Universidade do Minho
Escola de Engenharia
Licenciatura em Engenharia informática
Mestrado Integrado em Engenharia Informática

Unidade Curricular de Laboratórios de Informática IV

Ano Lectivo de 2021/2022

Friendcesinha

Guia para Restaurantes que servem francesinhas

Ana Catarina Oliveira Gonçalves A93259

Bruno Filipe Miranda Pereira A93298

Francisco José Martinho Toldy A93226

João Pedro Fontes Delgado A93240

Janeiro, 2021

LI4

Data de Recepção	
Responsável	
Avaliação	
Observações	

Friendcesinha

Guia para Restaurantes que servem francesinhas

Ana Catarina Oliveira Gonçalves A93259

Bruno Filipe Miranda Pereira A93298

Francisco José Martinho Toldy A93226

João Pedro Fontes Delgado A93240

Janeiro, 2021

Resumo

Este documento relata a fase de implementação de um guia de restaurantes cujas especialidades são as francesinhas, Friendcesinha. Em relação à contextualização que nos foi fornecida, o grupo considerou que esta era completa e efetivamente fundamentada. Por outro lado, a especificação não verificou estar ao mesmo nível de completude da contextualização. Isto deve-se a vários fatores, tal como a ausência de vários Use Cases, Mockups e elementos essenciais na base de dados fornecida. Em adição, existem vários conflitos nas informações fornecidas, principalmente, as relações entre requisitos, base de dados, use cases e mockups, como mencionado na apreciação anteriormente formulada.

Assim sendo, vamos iniciar a apresentação geral do trabalho realizado, incluindo o enquadramento geral, a caracterização do trabalho desenvolvido, os objetivos, os recursos utilizados e finalmente o plano de desenvolvimento.

Seguidamente, apresentaremos a fase de desenvolvimento da aplicação, constituída pela apresentação da estratégia escolhida para a criar, os requisitos que foram implementados, em conjunto com a justificação daqueles que não foram implementados. Em adição com a arquitetura final da aplicação e a caracterização dos seus componentes.

Concluímos então este documento com uma abordagem crítica do trabalho realizado como também algumas sugestões para um trabalho futuro.

Em anexo se seguiram as interfaces finais da aplicação elaborada em conjunto com o diagrama da base de dados.

Índice

Índice de Figuras

1. Introdução

1.1. Enquadramento Geral

Com o desenvolvimento desta aplicação, temos como propósito um software capaz de servir de guia para restaurantes, mais especificamente, de francesinhas. Um guia tem então de ser capaz de informar o cliente das várias opções que pode optar por, e ainda mais importantemente, apresentar informações que distinguem as várias opções. Assim sendo, a aplicação desenvolvida preenche estes requisitos e ainda fornece ao utilizador uma forma de o ajudar a deslocar-se ao local escolhido.

Adicionalmente, têm a versão direcionada aos proprietarios, onde estes podem adicionar os seus negócios, publicitando-os. Pode editá-los e eliminá-los ao seu gosto e também receber reservas e pedidos por parte dos utilizadores.

1.2. Caracterização do Trabalho a Desenvolver e Seus Objetivos

O software a desenvolver deveria ser capaz de interagir com utilizadores de 3 tipos, sendo estes Proprietários, Administrador e Clientes. Cada um dos tipos de utilizadores teria acesso a um conjunto específico de funcionalidades, interagindo com uma UI intuitiva.

A nível do Proprietário, a aplicação deveria fornecer as funcionalidades necessárias para suporte da gerência dos vários restaurantes do mesmo. Assim, o proprietário deveria, nomeadamente, poder alterar certos dados dos seus estabelecimentos, por exemplo, alterar o Menu, as horas de expediente, consultar e remover reservas e adicionar códigos promocionais.

O Cliente deve ser capaz de escolher entre uma seleção de restaurantes, filtrada mediante preferências do mesmo. Escolhendo um restaurante, o Cliente poderá consultar o seu Menu, as avaliações feitas ao mesmo, realizar a sua própria avaliação e ainda efectuar uma reserva. Além disso, será capaz de alterar os dados da sua conta e as suas preferências.

O software especificado tinha como objetivos providenciar ao utilizador uma experiência gastronómica de qualidade e um aumento significativo de clientes nos estabelecimentos inscritos.

1.3. Identificação, Caracterização e justificação dos Recursos Utilizados

No decorrer do desenvolvimento recorreu-se a uma variedade de recursos sobre os quais iremos elaborar nesta secção.

Foi necessário fazer uma pesquisa extensiva sobre estruturação de aplicações web, passando por aprendizagem relativa às funções da front-end e da back-end, escolha de linguagens, escolha das frameworks a utilizar, bem como o software a utilizar no desenvolvimento. Essa aprendizagem e pesquisa foi, também, suportada por reuniões e contacto com os docentes.

A nível de front-end, foi necessário aprender a programar em HTML e CSS, sendo essa aprendizagem auxiliada pelos cursos das plataformas ***codecademy*** e ***freeCodeCamp*** entre vários outros vídeos informativos. Além disso foi adquirido conhecimento da linguagem javascript, que mais tarde serviria de base para aprendizagem e utilização da framework React.js . Essa framework foi recomendada como solução mais indicada para client-side-rendering, alternativa escolhida em oposição a server-side-rendering (que significaria uma estrutura mais ambígua de responsabilidades do sistema).

Figura 1 : Recurso HTML

Figura 2: Recurso CSS

Figura 3: Recurso JavaScript

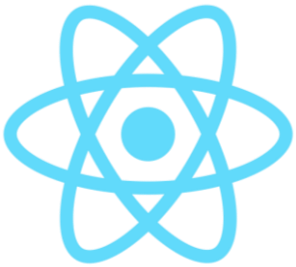

Figura 4: Recurso React.js

A programação nas linguagens do front-end foi realizada no Visual Studio Code, uma vez que além de ser um editor de texto com o qual os elementos estavam familiarizados, tinha sido referido em vários vídeos informativos como sendo uma boa opção, complementada com alguns plug-ins de melhoria de quality of life, nomeadamente o Emmet.

Figura 5: Recurso VSCode

A nível do backend a linguagem de programação escolhida foi Java, recorrendo-se à framework Spring. A aprendizagem necessária para a utilização do Java Spring foi feita recorrendo, tal como na front-end, a vídeos informativos e artigos. A familiaridade com a própria linguagem Java, permitiu que a aprendizagem se focasse maioritariamente em conceitos da framework e não propriamente na sintaxe, uma vez que a esse nível, esta é bastante simples.

Figura 6: Recurso SpringBoot

Figura 7: Recurso Java

A nível da base de dados foi escolhido, como recomendado pelos docentes, o software MySQLWorkbench para projetar o diagrama especificado pelo grupo anterior.

Figura 8: Recurso MySQL WorkBench

Em relação à comunicação entre a equipa de trabalho, recorreu-se à plataforma Discord para suportar a elevada comunicação entre cada membro das equipas front-end/back-end, bem como as reuniões virtuais do grupo todo. Por fim, recorreu-se ao GitHub para controlo de versões do projeto.

Figura 9: Recurso Discord

Figura 10: Recurso Github

1.4. Plano de Desenvolvimento Adotado

Quanto ao planeamento do desenvolvimento do projeto, inicialmente foi decidido a estrutura das equipas, separando o desenvolvimento da frontend e da backend, fazendo cada elemento do grupo o estudo de ferramentas e tecnologias que pudessem auxiliar ao desenvolvimento das mesmas.

A partir da segunda semana, já com a ideia das ferramentas a utilizar, cada elemento começou a aprender as linguagens necessárias à implementação. Uma vez aprendidos os conceitos essenciais que o grupo considerou serem suficientes, foi iniciada a fase de implementação, como se pode verificar no seguinte diagrama:

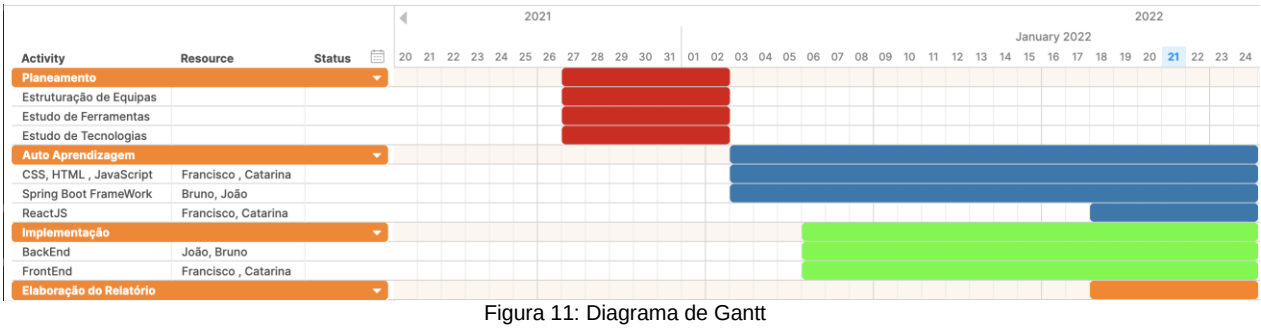

Figura 11: Diagrama de Gantt

2. Desenvolvimento da Aplicação

2.1. Apresentação da Estratégia e Método de Desenvolvimento Adotados

O método de desenvolvimento passou pela divisão do grupo em 2 pares, um dedicado à front-end e outro dedicado à back-end. O objetivo desta divisão seria realizar o máximo do trabalho em paralelo, devido ao tempo que teria que ser dedicado não só à implementação da especificação como também à aprendizagem necessária para a mesma.

O desenvolvimento de cada secção foi realizado em contacto constante com os membros do outro par, de forma a perceber como o trabalho de um par complementaria o trabalho do outro. Começou por ocorrer um período em que cada par se focou apenas na recolha de conhecimento. Seguidamente, por restrições de tempo, foi necessário continuar essa aprendizagem em paralelo ao desenvolvimento da aplicação. Aproximando-se o prazo de entrega, os dois pares começaram a trabalhar de forma mais interligada, acabando por redigir o relatório em conjunto.

Em termos da implementação de cada requisito, foi adotada a estratégia de ordená-los por ordem de prioridade baseada na qualidade de suporte fornecido a cada um deles na especificação. Além disso, foram necessárias algumas mudanças à especificação de forma a acomodar certos requisitos. A estratégia adotada nessas mudanças foi reduzi-las ao mínimo necessário para tornar um dado requisito implementável. Algumas das mudanças já foram indicadas no relatório de apreciação, no entanto algumas foram efectuadas e justificadas apenas após o início do trabalho de implementação.

2.2. Identificação e Caracterização da Aplicação e dos Seus Serviços

O presente capítulo irá abordar os vários requisitos especificados, detalhando o seu grau de implementação (variando entre totalmente implementado, implementado com alterações à especificação e não implementado) e assinalando todas as alterações efectuadas.

Iremos percorrer cada um dos requisitos na mesma ordem com que estes foram apresentados na especificação:

1. "Para criar conta na aplicação, o utilizador tem de inserir um username e um e-mail válidos, e uma password"

Classificaremos este requisito como totalmente implementado, uma vez que foi implementado como previsto no Use Case 1, bem como na Figura 8 do relatório, sendo necessário apenas assinalar a inclusão do NIF e número de telemóvel no processo de registo uma vez que este estava indicado como elemento da tabela Cliente da Base de Dados, apesar de não ser referido nos restantes elementos da especificação.

É importante referir ainda que foi efetuada uma distinção entre registo de Cliente e Proprietário.

2. "Um utilizador pode ser um cliente, um proprietário ou o administrador da plataforma"

Requisito parcialmente implementado, uma vez que, devido a restrições de tempo, não foi possível implementar o Administrador, sendo o foco colocado em implementar todos os requisitos relativos ao Proprietário e Cliente. Iremos abordar, quando pertinente, a forma como a ausência de Administrador alterou a resposta a cada requisito funcional.

3. “Para se registar como um proprietário, um cliente deve fornecer um comprovativo de propriedade do negócio, assim como as informações referentes ao mesmo”

Tal como referido previamente, foi feita uma distinção entre registo de Proprietário e Cliente a nível das páginas geradas para esse fim. Além disso, uma vez tomada a decisão de descartar o administrador, foi assim retirada a funcionalidade de fornecimento de comprovativo de propriedade de negócio para aprovação por parte do mesmo. Uma vez que é apresentado na Tabela 17, um Use Case “Registar Restaurante” independente, em termos sequenciais, do registo na plataforma. Foi tomada a decisão de fornecer a possibilidade de adicionar restaurante sem nenhuma dependência ao registo de conta, podendo-o fazer a qualquer momento que desejar após o registo de conta . Uma vez que não foi fornecida uma Mockup para suporte da operação de registo de proprietário foi decidido adaptar a Mockup da Figura 8 para o proprietário.

4. “O administrador da plataforma adiciona ou remove os restaurantes, em função dos pedidos recebidos”

Não implementado um administrador, a funcionalidade de adicionar e remover restaurantes foi transferida diretamente para o Proprietário, sendo este então capaz de adicionar e remover os restaurantes por ele registados. As opções de adicionar e remover restaurantes são acessíveis partindo de um menu principal do proprietário. Os parâmetros para adicionar um restaurante foram definidos com base nos elementos da tabela Restaurante da base de dados especificada, bem como a informação detalhada na Mockup da figura 11.

Em relação à remoção de restaurantes, devido à inexistência de mockups e use cases relativamente a este requisito foi nos dada a liberdade de criar uma opção no menu do proprietário de eliminar um dos seus restaurantes e uma view adicional onde é selecionado o restaurante que deseja eliminar.

5. “O administrador é responsável por validar os comprovativos de propriedade submetidos pelos proprietários”

Requisito não implementado, como consequência da não implementação do admistrador na aplicação

6. "A autenticação na aplicação deve ser feita com recurso ao username e a respetiva palavra-passe"

Requisito totalmente implementado. O utilizador da aplicação começa por selecionar se é um Cliente ou Proprietário e redirecionado para versões semelhantes de um ecrã que permite fazer LogIn ou Registo na aplicação. O botão de Registo irá redirecionar, dependendo da escolha no primeiro ecrã, para o ecrã de Registo de Cliente ou de Proprietário.

7. "Um utilizador pode editar username e palavra-passe do seu perfil, não estando, contudo, autorizado a alterar o seu e-mail. A aplicação permitirá definir um raio correspondente à distância máxima de procura de restaurantes e um número mínimo de estrelas para os restaurantes a procurar"

Requisito implementado na totalidade em concordância com a Mockup Figura 12 da especificação. Não foi necessário efetuar alterações ao especificado. Além disso, a opção de alterar a pontuação mínima para a filtragem de restaurantes foi implementada em conjunto com as restantes possibilidades de edição de perfil. A decisão de adicionar a opção da pontuação mínima nesse menu partiu do facto da mesma não estar apresentada nas Mockups, tendo o grupo decidido que faria sentido implementá-la em conjunto com a opção da distância máxima.

8. "A aplicação deve permitir que as preferências do cliente possam ser alteradas a qualquer momento"

Requisito implementado na totalidade, as alterações às preferências do cliente são acessíveis por via do menu principal do cliente, selecionando a opção "Perfil" no canto inferior direito.

9. "Os resultados apresentados, sob a forma de uma lista, devem estar ordenados pelo seu número de estrelas, respeitando a configuração de preferências previamente definida"

Sendo este um dos requisitos que se encontra em conflito (particularmente em relação à apresentação dos restaurantes em lista , como especificado nos requisitos, ou em forma de um mapa, como especificado nas mockups), optamos então por escolher uma apresentação em forma de mapa, devido à sua facilidade de visualização da parte do utilizador.

10. "Após a seleção do restaurante, o cliente pode consultar a informação relevante acerca do mesmo, como o menu das francesinhas e ingredientes que as constituem"

A implementação deste requisito passou por apresentar a informação do restaurante como indicado na Mockup da especificação correspondente. O Menu das francesinhas, tal como é indicado na Mockup, é acedido através do botão Menu. Uma vez que não foi apresentada Mockup para essa opção, foi tomada a decisão de manter grande parte do ecrã anterior, substituindo o painel direito por uma lista dos pratos. A nível da apresentação de ingredientes do prato, foi decidido não implementar essa vertente dos pratos, uma vez que não era suportada pela base de dados fornecida, nem pelos MockUps apresentados.

Por último, foi tomada a decisão de não incluir um ícone e/ou imagem relativa ao restaurante, uma vez que, apesar de apresentado na Mockup, a inclusão de imagens não era suportada nem pela base de dados nem use cases (relativos à visualização e registo de restaurante) nem é mencionada em qualquer dos requisitos.

11. "O cliente pode efetuar reservas num restaurante, indicando a data e hora da mesma e número de pessoas, bem como fazer o seu pedido para que este seja antecipadamente preparado"

Este requisito foi implementado da forma como está indicado nos Use Cases pertinentes (Tabela 7 e 15) e na Mockup apresentada na Figura 14. A única alteração efetuada foi a remoção da possibilidade de introduzir opções relativas a Entrada, Bebida, Acompanhamento e Sobremesa, que não estavam mencionadas em qualquer ponto da especificação exceto a Mockup em questão.

12. "A aplicação deverá ser capaz de fornecer direções, com recurso ao Global Positioning System (GPS), até ao restaurante selecionado pelo cliente"

Dado que este requisito não era suportado por Mockups ou Use Cases e não foi fornecido explicação de como seria implementado devidamente, foi decidido pelo grupo que a forma como esta funcionalidade se implementaria seria através de um redirecionamento para o Google Maps. Esse redirecionamento é conseguido ao carregar no botão "Direções" na página de um restaurante e a página redirecionada encontra-se preenchida com os dados do local e da localização atual do utilizador, de forma a apenas ser necessário iniciar o percurso.

13. "Após o registo na plataforma, o cliente tem direito a um código Quick Response (QR) de boas-vindas, com desconto 10% que pode usufruir num dos restaurantes aderentes"

Requisito totalmente implementado. Importante referir, no entanto, que apesar de existir suporte deste requisito nos Use Cases (Tabela 14, nomeadamente) não existe suporte dos códigos promocionais na Base de Dados apresentada, e por isso foi acrescentado o necessário na base de dados de modo a implementarmos este requisito.

14. "Os proprietários podem gerar códigos QR promocionais para o seu estabelecimento"

Requisito implementado na totalidade, sendo a possibilidade de gerar um código promocional é apresentada na página de edição do Código Promocional. O proprietário tem a possibilidade de introduzir a descrição do código promocional que pretende gerar, e essa descrição é enviada para a backend, onde é gerado um código QR que é armazenado na base de dados. Uma vez que não havia suporte, na base de dados especificada, para qualquer informação referente a códigos promocionais, foi necessário fazer as alterações necessárias à mesma.

15. "Os clientes podem avaliar os restaurantes através de um sistema de estrelas (entre 0 e 5) e efetuando comentários"

Funcionalidade suportada na sua totalidade em concordância com a MockUp apresentada na Figura 13 e com o Use Case referido na Tabela 8 da especificação.

16. "O cliente pode consultar as avaliações feitas por outros clientes"

Esta funcionalidade não estava suportada pelas Mockups apresentadas na especificação, sendo criado então uma interface simples, apresentada na página com as informações do restaurante, como uma lista de conteúdos Avaliação, sendo apresentados os usernames, as pontuações e comentários (caso existam).

17. “O proprietário pode alterar as informações referente ao seu restaurante”

Funcionalidade implementada como especificado, acrescentando a opção de adicionar códigos promocionais (por via de apresentação de uma descrição para ficar associado ao mesmo) e adicionar pratos (sendo necessário fornecer uma descrição/nome e o preço). O acrescento da opção de adicionar pratos neste ecrã partiu do facto de não ser apresentada de forma clara uma forma de adicionar um menu e/ou pratos em qualquer Use Case ou requisito ou Mockup, tendo o grupo decidido que este ecrã seria o local pertinente para esta função.

Concluindo a confirmação de implementação dos requistos exigidos, apresentamos a seguir alguns requisitos que não foram apresentados explicitamente no relatorio fornecido, mas que foram idealizados por exemplo nas mockups ou nos use cases.

18. Cancelar reserva

A nível de funcionalidades implementadas é ainda necessário referir que foi implementado o Use Case da tabela 16 apesar de este não ter sido precedido por menções nos requisitos, e mockups. Esta funcionalidade está presente na página inicial do proprietário, que quando acedida, apresenta a opção de selecionar o restaurante que deseja eliminar.

19. Apresentação dos códigos promocionais

Apesar de não ser um requisito, foi apresentado nos mockups que um utilizador tem a possibilidade de aceder a todos os códigos promocionais de todos os restaurantes (acedido a partir do menu principal do utilizador) e também a todos os códigos promocionais de um restaurante selecionado (a partir da página do restaurante selecionado)

20. Apresentar as reservas realizadas por um cliente

Apesar de também não ser apresentado nos requisitos, é apresentado numa das mockups a possibilidade de um cliente verificar as suas reservas, foi uma funcionalidade implementada.

Com tudo acima apresentado, apresentaremos a seguir uma funcionalidade extra que consideramos importante implementar:

21. Filtro por classificação e distância

Foi implementada a opção de filtrar os restaurantes pelo parâmetro da distância como também pelo parâmetro das classificações, sendo que esta filtragem será observável quando o utilizador estiver a vizualizar o mapa dos restaurantes perto de si. Estes poderão ser alterados consoante a vontade do cliente, mas terão um valor definido por defeito, aquando do registo do utilizador

2.3. Arquitetura Geral da Aplicação Desenvolvida

A aplicação desenvolvida é dividida em 3 camadas, a Front-End, a Back-End e a base de dados.

A front-end tem como responsabilidades a renderização das diversas interfaces com o utilizador, sendo capaz de apresentar informação fornecida pela back-end bem como recolher os inputs do utilizador.

A back-end tem como responsabilidades efetuar a comunicação com a front-end, recebendo pedidos HTTP e enviando respostas aos mesmos e garantir a comunicação com a base de dados, permitindo armazenar e consultar os mesmos.

A base de dados terá como responsabilidade garantir a persistência dos dados relativos à aplicação, de forma que estes estejam organizados corretamente, tendo em conta as relações entre as várias tabelas, para poderem ser consultados, posteriormente.

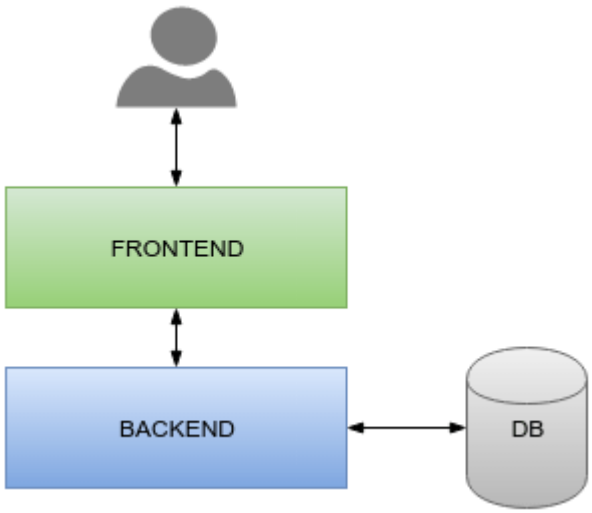

Figura 12: Esquema da Arquitetura

2.4. Definição e Caracterização de Cada um Dos Componentes

2.4.1 Front-End

Como referido anteriormente, foi tomada a decisão de recorrer a Client-Side-Rendering, que significaria que as páginas seriam produzidas na front-end ao contrário de serem produzidas na back-end e enviadas para a front-end, numa perspetiva Server-Side-Sendering. O facto de termos escolhido esta abordagem resulta em algumas vantagens como:

- Renderização da página é mais rápida
- É colocada menos carga no servidor
- A página renderizada é interativa

Devido a essa escolha, foi utilizada a framework React js para criar ficheiros .js correspondentes a todas as páginas que seriam necessárias para o funcionamento da Web App.

Assim, a front-end está dividida em duas componentes: os Ficheiros React .js responsáveis pela apresentação da informação, pedidos de input, e trocas de informação com a back-end necessárias para o funcionamento do UI; e o ficheiro CSS responsável pelo estilo das páginas.

Os ficheiros React.js comunicam com a back-end à base de HTTP requests, recorrendo ao método fetch para indicar especificamente o método da backend que deveria ser executado de forma a fornecer e receber a informação necessária. Além disso, cada ficheiro React tem definido o return da página HTML(especificada dentro do próprio ficheiro .js) que deverá ser executada.

O ficheiro de CSS foi definido com foco na uniformidade do aspecto de cada uma das páginas. Sendo que as páginas para a qual não foram fornecidas mockups (principalmente as do UI do utilizador do tipo proprietário) optou-se então por se reutilizar definições dos ficheiros de CSS elaborados em concordância com as outras mockups especificadas, de forma a assegurar essa uniformidade.

2.4.2 Back-End

A back-end tem, entre outras, a responsabilidade de armazenamento dos dados provenientes da front-end. Recebendo um pedido para um determinado endpoint, a back-end saberá como lidar com o mesmo, de forma a armazenar a informação necessária na base de dados. Com um compromisso entre as duas camadas, para cada funcionalidade necessária, foi convencionado um certo formato de serialização, que permitiria retirar os dados da mensagem HTTP e armazená-los corretamente.
Para além disso, é também a ponte entre a base de dados e a front-end no âmbito da consulta. Com base em informação pedida pela front-end, esta camada responde com informação pertinente que estará armazenada na base de dados, efetuando uma consulta à mesma.
Por fim, esta camada tratará também de todas as operações relativas à lógica de negócios da aplicação.

- **Controllers**

  Os controladores, são responsáveis por receber e tratar os pedidos da front end. Recebendo as mensagens provenientes da front end, extraem o seu conteúdo e redirecionam a informação pertinente para o método adequado da camada de lógica de negócio, onde a operação desejada será efetuada. Existem controladores para os métodos respetivos às operações sobre clientes, proprietários e restaurantes.

- **Model**

  O modelo é o package onde se encontram todas classes relativas às entidades do sistema, que serão mapeadas, posteriormente, para a base de dados, bem como classes auxiliares à lógica de negócios.

- **Respositories**

  Os repositórios são as interfaces que interligam a camada da back-end com a base de dados, não só permitindo persistir todos os dados relativos à lógica de negócios, como também executar queries sobre os mesmos, fornecendo métodos para que isso seja possível.

- **Services**

  A camada de serviços é responsável por executar os métodos relativos à lógica de negócio da aplicação. Para isso, recebe os dados, previamente desencapsulados por um controlador, tratando-os e, por fim, constrói a resposta que será então enviada para a front-end, passando mais uma vez pelo controlador.

Base de Dados

Por fim, a camada de dados apresenta um papel fundamental para o correto funcionamento da aplicação, permitindo a persistência dos dados relativos às principais entidades constituintes do sistema. Com o objetivo de permitir o correto funcionamento de todos os requisitos especificados pelo grupo anterior, foram necessárias alterações à base de dados proposta.

- **Cliente**

Esta tabela armazena toda a informação relevante para os utilizadores.

Figura 13: Tabela do Cliente no Relatório fornecido

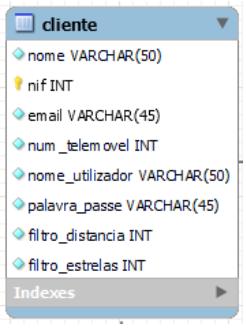

Figura 14: Tabela do Cliente

Em relação à tabela proposta na especificação, estavam em falta dois atributos essenciais à implementação dos requisitos de filtragem de restaurantes a partir do critério da distância e da classificação.

É de notar que esta tabela terá uma relação de 1 para n com as reservas, sendo que um restaurante poderá efetuar n reservas, e também uma relação do mesmo tipo com tabela avaliação, já que o utilizador poderá efetuar várias avaliações.

- **Proprietário**

Esta tabela será onde estarão armazenados os dados do proprietário.

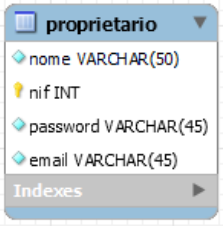

Figura 15: Tabela do Proprietário

Tendo a possibilidade de registar restaurantes, o proprietário terá uma ligação de n para 1 com a tabela Restaurante, uma vez que um proprietário poderá ter vários restaurantes registados.

- **Restaurante**

A tabela Restaurante mapeia toda a informação relevante sobre um restaurante. Para permitir a implementação dos requisitos relativos à obtenção de direções para um dado restaurante e ainda a sua apresentação num mapa foi necessário adicionar à tabela do restaurante a informação relativa às suas coordenadas.

A tabela do restaurante tem uma relação de n para 1 com a tabela proprietário, de 1 para n com a tabela avaliação, uma relação de 1 para n com a tabela reserva e ainda uma relação de n para n com a tabela prato utilizando a tabela de junção serve. Tem , por fim, uma relação de 1 para n com a tabela codigoqr, uma vez que a um restaurante poderão estar associados vários códigos promocionais.

Figura 16: Tabela do Restaurante do relatório fornecido

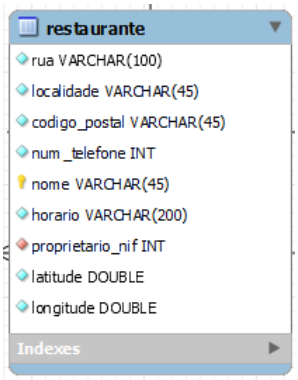

Figura 17: Tabela do Restaurante

- **Avaliação**

Esta tabela armazena toda a informação pertinente para as avaliações.

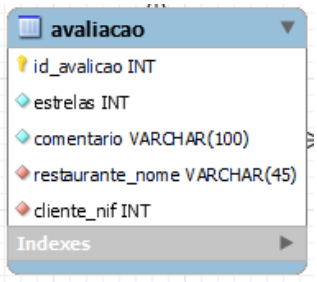

Figura 18: Tabela da Avaliação

Não foram efetuadas alterações relativas à tabela da avaliação, uma vez que esta possuía todas as informações necessárias ao suporte do requisito que permite que um cliente faça uma avaliação a um restaurante.

Esta terá uma ligação de n para 1 com a tabela Cliente, como já referido na secção da mesma, e possui também uma ligação de n para 1 com a tabela Restaurante como mencionado na tabela anterior.

- **Reserva**

Nesta tabela são armazenadas as informações importantes para uma reserva.

Figura 19: Tabela da Reserva do relatório fornecido

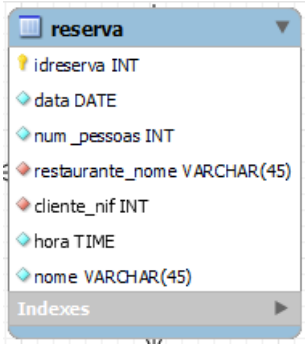

Figura 20: Tabela da Reserva

Para que fosse possível a implementação de alguns requisitos, nomeadamente a introdução da hora da reserva e do nome em que fica feita a reserva aquando da sua execução, adicionamos as colunas hora e nome, que não constavam na tabela proposta.

É de notar que esta tabela tem uma relação de n para 1 com a tabela Cliente, como referido na secção da mesma, e uma relação do mesmo tipo com o restaurante. Para além disso, possui uma relação de n para n com o prato, com auxílio da tabela de junção "engloba". Esta relação é justificada pela capacidade de uma reserva referir vários pratos, por um lado, e por um prato poder ser referenciado por várias reservas, por outro.

- **Prato**

Esta tabela armazena as informações relativas à entidade Prato.

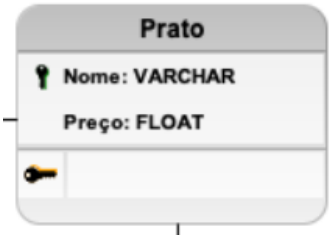

Figura 21: Tabela do Prato

Figura 22: Tabela do Prato do relatório fornecido

Relativamente a esta tabela, decidimos efetuar alterações a nível das colunas. Foi adicionada a coluna id, de forma a identificar o prato através de um número, em vez de este ser identificado pelo nome. A justificação para esta mudança reside na inconsistência que poderia gerar a nível conceptual o prato ter como identificador único o nome, uma vez que não poderiam existir dois pratos com o mesmo nome, mas com preços diferentes. Assim, tendo um identificador como chave primária, isto passará a ser possível.

A tabela Prato, como já referido na secção da tabela Reserva, possui uma relação de n para n com a mesma. Para além desta, tem também uma relação com a tabela Restaurante, como já referido na secção da mesma. Esta existe uma vez que um prato poderá constar no menu de vários restaurantes, bem como um restaurante poderá conter vários pratos.

- **Código QR**

Nesta tabela estarão armazenadas as informações relativas aos códigos existentes no sistema. Esta tabela surgiu com a necessidade da criação desta tabela de modo a suportar a funcionalidade dos códigos qr promocionais.

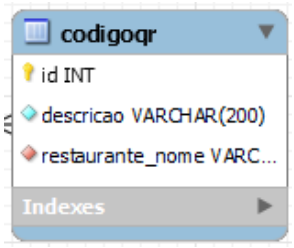

Figura 23: Tabela do QR code

Para que fosse possível aceder aos códigos QR de cada restaurante, quando estes fossem criados, teriam de ser armazenados na base de dados para posterior consulta. No entanto, guardar um caminho para um ficheiro não seria necessário, uma vez que o código QR propriamente dito poderia ser criado apenas quando fosse necessário apresentar. Assim, é guardada apenas a descrição relativa ao código introduzida pelo proprietário do restaurante, de forma que quando este for consultado, puder ser construído o código com a descrição obtida na base de dados.

Como este código é relativo a um restaurante, a tabela Codigo terá uma relação de n para 1 com a tabela restaurante, como já referido na secção respetiva.

3. Conclusões e Trabalho Futuro

No final do prazo estabelecido, foi produzido um sistema totalmente funcional, implementado a grande maioria de todos os requisitos especificados previamente, da forma o mais fiel possível a essa mesma especificação.

Tal como foi mencionado previamente, a implementação de alguns requisitos provocou mudanças relativamente ao que tinha sido explicitado. No entanto, essas mudanças excederam o previsto na apreciação da especificação realizada previamente ao trabalho de implementação. Esse facto, aliado ao tempo gasto para adquirir o conhecimento necessário para implementar uma Web App, levou a que, infelizmente, não fosse possível a implementação das funcionalidades do Administrador. Ainda assim, foi feito um esforço para tentar transferir algumas dessas responsabilidades deste utilizador para os restantes, como referido anteriormente.

A nível do trabalho futuro possível, o ponto de partida seria, então, implementar esse tipo de utilizador. Outra funcionalidade que seria melhorada seria o sistema de direções, que, futuramente, poder-se-ia implementar dentro da aplicação, não tendo, assim, de recorrer a redirecionamento para o Google Maps. E finalmente, gostariamos de ter adicionado o controlo de exceções a todas as funcionalidades presentes na aplicação.

Anexos

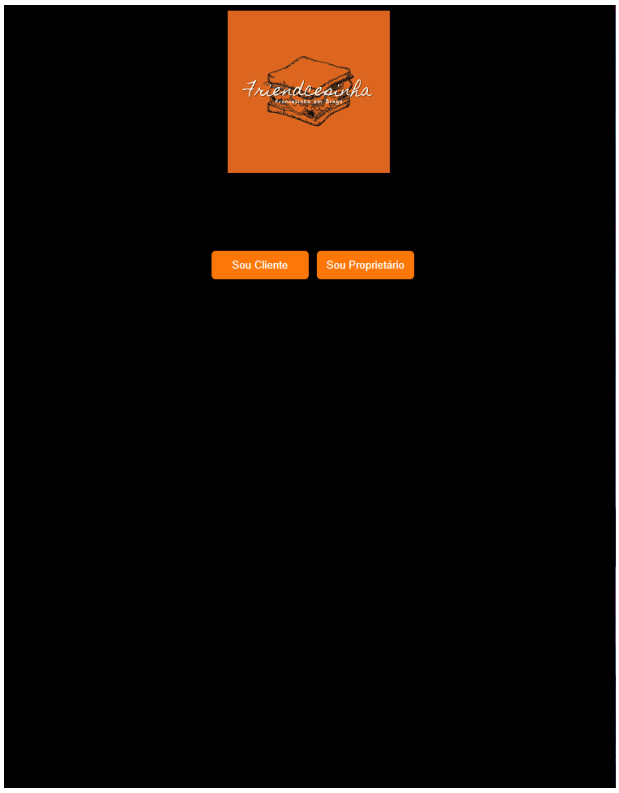

Figura 24: Interface Menu Inicial

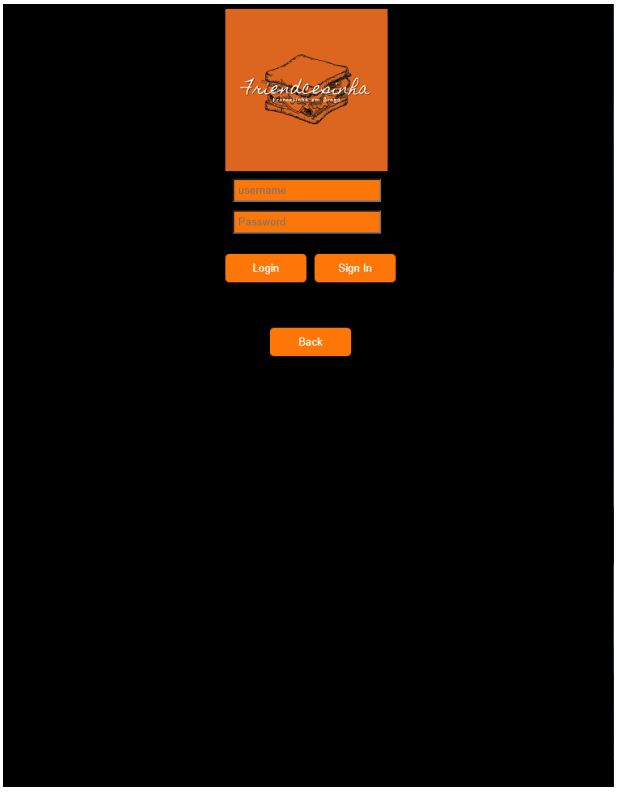

Figura 25: Interface Iniciar Sessão

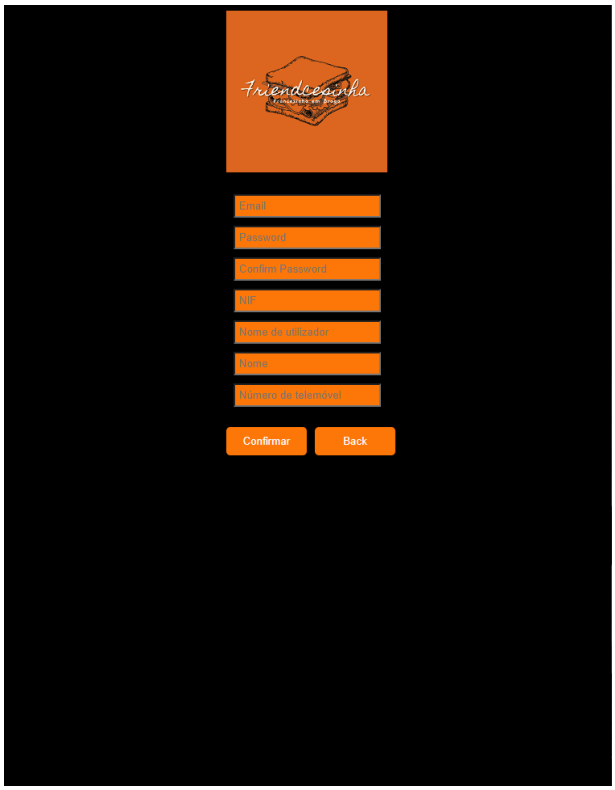

Figura 26: Interface Registar Cliente

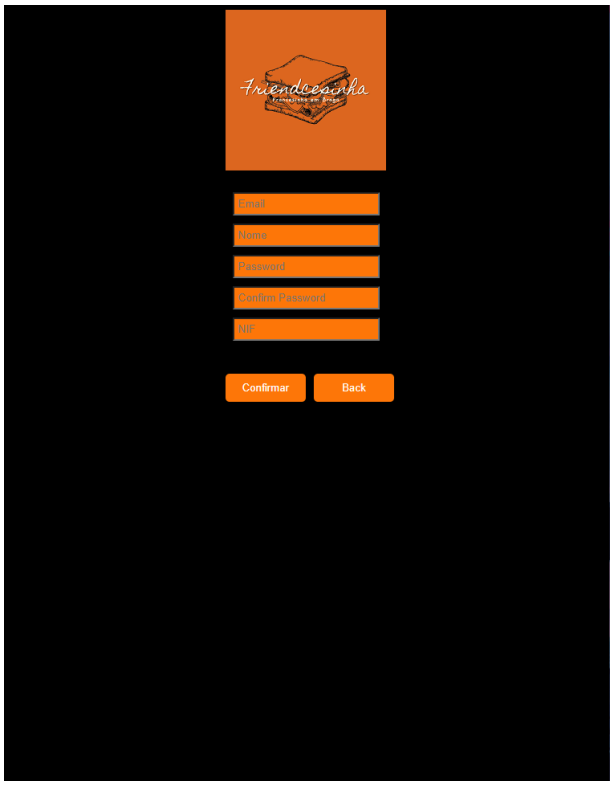

Figura 27: Interface Registar Proprietario

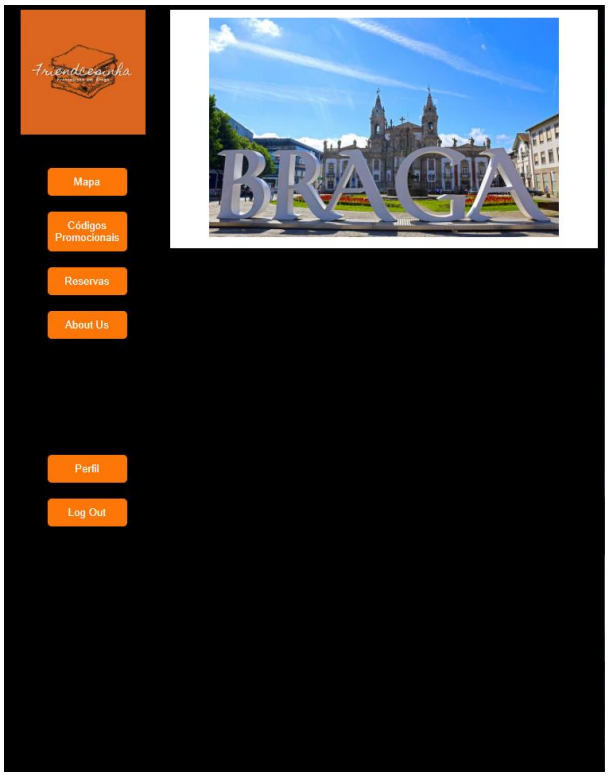

Figura 28: Interface Menu Cliente

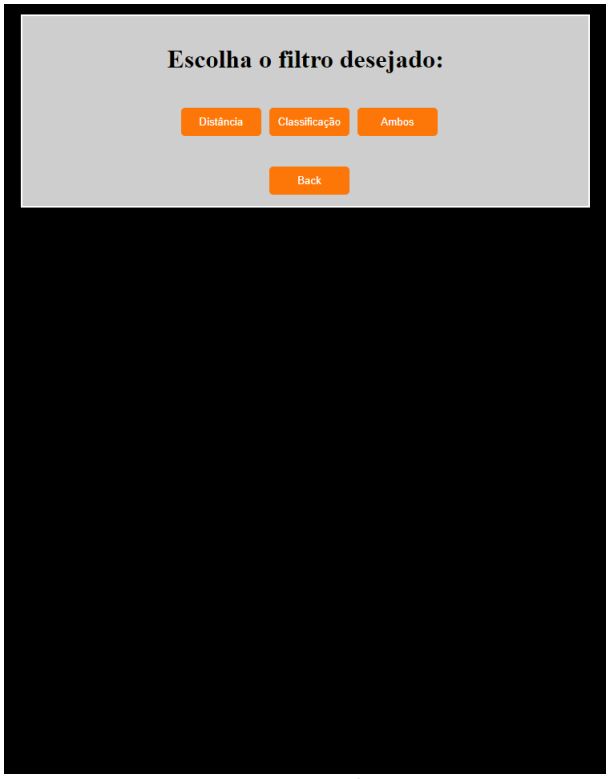

Figura 29: Interface Menu Intermédio Para o Mapa

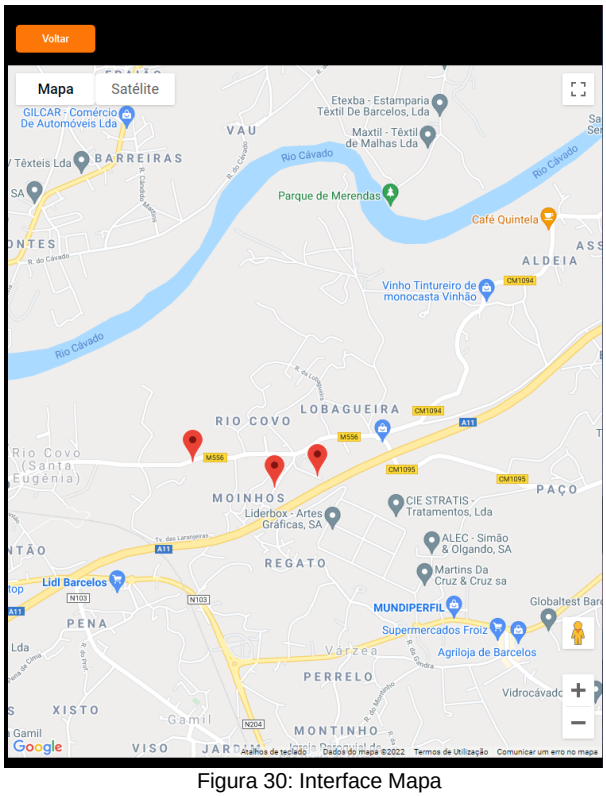

Figura 30: Interface Mapa

Figura 31: Interface Restaurante

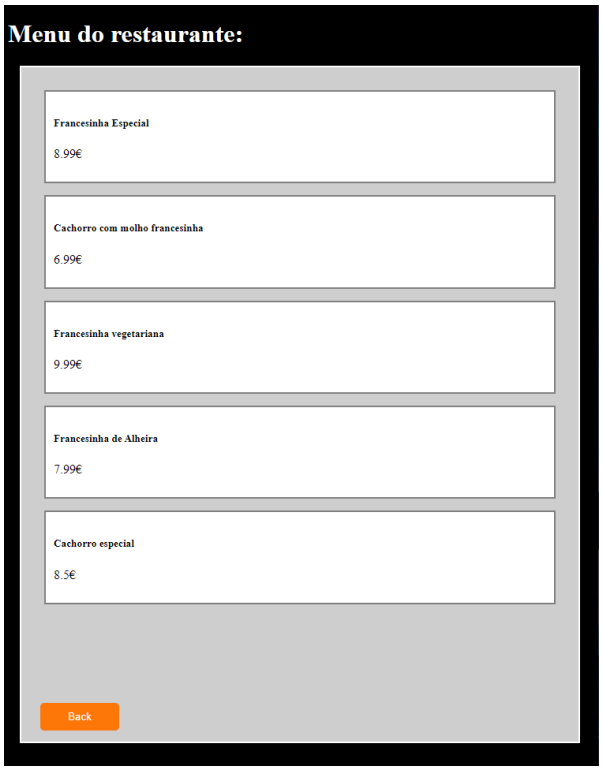

Figura 33: Interface Menu Restaurante

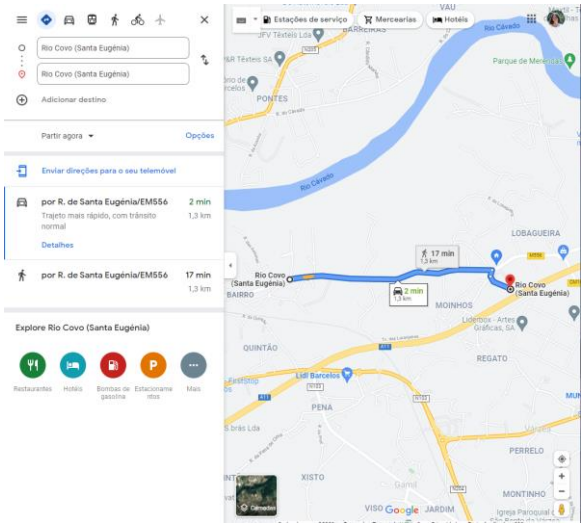

Figura 32: Página para o qual é redirecionado quando é pedido direções

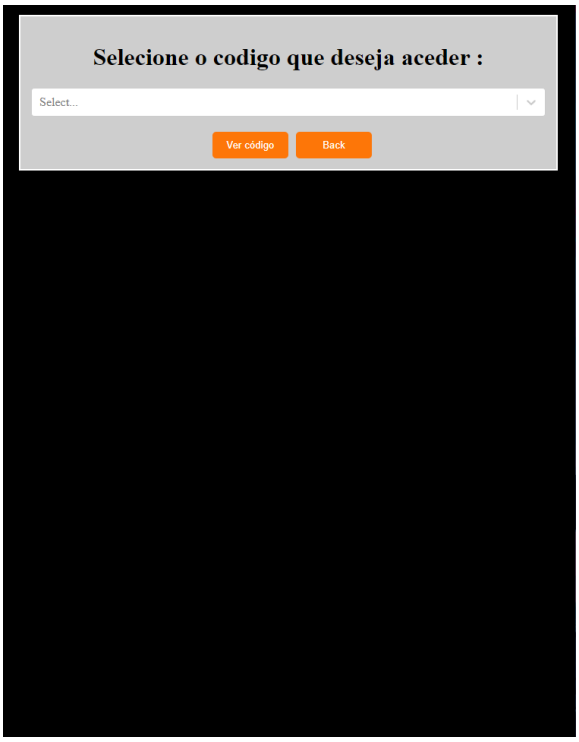

Figura 34: Interface Intermédio para Aceder ao Código

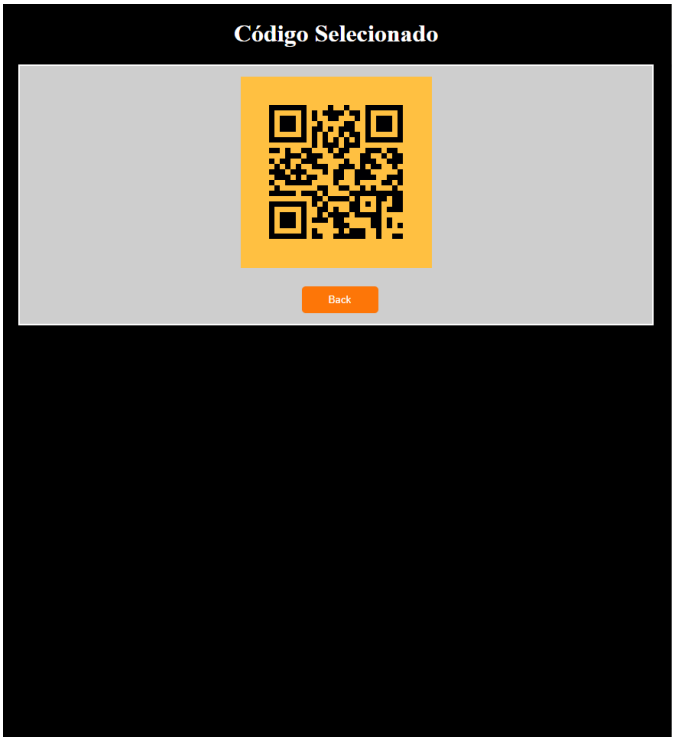

Figura 35: Interface Código

Figura 36: Interface Reservar

Figura 37: Código Escaneado

Figura 38: Interface Avaliar

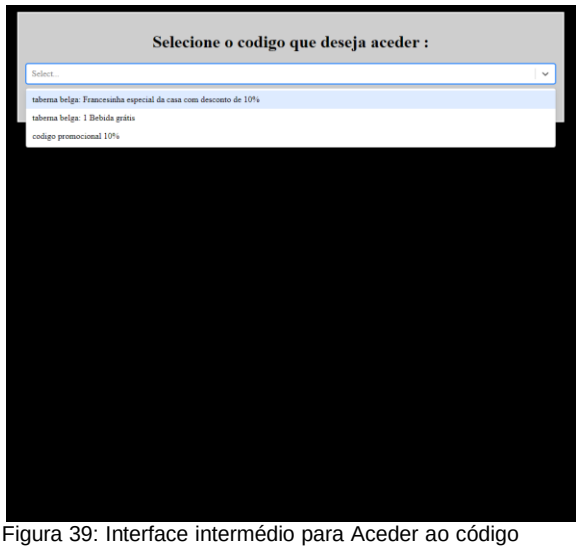

Figura 39: Interface intermédio para Aceder ao código

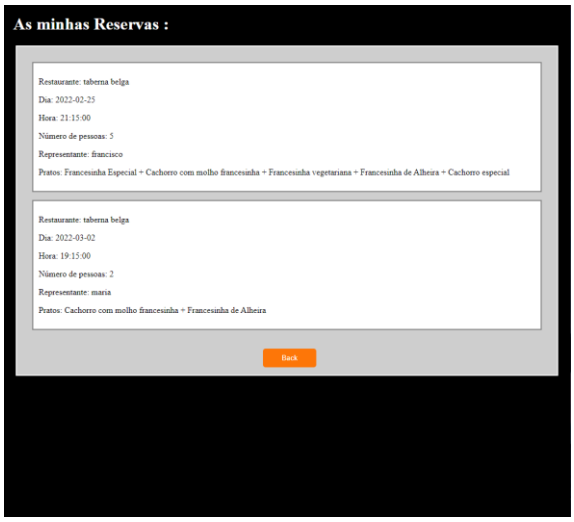

Figura 40: Interface Consultar Reservas

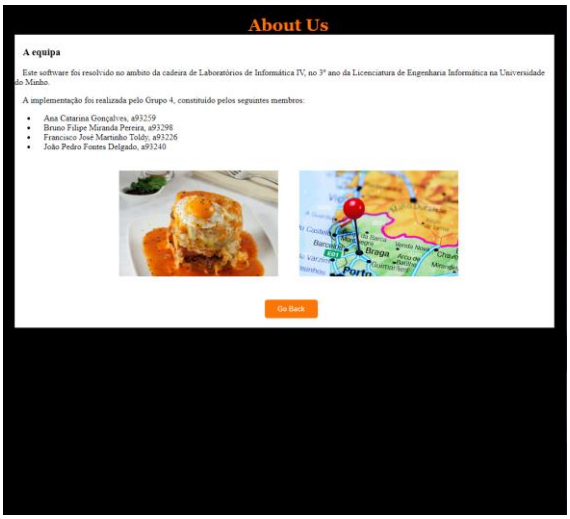

Figura 41: Interface About Us

Figura 42: Interface Editar Perfil

Figura 43: Interface Menu Proprietário

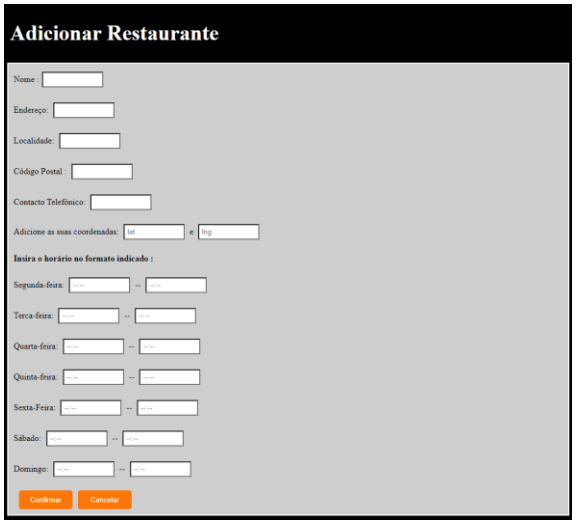

Figura 44: Interface Adicionar Restaurante

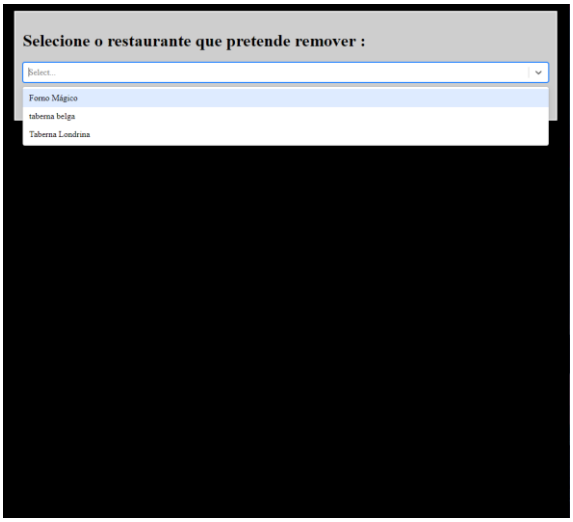

Figura 45: Interface Remover Restaurante

Figura 46: Interface intermédia para Editar Restaurante

Figura 47: Interface Editar Restaurante

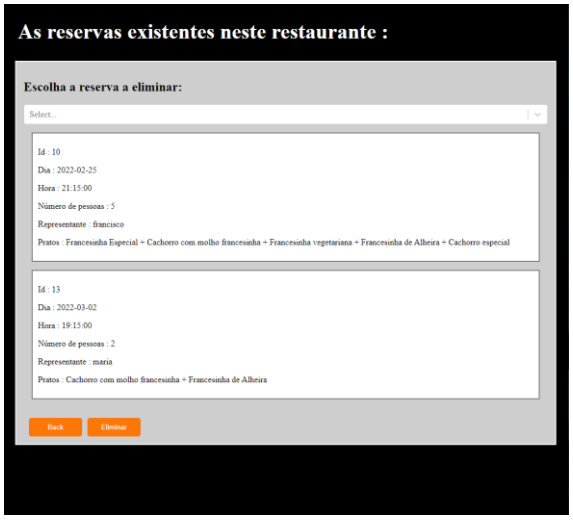

Figura 48: Interface intermédia para Eliminar Reserva

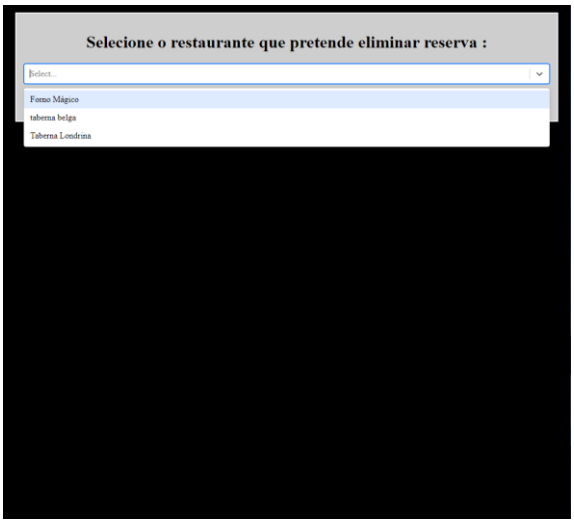

Figura 49: Interface Eliminar Reserva

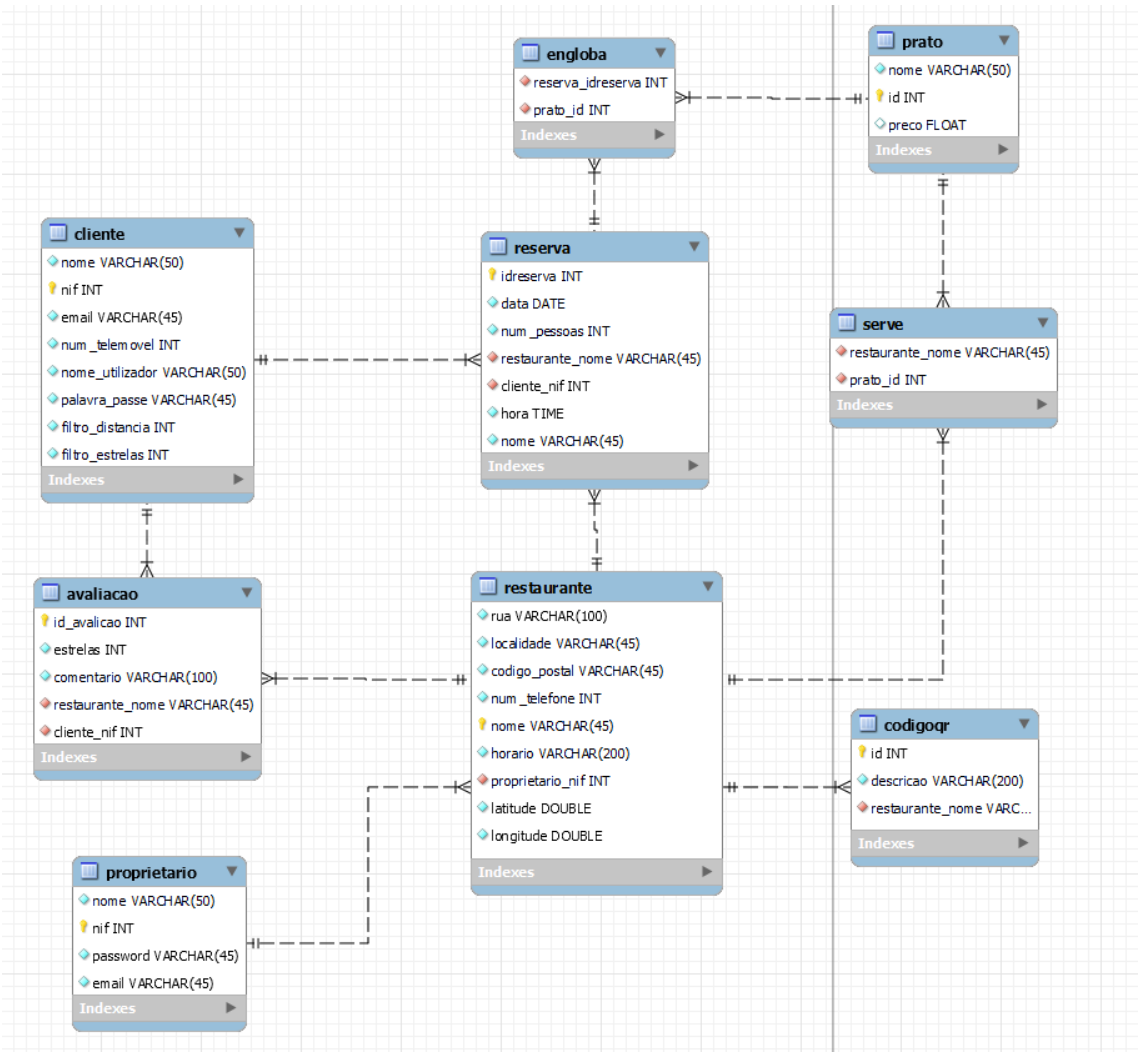

Figura 50: Base de Dados Final