



UNIVERSIDADE
LUSÓFONA

DEISI34

Ethical Fitness

Trabalho Final de curso

Relatório Final

Nome do Aluno: Beatriz Sampaio

Nome do Orientador: Prof. Rui Santos

Trabalho Final de Curso | LIG | Data 28/06/2023

www.ulusofona.pt

Direitos de cópia

Ethical Fitness, Copyright de *Beatriz Sampaio*, Universidade Lusófona.

A Escola de Comunicação, Arquitetura, Artes e Tecnologias da Informação (ECATI) e a Universidade Lusófona (UL) 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 objetivos educacionais ou de investigação, não comerciais, desde que seja dado crédito ao autor e editor.

Resumo

Este relatório apresenta o desenvolvimento do projeto Ethical Fitness, uma aplicação móvel destinada a clientes e *Personal Trainers* para facilitar a gestão de treinos. O projeto incluiu a implementação de funcionalidades essenciais como agendamento de treinos, publicação de *posts* e criação de planos de treino personalizados. Durante o desenvolvimento, foram introduzidas notificações em tempo real, melhorias na interface e métodos de pagamento para eventos e treinos organizados pelo estúdio.

A reestruturação da aplicação existente foi um desafio significativo, exigindo uma taxa de esforço superior à inicialmente prevista. Isso incluiu a separação clara entre a interface de utilizador e a lógica de negócios, além da otimização da base de dados para melhorar a eficiência e escalabilidade. Testes rigorosos foram conduzidos para validar a funcionalidade e estabilidade da aplicação, assegurando uma experiência do utilizador fluida e confiável.

O resultado é uma aplicação mais intuitiva e eficiente, que não apenas melhora a experiência dos clientes ao agendar e participar de treinos, mas também simplifica a gestão administrativa dos *Personal Trainers*. Este relatório detalha os passos do desenvolvimento, os desafios enfrentados e os resultados obtidos, destacando a importância das melhorias implementadas para o sucesso do projeto Ethical Fitness.

Abstract

This report outlines the development of the Ethical Fitness project, a mobile application designed for clients and Personal Trainers to streamline workout management. The project encompassed implementing essential features such as scheduling workouts, posting updates, and creating personalized training plans. Throughout development, real-time notifications, interface enhancements, and payment methods for studio events and workouts were introduced.

Restructuring the existing application posed a significant challenge, requiring a higher-than-anticipated level of effort. This involved clearly separating the user interface from business logic and optimizing the database for improved efficiency and scalability. Rigorous testing was conducted to validate the application's functionality and stability, ensuring a seamless user experience.

The result is a more intuitive and efficient application that not only enhances client experience in scheduling and participating in workouts but also simplifies administrative tasks for Personal Trainers. This report details the development steps, challenges faced, and outcomes achieved, emphasizing the importance of these enhancements for the success of the Ethical Fitness project.

Índice

Resumo.....	iii
Abstract	iv
Índice	v
Lista de Figuras	vii
Lista de Tabelas	viii
1 Identificação do Problema	9
2 Viabilidade e Pertinência.....	11
3 Benchmarking.....	13
4 Modelo Proposto	16
4.1 Levantamento e análise dos Requisitos.....	16
4.2 Diagramas de Casos de Uso	19
4.3 Diagramas de Atividade (BPMN)	21
4.4 Modelos Relevantes (Diagrama de Classes - UML).....	22
4.5 Estrutura	24
4.6 Mockups.....	25
5 Solução Desenvolvida.....	27
5.1 Introdução	27
5.2 Arquitetura	27
5.2.1 Desenho da arquitetura da solução	27
5.2.2 Componentes da arquitetura	27
5.2.3 Padrões arquiteturais e de software	28
5.3 Tecnologias e Ferramentas Utilizadas	29
5.3.1 Tecnologias Utilizadas	29
5.3.2 Perímetros Tecnológicos	29
5.3.3 Ferramentas Utilizadas.....	30
5.4 Implementação	30
5.5 Abrangência	32
6 Plano de testes e validação	33
6.1 Guia Detalhado de Testes	33
6.1.1 Testes de Notificações.....	33

6.1.2	Teste de Método de Pagamento	34
6.1.3	Teste de Detalhes do Evento	35
6.2	Resultados dos Testes Planeados	36
6.2.1	Resultados dos Testes de Notificações	36
6.2.2	Resultados dos Testes de Método de Pagamento	37
6.2.3	Resultados dos Testes de Detalhes do Evento	38
7	Método e Planeamento	40
8	Resultados	43
8.1	Resultados Obtidos	43
8.1.1	Notificações	43
8.1.2	Método de Pagamento.....	43
8.1.3	Detalhes do Evento	43
8.2	Outputs Gerados.....	44
8.3	Outcomes Alcançados.....	44
8.4	Cumprimento dos Critérios de Sucesso	44
8.5	Revisões e Melhorias Introduzidas	44
9	Conclusão e trabalhos futuros	48
9.1	Conclusão.....	48
9.2	Trabalhos Futuros	48
	Bibliografia	50
	Anexo 1 – Project Charter	51
	Anexo 2 – Vídeo Youtube e GitHub.....	52
	Anexo 3 – Excertos de código	53
	Glossário.....	58

Lista de Figuras

Figura 1 – Ecrãs GoFit	13
Figura 2 – Ecrãs Solinca	14
Figura 3 - Ecrãs FitOn	14
Figura 4 - Caso de Uso das Notificações	20
Figura 5 - Caso de Uso do Método de Pagamento	21
Figura 6 - BPMN do Método de Pagamento	22
Figura 7 - Diagrama de Classes (UML)	23
Figura 8 - Novo Diagrama de Classes (UML)	23
Figura 9 - Diagrama Aplicacional	24
Figura 10 - Novo Diagrama Aplicacional	25
Figura 11 - Mockups aplicação	25
Figura 12 - UI final dos eventos e agenda	26
Figura 13 - Arquitetura	27
Figura 14 - Arquitetura BLoC	29
Figura 15 - Perímetros Tecnológicos	30
Figura 16 - Calendário de Execução Previsto	41
Figura 17 - Calendário Final	42
Figura 18 - Antiga estrutura da aplicação Ethical Fitness	45
Figura 19 - Nova estrutura da aplicação Ethical Fitness	46
Figura 20 - Antigo ecrã de inserção de aulas de grupo	47
Figura 21 - Novo ecrã de inserção de aulas de grupo	47
Figura 22 - Código de busca de eventos para calendário	53
Figura 23 - Código para inserção de novas aulas pt	54
Figura 24 - Botão de marcar e desmarcar aulas	55
Figura 25 - Lógica do botão de marcar e desmarcar aulas	55
Figura 26 - Botão pagamento evento e emissão para API	56
Figura 27 - Cloud Functions código de nova aula de grupo	56
Figura 28 - Pedido de permissões de notificações	57

Lista de Tabelas

Tabela 1 - Funcionalidades das aplicações	15
Tabela 2 - Requisitos da aplicação	18
Tabela 3 - Novos requisitos	19

1 Identificação do Problema

Neste capítulo, pretende-se descrever o enquadramento prático e a envolvente do problema em análise, por meio de uma formulação detalhada do *case study* abordado no Trabalho Final de Curso (TFC). O problema em estudo resulta de circunstâncias reais observadas na aplicação móvel, destinada para a empresa Xstudio, desenvolvida inicialmente em 2022/2023 por um grupo de alunos, no âmbito do TFC. A aplicação, na sua versão inicial, incluía funcionalidades de agendamento de aulas, divulgação de publicações e desmarcação de aulas. No entanto, foram identificadas áreas de melhoria significativas que necessitavam de atenção.

A proposta de desenvolvimento para o ano letivo atual incluía a implementação de notificações em tempo real, com o objetivo de aumentar a interação dos utilizadores com a aplicação, tanto para os clientes quanto para os *personal trainers*. Além disso, foi sugerido um método de pagamento para eventos organizados pela administração, visando simplificar o processo de gestão financeira que, até então, era realizado manualmente. Ainda no decorrer do projeto constatou-se uma carência de atribuição de *personal trainers* aos clientes, sendo este apenas possível para os administradores. Também foi pedido a possibilidade de ter uma perspetiva semanal da agenda, visto que só tinha mensal. Estas melhorias iriam não só agilizar os processos administrativos, como também oferecer uma experiência de utilizador mais dinâmica e envolvente.

Durante o desenvolvimento, foi constatado que a aplicação carecia de dinamismo e apresentava más práticas de programação. Assim, foi necessário refazer o que já estava implementado, seguindo as boas práticas de desenvolvimento de *software*, e adicionar as novas funcionalidades propostas para este ano: notificações em tempo real, métodos de pagamento e melhorias no *design* da aplicação.

Os problemas encontrados na versão inicial da aplicação foram numerosos e variados, contribuindo para um atraso significativo no desenvolvimento do projeto, uma vez que foi necessário refazer muitos componentes já existentes, o que resultou numa carga de trabalho superior ao esperado. A má estrutura do projeto era evidente, com uma mistura inadequada de interface de utilizador (UI) com lógica de negócios, classes com atributos mal definidos, e uma organização geral deficiente do projeto. A base de dados estava mal estruturada e não havia um tema coerente para o projeto, incluindo uma paleta de cores e tipos de letra.

Nos ecrãs, vários problemas foram identificados. Na disponibilidade de aulas com *personal trainers* (antigo *availability*), o *personal trainer* tinha de adicionar manualmente cada aula diariamente. Se tivesse 8 aulas, teria de adicionar uma a uma, o que demonstrava a falta de dinamismo. Nas aulas de grupo (antiga *classes*), estas deveriam permitir mais do que um cliente, mas a classe e a base de dados estavam configuradas para apenas um cliente, impossibilitando o registo de vários participantes. Nas aulas com *personal trainers* e de grupo, a data e hora eram inseridas como *textField*. A data era guardada na base de dados tanto formatada como *String* quanto em *DateTime*, gerando redundância. Os alunos eram adicionados às aulas pelo nome, o que causava problemas se houvesse mais de um utilizador com o mesmo nome. Para indicar se a aula tinha participantes, utilizava-se um atributo *bool* na classe. Além disso, era necessário um botão para guardar alterações quando os alunos se inscreviam, não havia possibilidade de editar uma aula após a sua criação, apenas eliminá-la. As aulas eram recuperadas indiscriminadamente para todos os utilizadores, incluindo aquelas subscritas por outros utilizadores, permitindo alterações indevidas.

Na agenda, as aulas do *personal trainer* eram guardadas apenas quando marcadas, mas não podiam ser eliminadas, e mesmo após desmarcação, continuavam lá. No ecrã de novidades/inicial, apenas era possível colocar imagens, sem texto ou qualquer organização adequada.

A falta de um tema consistente para a aplicação, incluindo uma paleta de cores e tipos de letra, contribuiu para uma experiência de utilizador inconsistente e visualmente desorganizada. Estes problemas estruturais e funcionais não só afetaram a usabilidade da aplicação, como também impactaram negativamente a eficiência dos processos tanto para os clientes quanto para os *personal trainers*. Consequentemente, a necessidade de corrigir estas falhas resultou em um atraso significativo no cronograma do projeto, aumentando a carga de trabalho além do previsto.

Este relatório final expõe o âmbito dos resultados obtidos e realiza uma análise comparativa destes face ao inicialmente proposto, justificando eventuais diferenças entre proposta e resultados. A análise visa demonstrar de forma clara que a solução desenvolvida representa um avanço significativo na resolução dos problemas identificados na aplicação original, oferecendo uma ferramenta mais robusta, eficiente e agradável para os utilizadores. Além disso, destaca-se a importância de seguir boas práticas de desenvolvimento e *design* desde o início, para evitar retrabalho e garantir a entrega de um produto de alta qualidade.

2 Viabilidade e Pertinência

Na era em constante evolução da tecnologia, a necessidade de adaptar e aprimorar as infraestruturas digitais torna-se crucial para o sucesso organizacional. No contexto específico do desenvolvimento da aplicação móvel em Flutter, com Firebase como base de dados, a proposta de implementação de notificações em tempo real, métodos de pagamento e melhorias no *design surge* como uma abordagem proativa na modernização e otimização do sistema.

A aplicação utilizou diversas ferramentas do Firebase, incluindo Firebase Firestore, Firebase Authentication, Firebase Cloud Messaging e Firebase Cloud Functions, além da integração com a API do Stripe para processamento de pagamentos. Esta escolha tecnológica foi fundamental para garantir uma infraestrutura escalável, segura e eficiente, permitindo a implementação das novas funcionalidades de forma coesa e integrada.

A viabilidade da solução desenvolvida é evidente. A escolha do Flutter como framework de desenvolvimento assegurou uma experiência de utilizador consistente e um desempenho elevado em várias plataformas. O uso do Firebase como *backend* permitiu uma integração perfeita com serviços de autenticação, armazenamento de dados em tempo real, notificações e execução de funções na nuvem, tudo de forma segura e eficiente. A integração com a API do Stripe simplificou o processamento de pagamentos, tornando o sistema mais robusto e funcional.

A pertinência do projeto é muito elevada. As melhorias introduzidas, como as notificações em tempo real e o método de pagamento, atenderam diretamente às necessidades identificadas no uso inicial da aplicação. Adicionalmente, a reestruturação do projeto, a definição de um tema visual consistente e a reorganização da base de dados contribuíram significativamente para a estabilidade, usabilidade e manutenção do sistema a longo prazo.

Durante o desenvolvimento, foram encontrados vários problemas na aplicação anterior, que necessitaram de reformulação. A estrutura do projeto inicial era inadequada, com uma mistura de interface de utilizador (UI) com lógica de negócios, classes com atributos mal definidos e uma organização deficiente do projeto. A base de dados estava mal estruturada e faltava um tema consistente, como uma paleta de cores e fontes de letra. Estes problemas foram resolvidos através de uma reestruturação completa da aplicação, melhorando a organização do código e da base de dados, e definindo um tema visual coerente.

Nos ecrãs, vários problemas foram identificados e reformulados. A disponibilidade de aulas com *personal trainers* (antigo *availability*) exigia que o *personal trainer* adicionasse manualmente cada aula diariamente, o que foi otimizado para permitir uma gestão mais dinâmica. As aulas de grupo (antiga *classes*) estavam configuradas para apenas um cliente, o que foi alterado para permitir múltiplos participantes. Nas aulas com *personal trainers* e de grupo, a data e hora eram inseridas como *textField*, e os alunos eram adicionados às aulas pelo nome, o que causava problemas de duplicação de nomes. Estas funcionalidades foram reformuladas para utilizar *inputs* apropriados e evitar redundâncias na base de dados.

Outros problemas resolvidos incluíram a necessidade de um botão para guardar alterações nas inscrições de aulas, a incapacidade de editar uma aula após a criação e a recuperação indiscriminada de todas as aulas para todos os utilizadores. A agenda foi ajustada para permitir a eliminação de aulas e para que desmarcações fossem corretamente refletidas.

O ecrã de novidades/inicial, que só permitia a colocação de imagens sem texto, foi reformulada para suportar melhor organização e conteúdo mais rico.

A nova funcionalidade de escolha de *personal trainers* pelos administradores e a visão semanal da agenda foram adicionadas para melhorar a gestão e usabilidade da aplicação. Estas alterações não comprometeram a viabilidade nem a pertinência do TFC. Pelo contrário, ampliaram o escopo da aplicação, fornecendo uma solução mais completa e alinhada com as necessidades dos utilizadores finais.

Os testes de validação confirmaram a eficácia das novas funcionalidades e a estabilidade do sistema reestruturado. A aplicação demonstrou um desempenho confiável em notificações em tempo real, processamento de pagamentos e usabilidade geral. A visão semanal da agenda e a capacidade de os administradores escolherem *personal trainers* melhoraram significativamente a gestão e o agendamento de aulas, simplificando os processos administrativos e melhorando a experiência do utilizador.

3 Benchmarking

Para a realização desta análise, foram consultadas as seguintes aplicações:

- GoFit - <https://apps.apple.com/pt/app/my-go-fit/id786133717>
- Solinca - <https://apps.apple.com/pt/app/solinca/id1510608997>
- FitOn - <https://apps.apple.com/us/app/fiton-workouts-fitness-plans/id1442473191>

A. GoFit

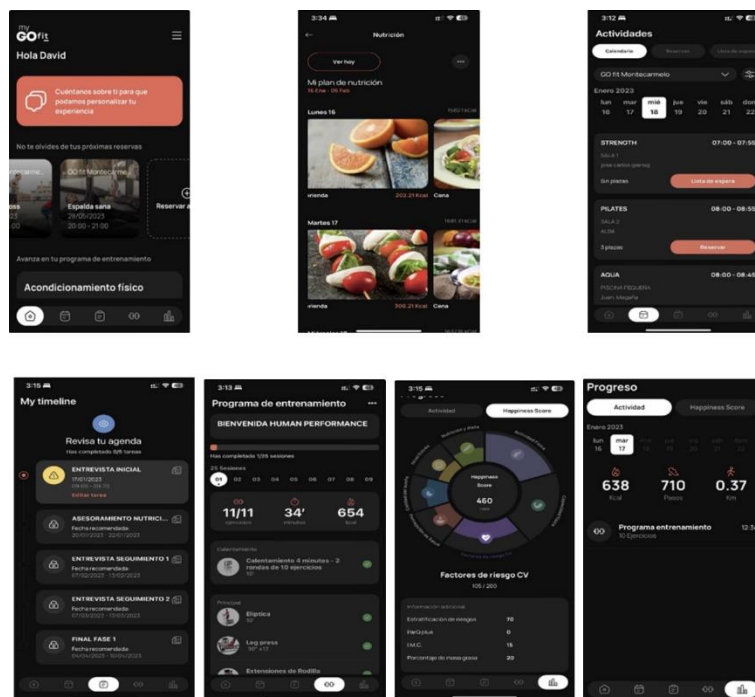


Figura 1 – Ecrãs GoFit

A APP do GoFit só está acessível para os membros do ginásio, sendo gratuita e oferecendo funcionalidades como a gestão de treinos em grupo, acesso a aulas específicas, receitas, agenda e estatísticas. A aplicação recebe críticas positivas em relação ao seu desempenho.

B. Solinca



Figura 2 – Ecrãs Solinca

A APP do Solinca só está acessível para os membros do ginásio, sendo gratuita e oferecendo funcionalidades como a gestão de treinos em grupo, acesso a aulas específicas e contacto com a equipa.

No entanto, a aplicação tem recebido críticas negativas em relação ao seu desempenho, indicando a presença de muitos *bugs*. Estas críticas foram reportadas no período entre 2020 e 2023.

C. FitOn

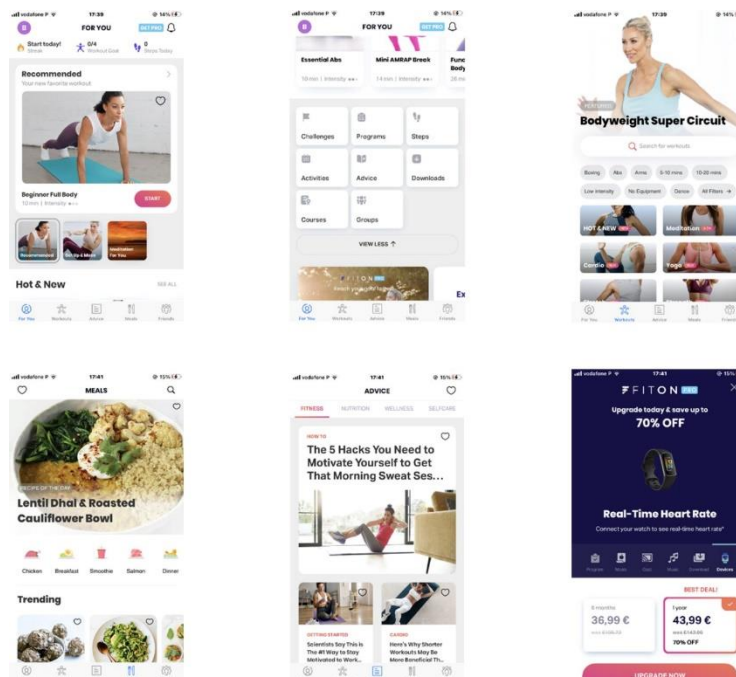


Figura 3 - Ecrãs FitOn

A APP do FitOn está disponível para qualquer pessoa, sendo gratuita e oferecendo funcionalidades como vídeos de treinos, acesso a cursos específicos, contacto com amigos, dicas e acesso a receitas mediante a subscrição do pacote semestral ou anual. A aplicação tem recebido críticas positivas em relação ao seu desempenho.

D. Ethical Fitness

A APP do Ethical Fitness está restrita a membros do estúdio, sendo gratuita e apresenta funcionalidades como vídeos de treinos, plano de treino especializado, agendamento de treinos e a capacidade de efetuar pagamentos para eventos organizados pelo estúdio.

A aplicação EF destaca-se das outras aplicações de fitness por oferecer uma variedade superior de funcionalidades para auxiliar os utilizadores. Na tabela 1, tem uma comparação entre as aplicações em estudo, nesta verifica-se a EF tem mais funcionalidades relevantes ao funcionamento, oferecendo assim uma maior diversidade, mas carece na criação de treinos personalizados.

Funcionalidade	GoFit	Solinca	FitOn	Ethical Fitness
Marcação de treinos	✓	✓	✓	✓
Desmarcação de treinos	✓	✓	✓	✓
Consulta de Agenda	✓	✓	✓	✓
Consulta de treinos personalizados	✓	✓	✓	✓
Marcação de treinos com PT	✗	✗	✗	✓
Marcar eventos	✗	✗	✗	✓
Criar <i>posts</i>	✗	✗	✗	✓
Pagamentos	✗	✗	✓	✓
Criação de treinos personalizados	✗	✗	✗	✗

Tabela 1 - Funcionalidades das aplicações

4 Modelo Proposto

Este segmento do relatório foca na proposta do modelo desenvolvido para a solução abordada no projeto. Vai para além dos requisitos implementados durante o desenvolvimento do Trabalho Final de Curso (TFC), incluindo também requisitos não implementados e possíveis cenários de continuidade do projeto em contextos académicos ou empresariais.




O objetivo deste capítulo é fornecer uma análise abrangente dos requisitos propostos, indicando se foram totalmente, parcialmente implementados ou não implementados. Além disso, serão discutidas as justificações para modificações, exclusões ou adições de requisitos, conforme aplicável.


Um aspeto fundamental desta secção é a avaliação do valor obtido, considerando a importância atribuída a cada requisito durante as fases intermédias do projeto. Esta avaliação não apenas oferece uma perspetiva retrospectiva do desenvolvimento, mas também orienta futuras iterações e melhorias na solução.

A estrutura deste capítulo segue um formato organizado, abrangendo desde o levantamento inicial dos requisitos até a apresentação dos modelos finais. Cada secção detalha a evolução dos requisitos ao longo do ciclo de desenvolvimento, destacando as contribuições significativas do modelo proposto para a solução final apresentada neste trabalho.

4.1 Levantamento e análise dos Requisitos

Na Tabela 2 encontram-se os requisitos iniciais a implementar no âmbito deste projeto. Estes requisitos foram delineados para garantir uma funcionalidade abrangente e eficiente do sistema proposto. Contudo, durante o desenvolvimento, deparei-me com várias dificuldades que impediram a implementação de alguns desses requisitos.

ID	Área	Requisito	Implementado
RF01	Notificações	Como utilizador quero receber notificações para estar atualizado com as informações relevantes da aplicação e alertado sobre eventos importantes.	
RF02	Notificações	Como utilizador quero ter a capacidade de desativar as notificações para controlar quando recebo notificações.	
RF03	Notificações	Como utilizador quero ser capaz de interagir com as notificações diretamente para abrir a aplicação ou	

		executar a ação associada à notificação com facilidade.	
RF04	Notificações	Como utilizador quero receber notificações mesmo quando a aplicação está em segundo plano para não perder informações importantes.	
RF05	Notificações	Como utilizador espero que as notificações sejam entregues em tempo real para garantir que recebo as informações a tempo.	
RNF01	Notificações	Como utilizador espero que as notificações não alterem o desempenho da aplicação, para ter uma boa experiência.	
RF06	Método de Pagamento	Como utilizador quero ser capaz de adicionar e gerir o método de pagamento, para facilitar as transações futuras e oferecer uma melhor experiência de <i>checkout</i> .	
RF07	Método de Pagamento	Como utilizador quero ter a possibilidade de guardar as informações do cartão de forma segura para facilitar futuras transações sem ter de voltar a introduzir as informações do cartão.	
RF08	Método de Pagamento	Como utilizador quero ter a opção de remover métodos de pagamentos anteriores, para manter a lista de métodos de pagamentos atualizados.	
RF09	Método de Pagamento	Como utilizador quero receber uma confirmação de transação bem-sucedida, para ter a certeza de que a transação foi concluída com sucesso.	
RF10	Método de Pagamento	Como utilizador quero que a aplicação suporte diferentes métodos de pagamentos para maior flexibilidade e satisfazer as preferências individuais dos utilizadores.	



RF11	Método de Pagamento	Como utilizador quero receber notificações sobre transações bem-sucedidas ou problemas de pagamentos para saber o estado das transações.	
RF12	Método de Pagamento	Como utilizador espero que as informações do cartão sejam armazenadas de forma segura para garantir a privacidade e segurança das informações do utilizador.	

Tabela 2 - Requisitos da aplicação

Primeiramente, não foi possível cumprir o requisito RF03, que consistia na interação com as notificações. Este requisito era essencial para melhorar a comunicação entre o sistema e os utilizadores, assegurando que estes fossem informados sobre atualizações importantes e eventos relevantes. A complexidade técnica envolvida e a necessidade de integração com sistemas de notificação externos ultrapassaram o escopo do projeto atual.

Em segundo lugar, o requisito RF07, que previa a capacidade de guardar os dados do método de pagamento para futuros pagamentos, também não foi implementado. Este requisito era crucial para simplificar o processo de pagamento para os utilizadores, permitindo uma experiência mais fluida e eficiente. No entanto, devido às restrições de segurança e à necessidade de conformidade com regulamentações de proteção de dados, a implementação revelou-se demasiado complexa dentro dos prazos estipulados.

Consequentemente, a não implementação do RF07 afetou diretamente o cumprimento do requisito RF08, que tratava da remoção dos dados de pagamento armazenados. Sem a capacidade de guardar os dados, a funcionalidade de removê-los tornou-se irrelevante e, portanto, foi descartada.

Por fim, o requisito RF10, que previa a possibilidade de adicionar mais do que um método de pagamento, também não pôde ser realizado. Esta funcionalidade visava proporcionar uma maior flexibilidade aos utilizadores, permitindo-lhes escolher entre diferentes opções de pagamento. Contudo, as mesmas limitações técnicas e de segurança que impediram a implementação do RF07 afetaram negativamente o RF10.

A principal razão para a não concretização destes requisitos foi a elevada taxa de esforço necessária para melhorar ou refazer a aplicação existente, desenvolvida pelo grupo do Trabalho Final de Curso do ano anterior (2022/2023). Como referido anteriormente na identificação do problema, o sistema existente apresentava uma arquitetura complexa e exigia uma reengenharia significativa para suportar as novas funcionalidades propostas.

Apesar destas limitações, durante o desenvolvimento identificou-se a necessidade de acrescentar novos requisitos para atender a pedidos específicos da administração da empresa Xstudio. Em particular, foram adicionadas duas funcionalidades: a possibilidade de administradores atribuírem um *personal trainer* aos alunos existentes e a visualização de uma

agenda semanal. Estes novos requisitos, detalhados na Tabela 3, foram implementados com sucesso, demonstrando a flexibilidade e a capacidade de adaptação do projeto às necessidades emergentes.



ID	Área	Requisito	Implementado
RF13	Gestão de Utilizadores	Como administrador, quero poder atribuir um <i>personal trainer</i> aos alunos existentes para melhorar o acompanhamento e suporte personalizado.	
RF14	Planeamento	Como utilizador, quero visualizar uma agenda semanal para gerir melhor as minhas atividades e compromissos.	

Tabela 3 - Novos requisitos

4.2 Diagramas de Casos de Uso

A Figura 4 ilustra um diagrama de casos de uso da aplicação "Ethical Fitness App", detalhando a interação entre diferentes atores e as funcionalidades principais relacionadas com as notificações e a gestão de eventos dentro do sistema.

Os atores envolvidos são o Cliente, o *Personal Trainer* (PT) e o Firebase. O Cliente representa o alunos do estúdio Xstudio, que recebe notificações sobre eventos relevantes. O Personal Trainer é o profissional que utiliza a aplicação para criar e atualizar eventos. O Firebase é um serviço externo utilizado para enviar notificações.

O diagrama apresenta os seguintes casos de uso: "Recebe notificação", "Envia notificação", "Cria evento" e "Atualiza evento". No caso de uso "Recebe notificação", o Cliente recebe notificações na aplicação sobre eventos criados ou atualizados pelo Personal Trainer. Este caso de uso inclui o caso de uso "Envia notificação".

O caso de uso "Envia notificação" descreve a funcionalidade onde a aplicação envia notificações aos Clientes. Estas notificações são acionadas pela criação ou atualização de eventos pelos *Personal Trainers* e são enviadas via Firebase, que é responsável pelo processamento e entrega das notificações.

Os casos de uso "Cria evento" e "Atualiza evento" descrevem as ações realizadas pelo Personal Trainer. O primeiro envolve a criação de novos eventos, como aulas ou sessões de treino, enquanto o segundo trata da atualização de eventos existentes, como alterar horários ou modificar detalhes de sessões. Ambas as ações desencadeiam o envio de notificações para os Clientes, mantendo-os informados sobre quaisquer mudanças ou novos eventos.

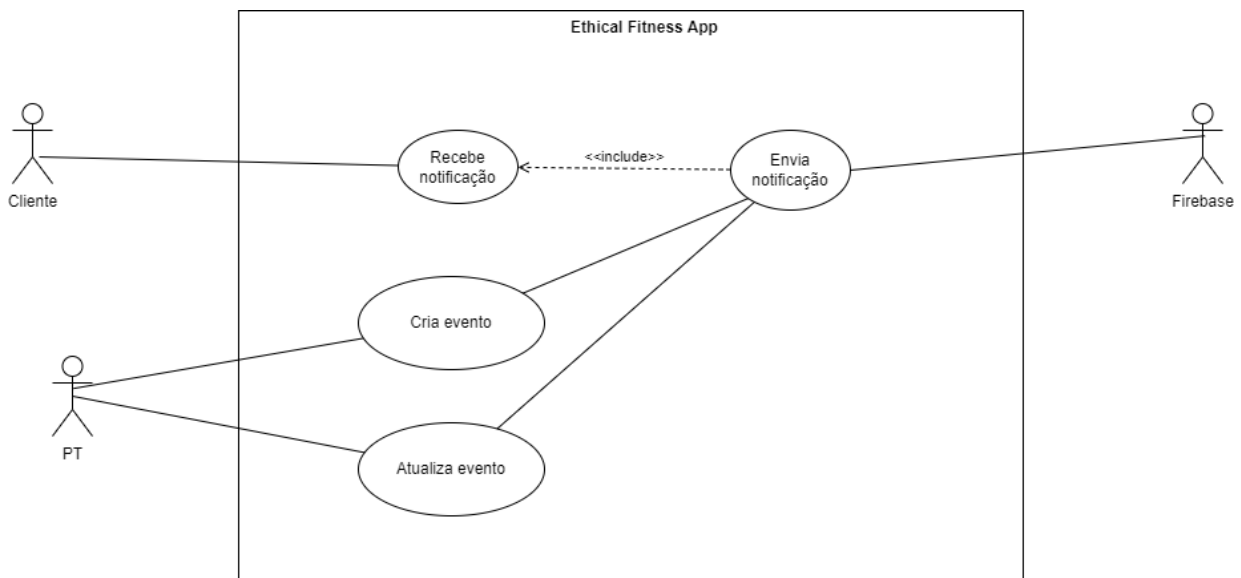


Figura 4 - Caso de Uso das Notificações

Em suma, a Figura 4 descreve um fluxo de comunicação eficiente entre *Personal Trainers* e Clientes, mediado por notificações automáticas enviadas através do serviço Firebase. Este diagrama demonstra como a aplicação "Ethical Fitness App" mantém os utilizadores informados sobre eventos relevantes, assegurando que as atualizações e novas informações sejam prontamente comunicadas.

A Figura 5 ilustra um diagrama de casos de uso da aplicação "Ethical Fitness App", detalhando a interação entre o Cliente, a aplicação e o serviço externo Stripe API no contexto das funcionalidades de pagamento.

O Cliente, representando os alunos do estúdio Xstudio, interage com o sistema para realizar pagamentos. O Stripe API é um serviço externo utilizado para processar e verificar os pagamentos.

O diagrama apresenta os seguintes casos de uso: "Inicializa pagamento", "Verifica pagamento" e "Confirma pagamento". No caso de uso "Inicializa pagamento", o Cliente começa o processo de pagamento na aplicação. Este caso de uso inclui o caso de uso "Verifica pagamento", que trata da validação e verificação dos detalhes do pagamento através do serviço Stripe API. A ligação direta entre "Inicializa pagamento" e "Verifica pagamento" indica que a verificação é uma parte essencial do processo de inicialização do pagamento.

O caso de uso "Verifica pagamento" é responsável por confirmar se os detalhes do pagamento são válidos e se a transação pode ser processada com sucesso. Este caso de uso inclui a interação com o Stripe API, que é a entidade responsável pela validação dos dados de pagamento fornecidos pelo Cliente.

O caso de uso "Confirma pagamento" está diretamente ligado ao caso de uso "Verifica pagamento" através de uma relação de extensão (`<<extend>>`). Isto significa que a confirmação do pagamento é uma extensão do processo de verificação, ocorrendo apenas quando a verificação é bem-sucedida. A confirmação do pagamento é uma extensão do processo de verificação, ocorrendo apenas quando a verificação é bem-sucedida. A confirmação do

pagamento só é efetuada após a verificação dos dados pelo Stripe API, assegurando que a transação é segura e válida.

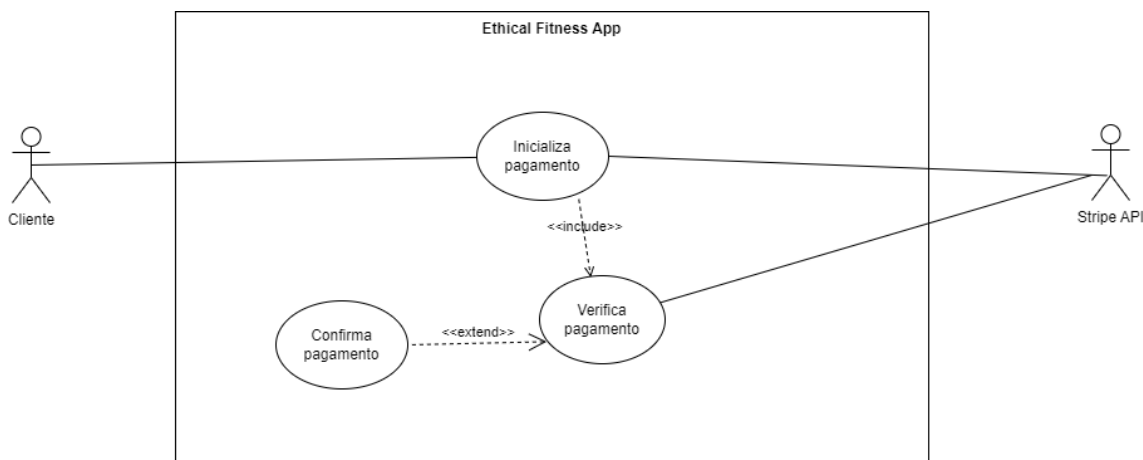


Figura 5 - Caso de Uso do Método de Pagamento

Em resumo, a Figura 5 demonstra como o "Ethical Fitness App" gere o processo de pagamento, desde a inicialização pelo Cliente, passando pela verificação dos detalhes do pagamento pelo Stripe API, até à confirmação final da transação. Este fluxo de operações garante que os pagamentos são processados de forma segura e eficiente, integrando-se perfeitamente com o serviço externo de pagamentos Stripe.

4.3 Diagramas de Atividade (BPMN)

A Figura 6 ilustra um diagrama BPMN (*Business Process Model and Notation*) do processo de pagamento na aplicação "Ethical Fitness App". Este diagrama detalha o fluxo de atividades desde a inicialização do pagamento pelo Cliente até à confirmação final da transação, envolvendo a interação com o serviço externo Stripe API.

O processo começa com o Cliente a iniciar um pagamento na aplicação. A primeira atividade no diagrama é a "Inicializa pagamento", onde o Cliente introduz os detalhes do pagamento, como o método de pagamento. Em seguida, o processo avança para a atividade "Envia dados para Stripe", onde os detalhes do pagamento são enviados para o Stripe API. O Stripe API é um serviço de pagamento externo responsável pela verificação e processamento dos dados recebidos.

O Stripe API realiza a atividade "Verifica pagamento". Nesta etapa, o Stripe API valida os dados do pagamento para assegurar que são corretos e que a transação pode ser processada. Se a verificação for bem-sucedida, o processo prossegue para a próxima etapa. Após a verificação bem-sucedida, o Stripe API envia uma resposta à aplicação, indicando o estado do pagamento. Esta atividade é representada no diagrama como "Recebe resposta do Stripe".

Com a resposta positiva do Stripe, o processo avança para a atividade "Confirma pagamento", onde a aplicação confirma ao Cliente que o pagamento foi efetuado com sucesso.

Se a verificação falhar, o processo desvia-se para uma atividade de tratamento de erro, onde é informado ao Cliente que o pagamento não pôde ser processado e são fornecidas instruções para corrigir o problema.

O diagrama BPMN destaca a sequência lógica das atividades, as decisões baseadas nas verificações e a interação com o serviço externo Stripe API, assegurando que o fluxo de pagamento seja conduzido de forma segura e eficiente.

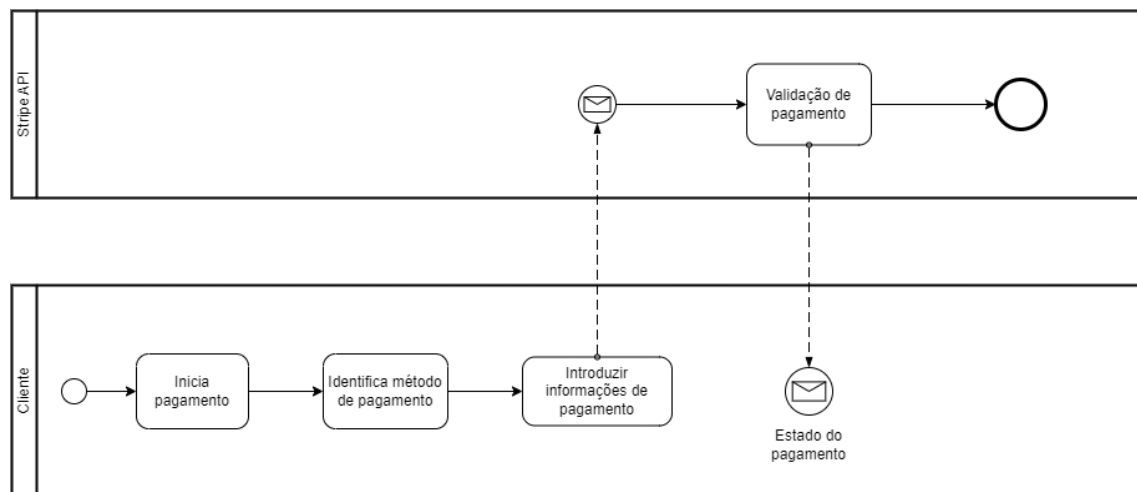


Figura 6 - BPMN do Método de Pagamento

Em suma, a Figura 6 descreve o processo de pagamento no "Ethical Fitness App", desde a inicialização pelo Cliente até à confirmação final, envolvendo a interação com o Stripe API para verificação e processamento. Este diagrama BPMN detalha as atividades e decisões críticas necessárias para assegurar que os pagamentos são geridos de forma segura e eficiente, proporcionando uma visão clara do fluxo de trabalho e das interações envolvidas.

4.4 Modelos Relevantes (Diagrama de Classes - UML)

A Figura 7 corresponde ao diagrama de classes inicial. Nesta figura, existia uma classe "Evento" que possuía atributos e dependia das classes "Pagamento" e "Aluno". No entanto, foram definidos campos indevidamente, faltando alguns campos necessários e atribuindo tipos que não correspondiam às necessidades da aplicação. Durante o desenvolvimento, percebi a necessidade de alterar este diagrama para acomodar melhor as necessidades da aplicação.

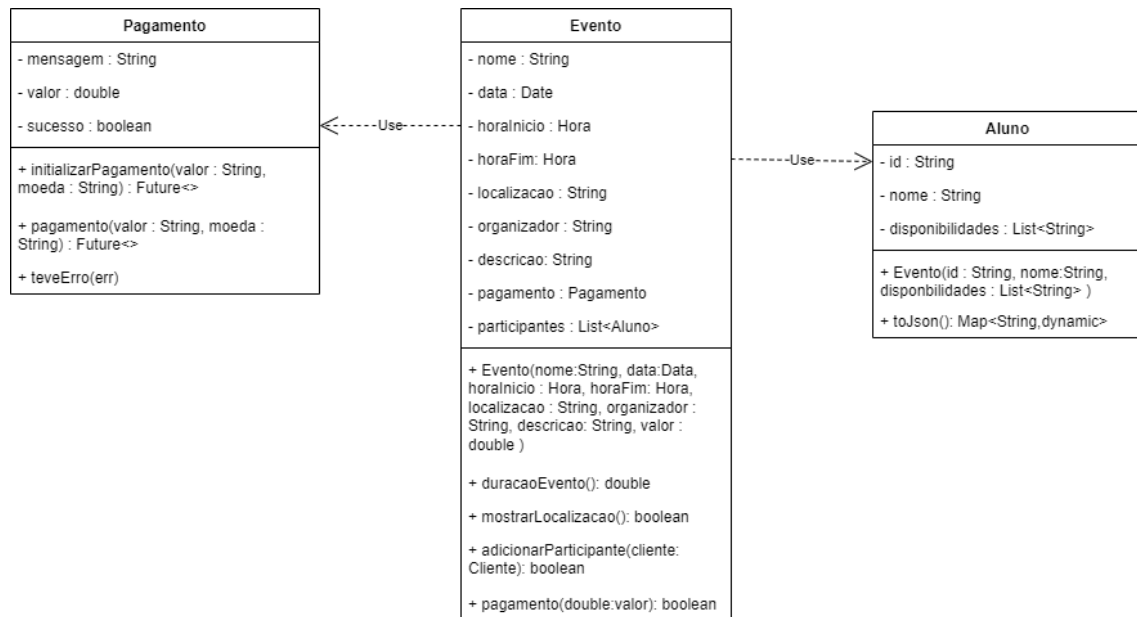


Figura 7 - Diagrama de Classes (UML)

A modificação resultou na Figura 8, onde simplifiquei a estrutura ao utilizar apenas a classe "Evento". Nesta nova versão, em vez de guardar o objeto completo do "Aluno", a classe "Evento" passou a armazenar apenas o "uid" do aluno, que corresponde ao identificador único do utilizador. Além disso, foram atribuídos novos atributos necessários à classe "Evento", garantindo que a estrutura de dados estivesse mais alinhada com os requisitos funcionais da aplicação.

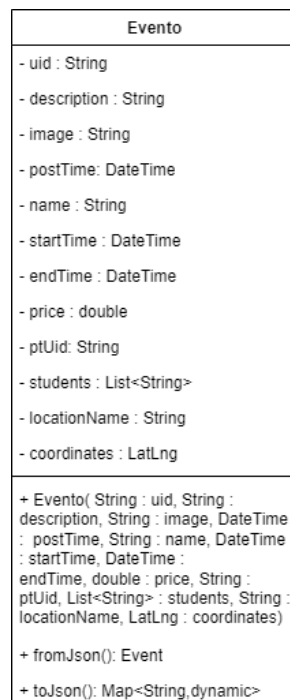


Figura 8 - Novo Diagrama de Classes (UML)

4.5 Estrutura

Ao longo do desenvolvimento do projeto, foi identificada a necessidade de melhorar o diagrama aplicacional originalmente concebido. A Figura 9 inicial representava as conexões entre os ecrãs principais do sistema: home, plano, aulas e agenda. A partir do ecrã inicial, os utilizadores podiam aceder aos detalhes de eventos específicos, enquanto no ecrã de agenda estavam disponíveis funcionalidades para criar e consultar eventos.

Durante a implementação dessas funcionalidades no trabalho final de curso anterior (2022/2023), constatou-se que a lógica dos ecrãs de criação e consulta de eventos não estava alinhada com o propósito do sistema. Era necessário rever estas funcionalidades para melhorar a sua usabilidade e relevância dentro do contexto da aplicação.

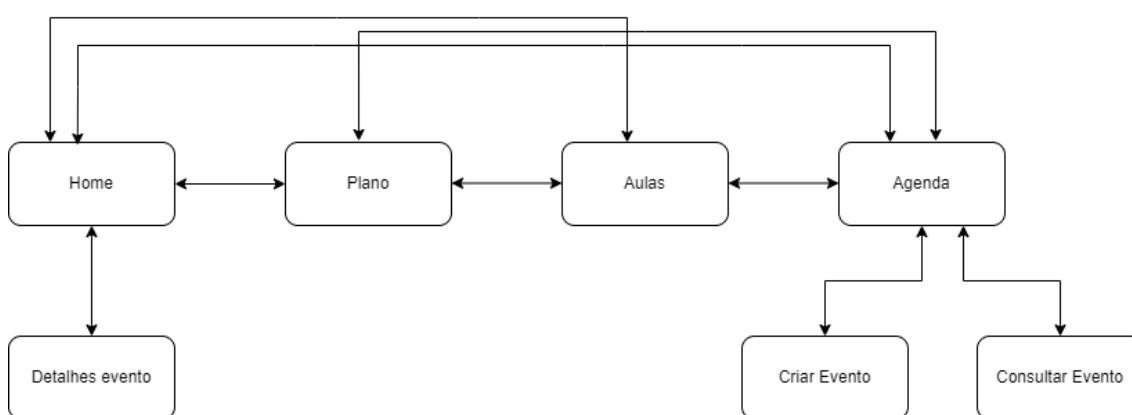


Figura 9 - Diagrama Aplicacional

Assim, na Figura 10 apresentada neste relatório, procedeu-se ao redesenho do diagrama aplicacional. As principais modificações incluem o refinamento da lógica dos ecrãs de criação e consulta de eventos, ajustando-as para melhor se alinharem com os objetivos do sistema. Também foram efetuados ajustes nas conexões entre os ecrãs principais (home, plano, aulas e agenda) para refletir as melhorias implementadas.

Estas alterações foram cruciais para melhorar a eficiência e eficácia do sistema, assegurando uma experiência do utilizador mais intuitiva e um alinhamento mais preciso com os requisitos do projeto.

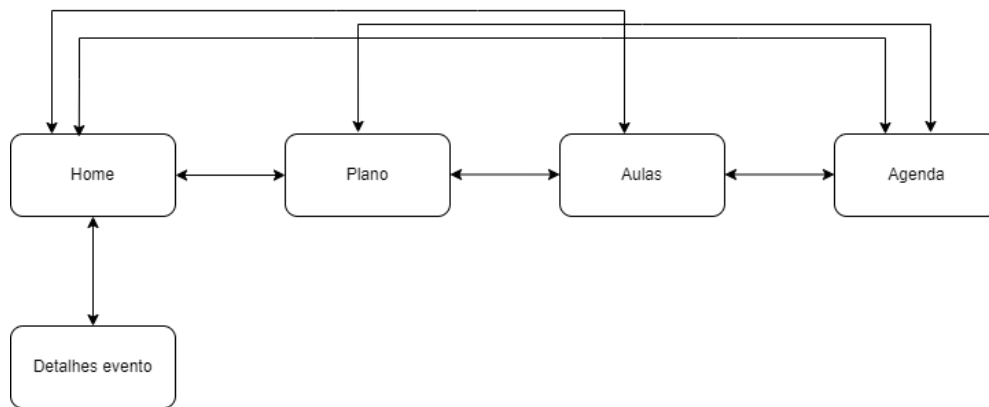


Figura 10 - Novo Diagrama Apicacional

4.6 Mockups

Na Figura 11, encontra-se o mockup inicial da aplicação, que inclui os ecrãs de eventos, detalhes de eventos, e as visualizações da agenda semanal e mensal. Estes mockups foram projetados para estabelecer a estrutura básica e a aparência inicial da aplicação, servindo como um guia para o desenvolvimento subsequente.

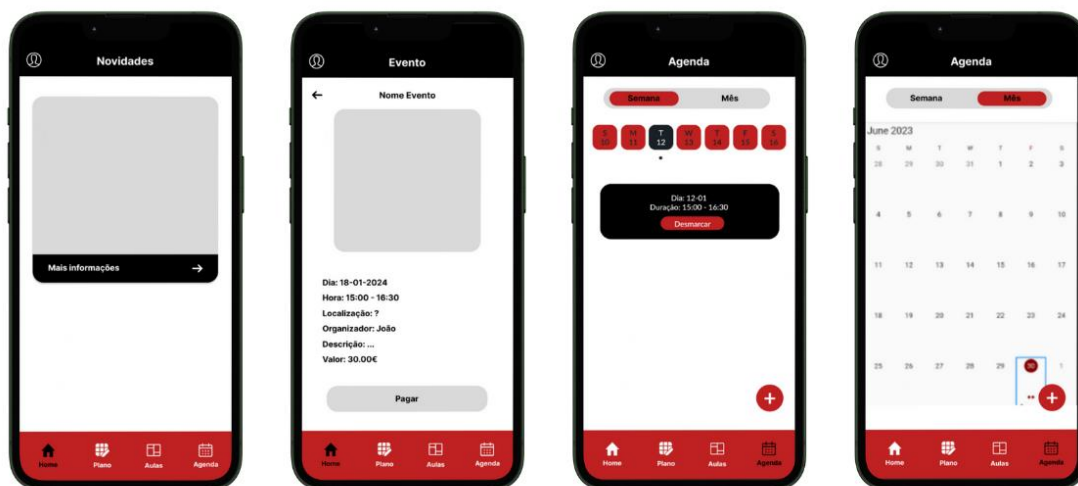


Figura 11 - Mockups aplicação

Na Figura 12, é apresentado o UI final desses ecrãs. Embora existam pequenas diferenças em relação ao mockup inicial, as alterações foram feitas para melhorar a usabilidade e a estética da aplicação. No entanto, o estilo visual geral foi mantido, garantindo uma experiência de utilizador consistente e coerente. Estas diferenças refletem o processo iterativo de design e desenvolvimento, onde o feedback e testes contínuos permitiram refinamentos e melhorias ao longo do tempo.

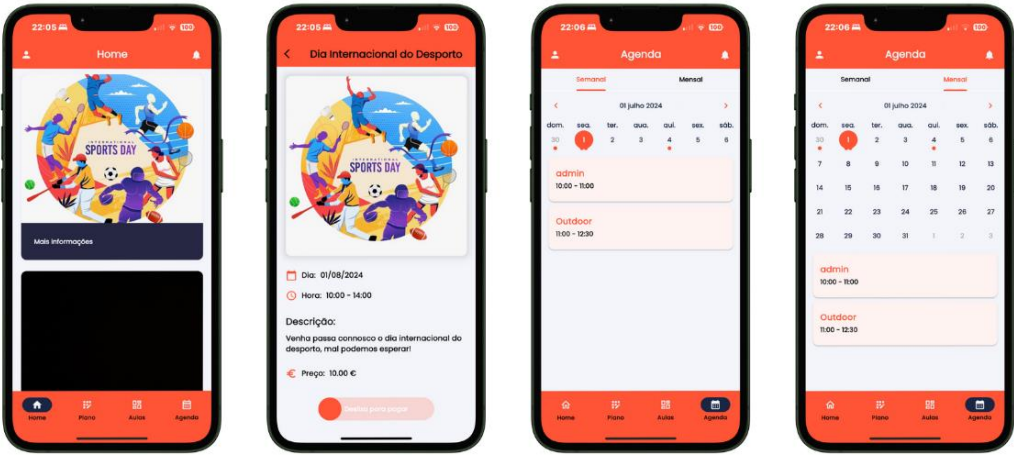


Figura 12 - UI final dos eventos e agenda

5 Solução Desenvolvida

5.1 Introdução

A aplicação "Ethical Fitness" é baseada numa arquitetura robusta e escalável desenvolvida com tecnologias modernas. Um vídeo demonstrativo detalha suas funcionalidades desenvolvidas, proporcionando uma visão clara do resultado final da aplicação. O código-fonte atualizado está disponível na branch do GitHub, permitindo verificar tudo o que foi desenvolvido. Detalhes sobre a instalação e acesso à versão funcional estão disponíveis no guião correspondente.

Esta secção fornece uma análise detalhada das principais escolhas técnicas adotadas no desenvolvimento da aplicação Ethical Fitness. Inclui a descrição completa da arquitetura implementada, destacando o uso do padrão BLoC para separação de lógica de negócios e interface do utilizador, integração com Firebase para gestão de dados e notificações, e Stripe API para processamento de pagamentos. A explicação das tecnologias selecionadas e ferramentas específicas utilizadas, com justificativas para suportar as funcionalidades exigidas pela aplicação, é complementada pelo detalhamento do progresso atual do desenvolvimento, com foco na integração da lógica do código para complementar o trabalho realizado na interface do utilizador. Durante este capítulo, serão apresentados excertos de código para ilustrar a evolução da implementação, comparando o código anterior com as atualizações realizadas. A discussão sobre as disciplinas e áreas científicas aplicadas na solução, juntamente com a análise das componentes consideradas mais inovadoras em comparação com o benchmarking e o conteúdo programático do curso, completa o capítulo. Este visa proporcionar uma compreensão abrangente da solução desenvolvida, sua implementação e os fundamentos teóricos que a sustentam, alinhando-se com os critérios de avaliação definidos para este trabalho final de curso.

5.2 Arquitetura

5.2.1 Desenho da arquitetura da solução

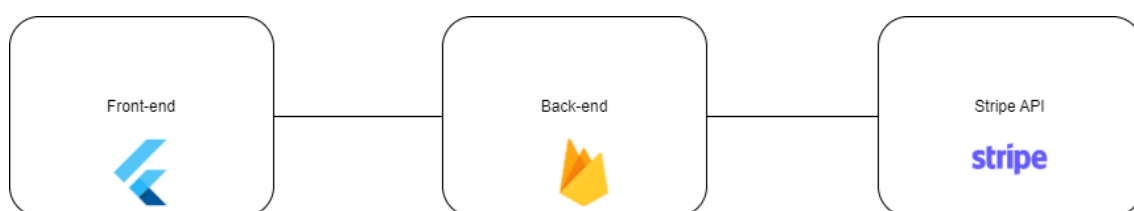


Figura 13 - Arquitetura

5.2.2 Componentes da arquitetura

Front-end: uma aplicação móvel que permite aos utilizadores agendar e cancelar treinos personalizados e aulas de grupo, inscrever-se em eventos e consultar a sua agenda.

Back-end: um serviço BaaS (*Business as a Service*) central responsável pelo processamento de dados. Este componente trata do agendamento e cancelamento de treinos, onde se integra o *Stripe API* para o processamento seguro de pagamentos.

Lógica de Agendamento e Cancelamento: um conjunto de regras e lógica que garantem que os treinos personalizados e aulas de grupo só podem ser agendados durante os horários disponíveis, e que os treinos personalizados e aulas de grupo só podem ser cancelados pelo utilizador que os agendou, ou por um administrador.

Autenticar e Autorização: um sistema que garante que apenas utilizadores autorizados possam aceder à aplicação e realizar ações como agendamento e cancelamento de treinos personalizados e aulas de grupo.

Notificações: um sistema que envia notificações aos utilizadores, quando um treino personalizado ou aula de grupo é agendado, cancelado ou alterado.

Método de Pagamento (*Stripe API*): a integração assegura transações financeiras seguras. Este método de pagamento oferece garantias de segurança e eficácia, contribuindo para uma experiência de utilizador confiável e conveniente.

O *front-end* é a interface principal para a interação do utilizador, comunicando com o *back-end* para realizar operações como agendamento e consulta de agenda. O *back-end* trata da lógica de negócios, incluindo agendamento, cancelamento e integração com a base de dados.

A lógica de agendamento e cancelamento assegura que as operações são realizadas dentro de horários disponíveis, com restrições específicas para cancelamento. O sistema de autenticação e autorização controla o acesso, garantindo que apenas utilizadores autorizados possam efetuar ações na aplicação.

O sistema de notificações mantém os utilizadores informados sobre eventos relevantes, como agendamento ou cancelamento. A adição da *Stripe API* permite a realização de transações seguras e eficazes para os pagamentos associados aos treinos e eventos.

Esta arquitetura visa proporcionar uma aplicação robusta, segura e eficiente, integrando funcionalidades essenciais para uma experiência completa do utilizador.

5.2.3 Padrões arquiteturais e de software

No desenvolvimento da aplicação, a escolha de padrões arquiteturais é crucial para proporcionar uma boa experiência ao utilizador e uma gestão eficiente de estados e de eventos. Neste contexto, a aplicação adota o padrão de *BLoC (Business Logic Component)*, aliado ao uso de *Streams* no Flutter.

A aplicação EF integra o padrão *BLoC* de forma a separar distintamente a lógica de negócios da interface do utilizador. Este padrão é composto por componentes chave:

- **Lógica de Negócios:** Responsável por processar as regras de negócios, validações e manipulação de dados específicos à EF.

- *StreamController* (SC) na Gestão de Estados: Utilização de SC para gerir a transmissão de eventos da lógica de negócios para a interface do utilizador, proporcionando uma abordagem reativa.
- *StreamBuilder* (SB) na Interface Responsiva: A aplicação de SB para vincular a interface do utilizador aos *Streams* de saída do *BLoC*, garantindo uma resposta imediata às alterações de estado.

Para uma visualização mais detalhada da arquitetura BLoC adotada, consulte a Figura 13, que representa o desenho esquemático dessa estrutura na aplicação EF.

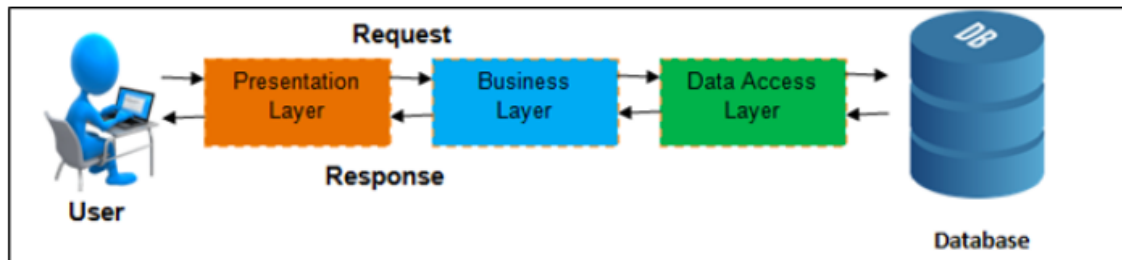


Figura 14 - Arquitetura BLoC

5.3 Tecnologias e Ferramentas Utilizadas

5.3.1 Tecnologias Utilizadas

- **Flutter**: Framework open-source para desenvolvimento de aplicações móveis e web. Foi escolhida pela sua capacidade de criar interfaces ricas e consistentes em diferentes plataformas, permitindo um desenvolvimento eficiente e multiplataforma.
- **Firebase**: Plataforma de desenvolvimento móvel e web da Google, que oferece diversos serviços, incluindo base de dados em tempo real, autenticação e armazenamento. Utilizado para guardar os dados da aplicação, garantindo uma base de dados segura e escalável.
- **Stripe API**: Plataforma de pagamentos online, proporciona uma API para o processamento seguro de transações financeiras. Essencial para implementar um método de pagamento seguro e eficaz, que será utilizado para o pagamento de eventos e treinos.

5.3.2 Perímetros Tecnológicos

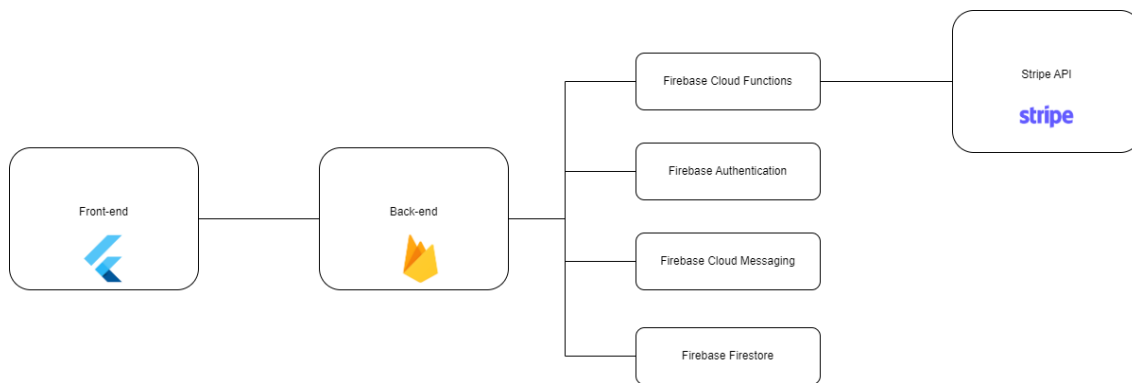


Figura 15 - Perímetros Tecnológicos

5.3.3 Ferramentas Utilizadas

- Android Studio: Ambiente de desenvolvimento integrado (IDE) para Flutter. Facilita o desenvolvimento eficiente de aplicações Flutter, oferecendo uma experiência de desenvolvimento coesa.
- Firebase Console: Interface web para gerir os serviços da Firebase. Permite a gestão centralizada da base de dados, autenticação e outros serviços Firebase.
- Stripe Dashboard: Plataforma online para a gestão de pagamentos com a Stripe API. Oferece ferramentas de administração e monitorização de transações, garantindo controle e segurança nos pagamentos online.

5.4 Implementação

Esta aplicação tem como objetivo fornecer uma plataforma digital para os alunos e PTs interagirem e gerirem as suas atividades no estúdio. A aplicação foi desenvolvida utilizando a linguagem de programação Flutter e integra o Firebase como back-end.

Funcionalidades da Aplicação: A aplicação "Ethical Fitness" consiste em quatro ecrãs principais, com funcionalidades distintas para alunos e PTs. De seguida, descrevemos cada ecrã e as suas funcionalidades específicas para cada tipo de utilizador. No Anexo 3 – Excertos de código, encontra-se excertos de código das diferentes funcionalidades implementadas.

Ecrã Inicial: O ecrã inicial serve como um feed de notícias e eventos relevantes para os utilizadores da aplicação. Para os alunos, este ecrã exhibe os posts criados pelos PTs, incluindo eventos e notícias do estúdio. Para os PTs, este ecrã permite a criação de posts, partilhando informações importantes com os alunos, além de oferecer a possibilidade de saber mais sobre os eventos e efetuar o seu pagamento. Este ecrã também dispõe de um feed de eventos que, ao ser selecionado, direciona o utilizador para o ecrã de detalhes do evento, onde pode encontrar todas as informações relevantes sobre o evento escolhido.

Ecrã Plano de Treino: Neste ecrã, os alunos podem visualizar o seu plano de treino personalizado. Infelizmente, devido a restrições de tempo, a funcionalidade de atribuir planos de treino aos alunos pelos PTs não foi implementada neste projeto, no projeto referente ao ano anterior, ano 2022/23.

Ecrã Aulas: Os alunos têm a possibilidade de inscrever-se ou desmarcar as aulas de grupo disponíveis no estúdio através deste ecrã. Os PTs, por sua vez, podem criar e apagar aulas de grupo. Também tem a vertente das aulas individuais onde aparecem as aulas do pt que foi atribuído a este aluno. Na implementação da disponibilidade dos PTs/admin para aulas individuais, adiciona-se o espaço de tempo ocupado e o código cria automaticamente várias aulas com duração de 1 hora entre as 8h e as 22h. Se o PT/admin desejar. Na inscrição dos alunos, tanto para aulas de grupo quanto individuais, a aplicação mostra as aulas de grupo de todos os PTs e as individuais do respetivo PT. Para as aulas do PT, aparecem apenas as que não têm ninguém inscrito ou onde o utilizador está inscrito. Em ambos os casos, tanto aulas de grupo quanto individuais, só são exibidas as aulas que ainda não tiveram início, também ambos os tipos de aulas podem ser editados ou eliminados pelo criador desta.

Ecrã Agenda: Neste ecrã, os alunos podem consultar a sua agenda tanto no formato mensal como semanal, onde são exibidas as aulas e treinos marcados. Os PTs também podem utilizar este ecrã para consultar a sua agenda pessoa. Na implementação da agenda, para os alunos, verifica-se em que aulas e eventos estes estão inscritos e guarda-se essa informação, exibindo-a na agenda mensal e semanal. Para os PTs, a lógica é similar, mas em vez de mostrar onde estão inscritos, exibe-se quais as aulas e eventos que são dados por eles.

Além disso, a aplicação Ethical Fitness inclui um robusto sistema de notificações para manter os utilizadores sempre informados sobre as suas atividades no estúdio. Este sistema de notificações foi implementado utilizando o Firebase Cloud Messaging, garantindo que os utilizadores recebam atualizações em tempo real.

Sempre que uma nova aula ou evento é adicionado ao sistema, são enviadas notificações a todos os alunos para os manter atualizados sobre as novas oportunidades. No caso específico das aulas dadas por PTs, as notificações são enviadas apenas para os alunos que estão associados ao respetivo PT, proporcionando uma comunicação direcionada e relevante.

Para garantir que os alunos estejam sempre preparados e cientes das suas atividades, o sistema realiza verificações automáticas através do Firebase Cloud Functions. Diariamente, há uma verificação das aulas agendadas para o dia seguinte, enviando notificações para lembrar os alunos dos compromissos futuros. Além disso, existe uma verificação contínua para identificar se alguma aula está prestes a começar nos próximos trinta minutos. Caso uma aula esteja iminente, uma notificação é enviada para avisar os alunos, garantindo que possam se preparar a tempo.

Este sistema de notificações foi cuidadosamente implementado para proporcionar uma experiência de utilizador otimizada, assegurando que os alunos e PTs estejam sempre bem informados sobre as suas atividades e compromissos no estúdio. O uso do Firebase Cloud Messaging e Firebase Cloud Functions permite uma comunicação eficiente e em tempo real, fundamental para a eficácia da aplicação Ethical Fitness.

Para a implementação da aplicação Ethical Fitness, foram utilizadas as seguintes bibliotecas e ferramentas:

Flutter: O Flutter é um framework de desenvolvimento de aplicações multiplataforma que permite criar interfaces de utilizador bonitas e responsivas. Foi escolhido devido à sua facilidade de utilização e à sua capacidade de compilar código para iOS e Android.

Firestore: O Firestore é uma plataforma de desenvolvimento de aplicações móveis que fornece uma ampla gama de serviços em cloud, incluindo autenticação de utilizadores, armazenamento de dados em tempo real e hospedagem. Para este projeto, foram utilizados os módulos *Firestore Auth*, *Firestore Firestore*, *Firestore Storage* e *Firestore Cloud Messaging*.

Stripe API: A integração do método de pagamento *Stripe* na aplicação oferece uma solução eficiente e segura para transações financeiras. Ao incorporar o *SDK* do *Stripe* para a APP, proporciona ao cliente uma compra fluida e confiável. O processo de implementação envolve a configuração de uma conta *Stripe*, obter as chaves da *API* necessárias e a integração do *SDK* no código Flutter.

Outras bibliotecas do Flutter: Além do Firestore, foram utilizadas várias bibliotecas Flutter para auxiliar no desenvolvimento da aplicação. Algumas dessas bibliotecas incluem o *Syncfusion Flutter Calendar* para exibir a agenda, o *Provider* para gestão de estado, o *Video Player* para reproduzir vídeos, o *Google Fonts* para estilização de texto e o *File Picker* para selecionar ficheiros.

5.5 Abrangência

Para a solução foi utilizado os conhecimentos adquiridos das cadeiras do curso de LIG, tais como:

- Engenharia de Requisitos e Testes – Conseguimos aprender a preparar um projeto previamente, como também é uma grande ajuda para o relatório do TFC.
- Interação Humano-Máquina – Esta cadeira foca-se na interação do utilizador com a aplicação, o estudo das cores, do design e acessibilidade. Podendo assim criar um protótipo atendendo estes aspetos.
- Sistemas Móveis Empresariais – Aprendemos as bases do Flutter e do Firestore.

6 Plano de testes e validação

Este plano de testes e validação visa garantir que a solução desenvolvida atenda aos requisitos especificados e funcione de forma eficaz e confiável em um ambiente de produção real. Os testes serão realizados com o objetivo de verificar a qualidade da solução, a sua operacionalidade e a sua capacidade de resolver problemas reais identificados pelo estúdio Xstudio. O plano aborda as áreas de notificações método de pagamento e detalhes do evento, conforme descrito nos requisitos levantados.

Os testes foram realizados de forma abrangente e sistemática, utilizando uma variedade de técnicas e ferramentas de teste. Os testes incluíram a verificação da conformidade com os requisitos especificados, a funcionalidade correta da solução em diferentes cenários e a validação do desempenho e segurança da aplicação. Foram realizados tantos testes manuais.

Os testes são funcionais para garantir a qualidade e a confiabilidade da solução desenvolvida. Eles são essenciais para identificar e corrigir quaisquer defeitos ou problemas que possam afetar a experiência do utilizador ou comprometer a segurança da aplicação. Além disso, os testes são importantes para validar a funcionalidade da solução em cenários de uso real, garantindo que atende às expectativas e necessidades dos utilizadores finais.

A aplicação foi testada em dispositivos iOS e Android para garantir a compatibilidade e funcionalidade em ambos os sistemas operativos. Os testes confirmaram que a aplicação corre com sucesso em ambos os ambientes, oferecendo uma experiência de utilizador consistente e confiável.

6.1 Guia Detalhado de Testes

O guia de testes detalhado incluirá os seguintes cenários de teste, entre outros que possam surgir durante o processo de desenvolvimento:

6.1.1 Testes de Notificações

6.1.1.1 *Verificar a Entrega em Tempo Real:*

Objetivo: Garantir que as notificações sejam entregues instantaneamente aos dispositivos dos utilizadores.

Passos:

1. Enviar uma notificação da aplicação.
2. Verificar se a notificação é entregue instantaneamente no dispositivo do utilizador.

Resultado esperado: A notificação é entregue imediatamente após o envio, sem atrasos perceptíveis.

6.1.1.2 *Interagir com Notificações:*

Objetivos: Testar se os utilizadores podem interagir com as notificações para abrir a aplicação ou executar ações associadas.

Passos:

1. Enviar uma notificação com uma ação associada, como abrir a aplicação ou visualizar um evento.
2. Verificar se os utilizadores podem interagir com a notificação e executar a ação associada.

Resultado esperado: Os utilizadores podem interagir com a notificação e executar a ação associada conforme esperado.

6.1.1.3 Desativar Notificações:

Objetivos: Confirmar se os utilizadores podem desativar as notificações e se isso é refletido corretamente na aplicação.

Passos:

1. Aceder às configurações das notificações na aplicação.
2. Desativar a opção de receber notificações.
3. Verificar se as notificações são desativadas e os utilizadores não recebem mais notificações após desativar essa opção.

Resultado esperado: As notificações são desativadas com sucesso e os utilizadores não recebem mais notificações após desativar essa opção.

6.1.2 Teste de Método de Pagamento

6.1.2.1 Adicionar e Gestão de Métodos de Pagamento:

Objetivo: Verificar se os utilizadores podem adicionar e gerir os métodos de pagamento na aplicação.

Passos:

1. Tentar adicionar um novo método de pagamento na aplicação.
2. Gerir os métodos de pagamento existentes, como editar ou excluir.

Resultado esperado: Os utilizadores conseguem adicionar novos métodos de pagamento e gerir os métodos de pagamento existentes sem problemas.

6.1.2.2 Guardar e Remover Informações do Cartão:

Objetivo: Testar a capacidade de guardar e remover informações do cartão de forma segura.

Passos:

1. Adicionar um novo cartão de crédito na aplicação.
2. Remover as informações do cartão.

Resultado esperado: Os utilizadores podem adicionar informações do cartão de forma segura e remover as informações do cartão quando necessário.

6.1.2.3 Receber Confirmações de Transações Bem-Sucedidas:

Objetivo: Confirmar se os utilizadores recebem confirmações de transações bem-sucedidas após realizar um pagamento.

Passos:

1. Realizar uma transação de teste na aplicação.
2. Verificar se o utilizador recebe uma confirmação de transação bem-sucedida.

Resultado esperado: Os utilizadores recebem uma confirmação de transação bem-sucedida após realizar um pagamento com sucesso.

6.1.3 Teste de Detalhes do Evento

6.1.3.1 Exibição Correta dos Eventos:

Objetivo: Verificar se os eventos são corretamente exibidos na aplicação.

Passos:

1. Aceder à secção de eventos na aplicação.
2. Verificar se todos os eventos estão listados corretamente, incluindo detalhes como data, hora e descrição.

Resultado esperado: Todos os eventos são exibidos corretamente na aplicação, conforme esperado.

6.1.3.2 Funcionalidade de Inscrição em Eventos:

Objetivo: Testar a funcionalidade de inscrição em eventos na aplicação.

Passos:

1. Tentar se inscrever num evento disponível na aplicação.

Resultado esperado: Os utilizadores conseguem se inscrever em eventos disponíveis na aplicação sem problemas.

6.1.3.3 Receber Notificações sobre Eventos Importantes:

Objetivo: Confirmar se os utilizadores recebem notificações sobre eventos importantes.

Passos:

1. Criar um evento importante na aplicação.
2. Verificar se os utilizadores recebem uma notificação sobre esse evento.

Resultado esperado: Os utilizadores recebem uma notificação sobre os eventos importantes, garantindo que estejam cientes das atualizações e novidades na aplicação.

6.2 Resultados dos Testes Planeados

A última entrega desta fase de desenvolvimento inclui os resultados dos testes detalhados anteriormente, demonstrando que a solução cumpre os objetivos propostos. Esses resultados são organizados de forma a evidenciar a implementação e a operacionalidade dos requisitos trabalhados ao longo do projeto. Sempre que aplicável, melhorias introduzidas na solução como resultado dos testes realizados são indicadas. Os testes e validações refletiram cenários reais de utilização da solução.

6.2.1 Resultados dos Testes de Notificações

6.2.1.1 Verificar a Entrega em Tempo Real

Objetivo: Garantir que as notificações sejam entregues instantaneamente aos dispositivos dos utilizadores.

Passos:

1. Enviar uma notificação da aplicação.
2. Verificar se a notificação é entregue instantaneamente no dispositivo do utilizador.

Resultado esperado: A notificação é entregue imediatamente após o envio, sem atrasos perceptíveis.

Resultados Obtidos:

- **Data do Teste:** 15/06/2024
- **Dispositivos Testados:** iPhone 15 Plus (iOS 17.5.1), Samsung Galaxy S22 (Android 14)
- **Resultado:** Notificação entregue em menos de 3 segundo em ambos os dispositivos.

6.2.1.2 Interagir com Notificações

Objetivo: Testar se os utilizadores podem interagir com as notificações para abrir a aplicação ou executar ações associadas.

Passos:

1. Enviar uma notificação com uma ação associada, como abrir a aplicação ou visualizar um evento.
2. Verificar se os utilizadores podem interagir com a notificação e executar a ação associada.

Resultado esperado: Os utilizadores podem interagir com a notificação e executar a ação associada conforme esperado.

Resultados Obtidos:

O requisito necessário para a testagem deste teste não foi implementado.

6.2.1.3 *Desativar Notificações*

Objetivo: Confirmar se os utilizadores podem desativar as notificações e se isso é refletido corretamente na aplicação.

Passos:

1. Aceder às configurações das notificações na aplicação.
2. Desativar a opção de receber notificações.
3. Verificar se as notificações são desativadas e os utilizadores não recebem mais notificações após desativar essa opção.

Resultado esperado: As notificações são desativadas com sucesso e os utilizadores não recebem mais notificações após desativar essa opção.

Resultados Obtidos:

- **Data do Teste:** 17/06/2024
- **Dispositivos Testados:** iPhone 15 Plus (iOS 17.5.1), Samsung Galaxy S22 (Android 14)
- **Resultado:** Notificações desativadas com sucesso em ambos os dispositivos.

6.2.2 *Resultados dos Testes de Método de Pagamento*

6.2.2.1 *Adicionar e Gestão de Métodos de Pagamento*

Objetivo: Verificar se os utilizadores podem adicionar e gerir os métodos de pagamento na aplicação.

Passos:

1. Tentar adicionar um novo método de pagamento na aplicação.
2. Gerir os métodos de pagamento existentes, como editar ou excluir.

Resultado esperado: Os utilizadores conseguem adicionar novos métodos de pagamento e gerir os métodos de pagamento existentes sem problemas.

Resultados Obtidos:

- **Data do Teste:** 18/06/2024
- **Dispositivos Testados:** iPhone 15 Plus (iOS 17.5.1), Samsung Galaxy S22 (Android 14)
- **Resultado:** Adição e gestão de métodos de pagamento funcionaram conforme esperado.

6.2.2.2 *Guardar e Remover Informações do Cartão*

Objetivo: Testar a capacidade de guardar e remover informações do cartão de forma segura.

Passos:

1. Adicionar um novo cartão de crédito na aplicação.
2. Remover as informações do cartão.

Resultado esperado: Os utilizadores podem adicionar informações do cartão de forma segura e remover as informações do cartão quando necessário.

Resultados Obtidos:

O requisito necessário para a testagem deste teste não foi implementado.

6.2.2.3 Receber Confirmações de Transações Bem-Sucedidas

Objetivo: Confirmar se os utilizadores recebem confirmações de transações bem-sucedidas após realizar um pagamento.

Passos:

1. Realizar uma transação de teste na aplicação.
2. Verificar se o utilizador recebe uma confirmação de transação bem-sucedida.

Resultado esperado: Os utilizadores recebem uma confirmação de transação bem-sucedida após realizar um pagamento com sucesso.

Resultados Obtidos:

- **Data do Teste:** 20/06/2024
- **Dispositivos Testados:** iPhone 15 Plus (iOS 17.5.1), Samsung Galaxy S22 (Android 14)
- **Resultado:** Confirmação de transação recebida imediatamente após a conclusão do pagamento.

6.2.3 Resultados dos Testes de Detalhes do Evento

6.2.3.1 Exibição Correta dos Eventos

Objetivo: Verificar se os eventos são corretamente exibidos na aplicação.

Passos:

1. Aceder à secção de eventos na aplicação.
2. Verificar se todos os eventos estão listados corretamente, incluindo detalhes como data, hora e descrição.

Resultado esperado: Todos os eventos são exibidos corretamente na aplicação, conforme esperado.

Resultados Obtidos:

- **Data do Teste:** 21/06/2024
- **Dispositivos Testados:** iPhone 15 Plus (iOS 17.5.1), Samsung Galaxy S22 (Android 14)
- **Resultado:** Eventos exibidos corretamente com todos os detalhes necessários.

6.2.3.2 *Funcionalidade de Inscrição em Eventos*

Objetivo: Testar a funcionalidade de inscrição em eventos na aplicação.

Passos:

1. Tentar se inscrever num evento disponível na aplicação.

Resultado esperado: Os utilizadores conseguem se inscrever em eventos disponíveis na aplicação sem problemas.

Resultados Obtidos:

- **Data do Teste:** 22/06/2024
- **Dispositivos Testados:** iPhone 15 Plus (iOS 17.5.1), Samsung Galaxy S22 (Android 14)
- **Resultado:** Inscrição em eventos realizada com sucesso em ambos os dispositivos.

6.2.3.3 *Receber Notificações sobre Eventos Importantes*

Objetivo: Confirmar se os utilizadores recebem notificações sobre eventos importantes.

Passos:

1. Criar um evento importante na aplicação.
2. Verificar se os utilizadores recebem uma notificação sobre esse evento.

Resultado esperado: Os utilizadores recebem uma notificação sobre os eventos importantes, garantindo que estejam cientes das atualizações e novidades na aplicação.

Resultados Obtidos:

- **Data do Teste:** 23/06/2024
- **Dispositivos Testados:** iPhone 15 Plus (iOS 17.5.1), Samsung Galaxy S22 (Android 14)
- **Resultado:** Notificações sobre eventos importantes recebidas corretamente.

7 Método e Planeamento

A continuação do desenvolvimento da aplicação *EF* será realizada com a *framework* Flutter e *Firebase*, escolhidas no início deste TFC no ano de 2022/23, permitindo a criação de uma APP multiplataforma de forma mais facilitada. O uso de metodologias ágeis, como o SCRUM, será adotado para facilitar o desenvolvimento iterativo e incremental, permitindo ajustes e melhorias com base no feedback recebido ao longo do processo.

O desenvolvimento deste trabalho está dividido em quatro fases, correspondentes às quatro entregas agendadas do TFC.

Na primeira entrega, correspondente à fase inicial do trabalho, foram elaborados o Project Charter, a identificação do problema, viabilidade e pertinência, solução do problema, benchmarking e método e planeamento, onde é desenvolvido o primeiro relatório intercalar.

Na segunda fase, iniciarei o levantamento de requisitos, desenvolver o protótipo da APP, diagramas e estruturas relacionados a esta fase. Posteriormente, o relatório intercalar será aprimorado, levando em consideração as sugestões dos jurados, resultando no relatório intermédio.

Na terceira etapa, será efetuado o desenvolvimento efetivo da APP, incluindo um protótipo funcional, *test case*, e a continuação do relatório, incorporando as recomendações dos jurados.

Na última fase, correspondente à entrega final do trabalho, será feita uma complementação do trabalho desenvolvido anteriormente, considerando as sugestões dos jurados. Serão incluídos texto e imagem para apresentação do trabalho, juntamente com um vídeo de 5 minutos a demonstrar o funcionamento da aplicação.

O calendário proposto encontra-se representado na Figura 16 incluindo todas as etapas previstas para a execução do trabalho final de curso.

No entanto, na Figura 17, observa-se uma diferença significativa em relação ao calendário proposto inicialmente. Esta discrepância deve-se ao facto de a reestruturação da aplicação existente não estar prevista no planeamento original. Esta atividade adicional revelou-se necessária para garantir a integridade e a funcionalidade da aplicação, mas consumiu uma quantidade substancial de tempo. Como resultado, houve um impacto direto nas outras atividades planeadas, causando um atraso no cronograma geral. Esta mudança de cronograma exigiu uma reavaliação das etapas subsequentes e uma adaptação das entregas para assegurar que todas as metas do TFC fossem atingidas com a qualidade esperada. A reestruturação incluiu revisões do código da aplicação que era essencial para o progresso do desenvolvimento, mas que não foram consideradas inicialmente, demonstrando a importância de flexibilidade e adaptação na gestão de projetos.

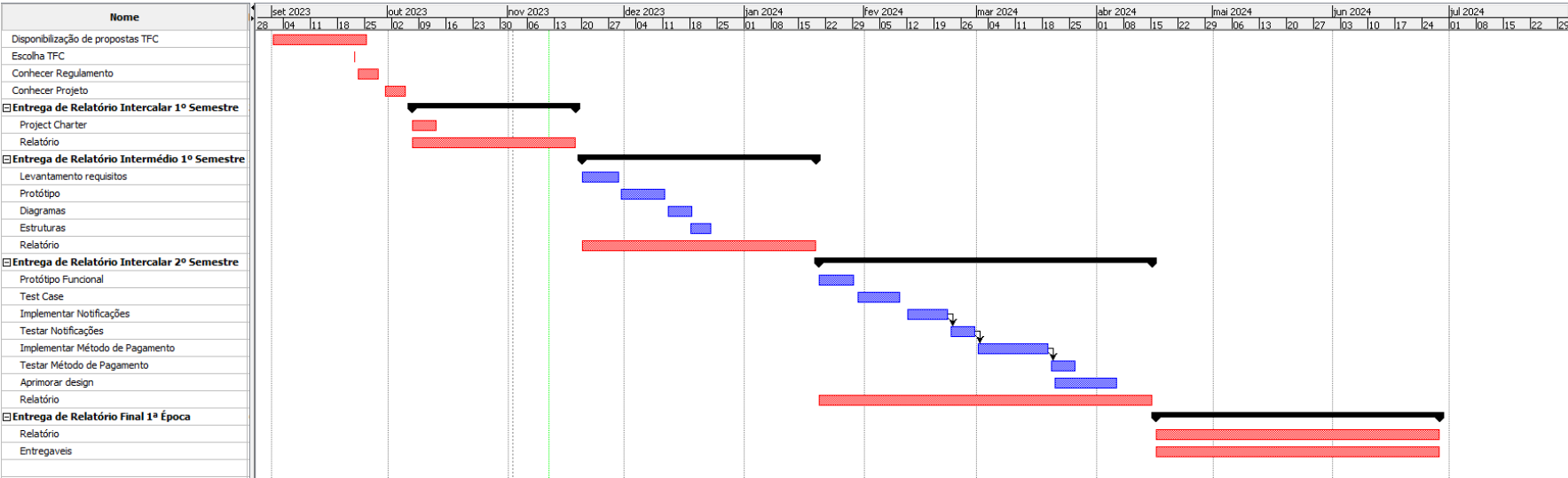


Figura 16 - Calendário de Execução Previsto

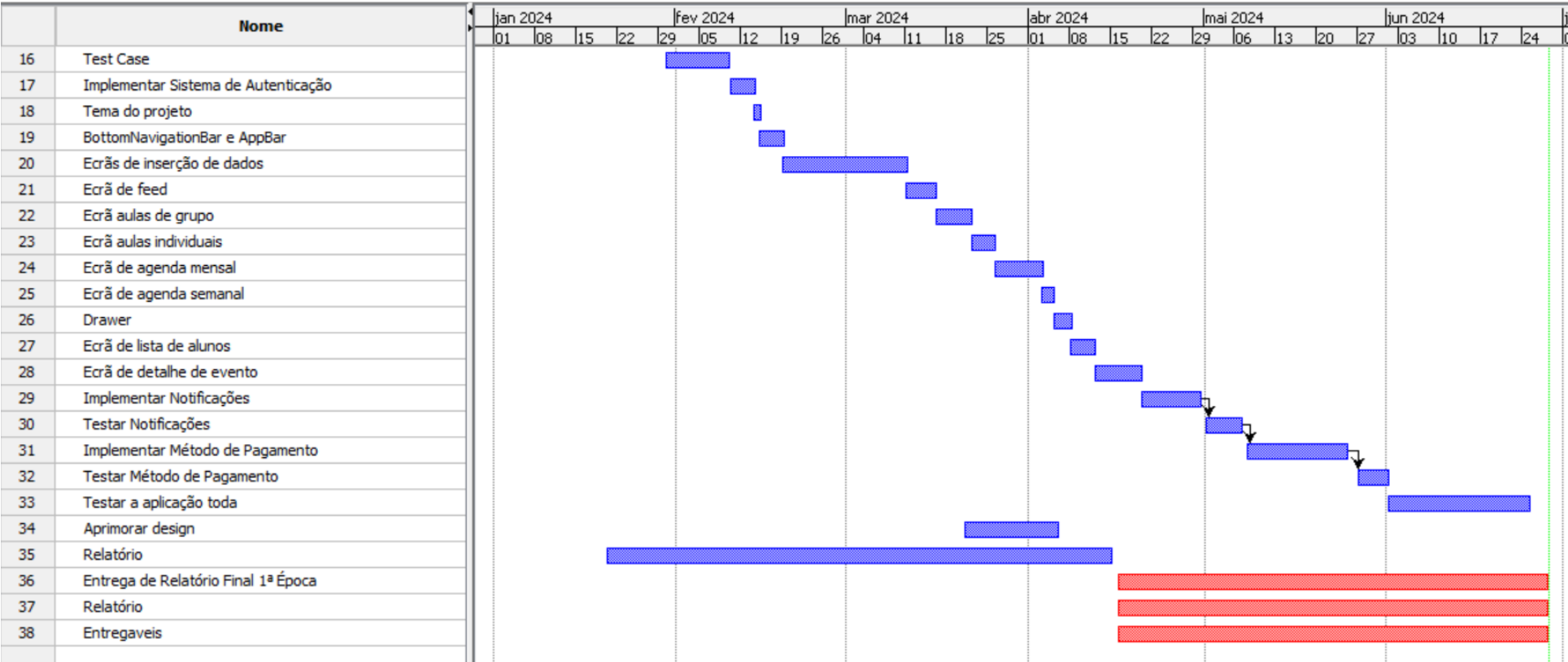


Figura 17 - Calendário Final

8 Resultados

Este capítulo apresenta uma análise detalhada dos resultados obtidos, outputs gerados e outcomes alcançados durante o desenvolvimento da solução. A identificação de resultados foi realizada em paralelo com a análise da viabilidade e pertinência, considerando o cumprimento dos critérios de sucesso determinados no levantamento de requisitos. Além disso, as revisões realizadas ao longo do desenvolvimento do TFC e os resultados dos test cases definidos no segundo relatório intercalar são detalhados e complementados por um anexo com os test cases e respectivos resultados.

8.1 Resultados Obtidos

Os resultados dos testes realizados confirmaram que a solução desenvolvida atende à maioria dos requisitos especificados e funciona de forma eficaz e confiável. A seguir, os principais resultados são organizados de acordo com os critérios de sucesso estabelecidos no levantamento de requisitos.

8.1.1 Notificações

Os critérios de sucesso incluíam a entrega instantânea das notificações e a opção de desativação de notificações. Nos testes, verificou-se que as notificações foram entregues instantaneamente nos dispositivos iOS e Android, atendendo ao critério de sucesso para entrega em tempo real. A opção de desativação de notificações funcionou conforme especificado, permitindo que os utilizadores desativassem e não recebessem mais notificações após desativar essa opção.

8.1.2 Método de Pagamento

Os critérios de sucesso incluíam a adição e gestão de métodos de pagamento, e o recebimento de confirmações de transações bem-sucedidas. Nos testes, os utilizadores puderam adicionar e gerir métodos de pagamento sem dificuldades, confirmando a funcionalidade conforme especificada. As confirmações de transações foram recebidas imediatamente após a conclusão dos pagamentos, conforme esperado.

8.1.3 Detalhes do Evento

Os critérios de sucesso incluíam a exibição correta dos eventos, a funcionalidade de inscrição em eventos e o recebimento de notificações sobre eventos importantes. Nos testes, todos os eventos foram exibidos corretamente com detalhes como data, hora e descrição, atendendo ao critério de exibição correta. Os utilizadores conseguiram se inscrever em eventos disponíveis sem problemas, confirmando a funcionalidade conforme especificada. As notificações sobre eventos importantes foram recebidas corretamente, garantindo que os utilizadores estivessem cientes das atualizações.

8.2 Outputs Gerados

Os outputs gerados durante o desenvolvimento e testes da aplicação incluem relatórios de testes, documentos detalhando cada teste realizado, os dispositivos utilizados, os passos seguidos e os resultados obtidos. O código fonte da aplicação foi atualizado com as melhorias implementadas como resultado dos testes.

8.3 Outcomes Alcançados

Os outcomes alcançados indicam o impacto real da aplicação no contexto de uso e sua contribuição para resolver os problemas identificados. A melhoria na experiência do utilizador foi significativa, com a entrega instantânea de notificações. A interação com eventos foi incentivada, com a exibição correta dos eventos e a funcionalidade de inscrição fácil promovendo maior participação dos utilizadores.

8.4 Cumprimento dos Critérios de Sucesso

Os testes realizados confirmam que a aplicação cumpre os critérios de sucesso estabelecidos no levantamento de requisitos. As funcionalidades de notificação atendem aos requisitos de entrega instantânea e desativação. As funcionalidades de método de pagamento são seguras e fáceis de usar, com confirmações de transações entregues conforme esperado. A exibição e a inscrição em eventos funcionam conforme especificado, com notificações importantes entregues de maneira confiável.

8.5 Revisões e Melhorias Introduzidas

Ao longo do desenvolvimento e testes, foram realizadas diversas revisões e melhorias na solução, incluindo a otimização do serviço de notificações para reduzir a latência, ajustes na interface de notificação para tornar as ações mais intuitivas, implementação de criptografia adicional para proteger os dados do cartão e a otimização do layout de exibição de eventos para melhor visualização.

Além disso, foram encontrados vários erros na aplicação existente, que necessitaram de uma reformulação completa da aplicação com foco em colmatar esses aspetos. No ecrã inicial, foi introduzido um feed onde podemos aceder todos os eventos que irão acontecer e visualizar suas informações. Também há a possibilidade de se inscrever e pagar. No ecrã das aulas, aparecem as aulas de grupo e individuais, onde os alunos podem se inscrever, e os *personal trainers* podem ver o número de alunos inscritos e quais os alunos inscritos, respetivamente. Na agenda, aparecem as aulas e eventos que cada utilizador tem atribuído, numa visão semanal e mensal. Esta agenda atualiza conforme as marcações e desmarcações de eventos e aulas. Ainda há um *drawer* que permite fazer *logout*, ligar e desligar notificações, e, para o administrador, atribuir *personal trainers* aos alunos. No canto superior direito da *AppBar*, encontra-se a lista de notificações que aquele utilizador recebeu.

Podemos ver que a Figura 19, tem a estrutura muito mais organizada e usa melhores práticas de desenvolvimento, do que a Figura 18.









Name
 ..
 models
 pages
 services
 utils
 firebase_options.dart
 homePage2.dart
 main.dart

Figura 18 - Antiga estrutura da aplicação Ethical Fitness

Name
..
add_event
add_group_class
add_post
add_pt_class
agenda
app
classes
forgot_password
form_inputs
home
login
models
notification
payment
repository
services
sign_up
widgets
firebase_options.dart
generated_plugin_registrant.dart
main.dart
themes.dart

Figura 19 - Nova estrutura da aplicação Ethical Fitness

Na Figura 20, podemos verificar que a inserção de horários e de tipos de aulas são manualmente, este tipo de *input* se forem mal colocados, por erro humano, que é bastante comum, podem prejudicar a coerência dos dados, levando a outros erros. Na Figura 21, com *inputs* de data garante que não esta exposto a riscos de falta de coerência de dados e erros futuros, o mesmo acontece no *dropdown*, os valores destes são retirados da base de dados para manter a coerência.

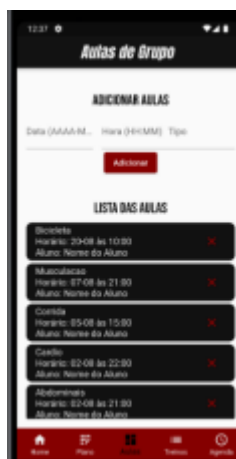


Figura 20 - Antigo ecrã de inserção de aulas de grupo

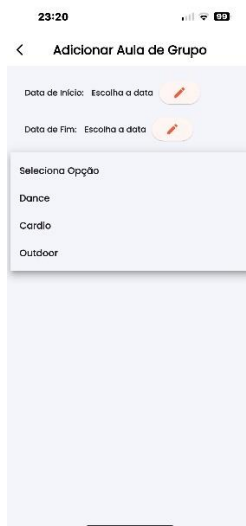


Figura 21 - Novo ecrã de inserção de aulas de grupo

9 Conclusão e trabalhos futuros

9.1 Conclusão

O desenvolvimento da aplicação móvel utilizando Flutter e Firebase resultou numa solução robusta e eficiente que atendeu aos requisitos especificados. A reestruturação completa da aplicação foi essencial para resolver problemas significativos encontrados na versão inicial, como a mistura de interface de utilizador com lógica de negócios, uma base de dados mal estruturada e a falta de um tema consistente. Essa reestruturação exigiu uma taxa de esforço muito alta, demandando tempo e recursos significativos para garantir a correção das falhas existentes e a implementação de novas funcionalidades de forma eficaz.

A implementação de novas funcionalidades, como notificações em tempo real, métodos de pagamento e melhorias no design, melhoraram significativamente a experiência do utilizador. Os testes de validação confirmaram a eficácia dessas funcionalidades e a estabilidade do sistema reestruturado, com desempenho confiável em todos os aspetos testados.

A adição de um *feed* inicial para eventos, a capacidade de inscrição e pagamento, a organização de aulas de grupo e individuais, uma agenda dinâmica e a interface de notificações contribuíram para um sistema mais intuitivo e funcional. A criação de um *drawer* com funcionalidades adicionais, como *login/logout*, gestão de notificações e atribuição de *personal trainers*, bem como uma lista de notificações recebidas, melhorou a usabilidade e a gestão do sistema.

9.2 Trabalhos Futuros

Para continuar a melhorar a aplicação e atender às necessidades dos utilizadores, sugerem-se os seguintes trabalhos futuros:

Chat: Implementação de um chat privado para que os clientes e os *personal trainers* possam comunicar, solicitar orientações e esclarecer dúvidas. Essa funcionalidade facilitará a interação direta e personalizada entre os utilizadores, melhorando o suporte e a satisfação dos clientes.

Nutrição e Dieta: Implementar uma secção que inclua receitas, indicações de alimentos saudáveis e sugestões de planos de dietas. Esta funcionalidade proporcionará aos utilizadores um suporte adicional na área de nutrição, ajudando-os a alcançar seus objetivos de forma mais eficaz.

Avaliações e Comentários: Permitir que os utilizadores avaliem os eventos, exercícios e *personal trainers*. A inclusão de avaliações e comentários contribuirá para a transparência e a melhoria contínua dos serviços oferecidos, além de ajudar outros utilizadores a fazer escolhas informadas.

Sincronização com Smartwatch: Introdução de uma funcionalidade que permita aos utilizadores sincronizar suas informações de treino em tempo real com smartwatches, facilitando o acompanhamento do exercício do aluno e a compreensão do seu desempenho. Esta funcionalidade permitirá um monitoramento mais preciso e motivará os utilizadores a se manterem ativos.

"Redes Sociais": Criação de uma espécie de "Redes Sociais" onde alunos e *personal trainers* possam partilhar eventos, conquistas e treinos. Esta plataforma promoverá a interação social e o engajamento entre os utilizadores, criando uma comunidade mais unida e ativa.

Filtros nos Treinos: Implementação de filtros para que os utilizadores possam aceder a vídeos de exercícios que se concentrem numa parte específica do seu interesse, por exemplo, exercícios focados em braços. Esta funcionalidade tornará a busca por treinos mais eficiente e personalizada, atendendo melhor às necessidades dos utilizadores.

Página de Perfil: Criação de uma página que mostre o perfil do cliente e do *personal trainer*, onde podem constar várias informações, como nome, email, possibilidade de alterar senha, *logout* e opção de modo escuro (*dark mode*). Sugere-se a substituição do símbolo de *logout* por um ícone de perfil que, quando clicado, abre um perfil de utilizador com várias opções. Esta funcionalidade centralizará as informações pessoais e as configurações, melhorando a experiência do utilizador.

Com a implementação dessas funcionalidades futuras, espera-se aumentar ainda mais a usabilidade, funcionalidade e satisfação dos utilizadores, tornando a aplicação uma ferramenta completa e indispensável para a gestão de treinos, eventos e interações entre clientes e *personal trainers*.

Bibliografia

- [DEISI23] DEISI, Regulamento de Trabalho Final de Curso, Set. 2023.
- [ULHT23] Universidade Lusófona de Humanidades e Tecnologia, www.ulusofona.pt,
acedido em Out. 2023.
- [GF23] Aplicação GoFit, <https://apps.apple.com/pt/app/my-go-fit/id786133717>,
acedido a Nov. 2023
- [SOL23] Aplicação Solinca, <https://apps.apple.com/pt/app/solinca/id1510608997>,
acedido a Nov. 2023
- [FO23] Aplicação FitOn, <https://apps.apple.com/us/app/fiton-workouts-fitness-plans/id1442473191>,
acedido a Nov. 2023

Anexo 1 – Project Charter

<https://drive.google.com/file/d/13Z14Hs0K95u96uJBrz5li39WjAOhms8C/view?usp=sharing>

Anexo 2 – Vídeo Youtube e GitHub

[Vídeo YouTube](#)

[Link GitHub](#)

Anexo 3 – Excertos de código

```
static Future<Map<DateTime, List<CalendarEvent>>> fetchGroupEventsStudent() async {
  Map<DateTime, List<CalendarEvent>> events = {};
  User? user = FirebaseAuth.instance.currentUser;

  if (user == null) return events;

  try {
    QuerySnapshot snapshot = await FirebaseFirestore.instance
      .collection('groupClass')
      .where('students', arrayContains: user.uid)
      .get();

    for (var doc in snapshot.docs) {
      Map<String, dynamic> data = doc.data() as Map<String, dynamic>;
      DateTime start = (data['startTime'] as Timestamp).toDate();
      DateTime end = (data['endTime'] as Timestamp).toDate();

      // Fetch the class type
      String classTypeId = data['classType'];
      String? classType = await getClassType(classTypeId);

      // Use classType in the event name or in other appropriate manner
      String eventName = classType ?? 'Group Class';

      DateTime eventDay = DateTime(start.year, start.month, start.day);
      events[eventDay] ??= [];
      events[eventDay]!.add(CalendarEvent(eventName, start, end));
    }
  } catch (e) {
    print("Error fetching group events: $e");
  }

  return events;
}
```

Figura 22 - Código de busca de eventos para calendário

```
Future<void> insertPTClass() async {  
  DateTime start = state.startTime.value!.subtract(const Duration(hours: 1));  
  DateTime end = state.startTime.value!;  
  
  while (start.hour >= 8) {  
    PTClass ptClass = PTClass(  
      startTime: start,  
      endTime: end,  
      student: null,  
    );  
    firebaseFirestorePTClass.insertPTClass(  
      ptClass: ptClass,  
    );  
    end = start;  
    start = start.subtract(const Duration(hours: 1));  
  }  
  
  start = state.endTime.value!;  
  end = state.endTime.value!.add(const Duration(hours: 1));  
  
  while (end.hour <= 22) {  
    PTClass ptClass = PTClass(  
      startTime: start,  
      endTime: end,  
      student: null,  
    );  
    firebaseFirestorePTClass.insertPTClass(  
      ptClass: ptClass,  
    );  
    start = end;  
    end = start.add(const Duration(hours: 1));  
  }  
  return;  
}
```

Figura 23 - Código para inserção de novas aulas pt

```

Widget _buildEnrollmentButton(GroupClass groupClass) {
  return FutureBuilder<bool>({
    future: _groupClassRepository.isEnrolled(groupClass),
    builder: (context, snapshot) {
      if (snapshot.hasData) {
        final enrolled = snapshot.data!;
        return ElevatedButton(
          onPressed: () {
            _groupClassRepository.toggleEnrollment(groupClass);
          },
          child: Text(enrolled ? 'Cancelar' : 'Marcar'),
        ); // ElevatedButton
      } else if (snapshot.hasError) {
        return Text('Error: ${snapshot.error}');
      } else {
        return const CircularProgressIndicator();
      }
    },
  ); // FutureBuilder
}

```

Figura 24 - Botão de marcar e desmarcar aulas

```

Future<void> toggleEnrollment(GroupClass groupClass) async {

  final currentUserUid = FirebaseAuth.instance.currentUser?.uid;

  if (currentUserUid == null) {
    throw Exception("User not logged in");
  }

  final groupClassRef = _firebaseFirestoreGroupClass.collection('groupClass').doc(groupClass.uid);
  final snapshot = await groupClassRef.get();

  if (!snapshot.exists) {
    throw Exception("Group class not found");
  }

  List<String> students = List<String>.from(snapshot.data()?['students'] ?? []);
  if (students.contains(currentUserUid)) {
    students.remove(currentUserUid);
  } else {
    students.add(currentUserUid);
  }

  await groupClassRef.update({'students': students});
}

```

Figura 25 - Lógica do botão de marcar e desmarcar aulas

```
child: SwipeButton(  
  width: 250,  
  activeThumbColor: colorScheme.primary,  
  activeTrackColor: colorScheme.primary.withOpacity(0.2),  
  child: Text(  
    "Desliza para pagar",  
    style: TextStyle(  
      color: colorScheme.onPrimary,  
    ), // TextStyle  
  ), // Text  
  onSwipe: () async {  
    await makePayment(  
      context: context,  
      amount: widget.event!.price.toString(),  
      currency: 'EUR',  
      merchantDisplayName: 'Ethical Fitness',  
      eventId: widget.event!.uid!,  
    );  
  },  
), // SwipeButton  
) , // Padding  
, // Center
```

Figura 26 - Botão pagamento evento e emissão para API

```
exports.sendNotificationOnNewGroupClass = functions.firestore  
  .document("groupClass/{docId}")  
  .onCreate(async (snapshot, context) => {  
    const newGroupClassData = snapshot.data();
```

Figura 27 - Cloud Functions código de nova aula de grupo


```
final settings = await messaging.requestPermission(  
  alert: true,  
  announcement: false,  
  badge: true,  
  carPlay: false,  
  criticalAlert: false,  
  provisional: false,  
  sound: true,  
);
```

Figura 28 - Pedido de permissões de notificações

Glossário

LEI	Licenciatura em Engenharia Informática
LIG	Licenciatura em Informática de Gestão
TFC	Trabalho Final de Curso
<i>PT</i>	<i>Personal Trainer</i>
APP	Aplicação
<i>EF</i>	<i>Ethical Fitness</i>
<i>FCM</i>	<i>Firebase Cloud Messaging</i>
<i>API</i>	<i>Application Programming Interface</i>
<i>SDK</i>	<i>Software Development Kit</i>
<i>BLoC</i>	<i>Business Logic Component</i>
<i>SC</i>	<i>StreamController</i>
<i>SB</i>	<i>StreamBuilder</i>
<i>UML</i>	<i>Unified Modeling Language</i>