SSH (Secure Shell)

Se o FTP é para transferir arquivos, o SSH é para **controlar o servidor**. Ele cria um túnel de comunicação seguro e criptografado que permite a você acessar a linha de comando do servidor como se estivesse sentado na frente dele. É usado para instalar programas, configurar o ambiente, gerenciar permissões e realizar tarefas administrativas.

Prática Rápida com SSH: ***(Atente-se às instruções do monitor)***

1. Abra o terminal do seu computador (no Windows, pode ser o PowerShell ou o WSL).
2. Digite o comando de conexão: `ssh seu_usuario@ip_do_servidor`.
3. Após digitar a senha, o prompt de comando mudará para indicar que você está conectado ao servidor remoto (ex: `seu_usuario@nome_do_servidor:~$`).
4. Digite um comando simples como `ls -la` e aperte Enter. A lista de arquivos que aparecerá é a que está *no servidor*, não no seu computador. Você está no controle!

DNS (Domain Name System)

O DNS é a "lista telefônica"da internet. Nenhum dos protocolos acima funcionaria de forma amigável sem ele. Computadores se encontram por números (endereços IP, como `172.217.25.142`), mas nós, humanos, usamos nomes (como `google.com`). O DNS é o sistema cliente-servidor distribuído que traduz os nomes de domínio que digitamos nos endereços IP que as máquinas entendem.

Prática Rápida com DNS: ***(Atente-se às instruções do monitor)***

1. Abra o terminal do seu computador.
2. Digite o comando `nslookup google.com` (ou `dig google.com` em sistemas Linux/-Mac).
3. O resultado mostrará o(s) endereço(s) IP associados àquele domínio. Você acabou de fazer uma consulta DNS manualmente, exatamente o que seu navegador faz secretamente toda vez que você acessa um site.

Espero que as informações aqui apresentadas ajudem a complementar o conteúdo visto na aula de hoje. Até a próxima!

2
HTML: A Linguagem que Estrutura a Web

2.1 O que é HTML?

Seja bem-vindo ao esqueleto da web! No capítulo anterior, vimos como o navegador (cliente) e o servidor conversam usando o protocolo HTTP. Agora, vamos mergulhar na linguagem que o navegador usa para entender e construir o que você vê na tela: o **HTML** (*HyperText Markup Language*, ou Linguagem de Marcação de Hipertexto).

Vamos quebrar esse nome:

- **Linguagem de Marcação:** Diferente de uma linguagem de programação (que executa lógica, cálculos e toma decisões), uma linguagem de marcação serve para **anotar** um documento. Ela não processa informação, mas sim descreve e organiza o conteúdo, dizendo ao navegador: "isto é um título", "isto é um parágrafo", "isto é um link".
- **Hipertexto:** É a parte "mágica"que conecta tudo. Um documento de hipertexto não é isolado; ele pode conter links (*hyperlinks*) que nos levam a outras páginas ou a outras partes da mesma página, criando a teia de informações que chamamos de web.

O fluxo de desenvolvimento de páginas web geralmente acontece em duas etapas: primeiro, criamos e testamos nossas páginas localmente no nosso computador (offline) e, quando estão prontas, nós as enviamos para um servidor para que o mundo inteiro possa acessá-las (online). As atividades aqui descritas seguem esse fluxo, para proporcionar uma imersão prática adequada.

2.1.1 As Ferramentas do Ofício: Editores de Código

Para escrever HTML, tudo o que você precisa é de um editor de texto simples, como o Bloco de Notas ou o Gedit. No entanto, para sermos mais produtivos, usamos editores de código especializados, como o **VSCode**, Atom, ou Sublime Text. Eles nos ajudam com:

- **Destaque de Sintaxe (*Syntax Highlighting*):** Colorem as diferentes partes do código, facilitando a leitura.
- **Autocompletar:** Sugerem código enquanto digitamos.
- **Identificação de Erros:** Apontam quando esquecemos de fechar uma marcação.

2.2 A Estrutura Fundamental de uma Página HTML

Toda página HTML segue uma estrutura básica, um esqueleto que o navegador espera encontrar. Pense nela como a planta de uma casa: tem o terreno, a fundação e os cômodos.

2.2.1 Tags: Os Tijolos da Construção

A base do HTML são as **tags** (etiquetas ou marcações). Elas são os comandos que damos ao navegador, sempre envoltos por sinais de menor e maior (‘< >‘).

A maioria das tags funciona em pares: uma tag de abertura e uma de fechamento. O conteúdo que queremos "marcar"fica entre elas.

- **Abertura:** `<nome-da-tag>`
- **Fechamento:** `</nome-da-tag>` (note a barra ‘/‘)

Por exemplo, para marcar um texto como um parágrafo, usamos a tag `<p>`: `<p>Meu texto aqui.</p>`.

2.2.2 A Estrutura Mínima de um Documento

Vamos analisar a estrutura básica de um arquivo HTML5, a versão mais moderna da linguagem.

Listing 2.1 – Estrutura básica de uma página HTML.

```
1 <!DOCTYPE html>
2 <html>
3     <head>
4         <meta charset="UTF-8">
5         <title>Titulo da Pagina</title>
6     </head>
7     <body>
8     </body>
9 </html>
```

Vamos entender cada parte:

- `<!DOCTYPE html>`: É a primeira linha e serve como uma declaração. Ela diz ao navegador: "Atenção, este documento é uma página HTML5!".
- `<html>`: É a tag raiz, o "terreno"onde toda a nossa página será construída. Todo o resto fica dentro dela.
- `<head>`: O "cérebro"da página. Aqui colocamos meta-informações, ou seja, dados *sobre* o documento que não são exibidos diretamente no conteúdo principal.
  - `<meta charset="UTF-8»`: Essencial! Garante que o navegador interprete corretamente caracteres especiais e acentos do nosso idioma.
  - `<title>`: Define o título que aparece na aba do navegador. É uma tag única por página.

- <body>: O "corpo"da página. Todo o conteúdo visível — textos, imagens, vídeos, links — deve ser colocado aqui dentro.

2.2.3 Organizando o Conteúdo: Títulos e Parágrafos

Dentro do <body>, usamos tags para dar significado e hierarquia ao nosso conteúdo.

- **Títulos (*Headings*):** São definidos pelas tags <h1> a <h6>. <h1> é o título mais importante (geralmente usado apenas uma vez por página), e a importância diminui até o <h6>.
- **Parágrafos (*Paragraphs*):** A tag <p> é usada para agrupar textos em blocos de parágrafos. O navegador automaticamente adiciona um espaçamento antes e depois de cada parágrafo.

Veja no exemplo abaixo como essas tags estruturam o conteúdo visível da página.

Listing 2.2 – Exemplo com títulos e parágrafos.

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Minha Primeira Página</title>
6  </head>
7  <body>
8      <h1>Meu primeiro título</h1>
9      <p>Meu primeiro parágrafo.</p>
10     <p>Meu segundo parágrafo, com um pouco mais de texto para
           preencher o espaço.</p>
11 </body>
12 </html>
```

2.3 Tags Essenciais para Formatação e Estrutura

Além de títulos e parágrafos, existem muitas outras tags que nos ajudam a formatar e organizar o conteúdo.

2.3.1 Formatação de Texto

- <b> e : Ambas deixam o texto em **negrito**. A diferença é semântica: indica que o texto tem uma forte importância, enquanto <b> é apenas para destaque visual.
- <i> e : Ambas deixam o texto em *itálico*. Semanticamente, dá ênfase ao texto.

Vamos ver um exemplo prático de como a falta de fechamento de uma tag afeta o documento.

Listing 2.3 – Tag <b> sem fechamento.

```
1  <!DOCTYPE html>
2  <html>
3  <body>
```

```
4     <p>Aula 2 <b>Este texto eu quero em negrito. Este texto eu não
          quero em negrito.</p>
5 </body>
6 </html>
```

O resultado é que todo o texto após a abertura da tag fica em negrito. Para corrigir, precisamos delimitar exatamente o trecho que queremos formatar.

Listing 2.4 – Tag <b> com fechamento correto.

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4     <p>Aula 2 <b>Este texto eu quero em negrito.</b> Este texto eu
          não quero em negrito.</p>
5 </body>
6 </html>
```

2.3.2 Estrutura e Separação

Existem tags que não envolvem conteúdo, pois representam um elemento único. Elas são chamadas de "tags vazias"ou "de auto-fechamento".

-
: A tag de quebra de linha (*break*). Ela força o conteúdo seguinte a começar em uma nova linha, sem criar um novo parágrafo. Não precisa de fechamento.
- <hr>: A tag de linha horizontal (*horizontal rule*). Ela cria uma linha que cruza a página, usada para separar seções de conteúdo. Também não precisa de fechamento.

2.3.3 Comentários no Código

Às vezes, queremos deixar anotações no nosso código que não devem aparecer para o usuário final. Para isso, usamos os comentários.

- **Sintaxe:**
- **Utilidade:** Serve para explicar trechos de código, deixar lembretes ou "desativar"temporariamente um bloco de HTML sem apagá-lo.

Listing 2.5 – Exemplo de comentário em HTML.

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <title>Página com Comentários</title>
5 </head>
6 <body>
7         <!-- Abaixo um header -->
8     <h1>Meu primeiro título</h1>
9     <p>Meu primeiro parágrafo.</p>
10
```

```
    </body>
</html>
```

2.4 Navegação: Criando Links

O que torna a web uma "teia"é a capacidade de navegar entre páginas através de **hiperlinks**.

2.4.1 A Tag de Âncora: <a>

A tag principal para criar links é a <a> (*anchor*, ou âncora). O texto que fica entre <a> e é o que se tornará clicável. Mas como o navegador sabe para onde o link deve levar? Através de **atributos**. Atributos são informações extras que adicionamos a uma tag para modificar seu comportamento.

- **Atributo `href`:** A "Referência de Hiperlink"é o atributo mais importante da tag <a>. É aqui que especificamos a URL de destino do link.

Listing 2.6 – Página inicial com links.

```
<!DOCTYPE html>
<html>
<head>
    <title>Página Inicial</title>
</head>
<body>
    <h1>Navegar nas páginas</h1>
    <h3>
        Vai para <a href="pagina1.html">página 1!</a>
    </h3>
    <h3>
        Vai para <a href="https://www.wikipedia.org"
           target="_blank">Wikipédia (em nova aba)!</a>
    </h3>
</body>
</html>
```

Listing 2.7 – Página de destino.

```
<!DOCTYPE html>
<html>
<head>
    <title>Página 1</title>
</head>
<body>
    <h1>Entrou na página 1!</h1>
    <h3>
        Voltar para a <a href="pagina_inicial.html">página
           inicial!</a>
    </h3>
</body>
</html>
```

2.4.2 Caminhos: Relativos vs. Absolutos

O valor que colocamos no `href` é um caminho. Existem dois tipos principais:

- **Caminhos Absolutos:** É a URL completa de um recurso na internet, incluindo o protocolo (`http://` ou `https://`). Usamos para criar links para sites externos.
  - Exemplo: `<a href="https://www.google.com»Ir para o Google</a>`
- **Caminhos Relativos:** Apontam para um arquivo dentro da estrutura do nosso próprio site. O caminho é "relativo"à localização da página atual.
  - `pagina1.html`: Procura o arquivo na mesma pasta.
  - `imagens/logo.png`: Procura o arquivo ‘logo.png‘ dentro da subpasta ‘imagens‘.
  - `../pagina_anterior.html`: Volta um nível de pasta e procura o arquivo.

2.4.3 Atributo `target`

Por padrão, quando clicamos em um link, a nova página carrega na mesma aba. Podemos mudar isso com o atributo `target`.

- `target="_self"`: Comportamento padrão. Carrega na mesma aba.
- `target="_blank"`: Abre o link em uma nova aba ou janela.

Exemplo:
`<a href="https://www.wikipedia.org" target=”_blank“>Abrir Wikipédia em nova aba</a>`

Regras para Nomes de Arquivos

Uma dica de ouro: ao nomear seus arquivos HTML, evite usar espaços, letras maiúsculas e acentos. Prefira nomes curtos, descritivos e em minúsculas, usando hífens para separar palavras (ex: `sobre-nos.html`).

2.4.4 Entendendo a Hierarquia: Pais, Filhos e o DOM

Quando aninhamos uma tag dentro da outra, como em `<p><b>texto</b></p>`, estamos criando uma relação de hierarquia. Nesse caso, a tag `<p>` é a "mãe"da tag `<b>`, que é a "filha". Tags no mesmo nível de aninhamento, como dois parágrafos seguidos, são consideradas "irmãs".

Essa estrutura de aninhamento não é apenas uma boa prática de organização; é a forma como o navegador entende a sua página. Quando o navegador lê o seu arquivo HTML, ele o converte em uma estrutura de dados em memória, que se parece com uma árvore genealógica. Essa representação é chamada de **DOM (*Document Object Model*, ou Modelo de Objeto do Documento)**.

Pense no DOM como um mapa vivo da sua página. Cada tag, cada texto, cada atributo se torna um "nó"nessa árvore.

- A tag `<html>` é o nó raiz da árvore.

- <head> e <body> são seus primeiros filhos.
- E assim por diante, até o último pedaço de texto.

É essa estrutura do DOM que permite que outras tecnologias, como o CSS e principalmente o JavaScript, encontrem, modifiquem, adicionem ou removam elementos da sua página de forma dinâmica, mesmo depois que ela já foi carregada.

2.4.4.0.1 Visualizando o DOM:

Para ter uma ideia clara de como essa árvore é formada, você pode usar ferramentas online. Você pode usar o **DOM Visualizer**[1], que é uma ótima forma de ver a hierarquia do seu código HTML de maneira gráfica. Por exemplo, o código a seguir:

Listing 2.8 – HTML de referencia para demonstração do DOM.

```
<!DOCTYPE html>
<html>
    <head>
  <meta charset="UTF-8">
        <title>Título da Página</title>
    </head>
    <body>
        <h1>Meu primeiro título</h1>
        <p>Meu primeiro parágrafo</p>
    </body>
</html>
```

Pode ser visto graficamente como na árvore da Figura 1:

2.5 Organização e Estrutura Avançada de Conteúdo

Agora que já dominamos a estrutura básica, os textos e os links, vamos avançar para os blocos de construção que transformam uma página simples em um documento rico e funcional. Vamos aprender a organizar informações em listas, exibir imagens, estruturar dados em tabelas e, o mais importante, interagir com o usuário por meio de formulários.

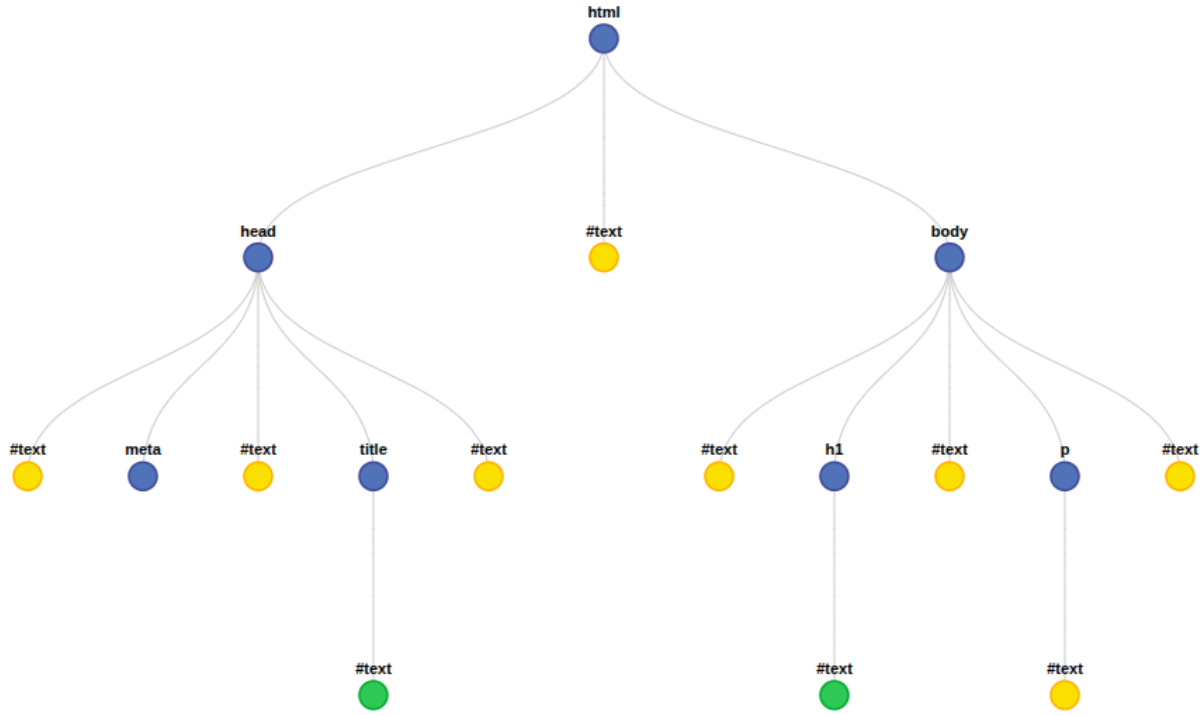
2.5.1 Organizando Informações com Listas

Listas são essenciais para agrupar e organizar informações de forma clara. O HTML nos oferece dois tipos principais de listas.

- **Listas Não Ordenadas <ul>:** Usadas quando a ordem dos itens não importa, como uma lista de ingredientes ou características. O navegador exibe os itens com marcadores (geralmente bolinhas). A tag <ul> (*unordered list*) envolve toda a lista.
- **Listas Ordenadas <ol>:** Usadas quando a sequência é importante, como um passo a passo de uma receita ou um ranking. O navegador numera os itens automaticamente. A tag <ol> (*ordered list*) é a base.

[1] Acesse em: https://bioub.github.io/dom-visualizer/

Figura 1 – DOM de Exemplo



- **Itens da Lista <li>:** Independentemente do tipo de lista, cada item individual é definido pela tag <li> (*list item*).

O poder das listas é ampliado quando as aninhamos, ou seja, colocamos uma lista dentro de outra. Isso é perfeito para criar subcategorias e menus complexos.

Listing 2.9 – Exemplo de listas ordenadas, não ordenadas e aninhadas.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <title>Exemplo de Listas</title>
</head>
<body>

    <h1>Cardápio</h1>

    <h2>Salgados (Ordem não importa)</h2>
    <ul>
        <li>Pastel de Queijo</li>
        <li>Coxinha de Frango</li>
        <li>Esfiha de Carne</li>
    </ul>

    <h2>Como Fazer o Pedido (Ordem importa)</h2>
    <ol>
        <li>Escolha os itens no cardápio.</li>
        <li>Vá até a página "Meu Pedido".</li>
```

```
22          <li>Confirme a quantidade.
23              <ul>
24                  <li>Verifique o subtotal.</li>
25                  <li>Adicione observações se necessário.</li>
26              </ul>
27          </li>
28          <li>Prossiga para o pagamento.</li>
29      </ol>
30
31  </body>
32  </html>
```

2.5.2 Dando Vida à Página com Imagens

Uma página sem imagens é como um livro sem figuras. Para inserir imagens, usamos a tag `<img>`, que é uma tag de auto-fechamento (não precisa de `</img>`). Ela funciona com atributos essenciais:

- `src` (*source*): O atributo mais importante. É aqui que colocamos o caminho (relativo ou absoluto) para o arquivo de imagem que queremos exibir.
- `alt` (*alternative text*): O texto alternativo é crucial para a acessibilidade. Ele descreve a imagem para leitores de tela (usados por pessoas com deficiência visual) e é exibido caso a imagem não consiga carregar.
- `width` e `height`: Definem a largura e a altura da imagem em pixels. É uma boa prática definir esses valores para que o navegador reserve o espaço correto na página enquanto a imagem carrega, evitando que o layout "pule"e melhorando o desempenho.

Listing 2.10 – Exemplo de inserção de imagem com boas práticas.

```
1   <!DOCTYPE html>
2   <html lang="pt-br">
3   <head>
4       <meta charset="UTF-8">
5       <title>Exemplo de Imagens</title>
6   </head>
7   <body>
8
9       <h1>Bem-vindo à Esquina do Sabor!</h1>
10
11      <p>Conheça nossos deliciosos pastéis.</p>
12
13      <img
14          src="imagens/pastel-de-queijo.jpg"
15          alt="Um pastel de queijo dourado e crocante em um prato
                branco."
16          width="300"
17          height="200">
18
19      <p>Imagem de um pastel meramente ilustrativa.</p>
```

```
20
21 </body>
22 </html>
```

2.5.3 Estruturando Dados com Tabelas

Tabelas são a ferramenta perfeita para exibir dados tabulares, como planilhas, comparações de produtos ou grades de horários. A estrutura de uma tabela HTML é bem definida e semântica:

- `<table>`: A tag que envolve toda a tabela.
- `<thead>`: Define o cabeçalho da tabela, onde ficam os títulos das colunas.
- `<tbody>`: O corpo da tabela, onde os dados principais são inseridos, linha por linha.
- `<tfoot>`: O rodapé da tabela, útil para totais e resumos.
- `<tr>` (*table row*): Define uma linha da tabela.
- `<th>` (*table header*): Define uma célula de cabeçalho. O navegador geralmente a formata em negrito e centralizada.
- `<td>` (*table data*): Define uma célula de dados padrão.
- **Acessibilidade:** O atributo `scope` (ex: `<th scope="col">`) ajuda leitores de tela a entender que aquele cabeçalho se refere a toda a coluna, melhorando a experiência para todos os usuários.

Listing 2.11 – Exemplo de tabela semântica para um cardápio.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4      <meta charset="UTF-8">
5      <title>Exemplo de Tabelas</title>
6  </head>
7  <body>
8      <h1>Tabela de Preços</h1>
9
10     <table border="1">
11         <thead>
12             <tr>
13                 <th scope="col">Produto</th>
14                 <th scope="col">Categoria</th>
15                 <th scope="col">Preço (R$)</th>
16             </tr>
17         </thead>
18         <tbody>
19             <tr>
20                 <td>Pastel de Queijo</td>
21                 <td>Salgados</td>
22                 <td>7,00</td>
```

```
            </tr>
            <tr>
                <td>Coxinha de Frango</td>
                <td>Salgados</td>
                <td>6,00</td>
            </tr>
            <tr>
                <td>Caldo de Cana</td>
                <td>Bebidas</td>
                <td>5,00</td>
            </tr>
        </tbody>
        <tfoot>
            <tr>
                <td colspan="2">Total de Itens:</td>
                <td>3</td>
            </tr>
        </tfoot>
    </table>

</body>
</html>
```

Observe o resultado da renderização da tabela na Figura 2.

Figura 2 – Tabela renderizada

Tabela de Preços

Produto	Categoria	Preço (R$)
Pastel de Queijo	Salgados	7,00
Coxinha de Frango	Salgados	6,00
Caldo de Cana	Bebidas	5,00
Total de Itens:		3

2.5.4 Interagindo com o Usuário: Formulários

Formulários são o principal meio de coletar dados do usuário. Seja para um login, uma busca, um pedido ou um cadastro, tudo começa com a tag `<form>`.

Dentro do formulário, usamos diferentes tipos de controles para capturar a informação:

- `<form>`: O contêiner (é, tipo uma caixa) que agrupa todos os controles do formulário. Os atributos `action` (para onde enviar os dados) e `method` (como enviar) são essenciais para o back-end.

- `<input>`: A tag mais versátil. Seu comportamento muda drasticamente de acordo com o atributo `type`.
  - `type="text"`: Campo de texto de uma linha.
  - `type="email"`: Valida se o formato do texto parece um e-mail.
  - `type="password"`: Oculta os caracteres digitados.
  - `type="radio"`: Botão de opção (permite selecionar apenas um de um grupo com o mesmo `name`).
  - `type="checkbox"`: Caixa de seleção (permite selecionar vários de um grupo).
- `<textarea>`: Campo de texto de múltiplas linhas.
- `<select>`: Cria uma lista de opções (dropdown). Cada opção é definida por uma tag `<option>`.
- `<button>`: Cria um botão clicável, geralmente para enviar o formulário.

2.5.4.0.1 Atributos Úteis para Formulários:

Para melhorar a experiência do usuário e garantir dados de qualidade, usamos atributos de validação e ajuda:

- `placeholder`: Mostra um texto de exemplo dentro do campo, que desaparece quando o usuário começa a digitar.
- `required`: Impede o envio do formulário se o campo estiver vazio.
- `maxlength`: Limita o número máximo de caracteres que podem ser digitados.
- `pattern`: Permite definir uma expressão regular para validar o formato dos dados (ex: CPF, CEP).

Listing 2.12 – Exemplo de formulário de pedido para a lanchonete.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4      <meta charset="UTF-8">
5      <title>Exemplo de Formulários</title>
6  </head>
7  <body>
8      <h1>Faça seu Pedido</h1>
9
10     <form action="/enviar-pedido" method="post">
11
12         <p>
13             <label for="nome">Seu Nome:</label><br>
14             <input type="text" id="nome" name="nome_cliente"
                   required placeholder="Digite seu nome completo">
15         </p>
16
```

```
17          <p>
18              <label for="email">Seu E-mail:</label><br>
19              <input type="email" id="email" name="email_cliente"
                    required placeholder="exemplo@email.com">
20          </p>
21
22          <p>
23              <label for="produto">Escolha seu lanche:</label><br>
24              <select id="produto" name="produto_escolhido">
25                  <option value="pastel">Pastel de Queijo</option>
26                  <option value="coxinha">Coxinha de Frango</option>
27                  <option value="caldo_cana">Caldo de Cana</option>
28              </select>
29          </p>
30
31          <p>Para viagem?</p>
32          <input type="radio" id="viagem_sim" name="para_viagem"
                value="sim">
33          <label for="viagem_sim">Sim</label><br>
34          <input type="radio" id="viagem_nao" name="para_viagem"
                value="nao" checked>
35          <label for="viagem_nao">Não, consumir no local</label>
36
37          <p>
38              <input type="checkbox" id="guardanapo" name="adicionais"
                    value="guardanapo">
39              <label for="guardanapo">Enviar guardanapos extras</label>
40          </p>
41
42          <p>
43              <label for="obs">Observações (máx 100
                    caracteres):</label><br>
44              <textarea id="obs" name="observacoes" rows="4" cols="50"
                    maxlength="100"></textarea>
45          </p>
46
47          <button type="submit">Enviar Pedido</button>
48      </form>
49
50  </body>
51  </html>
```

O formulário apresentado, quando renderizado, será semelhante ao apresentado na Figura 3, que segue.

2.5.5 A Semântica do HTML5: Construindo com Significado

No passado, era comum organizar toda a estrutura de uma página usando apenas tags genéricas como (um bloco) e (um trecho de texto). Embora isso funcione, não dá nenhum significado à estrutura.

Figura 3 – Formulário renderizado

O HTML5 introduziu as **tags semânticas**, que descrevem o papel do seu conteúdo. Usá-las é fundamental por três motivos:

1. **Acessibilidade:** Leitores de tela usam essas tags para entender e navegar pela página.
2. **SEO (*Search Engine Optimization*):** Mecanismos de busca como o Google usam a semântica para entender a relevância de cada parte do seu site.
3. **Manutenção do Código:** Fica muito mais fácil para outros desenvolvedores (e para você no futuro) entender a estrutura da página.

- `<header>`: O cabeçalho da página ou de uma seção (geralmente contém o logo e o menu).
- `<nav>`: Agrupa os links de navegação principal.
- `<main>`: Onde fica o conteúdo principal e único daquela página.
- `<section>`: Agrupa um bloco de conteúdo tematicamente relacionado.
- `<article>`: Para um conteúdo autossuficiente e distribuível (ex: um post de blog, uma notícia).
- `<aside>`: Para conteúdo secundário, como uma barra lateral.
- `<footer>`: O rodapé da página ou de uma seção.

Listing 2.13 – Estrutura de uma página usando tags semânticas do HTML5.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <title>Esquina do Sabor - Início</title>
</head>
<body>
```

```
8
9       <header>
10          <h1>Esquina do Sabor</h1>
11          <nav>
12              <ul>
13                  <li><a href="index.html">Início</a></li>
14                  <li><a href="cardapio.html">Cardápio</a></li>
15                  <li><a href="pedido.html">Meu Pedido</a></li>
16              </ul>
17          </nav>
18      </header>
19
20      <main>
21          <section>
22              <h2>Bem-vindo!</h2>
23              <p>Peça já seu lanche: Pastel de Queijo, Caldo de Cana e
                    Coxinha de Frango.</p>
24          </section>
25
26          <article>
27              <h3>Promoção do Dia</h3>
28              <p>Hoje, na compra de um pastel, o caldo de cana sai
                    pela metade do preço!</p>
29          </article>
30      </main>
31
32      <aside>
33          <h4>Horário de Funcionamento</h4>
34          <p>Aberto de Segunda a Sábado, das 8h às 20h.</p>
35      </aside>
36
37      <footer>
38          <p>&copy; 2025 Esquina do Sabor. Todos os direitos
                reservados.</p>
39      </footer>
40
41  </body>
42  </html>
```

Quando renderizado, o código HTML apresentado com a estrutura semântica ficará como disposto na Figura 4. Sem graça, não é? Mas é funcional para quem importa. Percebam que a razão de existência dessas estruturas é para facilitar a compreensão de conteúdo feita por outras máquinas (buscadores, leitores de tela) ou desenvolvedores. Para o usuário da página, não deveria mesmo mudar nada de forma perceptível. Isso, contudo, será percebido com a adição de CSS, nosso próximo conteúdo.

2.5.6 WYSIWYG: Editores Visuais

Por fim, vale a pena mencionar os editores **WYSIWYG** (*What You See Is What You Get* - "O que você vê é o que você obtém"). São ferramentas que permitem criar páginas web de forma visual, arrastando e soltando elementos, sem escrever código diretamente.