# **Documentação de Uso da API**

## **Visão Geral**

Este projeto consiste em uma aplicação que consome dados da API pública de Star Wars e exibe informações detalhadas sobre os filmes da franquia. O backend foi desenvolvido utilizando PHP, MySQL com POO (Programação Orientada a Objetos) no padrão MVC. No frontend, foi utilizado Bootstrap para a criação de uma interface responsiva e JQuery/AJAX para as interações dinâmicas.

A aplicação permite aos usuários visualizar um catálogo de filmes, acessar detalhes sobre cada filme, como título, número do episódio, sinopse, diretor, produtor, personagens e até mesmo a idade dos filmes. As interações são registradas no banco de dados, com logs sobre as consultas e possíveis erros de retorno da API. também salva os registros em um arquivo de log, dando mais alternativas de monitoramento da API

## **Tecnologias e Ferramentas Utilizadas**

* **Backend**: PHP 7.4, MySQL, POO (Programação Orientada a Objetos), padrão MVC ( Model, View, Controller)
* **Frontend**: HTML, CSS, Bootstrap, JQuery, AJAX, SweetAlerts2, dataTable,
* **Banco de Dados**: MySQL, utilizando PDO, o que permite conectar com multiplos bancos de dados e mais segurança a sql injection e mais alinhado as boas práticas de POO
* **Ferramentas**: cURL (para consumo da API externa), VSCode.

## **Estrutura do Projeto**

* **/app**: Contém os diretórios controllers, models, views, services, config, database
* **/public**: Contém os diretórios imgs, css, js, fonts
* **/logs**: Contém o arquivo logs.txt para registros de interações com a API
* **/routes**: Contém as rotas da aplicação
* **/readme.md**: Instruções de instalação e detalhes do projeto
* **/database/dump**: Contém o arquivo de dump do banco de dados

## **Instruções de Instalação**

Clone o repositório:  
git clone https://github.com/dimagno/starwars

Crie um arquivo .env na raiz do projeto com as seguintes informações:  
  
DB\_USER = 'root'

DB\_HOST = 127.0.0.1

DB\_NAME = sw

DB\_PASS =

Crie uma base de dados com o nome definido em DB\_NAME no arquivo .env e execute o dump do banco localizado em /database/dump.

## **Endpoints da Api**

### **GET /starwars/api/filmes**

Retorna a lista de filmes ordenada por data de lançamento.  
Exemplo de resposta:

[

{

"fields": {

"title": "A New Hope",

"release\_date": "1977-05-25",

"director": "George Lucas",

"producer": "Gary Kurtz, Rick McCallum",

"episode\_id": 4,

"characters": [

"/api/people/1",

"/api/people/2"

],

"url": "/api/films/1"

}

},

...

]

### **GET /starwars/api/filme/{id}**

Retorna os detalhes de um filme específico, incluindo nome, número do episódio, sinopse, data de lançamento, diretor(a), produtor(es), nome dos personagens e a idade do filme em anos, meses e dias. A resposta é semelhante a /starwars/api/filmes, porém contendo apenas um item especificado

## **Paginas do site:**

/ **/starwars :** retorna a página inicial contendo um carrossel de imagens em alta definição dos filmes da saga e uma descrição sobre o conteúdo do site.

### **/starwars/filmes :** retorna a página contendo a lista dos filmes com as informações da api tratadas e prontas para exibição

## **/starwars/filme/id :** retorna a página de detalhes do filme, contendo informações do filme selecionado como, diretor, produtor, data de lançamento e idade do filme (em dias, meses e anos).

### **/starwars/log :** retorna a página contendo a tabela com as informações de atividades (logs) com filtros de pesquisas. requer senha de administrador para visualizar. (senha: 1234)

### **/starwars/sobre:** retorna a página contendo informação geral sobre o projeto

## 

## **Logs**

As interações com a API externa são registradas em um arquivo de log (/logs/api\_logs.txt). Cada entrada contém as seguintes informações:

* **Data/Hora**: Quando a solicitação foi realizada.
* **Tipo**: Tipo da solicitação (erro, informação).
* **Código**: Código de resposta da API.
* **Mensagem**: Mensagem detalhada da solicitação ou erro.

## **Melhorias Aplicadas**

1. **Responsividade do Site**A interface foi adaptada para garantir que o layout seja adequado em diferentes tamanhos de tela, proporcionando uma boa experiência para usuários em dispositivos móveis e desktops.
2. **Busca Secundária Após o Carregamento dos Detalhes do Filme**Implementação de uma funcionalidade de busca secundária, que realiza chamadas AJAX para buscar informações complementares de forma assíncrona, como naves, planetas e espécies que aparecem no filme, diminuido consideravelmente o tempo de consulta das informações e o tempo de espera dos dados para visualizar a página. Assim que a consulta é concluída, o juery se encarrega de inseri-las no DOM dinamicamente.
3. **Página de logs**Adição de uma página para consulta de logs do banco de dados, com filtros para qualquer campo utilizando a biblioteca dataTable. A página solicita uma senha para visualizar. A senha é 1234

## **Considerações Finais**

Este projeto foi desenvolvido com foco em boas práticas de programação, com um backend em PHP utilizando o padrão MVC, integração com banco de dados via PDO, e uma interface amigável e responsiva no frontend. O uso de AJAX e cURL facilita a interação com a API externa, proporcionando uma experiência dinâmica e eficiente para os usuários. A implementação das melhorias, como busca secundárias assíncronas, página de visualização de logs com filtros, responsividade, e uso de imagens que a API não fornece, contribui para tornar a navegação mais rica e interativa.

Com isso, o projeto oferece uma solução completa para consultar os filmes de Star Wars, exibindo suas informações de maneira organizada e agradável para o usuário, além de manter logs para acompanhar o funcionamento da aplicação.