**Desenvolvimento Web**

**Front-end x Back-end:**

O front-end (também chamado de cliente da aplicação) é responsável por toda a parte visual e de interação com o usuário. Estudamos html, css e Javascript, que são os pilares do front-end, além de frameworks. O desenvolvedor front-end tem o objetivo de criar aplicações para os usuários tanto para computadores, notebooks e celulares. Para isso, ele recebe um design (do UX/UI designer, geralmente usando Figma) e coloca em prática esse design.

Já o back-end tem a responsabilidade de aplicar toda a lógica da aplicação, desde a validação de logins, cadastros, integração com banco de dados, integração com sistemas etc.

Interface gráfica do usuário

Descrição gerada automaticamente com confiança média

A comunicação entre o front-end e o back-end é feita por meio do protocolo HTTP. Nos dias atuais, o mais comum é encontrar sistemas separados de front-end e back-end se comunicando por meio do HTTP. Porém, antigamente o mais comum era utilizar do sistema Monolito, onde o front e o back estavam juntos dentro do mesmo sistema funcionando em conjunto.

**SPA:**

SPA (Single Page Application) é o conceito responsável pela criação do framework React. Esse conceito diz que teremos apenas um único documento HTML no nosso projeto. Dessa forma, o usuário estará sempre dentro da mesma página e o React fará o trabalho de redirecionar todas as tags para inserir a página que o usuário quiser. Ao invés de recarregar toda a página ou redirecionar o usuário para uma página nova, apenas o conteúdo principal é atualizado de forma assíncrona, mantendo toda a estrutura da página estática.

Principais vantagens do SPA:

* Não repetição de código;
* Aplicação front-end mais rápida (reduz o conteúdo a ser carregado) e profissional;

Desvantagem:

* Dificuldade em serem encontradas pelo SEO (motores de busca);

Exemplos de SPA: React, VUE.js, Angular.

**SSR:**

SSR (Server Side Rendering) inverte o processo de renderização de uma página em um navegador, trazendo uma parte do esforço de renderização para o servidor. O SSR pode fornecer aos usuários um carregamento mais rápido para as aplicações, além de ajudar com os problemas de SEO, já que parte da aplicação é carregada pelo servidor. Um exemplo de SSR é o Next.js, que é muito similar ao React.

**SSG:**

SSG (Static Site Generation) que é um formato de renderização em que as páginas da aplicação são renderizadas na fase de build da aplicação. O build é basicamente o momento em que o seu código é transformado em um executável final, para que você consiga disponibilizar na rede. Ou seja, você vai fazer toda a sua aplicação simulando um SPA, mas no momento de build ele vai realmente criar páginas HTML e isso resolve os problemas de SEO. A performance nesse formato é superior ao SSR, pelo fato de não ter o trabalho de renderização das páginas por demanda de um servidor, porque todas as páginas HTML já foram criadas no build.