PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO

TransferWindow

Aplicativo IOS para construção de um ambiente digital para o mercado de negociação de jogadores para clubes de futebol e empresários do setor

Gustavo Oliveira de Mendonça

PROJETO FINAL DE GRADUAÇÃO

CENTRO TÉCNICO CIENTÍFICO - CTC
DEPARTAMENTO DE INFORMÁTICA
Curso de Graduação em Engenharia da Computação

Rio de Janeiro, Julho de 2022

Gustavo Oliveira de Mendonça

TransferWindow

Aplicativo IOS para construção de um ambiente digital para o mercado de negociação de jogadores para clubes de futebol e empresários do setor

Relatório de Projeto Final, apresentado ao curso de Engenharia da Computação da PUC-Rio como requisito parcial para a obtenção do título de Engenheiro de Computação.

Orientador: Marcos Vianna Villas

Rio de Janeiro,
Julho de 2022.

Agradecimentos

Agradeço aos meus pais Divaldo e Marcia e também aos familiares, por todo encorajamento que recebi durante toda minha caminhada. Sempre me alertando para sair da zona de conforto, buscar novos desafios, me acompanhando e me incentivando a superar as dificuldades que eu encontrei ao longo da minha vida.

Agradeço à minha namorada, Luiza Carpilovsky, pelo apoio, e por me ajudar a crescer como ser humano, estudante e profissional. Esse apoio foi fundamental para conquistar tudo que consegui e me tornar a pessoa que sou hoje.

Agradeço ao meu colega de equipe, Fred Gall, por trabalharmos juntos no pensamento e desenvolvimento da ideia, buscando levar esse projeto adiante para torná-lo realidade dentro do mercado esportivo.

Agradeço ao meu orientador, Marcos Vianna Villas, por todo apoio, atenção e disponibilidade. Sem ele o projeto teria sido muito mais complexo e essa jornada mais árdua.

Resumo

Oliveira de Mendonça, Gustavo. Vianna Villas, Marcos. **TransferWindow** - Aplicativo IOS para construção de um ambiente digital para o mercado de negociação de jogadores para clubes de futebol e empresários do setor. Rio de Janeiro, 2022. 76p. Projeto Final de Graduação - Departamento de Informática. Pontifícia Universidade Católica do Rio de Janeiro.

O atual projeto de graduação tem como objetivo implementar uma aplicação de ambiente digital para o mercado de negociação de jogadores para clubes de futebol. Esta proposta foi inspirada no trabalho do ex-aluno, Frederico de Freitas Martins Gall, que apresentou em sua monografia de bacharelado em Administração Pública e de Empresas na Fundação Getúlio Vargas, no ano de 2021, um plano de negócios para o *TransferWindow*. Em sua monografia, Fred buscou como principal objetivo, a viabilidade financeira da construção de uma aplicação digital para a negociação de jogadores, que através da análise do mercado de futebol brasileiro pudesse justificar a opção de aquisição de profissionais do futebol.

O projeto ora apresentado busca desenvolver, detalhar e materializar a solução como um todo. Informações utilizadas neste documento, a partir da pesquisa de TCC serão devidamente citadas no transcorrer da apresentação dos tópicos principais e na bibliografia pertinente. Esse documento, por enquanto, não se encontra disponível online. Assim sendo, o projeto terá como objetivo principal desenvolver uma solução voltada para o mercado de transferência entre clubes, de atletas de futebol no Brasil.

Procurando atender as necessidades dos três stakeholders deste mercado, quais sejam: 1-Clubes; 2-Empresários e, 3-Jogadores, a *TransferWindow* busca profissionalizar e agilizar o processo de negociação dos profissionais do futebol, evitando a excessiva quantidade de terceiros envolvidos nos processos de contratação e tornando essa operação menos exaustiva. Com a intenção de aumentar o número de transferências realizadas entre clubes, a *TransferWindow* buscará disponibilizar o máximo de informações e utilizará a tecnologia para tornar esse processo mais transparente e menos complexo tanto para o clube quanto para os atletas através de seus empresários.

Palavras-chave

futebol, integração, mercado digital, IOS, swift

Abstract

Oliveira de Mendonça, Gustavo. Vianna Villas, Marcos. **TransferWindow** - IOS application for building a digital environment for the player trading market for football clubs and entrepreneurs in the sector. Rio de Janeiro, 2022. 76p. Projeto Final de Graduação - Departamento de Informática. Pontifícia Universidade Católica do Rio de Janeiro.

The current graduation project aims to implement a digital environment application for the player trading market for football clubs. This proposal was inspired by the work of former student, Frederico de Freitas Martins Gall, who presented in Public and Business Administration at Fundação Getúlio Vargas, in the year 2021, a business plan for TransferWindow. In his monograph, Fred sought as his main objective, the financial viability of building a digital application for the negotiation of players, which, through the analysis of the Brazilian football market, could justify the option of acquiring football professionals.

The project presented seeks to develop, detail and materialize the solution as a whole. Information used in this document, based on the TCC research, will be duly cited during the presentation of the main topics and in the relevant bibliography. This document is currently not available online. Therefore, the project's main objective will be to develop a solution aimed at the transfer market between clubs, for soccer athletes in Brazil.

Seeking to meet the needs of the three stakeholders of this market, namely, 1- Clubs; 2-Businessmen and 3-Players, TransferWindow seeks to professionalize and streamline the negotiation process of football professionals, avoiding the excessive amount of third parties involved in the hiring processes and making this operation less exhausting. With the intention of increasing the number of transfers carried out between clubs, TransferWindow will seek to provide as much information as possible and will use technology to make this process more transparent and less complex, both for the club and for the athletes through its agents.

Keywords

football, integration, digital market, IOS, swift

Sumário

Tabela de Figuras

1.Introdução

O futebol é o esporte mais popular e um dos principais negócios no mundo. Apesar de não se saber determinar ao certo quando surgiu e quem foi o seu inventor, dado que existem registros de rituais parecidos na China antiga (2.600 a.C.), civilizações gregas e romanas (753 a.C) e América pré-colombiana (séc. XV), foi a partir do século XIX, mais precisamente na Inglaterra, por volta de 1863, que o jogo foi regulamentado pela *Football Association*. O esporte chegou ao Brasil em 1894, através de Charles Miller, e rapidamente se difundiu pelo país até alcançar o posto de potência mundial, a partir da década de 50.

Com o passar do tempo, passou a liderar estatísticas de preferência de atividade física da população e ao mesmo tempo que se desenvolvia e aumentava seu número de fãs ao redor do mundo, cresceu também o potencial econômico com as oportunidades de negócio que giravam dentro e em torno do futebol. Esse potencial se mostra refletido nos números disponibilizados pela FIFA[1], em que, no mundo, existem cerca de 270 milhões de pessoas ativas no futebol.

Em se tratando de movimentação financeira e volume de capital, de acordo com os últimos levantamentos feitos pela Delloite[2], o futebol movimentou um valor estimado em R$ 200 bilhões de reais em 2019, com base nos informativos divulgados pelo Banco Mundial. Já no Brasil, o valor movimentado por essa indústria corresponde a aproximadamente 52,9 bilhões, o que representa 0,72% do PIB brasileiro [3], e foi responsável por gerar mais de 156 mil empregos.

Atualmente, a grande maioria das negociações que são feitas pelo mercado de transferências de atletas entre clubes de futebol são realizadas por meio de contatos, sejam eles diretos ou indiretos, entre alguns clubes ou por meio de empresários que participam da negociação como facilitadores ou intermediadores. Segundo o relatório de intermediários da CBF[4], o valor dessas intermediações chegou a custar para os clubes mais de R$ 160 milhões de reais. Porém, muitas delas acontecem para benefício próprio deste terceiro, que por diversas vezes acabam transmitindo informações imprecisas aos envolvidos e cobrando valores superiores ao que seria adequado, caso os clubes tivessem alguma forma de contato direto com outros clubes ou com empresários do ramo.

Portanto, partindo da necessidade de tornar as transferências mais profissionais, diretas e menos burocráticas, surgiu a ideia de criar um aplicativo que atue como um facilitador nesse meio, servindo como uma ponte de integração entre clubes e permitindo uma negociação mais clara, direta e justa. A importância desse tema está no fato de que o mercado de transferências movimenta um grande volume de recursos financeiros, mas não apresenta muitas soluções que atuam como uma fusão de marketplace e aplicativo de relacionamento e principalmente por não existir esse tipo de solução difundida no mercado brasileiro de futebol.

2.Situação Atual

O aplicativo de uma maneira geral é uma grande oportunidade, pois se refere a uma demanda que não foi totalmente atendida, em um mercado que vem crescendo de forma exponencial ao longo dos anos. Por ser algo inovador no mercado esportivo, existem poucos concorrentes diretos, os quais possuem diversas limitações, principalmente no que tange às interfaces de usuário, números de clubes na plataforma, língua e aspectos próprios, característicos do irreverente futebol brasileiro.

A primeira aplicação que deve ser citada é a **TransferRoom**[5], e pela similaridade dos nomes, é possível perceber que o objetivo é ser concorrente no mesmo setor em que ela se insere, na disputa pelo protagonismo. A **TransferRoom** atua no mercado desde 2017, e de maneira simplificada, ela faz todo o trabalho de pré-scouting[1], identificação de necessidades do clube, identificação de jogadores potenciais e negociação de termos. Segundo uma matéria feita pelo Globo Esporte[6], a empresa possui mais de 550 clubes como usuários, de 78 ligas e em mais de 40 países diferentes, trabalhando com um modelo de assinatura anual de £ 12 mil, o equivalente a aproximadamente R$75 mil na cotação atual(2022) e totalizando um faturamento anual de aproximadamente R$ 50 milhões de reais.

A **TransferRoom** exerce uma boa influência no mercado esportivo Europeu, onde concentra grande parte de sua base de clientes, porém devido a fatores limitadores como valor elevado de assinatura , linguagem, funcionalidades disponíveis e plataforma, teve baixa adesão no mercado sul-americano e principalmente, mais especificamente no Brasil. Além disso, outro problema relatado por profissionais do esporte é que as informações necessárias que são divulgadas para ajudar na identificação de necessidades da aquisição de um jogador, acabam expondo as táticas e estratégias aplicadas, além de expor as fraquezas e defasagens que o clube ou empresário ou jogador possa estar passando.

Em seguida temos como potencial concorrente a **WyScout**[7]. Este produto foi lançado no mercado em 2004, e atualmente, possui a tecnologia mais utilizada no mundo quando o assunto é scouting, match e análise de desempenho. Fornecem um conjunto de produtos que reúnem vídeos, dados e informações sobre cada jogador, competição ou jogo. Possuem diversos serviços em alguns mercados, sendo um deles a negociação de atletas entre clubes. Atuam com segmentos que o *TransferWindow* pretende atuar, porém o foco é apoiar a busca de talentos através de análises de performances.

A **WyScout** declara possuir mais de 42 mil clubes registrados, sendo 89 das 5 melhores Ligas do mundo. Apresentam um modelo de assinatura anual com 3 tipos de planos, independentemente do segmento: Bronze, Silver e Extra, variando de R$ 1,7 mil a R$ 4,0 mil por ano, mais possíveis extras caso ocorra a inclusão de benefícios. Apesar de ser a maior plataforma atualmente disponível no mercado, ainda não é reconhecida internacionalmente como adequada para tratar de negociações e transferências entre clubes. Além disso, possui algumas limitações em

[1] Processo de observação e análise que tem como objetivo recolher o máximo de informações de algum indivíduo.

relação aos serviços de negociação, apresentando um fluxo de tela pouco claro e intuitivo, um layout e UI pouco atraente e funcionalidades limitadas para tipos de usuários, como jogadores, que não conseguem uma via fácil de contato com clubes ou outras entidades.

Já no mercado brasileiro, a única plataforma atuante é o **Caça Atletas**[8]. Foi fundada com intuito de auxiliar atletas e profissionais do esporte a conseguirem encontrar oportunidades de trabalho. Também passaram a atuar no mercado de transferências, porém com maior enfoque em clubes com menos expressão, divisões inferiores e amadores.

Trabalha principalmente com dois segmentos que a *TransferWindow* se propõe a atender, que são os atletas e os empresários. Possui registrados aproximadamente 250 clubes e empresários, e mais de 8 mil atletas. O modelo de precificação não é divulgado de forma pública. Entretanto, de todas as plataformas citadas anteriormente, é a que tem menor relevância em termos comparativos. Isso porque além da grande limitação da plataforma, seja tecnológica, visual ou intuitiva, possui pouca representação no Brasil devido a baixa quantidade de serviços realizados pela plataforma ou volume de operações realizadas.

Existem alguns problemas visíveis nas soluções existentes e isso abre uma janela importante de oportunidade. Tendo em vista que a maioria das aplicações existentes são estrangeiras, não é aderente às necessidades reais dos clubes, jogadores e empresários. Além disso, eventualmente o custo pode não se apresentar como impedimento para usuários mais estruturados financeiramente, como os clubes da série A, porém se tornam proibitivos para clubes de Série B, C e D, além de jogadores e empresários que estão dispersos no vasto território nacional e que são demandantes de uma solução eficiente e eficaz para viabilizar seus projetos de crescimento financeiro e profissional.

Um ponto de extrema relevância na especificação do produto *TransferWindow,* em atual estágio de desenvolvimento, é a verificação de uma característica exclusiva de que no Brasil os empresários, por força contratual, acabam possuindo forte poder de decisão e controle do mercado. Sendo assim, foi importante incluir na plataforma esse tipo de cliente, dado que todo jogador precisa de um empresário. Além disso, existem muitos outros jogadores que estão fora de clubes e que precisam de visibilidade e podem gerar interesses aos clubes e que precisam de uma plataforma para terem visibilidade e valor.

Outro problema percebido e já comentado anteriormente se refere às informações divulgadas nas aplicações concorrentes, que acabam entregando decisões táticas e estratégicas dos clubes, devido ao fato de não promover a correta ação de sigilo nas etapas relevantes das negociações .

Numa entrevista realizada pelo Globo Esporte[9] com Alexandre Pássaro, gerente do São Paulo, ele aborda sua preocupação de que ***"Vai atrapalhar muito se eu começar a divulgar minhas informações, ou minhas estratégias, e este mercado incluir não só os rivais, mas os times que competem comigo".*** Portanto, um dos objetivos é fornecer uma solução que permita

proteger os interesses dos clubes através do anonimato, enquanto a negociação estiver nas etapas iniciais.

3.Propostas e Objetivos do Trabalho

O principal objetivo deste trabalho é propor alternativas para desenvolver um produto com facilidades que seja capaz de suprir os pontos negativos encontrados em outras soluções, em específico, aquelas citadas anteriormente. A busca é tornar o aplicativo mais atraente para os clientes, além de proporcionar uma experiência mais completa para seus usuários.

Pensando nisso, foi decidido que para a execução desse trabalho seria necessário recorrer ao uso de uma linguagem que facilitasse o desenvolvimento em um ambiente que fosse ao mesmo tempo atraente e eficiente. O **Swift[10]**[2] atende esses requisitos, já que funciona como front-end. Para esse desenvolvimento, pretendo utilizar o **Firebase[11]** como arquitetura serverless e na seção 7.1 do Projeto essa arquitetura será apresentada juntamente com o motivo da escolha deste provedor.

Diferentemente dos seus concorrentes estrangeiros, mais especificamente a TransferRoom, a *TransferWindow* pretende englobar na sua aplicação não apenas os empresários que já possuem jogadores em clubes como também aqueles que ainda não os possuem. A incorporação desses atores é de extrema relevância por serem os elementos chaves para o processo de contratação de um atleta e, por conseguinte, a sua presença na plataforma facilitará a difusão e aceitação deste aplicativo na comunidade.

Entendendo que a dependência de redes de contato e intermediários é uma das grandes barreiras para a evolução de um processo de contratação, a *TranferWindow* foi criada com o propósito de permitir que as transferências de atletas sejam realizadas de forma mais profissional, eficiente, transparente e ágil. Ao atender essa demanda, fica claro que além do grande potencial financeiro, a solução tem como princípio atuar como potencializador para melhorar e transformar a indústria do futebol.

A proposta é desenvolver um aplicativo capaz de integrar todos os 3 segmentos que a *TransferWindow* está se propondo a atender, que são os Clubes das séries A, B e C do Campeonato Brasileiro, os empresários e empresas de agenciamento e os atletas. Após a nossa consolidação no mercado nacional, o foco será atuar no mercado sul-americano e europeu, principalmente nos países que costumam contratar atletas brasileiros.

Os benefícios da aplicação para os segmentos são:

- Maior agilidade e segurança nas buscas por novos profissionais e tempo reduzido para o início das negociações;
- Facilidade para qualificar o mercado de transferências com as oportunidades identificadas;
- Mapeamento do mercado com a utilização de relatórios consolidados e informações valiosas sobre o mercado;

[2] Swift é uma linguagem de programação desenvolvida pela Apple para desenvolvimento no iOS, macOS, watchOS, tvOS e Linux.-

- Aumento das negociações e criação de contatos com um ambiente integrado e de fácil utilização;

Em relação à questão financeira, inicialmente o app será lançado sem custo, com o objetivo de atrair os clubes e os empresários para movimentar e popularizar o aplicativo. Posteriormente, passaria a haver cobrança por assinatura anual ou mensal, mas o objetivo seria uma taxa inferior se comparada às soluções estrangeiras, por se tratar de uma mão de obra nacional, cuja remuneração é inferior aos custos de serviços no mercado internacional. Outro importante aspecto financeiro é a possibilidade de obtenção de receita que seria através da utilização de anúncios, e comporia o modelo de negócios. Por ser uma plataforma que possuirá uma página de Feed, possui uma estrutura que se encaixa com esse tipo de monetização.

4.Método de desenvolvimento de Software

4.1 Cronograma

Enquanto que o método é um modo de colocar em prática uma determinada ação, a metodologia busca desenvolver o caminho que será seguido na execução do projeto, a ordem em que ocorrerá e implementar o planejamento do que precisa ser executado.

Para este projeto, em consonância com a declaração acima, foram elaboradas as etapas de definição de requisitos, mais especificamente a parte que trata de **Elicitação de requisitos** e **Especificação**. Na elicitação, foram classificados os requisitos funcionais e não funcionais, realizadas entrevistas com usuários com o objetivo de entender suas necessidades e demandas e, compreender o estado-da-arte (**Situação Atual**), tanto no Brasil como no exterior (parte do estado-da-arte já foi abordado anteriormente nos tópicos de Motivação e Situação Atual). Na tabela abaixo é possível verificar como o tempo e as tarefas foram divididas para os tópicos citados anteriormente.

2021	Outubro semana1	Outubro semana2	Outubro semana3	Outubro semana4	Novembro semana1	Novembro semana2	Novembro semana3	Novembro semana4
Definição do Estado de Arte	X	X						
Requisitos Funcionais		X	X					
Requisitos Não Funcionais				X	X			
Entrevistas			X	X			X	X
Diagrama de Classes					X	X	X	
Diagrama de Casos de Uso						X	X	X

Figura 1 - Cronograma Outubro/Novembro 2021

Em seguida foi executada a elaboração do **Projeto**, **Construção** da aplicação e **Testes** com usuários. Para a realização destas etapas, foi utilizado uma junção de metodologias que será apresentado posteriormente, com o intuito de aprimorar a estrutura de organização do projeto como um todo. Na elaboração do Projeto foi definido e implementado uma linguagem de notação, provavelmente UML. Na construção, será definido a linguagem usada na implementação, como a solução é para aplicativos, Swift talvez seja a melhor opção. E por fim, na etapa de testes, haverá a verificação de qualidade do software. Nas tabelas abaixo, é possível verificar como o tempo e as tarefas foram divididas para os tópicos citados anteriormente.

2022	Março semana1	Março semana2	Março semana3	Março semana4	Abril semana1	Abril semana2	Abril semana3	Abril semana4
Diagrama de Casos de Uso	X							
Descrição dos Casos de Uso		X	X	X				
Prototipagem				X	X	X	X	X

Figura 2 - Cronograma Março/Abril 2022

2022	Maio semana1	Maio semana2	Maio semana3	Maio semana4	Junho semana1	Junho semana2	Junho semana3	Junho semana4
Prototipagem	X							
Padrão de Projeto		X	X					
Front-end				X	X			
Arquitetura						X		
Construção		X	X	X	X	X	X	X
Documentação		X		X			X	X

Figura 3 - Cronograma Maio/Junho 2022

2022	Julho semana1	Julho semana2
Construção	X	X
Documentação	X	X
Testes	X	X
Dicionário de Atributos	X	X

Figura 4 - Cronograma Julho 2022

4.2 Metodologia Ágil

Conhecer as melhores metodologias de desenvolvimento de software é imprescindível para que a criação de qualquer sistema seja bem-sucedida. Cada software possui as suas próprias características e a escolha de determinada metodologia precisa atender a necessidades

específicas. Além disso, vêm se tornando uma exigência dos clientes que, em pouco tempo de projeto, já seja possível enxergar resultados, portanto uma boa organização de projeto é indispensável para a evolução da solução.

Foi escolhido para esse projeto o uso da metodologia de desenvolvimento ágil. Na busca de evitar o excesso de uso do tempo para levantamento de requisitos, mapeamento de necessidades, e outras demandas que geram uma demora do entregável, foi optado por usar uma mistura de Scrum e Kanban. Nesse projeto o uso dessas metodologias foi adaptado, mas como parte do objetivo geral, permitiu o foco na funcionalidade do aplicativo em si e possibilitou que acontecessem vários pequenos avanços na implementação em um curto espaço de tempo

Combinando as melhores características dessas duas metodologias para esse projeto, foi possível aproveitar do **Scrum[12]** a organização através de **Sprints**, que são intervalos de tempo onde são definidos **entregáveis** para determinado período, e no caso desse projeto cada sprint tem duração de **duas semanas**. Outra importante área da gestão é o **planejamento**, servindo para decidir as metas de desenvolvimento do sprint e definir as histórias de uso[3] a partir de um backlog do produto.

Já no **Kanban[13]** foi possível utilizar a visualização do fluxo de trabalho através da definição de microtarefas, que auxiliam no acompanhamento do desenvolvimento e na evolução da aplicação. Também foi utilizado a definição de limites de WIP[4] na tentativa de minimizar atividades em progresso, mas que num mundo real seriam produtos semi acabados que representam um custo, mas sem possibilidade de lucro devido a falta de conclusão. Portanto, a partir dessa breve explicação dos métodos utilizados, será introduzido a seguir, a organização real do projeto.

[3] História de uso é uma ferramenta que auxilia na mudança de foco de detalhar requerimentos de sistema para conversar sobre eles.
[4] Na cadeia de suprimentos, o termo Work in Progress é usado para definir um produto semi-acabado

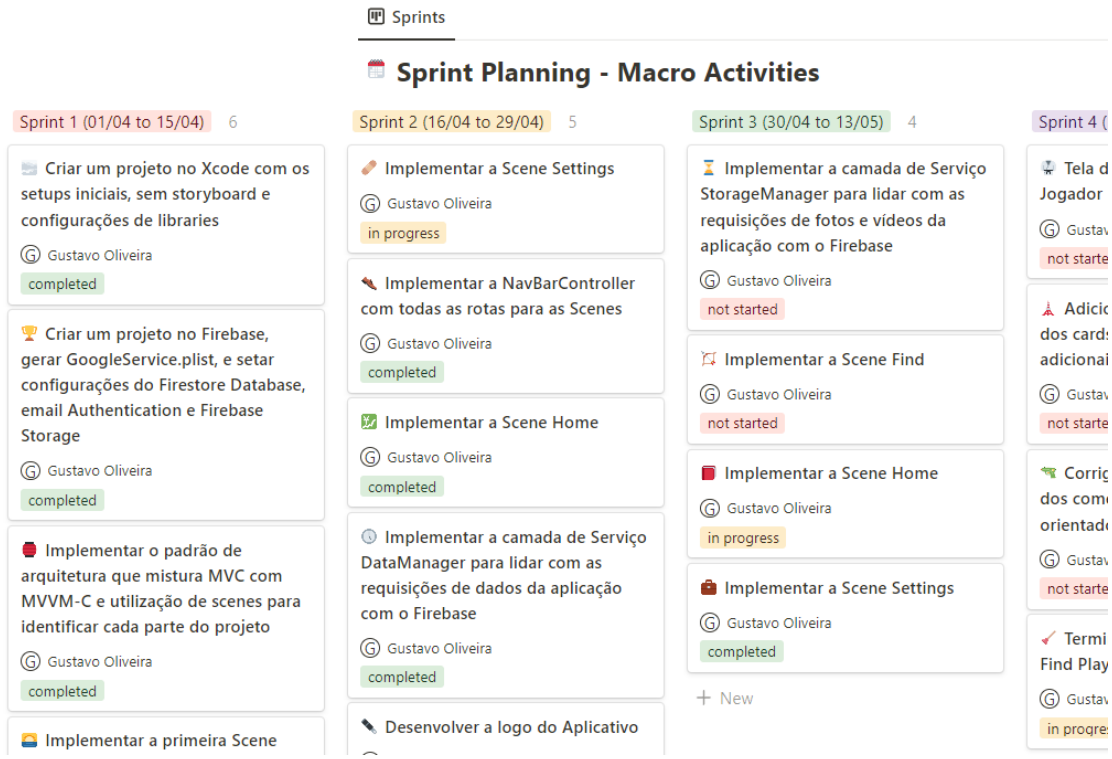

Figura 5 - Exemplo do quadro de Macroatividades do TransferWindow

Como é possível ver no trecho do quadro de macroatividades utilizado como exemplo, na figura 1 acima, para cada sprint foi definido um conjunto de macrotarefas que deveriam ser executadas durante o intervalo de tempo definido. Cada macrotarefa foi definida de forma bem abrangente e caso o tempo estabelecido não tenha sido suficiente para a conclusão de todos os itens, um novo planning era realizado, adicionando a tarefa pendente para o próximo ciclo, e uma nova tentativa de planejar um escopo compatível com o tempo/capacidade de implementação era realizado.

Porém, ao longo de um sprint, identificar o progresso de uma atividade apenas com o seu macro card torna-se complicado, uma vez que o rastreio de evolução da tarefa encontra-se inviabilizado. Portanto, para facilitar o entendimento do processo de evolução de cada macroatividade, foi estabelecido outro quadro contendo as microatividades que estavam intrínsecas nesse processo. Além disso, essa organização permitiu também visualizar o fluxo de trabalho que estava acontecendo e manter um controle das datas de entrega. Nas figuras 2 e 3 a seguir, é possível checar como a divisão desses objetivos foi feita.

Figura 6 - Exemplo do quadro de Microatividades do TransferWindow

As macrotarefas definidas no quadro anterior foram divididas em colunas que basicamente são desmembradas em função da tarefa dentro do código, ou seja, se o próximo objetivo for implementar um botão, significa que é uma tarefa de front-end, e portanto vai ser separado na coluna de *Views*. Naturalmente, dentre os atributos criados existe o progresso da atividade e onde ela está localizada (*Root, Folder* e *File*). Isso facilita na hora de fazer uma varredura por uma determinada implementação, já que tudo que foi realizado está devidamente organizado.

Na coluna *Settings* são criados todos os cartões referentes a configurações do ambiente de desenvolvimento Xcode, importação de pacotes e bibliotecas e arquivos de configuração do banco de dados Firebase. Em *Design* é representado tudo que se refere a ideias de mudança da interface, criação de efeitos, gestures, ícones do aplicativo e logo da marca. Nas colunas de *Model*, *View* e *Controller* estão os cartões que possuem tarefas posicionadas em cada uma das três camadas. No *Handle Erros*, é tratado a recuperação da aplicação caso alguma requisição não ocorra como planejado.

Na coluna *Firebase* estão as atividades que envolvem a comunicação com o serviço de banco de dados. No *Code Review* estão as tarefas que cuidam dos bugs ou más implementações que precisam ser revisadas futuramente. A coluna *Alert* informa os alertas que precisam ser implementados. Finalmente o *Testing* é quem cuida dos testes unitários e como a aplicação reage a determinados casos. Na figura 3 a seguir, é possível visualizar uma segunda parte do quadro de microatividades.

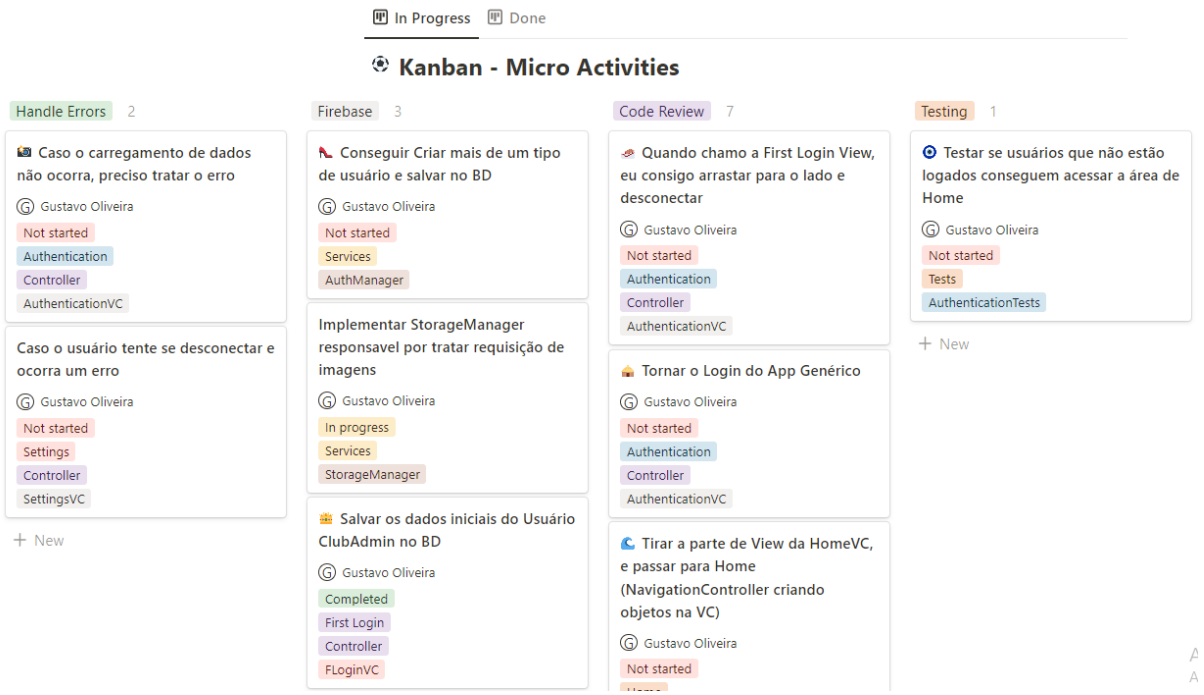

Figura 7 - Continuação Quadro de Microatividades do TransferWindow

5.Requisitos

O levantamento de requisitos é uma das partes mais importantes para o desenvolvimento de qualquer software e é responsável por definir os serviços que o sistema oferecerá. Neste projeto sua elaboração foi feita em conjunto com a prototipagem, já que a visualização do projeto depois da validação da arquitetura da informação e da interatividade com os elementos da interface possibilitou um levantamento de requisitos mais preciso. A figura 3 exibida a seguir mostra um trecho dos requisitos funcionais elaborados, regras de negócio e suas descrições.

Durante o levantamento dos requisitos, é de fundamental importância atentar em buscar informações do cliente para construir um software, organizar o projeto antes de iniciar qualquer implementação e entregar um produto de qualidade que atenda de fato às necessidades do usuário. Este modelo, juntamente com o diagrama de casos de uso e as entrevistas piloto que serão apresentados posteriormente, possibilitou entender antes de começar o desenvolvimento da aplicação, quais módulos seriam necessários para o projeto.

Identificador	Categoria	Tipo	Descrição
RF01	Cadastro do usuário Clube	Funcional	Deve ser possível criar um novo usuário Clube
RF02	Cadastro do usuário Empresário	Funcional	Deve ser possível criar um novo usuário Empresário
RF03	Cadastro do usuário Clube	Funcional	Não deve ser possível criar uma conta diretamente pelo aplicativo. Usuários clubes são convidados diretamente pelo email oficial.
RNF01	Cadastro do usuário Empresário	Regra de Negócio	Não deve ser possível cadastrar uma conta que não tenha cadastro na CBF.
RNF02	Cadastro do usuário Empresário	Regra de Negócio	Não deve ser possível cadastrar dois usuários com o mesmo ‘email'.
RNF03	Cadastro do usuário Empresário	Regra de Negócio	Não deve ser possível cadastrar dois usuários com o mesmo ‘nome’.

RNF04	Usuários (Clubes, Empresários e Subcontas)	Regra de Negócio	Não deve ser possível cadastrar/modificar um usuário usando uma 'senha' que não tenha tamanho mínimo de 8 caracteres, caracteres especiais, letras maiúsculas e minúsculas e números.
RF04	Autenticação do usuário Clube	Funcional	Deve ser possível autenticar um usuário Clube.
RNF05	Autenticação do usuário Clube	Regra de Negócio	Deve ser possível autenticar mais de uma conta dentro do usuário Clube.
RNF6	Autenticação do usuário Clube	Regra de Negócio	Não deve ser possível autenticar um sub-usuario clube que não esteja ativo na conta principal
RNF07	Autenticação do usuário Clube	Regra de Negócio	Não deve ser possível autenticar um usuário clube sem preencher 'email/nome de usuário' e 'senha'
RNF08	Autenticação do usuário Clube	Regra de Negócio	Não deve ser possível autenticar um usuário clube que não esteja cadastrado no banco de dados
RNF09	Autenticação do usuário Empresário	Regra de Negócio	Não deve ser possível autenticar um usuário empresário sem preencher ‘username’ e ‘password'
RNF10	Autenticação do usuário Empresário	Regra de Negócio	Não deve ser possível autenticar um usuário empresário que não

			esteja cadastrado no banco de dados
RNF11	Autenticação do usuário Empresário	Regra de Negócio	Caso um o nome ou o email seja criado ou alterado, é necessário uma validação de unicidade no sistema.
RF05	Autenticação do usuário Clube	Funcional	Toda senha/dado sensível que for armazenada no banco de dados deve ser criptografada
RF06	Autenticação do usuário Empresário	Funcional	Toda senha/dado sensível que for armazenada no banco de dados deve ser criptografada
RF07	Autenticação do usuário Clube	Funcional	Todos os dados sensíveis que forem enviados via requisições HTTP devem ser criptografados
RF08	Autenticação do usuário Empresário	Funcional	Todos os dados sensíveis que forem enviado via requisições HTTP devem ser criptografados
RF09	Exclusão de um usuário Clube	Funcional	Um usuário clube deve ser capaz de excluir seu cadastro
RNF12	Exclusão de um usuário Clube	Regra de Negócio	Caso um usuário clube seja excluído, todas as demais sub-contas devem ser excluídas em cascata
RNF13	Exclusão de um usuário Empresário	Regra de Negócio	Caso um usuário empresário seja excluído, todas as

			demais sub-contas devem ser excluídas em cascata
RF10	Exclusão de um usuário Empresário	Funcional	Um usuário empresário deve ser capaz de excluir seu cadastro
RNF14	Exclusão de um sub-usuário	Regra de Negócio	Uma subconta deve poder ser excluída pelo portador, ainda que esteja como desativado na conta principal
RNF15	Exclusão de um usuário Clube	Regra de Negócio	Uma mensagem de confirmação deve ser exibida antes da exclusão da conta. Após a mensagem a conta ficará 48hrs suspensa até a efetiva exclusão de todos os dados
RNF16	Exclusão de um usuário Empresário	Regra de Negócio	Uma mensagem de confirmação deve ser exibida antes da exclusão da conta. Após a mensagem a conta ficará 48hrs suspensa até a efetiva exclusão de todos os dados.
RF11	Módulo de Buscas do Clube	Funcional	Deve ser possível procurar dentro da busca clubes, empresários e jogadores cadastrados e com perfil aberto.
RF12	Módulo de Buscas do Empresário	Funcional	Deve ser possível procurar dentro da busca clubes, empresários e jogadores cadastrados e com perfil aberto.

RF13	Módulo de Buscas do Clube	Funcional	Deve ser possível salvar as procuras recentes nas de sugestões de busca
RF14	Módulo de Buscas do Empresário	Funcional	Deve ser possível salvar as procuras recentes nas de sugestões de busca
RNF17	Protocolo para cadastrar Jogadores	Regra de Negócio	Para cadastrar um jogador, é preciso preencher características específicas que facilitem sua classificação (pé dominante, idade, altura..)
RNF18	Anúncio de Jogadores do Clube	Regra de Negócio	Um clube deve ser capaz de anunciar, seja de forma anônima ou não, jogadores que estão disponíveis para transferência, seja para compra ou empréstimo.
RNF19	Anúncio de Jogadores do Empresário	Regra de Negócio	Um clube deve ser capaz de anunciar, seja de forma anônima ou não, jogadores que estão disponíveis para transferência, seja para compra ou empréstimo.
RF15	Favoritos do Clube	Funcional	Um clube deve ser capaz de favoritar jogadores (disponíveis ou não para compra/venda), empresários e ofertas.

RNF20	Favoritos do Clube	Regras de Negócio	Um clube deve ser capaz de organizar cada tipo de favorito em uma tabela própria e poder arrumá-la com os critérios que preferir.
RNF21	Favoritos do Empresário	Regras de Negócio	Um clube deve ser capaz de organizar cada tipo de favorito em uma tabela própria e poder arrumá-la com os critérios que preferir.
RNF22	Demanda de Jogadores para o Empresário	Regras de Negócio	Um empresário pode se inscrever em algum clube para receber notificações quando o clube anunciar uma demanda.
RF16	Demanda de Jogadores para o Empresário	Funcional	Um empresário pode se desinscrever quando quiser de um clube inscrito.
RNF23	Demanda de Jogadores para o Empresário	Regras de Negócio	Um empresário pode criar regras para receber propostas de clubes que se encaixem no perfil escolhido.
RNF24	Demanda de Jogadores para o Clube	Regras de Negócio	Um clube pode criar regras para receber propostas de clubes e empresários que se encaixem no perfil escolhido.
RNF25	Cadastro de Jogadores do Clube	Regras de Negócio	A criação de um Jogador poderá ser efetivada pelo clube.

RNF26	Cadastro de Jogadores do Empresário	Regras de Negócio	A criação de um Jogador só poderá ser efetivada quando os documentos obrigatórios forem entregues e validados pelo suporte.
RF17	Cadastro de Jogadores	Funcional	Um jogador não pode ser criado mais de uma vez dentro do Banco de Dados.
RF18	Transferência de Jogadores	Funcional	Um clube ou empresário deve ser capaz de transferir um determinado jogador para o novo “proprietário" contratante
RF19	Módulo de Recomendações do Clube	Funcional	Um usuário clube e suas sub-contas devem ser capazes de procurar na aba especial recomendações de jogadores.
RNF27	Módulo de Recomendações do Clube	Regras de Negócio	Um usuário clube deve ser capaz de favoritar um jogador de interesse
RNF28	Módulo de Recomendações do Clube	Regras de Negócio	Um usuário clube deve ser capaz de entrar em contato com o empresário responsável pelo jogador de interesse
RF20	Chat de Mensagens do Clube	Funcional	Um usuário clube e suas sub-contas devem ser capazes de entrar em contato com clubes e empresários de seu interesse

RF21	Chat de Mensagens do Empresário	Funcional	Um usuário empresário deve ser capaz de entrar em contato com clubes e empresários de seu interesse
RF22	Editar Clube	Funcional	Um usuário clube, após receber o acesso a plataforma, precisa editar os seguintes campos: 'nomeUser', 'senha', 'email', 'gerente', 'fotoGerente', 'cnpj', 'escudoTime' e 'nomeClube'
RNF29	Editar Clube	Regras de Negócio	Um usuário clube só mudar seu gerente se trocar a 'fotoGerente' junto.
RNF30	Jogador	Regras de Negócio	O atributo 'Idade' precisa estar entre 17 e 50 anos
RNF31	Jogador	Regras de Negócio	O atributo 'Altura' precisa estar entre 150 cm e 210 cm
RNF32	Jogador	Regras de Negócio	O atributo 'Peso' precisa estar entre 50 kg e 110 kg
RNF33	Jogador	Regras de Negócio	O atributo 'FotoJogador' precisa ter tamanho 152x152 pixels
RNF34	Jogador	Regras de Negócio	O atributo 'Posicao' não pode ser nulo
RNF35	Empresário	Regras de Negócio	O atributo 'cpf' precisa ser um atributo válido

RNF36	Empresário	Regras de Negócio	O atributo 'FotoEmpresario' precisa ter tamanho 152x152 pixels
RNF37	Usuários (Clube e Empresário)	Regras de Negócio	Os atributos 'telefone', 'linkedin' e 'curriculoCV' podem ser nulos
RNF38	Usuários (Clube e Empresário)	Regras de Negócio	O atributo 'nomeUser' precisa ter 6 caracteres e não podem ser especiais
RNF39	Usuários (Clube e Empresário)	Regras de Negócio	Para mudar o atributo 'nomeUser' é preciso verificar se já existe o novo no sistema.
RNF40	Usuários (Clube e Empresário)	Regras de Negócio	O usuário precisa ter pelo menos 1 jogador anunciado para receber uma proposta
RNF41	Usuários (Clube e Empresário)	Regras de Negócio	O usuário precisa ser verificado e autenticado pelo sistema.
RNF42	Clube	Regras de Negócio	O atributo 'FotoGerente' e ‘FotoMascote’ precisam ter tamanho 152x152 pixels
RNF43	Clube	Regras de Negócio	O atributo ‘cnpj’ precisa ser um CNPJ válido
RNF44	Clube	Regras de Negócio	O atributo ‘nomeClube’ não pode ser nulo
RNF45	Clube	Regras de Negócio	O atributo ‘Mascote’ é opcional

RNF46	Consultar Clube	Regras de Negócio	A procura pode ser feita usando ‘Nome de Usuário’, ‘NomeClube’ e ‘CNPJ’, e portanto esses campos não podem ser nulos ou iguais aos de outro usuário.
RNF47	Sub-Contas do Clube	Regras de Negócio	Caso o nome de usuário seja modificado, é preciso verificar se existem alguém usando o novo nome.
RNF48	Anúncios	Regras de Negócio	O sistema só vai apresentar anúncios caso o usuário seja um ‘jogador’ e possua pelo menos 1 anúncio ativo e esteja com estado de disponibilidade.
RNF49	Consultar Empresário	Regras de Negócio	A procura pode ser feita usando ‘Nome de Usuário’, ‘CredencialCBF’ e ‘CPF’, e portanto esses campos não podem ser nulos ou iguais aos de outro usuário.
RNF50	Consultar Empresário	Regras de Negócio	A procura pode ser feita usando a empresa que o empresário trabalha, tanto pelo ‘NomeEmpresa’ como pelo seu ‘CNPJ’.
RNF51	Empresário	Regras de Negócio	O atributo ‘CredencialCBF’ precisa ser uma credencial válida
RNF52	Anunciar Jogador (Clubes e Empresários)	Regras de Negócio	O campo ‘TempoEmprestimo’ não é obrigatório e a criação do anúncio pode ocorrer sem o preenchimento deste, desde que o ‘Tipo’ não seja empréstimo.

RNF53	Anunciar Jogador (Clubes e Empresários)	Regras de Negócio	O campo 'Tempo Empréstimo', caso seja preenchido, precisa ter uma duração de pelo menos 12 meses.
RNF54	Anunciar Jogador (Clubes e Empresários)	Regras de Negócio	Caso a Proposta seja editada, o valor precisa ser maior que a proposta anterior.

Figura 8 - Requisitos Funcionais e Não Funcionais

6.Especificação

6.1 Casos de Uso

Essa etapa do projeto foi fundamental para entender o escopo do projeto, já que evidencia de forma mais clara a fronteira do sistema, determinando objetivamente quais atores interagem com o sistema e como interagem. Por descrever os requisitos e funcionalidades do sistema, os Casos de Uso foram utilizados nas entrevistas piloto para tentar facilitar a comunicação com o usuário do sistema. Em seguida, conjuntamente com os feedbacks das entrevistas, os casos de uso foram utilizados no momento da construção do protótipo de média fidelidade[5], sendo este o responsável por ilustrar as funcionalidades reais do projeto.

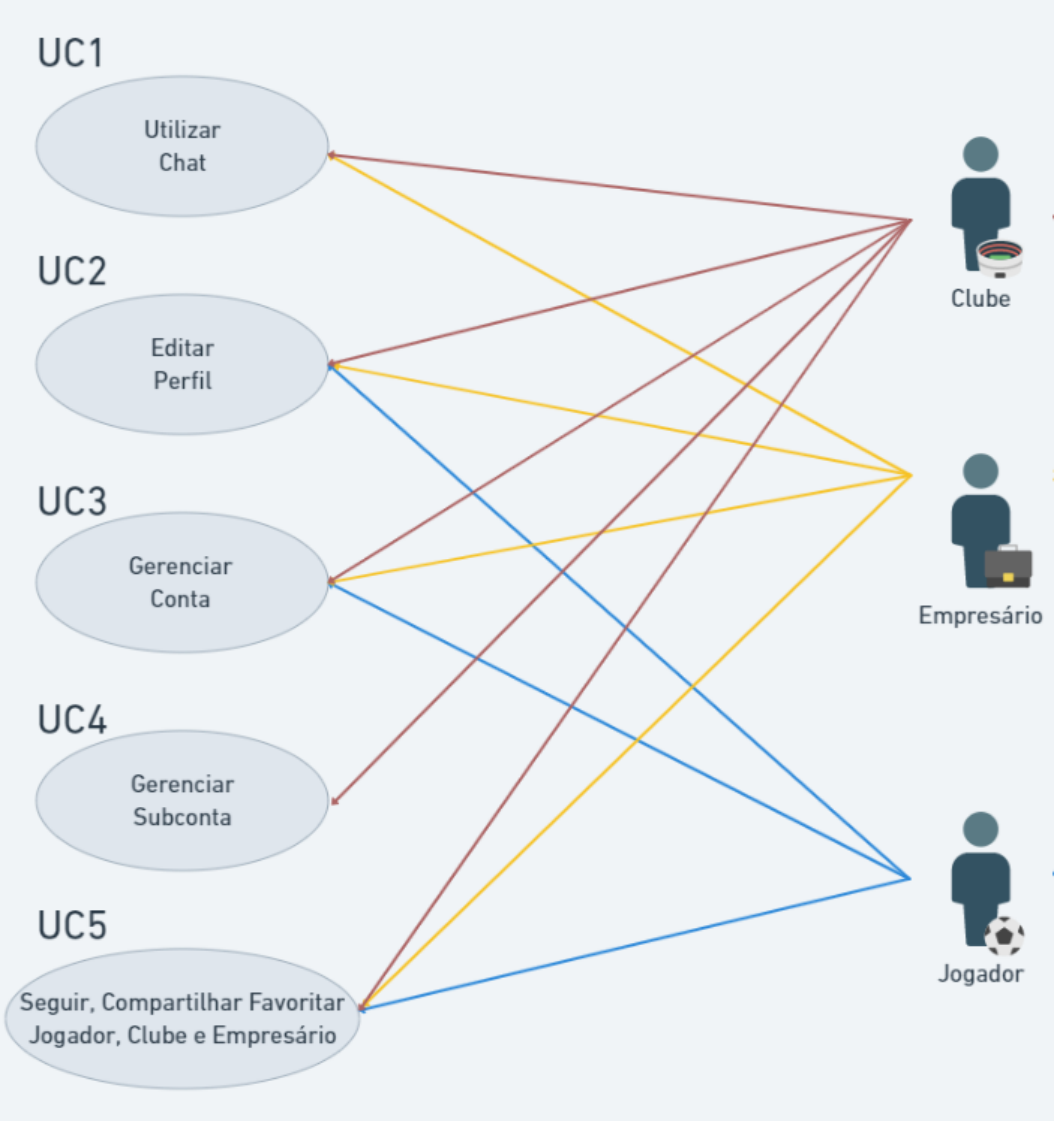

Figura 9 - Primeira parte do Diagrama de Casos de Uso

[5] Também conhecido como wireframe, é usado quando o objetivo é fazer a validação da arquitetura da informação e a interatividade com os elementos que compõem a interface.

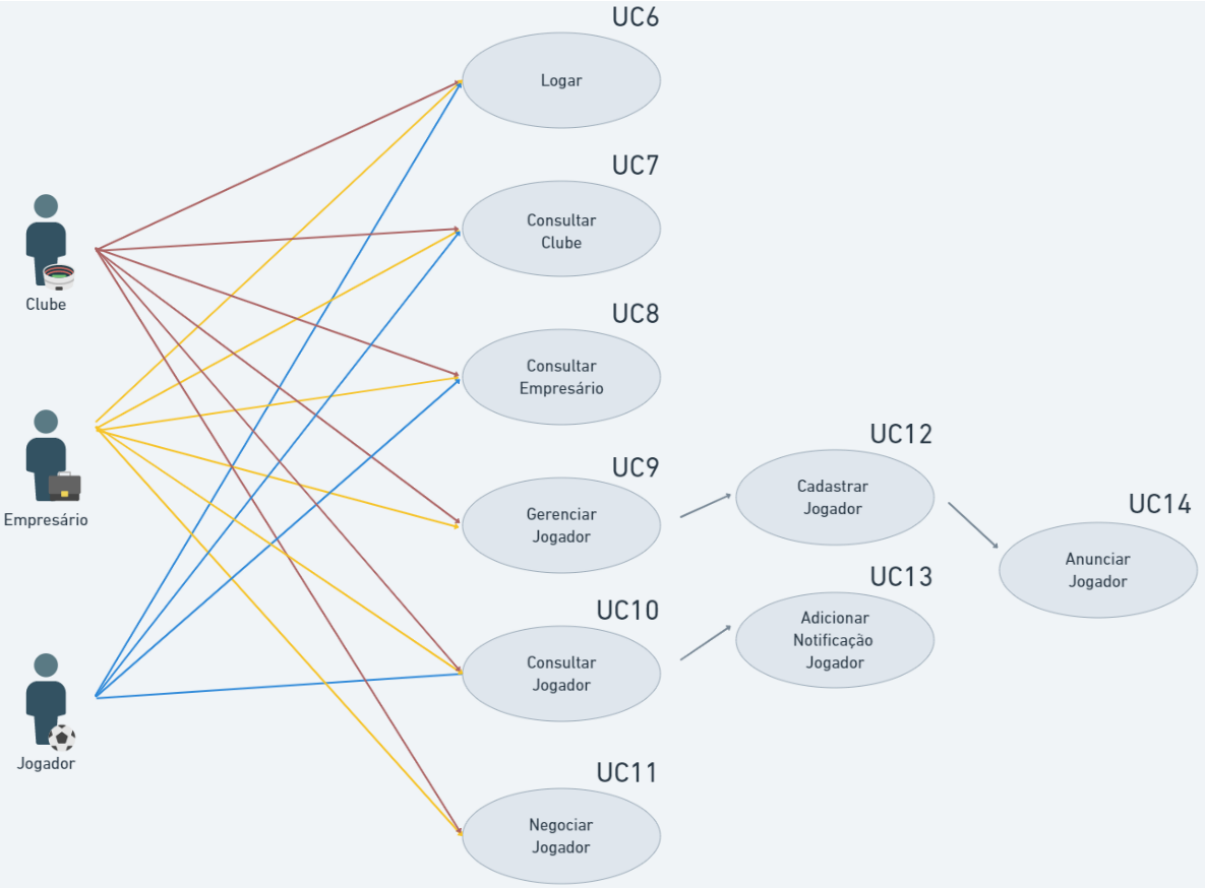

Figura 10 - Segunda parte do Diagrama de Casos de Uso

O uso desses Casos de Uso e suas correspondentes descrições foram extremamente importantes para a prototipagem da aplicação. Para cada uma das telas prototipadas na seção *6.3 Prototipagem*, foi usado um ou mais casos de uso, responsáveis por guiar o processo de desenvolvimento das telas, que serão citados. As relações entre as descrições de caso de uso e as telas prototipadas se encontram nos bullets abaixo, seguidas de todos as descrições de casos de uso criadas:

- Tela de Login e Configurações - Descrição de casos de uso UC14;
- Tela de Encontrar Jogador - Descrição de casos de uso UC05 e UC13;
- Tela da Página Inicial - Descrição de casos de uso UC08, UC09 e UC10;
- Tela de Favoritos - Descrição de casos de uso UC06;
- Tela de Notificações - Descrição de casos de uso UC06 e UC07;
- Tela de Anúncios - Descrição de casos de uso UC12;
- Tela de Chat - Descrição de casos de uso UC01 e UC13;
- Tela de Perfil - Descrição de casos de uso UC02, UC03, UC04, UC05 e UC11;

Descrição de Caso de Uso UC01

Nome:	Utilizar Chat.
Objetivo:	Permitir que os usuários possam se comunicar entre si e dar andamento às negociações de jogadores.
Pré condição:	

Atores:	Clubes e Empresários.
Trigger:	O ícone de balão com texto (Chat), no canto superior direito da tela é acionado.
Fluxo Principal:	1. O usuário seleciona o botão do Chat no canto superior direito da tela.[A1] 2. O sistema apresenta a tela de conversas e todas as conversas que existem.[A2] 3. O ator seleciona uma conversa com interessado[A3]. 4. O ator seleciona uma conversa com um clube que tem interesse[A5]. 5. O sistema retorna para o passo 2 do fluxo principal.
Fluxos Alternativos:	[A1] O ator não possui permissão para utilizar o chat. 1. O caso de uso retorna à tela anterior a ação do ator. [A2] O ator não possui nenhuma conversa. 1. O sistema sugere que o ator comece novas conversas. 2. O sistema aponta para o usuário a tela de 'Encontrar Jogador', para começar a busca por talentos. 3. O caso de uso cria um link para a tela de 'Encontrar Jogador'. [A3] O ator acessa uma conversa com algum interessado que criou uma proposta.[RN40] 1. O ator rejeita a proposta.[A4] 2. O ator aceita continuar a negociação. [A4] O ator recusa negociar com o interessado. 1. O sistema retorna para a tela de Chats. [A5] Acessou a conversa com um empresário/clube que tem interesse.[RN41] 1. O clube de interesse não quis negociar.[A6] 2. O ator continua a negociação.[PE1] [A6] Negociação não é continuada. 1. O sistema apresenta a mensagem 'Negociação não executada'. 2. O sistema retorna para o passo 2 do fluxo principal.
Pontos de Extensão	[PE1] O usuário seleciona a opção 'Continuar Negociação'. Ponto de extensão para o caso UC11 'Negociar Jogadores'.
Regras de Negócio	[RN40] O usuário precisa ter pelo menos 1 jogador anunciado para receber uma proposta. [RN41] O usuário precisa ser verificado e autenticado pelo sistema.

Figura 11 - Descrição do Caso de Uso UC01

Descrição de Caso de Uso UC02

Nome:	Editar Perfil.
Objetivo:	Permitir que o usuário atualize dados pessoais e profissionais.
Atores:	Jogadores, Clubes e Empresários.
Trigger:	O botão embaixo do nome do usuário, de 'Ver Perfil'.
Fluxo Principal:	1. O sistema apresenta uma nova tela lateral. 2. O sistema apresenta um botão embaixo do nome do usuário, de 'Ver Perfil'. 3. O ator seleciona o botão. 4. O sistema apresenta os campos com as informações atuais do usuário. 5. O ator modifica suas informações. 6. O ator seleciona a opção de 'Confirmar Modificações'.[A1] 7. O sistema fecha a tela de informações e retorna para o passo 1 do fluxo principal
Fluxos Alternativos:	[A1] Há campos obrigatórios inválidos/não preenchidos. [RNJ][RNU][RNC][RNE] 1. O sistema apresenta a mensagem 'Há informações inválidas'. 2. O sistema retorna para o passo 3 do fluxo principal, com as informações modificadas do usuário.
Pontos de Extensão	
Regras de Negócio	[RNJ] (Caso seja jogador) - [RN30] 'Idade' deve possuir um valor pertencente a faixa etária de 17 a 50 anos. - [RN31] 'Altura' deve estar entre 150cm e 210cm. - [RN32] 'Peso' deve estar entre 50kg e 110kg. - [RN33] 'FotoJogador' precisa ter um tamanho de 152x152 pixels. [RNU] (Caso seja Usuário) [RN37] O campo 'Telefone' não é obrigatório e o cadastro pode ocorrer sem o preenchimento deste. - [RN37] O campo 'Linkedin' não é obrigatório e o cadastro pode ocorrer sem o preenchimento deste. - [RN37] O campo 'CurriculoCV' não é obrigatório e o cadastro pode ocorrer sem o preenchimento deste.

	[RNC] (Caso seja Clube) [RN29] O campo 'Gerente' só pode ser modificado caso a 'FotoGerente' também seja alterada. - [RN42] O campo 'FotoGerente' precisa ter um tamanho de 152x152 pixels. - [RN43] O campo 'CNPJ' precisa ser um CNPJ válido. - [RN45] O campo 'Mascote' não é obrigatório e a modificação pode ocorrer sem o preenchimento deste. - [RN42] O campo 'EscudoTime' precisa ter um tamanho de 152x152 pixels. - [RN44] O campo 'NomeClube' não pode ser nulo. [RNE] (Caso seja Empresário) [RN01] O campo 'CredencialCBF' precisa ser uma credencial válida. - [RN35] O campo 'CPF' precisa ser um CPF válido. - [RN36] O campo 'FotoEmpresário' precisa ter um tamanho de 152x152 pixels.

Figura 12 - Descrição do Caso de Uso UC02

Descrição de Caso de Uso UC03

Nome:	Gerenciar Conta.
Objetivo:	Permitir que o usuário modifique configurações da conta, como notificações, nome de usuário, email, senha entre outros.
Atores:	Clubes e Empresários.
Trigger:	A opção 'Configurações' na tela lateral de perfil, embaixo do nome do usuário.
Fluxo Principal:	1. O sistema apresenta uma nova tela lateral com a lista de ações do perfil. 2. O sistema apresenta um botão embaixo do nome do usuário, de 'Configurações'. 3. O ator seleciona o botão. 4. O sistema apresenta uma lista de opções que podem ser selecionadas. 5. O ator seleciona 'Notificações'.[A1] 6. O ator seleciona 'Conta'.[A2] 8. O sistema fecha a tela de informações e retorna para o passo 1 do fluxo principal.
Fluxos Alternativos:	[A1] Abre uma nova tela/modal com a possibilidade de ativar/desativar notificações 1. O ator escolhe se quer ou não receber notificações. 2. Confirme sua opção. 3. O sistema retorna para o passo 4 do fluxo principal. [A2] Abre uma nova tela/modal com a possibilidade de gerenciar os dados da conta. 1. O ator modifica/adiciona os campos que quer modificar. 2. O ator seleciona a opção de 'Confirmar Modificações'.[A3] [A3] Há campos obrigatórios inválidos/não preenchidos. [RN11][RN04] 1. O sistema apresenta a mensagem 'Há informações inválidas'. 2. O sistema retorna para o passo 1 do fluxo alternativo A2, com as informações modificadas do usuário.
Pontos de Extensão	
Regras de Negócio	[RN11] Caso o nome de usuário ou email seja modificado, é preciso verificar se existe alguém usando o novo nome ou email. [RN04] Caso a senha seja modificada, ela deverá ter mais de 8 dígitos, caracteres especiais, letras maiúsculas e minúsculas.

Figura 13 - Descrição do Caso de Uso UC03

Descrição de Caso de Uso UC04

Nome:	Gerenciar Subconta.
Objetivo:	Permitir que o usuário administrador crie subcontas e atribua permissões para cada uma delas.
Atores:	Clubes.
Trigger:	A opção de 'Gerenciar Subcontas' na tela lateral de Perfil.
Fluxo Principal:	1. O sistema apresenta uma nova tela lateral. 2. O sistema apresenta um botão embaixo do nome do usuário, chamado de 'Gerenciar Sub-contas'. 3. O ator seleciona o botão. 4. O sistema apresenta um drop down com os usuários criados e a possibilidade de criar outro. 5. O ator seleciona um usuário criado.[A1] 6. O ator seleciona a opção de criar um novo usuário. [A2]

	7. O sistema fecha a tela de informações e retorna para o passo 1 do fluxo principal
Fluxos Alternativos:	[A1] Abre uma tela com as informações da conta selecionada. 1. O ator modifica as permissões da conta. 2. O ator modifica informações da conta. 3. O sistema apresenta na tela a opção de salvar as modificações.[A3] 4. O usuário administrador seleciona a opção de 'Confirmar modificações'.[RN47][RN04][RN11][RN37] [A2] Abre uma tela para criar uma nova subconta. 1. O ator preenche os dados da nova conta: - os campos 'Email', 'Nome de usuário' e 'Senha' como obrigatórios. - os campos 'linkedin' e 'currículo CV' como não obrigatórios. 2. O usuário administrador seleciona a opção de 'Criar nova conta'. [RN47][RN04][RN11][RN37] [A3] O ator seleciona a opção 'Cancelar'. 1. O caso de uso retorna a tela da conta.
Pontos de Extensão	
Regras de Negócio	[RN47] Caso o nome de usuário seja criado ou modificado, é preciso verificar se existem alguém usando o novo nome. [RN04] Caso a senha seja criada ou modificada, ela deverá ter mais de 8 dígitos, caracteres especiais, letras maiúsculas e minúsculas. [RN11] Caso o email seja criado ou modificado, é necessário fazer a confirmação do email. [RN37] Os campos 'Telefone', 'Linkedin' e 'currículo CV' não são obrigatórios e o cadastro pode ocorrer sem o preenchimento destes.

Figura 14 - Descrição do Caso de Uso UC04

Descrição de Caso de Uso UC05

Nome:	Seguir, Favoritar e Compartilhar
Objetivo:	Permite que o usuário consiga seguir, favoritar e compartilhar jogadores, empresários e clubes.
Atores:	Jogadores, Clubes e Empresários. Obs: o **usuário** representa qualquer um dos três tipos.
Trigger:	A opção de 'seguir, favoritar e compartilhar' embaixo da foto da pessoa que queira performar a ação.
Fluxo Principal:	1. O sistema apresenta a tela de início. 2. O ator seleciona a barra de busca no topo da tela. 3. O ator procura pelo jogador/clube/empresário que tem interesse(Consultar Jogador). [A1] 4. O ator seleciona o usuário que buscava. 5. O sistema apresenta o perfil do usuário, com suas informações, posts e anúncios.[RN48] 6. O ator seleciona o botão de compartilhar usuário.[A2] 7. O ator seleciona o botão de seguir/favoritar o usuário. 8. O sistema mostra uma mensagem avisando que a ação foi concluída. 9. O caso de uso retorna para o passo 5 do fluxo principal.
Fluxos Alternativos:	[A1] Não existe nenhum usuário com o nome dado pelo ator. 1. O sistema retorna para o passo 2 do fluxo principal. [A2] O sistema apresenta uma nova tela oferecendo possibilidades de onde compartilhar. 1. O ator seleciona para onde quer compartilhar. 2. O sistema retorna para o passo 5 do fluxo principal.
Pontos de Extensão	
Regras de Negócio	[RN48] O sistema só vai apresentar anúncios caso o usuário seja um jogador e possua pelo menos 1 anúncio ativo e esteja com estado de disponibilidade.

Figura 15 - Descrição do Caso de Uso UC05

Descrição de Caso de Uso UC06

Nome:	Logar
Objetivo:	Permite que os usuários acessem o sistema.
Pré condição:	UC01 executada, no caso dos jogadores. Email de acesso aos Empresários e Clubes, liberando troca de senha.
Atores:	Clubes e Empresários.
Trigger:	A opção de 'Acessar' na tela de login do sistema é acionada.
Fluxo Principal:	O sistema apresenta uma nova tela contendo: - os campos 'Email/ Nome de Usuário' e'Senha' como obrigatórios.

	2. O ator preenche os campos ‘Email/ Nome de Usuário’ e ‘Senha’. 3. O ator seleciona a opção 'Acessar'. [A1][RN1] 4. O sistema fecha a tela de login do usuário e carrega a página principal da aplicação.
Fluxos Alternativos:	[A1] Há campos obrigatórios inválidos/não preenchidos. [RN11][RN04] 1. O sistema apresenta a mensagem ‘Há informações inválidas’. 2. O sistema retorna para o passo 1 do fluxo principal.
Pontos de Extensão	
Regras de Negócio	[RN11] O campo ‘Nome de usuário' e ‘Email’ precisam ser únicos. [RN04] O campo ‘Senha’ precisa possuir mais de 8 caracteres (contendo letras maiúsculas, minúsculas, caracteres especiais e números).

Figura 16 - Descrição do Caso de Uso UC06

Descrição de Caso de Uso UC07

Nome:	Consultar Clube.
Objetivo:	Permite que o ator visualize informações de clubes que têm interesse.
Pré condição	
Atores:	Jogadores, Clubes e Empresários.
Trigger:	A opção de busca na barra de navegação.
Fluxo Principal:	1. O sistema apresenta a tela de início. 2. O ator seleciona a barra de busca no topo da tela. 3. O ator procura pelo clube que tem interesse.[A1][RN46] 4. O ator seleciona o clube que buscava, dentre as opções que apareceram. 5. O sistema apresenta o perfil do clube , com suas informações, posts e anúncios. 6. O ator termina de olhar o clube . 7. O ator seleciona o botão de ‘voltar’. 8. O sistema retorna para o passo 2 do fluxo principal, com a última busca do usuário escrita.
Fluxos Alternativos:	[A1] Não existe nenhum clube com o nome dado pelo ator. 1. O sistema retorna para o passo 2 do fluxo principal.
Regras de Negócio	[RN46] A procura pode ser feita usando ‘Nome de Usuário’, ‘NomeClube’ e ‘CNPJ’, e portanto esses campos não podem ser nulos ou iguais aos de outro usuário.

Figura 17 - Descrição do Caso de Uso UC07

Descrição de Caso de Uso UC08

Nome:	Consultar Empresário.
Objetivo:	Permite que o ator visualize informações de empresários que têm interesse.
Pré condição:	
Atores:	Jogadores, Clubes e Empresários.
Trigger:	A opção de busca na barra de navegação.
Fluxo Principal:	1. O sistema apresenta a tela de início. 2. O ator seleciona a barra de busca no topo da tela. 3. O ator procura pelo empresário que tem interesse.[A1][RN49][RN50][RN51][RN35] 4. O ator seleciona o empresário que buscava, dentre as opções que apareceram. 5. O sistema apresenta o perfil do empresário, com suas informações, posts e anúncios. 6. O ator termina de olhar o empresário. 7. O ator seleciona o botão de ‘voltar’. 8. O sistema retorna para o passo 2 do fluxo principal, com a última busca do usuário escrita.
Fluxos Alternativos:	[A1] Não existe nenhum empresário com o nome dado pelo ator. 1. O sistema retorna para o passo 2 do fluxo principal.
Pontos de Extensão	
Regras de Negócio	[RN49] A procura pode ser feita usando ‘Nome de Usuário’, ‘CredencialCBF’ e ‘CPF’, e portanto esses campos não podem ser nulos ou iguais aos de outro usuário. [RN50] A procura pode ser feita usando a empresa que o empresário trabalha, tanto pelo ‘NomeEmpresa’ como pelo seu ‘CNPJ’. [RN51] O atributo ‘CredencialCBF’ precisa ser uma credencial válida. [RN35] O atributo ‘CPF’ precisa ser um CPF válido.

Figura 18 - Descrição do Caso de Uso UC08

Descrição de Caso de Uso UC09

Nome:	Gerenciar Jogador.
Objetivo:	Permitir que o usuário administrador gerencie jogadores.
Atores:	Clubes e Empresários.
Trigger:	A opção de 'Gerenciar Jogadores' na tela lateral de Perfil.
Fluxo Principal:	1. O sistema apresenta uma nova tela lateral. 2. O sistema apresenta um botão embaixo do nome do usuário, chamado de 'Gerenciar Jogadores'. 3. O ator seleciona o botão. 4. O sistema apresenta um dropdown com os jogadores criados.[PE1] 5. O ator seleciona um jogador existente.[A1] 7. O sistema fecha a tela de informações e retorna para o passo 1 do fluxo principal.
Fluxos Alternativos:	[A1] Abre uma tela com as informações do jogador selecionado. 2. O ator muda informações do jogador. 3. O sistema apresenta na tela a opção de salvar as modificações.[A3][RN30][RN31][RN32][RN33] 4. O usuário administrador seleciona a opção de 'Confirmar mudanças'. 5. O caso de uso avança para o passo 7 do fluxo principal. [A3] O ator seleciona a opção 'Cancelar'. 1. O caso de uso retorna ao passo 1 do fluxo principal.
Pontos de Extensão	[PE1] Caso o usuário não tenha cadastrado nenhum jogador, ponto de extensão para o caso de uso UC12 'Cadastrar Jogador'.
Regras de Negócio	[RN30] 'Idade' deve possuir um valor pertencente a faixa etária de 17 a 50 anos. [RN31] 'Altura' deve estar entre 150cm e 210cm. [RN32] 'Peso' deve estar entre 50kg e 110kg. [RN33] 'FotoJogador' precisa ter um tamanho de 152x152 pixels.

Figura 19 - Descrição do Caso de Uso UC09

Descrição de Caso de Uso UC10

Nome:	Consultar Jogador.
Objetivo:	Permite que o ator visualize informações de jogadores que têm interesse.
Pré condição	
Atores:	Jogadores, Clubes e Empresários.
Trigger:	A opção de busca na barra de navegação.
Fluxo Principal:	1. O sistema apresenta a tela de início. 2. O ator seleciona a barra de busca no topo da tela. 3. O ator procura pelo jogador que tem interesse.[A1] 4. O ator seleciona o usuário que buscava. 5. O sistema apresenta o perfil do jogador, com suas informações, posts e anúncios. 6. O ator termina de olhar o jogador. 7. O ator seleciona o botão de 'voltar'. 8. O sistema retorna para o passo 2 do fluxo principal, com a última busca do usuário escrita.
Fluxos Alternativos:	[A1] Não existe nenhum jogador com o nome dado pelo ator. 1. O sistema retorna para o passo 2 do fluxo principal.
Pontos de Extensão	
Regras de Negócio	

Figura 20 - Descrição do Caso de Uso UC10

Descrição de Caso de Uso UC11

Nome:	Negociar Jogador
Objetivo:	Permite que os usuários consigam negociar seus jogadores entre si.
Pré condição:	UC01 executada, com ao menos 1 fluxo alternativo, 3 ou 5 executados. UC12 executada, ao menos 1 jogador criado.
Atores:	Clubes e Empresários.

Trigger:	A opção ‘Continuar Negociação’ em uma conversa, no Chat.
Fluxo Principal:	1. O sistema apresenta a tela de chat. 2. O usuário seleciona uma das conversas disponíveis. 3. O sistema apresenta uma mensagem com a proposta do interessado. 3. O usuário visualiza a proposta encaminhada. 4. O usuário seleciona a ação de ‘Continuar Negociação’.[A1] 5. Negociação ocorre enquanto o usuário possui interesse. 6. O sistema retorna ao passo 1 do fluxo principal.
Fluxos Alternativos:	[A1] O ator seleciona a opção ‘Recusar Proposta’. 1. O chat com o interessado é encerrado. 2. O caso de uso retorna ao passo 1 do fluxo principal.
Pontos de Extensão	
Regras de Negócio	

Figura 21 - Descrição do Caso de Uso UC11

Descrição de Caso de Uso UC12

Nome:	Cadastrar Jogador.
Objetivo:	Permitir que o usuário crie jogadores.
Atores:	Clubes e Empresários.
Trigger:	A opção de ‘Gerenciar Jogadores’ na tela lateral de Perfil.
Fluxo Principal:	1. O sistema apresenta uma nova tela lateral. 2. O sistema apresenta um botão embaixo do nome do usuário, chamado de 'Gerenciar Jogadores’. 3. O ator seleciona o botão. 4. O sistema apresenta uma opção de criar jogadores. 5. O seleciona o botão de criar um jogador.[A1] 7. O sistema fecha a tela de informações e retorna para o passo 1 do fluxo principal.
Fluxos Alternativos:	[A1] Abre uma tela para criar um novo jogador. 1. O ator preenche os dados da novo jogador: - os campos ‘Email', 'Nome de usuário' e ‘Senha’ como obrigatórios. - os campos ‘Nome’, ‘Idade’ e ‘PeDominante’ também são obrigatórios. - os campos ‘Altura’, ‘Posicao’, ‘Peso’ e ‘FotoJogador’ também são obrigatórios. 2. O sistema apresenta na tela a opção de 'Criar nova conta’.[A3] 3. O usuário seleciona e confirma a ação de criar jogadores. [RN30][RN31][RN32][RN33] [A3] O ator seleciona a opção 'Cancelar'. 1. O caso de uso retorna ao passo 1 do fluxo principal.
Pontos de Extensão	
Regras de Negócio	[RN30] ‘Idade’ deve possuir um valor pertencente a faixa etária de 17 a 50 anos. [RN31] ‘Altura’ deve estar entre 1.50cm e 2.10cm. [RN32] ‘Peso’ deve estar entre 50kg e 110kg. [RN33] ‘FotoJogador’ precisa ter um tamanho de 152x152 pixels.

Figura 22 - Descrição do Caso de Uso UC12

Descrição de Caso de Uso UC13

Nome:	Adicionar Notificação Jogador
Objetivo:	Permite que o usuário receba notificações de determinado jogador.
Atores:	Jogadores, Clubes e Empresários. Obs: o **usuário** representa qualquer um dos três tipos.
Trigger:	A opção de ‘criar regra’ embaixo da pessoa que queria performar a ação.
Fluxo Principal:	1. O sistema apresenta a tela de início. 2. O ator seleciona a barra de busca no topo da tela. 3. O ator procura pelo jogador/clube/empresário que tem interesse(Consultar Jogador).[A1] 4. O ator seleciona o usuário que buscava. 5. O sistema apresenta o perfil do usuário, com suas informações, posts e anúncios.[RN1] 7. O ator seleciona a opção de ‘criar regra’ no perfil do jogador.[A2]
Fluxos Alternativos:	[A1] Não existe nenhum usuário com o nome dado pelo ator. 1. O sistema retorna para o passo 2 do fluxo principal.

	[A2] O sistema apresenta uma nova tela oferecendo possibilidades de alarme. 1. O ator seleciona quando quer ser notificado. 2. O sistema mostra uma mensagem avisando que a ação foi concluída. 3. O sistema retorna para o passo 5 do fluxo principal.
Pontos de Extensão	
Regras de Negócio	

Figura 23 - Descrição do Caso de Uso UC13

Descrição de Caso de Uso UC14

Nome:	Anunciar Jogador
Objetivo:	Permitir que o usuário anuncie seus jogadores para os clubes.
Pré condição:	UC12 executada, ao menos 1 jogador criado.
Atores:	Clubes e Empresários.
Trigger:	A opção de 'Fazer Anúncio' na aba de Anúncios.
Fluxo Principal:	1. O sistema apresenta a tela na aba de negócios. 2. O usuário seleciona o botão de 'Fazer Anúncio'.[A1] 3. O usuário preenche os dados da proposta: - os campos 'Jogador', 'Valor', 'Tipo' e 'AbertoANegociacao' como obrigatórios. - o campo 'TempoEmprestimo' como não obrigatório. 4. O sistema apresenta na tela a opção de 'Criar anúncio'.[A2] 5. O usuário seleciona e confirma a ação de criar anúncio. [RN52][RN53][RN54] 6. O sistema fecha a tela de informações e retorna para o passo 1 do fluxo principal.
Fluxos Alternativos:	[A1] O ator não possui nenhum jogador. 1. O sistema sugere que o ator crie jogadores.[PE1] [A2] O ator seleciona a opção 'Cancelar'. 1. O caso de uso retorna ao passo 1 do fluxo principal.
Pontos de Extensão	[PE1] - Caso o usuário não tenha cadastrado nenhum jogador, ponto de extensão para o caso de uso UC12 'Cadastrar Jogador'. 1. O sistema é direcionado para a tela de criar jogadores.
Regras de Negócio	[RN52] O campo 'TempoEmprestimo' não é obrigatório e a criação do anúncio pode ocorrer sem o preenchimento deste, desde que o 'Tipo' não seja empréstimo. [RN53] O campo 'Tempo Empréstimo', caso seja preenchido, precisa ter uma duração de pelo menos 12 meses. [RN54] Caso a Proposta seja editada, o valor precisa ser maior que a proposta anterior.

Figura 24 - Descrição do Caso de Uso UC14

6.2 Entrevistas

A partir dos requisitos, dos casos de uso e de um maior amadurecimento em relação a aplicação, foram realizadas duas entrevistas, cada uma com um tipo de usuário diferente (Clube e Empresário), cuja finalidade é entender melhor o que cada um pensa em relação ao que foi proposto, e dessa forma, desenvolver algo que tenha utilidade para ambos. Futuramente mais entrevistas serão feitas na tentativa de lapidar o produto e garantir que a vontade do cliente esteja sempre em primeiro lugar.

Uma boa estratégia de Marketing Digital só é feita a partir da determinação do público-alvo e criação da persona ideal. Apesar de só terem havido 2 entrevistas, o tempo que passei trabalhando na Apple Developer Academy me mostrou que meu maior desafio é atrair novos usuários para convertê-los em leads (potenciais clientes), e, posteriormente transformá-los em clientes efetivos. Portanto, as personas[6] ajudam a desenvolver uma estratégia assertiva e a identificar o perfil do cliente para os serviços ideais.

[6] Representação fictícia do cliente ideal de um negócio.

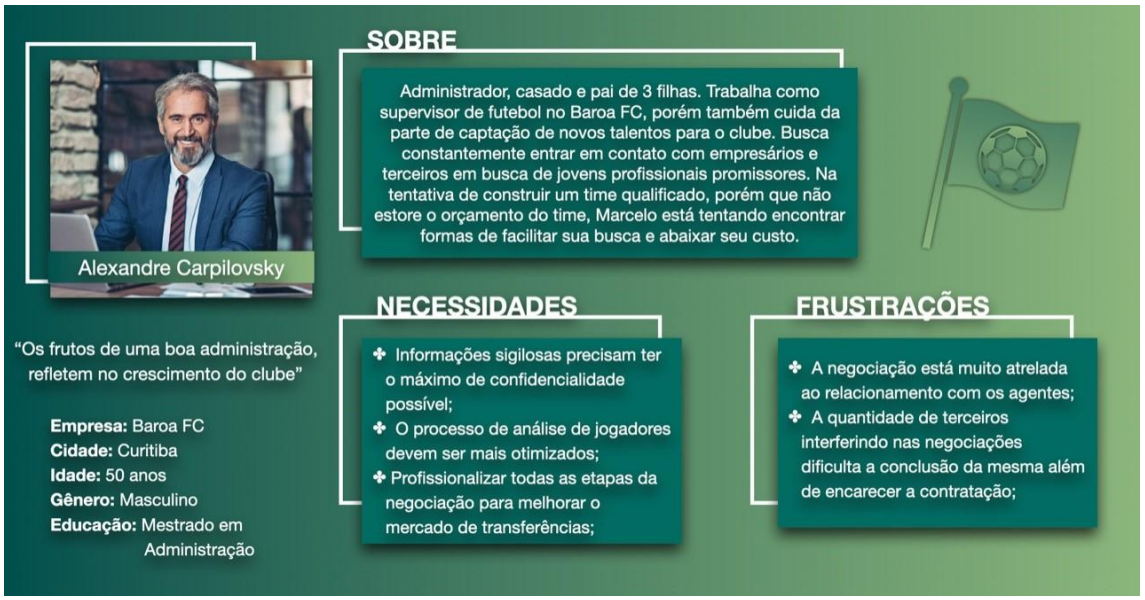

Figura 25 - Persona Clube da TransferWindow

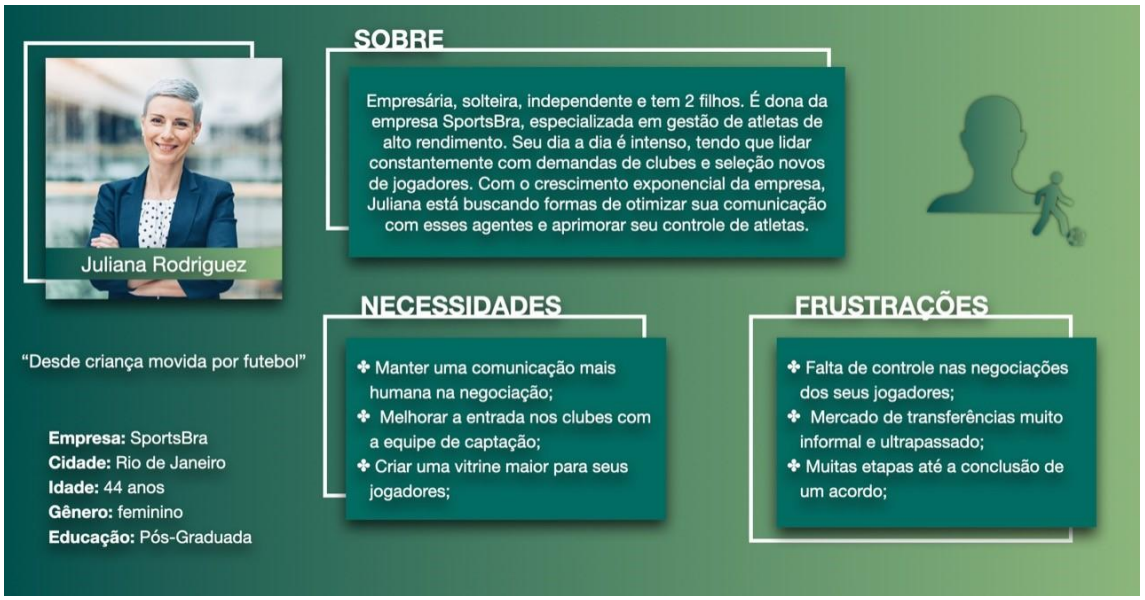

Figura 26 - Persona Empresária da TransferWindow

As **personas** são desenvolvidas através da comunicação com os usuários e são baseadas em dados reais sobre os comportamentos e características dos clientes, histórias e relatos pessoais, motivações, objetivos, preocupações e desafios que eles possuem. Uma boa definição de *persona*, passa pelo contato com o *público-alvo* e depois de uma rápida análise já é possível identificar características comuns entre os usuários. Porém, existe uma grande diferença entre esses dois termos apesar de sempre estarem relacionados.

Enquanto o público-alvo, de maneira geral, define uma parcela mais abrangente do grupo que eu tenho interesse em vender meu projeto, a persona é uma representação mais humanizada e personalizada desse conjunto. Quanto mais pessoas analisadas, mais fácil fica de entender qual a real "***dor"*** ou demanda de um determinado grupo. Para o estágio atual, foi realizada apenas uma

entrevista para cada segmento, com o intuito de demonstrar o conhecimento sobre o assunto em questão e a preocupação com os objetivos dos usuários da aplicação. Na seção 10 de Implementação planejada será abordado as futuras etapas de entrevistas e o aprimoramento das personas.

Ambas as entrevistas foram realizadas por conversa telefônica e a partir de perguntas como: o dia-a-dia da pessoa; dificuldades que enfrenta; possíveis facilitadores no trabalho; conhecimento de algum similar, entre outros. Foi possível ainda entender um pouco mais do que cada um dos entrevistados de cada segmento pensa ou planeja para ser plenamente atendido, além criar uma personificação. O segmento dos jogadores não teve uma persona, isso porque no escopo inicial eles não compunham a parte ativa dos usuários, ou seja, apesar de serem o foco principal, não seriam eles os seus próprios administradores. Com a evolução do projeto, os jogadores também farão parte dos usuários ativos e terão uma persona.

6.3 Prototipagem

Antes de iniciar o desenvolvimento da aplicação, conjuntamente com o levantamento de requisitos, foi criado um protótipo de média fidelidade na ferramenta de prototipagem **Miro[14]**, que engloba toda a idealização do projeto inicial. Este protótipo teve como propósito auxiliar na construção do front-end já que serve como um guia, para a estilização dos componentes como botões e labels, quanto para facilitar a forma que as telas serão estruturadas, buscando sempre manter uma apresentação clara e intuitiva para o usuário e permitindo que a implementação seja feito de forma mais ágil e objetiva.

Outra grande vantagem desse processo foi que, desde antes de implementar a solução, já foi possível identificar e resolver problemas no design pensado, detectar erros e inconsistências, além de perceber a ausência de funcionalidades importantes que passaram despercebidas na etapa de ideação. Ademais, ficou claro que desenvolver uma solução para 2 tipos diferentes de usuário, no escopo de tempo estipulado seria inviável.

6.3.1 Login e Configurações Iniciais

De comum acordo com o orientador do projeto, foi decidido que o foco do projeto seria em apenas um tipo de usuário, e portanto toda a parte de prototipagem até implementação seria criada em cima dos administradores dos Clubes. Sendo assim, para facilitar a visualização e entendimento, a prototipagem foi dividida em seções e serão brevemente explicadas abaixo, começando pela etapa de login e um trecho das configurações iniciais do usuário administrador do clube, apresentada na figura 9 a seguir.

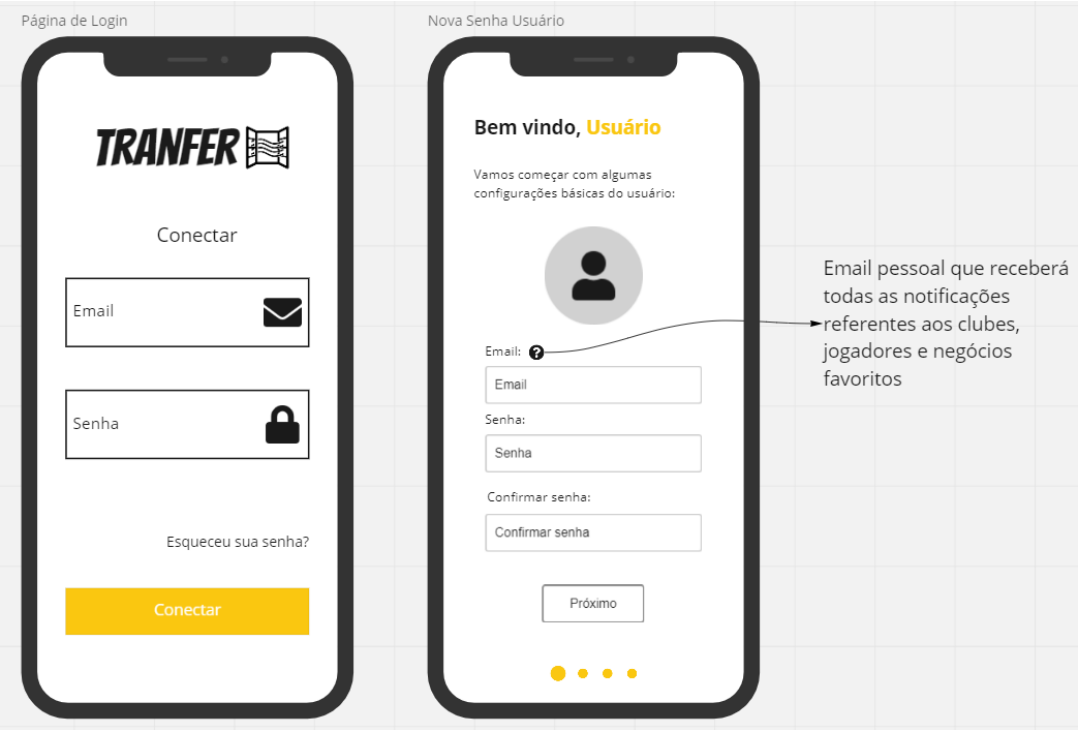

Figura 27 - Tela de Login e Trecho das Configurações iniciais do protótipo da TransferWindow

Os administradores, diferentemente de usuários comuns de redes sociais, não conseguem se cadastrar diretamente pela plataforma, e precisam receber convites enviados diretamente pela equipe de desenvolvimento. Isso porque, como se trata de usuários com possibilidades de criar clubes dentro da plataforma, se mostrou necessário ter um controle maior de quem receberá esse acesso e principalmente impedir que sejam criados clubes por usuários mal intencionados.

A partir do momento que a pessoa encarregada pelo clube recebe o email da equipe do TransferWindow com seu acesso temporário, ela efetua sua entrada na tela de Página de Login. O primeiro passo ao acessar a plataforma é escolher o email que vai ser responsável por receber as notificações do aplicativo, na tela de Nova Senha Usuário. Esse email pode ser o oficial do clube ou o pessoal do encarregado.

Nele serão enviadas notificações relativas a informações específicas escolhidas pelo usuário ao usar a plataforma, como revelação de novos jogadores em outros clubes, atualizações de preços de jogadores, entre outros. E por fim, fazer a configuração com a nova senha que será utilizada para acessar a plataforma. Já na figura 10, é possível ver as telas finais de configuração do usuário, para poder acessar o aplicativo.

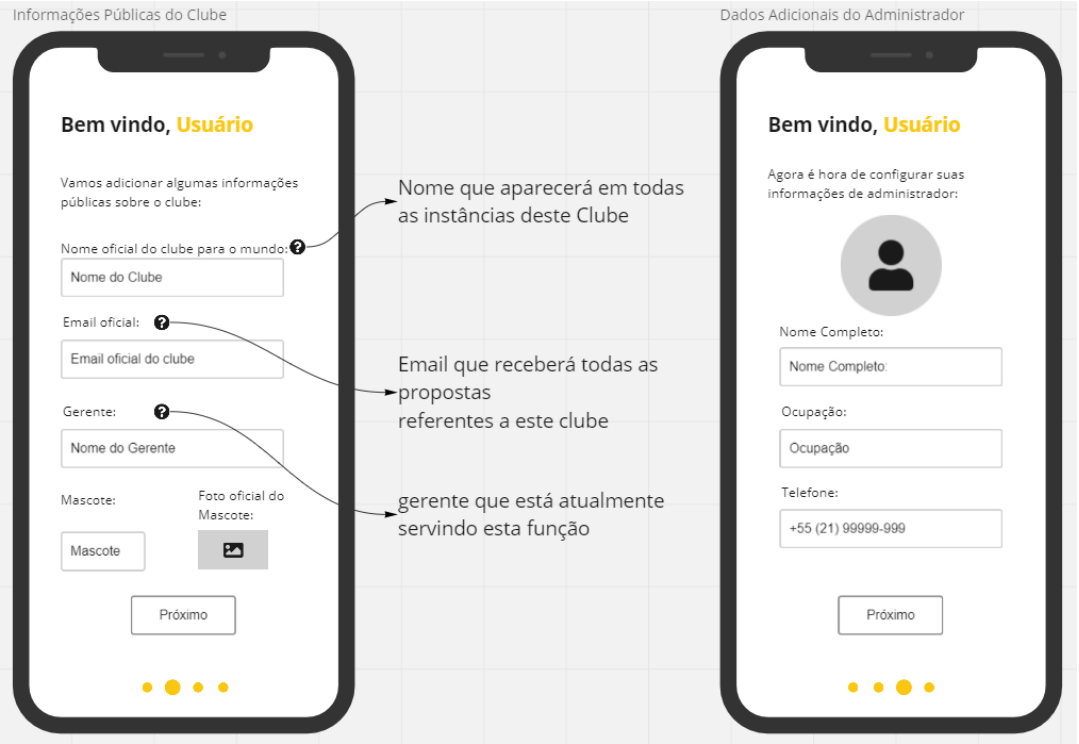

Figura 28 - Configurações iniciais do protótipo da TransferWindow

Em seguida, o administrador precisa adicionar informações básicas referentes ao seu time. Esses dados serão utilizados para identificar o clube dentro da plataforma, e permitir uma simples personalização do clube. Dentre eles,como é possível ver na tela de Informações Públicas do Clube estão o nome oficial, o email oficial, gerente e mascote do time. Por último, na tela de Dados Adicionais do Administrador, o usuário vai preencher seu nome, o email, telefone e a ocupação, que ficarão disponibilizados publicamente.

6.3.2 Encontrar Jogador

Após configurar a conta, o administrador é direcionado para a primeira tela, e talvez a mais importante, de **Encontrar Jogadores**, como é possível visualizar na figura 5, que faz a busca por profissionais de acordo com a demanda de cada clube. A partir do momento que o usuário clica no botão “Filtros”, ele é levado para a próxima tela de Filtros, onde pode escolher os atributos do jogador, de acordo com suas preferências de busca.

Após confirmar suas escolhas, cartões com jogadores que se encaixam no perfil selecionado são apresentados. Cada cartão conta com a idade, posição que atua, time atual, altura, tipo de oferta e valor do atleta. Caso o usuário se interesse por olhar dados mais avançados do atleta, basta clicar no mesmo, e então poderá acessar informações mais avançadas, como o histórico de transferências, valor de mercado, entre outras informações.

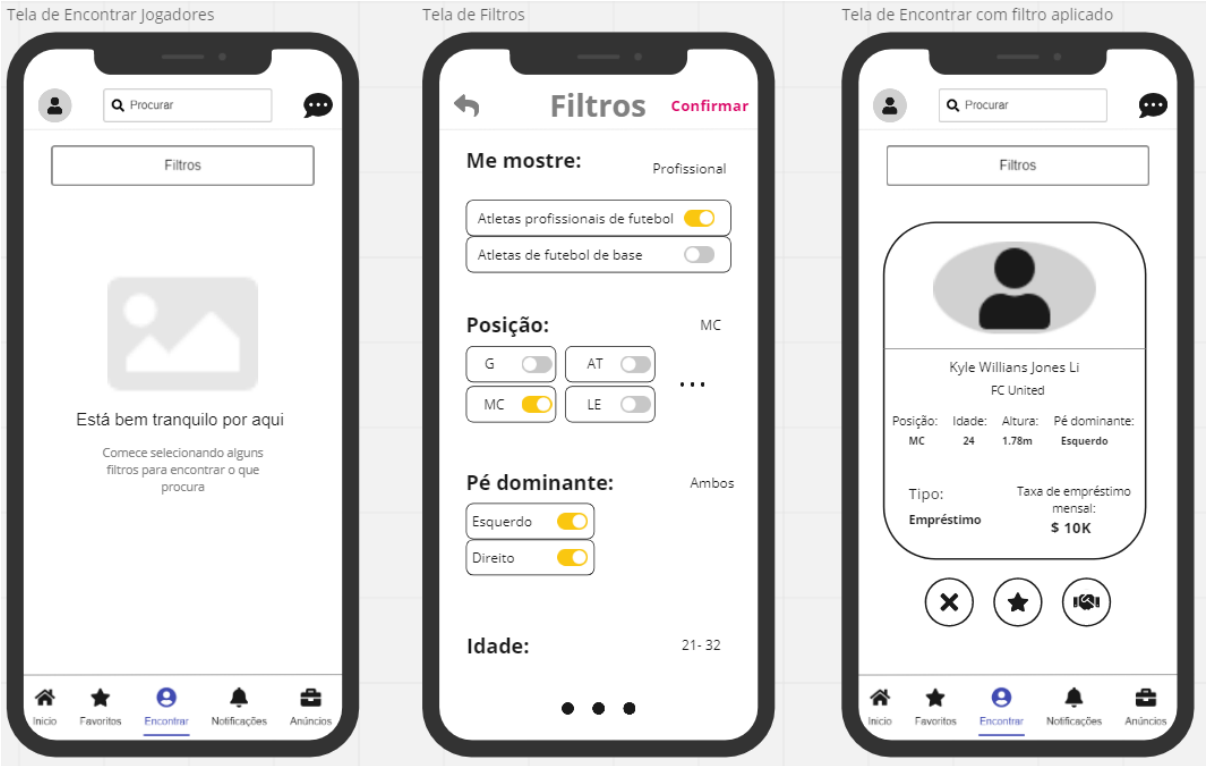

Figura 29 - Tela de Encontrar, Selecionar Filtro e visualizar jogadores da TransferWindow

Assim que o olheiro termina de analisar o atleta, ele pode tomar algumas decisões em cima desse cartão. Caso não haja interesse no jogador, basta apertar o botão de “Passar”, representado pelo ícone de X, ou arrastar o card para a esquerda. Na tela de Recusou Jogador, é possível perceber que assim que o jogador é negado, ocorre uma animação que desloca o cartão para a esquerda. Em seguida, uma nova pessoa que se encaixa no mesmo perfil procurado aparecerá no lugar, podendo ser visualizada na tela de Encontrar com Novo Jogador na figura 12.

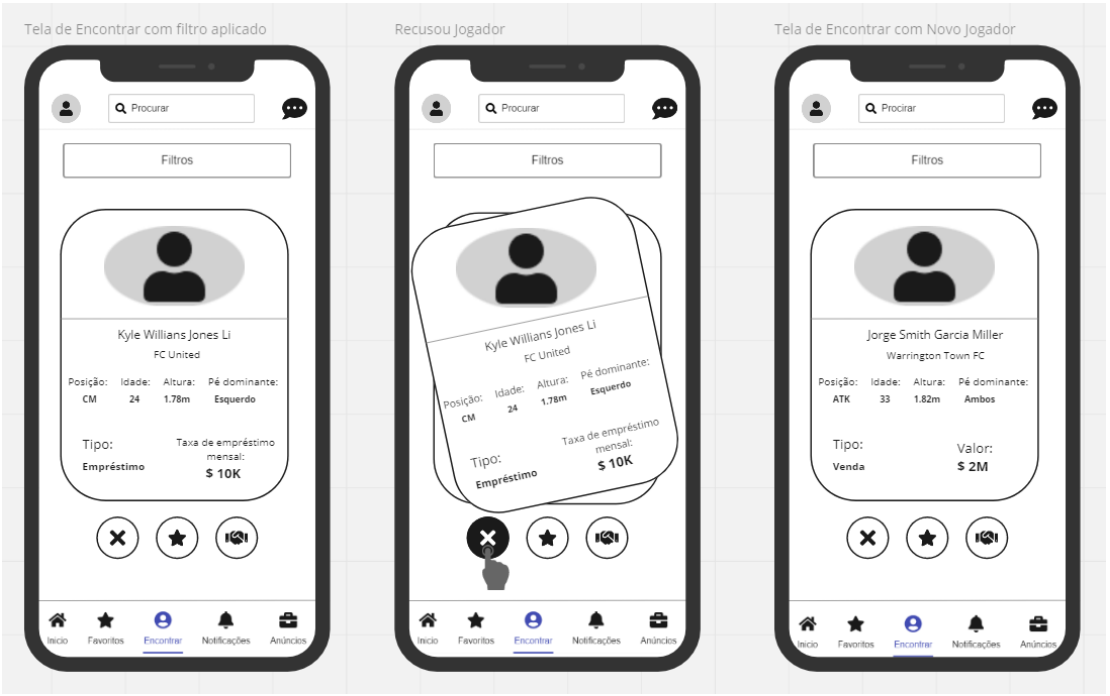

Figura 30 - Tela de Encontrar, Recusou Jogador e visualizou nova opção de jogador da TransferWindow

Outra opção é favoritar o jogador, apertando o botão com ícone de estrela ou apenas arrastando para cima, como na tela de Favoritando Jogador. Nesse caso, é possível determinar qual o tipo de agrupamento que ele vai pertencer, ou seja, qual tabela e como ela está organizada. Além disso, o usuário pode configurar notificações caso informações como valor, data referente ao fim do contrato, valor do jogador no mercado ou outras informações mudem. Dessa forma, é possível acompanhar estrategicamente cada atleta, e tomar a melhor decisão se vale a pena ou não efetuar a contratação.

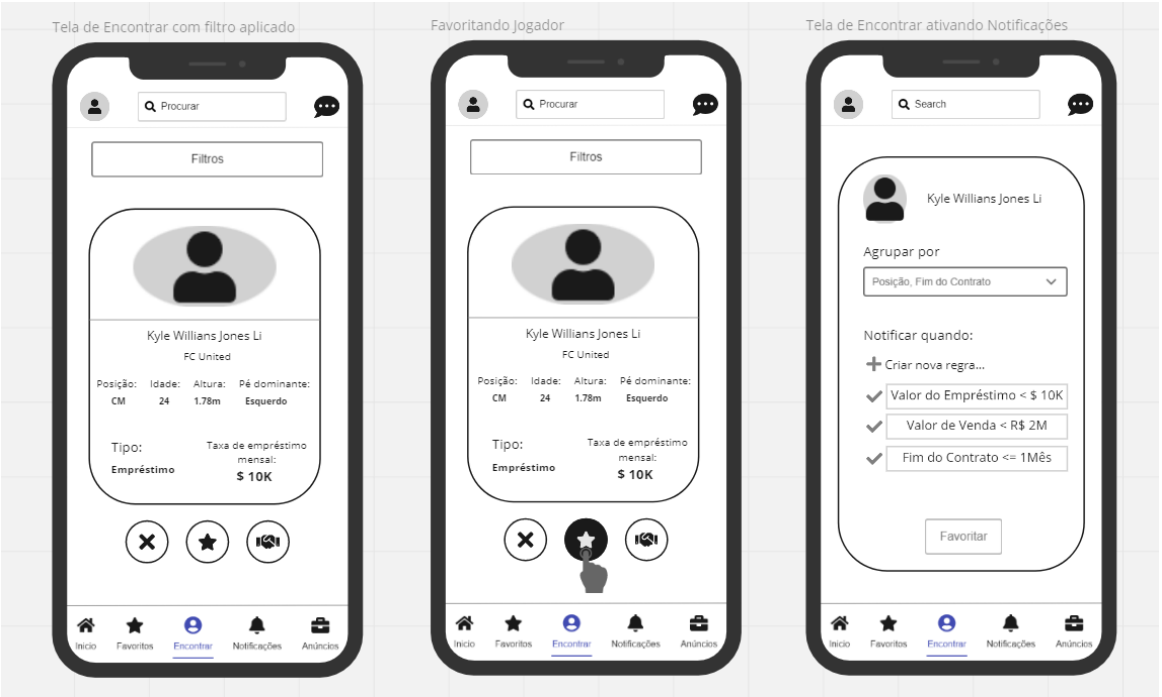

Figura 31 - Tela de Encontrar com filtro Aplicado, Favoritou Jogador e ativando Notificações da TransferWindow

Nesse menu, a última funcionalidade é a oferta propriamente dita. A partir do momento que o administrador demonstrou interesse pelo atleta, ele pode sinalizar essa vontade pelo jogador por meio do botão de **“Aceitar Oferta”** representado pelo ícone de aperto de mãos, ou arrastando o cartão para direita, na tela de **Aceitando Proposta.** Se aceitar a oferta, na tela de Aceitou Proposta, uma mensagem personalizada que será introduzida na seção de Chat aparecerá para o anunciante e assim a negociação pode dar andamento, se for interesse de ambas as partes. A partir daí, um outro jogador aparece, dando continuidade ao processo de busca na tela de Encontrar com Novo Jogador.

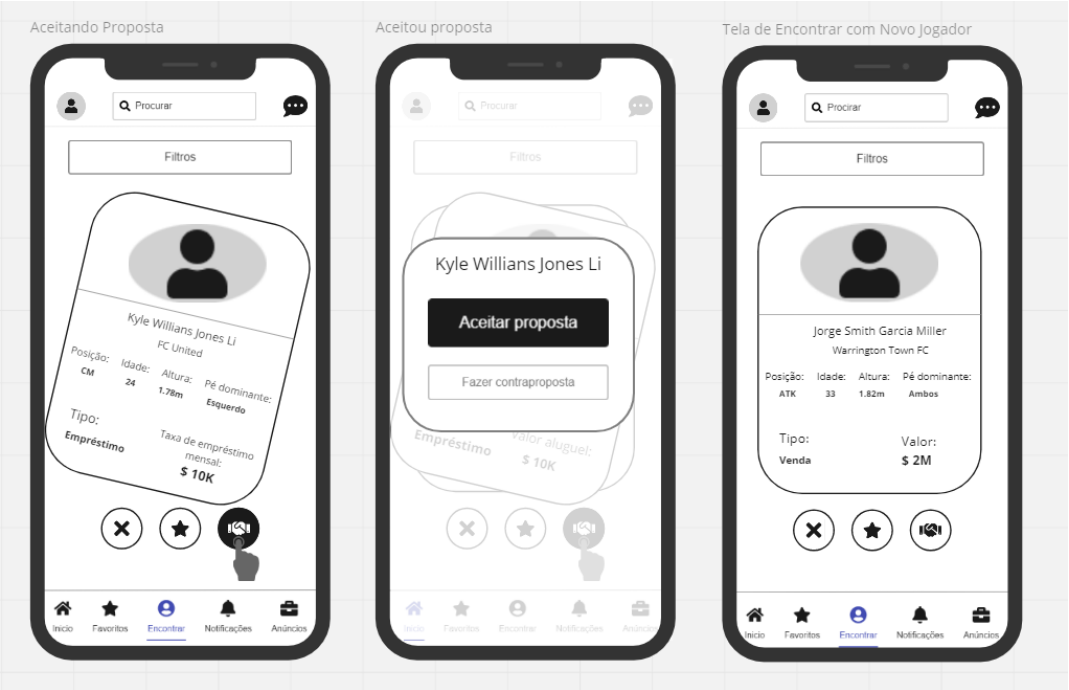

Figura 32 - Tela de Aceitando Proposta, Aceitou proposta e Encontrar com Novo Jogador da TransferWindow

Decidindo fazer uma contraproposta (Tela Fazendo Contraproposta), o usuário tem a capacidade de sugerir modificações ou até mesmo mudar por completo a proposta do atleta (Tela Personalizando Nova Proposta). Da mesma forma que antes, a partir do momento que a nova oferta é enviada, uma mensagem personalizada com as novas características levantadas pelo interessado é direcionada ao anunciante (clube ou empresário), e assim a negociação pode dar andamento, caso seja interesse de ambas as partes..

Importante frisar que, como levantado por muitos players no mercado esportivo, existe uma preocupação em proteger as informações divulgadas já que elas podem acabar entregando decisões táticas e estratégicas dos times. Por isso, o administrador também pode aceitar uma proposta ou fazer uma contraproposta de forma anônima, até a negociação sair das etapas iniciais, protegendo assim os interesses do clube.

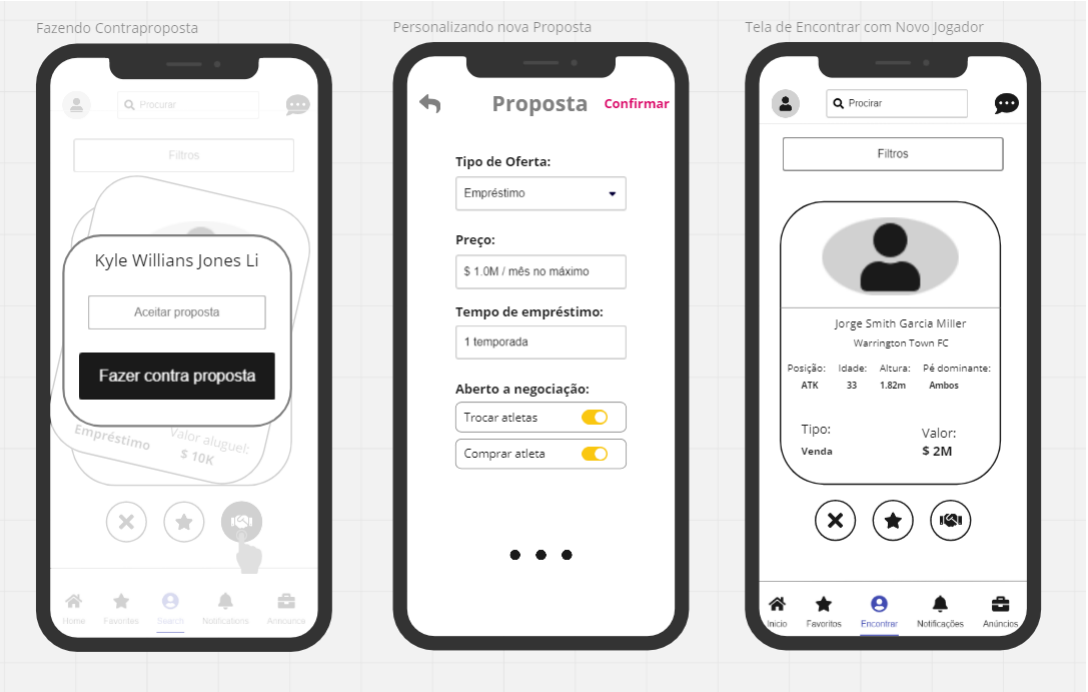

Figura 33 - Fazer Contraproposta e Configurar da TransferWindow

6.3.3 Página Inicial

Os atletas favoritados podem ser vistos tanto na Seção de Favoritos quanto na Página Inicial. A partir desse momento, o usuário passa a seguir o jogador e consegue ter acesso a postagens que o mesmo fizer (Tela Inicial com conteúdo). Diferentemente do instagram e linkedin, por exemplo, o foco da ***TransferWindow*** é criar um ambiente exclusivo e focado no mercado esportivo de futebol. Dessa forma, a solução não se preocupou em se aproximar dos modelos de negócio propostos pelas redes acima citadas, como publicidade para geração de receitas ou preferências de usuários para ofertas de produtos diversos.

De início, fica patente a diferença entre as aplicações, dado que a ***TransferWindow*** é voltada para o matchmaking de negócios e tem como objetivo o relacionamento e contato direto entre interessados, não permitindo uma livre circulação do usuário na aplicação em função de ser estruturado e com permissionamento. Sendo assim, a importância da página inicial é possibilitar que os usuários possam criar uma vitrine das atividades que estão realizando, treinamentos, grandes jogadas, entre outros motivos na tentativa de se destacar e conseguirem melhores propostas ou até mesmo ingressar no mundo esportivo.

Na figura 16, uma rápida ilustração de como vai ser a página inicial vazia e como ficará a partir do momento que novas pessoas forem favoritadas. O objetivo dessa rede não é se destacar em popularidade, e portanto dados como quantos seguidores e quantas pessoas o usuário segue não são tão relevantes, e por isso essas informações não possuem destaque.

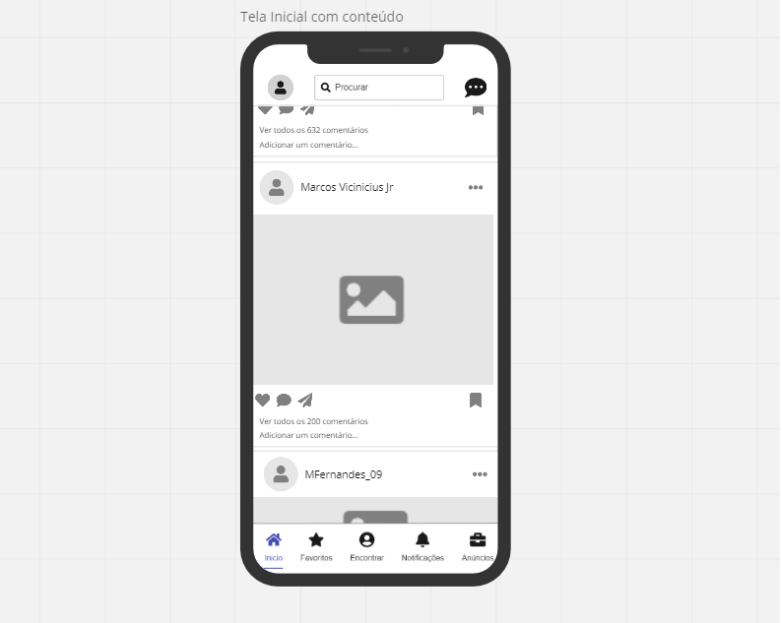

Figura 34 -Tela Inicial com conteúdo da TransferWindow

6.3.4 Favoritos

O usuário pode criar diferentes tabelas, cada uma delas com a possibilidade de organizar os jogadores de acordo com as características de preferência, permitindo assim definir de forma estratégica qual atleta ficará em cada tabela(Tela de Favoritos). Também é possível favoritar clubes e empresários, e agrupá-los da forma que preferir. Quando alguém que está nos favoritos fizer alguma postagem ou mudar alguma informação, um sinal sinalizará que existe conteúdo novo.

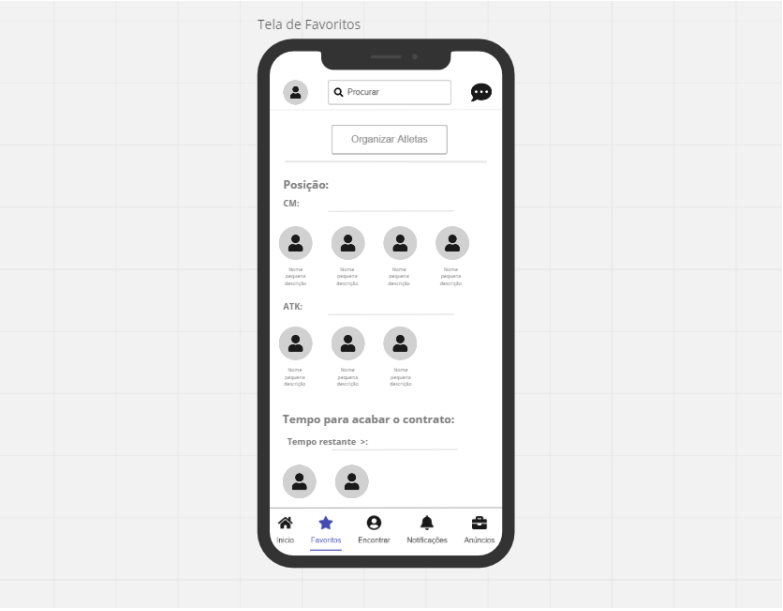

Figura 35 - Tela de Favoritos da TransferWindow

6.3.5 Notificações

Muitas aplicações atualmente, na busca de criar um maior engajamento em suas plataformas, acabam por sobrecarregar o usuário com mensagens. O problema é que, grande parte dessas notificações não agregam em nada, por serem de baixa relevância como, por exemplo, mensagens que apontam tempo de inatividade ou ausência na plataforma. Situações como essa, por vezes, geram resultado oposto ao esperado e acabam fazendo com que o usuário as desabilite. Portanto, na ***TransferWindow***, apenas as notificações que foram criadas pelo usuário serão enviadas. A vantagem é manter a experiência do usuário com o app mais eficiente e menos maçante.

Uma possível situação seria um olheiro poder criar uma regra para receber uma notificação de determinado clube quando este anunciar um novo jogador. Através desta facilidade, o olheiro, terá conhecimento dessa novidade no mercado e poderá agir da forma mais efetiva possível, caso seja do seu interesse. O objetivo é diminuir as distrações com assuntos pouco relevantes, manter o foco nas negociações e fomentar a criar novas regras, ao invés de retirar as notificações do aplicativo.

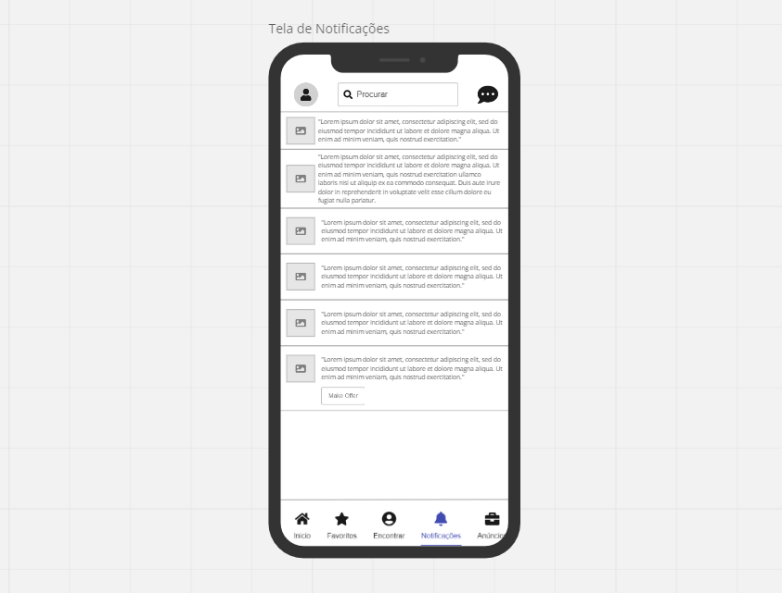

Figura 36 - Tela de Notificações da TransferWindow

6.3.6 Anúncios

O administrador ou qualquer outro usuário que tenha permissão pode adicionar os jogadores que fazem parte do próprio time. Depois de fazer a inserção deles na plataforma, é possível anunciá-los na seção de Anúncios, e verificar as ofertas que já foram criadas para outros jogadores (Tela de Anúncios). Quando o usuário decide fazer um novo anúncio, como se pode observar na figura 19, é possível definir as características da oferta, relacionando-as a qual será o tipo de anúncio, o valor desejado, tempo de contratação do atleta, entre outras informações (Tela de configurar anúncio).

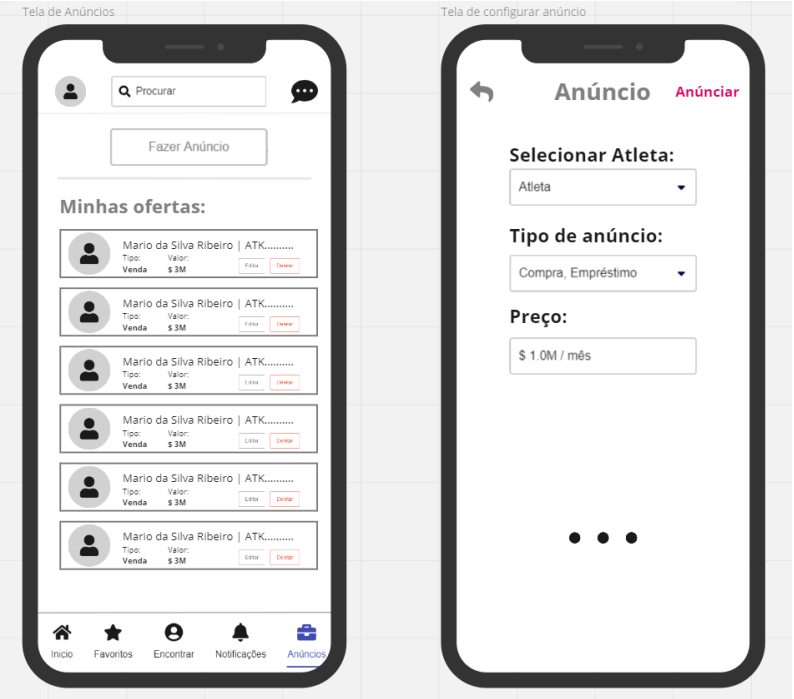

Figura 37 -Tela de Anúncios e Configurar Anúncio da TransferWindow

6.3.7 Chat

A funcionalidade do chat já é de maneira geral auto explicativa. Nele é possível se comunicar com outros clubes, empresários e jogadores. O diferencial vem da capacidade de agrupar conversas em relação ao assunto(Tela de Mensagens Agrupadas por Assunto). Um exemplo de fácil visualização é quando um clube anuncia um jogador e o administrador não deseja que o chat se encontre poluído com uma grande quantidade de mensagens de pessoas diferentes, cada uma com sua proposta. Sendo assim, é possível agrupar todas essas conversas por assuntos. No caso citado como exemplo, o assunto seria a oferta do jogador.

A partir do momento que um clube interessado enviar uma proposição para outro clube, com os seus termos para o acordo, uma mensagem será encaminhada para o responsável pela proposta do jogador, com as novas sugestões de mudança(Conversa com usuário). O responsável então poderá analisar a oferta encaminhada, e decidir se começa uma conversa com o clube ou não.

Figura 38 - Tela de Mensagens e Conversa com usuário da TransferWindow

6.3.8 Perfil

Na TransferWindow, é possível gerenciar tanto as subcontas que estão atreladas a determinado administrador, como gerenciar os atletas pertencentes ao clube. O uso de subcontas é essencial para delegar as funções a outros integrantes do time de busca de talentos nos clubes, sem que todos tenham que necessariamente usar o mesmo perfil. Outro ponto é que, com o permissionamento, o administrador consegue ter controle do que cada um dos sub-usuários pode fazer, aumentando a segurança da aplicação como um todo.

Portanto, quando o administrador clica no seu perfil, ele consegue visualizar a aba de gerenciar subcontas e gerenciar atletas (Tela de Perfil). Selecionando uma conta, é possível escolher os tipos de acesso e direitos que a mesma possui, mantendo um controle maior do papel que cada integrante da equipe realiza na aplicação. Caso haja pessoas que possuam as mesmas permissões, é possível criar times para evitar o preenchimento manual de cada usuário.

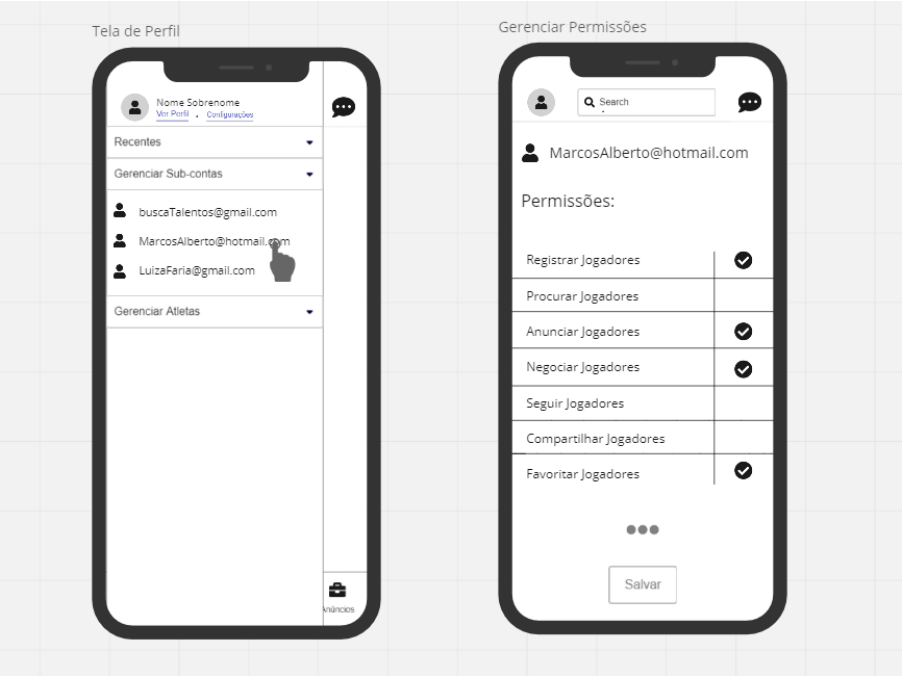

Figura 39 -Tela de Perfil e Gerenciar Permissões da TransferWindow

O administrador ou usuário responsável também pode gerenciar os atletas que fazem parte da base de dados do clube em questão(Gerenciar Atletas). Selecionando um jogador, ele tem acesso a todas as suas estatísticas, dados gerais e permissão para adicionar, editar ou remover anúncio de determinado atleta(Perfil do Atleta). Descendo a página, é possível ainda ver as imagens e vídeos referentes ao mesmo, que compõem o feed do atleta(Perfil do Atleta - complemento).

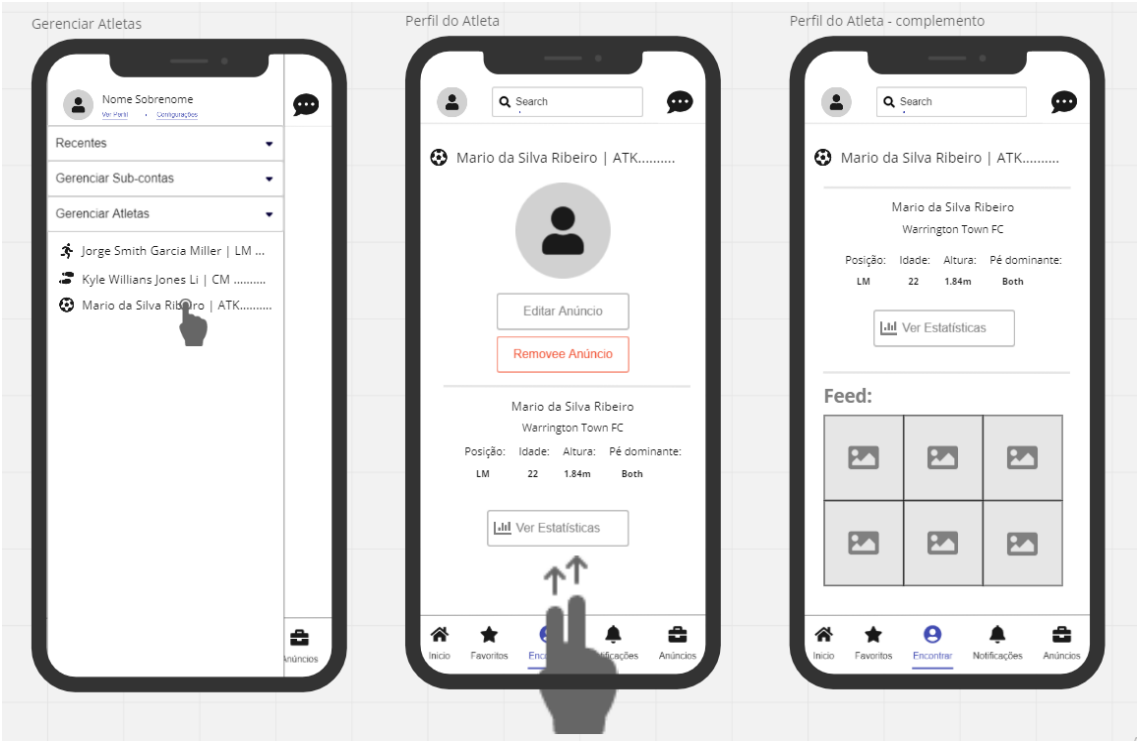

Figura 40 - Gerenciar Atletas e Perfil do Atleta da TransferWindow

7.Projeto

A TransferWindow é uma aplicação que tem como objetivo profissionalizar a transferência de jogadores de futebol no Brasil. Neste projeto, o objetivo principal foi fazer uma rápida validação com o público-alvo, e executar todas as etapas do processo de desenvolvimento de software.

A proposta inicial do projeto era criar uma aplicação IOS, onde usuários de diferentes segmentos desse mercado(Clubes, Empresários e Jogadores), pudessem interagir em um único ambiente digital. Devido ao tamanho da equipe, e do prazo estipulado nesta etapa atual, o escopo teve que ser reduzido para interação apenas entre os Clubes, porém as funcionalidades serão mantidas, reduzindo apenas os tipos de usuários, e portanto, a quantidade de layouts de tela personalizados para cada um.

Sendo assim, o projeto é composto por quatro partes: a arquitetura do projeto, onde será apresentado a arquitetura que mais se encaixou com a proposta desse aplicativo, sendo uma etapa essencial pois essa estrutura terá um papel fundamental de tornar o sistema robusto, com bom desempenho e escalável; definição e explicação dos padrões de projeto que foram usados no sistema; o front-end, a linguagem utilizada, o padrão de arquitetura MVC e como o sistema está se comportando; e, por fim, o diagrama de classes, que representa a estrutura e relações das classes que servem de modelo para objetos da aplicação.

7.1 Arquitetura

A arquitetura definida para essa aplicação foi a **serverless[15]**. Essa é uma arquitetura orientada a eventos, e possui como principal proposta, permitir que as empresas de software, e mais especificamente, os programadores desenvolvam e mantenham suas aplicações, sem se preocupar com a infraestrutura em que estão executando. Como atualmente o projeto está sendo desenvolvido apenas por mim, não precisar administrar a infraestrutura de servidores permite que eu dedique mais tempo às funções primárias e mais importantes da aplicação, aumentando a minha capacidade de entrega.

Serverless é um modelo de execução de computação em nuvem, onde o provedor que, nesse caso é o Firebase, gerencia dinamicamente a alocação de servidores para atender a demanda da aplicação. Esse provedor vai ser responsável por gerenciar a capacidade de processamento, sistemas de armazenamento, provisionamento, atualização de servidores, entre outras configurações recorrentes. Nesse modelo, a lógica da aplicação fica no lado do cliente (*client-side logic*), enquanto que a segurança, o banco de dados e o back-end são terceirizados (*third-party services*) e administrados nesse caso pelo Firebase. Na figura 23, é possível visualizar uma abstração de como é essa arquitetura.

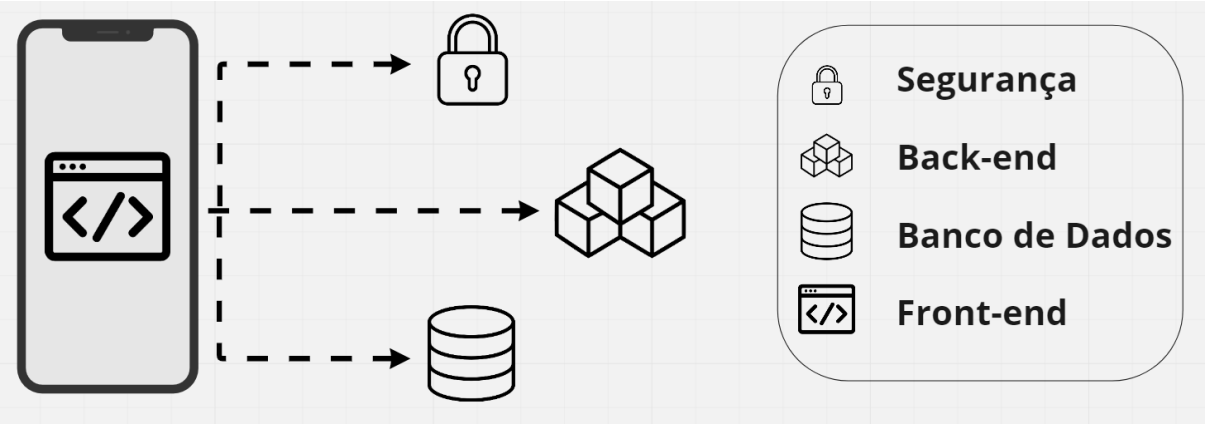

Figura 41 - Arquitetura Serverless

Além da questão do gerenciamento, outra vantagem é o custo reduzido. Já que o modelo de cobrança é baseado em uso, principalmente no período de desenvolvimento e crescimento do aplicativo, o custo vai ser muito pequeno ou nulo. E, caso no futuro seja necessário escalar, esse processo para Serverless é automático e contínuo. Outro ponto é em relação a configuração de ambientes diferentes, já que não existe a necessidade de configurar máquinas para desenvolvimento, teste e produção.

O Firebase foi escolhido devido aos produtos e soluções que ele oferece, que são extremamente importantes para o crescimento da aplicação. Foram utilizados, o *Cloud Firestore*, que é o banco de dados NoSQL para armazenamento de dados. O *Authentication*, que oferece autenticação de usuário e é compatível com as regras de segurança do *Firestore*, na hora de acessar os dados. O *Crashlytics*, uma ferramenta de relatório de erros em tempo real para correção de falhas. E por fim, o *A/B Testing*, que será implementado futuramente, mas que utiliza uma análise bayesiana[7] no back-end, que ajuda a determinar se alterações no aplicativo, como a mudança de um estilo de interface para outro, melhoraram o desempenho e se os resultados possuem maior relevância.

7.2 Padrões de Projeto

Em Engenharia de Software, Design Patterns ou padrões de projetos são soluções generalistas para problemas que acontecem frequentemente durante o desenvolvimento de um software. Não se trata de um código pronto ou um framework, mas de uma definição de alto nível de como um problema que é recorrente pode ser resolvido. São modelos que já foram utilizados, testados e validados anteriormente, e portanto geralmente apresentam um bom ganho de produtividade para os desenvolvedores.

Seu uso facilita na estruturação e manutenção das aplicações, já que esses padrões se baseiam em baixo acoplamento entre as classes e padronização do código[16]. Conhecer e utilizar padrões de projeto é algo muito importante no desenvolvimento de Software e esta técnica foi

[7] A inferência bayesiana consiste na avaliação de hipóteses pela máxima verossimilhança.

aplicada no projeto em questão. Seu uso auxilia o desenvolvimento de forma mais rápida quando a necessidade do projeto se depara com desafios já conhecidos no universo de programação; é capaz de prover uma linguagem comum durante a documentação ou quando são encontradas dúvidas técnicas; e adicionalmente contribui de forma definitiva para a organização do código que está sendo escrito ou desenvolvido.

7.2.1 Coordinator

Antes de começar a explicar o padrão de projeto, vamos começar visualizando o problema que enfrentamos. Na linguagem *Swift*, quando se começa o desenvolvimento *IOS*, todos se deparam com a *UIViewController*. Ela está no centro de tudo que fazemos quando programamos em *IOS* e todos os aplicativos contém pelo menos uma subclasse personalizada de *UIViewController*. Entretanto, é frequente que os aplicativos contenham muitos controladores de exibição personalizados (*UIViewControllers*).

Estes controladores de exibição definem os comportamentos gerais do aplicativo, incluindo sua aparência e como ele responde às interações do usuário. Porém, esse excesso de funcionalidades pode gerar o que é chamado de *MassiveViewControllers*, considerando que em uma classe pode se designar as tarefas de:

- Design de interface do usuário(*UI*);
- Atualização das UI*Views* com os dados do modelo;
- Responder os inputs do usuário;
- Salvar e Restaurar estados da aplicação;
- *Data sources* e *delegates*;
- Animações;
- *Networking*;
- Navegação entre telas;
- Outras funções;

Tarefas como animações e UI design podem ser facilmente tratadas com o objetivo de evitar o *MassiveViewControllers,* colocando-as dentro de uma subclasse *UIView.* Da mesma forma, conseguimos colocar a tarefa de criação de *data sources* e *delegates* em outras classes. Porém é preciso resolver o problema da navegação, que é responsável pelo controle de fluxo de telas do aplicativo, além de ser necessário lidar com testes entre *ViewControllers* e com variantes relacionadas a diferentes dispositivos.

Caso a lógica de fluxo esteja codificada no próprio controlador de exibição, para chamar uma nova tela, o próprio controlador de exibição deve conter a informação sobre a próxima tela, fazer a configuração e criar a próxima *UIViewController*. Isto gera um forte acoplamento entre esses controladores. Como esse flow está *hard-coded* [8], nesse cenário, o desenvolvedor perde a

[8] Fixa (dados ou parâmetros) em um programa de tal forma que eles não podem ser alterados sem modificar o programa.

capacidade de reorganizar ou reusar esses controladores. Na figura 24, é possível ver a representação de como funciona essa comunicação e seu acoplamento.

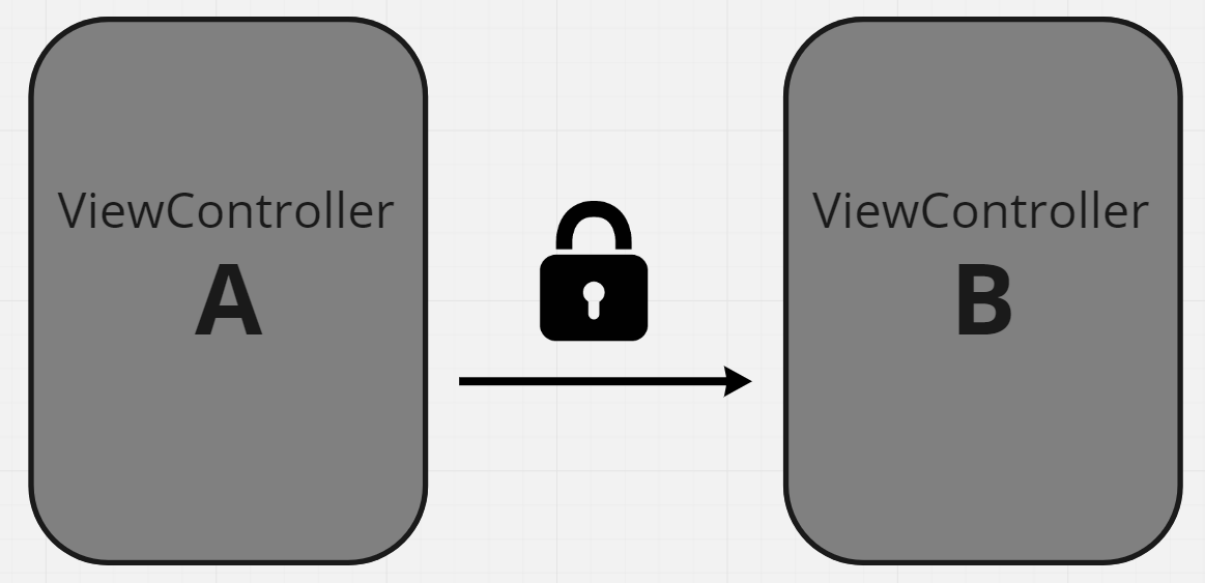

Figura 42 - Fluxo de navegação através de UIViewControllers

Portanto, a solução será utilizar o padrão de projeto *Coordinator* que vai substituir o controle de fluxo dos controladores por um objeto separado que cuidará da navegação de forma clara e eficiente. Dessa forma, o controlador A não tem conhecimento da existência do controlador B, já que agora a comunicação é feita diretamente com o protocolo, e ele vai ser responsável por tomar ações apropriadas e chamar a próxima tela. Na figura 25, é possível ver como funciona a nova comunicação com o uso do *Coordinator*.

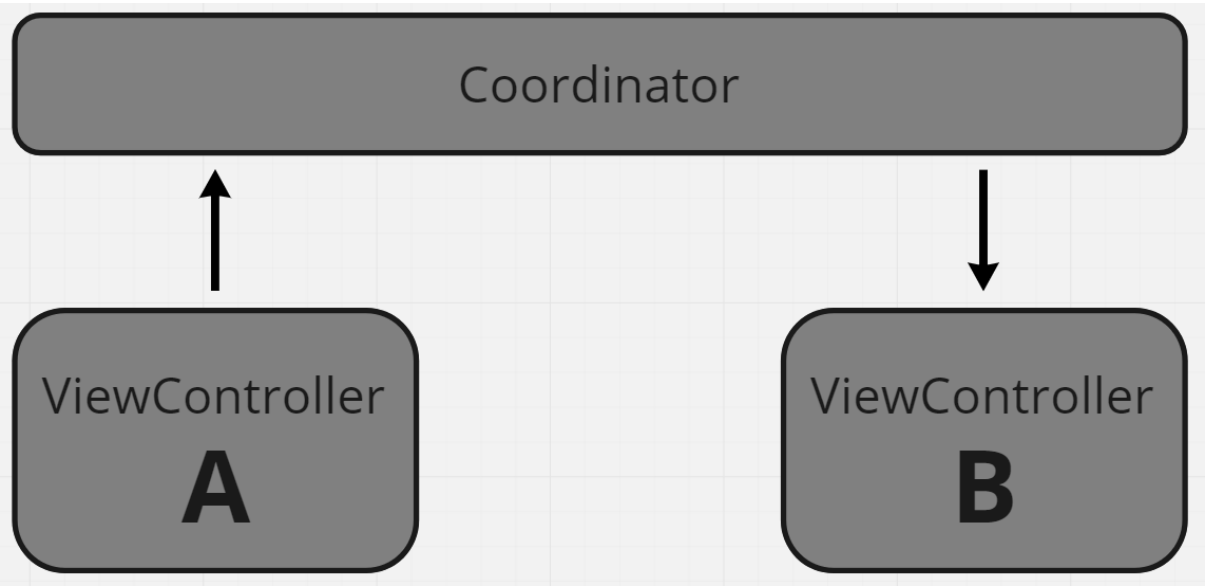

Figura 43 - Fluxo de navegação através do Coordinator Design Pattern

Além disso, é possível quebrar a aplicação em pequenos subcoordenadores que vão lidar cada um com uma parte da jornada do usuário, como por exemplo um coordenador que vai cuidar

de todo o fluxo de telas da aba de Encontrar Jogador e outro que cuidará do fluxo da página principal. Com essa mudança de fluxo, agora os controladores podem ser reutilizados na aplicação, tendo em vista que estão completamente isolados um do outro. E agora cada tarefa ou subtarefa no aplicativo tem uma maneira dedicada de ser encapsulada.

7.2.2 Singleton

Singleton é um padrão de projeto criacional que garante que uma classe possua apenas uma instância, enquanto provê um ponto de acesso global para essa instância. A vantagem desse padrão é conseguir controlar de forma mais eficiente o acesso a recursos compartilhados que existem nesta aplicação, como por exemplo, uma base de dados ou um arquivo específico. Não é possível repetir esse comportamento em um construtor regular, dado que uma chamada a ele deve sempre retornar um novo objeto por design.

Outro problema que é evitado pelo uso desse design pattern é a criação de variáveis globais na aplicação. Apesar de serem úteis em muitos casos, elas também apresentam um alto nível de insegurança para a aplicação, uma vez que qualquer código pode potencialmente sobrescrever algum conteúdo dessas variáveis e quebrar o app. Mas por ser bem mais prático e consistente, será mantido esse código com variáveis globais, já que resolvem problemas de vários View Controllers dentro de uma classe do que espalhados por todo o programa.

Um exemplo prático utilizado nesta aplicação foi a classe *FirebaseDataManager*. Essa classe serve como uma camada intermediária da aplicação para a utilização dos serviços do Firebase relacionados a envio e recebimento de dados. Como a função dela é enviar uma requisição ao Banco de Dados e devolver o resultado, não existe uma necessidade de criar novas instâncias dessa classe, e ainda contribui para redução do consumo de memória.

7.2.3 Observer

Vamos visualizar um problema que enfrentamos antes de apresentar as vantagens desse padrão de projeto e a explicação de como funciona. Dentro da aplicação, existem dois tipos de objetos: um sendo o Clube e o outro o Jogador. O clube está interessado em um jogador específico que, em breve, ficará disponível para contratação e portanto seu clube contratante pode a qualquer momento criar um lance para o atleta. O clube interessado poderia acessar todos os dias, várias vezes, a página do clube que detém esse jogador para ver se ele já está disponível para contratação.

Por outro lado, o clube contratante poderia enviar milhares de notificações, para todos os clubes, sempre que algum jogador se tornasse disponível ou houvesse uma alteração na proposta de algum jogador. Por mais que ajudasse alguns interessados a não precisar visitar a página do clube contratante, frequentemente, esse método atrapalharia todo o resto que não tem interesse. Portanto, existe um conflito, já que ou o clube interessado investe tempo verificando a disponibilidade do jogador ou o contratante investe recursos notificando os clubes errados.

A solução para esse impasse é o uso do padrão de projeto comportamental *Observer*, que permite definir um mecanismo de assinatura para notificar objetos sobre quaisquer mudanças que ocorram com o objeto que eles estão observando. Este padrão envolve três tipos: O *subscriber*, que é o objeto "observador" e recebe atualizações; O *publisher,* que é o objeto "observável" e envia atualizações; E o *valor*, que é o objeto que foi alterado. Na figura 26, é possível visualizar uma representação de como seria a comunicação entre esses tipos.

Figura 44 - Comunicação entre tipos do Observer Design Pattern

Esse padrão sugere um mecanismo de assinatura para os clubes contratantes, para que os objetos individuais, no caso os clubes interessados, possam assinar ou desassinar eventos advindos dessa *publisher*. Agora, sempre que um evento importante acontece com a publicadora, ele passa para seus *subscribers* e chama um método específico de notificação em seus objetos, já que existe um vetor armazenando uma lista de referências aos objetos dos assinantes.

Outro ponto positivo é que permite que qualquer outro objeto que implemente essa interface de assinante possa se inscrever para notificações de eventos em objetos da publicadora. Se torna então mais fácil escalar a aplicação, dado que é possível introduzir novas classes assinantes sem precisar mudar o código da *publisher.* E para conclusão do exemplo, o clube interessado pode receber a notificação apenas do jogador, quando o mesmo estiver disponível para negociação.

7.3 Front-end

O front-end é uma interface IOS que foi desenvolvida em Swift, seguindo os princípios de arquitetura MVC. Como a aplicação é serverless, todo o processamento de dados é feito no lado do cliente. Nesse caso o front-end se comunica diretamente com o banco de dados, ao invés de regularmente haver uma camada intermediária de back-end, que seria responsável pela criação de rotas e comunicação com o banco de dados através de um framework.

O diagrama da figura a seguir mostra como as camadas de front-end estão sendo divididas. Se olharmos para o padrão de arquitetura MVC (*Model View Controller*), podemos entender as *UIViews* como as *Views*. São responsáveis apenas por apresentar as informações de forma visual ao usuário, seja pela criação dos objetos da tela ou as animações que ocorrem. Toda ação que a *View* recebe, ela transmite diretamente para seu controlador, e entrega as respostas obtidas ao usuário.

As *UIViewControllers* são os *Controllers* da aplicação, pois são responsáveis por intermediar as requisições enviadas pela *View* com as respostas fornecidas pelo *Model*. Sua tarefa é processar os dados enviados pelo usuário e repassar para outras camadas. Os dados retornados das chamadas às API respeitam os formatos definidos previamente nos modelos e interfaces, que representam o *Model* da arquitetura. Portanto, a *UIViewControllers* vai gerenciar e controlar a forma que os dados se comportam, através da lógica, regras de negócio e funções.

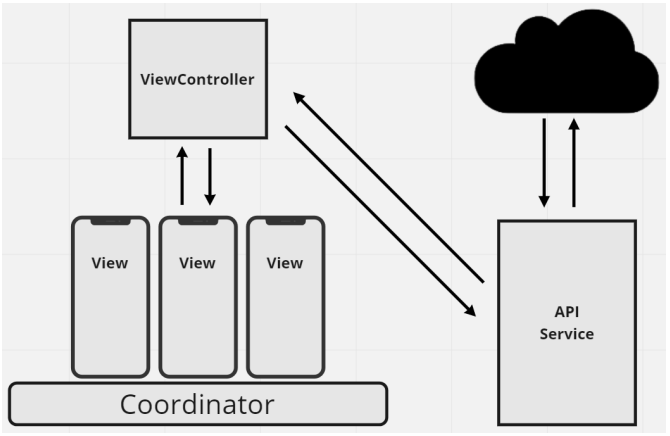

Figura 45 - Arquitetura do Front-End

O *API Service* é um conjunto de classes que tem como objetivo realizar as requisições ao provedor. A classe *AuthManager* tem a responsabilidade de cuidar de todas as ações envolvendo autorização de usuário, seja, entrada e saída no sistema, troca de senha, verificação de email entre outros. As Classes *DataManager* e *StoreManager* são responsáveis por todas as ações envolvendo gerenciamento de informações e assets, respectivamente. O objetivo de criar classes específicas com determinadas atribuições, é manter um código organizado e limpo.

7.4 Diagrama de Classes

Neste projeto foi escolhido o uso de diagrama de classes por serem fundamentais para o processo de modelagem de objetos e por apresentarem toda a estrutura estática do sistema. Isso porque são responsáveis por modelar os objetos, exibir seus relacionamentos, descrever o que fazem dentro da aplicação e que serviços oferecem. A partir de sua elaboração, foi possível visualizar as necessidades específicas do sistema, e implementar de forma mais eficiente as classes na hora de desenvolver o código. Na figura 46, está disponível a versão mais recente da diagramação.

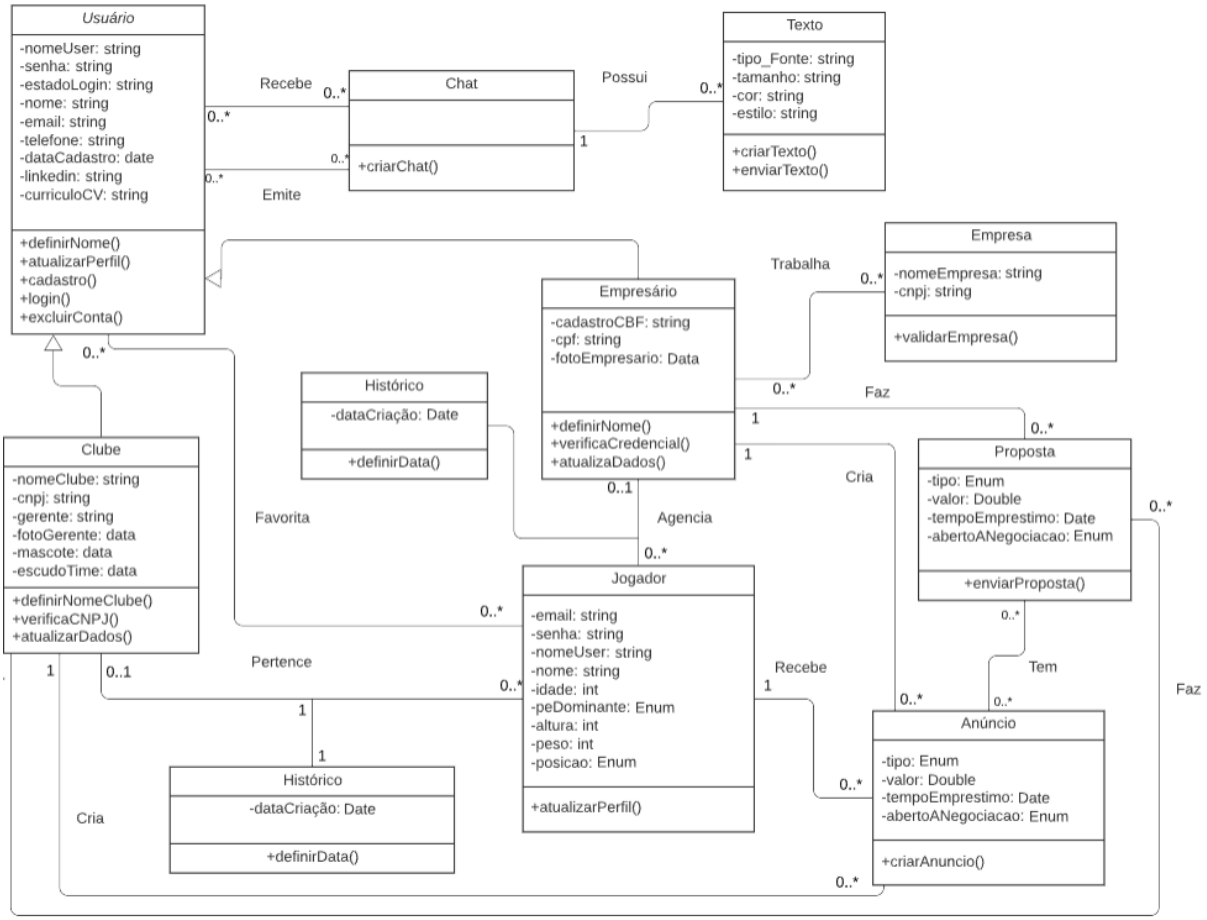

Figura 46 - Diagrama de Classes

O diagrama apresentado na figura 46 conta com uma classe Usuário, no lado superior esquerdo, que armazena os dados básicos, como nome, nome de usuário, senha, email, telefone, entre outros. Existem duas classes que herdam as características dessa classe, que são os Clubes e os Empresários. Essas classes possuem seus próprios atributos específicos, e são responsáveis por fazer o "CRUD"[9] dos seus jogadores. Os empresários também podem associar suas contas a uma empresa a qual fazem parte.

Tanto os Clubes quanto os Empresários podem anunciar os seus jogadores ao mercado. Dentro da proposta, pode ser escolhido o tipo de oferta, o valor que será negociado e o período de validade da proposta. Também possuem a funcionalidade de fazer oferta para jogadores que se apresentam disponíveis. Se uma oferta for feita, o usuário que recebeu a oferta será notificado. A partir disso um Chat será inicializado e sua classe será composta pelo texto da conversa e futuramente fotos e áudios.

Clubes e Empresários também podem se "favoritar" e "favoritar" jogadores. Dentro das regras de negócio é possível estabelecer triggers de notificações, caso queiram saber de mudanças relacionadas a algum usuário específico. Os administradores dos clubes podem gerenciar a conta

[9] CRUD são as quatro operações básicas(create, read, update e delete) utilizadas em bases de dados relacionais fornecidas aos utilizadores do sistema.

de outros integrantes e determinar suas permissões dentro da plataforma. A partir do permissionamento, é possível rastrear as atividades das sub-contas dentro do aplicativo.

7.5 Dicionário de Atributos

O dicionário de atributos tem como finalidade apresentar de forma clara e objetiva os motivos de cada classe descrita no diagrama de classes possuir atributos específicos. O dicionário desenvolvido irá conter o nome do atributo, a sua descrição e o seu domínio. Além disso, como os casos de uso foram utilizados para a execução desta etapa, também serão feitas breves associações entre essas propriedades e os casos de uso elaborados na seção 6.1 de Especificação.

Classe **Usuário**:

- *nomeUser*: O nome de usuário é um apelido que é utilizado para acessar a aplicação e para identificar o usuário dentro da plataforma. O *Usuário.nomeUser* precisa ter valor único, possuir mais de 6 caracteres e não pode ter caracteres especiais e caso seja modificado precisa ser verificado se já existe alguém com o mesmo apelido. Está expresso na descrição UC03, UC04, UC06, UC07 e UC08.
- *senha*: A senha será utilizada pelo usuário para poder acessar a aplicação, ou quando precisar tomar alguma atitude irreversível, como apagar a conta. O *Usuário.senha* deverá possuir mais de 8 caracteres (contendo letras maiúsculas, minúsculas, caracteres especiais e números). Está expresso na descrição UC03, UC04 e UC06.
- *nome*: O nome completo do usuário. O *Usuário.nome* não pode ser vazio. Está expresso na descrição UC02.
- *email*: O email é utilizado para acessar a aplicação e para identificar o usuário dentro da plataforma. O *Usuário.email* só pode ser usado uma vez e caso seja alterado é preciso fazer a confirmação do novo email. Está expresso na descrição UC03, UC04 e UC06.
- *telefone:* O telefone é utilizado na plataforma apenas para compartilhamento rápido com outro usuário. O *Usuário.telefone* é opcional. Está expresso na descrição UC02 e UC04.
- *linkedin:* O linkedin é utilizado na plataforma apenas para compartilhamento rápido com outro usuário. O *Usuário.linkedin* é opcional. Está expresso na descrição UC02 e UC04.
- *curriculoCV:* O currículo é utilizado na plataforma apenas para compartilhamento rápido com outro usuário. O *Usuário.curriculoCV* é opcional. Está expresso na descrição UC02 e UC04.

Classe **Clube** (herda as características da classe Usuário):

- *cnpj:* O CNPJ do clube, para fazer a identificação do negócio. O *Clube.cnpj* precisa ser um CNPJ válido. Está expresso na descrição UC02 e UC07.

- *gerente:* O atual gerente do clube. O *Clube.gerente* não pode ser nulo e só pode ser alterado junto com a foto do gerente. Está expresso na descrição UC02.
- *fotoGerente:* A foto do jogador que será utilizada dentro da plataforma para fazer sua identificação. O *Clube.fotoGerente* não pode ser vazio e precisa ter um tamanho de 152x152 pixels. Está expresso na descrição UC02.
- *mascote:* A foto do mascote do clube. O *Clube.mascote* pode ser nulo. Está expresso na descrição UC02.
- *escudoTime:* A foto do escudo do clube. O *Clube.escudoTime* não pode ser nulo e precisa ter um tamanho de 212x212 pixels. Está expresso na descrição UC02.
- *nomeClube:* O nome que será apresentado publicamente. O *Clube.nomeClube* não pode ser nulo. Está expresso na descrição UC02 e UC07.

Classe **Empresário** (herda as características da classe Usuário):

- *credencialCBF:* A credencial do empresário dentro da CBF. O *Empresário.credencialCBF* precisa ser uma credencial válida. Está expresso na descrição UC02 e UC08.
- *cpf:* O cpf do empresário. O *Empresário.cpf* precisa ser um cpf válido. Está expresso na descrição UC02 e UC08.
- *fotoEmpresario:* A foto do empresário que será utilizada dentro da plataforma para fazer sua identificação. O *Empresario.fotoEmpresario* não pode ser vazio e precisa ter um tamanho de 152x152 pixels. Está expresso na descrição UC02.

Classe **Jogador**:

- *nome:* O nome completo do jogador. O *Jogador.nome* não pode ser vazio. Está expresso na descrição UC01.
- *idade:* A idade do jogador. O *Jogador.idade* não pode ser vazio e tem que pertencer a faixa etária de 17 a 50 anos. Está expresso na descrição UC01 e UC09.
- *peDominante:* O pé dominante do jogador. O *Jogador.peDominante* não pode ser vazio. Está expresso na descrição UC01.
- *altura:* A altura do jogador. O *Jogador.altura* não pode ser vazio e precisa estar entre 150cm e 210cm. Está expresso na descrição UC01 e UC09.
- *peso:* O peso do jogador. O *Jogador.peso* não pode ser vazio e precisa estar entre 50kg e 110kg. Está expresso na descrição UC01 e UC09.
- *posicao:* A(s) posição(ões) que o jogador atua dentro de campo. O *Jogador.posicao* não pode ser vazio. Está expresso na descrição UC01.
- *fotoJogador:* A foto do jogador que será utilizada dentro da plataforma para fazer sua identificação. O *Jogador.fotoJogador* não pode ser vazio e precisa ter um tamanho de 152x152 pixels. Está expresso na descrição UC01 e UC09.

Classe **Proposta**:

- *valor:* O valor que se pretende negociar o jogador. A *Proposta.valor* não pode ser nula e caso a proposta seja editada, o valor precisa ser maior do que a proposta anterior. Está expresso na descrição UC14.
- *tipo:* O tipo de proposta que foi feita (empréstimo ou venda). A *Proposta.tipo* precisa ser uma das duas opções. Está expresso na descrição UC14.
- *tempoEmprestimo:* O tempo de duração do empréstimo. A *Proposta.tempoEmprestimo* não pode ser nulo caso o tipo seja empréstimo e o tempo de empréstimo precisa ser maior que 12 meses. Está expresso na descrição UC14.
- *abertoANegociacao:* Variável que indica se está aberto a novas propostas. A *Proposta.abertoANegociacao* não pode ser nula. Está expresso na descrição UC14.

Classe **Empresa**:

- *nomeEmpresa:* O nome da empresa em que o empresário está associado. A *Empresa.nomeEmpresa* pode ser nulo. Está expresso na descrição UC08.
- *cnpj:* O CNPJ da empresa, para fazer a identificação do negócio. O *Empresa.cnpj* precisa ser um CNPJ válido. Está expresso na descrição UC08.

Classe **Texto**:

- *tipoFonte:* O tipo da fonte usada no texto.
- *tamanho:* O tamanho da letra que está sendo usada no texto.
- *cor:* A cor do texto.
- *estilo:* O estilo que vai ser usado no texto.

8.Construção

Para a implantação deste sistema, foi utilizado um serviço do tipo BasS (Backend as a Service) da nuvem do Firebase. O BasS soluciona um grande problema que é depender de ferramentas complexas e linguagens de programação para trabalhar um backend da aplicação. Utilizando o Firebase, é possível focar no front-end, se concentrando nas regras de negócio, usabilidade do sistema e experiência do usuário. Outro aspecto importantíssimo para a execução da aplicação foi o uso dos packages Firebase e Charts. O pacote Firebase auxiliou em todo o acesso ao banco de dados, enquanto que o Charts facilitou a criação de gráficos personalizados.

Além disso, a etapa de construção da *TransferWindow* apenas começou depois de concluir a **Especificação** contida na seção 6, já que era estritamente necessário ter tido o primeiro contato com os usuários e desenvolvido a prototipagem. Concluir o **Projeto** contido na seção 7, foi uma condição sine qua non para a finalização do projeto, visto que é nessa seção que, entre outros tópicos importantes, eu determinei a arquitetura do software, baseada no padrão de arquitetura MVC (Model View Controller). Na figura 29, será apresentada as divisões das funcionalidades dentro da aplicação. Ainda que a divisão das funcionalidades tenha sido um pouco alterada, a ideia de mantê-las divididas em camadas se manteve.

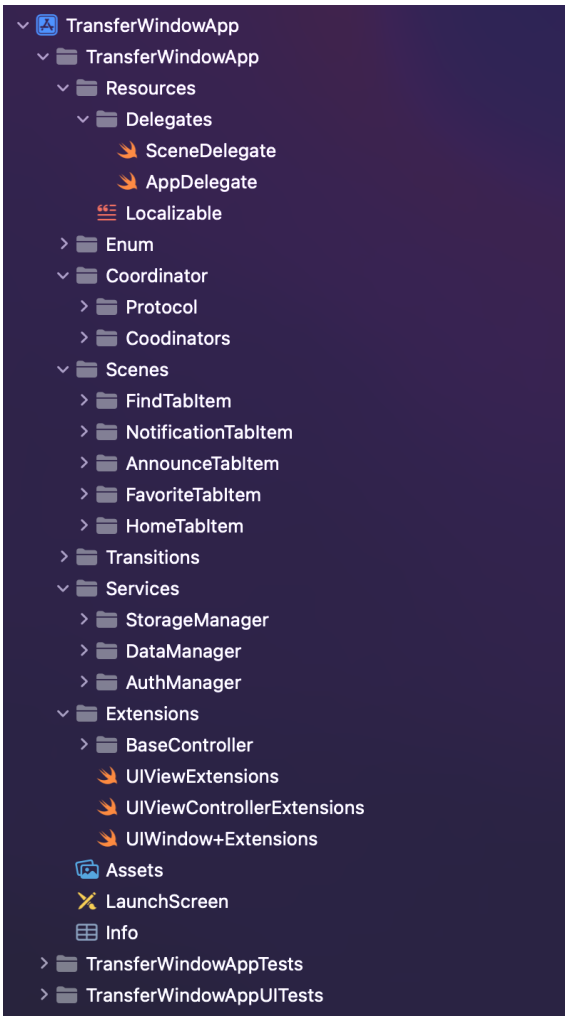

Figura 47 - Estrutura da aplicação da TransferWindow

Dentro da pasta de Recursos (*Resources*), na sub-pasta de *Delegates* estão os arquivos responsáveis por lidar com eventos de nível de aplicativo, como sua inicialização e por ciclos de criação, restauração e destruição das cenas que são apresentadas na aplicação. Nela também está contido o arquivo *Localizable*, responsável pela tradução de textos usados no app, para a linguagem usada no celular do usuário. Uma pasta *Enum,* onde estão todos os arquivos de enumerações. A pasta *Coordinator*, onde estão todos os componentes necessários para o funcionamento da navegação dentro da *TransferWindow*, através do design pattern apresentado previamente.

A pasta *Scenes* contém todo o fluxo de telas da aplicação, com seus respectivos modelos e lógicas. Cada *TabItem* possui seu próprio *coordenador*, que é responsável pelo fluxo de telas dentro dessa seção do aplicativo e todas as demais telas. Como pode ser visto na figura 30, o *FindTabItem* por exemplo, possui as telas de *Find* (Achar jogador), *Detail* (Detalhes do Jogador) e *MakeProposal* (Fazer Proposta). Para cada uma delas, existe a divisão *Model View Controller*, separando a lógica, o modelo e a interface.

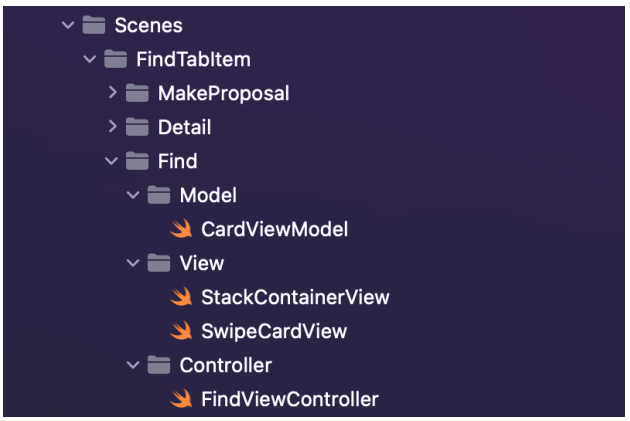

Figura 48 - Organização dos arquivos na FindTabItem

Na Figura 29 diversas pastas são apresentadas, A pasta *Transitions* possui os arquivos utilizados para fazer algumas animações entre telas. A pasta *Extensions* é onde são criadas funções que concordam com determinados tipos dentro da aplicação, como a *UIView*, e podem ser utilizadas por qualquer elemento que seja subclasse. A pasta Assets, onde são armazenados imagens, fontes e cores usadas na aplicação. As pastas de testes, que serão apresentadas e aprofundadas na próxima seção. Finalmente, a pasta *Services,* responsável por fazer toda a comunicação com o banco de dados Firebase.

Na camada de Serviços, apesar de o *AuthManager* (Gerenciador de Acesso), o *DataManager* (Gerenciador de dados menores que 1 MB) e o *StorageManager* (Gerenciador de arquivos pesados) possuírem atribuições diferentes, todos eles fazem verificações de permissão antes de executar uma tarefa, através das atribuições criadas pelos administradores dos clubes. Porém, além da verificação feita dentro da aplicação, foi preciso criar regras de acesso ao banco de dados, para garantir que nenhum usuário mal intencionado ou por acidente crie um *fake request* e acesse informações no banco.

```
service cloud.firestore {
  match /databases/{database}/documents {
    match /announces/{announceID} {
      match /users/{userID}{

        function resourceValueIsValid() {
          return request.resource.data.value is number &&
          request.resource.data.value >= 0
        }

        function resourceTypeIsValid() {
          return request.resource.data.type is string
        }

        function resourceLoanTimeIsValid() {
          return request.resource.data.loanTime is number
        }

        function resourceOpenToNegotiation() {
          return request.resource.data.openToNegotiation is string
        }

        function userIsAuthorOfAnnounce() {
          return request.resource.data.userID == request.auth.uid;
        }

        function resourceStateFieldIsValid() {
          return request.resource.data.state in ["draft", "published"]
        }

        function resourceIsValidProposal() {
          return resourceValueIsValid() &&
          resourceTypeIsValid() &&
          resourceLoanTimeIsValid() &&
          resourceOpenToNegotiation() &&
          userIsAuthorOfAnnounce() &&
          resourceStateFieldIsValid()
        }

        allow create: if resourceIsValidProposal();

        allow update: if resourceIsValidProposal() &&
        resource.data.userID == request.auth.uid &&
        request.resource.data.value >= resource.data.value;

        allow read: if resource.data.state == "published" ||
        resource.data.userID == request.auth.uid;
      }
    }
```

Figura 49 - Trecho das regras de Fazer Proposta no Firebase

Por mais que o Firebase seja um banco de dados *schemaless,* fazer a verificação de que os dados que estão chegando são os esperados, é importante para manter a integridade e o funcionamento da aplicação. No trecho acima, para poder criar uma proposta, é preciso que o usuário esteja autenticado, que o valor da proposta seja maior que zero, que o tipo do valor seja um número, que o tempo de empréstimo seja um número, que a variável aberta a negociação seja uma string e que o estado seja um dos presentes.

Para fazer uma atualização no documento, além de todas as validações anteriores, é preciso que o valor seja maior que o anterior. Enquanto que, para poder ver uma proposta, ela precisa ou estar com estado de publicado, ou o usuário que esteja tentando acessar essa informação seja o próprio criador. Essas verificações acontecem também na aplicação e são

devidamente tratadas, mas é importante existir uma segunda camada de segurança, para proteger os dados da aplicação de usuários mal intencionados.

9.Testes

Avaliar a qualidade de um software é essencial para assegurar a qualidade do produto e o seu funcionamento correto. A partir dos testes de software, é possível identificar erros que aconteçam durante as etapas de desenvolvimento e garantir que os defeitos encontrados sejam corrigidos o mais brevemente possível. Neste projeto, os testes foram realizados tanto no lado da aplicação quanto no laboratório de testes disponibilizados pelo Firebase (lado do banco de dados), que como comentado na seção 8 de Construção, servem como uma segunda camada de proteção para tentativas de leitura e escrita de dados advindos de fora da plataforma.

Na intenção de facilitar a visualização de como a segurança no lado do servidor/banco de dados atua quando uma requisição é feita, foi retirado um trecho do código que cuida da capacidade de um usuário de criar uma proposta para um jogador. O primeiro exemplo simula uma execução que cumprirá as regras e portanto a gravação do dado será permitida. Para realizar essa simulação, os seguintes dados precisam ser inseridos:

- Tipo de requisição (*create*);
- Local onde será armazenada a informação (coleção de anúncios, dentro do documento do jogador);
- Um documento (contendo todos os campos que estão presentes nas regras, e seguindo os requisitos estipulados de tipagem e valor);
- Usuário autenticado (com um *id* do recurso compatível com o *id* da requisição);

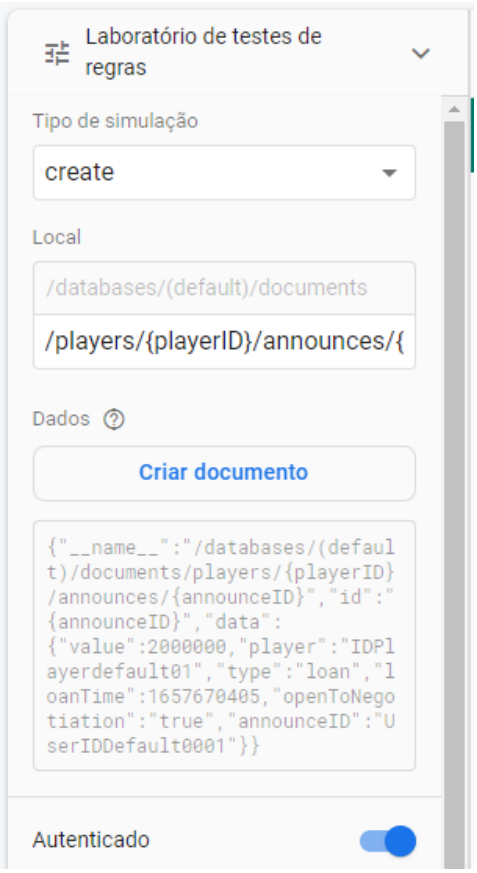

Figura 50 - Laboratório de teste de regras Firebase

Assim que o teste é executado, a regra de criação entra em ação. Neste exemplo o usuário está querendo criar uma proposta de 2 milhões, para o jogador de id = “IDPlayerdefault01”. O tipo de proposta é empréstimo e o tempo de duração do empréstimo é um valor simbólico, que está em segundos. Está aberto a negociação e o id do anunciante é o seu próprio id.

Como é possível verificar na figura 33, a gravação foi permitida pois todas as variáveis explicitadas na regra estavam presentes no corpo do documento. O valor da proposta era maior que 0 (zero). A identificação do jogador que pertence ao usuário era compatível. As tipagens das variáveis estavam corretas. O *id* do anunciante (*announceID*) era compatível com o *id* da requisição (*uid*), impedindo que outros usuários fizessem proposta no nome de algum outro usuário.

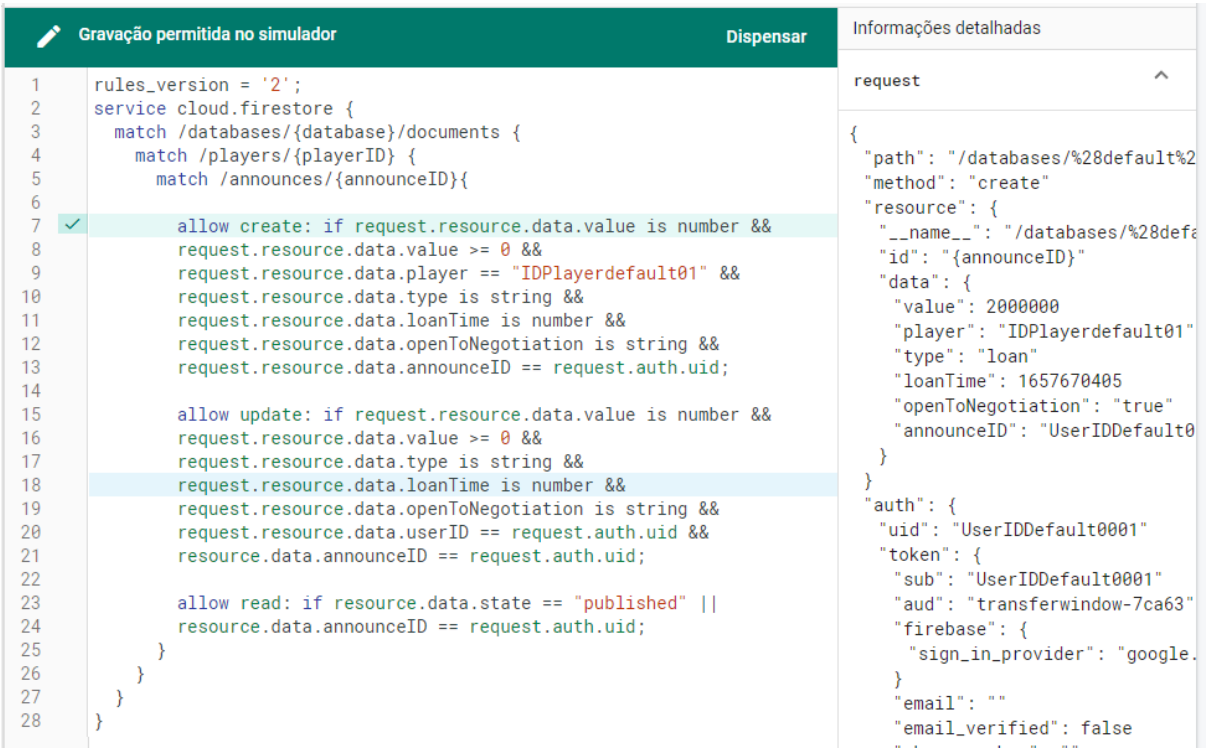

Figura 51 - Trecho das regras de Fazer Proposta executada com sucesso no laboratório de testes

Porém, como se pode observar na figura 34 a seguir, caso um usuário de id igual a “UserIDDefault0011” tentasse submeter uma proposta em nome de outro usuário de id igual a “UserIDDefault0001”, um erro ocorreria na hora de fazer a gravação no banco de dados.

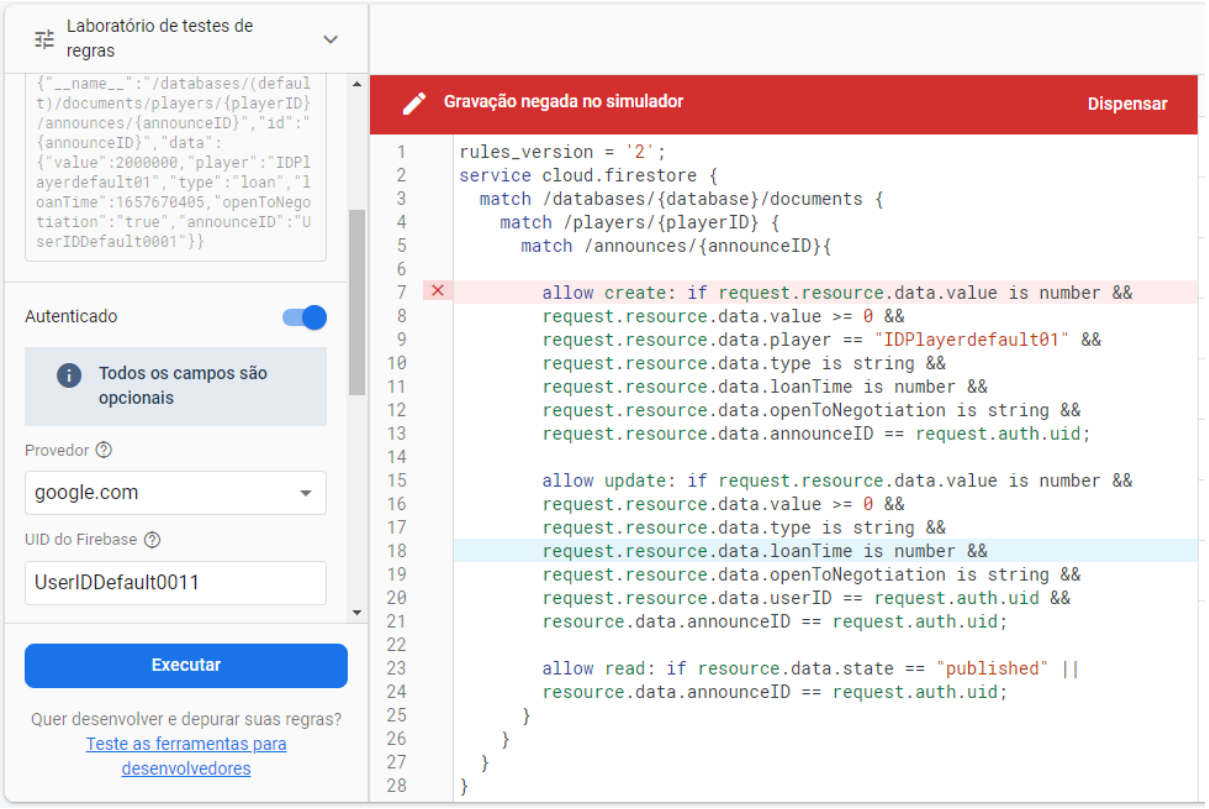

Figura 52 - Trecho das regras de Fazer Proposta executada falhou no laboratório de testes

Em relação ao lado da aplicação, também foram realizados testes para fazer a verificação do comportamento dos dados dentro do aplicativo. Como podemos observar na figura 34 a seguir, o mesmo teste que foi executado anteriormente (lado do banco de dados), foi realizado no arquivo de teste da minha aplicação do Swift. Na Figura 34 foi simulado um usuário acessando a página de “Fazer Proposta”, com um determinado jogador pré-selecionado. Através do teclado, é inserido o valor do jogador em “milhões de reais” , tipo de proposta “empréstimo”, tempo total de empréstimo e se está aberto à negociação.

É simulado então o recebimento dos dados enviados pelo teclado na *View*. A função *setRightTypes()* pega internamente os valores recebidos pelo campos preenchidos e faz a verificação dos tipos, para então realizar a comparação com as regras de negócio, checar campos vazios e **campos já tratados**. É verificado se o valor do jogador é maior que zero e se o tempo de empréstimo está acima de 12 meses. A etapa da simulação é concluída com o envio dos dados para a função *sendProposal()* que vai verificar a autenticidade e permissão do usuário, e prosseguir na gravação e o registro da proposta, se for aprovado.

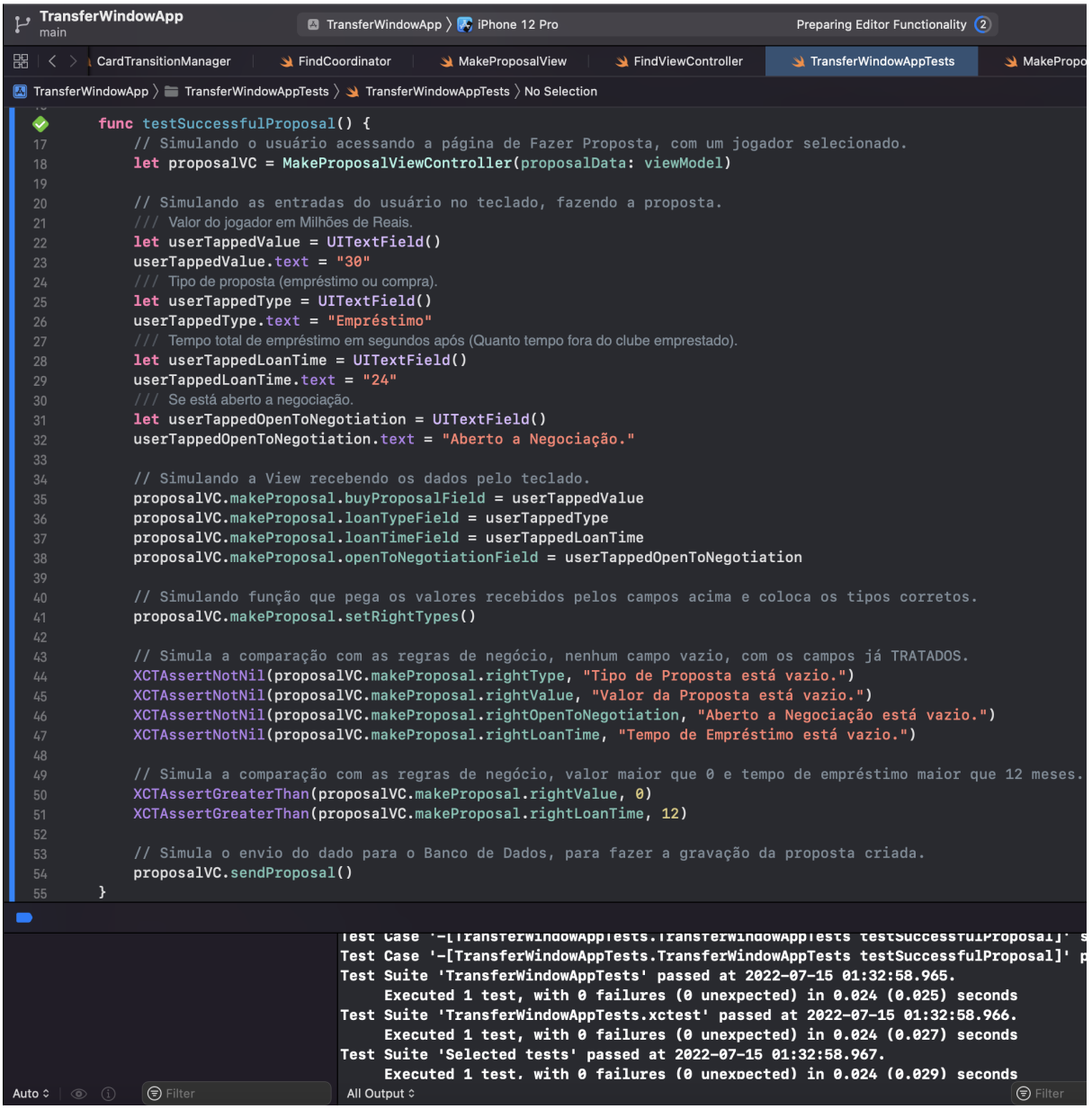

Figura 53 - Trecho das regras de Fazer Proposta executada no Swift

10.Implantação planejada

Antes de estabelecermos um modelo de implantação para o projeto do *TransferWindow*, é importante ressaltar as diversas etapas desde a idealização e construção da ideia, até a apresentação dos casos de uso, fundamentais para a prova de conceito final da oferta proposta .

Na definição da ideia, foi necessário, portanto, buscar os pontos básicos da aplicação, apontando os elementos fundamentais que irão compor a aplicação, brainstorm e entendimento dos aspectos que serão necessários para integração dos diversos setores, tais como TI, marketing, vendas etc. Robustecendo cada conceito e consolidando a proposição final.

Seguindo as melhores práticas, foi necessário conhecer o perfil dos potenciais usuários. Nesse aspecto, o projeto inspirado no TCC do Fred Gall, bacharel em Adm da FGV, trouxe elementos que permitiram a convergência das necessidades do mercado, perspectivas fundamentais sobre segurança da aplicação, mormente no tocante ao sigilo das informações estratégicas de cada usuário. Nessa toada, a escolha da linguagem de programação, também deveria estar em sintonia com os aspectos da qualidade da interface e experiência do usuário.

A busca de um layout convidativo e linguagem visual mais leve, também foi considerada e explorada. Encontrar um equilíbrio na navegação, alinhada com tendências do mercado para os aplicativos mais usados, mantendo contudo, identidade própria, resultou em uma apresentação simples e objetiva.

O Modelo de Negócio passa a ser então o ponto crucial para o sucesso da oferta do *TransferWindow.* Assim sendo, para prosseguir com o desenvolvimento do projeto e buscando amadurecer sua existência até a sua entrada propriamente dita no mercado, algumas metas foram traçadas:

1 - **METODOLOGIA** - Seguindo com as metodologias especificadas anteriormente, a ideia é apresentar a V1 do produto para os primeiros possíveis interessados na plataforma num período de 3 meses, a partir do lançamento oficial do projeto. Nesse tempo é preciso desenvolver uma breve identificação visual para a plataforma, criar uma base de dados consistente e adequada à proposta da aplicação, finalizar a implementação das funcionalidades apontadas como essenciais para o projeto e realizar rodadas de testes com usuários para validação do aplicativo.

2 - **START DO PROJETO** - A proposta é que, para o primeiro mês de desenvolvimento, as funcionalidades já estejam implementadas e já tenha sido contratado um profissional em design para desenvolver a parte de UX e UI. Devido ao fato de o Fred Gall, já citado anteriormente, fazer parte do mercado esportivo de futebol, também possuímos uma rede de contatos onde será possível fazer apresentações para grandes jogadores dentro desse nicho. Essa é a hora de obter todo o feedback possível para validar o rumo que foi traçado e fazer eventuais correções de rota.

3 - **POPULAR A BASE** - No primeiro momento, temos como objetivo popular a base de dados através do uso da plataforma *Transfermarkt*, que já possui uma API que atende aos nossos requisitos. Dependendo da quantidade de requisições, o uso Transfermarkt é gratuito. Caso seja

percebido em conversas com usuários, que as informações apresentadas não são suficientes, existem outros provedores de dados de jogadores pagos que podem agregar o sistema. Mas a ideia é que, como a plataforma vai ser especializada em negociações, ao longo do tempo, seremos responsáveis por ter todo o controle de negociações do mercado brasileiro, e portanto, sermos os próprios responsáveis pelos dados, sem a necessidade de uso de outras plataformas.

4 - **TESTES** - Os testes com usuários precisam acontecer logo após o protótipo ser finalizado, já que é através dos testes de usabilidade que será possível verificar se o usuário compreendeu claramente o objetivo do software e descobrir problemas e pontos de melhoria. Num primeiro momento, talvez tenhamos dificuldade de encontrar usuários de todos os tipos que pretendemos atender. Principalmente os jogadores, mas nesse caso é possível, para teste, colocar jogadores amadores que gostariam de participar do app, para no futuro usá-lo como vitrine ou quem sabe até, um dia ingressar no futebol profissional. A partir desses testes é possível ainda atualizar as personas criadas com as informações coletadas, refletindo assim num plano de marketing mais abrangente e interfaces de usuário mais intuitivas.

Após as etapas de validação do produto, testes com usuários e criação de identidade, é preciso buscar investidores que acreditem no potencial do negócio. Como citado anteriormente, por possuirmos alguns contatos dentro do esporte, é possível que consigamos vender a ideia para os grandes clubes cariocas devido à proximidade geográfica junto aos grandes players desse mercado como Flamengo, Vasco, Botafogo, ou Fluminense. Também estou trabalhando num Business Plan para formatar a oferta para Venture Capital[10].

Com uma entrada de capital, a ideia é contratar uma equipe de desenvolvimento e dar seguimento ao projeto. Prezando sempre pela identidade, funcionalidade e experiência do usuário, vamos formar 2 times de desenvolvimento em vez de aderir a uma linguagem única. Além do time de programadores, é preciso também, contratar profissionais de marketing para cuidar das vendas e design para manter sempre a UI e UX no padrão diferenciado em comparação com os semelhantes.

Seguindo uma estratégia um pouco diferente das empresas atuais, a ideia é manter a contratação de uma equipe que funcione trabalhando em home office, para evitar custos com infraestrutura e adequando a empresa com os tempos atuais de trabalho.

[10] Modalidade de investimentos alternativos usada para apoiar negócios a partir da compra de uma participação acionária.

11.Considerações Finais

Primeiramente gostaria de reforçar a importância como aluno, em minha formação, pela oportunidade de conviver em um ambiente sólido de ensino e poder estar cercado de magistrados de alto nível de formação técnica e didática. Creio que um fator de extrema relevância, que tornou possível executar e concluir este projeto, considerando as diversas áreas de conhecimentos necessárias, foi devido às disciplinas que tive a honra de cursar na PUC, como formando em Engenharia da Computação. Ao longo do processo da minha formação superior, cursei matérias de extrema importância que aqui destaco: **Engenharia de Requisitos [INF1377]**, que apresentou a elicitação e análise de requisitos, além de técnicas de modelagem de software; **Programação Orientada a Objetos [INF1636]**, que apresentou o paradigma de orientação a objetos e a vantagem de usar design patterns; **Bancos de Dados(I, II, III e IV)**, que contemplavam todos os assuntos possíveis, principalmente o **IV [INF1374]**, que estudamos sobre bancos NoSQL; **Interação Humano-Computador [INF1403]**, que introduziu assuntos como pesquisa com usuário, personas e outros conceitos amplamente utilizados neste projeto; além de muitas outras que em conjunto formaram a bagagem necessária para o projeto.

Em relação ao projeto aqui apresentado, este foi repleto de desafios técnicos que me permitiram adquirir uma ampla capacidade de trabalhar com front-end, descobrindo e entendendo temas que vão desde a utilização dos design patterns até a criação de telas e animações. Além de ter sido uma ótima experiência relacionada ao desenvolvimento de interfaces e preocupação com a experiência do usuário dentro da plataforma, esse projeto também me permitiu aprofundar mais na linguagem escolhida. Como o Swift é uma linguagem de programação, que é uma mistura de orientado a objetos e orientado a protocolos, também pude obter outras experiências ao explorar com mais qualidade o uso de protocolos responsáveis por realizar as funções de comunicação e transferências de dados dentro da aplicação. Essa experiência foi sem dúvida, fundamental para a minha inserção futura no mercado de trabalho.

Um importante momento no desenvolvimento do projeto, foi a oportunidade de trabalhar toda a parte de comunicação da aplicação com o banco de dados **Firebase**, a partir da construção de camadas de serviço. Pude aprimorar meus conhecimentos em banco de dados orientados a documentos e utilizar de forma mais avançada a criação de regras de acesso a dados. O trabalho contínuo com essa matéria, ajudou sobremaneira o meu entendimento sobre as necessidades de soluções que apoiem projetos dessa envergadura. Ressalto que ter cumprido os estágios em laboratórios da PUC-Rio foram vitais para a finalização do projeto e me encorajaram a desvendar esse universo mais profundamente e de forma crítica.

O Laboratório de Bioinformática e Bancos de Dados foi de grande valia e mostrou-se também importante para o meu desenvolvimento com trabalho em equipe. Introdução a frameworks e uso de design patterns e, principalmente o aprimoramento na utilização de bancos de dados, onde usei pela primeira vez o Firebase, foi essencial para esta aplicação.

Na Apple Developer Academy PUC-Rio eu aprendi que o processo de entendimento de um problema, passa pela análise conceitual daquilo que se pretende obter como produto final. O

estágio me ensinou que é importante construir a aplicação conjuntamente com os usuários, levando sempre em consideração seus feedbacks e sugestões para o projeto. Além disso, pude ter experiências que me aproximaram cada vez mais do mercado de trabalho. Estar envolvido com pessoas de diferentes cursos, resolvendo tarefas multidisciplinares e criando apresentações para vender a ideia das minhas soluções serão, seguramente, um legado para a minha vida.

12.Bibliografia

[1] FIFA. FIFA Big Count 2006: 270 million people active in football. Disponível em: https://digitalhub.fifa.com/m/55621f9fdc8ea7b4/original/mzid0qmguixkcmruvema-pdf.pdf Acesso em: 27/02/2022.

[2] DELLOIT. Welcome to the 22nd edition of the Deloitte Football Money League (‘DFML’) in which we profile the highest revenue generating clubs in world football. Disponível em: https://www2.deloitte.com/global/en/pages/consumer-business/articles/deloitte-football-money-league.html. Acesso em: 15/02/2022.

[3] CBF. Documento produzido pela consultoria EY detalha todos os pilares que envolvem a indústria do futebol, que representa 0,72% do PIB nacional. Disponível em: https://www.cbf.com.br/a-cbf/informes/index/cbf-apresenta-relatorio-sobre-papel-do-futebol-na-economia-do-brasil. Acesso em: 17/02/2022.

[4] CBF. Relatório de Intermediários 2021. Disponível em: https://www.cbf.com.br/a-cbf/informes/registro-transferencia/relatorio-de-intermediarios-2021 Acesso em: 21/02/2022.
EY. Levantamento Financeiro dos Clubes Brasileiros 2020. Disponível em: https://www.ey.com/pt_br/media-entertainment/levantamento-financeiro-dos-clubes-brasileiros-2020. Acesso em: 17/02/2022.

[5] TRANSFERROOM. TransferRoom. Disponível em: https://www.transferroom.com/. Acesso em: 02/03/2022.

[6] GLOBOESPORTE. Conheça o TransferRoom, o aplicativo de negociações de futebol. Disponível em: https://ge.globo.com/video/conheca-o-transferroom-o-aplicativo-de-negociacoes-de-futebol-8722090.ghtml. Acesso em: 07/03/2022.

[7] WYSCOUT. Wyscout. The professional platform for people working in the football world: videos, data, statistics and tools. Disponível em: https://wyscout.com. Acesso em: 07/03/2022.

[8] O CAÇA ATLETAS. O Caça Atletas. Disponível em: https://www.ocacaatletas.com.br/ . Acesso em: 02/03/2022.

[9] GLOBOESPORTE. Deu match: aplicativo reúne clubes de 78 ligas com novo jeito de fazer negociações no futebol. Disponível em: https://ge.globo.com/futebol/futebol-internacional/noticia/deu-match-aplicativo-reune-clubes-de-78-ligas-com-novo-jeito-de-fazer-negociacoes-no-futebol.ghtml. Acesso em: 12/03/2022.

[10] SWIFT. Uma linguagem aberta e poderosa para todo mundo criar apps incríveis. Disponível em: https://www.apple.com/br/swift/. Acesso em: 08/07/2022.

[11] FIREBASE. Make your app the best it can be . Disponível em: https://firebase.google.com/ Acesso em: 08/07/2022.

[12] SCRUM. O que é, como funciona e por que é incrível - Atlassian. Disponível em : https://www.atlassian.com/br/agile/scrum. Acesso em: 09/07/2022.

[13] KANBAN. Kanban: conceito, como funciona, vantagens e implementação. Disponível em: https://www.totvs.com/blog/negocios/kanban/. Acesso em: 09/07/2022.

[14] MIRO. Onde equipes produzem resultados. Disponível em: https://miro.com/pt/. Acesso em: 08/07/2022.

[15] Vinicius Cardoso Garcia, D.C.Sc. O que é arquitetura Serverless (sem servidor)? Disponível em: https://viniciusgarcia.me/development/o-que-eh-arquitetura-serverless/. Acesso em 05/07/2022.
TECNOSPEED. O que é Arquitetura Serverless? Disponível em: https://blog.tecnospeed.com.br/o-que-e-arquitetura-serverless/#:~:text=Arquitetura%20Serverless%2C%20ou%20%E2%80%9Ccomputa%C3%A7%C3%A3o%20sem,que%20esses%20aplicativos%20est%C3%A3o%20rodando. Acesso em: 05/07/2022.

[16] OPUS-SOFTWARE. Design Patterns – O que são e quais os benefícios? Disponível em: https://www.opus-software.com.br/design-patterns/. Acesso em: 01/07/2022.