INSTITUTO POLITÉCNICO DE BEJA

Escola Superior de Tecnologia e Gestão

Licenciatura em Engenharia Informática

Desenvolvimento de Aplicações Web

Projeto – Análise/Desenho e Implementação

Celso Pedro nº20345

2022

INSTITUTO POLITÉCNICO DE BEJA

Escola Superior de Tecnologia e Gestão

Licenciatura em Engenharia Informática

Desenvolvimento de Aplicações Web

Projeto – Análise/Desenho e Implementação

Elaborado por:

Celso Pedro nº20345

Docente:

PhD Luís Carlos Bruno

Relatório de projeto para a Unidade Curricular de Desenvolvimento de Aplicações Web, apresentado na

Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Beja

2022

Resumo

As fases de análise e desenho são a fase inicial do desenvolvimento de software. São, portanto, duas fases cruciais, pois é onde serão tomadas todas as decisões inerentes ao desenvolvimento do software. É na fase de análise que são identificados os requisitos do sistema, que permitirão aos utilizadores realizador tarefas desejadas. Entender as características dos utilizadores do sistema é algo de suma importância, para garantir a sua usabilidade, ou seja, a facilidade com que estes irão desenvolver as tarefas no sistema.

Na fase de desenho são tomadas decisões sobre os elementos que farão parte da interação entre o utilizador e o sistema, estes elementos devem utilizar um modelo simples, que se adeque aos utilizadores.

Na fase de implementação, é feita a programação do protótipo funcional da aplicação. O resultado deste projeto foi o protótipo funcional de uma aplicação web que permite a um utilizador publicar uma receita e aos outros consultar.

Palavras-chave: *decisão, requisitos, tarefas, utilizadores, protótipos, codificação, implementação*

Abstract

The analysis and design phases are the initial phase of software development. They are, therefore, two crucial phases, because this is where all the decisions inherent to software development will be made. It is in the analysis phase that the system requirements are identified, which will allow users to perform desired tasks. Understanding the characteristics of system users is something of paramount importance, to ensure their usability, that is, the ease with which they will develop the tasks in the system.

In the design phase decisions are made about the elements that will be part of the interaction between the user and the system, these elements should use a simple model, which is appropriate to the users.

In the implementation phase, the functional prototype of the application is programmed. The result of this project was the functional prototype of a web application that allows one user to publish a recipe and others consult.

Keywords: *decision, requirements, tasks, users, prototypes, coding, implementation*

Índice

Índice de figuras

Índice de tabelas

1. Introdução

Este relatório descreve as etapas seguidas na fase de análise e desenho do sistema web que está a ser desenvolvido no contexto do projeto da disciplina de Desenvolvimento de Aplicações Web.

O tema escolhido para o projeto é o "My Recipe World", que vai ser complementar a um projeto realizado no âmbito da unidade curricular de Tecnologias para a Web e Ambientes Móveis. O tema está voltado para a área da culinária, e a principal motivação para a escolha deste tema é melhorar o protótipo desenvolvido previamente, de modo a torná-lo ainda mais funcional e aproximado á realidade.

O sistema a ser desenvolvido permitirá, essencialmente, aos seus utilizadores publicar receitas e visualizar as receitas publicadas pelos outros utilizadores, no entanto, haverá um intermediário nesse processo, um administrador, responsável por gerir os conteúdos antes e depois destes serem publicados no sistema.

Em termos estruturais, este relatório está dividido em 2 capítulos, o primeiro que trata da análise do sistema, e o segundo que trata da fase desenho do sistema. O capítulo da análise do sistema apresenta temas referentes aos seguintes tópicos: Caraterização dos atores, que é onde serão caracterizados os atores do sistema, bem como a caracterização das suas funções em relação ao sistema e criação de personas; o diagrama dos dois casos de uso e as respetivas tabelas de especificação de requisitos na notação UML (Unified Modelling Language), identificação dos requisitos funcionais e dos requisitos não funcionais. O capítulo da fase de desenho abrange o desenvolvimento do protótipo de média fidelidade das interfaces gráficas com o utilizador: storyboards e ecrãs (wireframes) de cada caso de uso e, por fim, a Modelação da Base Dados: diagrama E/R e o modelo físico.

2. Análise do sistema

Este capítulo destina-se ao levantamento e a especificação de requisitos e de características de elementos do sistema que foram desenvolvidos durante esta fase de análise do sistema.

2.1. Caracterização dos atores do sistema

Nesta secção será feita a caracterização dos principais atores do sistema, que consiste essencialmente na identificação dos utilizadores do sistema, suas características e identificação das tarefas realizadas por estes no sistema. A identificação dos atores relevantes do sistema foi feita em pesquisas feitas na internet.

O sistema terá dois atores. O primeiro ator é o utilizador comum, que utiliza o sistema para publicar as suas receitas ou visualizar as receitas de outros utilizadores. O segundo utilizador é o administrador do website, que é definido como aquele que é responsável por administrar os conteúdos que serão mostrados aos outros utilizadores, as suas funções serão aprovar as receitas para serem visualizadas por outros utilizadores e eliminar as receitas que os utilizadores queiram eliminar, para que já não sejam mais visualizadas no sistema.

2.1.1. Caracterização do ator: Utilizador comum

O utilizador comum será o utilizador que vai publicar as suas receitas e visualizar as receitas publicadas por outros utilizadores. Este ator poderá ter idades compreendidas entre os 12 anos em diante, ser do sexo masculino ou feminino. Não é necessário que este ator possua muita experiência no domínio de informática, desde que saiba utilizar um dispositivo com acesso a internet, que suporte a utilização ou visualização de websites com recurso aos browsers. Para utilizar o sistema, este ator deverá estar autenticado com o seu nome de utilizador e palavra-passe.

2.1.2. Caracterização do ator: Administrador

O administrador do sistema será o utilizador responsável por gerir as informações/receitas publicadas no sistema. Este deverá aprovar ou não as receitas que o utilizador comum publicar, antes destas serem disponibilizadas aos outros utilizadores e gerir os conteúdos que são apresentados aos utilizadores quando estes utilizam o website. Este utilizador deve ter uma formação na área da culinária e algum conhecimento na utilização de websites.

2.1.3. Criação das Personas

Nesta secção, serão criadas duas personas representativas dos utilizadores.

- **João Silva**: 22 anos de idade, mora em Beja, é solteiro. Estudante de Enfermagem na Escola Superior de Saúde do IP Beja há 5 anos, trabalha num famoso restaurante da cidade, como ajudante de cozinha do seu pai. A sua principal responsabilidade, no seu local de trabalho, é ajudar o seu pai a guardar as receitas novas que são inventadas na cozinha do restaurante. João se adapta com facilidade às novas tecnologias, mesmo nunca tendo tido uma formação em informática, e, tem sentido, ultimamente, falta de uma aplicação que a ajude a guardar as receitas que o seu pai tem inventado.
- **Marina Margão**: 46 anos de idade, é uma cozinheira, que já ganhou vários prémios, sendo a sua maior conquista o grande prémio do Master Chef Portugal. Ela já trabalhou em conceituados restaurantes em Portugal e na Europa. Formada em Culinária, é atualmente júri de vários concursos de culinária e trabalha para a Associação de Cozinheiros de Portugal com a validação de novas receitas criadas por cozinheiros de diversos restaurantes. Marina tem conhecimentos médios a avançados de informática.

2.2. Especificação dos requisitos

Nesta secção, será feita a especificação dos requisitos funcionais que foram identificados, após a recolha de informações que foi feita, com base em pesquisas na internet e na utilização da aplicação do website TudoGostoso.

Com base nas informações obtidas, puderam ser identificados os requisitos listados nos pontos a seguir, os requisitos foram separados de acordo com os utilizadores a que se destinam.

2.2.1. Funções dos utilizadores comuns

A aplicação deve permitir aos utilizadores comuns:

- Criar uma receita, especificando os ingredientes, tempo de preparo o rendimento (porções), escolher a categoria da receita e o modo de preparação dessas receitas e, publicar esta receita, se desejar.
- Visualizar a receita que criou, bem como as receitas de outros utilizadores, que estejam publicadas no website.

2.2.2. Funções de Administrador:

A aplicação deve permitir ao administrador realizar as seguintes tarefas:

- Aprovar as receitas enviadas pelo utilizador comum e publicar as receitas que tenham sido aprovadas.

2.2.3. Diagrama de casos de uso

Nesta secção será apresentado o diagrama de casos de uso, elaborado com base nos requisitos identificados na secção anterior.

O diagrama de casos de uso da figura abaixo tem como ator principal o utilizador comum, que será quem vai muitas das vezes iniciar a interação com o sistema, sendo que o administrador é o ator secundário, uma vez que vai realizar tarefas complementares às do utilizador comum.

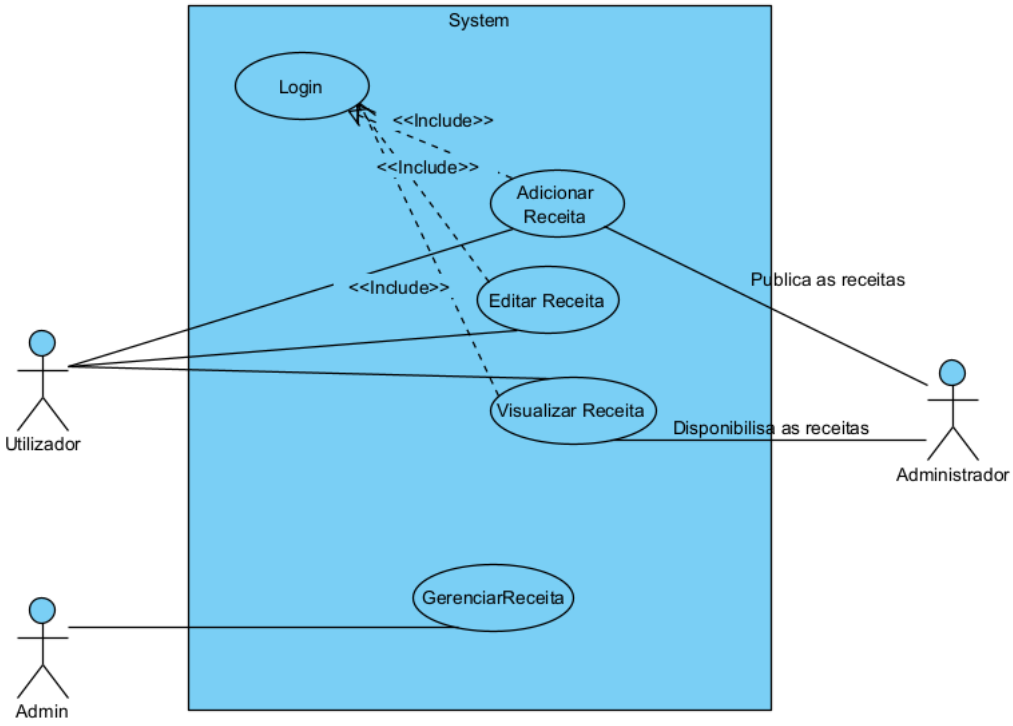

Figura 1. Diagrama de casos de uso

2.2.4. Descrição dos casos de uso

Nesta subsecção será feita a especificação dos casos de uso apresentados no diagrama apresentado na subsecção anterior.

2.2.5. Caso de uso adicionar receita

<table>
<tr><td>Nome</td><td colspan="3">Adicionar receita</td></tr>
<tr><td>Descrição</td><td colspan="3">O utilizador utiliza o sistema para adicionar uma receita.</td></tr>
<tr><td>Pré-condições</td><td colspan="3">O Utilizador tem de efetuar Login</td></tr>
<tr><td>Pós-condições</td><td colspan="3">Os outros cozinheiros poderão ver a receita publicada</td></tr>
<tr><td>Atores</td><td colspan="3">Utilizador, Administrador</td></tr>
<tr><td>Cenário principal</td><td colspan="3"><table>
<tr><td></td><td>Ação do ator</td><td>Resposta do sistema</td></tr>
<tr><td>1</td><td>O utilizador informa ao sistema que pretende adicionar uma receita</td><td></td></tr>
<tr><td>2</td><td></td><td>O sistema recebe o pedido do utilizador e espera que ele adicione todas as informações da receita: nome, ingredientes, tempo de preparo, rendimento, categoria, modo de preparo e se quer que a sua receita seja tornada pública</td></tr>
<tr><td>3</td><td>O Utilizador fornece os dados pedidos pelo sistema e diz ao sistema que quer que a sua receita seja tornada pública</td><td></td></tr>
<tr><td>4</td><td></td><td>O sistema recebe o pedido do utilizador e regista a receita e as escolhas que o utilizador fez</td></tr>
<tr><td>5</td><td></td><td>O sistema vai informa ao Utilizador que a sua receita aguarda validação para ser publicada.</td></tr>
<tr><td>6</td><td>O administrador é informado que uma nova receita acaba de</td><td></td></tr>
</table></td></tr>
</table>

		ser publicada e que necessita da sua aprovação	
	7	O administrador analisa a receita enviada pelo sistema e aprova a receita	
	8		O sistema recebe a indicação da aprovação da receita do utilizador e informa ao cozinheiro que a sua receita foi aprovada.
	9		O sistema disponibiliza a receita do cozinheiro aos outros utilizadores do sistema, para serem consultadas
Cenário alternativo	1		O sistema recebe o pedido do utilizador e regista a receita e as escolhas que o utilizador fez e O sistema vai informa ao Utilizador que a sua receita aguarda validação para ser publicada.
	2	O administrador verifica que a receita enviada pelo utilizador está incompleta e rejeita essa receita	
	3		O sistema recebe a informação de que a receita publicada pelo utilizador fora rejeitada e notifica ao utilizador tal facto
	4	O utilizador é notificado da rejeição da sua receita, com indicação do motivo	

Tabela 1: Caso de uso adicionar receita

2.2.6. Caso de uso editar receita

<table>
<tr><td>Nome</td><td colspan="3">Editar Receita</td></tr>
<tr><td>Descrição</td><td colspan="3">O utilizador utiliza o sistema para editar uma receita que já tenha enviado</td></tr>
<tr><td>Pré-condições</td><td colspan="3">1. O utilizador tem de efetuar Login
2. Terá de ter efetuado o caso de uso adicionar receita</td></tr>
<tr><td>Pós-condições</td><td colspan="3">A receita publicada será modificada</td></tr>
<tr><td>Atores</td><td colspan="3">Utilizador</td></tr>
<tr><td>Cenário principal</td><td colspan="3"><table>
<tr><td></td><td>Ação do ator</td><td>Resposta do sistema</td></tr>
<tr><td>1</td><td>O utilizador informa ao sistema que pretende editar uma receita</td><td></td></tr>
<tr><td>2</td><td></td><td>O sistema busca todas as receitas enviadas pelo utilizador e pede para ele selecionar a receita que deseja editar</td></tr>
<tr><td>3</td><td>O utilizador escolhe a receita que pretende editar</td><td></td></tr>
<tr><td>4</td><td></td><td>O sistema mostra a receita selecionada pelo utilizador, no modo de edição</td></tr>
<tr><td>5</td><td>O utilizador faz as alterações que pretende e guarda a receita</td><td></td></tr>
<tr><td>6</td><td></td><td>O sistema recebe a interação do utilizador e atualiza a receita enviada pelo utilizador.</td></tr>
</table></td></tr>
<tr><td>Cenário alternativo</td><td colspan="3"><table>
<tr><td>1</td><td>O utilizador informa ao sistema que pretende editar uma receita</td><td></td></tr>
<tr><td>2</td><td></td><td>O sistema busca todas as receitas enviadas pelo utilizador e pede para ele selecionar a receita que deseja editar</td></tr>
<tr><td>3</td><td>O utilizador escolhe a receita que pretende eliminar</td><td></td></tr>
</table></td></tr>
</table>

	4		O sistema mostra a receita selecionada pelo utilizador, no modo de edição
	5	O utilizador informa ao sistema que pretende eliminar aquela receita	
	6		O sistema recebe a interação do utilizador, informa que a ação é irreversível e pergunta se ele deseja continuar.
	7	O utilizador confirma a sua escolha	
	8		O sistema recebe a escolha do utilizador e informa ao administrador que a receita em questão foi eliminada e informa ao utilizador a sua receita será eliminada
	9	O administrador a informação do sistema e remove a visualização dos outros utilizadores e procede a eliminação da receita do sistema	
	10	1.	O sistema informa ao utilizador que a sua receita foi eliminada

Tabela 2: Caso de uso editar receita

2.2.7. Caso de uso visualizar receita

Nome	Visualizar receita
Descrição	O Utilizador utiliza o sistema para visualizar as receitas de outros utilizadores
Pré-condições	- O Utilizador deve esta autenticado

<table>
<tr><td>Pós-condições</td><td colspan="3">---</td></tr>
<tr><td>Atores</td><td colspan="3">Utilizador</td></tr>
<tr><td>Cenário principal</td><td></td><td>Ação do ator</td><td>Resposta do sistema</td></tr>
<tr><td></td><td>1</td><td>O utilizador requisita que o sistema lhe mostre as receitas dos outros utilizadores</td><td></td></tr>
<tr><td></td><td>2</td><td></td><td>O sistema busca por todas as receitas que tenham sido aprovadas pelo administrador e as mostra ao utilizador</td></tr>
<tr><td></td><td>3</td><td>O utilizador visualiza todas as receitas que foram aprovadas e escolhe uma para visualizar</td><td></td></tr>
<tr><td></td><td>4</td><td></td><td>O sistema carrega a receita escolhida pelo utilizador</td></tr>
<tr><td></td><td>5</td><td>O utilizador visualiza a receita que escolheu</td><td></td></tr>
<tr><td>Cenário alternativo</td><td>1</td><td>O utilizador informa ao sistema que pretende visualizar uma receita enviada por si</td><td></td></tr>
<tr><td></td><td>2</td><td></td><td>O sistema busca todas as receitas enviadas pelo utilizador e pede para ele selecionar a receita que deseja visualizar</td></tr>
<tr><td></td><td>3</td><td>O utilizador escolhe a receita que pretende visualizar</td><td></td></tr>
<tr><td></td><td>4</td><td></td><td>O sistema exibe a receita selecionada pelo utilizador</td></tr>
<tr><td></td><td>5</td><td>O utilizador visualiza a receita enviada por si</td><td></td></tr>
</table>

Tabela 3. Caso de uso visualizar receita

2.2.8. Caso de uso gerenciar receita

<table>
<tr><td>Nome</td><td colspan="3">Gerenciar receita</td></tr>
<tr><td>Descrição</td><td colspan="3">O Administrador utiliza o sistema para gerenciar as receitas dos utilizadores</td></tr>
<tr><td>Pré-condições</td><td colspan="3">- O Administrador deve esta autenticado
- Pelo menos um utilizador deve ter realizado o caso de uso adicionar receita</td></tr>
<tr><td>Pós-condições</td><td colspan="3">Os outros utilizadores poderão visualizar as receitas aprovadas</td></tr>
<tr><td>Atores</td><td colspan="3">Administrador</td></tr>
<tr><td>Cenário principal</td><td></td><td>Ação do ator</td><td>Resposta do sistema</td></tr>
<tr><td></td><td>1</td><td>O utilizador administrador pede que o sistema lhe mostre as receitas dos utilizadores que ainda não tenham sido aprovadas</td><td></td></tr>
<tr><td></td><td>2</td><td></td><td>O sistema busca por todas as receitas que ainda não tenham sido aprovadas pelo administrador e as mostra ao administrador</td></tr>
<tr><td></td><td>3</td><td>O administrador visualiza todas as receitas que aguardam aprovação e escolhe a receita que deseja analisar</td><td></td></tr>
<tr><td></td><td>4</td><td></td><td>O sistema carrega a receita escolhida pelo administrador</td></tr>
<tr><td></td><td>5</td><td>O administrador visualiza a receita enviada pelo utilizador, verifica que esta reúne as condições mínimas para ser aprovada e aprova a receita enviada pelo utilizador</td><td></td></tr>
</table>

	6		O sistema informa ao utilizador que a sua receita foi aprovada e disponibiliza a receita para que outros utilizadores a possam consultar
Cenário alternativo	1	O administrador verifica que a receita enviada pelo utilizador não reúne as condições mínimas para ser publicada e não aprova a receita	
	2		O sistema notifica ao utilizador que a sua receita não reúne as condições mínimas para ser aprovada e remove a marca a receita como não aprovada
	3	O utilizador recebe a informação de que a sua receita não foi aprovada, mas que ainda continuará visível para si	

Tabela 4. Caso de uso gerenciar receita

2.2.9. Requisitos não funcionais

Nesta secção serão os requisitos não funcionais que foram identificados, com base nas normas ABNT NBR ISO/IEC 27002. [1]

RNF 1 - Disponibilidade

As informações armazenadas em Mídias que precisam estar disponíveis por muito tempo (em conformidade com as especificações dos fabricantes) devem ser também armazenadas em outro local para evitar perda de informações devido à deterioração das Mídias. [1]

RNF 2 – Autenticação

- Devem ser aplicadas tecnologias aplicadas para segurança de serviços de redes com autenticação, encriptação e controles de conexões de rede. [1]

- Deve-se utilizar um identificador de usuário (ID de usuário) único para assegurar a responsabilidade de cada usuário por suas ações; convém que o uso de grupos de ID somente seja permitido onde existe a necessidade para o negócio ou por razão operacionais, e isso seja aprovado e documentado. [1]

RNF 3 - Segurança

Deve-se solicitar aos usuários a assinatura de uma declaração, para manter a confidencialidade de sua senha pessoal e das senhas de grupos de trabalho, exclusivamente com os membros do grupo; esta declaração assinada pode ser incluída nos termos e condições da contratação. [1]

3. Desenho do sistema

Nesta fase são apresentadas as decisões tomadas na fase de desenho da interface do sistema. Primeiramente será feita uma breve descrição do modelo de desenho adotado e finalmente, é apresentado o protótipo não funcional desenvolvido com base no modelo escolhido e os aspetos relativos à modelação e desenho da base de dados relacional.

2.3. Modelo de desenho

Nesta subseção, será descrito o modelo de desenho a ser adotado para o projeto.

O modelo de desenho que será seguido é o desenho plano (flat design). O desenho plano é um modelo de desenho popular, definido pela ausência de efeitos visuais brilhantes ou tridimensionais nos elementos gráficos de uma página da web. É considerado por muitos designers como um desenho minimalista. [2]

2.4. Modelação de Interfaces Gráficas com o Utilizador

Em seguida são descritos os modelos desenvolvidos relativos ao protótipo de baixa/média fidelidade das interfaces gráficas com o utilizador, compostos por um ou mais *storyboard* e pela proposta de ecrãs (*wireframes*) da aplicação. Os *storyboards* desenvolvidos permitem perceber genericamente a sequência de ações de navegação do utilizador no sistema. As propostas de interfaces gráficas com o utilizador são agrupadas por caso de uso e mostram o *layout* e o conteúdo de apresentação de cada página Web, bem como, a forma como o utilizador interage com ela. As wireframes e os storyboards criados podem também ser consultados em anexo a este relatório, nas pastas wireframes e storyboards, respetivamente.

2.4.1. Storyboards

2.4.2. Storyboard caso de uso Adicionar Receita - Utilizador

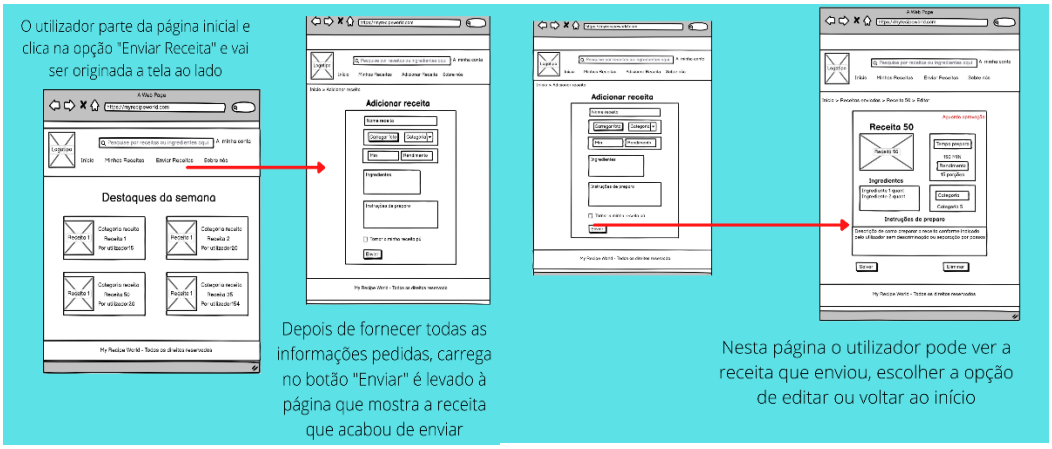

Figura 2. Storyboard caso de uso Adicionar Receita - Utilizador

2.4.3. Storyboard do caso de uso Gerenciar Receita – Administrador

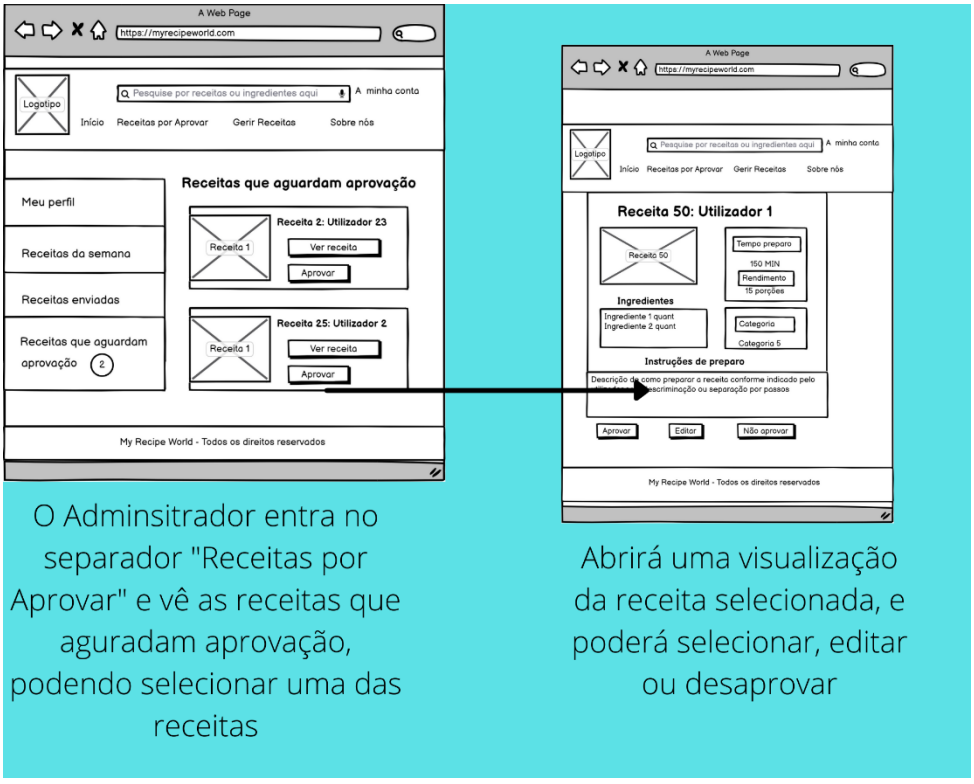

Figura 3. Storyboard do caso de uso Gerenciar Receita – Administrador

2.4.4. Storyboard do caso de uso Editar Receita

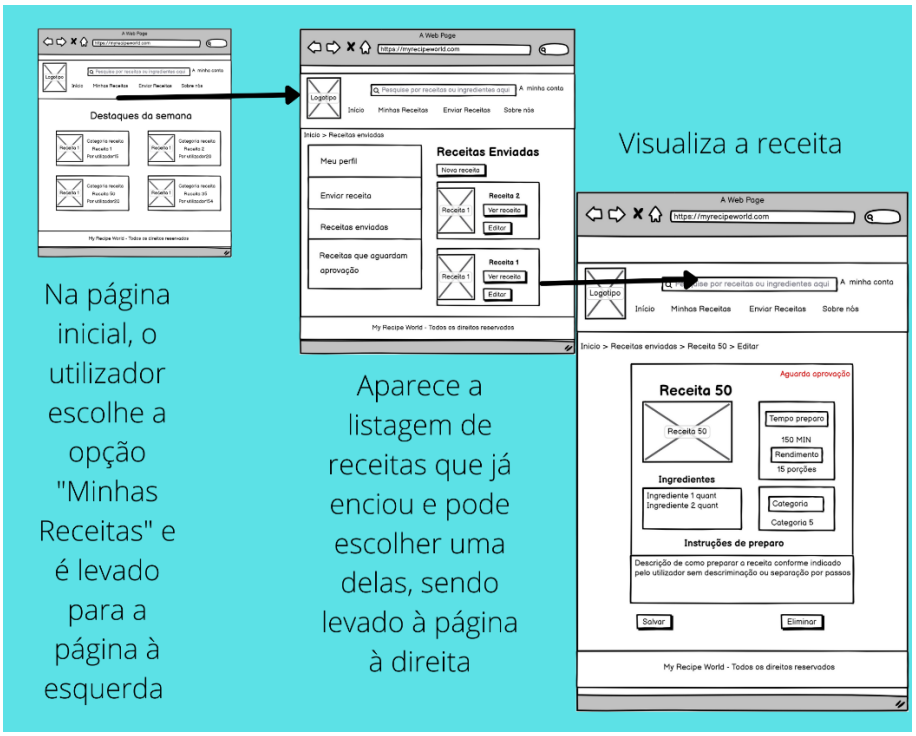

Figura 4. Storyboard do caso de uso Editar Receita

2.4.5. Interfaces caso de uso Adicionar Receita

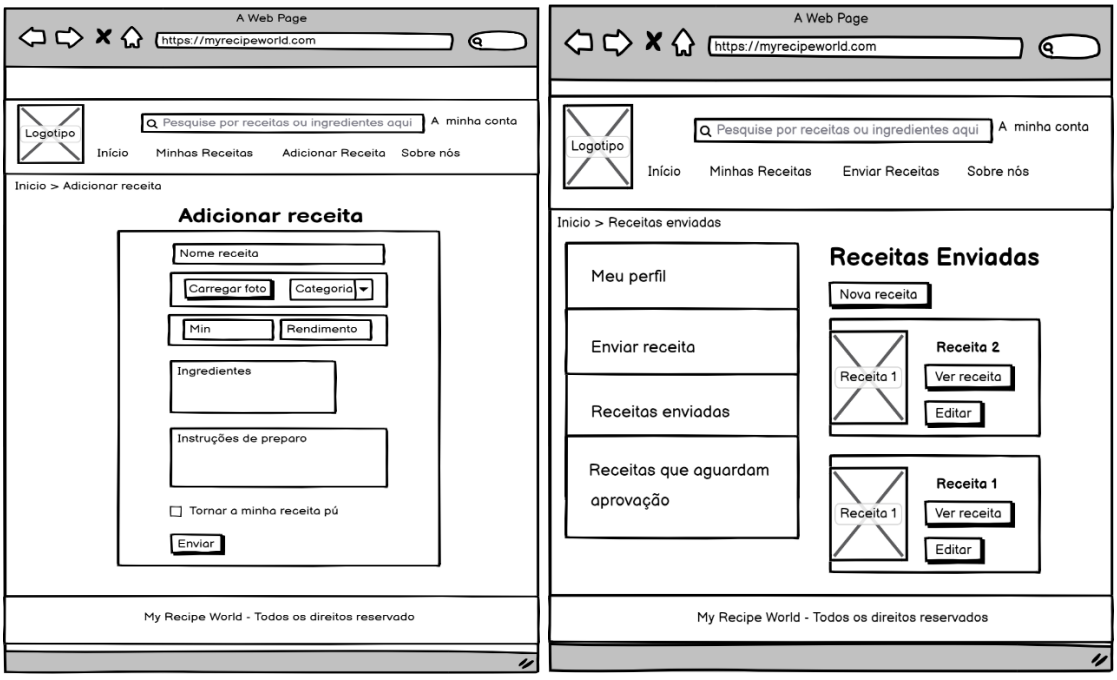

Figura 5. Interfaces caso de uso Adicionar Receita

2.4.6. Interfaces do caso de uso Editar Receita / Visualizar Receita

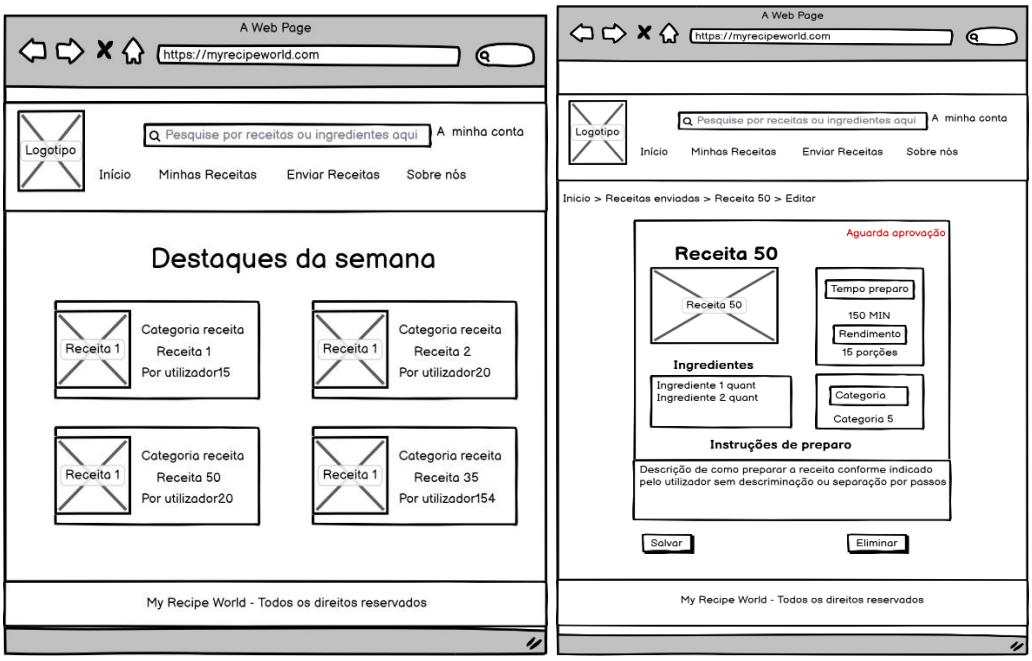

Figura 6. Interfaces do caso de uso Editar Receita

2.4.7. Interface do caso de uso Gerenciar Receita

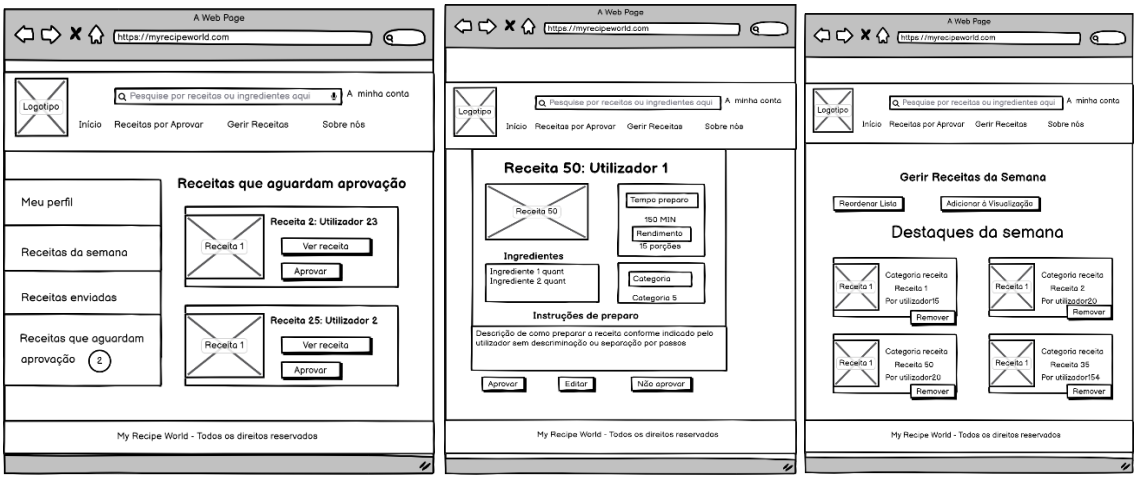

Figura 7. Interface do caso de uso Gerenciar Receita

2.5. Modelação da base de dados relacional

Nesta seção será apresentado o processo da modelação da base de dados relacional do sistema a ser desenvolvido. Inicialmente, será feita a identificação das possíveis entidades da base de dados e seus atributos, posteriormente, será desenhado um diagrama entidade-relação, seguido da apresentação do modelo relacional, a normalização, e redesenho do

diagrama entidade-relação normalizado, se necessário. Por fim, será apresentado o modelo físico da base de dados.

2.5.1. Identificação das entidades

Antes de se fazer a identificação das entidades, será feita uma pequena abordagem sobre os elementos constituintes de um diagrama E/R.

Uma **Entidade** é qualquer objeto ou conceito com interesse para a organização, sobre o qual se quer guardar informação e que possa ser identificável de forma inequívoca, como por exemplo, uma pessoa, uma fatura, uma organização, etc. As entidades possuem **Atributos,** que são propriedades que caracterizam uma entidade. Uma entidade pode ter mais do que um atributo, sendo que necessita de ter uma **Chave Primária,** que são os atributos de uma entidade que identificam, de forma inequívoca, uma ocorrência específica dessa entidade, distinguindo-a das restantes e outros, normalmente são identificados com um sublinhado ou colocados a negrito. Os **Atributos Descritores** são os atributos que apenas descrevem ou caracterizam as ocorrências de uma entidade. O diagrama E/R é uma interligação relevante entre entidades do sistema, essa interligação é denominada por **Associação**. [3]

As entidades relevantes que foram identificadas para o sistema são:

- **Utilizador**, para guardar as informações dos utilizadores do sistema;
- **Administrador**, para guardar as informações dos administradores do sistema;
- **Receita**, que guarda as informações do sistema,

Tendo sido identificadas as entidades relevantes do sistema, foi desenhado o modelo E/R, que é apresentado na figura abaixo.

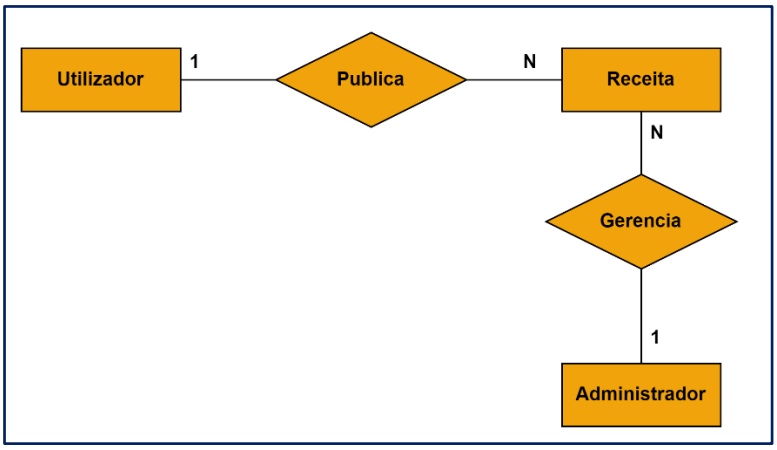

Figura 8. Modelo E-R antes da normalização

Com base no modelo E/R, foi possível fazer o modelo relacional dos dados como é apresentado a seguir.

- **Utilizador** (idUtilizador, nomeUtilizador, emailUtilizador, idade);
- **Administrador** (idAdmin, nomeAdmin, emailAdmin);
- **Receita** (idReceita, nomeReceita, categoria, rendimento, tempoPreparo, ingredientes, modoPreparo, estadoReceita, publicidadeReceita, idUtilizador, idAdmin). Os atributos estadoReceita e publicidadeReceita são para identificar se a receita foi aprovada ou não e se a receita é pública ou não, respetivamente.

2.5.2. Desenho do Modelo E/R

2.5.3. Normalização

Após a construção do modelo conceptual de dados foram identificadas algumas redundâncias, portanto seguiu a fase de **normalização dos dados**, com o objetivo de transformar o modelo E/R o menos redundante possível.

Fez-se a normalização até à 3º forma normal, na **primeira forma normal**, verificou-se que os valores de alguns atributos não eram atómicos, ou seja, teriam de ser colados múltiplos valores na mesma linha, como é o caso do atributo dos ingredientes. [4] Portanto, esse atributo foi removido, originando deste, a tabela **Ingrediente** (idIngrediente, nomeIngrediente, unidadeMedida). Esta tabela estaria relacionada com a tabela **Receita,** mas como formavam uma relação de muitos para muitos, foi criada a tabela **IngredienteReceita** (idReceita, idIngrediente, quantidadeIngrediente). A segunda forma normal foi alcançada quando foram eliminadas as dependências funcionais parciais, ou seja, forma eliminados os atributos que não dependiam inteiramente da chave primária, foi removido o atributo categoria, que originou a tabela **Categoria** (idCategoria, desigCategoria), que faz relação de 1 para n com a tabela **Receita.** Feito isto, o modelo de dados estava na **3FN**, pois estava na **2FN** e não havia nenhuma dependência transitiva entre os atributos e todos os atributos de cada entidade dependem só da chave e unicamente da chave.

Depois de se ter feito o processo de normalização, fez-se um redesenho do diagrama entidade-associação e do modelo relacional de dados, como é apresentado abaixo.

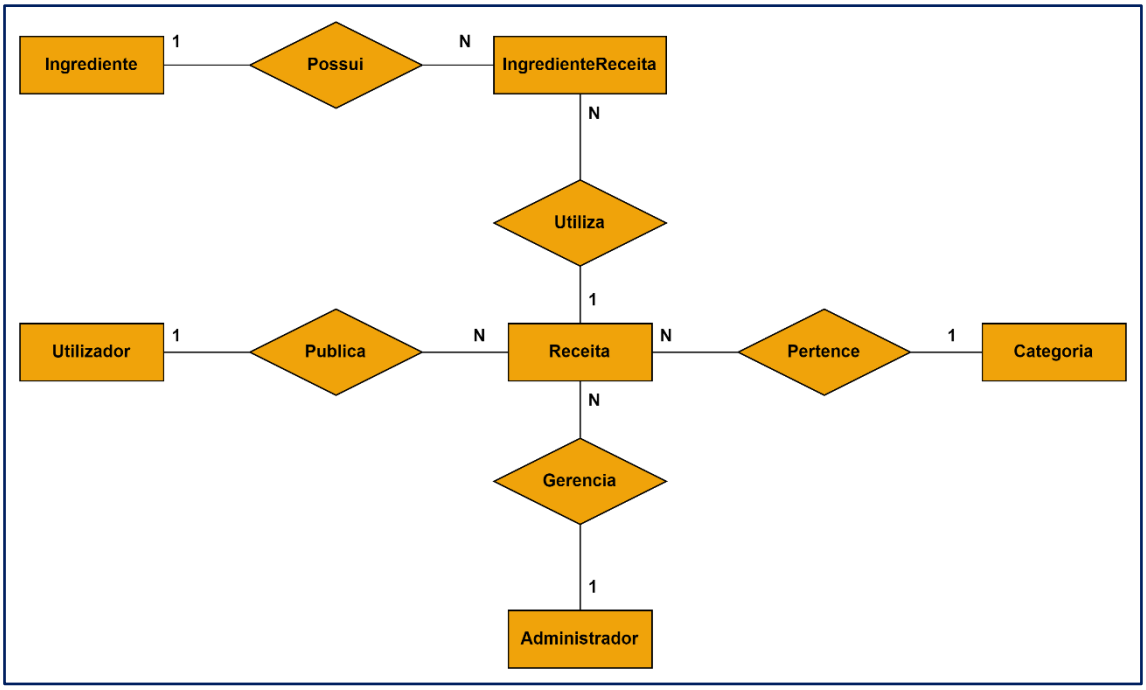

Figura 9. Modelo E-R normalizado

2.5.4. Modelo Relacional de dados

- **Utilizador** (idUtilizador, nomeUtilizador, emailUtilizador, idade);
- **Administrador** (idAdmin, nomeAdmin, emailAdmin);
- **Receita** (idReceita, nomeReceita, rendimento, tempoPreparo, modoPreparo, estadoReceita, publicidadeReceita, idUtilizador, idAdmin, idCategoria);
- **IngredienteReceita** (idReceita, idIngrediente, quantidadeIngrediente);
- **Ingrediente** (idIngrediente, nomeIngrediente, unidadeMedida);
- **Categoria** (idCategoria, desigCategoria).

2.5.5. Modelo Físico

Nesta subseção será apresentado o modelo físico desenvolvido com base no modelo conceptual desenvolvido. Para criar o modelo físico recorreu-se ao SQL Server, onde foi criada a base de dados e extraiu-se de lá o modelo físico. O script para a criação das tabelas da base de dados está em anexo na pasta scripts. A figura abaixo apresenta o modelo físico da base de dados.

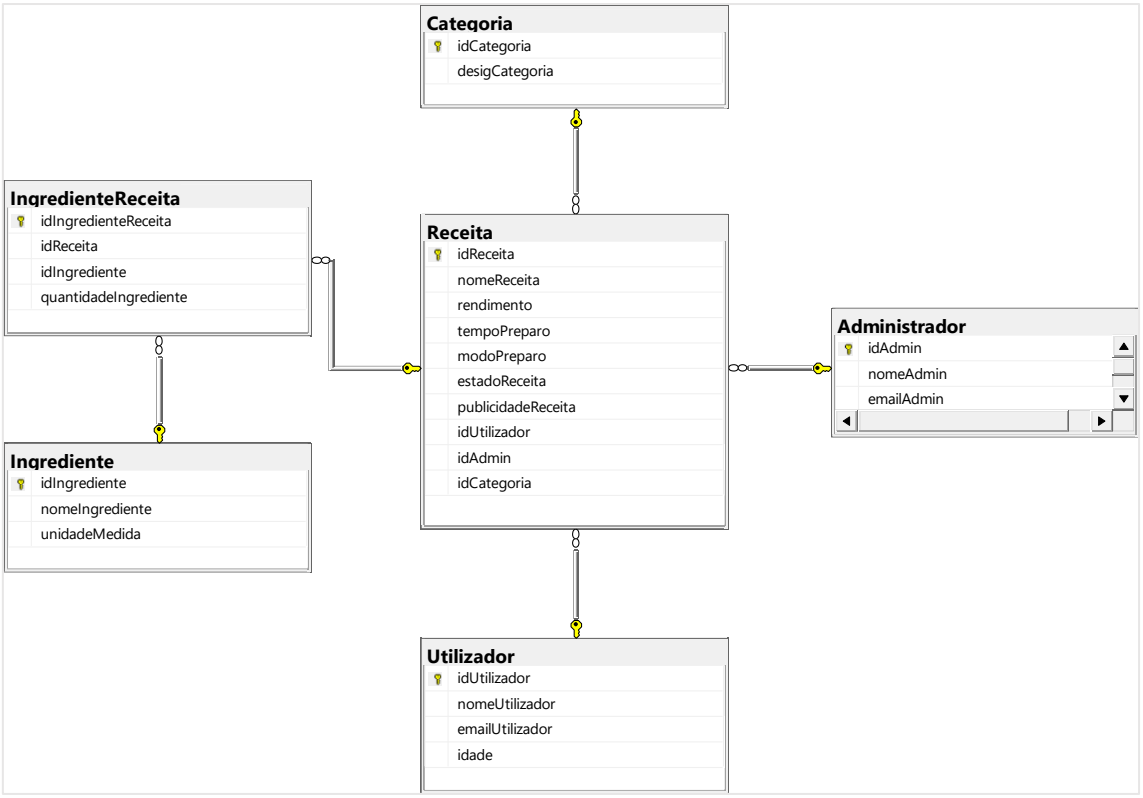

Figura 10. Modelo Físico

4. Implementação

Nesta seção são descritas as principais decisões e ações desenvolvidas na parte da implementação técnica deste projeto. Inicialmente, será feita a definição da arquitetura do sistema na integração da aplicação com a API. Posteriormente, serão apresentados aspetos relevantes na codificação do sistema, que correspondem às decisões de implementação.

4.1. Definição da arquitetura do sistema na integração da aplicação com a API

O diagrama abaixo representa os principais blocos (FrontEnd, base de dados, App MVC) que constituem a arquitetura global do sistema implementado.

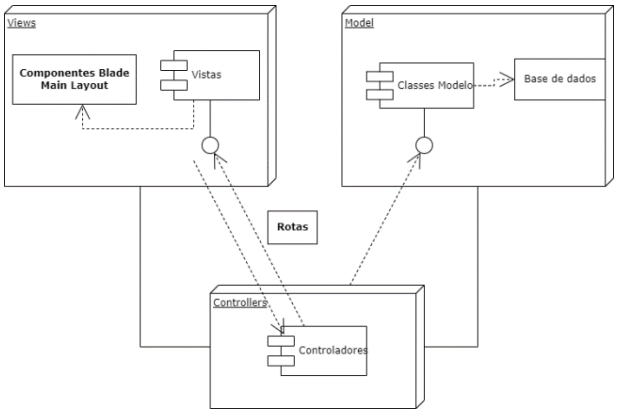

Figura 11: Principais componentes do sistema

4.2. Especificação da interface da API

A documentação da API REST desenvolvida pode ser consultada **nesta hiperligação e** nos anexos, na pasta **API-Doc**, nela foram descritas as operações utilizadores que o utilizador pode realizar utilizando a API e, constitui a documentação necessária para um analista ou programador perceber quais são os seus *endpoints* (método http, URL, dados a enviar no pedido e dados da resposta). A figura abaixo apresenta a tabela que contém a documentação da API.

Figura 12: Documentação da API

4.3. Decisões de implementação

Nas subsecções seguintes serão explicadas as principais decisões tomadas antes da codificação programática do sistema.

4.3.1. Tecnologias usadas

O processo de desenvolvimento deste trabalho, consistiu essencialmente na exploração de tecnologias para desenvolvimento do lado do servidor, nomeadamente, a plataforma de desenvolvimento PHP – **Laravel** [5], onde foram explorados, sob o ponto de vista técnico, os conceitos de **Controladores**, **Vistas**, **Modelos**, **Autenticação**, **Autorização**, desenvolvimento de **API**, entre tantos outros. Foram também utilizadas as seguintes tecnologias:

- **Alphine.js** [6] – o Alpine é uma ferramenta robusta e mínima para compor o comportamento diretamente em *tags* HTML. É uma biblioteca de JavaScript, similar ao *jQuery* para a web moderna.
- **Bootstrap** [7] – o Bootstrap é utilizado para projetar e personalizar sites responsivos voltados para dispositivos móveis. É um kit de ferramentas de código aberto de front-end dos mais populares do mundo, possuindo extensos componentes pré-construídos e poderosos plugins JavaScript. Neste projeto, utilizou-se mais a biblioteca do CSS, para arrumar os elementos na tela e deixar o sistema responsivo.
- **GitHub** [8] – o GitHub é um sistema de controle de versão, que ajuda a acompanhar as mudanças feitas no código base. [9] Além disso, o GitHub regista quem efetuou a mudança e permite a restauração do código removido ou modificado, o que é bastante útil no desenvolvimento de projetos desta dimensão e importância, uma vez que é possível fazer um "*rollback*" para um ponto em que o programa estava funcional, se houver algum erro, que seja difícil de concertar. Esta ferramenta foi utilizada para armazenar o código e para o controlo de versões, **neste repositório**.
- **TablePlus** [10] **–** o TablePlus é uma aplicação que ajuda a gerenciar um sistema de base de dados relacional, como o *MySQL*, *PostgreSQL*, entre outros. Tem uma interface amigável, e apesar de não ser um software com licença paga, possui versões gratuitas, que permitem realizar a maior parte das operações desejadas, mesmo que seja com algumas limitações. Neste trabalho, o TablePlus foi utilizado para dar uma interface aos sistemas de bases de dados que foi utilizado, o *MySQL*.

Em termos de decisões tomadas, grande parte delas forma baseadas no tutorial de Laravel, oferecido gratuitamente pelo Laracasts [11]. É um tutorial com 70 vídeos, duração total de cerca de dez horas e cobre os principais temas do processo de construção de um sítio web. No entanto, houve certa autonomia, no desenvolvimento deste trabalho, principalmente por ter uma temática diferente da que é apresentada no tutorial.

A abordagem de desenvolvimento tomada foi a do ***codefirst,*** isto é, primeiro foi feita a codificação base do sistema, e só depois seguiu a criação da base de dados, pelo facto de parecer mais conveniente que houvesse uma interface para ver como os dados seriam apresentados, tendo se recorrido a dados ***hard-coded***, e só depois foi criada uma estrutura para alocar os dados de forma dinâmica. No caso das escolhas das rotas, foi feita uma tentativa de seguir os padrões de outros sites similares como é o caso do Tudogostoso [12] que serviu de inspiração para grande parte do trabalho desenvolvido. Por outro lado, procurei escrever rotas simples, que fossem o mais informativas possível, de modo que dessem a entender, de certa forma, os efeitos das ações do utilizador, sem, no entanto, deixar de ser seguro.

No que se refere aos nomes dos controladores, estão diretamente associados à classe modelo sobre a qual vão operar, por exemplo, o controlador ***RecipesController.php*** opera sobre a classe Modelo das receitas e lida com as 7 operações CRUD para esta classe. O controlador ***SessionsController.php*** está relacionado com a sessão dos utilizadores, que pode ser criada ou eliminada, e assim por diante. No que toca à visibilidade dos métodos, já que não houve necessidade de criar métodos que fizessem um processamento interno dentro da classe, ou seja, os métodos foram criados na sua maioria como sendo públicos, para que pudessem ser chamados por outras classes, nos casos em que os métodos diziam respeito ao processamento interno da classe, estes foram criados como ***protected*** para que pudessem ser acedido só dentro da classe e por objetos que herdassem daquela classe, um exemplo disso, é o método ***validateRecipe(Recipe $recipe),*** que foi criado como ***protected,*** pelo facto de ser utilizado para validar as receitas naquela classe o que faz com que este seja de processamento interno da classe, e assim sendo, as classes externas não precisam saber como é feita a validação das receitas. Os nomes dos métodos foram criados com base nas operações CRUD, a listar***: index(), create(), store(), show(), edit(), update() e destroy().***

No que se refere ao modelo, a validação dos dados fornecidos através de formulários foi feita na classe do respetivo controlador, utilizando a variável ***request()*** que possui todos esses dados e sobre este *array* de variáveis, podem ser aplicados parâmetros de validação, como por exemplo, não deixar que um campo seja deixado em branco, não deixar que haja repetição de atributos, nos casos em que os atributos tinham de ser únicos.

Para a criação das vistas foi utilizado o "motor" de vistas ***blade***, em que foram criadas vistas para cada uma das ações dos controladores. Assim, para o método ***create*** do controlador, foi criada uma vista ***create.blade.php*** correspondente. No caso em que houvesse muita

repetição de código para gerar um elemento da vista, como por exemplo um campo de input de um formulário, esses elementos foram extraídos para fazer componentes, e assim puderem ser reutilizados. Para passar os dados para as vistas forma utilizadas ***@props()*** que funcionam de forma similar às variáveis, mas não são variáveis e sim uma diretiva dos componentes blade, que permitem guardar valores. Nalguns cenários, foi necessário utilizar JavaScript, seja através de bibliotecas, como as que foram mencionadas na subsecção anterior, ou codificando funções no ficheiro ***app.js.***

A última parte do projeto que considero relevante descrever, está relacionada aos acessos de utilizadores a determinadas funcionalidades. Uma vez que o sistema possui dois tipos de utilizadores, um administrador e outro utilizador comum, havia necessidade de limitar algumas funcionalidades que eram destinadas ao administrador, para que só este tivesse acesso a elas, ficando vedadas para os outros utilizadores que não são administradores. Havia várias soluções para alcançar esse objetivo, tendo sido pensado primeiramente em criar roles na base de dados, e com base nestes roles os utilizadores teriam acesso ou não a determinadas funcionalidades. No entanto, preferi adotar outra estratégia, mais focada na programação do que na base de dados, que foi a utilização do ***Middleware***. Essa escolha foi eficiente e eficaz, pois consegui alcançar os objetivos que esperava sem muito esforço. Foram criado dois *Middleware*, um ***MustBeAdmin*** para nomear o administrador e um ***MustBeUser*** para nomear os outros utilizadores do sistema. Feito isso, foi só necessário aplicar o *Middleware* específico às rotas específicas, para que cada tipo de utilizador pudesse aceder somente ao conteúdo a que era autorizado aceder.

4.4. Codificação

Nesta secção será feita uma explicação objetiva das decisões de código mais importantes e críticas de cada caso de uso, exemplificando com pequenos trechos de código.

4.4.1. Autenticação e autorização

A autenticação e autorização foram dois conceitos importantes no desenvolvimento deste trabalho e na codificação do sistema, pois tiveram de ser implementadas métricas de autorização nas rotas uma vez que o utilizador consegue navegar na aplicação utilizando *urls*. Foram utilizados, para o efeito, o *Middleware*. A figura abaixo mostra a codificação de uma rota com *Middleware*. Com o *Middleware* é fácil redirecionar uma rota para a outra se a pessoa que tentar aceder a esta não tiver autorização.

```php
Route::post( uri: 'login', [AuthController::class, 'signin']);
Route::post( uri: 'register', [AuthController::class, 'signup']);

Route::middleware( middleware: 'auth:sanctum')->group( function () {
    Route::resource( name: 'recipes', controller: RecipesController::class);
});
```

```php
public function handle(Request $request, Closure $next)
{
    if (auth()->guest()) {
        return redirect( to: '/register');
    }
    return $next($request);
}
```

Figura 13: Codificação da autenticação e autorização

4.4.2. Controladores

A codificação dos controladores serviu não só para estabelecer a comunicação entre as classes do modelo e as vistas, mas também para fazer a validação dos campos do formulário do envio de receitas, nos casos em que se aplica.

```php
public function store()
{

    $attributes = $this->validateRecipe();

    $attributes['name'] = request()->get('name');
    $attributes['user_id'] = auth()->id();
    $attributes['publish'] = isset($_POST['publish']);
    $attributes['picture'] = request()->file( key: 'picture')->store( path: 'public/thumbnails');


    $recipe = Recipe::create($attributes);
```

Figura 14: Codificação dos controladores

4.4.3. Modelos

Nas classes de modelo, foi necessário restringir o acesso a determinados campos da base de dados, para evitar que fossem inseridos dados indesejados, evitando assim, ataques maliciosos. Foi utilizado o array *fillable*, que permite declarar que propriedades podem ser preenchidas utilizando fontes de dados externas.

```
protected $fillable = [
    'slug', 'picture', "name", "servings",
    "time", "instructions", "publish", "approval",
    "category_id", 'user_id'];
```

Figura 15: Codificação dos Modelos

4.4.4. Preenchimento da base de dados

Foram utilizados *database seeders* para preencher a base de dados com alguns valores.

```
$user = User::factory()->create();


Category::create([
    'name' => 'Bolos e Tortas Doces',
    'slug' => 'bolos-tortas-doces'
]);

Category::create([
    'name' => 'Carnes',
    'slug' => 'carnes'
]);
```

Figura 16: Database Seeders

Conclusões e Perspetivas de Trabalho Futuro

Como jeito de conclusão, pode-se dizer que o trabalho cumpre com os objetivos esperados, tendo se desenvolvido mais do que era pedido, por exemplo, com a especificação dos requisitos não funcionais. O desenvolvimento do trabalho foi bastante interessante pois permitiu rever e consolidar os conhecimentos adquiridos em contextos de outras unidades curriculares, tornando o trabalho bastante rico em termos de conteúdo e desenvolvendo o sentido de estudo autónomo.

Na fase de análise do sistema foram identificados os atores do sistema e as funções que estes vão desempenhar no sistema e os requisitos funcionais do sistema, que servirão de suporte às tarefas que os atores terão o desejo de realizar. Na fase de desenho, foram desenhadas as interfaces com o utilizador que são a forma como os utilizadores vão interagir com o sistema, optou-se por se utilizar o desenho minimalista, sem muitos elementos na tela, além daqueles que são estritamente necessários para o desenvolvimento das tarefas. A base de dados desenhada permitirá armazenar as receitas de forma que estas possam ser consultadas no futuro.

Na fase de implementação foram feitas algumas alterações em relação à fase de desenho, nomeadamente, na forma como o utilizador realiza as tarefas no sistema e também houve alteração do funcionamento dos casos de uso. Na fase de análise e desenho, era suposto que todos os utilizadores adicionassem receitas e o administrador tivesse uma função mais passiva que era só de aprovar ou não as receitas enviadas pelo utilizador, no entanto, essa abordagem não pareceu muito viável uma vez que não tenho conhecimento das métricas de validação de uma receita, em termos programáticos, seria complicado perceber quando é que uma receita seria aprovada ou não e assim, o administrador teria de validar as receitas com base nos seus conhecimentos em culinária ou na sua intuição, colocando, deste modo, o seu cunho pessoal para validar uma receita ou não, e se não validasse e tivesse de editar a receita, por exemplo, acabaria por reformatar a receita como bem entende, o que em termos de conteúdo de website não seria muito interessante num contexto de vida real. Por isso, pareceu conveniente que fosse o administrador a publicar as receitas e os restantes utilizadores só vão consultar tais receitas.

Futuramente, poderiam existir funcionalidades que permitam ao utilizador comentar ou classificar as receitas consultadas, adicionar aos favoritos, entre outras. Além disso, espero melhorar alguns aspetos no trabalho que reconheço eu podiam estar melhores, de modo a tornar o website desenvolvido o mais aproximado possível da realidade.

Em termos de resultados, foi desenvolvido um protótipo funcional de uma aplicação web, que permite a autenticação dos seus utilizadores e com base nessa autenticação, é feita a autorização para limitar o acesso a determinadas funcionalidades a pessoas autorizadas. Uma limitação que se fez sentir em peso foi o fato de os meus conhecimentos de CSS serem rudimentares, o que dificultou o desenvolvimento de um protótipo mais visualmente atrativo. Em termos de aprendizagens, posso dizer que aprendi a 100%, uma vez que apesar de já ter tipo contacto com a matéria no contexto de sala de aulas, estar a desenvolver de forma autónoma enriqueceu mais os meus conhecimentos.

Referências

[1] ABNT, Norma Brasileira ABNT NBR ISO/IEC 27002, Rio de Janeiro, 2005.

[2] K. Moran, "Flat Design: Its Origins, Its Problems, and Why Flat 2.0 Is Better for Users," 27 Setembro 2015. [Online]. Available: https://www.nngroup.com/articles/flat-design/. [Acesso em 30 Março 2021].

[3] A. Lucas, P. Cristiane, D. P. d. Silva, J. Camacho e L. V. Henriques, "Modelo Entidade-Associação," 2008. [Online]. Available: https://www.iseg.ulisboa.pt/aquila/getFile.do?fileId=21402&method=getFile. [Acesso em 17 Novembro 2021].

[4] A. Lucas, P. Cristiane, D. P. d. Silva, J. Camacho e L. V. Henriques, "Normalização de tabelas," 2008. [Online]. Available: https://www.iseg.ulisboa.pt/aquila/getFile.do?fileId=19012&method=getFile. [Acesso em 17 Novembro 2021].

[5] "Laravel," [Online]. Available: https://laravel.com/. [Acesso em 29 Janeiro 2022].

[6] [Online]. Available: https://alpinejs.dev/. [Acesso em 29 Janeiro 2022].

[7] [Online]. Available: https://getbootstrap.com/. [Acesso em 29 Janeiro 2022].

[8] "GitHub," [Online]. Available: https://github.com/. [Acesso em 29 Janeiro 2022].

[9] L. Andrei, "O Que é GitHub e Como Usá-lo," Hostinger Tutoriais, 15 Outubro 2021. [Online]. Available: https://www.hostinger.com.br/tutoriais/o-que-github. [Acesso em 29 Janeiro 2022].

[10] "TablePlus," [Online]. Available: https://tableplus.com/. [Acesso em 29 Janeiro 2022].

[11] "Laravel 8 From Scratch," [Online]. Available: https://laracasts.com/series/laravel-8-from-scratch. [Acesso em 29 Janeiro 2022].

[12] [Online]. Available: https://www.tudogostoso.com.br/. [Acesso em 29 janeiro 2022].

[13] "Função e tarefas do administrator de site," Tableau, [Online]. Available: https://help.tableau.com/current/online/pt-br/to_site_startup.htm. [Acesso em 29 10 2021].