Centro Universitário Unifacvest
Pamela Vitoria Basqueira de Lima
Larissa Rafaela Fogaça
Ciência da Computação
4º Fase 0104N

2024
Lages

Introdução

Este relatório tem como objetivo analisar as diferenças de execução de cálculos complexos realizados no lado do cliente (client-side), utilizando Javascript, e no lado do servidor (server-side), utilizando PHP. O código em questão implementa operações matemáticas como o cálculo de fatoriais, sequências de Fibonacci, soma de matrizes e resolução de sistemas lineares, para avaliar o desempenho em diferentes dispositivos. A principal finalidade é comparar o tempo de execução, consumo de recursos (CPU e memória) e a experiência do usuário, além de discutir impactos no desenvolvimento de aplicações web, como escalabilidade, eficiência e segurança.

1. Diferenças de Execução entre Cliente e Servidor

Execução de Cálculos no Cliente (JavaScript)

Quando os cálculos são realizados no cliente, o Javascript é executado diretamente no navegador do usuário. Essa abordagem tem suas vantagens e desvantagens:

Tempo de Resposta: A execução no cliente geralmente apresenta menor latência, já que elimina a comunicação constante com o servidor. No entanto, o desempenho pode variar dependendo do dispositivo.

Uso de Recursos (CPU e Memória): O uso de recursos ocorre diretamente no dispositivo do cliente. Em dispositivos mais antigos ou com hardware limitado, isso pode resultar em lentidão ou travamentos.

Experiência do Usuário: Para cálculos leves, a experiência é geralmente boa. Entretanto, cálculos mais complexos podem degradar a desempenho em dispositivos menos potentes.

Execução de Cálculos no Servidor (PHP)

Quando os cálculos são realizados no servidor, o PHP é executado no ambiente do servidor, e os resultados são enviados ao cliente. Essa abordagem é útil para cálculos mais pesados.

Tempo de Resposta: Embora a latência da comunicação possa aumentar o tempo de resposta, cálculos pesados podem ser processados mais rapidamente no servidor, que geralmente possui mais capacidade de processamento.

Uso de Recursos (CPU e Memória): O servidor assume a carga de processamento, beneficiando dispositivos com menos recursos, mas pode ser sobrecarregado em situações de alta demanda.

Experiência do Usuário: O usuário recebe resultados rapidamente, sem sobrecarregar seu dispositivo, melhorando a experiência em tarefas complexas

2. Impactos no Desenvolvimento Web

Cálculos Leves no Cliente

Executar cálculos leves no cliente como validação de formulários, pode melhorar a interatividade e reduzir a carga no servidor. Isso aumenta a escalabilidade, permitindo que o sistema suporte mais usuários simultaneamente.

Escalabilidade: Como os cálculos são realizados no cliente, o servidor pode atender a mais requisições.

Limitações do Cliente: Em dispositivos com hardware limitado, até cálculos simples podem resultar em uma experiência ruim.

Cálculos Pesados no Servidor

Cálculos mais complexos são melhores no servidor, que pode lidar com operações intensivas de forma mais eficiente.

Sobrecarga do Servidor: Com muitos usuários, o servidor pode ficar sobrecarregado, exigindo balanceamento de carga.

Latência: A latência pode ser um problema, especialmente se o servidor estiver distante. Soluções como CDNs (Content Delivery Networks, ou Redes de Distribuição de Conteúdo) podem ajudar a mitigar esse problema. CDNs são redes de servidores distribuídos geograficamente, projetadas para entregar conteúdo da web de forma rápida e eficiente aos usuários. O principal objetivo de uma CDN é reduzir a latência e melhorar o desempenho do site.

3. Segurança e Eficiência

Segurança no Cliente e no Servidor

Validação de Dados: Enquanto validações leves podem ocorrer no cliente, as críticas devem ser realizadas no servidor para evitar manipulações maliciosas.

Cálculos Sensíveis: Operações que envolvem dados sensíveis devem sempre ser feitas no servidor para garantir a segurança.

Eficiência de Processamento

Cálculos Distribuídos: Para animações e gráficos, a execução no cliente é mais eficiente. Já operações intensivas devem ser processadas no servidor.

4. Medição de Desempenho

Os testes de desempenho foram conduzidos em diferentes dispositivos para analisar o impacto de cada abordagem (client-side e server-side) nos tempos de execução e uso de recursos. Para a medição de desempenho utilizamos os dispositivos a seguir:

Dispositivo 1: Nitro V I5 Acer Intel i5, 16 GB de RAM DDR5, Windows 11 home ver 23 H2 (13 th Gen Interl I5 2.10Ghz), navegador Google Chrome.

Dispositivo 2: Macbook Air, M1, 2020, 8GB, Sonoma 14.6.1, navegador Safari

Dispositivo 3: Acer, Intel i3, 4 GB de RAM, Windows 10, ver 22 H2 (Intel Celeron CPU N3450 1.10GHz 1.10), navegador Microsoft Edge.

Dispositivo 4: ASUS, Intel i3, 4 GB de RAM, Windows 10, ver 22 H2 (Intel Core i3-2330M CPU 2.20GHz), navegador Microsoft Edge.

Ambiente de Teste

Dispositivo 1: Nitro V I5				Dispositivo 2: Macbook M1				Dispositivo 1: ACER				Dispositivo 1: ASUS			
Javascript (Cliente)		PHP (Servidor)		Javascript (Cliente)		PHP (Servidor)		Javascript (Cliente)		PHP (Servidor)		Javascript (Cliente)		PHP (Servidor)	
Fibonacci		Fibonacci		Fibonacci		Fibonacci		Fibonacci		Fibonacci		Fibonacci		Fibonacci	
Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução
1)10	0.00000 ms	1)10	0.00691 ms	1)10	0.00000 ms	1)10	0.77796 ms	1)10	0.30000 ms	1)10	0.10681 ms	1)10	0.20000 ms	1)10	0.02885 ms
2)100	0.00000 ms	2)100	0.02289 ms	2)100	0.00000 ms	2)100	1.51992 ms	2)100	0.10000 ms	2)100	0.06700 ms	2)100	0.00000 ms	2)100	0.02503 ms
3)1000	0.10000 ms	3)1000	0.09394 ms	3)1000	0.20000 ms	3)1000	3.42989 ms	3)1000	0.40000 ms	3)1000	0.15283 ms	3)1000	0.20000 ms	3)1000	0.26417 ms
4)10000	0.10000 ms	4)10000	0.33498 ms	4)10000	0.20000 ms	4)10000	1.67236 ms	4)10000	0.20000 ms	4)10000	2.15197 ms	4)10000	0.20000 ms	4)10000	1.33204 ms
5)100000	0.70000 ms	5)100000	3.43299 ms	5)100000	2.50000 ms	5)100000	6.46687 ms	5)100000	1.80000 ms	5)100000	22.31503 ms	5)100000	1.70000 ms	5)100000	11.33084 ms
6)1000000	5.20000 ms	6)1000000	23.16933 ms	6)1000000	6.60000 ms	6)1000000	70.81389 ms	6)1000000	31.80000 ms	6)1000000	114.02583 ms	6)1000000	9.90000 ms	6)1000000	104.33388 ms
Fatorial		Fatorial		Fatorial		Fatorial		Fatorial		Fatorial		Fatorial		Fatorial	
Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução
1)10	0.00000 ms	1)10	0.00536 ms	1)10	0.20000 ms	1)10	0.00000 ms	1)10	87.60000 ms	1)10	23.13503 ms	1)10	0.20000 ms	1)10	51.40901 ms
2)100	0.00000 ms	2)100	0.02694 ms	2)100	1.30000 ms	2)100	1.67704 ms	2)100	SSSS	2)100	165.56811 ms	2)100	0.10000 ms	2)100	65.12380 ms
3)1000	0.10000 ms	3)1000	0.02694 ms	3)1000	0.30000 ms	3)1000	0.08798 ms	3)1000	1.20000 ms	3)1000	0.06503 ms	3)1000	0.50000 ms	3)1000	0.03386 ms
4)10000	0.10000 ms	4)10000	0.09799 ms	4)10000	0.10000 ms	4)10000	0.38004 ms	4)10000	0.80000 ms	4)10000	0.57507 ms	4)10000	0.10000 ms	4)10000	0.54884 ms
5)100000	0.30000 ms	5)100000	0.98300 ms	5)100000	0.40000 ms	5)100000	3.52192 ms	5)100000	1.10000 ms	5)100000	8.14199 ms	5)100000	0.60000 ms	5)100000	2.98405 ms
6)1000000	2.00000 ms	6)1000000	18.63503 ms	6)1000000	2.90000 ms	6)1000000	40.98296 ms	6)1000000	5.10000 ms	6)1000000	132.22694 ms	6)1000000	6.60000 ms	6)1000000	37.18400 ms
Matriz		Matriz		Matriz		Matriz		Matriz		Matriz		Matriz		Matriz	
Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução
1)10	11.80000 ms	1)10	0.01121 ms	1)10	30.40000 ms	1)10	0.12898 ms	1)10	39.20000 ms	1)10	0.20403 ms	1)10	0.20409 ms	1)10	0.13208 ms
2)100	0.10000 ms	2)100	0.03791 ms	2)100	8.10000 ms	2)100	18.18109 ms	2)100	44.10000 ms	2)100	79.68497 ms	2)100	79.68497 ms	2)100	20.38598 ms
3)1000	4.40000 ms	3)1000	3.25894 ms	3)1000	255.20000 ms	3)1000	14.41908 ms	3)1000	1555.00000 ms	3)1000	erro	3)1000	1013.80000 ms	3)1000	erro
4)10000	196.60000 ms	4)10000	396.45700 ms	4)10000	erro	4)10000	erro	4)10000	erro	4)10000	erro	4)10000	erro	4)10000	erro
Sistema Linear		Sistema Linear		Sistema Linear		Sistema Linear		Sistema Linear		Sistema Linear		Sistema Linear		Sistema Linear	
Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução	Entrada	Execução
1)10	11.00000 ms	1)10	0.01788 ms	1)10	27.9000 ms	1)10	0.22101 ms	1)10	35.10000 ms	1)10	36.27205 ms	1)10	38.30000 ms	1)10	68.82191 ms
2)100	0.50000 ms	2)100	0.10300 ms	2)100	19.70000 ms	2)100	82.36599 ms	2)100	21.80000 ms	2)100	204.83685 ms	2)100	14.30000 ms	2)100	151.85595 ms
3)1000	6.10000 ms	3)1000	34.32220 ms	3)1000	1751.30000 ms	3)1000	80.67238 ms	3)1000	9042.500000 ms	3)1000	erro	3)1000	5985.90000 ms	3)1000	111.11617 ms
4)10000	1613.00000 ms	4)10000	28.217.72408 ms	4)10000	erro	4)10000	erro	4)10000	erro	4)10000	erro	4)10000	erro	4)10000	erro

Análise dos Resultados

Os resultados indicam que o servidor foi mais eficiente para cálculos pesados, como a resolução de sistemas lineares, enquanto cálculos leves, como o fatorial, não apresentaram diferenças significativas entre cliente e servidor em termos de tempo de execução. Deste modo pudemos observar o que tínhamos como conhecimento na teoria de que:

- Servidor: Ideal para cálculos intensivos e que requerem alto poder de processamento.
- Cliente: Adequado para cálculos simples e interações dinâmicas com o usuário, desde que os dispositivos dos usuários tenham capacidade suficiente.

Comparando a execução de códigos como Fibonacci e manipulação de matrizes em JavaScript (Cliente) e PHP (Servidor) em diferentes dispositivos, podemos observar questões de velocidade, complexidade do código, capacidade do dispositivo, escalabilidade, latência, desempenho e segurança.

1. **Velocidade:** O JavaScript, executado localmente no cliente, oferece maior velocidade para operações simples, enquanto o PHP, rodando no servidor, apresenta maior estabilidade para códigos mais complexos, mas sofre com a latência da comunicação de rede.

2. **Complexidade do Código:** O JavaScript é adequado para tarefas simples e moderadas, mas pode ter dificuldades em cálculos intensivos em dispositivos menos potentes. O PHP no servidor é mais eficaz para operações complexas, aproveitando os recursos robustos do servidor.

3. **Capacidade do Dispositivo:** Dispositivos com menor capacidade beneficiam-se da execução de código no servidor com PHP, uma vez que o processamento pesado é delegado ao servidor, aliviando o cliente.

4. **Escalabilidade:** O PHP é mais escalável, pois os servidores podem ser ajustados para lidar com volumes maiores de usuários. O JavaScript depende das capacidades do dispositivo do cliente, o que limita sua escalabilidade.

5. **Latência:** O JavaScript no cliente tem menor latência, uma vez que elimina a necessidade de comunicação com o servidor, tornando-o ideal para respostas rápidas. O PHP pode ser mais lento devido à latência de rede.

6. **Desempenho e Ambiente de Teste:** Em dispositivos de diferentes capacidades, o PHP no servidor apresentou uma performance mais consistente para códigos complexos. O JavaScript teve melhor desempenho em tarefas leves, mas variou consideravelmente em dispositivos menos potentes.

7. **Segurança:** O PHP é considerado mais seguro porque processa código e dados no servidor, longe do acesso do usuário, enquanto o JavaScript, executado no cliente, está mais exposto a ataques. Essa diferença confere ao PHP uma camada extra de segurança, mantendo a lógica de processamento oculta. Isso ajuda a mitigar riscos, mas não elimina completamente as vulnerabilidades.

Considerações Finais

A decisão entre executar cálculos no cliente ou no servidor deve considerar a complexidade das operações e a capacidade dos dispositivos dos usuários. Cada abordagem tem seus pontos fortes, e a escolha depende do contexto da aplicação.

Cálculos no Cliente (JavaScript): Ideais para operações simples e que requerem respostas rápidas, essa abordagem reduz a carga no servidor, melhorando a escalabilidade e diminuindo a latência. No entanto, sua eficácia pode ser limitada em dispositivos com menor capacidade de processamento ou em cálculos mais complexos.

Cálculos no Servidor (PHP): Recomendados para operações complexas, que exigem maior segurança e precisam ser executadas de forma consistente, independentemente das limitações dos dispositivos dos usuários. A execução no servidor oferece maior controle sobre o desempenho e a escalabilidade, sendo mais adequada para aplicações robustas e de grande escala.

Em sistemas críticos, a combinação de ambas as abordagens é comum. Operações simples podem ser realizadas no cliente, enquanto cálculos mais pesados são delegados ao servidor, garantindo uma experiência de usuário mais fluida e segura, além de uma maior eficiência no uso de recursos.

Referencias

https://portaldesenvolvedor.com/blog/javascript-vs-php-veja-os-pros-e-contras-de-cada-tecnologia/

https://www.monografias.ufop.br/bitstream/35400000/622/1/MONOGRAFIA_EstudoMecanismosFatores.pdf

https://www.google.com/search?q=avaScript%3A+The+Good+Parts%22+ou+%22PHP+Objects%2C+Patterns%2C+and+Practice%22.&oq=avaScript%3A+The+Good+Parts%22+ou+%22PHP+Objects%2C+Patterns%2C+and+Practice%22.&gs_lcrp=EgZjaHJvbWUyBggAEEUYOTIGCAEQRRg60gEHODc2ajBqN6gCALACAA&sourceid=chrome&ie=UTF-8#fpstate=ive&vld=cid:de938133,vid:hQVTIJBZook,st:0

https://medium.com/@joaovitormunizlopes/client-side-x-server-side-diferen%C3%A7as-conceituais-bc03d01f954c

https://iopscience.iop.org/article/10.1088/1757-899X/801/1/012136/pdf