**Projeto: Agenda Tarefas**

**Código Comentado:**

listar\_tarefas\_simples.php

<?php

// Inclui o arquivo de conexão com o banco de dados para poder interagir com ele

require\_once 'conexao.php';

// Define a consulta SQL para selecionar as colunas 'id', 'titulo' e 'status' da tabela 'tarefas'

// As tarefas serão ordenadas pelo 'id' em ordem decrescente (as tarefas mais recentes primeiro)

$sql = "SELECT id, titulo, status FROM tarefas ORDER BY id DESC";

// Executa a consulta SQL na conexão com o banco de dados e armazena o resultado da consulta na variável '$resultado'

$resultado = $conexao->query($sql);

// Inicializa um array vazio chamado '$tarefas' para armazenar os dados das tarefas que serão buscados do banco de dados

$tarefas = array();

// Verifica se o número de linhas retornadas pela consulta ($resultado->num\_rows) é maior que zero,

// o que significa que foram encontradas tarefas no banco de dados

if ($resultado->num\_rows > 0) {

    // Inicia um loop 'while' para percorrer cada linha (cada tarefa) retornada pelo resultado da consulta

    while ($row = $resultado->fetch\_assoc()) {

        // A função 'fetch\_assoc()' retorna a linha atual do resultado como um array associativo

        // onde as chaves são os nomes das colunas da tabela.

        // Adiciona o array associativo (que representa uma tarefa) ao final do array '$tarefas'

        $tarefas[] = $row;

    }

}

// Fecha a conexão com o banco de dados para liberar os recursos utilizados

$conexao->close();

// Define o cabeçalho da resposta HTTP para informar ao navegador que o conteúdo que está sendo enviado é do tipo JSON (JavaScript Object Notation)

header('Content-Type: application/json');

// Codifica o array PHP '$tarefas' para o formato JSON e o envia como saída para o navegador

echo json\_encode($tarefas);

?>

<?php

// Inclui o arquivo de conexão com o banco de dados para poder interagir com ele

require\_once 'conexao.php';

// Verifica se o método da requisição HTTP é POST, o que indica que dados foram enviados para o servidor

if ($\_SERVER['REQUEST\_METHOD'] === 'POST') {

    // Obtém o valor do campo 'titulo' enviado através do método POST

    $titulo = $\_POST['titulo'];

    // Obtém o valor do campo 'status' enviado através do método POST

    $status = $\_POST['status'];

    // Verifica se o campo 'titulo' não está vazio

    if (!empty($titulo)) {

        // Define a consulta SQL para inserir um novo registro na tabela 'tarefas'

        // Os valores de 'titulo' e 'status' serão inseridos de forma segura usando prepared statements

        $sql = "INSERT INTO tarefas (titulo, status) VALUES (?, ?)";

        // Prepara a consulta SQL para execução

        $stmt = $conexao->prepare($sql);

        // Vincula os parâmetros (os valores de $titulo e $status) aos placeholders (?) na consulta SQL

        // 's' indica que os parâmetros são strings

        $stmt->bind\_param("ss", $titulo, $status);

        // Executa a consulta preparada

        if ($stmt->execute()) {

            // Se a execução da consulta for bem-sucedida, cria um array associativo com a chave 'success' definida como true

            // e uma mensagem de sucesso

            echo json\_encode(['success' => true, 'message' => 'Tarefa cadastrada!']);

        } else {

            // Se houver um erro na execução da consulta, cria um array associativo com a chave 'success' definida como false

            // e uma mensagem de erro

            echo json\_encode(['success' => false, 'error' => 'Erro ao cadastrar.']);

        }

        // Fecha a declaração preparada para liberar os recursos

        $stmt->close();

    } else {

        // Se o campo 'titulo' estiver vazio, cria um array associativo com a chave 'success' definida como false

        // e uma mensagem de erro indicando que o título é obrigatório

        echo json\_encode(['success' => false, 'error' => 'Título é obrigatório.']);

    }

} else {

    // Se o método da requisição não for POST, cria um array associativo com a chave 'success' definida como false

    // e uma mensagem de erro indicando que o método de requisição é inválido

    echo json\_encode(['success' => false, 'error' => 'Método inválido.']);

}

// Fecha a conexão com o banco de dados para liberar os recursos

$conexao->close();

?>

**script\_simples.js**

$(document).ready(function() { // Garante que o código jQuery só execute após o DOM (Document Object Model) estar totalmente carregado e pronto

    // Função para carregar a lista de tarefas do servidor via AJAX e atualizar a tabela na página

    function atualizarLista() {

        $.ajax({ // Inicia uma requisição AJAX do jQuery

            url: 'listar\_tarefas\_simples.php', // Define a URL do script PHP que fornecerá a lista de tarefas

            type: 'GET', // Define o método da requisição como GET (para obter dados)

            dataType: 'json', // Define o tipo de dados esperado na resposta como JSON (JavaScript Object Notation)

            success: function(data) { // Função a ser executada quando a requisição for bem-sucedida e o servidor retornar os dados

                $('#tarefas-lista').empty(); // Limpa o conteúdo atual da tbody (corpo da tabela) com o ID 'tarefas-lista' para evitar duplicações

                $.each(data, function(index, tarefa) { // Itera sobre cada objeto 'tarefa' dentro do array 'data' recebido do servidor

                    // Adiciona uma nova linha (<tr>) à tbody da tabela com os dados de cada tarefa

                    $('#tarefas-lista').append(`<tr><td>${tarefa.id}</td><td>${tarefa.titulo}</td><td>${tarefa.status}</td></tr>`);

                });

            },

            error: function() { // Função a ser executada se houver algum erro na requisição AJAX (por exemplo, falha na comunicação com o servidor)

                $('#tarefas-lista').html('<tr><td colspan="3">Erro ao carregar tarefas.</td></tr>'); // Exibe uma mensagem de erro na tabela, ocupando todas as colunas

            }

        });

    }

    // Chama a função 'atualizarLista()' assim que a página é carregada para exibir as tarefas iniciais

    atualizarLista();

    // Adiciona um evento de 'submit' ao formulário com o ID 'nova-tarefa'

    $('#nova-tarefa').on('submit', function(e) {

        e.preventDefault(); // Impede o comportamento padrão de envio do formulário, que faria a página recarregar

        $.ajax({ // Inicia uma requisição AJAX do jQuery

            url: 'cadastrar\_tarefa\_simples.php', // Define a URL do script PHP que irá processar o cadastro da nova tarefa

            type: 'POST', // Define o método da requisição como POST (para enviar dados para o servidor)

            dataType: 'json', // Define o tipo de dados esperado na resposta do servidor como JSON

            data: $(this).serialize(), // Serializa os dados do formulário (campos e seus valores) em um formato adequado para envio (query string)

            success: function(response) { // Função a ser executada quando a requisição for bem-sucedida e o servidor responder

                $('#msg').html(''); // Limpa qualquer mensagem anterior na div com o ID 'msg'

                if (response.success) { // Verifica se a propriedade 'success' na resposta JSON é true (indicando sucesso no cadastro)

                    $('#msg').html('<span style="color: green;">' + response.message + '</span>').fadeIn().delay(1500).fadeOut(); // Exibe uma mensagem de sucesso em verde, com efeito de fade-in e fade-out após 1.5 segundos

                    $('#nova-tarefa')[0].reset(); // Reseta (limpa) os campos do formulário

                    atualizarLista(); // Chama a função para recarregar e exibir a lista de tarefas atualizada

                } else { // Se a propriedade 'success' na resposta JSON for false (indicando erro no cadastro)

                    $('#msg').html('<span style="color: red;">' + response.error + '</span>').fadeIn().delay(2000).fadeOut(); // Exibe uma mensagem de erro em vermelho, com efeito de fade-in e fade-out após 2 segundos

                }

            },

            error: function() { // Função a ser executada se houver um erro na comunicação com o servidor

                $('#msg').html('<span style="color: red;">Erro no servidor.</span>').fadeIn().delay(2000).fadeOut(); // Exibe uma mensagem de erro genérico do servidor em vermelho, com efeito de fade-in e fade-out após 2 segundos

            }

        });

    });

});