**1. Sobre o Projeto**

O "Livros-app" é um exercício prático para aplicar os conhecimentos de PHP, MySQL e PDO. Ele me permitiu entender como funciona um sistema CRUD (Criar, Ler, Atualizar, Excluir) na prática, desde a criação do banco de dados até a interface do usuário.

**2. O Que o Sistema Faz (Funcionalidades)**

* **Adicionar Livro:** Consigo cadastrar novos livros, informando o título, autor, ano de publicação e o ISBN.
* **Listar Livros:** Ele mostra uma lista de todos os livros que já cadastrei. Cada livro na lista tem um link para eu poder editar.
* **Editar Livro:** Se eu precisar corrigir alguma informação de um livro, posso clicar em "Editar" e mudar os dados.
* **Excluir Livro:** Também consigo remover um livro do cadastro se ele não for mais necessário.
* **Validação de Dados:** Tentei colocar umas validações para garantir que os dados que eu digito estejam corretos, tanto no navegador (com JavaScript) quanto no servidor (com PHP). Isso ajuda a evitar que dados estranhos sejam salvos.

**3. As Ferramentas que Usei (Tecnologias e Versões)**

Para desenvolver e rodar este projeto, utilizei as seguintes tecnologias e suas versões (ou versões compatíveis):

* **PHP:** Linguagem principal do projeto.
  + **Versão Utilizada/Recomendada:** PHP 8.0 ou superior (com a extensão **pdo\_mysql** habilitada).
* **MySQL:** O banco de dados onde as informações dos livros são guardadas.
  + **Versão Utilizada/Recomendada:** MySQL 8.0 ou superior.
* **PDO:** Uma forma mais segura de conectar o PHP ao MySQL.
  + **Versão:** Faz parte do PHP, então a versão é a mesma do PHP que você estiver usando.
* **JavaScript:** Para fazer umas validações rápidas nos formulários.
  + **Versão:** ECMAScript 5 (ES5) ou superior (compatível com a maioria dos navegadores modernos).
* **HTML:** Para montar a estrutura das páginas web.
  + **Versão:** HTML5.
* **Servidor Web:** Para rodar os arquivos PHP.
  + **Exemplos:** Apache HTTP Server (versão 2.4 ou superior) ou Nginx (versão 1.10 ou superior). Geralmente vêm empacotados em soluções como XAMPP, WAMP ou MAMP.

**4. Como o Projeto Está Organizado (Estrutura)**

Eu tentei organizar o código em pastas para ficar mais fácil de entender:

* **class/**: Aqui estão as classes PHP.
  + **Livro.php**: É como se fosse a "carteira de identidade" do livro, definindo o que um livro tem (título, autor, ano, ISBN).
  + **LivroRepository.php**: Essa classe é responsável por conversar com o banco de dados, fazendo as operações de adicionar, listar, editar e excluir livros.
* **assets/js/**: Onde guardo os arquivos JavaScript.
  + **validacao.js**: Tem as funções JavaScript que validam o que eu digito nos campos do formulário.
* **views/**: As páginas que o usuário vê.
  + **adicionar.php**: A página com o formulário para cadastrar um livro novo.
  + **editar.php**: A página para mudar os dados de um livro ou excluí-lo.
* **index.php**: A página inicial do sistema, que mostra a lista de todos os livros.
* **sistema\_livros\_livros.sql**: O arquivo SQL para criar a tabela **livros** no banco de dados.
* **README.md**: Este arquivo de documentação que você está lendo!

**5. O Que Funcionou Bem (Pontos Fortes)**

* **Todas as Operações Básicas (CRUD):** Consegui fazer o sistema adicionar, listar, editar e excluir livros sem problemas. Isso foi o principal objetivo e funcionou direitinho.
* **Validação de Dados:** As validações em JavaScript (no navegador) e em PHP (no servidor) estão funcionando. Se eu digitar algo errado ou com caracteres estranhos, o sistema tenta corrigir ou me avisa.
* **Segurança com PDO:** Usar o PDO com "prepared statements" foi importante para aprender a proteger o sistema contra injeção SQL.
* **Organização do Código:** A divisão em classes (**Livro** e **LivroRepository**) e em pastas (**class/**, **views/**) me ajudou a manter o código mais organizado e fácil de entender.
* **Redirecionamento:** Depois de adicionar ou editar um livro, o sistema me redireciona para a lista, o que é bom para não reenviar o formulário sem querer.

**6. O Que Poderia Melhorar (Pontos Fracos e Próximos Passos)**

* **Tratamento de Erros do Banco de Dados:** Se der algum erro no banco (tipo, tentar cadastrar um ISBN que já existe), o sistema não mostra uma mensagem muito amigável.
  + **Como melhorar:** Eu deveria capturar esses erros e mostrar uma mensagem mais clara para o usuário, tipo "Este ISBN já existe!".
* **Validação de ISBN Mais Completa:** A validação do ISBN só verifica se tem números e hífens.
  + **Como melhorar:** Seria legal validar se o ISBN tem o número certo de dígitos (10 ou 13) e se o formato é válido.
* **Código Repetido:** Percebi que a conexão com o banco de dados e algumas validações estão repetidas em vários arquivos.
  + **Como melhorar:** Posso criar um arquivo de configuração para a conexão e funções de validação que eu possa usar em qualquer lugar, evitando repetição.
* **Mensagens de Sucesso:** Quando eu adiciono ou edito um livro, ele só redireciona, mas não aparece uma mensagem "Livro adicionado com sucesso!".
  + **Como melhorar:** Posso usar sessões para guardar essas mensagens e mostrá-las na próxima página.
* **Visual do Sistema:** O sistema é bem básico visualmente.
  + **Como melhorar:** Adicionar um arquivo CSS para deixar as páginas mais bonitas e fáceis de usar.
* **Segurança das Credenciais:** As informações de login do banco de dados estão direto no código.
  + **Como melhorar:** Colocar essas informações em um arquivo separado e seguro, que não seja acessível pela web.

**7. Conclusão**

Este projeto foi um ótimo aprendizado para mim. Consegui colocar em prática muitos conceitos de desenvolvimento web e orientação a objetos. Sei que ainda tem bastante coisa para melhorar, mas estou feliz com o que consegui fazer até agora e já tenho uma lista de próximos passos para deixar o "Livros-app" ainda melhor!

**Arquivo para Download:**

Você pode copiar o projeto completo através do link do GitHub:

[**https://github.com/GalaxyLuckas/Livros-app.git**](https://github.com/GalaxyLuckas/Livros-app.git)

**Instruções para Download:**

1. Copie o link acima.
2. Abra o Git Bash Here na pasta desejada.
3. Escreva “git clone” e cole o link.

Após o download, siga as instruções no **README.md** (que está incluído no ZIP) para configurar o banco de dados e rodar o projeto no seu ambiente local (XAMPP, WAMP, etc.).