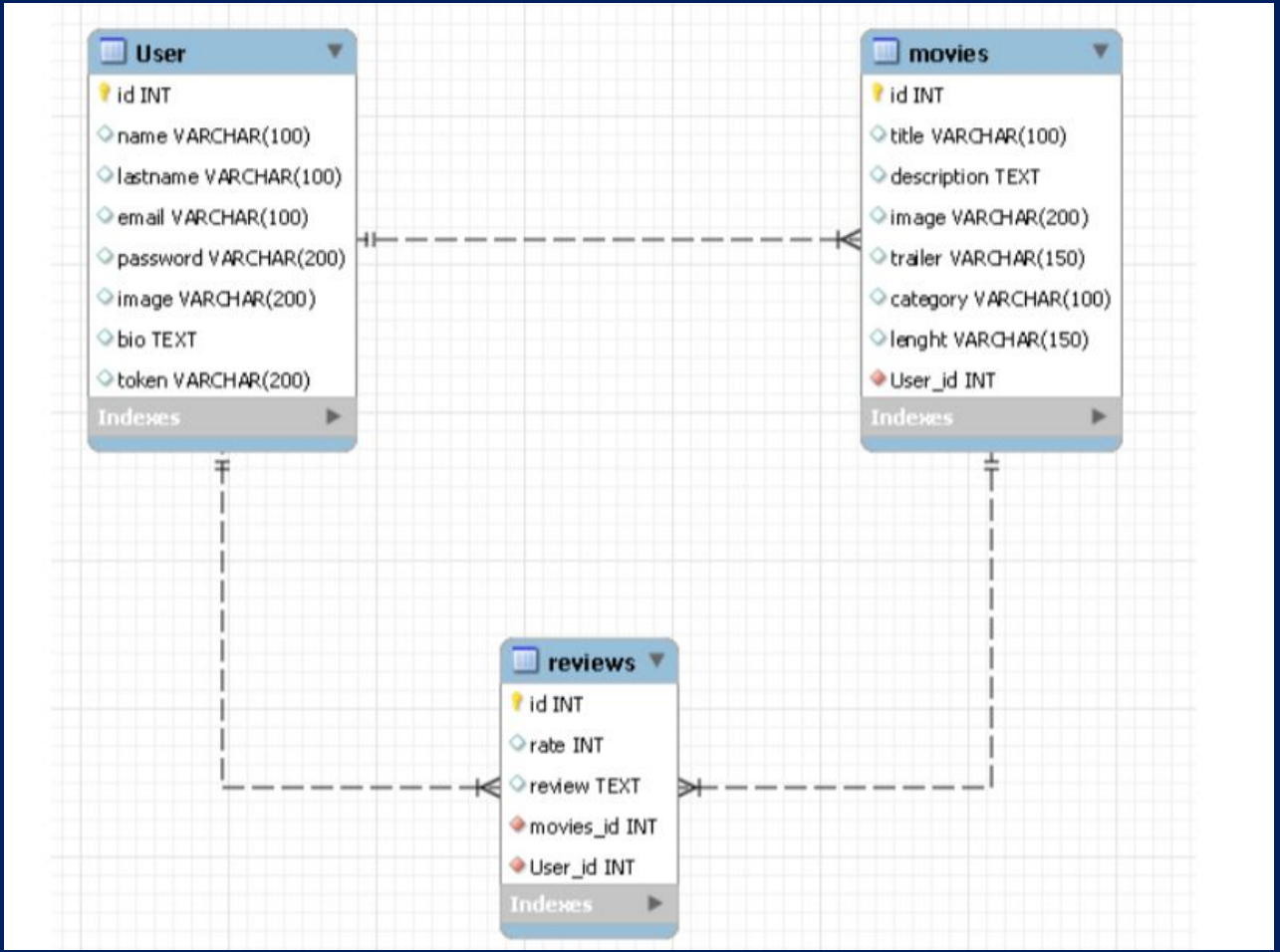
Programação Avançada / Programação Mobile

Alexandre Erdmann Silva

Conteúdo

Sumário


User
id INT
name VARCHAR(100)
lastname VARCHAR(100)
email VARCHAR(100)
password VARCHAR(200)
image VARCHAR(200)
bio TEXT
token VARCHAR(200)
Indexes
movies
id INT
title VARCHAR(100)
description TEXT
image VARCHAR(200)
trailer VARCHAR(150)
category VARCHAR(100)
lenght VARCHAR(150)
User_id INT
Indexes
reviews
id INT
rate INT
review TEXT
movies_id INT
User_id INT
Indexes

Criando a tabela users

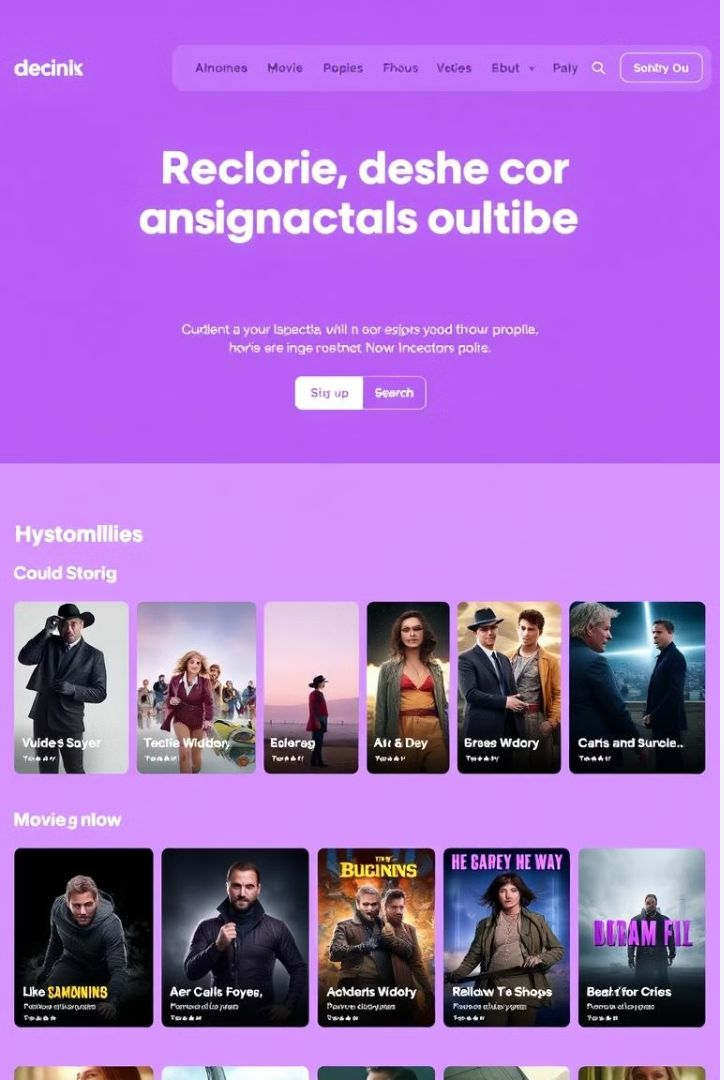
```
CREATE TABLE users(
    id INT(11) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100),
    lastname VARCHAR(100),
    email VARCHAR(200),
    password VARCHAR(200),
    image VARCHAR(200),
    token VARCHAR(200),
    bio TEXT
);
```

Criando a tabela movies

```
CREATE TABLE movies(
    id INT(11) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(100),
    description TEXT,
    image VARCHAR(200),
    trailer VARCHAR(150),
    category VARCHAR(50),
    length VARCHAR(50),
    users_id INT(11) UNSIGNED,
    FOREIGN KEY(users_id) REFERENCES users(id)
)
```

Criando a tabela movies

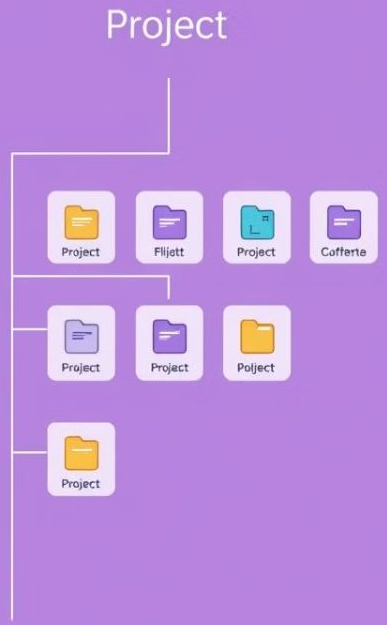
```
CREATE TABLE reviews(
      id  INT(11)  UNSIGNED  AUTO_INCREMENT  PRIMARY
KEY,
    rating INT,
    review TEXT,
    users_id INT(11) UNSIGNED,
    movies_id INT(11) UNSIGNED,
    FOREIGN KEY(users_id) REFERENCES users(id),
    FOREIGN KEY(movies_id) REFERENCES movies(id)
);
```



Estruturando um Projeto PHP: Moviestar

Bem-vindos à aula de estruturação de projetos PHP! Neste tutorial, vamos construir um projeto web completo, utilizando o framework PHP e as melhores práticas de desenvolvimento. Iremos construir o Moviestar, um aplicativo web que simula um serviço de streaming de filmes, focando em organização de arquivos, conexão com banco de dados e código limpo e eficiente. Acompanhe este passo a passo detalhado para construir seu próprio projeto PHP do zero!

por Alexandre Erdmann



Organização de Pastas e Arquivos

Pasta raiz do projeto

A pasta raiz do projeto deve conter todos os arquivos e pastas necessários para a aplicação web. É a pasta principal do projeto.

Pasta public

Esta pasta contém os arquivos que serão acessíveis ao usuário, como arquivos HTML, CSS, JavaScript e imagens.

Pasta src

A pasta src contém o código fonte PHP da aplicação. É aqui que serão armazenados os arquivos com a lógica do projeto.

Pasta config

A pasta config contém os arquivos de configuração do projeto, como as credenciais de acesso ao banco de dados.

Conexão com o Banco de Dados usando PDO

Introdução ao PDO

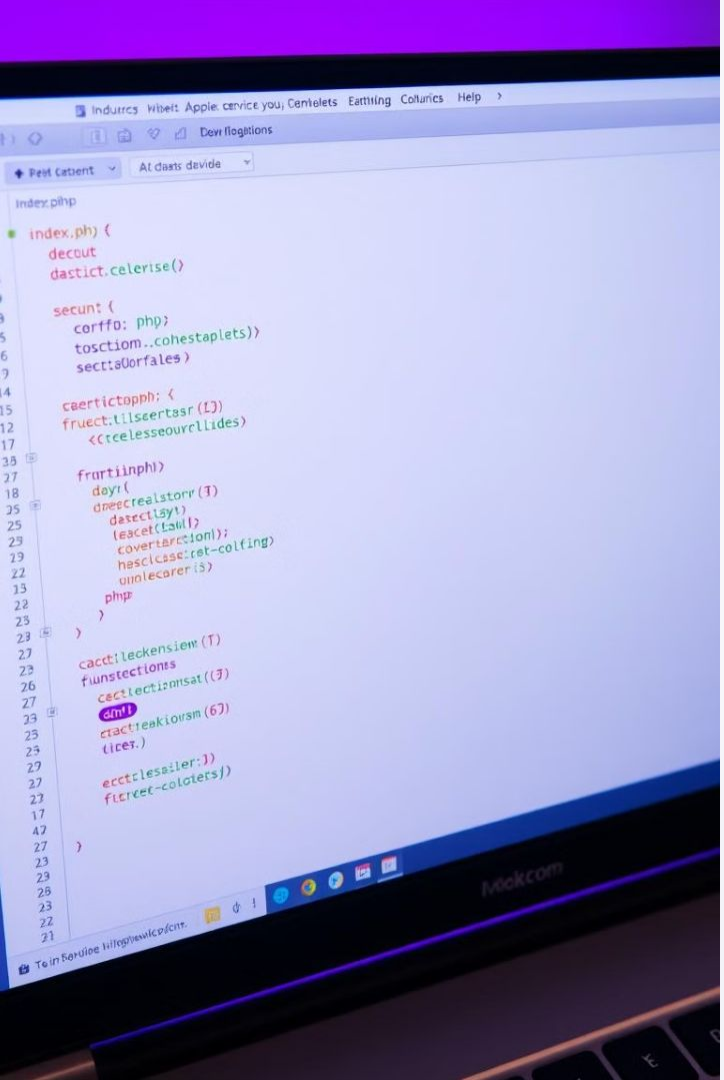
PDO (PHP Data Objects) é uma interface de acesso a dados que fornece uma maneira consistente de conectar e interagir com diferentes bancos de dados.

Vantagens do PDO

- Suporte a diversos bancos de dados.
- Prevenção de SQL Injection.
- Melhor organização do código.

Conexão com o Banco de Dados

A conexão com o banco de dados é feita através de uma classe PDO que será criada no arquivo config.php. Esta classe será responsável por gerenciar a conexão com o banco de dados e fornecer métodos para realizar operações de CRUD (Create, Read, Update, Delete).



Criar os arquivos básicos: index.php, config.php, functions.php

1 **index.php**

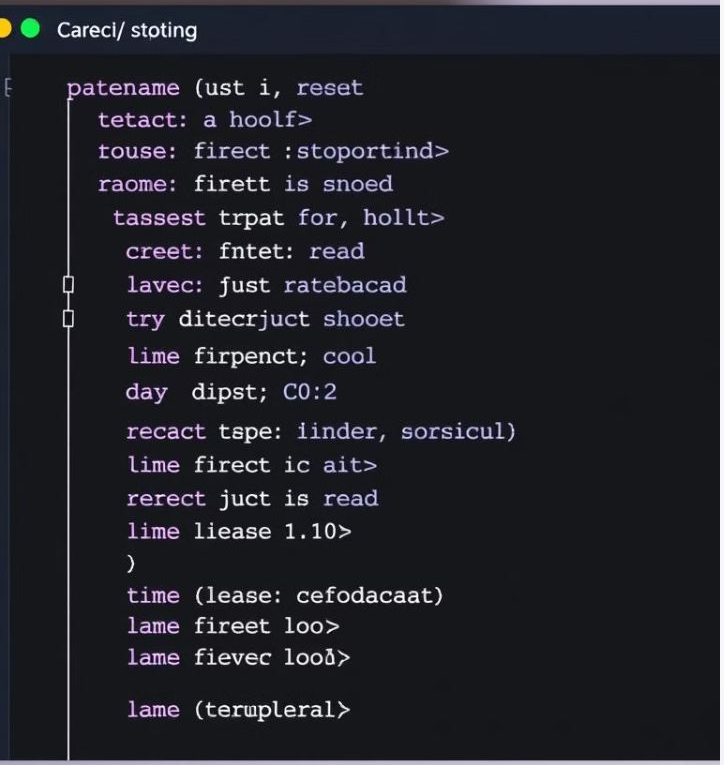
O arquivo index.php é a página inicial do projeto e é o ponto de entrada para a aplicação. Ele será responsável por renderizar o conteúdo HTML e chamar as funções necessárias para o funcionamento do site.

2 **config.php**

O arquivo config.php contém as configurações do projeto, como as credenciais de acesso ao banco de dados, configurações de ambiente e outras informações essenciais.

3 **functions.php**

O arquivo functions.php contém as funções que serão utilizadas em diferentes partes do projeto. Essas funções podem ser utilizadas para realizar tarefas como validação de dados, acesso ao banco de dados e processamento de informações.



Configuração do arquivo config.php

Nome da variável	Descrição
DB_HOST	Endereço do servidor do banco de dados
DB_NAME	Nome do banco de dados
DB_USER	Nome de usuário do banco de dados
DB_PASSWORD	Senha do banco de dados

Implementando a conexão com o banco de dados

1

Criando a classe de conexão

Crie uma classe chamada Database com um método construtor que receberá as credenciais do banco de dados e irá estabelecer a conexão.

2

Gerenciando a conexão

A classe Database também deve ter métodos para abrir e fechar a conexão com o banco de dados.

3

Preparando instruções SQL

Para evitar SQL Injection, utilize o método prepare() do PDO para preparar as instruções SQL antes de executá-las.

```
18
38          {pr Cave 15 therageional.Aperts(unltinifile)
13
20          [] Apceo 15 theragefifr Papecr.(unlltfiteBour.bescive()
24
27          {pr Cave 15 therageional.Apert.(onltinifion"
128
116         {] Apceo 15 theragefifr Papecr.(unlettit.raleave()
111     )>
```

Criando as funções no arquivo functions.php

Funções de pesquisa

As funções de pesquisa são responsáveis por buscar informações no banco de dados, como encontrar filmes por título, gênero ou ator.

Funções de manipulação de filmes

Essas funções são responsáveis por adicionar, editar, remover e listar filmes no banco de dados.

Funções de autenticação de usuários

Funções para gerenciar o login e logout de usuários, além de verificar a autenticação do usuário.

Funções de configuração

Funções para definir configurações do sistema, como idioma, tema e outras preferências.

Construindo as páginas do Moviestar

Página inicial

A página inicial do Moviestar deve apresentar um layout atrativo, com destaque para filmes em destaque, categorias populares e uma área de pesquisa.

1

Página de perfil do usuário

A página de perfil do usuário deve permitir que o usuário visualize seus dados, histórico de filmes assistidos, lista de favoritos e outras informações relevantes.

3

2

Página de detalhes do filme

Ao clicar em um filme, o usuário deve ser redirecionado para uma página com informações detalhadas sobre o filme, como sinopse, elenco, trailer e avaliações.

Utilizando boas práticas de programação PHP

1 Indentação e formatação

Use uma indentação consistente para tornar o código mais legível e fácil de entender.

2 Nomes de variáveis e funções

Use nomes descritivos para variáveis e funções, facilitando a compreensão do código.

3 Comentários

Adicione comentários explicativos para explicar o propósito de cada parte do código.

4 Modularização

Divida o código em funções e classes para facilitar a manutenção e reutilização de código.

Conclusão e Próximos Passos

Nesta aula, você aprendeu a estruturar um projeto PHP completo, desde a organização de pastas e arquivos até a conexão com o banco de dados. Ao longo deste processo, você aprendeu a utilizar PDO para se conectar a um banco de dados de forma segura e eficiente, a criar funções reutilizáveis para realizar diversas tarefas e a aplicar boas práticas de programação para construir código limpo e legível.

Agora, com o Moviestar pronto, você pode explorar outros recursos, como:

- Adicionar novas funcionalidades, como a criação de contas de usuário, listas de reprodução personalizadas e integração com plataformas de pagamento.
- Implementar um sistema de recomendações de filmes, utilizando algoritmos de aprendizado de máquina para oferecer sugestões relevantes aos usuários.
- Integrar o Moviestar com outras plataformas, como redes sociais, para aumentar o alcance do aplicativo.

Atividade 3

Sendo assim você deve montar na pasta com o seu nome_sobrenome no htdocs as pastas que serão utilizadas nesse projeto, elas são apresentadas a seguir:

- CSS
- DAO
- img
  - movies
  - users
- models
- templates

Atividade 3

1 – Arquivo index.php

Insira a estrutura base do html

a) Doctype
b) Idioma pt_BR
c) Cabeçalho <head></head>
d) Meta charset com valor UTF-8
e) Meta name viewport com o conteúdo seguindo a largura do dispositivo (width) e escala igual a 1.0
f) Crie um título para a página chamada Filmes
g) Faça o link do logo e dos css a seguir:

Atividade 3

a. https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/5.2.3/css/bootstrap.css
b. css/styles.css
c. https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/5.2.3/css/bootstrap.css
d. https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/5.2.3/css/bootstrap.css
e. https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.3.0/css/all.min.css" integrity="sha512-SzlrxWUlpfuzQ+pcUCosxcglQRNAq/DZjVsC0lE40xsADsfeQoEypE+enwcOiGjk/bSuGGKHEyjSoQ1zVisanQ==" crossorigin="anonymous" referrerpolicy="no-referrer" />

Atividade 3

a. https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/5.2.3/css/bootstrap.css
b. css/styles.css
c. https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/5.2.3/css/bootstrap.css
d. https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/5.2.3/css/bootstrap.css
e. https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.3.0/css/all.min.css" integrity="sha512-SzlrxWUlpfuzQ+pcUCosxcglQRNAq/DZjVsC0lE40xsADsfeQoEypE+enwcOiGjk/bSuGGKHEyjSoQ1zVisanQ==" crossorigin="anonymous" referrerpolicy="no-referrer" />

Atividade 4

Enunciado: Crie um arquivo chamado globals.php, com a variável dentro da sua pasta com o seu nome e sobrenome.

```
<?php

 session_start();

 $BASE_URL = "http://" . $_SERVER["SERVER_NAME"] . dirname($_SERVER["REQUEST_URI"]."?") . "/";
```

Crie um arquivo db.php para realizar a conexão ao banco de dados.
Insira as variáveis:
$db_name = “nome_do_seu_banco_de_dados”;
$db_host = “localhost”;
$db_user = “root”;
$db_pass=””; (senha do banco de dados em branco)

Atividade 4

```
$conn->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
$conn->setAttribute(PDO::ATTR_EMULATE_PREPARES, false);
```

No Arquivo index.php insira o código:

```
<?php
        require_once(“db.php”);
        require_once(“globals.php”);
?>
```

OBRIGADO!