



ISEL – INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA
ADEETC – ÁREA DEPARTAMENTAL DE ENGENHARIA DE
ELECTRÓNICA E TELECOMUNICAÇÕES E DE COMPUTADORES

LEIM

LICENCIATURA EM ENGENHARIA INFORMÁTICA E MULTIMÉDIA
UNIDADE CURRICULAR DE PROJETO

iMeal



Raquel Fortes (A46310)

Rita de Sousa Marques (A46313)

Orientador

Professor Rui Jesus

Julho, 2022

Resumo

O planeamento das refeições do quotidiano pode ser uma tarefa trabalhosa e, por vezes, difícil. Com a azáfama do dia a dia podemos ficar sem ideias do que cozinhar, problema que se torna ainda mais desafiante quando temos menos conhecimentos culinários.

Neste projeto foi desenvolvida a aplicação móvel iMeal, com o objetivo de permitir que os utilizadores realizem pesquisas rápidas de receitas, fornecendo os ingredientes que tenham disponíveis.

A aplicação utiliza um sistema de pesquisas personalizadas, de modo a tornar mais fácil obter receitas que possa cozinhar à hora da refeição. Neste documento é descrito todo o processo de desenvolvimento da aplicação incluindo os testes de avaliação de usabilidade realizados.

Abstract

Planning everyday meals can be a laborious and sometimes a difficult task. With the worries of the day to day we sometimes run out of ideas on what to cook, a problem that becomes even more challenging when we have less culinary knowledge.

In this project, the iMeal mobile application was developed, with the aim of allowing to perform quick searches for recipes providing the ingredients at their disposal.

The application uses a system of personalized searches, in order to make it easier to obtain recipes that can be cooked at mealtime. This document describes the entire application development process, including the usability evaluation tests performed.

Índice

Resumo	i
Abstract	iii
Índice	v
Lista de Tabelas	vii
Lista de Figuras	ix
Acrónimos e Siglas	xiii
1 Introdução	1
1.1 Motivação	1
1.2 Objetivos	2
1.3 Organização do Documento	2
2 Trabalho Relacionado	5
2.1 Mealime e Receitas fáceis e deliciosas	5
2.2 Healthy Food	7
2.3 Análise de funcionalidades	7
3 Modelo Proposto	9
3.1 Requisitos	9
3.1.1 Caracterização geral	9
3.1.2 Caracterização de Pormenor	10
3.2 Casos de utilização	13
3.3 Protótipo	16
3.4 Fundamentos	17

3.4.1	Receita	18
3.4.2	Filtro	18
3.4.3	Partilha e interação	18
4	Arquitetura	19
4.1	Abordagem	19
4.2	Arquitetura proposta	19
5	Implementação do Modelo	23
5.1	Servidor	23
5.1.1	Base de dados	24
5.1.2	Serviço de autenticação	29
5.1.3	Serviço de armazenamento de ficheiros	29
5.2	Cliente	29
5.2.1	Tecnologias	30
5.2.2	Estruturação do Modelo	31
5.2.3	Comunicação com o servidor	35
5.2.4	Autenticação	40
5.2.5	Módulos da Aplicação	47
6	Validação e Testes	61
6.1	Contexto	61
6.2	Participantes	62
6.3	Análise e Discussão de Resultados	63
7	Conclusões e Trabalho Futuro	65
7.1	Conclusões	65
7.2	Trabalho futuro	66
A	Modelo da base de dados	69
B	Modelo UML dos módulos do Cliente	73
C	Gráficos do questionário	81
	Bibliografia	89

Lista de Tabelas

2.1 Funcionalidades das aplicações relacionadas	8
2.2 Funcionalidades das aplicações relacionadas parte 2	8
3.1 Categorias de funções do sistema	11
3.2 Funcionalidades de Registo	11
3.3 Funcionalidades de Listagem	12
3.4 Funcionalidades de Autenticação	12
3.5 Atributos do sistema	13
5.1 Métodos <code>Get</code> da classe <code>DatabaseServices</code>	37
5.2 Métodos <code>Add</code> da classe <code>DatabaseServices</code>	38
5.3 Métodos <code>Remove</code> da classe <code>DatabaseServices</code>	39
5.4 Métodos <code>Update</code> da class <code>DatabaseServices</code>	39
5.5 Métodos <code>Set</code> da class <code>DatabaseServices</code>	39
5.6 Métodos da classe <code>AuthServices</code>	41

Listas de Figuras

2.1	App Mealime	6
2.2	App da Receitas fáceis e deliciosas	6
2.3	App da Healthy Food	7
3.1	Esquema geral	14
3.2	Esquema de publicação de uma receita	15
3.3	Esquema de pesquisa por uma receita	16
3.4	Protótipo da aplicação	17
4.1	Arquitetura proposta da aplicação	20
5.1	Modelo Entidade-Associação	24
5.2	Coleção recipes	28
5.3	UML dos módulos do cliente	33
5.4	Métodos da classe DatabaseServices	35
5.5	UML da classe AuthServices	40
5.6	Página de registo	42
5.7	Página de login	43
5.8	Sequência para reset da password	44
5.9	Email não verificado	45
5.10	Password incorreta	45
5.11	Email não registado	46
5.12	Página Welcome	47
5.13	Página Connect	48
5.14	Homepage sem sessão iniciada	49
5.15	Homepage com sessão iniciada	50
5.16	Página do utilizador	51
5.17	Página Followers	52

5.18	Página de utilizador da Rachel Fuertez	53
5.19	Página Favorites	54
5.20	Página Shoppinglist	55
5.21	Página Recipe	56
5.22	Sequência para publicação de um comentário	57
5.23	Menu na página de receitas	58
5.24	Página Addrecipe	59
5.25	Página Editrecipe	60
6.1	Gráfico correspondente ao género	62
6.2	Gráfico correspondente à faixa etária	63
6.3	Escala de pontuação do questionário SUS	63
A.1	Coleção comments	69
A.2	Subcoleção comment	69
A.3	Coleção ingredients	70
A.4	Coleção ratings	70
A.5	Coleção recipes	71
A.6	Coleção shoppinglist	71
A.7	Coleção users	71
B.1	Diagrama UML do módulo Login	73
B.2	Diagrama UML do módulo ForgotPassword	73
B.3	Diagrama UML do módulo Register	74
B.4	Diagrama UML do módulo HomePage	75
B.5	Diagrama UML do módulo UserPage	76
B.6	Diagrama UML do módulo Followers	77
B.7	Diagrama UML do módulo Favorites	77
B.8	Diagrama UML do módulo Shopping List	77
B.9	Diagrama UML do módulo RecipePage	78
B.10	Diagrama UML do módulo Addrecipe	79
B.11	Diagrama UML do módulo EditRecipe	80
C.1	Gráfico correspondente à frequência com que cozinha	81
C.2	Gráfico correspondente à Tarefa 1: Criar conta na aplicação e fazer login na aplicação	81
C.3	Gráfico correspondente à Tarefa 2: Visualizar uma receita	82

C.4 Gráfico correspondente à Tarefa 3: Classificar uma receita	82
C.5 Gráfico correspondente à Tarefa 4: Seguir outro utilizador de interesse	82
C.6 Gráfico correspondente à Tarefa 5: Adicionar uma receita	83
C.7 Gráfico correspondente à Tarefa 6: Pesquisar uma receita por ingredientes	83
C.8 Gráfico correspondente à Tarefa 7: Adicionar uma receita à lista de compras	83
C.9 Gráfico correspondente à frequência com que usaria a aplicação	84
C.10 Gráfico correspondente à complexidade da aplicação	84
C.11 Gráfico correspondente à facilidade da aplicação	84
C.12 Gráfico correspondente à necessidade de ajuda para usar a aplicação	85
C.13 Gráfico correspondente à integridade das funções da aplicação	85
C.14 Gráfico correspondente à inconsistência da aplicação	85
C.15 Gráfico correspondente à facilidade em aprender a utilizar a aplicação	86
C.16 Gráfico correspondente à confusão a utilizar a aplicação	86
C.17 Gráfico correspondente à confiança ao usar a aplicação	86
C.18 Gráfico correspondente às coisas novas aprendidas para usar a aplicação	87
C.19 Gráfico correspondente à clareza da informação	87
C.20 Gráfico correspondente à avaliação da aplicação em certos campos	87
C.21 Gráfico correspondente à utilidade da aplicação	88
C.22 Gráfico correspondente à classificação geral da aplicação	88

Acrónimos e Siglas

API - Application Programming Interface

CRUD - Create, Read, Update and Delete

JSON - JavaScript Object Notation

SCSS - Syntactically Awesome Style Sheet

SQL - Structured Query Language

SUS - System Usability Scale

UML - Unified Modeling Language

Capítulo 1

Introdução

Existe um tema comum ao quotidiano dos cidadãos de diversas partes do mundo, que é a preparação das nossas refeições, que sempre foi parte integral do estilo de vida do ser humano.

Todos nós já nos deparamos com a escolha, por vezes difícil, do que cozinhar à hora do jantar, almoço e entre outras refeições, sem termos, por vezes, a mínima ideia do que fazer. Olhar para dentro do frigorífico e ver que de facto não sabemos o que fazer com os ingredientes que temos... E quem nunca passou por este problema de certo que já viu familiares ou amigos vivenciarem-no.

1.1 Motivação

Dentro do contexto pandémico notou-se ainda mais como esta situação era de facto um problema real e global, sendo que a elaboração de refeições mais regulares e com mais pessoas vieram a salientar a dificuldade que, por vezes, temos em ter criatividade suficiente para decidir o que cozinhar.

Da vontade de solucionar este problema surgiu a ideia de criar uma ferramenta que permitisse aos utilizadores realizarem uma pesquisa rápida de receitas que possam cozinhar, fornecendo os ingredientes que pretendessem usar.

Esta ferramenta permitiria aos utilizadores realizarem uma pesquisa completamente personalizada, obtendo resultados imediatos e adequados às suas necessidades.

1.2 Objetivos

Deste conceito surge a aplicação Ideal Meal, abreviada para iMeal.

Este projeto assenta numa aplicação móvel que permite a utilizadores, com diferentes graus de conhecimento culinário, acederem de modo rápido e fácil a receitas que possam cozinar utilizando os ingredientes que têm ao seu dispor.

Recorrendo a um sistema de filtros por ingredientes ou tipos de dietas, por exemplo, dieta vegetariana, vegan, gluten-free, entre outros, são oferecidos ao utilizador resultados personalizados e adequados às necessidades específicas de cada um.

Numa vertente ainda pouco explorada nas aplicações deste área, existe a possibilidade da interação entre utilizadores na aplicação, em que são os próprios que publicam as suas receitas, gerando um sistema interligado de partilha de conhecimentos entre os diversos utilizadores da aplicação.

Desta forma, dar aos utilizadores a possibilidade de manterem o contacto indireto com outros utilizadores do seu interesse, ao seguirem-nos por exemplo, podendo seguir as suas novas publicações de receitas.

Pretendemos dar a oportunidade de explorar e conhecer novas receitas e métodos culinários dentro do perfil das necessidades pessoais de cada utilizador e, com isto tornar a aplicação numa ferramenta de aprendizagem e transmissão de novos conhecimentos.

1.3 Organização do Documento

A estrutura do relatório está de acordo com o processo de desenvolvimento da aplicação:

Introdução (capítulo 1) - neste capítulo é apresentado, de modo claro e sucinto, o conceito do projeto desenvolvido. É feita uma breve introdução da motivação que levou à iniciativa por detrás do desenvolvimento do projeto, o problema que pretende solucionar e os objetivos pretendidos com esta solução.

Trabalho Relacionado (capítulo 2) – neste capítulo são referenciados vários projetos analisados antes da realização do projeto. Estes serviram

como referências e inspiração para o desenvolvimento nas seguintes vertentes: tema, design, funcionalidades e informação apresentada.

Modelo Proposto (capítulo 3) – neste capítulo é feita uma avaliação inicial do modelo do projeto, sendo apresentados os requisitos funcionais e não funcionais do sistema, organizados em casos de utilização. Ainda neste capítulo são mencionados fundamentos importantes relativos ao desenvolvimento do projeto.

Arquitetura (capítulo 4) – neste capítulo é apresentada a arquitetura proposta sem que hajam especificações relativamente a dependências tecnológicas. Serão ainda descritos serviços e conceitos relativos ao projeto, tendo em conta a abordagem adotada para a implementação do mesmo.

Implementação do Modelo (capítulo 5) – neste capítulo é descrito todo o processo da implementação do projeto sendo expostas as dependências tecnológicas. É especificado o processo de implementação, assim como, a forma como foi concretizado. Será também apresentada e analisada com maior detalhe a interface gráfica da aplicação.

Validação e Testes (capítulo 6) – neste capítulo são descritos e analisados os testes realizados para avaliar o projeto desenvolvido.

Conclusões e Trabalho Futuro (capítulo 7) – neste capítulo são expostas as conclusões da realização do projeto e algumas referências de melhorias e objetivos que possam ser futuramente implementados.

Capítulo 2

Trabalho Relacionado

Antes da implementação da aplicação foi realizada uma pesquisa prévia por aplicações já existentes de conceito semelhante à aplicação proposta.

De entre os vários resultados das nossas pesquisas, destacaram-se três aplicações que foram utilizadas como maior referência para o nosso trabalho, devido ao seu design e conceito básico da nossa aplicação.

2.1 Mealime e Receitas fáceis e deliciosas

A aplicação Mealime é uma aplicação móvel cuja funcionalidade principal se destina a permitir ao utilizador criar um plano de refeições para a semana, baseado num sistema extremamente personalizado e adaptado ao perfil do utilizador.

Para este plano o utilizador tem acesso às seguintes opções.

- Seleção do tipo de dieta

Especificação do seu tipo de dieta, sendo-lhe mostrados apenas resultados enquadrados neste perfil.

- Seleção de ingredientes de que o utilizador não gosta

Especificação dos *dislikes*, ou seja ingredientes que não irão aparecer nas receitas propostas ao utilizador.

- Seleção de alergias e restrições

Especificação de alergias e restrições que devem ser excluídas das receitas dos utilizadores.

Esta aplicação serviu de referência para o nosso projeto devido:

- ao sistema de personalização utilizado como **filtro**;
- às receitas propostas ao utilizador na realização do seu plano de refeições;
- ao seu design simplista, que oferece uma interface onde a informação está organizada num sistema de *cards* (o utilizador tem uma lista na qual pode escolher as receitas que pretende adicionar ao seu plano).

A app Receitas fáceis e deliciosas tem um modelo muito semelhante à aplicação Mealime. No entanto, esta não permite a pesquisa pelo tipo de dieta de uma receita. As aplicações descritas são retratadas pelas Figuras 2.1 e 2.2.

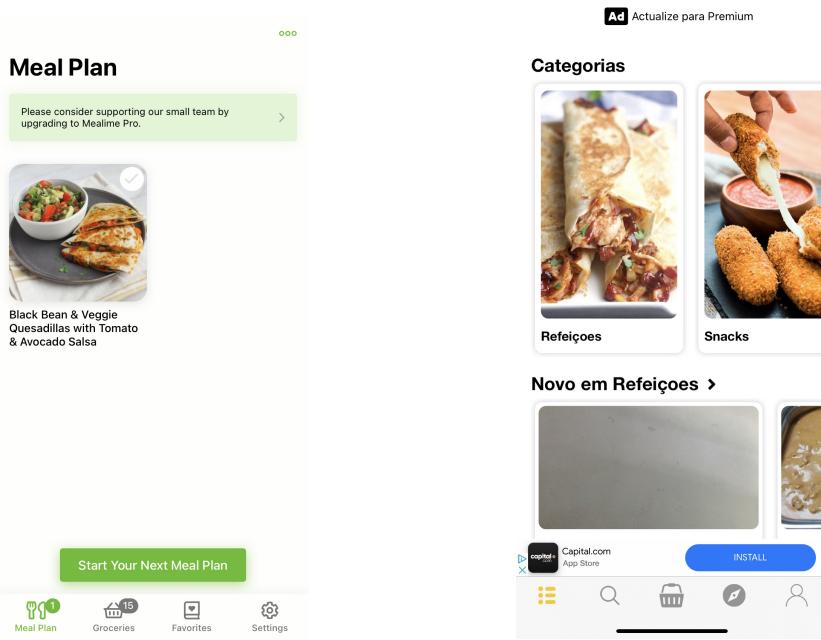


Figura 2.1: App Mealime

Figura 2.2: App da Receitas fáceis e deliciosas

2.2 Healthy Food

Na aplicação Healthy Food, Figura 2.3, vemos um sistema que permite ao utilizador adicionar receitas à sua lista de favoritos, classificar as receitas disponíveis na aplicação e ainda comentar na página da receita.

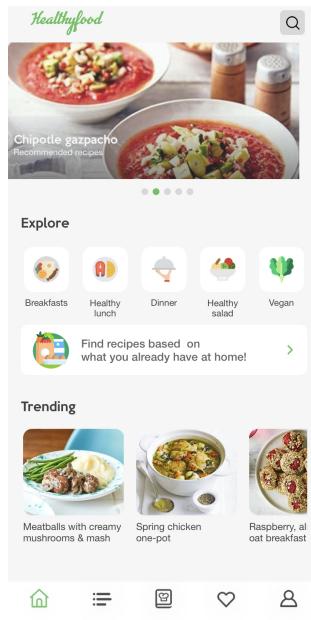


Figura 2.3: App da Healthy Food

2.3 Análise de funcionalidades

Para o desenvolvimento do nosso projeto retirámos algumas funcionalidades das aplicações mencionadas, conjugando-as às funcionalidades originais que foram integradas no projeto.

Nas Tabelas 2.1 e 2.2 podemos visualizar de modo mais sucinto as funcionalidades agregadas.

Tabela 2.1: Funcionalidades das aplicações relacionadas

Aplicações	Filtro por tipo de dieta	Lista de compras	Publicação de receitas	Edição de receitas
Mealime	✓	✓	✓	✓
Receitas fáceis e deliciosas		✓	✓	
Healthy Food	✓	✓		

Tabela 2.2: Funcionalidades das aplicações relacionadas parte 2

Aplicações	Comentários	Classificação	Favoritos	Disponibilidade IOS e Android
Mealime			✓	✓
Receitas fáceis e deliciosas			✓	✓
Healthy Food	✓	✓	✓	

Apesar de recorrer a funcionalidades existentes noutras aplicações, o projeto distingue-se das mesmas pela inclusão de funcionalidades originais, tais como: (1) permitir realizar um conjunto diferente de pesquisas; (2) ter características sociais distintas, como seguir e poder ser seguido por outro utilizador.

Capítulo 3

Modelo Proposto

Neste capítulo é apresentado o modelo proposto para a realização do presente projeto.

3.1 Requisitos

A análise dos requisitos do projeto foi realizada numa fase inicial, em que são definidos os requisitos com maior detalhe e pormenor.

3.1.1 Caracterização geral

- Objetivos

Este projeto assenta numa aplicação móvel que serve como uma ferramenta que permite aos utilizadores acederem de modo fácil e rápido a receitas que possam cozinar, com os ingredientes que têm ao seu dispor. Também é pretendido que, recorrendo a filtros, sejam oferecidos resultados personalizados e adequados às suas necessidades. Outro objetivo é a possibilidade da interação entre utilizadores.

- Públíco alvo

A aplicação iMeal, por ser direcionada como ferramenta complementar na preparação de refeições, é recomendada para um público mais adulto. Não obstante, pode ser utilizada por um público mais jovem dada a sua vertente didática, que permite uma maior facilidade em encontrar receitas acessíveis e simples de confeccionar.

- Funcionalidades

As funcionalidades do projeto foram possíveis de agrupar e sintetizar, tendo em conta as necessidades de um utilizador da aplicação e as funções que esta pretende disponibilizar, sendo demonstradas na seguinte lista.

- Filtros

De modo a alcançar o objetivo principal de permitir aos utilizadores uma pesquisa personalizada, foi identificado o sistema de filtros como uma das funções principais da aplicação.

- Publicação e partilha de receitas

A publicação de receitas surge como uma funcionalidade inovadora que permite uma maior interação entre utilizadores.

- Comentários e classificações

Uma das vertentes relevantes da interação entre utilizadores foi a possibilidade de publicar comentários em receitas e classificá-las.

- Favoritos e lista de compras

Os favoritos e a lista de compras servem como complemento para a filtragem, permitindo aos utilizadores ter acesso aos ingredientes que devem comprar ou adquirir para as refeições que pretendem confeccionar e ”guardar”.

3.1.2 Caracterização de Pormenor

Neste tópico são analisadas as funções e os atributos do sistema.

Funções do sistema

As funções do sistema traduzem o que o sistema deve implementar para que as suas funcionalidades sejam cumpridas. Devem ser definidas por permitirem estabelecer prioridades e identificar um possível consumo de recursos.

Para determinar a prioridade das funções são utilizadas as seguintes categorias, Tabela 3.1.

Tabela 3.1: Categorias de funções do sistema

Categoría	Descripción
Evidente	Esta função deve ser realizada com o conhecimento do utilizador.
Invisível	Esta função deve ser realizada, mas não será visível para o utilizador.
Adorno	Função opcional, ou seja a sua realização não irá afetar de modo significativo o custo ou outras funções.

As principais funções do nosso sistema podem ser agrupadas nas seguintes tabelas, divididas em três módulos distintos: os registos, as listagens e a autenticação.

Nos módulos de registo temos funções relativas ao registo de receitas, comentários, classificação e favoritos, Tabela 3.2.

Tabela 3.2: Funcionalidades de Registo

Registrar nova palavra-passe	Registrar nova palavra-passe de um utilizador	Invisível
Publicar receita	Registrar uma nova receita na aplicação	Invisível
Adicionar comentário	Registrar um comentário adicionado por um utilizador	Invisível
Adicionar classificação	Registrar uma classificação atribuída por um utilizador	Invisível
Adicionar aos favoritos	Registrar receita nos favoritos de um utilizador	Invisível
Adicionar à lista de compras	Registrar receita na lista de compras de um utilizador	Invisível

No módulo de listagens temos integradas várias funções que têm como principal objetivo demonstrar informação armazenada na aplicação ao utilizador, Tabela 3.3.

Tabela 3.3: Funcionalidades de Listagem

Requisito	Função	Categoria
Listar receitas	Visualizar receitas na página principal da aplicação.	Evidente
Listar seguidores	Visualizar lista de seguidores de um utilizador.	Evidente
Listar favoritos	Visualizar lista de favoritos de um utilizador.	Evidente
Listar lista de compras	Visualizar lista de compras de utilizador.	Evidente

Finalmente, para as funções de autenticação, encontram-se funcionalidades relativas ao registo e *login* do utilizador na aplicação, Tabela 3.4.

Tabela 3.4: Funcionalidades de Autenticação

Requisito	Função	Categoria
Autenticar utilizador	Efetuar início de sessão de um utilizador	Evidente
Registar utilizador	Registar utilizador na aplicação	Invisível
Adicionar imagem de perfil	Adicionar uma imagem por <i>default</i> ao utilizador	Invisível

Atributos do sistema

Os atributos do sistema traduzem características ou dimensões do sistema, ou seja, não são funções mas sim detalhes da estrutura do próprio sistema.

Os atributos podem ser **obrigatórios** ou **desejáveis**, de modo respetivo, tendo que ser contemplados pelo sistema ou haver uma adaptação dos sistema para os alcançar, Tabela 3.5.

Tabela 3.5: Atributos do sistema

Atributo	Detalhe/Restrição de fronteira	Categoría
Plataforma	Android	Obrigatório
Design simples e intuítivo	O design da aplicação deve ser simples e claro para os utilizadores da aplicação	Obrigatório
Interação pessoa-máquina	Formulários / caixas de texto	Obrigatório

3.2 Casos de utilização

Após a análise das funções e atributos do sistema, é possível identificar e construir os casos de utilização do sistema. Um caso de utilização permite-nos mapear a sequência de interações entre um sistema e um utilizador, num determinado ambiente, com o intuito de atingir um determinado objetivo.

Antes de passarmos à análise dos casos de utilização identificados no projeto devemos definir os **atores** identificados.

- Atores do sistema

São considerados atores quaisquer entidades que executam algum papel no sistema.

No projeto foram considerados atores o **utilizador registado** e o **utilizador não registado** também denominado como **convidado**. São responsáveis pela interação com os componentes da aplicação, havendo ligeiras diferenças entre si.

Tendo identificado os atores podemos passar à análise dos casos de utilização do projeto.

Estes foram divididos em três esquemas distintos: um esquema geral (Figura 3.1); um de criação de receita; e um de pesquisa de receita.

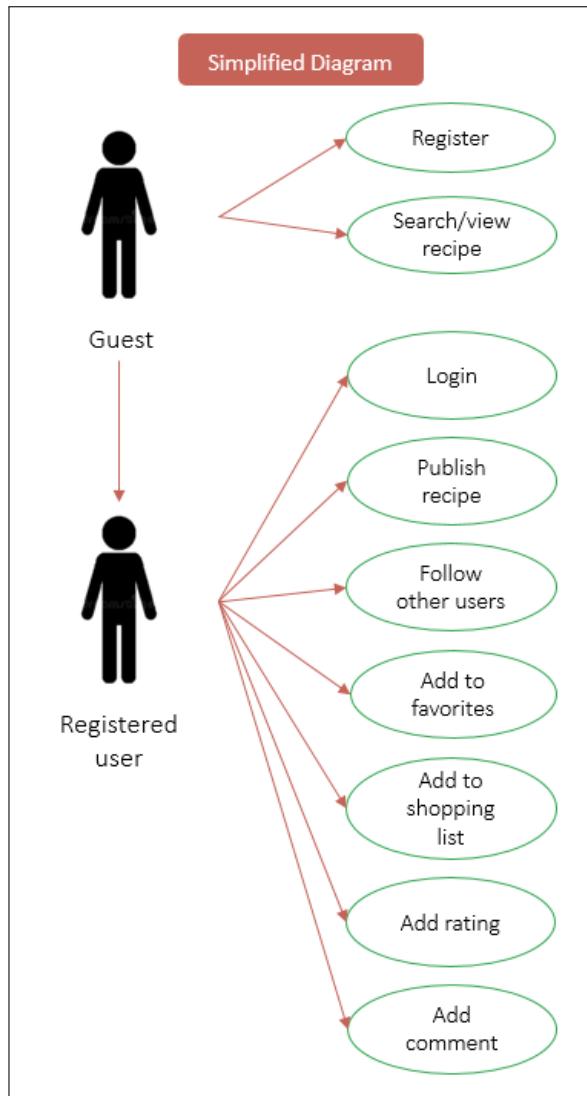


Figura 3.1: Esquema geral

O primeiro esquema representa 9 casos de utilização divididos entre os dois atores do sistema.

O primeiro refere-se ao Registo/Início de Sessão, no qual o utilizador realiza o seu registo. Ao realizar o registo deixa de ser convidado e passa a ser um utilizador registado. No segundo caso de utilização do convidado, este pode também realizar a visualização e pesquisa de receitas.

Os seguintes sete casos de utilização têm como ator o utilizador registado.

O primeiro caso demonstrado para este ator refere-se ao *Login*, realizado

pelo preenchimento de um formulário que permite ao utilizador aceder à sua conta de perfil e a outras funcionalidades da aplicação, descritas nas tabelas de requisitos demonstradas neste capítulo.

O seguinte caso de utilização, referente à **publicação de receitas**, permite ao utilizador adicionar uma receita à aplicação. De seguida, temos o caso **seguir outros utilizadores**, que representa a funcionalidade que permite ao utilizador seguir outros utilizadores de interesse registados na aplicação.

O seguinte caso de utilização refere-se ao registo de uma receita à lista de favoritos do utilizador.

De modo semelhante ao caso anterior, refere-se ao registo de uma receita à lista de compras do utilizador.

No caso **adicionar classificação**, o utilizador recorre a uma interface onde poderá selecionar qual a classificação que pretende atribuir à receita que está a visualizar de momento.

Finalmente, no caso de adicionar um comentário, o utilizador poderá inserir o comentário que pretende registar para a receita a ser visualizada.

Posto isto, iremos passar à análise dos esquemas de adição e de pesquisa por uma receita, Figura 5.1.

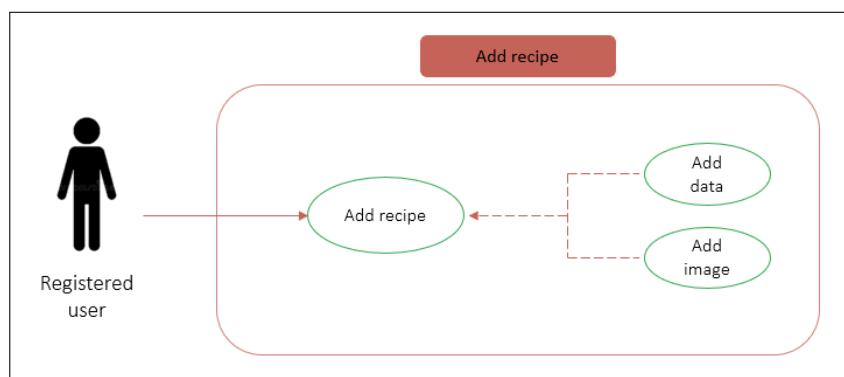


Figura 3.2: Esquema de publicação de uma receita

Para esta operação foram identificados três casos de utilização, sendo dois destes derivados de um caso principal.

Este caso principal refere-se à situação em que o utilizador adiciona uma receita à aplicação, podendo indicar informações sobre a receita e ainda adicionar imagens.

Já para a pesquisa de uma receita identificámos três casos de utilização,

Figura 3.3.

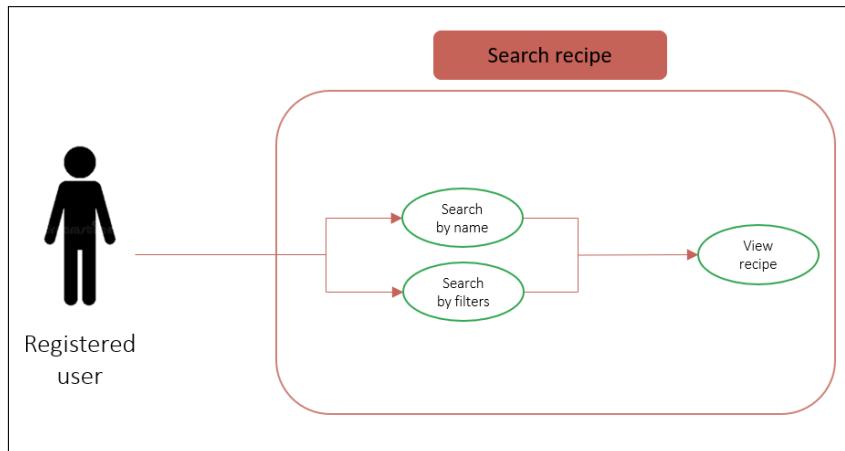


Figura 3.3: Esquema de pesquisa por uma receita

O primeiro, referindo-se à **Pesquisa por nome**, em que o utilizador poderá proceder à pesquisa de uma receita utilizando um termo.

O segundo, de modo semelhante ao primeiro caso, traduz uma pesquisa com mais parâmetros, a **Pesquisa por filtros**, onde o utilizador especifica os ingredientes e tipos de dieta das receitas que pretende obter.

O caso de utilização derivado destas pesquisas resulta na **Visualização de uma lista de receitas**, onde lhe são apresentadas as receitas correspondentes à sua pesquisa.

3.3 Protótipo

Depois de identificarmos e analisarmos os casos de utilização do sistema foi possível desenvolver um protótipo da aplicação, utilizando a ferramenta Figma.

Para a realização deste protótipo foi feito um pequeno estudo/pesquisa de modo a perceber quais os elementos que permitiriam ao utilizador ter uma experiência mais agradável e intuitiva.

Com esse estudo, chegámos à conclusão que, para demonstrar as receitas na *HomePage*, a melhor solução seria optar pela utilização de *cards*, e foi o que fizemos. Também escolhemos demonstrar a classificação média de uma receita logo nesse *card*, para que, sem ter que entrar na receita, o utilizador saiba logo como está classificada.

Na Figura 3.4 estão algumas das páginas desenvolvidas.

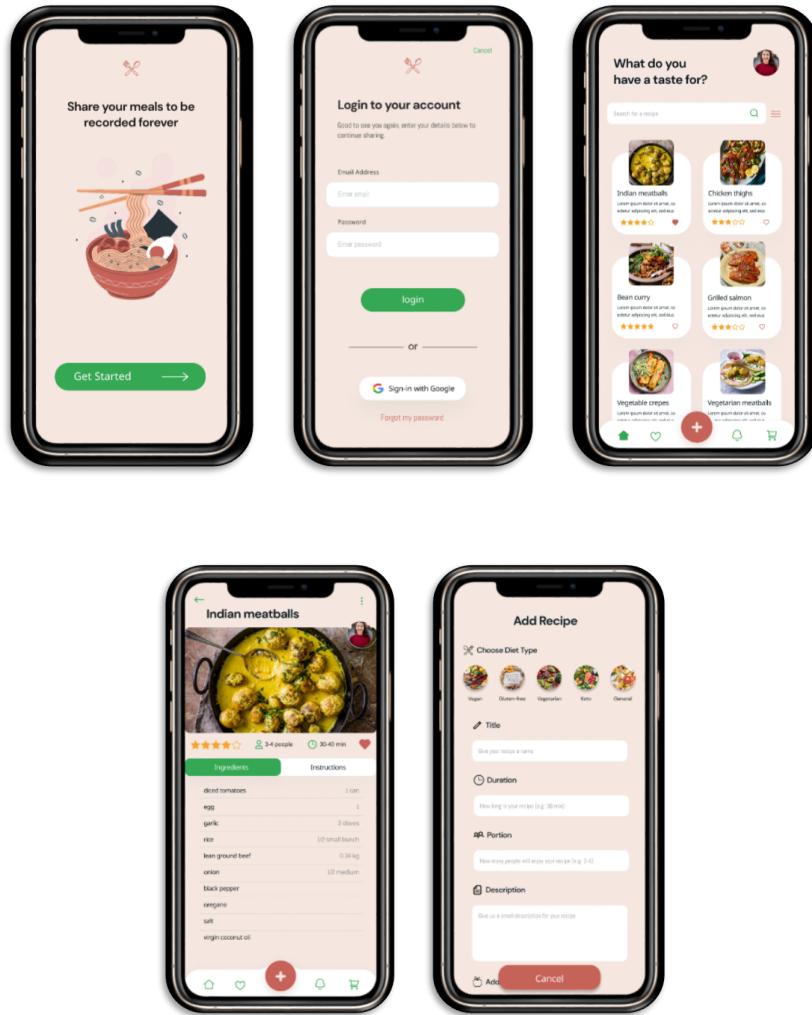


Figura 3.4: Protótipo da aplicação

O protótipo completo está disponível em: Protótipo iMeal.

3.4 Fundamentos

Nesta secção são abordados os conceitos fundamentais para o sustento das vertentes teóricas e tecnológicas do trabalho realizado.

Os fundamentos teóricos do projeto assentam nas noções de receita, de filtro e de partilha.

3.4.1 Receita

Sendo a aplicação desenvolvida na sua totalidade com o objetivo de fornecer receitas ao utilizador para facilitar o processo culinário, o conceito de receita tem grande relevância para o projeto em questão.

Uma receita, no contexto da aplicação, refere uma página onde é apresentado ao utilizador informação visual e textual sobre um conjunto de instruções a realizar para a preparação de determinada refeição.

Deste modo, a informação dos ingredientes constituintes da receita e as suas quantidades devem ser informação explícita na página, assim como os passos de confecção e ainda imagens que possam ilustrar o resultado final.

3.4.2 Filtro

O conceito de filtro é utilizado para concretizar a ideia de mostrar ao utilizador resultados, ou seja receitas que se enquadrem ao seu perfil e necessidades.

Deste modo o filtro permite-nos decidir de entre as receitas demonstradas na aplicação quais são adequadas para o utilizador em questão no momento da pesquisa.

3.4.3 Partilha e interação

Uma das funcionalidades da aplicação, tendo como meta a partilha de receitas entre utilizadores, necessita que haja uma definição concreta do conceito de partilha.

A conceito de partilha refere todas as interações permitidas entre os utilizadores da aplicação.

Chegámos assim à noção de partilha na situação em que um utilizador publica uma receita na aplicação, que estará de seguida disponível para visualizar por parte de todos os utilizadores da app.

Esta noção também é alcançada quando um utilizador publica um comentário na receita de outro utilizador, obtendo então uma interação menos conceptual e mais direta entre utilizadores.

O mesmo se dá quando um utilizador segue outro.

Capítulo 4

Arquitetura

Neste capítulo analisamos com maior detalhe as decisões tomadas para o desenvolvimento do projeto e que nos levaram à arquitetura proposta.

4.1 Abordagem

O projeto utiliza um sistema que apresenta uma arquitetura **Cliente-Servidor**.

Optou-se por esta arquitetura dada a necessidade de armazenar informação que deve estar disponível a diversos utilizadores autenticados em dispositivos distintos.

Posto isto, o cliente consiste numa aplicação móvel que comunica com a componente *back-end*.

Para a componente de *back-end*, foi utilizado um serviço que permite a interação com a base de dados através de uma API.

Assim, ao reunir todos os módulos necessários para obter uma abordagem adequada, resultou a arquitetura do sistema, no qual todos esses módulos se encontram interligados.

4.2 Arquitetura proposta

Para o correto funcionamento da aplicação optou-se por uma arquitetura que garante uma comunicação rápida e flexível com o servidor de modo a assegurar um acesso rápido à informação contida na base de dados.

Na arquitetura proposta, Figura 4.1, constatamos que existem 4 componentes principais do *back-end*. O servidor e três serviços complementares: a

base de dados; o serviço de autenticação; e o serviço de armazenamento de ficheiros. Todos funcionando de forma independente e integrada.

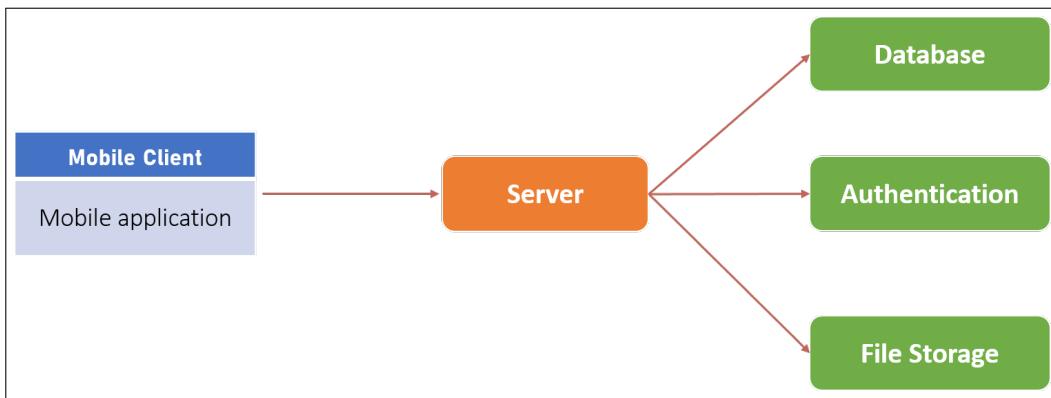


Figura 4.1: Arquitetura proposta da aplicação

- Base de dados

Na base de dados armazenamos toda a informação relativa às receitas da aplicação, detalhes dos utilizadores e diversos componentes pertencentes a funcionalidades da aplicação, tais como, as classificações e os comentários publicados em cada receita, a lista de compras de cada utilizador e a lista completa de ingredientes disponíveis na aplicação.

- Serviço de autenticação

Optámos por separar este componente da base de dados por ser mais difícil assegurar o armazenamento e tráfego seguro das credenciais de autenticação de um utilizador entre o cliente e a base de dados.

Assim, foi utilizado um serviço próprio que nos permite o armazenamento seguro das informações do utilizador sem correr riscos.

- Serviço de armazenamento de ficheiros

No serviço de armazenamento de ficheiros são armazenadas todas as imagens utilizadas na aplicação. Desde imagens utilizadas na página das receitas às imagens de perfil dos utilizadores.

Relativamente ao servidor, temos um serviço que funciona como um *Backend-as-a-Service*, ou seja, é um modelo de serviço da Cloud que maneja todas as operações realizadas no *back-end* da aplicação, sem que seja

necessário implementar o servidor e a comunicação com os restantes componentes da aplicação, sendo esta feita por API's e SDK's disponibilizadas pelo serviço.

Capítulo 5

Implementação do Modelo

Após o estudo prévio, descrito nos capítulos anteriores, passou-se à implementação do projeto.

Depois da definição da arquitetura do projeto começou-se com uma pesquisa exaustiva sobre as ferramentas que poderiam ser utilizadas para o desenvolvimento da aplicação.

Para a configuração do lado do servidor optou-se pela utilização do serviço Firebase disponibilizado pela plataforma Google.

Para o ambiente de desenvolvimento do lado do cliente foi utilizada a *framework* Ionic.

Neste capítulo são discutidas, justificadas e analisadas as opções de *software* para a implementação da aplicação, bem como todos os componentes implementados e o seu respetivo comportamento.

5.1 Servidor

Para a opção do servidor foi utilizado o serviço Firebase [1].

O Firebase é categorizado como um conjunto de ferramentas desenvolvidas pela Google, que permite uma comunicação direta entre o cliente e os serviços que disponibiliza, sem necessitar de implementar um servidor de raiz.

Nas seguintes secções iremos analisar com maior detalhe os serviços disponibilizados por esta plataforma, os motivos por trás da escolha de cada um e uma breve explicação do seu funcionamento.

5.1.1 Base de dados

Sendo um dos objetivos da nossa aplicação a distinção entre utilizadores registados e não registados e um serviço que permite publicar receitas na aplicação e visualizar receitas publicadas por outros utilizadores, é imprescindível que exista o conceito de base de dados.

Modelo

Para a arquitetura da base de dados foi necessária a descrição do modo como a informação deveria ser organizada.

Foi realizado o **Modelo Entidade-Associação**, demonstrado na Figura 5.1, para expressar os objetos que interagem entre si na aplicação, traduzidos no modelo por entidades e associações.

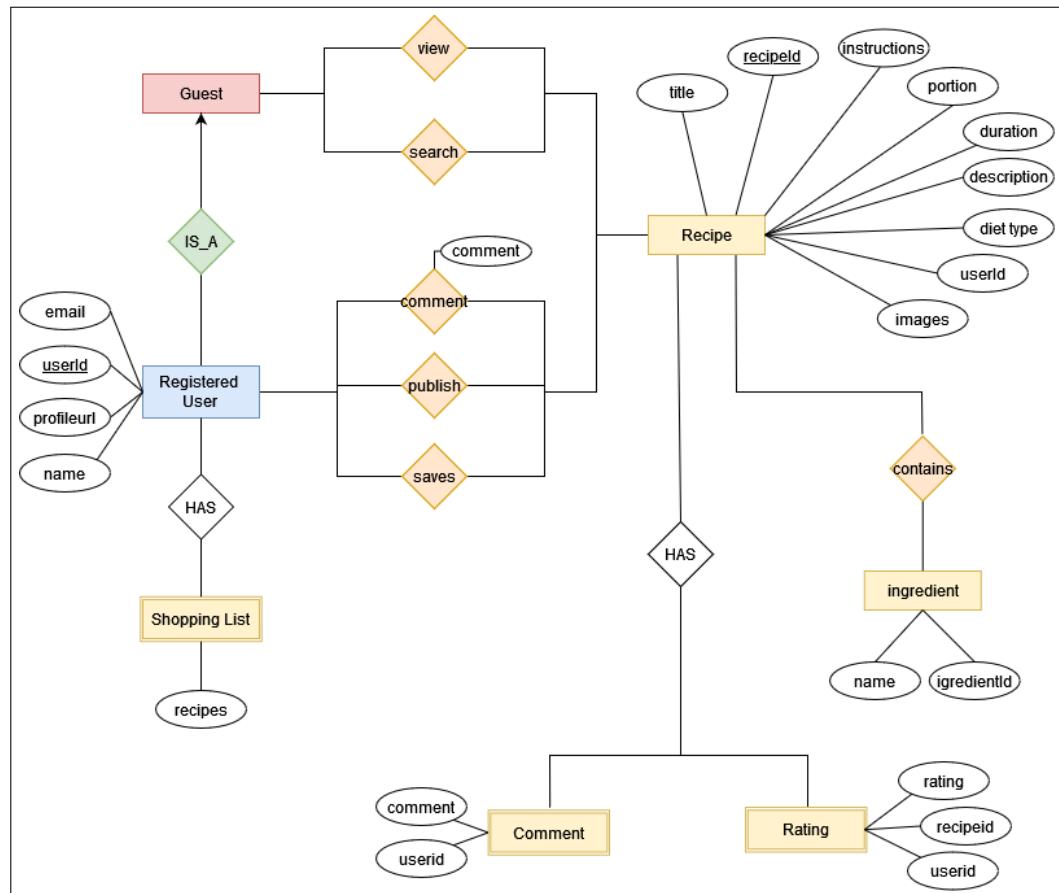


Figura 5.1: Modelo Entidade-Associação

Neste diagrama existem seis entidades principais: **Guest**, **Registered User**, **Recipe**, **Ingredient**, **Shopping List** e **Rating**.

Para representar a interação entre entidades temos as associações representadas por um losango laranja.

Guest

A entidade **Guest** não possui atributos, tendo duas associações: **view** e **search**.

A primeira permite ao utilizador visualizar receitas e a segunda permite-lhe pesquisar receitas.

Registered User

Como indicado no modelo, um **Registered User** é um **Guest** que tem funcionalidades adicionais. Esta entidade tem os seguintes atributos: **email**, **profileurl**, **name** e **userid**, sendo identificado na base de dados pelo **userid**.

Tem três associações com a receita: **comment**, que permite ao Utilizador Registado (português) comentar uma receita; **publish**, que indica o ato de publicar uma receita e finalmente **save**, que permite ao utilizador guardar uma receita nos favoritos.

Recipe

A **Recipe** corresponde à entidade que refere uma receita.

Este entidade tem um **título**, uma **descrição**, **instruções**, **porções**, **duração**, **tipo de dieta**, um **recipeid** e uma referência para a **chave estrangeira user.id**.

Contém **ingredientes** e tem um **rating** e um ou mais comentários.

Ingredient

A entidade **Ingredient**, tem um nome e um id, podendo ser contida por uma receita.

Shopping List

A entidade **Shopping List** é entidade fraca de **Registered User**, dado não existir independentemente.

Não faz sentido haver uma lista de compras se esta não for contida por um utilizador.

Rating

A entidade **Rating** é entidade fraca de uma **Recipe**, sendo identificada pelos atributos **userid** e **recipeid**. Possui também um **rating**.

Comment

A entidade **Comment** é entidade fraca de uma **Recipe**, sendo identificada pelo atributo **userid**. Possui também o atributo **comment**.

Implementação

Para a nossa aplicação ponderamos a escolha de dois tipos de bases de dados. A base de dados SQL ou relacional e a base de dados NoSQL.

Passemos a uma análise das principais diferenças entre estas duas arquiteturas de bases de dados.

Características de uma base de dados SQL

- Base de dados relacional
- Esquema fixo ou predefinido

Para as bases de dados SQL, o esquema de relação deve ser pré definido e maioritariamente fixo.

- São preferidas para sistemas mais complexos

Sistemas onde o esquema de relação é muito complexo, com várias entidades e associações são mais adequados para bases de dados do tipo SQL.

- Escalamento vertical

Permitem o aumento do armazenamento de informação num mesmo servidor, adicionando à base de dados mais tabelas por exemplo.

Características de uma base de dados NoSQL

- Base de dados não relacional
- Esquema dinâmico

Um banco de dados NoSQL possui esquema dinâmico para dados não estruturados. Os dados são armazenados de várias maneiras, o que

significa que podem ser orientados a documentos, orientados a colunas, baseados em gráficos ou organizados como um armazenamento de *Key-Value*. Essa flexibilidade significa que os documentos podem ser criados sem existir uma estrutura definida. Além disso, cada documento pode ter sua própria estrutura exclusiva. Podem ser adicionados campos à medida que se progride.

- São preferidas para sistemas de menor complexidade
- Escalamento horizontal

Permitem o aumento do armazenamento de informação através da utilização de mais servidores.

Após identificação das principais diferenças entre uma base de dados SQL e NoSQL, optámos por utilizar uma base de dados NoSQL, por nos permitir adicionar novas funcionalidades à base de dados sem necessitar de a reestruturar totalmente e sem causar *downtimes* (periódos em que um sistema se encontra indisponível).

Como o projeto consiste numa aplicação a ser utilizada por vários utilizadores, sendo inclusivamente suscetível a atualizações à base de dados, esta característica torna-se muito útil para o desenvolvimento do projeto, por permitir a adição de novas funcionalidades sem comprometer o funcionamento de outros serviços.

Em contrapartida é importante referir que este tipo de base de dados, NoSQL, não oferece grande consistência, pelo que é possível, por exemplo, adicionar dados repetidos sem que seja gerado qualquer erro.

Escolhemos a tecnologia Cloud Firestore [2], base de dados NoSQL real-time, que funciona com uma arquitetura documento-coleção, ou seja, a informação é armazenada em documentação estruturada por ficheiros JSON.

A utilização desta tecnologia permitiu realizar as operações de *Get*, *Update* e *Add* de modo rápido e flexível.

Foi possível adicionar funcionalidades novas à aplicação durante o seu desenvolvimento, refazendo apenas os modelos de dados por questões de coerência.

Posto isto, vejamos com um pouco mais de detalhe o modo como funciona este tipo de base de dados, que, como já foi referido anteriormente, funciona com uma arquitetura de documento-coleção.

- Documento

No Cloud Firestore a unidade de armazenamento é o documento, sendo cada documento identificado por um nome.

Um documento contém campos que são mapeados para valores, podendo também ter subcoleções e objetos encapsulados dentro de outros (denominados mapas).

Estes documentos funcionam na sua forma mais elementar como registros JSON.

- Coleção

As coleções funcionam como recipientes onde são contidos os documentos.

Uma coleção só pode, única e exclusivamente, conter documentos.

Na Figura 5.2 é possível observar uma imagem ilustrativa de uma coleção na base de dados.

```

+ Iniciar coleção
comments
ingredients
ratings
recipes >
shoppinglist
users

+ Adicionar documento
1PWsABtA3UPwz4hPwfP
44a0Kqhmxx2tCzUUvvQ3
HUsNP08LwzzDkE18Pg5q
L1z8qY1cQ3n2vfV7s8lp
LqYitjZcfqm2nauZsGqZ
XzeEFwZT1WN85zbsaVwP
ZCzvPgyKAh1AIxnWk1ph
cqytFQuiTxtCzxADgm5q
gzhGdcuoln4ak0eDmuuA
huK1iCYnTB01Wkx046aT
pqYrrCs5QvRVK01muCR
rVEVmtnNbUY6qZvnTfBZ
wZEHE8vY8CeA8h0Rv7Y1

+ Iniciar coleção
+ Adicionar campo
description: "Cheese tortellini makes the best quick dinner recipel Serve it up in a creamy marinara sauce flavoured with goat cheese and basil."
dietType: "Vegetarian"
duration: "15"
images: ["https://firebasestorage...."]
ingredients: [{"ingunit: "12 to 16 ounce..."]}
instructions: ["Boil the tortellini acco..."]
portion: "4"
title: "Cheese Tortellini"
userid: "7pBxFbDFZzBzYebfsA06DEgjQAE3"

```

Figura 5.2: Coleção recipes

Na Figura 5.2 pode visualizar-se a coleção **recipes**, que contém diversos documentos que representam uma receita e os seus campos constituintes.

No apêndice A é possível observar a implementação de todas as coleções da base de dados.

5.1.2 Serviço de autenticação

Para o serviço de autenticação, como referido na Secção 4.2, optou-se por um serviço que assegurasse a segurança e privacidade dos dados dos utilizadores.

Foi utilizada a ferramenta Firebase Authentication [3] que torna a autenticação mais segura e suporta várias formas de a realizar, tais com a autenticação normal - através do e-mail e password -, das redes sociais ou através do número de telefone.

No momento em que a autenticação ocorre, no cliente, é gerado um *token*, de duração limitada, que vai permitir identificar o utilizador nos pedidos ao servidor do sistema. Este *token* é incluído em todas as comunicações com o servidor, sendo validado por este através da utilização do serviço de autenticação do Firebase.

Erros de Autenticação

Para além de permitir uma autenticação segura e fiável, o serviço de autenticação do Firebase também dispõe de um sistema de deteção de erros, ou seja, o utilizador é notificado quando a autenticação não funciona sendo inclusive notificado acerca do respetivo erro.

5.1.3 Serviço de armazenamento de ficheiros

Recorrendo novamente ao Firebase, utilizámos o serviço Firebase Cloud Storage [4] para o armazenamento dos ficheiros utilizados na aplicação.

Este serviço permite que seja realizado o upload e a partilha de conteúdo inserido pelo utilizador, tal como imagens e vídeos.

Todos os dados são armazenados numa cloud da Google sendo possível aceder-lhes através de uma referência para este banco de dados no código implementado na vertente do cliente.

5.2 Cliente

Nesta secção iremos descrever a implementação do modelo na aplicação cliente. Esta passa pelos modelos utilizados, pelas decisões de software e pela interface da aplicação.

5.2.1 Tecnologias

Para a implementação do projeto começou-se com uma pesquisa exaustiva sobre as ferramentas que poderiam ser utilizadas para o desenvolvimento da aplicação.

Foram isolados os seguintes ambientes de desenvolvimento, Android Studio, Ionic, Flutter e React Native.

Optou-se pela utilização da *framework* Ionic, pelos motivos descritos:

- Ferramenta multi-plataforma

Utilizando o mesmo código base é possível exportar uma aplicação implementada nesta ferramenta para os sistemas Web, Android e IOS;

- Utilização de tecnologias da Web;

Através desta ferramenta é possível criar uma aplicação dinâmica e apelativa, utilizando técnicas já conhecidas e trabalhadas tais como a linguagem de marcas HTML e a linguagem de programação Typescript com a integração de *frameworks* tais como o Angular, o React Native e o Vue.

Optou-se por não utilizar as restantes plataformas pelas seguintes razões:

- Android SDK

O Android SDK foi recusado principalmente por não permitir o desenvolvimento de uma aplicação, tanto para IOS como para Android, e devido à sua interface não permitir a criação de interfaces tão fluídas e dinâmicas como as criadas por outras tecnologias.

A dificuldade de configuração do Android SDK devido aos erros de incompatibilidade e de versões solidificou ainda mais esta decisão, por não ser possível disponibilizar tempo extra na resolução de problemas no ambiente de desenvolvimento.

- Flutter

A *framework* Flutter, permite ao programador o desenvolvimento de aplicações para os sistemas IOS e Android utilizando apenas um código base, recorrendo à linguagem de programação Dart.

Por ser uma plataforma inexplorada implica a aprendizagem de uma linguagem ainda não estudada e dada à incerteza relativa quanto ao tempo que poderia ser disponibilizado para a aprendizagem desta nova tecnologia, esta opção foi posta de parte.

- React Native

O React Native é uma *framework* desenhada para a criação de aplicações para plataformas IOS, Android e também aplicações web, utilizando o mesmo código base.

De modo similar ao Flutter, o React Native, não foi escolhido devido à utilização de tecnologias inexploradas.

Dada por escolhida a *framework* de desenvolvimento da aplicação móvel, passou-se à implementação concreta da app.

5.2.2 Estruturação do Modelo

Para uma melhor organização da implementação, foi estruturado um modelo que permitiu a divisão das componentes da aplicação em módulos diferentes, cada um correspondente a uma página da aplicação.

Façamos uma breve análise da estrutura dos ficheiros mais relevantes do Ionic quando se dá a criação de uma nova componente.

- Ficheiro module.ts

O ficheiro module.ts gerado em cada página permite a implementação de módulos e plugins implementados no módulo raiz da aplicação.

- Ficheiro page.html

O ficheiro page.html é onde é implementado o html da página. Incorpora componentes próprios do Ionic, que podem ser utilizados em vez do html tradicional, sendo especificamente adaptados para a *framework*.

- Ficheiro page.scss

O ficheiro page.scss suporta a linguagem SCSS, um subconjunto do CSS que incorpora funcionalidades que não estão presentes neste último.

- Ficheiro page.ts

O ficheiro page.ts suporta a linguagem Typescript, um superconjunto do Javascript, com uma linguagem mais orientada à programação por objetos, onde é implementada uma classe que permite o acesso à página html e a implementação de qualquer código necessário à aplicação.

Posto isto, passemos à descrição dos módulos gerados para o funcionamento da aplicação.

Para a implementação da app foram geradas 14 páginas constituídas pelos componentes descritos anteriormente na secção 5.2.3.

Cada uma das classes geradas desempenha uma função no contexto da aplicação. É de notar que cada uma destas classes vai funcionar de modo independente, interagindo apenas com os serviços Ionic implementados para a interação com a Firebase.

Na Figura 5.3, podemos observar o esquema UML correspondente à relação entre as várias classes. Uma versão mais detalhada deste esquema pode ser encontrada no Apêndice B

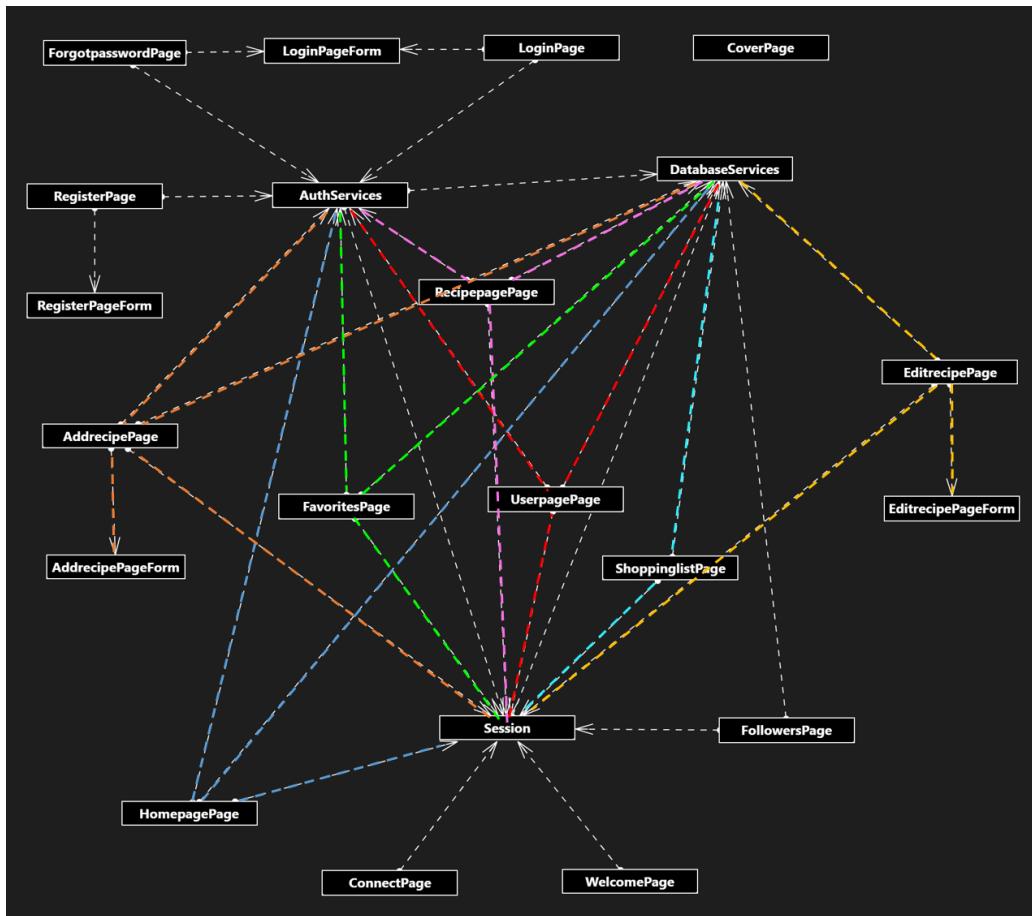


Figura 5.3: UML dos módulos do cliente

Podemos resumir a divisão destes módulos da seguinte forma.

Serviços

- DatabaseServices
- AuthService
- Session

Módulos

- Geral
 - WelcomePage;
 - ConnectPage;

```
LoginPage;  
ForgotpasswordPage;  
RegisterPage;  
HomePage;
```

- Utilizador

```
UserPage;  
FollowersPage;  
FavoritesPage;  
ShoppinglistPage;
```

- Receita

```
RecipePage;  
AddrecipePage;  
EditrecipePage;
```

Nas secções seguintes será feita uma análise mais detalhada destes módulos e a sua respetiva função no desempenho do projeto.

5.2.3 Comunicação com o servidor

Uma vez que toda a informação da aplicação se encontra na componente back-end, é necessário que haja uma comunicação entre esta componente e o cliente.

Nesse sentido foi desenvolvida uma classe responsável por esta comunicação, denominada de `DatabaseServices`, representada na Figura 5.4.

DatabaseServices	
recipes :any	
constructor(firestore)	
addComment(authorid,recipeid,commentid) :Promise	
addFollower(userid,followerid) :Promise	
addFollowing(userid,followid) :Promise	
addIngredient(ingname:any) :Promise	
addRating(recipeid,userid,rating) :Promise	
addRecipe(authorid,recipeInfo) :Promise	
addRecipeFavorites(userid,recipeid) :Promise	
addShoppinglist(userid,recipeid) :Promise	
adduserInfo(username,usemail,userid) :Promise	
getComments(recipeid:string) :Promise	
getfavoriteRecipes(userid:string) :Promise	
getFollowers(userid:string) :Promise	
getFollowing(userid:string) :Promise	
getIngredients() :Promise	
getIngredientInfo(ingredientid) :Promise	
getmedianRating(recipeid:string) :Promise	
getRating(recipeid,userid) :Promise	
getRecipe(recipeid:string) :Promise	
getRecipes() :Promise	
getRecipesByDiet(dietType:string) :Promise	
getRecipesByExclusiveIng(ingSearchList) :Promise	
getRecipesByFilter(ingFilter,dietType,exclude) :Promise	
getRecipesByIng(ingSearchList) :Promise	
getRecipesByIngredient(ingredientList,exclude) :Promise	
getShoppinglist(userid:string) :Promise	
getUserInfo(userid:string) :Promise	
getUserRecipes(userid:string) :Promise	
removeFollower(userid,followerid) :Promise	
removeFollowing(userid,followid) :Promise	
removeRecipeFavorites(userid,recipeid) :Promise	
removeShoppinglist(userid,recipeid) :Promise	
setRecipeImages(recipeid,imageurl) :Promise	
setUserAvatar(userid,imageurl) :Promise	
updateRating(recipeid,userid,newrating) :Promise	
updateRecipe(recipeid,parameters) :Promise	

Figura 5.4: Métodos da classe `DatabaseServices`

O serviço `DatabaseServices` é responsável por toda a comunicação com a base de dados, acedendo diretamente a uma referência do Firestore.

Nesta classe foram implementadas todas as operações de `create`; `read`; `update`; `delete` à base de dados, ou operações CRUD, sendo instanciadas em várias classes da aplicação conforme seja necessário.

Todos os métodos implementados nesta classe irão retornar um objeto do tipo `Promise`, dado os métodos de acesso à base de dados da API do Firestore serem assíncronos. Deste modo, foi possível detetar as respostas provenientes da base de dados, realizando determinadas operações apenas quando estas estivessem disponíveis.

Nas tabelas seguintes estão expressos os principais métodos implementados. Não foram incluídos os métodos auxiliares.

Na Tabela 5.1 encontram-se os métodos `Get`, ou seja, ir buscar informação à base de dados.

Na Tabela 5.2 os métodos `Add` - adicionar informação à base de dados.

Na Tabela 5.3 os métodos `Remove` - remover informação da base de dados.

Na Tabela 5.4 os métodos `Update` - atualizar informação da base de dados.

E na Tabela 5.5 os métodos `Set` - atribuir novos parâmetros a coleções já existentes.

Tabela 5.1: Métodos Get da classe DatabaseServices

Método	Descrição	Resposta
<code>getUserInfo()</code>	Obtém informação relativa ao utilizador especificado pelo userid	Object com userid, nome, email e imagem de perfil
<code>getUserRecipes()</code>	Obtém todas as receitas publicadas pelo utilizador especificado pelo userid	Array de Objects com toda a informação relativa às receitas correspondentes
<code>getFavoritesRecipes()</code>	Obtém todas as receitas armazenadas na lista de favoritos do utilizador especificado pelo userid	Array de Objects com toda a informação relativa às receitas correspondentes
<code>getShoppingList()</code>	Obtém todas as receitas armazendas na lista de compras do utilizador especificado pelo user id	Array de Objects com toda a informação relativa às receitas correspondentes
<code>getFollowers()</code>	Obtém todos os seguidores do utilizador especificado pelo user id	Array com id de todos os utilizadores correspondentes
<code>getFollowing()</code>	Obtém todos os utilizadores que o utilizador especificado pelo user id segue	Array com id de todos os utilizadores correspondentes
<code>getRecipes()</code>	Obtém todas as receitas registadas na base de dados	Array de Objects com informação sobre todas as receitas na base de dados
<code>getRecipe()</code>	Obtém informação relativa ao utilizador especificado pelo recipeid	Object com toda a informação relativa à receita especificada
<code>getRecipesByFilter()</code>	Obtém todas as receitas selecionadas pelos filtros passados como argumento da função	Array de Objects com toda a informação relativa às receitas correspondentes
<code>getRating()</code>	Obtém a classificação dada pelo utilizador especificado pelo userid à receita especificado pelo recipeid	Objeto do tipo number
<code>getComments()</code>	Obtém todos os comentários registados na receita especificada pelo recipeid	Array de Object com toda a informação relativa aos comentários correspondentes (comentário e autor)

Tabela 5.2: Métodos Add da classe DatabaseServices

Método	Descrição
<code>adduserInfo()</code>	Adiciona um novo utilizador à base de dados com os dados especificados como argumento
<code>addrecipeFavorites()</code>	Adiciona a receita especificada pelo recipeid à lista de favoritos do utilizador especificado pelo userid
<code>addShoppinglist()</code>	Adiciona a receita especificada pelo recipeid à lista de compras do utilizador especificado pelo userid
<code>addFollower()</code>	Adiciona o utilizador especificado pelo followerid à lista de seguidores do utilizador especificado pelo userid
<code>addFollowing()</code>	Adiciona o utilizador especificado pelo followid à lista de utilizadores que o utilizador especificado pelo userid segue
<code>addRating()</code>	Adiciona um valor de classificação de rating à receita especificada pelo recipeid, no documento de classificações
<code>addComment()</code>	Adiciona um comentário à receita especificada pelo recipeid no documento dos comentários

Tabela 5.3: Métodos `Remove` da classe `DatabaseServices`

Método	Descrição
<code>removeFollower()</code>	Remove da lista de seguidores do utilizador especificado pelo userid o utilizador especificado pelo followerid
<code>removeFollowing()</code>	Remove da lista de utilizadores do utilizador especificado pelo userdid o utilizador especificado pelo followid
<code>removerecipeFavorites()</code>	Remove da lista de favoritos do utilizador especificado pelo userid a receita especificada pelo recipeid
<code>removeShoppinglist()</code>	Remove da lista de compras do utilizador especificado pelo userid a receita especificado pelo recipeid

Tabela 5.4: Métodos `Update` da class `DatabaseServices`

Método	Descrição
<code>updateRating()</code>	Atualiza a classificação da receita especificada pelo recipeid
<code>updateRecipe()</code>	Atualiza os parâmetros da receita especificado pelo recipeid

Tabela 5.5: Métodos `Set` da class `DatabaseServices`

Método	Descrição
<code>setrecipeImages()</code>	Define as imagens da receita especificada pelo recipeid
<code>setuserAvatar()</code>	Define a image de perfil do utilizador especificado pelo userid

5.2.4 Autenticação

Para a comunicação e integração dos serviços de autenticação do Firebase Authentication foi implementada a classe `AuthServices`.

É nesta classe que são implementados os métodos de início de sessão, de registo, de redefinição da senha e de fim de sessão.

Este serviço é instanciado nos módulos de *login*, registo, página de utilizador (para efetuar o *logout*), e de redefinição da senha, tratando também os erros que possam ocorrer durante estas operações.

Na Figura 5.5 temos a representação desta classe.

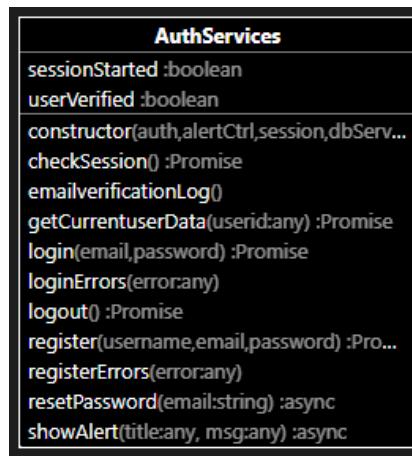


Figura 5.5: UML da classe `AuthServices`

Na Tabela 5.6 estão expressos os principais métodos implementados nesta classe. Mais uma vez, não foram incluídos os métodos auxiliares por questões de relevância para o tópico a ser abordado.

Tabela 5.6: Métodos da classe `AuthServices`

Método	Descrição
<code>register()</code>	Regista um novo utilizador na aplicação, permitindo ao utilizador efetuar o <i>login</i> através das suas credenciais
<code>login()</code>	Efetua o <i>login</i> do utilizador na aplicação
<code>logout()</code>	Efetua o <i>logout</i> do utilizador na aplicação
<code>resetPassword()</code>	Envia ao utilizador um email para a redefinição da palavra-passe
<code>checkSession()</code>	Verifica se existe algum utilizador com sessão iniciada na aplicação
<code>loginErrors()</code>	Trata os erros que possam ocorrer durante o <i>login</i> do utilizador na aplicação
<code>registerErrors()</code>	Trata os erros que possam ocorrer durante o registo do utilizador na aplicação

Posto isto, vejamos o modo de funcionamento de algumas destas operações.

O registo é efetuado inserindo uma conta de email válida e uma password, preenchendo os campos visualizados na Figura 5.6.

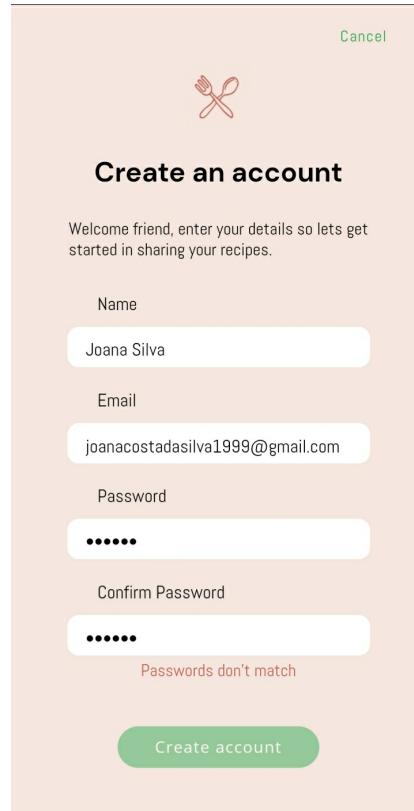


Figura 5.6: Página de registo

Para ter acesso ao botão de ”Create account”, que permite que seja efetuado o registo do utilizador, deve garantir que inseriu um email válido e uma password com um mínimo de 6 caracteres que deve confirmar noutra campo do registo.

Após efetuar o registo, será enviado um email de verificação para o email inserido pelo utilizador, onde este poderá ativar a sua conta.

Após ativada, o utilizador poderá proceder ao *login* na aplicação, situação ilustrada na Figura 5.7

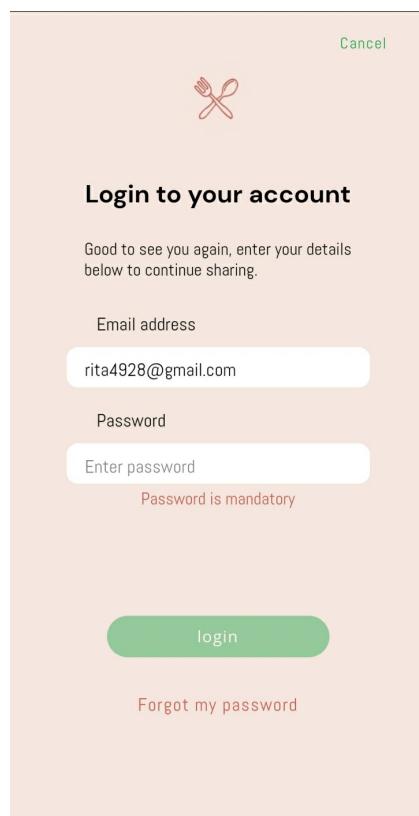


Figura 5.7: Página de login

Caso o utilizador não se recorde da palavra passe que inseriu, poderá alterar a sua palavra passe acedendo à página Forgotpassword onde insere o seu email, sendo-lhe enviado um email com o link onde poderá então redefinir a sua palavra passe.

Este processo pode ser visualizado na Figura 5.8.

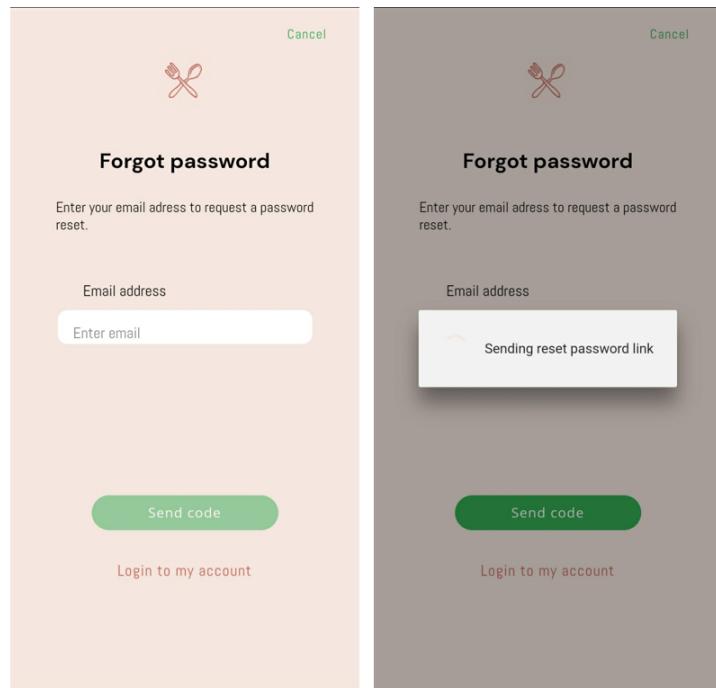


Figura 5.8: Sequência para reset da password

Erros de autenticação

Ao efetuar o *login* na aplicação são detetados alguns erros que possam decorrer durante esta operação, nomeadamente: o endereço de email não foi verificado; o endereço de email inserido é inválido; o utilizador com as credenciais inseridas foi desativado (operação que pode ser efetuada pelo administrador do serviço de autenticação); não existe nenhum utilizador com o email especificado; a password inserida está incorreta.

Nas Figuras 5.9, 5.10 e 5.11 podemos ver alguns dos erros resultantes do *login* mal sucedido.

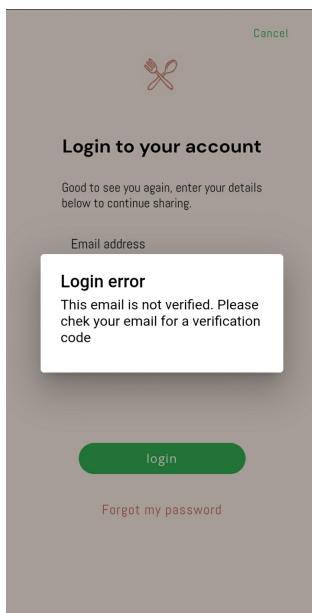


Figura 5.9: Email não verificado

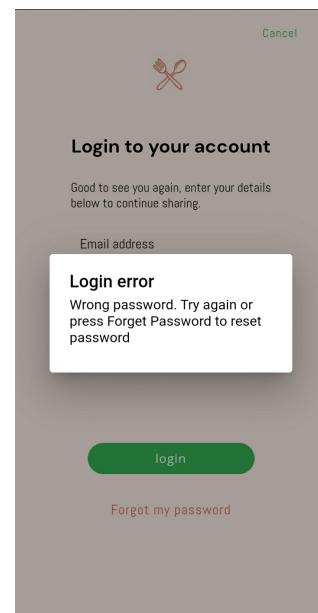


Figura 5.10: Password incorreta

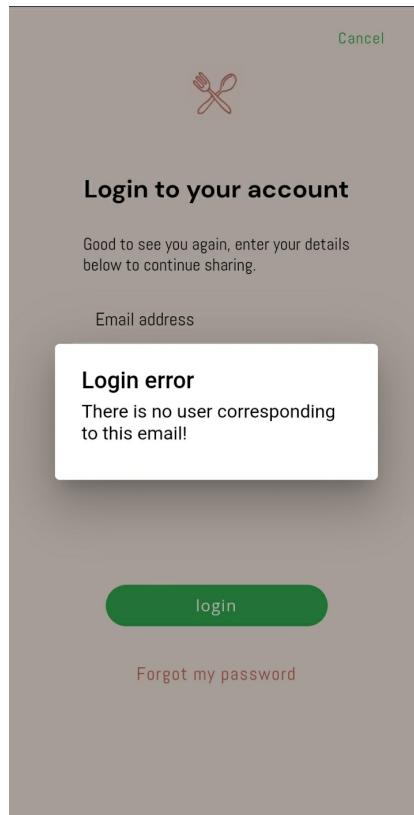


Figura 5.11: Email não registado

A aplicação suporta o *login* automático após a autenticação inicial, guardando as credenciais na memória do dispositivo. Para verificar se existe alguma sessão iniciada no dispositivo é utilizado o método `checkSession()` referido na Tabela 5.6. Se existir, o utilizador é redirecionado para a página inicial da aplicação - Homepage -, caso contrário, para a página de *login*.

5.2.5 Módulos da Aplicação

Os módulos da aplicação consistem em todas as páginas a que o utilizador pode aceder durante a utilização da aplicação.

Será feita uma breve análise e demonstração de cada uma destas páginas, assim como das funções que desempenham no contexto da aplicação.

Páginas gerais

As páginas gerais consistem nas páginas que não são específicas a nenhuma entidade da aplicação, contendo informação sobre diversas componentes.

- WelcomePage

A página de Welcome, referida na Figura 5.12, é a primeira página com a qual o utilizador interage caso não tenha efetuado *login* prévio na aplicação.

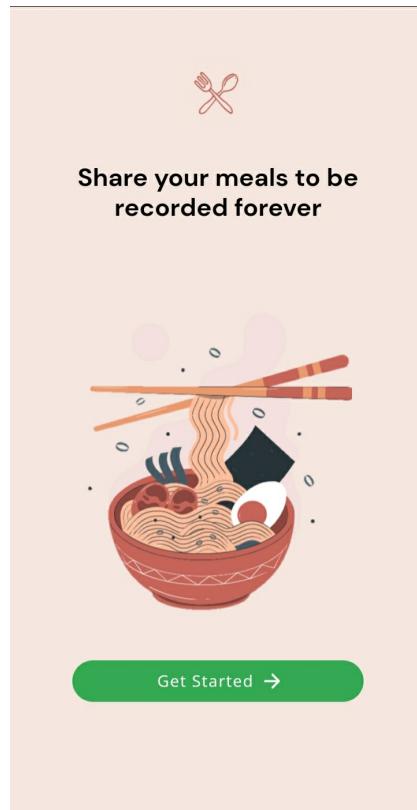


Figura 5.12: Página Welcome

Ao pressionar no botão "Get Started" é redirecionado para a página onde poderá efetuar o *login* ou o registo.

- ConnectPage

Na página de Connect o utilizador poderá então seguir para as páginas de *login* (Figura 5.7), de registo (Figura 5.6), ou de Forgotpassword (Figura 5.8).

Esta página está ilustrada na Figura 5.13



Figura 5.13: Página Connect

- HomePage

A Homepage é a página principal da aplicação, Figura 5.14, sendo onde o utilizador pode visualizar todas as receitas registadas na aplicação e aceder à funcionalidade de filtros e pesquisa de receitas.

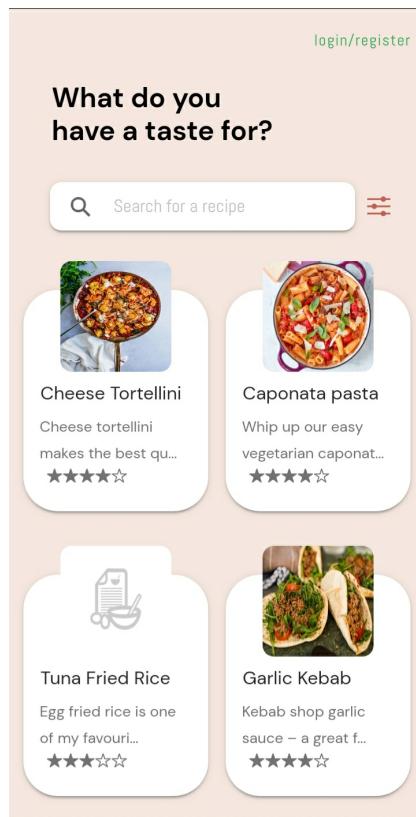


Figura 5.14: Homepage sem sessão iniciada

Caso o utilizador esteja autenticado na aplicação, a partir desta página poderá aceder à sua página de utilizador. Através do *footer* pode ainda aceder a uma página onde poderá adicionar uma receita à aplicação (pressionando o botão vermelho), ou aceder às páginas de favoritos e da lista de compras, através dos ícones do coração e do carrinho de compras, respetivamente. Na Figura 5.15, pode ver o aspeto da Homepage para um utilizador com a sessão iniciada.

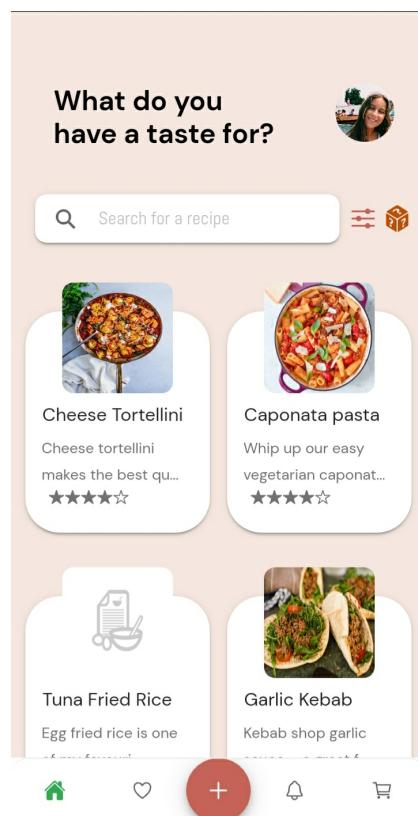


Figura 5.15: Homepage com sessão iniciada

Páginas de utilizador

As páginas de utilizador correspondem às que só podem ser acedidas por utilizadores autenticados, não estando disponíveis a convidados.

Nestas páginas o utilizador poderá aceder e fazer alterações ao seu perfil.

- Userpage

Na página de utilizador, ilustrada pela Figura 5.16, é disponibilizada informação acerca de quantos seguidores e quantas pessoas são seguidas pelo utilizador. Ao pressionar num destes parâmetros é redirecionado para a página de seguidores onde poderá ver quem o segue e quem segue.

Pode também visualizar todas as receitas por si publicadas e seu respetivo número, efetuar o *logout* ou alterar a foto de perfil, pressionando o ícone do lápis verde.

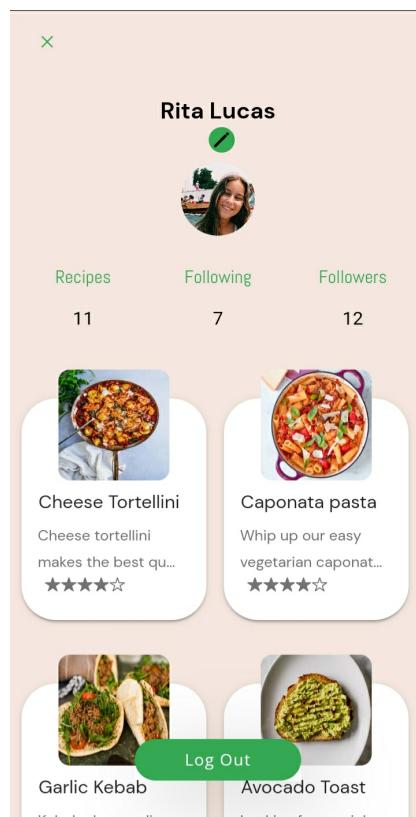


Figura 5.16: Página do utilizador

- FollowersPage

Na página de seguidores são apresentadas duas listas ao utilizador com as fotos e nomes dos utilizadores pertencentes à sua lista de seguidores e à lista de utilizadores que segue. Um exemplo desta página é observável na Figura 5.17.

Pode aceder a cada uma das listas pressionando as secções Followers e Following.

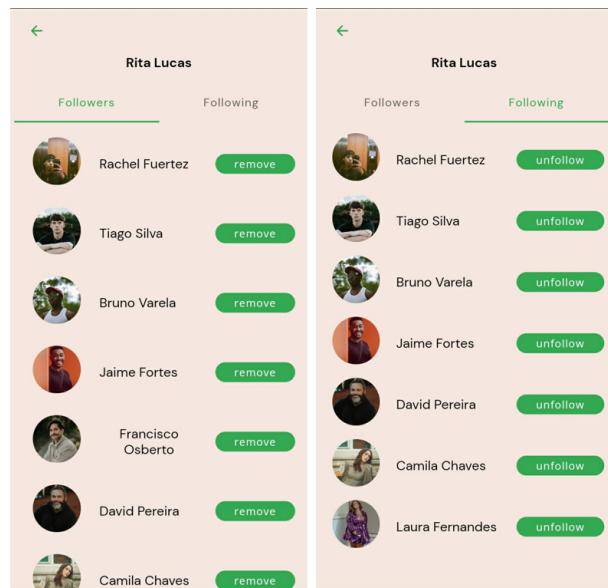


Figura 5.17: Página Followers

Se pressionar o botão ”remove”, na secção dos Followers, irá remover o seguidor da lista de seguidores do utilizador autenticado.

Se pressionar o botão ”unfollow”, na secção de Following, irá remover o utilizador autenticado da lista de seguidores do utilizador.

Se pressionar a imagem ou nome de uns dos utilizadores listados irá redirecionar o utilizador para a página desse utilizador, podendo optar por seguir ou deixar de seguir o utilizador pressionando o botão acima da sua fotografia de perfil.

Na Figura 5.18 é visível a página de utilizador da Rachel Fuertez.

Numa pequena nota é de referir que, como este utilizador não adicionou fotografias às suas receitas, estas aparecem com a fotografia por

omissão. Caso pretenda, pode ir à página das suas receitas e proceder à adição de uma foto.

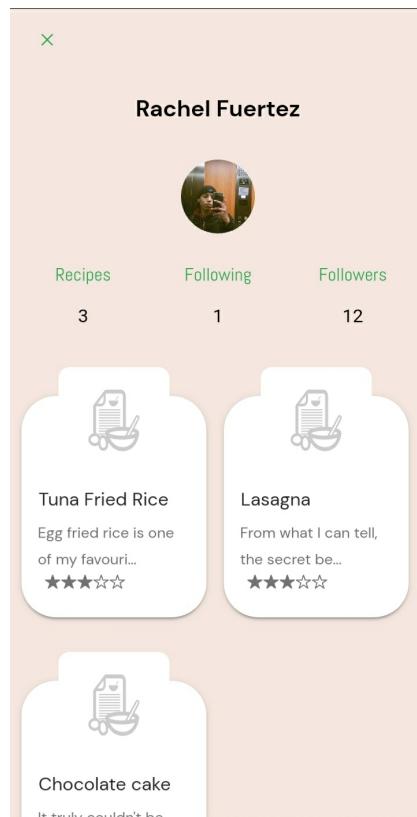


Figura 5.18: Página de utilizador da Rachel Fuertez

- Favoritespage

Na página de favoritos, ilustrada pela Figura 5.19, o utilizador pode aceder a todas as receitas que adicionou à sua lista de favoritos.

Ao pressionar num dos *cards* presentes no ecrã é redirecionado para a página da receita que pressionou.

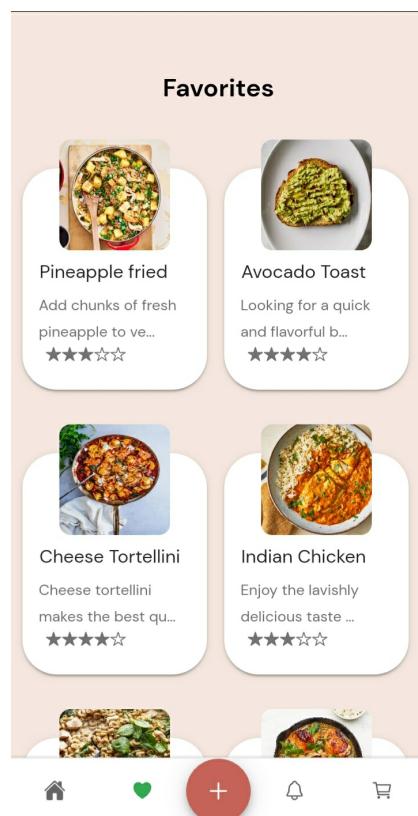


Figura 5.19: Página Favorites

- ShoppinglistPage

De modo semelhante à página de favoritos, na página da lista de compras, Figura 5.20, o utilizador também poderá aceder a uma listagem com todas as receitas que já adicionou à sua lista de compras, com o detalhe adicional de ter um sistema de *checkboxes*, onde pode registar quais os ingredientes que já comprou.

Para remover uma receita da sua lista de compras pode fazer *swipe* para a esquerda do *card* e este será removido.



Figura 5.20: Página Shoppinglist

Páginas da receita

- Recipepage

A página da receita, Figura 5.21, é uma das páginas mais importantes da aplicação, pois contém toda a informação relativa a uma receita.

Nesta página o utilizador pode aceder às instruções para cozinhar a receita, aos ingredientes constituintes, às porções e ao tempo de preparação.

No canto superior direito é apresentado um ícone com a imagem do autor da receita que, quando pressionado, redireciona o utilizador para a página do autor.

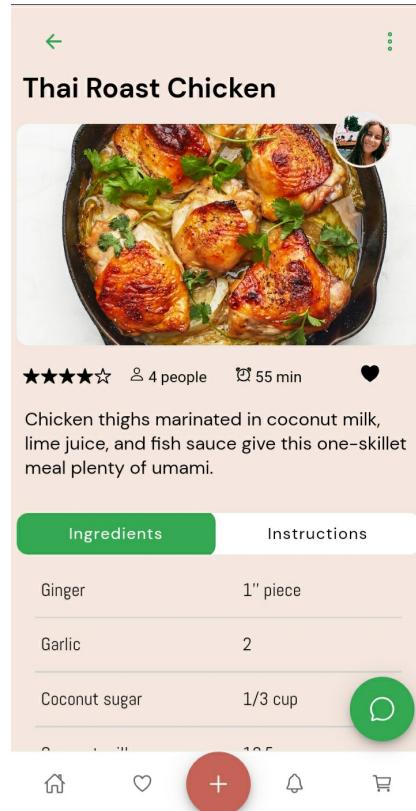


Figura 5.21: Página Recipe

Pressionando o botão verde com o ícone de uma *chat bubble*, tem acesso aos comentários realizados por outros utilizadores na receita e, caso seja um utilizador autenticado, pode ainda publicar um comentário

seu. Esta situação é ilustrada na Figura 5.22, onde o utilizador Rachel Fuertez publica o comentário "Best recipe I've ever had".

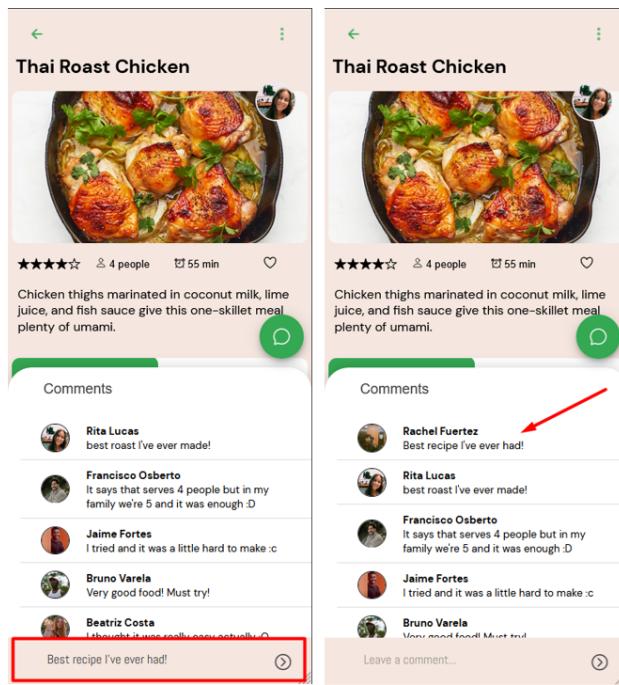


Figura 5.22: Sequência para publicação de um comentário

Tem também acesso ao botão que lhe permite adicionar ou remover uma receita dos favoritos. Para o efeito, tem apenas de pressionar o ícone em forma de coração (abaixo da imagem da receita), que, quando preenchido, indica que a receita está presente na lista de favoritos. Quando vazio, ilustra o caso oposto.

No menu formado por *bullet points*, no canto superior direito, o utilizador tem acesso a um *drop down* menu onde pode adicionar a receita à lista de compras pressionando a opção "Add to Shopping list", ou, caso seja o autor da receita, aceder à página de edição. O cenário descrito é ilustrado pela Figura 5.23, com o utilizador Rachel Fuertez a visualizar a página de uma das suas receitas.

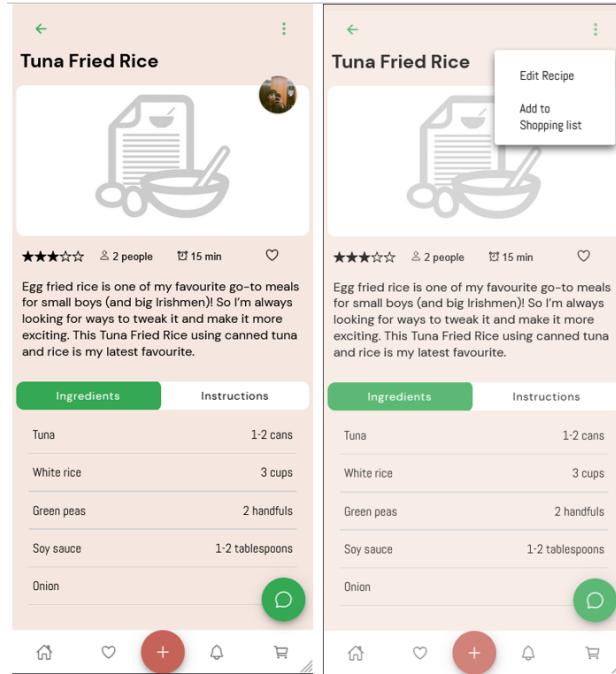


Figura 5.23: Menu na página de receitas

- Addrecipepage

Na página de publicação de uma receita o utilizador pode adicionar a sua própria receita à aplicação. Para aceder a esta funcionalidade deve ter a sessão iniciada na aplicação.

Nesta página pode especificar toda a informação relativa à receita que pretende publicar, tal como, o tipo de dieta, o título, ingredientes e imagens que serão mostradas na página da receita.

A Figura 5.24, ilustra a página descrita.

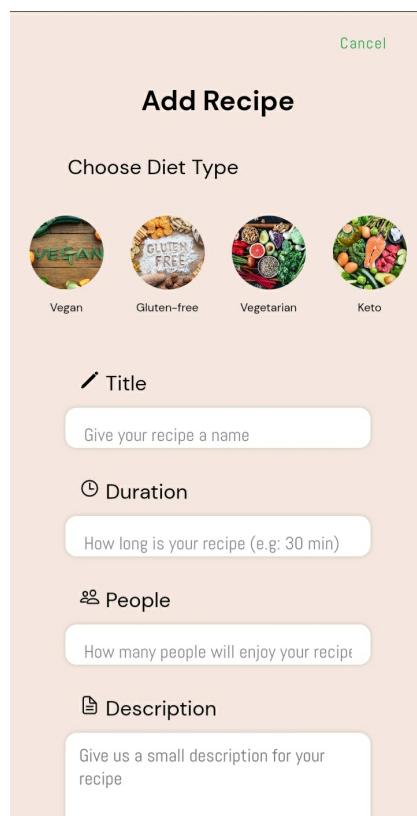


Figura 5.24: Página Addrecipe

- Editrecipepage

A página de edição de receita, Figura 5.25, permite ao utilizador alterar informação de uma receita que tenha publicado.

Esta só pode ser acedida pelo utilizador autor da receita.

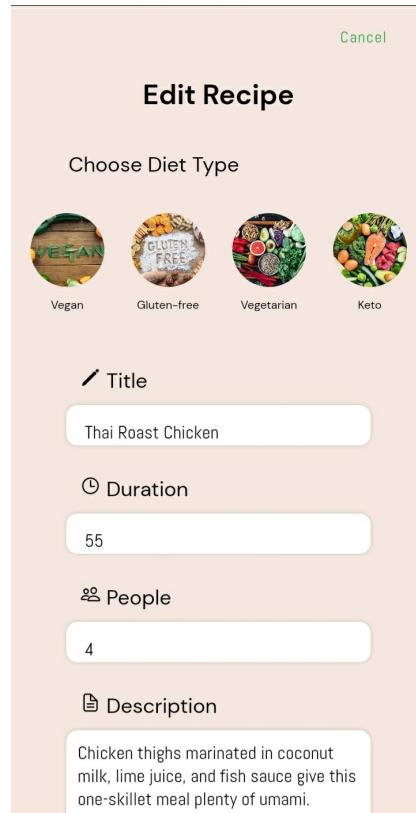


Figura 5.25: Página Editrecipe

Capítulo 6

Validação e Testes

De forma a testar a aplicação desenvolvida foram realizados testes de usabilidade em finais de junho e início de julho, ou seja, na fase final da implementação da aplicação, no qual participaram 21 utilizadores.

6.1 Contexto

O primeiro passo para a realização dos testes foi definir quais os objetivos do mesmo, neste caso, quais as respostas que pretendemos obter. Através de um questionário, construído no Google Forms, foi feita uma breve descrição para introduzir o utilizador à aplicação. As respostas a este devem ser específicas. Assim, os objetivos definidos para os testes de usabilidade foram:

- se o utilizador considera a aplicação **útil** e se ponderaria **utilizá-la no futuro**.
- entender se os utilizadores consideraram as **funcionalidades intuitivas e interessantes**.
- se a componentes de interação entre utilizadores era **útil e interessante**.
- se a aplicação é **visualmente apelativa e intuitiva** e se os **conteúdos e funcionalidades estão bem integrados**.

Começou-se por dar alguns minutos aos utilizadores para explorarem a aplicação, antes de procederem à avaliação da mesma. Após experimenta-

rem a aplicação, os participantes responderam a um questionário de modo a caracterizarem a experiência.

O questionário está disponível em: Questionário Google Forms.

6.2 Participantes

Os utilizadores que tomaram parte desta avaliação foram maioritariamente familiares, amigos e colegas.

Neste teste participaram 11 pessoas do sexo masculino e 10 do sexo feminino, Figura 6.1.

1. Indique o seu género

21 respostas

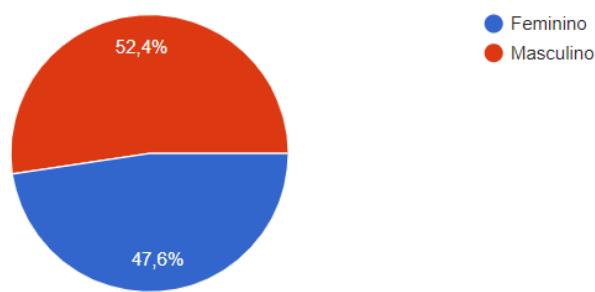


Figura 6.1: Gráfico correspondente ao género

Quanto à faixa etária, a mais popular foi entre os 18 e 25 anos (jovens adultos). No entanto, também pudemos verificar alguma audiência por parte de pessoas com idade superior a 40 anos e registámos alguns utilizadores menores de idade, Figura 6.2.

2. Em que intervalo de idade se encontra?

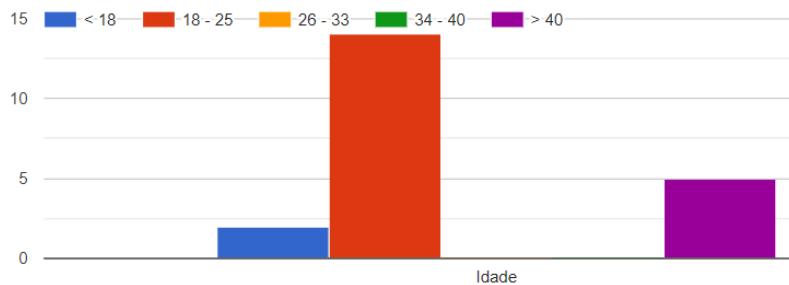


Figura 6.2: Gráfico correspondente à faixa etária

6.3 Análise e Discussão de Resultados

No formulário de usabilidade, de entre outras questões, foram utilizadas as questões do SUS. Com estas questões foi possível determinar o *score* da nossa aplicação [5]. Fazendo o cálculo necessário de acordo com as respostas dos participantes ao questionário, obtivemos uma pontuação de 85 o que, através da Figura 6.3, podemos classificar como excelente.

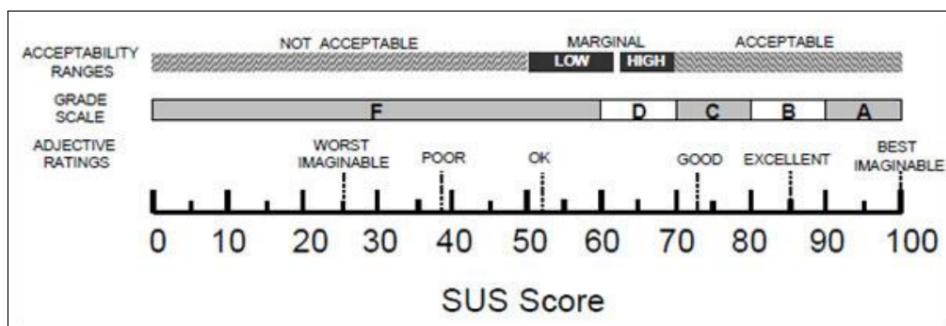


Figura 6.3: Escala de pontuação do questionário SUS

Tendo sido perguntado se os utilizadores consideravam útil uma aplicação na qual se partilham receitas, Questão 23 (Figura C.21), 76,2% concordou. Com este resultado concluímos que a aplicação é proveitosa e conveniente para a maioria dos participantes. Todos os gráficos com as respostas ao questionário estão disponíveis no Apêndice C.

As tarefas que pedimos para os utilizadores realizarem foram, em geral, todas bem sucedidas, visto que, das 7 tarefas, os participantes apenas mostraram dificuldades em duas delas (Tarefa 5: Adicionar uma receita e Tarefa 6: Pesquisar uma receita por ingredientes). No entanto, as tarefas 5 e 6 eram as mais importantes e as que demonstraram resultados menos bons, embora positivos, o que revela dificuldades por parte dos utilizadores a completá-las e, por isso, indicam que as tarefas precisam de ser melhoradas.

Capítulo 7

Conclusões e Trabalho Futuro

Este capítulo encontra-se dividido em duas secções: na secção 7.1 estão incorporadas as conclusões referentes ao desenvolvimento das funcionalidades já integradas na aplicação, enquanto que na secção 7.2 estão descritas as ideias e melhorias mais relevantes para implementar futuramente.

7.1 Conclusões

Finalizada a elaboração do projeto, conclui-se que os principais objetivos foram cumpridos, nomeadamente o desenvolvimento de uma aplicação que facilita o processo de decisão e oferece resultados adequados ao utilizador na hora de preparar uma refeição. Assim sendo, foi implementado um sistema de pesquisa que permite obter resultados personalizados e enquadrados na necessidade do utilizador, que, caso não deseje utilizar a aplicação como complemento para o seu processo culinário, pode utilizá-la para explorar novas receitas e aumentar os seus conhecimentos na área.

Foi concretizado com sucesso o objetivo de criar uma aplicação que permitisse uma vertente social, possibilitando que o utilizador interaja de forma dinâmica e direta com os outros utilizadores da aplicação.

A aplicação iMeal foi avaliada com 21 utilizadores relativamente à sua usabilidade. Apesar de algumas dificuldades apresentadas por parte dos participantes ao concluir as tarefas pedidas, obteve uma pontuação de 85 na escala utilizada no questionário SUS.

Resumindo, foi desenvolvido um sistema com um servidor Firebase que fornece uma base de dados da aplicação, um serviço sofisticado de auten-

ticação e um serviço de armazenamento de ficheiros que permitiram a comunicação com clientes Android.

Os serviços utilizados foram imprescindíveis para o desenvolvimento de uma aplicação com um sistema de pesquisas, alteração de dados e interação entre utilizadores, rápido e consistente.

7.2 Trabalho futuro

Durante o desenvolvimento da aplicação foram surgindo algumas ideias que não foram implementadas, mas que seriam funcionalidades que, para além de interessantes, iriam adicionar novas características à experiência dos utilizadores.

- Notificações

A adição de notificações foi um sistema ponderado como parte integrante da aplicação.

Seria utilizado para notificar o utilizador de novas receitas publicadas pelos utilizadores que segue ou notificar o utilizador acerca de novos seguidores.

- Cooking mode, tutorial em vídeo e recomendações

Na página de uma receita foram ponderadas duas funcionalidades cujo intuito seria ajudar o utilizador a cozinhar a receita que estivesse a visualizar, sendo estas a possibilidade de adicionar vídeos tutoriais e uma funcionalidade de *Cooking Mode*, em que o utilizador poderia seguir as instruções passo a passo, passando para o passo seguinte ou quando carregasse num botão ou quando colocasse a mão por cima do ecrã.

A funcionalidade de recomendações permitiria ao utilizador, dentro da página de uma receita, obter recomendações de receitas semelhantes.

- Preferências e alergias

Com o intuito de tornar a aplicação ainda mais útil e acessível para um maior número de utilizadores, seria interessante para trabalho futuro adicionar uma vertente, na página do utilizador, que lhe permitisse especificar as suas intolerâncias e/ou alergias alimentares.

No random meal generator também só seriam geradas receitas "random" que se enquadrasssem neste perfil.

- **Planeamento da Semana**

Durante os testes de usabilidade notou-se que os utilizadores sugeriram como funcionalidade adicional um planeamento semanal, que lhes permitisse definir um plano de refeições para a semana toda e assim definir também uma lista de compras para todos os ingredientes necessários.

Apêndice A

Modelo da base de dados

The screenshot shows a MongoDB interface with three main sections. On the left, under '+ Iniciar coleção', there is a list of collections: comments, ingredients, ratings, recipes, shoppinglist, and users. The 'comments' collection is highlighted with a grey background. In the center, under '+ Adicionar documento', there is a list of document IDs: 1PWsABtA3UPwz4hPwfFrF, L1z8qY1cQ3n2vfV7s8lp, and huK1iCYnTBo1Wkx046a7. The third document ID is highlighted with a grey background. On the right, under '+ Iniciar coleção', there is a single collection named 'comment'. Below it, under '+ Adicionar campo', there is a button labeled '+ Adicionar campo'.

Figura A.1: Coleção comments

The screenshot shows a MongoDB interface with two main sections. On the left, under '+ Iniciar coleção', there is a list of collections: comment. The 'comment' collection is highlighted with a grey background. Below it, under '+ Adicionar campo', there is a button labeled '+ Adicionar campo'. In the center, under '+ Adicionar documento', there is a list of document IDs: 1GjH11V9LooWhhzTwQ2, D89crAGMEQwAZWF7K0sl, OPMN7DHgze5oRhrW4jI3, WPd7MVLiUH1oDRoVe8iT, Yh38f26tkbCxYfY0lHC9, ZBQz6ZjMaLkHf6e33F28, er0sH0c6Nb2uuTPKnv1C, and iS8tI6h24HS4uvVVc3Gn.

Figura A.2: Subcoleção comment

Iniciar coleção

- comments
- ingredients**
- ratings
- recipes
- shoppinglist
- users

Adicionar documento

- 09ILNOQXDdZ2pXfVuQJE
- 0b4U4R3MZHqCQcdmzTyM
- 0tRlUlplFiTY0g0bsqmd
- 1Hj1QV5zdjFDPcs65Y0Z
- 1RucMTRKVfuknvgceXXZ
- 1WMKB5UjyjLCpJuoWrH2
- 1bFyhzOHwsKAFCLB3kyX
- 1gviQIHxME3vFC0e1FS
- 1wpqBXvwxxUegr0KSv0DX
- 1yjNyrTWRcaA00aauxxf
- 2Rmq4bTvL5I76UAwtfwP
- 2exbCJJZ4E9UhKUiRyTX
- 2v3QiHzeJ0OiIHsddSkr

Iniciar coleção

Adicionar campo

- name : "Chicken breast"
- recipes
- 0 "wZEHE8vY8CeA8hORV7Yi"
- 1 "xGhVoAKbYmpkaPRvWy00"

Figura A.3: Coleção ingredients

Iniciar coleção

- comments
- ingredients
- ratings**
- recipes
- shoppinglist
- users

Adicionar documento

- 1PWsABtA3UPwz4hPwfP_7pBxFbDFZZ
- 1PWsABtA3UPwz4hPwfP_s8Gq5a4pMZc
- 44a0Kqhmx2tCzUuvvQ3_s8Gq5a4pMZc
- HUsNP08LwzzDkEl8Pg5q_DRyrcsj34D3e
- HUsNP08LwzzDkEl8Pg5q_s8Gq5a4pMZc
- L1z8qY1cQ3n2vfV7s8lp_3tsGg2X30GX
- L1z8qY1cQ3n2vfV7s8lp_qnems47TGAe
- LqYitjZcfqm2nauZsGqZ_jNe6qoyBiPe
- XzeEFwZT1WN85zbsaVwP_9nVCre5ntgS
- XzeEFwZT1WN85zbsaVwP_tDah7TNN6qW
- ZCZvPgyKAh1AIxnWk1ph_DRyrcsj34D3e
- ZCZvPgyKAh1AIxnWk1ph_ZejG0c7dudc
- cqytFQyuTXXtCzxADGm5q_ZejG0c7dudc

Iniciar coleção

Adicionar campo

- rating: 5
- recipeid: "1PWsABtA3UPwz4hPwfP"
- userid: "s8Gq5a4pMZdLoCOZp4DFWPLp0ml2"

Figura A.4: Coleção ratings

Iniciar coleção

- comments
- ingredients
- ratings
- recipes**
- shoppinglist
- users

Adicionar documento

1PWsABtA3UPwz4hPwfrP	>
44aQKqhmxx2tCzUvvl3	
HUsNP08LwzzDkEl8Pg5q	
L1z8qY1cQ3n2vfV7s8lp	
LqYitjZcfqm2nauZsGqZ	
XzeEFwZT1WN85zbsaVwP	
ZCZvPgyKAhlAIxnWk1ph	
cqytFQyuXTxCzxADGm5q	
gzhGdcuoIn4ak0eDmuUA	
huK1iCYnTB01Wkx046aT	
pqYYrrCs5QvRVK01muCR	
rVEVmtnbUY6qZVnTfBZ	
wZEHE8vY8CeA8h0RV7Yi	

Iniciar coleção

Adicionar campo

```

description: "Cheese tortellini makes the best quick dinner recipe! Serve it
up in a creamy marinara sauce flavoured with goat cheese and
basil."
dietType: "Vegetarian"
duration: "15"
images: ["https://firebasestorage...."]
ingredients: [{"ingunit: "12 to 16 ounce..."}]
instructions: ["Boil the tortellini acco..."]
portion: "4"
title: "Cheese Tortellini"
userId: "7pBxFbDFZzBbYebfsA06DEgjQAE3"

```

Figura A.5: Coleção recipes

Iniciar coleção

- comments
- ingredients
- ratings
- recipes
- shoppinglist**
- users

Adicionar documento

7pBxFbDFZzBbYebfsA06DEgjQAE3	>
jNe6q0yBiPek0rdZtCjFVGSuovk1	>
s8Gq5a4pMZdLoCOZp4DFWPLp0mI2	

Iniciar coleção

Adicionar campo

- recipes**
 - 0 "1PWsABtA3UPwz4hPwfrP"
 - 1 "HUsNP08LwzzDkEl8Pg5q"

Figura A.6: Coleção shoppinglist

Iniciar coleção

- comments
- ingredients
- ratings
- recipes
- shoppinglist
- users**

Adicionar documento

3tsGg2X30GXecPqCPZ086qBfGmE2	>
7pBxFbDFZzBbYebfsA06DEgjQAE3	>
9nVCre5ntgSpsUExxAVK5pkysG2	
DRycts34D3awhfjv8gScj0BFANR2	
HyPCPce3hqdlqYtxDPxhfpLapPm1	
NcJdpckudXFV8KhmYl0n0g0cYo2	
P0sKdWQ3EbYvBCtwMKnBhexsBw1	
ZejG0c7dudc03xHmCkX8GG8969e2	
cRLUmqqIt60NbN0oXp3K4p3uUti02	
jNe6q0yBiPek0rdZtCjFVGSuovk1	
jwTjolsqz609tljpBTX02Asfidj1	
qnems47TGaeDHTMkrVjoPumgr9j1	
s8Gq5a4pMZdLoCOZp4DFWPLp0mI2	

Iniciar coleção

Adicionar campo

```

email: "rita4928@gmail.com"
favorites: ["72ZmmmsgZo8a4uiWyP0N", ...]
followers: [s8Gq5a4pMZdLoCOZp4DFWPLp...]
following: [s8Gq5a4pMZdLoCOZp4DFWPLp...]
name: "Rita Lucas"
profileUrl: "https://firebasestorage.googleapis.com/v0/b/imeal-47e6b.appspot.com/o/uploads%2F7pBxFbDFZzBbYebfsA06DEgjQalt=media&token=6cfe5b8d-7b5d-4a04-bfad-b06c8478f00c"

```

Figura A.7: Coleção users

Apêndice B

Modelo UML dos módulos do Cliente

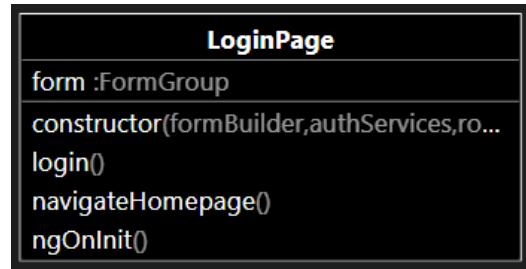


Figura B.1: Diagrama UML do módulo Login

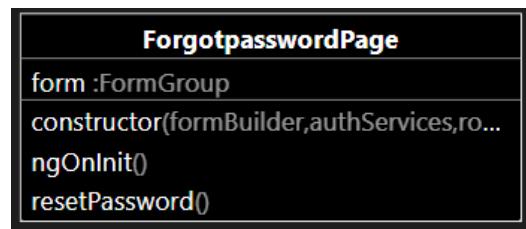


Figura B.2: Diagrama UML do módulo ForgotPassword

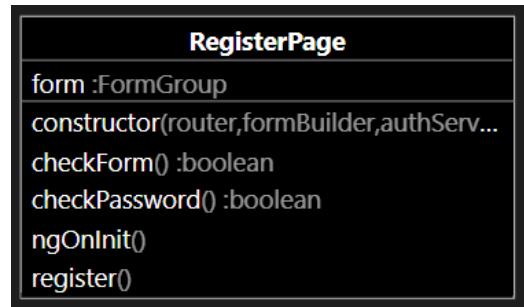


Figura B.3: Diagrama UML do módulo Register

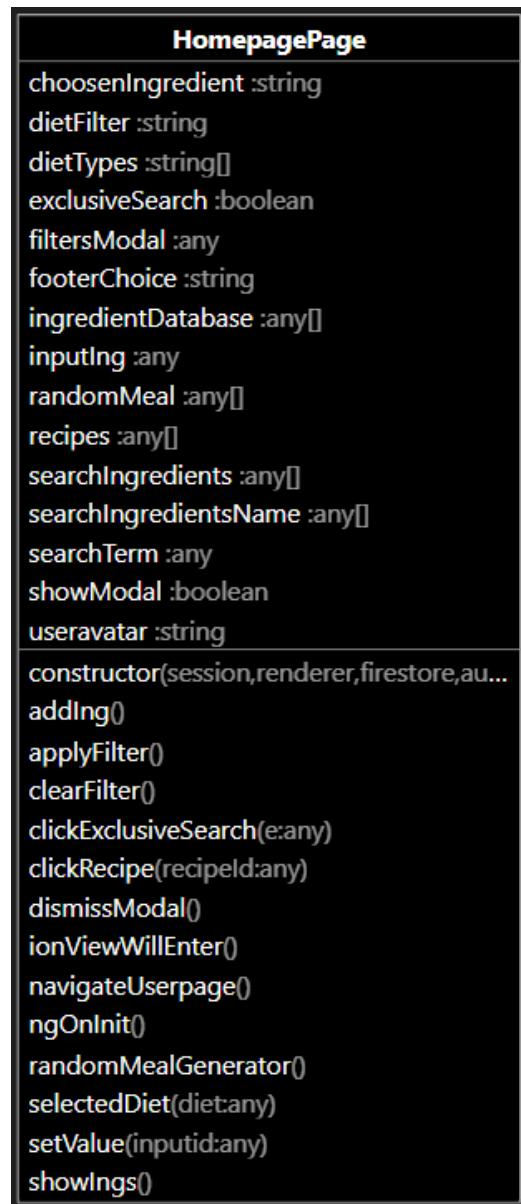


Figura B.4: Diagrama UML do módulo HomePage

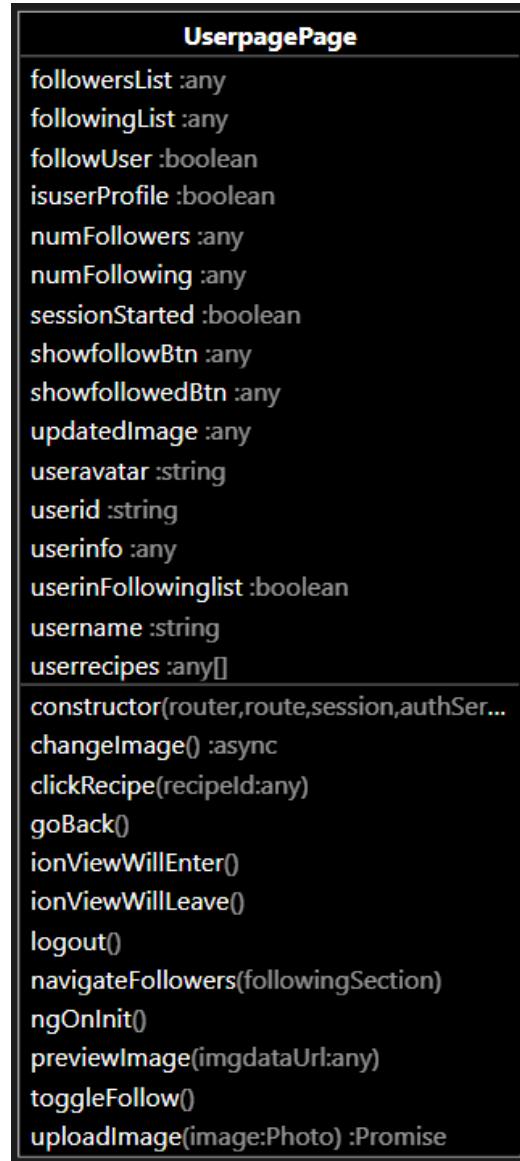


Figura B.5: Diagrama UML do módulo UserPage

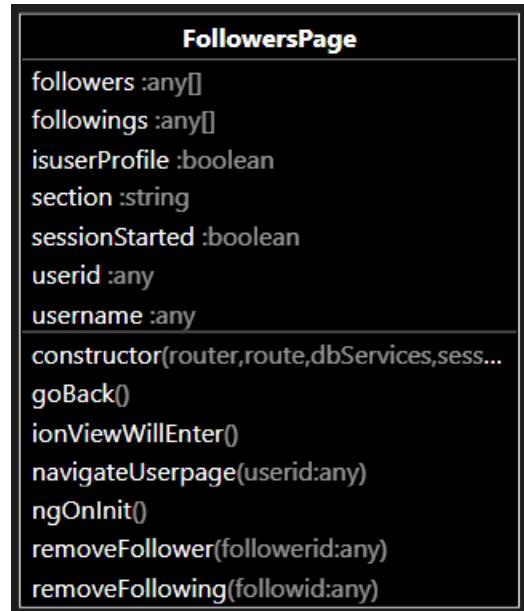


Figura B.6: Diagrama UML do módulo Followers

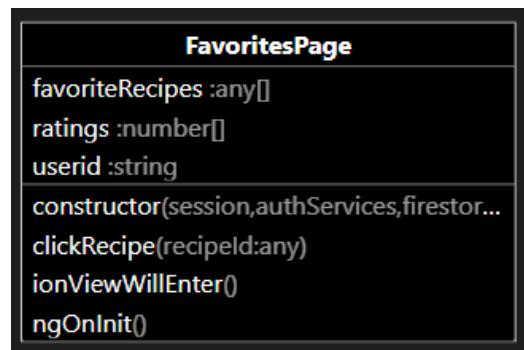


Figura B.7: Diagrama UML do módulo Favorites

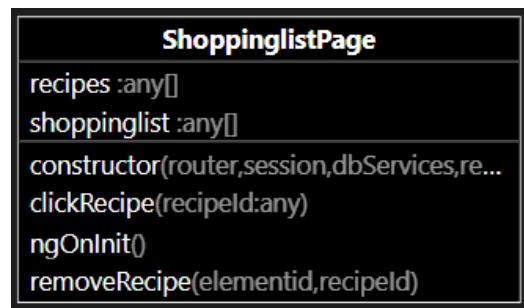


Figura B.8: Diagrama UML do módulo Shopping List

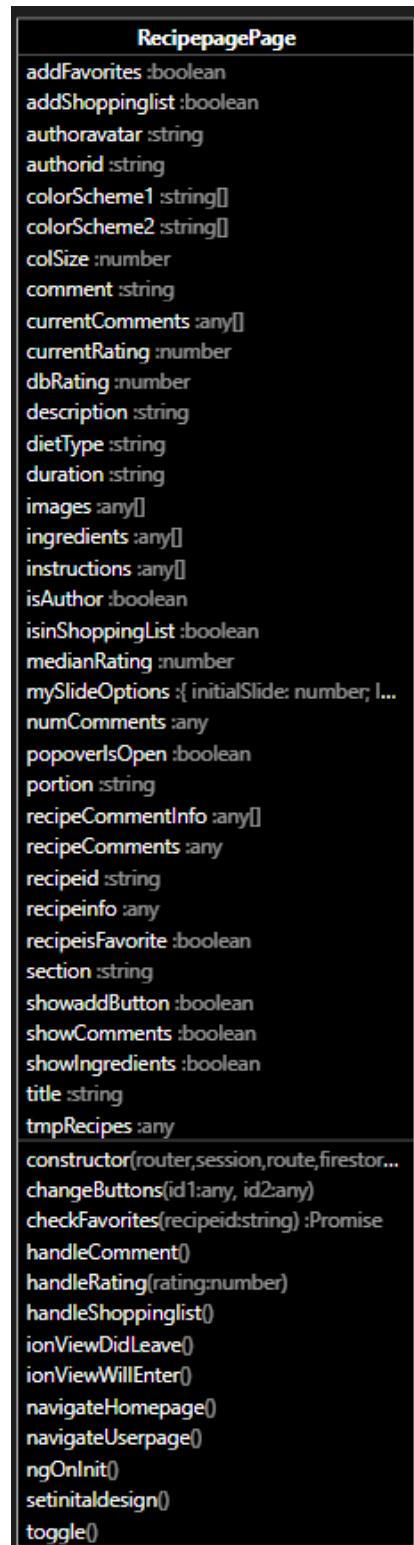


Figura B.9: Diagrama UML do módulo RecipePage

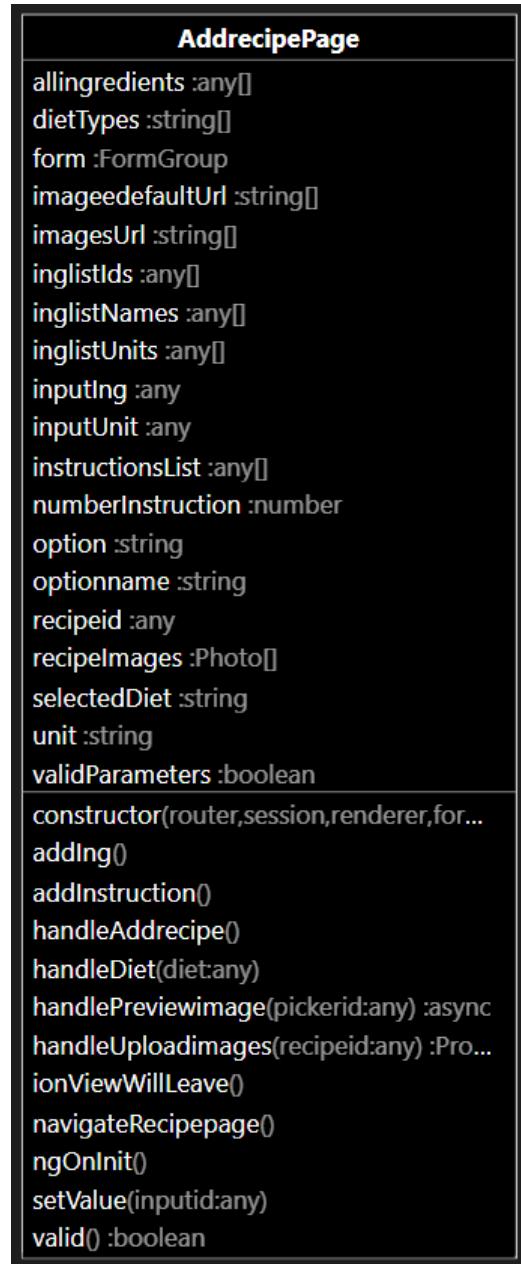


Figura B.10: Diagrama UML do módulo Addrecipe

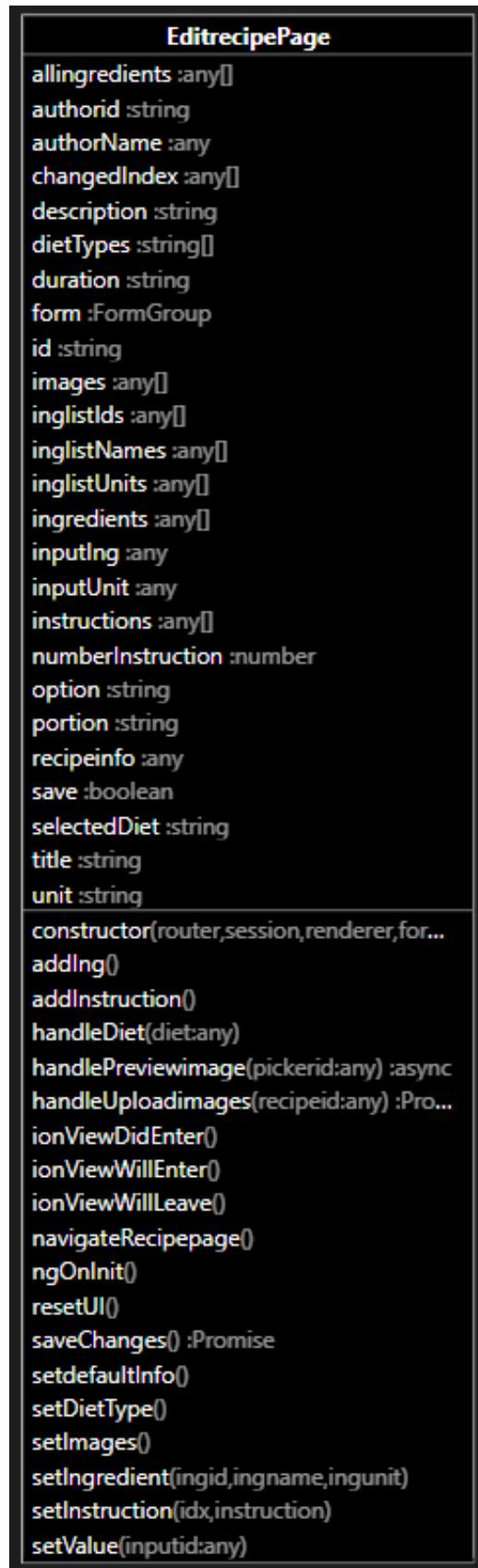


Figura B.11: Diagrama UML do módulo EditRecipe

Apêndice C

Gráficos do questionário

3. Com que frequência cozinha?

21 respostas

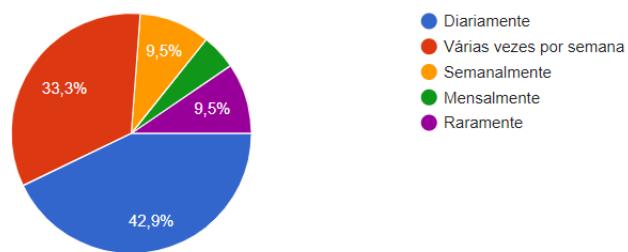


Figura C.1: Gráfico correspondente à frequência com que cozinha

4. Conseguí concluir a tarefa com sucesso.

Copiar

21 respostas

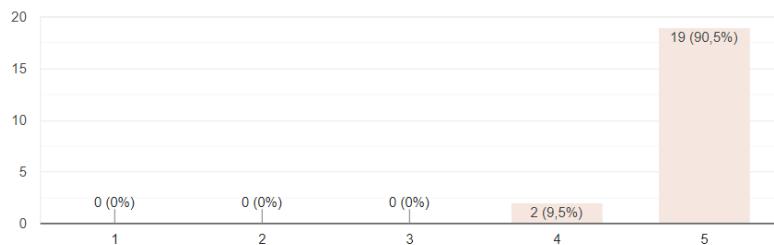


Figura C.2: Gráfico correspondente à Tarefa 1: Criar conta na aplicação e fazer login na aplicação

5. Conseguí concluir a tarefa com sucesso.

21 respostas

 Copiar

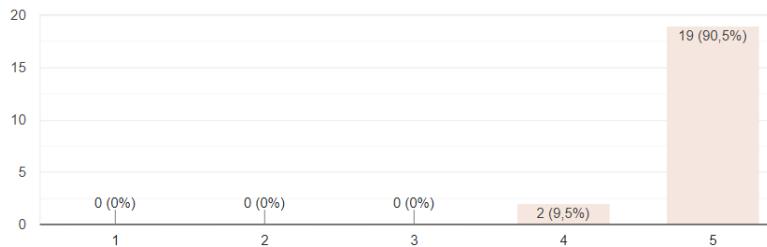


Figura C.3: Gráfico correspondente à Tarefa 2: Visualizar uma receita

6. Conseguí concluir a tarefa com sucesso.

21 respostas

 Copiar

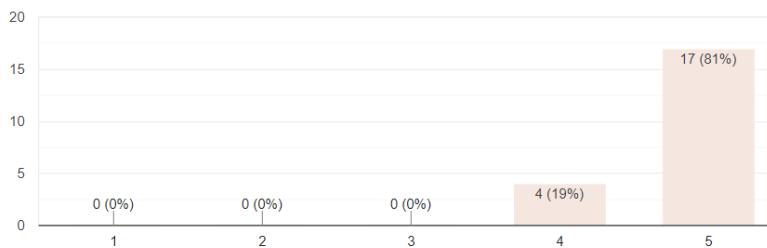


Figura C.4: Gráfico correspondente à Tarefa 3: Classificar uma receita

7. Conseguí concluir a tarefa com sucesso.

21 respostas

 Copiar

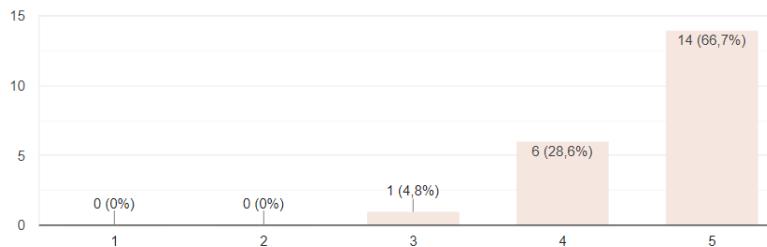


Figura C.5: Gráfico correspondente à Tarefa 4: Seguir outro utilizador de interesse

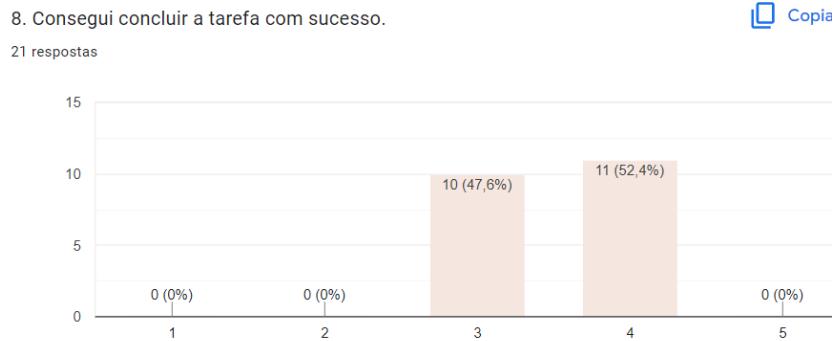


Figura C.6: Gráfico correspondente à Tarefa 5: Adicionar uma receita



Figura C.7: Gráfico correspondente à Tarefa 6: Pesquisar uma receita por ingredientes

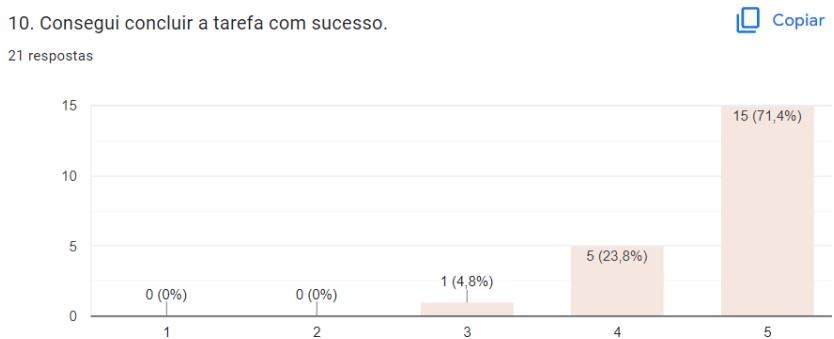


Figura C.8: Gráfico correspondente à Tarefa 7: Adicionar uma receita à lista de compras

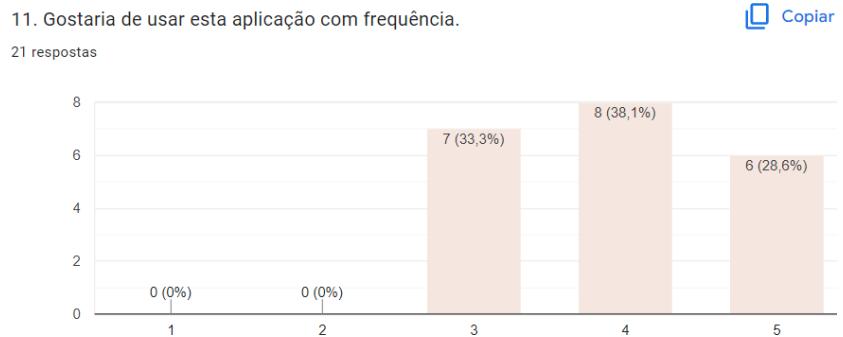


Figura C.9: Gráfico correspondente à frequência com que usaria a aplicação

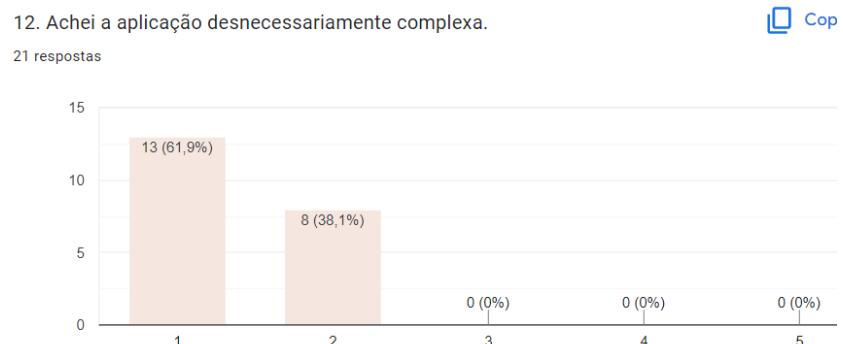


Figura C.10: Gráfico correspondente à complexidade da aplicação

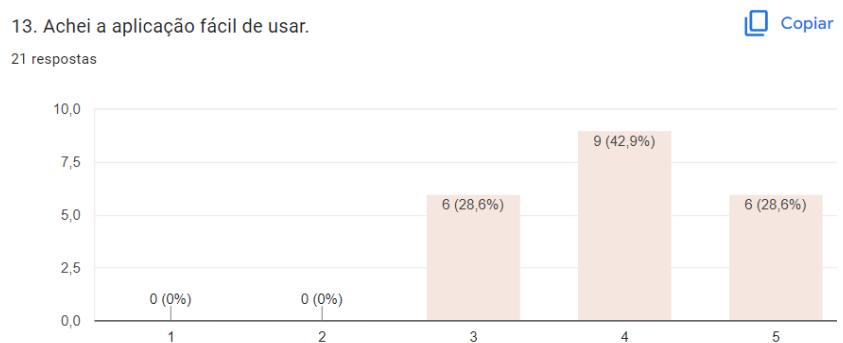


Figura C.11: Gráfico correspondente à facilidade da aplicação

14. Achei que precisaria de ajuda de uma pessoa com conhecimentos técnicos para usar a aplicação.

21 respostas

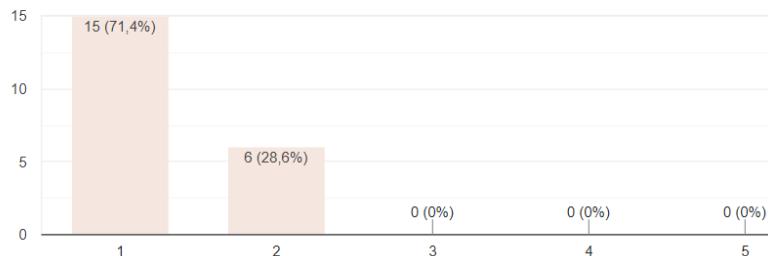


Figura C.12: Gráfico correspondente à necessidade de ajuda para usar a aplicação

15. Achei que as várias funções da aplicação estão muito bem integradas.

Copiar

21 respostas

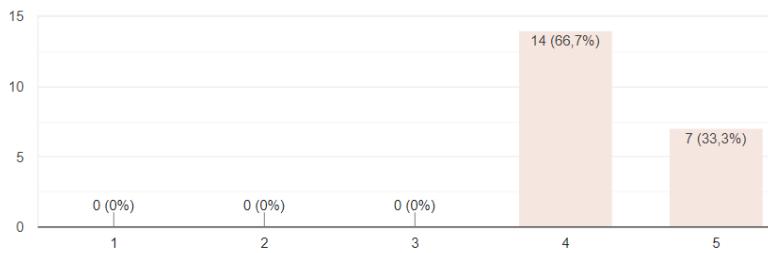


Figura C.13: Gráfico correspondente à integridade das funções da aplicação

16. Achei que a aplicação apresentava muita inconsistência.

Copiar

21 respostas

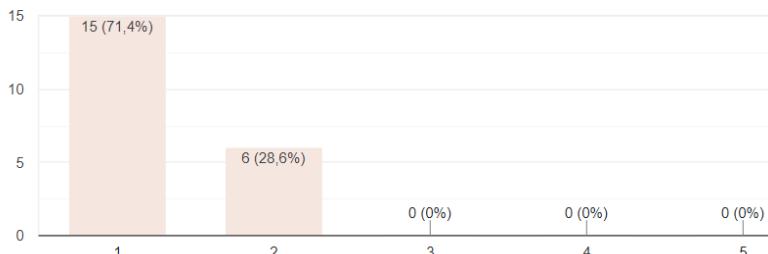


Figura C.14: Gráfico correspondente à inconsistência da aplicação

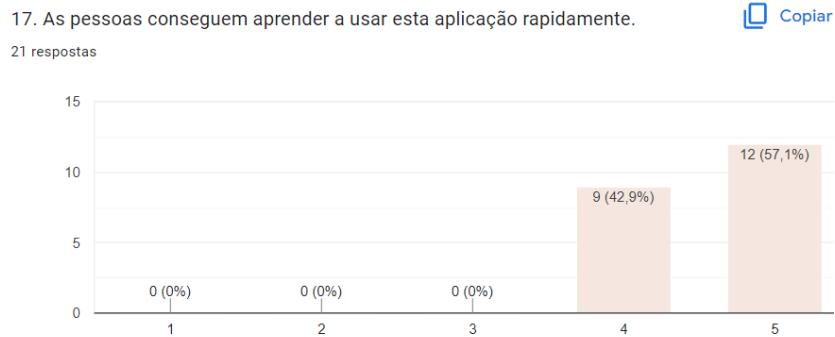


Figura C.15: Gráfico correspondente à facilidade em aprender a utilizar a aplicação

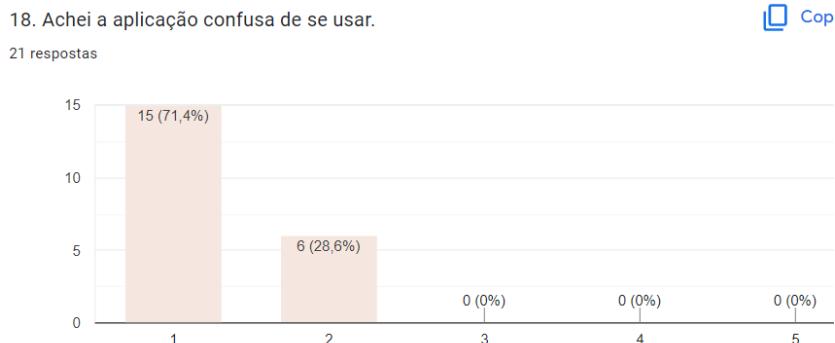


Figura C.16: Gráfico correspondente à confusão a utilizar a aplicação

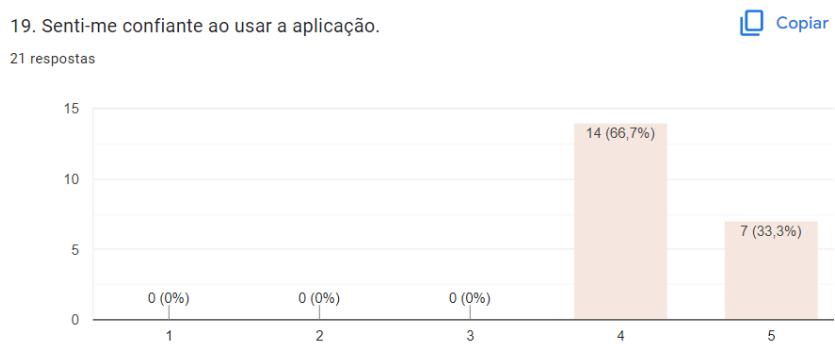


Figura C.17: Gráfico correspondente à confiança ao usar a aplicação

20. Precisei de aprender várias coisas novas antes de conseguir usar a aplicação. [Copiar](#)

21 respostas

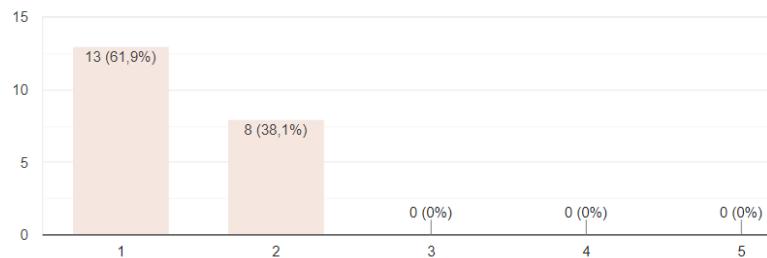


Figura C.18: Gráfico correspondente às coisas novas aprendidas para usar a aplicação

21. A informação é apresentada claramente.

[Copiar](#)

21 respostas

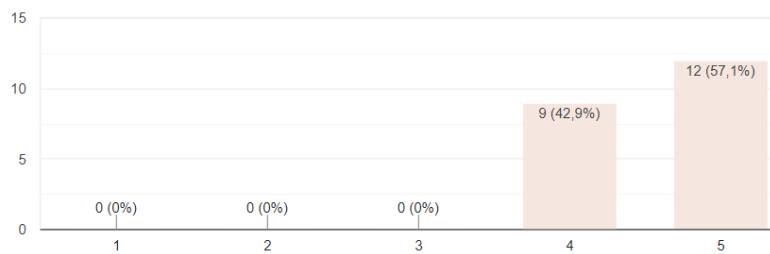


Figura C.19: Gráfico correspondente à clareza da informação

22. Avalie a aplicação relativamente aos seguintes campos

[Copiar](#)

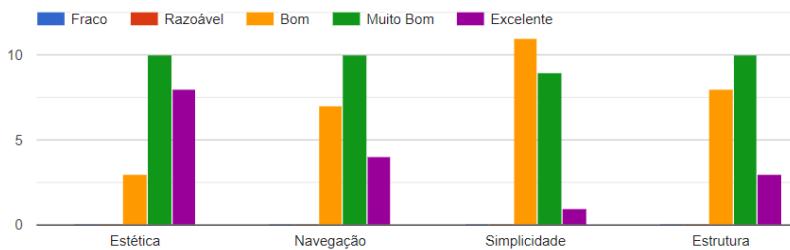


Figura C.20: Gráfico correspondente à avaliação da aplicação em certos campos

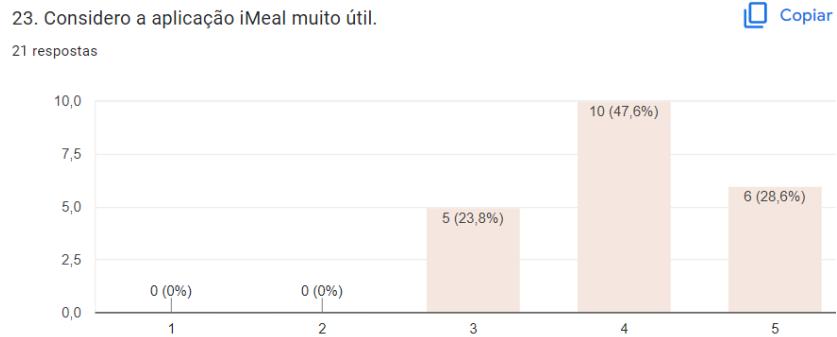


Figura C.21: Gráfico correspondente à utilidade da aplicação

25. Em geral, como classifica a qualidade da aplicação?

21 respostas

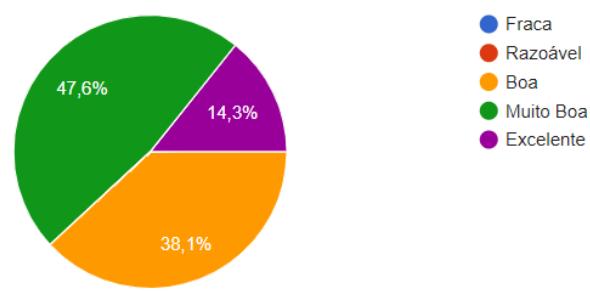


Figura C.22: Gráfico correspondente à classificação geral da aplicação

Bibliografia

- [1] Firebase documentation, <https://firebase.google.com/docs/firestore>
- [2] Cloud Firestore, <https://firebase.google.com/docs/firestore>
- [3] Firebase Authentication, <https://firebase.google.com/docs/auth>
- [4] Cloud Storage, <https://firebase.google.com/docs/storage>
- [5] How to Measure Product Usability with the System Usability Scale (SUS) Score,
<https://uxplanet.org/how-to-measure-product-usability-with-the-system-usability-scale-sus-score-69f3875b858f>
- [6] Ionic documentation, <https://ionicframework.com/docs>
- [7] Firebase Persistence,
<https://firebase.google.com/docs/auth/web/auth-state-persistenceweb-version-8>
- [8] What is firebase?,
<https://medium.com/firebase-developers/what-is-firebase-the-complete-story-abridged-bcc730c5f2c0>
- [9] Angular documentation, <https://angular.io/docs>
- [10] Firebase Authentication tutorial,
<https://www.youtube.com/watch?v=U7RvTTF9dnkt=2039s>

[11] How to Pass Data Between Pages in Ionic Apps using Angular,
<https://www.youtube.com/watch?v=1ZbhOJ5coY4>

[12] How to Upload Files from Ionic to Firebase Storage,
<https://www.youtube.com/watch?v=1VoZ7sqyd30>