



Universidade do Minho

Escola de Engenharia

Mestrado Integrado em Engenharia Informática

Unidade Curricular de Laboratórios de Informática IV

Ano Letivo de 2020/2021

Guia Turístico

Grupo 5

Ana Murta (A93284) Ana Henriques (A93268) Leonardo Freitas (A93281)
Rui Coelho (A58898) Tiago Carneiro (A93207)

janeiro, 2022

L | 4

Data de Recepção	
Responsável	
Avaliação	
Observações	

Guia Turístico

Grupo 5

Ana Murta (A93284) Ana Henriques (A93268) Leonardo Freitas (A93281) Rui Coelho (A58898) Tiago Carneiro (A93207)

janeiro, 2022

Resumo

Com o aumento significativo da carga de trabalho no quotidiano das pessoas, a capacidade de escolha de bons pontos turísticos onde desfrutar o tempo livre encontra-se diminuta. O projeto *Add&SEEK* surge para auxiliar os seus utilizadores a nestas decisões: através sua plataforma *web*, o processo de pesquisa de locais de interesse é facilitado, podendo o utilizador planear, de uma forma cómoda e simples, como ocupar o seu tempo livre.

Ao longo do presente relatório serão dados a conhecer diversos pontos relevantes acerca da plataforma desenvolvida, tais como a sua caracterização, as especificações inicialmente levantadas e as diversas funcionalidades efetivamente implementadas para o *software*. Serão, também, abordados aspectos de teor técnico referentes à arquitetura de implementação adotada para o desenvolvimento do sistema – desde o desenvolvimento do sistema de dados, até ao desenvolvimento do sistema responsável pela interação com o utilizador.

Área de Aplicação: Engenharia de Software, Desenho e Arquitetura de Sistemas de Base de Dados, Sistemas de Recomendação, Guia para Pontos Turísticos.

Palavras-Chave: Planeamento de *software*, Bases de Dados Relacionais, *Unified Modelling Language*, Diagrama de Gantt, Aplicação *web-based*, Plataforma, Pontos Turísticos.

Índice

1	Introdução	1
1.1	Enquadramento geral	1
1.2	Apresentação do Caso de Estudo	2
1.3	Recursos Necessários	3
1.4	Plano de desenvolvimento	4
2	Desenvolvimento da Aplicação	5
2.1	Estratégia de desenvolvimento adotada	5
2.2	Caracterização da aplicação e dos seus serviços	5
2.3	Arquitetura geral da aplicação e descrição de componentes	8
2.4	Manual de utilização	14
3	Conclusões e Trabalho Futuro	18
4	Anexos	20
4.1	Criação da base de dados	20
4.2	Povoamento da base de dados	22

Lista de Figuras

1.1	Modelo de Domínio.	2
1.2	Diagrama de Gantt do projeto.	4
2.1	Arquitetura da Base de Dados.	9
2.2	Diagrama de Componentes	10
2.3	Diagrama de Classes fornecido pela equipa de especificações.	11
2.4	Diagrama de Classes usado pela equipa de desenvolvimento.	12
2.5	Diagrama de dependências do <i>front-end</i> .	13
2.6	Página inicial.	14
2.7	Página de <i>login</i> .	15
2.8	<i>Login</i> efetuado com sucesso.	15
2.9	Página de <i>signup</i> .	16
2.10	<i>Signup</i> efetuado com sucesso.	16
2.11	Página de perfil.	17
2.12	Filtragem de lugares turísticos através de filtros	17

1 Introdução

1.1 Enquadramento geral

O presente relatório pretende acompanhar o desenvolvimento de uma aplicação *web*, que consiste num guia para locais de interesse. O projeto aqui detalhado, designado *add&SEEK*, tem como objetivo principal auxiliar na resolução de um problema comum do quotidiano: como ocupar, da melhor forma, o tempo livre? Atualmente, o dia-a-dia das pessoas é bastante atarefado, sendo que isso resulta na falta de tempo para planear visitas a locais turísticos, de modo a que seja possível relaxar e descansar.

Deste modo, a aplicação a desenvolver procura responder a este problema, facilitando, por um lado, a vida dos seus utilizadores e, por outro, estimulando a economia local, movendo mais visitantes para as diversas atrações – sejam estas monumentos, parques, ou até mesmo restaurantes. Com efeito, a aplicação visa disponibilizar uma lista de locais turísticos, de tal modo a que seja possível elaborar um plano das atividades a realizar. Através de um conjunto de filtros é possível limitar as opções apresentadas, como por exemplo através da distância do local até ao utilizador, da classificação ou da categoria do mesmo. Adicionalmente, é também possível que um utilizador elabore avaliações de um determinado local, e que mantenha uma lista dos seus locais favoritos – podendo estes ser adicionais ou eliminados quando o utilizador desejar. Finalmente, a aplicação deve permitir a criação de um plano, associado a uma cidade, e um horário para efetuar visita a um conjunto de locais.

A Figura 1.1 representa o modelo de domínio desenvolvido pela equipa de especificações, com recurso à *Unified Modeling Language* (UML). Deste modo, é possível obter uma imagem mais clara da representação abstrata do sistema, e de como as diversas entidades do mesmo se relacionam.

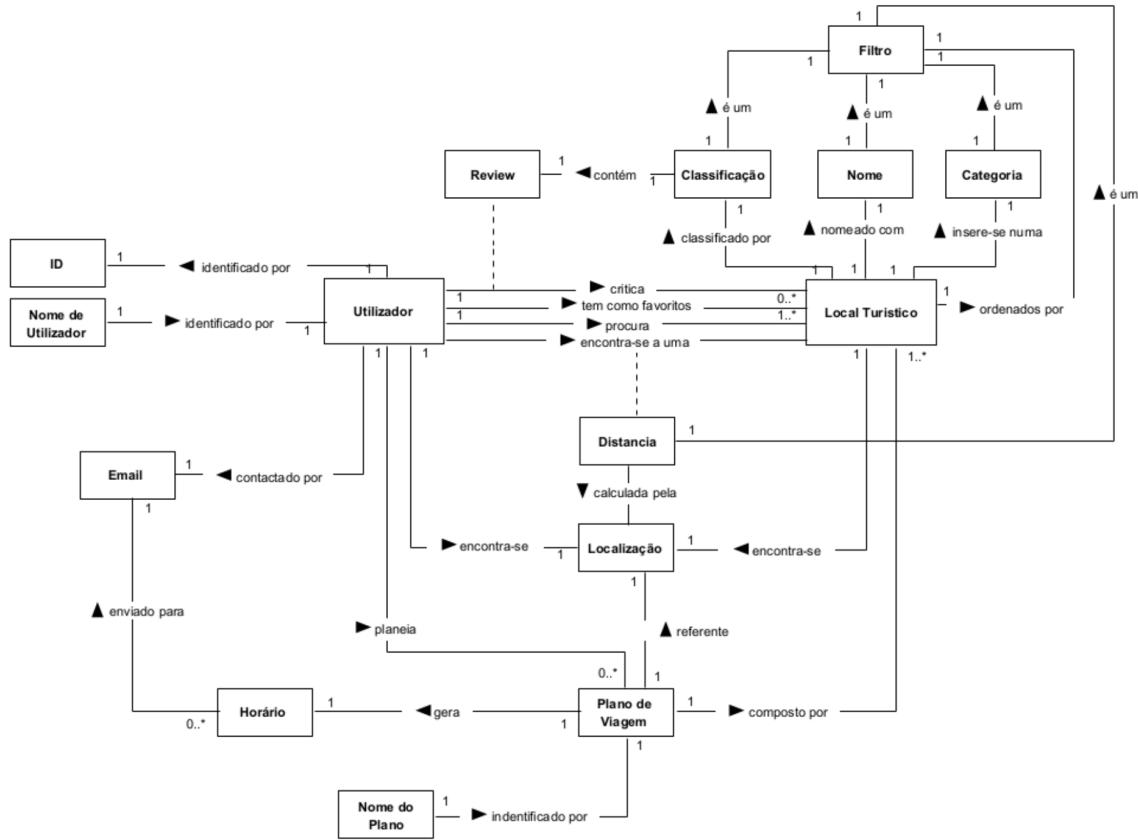


Figura 1.1: Modelo de Domínio.

1.2 Apresentação do Caso de Estudo

Como foi supramencionado, este *website* pretende facilitar as vidas dos utilizadores, enquanto simultaneamente estimula a economia local. A aplicação apresenta-se como uma alternativa a outras disponibilizadas no mercado, revelando-se inovadora pela sua simplicidade e pelas funcionalidades que oferece. Para além disto, tal como foi referido, o *website* disponibiliza um leque de locais turísticos, listados de modo a disponibilizar um leque vasto de opções que possam satisfazer as preferências do utilizador.

Para o uso da aplicação, o utilizador necessita inevitavelmente de fornecer a sua localização aquando do *login* na sua conta – que necessita da introdução do seu email e da respetiva palavra-passe. Para efetuar um registo, é necessário inserir um *username* válido, i.e., que ainda não tenha sido usado por nenhum utilizador, o email e a palavra-passe. Após estar autenticado na plataforma, o utilizador tem acesso às várias funcionalidades disponíveis. Entre estas podem ser realçadas a possibilidade de editar o seu perfil, os seus locais favoritos e os seus planos.

Aditivamente, um utilizador pode ainda operar sobre os planos, consultando os planos existentes ou criando um novo. A criação de um plano requer que o utilizador forneça o nome, o dia em que ocorre e o intervalo de tempo de visita dos locais. O utilizador tem acesso a uma lista de todos os seus planos, ordenados em função dos dias da semana em que ocorrem.

Por fim, a *add&SEEK* oferece ao utilizador a opção de aplicar filtros à sua pesquisa, apresentando uma lista dos locais que validam os filtros aplicados. Ao selecionar um local desta lista, o utilizador é redirecionado para a página correspondente ao mesmo, onde é apresentada uma descrição, a classificação geral e as avaliações efetuadas ao local – o utilizador pode, neste ponto, também efetuar uma avaliação do local pesquisado, ou até mesmo selecioná-lo como destino.

1.3 Recursos Necessários

A implementação da plataforma pode ser separada em três fases: implementação do sistema de dados; desenvolvimento do *back-end* do website; e criação do *front-end*.

A implementação do sistema de dados recorreu à *Structured Query Language*, vulgo, SQL, tendo sido escolhido o MySQL como sistema de gestão de base de dados. O desenvolvimento da base de dados foi efetuado através do MySQL Server, tomando como base o modelo fornecido pelo grupo encarregue de efetuar a especificação do sistema.

A construção da lógica de negócio da plataforma foi desenvolvida com recurso à linguagem de programação orientada aos objetos JAVA. A escolha particular por esta linguagem deveu-se, principalmente, pela familiaridade da equipa de desenvolvimento com a mesma – o que possibilitou a execução de um trabalho mais rigoroso e cuidado por parte da equipa. Como *Integrated Development Environment* foi usado o IntelliJ IDEA, pelo vasto suporte que oferece para a programação em JAVA – particularmente no que diz respeito às ferramentas de *debug* que oferece.

A integração deste componente com o *front-end* foi efetuada com recurso à *framework* Spring Boot. A escolha desta *framework* deveu-se, essencialmente, ao facto de que recorre à linguagem de programação JAVA para o desenvolvimento de serviços web. Com recurso a JAVA e JavaScript, vulgo JS, foi desenvolvida a camada responsável por interligar os dois grandes componentes da aplicação.

O serviço de georeferência utilizado ao longo do projeto consiste na *Application Programming Interface* (API) disponibilizada pelo *Bing Maps*. Através desta ferramenta foi possível determinar a localização atual do utilizador, para que possa ser efetuada uma pesquisa dos locais disponíveis, com a apresentação dos mesmos no mapa.

Por fim, o *front-end* foi elaborado com recurso a *HyperText Markup Language* (HTML) e *Cascading Style Sheets* (CSS), que permitem a exposição das diversas funcionalidades desenvolvidas no *back-end* ao utilizador. A escolha destas ferramentas deve-se à sua popularidade

no desenvolvimento de aplicações web.

1.4 Plano de desenvolvimento

O presente projeto faz-se acompanhar por um plano de desenvolvimento que visa dar a conhecer o planeamento idealizado para a corrente fase. A concreta definição de um plano estruturado e organizado é essencial para a realização de um bom projeto, pois possibilita que o trabalho ocorra de forma faseada, permitindo efetuar avaliações periódicas do progresso alcançado.

A Figura 1.2 apresenta o Diagrama de Gantt desenhado para o desenvolvimento da aplicação *add&SEEK*, contemplado exclusivamente as diversas tarefas essenciais para a construção da plataforma na fase de desenvolvimento corrente.

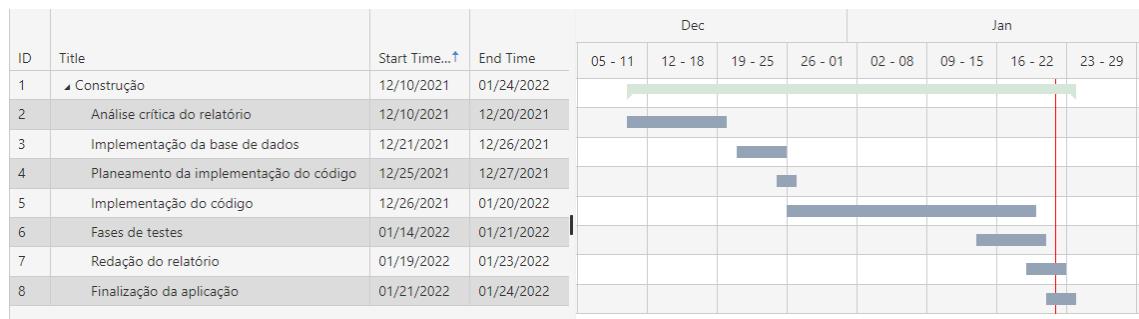


Figura 1.2: Diagrama de Gantt do projeto.

2 Desenvolvimento da Aplicação

2.1 Estratégia de desenvolvimento adotada

O desenvolvimento do projeto iniciou-se por uma leitura cuidada e atenta às especificações previamente fornecidas. Após a análise das especificações, foi extraída a informação de interesse referente: i) aos requisitos que a aplicação deve cumprir; ii) às arquiteturas de desenvolvimento – quer do sistema de dados, quer da lógica de negócios inerente à plataforma. De um modo geral, as especificações foram seguidas de perto, tendo sido efetuadas algumas alterações de modo a ser possível responder a um maior número de requisitos especificados pela equipa anterior – alterações que serão referidas posteriormente.

O desenvolvimento da aplicação passou, tal como foi supramencionado, pela elaboração de três componentes – sistema de dados, *back-end* e *front-end* – tendo o seu desenvolvimento sido delegado pelos membros da equipa responsável pela criação do sistema. De modo a promover o envolvimento de todos os elementos nas diversas componentes do projeto, foram elaboradas diversas reuniões para dar a conhecer as propostas de soluções pensadas e para poderem ser tomadas decisões, em grupo, relativas à arquitetura e estrutura da aplicação – tal facto possibilitou que os diversos elementos contribuissem de uma forma ativa para o desenvolvimento do projeto.

2.2 Caracterização da aplicação e dos seus serviços

Tal como foi especificado, a aplicação *add&Seek* deve respeitar um conjunto de requisitos funcionais e não funcionais. Como foi previamente referido, para aceder às funcionalidades da aplicação um utilizador necessita de se encontrar autenticado. Para tal, e para efetuar o registo da sua conta, é imperativo que esta forneça um endereço de email. A alteração de dados deve ser também suportada, podendo o utilizador alterar o seu e-mail, palavra-passe, os seus locais favoritos e os planos de viagem. Adicionalmente, e tendo em conta a utilização dos serviços de geolocalização, o utilizador necessita de providenciar as coordenadas *Global Positioning System* (GPS), para que possam ser aplicados filtros que limitem as opções apresentadas em função da distância. Aditivamente, a plataforma deve encontrar-se em constante funcionamento e deve primar por assegurar a privacidade e a segurança dos dados dos utilizadores.

De modo a obter uma clara visão das funcionalidades a suportar pela aplicação, segue-

se uma lista com os diversos requisitos, funcionais, levantados pela equipa encarregue da especificação do sistema:

1. Registo na aplicação
2. Autenticação na aplicação
3. Edição da informação do perfil do utilizador
4. Pesquisa de locais turísticos
5. Apresentação da página de um local turístico
6. Listagem dos planos de viagem
7. Criação de um plano de viagem
8. Consulta de um plano de viagem
9. Edição de um plano de viagem

O cumprimento destas funcionalidades encontra-se intimamente relacionado com as especificações previamente definidas pela equipa encarregue das mesmas. Considerando este facto, resta enumerar as funcionalidades efetivamente implementadas, e as alterações adoptadas à abordagem proposta pela equipa de especificação. As poucas alterações realizadas, posteriormente descritas, procuraram permitir a implementação das funcionalidades acima supracitadas. Importa, aqui, referir que a equipa de desenvolvimento procurou trabalhar o mais perto possível das configurações iniciais definidas pela equipa de especificação do sistema, contudo as poucas alterações realizadas não foram suficientes para desenvolver completamente as funcionalidades previstas inicialmente.

Assim sendo, efetua-se, agora, uma listagem das diversas funcionalidades efetivamente implementadas, com uma breve descrição das mesmas:

1. Registo na aplicação: através do fornecimento dos dados relevantes (*nome, palavra-passe, email*) o utilizador é capaz de se registar na plataforma;
2. Autenticação na aplicação: o utilizador fornece as credenciais de acesso e efetua a autenticação – que é validada pela plataforma, conseguindo interagir com a mesma apenas no caso de estas serem válidas;
3. Edição da informação do perfil do utilizador: o sistema permite que o utilizador atualize a informação do seu perfil (*email, password* e planos de viagem). A edição dos lugares favoritos não foi implementada, uma vez que a camada de dados não prevê a persistência desta lista de lugares;
4. Pesquisa de locais turísticos: a aplicação de filtros na pesquisa de locais encontra-se implementada, permitindo ao utilizador filtrar locais em função do nome, da categoria

e da sua distância ao local. A seleção de um local de interesse redireciona o utilizador para a página correspondente do local;

5. Apresentação da página de um local turístico: cada página de um local de interesse apresenta a informação relativa ao mesmo, tal como a sua classificação e as avaliações efetuadas. O utilizador pode efetuar uma avaliação do local e pode adicioná-lo ao seu plano de viagem;
6. Listagem dos planos de viagem: a plataforma disponibiliza a apresentação dos planos associados ao utilizador. É, aqui, importante referenciar que os dados dos planos não podem ser gravados, na íntegra, na base de dados. Cada plano possui um identificador único que o permite associar ao utilizador que o criou, contudo considerando o sistema de dados proposto apenas é possível ter um único plano associado a um utilizador. Como tal, será sempre armazenado no sistema de dados o plano mais recente do utilizador. Adicionalmente, a informação dos locais associados ao plano também não é armazenada, uma vez que a relação definida para os planos não incorpora um campo que permita tal funcionalidade;
7. Criação de um plano de viagem: o utilizador pode criar um novo plano, fornecendo os dados relativos ao nome do plano, ao intervalo de tempo associado e à cidade referente ao mesmo;
8. Consulta de um plano de viagem: a plataforma permite que um utilizador aceda aos planos, consultando a informação referente aos mesmos. No que diz respeito aos locais de interesse, a plataforma não prevê informação associada ao tempo disponibilizado para as visitas dos diversos locais – uma vez que tal informação se encontra omissa no sistema de dados proposto;
9. Edição de um plano de viagem: a edição de um plano na plataforma permite adicionar novos locais ao plano – mesmo estes não sendo posteriormente guardados na base de dados. O trabalho efetuado sobre o tempo dos locais não foi implementado devido à falha de informação necessária para o efeito no sistema de dados.

Importa, agora, realizar uma pequena nota acerca da implementação das funcionalidades. A base de dados estruturada pelo grupo de especificação encontra-se desenvolvida de um modo pobre para o desenvolvimento completo das funcionalidades inicialmente especificadas. Como tal, a equipa de desenvolvimento tentou procurar alternativas para, respeitando de perto o modelo de construção do sistema de dados proposto, implementar as funcionalidades fielmente ao que tinham sido descritas.

Deve ser, também, realçado que o diagrama de classes apresentado pela equipa de especificações prevê informação adicional à que se encontra armazenada na base de dados, pelo que tal efeito permitiu alguma mobilidade para a equipa de desenvolvimento implementar as funcionalidades acima descritas. Deste modo, após o carregamento dos dados presentes no sistema de dados foram introduzidos valores complementares para que fosse possível implementar um maior número de funcionalidades – mesmo esta informação não sendo armazenada em

momento algum no sistema de dados. Tal facto condicionou o modo como o acesso ao sistema de dados é efetuado, sendo, portanto, efetuado um carregamento da informação do sistema de dados ao arrancar a aplicação, e um armazenamento periódico da informação contida na lógica de negócio – significando que, na eventualidade de uma falha no sistema, ou de outro evento, que impeleque o *reboot* da plataforma, é perdida a informação que não é aramazenada na base de dados, não havendo persistência de todos os dados gerados pela aplicação.

O desenvolvimento da lógica de negócio possibilitou colmatar algumas das falhas presentes no desenvolvimento do sistema de dados. Tomando como base o diagrama de classes providenciado pela equipa de especificação, foi possível construir a plataforma. A construção dos diversos métodos usados no desenvolvimento da plataforma tomou como base os diagramas de sequência disponibilizados. De modo a maximizar a implementação das funcionalidades já descritas, a assinatura de alguns métodos foi ajustada – tendo igualmente sido necessário a criação de métodos adicionais de modo a facilitar o desenvolvimento da lógica de negócio inerente à plataforma. A título de exemplo, os métodos responsáveis por filtrar os diversos locais em função de determinadas características passaram a devolver um conjunto de lugares que respeitam esses esses parâmetros – em oposição a retornar *void* como inicialmente tinha sido definido.

2.3 Arquitetura geral da aplicação e descrição de componentes

A presente seção do relatório visa demonstrar as opções arquiteturais tomadas pela equipa de desenvolvimento, para a criação da plataforma *add&seek*. O modelo de domínio, previamente apresentado na Figura 1.1, constituiu A base para a divisão estrutural da aplicação, uma vez que permitiu identificar os diversos componentes a ser usados para a organização do *software*. Tal como foi mencionado previamente referido, a aplicação integra três grandes componentes: sistema de dados, *back-end* e *front-end* – seguidamente descritas em maior detalhe.

O sistema de dados, desenvolvido e mantido com *MySQL Server*, seguiu de perto o modelo apresentado na Figura 2.1, criado pela equipa de especificações. A construção do modelo referente à base de dados foi efetuado através da replicação do modelo apresentado na Figura 2.1, em conjunto com a ferramenta de *forward engineering*, que possibilitou gerar o *scrpit*¹, em SQL, de criação da base de dados modelada. Após a criação do sistema, foram povoadas as tabelas, com recurso a um *scrpit*², escrito em SQL, com os dados necessários para poder operar sobre o sistema de dados, visando efetuar alguns testes das funcionalidades aquando da sua implementação.

¹O *script* usado para o efeito pode ser consultado no Anexo 4.1

²Parte do *script* usado pode ser observado no Anexo 4.2

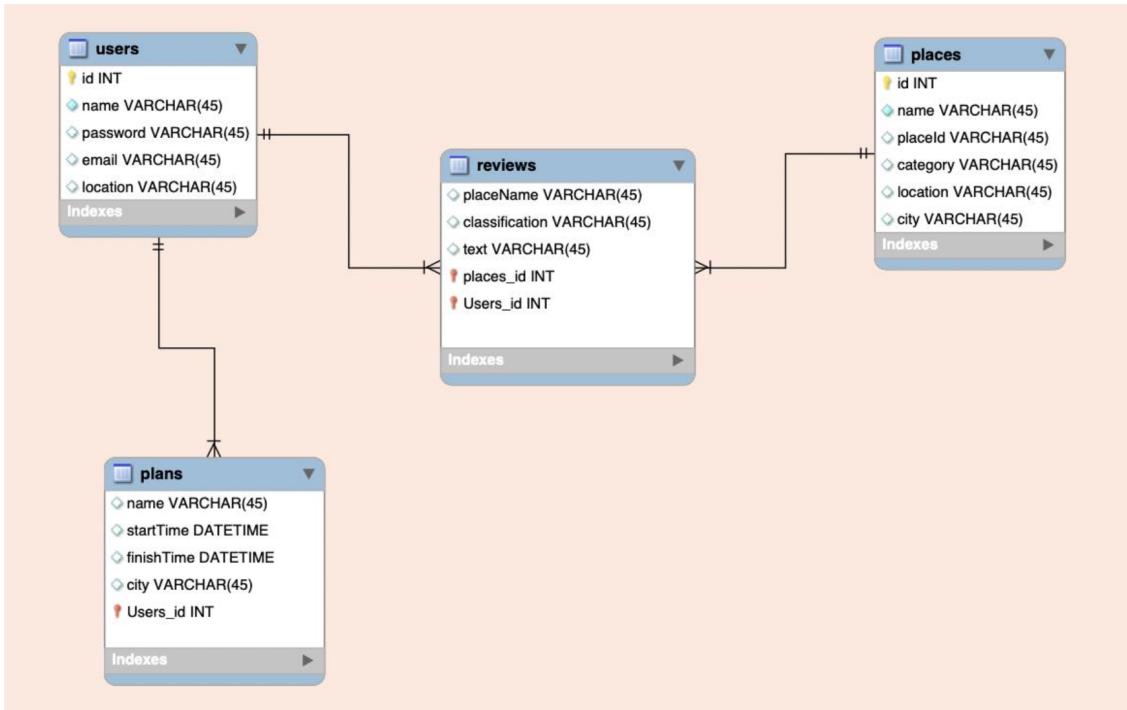


Figura 2.1: Arquitetura da Base de Dados.

No que diz respeito à camada de negócio, correspondente ao *back-end* da aplicação, esta foi organizada em diversos componentes, vulgo *packages*, de modo a permitir um desenvolvimento claro e modular do código, atribuindo aos diversos componentes a possibilidade de efetuarem um conjunto específico de operações. A Figura 2.2 apresenta o diagrama de componentes proposto pela equipa de especificações, que representa os subsistemas a implementar no sistema, juntamente com as interações e dependências entre os componentes que os integram. A equipa de desenvolvimento procurou seguir esta estruturação, uma vez que parecia representar de um modo claro os principais subsistemas que poderiam ser inferidos através do modelo de domínio apresentado anteriormente. Contudo, a agregação dos diversos módulos desenvolvidos, não seguiu a abordagem sugerida pela equipa de especificações, uma vez que se tornou, à medida que o código foi desenvolvido, difícil manter a clara separação entre os subsistemas.

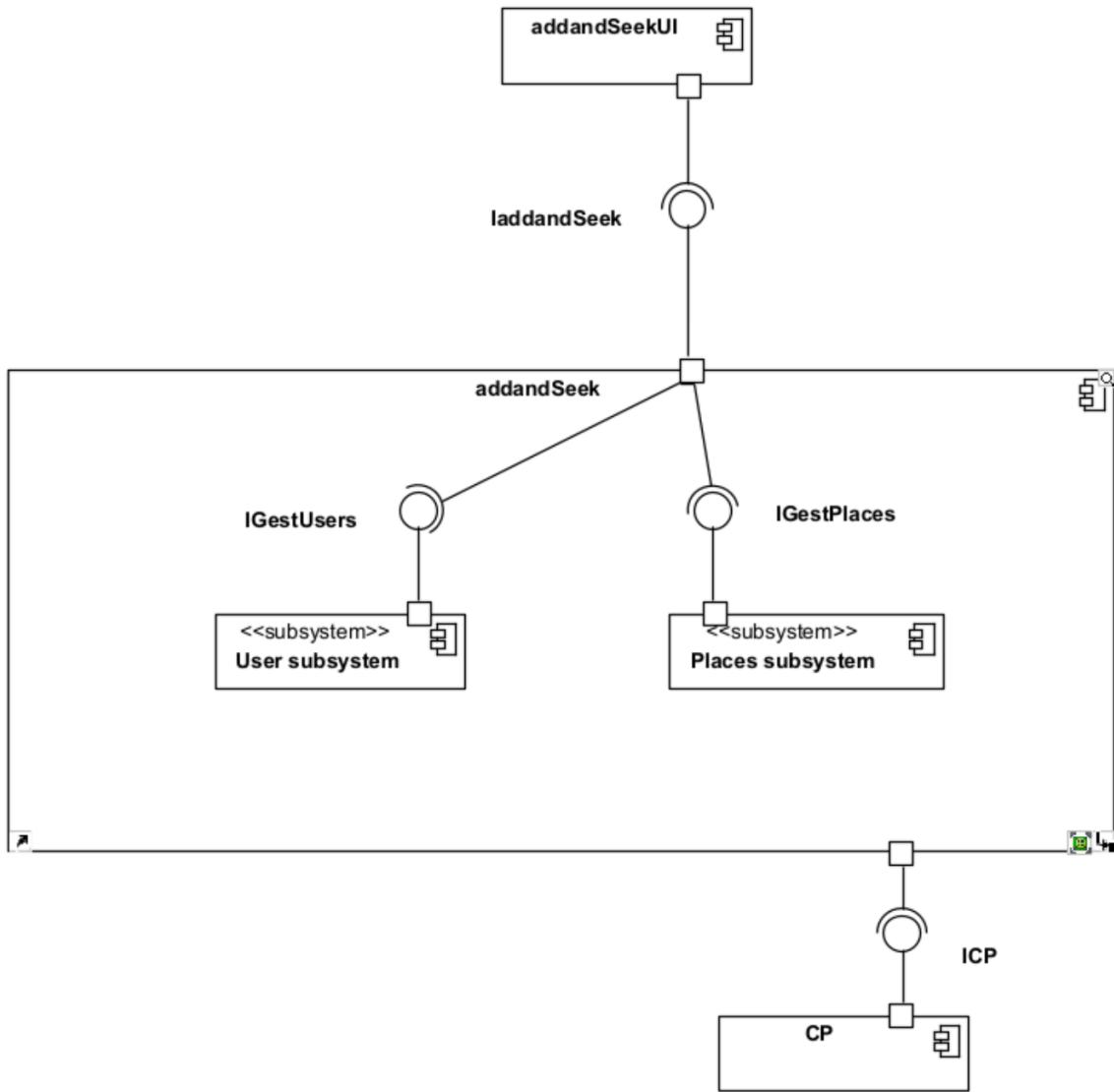


Figura 2.2: Diagrama de Componentes

Focando um pouco nos componentes que integram o sistema da camada da lógica de negócio, a restante estruturação teve como base o diagrama de classes fornecido aquando da especificação. O diagrama de classes, apresentado na Figura 2.3, representou uma peça fundamental para o desenvolvimento da plataforma, uma vez que permitiu saber de forma clara que classes deveriam ser implementadas e que métodos deveriam possuir essas classes.

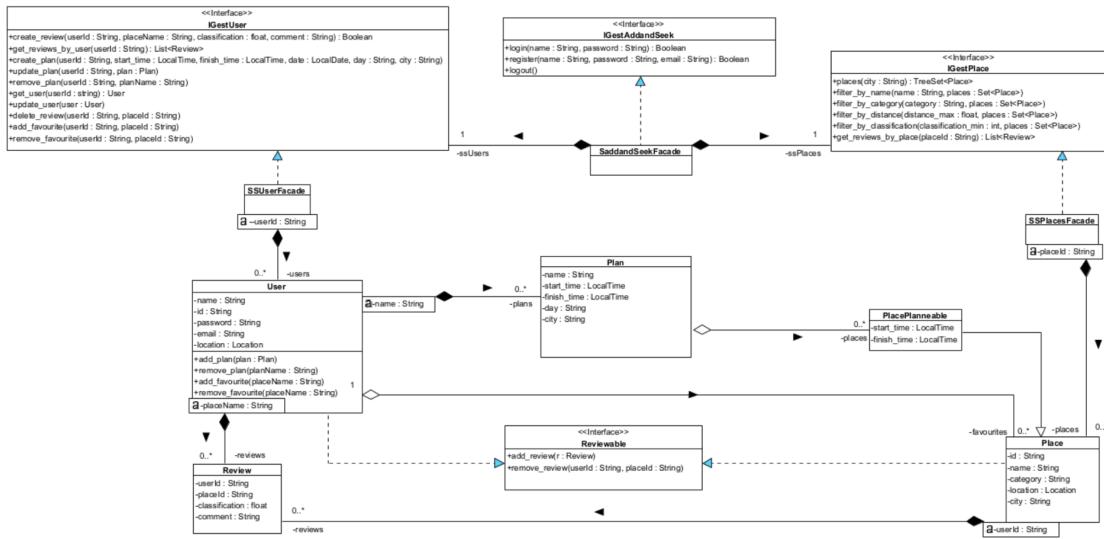


Figura 2.3: Diagrama de Classes fornecido pela equipa de especificações.

Como foi referido, as especificações providenciadas das classes, das suas variáveis e dos seus métodos foram seguidas com bastante rigor, tendo sido efetuadas pequenas alterações. Estas alterações podem ser observadas na Figura 2.4 que apresenta o diagrama de classes da plataforma construída. Estas alterações visaram, essencialmente, facilitar a implementação das diversas funcionalidades da plataforma e, em alguns casos, obter meios para poder efetuar eventuais controlos de erros.

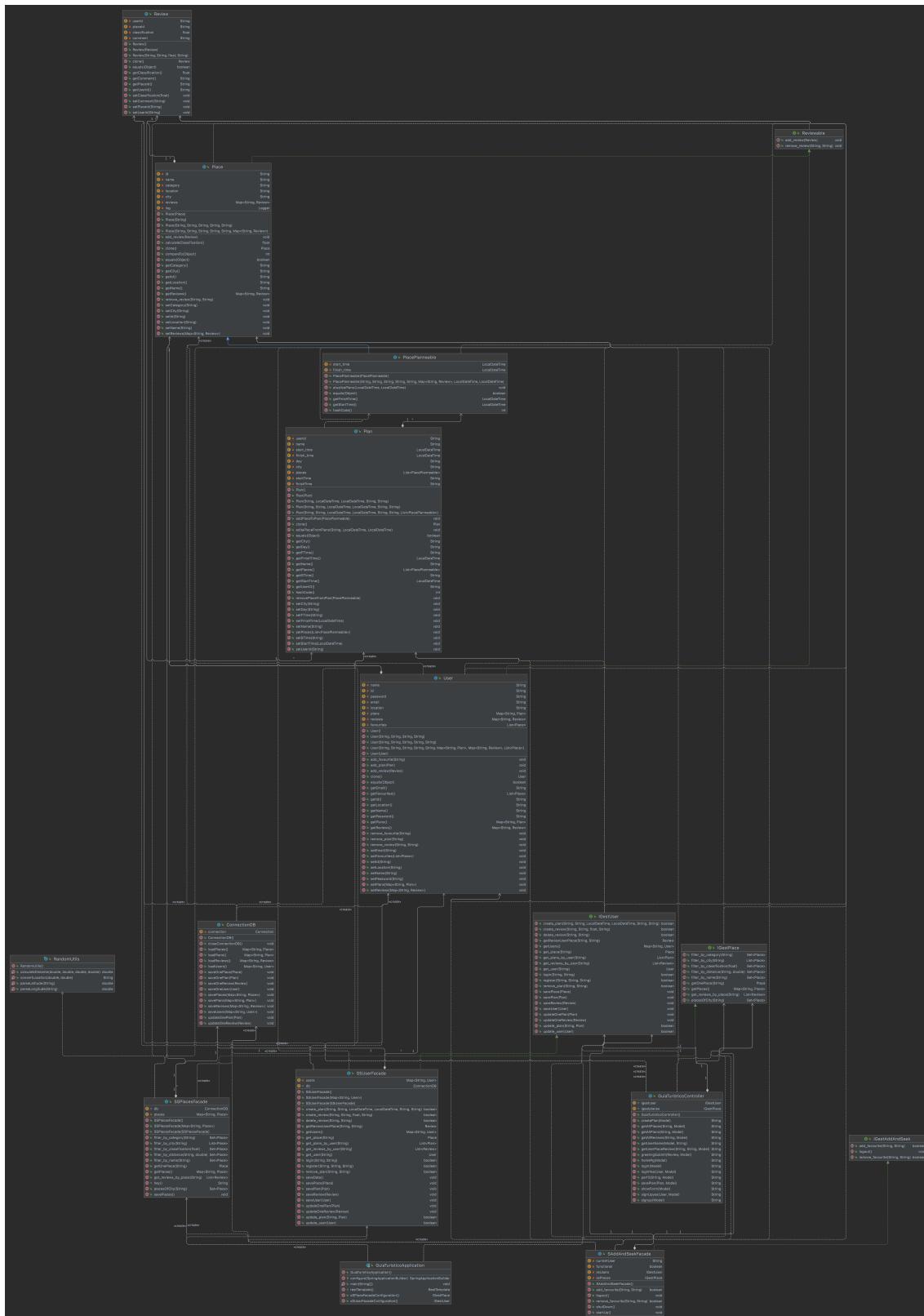


Figura 2.4: Diagrama de Classes usado pela equipa de desenvolvimento.

Por fim, a componente de *front-end* foi estruturada com base nas maquetas facultadas pela equipa de especificação e pela subsequente interação a ser estabelecida com o *back-end* – trabalho que se evidenciou como sendo uma das tarefas mais desafiantes do presente projeto. Para esta parte particular da plataforma, foram desenvolvidas um conjunto de páginas em HTML, com o correspondente CSS que possibilitou aproximar o *layout* final da plataforma daquele que tinha sido proposto pelos *mock-ups* fornecidos. O desenvolvimento desta componente assentou, adicionalmente, no uso da *framework Spring Boot*, que permitiu relacionar o código desenvolvido no *back-end* com o *front-end* criado. A Figura 2.5 apresenta o diagrama de dependências do *front-end* desenvolvido.

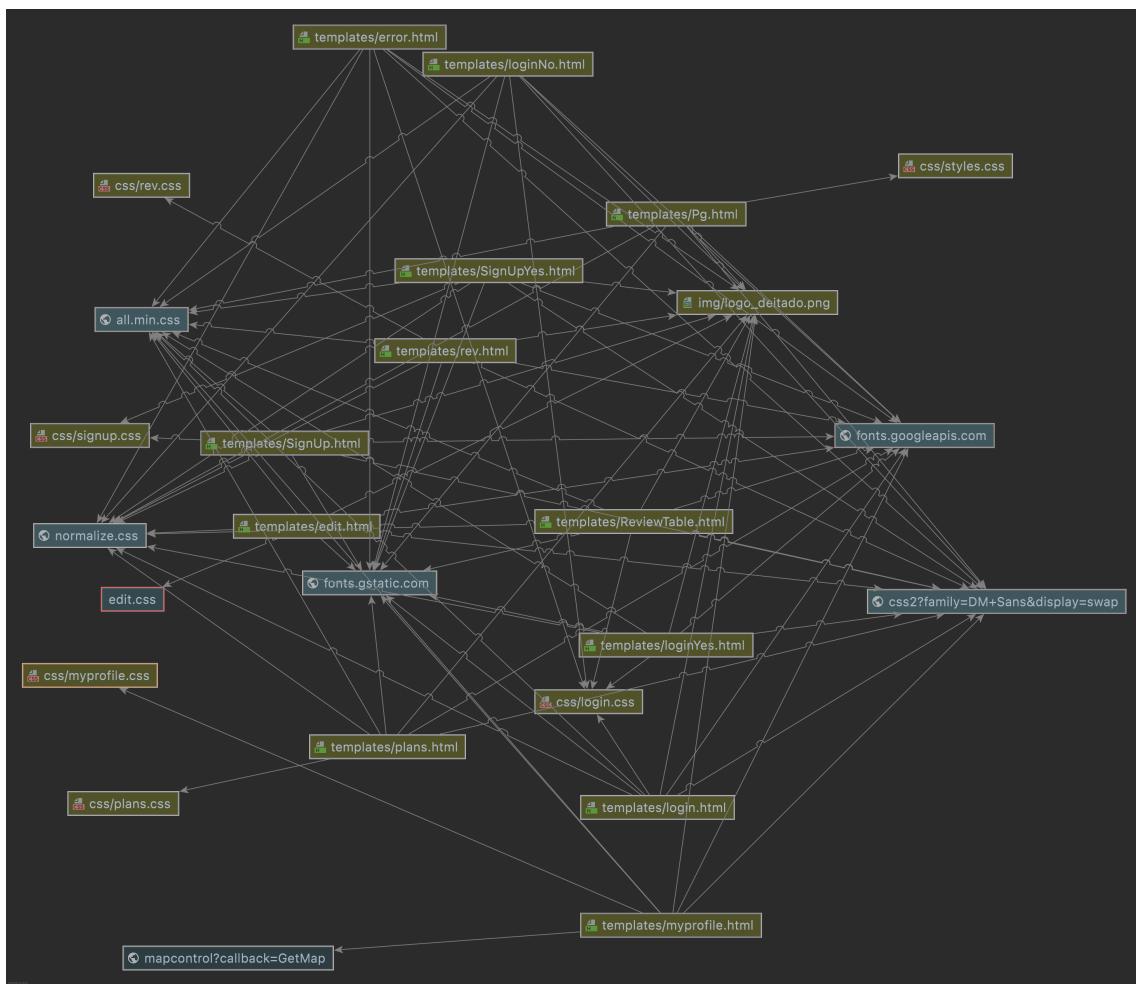


Figura 2.5: Diagrama de dependências do *front-end*.

2.4 Manual de utilização

A presente seção do relatório pretende acompanhar a navegação do utilizador na plataforma desenvolvida, descrevendo as funcionalidades disponibilizadas em cada página da mesma. Como tal, seguem-se um conjunto de figuras ilustrativas que demonstram os componentes do *front-end* já desenvolvidos.

Assim que entra no *website*, o usuário visualiza a página inicial, onde são explicitadas as três componentes da *add&SEEK*: *planning* (planear), *profile* (perfil), *search* (procurar). Nesta página, o utilizador apenas tem permissão para efetuar *login* ou *signup*.

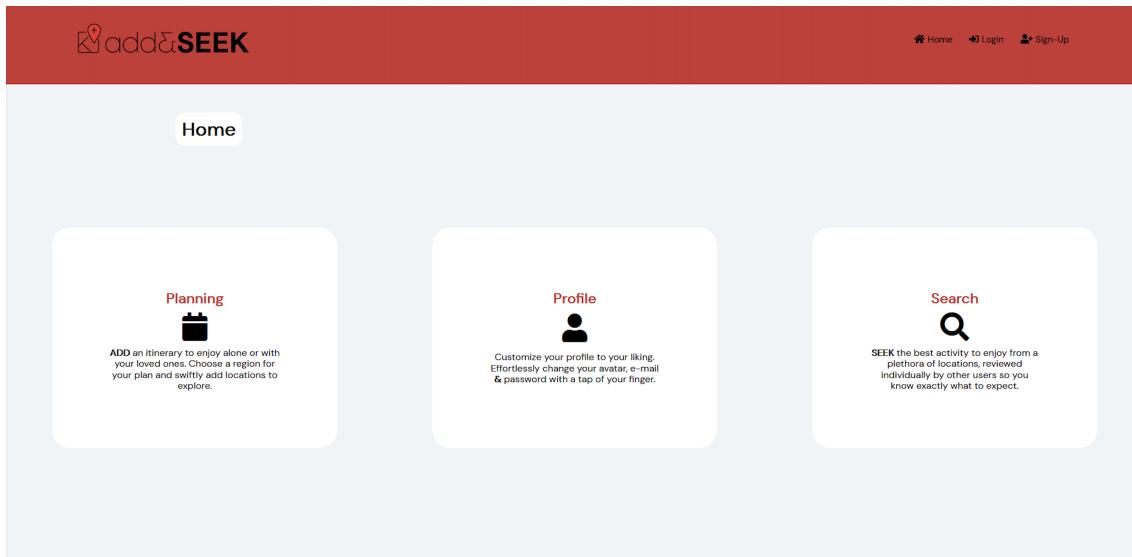


Figura 2.6: Página inicial.

Ao escolher a opção para efetuar *login*, o utilizador encontrará uma página, onde lhe serão pedidos o seu nome e a sua palavra-passe, e, após os dados introduzidos serem autenticados, ser-lhe-á notificado que o *login* foi efetuado com sucesso.

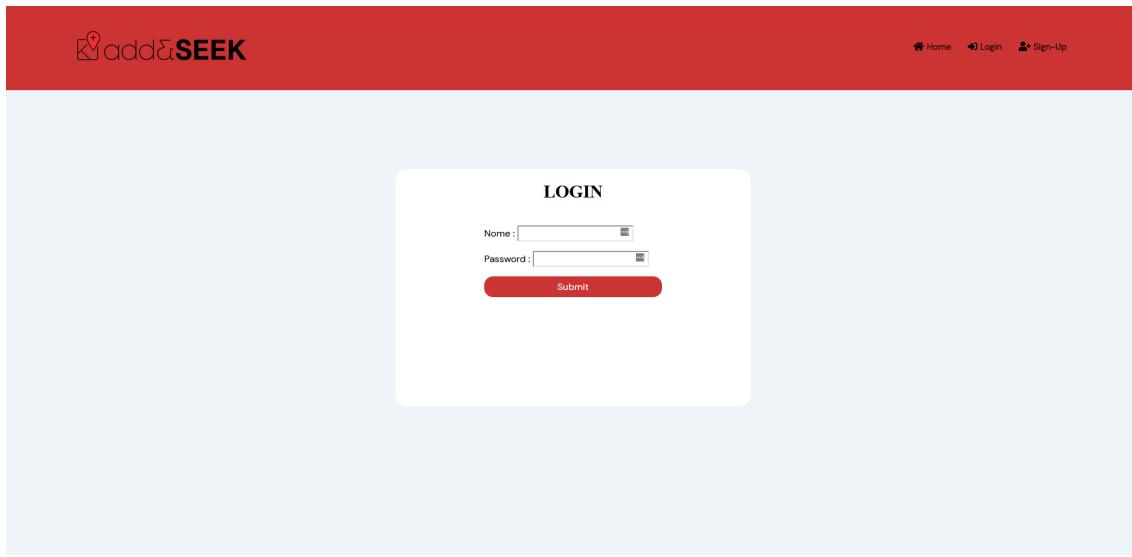


Figura 2.7: Página de *login*.



Figura 2.8: *Login* efetuado com sucesso.

O procedimento para realizar o *signup* é similar ao processo anteriormente descrito para o *login*: o utilizador introduz um nome, email e password e, no caso destes serem válidos, será informado de que se encontra registado no *website*.

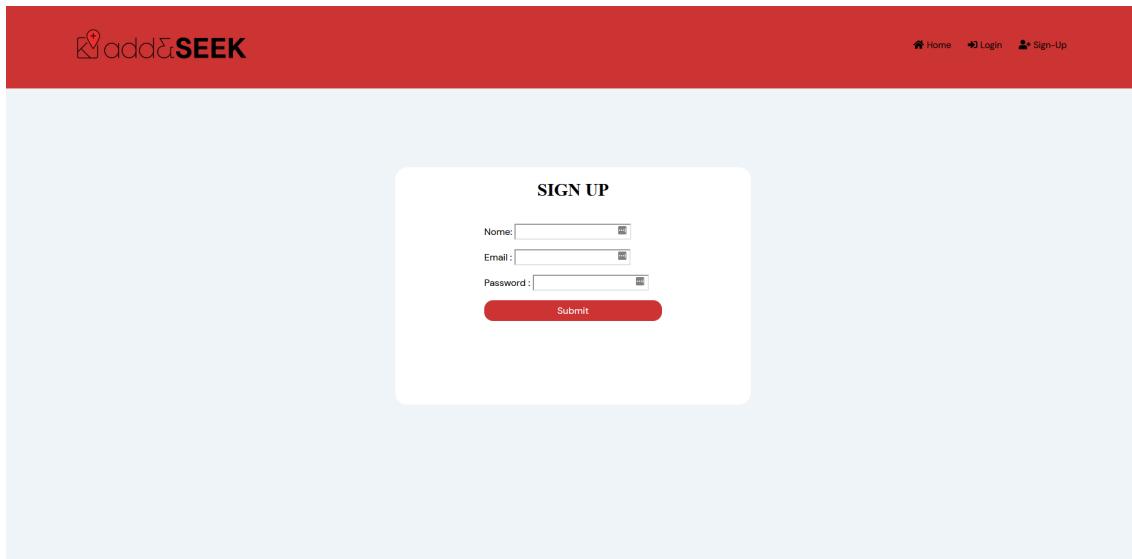


Figura 2.9: Página de *signup*.

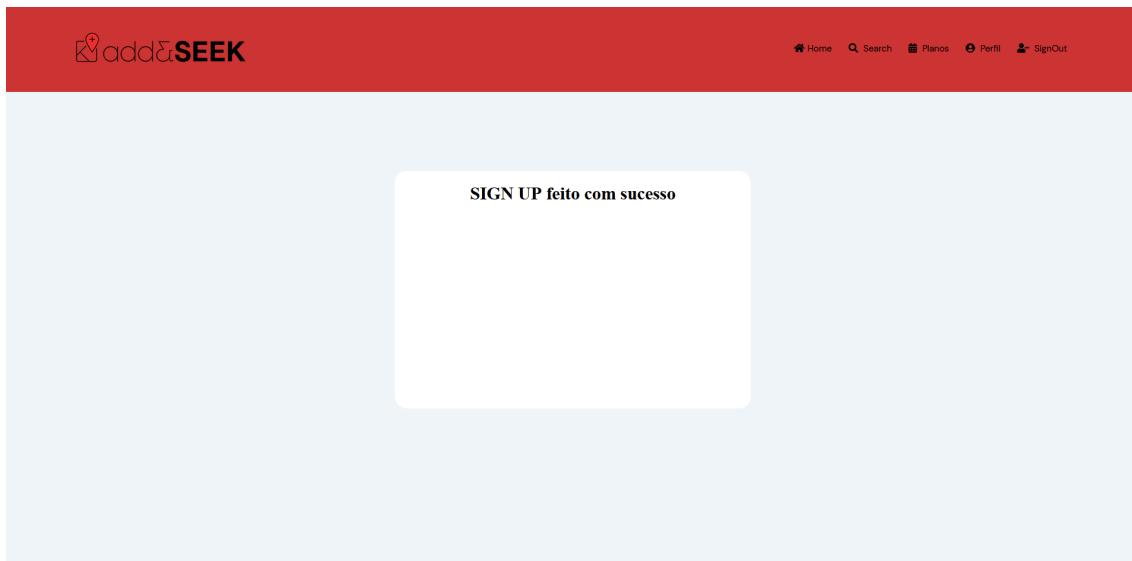


Figura 2.10: *Signup* efetuado com sucesso.

Após estar autenticado, o utilizador pode aceder à sua página de perfil, onde são mostrados todos os seus dados pessoais, como o seu email, que pode ser editado, os seus lugares favoritos e as suas avaliações. Para além disto, este ainda pode visualizar um mapa com a sua localização atual.

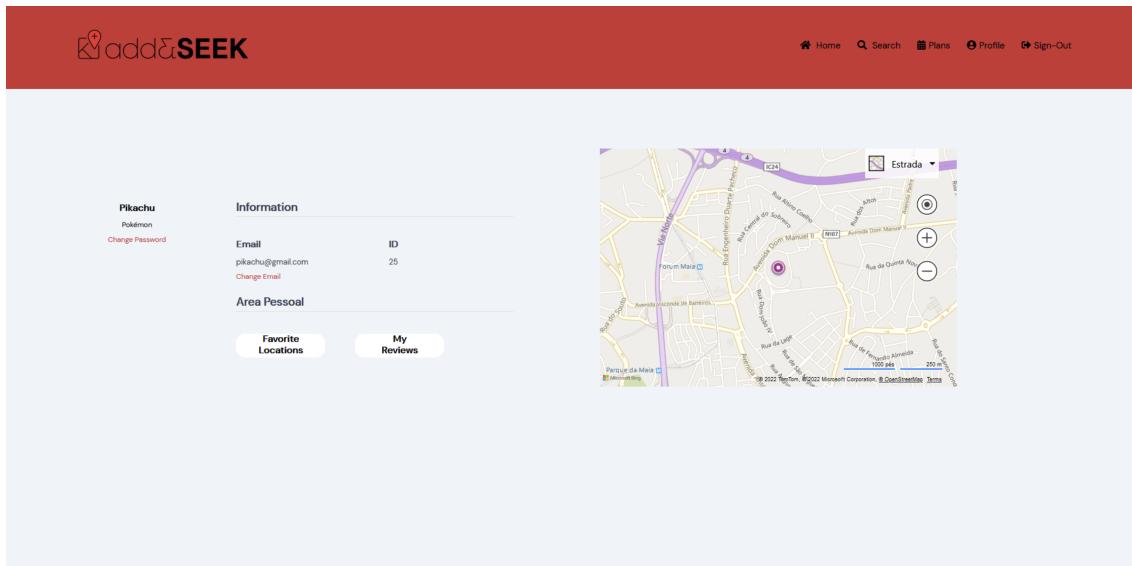


Figura 2.11: Página de perfil.

Outra funcionalidade a que o usuário tem acesso é a filtragem de lugares turísticos aplicando três tipos de critérios: distância ao local, rating e categoria. Deste modo, o mesmo pode pesquisar por um lugar em específico a partir dos critérios previamente mencionados.

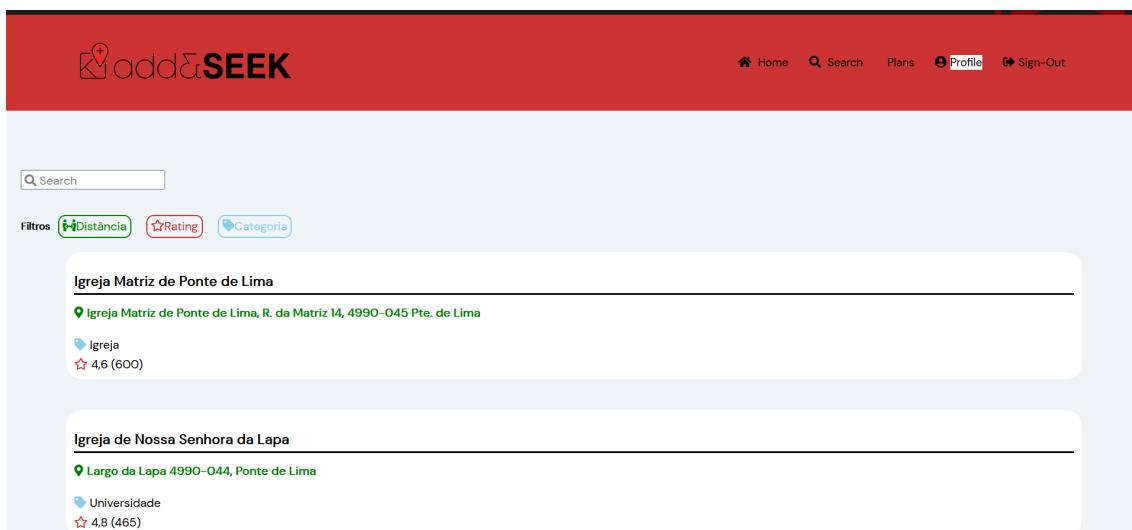


Figura 2.12: Filtragem de lugares turísticos através de filtros

3 Conclusões e Trabalho Futuro

O presente projeto centrou-se no desenvolvimento de um guia turístico para locais de interesse. O desenvolvimento do projeto integrou duas fases distintas, mas complementares: iniciando-se com a especificação das características e funcionalidades do sistema, seguindo-se pelo desenvolvimento do mesmo. O relatório e trabalho aqui apresentados refletem a segunda fase de desenvolvimento, que corresponde à implementação do sistema conceitualizado.

De um modo geral, a equipa de desenvolvimento procurou seguir de forma um tanto rígida as especificações delimitadas pela equipa de conceitualização da plataforma, o que em parte facilitou o trabalho pois possibilitou direcionar desde cedo o trajeto a percorrer aquando do desenvolvimento do sistema requerido. Por outro lado, devido ao teor pouco claro das especificações, foram encontrados alguns contratemplos e dificuldades acrescidas que, essencialmente, adviriam da falta de rigor e clareza provenientes de algumas especificações.

Tal como foi mencionado ao longo do presente relatório, a conceitualização da camada de dados revelou-se insuficiente para desenvolver algumas das funcionalidades pretendidas. Muita da informação necessária, e relevante, para ser usada nas diversas funcionalidades encontra-se omitida no sistema de dados. Adicionalmente, há informação que deveria poder ser relacionada de forma direta – como o caso dos locais de interesse e dos planos de viagem – para facilitar a construção de algumas funcionalidades. A título de exemplo, a ausência de informação relativa ao tempo necessário a disponibilizar para a visita dos locais tornou impossível a implementação de funcionalidades associadas com a edição dos planos de viagem – como o caso do controlo de horário a ser efetuado aquando da alteração do plano.

No entanto, a especificação recebida em termos da lógica de negócio permitiu que a equipa de desenvolvimento implementasse algumas funcionalidades, mesmo que a informação gerada por estas não persistisse na camada de dados. Em foco nesta camada, desenvolvida com recurso a JAVA, a equipa de desenvolvimento considera que cumpriu os objetivos propostos disponibilizando as operações essenciais para a implementação das funcionalidades referidas anteriormente.

O grande, e principal, desafio surgiu no desenvolvimento do *front-end* e com a sua integração com a camada de lógica de negócio. A utilização da framework *Spring Boot* demonstrou-se um enorme auxílio nesta tarefa, pois permitiu integrar, com recurso a JS, o código desenvolvido em JAVA com o código HTLM e CSS desenvolvido para a interface de interação com o utilizador. Porém, a utilização destas ferramentas de desenvolvimento representou um desafio na medida em que, até à data, nenhum elemento da equipa de desenvolvimento tinha as tinhado utilizado previamente. Tal acontecimento dificultou o a integração dos componentes desenvol-

vidos, resultando em que algumas funcionalidades estivessem prontas para ser implementadas, do ponto de vista do *back-end*, mas ficaram em falta a nível do *front-end*. Aditivamente, neste sentido, a equipa sente que alguns aspectos relativos ao *front-end* poderiam ser melhorados, com vista a obter uma plataforma mais atraente do ponto de vista visual e um pouco mais funcional.

Posto isto, a equipa de desenvolvimento sente que o projeto pode ser, futuramente, melhorado. Em primeira instância, seria imperativo melhorar o sistema de dados, construindo um sistema capaz de armazenar a informação necessária para responder completamente aos requisitos. Tal feito alteraria o modo como a camada de negócio interage com a camada de dados, evitando realizar um carregamento massivo dos dados necessários, podendo efectuar *queries* à base de dados, para obter a informação necessária no momento em que esta é relevante. Esta alteração permitiria libertar trabalho computacional do lado do servidor, movendo-o para a base de dados. Adicionalmente, poderiam ser adotado um paradigma de programação concorrente, com recurso a fios de execução, com vista a permitir que o servidor descentralize o trabalho computacional que tem de executar, podendo-o distribuir por outras máquinas diferentes. Adicionalmente, tal como foi já referido, a interface de interação com o utilizador pode, e deve, ser melhorada, para fornecer um *layout* mais atrativo.

4 Anexos

4.1 Criação da base de dados

```
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,
NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

-----
-- Schema GuiaTuristico
-----

-----
-- Schema GuiaTuristico
-----

CREATE SCHEMA IF NOT EXISTS `GuiaTuristico` DEFAULT CHARACTER SET utf8 ;
USE `GuiaTuristico` ;

-----
-- Table `GuiaTuristico`.`users`
-----

CREATE TABLE IF NOT EXISTS `GuiaTuristico`.`users` (
  `id` INT NOT NULL,
  `name` VARCHAR(45) NULL,
  `password` VARCHAR(45) NULL,
  `email` VARCHAR(45) NULL,
  `location` VARCHAR(45) NULL,
  PRIMARY KEY (`id`)
ENGINE = InnoDB;
```

```

-- -----
-- Table `GuiaTuristico`.`places` -
-- -----
CREATE TABLE IF NOT EXISTS `GuiaTuristico`.`places` (
  `id` INT NOT NULL,
  `name` VARCHAR(45) NULL,
  `placeId` VARCHAR(45) NULL,
  `category` VARCHAR(45) NULL,
  `location` VARCHAR(45) NULL,
  `city` VARCHAR(45) NULL,
  PRIMARY KEY (`id`)
)
ENGINE = InnoDB;

-- -----
-- Table `GuiaTuristico`.`reviews` -
-- -----
CREATE TABLE IF NOT EXISTS `GuiaTuristico`.`reviews` (
  `placeName` VARCHAR(45) NULL,
  `classification` VARCHAR(45) NULL,
  `text` VARCHAR(45) NULL,
  `users_id` INT NOT NULL,
  `places_id` INT NOT NULL,
  PRIMARY KEY (`users_id`, `places_id`),
  INDEX `fk_reviews_places1_idx` (`places_id` ASC) VISIBLE,
  CONSTRAINT `fk_reviews_users1`
    FOREIGN KEY (`users_id`)
    REFERENCES `GuiaTuristico`.`users` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_reviews_places1`
    FOREIGN KEY (`places_id`)
    REFERENCES `GuiaTuristico`.`places` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
)
ENGINE = InnoDB;

-- -----
-- Table `GuiaTuristico`.`plans` -
-- -----
CREATE TABLE IF NOT EXISTS `GuiaTuristico`.`plans` (
  `name` VARCHAR(45) NULL,
  `startTime` DATETIME NULL,

```

```

`finishTime` DATETIME NULL,
`city` VARCHAR(45) NULL,
`users_id` INT NOT NULL,
PRIMARY KEY (`users_id`),
CONSTRAINT `fk_plans_users`
    FOREIGN KEY (`users_id`)
    REFERENCES `GuiaTuristico`.`users` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

USE `GuiaTuristico` ;

-----
-- Placeholder table for view `GuiaTuristico`.`view1`
-----

CREATE TABLE IF NOT EXISTS `GuiaTuristico`.`view1` (`id` INT);

-----
-- View `GuiaTuristico`.`view1`
-----

DROP TABLE IF EXISTS `GuiaTuristico`.`view1`;
USE `GuiaTuristico`;

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;

```

4.2 Povoamento da base de dados

```

/* TABELA: places */
INSERT INTO GuiaTuristico.places (id, name, placeID, category, location, city)
VALUES (1, 'Igreja do Carmo', '1', 'Monumento',
'41.554186849999994 -8.425564314112549', 'Braga');
INSERT INTO GuiaTuristico.places (id, name, placeID, category, location, city)
VALUES (2, 'Bom Jesus', '2', 'Monumento',
'41.55477585 -8.376944908792922', 'Braga');
INSERT INTO GuiaTuristico.places (id, name, placeID, category, location, city)
VALUES (3, 'Sé Catedral de Braga', '3', 'Monumento',
'41.55000115 -8.426994750163198', 'Braga');
INSERT INTO GuiaTuristico.places (id, name, placeID, category, location, city)

```

```

VALUES (4, 'Igreja dos Terceiros', '4', 'Monumento',
'41.55167805 -8.424466941185162', 'Braga');
INSERT INTO GuiaTuristico.places (id, name, placeID, category, location, city)
VALUES (5, 'Parque da Ponte', '5', 'Parque',
'41.54201245 -8.4187894626297', 'Braga');
INSERT INTO GuiaTuristico.places (id, name, placeID, category, location, city)
VALUES (6, 'Arco da Porta Nova', '6', 'Monumento',
'41.5502344 -8.429343', 'Braga');
INSERT INTO GuiaTuristico.places (id, name, placeID, category, location, city)
VALUES (7, 'Campo da Rodovia', '7', 'Parque',
'41.5521128 -8.405527492111963', 'Braga');
INSERT INTO GuiaTuristico.places (id, name, placeID, category, location, city)
VALUES (8, 'Jardim de Santa Bárbara', '8', 'Parque',
'41.551389099999994 -8.425902995873047', 'Braga');
INSERT INTO GuiaTuristico.places (id, name, placeID, category, location, city)
VALUES (9, 'Museu dos Biscaínhos', '9', 'Museu',
'41.55119075 -8.42973893705529', 'Braga');

-- (...)

INSERT INTO GuiaTuristico.places (id, name, placeID, category, location, city)
VALUES (25, 'Naco na Pedra', '25', 'Restaurante',
'41.5602697 -8.4060323', 'Braga');
INSERT INTO GuiaTuristico.places (id, name, placeID, category, location, city)
VALUES (26, 'Museu Nogueira da Silva', '26', 'Museu',
'41.55261555 -8.421848558081951', 'Braga');
INSERT INTO GuiaTuristico.places (id, name, placeID, category, location, city)
VALUES (27, 'Museu de Arquiologia Dom Diogo de Sousa', '27', 'Museu',
'41.5454942 -8.4275215', 'Braga');
INSERT INTO GuiaTuristico.places (id, name, placeID, category, location, city)
VALUES (28, 'Museu da Imagem', '28', 'Museu',
'41.5502808 -8.4294229', 'Braga');
INSERT INTO GuiaTuristico.places (id, name, placeID, category, location, city)
VALUES (29, 'Pastelaria Desejos', '29', 'Pastelaria',
'41.5052831 -8.4351611', 'Braga');
INSERT INTO GuiaTuristico.places (id, name, placeID, category, location, city)
VALUES (30, 'Palácio do Raio', '30', 'Monumento',
'41.5483181 -8.422714080097997', 'Braga');

/* TABELA: users */
INSERT INTO GuiaTuristico.users (id, name, password, email, location)
VALUES (1, 'Rui', 'user', 'rui@fakemail.com', 'Braga');
INSERT INTO GuiaTuristico.users (id, name, password, email, location)
VALUES (2, 'Pedro', 'user', 'pedro@fakemail.com', 'Braga');

```

```

INSERT INTO GuiaTuristico.users (id, name, password, email, location)
    VALUES (3, 'Paula', 'user', 'paula@fakemail.com', 'Braga');
INSERT INTO GuiaTuristico.users (id, name, password, email, location)
    VALUES (4, 'Filipa', 'user', 'filipa@fakemail.com', 'Braga');
INSERT INTO GuiaTuristico.users (id, name, password, email, location)
    VALUES (5, 'Leonardo', 'user', 'leonardo@fakemail.com', 'Braga');

-- (...)

INSERT INTO GuiaTuristico.users (id, name, password, email, location)
    VALUES (9, 'Fernando', 'user', 'fernando@fakemail.com', 'Braga');
INSERT INTO GuiaTuristico.users (id, name, password, email, location)
    VALUES (10, 'Teresa', 'user', 'teresa@fakemail.com', 'Braga');

/* TABELA: plans */
INSERT INTO GuiaTuristico.plans (name, startTime, finishTime, city, users_id)
    VALUES ('Visita Igreja','2022-01-20 13:23:44', '2022-01-20 14:20:44', 'Braga' , 1);
INSERT INTO GuiaTuristico.plans (name, startTime, finishTime, city, users_id)
    VALUES ('Visita Parques','2022-01-19 13:23:44', '2022-01-20 14:20:44', 'Braga' , 2);
INSERT INTO GuiaTuristico.plans (name, startTime, finishTime, city, users_id)
    VALUES ('Visita Igreja','2022-01-15 10:23:44', '2022-01-16 15:20:44', 'Braga' , 3);
-- (...)

/* TABELA: reviews */
INSERT INTO GuiaTuristico.reviews (placeName, classification, text, users_id, places_id)
    VALUES ('Igreja do Carmo', '3.5', 'Muito bonito.', 1, 1);
INSERT INTO GuiaTuristico.reviews (placeName, classification, text, users_id, places_id)
    VALUES ('Parque da Ponte', '4.3', 'Bastante verde.', 2, 5);
INSERT INTO GuiaTuristico.reviews (placeName, classification, text, users_id, places_id)
    VALUES ('Forno Mágico', '3', 'Comida boa.', 1, 21);
INSERT INTO GuiaTuristico.reviews (placeName, classification, text, users_id, places_id)
    VALUES ('Pastelaria Desejos', '2', 'Razoável.', 2, 29);

```