

3. Explicação dos testes e comentários do código

No teste com **Python**, usei o Selenium para abrir o navegador, acessar a página de login e preencher os campos de usuário e senha com o texto "admin" e "admin123456". Criei uma função que digita lentamente, simulando uma pessoa real digitando. Depois cliquei no botão de login e verifiquei se o endereço da página mudou para destino.html, o que indica que o login funcionou. Usei try e except para capturar erros e driver.quit() para fechar o navegador no fim.

No teste com **Node.js**, a lógica foi a mesma: acessei a URL, esperei carregar, encontrei os campos e preenchi com a digitação letra por letra. Também usei uma pausa entre as teclas e depois cliquei no botão. No final, verifiquei se a URL tinha o texto esperado e fechei o navegador. O código foi feito de forma assíncrona com async/await, como é padrão no JavaScript.

4. Como o Selenium funciona em Python e Node.js

O **Selenium** é uma ferramenta que automatiza o navegador, simulando ações de um usuário como clicar, digitar e navegar. Em **Python**, usamos webdriver e outras bibliotecas para esperar elementos da página e interagir com eles. Em **Node.js**, o funcionamento é parecido, mas usamos async/await para controlar o tempo de execução. Em ambas as linguagens, o Selenium executa testes de forma automática e eficiente, ideal para verificar se páginas estão funcionando corretamente.

5. JMeter

The image shows the configuration window for an HTTP Request in Apache JMeter. The 'Name' field is set to 'Requisição HTTP'. The 'Comments' field is empty. The 'Basic' tab is selected, showing the 'Server Web' section with 'Protocolo [http]:' set to 'http', 'Nome do Servidor ou IP:' set to 'www.hankeds.com.br', and 'Número da Porta:' set to '80'. The 'Requisição HTTP' section shows the 'Method' set to 'GET' and the 'Path' set to '/prova/login1.html'. The 'Content-Type' field is empty. Below this, there are checkboxes for 'Redirecionar automaticamente' (unchecked), 'Seguir redireções' (checked), 'Usar Manter Ativo (KeepAlive)' (checked), 'Usar multipart/form-data para HTTP POST' (unchecked), and 'Browser-compatible headers' (unchecked). The 'Parameters' tab is selected, showing a table for 'Enviar Parâmetros Com a Requisição'.

Nome:	Valor	Codificar?	Content-Type	Incluir Igual?
-------	-------	------------	--------------	----------------

Grupo de Usuários

Nome:

Grupo de Usuários

Comentários:

Ação a ser tomada depois de erro do testador

☒ Continuar

☐ Start Next Thread Loop

☐ Interromper Usuário Virtual

☐ Interromper Teste

☐ Interrompe Teste Agora

Propriedades do Usuário Virtual

Número de Usuários Virtuais (threads):

50

Tempo de inicialização (em segundos)

10

Contador de Iteração

☐ Infinito

10

☒ Same user on each iteration

☐ Delay Thread creation until needed

☐ Agendador

Duração (segundos)

Atraso para início (segundos)

[illegible]

Mesmo com uma quantidade moderada de usuários, o tempo médio de resposta ficou bastante alto, ultrapassando 9 segundos. Isso indica que o servidor **demora para responder** mesmo sob uma carga relativamente leve. Além disso, quase **1 a cada 5 requisições falhou** (18,52%

de erro), o que mostra que o sistema já começa a apresentar instabilidade com 70 usuários simultâneos.

Teste 2 – 150 usuários / 10 loops / ramp-up 10

Total de amostras: 336

Tempo médio de resposta: 32ms

Erro: 35,71%

Tempo máximo: 264ms

Vazão: 33,1 requisições por segundo

O tempo médio de resposta foi bom(32ms), mesmo com alta carga, mostrando que a aplicação está **otimizada para velocidade**. Porém, o número de erros subiu bastante, com **mais de 1/3 das requisições falhando** (35,71%). Isso indica que, embora o servidor seja rápido, **ele não está conseguindo lidar com o volume total de requisições** e começa a recusar ou falhar em algumas.

teste com 70/10/10

Nome do arquivo					Procurar...		Apenas Logar/Exibir		<input type="checkbox"/> Erros	<input type="checkbox"/> Sucessos	Configurar	
Rótulo	# Amostras	Média	Min.	Máx.	Desvio Padrão	% de Erro	Vazão	KB/s	Sent KB/sec	Média de Bytes		
Requisição HTTP	27	9139	16	17068	5278,40	18,52%	1,5/sec	4,27	0,20	2867,		
TOTAL	27	9139	16	17068	5278,40	18,52%	1,5/sec	4,27	0,20	2867,		

teste com 150/10/10

Escrever resultados para arquivo / Ler a partir do arquivo

Nome do arquivo ☐ Erros ☐ Sucessos

Rótulo	# Amostras	Média	Min.	Máx.	Desvio Padrão	% de Erro	Vazão	KB/s	Sent KB/sec	Média de Bytes
Requisição HTTP	336	32	15	264	18,10	35,71%	33,1/sec	81,62	4,42	2528,1
TOTAL	336	32	15	264	18,10	35,71%	33,1/sec	81,62	4,42	2528,1