Codigo:

conexao.php

```php
<?php
$hostname = "localhost";
$bancodedados = "cliente";
$usuario = "root";
$senha = "";
$mysqli = new mysqli( $hostname,$usuario,$senha,$bancodedados);

if ($mysqli->connect_error){
    echo "Falha ao conectar ao banco de dados:
(".$mysqli->connect_errno.")".$mysqli->connect_error;
}else{
    echo "Banco de dados conectado com sucesso.<br>";
}
?>
```

grafico.php

```php
<?php
include("conexao.php");
// Verifique se a conexão foi bem-sucedida
if (!$mysqli) {
    die("Falha na conexão: " . $mysqli->connect_error);
}
// Contagem de carros
$sql = "SELECT COUNT(*) as total FROM usuarios";
$result = $mysqli->query($sql);
if ($result) {
    $totalUsers = ($result->num_rows > 0) ?
$result->fetch_assoc()['total'] : 0;
} else {
    echo "Erro ao consultar o banco de dados: " . $mysqli->error;
    $totalUsers = 0;  // Defina um valor padrão caso haja erro
}
$mysqli->close();
?>
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Relatório de Carros</title>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
```

```
</head>
<body>
    <h2>Gráfico de Carros e Usuarios Cadastrados</h2>
    <canvas id="allChart" width="400" height="200"></canvas>
    <script>
        var totalUsers = <?php echo $totalUsers; ?>;
        var ctx = document.getElementById('allChart').getContext('2d');
        var allChart = new Chart(ctx, {
            type: 'bar',
            data: {
                labels: ['quantidade total'],
                datasets: [{
                    label: 'usuarios',
                    data: [totalUsers],
                    backgroundColor: 'rgba(205,90,90,0.2)',
                    borderColor: 'rgba(192,75,75,1)',
                    borderWidth: 2
                }]
            },
            options: {
                scales: {
                    y: {
                        beginAtZero: true
                    }
                }
            }
        });
    </script>
    <br>
    <a href="user.php">Voltar ao Cadastro de Usuarios</a>
    <br>
<br>
<a href="index.php">
    Voltar a tela de inicial
</a>
</body>
</html>
```

index.php

```
<form action = "user.php" method ="get">
    <input type="submit" value="CADASTRAR CLIENTE">
</form>
<form action = "listaruser.php" method ="get">
```

```
    <input type="submit" value="LISTA">
</form>
<form action = "grafico.php" method ="get">
    <input type="submit" value="GRAFICO">
</form>
```

listaruser.php

```
<?php
include("conexao.php");
$sql= "SELECT Nome, Email FROM usuarios";
$result = $mysqli->query($sql);
if ($result->num_rows >0){
    echo"<h2> LISTA DE CLIENTES CADASTRADOS </h2>";
    echo"<table border ='1'>
        <tr>
            <th> NOME </th>
            <th> E-MAIL </th>
        </tr>";
    while ($row = $result->fetch_assoc()){
        echo"<tr>
                <td> " . $row['Nome'] . " </td>
                <td> " . $row['Email'] . " </td>
            </tr>";
    }
}else {
    echo "Nehum usuario no sistema.";
}
$mysqli->close();
?>
<br>
<a href="user.php">
    Voltar a tela de cadastro
</a>
<br>
<br>
<a href="index.php">
    Voltar a tela de inicial
</a>
```

user.php

```
<?php
include("conexao.php");
// cadastro usuario
if ($_SERVER["REQUEST_METHOD"]== "POST"){
```

```php
    $Nome = $_POST['Nome'];
    $Email = $_POST['Email'];
    if (empty($Nome) || empty($Email)){
        echo "campos são obrigatorios";
    }else{
        $stmt = $mysqli->prepare("INSERT INTO usuarios (nome, email)
VALUES (?, ?)");
        $stmt->bind_param("ss",$Nome, $Email);
        if($stmt->execute()){
            echo "usuario cadastrado";
        }else{
            echo "deu ruim! ". $stmt->error;
        }
        $stmt->close();
    }
}
?>
<h2>Cadastro de usuario</h2>
<form method="POST" action="">
    Nome: <input type="text" name="Nome"><br>
    Email: <input type="text" name="Email"><br>
    <input type="submit" value="Cadastrar">
</form>
<form action = "listaruser.php" method ="get">
    <input type="submit" value="LISTA">
</form>
<form action = "grafico.php" method ="get">
    <input type="submit" value="GRAFICO">
</form>
<br>
<br>
<a href="index.php">
    Voltar a tela de inicial
</a>
```

Tutorial: Criando um Sistema de Cadastro, Listagem e Visualização de Dados em PHP

Objetivo

Desenvolver um sistema PHP que permita:

1. Conectar a um banco de dados MySQL.
2. Cadastrar usuários.
3. Listar usuários cadastrados.
4. Visualizar dados em formato de gráfico.

Passo 1: Preparando o Ambiente

Requisitos

- Servidor local (XAMPP, WAMP ou similar).
- Banco de dados MySQL.
- Editor de código (VS Code, Sublime, etc.).
- Biblioteca Chart.js (para gráficos).

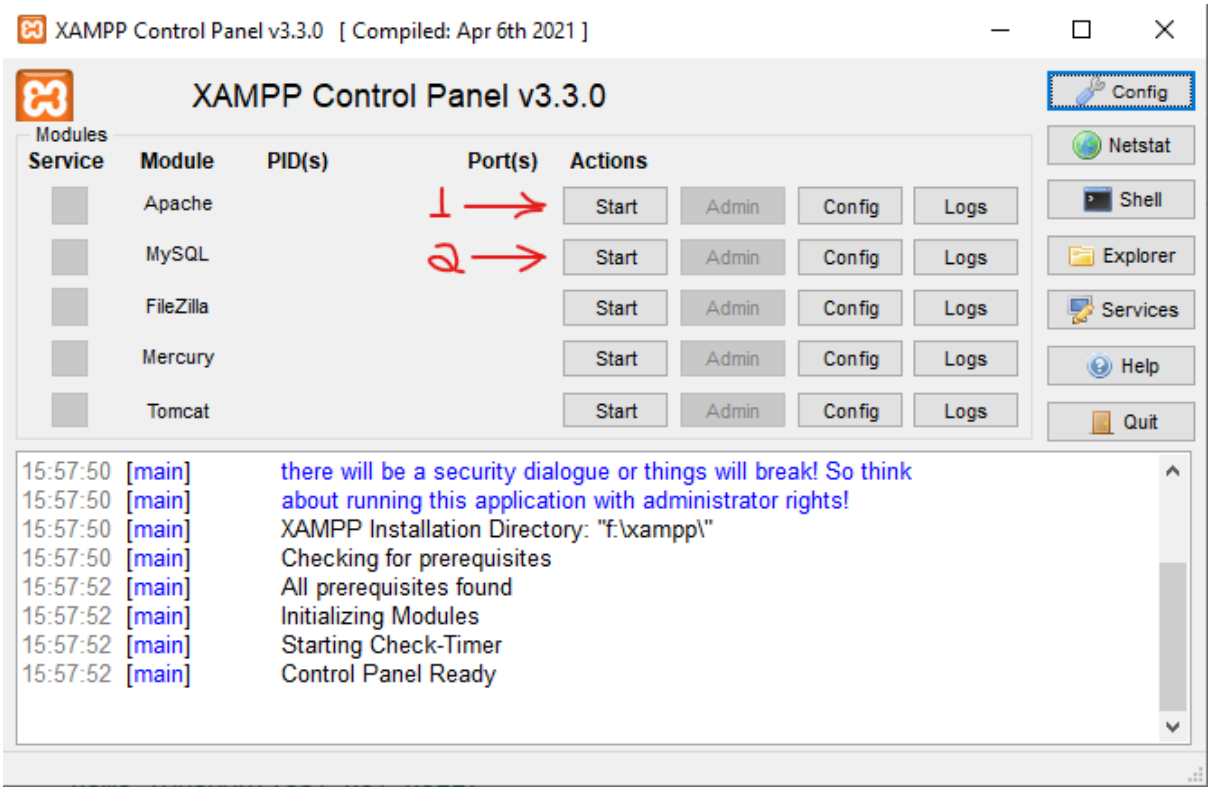
Configuração Inicial

Crie um banco de dados no MySQL com o nome `cliente`.

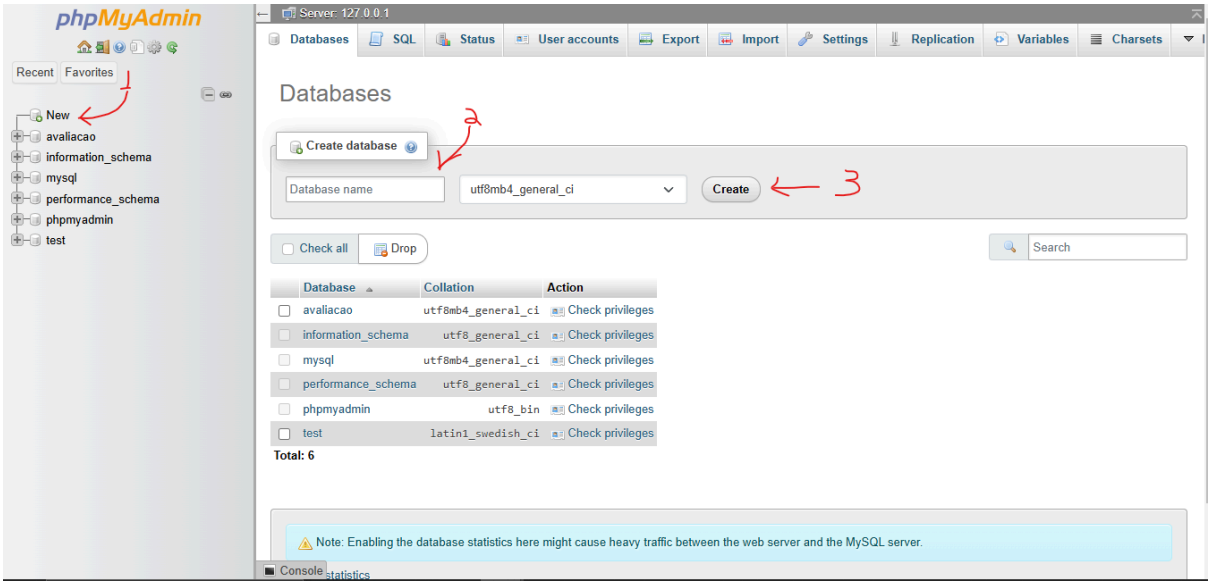
por codigo:

```
CREATE DATABASE cliente;
```

com XAMPP:



acesse http://localhost/phpmyadmin/ e crie um novo banco de dados

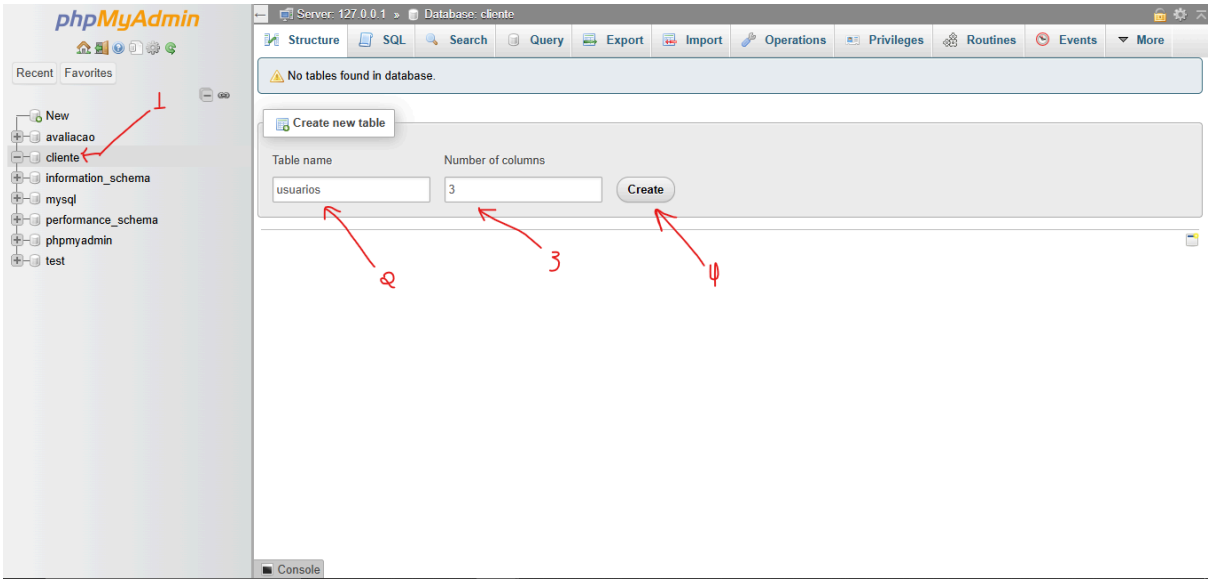


Dentro do banco, crie a tabela `usuarios`:
por codigo:

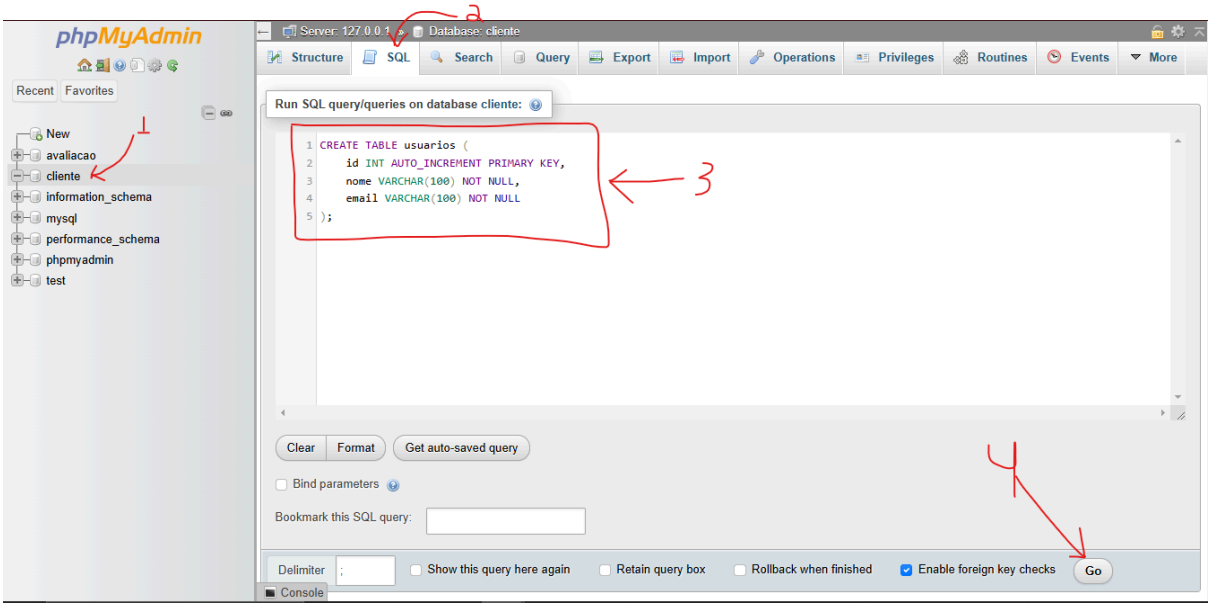
```
CREATE TABLE usuarios (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(100) NOT NULL,
```

```
    email VARCHAR(100) NOT NULL
);
```

pelo XAMPP:



e crie os campos manualmente OU



Passo 2: Criando o Arquivo de Conexão

Crie o arquivo `conexao.php` para gerenciar a conexão com o banco de dados.

```
<?php
```

```
$hostname = "localhost";
$bancodedados = "cliente";
$usuario = "root";
$senha = "";

// Cria a conexão
$mysqli = new mysqli($hostname, $usuario, $senha, $bancodedados);

// Verifica a conexão
if ($mysqli->connect_error) {
    die("Falha ao conectar ao banco de dados: " .
$mysqli->connect_error);
}
?>
```

O que esse arquivo faz?

- Configura os dados de conexão.
- Usa `mysqli` para conectar ao banco.
- Termina o script com `die()` caso a conexão falhe.

Passo 3: Criando o Cadastro de Usuários

Crie o arquivo `user.php`. Ele terá um formulário para entrada de dados e lógica para inseri-los no banco.

```
<?php
include("conexao.php");

// Processa o formulário
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $Nome = $_POST['Nome'];
    $Email = $_POST['Email'];

    // Valida os dados
    if (empty($Nome) || empty($Email)) {
        echo "Todos os campos são obrigatórios!";
    } else {
        // Insere no banco
        $stmt = $mysqli->prepare("INSERT INTO usuarios (nome, email)
VALUES (?, ?)");
        $stmt->bind_param("ss", $Nome, $Email);
```

```
        if ($stmt->execute()) {
            echo "Usuário cadastrado com sucesso!";
        } else {
            echo "Erro: " . $stmt->error;
        }
        $stmt->close();
    }
}
?>
<h2>Cadastro de Usuários</h2>
<form method="POST" action="">
    Nome: <input type="text" name="Nome"><br>
    Email: <input type="text" name="Email"><br>
    <input type="submit" value="Cadastrar">
</form>
<a href="index.php">Voltar</a>
```

Explicação:

Uso do Método `POST`

- `POST` é usado para enviar dados ao servidor, como os valores inseridos no formulário.
- Ele é mais seguro do que `GET` para envio de informações sensíveis, pois os dados não aparecem na URL.

No PHP:

- `$_SERVER["REQUEST_METHOD"]` verifica se o método usado foi `POST`.
- `$_POST` acessa os valores enviados pelo formulário.

No HTML:

- `method="POST"` define que o formulário usará o método `POST`.
- `action=""` envia os dados para o próprio arquivo.

Função `prepare()`

- Protege contra injeção de SQL, permitindo a execução de consultas parametrizadas.

Passo 4: Criando a Listagem de Usuários

Crie o arquivo `listaruser.php` para exibir uma tabela com os usuários cadastrados.

```php
<?php
include("conexao.php");

// Consulta os dados
$sql = "SELECT nome, email FROM usuarios";
$result = $mysqli->query($sql);

echo "<h2>Lista de Usuários</h2>";

if ($result->num_rows > 0) {
    echo "<table border='1'>
            <tr>
                <th>Nome</th>
                <th>Email</th>
            </tr>";
    while ($row = $result->fetch_assoc()) {
        echo "<tr>
                <td>" . $row['nome'] . "</td>
                <td>" . $row['email'] . "</td>
              </tr>";
    }
    echo "</table>";
} else {
    echo "Nenhum usuário cadastrado.";
}

$mysqli->close();
?>
<a href="index.php">Voltar</a>
```

Explicação:

- Usa `query()` para executar uma consulta SQL.
- `fetch_assoc()` retorna cada linha da tabela como um array associativo.
- Gera uma tabela HTML dinamicamente para exibir os resultados.

Passo 5: Criando a Visualização Gráfica

Crie o arquivo `grafico.php` para exibir um gráfico com o total de usuários.

```php
<?php
include("conexao.php");

// Conta os usuários
$sql = "SELECT COUNT(*) as total FROM usuarios";
$result = $mysqli->query($sql);
$totalUsers = $result->fetch_assoc()['total'];
$mysqli->close();
?>
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <title>Gráfico de Usuários</title>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
    <h2>Gráfico de Usuários</h2>
    <canvas id="userChart" width="400" height="200"></canvas>
    <script>
        var totalUsers = <?php echo $totalUsers; ?>;
        var ctx =
document.getElementById('userChart').getContext('2d');
        var chart = new Chart(ctx, {
            type: 'bar',
            data: {
                labels: ['Total de Usuários'],
                datasets: [{
                    label: 'Usuários',
                    data: [totalUsers],
                    backgroundColor: 'rgba(75, 192, 192, 0.2)',
                    borderColor: 'rgba(75, 192, 192, 1)',
                    borderWidth: 1
                }]
            },
            options: {
                scales: {
                    y: {
                        beginAtZero: true
                    }
                }
            }
        });
```

```
    </script>
    <a href="index.php">Voltar</a>
</body>
</html>
```

Explicação:

Chart.js

- É uma biblioteca JavaScript de código aberto para criar gráficos interativos.
- Neste exemplo, o tipo de gráfico usado é `bar` (barra).
- A API permite configurar cores, labels, escalas e outros elementos visuais.

Integração com PHP

- Os dados do PHP (número total de usuários) são passados para o JavaScript usando `echo`.

Passo 6: Criando a Página Inicial

Crie o arquivo `index.php` para servir como menu principal.

```
<h1>Sistema de Gerenciamento</h1>
<form action="user.php" method="get">
    <input type="submit" value="Cadastrar Usuário">
</form>
<form action="listaruser.php" method="get">
    <input type="submit" value="Listar Usuários">
</form>
<form action="grafico.php" method="get">
    <input type="submit" value="Visualizar Gráfico">
</form>
```

Explicação:

simplesmente uma página central que permite a navegação

Passo 7: Testando o Sistema

1. Suba o projeto no servidor local (ex.: XAMPP).
2. Acesse `http://localhost/seu_projeto/index.php`.

3. Teste as funcionalidades:
    - Cadastre usuários.
    - Confira a lista.
    - Visualize o gráfico.

codigo do trabalho (acho muito complexo para ser pedido em prova)

pasta raiz:
index.php

```php
<?php
/*
- Trabalho Parcial = Desenvolva um sistema WEB em PHP que tenha as seguintes opções. FrontEnd
    - Cadastro de imagens. Inserir, deleta, atualizar.
    - Cadastro de textos. Inserir, deletar atualizar.
    - Opções de pontuação.
    - Opções de comentários.
    - Opções de compartilhamento (Pode ser link).
    - Estrutura: Deve estar implementado as estruturas de classes e funções em PHP.
    - Deve conter os nomes dos integrantes do grupo e a matricula.
*/
include("php/conexao.php");
require_once 'php/Post.php';
require_once 'php/usuario.php';
require_once 'php/PostManager.php';


session_start(); // Mover esta linha para depois dos require_once

// Inicializa o PostManager na sessão, caso ainda não esteja inicializado
if (!isset($_SESSION['postManager'])) {
    $_SESSION['postManager'] = new PostManager();
}

// Atribui o PostManager à variável local $postManager
$postManager = $_SESSION['postManager'];

// Verifica se o usuário está logado
if (isset($_SESSION['usuario']) && $_SESSION['usuario'] instanceof Usuario) {
    $usuario = $_SESSION['usuario'];
}else{
    $usuario = null;
}


// Lógica para curtir ou descurtir um post
```

```php
if (isset($_GET['curtir']) && $usuario) {
    $postManager->curtirPost($_GET['curtir']);
    header("Location: index.php");
    exit;
}

if (isset($_GET['descurtir']) && $usuario) {
    $postManager->descurtirPost($_GET['descurtir']);
    header("Location: index.php");
    exit;
}
?>



<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <title>Sistema de Posts</title>
    <link rel="stylesheet" type="text/css" href="css/style.css"> <!--
CSS -->
    <script>
    // Salva a posição do scroll no localStorage
    window.onbeforeunload = function () {
        localStorage.setItem('scrollPosition', window.scrollY);
    };

    // Restaura a posição do scroll após o carregamento
    window.onload = function () {
        const scrollPosition = localStorage.getItem('scrollPosition');
        if (scrollPosition) {
            window.scrollTo(0, parseInt(scrollPosition));
        }
    };
</script>
</head>
<body>
    <header>
        <h1>Bem-vindo ao Sistema de Posts</h1>
        <nav>
            <?php if ($usuario): ?>
                <span>Olá, <?= htmlspecialchars($usuario->getNome());
?>!</span>
```

```
                <a href="logout.php">Logout</a>
            <?php else: ?>
                <a href="login.php">Login</a>
                <a href="registrar.php">Registrar</a>
            <?php endif; ?>
        </nav>
    </header>

    <main>
        <div class="posts">
            <?php
            $posts = $postManager->carregarPosts();
            if (!empty($posts)) {
                foreach ($posts as $post) {

                    echo "<div class='post'>";
                    echo "<a
href='post_detalhes.php?id={$post->getId()}'><h2>{$post->getTitulo()}</
h2></a>";
                    echo $post->exibirPost();
                    // Verifica se o usuário curtiu/descurtiu o post
                    if ($usuario){
                        $voto =
$postManager->verificarVoto($post->getId(), $usuario->getId());
                        // Botões Like e Dislike com cores dinâmicas
                        $likeClass = ($voto === 'like') ?
'btn-like-active' : 'btn-like';
                        $dislikeClass = ($voto === 'dislike') ?
'btn-dislike-active' : 'btn-dislike';

                        echo "<a
href='index.php?curtir={$post->getId()}' class='$likeClass'>Like</a> ";
                        echo "<a
href='index.php?descurtir={$post->getId()}'
class='$dislikeClass'>Dislike</a>";
                    }

                    echo "</div>";
                }
            } else {
                echo "<p>Nenhum post disponível.</p>";
            }
            ?>
```

```
        </div>
        <?php
            if ($usuario) {
                echo "<br><a
href='cadastro_post.php'class='btn-new'>Cadastrar Novo Post</a>";
            }
        ?>
    </main>
</body>
</html>
```

cadastro_post.php

```
<?php
include("php/conexao.php");
require_once 'php/Post.php';
require_once 'php/usuario.php';
require_once 'php/PostManager.php';


session_start(); // Inicia a sessão

// Inicializa o PostManager na sessão, caso ainda não esteja
inicializado
if (!isset($_SESSION['postManager'])) {
    $_SESSION['postManager'] = new PostManager();
}

$usuario = $_SESSION['usuario'];

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $titulo = $_POST['titulo'];
    $tipo = $_POST['tipo'];
    $conteudo = '';

    // Verifica se o post é de texto ou imagem
    if ($tipo === 'texto') {
        $conteudo = $_POST['conteudo'];
    } elseif ($tipo === 'imagem' && isset($_FILES['imagem'])) {
        // Upload da imagem e salvando o caminho
        $targetDir = "img/";
        $targetFile = $targetDir . basename($_FILES["imagem"]["name"]);
        if (move_uploaded_file($_FILES["imagem"]["tmp_name"],
$targetFile)) {
```

```
            $conteudo = $targetFile;
        } else {
            echo "Erro ao carregar a imagem.";
            exit;
        }
    }

    // Adiciona o novo post ao PostManager na sessão
    //$novoPost = new Post(rand(), $titulo, $conteudo, $tipo,
$usuario->getId()); // ID aleatório para o post
    $_SESSION['postManager']->adicionarPost($titulo, $conteudo, $tipo,
$usuario->getId());

    // Redireciona de volta para a página principal
    header("Location: index.php");
    exit;
}
?>

<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <title>Cadastro de Post</title>
    <script>
        // Função para exibir/esconder campos com base no tipo
selecionado
        function atualizarCampos() {
            var tipo = document.getElementById('tipo').value;
            var conteudoCampo =
document.getElementById('conteudoCampo');
            var imagemCampo = document.getElementById('imagemCampo');

            if (tipo === 'texto') {
                conteudoCampo.style.display = 'block';
                imagemCampo.style.display = 'none';
            } else if (tipo === 'imagem') {
                conteudoCampo.style.display = 'none';
                imagemCampo.style.display = 'block';
            }
        }
```

```
        // Chama a função ao carregar a página para garantir que os
campos corretos estejam visíveis
        window.onload = function() {
            atualizarCampos();
        };
    </script>
</head>
<body>
    <h1>Cadastrar Novo Post</h1>

    <form action="cadastro_post.php" method="post"
enctype="multipart/form-data">
        <label for="titulo">Título:</label>
        <input type="text" name="titulo" id="titulo" required>

        <label for="tipo">Tipo de Post:</label>
        <select name="tipo" id="tipo" required
onchange="atualizarCampos()">
            <option value="texto">Texto</option>
            <option value="imagem">Imagem</option>
        </select>

        <!-- Campo de Conteúdo (Texto) -->
        <div id="conteudoCampo">
            <label for="conteudo">Conteúdo (Texto):</label>
            <textarea name="conteudo" id="conteudo"></textarea>
        </div>

        <!-- Campo de Imagem -->
        <div id="imagemCampo">
            <label for="imagem">Imagem (se aplicável):</label>
            <input type="file" name="imagem" id="imagem">
        </div>

        <button type="submit">Cadastrar Post</button>
    </form>

    <script>
        // Garantir que o campo correto esteja visível ao carregar a
página
        document.addEventListener("DOMContentLoaded", function() {
            atualizarCampos();
        });
```

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

editar_post.php

```
<?php
require_once 'php/Post.php';
require_once 'php/PostManager.php';

session_start();

if (!isset($_SESSION['postManager']) || !isset($_GET['id'])) {
    echo "Post não encontrado.";
    exit;
}

$postId = $_GET['id'];
$postManager = $_SESSION['postManager'];
$post = null;

foreach ($postManager->exibirPosts() as $p) {
    if ($p->getId() == $postId) {
        $post = $p;
        break;
    }
}

if (!$post) {
    echo "Post não encontrado.";
    exit;
}

// Lógica de edição
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $novoTitulo = $_POST['titulo'];
    if ($post->getTipo() === 'texto') {
        $novoConteudo = $_POST['conteudo'];
        $postManager->atualizarPost($postId, $novoTitulo,
$novoConteudo);
    } elseif ($post->getTipo() === 'imagem' &&
isset($_FILES['nova_imagem'])) {
        // Upload da nova imagem e atualiza o caminho
```

```
        $targetDir = "img/";
        $targetFile = $targetDir .
basename($_FILES["nova_imagem"]["name"]);

        if (move_uploaded_file($_FILES["nova_imagem"]["tmp_name"],
$targetFile)) {
            // Remove a imagem antiga, se necessário
            if (file_exists($post->getConteudo())) {
                unlink($post->getConteudo()); // Remove o arquivo da
imagem antiga
            }
            $novoConteudo = $targetFile;
            $postManager->atualizarPost($postId, $novoTitulo,
$novoConteudo);
        } else {
            echo "Erro ao carregar a nova imagem.";
            exit;
        }
    }
    header("Location: post_detalhes.php?id=$postId");
    exit;
}
?>

<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <title>Editar Post</title>
</head>
<body>
    <h1>Editando: <?php echo $post->getTitulo(); ?></h1>

    <form action="editar_post.php?id=<?php echo $postId; ?>"
method="post" enctype="multipart/form-data">
        <label for="titulo">Título:</label>
        <input type="text" name="titulo" value="<?php echo
$post->getTitulo(); ?>" required>

        <?php if ($post->getTipo() === 'texto'): ?>
            <label for="conteudo">Conteúdo:</label>
            <textarea name="conteudo" required><?php echo
$post->getConteudo(); ?></textarea>
```

```
        <?php elseif ($post->getTipo() === 'imagem'): ?>
            <p>Imagem Atual:</p>
            <img src="<?php echo $post->getConteudo(); ?>" alt="Imagem
do Post" style="max-width: 200px;">
            <label for="nova_imagem">Substituir Imagem:</label>
            <input type="file" name="nova_imagem" id="nova_imagem">
        <?php endif; ?>

        <button type="submit">Salvar Alterações</button>
    </form>
</body>
</html>
```

login.php

```
<?php
include("php/conexao.php");
require_once 'php/usuario.php';
session_start();


if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $email = $_POST['email'];
    $senha = $_POST['senha'];

    if (empty($email) || empty($senha)) {
        echo "Preencha todos os campos.";
    } else {
        // Consulta o banco de dados para buscar o usuário
        $stmt = $mysqli->prepare("SELECT * FROM usuarios WHERE email =
?");
        $stmt->bind_param("s", $email);
        $stmt->execute();
        $result = $stmt->get_result();

        if ($result->num_rows > 0) {
            $row = $result->fetch_assoc();
            // Verifica se a senha fornecida corresponde à armazenada
            if (password_verify($senha, $row['senha'])) {
                // Salva o email do usuário na sessão
                $usuario = new Usuario($row['id'], $row['nome'],
$row['email'], $row['senha']);
                $_SESSION['usuario'] = $usuario;
```

```
                header("Location: index.php"); // Redireciona para o
index
                exit;
            } else {
                echo "Senha inválida.";
            }
        } else {
            echo "Usuário não encontrado.";
        }
        $stmt->close();
    }
}
?>

<h2>Login</h2>
<form method="POST" action="">
    Email: <input type="text" name="email"><br>
    Senha: <input type="password" name="senha"><br>
    <input type="submit" value="Entrar">
</form>
```

logout.php

```
<?php
session_start();
session_unset(); // Remove todas as variáveis de sessão
session_destroy(); // Destrói a sessão
header("Location: index.php"); // Redireciona para a página inicial
exit;
?>
```

post_detalhes.php

```
<?php
error_reporting(E_ALL);
ini_set('display_errors', 1);

include("php/conexao.php");
require_once 'php/Post.php';
require_once 'php/usuario.php';
require_once 'php/PostManager.php';
require_once 'php/Comment.php';

session_start();
```

```php
if (!isset($_SESSION['postManager']) || !isset($_GET['id'])) {
    echo "Post não encontrado1.";
    exit;
}

$postId = $_GET['id'];
$postManager = $_SESSION['postManager'];
$post = null;
$posts = $postManager->exibirPosts();
if (!empty($posts)) {
    foreach ($posts as $p) {
        if ($p->getId() == $postId){
            $post = $p;
            break;
        }
    }
}else{
    echo "não quis acessar banco de dados ";
}
// Verifica se o usuário está logado
if (isset($_SESSION['usuario']) && $_SESSION['usuario'] instanceof
Usuario) {
    $usuario = $_SESSION['usuario'];
}else{
    $usuario = null;
}

if (!$post) {
    echo "Post não encontrado2.";
    exit;
}

// Lógica para adicionar um comentário
if ($_SERVER['REQUEST_METHOD'] === 'POST' &&
isset($_POST['comentario'])) {
    $textoComentario = trim($_POST['comentario']);
    if (!empty($textoComentario)) {
        $novoComentario = new Comment(rand(), $textoComentario,
"Usuário Exemplo");
        $post->adicionarComentario($novoComentario);
    }
    header("Location: post_detalhes.php?id=$postId");
    exit;
```

```php
}

// Lógica para excluir um comentário
if (isset($_POST['deleteComment'])) {
    $comentarioId = $_POST['commentId'];
    $post->removerComentario($comentarioId);
    header("Location: post_detalhes.php?id=$postId");
    exit;
}

// Lógica para curtir ou descurtir um post
if (isset($_GET['curtir']) && $usuario) {
    $postManager->curtirPost($_GET['curtir']);
    header("Location: post_detalhes.php?id=$postId");
    exit;
}

if (isset($_GET['descurtir']) && $usuario) {
    $postManager->descurtirPost($_GET['descurtir']);
    header("Location: post_detalhes.php?id=$postId");
    exit;
}

// Lógica para exclusão
if (isset($_POST['delete'])) {
    if ($usuario && $usuario->getId() == $post->getAutor()) {
        $postManager->removerPost($postId);
        header("Location: index.php");
        exit;
    } else {
        echo "Você não tem permissão para excluir este post.";
        exit;
    }
}
?>

<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <title>Detalhes do Post</title>
    <link rel="stylesheet" type="text/css" href="css/style.css">
    <script>
```

```
        function copiarLink() {
            // Cria um elemento temporário para armazenar o link
            var link = window.location.href;
            var tempInput = document.createElement("input");
            tempInput.value = link;
            document.body.appendChild(tempInput);
            tempInput.select();
            document.execCommand("copy");
            document.body.removeChild(tempInput);
            alert("Link copiado para a área de transferência!");
        }

        // Salva a posição do scroll no localStorage
        window.onbeforeunload = function () {
            localStorage.setItem('scrollPosition', window.scrollY);
        };

        // Restaura a posição do scroll após o carregamento
        window.onload = function () {
            const scrollPosition =
localStorage.getItem('scrollPosition');
            if (scrollPosition) {
                window.scrollTo(0, parseInt(scrollPosition));
            }
        };
    </script>
</head>
<body>
    <header>
        <a href="index.php" class="btn-back">Voltar para a página
inicial</a>
    </header>
    <main>
        <h1><?php echo $post->getTitulo(); ?></h1>
        <?php
        echo $post->exibirPostDetalhe();
        echo "<p>Autor: {$post->getNomeAutor()}</p>";
        ?>

        <?php if ($usuario && $usuario->getId() == $post->getAutor()):
?>
            <form action="post_detalhes.php?id=<?php echo $postId; ?>"
method="post">
```

```
                <button type="submit" name="delete"
class='btn-delete'>Excluir Post</button>
                <a href="editar_post.php?id=<?php echo $postId;
?>">Editar Post</a>
            </form>
        <?php endif; ?>
        <?php
        if ($usuario){
            $voto = $postManager->verificarVoto($post->getId(),
$usuario->getId());
            // Botões Like e Dislike com cores dinâmicas
            $likeClass = ($voto === 'like') ? 'btn-like-active' :
'btn-like';
            $dislikeClass = ($voto === 'dislike') ?
'btn-dislike-active' : 'btn-dislike';

            echo "<p>";
            echo "<a
href='post_detalhes.php?id={$postId}&curtir={$post->getId()}'
class='$likeClass'>Like</a> ";
            echo "<a
href='post_detalhes.php?id={$postId}&descurtir={$post->getId()}'
class='$dislikeClass'>Dislike</a>";
            echo "</p>";
        }
        ?>

        <h3>Compartilhar</h3>
        <button onclick="copiarLink()">Copiar Link</button>

        <h3>Comentários</h3>
        <?php if ($usuario): ?>
            <form action="post_detalhes.php?id=<?php echo $postId; ?>"
method="post">
            <textarea name="comentario" placeholder="Escreva seu
comentário"></textarea>
            <button type="submit">Enviar Comentário</button>
        </form>
        <?php endif; ?>
        <div class="comentarios">
            <?php foreach ($post->getComentarios() as $comentario): ?>
                <div class="comentario">
```

```
                    <p><?php echo $comentario->getTexto(); ?> -
<strong><?php echo $comentario->getAutorNome(); ?></strong></p>
                    <form action="post_detalhes.php?id=<?php echo
$postId; ?>" method="post" style="display:inline;">
                        <input type="hidden" name="commentId"
value="<?php echo $comentario->getId(); ?>">
                        <?php if ($usuario && ($usuario->getId() ==
$post->getAutor() || $usuario->getId() == $comentario->getAutor())): ?>
                            <button type="submit" name="deleteComment"
class='btn-delete'>Excluir Comentário</button>
                        <?php endif; ?>
                    </form>
                </div>
            <?php endforeach; ?>
        </div>
    </main>
</body>
</html>
```

registrar.php

```
<?php
include("php/conexao.php");

/*
// cadastro usuario
*/

if ($_SERVER["REQUEST_METHOD"]== "POST"){

    $Nome = $_POST['Nome'];
    $Email = $_POST['Email'];
    $Senha = $_POST['Senha'];
    $ConfirmarSenha = $_POST['ConfirmarSenha'];

    if (empty($Nome) || empty($Email) || empty($Senha)||
empty($ConfirmarSenha)){
        echo "campos são obrigatorios";//nunca usado ja que os camps
tem required
    }else if($Senha != $ConfirmarSenha){
        echo "Senhas diferentes! Corfimre e entre as senhas
novamente.";
    }else{
```

```
        // Criptografar a senha
        $senhaHash = password_hash($Senha, PASSWORD_DEFAULT);
        $stmt = $mysqli->prepare("INSERT INTO usuarios (nome, email,
senha) VALUES (?, ?, ?)");
        $stmt->bind_param("sss",$Nome, $Email, $senhaHash);
        if($stmt->execute()){
            echo "usuario cadastrado";
            header("Location: index.php");
            exit;
        }else{
            echo "Erro: ". $stmt->error;
        }
        $stmt->close();
    }
}
?>

<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Cadastro</title>
</head>
<body>
    <h2>Cadastro</h2>
    <?php if (isset($mensagem)) { echo "<p>$mensagem</p>"; } ?>
    <form method="POST" action="">
        Nome: <input type="text" name="Nome" required><br>
        Email: <input type="email" name="Email" required><br>
        Senha: <input type="password" name="Senha" required><br>
        Confirmar senha: <input type="password" name="ConfirmarSenha"
required><br>
        <button type="submit">Cadastrar</button>
    </form>
    <br>
    <br>
    <a href="index.php">
    Voltar a tela de inicial
    </a>
</body>
</html>
```

dentro da pasta php:
Comment.php

```php
<?php
class Comment {
    private $id;
    private $texto;
    private $autor;

    public function __construct($id, $texto, $autor) {
        $this->id = $id;
        $this->texto = $texto;
        $this->autor = $autor;
    }

    public function getId() {
        return $this->id;
    }

    public function getTexto() {
        return $this->texto;
    }

    public function getAutor() {
        return $this->autor;
    }

    public function getAutorNome() {
        global $mysqli;

        // Prepara a consulta para buscar o nome do autor pelo ID
        $stmt = $mysqli->prepare("SELECT nome FROM usuarios WHERE id =
?");
        $stmt->bind_param("i", $this->autor);
        $stmt->execute();
        $result = $stmt->get_result();

        // Verifica se encontrou o autor
        if ($row = $result->fetch_assoc()) {
            return $row['nome'];
        } else {
            return "Autor desconhecido";
        }
    }
```

```
}

?>
```

conexao.php

```
<?php
$hostname = "localhost";
$bancodedados = "avaliacao";
$usuario = "root";
$senha = "";

$mysqli = new mysqli( $hostname,$usuario,$senha,$bancodedados);

if ($mysqli->connect_error){
    echo "Falha ao conectar ao banco de dados:
(".$mysqli->connect_errno.")".$mysqli->connect_error;
}
?>
```

Post.php

```
<?php

class Post {
    private $id;
    private $titulo;
    private $conteudo;
    private $tipo;
    private $autor; // ID do autor
    private $pontuacao;

    public function __construct($id, $titulo, $conteudo, $tipo, $autor,
$pontuacao = 0) {
        $this->id = $id;
        $this->titulo = $titulo;
        $this->conteudo = $conteudo;
        $this->tipo = $tipo;
        $this->autor = $autor;
        $this->pontuacao = $pontuacao;
    }

     // Métodos Getters
     public function getId() { return $this->id; }
     public function getTitulo() { return $this->titulo; }
```

```php
    public function getConteudo() { return $this->conteudo; }
    public function getTipo() { return $this->tipo; }
    public function getAutor() { return $this->autor; }
    public function getNomeAutor() {
        global $mysqli;

        // Prepara a consulta para buscar o nome do autor pelo ID
        $stmt = $mysqli->prepare("SELECT nome FROM usuarios WHERE id =
?");
        $stmt->bind_param("i", $this->autor);
        $stmt->execute();
        $result = $stmt->get_result();

        // Verifica se encontrou o autor
        if ($row = $result->fetch_assoc()) {
            return $row['nome'];
        } else {
            return "Autor desconhecido";
        }
    }
    public function getPontuacao() { return $this->pontuacao; }

    // Atualiza título ou conteúdo
    public function atualizar($titulo, $conteudo) {
        global $mysqli;
        $stmt = $mysqli->prepare("UPDATE posts SET titulo = ?, conteudo
= ? WHERE id = ?");
        $stmt->bind_param("ssi", $titulo, $conteudo, $this->id);
        $stmt->execute();
    }

    // Remover post (somente pelo autor)
    public function remover() {
        global $mysqli;
        $stmt = $mysqli->prepare("DELETE FROM posts WHERE id = ?");
        $stmt->bind_param("i", $this->id);
        $stmt->execute();
    }

    function atualizarPontuacoes($mysqli) {
        $query = "
            UPDATE posts p
            SET p.pontuacao = (
```

```
                SELECT
                    COALESCE(SUM(CASE
                        WHEN v.tipo = 'like' THEN 1
                        WHEN v.tipo = 'dislike' THEN -1
                        ELSE 0
                    END), 0)
                FROM votos v
                WHERE v.post_id = p.id
            );
        ";

        if ($mysqli->query($query) === TRUE) {
            return true;
        } else {
            echo "Erro ao atualizar pontuações: " . $mysqli->error;
            return false;
        }
    }

    public function exibirPost() {
        $conteudoExibido = $this->tipo === 'texto' ?
"<p>{$this->conteudo}</p>" : "<img src='{$this->conteudo}'
alt='{$this->titulo}' class='thumbnail' />";
        return "{$conteudoExibido}<p>Pontuação:
{$this->pontuacao}</p>";
    }

    public function exibirPostDetalhe() {
        $conteudoExibido = $this->tipo === 'texto' ?
"<p>{$this->conteudo}</p>" : "<img src='{$this->conteudo}'
alt='{$this->titulo}'/>";
        return "{$conteudoExibido}<p>Pontuação:
{$this->pontuacao}</p>";
    }

    // Curtir post
    public function curtir() {
        global $mysqli;

        $usuarioId = $_SESSION['usuario']->getId();
        $id = $this->id;

        // Verifica se já existe um voto
```

```
        $stmt = $mysqli->prepare("SELECT tipo FROM votos WHERE post_id
= ? AND usuario_id = ?");
        $stmt->bind_param("ii", $id, $usuarioId);
        $stmt->execute();
        $result = $stmt->get_result();

        if ($voto = $result->fetch_assoc()) {
            if ($voto['tipo'] !== 'like') {
                // Atualiza para "like"
                $stmt = $mysqli->prepare("UPDATE votos SET tipo =
'like' WHERE post_id = ? AND usuario_id = ?");
                $stmt->bind_param("ii", $id, $usuarioId);
                $stmt->execute();
            } else {
                // Remove voto
                $stmt = $mysqli->prepare("DELETE FROM votos WHERE
post_id = ? AND usuario_id = ?");
                $stmt->bind_param("ii", $id, $usuarioId);
                $stmt->execute();
            }
        } else {
            // Novo voto
            //echo "LIKE";
            $stmt = $mysqli->prepare("INSERT INTO votos (post_id,
usuario_id, tipo) VALUES (?, ?, 'like')");
            $stmt->bind_param("ii", $id, $usuarioId);
            $stmt->execute();
        }
        $this->atualizarPontuacoes($mysqli); // Atualiza pontuações do
post
    }

    // Descurtir post
    public function descurtir() {
        global $mysqli;

        $usuarioId = $_SESSION['usuario']->getId();
        $id = $this->id;

        // Verifica se já existe um voto
        $stmt = $mysqli->prepare("SELECT tipo FROM votos WHERE post_id
= ? AND usuario_id = ?");
        $stmt->bind_param("ii", $id, $usuarioId);
```

```
        $stmt->execute();
        $result = $stmt->get_result();

        if ($voto = $result->fetch_assoc()) {
            if ($voto['tipo'] !== 'dislike') {
                // Atualiza para "dislike"
                $stmt = $mysqli->prepare("UPDATE votos SET tipo =
'dislike' WHERE post_id = ? AND usuario_id = ?");
                $stmt->bind_param("ii", $id, $usuarioId);
                $stmt->execute();
            } else {
                // Remove voto
                $stmt = $mysqli->prepare("DELETE FROM votos WHERE
post_id = ? AND usuario_id = ?");
                $stmt->bind_param("ii", $id, $usuarioId);
                $stmt->execute();
            }
        } else {
            // Novo voto
            $stmt = $mysqli->prepare("INSERT INTO votos (post_id,
usuario_id, tipo) VALUES (?, ?, 'dislike')");
            $stmt->bind_param("ii", $id, $usuarioId);
            $stmt->execute();
        }
        $this->atualizarPontuacoes($mysqli); // Atualiza pontuações do
post
    }

    public function adicionarComentario ($comentario){
        global $mysqli;

        $postId = $this->id;
        $usuarioId = $_SESSION['usuario']->getId();
        $texto = $comentario->getTexto();
        $stmt = $mysqli->prepare("INSERT INTO comentarios (post_id,
usuario_id, texto) VALUES (?, ?, ?)");
        $stmt->bind_param("iis", $postId, $usuarioId, $texto);
        $stmt->execute();
    }

    public function removerComentario($comentarioId){
        global $mysqli;
```

```
        $stmt = $mysqli->prepare("DELETE FROM comentarios WHERE id =
?");
        $stmt->bind_param("i", $comentarioId);
        $stmt->execute();
    }

    public function getComentarios(){
        global $mysqli;
        $stmt = $mysqli->query("SELECT * FROM comentarios WHERE post_id
= $this->id ORDER BY id DESC");
        $comentarios = [];

        while ($row = $stmt->fetch_assoc()) {
            $comentarios[] = new Comment($row['id'], $row['texto'],
$row['usuario_id']);
        }
        return $comentarios;
    }
}
?>
```

PostManager.php

```
<?php
include("conexao.php");
class PostManager {
    private $posts = [];

    public function adicionarPost($titulo, $conteudo, $tipo, $autor) {
        global $mysqli;
        $stmt = $mysqli->prepare("INSERT INTO posts (titulo, conteudo,
tipo, autor, pontuacao) VALUES (?, ?, ?, ?, 0)");
        $stmt->bind_param("sssi", $titulo, $conteudo, $tipo, $autor);
        $stmt->execute();
    }

    public function removerPost($id) {
        $this->posts = $this->carregarPosts();
        foreach ($this->posts as $post) {
            if ($post->getId() == $id) {
                global $mysqli;

                // Prepara a query para deletar o post
```

```
                $stmt = $mysqli->prepare("DELETE FROM posts WHERE id =
?");
                if (!$stmt) {
                    echo "Erro na preparação da query: " .
$mysqli->error;
                    return false;
                }

                // Vincula o parâmetro
                $stmt->bind_param("i", $id);

                // Executa a query
                if ($stmt->execute()) {
                    // Verifica se é um post com imagem e remove o
arquivo
                    if ($post->getTipo() === 'imagem' &&
file_exists($post->getConteudo())) {
                        $filePath = realpath($post->getConteudo());
                        unlink($filePath);
                    }
                    echo "Post deletado com sucesso.";
                    return true;
                } else {
                    echo "Erro ao deletar post: " . $stmt->error;
                    return false;
                }

            }
        }

    }

    public function atualizarPost($id, $novoTitulo, $novoConteudo) {
        foreach ($this->posts as $post) {
            if ($post->getId() == $id) {
                $post->atualizar($novoTitulo,$novoConteudo);
            }
        }
    }

    public function carregarPosts() {
        global $mysqli;
        $stmt = $mysqli->query("SELECT * FROM posts ORDER BY id DESC");
```

```
        $posts = [];

        while ($row = $stmt->fetch_assoc()) {
            $posts[] = new Post($row['id'], $row['titulo'],
$row['conteudo'], $row['tipo'], $row['autor'], $row['pontuacao']);
        }
        return $posts;
    }

    public function verificarVoto($postId, $usuarioId){
        global $mysqli;

    $stmt = $mysqli->prepare("SELECT tipo FROM votos WHERE post_id = ?
AND usuario_id = ?");
    $stmt->bind_param("ii", $postId, $usuarioId);
    $stmt->execute();
    $result = $stmt->get_result();

    if ($voto = $result->fetch_assoc()) {
        return $voto['tipo']; // Retorna 'like' ou 'dislike'
    }
    return null; // Não há voto
    }


    public function exibirPosts() {
        $this->posts = $this->carregarPosts();
        return $this->posts;
    }

    public function curtirPost($id) {
        var_dump($id);
        $this->posts= $this->carregarPosts();
        foreach ($this->posts as $post) {
            if ($post->getId() == $id) {
                $post->curtir();
            }
        }
    }

    public function descurtirPost($id) {
        $this->posts= $this->carregarPosts();
        foreach ($this->posts as $post) {
```

```
            if ($post->getId() == $id) {
                $post->descurtir();
            }
        }
    }
}
?>
```

usuario.php

```
<?php

class Usuario
{
    private $id; // ID único do usuário
    private $nome; // Nome do usuário
    private $email; // Email do usuário
    private $senha; // Senha (já deve ser armazenada criptografada no
banco)

    // Construtor para inicializar o objeto
    public function __construct($id, $nome, $email, $senha)
    {
        $this->id = $id;
        $this->nome = $nome;
        $this->email = $email;
        $this->senha = $senha;
    }

    // Métodos Getters e Setters (mantidos como antes)
    public function getId() { return $this->id; }
    public function getNome() { return $this->nome; }
    public function setNome($nome) { $this->nome = $nome; }
    public function getEmail() { return $this->email; }
    public function setEmail($email) { $this->email = $email; }
    public function getSenha() { return $this->senha; }
    public function setSenha($senha) { $this->senha = $senha; }

    // Método para atualizar no banco de dados
    public function atualizarNoBanco($mysqli)
    {
        $stmt = $mysqli->prepare("UPDATE usuarios SET nome = ?, email =
?, senha = ? WHERE id = ?");
```

```
        $stmt->bind_param("sssi", $this->nome, $this->email,
$this->senha, $this->id);
        $stmt->execute();
        $stmt->close();
    }


    // Métodos específicos


    // Recuperar posts do usuário
    public function getPosts($mysqli)
    {
        $stmt = $mysqli->prepare("SELECT * FROM posts WHERE usuario_id
= ?");
        $stmt->bind_param("i", $this->id);
        $stmt->execute();
        $result = $stmt->get_result();

        $posts = [];
        while ($row = $result->fetch_assoc()) {
            $posts[] = $row; // Aqui você pode retornar objetos Post,
se tiver uma classe Post
        }

        $stmt->close();
        return $posts;
    }

    // Verificar se o usuário já deu like em um post
    public function jaDeuLike($mysqli, $postId)
    {
        $stmt = $mysqli->prepare("SELECT * FROM votos WHERE usuario_id
= ? AND post_id = ? AND tipo = 'like'");
        $stmt->bind_param("ii", $this->id, $postId);
        $stmt->execute();
        $result = $stmt->get_result();

        $jaDeuLike = $result->num_rows > 0;
        $stmt->close();
        return $jaDeuLike;
    }
```

```php
    // Verificar se o usuário já deu dislike em um post
    public function jaDeuDislike($mysqli, $postId)
    {
        $stmt = $mysqli->prepare("SELECT * FROM votos WHERE usuario_id
= ? AND post_id = ? AND tipo = 'dislike'");
        $stmt->bind_param("ii", $this->id, $postId);
        $stmt->execute();
        $result = $stmt->get_result();

        $jaDeuDislike = $result->num_rows > 0;
        $stmt->close();
        return $jaDeuDislike;
    }
}

?>
```

Resumo do Projeto: Código PHP e Scripts

1. Gerenciamento de Sessão e Usuário

- **Sessões:**
    - O projeto usa `session_start()` para gerenciar sessões de usuários.

Verifica se um usuário está logado com:

```
if (isset($_SESSION['usuario']) && $_SESSION['usuario'] instanceof Usuario)
```

    -
    - Quando nenhum usuário está logado, `$usuario` é definido como `null`.
- **Permissões Baseadas em Usuário:**
    - Apenas usuários logados podem interagir com posts (comentar, curtir, excluir).
    - Autores dos posts têm permissões adicionais, como excluir o próprio post e seus comentários.

2. Exibição e Manipulação de Posts

Exibição de Posts

Um `PostManager` gerencia a criação e exibição de posts. Ele é carregado da sessão:

```
$postManager = $_SESSION['postManager'];
```

-

Detalhes de um post específico são recuperados pelo ID, passado como parâmetro na URL:

```
$postId = $_GET['id'];
```

-

Exclusão de Posts

Apenas o autor do post pode excluí-lo:

```
if ($usuario && $usuario->getId() == $post->getAutor()) {
    $postManager->removerPost($postId);
    header("Location: index.php");
}
```

-

3. Interações com Likes e Dislikes

- **Likes e Dislikes:**
    - Usuários podem curtir ou descurtir posts clicando em links específicos.

A interação é registrada no banco de dados via `PostManager`, e a página é recarregada:

```
if (isset($_GET['curtir']) && $usuario) {
    $postManager->curtirPost($_GET['curtir']);
    header("Location: post_detalhes.php?id=$postId");
}
```

    -
- **Prevenção de Múltiplos Likes/Dislikes:**
    - O método `verificarVoto` impede múltiplos votos no mesmo post por um usuário.
    - Os botões de like/dislike mudam visualmente com base no estado do voto.

4. Comentários

Adição de Comentários

Um formulário `POST` permite que usuários logados adicionem comentários:

```
if ($_SERVER['REQUEST_METHOD'] === 'POST' &&
isset($_POST['comentario'])) {
    $novoComentario = new Comment(rand(), $textoComentario,
$usuario->getNome());
    $post->adicionarComentario($novoComentario);
}
```

-

Exclusão de Comentários

Autores dos comentários e autores do post podem excluir comentários:

```
if ($usuario && ($usuario->getId() == $post->getAutor() ||
$usuario->getId() == $comentario->getAutor())) {
    $post->removerComentario($comentarioId);
}
```

-

5. Controle de Erros

- **Erros Comuns:**
    - Mensagens de erro aparecem quando:

Um post não é encontrado:

```
if (!$post) {
    echo "Post não encontrado.";
    exit;
}
```

        - O usuário tenta acessar funcionalidades sem estar logado.

6. Scripts para Melhorar a Experiência do Usuário

Posição do Scroll
Para manter a posição do scroll após recarregar a página:

```
window.onload = function () {
    if (sessionStorage.getItem("scrollPosition") !== null) {
        window.scrollTo(0, sessionStorage.getItem("scrollPosition"));
    }
};
window.onscroll = function () {
    sessionStorage.setItem("scrollPosition", window.scrollY);
};
```

-

7. Estrutura do Banco de Dados

- **Posts:**
    - Tabelas armazenam informações básicas como ID, título, conteúdo e autor.
- **Comentários:**
    - Cada comentário está vinculado a um post e inclui informações como ID, texto, e autor.
- **Interações:**
    - Likes e dislikes são armazenados em uma tabela separada para controlar o estado de cada interação.

8. Fluxo de Controle

1. **Acessar Página Detalhes (post_detalhes.php):**
    - Recupera o ID do post na URL e valida sua existência.

    - Carrega o post e seus comentários.
2. **Interações (Like/Dislike):**
    - Identifica a ação via `$_GET` e atualiza o banco de dados.
3. **Adicionar ou Excluir Comentários:**
    - Processa o formulário de comentários via `$_POST`.
    - Exclui comentários verificando permissões.
4. **Redirecionamentos e Atualizações:**

Após cada ação (ex.: curtir/descurtir, excluir), o script redireciona para evitar reenvio do formulário:

```
header("Location: post_detalhes.php?id=$postId");
```