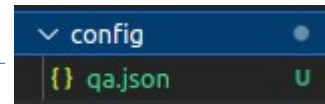


Ferramentas utilizadas: VScode - Cypress

1. Nenhum dado chumbado no script
2. Todas as variáveis são puxadas do arquivo config



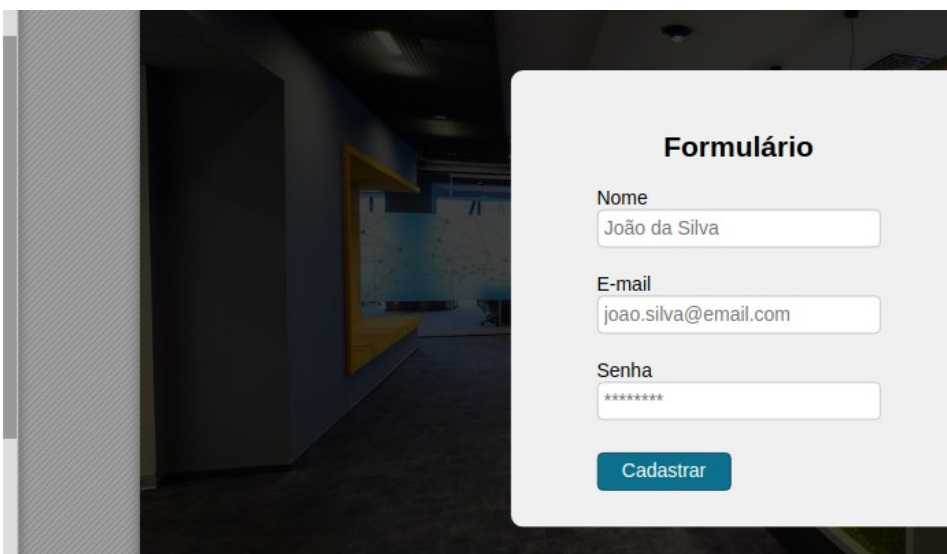
```
// Como o cy.visit pode conter dados sensíveis a melhor forma é importar os dados de um arquivo config.json
// beforeEach será o primeiro comando a ser executado em todos os "it"
beforeEach(() => cy.visit(`${Cypress.env('baseUrl')}`));
beforeEach(() => cy.VerificarCampos()); // Verifica se os campos estão visíveis e vazios
afterEach(() => cy.screenshot()); // Captura de Tela do caso de teste
```

3. Verifica se os campos estão visíveis e limpos antes de todos os testes. Por isso o uso do “beforeEach”

▼ BEFORE EACH (2)

```
1 get      #name
2 -assert
expected <input#name> to
be empty
3 get      #email
4 -assert
expected <input#email> to
be empty
5 get      #password
6 -assert
expected <input#password>
to be empty
7 get      #register
8 -assert
expected
<button#register> to be
visible
```

▼ TEST BODY



4. Preenche os campos e faz a verificação do erro

```
it('Realizando Cadastro com nome errado', () => { //Define o caso de teste
  cy.get('#name').type(Cypress.env('user-erro')); // Busca o campo e importa a variável do arquivo config/qa
  cy.get('#email').type(Cypress.env('email2'));
  cy.get('#password').type(Cypress.env('pass2'));
  cy.get('#register').click(); // Executa o click
  cy.contains('Por favor, insira um nome completo.')
    .should('be.visible'); // Validação do campo
});
```

TEST BODY

1	get	#name
2	- type	João
3	get	#email
4	- type	joakleber98908@teste.com
5	get	#password
6	- type	Teste@123
7	get	#register
8	- click	
9	contains	Por favor, insira um nome...
10	- assert	expected <p.error> to be visible

5. Faz verificação do campo e-mail

```
it.only('Realizando Cadastro com email errado', () => {
  cy.get('#name').type(Cypress.env('user2'));
  cy.get('#email').type(Cypress.env('email-erro'));
  cy.get('#password').type(Cypress.env('pass2'));
  cy.get('#register').click();
  cy.contains('Por favor, insira um e-mail válido.')
    .should('be.visible');
});
```

1	get	#name
2	- type	João Kleber
3	get	#email
4	- type	joat teste.com
5	get	#password
6	- type	Teste@123
7	get	#register
8	- click	
9	contains	Por favor, insira um e-ma...
10	- assert	expected <p.error> to be visible

6. Faz a verificação do campo senha

```
it('Realizando Cadastro com 1 caracter na senha', () => {
  cy.get('#name').type(Cypress.env('user2'));
  cy.get('#email').type(Cypress.env('email2'));
  cy.get('#password').type(Cypress.env('pass-erro'));
  cy.get('#register').click();
  cy.contains('A senha deve conter ao menos 8 caracteres.')
    .should('be.visible');
});
```

```

▼ TEST BODY
1  get      #name
2  - type   João Kleber
3  get      #email
4  - type   joaokleber98908@teste.com
5  get      #password
6  - type   T
7  get      #register
8  - click
9  contains
A senha deve conter ao me...
✚ - assert
expected <p.error> to be
visible

```

Formulário

Nome
João Kleber

E-mail
joaokleber98908@teste.com

Senha
•

A senha deve conter ao menos 8 caracteres.

Cadastrar

7. Faz a verificação de todos os campos errados

```

it.only('Realizando Cadastro com todos campos errados', () => {
  cy.get('#name').type(Cypress.env('user-erro'));
  cy.get('#email').type(Cypress.env('email-erro'));
  cy.get('#password').type(Cypress.env('pass-erro'));
  cy.get('#register').click();
  cy.contains('Por favor, insira um nome completo.')
    .should('be.visible');
  cy.contains('Por favor, insira um e-mail válido.')
    .should('be.visible');
  cy.contains('A senha deve conter ao menos 8 caracteres.')
    .should('be.visible');
});

```

```

9  contains
Por favor, insira um nome...
10 - assert
expected <p.error> to be
visible
11 contains
Por favor, insira um e-ma...
12 - assert
expected <p.error> to be
visible
13 contains
A senha deve conter ao me...
✚ - assert
expected <p.error> to be
visible

```

Nome
João
Por favor, insira um nome completo.

E-mail
joaoteste.com
Por favor, insira um e-mail válido.

Senha
•
A senha deve conter ao menos 8 caracteres.

Cadastrar

8. Faz o cadastro corretamente e seleciona os elementos da tabela

```
it('Realizando Cadastro Corretamente', () => {
  cy.get('#name').type(Cypress.env('user2'));
  cy.get('#email').type(Cypress.env('email2'));
  cy.get('#password').type(Cypress.env('pass2'));
  cy.get('#register').click();
  cy.log('Novo usuário cadastro');
  cy.get('table')
    .should('be.visible')
    .contains('1').siblings(); // cy.get('table') seleciona o elemento ancora (tabela) e o siblings seleciona os elementos irmãos
});
```

TEST BODY

1	get	#name
2	- type	João Kleber
3	get	#email
4	- type	joakleber98908@te...
5	get	#password
6	- type	Teste@123
7	get	#register
8	- click	
9	log	Novo usuário cadastro
10	get	table
11	- assert	expected <table> to be visible
12	- contains	1
	- siblings	3

The screenshot shows a web application with a registration form and a table of registered users. The form has fields for Name, E-mail, and Password, and a 'Cadastrar' button. Below the form is a table titled 'Usuários cadastrados' with columns for Id, Nome, E-mail, and Ações. The table contains one row with Id 1, Nome João Kleber, E-mail joakleber98908@te..., and Ações Excluir.

Id	Nome	E-mail	Ações
1	João Kleber	joakleber98908@te...	Excluir

9. Faz o cadastro corretamente de dois usuários, Faz a exclusão do primeiro e verifica se o segundo usuário cadastrado permanece

```
it('Realizando Cadastro Corretamente e excluindo o usuário', () => {
  cy.get('#name').type(Cypress.env('user2'));
  cy.get('#email').type(Cypress.env('email2'));
  cy.get('#password').type(Cypress.env('pass2'));
  cy.get('#register').click();
  cy.log('Novo usuário cadastro');
  cy.get('#name').type(Cypress.env('user3'));
  cy.get('#email').type(Cypress.env('email3'));
  cy.get('#password').type(Cypress.env('pass3'));
  cy.get('#register').click();
  cy.log('Novo usuário cadastro');
  cy.get('table')
    .should('be.visible')
    .contains('1').siblings().eq(0).next(); // cy.get('table') pega o elemento ancora (tabela),
                                           // siblings pega os elementos irmãos, "eq" seleciona um campo e o
                                           // next seleciona o próximo
  cy.get('#removeUser1').click();
  cy.get('table')
    .contains('2').should('be.visible');
});
```

```

14 get #password
15 - type Teste@123
16 get #register
17 - click
18 log Novo usuário
   cadastro
19 get table
20 - assert expected <table>
   to be visible
21 - contains 1
22 - siblings 3
23 - eq 0
24 - next
25 get #removeUser1
26 - click
27 get table
28 - contains 2
- assert expected
  <td#userId2> to
  be visible

```



Formulário

Nome

E-mail

Senha

[Cadastrar](#)

Usuários cadastrados

Id	Nome	E-mail	Ações
2	John Dohn	johndohn89303@te...	Excluir

10. Arquivo de variáveis

```

> {} qa.json > ...
{
  "chromeWebSecurity": false,
  "env": {
    "baseUrl": "http://prova.stefanini-jgr.com.br/teste/qa/",
    "user-erro": "João",
    "email-erro": "joaoteste.com",
    "pass-erro": "T",
    "user2": "João Kleber",
    "email2": "joaokleber98908@teste.com",
    "pass2": "Teste@123",
    "user3": "John Dohn",
    "email3": "johndohn89303@testeprojects.com.br",
    "pass3": "Teste@123"
  }
}

```

11. Configuração de Ambientes

Aqui foi definido o caminho que será feito ao executar os testes...

Exemplo: Poderemos executar apenas 1 script em vários ambientes apenas passando o nome do arquivo de configuração

```
function getConfigurationByFile (file) {  
  //caminho da pasta onde estão presentes os arquivos JSON dos ambientes  
  const pathToConfigFile = path.resolve('config', `${file}.json`)  
  
  return fs.readJson(pathToConfigFile)  
}  
  
module.exports = (on, config) => {  
  //aceita um valor de "configFile" ou usa "qa" por padrão  
  const file = config.env.configFile || 'qa'  
  return getConfigurationByFile(file)  
}
```