

Instituto Politécnico de Setúbal
Curso Técnico de Gestão e Programação de Sistemas Informáticos

Manual Técnico



**CTeSP em Tecnologias e Programação de Sistemas de Informação
Programação e Integração de Serviços**

Diogo Duarte, Mateus Duarte, Tomás Faneca

Docente: Fabiana Glória

Conteúdo

1. Introdução e Tecnologias.....	3
2. Estrutura do Projeto (File System)	4
3. Arquitetura de Dados (MySQL)	4
Entidades Principais	4
4. Web Services e Especificações REST.....	5
4.1 Serviços Elementares (Gestão Interna)	5
4.2 Serviços Compostos (Integração API TMDB)	5
5. Instalação e Configuração	6
6. Segurança e Boas Práticas	6

1. Introdução e Tecnologias

O presente documento descreve o **manual técnico da plataforma MovieTrain**, desenvolvida no âmbito da unidade curricular **Programação e Integração de Serviços**. O sistema tem como objetivo a **gestão e visualização de filmes, séries, pessoas (atores e realizadores)**, permitindo a integração de dados locais com informação proveniente da API externa **The Movie Database (TMDB)**.

A solução segue uma arquitetura **cliente-servidor**, utilizando tecnologias modernas de desenvolvimento web:

- **Runtime:** Node.js
- **Framework Web:** Express.js
- **Base de Dados:** MySQL
- **Frontend:** HTML5, CSS3 e JavaScript
- **Comunicação:** JSON (RESTful APIs)
- **Integração Externa:** API The Movie Database (TMDB)
- **Versionamento:** Git (GitHub)
- **IDE:** Visual Studio Code

2. Estrutura do Projeto (File System)

O projeto encontra-se organizado segundo o princípio de **separação de responsabilidades**, promovendo manutenção, escalabilidade e clareza do código.

3. Arquitetura de Dados (MySQL)

A base de dados **projetopis** foi desenhada para suportar os principais componentes da aplicação.

Entidades Principais

- **Utilizador** – Gestão de contas e permissões (admin/normal).
- **Pessoa** – Atores e diretores associados a filmes.
- **Filme** – Filmes e séries com metadados locais.
- **Genero** – Classificação dos conteúdos.
- **Review** – Avaliações e críticas dos utilizadores.
- **Favorito** – Filmes marcados como favoritos.
- **FilmePessoa** – Relação entre filmes e pessoas (papel no elenco).
- **FilmeGenero** – Associação entre filmes e géneros.
- **Pegi** – Classificação etária.

A modelação utiliza **chaves primárias, estrangeiras e tabelas de relação**, garantindo integridade referencial.

4. Web Services e Especificações REST

A aplicação utiliza uma arquitetura baseada em **Web Services REST**, com endpoints organizados por domínio funcional.

4.1 Serviços Elementares (Gestão Interna)

Estes serviços comunicam diretamente com a base de dados MySQL.

Pessoas

- GET /api/pessoas – Lista todas as pessoas
- GET /api/pessoas/:id – Detalhes de uma pessoa
- POST /api/pessoas – Criação de nova pessoa
- PUT /api/pessoas/:id – Atualização de dados
- DELETE /api/pessoas/:id – Remoção de pessoa

Filmes

- GET /api/filmes
- GET /api/filmes/:id
- POST /api/filmes
- PUT /api/filmes/:id
- DELETE /api/filmes/:id

Relação Filme–Pessoa

- GET /api/filmePessoa/:id – Retorna uma pessoa com os filmes em que participou (ator/diretor)

4.2 Serviços Compostos (Integração API TMDB)

A plataforma integra a **API externa do TMDB**, permitindo:

- Pesquisa de filmes e séries
- Importação de metadados (título, poster, sinopse, data de lançamento)
- Consulta de elenco e realizadores

Exemplo de Endpoint:

- GET /api/tmdb/search?query=harry&type=movie

A comunicação é realizada através de **axios**, com a chave da API armazenada em variáveis de ambiente.

💡 Documentação oficial TMDB:
<https://developer.themoviedb.org/docs/getting-started>

5. Instalação e Configuração

- 1. Clonar o Repositório:
<https://github.com/TomasFaneca28/Projeto-PIS.git>
- 2. Dependências: Executar npm install
- 3. Configurar: Executar o "BD_ProjetoPIS.sql" no Mysql
- 4. API: Criar um ficheiro .env na raiz do projeto com
TMDB_API_KEY = Insira aqui a sua key
- 5. Executar: node index.js na consoloa do VSCode

6. Segurança e Boas Práticas

- **Separação de camadas:** frontend e backend independentes
- **Uso de variáveis de ambiente:** credenciais sensíveis fora do código
- **Validação de erros:** respostas HTTP adequadas (404, 500)
- **Integridade de dados:** uso de chaves estrangeiras no MySQL
- **Organização modular:** rotas separadas por domínio
- **Escalabilidade:** fácil extensão para autenticação e permissões futuras