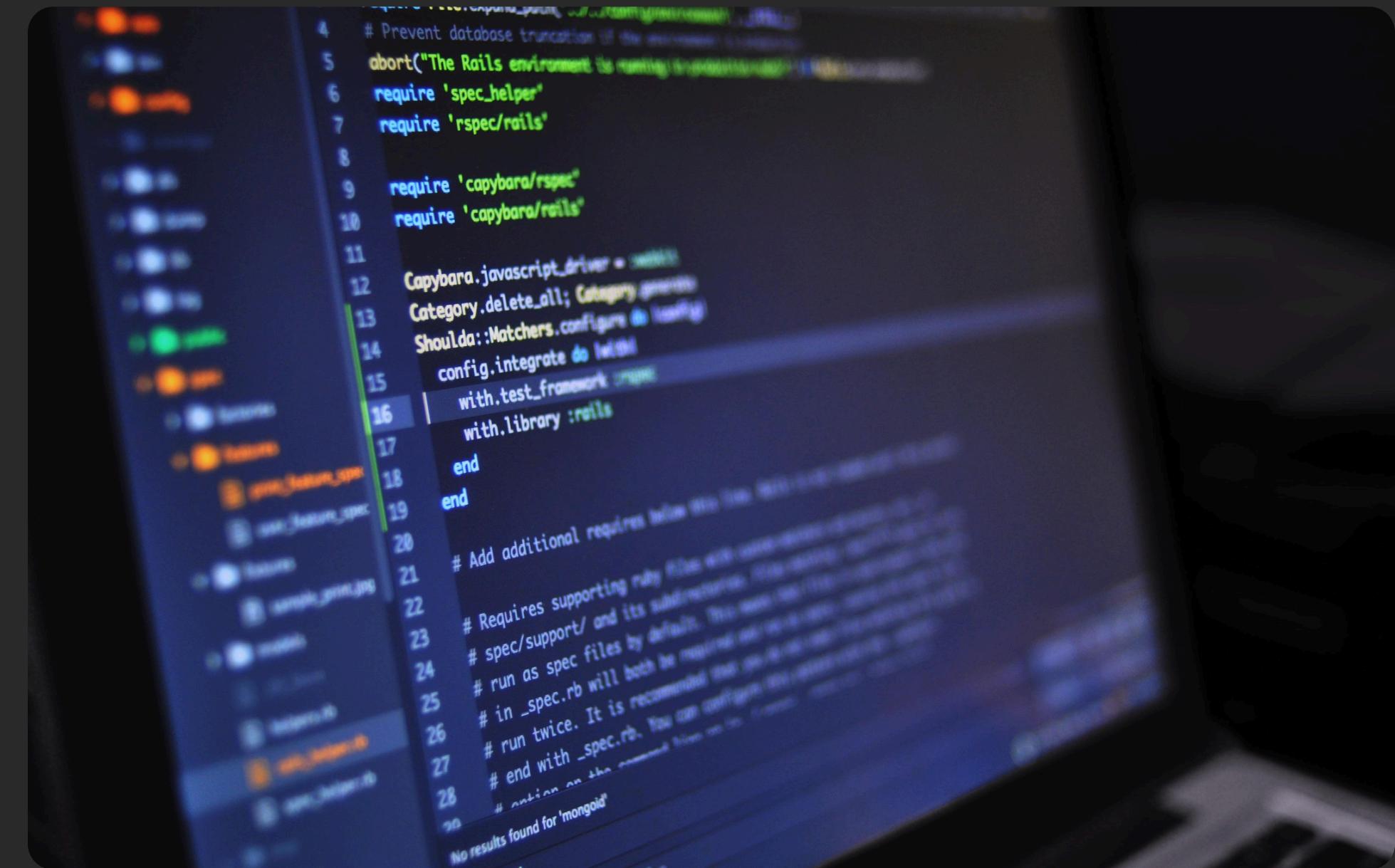


TESTE DE SOFTWARE



```
4 # Prevent database truncation if the database needs cleaning
5 abort("The Rails environment is running in production mode!")
6 require 'spec_helper'
7 require 'rspec/rails'
8
9 require 'capybara/rspec'
10 require 'capybara/rails'
11
12 Capybara.javascript_driver = :webkit
13 Category.delete_all; Category.create!(name: "Category 1")
14 Shoulda::Matchers.configure do |config|
15   config.integrate do |with|
16     with.test_framework :rspec
17     with.library :rails
18   end
19 end
20
21 # Add additional requires below this line to register new matchers
22
23 # Requires supporting ruby files with custom matchers and helpers
24 # in spec/support/ and its subdirectories. This file is normally loaded
25 # in _spec.rb and its subresources. You can add additional files to
26 # run twice. It is recommended that you do not name them spec/support/
27 # end with _spec.rb. You can configure this pattern with the following
28 # option on the command line or in configuration files:
# require 'spec/support'
```

Equipe: Artur Paes de Medeiros e Pedro Paulo Fernandes Cardoso

Site Escolhido

Escolhemos o site do ge(antigo Globo Esporte), que é o portal de esportes da Grupo Globo – ele reúne notícias, reportagens,

- estatísticas e cobertura esportiva sobre futebol, vôlei, basquete, surfe, tênis, lutas, Olimpíadas e várias outras modalidades.

No site há notícias em texto, análises, estatísticas e tabelas de campeonatos (nacionais e internacionais), bem como

- resultados, partidas do dia, mercado de transferências, e conteúdo multimídia(imagens, vídeos).

- Link do site: <https://ge.globo.com>

Print do site:

The screenshot shows the homepage of the ge.globo.com website. At the top, there's a navigation bar with links to globo.com, g1, ge, gshow, globoplay, cartola, seja pro, and sportv. On the right side of the bar are icons for Conta Globo and a user profile. Below the navigation is a green header bar with a 'MENU' button, a 'TIME'S' logo, the 'ge | 100 ANOS DE GLOBO' logo, and a 'BUSCAR' search bar. The main visual is a large banner featuring a roller coaster track at sunset with the text 'É AMANHÃ' overlaid. Below the banner is a caricature collage of various football players. To the right of the collage is a news article about Militão's injury. At the bottom of the page, there's a partial view of a Formula 1 race car.

ge.globo - É esporte sempre

ge.globo.com

globo.com g1 ge gshow globoplay cartola seja pro sportv

Conta Globo

☰ MENU TIME'S

ge | 100 ANOS DE GLOBO

BUSCAR

É AMANHÃ

SELEÇÃO DO BRASILEIRÃO:
ESCOLHA OS SEUS CRAQUES

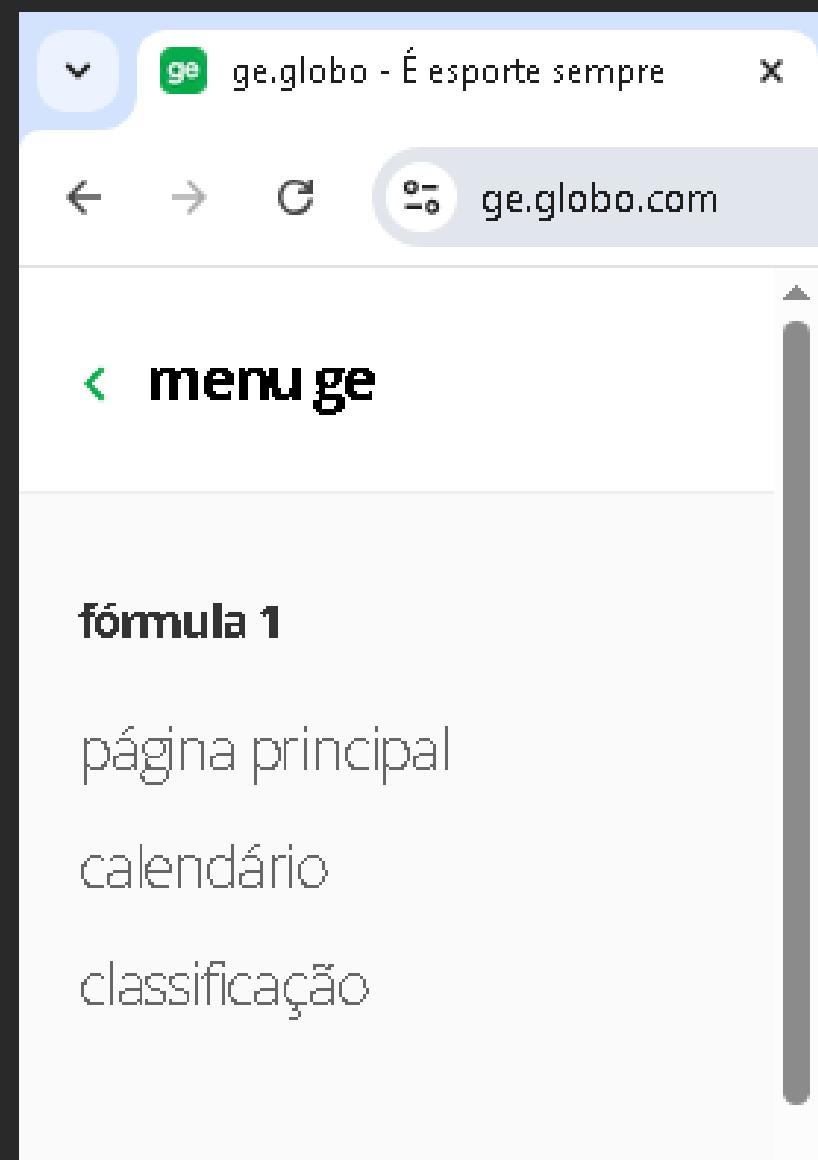
Militão sofre lesão grave no Real Madrid e só volta perto da Copa

Real refuta queixas árbitro após vexame: "Desculpa de m..."

Crise no Liverpool: Salah não viaja para jogo da Champions

Primeiro Teste

- O que o teste faz: Navega pelo menu lateral até a classificação da Fórmula 1 e valida a classificação.
- Resultado Esperado: A página de classificação deve carregar e a tabela de pilotos deve estar visível na tela.

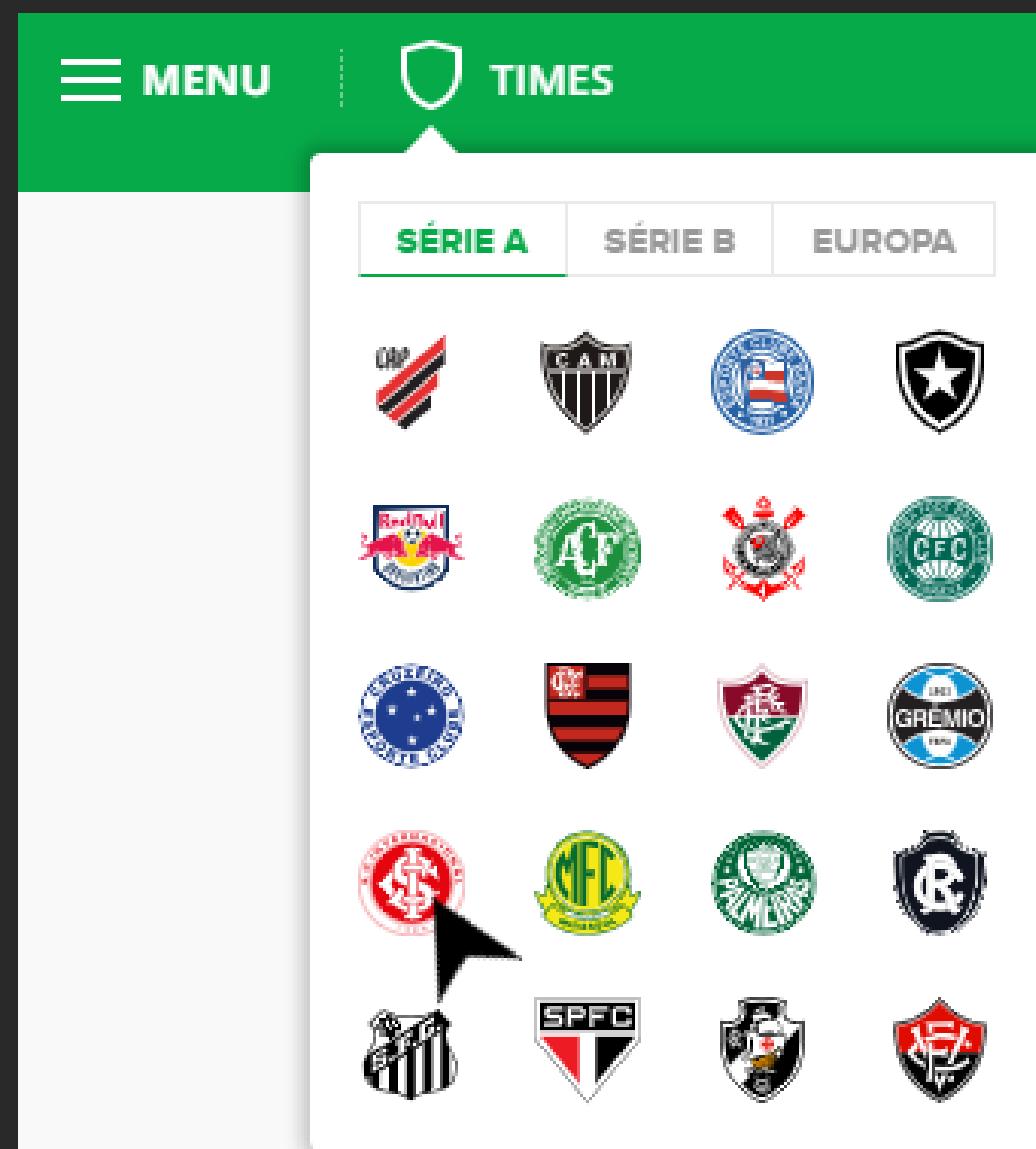


POS.	PILOTOS	EQUIPES	PONTOS
1	Lando Norris GBR	McLaren / Mercedes	423
2	Max Verstappen HOL	RBR / Honda	421
3	Oscar Piastri AUS	McLaren / Mercedes	410
4	George Russell GBR	Mercedes	319
5	Charles Leclerc MON	Ferrari	242

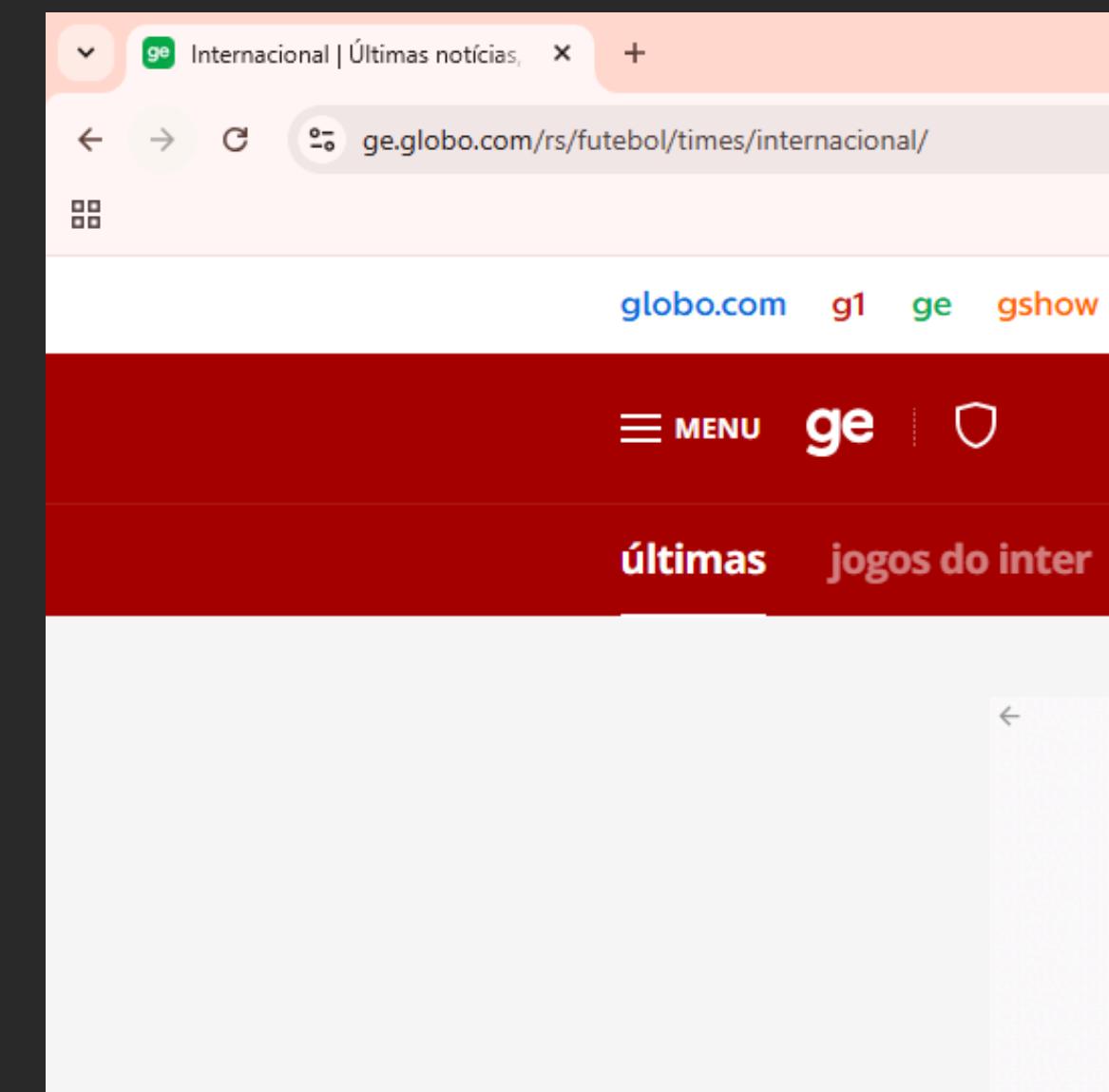
Segundo Teste

O que o teste faz: Clica no escudo do Internacional, na

- aba de times no menu superior, e verifica se vai para a página do time.

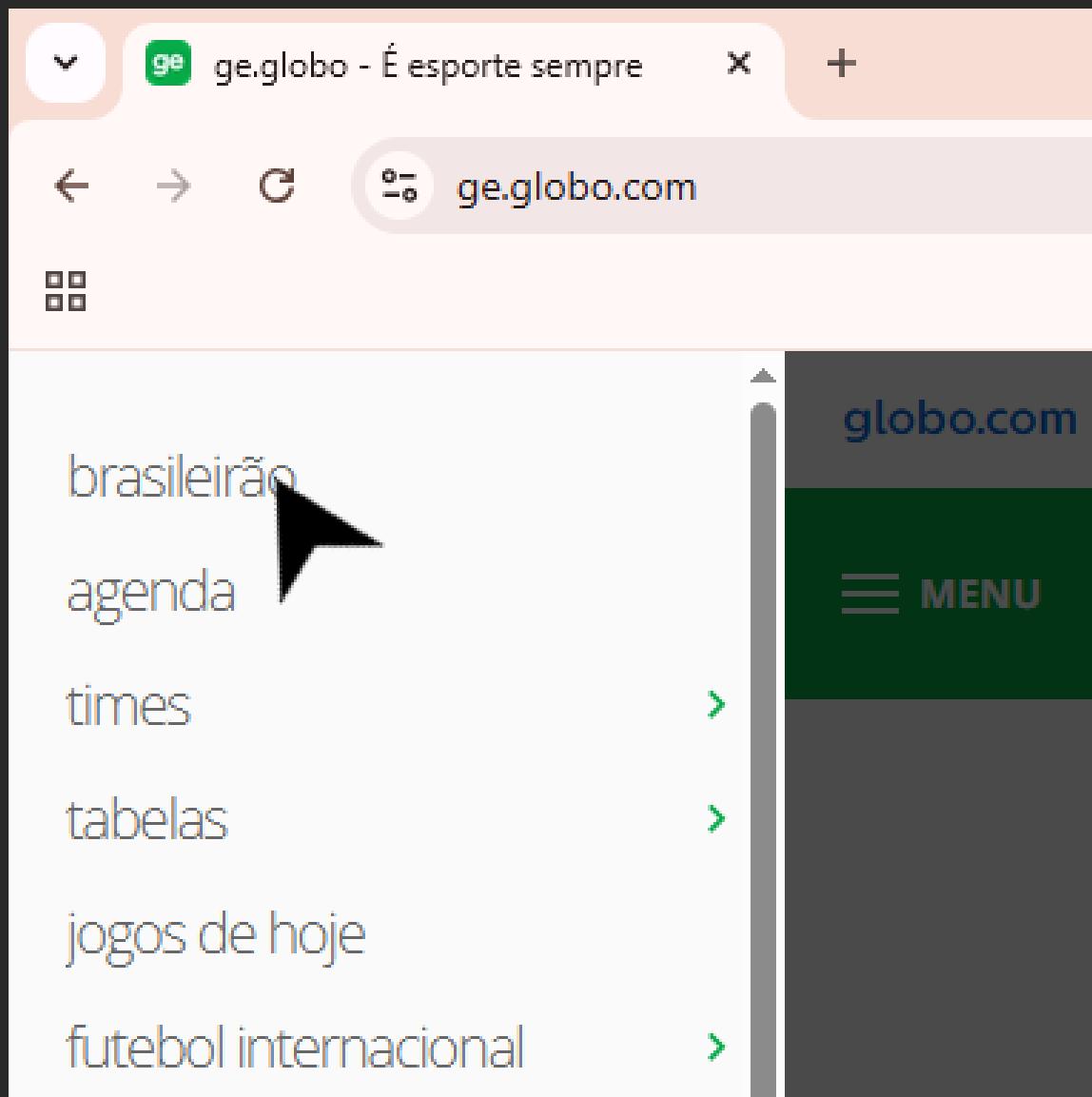


- Resultado Esperado: URL deve conter "times/internacional".



Terceiro Teste

- O que o teste faz: Abre o menu lateral e clica na opção 'brasileirão'. Após, verifica se a tabela carrega a lista completa de times.
- 'brasileirão'. Após, verifica se a tabela carrega a lista completa de times.

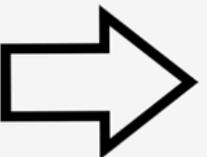
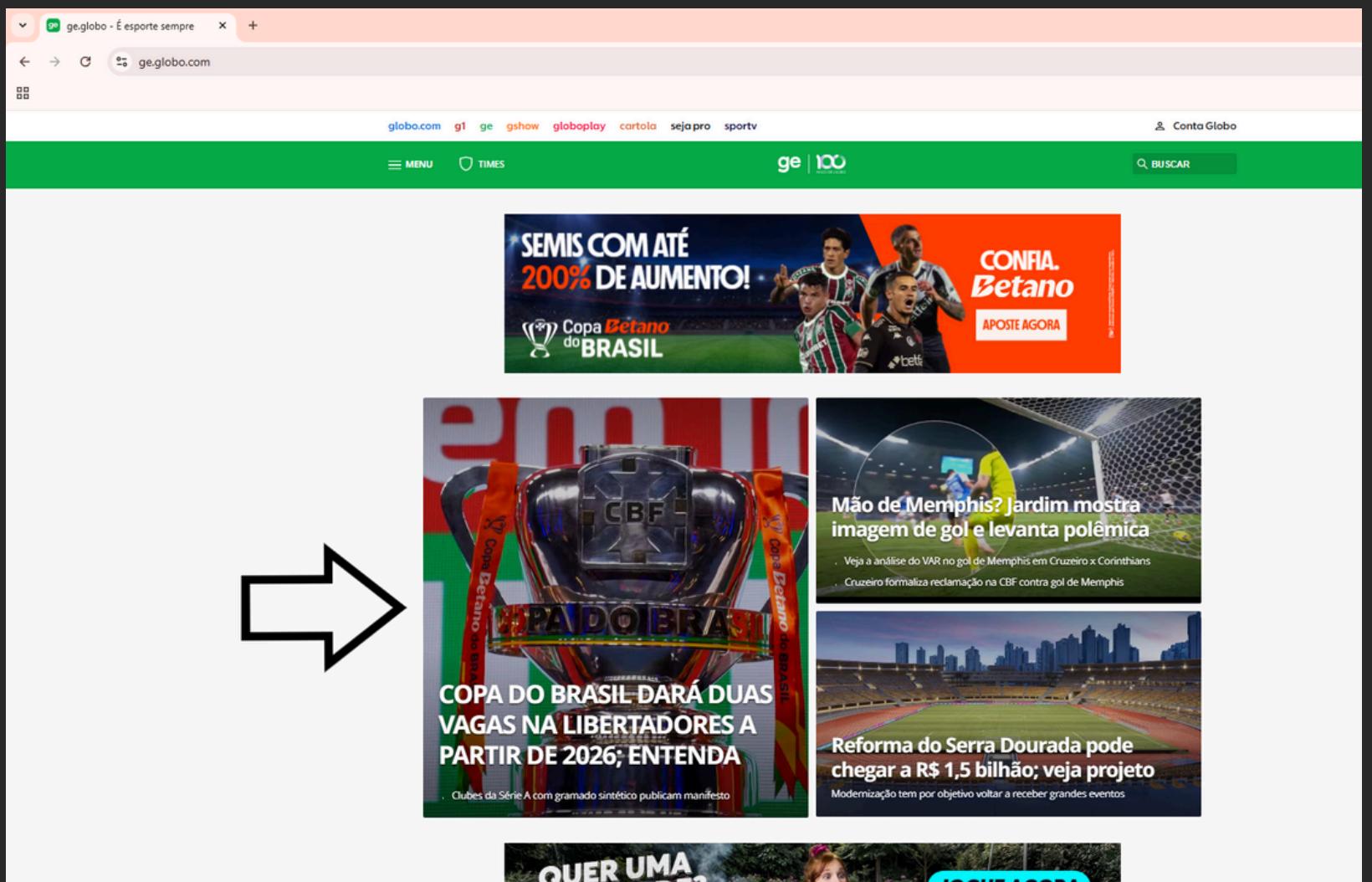


Resultado Esperado: Devem ser encontrados pelo menos 20 times (Série A).

BRASILEIRÃO BETANO SÉRIE A											
CLASSIFICAÇÃO	P	J	V	E	D	GP	GC	SG	%	ÚLT. JOGOS	
1	Flamengo	0	79	38	23	10	5	78	27	51	69
2	Palmeiras	0	76	38	23	7	8	66	33	33	66
3	Cruzeiro	0	70	38	19	13	6	55	31	24	61
4	Mirassol	0	67	38	18	13	7	63	39	24	58
5	Fluminense	0	64	38	19	7	12	50	39	11	56
6	Botafogo	1	63	38	17	12	9	58	38	20	55
7	Bahia	1	60	38	17	9	12	50	46	4	52
8	São Paulo	0	51	38	14	9	15	43	47	-4	44
9	Grêmio	2	49	38	13	10	15	47	50	-3	42
10	Bragantino	1	48	38	14	6	18	45	57	-12	42
11	Atlético-MG	2	48	38	12	12	14	43	44	-1	42
12	Santos	2	47	38	12	11	15	45	50	-5	41
13	Corinthians	3	47	38	12	11	15	42	47	-5	41
14	Vasco	2	45	38	13	6	19	55	60	-5	39
15	Vitória	2	45	38	11	12	15	35	52	-17	39
16	Internacional	2	44	38	11	11	16	44	57	-13	38
17	Ceará	2	43	38	11	10	17	34	40	-6	37
18	Fortaleza	2	43	38	11	10	17	43	58	-15	37
19	Juventude	0	35	38	9	8	21	35	69	-34	30
20	Sport	0	17	38	2	11	25	28	75	-47	14

Quarto Teste

- O que o teste faz: Procura a primeira notícia com apenas texto no artigo e a abre.



- Resultado Esperado: A URL aberta deve corresponder ao link da manchete.

The browser window shows the URL <https://ge.globo.com/sp/futebol/copa-do-brasil/noticia/2025/12/11/copa-do-brasil-classificara-dois-clubes-para-a-libertadores-a-partir-de-2026.shtml>. The page content is identical to the one shown in the first screenshot, confirming the test result.

Trechos de código

```
class HomePage extends BasePage {  
    constructor(driver) {  
        super(driver);  
  
        this.botaoMenu = By.css(".gl-header__menu-button, div.menu-button");  
        this.menuF1 = By.id("menu-1-formula-1");  
        this.linkClassificacao = By.css("#menu-2-classificacao a");  
    }  
  
    async navegarParaClassificacaoF1() {  
        await this.click(this.botaoMenu);  
        await this.driver.sleep(2000);  
  
        const f1Element = await this.find(this.menuF1);  
        await f1Element.click();  
        await this.driver.sleep(2000);  
  
        const classifElement = await this.find(this.linkClassificacao);  
        await this.driver.executeScript("arguments[0].click()", classifElement);  
  
        await this.driver.wait(until.urlContains("classificacao"), 10000);  
    }  
}
```

- Encapsulamento (Classe HomePage):

Na classe, centralizamos seletores complexos (como `By.id("menu-1-formula-1")`) e a lógica técnica de interação (clicar, esperar elemento). O método `navegarParaClassificacaoF1` esconde toda a "sujeira" do Selenium, permitindo que, se o menu do site mudar, a manutenção seja feita apenas neste arquivo.

```
it("Deve navegar pelo menu até a classificação da F1 e validar a tabela", async  
function () {  
    awaitHomePage.navegarParaClassificacaoF1();  
  
    const tabela = await f1Page.obterTabela();  
  
    const estaVisivel = await tabela.isDisplayed();  
    await driver.sleep(3000);  
  
    assert.isTrue(  
        estaVisivel,  
        "A tabela de pilotos deveria estar visível após a navegação"  
    );  
});
```

- Legibilidade e Abstração (Teste it):

No script de teste, o código transforma-se numa narrativa simples. A linha `await HomePage.navegarParaClassificacaoF1()` descreve a ação do utilizador de forma clara, sem poluir o teste com `driver.findElement` ou seletores CSS longos.

— Configuração pré-teste (Setup)

```
describe("Automação de Testes - GE Globo", function () {
  this.timeout(60000);

  let driver;
  let HomePage;
  let f1Page;

  beforeEach(async function () {
    let options = new chrome.Options();
    options.addArguments("--start-maximized");
    options.addArguments("--disable-notifications");

    driver = await new Builder()
      .forBrowser("chrome")
      .setChromeOptions(options)
      .build();

    HomePage = new HomePage(driver);
    f1Page = new ClassificacaoF1Page(driver);

    await HomePage.visit("https://ge.globo.com");
  });
});
```

Mocha (TestRunner):

- Responsável pela estrutura do teste. Utilizamos o describe para agrupar a suite "GE Globo" e o beforeEach para garantir que, antes de cada cenário, o browser abre limpo e os Page Objects são reiniciados, garantindo a independência dos testes.

— Teste Legível e Validação(Assertion)

```
it("Deve abrir a notícia correta ao clicar na manchete principal", async function () {
  const expectedUrl = await HomePage.getMainHeadlineUrl();
  console.log("URL esperada da manchete: " + expectedUrl);
  expect(expectedUrl).to.not.be.null;
  await HomePage.clickMainHeadline();
  await driver.wait(until.urlContains(expectedUrl), 5000);
  const currentUrl = await driver.getCurrentUrl();
  console.log("URL atual aberta: " + currentUrl);
  await driver.sleep(3000);
  expect(currentUrl).to.include(expectedUrl);
});
```

- Chai (Assertion Library): Utilizada para realizar as validações. Através da interface expect, o Chai permite escrever verificações semânticas que se assemelham à linguagem natural (ex: expect(url).to.include(...)), tornando o teste autoexplicativo e fácil de manter."

Dificuldades Encontradas

- Validação de Conteúdo (Manchete)

- Problema: O texto da manchete algumas vezes diferia do título interno da matéria quebrando assim a validação por texto que havia sido implementada.

- Solução: Alteramos a estratégia de validação para Conferência de URL. Capturamos o href (link) da manchete antes do clique e validamos se a URL final continha esse mesmo link, criando um teste muito mais confiável.

Testes Rodando

Link do vídeo: <https://youtu.be/odVqcHMOkwY>