

UNIVERSIDADE DO MINHO

DEPARTAMENTO DE INFORMÁTICA

Laboratórios de Informática IV
PL1 - Grupo 1

José Ferreira (A83683)
João Teixeira (A85504)
Maria Silva (A83840)
Miguel Solino (A86435)
Pedro Oliveira (A83762)

5 de julho de 2020

Conteúdo

| | | |
|----------|----------------------------------------------|-----------|
| 1 | Introdução | 4 |
| 1.1 | Motivação e Objetivos | 4 |
| 1.2 | Estrutura do Relatório | 4 |
| 2 | Viabilidade | 5 |
| 3 | Plano de Desenvolvimento | 6 |
| 4 | Levantamento de Requisitos | 8 |
| 4.1 | Requisitos Funcionais | 8 |
| 4.1.1 | Gestão de Utilizador | 8 |
| 4.1.2 | Gestão de Despesas | 8 |
| 4.1.3 | Gestão de Wishlists | 9 |
| 4.1.4 | Gestão de lista de compras | 9 |
| 4.1.5 | Adicionar utilizador a uma despesa | 9 |
| 4.1.6 | Gestão de produtos | 10 |
| 4.1.7 | Gestão de Receitas Favoritas | 10 |
| 4.1.8 | Procura de Produtos | 10 |
| 4.1.9 | Procura de Receitas | 11 |
| 4.1.10 | Gestão de amizades | 11 |
| 4.2 | Requisitos Não Funcionais | 11 |
| 5 | Modelo de Domínios | 13 |
| 6 | Diagrama de Use Cases | 14 |
| 7 | Especificação de Use Cases | 15 |
| 7.1 | Registar | 15 |
| 7.2 | Autenticar | 16 |
| 7.3 | Editar informação pessoal | 16 |
| 7.4 | Editar receitas favoritas | 17 |
| 7.5 | Ver perfil | 17 |
| 7.6 | Editar inventários | 18 |
| 7.7 | Editar wishlists | 19 |
| 7.8 | Editar listas de compras | 21 |
| 7.9 | Editar amigos | 22 |
| 7.10 | Pesquisar produtos | 23 |
| 7.11 | Ver receita | 24 |
| 8 | Arquitetura da Camada de Negócios | 25 |
| 8.1 | Dicionário das Principais Classes | 25 |
| 8.1.1 | User | 25 |

| | | |
|-----------|------------------------------------------------|-----------|
| 8.1.2 | Product | 25 |
| 8.1.3 | Inventory | 25 |
| 8.1.4 | Recipe | 25 |
| 8.2 | Descrição da Arquitetura | 25 |
| 8.3 | Diagrama de Classes | 26 |
| 8.4 | Diagrama de ORM | 26 |
| 9 | Proposta de Interface | 27 |
| 10 | Metodologia de Implementação | 31 |
| 11 | Ferramentas utilizadas na implementação | 32 |
| 12 | Desenvolvimento do Projeto | 33 |
| 12.1 | Conexão da Base de Dados | 33 |
| 12.2 | Pesquisa e Sugestão de Receitas | 33 |
| 12.3 | Gestão de Logins | 33 |
| 13 | Arquitetura Final do Projeto | 34 |
| 14 | Produto Final | 35 |
| 14.1 | Página inicial (Sem sessão iniciada) | 35 |
| 14.2 | Página inicial (com sessão iniciada) | 36 |
| 14.3 | Inventário | 36 |
| 14.4 | Produtos | 41 |
| 14.5 | Receitas | 42 |
| 14.6 | Receitas favoritas | 44 |
| 14.7 | Perfil | 44 |
| 14.8 | Amigos | 46 |
| 14.9 | Popup Erro/Sucesso | 48 |
| 14.10 | Painel de Notificações | 49 |
| 15 | Conclusões | 51 |

Capítulo 1

Introdução

1.1 Motivação e Objetivos

Com este projeto pretendemos simplificar o processo de ir as compras e de gestão das nossas dispensas de casa. Queremos resolver problemas comuns do quotidiano, como comprar produtos que já temos em demasia ou querer cozinhar uma refeição e não saber o que fazer com os ingredientes que temos na dispensa.

Queremos resolver problemas comuns do quotidiano, como comprar produtos que já temos em demasia ou querer cozinhar aquele prato que nos apetece mesmo e faltar algum dos ingredientes. Desta forma, ser-nos-á possível facilitar a vida aos utilizadores e, ao mesmo tempo, reduzir o desperdício alimentar.

1.2 Estrutura do Relatório

Este relatório é composto por 15 capítulos.

Primeiro começamos por apresentar e contextualizar o projeto em si no mundo real explicando as motivações para a criação deste produto.

No segundo capítulo fundamentamos o projeto da aplicação a desenvolver apresentando um estudo de mercado e a viabilidade deste projeto no mundo real.

No terceiro capítulo começamos o planeamento do projeto fazendo uma análise das diversas etapas do projeto ao longo do tempo apresentando um plano de desenvolvimento.

No quarto capítulo descrevemos o **levantamento de requisitos** efetuado com a explicação dos requisitos funcionais e não-funcionais de forma detalhada.

No quinto capítulo apresentamos o **modelo de domínios** da aplicação.

No sexto e sétimo capítulo apresentamos e descrevemos os **Use Cases** do projeto.

Finalmente, a partir do capítulo oito, inclusive, começamos a descrever o desenvolvimento do projeto propriamente dito.

No capítulo oito apresentamos a **arquitetura da camada de negócios** e a estrutura desta.

No capítulo nove apresentamos os *mockups* da aplicação realizados com *AdobeXD*.

No capítulo dez explicamos a **metodologia da implementação** e no capítulo seguinte apresentamos as ferramentas utilizadas para a implementação.

No capítulo doze realçamos algumas partes do trabalho mais relevantes que exemplificam as tecnologias utilizadas.

Finalmente, no capítulo treze e catorze apresentamos o produto final com exemplos de utilização deste.

Capítulo 2

Viabilidade

Após realizar um extensivo estudo de mercado constatamos que existiam algumas aplicações móveis que realizavam uma gestão rudimentar da dispensa em casa.

No entanto faltavam várias funcionalidades para que se tornassem uma ferramenta para utilização diária. Como por exemplo não suportam a sincronização e partilha da dispensa de casa.

Outrossim não existe nenhuma *webapp* que proporcione um serviço semelhante. Também não encontramos nenhum serviço que combine a gestão de dispensas e a sugestão de receitas.

Tendo em conta esta lacuna no mercado concluímos que o nosso projeto seria viável para ser desenvolvido.

Capítulo 3

Plano de Desenvolvimento

Inicialmente o projeto foi planeado para ser desenvolvido ao longo de 10 semanas. Assim, dividimos o projeto em duas fases distintas: planeamento do projeto e desenvolvimento. Em paralelo foi planeado ser escrito o presente relatório.

O seguinte diagrama apresenta o planeamento feito inicialmente.



Para facilitar o desenvolvimento em paralelo e o bom cumprimento dos timings definidos utilizamos os projetos do github complementados com issues ilustrativos das tarefas a fazer.

Capítulo 4

Levantamento de Requisitos

4.1 Requisitos Funcionais

4.1.1 Gestão de Utilizador

Definição de Requisitos de Utilizador

- O utilizador tem que se registar na aplicação.
- O utilizador tem que se autenticar na aplicação.
- O utilizador deve conseguir editar o seu nome, número de telemóvel e data de nascimento.

Definição de Requisitos de Sistema

- Durante o registo, o sistema deve pedir o nome, email, telemóvel, data de nascimento e password.
- O sistema não deve permitir registo de utilizadores com o mesmo email.
- O sistema deve armazenar os dados do utilizador
- O sistema deve mostrar os dados do utilizador aquando a edição dos mesmos.

4.1.2 Gestão de Despesas

Definição de Requisitos de Utilizador

- O utilizador deve conseguir ver, adicionar, editar e remover as suas Despesas.
- O utilizador deve conseguir gerir as despesas que lhe foram partilhadas.

Definição de Requisitos de Sistema

- O sistema deverá apresentar ecrãs ao utilizador onde pode ver todas as suas despesas (próprias e partilhadas) e o conteúdo do interior de cada despesa.
- O sistema deverá permitir o utilizador adicionar, editar e remover as suas despesas.
- O sistema deverá restringir o acesso a despesas às quais o utilizador não tem acesso.
- O sistema deverá restringir a edição de nome e partilha de despesas às quais não é o dono.
- O sistema deverá armazenar todos os dados referentes às despesas do utilizador.

4.1.3 Gestão de Wishlists

Definição de Requisitos de Utilizador

- O utilizador deve conseguir ver, adicionar, editar e remover os seus Wishlists.
- O utilizador deve conseguir gerir os wishlists que lhe foram partilhadas.

Definição de Requisitos de Sistema

- O sistema deverá apresentar ecrãs ao utilizador onde pode ver todas as seus wishlists (próprios e partilhados) e o conteúdo do interior de cada wishlist.
- O sistema deverá permitir o utilizador adicionar, editar e remover os seus wishlists.
- O sistema deverá restringir o acesso aos wishlists aos quais o utilizador não tem acesso.
- O sistema deverá restringir a edição de nome e partilha de wishlists aos quais não é o dono.
- O sistema deverá armazenar todos os dados referentes aos wishlists do utilizador.

4.1.4 Gestão de lista de compras

Definição de Requisitos de Utilizador

- O utilizador deve conseguir ver, adicionar, editar e remover as suas listas de compras.
- O utilizador deve conseguir gerir as listas de compras que lhe foram partilhadas.

Definição de Requisitos de Sistema

- O sistema deverá apresentar ecrãs ao utilizador onde pode ver todas as suas listas de compras (próprias e partilhadas) e o conteúdo do interior de cada lista de compras.
- O sistema deverá permitir o utilizador adicionar, editar e remover as suas listas de compras.
- O sistema deverá restringir o acesso a lista de compras às quais o utilizador não tem acesso.
- O sistema deverá restringir a edição de nome e partilha de listas de compras às quais não é o dono.
- O sistema deverá armazenar todos os dados referentes às listas de compras do utilizador.

4.1.5 Adicionar utilizador a uma despesa

Definição de Requisitos de Utilizador

- O utilizador deve conseguir adicionar utilizadores a uma das suas despesas.
- O utilizador deverá introduzir um email de utilizador válido e existente.

Definição de Requisitos de Sistema

- O sistema deverá exigir um email ao utilizador aquando a realização da partilha da despesa.
- O sistema deverá dar permissões de adicionar e editar produtos da despesa ao utilizador que foi partilhado.
- O sistema deverá restringir a edição do nome do inventário, remoção do inventário e partilha do mesmo com outros utilizadores ao utilizador adicionado.
- O sistema deverá armazenar o utilizador adicionado na despesa partilhada.
- O sistema deverá negar a adição de um email que não existe ou não é válido.

4.1.6 Gestão de produtos

Definição de Requisitos de Utilizador

- O utilizador deve conseguir adicionar, remover e editar produtos nas suas Despesas, Wishlists e listas de compras.
- O utilizador deve conseguir adicionar, remover e editar produtos nas suas Despesas, Wishlists e listas de compras partilhadas.

Definição de Requisitos de Sistema

- O sistema deverá permitir o utilizador de adicionar, editar e remover os produtos das suas despesas.
- O sistema deverá apresentar ao utilizador na adição de produtos as categorias existentes, os produtos respetivos a essas mesmas e exigir quantidade e data de validade (quando necessário).
- O sistema deverá restringir a adição, edição e remoção de produtos de despesas, wishlists e listas de compras às quais não tem acesso.
- O sistema deverá armazenar todos os produtos adicionados nas despesas, wishlists e listas de compras como não armazenar os que foram removidos.

4.1.7 Gestão de Receitas Favoritas

Definição de Requisitos de Utilizador

- O utilizador deverá conseguir adicionar e remover receitas da sua lista de receitas favoritas.

Definição de Requisitos de Sistema

- O sistema deverá apresentar ao utilizador um ecrã com a sua lista de receitas favoritas.
- O sistema deverá permitir o utilizador de adicionar e remover receitas favoritas da sua lista.
- O sistema deverá restringir o acesso ao utilizador a listas de favoritos às quais não tem acesso.
- O sistema deverá armazenar as receitas favoritas adicionadas pelo utilizador como deverá não armazenar as receitas que o utilizador remover da sua lista.

4.1.8 Procura de Produtos

Definição de Requisitos de Utilizador

- O utilizador deverá conseguir fazer uma pesquisa de produtos através da inserção de palavras na caixa de pesquisa da página de produtos.
- O utilizador deverá conseguir adicionar produtos às suas despesas, wishlists e listas de compras dos produtos visíveis na procura de produtos.

Definição de Requisitos de Sistema

- O sistema deverá apresentar ao utilizador um ecrã com um campo de procura e uma lista de produtos com base na procura realizada.
- O sistema deverá permitir o utilizador de pesquisar produtos através de palavras.
- O sistema deverá permitir o utilizador de adicionar produtos apresentados na pesquisa às suas despesas, wishlists e listas de compras.
- O sistema deverá mostrar produtos que tem armazenados.

- O sistema deverá armazenar os produtos que o utilizador adicionou às suas despensas, wish-lists e listas de compras.

4.1.9 Procura de Receitas

Definição de Requisitos de Utilizador

- O utilizador deverá conseguir fazer uma pesquisa de receitas através da inserção de palavras na caixa de pesquisa da página de receitas.
- O utilizador deverá conseguir adicionar receitas à sua lista de receitas favoritas das receitas visíveis na procura de receitas.

Definição de Requisitos de Sistema

- O sistema deverá apresentar ao utilizador um ecrã com um campo de procura e uma lista de receitas com base na procura realizada.
- O sistema deverá permitir o utilizador de pesquisar receitas através de palavras.
- O sistema deverá permitir o utilizador de adicionar receitas apresentadas na pesquisa à sua lista de receitas favoritas.
- O sistema deverá mostrar receitas da API externa.
- O sistema deverá armazenar as receitas que o utilizador adicionou à sua lista de receitas favoritas.

4.1.10 Gestão de amizades

Definição de Requisitos de Utilizador

- O utilizador deve conseguir ver, adicionar e remover utilizadores (amigos) da sua lista de amigos.
- O utilizador deve conseguir ver e remover pedidos de amizade enviados.
- O utilizador deve conseguir aceitar e recusar pedidos de amizade que lhe foram enviados.

Definição de Requisitos de Sistema

- O sistema deverá apresentar ecrãs ao utilizador onde pode ver todos os seus pedidos de amizade enviados, recebidos e amigos adicionados.
- O sistema deverá permitir o utilizador adicionar e remover pedidos de amizade enviados.
- O sistema deverá permitir o utilizador adicionar e remover amigos.
- O sistema deverá permitir o utilizador aceitar e recusar pedidos de amizade recebidos.
- O sistema deverá restringir o acesso a lista de amigos ao quais o utilizador não tem acesso.
- O sistema deverá armazenar todos os dados referentes às amizades do utilizador.

4.2 Requisitos Não Funcionais

- Um utilizador deve ser criado fornecendo um nome, password, email e data de nascimento
- Uma despesa, wishlist e lista de compras devem ser criadas com nome
- Os produtos devem ser adicionados com nome, quantidade e data de validade
- As receitas podem ser pesquisadas conforme os produtos de uma despesa

- Um utilizador pode ter receitas favoritas
- Um utilizador pode ser amigo de um ou mais utilizadores
- A aplicação deverá estar disponível durante os 7 dias da semanas, 24 horas por dia

Capítulo 5

Modelo de Domínios

Após a análise de requisitos desenvolvemos o seguinte modelo de domínio que representa todas as entidades relevantes à implementação dos requisitos previamente definidos.

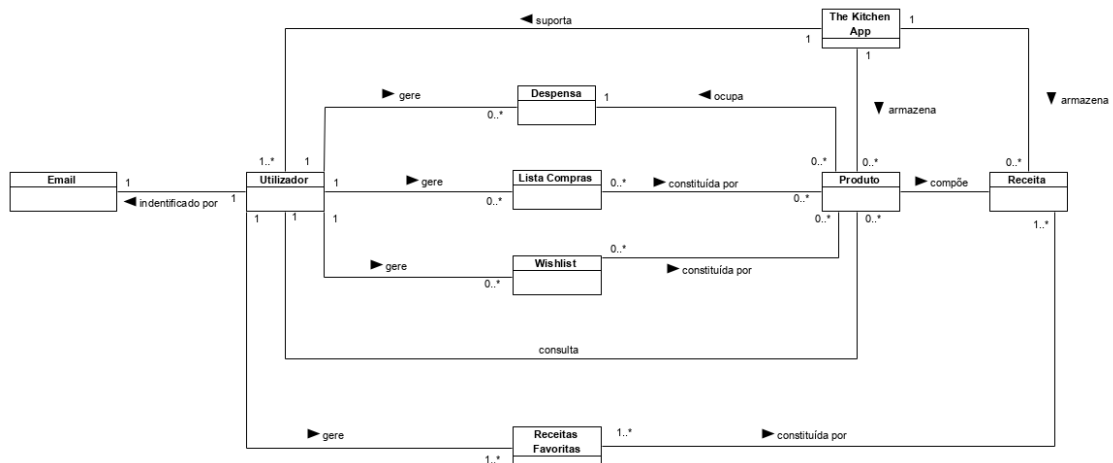
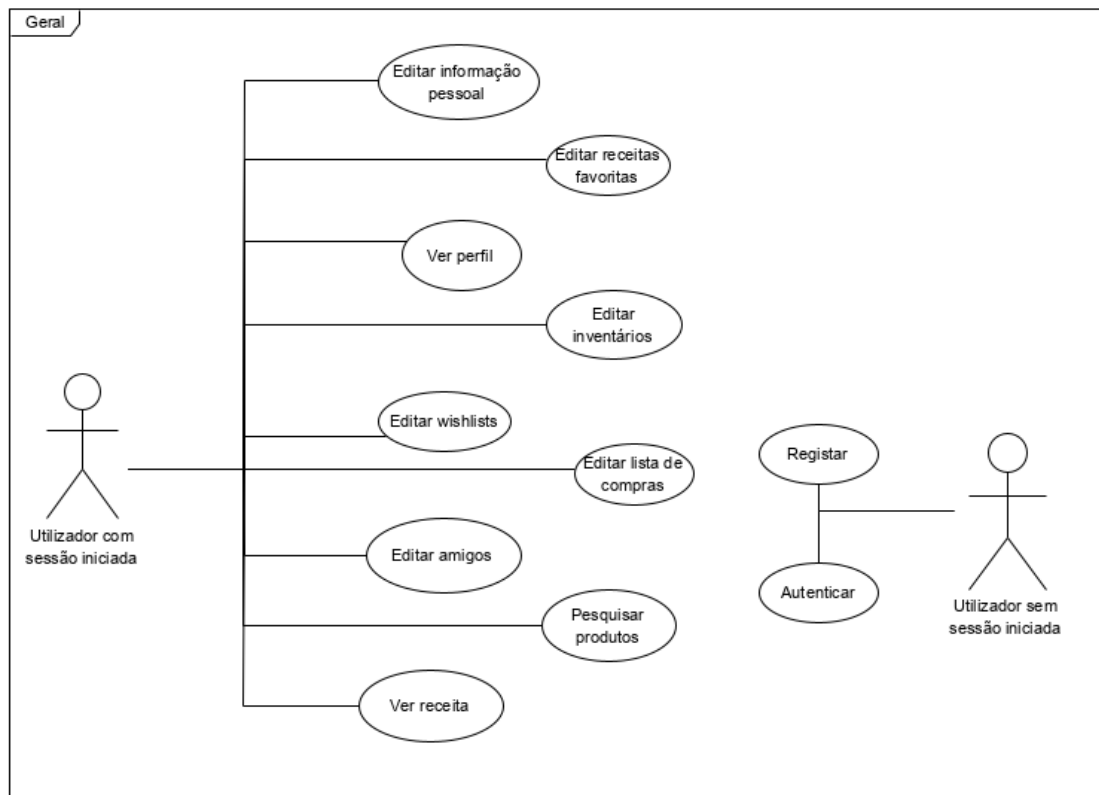


Figura 5.1: Modelo de Domínios

Capítulo 6

Diagrama de Use Cases



Capítulo 7

Especificação de Use Cases

7.1 Registrar

| Use Case | | |
|-----------------------------------------------------------------|-------------------------------------------|---------------------------------------------------------------------------------------|
| Use Case: | Registrar | |
| Ator: | Utilizador | |
| Pré-condição: | - | |
| Pós-condição: | Um utilizador novo é registado no sistema | |
| | Ator | Sistema |
| Cenário Normal | 1. Insere os dados para registo | |
| | | 2. Verifica se já existe um utilizador com o mesmo email no sistema |
| | | 3. Guarda no sistema o novo utilizador |
| | | 4. Informa que o registo foi efetuado |
| Exceção 1 [Já existe um utilizador com o mesmo email] (Passo 2) | | 3.1. Informa que as credenciais não são válidas ou já existe um utilizador no sistema |

7.2 Autenticar

| Use Case | | |
|-------------------------------------------|-----------------------------------|-------------------------------------------------|
| Use Case: | Autenticar | |
| Ator: | Utilizador | |
| Pré-condição: | O utilizador não está autenticado | |
| Pós-condição: | O utilizador está autenticado | |
| | Ator | Sistema |
| Cenário Normal | 1. Insere as suas credenciais | |
| | | 2. Valida as credenciais |
| | | 3. Redireciona para a dashboard |
| Exceção 1 [Credenciais erradas] (Passo 2) | | 3.1. Informa que as credenciais não são válidas |

7.3 Editar informação pessoal

| Use Case | | |
|------------------------------------------|-----------------------------------------|-----------------------------------------------------|
| Use Case: | Editar Informação Pessoal | |
| Ator: | Utilizador | |
| Pré-condição: | Utilizador tem sessão iniciada | |
| Pós-condição: | Alguma informação pessoal é alterada | |
| | Ator | Sistema |
| Cenário Normal | | 1. Apresenta os campos editáveis do perfil |
| | 2. Seleciona qual campo pretende editar | |
| | 3. Grava as alterações | |
| | | 4. Grava as alterações e retorna a página do perfil |
| Exceção 1 [Cancela alterações] (Passo 3) | | 3.1. Fecha o popup com o campo editável |

7.4 Editar receitas favoritas

| Use Case | | |
|---------------------------------------------------------------------------|-------------------------------------------------------------------|---------------------------------------------------------------------------------------------|
| Use Case: | Editar Receitas Favoritas | |
| Ator: | Utilizador | |
| Pré-condição: | Utilizador tem sessão iniciada | |
| Pós-condição: | As receitas favoritas do utilizador são alteradas | |
| | Ator | Sistema |
| Cenário Normal | 1. Acede à página de pesquisa de receitas e pesquisa por receitas | |
| | | 2. Apresenta as receitas com base na pesquisa |
| | 3. Coloca uma receita como favorita | |
| | | 4. Guarda na página de receitas favoritas do utilizador a receita que colocou como favorita |
| Comp. alternativos [Acede à pagina de receitas favoritas] (Passo 1) | | 1.1 Apresenta as receitas favortias |
| | 1.2 Remove uma receita dos favoritos | |
| | | 1.3 Elimina a receita da lista de favoritos |

7.5 Ver perfil

| Use Case | | |
|----------------|--------------------------------|-------------------------------------|
| Use Case: | Ver Perfil | |
| Ator: | Utilizador | |
| Pré-condição: | Utilizador tem sessão iniciada | |
| Pós-condição: | - | |
| | Ator | Sistema |
| Cenário Normal | | 1. Apresenta o perfil ao utilizador |

7.6 Editar inventários

| Use Case | | |
|-----------------------------------------------------------------------------------|-----------------------------------------------------------------|------------------------------------------------------------|
| Use Case: | Editar inventários | |
| Ator: | Utilizador | |
| Pré-condição: | Utilizador tem sessão iniciada | |
| Pós-condição: | Alguma informação dos inventários é alterada | |
| | Ator | Sistema |
| Cenário Normal | | 1. Apresenta a lista de inventários |
| | 2. Selecciona um inventário | |
| | | 3. Apresenta os produtos presentes no inventário |
| | 4. Selecciona que pretende adicionar um produto | |
| | | 5. Apresenta as categorias disponíveis |
| | 6. Selecciona uma categoria | |
| | | 7. Apresenta os produtos da categoria |
| | 8. Selecciona o produto e fornece quantidade e data de validade | |
| | | 9. Adiciona o produto ao inventário |
| Comp. alternativos [Selecciona que pretende editar um inventário] (Passo 4) | 4.1 Indica qual o nome novo do inventário | |
| | | 4.2 Atualiza o nome do inventário |
| Comp. alternativos [Selecciona que pretende editar um produto] (Passo 4) | 4. Selecciona que pretende editar um produto | |
| | 5. Fornece quantidade e/ou data de validade | |
| | | 6. Atualiza o produto |
| Comp. alternativos [Selecciona que pretende partilhar um inventário] (Passo 4) | 4.1 Indica qual o email do utilizador | |
| | | 4.2 Verifica se o email é válido |
| | | 4.3 Partilha o inventário com o utilizador |
| Comp. alternativos [Selecciona que pretende remover um produto] (Passo 4) | | 4.1 Pede confirmação ao utilizador |
| | 4.2 Confirma | |
| | | 4.3 Remove o produto do inventário |
| Comp. alternativos [Selecciona que pretende remover um inventário] (Passo 4) | | 4.1 Pede confirmação ao utilizador |
| | 4.2 Confirma | |
| | | 4.3 Remove o inventário |
| Exceção 1 [Quantidade ou data de validade inválidas] (Passo 9) | | 9.1 Informa o utilizador que inseriu valores inválidos |
| Exceção 2 [Nome de inventário igual a um que já possui] (Passo 4.2) | | 4.2.2 Informa o utilizador que o inventário não foi criado |
| Exceção 3 [O utilizador diz que não] (Passo 4.2) | | 4.2.2 Não remove |
| Exceção 4 [Email inválido] (Passo 4.2) | | 4.3 Não partilha e mostra mensagem de erro |

7.7 Editar wishlists

| Use Case | | |
|--------------------------------------------------------------------------------|------------------------------------------------|----------------------------------------------------------|
| Use Case: | Editar wishlists | |
| Ator: | Utilizador | |
| Pré-condição: | Utilizador tem sessão iniciada | |
| Pós-condição: | Algum informação dos wishlists é alterada | |
| | Ator | Sistema |
| Cenário Normal | | 1. Apresenta a lista de wishlists |
| | 2. Seleciona um wishlist | |
| | | 3. Apresenta os produtos presentes no wishlist |
| | 4. Seleciona que pretende adicionar um produto | |
| | | 5. Apresenta as categorias disponíveis |
| | 6. Seleciona uma categoria | |
| | | 7. Apresenta os produtos da categoria |
| | 8. Seleciona o produto | |
| | | 9. Adiciona o produto ao wishlist |
| Comp. alternativos [Seleciona que pretende editar um wishlist] (Passo 4) | 4.1 Indica qual o nome novo do wishlist | |
| | | 4.2 Atualiza o nome do wishlist |
| Comp. alternativos [Seleciona que pretende partilhar um wishlist] (Passo 4) | 4.1 Indica qual o email do utilizador | |
| | | 4.2 Verifica se o email é válido |
| | | 4.3 Partilha o wishlist com o utilizador |
| Comp. alternativos [Seleciona que pretende remover um produto] (Passo 4) | | 4.1 Pede confirmação ao utilizador |
| | 4.2 Confirma | |
| | | 4.3 Remove o produto do wishlist |
| Comp. alternativos [Seleciona que pretende remover um wishlist] (Passo 4) | | 4.1 Pede confirmação ao utilizador |
| | 4.2 Confirma | |
| | | 4.3 Remove o wishlist |
| Exceção 1 [Nome de wishlist igual a um que já possui] (Passo 4.2) | | 4.2.2 Informa o utilizador que a wishlist não foi criado |
| Exceção 2 [O utilizador diz que não] (Passo 4.2) | | 4.2.2 Não remove |
| Exceção 3 [Email inválido] (Passo 4.2) | | 4.3 Não partilha e mostra mensagem de erro |

7.8 Editar listas de compras

| Use Case | | |
|--------------------------------------------------------------------------------------|-------------------------------------------------|---------------------------------------------------------------|
| Use Case: | Editar shopping lists | |
| Ator: | Utilizador | |
| Pré-condição: | Utilizador tem sessão iniciada | |
| Pós-condição: | Alguma informação dos shopping lists é alterada | |
| | Ator | Sistema |
| Cenário Normal | | 1. Apresenta a lista de shopping lists |
| | 2. Selecciona um shopping list | |
| | | 3. Apresenta os produtos presentes no shopping list |
| | 4. Selecciona que pretende adicionar um produto | |
| | | 5. Apresenta as categorias disponíveis |
| | 6. Selecciona uma categoria | |
| | | 7. Apresenta os produtos da categoria |
| | 8. Selecciona o produto e fornece quantidade | |
| | | 9. Adiciona o produto ao shopping list |
| Comp. alternativos [Selecciona que pretende editar um shopping list] (Passo 4) | 4.1 Indica qual o nome novo do shopping list | |
| | | 4.2 Atualiza o nome do shopping list |
| Comp. alternativos [Selecciona que pretende editar um produto] (Passo 4) | 4. Selecciona que pretende editar um produto | |
| | 5. Fornece quantidade | |
| | | 6. Atualiza o produto |
| Comp. alternativos [Selecciona que pretende partilhar um shopping list] (Passo 4) | 4.1 Indica qual o email do utilizador | |
| | | 4.2 Verifica se o email é válido |
| | | 4.3 Partilha o shopping list com o utilizador |
| Comp. alternativos [Selecciona que pretende remover um produto] (Passo 4) | | 4.1 Pede confirmação ao utilizador |
| | 4.2 Confirma | |
| Comp. alternativos [Selecciona que pretende remover um shopping list] (Passo 4) | | 4.3 Remove o produto do shopping list |
| | | 4.1 Pede confirmação ao utilizador |
| | 4.2 Confirma | |
| Exceção 1 [Quantidade inválida] (Passo 9) | | 4.3 Remove o shopping list |
| | | 9.1 Informa o utilizador que inseriu valores inválidos |
| Exceção 2 [Nome de shopping list igual a um que já possui] (Passo 4.2) | | 4.2.2 Informa o utilizador que o shopping list não foi criado |
| Exceção 3 [O utilizador diz que não] (Passo 4.2) | | 4.2.2 Não remove |
| Exceção 4 [Email inválido] (Passo 4.2) | | 4.3 Não partilha e mostra mensagem de erro |

7.9 Editar amigos

| Use Case | | |
|----------------------------------------------------------------------|-----------------------------------------|----------------------------------------------------------------------|
| Use Case: | Editar amigos | |
| Ator: | Utilizador | |
| Pré-condição: | Utilizador tem sessão iniciada | |
| Pós-condição: | Alguma informação dos amigos é alterada | |
| | Ator | Sistema |
| Cenário Normal | | 1. Apresenta a lista de pedidos recebidos, pedidos enviados e amigos |
| | 2. Seleciona um pedido recebido | |
| | 3. Escolhe se pretende ou não aceitar | |
| | 4. Aceita | |
| | | 5. Adiciona o utilizador à lista de amigos |
| Comp. alternativos [Pretende adicionar um amigo] (Passo 2) | 2.1. Indica o email do utilizador | |
| | | 2.2 Valida o email |
| | | 2.3 Envia o pedido para o utilizador |
| Comp. alternativos [Pretende remover um amigo] (Passo 2) | 2.1 Seleciona o amigo a remover | |
| | | 2.2 Pede confirmação |
| | 2.3 Aceita | |
| | | 2.4. Remove o amigo |
| Comp. alternativos [Pretende remover um pedido enviado] (Passo 2) | 2.1 Seleciona o pedido enviado | |
| | 2.2 Seleciona para remover | |
| | | 2.3. Remove o pedido |
| Exceção 1 [Recusa pedido] (Passo 4) | | 4.1 Elimina o pedido |
| Exceção 2 [Email inválido] (Passo 2.2) | | 4.2.2 Informa o utilizador que o pedido não foi enviado |

7.10 Pesquisar produtos

| Use Case | | |
|-----------------------------------------------------------------------|--------------------------------------------------|-----------------------------------------------------|
| Use Case: | Pesquisar produtos | |
| Ator: | Utilizador | |
| Pré-condição: | Utilizador tem sessão iniciada | |
| Pós-condição: | Produtos serão ou não apresentados | |
| | Ator | Sistema |
| Cenário Normal | 1. Pesquisa por produtos | |
| | | 2. Apresenta lista de produtos com base na pesquisa |
| | 3. Seleciona que pretende adicionar esse produto | |
| | | 4. Mostra o tipo de listas |
| | 5. Seleciona a opção de inventários | |
| | | 6. Mostra os inventários que possui |
| | 7. Seleciona o inventário | |
| | 8. Fornece quantidade e data de validade | |
| | | 9. Adiciona o produto ao inventário |
| Comp. alternativos [Seleciona a opção de wishlists] (Passo 5) | | 5.1. Mostra os wishlists que possui |
| | 5.2 Seleciona o wishlist | |
| | | 5.3. Adiciona o produto ao wishlist |
| Comp. alternativos [Seleciona a opção de shopping lists] (Passo 2) | | 5.1. Mostra os shopping lists que possui |
| | 5.2 Seleciona o shopping list | |
| | 5.3. Fornece quantidade | |
| | | 5.4. Adiciona o produto ao shopping list |
| Exceção 1 [Dados inseridos inválidos] (Passo 6) | | 4.1 Informa que os dados introduzidos são inválidos |

7.11 Ver receita

| Use Case | | |
|--------------------------------------------------------------------|-------------------------------------------------------------|---------------------------------------------------------------------|
| Use Case: | Ver receita | |
| Ator: | Utilizador | |
| Pré-condição: | Utilizador tem sessão iniciada e a receita existe | |
| Pós-condição: | Informações de uma receita são apresentadas | |
| | Ator | Sistema |
| Cenário Normal | 1. Acede à página de uma receita | |
| | | 2. Mostra imagem, sumário, ingredientes e instruções de uma receita |
| Comp. alternativos [Adiciona a receita aos favoritos] (Passo 3) | 3. Seleciona que pretende adicionar a receita aos favoritos | |
| | | 4. Atualiza as receitas favoritas |

Capítulo 8

Arquitetura da Camada de Negócios

8.1 Dicionário das Principais Classes

8.1.1 User

Corresponde à representação no sistema dos utilizadores registados, contendo a sua informação pessoal e credenciais de acesso.

8.1.2 Product

Representa a unidade básica de todas as dispensas, listas de compras e listas de favoritos. Contém informação do nome, categoria, preço e quantidades por embalagem. Existem classes referentes aos vários tipos de produtos, que herdam desta, e contêm informação extra para ser possível acomodar os diferentes tipos de listas presentes no programa.

8.1.3 Inventory

A classe Inventory representa uma lista de produtos de um utilizador, contendo informação sobre o seu dono, nome e produtos disponíveis nesta, e com quem é partilhada. A lista de produtos tem que pertencer à super classe *Product*, e vai ser isto que vai determinar a que corresponde o inventário em questão, se é uma dispensa, lista de favoritos ou de compras.

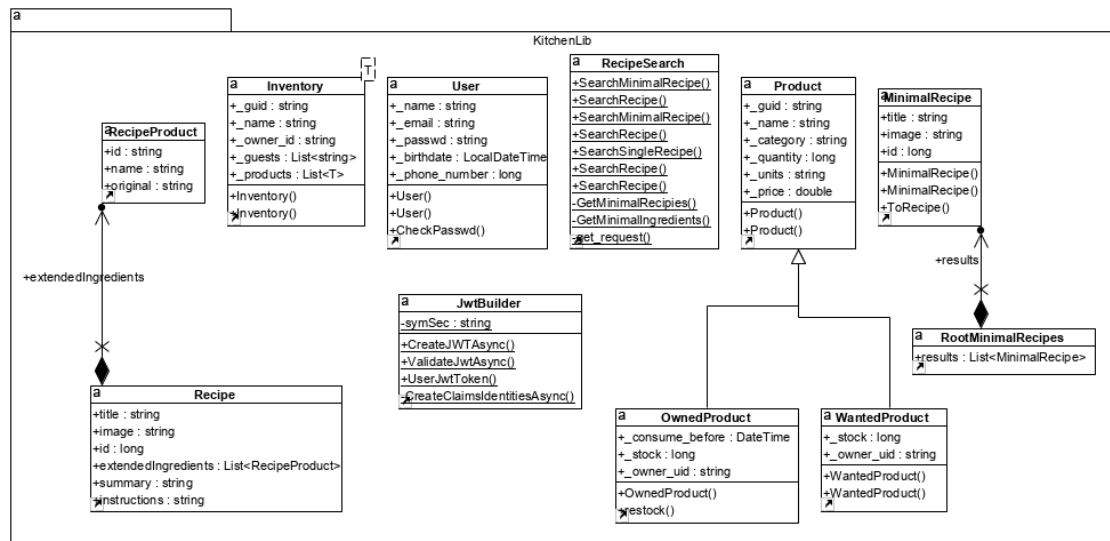
8.1.4 Recipe

Esta classe representa uma receita, contendo informação sobre o seu nome, resumo, ingredientes necessários à confeção e instruções de preparação.

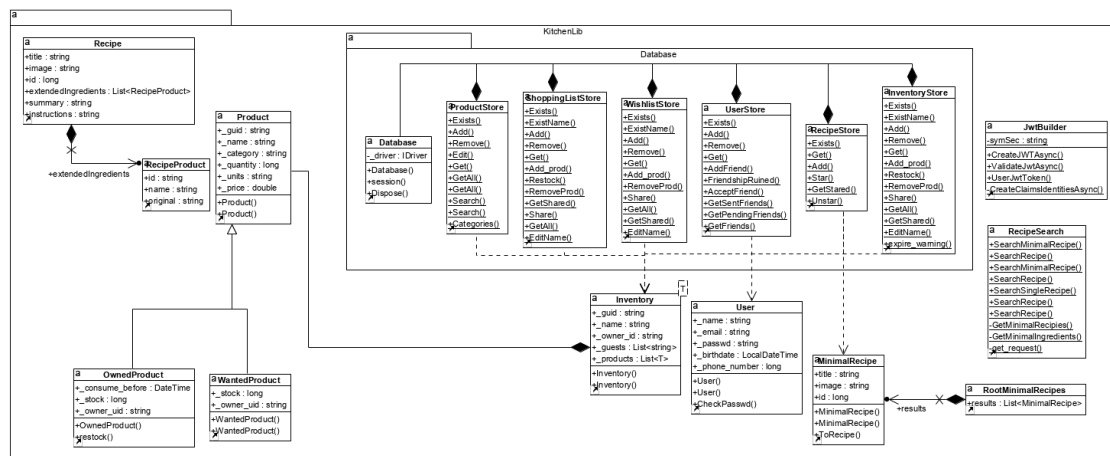
8.2 Descrição da Arquitetura

Cada utilizador possui uma lista de dispensas, uma lista de listas de produtos favoritos, uma lista de listas de compras e uma lista de receitas favoritas. Também existe a possibilidade de um utilizador partilhar as suas listas de produtos, tendo cada utilizador uma lista de inventários partilhados consigo.

8.3 Diagrama de Classes



8.4 Diagrama de ORM



Capítulo 9

Proposta de Interface

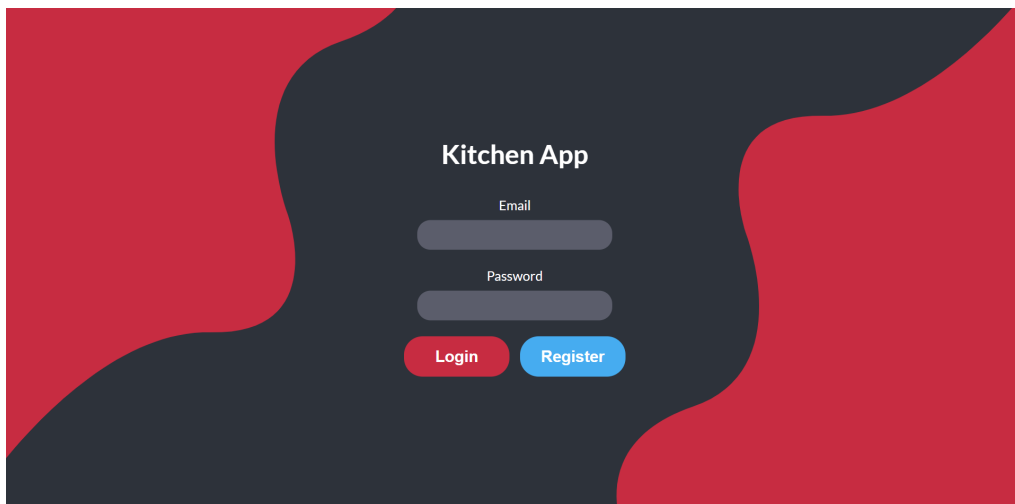


Figura 9.1: Login no Desktop

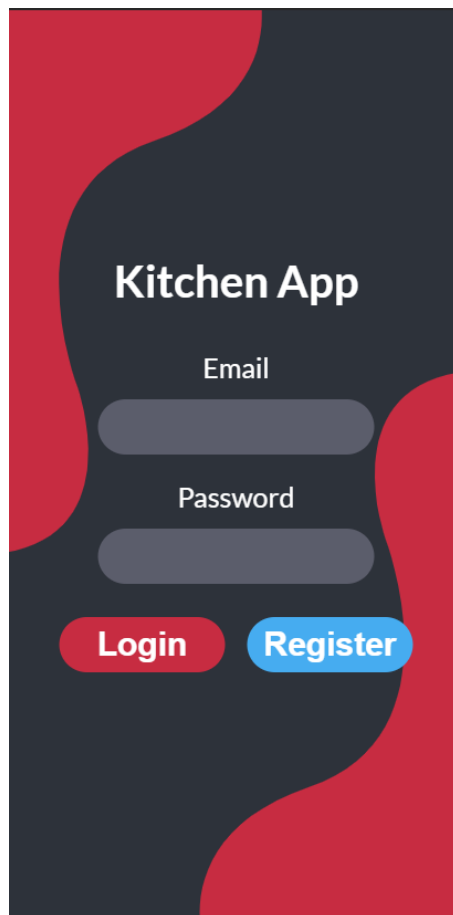


Figura 9.2: Login em Mobile

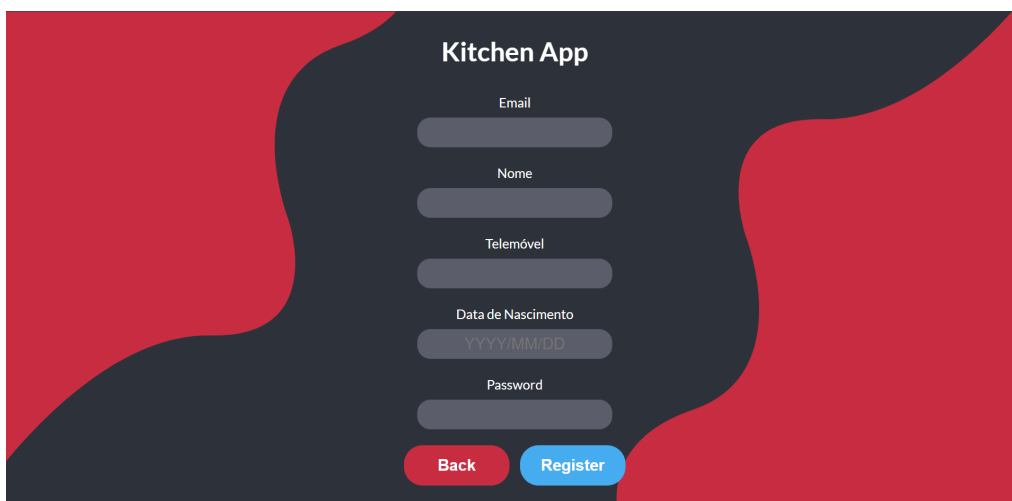
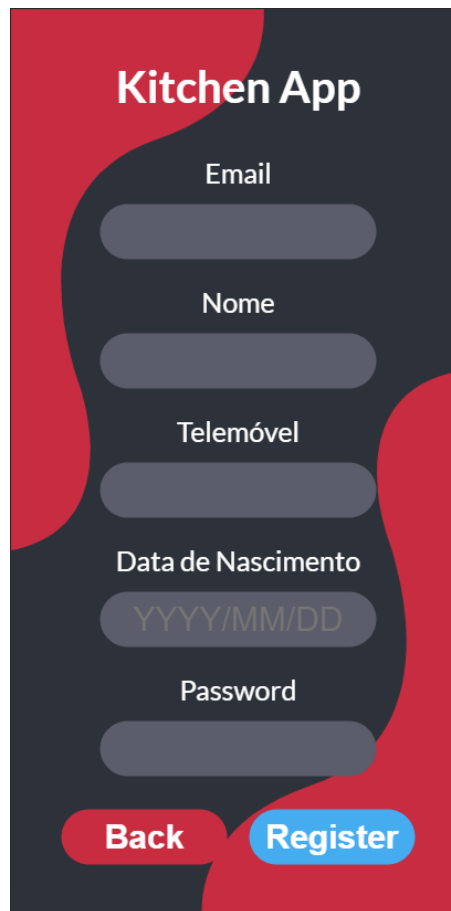


Figura 9.3: Registo no Desktop



Kitchen App

Email

Nome

Telemóvel

Data de Nascimento

YYYY/MM/DD

Password

Back **Register**

The image shows a mobile registration screen for the 'Kitchen App'. It features a dark blue background with red abstract shapes on the left and right sides. The form fields are arranged vertically: Email, Nome, Telemóvel, Data de Nascimento (with a placeholder 'YYYY/MM/DD'), and Password. Each field has a corresponding grey input box. At the bottom, there are two buttons: a red 'Back' button and a blue 'Register' button.

Figura 9.4: Registo em Mobile

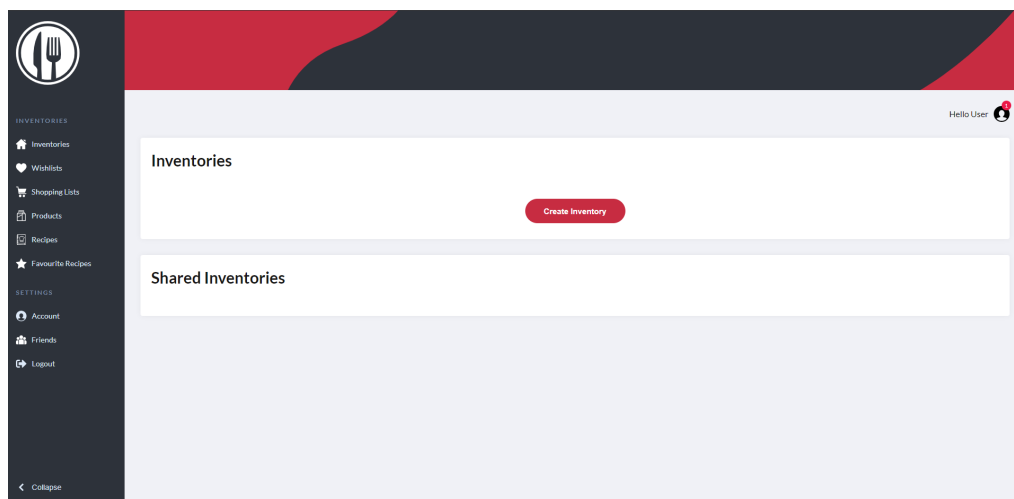


Figura 9.5: Dashboard no Desktop

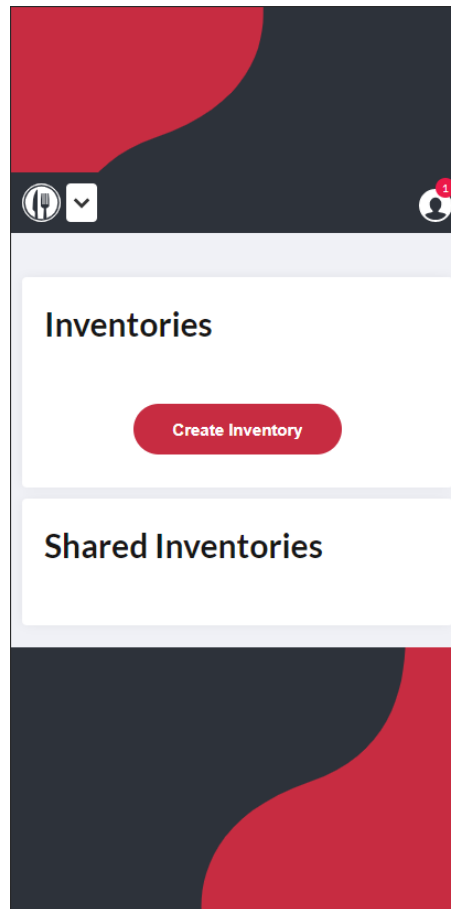


Figura 9.6: Dashboard em Mobile

Capítulo 10

Metodologia de Implementação

O padrão arquitetural escolhido para a implementação da aplicação foi o MVC (*Model View Controller*), por forma a obter um conjunto de pequenos componentes modulares de fácil integração, e assim facilitar o desenvolvimento em paralelo. De forma a tornar a implementação ainda mais modular, a componente do controlador foi decomposta em pequenos controladores individuais, cada um responsável por lidar com uma pequena parte da aplicação, como por exemplo, lidar com as dispensas, ou com a autenticação.

A escolha deste padrão arquitetural proporciona também uma camada de abstração, deixando a possibilidade de implementar a aplicação em mais do que uma plataforma, sem fazer alterações ao previamente feito.

A componente do modelo foi decomposta em camada de negócio e de dados, por forma a garantir a consistência, integridade e disponibilidade dos dados da aplicação, qualquer que seja a tecnologia utilizada para o suporte da base de dados.

Capítulo 11

Ferramentas utilizadas na implementação

A principal ferramenta utilizada para a implementação da aplicação apresentada foi a *framework* de desenvolvimento ASP.NET Core. Esta escolha foi tomada graças à excelente documentação disponível e simplicidade de utilização.

Como gestor de base de dados escolhemos o *Neo4j*, que utiliza um modelo não relacional e que se enquadra com a disposição definida dos dados da aplicação.

Para a criação da interface do produto final, foi utilizada a biblioteca *React*, assente na linguagem de programação *JavaScript*, pela sua popularidade e facilidade de utilização.

Para tornar o *deployment* da aplicação o mais agnóstico da máquina onde é feito, e para facilitar os testes feitos durante o processo de desenvolvimento, foi utilizada a ferramenta *Docker* e *Docker Compose*, na qual foram criados *containers* contendo a aplicação, todas as suas configurações, tornando o processo de correr a aplicação apenas um comando, sem haver qualquer preocupação ao nível de *networking*.

De forma a validar os pedidos chegados à aplicação, e encaminha-los para o devido controlador, utilizamos como *Reverse Proxy* a ferramenta *Nginx*.

Para garantir a disponibilidade da aplicação a qualquer hora, esta foi hospedada na plataforma *Azure*, tirando partido do esforço anterior de utilização da ferramenta *Docker*.

Por fim, foram utilizados diversos *browsers*, como o *Firefox* ou o *Brave*, em várias plataformas, para garantir a boa aparência da interface apresentada ao utilizador final.

Capítulo 12

Desenvolvimento do Projeto

12.1 Conexão da Base de Dados

Após a instalação e configuração do motor de base de dados Neo4j, passamos à integração deste com a aplicação em desenvolvimento. Para o processo de conexão entre ambas as partes utilizamos o *Neo4jDotNetDriver*, pacote oficial deste motor de base de dados para a linguagem de programação em uso, *C#*.

Como o Neo4j dispensa uma geração inicial da base de dados, partimos para a escrita de queries que manipulasse esta, de forma a que, quando esta estiver povoada, se assemelhe ao inicialmente proposto.

Para povoar a base de dados, optamos por apenas introduzir produtos, sendo contas de utilizadores criadas à medida que necessário.

12.2 Pesquisa e Sugestão de Receitas

Para complementar a funcionalidade chave desta aplicação, gestão de stocks de despensas, foi integrado na aplicação uma API externa, chamada *Spoonacular*, capaz de fornecer receitas culinárias, permitindo uma pesquisa bastante detalhada sobre estas.

Assim, implementamos na aplicação a capacidade de pesquisa de receitas por nome, e a capacidade de sugerir receitas com base no que existe numa despesa.

12.3 Gestão de Logins

Por forma a simplificar a gestão do utilizador que faz os pedidos, implementamos um sistema de tokens, assente no standard *JSON Web Token (JWT)*, que contém informação sobre o utilizador, prazo de validade e também uma assinatura digital que marca a autenticidade do mesmo, não dando assim acessos indevidos às páginas privadas do utilizador. Este token é validado e renovado a cada pedido efetuado, e é transmitido num header do pedido *HTTP*.

Capítulo 13

Arquitetura Final do Projeto

Após a implementação da aplicação, a arquitetura final tem o seguinte aspeto:

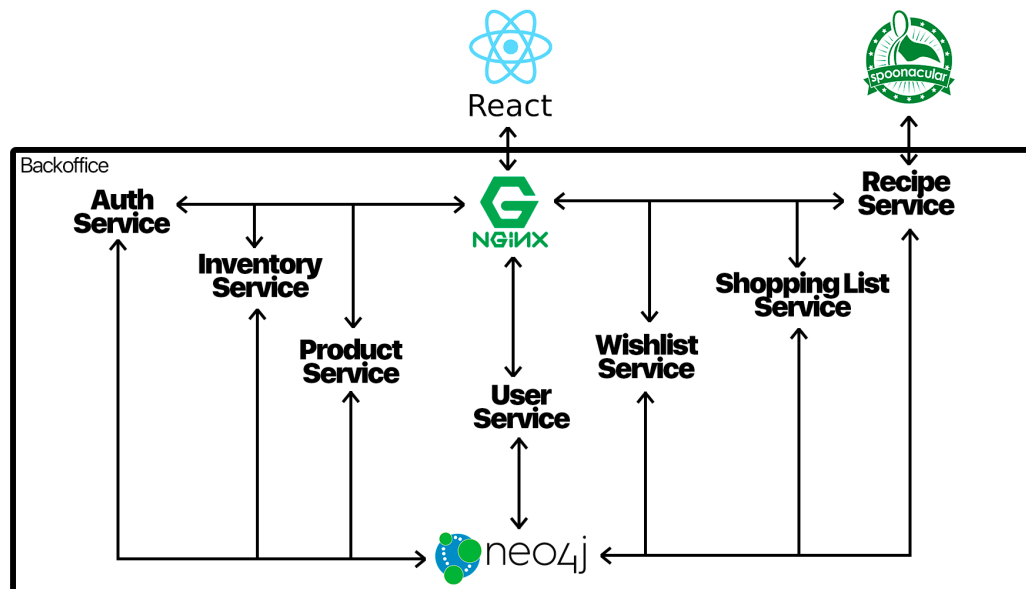


Figura 13.1: Arquitetura Final da Aplicação

No backoffice, os pedidos são efetuados pela *Web API* definida e documentada, e são atendidos pelo *nginx*, e posteriormente encaminhados para o serviço referente ao pedido recebido. Caso sejam pedidos que necessitem autenticação, antes de encaminhados para o serviço correto, ainda passam pelo *Auth Service*, responsável pela autenticação dos utilizadores. Apenas é exposta a ligação ao *proxy* ao exterior, sendo todas as restantes conexões aos serviços e base de dados feitas internamente. Apenas um dos serviços faz conexões diretas ao exterior, sendo este o referente às receitas, que faz pedidos à API externa integrada.

O frontend comunica com os backend com os mesmos métodos acima descritos, sendo o backend completamente independente do frontend.

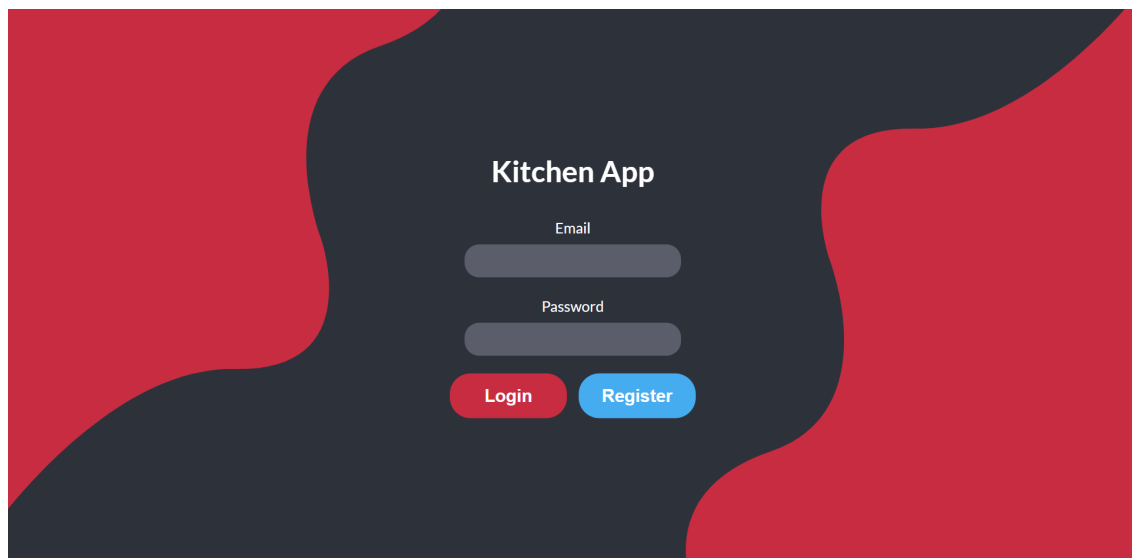
Capítulo 14

Produto Final

Ao longo de várias semanas a nossa equipa desenvolveu o Kitchen App até que o produto de software final fosse o inicialmente idealizado. Abaixo, iremos demonstrar como a aplicação funciona no geral e onde se encontra e como estão implementadas as nossas funcionalidades.

14.1 Página inicial (Sem sessão iniciada)

A página de login é a inicial e principal quando não se está autenticado. Redirecionados para esta página são os utilizadores que ou não se autenticaram ou então não possuem um token válido (exemplos destes são os utilizadores que se encontram autenticados na aplicação há tempo suficiente para que o seu token perdesse a validade). Nesta página é então possível um utilizador autenticar se ou aceder à página de registo.



Através então **da opção de Registo** é se direcionado para esta página onde se é possível realizar o registo de um utilizador. Caso não se pretenda realizar um registo podemos voltar para a página de login na opção **back**.

Kitchen App

Email

Nome

Telemóvel

Data de Nascimento

Password

14.2 Página inicial (com sessão iniciada)

Estando autenticado, é apresentado ao utilizador a sua **dashboard**. Nesta é possível ver todos os **inventários** que o utilizador possui (próprios e partilhados), as **notificações** (serão mostradas mais tarde) e todas as opções no **menu da esquerda** para nos ser possível mover na aplicação (tanto as notificações como o menu lateral esquerdo estão presentes em todas as páginas que é necessário autenticação).

Hello Miguel Solino

Inventories

Braga

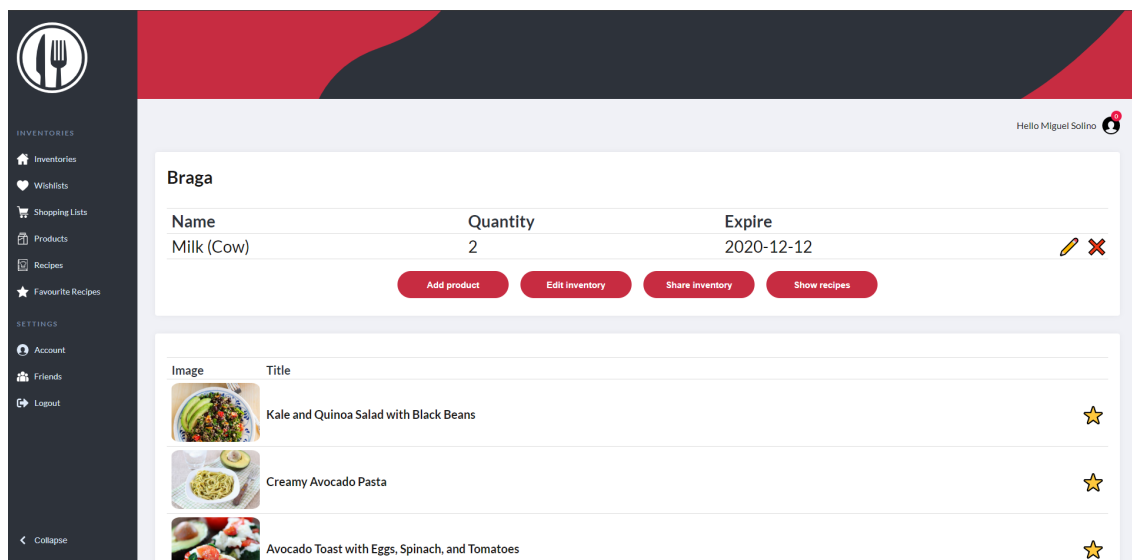
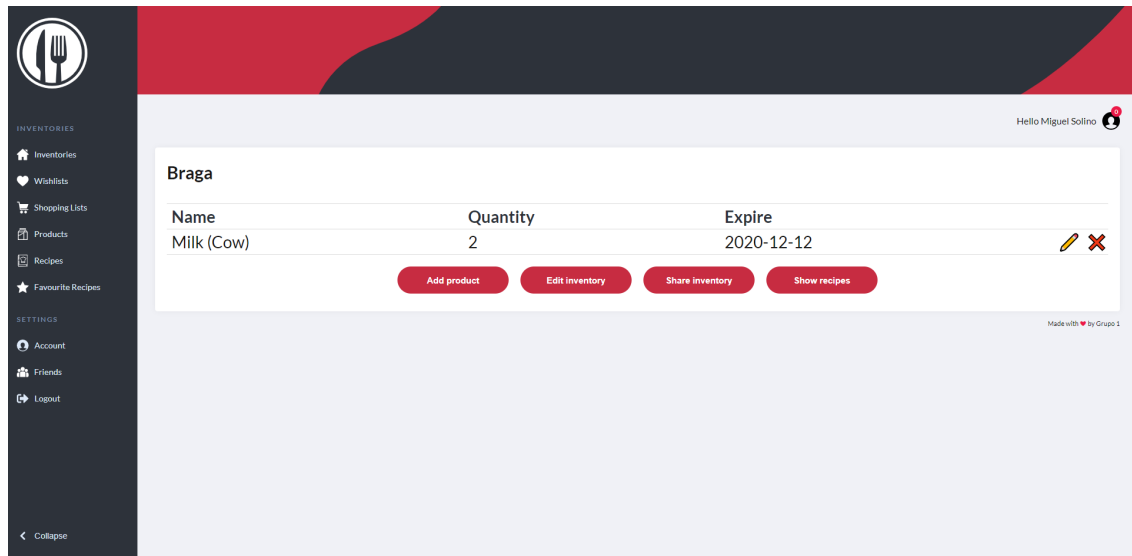
Shared Inventories

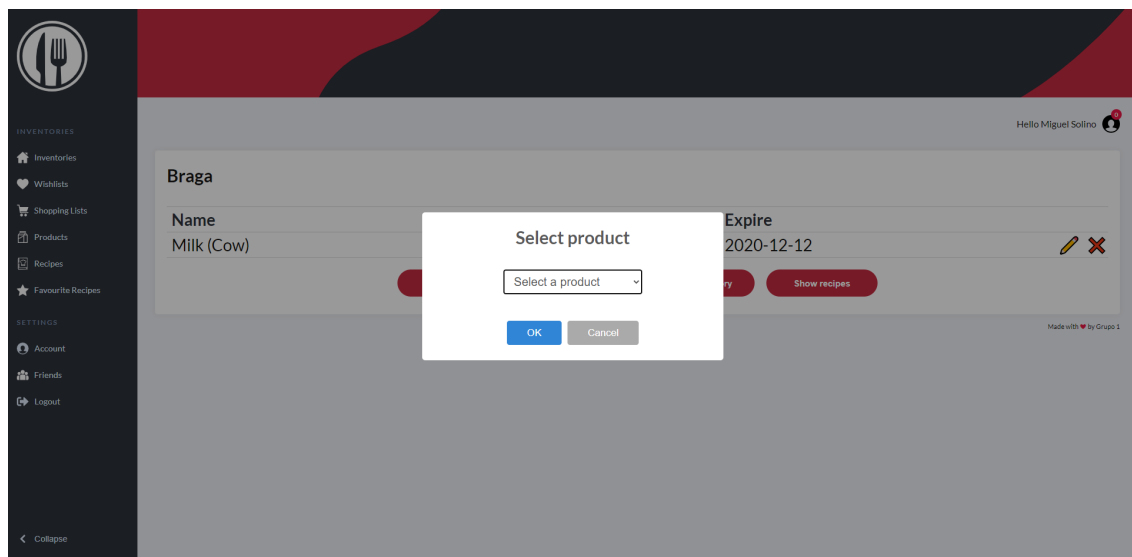
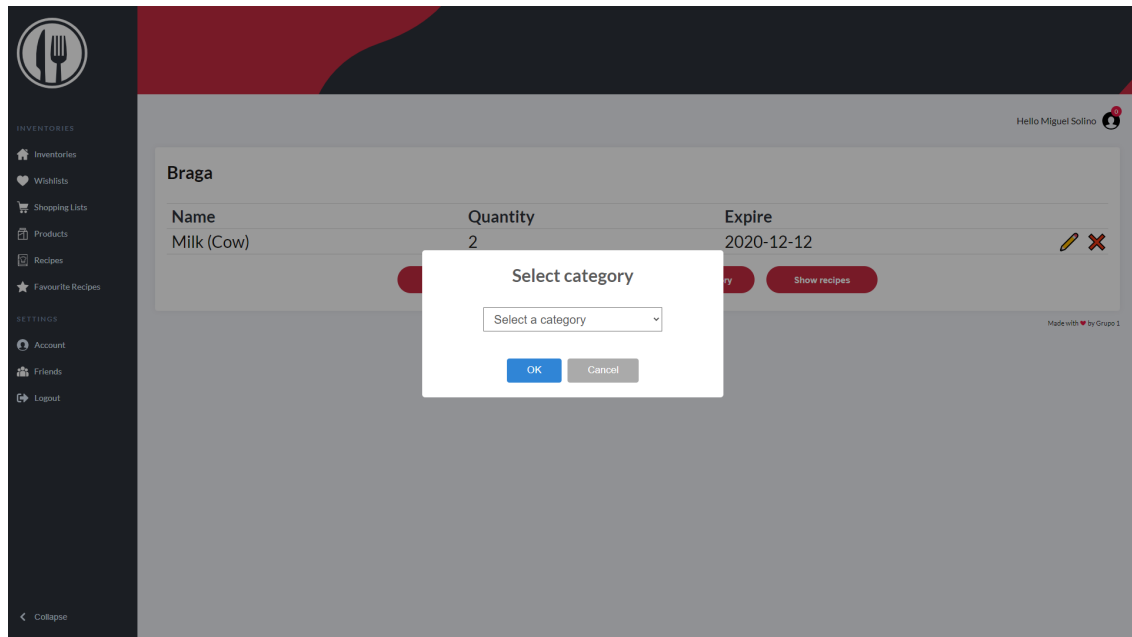
Home

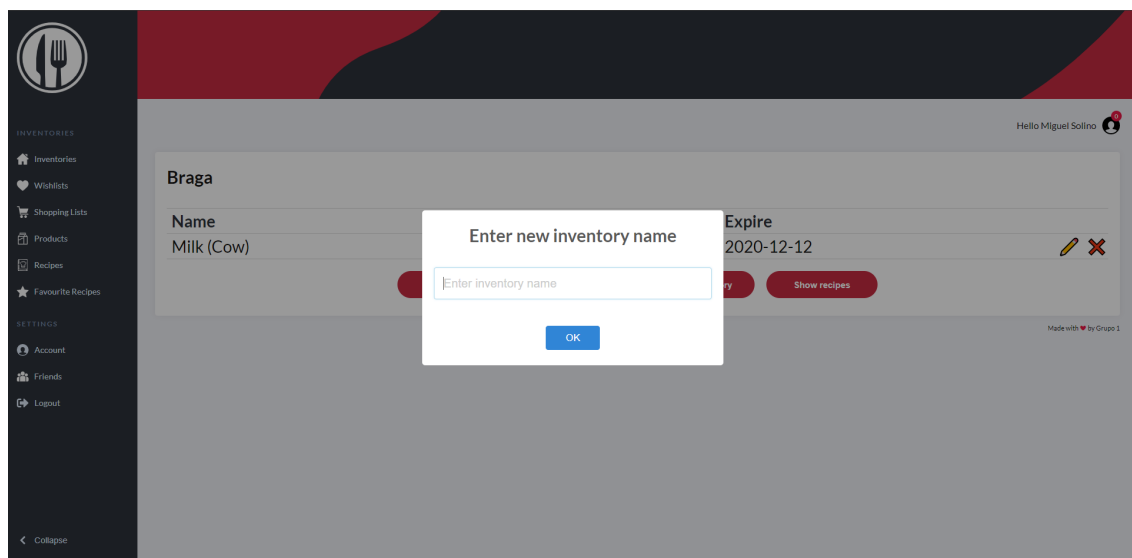
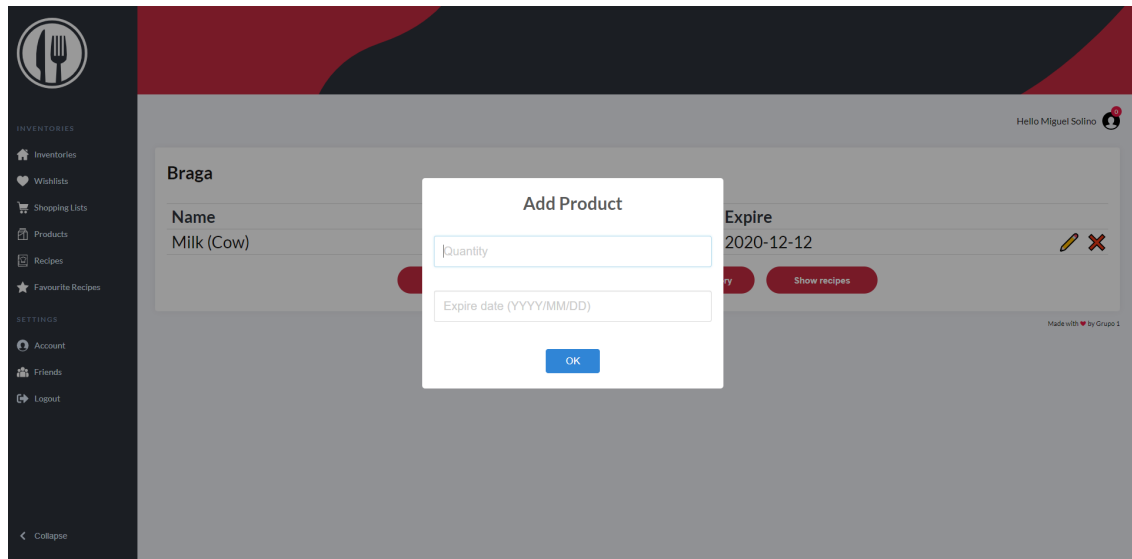
Made with by Grupo 1

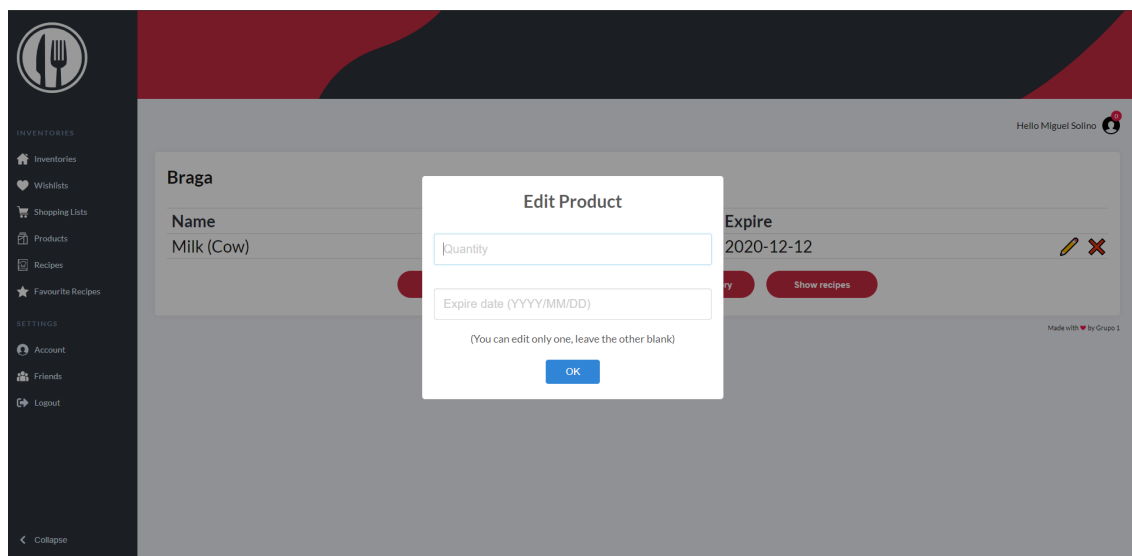
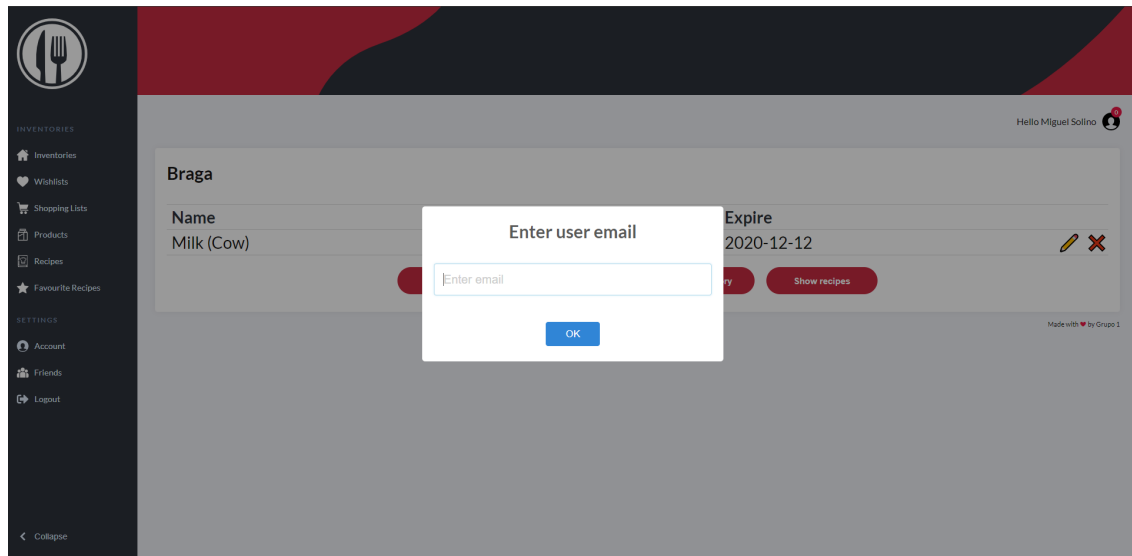
14.3 Inventário

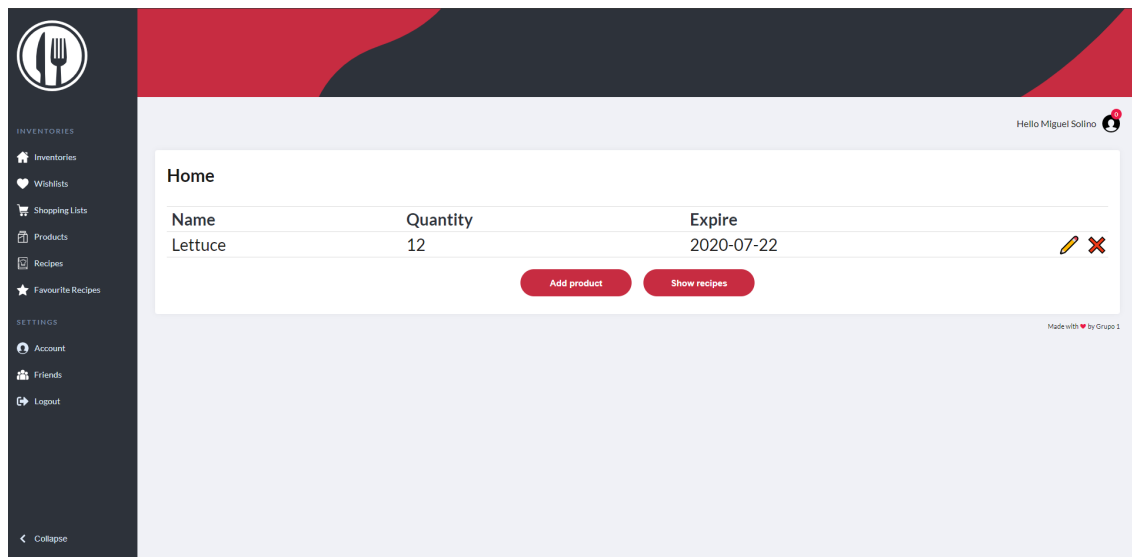
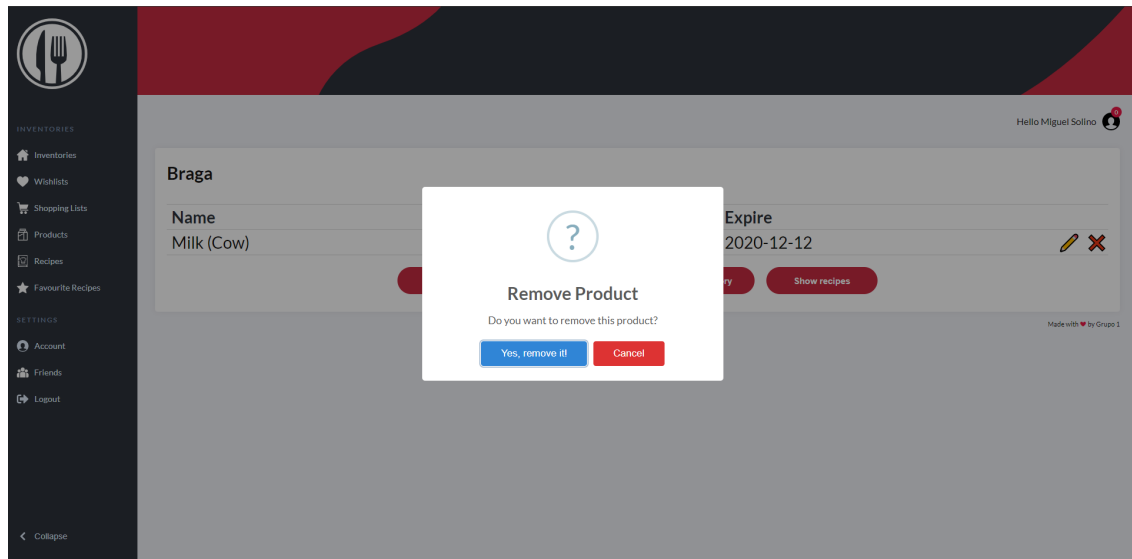
Selecionando um inventário na página inicial é nos apresentada a página abaixo com todos os produtos que possui e as opções necessárias para ser possível gerir o inventário e os produtos presentes. Nas figuras abaixo conseguimos ver o workflow de como se utiliza cada funcionalidade de um inventário.







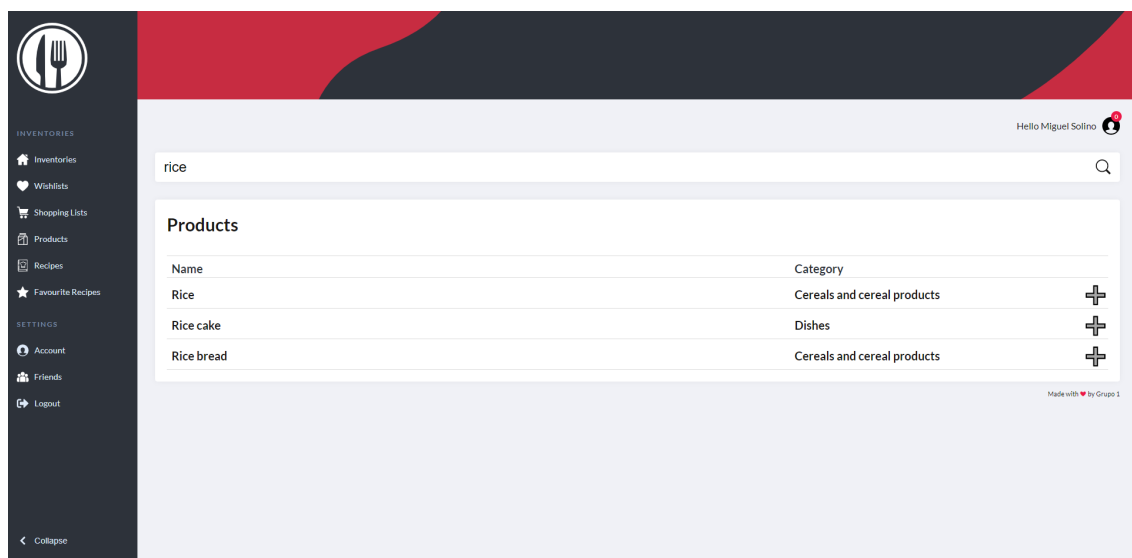
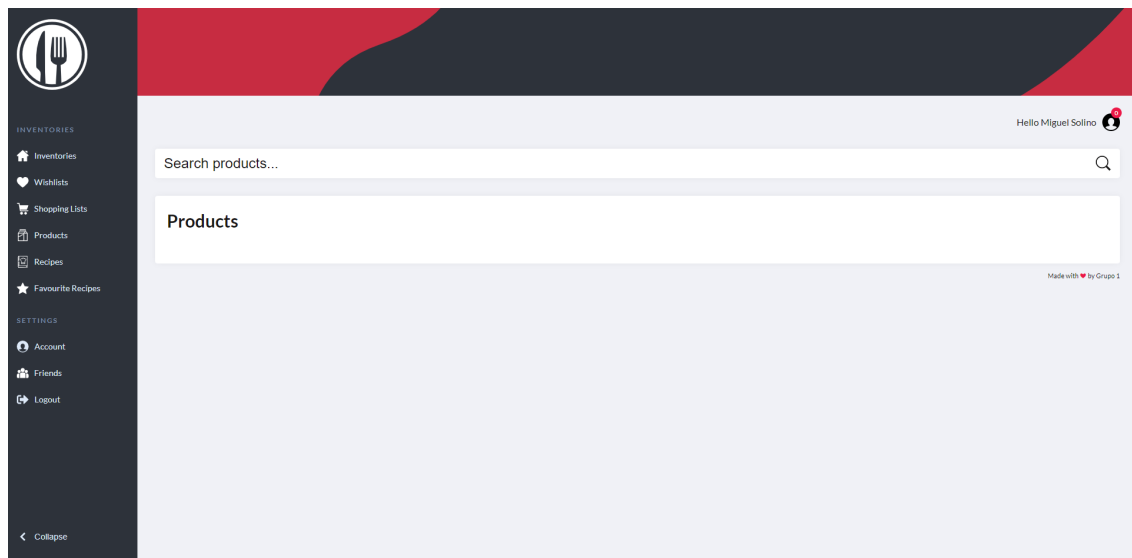




Os wishlists e as listas de compras não são apresentados aqui devido ao design e as funcionalidades implementadas em cada um deles serem minimamente semelhantes para não ser necessário apresentar capturas de ecrã.

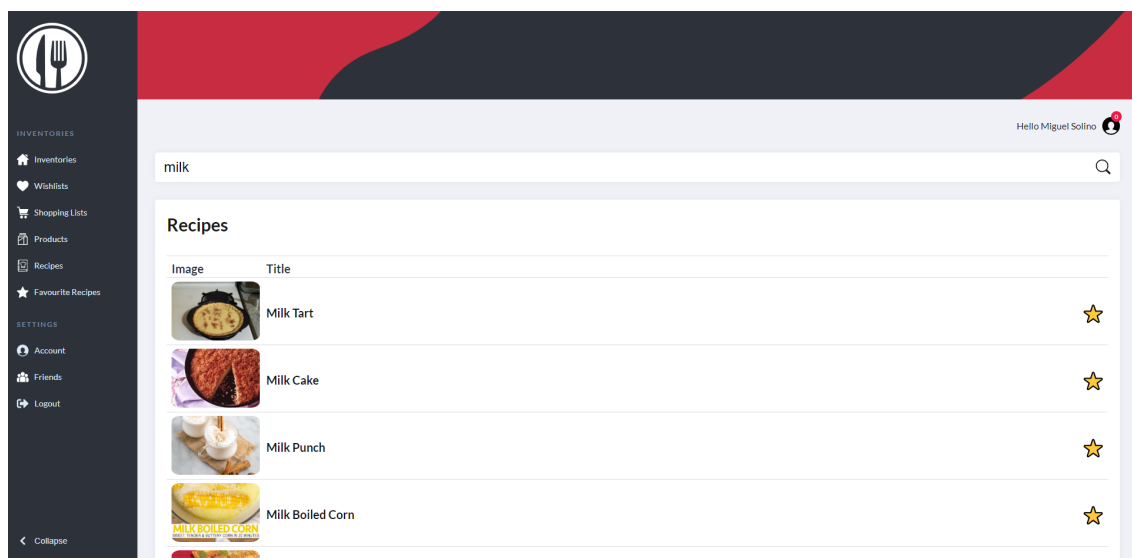
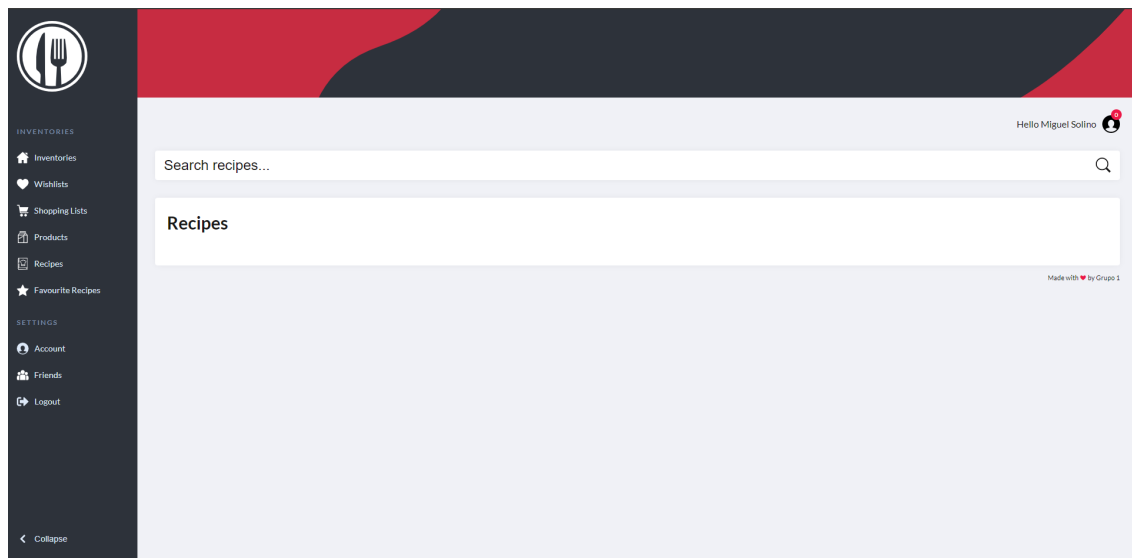
14.4 Produtos

Selecionando a opção dos **Products** do menu lateral somos redirecionados para esta página. Aqui é possível realizarmos uma pesquisa dos produtos existentes na aplicação como também adicionar os mesmos a cada uma das listas existentes (despensas, wishlists e listas de compras). Abaixo é apresentado o processo de como esta página funciona.

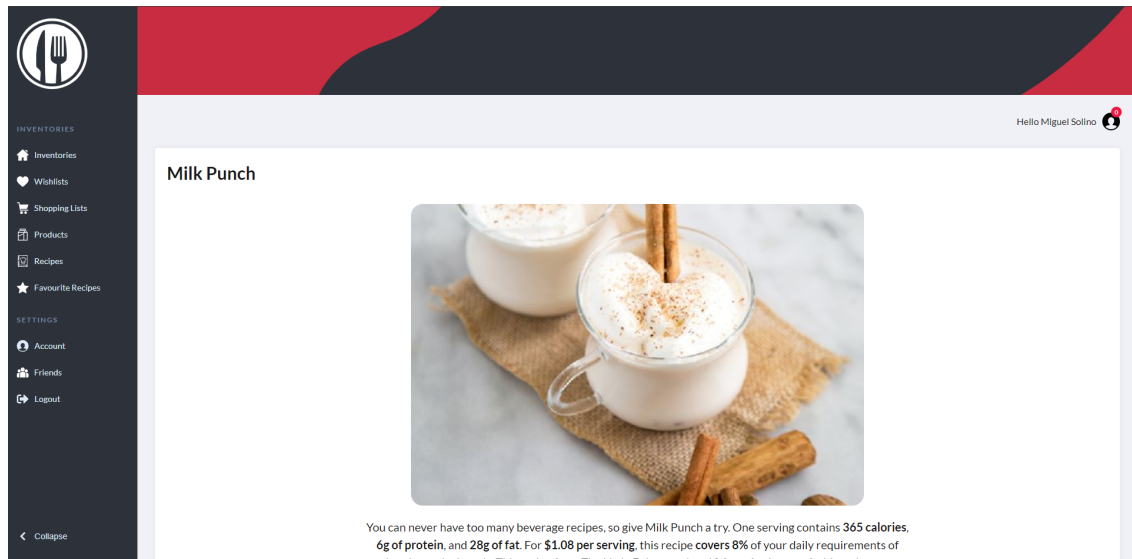


14.5 Receitas

Continuando agora para a opção de **Recipes** no menu lateral a aplicação redireciona o utilizador para a página abaixo onde poderá, semelhante à página de procura de receitas, realizar uma pesquisa das receitas existentes na aplicação e escolher as que mais gostar como favoritas para mais tarde serem apresentadas e consultadas na página de receitas favoritas do utilizador.

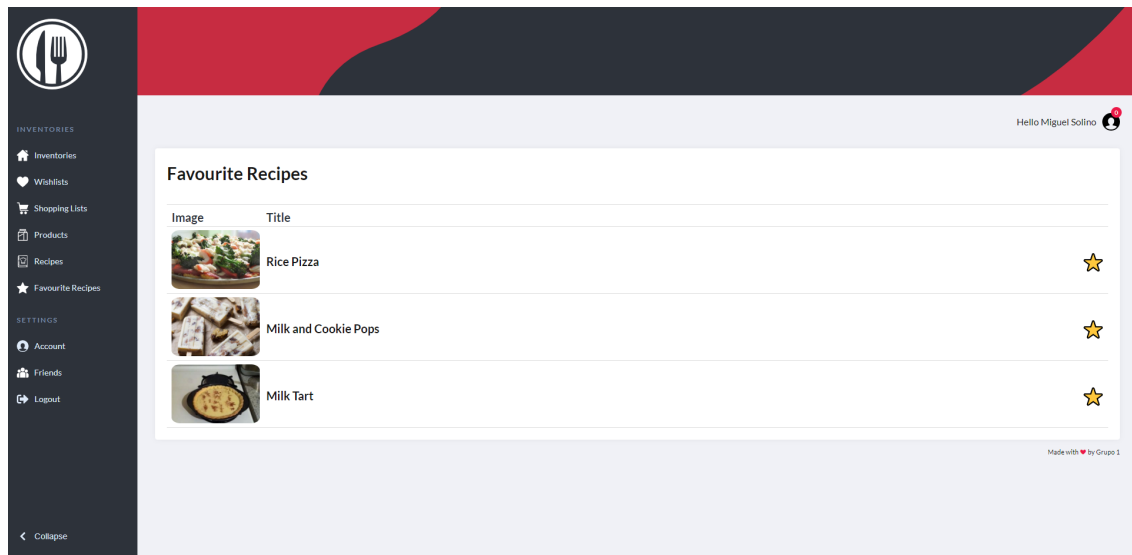


Selecionando uma receita da lista será possível ver os detalhes da mesma (imagem, sumário, ingredientes e instruções), como na figura abaixo.



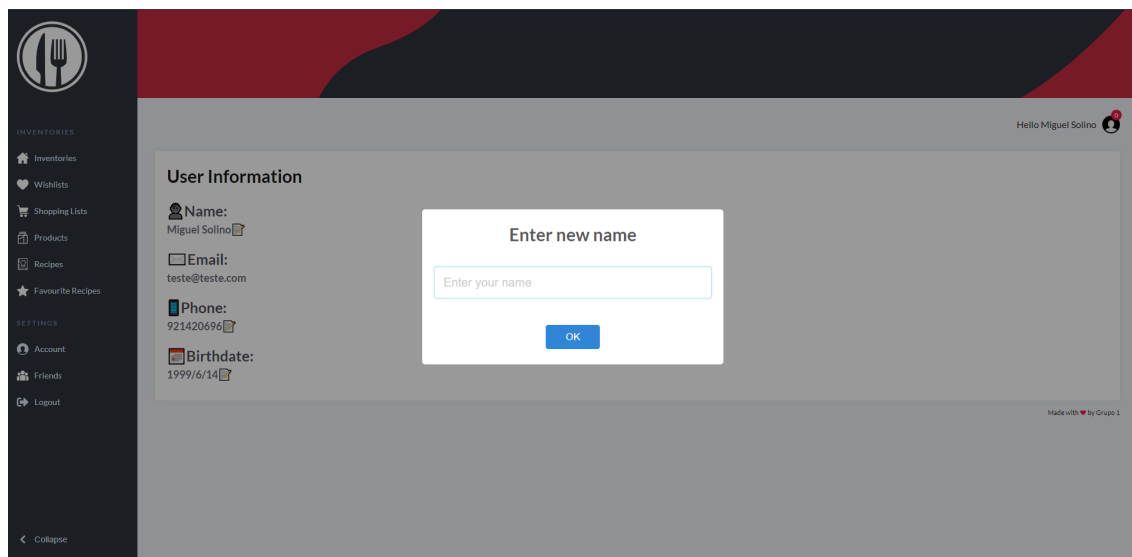
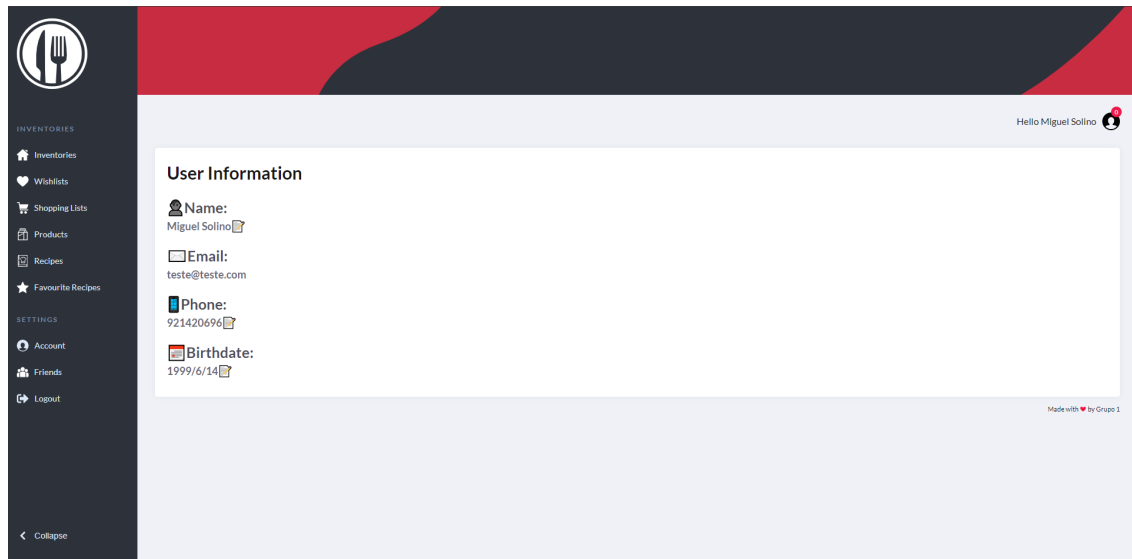
14.6 Receitas favoritas

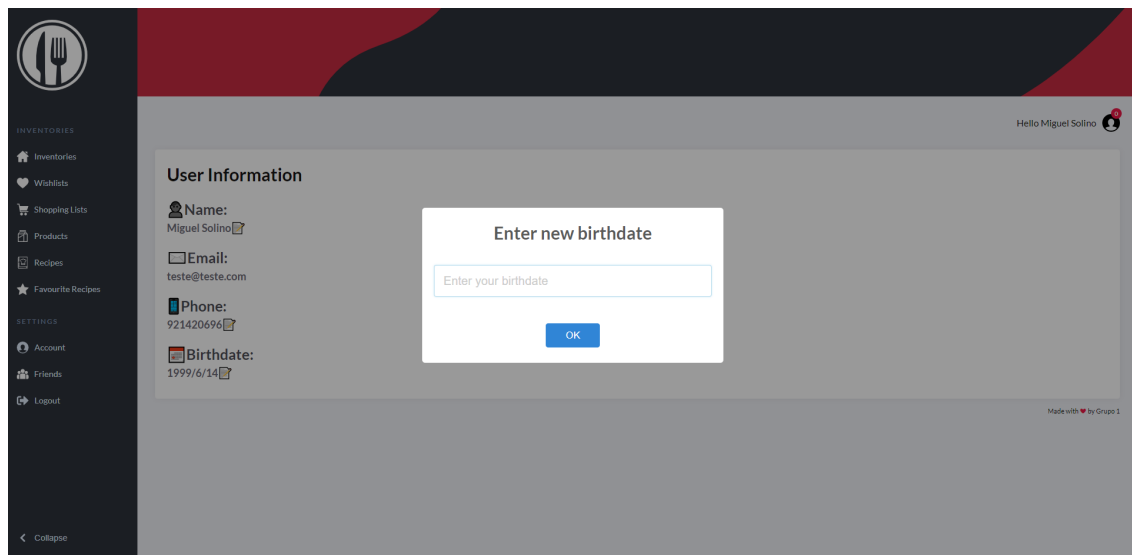
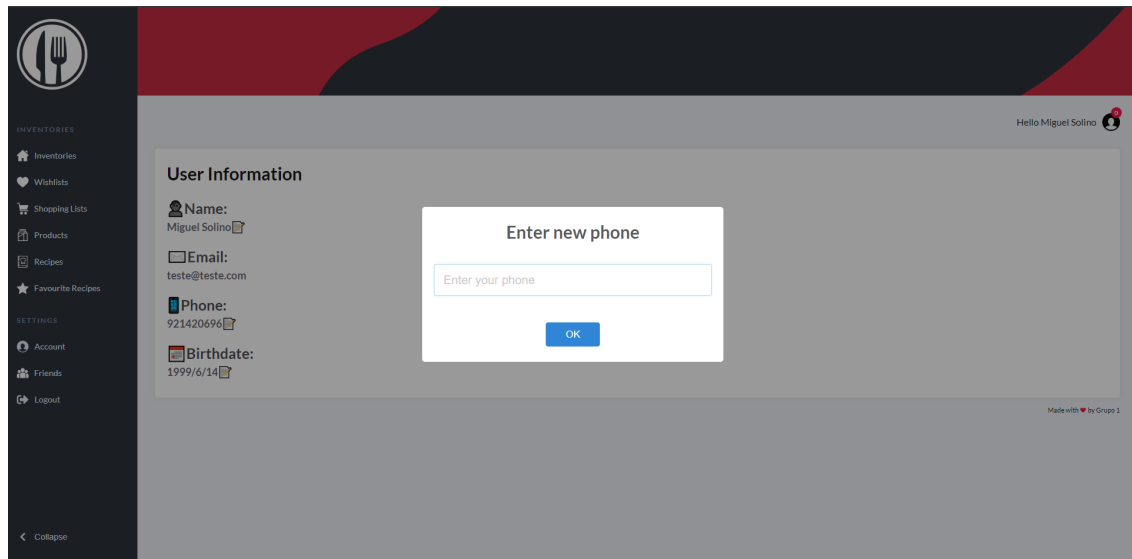
Na secção anterior foi feita uma referência à pagina de receitas favoritas de um utilizador. Esta é acessível a partir da opção **Favourite Recipes** do menu lateral e a página apresentada ao utilizador é a figura abaixo. Aqui será possível serem consultadas as receitas que o utilizador colocou como favoritas e removê-las.



14.7 Perfil

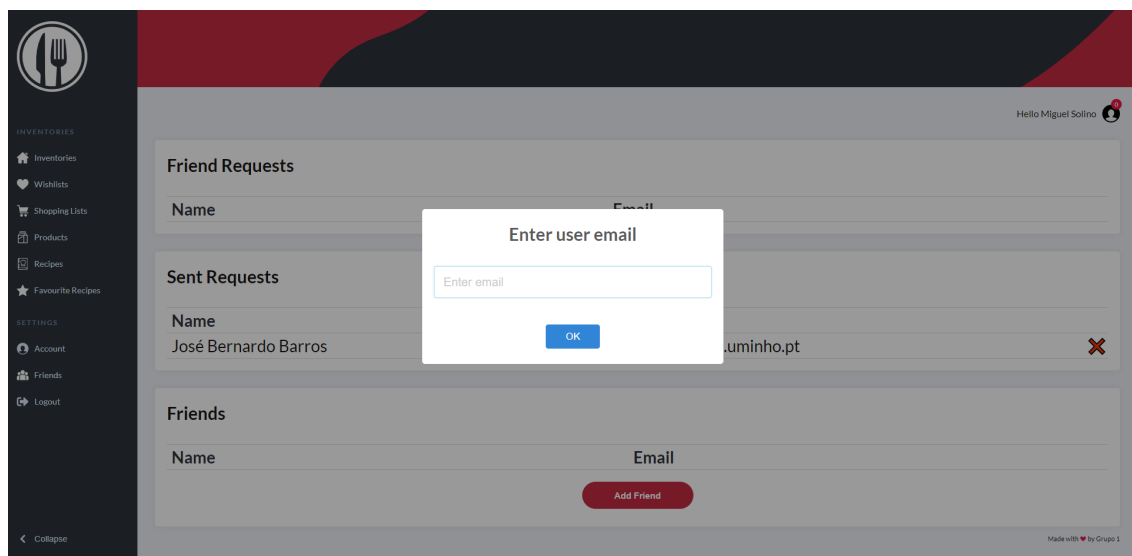
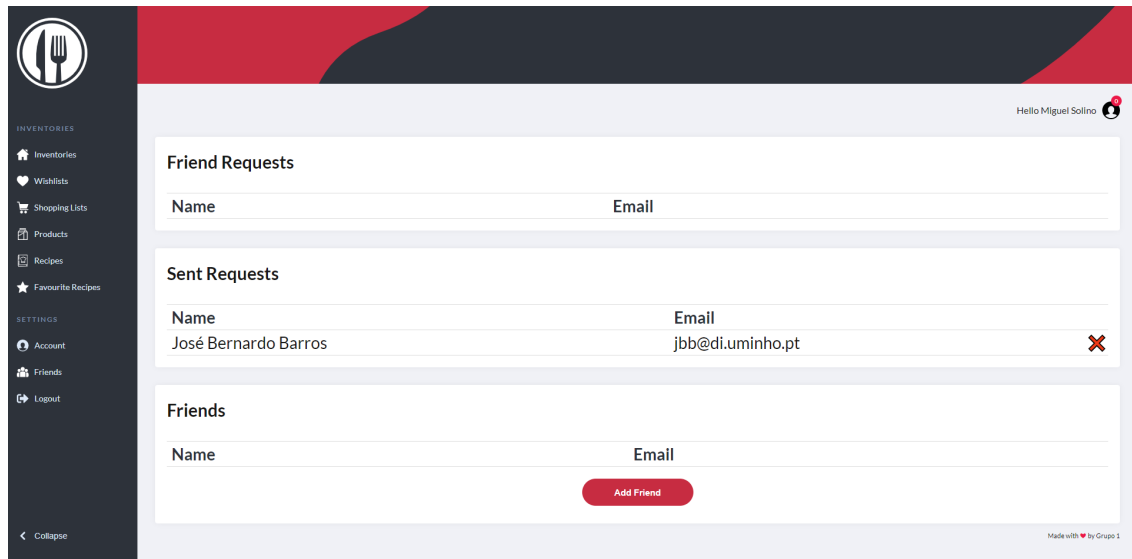
Passando à secção de **Settings** do menu lateral, e começando pelo **Account**, é apresentado ao utilizador os seus dados, nomeadamente, nome, email, número de telemóvel e data de nascimento. Todos estes dados, exceto o de email, são editáveis caso o utilizador os pretenda mudar.

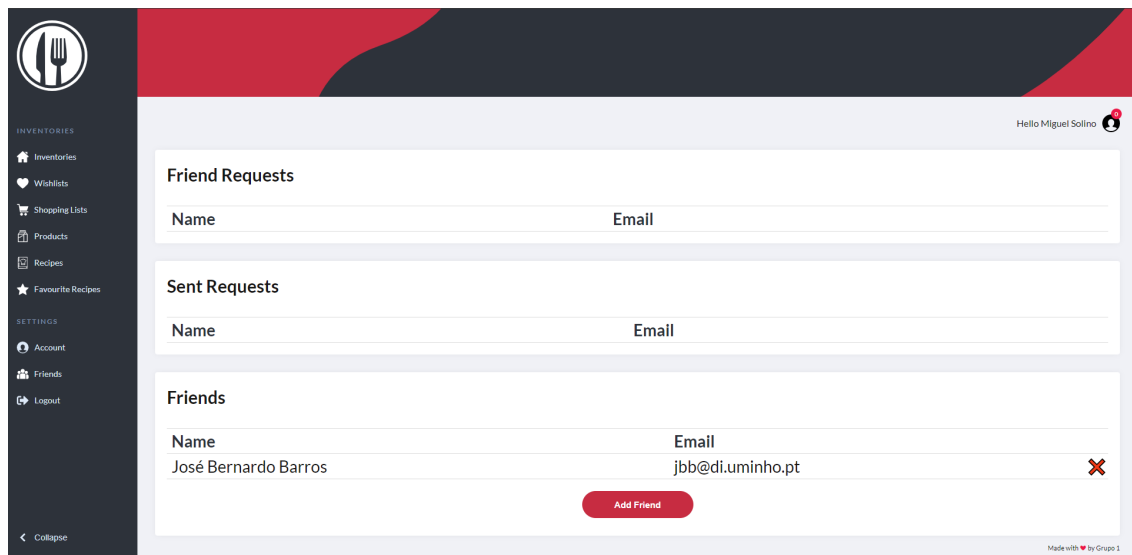
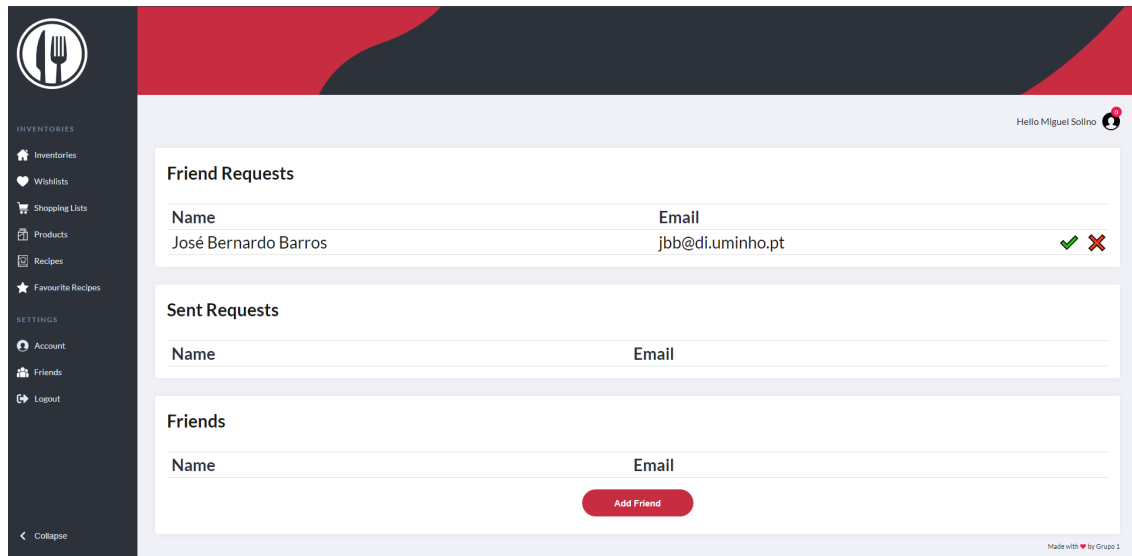




14.8 Amigos

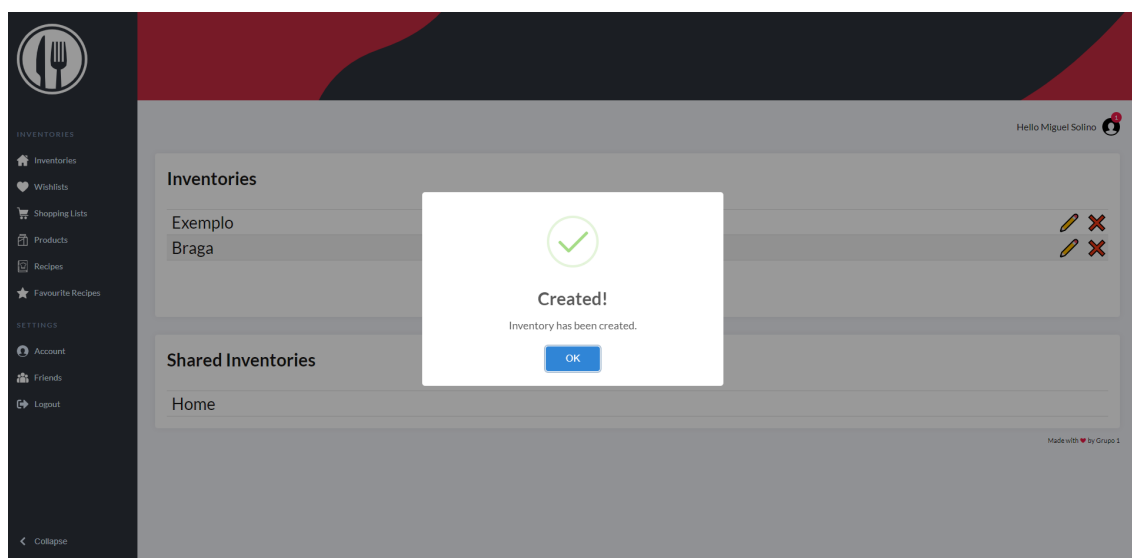
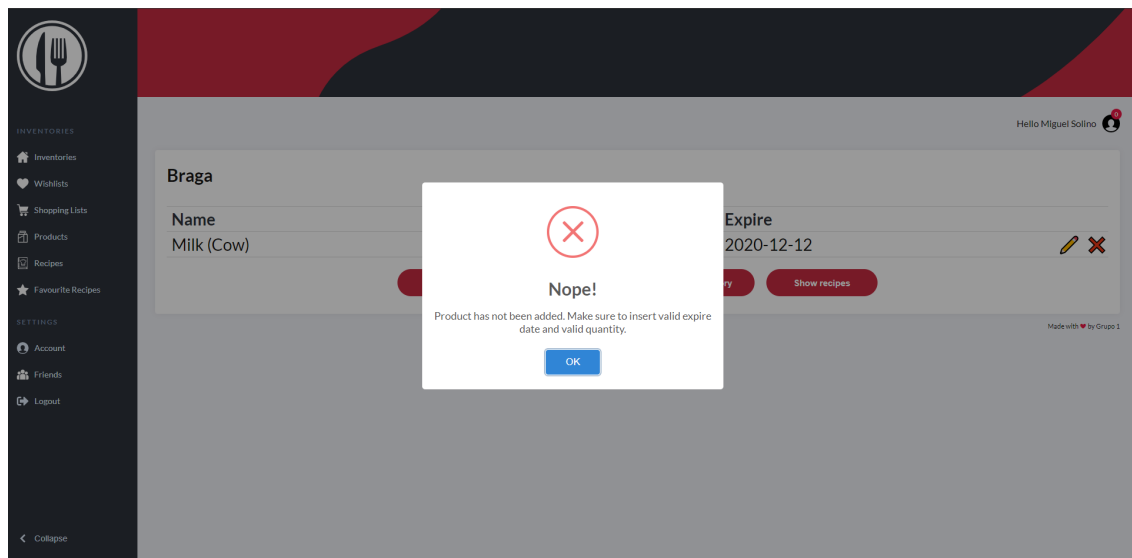
Na segunda opção da mesma secção, **Friends**, temos presente o botão que redireciona o utilizador para a sua página de amigos. Aqui é onde se poderá adicionar e remover amigos e responder a pedidos de amizade recebidos.





14.9 Popup Erro/Sucesso

Abaixo são apresentados exemplos de como são os popups de **Sucesso** e **Erro** para as diferentes funcionalidades da aplicação.



14.10 Painel de Notificações

Finalmente, abaixo está o nosso painel de notificações que apresentará notificações quando o utilizador tiver pedidos de amizade por aceitar ou produtos dos inventários perto de ficarem fora de validade ou já expirados.

Hello Miguel Solino



Notifications



Friends

You have 1 friend requests.



Inventory Braga Tuning

Muskrat expired/is expiring in less than 30 days.

Capítulo 15

Conclusões

Concluindo, atingimos todos os objetivos propostos cumprindo todas as tarefas propostas entre cada *sprint* e cumprir o calendário estipulado inicialmente no diagrama de Gantt.

Para além do melhor domínio das novas tecnologias utilizadas na produção deste *software* também ganhamos um melhor domínio sobre metodologias de desenvolvimento e organização de tarefas em equipa.

Como trabalho futuro gostaríamos de desenvolver uma aplicação mobile facilitando a utilização do produto e assim expandir o público alvo. Gostaríamos também de trabalhar melhor a integração com a API externa *spoonacular*.