Relatório do Projeto

Monumentos da Diocese de Beja



Instituto Politécnico de Beja

Escola superior de Tecnologia e Gestão

Curso: Tecnologias Web e Dispositivos Móveis

Nome: Diogo Alexandre Beco Silva

Numero: 20307

Tema do projeto: Monumentos da Diocese de Beja

Ano Letivo: 2020/21

Local de apresentação: ESTIG

Orientadores de Estágio:

Carlos Sancho (professor de Programação para Dispositivos Móveis)

Luís Rosário (professor de Programação para a Web)

Relatório do Projeto de fim de curso apresentado na Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Beja

Índice

[Introdução 4](#_Toc81686321)

[Faseamento do Projeto 5](#_Toc81686322)

[Desenho da interface da aplicação 6](#_Toc81686323)

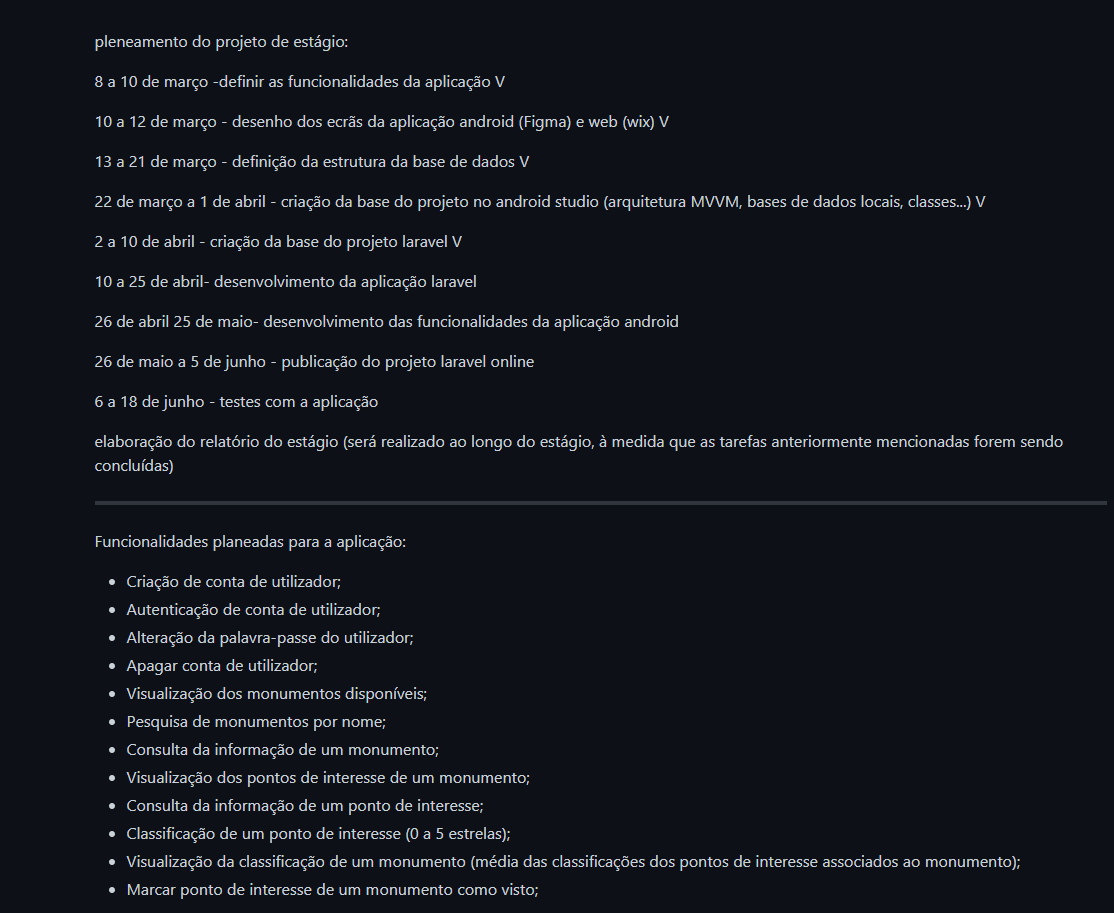
[Projeto Android Studio 12](#_Toc81686324)

## Introdução

Neste projeto de fim de curso, pretende-se que o aluno aprimore os seus conhecimentos com as tecnologias utilizadas no curso, relativamente ao “Android Studio” e ao “Laravel”. Será entregue ao Aluno o tema “Monumentos de Beja”, o qual pretende que seja criada uma aplicação android que possibilite que o utilizador seja guiado pelos vários pontos de interesse dos monumentos de Beja e forneça informação sobre os mesmos.

## Faseamento do Projeto

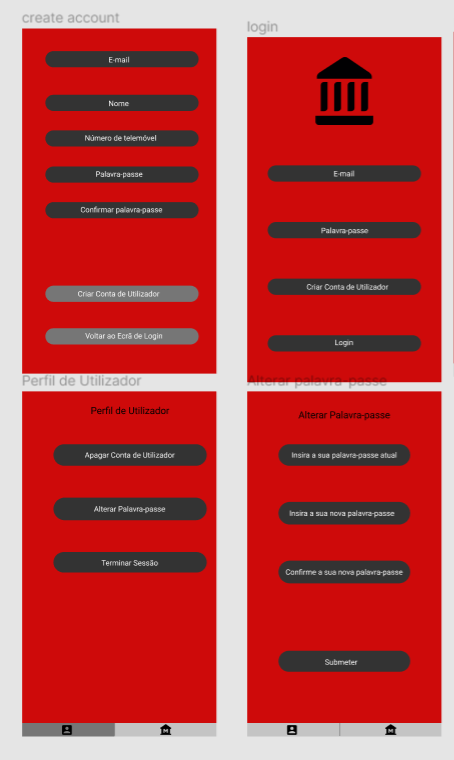
Para a realização do Projeto, foi pedido que se realizasse um faseamento, no qual o aluno deveria organizar as várias tarefas que seriam realizadas ao longo do semestre, assim como uma estimativa do tempo que cada uma dessas tarefas levaria e uma lista de funcionalidades pretendidas para a aplicação.



## Desenho da interface da aplicação

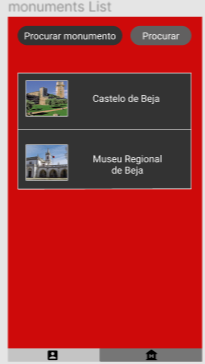
Para o desenho da interface da aplicação android foi utilizado o figma e para o desenho da aplicação laravel foram utilizados templates disponível no wix.

Começando pelo Figma, foram criados os vários ecrãs que se pretendiam usar na aplicação android. Foram criados ecrãs para a autenticação do utilizador (que foi decidido que seria uma função exclusiva da aplicação android). Esta função permite que o utilizador registe uma conta de utilizador e faça login. Permite ainda que, quando o utilizador acessa o seu perfil, o mesmo possa alterar a palavra-passe, terminar sessão, ou apagar a conta de utilizador.

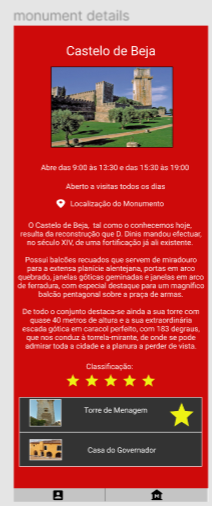


Em seguida, foram criados os ecrãs necessários para a consulta de informação dos monumentos.

Primeiramente foi criado um ecrã que possui uma lista de monumentos, constituída por uma imagem do monumento e nome do monumento. Juntamente com esta lista, foi criada uma barra de pesquisa que permite procurar um monumento pelo nome (embora não tenha conseguido de facto implementar no design do Figma).



A seguir, foi criado mais um ecrã da aplicação, que representa a página de um monumento. Neste ecrã, é mostrado as várias informações de um monumento, sendo estas informações o nome do monumento, imagem, horário de funcionamento, dia de encerramento, localização do monumento, descrição do monumento, classificação do monumento (a qual será determinada através da média das avaliações dos pontos de interesse correspondentes ao monumento) e uma lista que contém os pontos de interesse de cada monumento, constituída por uma imagem do ponto de interesse, pelo seu nome e por um possível ícone, que representaria se o utilizador já viu aquele ponto de interesse específico ou não.



Por fim, foi criado um último ecrã, responsável por apresentar ao utilizador os detalhes do ponto de interesse selecionado no ecrã anterior.

Neste ecrã, é possível ver o nome do ponto de interesse, foto, e descrição.

É possível ainda avaliar o ponto de interesse com entre 1 e 5 estrelas.

Por fim, foi criado ainda um botão que, ao ser clicado, irá selecionar/desseleccionar o ponto de interesse como visto



Relativamente ao protótipo do Website, realizado pela Wix, foi escolhido um template para a sua realização.

Começou-se por criar a barra de navegação, que contem o menu home, que leva para a página inicial, e um menu de monumentos. Dentro do item de Monumentos existem ainda subitens, correspondentes aos monumentos contidos na base de dados.

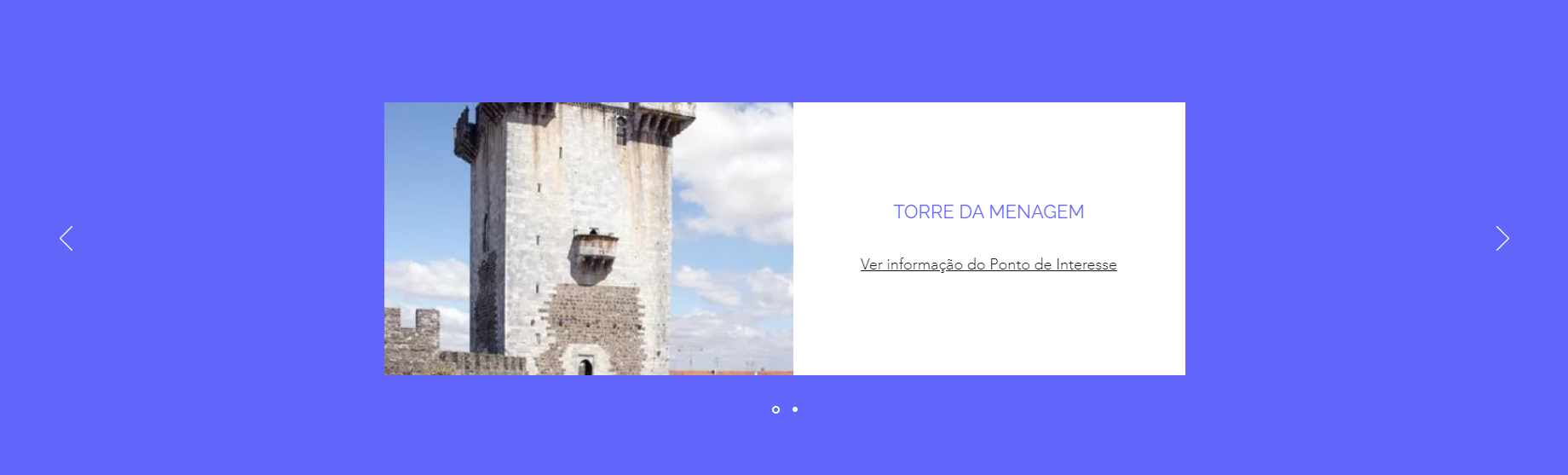
Foram então criadas as duas seções da página inicial, que consistem numa capa para o website, e uma secção onde são disponibilizados os vários monumentos presentes na base de dados, mostrando a foto do monumento, nome, descrição e uma opção “learn more”, que possui uma hiperligação para a página do monumento em questão.





Ao clicar na opção “Learn more” de determinado monumento, o utilizador será redirecionado para outra página, que lhe apresentará os detalhes do monumento, sendo esses detalhes o nome do monumento, imagem, horário de funcionamento, descrição e localização, sendo esta última uma hiperligação para o link do *Google Maps* do monumento, permitindo assim que o utilizador confira onde o monumento se encontra.

Numa outra secção abaixo, foi criado um slide show, que apresenta os vários pontos de interesse presentes no monumento a ser visualizado, apresentando cada slide a foto do ponto de interesse, o nome e uma hiperligação que irá redirecionar o utilizador para a página do ponto de interesse.



Por fim, foi criada uma página adicional para cada ponto de interesse, que possui a foto do ponto de interesse como fundo da página, com uma caixa de texto no centro da página. Dentro desta caixa de texto é apresentado o nome do ponto de interesse e a sua descrição, assim como um botão para que o utilizador consiga voltar para trás.



## Projeto Laravel

Para criar o projeto, foi inserido, na linha de comandos, o comando “laravel new monuments”, que gerou a pasta do projeto denominada “monuments”, assim como os ficheiros necessários para o desenvolvimento da app web.

Em seguida, foi instalado o livewire, que criou o código base necessário para a autenticação referente ao acesso do back end da aplicação, através dos comandos “php artisan jetstream:install livewire”, seguido dos comandos “npm install run dev” e “php artisan migrate”, assim criando os ficheiros necessários.

Após a criação do projeto, foi criada a base de dados no xampp. Em seguida, foram criadas as migrações referentes às tabelas da base de dados, através do comando “php artisan make:migration (nome da tabela)”, que cria os ficheiros onde os atributos da tabela são definidos, seguido do comando “php artisan migrate”, para que as tabelas fossem criadas no xampp.

A seguir, foram criados os modelos necessários, que seriam o modelo de Monumentos e o Modelo de Pontos de Interesse, através do comando “php artisan make:model (nome do model)”, onde foram definidos .

Foram ainda criados controladores para cada modelo, pelo comando “php artisan make:Controller (nome do controlador)”, para criar os controladores necessários.

Em seguida foram criadas as Routes, no ficheiro “web.php”. Aqui foram criadas as várias routes responsáveis pela navegação do site, tanto para o back end, que requeriria validação de conta, como para o utilizador. No ficheiro “api.php”, foi ainda criado o acesso à api, através dos controladores, que permitiria que o website conseguisse ir buscar as informações presentes nas tabelas da base de dados.

Por fim, foram criadas as Resources necessárias para o back end, através do comando “php artisan make:Resource (nome da resource)”, para criar as pastas referentes aos monumentos e aos pontos de interesse. Foram ainda criadas blades para a lista de monumentos/pontos de interesse disponíveis, visualização de um monumento/ponto de interesse específico, alteração da informação e para apagar um monumento/ponto de interesse específico.

A seguir, foi criada uma blade que representaria a página principal do website. À medida que se foi escrevendo o código html, acabou por ser necessário adicionar Routes que permitissem que fosse possível obter os dados acerca dos monumentos e pontos de interesse.

Conforme se foi desenvolvendo as páginas do website, foram surgindo várias dúvidas. Entre essas dúvidas, a única que não se conseguiu resolver a tempo foi a do carousel, atributo necessário para a implementação de um slideshow que mostrasse ao utilizador os vários pontos de interesse associados ao monumento selecionado, acabando assim por optar por uma lista de pontos de interesse para que não se perdesse muito tempo, já que na altura o tempo até a apresentação já estava escasso.

Por fim, após terminado, a aplicação laravel foi publicada através do site …, para que a aplicação android pudesse acessar a api e, consequentemente, fosse possível mostrar os dados dos monumentos e pontos de interesse na aplicação móvel.

## Aplicação Android

Começou-se por criar um projeto no Android Studio.

Depois da criação do projeto, foram criadas as várias classes que se viram necessárias para o desenvolvimento do projeto. Estas classes foram, inicialmente, três *activities*, sendo estas a *activity* de registo de conta de utilizador, a *activity* de login (que é também a *activity* principal) e a *activity* de monumentos.

Na *activity* de login foram criadas 2 *edittext*, uma para o e-mail e uma para a password, juntamente com 2 botões, sendo um utilizado para navegar para a *activity* de registo de conta e outro para navegar para a *activity* de monumentos.

Na *activity* de registo de conta, foram criadas 6 *edittext*, sendo estas para o primeiro nome, último nome, número de telemóvel, e-mail, password e confirmação da password. Foi criado ainda um botão para registar a conta e um botão para regressar ao ecrã de login, caso o utilizador tenha navegado para este ecrã acidentalmente.

A ideia inicial seria criar um *fragment container*, que conteria o conteúdo da lista de monumentos, que iria ser substituído pelo *fragment* seguido conforme necessário. No entanto, não se conseguiu fazer isso uma vez que surgiram dificuldades relativamente à passagem de argumentos de um *fragment* para outro. Após várias tentativas de passar argumentos entre os *fragments*, desistiu-se dessa ideia inicial, e a aplicação foi feita apenas com *Activities*, uma vez que o tempo já estava curto e era a forma mais simples para que houvesse o que apresentar.

Sendo assim, foram criadas mais duas *activities*, sendo elas a *activity* de detalhes de um monumento específico e a *activity* de detalhes de um ponto de interesse específico.

Na *activity* de detalhes de um monumento, foram criadas *textview* para o nome, horário de funcionamento, dia de encerramento, localização e descrição do monumento, uma *imageview* que contem a foto do monumento, uma *ratingbar* que apresenta a classificação do monumento, baseado na média das classificações dos pontos de interesse ligados ao monumento em questão. Esta *activity* possui ainda uma lista de pontos de interesseligados a este monumento que, podendo cada linha desta lista conter um ícone de estrela caso o utilizador o marque como visto.

Por fim, para a *Activity* de detalhes de um ponto de interesse, foram criadas *textview* para o nome do ponto de interesse e para a descrição, assim como uma *imageview* para a foto do monumento. Foi criado ainda um botão que permite alterar se o ponto de interesse já foi visualizado ou não. Foi criada ainda uma *ratingbar*, para que o utilizador possa avaliar o ponto de interesse. Por fim, foi criado ainda mais um botão, que permite submeter a classificação.

Foi também criada uma API fake, através do “my JSON server”, que permitiu o desenvolvimento da aplicação antes de conectá-la à aplicação Laravel.

Foram criada duas interfaces, uma para servir de *Dao* para os pontos de interesse, onde seria realizada a *query* necessária que fosse possível marcar um ponto de interesse como visto e outra para os utilizadores, onde seria realizada a *query* para a autenticação dos utilizadores (embora não tenha havido tempo para implementar as funcionalidades desejadas).

Foi criada também uma classe abstrata, com o intuito de servir como a base de dados da aplicação. Para tal foi necessário importar a biblioteca *Room*.

Foram criados *adapters* para os monumentos e para os pontos de interesse, para que fosse possível preencher as *listview* com dados programaticamente.

Para além da biblioteca *Room*, viu-se necessário incluir também outras bibliotecas, sendo estas o G*lide*, para a implementação das imagens e o *Retrofit*, para o uso dos *viewmodels* e do *repository.*

Após a criação da base de dados da aplicação, foi criado o *datasource*, responsável por fornecer à aplicação o url base da api, assim como o *appservice*, que fornece os caminhos necessários para que a aplicação consiga obter as informações pretendidas, relativamente aos monumentos e pontos de interesse (não se viu necessidade de fazer o mesmo para os utilizadores, uma vez que foi decidido que ficariam apenas na aplicação android).

Foram então criados os *viewmodels* para os monumentos e pontos de interesse, ambos ligados a um único *repository*. O *Repository* tem a responsabilidade deacessar o *appservice*, para que consiga os dados necessários e devolvê-los para o *viewmodel* que, por sua vez, tem a responsabilidade de receber estes dados e devolvê-los para as activities.

## Dificuldades Sentidas ao longo do projeto

Logo de Início, senti alguma insegurança relativamente ao design que foi apresentado inicialmente, como esboço do projeto. Sinto que acabo por ser muito simples em relação ao desenho de ecrãs, e que é algo que eu deveria explorar mais.

Relativamente às dificuldades sentidas na prática, a principal parte do projeto que eu senti mais dificuldades foi na aplicação Laravel. Foi a parte que, sem dúvida, eu mais precisei da ajuda dos orientadores de Estágio. Acredito que se deve à falta de prática, acabei por me preparar bem mais para o desenvolvimento de aplicações android e por não praticar tanto assim na parte do desenvolvimento Web, o que me causou algum desânimo em determinado momento do estágio, por sentir que não era capaz de o fazer.

Acredito que o principal erro cometido ao longo do Estágio foi tentar desenvolver tudo por mim mesmo. Acabei por consultar a internet quando precisava, mas em determinado momento acabei por ficar cada vez mais confuso conforme tentava relembrar os conteúdos de html e css. Acabei por pedir ajuda aos orientadores tarde demais, o que resultou em uma correria para tentar terminar o projeto a tempo de entrega, já não tendo tempo suficiente para desenvolver tudo o que pretendia para este projeto.

## Conclusão

Com este estágio, aprimorei os conhecimentos que já tinha sobre desenvolvimento de aplicações, tanto laravel como android. Acabei por ficar com uma ideia melhor sobre os fragments (mesmo não tendo conseguido implementar completamente na aplicação), que era algo que eu não tinha entendido bem no primeiro semestre do curso.

Gostaria, no entanto, de ter conseguido implementar uma procura de monumentos por nome na aplicação android, mas acabei por não conseguir implementar essa funcionalidade a tempo. Para além disso, gostaria ainda de melhorar a questão de design de ecrãs daqui para a frente.

Embora tenha tido vários contratempos, acredito que tenha feito um bom trabalho no geral.