Introdução

A avaliação da componente prática da disciplina de Programação na Internet será realizada com base na aplicação COIMA (*Chelas Open Internet Movie Application*), a ser desenvolvida ao longo do semestre. A aplicação fornece acesso, através de uma interface web (hipermédia), às funcionalidades disponibilizadas pelo sítio TMDb[1] (https://www.themoviedb.org), fazendo para esse efeito uso da Web API correspondente (https://developers.themoviedb.org/3/getting-started), bem como funcionalidades autónomas.

O desenvolvimento será realizado de forma incremental, envolvendo necessariamente vários ciclos de *refactoring* do código produzido e, por isso, é fundamental que faça uso de técnicas *Object-Oriented* e de boas práticas de programação em geral, de modo a reduzir o esforço associado a cada ciclo.

A especificação das funcionalidades (i.e. requisitos funcionais) e requisitos não funcionais é também apresentada de forma faseada, sendo para o efeito produzidos três enunciados. Em cada enunciado é definida a data limite de entrega da solução, um detalhe não negociável. A entrega é realizada através da criação da tag **ParteX** no repositório GitHub do grupo, onde X corresponde ao número da parte a entregar.

Requisitos Funcionais

Desenvolver uma aplicação Web que suporta pesquisa e a apresentação de informação detalhada sobre filmes. A fonte de dados é fornecida pela API RESTful: https://api.themoviedb.org.

A aplicação disponibiliza dois *endpoints* cujas *paths* são as seguintes:

URI *endpoint*	Descrição do resultado	Exemplo
/search?name={query}	Página HTML com a listagem paginada dos filmes que correspondem à pesquisa de {query}.	/search?name=Ninja
/movies/{movie-id}	Página HTML com os detalhes de um filme, que incluem o nome, a imagem correspondente ao poster do filme e a informação do realizador e dos atores participantes no filme, com o nome do ator e o nome da personagem interpretada. O nome do actor é um link para a página com os seus detalhes.	/movies/36326
/actors/{actor-id}	Página HTML com os detalhes de ator, com a sua biografia e a lista de filmes em que participou. O nome de cada filme deve constituir uma hiperligação para a página de detalhe do filme.	/actors/680

[1] The Movie Database

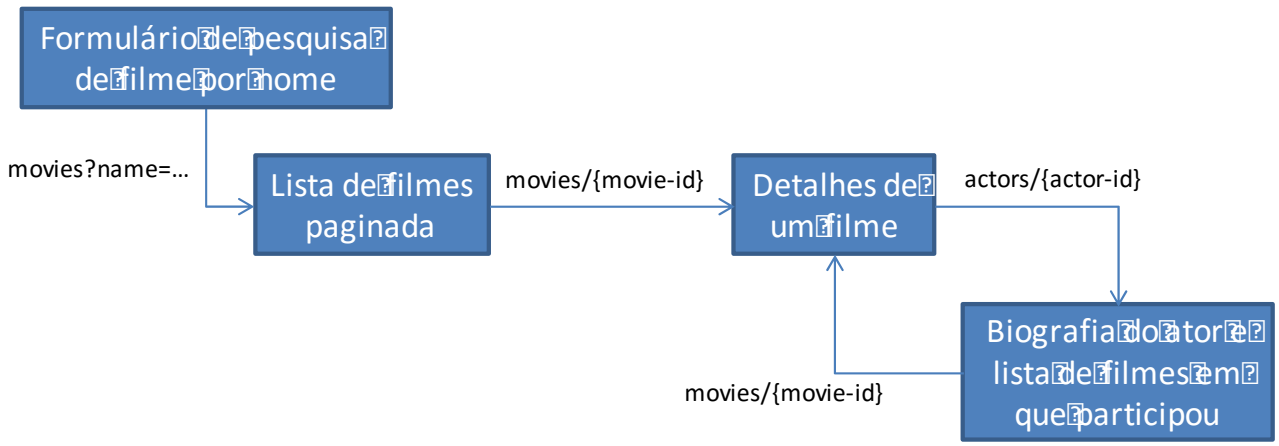
Instituto Superior de Engenharia de Lisboa

Licenciatura em Engenharia Informática e de Computadores

Programação na Internet

Semestre de Inverno de 2017/2018 –1ª Parte do Trabalho prático

As páginas e ligações da aplicação Web devem obedecer ao esquema de navegação seguinte:



Cada filme da listagem deve ter um *link* para os seus detalhes. Cada página de detalhes de um filme deve terá lista de todos os seus intervenientes. No caso dos atores deve ter o nome da personagem e do ator. O nome do ator constitui uma hiperligação para a página do ator, onde consta uma lista dos filmes em que este participou. Cada filme do ator deve ter um *link* para os seus detalhes.

As informações da fonte de dados são obtidas dos seguintes *endpoints* da API RESTful https://api.themoviedb.org:

1. Pesquisa: *https://developers.themoviedb.org/3/search/search-movies*
2. Detalhes de um filme: https://developers.themoviedb.org/3/movies
3. Lista de personagens de um filme: https://developers.themoviedb.org/3/movies/get-movie-credits
4. Participações de um actor: https://developers.themoviedb.org/3/people/get-person-movie-credits
5. Detalhes de uma pessoa: https://developers.themoviedb.org/3/people/get-person-details

Exemplos:

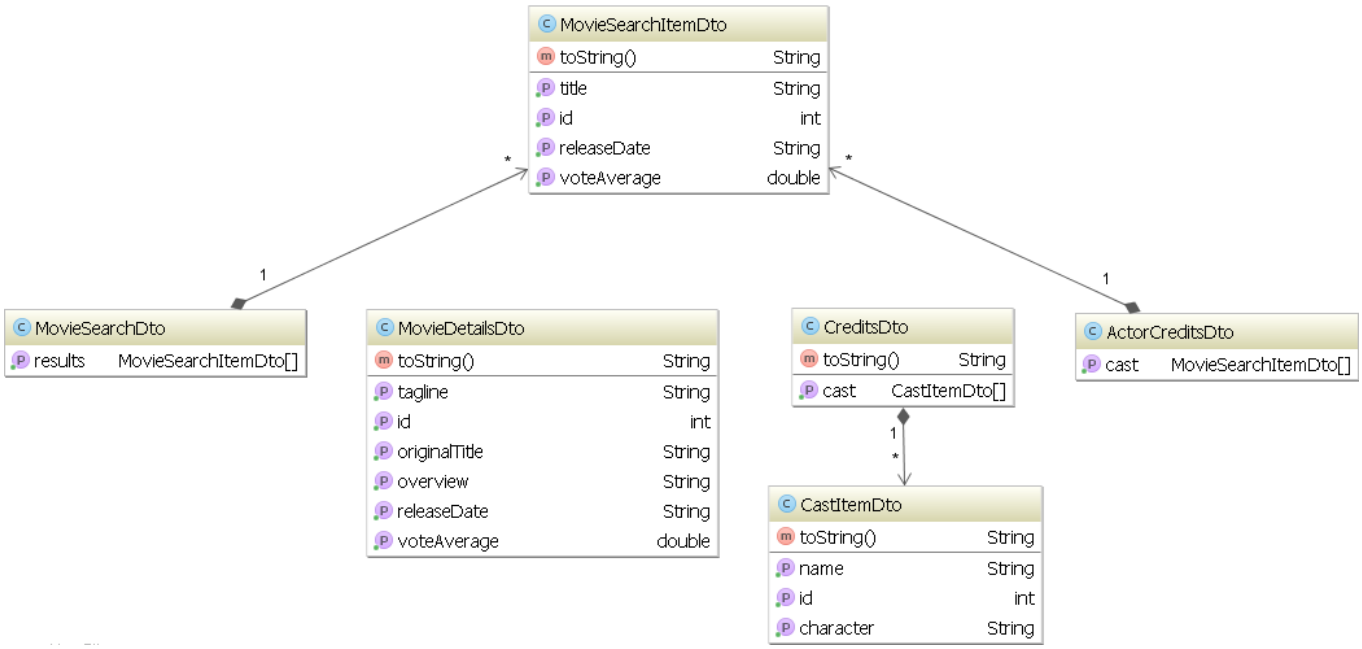
1. *https://api.themoviedb.org/3/search/movie?api_key=*****&name=war+games*
2. *https://api.themoviedb.org/3/movie/860?api_key=*****
3. *https://api.themoviedb.org/3/movie/860/credits?api_key=*****
4. *https://api.themoviedb.org/3/person/4756/movie_credits?api_key=*****
5. *https://api.themoviedb.org/3/person/8891?api_key=*****

NOTA:

A API RESTful: https://api.themoviedb.org tem limite de pedidos. Para que não ultrapasse esse limite poderá controlar o número de pedidos através de bibliotecas auxiliares node.js específica para esse efeito. Por exemplo: limiter, request-rate-limiter, entre outras.

Os resultados dos *endpoints* 2, 3, 4 e 5 deverão ainda ser armazenados em cache (apenas memória) de modo a que pedidos subsequentes ao mesmo URI não desencadeiem novos pedidos à API RESTful. Realize testes unitários que verifiquem o funcionamento da cache e que não é realizado mais que um pedido HTTP para cada informação.

Todos os resultados da API RESTFul podem ser desserializados a partir de JSON para objetos com propriedades semelhantes às exemplificadas no diagrama de tipos seguinte:



Requisitos Não Funcionais

A aplicação deve ser desenvolvida com a tecnologia Node,js. Para o atendimento de pedidos HTTP deve ser usado o módulo http. Pare realização de pedidos, pode ser usado o módulo http ou em alternativa o módulo request.

Qualquer dos módulos base do Node.js podem ser usados. Além destes, nesta 1ª parte do trabalho, apenas podem ser usados os seguintes módulos:

- request – Realização de pedidos HTTP
- handlebars – *View Engine*
- debug – Mensagens de debug
- nodeunit – Testes unitários

Incluir nas representações HTML regras de estilo utilizando a *framework* Bootstrap.

A solução entregue deve incluir **todos os testes unitários necessários** para validar o correto funcionamento do **modelo de domínio e acesso a dados**.