

# **Universidade da Beira Interior**

## **Departamento de Informática**



**Departamento de  
Informática**

### ***Casas Para Passar Férias***

Elaborado por:

**António José Santos Sardinha – Nº 39477**

**Pedro Jorge Franco Brito – Nº 39490**

**Ricardo André Pereira Ribeiro – Nº 39529**

**Rui Miguel Monteiro Raposo – Nº 39986**

Professores:

**Professora Doutora Maria Paula Prata de Sousa**

**Professor Doutor Luís Pedro Arrojado da Horta**

12 de Junho de 2020

# Conteúdo

|   |            |
|---|------------|
| <b>Conteúdo</b>   | <b>i</b>   |
| <b>Lista de Figuras</b>                                 | <b>iii</b> |
| <b>Lista de Tabelas</b>                                 | <b>iv</b>  |
| <b>1 Introdução</b>                                     | <b>1</b>   |
| 1.1 Enquadramento . . . . .                             | 1          |
| 1.2 Objetivos e Planificação . . . . .                  | 1          |
| 1.3 Organização do Documento . . . . .                  | 2          |
| <b>2 Engenharia de Software</b>                         | <b>4</b>   |
| 2.1 Introdução . . . . .                                | 4          |
| 2.2 Modelação da Base de Dados . . . . .                | 4          |
| 2.2.1 Modelo Concetual . . . . .                        | 5          |
| 2.2.2 Modelo Físico . . . . .                           | 5          |
| 2.3 Arquitetura do Sistema . . . . .                    | 6          |
| 2.4 Conclusões . . . . .                                | 6          |
| <b>3 Implementação</b>                                  | <b>7</b>   |
| 3.1 Introdução . . . . .                                | 7          |
| 3.2 Decisões de Implementação . . . . .                 | 7          |
| 3.2.1 Beans . . . . .                                   | 7          |
| 3.2.2 Exportar projeto do <i>NetBeans</i> . . . . .     | 8          |
| 3.3 Detalhes de Implementação . . . . .                 | 9          |
| 3.3.1 Autenticação – <i>intro.xhtml</i> . . . . .       | 9          |
| 3.3.1.1 Restart data . . . . .                          | 9          |
| 3.3.1.2 Guest e Users . . . . .                         | 10         |
| 3.3.2 Home – <i>home.xhtml</i> . . . . .                | 11         |
| 3.3.2.1 Desenhar Gráfico . . . . .                      | 11         |
| 3.3.2.2 Apresentação das casas ao <i>user</i> . . . . . | 12         |
| 3.3.2.3 Request de uma casa . . . . .                   | 13         |
| 3.3.3 Perfil – <i>profile.xhtml</i> . . . . .           | 13         |

---

|          |   |           |
|----------|---|-----------|
| 3.3.3.1  | Verificação inicial . . . . .                   | 14        |
| 3.3.3.2  | Mudança de <i>password</i> . . . . .            | 14        |
| 3.3.3.3  | Apresentação das casas ao <i>user</i> . . . . . | 14        |
| 3.3.3.4  | Entregar casa . . . . .                         | 15        |
| 3.3.3.5  | Adicionar casa . . . . .                        | 15        |
| 3.3.4    | Conclusões . . . . .                            | 16        |
| <b>4</b> | <b>Manuais</b>                                  | <b>17</b> |
| 4.1      | Manual de instalação . . . . .                  | 17        |
| 4.2      | Manual de Utilização . . . . .                  | 18        |
| 4.2.1    | <i>Sign In</i> . . . . .                        | 18        |
| 4.2.2    | Inserir uma nova casa . . . . .                 | 18        |
| 4.2.3    | Visualizar casas . . . . .                      | 19        |
| 4.2.4    | Casa requisitada . . . . .                      | 19        |
| 4.2.5    | Casa em espera . . . . .                        | 20        |
| 4.2.6    | <i>Rating</i> da casa . . . . .                 | 20        |
| 4.2.7    | Estatísticas de casa . . . . .                  | 21        |
| 4.2.8    | Múltiplas casas . . . . .                       | 21        |
| 4.2.9    | Estatísticas de várias casas . . . . .          | 22        |
|          | <b>Bibliografia</b>                             | <b>23</b> |

## ***Lista de Figuras***

|     |  |    |
|-----|--|----|
| 2.1 | Modelo entidade relacionamento da base de dados. . . . .                             | 5  |
| 2.2 | Modelo físico da base de dados. . . . .  | 5  |
| 2.3 | Diagrama de casos de uso. . . . .  | 6  |
| 3.1 | Dados a inserir para fazer o encaminhamento de portos. . . . .                       | 8  |
| 3.2 | Dados a inserir para fazer a ativação de portos. . . . .                             | 9  |
| 3.3 | Diagrama da gestão de variáveis quando apresentada a página de autenticação. . . . . | 10 |
| 3.4 | Diagrama da gestão de variáveis aquando da autenticação. . . . .                     | 11 |
| 3.5 | Exemplo do mapeamento das casas. . . . .   | 13 |
| 3.6 | Diagrama demonstrativo da verificação inicial aquando na página de perfil. . . . .   | 14 |
| 3.7 | Diagrama para formatação do botão associado à casa no perfil. . . . .                | 15 |
| 4.1 | Página de autenticação. . . . .  | 18 |
| 4.2 | Página de inserir casas . . . . .  | 18 |
| 4.3 | Página onde se dá <i>display</i> as casas . . . . .                                  | 19 |
| 4.4 | Página de casas requisitadas . . . . .   | 19 |
| 4.5 | Quando a casa não está disponível para requisitar imediatamente. . . . .             | 20 |
| 4.6 | <i>popup</i> . . . . .   | 20 |
| 4.7 | Página do gráfico das estatísticas de uma só casa. . . . .                           | 21 |
| 4.8 | <i>Display</i> de várias casas. . . . .  | 21 |
| 4.9 | Página do gráfico das estatísticas de várias casas. . . . .                          | 22 |

## ***Lista de Tabelas***

|     |   |   |
|-----|---|---|
| 1.1 | Planificação das diversas tarefas deste projeto. . . . .            | 2 |
| 3.1 | <i>Beans</i> e os respetivos tipos utilizados na aplicação. . . . . | 8 |

# ***Acrónimos***

**TLS** *Transport Layer Security*

**IDE** *Integrated Development Environment*

## Capítulo

# 1

## Introdução

Este capítulo está repartido em 3 secções que descrevem o **enquadramento** do projeto, os **objetivos** a serem atingidos, a sua **planificação** e, por fim, a **organização** deste documento no seu total.

### 1.1 Enquadramento

Numa época em que muitas pessoas procuram viajar e passar férias, surge cada vez mais o problema de encontrar alojamento rápido e de confiança. Muitas das vezes, os turistas acabam por não saber qual a melhor maneira de conseguirem arranjar alojamento, quer por falta de informação, quer por falta de plataformas que facilitem este processo ao turista.

Tendo em conta estes problemas antes descritos, este projeto pretende fornecer aos turistas uma plataforma *web* em que tenham fácil acesso a uma panóplia de casas de férias que podem facilmente requisitar ou consultar.

Este projeto foi proposto no âmbito da unidade curricular sistemas distribuídos. Para além de abordar a área de sistemas distribuídos, enquadra-se também nas áreas de redes de computadores e segurança informática.

### 1.2 Objetivos e Planificação

Este projeto tem foco na criação de um sistema *web based* dedicado à gerência de casas para passar férias. Este sistema "*deverá usar uma base de dados para persistir a informação, ter uma interface web em Java Server Faces, usando "entities" e "enterprise java beans" [1]. Neste sistema devem haver 2 tipos de utilizadores sendo eles o *normal user* e o *guest*, em que este último apenas poderá consultar a informação disponível no *web site*.*

Para atingir todos os objetivos propostos foi implementada a seguinte planificação:

- **Tarefa 1** – Contextualização dos objetivos propostos e preparação do ambiente de trabalho;
- **Tarefa 2** – Definição de todas as ferramentas e tecnologias a serem utilizadas para a elaboração do projeto;
- **Tarefa 3** – Implementação do *Backend*;
- **Tarefa 4** – Implementação do *Frontend*;
- **Tarefa 5** – Análise e Teste do sistema;
- **Tarefa 6** – Escrita do relatório.

A distribuição de todas estas tarefas, no que diz respeito à questão temporal, encontra-se devidamente esquematizada na figura abaixo.

| Tarefa | Semana 1 | Semana 2 | Semana 3 | Semana 4 |
|--------|----------|----------|----------|----------|
| 1      | x        |          |          |          |
| 2      | x        |          |          |          |
| 3      | x        | x        | x        |          |
| 4      |          | x        | x        |          |
| 5      | x        | x        | x        | x        |

Tabela 1.1: Planificação das diversas tarefas deste projeto.

## 1.3 Organização do Documento

De modo a refletir o trabalho que foi feito, este documento encontra-se estruturado da seguinte forma:

1. O primeiro capítulo – **Introdução** – Apresentação do projeto e enquadramento do mesmo com uma apresentação dos seus objetivos e a respetiva organização do documento.
2. O segundo capítulo – **Engenharia de Software** – Descrição dos conceitos mais importantes no âmbito deste projeto.
3. O terceiro capítulo – **Implementação** – Descrição detalhada de todos os métodos de implementação que usámos no desenvolvimento do projeto.



4. O quarto capítulo – **Manuais** – Demonstração do funcionamento da aplicação.

## **Capítulo**

# 2

## ***Engenharia de Software***

### **2.1 Introdução**

Este capítulo descreve a engenharia de *software* da aplicação *web*. Existe uma tripartição deste capítulo em que, numa primeira fase, será apresentado o modelo da base de dados, em segundo plano a arquitetura do sistema e, por último, as conclusões.

### **2.2 Modelação da Base de Dados**

Os modelos de base de dados têm como principais focos a esquematização e a organização dos dados de uma aplicação permitindo uma melhor compreensão sobre a mesma. Nesta subsecção serão apresentados os modelos de entidade relacionamento da base de dados e o modelo físico da mesma.

2.2.1 Modelo Concetual

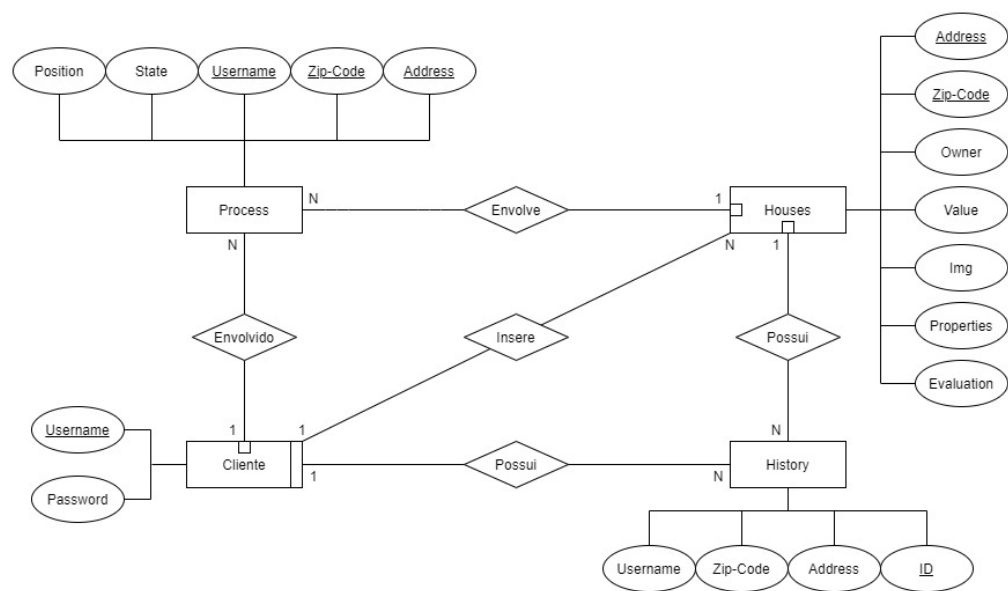


Figura 2.1: Modelo entidade relacionamento da base de dados.

2.2.2 Modelo Físico

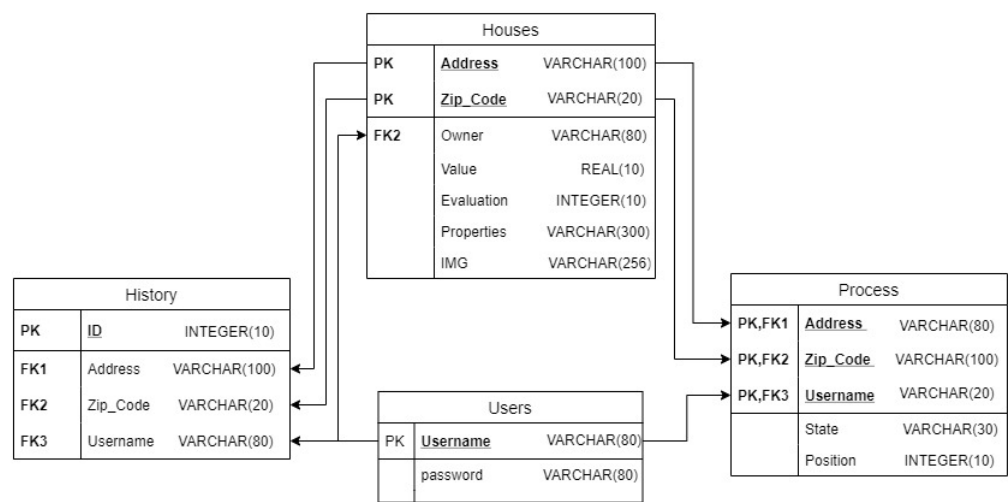


Figura 2.2: Modelo físico da base de dados.

## 2.3 Arquitetura do Sistema

A arquitetura de um sistema consiste no esqueleto do sistema em termos do posicionamento das suas componentes, do papel que cada uma ocupa dentro do mesmo e a maneira como todas essas se interrelacionam.

De seguida, será apresentada a arquitetura do sistema desenvolvido neste projeto.

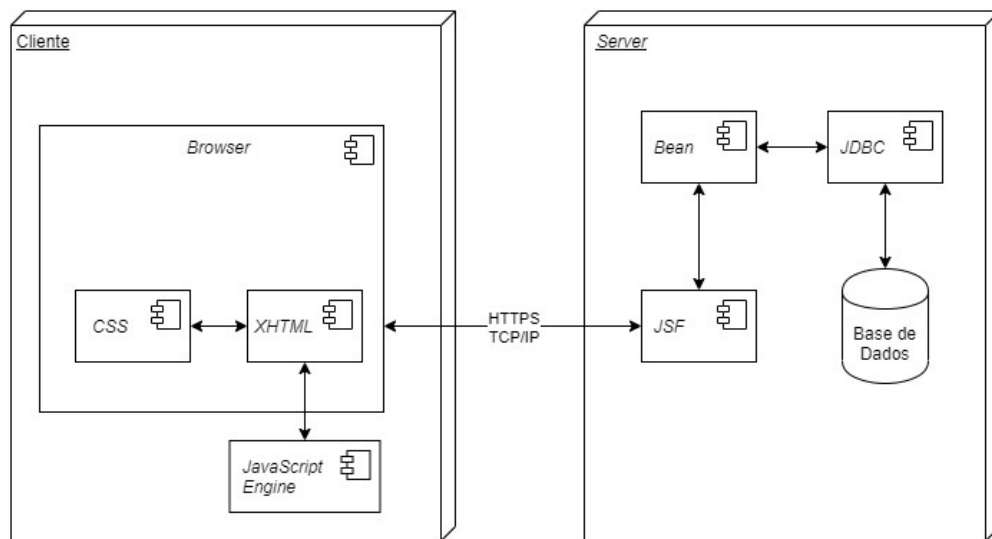


Figura 2.3: Diagrama de casos de uso.

## 2.4 Conclusões

Neste capítulo encontramos não só a modelação da base de dados, mas também a arquitetura do sistema que foram implementadas no desenvolvimento deste projeto. Tendo estes esquemas em consideração podemos compreender melhor a forma de como o sistema se comporta e os seus objetivos.

## Capítulo

# 3

## Implementação

### 3.1 Introdução

Neste capítulo descrevemos as decisões de implementação que tomámos consoante os modelos de dados e arquitetura do sistema descritos no Capítulo 2.

### 3.2 Decisões de Implementação

#### 3.2.1 Beans

No desenvolvimento do nosso projeto, usámos *Stateless Beans* e *Stateful Session Beans*.

Os *Stateless Beans* são *beans* que apenas executam operações automaticamente e não mantêm um estado entre chamadas a ele mesmo. Na necessidade de o *bean* ter de manter um estado do objeto do utilizador entre chamadas do método, então nesse caso usam-se os *Stateful Beans*. No contexto da nossa aplicação, usámos este tipo de *bean* para gerar o gráfico das estatísticas do *rating* das casas, sendo que não precisamos de guardar informação nenhuma durante chamadas aos métodos, este tipo de *bean* serve perfeitamente para esta funcionalidade.

Por outro lado, os *Stateful Session Beans* são *beans* que mantêm os seus dados ao atualizar as suas variáveis sempre que uma chamada ao *bean* ocorre. Na nossa aplicação, este tipo de *beans* foram os mais necessários porque na maior parte dos casos é necessário guardar informação entre chamadas aos métodos.

Na tabela 3.1 temos uma lista de todos os *beans* que implementámos, juntamente com o seu tipo.

| <i>Bean</i>            | <i>Stateless</i> | <i>Stateful</i> |
|------------------------|------------------|-----------------|
| <i>User</i>            |                  | x               |
| <i>Home_Service</i>    |                  | x               |
| <i>Profile_Service</i> |                  | x               |
| <i>Service_Stuff</i>   | x                |                 |

Tabela 3.1: *Beans* e os respetivos tipos utilizados na aplicação.

### 3.2.2 Exportar projeto do *NetBeans*

Na fase final do desenvolvimento do projeto, exportámos o projeto do *netbeans* de forma a tornar o site acessível para toda a gente. Em baixo descrevemos todos os passos que seguimos para pormos o nosso *website online*.

É de notar que vamos discutir 2 números de *ports* diferentes, o *port 443*, que é usado para proteger a segurança das comunicações do *web browser* e também o *port 8181* devido ao facto de termos implementado no nosso servidor *Glassfish* conexões Transport Layer Security (*TLS*).

Agora iremos descrever todas as configurações que fizemos na página do *router*. **É de salientar que estas configurações foram feitas num *router* da *meo*.**

1. Acedemos à página do *router* pelo *ip* 192.168.1.254;
2. Introduzir *meo* no *username* e *meo* no *password*;
3. Entrar na página de Segurança;
4. Entrar na página de Acesso;
5. Criar uma nova regra de encaminhamento de portos;

The screenshot shows a 'Create port forwarding rule' window. The 'Interface' is set to 'veip0.1'. The 'Service name' is set to 'Custom'. The 'Server IP address' is set to 'LOCAL IPV4', which is highlighted with a red box and has an 'Invalid IP address' error message below it. The 'External ports' are set to '443' and '443'. The 'Protocol' is set to 'TCP'. The 'Internal ports' are set to '8181' and '8181'. There are 'create' and 'cancel' buttons at the bottom right.

Figura 3.1: Dados a inserir para fazer o encaminhamento de portos.

6. Criar uma nova regra de ativação de portos.

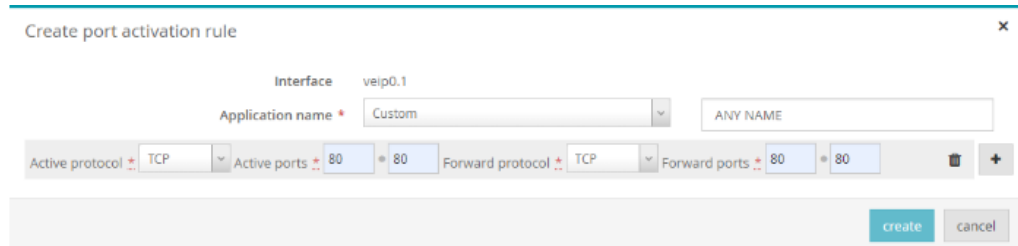


Figura 3.2: Dados a inserir para fazer a ativação de portos.

No final da implementação destas configurações, o nosso *website* encontra-se *online* e disponível para todos acederem a partir do endereço [https://DOMAIN\\_IPV4/HouseManager\\_2/faces/pages/intro.xhtml](https://DOMAIN_IPV4/HouseManager_2/faces/pages/intro.xhtml). Onde o *DOMAIN\_IPV4* é o endereço do router onde se fizeram estas configurações.

## 3.3 Detalhes de Implementação

### 3.3.1 Autenticação – *intro.xhtml*

Aquando na entrada da nossa aplicação, o *user* poderá efetuar a autenticação com recurso à sua conta e, caso ainda não possua nenhuma associada a ele mesmo, poderá sempre carregar no *link* de registo e, consequentemente, será redirecionado para uma página onde poderá proceder ao mesmo, ou poderá autenticar-se como um *user guest*.

Assim que o *user* introduza os dados da sua conta, ou não, e pressione o botão de *sign in*, é invocado o método *signInFunction* no *bean User* cujo vai conectar-se à base de dados para efetuar a verificação dos dados introduzidos. Caso estes sejam válidos, a autenticação é efetuada com sucesso e o *user* será redirecionado para a página seguinte, *home.xhtml*, caso não sejam, o *user* permanecerá na página atual.

Todos estes processos em volta da autenticação do *user* na aplicação estão detalhados nas subsecções listadas abaixo.

#### 3.3.1.1 *Restart data*

O método *restart data* é invocado assim que a página de autenticação é apresentada, desta forma, as variáveis *username* e *password* no *bean User*, associadas ao *user*, tomam o valor de uma *string* de tamanho igual a 0, "vazia", de modo a que, quando o *user* faz o *logout*, inicie a aplicação ou retroceda

na página (de tal forma a que seja redirecionado para a página em questão), os seus dados não sejam associados a nenhum *user* listado na base de dados. Este processo pode ser visualizado com maior detalhe na figura apresentada de seguida, 3.3.

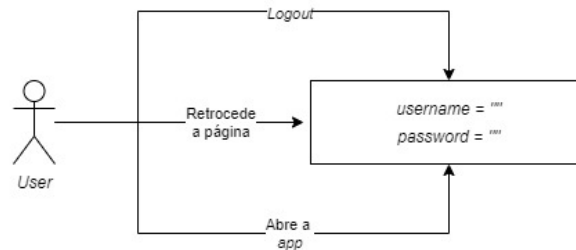


Figura 3.3: Diagrama da gestão de variáveis quando apresentada a página de autenticação.

#### 3.3.1.2 Guest e Users

Como foi referido na subsecção 3.3.1, o *user* poderá visitar a aplicação de duas maneiras, por um lado tem a possibilidade de efetuar a autenticação, tendo sido efetuado o registo previamente, introduzindo o seu *username* e a respetiva *password*, por outro lado, caso o *user* prefira não se autenticar com a sua conta, caso a possua, poderá entrar em modo *guest* o que, limitará o número de funcionalidades cujo possa usufruir, abrangendo-se apenas pela consulta geral de todas as casas.

Para controlar a maneira de como o *user* entra na aplicação, temos duas variáveis no *bean User* responsáveis para este processo, o *username* e a *password*, que conforme a opção escolhida pelo *user* para acessar o *web site*, os seus valores encaixarão em dois casos; o primeiro assenta nos valores introduzidos aquando da autenticação e, num segundo caso os valores associadas a ambas as variáveis de classe serão uma *string* de tamanho 0, "vazia". Estes dois casos podem ser observados na próxima figura, 3.7.



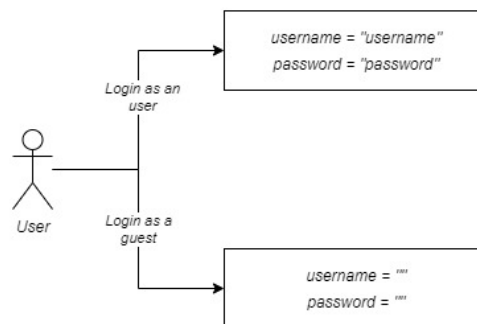


Figura 3.4: Diagrama da gestão de variáveis aquando da autenticação.

### 3.3.2 Home – *home.xhtml*

Após a realização da autenticação, o *user* depara-se com uma imagem alusiva à temática, podendo encontrar no topo, uma *navbar*, que permite uma melhor navegação pela aplicação. Nesta, podemos encontrar ligações à *Home*, *Houses*, *About*, *Team*, *FAQ*, *Profile* e *Logout*, se tiver efetuado a autenticação, caso contrário irá deparar-se com uma ligação para o *Sign In* e uma para o *Register Now*.

De forma a tornar a nossa aplicação intuitiva e profissional, acrescentámos uma secção, nesta página, dedicada à estatística. Um gráfico circular que referencia a frequência relativa percentual do número de casas devidamente agrupadas pelo número de estrelas atribuídas, pode encontrar a informação respetiva à atribuição do número de estrelas a uma casa na subsecção 3.3.2.1.

Para esclarecer os *users*, sobre o trabalho que foi desenvolvido, foi adicionado uma secção que responde a esta questão, o *About*, onde ainda se realçam algumas das funcionalidades que os *users* tem acesso, nesta aplicação.

Foi implementada um secção onde se faz uma breve menção aos elementos da equipa por trás da elaboração deste projeto, apresentando os nomes e *emails*, para possível contacto.

Por fim, foi adicionada uma secção onde são apresentadas as questões mais frequentes bem como as suas respostas.

#### 3.3.2.1 Desenhar Gráfico

Como referido anteriormente, o gráfico representa a frequência relativa percentual do número de casas devidamente agrupadas pelo número de estrelas atribuídas.

O gráfico é gerado quando um *user* acessa esta página, independentemente de ser um *user* autenticado ou um mero *guest*, recorrendo à biblioteca

*Canvas.js* do *JavaScript*. Deste modo, é evidente que tem de existir uma ligação entre a linguagem *Java* (*server side*) e o *JavaScript* (*client side*) e, esta está visível na *tag h:body* presente no ficheiro *home.xhtml*. É apenas necessária uma única linha de código em que é invocada uma função *test* do *JavaScript* em que é passado como parâmetro o resultado de um método, *setGraphic*, associado ao único *stateless bean*.

Esse resultado do método será uma *string* que, irá conter os resultados associados a cada estrela separada por vírgulas, o excerto de código ilustrado abaixo demonstra um exemplo dessa *string*.

|   |      |       |       |       |       |
|---|------|-------|-------|-------|-------|
| 1 | star | 3     | stars | 5     | stars |
|   |      |       |       |       |       |
|   | v    |       | v     |       | v     |
|   | 0,   | 0,    | 20,   | 0,    | 80    |
|   |      | ^     |       | ^     |       |
|   |      |       |       |       |       |
|   | 2    | stars | 4     | stars |       |

Excerto de Código 3.1: Exemplo de formatação da *string* resultante do método *setGraphic*

Na função *test* do *JavaScript* é separada a *string* pelas vírgulas, gerando um *array* automaticamente e, de seguida, é chamada outra função, *draw*, que recebe como parâmetro o *array* anteriormente criado. Nessa função, apenas é necessário atribuir os valores de cada elemento do *array* nos respetivos *datapoints*.

### 3.3.2.2 Apresentação das casas ao user

Apenas as casas que não tenham sido requisitadas pelo *user* irão aparecer nesta secção, para o caso do *user guest*, aparecerão todas as casas presentes na base de dados.

Quando é apresentada a página, é invocado o método *updateHousesArray* referente ao *Home\_Service bean* que, com base no *username* recebido por parâmetro, gera o *arraylist*, da classe *Houses*, com as casas cujos ainda não tenham sido requisitadas pelo *user* em questão; para evitar sobreposição de casas quando efetuado um *refresh* da página, é sempre verifica com recurso ao método *checkData* se a casa que está a ser adicionada ao *arraylist* não está já contida neste. Após ultrapassada esta etapa, segue-se a questão de como iremos disponibilizar toda a informação depositada no *arraylist* no *frontend* e, com recurso à *tag ui:repeat*, que funciona como um *loop* de um determinado *arraylist*, podemos efetuar o pretendido de uma forma bastante simples.

Relativamente ao *layout* em si decidimos dispôr as casas duas a duas, sendo que todas estas apresentam um padrão comum, imagem da casa, se-

guida de um botão para dar *request*, por baixo, são apresentadas as características da casa, rua, código-postal, o valor, o nome do proprietário, a avaliação atribuída e, por fim, a descrição.

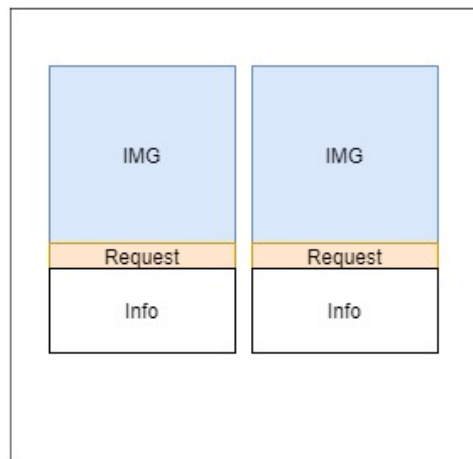


Figura 3.5: Exemplo do mapeamento das casas.

### 3.3.2.3 Request de uma casa

Para efetuar um *request* de uma determinada casa, o *user* terá apenas de premir o botão situado abaixo da imagem correspondente à casa em questão. Após premido, é invocado o método *requestHouse* do *Home\_Service bean* que, com base no *address* e o *zip code* recebidos como parâmetro, irá declarar que essa casa está a partir de agora na *request list* do determinado *user*.

### 3.3.3 Perfil – *profile.xhtml*

A página de perfil, contém, inicialmente, uma secção destinada à alteração da *password* do *user* em questão. De seguida, encontrará a secção das casas que reservou e requisitou. Caso a casa não esteja na sua posse, terá uma indicação de que se encontra em fila de espera "*Queue*" ou, caso a possua, o *user* terá um botão que pode pressionar quando pretender entregá-la, "*Deliver*", e consequentemente atribuindo-lhe uma avaliação. Por fim, existe a secção onde poderá adicionar uma nova casa.

Todas as secções referidas no parágrafo anterior estão devidamente especificadas/detalhadas nas próximas subsecções.

### 3.3.3.1 Verificação inicial

Quando a página do *profile* é apresentada, é invocado o método *validUserData* do *Stateful bean User* que vai validar os dados do utilizador. Caso estes sejam verificados, a página *profile* será carregada com sucesso, caso contrário, será mostrado um erro ao *user*. Posteriormente à verificação dos dados, a página do *profile* do *user* irá dispôr todas as casas que este *user* requisitou, que estão disponíveis para devolução, e reservou.

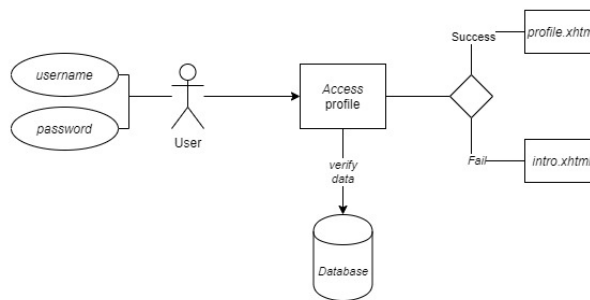


Figura 3.6: Diagrama demonstrativo da verificação inicial aquando na página de perfil.

### 3.3.3.2 Mudança de *password*

O *user* vai ter a possibilidade de mudar a sua *password* preenchendo as caixas de texto que estão no topo da página do *profile*, onde irá inserir a nova *password*, a confirmação desta e a sua *password* atual.

Ao premir o botão de *Submit*, é invocado o método *changePassword* onde se vai dar a verificação da nova *password* com a confirmação da mesma. De seguida é feita a ligação à base de dados e se a *password* atual do *user* corresponder com a que está na base de dados a *password* atual é atualizada para a nova *password*.

### 3.3.3.3 Apresentação das casas ao *user*

Apenas serão listadas todas as casas que o *user* tenha requisitado/reservado.

De forma a aproveitar o que tinha já sido elaborado na página do *home.xhtml*, utilizámos o mesmo processo para dispôr as casas e consequentes botões e informação, mas havia um impasse relativamente ao novo que constava dentro desse botão uma vez que, dependendo do estado da casa o texto contido no botão, e consequente cor, teria de alterar de *user* para *user*. Optámos por criar uma nova classe, *HousesProcess* que tem como variáveis:

- Casa em questão (*Object Houses*);
- Cor do botão (*string*);
- Texto contido no botão (*string*).

Deste modo, o *loop* presente no *frontend* irá ser respetivo ao *arraylist* da classe *HousesProcess* e não das *Houses*. A atribuição dos valores da cor e do texto, para cada objeto contido no *arraylist*, está devidamente ilustrado na figura abaixo para uma maior clareza.

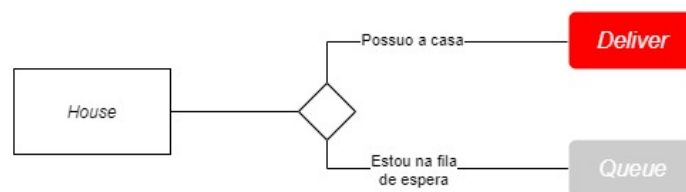


Figura 3.7: Diagrama para formatação do botão associado à casa no perfil.

#### 3.3.3.4 Entregar casa

O *user* para devolver uma casa terá de premir o botão *Deliver* da respetiva casa, este, tem associado uma função de *JavaScript*, *validateButton*, em que é passado como parâmetro o estado da casa atual e o respetivo *id* do *popup* associado a esta casa. Caso seja validado, estado igual a "*Deliver*", o *popup* com o *id* igual ao que foi passado como parâmetro irá ficar visível.

Neste *popup*, é disponibilizado ao *user* 5 botões em que, cada um deles, está associado a um nível de avaliação. Todos estes botões, têm associados o método *updateRate* do *Stateful bean Profile\_Service*. Este método permitirá entregar a casa, através de uma chamada ao método *deliverHouse* e, atribuir a avaliação à respetiva casa. Especificamente, se for a primeira vez que a casa está a ser devolvida, a avaliação que o *user* der vai ser o *rate* final da casa, caso contrário vai ser realizada a média de todas as avaliações previamente atribuídas com a nova, dando um novo valor ao *rate* final da casa.

#### 3.3.3.5 Adicionar casa

Para o *user* introduzir uma nova casa para futuramente ser alugada, terá de introduzir a morada respetiva à casa, de seguida o correspondente código postal, seguido da renda que pretende receber, uma caixa de texto onde poderá referir todas as propriedades da sua casa e, por fim, poderá efetuar um *upload* de uma imagem da sua casa.

Ao premir o botão associado à submissão dos dados, o método *insert-NewHouse* é invocado. Aqui tudo o que o *user* escreveu vai ser adicionado à base de dados na tabela *Houses*.

Se o *user* não introduzir nenhuma imagem irá ser atribuída uma imagem padrão, mas se introduzir uma imagem, o nome do ficheiro irá ser alterado para um novo nome composto pela morada, código postal e o nome do ficheiro.

### 3.3.4 Conclusões

Neste capítulo ficou demonstrado os processos, ideias, classes e funções que serviram de base para a nossa estruturação e implementação do trabalho que foi proposto. De destacar a devida documentação realizada no código para uma melhor compreensão deste.

As várias secções aqui apresentadas são referentes aos mais importantes aspetos que a nossa aplicação exhibe.

Com estas explicações pretendemos esclarecer qualquer dúvida que possa surgir durante a leitura, compreensão e interpretação do código criado.

## Capítulo

# 4

## Manuais

### 4.1 Manual de instalação

Para configurar este projeto é muito simples, basta seguir os seguintes passos:

1. *Clone*

- Para clonar este projeto, se utilizar *Linux* ou *Unix* escreva o seguinte na linha de comandos:  
`git clone https://github.com/rui-raposo/SD-Project.git`

2. Abrir o projeto

- Para abrir o projeto use o seu *Integrated Development Environment* (IDE) pessoal.

3. Dar *Deploy* à aplicação *web*

- No seu IDE pessoal clique na pasta do projeto com o botão direito do rato e selecione *Deploy*.

4. Executar a aplicação *web*

- No seu IDE pessoal clique na pasta do projeto com o botão direito do rato e selecione *Run*.

## 4.2 Manual de Utilização

### 4.2.1 *Sign In*

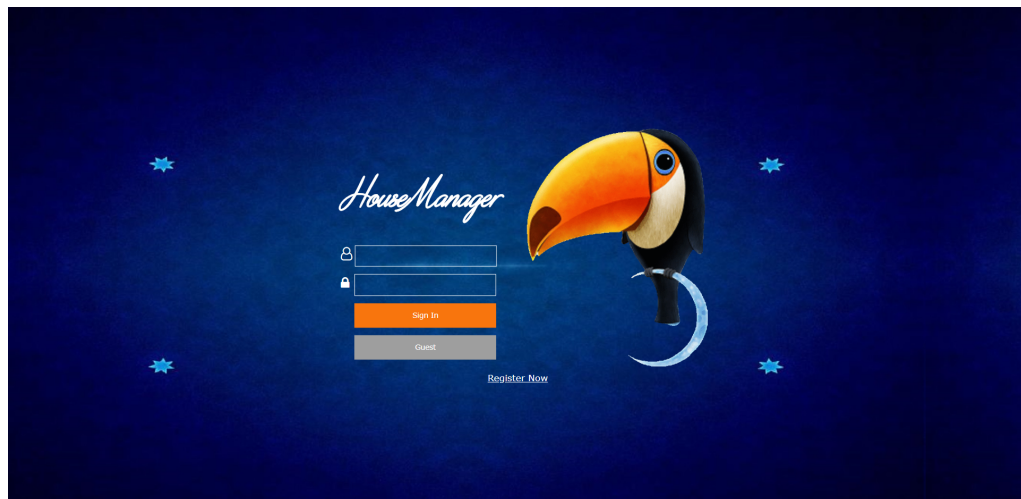


Figura 4.1: Página de autenticação.

### 4.2.2 Inserir uma nova casa

The image shows a web form titled 'Adding Houses'. At the top, there is a navigation bar with links: HOME, NEWS, ABOUT, ABOUT, TEAM, FAQ, PROFILE, and LOGOUT. Below the navigation bar, the text 'my HOUSES' is displayed, followed by the subtitle 'Here, you can find all your'. The form itself has a title 'Adding Houses' and a subtitle 'Here, you can find all your'. It contains several input fields: 'Address' (with the value 'Casa das Muralhas'), 'Zip Code' (with the value '6200' and a dropdown for '162'), 'Lease' (with the value '1500' and a dropdown for 'c'), and 'Features' (with the text 'A Casa das Muralhas disponibiliza um restaurante, uma piscina exterior sazonal, um bar e um salão partilhado na Covilhã.'). Below the 'Features' field, there is a file upload section with the text 'Escolher ficheiro' and a file name '177810269.jpg'. At the bottom of the form, there is an orange 'Submit' button.

Figura 4.2: Página de inserir casas



### 4.2.3 Visualizar casas

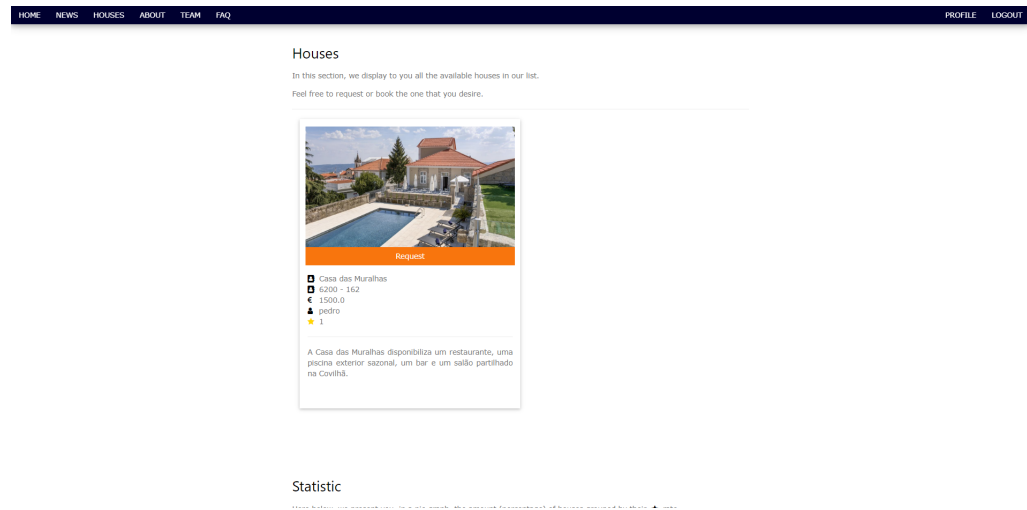


Figura 4.3: Página onde se dá *display* as casas

### 4.2.4 Casa requisitada

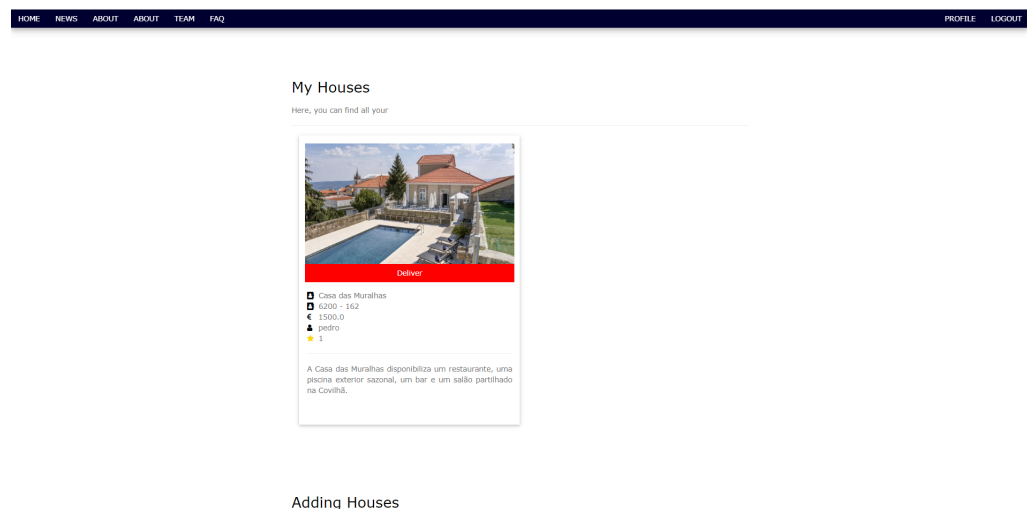


Figura 4.4: Página de casas requisitadas

### 4.2.5 Casa em espera

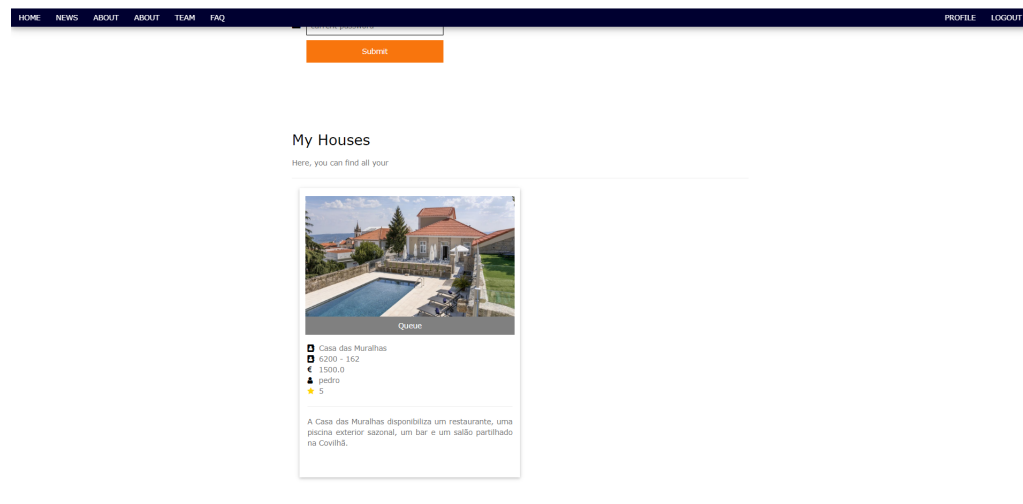


Figura 4.5: Quando a casa não está disponível para requisitar imediatamente.

### 4.2.6 Rating da casa

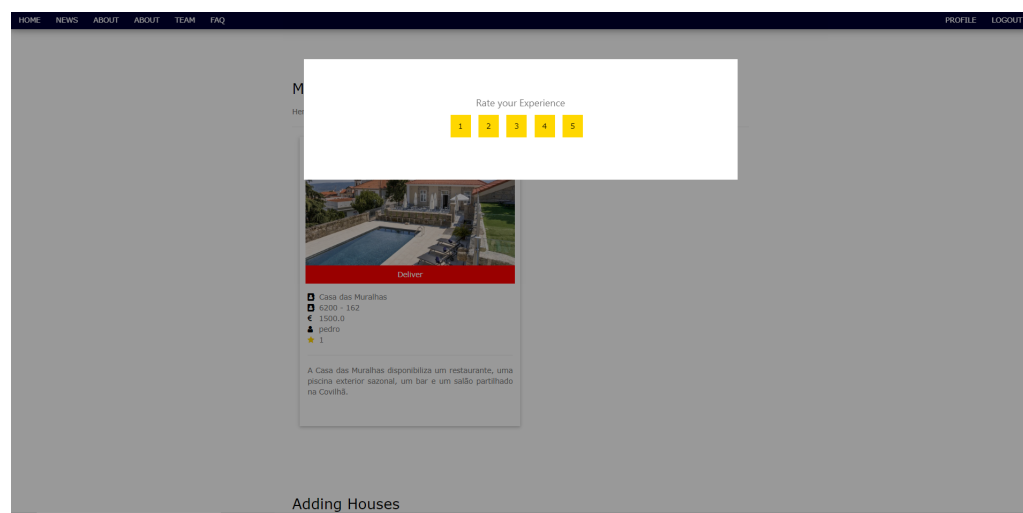


Figura 4.6: *popup*

4.2.7 Estatísticas de casa

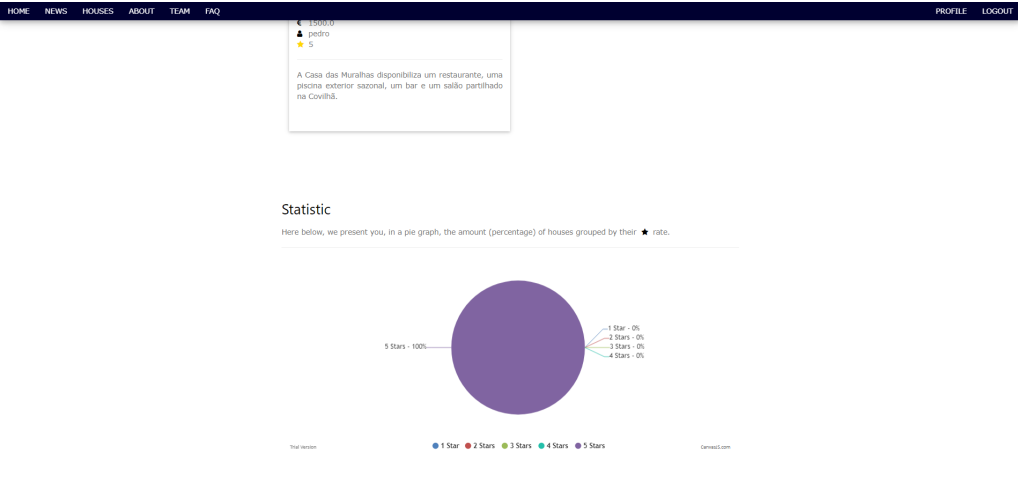


Figura 4.7: Página do gráfico das estatísticas de uma só casa.

4.2.8 Múltiplas casas

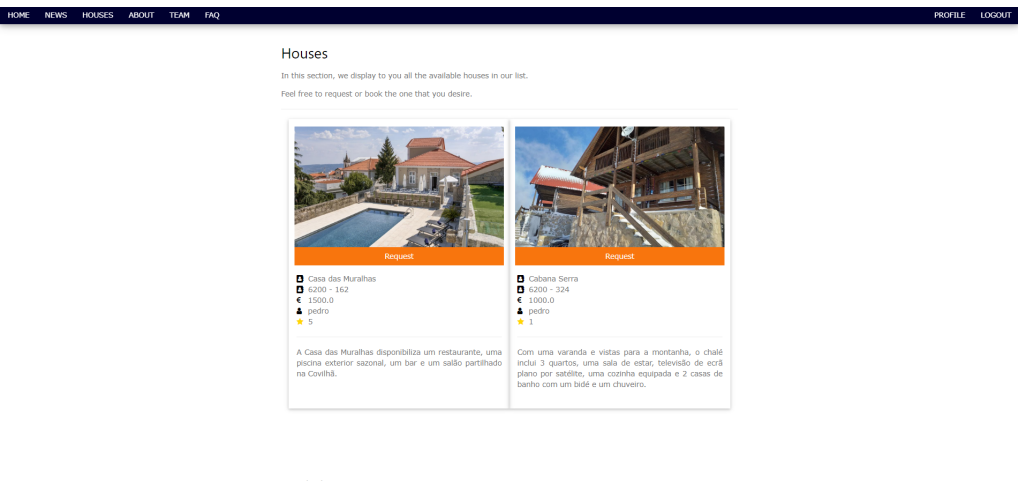


Figura 4.8: *Display* de várias casas.

4.2.9 Estatísticas de várias casas



Figura 4.9: Página do gráfico das estatísticas de várias casas.

## ***Bibliografia***

- [1] SPPP *Web Site* professora Paula Prata. Universidade da Beira Interior Sistemas Distribuídos – 2019/2020, 2020. [Online] [https://www.di.ubi.pt/~pprata/spd/SD\\_19\\_20\\_TrabalhoPratico2.pdf](https://www.di.ubi.pt/~pprata/spd/SD_19_20_TrabalhoPratico2.pdf). Último acesso 12 de junho de 2020.