UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

ESPECIFICAÇÃO E DESENVOLVIMENTO DE UM APLICATIVO IOS PARA MONITORAMENTO DE PACIENTES DE NUTRIÇÃO

TRABALHO DE GRADUAÇÃO

Evandro Bianquin Machado

Santa Maria, RS, Brasil
2017

ESPECIFICAÇÃO E DESENVOLVIMENTO DE UM APLICATIVO IOS PARA MONITORAMENTO DE PACIENTES DE NUTRIÇÃO

Por

Evandro Bianquin Machado

Trabalho de Graduação apresentado ao Curso de Ciência da Computação da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do grau de
Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Giovani Rubert Librelotto

Trabalho de Graduação Nº 433
Santa Maria, RS, Brasil
2017

Universidade Federal de Santa Maria
Centro de Tecnologia
Curso de Ciência da Computação

A Comissão Examinadora, abaixo assinada,
aprova o Trabalho de Graduação

ESPECIFICAÇÃO E DESENVOLVIMENTO DE UM APLICATIVO IOS PARA MONITORAMENTO DE PACIENTES DE NUTRIÇÃO

elaborado por
Evandro Bianquin Machado

como requisito parcial para obtenção do grau de
Bacharel em Ciência da Computação

COMISSÃO EXAMINADORA:

_____
Giovani Rubert Librelotto, Dr.
(Presidente/Orientador)

_____
Lisandra Manzoni Fontoura, Drª. (UFSM)

_____
Giliane Bernardi, Drª. (UFSM)

Santa Maria, 05 de Julho de 2017.

AGRADECIMENTOS

Agradeço em primeiro lugar ao meus pais, que foram desde o início a razão de tudo isto e o combustível necessário para chegar até aqui. Em reconhecimento ao esforço realizado durante todos estes anos, todo sucesso que obtiver sempre será graças a vocês.

Gostaria de agradecer também aos amigos que sempre se fizeram presentes na minha jornada, alguns desde muito cedo, sempre me apoiando a completar esta graduação. Em especial ao Tiago Sanchotene, que nesta etapa final, ajudou muito me oferecendo toda estrutura que eu precisei.

E por fim gostaria de agradecer ao meu orientador, Prof. Giovani, por ter me apoiado em todas as tentativas que passei até concluir este trabalho.

RESUMO

Trabalho de Graduação
Curso de Ciência da Computação
Universidade Federal de Santa Maria

ESPECIFICAÇÃO E DESENVOLVIMENTO DE UM APLICATIVO IOS PARA MONITORAMENTO DE PACIENTES DE NUTRIÇÃO

AUTOR: EVANDRO BIANQUIN MACHADO
ORIENTADOR: PROF DR GIOVANI RUBERT LIBRELOTTO
Data e Local da Defesa: Santa Maria, 05 de julho de 2017

O aumento dos casos de doenças nutricionais, que levou a obesidade a ser considerada um problema de saúde pública mundial, vem ganhando destaque na atualidade. Neste contexto se inserem diversas ações que vão desde estratégias de saúde pública até formas de controle nutricional de pacientes, que é o objetivo deste trabalho. Criar ferramentas capazes de realizarem o monitoramento de pacientes pode ajudar profissionais nutricionistas a obterem informações mais precisas e a capacidade de tomar ações corretivas com mais agilidade. O presente trabalho busca desenvolver um aplicativo para iOS que torne os dados de consumo dos usuários disponível ao seu nutricionista tornando o processo mais rápido e preciso.

Palavras-chave: iOS. Nutrição. Monitoramento de Pacientes.

ABSTRACT

Undergraduate Final Work

Undergraduate Program in Computer Science

Federal University of Santa Maria

SPECIFICATION AND DEVELOPMENT OF AN IOS APPLICATION FOR MONITORING NUTRITION PATIENTS

AUTHOR: EVANDRO BIANQUIN MACHADO

ADVISOR: PROF DR GIOVANI RUBERT LIBRELOTTO

The increase in cases of nutritional diseases, which led obesity to be considered a global public health problem, has been gaining prominence nowadays. In this context, several actions are included, ranging from public health strategies to forms of nutritional control of patients, which is the objective of this study. Creating tools capable of patient monitoring can help nutritionists gain more accurate information and the ability to take corrective action more quickly. This work aims to develop an application for iOS that makes the consumption data of users available to your nutritionist making the proccess much faster and more accurate.

Keywords: iOS. Nutrition. Patient Monitoring.

LISTA DE FIGURAS

Figura 1 - Console de uma aplicação no *Firebase* 18
Figura 2 - Telas do *App Saúde* instalado nos dispositivos da *Apple* 20
Figura 3 - Telas do aplicativo *MyFitnessPal* utilizado como referência 22
Figura 4 - Casos de Uso 27
Figura 5 - Diagrama de classes 28
Figura 6 - Modelo relacional do banco de dados 31
Figura 7 - Representação das comunicações entre entidades 33
Figura 8 - Configuração de autenticação no *Firebase* 34
Figura 9 - Configuração do banco de dados do *Firebase* 35
Figura 10 - Tela da IDE de desenvolvimento para criação de interfaces 37
Figura 11 - Telas do aplicativo desenvolvido 40
Figura 12 - Exemplo de fluxo quando o usuário não realiza a meta proposta 41
Figura 13 - Exemplo de como os dados são armazenados no *Firebase* 44
Figura 14 - Fluxo das telas do histórico 45

LISTA DE ABREVIATURAS

iOS - Sistema Operacional da Apple para Dispositivos Móveis

HTML - HyperText Markup Language

CSS - Cascading Style Sheets

MVC - Model View Controller

IDE - Integrated Development Environment

API - Application Programming Interface

JSON - JavaScript Object Notation

UML - Unified Model Language

UI - User Interface

SUMÁRIO

1 INTRODUÇÃO 10
1.1 Contextualização 10
1.2 Problema abordado 12
1.3 Objetivos 13
1.4 Organização do Texto 13
2 TECNOLOGIAS UTILIZADAS 15
2.1 iOS e iPhone 15
2.2 Swift 16
2.3 Firebase 17
2.4 HealthKit e ResearchKit 19
2.5 Sistemas de Monitoramento de Nutrição 21
2.6 Aplicativo Referência 22
3 MODELAGEM DO PROJETO 24
3.1 Requisitos 24
3.2 Casos de Uso 25
3.3 Diagramas 27
3.3.1 Diagrama de Classes 27
3.3.2 Modelo Relacional 30
3.4 Fluxo de Dados 32
4 DESENVOLVIMENTO 34
4.1 Configuração da Infraestrutura 34
4.2 Criação da Interface 36
4.3 Programação dos *Controllers* 37
5 RESULTADOS OBTIDOS 40
5.1 Aplicativo Simulado 40
5.2 Monitoramento em Segundo Plano 42
5.3 Histórico de Refeições 44
6 CONCLUSÃO 46
6.1 Trabalhos futuros 47
REFERÊNCIAS 49

1 INTRODUÇÃO

Este capítulo contextualiza o tema proposto neste trabalho além de abordar a motivação para o desenvolvimento, a descrição do problema, os objetivos que se deseja alcançar com o desenvolvimento deste trabalho e sua estrutura.

1.1 Contextualização

Nutrição é uma ciência que está presente todos os dias na vida de qualquer pessoa, sendo um fator muito importante no estilo de vida da população e principalmente no desenvolvimento e manutenção de um estado positivo de saúde. (WARDLAW, 2013)

Segundo Wardlaw (2013, p. 24), uma pessoa realiza em toda sua vida uma média de 70 mil refeições e são consumidas aproximadamente 60 toneladas de alimentos. O que faz da nutrição um importante fator de saúde, ele explica “Nutrição é a ciência que relaciona os alimentos à saúde e às doenças. Além disso, estuda os processos de ingestão, digestão, absorção, transporte e excreção de substâncias alimentares pelo organismo humano.”.

Os casos de problemas e doenças nutricionais vem crescendo muito atualmente, usando a obesidade como exemplo, ela já se tornou um problema de saúde pública mundial tanto em países desenvolvidos como em países em desenvolvimento, observa-se que o aumento dos casos é relevante em todo o globo. (PINHEIRO, FREITAS e CORSO, 2004, p. 3)

Parte da responsabilidade destes problemas é da transição nutricional pela qual a população mundial passou, com mudanças no padrão de consumo que acompanham as mudanças econômicas, sociais e demográficas, e estas mudanças são muito atribuídas ao processo de urbanização que a população vem passando através dos anos.

> *“No Brasil, a obesidade como problema de Saúde Pública é um evento recente. Apesar da existência de relatos a partir da Era Paleolítica sobre “homens corpulentos”, a prevalência de obesidade*

> *nunca se apresentou em grau epidêmico como na atualidade. Enquanto agravo nutricional, a desnutrição era assumida como um problema relevante para os países em desenvolvimento, e a obesidade seria para países desenvolvidos. Atualmente, tanto os países desenvolvidos como os países em desenvolvimento não se apresentam como unidades homogêneas, quer para a prevalência da desnutrição, quer para a da obesidade. Ao contrário, podem ser caracterizados em uma fórmula mista tanto de excesso de peso quanto de déficit nutricional."* (PINHEIRO, FREITAS e CORSO, 2004).

Existem diversos outros problemas nutricionais além da obesidade que fazem as pessoas buscarem auxílio profissional, ou mesmo para acompanhamento de dietas com objetivos apenas estéticos.

Um dos desafios de toda dieta é seguir à risca as orientações do nutricionista, e a forma como este profissional realizará o acompanhamento do seu paciente. Tratamentos nutricionais podem ser muito complexos e envolver grandes mudanças no estilo de vida dos pacientes. Seguir as prescrições é parte fundamental de todo tratamento. (BALDO et al. 2015)

Também pode-se identificar que a sociedade vem numa crescente busca de informações sobre nutrição, uma busca cada vez mais crescente por dietas e materiais da área da nutrição, e cada vez mais aplicativos especializados em informações e controle nutricional vem sendo criados e utilizados, alcançando marcas expressivas de *downloads* nas lojas de aplicativos de diversos sistemas operacionais para dispositivos móveis.

Como citado anteriormente, este trabalho se encaixa num contexto maior, sendo uma ferramenta auxiliar de outro projeto que desenvolve um sistema de *homecare*. Este será responsável por uma série de ferramentas para profissionais de nutrição, que disponibilizará ferramentas de gestão ao profissional, métricas de seu paciente, dentre outras funcionalidades. A ferramenta desenvolvida aqui será mais uma fonte de dados do sistema.

Este trabalho foi realizado para ser mais uma ferramenta neste contexto, disponibilizando ainda mais dados sobre pacientes e deixá-los à disposição para o sistema dos nutricionistas utilizar da forma que for necessário. Por ser um sistema que irá trabalhar com os dados de pacientes para obter informações relevantes aos

nutricionistas, quanto mais fontes de dados ele tiver, mais precisas serão as informações mineradas por ele.

Sistemas de *homecare* vem crescendo com o passar do tempo e uma das principais razões são os problemas relacionados aos serviços de saúde, para sanar essas dificuldades, sistemas de monitoramento em ambientes pervasivos podem se tornar uma ferramenta aliada muito poderosa. Monitorar pacientes em casa, além de proporcionar muito mais conforto a eles, evita problemas como superlotação de hospitais ou ambientes institucionais.

"Com o auxílio da computação pervasiva, não só as condições do paciente, mas também do ambiente podem ser monitoradas e avaliadas. Assim, dependendo do contexto, o ambiente pervasivo pode ser adaptar as necessidades do paciente de forma proativa. Porém, a forma como a descrição do contexto atual de cada uma dessas entidades deve ser expressa se torna complexa de representar computacionalmente, de forma dinâmica. Ontologias podem ser utilizadas para representação do conhecimento. " (BASTIANI, SOARES, LIBRELOTTO, 2012)

1.2 Problema abordado

Neste contexto, buscamos identificar as principais dificuldades dos pacientes e como poderíamos facilitar o trabalho dos nutricionistas, visto que pela complexidade de se lidar com a vida humana, qualquer projeto de tecnologia para minimizar erros fatais deve evitar tomar decisões diretas. Com base nisto, a abordagem escolhida foi disponibilizar ferramentas que aumentem a agilidade e a eficácia de diagnósticos e ações de profissionais habilitados.

Com o mesmo foco, queremos abordar dois problemas diferentes. O primeiro problema que queremos resolver é a dificuldade dos pacientes de nutrição em ter acesso de forma fácil e em qualquer lugar das orientações que são passadas pelo nutricionista, e a falta de um registro unificado destas informações completas e em tempo real.

Por outro lado, os profissionais de nutrição precisam de um ambiente unificado no qual eles sejam capazes de gerir melhor seus pacientes, reunindo a maior quantidade de dados relevantes possíveis, preferencialmente que estas informações

cheguem conforme são geradas, para que seus diagnósticos e tratamentos sugeridos sejam baseados em dados cada vez mais precisos e que ações possam ser tomadas com muito mais agilidade.

1.3 Objetivos

O objetivo deste trabalho é especificar e desenvolver um aplicativo para iOS que seja utilizado por pacientes de nutrição para realizar o acompanhamento diário de todas as suas refeições do dia, no qual ele receberá a informação do que comer na sua próxima refeição e responderá se conseguiu ou não cumprir aquela meta.

Além disso, ele deverá ser capaz de, com a permissão explícita do usuário, buscar informações sobre a sua nutrição provenientes de outras fontes, sejam aplicativos ou dispositivos como relógios inteligentes, que tenham disponibilizados tais informações com o sistema

Para disponibilizar estas informações sobre seu estado nutricional com o profissional responsável sem nenhuma interação extra dentro do aplicativo desenvolvido neste trabalho além de ativar a função e liberar as permissões solicitadas.

Esses dados são preparados e disponibilizados num servidor online para acessos dos profissionais que acompanham aquele usuário, como será abordado a seguir neste trabalho. Este projeto faz parte de um contexto maior de um sistema de *homecare* que é o responsável por tratar e entregar estes dados aos nutricionistas, o objetivo deste trabalho é receber os dados do paciente e entrega-los na nuvem para esta ferramenta.

1.4 Organização do Texto

O texto está estruturado da seguinte maneira: no Capítulo 2 serão apresentadas as tecnologias que foram utilizadas no decorrer do desenvolvimento do projeto, bem como aplicativos de terceiros que além de referência para o

desenvolvimento também poderão ser utilizados por pacientes para realizar seu monitoramento das suas refeições e compartilhar estes dados com o aplicativo desenvolvido neste projeto.

No Capítulo 3 serão descritas as etapas da modelagem do software como a definição dos requisitos, os casos de uso do aplicativo e a modelagem utilizada como base para o desenvolvimento.

O Capítulo 4 apresenta a sequência de desenvolvimento do aplicativo e como as tecnologias escolhidas foram utilizadas para obtenção de uma aplicação que cumprisse todos os requisitos estipulados e alcançasse o objetivo deste trabalho. O Capítulo 5 descreve os resultados obtidos com o desenvolvimento, e simulações dos casos de uso planejados para validação do trabalho, verificando de que forma os objetivos definidos foram alcançados. O Capítulo 6 encerra este trabalho com a conclusão e sugestões para trabalhos futuros.

2 TECNOLOGIAS UTILIZADAS

Este capítulo apresenta todas as tecnologias utilizadas durante a execução do projeto descrito neste trabalho, tratando os conceitos mais importantes de cada uma delas, e expondo as razões pelas quais estas foram as tecnologias escolhidas para serem utilizadas.

Também são abordados aplicativos de terceiros, que como citado anteriormente, não apenas serviram como base para o desenvolvimento deste trabalho, poderão ser utilizados pelos usuários para controlar sua dieta, e utilizando de tecnologias desenvolvidas pela *Apple* de compartilhamento de dados de saúde e de pesquisa poderemos utilizar destes dados inseridos em outros aplicativos para alimentar nosso sistema.

2.1 iOS e iPhone

Umas das principais escolhas quando se decide desenvolver um projeto de desenvolvimento de aplicativos móveis é a plataforma que será utilizada para ele. Atualmente no mercado existem duas grandes plataformas que lideram em quantidade de usuários, o *iOS* e o *Android*.

Android é o sistema operacional desenvolvido pela Google para dispositivos móveis, é um projeto de código aberto no qual cada fabricante pode ter acesso, realizar suas modificações e implantar em seus dispositivos para distribuição (ROCHA, NETO, 2011). Isso gera uma gama enorme de diferentes dispositivos sendo comercializados e também diversas versões do sistema operacional visto que cada fabricante pode realizar as alterações que julgar necessárias.

Já o *iOS* é o sistema operacional desenvolvido pela *Apple* para utilização em seus próprios dispositivos, como o *iPhone* e o *iPad*. Este sistema é mantido única e exclusivamente pela proprietária e não é de código aberto, o que faz com que a *Apple* tenha controle sobre todas as etapas de criação do produto, do *software* ao *hardware*. Aqui a gama de dispositivos é bem menor e não existem variações do sistema, o que

facilita bastante o desenvolvimento. A *Apple* também desenvolveu ferramentas que foram muito úteis para a construção deste projeto, como o *HealthKit* e o *ResearchKit*, que serão explicados a seguir, os quais nos possibilitam compartilhar dados de saúde para pesquisa e aprimoramento dos aplicativos com este fim.

Existem ainda opções de desenvolvimento em ferramentas que utilizam outras linguagens de programação e geram código nativo para todos os dispositivos, como o *Cordova* ou o *Xamarim*. Nestas ferramentas podemos utilizar linguagens como as focadas em desenvolvimento web, como *HTML, CSS* e *JavaScript*, e elas se encarregam de gerar aplicativos multiplataforma, mas assim perdemos a capacidade de utilizar ferramentas as acima citadas para compartilhar dados entre aplicativos por serem ferramentas da *Apple* e que tiveram um papel importante neste projeto aqui descrito.

Com base nestas informações, a plataforma escolhida foi o iOS, com o foco nos iPhones, visto que dispositivos celulares estão presentes o tempo todo com seus usuários, o que deixa a dieta passada pelo nutricionista sempre junto do paciente e torna o retorno dos dados para o profissional quase em tempo real. E que juntamente com as tecnologias empregadas disponíveis apenas pela *Apple* tornaram o alcance dos objetivos propostos possíveis.

2.2 Swift

Swift é uma linguagem de programação desenvolvida pela *Apple* para as suas plataformas, substituindo o anterior *Objective-C* para trazer maior poder aos programadores, mesmo que ainda mantenha a compatibilidade com esta. Manteve muito dos conceitos de desenvolvimento como o modelo MVC, que deixa o código muito bem organizado e estruturado. (WELLS, 2015)

Objective-C é uma linguagem criada em 1980 e licenciada em 1988 por uma empresa do fundador da *Apple*, a *NeXT Software*. Esta empresa foi adquirida pela *Apple* em 1996 e então a linguagem foi incorporada aos seus sistemas, tanto para computadores quanto para dispositivos móveis. (KOCHAN, 2011)

Para o desenvolvimento recomenda-se a utilização da IDE criada especialmente para a plataforma, chamada *xCode* e que além de compreender

perfeitamente a linguagem *Swift*, traz uma série de ferramentas como criação de interfaces gráficas, o que facilita muito o desenvolvimento dos aplicativos.

Ainda é possível utilizar *Objective-C* para criar aplicativos para *iOS,* porém após programar em ambas as linguagens é notável a evolução apresentada pelo *Swift*. Além de ser mais otimizada e produzir códigos muito mais eficientes, é uma linguagem bem moderna e flexível, oferecendo conceitos como os *Optionals*, que tratam valores nulos para evitar falhas nos aplicativos em produção, *Type Safety*, que é uma propriedade de algumas linguagens para evitar erros de tipagem. Além disso é uma linguagem extremamente legível, sua sintaxe favorece a leitura do código, facilitando que quando outros programadores vejam o código, consigam entender mais facilmente do que se trata.

2.3 Firebase

Esta plataforma desenvolvida pela *Google* foi construída pensando em abstrair dos programadores toda a parte de infraestrutura na nuvem. Não é um serviço totalmente gratuito mas tem limites de uso sem custo que são suficientes para o início do projeto. (GOOGLE, 2017)

Muitos serviços são oferecidos dentro da plataforma do *Firebase*, como banco de dados, sistema de autenticação, sistema de funções online que permite produzir código personalizado para o *back-end* e armazenamento tanto de mídia quanto de páginas *web*. (CHENG, 2017)

Como foi descrito previamente, este aplicativo será incorporado dentro de um projeto maior de *homecare*, sendo que a sua função é coletar dados dos pacientes e alimentar este sistema. Para garantir a simulação apropriada deste trabalho, o *Firebase* foi escolhido para fazer o papel deste sistema no desenvolvimento desta aplicação, trocando a comunicação com uma *API* hospedada no servidor do projeto pela comunicação com a infraestrutura já pronta do *Firebase*, tornando o desenvolvimento deste trabalho mais independente.

Dos serviços oferecidos pela plataforma, este trabalho utilizou os serviços de banco de dados, no qual o aplicativo busca as prescrições do nutricionista responsável e salva a resposta dos usuários se realizaram ou não a meta nutricional. Além disso

o aplicativo, se tiver a permissão do usuário, busca dados compartilhados de outros aplicativos e também envia esses dados para serem persistidos neste banco de dados.

Além disso, são utilizados os serviços de autenticação oferecidos pela plataforma, assim é possível simular com exatidão o funcionamento de uma *API* real. O usuário será autenticado e receberá os seus dados do servidor. Na própria infraestrutura do serviço é que fica armazenada a sessão do usuário, tornando o processo de identificação do usuário muito mais fácil e rápida ao se programar. O *Firebase* foi escolhido justamente por oferecer todas as ferramentas necessárias para realizar o papel deste servidor em uma única ferramenta.

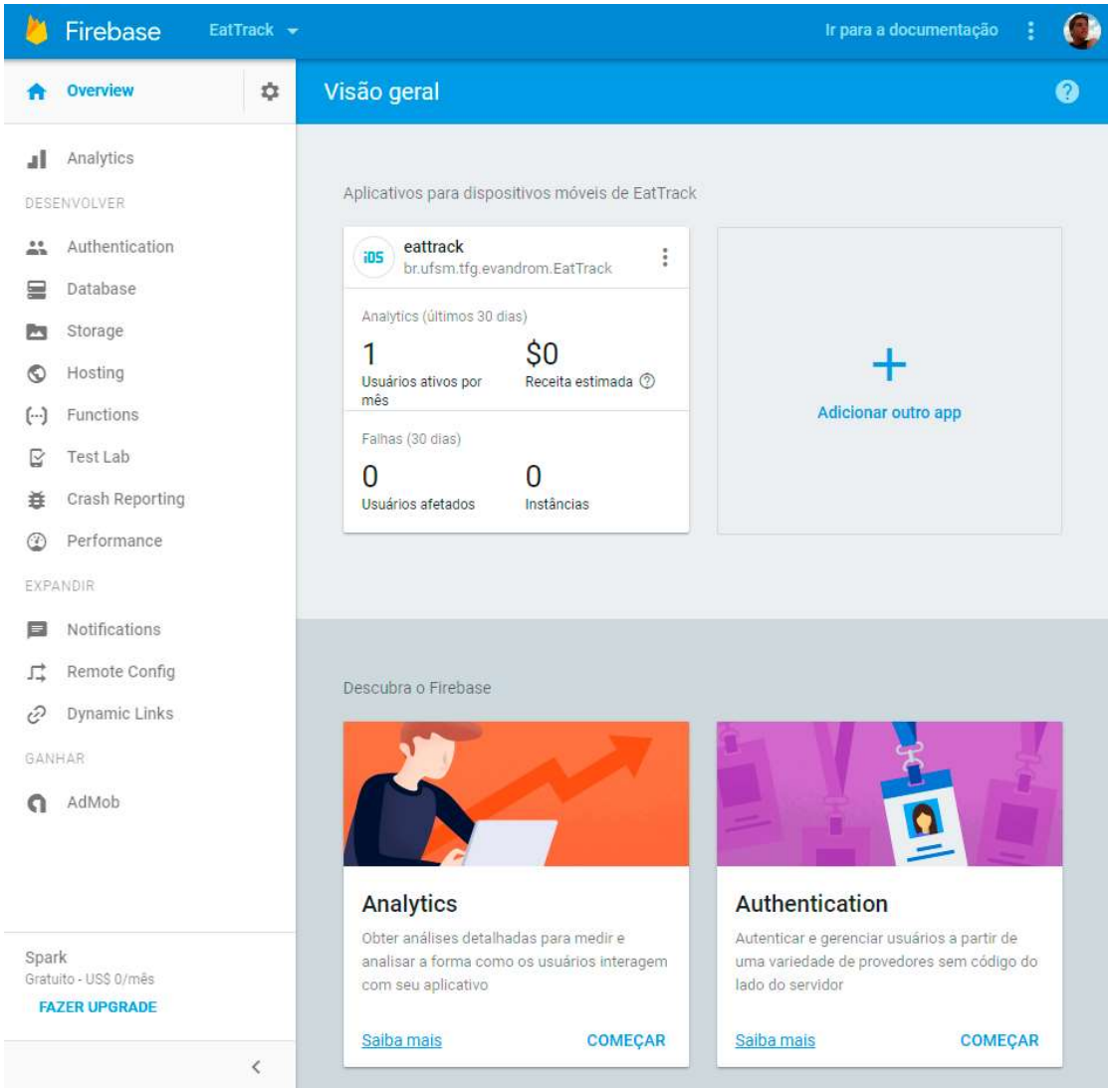

Figura 1 - Console de uma aplicação no *Firebase*

Para desenvolver o aplicativo, o *Firebase* facilita muito o trabalho do programador, uma vez que o serviço está configurado no aplicativo, o acesso aos dados hospedados é feito através de objetos, da mesma forma que já criamos classes dentro de um programa.

Internamente o serviço armazena os dados do formato *JSON* ao invés de utilizar um modelo de dados relacional como a maioria dos bancos utilizados para este caso. Como para este projeto os serviços do *Firebase* têm a finalidade de simular o banco de dados real para deixar o desenvolvimento desta ferramenta mais independente do trabalho no qual ela se engloba.

O formato utilizado por ele para salvar os dados não é tão relevante e não teve influência na escolha da ferramenta. O serviço é capaz de armazenar e distribuir todas as informações necessárias ao projeto de forma satisfatória.

2.4 HealthKit e ResearchKit

Estas ferramentas foram desenvolvidas pela *Apple* para serem utilizadas nas suas plataformas, para aprimorar aplicativos da área da saúde e permitir o compartilhamento destes dados entre projetos para pesquisa, criando um ecossistema.

Foi introduzido dentro do sistema operacional dos *iPhones* um aplicativo chamado de *Saúde* que funciona como um centralizador de informações sobre o usuário, ele reúne em um único lugar dados de nutrição, qualidade do sono, atividades físicas, medidas corporais, registros médicos, entre outros. Estes dados são coletados de diversos aplicativos e inclusive de outros dispositivos como o *Apple Watch*, que é um relógio inteligente da empresa e entre outras funções monitora os sinais vitais dos usuários.

Os aplicativos desenvolvidos para o *iOS* podem solicitar permissão para ler os dados salvos dentro deste serviço, com a permissão explícita do usuário. Esta *API* chamada de *HealthKit* que é utilizada para buscar os dados que aplicativos de terceiros estão inserindo sobre nutrição para serem tratados e enviados para o profissional nutricionista responsável pelo paciente. (APPLE, 2017)

Rocha et al. (2015) citam as aplicações da API, eles afirmam “O Health Kit é um framework nativo do sistema iOS que funciona como repositório de informações sobre a saúde do usuário, tais como: peso, altura, frequência cardíaca, gasto calórico etc.”

Assim mantemos a ideia de que se um usuário já costuma utilizar algum aplicativo para controle da sua dieta, esses dados que ele já está habituado a inserir não serão desperdiçados sem que ele precise se acostumar à uma nova rotina.

O *ResearchKit* é um *framework* de código aberto introduzido pela *Apple* para reunir desenvolvedores e pesquisadores na criação de aplicativos para medicina, idealizado para funcionar em conjunto com o *HealthKit*, esta ferramenta busca dar ainda mais poder aos desenvolvedores para captar e obter dados compartilhados por outras fontes. (JARDINE, FISHER, e CARRICK, 2015)

Além de aumentar o acesso a dados de pesquisa na área, este *framework* oferece uma série de ferramentas para dar mais poder na aquisição de dados relevantes, como padronizações para enquetes e pesquisas, ferramentas gráficas completas para aquisição e apresentação de dados em tempo real, buscando dar unidade à forma como estes dados são adquiridos e representados por diversos aplicativos e sistemas.

Figura 2 - Telas do *App Saúde* instalado nos dispositivos da *Apple*

Estas tecnologias foram decisivas na opção pela plataforma *iOS* na definição do projeto, pois são fatores importantes que colaboram com a aquisição de dados, visto que muito usuários que buscam profissionais de nutrição já experimentam aplicativos de controle de alimentação, e se estes compartilham seus dados com o sistema, nenhuma informação será desperdiçada, e o projeto aqui descrito realizará a intermediação entre os aplicativos de terceiros e o sistema ao qual este trabalho será incorporado.

O *Healthkit* salva os dados de uma maneira muito característica, diferente dos aplicativos pesquisados e da proposta deste trabalho, o sistema da *Apple* salva dados nutricionais do usuário. Ao invés de guardar a informação do alimento que foi consumido, como o *HealthKit* foi criado com a intenção de pesquisa, ele salva dados como quantidade de proteína, cálcio, vitaminas, gorduras, entre outros. Ao total são 40 medidas destas que o sistema se utilizará, além das propriedades dos alimentos que são ingeridos, este trabalho busca no sistema entradas referentes ao peso do paciente, visto que esta pode ser uma informação útil ao nutricionista e está disponível na ferramenta.

2.5 Sistemas de Monitoramento de Nutrição

Sistemas de monitoramento já são utilizados para analisar dados de pacientes nutrição, como por exemplo para estudar grupos ou populações e obter informações para formação de políticas públicas na área de nutrição.

No Brasil são implementados os *SISVAN*, que são sistemas de vigilância e que atualmente por lei, segundo Venânicio (2007), são requisitos básicos para obtenção de repasses de recursos financeiros na área de saúde, buscando criar perfis cada vez mais precisos sobre a população alvo dos investimentos solicitados, buscando que os valores investidos sejam cada vez mais eficazes.

> *"O Sistema de Vigilância Alimentar e Nutricional (SISVAN) corresponde a um sistema de coleta, processamento e análise contínuo dos dados de uma população, possibilitando diagnóstico atualizado da situação nutricional, suas tendências temporais e,*

> *também, dos fatores de sua determinação. Contribui para que se conheçam a natureza e a magnitude dos problemas de nutrição, caracterizando grupos sociais de risco e dando subsídios para a formulação de políticas e estabelecimento de programas e intervenções."* (VENANCIO, 2007)

2.6 Aplicativo Referência

Para auxiliar no levantamento dos requisitos do trabalho e servir como aplicativo de referência para realizar o estudo de caso no qual buscamos dados compartilhados inseridos por outras fontes no sistema operacional, foi escolhido o *MyFitnessPal*. Por ser um dos aplicativos do gênero mais baixados da *App Store*, a loja de aplicativos da *Apple*, permanecendo por 4 anos em primeiro lugar na categoria *Saúde e Fitness*, e por realizar um monitoramento das refeições nos mesmos moldes proposto por este trabalho, este foi o aplicativo escolhido.

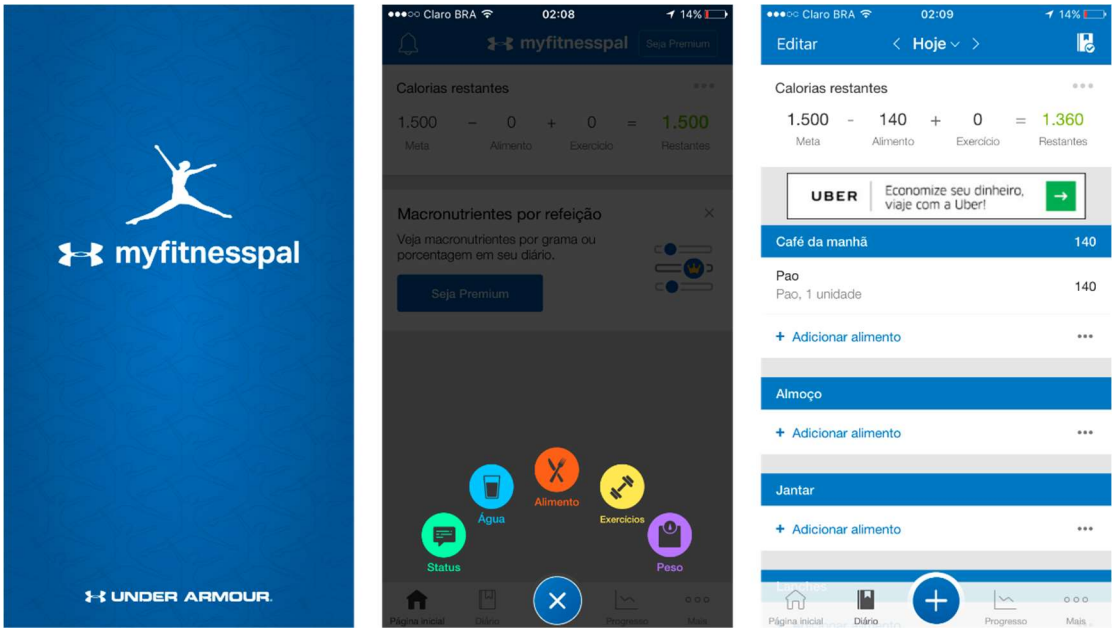

Figura 3 - Telas do aplicativo *MyFitnessPal* utilizado como referência

Ele oferece uma quantidade grande de ferramentas, algumas delas parte de um pacote pago de mensalidade, como criação de metas, relatórios de progresso e uma base de dados de alimentos com valor nutricional e quantidade de calorias. Ele também compartilha os dados que coleta com o sistema operacional para que o aplicativo desenvolvido neste trabalho possa buscar e compartilhar com o nutricionista, neste contexto, este foi o aplicativo que mais se encaixou com os objetivos do projeto e foi o escolhido para servir de referência.

3 MODELAGEM DO PROJETO

Neste capítulo são descritos os processos de planejamento aplicados neste projeto, os requisitos levantados para o aplicativo cumprir os objetivos propostos, a especificação dos casos de uso. Também são apresentados os diagramas utilizados para modelar a estrutura e o comportamento do software.

3.1 Requisitos

Para servir de base para a implementação e assim garantir o sucesso no desenvolvimento do aplicativo, foram definidos requisitos para o aplicativo que especificassem de forma objetiva as funcionalidades esperadas do mesmo, garantindo que ao final do desenvolvimento o resultado obtido no projeto atendesse as expectativas iniciais e cumprisse de forma satisfatória os objetivos definidos previamente.

O principal requisito funcional do projeto é responder de forma direta e clara às recomendações do profissional nutricionista, que definirá previamente cada refeição do paciente, e este deverá responder se conseguiu cumprir a meta definida ou se, por ignorar a refeição ou por outras circunstâncias precisar mudar o cardápio, acabou não cumprindo a recomendação.

As prescrições de cardápios feitas pelos nutricionistas são de responsabilidade do sistema de *homecare*, este aplicativo apenas solicita ao servidor do sistema as informações que precisam ser apresentadas ao usuário. O profissional de nutrição acessa a sua interface no sistema para criar uma recomendação de refeição, esta recomendação é realizada inserindo cada alimento, que está previamente cadastrado no sistema, e diz a quantidade de cada um desses alimentos, por exemplo, duas colheres de arroz ou um bife de frango.

Outro requisito muito importante é ter a capacidade de buscar informações sobre o estado nutricional do paciente e disponibilizar estas informações no sistema, assim usuários que já utilizam outros aplicativos e desejam continuar utilizando não

precisam mudar seus hábitos. Pela forma peculiar como estes dados são salvos no sistema operacional do dispositivo, não é possível definir se a meta passada foi cumprida ou não, mas ainda assim é possível buscar informações importantes para os nutricionistas se basearem em seus diagnósticos.

Também são requisitos deste aplicativo, que os usuários precisem realizar *login* para acessar suas informações, e apenas visualizar seus dados, garantindo o sigilo das informações de cada paciente. E que eles sejam capazes de acessar um histórico das refeições realizadas, para não deixar esta informação apenas com o nutricionista, mas também com os pacientes, e assim dar uma noção global das refeições que conseguiu realizar corretamente e as que não conseguiu seguir a recomendação prescrita.

Além destas, como requisitos não funcionais, o software deve ser de uso fácil, evitando que pacientes deixem de adicionar as informações devido a achar o manuseio de aplicativo complicado. O aplicativo também não deve consumir muitos dados móveis, buscando dados apenas quando necessário e evitando assim utilizar a rede de dados.

3.2 Casos de Uso

Este projeto possui apenas três cenários para os usuários, um deles é o fluxo principal de utilização do aplicativo no qual o usuário responde às recomendações do seu nutricionista, outro cenário, e que será menos utilizado, é aquele em que o paciente apenas dá a permissão ao sistema para buscar os seus dados que ele compartilha de outros aplicativos. Por fim o último cenário é quando o usuário acessa o histórico das refeições que realizou no aplicativo

Em todos os casos o usuário precisa estar autenticado no sistema, então seja para receber sua próxima meta de refeição, seja para configurar o seu aplicativo com o *App Saúde*, ou para acessar o seu histórico, ele precisará se autenticar utilizando *e-mail* e senha, conforme citado anteriormente esse processo utiliza o *Firebase* como infraestrutura para autenticação e gerenciamento das sessões.

No fluxo principal do aplicativo o usuário entra e recebe a próxima refeição que precisa realizar, com o horário recomendado e o cardápio escolhido pelo nutricionista,

ele terá duas opções de interação apenas, confirmar que conseguiu realizar com sucesso a meta estipulada ou dizer que não conseguiu cumprir a recomendação, assim que enviar o resultado de uma meta, o sistema automaticamente busca a próxima meta no servidor.

O outro cenário é bem mais simples e na maioria dos casos será realizado apenas uma vez, aqui o usuário apenas precisa dar a permissão para o aplicativo receber acesso às informações compartilhadas. O paciente irá nas configurações do aplicativo e marcará que deseja utilizar o *App Saúde*, ao fazer isso pela primeira vez o sistema operacional perguntará se ele tem certeza que deseja dar permissão ao aplicativo.

Uma vez configurado o sistema se encarrega de periodicamente, quando receber tempo de processamento do sistema operacional, executar em background uma rotina que busca os dados do *HealthKit* e realiza o envio ao servidor, tudo transparente ao usuário.

O último cenário é onde o usuário acessa as configurações para visualizar seu histórico, lá ele terá um panorama das refeições que conseguiu cumprir as recomendações e aquelas em que falhou, o sistema listará as refeições das mais recentes para as mais antigas, mostrando quais foram realizadas com sucesso e quais o usuário falhou.

Como a ferramenta aqui desenvolvida se comunica com o servidor do sistema de *homecare* e salva as informações lá, as interações de outros atores como o nutricionista são todas realizadas neste sistema. A aplicação desenvolvida neste trabalho interage diretamente apenas com os pacientes.

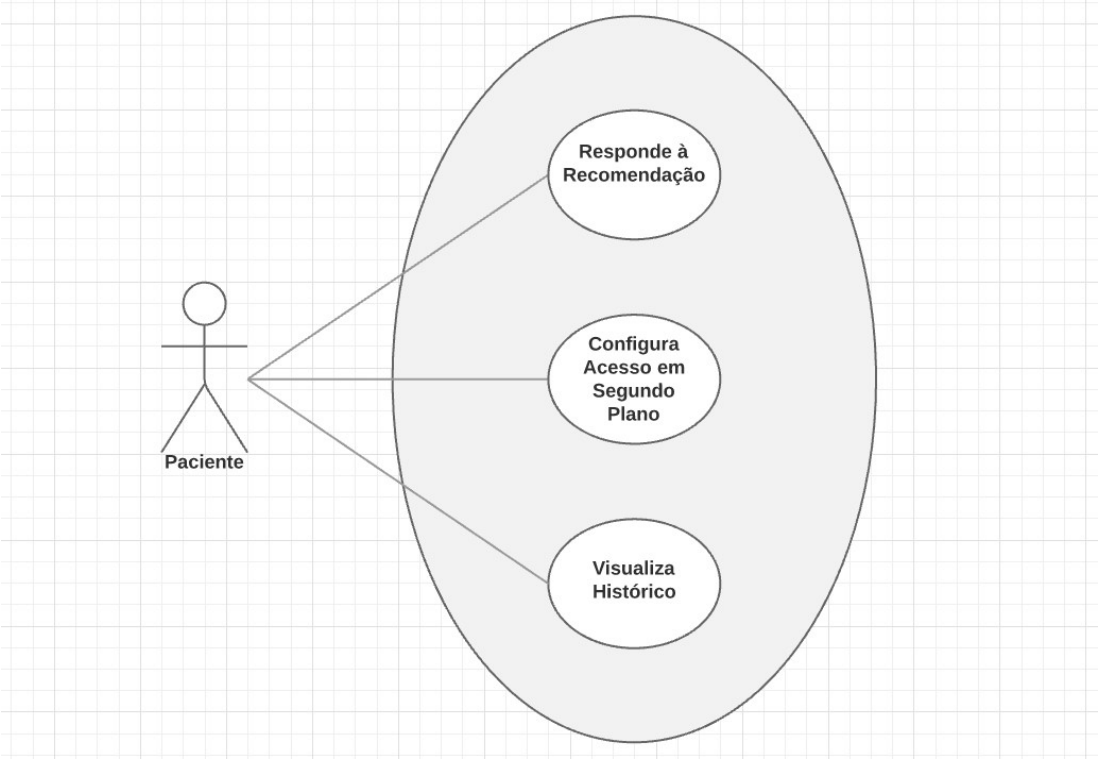

Figura 4 - Casos de Uso

3.3 Diagramas

Afim de modelar com mais precisão a estrutura do sistema, foram criados dois diagramas, primeiramente o diagrama de classes e logo após o modelo relacional do banco de dados. Com a estrutura necessária para o software funcionar e o conhecimento do banco de dados com o qual será feita a comunicação como ponto de partida, o desenvolvimento foi estruturado em cima dos diagramas para manter-se no objetivo proposto.

3.3.1 Diagrama de Classes

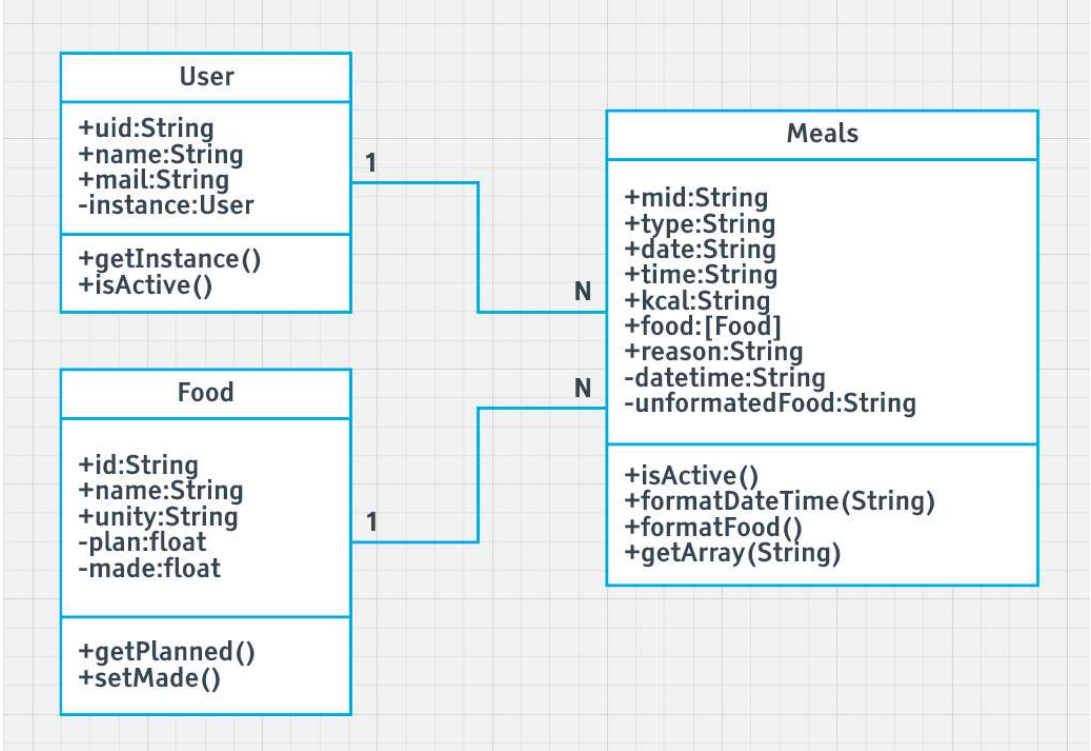

Figura 5 - Diagrama de classes

Este é um dos mais importantes diagramas do UML pois serve de apoio para a grande maioria dos outros diagramas, por ser um projeto não tão complexo, ele já foi suficiente para atender as expectativas de suporte ao desenvolvimento deste projeto. Como a persistência dos dados e as informações geradas são todas mantidas na infraestrutura do *Firebase*, as classes que foram criadas no código do aplicativo são utilizadas apenas para receber e centralizar as informações do servidor, e quando necessário realizar o tratamento dos dados e prepará-los para serem enviados ao banco de dados online.

São três classes apenas que foram modeladas para a programação, a classe de usuários, a classe das refeições e a classe dos alimentos que compões uma refeição, elas são usadas basicamente para receber os dados do servidor remoto e lidar com essa informação dentro do programa, garantindo que as mesmas informações sejam utilizadas independente da tela que o programa está exibindo ou do método que está sendo executado.

A classe de usuários foi implementada seguindo o padrão de desenvolvimento de software *Singleton*, este *Design Pattern* serve para garantir que sempre seja utilizada a mesma instância de uma classe, para garantir a segurança dos dados esta

foi uma estratégia utilizada, além de verificar a cada requisição se a sessão do usuário permanece ativa.

Além do método estático foi implementada uma função para verificar se existe um usuário ativo no momento, este método é utilizado dentro da aplicação para evitar que operações sejam realizadas sem um usuário autenticado. Além do e-mail que é parte da autenticação, esta classe armazena o nome do paciente, o que é suficiente para utilizar a interface criada no projeto, futuramente se mais informações sobre os usuários forem necessárias para as interações do usuário, é nesta classe que estão reunidas todas informações relativas ao paciente.

Outra classe definida foi a das refeições, cada instância desta classe representa uma refeição, com seu tipo (Jantar, Almoço, Café...), data, horário recomendado, o valor calórico para informar o usuário e o cardápio prescrito pelo nutricionista, estas informações vem do servidor da aplicação e são armazenadas em um objeto para receber tratamento quando necessário.

Além de um método semelhante à classe do usuário para verificar se a instancia possui uma refeição completa, esta classe oferece funções para formatar data, horário e a lista de alimentos do cardápio para exibição na *view*. Da mesma forma ela é capaz de tratar as entradas do usuário e deixar prontas para serem enviadas para o servidor no formato apropriado.

Deixando essas operações dentro de uma classe, respeitamos o padrão de projeto MVC, jogando todas as operações referentes ao modelo de representação dos dados numa classe apropriada chamada de *model* pelo padrão, e tornando o *controller* independente da implementação para exibir as informações na *view.*

A última classe representa os alimentos que compõem o cardápio prescrito, cada alimento, que faz parte de um *array* de alimentos dentro de um cardápio, possui seu nome, a unidade, por exemplo colheres ou gramas, e a quantidade planejada pelo nutricionista.

Além disso a classe também armazena o valor que o usuário realmente consumiu, o aplicativo mostra a quantidade recomendada e o usuário ao responder que conseguiu cumprir aquela recomendação, indica a quantidade de cada alimento que ingeriu, tornando as informações ainda mais precisas para serem trabalhadas no sistema de *homecare* e apresentadas ao nutricionista.

Com uma tabela de alimentos, será possível para o sistema de homecare trabalhar com mais precisão estes dados, pois cada alimento é uma entrada única,

podendo por exemplo saber quantas vezes este alimento aparece na dieta, ou quais alimentos os pacientes têm mais problemas em seguir às recomendações prescritas..

3.3.2 Modelo Relacional

Este modelo é usado para representar a implementação do banco de dados que o aplicativo usará para buscar as informações. Como citado, a infraestrutura não utiliza um banco de dados relacional e sim salva os dados formatados em JSON. Mas esta infraestrutura é utilizada neste trabalho apenas para fins de simulação, a aplicação real utiliza um banco relacional e esta modelagem também pode ser utilizada para representar os dados armazenados pelo *Firebase*. A compreensão do funcionamento do banco é a mesma.

Este modelo representa apenas as necessidades deste trabalho, e não toda a modelagem do banco de dados da aplicação para a qual esta ferramenta será integrada, para este aplicativo o restante da aplicação não tem importância desde que o sistema seja capaz de enviar e receber os dados no formato que este projeto requer para representar e salvar o conteúdo.

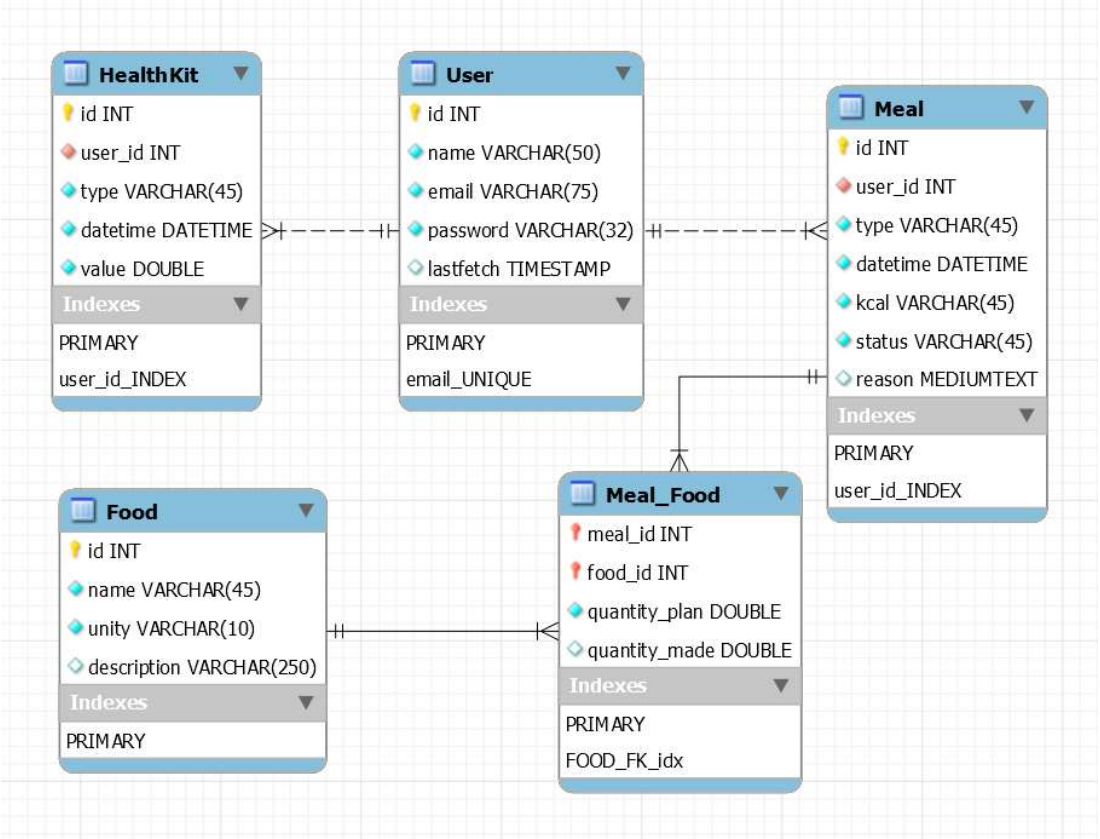

Figura 6 - Modelo relacional do banco de dados

A tabela de usuários precisa ter as informações do paciente necessárias para exibir seus dados na visualização das telas, também foi inserida a informação da última vez que este usuário realizou uma pesquisa nos dados compartilhados com o *HealthKit* para evitar que sejam enviados dados duplicados. Depois existem outras duas tabelas que se referenciam a ela, estas tabelas representadas no formato de um *JSON* ficam dentro de um usuário ao invés de conter uma chave estrangeira como referência ao usuário, mas a compreensão é a mesma.

Uma destas tabelas é a tabela de refeições, na qual cada entrada representa uma prescrição de alimentação, com as informações que a classe representa no código, mas além disso a tabela salva informações referentes ao estado da entrada, se ainda está pendente ou se o usuário já respondeu se cumpriu ou não a meta prescrita, e uma informação textual sobre porque a meta não foi cumprida e o que foi consumido no lugar do cardápio recomendado, para os casos onde o usuário não conseguiu respeitar a sugestão do nutricionista.

Para facilitar o trabalho de mineração futura dos dados, cada alimento que compõem uma refeição, está presente em uma tabela própria, evitando assim que

buscas textuais tenham que ser realizadas no futuro. Uma tabela intermediária une os alimentos a uma refeição, especificando a quantidade recomendada pelo nutricionista e após a realização da refeição pelo paciente, a quantidade informada que foi realmente consumida.

Outra tabela presente é a que representa os dados disponibilizados pelo *HealthKit*, que por serem modeladas de forma distinta pelo *framework* criado pela *Apple*, precisaram ser salvar em uma tabela à parte que respeita a natureza destes dados.

Cada entrada na tabela representa um tipo de dado, que define se são dados de peso do usuário, de consumo de proteínas, gorduras, vitaminas, dentre outros. Também representa a data e hora da entrada destes dados no sistema e o valor inserido pelo paciente. Cada entrada de dados de qualquer fonte no *App Saúde* é representada aqui por uma única entrada nesta tabela.

3.4 Fluxo de Dados

Dentro da modelagem do projeto, desde o início buscou-se definir como as partes envolvidas se comunicariam, já que o aplicativo desenvolvido além de se comunicar com o servidor online, que no desenvolvimento deste trabalho foi utilizado o *Firebase* substituindo o servidor da aplicação real em que este trabalho se encaixa, ainda precisa se comunicar com o *App Saúde* que centraliza informações que os usuários disponibilizam em outras fontes ou outros aplicativos e compartilham com o sistema operacional.

O aplicativo trabalha com dois cenários bem distintos, no qual cada um destes cenários representa a comunicação com uma entidade diferente. Enquanto o fluxo normal de utilização do aplicativo se comunica exclusivamente com a infraestrutura que armazena os dados, seja para buscar as refeições pendentes, seja para enviar a resposta dada pelo usuário, este processo recebe um dicionário, que para a linguagem *Swift* é a representação de um *array* com chave e valor, e devolve os dados no mesmo formato, no qual as informações são a exata reprodução dos atributos das tabelas do banco de dados modeladas no capítulo anterior.

Figura 7 - Representação das comunicações entre entidades

Do outro lado temos a comunicação do aplicativo com as informações compartilhadas por outros aplicativos com o sistema operacional. O *App Saúde* armazena estes dados de um modo bem particular como descrito anteriormente, por isso as suas informações precisam ser tratadas e enviadas para uma tabela diferente. Estas métricas não são armazenadas no aplicativo, que recebe os dados e salva no servidor para uso do sistema no qual esta aplicação se insere.

Esta comunicação só é realizada por um processo que roda em *background* pelo aplicativo e é completamente transparente ao usuário, depois de buscar as informações junto do sistema, é realizada uma comunicação com o servidor de banco de dados para salvar os mesmos e disponibilizar estes para o acesso do nutricionista responsável pelo paciente.

4 DESENVOLVIMENTO

Este capítulo tem o objetivo de descrever todas as etapas do desenvolvimento do aplicativo para *iPhone*, explicando como foi feita a implementação de cada parte do projeto, desde a configuração da infraestrutura na nuvem, até a programação do aplicativo no *xCode*.

4.1 Configuração da Infraestrutura

O primeiro passo do desenvolvimento foi preparar a utilização do *Firebase*, o processo é realizado através de um console numa interface *web*, selecionando quais serviços foram utilizados dentro da plataforma. Para este projeto utilizamos o banco de dados e a autenticação dos usuários.

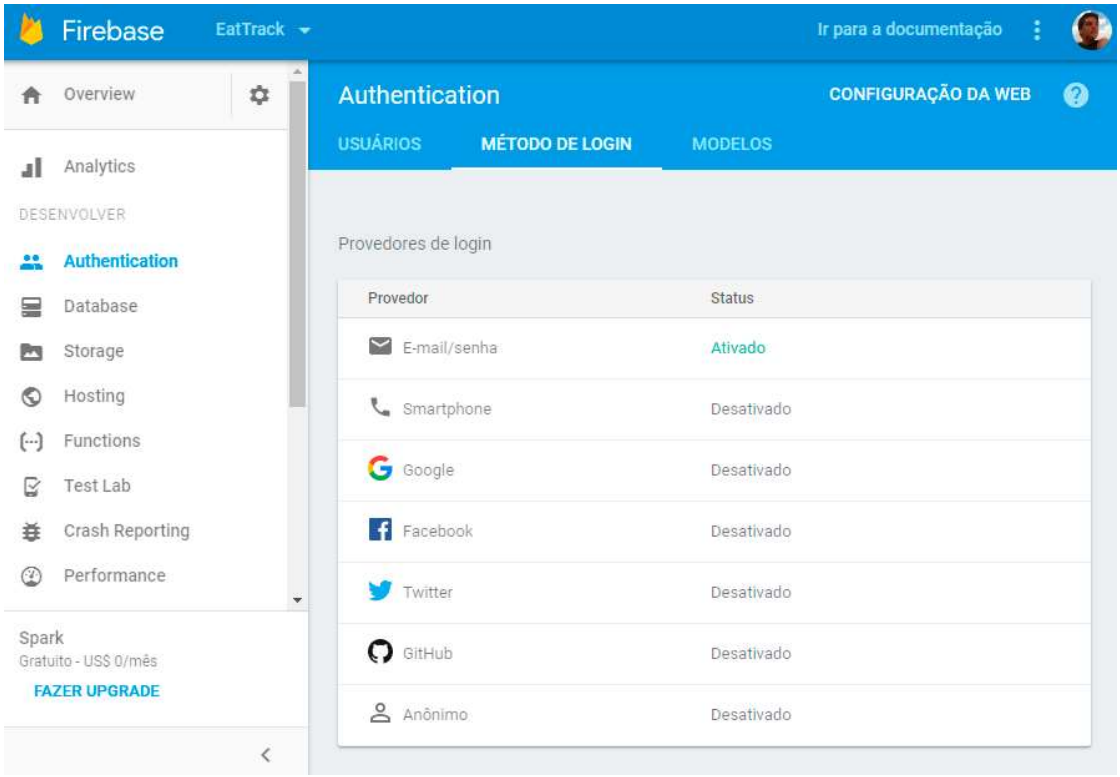

Figura 8 - Configuração de autenticação no *Firebase*

Para configurar a autenticação é necessário escolher o método, eles oferecem opções de login com e-mail e senha, que foi a opção escolhida, e também opções de utilizar contas da *Google* ou *Facebook* por exemplo. Dentro do console é possível criar usuários diretamente ou cadastrar novos utilizando a API deles. Como a intenção é simular um banco de dados real da aplicação, e nesta os pacientes serão previamente cadastrados pelo nutricionista, então dentro do aplicativo não haverá cadastro de novos usuários, apenas login, e um usuário de testes foi inserido manualmente no console.

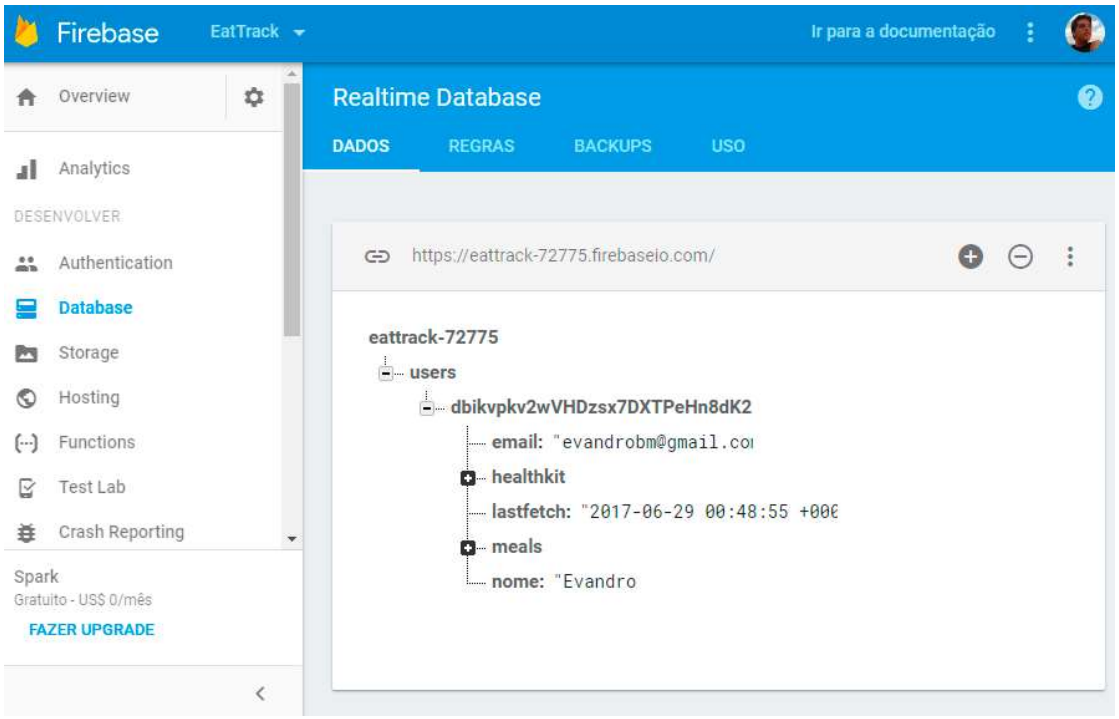

Figura 9 - Configuração do banco de dados do *Firebase*

O banco de dados precisa apenas ser ativado, que o *Firebase* já disponibiliza um nó raiz do JSON utilizado para armazenar os dados, e toda informação salva após é inserida dentro deste nó. Seguindo a modelagem proposta do banco de dados, foram inseridos dados de acordo para simular os dados do usuário de teste, e algumas refeições para alimentar os testes do aplicativo.

Dentro do código do aplicativo, é necessário carregar um arquivo de informações que pode ser baixado na interface da plataforma, e instalar as classes do *Firebase* dentro do projeto, essa instalação se torna bastante simples utilizando o *pod*, que é um gerenciador de dependências que ao ser inicializado gera um arquivo onde

descrevemos todas as dependências do projeto, no caso as classes *Firebase, FirebaseAuth* e *FirebaseDatabase*, e ao rodar um comando de instalação, o gerenciador se encarrega de baixar todos os arquivos necessários e *linkar* os mesmos ao projeto do xCode.

A utilização dos dados de autenticação e o acesso à base de dados é feita por meio das classes importadas pelas bibliotecas do *Firebase*, estas abstraem toda a parte de comunicação pela rede, tornando o acesso aos dados intuitivos como se fossemos acessar atributos de uma classe. Por necessitar se comunicar com o servidor através de uma conexão com a internet, a busca de dados do banco é realizada através de chamadas assíncronas, evitando o travamento do aplicativo enquanto os dados são retornados pela rede.

4.2 Criação da Interface

O próximo passo no desenvolvimento foi a criação da interface do aplicativo, utilizando a ferramenta chamada de *Interface Builder* da IDE de desenvolvimento padrão da *Apple* para as suas aplicações, o *xCode*. Aqui utilizamos um arquivo chamado de *StoryBoard* que reúne as informações da nossa interface, que é criada de maneira totalmente visual, arrastando e soltando os elementos de UI diretamente dentro de cada *view*, as ligações entre elas para montar o fluxo da aplicação também são definidas aqui.

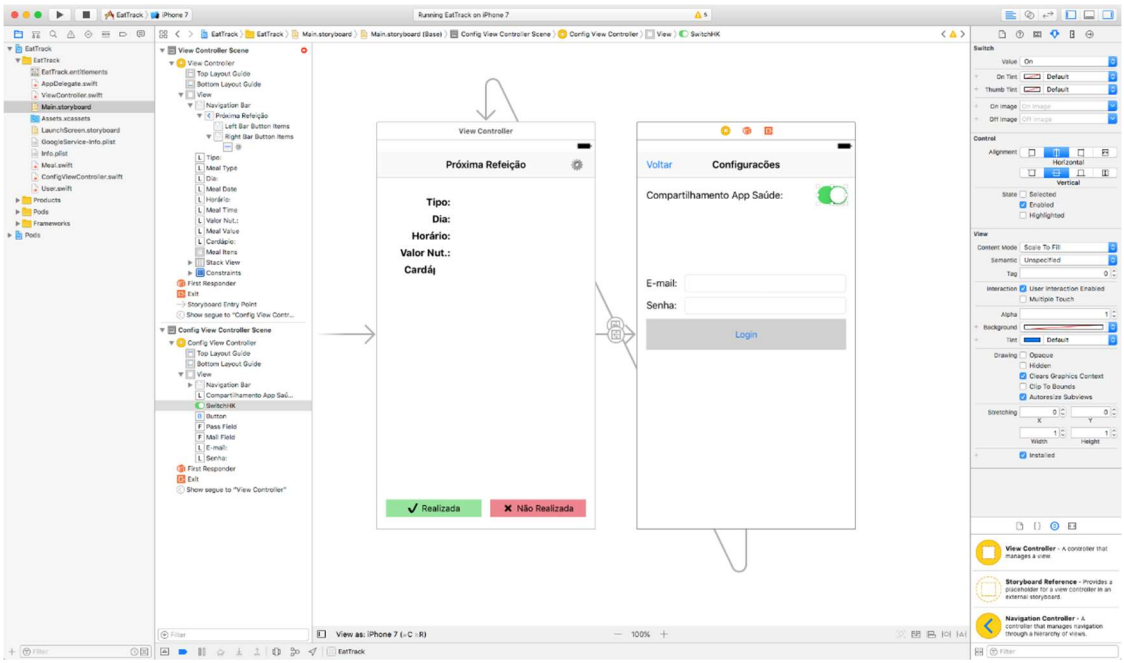

Figura 10 - Tela da IDE de desenvolvimento para criação de interfaces

Seguindo o planejamento proposto para atender aos requisitos, foram criadas duas *views* apenas, para tentar manter o foco na simplicidade de uso da aplicação. A primeira que é encarregada de toda operação de exibir a próxima refeição a ser realizada e reunir os elementos de interação para obter do usuário o retorno se a meta foi cumprida ou não. E uma tela que reúne as configurações do aplicativo, aqui o paciente configura se deseja compartilhar os dados reunidos no *App Saúde* com o seu nutricionista e tem a opção de se autenticar para fazer o login ou logout do sistema. Quando o usuário não está com o login ativo, ele é redirecionado para esta tela para realizar seu login.

4.3 Programação dos *Controllers*

Com a infraestrutura pronta para o uso, e as *views* desenhadas, é necessário criar a lógica que utiliza dos modelos de dados para representar os mesmos na interface e gerenciar as ações do usuário.

O primeiro de cada operação é verificar se a sessão do usuário permanece ativa, esta verificação precisa ser sempre realizada para garantir segurança dos

dados. Ao identificar o usuário é realizada uma consulta ao banco de dados afim de obter quais refeições estão programadas para este paciente e que ainda estão pendentes de resposta do usuário, o retorno dessas informações já vem ordenado conforme o nutricionista determinou no sistema. A primeira é exibida ao usuário, e o mesmo só tem acesso à próxima quando enviar o *feedback* da atual, garantindo que todas as recomendações serão seguidas e que nenhuma ficará sem a informação de que foi realizada ou não.

A conexão com o servidor de banco de dados é assíncrona e contínua, o controlador fica sempre ativo gerenciando a interface e se comunicando com a infraestrutura na nuvem. Se uma nova informação for inserida no servidor com uma recomendação anterior a que está sendo exibida, imaginando por exemplo que o nutricionista esqueceu de recomendar um café da manhã e o paciente está visualizando de manhã a refeição do almoço, assim que a nova informação é inserida o aplicativo recebe este novo dado e exibe o mesmo, visto que sempre a refeição mais antiga entre as pendentes é que é exibida na interface, se um dado anterior chegar ele precisa substituir o que vem na sequência.

Para o caso específico do não cumprimento de uma meta, é disparado um alerta ao usuário solicitando que ele informe porque não realizou a meta e qual foi o cardápio alternativo utilizado para aquela refeição, assim o profissional responsável terá noção de quão fora da dieta o paciente está sempre que não conseguir cumprir uma recomendação.

O controlador da tela das configurações é o responsável por informar ao paciente se o compartilhamento dos dados do *HealthKit* está ativo e receber as ações do usuário para ativar ou desativar esta função. E gerenciar o estado de autenticação do usuário, ele recebe as credenciais para realizar o login e quando um usuário já está logado é ele o responsável por solicitar o fim da sessão ao servidor de autenticação do *Firebase*.

Por fim é necessário informar ao sistema qual a função que ele delegará o tempo de execução em segundo plano que foi solicitado. Esta função deve verificar qual o usuário com sessão ativa e a permissão dele para buscar os dados compartilhados de outras fontes. Após essas verificações, ela é responsável por solicitar à API do *HealthKit* os dados desde a última solicitação deste usuário e abrir uma conexão com o banco de dados para salvar os dados do usuário no servidor. Não temos como saber a periodicidade exata com que o sistema operacional rodará o

aplicativo em segundo plano, visto que esta função varia de acordo com diversos fatores como a quantidade de processos utilizando o processador, por isso foi necessário salvar o momento da última atualização dos dados. Desta forma também conseguimos manter o nutricionista informado da última vez em que estes dados foram lidos.

5 RESULTADOS OBTIDOS

Este capítulo irá demonstrar os resultados obtidos com o desenvolvimento do aplicativo, mostrando o aplicativo criado e o seu fluxo de funcionamento, bem como mostrar as interações do mesmo com o servidor afim de demonstrar a realização dos requisitos propostos neste trabalho.

5.1 Aplicativo Simulado

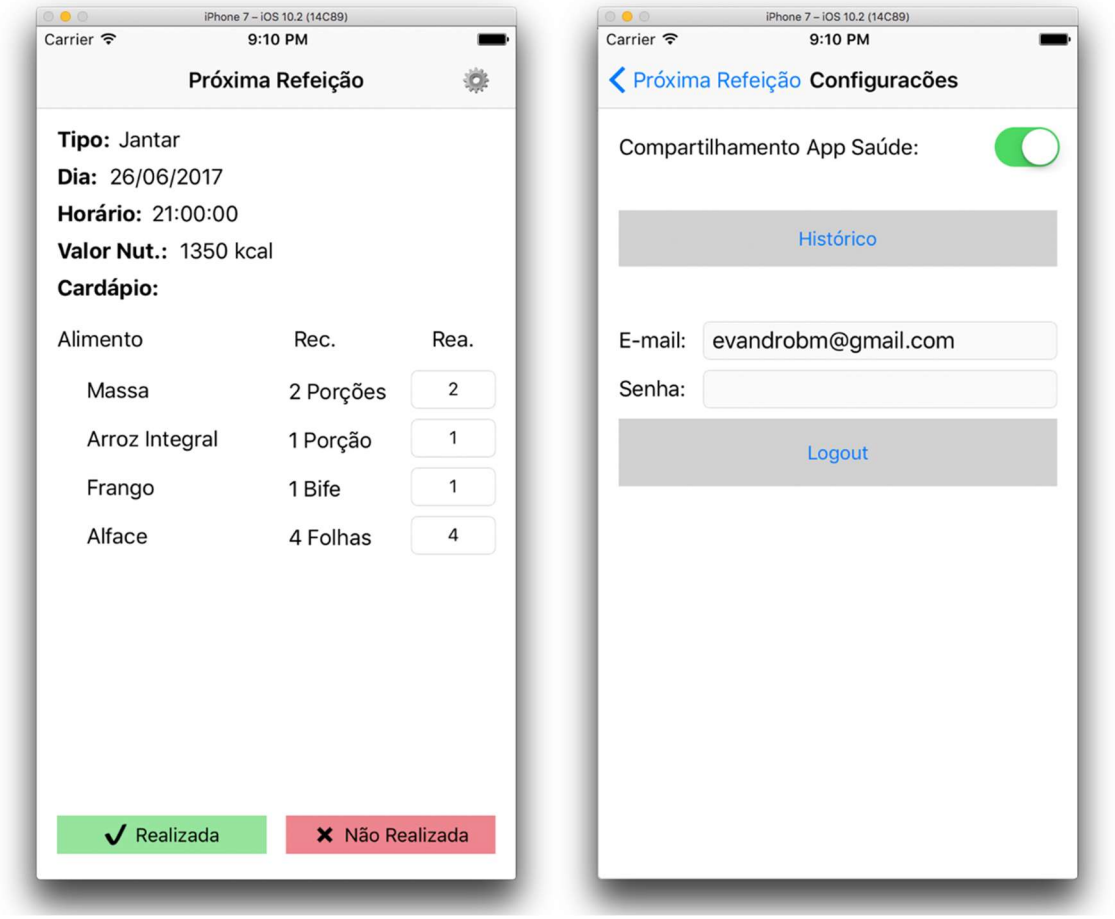

Figura 11 - Telas do aplicativo desenvolvido

A *Apple* tem um controle muito rígido sobre os aplicativos que são desenvolvidos e instalados em seus dispositivos, devido a isto somente

desenvolvedores cadastrados no programa de desenvolvimento, que é pago, conseguem testar os aplicativos em seus dispositivos reais. Desenvolvedores que não são assinantes do programa são capazes somente de testar suas aplicações no simulador incorporado à IDE de desenvolvimento, o que traz algumas limitações, mas não impediu a obtenção e documentação dos resultados obtidos com o desenvolvimento deste trabalho.

Estas acima são as telas que o sistema possui, a primeira na qual os usuários realizam as interações com a aplicação, nela são exibidas as recomendações do nutricionista com todas as informações necessárias ao paciente, como o cardápio escolhido, a recomendação do horário, informações do valor calórico e o tipo da refeição.

A segunda é a tela das configurações, responsável por gerenciar a sessão do usuário, a preferência dele sobre a utilização da ferramenta para compartilhar os dados do *HealthKit* com o nutricionista e um botão de acesso à terceira tela do histórico. Esta última lista em ordem decrescente de data e hora as refeições do usuário mostrando se foram realizadas com sucesso ou não.

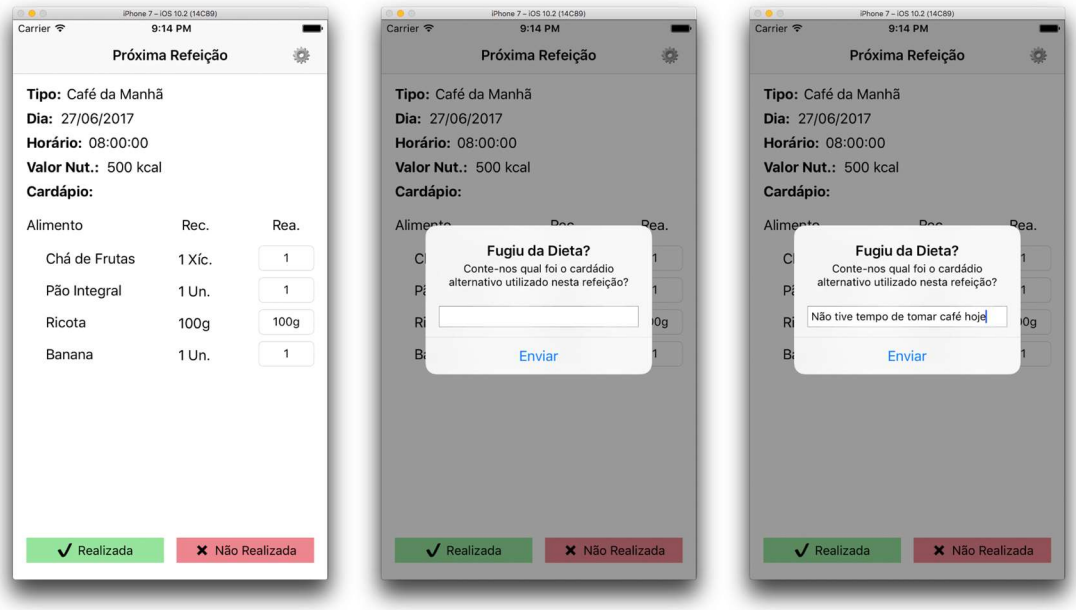

Figura 12 - Exemplo de fluxo quando o usuário não realiza a meta proposta

Para manter uma experiência de uso simples e agradável, o usuário recebe uma recomendação de refeição e as suas opções são responder positivamente ou negativamente quanto ao sucesso em realizar a recomendação do nutricionista. Como

pode ser visualizado na imagem abaixo, quando um usuário não conseguir realizar uma das metas, será questionado sobre o cardápio utilizado para aquela refeição. O usuário também pode dizer que conseguiu cumprir a meta, mas alterar as quantidades consumidas de cada alimento do cardápio, por exemplo caso ele tenha comido apenas uma colher a mais de arroz, não precisa dizer que falhou e apenas mostrar que ingeriu uma quantidade maior.

Quando responde à uma solicitação, esta é removida da interface e automaticamente a próxima é exibida, já deixando o usuário ciente da seguinte meta de refeição. No momento em que o usuário responde para o aplicativo, uma comunicação é realizada com o servidor e somente com o sucesso na atualização do banco de dados remoto é que a próxima meta é exibida.

Nos testes realizados o resultado foi satisfatório, com o fluxo de operação seguindo conforme o esperado pelo trabalho, a comunicação com o servidor é muito rápida, o que se deve também à baixa quantidade de dados trafegados pela rede em cada solicitação, o que torna este processo de comunicação remota transparente ao usuário.

Vale ressaltar também que a primeira vez que o paciente ativar o compartilhamento dos seus dados do *HealthKit*, o sistema operacional perguntará ao usuário se ele realmente deseja permitir que o aplicativo tenha acesso a cada uma das características solicitadas, que como citado anteriormente, para este trabalho são quarenta ao total, o usuário ainda tem a opção de liberar somente alguns dados e outros não, essa permissão pode ser revogada a qualquer momento dentro do *App Saúde*.

5.2 Monitoramento em Segundo Plano

A utilização de processos em segundo plano segue um modelo bem definido, o aplicativo precisa solicitar ao sistema operacional tempo de execução em *background*, o que é feito por meio da interface da IDE de desenvolvimento, abstraindo do programador como esta solicitação é de fato processada, e sempre que julgar necessário o *iOS* concederá tempo de execução chamando uma função padrão do sistema que deve ser sobrescrita pelo programador.

Não é possível precisar a periodicidade com que estas chamadas serão realizadas, pode-se solicitar uma frequência com que elas irão ocorrer, mas o sistema operacional escalona isso de diversas maneiras, levando em conta fatores como a quantidade de processos solicitando tempo de processamento. Quando o *iOS* disponibiliza o processador para o aplicativo, ele realiza a chamada de uma função específica na qual colocamos as tarefas que devem ser executadas.

Para este projeto o fluxo a cada chamada é verificar se tem um usuário autenticado no sistema, se o paciente configurou o aplicativo para buscar as informações compartilhadas e se temos as permissões necessárias para ler os dados do *HealthKit*.

Caso passe por todas as verificações, são disparadas *querys* para cada uma das quarenta características que o aplicativo monitora de forma assíncrona, para garantir desempenho executando mais de uma tarefa por vez. As informações retornadas são armazenadas em um dicionário, e ao final de todas as chamadas, estes dados retornados são tratados e enviados para o servidor remoto de banco de dados para serem salvos.

As buscas por dados salvos no *HealthKit* são realizadas com um predicado de tempo, só são retornadas informações compartilhadas desde a última verificação do sistema até o momento da chamada atual. Sempre ao final de uma verificação a data e hora desta é salva para que a próxima chamada consiga saber a partir de quando precisa buscar novos dados.

Nesta implementação buscamos individualmente por dados, assim conseguimos compartilhar alguns dados mesmo que não existam todos os tipos de dados possíveis, como é possível obter dados de diversas fontes, pode ser que o usuário utilize um aplicativo para monitorar o peso e outro para monitorar os alimentos, e no momento de uma busca só tenham dados novos sobre o peso do paciente, então estes dados são tratados individualmente e não como uma única informação de quarenta atributos.

Ao realizar o envio para a tabela de dados provenientes do *HealthKit* para a infraestrutura na nuvem, a API gera uma chave única de identificação desta entrada, e salva o tipo de dado, como peso por exemplo, a data e hora que a informação foi inserida no *HealthKit* e o valor do dado.

Podemos ver na imagem acima, o formato como o *Firebase* salva os dados, como dito previamente ele utiliza um JSON. Dentro de um usuário temos uma chave

“healthkit” que armazena cada entrada proveniente do serviço. Este formato simula uma tabela no banco de dados relacional que se refere ao usuário através de uma chave estrangeira.

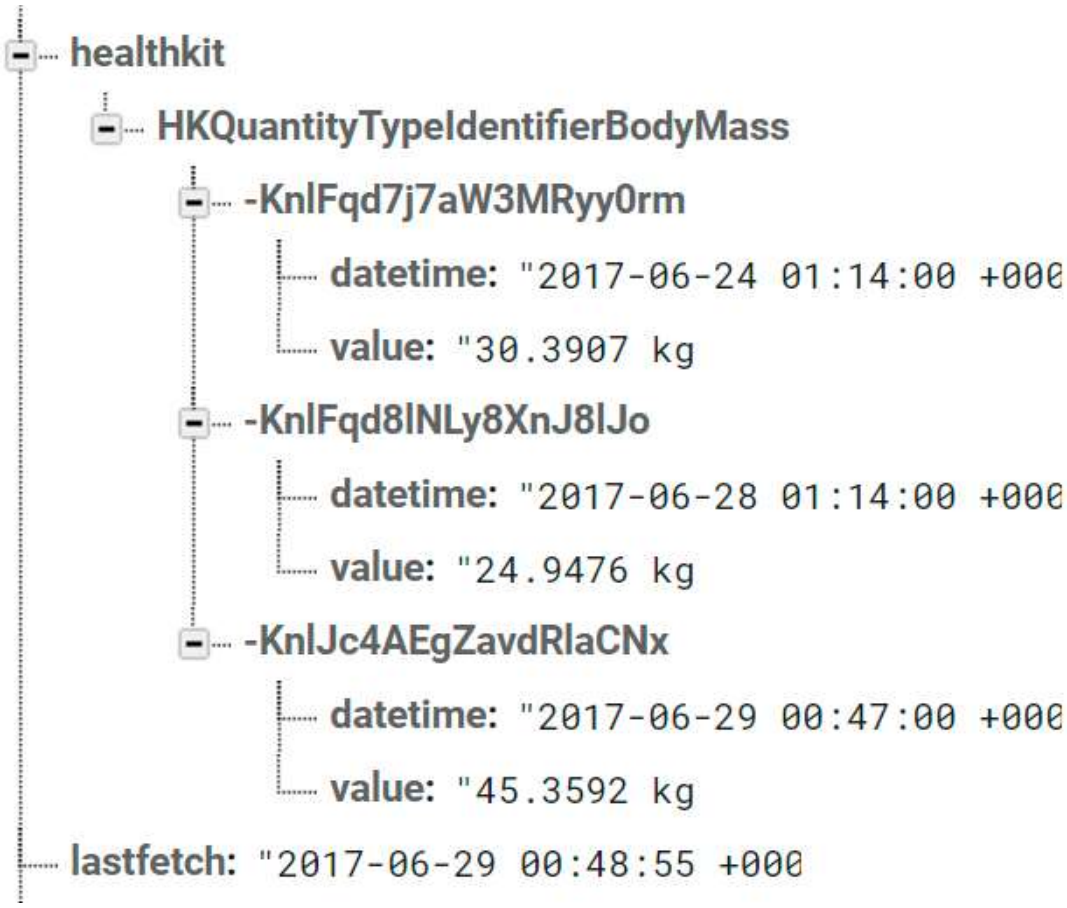

Figura 13 - Exemplo de como os dados são armazenados no *Firebase*

5.3 Histórico de Refeições

Ao acessar seu histórico, o usuário tem uma ideia geral de como anda sua dieta, conseguindo saber se tem cumprido regularmente às recomendações do nutricionista e tendo um panorama do que vem consumindo. Os dados são exibidos em ordem decrescente de data e hora, ou seja, primeiro os mais recentes e por último os mais antigos.

Assim os pacientes conseguem eles mesmos realizar o monitoramento da sua dieta, não deixando essa tarefa apenas a cargo do profissional de nutrição que o acompanha.

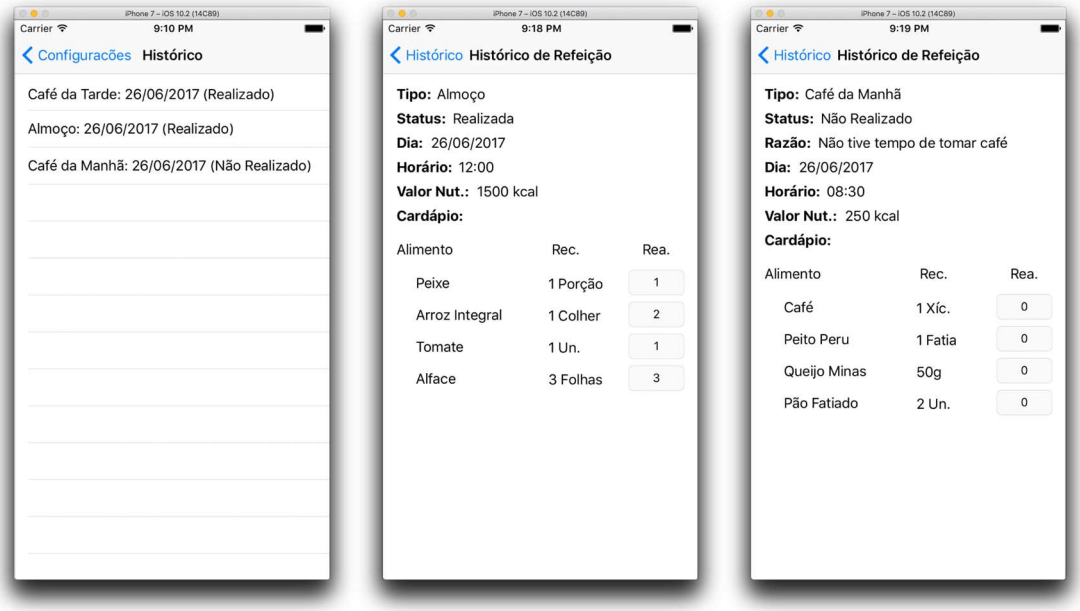

Figura 14 - Fluxo das telas do histórico

6 CONCLUSÃO

Após o desenvolvimento deste trabalho, foi possível aferir que os objetivos e requisitos especificados inicialmente foram alcançados, obtendo como produto um aplicativo para *iOS* capaz de monitorar a dieta de pacientes de nutrição e compartilhar os dados dele com o profissional responsável por realizar seu acompanhamento e definir seu tratamento.

Atendendo aos objetivos descritos é possível oferecer ao sistema de *homecare* em que este trabalho busca se inserir, a ferramenta proposta para facilitar a aquisição de dados para ele, agregando maior poder de decisão para o sistema e para o profissional, com informações mais precisas e coletadas instantaneamente ao momento em que são geradas.

Além disso, podemos promover um acréscimo de qualidade no contexto nutricional dos pacientes que utilizarem o aplicativo, já que estes terão uma dieta equilibrada e com os nutrientes adequados para as suas necessidades, já que cada refeição é prescrita por um profissional qualificado.

As informações disponibilizadas aos nutricionistas através do sistema de *homecare* também afetam muito positivamente o estado nutricional dos pacientes, pois o acesso à esta maior quantidade de dados sobre o usuário do aplicativo, tornará os diagnósticos e os tratamentos dos profissionais de nutrição muito mais assertivos.

Isto proporcionará cada vez mais resultados mais precisos e ações mais rápidas, impactando diretamente na saúde e qualidade de vida dos pacientes. Essa grande quantidade de dados também torna possível a mineração destes dados, deixando o sistema de *homecare* apto a extrair informações e padrões nutricionais que poderão servir de base para avanços na área.

Com o desenvolvimento deste trabalho, foi possível entender o funcionamento de serviços de infraestrutura como o *Firebase* que além de ficar responsável pela configuração e manutenção de servidores, abstrai do programador a comunicação com ele, oferecendo uma API moderna e completa para o acesso aos dados armazenados e ao sistema de autenticação.

Também foi possível verificar a evolução da plataforma da *Apple*, que além de construir uma linguagem mais moderna, o *Swift*, para o desenvolvimento com

características que além de deixar o código eficiente o torna extremamente legível, constrói ferramentas de desenvolvimento que facilitam bastante o trabalho do programador em criar interfaces para o projeto e trabalhar com as interações do usuário nela.

Buscar dados compartilhados de outros aplicativos é outro ponto de destaque, que poderá dar ainda mais informações para nutricionistas se basearem nos diagnósticos e tratamentos. Com a crescente quantidade de dispositivos vestíveis sendo produzidas, como relógios inteligentes por exemplo, cada vez mais novas fontes de dados nutricionais e de saúde estarão entrando no mercado, e a aquisição dos dados compartilhados por eles será cada vez mais completa e mais precisa, tornando a ferramenta desenvolvida neste trabalho cada vez mais útil para o sistema que a utilizará.

Os resultados obtidos aqui, são sempre disponibilizados para o sistema de *homecare* do qual faz parte, e este é responsável por tratar e realizar as operações que julgar necessário para entregar aos profissionais de nutrição. Este sistema é a forma que os nutricionistas têm de receber tudo que foi coletado do seu paciente, seja utilizando o aplicativo desenvolvido aqui ou outras ferramentas que também façam parte do sistema de *homecare*.

6.1 Trabalhos futuros

O aplicativo desenvolvido neste trabalho já é uma ferramenta funcional para a utilização dos pacientes, porém novas funcionalidades poderão ser implementadas afim de oferecer ainda mais informação relevante para os nutricionistas ou deixar o aplicativo mais completo.

Uma opção de melhoria seria modelar a informação que vem do sistema de *homecare* com dados extras que se enquadrem no modelo de armazenamento do *HealthKit* para que além de consumir os dados do sistema, este aplicativo possa colaborar com desenvolvedores e pesquisadores que estejam trabalhando em projetos da área, assim toda informação gerada por esse aplicativo também ficaria disponível a outros aplicativos instalados, caso o usuário assim deseje.

Outra possibilidade de trabalho futuro é a criação de um sistema de notificações, no qual os nutricionistas possam interagir com os pacientes enviando mensagens que seriam notificadas no aparelho do usuário. Como os profissionais estarão acompanhando os dados de cada paciente conforme eles são inseridos no sistema, pequenas correções de rotina ou solicitações para agendamento de consultas podem ser enviadas através de notificações, oferecendo um meio dos profissionais agirem de forma mais rápida quando julgarem necessário.

REFERÊNCIAS

WARDLAW, Gordon; SMITH, Anne. **Nutrição contemporânea**, 8ª Edição, AMGH Editora Ltda., 2013.

PINHEIRO, Anelise Rízzolo de Oliveira; FREITAS, Sérgio Fernando Torres de; CORSO, Arlete Catarina Tittoni. **Uma abordagem epidemiológica da obesidade. Revista de Nutrição**, Campinas, v. 17, n. 4, p. 523-533, 2004. Disponível em: <http://www.scielo.br/scielo.php?script=sci_abstract&pid=S1415-52732004000400012&lng=pt&nrm=iso&tlng=pt>. Acesso em: 19 de junho de 2017.

BASTIANI, Ederson; SOARES, Karlise; LIBRELOTTO, Giovani Rubert. **Uma abordagem para monitoramento de pacientes com alzheimer em ambientes homecare pervasivos**, Workshop de Informática Médica, WIM ’12, p. 1–10, Curitiba, Paraná, Brasil, 2012. Disponível em: < http://www.lbd.dcc.ufmg.br/colecoes/wim/2012/0030.pdf >. Acesso em: 20 de junho de 2017.

APPLE. **O que é o iOS?**. 2017. Disponível em <http://www.apple.com/br/ios/what-is/>. Acesso em: 23 de abril de 2017.

ROCHA, Nielson de Jesus Lima; NETO, Pedro Brandão. **Comparativo entre sistemas operacionais para *Smatphones***, Disciplina de Sistemas Operacionais, Unidade de Ensino Dom Bosco – UNDB, 2011. Disponível em: <http://www.academia.edu/download/34753921/PaperSO.docx>. Acesso em: 17 de julho de 2017.

APPLE. **Swift 3**. 2017. Disponível em <https://developer.apple.com/swift/>. Acesso em: 23 de abril de 2017.

KOCHAN, Stephen. **Programming in Objective-C**, 3ª Edição, Pearson Education, 2011.

WELLS, Garrett. **The Future of iOS Development: Evaluating the Swift Programming Language**, CMC Senior Theses. Paper 1179, 2015. Disponível em: < http://scholarship.claremont.edu/cmc_theses/1179>. Acesso em: 18 de julho de 2017.

GOOGLE. **Firebase**. 2017. Disponível em <https://firebase.google.com>. Acesso em: 30 de abril de 2017.

CHENG, Fu. **Build Mobile Apps with Ionic 2 and Firebase: Hybrid Mobile App Development**, 1ª Edição, Apress, 2017.

VENANCIO, Sonia Isoyama; LEVY, Renata Bertazzi; SALDIVA, Silvia Regina Dias Médici; MONDINI, Lenise; STEFANINI, Maria Lúcia Rosa. **Sistema de vigilância alimentar e nutricional no Estado de São Paulo, Brasil: experiência da implementação e avaliação do estado nutricional de crianças**. Rev. Bras. Saúde Mater. Infant., Recife, v.7, n.2, p.213-220, 2007 . Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S1519-38292007000200012&lng=en&nrm=iso>. Acesso em: 06 de maio de 2017.

APPLE. **HealthKit - Apple Developer**. 2017. Disponível em < https://developer.apple.com/healthkit/ >. Acessado em: 10 de maio de 2017.

APPLE. **ResearchKit**. 2017. Disponível em <http://researchkit.org/>. Acessado em: 10 de maio de 2017.

JARDINE, Jennifer; FISHER, Jonathan; CARRICK, Benjamin. **Apple's ResearchKit: smart data collection for the smartphone era?**, Journal of the Royal Society of Medicine; 2015, Vol. 108(8) 294–296 Disponível em: <http://journals.sagepub.com/doi/pdf/10.1177/0141076815600673>. Acesso em: 17 de julho de 2017.

ROCHA, Ulysses; OLIVEIRA, Davi; CARVALHO, Larissa; MOREIRA, Leonardo; VIANA, Windson. **Heart Wars: um exergame de batalha controlado por dispositivos vestíveis**, Instituto Universidade Virtual, Universidade Federal do Ceará, Fortaleza, Ceará, Brasil, 2015. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/wim/2015/022.pdf>. Acesso em: 17 de julho de 2017.

Universidade Federal de Santa Maria
Centro de Tecnologia
Curso de Ciência da Computação

A Comissão Examinadora, abaixo assinada,
aprova o Trabalho de Graduação

ESPECIFICAÇÃO E DESENVOLVIMENTO DE UM APLICATIVO IOS PARA MONITORAMENTO DE PACIENTES DE NUTRIÇÃO

elaborado por
Evandro Bianquin Machado

como requisito parcial para obtenção do grau de
Bacharel em Ciência da Computação

COMISSÃO EXAMINADORA:

Giovani Rubert Librelotto, Dr.
(Presidente/Orientador)

Lisandra Manzoni Fontoura, Drª. (UFSM)

Giliane Bernardi, Drª. (UFSM)

Santa Maria, 05 de Julho de 2017.