



Universidade do Minho
Escola de Engenharia

UNIVERSIDADE DO MINHO

DEPARTAMENTO DE INFORMÁTICA

Serve.Me
Engenharia de Aplicações
Grupo 6

Catarina Machado (A81047)
João Costa (a81822) Tiago Fontes (a80987)

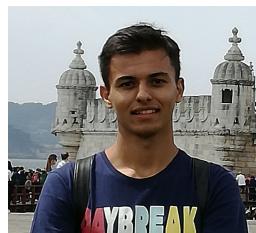
21 de Julho de 2020



(a) Catarina.



(b) João.



(c) Tiago.

Conteúdo

1	Introdução	6
1.1	Motivação	6
1.2	Contextualização	6
1.3	Caracterização dos utilizadores	7
1.3.1	Cliente	7
1.3.2	Prestador de Serviços	7
1.4	Fluxo geral da plataforma	8
1.4.1	Prestador de Serviços	8
1.4.2	Cliente	8
2	Modelação	9
2.1	Requisitos	9
2.1.1	Requisitos Funcionais	9
2.1.2	Requisitos Não Funcionais	11
2.2	Diagrama de Use Cases	12
2.3	Especificação de Use Cases	13
2.3.1	Classificar um serviço	13
2.3.2	Cancelar um agendamento	14
2.3.3	Aceitar proposta de agendamento	14
2.3.4	Consultar perfil do Prestador de Serviços	15
2.3.5	Publicar um pedido de serviço	15
2.3.6	Aceder ao horário	16
2.3.7	Propor agendamento de um serviço	16
2.4	Modelos de Tarefas	17
2.4.1	Publicar um pedido de serviço	17
2.4.2	Aceitar proposta de agendamento	18
2.4.3	propor agendamento de um serviço	18
2.5	Diagrama de Classes	19
2.6	Protótipo da Interface	22
2.6.1	Página Inicial	22
2.6.2	Página Inicial - Login	23
2.6.3	Página Inicial - Registo	24
2.6.4	Login Cliente	25
2.6.5	Registo Cliente	26
2.6.6	Login Prestador	27
2.6.7	Registo Prestador	28
2.6.8	Cliente - Serviços Agendados	29
2.6.9	Cliente - Serviços Publicados	30
2.6.10	Cliente - Histórico	31
2.6.11	Cliente - Inbox	32
2.6.12	Cliente - Publicar pedido	33
2.6.13	Cliente - Ver Perfil	34
2.6.14	Cliente - Ver Perfil Prestador	35
2.6.15	Cliente - Classificar serviço	36
2.6.16	Prestador - Serviços Agendados	37
2.6.17	Prestador - Histórico	38
2.6.18	Prestador - Estatísticas	39
2.6.19	Prestador - Inbox	40

2.6.20	Prestador - Consultar pedidos	41
2.6.21	Prestador - Pedido de Agendamento	42
2.6.22	Prestador - Ver Perfil	43
2.6.23	Prestador - Ver Perfil Cliente	44
2.6.24	Prestador - Classificar serviço	45
3	Implementação	46
3.1	Considerações Iniciais	46
3.2	Frontend	47
3.2.1	Views	47
3.2.2	Pedidos à API	47
3.3	Backend	48
3.3.1	Controllers	48
3.3.2	Beans	48
3.3.3	Base de Dados	49
4	Deploy	50
4.1	Considerações Iniciais	50
4.2	Frontend	52
4.3	Balanceador	53
4.4	Backend	53
5	Análise de Carga	55
5.1	Considerações gerais	55
5.2	Testes de Carga	55
5.2.1	Interação 1 - Cliente	56
5.2.2	Interação 2 - Prestador	58
6	Manual de Utilização	61
6.1	Página Inicial	61
6.2	Página Inicial - Login	62
6.3	Página Inicial - Registo	62
6.4	Login Cliente	63
6.5	Login Prestador	63
6.6	Login Cliente - Mensagem de Erro	64
6.7	Registo Cliente	65
6.8	Registo Prestador	66
6.9	Registo - Erros	67
6.10	Páginas web do Cliente	71
6.10.1	Serviços Agendados	71
6.10.2	Pedidos Publicados	72
6.10.3	Histórico	75
6.10.4	Inbox	75
6.10.5	Publicar pedido	78
6.10.6	Ver/editar perfil	81
6.10.7	Alterar password	81
6.11	Páginas web do Prestador	84
6.11.1	Serviços Agendados	84
6.11.2	Histórico	84
6.11.3	Estatísticas	85

6.11.4	Inbox	86
6.11.5	Propostas	86
6.11.6	Consultar pedidos	87
6.11.7	Perfil	92
6.12	404 - <i>Page not found</i>	92
7	Análise de Usabilidade	93
7.1	Princípios de Usabilidade	93
7.1.1	<i>Learnability</i>	93
7.1.2	<i>Flexibility</i>	93
7.1.3	<i>Robustness</i>	93
7.2	Heurísticas de Nielsen	94
7.2.1	<i>Visibility of system status</i>	94
7.2.2	<i>Match between system and the real world</i>	94
7.2.3	<i>User control and freedom</i>	94
7.2.4	<i>Consistency and standards</i>	95
7.2.5	<i>Error prevention</i>	95
7.2.6	<i>Recognition rather than recall</i>	95
7.2.7	<i>Flexibility and efficiency of use</i>	96
7.2.8	<i>Aesthetic and minimalist design</i>	96
7.2.9	<i>Help users recognize and recover from errors</i>	96
7.2.10	<i>Help and documentation</i>	96
8	Conclusões e Trabalho Futuro	97

Lista de Figuras

2	Diagrama de Use Cases.	13
3	Especificação do Use Case Classificar um serviço.	13
4	Especificação do Use Case Cancelar um agendamento.	14
5	Especificação do Use Case Aceitar proposta de agendamento.	14
6	Especificação do Use Case Consultar perfil do Prestador de Serviços.	15
7	Especificação do Use Case Publicar um pedido de serviço.	15
8	Especificação do Use Case Aceder ao horário.	16
9	Especificação do Use Case Propor agendamento de um serviço.	16
10	Modelo de Tarefas: Publicar um pedido de serviço.	17
11	Modelo de Tarefas: Aceitar proposta de agendamento.	18
12	Modelo de Tarefas: propor agendamento de um serviço.	18
13	Diagrama Conceptual de Classes.	19
14	Diagrama de classes.	21
15	<i>Mockups:</i> Página Inicial.	22
16	<i>Mockups:</i> Página Inicial (Login).	23
17	<i>Mockups:</i> Página Inicial (Registo).	24
18	<i>Mockups:</i> Login Cliente.	25
19	<i>Mockups:</i> Registo Cliente.	26
20	<i>Mockups:</i> Login Prestador.	27
21	<i>Mockups:</i> Registo Prestador.	28
22	<i>Mockups:</i> Cliente - Serviços Agendados.	29
23	<i>Mockups:</i> Cliente - Serviços Publicados.	30
24	<i>Mockups:</i> Cliente - Histórico.	31
25	<i>Mockups:</i> Cliente - Inbox.	32
26	<i>Mockups:</i> Cliente - Publicar pedido.	33
27	<i>Mockups:</i> Cliente - Ver Perfil.	34
28	<i>Mockups:</i> Cliente - Ver Perfil Prestador.	35
29	<i>Mockups:</i> Cliente - Classificar serviço.	36
30	<i>Mockups:</i> Prestador - Serviços Agendados.	37
31	<i>Mockups:</i> Prestador - Histórico.	38
32	<i>Mockups:</i> Prestador - Estatísticas.	39
33	<i>Mockups:</i> Prestador - Inbox.	40
34	<i>Mockups:</i> Prestador - Consultar pedidos.	41
35	<i>Mockups:</i> Prestador - Pedido de Agendamento.	42
36	<i>Mockups:</i> Prestador - Ver Perfil.	43
37	<i>Mockups:</i> Prestador - Ver Perfil Cliente.	44
38	<i>Mockups:</i> Prestador - Classificar serviço.	45
39	<i>Deploy:</i> Arquitetura.	51
40	Testes de Carga: Gráfico resultante simulação com 100 <i>threads</i> para Cliente.	57
41	Testes de Carga: Gráfico resultante simulação com 500 <i>threads</i> para Cliente.	57
42	Testes de Carga: Gráfico resultante simulação com 100 <i>threads</i> para Prestador.	59
43	Testes de Carga: Gráfico resultante simulação com 400 <i>threads</i> para Prestador.	59
44	Página Inicial.	61
45	Página Inicial - Login.	62

46	Página Inicial - Registo.	62
47	Login Cliente.	63
48	Login Prestador.	63
49	Login Cliente - Mensagem de Erro.	64
50	Registo Cliente.	65
51	Registo Prestador.	66
52	Registo - Erro e-mail.	67
53	Registo - Erro número de contribuinte.	67
54	Registo - Erro: informações já em uso.	68
55	Registo - Distritos.	68
56	Registo - Concelhos Braga.	69
57	Registo - Campos em falta.	70
58	Cliente - Serviços Agendados.	71
59	Cliente - Cancelar Agendamento.	71
60	Cliente - Classificar Serviço.	72
61	Cliente - Navbar: Os meus serviços.	73
62	Cliente - Pedidos publicados.	73
63	Cliente - Cancelar pedido.	73
64	Cliente - Sucesso ao cancelar pedido.	74
65	Cliente - Editar pedido.	74
66	Cliente - Histórico.	75
67	Cliente - Inbox (com propostas).	76
68	Cliente - Aceitar pedido agendamento.	76
69	Cliente - Recusar pedido agendamento.	76
70	Cliente - Sucesso ao aceitar pedido agendamento.	77
71	Cliente - Inbox (com serviços por classificar e avisos de cancelamento).	77
72	Cliente - Publicar pedido.	78
73	Cliente - Inserir data.	79
74	Cliente - Inserir hora de início.	80
75	Cliente - Sucesso ao publicar pedido.	80
76	Cliente - Ver perfil.	81
77	Cliente - Alterar password.	82
78	Cliente - Erro passwords não coincidem.	82
79	Cliente - Erro password atual incorreta.	83
80	Prestador - Serviços Agendados.	84
81	Prestador - Navbar: Dashboard.	85
82	Prestador - Estatísticas.	85
83	Prestador - Inbox.	86
84	Prestador - Propostas.	86
85	Prestador - Consultar pedidos.	88
86	Prestador - Consultar pedidos (ordenar).	89
87	Prestador - Ver perfil do Cliente.	90
88	Prestador - Efetuar proposta.	90
89	Prestador - Erro ao efetuar proposta.	91
90	Prestador - Sucesso ao efetuar proposta.	91
91	Erro 404: <i>Page not found.</i>	92

1 Introdução

1.1 Motivação

Desde o início da humanidade, praticamente todas as ações que são diariamente realizadas pelos seres humanos podem ser consideradas serviços, seja para usofruto próprio ou de outra pessoa. A área dos serviços é vasta e imensa, e inevitavelmente ao longo dos anos surgiu a necessidade de comercializar estas ações. Por um lado, temos clientes que devido a diversos fatores como a correia das suas profissões e a falta de jeito, sentiram a necessidade de uma ajuda extra em certas tarefas. No outro lado, temos os prestadores de serviços, que tiram partido das suas mais-valias e do seu tempo livre para embolsar dinheiro, podendo até fazer disso profissão.

Desta forma, vários anúncios de serviços começaram a ser colocados em postes de eletricidade, jornais e até nos simples passa-a-palavra do nosso quotidiano. Surge assim muita informação descentralizada e dispersa. A própria comunidade de prestadores é desconhecida e até para os clientes por vezes torna-se complexo e desgastante esta procura, muitas vezes sem sucesso. Para além disso, muitas vezes estes anúncios são orientados às características do prestador, e não às necessidades do cliente, dificultando ainda mais a tarefa.

Surge assim imperativamente a necessidade de uma Plataforma de Procura e Contratação de Serviços, focada em construir uma atmosfera de entreajuda e de partilha de anúncios, garantindo a qualidade e compromisso de que todos os envolvidos precisam.

1.2 Contextualização

A Serve.Me é uma **Plataforma de Procura e Contratação de Serviços** que vem dar resposta às necessidades dos seres humanos, com todas as funcionalidades essenciais para que o processo de procura e de contratação de serviços seja para sempre simplificada, fácil e rápida.

Os principais intervenientes são o **Prestador de Serviços** e o **Cliente**. Consequentemente, as principais características da aplicação são as seguintes:

- Permitir a prestadores de serviços e clientes registarem-se;
- Permitir aos clientes apresentarem o serviço que pretendem, adicionando a categoria (predefinidas pelo sistema), uma descrição (opcional), o preço que estão dispostos a pagar por hora, a data e horário da disponibilidade e a duração prevista do serviço;
- O prestador de serviços pode consultar a lista de serviços pretendidos pelos clientes e acolher pedidos, tendo para isso que indicar o horário para a realização do mesmo e o preço que pretende receber. O cliente terá que aceitar o pedido para desta forma confirmar o agendamento do serviço;
- Ambas as partes classificam o serviço no final (o prestador e o cliente);
- Tanto o prestador como o cliente têm um perfil com a sua classificação, comentários e número de serviços prestados/usufruídos. Os clientes apenas conseguem ver as opiniões acerca do prestador que respondeu ao seu anúncio e os prestadores apenas conseguem ver as opiniões acerca dos clientes que pretendem um serviço.

Numa fase inicial, decidimos que o domínio se debruçaria por **Serviços de Jardinagem e Bricolage**, mas no futuro poderão ser adicionados todos os domínios desejados.

1.3 Caracterização dos utilizadores

De modo a concluir a fundamentação da criação desta plataforma, fizemos ainda um pequeno estudo onde foi possível perceber a elevada abrangência do público-alvo da plataforma. Por um lado, temos os Prestadores de Serviços, jardineiros e/ou habilidosos, e por outro os Clientes, que constituem a comunidade em geral, com aproximadamente entre os 20 e os 60 anos.

Recorremos assim à caracterização de duas *personas* que vieram indubitavelmente comprovar a necessidade da criação desta plataforma e que passamos a apresentar em seguida.

1.3.1 Cliente

Joel Carvalho, 32 anos, Braga

Joel é gestor de uma empresa multinacional, sediada em Braga, e que acabou de se mudar da sua terra natal, Viseu, para o distrito onde se encontra o escritório da sua empresa. Mudou de casa, tendo escolhido uma com grande jardim.

Ultimamente tem pouco tempo para cuidar do jardim, mas não conhece ninguém em Braga. Tem medo de não gostar do trabalho realizado, mas não tem qualquer conhecido que lhe possa informar sobre jardineiros na região. Sabendo dos preços de algumas empresas especialistas, considera-os demasiado caros (pois conhece bem as políticas das empresas), mas necessita realmente que alguém lhe cuidem do jardim.

É um homem inteligente, que gosta de primar pela organização e bastante cuidadoso no que toca a contratar alguém para realizar trabalhos no seu espaço. Gosta de ajudar o próximo e não tem problemas em recomendar alguém que realmente lhe faça um bom trabalho.

1.3.2 Prestador de Serviços

Ricardo Silva, 27 anos, Ferreiros

Ricardo é trabalhador numa estufa em Ferreiros e desde criança gosta bastante da jardinagem e bricolage. Ricardo tem um belo jardim em sua casa, totalmente construído e mantido por si.

Realiza pequenos trabalhos em várias casas que o conhecem desde miúdo, mas sente que poderia ganhar mais algum dinheiro se mais pessoas conseguissem saber das suas capacidades. É bastante limpo e pontual, segundo os seus familiares.

É um homem muito esquecido e que necessita de ter sempre tudo apontado. Precisa de uma agenda para poder saber o que tem para fazer em determinada semana e gosta particularmente de apontar todos os seus ganhos. No entanto, como estes serviços ao domicílio são pagos em numerário, esquece-se muitas vezes de apontar os valores.

Tem alguma disponibilidade durante a semana, mas está habituado a realizar trabalhos em muitos dos sábados do ano. Gosta de falar com o cliente e de

propor preços para os seus serviços, uma vez que considera que as pessoas que o chamam por vezes não sabem o valor de certos trabalhos.

1.4 Fluxo geral da plataforma

O fluxo geral da plataforma deve ser simples e minimalista. A plataforma deverá possuir como funcionalidades base a **criação** de um perfil por parte de prestadores de serviço e de clientes, a **adicação** de pedidos de serviços por parte de clientes e a **consulta** da lista dos pedidos que os clientes pretendem ver satisfeitos.

Todos os pedidos devem pertencer a uma **categoria** e os prestadores de serviço e clientes poderão adicionar alguma descrição mais detalhada, caso seja necessário.

Tal como já mencionado, a plataforma possui dois diferentes atores, o Prestador de Serviços e o Cliente, e, em seguida, encontra-se explicado o fluxo de utilização ideal da aplicação do ponto de vista de cada um destes elementos.

1.4.1 Prestador de Serviços

Qualquer membro pode registar-se na aplicação como Prestador de Serviços. Para tal, é necessário que forneça **informações pessoais** como o nome, password, número de contribuinte, morada, número de telemóvel e o e-mail.

O prestador de serviços pode, a qualquer momento, **consultar a lista dos pedidos** inseridos pelos clientes e acolher pedidos, indicando o horário para a realização do mesmo e o preço por hora. Desta forma, será enviada uma notificação ao cliente para aceitar e efetivar o agendamento do serviço.

O prestador de serviços pode consultar a lista de serviços agendados, de serviços que estão pendentes de confirmação, do histórico de serviços e uma *dashboard* com estatísticas.

Por fim, deve ainda classificar todos os clientes dos serviços que realizar e consultar as opiniões de outros prestadores relativamente aos seus futuros clientes.

1.4.2 Cliente

Para usufruírem de algum serviço, os clientes deverão registar-se na aplicação como Clientes. Para tal, é necessário que forneçam **informações pessoais** como o nome, password, número de contribuinte, morada, número de telemóvel e o e-mail. Desta forma, depois de ter um **perfil** criado, pode **adicionar os pedidos de serviços** que pretende, indicando a categoria, descrição do mesmo (opcional), o preço por hora e a duração prevista.

Desta forma, o cliente tem uma caixa de entrada, onde pode consultar os pedidos de agendamento propostos pelos prestadores de serviços e aceitá-los ou recusá-los.

O cliente pode consultar a lista dos pedidos publicados, dos serviços agendados e o histórico. Deve também classificar os prestadores de serviços no final de cada serviço e pode também consultar as opiniões relativas aos determinados prestadores, para assim poder ter essas opiniões em consideração na hora de aceitar uma proposta de agendamento de serviços.

2 Modelação

A fase de modelação consiste na definição e formalização das necessidades da plataforma. Esta fase engloba os requisitos funcionais e não funcionais da aplicação, o diagrama e consecutiva especificação dos *use cases*, modelos de tarefas, diagrama de classes e o protótipo da interface gráfica.

2.1 Requisitos

Para fazer o levantamento de requisitos foi feita uma auscultação aos potenciais utilizadores da aplicação, o que nos levou aos resultados que se encontram nas seguintes secções.

2.1.1 Requisitos Funcionais

Para uma melhor apresentação dos requisitos funcionais da aplicação, decidimos dividi-los em dois tópicos: os requisitos funcionais do ponto de vista do ator **Cliente** e do ponto de vista do ator **Prestador de Serviços**.

Cliente

1. Permitir o **registro** na aplicação, tendo que indicar nome, password, número de contribuinte, morada, distrito, concelho, freguesia, número de telemóvel e o e-mail;
2. Permitir o **login** na aplicação, indicando o email e a password;
3. **Publicar um pedido de serviço**, indicando a classe, categoria, descrição (opcional), o preço por hora, a data e horário da disponibilidade e a duração prevista do serviço.

Estão disponíveis as seguintes categorias: “Vedação para Jardim”, “Decoração de Jardins”, “Manutenção de Canteiros”, “Construção de Jardim”, “Corte de Árvores”, “Remoção de Ervas Daninhas”, “Preparação do Solo para Jardinagem”, “Limpeza de Jardim”, “Plantação de Árvores” e “Outros”;

4. **Aceitar um pedido de agendamento**, confirmado as condições propostas;
5. **Classificar** um serviço, indicando uma classificação de 0 a 5 estrelas e um comentário;
6. **Consultar os serviços** (publicados, agendados, expirados, cancelados e realizados);
7. **Editar** uma publicação;
8. **Cancelar** uma publicação;
9. **Cancelar** um agendamento de serviço;
10. Consultar o **perfil de um prestador** de serviço que lhe propõe um agendamento;

11. **Consultar as propostas** de agendamento recebidas;
12. **Editar** perfil.

Prestador de Serviços

1. Permitir o **registo** na aplicação, indicando nome, password, número de contribuinte, morada, distrito, concelho, freguesia, número de telemóvel e e-mail;
2. Permitir o **login** na aplicação, indicando o email e a password;
3. **propor o agendamento de um serviço**, indicando o preço por hora e a hora de início;
4. **Classificar** um serviço, indicando uma classificação de 0 a 5 estrelas e um comentário;
5. **Consultar os serviços** (agendados, cancelados e realizados);
6. **Cancelar** um agendamento de serviço;
7. Aceder ao **horário** de serviços agendados;
8. Consultar **dashboard** de ganhos monetários anuais, número de serviços prestados anual e gráficos de ganhos/mês, número de serviços/mês e número de serviços por categorias;
9. Consultar o **perfil de um cliente** que disponibiliza um pedido de serviço;
10. Consultar o **estado das propostas** de agendamento enviadas (pendente, aceite ou rejeitada);
11. Consultar a **lista de pedidos** dos clientes;
12. **Editar** o perfil.

2.1.2 Requisitos Não Funcionais

A nível de requisitos não funcionais, é essencial que a plataforma tenha as seguintes características:

1. A aplicação tem que estar disponível em **língua portuguesa**;
2. A aplicação **não deve ser ofensiva** em termos religiosos, étnicos e sexuais;
3. A aplicação deve ser suportada pelos seguintes **browsers**: Safari, Google Chrome, Firefox, Microsoft Edge;
4. Ao fim de **10 minutos**, um utilizador do público-alvo deve ser capaz de realizar a principal funcionalidade do sistema;
5. O sistema deve suportar o registo de pelo menos 5000 pessoas num ano;
6. O sistema deve estar operacional a maioria do ano, pelo menos 350 dias por ano (95%);
7. O sistema deve representar resposta a todas as ações do utilizador em menos de 2 segundos (desprezando atrasos da rede);
8. Após a abertura da página, a mesma deve estar totalmente carregada em menos de 2 segundos (desprezando atrasos da rede);
9. Para incluir uma nova funcionalidade, deve ser lançada durante a noite no horário local;
10. O sistema deve prevenir a introdução de dados errados, por exemplo a validação de e-mails.

2.2 Diagrama de Use Cases

O processo de desenvolvimento do **Diagrama de Use Cases**, a par dos Requisitos Funcionais apresentados na Secção 2.1.1, tratou-se de um processo iterativo e incremental, sempre com o intuito de elaborar uma plataforma com funcionalidades que realmente vão ao encontro das necessidades dos seus futuros utilizadores.

Após várias deliberações, chegamos ao Diagrama de Use Cases presente na Figura 2. Através deste diagrama, é possível perceber que o sistema terá dois atores: o Cliente e o Prestador de Serviços.

O Cliente tem como funcionalidades o **Registo** na plataforma, a **Autenticação** (login/logout) e também a possibilidade de **Editar o seu Perfil**, incluindo a alteração da palavra-passe. Como seria de esperar, o Cliente pode **Publicar um pedido de serviço** (Figura 7), **Editar um pedido de serviço** e ainda **Eliminar uma publicação**. Tem a possibilidade de **Consultar os pedidos publicados**, **Consultar os pedidos expirados** e de **Consultar as propostas de agendamento** efetuadas por parte dos Prestadores de Serviço, podendo **Aceitar as propostas de agendamento** (Figura 5). O Cliente pode também **Consultar o Perfil dos Prestadores** (Figura 6) que lhe fazem essas propostas. Tem a oportunidade de **Consultar os serviços agendados**, **Cancelar um agendamento** (Figura 4) e **Classificar um serviço** (Figura 3) depois de ele ter sido finalizado. Pode ainda **Consultar os serviços realizados** (histórico).

Por sua vez, o Prestador de Serviços também tem como funcionalidades o **Registo** na plataforma, a **Autenticação** (login/logout) e a possibilidade de **Editar o seu Perfil**, incluindo a alteração da palavra-passe. Pode **Consultar a lista de publicações** criadas pelos Clientes e **Consultar o respetivo perfil do cliente**, bem como **Propor agendamentos de serviços** (Figura 9). Consequentemente, tem a possibilidade de **Consultar o estado das propostas de agendamento enviadas**. O Prestador de Serviços pode também **Consultar os serviços agendados** e **Cancelar um agendamento** (Figura 4). No final de realizar cada serviço, deve também **Classificar o serviço** (Figura 3). Pode ainda **Consultar os serviços realizados**, bem como **Consultar uma dashboard de estatísticas**, que contém informações sobre o total anual de ganhos monetários, o número anual de serviços prestados e gráficos de ganhos/mês, número de serviços/mês e número de serviços por categorias. Por fim, tem também acesso a um calendário que lhe permite **Aceder a um horário** (Figura 8) com todos serviços que tem agendados.

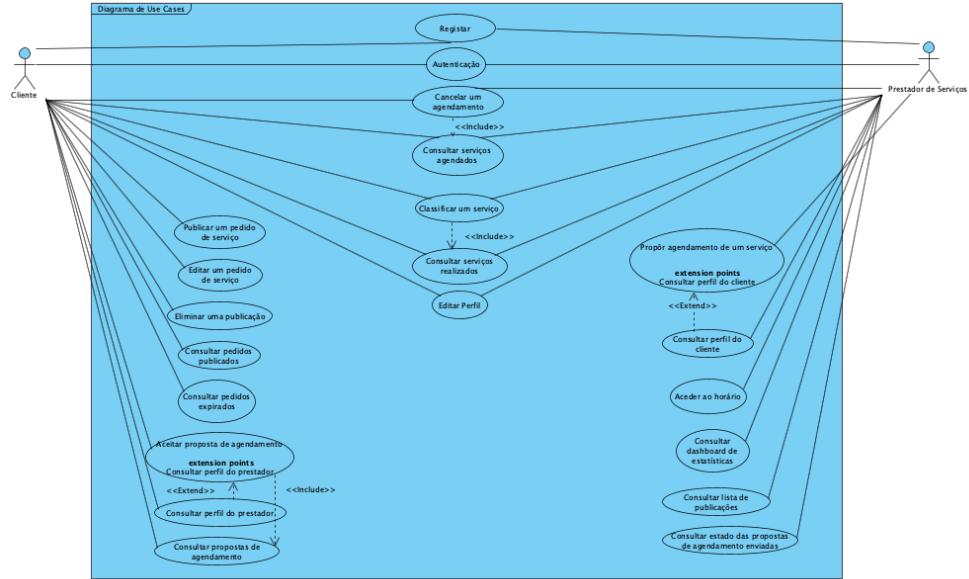


Figura 2: Diagrama de Use Cases.

2.3 Especificação de Use Cases

Iremos agora proceder à especificação dos use cases presentes no Diagrama de Use Cases (Figura 2) que consideramos mais relevantes para a plataforma.

2.3.1 Classificar um serviço

Use Case:	Classificar serviço	
Actor:	Cliente/Prestador	
Pré-condição:	Serviço agendado não classificado	
Pós-condição:	Serviço classificado	
	<i>Actor input</i>	<i>System response</i>
Cenário Normal	2. Insere classificação entre 0 e 5 estrelas 3. Insere opinião	1. Apresenta página de classificação serviço 4. Regista avaliação e altera estado do serviço para Finalizado

Figura 3: Especificação do Use Case Classificar um serviço.

Ambos os atores devem classificar um serviço depois de o realizarem/usufruírem dele. Para tal, basta clicar no serviço que poderá estar presente na sua *inbox* ou na lista dos serviços agendados, e seleccionar o número de estrelas (classificação) que achar mais justa, bem como deixar um comentário (opcional). Desta forma, o serviço ficará finalizado.

2.3.2 Cancelar um agendamento

Use Case:	Cancelar um agendamento	
Actor:	Cliente/Prestador	
Pré-condição:	Serviço confirmado e agendado	
Pós-condição:	Serviço cancelado	
	<i>Actor input</i>	<i>System response</i>
Cenário Normal		1.<<include>> Consultar serviços agendados
	2. Seleciona serviço	3. Regista cancelamento 4. Notifica cliente/prestador do cancelamento do serviço

Figura 4: Especificação do Use Case Cancelar um agendamento.

Tanto o Cliente, como o Prestador, têm a possibilidade de cancelar o agendamento de um serviço. Para isso, devem selecionar o serviço na lista dos serviços agendados e clicar no respetivo botão de cancelamento. O cliente/prestador irá ser notificado deste cancelamento.

2.3.3 Aceitar proposta de agendamento

Use Case:	Aceitar proposta de agendamento	
Actor:	Cliente	
Pré-condição:	Autenticação efetuada e proposta de agendamento pendente	
Pós-condição:	Serviço confirmado e agendado	
	<i>Actor input</i>	<i>System response</i>
Cenário Normal		1.<< include>> Consultar propostas recebidas
	2. Escolher pedido que pretende confirmar	
		3. Apresenta detalhes sobre proposta
	4. <<extend>> Ver perfil prestador	
	5. Aceitar proposta	6. Regista serviço agendado 7. Atualiza serviços agendados cliente 8. Atualiza serviços agendados prestador 9. Atualiza horário do prestador
Exceção 1 [Rejeita proposta recebida] Passo 5.		5.1. Elimina proposta recebida

Figura 5: Especificação do Use Case Aceitar proposta de agendamento.

O Cliente recebe na sua *inbox* as propostas de agendamento de serviços efetuadas pelos Prestadores de Serviços. Desta forma, o Cliente pode consultar os detalhes acerca da proposta, nomeadamente o preço/hora apresentado pelo Prestador e poderá ainda consultar o perfil do mesmo. Assim, o Cliente pode optar por aceitar ou recusar essa proposta. Caso aceite, a sua lista de serviços agendados será atualizada, bem como a lista de serviços agendados do prestador. Para além disso, o horário do prestador será também modificado.

2.3.4 Consultar perfil do Prestador de Serviços

Use Case:	Consultar perfil do prestador	
Actor:	Cliente	
Pré-condição:	Recebida proposta de agendamento por parte do prestador em questão	
Pós-condição:	Visualizar informações sobre prestador	
	<i>Actor input</i>	<i>System response</i>
Cenário Normal		1. Apresenta informações sobre o prestador: número de serviços realizados, classificação média, entre outros

Figura 6: Especificação do Use Case Consultar perfil do Prestador de Serviços.

Quando recebe uma proposta de agendamento de um Prestador de Serviços, o Cliente pode consultar o perfil do determinado Prestador, ficando assim a ter conhecimento relativo ao número de serviços que o Prestador já realizou, bem como o número de serviços agendados que o mesmo já cancelou. Para além disso, fica também a par de informações pessoais como a sua morada e o número de telemóvel, e ainda da classificação média dos seus serviços (0 a 5 estrelas), como também da lista dos respetivos comentários e opiniões por parte dos Clientes.

2.3.5 Publicar um pedido de serviço

Use Case:	Publicar um pedido de serviço	
Actor:	Cliente	
Pré-condição:	Estar autenticado	
Pós-condição:	Novo pedido de serviço publicado	
	<i>Actor input</i>	<i>System response</i>
Cenário Normal	2. Insere informações sobre pedido: classe, categoria, descrição, preço por hora, data e horário de disponibilidade e duração prevista	1. Apresenta página para inserir nova publicação 3. Regista publicação
Exceção 1 [Horário Inválido] Passo 2.		2.1 Apresenta mensagem de erro porque horário está incorreto (ex.: hora de término anterior à hora inicial) 2.2 Insere novo horário
Exceção 2 [Duração prevista é superior ao intervalo de disponibilidade] Passo 2.	2.4 Insere nova duração	2.3 Apresenta mensagem de erro porque duração é inválida

Figura 7: Especificação do Use Case Publicar um pedido de serviço.

O Cliente quando sente a necessidade que outrem lhe dê uma ajudinha extra em alguma tarefa, pode publicar um pedido de serviço. Para tal, deve selecionar essa opção na plataforma e inserir as informações necessárias, nomeadamente a classe, a categoria, descrição (opcional), preço/hora, data e horário de disponibilidade e duração prevista. Caso todos os campos seja corretamente preenchidos, a publicação fica registada na plataforma.

No entanto, todos os campos são validados e caso algum esteja incorreto, o Cliente é convidado a alterá-lo. Para além disso, ocorre uma verificação do horário de disponibilidade, visto que a hora de término deve ser posterior à hora inicial e ocorre ainda uma verificação da duração prevista, pois esta deve ser igual ou superior ao intervalo da disponibilidade.

2.3.6 Aceder ao horário

Use Case:	Aceder Horário	
Actor:	Prestador	
Pré-condição:	<i>Login Efetuado</i>	
Pós-condição:	Horário consultado	
	<i>Actor input</i>	<i>System response</i>
Cenário Normal	1. Selecionar "Serviços Agendados"	2. Apresenta um calendário com o horário dos serviços agendados

Figura 8: Especificação do Use Case Aceder ao horário.

O Prestador de Serviços ao consultar a lista dos serviços que tem agendados, pode também consultar um calendário, onde consegue ter acesso a um horário com uma vista anual, mensal, semanal e diária dos seus serviços agendados.

2.3.7 Propor agendamento de um serviço

Use Case:	Propôr agendamento de um serviço	
Actor:	Prestador	
Pré-condição:	Autenticação efetuada	
Pós-condição:	Proposta de agendamento enviada	
	<i>Actor input</i>	<i>System response</i>
Cenário Normal	2. Seleciona serviço pretendido	1. <<include>> Consultar pedidos 3. Apresenta detalhes do pedido (classe, categoria, descrição, nome do cliente, localização, preço por hora, data e horário de disponibilidade e duração prevista) 4. Insere hora de início serviço e preço/hora requerido 5. Regista proposta 6. Atualiza propostas no cliente
Exceção 1 [Hora de início fora do intervalo autorizado] Passo 4.		4.1 Apresenta mensagem de erro sobre hora inválida 4.2 Insere nova hora, de acordo com disponibilidade do cliente

Figura 9: Especificação do Use Case Propor agendamento de um serviço.

Depois de consultar a lista dos pedidos efetuados pelos Clientes, o Prestador pode selecionar o pedido que lhe parecer que vai mais ao encontro das suas capacidades e da sua disponibilidade. Desta forma, tem a oportunidade de visualizar mais informações acerca deste pedido de serviço. Caso tenha intenção de propor um agendamento, deve inserir a hora de início desejada e o preço/hora que requer. Assim, a proposta é registada e é enviada uma notificação ao Cliente.

Se o Prestador inseriu uma hora de início fora do intervalo autorizado, é apresentada uma mensagem de erro e este deve inserir uma nova hora.

2.4 Modelos de Tarefas

Decidimos ainda completar alguns Use Cases com a elaboração do respetivo Modelo de Tarefas, para desta forma ser possível visualizar mais detalhadamente as tarefas a realizar, bem como a entidade que o deverá fazer (Utilizador ou Sistema) e a ordem pela qual deverá ser efetuada.

2.4.1 Publicar um pedido de serviço

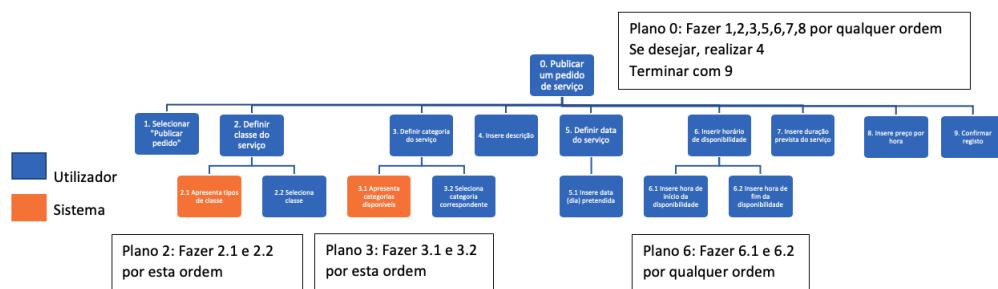


Figura 10: Modelo de Tarefas: Publicar um pedido de serviço.

Para publicar um pedido de serviço, o ator Cliente começa por selecionar a opção de "Publicar pedido" presente na interface gráfica, e, em seguida, deve selecionar todas as opções necessárias para que, no final, possa confirmar o registo do pedido.

2.4.2 Aceitar proposta de agendamento

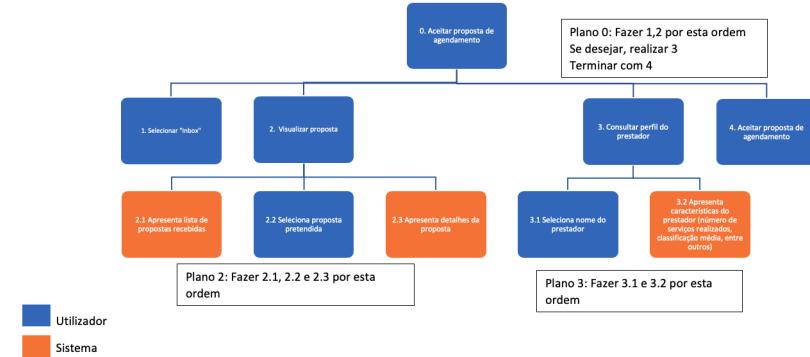


Figura 11: Modelo de Tarefas: Aceitar proposta de agendamento.

Para aceitar uma proposta de agendamento, o Cliente deverá consultar a sua *inbox* e aceitar ou recusar a mesma. Tem ainda a possibilidade de consultar o perfil do prestador, caso pretenda.

2.4.3 propor agendamento de um serviço

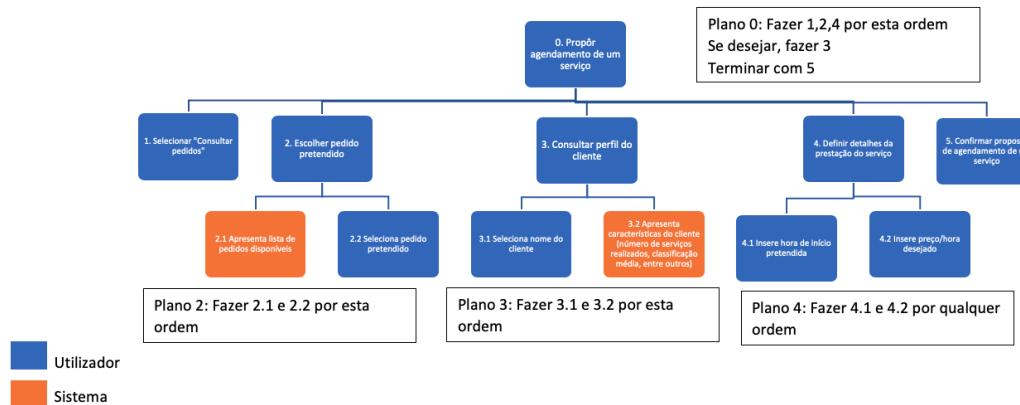


Figura 12: Modelo de Tarefas: propor agendamento de um serviço.

O ator Prestador de Serviços pode propor o agendamento de um serviço. Para tal, deve começar por visualizar a lista dos pedidos publicados pelos Clientes e, caso tenha interesse em algum deles, pode apresentar uma proposta, indicando a hora de início pretendida e o preço/hora desejado. Pode ainda consultar o perfil do cliente, caso pretenda.

2.5 Diagrama de Classes

A arquitetura do sistema começa o seu desenvolvimento com um diagrama conceptual, que tem o objetivo de nos mostrar como os diferentes componentes do sistema se vão relacionar entre si, como cada entidade será tratada e gerida. Uma vez que seria demasiado extensivo mostrar todas as fase da arquitetura do sistema, a decisão foi que o melhor seria mostrar o primeiro diagrama e o último, de maneira a ver-se a transformação e explicar um pouco das fases de aperfeiçoamento.

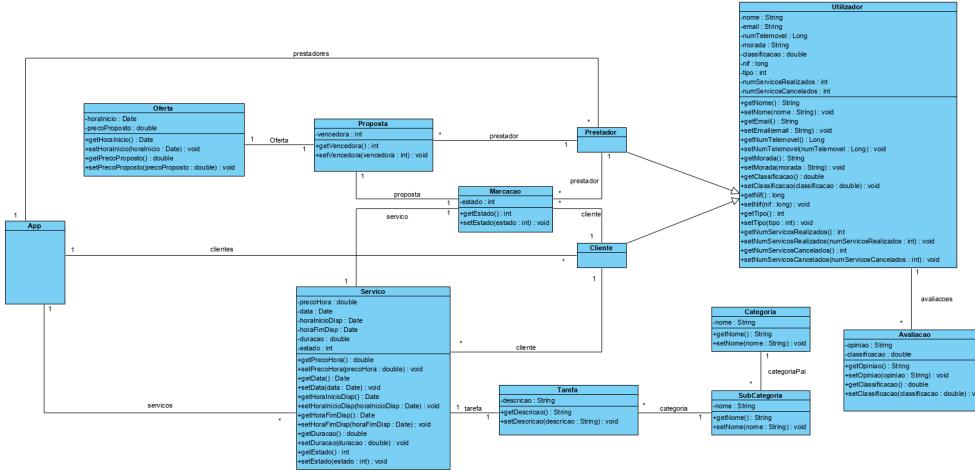


Figura 13: Diagrama Conceptual de Classes.

A Fig. 13, mostra a maneira como o sistema foi idealizado, de maneira a responder ao *flow* anteriormente descrito. A primeira alteração a fazer foi alterar a nomenclatura das operações uma vez que estas estavam confusas e, de certo modo, contra a 'ordem natural' do processo. Nesse seguimento, as alterações foram as seguintes:

- **Serviço** → **Pedido** (e.g., 'Um cliente publica um serviço pedido');
- **Marcação** → **Serviço** (e.g., 'Cliente x tem um(a) marcação serviço agendado para amanhã');
- **Categoria** → **Classe** e **Subcategoria** → **Categoria** (e.g., 'Dentro da categoria classe Jardinagem temos a subcategoria categoria limpeza de jardins').

Na fase seguinte, foi feita uma 'limpeza' no diagrama. As entidades que, do nosso ponto de vista, eram desnecessárias foram removidas. Com 'desnecessárias' queremos dizer aquelas que não precisam de ter uma representação própria. Realizamos assim um *merge* das variáveis destas classes com a sua classe 'mãe'. Foram removidas assim as seguintes classes:

- **Tarefa:** Adicionando a **Descrição** e a **Categoria** ao *Pedido*;
- **Oferta:** Movendo a **hora de inicio** e o **preço** para a *Proposta*.

Uma vez compactado o diagrama, realizamos a divisão das classes por *packages* de maneira a fazer uma montagem organizada da arquitetura. Desta análise resultaram os *packages*:

- **Utilizador:** *Package* onde estarão as classes relativas ao utilizador (**Utilizador** e **Avaliação**);
- **Serviço:** As classes **Pedido**, **Proposta**, **Serviço** ficam neste *package*;
- **Categoria:** Criado de maneira a 'armazenar' as classes relativas as categorias de serviço, **Classe** e **Categoria**.

Nesta fase, com o diagrama de classes **PIM** estabilizado, é momento de iniciar o processo de construção de **PSM**. O processo de escolha de tecnologia será detalhado posteriormente. Uma vez escolhida a tecnologia, é necessária transformar o diagrama de classes num diagrama mais específico e fiel à implementação.

Por questões tecnológicas, foi necessário fazer algumas alterações ao diagrama de acordo a ir de encontro à nossa ideia de classes a manipular e a persistir, por conseguinte as classes *Utilizador* e *Avaliação*, foram **separadas** de acordo com a função do utilizador (cliente e prestador).

As classes foram anotadas de maneira a permitir uma melhor **compreensão** do diagrama e **geração automática** de código.

A última adição no diagrama, depois de escolhida a linguagem, foi o *package* **Beans** onde estarão as classes que farão as operações de negócio sobre a camada de dados.

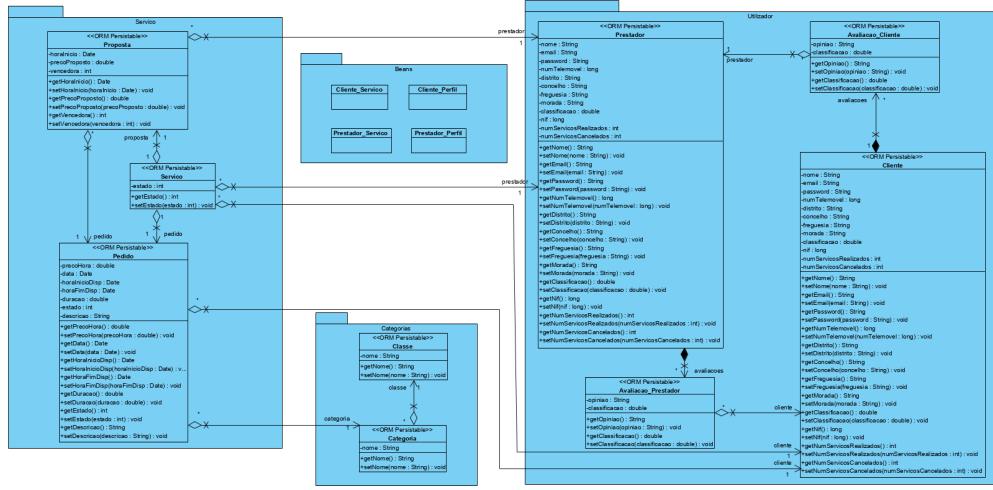


Figura 14: Diagrama de classes.

A Fig. 14 representa o diagrama final do sistema e que foi a sustenção de tudo o resto.

2.6 Protótipo da Interface

Na fase de modelação, elaboramos um protótipo de **baixa fidelidade** da interface gráfica da plataforma Serve.Me, com recurso à aplicação Pencil¹.

Desta forma, conseguimos ter uma noção de como as informações e as funcionalidades seriam apresentadas aos utilizadores, começando assim logo nesta fase a iteração e a procura da melhoria contínua da UI e UX para que, quando procedêssemos ao desenvolvimento efetivo da interface, pudéssemos obter uma interface gráfica com o máximo grau de usabilidade possível.

Em seguida, encontram-se apresentadas as *mockups* elaboradas.

2.6.1 Página Inicial

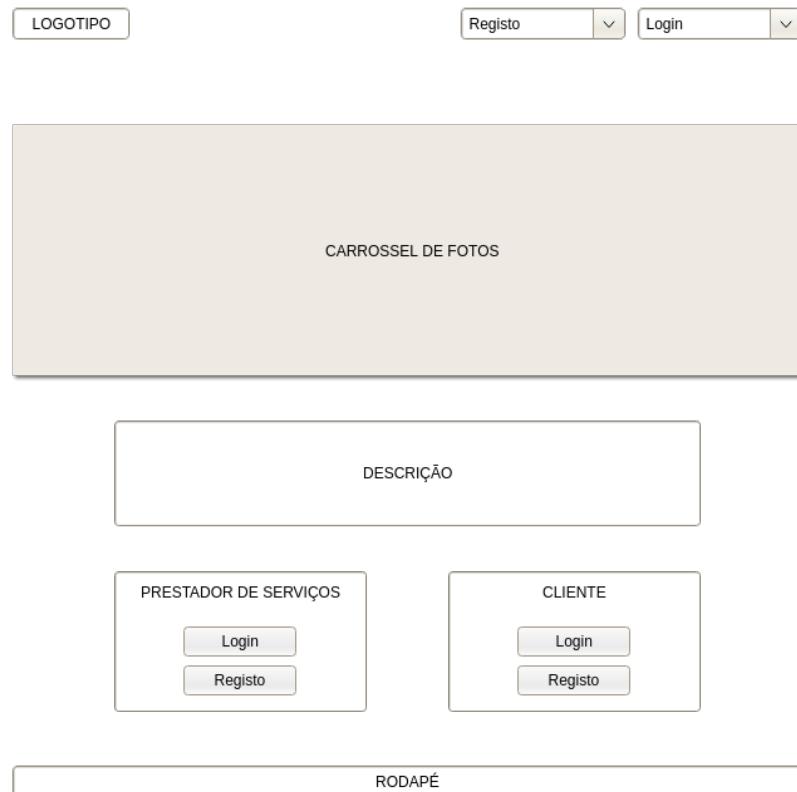


Figura 15: *Mockups*: Página Inicial.

¹Pencil: <https://pencil.evolus.vn/>

2.6.2 Página Inicial - Login

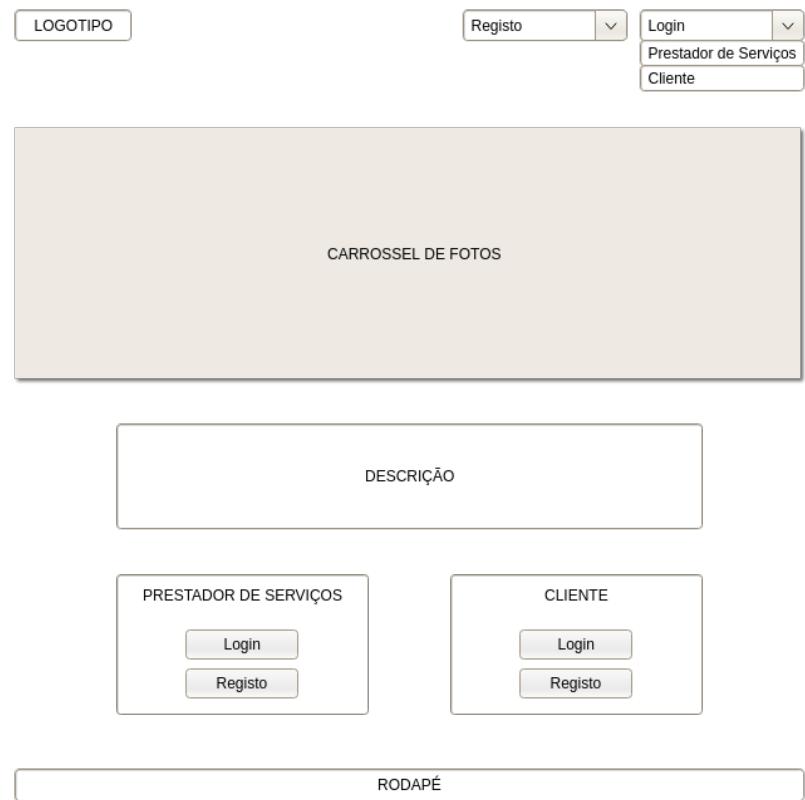


Figura 16: *Mockups*: Página Inicial (Login).

2.6.3 Página Inicial - Registro

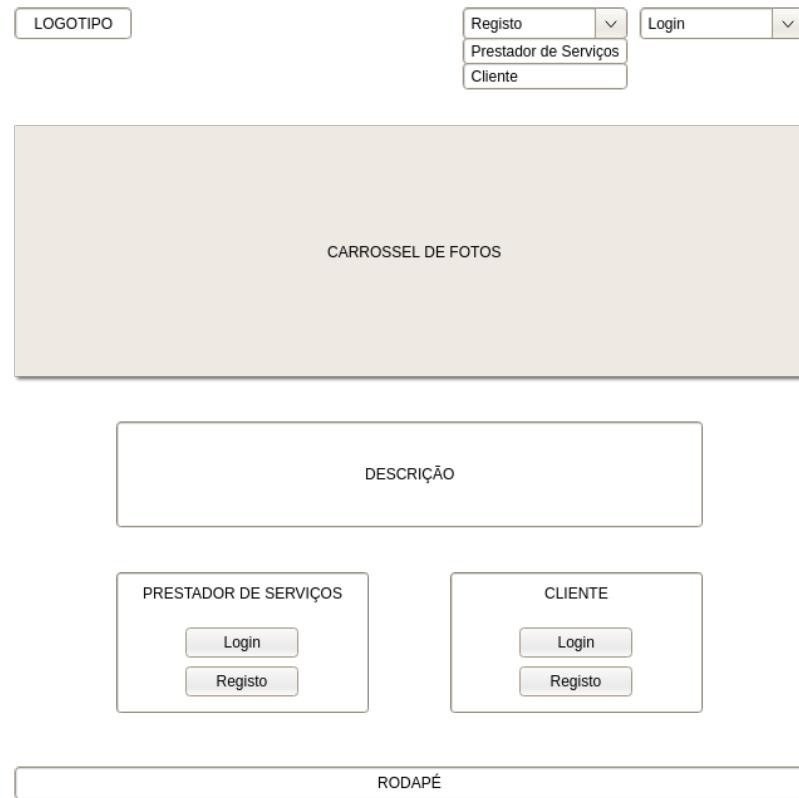


Figura 17: *Mockups*: Página Inicial (Registro).

2.6.4 Login Cliente

The mockup illustrates a client login interface. At the top left is a placeholder for a logo. To its right are two buttons: "Registro" and "Login", each accompanied by a dropdown arrow. Below these buttons is the text "Login como Cliente". Underneath this heading are two input fields: one labeled "E-mail" and another labeled "Password", both represented by empty rectangular boxes. To the right of the "Password" field is a large, empty rectangular area. At the bottom right of the form is a button labeled "OK".

Figura 18: *Mockups*: Login Cliente.

2.6.5 Registo Cliente

The mockup shows a user interface for client registration. At the top left is a placeholder 'LOGOTIPO'. To the right are two dropdown menus: 'Registo' and 'Login'. Below this is a section titled 'Registo como Cliente' containing several input fields:

- Nome: A single-line text input field.
- E-mail: A single-line text input field.
- Password: A single-line text input field.
- N.º Contribuinte: A single-line text input field.
- N.º de Telemóvel: A single-line text input field.
- Morada: A single-line text input field.
- Freguesia: A single-line text input field.
- Concelho: A single-line text input field.
- Distrito: A single-line text input field.

A 'Registrar' button is located at the bottom right of the form area. Below the form is a horizontal bar labeled 'RODAPÉ'.

Figura 19: *Mockups*: Registo Cliente.

2.6.6 Login Prestador

The mockup shows a user interface for logging in as a service provider. At the top left is a placeholder for a logo. To its right are two buttons: 'Registo' and 'Login', each with a dropdown arrow icon. Below this is a section titled 'Login como Prestador de Serviços'. It contains two input fields: one for 'E-mail' and one for 'Password'. To the right of these fields is a large 'OK' button. At the bottom of the page is a horizontal bar labeled 'RODAPÉ'.

LOGOTIPO

Registo ▾ Login ▾

Login como Prestador de Serviços

E-mail

Password

OK

RODAPÉ

Figura 20: *Mockups*: Login Prestador.

2.6.7 Registo Prestador

The mockup shows a registration form for service providers. At the top right are 'Registo' and 'Login' buttons. The form fields include: Nome (Name), E-mail (Email), Password (Password), N.º Contribuinte (Tax ID), N.º de Telemóvel (Mobile Number), Morada (Address), Freguesia (Parish), Concelho (Municipality), Distrito (District), and a 'Registrar' (Register) button. A 'RODAPÉ' (Footnote) section is at the bottom.

LOGOTIPO

Registo Login

Registo como Prestador de Serviços

Nome

E-mail

Password

N.º Contribuinte

N.º de Telemóvel

Morada

Freguesia

Concelho

Distrito

Registrar

RODAPÉ

Figura 21: *Mockups*: Registo Prestador.

2.6.8 Cliente - Serviços Agendados

The mockup shows a user interface for managing scheduled services. At the top, there is a header bar with a logo placeholder labeled 'LOGOTIPO' and a dropdown menu titled 'Os meus serviços' containing options: 'Agendados', 'Publicados', and 'Histórico'. To the right of the dropdown are three buttons: 'Inbox', 'Publicar serviço', and 'Perfil'. Below the header is a section titled 'Serviços Agendados' which contains a table with columns: Categoria, Subcategoria, Descrição, Prestador, Data, Hora Início, Duração, Preço/hora, and two buttons: 'Finalizar' and 'Cancelar'. The table has four rows, with the last row being significantly taller than the others. At the bottom of the page is a horizontal footer bar with the word 'RODAPÉ'.

Figura 22: *Mockups*: Cliente - Serviços Agendados.

2.6.9 Cliente - Serviços Publicados

The mockup shows a user interface for managing published services. At the top, there is a navigation bar with a logo placeholder, a dropdown menu labeled 'Os meus serviços', and links for 'Inbox', 'Publicar serviço', and 'Perfil'. Below the navigation is a section titled 'Serviços Publicados' containing a table. The table has columns for Categoría, Subcategoria, Descrição, Data, H. Início, H. Fim, Duração, Preço/hora, Estado, and actions (Editar, Cancelar). The table rows represent service status: Ativo, Pendente, and Expirado. A large empty space follows the table, and at the bottom right, there is a footer area labeled 'RODAPÉ'.

Categoría	Subcategoria	Descrição	Data	H. Início	H. Fim	Duração	Preço/hora	Estado	
								Ativo	Editar Cancelar
								Pendente	Editar Cancelar
								Expirado	Editar Cancelar

Figura 23: *Mockups*: Cliente - Serviços Publicados.

2.6.10 Cliente - Histórico

LOGOTIPO Os meus serviços Inbox Publicar serviço Perfil

Histórico de Serviços

Categoría	Subcategoria	Descrição	Prestador	Data	Hora Início	Duração	Preço/hora	Estado
								Realizado
								Cancelado
								Por classificar

RODAPÉ

Figura 24: *Mockups*: Cliente - Histórico.

2.6.11 Cliente - Inbox

The mockup shows a top navigation bar with five items: 'LOGOTIPO' (highlighted in blue), 'Os meus serviços' (with a dropdown arrow), 'Inbox' (highlighted in blue), 'Publicar serviço', and 'Perfil'. Below this is a section titled 'Inbox'. A table is displayed with columns: Categoría, Subcategoria, Descrição, Prestador, Data, H. Inicio proposta, Duração, Preço/hora proposto, and Informação. The 'Informação' column contains two buttons: 'Proposta de Agendamento' and 'Aceitar' (highlighted in blue) or 'Recusar'. The table has four rows, with the first row being the header. At the bottom is a horizontal bar labeled 'RODAPÉ'.

Figura 25: *Mockups*: Cliente - Inbox.

2.6.12 Cliente - Publicar pedido

The mockup shows a user interface for posting a service request. At the top, there is a navigation bar with a logo placeholder, a dropdown menu for 'Os meus serviços', and links for 'Inbox', 'Publicar serviço', and 'Perfil'. Below the navigation, the main form area has a title 'Publicar Serviço'. It contains several input fields: 'Categoria' (Category) with a dropdown menu, 'Subcategoria' (Subcategory) with a dropdown menu, 'Descrição' (Description) with a text input field, 'Data' (Date) with a date input field, 'Hora inicial de disponibilidade' (Initial availability time) with a time input field, 'Hora fim de disponibilidade' (End availability time) with a time input field, 'Duração' (Duration) with a time input field, and 'Preço/hora' (Price per hour) with a text input field. A large 'Publicar' (Post) button is located at the bottom right of the form.

LOGOTIPO Os meus serviços Inbox Publicar serviço Perfil

Publicar Serviço

Categoria:

Subcategoria:

Descrição:

Data:

Hora inicial de disponibilidade:

Hora fim de disponibilidade:

Duração:

Preço/hora:

RODAPÉ

Figura 26: *Mockups*: Cliente - Publicar pedido.

2.6.13 Cliente - Ver Perfil

The mockup shows a user interface for viewing a profile. At the top, there is a navigation bar with a 'LOGOTIPO' button, a dropdown menu labeled 'Os meus serviços', and three buttons: 'Inbox', 'Publicar serviço', and 'Perfil'. Below the navigation is a section titled 'Perfil' containing various input fields and buttons. The fields include:

- Nome (Name): An input field.
- E-mail (Email): An input field.
- Password (Password): An input field.
- N.º Contribuinte (Taxpayer Number): An input field.
- N.º de Telemóvel (Mobile Number): An input field.
- Morada (Address): An input field.
- Freguesia (Parish): A dropdown menu.
- Concelho (Municipality): A dropdown menu.
- Distrito (District): A dropdown menu.
- Buttons: 'Editar' (Edit), 'Guardar' (Save), and 'Terminar Sessão' (End Session).

At the bottom of the page is a horizontal bar labeled 'RODAPÉ' (Footnote).

Figura 27: *Mockups: Cliente - Ver Perfil.*

2.6.14 Cliente - Ver Perfil Prestador

The mockup shows a user interface for viewing a service provider's profile. At the top, there is a navigation bar with a logo placeholder, a dropdown menu labeled "Os meus serviços", and links for "Inbox", "Publicar serviço", and "Perfil". Below the navigation, the title "Perfil do Prestador de Serviços" is displayed. The form contains several input fields for personal information: "Nome" (Name), "E-mail" (Email), "N.º de Telemóvel" (Mobile Number), "Freguesia" (Parish) and "Concelho" (Municipality), "Distrito" (District), "Classificação média" (Average Rating), and two numerical fields for "N.º serviços realizados" (Number of services performed) and "N.º serviços cancelados" (Number of services canceled). There is also a section for "Comentários" (Comments) with two text input fields, each preceded by "<

LOGOTIPO

Os meus serviços ▾

Inbox

Publicar serviço

Perfil

Perfil do Prestador de Serviços

Nome:

E-mail:

N.º de Telemóvel:

Freguesia: Concelho:

Distrito:

Classificação média:

N.º serviços realizados:

N.º serviços cancelados:

Comentários:

<<Nome Cliente>>

<<Nome Cliente>>

RODAPÉ

Figura 28: *Mockups*: Cliente - Ver Perfil Prestador.

2.6.15 Cliente - Classificar serviço

The mockup shows a user interface for classifying a service. At the top, there is a navigation bar with a logo placeholder, a dropdown menu labeled "Os meus serviços", and three buttons: "Inbox", "Publicar serviço", and "Perfil". Below the navigation bar is a large rectangular form area with a thin black border. Inside this area, the title "Classificação de Serviço" is centered at the top. The form contains several input fields and labels:

- Categoria: XXXXXXXXXXXXXXXXXXXX
- Subcategoria: XXXXXXXXXXXX
- Descrição: XXXXXXXXXXXX
- Nome do Prestador: XXXXXXXXXXXX
- Local: XXXXXXXXXXXX
- Data: XX/XX/XXXX
- Hora Início: XXXX
- Duração: XXXX
- Preço/hora: XXXX
- Classificação:
- Comentários:

A small "Enviar" button is located at the bottom right of the form area. Below the form area is a horizontal bar with the text "RODAPÉ" centered.

Figura 29: *Mockups*: Cliente - Classificar serviço.

2.6.16 Prestador - Serviços Agendados

LOGOTIPO

Dashboard ▾
Serviços Agendados
Histórico
Estatísticas

Inbox Consultar Serviços Perfil

Horário Semanal

Hora	Segunda	Terça	Quarta	Quinta	Sexta	Sábado	Domingo
09:00							
11:00							
13:00							
15:00							
17:00							
19:00							
21:00							

Serviços Agendados

Categoria	Subcategoria	Descrição	Cliente	Data	Hora Início	Duração	Preço/hora	Local	
									Finalizar Cancelar

RODAPÉ

Figura 30: *Mockups*: Prestador - Serviços Agendados.

2.6.17 Prestador - Histórico

LOGOTIPO

Dashboard ▾

Inbox

Consultar Serviços

Perfil

Histórico de Serviços

Categoría	Subcategoria	Descrição	Cliente	Data	Hora Início	Duração	Preço/hora	Estado
								Realizado
								Cancelado
								Por classificar

RODAPÉ

Figura 31: *Mockups*: Prestador - Histórico.

2.6.18 Prestador - Estatísticas

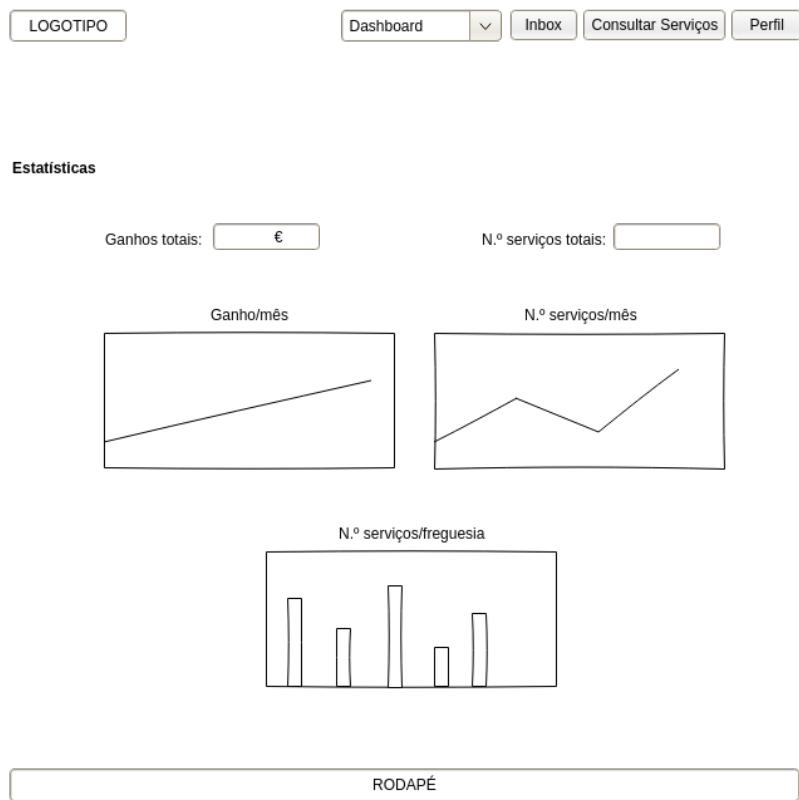


Figura 32: *Mockups*: Prestador - Estatísticas.

2.6.19 Prestador - Inbox

The mockup displays a user interface for a 'Prestador' (Provider) account. At the top, there is a header bar with the following elements from left to right: a placeholder for 'LOGOTIPO' (Logo), a dropdown menu labeled 'Dashboard' with a downward arrow, a button labeled 'Inbox' (highlighted in orange), a button labeled 'Consultar Serviços' (Services Search), and a button labeled 'Perfil' (Profile). Below the header, the word 'Inbox' is centered above a table. The table has ten columns: Categoría, Subcategoria, Descripción, Cliente, Data, H. Inicio proposta, Duração, Preço/hora proposto, Informação, and two additional columns at the end. The last two columns contain buttons labeled 'Pendente' (Pending), 'Cancelar' (Cancel), 'Aceite' (Accept), 'Rejeitado' (Rejected), and 'Aviso de Cancelamento' (Cancellation Notice). The table is currently empty.

Figura 33: *Mockups*: Prestador - Inbox.

2.6.20 Prestador - Consultar pedidos

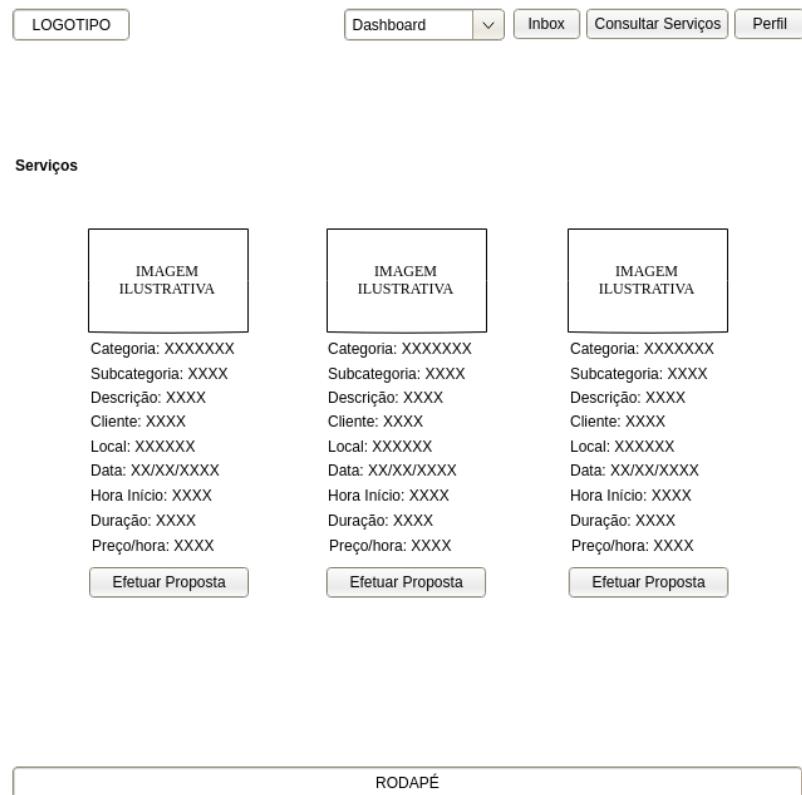


Figura 34: *Mockups*: Prestador - Consultar pedidos.

2.6.21 Prestador - Pedido de Agendamento

The mockup shows a user interface for a service provider. At the top, there is a navigation bar with a logo placeholder labeled "LOGOTIPO" and menu items: "Dashboard" (with a dropdown arrow), "Inbox", "Consultar Serviços", and "Perfil". Below the navigation is a large central box titled "Pedido de Agendamento". Inside this box, there are several input fields and labels:

- Categoria: XXXXXXXXXXXXXXXXXXXX
- Subcategoria: XXXXXXXXXXXX
- Descrição: XXXXXXXXXXXX
- Nome do Cliente: XXXXXXXXXXXX
- Local: XXXXXXXXXXXX
- Data: XX/XX/XXXX
- Duração: XXXX
- Hora Início: h : min
- Preço/hora: €/hora

At the bottom right of the central box is a "Enviar" button.

At the very bottom of the page is a horizontal footer bar labeled "RODAPÉ".

Figura 35: *Mockups*: Prestador - Pedido de Agendamento.

2.6.22 Prestador - Ver Perfil

The mockup shows a user interface for editing a provider's profile. At the top, there is a header bar with a 'LOGOTIPO' button, a dropdown menu labeled 'Dashboard' with a downward arrow, and three other buttons: 'Inbox', 'Consultar Serviços', and 'Perfil'. Below the header, the page title 'Perfil' is displayed. The form fields are organized into several rows:

- Nome:** An input field for the provider's name.
- E-mail:** An input field for the provider's email address.
- Password:** An input field for the provider's password.
- N.º Contribuinte:** An input field for the provider's taxpayer number.
- N.º de Telemóvel:** An input field for the provider's mobile phone number.
- Morada:** An input field for the provider's address.
- Freguesia:** A dropdown or input field for the provider's parish.
- Concelho:** A dropdown or input field for the provider's municipality.
- Distrito:** A dropdown or input field for the provider's district.

At the bottom of the form are three buttons: 'Editar' (Edit), 'Guardar' (Save), and 'Terminar Sessão' (End Session). Below the form is a horizontal bar labeled 'RODAPÉ' (Footnote).

Figura 36: *Mockups*: Prestador - Ver Perfil.

2.6.23 Prestador - Ver Perfil Cliente

The mockup displays a user interface for viewing a client profile. At the top, there is a header bar with a 'LOGOTIPO' button on the left and navigation buttons for 'Dashboard', 'Inbox', 'Consultar Serviços', and 'Perfil' on the right. Below the header, the title 'Perfil do Cliente' is centered. The main content area contains several input fields for client information: 'Nome' (Name), 'E-mail' (Email), 'N.º de Telemóvel' (Mobile Number), 'Freguesia' (Parish) and 'Concelho' (Municipality) side-by-side, 'Distrito' (District), 'Classificação média' (Average Rating), 'N.º serviços realizados' (Number of services performed), and 'N.º serviços cancelados' (Number of services canceled). Below these fields is a section for 'Comentários' (Comments) with two large input fields labeled '<<Nome Prestador>>' (Prestator's name). At the bottom, a horizontal bar contains the word 'RODAPÉ' (Footnote).

Figura 37: *Mockups*: Prestador - Ver Perfil Cliente.

2.6.24 Prestador - Classificar serviço

The mockup shows a user interface for classifying a service. At the top, there is a header with a logo placeholder and navigation links: Dashboard, Inbox, Consultar Serviços, and Perfil. Below this is a large central box labeled "Classificação de Serviço". Inside this box, there are several input fields and labels:

- Categoria: XXXXXXXXXXXXXXXXXXXX
- Subcategoria: XXXXXXXXXXXX
- Descrição: XXXXXXXXXXXX
- Nome do Cliente: XXXXXXXXXXXX
- Local: XXXXXXXXXXXX
- Data: XX/XX/XXXX
- Hora Início: XXXX
- Duração: XXXX
- Preço/hora: XXXX
- Classificação:
- Comentários:

At the bottom right of the central box is a "Enviar" button.

At the very bottom of the page is a horizontal bar labeled "RODAPÉ".

Figura 38: *Mockups*: Prestador - Classificar serviço.

3 Implementação

Chegando a um *plateau* da fase de Modelação, é tempo de voltar o foco para o desenvolvimento, que se inicia com a escolha das tecnologias mais indicadas para desenvolver o que foi anteriormente idealizado. Nesta secção, vamos apresentar o que foi a nossa fase de desenvolvimento. Começamos por expor umas reflexões acerca do **Estado de Arte** no que diz respeito a *frameworks* e linguagens de programação.

3.1 Considerações Iniciais

A pesquisa iniciou-se pela procura de uma linguagem estável, robusta e utilizada na 'indústria'. A escolha recaiu pelo **Java** que, de acordo com a comunidade, continua a ser utilizada por grande parte do mercado de *WebApps*. De maneira a facilitar o processo de desenvolvimento de uma aplicação é frequente o uso de *frameworks*, estas 'peças' de *software* realçam a reutilização de componentes. Depois de assinalar as *frameworks* que nos pareceram mais úteis, iniciou-se o momento de comparação das mesmas. Comparamos fatores como **importância no mercado, características diferenciadoras, curva de aprendizagem e dimensão da comunidade de suporte**. Assim, inicialmente, a *framework Spring* dominava as camadas de apresentação e negócio. Já a camada dos dados seria controlada pela líder na área, **Hibernate**. Esta seria uma arquitetura designada **Server-side** o que significa que todo o processamento é feito no servidor. No entanto, esse tipo de arquitetura não é o mais recomendado pela comunidade no que diz respeito a *WebApps*. Utilizar arquitetura **Client-side** torna a aplicação mais interativa e responsiva uma vez que não necessita de estar constantemente a fazer pedidos ao servidor. Uma vez que a experiência do utilizador era um fator de extrema importância decidimos procurar ferramentas que nos ajudassem nesse objetivo. As *frameworks* client-side mais utilizadas no mercado são: **React, Angular e Vue.js**, contudo as primeiras duas possuem uma curva de aprendizagem relativamente maior que o desejado. Fator este que nos leva a escolher **Vue.js**.

Assim a arquitetura a ser implementada é a seguinte:

- **Vue.js**: Para a camada de apresentação;
- **Spring Boot**: Para a lógica de negócio;
- **Hibernate**: Na camada dos dados.

Nas seguintes secções será abordado em mais detalhe o processo de desenvolvimento.

3.2 Frontend

Para o desenvolvimento da camada de apresentação foi utilizada a *framework* **Vue.js**. Esta camada é totalmente independente da camada de negócio e da camada de dados, sendo possível obter as informações desejadas através de pedidos à API criada para o efeito.

A estrutura deste módulo foi assim construída através do comando “vue create”. Para tornar o resultado da interface gráfica mais apelativo, recorremos ao **BootstrapVue**. Este *package* permite a utilização de todos os componentes (*grid system*, entre outros) de **Bootstrap**, que é a framework mais popular de CSS.

Ao longo do desenvolvimento do trabalho, foi surgindo a necessidade de instalar outros *packages* que vieram melhorar o resultado final, seja ao nível da interface gráfica em si, seja ao nível da modularidade do código. São exemplos disso o **mdbvue**, para a criação de gráficos de linhas, barras e circulares, o **vue-cal** para a apresentação do calendário de serviços agendados e o **vue-router** para a criação das rotas da plataforma.

3.2.1 Views

Foram criados dois componentes, transversais aos dois tipos de utilizadores da plataforma, nomeadamente a **Navbar** e o **Footer**. O Footer é sempre o mesmo, independente do utilizador, já a Navbar pode variar entre as seguintes características:

1. Nenhum utilizador loggado;
2. Cliente loggado;
3. Prestador loggado;

O conteúdo propriamente dito das páginas *web* poderá corresponder a um dos quatro tipos:

1. *Landing page* quando nenhum utilizador está loggado;
2. Páginas *web* das funcionalidades do Cliente loggado;
3. Páginas *web* das funcionalidades do Prestador loggado;
4. 404: *page not found*.

3.2.2 Pedidos à API

Para efetuar os pedidos à API recorremos ao *package* **axios**, que nos permite definir o *endpoint* requerido, configurar os *headers*, o tipo de método a utilizar (POST, GET,...) e receber assim a resposta (dados) por parte da API.

3.3 Backend

Para o backend, idealizamos uma **Rest API** que irá satisfazer os pedidos do frontend. Escolhemos esta arquitetura por facilitar a independência dos componentes sem perder a facilidade de acesso à lógica de negócio. Desse modo, utilizamos a *framework* **Spring Boot** com a construção de **Rest Controllers** de maneira a fazer o mapeamento do pedido e assim obter a resposta adequada.

3.3.1 Controllers

Em **Spring** os *HTTP Requests* são manipulados por **Rest Controllers**. A divisão que achamos mais adequada foi a separação por função no sistema, isto é, foram criados *controllers* que reúnem as operações idênticas. Assim foram criados os seguintes *controllers*:

- **Register:** Terá os mapeamentos relacionados com o registo de utilizadores (Clientes e Prestadores);
- **Login:** Como o nome indica, tratará os pedidos de *login* dos utilizadores;
- **Profile:** Todos os *requests* que tenham a ver com o perfil dos utilizadores, são manipulados por este controlador;
- **Services:** Para tratar tudo o que esteja relacionado com o serviço fornecido pela *App* (i.e. **pedidos**, **propostas** e **serviços**), é utilizado este *controller*;
- **Inbox:** As operações relacionadas com as notificações são mapeadas por este controlador;
- **Rating:** *Controller* responsável por responder às avaliações;
- **Stats:** Controlador que mapeia os pedidos por estatísticas.

De maneira a prevenir o uso indevido da API, todos os mapeamentos (com exceção do login e registo, por razões óbvias) são **filtrados** com o objetivo de perceber se o pedido HTTP possui um **Token** e se este é certificado.

Depois de validar o *token*, o pedido HTTP pode assim ser 'enviado' para o respetivo *controller*. O controlador, por sua vez, invoca o **bean** necessário para satisfazer o *request*.

3.3.2 Beans

Os Java Beans são pequenos componentes de um sistema cuja função é a execução de tarefas para um cliente. Caracterizam-se pela sua portabilidade, segurança e rapidez.

Pelas vantagens destes componentes, decidimos que as operações de lógica de negócio seriam executadas por **Beans**. Procuramos fazer uma divisão de *beans* que fizesse sentido no contexto da aplicação.

Nesse seguimento, a divisão que achamos mais adequada e que foi implementada na aplicação é a seguinte:

- **Cliente-Perfil:** Classe de Beans que é invocada sempre que o cliente necessita de fazer alguma operação que envolva o seu perfil;

- **Prestador-Perfil:** Aplica-se o mesmo que a classe anterior, porém quando as operações dizem respeito ao Prestador;
- **Cliente-Servicos:** Neste classe estão, por exclusão de partes, todos as restantes operações. Serão assim, os procedimentos relacionados com a lógica de negócio da aplicação (e.g., Adicionar um pedido, aceitar propostas, etc.);
- **Prestador-Servicos:** O homólogo ao *Cliente-servicos* em relação aos prestadores. Inclui operações como: Ver os pedidos existentes, fazer proposta a um pedido, etc..

A grande maioria das operações de *beans* envolve acesso à Base de Dados. De seguida, vamos falar um pouco acerca a camada de dados.

3.3.3 Base de Dados

Nesta subdivisão será abordada a camada de dados e como é feito o acesso aos mesmos.

Como dito anteriormente, para facilitar a gestão desta camada utilizamos uma ORM, **Hibernate**. Esta ferramenta é a mais famosa do mercado na sua área, conhecida pelo seu **alto desempenho** e **confiabilidade**. O facto de ser bastante **completa** proporciona uma redução de trabalho na administração do acesso aos dados. Existem duas formas de configurar *Hibernate*: 1. Ficheiros de Configuração; 2. Através de Anotações.

Uma vez que existem ferramentas, como *Visual Paradigm*, que ajudam na tarefa de escrever os ficheiros de configuração, optamos por esta abordagem. O diagrama de classes foi anotado de maneira a indicar à ferramenta as entidades e ligações a **persistir**. Com o diagrama anotado, basta selecionar o sistema desejado, algumas configurações e a ferramenta é capaz de gerar os ficheiros.

No fim deste processo, todo acesso aos dados **MySQL** (sistema escolhido), é feito através de DAO's e gerido pelo **Hibernate**.

4 Deploy

No fim de construído o sistema, o projeto inicia a fase de *deploy*, com o objetivo de possibilitar a aplicação disponível para uso. Na presente secção, depois de um pequeno resumo da pesquisa pela comunidade, iremos explicar as nossas decisões, bem como uma explicação do processo necessário para colocar a aplicação no “ar”.

4.1 Considerações Iniciais

Com o intuito de providenciar uma instalação versátil, simples e rapidamente eficaz do ponto de vista da manutenção e configuração, o grupo decidiu recorrer a uma das ferramentas mais recomendadas pela comunidade do ponto de vista da instalação de componentes, o **Docker**. Esta plataforma tem como principal objetivo facilitar a criação e gestão de ambientes isolados com recurso à tecnologia de *containers*, que são basicamente micro-ambientes, isolados e bastante leves, podendo aprovisionar praticamente qualquer aplicação, sendo que este *container* pode posteriormente ser executado em qualquer máquina que tenha o Docker instalado.

Assim, o grupo decidiu enveredar pela incorporação de *containers* na nossa fase de aprovisionamento por todas as vantagens e facilidades que estes nos apresentam, entre as quais queremos destacar o facto de estes serem considerados *Lightweight* por não ser necessária a utilização de, por exemplo, um *Hypervisor*, que aumenta a carga computacional. O que resulta numa velocidade de inicialização superior e o maior desempenho em *runtime* pelo facto dos recursos serem da máquina nativa. Por outro lado, das mais variadas vantagens destacamos também a inexistência do sistema operativo, e por isso não existe necessidade de configurá-lo, poupano tempo.

Através desta decisão permitimos:

- **Isolamento:** Cada componente está completamente isolado dos outros em termos de **segurança, desempenho e falhas**;
- **Deployment descomplicado:** Sempre que existir a necessidade de adicionar um serviço (mais servidor de *backend*, base de dados, etc.) novo à aplicação, apenas é necessário criar um novo *Container* com esse serviço e inclui-lo na arquitetura;
- **Portabilidade:** Esta distribuição permite a migração para novos ambientes (Servers).

Por todas estas razões, a escolha de *containers* passou a ser uma realidade, sendo que o grupo planeou e concretizou a instalação dos vários componentes da aplicação em diferentes *container*. Para tal, criaram-se *Dockerfile* ou *docker-compose*, que são ficheiros que irão lançar esses mesmos *containers*.

A divisão, tendo em conta os vários componentes, realizou-se da seguinte forma:

- 1 *container* com o *frontend* da aplicação, uma vez ser independente do resto da aplicação;
- 1 *container* que irá **balancear** a carga proveniente da camada de apresentação;
- 2 *containers* com a parte lógica da aplicação, para poder lidar melhor com elevado número de pedidos;
- 1 *container* com a base de dados MySQL que alberga toda a informação necessária da aplicação.

De modo arquitetural, o *deployment* da nossa aplicação poderá ser representado pelo seguinte diagrama:

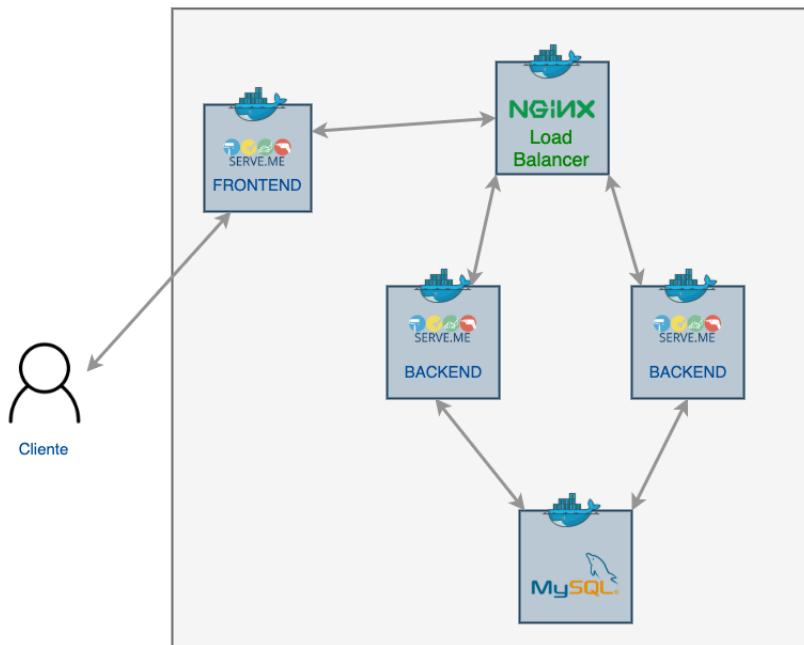


Figura 39: *Deploy*: Arquitetura.

4.2 Frontend

Como temos vindo a descrever, existem vários *containers* para albergar toda a nossa aplicação, de modo a permitir organizar e alojar todos os recursos necessários.

No que diz respeito ao *frontend*, o processo de containerização foi bastante simples. Como já explicamos na secção 3, decidimos enveredar pela construção da camada de apresentação recorrendo a **Vue.js**, e, como tal, o processo para passar o nosso projeto desenvolvido para o interior de um *container* foi relativamente simples, uma vez que a documentação da própria *framework* Vue se encontra explícita e ensina de forma rápida a integrar com Docker.

Assim, e tendo em conta que o nosso interesse como foi dito é gerar as imagens Docker para poderem ser reutilizadas onde quisermos, a imagem correspondente ao *frontend* é gerada através dos seguintes comandos:

```
FROM node:lts-alpine

# install simple http server for serving static content
RUN npm install -g http-server

# make the 'app' folder the current working directory
WORKDIR /app

# copy both 'package.json' and 'package-lock.json' (if available)
COPY package*.json .

# install project dependencies
RUN npm install

# copy project files and folders to the current working dir
COPY . .

# build app for production with minification
RUN npm run build

EXPOSE 8080

CMD [ "http-server", "dist" ]
```

Esta é uma *Dockerfile* bastante simples e foi inspirada por alguns exemplos que o grupo encontrou na documentação online, pelo que o ficheiro apenas irá a um repositório aceder a uma imagem remota para poder instalar e executar os comandos abaixo definidos.

4.3 Balanceador

Tal como descrito anteriormente e apresentado no diagrama arquitetural de *deploy*, o grupo fez uso de um balanceador de carga fornecido pelo NGINX. Este é um servidor de HTTP que nos permitiu reencaminhar os pedidos para os outros dois *containers* que contém o *backend* da aplicação a correr.

Dessa forma, o procedimento para criar o *container* é bastante simples, uma vez que já existe uma imagem na Docker Hub preparada para usufruirmos e configurarmos o NGINX da forma pretendida.

A Dockerfile resultante é a seguinte:

```
FROM nginx
COPY nginx.conf /etc/nginx/nginx.conf
```

No entanto, é o ficheiro de configuração do NGINX onde é necessário definir quais são os nodos para os quais o servidor irá reencaminhar os pedidos e a porta em que os irá fazer. Por outro lado, é aqui que definimos qual a porta que estará à escuta para realizar o reencaminhamento dos pedidos.

Por uma questão de organização e leveza de leitura, o ficheiro **nginx.conf** encontra-se anexado a este documento, onde estão detalhados os procedimentos de implementação do mesmo.

Basta por isso ressalvar a **política de balanceamento** utilizada, que salvo melhor alternativa, a que foi adoptada foi o reencaminhamento de 50% para a instância1 e 50% para a instância2, resultando assim numa divisão de **metade** dos **pedidos** para cada uma das instâncias.

4.4 Backend

Nesta subsecção iremos abordar então os procedimentos que realizamos para a implementação do *backend* da aplicação. Tal como mostrado anteriormente, são criados dois *containers* que irá colocar em modo de execução o ficheiro **.jar** da aplicação.

A Dockerfile correspondente é a seguinte:

```
FROM openjdk:8-jdk-alpine
ADD target/ServeMe-0.0.1-SNAPSHOT.jar ServeMe.jar
EXPOSE 8083
ENTRYPOINT [ "sh", "-c", "java $JAVA_OPTS
-Djava.security.egd=file:/dev/.urandom -jar /ServeMe.jar" ]
```

Assim sendo, para gerar a segunda instância será apenas necessário alterar a porta que estamos a realizar *expose* para permitir que cada uma das instâncias escute na sua respetiva porta.

Relativamente à **base de dados**, o grupo construiu um ficheiro que também permite a criação de uma imagem Docker, mas desta vez com recurso ao *docker compose*, uma vez que é mais fácil passar argumentos, como por exemplo a *password* de acesso e até mesmo o nome da base de dados.

O ficheiro apresenta a seguinte constituição:

```
services: '3'

mysql-serveme:
    image: mysql:8.0.17
    environment:
        MYSQL_ROOT_PASSWORD: serveme
        MYSQL_DATABASE: serveme
    ports:
        - "3308:3306"
```

Desta feita, a base de dados corresponde ao motor MySQL, pelo que a imagem irá ser criada com uma base de dados com o nome “serveme” e as respetivas credenciais de acesso também são definidas no ficheiro.

Assim, o grupo consegue ter imagens que são rapidamente criadas, clonadas e acima de tudo apresentam grandes vantagens, como foi enumerado anteriormente. Permitiu-nos assim realizar a divisão dos diversos componentes segundo a sua função e tipologia, permitindo-nos no futuro criar réplicas de qualquer um dos serviços de forma bastante simples, e de forma a que a sua integração seja rápida. Isto é, a possibilidade de tornar o sistema mais **robusto**, e **mais tolerante a falhas**, será um processo fácil e irá garantir maior disponibilidade do sistema.

5 Análise de Carga

5.1 Considerações gerais

Analisar a capacidade do *software* que produzimos é de extrema importância uma vez que assim podemos **anticipar futuros problemas, perceber real capacidade** do sistema, **corrigir problemas** antes do sistema estar disponível para a utilização de centenas, milhares ou milhões de pessoas.

Desta forma, e tendo em conta que alimentos, carros e os mais diversos produtos são sujeitos a testes antes do seu lançamento para o público, também esta prática deve ser aplicada ao desenvolvimento de aplicações. Por mais relevantes ou menos, mais ou menos perigosas, os testes irão mostrar-nos indicadores muitíssimo importantes no que toca ao trabalho desenvolvido, possíveis *bugs*, *bottlenecks*, etc..

Essencialmente, utilizamos os testes de carga porque irão permitir quantificar e avaliar algumas métricas gerais que devemos ter em conta quando produzimos *software*, sendo elas:

1. **Confiança:** O sistema é resistente a falhas durante a execução, isto é, não entra em *loop*, não interrompe a execução por falta de recursos, etc.;
2. **Funcionalidade:** O sistema apresenta um comportamento de acordo com o esperado e definido nos requisitos;
3. **Performance:** O sistema apresenta tempos de resposta adequados e aceitáveis, mesmo quando submetido a um volume de processamento próximo de situações reais ou de pico?

Com estas noções em mente, foi então que o grupo decidiu realizar testes de carga tendo em conta os dois principais atores do nosso sistema, em situações que de facto sejam custosas para o sistema, numa tentativa de perceber os seus limites em termos computacionais, bem como os resultados práticos da sua utilização (em termos de tempo de espera, por exemplo).

Para evitar qualquer tipo de enviesamento o sobrecarga adicional, o grupo executou os testes a partir de uma máquina diferente daquela que alojava todos os serviços da aplicação, de forma a obter os resultados mais fidedignos possível.

5.2 Testes de Carga

Para a realização dos testes de carga o grupo decidiu utilizar a ferramenta mais popular para o efeito, o **JMeter**. Esta é uma aplicação que permite simular a utilização simultânea do número de utilizadores que pretendemos, com a possibilidade de gerar e apresentar **gráficos, relatórios estatísticos**, entre outros.

Assim, esta ferramenta permite executar de forma sucessiva pedidos HTTP, permitindo a edição do seu conteúdo, tipologia, etc..

Com ajuda de uma extensão para esta ferramenta, o **BlazeMeter**, foi possível gravar sessões de utilização que simulem uma interacção normal de um utilizador apenas. Com recurso a esta ferramenta, foi possível agrupar os pedidos HTTP das sessões pretendidas, sendo que depois bastou realizar algumas alterações a nível do conteúdo dos pedidos para simular interações diferentes.

Neste sentido, o grupo focou-se em **duas interações**, uma para cada ator, que no nosso entender são particularmente custosas, sendo elas:

- **Cliente:** Registar, Autenticar, Ver Serviços Agendados e Publicar um Pedido de Serviço;
- **Prestador:** Registar, Autenticar, Ver Serviços Agendados, Consultar todos os serviços da aplicação e efetuar uma proposta a um deles.

5.2.1 Interação 1 - Cliente

Após gravada a interação, o grupo decidiu efetuar os testes de carga até uma situação de rutura ou neste caso de **erro**, isto é, quando o sistema atingiu a sua **capacidade máxima** para responder a pedidos simultaneamente.

Neste seguimento, a interação em termos de pedidos HTTP consistiu em:

- Pedido POST para registrar;
- Pedido POST para autenticar;
- Pedido GET aos serviços agendados;
- Pedido POST para publicar pedido.

De forma simultânea, o grupo executou pedidos com 100, 200, 300, 400 e 500 utilizadores, representados por esse mesmo número de *threads*, que originaram os seguintes resultados:

Nº de Threads	Throughput (/min)	Tempo Resposta (máx) (s)	Erro (%)
100	3,746.722	7.106	0.0
200	3,634.601	14.898	0.0
300	3,873.3	21.620	0.0
400	1,164.267	25.953	0.0
500	4,698.659	30.624	11.8

Como podemos ver pela observação da tabela, conforme vamos aumentando a carga o **débito** da nossa aplicação tem tendência a aumentar em termos gerais, pelo que também observamos alguma linearidade no que concerne ao **tempo de resposta**.

Note-se que o grupo utilizou uma **connection pool size** de **150**, uma vez que o valor predefinido pelo Hibernate é muito inferior a este. Com estes valores, conseguimos concluir o nosso ponto de ruptura, em termos de acessos concorrentes e simultâneos, se encontra entre as 400 e 500 pessoas.

Como forma de melhor visualizar o **impacto** que acessos em massa podem provocar no sistema, iremos agora comparar os gráficos da execução de 100 *threads* vs. 500 *threads*.

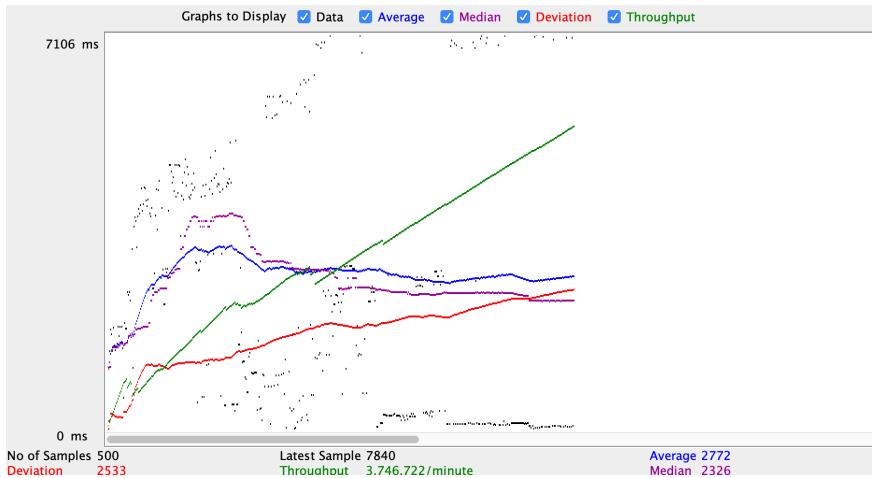


Figura 40: Testes de Carga: Gráfico resultante simulação com 100 *threads* para Cliente.

Como podemos ver pelo gráfico apresentado, todos os pedidos são atendidos e respondidos com particular brevidade, pelo que o **throughput** apresenta-se praticamente como uma reta linear, significando que o sistema, para a quantidade de pedidos que foi submetido, ainda tem uma capacidade de resposta longe do máximo.

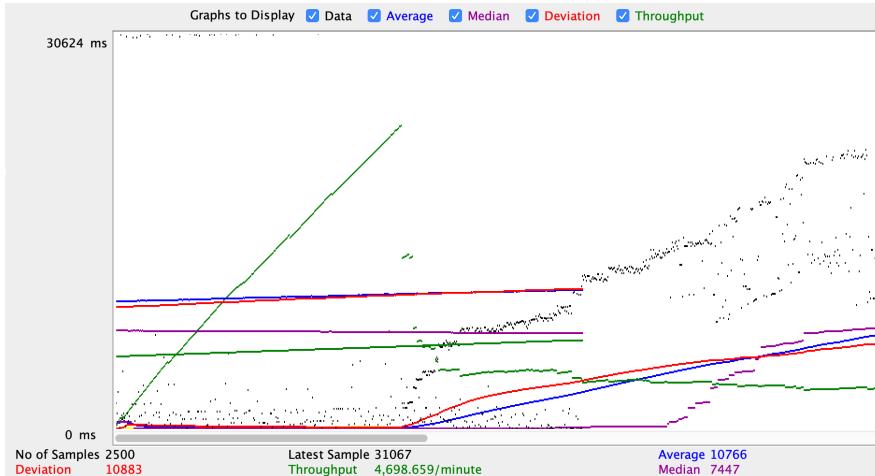


Figura 41: Testes de Carga: Gráfico resultante simulação com 500 *threads* para Cliente.

Em comparação, este gráfico com 500 *threads* é particularmente diferente do anterior, com particular ênfase aos efeitos dos pedidos em massa e também já se evidencia uma situação de **ruptura**, atingido-se já a **capacidade máxima** do sistema para **responder a todos os pedidos**.

Assim, como conseguimos perceber pela linha a verde (representa o débito), esta apresenta uma quebra, bem como outras linhas mais abaixo, significando

que em determinado momento o sistema **quebrou**, e atendeu menos pedidos de cada vez. Por consequência, os pontos mais a preto que representam a **troca de dados**, estendem-se mais para a frente, significando que esta troca foi mais demorada e dispersa. As várias quebras de débito e o facto do sistema ter atingido uma zona de **stress**, resultam nos **11%** de **erro** que verificamos neste teste.

5.2.2 Interação 2 - Prestador

De forma análoga para esta interação, foi gravada a interação, parecido ao modo como são feitos os testes de Selenium, e foi gerado o ficheiro que continha os pedidos e que seria utilizado no JMeter para proceder à simulação dos pedidos em massa.

No seu todo, a gravação reunia os seguintes pedidos:

- Pedido POST para registar;
- Pedido POST para autenticar;
- Pedido GET aos serviços agendados;
- Pedido GET a todos os pedidos publicados;
- Pedido POST para efectuar proposta.

Tendo em conta que nestes dois cenários tentamos proporcionar um ambiente real, foram registados cerca de **700 pedidos** de **serviço**, o mesmo número de utilizadores e algumas dezenas de **serviços agendados**.

Para este segundo cenário foram realizados testes com 100, 200, 300 e 400 *threads*. Os resultados são os que se seguem:

Nº de Threads	Throughput (/min)	Tempo Resposta (máx.) (s)	Erro (%)
100	2,886.697	9.604	0.0
200	3,289.774	16.887	0.0
300	2,768.166	34.265	0.0
400	1,164.267	37.982	4.42

Extraindo conclusões pela análise da tabela, percebemos que conforme vamos aumentando a carga o **débito** da nossa aplicação tem tendência a aumentar até certo ponto, mas que após determinada marca (cerca de 300 *threads*), esse valor começa a reduzir consecutivamente.

Relativamente ao *response time*, este vai aumentando de forma gradual, e tendo em conta o cenário de teste que realizamos com o Cliente, percebemos que este é mais custoso e demorado, uma vez que também são carregados todos os pedidos da base de dados, que para o cenário desta simulação, eram cerca de **700 pedidos**, tal como já havíamos mencionado anteriormente.

Como forma de perceber o **stress** criado na plataforma com estes testes, iremos exibir mais dois gráficos para uma breve análise.

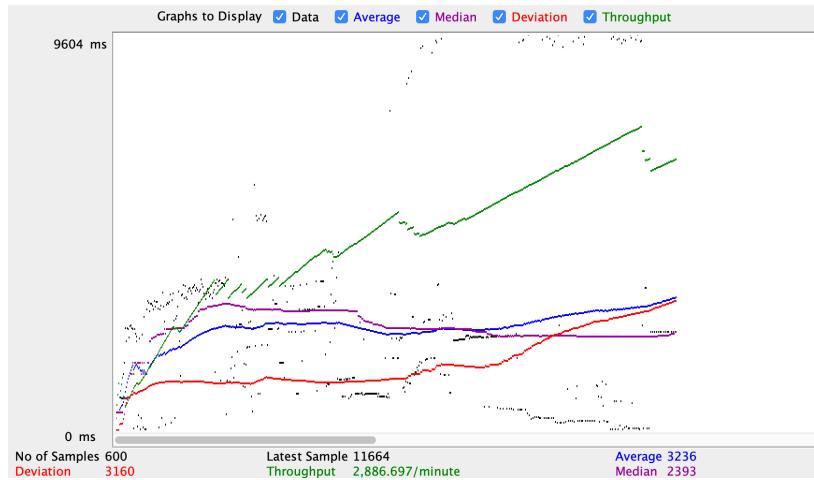


Figura 42: Testes de Carga: Gráfico resultante simulação com 100 *threads* para Prestador.

Tal como no cenário anterior, o primeiro gráfico que decidimos apresentar é relativo ao primeiro teste que realizamos para que este se torne uma **base** de **análise** para os restantes casos.

Dessa forma, a nível de **débito** vemos uma linha com sentido crescente, com um **tempo de resposta** também relativamente baixo para o tipo de interacção que estamos a tratar, bem como as características que já falamos.

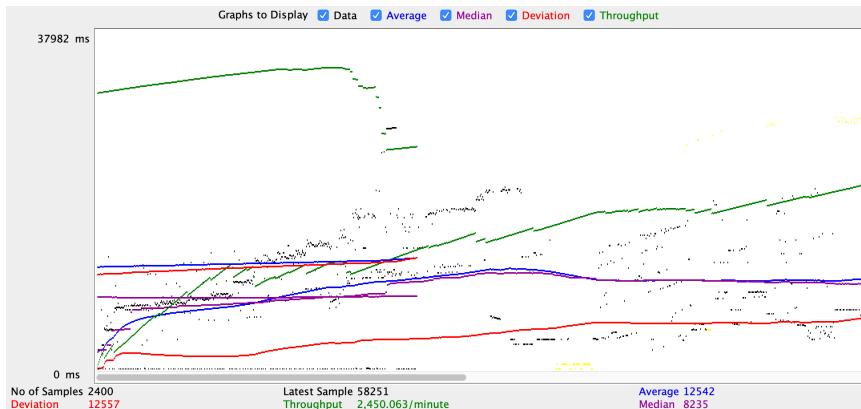


Figura 43: Testes de Carga: Gráfico resultante simulação com 400 *threads* para Prestador.

Por outro lado, neste gráfico está representado o caso onde o sistema começa a apresentar erro no teste, ou seja, atingiu uma situação de **stress** tão elevada que a sua capacidade de resposta máxima foi atingida. Isto é comprovado pela linha verde do *throughput* que sofre também uma quebra bastante grande, fazendo com que vários pedidos sofram atrasos no seu atendimento.

Conforme se vê pelos pontos a negro, estes também estão mais espalhados do que no gráfico anterior, mostrando o maior tempo que os pedidos demoram

a ser atendidos.

Ainda assim, o grupo encontra-se **satisfeito** com os resultados obtidos, uma vez que obteve capacidade de resposta por volta das 4 centenas de utilizadores a aceder de forma simultânea em ambos os cenários de teste.

6 Manual de Utilização

Antes de procedermos à análise da usabilidade da Serve.Me, começamos por apresentar o fluxo da plataforma e a evolução que a mesma sofreu desde o protótipo de baixa fidelidade (Secção 2.6) desenvolvido na primeira fase do projeto.

Começando pela página inicial, quando um utilizador não loggado entra na plataforma depara-se com a seguinte página *web*:

6.1 Página Inicial

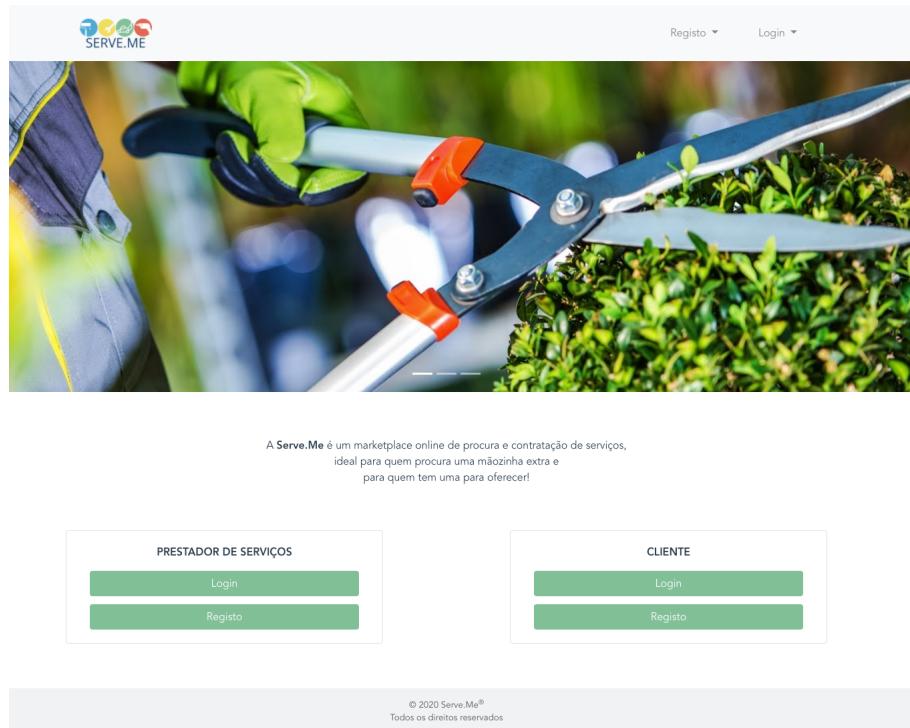


Figura 44: Página Inicial.

Nesta página, para efetuar login o utilizador pode optar por passar o rato pelo botão de **Login** presente na **Navbar** e escolher a opção (**Cliente/Prestador de Serviços**) pretendida (Figura 45) e para efetuar o registo pode encontrar essas opções passando pelo botão de **Registo** (Figura 46). O utilizador pode também utilizar algum dos botões presentes no final da página.

6.2 Página Inicial - Login

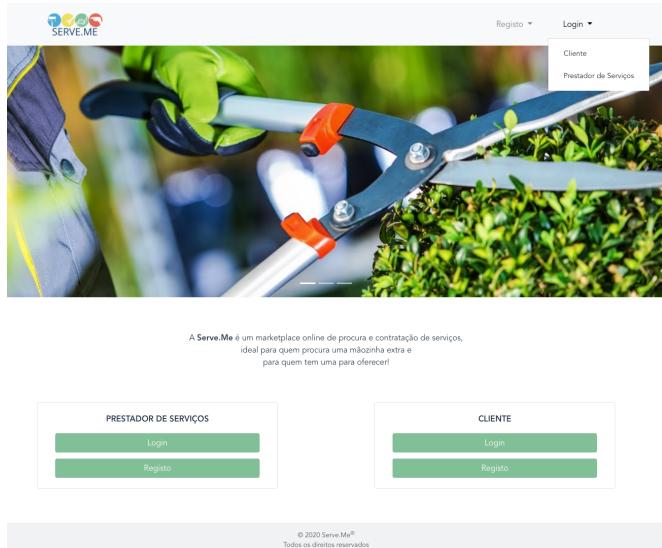


Figura 45: Página Inicial - Login.

6.3 Página Inicial - Registo

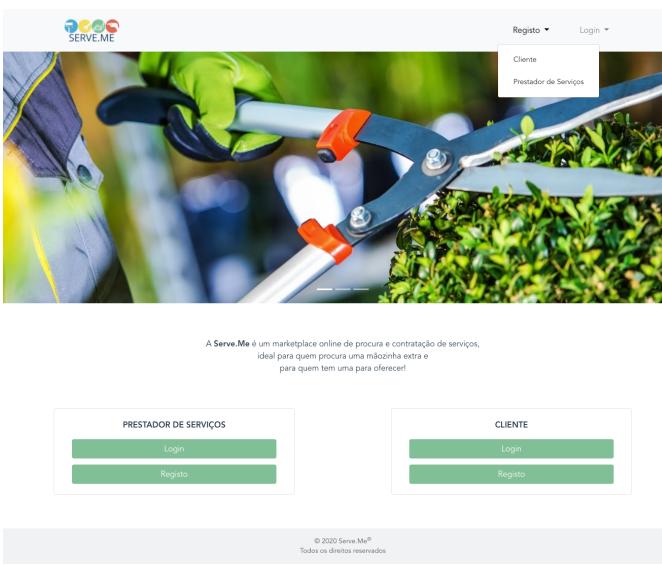


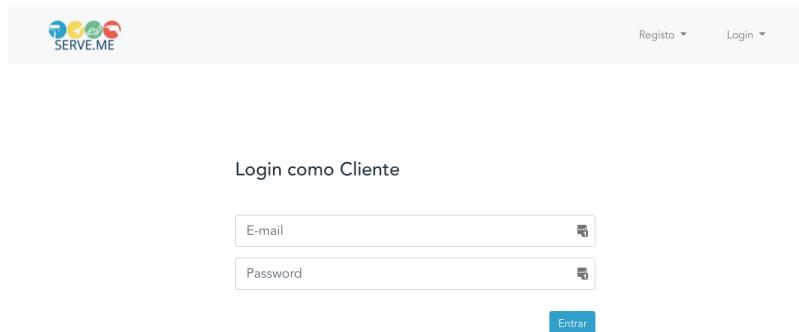
Figura 46: Página Inicial - Registo.

Em seguida, encontram-se respetivamente as páginas de **Login do Cliente** (Figura 47) e de **Login do Prestador** (Figura 48).

Caso o Cliente ou o Prestador de Serviços coloquem o **e-mail e/ou uma password incorreta** é apresentada uma mensagem de aviso, tal como se pode

ver na Figura 49.

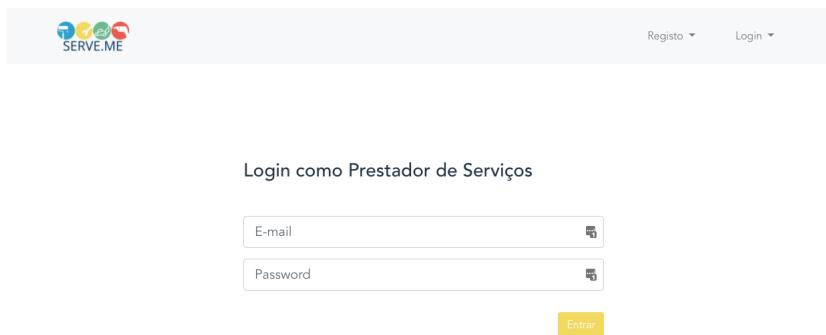
6.4 Login Cliente



The screenshot shows the login interface for clients. At the top, there is a logo consisting of four colored circles (blue, green, yellow, red) followed by the text "SERVE.ME". To the right of the logo are two buttons: "Registro" and "Login", both with dropdown arrows. Below the header, the text "Login como Cliente" is centered. There are two input fields: "E-mail" and "Password", each with a small icon of a person and a lock. To the right of these fields is a blue "Entrar" button.

Figura 47: Login Cliente.

6.5 Login Prestador



The screenshot shows the login interface for providers. At the top, there is a logo consisting of four colored circles (blue, green, yellow, red) followed by the text "SERVE.ME". To the right of the logo are two buttons: "Registro" and "Login", both with dropdown arrows. Below the header, the text "Login como Prestador de Serviços" is centered. There are two input fields: "E-mail" and "Password", each with a small icon of a person and a lock. To the right of these fields is a yellow "Entrar" button.

Figura 48: Login Prestador.

6.6 Login Cliente - Mensagem de Erro

The screenshot shows the SERVE.ME login interface. At the top, there is a logo with three colored circles (blue, green, red) followed by the text 'SERVE.ME'. To the right of the logo are two buttons: 'Registo' and 'Login'. Below the header, the main title 'Login como Cliente' is centered. Underneath it, a red error message 'Credenciais de login inválidas!' is displayed. Below the message are two input fields: one for 'e-mail' containing 'mariasa@serveme.pt' and another for 'password' containing '*****'. To the right of each input field is a small icon of a clipboard with a pencil. At the bottom right of the form is a blue 'Entrar' button.

Figura 49: Login Cliente - Mensagem de Erro.

No caso do **Registro**, os campos a preencher por parte do **Cliente** (Figura 50) e por parte do **Prestador de Serviços** (Figura 51) são os mesmos.

No que diz respeito à validação dos dados inseridos pelos utilizadores, todos os campos são validados antes de ser efetuado o pedido à API. Assim, caso o utilizador insira um **e-mail** inválido é apresentada a mensagem de aviso presente na Figura 52. No caso do **número de contribuinte**, por se tratar de um campo do tipo “number”, não é permitido a inserção de outro tipo de caracteres, sendo apenas apresentada uma mensagem de erro caso o número de contribuinte não contenha exatamente 9 dígitos (Figura 53). O mesmo acontece com o número de telemóvel.

Quando são inseridas informações que já estão presentes na base de dados, nomeadamente o número de contribuinte, o número de telemóvel e/ou o e-mail, é apresentada uma mensagem de erro avisando quais são os **campos que estão já em uso** (Figura 54).

Para facilitar diversos aspectos como a facilidade de utilização por parte dos utilizadores e uma maior facilidade de filtração e ordenação por parte da plataforma, os campos correspondentes ao **Distrito** e ao **Concelho** são *dropdown items*. Assim, caso o utilizador selecione, por exemplo, o Distrito Braga (Figura 55), apenas lhe aparecerão os Concelhos de Braga na caixa seguinte (Figura 56).

Todos os campos são de preenchimento obrigatório, pelo que irá aparecer uma mensagem de erro caso tal requisito não seja cumprido, tal como se pode ver na Figura 57.

É ainda de salientar que quando um utilizador efetua registo na plataforma, depois de clicar no botão de **Registrar**, é reencaminhado para a mesma página de quando efetua Login.

6.7 Registo Cliente

The screenshot shows a registration form titled "Registo como Cliente". At the top right are links for "Registo" and "Login". The form consists of nine input fields: "Primeiro e Último Nome", "E-mail", "Password", "N.º de Contribuinte", "N.º de Telemóvel", "Morada", "Distrito", "Concelho", and "Freguesia". A blue "Registrar" button is located at the bottom right of the form area.

Registo como Cliente

Primeiro e Último Nome

E-mail

Password

N.º de Contribuinte

N.º de Telemóvel

Morada

Distrito ▼

Concelho ▼

Freguesia

Registrar

Figura 50: Registo Cliente.

6.8 Registo Prestador

The screenshot shows the registration form for service providers on the SERVE.ME platform. At the top left is the SERVE.ME logo, which consists of four colored circles (blue, yellow, green, red) followed by the text 'SERVE.ME'. At the top right are two buttons: 'Registo ▾' and 'Login ▾'. The main title 'Registo como Prestador de Serviços' is centered above the form fields. There are nine input fields arranged vertically: 'Primeiro e Último Nome', 'E-mail', 'Password', 'N.º de Contribuinte', 'N.º de Telemóvel', 'Morada', 'Distrito ▾', 'Concelho ▾', and 'Freguesia'. A yellow 'Registrar' button is located at the bottom right of the form area.

Registo como Prestador de Serviços

Primeiro e Último Nome

E-mail

Password

N.º de Contribuinte

N.º de Telemóvel

Morada

Distrito ▾

Concelho ▾

Freguesia

Registrar

Figura 51: Registo Prestador.

6.9 Registo - Erros

The screenshot shows the 'Registo como Cliente' (Registration as Client) form. The first input field contains 'Mário Gomes'. The second input field contains 'mariogomes@', which is highlighted with a blue border and has an orange error message box above it stating 'Please enter a part following '@'. 'mariogomes@' is incomplete.' Below these fields are several empty input fields for phone numbers (210000000, 902000000), address (Rua 7 de maio), and city/town (Braga, Terras de Bouro). A 'Registrar' (Register) button is at the bottom.

Figura 52: Registo - Erro e-mail.

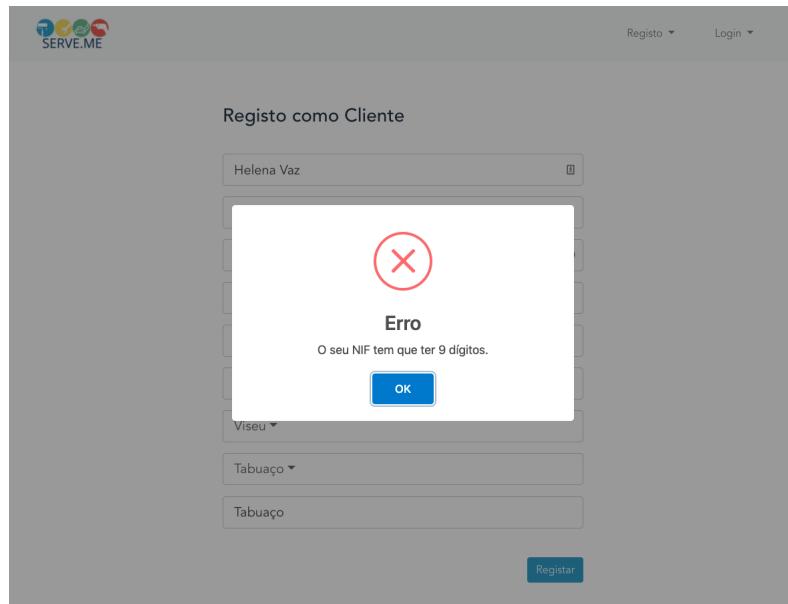


Figura 53: Registo - Erro número de contribuinte.

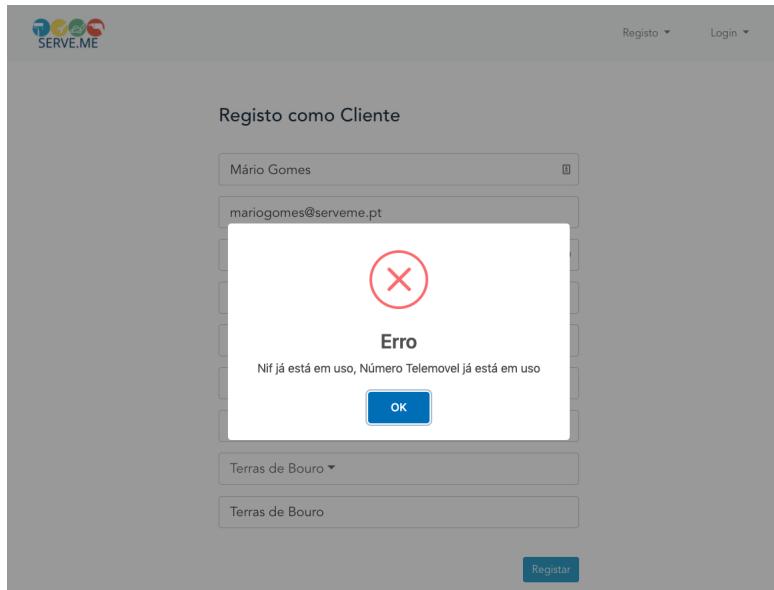


Figura 54: Registo - Erro: informações já em uso.

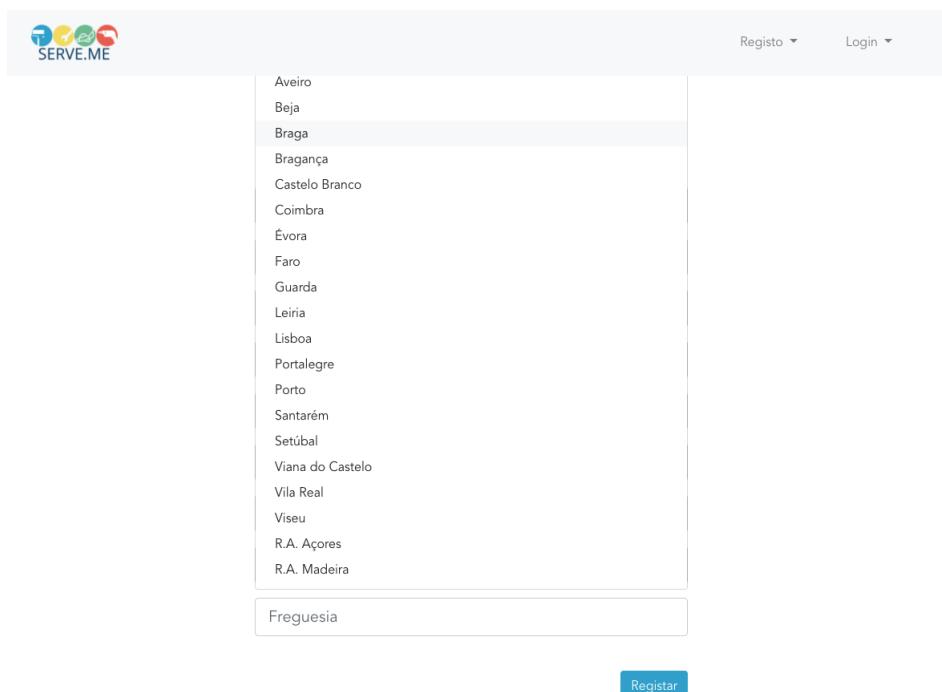
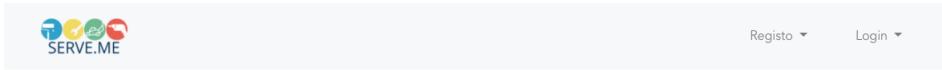


Figura 55: Registo - Distritos.



Registo como Cliente

Amares	<input type="text"/>
Barcelos	<input type="text"/>
Braga	<input type="text"/>
Cabeceiras de Basto	<input type="text"/>
Celorico de Basto	<input type="text"/>
Esposende	<input type="text"/>
Fafe	<input type="text"/>
Guimarães	<input type="text"/>
Póvoa de Lanhoso	<input type="text"/>
Terras de Bouro	<input type="text"/>
Vieira do Minho	<input type="text"/>
Vila Nova de Famalicão	<input type="text"/>
Vila Verde	<input type="text"/>
Vizela	<input type="text"/>
Concelho ▾	<input type="text"/>
Freguesia	<input type="text"/>

Registrar

Figura 56: Registo - Concelhos Braga.

The screenshot shows a registration form for a client. The fields filled are Name (Maria Silva), Email (mariasilva@serveme.pt), and Password (*****). The field 'N.º de Contribuinte' (Tax ID) is highlighted with a blue border and contains the value 91000123, which is marked as required with an orange exclamation icon and the message 'Please fill in this field.' Below the form is a 'Registrar' (Register) button.

Maria Silva

mariasilva@serveme.pt

N.º de Contribuinte

91000123

Please fill in this field.

Rua da Beira

Viseu

Sátão

Sátão

Registrar

Figura 57: Registo - Campos em falta.

6.10 Páginas web do Cliente

6.10.1 Serviços Agendados

Depois do Cliente efetuar login é reencaminhado para a página dos **Serviços Agendados**, tal como se pode ver na figura seguinte (Figura 58).

Nessa página, o utilizador pode consultar todas as informações de cada um dos serviços que tem agendados e tem a possibilidade de efetuar duas ações para cada uma deles (botão a vermelho e botão a azul). A primeira é a opção de **cancelar o agendamento do serviço**. Caso tencione cancelá-lo, é apresentada uma mensagem de verificação, onde o utilizador deve efetivar (ou não) a confirmação do cancelamento do serviço (Figura 59). Em caso de sucesso, é uma apresentada uma mensagem de confirmação. A segunda opção é a de **classificar o serviço**, dando-o assim como finalizado. Para classificá-lo, o utilizador deverá preencher dois campos num *modal* que contém já todas as informações do serviço, faltando somente preencher o número de estrelas e um comentário (opcional) relativo à sua experiência com o determinado prestador (Figura 60).

Figura 58: Cliente - Serviços Agendados.

Figura 59: Cliente - Cancelar Agendamento.

The screenshot shows a service classification form. At the top left is the SERVE.ME logo. The main form has several input fields: 'Jardinagem e bricolage' (Category), 'Corte de Árvores' (Description), '3 palmeiras de 1m' (Quantity), 'Data' (20/07/2020), 'Hora Início' (15h00), 'Duração' (3h), 'Preço/hora' (11€), 'Prestador' (Manuel Oliveira), 'Classificação' (5 stars), and 'Comentários' (empty). At the bottom right are 'Cancel' and 'OK' buttons. On the right side, there's a sidebar titled 'Filtro' showing a list of services with filters for 'Duração' (3h) and 'Preço/hora' (11€).

Figura 60: Cliente - Classificar Serviço.

6.10.2 Pedidos Publicados

Passando o rato na **Navbar**, nomeadamente no item “Os meus serviços” (Figura 61), a segunda opção possível é a de **consultar os pedidos publicados** pelo Cliente.

Clicando nessa opção, é apresentada ao utilizador a tabela que se pode ver através da Figura 62. Nesta tabela, podem estar presentes pedidos com cinco possíveis estados:

- Cancelado;
- Expirado;
- Pendente;
- Agendado;
- Realizado.

Depois de publicar um pedido, o primeiro estado é **Pendente**, tendo o Cliente neste momento a possibilidade de **cancelar** ou **editar** o pedido. No caso de **cancelar**, é apresentada uma caixa para que a ação seja confirmada (Figura 63) e, em caso de sucesso, é apresentada uma mensagem de confirmação (Figura 64). No caso de **editar**, os campos permitidos passam a ser passíveis de serem alterados (Figura 65), tendo o Cliente no final de voltar a clicar no botão do lápis para salvar as alterações. Se houver sucesso na alteração é apresentada uma mensagem de confirmação e, se não houver, é apresentada uma mensagem de insucesso.

Serviços

Os meus serviços

Inbox

Publicar pedido

Francisco Sousa

Agendados

Publicados

Histórico

endedos

Linhas por página

Pesquisa

Filtro

Limpar

Classe	Categoria	Descrição	Prestador	Data	Hora Início	Hora Fim	Duração	Preço/hora	Estado
Jardinagem e Bricolage	Preparação do Solo para Jardinagem	Quintal de 25m2	Manual Oliveira	23/07/2020	14h00		3h	11€	

© 2020 Serve.Me®
Todos os direitos reservados.

Figura 61: Cliente - Navbar: Os meus serviços.

Pedidos Publicados

Os meus serviços

Inbox

Publicar pedido

Francisco Sousa

Linhas por página

Pesquisa

Filtro

Limpar

Classe	Categoria	Descrição	Data	Hora Início	Hora Fim	Duração	Preço/hora	Estado
Jardinagem e Bricolage	Limpeza de Jardim	Jardim com muitas folhas	24/07/2020	9h00	14h00	2h	10€	Pendente
Jardinagem e Bricolage	Limpeza de Jardim	Jardim 20m2	23/07/2020	9h00	12h00	2h	9€	Agendado
Jardinagem e Bricolage	Preparação do Solo para Jardinagem	Quintal de 25m2	23/07/2020	14h00	18h00	3h	10€	Agendado
Jardinagem e Bricolage	Outro	Cortar a relva	22/07/2020	9h00	18h00	1h	9€	Agendado
Jardinagem e Bricolage	Preparação do Solo para Jardinagem	Quintal de 25m2	21/07/2020	9h30	12h30	3h	10€	Cancelado
Jardinagem e Bricolage	Remoção de Ervas Daninhas	Quintal com cerca de 20m2	20/07/2020	9h00	18h00	2h	8€	Cancelado
Jardinagem e Bricolage	Plantação de Árvores	Árvores de fruto num quintal 20m2	20/07/2020	9h00	18h00	3h	9€	Cancelado
Jardinagem e Bricolage	Corte de Árvores	3 palmeiras de 1m	20/07/2020	10h00	18h00	3h	10€	Realizado

Figura 62: Cliente - Pedidos publicados.

Cancelar pedido

Deseja eliminar a publicação deste serviço?

Cancel OK

Linhas por página

Pesquisa

Filtro

Limpar

Classe	Categoria	Descrição	Data	Hora Início	Hora Fim	Duração	Preço/hora	Estado
Jardinagem e Bricolage	Remoção de Ervas Daninhas	Quintal com cerca de 20m2	20/07/2020	9h00	18h00	2h	8€	Pendente

Figura 63: Cliente - Cancelar pedido.

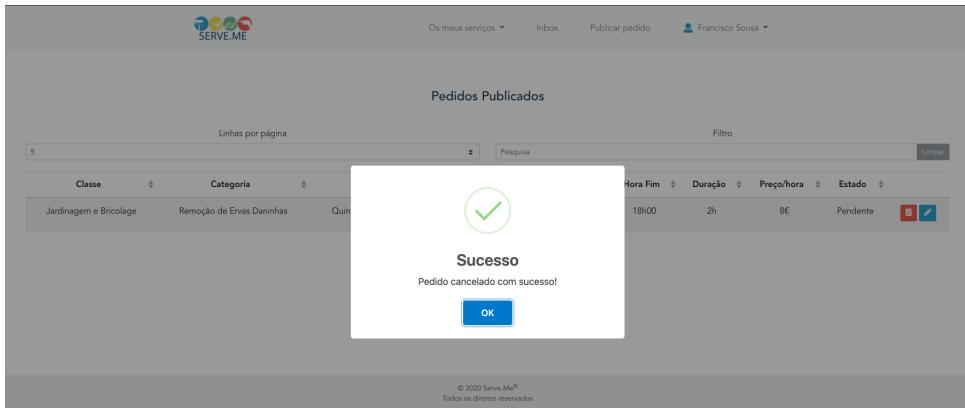


Figura 64: Cliente - Sucesso ao cancelar pedido.



Figura 65: Cliente - Editar pedido.

6.10.3 Histórico

Por fim, na secção de “Os meus serviços” temos ainda a opção do Histórico (Figura 66). Nesta, são passíveis de ser observados serviços com um dos três seguintes tipos:

- Finalizado;
- Realizado com botão para classificar;
- Cancelado.

Caso pretenda classificar o serviço, é apresentado ao Cliente o mesmo *modal* da Figura 60, desaparecendo o botão com a estrelinha apόs o final da conclusão da classificação.

Classe	Categoria	Descrição	Prestador	Data	Hora Início	Duração	Preço/hora	Estado
Jardinagem e Bricolage	Corte de Árvores	3 palmeiras de 1m	Manuel Oliveira	20/07/2020	10h00	3h	11€	Finalizado
Jardinagem e Bricolage	Outro	Cortar a relva	Joel Almeida	22/07/2020	9h00	1h	12€	Cancelado
Jardinagem e Bricolage	Limpeza de Jardim	Jardim 20m ²	Manuel Oliveira	23/07/2020	9h00	2h	11€	Cancelado
Jardinagem e Bricolage	Preparação do Solo para Jardinagem	Quintal de 25m ²	Manuel Oliveira	23/07/2020	14h00	3h	11€	Cancelado
Jardinagem e Bricolage	Limpeza de Jardim	Jardim com muitas folhas	Manuel Oliveira	24/07/2020	9h00	2h	11€	Realizado

Figura 66: Cliente - Histórico.

6.10.4 Inbox

O segundo item da Navbar diz respeito à **Inbox**. Nesta, o Cliente pode **consultar as propostas de agendamento de serviço** que recebe por parte do Prestador de Serviços (Figura 67), podendo **aceitá-las** ou **recusá-las**. Tanto numa, como noutra, é apresentada uma dupla verificação, Figura 68 e Figura 69, respetivamente. É apresentada uma mensagem de sucesso em caso de confirmação (Figura 70).

Nesta tabela, também são apresentados os serviços que o Cliente tiver por **classificar** (Figura 71). Clicando na estrelinha, é apresentado o *modal* que se pode ver na Figura 60.

Por fim, é aqui que pode consultar também os avisos de cancelamento de serviços (Figura 71).

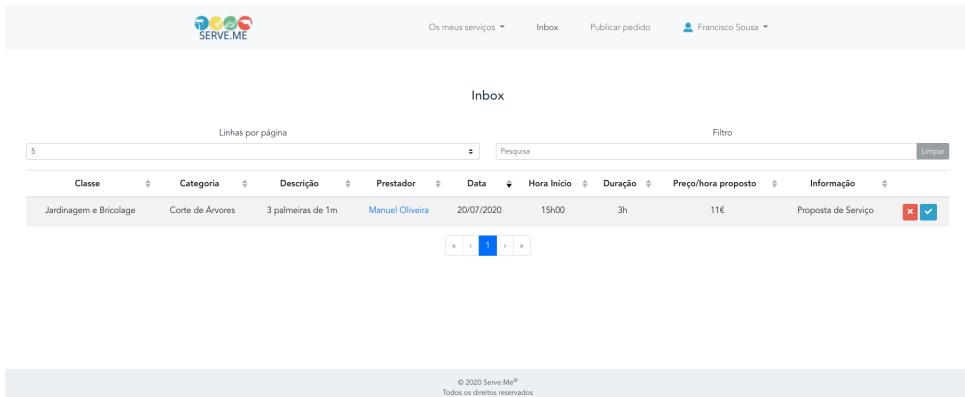


Figura 67: Cliente - Inbox (com propostas).

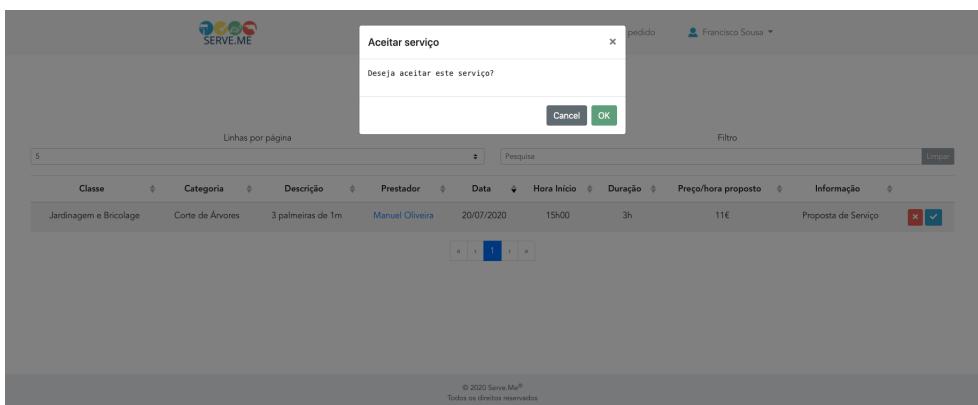


Figura 68: Cliente - Aceitar pedido agendamento.

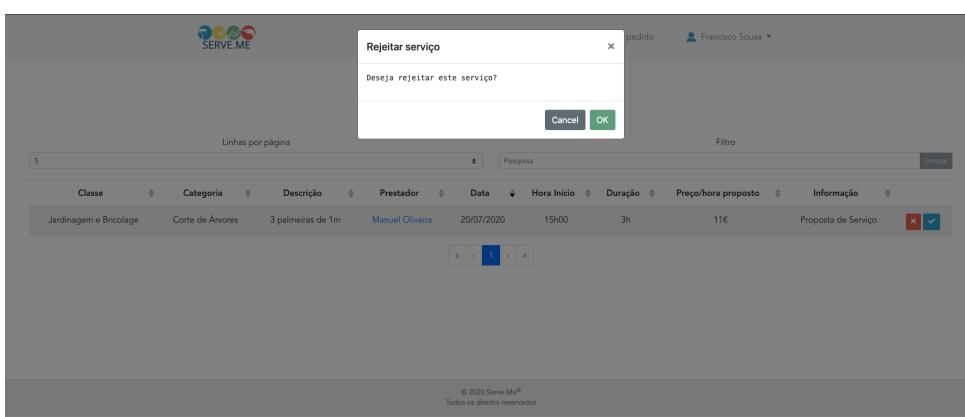


Figura 69: Cliente - Recusar pedido agendamento.

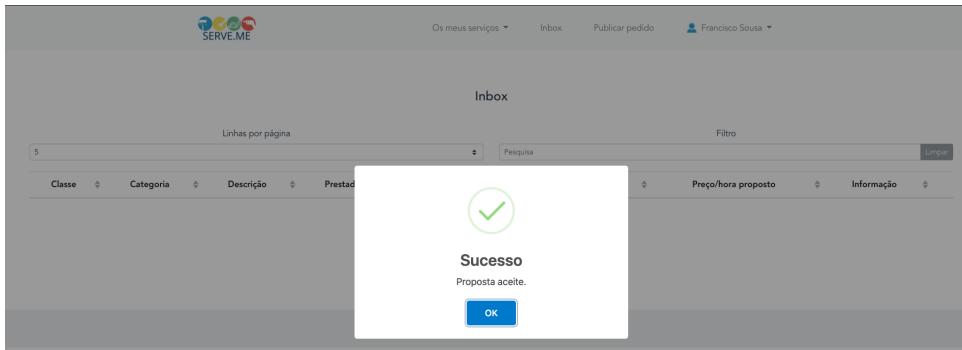


Figura 70: Cliente - Sucesso ao aceitar pedido agendamento.

This screenshot shows the same client inbox interface as Figure 70. The table displays two service entries:

Classe	Categoria	Descrição	Prestador	Data	Hora Início	Duração	Preço/hora proposto	Informação
Jardinagem e Bricolage	Outro	Cortar a relva	Joel Almeida	29/07/2020	14h30	2h	15€	Aviso de cancelamento OK
Jardinagem e Bricolage	Limpeza de Jardim	Jardim com muitas folhas	Manuel Oliveira	24/07/2020	9h30	2h	11€	Por classificar ★

Below the table are standard navigation icons for back, forward, and search.

Figura 71: Cliente - Inbox (com serviços por classificar e avisos de cancelamento).

6.10.5 Publicar pedido

O terceiro item da **Navbar** diz respeito à opção de **Publicar um pedido de serviço**. Tal como se pode ver na figura seguinte (Figura 72), o Cliente tem que fornecer algumas informações. A **Classe** e a **Categoria** são escolhidas através de uma *dropdown box*, não havendo margem para erros.

No caso do campo da **Data, Hora inicial, Hora fim e Duração**, como são dos tipos “date” (Figura 73), “time” (Figura 74), “time” e “time”, respetivamente, o utilizador não tem também sequer a oportunidade de inserir dados inválidos. No caso do **Preço por hora**, a *input box* é do tipo “number” e tem que ter um valor “min=0”, logo, o utilizador também não consegue digitar caracteres não desejados.

Para além disso, todos os campos são de preenchimento obrigatório (excepto a descrição), aparecendo uma mensagem de erro caso algum deles esteja por preencher (o aviso é semelhante ao da Figura 57).

Em caso de sucesso, é apresentada uma mensagem de confirmação, tal como se pode ver na Figura 75.

The screenshot shows the 'Publicar pedido' (Post Request) form. At the top, there's a header with the SERVE.ME logo, navigation links like 'Os meus serviços', 'Inbox', 'Publicar pedido', and a user profile for 'Francisco Sousa'. The main form area has the title 'Publicar pedido'. It contains several input fields: 'Classe' (Class) with a dropdown menu; 'Categoria' (Category) with a dropdown menu; 'Descrição' (Description) with a text input field; 'Data' (Date) with a date input field; 'Hora inicial de disponibilidade' (Initial Availability Time) with a time input field; 'Hora fim de disponibilidade' (Final Availability Time) with a time input field; 'Duração' (Duration) with a time input field; and 'Preço por hora (€)' (Hourly Price) with a text input field. A blue 'Publicar' (Post) button is at the bottom right.

Figura 72: Cliente - Publicar pedido.

Publicar pedido

Classe

Categoria

Descrição

Data dd/mm/yyyy

July 2020 ▾						
S	M	T	W	T	F	S
28	29	30	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1
2	3	4	5	6	7	8

Hora

Hora

Duração

Preço

Figura 73: Cliente - Inserir data.

Publicar pedido

Classe

Categoria

Descrição

Data dd/mm/yyyy

Hora inicial de disponibilidade --:--

Hora final de disponibilidade --:--

Duração --:--

Preço por hora (€)

A dropdown menu for selecting the start time shows the following options:

17	05
18	06
19	07
20	08
21	09
22	10
23	11

Figura 74: Cliente - Inserir hora de início.

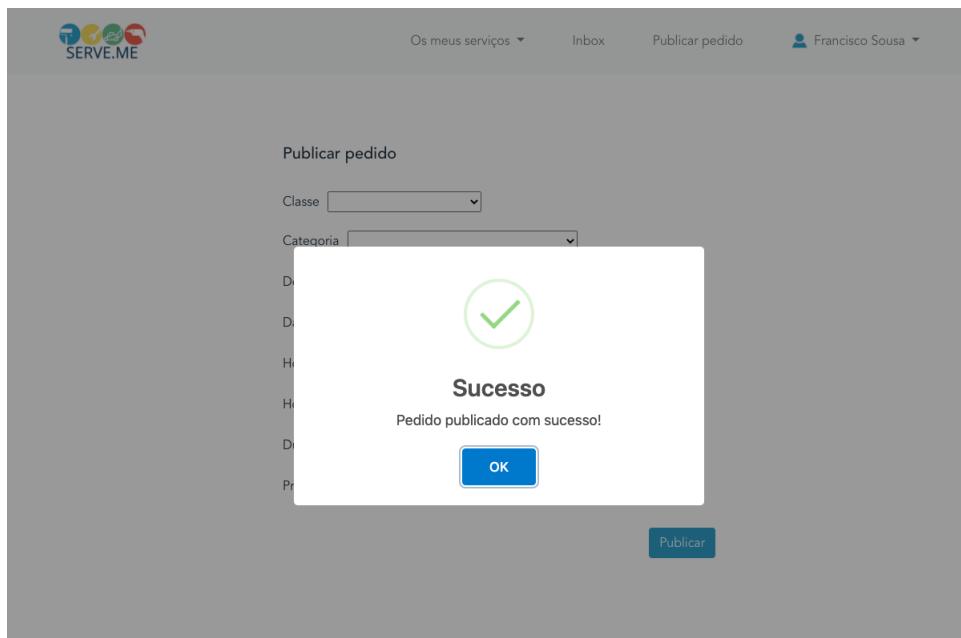


Figura 75: Cliente - Sucesso ao publicar pedido.

6.10.6 Ver/editar perfil

No quarto e último item da navbar é possível visualizar o nome do Cliente. Passando o rato por cima do nome, são apresentadas três diferentes opções: **ver perfil**, **alterar password** e **terminar sessão** (Figura 76).

Tal como se pode ver na figura seguinte, a primeira opção diz respeito a ver e editar o perfil. Aqui, assim como no caso do Registo, são efetuadas validações ao nível do número de telemóvel e todos os campos têm que estar preenchidos. O e-mail e o número de contribuinte não poderão ser alterados.

The screenshot shows the SERVE.ME client profile editing interface. At the top, there's a navigation bar with the SERVE.ME logo, 'Os meus serviços' dropdown, 'Inbox', 'Publicar pedido', and a user profile for 'Francisco Sousa'. A dropdown menu from the profile icon shows options: 'Perfil' (selected), 'Alterar password', and 'Terminar sessão'. The main area is titled 'Perfil' and contains fields for 'Nome' (Francisco Sousa), 'E-mail' (franciscosousa@serveme.pt), 'N.º de Contribuinte' (259000000), 'N.º de Telemóvel' (930000000), 'Morada' (Rua 10 de maio), and dropdowns for 'Distrito' (Bragança), 'Concelho' (Vila Flor), and 'Freguesia' (Vila Flor). A blue 'Guardar' button is at the bottom right.

Figura 76: Cliente - Ver perfil.

6.10.7 Alterar password

Para **alterar a sua password**, o utilizador terá que inserir a **password atual**, **a nova password** e **confirmar a nova password** (Figura 77).

Neste processo, existem dois diferentes erros que poderão surgir. O primeiro diz respeito à inserção de **novas passwords que não coincidem** (Figura 78) e o segundo refere-se a uma **incorrecta inserção da password atual** (Figura 79).



Figura 77: Cliente - Alterar password.

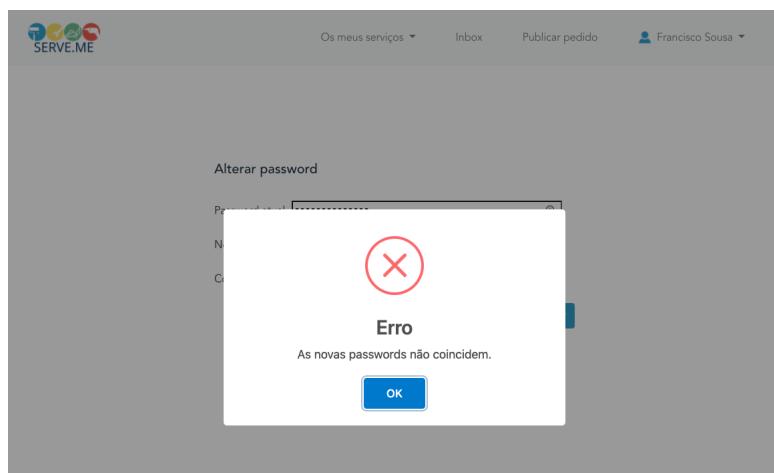


Figura 78: Cliente - Erro passwords não coincidem.

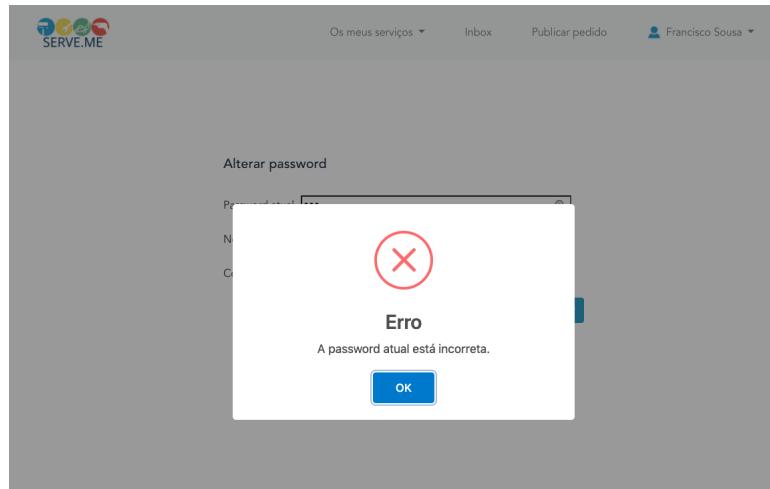


Figura 79: Cliente - Erro password atual incorreta.

6.11 Páginas web do Prestador

6.11.1 Serviços Agendados

Assim que inicia sessão, o Prestador é reencaminhado para a página dos **Serviços Agendados** (Figura 80). Nesta página, o Prestador consegue visualizar um **calendário** onde tem acesso à vista semanal dos seus serviços agendados. Pode também optar por uma vista diária, mensal ou anual.

Assim como acontece com o Cliente, como apresentado na Secção 6.10.1, o Prestador pode também **cancelar** um serviço agendado ou **classificar** um serviço.

The screenshot shows the 'Serviços Agendados' (Scheduled Services) page. At the top, there's a navigation bar with icons for Dashboard, Inbox, Propostas (Proposals), Consultar pedidos (Check Requests), and a user dropdown for 'Manual Oliveira'. Below the navigation is a title 'Serviços Agendados' and a subtitle 'Semana 30 (Julho 2020)'. The main area is a grid calendar for July 30, 2020, from 08:00 to 18:00. A green box highlights an appointment at 15:00 for 'Corte de Árvores' (Tree Cutting). Below the calendar is a table with service details:

Classe	Categoria	Descrição	Cliente	Data	Hora Início	Duração	Preço/hora
Jardinagem e Bricolage	Corte de Árvores	3 palmeiras de 1m	Francisco Sousa	20/07/2020	15h00	3h	11€

At the bottom right of the table are 'X' and '★' buttons. The footer contains the text '© 2020 Serve Me®' and 'Todos os direitos reservados'.

Figura 80: Prestador - Serviços Agendados.

6.11.2 Histórico

Passando o rato pelo item “Dashboard”, o Prestador tem acesso a duas outras opções (Figura 81). A segunda diz respeito ao **Histórico**.

Tal como acontece com o Cliente (Secção 6.10.3), o Prestador aqui pode visualizar os serviços **finalizados**, **realizados** por **classificar** e **cancelados**.

Figura 81: Prestador - Navbar: Dashboard.

6.11.3 Estatísticas

O Prestador pode ainda consultar as suas **Estatísticas** anuais, tal como se pode ver na figura seguinte, conseguindo ter acesso ao **valor total de ganhos** e ao **número total de serviços realizados** no ano corrente.

Para além disso, tem ainda acesso a um gráfico barras com o **total de ganhos por mês**, a um gráfico de linhas com o **número total de serviços por mês** e também a um gráfico circular que apresenta o **número de serviços por categoria**.

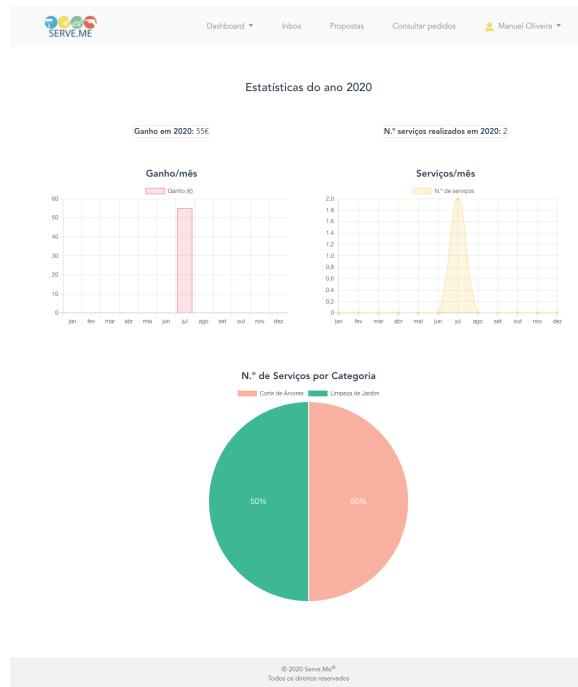


Figura 82: Prestador - Estatísticas.

6.11.4 Inbox

Na sua **Inbox**, o Prestador recebe notificações como, por exemplo, quando uma **proposta é aceite** ou **rejeitada** podendo dar “**vista**” nessa mesma notificação. O utilizar consegue também consultar os **avisos de cancelamento** e também quando tem algum **serviço por classificar**.

Classe	Categoria	Descrição	Cliente	Data	Hora Início	Duração	Preço/hora proposto	Informação
Jardinagem e Bricolage	Preparação do Solo para Jardinagem	Quintal de 25m2	Francisco Sousa	23/07/2020	14h00	3h	11€	Aviso de cancelamento OK
Jardinagem e Bricolage	Manutenção de Canteiros	10 canteiros de 5m2	Jose Fonseca	24/07/2020	14h00	1h	10€	Por classificar ★
Jardinagem e Bricolage	Limpeza de Jardim	Jardim com muitas folhas	Francisco Sousa	24/07/2020	9h30	2h	11€	Proposta aceite OK
Jardinagem e Bricolage	Vedação para Jardim	Vedação de madeira 10m2	Carolina Loureiro	23/07/2020	11h00	2h	20€	Proposta rejeitada OK

Figura 83: Prestador - Inbox.

6.11.5 Propostas

Nesta página o Prestador pode consultar o estado de todas as suas propostas enviadas, podem estas estarem com um estado igual a uma das três seguintes opções:

- Pendente;
- Aceite;
- Rejeitada.

Classe	Subcategoria	Descrição	Cliente	Data	Hora Início	Duração	Preço/hora	Informação
Jardinagem e Bricolage	Vedação para Jardim	Vedação de madeira 10m2	Carolina Loureiro	23/07/2020	11h00	2h	20€	Rejeitada
Jardinagem e Bricolage	Plantação de Árvores	Muitas árvores	Patrick Maia	13/08/2020	10h00	1h30	10€	Pendente
Jardinagem e Bricolage	Corte de Árvores	3 palmeiras de 1m	Francisco Sousa	20/07/2020	15h00	3h	11€	Aceite
Jardinagem e Bricolage	Limpeza de Jardim	Jardim 20m2	Francisco Sousa	23/07/2020	9h00	2h	11€	Aceite
Jardinagem e Bricolage	Preparação do Solo para Jardinagem	Quintal de 25m2	Francisco Sousa	23/07/2020	14h00	3h	11€	Aceite
Jardinagem e Bricolage	Limpeza de Jardim	Jardim com muitas folhas	Francisco Sousa	24/07/2020	9h30	2h	11€	Aceite
Jardinagem e Bricolage	Manutenção de Canteiros	10 canteiros de 5m2	Jose Fonseca	24/07/2020	14h00	1h	10€	Aceite

Figura 84: Prestador - Propostas.

6.11.6 Consultar pedidos

O Prestador pode **consultar todos os pedidos** efetuados pelos Clientes, tal como se pode ver na Figura 85, tendo a opção de **ordená-los** por Classe, Categoria, Concelho, Data e por Preço/hora, como se pode ver na Figura 86.

O Prestador pode **consultar o perfil do Cliente**, conseguindo assim saber mais algumas informações acerca deste, tais como a morada, a classificação média, número de serviços realizados e cancelados, os comentários, entre outros (Figura 87).

Caso se interesse por algum pedido, pode efetuar uma proposta clicando para isso no botão destinado para esse efeito. Consequentemente, é apresentado um *modal* com as informações do pedido e é necessário que o Prestador preencha o campo correspondente à **hora que pretenderá iniciar o serviço** e também o **preço/hora** que pretende receber (Figura 88). Preenchendo esses campos, ocorre a verificação de que a hora de início se encontra no intervalo de disponibilidade apresentado previamente pelo Cliente. Em caso contrário, é apresentada a mensagem de erro que se pode ver na Figura 89.

Em caso de sucesso, é apresentada uma mensagem de confirmação (Figura 90).

SERVE.ME

Dashboard ▾

Inbox

Propostas

Consultar pedidos

Manuel Oliveira ▾

Pedidos

Ordenar por: Classe ▾

Filtrar por: [campo de texto]

	Jardinagem e Bricolage Corte de Árvores	Jardinagem e Bricolage Vedação para Jardim	Jardinagem e Bricolage Manutenção de Canteiros
Descrição: 3 palmeiras de 1m	Descrição: Vedação de madeira 10m2	Descrição: 10 canteiros de 5m2	
Concelho: Vila Flor	Concelho: Coimbra	Concelho: Guarda	
Data: 20/07/2020	Data: 23/07/2020	Data: 24/07/2020	
Hora inicio: 10h00	Hora inicio: 11h00	Hora inicio: 13h00	
Hora fim: 18h00	Hora fim: 16h00	Hora fim: 19h00	
Duração: 3h	Duração: 2h	Duração: 1h	
Preço/hora: 10€	Preço/hora: 10€	Preço/hora: 7€	
Cliente: Francisco Sousa	Cliente: Carolina Loureiro	Cliente: Jose Fonseca	
Efetuar proposta	Efetuar proposta	Efetuar proposta	

	Jardinagem e Bricolage Plantação de Árvores
Descrição: Muitas árvores	
Concelho: Leiria	
Data: 13/08/2020	
Hora inicio: 9h00	
Hora fim: 15h00	
Duração: 1h30	
Preço/hora: 9€	
Cliente: Patrick Maia	
Efetuar proposta	

Figura 85: Prestador - Consultar pedidos.

SERVE.ME

Dashboard ▾

Inbox

Propostas

Consultar pedidos

Manuel Oliveira ▾

Pedidos

Ordenar por: Classe ▾

Filtrar por: [campo de texto]

	Jardinagem e Bricolage Corte de Árvores	Jardinagem e Bricolage Vedação para Jardim
	Jardinagem e Bricolage Manutenção de Canteiros	
	Jardinagem e Bricolage Plantação de Árvores	
	Descrição: 3 palmeiras de 1m Concelho: Vila Flor Data: 20/07/2020 Hora inicio: 10h00 Hora fim: 18h00 Duração: 3h Preço/hora: 10€ Cliente: Francisco Sousa	Descrição: Vedação de madeira 10m2 Concelho: Coimbra Data: 23/07/2020 Hora inicio: 11h00 Hora fim: 16h00 Duração: 2h Preço/hora: 10€ Cliente: Carolina Loureiro
	Efetuar proposta	Efetuar proposta
	Efetuar proposta	
	Efetuar proposta	
	Efetuar proposta	

© 2020 Serve Me®
Todos os direitos reservados

Figura 86: Prestador - Consultar pedidos (ordenar).

The screenshot shows the SERVE.ME platform interface. At the top, there is a navigation bar with the logo 'SERVE.ME' and links for 'Dashboard', 'Inbox', 'Propostas', 'Consultar pedidos', and a user profile for 'Manuel Oliveira'. The main content area is titled 'Perfil' (Profile) and displays the following information for a client named Carolina Loureiro:

- Name:** Carolina Loureiro
- E-mail:** carolinaloureiro@serveme.pt
- N.º de Telemóvel:** 936674453
- Morada:** Rua 17
- Freguesia:** Coimbra **Concelho:** Coimbra **Distrito:** Coimbra
- Classificação média:** ★ ★ ★ ★ ★ 5.000
- N.º de serviços realizados:** 1
- N.º de serviços cancelados:** 0

Below this, there is a section for 'Comentários:' (Comments) with a box containing a review from 'Nuno Paulo' with a 5-star rating and the text 'Foi muito agradável.' (Was very pleasant).

Figura 87: Prestador - Ver perfil do Cliente.

The screenshot shows a modal dialog box for proposing a service. The form fields include:

- Título:** 3 palmeiras de 1m
- Concelho:** Vila Flor
- Data:** 20/07/2020
- Hora Início Disponibilidade:** 10h00
- Hora Fim Disponibilidade:** 18h00
- Duração:** 3h
- Descrição:** 3 palmeiras de 1m
- Concelho:** Vila Flor
- Data:** 20/07/2020
- Hora início:** Cliente
- Hora fim:** Francisco Sousa
- Duração:** 3h
- Hora início proposto:** --:--
- Preço/hora:** 10€
- Cliente:** Francisco Sousa
- Preço/hora proposto (€):** [Empty input field]

At the bottom right of the dialog are 'Cancel' and 'OK' buttons.

Figura 88: Prestador - Efetuar proposta.

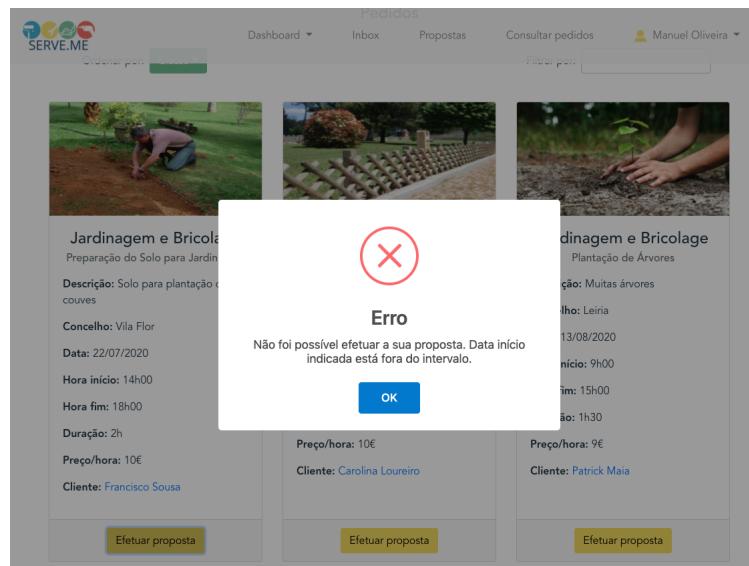


Figura 89: Prestador - Erro ao efetuar proposta.

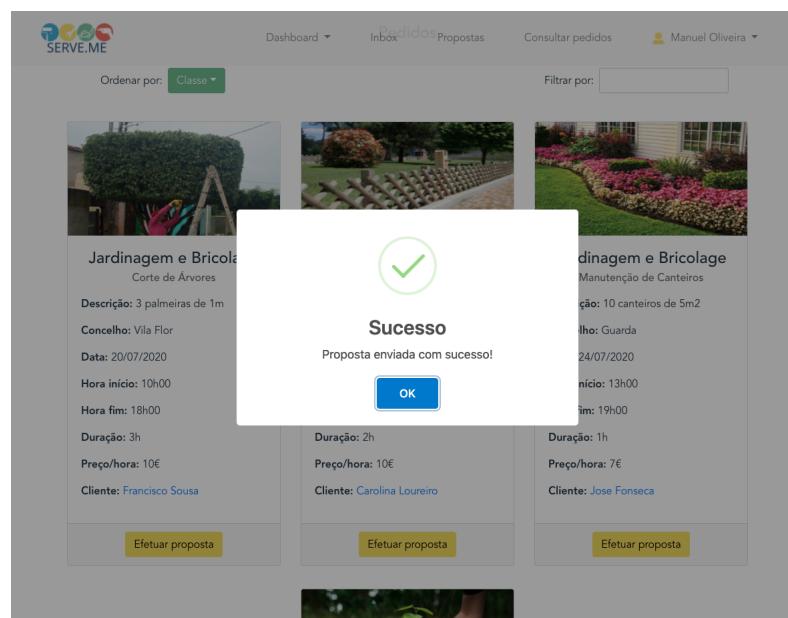


Figura 90: Prestador - Sucesso ao efetuar proposta.

6.11.7 Perfil

No caso das funcionalidades de **ver e editar perfil, alterar password e terminar sessão**, o processo e o layout é análogo ao do Cliente, que foi já explicado nas Secções 6.10.6 e 6.10.7.

6.12 404 - *Page not found*

No caso do URL visitado pelo utilizador não existir na plataforma, o mesmo é reencaminhado para a página apresentada na figura seguinte.

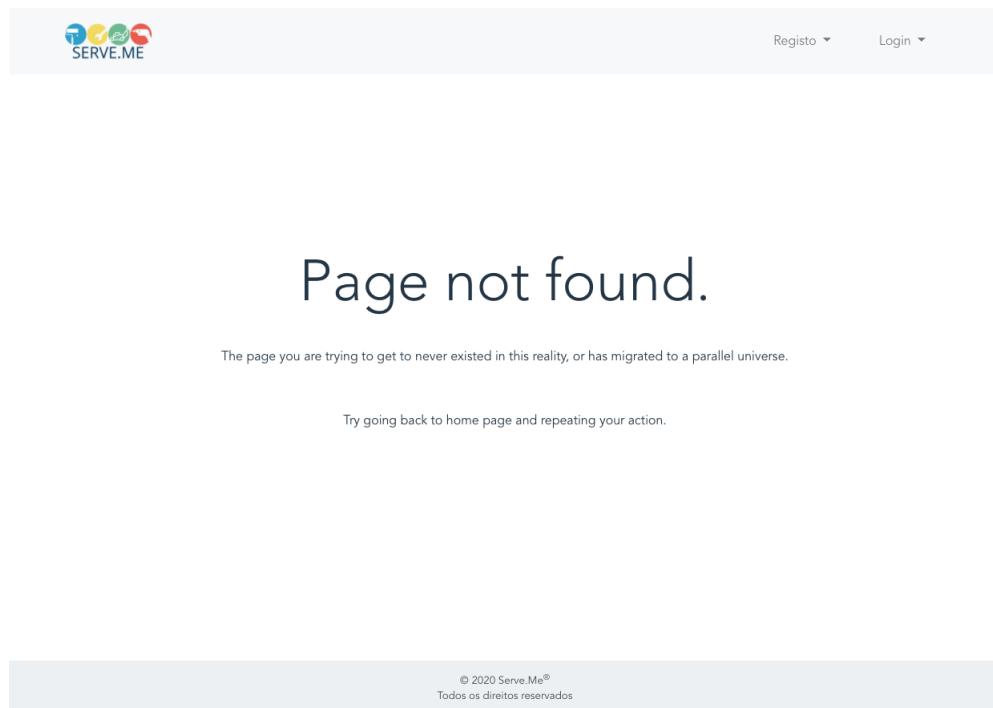


Figura 91: Erro 404: *Page not found*.

7 Análise de Usabilidade

Segundo a norma ISO/IEC 25010:2011, para um produto de software ter **qualidade** tem que cumprir um conjunto de dimensões. Entre elas, está a **Usabilidade**.

7.1 Princípios de Usabilidade

Quando desenhamos uma peça de software temos que **desenhar para a usabilidade**, isto é, desenhá-la com o objetivo que proporcionar o máximo de usabilidade possível aos nossos futuros utilizadores.

Os **Princípios de Usabilidade** devem ser tidos em conta desde o início do design da interface gráfica e devem ser revisitadas vários vezes ao longo do processo de desenvolvimento. Estes Princípios dividem-se em três grandes grupos, que passaremos a analisar superficialmente nas secções seguintes.

7.1.1 *Learnability*

O primeiro princípio diz respeito à **Learnability**, que é a facilidade com que novos utilizadores podem iniciar uma interação eficaz e alcançar máxima performance. Este princípio divide-se em cinco outros princípios, nomeadamente a **Predictability**, a **Synthesizability**, a **Familiarity**, a **Generalizability** e a **Consistency**.

No que diz respeito à **Predictability**, temos como exemplo os formulários de inserção de dados, onde, por exemplo, no caso do número de telemóvel e do número de contribuinte no formulário de Registo (Figura 51) não é sequer permitida a inserção de caracteres não numéricos. No caso da **Familiarity** temos por exemplo o calendário diário/ semanal/ mensal/ anual (Figura 80), onde essa forma de apresentar os serviços é bastante familiar para os utilizadores.

7.1.2 *Flexibility*

A **Flexibility** é um princípio que diz respeito à multiplicidade de formas pelas quais o utilizador e o sistema trocam informação. Este princípio engloba cinco subcategorias, entre as quais o **Dialogue initiative**, **Multithreading**, **Task migrability**, **Substitutivity** e **Customizability**.

Na nossa plataforma, podemos reconhecer este princípio, por exemplo, na flexibilidade em optar por apresentar 5, 10 ou 15 items por página em cada tabela (Figura 62 vs. Figura 66) ou então por exemplo no caso do Prestador poder ver os serviços agendados no formato tabela ou no formato horário (Figura 80).

7.1.3 *Robustness*

Por fim, a **Robustness** refere-se ao nível de suporte fornecido ao utilizador na determinação do êxito e na avaliação do comportamento direcionado a objetivos. Este divide-se em quatro princípios, que são os seguintes: **Observability**, **Recoverability**, **Responsiveness** e **Task conformance**.

Podemos comprovar este princípio, por exemplo a **Observability**, em todas as tabelas da Serve.Me, visto que todas elas têm informação relativa à página da tabela que o utilizador se encontra (Figura 83, por exemplo).

7.2 Heurísticas de Nielsen

De modo a perceber se atingimos o nosso objetivo de construir uma interface gráfica com o máximo grau de usabilidade podemos recorrer a Métodos Analíticos, neste caso de Inspecção por Peritos, mais precisamente às **Heurísticas de Nielsen**. Tratam-se assim de 10 heurísticas ou conselhos para se criar uma boa interface.

É de frisar que estas Heurísticas foram revisitadas várias vezes ao longo do processo de **design** e **construção** da interface, visto se tratar de um processo iterativo (com avaliações formativas). Neste momento, chegamos à **solução** final e, por isso, procederemos em seguida à avaliação sumativa da solução.

Desta forma, vamos visitar cada uma das heurísticas de forma a avaliar se todas elas estão a ser cumpridas.

7.2.1 *Visibility of system status*

A primeira heurística diz respeito à **visibilidade do estado do sistema** e se o mesmo mantém o utilizador informado sobre o que está a acontecer.

No caso da **Serve.Me**, não existe nenhuma tarefa demorada do ponto de vista do sistema que requeira barras de progresso. No que diz respeito a mensagens de confirmação, são várias as mensagens que vão aparecendo ao longo do decurso da utilização da aplicação *web*, tal como se pode ver através da Figura 64, 70 e 75, por exemplo.

7.2.2 *Match between system and the real world*

Na segunda heurística procura-se garantir que existe uma **correspondência entre o sistema e o mundo real**, procurando que a linguagem e a estruturação da interface sejam familiares aos utilizadores-alvo.

No que diz respeito, por exemplo, à inserção de datas e de horários, a Serve.Me cumpre este requisito, tal como se pode ver na Figura 73 e 74, respetivamente. Também no caso da consulta do seu calendário diário/ semanal/ mensal/ anual, a interface gráfica é também bastante intuitiva e vai ao encontro do que o utilizador já está habituado (Figura 80).

7.2.3 *User control and freedom*

Nesta heurística, tenta garantir-se que o **utilizador se sente livre e em controlo** ao utilizar a interface. Desta forma, é importante que se forneça saídas, *undo* e *redo*, e que se guie o utilizador pela navegação da interface sem ele praticamente se aperceber.

Pode-se encontrar exemplos disto nas Figuras 59, 68 e 69, onde o utilizador é questionado se pretende mesmo efetuar a determinada ação, tendo em consideração a importância da mesma. Em casos de agendamento, o utilizador tem ainda a possibilidade de cancelar o mesmo, caso se arrependa ou a sua disponibilidade inesperadamente se altere (Figura 80). Também no caso de publicar pedidos, o Cliente pode sempre fazer *undo* dessa publicação sem problemas (Figura 62).

7.2.4 Consistency and standards

É também importante que ao construir uma aplicação *web* se **sigam os standards e se seja consistente**, uma vez que os utilizadores usam várias aplicações no seu dia a dia e estão já habituados a diversos tipos de abordagens.

Desta forma, procurou-se seguir esta heurística em cada pequeno detalhe, como no facto de se colocar a ação mais comum no lado direito e a menos comum no lado esquerdo, Figura 62 e 80, por exemplo. Para além destes, existem vários outros exemplos que comprovam a utilização desta heurística visto que toda a aplicação *web* foi construída assente neste princípio.

7.2.5 Error prevention

Antes de permitir que o utilizador cometa erros e só se aperceba no final correndo o risco de ter que repetir a sua ação novamente, é importante que se **previna os erros**.

Nesta aspeto, podemos dar como exemplo o preenchimento do formulário de Registo (Figura 50), onde não é sequer possível que o utilizador digite caracteres não numéricos no campos correspondentes ao número de contribuinte e ao número de telemóvel (clicando em outras teclas não aparece nada na caixa de texto).

7.2.6 Recognition rather than recall

Nesta sexta heurística, o objetivo é promover que a pessoa **se lembre das coisas porque as vê e reconhece**, e não porque tem que “puxar” pela memória. Assim, o utilizador não deve ter que se lembrar de informação de uma parte da interface para outra.

Para ilustrar o cumprimento desta heurística, temos o caso de Efetuar Proposta por parte do Prestador de Serviços. Quando o utilizador clica nesse botão (Figura 85) é-lhe apresentado um *modal* que volta a apresentar as informações que estavam presentes na *card* que este clicou, tal como se pode ver na Figura 88. O mesmo acontece com o caso de classificar um serviço, onde o *modal* também repete todas as informações referentes ao serviço realizado (Figura 60).

7.2.7 Flexibility and efficiency of use

Outro princípio a ter em conta é o da **flexibilidade e eficiência**. É importante que os utilizadores experientes consigam acelerar a interação e, para isso, a interface fornece atalhos (*shortcuts*) com acesso direto a itens utilizados mais frequentemente.

Como esta aplicação *web* não é muito complexa, as ações aparecem já de forma bastante visível aos utilizadores. As ações mais frequentes estão já expostas na Navbar, como, por exemplo Publicar um pedido por parte do Cliente (Figura 72) ou o Consultar pedidos por parte do Prestador de Serviços (Figura 85).

7.2.8 Aesthetic and minimalist design

O design da interface gráfica deve ter uma **boa estética e deve ser minimalist**a. Desta forma, os diálogos não devem ter informação irrelevante/desnecessária e a informação deve ser apresentada de forma simples e compreensível.

A Serve.Me foi construída com este mesmo objetivo, visto também ser fundamental para a permanência do utilizador no nosso *website*. Como exemplos, qualquer uma das interfaces construídas servem, na nossa opinião, para este efeito, visto que todas elas são fáceis de perceber o que significam e o que pretendem do utilizador.

7.2.9 Help users recognize and recover from errors

É também importante possibilitar aos utilizadores que **reconhecerem e recuperarem dos erros**. Assim, as mensagens de erro apresentadas devem conter uma linguagem simples e indicar qual o problema ocorrido.

Para demonstrar a aplicação deste princípio temos como exemplo a Figura 54, 57, 78, 79 e 89.

7.2.10 Help and documentation

Por fim, a aplicação *web* deve ainda oferecer **ajuda e documentação** para guiar as pessoas para o sítio certo. Como a Serve.Me é bastante simples e tem um número limitado de funcionalidades, não sentimos a necessidade de fornecer uma página de ajuda para esclarecer os utilizadores visto que existem poucos menus e pouquíssimas *tabs* aninhadas.

8 Conclusões e Trabalho Futuro

O primeiro desafio enfrentado no processo de génesis da **Serve.Me** foi a própria concepção da **ideia**. Queríamos que fosse algo que realmente fosse útil para a sociedade e que colmatasse um problema real. Todo o processo de **modelação, implementação e deploy** foi meticulosamente pensado com o grande objetivo de que esta plataforma *web* pudesse mesmo um dia ser colocada online e que pudesse ser utilizada por centenas ou milhares de pessoas.

Começando pela **fase de Modelação** (Secção 2), recorremos a familiares para percebermos quais os requisitos que a aplicação deveria ter e, partindo desses requisitos, elaboramos o Diagrama de Use Cases, que sintetizou esses requisitos e traduziu-os em *features* (Use Cases) que deveríamos, nas fases seguintes, dar resposta. Com o Diagrama de Classes, começamos a aproximar-nos da fase de implementação, percebendo mais concretamente como é que os dados estariam organizados de forma a solucionar todos os Use Cases de forma simples, prática e inteligente. Por fim, começamos a cogitar acerca da interface gráfica, tentando perceber como poderíamos obter um resultado que fosse apelativo e ao mesmo tempo bastante usável. Tentamos ser bastante pacientes neste processo, por se tratar de um processo iterativo e que requeria bastante reflexão e imparcialidade na hora de avaliar formativamente se os princípios de usabilidade aprendidos em contexto de sala de aula estavam realmente a serem aplicados.

Na **fase da implementação** (Secção 3), começamos a pôr as “mãos na massa” e a pôr em prática as conclusões que tiramos na fase anterior. Fizemos uma análise exaustiva de quais seriam as melhores tecnologias e linguagens a utilizar, tendo por base fatores como a importância no mercado, características diferenciadoras, curva de aprendizagem e a dimensão da comunidade de suporte, levando-nos a tomar a decisão de utilizar *Vue.js* para a camada de apresentação, *Spring Boot* para a lógica de negócio e *Hibernate* para a camada de dados. O frontend e o backend são módulos totalmente independentes, onde o frontend recorre a uma API para obter os dados e fazer as alterações que precisa.

A implementação do frontend correu sem sobressaltos com a utilização de *Vue* e *BootstrapVue* essencialmente, tirando partido das mais-valias desta *framework*, nomeadamente do *routing*, da utilização de componentes pré-criados e da facilidade de importação de outras bibliotecas. A curva de aprendizagem de *Vue.js* é bastante amigável, pelo que o facto de nunca termos utilizado esta *framework* não atrapalhou o processo, apenas exigiu um pouco mais de esforço e tempo da nossa parte. No final, percebemos o motivo de esta estar entre as *frameworks* mais utilizadas atualmente, concordando desta forma com os comentários da comunidade.

No que toca à construção do backend, o facto de trabalharmos com algo novo, obrigou, por si só, à realização de muita pesquisa e capacidade de superação. *Soft Skills* que serão de extrema utilidade para o nosso futuro. No que toca aos conteúdos, consideramos que a utilização de Spring e nomeadamente Rest e Hibernate, foram adições importantes à nossa ‘bagagem’, devido ao facto de estes serem tópicos em voga na comunidade. Tudo o resto não representou uma dificuldade uma vez que a linguagem de programação Java, já nos era familiar.

No **deploy** (Secção 4) a utilização de maven como ferramenta de compilação da aplicação, necessitou de algum tempo e esforço devido a alguns problemas com os quais não estávamos habituado mas que, definitivamente, nos fez apren-

der mais sobre o funcionamento do maven, o que é bastante **positivo**. A opção por docker também não revelou muita dificuldade, já que esta ferramenta já tinha sido utilizada. A (pouca) experiência dizia-nos que o backend iria sofrer uma grande sobrecarga, o que nos levou a construir um balanceador simples cuja função se trata de distribuir carga entre dois backends que estariam a correr em simultâneo. Em modo retrospectiva, o deploy correu como esperado.

Quanto à **Análise de Carga** (Secção 5), tentamos que os testes realizados fossem os mais fidedignos possível, tentando ao máximo eliminar fatores de enviesamento como cache e operações menos custosas. Apesar de não ser a primeira vez a usar o JMeter, colocar variáveis de maneira a tornar o teste mais fidedigno foi um desafio que superamos com sucesso. Analisamos o ponto em que o sistema começa a não conseguir atender todos os pedidos, pois é esse o ponto que queremos focar e no futuro comparar.

A elaboração de um **Manual de Utilização** (Secção 6) serviu para apresentar todas as *views* criadas, explicando o fluxo de utilização da plataforma do ponto de vista de quem a está a utilizá-la pela primeira vez. Neste manual, apresentamos também a maioria das mensagens de erro, de sucesso e de dupla confirmação que vão aparecendo no decorrer da sua utilização.

O Manual de Utilização serviu também para simplificar o processo de **Análise de Usabilidade** (Secção 7), visto que facilmente demonstrávamos os exemplos positivos da Serve.Me através da menção das figuras presentes no manual. No final da Análise, tal como seria de esperar visto que durante todo o processo de design e implementação da interface gráfica fomos tendo em conta todos os princípios, podemos concluir que a nossa aplicação *web* é de alta usabilidade.

Como **Trabalho Futuro**, seria extremamente importante adicionar um painel de administrador de modo a facilitar a inserção de mais classes (para além da única existente, Serviços de Jardinagem e Bricolage) e de mais categorias, visto que neste momento apenas é possível fazê-lo adicionando diretamente na base de dados. Seria também interessante a inclusão de novas funcionalidades na aplicação como 'match' entre um pedido e um prestador, como faz a aplicação *Tinder*. No que toca ao *deploy*, a próxima abordagem seria a replicação da base de dados de modo a trazer redundância à aplicação, **tolerando** assim uma falha. Posteriormente, e de maneira a tolerar uma falha em qualquer ponto da arquitetura, colocar a aplicação com frontend, backend e base de dados replicados em modo **failover** seria notável.