🚀 Plano de Testes de API (com Cypress)

✅ Pré-requisitos

Instale o plugin de validação de schemas se quiser deixar mais avançado:

```
npm install --save-dev chai-json-schema
```

📋 Plano de Testes Detalhado

🔹 API 1 - GET All Products List

Objetivo: Validar que a lista de produtos é retornada com sucesso.

```
describe('GET - Lista de produtos', () => {
  it('deve retornar status 200 e produtos', () => {
    cy.request('/api/productsList').then((res) => {
      expect(res.status).to.eq(200);
      expect(res.body.products.length).to.be.greaterThan(0);
    });
  });
});
```

🔹 API 2 - POST To All Products List

Objetivo: Testar comportamento inadequado (método errado).

```
it('não deve permitir POST na rota de produtos', () => {
  cy.request({
    method: 'POST',
    url: '/api/productsList',
    failOnStatusCode: false
  }).then((res) => {
    expect(res.status).to.eq(405); // Método não permitido
  });
});
```

🔹 API 3 - GET All Brands List

```
it('deve retornar todas as marcas', () => {
  cy.request('/api/brandsList').then((res) => {
    expect(res.status).to.eq(200);
    expect(res.body.brands.length).to.be.greaterThan(0);
  });
});
```

🔹 API 5 - POST Search Product

```
it('deve retornar resultados ao buscar "jeans"', () => {
  cy.request('POST', '/api/searchProduct', { search_product: 'jeans' })
    .then((res) => {
      expect(res.status).to.eq(200);
      expect(res.body.products.length).to.be.greaterThan(0);
    });
});
```

🔻 API 6 - POST Search Product sem parâmetro

```
it('deve retornar erro se não enviar search_product', () => {
  cy.request({
    method: 'POST',
    url: '/api/searchProduct',
    body: {},
    failOnStatusCode: false
  }).then((res) => {
    expect(res.status).to.eq(400);
    expect(res.body.message).to.include('Missing search_product parameter');
  });
});
```

🔻 API 7 - POST Login com dados válidos

```
it('login deve funcionar com dados válidos', () => {
  cy.request('POST', '/api/verifyLogin', {
    email: 'usuario@teste.com',
    password: '123456'
  }).then((res) => {
    expect(res.status).to.eq(200);
    expect(res.body.message).to.eq('User exists!');
  });
});
```

🔻 API 10 - POST Login com dados inválidos

```
it('deve falhar ao logar com senha errada', () => {
  cy.request({
    method: 'POST',
    url: '/api/verifyLogin',
    body: {
      email: 'usuario@teste.com',
      password: 'senhaErrada'
    },
    failOnStatusCode: false
  }).then((res) => {
    expect(res.status).to.eq(404);
    expect(res.body.message).to.eq('User not found!');
  });
});
```

🔻 API 11 - POST Create/Register User

```
it('deve cadastrar novo usuário com sucesso', () => {
  const email = `lucas${Date.now()}@mail.com`;
  cy.request('POST', '/api/createAccount', {
    name: 'Lucas QA',
    email: email,
    password: '123456',
    title: 'Mr',
    birth_date: '10',
    birth_month: 'June',
    birth_year: '1990',
    firstname: 'Lucas',
    lastname: 'Testador',
    company: 'QA Ltda',
    address1: 'Rua Teste',
    address2: 'Ap 101',
    country: 'Brazil',
    zipcode: '12345',
    state: 'SP',
    city: 'Testópolis',
```

```
    mobile_number: '11999999999'
  }).then((res) => {
    expect(res.status).to.eq(200);
    expect(res.body.responseCode).to.eq(201);
  });
});
```

🔹 API 12 - DELETE para deletar usuário

Necessita `email` e `password` válidos

```
it('deve deletar conta do usuário', () => {
  cy.request('DELETE', '/api/deleteAccount', {
    email: 'usuario@teste.com',
    password: '123456'
  }).then((res) => {
    expect(res.status).to.eq(200);
    expect(res.body.message).to.eq('Account deleted!');
  });
});
```

🔹 API 14 - GET Detalhes do usuário por e-mail

```
it('deve retornar detalhes do usuário por e-mail', () => {
  cy.request('GET', '/api/getUserDetailByEmail?email=usuario@teste.com')
    .then((res) => {
      expect(res.status).to.eq(200);
      expect(res.body.user.email).to.eq('usuario@teste.com');
    });
});
```

□ Dicas Extras

- Use o comando Cypress `cy.request()` para todas essas APIs.
- Valide headers, tempo de resposta, e mensagens de erro.
- Crie testes encadeados: **cadastrar → logar → deletar usuário**.
- Teste também os casos de erro (sem e-mail, sem senha etc.).