Linguagem de Programação para Web

Witalo Diego Matias Nunes

Curso Técnico em Desenvolvimento de Sistemas
Educação a Distância
2019

Linguagem de Programação para Web

Witalo Diego Matias Nunes

Curso Técnico em Desenvolvimento de Sistemas

Educação a Distância

Escola Técnica Estadual Professor Antônio Carlos Gomes da Costa

Recife - PE

1.ed. | out. 2019

Professor(es) Autor(es)
Witalo Diego Matias Nunes

Revisão
Witalo Diego Matias Nunes
José Américo Barros

Coordenação de Curso
José Américo Barros

Coordenação Design Educacional
Deisiane Gomes Bazante

Design Educacional
Ana Cristina do Amaral e Silva Jaeger
Fernanda Paiva Furtado da Silveira
Izabela Pereira Cavalcanti
Jailson Miranda
Roberto de Freitas Morais Sobrinho

Descrição de imagens
Sunnye Rose Carlos Gomes

Catalogação e Normalização
Hugo Cavalcanti (Crb-4 2129)

Diagramação
Jailson Miranda

Coordenação Executiva
George Bento Catunda
Renata Marques de Otero
Manoel Vanderley dos Santos Neto

Coordenação Geral
Maria de Araújo Medeiros Souza
Maria de Lourdes Cordeiro Marques

Secretaria Executiva de
Educação Integral e Profissional

Escola Técnica Estadual
Professor Antônio Carlos Gomes da Costa

Gerência de Educação a distância

Dados Internacionais de Catalogação na Publicação (CIP) de acordo com ISDB

N972l

Nunes, Witalo Diego Matias.

Linguagem de Programação para Web: Curso Técnico em Desenvolvimento de Sistemas: Educação a distância / Witalo Diego Matias Nunes. – Recife: Escola Técnica Estadual Professor Antônio Carlos Gomes da Costa, 2019.

77 p.: tab.

Inclui referências bibliográficas.

Caderno eletrônico produzido em outubro de 2019 pela Escola Técnica Estadual Professor Antônio Carlos Gomes da Costa.

1 Programação computacional. 2. Linguagem de programação. II. Título.

CDU – 004.43

Elaborado por Hugo Carlos Cavalcanti | CRB-4 2129

Sumário

Introdução

Caro(a) Aluno(a),

A linguagem de programação para web desempenha um papel muito importante na evolução da internet. Redes sociais, sites de busca, lojas virtuais e sistemas de gerenciamento empresarial são apenas alguns exemplos da capacidade e da sua relevância na rede mundial de computadores.

Você lembra que há algum tempo as páginas web eram estáticas, ou seja, não possuíam interação com o usuário, funcionando apenas como uma fonte de leitura, semelhante a um livro? O tempo passou e a web evoluiu, trazendo consigo uma série de novidades e padrões tecnológicos, permitindo ações que vão de um simples formulário de contato até a compra de um celular novo. Se você parar para perceber melhor, a aquisição de um produto na internet exige uma série de detalhes e este processo tem sido possível graças as modernizações que ocorreram, tendo as linguagens de programação papel fundamental neste cenário.

Para tanto, existem no mercado diversas tecnologias voltadas para o desenvolvimento web, cada uma com suas qualidades e características técnicas. Neste caderno utilizaremos a linguagem de programação PHP (Hypertext Preprocessor).

O PHP é uma linguagem muito popular, poderosa e especialmente projetada para o desenvolvimento web. Com ela podemos criar websites dinâmicos e rápidos integrando desde o HTML (Hypertext Markup Language) até os mais variados bancos de dados.

Nesta disciplina vamos conhecer um pouco deste vasto assunto, começando pelas definições sobre a programação web, enfatizando o PHP e prosseguiremos com exercícios práticos, desenvolvendo nossos próprios scripts usando esta maravilhosa linguagem de programação.

Vamos começar?

1.Competência 01 | Conhecer a estrutura básica para início do desenvolvimento de uma aplicação, tais como declaração de variáveis, projeto, ambiente, entre outros

Seja muito bem-vindo a competência um. Vamos conhecer por aqui a história do PHP e como esta linguagem funciona. Aprenderemos também como construir passo-a-passo o ambiente necessário para testar nossos projetos.

Vamos voltar um pouco no tempo e entender como funciona a programação para web?

1.1 Programação para Web

Os computadores são máquinas modernas e que possuem alto poder de processamento. Com elas podemos efetuar cálculos, criar imagens incríveis e passear virtualmente pelo mundo. Por mais incrível que pareça, cada uma dessas tarefas é resultado de uma sucessão de zeros e uns.

Como assim? Vamos lembrar que os computadores são na sua essência máquinas binárias, produzindo resultados através do sistema binário.

Para nós humanos seria muito difícil enviar instruções e receber resultados desta forma, afinal, não temos a mesma capacidade de processamento de um computador. Por isso, um conjunto de códigos específicos foram criados, facilitando a nossa comunicação com o computador. Este conjunto composto de palavras, símbolos e regras é a linguagem de programação.

DICA IMPORTANTE!

Que tal pesquisar um pouco mais sobre o sistema binário? Acesse: https://brasilescola.uol.com.br/matematica/sistema-numeracao-binaria.htm

Esta linguagem é usada para criar aplicações, mas isso exige conhecimento específico e ao mesmo tempo amplo, pois a falta de visão geral das tecnologias e protocolos envolvidos neste processo comprometem o seu desenvolvimento.

Figura 01 – Exemplo de tecnologias ligadas ao desenvolvimento web
Fonte: SW Agência
Descrição: Notebook ao centro representando o ambiente desenvolvimento e ao seu redor, balões coloridos com diversas tecnologias ligadas ao desenvolvimento web (PHP, JAVA, JS, .NET, XML, DATABASE...).

Portanto, é natural que desenvolvedores iniciantes e até mesmo os mais experientes, oriundos de outras plataformas sintam dificuldade em ingressar no ramo da programação para web, pois a vasta gama de padrões e tecnologias disponíveis levam uma pessoa que está iniciando essa nova carreira a se sentir perdida, sem saber por onde começar.

Começando, ao escolher o PHP como linguagem, estamos optando por uma linguagem interpretada, ou seja, o código é executado todas as vezes em que o usuário requisita uma página web.

Mas como isso funciona?

Veja o exemplo abaixo, figura 2. O usuário acessa um link no navegador. Em seguida, o *browser* entende a solicitação feita e envia ao servidor. Este encontra a página solicitada e processa a informação, transmitindo de volta para o navegador. Desta forma, o usuário recebe a informação solicitada na tela do computador.

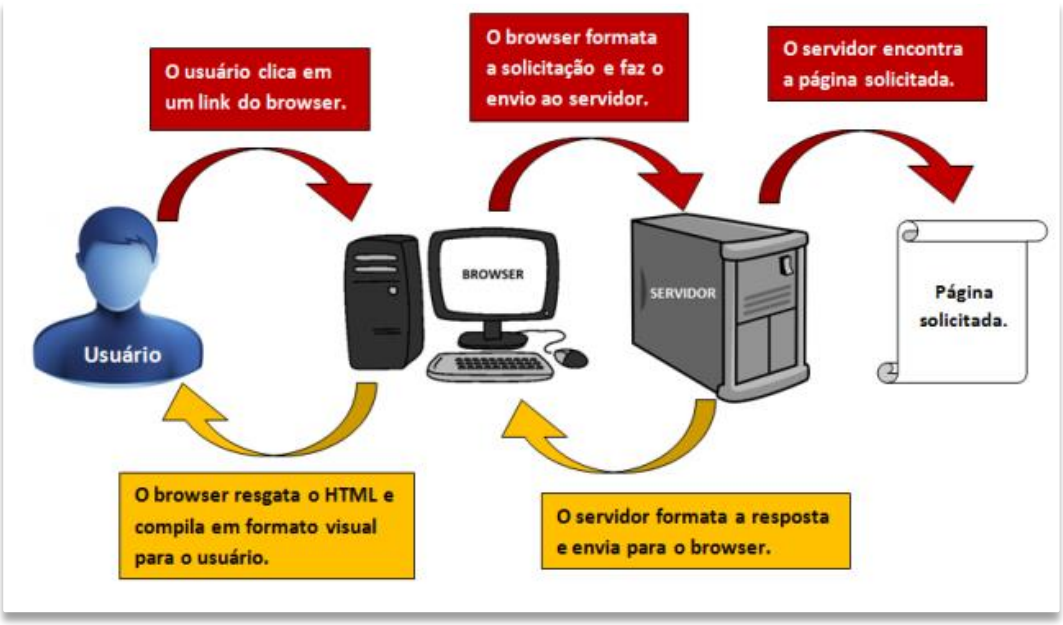

Figura 02 – Funcionamento da requisição de uma página web
Fonte: www.devmedia.com.br
Descrição: Usuário representado por um avatar, acessa uma página web através do navegador em um computador e o servidor encontra a página solicitada, retornando à informação após interpretar a solicitação.

Não parece muito complicado não é mesmo? Agora que conhecemos um pouco mais sobre o funcionamento das páginas web, vamos nos aprofundar na linguagem que escolhemos, o PHP.

DICA IMPORTANTE!
Quanto a execução, linguagens de programação podem ser divididas em interpretadas e compiladas. Você sabe as diferenças entre elas? Acesse o link: https://tinyurl.com/y5nxt6ll

1.2 O que é PHP?

O PHP (um acrônimo recursivo para *Hypertext Preprocessor*) é uma linguagem de script de código aberto e de uso geral, muito utilizada e especialmente adequada para o desenvolvimento web e que pode ser embutida dentro do HTML.

O que significa embutir dentro do código HTML? Veja o exemplo 1 abaixo:

```
<!DOCTYPE HTML>
<html>
    <head>
        <title>Exemplo</title>
    </head>
    <body>

        <?php
            echo "Olá, eu sou um script PHP!";
        ?>

    </body>
</html>
```

Quadro 01 – Código PHP embutido em uma página HTML
Fonte: Manual do PHP – https://www.php.net/manual/pt_BR/intro-whatis.php
Descrição: Caixa de texto contendo código PHP embutido em uma página HTML.

Em vez de muitos comandos para mostrar HTML (como acontece com as linguagens C ou Perl), as páginas PHP contêm HTML em código mesclado que faz "alguma coisa" (neste caso, mostra "Olá, eu sou um script PHP!"). O código PHP é delimitado por instruções de processamento (tags) de início e fim *<?php* e *?>* permitindo que você entre e saia do "modo PHP". (The PHP Group, 2019).

PHP é também uma linguagem de programação dinâmica repleta de funções capazes de manipular banco de dados, formulários, documentos e até mesmo diretórios. Praticamente qualquer projeto relacionado a websites pode ser desenvolvido utilizando esta linguagem.

Vamos conhecer um pouco da sua história?

1.3 História

O PHP nasceu em 1994 e traçou um caminho muito longo até os dias de hoje. Criado por Rasmus Lerdof, foi inicialmente um conjunto simples de binários CGI (Common Gateway Interface) escrito em linguagem C. Rasmus projetou neste início alguns scripts para acompanhar as visitas ao

seu currículo online. Posteriormente, novas funcionalidades foram desenvolvidas dando um novo corpo ao que ele denominou de PHP Tools.

Em 1995 o código fonte tornou-se público, encorajando a participação da comunidade tecnológica. Três anos depois, uma pesquisa feita pela Netcraft indicou que cerca de 60.000 domínios já utilizavam o PHP de alguma forma, perfazendo aproximadamente 1% de todos os domínios presentes na internet daquela época.

Hoje o PHP encontra-se na versão 7 e diversos projetos de código aberto estão disponíveis de forma gratuita e personalizável, demonstrando solidez e a preferência de muitos programadores ao redor do mundo.

SAIBA MAIS!
Ficou curioso para conhecer toda a história e evolução desta linguagem? Acesse:
https://www.php.net/manual/pt_BR/history.php

1.4 Características da linguagem PHP

A linguagem PHP é bastante completa e possui uma série de vantagens que facilitam a vida do desenvolvedor web. Podemos elencar algumas características importantes:

- É uma linguagem fácil de aprender e possui ampla documentação;
- Possui estabilidade e performance excelentes;
- Seu código fonte é aberto, não é preciso pagar por sua utilização e é possível modifica-la de acordo com a necessidade de cada usuário;
- Tem suporte nos principais servidores web do mercado e suporte nativo no servidor web Apache (o mais utilizado no mundo);
- Suporta conexão com os bancos de dados mais utilizados, como por exemplo, MySQL, PostgreSQL, Oracle e DB2;

- É multiplataforma, tem suporte nos mais variados sistemas operacionais (Windows, Linux, Unix, FreeBSD);
- Suporta uma série de padrões e protocolos como o XML, DOM, IMAP, POP3, LDAP, HTTP, entre outros;
- Não necessita ser compilado.

1.5 IDE

IDE significa em inglês *integrated development environment*, traduzindo, ambiente de desenvolvimento integrado. Este ambiente de desenvolvimento agiliza e facilita a construção de scripts através de funções assistentes, como realçar e auto completar o código.

Existem diversas IDEs disponíveis no mercado, distribuídas em versões gratuitas e pagas. Entre as gratuitas estão a Notepad++, Aptana Studio e NetBeans.

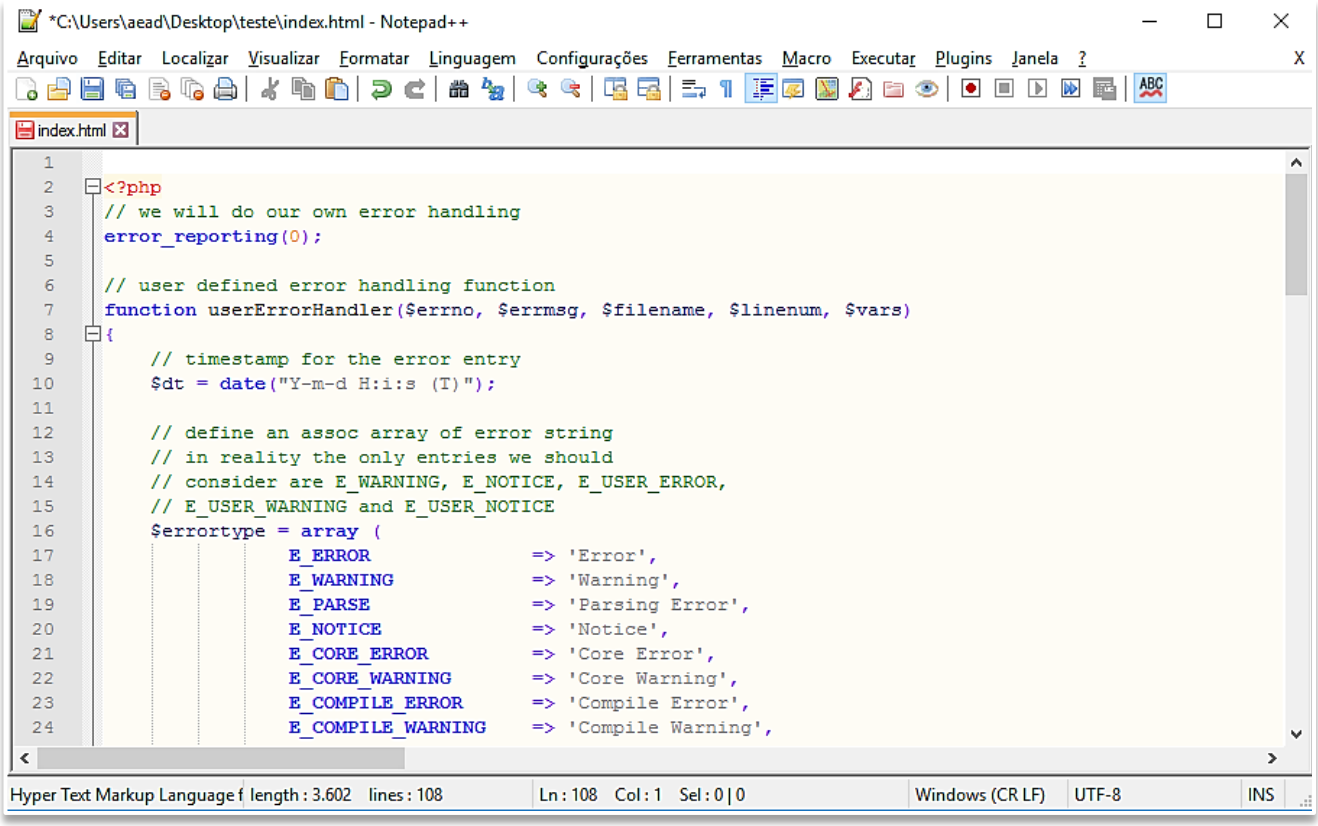

Figura 03 – Notepad++
Fonte: autor
Descrição: Tela principal da IDE Notepad++. A parte superior da tela é composta de menu e abas de formatação,

configuração e edição dos códigos em desenvolvimento. Logo abaixo, todo o espaço para digitação do código fonte da aplicação em desenvolvimento.

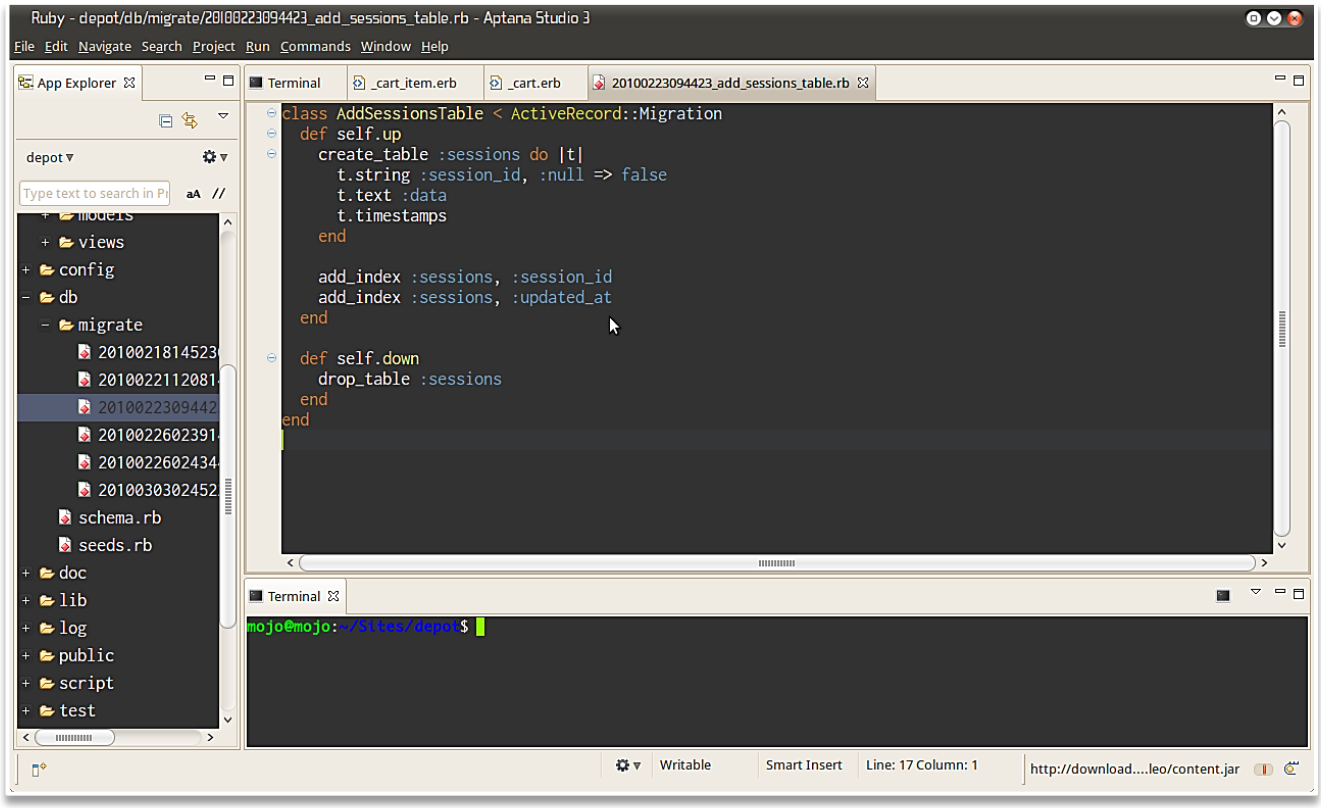

Figura 04 – Aptana Studio

Fonte: https://joneslee85.wordpress.com/2010/03/05/first-impression-with-aptana-studio-3-preview/

Descrição: Tela principal da IDE Aptana Studio 3. No topo, o menu de configuração e edição. Logo abaixo, dividida em duas colunas sendo a da esquerda a lista de projetos, e a coluna da direita as linhas do código fonte do projeto.

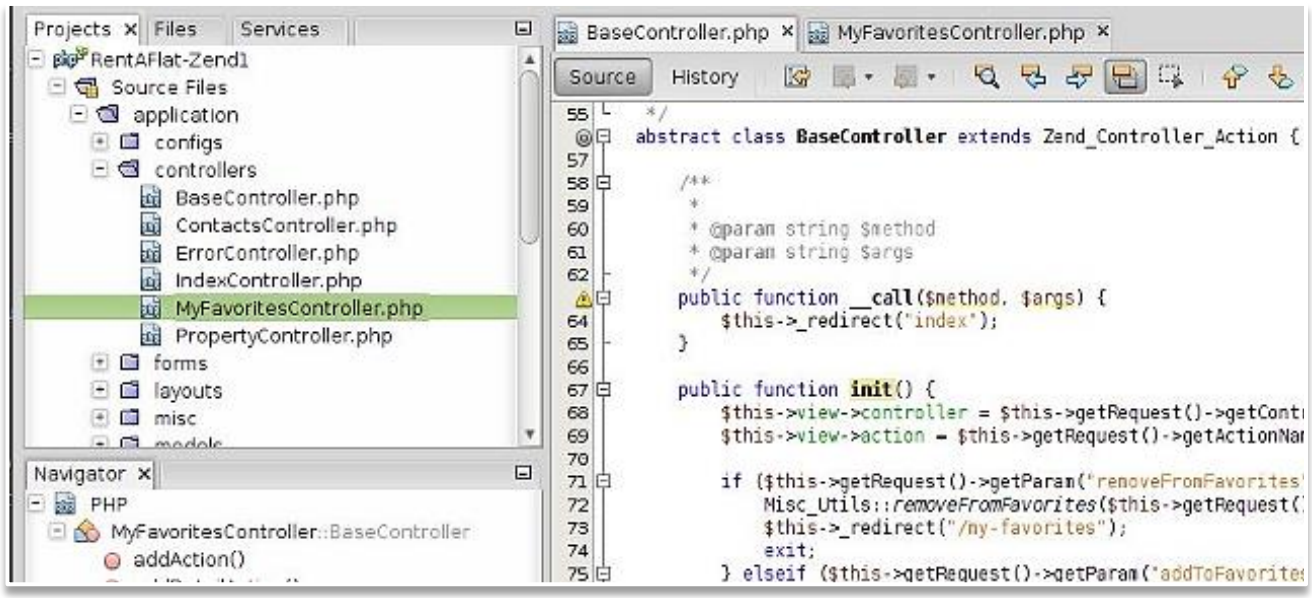

Figura 05 – NetBeans 8.1
Fonte: https://netbeans.org
Descrição: Tela principal da IDE NetBeans 8.1. Dividida em duas colunas sendo a da esquerda a lista de projetos, arquivos e serviços e a coluna da direita, as linhas do código fonte do projeto.

Devemos entender que as IDEs possuem um papel importante no processo de desenvolvimento, mas que neste momento inicial de aprendizagem é interessante que escrevamos os códigos “na mão”, ou seja, usando programas como o bloco de notas.

Ao usar o bloco de notas não dispomos de funções que auxiliam o desenvolvimento, nos obrigando a depurar o código caractere por caractere. Se você utiliza outro sistema operacional como Linux ou macOS, outros programas semelhantes de simples edição de texto estão disponíveis. Para o Linux, sugerimos o gEdit e para o macOS o text-edit.

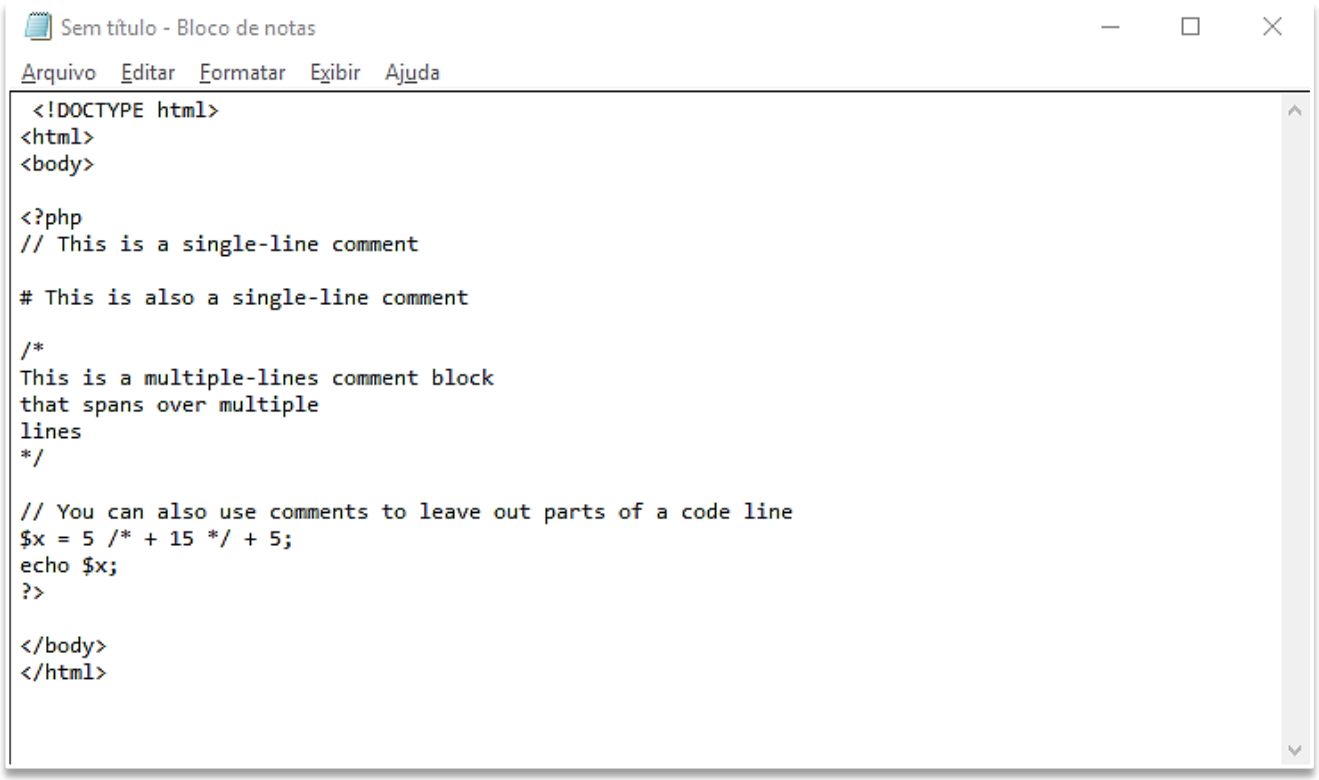

Figura 06 – Bloco de notas
Fonte: autor
Descrição: Tela principal do programa bloco de notas. Na parte superior composta da barra de título e menu. Abaixo, espaço para digitação do código fonte.

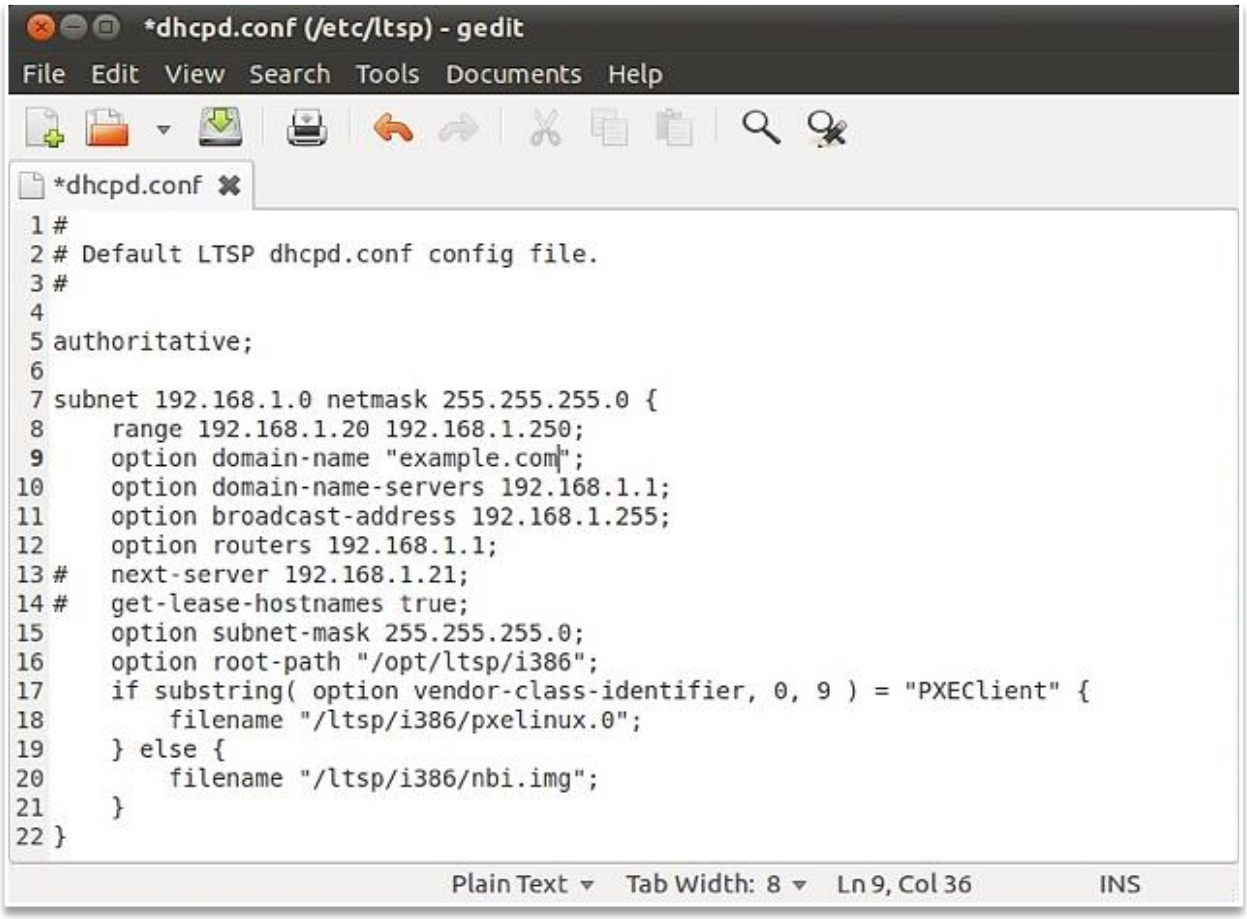

Figura 07 – gEdit
Fonte: https://commons.wikimedia.org/wiki/File:Gedit_screenshot_(en).png
Descrição: Tela principal do programa gEdit. Na parte superior encontra-se a barra de título e menu. Abaixo, espaço para digitação do código fonte.

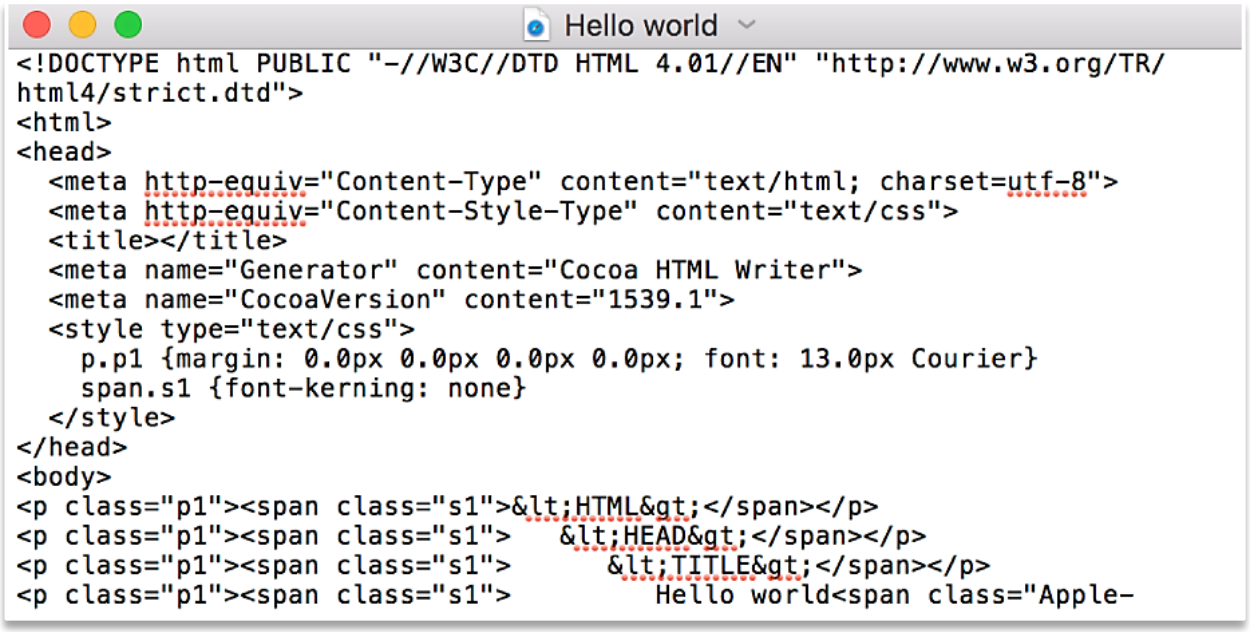

Figura 08 – text-edit
Fonte: https://support.apple.com/pt-br/guide/textedit/welcome/mac
Descrição: Tela principal do programa text-edit. Na parte superior encontra-se a barra de título e logo abaixo, espaço para digitação do código fonte.

Já conhecemos um pouco da história do PHP, suas características e agora sabemos quais programas podemos utilizar para desenvolver os códigos. O próximo passo é configurar o ambiente de testes.

Lembra que no começo do nosso e-book dissemos que o PHP é uma linguagem interpretada? Por ser interpretada, precisamos de um ambiente para “rodar” os códigos que iremos desenvolver ao longo do curso.

Então, mãos-a-obra!

1.6 Configurando o ambiente de testes

Para executar os scripts escritos em PHP, precisamos de um servidor web que contenha o módulo PHP interligado. Este servidor pode ser instalado em nosso próprio computador (servidor local).

Porém, instalar um servidor web manualmente não é uma tarefa fácil. E agora?

Primeiro vamos conhecer como funciona um servidor web com o módulo PHP. O usuário acessa o website (solicitação). Em seguida, o Apache executa o módulo PHP para interpretar o código

e caso necessário, consulta também o banco de dados. Por fim, o servidor gera o resultado em HTML e envia a resposta ao usuário. Veja o esquema na figura 9:

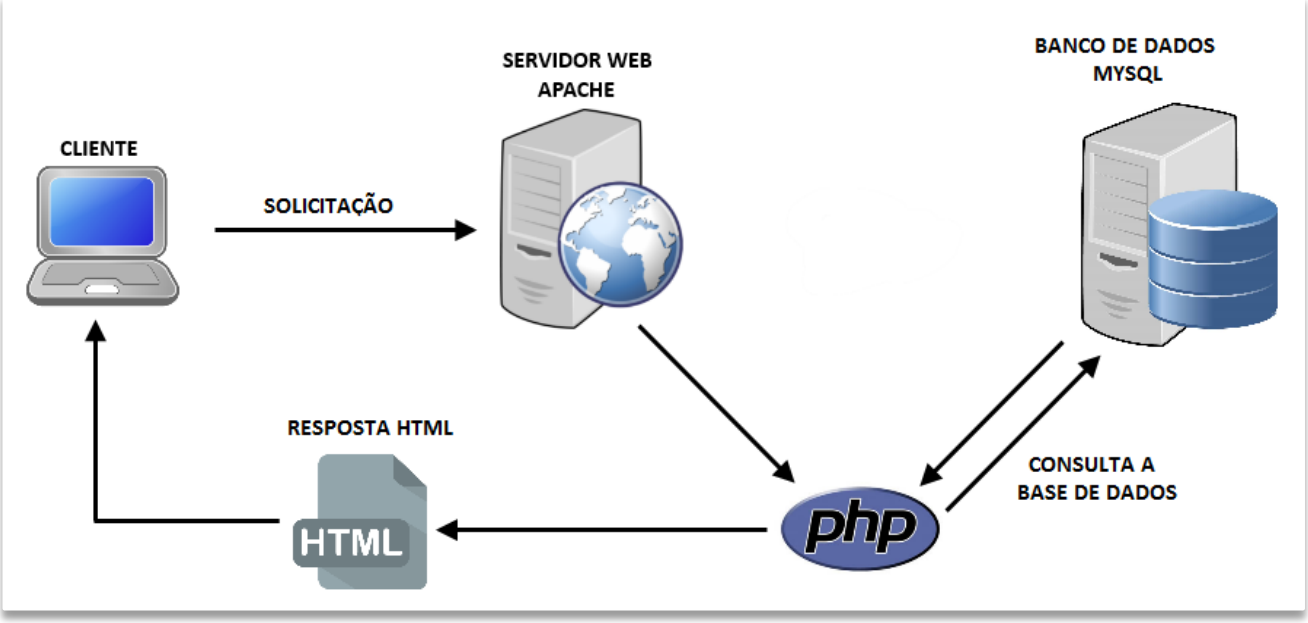

Figura 09 – Esquema de funcionamento do servidor web
Fonte: Adaptado de http://diymakers.es/raspberry-pi-como-servidor-web/
Descrição: No lado esquerdo da imagem, um notebook representa o usuário que faz uma solicitação ao servidor web, este representado pela figura de um computador com um globo. Em seguida, este servidor web executa o módulo PHP representado por uma elipse azul contendo PHP escrito em preto. Por conseguinte, este módulo processa a solicitação em PHP e retorna a resposta em HTML para o usuário.

Entendemos com um servidor web funciona? Vamos agora aprender como configura-lo de maneira simples e fácil.

Graças a Apache Friends, um projeto sem fins lucrativos, temos o XAMPP. O XAMPP é um pacote de softwares que reúne o servidor web Apache, o módulo PHP e Perl e também o banco de dados MySQL. Sua instalação é simples e compatível com diversos sistemas operacionais.

LINK PARA DOWNLOAD XAMPP:

https://sourceforge.net/projects/xampp/files/XAMPP%20Windows/5.6.39/xampp-portable-win32-5.6.39-0-VC11.zip/download

VIDEOAULA!
A configuração do XAMPP é explicada em nossa primeira videoaula, assista.

Chegamos ao fim desta primeira competência. Entendemos neste momento inicial o funcionamento da programação para web focando na linguagem PHP. Aprendemos também como configurar o ambiente necessário para executar nossos scripts.

Agora, acesse o AVA e responda as questões da competência 1.

Ficou com alguma dúvida na competência 1? Acesse o **Fórum - "Competência 1"** para saná-las e discutir com seus colegas sobre os assuntos estudados.

Nosso próximo passo é mergulhar na competência 2, vamos nessa?

2.Competência 02 | Desenvolver uma aplicação para realizar operações matemáticas simples

Nesta competência vamos iniciar o desenvolvimento prático através de operações matemáticas simples. Mas para que possamos iniciar esse nosso voou, alguns conhecimentos prévios serão fundamentais.

Sintaxe, variáveis, operadores, arrays, estruturas de controle condicional e repetição, por exemplo, são familiares? Talvez não, correto? Conhecer estas referências é estritamente necessário para progredir em nossa disciplina e conseguir bons resultados na competência desta semana.

2.1 Sintaxe básica

Cada linguagem de programação possui uma maneira específica de ser escrita. Em PHP, para que a linguagem seja compreendida pelo computador são necessárias duas tarefas. A primeira é que todo o script desenvolvido seja salvo com a extensão .php. A segunda e tão importante quanto é que todo o código escrito seja delimitado pelas tags *<?php* no início e *?>* ao final.

Vejamos o exemplo de sintaxe no arquivo exemplo2.php:

```
<html>
 <head>
  <title>Teste PHP</title>
 </head>
 <body>
    <?php echo "<p>Olá Mundo</p>"; ?>
 </body>
</html>
```

Quadro 02 – Uso correto da sintaxe no arquivo exemplo2.php
Fonte: https://www.php.net/manual/pt_BR/tutorial.firstpage.php
Descrição: Código fonte do arquivo exemplo2.php demonstrando corretamente a sintaxe da linguagem PHP.

No exemplo2, criamos um arquivo chamado exemplo2.php cuja função é exibir na tela a frase Olá Mundo. Em seu código encontramos as tags de início e fim referentes a linguagem PHP que será interpretada pelo servidor web.

Perceba que na construção das páginas web com PHP, códigos HTML também estão inseridos em conjunto e que o resultado final gerado pelo servidor será uma página em HTML. Veja o fluxo de processamento na figura 10, abaixo:

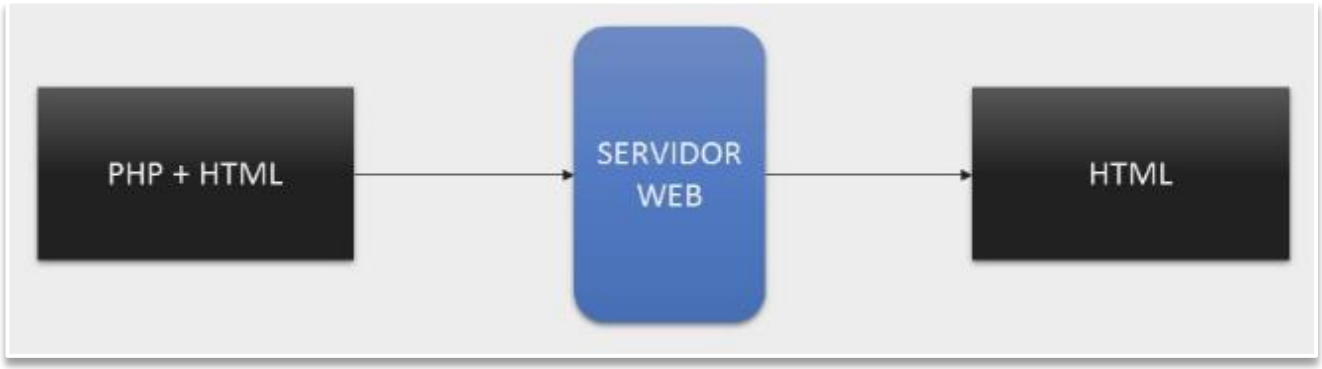

Figura 10 – Fluxo de processamento das páginas em PHP
Fonte: autor
Descrição: Fluxo de processamento das páginas em PHP. No lado esquerdo da imagem um retângulo preto com o termo PHP + HTML inserido. Ao centro, ligado por uma seta, um retângulo vertical com o termo servidor web inserido. No lado direito da imagem, também interligado por uma seta, outro retângulo preto com o termo HTML inserido.

Vale salientar que somente o código presente entre as tags *<?php* e *?>* é interpretado pelo servidor.

SAIBA MAIS!

À medida que vamos escrevendo nossos códigos, centenas de linhas podem compor o projeto. Para organizar, podemos utilizar comentários dentro do próprio script. Aprenda a usar os comentários em: https://www.php.net/manual/pt_BR/language.basic-syntax.comments.php

2.2 Echo

A instrução echo é utilizada em PHP para exibir uma ou mais strings (sequência de caracteres). Veja o exemplo:

```
<html>
 <head>
    <title>Exemplo echo</title>
  </head>
  <body>
    <?php echo "exibindo texto"; ?>
  </body>
 </html>
```

Quadro 03 – Sintaxe da instrução echo no arquivo exemplo3.php
Fonte: autor
Descrição: Código fonte do arquivo exemplo3.php demonstrando corretamente o uso da instrução echo da linguagem PHP.

No exemplo 3 acima, a mensagem que irá aparecer no navegador após a execução da página é exibindo texto, sem aspas. Lembre-se que uso de aspas faz parte da sintaxe da linguagem, ou seja, para que possamos definir o conteúdo da variável como string, o uso de aspas é obrigatório.

Mais adiante em nosso projeto, a instrução echo será bastante utilizada.

2.3 Variáveis

Em linguagem de programação, as variáveis são como gavetas cuja função é guardar dados temporariamente na memória do computador.

Quando falamos especificamente do PHP, algumas características próprias devem ser levadas em consideração, vejamos:

- As variáveis são representadas por cifrão ($);
- O nome da variável deve ser iniciado com uma letra ou sublinhado, seguido de qualquer número de letras, sublinhados ou números e não deve conter espaços;

- Ao contrário de linguagens de programação como Java ou C, por exemplo, não precisamos definir o tipo da variável em PHP, pois esta é uma linguagem de tipagem fraca e dinâmica;
- O PHP distingue caracteres maiúsculos e minúsculos, ou seja, é case sensitive. Portanto, $var, $Var e $VAR são variáveis diferentes;

2.3.1 Tipos

O PHP suporta diversos tipos de dados. Quando declaramos variáveis, o PHP utiliza a checagem de tipos dinâmica, ou seja, uma variável pode conter valores de diferentes tipos durante a execução de um script. Deste modo, podemos afirmar que o PHP é uma linguagem dinamicamente tipada ou ainda, fracamente tipada.

Os tipos de dados suportados são:

- Inteiro – utilizado para números inteiros.
- Ponto flutuante – utilizado para números reais.
- String – usado para strings de caracteres.
- Array – usado para comportar vários itens de dados do mesmo tipo.
- Objeto – específico para armazenar instâncias de classes.
- Booleanos – para valores verdadeiros ou falsos.

Confira em detalhes sobre os tipos de dados em:
https://www.devmedia.com.br/tipos-de-dados-do-php/2556

2.3.2 Declarando variáveis

Usamos o termo declarar para dizer que estamos criando uma variável. Para criar uma variável, devemos respeitar as características da linguagem e atribuir um valor utilizando o sinal de igualdade (=).

Confira o exemplo 4:

```php
<?php
    $var = 'Marcos';
    $Var = 'Helena';
    echo "$var, $Var";      // exibe "Marcos, Helena"

    $4var = 'valor';      // inválido; começa com um número
    $_4var = 'valor';     // válido; começa com um sublinhado
?>
```

Quadro 04 – Declaração de variáveis
Fonte: adaptado de https://www.php.net/manual/pt_BR/language.variables.basics.php
Descrição: Código fonte demonstrando a declaração de variáveis em PHP.

É importante salientar que as variáveis são guardadas temporariamente na memória. A permanência dessa variável é dada pelo seu escopo.

Vamos entender o que é escopo de variáveis?

2.3.3 Escopo de variáveis

Quando declaramos uma variável, esta pertence a um determinado local do código e este local se chama escopo. A maioria das variáveis que criamos pertencem ao escopo local, mas nem sempre é assim. Vamos conhecer os tipos de escopo?

- Escopo local
- Escopo global
- Escopo estático
- Escopo parâmetro

2.3.3.1 Escopo local

Variáveis criadas dentro de uma função, por exemplo, são por padrão introduzidas ao escopo local. Desta maneira seu escopo está restrito, não havendo visibilidade de seu valor fora da função.

SAIBA MAIS!

Funções em PHP são conjuntos de códigos responsáveis por desempenhar uma determinada tarefa. Fazer um cálculo matemático específico é um exemplo de função. Conheça um pouco mais em: https://www.php.net/manual/pt_BR/functions.user-defined.php

Perceba no exemplo 5 que a variável $y está dentro da função teste(). Assim, a instrução echo que neste código está fora da função teste(), não trará nenhum resultado na tela.

```php
<?php
    function teste() { //início da função
        $y = 2 + 4; // variável $y no escopo local
    } //fim da função
    echo $y;
?>
```

Quadro 05 – Declaração de variável em escopo local
Fonte: autor
Descrição: Código fonte demonstrando a declaração de variável em escopo local em PHP.

2.3.3.2 Escopo global

No escopo global as variáveis declaradas têm seus valores disponíveis na memória para todo o script, até o fim de sua execução. Entretanto, para que estas possam ser utilizadas dentro de funções, precisamos “chama-las” usando a palavra-chave *global*. Confira no exemplo 6 abaixo:

```php
<?php
$y = 4; //variável $y declarada globalmente
$z = 2; //variável $z declarada globalmente
    function teste() { //iníco da função
        global $y, $z; // declarando globalmente as variáveis dentro da
função teste()
        $y = $y + $z; //soma das variáveis
    } //fim da função
    teste(); //carrega a função
    echo $y; //exibe o valor de $y que é 6
?>
```

Quadro 06 – Declaração de variável em escopo global
Fonte: autor

Descrição: Código fonte demonstrando a declaração de variável em escopo global em PHP.

2.3.3.3 Escopo estático

O escopo estático, *static* em inglês, é um recurso muito importante na declaração de variáveis. Aprendemos anteriormente que os escopos local e global extinguem as variáveis em algum momento. Para contornar, utilizamos a palavra-chave static ao declarar nossas variáveis.

Conforme o exemplo 7 abaixo, entenda que cada vez que a função teste() for requisitada, haverá um incremento no valor da variável $x, pois a mesma foi declarada como estática. O resultado que irá aparecer na tela após a execução deste script é 123.

```php
<?php
    function teste() { //inicio da função teste
        static $x=1; // variável $x declarada com a palavra-chave static
        echo $x;
        $x++; //operação de incremento
    }
teste();
teste();
teste();
?>
```

Quadro 07 – Declaração de variável estática
Fonte: autor
Descrição: Código fonte demonstrando a declaração de variável estática em PHP.

Vamos fixar o conceito de escopo estático?
Experimente copiar este script do exemplo 7 removendo a palavra-chave static e confira o resultado.

2.4 Operadores

Nesta competência iremos desenvolver uma aplicação simples em PHP capaz de realizar uma ou mais operações matemáticas. Para tanto, precisamos aprender também como funciona os operadores.

Assim como na matemática, o PHP possui operadores responsáveis por receber um ou mais valores e executar alguma ação, como por exemplo, somar.

Vejamos um exemplo:

```php
<?php
$x = 1;
$y = 2;
    echo $x + $y; //exemplo do uso do operador aritmético de soma
?>
```

Quadro 08 – Operador aritmético de adição
Fonte: autor
Descrição: Código fonte demonstrando o uso do operador aritmético de adição.

É óbvio que o PHP não possui apenas este operador. Vamos conhece-los?

2.4.1 Operadores aritméticos

OPERADOR	NOME	RESULTADO
$x + $y	Adição	Soma de $x e $y
$x - $y	Subtração	Diferença de $x e $y
$x * $y	Multiplicação	Produto de $x e $y
$x / $y	Divisão	Quociente de $x e $y
$x % $y	Módulo	Resto de $x dividido por $y
$x ** $y	Exponencial	$x elevado a $y

Tabela 01 – Operadores aritméticos utilizados em PHP
Fonte: autor
Descrição: Tabela composta de três colunas, sendo a primeira a lista dos operadores, a segunda o seu nome e a terceira, o seu resultado.

2.4.2 Operadores de atribuição

OPERADOR	EQUIVALENTE A	DESCRIÇÃO
$x = $y	$x = $y	Atribui a $x o valor de $y
$x .= $y	$x = $x . $y	Concatena as strings $x a $y

$x += $y	$x = $x + $y	Soma $y a $x
$x -= $y	$x = $x - $y	Subtrai $y de $x
$x *= $y	$x = $x * $y	Multiplica $x por $y
$x /= $y	$x = $x / $y	Divide $x por $y
$x %= $y	$x = $x % $y	Atribui a a $x o resto da divisão de $x por $y

Tabela 02 – Operadores de atribuição utilizados em PHP
Fonte: autor
Descrição: Tabela composta de três colunas, sendo a primeira a lista dos operadores, a segunda a sua equivalência aritmética e a terceira, a sua descrição.

2.4.3 Operadores de incremento e decremento

OPERADOR	NOME	DESCRIÇÃO
++$x	Pré-incremento	Incrementa um ao valor de $x e retorna $x
$x++	Pós-incremento	Retorna $x e soma um a $x
--$x	Pré-decremento	Decrementa um ao valor de $x e retorna $x
$x--	Pós-decremento	Retorna $x e subtrai um de $x

Tabela 03 – Operadores de incremento e decremento utilizados em PHP
Fonte: autor
Descrição: Tabela composta de três colunas, sendo a primeira a lista dos operadores, a segunda o seu nome e a terceira, a sua descrição.

2.4.4 Operadores de comparação

OPERADOR	NOME	DESCRIÇÃO
$x == $y	Igual	Verdadeiro se $x for igual a $y
$x != $y	Diferente	Verdadeiro se $x não for igual a $y
$x < $y	Menor que	Verdadeiro se $x for menor que $y
$x > $y	Maior que	Verdadeiro se $x for maior que $y
$x <= $y	Menor ou igual	Verdadeiro se $x for menor ou igual a $y
$x >= $y	Maior ou igual	Verdadeiro se $x for maior ou igual a $y

Tabela 04 – Operadores comparação utilizados em PHP
Fonte: autor

Descrição: Tabela composta de três colunas, sendo a primeira a lista dos operadores, a segunda o seu nome e a terceira, a sua descrição.

2.4.5 Operadores lógicos

OPERADOR	NOME	DESCRIÇÃO
$x and $y	E	Verdadeiro se $x e $y são verdadeiros
$x or $y	OU	Verdadeiro se $x ou $y for verdadeiro
! $x	NÃO	Verdadeiro se $x não for verdadeiro
$x && $y	E	Verdadeiro se $x e $y são verdadeiros
$x || $y	OU	Verdadeiro se $x ou $y for verdadeiro

Tabela 05 – Operadores lógicos utilizados em PHP
Fonte: autor
Descrição: Tabela composta de três colunas, sendo a primeira a lista dos operadores, a segunda o seu nome e a terceira, a sua descrição.

Como visto, o PHP possui uma série de operadores capazes de executar operações específicas como as aritméticas, comparativas, incremento e decremento (unárias) e também lógicas. Entenda também que estes operadores podem ser utilizados em conjunto.

2.5 Arrays

Vimos anteriormente que uma variável é capaz de armazenar um valor por vez. Porém, ao contrário das variáveis, os arrays são capazes de guardar uma lista de dados, relacionados entre chave e valor.

Fazendo uma breve analogia, podemos dizer que as variáveis são gavetas e que os arrays são os armários.

Veja na figura 11 abaixo a exemplificação gráfica de como funciona um array, comparando-o a uma variável:

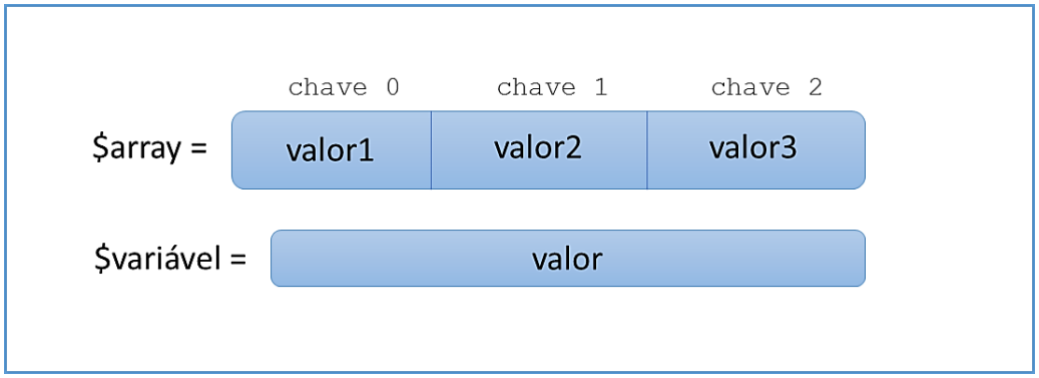

Figura 11 – Definição gráfica do funcionamento de arrays
Fonte: autor
Descrição: Dentro de um retângulo, temos na primeira linha o termo $array seguido de = contendo três valores (valor1, valor2 e valor 3), referenciados respectivamente por chave 0, chave 1 e chave 2. Abaixo, temos o termo $ variável seguido de = contendo um único valor (valor).

Você pode estar se perguntando para que usar arrays, correto? Se você tivesse uma lista com centenas de alunos e precisasse utilizar estes nomes em um script PHP, você iria declarar variáveis com estes nomes uma a uma? Isso seria muito trabalhoso! Com os arrays esta tarefa se torna muito mais simples, veja um exemplo prático:

```php
<?php
    $alunos = array("Joao", "Pedro", "Alberto", "Maria", "Helena");
?>
```

Quadro 09 – Criação de um array simples em PHP
Fonte: autor
Descrição: Código fonte demonstrando a criação de um array simples em PHP.

Em competências futuras do nosso curso, o array será necessário para exibir informações que estão em um banco de dados, por exemplo. Desta maneira, é importantíssimo que o conceito de array fique bastante claro. Se ficou alguma dúvida, confira nossos materiais completares e discuta no fórum!

2.6 Estruturas de controle condicional

As estruturas de controle estão presentes nas linguagens de programação e são responsáveis pelos controles na execução do script. A função básica das estruturas de controle condicional é verificar uma condição.

2.6.1 If

O if é uma estrutura de controle condicional que verifica uma condição e se esta for verdadeira, o comando é executado. Vejamos a sua sintaxe:

```php
<?php
    $nota = 5;
    $media = 6;
        if ($nota < $media) {
            echo "Sua nota está abaixo da média, estude mais um pouco!";
        }
?>
```

Quadro 10 – Estrutura de controle condicional IF
Fonte: autor
Descrição: Código fonte demonstrando a execução da estrutura condicional IF em PHP.

No exemplo 10 acima, temos a estrutura if verificando se a variável $nota é menor que $media. Sendo verdadeiro, o script exibe a mensagem “Sua nota está abaixo da média, estude mais um pouco!” na tela.

2.6.2 If-Else

Na estrutura if-else temos uma espécie de duplo teste. Quando usamos if e else juntos, podemos determinar ações distintas quando o if for falso. Veja a sintaxe:

```php
<?php
    $nota = 10;
    $media = 6;
        if ($nota < $media) {
            echo "Sua nota está abaixo da média, estude mais um pouco!";
```

```
        } else {
            echo "A sua nota está acima da média, parabéns!";
        }
?>
```

Quadro 11 – Estrutura de controle condicional IF-ELSE
Fonte: autor
Descrição: Código fonte demonstrando a execução da estrutura condicional IF-ELSE em PHP.

Se você não percebeu ainda, if significa SE e else significa SENÃO. Portanto, lendo o script do exemplo 11 temos que: " - Se $nota for menor que $media, exiba a mensagem: Sua nota está abaixo da média, estude mais um pouco! Senão, exiba a mensagem: A sua nota está acima da média, parabéns!

2.6.3 Switch

A estrutura de controle condicional switch é semelhante ao uso encadeado de if e else, permitindo verificar uma série de declarações ao mesmo tempo. Vejamos a sua sintaxe:

```
<?php
    $nota = 5;

    switch ($nota) {
        case 0:
            echo "a sua nota é zero";
            break;
        case 1:
            echo "a sua nota é 1";
            break;
        case 5:
            echo "a sua nota é cinco, você está próximo da média";
            break;
        case 10:
            echo "parabéns! Você atingiu a maior nota.";
            break;
    }
?>
```

Quadro 12 – Estrutura de controle condicional SWTICH
Fonte: autor
Descrição: Código fonte demonstrando a execução da estrutura condicional SWTICH em PHP.

Note no exemplo 12 acima que quatro casos foram disponibilizados para a estrutura de controle switch. Caso o valor da variável $nota fosse 0, o comando echo exibe na tela a mensagem “a sua nota é zero”. Caso o valor de $nota fosse 5, o comando echo exibe na tela a mensagem "a sua nota é cinco, você está próximo da média” e assim sucessivamente.

As estruturas de controle condicional possibilitam uma infinidade de maneiras para controlar a execução das tarefas em um script PHP. Nesta seção vimos uma introdução sobre o assunto. Para absorver melhor este conteúdo, pratique! Faça simulações e poste no fórum da competência 2 para possamos discutir em conjunto.

2.7 Estruturas de controle de repetição

O propósito das estruturas de repetição é executar continuadamente determinada instrução. Estas estruturas também são conhecidas como laço ou ainda, loop.

Vamos usar um exemplo para entender melhor?

Um boletim escolar é composto de uma série de informações como notas, disciplinas e médias. Quando criamos um script para gerar este boletim, as estruturas de repetição fazem o trabalho de criar cada linha e coluna do layout de forma automática.

Vamos conhecer essas estruturas?

2.7.1 While

A estrutura while tem a função de repetir uma instrução enquanto uma expressão for verdadeira.

```php
<?php
    $i = 1;
    while ($i <= 10) {
        echo "Esse número é " .$i. "\n";
    $i++;
    }
?>
```

Quadro 13 – Estrutura de controle de repetição WHILE
Fonte: autor
Descrição: Código fonte demonstrando a execução da estrutura de controle de repetição WHILE em PHP.

No exemplo 13 acima temos que enquanto a variável $i for menor ou igual a 10, execute o comando echo e incremente a variável em 1. Esse laço irá se repetir até que a variável $i seja igual a 10.

2.7.2 Do-while

A estrutura do-while tem e função de executar um bloco de comandos inicial e em seguida repete o laço, desde que a condição seja verdadeira. Observe que nesta estrutura a condição é testada após a execução das instruções dentro do laço. Vejamos:

```php
<?php
$x = 0;

do {
    echo "esse numero é " .$x. "\n";
    $x++;
} while ($x <= 4);
?>
```

Quadro 14 – Estrutura de controle de repetição DO-WHILE
Fonte: autor
Descrição: Código fonte demonstrando a execução da estrutura de controle de repetição DO-WHILE em PHP.

No exemplo acima, o script ordena primeiramente que execute o comando echo e incremente a variável $x em um. Em seguida, com o while, é verificado se a condição ($x <= 4) é

verdadeira e então o comando echo é novamente acionado. Esse laço irá ocorrer até que $x seja menor ou igual a quatro.

2.7.3 For

O for é bastante utilizando quando se sabe previamente a quantidade de vezes em que a instrução deve ser executada.

```php
<?php
for ($x = 0; $x <= 5; $x++) {
    echo "Esse numero é: " .$x. "\n";
}
?>
```

Quadro 15 – Estrutura de controle de repetição FOR
Fonte: autor
Descrição: Código fonte demonstrando a execução da estrutura de controle de repetição FOR em PHP.

No exemplo 15, sabemos que a instrução deverá ser executada 5 vezes, portanto, o comando echo será executado até que $x seja menor que ou igual a cinco.

2.7.4 Foreach

O foreach é utilizando apenas para criar laços a partir de arrays. Para cada repetição, o valor do elemento é selecionado e atribuindo e sua respectiva chave relacionada, percorrendo a matriz até o último elemento. Vejamos o exemplo 16 para um melhor entendimento:

```php
<?php
$alunos = array("Pedro", "Helena", "Marcos", "Maria");

foreach ($alunos as $chave => $valor) {
  echo "$chave: $valor \n";
}
?>
```

Quadro 16 – Estrutura de controle de repetição FOREACH
Fonte: autor
Descrição: Código fonte demonstrando a execução da estrutura de controle de repetição FOREACH em PHP.

Entendeu como o foreach funciona?

Neste exemplo, para cada aluno, temos uma chave que o identifica e qual o seu valor (no caso nome). Confira o resultado desse script em: https://ideone.com/PLPKBE

Você percebeu a importância das estruturas de repetição? Elas são indispensáveis para o desenvolvimento de scripts em PHP e por isso, dedique-se e simule os exemplos no seu computador, através do ambiente de testes que configuramos ou através da plataforma online **Ideone**.

Ideone: é uma ferramenta on-line de compilação e depuração que permite compilar código-fonte e executá-lo pelo próprio navegador em mais de 60 linguagens de programação.
https://ideone.com/

2.8 Criando uma aplicação para realizar operações matemáticas simples

Como dito anteriormente, o objetivo desta competência é criar uma aplicação para realizar operações matemáticas simples.

Esta aplicação deverá ser entregue na atividade roteirizada da competência, pelo fórum.

Para norteá-lo, vamos usar alguns temas como exemplo:

- calcular a área de uma circunferência;
- identificar a idade atual de uma pessoa;
- descobrir o resto de uma divisão;

Simule os scripts e discuta com os professores e colegas no **fórum da competência 02**.

Vamos prosseguir? A competência três nos aguarda!

3.Competência 03 | Desenvolver uma aplicação para realizar inclusão de registro em banco de dados

Olá, aluno (a).

Pronto para se aprofundar um pouco mais? Tenho certeza que sim! Nesta competência vamos aprender como o PHP se relaciona com o banco de dados MySQL.

Nos primeiros momentos desta competência iremos explorar e compreender alguns conceitos importantes para manipular banco de dados usando PHP. Estudaremos as funções, os formulários em HTML, as sessões e por fim, a introdução ao banco de dados propriamente dito.

Utilizar banco de dados com PHP não é uma tarefa difícil, basta dedicação a atenção ao conteúdo que será estudado.

Agora vamos iniciar a competência estudando funções.

3.1 Funções

Funções são trechos de programação capazes de realizar instruções específicas. É possível por exemplo exibir automaticamente em maiúsculo a primeira letra de cada palavra de uma *string* usando a função *ucwords()*.

```php
<?php
echo ucwords("seja bem-vindo");
?>
```

Quadro 17 – Uso da função ucwords()
Fonte: autor
Descrição: Código fonte demonstrando o uso da função ucwords() em PHP

O resultado alcançado no exemplo 17 é a exibição do texto “Seja Bem-vindo” na tela.

Outro exemplo a destacar é a função *mysqli_connect()*. Com ela faremos a conexão com o banco de dados MySQL sem dificuldades.

```php
<?php
mysqli_connect("localhost", "usuario", "senha", "base_dados");
?>
```

Quadro 18 – Sintaxe da função mysqli_connect()
Fonte: autor
Descrição: Código fonte demonstrando a sintaxe da função mysqli_connect() em PHP

Mais à frente, na sessão 3.7 desta competência, iremos aprender como fazer na prática esta conexão.

3.2 Funções internas (built-in)

Umas das características mais marcantes da linguagem PHP é quantidade de funções prontas, são centenas. Conectar ao bando de dados, exibir data em diversos formatos, converter letras em maiúsculas, criar arquivos ou diretórios no servidor são alguns exemplos de tarefas das funções PHP.

Saiba também que determinadas funções precisam que extensões específicas estejam compiladas no servidor para que funcione. Algumas funções *image*, responsáveis por uma série de modificações relacionadas a arquivos de imagem, necessitam da biblioteca GD compilada, por exemplo.

Ficou curioso para conhecer a biblioteca GD? Confira na fonte https://www.php.net/manual/pt_BR/book.image.php

A cada lançamento de versões do PHP, novas funções são implementadas, aumentando o leque de tarefas nativas da linguagem.

LISTA DE FUNÇÕES PHP
https://www.php.net/manual/pt_BR/indexes.functions.php

3.3 Formulários HTML

Os formulários HTML são responsáveis por receber dados do usuário e encaminha-los para um *script*. Através deles podemos fazer login em websites, buscar no Google, enviar uma mensagem para um amigo em redes sociais e inserir dados em um banco de dados, por exemplo. O PHP trabalha nativamente com estes formulários, capturando os dados e utilizando-os para diversos fins.

Na figura 12 abaixo, temos a representação gráfica do formulário de acesso ao nosso ambiente de aprendizagem, o AVA. Ao digitar o CPF na identificação do usuário e a senha, estes dados serão encaminhados a um *script* PHP quando clicado em Acessar.

Figura 12 – Formulário de login do AVA
Fonte: https://ead.educacao.pe.gov.br

Descrição: Formulário de login do ava, contendo no topo a palavra acessar. Logo abaixo, o campo identificação do usuário seguido do campo senha.

Quando as informações são enviadas para o *script*, o PHP recebe a informação através de métodos e as manipula de acordo com o que foi programado. Vamos conhecer estes métodos?

3.3.1 Método POST

Dissemos anteriormente que o PHP pode receber dados através de formulários HTML, mas para que essa comunicação ocorra, é necessário utilizar um método. O POST é um dos métodos responsáveis por capturar dados.

Veja no exemplo 19 abaixo a construção básica do código em HTML de um formulário que utiliza o *POST*:

```
<form action="acao.php" method="post">
 <p>Nome: <input type="text" name="nome" /></p>
 <p>Sobrenome: <input type="text" name="sobrenome" /></p>
 <p><input type="submit" name="submit" value="cadastrar" /></p>
</form>
```

Quadro 19 – Código fonte de um formulário em HTML
Fonte: autor
Descrição: Código fonte demonstrando a construção de um formulário em HTML.

Quando criamos o formulário, devemos indicar no parâmetro *method* o seu tipo, neste caso o *post*. Assim, quando clicarmos em cadastrar, o formulário irá enviar os dados dos campos nome e sobrenome para a página acao.php.

Ótimo! Entendemos como funciona este método. Agora vamos aprender como o PHP "captura" os dados deste formulário. Leia o código abaixo:

```
<?php
    $nome = $_POST["nome"]; //captura o nome
    $sobrenome = $_POST["sobrenome"]; //captura o sobrenome

    echo $nome . $sobrenome; //exibe o nome e o sobrenome concatenados
?>
```

Quadro 20 – Código fonte do script PHP acao.php
Fonte: autor
Descrição: Código fonte do script em PHP acao.php demonstrando a captura de dados.

Compreendeu? Neste exemplo temos o script "acao.php", responsável por receber os dados. Para capturar os dados dos campos, usamos o verbo *$_POST[ ]*.

O resultado final desse script é exibir o nome e o sobrenome juntos, ou melhor, concatenados.

Para exercitar e fixar este assunto, copie e cole os códigos e execute-os em seu computador. Lembre-se que para isso é necessário ativar o XAMPP.

3.3.2 Método GET

Além do método POST que já estudamos, o PHP trabalha também com outro método semelhante, o GET. As requisições do tipo GET são muito comuns e estão presentes em grande parte dos websites da internet, com certeza você já deve ter visto o seu funcionamento na prática.

```
http://www.meusite.com.br/pagina.php?nome=maria
```

Quadro 21 – URL encaminhando informações para aplicar o método GET
Fonte: autor
Descrição: Exemplo de uma URL encaminhando a ID para o método GET

Outra informação interessante é que o GET não depende exclusivamente de um formulário para funcionar, podemos enviar as informações através de uma *URL* (Uniform Resource Locator), conforme o exemplo 21.

Adicionalmente, não é interessante utilizar este método para enviar dados confidenciais como senhas, haja visto que a informação será visível a todos na própria URL. Para estes casos é mais viável utilizar o método POST.

Que legal! Até agora entendemos como a informação pode ser enviada, só falta entendermos como o PHP captura as informações utilizando este novo método.

```php
<?php
    $nome = $_GET["nome"]; //captura o nome
    echo $nome; //exibe o nome na tela
?>
```

Quadro 22 – Código fonte do script pagina.php
Fonte: autor
Descrição: Código fonte do script pagina.php demonstrando o método GET.

Conforme o exemplo 22 acima, para o que o PHP capture a informação vinda da URL, basta que o verbo $_GET[ ] seja utilizado.

Desta forma, o script “pagina.php” receberá o valor “maria” da variável “nome” contidos na url do exemplo 21 e exibirá o respectivo nome na tela.

3.4 Sessões

A sessão em PHP é um recurso fundamental na interação do usuário com o site. Ela permite que algumas informações permaneçam salvas ao longo da visitação. Cada sessão é particular e um *cookie* é criado automaticamente no navegador, tendo este uma identificação única.

Um exemplo prático do uso de sessões são os carrinhos de compra das lojas virtuais. Quando você está fazendo uma compra, a lista de produtos fica armazenada em uma sessão. À medida que novos itens são adicionados, estes são inseridos na sessão, enchendo o carrinho.

Figura 13 – Sessão em PHP
Fonte: autor
Descrição: Ao lado esquerdo um avatar representando o usuário e ao lado direito um computador representando o servidor de uma loja virtual. Ao centro, uma linha interligando estes e em sobreposição escrito “sessão do usuário”. Logo abaixo, a figura de um carrinho de compras.

Os carrinhos de compra são apenas um exemplo do que as sessões são capazes de fazer. Outras interações como login de acesso e personalização de cores e estilos nos websites também fazem parte deste contexto.

Neste cenário o PHP mais uma vez facilita o nosso trabalho, disponibilizando funções específicas para criar e manipular sessões. Para começar, devemos declarar no script a sua criação, utilizando a função session_start(). Em seguida, definimos o(s) valor(res) que queremos gravar usando o comando $_SESSION[ ].

```
<?php
    session_start();

    $_SESSION['nome'] = "Roberto";
            if(isset($_SESSION['nome']))
        echo "A sessão " . $_SESSION['nome'] . " existe!";
            else
        echo "A sessão não existe";
?>
```

Quadro 23 – Código fonte de criação de uma sessão em PHP
Fonte: autor
Descrição: Código fonte demonstrando a criação de uma sessão em PHP.

No exemplo 23 acima, além de iniciar a sessão com a função “session_start()” você percebeu o uso da função “isset()”? Esta função tem o papel de verificar se a variável foi iniciada ou não.

3.5 O banco de dados MySQL

O MySQL é um SGBD (Sistema de Gerenciamento de Banco de Dados) importante e muito popular na web. Com ele, milhares de sites e sistemas online registram seus dados, permitindo que suas funções sejam desempenhadas com sucesso.

Assim, grandes corporações como o Facebook, Google, Adobe, Alcatel e tantas outras, confiam neste banco para guardar suas informações (Oracle Corporation and/or its affiliates, 2019).

De toda forma, apesar da sua importância, estudar o MySQL não será o foco principal desta disciplina. Nosso objetivo agora será conhecer um pouco das suas funcionalidades em conjunto com o PHP para que possamos desenvolver as atividades propostas pela ementa do nosso curso.

3.6 Características do MySQL

Atualmente na versão 8.0, podemos destacar algumas das suas principais características de acordo com (Oracle Corporation and/or its affiliates , 2019):

- Escrito em C e C++;
- Suporte a diversas plataformas como Linux, Windows, Solaris, Apple e FreeBSD;
- Armazena vários tipos de dados (https://dev.mysql.com/doc/refman/8.0/en/data-types.html);
- Uso de autenticação criptografada, tornando-o seguro;
- Alta capacidade de armazenamento;

3.7 Conectando ao banco de dados

Para que possamos inserir dados no MySQL é preciso que haja a conexão entre o banco e o script. Esta conectividade ocorrerá usando funções específicas do PHP.

Vamos começar?

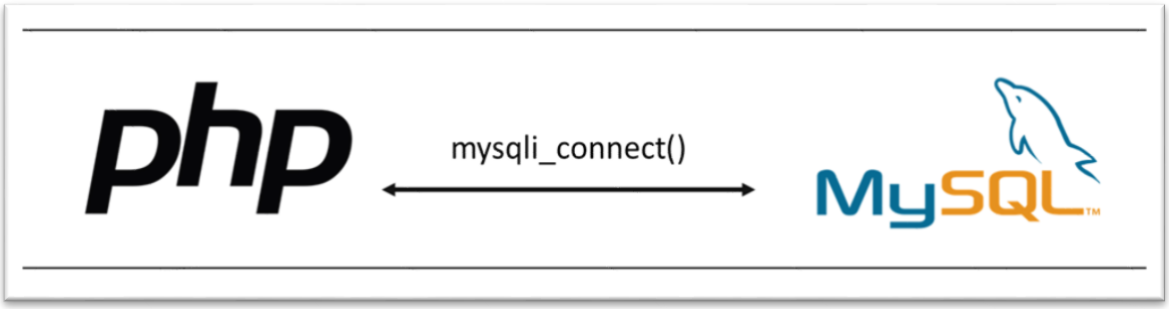

Figura 14 – Conectado o PHP ao Mysql com a função mysqli_connect()
Fonte: autor
Descrição: Ao lado esquerdo a palavra PHP e ao lado direito a palavra MySQL com um golfinho sobreposto. Ao centro, uma seta com a função php mysqli_connect() interligando-os.

O primeiro passo é iniciar a conexão utilizando a função *mysqli_connect()*. Está função possui alguns parâmetros fundamentais que devem ser configurados corretamente para que funcione.

```
<?php
    $link = mysqli_connect("localhost", "root", "", "test");

    //checando a conexão
        if(!$link) {
            printf ("Erro ao conectar ao banco de dados: ",
mysqli_connect_errno());
        }
?>
```

Quadro 24 – Código fonte do script "conectar.php" para conectar ao banco de dados MySQL
Fonte: autor
Descrição: Código fonte demonstrando a conexão com PHP ao banco de dados MySQL.

De acordo com o exemplo 24, temos quatro parâmetros configurados na função mysqli_connect(), sendo eles o host, o usuário, a senha e o banco.

No primeiro parâmetro, identificado por "localhost", definimos o endereço lógico do MySQL. Em nosso caso, por se tratar de uma base local, usamos "localhost". Opcionalmente, podemos usar também o endereço numérico "127.0.0.1".

No segundo parâmetro temos o usuário. Para este, o XAMPP define por padrão o usuário *root*. O root possui todos os privilégios de administrador, permitindo que qualquer tarefa relacionada ao banco de dados seja executada.

A senha, terceiro parâmetro presente na função, deixamos em branco. Sabemos que o ambiente que configuramos no início do curso é de testes, então não há preocupação neste momento como senhas.

O quarto e último parâmetro é o nome do banco de dados. Esta informação é importante pois o MySQL é capaz de armazenar muitas bases ao mesmo tempo, por isso devemos identificar exatamente qual deles iremos usar. Neste caso vamos usar o nome da nossa escola, etepac.

3.8 Incluindo dados

No tópico anterior aprendemos como conectar o PHP ao banco de dados MySQL. Nosso trabalho agora será incluir registros.

Para fazer estas inclusões, utilizamos uma linguagem específica chamada *SQL* (Structured Query Language) junto com a função *mysqli_query()* do PHP. O SQL é uma linguagem padrão para criar, armazenar, manipular bancos e também recuperar dados (W3Schools, 2019).

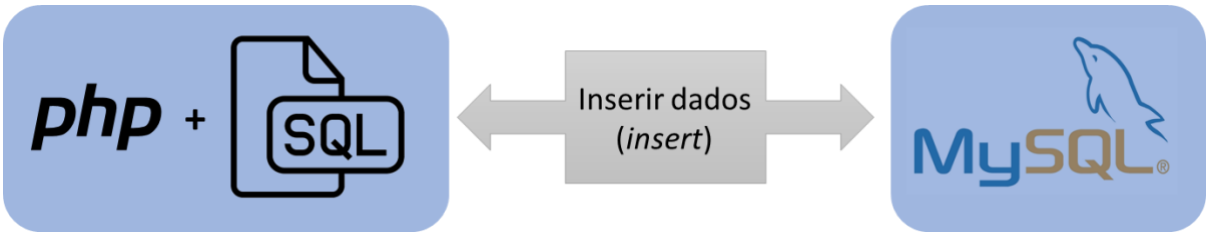

Figura 15 – Linguagem PHP e SQL unidas para manipular um banco de dados MySQL
Fonte: autor
Descrição: Ao lado esquerdo em um retângulo azul, as linguagens PHP e SQL unidas. Ao centro, uma caixa de texto com os dizeres “inserir dados (insert)”. Ao lado direito, o banco de dados MySQL.

Assim como acontece com todas as linguagens de programação, para programar em SQL é preciso conhecer a sua sintaxe básica. Com este entendimento poderemos inserir, consultar, atualizar e deletar informações em um banco de dados MySQL.

Vamos começar com o comando inserir (*INSERT*), pois o objetivo desta competência três é que você aprenda a registrar dados.

```
INSERT INTO tabela (campo1, campo2, campo3...campoN) VALUES (valor1, valor2, valor3...valorN)
```

Quadro 25 – Sintaxe básica do comando INSERT da linguagem SQL
Fonte: autor
Descrição: Sintaxe básica descrevendo o uso do comando INSERT da linguagem SQL.

De acordo como exemplo 25 acima, podemos entender que ao utilizar o comando *insert*, devemos informar o nome da tabela e entre colchetes, os respectivos nomes dos campos. Em seguida, definimos os valores.

Mas o que é tabela?

As tabelas são objetos de um banco de dados, contendo todas as informações. Estas tabelas são organizadas de forma lógica, tendo um formato semelhante ao de uma planilha Excel, em linha e coluna.

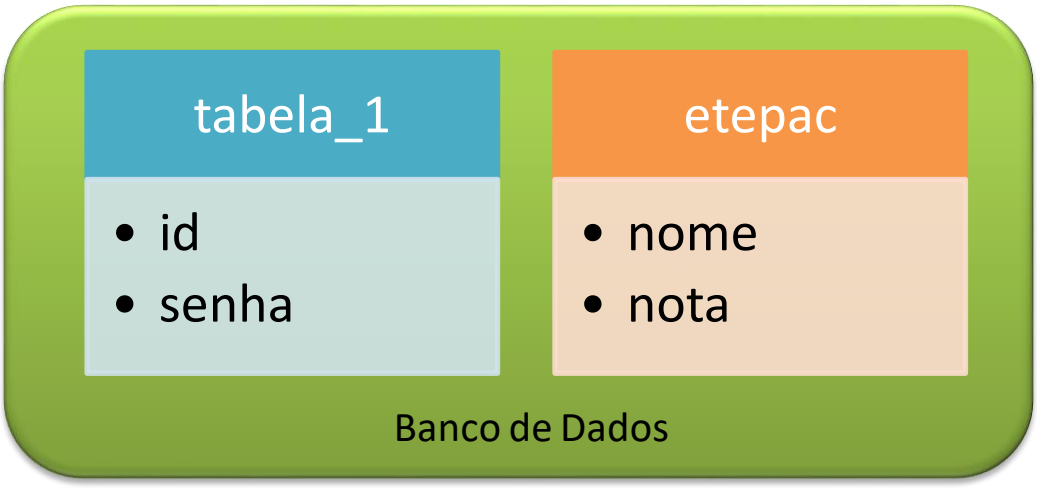

Figura 16 – Tabelas tabela_1 e etepac em um banco de dados
Fonte: autor
Descrição: Retângulo cinza contendo em seu lado esquerdo uma tabela denominada tabela_1 com os campos id e senha. Ao lado direito uma outra tabela denominada etepac com os campos nome e nota.

Muito bom! Já sabemos como conectar ao banco, o que é SQL e o que é tabela, agora podemos desenvolver a nossa aplicação.

3.9 Criando uma aplicação para realizar inclusão de registro em banco de dados

Você lembra que na competência dois foi proposto que a aplicação fosse desenvolvida na atividade roteirizada? Nesta competência não será diferente. Porém, precisaremos de algumas informações preliminares para criá-la.

Esta aplicação, responsável por cadastrar o nome, a disciplina e as notas do aluno é composta de três partes:

- **Parte 1:** construção de um formulário HTML para receber os dados e encaminhar para o script PHP.
- **Parte 2:** criação da tabela etepac no banco de dados MySQL.
- **Parte 3:** desenvolvimento do script PHP com as instruções em SQL para inserir as informações no banco de dados.

Diante do proposto e sabendo que o foco da nossa disciplina é a linguagem PHP, a primeira e segunda parte será desenvolvida pelo professor. Adicionalmente, os arquivos estão livres para personalização. Use sua criatividade e dedicação!

Vamos ao modelo do formulário HTML:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Formulário de cadastro</title>
    <style>
        body { text-align:center; }

        .retangulo {margin: auto; border: 1px solid black; position:
relative;}

        #ret {width: 450px; background-color: georgian;}

        #ret div {
          margin: 20px auto;
          width: 50px;
          height: 50px;
        }
```

```
      </style>
    </head>
      <body>
          <div id="ret" class="retangulo" style="text-align: center;" >
          <br />
          <label id="texto01">CADASTRO</label>
          <br /><br />
          <form action="inserir.php" method="post">
              <p>Nome do Aluno: <input type="text" name="aluno" /></p>
              <p>Disciplina: <input type="text" name="disciplina" /></p>
              <p>Nota1: <input type="text" name="nota1" /></p>
              <p>Nota2: <input type="text" name="nota2" /></p>
              <p>Nota3: <input type="text" name="nota3" /></p>
              <p><input type="submit" name="submit" value="cadastrar" /></p>
          </form>
      </body>
</html>
```

Quadro 26 – Código fonte do formulário modelo em HTML
Fonte: autor
Descrição: Código fonte do formulário modelo em HTML.

Agora, vamos ao código SQL responsável por criar a tabela etepac:

```
CREATE TABLE `test`.`etepac`(
    `id` INT NOT NULL AUTO_INCREMENT,
    `aluno` VARCHAR(50) NOT NULL,
    `disciplina` VARCHAR(20) NOT NULL,
    `nota1` FLOAT NOT NULL,
    `nota2` FLOAT NOT NULL,
    `nota3` FLOAT NOT NULL,
    PRIMARY KEY(`id`)
) ENGINE = INNODB;
```

Quadro 27 – Código SQL
Fonte: autor
Descrição: Código em SQL responsável pela criação da tabela etepac.

Muito bem! Temos a parte um e dois disponíveis, partiremos então para a parte três. Este script PHP é responsável por registrar as informações no banco de dados e será seu papel desenvolvê-lo.

Para este processo precisaremos definir inicialmente cinco variáveis, sendo elas: aluno, disciplina, nota1, nota2 e nota3. Após isso, escolhemos o método para capturar os valores do formulário e em seguida, implementamos a instrução *SQL* com o comando *INSERT*.

Compreendido? Se você ainda não assistiu à nossa videoaula, dê um pulinho no AVA. Na aula, fazemos algumas simulações práticas para que você possa fixar melhor o conteúdo. E lembre-se também do fórum de dúvidas, utilize-o caso necessário.

Chegamos ao fim da competência três e já aprendemos bastante sobre a linguagem de programação para web PHP. Nosso próximo desafio é a competência quatro, onde iremos aprender a manipular os dados de outras formas.

4.Competência 04 | Desenvolver uma aplicação de consulta, alteração e exclusão de um registro em um banco de dados

Bem-vindo a quarta semana do nosso curso!

Na competência anterior aprendemos os primeiros passos de como integrar o PHP com o banco de dados MySQL. Nosso trabalho agora será desenvolver uma aplicação capaz de consultar, alterar e excluir registros.

4.1 Manipulação de dados

Como já sabemos, a manipulação de dados utilizando a linguagem PHP exige conhecimento de algumas funções e instruções em *SQL*. Para consultar, alterar e excluir registros, usamos os comandos *SELECT*, *UPDATE*, e *DELETE* respectivamente.

4.2 Consultando dados

Com o comando *SELECT* podemos realizar pesquisas e colher informações das tabelas do banco de dados. Certamente este comando é uma das mais importantes instruções SQL, pois é com ele que buscamos os dados de forma precisa.

Dentro da instrução SELECT, alguns parâmetros e cláusulas podem ser utilizados para organizar os dados. Temos como exemplo o caractere asterisco “*” cuja a função é selecionar todas as colunas de uma tabela. Outro parâmetro importante é o FROM, responsável por selecionar uma tabela específica em um banco de dados.

Vejamos no exemplo a seguir como é simples utilizar a instrução:

```
SELECT * FROM etepac
```

Quadro 28 – Comando SELECT
Fonte: autor
Descrição: Código em SQL responsável pela seleção de dados na tabela etepac.

Neste exemplo, dissemos através da linguagem SQL que todas as colunas da tabela etepac devem ser selecionadas.

Quer se aprofundar um pouco mais nas cláusulas do comando SELECT?
Acesse: https://becode.com.br/guia-comando-select/

Ótimo! Chegamos a um ponto muito importante da nossa competência, pois agora iremos desenvolver a parte da aplicação que consulta e exibe os dados.

Você lembra que na competência três desenvolvemos uma aplicação que registra dados em um banco de dados? Tenho certeza que sim. Abra novamente esta aplicação e adicione alguns alunos, sua disciplina e suas respectivas notas.

Feito isto, vamos aprender como utilizar uma função PHP muito importante, a *mysqli_fetch_array()*. Esta função tem o papel fundamental de "puxar" os dados e guarda-los em índices associativos, usando o nome dos campos como chave, em uma matriz. Talvez seja importante rever o conceito de arrays, então, siga para a sessão 2.5 deste e-book.

O próximo passo será criar um script que consulte os dados no banco MySQL e retorne o(s) valor(res), exibindo na tela.

Vamos recordar também que o PHP funciona com o HTML embutido, então não teremos dificuldade em gerar uma página que contenha tanto a parte dinâmica, responsável pela manipulação dos dados com o PHP, como também a parte visual, promovida pelo HTML.

Vamos a pratica:

Inicialmente, precisamos definir o layout HTML que irá mostrar as notas.

```
<!DOCTYPE html>
<html>
<head>
<style>
table, th, td {
  border-collapse: collapse;
}
th, td {
```

```
    padding: 5px;
  }
  th {
    text-align: left;
  }
  </style>
  </head>
  <body>

  <h2>Notas</h2>

  <table style="width:100%">
    <tr>
      <th>Nome do Aluno</th>
      <th>Disciplina</th>
      <th>Nota 1</th>
      <th>Nota 2</th>
      <th>Nota 3</th>
    </tr>
    <tr>
      <td>Aluno</td>
      <td>Disciplina</td>
      <td>Nota1</td>
      <td>Nota2</td>
      <td>Nota3</td>
    </tr>
  </table>

  </body>
  </html>
```

Quadro 29 – Página em HTML definindo o layout de exibição das notas
Fonte: autor
Descrição: Código em HTML definindo o layout de exibição das notas.

Este exemplo 29 trata-se apenas dos comandos HTML, ilustrando como as informações serão exibidas na tela. Veja o resultado no navegador:

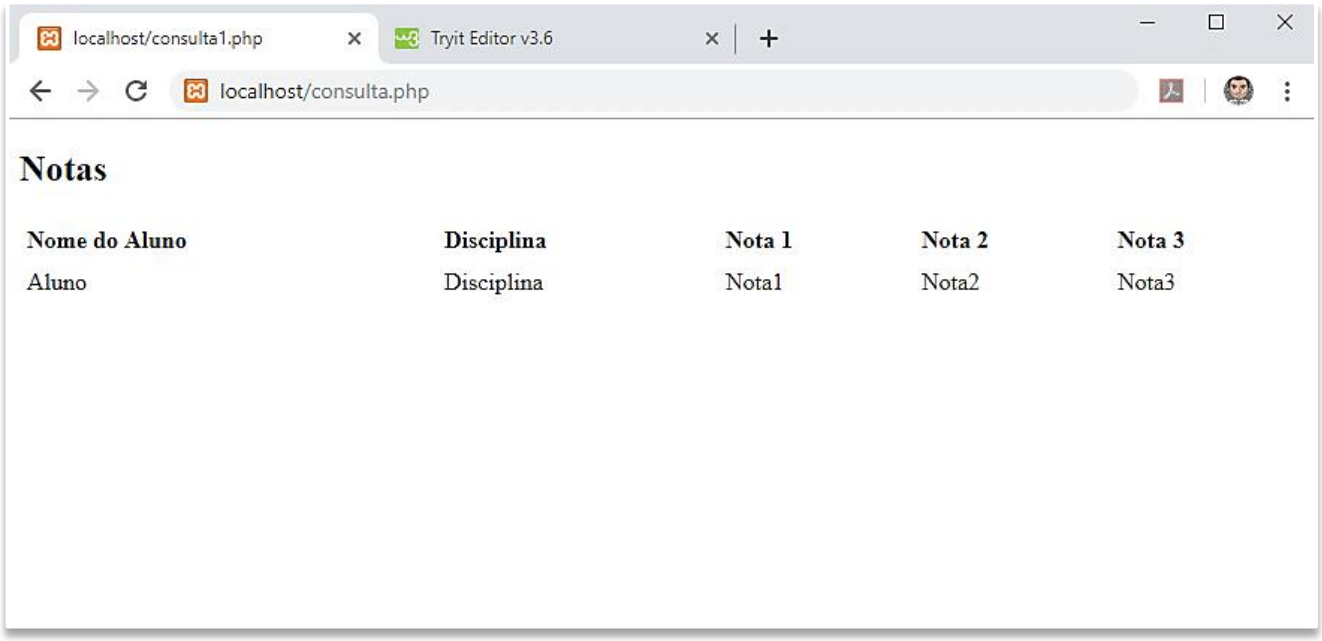

Figura 17 – Página consulta.php apenas com comandos em HTML
Fonte: autor
Descrição: Navegador web representado por um retângulo onde no topo contém o título Notas. Logo abaixo, em linha, estão descritos os termos nome do aluno, disciplina, nota 1, nota 2 e nota 3. Na linha abaixo seguinte, temos os dados aluno, disciplina, nota 1, nota 2 e nota 3.

Na figura 17 temos a representação gráfica exata dos comandos HTML descritos no exemplo 29.

Mas e as informações que registramos anteriormente no banco, como fazemos para efetivamente lista-los? É simples! Tratamos a pouco da função *mysqli_fetch_array()* e é com o auxílio dela que faremos isto acontecer.

Veja como o código fonte do script consulta.php que continha somente comandos HTML ficou, com a adição do PHP:

```
<?php
//fazendo a conexão com o banco de dados test
$link = mysqli_connect("localhost", "root", "", "test");

    //checando a conexão
        if(!$link) {
            printf ("Erro ao conectar ao banco de dados: ",
mysqli_connect_errno());
        }
    //query SQL para selecionar os dados
        $lista = mysqli_query($link, "SELECT * FROM etepac");
?>
<!DOCTYPE html>
```

```
<html>
<head>
<style>
table, th, td {
  border-collapse: collapse;
}
th, td {
  padding: 5px;
}
th {
  text-align: left;
}
</style>
</head>
<body>

<h2>Notas</h2>

<table style="width:100%">
  <tr>
    <th>Nome do Aluno</th>
    <th>Disciplina</th>
    <th>Nota 1</th>
  </tr>
<?php
//instrução de repetição while com a função mysqli_fetch_array
while($linha = mysqli_fetch_array($lista)) {
?>
  <tr>
    <td><?= $linha['aluno'] ?></td>
    <td><?= $linha['disciplina'] ?></td>
    <td><?= $linha['nota1'] ?></td>
  </tr>
<?php
}
?>
</table>

</body>
</html>
```

Quadro 30 – Script consulta.php modificado com os comandos em PHP inseridos
Fonte: autor
Descrição: Código fonte em HTML e PHP responsável por exibir e selecionar os dados em um banco MySQL.

Desta forma, no exemplo 30 se faz a conexão com o banco de dados e em seguida processa o código HTML. Perceba que na parte final do código, onde as linhas são criadas (<tr> </tr>), inicia-se uma instrução de repetição WHILE. Essa instrução significa enquanto. Desta forma, enquanto a função *mysqli_fetch_array()* encontrar registros no banco, ocorre o armazenamento dos dados na variável $linha. Por fim, em cada célula (<td> </td>), inserimos o código PHP para exibir um índice

específico, usando o nome do campo que desejamos. No caso do campo aluno, usamos a expressão "<?= $linha['aluno'] ?>". Veja o resultado final no navegador:

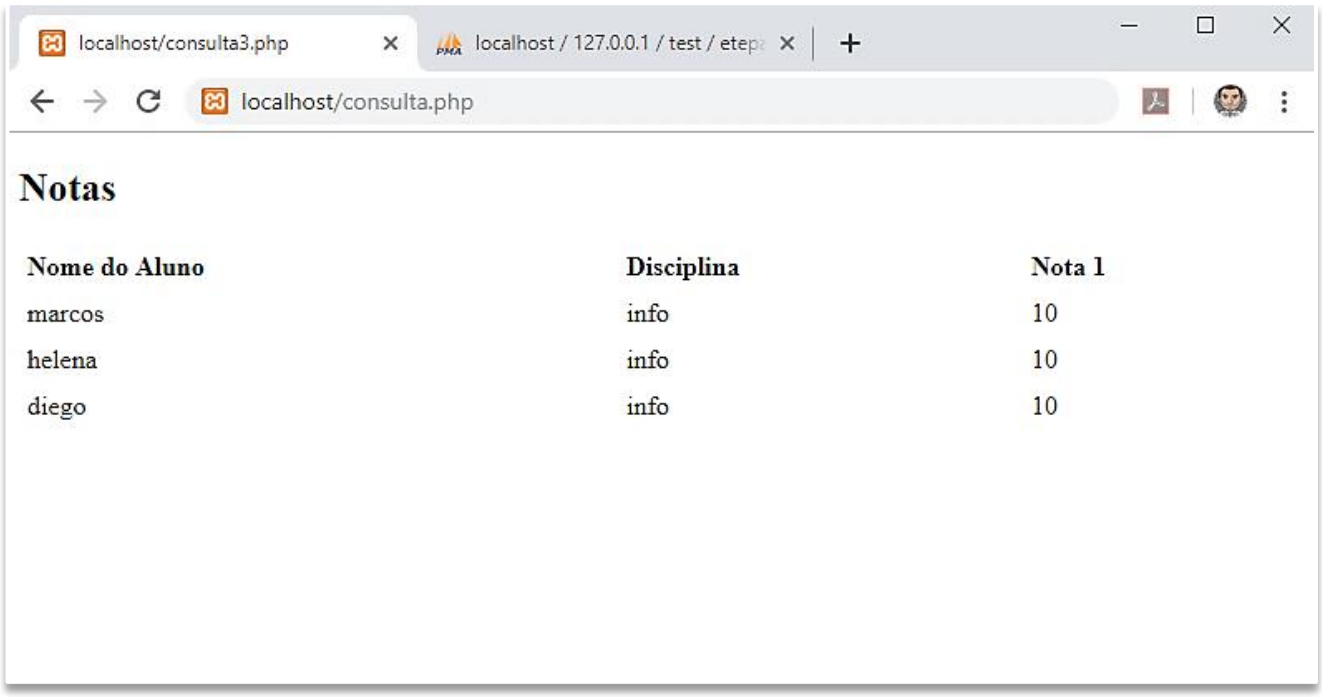

Figura 18 – Página consulta.php apenas com comandos em HTML
Fonte: autor
Descrição: Navegador web representado por um retângulo onde no topo contém o título Notas. Logo abaixo, em linha, estão descritos os termos nome do aluno, disciplina e nota 1. Na linha abaixo seguinte, temos os dados aluno, disciplina e nota 1.

Você percebeu que no código novo do script consulta.php os campos nota 2 e nota 3 foram removidos? Esta ação foi intencional, pois você é quem irá desenvolver esta aplicação por completo. 😊

Mas tenha calma, temos mais dois tópicos para estudar antes de postar a aplicação na atividade roteirizada de número quatro!

Vamos em frente. Agora é a vez de aprender a alterar os dados que já foram registrados anteriormente.

4.3 Alterando dados

O comando *UPDATE* é mais uma instrução de suma importância na linguagem SQL. Com esta, podemos modificar os dados que já foram inseridos no banco.

Se um cadastro foi feito erroneamente, este precisa ser corrigido e será com a instrução UPDATE que fazemos isso.

Sintaxe do *UPDATE*:

```
UPDATE tabela SET coluna1 = valor1, coluna2 = valor2, ... WHERE condição
```

Quadro 31 – Sintaxe do comando UPDATE em SQL
Fonte: autor
Descrição: Sintaxe do comando UPDATE em linguagem SQL

Para usar o UPDATE, devemos definir o nome da tabela e informar o parâmetro SET, seguido das colunas e dos seus respectivos novos valores. Para selecionarmos uma determinada informação no banco, usamos o parâmetro WHERE e expressamos a condição.

Na nossa aplicação, criamos uma coluna chamada “id” no banco de dados e será ela a nossa referência para definir a condição.

Porém, antes de aplicar o comando UPDATE, precisamos recuperar os dados do banco e gerar um link para efetivar essa modificação.

Vamos praticar? Analise o código a seguir:

```
<?php
$link = mysqli_connect("localhost", "root", "", "test");

    //checando a conexão
        if(!$link) {
            printf ("Erro ao conectar ao banco de dados: ",
mysqli_connect_errno());
        }
    //query SQL para selecionar os dados
        $lista = mysqli_query($link, "SELECT * FROM etepac");
?>
<!DOCTYPE html>
<html>
```

```
<head>
<style>
table, th, td {
  border-collapse: collapse;
}
th, td {
  padding: 5px;
}
th {
  text-align: left;
}
</style>
</head>
<body>

<h2>Notas</h2>

<table style="width:100%">
  <tr>
    <th>Nome do Aluno</th>
    <th>Disciplina</th>
    <th>Nota 1</th>
  </tr>
<?php
while($linha = mysqli_fetch_array($lista)) {
?>
  <tr>
    <td><?= $linha['aluno'] ?></td>
    <td><?= $linha['disciplina'] ?></td>
    <td><?= $linha['nota1'] ?></td>
    <td><a href="formularioalterar.php?id=<?= $linha['id'] ?>">Alterar</td>
  </tr>
<?php
}
?>
</table>

</body>
</html>
```

Quadro 32 – Sintaxe do comando UPDATE em SQL
Fonte: autor
Descrição: Sintaxe do comando UPDATE em linguagem SQL

Ao analisar o código do exemplo acima, identificamos uma linha realçada em amarelho. Esta linha é responsável por criar um link para outro script, o formularioalterar.php. Este novo script será responsável por receber os dados que precisam ser modificados e através de um formulário HTML, atualiza-los. Veja o resultado no navegador do exemplo 32:

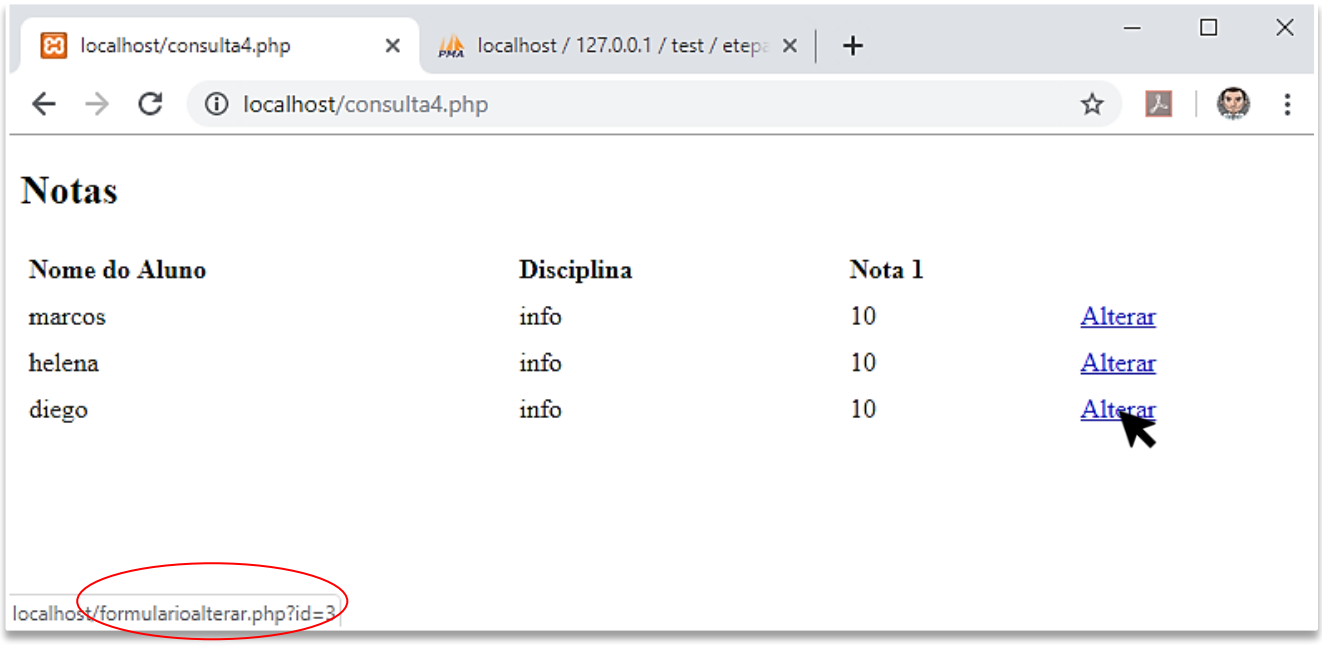

Figura 19 – Página consulta.php com o link para alterar os dados
Fonte: autor
Descrição: Navegador web representado por um retângulo onde no topo contém o título Notas. Logo abaixo, em linha, estão descritos os termos nome do aluno, disciplina e nota 1. Na linha abaixo seguinte, temos os dados aluno, disciplina e nota 1.

Excelente! Agora temos o link e vamos prosseguir criando o formulário (formularioalterar.php) para receber os dados que devem ser corrigidos.

```
<?php
//usando o método GET para capturar o valor do campo id do formulário
$id = $_GET['id'];

//fazendo a conexão
$link = mysqli_connect("localhost", "root", "", "test");

    //checando a conexão
        if(!$link) {
            printf ("Erro ao conectar ao banco de dados: ",
mysqli_connect_errno());
        }
    //query SQL para selecionar os dados
        $lista = mysqli_query($link, "SELECT * FROM etepac WHERE id='$id'");

    //coleta a linha solicitada
        $linha = mysqli_fetch_array($lista);

?>
<!DOCTYPE html>
<html>
```

```
  <head>
    <title>Formulario de cadastro</title>
    <style>
        body { text-align:center; }

        .retangulo {margin: auto; border: 1px solid black; position:
relative;}

        #ret {width: 450px; background-color: georgian;}

        #ret div {
          margin: 20px auto;
          width: 50px;
          height: 50px;
         }
    </style>
  </head>
    <body>
        <div id="ret" class="retangulo" style="text-align: center;" >
        <br />
        <label id="texto01">ATUALIZACAO DE CADASTRO</label>
        <br /><br />
        <form action="alterar.php" method="post">
            <p>Nome do Aluno: <input type="text" name="aluno" value="<?=
$linha['aluno'] ?>" /></p>
            <p>Disciplina: <input type="text" name="disciplina" value="<?=
$linha['disciplina'] ?>" /></p>
            <p>Nota1: <input type="text" name="nota1" value="<?=
$linha['nota1'] ?>" /></p>
            <input type="hidden" name="id" value="<?= $linha['id'] ?>" />
            <p><input type="submit" name="submit" value="alterar" /></p>
        </form>
    </body>
</html>
```

Quadro 33 – Código fonte do script formularioalterar.php
Fonte: autor
Descrição: Código fonte do script formularioalterar.php responsável por receber os dados a serem modificados.

Outra função presente neste formulário é o envio das informações para outro script, o alterar.php, utilizando o método POST. Portanto, quando clicarmos no botão “alterar”, a modificação será concluída.

O resultado do exemplo acima, quando clicamos em alterar a linha do aluno Diego, é:

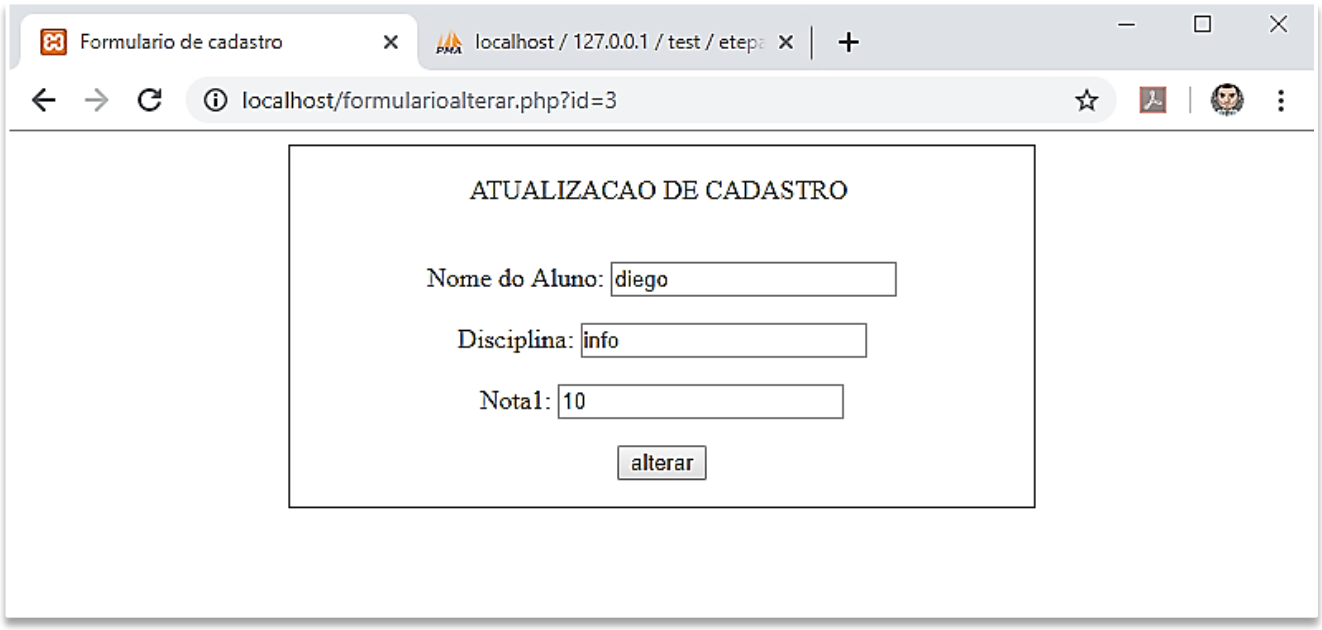

Figura 20 – Formulário formularioalterar.php recebendo os dados do aluno Diego com id = 3.
Fonte: autor
Descrição: Navegador web representado por um retângulo onde no topo contém o título atualização de cadastro. Logo abaixo, os campos nome do aluno, disciplina e nota 1 com os valores diego, info e 10, respectivamente. Sob o ultimo campo, nota 1, o botão alterar.

O próximo passo é confeccionar o script alterar.php, vejamos o seu código fonte:

```php
<?php
    $link = mysqli_connect("localhost", "root", "", "test");

    //checando a conexão
        if(!$link) {
            printf ("Erro ao conectar ao banco de dados: ",
mysqli_connect_errno());
        }
    //definindo as variaveis para receber os dados do formulario
        $id = $_POST['id'];
        $aluno = $_POST['aluno'];
        $disciplina = $_POST['disciplina'];
        $nota1 = $_POST['nota1'];

    //criando a query em SQL para alterar os dados
        $query = "UPDATE etepac SET aluno='$aluno',
disciplina='$disciplina', nota1='$nota1' WHERE id=$id";

    //executando o comando SQL
        mysqli_query($link, $query);

    //exibe mensagem de confirmação
        echo "Dados atualizados!"
?>
```

Quadro 34 – Código fonte do script alterar.php
Fonte: autor
Descrição: Código fonte do script alterar.php responsável por receber os dados e executar a instrução SQL UPDATE.

Como podemos ver, o script alterar.php faz a conexão com o banco de dados, recebe os valores das variáveis através do método POST e em seguida executa a instrução UPDATE, modificando os dados conforme configurado. Ao final, a mensagem “Dados atualizados!” é exibida na tela.

4.4 Excluindo dados

Estamos chegando ao final da competência quatro e o nosso objetivo neste momento é aprender a deletar uma informação existente no banco de dados.

Para esta tarefa, a instrução em SQL “DELETE” será utilizada e a sua sintaxe apresenta-se da seguinte forma:

```
DELETE FROM tabela WHERE condição
```

Exemplo 35 – Sintaxe do comando em SQL DELETE
Fonte: autor
Descrição: Sintaxe do comando DELETE em linguagem SQL.

Assim como no comando UPDATE, será necessário criar um link para deletar a informação. Desta forma, iremos implementar este novo link no script consulta.php já criado anteriormente.

Código fonte do arquivo consulta.php com o novo link excluir:

```
<?php
$link = mysqli_connect("localhost", "root", "", "test");

    //checando a conexão
        if(!$link) {
            printf ("Erro ao conectar ao banco de dados: ",
mysqli_connect_errno());
        }
    //query SQL para selecionar os dados
        $lista = mysqli_query($link, "SELECT * FROM etepac");
?>
```

```
<!DOCTYPE html>
<html>
<head>
<style>
table, th, td {
  border-collapse: collapse;
}
th, td {
  padding: 5px;
}
th {
  text-align: left;
}
</style>
</head>
<body>

<h2>Notas</h2>

<table style="width:100%">
  <tr>
    <th>Nome do Aluno</th>
    <th>Disciplina</th>
    <th>Nota 1</th>
  </tr>
<?php
while($linha = mysqli_fetch_array($lista)) {
?>
  <tr>
    <td><?= $linha['aluno'] ?></td>
    <td><?= $linha['disciplina'] ?></td>
    <td><?= $linha['nota1'] ?></td>
    <td><a href="formularioalterar.php?id=<?= $linha['id'] ?>">Alterar</td>
    <td><a href="excluir.php?id=<?= $linha['id'] ?>">Excluir</td>
  </tr>
<?php
}
?>
</table>

</body>
</html>
```

Quadro 36 – Script consulta.php com o link excluir implementado
Fonte: autor
Descrição: Código fonte do arquivo consulta.php com o novo link excluir implementado.

Lendo o código acima, identifica-se na linha realçada em amarelo a presença do link "Excluir", encaminhando a tarefa para o script excluir.php.

O resultado que temos agora do script consulta.php é:

Figura 21 – Página consulta.php com o novo link excluir
Fonte: autor
Descrição: Navegador web representado por um retângulo onde no topo contém o título Notas. Logo abaixo, em linha, estão descritos os termos nome do aluno, disciplina e nota 1. Na linha abaixo seguinte, temos os dados aluno, disciplina e nota 1.

O próximo passo é desenvolver o código do arquivo excluir.php. Vejamos:

```
<?php
    $link = mysqli_connect("localhost", "root", "", "test");

    //checando a conexão
        if(!$link) {
            printf ("Erro ao conectar ao banco de dados: ",
mysqli_connect_errno());
        }
    //definindo a variavel id utilizando o método GET
        $id = $_GET['id'];

    //criando a query em SQL para deletar os dados
        $query = "DELETE FROM etepac WHERE id=$id";

    //executando o comando SQL
        mysqli_query($link, $query);

    //exibe mensagem de confirmação
        echo "Dados excluidos com sucesso!"
?>
```

Quadro 37 – Script excluir.php
Fonte: autor

Descrição: Código fonte do script excluir.php responsável por apagar os dados em um banco utilizando o comando DELETE em linguagem SQL.

Ao analisar o código do exemplo 37, encontramos o comando DELETE sendo utilizado na *query SQL*, executando a tarefa de excluir a informação no banco de dados MySQL.

Finalmente, após muitos códigos e conceitos, chegamos ao fim da competência quatro. Estudamos nesta sessão as maneiras de como inserir, consultar, alterar e apagar dados.

Desta forma, fique atento a atividade roteirizada proposta para esta semana e a desenvolva como se pede.

Se você ficou com dúvidas, acesse o **fórum da competência 4** e faça a sua pergunta.

5.Competência 05| Projeto: Desenvolver uma aplicação para emissão de um relatório a partir de um banco de dados.

Bem-vindo a última semana da nossa disciplina!

Nesta competência vamos aprender como gerar relatórios em PDF utilizando uma biblioteca PHP específica, a FPDF.

Bibliotecas são conjuntos de funções prontas. Este recurso é muito utilizado para agilizar o desenvolvimento de ações específicas em um website, como por exemplo, a emissão de relatórios em PDF.

Os relatórios têm a importante função de organizar os dados em um layout, facilitando o entendimento das informações. Compostos por imagens, gráficos, palavras, linhas e colunas, estes documentos constituem uma fonte organizada de dados.

Sendo assim, o nosso objetivo específico será gerar um boletim de notas em PDF, usando os dados que registramos anteriormente no banco MySQL.

5.1 O que é PDF?

Portable Document Format (PDF) é um formato de arquivo usado para exibir e compartilhar documentos com segurança, independentemente do software, do hardware ou do sistema operacional. Inventado pela Adobe, o PDF agora é um padrão aberto mantido pela *International Organization for Standardization* (ISO). Estes documentos podem conter links e botões, campos de formulário, áudio, vídeo e lógica de negócios.

5.2 A biblioteca FPDF

FPDF é uma biblioteca livre e de código aberto escrita em PHP que permite gerar arquivos no formato PDF.

São algumas características desta biblioteca:

- Permite escolha da unidade de medida, formato da página e margens
- Gerenciamento de cabeçalho e rodapé de página
- Quebra de página automática
- Quebra automática de linha e justificação de texto
- Suporte de imagens (JPEG, PNG e GIF)

Para utilizar a FPDF precisaremos antes de mais nada, fazer o seu download no link http://www.fpdf.org/en/dl.php?v=181&f=zip.

Ao concluir o download, descompacte os arquivos na pasta onde está o projeto que estamos trabalhando desde o começo da disciplina.

5.3 Gerando relatório em PDF com PHP e MySQL

Com os arquivos descompactados no devido lugar, vamos começar a desenvolver o script PHP capaz de gerar boletins em PDF.

O primeiro passo é criar um link redirecionando para o script criapdf.php. Vamos mais uma vez utilizar o arquivo consulta.php que já conhecemos.

```
<?php
$link = mysqli_connect("localhost", "root", "", "test");

    //checando a conexão
        if(!$link) {
            printf ("Erro ao conectar ao banco de dados: ",
mysqli_connect_errno());
        }
```

```
    //query SQL para selecionar os dados
        $lista = mysqli_query($link, "SELECT * FROM etepac");
?>
<!DOCTYPE html>
<html>
<head>
<style>
table, th, td {
  border-collapse: collapse;
}
th, td {
  padding: 5px;
}
th {
  text-align: left;
}
</style>
</head>
<body>

<h2>Notas</h2>

<table style="width:100%">
  <tr>
    <th>Nome do Aluno</th>
    <th>Disciplina</th>
    <th>Nota 1</th>
  </tr>
<?php
while($linha = mysqli_fetch_array($lista)) {
?>
  <tr>
    <td><?= $linha['aluno'] ?></td>
    <td><?= $linha['disciplina'] ?></td>
    <td><?= $linha['nota1'] ?></td>
    <td><a href="formularioalterar.php?id=<?= $linha['id'] ?>">Alterar</td>
    <td><a href="excluir.php?id=<?= $linha['id'] ?>">Excluir</td>
    <td><a href="criapdf.php?id=<?= $linha['id'] ?>">Gerar boletim</td>
  </tr>
<?php
}
?>
</table>

</body>
</html>
```

Quadro 38 – Script consulta.php com o link gerar boletim adicionado
Fonte: autor
Descrição: Código fonte do arquivo consulta.php com o novo link gerar boletim implementado.

Diante do código acima, a linha hachurada em amarelo demonstra a criação do link que encaminha a requisição ao script criapdf.php, responsável por gerar o documento em questão. Mais uma vez utilizamos o campo “id” como referência na consulta ao banco de dados.

Veja o resultado na tela após esta modificação:

Figura 22 – Página consulta.php com o novo link gerar boletim
Fonte: autor
Descrição: Navegador web representado por um retângulo onde no topo contém o título Notas. Logo abaixo, em linha, estão descritos os termos nome do aluno, disciplina e nota 1. Na linha abaixo seguinte, temos os dados aluno, disciplina e nota 1 e em sequência os alunos marcos, helena e ivo.

Com o novo recurso adicionado, o próximo e último passo é desenvolver o script criapdf.php

Acompanhe o código:

```php
<?php
//informa ao script que o arquivo fpdf.php é requerido
require 'fpdf/fpdf.php';

        $pdf = new FPDF('P','pt','A4');
        $pdf->AddPage();
        $pdf->Image('imagens/logo.jpg');
        $pdf->SetFont('arial','',12);
        $pdf->Cell(0,30,"BOLETIM DE NOTAS",0,1,'L');
        $pdf->Cell(0,1,"","B",1,'C');
        $pdf->ln(20);

//conecta ao banco de dados
$link = mysqli_connect("localhost", "root", "", "test");

    //checando a conexão
    if(!$link) {
```

```
        printf ("Erro ao conectar ao banco de dados: ",
mysqli_connect_errno());
    }
    //definindo a variável id
    $id = $_GET['id'];

    //criando a query em SQL para selecionar os dados
    $lista = mysqli_query($link, "SELECT * FROM etepac WHERE id=$id");


    //início do da estrutura de repetição
    while ($linha = mysqli_fetch_array($lista)) {
        $pdf->SetFont('arial','B',12);
        $pdf->Cell(30,20,"ID: ",0,0,'L');
        $pdf->setFont('arial','',12);
        $pdf->Cell(0,20,$linha['id'],0,1,'L');

        $pdf->SetFont('arial','B',12);
        $pdf->Cell(100,20,"Nome do Aluno: ",0,0,'L');
        $pdf->setFont('arial','',12);
        $pdf->Cell(0,20,$linha['aluno'],0,1,'L');

        $pdf->SetFont('arial','B',12);
        $pdf->Cell(100,20,"Disciplina: ",0,0,'L');
        $pdf->setFont('arial','',12);
        $pdf->Cell(0,20,$linha['disciplina'],0,1,'L');

        $pdf->Ln(20);

        $pdf->SetFont('arial','B',12);
        $pdf->Cell(100,20,"NOTA 1: ",0,0,'L');
        $pdf->setFont('arial','',12);
        $pdf->Cell(0,20,$linha['nota1'],0,1,'L');
}


    //gerando o arquivo boletim.pdf
    $pdf->Output('boletim.pdf', 'I');
?>
```

Quadro 39 – Script criapdf.php
Fonte: autor
Descrição: Código fonte do arquivo criapdf.php responsável por gerar o boletim em PDF.

Uma das novidades apresentadas no exemplo acima é a declaração *require*. Esta função tem por objetivo informar ao servidor PHP que o arquivo mencionado é necessário para que o resto do script funcione. Caso este arquivo não esteja presente ou ainda com a nomenclatura errada, o processo será interrompido e uma mensagem de erro será mostrada, conforme a imagem abaixo:

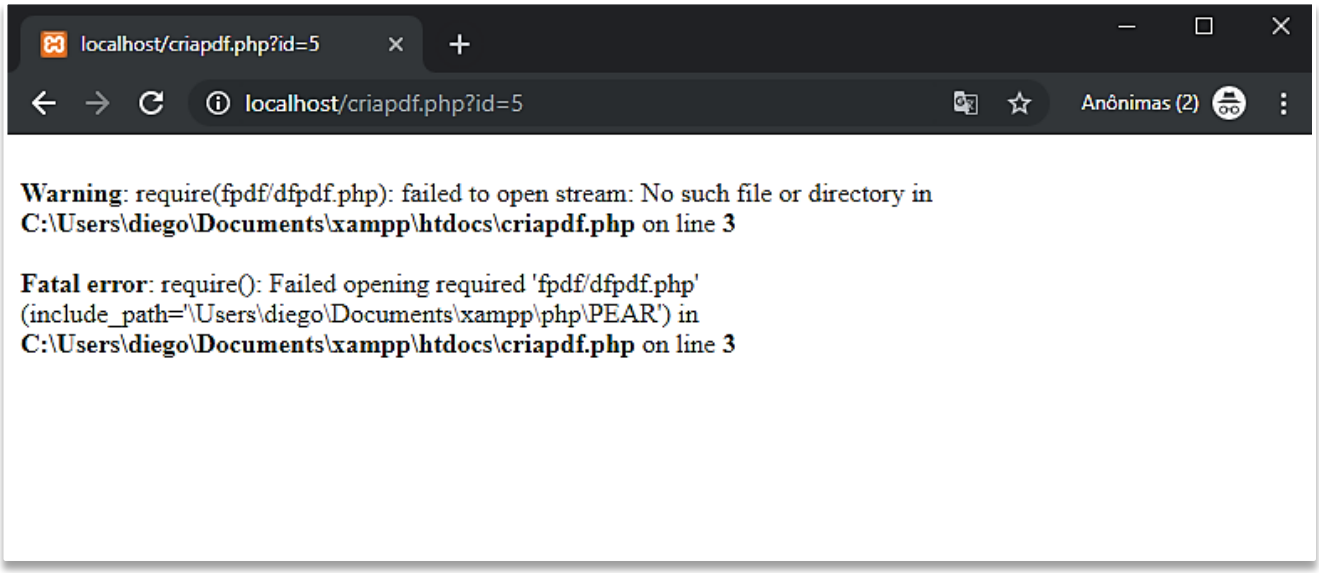

Figura 23 – Mensagem de erro no uso incorreto da declaração *require*
Fonte: autor
Descrição: Navegador web representado por um retângulo informando dois erros. O primeiro é: "Warning: require(fpdf/dfpdf.php): failed to open stream: No such file or directory in C:\Users\diego\Documents\xampp\htdocs\criapdf.php on line 3" e o segundo é: "Fatal error: require(): Failed opening required 'fpdf/dfpdf.php' (include_path='\Users\diego\Documents\xampp\php\PEAR') in C:\Users\diego\Documents\xampp\htdocs\criapdf.php on line 3".

Você conseguiu identificar o motivo pelo qual este erro ocorreu? Caso não, perceba que o arquivo chamado foi "dfpdf.php" e não "fpdf.php" que é o correto.

Dando continuidade na leitura do código, temos as linhas referentes a criação do nosso boletim em PDF.

Vamos entender como funciona estas linhas:

- `$pdf = new FPDF('P','pt','A4');`

  esta linha é responsável por construir a classe, definindo o formato da página, a unidade de medida e o tamanho da página. Os parâmetros 'P', 'pt' e 'A4' significam respectivamente retrato, pontos e o formato A4.

- `$pdf->Image('imagens/logo.jpg');`

  é a linha destinada para colocar uma imagem na página.

- `$pdf->SetFont('arial','',12);`

define a fonte que será usada para imprimir a informação. O primeiro parâmetro define a família da fonte, o segundo o estilo de fonte e o terceiro, o seu tamanho.

- `$pdf->Cell(30,20,"ID: ",0,0,'L');`

  tem a função de imprimir uma célula na página. Os parâmetros 30, 20, "ID: ", 0, 0 e 'L' são respectivamente largura, altura, texto a ser impresso, borda, posição e alinhamento do texto.

- `$pdf->Ln(20);`

  efetua a quebra de linha. O valor 20 é um parâmetro que define a altura desta quebra.

- `$pdf->Output('boletim.pdf', 'I');`

  gera o documento final com o nome “boletim.pdf” e com o parâmetro ‘I’, abre automaticamente no navegador, caso um plugin para leitura de pdf esteja instalado.

Para conhecer todas as funções presentes na biblioteca FPDF, acesse o manual em português:
http://www.fpdf.org/en/dl.php?id=139

Agora que sabemos com estas linhas de código funcionam, vamos acessar o script consulta.php e gerar o boletim da aluna Helena, clicando no link “Gerar boletim”.

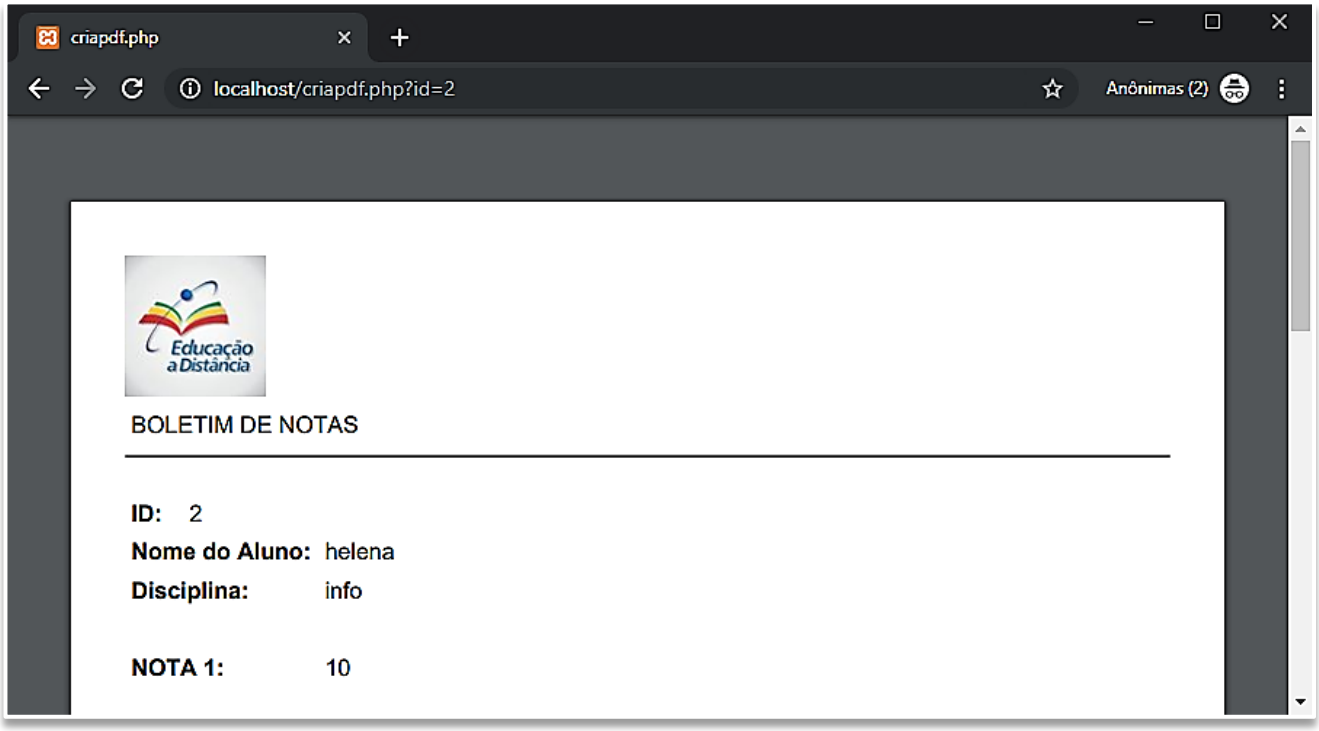

Figura 24 – Boletim da aluna Helena
Fonte: autor
Descrição: Folha branca com uma imagem de um livro colorido e o texto “educação à distância” em azul. Logo abaixo desta imagem, o texto boletim de notas em maiúsculo e uma linha horizontal cinza. Abaixo desta da linha, os campos ID, nome do aluno, disciplina e nota 1, com as respectivas informações: 2, helena, info e 10.

Diante deste exemplo, finalizamos a nossa competência cinco.

Em caso de dúvidas, acesse o **fórum da competência 5** e poste a sua pergunta.
Também não esqueça da nossa atividade roteirizada.

Conclusão

Prezado Aluno,

Nestas cinco semanas de aprendizado, tivemos a oportunidade de estudar os conceitos fundamentais sobre a linguagem de programação para web PHP. Vimos como a web funcionava e passamos a entender como funciona nos dia de hoje.

Diante disso, é muito importante entender que os estudos relacionados nesta disciplina não acabam por aqui. A linguagem PHP é repleta de conceitos e funções, requerendo do profissional muito tempo e dedicação. Não esqueça também que a web sofre mudanças muito rapidamente, exigindo do programador atualizações constantes.

Deste modo, as cinco competências de linguagem de programação para web tem o papel de ingressa-lo tecnicamente neste fabuloso ambiente, apresentando de forma didática um conteúdo criado para que possa contribuir com o início ou aperfeiçoamento da sua carreira em informática, aplicada ao desenvolvimento de aplicações para a web.

Desejo-lhe muito sucesso e até uma próxima oportunidade.

Referências

ADOBE. O que é PDF. **Site da Adobe**, 2019. Disponivel em: <https://acrobat.adobe.com/br/pt/acrobat/about-adobe-pdf.html>. Acesso em: 10 Julho 2019.

DEVMEDIA. Tipos de dados do PHP. **Site da Devmedia**, 2012. Disponivel em: <https://www.devmedia.com.br/tipos-de-dados-do-php/25566>. Acesso em: 25 Junho 2019.

ORACLE CORPORATION AND/OR ITS AFFILIATES. www.mysql.com. **Mysql,** 13 Agosto 2019. Disponivel em: <https://www.mysql.com/why-mysql/>. Acesso em: 13 Agosto 2019.

SILVA, M. N. P. D. Sistema de Numeração Binária. **Brasil Escola**. Disponivel em: <https://brasilescola.uol.com.br/matematica/sistema-numeracao-binaria.htm>. Acesso em: 14 Agosto 2019.

THE PHP GROUP. Sobre o PHP. **Site do PHP**, 2019. Disponivel em: <https://www.php.net/manual/pt_BR/intro-whatis.php>. Acesso em: 04 Julho 2019.

WELLING, L.; THOMSON, L. **PHP e Mysql - Desenvolvimento Web**. 3ª. ed. Rio da Janeiro: Elsevier, v. II, 2005.

FPDF Library. **Site da biblioteca FPDF**, 2019. Disponível em: < http://www.fpdf.org/>. Acesso em: 23 de Agosto de 2019.

Minicurrículo do Professor

Witalo Diego Matias Nunes

Formado pela Universidade do Sul de Santa Catarina - UNISUL em graduação em Gestão da Tecnologia da Informação e técnico em Redes e Sistemas Operacionais pelo Ibratec, é professor da Escola Técnica Estadual Professor Antônio Carlos Gomes - ETEPAC do curso de Desenvolvimento de Sistemas EAD.

Possui experiência como consultor em informática deste 1999, atuando nos campos de redes, sistemas operacionais e help desk.