**Desafio de Automação da tela de campo de treinamento e seus componentes principais. Desafio proposto pela Cubos como parte do processo seletivo.**

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](https://en.wikipedia.org/wiki/Test_automation) baseado na web que simula cenários para [histórias de usuários](https://en.wikipedia.org/wiki/User_stories) e automatiza testes de [aplicativos da web](https://en.wikipedia.org/wiki/Web_application) para desenvolvimento de software [orientado a comportamento](https://en.wikipedia.org/wiki/Behavior-driven_development) . Está escrito na [linguagem de programação Ruby](https://en.wikipedia.org/wiki/Ruby_(programming_language)).
* Rspec – DLS Ruby, para testar código Ruby.
* SitePrism - [DSL](https://martinfowler.com/books/dsl.html) (Domain Specific Language) criada para facilitar a criação de page objects para testes automatizados em Ruby, utilizando o [Capybara](http://www.rubydoc.info/github/jnicklas/capybara" \t "_blank).

Links úteis

<https://www.guru99.com/alert-popup-handling-selenium.html>

<https://makandracards.com/makandra/1549-accept-or-deny-javascript-confirmation-dialogs-in-capybara-selenium>

<https://imasters.com.br/agile/3-dicas-para-documentacao-de-teste-em-uma-equipe-agil>

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.

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

Obs: Alguns cenários estão em Inglês e outros em Português.

**Arquivo: cadastrar\_feature**

# language: pt

Funcionalidade: Desafio - Campo de Treinamento

-Como um QA Automatizador, quero cadastrar um usuário no campo de treinamento.

@validar

Cenario: Campos obrigatórios devem ser preenchidos

Dado que estou no formulario campo de treinamento

E que eu não preencha as informações obrigatórias

Então uma mensagem de validação é exibida.

Cenario: Cadastrar um usuário com sucesso

Dado que estou no formulario campo de treinamento

E que eu preencha as informações obrigatórias

Quando eu clico em cadastrar

Então eu verifico que o usuário foi cadastrado

**Arquivo:diversos\_feature.**

# language: en

@diversos

Feature: Performing various tests

-As a QA Automator, I want to run several exploratory tests

Scenario: click on button 'Clique Me!'

Given that I am in training form form

Given that I click on button 'Clique Me!'

Then I get the answers "Obrigado"

Scenario: click on popup 'Abrir Popup'

Given that I am in training form form

Given that I click on popup 'Abrir Popup'

Then a pop-up window opens

Scenario: click on button Alert

Given that I am in training form form

Given that I click on button alert 'Alert'

Then a simple alert is displayed

Scenario: click on button Prompt

Given that I am in training form form

Given that I click on button 'Prompt'

And I type a number in the prompt

Then the page displays the number I entered

Passos para executar o projeto.

1. Pré-Condições:
   1. Variáveis de ambiente Java configuradas corretamente
   2. 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>

1. Descompacte o arquivo **desafio\_cubos.rar** em seu diretório de desenvolvimento. Ou efetue o clone do projeto no github   
   <https://github.com/IsaiaSilva/cubos>
2. Abra no cmder ou outro prompt de comando no diretório **tests** e execute o projeto de automação.

bundle install

Bundle exec cucumber

Atenção

No **diretório …tests\features\support\data\homolog.yml**

**Altere a URL para o caminho onde o arquivo foi extraído em sua máquina local.**

**Para mais informações: veja a apresentação do desafio no link a seguir:**

<https://www.youtube.com/watch?v=L5eAe9ZzdvA&t=33s>