PW III

Aula Introdutória

Biblioteca vs. Framework

Conceito: É uma coleção de códigos prontos que resolvem um problema específico. Você tem o controle total.

A Analogia: É como ir a uma loja de ferramentas. Você compra um martelo e decide quando, onde e como usá-lo.

Característica principal: **Você** chama a biblioteca no seu código.

Exemplos: React (para criar interfaces), Axios (para requisições), Lodash (para manipulação de dados).

O que é uma Biblioteca (Library)

Conceito: É uma coleção de códigos prontos que resolvem um problema específico. Você tem o controle total.

A Analogia: É como ir a uma loja de ferramentas. Você compra um martelo e decide quando, onde e como usá-lo.

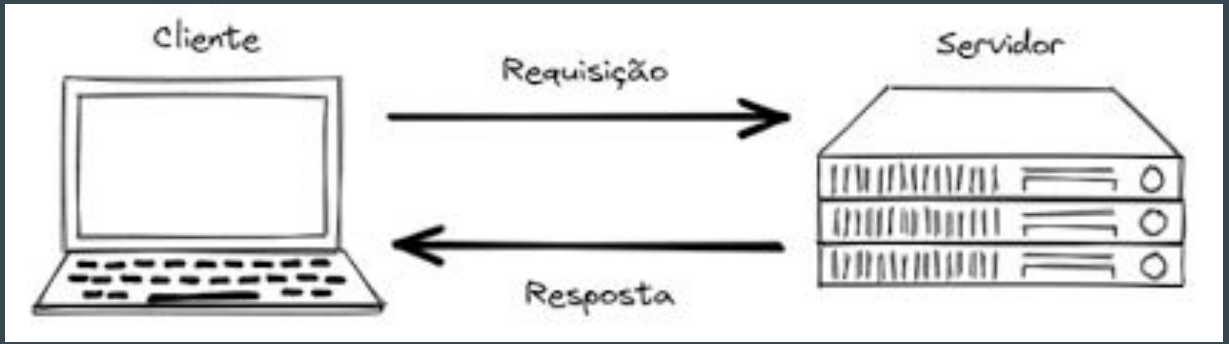
Característica principal: **Você** chama a biblioteca no seu código.

Exemplos: React (para criar interfaces), Axios (para requisições), Lodash (para manipulação de dados).

O que é um Framework?

- **Conceito:** É uma estrutura de trabalho completa que dita a arquitetura do seu projeto. Ele possui regras claras de onde colocar cada arquivo.
- **A Analogia:** É como comprar uma casa pré-fabricada. As paredes e a fundação já estão lá; você apenas decide os móveis e a decoração de cada cômodo.
- **Característica principal:** A Inversão de Controle (*Inversion of Control*). **O Framework** chama o seu código nos momentos certos.
- **Exemplos:** Next.js, Angular, Django, Laravel.

Renderização



Client-Side Rendering (CSR) - Renderização no Cliente

Como funciona: O servidor envia uma página HTML praticamente vazia e um arquivo JavaScript pesado. O navegador (cliente) do usuário baixa esse JS e constrói a tela toda "do zero" ali na hora.

Vantagem: Depois que carrega a primeira vez, a navegação entre as telas é extremamente rápida e fluida (como um aplicativo de celular).

Desvantagem: O carregamento inicial pode ser lento e os robôs do Google (SEO) têm dificuldade de ler a página no primeiro momento.

Exemplo clássico: Aplicações React padrão (Create React App) ou Single Page Applications (SPAs).

Server-Side Rendering (SSR) - Renderização no Servidor

Como funciona: Quando o usuário acessa o site, o servidor processa o código, vai no banco de dados, monta a tela inteira em HTML e entrega a página "pronta e pintada" para o navegador.

Vantagem: Excelente para SEO (o Google já recebe o conteúdo pronto) e o usuário vê a tela muito mais rápido no primeiro acesso.

Desvantagem: Cada clique em um link novo pode exigir que o servidor trabalhe tudo de novo (se não houver otimizações).

Exemplo clássico: Next.js (com Server Components), PHP, ASP.NET.

Como a internet conversa? (Protocolos)

HTTP (Hypertext Transfer Protocol)

- **O que é:** É o idioma básico que o navegador e o servidor usam para trocar informações (como pedir uma página ou enviar um formulário).
- **O Problema:** A comunicação é feita em **texto limpo** .
- **O Risco:** Se alguém interceptar a conexão no meio do caminho (num Wi-Fi público, por exemplo), consegue ler senhas, mensagens e dados de cartão de crédito como se estivesse lendo um livro aberto.

HTTPS (HTTP + Secure)

O que é: É o mesmo protocolo HTTP, mas passando por um "túnel" blindado e criptografado (usando protocolos como TLS/SSL).

A Solução: Antes de enviar os dados, o navegador e o servidor trocam chaves de segurança.

A Segurança: Mesmo que um hacker intercepte os dados no Wi-Fi, ele só verá um embaralhado de letras e números incompreensíveis. É o famoso "cadeadinho" do navegador.