**🧩 GIT E GITHUB**

**1. INIT**

Cria um novo repositório Git no diretório atual.

git init

Após esse comando, o Git começa a monitorar os arquivos da pasta.

**2. ADD**

Adiciona arquivos para a área de *staging* (preparação para commit).

git add arquivo.html

ou todos os arquivos:

git add .

**3. COMMIT**

Registra as mudanças no repositório local.

git commit -m "Mensagem explicando o que foi feito"

**4. PUSH**

Envia os commits do repositório local para o repositório remoto (GitHub).

git push origin main

**5. PULL**

Atualiza o repositório local com as mudanças do repositório remoto.

git pull origin main

**🌳 ESTRUTURA DE PASTAS**

Exemplo comum em um projeto web:

meu-site/

│

├── index.html

├── css/

│ └── style.css

├── js/

│ └── script.js

└── img/

└── logo.png

* **css/** → arquivos de estilo
* **js/** → scripts JavaScript
* **img/** → imagens usadas no site

**🧱 ESTRUTURA BÁSICA DO HTML**

<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="UTF-8">

<title>Título da Página</title>

<link rel="stylesheet" href="css/style.css">

</head>

<body>

<h1>Bem-vindo!</h1>

<p>Exemplo de estrutura HTML.</p>

</body>

</html>

**🏷️ TAGS MAIS UTILIZADAS**

| **TAG** | **Função** |
| --- | --- |
| <h1> a <h6> | Títulos |
| <p> | Parágrafo |
| <a> | Link |
| <img> | Imagem |
| <ul>, <ol>, <li> | Listas |
| <div> | Bloco genérico |
| <span> | Texto em linha |
| <form> | Formulário |
| <input>, <label>, <button> | Campos de formulário |

**🎨 PRINCIPAIS PROPRIEDADES DO CSS**

| **Propriedade** | **Função** |
| --- | --- |
| color | Cor do texto |
| background-color | Cor de fundo |
| font-size | Tamanho da fonte |
| margin | Espaçamento externo |
| padding | Espaçamento interno |
| border | Borda |
| width / height | Largura / altura |
| display | Tipo de exibição (block, flex, etc.) |
| text-align | Alinhamento do texto |

**🧾 CLASSES DOS FORMULÁRIOS (HTML/CSS)**

As **classes** servem para estilizar os elementos:

<form class="form-login">

<input type="text" class="input-campo" placeholder="Usuário">

<input type="password" class="input-campo" placeholder="Senha">

<button class="btn-enviar">Entrar</button>

</form>

.input-campo {

width: 200px;

padding: 8px;

margin: 5px 0;

}

.btn-enviar {

background-color: blue;

color: white;

}

**🌐 AMBIENTE CLIENTE / SERVIDOR**

* **Cliente:** navegador do usuário (front-end – HTML, CSS, JS)
* **Servidor:** máquina que processa dados e responde às requisições (back-end – PHP, Python, Node, etc.)

Fluxo:

Cliente → Envia requisição (GET/POST)

Servidor → Processa → Retorna resposta (HTML ou JSON)

**📩 FORMULÁRIOS, VARIÁVEIS E MÉTODOS (PHP)**

**Exemplo básico:**

<form action="recebe.php" method="POST">

<input type="text" name="usuario">

<input type="submit" value="Enviar">

</form>

**Arquivo recebe.php**

<?php

$usuario = $\_POST['usuario']; // recebe o valor enviado

echo "Usuário: $usuario";

?>

**🚀 GET e POST**

| **Método** | **Características** |
| --- | --- |
| **GET** | Envia dados pela URL (menos seguro) |
| **POST** | Envia dados "por baixo", no corpo da requisição (mais seguro) |

**🧠 Validação com IF e isset()**

<?php

if (isset($\_POST['usuario'])) {

$usuario = $\_POST['usuario'];

echo "Bem-vindo, $usuario!";

} else {

echo "Usuário não informado.";

}

?>

isset() verifica se a variável existe e não é nula.

**⚙️ ESTRUTURAS CONDICIONAIS**

**IF / ELSE**

if ($idade >= 18) {

echo "Maior de idade";

} else {

echo "Menor de idade";

}

**🔁 WHILE**

Estrutura de repetição:

$contador = 1;

while ($contador <= 5) {

echo "Número: $contador <br>";

$contador++;

}