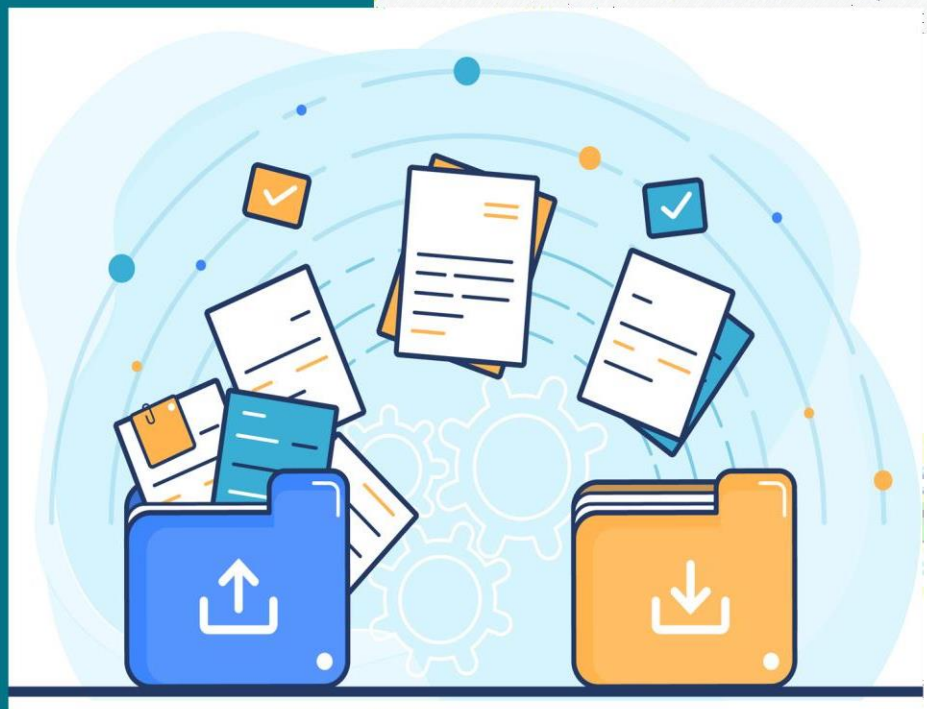

AGENDA 8

PHP:
IMPORTAÇÃO
DE ARQUIVOS,
VARIÁVEIS DE
SESSÃO, COOKIES
E HOSPEDAGEM



GEEaD - Grupo de Estudos de Educação a Distância
Centro de Educação Tecnológica Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO
EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO
CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS
PROGRAMAÇÃO MOBILE I

Expediente

Autor:

Paulo Eduardo Cardoso Andrade

Revisão Técnica:

Eliana Cristina Nogueira Barion

Revisão Gramatical:

Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação: Flávio Biazim



MERGULHANDO
NO TEMA...

Importação de Arquivo (Require e Include)

No PHP, há disponível quatro funções com o objetivo de importar arquivos, e cada uma exerce uma tarefa específica. As quatro funções são: `include()`, `include_once()`, `require()` ou `require_once()`. As funções que contém a palavra “once” evita a importação de um arquivo que já tenha sido importado anteriormente, em alguma etapa do projeto.

Obs: Um dos grandes problemas em importar um arquivo que já está importado, é que as variáveis contidas no escopo desse arquivo serão resetadas, assim, os valores contidos nas variáveis já definidas em outro arquivo que já utilizou a importação, serão perdidos. Isso é somente um dos problemas, podemos encontrar outros também, no entanto, em alguns casos esse problema é utilizado como solução.

As funções com nome `include` do PHP têm como objetivo adicionar um arquivo dentro de outro quando acessado. Em qualquer problema encontrado na inclusão, será apresentado um aviso (Warning). Esse aviso vai indicar que não foi possível realizar sua inclusão e exibirá a página normalmente (sem a inclusão do arquivo). As funções com nome `require` PHP tem o mesmo objetivo das funções `include`, a diferença está, em situações em que o arquivo que está sendo incluído não exista ou não seja encontrado, um Fatal Error (erro fatal) é gerado, interrompendo a execução de qualquer codificação que venha após a linha da função `require`.

Obs: outra divergência é o fato das funções `require` não aceitarem parâmetros via GET para o arquivo chamado. Caso você utilize este parâmetro, ele será ignorado. Já, as funções de nome `include`, aceitam parâmetros via GET.

Com os conceitos em mente, vamos começar! Para exemplificar, vamos fazer atualização do projeto da agenda 6 – Lista de Amigos.

Inicialmente vamos criar uma página de login, para ser possível restringir o acesso aos dados da lista de amigos. Então crie uma tabela denominada usuário, que contenha os campos:

- id (chave primária e auto incremento).
- Nome.
- senha.

Conforme mostra o diagrama a seguir.

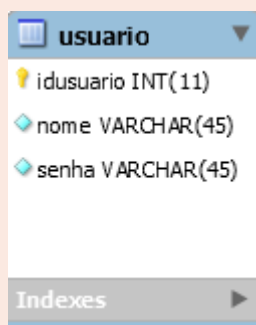


Imagem 3. Diagrama Banco de dados.

Nesta tabela, vamos adicionar um registro de login através do comando SQL:

```
INSERT INTO `pwii`.`usuario` (`nome`, `senha`) VALUES ('gabi', 'gabi123');
```

Obs: Utilizamos para o campo nome o valor “gabi” e para o campo senha “gabi123”, mas você pode utilizar qualquer valor, lembre-se apenas que estes valores serão utilizados para login no site.

Você pode copiar o projeto da pasta utilizada na agenda 6, para uma pasta chamada Agenda8 (ou outra pasta de sua preferência) ou continuar utilizando a mesma pasta do projeto anterior, isso tudo fica a seu critério.

Para iniciar, o atual arquivo “index” deverá ser renomeado para “principal”, assim, podemos no visual studio Code, dentro da pasta Agenda8 (ou outra pasta de sua preferência), criar o primeiro arquivo da atualização com o nome “index.php” - Neste arquivo vamos criar a página de login, que vai consistir em um formulário, com dois campos (nome e senha) e um botão, que terá o propósito de fornecer acesso a página recentemente nomeada como principal. Então codifique.

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
  <title></title>
</head>
<body>
  <div class="w3-container w3-round-xxlarge w3-display-middle w3-card-4 w3-
third " style="">

    <div class="w3-center">
      <br>
      
    </div>

    <form class="w3-container " action="loginAction.php" method="post">
      <div class="w3-section">
        <label style="font-weight: bold;">Usuário</label>
        <input class="w3-input w3-border w3-margin-
bottom" type="text" placeholder="Digite o nome" name="txtNome" required>
        <label style="font-weight: bold;">Senha</label>
        <input class="w3-input w3-
border" type="text" placeholder="Digite a Senha" name="txtSenha" required>
        <button class="w3-button w3-block w3-teal w3-section w3-
padding" type="submit">Entrar</button>
      </div>
    </form>
    <br>
  </div>
</body>
</html>

```

O resultado no Navegador será como o representado na imagem a seguir.

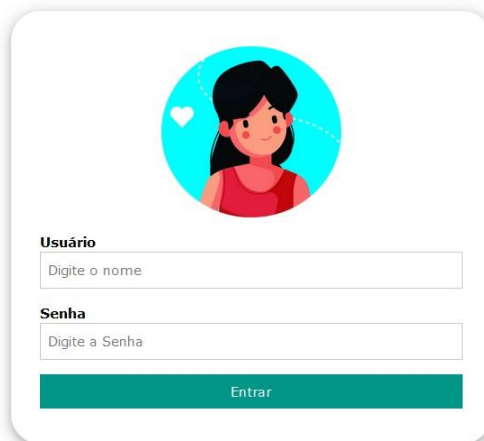
A login form mockup with a white background and rounded corners. At the top center is a circular profile picture of a woman with black hair and a red top. Below the picture are two input fields: the first is labeled 'Usuário' with a placeholder 'Digite o nome'; the second is labeled 'Senha' with a placeholder 'Digite a Senha'. At the bottom is a teal button labeled 'Entrar'.

Imagem 4. Resultado no Navegador.

Obs: Ao digitar a senha perceba que ela está sendo exibida, para não exibir basta alterar o type do input pra “password”.

Senha

••••••••••

Imagem 5. Input configurado como password.

Como é possível perceber pelo código anterior, a action do formulário indica para o arquivo “loginAction.php”, que será o próximo arquivo que vamos desenvolver.

Então, no visual studio Code, dentro da pasta Agenda8 (ou outra pasta de sua preferência), crie o arquivo com o nome “loginAction.php” - Neste arquivo, será realizada a conexão com o banco de dados através do Mysql, a criação e execução da sentença sql de select, verificando se existe ou não o usuário, e por fim verificando se a senha digitada no formulário é a mesma gravada no banco de dados. Para isso Codifique.

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
    <title></title>
</head>
<body>
    <div class="w3-padding w3-content w3-text-grey w3-third w3-display-
        middle" >

        <?php
            $nome = $_POST['txtNome'];
            $senha = $_POST['txtSenha'];
            $servername = "localhost";
            $username = "root";
            $password = "usbw";
            $dbname = "pwii";
            $conexao = new mysqli($servername, $username, $password, $dbname);
            if ($conexao->connect_error) {
                die("Connection failed: " . $conexao->connect_error);
            }

            $sql = "SELECT * FROM usuario WHERE nome = '$nome.'";
            $resultado = $conexao->query($sql);
            //echo $sql;
            $linha = mysqli_fetch_array($resultado);
            if($linha != null)
            {
                if($linha['senha'] == $senha)
                {
                    echo '
                        <a href="principal.php">
                            <h1 class="w3-button w3-teal">'. $nome.', Seja Bem-
Vinda! </h1>

                        </a>
                    ';
                }
                else
                {
                    echo '
                        <a href="index.php">
                            <h1 class="w3-button w3-teal">Login Inválido! </h1>
                        </a> ';
                }
            }
        }
        else
        {
            echo '
                <a href="index.php">

```



```

        <h1 class="w3-button w3-teal">Login Inválido! </h1>
    </a>
    ';
}

$conexao->close();
?>
</div>

</body>
</html>

```

Perceba que existe um desvio condicional, que em caso a senha digitada no formulário e a senha do banco sejam as mesmas, será exibido um link que redirecionará o usuário para a página principal (imagem 6).



Imagem 6. Login Realizado com sucesso.

Em caso negativo (senhas diferentes), será exibido um link que retornará para página principal.



Imagem 7. Login inválido.

Pronto, nosso login está criado! Porém perceba o seguinte, toda vez que vamos realizar o acesso ao banco de dados, seja, para inserir, deletar, atualizar ou buscar uma informação precisamos passar os dados de conexão. Então, estamos utilizando acesso ao banco em pelo menos 4 arquivos. Vamos imaginar que por algum motivo a senha para login no banco, mudou, neste projeto precisaríamos ir aos quatro arquivos e fazer a alteração um por vez, mas se utilizarmos o conceito de importação de arquivos (include ou require), podemos criar um arquivo chamado “conexaoBD” e dentro dele programar:


```
<?php

$servername = "localhost";
$username = "root";
$password = "usbw";
$dbname = "pwii";
$conexao = new mysqli($servername, $username, $password, $dbname);
if ($conexao->connect_error) {
    die("Connection failed: " . $conexao->connect_error);
}

?>
```

E substituir em todos os quatro arquivos esse mesmo código por:

```
require_once 'conexaoBD.php';
```

ou

```
require_once (conexaoBD.php);
```

Assim, em uma possível troca de senha, precisaríamos apenas alterar o arquivo “conexaoBD”, e todo nosso projeto seria “atualizado” e utilizaria a senha correta. Além disso, diminui a quantidade de códigos digitados, evita esquecimentos e erros. Deve resultar em um código semelhante com:

```
<?php
$nome = $_POST['txtNome'];
$senha = $_POST['txtSenha'];
require_once 'conexaoBD.php';
$sql = "SELECT * FROM usuario WHERE nome = '". $nome . "'";
$resultado = $conexao->query($sql);
//echo $sql;
$linha = mysqli_fetch_array($resultado);
if($linha != null)
{
    if($linha['senha'] == $senha)
    {
        echo '
        <a href="principal.php">
            <h1 class="w3-button w3-teal">'. $nome .', Seja Bem-
Vinda! </h1>
        </a>
        ';
    }
}
```

```

        else
        {
            echo '
            <a href="index.php">
                <h1 class="w3-button w3-teal">Login Inválido! </h1>
            </a>
            ';
        }
    }
else
    {
        echo '
        <a href="index.php">
            <h1 class="w3-button w3-teal">Login Inválido! </h1>
        </a>
        ';
    }

$conexao->close();
?>

```

Outra boa utilização, é criar arquivos de cabeçalho e rodapé, e utilizar a importação de arquivos, assim quando alterar nester arquivos, todo o site será atualizado.

Então no visual studio code, crie dois arquivos salvando-os na mesma pasta do projeto. O primeiro denominado “cabecalho.php” com o seguinte código.

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title></title>
</head>
<body>

```

O Segundo arquivo denominado de rodapé, com o seguinte código:

```

</body>
</html>

```

Obs: Este exemplo é bem simples e a quantidade de código no rodapé é mínima, porém, é possível em sites mais complexos ter um rodapé bem mais complexo e recheado de código e informações relevantes.

Para finalizar atualize todos os arquivos do projeto, substituindo todos os códigos iguais por:

```
<?php require_once ('cabecalho.php'); ?>
```

No Cabeçalho e para o rodapé:

```
<?php require_once ('rodape.php'); ?>
```

Para melhor exemplificar, a seguir o código atualizado do arquivo “index.php” com os requires.

```
<?php require_once ('cabecalho.php'); ?>
    <div class="w3-container w3-round-xxlarge w3-display-middle w3-card-4 w3-
third " style="">

        <div class="w3-center">
            <br>
            
        </div>

        <form class="w3-container " action="loginAction.php" method="post">
            <div class="w3-section">
                <label style="font-weight: bold;">Usuário</label>
                <input class="w3-input w3-border w3-margin-
bottom" type="text" placeholder="Digite o nome" name="txtNome" required>
                <label style="font-weight: bold;">Senha</label>
                <input class="w3-input w3-
border" type="password" placeholder="Digite a Senha" name="txtSenha" required>
                <button class="w3-button w3-block w3-teal w3-section w3-
padding" type="submit">Entrar</button>
            </div>
        </form>
        <br>

    </div>
<?php require_once ('rodape.php'); ?>
```

Com esta etapa concluída, realize um teste! Digitando no navegador “localhost/nomeDaPastaDoSeuProjeto/principal.php”, você vai perceber que foi possível burlar o login, para isso precisamos estudar o próximo tópico SESSION!

SESSION e suas variáveis

Na internet há um problema: um servidor web não sabe quem é você ou o que você faz quando você acessa um site, porque o endereço HTTP não mantém o estado ou condição que você está no site. Para resolver este problema, entra em ação as variáveis de sessões, elas conseguem realizar o armazenamento de informações do usuário para serem utilizadas em várias páginas (por exemplo, nome de usuário, cor favorita etc.). Desta forma as variáveis de sessão contêm informações sobre um único usuário e estão disponíveis para todas as páginas durante seu acesso.

Para iniciar uma sessão em php você precisa da função `session_start()`, e as variáveis de sessão serão configuradas com a variável global PHP: `$_SESSION`.

```
<?php
session_start();
$_SESSION["Nome"] = "Gabi";
$_SESSION["Idade"] = 17;
?>
```

No Código acima foram criadas duas variáveis de sessão a primeira denominada Nome e atribuindo o valor dentro da mesma, a segunda denominada idade e atribuição do valor 17 na mesma.

Para obter os valores contido nas variáveis em outras páginas, basta iniciar novamente a sessão, e utilizá-las como o exemplo a seguir.

```
<?php
session_start();
echo "Nome: " . $_SESSION["Nome"] . " e idade: " . $_SESSION["Idade"];
?>
```

Com esse conceito explanado, vamos atualizar nosso projeto e proteger o acesso via url de usuário não identificado, atualizando o arquivo “loginAction”. A atualização está em iniciar a sessão (“session_start();”) e dentro da estrutura condicional em caso verdadeiro para login e senha corretos, o valor da variável \$nome, dentro de uma variável de sessão denominada “logado”, resultando código a seguir.

```
<?php require_once ('cabecalho.php'); ?>
<div class="w3-padding w3-content w3-text-grey w3-third w3-display-middle" >
    <?php
        session_start();
        $nome = $_POST['txtNome'];
        $senha = $_POST['txtSenha'];
        require_once 'conexaoBD.php';
        $sql = "SELECT * FROM usuario WHERE nome = '". $nome ."'";
        $resultado = $conexao->query($sql);
        //echo $sql;
        $linha = mysqli_fetch_array($resultado);
        if($linha != null)
        {
            if($linha['senha'] == $senha)
            {
                echo '
                <a href="principal.php">
                    <h1 class="w3-button w3-teal">'. $nome .', Seja Bem-
Vinda! </h1>
                </a>
                ';
                $_SESSION['logado'] = $nome;
            }
            else
            {
                echo '
                <a href="index.php">
                    <h1 class="w3-button w3-teal">Login Inválido! </h1>
                </a>
                ';
            }
        }
    else
    {
        echo '
        <a href="index.php">
            <h1 class="w3-button w3-teal">Login Inválido! </h1>
        </a>
        ';
    }
}

$conexao->close();
```

```

    ?>
</div>
<?php require_once ('rodape.php'); ?>

```

Agora, crie um arquivo chamado “verificarAcesso”, este arquivo sempre vai verificar se há uma sessão aberta e se a variável logado está iniciada. Caso ela não esteja, utilizando a função nativa header, redirecionaremos o acesso para outro arquivo denominado acessoNegado. O código para esse arquivo de ser.

```

<?php
if(!isset($_SESSION))
{
    session_start();
}
if((!isset ($_SESSION['logado']) == true))
{
    header('location:acessoNegado.php');
    die();
}
?>

```

Por fim, é necessário criar o arquivo “acessoNegado.php”, que terá apenas uma mensagem de Acesso Negado e um link para a página index, para a realização do login. Segue o código.

```

<?php require_once ('cabecalho.php'); ?>
<div class="w3-padding w3-content w3-text-grey w3-third w3-display-middle">

    <?php
        echo '
            <a href="index.php">
                <h1 class="w3-button w3-teal">Acesso NEGADO! </h1>
                <br>
                <h2 class="w3-button w3-teal">É necessária a Realização do Login Inválido! </h2>
            </a>
        ';
    ?>
</div>
<?php require_once ('rodape.php'); ?>

```

Por fim, precisamos atualizar todos os arquivos inserindo na primeira linha o seguinte código:

```

<?php require_once ('verificarAcesso.php');?>

```

Esta simples linha de código fará, com que sempre que um usuário tente acessar via url as páginas do nosso projeto, seja verificado o acesso do mesmo. E caso ele não tenha realizado o login, sempre aparecerá a mensagem “Acesso Negado”.

Para exemplificar segue o código do arquivo principal.php atualizado.

```
<?php require_once ('verificarAcesso.php');?>
<?php require_once ('cabecalho.php'); ?>
<div class="w3-padding w3-text-grey w3-half w3-display-middle w3-center">

    <h1 class="w3-center w3-teal w3-round-large w3-
margin">Projeto Lista de Amigos</h1>
    <div class="w3-row">
        <div class="w3-col w3-button w3-teal w3-cell w3-round-
large" style="width:45%;">
            <a href="cadastro.php" style="text-decoration: none;">
                <i class=" fa fa-user-plus" style="font-size: 10.5em"></i>
                <p style="font-size: 2em">Adicionar </p>
            </a>
        </div>
        <div class="w3-col w3-button w3-teal w3-cell w3-round-large w3-
right" style="width:45%;">
            <a href="listar.php" style="text-decoration: none;">
                <i class="fa fa-vcard-o" style="font-size: 10.5em"></i>
                <p style="font-size: 2em">Listar</p>
            </a>
        <div>
        </div>
    </div>
</div>
<?php require_once ('rodape.php'); ?>
```

Agora podemos observar o resultado na imagem a seguir, ao tentar acessar essa página através da url.



Acesso NEGADO!

É necessária a Realização do Login Inválido!

Obs: Para saber mais sobre função header para redirecionamento, vale a leitura do link a seguir.

<https://www.hostinger.com.br/tutoriais/redirecionamento-php/>

Por fim vamos criar o logout para quando o usuário decidir deixar nossa página, crie um arquivo chamado “logoutAction.php”, este arquivo vai basicamente remover a variável de sessão “logado” e redirecionar (função header) para a página inicial “index”, o que tornará o login novamente necessário.

```
<?php require_once ('verificarAcesso.php'); ?>
<?php
    unset( $_SESSION['logado'] );
    header("location:index.php");
?>
```

Para o usuário realizar o logout, a criação de um botão no arquivo “principal.php” se faz necessária, o que poder ser realizado inserindo o código a seguir logo depois do: <?php require_once ('cabecalho.php');?>.

```
<div class="w3-padding w3-content w3-text-grey w3-third w3-margin w3-display-topright">
    <form action="logoutAction.php" class="w3-container" method='post'>
        <button name="btnLogout" class="w3-button w3-red w3-cell w3-round-large w3-right w3-margin-right">
            <i class="w3-xxlarge fa fa-times-rectangle"> </i> Logout
        </button>
    </form>
</div>
```

Obs: O formato dessa agenda abordou o conceito utilizando apenas o conteúdo desenvolvido até o momento nas disciplinas, há outros métodos para fazer todas as operações contidas nessa agenda.

ATENÇÃO - [Baixe aqui o arquivo usbwebserver – Agenda8](#), nele você encontra todos os arquivos php e tabelas no banco de dados do mergulhando no tema e você no comando.

Com esta etapa concluída, realize diversos testes, implemente diversas atualizações isso vai te trazer o desenvolvimento de mais competências!

Cookie

Cookie são dados ou grupo de dados que browser (navegador) e o servidor Apache trocam dados e informações entre si, enquanto o usuário utiliza as páginas do seu site. Estes dados são salvos em arquivo de texto direto no computador do usuário.

Para criar e manipular um cookie você precisa basicamente aprender basicamente apenas uma função (setcookie()), para o primeiro parâmetro deve-se definir um nome para o cookie, o segundo por sua vez refere-se ao valor do cookie.

```
<?php
setcookie('usuario', 'gabi');

?>
```

Há também um terceiro parâmetro utilizado para determinar um tempo de “vida” para esse cookie.

```
<?php
// Criando um cookie para durar um dia
setcookie('usuario', 'gabi', (time() + (24 * 3600)));

?>
```

Obs.: Neste exemplo, o cookie está obtendo a hora do servidor, e adicionando a quantidade de segundos de um dia, fazendo com que o cookie tenha a duração por 24 horas (1 dia) em segundos.

Assista o vídeo, e entenda mais sobre cookies e suas utilizações:

Cookie em PHP



Node Studio Terinamentos - Curso de PHP 7 - Aula 52 - \$_COOKIE- Disponível em:
<https://www.youtube.com/watch?v=ilHJx8hB8yA> . Acessado em 29/07/2020.

Hospedagem Registro de Domínios e mais

Para ser possível, acessar sua página, site ou porta de qualquer lugar com internet é necessário alguns recursos: como um domínio e hospedagem. Existem diversas formas para obter esse recursos a seguir uma sequência de 3 vídeos que que explica de forma simples e objetiva o mínimo necessário. E também mais alguns vídeos com opções para hospedagens e criação de domínios.

Registro de Domínio | O que é um domínio?



Fonte: Hostnet Internet - Registro de Domínio | O que é um domínio?.

Disponível <https://www.youtube.com/watch?v=bFpSJrzzqPU> . Acessado em 15/07/2020.

Registro de Domínio | Por que eu preciso de um domínio?



Fonte: Hostnet Internet - Registro de Domínio | Por que eu preciso de um domínio?.

Disponível <https://www.youtube.com/watch?v=OcJk8P5imS8>. Acessado em 15/07/2020.

Registro de Domínio | 10 Dicas Para Escolher o Nome do Seu Domínio



Fonte: Hostnet Internet - Registro de Domínio | O que é um domínio?.

Disponível <https://www.youtube.com/watch?v=y5oVU34R0Zk>. Acessado em 15/07/2020.

Hospedagem de Site com Domínio Grátis e SSL (como registrar um domínio na HostGator)



Fonte: Hospedagem de Site com Domínio Grátis e SSL (como registrar um domínio na HostGator).

Disponível em: <https://www.youtube.com/watch?v=DeosapvOQrE>. Acessado em 29/07/2020.

Curso de HTML5 - 35 - Como Hospedar um Site - by Gustavo Guanabara

Fonte: Filipe Deschamps - Curso em Vídeo Curso de HTML5 - 35 - Como Hospedar um Site - by Gustavo Guanabara.

Disponível em <https://www.youtube.com/watch?v=O5mKRNvoWbE> . Acessado em 20/07/2020.

Criando e hospedando sites com a godaddy!

Fonte: Loop Infinito. CRIANDO E HOSPEDANDO SITES COM A GODADDY!.

Disponível em <https://www.youtube.com/watch?v=X8jNCKvPRjQ> . Acessado em 29/06/2020.

Obs: O formato dessa agenda abordou o conceito utilizando apenas o conteúdo desenvolvido até o momento nas disciplinas, há outros métodos para fazer todas as operações contidas nessa agenda.

ATENÇÃO - Será disponibilizado o arquivo **usbwebserver – Agenda8**, nele terá todos os arquivos php e tabelas no banco de dados do mergulhando no tema e você no comando.



Utilizando o que foi visto até agora, foi disponibilizado uma página com o nome, disciplina e dados de vários professores. Nossa missão é restringir o acesso a a essa página, através de identificação. Esta identificação deve ser validada através da tabela (professor) criada no **banco de dados** com os atributos: idprofessor, nome, disciplina e senha, conforme diagrama a seguir.

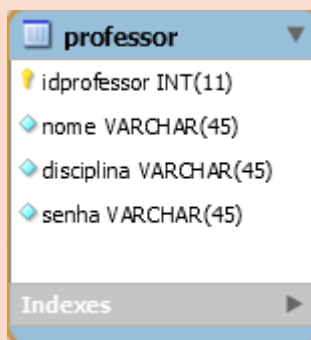


Imagem 16. Diagrama Banco de dados.

1. Obtenha os dados da tabela, utilizando o driver PDO ou Mysqli
2. Crie uma Página para um formulário para login e senha
3. Utilize Session para impedir o acesso via URL da página principal
4. Crie um botão para realizar logout na página em que os professores estão listados
5. Crie arquivos para auxiliar no desenvolvimento

Dicas:

- Utilize os conceitos e exemplos de estrutura de repetição da Agenda 5.
- De preferência ao uso da estrutura de repetição Foreach.
- Agenda 6 ou 7 para acesso ao banco de dados.
- Os conceitos de importação podem te ajudar a economizar tempo

Caso esteja com dificuldade a seguir script sql para criação da tabela estado.

```
CREATE TABLE `pwii`.`professor` (
  `idprofessor` INT NOT NULL AUTO_INCREMENT,
  `nome` VARCHAR(45) NOT NULL,
  `disciplina` VARCHAR(45) NOT NULL,
  `senha` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idprofessor`));
```

Caso esteja com dificuldade, a seguir script sql inserção de todos registros de estados.

```
INSERT INTO `pwii`.`professor` (`nome`, `disciplina`, `senha`) VALUES ('Paulo', 'Programação WEB II', 'paulo123');
INSERT INTO `pwii`.`professor` (`nome`, `disciplina`, `senha`) VALUES ('Guilherme', 'Programação Mobile', 'guilherme123');
INSERT INTO `pwii`.`professor` (`nome`, `disciplina`, `senha`) VALUES ('Eliana', 'Lógica de Programação', 'eliana123');
```

Caso esteja com dificuldade, Código da página dos professores.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
  <title>Listagem Professores</title>
</head>
<body class="w3-black">
```

```

<div class="w3-paddingw3-content w3-half w3-display-topmiddle w3-
margin">
  <h1 class="w3-center w3-green w3-margin">Professores</h1>
  <table class="w3-table-all w3-centered w3-text-black">
    <thead>
      <tr class="w3-center w3-green ">
        <th>Código</th>
        <th>Nome</th>
        <th>Disciplina</th>
      </tr>
    </thead>
  </table>
  <?php
    $servername = "localhost";
    $username = "root";
    $password = "usbw";
    $dbname = "pwii";
    $conexao = new mysqli($servername, $username, $password, $dbname);
    if ($conexao->connect_error) {
      die("Connection failed: " . $conexao->connect_error);
    }

    $conexao->set_charset("utf8");
    if ($conexao->connect_error) {
      die("Connection failed: " . $conexao->connect_error);
    }
    $sql = "SELECT * FROM professor" ;
    $resultado = $conexao->query($sql);
    if($resultado != null)
      foreach($resultado as $linha) {
        echo '<tr>';
        echo '<td>'.$linha['idprofessor'].'</td>';
        echo '<td>'.$linha['nome'].'</td>';
        echo '<td>'.$linha['disciplina'].'</td>';
        echo '</tr>';
      }
    $conexao->close();
  <?>
</table>

```

A seguir, confira se você conseguiu resolver os desafios propostos!



Para a Resposta foram criados os seguintes arquivos:

- Index - contém o formulário para login e action direcionando para o arquivo "LoginAction".

- LoginAction - contém a consulta no banco para verificação se há o professor ou não, em caso verdadeiro exibirá mensagem de login realizado, com link para página de professores, em caso negativo uma mensagem de falhar com link para a página index.
- LogoutAction - contém a codificação para o logout do usuário conectado e o redirecionando para a página index.
- conexaoBD - criado para concentrar os dados e conexão em um único arquivo.
- acessoNegado - utilizado para quando o usuário tentar realizar o acesso a página sem login.

A seguir, código e imagens de tudo o que foi desenvolvido.



The image shows a login interface on a black background. It features two white input fields. The first field is labeled 'Usuário' and contains the placeholder text 'Digite o nome'. The second field is labeled 'Senha' and contains the placeholder text 'Digite a Senha'. Below these fields is a green button with the text 'Entrar' in white.

Imagem 17. Index.

Código:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
  <title></title>
</head>
<body class="w3-black">
  <div class="w3-container w3-round-xxlarge w3-display-middle w3-
third " style="">
```

```

        <form class="w3-container " action="loginAction.php" method="post">
            <div class="w3-section">
                <label style="font-weight: bold;">Usuário</label>
                <input class="w3-input w3-border w3-margin-
bottom" type="text" placeholder="Digite o nome" name="txtNome" required>
                <label style="font-weight: bold;">Senha</label>
                <input class="w3-input w3-
border" type="text" placeholder="Digite a Senha" name="txtSenha" required>
                <button class="w3-button w3-block w3-green w3-section w3-
padding" type="submit">Entrar</button>
            </div>
        </form>
        <br>

    </div>
</div>
</body>
</html>

```



Login Realizado com Sucesso!



Login Inválido!

Imagem 18. LoginAction.

Código:

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
    <title></title>
</head>
<body class="w3-black">
<div class="w3-padding w3-content w3-text-grey w3-third w3-display-middle" >

```

```

<?php
session_start();
$nome = $_POST['txtNome'];
$senha = $_POST['txtSenha'];
require_once 'conexaoBD.php';
$sql = "SELECT * FROM professor WHERE nome = '". $nome . "'";
$resultado = $conexao->query($sql);
//echo $sql;
$linha = mysqli_fetch_array($resultado);
if($linha != null)
{
    if($linha['senha'] == $senha )
    {
        echo '
        <a href="professor.php">
            <h1 class="w3-button w3-
green">Login Realizado com Sucesso! </h1>
        </a>
        ';
        $_SESSION['logado'] = $nome;
    }
    else
    {
        echo '
        <a href="index.php">
            <h1 class="w3-button w3-green">Login Inválido! </h1>
        </a>
        ';
    }
}
else
{
    echo '
    <a href="index.php">
        <h1 class="w3-button w3-green">Login Inválido! </h1>
    </a>
    ';
}

$conexao->close();

?>
</div>
</body>
</html>

```



Imagem 19. Professor com atualização.

Código:

```
<?php
if(!isset($_SESSION))
{
    session_start();
}
if((!isset ($_SESSION['logado']) == true))
{
    header('location:acessoNegado.php');
    die();
}
?>

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title>Listagem Professores</title>
</head>
<body class="w3-black">
    <div class="w3-paddingw3-content w3-half w3-display-topmiddle w3-margin">
```

```

<h1 class="w3-center w3-green w3-margin">Professores</h1>
<table class="w3-table-all w3-centered w3-text-black">
  <thead>
    <tr class="w3-center w3-green ">
      <th>Código</th>
      <th>Nome</th>
      <th>Disciplina</th>
    </tr>
  </thead>
  <?php
    require_once 'conexaoBD.php';
    $conexao->set_charset("utf8");
    if ($conexao->connect_error) {
      die("Connection failed: " . $conexao->connect_error);
    }
    $sql = "SELECT * FROM professor" ;
    $resultado = $conexao->query($sql);
    if($resultado != null)
      foreach($resultado as $linha) {
        echo '<tr>';
        echo '<td>'.$linha['idprofessor'].'</td>';
        echo '<td>'.$linha['nome'].'</td>';
        echo '<td>'.$linha['disciplina'].'</td>';
        echo '</tr>';
      }
    $conexao->close();
  ?>
</table>

</div>
<div class="w3-padding w3-content w3-text-grey w3-third w3-margin w3-display-
bottomright">
  <form action="logoutAction.php" class="w3-container" method='post'>
    <button name="btnLogout" class="w3-button w3-red w3-cell w3-round-
large w3-right w3-margin-right">
      <i class="w3-xxlarge fa fa-times-rectangle"> </i> Logout
    </button>
  </form>
</div>
</body>
</html>

```

Código Conexão BD:

```

<?php

$servername = "localhost";
$username = "root";
$password = "usbw";
$dbname = "pwii";

```



```
$conexao = new mysqli($servername, $username, $password, $dbname);
if ($conexao->connect_error) {
    die("Connection failed: " . $conexao->connect_error);
}

?>
```



Acesso Negado

Imagem 20. Professor com atualização.

Código Acesso Negado:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
    <title></title>
</head>
<body class="w3-black">
<div class="w3-padding w3-content w3-third w3-display-middle">

    <?php
        echo '
        <a href="index.php">
            <h1 class="w3-button w3-green">Acesso Negado </h1>
        </a>
        ';
    ?>
</div>
</body>
</html>
```

Obs: O formato dessa da reposta abordou apenas o conteúdo desenvolvido até o momento nas disciplinas, há outros métodos para fazer todas as operações contidas nesse exercício.

ATENÇÃO - Será disponibilizado o arquivo **usbwebserver – Agenda8.rar**, nele terá todos os arquivos php e tabelas no banco de dados do mergulhando no tema, você no comando disponíveis para uso e consulta.