   "cypress:run:mobile": "npx cypress run viewportWidth=411, viewportHeight=823",  
3   "cypress:run": "npx cypress run"  
4 }
```

Para executar os testes, você precisará ter instalado o framework de testes Cypress e ter configurado o ambiente de teste. Você pode executar os testes utilizando o comando:

Com resolução normal:

```
bash Verify Open In Editor 1 npm run cypress:run
```




Com resolução mobile:

```
bash Verify Open In Editor 1 npm run cypress:run:mobile
```

e:

Código dos Testes

O código dos testes está disponível no arquivo `rpv-finance-controls.spec.js` e é apresentado abaixo:

```
javascript Verify Open In Editor   

1  /// <reference types='cypress' />
2
3  describe('RPV Finance Controls', () => {
4    //hooks
5    //trechos que executam antes e depois dos testes
6    //before → antes de todos os testes
7    //beforeEach → antes de cada teste
8    //after → depois de todos os testes
9    //afterEach → depois de cada teste
10
11    beforeEach(() => {
12      cy.visit('https://dev-finance.netlify.app/');
13    })
14
15    // it('cadastrar entradas', () => {
16    //   // fluxo manual enter
17    //   // mapear os elementos que vamos interagir
18    //   // descrever as interações com cypress
19    //   // adicionar as asserções que vamos precisar
20    //   // cy.get → mapear um elemento
21
22    //   cy.get('#data-table tbody tr').should('have.length',0);
23
24    //   cy.visit('https://dev-finance.netlify.app/');
25
26    //   cy.get('#transaction .button').click(); // id + class
27    //   cy.get('#description').type('Puteiro de Luxo RPV');
28    //   cy.get('[name=amount]').type('3000000'); // atributo
29    //   cy.get('#date').type('2024-07-17'); // atributo
30    //   cy.get('button').contains('Salvar').click(); // tipo
31
32    //   cy.get('#data-table tbody tr').should('have.length',1);
33    // })
34    // it.only('remover lançamento', () => {
35
36    //   cy.get('#data-table tbody tr').should('have.length', 0);
37    //   cy.get('#transaction .button').click(); // id + class
38    //   cy
```