Logotipo

Descrição gerada automaticamente

Licenciatura em Engenharia Informática (BLEI)

Unidade Curricular de Computação Móvel

3.º Ano

**TG2 (Trabalho de Grupo 2)**

Docente Responsável

Prof. Dr. Célio Carvalho

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Trabalho elaborado por:

**João Madail e Joaquim Ferreira**

Vila Nova de Gaia, 3 de Fevereiro de 2023

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Índice**

[Índice de Figuras 4](#_Toc126203090)

[No table of figures entries found. 4](#_Toc126203091)

[Introdução 5](#_Toc126203092)

[Ferramentas 5](#_Toc126203093)

[Estado de Arte 6](#_Toc126203094)

[Descrição do Problema e Sua Resolução 7](#_Toc126203095)

[Enunciado 7](#_Toc126203096)

[Armazenamento 8](#_Toc126203097)

[Interface 9](#_Toc126203098)

[Conclusão 12](#_Toc126203099)

[Bibliografia 12](#_Toc126203100)

# Índice de Figuras

[Figura 1- Objetos por segundo 8](#_Toc126203115)

[Figura 2 - Query ObjectBox 8](#_Toc126203116)

[Figura 3 - Jogadores 1 9](#_Toc126203117)

[Figura 4 - Jogadores 2 10](#_Toc126203118)

[Figura 5 - Jogadores 3 10](#_Toc126203119)

[Figura 6 - Jogadores 4 11](#_Toc126203120)

[Figura 7 - Equipas 11](#_Toc126203121)

# Introdução

O presente relatório, descreve as tarefas desenvolvidas ao longo da resolução exercícios propostos para o Trabalho de Grupo 2 (TG2).

TG1 teve como principais objetivos consolidar os conhecimentos obtidos nas aulas e ao longo do semestre, mantendo o foco na prática de regras de semântica da linguagem de programação Dart, explorando as suas funcionalidades do kit de desenvolvimento Flutter, utilizando widgets do material design e explorando a sua personalização.

Pegando nas palavras da equipa da Google destacada no desenvolvimento de Flutter “Flutter is an open source framework by Google for building beautiful, natively compiled, multi-platform applications from a single codebase.”, é esta a essência de Flutter, desenvolvimento multiplataforma rápido, produtivo e bonito, sem nunca esquecer a simplicidade.

# Ferramentas Utilizadas

* Flutter/DART
* ObjectBox
* Repositório Git : <https://github.com/H2OAguas/cm_trab2>
* Visual Studio Code
* Draw.io

# Estado de Arte

Com a aumento da utilização de dispositivos, e, consequente, evolução do mercado móvel no início da passada década e explosão da Facebook tornou-se bastante natural a criação de alternativas às frameworks e formas de desenvolver em ambiente móvel. Assim, em 2013 a Facebook lançou a React Native, uma framework multiplataforma, open source criada com o foco no utilizador e sua interface. Performance excecional, reutilização e compatibilidade com extensões de terceiros, assim criou-se um líder de mercado.

Contudo, é impossível soletrar FAANG (Facebook, Apple, Amazon, Netflix e Google) sem Google e, assim, em Maio de 2017 a Google lança o Flutter, um kit de desenvolvimento com base em Dart criado para simplificar, sem subtrair qualidade, o desenvolvimento de aplicações multiplataforma.

Embora ambas serem excelentes opções, ambas open source, ambas com bastante reutilização de código, o Flutter não só é capaz de compilação, mas também é capaz de o fazer nativamente ao contrário de React. Construção simples e bela. Foi assim que Flutter conquistou uma grande fatia do mercado móvel e tem hoje, percentualmente, quase tantos e utilizadores e criados quanto React

# Descrição do Problema e Sua Resolução

## Enunciado

“Crie uma aplicação que permita gerir as diferentes competições, equipas e jogadores inscritos, de uma época de futebol. Além dos dados genéricos dos jogadores (e.g. nome, idade, peso, altura, escolaridade), é necessário manter o histórico de contratações dos jogadores (incluindo as contratações pelos clubes

estrangeiros).

Entre outras funcionalidades, deverá ser possível:

• Visualizar uma listagem dos jogadores ativos por clube, apresentando a sua data de contratação

e a duração do seu contrato em dias até o momento da apresentação da listagem;

• Visualizar os jogadores cujos contratos terão de ser renovados nos próximos 6 meses (a listagem

deve ser agrupada pelo clube a que pertence cada jogador);

• Apresentar uma lista dos jogadores inscritos numa determinada competição, indicando o clube

que o inscreveu (assuma o passaporte como um documento obrigatório para a inscrição de um

jogador na LPFP);

• Consultar a lista dos jogadores que não foram sujeitos a exames de controlo antidoping há mais que 𝑥 dias, em que o valor de 𝑥 pode ser indicado pelo utilizador;

• Deverá também ser possível visualizar a classificação dos diferentes clubes de uma competição, numa jornada indicada pelo utilizador (mesmo que seja anterior à atual); e

• Outas funcionalidades que considere relevantes.”

## Armazenamento

As possibilidades identificadas no que toca a armazenamento passaram por um sistema de base de dados embutido e um sistema *cloud*. O mecanismo escolhido foi ObjectBox, um sistema de armazenamento embutido próprio para dispositivos móveis e de IoT (Internet of Things), compatível com Dart e que tem na suma melhor característica a velocidade.

Contudo, embora solução escolhida tenha passado por um sistema nativo, é importante referir a existência de um mecanismo *cloud based* chamado de Firebase.

Outras boas opções também seriam sqflite e Hive, mas comparando as CRUD (Create, Read, Update e Delete) é possível ver que, no dado mais importante para este trabalho (leitura), é possível ver que existe um claro destaque do ObjectBox em relação aos restantes.

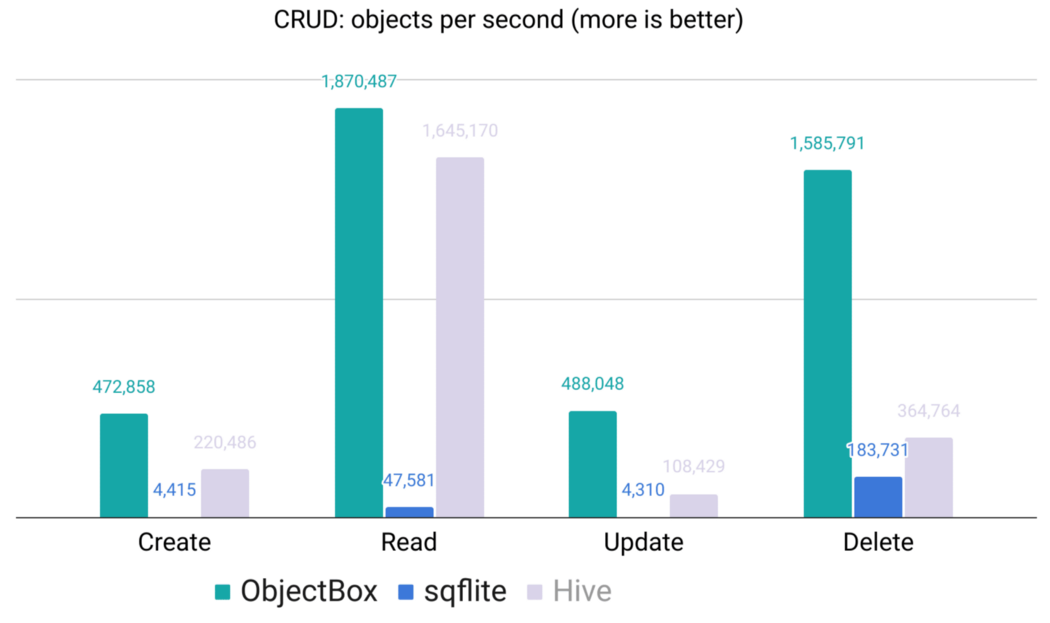


Figura - Objetos por segundo

As consultas à base de dados são bastante simples. Este processo passa pela criação de métodos personalizados dependendo das funcionalidades, no exemplo abaixo apresentamos o ‘’getEquipas” que, como o nome indica, faz uma seleção das equipas presentes na base de dados, posteriormente ordenando as mesmas pelo nome.

Uma imagem com texto

Descrição gerada automaticamente

Figura - Query ObjectBox

Sem inserções de dados, não consultas, as mesmas forem efetuadas da seguinte maneira:

Inicialmente, a classe ObjectBox é criada e envocada em toda a app assim que a mesma é iniciada. De seguida, são criadas as “boxes".

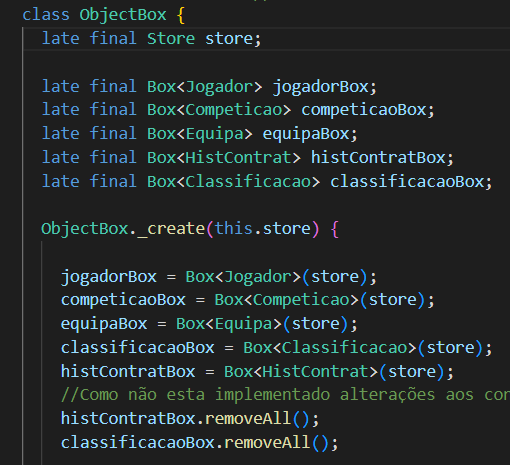


Figura - Criação de Boxes

As boxes estando criadas, cria-se os registos. Os campos destes registos estão ligados às classes previamente criadas.



Figura – Criação dos dados

Estando os registos criados, os mesmos são inseridos nas respetivas “boxes”.

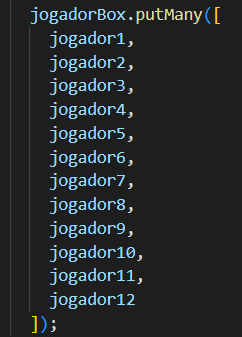


Figura - Inserção dos dados

Como verificação da existência de dados nas ‘’boxes’’ criou-se um ciclo de ‘’if’s’’ que adiciona os dados criados caso as mesmas estejam vazias.

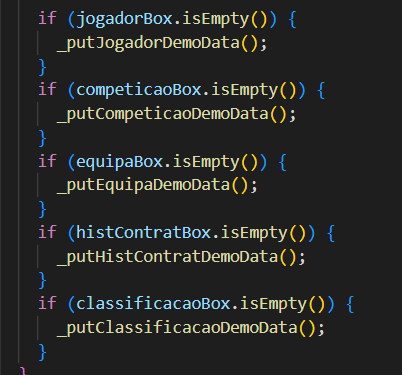


Figura - Verificação de dados nas boxes

## Interface

Como raiz de interface, criou-se uma AppBar onde se acede todas as áreas disponíveis. Estas áreas são Jogadores, Competições e Equipas.

Na área de jogadores é possível consultar os jogadores, os seus dados (idade e se o mesmo se encontra no ativo) inclusive a data do ultimo controlo antidoping e os seus contratos. Também é possível editar e eliminar estes registos de jogadores.

Na área das equipas, é possível consultar todas as equipas registadas na app, organizadas por ordem alfabética.

Por último, na área das Competições, é possível consultar as competições e suas classificações por jornada a partir de um widget DropDownMenu. Cada jornada tem associada a si um método de seleção dos pontos que cada equipa tinha numa dada jornada a partir de um switch case.

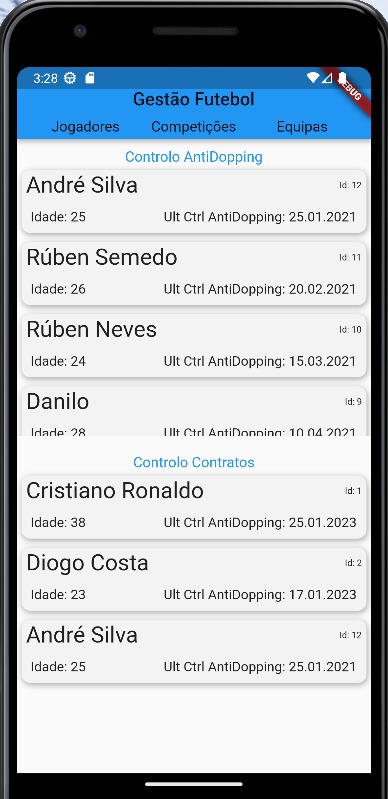


Figura - Controlo de Jogadores

Figura - Lista de Jogadores

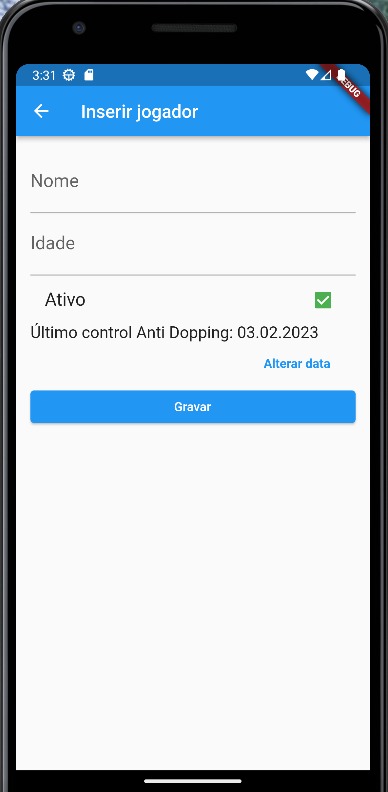
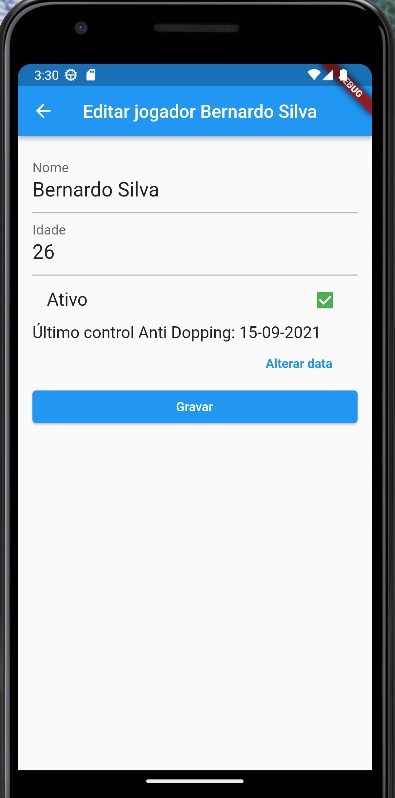


Figura - Editar Jogador

Figura - Inserir Jogador

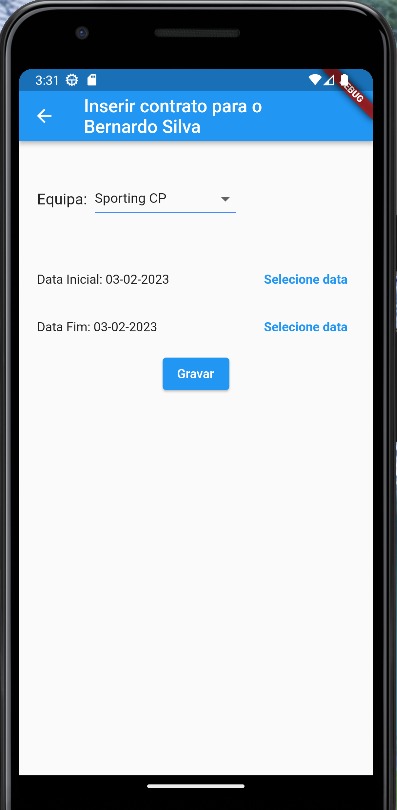
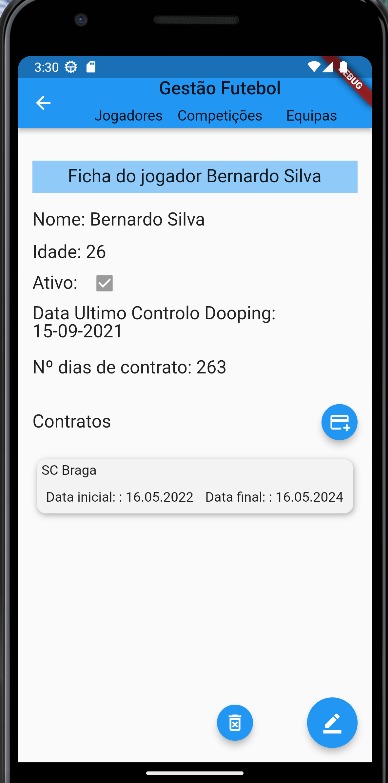


Figura - Ficha de Jogador

Figura - Inserir Contrato

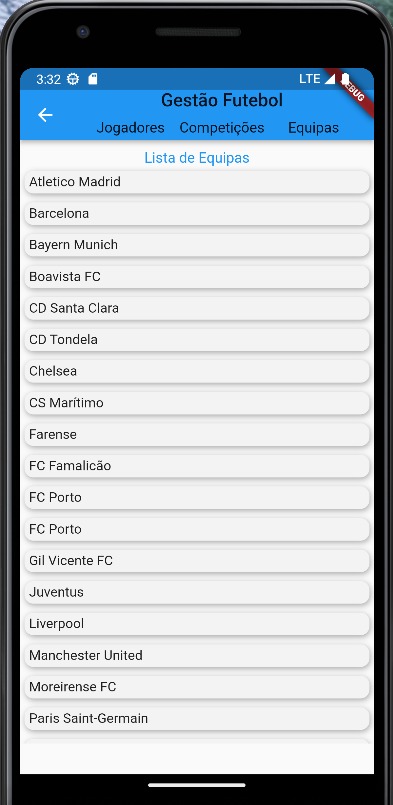
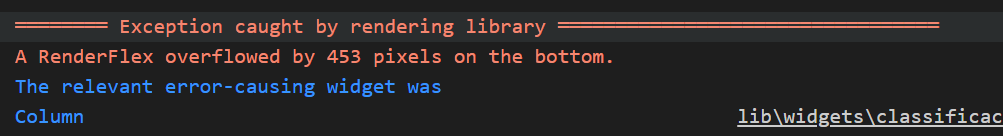


Figura - Lista de Equipas

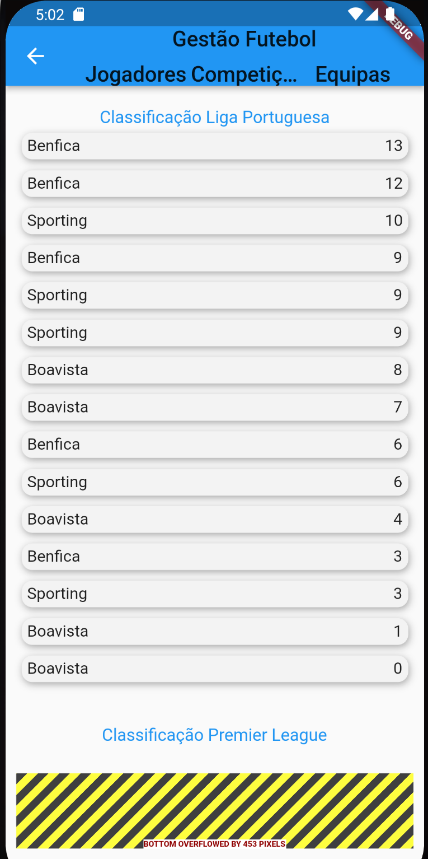
Figura - Classificações

# Erros Encontrados Mais Comuns

* RenderFlex Overflow



Neste erro em questão, uma coluna de widget de classificação tinha os ‘’scroll’’ máximo superior à capacidade do ecrã, resultando assim no ambiente abaixo.



A solução deste erro passou por envolver a coluna em questão com o Widget SingleChildScrollView, um widget que delimita o espaço onde é possível fazer ‘’scroll’’ ao criar uma espécie de caixa capaz de responder às necessidades do(s) widgets que o mesmo envolve.

* Bugs Atualização de dados no ObjectBox

Na imagem acima é possível identificar uma classificação um pouco estranha, a mesma é resultado de um erro de atualização das ditas “boxes”.

Como solução identificou-se uma sequência de comandos e um método que, logo após a inicialização da app, apaga as ‘’boxes”, as mesmas recebem os dados atualizados logo de seguida.

Numa fase inicial este erro dificultou bastante testes de consultas às ‘’boxes’’, embora existisse uma clara noção dos dados existentes nas mesmas e, também, de as queries executadas estarem corretas, os dados não eram selecionados corretamente visto que os mesmos desatualizados.

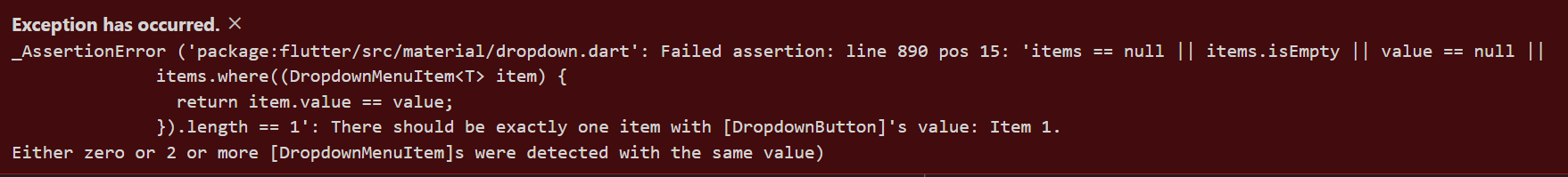
Os comandos mencionados acima são os seguintes:





O método mencionado é o seguinte:





Este erro ocorre ao popular os DropdownMenuItem quando todos os valores do mesmo não são únicos, isto é, assumindo que um dado DropdownMenuItem tem 4 opções cujos textos são “A”, “B”, “C” e “D” os valores, respectivamente, são “A”, “A”, “C” e “D”. Como existem duas opções com o mesmo valor o erro ocorre.

# Conclusão

Em resumo, o uso da tecnologia Flutter para o desenvolvimento de aplicações móveis trouxe consigo uma série de benefícios para a criação de interfaces atrativas e intuitivas. Com a utilização do ObjectBox como sistema de gestão de banco de dados, é possível aprimorar ainda mais a eficiência e a segurança na manipulação de dados.

O desenvolvimento deste projeto permitiu explorar as vantagens de utilizar o Flutter juntamente com o ObjectBox, mostrando como essas tecnologias podem ser combinadas para produzir resultados notáveis em termos de qualidade e performance. Além disso, a combinação desses dois elementos oferece uma solução completa para o desenvolvimento de aplicações móveis, tornando o processo mais fácil e eficiente e, assim, glorificando a essência de simplicidade que está na base do flutter.

Em conclusão, o projeto mostra a importância de utilizar tecnologias avançadas como o Flutter e o ObjectBox para garantir a qualidade e a eficiência no desenvolvimento de aplicativos móveis.

# Bibliografia

objectbox | Dart Package. (n.d.). Dart Packages. Retrieved February 2, 2023, from https://pub.dev/packages/objectbox

ObjectBox Queries - ObjectBox Docs. (2020). Objectbox.io. https://docs.objectbox.io/queries

Create a Flutter App Using ObjectBox as a Local Database | Event Management App. (n.d.). Www.youtube.com. Consultado em 20 janeiro , 2023, disponível em: