DIFERENÇAS ENTRE FRONT-END E BACK-END

Olá, turma! Sejam todos muito bem-vindos à nossa primeira aula sobre o fascinante mundo do desenvolvimento web. Hoje, vamos desvendar dois termos que vocês ouvirão constantemente: **Front-end** e **Back-end**. Pode parecer complicado no início, mas garanto que ao final desta aula, vocês terão uma compreensão clara do que cada um significa e como eles trabalham juntos.

Uma Analogia para Começar: O Restaurante

Imaginem que um site ou aplicativo é como um restaurante. O que vocês veem ao entrar? As mesas, as cadeiras, a decoração, o cardápio. Essa é a parte com a qual vocês interagem diretamente. Isso é o **Front-end**.

Agora, o que acontece nos bastidores para que sua comida chegue deliciosa à mesa? A cozinha, os chefs, os ingredientes, a despensa onde tudo é guardado. Vocês não veem essa parte, mas ela é fundamental para que o restaurante funcione. Isso é o **Back-end**.

A comunicação entre a mesa (Front-end) e a cozinha (Back-end) é feita pelo garçom, que anota o pedido e o leva para ser preparado. No mundo da tecnologia, essa comunicação é feita por algo que chamamos de **API** (Interface de Programação de Aplicativos).

Mergulhando no Front-end: A Aparência do Site

O Front-end é tudo aquilo com que o usuário interage diretamente no navegador. É a parte visual do site ou aplicativo. Para construir essa "fachada" do nosso restaurante digital, utilizamos três ferramentas principais, conhecidas como a "tríade da web":

- **HTML (HyperText Markup Language):** Pensem no HTML como o esqueleto da nossa página. Ele define a estrutura e o conteúdo, como títulos, parágrafos, imagens e links. Não é uma linguagem de programação, mas sim uma linguagem de marcação.
- **CSS (Cascading Style Sheets):** Se o HTML é o esqueleto, o CSS é a "roupa" e a "maquiagem".[3] Ele é responsável pela aparência visual, definindo cores, fontes, espaçamentos e o layout geral da página. Com o CSS, transformamos uma estrutura básica em algo visualmente atraente.
- **JavaScript:** Agora que temos a estrutura (HTML) e a beleza (CSS), o JavaScript entra para dar vida e interatividade. Ele permite criar elementos dinâmicos, como menus que aparecem e desaparecem, animações, validação de formulários e respostas a ações do usuário, como um clique de botão.

Em resumo, o desenvolvedor Front-end é o profissional que constrói a interface com a qual você interage, focando na experiência do usuário.

Explorando o Back-end: O Cérebro por Trás da Operação

O Back-end, também chamado de *server-side* (lado do servidor), é a parte que o usuário não vê. Ele é responsável por tudo que acontece "nos bastidores" para que o site funcione corretamente. Suas principais funções incluem:

- **Processar e armazenar dados:** O Back-end gerencia as informações.[5] Por exemplo, quando você faz um cadastro em um site, seus dados são enviados para o Back-end para serem guardados.
- **Lógica de negócios:** Todas as regras e cálculos complexos são executados no Back-end.
- **Comunicação com o banco de dados:** O Back-end é quem conversa com o "depósito" de informações, o banco de dados.

Para construir a "cozinha" do nosso restaurante digital, os desenvolvedores utilizam linguagens de programação que rodam em um servidor. Hoje, vamos focar em duas muito populares: **PHP** e **Python**.

O Banco de Dados: MySQL

Antes de falarmos de PHP e Python, precisamos entender o que é um banco de dados. Pensem nele como uma grande e organizada "biblioteca" ou "despensa" de informações. O **MySQL** é um dos sistemas de gerenciamento de banco de dados mais populares do mundo. Ele nos permite guardar, organizar, buscar e manipular dados de forma eficiente. Por exemplo, em uma loja online, o MySQL guardaria informações sobre produtos, clientes e pedidos.

PHP com MySQL

PHP (Hypertext Preprocessor) é uma linguagem de programação amplamente utilizada, especialmente projetada para o desenvolvimento web. Uma de suas grandes vantagens é a facilidade de comunicação com bancos de dados, como o MySQL.

Como funciona?
Imagine que você preenche um formulário de cadastro em um site feito com PHP.

1. O Front-end envia os dados que você digitou (nome, e-mail, etc.).
2. O código PHP no servidor recebe esses dados.
3. O PHP se conecta ao banco de dados MySQL.
4. Ele então executa um comando para "inserir" seus dados em uma tabela de usuários.

Da mesma forma, quando você faz login, o PHP recebe seu e-mail e senha, consulta o MySQL para ver se o usuário existe e se a senha está correta, e então permite ou nega o seu acesso.

Python com MySQL

Python é outra linguagem de programação extremamente popular, conhecida por sua simplicidade e versatilidade. Embora seja usada em muitas áreas, como ciência de dados e inteligência artificial, ela também é excelente para o desenvolvimento Back-end.

Como funciona?
Assim como o PHP, o Python pode se conectar e interagir com um banco de dados MySQL. Para isso, ele utiliza "pacotes" ou "bibliotecas", como o mysql-connector-python, que facilitam essa comunicação.

O processo é muito similar ao do PHP:

1. Os dados do usuário chegam do Front-end.
2. Um script Python no servidor recebe esses dados.
3. Usando a biblioteca de conexão, o Python estabelece uma comunicação com o servidor MySQL.
4. O script então envia comandos SQL (a linguagem dos bancos de dados) para inserir, buscar, atualizar ou deletar informações no banco de dados.

Front-end vs. Back-end: Resumo das Diferenças

Front-end (Lado do Cliente)	**Back-end (Lado do Servidor)**
É o que o usuário vê e com o que interage.	É o que acontece nos bastidores.
Focado na experiência visual e de usabilidade.	Focado na lógica, no processamento e no armazenamento de dados.
Tecnologias principais: HTML, CSS, JavaScript.	Tecnologias principais: PHP, Python, Java, Ruby, e bancos de dados como MySQL.
Roda no navegador do usuário.	Roda em um servidor.
Exemplo: O design da página de um produto.	Exemplo: Verificar se o produto está em estoque e processar o pagamento.

Espero que esta aula tenha clareado as ideias sobre o que são Front-end e Back-end. Nas próximas aulas, vamos explorar cada uma dessas tecnologias com mais detalhes e colocar a mão na massa! Alguma pergunta?

ATIVIDADES

1. Olá, Mundo com HTML Puro

Este é o exemplo mais simples. O HTML estrutura a informação que será exibida no navegador.

Passos:

1. Crie e nomeie um arquivo como index.html.
2. Copie e cole o código abaixo dentro dele.

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
```

```
  <title>Exemplo HTML</title>
</head>
<body>
  <h1>Olá, Mundo!</h1>
</body>
</html>
```

Como ver o resultado?
Simplesmente salve o arquivo e clicar 2x sobre ele. O arquivo será interpretado pelo navegador padrão da máquina.

Olá, Mundo!

Explicação:

- <!DOCTYPE html>: Define que o documento é um HTML5.
- <html>: O elemento raiz da página.
- <head>: Contém informações sobre o documento (metadados), como o título da aba (<title>).
- <body>: Contém o conteúdo visível da página.
- <h1>: Define um título de nível 1, o mais importante.

2. Olá, Mundo com HTML e CSS

Agora, vamos adicionar um pouco de estilo! O CSS (a "roupa" da página) pode ser adicionado de algumas formas. Vamos usar a mais organizada: um arquivo separado.

Passos:

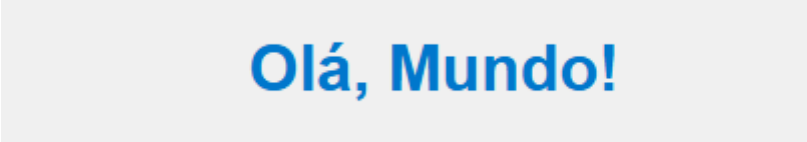
1. Mantenha o seu arquivo index.html.
2. Crie um novo arquivo chamado style.css na mesma pasta.
3. **No arquivo style.css**, cole o seguinte código:

```
body {
  background-color: #f0f0f0; /* Um cinza bem clarinho */
  font-family: sans-serif;
}

h1 {
  color: #0077cc; /* Um tom de azul */
  text-align: center; /* Centraliza o texto */
}
```

Agora, modifique o seu index.html para "chamar" o arquivo CSS. Adicione a linha <link...> dentro da tag <head>:

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <title>Exemplo HTML com CSS</title>
  <!-- Link para o arquivo de estilo CSS -->
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <h1>Olá, Mundo!</h1>
</body>
</html>
```

Como ver o resultado?
Salve os dois arquivos e abra/atualize o index.html no seu navegador. Você verá que o "Olá, Mundo!" está azul, centralizado e o fundo da página está cinza.



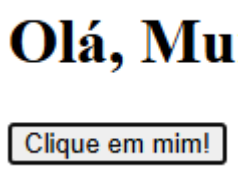
3. Olá, Mundo com HTML e JavaScript (JS)

Vamos adicionar interatividade. O JavaScript pode manipular os elementos da página.
Passos:

1. Crie um novo arquivo index.html.
2. Cole o código abaixo. O JavaScript estará dentro da tag <script>.

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <title>Exemplo HTML com JS</title>
</head>
<body>
  <h1 id="titulo">Olá...</h1>
  <button onclick="mudarTexto()">Clique em mim!</button>

  <script>
    // Função que será chamada pelo botão
    function mudarTexto() {
      // Pega o elemento com o id "titulo"
      var elementoH1 = document.getElementById("titulo");
      // Altera o conteúdo de texto dele
      elementoH1.innerHTML = "Olá, Mundo!";
    }
  </script>
</body>
</html>
```

Como ver o resultado?
Abra o index.html no navegador. A página começará com "Olá...". Ao clicar no botão, o JavaScript será executado e mudará o texto para "Olá, Mundo!".



4. Olá, Mundo com HTML, CSS e JavaScript

Agora vamos juntar tudo: a estrutura (HTML), o estilo (CSS) e a interatividade (JS).

Passos:
Crie a seguinte estrutura de arquivos:

- index.html
- style.css
- script.js

No index.html, cole:

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <title>Exemplo Completo</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <h1 id="titulo">Olá...</h1>
  <button id="meu-botao">Clique em mim!</button>

  <!-- O script é geralmente colocado no final do body -->
  <script src="script.js"></script>
</body>
</html>
```

No style.css, cole:

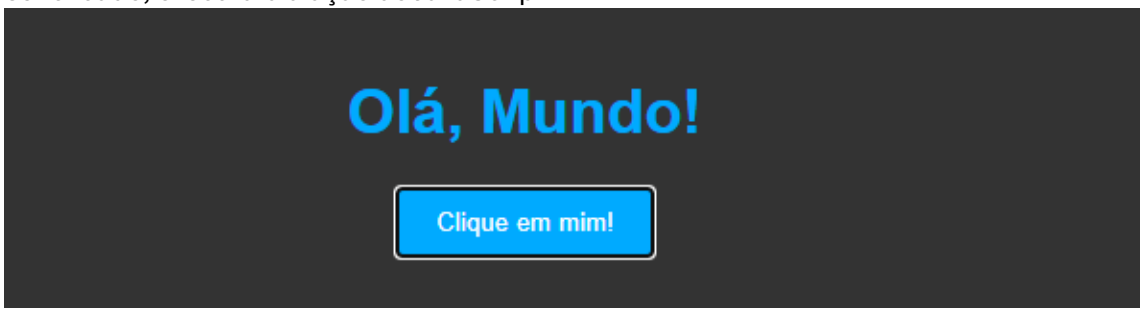
```
body {
  background-color: #333; /* Fundo escuro */
  font-family: sans-serif;
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
  flex-direction: column;
}

h1 {
  color: #00aaff; /* Azul claro */
```

```
}

button {
  padding: 10px 20px;
  border: none;
  background-color: #00aaff;
  color: white;
  cursor: pointer;
}
```

No script.js, cole:

```
// Pega o botão pelo seu id
const botao = document.getElementById("meu-botao");

// Adiciona um "ouvinte de evento" que espera por um clique
botao.addEventListener("click", function() {
  // Quando clicado, pega o título e muda o texto
  const titulo = document.getElementById("titulo");
  titulo.innerHTML = "Olá, Mundo!";
});
```

Como ver o resultado?
Salve todos os arquivos e abra o index.html. Você terá uma página estilizada e o botão, ao ser clicado, executará a ação do JavaScript.



5. Python com HTML e CSS (Usando Flask)

Aqui entramos no **Back-end**. O Python não roda diretamente no navegador. Ele roda em um servidor e *gera* o HTML que é enviado ao navegador. Usaremos o **Flask**, um micro-framework web para Python, para facilitar.

Pré-requisito: Você precisa ter o Python instalado e o Flask. Se não tiver o Flask, abra o terminal/prompt de comando e digite: pip install Flask.

```
C:\Users\User\Desktop\PROZ_BACK\PHP_1\5. Python com HTML e CSS (Usando Flask)\AulaWeb>pip install Flask
Requirement already satisfied: Flask in c:\users\user\appdata\local\programs\python\python38\lib\site-packages (2.2.3)
Requirement already satisfied: Werkzeug>=2.2.2 in c:\users\user\appdata\local\programs\python\python38\lib\site-packages (from Flask) (2.2.3)
Requirement already satisfied: Jinja2>=3.0 in c:\users\user\appdata\local\programs\python\python38\lib\site-packages (from Flask) (3.1.6)
Requirement already satisfied: itsdangerous>=2.0 in c:\users\user\appdata\local\programs\python\python38\lib\site-packages (from Flask) (2.2.0)
Requirement already satisfied: click>=8.0 in c:\users\user\appdata\local\programs\python\python38\lib\site-packages (from Flask) (8.1.8)
Requirement already satisfied: importlib-metadata>=3.6.0 in c:\users\user\appdata\local\programs\python\python38\lib\site-packages (from Flask) (8.5
.0)
Requirement already satisfied: colorama in c:\users\user\appdata\local\programs\python\python38\lib\site-packages (from click>=8.0->Flask) (0.4.6)
Requirement already satisfied: zipp>=3.20 in c:\users\user\appdata\local\programs\python\python38\lib\site-packages (from importlib-metadata>=3.6.0-
>Flask) (3.20.2)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\user\appdata\local\programs\python\python38\lib\site-packages (from Jinja2>=3.0->Flask) (
2.1.5)
```

Passos:

1. Na sua pasta "AulaWeb", crie a seguinte estrutura:

- app.py (o nosso servidor Python)
- Uma pasta chamada templates
  - Dentro de templates, crie o index.html
- Uma pasta chamada static
  - Dentro de static, crie o style.css

No app.py, cole o código Python:

```
from flask import Flask, render_template

# Cria a aplicação Flask
app = Flask(__name__)

# Define a rota para a página inicial ("/")
@app.route("/")
def home():
  # A variável 'mensagem' será enviada para o HTML
  mensagem_ola = "Olá, Mundo, vindo do Python!"
  return render_template("index.html", mensagem=mensagem_ola)

# Linha para permitir rodar o script diretamente
if __name__ == "__main__":
  app.run(debug=True)
```

Na pasta templates/index.html, cole:

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <title>Python com HTML</title>
  <!-- O Flask procura arquivos estáticos na pasta 'static' -->
  <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
  <!-- O texto entre chaves duplas é substituído pelo Python -->
  <h1>{{ mensagem }}</h1>
</body>
</html>
```

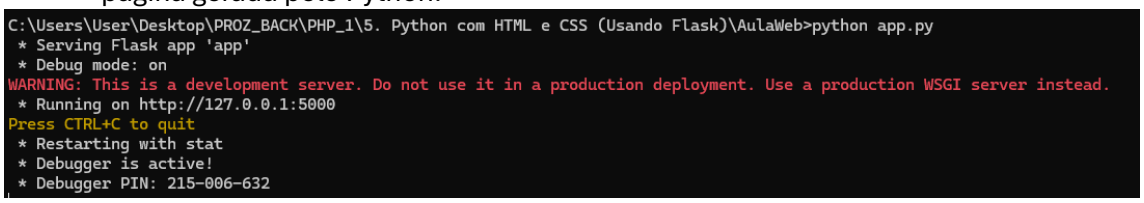
Na pasta static/style.css, cole:

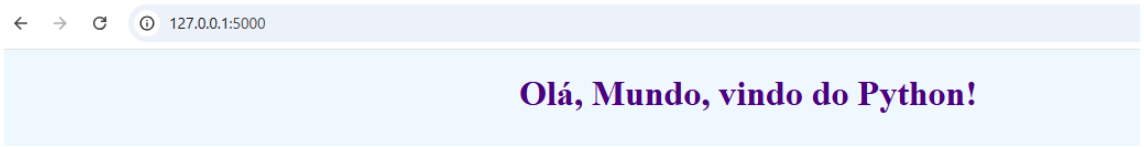
```
body {
  background-color: #f0f8ff; /* Alice Blue */
}
h1 {
  color: #4b0082; /* Indigo */
  text-align: center;
}
```

Como ver o resultado?

1. Abra o terminal do VS Code (Ctrl+ ou Cmd+).

2. Verifique se você está na pasta "AulaWeb".
3. Digite o comando: python app.py (ou python3 app.py).
4. O terminal mostrará algo como "Running on http://127.0.0.1:5000/".
5. Abra seu navegador e acesse esse endereço: http://127.0.0.1:5000. Você verá a página gerada pelo Python!





6. PHP com HTML e CSS

O PHP também é **Back-end**. Assim como o Python, ele precisa de um servidor. A forma mais fácil para iniciantes é usar um pacote como **XAMPP** ou **WAMP** (para Windows) ou **MAMP** (para Mac). Outra opção é usar o servidor embutido do PHP.

Pré-requisitos

1. **XAMPP Instalado:** Garanta que você tenha o XAMPP instalado no seu computador.
2. **XAMPP em Execução:** Abra o Painel de Controle do XAMPP (XAMPP Control Panel). **Inicie o módulo "Apache"** clicando no botão "Start". Ele deve ficar com o fundo verde, indicando que o servidor web está ativo.

Passo 1: Encontrar a Pasta Correta (htdocs)

O servidor Apache do XAMPP não lê arquivos de qualquer lugar do seu computador. Ele tem uma pasta raiz específica chamada htdocs. **Todo e qualquer projeto PHP que você quiser executar através do XAMPP deve estar dentro desta pasta.**

A localização da pasta htdocs geralmente é:

- **Windows:** C:\xampp\htdocs
- **macOS:** /Applications/XAMPP/htdocs
- **Linux:** /opt/lampp/htdocs

Passo 2: Criar a Pasta do Seu Projeto

1. Navegue até a pasta htdocs no seu computador.
2. Dentro de htdocs, crie uma nova pasta para o nosso projeto. Vamos chamá-la de ola_mundo_php.
    - O caminho final será algo como: C:\xampp\htdocs\ola_mundo_php

Passo 3: Abrir o Projeto no VS Code

1. No VS Code, vá em "File" (Arquivo) -> "Open Folder" (Abrir Pasta).
2. Selecione a pasta ola_mundo_php que você acabou de criar dentro de htdocs.

Passo 4: Criar os Arquivos do Projeto

Agora, dentro do VS Code, com a pasta do projeto aberta, crie os dois arquivos:

1. Crie o arquivo index.php. Cole o seguinte código nele:

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <title>PHP com XAMPP</title>
  <!-- O link para o CSS funciona normalmente -->
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <h1>
    <?php
      // Este código PHP é processado pelo servidor Apache do XAMPP
      $texto = "Olá, Mundo, rodando no XAMPP!";
      echo $texto;
    ?>
  </h1>
  <p>Este arquivo está em: <?php echo __DIR__; ?></p>
</body>
</html>
```

(Adicionei uma linha extra que mostra o caminho do arquivo no servidor, o que é útil para confirmar que você está no lugar certo!)

2. Crie o arquivo style.css. Cole o seguinte código nele:

```
body {
  background-color: #fffacd; /* Lemon Chiffon */
  font-family: Arial, Helvetica, sans-serif;
  padding-top: 50px;
}
h1 {
  color: #8b0000; /* Dark Red */
  text-align: center;
  font-size: 3em;
}
p {
  text-align: center;
  color: #555;
}
```

Passo 5: Acessar o Projeto no Navegador (A Mágica do XAMPP)

Esta é a parte crucial e diferente. Você **não** vai abrir o arquivo index.php diretamente no navegador. Em vez disso:

1. Abra seu navegador de internet (Chrome, Firefox, etc.).
2. Na barra de endereços, digite localhost seguido do nome da sua pasta de projeto. A URL será:

http://localhost/ola_mundo_php/

Ao pressionar Enter, o que acontece é:

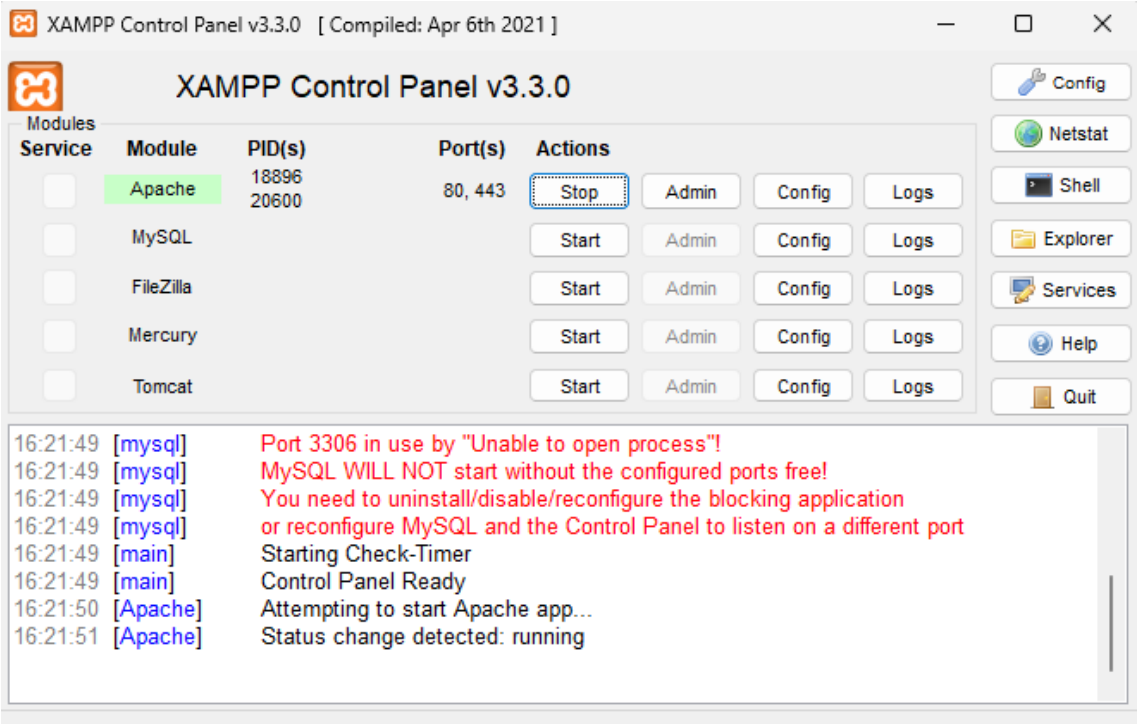
- O navegador envia uma requisição para o servidor Apache rodando na sua máquina (localhost).

- O Apache procura dentro de sua pasta raiz (htdocs) por um subdiretório chamado ola_mundo_php.
- Ao encontrá-lo, ele procura por um arquivo padrão para exibir, como index.php ou index.html.
- Ele encontra o index.php, processa o código PHP que está nele (transforma <?php echo $texto; ?> em Olá, Mundo, rodando no XAMPP!), e envia o resultado (o HTML puro) de volta para o seu navegador.
- O navegador então renderiza o HTML e o CSS, mostrando a página estilizada.

Resumo do Fluxo com XAMPP

1. **Inicie o Apache** no painel do XAMPP.
2. Coloque seus projetos em pastas **dentro de htdocs**.
3. Acesse-os no navegador usando a URL **http://localhost/nome_da_pasta_do_projeto/**.

É isso! Agora você está usando uma configuração de servidor local muito mais próxima de um ambiente de produção real.



USANDO GITHUB

Excelente, turma! Agora que já sabemos como criar nossos projetos no computador, vamos aprender a usar uma ferramenta essencial para todo desenvolvedor: o **GitHub**. Pensem no GitHub como uma **grande biblioteca online para os seus projetos de código**, ou uma espécie de **"pen drive superpoderoso na nuvem"**. Ele serve para:

- **Guardar seus projetos com segurança:** Se o seu computador quebrar, seu código está salvo online.
- **Criar um portfólio:** Você pode mostrar seus projetos para outras pessoas e futuros empregadores.
- **Colaborar com outros desenvolvedores:** Várias pessoas podem trabalhar no mesmo projeto de forma organizada (este é um uso mais avançado).

Hoje, vamos focar em fazer tudo **diretamente pelo site do GitHub, sem precisar instalar ou usar comandos complicados no terminal (Git)**. É o jeito perfeito para começar!

Guia de Uso do GitHub (Modo Navegador)

Primeiro, você precisa de uma conta. Se ainda não tiver, crie uma gratuitamente em github.com.

Parte 1: Como Baixar o Código de um Repositório (Fazer o "Download")

Imagine que você encontrou um projeto legal no GitHub e quer ter os arquivos no seu computador para estudar ou usar. No jargão, isso é chamado de "clonar", mas aqui vamos simplesmente "baixar".

Passo a passo:

1. **Encontre o Repositório:** Navegue até a página do repositório que você quer baixar. Por exemplo, o repositório do Bootstrap, um framework de CSS muito famoso.
2. **Localize o Botão de Código:** Na página principal do repositório, procure por um botão verde bem visível, geralmente no lado direito, que diz **< > Code**.
3. **Clique em "Download ZIP":** Ao clicar no botão < > Code, um menu aparecerá. Na parte de baixo desse menu, você encontrará a opção **"Download ZIP"**. Clique nela.
4. **Descompacte o Arquivo:** O GitHub irá compactar todos os arquivos do projeto em um único arquivo .zip e o download começará. Depois de baixar, encontre o arquivo no seu computador (geralmente na pasta "Downloads") e descompacte-o.

Pronto! Você agora tem uma cópia completa do projeto no seu computador para explorar. Simples assim!

Parte 2: Como Criar Seu Próprio Repositório e Enviar Seus Arquivos (Fazer o "Upload")

Agora, vamos fazer o caminho inverso. Vamos pegar aquele projeto "Olá, Mundo" que fizemos e colocá-lo no seu portfólio online no GitHub.

A) Criando o Repositório Vazio (A "Pasta" na Nuvem)

1. **Vá para a Página Inicial:** Faça login no GitHub. Na página inicial, ou em qualquer página, procure por um sinal de **"+"** no canto superior direito e clique nele. Selecione **"New repository"**.
2. **Preencha o Formulário de Criação:**
    - **Repository name (Nome do repositório):** Dê um nome curto e descritivo, sem espaços (use hifens se precisar). Ex: meu-primeiro-projeto-web.
    - **Description (Descrição):** (Opcional) Escreva uma frase curta explicando o que é o projeto. Ex: "Meu primeiro projeto com HTML, CSS e PHP."
    - **Public / Private:** Esta é uma escolha importante.

        - **Public:** Qualquer pessoa na internet pode ver seu projeto. Ótimo para portfólio.
        - **Private:** Apenas você e pessoas que você convidar podem ver.
        - **Para nosso exemplo, vamos escolher Public**.
    - **Initialize this repository with a README: Marque esta caixa!** Um arquivo README é como a "capa do livro" do seu projeto. Marcando essa opção, o GitHub já cria o repositório com um arquivo inicial, o que facilita muito o upload de outros arquivos pelo navegador.
3. **Clique em "Create repository"**: Ao clicar no botão verde no final da página, seu repositório será criado!

B) Enviando Arquivos Para o Repositório Criado

Agora que você tem sua "pasta" online, vamos colocar os arquivos dentro dela.

1. **Vá para a Página do Repositório:** Você já deve estar nela. Procure o botão **"Add file"** (Adicionar arquivo) e clique nele. Aparecerão duas opções: Create new file (Criar novo arquivo) e Upload files (Enviar arquivos).
2. **Escolha "Upload files":** Vamos enviar os arquivos que já existem no nosso computador (index.php e style.css, por exemplo).
3. **Arraste e Solte (ou Escolha) os Arquivos:** Uma nova página se abrirá. Você pode simplesmente arrastar os arquivos do seu computador para a área indicada ou clicar em **"choose your files"** para abrir o explorador de arquivos e selecioná-los.
4. **Faça o "Commit" (Salve as Alterações):** Esta é a parte mais importante. Depois de enviar os arquivos, você precisa "salvar" essa alteração no histórico do projeto. Isso é chamado de **commit**.
    - Logo abaixo da área de upload, você verá um formulário chamado **"Commit changes"**.
    - No primeiro campo (que já vem com um texto como "Add files via upload"), você pode deixar como está ou escrever um resumo do que fez. Por exemplo: "Adiciona arquivos iniciais do projeto Olá Mundo".
    - O segundo campo, de descrição, é opcional.
    - Clique no botão verde **"Commit changes"**.

Parabéns! Seus arquivos agora estão no GitHub. Você pode recarregar a página e verá seu index.php e style.css listados no repositório.

Resumindo

- **Para Baixar:** Botão verde < > Code -> Download ZIP.
- **Para Criar e Subir:**
    1. + -> New Repository -> Preencha e crie (lembre de marcar o README).
    2. Add File -> Upload files.
    3. Arraste os arquivos.
    4. Escreva uma mensagem de Commit changes e salve.