Aluno: Gabriel Soares De Oliveira Roza - SIS

1. O que é HTML?

HTML (HyperText Markup Language) é a linguagem padrão usada para criar e estruturar o conteúdo de páginas web. HTML utiliza tags para delinear elementos como textos, imagens, links e outros componentes que formam a estrutura de um site.

Importância de HTML não ser usado para lógica ou estilo:

Estrutura vs. Apresentação: HTML é focado em estruturar o conteúdo, não em como ele é apresentado ou se comporta. A lógica interativa e a aparência visual são responsabilidade do JavaScript e CSS, respectivamente.

Manutenção e Organização: Manter a estrutura separada da lógica e estilo facilita a manutenção e a colaboração entre desenvolvedores de front-end.

2. Papel do CSS

CSS (Cascading Style Sheets) é utilizado para definir a aparência e o estilo de uma página web, separando a apresentação do conteúdo.

Melhora a Acessibilidade e Flexibilidade:

Separação de Conteúdo e Apresentação: Facilita alterações no design sem modificar o HTML, mantendo a acessibilidade.

Acessibilidade: Proporciona layouts responsivos e melhor contraste de cores, tornando o conteúdo acessível a usuários com diferentes necessidades.

3. Características do JavaScript

JavaScript permite a criação de aplicações complexas e dinâmicas através de características como:

DOM Manipulation: Interage e manipula a estrutura do documento web.

Event Handling: Responde a eventos do usuário (cliques, teclas, etc.).

Asynchronous Programming: Permite operações assíncronas com Promises, async/await.

Client-Side e Server-Side: Utilizável tanto no navegador quanto no servidor com Node.js.

4. Versatilidade do JavaScript

Multiplos Paradigmas:

- **Orientação a Objetos:** Permite a criação de objetos e herança prototípica.
- **Imperativo:** Instruções sequenciais e controle do fluxo.
- **Funcional:** Funções de primeira classe, map, reduce, filter. Essa versatilidade torna o JavaScript adequado para uma ampla gama de aplicações, de scripts simples a grandes aplicações empresariais.

5. Node.js e Desenvolvimento Server-Side

Node.js permite o uso de JavaScript fora do navegador, utilizando o motor V8 do Google Chrome para executar código no servidor. Isso transforma o desenvolvimento server-side ao permitir:

JavaScript Universal: Uso da mesma linguagem no cliente e no servidor.

Performance: Gerenciamento eficiente de operações I/O não-bloqueantes.

6. Interação entre HTML e CSS

Estruturação e Estilização:

- **Separação de Conteúdo e Design:** HTML fornece a estrutura enquanto CSS aplica o estilo, melhorando manutenção e acessibilidade.
- **Benefícios:** Facilita a atualização de estilos sem alterar a estrutura HTML, promove reutilização e consistência no design.

7. CSS Externo, Interno e Inline

CSS Externo:

- **Vantagens:** Estilos centralizados em um arquivo separado.
- **Desvantagens:** Necessita de uma requisição HTTP adicional.
- **Uso:** Ideal para grandes sites com muitos estilos.

CSS Interno:

- **Vantagens:** Estilos aplicados diretamente no arquivo HTML.
- **Desvantagens:** Difícil de manter em grandes projetos.
- **Uso:** Quando estilos são específicos para uma única página.

CSS Inline:

- **Vantagens:** Estilos aplicados diretamente aos elementos.
- **Desvantagens:** Não reutilizável, difícil de manter.
- **Uso:** Ajustes rápidos ou estilos específicos para um elemento.

8. JSON

JSON (JavaScript Object Notation): Formato leve de intercâmbio de dados, utilizado principalmente em APIs web.

- **Importância:** Simplicidade e legibilidade, independente de linguagem de programação, facilitando a comunicação entre sistemas.

9. Importância das Tags h1 e p

Tags h1 e p em HTML:

- **h1:** Define o título mais importante, essencial para a hierarquia e SEO.
- **p:** Define parágrafos de texto, estruturando o conteúdo de forma legível. Essas tags contribuem para a legibilidade e acessibilidade, organizando o conteúdo de maneira clara.

10. Métodos ou Verbos HTTP

Métodos HTTP: Ações realizadas em recursos da web (GET, POST, PUT, DELETE).

11. Método GET

GET: Recupera dados de um servidor sem alterar o estado do recurso.

12. Método POST

POST: Envia dados ao servidor para criar ou atualizar recursos.

13. Método PUT

PUT: Atualiza completamente um recurso existente no servidor.

14. Método DELETE

DELETE: Remove um recurso existente do servidor.

15. Importância da Autenticação na API

Autenticação na API: Protege os dados e garante a integridade, controlando quem pode acessar e modificar os recursos, prevenindo acessos não autorizados e ataques maliciosos.