Aplicação web "TheyCinema"

Alejandro Carvalho, André Figueira e David Aguiar
Universidade da Madeira, Portugal

alejandr0123987@hotmail.com, igorffigueira@gmail.com,
daveaguiar26@gmail.com

1. Introdução

O presente trabalho prático insere-se no âmbito da unidade curricular Aplicações Centradas em Redes do Mestrado de Engenharia Informática da Universidade da Madeira.

Atendendo ao enunciado, foi-nos proposto elaborar um *website* de um tema à escolha do grupo. Desta forma optou-se por desenvolver uma aplicação *web* baseada na gestão de filmes por parte do administrador e vendas de bilhetes de cinema, em que um utilizador (visitante ou registado) poderá navegar pela aplicação com o intuito de obter informações básicas acerca dos mesmos, como por exemplo o género, classificação e a descrição dos filmes, como também proceder à compra de bilhetes para visualização dos filmes.

No presente documento expõe-se uma breve descrição da aplicação *web,* evidenciando os principais pontos de concretização da mesma, bem como o seu funcionamento em termos gerais.

Primeiramente aborda-se a questão a solucionar, mencionando a ideia do projeto, tendo em conta as exigências e características expostas pelo docente.

Na fase seguinte pretende-se explicar através da revisão da literatura e do plano de ação as várias etapas de desenvolvimento e projeção das *views* da aplicação *web*, baseando-se nas pesquisas efetuadas sobre os *websites* análogos.

Posteriormente refere-se a solução proposta, elucidando os diagramas da base de dados e dos casos de utilização para os diferentes tipos de utilizador, de forma a facilitar a compreensão e funcionamento geral da aplicação, assim como as características mais pertinentes.

Seguidamente será descrito os passos a seguir para utilizar a aplicação *web* e para finalizar será elaborada uma breve reflexão sobre projeto desenvolvido, expondo as suas vantagens e limitações.

2. Problema

Atualmente a maioria das empresas procuram dar a conhecer os seus projetos através das aplicações *web*, o que se traduz num excelente meio para chegar aos clientes/utilizadores. Como já mencionado o presente projeto foi proposto pelo docente e consiste no desenvolvimento de uma aplicação *web* de um tema à escolha, utilizando a *framework Laravel* na *versão 5.3,* de modo a facilitar tarefas comuns exercitadas na maioria dos projetos *web* como autenticação, roteamento, sessões, entre outras, através da arquitetura *MVC* (*Model – View – Controller*). Portanto, para a criação deste *website* optou-se por desenvolver o mesmo sobre a temática da gestão de filmes e venda dos mesmos ao público. Assim sendo, pretende-se expor secções que permita visualizar os filmes em exibição e em estreia; comprar bilhetes, obter informações acerca da descrição do filme (poster do filme, sinopse e género e outros detalhes de um filme.

A execução deste trabalho prático teve que atender a vários requisitos. Por exemplo, possuir pelo menos três tipos de utilizadores diferentes; seis *views* diferentes; cinco tabelas distintas, bem como apresentar a funcionalidade de *API* (*Application Programming Interface*) e ser desenvolvido através da *framework Laravel*.

3. Revisão da literatura e plano de ação

Inicialmente procedeu-se uma breve discussão entre o grupo a fim de delinear quais são os pontos principais a considerar na aplicação *web*, tais como as funcionalidades essenciais, os aspetos de navegação e os *templates*. Posto isto, elaborou-se uma breve pesquisa sobre aplicações *web* semelhantes à temática selecionada, como por exemplo, *http://www.adorocinema.com/*. Através desta investigação obteve-se uma perspetiva geral acerca das principais ferramentas utilizadas e adquiriu-se uma melhor perceção da constituição da aplicação.

A elaboração de um projeto na *framework Laravel* implica um prévio estudo ao funcionamento desta. Assim, nesta fase recorreu-se à documentação fornecida pelo docente e pela página *web Laracasts* visando adquisição do maior número de conhecimentos sobre esta *framework*. Obtidos os dados essenciais, começou-se a implementação da autenticação já pré-definida pela *framework Laravel*, produzindo automaticamente uma tabela "*User*" na base de dados, os *routes, os controllers, model* e as *views* próprias para a autenticação. Logo, somente teve-se que adaptar as tabelas e as *views,*

referentes à autenticação, às necessidades do nosso *website*, como por exemplo, a verificação de registo de conta através do envio de um *email* de verificação e recuperação da *password*.

Posteriormente fomos progredindo na criação da nossa aplicação implementando uma *API* externa que nos fornece todos os filmes existentes. Cada um desses filmes está identificado por um identificador da *IMDb* (*Internet Movie Database*), bem como todos os outros detalhes dos mesmos.

Para a obtenção desses dados utilizamos a tecnologia *Axios* que consiste num cliente de *HTTP* (*Hypertext Transfer Protocol*) *promisse-based* para *JavaScript* que pode ser utilizado em aplicações *front-end* e em *node.js backend*. Este permite enviar de forma fácil e assíncrona pedidos *HTTP* para pontos *REST* (*Representational State Transfer*) e executar operações *CRUD* (*Create*, *Read*, *Update* and *Delete*).

Em seguida efetuamos uma alteração na tabela de '*U*ser' e acrescentamos um campo que identificasse o tipo de utilizador de forma a que a aquisição dos dados da *API* externa estivesse limitada ao administrador, favorecendo assim o "encapsulamento" de tal funcionalidade. Posteriormente foi criada uma vista de administrador e estabelecida a referida rota para essa *view* com a verificação de autenticação por parte do administrador para a obtenção desses dados da *API*.

Com o resultar da obtenção dos dados da *API*, fomos implementando as tabelas da base de dados conforme fomos sentindo a necessidade das mesmas. Portanto, com estas foram criados os modelos onde especificamos as relações entre as tabelas criadas, de maneira a definir as dependências de tabelas (1:1, 1: n ou m: n) para posteriormente atribuir as chaves estrangeiras às migrações das tabelas. Esta área da definição das chaves estrangeiras causou alguns problemas por não estarmos familiarizados com a atribuição de chaves estrangeiras na *framework Laravel*, porém tal dificuldade foi ultrapassada com o auxílio de alguma pesquisa em páginas como o *Stack Overflow* e na documentação do *Laracast.*

Com a finalização da implementação do *back-end* do nosso *web-site*, passamos à inserção de *CSS* (*Cascading Style Sheets*) nas *Views* implementadas de forma a tornar o nosso *web-site* mais apelativo e interessante e aplicamos verificações de autenticação para acessos às funcionalidades de cada tipo de utilizador, como ainda verificações dos campos de dados inseridos por parte do utilizador.

4. Solução proposta

Nesta secção será abordado de forma detalhada a descrição dos vários conteúdos presentes no *website*, nomeadamente a exposição geral da solução proposta, dos *controllers*, das tabelas, diagramas da base de dados e dos casos de utilização para diferentes tipos de utilizador e os aspetos particulares relevantes.

4.1. Descrição geral da solução proposta

Para a execução deste trabalho prático utilizou-se vários *softwares* e ferramentas, tais como o *PHP* (*Hypertext Preprocessor*), o *JavaScript*, o *CSS*, o *HTML* (*HyperText Markup Language*), *Jquery*, *Apache*, *Laravel* e *MySQL*.

Esta aplicação *web* é constituída principalmente por uma base de dados com nove tabelas, uma *API* externa e interna, dezasseis *views e* doze *controllers*, cujas características dos principais encontram-se descritas, nas seguintes secções.

A aplicação *"TheyCinema"* está destinada a três tipos de utilizadores: os visitantes que só têm acesso à exposição e descrição dos filmes; o registado, que pode comprar os bilhetes para os filmes disponíveis e selecionar os lugares (frente, centro e atrás) da sala da película desejada e para finalizar tem-se o administrador, que tem a capacidade de escolher, adicionar ou retirar os filmes do cartaz, bem como, editar a sala para a exibição de cada filme.

Procedendo à exploração da aplicação *web*, esta possui no topo da *template* uma barra de navegação, como as principais funcionalidades da aplicação e o registo e *login* do usuário. Esta barra de navegação encontra-se ilustrada na Figura 1 do anexo A.

Como podemos ver na ilustração da figura 1, a nossa aplicação *web* possui a possibilidade de um utilizador realizar a sua autenticação através de um registo prévio e posteriormente o seu *login*, e outras quatro secções da nossa aplicação que poderão ser acedidas através do "*Home*" em que a imagem no canto superior esquerdo a azul também direciona para a nossa página principal, os nossos "Parceiros", a "Localização" dos nossos espaços para que desfrute de um bom filme e informações auxiliares sobre a nossa "Equipa", estando todas estas quatro opções presentes num menu horizontal no cabeçalho da nossa aplicação *web*.

Além da barra de navegação, a página principal da aplicação contém um *slider*, que destaca alguns filmes em exibição, captando a atenção dos utilizadores. Abaixo do *slider*, são apresentados os vários filmes em cartaz,

através dos quais pode-se aceder as informações acerca desses filmes, bem como podemos proceder à compra do bilhete.

Para a obtenção dos filmes e dos detalhes dos mesmos foi utilizada uma *API* externa para consumo dos dados de cada filme, e uma base de dados que fornecia à *API* os filmes que o cinema já tinha adquirido para fornecer os detalhes dos mesmos.

Concluindo os conteúdos expostos na *homepage*, pode-se encontrar uma lista com os títulos dos filmes em cartaz que podem ser acedidos em qualquer *view*. A *homepage* encontra-se representada na Figura 2, no anexo B.

Na *view* da informação do filme são apresentados todos os dados relativos ao mesmo, tais como título, ano, género, elenco, *rating* e descrição. A Figura 3 que consta no anexo C, tem-se representada a *view* acerca da informação de um filme. Além de aceder às informações revelantes em relação à pelicula, nesta página pode-se comprar o *ticket* para visualizar o filme selecionado, desde que o utilizador esteja autentificado.

Se o cliente pretender comprar um bilhete obterá acesso à localização dos lugares disponíveis (frente, centro e atrás) da sala para a visualização do filme, podendo assim selecionar o lugar pretendido, caso este não esteja ocupado. Através da Figura 4, que se encontra no anexo D pode-se observar a página da aplicação *web* com a funcionalidade de seleção dos lugares para a visualização do filme.

Após a escolha do lugar, o usuário será direcionado para um formulário, em que devidamente preenchido, permitirá efetuar o pagamento do bilhete. A Figura 5, que se encontra no anexo E ilustra o formulário para proceder ao pagamento do *ticket*.

Uma vez completado o formulário pelo cliente, é enviado um *email* com os dados para confirmação do pagamento.

No que se relaciona ao administrador, este pode executar várias funções que apenas estão disponíveis para o próprio. Assim sendo, na sua *view* existe um motor de busca de filmes, através do qual é possível escrever o nome do filme pretendido. Pelo que, realizada uma pesquisa de um dado filme, são exibidos os títulos e os *posters* dos filmes relacionados com o que escreveu no motor de busca. Por exemplo, escrevendo "*Harry*" no motor de busca, temos acesso a todos os filmes que possuam "*Harry*" como título. Nas Figura 6 e Figura 7 presentes nos anexos F e G, respetivamente, encontram-se representados todos as funcionalidades que o administrador possui e os filmes relacionados com o exemplo da pesquisa da palavra "*Harry*".

Após pesquisar o filme, o administrador tem a opção de adicionar o filme à lista do cartaz, sendo que após a adição este é automaticamente adicionado à lista de cartaz e ao *slider* da *homepage*. Da mesma forma que o administrador pode adicionar os filmes, também pode removê-los do cartaz.

O administrador pode ainda interligar o filme do cartaz à sala que pretende que este seja exibido, sendo que este cinema possui somente quatro salas para a exibição dos filmes. Na Figura 6 presente no anexo F, encontra-se representada a *view* com todas as funcionalidades que um administrador pode realizar.

4.1.1. Base de dados

Como já mencionado, a aplicação *"TheyCinema"* suporta três tipos de utilizadores, sendo que dois destes são referenciados na nossa base de dados na tabela *user*, através da variável "*usertype*".

Um utilizador que efetue um registo na nossa aplicação web será automaticamente considerado um "*client*", um administrador um "*admin*" e o outro tipo de *user* não terá uma atribuição de categoria na nossa base de dados, pois trata-se de um simples visitante da página.

É de salientar que o administrador não poderá fazer um registo pelo site, visto que este já é criado automaticamente quando é efetuada a migração, onde junto desta é feito um "*seed*" da base de dados com um administrador.

Na Figura 8 do anexo H, encontra-se ilustrado o modelo relacional desenvolvido através da ferramenta desenhador do *phpMyAdmin*.

4.1.2. *API* externa

A *API* utilizada neste projeto é intitulada por *omdb*. Esta oferece duas maneiras de obtenção de informação. A primeira é por uma pesquisa geral, onde colocamos o nome do filme que pretendemos e esta irá devolver uma lista de dez filmes que se assemelham-se com o nome que colocamos, dando alguns dados de cada filme, como o nome, o poster e o mais importante o *ID* do filme, entre outros dados. Com este *ID* é possível utilizar o outro método de pesquisa oferecida pela *API*. Nesta pesquisa, a *API* oferece muito mais informação acerca filme, tais como o nome, *ID*, o ano, género do filme, ou mesmo a idade permitida. Toda a informação está em formato de *json*.

4.1.3. *API* interna

Foi criada uma *API* interna para dar acesso aos utilizadores de algumas informações, como o nome dos filmes e os lugares disponíveis. Para ter acesso à *API* é necessário de ter uma chave de autentificação, estas chaves estão guardados na base de dados, dando assim a possibilidade de gerar mais chaves e disponibiliza-las.

4.1.4. *Controllers*

Na elaboração deste projeto foram definidos vários controladores para o tratamento das funcionalidades apresentadas nas *views*. Para tal, é utilizado um ficheiro de rotas, que redireciona para um controlador dependendo da rota em que se encontra, especificando a função a desempenhar. Em adição, podem ainda manipular os dados da base de dados e retornar vistas.

Os *controllers* foram criados consoante a evolução e necessidade dos mesmos para tratar das funcionalidades da aplicação.

Em seguida, encontram-se a descrição dos controladores considerados mais importantes para o funcionamento da aplicação.

4.1.4.1. *AdminController*

Neste controlador estão as funções do administrador, que contém as funções das salas, dos filmes em exibição, a pesquisa por filmes na *API*, para colocar na base de dados/ associar a salas, adicionar/remover filmes e "limpar" as salas de utilizadores.

A função *index* retorna a *view* de administrador, onde serão apresentados os filmes já existentes na base de dados e os filmes que estão associados a uma dada sala do cinema.

Em *savemovies* são recebidos os dados de um *movie* que foi procurado e escolhido para adicionar à base de dados, e através do *request* são passados os campos (*ID*, título, *ID* da sala e o *URL* do filme) é de destacar que o *ID* da sala não é passado pela *API* externa, mas sim "predefinido para a sala 1" para evitar problemas na base de dados devido a chaves estrangeiras. Estes campos são associados aos atributos da tabela *movie* da base de dados e submetidos.

A função *deletemovies* remove um filme através de uma *query* no formato *Laravel* (*Eloquent*) através do *ID* de *movie* passado pelo *request*. Este controlador tem mais duas funções que visam conectar um filme a uma determinada sala, e para tal é recebido na função um *request* que trás com ele o *ID* do filme e a sala escolhida para colocar o filme em questão em exibição e com esses dados é feito uma *query* de *update* à tabela *movies* e atualizado o

ID da sala, como já tinha referido anteriormente que é colocado como *default* a sala com o *ID* 1.

A outra função do controlador, consiste em colocar todos os lugares da sala de cinema livres/desocupados. Portanto, executa um *update* à tabela *positions* de uma sala em questão e coloca todos os atributos (a1,a2,a3....,c3) a 0 que significa "livre".

4.1.4.2. *BuyticketController*

Este controlador tem duas funções e um *middleware*, em que o *middleware* "obriga" que o utilizador esteja autenticado para ser capaz de comprar um bilhete.

A função *buy* recebe o *ID* do filme que o utilizador pretende comprar e faz uma busca da sala em que esse filme está em exibição, através de uma *query* à tabela *movie* através do *ID* do filme e com o *ID* da sala retorna as posições da sala em questão através da tabela *positions* e retorna os resultados num *array* e ainda retorna a *view buyticket* com esse *array* das posições.

A última função chamada *served* guarda o *ID* da sala e *ID* da posição (cadeira na sala de cinema) pretendida que é recebida através do r*equest* e depois dependendo do lugar escolhido faz uma *update* à tabela *positions* através do *ID* da sala e coloca a posição a 1, ou seja, como ocupada/reservada.

4.1.4.3. *JsonController*

Este controlador possui duas funções, pois pode verificar se a chave de autentificação está na base de dados e manda para exterior alguns dados, como o *ID*, o nome dos filmes e a sala onde se encontram, os lugares e as salas disponíveis pelo o cinema. Estes dados estão todos agrupados num *array* para facilitar a sua obtenção.

4.1.4.4. *SendEmailController*

Este controlador tem apenas uma função, que é a verificação e envio de *email* para o cliente.

Na primeira parte da função são validados os campos de pagamento, o número de cartão onde é apenas verificado se foram inseridos doze dígitos, a data de validade se tem quatro dígitos e o código de segurança com quatro dígitos, sendo todos os campos de preenchimento obrigatório.

Em seguida é colocado num *array* o número do cartão que vem do *request* e enviado o *email* com os campos *from* que é um *email* criado pelo grupo para envio de *emails*, o campo *to* que é o email destinatário/cliente autenticado e o

campo *assunto* a indicar a confirmação de compra. Por fim faz o redireccionamento para a página inicial.

4.2. *Seeders*

Foram definidos quatro *seeders* para inserir dados na base de dados, no momento inicial de arranque da base de dados de valores que consideramos serem necessários para o correto funcionamento do sistema. Para isso tivemos de ir ao *forlder App/database/seeds* e criar quatro novos ficheiros *PHP* e inserir os dados da tabela desejada com valores correspondente aos atributos da tabela desejada e com o correto tipo de dados esperado, e depois no *DatabaseSeeder.php* chamar esses quatro ficheiros.

Em seguida é necessário correr um comando *composer dump-autoload* na *bash*, no diretório do projeto.

Posteriormente quando executado o comando para migrar as migrações para a base de dados temos de utilizar o comando *Php artisan migrate –seed* em vez do *migrate*. Para o caso de ser necessário fazer um *refresh* utiliza-se o comando *Php artisan migrate:refresh –seed*.

Juntamente com as migrações são também inseridos valores em cinco tabelas da base de dados e na tabela *user* é inserido um utilizador do tipo administrador, sendo que esta tabela apresenta um atributo *usetype* que nesta inserção terá o valor *admin*. Na tabela "*cinema*" é inserido um cinema, noutra são criadas as "cadeiras"/posições das salas de cinema e colocadas como desocupadas, ou seja, disponíveis/livres. Na tabela *key* são introduzidas chaves para a *API* interna, na e na última tabela *salas* são inseridas 4 salas (norte, sul, este e oeste).

4.3. Diagramas auxiliares

Para esta secção foi desenvolvido um diagrama de casos de utilização, onde expusemos todas as funcionalidades/capacidades possíveis de realizar e em volta os nossos tipos de utilizadores, sendo eles: o visitante; o registado e o administrador. Estes tipos de utilizadores depois foram conectados às respetivas funcionalidades a que deverão ter autorização. Este diagrama encontra-se no anexo I, como sendo a Figura 9.

5. Utilização da aplicação

Para aceder ao nosso site como administrador terá de usar as seguintes credenciais:

- ***Email***: admin@admin.admin
- ***Password***: admin

Para aceder ao site como visitante não é necessário efetuar o *login* para navegar pelos diversos separados bem como "Parceiros", "Equipa" ou "Localização" ou até mesmo para visualizar todos os filmes em exibição, ver figuras Figura 10, Figura 11 e Figura 12 dos anexos J, K e L, respetivamente.

Porém para poder efetuar uma compra terá de efetuar um *login* ou registo. Assim, posteriormente é enviado um *email* para confirmar o registo e a partir dai pode-se efetuar todas as compras que desejar.

5.1. Tipos de Utilizadores

Nesta subsecção será descritas algumas das caracteristicas que osdiferentes tipos de utilizadores podem usufrir na aplicação *"TheyCinema"*.

5.1.1. Utilizador visitante

Um utilizador que não se encontre registado e ou com o *login* efetuado, poderá navegar pelo nosso site apenas para visualizar os filmes que se encontram em exibição e as suas caraterísticas,

Para efetuar um registo/*login* poderá dirigir-se ao canto superior direito tal como consta na como consta na Figura 2 do anexo B.

5.1.2. Utilizador registado

Um utilizador registado está apto a executar todas as funcionalidades de um utilizador visitante, contudo é-lhe conferida a possibilidade de poder comprar um bilhete para uma sessão de cinema.

Um utilizador registado e com o *email* confirmado, poderá efetuar o *login* no canto superior direito da aplicação. De seguida este é redirecionado para outra página onde lhe será pedido o *email* e a *password* de modo a efetuar a sua autenticação.

Caso a sua autenticação seja bem-sucedida, será redirecionado para a página principal já com a sessão iniciada.

Após um cliente selecionar um filme que se encontra no cartaz, poderá ver os detalhes do mesmo, caso pretenda ver outro filme poderá voltar atrás e escolher outro. Se estiver interessado no filme que está a ver, poderá carregar no botão comprar, como podemos ver na Figura 3 do anexo C. Depois de carregar no botão comprar irá ser apresentada uma nova vista para efetuar o pagamento, onde será pedido o número do cartão de pagamento, data de validade e o *CCV* (código) do mesmo, como consta na Figura 5 do anexo E

Ao carregar no botão "pagar" e os campos tenham sido devidamente preenchidos será direcionado para a página principal e será enviado um *email* para o cliente que se encontra autenticado com a confirmação do pagamento do bilhete e com um código para levar o bilhete no balcão do cinema, como consta na Figura 13 no anexo M.

5.1.3. Utilizador administrador

Um administrador poderá navegar pela aplicação como um utilizador registado, porém para poder pesquisar por filmes da *API* externa ou qualquer outra função terá de efetuar *login* com os dados de administrador. Então depois de autenticado poderá pesquisar por filmes que queira inserir na base de dados, através da opção de menu *Admin area*, como podemos ver na Figura 6 do anexo F.

Depois de adicionado à base de dados o administrador poderá atribuir um filme, que já tenha na base dados, a uma sala do cinema como podemos ver na Figura 14 do anexo N.

Um administrador poderá ainda remover um filme da base dados, carregando *Delete movie* no filme em questão, como podemos ver na Figura 15 do anexo O.

Após uma seção de cinema ou um horário de filme, o administrador terá de colocar as posições das salas novamente disponíveis/desocupadas, carregando "Limpar sala" na sala em questão, como consta na Figura 16 do anexo P.

6. Discussão

A aplicação desenvolvida apresenta algumas vantagens, a simplicidade da mesma possibilita o fornecimento de uma melhor experiência de utilização ao utilizador, como por exemplo fornecer informações acerca dos filmes em exibição, isto é, o ano de lançamento, o resumo do filme, o género e o *rating*. Além disso, este *website* permite a compra de bilhetes em apenas três passos (comprar, preencher campos de pagamento e pagar), o que é vantajoso aquando a estreia de um filme, permitindo evitar as filas de espera.

Em termos de desvantagens, esta aplicação não apresenta muita interatividade com os utilizadores, no que toca à opinião pessoal destes. Sente-se a lacuna de um espaço que permita aos clientes dar uma avaliação e comentário do filme.

A aplicação também não apresenta muitos meios de pagamento, o que pode tornar-se inconveniente para alguns clientes. Outro aspeto negativo a referir, é a ausência dos *trailers* dos filmes existentes neste *website*.

Sendo esta mais uma limitação do que uma desvantagem, a nossa aplicação *web* está limitada a apenas 1000 pesquisas diárias à *API* externa (obtenção de dados de filmes), o que para um site, que pode ter múltiplos utilizadores em simultâneo, é uma limitação.

7. Conclusão e trabalho futuro

Com a elaboração deste trabalho prático foi possível melhorar e enriquecer os conhecimentos relativos às plataformas e métodos de desenvolvimento de aplicações *web*.

Além disso, a realização deste projeto permitiu experienciar todas as etapas de criação uma *web* desde o problema inicial exposto: delinear soluções para a sua resolução; planificar a base de dados e desenvolver o repositório no *GitHub*. Em adição, também foi enriquecedor conhecer o potencial de uma ferramenta como a *framework Laravel* para o desenvolvimento de aplicações *web*.

Atendendo aos requisitos mínimos e extras propostos pelo docente, conclui-se que os mesmos foram alcançados com sucesso, embora com algumas dificuldades e falhas. É o caso da adaptação das *views* aos diferentes *zooms* (a aplicação ser responsiva aos diferentes tamanhos de janela).

Em relação a futuros trabalhos a desenvolver na aplicação *web "TheyCinema"* seria benéfico adicionar-se uma secção para a localização dos postos de cinema, recorrendo a uma *API* do *Google* de localização, com isto introduzir, além deste, outros espaços de cinema.

Igualmente seria favorável melhorar as informações dos filmes, como por exemplo, adicionar o *trailer* dos filmes em exibição, bem como um separador para os utilizadores expressarem as suas opiniões e sugestões acerca da sua experiência de filme.

Também poderia ser interessante adicionar uma opção de *menu* que apresentasse o cartaz dos filmes mais vistos (Top Semana).

Além disso, seria importante melhorar o método de pagamento dos bilhetes, criando uma estrutura mais compacta para o pagamento, e ainda adicionar outros artigos para compra, como outros bilhetes para a mesma/outra sessão de cinema, as pipocas, refrigerantes, entre outros.

Outro aspeto relevante para melhorar esta aplicação, passa por tornar mais dinâmico o *website* a nível do administrador, de modo a que este possa aceder a mais pontos de gestão da aplicação, como por exemplo, adicionar dinamicamente os preços dos bilhetes do cinema e adicionar mais lugares nas salas.

Por fim, de forma a tornar a aplicação mais interativa, poderia criar-se periodicamente passatempos para oferecer bilhetes ou brindes relacionados com os filmes, bem como criar menus de bilhetes com pipocas e refrigerantes.

8. Referências

"Autor anónimo A". ("sem ano"). CSS Tutorial. **Disponível em:** http://bit.ly/2geW91c **Data de acesso:** 20/12/2017

"Autor anónimo B". ("sem ano"). All questions. **Disponível em:** http://bit.ly/2ElCWoX **Data de acesso:** 20/12/2017

"Autor anónimo C". ("sem ano"). Laracasts. **Disponível em:** http://bit.ly/1skeCZu **Data de acesso:** 20/12/2017

"Autor anónimo D". ("sem ano"). Installation. **Disponível em:** http://bit.ly/2Czrqbc **Data de acesso:** 20/12/2017

"Autor anónimo E". 2015. Tutorials- Learn marketing & code. **Disponível em:** http://bit.ly/2DWxAzo **Data de acesso:** 20/12/2017

Quintal, F. 2017. Slides das aulas teórico-práticas 1-12.

9. Anexos

Nesta secção serão apresentados os anexos com as figuras com as caracteristicas da aplicação desenvolvida.

9.1. Anexo A

Figura 1: Barra de navegação da aplicação web "TheyCinema".

9.2. Anexo B

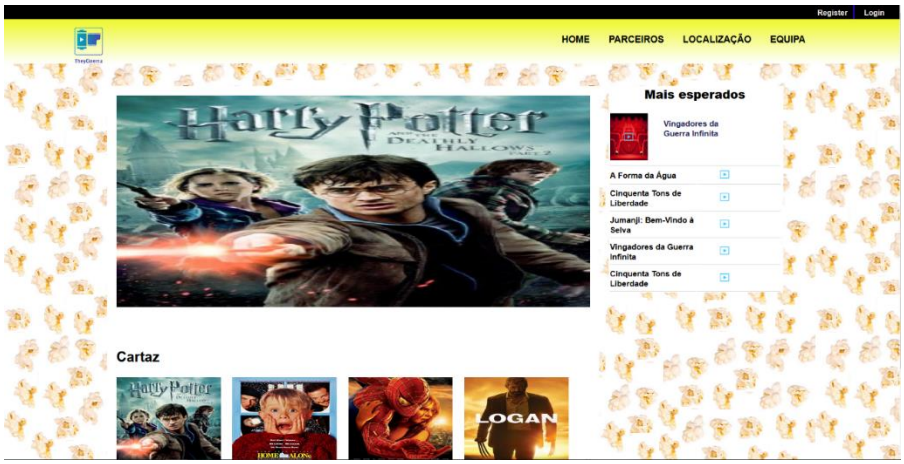

Figura 2: Página principal da aplicação web "TheyCinema".

9.3. Anexo C

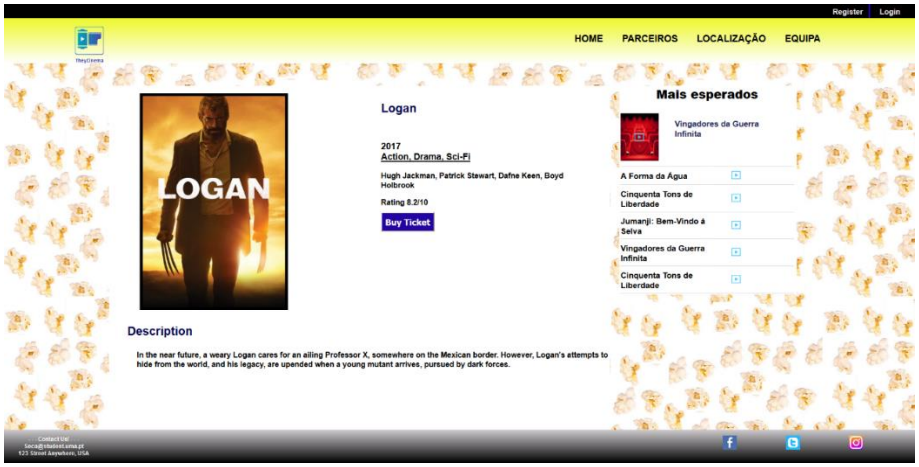

Figura 3: Descrição de um filme na aplicação web "TheyCinema".

9.4. Anexo D

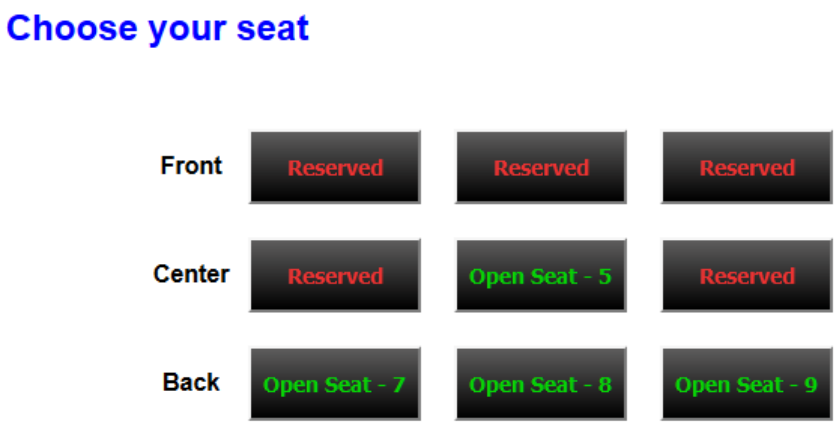

Figura 4: Escolha da cadeira para visualização de um filme numa sala.

9.5. Anexo E

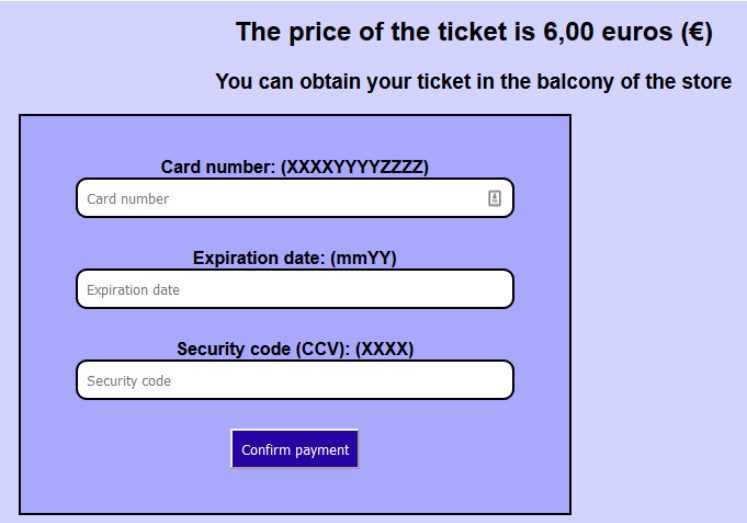

Figura 5: Formulário de pagamento do bilhete de cinema.

9.6. Anexo F

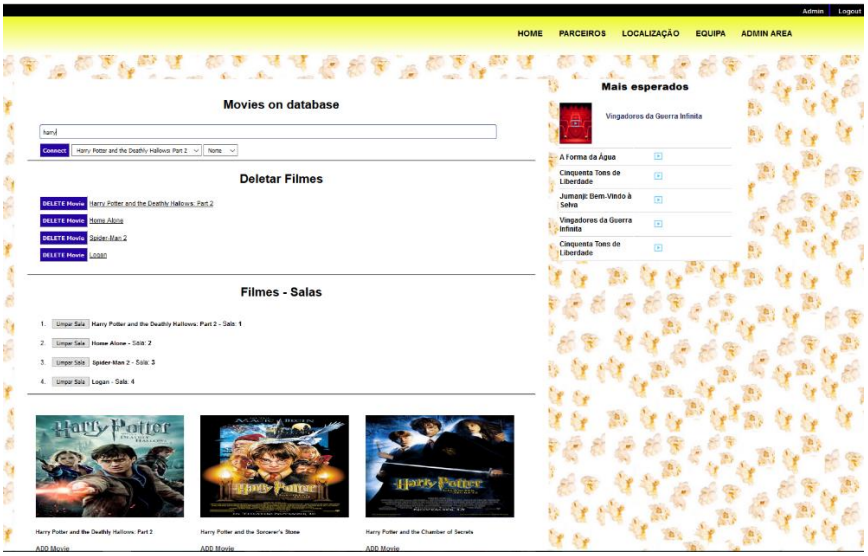

Figura 6: Funcionalidades de Admin e pesquisa na API externa por um filme com a palavra "Harry".

9.7. Anexo G

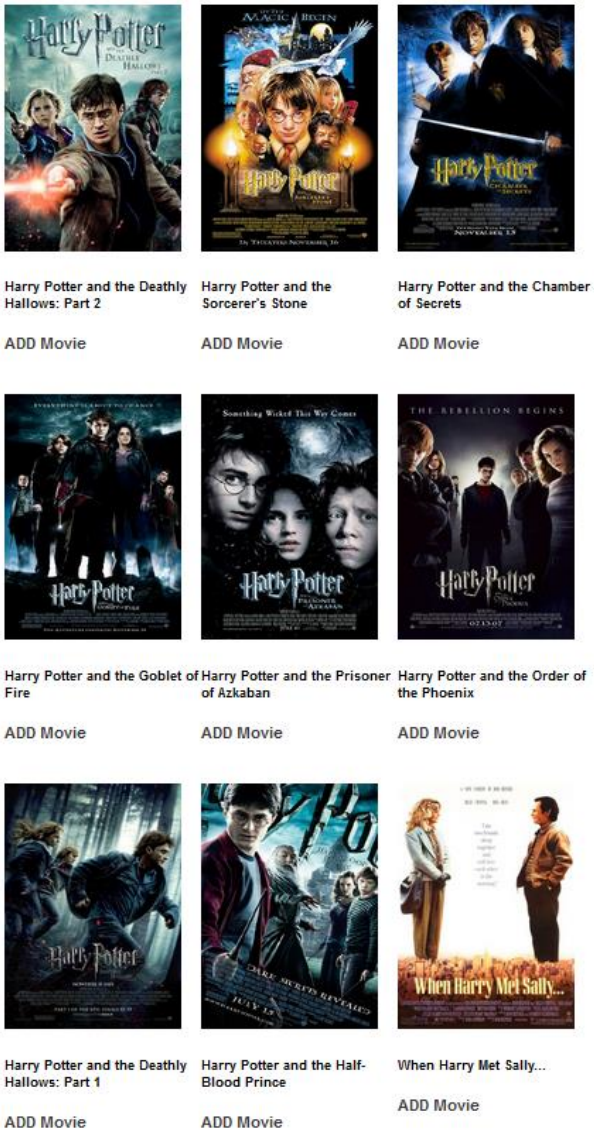

Figura 7: Resultados da pesquisa de filmes pela palavra "Harry" na API externa.

9.8. Anexo H

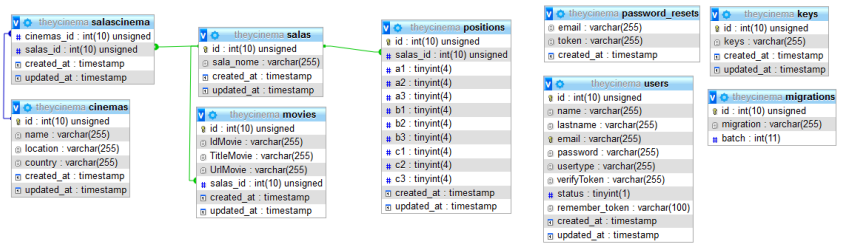

Figura 8: Modelo Relacional das tabelas da base de dados "TheyCinema"

9.9. Anexo I

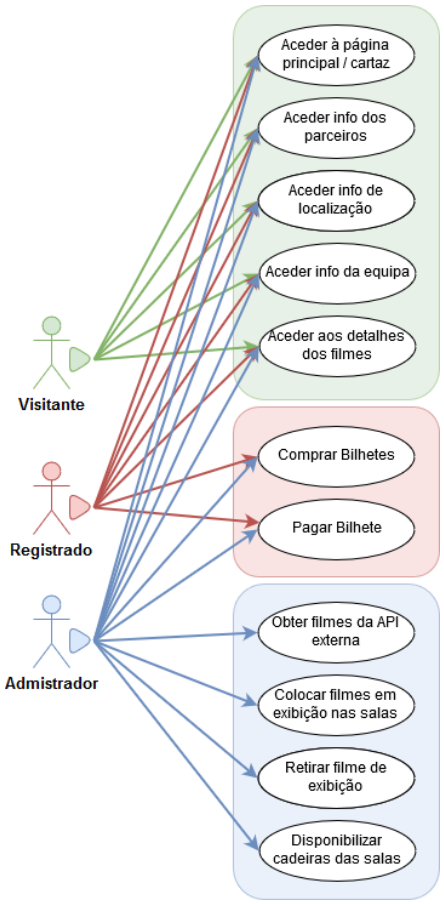

Figura 9: Diagrama de casos de utilização da aplicação "TheyCinema".

9.10. Anexo J

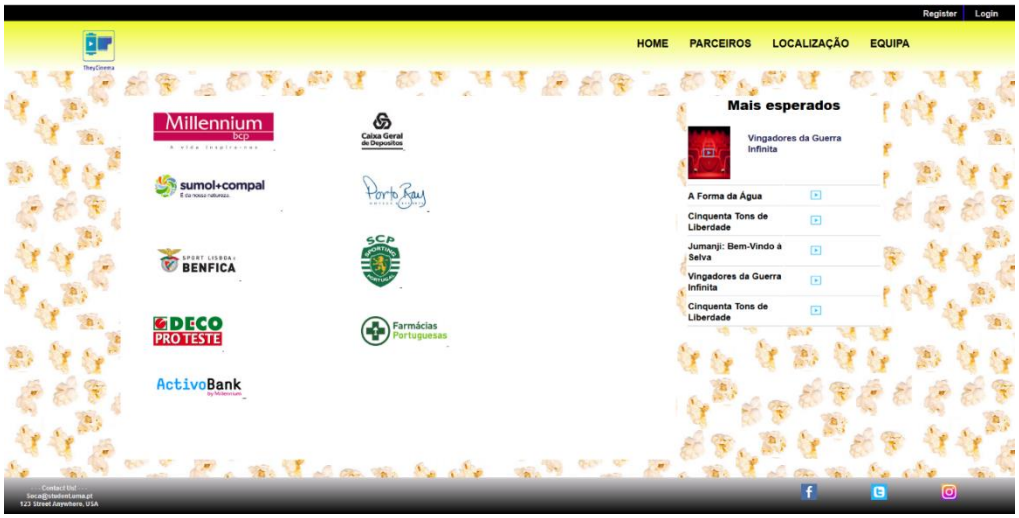

Figura 10: View de parceiros do grupo "TheyCinema".

9.11. Anexo K

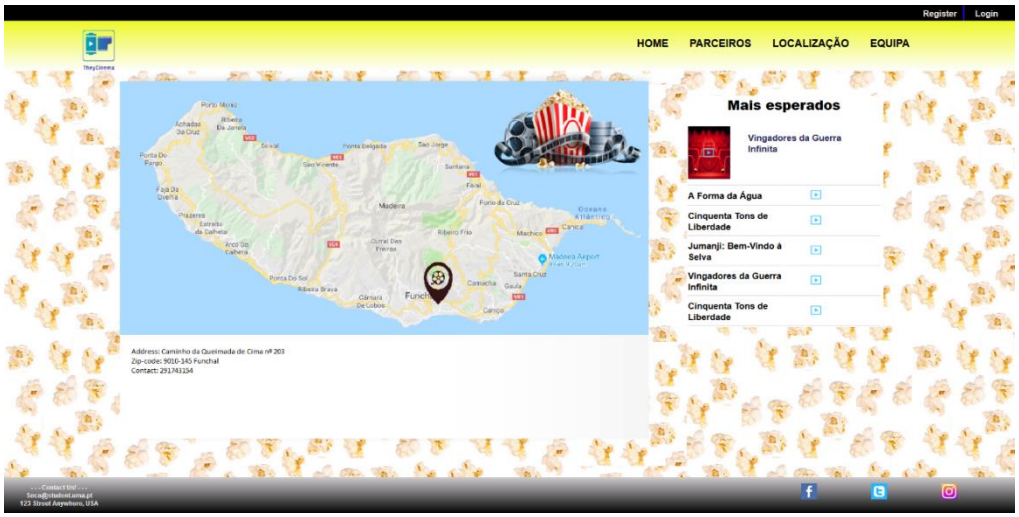

Figura 11: View de informações acerca da localização dos cinemas "TheyCinema".

9.12. Anexo L

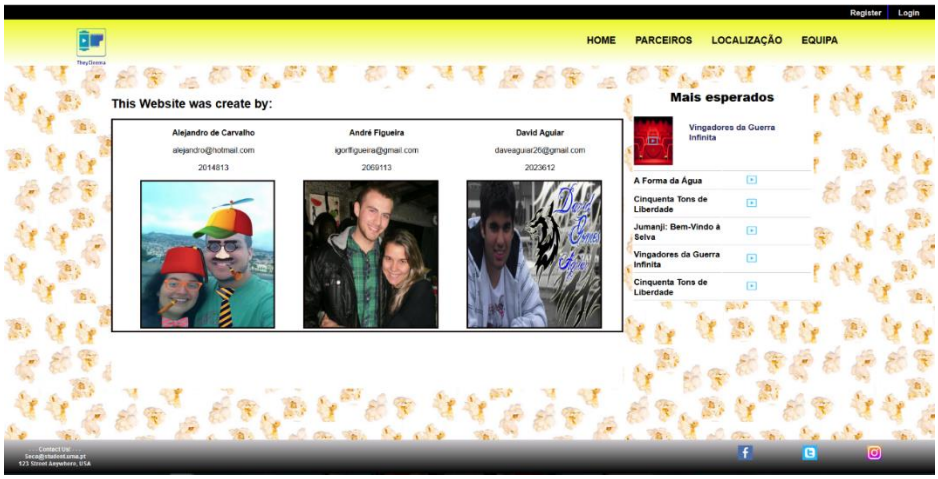

Figura 12: View de informações adicionais sobre a equipa "TheyCinema".

9.13. Anexo M

Figura 13: Confirmação de pagamento recebida no email do cliente autenticado.

9.14. Anexo N

Figura 14: Colocar um filme em exibição numa sala (associar filme a uma sala).

9.15. Anexo O

Figura 15: Retirar um filme de exibição.

9.16. Anexo P

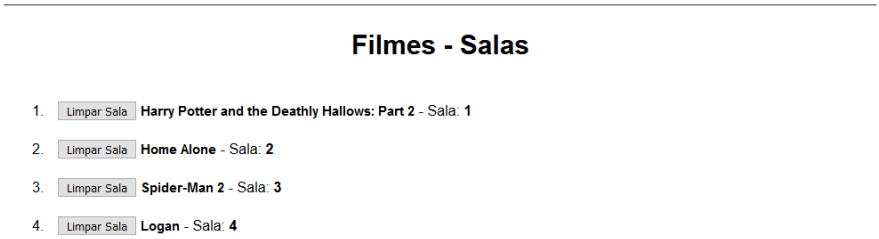

Figura 16: Colocação das cadeiras disponíveis/livres após uma sessão de cinema numa sala.