

Teste de APIs com

# Cypress e ServeRest

# Agenda

- SOBRE
- FERRAMENTAS
- OBJETIVO
- ESTRUTURA
- REPORTS
- HANDS ON

# Sobre

- Maximiliano Alves da Cruz - 29 Anos
- Líder de Engenharia - CWI Software
- Streamer de games nas horas vagas

[github.com/maximilianoalves](https://github.com/maximilianoalves)

[www.linkedin.com/in/maximilianodacruz/](https://www.linkedin.com/in/maximilianodacruz/)



# Ferramentas e Vantagens



- velocidade
- curva de aprendizado
- início de projeto rápido
- infinitas possibilidades

# Ferramentas e Vantagens

## ServeRest



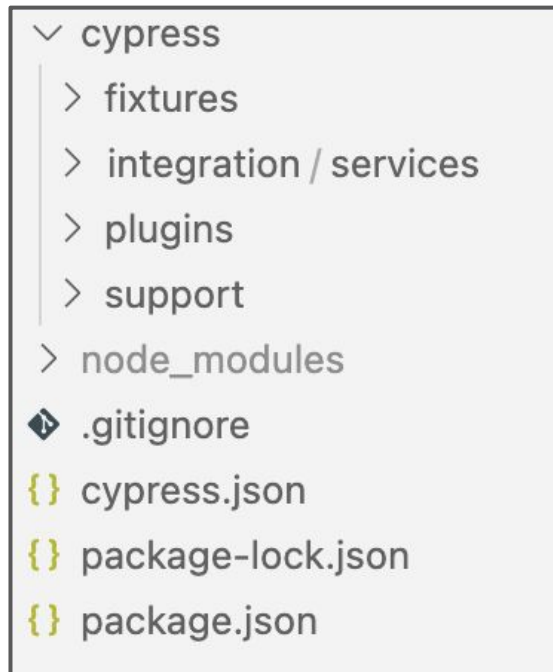
- serviços reais
- documentação
- localmente ou online
- complexidade em cenários (query params, auth e verbos)
- open source <3

# Objetivo



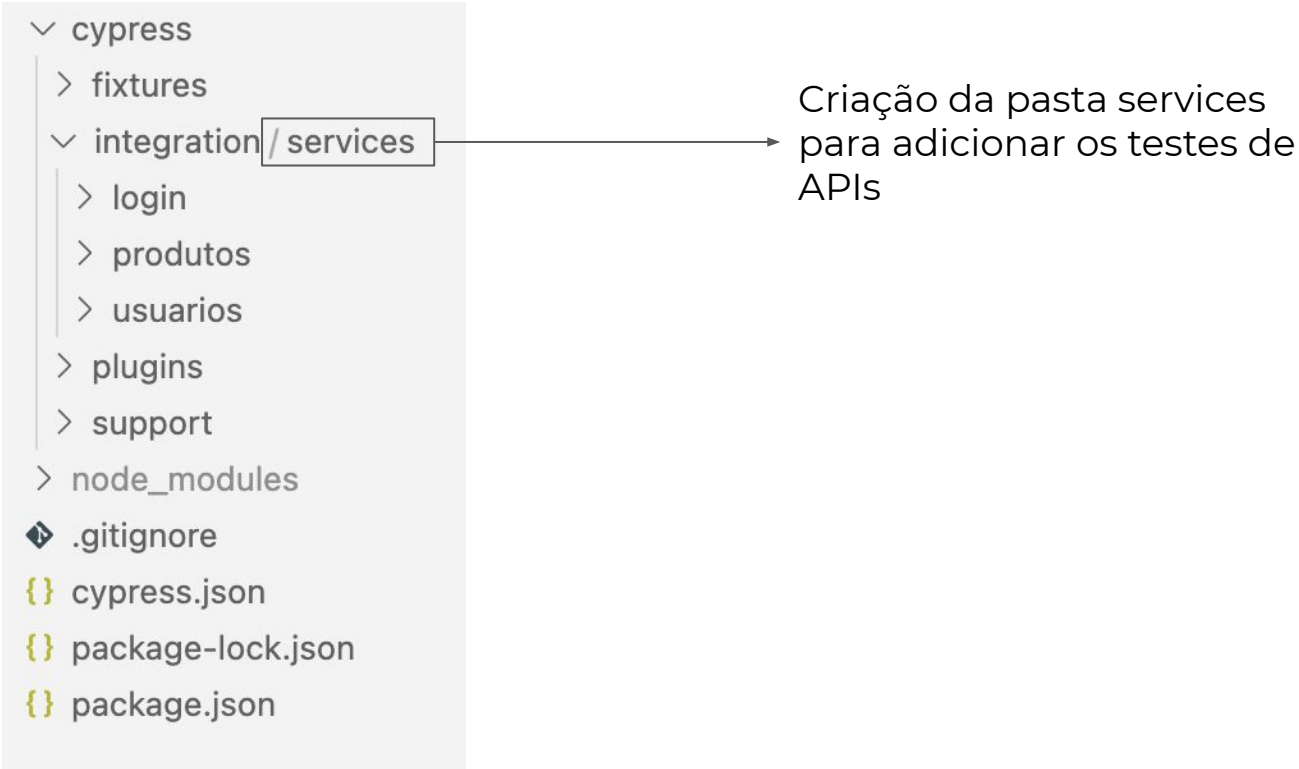
Cypress  
API Test

# Estrutura - Pastas



→ Inicialização da estrutura default do cypress

# Estrutura - Pastas



```

  ∨ cypress
    > fixtures
    ∨ integration / services
      > login
      > produtos
      > usuarios
      > plugins
      > support
    > node_modules
  ◆ .gitignore
  {} cypress.json
  {} package-lock.json
  {} package.json

```

The image shows a file explorer interface with a tree view of a project. The 'integration' folder is expanded, and the 'services' subfolder is highlighted with a rectangular box. A horizontal arrow points from this box to the right, towards a text block. The tree view includes folders like 'cypress', 'fixtures', 'integration', 'login', 'produtos', 'usuarios', 'plugins', 'support', and 'node\_modules'. It also lists files: '.gitignore', 'cypress.json', 'package-lock.json', and 'package.json'.

Criação da pasta services  
para adicionar os testes de  
APIs



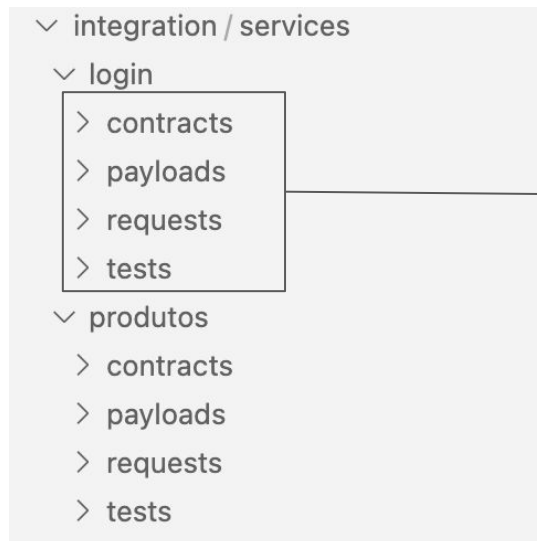
# Estrutura - Pastas



- ✓ cypress
  - > fixtures
  - ✓ integration / services
    - > login
    - > produtos
    - > usuarios
  - > plugins
  - > support
- > node\_modules
- ◆ .gitignore
- { } cypress.json
- { } package-lock.json
- { } package.json

Divisão dos testes path principal do endpoint ou funcionalidade

# Estrutura - Pastas



- **contracts:** Pasta aonde vamos colocar os arquivos Joi/js para validação do schema
- **payloads:** Pasta aonde vamos adicionar arquivos json para a criação do corpo da requisição
- **request:** Pasta para adicionar somente a requisição realizada para o path
- **tests:** Onde de fato adicionaremos nossos cenários de testes.

# Estrutura - Contratos

✓ contracts

JS produtos.contract.js



```
import Joi from 'joi'
```

```
const produtosSchema = Joi.object({  
  quantidade: Joi.number(),  
  produtos: Joi.array().items(  
    Joi.object({  
      nome: Joi.string(),  
      preco: Joi.number(),  
      descricao: Joi.string(),  
      quantidade: Joi.number(),  
      _id: Joi.string()  
    })  
  )  
})
```

```
export default produtosSchema;
```

# Estrutura - Payloads

▼ payloads

{ } add-produto.payload.json

{ } alterar-produto.payload.json



```
{  
  "nome": "Máquina de lavar Samsung",  
  "preco": 2200,  
  "descricao": "Lava e seca funcional.",  
  "quantidade": 5  
}
```

# Estrutura - Requests

▼ requests

JS deleteProdutos.request.js  
JS getProdutos.request.js  
JS postProdutos.request.js  
JS putProdutos.request.js



```
const payloadAddProduto = require('../payloads/add-produto.payload.json')

function adicionar(auth) {
  return cy.request({
    method: "POST",
    url: "produtos",
    headers: {
      accept: "application/json",
      Authorization: auth
    },
    failOnStatusCode: false,
    body: payloadAddProduto
  })
}

export {adicionar}
```

# Estrutura - Cenários

▼ tests

JS deleteProdutos.spec.js

JS getProdutos.spec.js

JS postProdutos.spec.js

JS putProdutos.spec.js

```
import * as GetProdutos from '../requests/getProdutos.request'  
import produtosSchema from '../contracts/produtos.contract'
```

```
describe('Get Produtos', () => {  
  it('Listar os produtos cadastrados', () => {  
    GetProdutos.listar().should((response) => {  
      expect(response.status).to.eq(200)  
      expect(response.body).to.be.not.null  
    })  
  });  
  
  it('Validar o contrato da listagem de produtos', () => {  
    GetProdutos.listar().should((response) => {  
      return produtosSchema.validateAsync(response.body)  
    })  
  });  
});
```

# Estrutura - Cenário complexo

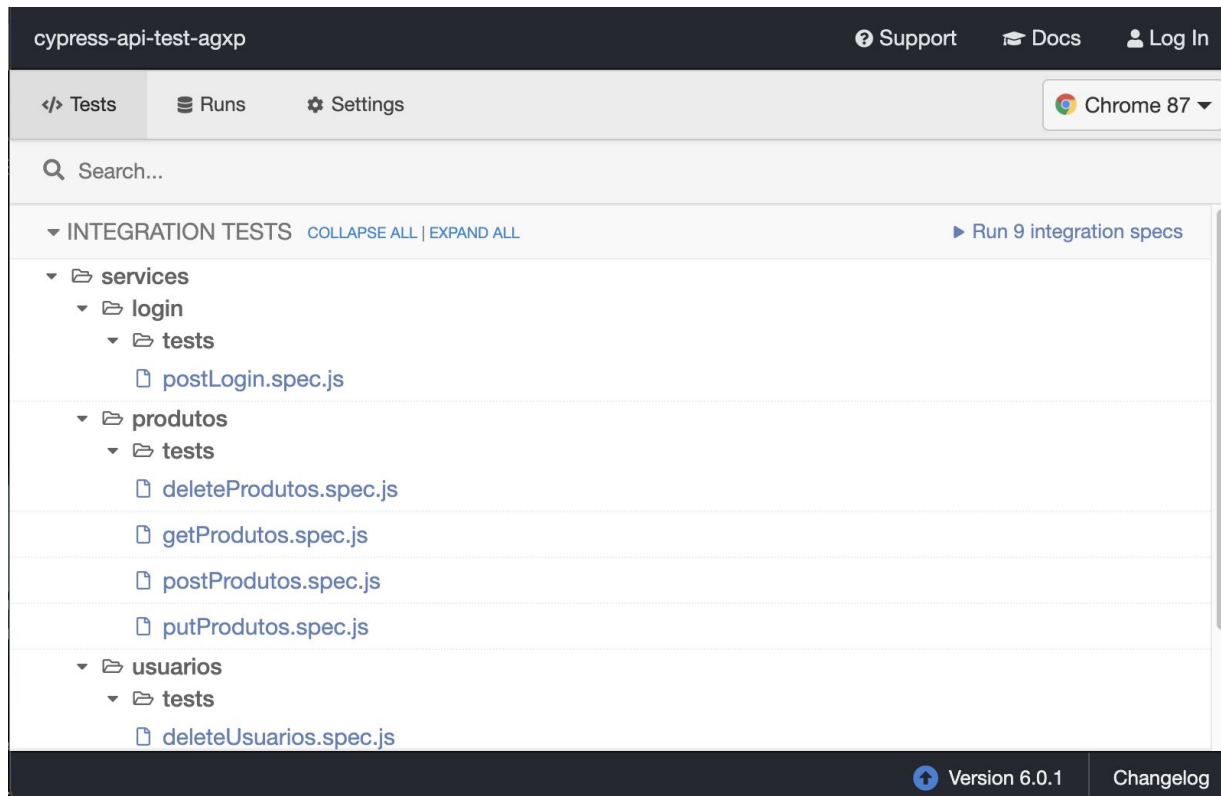
```
import * as PostProdutos from '../requests/postProdutos.request'
import * as PostLogin from '../login/requests/postLogin.request'
import * as DeleteProdutos from '../requests/deleteProdutos.request'

describe('Post Produtos', () => {
  it('Adicionar um produto', () => {
    PostLogin.autenticacao().should((response) => {
      expect(response.status).to.eq(200)
      PostProdutos.adicionar(response.body.authorization).should((resProduto) => {
        expect(resProduto.status).to.eq(201)
        expect(resProduto.body.message).to.eq('Cadastro realizado com sucesso')
        DeleteProdutos.apagar(resProduto.body._id, response.body.authorization).should((resDelete) => {
          expect(resDelete.status).to.eq(200)
          expect(resDelete.body.message).to.eq('Registro excluído com sucesso')
        })
      })
    })
  })
});
```



# Estrutura - Execução

npm run cypress:open





# Estrutura - Execução

npm run cypress:run

Running: services/login/tests/postLogin.spec.js

(1 of 9)

Post Login

✓ Fazer o login (250ms)

1 passing (267ms)

[mochawesome] Report JSON saved to /Users/Maximiliano/Documents/projetos/pessoal/cypress-api-test-agxp/mochawesome-report/json/mochawesome.json

## (Results)

Tests:	1
Passing:	1
Failing:	0
Pending:	0
Skipped:	0
Screenshots:	0
Video:	false
Duration:	0 seconds
Spec Ran:	services/login/tests/postLogin.spec.js

# Report

```
{  
  "baseUrl": "https://serverest.dev/",  
  "video": false,  
  "testFiles": "**/*.spec*",  
  "reporter": "mochawesome",  
  "reporterOptions": {  
    "reportDir": "mochawesome-report/json",  
    "overwrite": false,  
    "html": false,  
    "json": true  
  }  
}
```

Indicar qual report estamos usando.

Opções do report

# Report

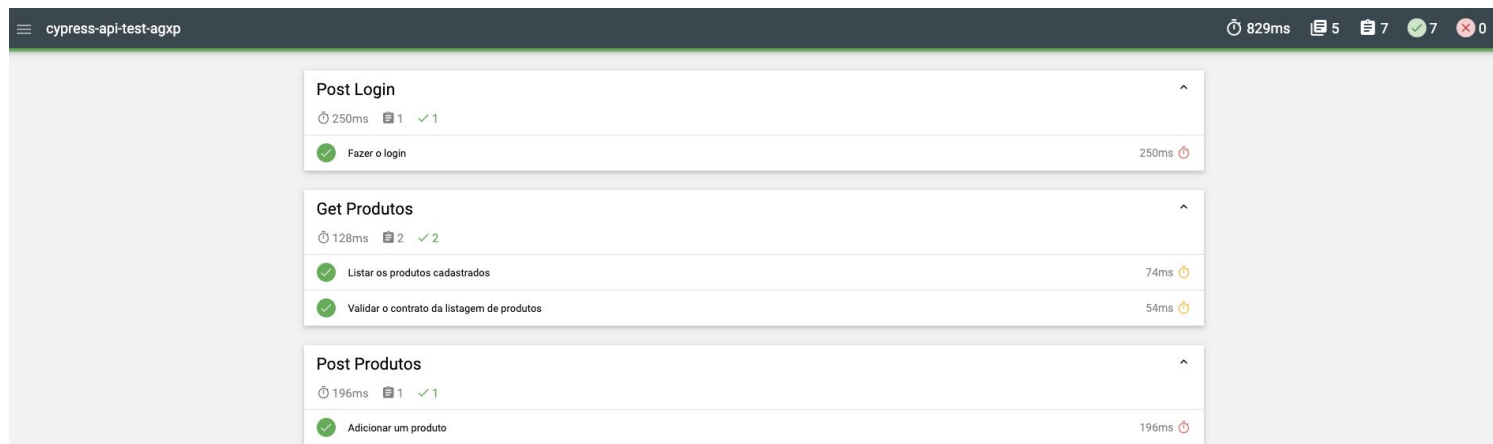
Após: npm run cypress:run

▼ mochawesome-report / json	●
{ } mochawesome_001.json	U
{ } mochawesome_002.json	U
{ } mochawesome_003.json	U
{ } mochawesome_004.json	U
{ } mochawesome_005.json	U
{ } mochawesome_006.json	U
{ } mochawesome_007.json	U
{ } mochawesome_008.json	U
{ } mochawesome.json	U

# Report

```
npx mochawesome-merge ./mochawesome-report/json/mochawesome*.json > mochawesome-report/report.json
```

```
npx marge mochawesome-report/report.json --autoOpen --showPending=false
```



**HANDS ON**

# **PERGUNTAS?**

**Obrigado <3**