Especificação Técnica e Funcional - Frameworks de Automação

1. Visão Geral

Este documento detalha a implementação de quatro projetos de automação de testes criados para fins de benchmarking e comparação de tecnologias. O objetivo principal foi validar a execução de um cenário de teste simples ("Abrir o Google e verificar o título") em diferentes stacks tecnológicas.

2. Especificação Funcional

2.1 Cenário de Teste

O cenário automatizado em todos os projetos segue o mesmo fluxo funcional:

1. **Ação**: Abrir o navegador.
2. **Ação**: Navegar para `https://www.google.com`.
3. **Validação**: Verificar se o título da página contém a palavra "Google".
4. **Ação**: Fechar o navegador.

2.2 Escopo

Os seguintes frameworks foram implementados e comparados:

1. **Karate** (Java)
2. **Playwright** (Java)
3. **Selenium** (Java)
4. **Selenium** (C# .NET)

---

3. Especificação Técnica

3.1 Projeto 1: KARATENOVO (Karate)

- **Path**: `c:\Users\migue\git\KARATENOVO`
- **Linguagem**: Java 17
- **Build Tool**: Maven
- **Engine**: Karate DSL (sobre Cucumber/Gherkin)
- **Principais Dependências**: `karate-junit5`
- **Execução**: `mvn clean test`

3.2 Projeto 2: KARATENOVO_PLAYWRIGHT (Playwright)

- **Path**: `c:\Users\migue\git\KARATENOVO\KARATENOVO_PLAYWRIGHT`
- **Linguagem**: Java 17
- **Build Tool**: Maven
- **Engine**: Microsoft Playwright Java
- **Principais Dependências**: `playwright`, `junit-jupiter`
- **Execução**: `mvn clean test`

3.3 Projeto 3: KARATENOVO_SELENIUM (Selenium Java)

- **Path**: `c:\Users\migue\git\KARATENOVO\KARATENOVO_SELENIUM`

- **Linguagem**: Java 17
- **Build Tool**: Maven
- **Engine**: Selenium WebDriver 4.x
- **Gerenciamento de Driver**: WebDriverManager
- **Execução**: `mvn clean test`

3.4 Projeto 4: KARATENOVO_SELENIUM_CSHARP (Selenium C#)

- **Path**: `c:\Users\migue\git\KARATENOVO\KARATENOVO_SELENIUM_CSHARP`
- **Linguagem**: C# (.NET 8.0)
- **Framework de Teste**: NUnit
- **Engine**: Selenium WebDriver 4.x
- **Gerenciamento de Driver**: WebDriverManager.Net
- **Execução**: `dotnet test`

---

4. Benchmark e Resultados

Foi realizada uma medição de tempo de execução total (Cold Start + Execução + Teardown) para cada projeto.

4.1 Tabela Comparativa

Ranking	Projeto	Stack	Tempo (s)	Observações
1º	**Selenium C#**	.NET 8 / NUnit	**4.30s**	Excelente performance de inicialização do .NET CLI.
2º	**Selenium Java**	Java / Maven	**9.10s**	Overhead padrão da JVM e Maven.
3º	**Karate**	Java / Karate	**12.07s**	Overhead adicional do interpretador Gherkin/Karate.
4º	**Playwright**	Java / Maven	**17.10s**	Overhead de download/verificação de binários do browser e JVM.

4.2 Análise

- **C# .NET** demonstrou ser a opção mais rápida para execução local de testes unitários/funcionais simples, com um tempo de startup significativamente menor que a stack Java/Maven.
- **Playwright em Java** apresentou o maior tempo, mas isso é comum em execuções "cold" onde ele verifica a integridade dos binários do navegador. Em execuções contínuas, essa diferença tende a diminuir.
- **Selenium Java** mantém-se como um meio-termo sólido e estável.

5. Artefatos Gerados

- **Relatório Gráfico**: `benchmark_report.html` (na raiz do workspace).
- **Código Fonte**: Disponível nos diretórios listados acima.