CENTRO PAULA SOUZA

FACULDADE DE TECNOLOGIA DE FRANCA
"Dr. THOMAZ NOVELINO"

TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

JOÃO PAULO FERNANDES RODRIGUES

A tecnologia digital de informação e comunicação para melhoria da qualidade de vida de pessoas diabéticas.

Trabalho de Graduação apresentado à Faculdade de Tecnologia de Franca - "Dr. Thomaz Novelino", como parte dos requisitos obrigatórios para obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador: Prof. Me. Claudio Eduardo Paiva

FRANCA/SP
2021

A tecnologia digital de informação e comunicação para melhoria da qualidade de vida de pessoas diabéticas.

João Paulo Fernandes Rodrigues

Resumo

O sistema multiplataformas tem como objetivo auxiliar no tratamento da doença diabetes mellitus, assumido como principal função verificar o índice glicêmico e efetuar suas devidas correções quando necessário. o sistema proporciona uma agenda com informações nutricionais de alimentos para possíveis cálculos de correções. Caso solicitado o sistema se encarregará de exibir um relatório contendo a média glicêmica, média do total de insulina aplicada durante o período selecionado, o maior e menor índice glicêmico registrado num intervalo de até 90 dias. O sistema permitirá ao usuário fazer eventuais anotações das atividades físicas e alimentações que foram realizadas durante o dia, dessa forma proporcionando um melhor monitoramento da doença evitando problemas futuros.

ATÉ 250 PALAVRAS.
Palavras-chave: Android, Controle, Diabetes, Insulina, Monitoramento.

Abstract

The android system aims to assist in the treatment of diabetes mellitus type one and type two, assumed as the main function to check the glycemic index and the amount of insulin applied by the patient during the day, in addition the system provides an agenda with nutritional information of food for possible correction calculations. As requested, the system will be in charge of displaying a report containing the glycemic average, average of the total insulin applied during the selected period, the highest and lowest glycemic index recorded in an interval of up to 90 days. The system will allow the user to make any notes of the physical activities and meals that were performed, thus providing a better monitoring of the disease, avoiding future problems.

Keywords: *Android, Control, Diabetes, Insulin, Monitoring*

1 Introdução

A função da insulina é promover a entrada de glicose para as células do organismo de tal maneira que ela possa ser aproveitada para as diversas atividades celulares. A falta de insulina ou um defeito na sua ação promove acúmulos de glicose no sangue, esse acumulo é chamado de hiperglicemia. É de extrema importância tratar da hiperglicemia, tendo em vista que está associada a lesões da

microcirculação, o que ocasiona um mal funcionamento em diversos órgãos como rins, olhos, os nervos e o coração.

> Atualmente, aproximadamente 415 milhões de adultos apresentam Diabetes Mellitus (DM) em todo o mundo e 318 milhões de adultos possuem intolerância à glicose, com risco elevado de desenvolver a doença no futuro.
>
> O Brasil é o quarto país com maiores taxas de DM na população adulta, com um total de 14,3 (12,9-15,8) milhões de pessoas de 20 a 79 anos com DM, com um gasto anual estimado de pelo menos US$ 21,8 bilhões (BRASil, 2017, p.09).

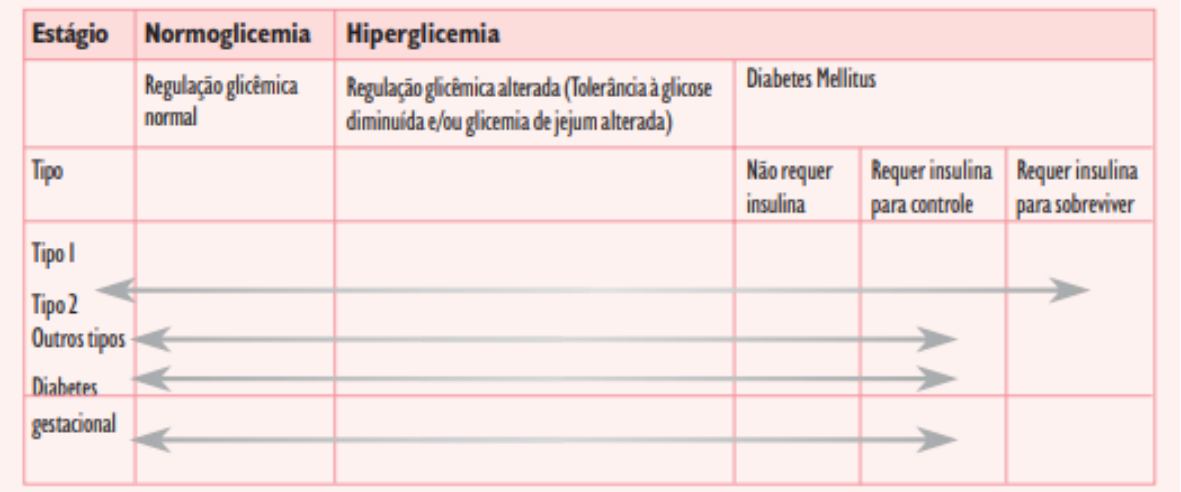
Os tipos de diabetes mais frequentes são o diabetes tipo 1 que corresponde a 10% do total dos casos, e o diabetes tipo 2 que corresponde a 90% dos casos.

O diabetes tipo 1, também conhecida como diabetes juvenil indica a deficiência absoluta do pâncreas na produção de insulina, nesse tipo especifico de diabetes o uso injetável de insulina passa a ser obrigatório e de extrema importância para que se possa prevenir cetoacidose, coma e morte.

O diabetes tipo 2 consiste no mal funcionamento do pâncreas que gera uma deficiência relativa a insulina. O diabetes tipo 2 em alguns casos é necessário o uso de insulina, porém ao contrário da diabetes tipo 1 o uso de insulina não visa evitar a cetoacidose, mais sim obter o controle de hiperglicemia, como pode-se ver na Figura 1. A cetoacidose nesse caso é rara, e quando presente geralmente está acompanhada de alguma infecção, trauma ou um estresse muito grave.

> Outro tipo de diabetes encontrado com maior frequência e cuja etiologia ainda não está esclarecida é o diabetes gestacional, que, em geral, é um estágio pré-clínico de diabetes, detectado no rastreamento pré-natal. Outros tipos específicos de diabetes menos frequentes podem resultar de defeitos genéticos da função das células beta, defeitos genéticos da ação da insulina, doenças do pâncreas exócrino, endocrinopatias, efeito colateral de medicamentos, infecções e outras síndromes genéticas associadas ao diabetes (BRASIL, 2006, p.12).

Figura 1 – Requisitos para uso de insulina



Fonte: Brasília (2016)

É possível identificar pessoas portadoras de diabetes mellitus tipo 1 pelos seguintes sintomas: vontade de urinar com muita frequência, sede excessiva, fome em excesso, perda involuntária de peso, cansaço frequente, sonolência, coceira em todo corpo, infecções frequentes, irritabilidade e mudanças de humor repentinas.

Já as pessoas com diabetes tipo 2 geralmente apresentam os seguintes sintomas: aumento de sede, boca constantemente seca, vontade de urinar com muita frequência, cansaço frequente, visão turva ou embaçadas, feridas com a cicatrização muito lentas e infecções frequentes.

Com relação ao tratamento do diabetes devemos elencar como sendo essencial a alimentação e a pratica de exercícios físicos que ajudam na distribuição de glicose entre as células. A alimentação a quantidade energética ingerida deve ser adequada à atividade física e ser fracionada em 5 a 6 refeições/lanches diários. Para tanto, os pacientes devem ser encorajados a comer alimentos ricos em fibras, como frutas, verduras, legumes, feijões e cereais integrais.

Com relação aos diabéticos tipo 1, deve obrigatoriamente ser feita a aplicação de insulina para determinada quantidade de carboidratos, que será definido pelo médico.

As correções no nível de glicemia devem ser feitas com base a exames de sangue, onde são relatados o exato valor no momento do exame, a base para o cálculo de correção é definida obrigatoriamente pelo médico.

O diabetes apresenta altos índices de internações e morbi -mortalidade, o que ocasiona uma perda importante na qualidade de vida do portador de tal doença, isso ocorre devido a erros ao fazer as correções através de insulina e mal

monitoramento da doença, devemos elencar também que pode ser um dos motivos algumas infecções encontradas nos pacientes. O diabetes é uma das principais causas de mortalidade juntamente com insuficiência renal, amputações de membros inferiores, cegueira e doenças cardiovasculares.

> Estima-se que uma em cada 12 mortes em adultos no mundo possa ser atribuída ao DM, totalizando, aproximadamente, cinco milhões de casos ao ano, o que equivale a uma morte a cada seis segundos. A proporção de óbitos é ligeiramente maior em mulheres do que em homens. O gasto com DM, na maioria dos países, varia entre 5% e 20% das despesas globais em saúde (Brasília, 2017, p.09).

Como obter uma melhor qualidade de vida para diabéticos, pelo controle da quantidade de insulina a ser injetada no sangue, através de um aplicativo multiplataforma desenvolvido com Xamarin para facilitar a definição de horários e cálculos rápidos de alimentação a ser consumida, possibilitando o registro dos dados de medições para monitoramento do tratamento?

O aplicativo tem por objetivo evitar erros de cálculos referentes a insulina a ser aplicada, alertar o usuário de tomar os remédios nos devidos horários, fazer exames de correções, registrar observações referentes aos horários que forem feitos os exames e atividades físicas, lembrar sobre as próximas consultas e exames marcados, além de auxiliar o médico a obter um cronograma e monitoramento mais detalhado, dessa forma proporcionando uma melhor qualidade de vida para pessoas portadoras do diabetes mellitus.

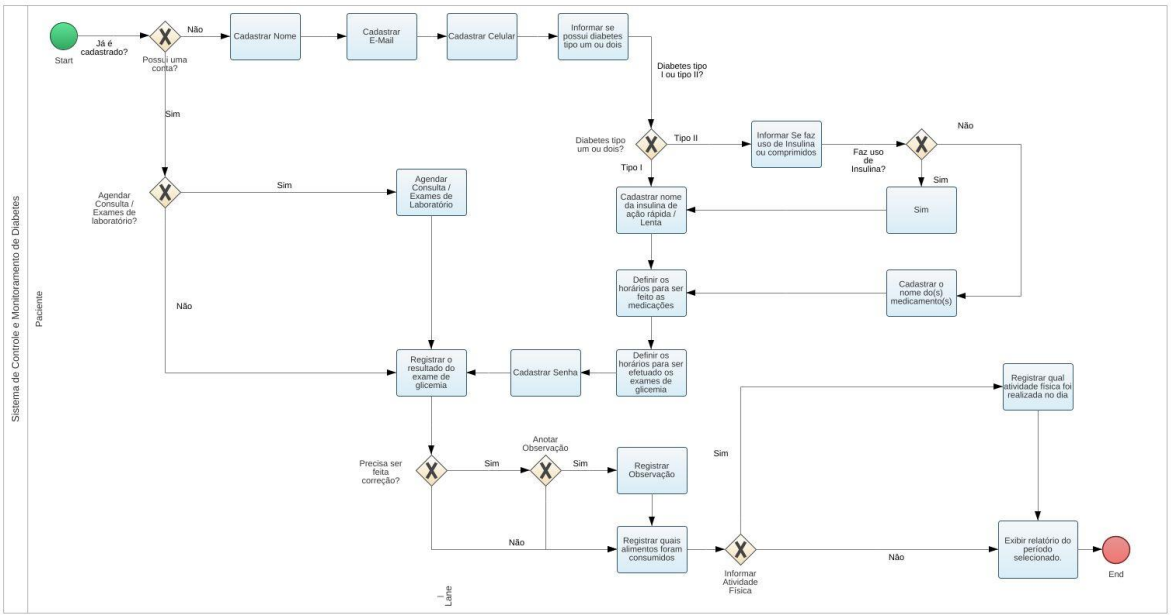
2 Levantamento de Requisitos

2.1 Elicitação e especificação dos Requisitos

Este projeto foi desenvolvido a partir de técnicas de entrevistas com pessoas portadoras da doença “diabetes mellitus”, onde foi utilizado um questionário pré-elaborado com perguntas referentes ao controle da doença, também foi utilizado a técnica de etnografia, que durante um período pré-definido foi acompanhado o tratamento dessa doença.

2.2 BPMN

Figura 2: BPMN



2.3 Requisitos Funcionais

Usar modelo estudado nas aulas de engenharia de software.

Quadro 1 – Requisitos Funcionais do sistema

RF001-Cadastro do exame de glicemia.	Categoria: ( ) Oculto (X)Evidente	Prioridade: (X) Altíssima ( ) Alta ( ) Média ( ) Baixa
Descrição: O sistema deve registrar o dia, horário e o valor obtido pelo exame de glicemia.		
RF002-Cadastro de medicamentos	Categoria: ( ) Oculto (X)Evidente	Prioridade: ( ) Altíssima (X) Alta ( ) Média ( ) Baixa
Descrição: O sistema deve registrar os medicamentos utilizados pelo usuário.		
RF003-Cadastro de dosagem para correção alimentar.	Categoria: ( ) Oculto (X)Evidente	Prioridade: ( ) Altíssima ( ) Alta (X) Média ( ) Baixa
Descrição: O sistema deve registrar a quantidade de insulina referente a determinada quantidade de carboidratos. Exemplo: 1 unidade para cada 15 gramas de carboidrato.		
RF004- Calcular média glicêmica	Categoria: (X) Oculto ( ) Evidente	Prioridade: ( ) Altíssima (X) Alta ( ) Média

		( ) Baixa

Descrição: Caso seja solicitado o sistema deve fazer a média glicêmica de determinados períodos definidos pelo usuário.
Fórmula para média glicêmica:
  Para cada exame registrado será somado o valor “1” em uma variável “X”.
  Em uma segunda variável “Y” será somado o valor obtido através dos registros dos exames de glicemias do usuário.
  Deste modo se o usuário registrar o valor do seu destro em 3 horários diferentes, com os seguintes valore: 250; 150; 80 o sistema deverá soma-los e dividi-los pôr 3 chegando ao resultado de 160.
  Média = Y/X.		

RF005-Calculo de correção de glicemia.	Categoria: (X) Oculto ( ) Evidente	Prioridade: ( ) Altíssima (X) Alta ( ) Média ( ) Baixa
Descrição: Com base nos dados obtidos através do cadastro do usuário, o sistema deverá calcular a quantidade de medicamento a ser utilizado e fazer a sugestão.		
RF006- Cadastro de Alimento.	Categoria: ( ) Oculto (X) Evidente	Prioridade: ( ) Altíssima ( ) Alta ( ) Média (X) Baixa
Descrição: O usuário do sistema poderá inserir novos alimentos informando a sua quantidade de carboidratos. O sistema já deverá conter alimentos básicos inseridos como: Arroz; Feijão; Ovo; Macarrão; Alface; Tomate; Batata; laranja, entre outros.		
RF007- Calculo para dosagem de alimentos.	Categoria: (X) Oculto ( ) Evidente	Prioridade: ( ) Altíssima (X) Alta ( ) Média ( ) Baixa
Descrição: O usuário deverá informar os alimentos que se deseja calcular, em seguida deverá informar a quantidade referentes à cada alimento. Exemplo: • 1 unidade para cada 15 gramas de carboidratos.		
RF008- Sugerir a quantidade de medicamento a ser utilizado	Categoria: (X) Oculto ( ) Evidente	Prioridade: (X) Altíssima ( ) Alta ( ) Média ( ) Baixa

Descrição: Ao se obter os cálculos de correção glicêmica e correção alimentar, o sistema deverá somar os dois resultados e exibir para o usuário a quantidade recomendada de medicamento a ser utilizado, não poderá ser um número com casas decimais, deverá ser um número inteiro.		
RF009- Informar atividades físicas	Categoria: (X) Oculto ( ) Evidente	Prioridade: ( ) Altíssima ( ) Alta ( ) Média (X) Baixa
Descrição: O usuário poderá informar sobre suas atividades físicas diária, podendo definir quais atividades foram realizadas em um determinado período.		
RF010- Monitorar a média de medicamento consumido.	Categoria: (X) Oculto ( ) Evidente	Prioridade: ( ) Altíssima ( ) Alta (X) Média ( ) Baixa
Descrição: O sistema deverá assim que solicitado pelo usuário fazer o cálculo da média da quantidade de insulina aplicada por um determinado período. O usuário informa o valor que foi aplicado, o sistema armazena o valor em uma variável “X” e soma o valor 1 em uma segunda variável “Y” para cada registro. Média = X / Y.		
RF011- Cadastro Pessoal	Categoria: (X) Oculto ( ) Evidente	Prioridade: (X) Altíssima ( ) Alta ( ) Média ( ) Baixa
Descrição: O usuário deverá cadastrar obrigatoriamente (e-mail, senha, telefone, tipo de diabetes que possui, quantidade de insulina recomendada pelo médico), e poderá optar em cadastrar (quantidade de insulina para correção e alimentos).		
RF012- Alarme	Categoria: (X) Oculto ( ) Evidente	Prioridade: ( ) Altíssima ( ) Alta ( ) Média (X) Baixa
Descrição: O usuário do sistema poderá configurar o aplicativo para lembra-lo de fazer seus devidos exames de destro em horários pré-definidos, além de alertar caso haja algum evento agendado. O sistema deverá disparar o alarme nos horários definidos pelo usuário.		

RF013- Relatório	Categoria: (X) Oculto ( ) Evidente	Prioridade: ( ) Altíssima ( ) Alta (X) Média ( ) Baixa
Descrição: o usuário poderá verificar todas as sugestões feitas pelo aplicativo através de um relatório.		

2.4 Requisitos Não Funcionais

Usar modelo estudado nas aulas de engenharia de software.

Quadro 2 – Requisitos Não Funcionais do sistema

RNF001-Login	O usuário deverá estar logado para utilizar o sistema.	Tipo Segurança	( ) Desejável (X) Obrigatório	(X) Permanente ( ) Transitório
RNF002-Acessibilidade	Deve ser compatível para Android, IOS e Windows 10.	Tipo Compatibilidade	( ) Desejável (X) Obrigatório	( ) Permanente (X) Transitório
RNF003- Facilidade de uso.	O sistema deverá ser de fácil manuseio, tendo uma taxa de até 3 erros por dia no início de sua utilização.	Tipo Usabilidade	(X) Desejável ( ) Obrigatório	( ) Permanente (X) Transitório
RNF004- Tempo estimado para acesso	O software deve conectar imediatamente o usuário, será estimando uma média de até 25 segundos de espera.	Tipo Desempenho	(X) Desejável () Obrigatório	( ) Permanente (X) Transitório
RNF005- Tempo estimado para realizar cálculos	O software deverá apresentar os resultados dos cálculos em um tempo estimado de até 10 segundos.	Tipo Desempenho	(X) Desejável ( ) Obrigatório	( ) Permanente (X) Transitório
RNF006- Disponibilidade de uso	O sistema deverá estar disponível durante as 24h do dia.	Tipo Disponibilidade	( ) Desejável (X) Obrigatório	(X) Permanente ( ) Transitório
RNF007- Identificação do usuário	O usuário deverá identificar qual tipo de diabetes que ele possui, se é tipo 1 ou tipo 2	Tipo Usabilidade	( ) Desejável (X) Obrigatório	(X) Permanente ( ) Transitório

RNF008- Entrega de relatório	O Relatório deverá ser entregue assim que solicitado, sendo estimado um tempo de até 20 segundos de espera.	Tipo Usabilidade	( ) Desejável (X) Obrigatório	( ) Permanente (X) Transitório
RNF09- Segurança	O usuário apenas poderá ver seus próprios dados, não será permitido ver dados de terceiros no aplicativo.	Tipo Segurança	( ) Desejável (X) Obrigatório	(X) Permanente ( ) Transitório
RNF010- Armazenamento de dados	O sistema deverá comunicar com o banco de dados SQLite.	Tipo Compatibilidade	( ) Desejável (X) Obrigatório	(X) Permanente ( ) Transitório
RNF011- Linguagem de programação	O sistema deverá ser desenvolvido utilizando a linguagem C#.	Tipo Compatibilidade	( ) Desejável (X) Obrigatório	(X) Permanente ( ) Transitório

Quadro 3 – Matriz de rastreabilidade.

	RF 1	RF 2	RF 3	RF 4	RF 5	RF 6	RF 7	RF 8	RF 9	RF1 0	RF1 1	RF1 2	RF1 3
RNF1	X	X	X	X	X	X	X	X	X	X		X	X
RNF2	X	X	X	X	X	X	X	X	X	X	X	X	X
RNF3	X	X	X	X	X	X	X	X	X	X	X	X	X
RNF4	X	X	X			X			X	X	X		
RNF5				X	X		X						
RNF6	X	X	X	X	X	X	X	X	X	X	X	X	X
RNF7											X		
RNF8													X
RNF9	X	X	X	X	X	X	X	X	X	X		X	X
RNF1 0	X	X	X	X	X	X	X	X	X	X	X	X	X

RNF11	X	X	X	X	X	X	X	X	X	X	X	X	X

2.5 Casos de Uso

Índice de casos de uso e Diagrama de casos de uso

UC001 – Gerenciar dados.

UC002 – Informar o resultado do exame de glicemia.

UC003 – Gerenciar alimentos e dosagens.

UC004 – Exibir sugestões.

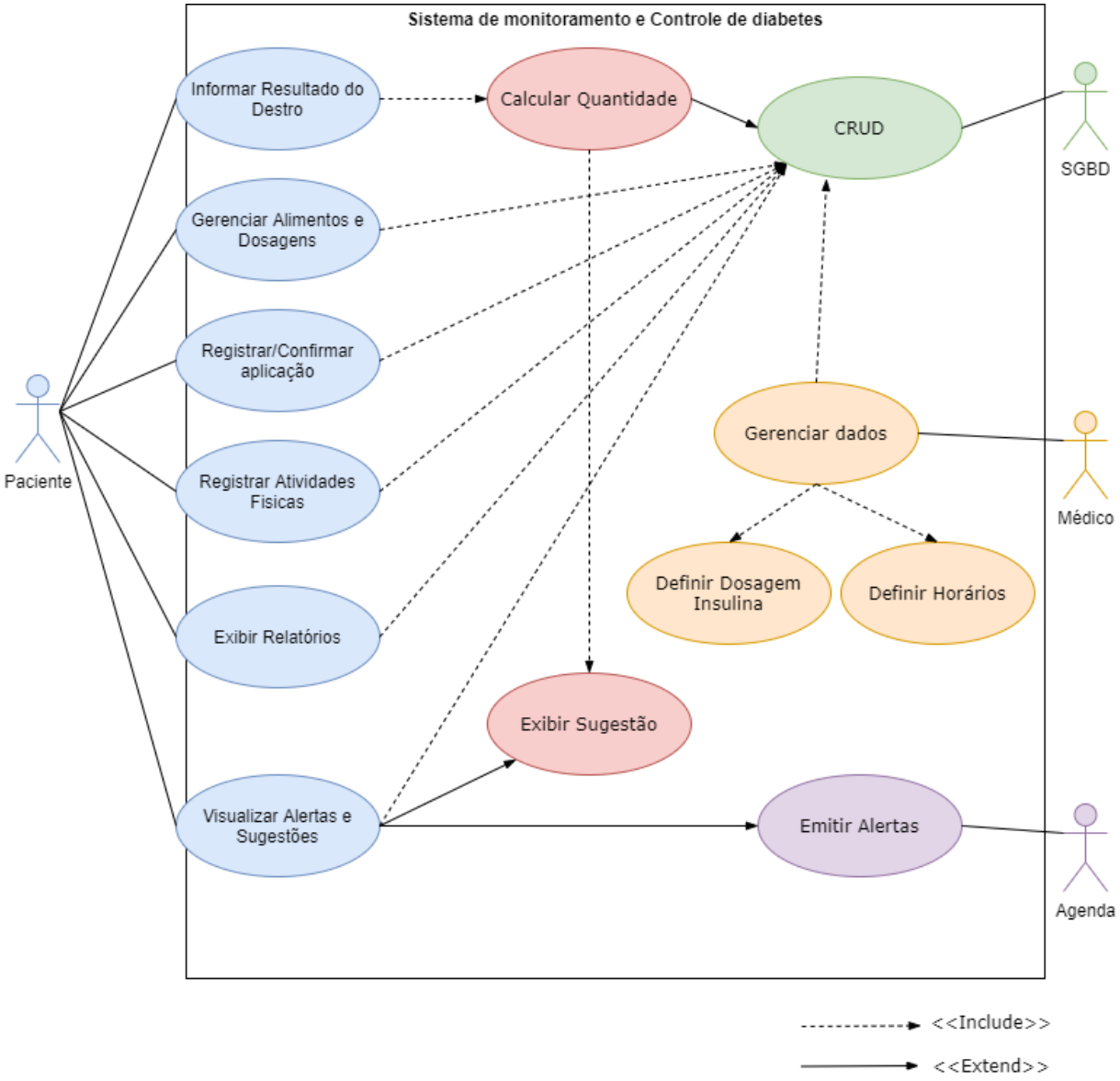
UC005 – Registrar/Confirmar aplicação.

UC006 – Registrar atividades físicas.

UC007 – Exibir relatório.

UC008 – Emitir Alertas.

Figura 3: Diagrama de Casos de Uso



Especificação de cada um dos casos de uso

Quadro 5 – Use Case Cadastrar Usuários

Caso de Uso – Gerenciar dados.	
ID	UC 001
Descrição	Este caso de uso tem como objetivo gerenciar os horários e dosagens de cada medicação, e agendamento de consultas e exames feitos em laboratórios.
Ator Primário	Paciente
Pré-condição	É obrigatório que os dados sejam preenchidos com base na receita médica mais recente.
Cenário Principal	1. O use case inicia após o usuário selecionar a opção de agendamento. 2. O sistema carrega um formulário de cadastro de horários.

	3. O paciente escolhe entre a opção de agendar um evento ou atualizar alertas. 4. O paciente defini os horários e dias quando necessário. 5. O sistema valida as definições. 6. O Sistema armazena os dados. 7. O sistema encerra o caso de uso.
Pós-condição	Nenhuma
Cenário Alternativo	4a – Horário não informado. 4a.1 O sistema informa ao usuário que este campo é obrigatório e deve ser preenchido. 4a.2 Retorna ao passo 3 do cenário principal. 4b – Horário definido não existe. 4b. 1 O Sistema informa ao usuário que o horário a ser definido não existe. 4b. 2 Retorna ao passo 3 do cenário principal. 4c – Evento sem dia marcado. 4c. 1 O sistema informa ao usuário que para se cadastrar um evento deve-se definir um dia e um horário. 4c. 2 Retorna ao passo 3 do cenário principal.

Caso de Uso – Informar o resultado do exame de glicemia	
ID	UC 002
Descrição	Este caso de uso tem por objetivo registrar o resultado obtido através do exame de glicemia.
Ator Primário	Paciente
Pré-condição	Deverá ser cadastrado no dia e no horário definidos pelo usuário.
Cenário Principal	1. O use case inicia ao selecionar a opção cadastro de exames. 2. O sistema carrega o formulário de cadastro. 3. O usuário informa o dia, o horário e o resultado do exame de glicemia. 4. O sistema valida os campos. 5. O sistema armazena os dados e informa ao usuário que os dados foram salvos. 6. O sistema encerra o caso de uso.
Pós-condição	Nenhuma
Cenário Alternativo	4a – Dia e horário não preenchidos ou inválidos 4a.1 O sistema informa que o atributo dia e horário são obrigatórios e que deveram ser preenchidos com valores válidos. 4a. 2 Retorna ao passo 3 no cenário principal. 4b – Exame de destro inválido ou não preenchido. 4b. 1 O sistema informa que o exame de destro deve ser um valor válido, e que este campo é obrigatório. 4b. 2 Retorna ao passo 3 no cenário principal.

Caso de Uso – Gerenciar alimentos e dosagens**.**	
ID	UC 003
Descrição	Este caso de uso tem por objetivo gerenciar a quantidade de insulina referentes aos carboidratos de alimentos cadastrados no sistema.
Ator Primário	Paciente
Pré-condição	Nenhuma.
Cenário Principal	1. O use case inicia após o usuário selecionar a opção de contagem de carboidratos. 2. O sistema carrega o formulário de seleção de alimentos. 3. O usuário seleciona/cadastra um alimento. 4. O sistema abre um formulário de adição de alimentos. 5. O usuário preenche os campos destacando quais alimentos foram consumidos e sua respectiva quantidade. 6. O sistema valida os campos. 7. O sistema calcula a quantidade total de carboidratos informadas armazena os dados obtidos. 8. O sistema verifica se o paciente faz uso de insulina injetável, caso faça o valor obtido dos carboidratos é convertido para unidades do medicamento. 9. O sistema informa ao usuário que foi concluído essa etapa com sucesso. 10. O sistema encerra o caso de uso.
Pós-condição	Nenhuma
Cenário Alternativo	6a – Alimento não cadastrado. 6a.1 O sistema informa ao usuário que o alimento informado não está cadastrado no banco de dados. 6a.2 Retorna para o passo 3 do cenário atual.

Caso de Uso – Exibir sugestões.	
ID	UC 004
Descrição	Este caso de uso tem por objetivo sugerir a quantidade correta de da medicação a ser utilizada.
Ator Primário	Paciente
Pré-condição	Os dados serão apresentados somente após ser executado pelo menos o UC002.
Cenário Principal	1. O use case inicia após o usuário informar o resultado do exame de destro e/ou os alimentos consumidos. 2. O sistema valida os campos obrigatórios

	3. O sistema deverá calcular a quantidade de insulina/comprimido, tendo como referência o resultado do destro e a quantidade de carboidratos ingeridos. 4. O sistema sugere a quantidade de medicamento.
Pós-condição	Nenhuma.
Cenário Alternativo	2a – Resultado do destro não informados 2a. 1 O sistema informa que o atributo resultado de destro é obrigatório. 2b. 2 Retorna ao passo 1 no cenário principal

Caso de Uso – Registrar/Confirmar aplicação	
ID	UC 005
Descrição	Este caso de uso tem por finalidade fazer o cadastro das quantidades das medicações feitas pelo usuário
Ator Primário	Paciente
Pré-condição	Deve ter sido apresentado a sugestão ao usuário.
Cenário Principal	1. O use case inicia após o sistema exibir a sugestão ao usuário. 1. O sistema abre um formulário de observações. 2. O usuário, anota a quantidade que foi aplicada caso seja diferente da sugestão. 3. O sistema carrega uma caixa de confirmação. 4. O usuário confirma a aplicação. 5. O sistema salva os dados. 6. O sistema encerra o caso de uso.
Pós-condição	Nenhuma
Cenário Alternativo	4a – Dados não confirmados. 4a.1 O sistema encerra o caso de uso e retorna uma tela de notificação.

Caso de Uso – Registrar atividades físicas.	
ID	UC 006
Descrição	Este caso de uso tem por objetivo monitorar as atividades físicas realizadas pelo usuário.
Ator Primário	Paciente
Pré-condição	Nenhuma.
Cenário Principal	1. O use case inicia após o sistema alertar o usuário de suas atividades físicas. 2. O sistema carrega o formulário de cadastro. 3. O usuário informa qual atividade física foi realizada (caminhada, corrida, futebol, luta, etc.) e também informa o período que foi realizada a atividade.

	4. O sistema valida os campos obrigatórios. 5. O sistema salva os dados no banco de dados e informa ao usuário que os dados foram salvos. 6. O sistema encerra o caso de uso.
Pós-condição	Nenhuma
Cenário Alternativo	1a – Usuário ignora o alerta. 1a – O sistema encerra o caso de uso. 4a – Campos Atividade realizada não preenchido. 4a.1 O sistema informa ao usuário que este campo é obrigatório e deve ser preenchido. 4a.2 Retorna ao passo 3 do cenário principal.

Caso de Uso – Exibir relatório	
ID	UC 007
Descrição	Este caso de uso tem por objetivo permitir que o usuário possa monitorar a doença.
Ator Primário	Paciente
Pré-condição	Deverá conter pelo menos um dado cadastrado.
Cenário Principal	1. O use case inicia após o usuário selecionar a opção relatório. 2. O sistema valida os dados que deveram conter no relatório. 3. O sistema converte o relatório para formato PDF. 4. O sistema inicia o download do PDF contendo os dados do relatório. 5. O sistema encerra o caso de uso.
Pós-condição	Nenhuma
Cenário Alternativo	2a – Nenhum exame de destro registrado. 2a.1 O sistema informa ao usuário que é necessário ter pelo menos um exame de destro cadastrado para gerar o relatório 2a.2 Retorna ao passo 1 do cenário principal.

Caso de Uso – Emitir alertas	
ID	UC 008
Descrição	Este caso de uso tem por objetivo gerar alertas de horários de medicações e de eventos agendados.
Ator Primário	Paciente
Pré-condição	UC 001 tenha sido executado com sucesso.
Cenário Principal	1. O use case inicia assim que o sistema emitir uma mensagem de alerta ao usuário. 2. O usuário confirma a mensagem. 3. O sistema encerra o caso de uso.
Pós-condição	Nenhuma
Cenário Alternativo	2a – O usuário não confirma o alerta. 2a.1 O sistema informa retorna um novo alerta 15 minutos depois.

	2a.2 Encerra o caso de uso.

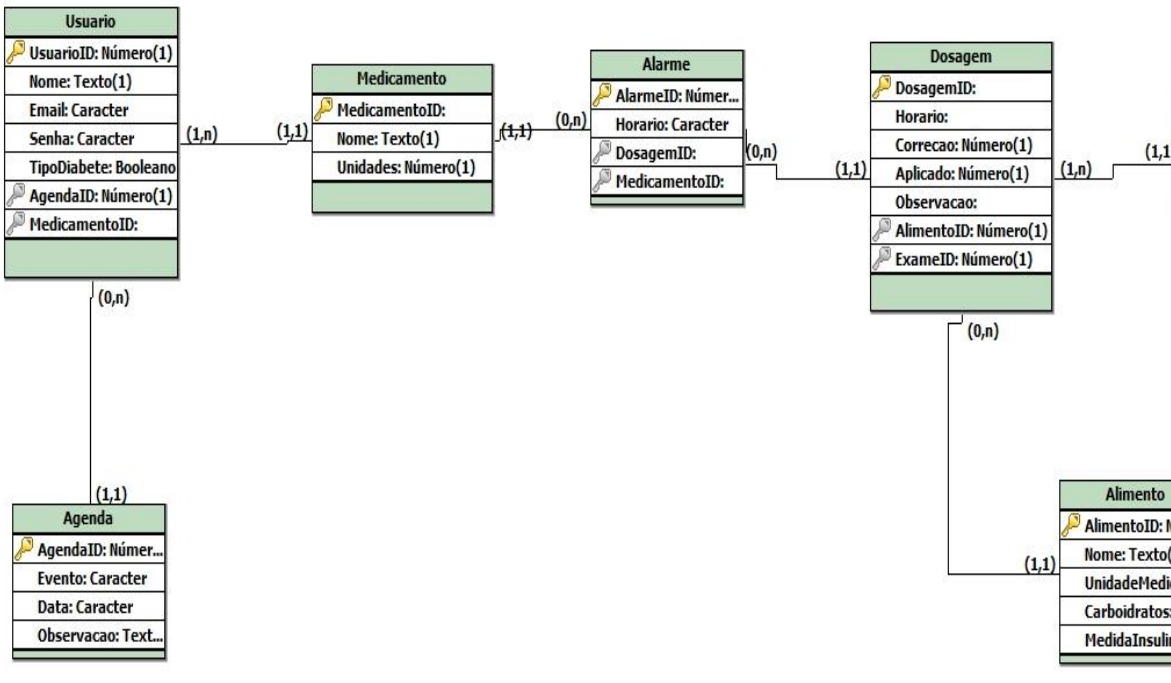
Quadro 6 – Matriz de rastreabilidade RF x UC

	RF 1	RF 2	RF 3	RF 4	RF 5	RF 6	RF 7	RF 8	RF 9	RF 10	RF 11	RF 12
UC1		X	X		X		X	X				X
UC2	X											
UC3			X			X	X	X				
UC4								X				
UC5			X		X		X	X	X			
UC6									X			
UC7												X
UC8										X		X

2.6 Diagrama Entidade-Relacionamento

Diagrama que representa a modelagem do banco de dados.

Figura 4: Diagrama Entidade-Relacionamento



3 Ferramentas e Métodos ou Desenvolvimento

3.1 Ferramentas

3.1.1 Implementação

A linguagem de programação escolhida para o desenvolvimento do projeto foi a C#, utilizando-se a IDE Visual Studio Community 2019, por facilitar os back-ups incrementaís feitos no projeto através da plataforma Git-Hub, além de propiciar a instalação completa do framework Xamarin-Forms na versão 5.0.0.2115.

A escolha do framework Xamarin-Forms se tornou necessária para distribuir a mesma linha de código para diversas plataformas diferentes, sendo elas o sistema operacional Windows 10, android e IOS, tendo o principal foco na plataforma android.

Foi utilizado o banco de dados SqLite na versão 1.8.116, que possibilita a gravação dos dados no próprio dispositivo do usuário, dessa forma não sendo necessário a utilização de internet para usufruir dos benefícios do sistema, sendo está a principal razão para o seu uso.

3.1.2 Línguagem de programação C#.

C# é uma linguagem orientada a objetos desenvolvida por Anders Hejlsberg para a Microsoft, fazendo parte da aplicação .NET. Em primeira instância a linguagem foi batizada como Cool no final do século XX, mais tarde após a virada do século a linguagem Cool foi renomeada para C#.

> Após o renome da linguagem, a Microsot submeteu-se à ECMA(European Computer Manufacturers Association), associação cujo objetivo é a padronização de sistemas de informação. Em 2001, a ECMA aprovou o C# e a linguagem recebeu a especificação ECMA-334. Mais tarde, em 2003, tornou-se padrão da ISSO, recebendo a especificação de ISSO/IEC 23270 (COIMBRA DE ARAÚJO, 2013).

O C# surgiu com o propósito de tornar menos rígido o desenvolvimento de aplicativos, a linguagem foi baseada nas linguagens de programação, Java, C e C++, juntando diversos recursos de cada linguagem citada, se tornando mais simples e flexível introduzindo diversos elementos como: expressões lambda, tipos primitivos com valores nulos, delegações e acessos diretos a memória.

> Em relação a linguagem C, o C# tem foco na compilação de soluções de alto nível, já a linguagem C é voltada para o desenvolvimento de baixo nível. Outras comparações, de acordo com MSDN (2012), o C#, simplifica de modo significativo a complexidade do C++ e introduz novos elementos não

> disponíveis no Java, tais como: tipos primitivos não nulo, delegações, expressões lambda e acesso direto à memória.
>
> MSDN (2012) ressalta que, o processo de compilação do C# é mais simples e mais flexível comparado ao C++ ou Java, pois elimina-se a necessidade de arquivos de cabeçalhos separados e não há a necessidade da declaração de métodos e tipos em uma ordem específica. Um arquivo contendo códigos de programação podem ser definidos quaisquer números de classes, estruturas, interfaces e eventos que forem necessários. (COIMBRA DE ARAÚJO, 2013).

3.1.3 Banco de dados SQLite.

Na prática o SQLite se assemelha à um Sistema de Gerenciamento de Banco de Dados, possibilitando fazer inserções, alterações e exclusões de dados, o SQLite é formado por um conjunto de bibliotecas escritas em C, com a finalidade de ser inserido em diversos programas escritos em diversas linguagens diferentes.

A principal diferença com outros bancos de dados, é que a manipulação dos dados pode ser feita sem que seja preciso acessar um SGBD (sistema de gerenciamento de banco de dados), ou seja, todas as instruções podem ser feitas através do código fonte, inclusive a criação do banco de dados e tabelas, sendo esse o principal motivo por ser implementada em aplicativos móveis (Android, IOS).

> Além disso, o Android oferece suporte completo ao banco, através de uma API com um rico conjunto de classes e métodos que abstraem as complexidades dos códigos SQL. Assim, não precisamos montas a cláusula SQL inteira para atualizar uma linha na tabela, ou ainda, para fazer uma pesquisa na mesma. O Android nos fornece um método, onde passando alguns parâmetros obtemos um apontador para os dados retornados, podendo navegar pelo resultado como se estivéssemos escolhendo uma folha em um arquivo. (CRIS BRITO, 2015).

3.1.3 Xamarin.Forms.

O Xamarin é uma plataforma de desenvolvimento utilizada para criar sistemas em diversas plataformas diferentes, o framework é uma extensão da plataforma de desenvolvimento .NET que é composta por diversas bibliotecas, ferramentas e linguagens de programação para desenvolver diversos aplicativos.

O Xamarin.Forms trata-se de uma estrutura do Xamarin, que tem por finalidade disponibilizar bibliotecas e ferramentas para o desenvolvimento de aplicativos móveis. O Xamarin.Forms é uma estrutura de código aberto da Microsoft, tendo seu principal objetivo a criação de aplicativos para Android, IOS e sistemas operacionais Windows através do .NET, através de uma única base de códigos compartilhados.

3.2 Métodos ou Desenvolvimento

Foi utilizado o patern MVVM (Model-View-Viewmodel) para o desenvolvimento do aplicativo, esse patern foi criado em 2005 por John Gossman, sendo ele um dos arquitetos do WPF e Silverlight da Microsoft. O partner se assemelha ao padrão MVC (Model-View-Controller), contudo o MVVM procura estabelecer uma separação mais apurada de responsabilidades em uma aplicação WPF e Silverlight.

Em termos gerais a View é responsável por definir o front-end do sistema, que faz uma interligação com a ViewModel através de um componente denominado de DataContext. A ViewModel tem como sua principal responsabilidade apresentar uma lógica de apresentação para a View, sendo assim fazendo a interligação entre a View e a Model. A Model tem a responsabilidade de encapsular a lógica e dados, são as classes que são utilizadas no desenvolvimento do sistema.

Figura 5 – View

```
1   <?xml version="1.0" encoding="utf-8" ?>
2   <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3                xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4                x:Class="TG.View.LoginPage">
5     <ContentPage.Content>
6       <StackLayout>
7         <Frame>
8           <Grid Padding="10" VerticalOptions="Center">
9             <Grid.RowDefinitions>
10              <RowDefinition Height="Auto"/>
11              <RowDefinition Height="*"/>
12            </Grid.RowDefinitions>
13
14            <StackLayout Padding="0" Grid.Row="0" VerticalOptions="Center">
15              <Image Source="AvatarLogin.png" WidthRequest="150" HeightRequest="150"/>
16
17              <Label Text="Login" TextColor="Blue" FontSize="30" FontAttributes="Bold" HorizontalOptions="Center"/>
18            </StackLayout>
19
20            <StackLayout Padding="0" Grid.Row="1">
21              <Entry x:Name="Login" Placeholder="Email do Usuário"/>
22              <Entry x:Name="Senha" Placeholder="Senha do Usuário" IsPassword="true"/>
23
24              <Button Text="Entrar" Clicked="EntrarAction"/>
25              <Button Text="Cadastrar novo usuário" BackgroundColor="Transparent" TextColor="DarkBlue" Command="{Binding ..
                 Cadastrar}"/>
26            </StackLayout>
27          </Grid>
28        </Frame>
29      </StackLayout>
30    </ContentPage.Content>
31  </ContentPage>
32
```

Fonte: Autor

A figura a cima mostra um breve exemplo da construção do front-end em uma View, nesse exemplo nos mostra a construção da tela de login, onde é possível perceber na linha 15 a manipulação de imagem, sendo definido altura e largura, também nota-se que todos elementos estão contidos dentro de um frame, que permitir destacar toda área contida nele, também causando efeitos 3D nas plataformas Android e IOS.

A próxima figura nos mostra um exemplo de uma ViewModel. É possível perceber na linha 18 a interface INotifyPropertyChanged, que é responsável por

proporcionar um mecanismo unificado para implementar um único evento para todos os eventos dos objetos utilizados, sendo assim apresenta uma interação de forma rápida e eficiente ao usuário. Como exemplo temos as validações que são feitas de forma dinâmica apresentando a mensagem de erro e impossibilitando o envio do formulário antes mesmo do usuário clicar no botão de confirmação.

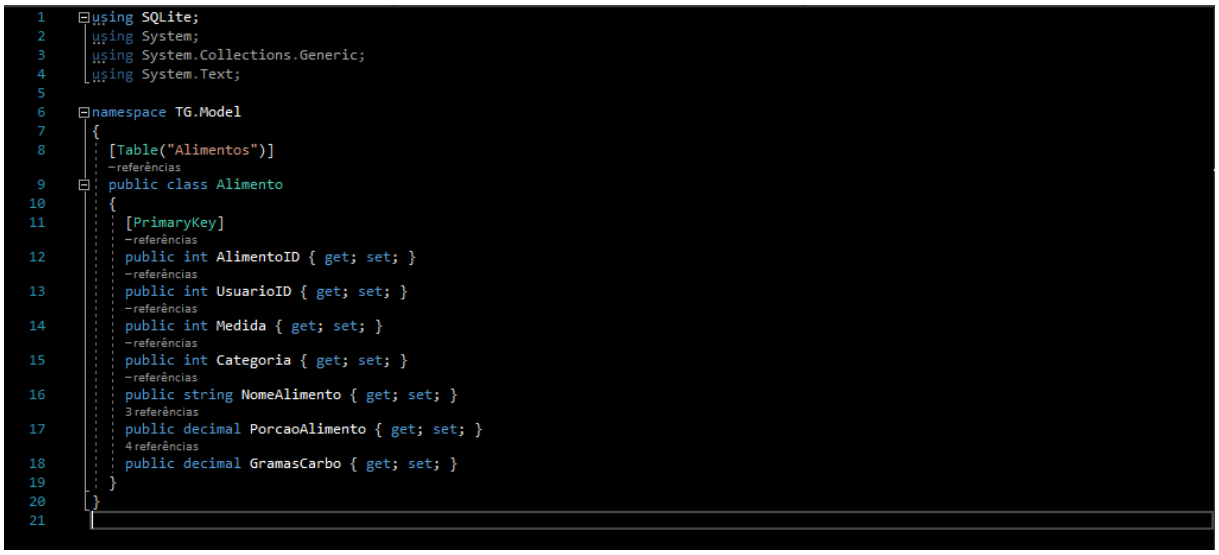
Figura 6 – ViewModel

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Text;

namespace TG_App.ViewModel
{
    1 referência
    class AlimentoViewModel : INotifyPropertyChanged
    {
        private int _Categoria;
        0 referências
        public int Categoria { get { return _Categoria; } set { _Categoria = value; OnPropertyChange("Categoria"); } }

        private int _Medida;
        0 referências
        public int Medida { get { return _Medida; } set { _Medida = value; OnPropertyChange("Medida"); } }

        public event PropertyChangedEventHandler PropertyChanged;
        2 referências
        private void OnPropertyChange(string param)
        {
            if (PropertyChanged != null)
            {
                PropertyChanged(this, new PropertyChangedEventArgs(param));
            }
        }
    }
}
```

Fonte: Autor

A seguinte figura nos mostra um exemplo de uma model, que é responsável por encapsular todos os dados utilizados. Através da model definimos qual tabela será utilizada no banco de dados e quais são as entidades.

Figura 6 – Model

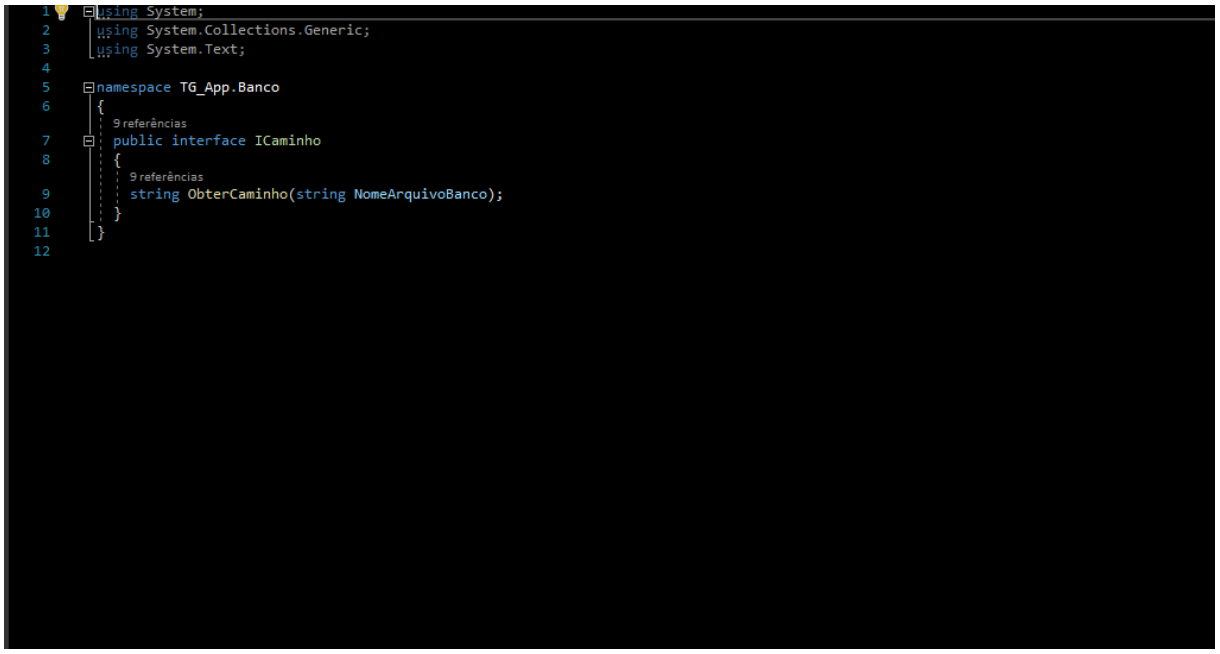


```
using SQLite;
using System;
using System.Collections.Generic;
using System.Text;

namespace TG.Model
{
    [Table("Alimentos")]
    -referências
    public class Alimento
    {
        [PrimaryKey]
        -referências
        public int AlimentoID { get; set; }
        -referências
        public int UsuarioID { get; set; }
        -referências
        public int Medida { get; set; }
        -referências
        public int Categoria { get; set; }
        -referências
        public string NomeAlimento { get; set; }
        3 referências
        public decimal PorcaoAlimento { get; set; }
        4 referências
        public decimal GramasCarbo { get; set; }
    }
}
```

Fonte: Autor

Para que fosse possível integrar a mesma linha de código para diversas plataformas diferentes foi-se implementado uma interface, com a finalidade de direcionar os dados para a plataforma correta.

Figura 6 – Interface



```
using System;
using System.Collections.Generic;
using System.Text;

namespace TG_App.Banco
{
    9 referências
    public interface ICaminho
    {
        9 referências
        string ObterCaminho(string NomeArquivoBanco);
    }
}
```

Fonte: Autor

A interface se comunica com as classes contidas em cada plataforma, dessa maneira possibilitando a reutilização do mesmo código, em cada classe é efetuado o código para cada tipo de plataforma retornando um resulta, no caso acima temos a manipulação de dados no banco de dados SQLite.

Figura 6 – Classe Da plataforma Android

```
using Xamarin.Forms;
using System.IO;
using TG_App.Droid.Banco;
using TG_App.Banco;

[assembly: Dependency(typeof(Caminho))]
namespace TG_App.Droid.Banco
{
    1 referência
    public class Caminho : ICaminho
    {
        7 referências
        public string ObterCaminho(string NomeArquivoBanco)
        {
            string caminhoDaPasta = System.Environment.GetFolderPath(System.Environment.SpecialFolder.Personal); //retorna o caminho até
             uma determinada pasta

            string caminhoBanco = Path.Combine(caminhoDaPasta, NomeArquivoBanco);

            return caminhoBanco;
        }
    }
}
```

Fonte: Autor

4 Resultados e Discussão (este item é obrigatório)

Apresentar *prints* do sistema, explicando cada funcionalidade que foi implementada. Caso o sistema tenha sido implantado em algum usuário, coletar e descrever informações sobre o processo de implantação e os benefícios levantados pelo usuário sobre a utilização do software.

Considerações finais

Relembrar quais foram objetivos iniciais, o que foi de fato desenvolvido, quais foram os principais desafios e quais serão os projetos futuros que poderão ser realizados.

Referências

BRASIL. Ministério da Saúde. **Diabetes Mellitus.** Brasília, 2006.

BRASIL. Ministério da Saúde. **Relatório de Recomendações.** Brasília, 2009.

BRASIL. Ministério da Saúde. **RASTREAMENTO E DIAGNÓSTICO DE DIABETES MELLITUS GESTACIONAL NO BRASIL**. Brasília, 2017.

Manual oficial de contagem de carboidratos regional / Sociedade Brasileira de Diabetes, Departamento de Nutrição. – Rio de Janeiro: Dois C: Sociedade Brasileira de Diabetes, 2009 il.

COIMBRA DE ARAÚJO, Everson. **Artigo Invista em você! Saiba como a DevMedia pode ajudar sua carreira. A evolução da linguagem de programação C#**. [*S. l.*], 2013. Disponível em: https://www.devmedia.com.br/a-evolucao-da-linguagem-de-programacao-csharp/28639. Acesso em: 15 out. 2021.

CRIS BRITO, Robson. **Utilizando SQLite em aplicativos Android**. [*S. l.*], 2015. Disponível em: https://www.devmedia.com.br/utilizando-sqlite-em-aplicativos-android/32117. Acesso em: 15 out. 2021.