



UNIVERSIDADE
LUSÓFONA

Plataforma de gestão de eventos – mobile app

Trabalho Final de curso

Relatório Final

Aluna: Catarina Moita

Orientador: Rodrigo Correia

Coorientador: Bruno Cipriano

Trabalho Final de Curso | LEI | 25/06/2021

www.ulusofona.pt

Direitos de cópia

Plataforma de gestão de eventos – mobile app, Copyright de *Catarina Moita, Rodrigo Correia & Bruno Cipriano*, ULHT.

A Escola de Comunicação, Arquitectura, Artes e Tecnologias da Informação (ECATI) e a Universidade Lusófona de Humanidades e Tecnologias (ULHT) têm o direito, perpétuo e sem limites geográficos, de arquivar e publicar esta dissertação através de exemplares impressos reproduzidos em papel ou de forma digital, ou por qualquer outro meio conhecido ou que venha a ser inventado, e de a divulgar através de repositórios científicos e de admitir a sua cópia e distribuição com objectivos educacionais ou de investigação, não comerciais, desde que seja dado crédito ao autor e editor.

Resumo

O presente Trabalho Final de Curso (TFC) consiste no desenvolvimento de uma aplicação móvel que integre um sistema capaz de suportar um clube de corrida de uma empresa.

Assente no objetivo de colmatar uma lacuna, apontada pelas partes interessadas, esta aplicação permite oferecer um veículo de comunicação entre os funcionários da empresa, no detrimento da divulgação e gestão de eventos desportivos.

Nesse sentido, ao longo deste trabalho, pretende-se documentar em detalhe, no âmbito do TFC, as várias fases a que foi exposto, desde o planeamento inicial ao desenvolvimento de uma versão preliminar inteiramente funcional e testada.

Estruturalmente, o relatório começa pela contextualização e identificação do problema concreto em estudo, apresentando todas circunstâncias que levaram a esta proposta a ser realizada, como esta parte de um problema real de uma organização e quais as soluções apresentadas para o resolver. Sempre que aplicável, serão produzidas conclusões, na medida de validar o produto final obtido, face ao proposto na primeira entrega.

Segue para a escrutinação da capacidade do trabalho poder ser continuado e implementado no dia a dia, mesmo após a conclusão da avaliação para fins académicos e as vantagens que traria o seu desenvolvimento.

A fim de avaliar o sucesso do trabalho, alusivo ao cumprimento de todas as tarefas propostas, é imprescindível a análise comparativa de quais os requisitos implementados e justificar eventuais diferenças. Será ainda dado especial atenção às tecnologias utilizadas, com o rigor técnico que assim o exige, o porquê destas serem as opções mais viáveis e o impacto que tiveram na construção da solução.

Palavras-chave: aplicação móvel, atividades desportivas, gestão de eventos, Flutter

Abstract

This Final Course Work (TFC) consists of the development of a mobile application that integrates a system capable of supporting a company's racing club.

Based on the objective of filling a gap, pointed out by the interested parties, this application offers a means of communication between the company's employees, in the dissemination and management of sporting events.

In this sense, throughout this work, it is intended to document in detail, within the scope of the TFC, the various phases to which it was exposed, from the initial planning to the development of a fully functional and tested preliminary version.

Structurally, the report begins with the contextualization and identification of the concrete problem under study, presenting all the circumstances that led to this proposal being carried out, such as this part of a real problem of an organization and the solutions presented to solve it. Whenever applicable, conclusions will be produced, to the extent that the final product obtained is validated against the one proposed in the first delivery.

It goes on to scrutinize the capacity of the work to be continued and implemented on a day-to-day basis, even after the completion of the assessment for academic purposes and the advantages that its development would bring.

In order to assess the success of the work, referring to the fulfillment of all the proposed tasks, it is essential to carry out a comparative analysis of the implemented requirements and justify any differences. Special attention will also be given to the technologies used, with the technical rigor that requires it, why these are the most viable options and the impact they had on the construction of the solution.

Keywords: mobile application, sports events, event management, Flutter

Índice

Resumo.....	iii
Abstract	iv
Índice.....	v
Lista de Figuras	vii
Lista de Tabelas	viii
1 Identificação do Problema	1
2 Levantamento e Análise dos Requisitos.....	2
2.1 Requisitos Funcionais	2
2.2 Requisitos Não Funcionais	3
3 Viabilidade e Pertinência.....	4
4 Solução Desenvolvida.....	6
4.1 Funcionalidades.....	6
4.2 Tecnologias.....	8
4.3 Arquitetura	10
4.4 <i>Back-end</i>	10
4.5 <i>Front-end</i>	11
4.5.1 Protótipo Axure	11
4.5.2 Conceção	12
4.6 Código Fonte	14
4.7 Demo	14
5 Benchmarking.....	15
6 Método e Planeamento	19
7 Resultados	21
8 Conclusão e Trabalhos Futuros	27
Bibliografia	28
Anexo 1 – Tabela de Requisitos Funcionais	31
Anexo 2 - Implementação do método GET	37
Anexo 3 - Implementação do método POST	38
Anexo 4 - Implementação do LayoutBuilder() e MediaQuery.of()	39
Anexo 5 - Implementação do Provider	40

Glossário.....	41
----------------	----

Lista de Figuras

Figura 1 - Gráfico das inscrições e provas ao longo dos anos	4
Figura 2 - Mapa aplicacional.....	7
Figura 3 - Quota de mercado mundial dos sistemas operativos móveis em setembro de 2020	9
Figura 4 - Arquitetura do sistema, com ênfase na aplicação móvel.....	10
Figura 5 - Ecrã <i>login</i> Axure.....	11
Figura 6 - Ecrã principal Axure.....	11
Figura 7 – Menu administrador Axure.....	12
Figura 8 – Ecrã estatísticas Axure	12
Figura 9 - Padrão de arquitetura do <i>software</i>	14
Figura 10 - Aplicação Eventsport	15
Figura 11 - Aplicação Running Lisboa	16
Figura 12 - Aplicação Pronto a Correr	16
Figura 13 - Aplicação Run Races	17
Figura 14 - Aplicação RamRun	17
Figura 15 - Gráfico de Gant proposto inicialmente	19
Figura 16 - Modelo em cascata modificado adaptado ao TFC	19
Figura 17 – Processo de login na aplicação móvel	21
Figura 18 - Menu administrador.....	22
Figura 19 - Menu membro.....	22
Figura 20 - Pesquisa de eventos	22
Figura 21 - Inscrição em evento	23
Figura 22 - Mensagem de erro quando o plafond é insuficiente	23
Figura 23 - Lista de eventos pendentes de aprovação	24
Figura 24 - Lista de membros pendentes de aprovação.....	24
Figura 25 - Mensagem de rejeição de membro	25
Figura 26 - Processo de filtragem de estatísticas	25
Figura 27 - Registo de tempos.....	26

Lista de Tabelas

Tabela 1 - Funcionalidades da aplicação móvel e o seu estado	6
Tabela 2 - Comparação da aplicação a ser desenvolvida com aplicações concorrentes	18

1 Identificação do Problema

No ano letivo 2019/2020 foi feita uma proposta no âmbito da unidade curricular Trabalho Final de Curso cujo objetivo era desenvolver um sistema de gestão de eventos, baseado em tecnologia web, para ser aplicado numa Pequena e Média Empresa (PME) portuguesa. Este sistema, realizado pelos alunos a que o tema foi atribuído, consiste numa plataforma para uso interno por parte da organização, onde os membros (funcionários da empresa associados a um *plafond* predefinido) têm a possibilidade de se inscreverem em eventos desportivos, tais como, corridas e caminhadas, que sejam do seu interesse.

A sua implementação veio contribuir para uma maior centralização dos dados e facilitar o esforço de gestão do clube que, outrora dependia de inúmeros ficheiros Excel para guardar todas as informações dos funcionários registados como membros e da plataforma externa Doodle [1] como meio para efetuar as inscrições em cada evento.

Contudo carecia de uma aplicação móvel que apoiasse esta componente web. Trata-se, portanto, de uma necessidade real de uma empresa.

Face ao problema exposto, o produto final deste trabalho é uma aplicação móvel complementar ao sistema web já existente, para uso interno de uma *software house* nacional, que permite a inscrição dos seus funcionários em atividades desportivas, fazer a gestão de membros e eventos e oferecer um serviço mais personalizado com o registo dos tempos pessoais de conclusão de provas e a opção de acompanhar a sua evolução estatística.

O sistema desenvolvido teve por base a proposta que serviu de prelúdio ao primeiro momento de avaliação mas, como em qualquer processo natural de desenvolvimento de projetos, sofreu algumas alterações resultantes de fatores como a evolução de conhecimentos da temática, alteração de prioridades dos intervenientes e algumas inconsistências encontradas.

Comparativamente ao idealizado inicialmente, não foi adotada a funcionalidade de registo do utilizador. Por outro lado, acrescentou-se a opção de filtrar os tempos de conclusão de prova por tipo de atividade, distância percorrida e/ou intervalo de datas e a opção de um administrador poder utilizar a barra de pesquisa para encontrar, pelo nome, membros e eventos pendentes de aprovação. Adicionalmente, integrou-se a possibilidade de realizar a avaliação de um evento.

2 Levantamento e Análise dos Requisitos

O processo de identificação detalhado de requisitos e, posteriormente, a sua análise, no desenvolvimento do sistema, de modo a responder aos problemas e expectativas da organização a que se destina, foi de extrema importância para o sucesso do projeto. Desta forma, evitou-se erros, como atrasos na conceção da aplicação (consequente da elaboração de requisitos pouco realistas dado ao tempo disponível e objetivos calendarizados) ou até casos mais graves como o produto final não corresponder ao idealizado pelo cliente.

Estes requisitos são, portanto, compatíveis mediante as necessidades comunicadas durante a fase de elicitação e têm em conta quaisquer conflitos e dependências que possam existir entre eles em prol da conservação da integridade de cada um. Dada a sua aprovação, este documento serviu como base para a implementação da aplicação móvel.

Foram identificados dois tipos de requisitos: funcionais e não funcionais.

Optou-se por realizar os requisitos funcionais na forma de *user stories*, aplicados no desenvolvimento Agile [2]. “No desenvolvimento de *software* e gestão de produto, uma *user story* é uma descrição informal em linguagem natural de uma ou mais características do sistema de *software*” [3]. Articulam na perspetiva do utilizador final o “para quem”, “o quê” e o “porquê” de forma simples e eficaz, poupando tempo na escrita exaustiva de requisitos e dando ênfase ao que o cliente realmente quer. Este método traduz-se num benefício a longo prazo perante um cenário de continuidade do projeto além do TFC, uma vez que, a sua versatilidade e a preocupação que existe em evitar entrar em muitos detalhes prematuramente vão permitir a exploração de novas funcionalidades e evitar a restrição de possíveis soluções no momento de implementação de código.

No decorrer deste trabalho surgiram assim quinze *user stories*, onde onze são do ponto de vista do membro e do administrador e os restantes quatro restritos ao utilizador do tipo administrador.

2.1 Requisitos Funcionais

Requisitos que descrevem o comportamento que a aplicação deve ter, quais as suas características e funcionalidades. A tabela correspondente encontra-se representada no Anexo 1.

Quanto ao estado de implementação, na componente móvel está tudo implementado em termos de Interface de Utilizador (UI), ficando apenas a faltar as ligações servidor/*web services* nas *user stories* correspondentes à consulta de estatísticas (por enquanto os dados são *hardcoded*) e na ação de rejeição de eventos e membros.

Tanto o requisito de ID 14 - filtrar os tempos de término de prova registados - como de ID 15 - avaliar um evento - foram acrescentados por orientação. Foram ainda aplicadas algumas modificações no requisito ID 4 (adição do ponto três nos critérios de aceitação, que impede a inscrição em eventos cujo prazo já tenha terminado) e ID 12 (adição do ponto três e quatro, que permitem ao utilizador, na devida ordem, verificar o total de quilómetros e de tempos percorridos).

2.2 Requisitos Não Funcionais

Requisitos que especificam os atributos de qualidade do sistema. Têm um papel tão crítico no sucesso da aplicação como os requisitos funcionais, visto que afetam diretamente a experiência do utilizador.

- a) A aplicação irá assegurar que apenas os utilizadores devidamente registados na Base de Dados do sistema e aprovados por um administrador possam efetuar o *login* com sucesso e ter acesso às funcionalidades da plataforma;
- b) A aplicação deverá ser compatível com todas as versões suportadas [4] dos dispositivos Android com Sistema Operativo (SO) Android Jelly Bean, v16, 4.1.x ou mais recente e dispositivos iOS com SO iOS 8 ou mais recente e *hardware* iPhone 4S ou mais recente;
- c) A Interface de Utilizador e Experiência de Utilizador (UI/UX) do sistema deverá estar em concordância com as práticas de usabilidade das aplicações móveis [5];

3 Viabilidade e Pertinência

Mas porquê uma aplicação móvel?

Como se pode observar na Figura 1, verifica-se que durante o período dos anos 2017 a 2019¹, apesar do número de provas disponíveis ter aumentado aproximadamente 47,8% em relação ao primeiro ano desta iniciativa, assistiu-se a uma queda de cerca 60% do número de inscrições.

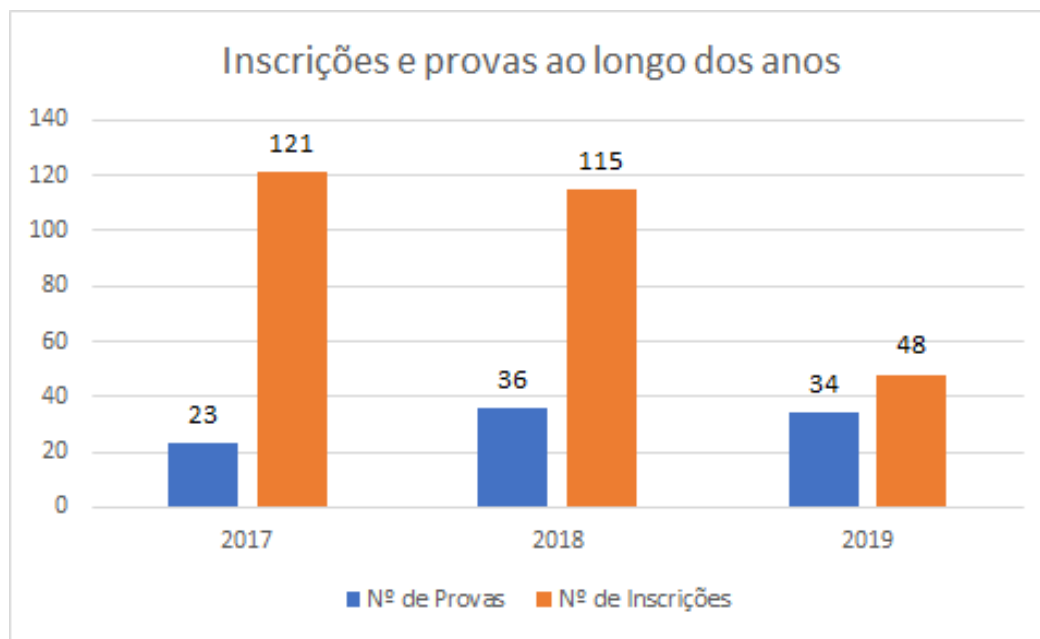


Figura 1 - Gráfico das inscrições e provas ao longo dos anos

A capacidade de facultar uma nova experiência aos utilizadores, através deste dispositivo, em qualquer local a qualquer hora é o cerne das aplicações móveis. A ideia de ter a plataforma de gestão de eventos em formato móvel resulta numa maior conveniência tanto para o administrador que pode gerir todos os eventos sempre que o entender, como para o utilizador na inscrição de provas, por exemplo, sem ter de andar constantemente com um computador atrás sempre que se quer realizar alguma ação.

Inclusive, o utilizador tem a possibilidade de receber notificações não intrusivas de novos eventos disponíveis sem ter de aceder ao email da empresa, de registar o tempo em que concluiu a prova e, com isto, consultar estatísticas personalizadas (prova mais rápida, mais lenta, entre outras) diretamente no telemóvel.

De destacar que a aplicação móvel oferece a opção de fazer a integração com as redes sociais (Twitter, Instagram, Facebook). Significa isto que, os funcionários da empresa, podem partilhar nas diversas redes sociais em que provas vão participar e associar o nome da empresa às publicações. Neste sentido, qualquer pessoa que se cruza-se com estas publicações poderia ter interesse em saber mais sobre a organização, contribuindo para a difusão da mesma.

Uma vez definida a pertinência do projeto deve-se agora estudar a sua viabilidade.

Durante o desenvolvimento de qualquer aplicação depara-se com várias questões mas algo tem de ser tido sempre em conta: o produto final tem de funcionar no maior número de

¹ Derivado da pandemia não tem havido participação em provas, logo não existe registo de dados mais recentes

dispositivos possíveis, tornando-o assim mais abrangente. A decisão de implementar a aplicação com suporte multiplataforma foi crucial neste aspeto (no capítulo correspondente à Solução Desenvolvida será dado a conhecer como é que este conceito se materializou) .

O facto deste projeto não só ter um enorme carácter interdisciplinar, por incluir muitos dos conteúdos programáticos abordados ao longo do curso de Engenharia Informática e relevantes para a conclusão do percurso académico, mas também de reunir todas as condições para o seu resultado final ser adotado num contexto real e até continuado no futuro (acrescentar ou mudar funcionalidades consoante as necessidades da empresa) mostra a sua aplicabilidade e viabilidade.

Assim que se reunir todas condições adequadas para colocar o *software* num ambiente real a ser testado pelas partes interessadas, será um processo imprescindível na confirmação de que a aplicação móvel desenvolvida representa um passo para a solução do problema proposto no presente TFC, contornando a tendência que se tem observado até agora.

4 Solução Desenvolvida

4.1 Funcionalidades

Ao fim de desenvolver a solução mais adequada para o problema em estudo e coerente com os requisitos expostos no capítulo anterior, foram implementadas as funcionalidades apresentadas na Tabela 1.

Tabela 1 - Funcionalidades da aplicação móvel e o seu estado

Funcionalidades	Estado Geral	Comentários
Login	Completo	-
Aprovar e/ou rejeitar membro	Incompleto	Sem <i>web service</i> associado à ação de rejeitar
Aprovar e/ou rejeitar evento	Incompleto	Sem <i>web service</i> associado à ação de rejeitar
Procurar membro pendente de aprovação	Completo	-
Procurar evento pendente de aprovação	Completo	-
Procurar evento	Completo	-
Inscriver em evento	Completo	-
Registar o tempo de conclusão de prova ²	Incompleto	Sem <i>web service</i> associado
Consultar estatísticas sobre tempos de prova: melhor/pior tempo e tempo/distância percorridos no total ²	Incompleto	Sem <i>web service</i> associado
Filtrar estatísticas: distância ²	Incompleto	Sem <i>web service</i> associado
Filtrar estatísticas: tipo de atividade ²	Incompleto	Sem <i>web service</i> associado
Filtrar estatísticas: intervalo de datas ²	Incompleto	Sem <i>web service</i> associado
Partilhar nas redes sociais ²	Completo	-
Avaliar evento ²	Completo	-

² Funcionalidades exclusivas do sistemas *mobile*

Considerando a Tabela 1, foi proposto converter para a vertente móvel a generalidade das funcionalidades de utilizador e algumas de administrador já definidas na componente web preexistente e outras opções extras anteriormente visadas na secção Levantamento e Análise de Requisitos.

Esta segmentação de funcionalidades para cada tipo de utilizador confere à aplicação a capacidade de se adaptar a cada contexto e situação. Tanto o membro como o administrador podem realizar o *login* através das credenciais que introduziram durante o registo efetuado na plataforma web e, consoante estas mesmas credenciais, será apresentado um conjunto de funcionalidades específicas a cada utilizador.

Enquanto membro, podemos aceder à lista completa de eventos, inscrever-nos tendo em conta o saldo do *plafond* disponível e procurá-los por características, como o nome, para facilitar a obtenção imediata dos resultados desejados, reduzindo assim a complexidade de navegação e poupando tempo ao utilizador. O membro pode ainda registar os tempos em que concluiu uma determinada prova, consultar as estatísticas resultantes da incorporação destes dados, aplicar filtros, partilhar os eventos em que vai participar nas suas redes sociais e avaliá-los de forma a medir o seu nível de satisfação. No registo de tempos de conclusão de provas optou-se por não proceder a nenhuma verificação da veracidade do tempo introduzido, tendo em conta fatores como a distância da prova e o mínimo humanamente atingível, pois tal poderia ser perceptível pelo utilizador como algo hostil. Atendendo que os participantes são funcionários da mesma empresa e muitos vão aos eventos pelo convívio, será deixado ao critério de cada um a verificação dos tempos que introduzem, sendo apenas validado se os valores são positivos e não nulos.

Já enquanto administrador, temos acesso às mesmas funcionalidades que o membro e, adicionalmente, assumir o papel de aprovar ou rejeitar membros e/ou eventos. O processo de pesquisa por nome (quer de membros como de eventos) foi igualmente implementado, mantendo o nível de sofisticação por toda a aplicação. De momento, tanto o registo de novos utilizadores como a criação de eventos caberá exclusivamente à plataforma web.

Esquemáticamente expõe-se na Figura 2 a estrutura da solução apresentada na forma de mapa aplicacional para o membro (a laranja) e para o administrador (a verde).

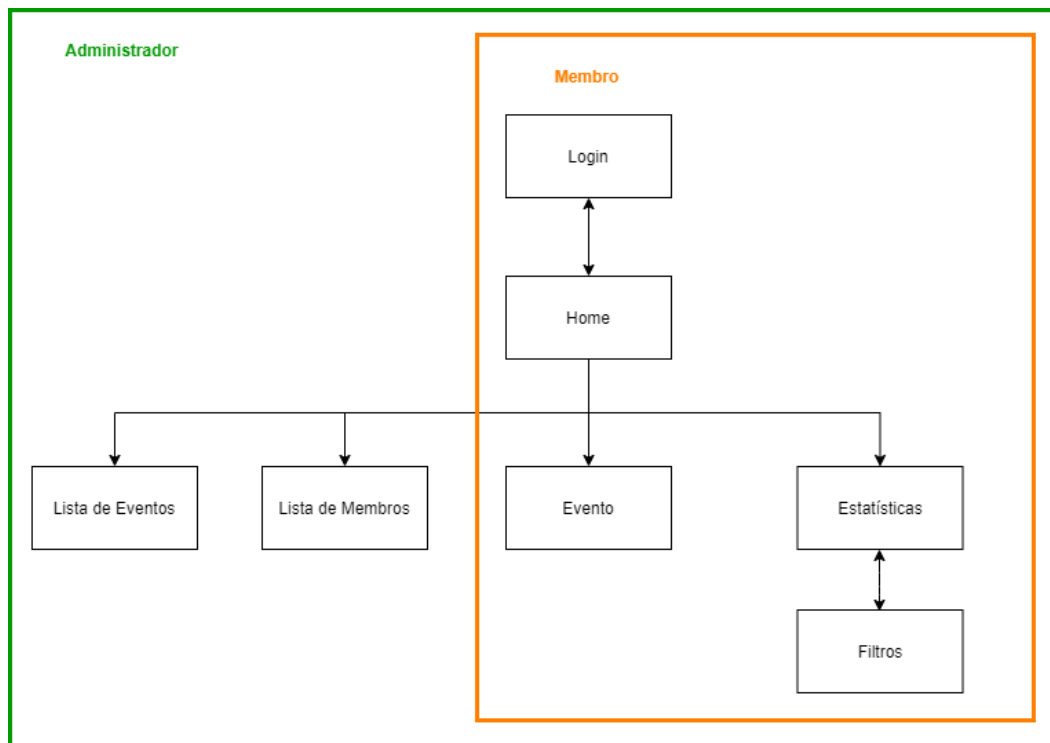


Figura 2 - Mapa aplicacional

De forma a abarcar todas as ideias fundamentais, a aplicação tem cinco ecrãs para o membro e sete ecrãs para o administrador. É apresentado um *splash screen* inicial em ambos os casos que, de seguida, exhibirá o ecrã de *login*. Feito o *login*, surge o ecrã principal com a lista de todos os eventos disponíveis. Ao interagir com um dos eventos, os utilizadores são direcionados para o ecrã do evento em questão que contém todos os detalhes associados (nome, tipo de atividade, localização, custo, data da prova e data limite de inscrição) e no qual podem inscrever-se e usufruir da opção de partilha nas redes sociais. Assim que o evento terminar, poderão então atribuir-lhe uma avaliação.

Ainda alusivo ao conjunto de ecrãs comuns aos dois tipos de utilizadores, tem-se o ecrã das estatísticas. Este ecrã será o foco do registo de tempos de término de provas e, face aos valores introduzidos, gera estatísticas sob forma de gráfico. Todos estes resultados podem ser filtrados por distância (5km, 10km, 21km e 42km), por tipo de prova (corrida ou caminhada) e por intervalo de datas.

Os ecrãs com a lista de membros e eventos (específicos do administrador) são dedicados à aprovação ou rejeição de membros e eventos, respetivamente, e fará uso da ferramenta de pesquisa abordada em pontos anteriores.

4.2 Tecnologias

Em vista da solução proposta, no que toca às tecnologias a utilizar para a sua concretização tem-se o Flutter para o *front-end*. O *back-end* irá fornecer dados no formato JSON, através de *web services* implementados em Common Lisp.

Através do mesmo código base o Flutter [6], uma *framework open source* criada pela Google, consegue construir aplicações quer seja para ambientes móveis (Android, iOS) como para web e desktop (Windows, macOS, Linux). O facto do Flutter ter esta flexibilidade e poupar o programador de reescrever o código para que a aplicação se possa adaptar a cada particularidade das diferentes plataformas traz vastos benefícios na gestão de tempo. A título de exemplo, se a empresa quisesse desenvolver a aplicação móvel tanto para Android como para iOS, não teria a necessidade de contratar duas equipas diferentes para cada sistema, contribuindo assim para a diminuição dos custos de produção ou ter uma equipa responsável pelo desenvolvimento para ambos os SO, aumentando a quantidade de tempo gasto na migração de plataformas que, como alternativa, pode ser investido na otimização da aplicação. É de extrema importância ter sempre o futuro em conta e pensar na estrutura de um programa de modo a que seja de fácil manutenção e mudança.

No caso deste projeto, o foco do Flutter será na construção de uma aplicação móvel que execute em diferentes SO. De acordo com o Statcounter [7], em setembro de 2020 o Android tinha uma taxa de ocupação no mercado de 74,44% e o iOS de 24,98% (Figura 3). Com base nestes dados pode-se assumir que público alvo da aplicação em questão está concentrado nestes SO, evitando o cenário à partida ideal de construir uma aplicação para todas as plataformas referidas na Figura 3, mas que se tornaria contraproducente.

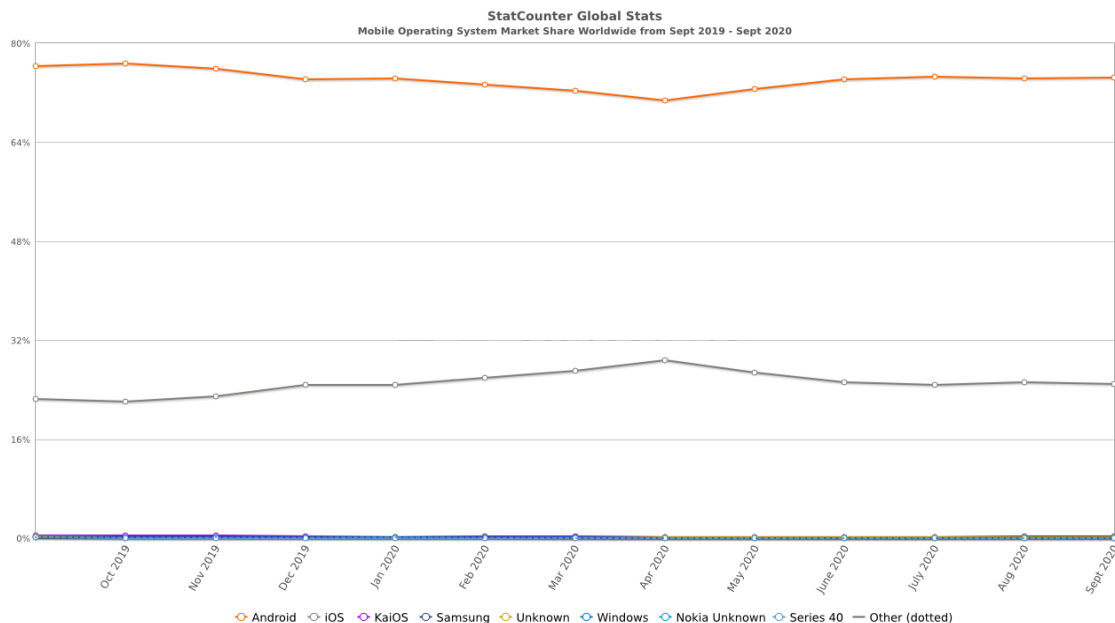


Figura 3 - Quota de mercado mundial dos sistemas operativos móveis em setembro de 2020

Para além da sua versatilidade, o Flutter distingue-se por ter uma performance equivalente a de uma aplicação nativa, já que não depende de intermediários para o código ser interpretado, por ser muito intuitivo, o que promove o rápido desenvolvimento, de usar a sua própria renderização de *widgets* dedicadas para cada SO e por oferecer a possibilidade de personalizar a UI de forma atrativa, respeitando as boas práticas de design, graças à linguagem de programação Dart [8] e ao mecanismo de renderização de alta performance Skia [9].

Tudo isto mostra o Flutter como sendo uma mais valia e a escolha acertada para o desenvolvimento deste trabalho.

Antes de explorar o conceito de JSON [10], deve-se definir a noção de *web services*. Os *web services* funcionam como um agregador de serviços, que comunicam com um servidor remoto através de tecnologias e protocolos Web da forma mais eficiente, eficaz e segura (não há acesso direto à Base de Dados) que os dispositivos móveis exigem. Oferecem uma enorme flexibilidade e dinamismo, atuando nos mais diversos SO e *hardware*, uma vez que cada aplicação pode ter a sua própria linguagem, que é mais tarde traduzida num formato intermediário. Neste caso será utilizado o formato JSON, uma sintaxe aplicada à troca e armazenamento de dados, usado em *web services* REST.

Mais importante ainda, o Flutter tem bibliotecas próprias para o desenvolvimento de *web services* REST baseados em JSON, coadunando estas duas tecnologias.

A necessidade de implementar *web services*, para dar resposta a requisitos específicos da aplicação móvel, foi atendida pela utilização da linguagem de programação multiparadigma Common-Lisp [11] adotada pelos alunos do ano passado durante a construção do lado do servidor, uma vez que os programadores da empresa alvo trabalham quase exclusivamente com esta linguagem.

4.3 Arquitetura

Em termos esquemáticos, a Figura 4 mostra a arquitetura do sistema em geral e como a componente móvel implementada (a azul) se liga ao sistema já existente (preto e branco):

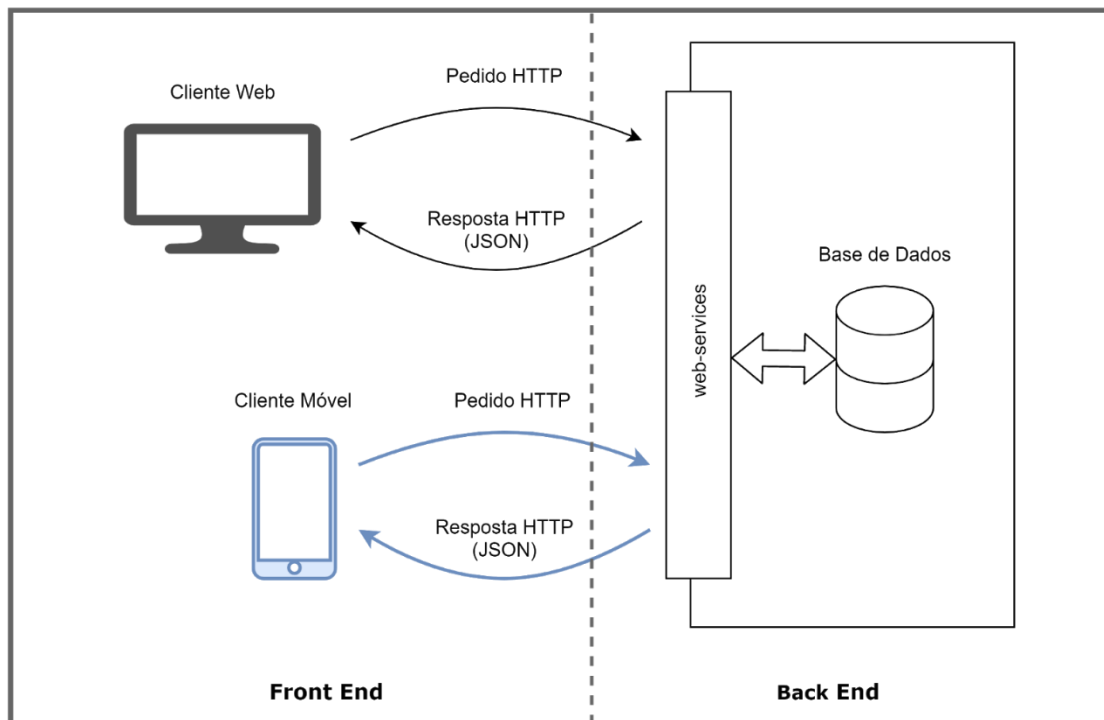


Figura 4 - Arquitetura do sistema, com ênfase na aplicação móvel

Tal como é visível na Figura 4, o sistema é constituído pelo *front-end* responsável pela UI, desenvolvida com recurso ao Flutter, onde tanto membro como administrador interagem diretamente, e comunica com o *back-end* via *web services* que, através do protocolo HTTP, transporta mensagens em formato JSON, de modo a consultar os dados presentes na mesma Base de Dados que a aplicação web usa, a fim de responder a todos os serviços requisitados.

4.4 Back-end

De forma a dispor um *back-end* capaz de operar na arquitetura do sistema definido no ponto anterior, a infraestrutura da Base de Dados usa uma instância de MongoDB [12], fundamentada pela Base de Dados construída pelos alunos do TFC que serviu de premissa a este, onde estão todas as informações relevantes como, por exemplo, a lista de eventos ou os membros registados na plataforma de gestão de eventos. A interação com a mesma é feita em Common-Lisp.

Para configurar o ambiente de desenvolvimento, foi fulcral a instalação do IDE Emacs [13] e das dependências do Steel Bank Common Lisp (SBCL) [14]. Uma vez que se faça a ponte entre estas duas ferramentas pelo pacote "sly" [15], torna-se possível interagir interativamente com uma REPL (Read-Eval-Print-Loop) e fazer *debug* com o Common-Lisp diretamente pelo Emacs.

O uso do gestor de bibliotecas Quicklisp [16] é igualmente intrínseco ao bom funcionamento do lado do servidor, debruçando-se na disponibilização de mais funcionalidades no Common-Lisp, como a de providenciar um servidor web e outras funcionalidades relacionadas ou uma interface para a Base de Dados MongoDB e outros utilitários.

Todos estes elementos são chave na dinâmica do *back-end* e, como mencionado precedentemente, vão possibilitar a ligação com o *front-end* através de *web services*. Dados como a lista de eventos ou a lista de membros serão transportados em formato JSON e, consoante o método de requisição definido no protocolo HTTP (GET ou POST), irão ser enviados para a aplicação móvel, em conjunto com o *package* “http” [17] do Flutter, e convertidos no objeto Dart referente, para mais tarde poderem ser manipulados e expostos ao utilizador final.

No Anexo 2 está um exemplo de excerto de código responsável pela receção da lista de todos utilizadores, aprovados ou não, provenientes dos *web services* e, no Anexo 3, o envio de um utilizador que teve os seus atributos alterados (por exemplo, passar de estado pendente de aprovação para aprovado).

Embora o foco do trabalho fosse a aplicação móvel, ainda houve espaço para, sempre que assim o exigia, mexer no *back-end*.

4.5 Front-end

4.5.1 Protótipo Axure

Seguindo o fluxo natural de planeamento e de modo a complementar a solução já apresentada, surgiu a fase de prototipagem. Com recurso à ferramenta Axure RP 9 [18], o protótipo da aplicação móvel vai dar forma aos requisitos definidos, garantir a coerência da informação e simular o comportamento do sistema.

Para efeitos de demonstração, foram selecionados o ecrã de *login* (Figura 5), o ecrã principal (Figura 6), o menu do utilizador administrador (Figura 7), e o ecrã das estatísticas (Figura 8).

De realçar que todos os ecrãs desenvolvidos no Axure não têm como objetivo representar o estado final da aplicação mas sim de guiar na tomada de decisões UI/UX e durante o processo de escrita de código.

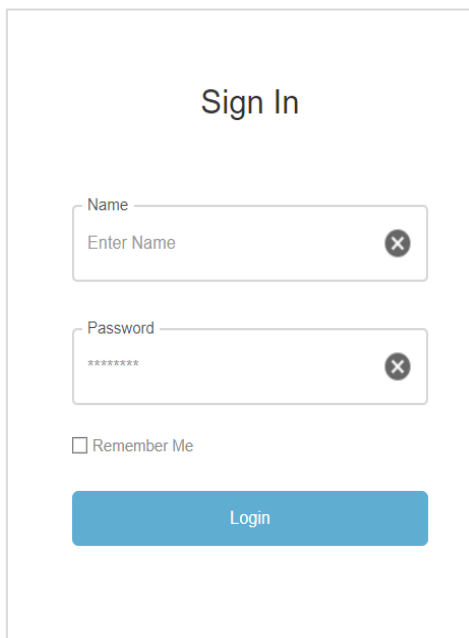


Figura 5 - Ecrã *login* Axure

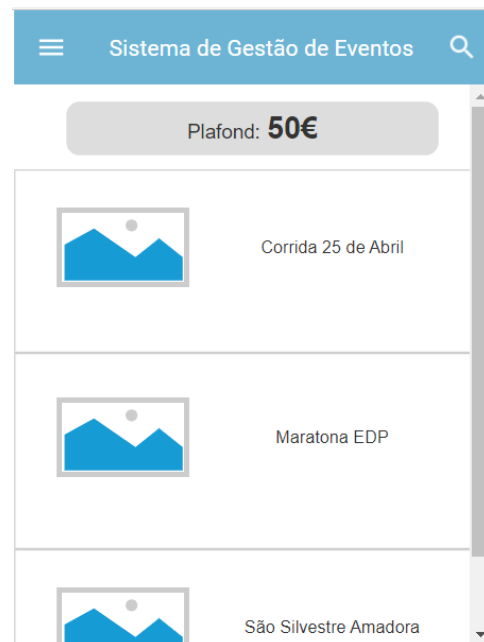


Figura 6 - Ecrã principal Axure

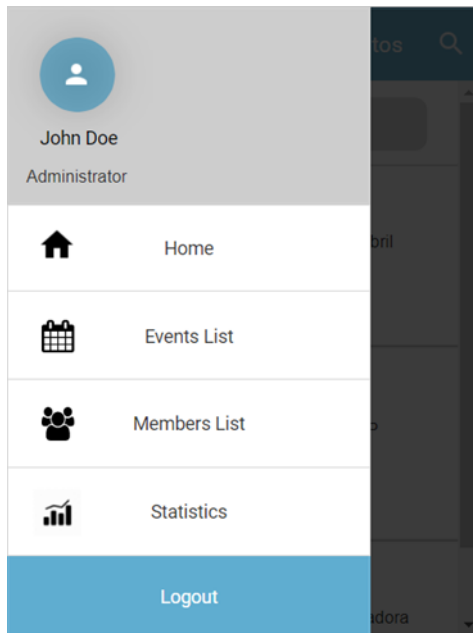


Figura 7 – Menu administrador Axure

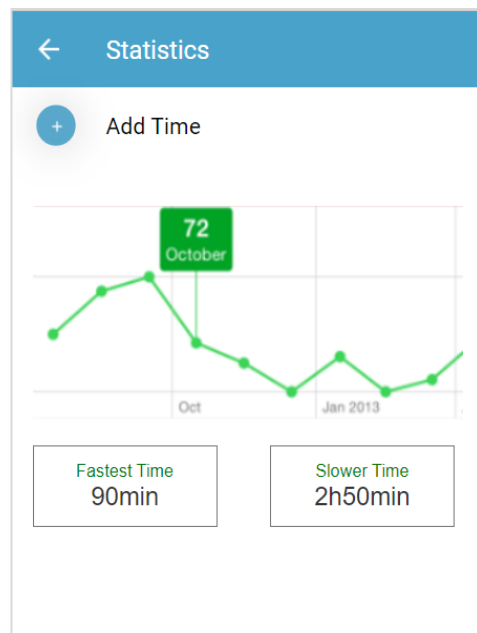


Figura 8 – Ecrã estatísticas Axure

4.5.2 Conceção

4.5.2.1 Responsive

Dada a construção deste protótipo, a aceitação dos requisitos apresentados no capítulo anterior pelas partes envolvidas e a configuração das tecnologias que a solução comporta, decidiu-se que estavam reunidas todas as condições para se proceder à fase de implementação de código.

O protótipo funcional foi desenvolvido através do Flutter, suportado pelo Android Studio, e concomitantemente testado através dos dispositivos móveis Pixel 2 API 30 na versão do Android 11.0 x86, Pixel API 23 na versão do Android 6.0 x86 e Nexus S API 23 na versão do Android 6.0 x86. Esta metodologia só foi possível recorrendo ao emulador integrado no IDE que garante quase todas as mesmas capacidades de um telemóvel real e permite simular a aplicação num leque de dispositivos com diferentes níveis de API de forma fácil e eficiente.

A escolha destes três dispositivos teve em conta fatores cruciais na preservação das características da aplicação:

- A versão do dispositivo - Analisar os diferentes comportamentos do projeto quando confrontado com uma API mais recente (R) *versus* uma API mais antiga (Marshmallow) é decisivo na escolha do tipo de ferramentas a aplicar, não limitando futuros utilizadores que tenham telemóveis mais antigos a não conseguir usufruir da aplicação no seu pleno potencial;
- O tamanho do ecrã - Procurar criar uma aplicação *responsive* e que com o mesmo código base se adapte às características de cada dispositivo, quer se esteja perante um ecrã de 5.0" ou de 4.0" deve ser tido constantemente em consideração. Este processo engloba uma estruturação do código de cada ecrã, recorrendo à classe `LayoutBuilder` [19] (constrói uma árvore de *widgets* dependente do tamanho do *widget* pai e quando esse tamanho é excedido, é ativado o *scroll*) e o método `MediaQuery.of()` [20] (retorna, por exemplo, as medidas do ecrã do dispositivo e adapta *widgets* consoante esse valor) como aparece nos excertos de código do Anexo 4 .

4.5.2.2 Navegação

Além do estudo inicial realizado acerca da natureza *responsive* da aplicação, foi analisado como seria gerida a sua navegação. A escrita de código não se resume exclusivamente à execução do mesmo, é necessário haver uma estruturação e organização da forma como os ecrãs comunicam uns com os outros, transferem dados e, em repercussão, mudam de estado.

Nesta ótica, foram estudadas três alternativas:

- `setState()` [21]
- BLoC [22]
- Provider [23]

A exclusão do uso do `setState()` por si só foi imediata. Este padrão obedece à hierarquia, não permitindo obter uma referência direta a *widgets* que estejam noutras ramificações da árvore senão abaixo. Noutras palavras, se o estado de um *widget* mudasse num dado ecrã, não se verificaria a condição primordial de poder atualizar todos os ecrãs que partilhem este componente. Já o BLoC distingue-se pelo uso de *streams* e *sinks* no transporte de dados e atua como uma camada intermediária entre a UI e lógica de negócio (tornando o código mais acessível de perceber e manter) mas peca pela sua complexidade de implementação e código *boilerplate*.

A partir da informação recolhida, optou-se pela utilização do *package* Provider.

O Provider, tal como o BLoC, é entendido como um modelo que separa a lógica de negócio da UI e que tem a particularidade de reconstruir a UI sempre que ocorre uma mudança de estado.

Considere-se o *widget* responsável pelo *plafond* que estará presente em todos os ecrãs. Sempre que um utilizador inscreve-se numa prova é debitado o valor da atividade e o estado, partilhado por diferentes *widgets* e ecrãs, tem de ser atualizado. Mas como é que cada ecrã sabe que o valor do *plafond* mudou? Como é que esse valor é atualizado?

Através desta abordagem, os *widgets* não têm qualquer conceito do tipo de dados que têm de mostrar nem são responsáveis pela sua alteração. Em vez disso, existe uma classe chamada Plafond que estende `ChangeNotifier` e notifica as partes interessadas se ocorreu mudanças (uma forma de `Observable`) e assim proceder ao *rebuild* do *widget* Text como mostra o Anexo 5.

Seguindo o princípio de uma arquitetura de lógica de negócio, o projeto foi dividido em cinco pastas:

- Constants - todos os valores que se mantêm imutáveis ao longo da aplicação (por exemplo, as cores);
- Models - estrutura de dados encarregue de converter os pares chave/valor JSON às correspondentes variáveis da entidade;
- Providers - contém o modelo Provider para cada *view* dos *widgets*;
- Screens - lida somente com a UI, isto é, como os dados são apresentados e como é realizada a receção de eventos espotelados pelo utilizador (cliques num dado botão, *scrolls*);
- Services - dá a resposta do servidor quando se faz chamadas à Base de Dados;

Como está ilustrado na Figura 9, os ficheiros presentes na pasta Providers e Models têm o compromisso em estabelecer a interação entre os ecrãs e os resultados dos pedidos dos *web services* à Base de Dados, fornecendo os dados de forma a que a UI os possa apresentar, mesmo não sabendo nada sobre a mesma.

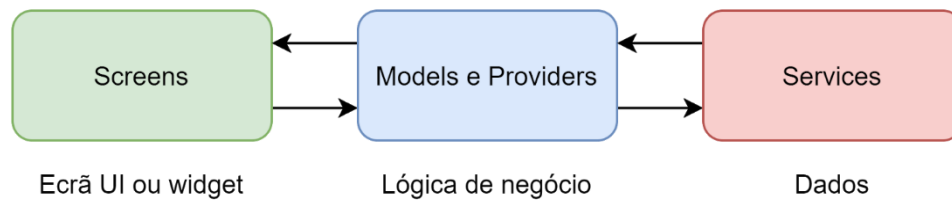


Figura 9 - Padrão de arquitetura do *software*

Este padrão é pertinente na medida em que promove uma maior disciplina durante a programação e, com a sua estratégia baseada na segregação de componentes, oferece uma cómoda oportunidade de crescimento da aplicação e de testagem de cada classe independente de outrem, o que traduz num benefício a médio/longo prazo se se proceder à continuidade do trabalho.

4.6 Código Fonte

O código fonte está disponível para consulta no seguinte repositório Git: https://github.com/catarina-21805632/projeto_tfc.git.

4.7 Demo

Todo funcionamento do protótipo funcional aqui descrito é posto em prática no seguinte link de Youtube: <https://youtu.be/yxxtl0q1ZJQ>.

5 Benchmarking

Para ganhar conhecimento do mercado, identificar pontos críticos que promovam o sucesso do projeto e arranjar metodologias que tornem a aplicação móvel capaz de responder aos serviços que as aplicações concorrentes já oferecem e saber o que fazer para inovar e preencher as lacunas que possam haver, é de extrema importância realizar o processo de análise e pesquisa de aplicações que sirvam o mesmo propósito que a deste trabalho.

Em comparação com o que já existe no que toca a plataformas de gestão de eventos desportivos temos:

- Eventsport [24]
- Running Lisboa [25]
- Pronto a Correr [26]
- Run Races [27]
- RamRun [28]

Eventsport

Descrição Oficial

“Aplicação pessoal para participação em eventos desportivos da Eventsport. Organização de eventos desportivos, gestão de inscrições, controlo de tempos, gestão de corridas e maratonas, equipas e atletas.”

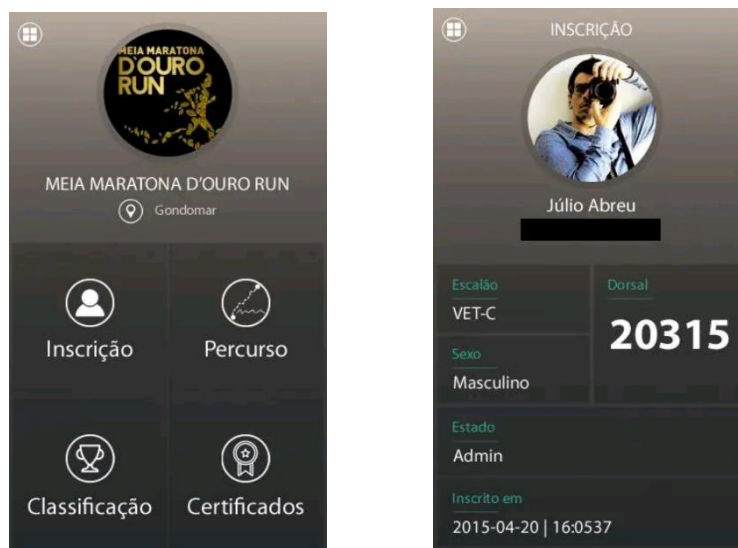


Figura 10 - Aplicação Eventsport

Running Lisboa

Descrição Oficial

“Aplicativo oficial do Maratona Clube de Portugal em Lisboa, desenvolvido por MYLAPS.”

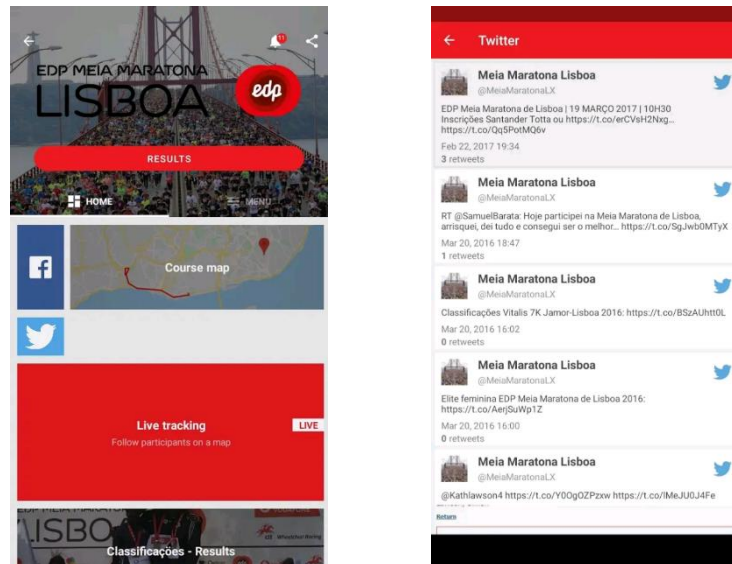


Figura 11 - Aplicação Running Lisboa

Pronto a Correr

Descrição Oficial

“Calendário de provas de estrada e de trail running.”

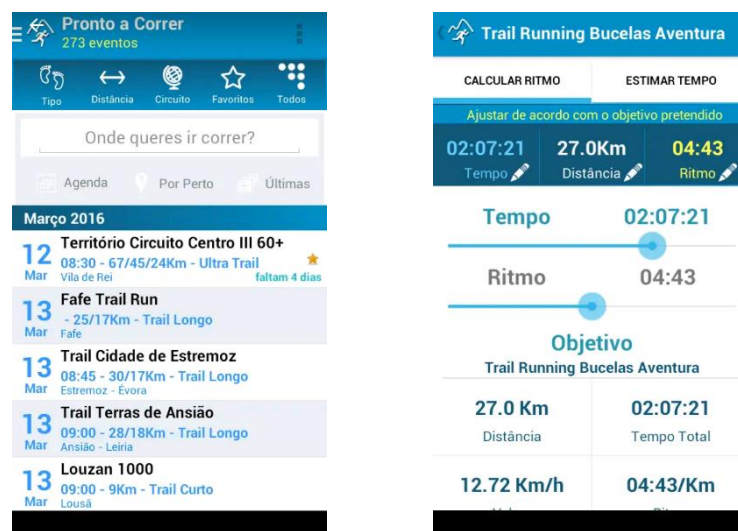


Figura 12 - Aplicação Pronto a Correr

Run Races

Descrição Oficial

“Encontre a sua próxima corrida em execução.”

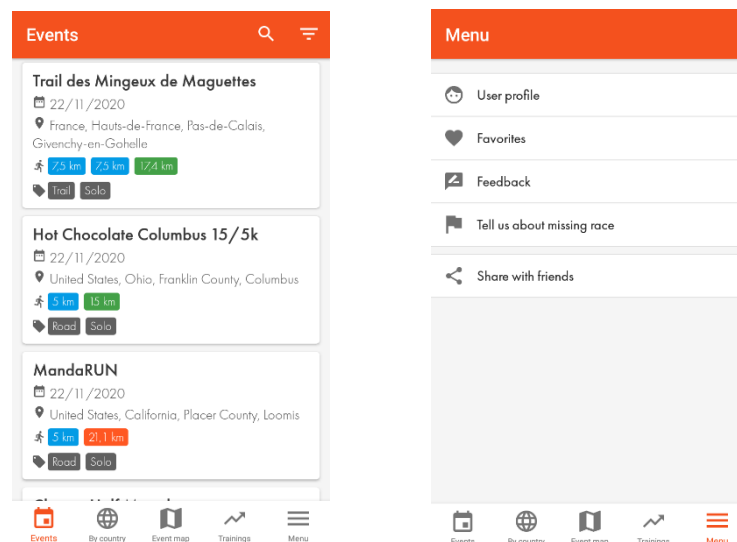


Figura 13 - Aplicação Run Races

RamRun

Descrição Oficial

“Calendário de eventos em execução (marchas, corridas, duatlo...) em todos os Estados Unidos.”

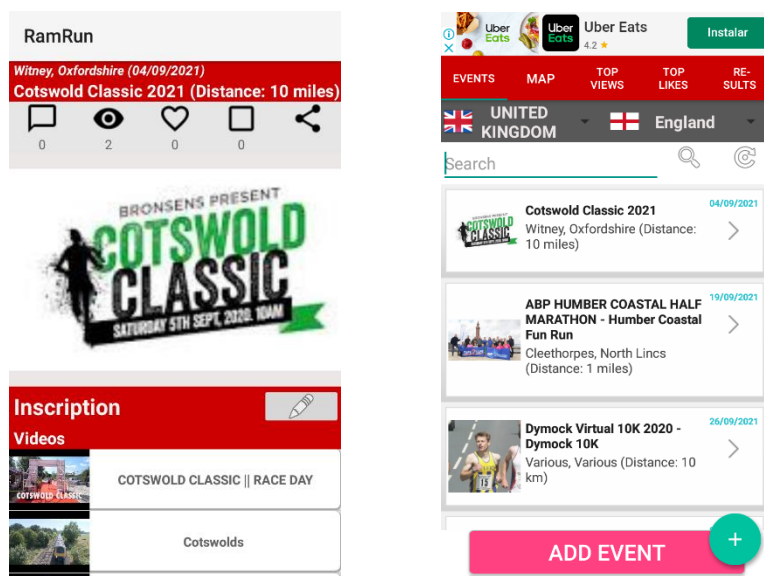


Figura 14 - Aplicação RamRun

Como referido, as aplicações acima definidas têm por base o mesmo propósito: os utilizadores inscrevem-se em eventos e os administradores realizarem a gestão dos mesmos. Estes sistemas concorrentes distinguem-se, nomeadamente, pelo elevado número de *downloads*, pela sua comunidade e utilizadores ativos e pelas excelentes classificações e *reviews*, que podem ser um indicador de popularidade e grau de satisfação, presentes em serviços de distribuição de aplicações móveis como o Google Play Store e/ou o serviço App Store da Apple.

De modo a proceder à comparação destas aplicações com a aplicação proposta a desenvolver foi construída a Tabela 2:

Tabela 2 - Comparação da aplicação a ser desenvolvida com aplicações concorrentes

Nome	Rating	Plataformas	Preço	Registar tempo (S/N)	Estatísticas (S/N)	Redes Sociais integradas (S/N)
Sistema de gestão de eventos ³	N/A	Android e iOS	Grátis	S	S	S
Eventsport	4,0	Android e iOS	Grátis	S	N	N
Running Lisboa	N/A	Android e iOS	Grátis	N	N	S
Pronto a Correr	4,5	Android	Grátis	S	S	N
Run Races	4,8	Android e iOS	Grátis	N	N	S
RamRun	3,8	Android	0,99€ - 3,09€	N	N	S

Por inspeção direta da Tabela 2, pode-se retirar que todas as aplicações móveis em objeto de análise apresentam algumas das características em estudo mas somente a solução desenvolvida inova neste sentido e incorpora todos os atributos considerados. É relevante mencionar que, tanto a Pronto a Correr como a RamRun só se encontram disponíveis para dispositivos Android. Com base na ideia explorada no capítulo anterior, esta característica é interpretada como uma desvantagem em relação às restantes aplicações que possuem caráter multiplataforma, sendo que, apesar do Android ter uma maior quota no mercado, o iOS representa uma parte bastante significativa dos utilizadores de *smartphones*.

Juntamente, constata-se que todas as aplicações com exceção da RamRun são grátis. Esta última difere-se no facto de ter anúncios que, ao pagar uma certa quantia, são omitidos.

³ Aplicação móvel desenvolvida no âmbito do presente TFC

6 Método e Planeamento

O método de trabalho adotado neste projeto teve como referência o calendário da Figura 15, proposto inicialmente.

	23/10/2020	27/10/2020	27/11/2020	27/12/2020	22/01/2021	22/02/2021	22/03/2021	23/04/2021	23/05/2021	25/06/2021
Instalação e configuração do Flutter no ambiente de trabalho										
Estudar a ferramenta Flutter e como aceder a web services a partir desta										
Levantamento e análise detalhada de requisitos										
Instalação e execução do sistema na sua forma web										
Implementação da aplicação móvel em código										
Implementação de testes de validação										
Implementação de um protótipo funcional										
Versão final do trabalho										

Figura 15 - Gráfico de Gant proposto inicialmente

O processo de trabalho decorreu de forma gradual e a sequência de tarefas foi executada de forma sistemática e, sempre que necessário, adaptada.

Em termos conceptuais, o modelo utilizado foi o desenvolvimento em cascata modificado [29], dado que o trabalho decorre num ambiente académico com um prazo de conclusão limitado e objetivos finais bem definidos.

Este modelo, ilustrado na Figura 16, possibilita o regresso a uma tarefa anterior ao ciclo em que se encontra, contrariando a rigidez do modelo em cascata tradicional ao contemplar alterações funcionais e/ou técnicas que possam surgir durante o projeto.

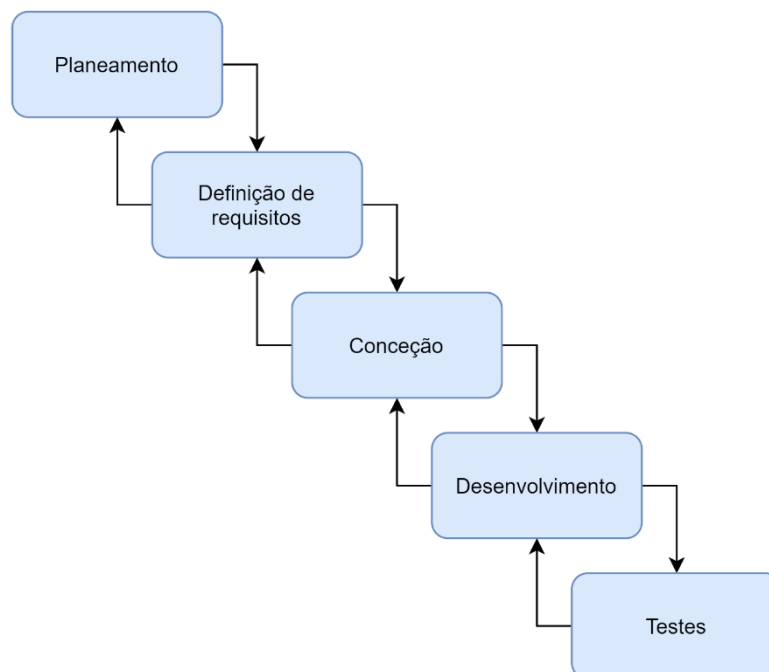


Figura 16 - Modelo em cascata modificado adaptado ao TFC

De um modo geral, o calendário proposto nos relatórios anteriores foi cumprido, ficando a faltar apenas a conclusão dos Testes de Usabilidade [30] efetuados por terceiros, derivada da escassez de tempo e da implementação incompleta de *web services*.

7 Resultados

Do culminar da fase de testagem, resultou um protótipo funcional capaz de responder às carências que condicionaram o surgimento desta proposta de TFC e que satisfaz os requisitos validados.

O protótipo começa com um *splash screen*, um ecrã inicial que introduz o utilizador à aplicação e responsável pela primeira perceção. A escolha do laranja como a cor predominante está interligada com o facto do logotipo da empresa ser composto por este tom.

Do *splash screen*, somos introduzidos ao ecrã de *login*, que irá direccionar o utilizador para o ecrã principal da aplicação se se verificarem as credenciais introduzidas nos devidos campos. Este processo é visível na Figura 17.

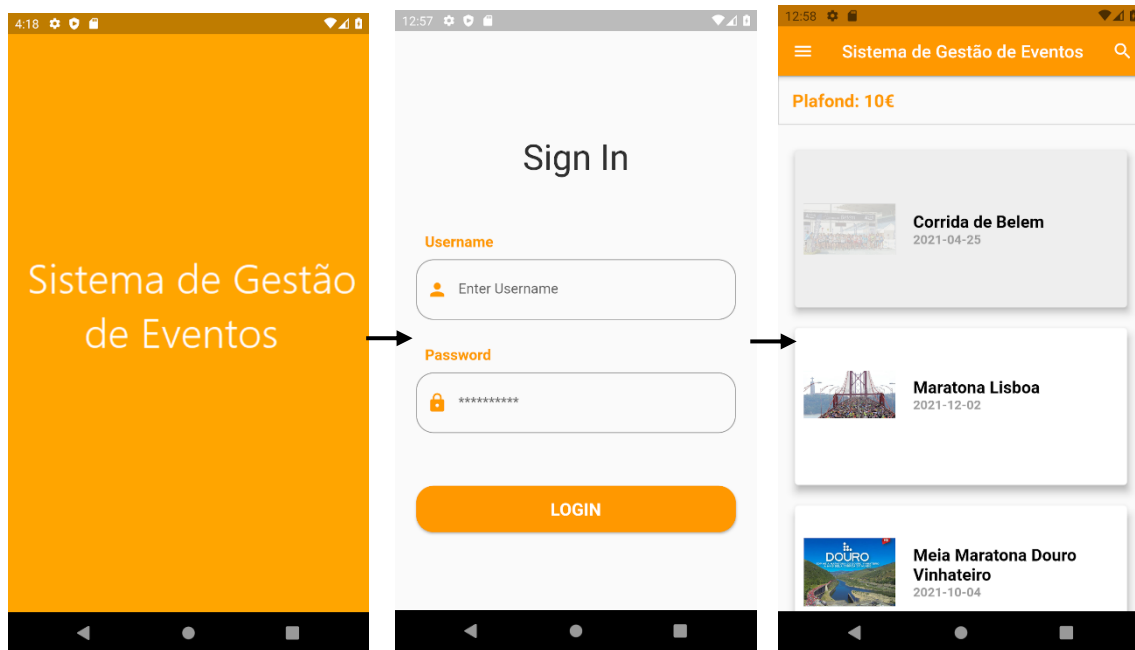


Figura 17 – Processo de login na aplicação móvel

O ecrã principal é comum aos dois tipos de utilizadores, com a particularidade de quando confrontados com o ícone do canto superior esquerdo, para o administrador é revelado o menu da Figura 18 e para o membro o menu da Figura 19.

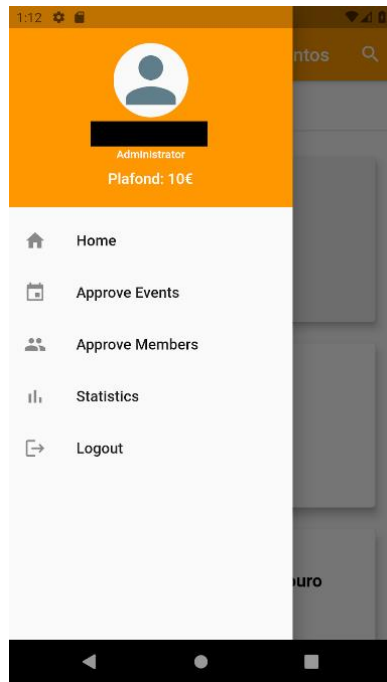


Figura 18 - Menu administrador

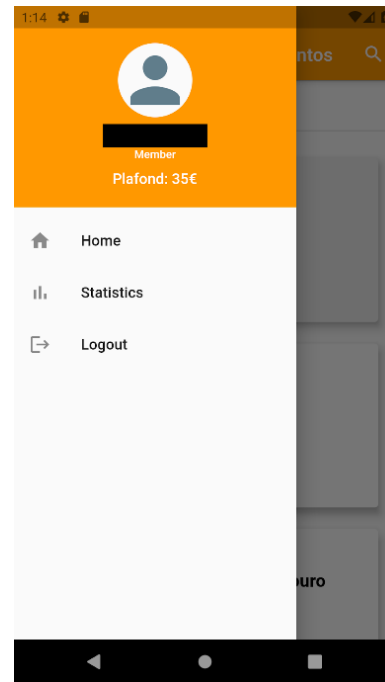


Figura 19 - Menu membro

Pelos *test cases* realizados, alterou-se o modo como os eventos cujo prazo já tinham terminado eram apresentados no ecrã principal, de forma a poder transmitir ao utilizador de que já não se poderia inscrever mais.

Ainda no ecrã principal, salienta-se que ambos os utilizadores, ao clicar no ícone presente no canto superior direito, têm disponível um campo de pesquisa que, ao introduzir o nome de um evento que desejem, o mesmo será apresentado, facilitando o processo de procura (Figura 20). É fundamental esclarecer que todas as provas desportivas que constituem o protótipo são de teor *dummy data*, isto é, dados fictícios, uma vez que com a atual crise pandémica não há organização de eventos desportivos na empresa.

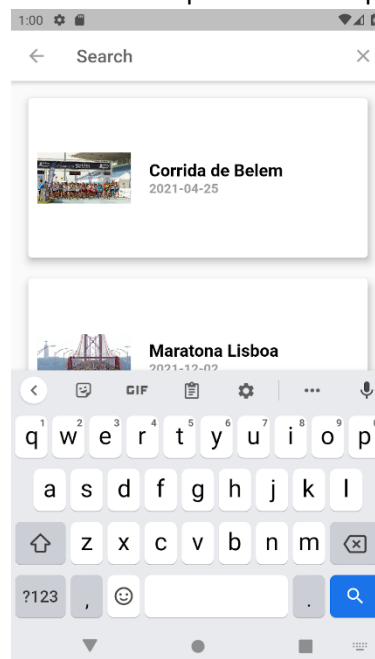


Figura 20 - Pesquisa de eventos

Ao selecionar um dado evento, neste caso, o segundo, os utilizadores serão encaminhados para o ecrã com todos os seus detalhes e podem proceder à sua inscrição.

Para se inscreverem, basta clicar no botão “Register” e o utilizador será registado na atividade desportiva se, e só se, o *plafond* for igual ou superior ao custo da mesma e o prazo de inscrição ainda não tiver terminado. Se estas condições se verificarem, o botão “Register” atualiza para um botão “Cancel” e o utilizador em causa pode cancelar a sua inscrição a qualquer momento, sendo novamente creditado o valor da prova. Este processo está ilustrado na Figura 21.

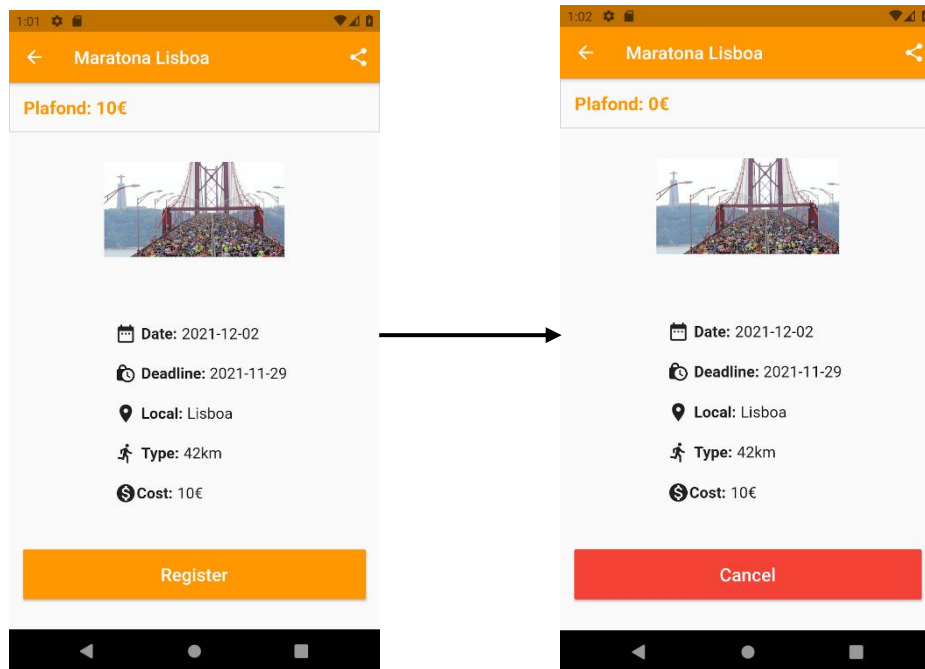


Figura 21 - Inscrição em evento

Comparativamente ao protótipo entregue na 2ª Avaliação Intercalar, foi alterada a cor do botão “Cancel” para vermelho, de maneira a poder enfatizar a natureza da operação.

Para um *plafond* inferior ao exigido pela atividade, surge a mensagem da Figura 22.

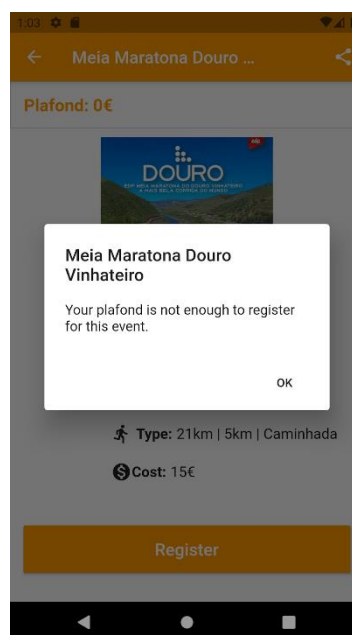


Figura 22 - Mensagem de erro quando o plafond é insuficiente

Adicionalmente, cada ecrã com o detalhe do evento oferece a opção de partilhar a prova nas redes sociais ao interagir com o ícone no canto superior direito.

Enquanto que os ecrãs acima são comuns aos dois tipos de utilizador, a Figura 23 e a Figura 24 incorpora as componentes de aprovação ou rejeição de eventos e membros, respetivamente, exclusivas do administrador.

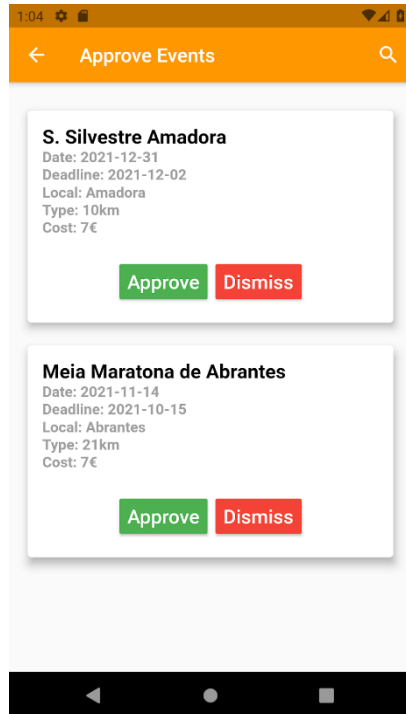


Figura 23 - Lista de eventos pendentes de aprovação

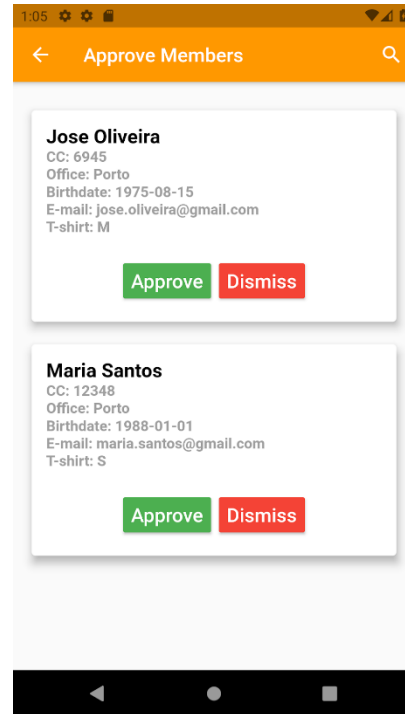


Figura 24 - Lista de membros pendentes de aprovação

O administrador tem o poder de aprovar o evento/membro ao clicar no botão “Approve”, de rejeitar ao clicar no botão “Dismiss” e de efetuar a pesquisa por nome. Quando um evento é aprovado, o mesmo é retirado desta lista e posteriormente adicionado ao *home* juntamente com os outros eventos. Para o membro, o processo é semelhante e este terá acesso à aplicação. Como se tratam de ações sensíveis, a aplicação pergunta sempre se o administrador tem a certeza que pretende proceder (Figura 25).

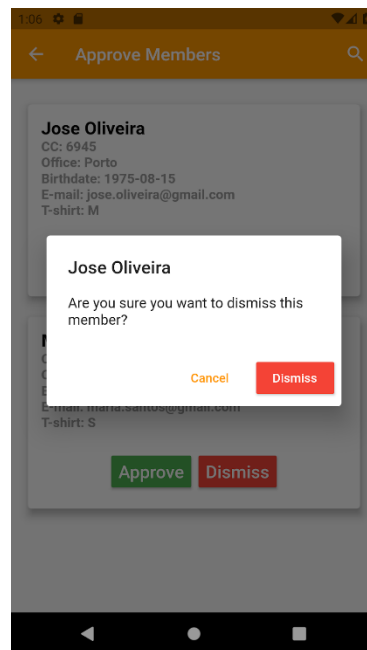


Figura 25 - Mensagem de rejeição de membro

Por fim, tem-se o ecrã cujo seu teor baseia-se numa relação simbiótica entre o tempo introduzido e o gráfico produzido - as estatísticas.

Neste ecrã, os utilizadores podem consultar um gráfico onde o eixo do x representa a data em que foi realizada a prova e o eixo do y o tempo, em horas, que demorou. Logo a seguir, surgem quatro *cards*, correspondentes ao tempo mais rápido, mais lento e o total de tempo e quilómetros percorridos. Estes dois últimos aspetos foram adicionados à posteriori. Por omissão, todos os valores inicialmente derivam do conjunto global de tempos inseridos.

Se o utilizador pretender filtrar os resultados, basta clicar no ícone no canto superior direito e selecionar o filtro que tenciona.

Estão disponíveis três filtros: distância percorrida (em km), tipo de atividade (corrida ou caminhada) e intervalo de datas. A título de exemplo, quando aplicados o filtro de distância de 10km e tipo de atividade corrida obtém-se os resultados da Figura 26.

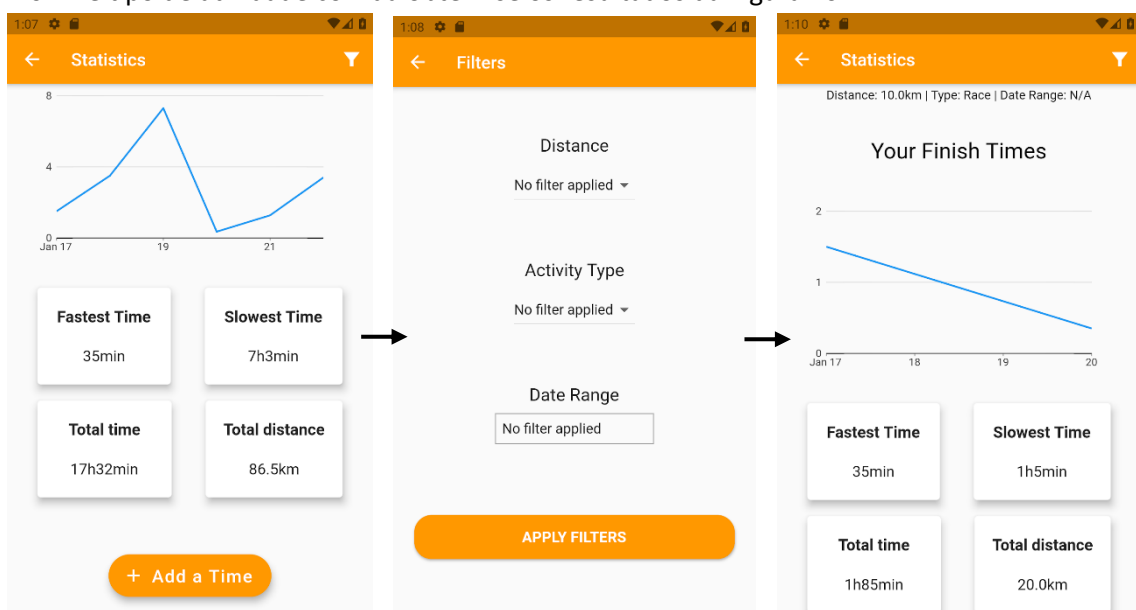
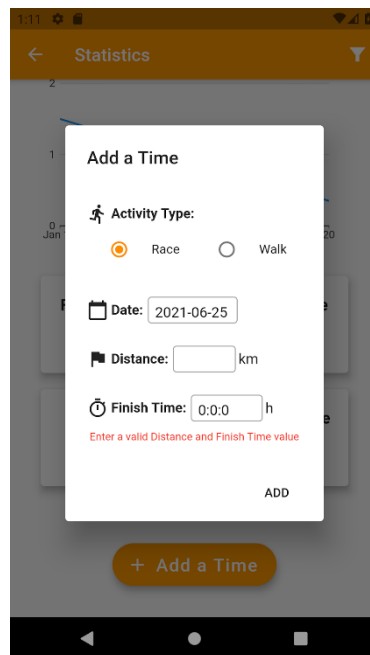


Figura 26 - Processo de filtragem de estatísticas

Todas estas variáveis podem ser inseridas quando o botão “Add Time”, que se encontra no fim do ecrã das estatísticas, é pressionado. Para um tempo ser registado com sucesso, a distância percorrida e o tempo de conclusão têm de ser valores positivos e não nulos (Figura 27).



The image shows a mobile application interface with a 'Statistics' screen in the background. A white dialog box titled 'Add a Time' is overlaid on the screen. The dialog contains the following elements: a running person icon followed by 'Activity Type:' with two radio buttons, 'Race' (selected) and 'Walk'; a calendar icon followed by 'Date:' and a text field containing '2021-06-25'; a flag icon followed by 'Distance:' and a text field, with 'km' to the right; a clock icon followed by 'Finish Time:' and a text field containing '0:0:0', with 'h' to the right. Below these fields is a red error message: 'Enter a valid Distance and Finish Time value'. At the bottom right of the dialog is an 'ADD' button. Below the dialog, on the 'Statistics' screen, is a brown button with a plus sign and the text '+ Add a Time'. The bottom of the screen shows the Android navigation bar.

Figura 27 - Registo de tempos

8 Conclusão e Trabalhos Futuros

A aplicação móvel Sistema de Gestão de Eventos foi concebida de forma a responder, de um modo geral, a todos os objetivos propostos, tornando o processo de gestão de eventos automatizado em dispositivos móveis, como uma alternativa cómoda e inovadora.

O TFC contribui para o contacto de novas tecnologias e aprofundamento de conhecimentos adquiridos durante o percurso académico. Toda a dinâmica envolvente neste processo foi bastante gratificante e traduziu-se numa mais valia na conclusão do curso de Engenharia Informática.

Verificaram-se algumas limitações de cariz académico que foram sentidas na iminência do cumprimento de prazos de períodos de avaliação.

Neste âmbito, o TFC deve ser encarado como uma “porta de entrada” para a multiplicidade de funcionalidades que a aplicação móvel pode vir a ter, no sentido de ser adotada na empresa alvo.

Futuramente, é idealizado um conjunto de sugestões que passam pela utilização de ícones que indiquem se o tempo das provas desportivas já foi ultrapassado, pela remoção de provas que já tenham acontecido do ecrã principal, de um método que permita mudar com mais facilidade as cores e a introdução de uma imagem no *splash screen*. Seria, igualmente, interessante ter uma funcionalidade dedicada ao histórico de eventos passados.

Bibliografia

- [1] Free online meeting scheduling tool | Doodle. [Online]. Available: <https://doodle.com/en/>. [Acesso: 22-10-2020].
- [2] What is Agile Software Development?, *Visual Paradigm* [Online]. Available: <https://www.visual-paradigm.com/scrum/what-is-agile-software-development/>. [Acesso: 25-01-2021].
- [3] What is User Story?, *Visual Paradigm*. [Online]. Available: <https://www.visual-paradigm.com/guide/agile-software-development/what-is-user-story/>. [Acesso: 01-02-2021].
- [4] FAQ, *Flutter*. [Online]. Available: <https://flutter.dev/docs/resources/faq#what-devices-and-os-versions-does-flutter-run-on>. [Acesso: 24-10-2020].
- [5] J. Nielsen and R. Budi, *Mobile Usability*. New Riders Press, 2012.
- [6] Flutter - Beautiful native apps in record time. [Online]. Available: <https://flutter.dev/>. [Acesso: 24-10-2020].
- [7] Statcounter. (2020, Setembro). *Mobile Operating System Market Share Worldwide*. [Online]. Available: <https://gs.statcounter.com/os-market-share/mobile/worldwide>. [Acesso: 08-11-2020].
- [8] Dart programming language | Dart. [Online]. Available: <https://dart.dev/>. [Acesso: 24-10-2020].
- [9] Skia in Flutter & Fuchsia. [Online]. Available: <https://skia.org/dev/flutter>. [Acesso: 24-10-2020].
- [10] JSON and serialization. [Online]. Available: <https://flutter.dev/docs/development/data-and-backend/json>. [Acesso: 24-10-2020].
- [11] Welcome to Common-Lisp.net!. [Online]. Available: <https://common-lisp.net/>. [Acesso: 14-11-2020].
- [12] The most popular database for modern apps | MongoDB. [Online]. Available: <https://www.mongodb.com/>. [Acesso: 14-11-2020].
- [13] GNU Emacs - GNU Project, *Gnu.org*. [online] Available: <https://www.gnu.org/software/emacs/> [Acesso: 1-06-2021].
- [14] About - Steel Bank Common Lisp, *Sbcl.org*. [online] Available: <http://sbcl.org/> [Acesso: 1-06-2021].
- [15] joaotavora/sly, *Github*. [online] Available: <https://github.com/joaotavora/sly> [Acesso: 1-06-2021].
- [16] Quicklisp beta, *Quicklisp.org*. [online] Available: <https://www.quicklisp.org/beta/> [Acesso: 1-06-2021].
- [17] http | Dart Package, *Dart packages*. [Online]. Available: <https://pub.dev/packages/http>. [Acesso: 1-06-2021].
- [18] Axure RP 9 - Prototypes, Specifications, and Diagrams in One Tool, *Axure*. [Online]. Available: <https://www.axure.com/>. [Acesso: 19-03-2021].
- [19] LayoutBuilder class - widgets library - Dart API, *Api.flutter.dev*. [Online]. Available: <https://api.flutter.dev/flutter/widgets/LayoutBuilder-class.html>. [Acesso: 01-04-2021].
- [20] of method - MediaQuery class - widgets library - Dart API, *Api.flutter.dev*. [Online]. Available: <https://api.flutter.dev/flutter/widgets/MediaQuery/of.html>. [Acesso: 01-04-2021].
- [21] setState method - State class - widgets library - Dart API, *Api.flutter.dev*. [Online]. Available: <https://api.flutter.dev/flutter/widgets/State/setState.html>. [Acesso: 01-04-2021].
- [22] Bloc State Management Library, *Bloclibrary.dev*. [Online]. Available: <https://bloclibrary.dev/#/>. [Acesso: 01-04-2021].
- [23] provider | Flutter Package, *Dart packages*. [Online]. Available: <https://pub.dev/packages/provider>. [Acesso: 01-04-2021].
- [24] Admeus. (2016). *Eventsport* (Version 0.0.2) [Mobile App]. Available: <https://play.google.com/store/apps/details?id=pt.eventsport.app2&hl=pt&gl=US>

- [25] MYLAPS Experience Lab. (2020). *Running Lisboa* (Version 1.0) [Mobile App]. Available: https://play.google.com/store/apps/details?id=com.mylaps.eventapp.runninglisboa&hl=en_IN
- [26] Outside Dreamz. (2014). *Pronto a Correr* (Version 1.5.3) [Mobile App]. Available: https://play.google.com/store/apps/details?id=com.outsidedream.myrun&hl=pt_PT
- [27] Mobilife. (2017). *Run Races* (Version 1.11.0) [Mobile App]. Available: https://play.google.com/store/apps/details?id=org.runraces&hl=pt_PT&gl=US
- [28] RamSoft. (2019). *RamRun* (Version 1.35) [Mobile App]. Available: https://play.google.com/store/apps/details?id=com.events.running&hl=pt_BR
- [29] Misenar, S., Feldman, J. and Conrad, E., 2011. *Eleventh Hour CISSP: Study Guide (Syngress eleventh hour series)*. Syngress, pp.129-145.
- [30] K. Moran, "Usability Testing 101", *Nielsen Norman Group*, 2019. [Online]. Available: <https://www.nngroup.com/articles/usability-testing-101/>.

Anexo 1 – Tabela de Requisitos Funcionais

ID	User Story	Como	Consigo	De Forma a	CrITÉrios de Aceitação
1	Login	membro e administrador	fazer <i>login</i>	poder usar as funcionalidades da aplicação	<ol style="list-style-type: none"> 1. Consigo ver um formulário com os campos para introdução do nome de utilizador e palavra-passe 2. A palavra-passe nunca será visível 3. Ao introduzir dados válidos corresponde ntes a um utilizador preexistente na Base de Dados, sou redirecionado para o ecrã principal da aplicação
2	Logout	membro e administrador	fazer <i>logout</i>	poder terminar a minha sessão na aplicação	<ol style="list-style-type: none"> 1. Saio da minha sessão e sou redirecionado para a o ecrã de <i>login</i>
3	Listar Eventos	membro e administrador	ter acesso à lista de provas disponíveis	poder inscrever-me nas que me interessam	<ol style="list-style-type: none"> 1. Consigo ver uma lista com as provas a se realizar em data futura 2. Ao clicar numa prova, posso aceder

					aos detalhes completos da mesma
4	Inscrição em Evento	membro e administrador	inscrever-me numa prova	poder participar no evento	<ol style="list-style-type: none"> 1. Consigo ver todos os detalhes respeitantes à prova (nome, tipo de atividade, data a realizar-se, data limite de inscrição, localização e custo) 2. Para que a inscrição seja aceite, o custo do evento tem de ser inferior ou igual ao valor disponível no meu <i>plafond</i> 3. Se o prazo de inscrição já tiver sido ultrapassado, o registo no evento não é válido
5	Aprovar/ Rejeitar Membros	administrador	aceder à lista de membros	poder aprovar ou rejeitar novos utilizadores que se tenham registado no sistema	<ol style="list-style-type: none"> 1. Consigo ver uma lista de novos utilizadores que ainda não foram aprovados 2. Tenho a capacidade de aprovar ou rejeitar o membro através dos

					respetivos botões
6	Aprovar/ Rejeitar Eventos	administrador	aceder à lista de eventos	poder aprovar ou rejeitar novos eventos que tenham sido criados no sistema	<ol style="list-style-type: none"> 1. Consigo ver uma lista de novos eventos que ainda não foram aprovados 2. Tenho a capacidade de aprovar ou rejeitar o evento através dos respetivos botões
7	Pesquisar Evento	membro e administrador	pesquisar uma prova	poder encontrá-la mais facilmente	<ol style="list-style-type: none"> 1. No campo da pesquisa da lista de eventos, ao introduzir o nome da prova, obtenho o evento correspondente
8	Pesquisar Membro Pendente de Aprovação	administrador	pesquisar um membro pendente de aprovação	poder encontrá-lo mais facilmente	<ol style="list-style-type: none"> 1. No campo da pesquisa da lista de membros pendentes de aprovação, ao introduzir o nome do utilizador, obtenho o membro correspondente

9	Pesquisar Evento Pendente de Aprovação	administrador	pesquisar uma prova pendente de aprovação	poder encontrá-la mais facilmente	1. No campo da pesquisa da lista de eventos pendentes de aprovação, ao introduzir o nome da prova, obtenho o evento correspondente
10	Consultar Plafond	membro e administrador	ver o meu <i>plafond</i>	ter conhecimento do valor que tenho disponível para gastar	1. O <i>plafond</i> é visível ao longo da aplicação 2. Sempre que me inscrevo numa prova, o custo desta é debitado no <i>plafond</i> e o valor é atualizado
11	Partilhar nas Redes Sociais	membro e administrador	partilhar nas minhas redes sociais os eventos que irei atender	poder efetuar publicações para os meus seguidores	1. Consigo ver um ícone que, ao clicar, irá me facultar uma caixa de texto <i>default</i> 2. Submeto a publicação nas redes sociais através de um botão

12	Consultar Estatísticas	membro e administrador	consultar as minhas estatísticas	ser capaz de observar a minha evolução em todas as provas que participei	<ol style="list-style-type: none"> 1. Consigo ver a prova que concluí mais rapidamente 2. Ver a prova que concluí mais lentamente 3. Ver o total de distância percorrida 4. Ver o total de tempos 5. Ter acesso ao gráfico linear da variável tempo, em horas, e a data de conclusão da prova
13	Adicionar Tempo de Conclusão	membro e administrador	adicionar o tempo em que terminei uma prova	poder observar a evolução estatística de todas as provas	<ol style="list-style-type: none"> 1. O formulário deve validar que apenas tempos positivos e não nulos sejam permitidos
14	Filtrar Tempo de Conclusão	membro e administrador	filtrar os tempos em que terminei uma prova	poder observar a evolução estatística de todas as provas segundo um dado parâmetro	<ol style="list-style-type: none"> 1. Consigo filtrar os resultados por distância percorrida, pelo tipo de prova (corrida ou caminhada) e/ou intervalo de datas

15	Avaliar Evento	membro e administrador	atribuir uma pontuação de 0 a 5	poder avaliar um evento em que participei	1. Apenas consigo avaliar um evento após a data indicada para sua realização
----	-----------------------	------------------------	---------------------------------	---	--

Anexo 2 - Implementação do método GET

```
List<User> parseUsers(String responseBody) {  
    final parsed = jsonDecode(responseBody).cast<Map<String, dynamic>>();  
  
    return parsed.map<User>((json) => User.fromJson(json)).toList();  
}  
  
Future<List<User>> fetchUsers() async {  
    final response = await http.get(Uri.parse('${serverAdress}users/'));  
  
    if (response.statusCode == 200) {  
        return parseUsers(response.body);  
    } else {  
        throw Exception('Failed to load users');  
    }  
}
```

Anexo 3 - Implementação do método POST

```
Future<User> updateUser(int id, bool approvedP) async {  
  final response = await http.post(Uri.parse('${serverAdress}update-  
user?id=$id&approved=$approvedP'));  
  
  if (response.statusCode == 200) {  
    return User.fromJson(json.decode(response.body));  
  } else {  
    throw Exception('Failed to update user');  
  }  
}
```

Anexo 4 - Implementação do LayoutBuilder() e MediaQuery.of()

```
body: new LayoutBuilder(  
  builder: (BuildContext context, BoxConstraints  
viewportConstraints){  
    return SingleChildScrollView(  
      child: ConstrainedBox(  
        constraints: BoxConstraints(  
          minHeight: viewportConstraints.maxHeight,  
        ),  
      ),  
    ),  
  );  
);
```

```
TextSpan(  
  text: '\n${event.date}',  
  style: TextStyle(  
    color: Colors.grey,  
    fontSize: (15/720)* MediaQuery.of(context).size.height  
  ),  
),
```

Anexo 5 - Implementação do Provider

```
class Plafond with ChangeNotifier{

  int _plafond = 0;

  int get value => _plafond;

  setPlafond(int value) {
    _plafond = value;
    notifyListeners();
  }

  void incrementPlafond(int value) {
    _plafond += value;
    notifyListeners();
  }

  void decrementPlafond(int value){
    _plafond -= value;
    notifyListeners();
  }
}
```

```
child: Consumer<Plafond>(
  builder: (context, plafond, child) => Text(
    'Plafond: ${plafond.value}€',
    style: TextStyle(fontSize: 20.0, color: Colors.orange, fontWeight:
    FontWeight.bold),
  ),
),
```


Glossário

LEI	Licenciatura em Engenharia Informática
TFC	Trabalho Final de Curso
PME	Pequena e Média Empresa
UI	Interface de Utilizador
SO	Sistema Operativo
UI/UX	Interface de Utilizador e Experiência de Utilizador
JSON	JavaScript Object Notation
REST	Representational State Transfer
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Environment
SBCL	Steel Bank Common Lisp
REPL	Read-Eval-Print-Loop
API	Interface de Programação de Aplicações
BLoC	Business Logic of Component