Desafio de Automação QA-VR

Documentação base: Teste_QA_API-converted.pdf

Ferramentas utilizadas:

- Visual Studio Code Ambiente de desenvolvimento
- Cmder Prompt de comando Linux, no Windows

Linguagem de programação utilizada.

• ruby 2.5.1p57

Framworks utilizados

- Cucumber BDD
- Capybara O Capybara é um software de automação de testes baseado na web que simula cenários para histórias de usuários e automatiza testes de aplicativos da web para desenvolvimento de software orientado a comportamento. Está escrito na linguagem de programação Ruby.
- Rspec DLS Ruby, para testar código Ruby.
- SitePrism DSL (Domain Specific Language) criada para facilitar a criação de page objects para testes automatizados em Ruby, utilizando o Capybara.

•

Documentação utilizada:

BDD e2e – Desenvolvimento Orientado ao Comportamento e2e (end two end)

Para melhor compreensão dos envolvidos no projeto de desenvolvimento de um software, faz se necessário o uso de uma documentação de fácil manutenção, e eficiente que possa ser escrita, lida e de fácil entendimento. Quando a *user stories* é escrita e aprovada pelo cliente, a mesma servirá de auxilio aos desenvolvedores no desenvolvimento da feature, porém para a equipe de desenvolvimento ela não anula os requisitos, mas servirá como um agregador. (a não ser que a equipe prefira não utilizar requisitos). Os gerentes, team líder e cliente, servirá como documentação funcional do comportamento de uma determinada funcionalidade. E ao tester engineer, servirá como guia para o desenvolvimento dos testes automáticos.

README.MD

```
TESTANDO A API QA-VR

[Aprender]
GET => Retorna todos os registros
POST => Cadastra um novo registro
PUT => Altera os dados do registro de acordo com o ID
DELETE => Deleta o registro de acordo com o ID

[objetivo]
Conforme a descrição de:
    Criar uma funcionalidade para consultar os dados de um endereço a partir de um CEP
```

```
utilizar o verbo GET.
_ambiente: https://viacep.com.br/ws/01001000/json
[ParaExecutar]
Navegue até o diretório: qa_vr\tests\
Execute: bundle exec cucumber
[frameworks]
cucumber
_HTTParty
_rspec
[specifications]
_get.feature: Api apenas para testar o verbo GET
[specs]
_get.feature: Funcionalidade apenas para usar o verbo get.
OBS:
_api disponibiliza apenas o uso do verbo GET, impedindo assim que outros
verbos sejam testados.
[steps]
get.steps: Testes validados atráves do .rspec
OBS:
_para rodar os teste => Clonar o projeto e dar um bundle install na raíz.
através do terminal utilizar o comando: cucumber
[macro_steps]
_001 - Instalar as gems necessárias (GemFile) -> bundle install
     - Adicionando requires no spec_helper.rb
Results API
200 - Sucess
201 - Criado com sucesso
201 - Sem conteúdo
400 - Parâmetros inválidos
401 – Não autenticado
403 - Não autorizado
404 - Não encontrado
500 - Erro interno do servidor
[contato]
qa.eng.isaiasilva@gmail.com
```

Abaixo, segue os cenários ou casos de testes criados e automatizados.

Obs: Cenários estão em Inglês

```
Arquivo: consulta_feature

Feature: Query data from an address from the provided zipcode
    As part of the selection process for QA Tester's vacancy in VR,
    I want to perform a query of data of an address by the provided CEP

#TC_001 - 2.1 - Criar um cenário de sucesso na consulta, printando o código do IBGE do endereço no stdout.

Scenario: Search for an address and return the IBGE code
Given client to provide an API for testing
When I make an valid CEP query "01001-000"
Then the IBGE code must be displayed and correspond to the provided zip code

#TC_002 - 2.2 - Criar um cenário passando um CEP inválido
Scenario: Search for an address with an invalid zip code
Given client to provide an API for testing
When I make an invalid CEP query "05001-XYZ"
Then an error message should be displayed
```

```
Given("client to provide an API for testing") do
    @response = HTTParty.get 'https://viacep.com.br/ws/01001000/json'
    #puts "response code: #{@response.code}"
    #puts "response message: #{@response.message}"
    #puts "response headers: #{@response.headers}"
    #puts "response body: #{@response.body}"
end
When("I make an valid CEP query {string}") do |cep|
    @cep = cep
    @response = HTTParty.get "https://viacep.com.br/ws/#{@cep}/json"
end
Then("the IBGE code must be displayed and correspond to the provided zip
code") do
    @address = @response.parsed_response
    #puts @address
    #puts @cep
    #Válida se o CEP da API é igual ao CEP passado por parâmetro na
consulta realizada anteriormente
    expect(@address['cep']).to eq(@cep)
    puts "O CÓDIGO DO IBGE: #{@address['ibge']}, CORRESPONDE AO CEP:
#{@cep}!"
end
```

```
When("I make an invalid CEP query {string}") do |cep|
    @cep_invalido = cep
    @response = HTTParty.get
"https://viacep.com.br/ws/#{@cep_invalido}/json"
end

Then("an error message should be displayed") do
    puts "response code: #{@response.code}"
    expect(@response.code).to eq(400)
    puts "COM ESTE CEP: #{@cep_invalido} INVÁLIDO, A MENSAGEM ->
#{@response.message} É ESPERADA!"
end
```

Passos para executar o projeto.

- 1. Pré-Condições:
 - a. Variáveis de ambiente Java configuradas corretamente
 - b. Efetue download da versão mais recente do Ruby com Devkit.

https://github.com/oneclick/rubyinstaller2/releases/download/rubyinstaller-2.4.5-1/rubyinstaller-devkit-2.4.5-1-x64.exe

C. Coloque na pasta Windows o arquivo gecko e chromedriver correspondente a versão do seu navegador

https://github.com/mozilla/geckodriver/releases http://chromedriver.chromium.org/downloads

2. Descompacte o arquivo qa_vr_api.rar em seu diretório de desenvolvimento.

Ou efetue o clone do projeto no github

https://github.com/IsaiaSilva/qa vr api/

 Abra no cmder ou outro prompt de comando no diretório tests e execute o projeto de automação.

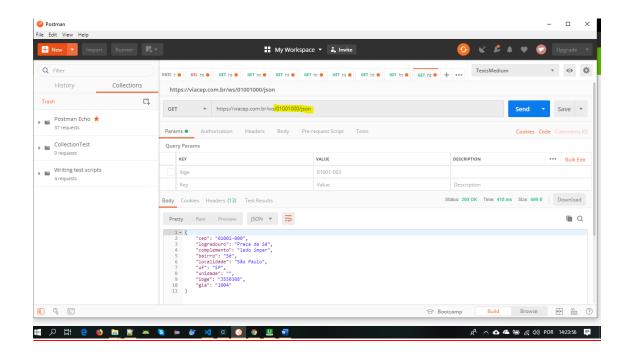
bundle install

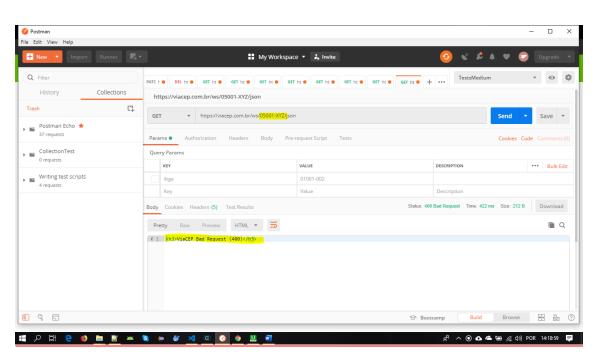
bundle exec cucumber

Atenção

No relatório -> O Screenshot não será exibido, pois não há interação com interface e a execução é realizada em modo headless (sem interação com o browser)

[Resultados]







When I make an invalid CEP query "05001-XYZ"	features/step_definitions/consulta.rb:23
Then an error message should be displayed	features/step_definitions/consulta.rb:28
response code: 400	
COM ESTE CEP: 05001-XYZ INVÁLIDO, A MENSAGEM -> Bad Request É ESPERADAI	



Given client to provide an API for testing

