Universidade do Minho
Escola de Engenharia

André Gonçalves Amaral

Desenvolvimento de plataforma e *App* DailyHire

julho de 2021

Desenvolvimento de plataforma e *App* DailyHire
André Gonçalves Amaral
UMinho|2021

Universidade do Minho
Escola de Engenharia

André Gonçalves Amaral

Desenvolvimento de plataforma e *App* DailyHire

Dissertação de Mestrado
Mestrado Integrado em Engenharia e Gestão de Sistemas de Informação

Trabalho efetuado sob a orientação da
Professora Doutora Ana Alice Baptista

junho de 2021

DIREITOS DE AUTOR E CONDIÇÕES DE UTILIZAÇÃO DO TRABALHO POR TERCEIROS

Este é um trabalho académico que pode ser utilizado por terceiros desde que respeitadas as regras e boas práticas internacionalmente aceites, no que concerne aos direitos de autor e direitos conexos.

Assim, o presente trabalho pode ser utilizado nos termos previstos na licença abaixo indicada.

Caso o utilizador necessite de permissão para poder fazer um uso do trabalho em condições não previstas no licenciamento indicado, deverá contactar o autor, através do RepositóriUM da Universidade do Minho.

Licença concedida aos utilizadores deste trabalho

Atribuição

CC BY

https://creativecommons.org/licenses/by/4.0/

Universidade do Minho, 22 de julho de 2021

André Gonçalves Amaral

DECLARAÇÃO DE INTEGRIDADE

Declaro ter atuado com integridade na elaboração do presente trabalho académico e confirmo que não recorri à prática de plágio nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração. Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

Universidade do Minho, 22 de julho de 2021

André Gonçalves Amaral

RESUMO

Esta dissertação enquadra-se no projeto EMPOWER-SSE, focando-se na conceptualização e desenvolvimento de uma estrutura *Web* Semântica e na tecnologia *Linked Data*, estando em conta com as ideologias da economia social e solidária. Este projeto é financiado pelo Programa Indo-Português de Cooperação em Ciência e Tecnologia, lançado pela Fundação para a Ciência e Tecnologia (FCT) e pelo Departamento de Ciência e Tecnologia da Índia (DST).

De forma a orientar todo o processo de desenvolvimento, foram utilizados os seguinte procedimentos metodológicos: *Design Science Research* como metodologia de investigação e *Feature Driven Development* como método para desenvolvimento do software.

Com esta dissertação objetivou-se o desenvolvimento de uma solução capaz de aproximar trabalhadores não qualificados dos seus potenciais clientes. Para tal, desenvolveu-se uma plataforma *Web* e *mobile*, capazes de anunciar serviços e de trabalhadores candidatarem-se à sua execução, estabelecer contacto entre cliente e trabalhador, exportar dados *Linked Data* e ainda gerir os serviços anunciados. As plataformas encontram-se num repositório, que permite a qualquer pessoa obter o código que constitui as ferramentas. É também possível a utilizadores externos acederem à plataforma *Web* através da internet, não estando apenas disponível localmente.

Para finalizar, o trabalho desenvolvido surge com o intuito de preencher um vazio no mercado, onde ferramentas já existentes ou obtêm um valor monetário por colocar os trabalhadores em contacto com os clientes ou não possuem a capacidade de exportar dados *Linked Data*. Posto isto, os resultados obtidos demonstram a capacidade das plataformas responderem aos objetivos e resultados esperados.

PALAVRAS-CHAVE

Economia Social e Solidária, *Linked Data*, *Web* Semântica

ABSTRACT

The project of this dissertation is part of the EMPOWER-SSE project, focusing on the conceptualization and development of a Semantic Web structure and Linked Data technology, taking into account the ideologies of the economy social and solidarity. This research project is funded by the Indo-Portuguese Science and Technology Cooperation Program, launched by the Foundation for Science and Technology (FCT) and the Indian Department of Science and Technology (DST).

In order to guide the entire development process, methods were used, such as Design Science Research as a research methodology and the Feature Driven Development methodology for software development. Other methods were also used, such as the validation of functional requirements and usability tests.

With this dissertation, the objective was to develop a solution capable of bringing unskilled workers closer to their potential customers. To this end, a web and mobile platform has been developed, capable of announce services and workers applying for their execution, establishing contact between customer and worker, exporting Linked Data and also managing the services announced. The platforms are in a repository, which allows anyone to obtain the code that constitutes the tools. It is also possible for external users to access the Web platform through the internet, not only being available locally.

Finally, the work developed appears to fill a gap in the market, where existing tools either obtain a monetary value for putting workers in contact with customers or do not have the ability to export Linked Data. That said, the results obtained demonstrate the platforms' ability to respond to the objectives and expected results.

KEYWORDS

Social and Solidarity Economy, Linked Data, Semantic Web

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

LISTA DE ABREVIATURAS, SIGLAS E ACRÓNIMOS

API – *Application Programming Interface*

CSS – *Cascading Style Sheets*

DSR – *Design Science Research*

DER – Diagrama Entidade Relacionamento

DTD – *Document Type Definition*

EJS - *Embedded JavaScript Templates*

ESS – Economia Social e Solidária

FDD – *Feature Driven Development*

HTML – *HyperText Markup Language*

HTTP – *Hypertext Transfer Protocol*

JS – *JavaScript*

JSON – *JavaScript Object Notation*

LD – *Linked Data*

NPM – *Node Package Manager*

SI – Sistema de Informação

UI – *User Interface*

URI – *Uniform Resource Identifier*

UML – *Unified Modeling Language*

WWW – *World Wide Web*

XML – *eXtensible Markup Language*

1. INTRODUÇÃO

Neste capítulo é efetuado o enquadramento, assim como os principais motivos que levaram à realização deste projeto, presente na secção 1.1. Posteriormente é apresentado na secção 1.2, o problema, objetivo e resultados esperados desta dissertação. Por último, é definida na secção 1.3 a estrutura do documento, onde de uma forma breve refere os capítulos que o compõe.

1.1. Enquadramento e Motivação

Esta dissertação enquadra-se no âmbito do projeto EMPOWER-SSE. Este projeto é financiado pelo Programa Indo-Português de Cooperação em Ciência e Tecnologia, lançado pela Fundação para a Ciência e Tecnologia (FCT) e pelo Departamento de Ciência e Tecnologia da Índia (DST). O projeto concentra-se na conceptualização e desenvolvimento de uma estrutura baseada na *Web* Semântica e na Tecnologia de Dados Abertos (*Linked Data*) para consolidar e organizar atividades de agentes do setor não organizado, rumo ao desenvolvimento de uma economia social e solidária, também apelidada de terceiro setor.

Neste âmbito foi desenvolvido uma plataforma que pretende facilitar a contratação de trabalhadores não qualificados, dando assim a possibilidade de os aproximar de potenciais clientes. Pretende-se que esta plataforma, denominada de DailyHire, possua uma vertente *Web* e uma vertente móvel (*App*).

Estudantes indianos do *National Institute of Technology* em colaboração com docentes e investigadores da Universidade do Minho, desenvolveram um perfil de aplicação. Este perfil de aplicação, necessitou de uma atualização e adaptação, de modo a adequar o mesmo ao projeto desta dissertação. A aplicação desenvolvida neste projeto difere de outras aplicações já existentes, uma vez que possui a capacidade de exportar dados *Linked Data*, assim como, a possibilidade de trabalhadores candidatarem-se à execução de um serviço, sem que a plataforma tenha qualquer benefício monetário nessa transação.

1.2. Problema, Objetivo e Resultados Esperados

Tendo em conta o enquadramento referido na secção 1.1, o problema deste projeto corresponde à dificuldade de aproximar trabalhadores não qualificados dos seus potenciais clientes, uma vez que estes trabalhadores apresentam dificuldade em expor os seus serviços e a obter um contacto direto com os seus clientes. De forma a solucionar este problema objetivou-se o desenvolvimento de uma solução *Web* e *mobile*, capaz de estabelecer um contacto entre cliente e trabalhador. Tendo em conta o objetivo definido, foram definidos os seguintes resultados esperados:

- Disponibilizar uma plataforma intuitiva, tanto em versão *Web* como *mobile*, capaz de anunciar serviços e realizar a gestão dos mesmos, estabelecer um contacto entre trabalhador e cliente e a capacidade de candidatar-se à execução de um serviço anunciado.
- Adaptar e atualizar o perfil de aplicação elaborado pelos estudantes indianos do *National Institute of Technology* em colaboração com os docentes e investigadores da Universidade do Minho, de modo, a adaptar o mesmo à dissertação em questão.
- Disponibilizar uma área capaz de exportar os dados como *Linked Data*, tendo em conta os dados definidos como públicos no perfil de aplicação. Estes dados deverão ser obtidos da base de dados no momento da sua requisição.
- Disponibilizar todo o código desenvolvido nesta dissertação, tanto da plataforma *Web* como *mobile*, num repositório, de forma a possibilitar a sua utilização por outras pessoas.
- Capacidade de expor a plataforma *Web* na internet e não apenas localmente. Para tal, é necessário a aquisição de um domínio e de uma plataforma que aloja o *Web Service*.

1.3. Estrutura do documento

1. **Introdução:** É apresentado o tema desta dissertação, inicialmente através do enquadramento e as suas motivações, sendo de seguida, descrito o problema, objetivo e resultados esperados. Para finalizar, é apresentada a estrutura do documento de forma a facilitar a sua compreensão.
2. **Contextualização:** Consiste na contextualização desta dissertação, onde inicialmente são apresentados os temas abordados ao longo de todo o desenvolvimento, seguido pela explicação do projeto EMPOWER-SSE, o qual enquadra o trabalho desenvolvido nesta dissertação. Por último, são apresentadas outras plataformas ou aplicações já existentes no mercado.
3. **Materiais e Métodos:** Este capítulo encontra-se dividido em duas partes. Inicialmente são apresentadas as plataformas e ferramentas utilizadas no desenvolvimento desta dissertação. Numa segunda parte, são descritas as metodologias utilizadas para o desenvolvimento da dissertação. Como metodologia de investigação será utilizada a metodologia *Design Science Research* (DSR), sendo por sua vez, para o desenvolvimento do *software* é utilizada a metodologia Agile com uma abordagem *Feature Driven Development* (FDD). Nesta secção, são ainda apresentados os métodos para a validação dos requisitos funcionais e testes de usabilidade.
4. **Resultados da *Initial Modeling*:** Este capítulo corresponde à primeira etapa da metodologia FDD, responsável por elaborar um modelo que detalhe todo o sistema e a lista de funcionalidades. Para tal, é apresentada a arquitetura tecnológica, juntamente com o diagrama entidade relacionamento (DER), demonstrando como o armazenamento dos dados é efetuado. Por último, é apresentada a validação dos requisitos funcionais.
5. **Resultados da *Model Storming*:** Este capítulo corresponde à segunda etapa do FDD, composto pelas restantes fases desta metodologia. É assim efetuada a apresentação do funcionamento do sistema por funcionalidade.
6. **Testes de Usabilidade:** Apresenta os resultados dos testes efetuados ao sistema, utilizando *personas* e cenários, com o objetivo de identificar erros e, melhorias a implementar futuramente.
7. **Conclusão:** Este capítulo, apresenta as considerações finais acerca do trabalho desenvolvido, assim como, o trabalho futuro a realizar.

2. CONTEXTUALIZAÇÃO

Neste capítulo, é apresentada a contextualização desta dissertação, abordando as temáticas presentes no seu desenvolvimento. Encontra-se dividido em quatro secções, sendo que na secção 2.1 é descrita a temática Economia Social e Solidária. Na secção 2.2 é apresenta a temática *Linked Data*. Na secção 2.3 é apresentado o projeto EMPOWER-SSE, finalizando na secção 2.4 com a descrição das plataformas e aplicações existentes.

2.1. Economia Social e Solidária

"A Economia Social e Solidária é um movimento que pretende promover a mudança em todo o sistema social e económico, defendendo um paradigma de desenvolvimento diferente, assente nos valores e princípios da economia solidária: Direitos Humanos, Democracia, Solidariedade, Inclusão, Diversidade, Desenvolvimento Sustentável, Igualdade, Equidade e Justiça". Está assente na ideia de solidariedade e reciprocidade, tendo em conta tanto os interesses individuais como coletivos, ou seja, a satisfação das necessidades de indivíduos e comunidades em todo o mundo (RIPESS, 2015).

Como objetivo principal deste movimento, define-se a criação de condições de vida dignas para toda a população. A economia reconhece as necessidades existentes das pessoas invés de criar mais necessidades, tornando a Economia Social e Solidária (ESS) uma realidade, pois é possível criar as condições necessárias para estabelecer políticas nacionais e internacionais. Com as pessoas no princípio da sua política, é um instrumento de coesão social e de luta contra a pobreza. É possível afirmar que a ESS coloca as pessoas e o ambiente em primeiro lugar, ao contrário do dinheiro e do lucro, possibilitando às pessoas o acesso a bens, recursos e serviços. A ESS promove ainda um comportamento mais justo em relação ao consumo: incentiva trocas, a reutilização, reparação, compras justas do ponto de vista social, ambiental e económico (Marques, 2017).

De forma a aprofundar e obter um melhor entendimento sobre esta temática irei abordar os conceitos de Economia Social e Economia Solidária uma vez que é segundo estes que se rege a ESS.

A Economia Social, identifica-se como as atividades econômicas que não têm como objetivo apenas a obtenção de lucro. Também apelidada de terceiro setor, esta corresponde ao conjunto de entidades da sociedade civil com fins públicos e não lucrativos (Wikipedia, 2019).

Tendo como ideais a solidariedade e o desenvolvimento integrado da comunidade e do Homem. Nesta sequência de ideias, a Economia Social ou Terceiro Sector pode eventualmente

substituir a ação do Estado ou ser um prolongamento deste na implementação de suas políticas sociais. Segundo Malta (2014), "A Economia Social nasce para resolver problemas sociais que o Estado não consegue resolver". De uma forma simplista, existem dois tipos de organizações pertencentes a este grupo. As organizações que funcionam como empresas (embora o lucro não seja o principal objetivo) e as organizações privadas (suportadas por donativos, trabalho voluntário, etc.).

A Economia Solidária, assenta na ideia de solidariedade e igualdade, em contraste com o individualismo presente nos comportamentos económicos das sociedades. Caracterizada pela ideia de que, pessoas com menor poder económico não significa obrigatoriamente menos capacidades, oferecendo assim, o direito de executar tarefas mais exigentes e aliciantes. Promove a igualdade entre os seus membros, o que resulta, numa filosofia inversa ao capitalismo. "Trata-se de uma forma de organização da produção, consumo e distribuição de riqueza centrada na valorização do ser humano e não do capital, caracterizada pela igualdade, tendo assim como objetivo um comércio mais justo" (Malta, 2014).

Ao contrário da economia capitalista, centrada sobre o capital a ser acumulado e que funciona a partir de relações competitivas, tendo por objetivo interesses individuais, a Economia Solidária organiza-se a partir de fatores humanos, favorecendo as relações onde o laço social é valorizado, através da reciprocidade (Henriques, 2010).

"A Economia Solidária apoia-se sobre uma economia de sujeitos desiguais, enquanto que a Economia Social é, pelos seus princípios e regras, uma economia de iguais" (Lechat, 2002). A Economia Solidária, sozinha não é capaz de definir a sociedade igualitária à qual aspira, sendo que a Economia Social, também não parece ser capaz de resolver o problema crescente das desigualdades. A solução que apresenta, passa por definir experiências de sociedades igualitárias, usando-as como base, aparecendo como um horizonte possível ou provável da Economia Solidária. Resumidamente, as duas economias podem ser consideradas complementares e que as vantagens de cada, possam se reforçar (Lechat, 2002).

Podemos assim afirmar que a ESS tem como objetivo, tornar a venda de produtos e serviços o mais justo possível para todos os elementos da cadeia, o que permite dar um maior valor aos produtores que muitas vezes eram explorados por intermediários. Com isto, é possível estabelecer relações diretas entre o próprio consumidor final e os fornecedores, o que poderá levar a uma redução ou até mesmo remoção da influência de algum agente intermediário. Outro fator relevante da ESS, é o facto de os preços já estarem previamente estabelecidos e assim evitar que os mesmos subam com a inflação.

2.2. *Linked Data*

Originária nos anos 90, a *Word Wide Web* (WWW), tinha como objetivo ser um sistema de comunicação entre computadores, que possibilitaria a troca de informações, em particular na época da Guerra Fria. Para tal, seria necessário existirem interfaces amigáveis e intuitivas, para aceder ao crescente repositório de documentos. Com este avanço tecnológico, assim como, as suas vastas formas de utilização, requereram uma nova abordagem (Souza & Alvarenga, 2004).

Derivado do seu próprio nome, pode-se assim entender que *Web* Semântica está relacionada com a WWW e ainda com semântica, sendo que esta por sua vez está relacionada com a compreensão da natureza do significado (Allemang & Hendler, 2011).

A informação na *Web* é tipicamente representada em linguagem natural permitindo assim às pessoas a sua compreensão. A *Web* Semântica tem como objetivo representar a informação, de uma maneira na qual, computadores sejam capazes de interpretá-la. Através desta representação, a pesquisa em *Web* Semântica propõe tecnologias para automação, integração e reuso da informação (Devedzic, 2006).

A introdução da *Web* Semântica implica a transformação progressiva da *Web* baseada nos documentos. As hiperligações entre documentos fundamentaram a primeira geração da *Web*, sendo que a criação de hiperligações entre dados ou informação, origina uma base de dados distribuída à escala da *Web* (Heath & Bizer, 2011).

Segundo Souza & Alvarenga (2004), a solução para alcançar as ideias definidas, encontra-se na padronização de tecnologias, de linguagens e metadados. Com isto, é possível os utilizadores da *Web*, baseando-se em regras comuns, armazenar e descrever dados. A padronização, possibilita a utilização e interpretação da informação, de modo, a ser utilizada por utilizadores humanos, ou não, de maneira automática e não ambígua. “Um documento na *Web* é composto por uma mistura de dados e metadados. "Meta" é um prefixo de autorreferência, de forma que "metadados" sejam "dados sobre dados"”. A sua função passa por especificar características dos dados que descrevem, a sua forma de utilização, como devem ser apresentados, ou até mesmo o seu significado em determinado contexto.

Segundo Bizer (2009), “Nos últimos anos, as principais fontes de dados da *Web*, como a *Google*, *Yahoo*, *Ebay* e *Amazon* começaram por fornecer acesso às suas bases de dados através da *Web* API”. A disponibilidade dos dados e a qualidade dos mesmos, graças às *Application Programming Interface* (API), possibilitaram o desenvolvimento de aplicações que combinam dados de diferentes fontes. A maioria das API, não atribuem identificadores únicos aos seus

dados, o que torna impossível definir *hyperlinks* para dados provenientes de diferentes API. Isto resulta, numa *Web* dividida, com bases de dados separadas, o que obriga aos desenvolvedores a escolher um conjunto de bases de dados para a sua aplicação. Com o objetivo de superar esta fragmentação, Tim Berners-Lee definiu um conjunto de práticas recomendadas para a publicação e conexão de dados estruturados na *Web*, "Os Princípios de *Linked Data*". "Uma vez que a *Web* de *Linked Data* tem como base padrões para identificação, recuperação e representação de dados, é possível usar browsers para explorar por completo o espaço de dados". Os dados provenientes de diferentes fontes, são conectados por *links*, o que possibilita a sua agregação e consulta, bastante semelhante à forma como uma base de dados é consultada atualmente.

Como referido anteriormente, *Linked Data* é caracterizado como um conjunto de práticas para publicar e interligar dados estruturados na *Web*. As práticas definidas por Tim Berners-Lee, são apresentadas de seguida:

1. "Usar URIs para nomear as coisas";
2. "Usar URIs HTTP, para que as pessoas possam procurar esses nomes";
3. "Quando alguém procurar um URI, forneça informações úteis, usando os padrões (RDF, SPARQL)";
4. "Incluir links para outros URIs, de modo a que seja possível descobrir mais coisas".

Uniform Resource Identifiers (URI), corresponde a um mecanismo de identificação global, utilizado para o desenvolvimento de documentos *Web*. É também necessário a utilização do mecanismo de acesso universal, *Hypertext Transfer Protocol* (HTTP) e, *Hypertext Markup Langueage* (HTML), como o formato do conteúdo presente nas páginas *Web*. "A *Web* está assente no princípio de manter *hyperlinks* entre documentos da *Web*" (Heath & Bizer, 2011).

Tendo em conta Heath e Bizer (2011), estes descrevem os quatro princípios da seguinte forma. "O primeiro princípio de *Linked Data* corresponde ao uso de referências de URI para identificar, não apenas documentos da *Web* e conteúdo digital, mas também objetos do mundo real e conceitos abstratos".

Face ao segundo princípio, o protocolo HTTP, corresponde a um mecanismo de acesso universal à Web. Assim sendo, o uso de URI HTTP tem como objetivo identificar objetos e conceitos abstratos, o que possibilita que esses URI sejam consultados no protocolo HTTP sobre uma descrição do objeto ou conceito identificado.

Uma vez que o conteúdo presente na web possui uma enorme variedade, é necessário concordar em formatos de conteúdo padronizado, possibilitando a que diversas aplicações

conseguiam interpretar esse mesmo conteúdo. Posto isto, o formato acordado de documentos *Web* corresponde ao HTML. O terceiro princípio, assenta na ideia de que, é necessário o uso de um único modelo de dados para a publicação de dados estruturados na *Web*. Este modelo, denominando de *Resource Description Framework* (RDF), é baseado em grafos e foi desenvolvido com o intuito de ser usado em contexto *Web*.

Relativamente ao último princípio de *Linked Data*, este indica para o uso de *hyperlinks* para conectar qualquer coisa e não apenas documentos. Por exemplo, é possível usar *hyperlinks* para conectar duas pessoas. "Em contraste com a Web clássica, onde os *hyperlinks* não apresentam um tipo, os *hyperlinks* em contexto de *Linked Data* possuem tipos que descrevem a relação entre as coisas" (Souza & Alvarenga, 2004). "*Hyperlinks* em contexto de *Linked Data* são chamados de *links* RDF para distingui-los dos *hyperlinks* entre documentos clássicos da *Web*" (Souza & Alvarenga, 2004).

"Vários servidores por toda a *Web*, são responsáveis por responder a solicitações de URIs HTTP em vários *namespaces* diferentes, retornando descrições RDF dos recursos identificados pelos URIs. Isto permite, que num contexto *Linked Data*, um *link* RDF ao conectar URIs em *namespaces* diferentes, conecta recursos em bases de dados diferentes" (Souza & Alvarenga, 2004).

O RDF, padroniza metadados para ser embutido na codificação *Extensible Markup Language* (XML). "A sua implementação é exemplificada pelo RDF *Schema*, ou RDFS, que faz parte da especificação do padrão". A sua ideia, está assente na descrição dos dados e metadados, através de um esquema de triplos, definidos através de sujeito, predicado e objeto. Na figura 1, é apresentado um exemplo de um código XML (Souza & Alvarenga, 2004).

```
<?xml version="1.0"?>
    <rdf:RDF
        xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:dc="http://purl.oclc.org/DC#"
        xmlns:v="http://ww.w3.org/2001/vcard-rdf/3.0#">
    <rdf:Description about="http://www.ucla.edu/~einstein"/>
        <dc:Creator>
            <rdf:Description about=" http://www.ucla.edu/staff/einstein"/>
                <v:Name> Isaac Einstein</v:Name>
                <v:Email> einstein@ucla.edu</v:Email>
                <v:Orgname>UCLA</v:Orgname>
                <v:Orgunit>Department of Physics</v:Orgunit>
        </dc:Creator>
    </rdf:Description>
    </rdf:RDF>
```

Figura 1 - Exemplo de um código XML. Retirado de (Souza & Alvarenga, 2004)

Na figura apresentada anteriormente, é possível visualizar um exemplo de um código XML que utiliza três *namespaces* diferentes respetivamente na segunda, terceira e quarta linha de código (o *namespace* do padrão RDF, o do padrão *Dublin Core* e a especificação de *Vcards* que padroniza a descrição dos dados geralmente encontrados em um cartão de visita). Uma vez especificado o *namespace*, é possível a sua utilização ao longo de todo o documento XML, referenciado sempre qual está a ser utilizado. É possível utilizador inúmeros *namespaces*, como o da especificação *Dublin Core* ou específicos como o do padrão Vcard, o que possibilita que os metadados estejam sempre disponíveis, caso seja necessário a utilização de um vocabulário controlado para descrever determinada informação. "O padrão RDF, as ontologias e os *namespaces* compartilhados permitem que qualquer indivíduo ou organização publique informações em sites *Web* de forma que produtos de *software* ou agentes possam interpretar a informação marcada semanticamente e agir sobre esta informação de forma mais inteligente" (Souza & Alvarenga, 2004).

Linked Data, assim como, a *Web* clássica, utilizam os *hyperlinks* com o objetivo de conectar dados de diferentes fontes num único espaço. Por exemplo, uma aplicação através de um URI, obtém dados RDF descrevendo uma pessoa. Através dos *links* presentes nos dados RDF, é possível seguir o seu caminho em diversos servidos *Web*, tendo assim a capacidade de descrever o local onde a pessoa mora ou a empresa que trabalha, etc. Este processo pode ser replicado, onde seguindo os novos *links* RDF, é possível navegar para novos conjuntos de dados, obtendo informações sobre a cidade onde a pessoa mora (Heath & Bizer, 2011).

Bizer et al. (2011) afirmam que, *Linked Data* depende de duas tecnologias fundamentais para a *Web*, sendo estas:

- "O protocolo HTTP, fornece um mecanismo simples e universal na recuperação de recursos serializados como um fluxo de *bytes* (por exemplo, uma fotografia de um cão) ou na recuperação de descrições de entidades que não podem por si só atravessar a rede dessa forma (por exemplo, o cão em si)".
- "O modelo RDF, codifica os dados sob a forma de triplos, sendo estes, sujeito, predicado e objeto. O sujeito e o objeto são caracterizados como URIs que identificam um recurso e o predicado, também representado por um URI, identifica como o sujeito e o objeto se relacionam".

Segundo Bizer et al. (2007), o sujeito de um triplo corresponde a um URI do recurso descrito. Em relação, ao objeto, este pode apresentar um valor literal, como uma *string*, número

ou data, ou até mesmo um URI de outro recurso relacionado com o objeto em questão. Por último, o predicado, também ele um URI, proveniente de vocabulários, *collections* de URIs, possibilita a representação da informação sobre determinado domínio.

Podem ser distinguidos dois principais tipos RDF, triplos literais e *links* RDF:

- **Triplos Literais:** Estes apresentam um RDF literal no objeto. Tal como referido anteriormente, poderá apresentar o valor de *string*, número ou data. Triplos literais são geralmente usados para descrever as propriedades dos recursos, tais como, nomes, data de nascimento, entre outros.
- ***Links* RDF:** Representam o relacionamento entre dois recursos, consistindo em três referências de URI. Os URIs apresentados no sujeito e no objeto identificam os recursos relacionados, enquanto o URI no predicado define o tipo de relacionamento entre os recursos.

Na Figura 2, é possível verificar que existe uma pessoa identificada pelo URI "http://www.w3.org/People/EM/contact#me", denominada Eric Miller, o seu endereço eletrónico "em@e3.org" e título académico "Dr.".

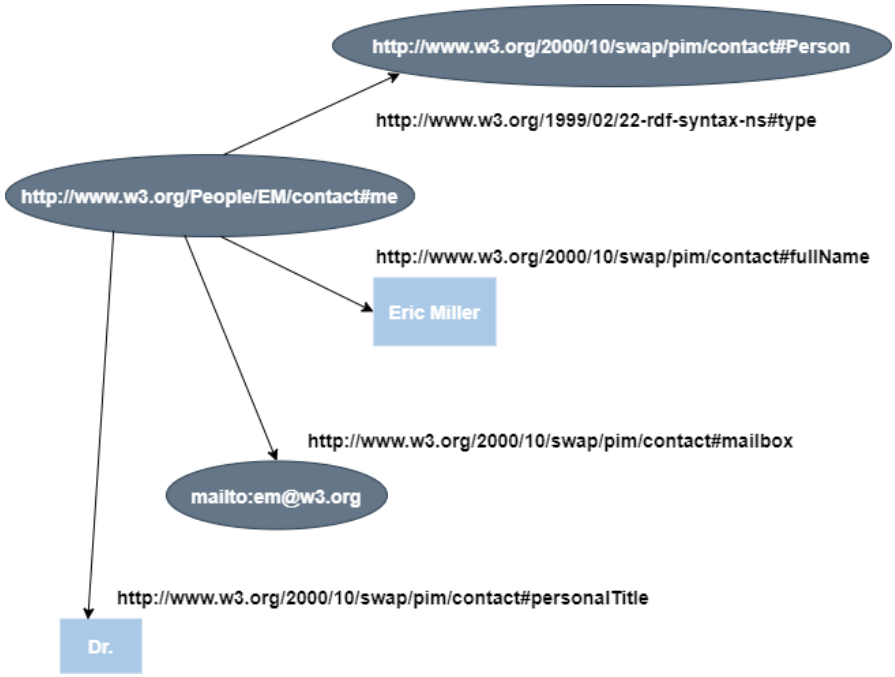

Figura 2 - Grafo RDF descrevendo Eric Miller. Retirado de (Firmino, 2013)

O RDF pode ser caracterizado como "uma linguagem para representação de informação sobre recursos na *Web*. É particularmente indicada para representação de metadados relativos a recursos *Web*, tais como título, autor e data de modificação de uma página *Web*, ou ainda

direitos de autor e licença de utilização de informações de um documento *Web*" (Firmino, 2013).

A Figura 3, demonstra um exemplo de *links* RDF. "O primeiro *link*, *http://www.w3.org/People/Berners-Lee/card#i*, indica que o URI do recurso é membro de outro recurso chamado *http://dig.csail.mit.edu/data#DIG*". Posto isto, aquando referenciado pelo protocolo HTTP, "o servidor *dig.csail.mit.edu* responde com uma descrição RDF do recurso identificado, neste caso o Grupo de Informações Descentralizadas do MIT". O URI do predicado *http://xmlns.com/foaf/0.1/member* "produz uma definição do *link* do tipo *member*" (Bizer, Heath & Berners -Lee, 2011).

Figura 3 - Exemplo de *links* RDF. Retirado de (Bizer, Heath & Berners -Lee, 2011)

Atualmente, um dos maiores exemplos da utilização de *Linked Data*, corresponde ao projeto *Linking Open Data* (LOD). Este projeto, objetiva o lançamento de *datasets*, conjunto de dados, o que por sua vez, origina uma nuvem de dados estruturada, apesar da sua existência pela vasta *Web* (Bauer & Kaltenböck, 2011).

Segundo Bizer et al. (2011), o foco do projeto LOD, passa pela identificação de *datasets*, disponíveis sob licenças livres, convertendo-os para o formato RDF, tendo como base os princípios de *Linked Data* previamente referidos, e posteriormente a sua publicação na *Web*. Com o crescimento e a natureza livre do projeto, possibilitou a acessibilidade de qualquer pessoa na publicação de um *dataset*, se o mesmo se regesse pelos princípios de *Linked Data* e interligado com *datasets* já existentes. Na Figura 4, é possível encontrar um diagrama em nuvem do projeto LOD, possibilitando uma melhor perceção da tamanho e escala da *Web* de dados. Cada nó presente no diagrama, corresponde a um *dataset* distinto, sendo que as ligações entre os diversos nós, representam as relações existentes entre os dois *datasets*. As ligações, apresentam diferentes espessuras, sendo que quanto maior, maior será o número de relações existentes entre os dois *datasets*. Por sua vez, ligações bidirecionais indicam a existência relações externas para outro *dataset*.

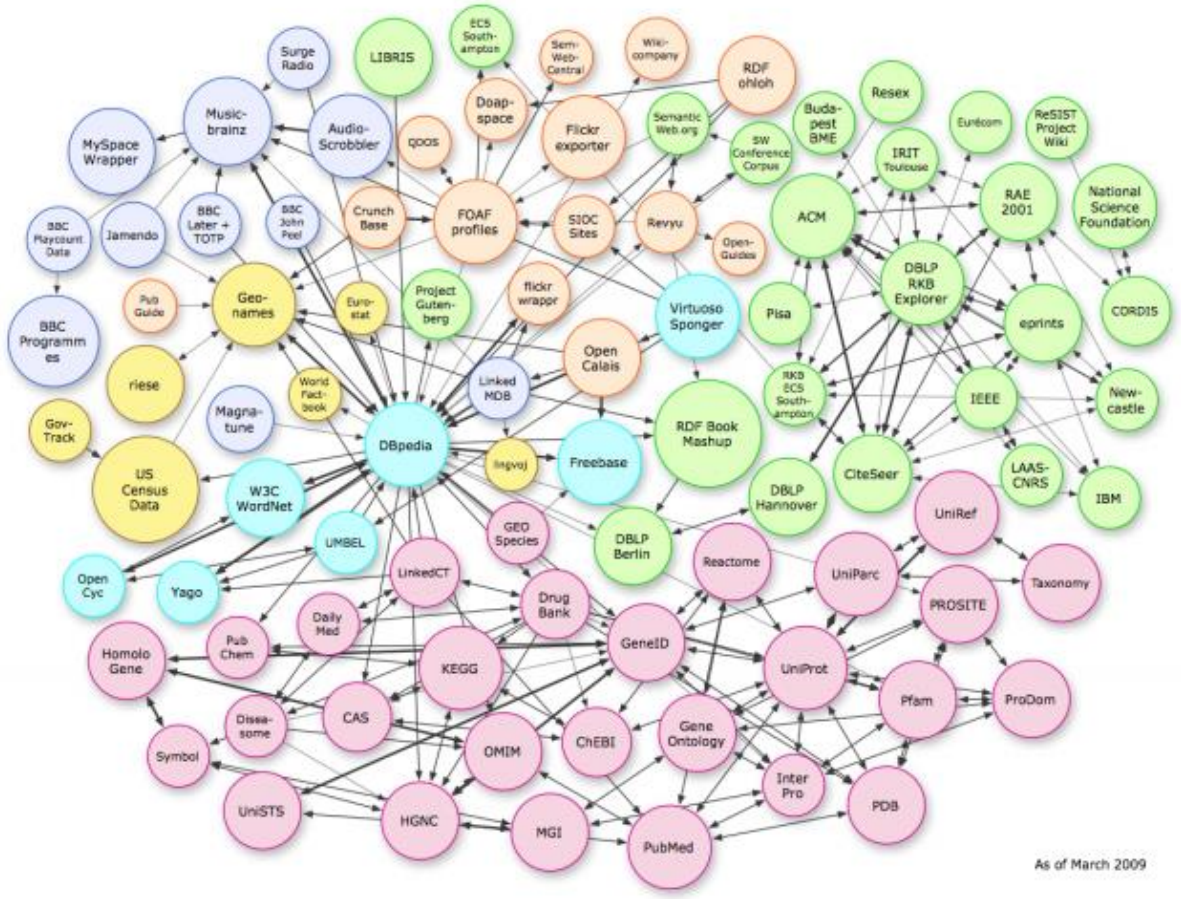

Figura 4 - Diagrama em nuvem da *Linking Open Data*. Retirado de (Bizer, Heath & Berners-Lee, 2011)

Um exemplo relativo à transformação de *datasets* em formato RDF, correspondendo a um *Linked Dataset*, é o *DBPedia*. Responsável pela transformação do conteúdo da página Wikipédia, não só inclui dados na página como também incorpora ligações de outros *datasets* existentes na *Web*. Posto isto, aplicações na sua fase inicial, possuem a capacidade de explorar informações existentes noutros *datasets*, assim como, integrar factos existentes nos mesmos (Firmino, 2013).

2.3. Projeto EMPOWER-SSE

O projeto EMPOWER-SSE, financiado pelo Programa Indo-Português de Cooperação em Ciência e Tecnologia, lançado pela Fundação para a Ciência e Tecnologia (FCT) e pelo Departamento de Ciência e Tecnologia da Índia (DST). Tem como objetivo, estabelecer uma relação entre a *National Institute of Technology*, *B.P. Poddar Institute Of Management And Technology* e a Universidade do Minho. Concentra-se na conceptualização e desenvolvimento de uma estrutura baseada na *Web* Semântica e na Tecnologia de Dados Abertos (*Linked Data*), consolidando e organizando atividades de agentes do setor não organizado, rumo ao desenvolvimento de uma economia social e solidária, também apelidada de terceiro setor. "Os setores não organizados apresentam dificuldades com a falta de informação, partilha de dados e interoperabilidade entre sistemas e fronteiras organizacionais e perdem os benefícios/lucros do negócio, uma vez que a maioria não alcança o fim da cadeia" (Sen et al., 2018).

Estudantes indianos do *National Institute of Technology* em colaboração com docentes e investigadores da Universidade do Minho, desenvolveram um perfil de aplicação e uma lista de funcionalidades, apresentadas nos anexos (Matriz de Restrições Obsoleta, Lista de Funcionalidades Obsoleta).

"Um perfil de aplicação de metadados (MAP) é um constructo que fornece um modelo semântico para publicação de dados na *Web* de Dados". O modelo semântico pode ser caracterizado como um modelo de dados, definindo propriedades e as suas restrições, sendo que cada propriedade é apresentada como um termo do vocabulário RDF, indicando o seu domínio, contradomínio e a sua cardinalidade. Isto permite, satisfazer necessidades específicas das aplicações, proporcionando interoperabilidade semântica com outras aplicações. O seguimento de um MAP, ou seja, de um conjunto de regras de publicação de dados como LOD, possibilita a que os dados publicados na LOD *cloud*, sejam processados automaticamente por agentes de *software* (Malta, 2020).

Tanto o perfil de aplicação como a lista de funcionalidades, foram sujeitos a uma atualização, de modo, a adaptar os mesmos à dissertação desenvolvida. É possível encontrar o perfil de aplicação atual nos apêndices (Matriz de RestriçõesMatriz de RestriçõesMatriz de Restrições), e a lista de funcionalidades na secção 4.4.

2.4. Plataformas e Aplicações Existentes

Foi realizado um *Benchmark* ao mercado com o objetivo de descobrir e avaliar aplicações, que se relacionassem com a temática desta dissertação. Um dos motivos, de não se ter recorrido à reutilização de código aberto de outras ferramentas, é não depender a aplicação desenvolvida de plataformas externas, ou seja, a aplicação estaria sujeita ao funcionamento de outras ferramentas.

A analise efetuada, permite criar uma base de como as funcionalidades requisitadas pela DailyHire deverão operar e, potenciais funcionalidades extra que possam ser adicionadas ao sistema, de forma a alcançar o sucesso. Para tal, utilizando o motor de pesquisa *Google*, foram pesquisadas ferramentas, utilizando as seguintes palavras-chave: aplicações para economia social e solidária e, aplicações de contratação de serviços, durante o mês de maio de 2020. Após a pesquisa realizada, selecionaram-se aplicações que se assemelhassem ao pretendido para a plataforma Dailyhire. Esta seleção, teve em conta os seguintes critérios:

- Possibilidade de requisitar um serviço em determinada área profissional, selecionando o trabalhador pretendido.
- Entrar em contacto com os diversos trabalhadores registados na plataforma.
- Possibilidade de um trabalhador se registar na plataforma.
- Possibilidade de obter dados *Linked-Data*.

Uma vez selecionadas aplicações tendo em contas os critérios anteriormente definidos, foi reduzida a amostra para apenas cinco aplicações, uma vez que, várias aplicações se assemelhavam no seu funcionamento, não introduzindo vantagens ao estudo em questão.
Posto isto, é necessário efetuar uma análise às funcionalidades de cada aplicação, sendo necessário registar um utilizador nas diversas. De seguida, é possível encontrar uma descrição de cada aplicação avaliada.

- A *ESSApp* é possível encontrar tanto em plataforma *Web* como *mobile*, tendo sido desenvolvida por *open software* (GCOOP). Esta permite conectar possíveis clientes com organizações de economia social e solidária, fornecedoras de produtos e serviços, na qual é possível encontrar a sua localização geográfica num mapa, com a possibilidade de aplicar diversos filtros. Possui mais de 3000 organizações, sendo que para além da sua localização, é ainda possível descobrir mais informações acerca das mesmas, como os seus projetos e os seus contactos. Por último, existe ainda uma área dedicada a noticias e um calendário de eventos, de forma a informar os utilizadores dos acontecimentos mais recentes

relativamente à economia social e solidária. Face aos seus aspetos negativos, podemos indicar, a sua disponibilidade apenas na Argentina, possuindo organizações unicamente dessa região. A inexistência de uma funcionalidade capaz de informar o serviço pretendido por parte de um cliente e a dificuldade em registar novos trabalhadores.

- A *Susy* é uma plataforma *Web*, que que tal como a *ESSApp*, possui um mapa interativo, apresentado organizações registadas por toda a Europa. Este mapa permite aplicar diversos filtros, sendo que todas as organizações apresentadas possuem uma descrição. Na descrição de cada organização, existe uma opção de *Linked Data*, sendo que no momento da análise não se encontrava em funcionamento. Esta plataforma possui também uma área dedicada à economia social solidária. Relativamente aos aspetos negativos, a inexistência de uma funcionalidade a requisitar um serviço, assim como, a dificuldade no registo de novos trabalhadores, sendo que para tal, é necessário entrar em contacto com a plataforma por e-mail.
- A *TaskRabbit* é uma plataforma *Web*, disponível também para *android* e *iOS*, que permite entrar em contacto com *taskers*. *Taskers*, é o nome atributo pela ferramenta aos seus trabalhadores, uma vez que estes, podem efetuar serviços não caracterizados por uma profissão, por exemplo, ajudar alguém na realização de mudanças entre imóveis, a montagem de um móvel *IKEA*, etc. A nível de funcionalidades, é possível indiciar a tarefa desejada, através de um conjunto de questões. É também possível realizar o pedido de uma tarefa, sem a necessidade de se registar na plataforma. A nível do trabalhador, este necessita de indicar a sua tarifa, ou seja, o valor a cobrar pelo seu serviço, sendo que a *TaskRabbit* retira uma comissão desse valor. O trabalhador, tem também, a possibilidade de indicar a sua disponibilidade num calendário, assim como, especificar o seu leque de competências, entre outras.
- A *Zaask*, possibilita a contratação de um profissional em diversas áreas. Disponível em *Web*, *android* e *iOS*, esta aplicação permite ao utilizador responder a um conjunto de perguntas com o objetivo de apresentar trabalhadores que se enquadrem nos parâmetros indicados. É apresentado o orçamento, sendo que a *Zaask* não possui uma comissão sobre este, obtendo rendimento através da venda de créditos aos trabalhadores. Estes créditos são necessários para o trabalhador contactar o cliente e, se disponibilizar a executar o serviço. Um trabalhador é assim obrigado a possuir créditos na sua conta, caso contrário, não tem a possibilidade de contactar clientes, resultando na impossibilidade de executar serviços. É também importante mencionar, que o facto de entrar em contacto com os clientes, não significa obrigatoriamente que este será o trabalhador selecionado.

〉 A Fixando, é uma plataforma bastante semelhante à *Zassk*, operando da mesma maneira a nível dos créditos. Também disponível em *Web* e *mobile*, esta plataforma possibilita a contração de serviços. A nível de funcionalidades, o cliente tem a possibilidade de analisar o trabalhador e avaliações efetuadas ao mesmo, permite o *upload* de imagens para indicar o serviço necessário ou o método tradicional de responder a um de conjunto questões, possibilita a troca de mensagens entre cliente e trabalhador, entre outros. Uma funcionalidade de salientar, é a possibilidade de qualquer utilizador ter permissão de visualizar os serviços requisitados, tendo acesso ao tipo de serviço requisitado, assim como, a sua localização, o que para muitos utilizadores poderá ser bastante incômodo.

	ESSApp	**Susy**	**TaskRabbit**	**Zaask**	**Fixando**
Localizar trabalhadores num mapa	Sim	Sim	Não	Não	Não
Exportar dados como *Linked Data*	Não	Não	Não	Não	Não
Contactar trabalhadores através da plataforma	Não	Não	Sim	Sim	Sim
Requisitar um serviço	Não	Não	Sim	Sim	Sim
Registar-se como trabalhador	Sim	Não	Sim	Sim	Sim
Versão *mobile* e *Web*	Sim	Não	Sim	Sim	Sim

Tabela 1 - Funcionalidades das ferramentas e aplicações existentes

3. MATERIAIS E MÉTODOS

O objetivo deste capítulo passa por explicar as plataformas e ferramentas utilizadas no desenvolvimento desta dissertação, abordada na secção 3.1, assim como, os procedimentos metodológicos que orientaram o desenvolvimento, apresentada na secção 3.2.

3.1. Plataformas e Ferramentas Utilizadas

Um dos critérios para a seleção das plataformas e ferramentas foi a possibilidade de implementação de melhorias futuras ao sistema, bem como a adição de extensões sem grandes restrições. Tendo isto em conta, foi necessário verificar os tipos de ferramentas que estão disponíveis, os benefícios que as mesmas possuem e as implicações da sua utilização. A escolha das ferramentas a utilizar num projeto possui um papel importante, já que pode ditar o sucesso ou o fracasso do mesmo.

Neste sentido, apresentam-se em seguida, as ferramentas selecionadas com uma breve explicação de cada, assim como, o motivo que levou à sua escolha.

3.1.1. React Native

React native corresponde a uma *framework* desenvolvida pelo Facebook, utilizada para desenvolvimento de aplicações nativas em *Javascript* tanto para *iOS* como para *android*. Baseada em *ReactJS*, introduz alguns conceitos para desenvolvimento de interfaces do utilizador, sendo que alguns desses conceitos podem ser aplicados não só em navegação web como em mobile. Apesar das aplicações serem desenvolvidas em *Javascript*, estas são compiladas em código *Native*, o que possibilita uma melhor performance do que as chamadas apps híbridas (Novick, 2017).

Para além do referido anteriormente, selecionou-se *React Native* para o desenvolvimento da app devido à possibilidade de utilizar elementos nativos da *user interface* (UI), bem como a execução de forma separada da principal UI *thread*. Este último aspeto proporciona à aplicação manter uma elevada performance e, ainda, a possibilidade de desenvolver apenas uma aplicação que é suportada tanto por *iOS* como *android*.

3.1.2. NodeJS

Conhecido como *Node*, corresponde a um ambiente *JavaScript* do lado do servidor. Implementado principalmente em C e C++, apresenta como foco a performance e o baixo consumo de memória, objetivando suporte de processos de servidor de longa duração (Tilkov & Vinoski, 2010).

O *NodeJS* revela benefícios que facilitaram a tomada de decisão na escolha desta ferramenta para desenvolvimento do servidor. Uma dessas vantagens é a sua flexibilidade e escalabilidade, uma vez que utiliza como gestor de bibliotecas externas o *Node Package Manager* (NPM). O NPM corresponde ao maior repositório de *softwares* do mundo, o que torna o *NodeJS* uma plataforma com potencial para ser utilizada em qualquer situação. Outras vantagens tidas em conta foram o seu alto desempenho e a capacidade de lidar com vários *requests* em simultâneo, assim como a sua rápida curva de aprendizagem, uma vez que utiliza *JavaScript* sendo uma linguagem padrão para desenvolvimento *Web*. Esta linguagem será assim utilizada tanto para o *frontend* como para o *backend*, o que pode representar ganhos no que respeita à reutilização de código (Lenon, 2018).

3.1.3. Heroku

Heroku é uma plataforma *cloud* como serviço (PaaS), que suporta diversas linguagens como *Java*, *NodeJS*, *Python*, entre outras. Disponibiliza diversas funcionalidades aos desenvolvedores para implementar, gerir e escalar aplicações, de maneira semelhante pelas várias linguagens suportadas (Heroku, 2021).

Esta ferramenta é gratuita com algumas restrições a nível de recursos, sendo que numa fase inicial deste projeto é suficiente, havendo a possibilidade de no futuro melhorar as características do serviço atual. Pelos motivos referidos anteriormente, aliados à sua fácil utilização, tornaram esta ferramenta a escolhida para realizar o *deploy* do servidor.

3.1.4. MongoDB

MongoDB é uma base de dados *NoSQL* em código aberto, desenvolvida em C++. Trata-se de uma base de dados de armazenamento de documentos onde estes documentos são agrupados em *collections* de acordo com a sua estrutura. Notar que, também é possível armazenar documentos com diferentes estruturas. Os dados são armazenados em documentos *JavaScript Object Notation* (JSON), permitindo que os campos possam variar de documento para documento e a sua estrutura também possa ser alterada ao longo do tempo (Abramova & Bernardino, 2013).

Uma *collection* é caracterizada como um conjunto de documentos do *MongoDB*, sendo que estes documentos podem apresentar diferentes campos. Em comparação com uma base de dados relacional, uma *collection* é o equivalente de uma tabela.

Posto isto, sendo uma base de dados orientada a documentos e não relacional, torna--se mais fácil a escalabilidade. Esta escalabilidade é possível pois corresponde a ferramenta flexível, não possuindo uma estrutura predefinida, facilitando a remoção e adição de campos. Permite um desenvolvimento rápido pois simplifica a experiência de novos modelos e a escolha da melhor opção (Chodorow, 2013).

Além disso, o *MongoDB* fornece um cluster gratuitamente com 512Mb de armazenamento, satisfazendo as necessidades de armazenamento nesta fase do projeto. Futuramente, caso haja a necessidade em expandir este armazenamento, o *MongoDB* também oferece essa capacidade. Todas as características expostas, possuíram um forte impacto na escolha da ferramenta *MongoDB*.

3.1.5. *Embedded JavaScript Templates*

EJS é uma linguagem simples de *templates* que permite gerar código HTML com *JavaScript*. Através destes *templates*, capazes de serem reutilizados, é possível gerar páginas com algumas funcionalidades iguais sem a necessidade de repetir essa parte do código. Para tal, basta escrever apenas uma vez o *template* e invocar o mesmo pelas diversas páginas, sempre que necessário. Compreende-se, portanto, o rápido desenvolvimento que esta linguagem oferece, bem como a sua velocidade de execução. Além disso, o facto de se utilizar a linguagem *JavaScript* no desenvolvimento do servidor, revela uma outra vantagem na utilização da EJS para desenvolvimento da parte *Web*.

De forma a complementar esta linguagem, serão utilizadas outras componentes como *Cascading Style Sheets* (CSS) e *Bootstrap* focadas na parte estética da aplicação. Por sua vez, as componentes *JavaScript* (JS) e *JQuery* focam-se na comunicação com o servidor, criação de conteúdo dinâmico e na integração de bibliotecas externas como a API do *Google* e do *AlgoliaPlaces*.

3.1.6. OnlyDomains

A *OnlyDomains*, é uma plataforma que permite adquirir domínios, tornando as páginas *Web* acessíveis para toda a WWW. Para tal, é necessário que o utilizador efetue a compra de um domínio. O motivo pela qual esta plataforma foi selecionada, foi o preço do domínio em utilização, “daily-hire.net”, sendo inferior em relação às restantes plataformas avaliadas. Um

domínio, corresponde ao endereço que os utilizadores digitam num *browser*, de modo, a visitar a página *Web* pretendida.

3.1.7. GitHub

O *GitHub* corresponde a uma "plataforma de hospedagem de código para controlo de versão e colaboração". Permite trabalhar com outras pessoas no mesmo projeto a partir de qualquer lugar. Para isso é necessário criar um repositório que irá possuir todos os ficheiros do projeto, podendo conter pastas, ficheiros, vídeos, conjunto de dados, etc. Esta plataforma possibilita assim, que os utilizadores divulguem os seus projetos e que outros utilizadores contribuam para o mesmo, através de recursos que relatam problemas ou incorporem outros repositórios.

3.2. Procedimentos Metodológicos

Nesta secção, são apresentados os procedimentos metodológicos que orientaram o desenvolvimento desta dissertação. Apresentam os prossupostos a ter em conta e a seguir, tendo em conta o método científico. Posto isto, esta secção encontra-se dividida nas seguintes subsecções, *Design Science Research* apresentado na subsecção 3.2.1, *Feature Driven Development* na subsecção 3.2.2, Validação dos Requisitos Funcionais na subsecção 3.2.3 e, por último, *Goal-Directed Design* na subsecção 3.2.4.

3.2.1. Design Science Research

A metodologia de investigação utilizada para o desenvolvimento da dissertação é a metodologia DSR. Esta metodologia concentra-se no desenvolvimento de artefactos tendo como base um processo de investigação científica. A Figura 5, apresenta uma representação gráfica dos 4 ciclos, sendo estes *Relevance*, *Rigor*, *Design* e por último *Change & Impact*, apesar de que este último ciclo não será abordado nesta dissertação, mas com a possibilidade de no futuro, vir a ser estudado noutro projeto.

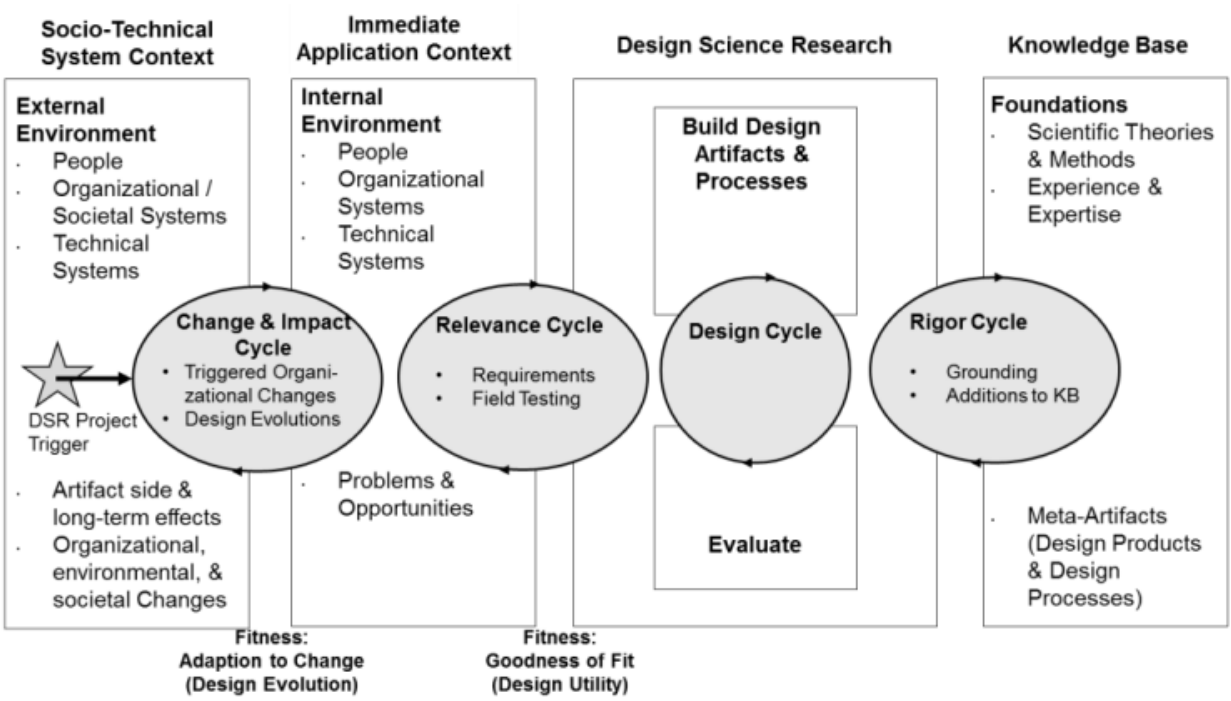

Figura 5 - Quatro ciclos do DSR. Retirado de (Drechsler & Hevner, 2016)

"Design Science Research é motivado pelo desejo de melhorar o ambiente pela introdução de novos e inovadores artefactos e o processo de construção destes artefactos". Tendo isto em conta, o DSR deverá se iniciar com a identificação de oportunidades e problemas

no ambiente de aplicação e, a solução proposta pelo artefacto que visa ao melhoramento do ambiente. O ciclo *Relevance*, conecta o ambiente contextual do projeto com as atividades do DSR, ou seja, conecta o ambiente ao artefacto. O domínio de aplicação consiste nas pessoas, sistemas organizacionais e sistemas técnicos, que interagem para alcançar um objetivo. Este ciclo, não só fornece os requisitos necessários para a realização da pesquisa, assim como, define os critérios de avaliação para os resultados obtidos (Hevner, 2007). Contextualizando este ciclo com o projeto desenvolvido, é definido o seu contexto de aplicação, caracterizado neste caso como a EMPOWER-SSE. Neste ambiente, deparamo-nos assim com os trabalhadores não qualificados, e todo este grupo, que apresenta dificuldades no contacto com possíveis clientes.

O ciclo *Rigor* conecta as atividades do DSR com a base de conhecimento de fundamentos científicos e experiências que sustentam o projeto de investigação e asseguram as suas inovações. Pesquisa prévia de sistemas de informação (SI) e resultados de disciplinas de referência, proporcionam teorias fundamentais, modelos e métodos usados na fase de desenvolvimento do projeto. As metodologias fornecem um guia, usado na fase de avaliação e justificação, sendo que o rigor só é alcançado com a correta aplicação destas metodologias e fundamentos (Hevner et al., 2004). "Simultaneamente, a avaliação do artefacto deve contribuir rigorosamente para estas bases de conhecimento, capturando o que funciona, o que não funciona e como os resultados das avaliações se encaixam e estendem as teorias e experiências existentes". Em suma, este ciclo possui a base de conhecimento, suportada por processos e artefactos, apresentados na contextualização anteriormente definida, assim como, nos processos metodológicos apresentados nesta secção, que visam à orientação, na construção e avaliação do artefacto.

Relativamente ao ciclo *Design*, este itera entre as atividades principais do desenvolvimento e avaliação de um artefacto e os processos de pesquisa. Este ciclo é o núcleo do qualquer projeto DSR, onde são geradas alternativas de *design* e feita a avaliação destas alternativas, até que um design satisfatório seja alcançado. Este ciclo, requer informação proveniente dos ciclos *Relevance* e *Rigor*, os métodos e teorias delineados nessas etapas (Hevner, 2007). De salientar, que a metodologia FDD, definida na secção 3.2, deverá ser implementada neste ciclo, orientando a vertente desenvolvimento do projeto.

Por último, o ciclo *Change & Impact*, é uma extensão à visão de três ciclos do DSR que pretende capturar uma melhorar dinâmica do artefacto para o contexto do mundo real. "Este ciclo abrange os impactos de segunda ordem dos artefactos para o vasto contexto organizacional e social". É necessário distinguir o contexto imediato de um artefacto, correspondendo aos utilizadores diretos do artefacto dentro do ambiente. Relacionando este ciclo com o projeto em

si, a *App*, os telemóveis, os clientes e trabalhadores que utilizam a *App* seriam o contexto imediato. Por sua vez, o seu contexto mais amplo correspondia às alterações que o projeto tivesse nos trabalhadores do terceiro setor. Este ciclo, não está incluído nesta dissertação, uma vez que só após o projeto estar desenvolvido, é possível verificar o impacto do mesmo, dando a possibilidade de no futuro o projeto ser estudado e avaliado e que alterações proporcionou (Drechsler & Hevner, 2016).

3.2.2. Feature Driven Development

Feature Driven Development é um método de desenvolvimento *agile* de *software*, sendo enquadrado no ciclo de *Design* do DSR. Divide-se em duas etapas, apresentadas como *Initial Modeling* e *Model Storming*. *Inital Modeling* é composto pelas duas primeiras fases da metodologia, sendo o *Model Storming* composto pelas restantes três. O uso desta metodologia suporta o desenvolvimento de um sistema de forma mais rápida, pois, possibilita a divisão de processos complexos em processos mais simples (Chowdhury & Huda, 2011).

Como é possível visualizar na figura 6, a metodologia FDD consiste em cinco processos sequenciais, focando-se num desenvolvimento por funcionalidade.

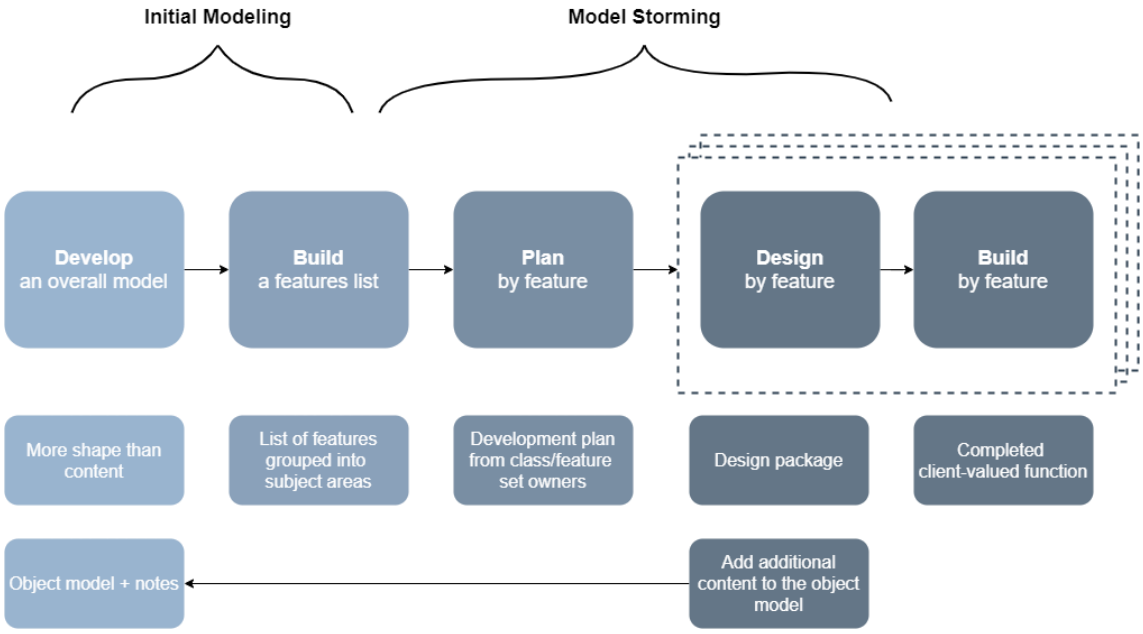

Figura 6 - Fases da metodologia FDD. Retirado de (Lucidchart, 2021)

Numa primeira fase, é necessário desenvolver o modelo detalhado, que funciona como um esboço de todo o sistema. Para tal, foi desenvolvido uma arquitetura tecnológica apresentando a finalidade e relações entre as diferentes tecnologias, assim como, um modelo entidade relacionamento, representando um modelo lógico do armazenamento dos dados.

Passando para a segunda fase, esta envolve a realização de uma lista detalhada de todas as funcionalidades necessárias tanto da aplicação *Web* como da *App*, sendo que se uma funcionalidade for demasiado complexa, esta deve ser dividida em funcionalidades de menor dimensão, para ser possível analisar o projeto e avaliar os seus progressos. Com o decorrer do projeto, caso alguma funcionalidade seja adicionada, essa deve ser tida em conta no modelo desenvolvido na fase anterior. Nesta etapa, foram avaliadas todas as funcionalidades existentes, através de uma validação de requisitos. Na secção seguinte, é contextualizado a validação de

requisitos, detalhando como a mesma foi realizada, sendo posteriormente apresentado os seus resultados.

Uma vez identificadas todas as funcionalidades necessárias, iniciamos a segunda etapa. Esta etapa é composta pelas três fases seguintes, onde a terceira fase, corresponde à avaliação da complexidade de cada funcionalidade e realizado o planeamento de como devem ser desenvolvidas essas funcionalidades. É também nesta fase que são atribuídas as funcionalidades pelos diversos desenvolvedores. O planeamento deverá ser incremental, ou seja, uma funcionalidade de cada vez. Nesta fase, é necessário determinar a ordem na qual as funcionalidades devem ser desenvolvidas e posteriormente implementadas, tendo em conta os seus riscos, dependências, carga de trabalho e qualquer outro obstáculo que possa interferir com o seu desenvolvimento. Neste projeto, esta fase não foi utilizada, pois o desenvolvimento foi efetuado por apenas uma pessoa, não sendo necessário a atribuição de funcionalidades. Face à ordem de desenvolvimento, esta foi efetuada consoante as necessidades existentes no momento de desenvolvimento. Isto permitiu agilizar o processo de desenvolvimento, pois permitiu saltar entre funcionalidades quando ocorrem adversidades, evitando assim perdas de tempo, sendo algo fulcral tendo em conta o prazo estabelecido para a entrega da dissertação.

Na quarta fase, "*Design by Feature*", é efetuado o detalhe por funcionalidade, ou seja, cada funcionalidade é analisada em particular e é criado um diagrama de sequência detalhado. O que indica os passos a seguir no desenvolvimento de uma funcionalidade e explicando o seu funcionamento.

Por último, na quinta fase são implementados os recursos necessários que darão suporte à etapa anterior, começando por estudar a informação proveniente dessa etapa, para que de seguida, seja possível começar a construção das funcionalidades. É iniciada a fase de desenvolvimento, sendo este desenvolvimento efetuado por funcionalidade e posteriormente os respetivos testes, de forma a verificar se foi possível responder aos requisitos propostos.

3.2.3. Validação dos Requisitos Funcionais

Validação de requisitos, trata-se de um processo, que tal como o próprio nome indica, possibilita a validação da consistência, precisão, contextualização de requisitos levantados. Este processo, permite corrigir incoerências e inconformidades durante o desenvolvimento do produto. A identificação destas incoerências, permite minimizar o risco de as encontrar numa fase já tardia, por exemplo, durante o término do desenvolvimento do produto, podendo a sua alteração não chegar a ser exequível ou demasiado dispendiosa (Wikipedia, 2020).

Uma vez definidos os objetivos do projeto, foi elaborada uma lista de requisitos funcionais que as plataformas *Web* e *mobile* deveriam cumprir. De forma a validar estes mesmos requisitos, foi convidado o professor Ricardo Sant'Anna, membro da EMPOWER-SSE, a participar numa reunião via Zoom de cerca de duas horas, onde se apresentou e discutiu os requisitos funcionais. Com isto, pretendeu-se obter a opinião do professor relativamente aos requisitos e à respetiva forma de implementação. Além disso, era importante percecionar qual a visão do docente e ideias que pudesse indicar, que constituíssem uma mais valia para o projeto.

A experiência do professor com agricultores e as plataformas que usam no seu quotidiano, bem como a sua experiência profissional, constituíram os fatores considerados na sua seleção. De momento, encontra-se como docente na Universidade Estadual Paulista "Júlio de Mesquita Filho" (UNESP), tendo sido anteriormente consultor na *Granol*.

A validação de requisitos, constituirá um elemento importante na elaboração da lista de requisitos final, necessária na segunda fase da metodologia FDD. Esta lista, contribuirá no desenvolvimento futuro das funcionalidades, atuando como a base do desenvolvimento.

3.2.4. *Goal-Directed Design*

O método *Goal-Directed Design*, foi utilizado para realizar os testes de usabilidade ao sistema. A necessidade de realizar testes de usabilidade está bastante interiorizada em todos aqueles que desenvolvem software. Usabilidade é caracterizada como uma qualidade que deve estar inerente ao documento, possibilitando que os utilizadores usem com satisfação, eficácia e eficiência na realização de tarefas. Um software pode estar corretamente desenvolvido a nível de funcionalidades, mas se a sua usabilidade não for boa, o utilizador rejeitá-lo-á (Carvalho, 2002).

"Teste de usabilidade é uma técnica de pesquisa, utilizada para avaliar um produto ou serviço. Os testes são realizados com usuários representativos do público-alvo. Cada participante tenta realizar tarefas típicas enquanto o analista observa, ouve e anota" (Voltapo, 2014).

De forma, a obter os melhores resultados, é necessário que os utilizadores representem uma *persona* para quem o produto é destinado. O número ideal de utilizadores para a realização de testes é de cinco, uma vez que, por muito que as pessoas possuam visões diferentes, os seus comportamentos básicos se assemelham. Os resultados, começam-se a repetir e poucas informações novas surgirão (Vieira, 2019).

A forma mais eficiente de entender o que funciona e o que não funciona numa interface, é observar pessoas durante a utilização. Com isto, é possível obter perceções do que poderá causar problemas aos utilizadores, e ajudar a melhorar as interfaces. Para tal, em vez de ordenar os utilizadores a executar determinada tarefa sem qualquer explicação, é necessário situar os mesmos com um cenário. Este cenário, contextualiza sobre o motivo que leva o utilizador a executar determinada tarefa (McCloskey, 2014).

Como referido anteriormente, é necessário que os utilizadores representarem uma *persona*. *Personas* são descritas como personagens fictícias que representam os comportamentos, emoções, atributos, motivações e objetivos do consumidor final do produto. Uma vez que não é possível alcançar todos os utilizadores finais, é necessário criar um modelo que os represente. Após a criação das *personas*, é necessário o desenvolvimento dos cenários. Estes correspondem a histórias sobre as *personas*. A história pretende descrever como uma *persona* em particular, completa uma determinada tarefa ou se comporta face a uma determinada situação (Courage & Baxter, 2005).

4. RESULTADOS DA *INITIAL MODELING*

Este capítulo engloba as duas primeiras fases da metodologia FDD, caracterizado como *Initial Modeling*. Inicialmente, é necessário a elaboração de um modelo que detalhe o funcionamento de todo o sistema. Para tal, é apresentado na secção 4.1, a arquitetura tecnológica, responsável por contextualizar o funcionamento do sistema e as relações existentes entre as várias ferramentas e plataformas. De forma a complementar a arquitetura, é apresentada na secção 4.2, o modelo entidade relacionamento, demonstrando como é efetuado o armazenamento de dados que visa a suportar ao sistema. Após a elaboração dos modelos, passamos para uma segunda fase, responsável pela elaboração da lista de funcionalidades resultante da validação dos requisitos funcionais, apresentada na secção 4.3.

4.1. Arquitetura Tecnológica

De forma a proporcionar um melhor entendimento do funcionamento das plataformas, apresenta-se, neste capítulo, a sua arquitetura tecnológica (Figura 10). Nesta constam as ferramentas utilizadas e as comunicações efetuadas entre si, detalhando tecnologicamente com o intuito de explicar a sua implementação.

O processo inicia-se com a interação do utilizador, ou na página *web* ou na *app*, que, por sua vez, comunica com o servidor *NodeJS*. Este é responsável por toda a lógica do sistema, encontrando-se hospedado numa plataforma como serviço, através da ferramenta *Heroku*. Esta permite expor o conteúdo do servidor para a internet, e não apenas localmente. Nas interfaces de utilizador recorre-se a duas API, da *Agolia Places* e da *Google*, permitindo estender o leque de funcionalidades das plataformas. Torna-se possível, recomendar a morada ao utilizador, considerando os caracteres inseridos pelo mesmo, bem como a conversão dessa mesma morada em coordenadas geográficas. Permite ainda a apresentação, num mapa, dos trabalhadores de acordo com as informações inseridas por estes.

A comunicação do servidor *NodeJS* com a plataforma *Web* é gerida pelo Express, correspondendo a uma *framework* do *NodeJS*, utilizada para controlar os pedidos HTTP. Uma vez efetuado o controlo dos pedidos, é utilizado o *Embedded JavaScript Templates* (EJS) para a criação de *templates* reutilizáveis. Responsável por gerar as páginas *web* em conjunto com as tecnologias de desenvolvimento *Web*. Destas tecnologias, CSS e Bootstrap possuem funções estéticas enquanto JS e *JQuery* focam-se na criação de conteúdo dinâmico, integração de bibliotecas externas e na comunicação com o servidor, em formato JSON. É possível aceder à plataforma *Web*, através do domínio adquirido na plataforma *OnlyDomains*, definido como

"www.daily-hire.net". Um domínio não funciona por si só, necessita de um serviço que aloje os ficheiros da página *Web*. Para tal, como mencionado anteriormente, é utilizado o *Heroku*, sendo que, juntos possibilitam o acesso à página *Web* em toda a WWW. Relativamente à *App*, esta foi desenvolvida inteiramente em *React Native*, tanto a nível de interface como na comunicação com o servidor.

Após a comunicação com o servidor, é necessário o armazenamento dos dados, sendo para tal utilizado o *MongoDB*. Este armazenamento é efetuado em *collections*, podendo dizer que estas são análogas a tabelas em bases de dados relacionais. Ambas as plataformas possuem acesso à mesma base de dados através das funções definidas no servidor. A utilização das mesmas funções nas duas plataformas permite guardar os dados da mesma forma oferecendo, portanto, uma maior coerência a todo o sistema.

A plataforma *Web* e a *mobile*, encontram-se armazenadas em dois repositórios na plataforma *Github*. Nestes repositórios é possível encontrar todo o código desenvolvido que suporta o funcionamento do sistema, assim como, uma explicação de como deve o utilizador proceder, caso pretenda utilizar as plataformas localmente. Em seguida, é possível encontrar o link do repositório da página Web (https://github.com/AndreAmaralGit/Dailyhire-Web-Page-and-Web-Service) e da App (https://github.com/AndreAmaralGit/DailyHire-App).

De seguida apresenta-se a Figura 7 que sintetiza tudo o que fora mencionado anteriormente, ou seja, expõe de forma clara e objetiva a arquitetura tecnológica do projeto. Além disso, apresenta-se as relações entre os vários tipos de tecnologia utilizadas.

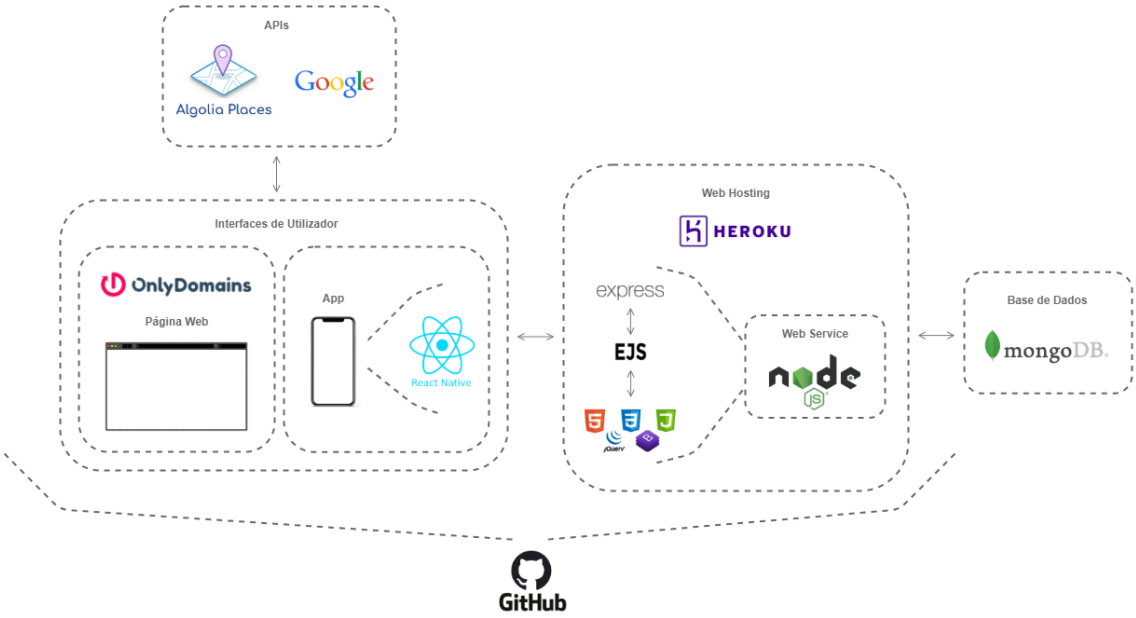

Figura 7 - Arquitetura tecnológica

4.2. Modelo Entidade Relacionamento

Ainda numa primeira fase da *Initial Modeling*, da metodologia FDD, desenvolveu-se este modelo, complementando a arquitetura tecnológica anteriormente apresentada. Assim sendo, é necessário identificar as principais partes e objetos envolvidos, suas possíveis ações e responsabilidades, suas características e como elas interagem entre si. As informações recolhidas, permitem o desenvolvimento de um modelo concetual que visa a orientação do desenvolvimento propriamente dito.

Um modelo ER (Entidade Relacionamento), caracterizado como um modelo de dados, é utilizado para descrever objetos (entidades) e as suas características (atributos), assim como os relacionamentos existentes entre si (relacionamentos). Regra geral, este modelo representa de forma abstrata, a estrutura que possui o sistema de armazenamento de dados (Rodrigues, 2021). Na Figura 8, expõe-se o DER, para um melhor entendimento.

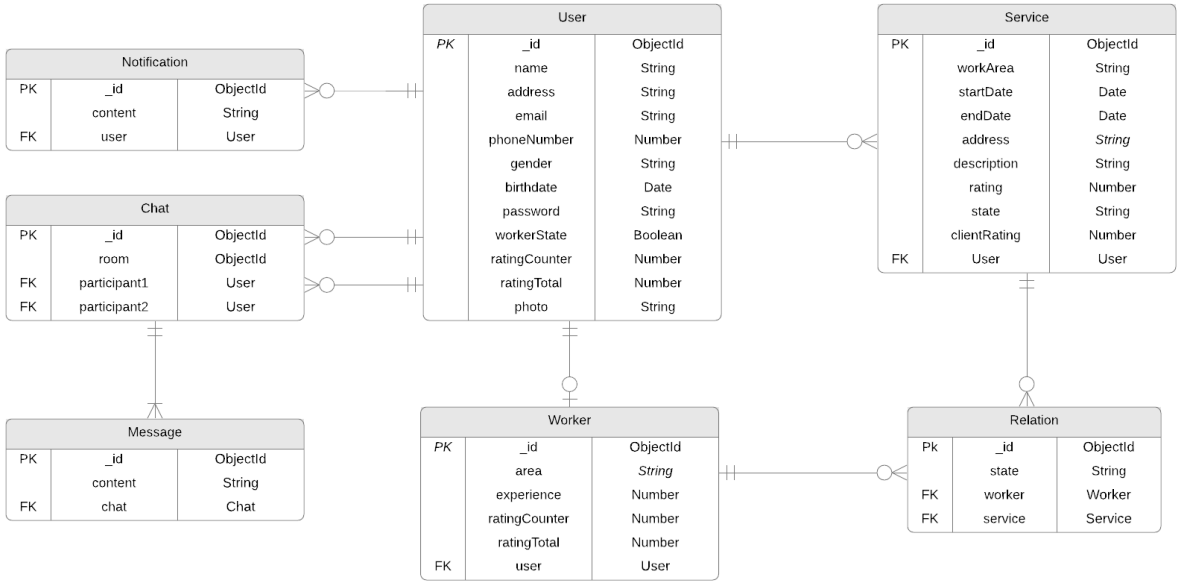

Figura 8 - Diagrama Entidade Relacionamento

Após revelada a figura anterior, são descritas as entidades, sendo que a descrição de cada atributo se encontra nos apêndices (Atributos do Modelo Entidade Relacionamento).

- **User:** Armazena os dados de todos os utilizadores. Apresenta a maioria dos relacionamentos uma vez que todos os dados necessitam de um utilizador associado.
- **Notification:** Armazena as notificações não visualizadas pelos utilizadores.
- **Chat:** Armazena todas as conversas existentes, permitindo ao sistema percecionar quais utilizadores já entraram em contacto com outros.

- **Message:** Armazena todas as mensagens enviadas nas plataformas, associando cada mensagem à respetiva conversa.
- **Worker:** Armazena todos os dados dos trabalhadores. Cada trabalhador está associado aos seus dados de utilizador.
- **Service:** Armazena todos os dados dos serviços anunciados.
- **Relation:** Armazena todas as candidaturas de trabalhadores aos serviços anunciados.

4.3. Validação dos Requisitos Funcionais

Tendo em conta a segunda fase da metodologia FDD, responsável pela elaboração de uma lista de funcionalidades, esta secção é responsável pela aplicação da validação dos requisitos funcionais e elaboração da lista de funcionalidades, que visa a suportar as fases sequentes da metodologia. Como mencionado na subsecção 3.2.3, a validação dos requisitos funcionais foi efetuada pelo docente Ricardo Sant'Anna, onde se apresentou e discutiu os requisitos funcionais.

Assim sendo, expõe-se em seguida, as anotações dadas pelo docente. Numa fase inicial do desenvolvimento do projeto, era requisitado ao trabalhador que, no momento da candidatura a um serviço, informasse o cliente de um orçamento. Uma vez terminada a demonstração, sugeriu-se a remoção dessa opção, já que iria implicar outras questões à aplicação. Estas questões teriam foco na segurança e na dificuldade de um trabalhador em estabelecer um orçamento sem uma avaliação presencial do serviço. De notar, que as aplicações em desenvolvimento não possuem qualquer tipo de interação com as transações entre os utilizadores.

Outra sugestão mencionada, passa pela possibilidade de clientes e trabalhadores entrarem em contacto, ou seja, a questão da comunicação. Existindo a necessidade de um cliente conhecer mais informações e agendar visitas de avaliação do serviço, a opção de troca de mensagens entre utilizadores constitui, naturalmente, uma mais-valia para os intervenientes. Esta funcionalidade tem como objetivo facilitar a escolha de um trabalhador para a execução de um serviço. Face ainda, à falta de informação relativamente aos trabalhadores, foi sugerido a possibilidade de visualizar o *feedback* de outros clientes que já obtiveram serviços desses trabalhadores.

A capacidade de um cliente visualizar no mapa os diversos trabalhadores foi outro ponto discutido na reunião. Neste sentido, sugeriu-se a possibilidade de filtragem por localização, nomeadamente, a distância dos trabalhadores em relação à localização atual do utilizador. Assim sendo, os utilizadores saberiam rapidamente quais os profissionais mais próximos de si, contribuindo para a sensação de proximidade ao cliente.

Após uma análise a todas sugestões apresentas, foi realizada uma avaliação do que seria implementado no projeto. A questão do orçamento foi tida em conta e levada para a prática. Deste modo, atualmente não existe qualquer tipo de interação, por parte da plataforma, com os valores monetários dos serviços prestados. Em relação, à questão da comunicação, a forma como foi abordada na plataforma passa por possibilitar aos utilizadores a opção de enviar uma

mensagem a outros utilizadores do seu interesse, pelas diversas páginas. Adicionou-se ainda, um chat, com a capacidade de trocar mensagem entre os utilizadores, em tempo real, que armazena toda a conversa entre ambos os intervenientes. Em complementação à temática anterior, foi adicionado a possibilidade de avaliar anonimamente tanto o prestador do serviço como o requerente do mesmo, após o término do serviço. Para finalizar, em relação aos trabalhadores apresentados no mapa, adicionou-se a possibilidade de entrar em contacto com os mesmos, assim como, as suas informações. Este mapa também possibilita a aplicação de filtros, como por exemplo, distância do trabalhador em relação à localização atual do utilizador.

Uma vez validados os requisitos, apresenta-se de seguida, a tabela atualizada, que contém todos os requisitos funcionais e a respetiva descrição.

Requisitos Funcionais	Descrição
Criar Utilizador	Registo de um novo utilizador no sistema, sendo inicialmente considerado como cliente. Futuramente, poderá habilitar a área de trabalhador.
Login	Possibilidade de entrar na plataforma, através do e-mail e da *password*.
Anunciar Serviço	Um cliente, assim que especificar todas as características sobre o serviço, como por exemplo local ou duração, …, poderá anunciar o seu serviço, de modo a receber candidaturas de possíveis trabalhadores.
Listar Serviços	Lista de serviços de um utilizador, sendo apresentados consoante a área que este se encontre (Cliente ou Trabalhador). Os serviços podem ser listados como “Pendentes”, “Ativos” ou “Completos”.
Cancelar Serviço	Enquanto um serviço estiver no estado pendente, um cliente tem a possibilidade de o cancelar.
Aprovar Trabalhador	Cada serviço pendente na área de cliente possui a opção de aprovar um trabalhador, responsável por realizar o serviço. São assim apresentados todos os trabalhadores que se candidataram a esse mesmo serviço, sendo que

	após um trabalhador ser aprovado, esse serviço passa a ser considerado como um serviço ativo.
Terminar Serviço	Possibilidade de um cliente dar um serviço como terminado, assim como, avaliar o trabalhador responsável por executar o serviço. Esta avaliação está disponível numa escala de 1 a 5.
Visualizar Perfil	É possível visualizar o perfil de qualquer trabalhador, sendo também possível a um trabalhador, visualizar o perfil de um cliente que tenha anunciado um serviço.
Editar Perfil	Um utilizador tem a possibilidade de alterar o seu perfil a qualquer momento.
Enviar Mensagem	A possibilidade de um utilizador entrar em contacto com outro utilizador.
Visualizar Mensagens	Área onde o utilizador poderá visualizar todas as mensagens trocadas com outros utilizadores.
Enviar Notificações	Alertar o utilizador de qualquer alteração que ocorra num serviço que este esteja associado, assim como de novas mensagens que receba.
Visualizar Notificações	Área onde o utilizador poderá encontrar todas as notificações que possui no momento.
Apresentar Mapa	Neste mapa são apresentados todos os trabalhadores registados. Existe a possibilidade de filtrar por distância em relação ao utilizador, assim como, por área de trabalho.
Exportar *Linked Data*	Exportar *Linked Data*, tendo em conta os dados existentes na base de dados no momento e a matriz de restrições desenvolvida. Apenas são exportados os dados definidos como públicos.

Alterar Modo Trabalhador	Caso um utilizador pretenda prestar serviços, será disponibilizado a opção para alternar entre modo trabalhador e modo cliente. A primeira vez que um utilizador pretenda alterar para o modo trabalhador, será requisitado as informações necessárias.
Listar Serviços Disponíveis	Lista de serviços que um trabalhador se pode candidatar. Estes serviços são apresentados ao trabalhador se a sua profissão for a profissão requisita para o serviço.
Candidatar-se a um Serviço	Dentro dos serviços disponíveis, é possível a um trabalhador candidatar-se à realização do serviço.
Avaliar Cliente	Assim que um cliente terminar um serviço, o trabalhador associado poderá avaliar esse mesmo cliente. Esta avaliação está disponível numa escala de 1 a 5.

Tabela 2 - Lista de requisitos funcionais

5. RESULTADOS DA MODEL STORMING

Neste capítulo, regendo-se pela metodologia FDD, será efetuada a apresentação do trabalho desenvolvido, por funcionalidade. Corresponde à segunda etapa da metodologia, sendo assim apresentada a quarta e quinta fase. Em relação à terceira fase, como mencionado anteriormente, esta não foi implementada no projeto. A quarta e quinta fase, estão agrupadas, de modo, a possibilitar uma melhor compreensão do funcionamento de cada funcionalidade.

As funcionalidades apresentadas neste capítulo, tanto da página *Web* como da *App*, mencionadas anteriormente na secção 4.3, possuem uma explicação do seu funcionamento juntamente com um diagrama de sequência, baseado em *Unified Modeling Language* (UML). Para finalizar, também é possível encontrar imagens, a demonstrar o resultado da implementação de cada funcionalidade.

5.1. Cliente

Qualquer utilizador que se regista na plataforma, inicialmente é considerado como cliente. Com isto, um utilizador, poderá sempre executar funções de cliente, mesmo estando registado como trabalhador. De modo, a exercer funções de trabalhador, o utilizador deve-se registar como tal. Nesta secção, estão presentes as funcionalidades de cliente, ordenando-as consoante a sua de ordem de utilização e semelhança, ou seja, funcionalidades que dependam ou interajam com outras funcionalidades. Algumas funcionalidades nesta secção, também estão presentes na área de trabalhador, sendo que, o motivo para a sua demonstração nesta secção, deve-se ao facto de o objetivo da funcionalidade se assemelhar a uma existente na área de cliente. Por exemplo, apesar da listagem de serviços na área de trabalhador se executar de diferente forma da listagem na área de cliente, o seu objetivo é mesmo, a listagem dos serviços do utilizador, enquanto cliente ou trabalhador.

5.1.1. Criar Utilizador

Esta funcionalidade consiste na criação de contas de utilizadores. Inicialmente quando uma conta é criada, apenas possui funcionalidades de cliente, sendo que para ter acesso às funcionalidades de trabalhador, deverá se registar como tal. Na figura 9, é possível encontrar o diagrama de sequência associado a esta funcionalidade.

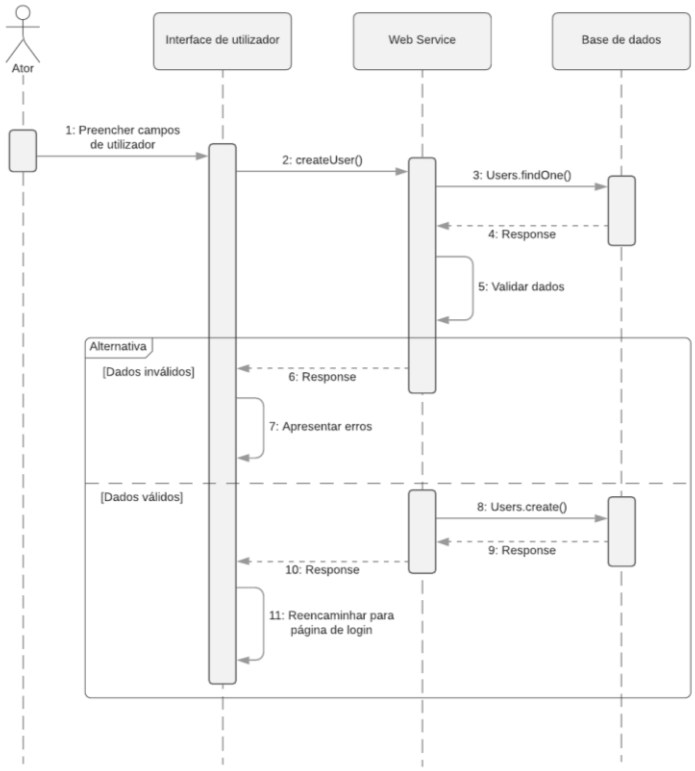

Figura 9 - Diagrama de Sequência: Criar Utilizador

Para a criação de um utilizador, é necessário preencher todos os campos apresentados tanto na Figura 10 como 11. De forma, a aumentar a veracidade dos dados é efetuada a validação dos mesmos, desde verificar se o e-mail inserido já se encontra registado, ou impossibilitando que os campos se encontrem vazios, entre outros. Na página *Web*, é utilizada a API da *AlgoliaPlaces* e a API da *Google* na *App*, com o objetivo de recomendar moradas aquando da escrita por parte do utilizador.

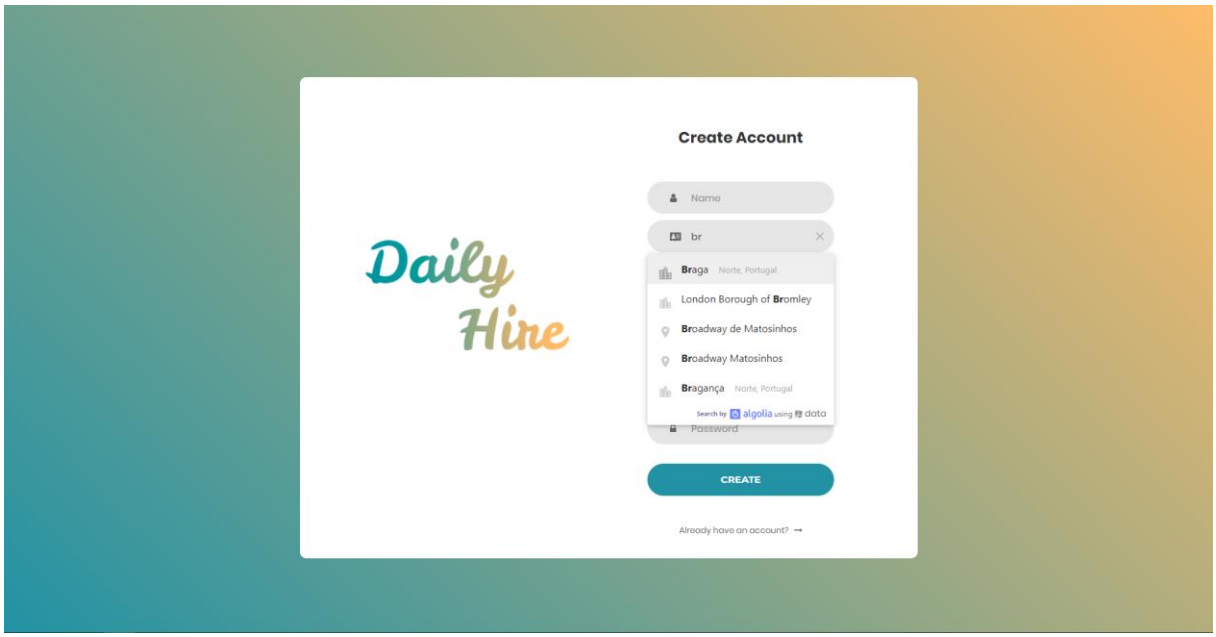

Figura 10 - Funcionalidade: Criar Utilizador - Página *Web*

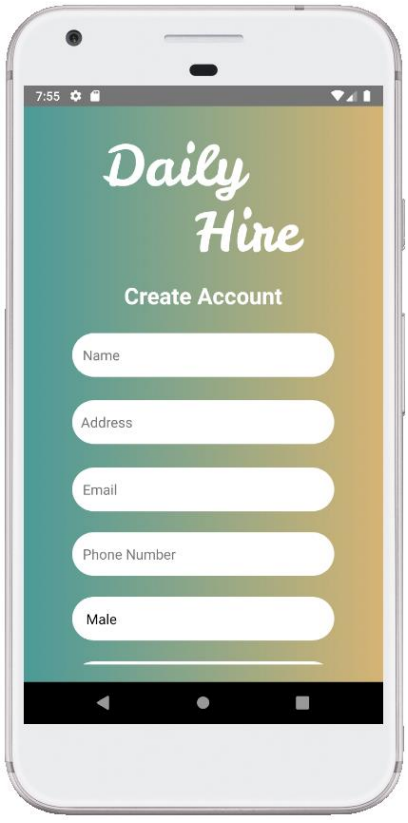

Figura 11 - Funcionalidade: Criar Utilizador - *App*

5.1.2. Login

Esta funcionalidade permite a utilizador aceder à plataforma *Web* ou *App*. Para tal, é necessário ao utilizador introduzir o seu email e sua password. Na figura 12, encontramos o diagrama de sequência responsável por explicar o funcionamento deste processo.

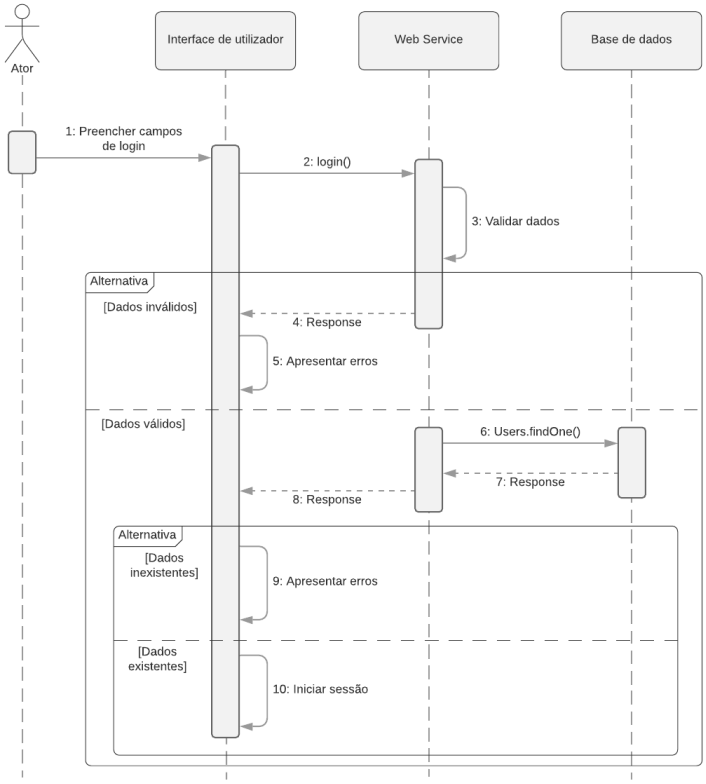

Figura 12 - Diagrama de Sequência: *Login*

Uma vez preenchidos os campos necessários, é efetuada a validação dos mesmos. Inicialmente é verificado se os campos se encontram preenchidos, e posteriormente, é verificado se o e-mail existe na base de dados, informando o utilizador caso o e-mail não seja encontrado. Uma vez encontrado o e-mail, é comparada a *password* inserida com a respetiva *password* dessa conta. Na figura 13 e 14 podemos encontrar, as interfaces *Web* e *App*, responsáveis por executar esta funcionalidade.

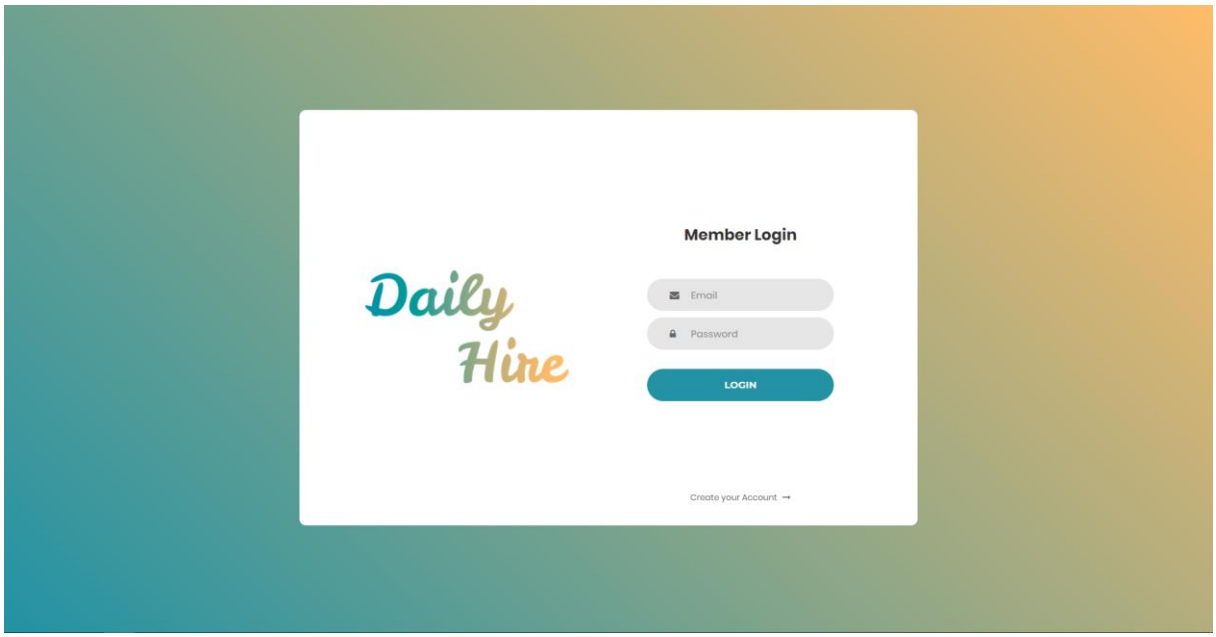

Figura 13 - Funcionalidade: Login - Página *Web*

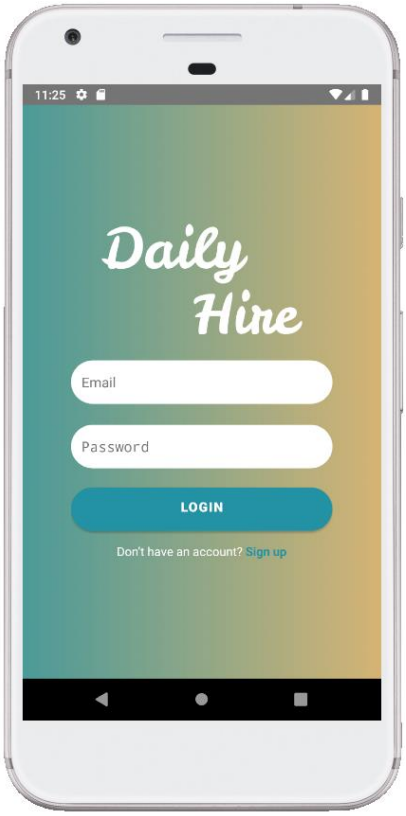

Figura 14 - Funcionalidade: Login - *App*

5.1.3. Anunciar Serviço

Esta funcionalidade é responsável por possibilitar ao utilizador, anunciar serviços após o preenchimento dos campos necessários e estes serem validados pelo sistema. Na figura 15, é possível encontrar o diagrama de sequência a explicar o processo.

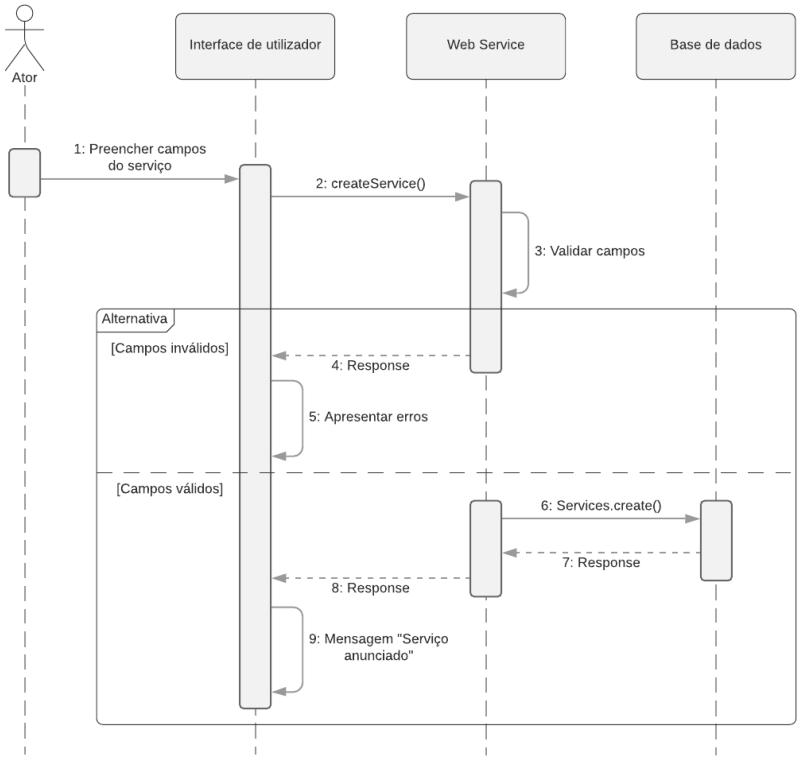

Figura 15 - Diagrama de Sequência: Anunciar Serviço

De seguida, é possível encontrar a interface de anunciar serviço, tanto na página *Web*, Figura 16, como na *App*, Figura 17. Os campos assinalados a vermelho, correspondem aos campos necessários para anunciar um serviço, estando assim demonstrada a validação desses mesmos campos. Uma vez que é necessária a indicação de uma localização para o serviço, foi utilizada a API da *Google* na *App* e da *AlgoliaPlaces* na página *Web*, com o objetivo de sugerir localizações à medida que o utilizador preenchesse este campo.

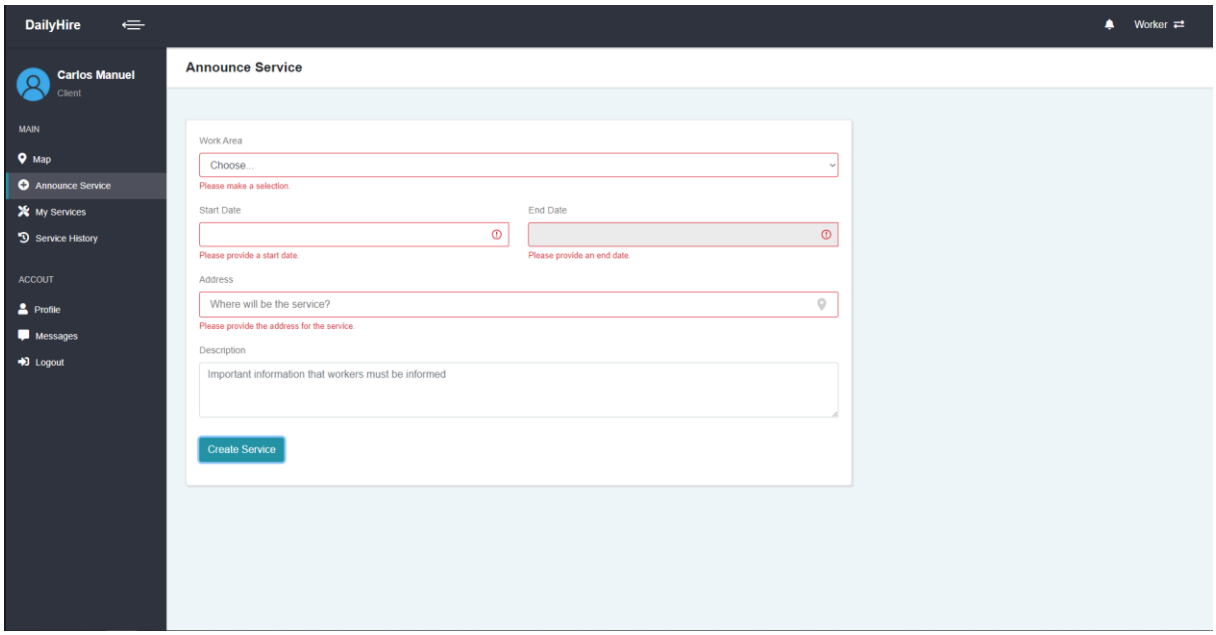

Figura 16 - Funcionalidade: Anunciar Serviço - Página *Web*

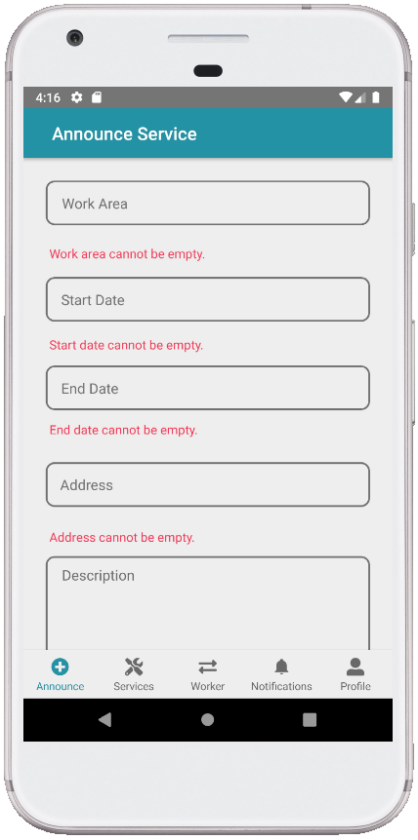

Figura 17 - Funcionalidade: Anunciar Serviço - *App*

5.1.4. Listar Serviços

A listagem de serviços é efetuada de duas formas, variando consoante a área que o utilizador esteja (área de cliente ou área de trabalhador). Um serviço pode ser caracterizado de três formas distintas. Para tal, é utilizado o atributo "*status*", podendo apresentar os seguintes valores "*Pending*", "*Open*", "*Completed*". De seguida, são apresentados os diagramas de sequência, de como o processo de listar os serviços é efetuado.

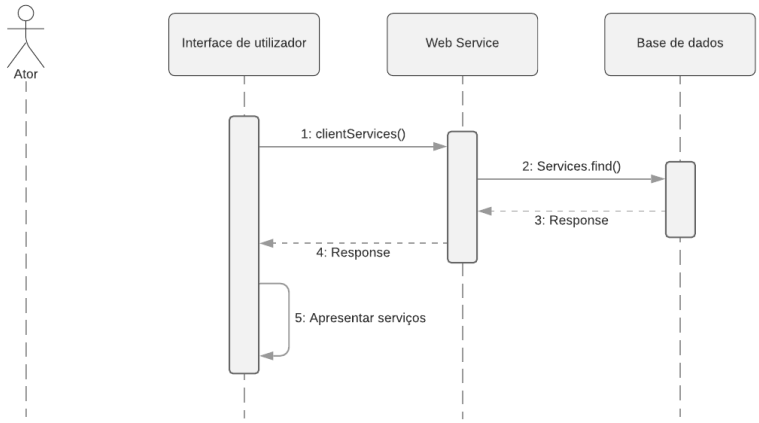

Figura 18 - Diagrama de Sequência: Listar Serviços Cliente

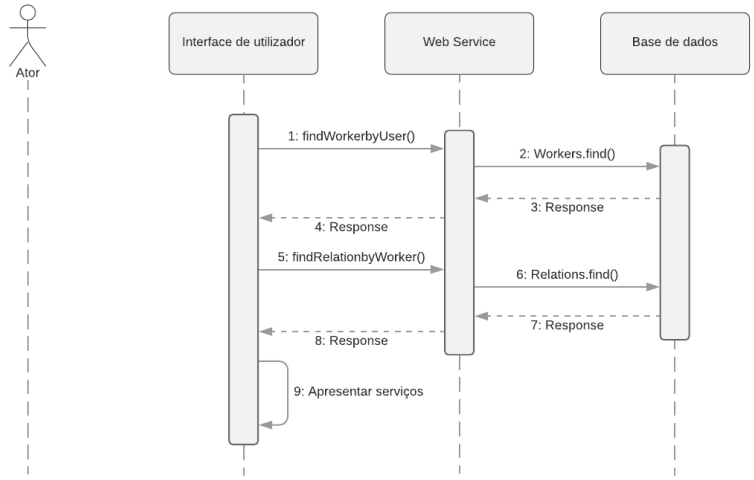

Figura 19 - Diagrama de Sequência: Listar Serviços Trabalhador

Na figura 18, é possível encontrar o diagrama para a listagem de serviços na área de cliente, onde é requisitado ao *Web Service* e posteriormente à base de dados, os serviços consoante o atributo "*status*". Na figura 19, deparamo-nos com o diagrama para a listagem de serviços na área de trabalhador. Neste caso, a listagem é efetuada através das "*Relations*", ou seja, as candidaturas de trabalhadores a serviços. Tendo isto em conta, podemos obter todos os serviços que um trabalhador se candidatou e, graças à chave estrangeira que esta tabela possui da tabela serviços, é possível diferenciar os serviços pelo seu "*status*". Tanto para a área de

cliente como para a área de trabalhador, a listagem de serviços é apresentada do mesmo modo. De seguida, nas Figuras 20 e 21, podemos encontrar a listagem de serviços na *Web* e na *App*, respetivamente.

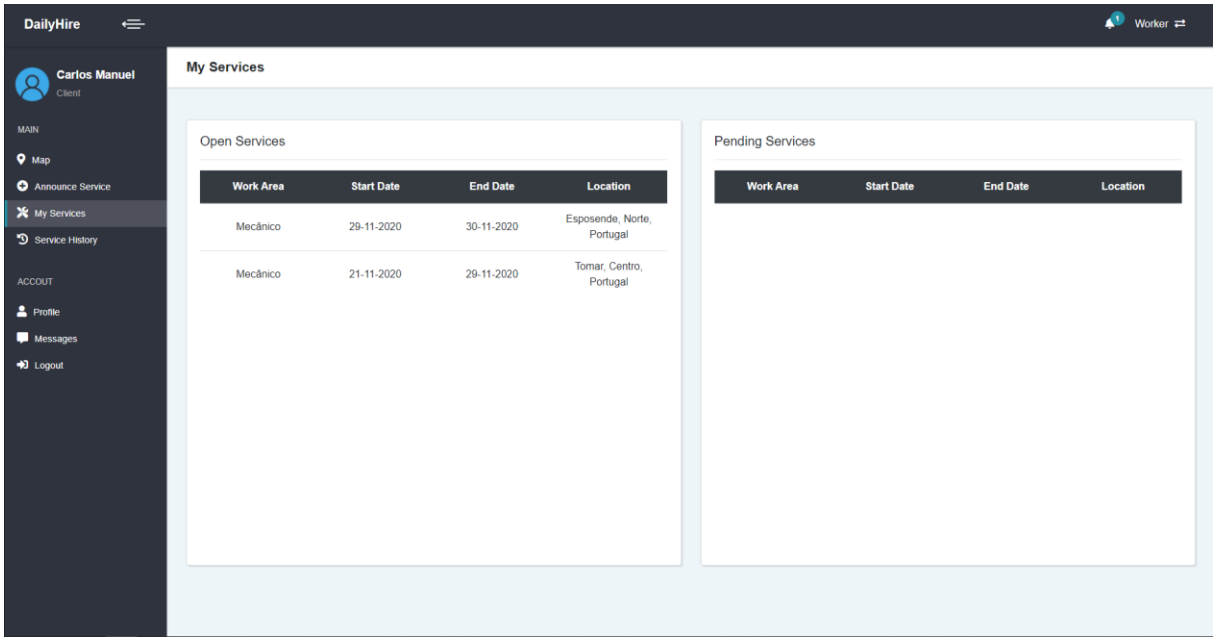

Figura 20 - Funcionalidade: Listar Serviços - Página *Web*

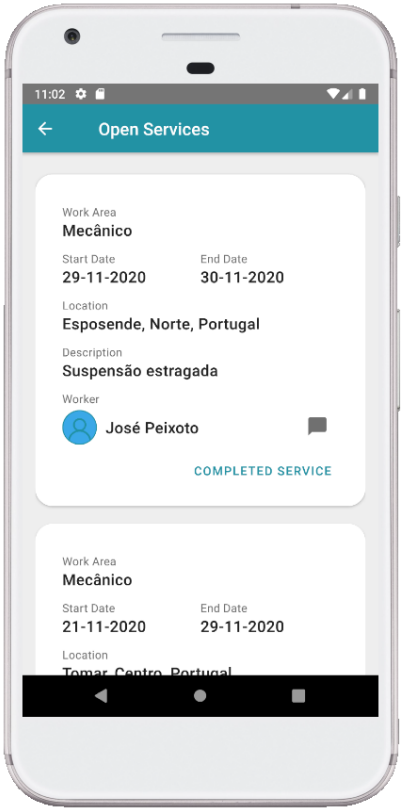

Figura 21 - Funcionalidade: Listar Serviços - *App*

5.1.5. Cancelar Serviço

Um serviço enquanto estiver no seu estado "*Pending*", tem a possibilidade de ser cancelado, caso o utilizador que anunciou o serviço assim pretenda. Na Figura 22, é possível encontrar o diagrama de sequência relativo a esta funcionalidade.

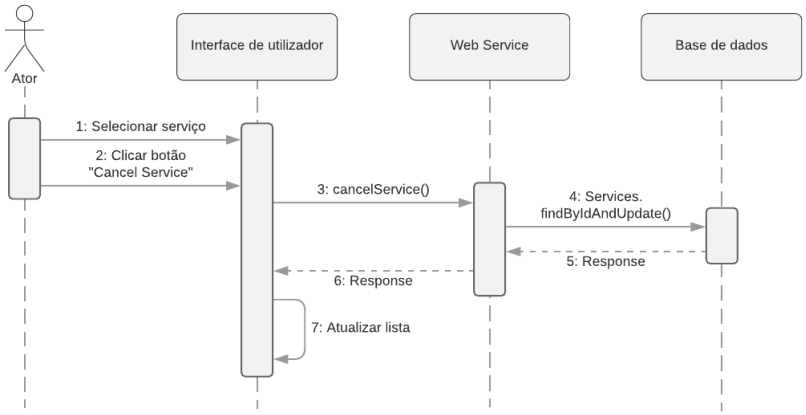

Figura 22 - Diagrama de Sequência: Cancelar Serviço

Esta funcionalidade dá a possibilidade ao cliente de cancelar o serviço, por qualquer motivo que o impossibilite de acontecer. Uma vez cancelado o serviço, este passa a estado "Canceled", com o objetivo de armazenar todos os serviços anunciados e não simplesmente apagar estes serviços da base de dados. Caso um trabalhador se tenha candidata a este serviço, o mesmo será notificado desta alteração. Na figura 23 e 24, é possível encontrar as interfaces da página *Web* e *App*, responsáveis por cancelar o serviço.

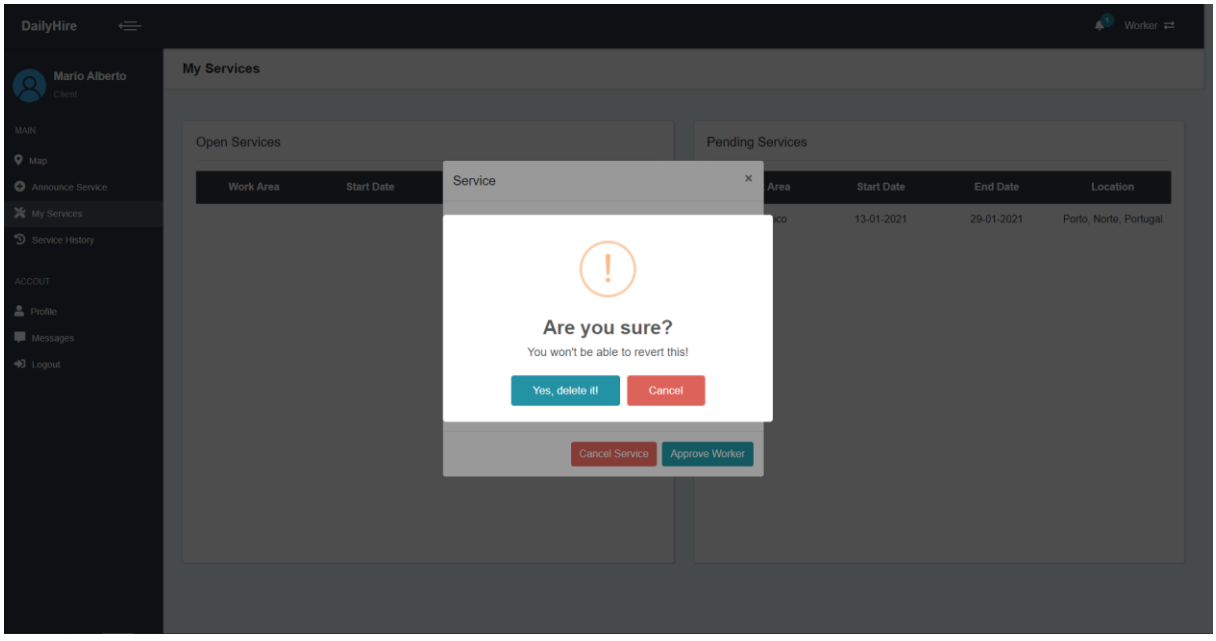

Figura 23 - Funcionalidade: Cancelar Serviço - Página *Web*

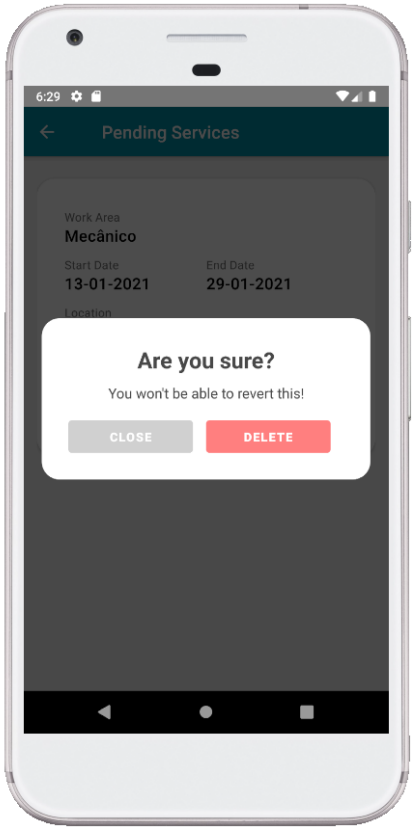

Figura 24 - Funcionalidades: Cancelar Serviço - *App*

5.1.6. Aprovar Trabalhador

Uma vez anunciado um serviço, este encontra-se no seu estado "*Pending*", estando disponível a receber candidaturas de trabalhadores para a execução do serviço. Assim que um cliente se sentir confortável em aprovar determinado trabalhador, o mesmo poderá o fazer através desta mesma funcionalidade. Na figura 25, é possível encontrar o diagrama de sequência que facilita o entendimento desta funcionalidade.

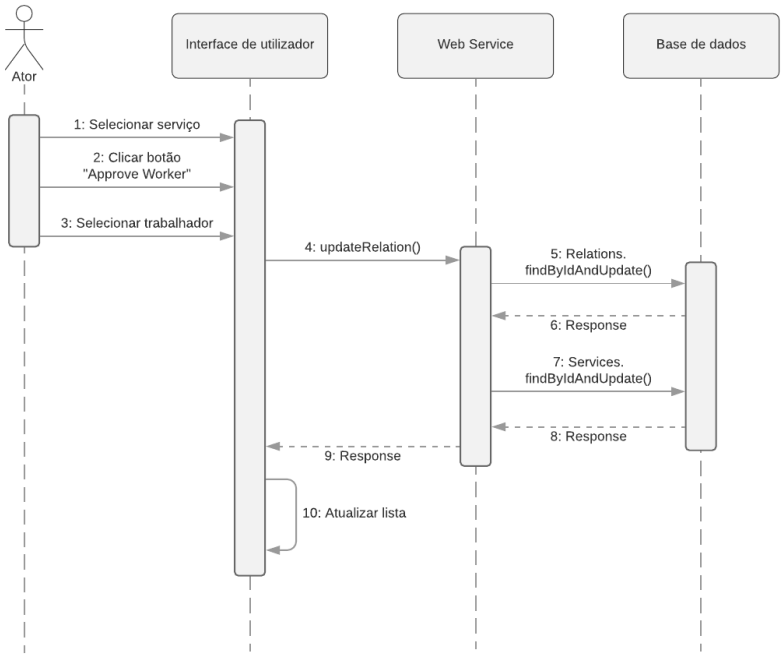

Figura 25 - Diagrama de Sequência: Aprovar Trabalhador

Assim que um utilizador selecionar um serviço, é possível encontrar todos os trabalhadores que se candidataram ao mesmo, tendo a possibilidade de aprovar o trabalhador que pretender. Uma vez aprovado o trabalhador, o estado do serviço passa a "*Open*" e o estado da relação passa a "*True*". Esta relação corresponde a uma candidatura de um trabalhador a um serviço, sendo que o "*status*" da mesma, indica se esse mesmo trabalhador foi ou não aprovado para o respetivo serviço. Na figura 26 e 27, é possível encontrar a interface da página *Web* e *App*, respetivamente.

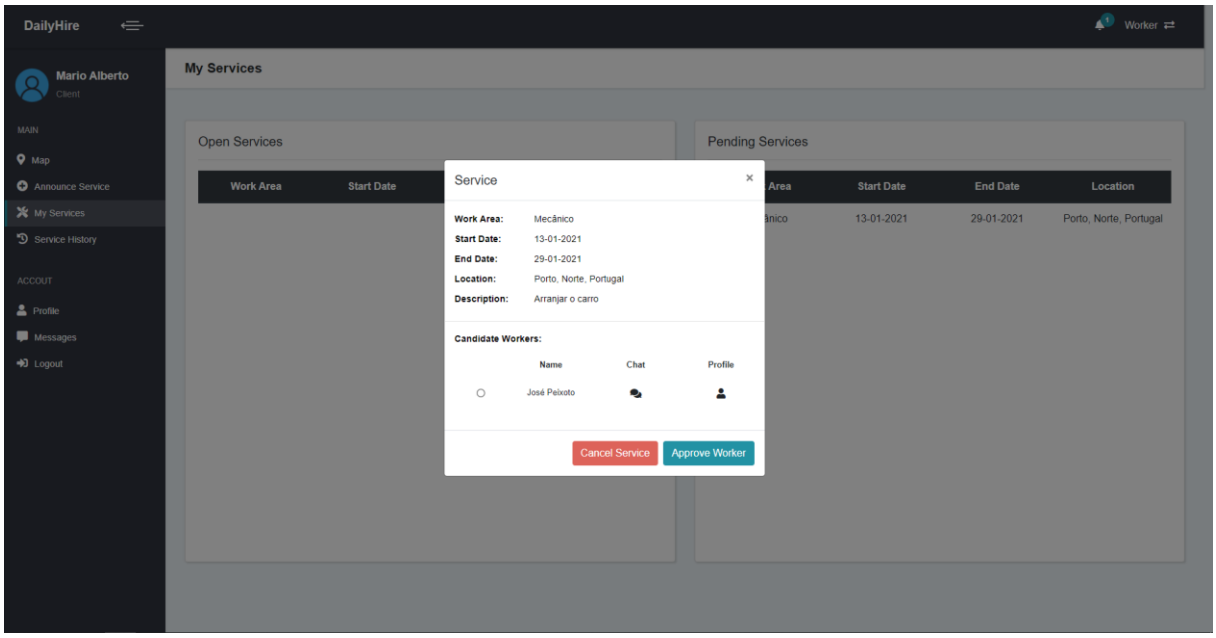

Figura 26 - Funcionalidade: Aprovar Trabalhador – Página *Web*

Figura 27 - Funcionalidade: Aprovar Trabalhador – *App*

5.1.7. Terminar Serviço

Esta funcionalidade permite ao cliente terminar um serviço caso o mesmo já tenha sido realizado. Uma vez terminado o serviço, o seu "*status*" será alterado para "*Completed*", podendo ser encontrado na tabela do histórico de serviços. Na Figura 28, é possível visualizar o diagrama de sequência que explica todo este processo.

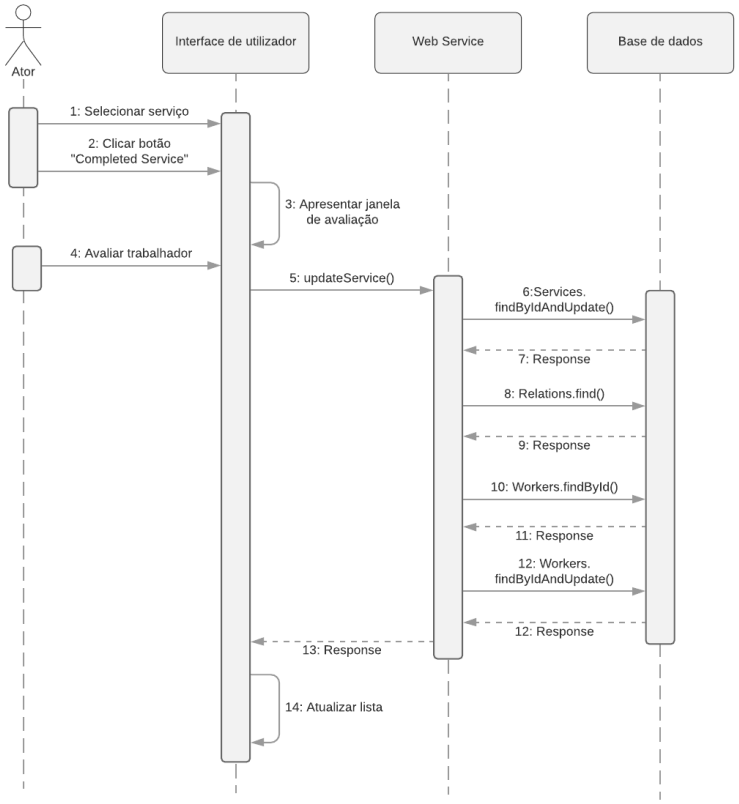

Figura 28 - Diagrama de Sequência: Terminar Serviço

No momento que o utilizador indica que pretende terminar o serviço, será questionado se pretende avaliar o trabalhador responsável. Esta avaliação é efetuada numa escala de 1 a 5, podendo o cliente optar pela opção de não avaliar o trabalhador, se assim pretender. Esta avaliação contribuirá para a avaliação total desse, que corresponde à divisão do valor total de avaliações pelo número de avaliações efetuadas. Esta informação é possível encontrar no perfil desse trabalhador. Este parâmetro poderá influenciar um cliente na decisão entre diversos trabalhadores que se candidataram à execução de um serviço. Na figura 29 e 30, podemos visualizar as interfaces *Web* e *App*, responsáveis por executar esta funcionalidade.

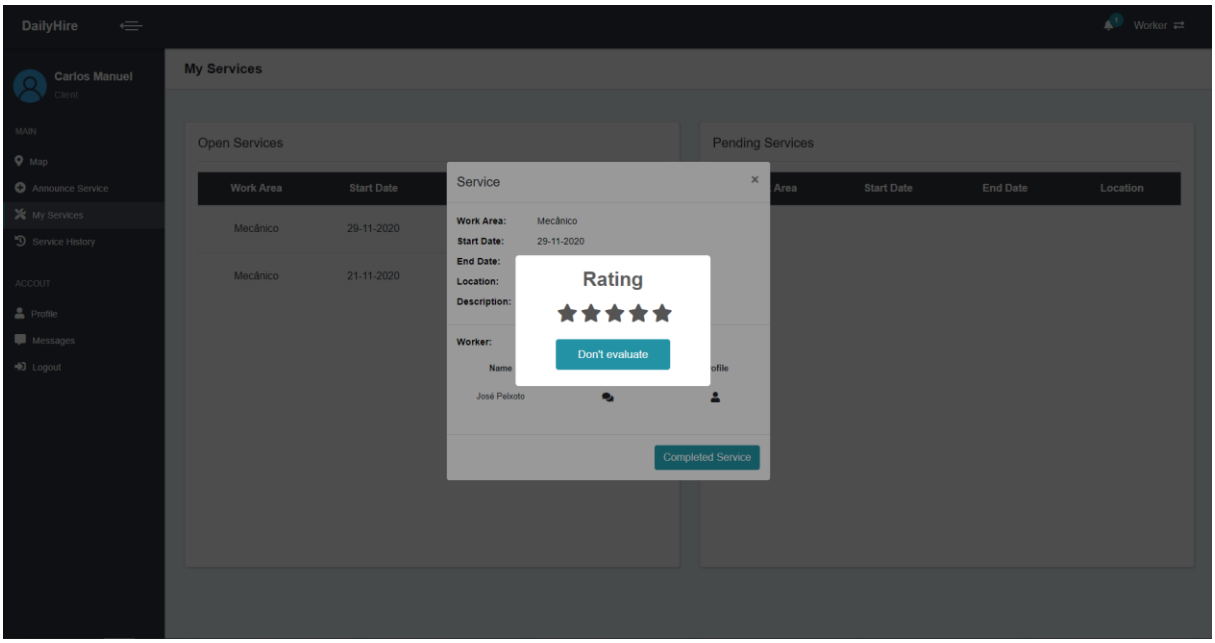

Figura 29 - Funcionalidade: Terminar Serviço - Página *Web*

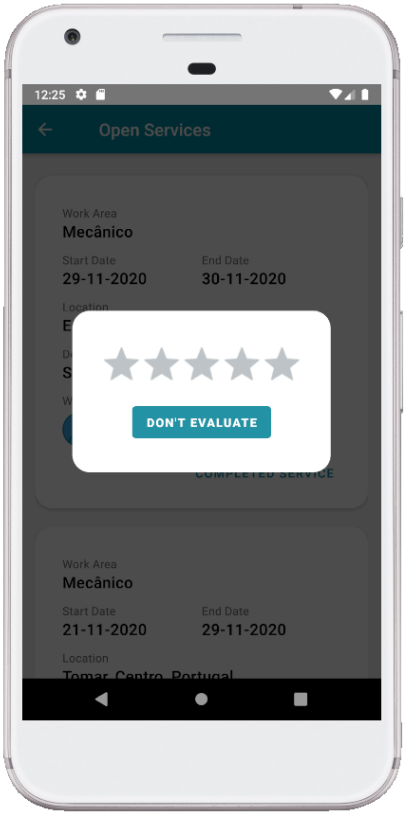

Figura 30 - Funcionalidade: Terminar Serviço – *App*

5.1.8. Visualizar Perfil

Esta funcionalidade encontra-se divida em três áreas distintas. A possibilidade de visualizar o perfil do cliente, o perfil do trabalhador nos serviços e o perfil do trabalhador no mapa. De seguida, encontram-se os diagramas de sequência responsáveis por explicar o todo processo para cada área mencionada anteriormente.

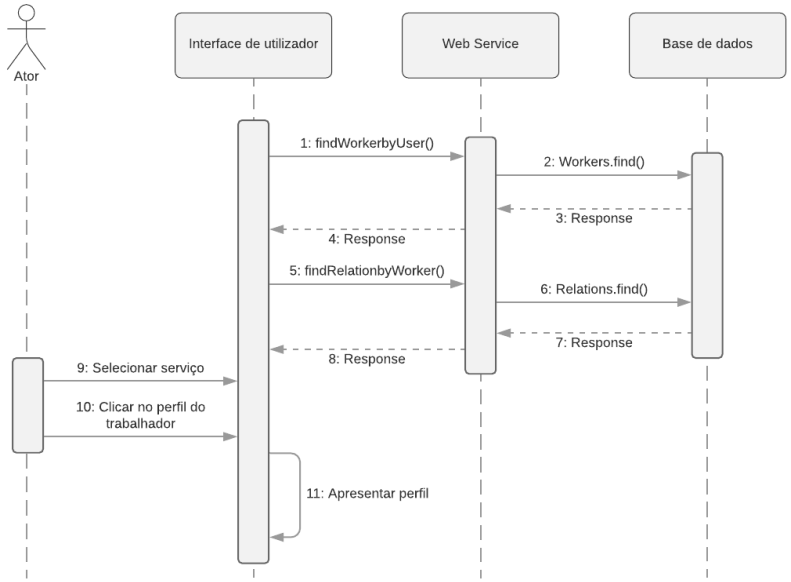

Figura 31 - Diagrama de Sequência: Visualizar Perfil do Cliente

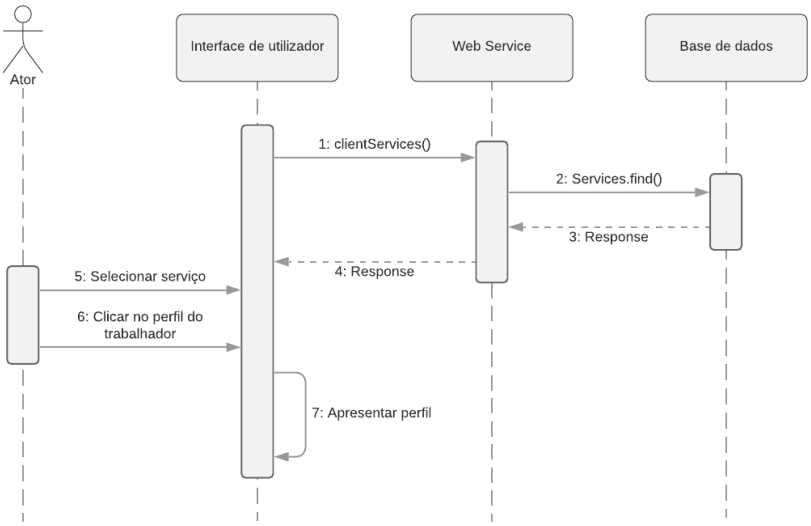

Figura 32 - Diagrama de Sequência: Visualizar Perfil do Trabalhador nos Serviços

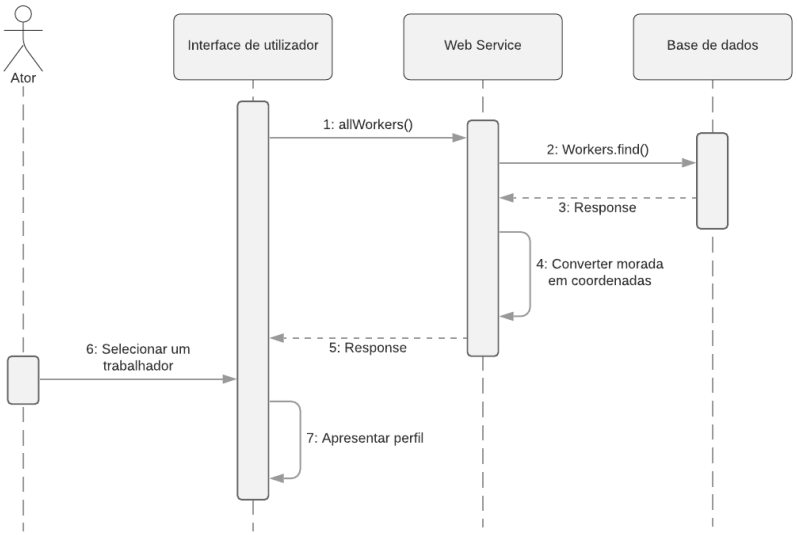

Figura 33 - Diagrama de Sequência: Visualizar Perfil do Trabalhador no Mapa

Tendo em conta os diagramas apresentados anteriormente, demonstram uma funcionalidade semelhante implementada de três maneiras distintas. Na figura 31, é possível um trabalhador visualizar o perfil do cliente que anunciou o serviço, uma vez que a tabela serviços possui a chave estrangeira da tabela utilizadores, o que permite identificar a que utilizador, o serviço está associado. Na figura 32, é possível o cliente visualizar o perfil do trabalhador candidato ao serviço, uma vez que, no ato da candidatura do trabalhador é criada uma "*Relation*" entre o trabalhador e o serviço. Através desta relação, é possível alcançar os dados do trabalhador em questão. Por último, na Figura 33, é possível visualizar o perfil do trabalhador no mapa, pois no carregamento da página mapa, são requisitados todos os trabalhadores registados na plataforma. Cada trabalhador possui a chave estrangeira da sua tabela utilizador associada, sendo desta maneira, possível aceder aos dados de cada trabalhador. De seguida, são apresentados dois exemplos dos três mencionados, nas interfaces de utilizador, página *Web* e *App*, Figura 34 e 35, respetivamente.

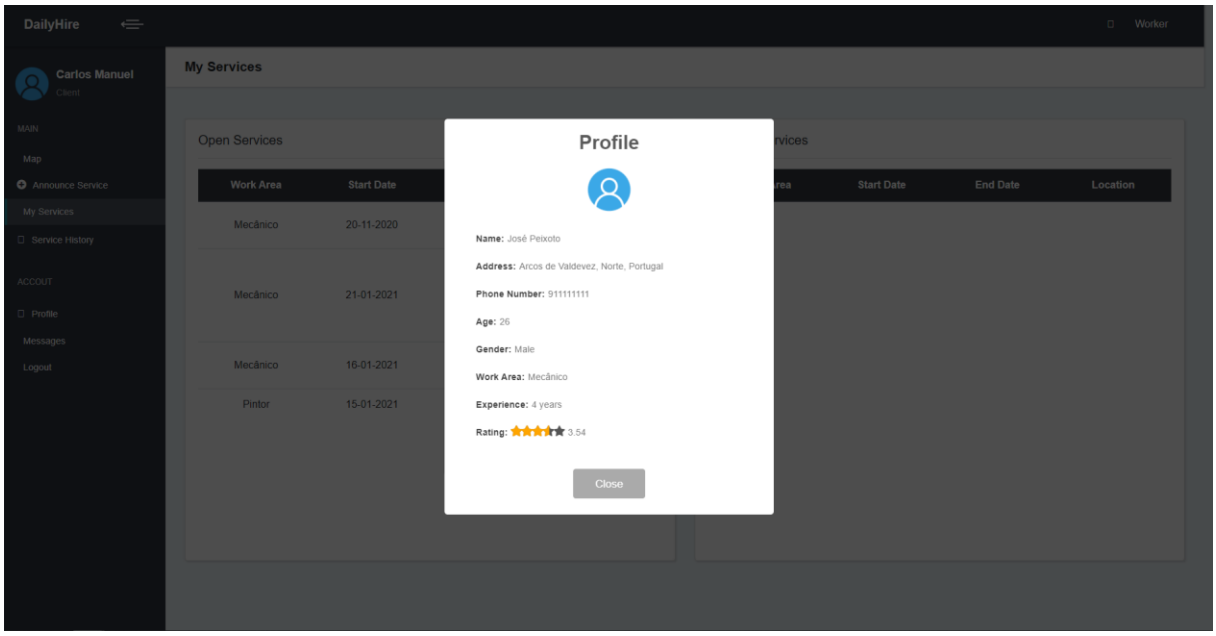

Figura 34 - Funcionalidade: Visualizar Perfil - Página *Web*

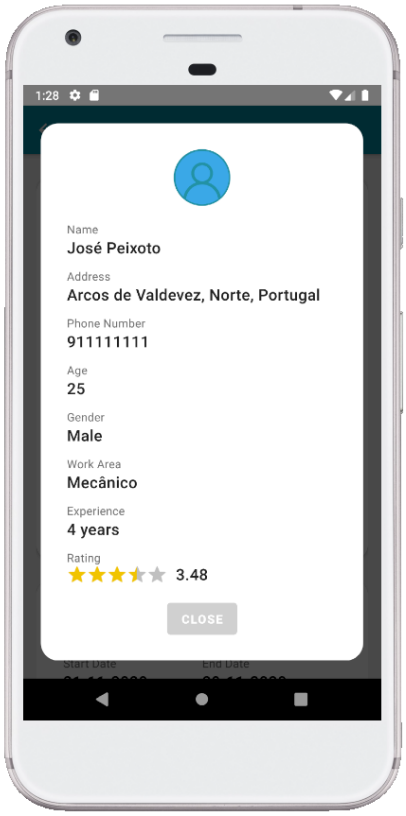

Figura 35 - Funcionalidade: Visualizar Perfil - *App*

5.1.9. Editar Perfil

Esta funcionalidade tem como objetivo possibilitar aos utilizadores editar o seu perfil. Nesta página também é possível ao utilizador, verificar as suas avaliações, tanto de cliente como de trabalhador caso esteja registado como tal. É também nesta página que o utilizador pode alterar a sua foto de perfil, uma vez que todos os utilizadores possuem uma fotografia base, quando se registam na plataforma. De seguida, é possível encontrar o diagrama de sequência a explicar todo este processo.

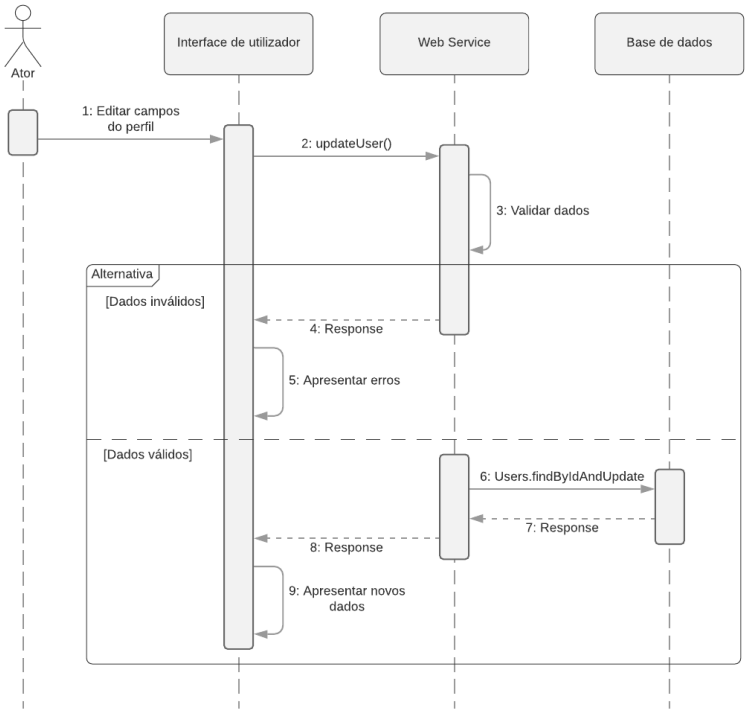

Figura 36 - Diagrama de Sequência: Editar Perfil

Uma vez requisitado o pedido para alteração dos dados do perfil, os mesmos são validados com as mesmas regras utilizadas no ato de criação de conta. Nas Figuras 37 e 38, é possível visualizar as interfaces para editar o perfil na página *Web* e *App*, respetivamente.

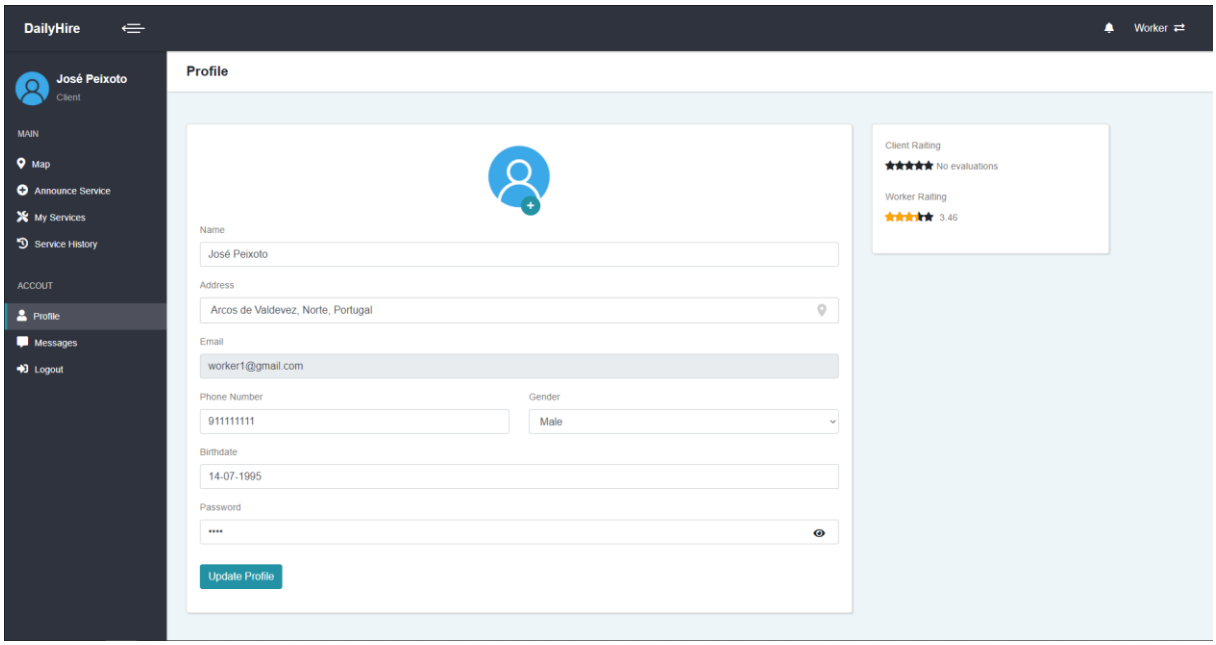

Figura 37 - Funcionalidades: Editar Perfil - Página *Web*

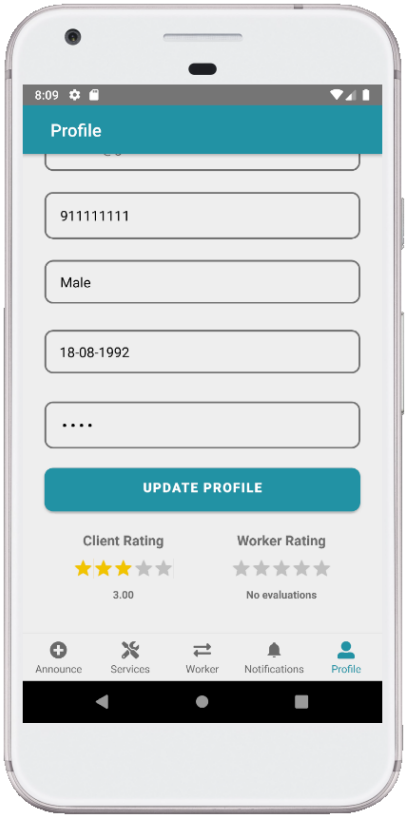

Figura 38 - Funcionalidade: Editar Perfil - *App*

5.1.10. Enviar Mensagem

Esta funcionalidade divide-se em três tipos. A capacidade de enviar uma mensagem quando o utilizador se encontra na conversa com outro utilizador, quando o utilizador se encontra nas diversas listas de serviços e por último, quando o utilizador se encontra no mapa. No primeiro tipo, é possível trocar mensagens com outro utilizador através de um *chat*, em tempo real. No segundo, é possível um utilizador, enviar uma mensagem que poderá ser encontrada posteriormente no *chat*, a outro utilizador. Por último, no terceiro é possível um utilizador enviar uma mensagem aos diversos trabalhadores apresentados no mapa. De seguida, é possível encontrar um diagrama de sequência exemplificando o funcionamento de um dos processos, uma vez que os restantes se assemelham a este.

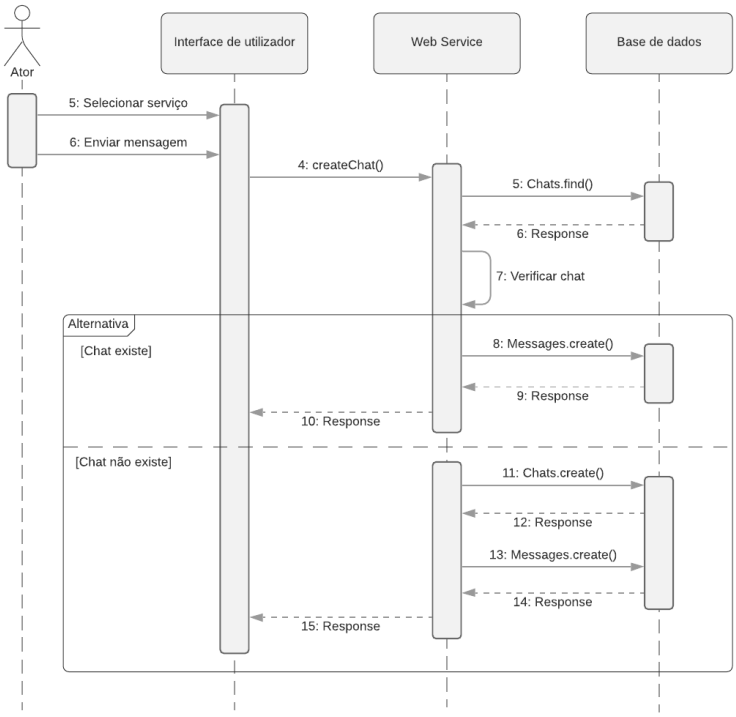

Figura 39 - Diagrama de Sequência: Enviar Mensagem

Como é possível interpretar no diagrama, inicialmente é verificado se uma conversa entre os dois utilizadores em questão, já existe. Caso a resposta a esta questão seja sim, a mensagem submetida será guarda tendo em conta o "*Chat*" já existente, caso contrário, será criado um "*Chat*" e associada a mensagem a esta nova conversa. De seguida é possível, visualizar nas Figuras 40, 41 e 42 as três formas de enviar mensagem na *Web* e na Figura 43 a forma de enviar mensagens na *App*.

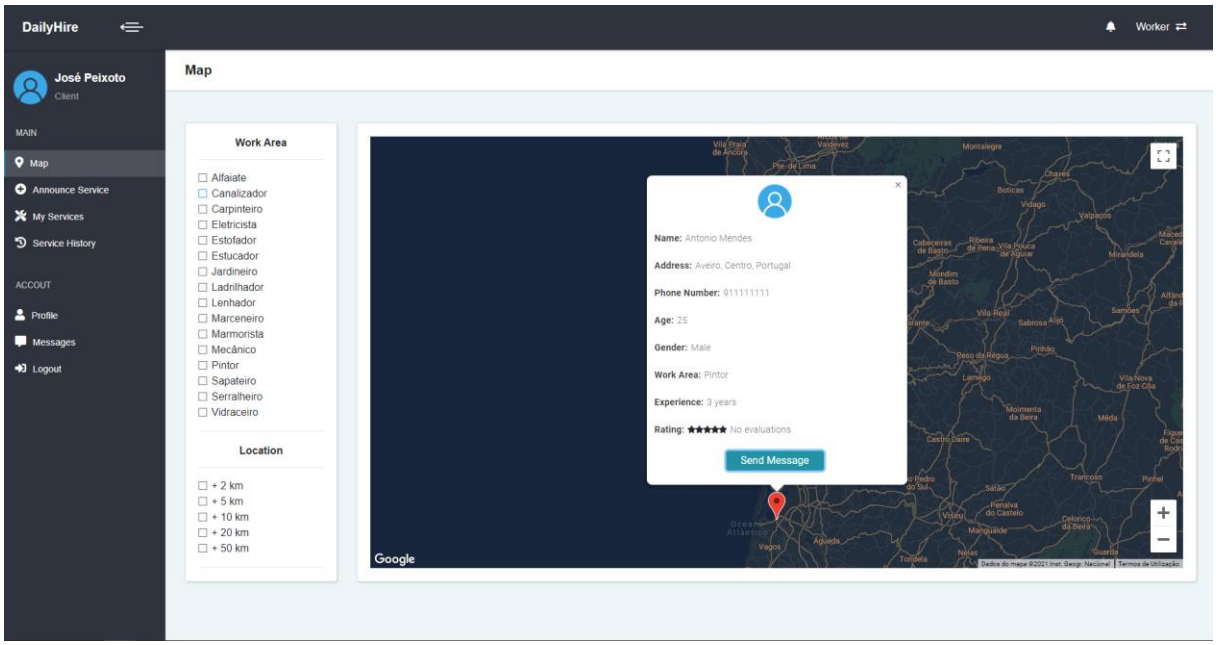

Figura 40 - Funcionalidade: Enviar Mensagem no Mapa - Página *Web*

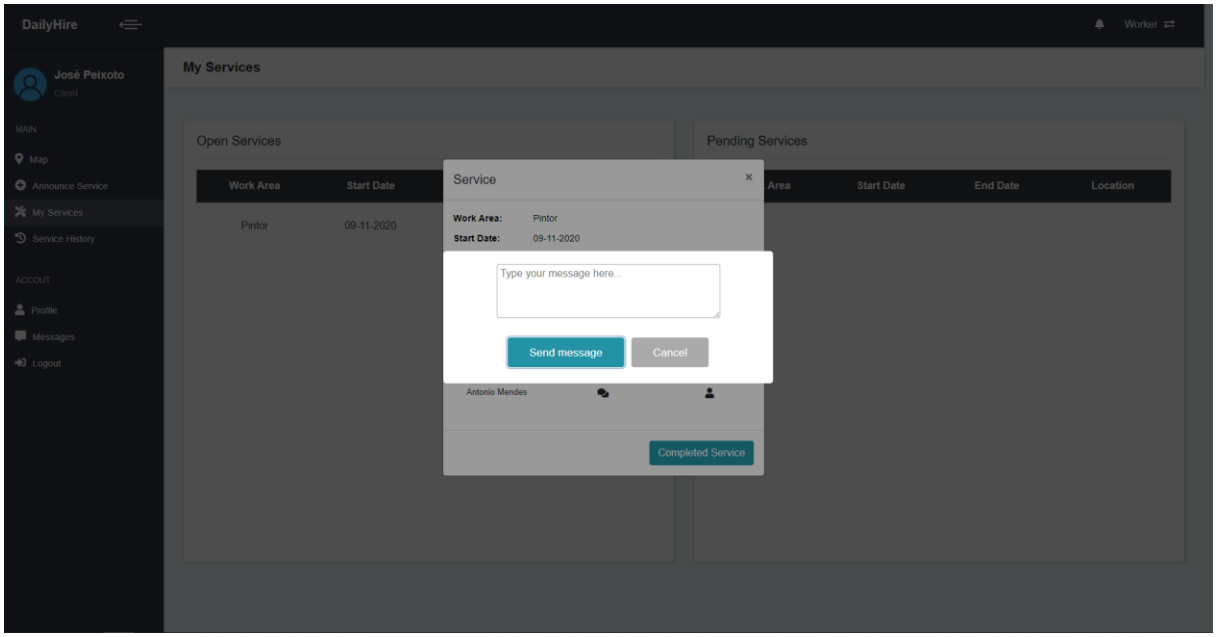

Figura 41 - Funcionalidade: Enviar Mensagem nos Serviços - Página *Web*

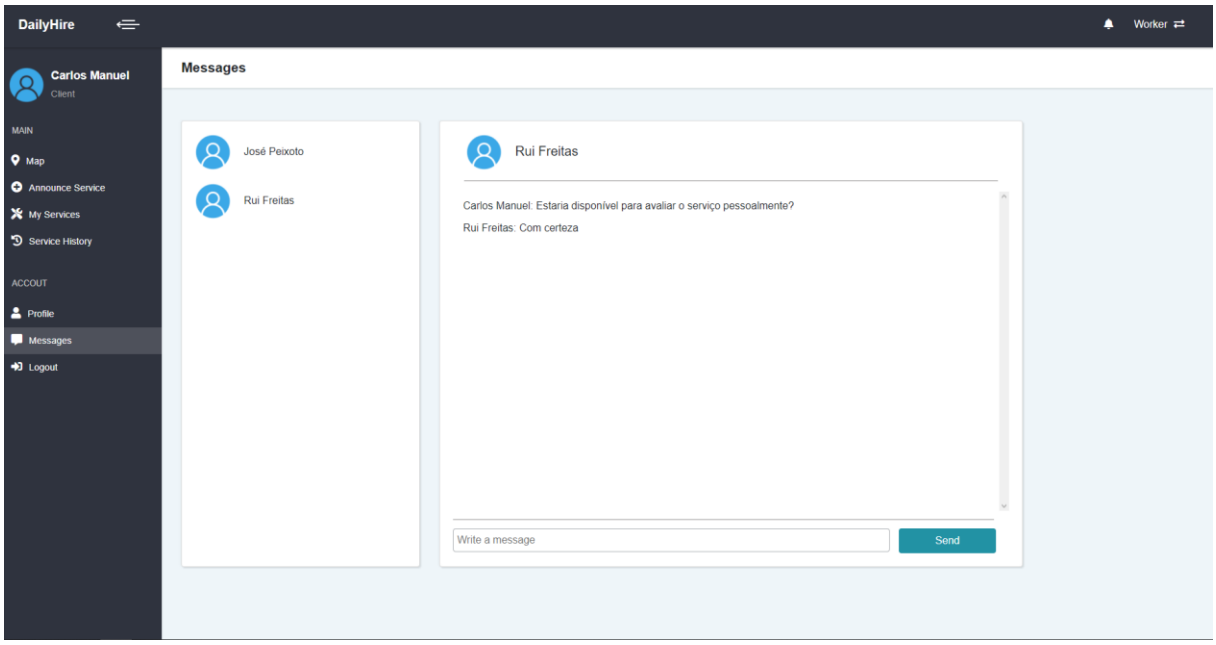

Figura 42 - Funcionalidade: Enviar Mensagem no Chat - Página *Web*

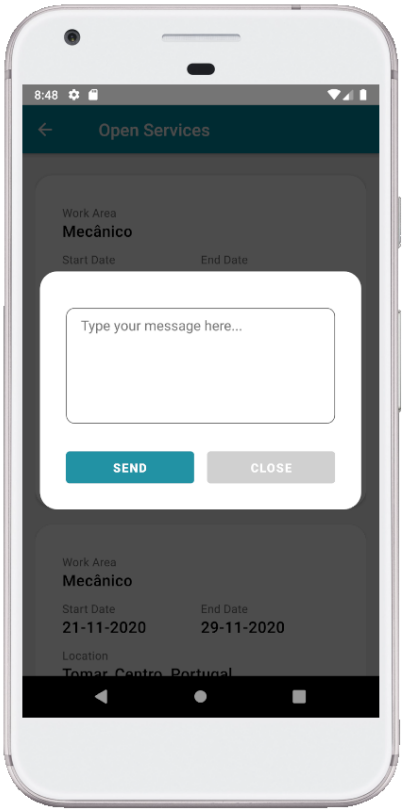

Figura 43 - Funcionalidade: Enviar Mensagem nos Serviços - *App*

5.1.11. Visualizar Mensagens

Na secção 5.10 foi referida a funcionalidade que possibilitava o envio de mensagens, sendo necessário uma funcionalidade que permita a visualização de todas as mensagens trocadas com outros utilizadores. Esta funcionalidade é responsável por permitir isso. Na Figura 44, podemos encontrar o diagrama de sequência que explica todo o funcionamento da visualização de mensagens.

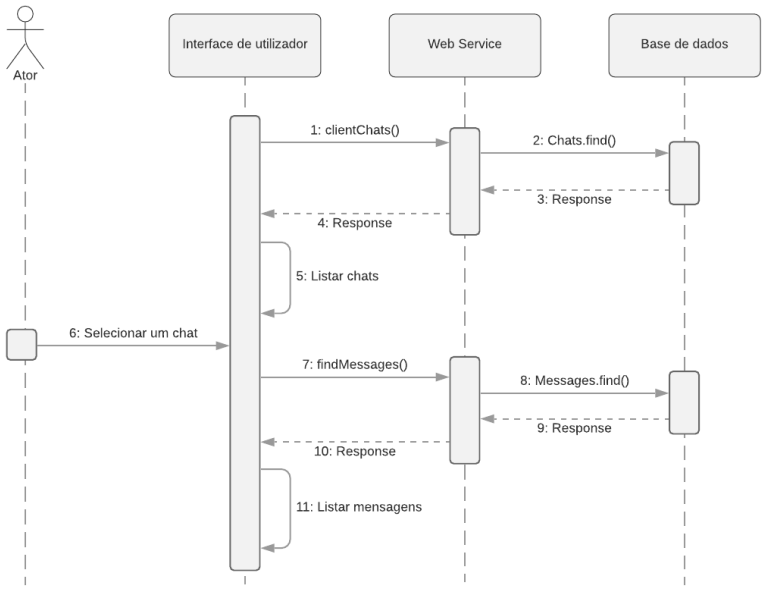

Figura 44 - Diagrama de Sequência: Visualizar Mensagens

Inicialmente, são apresentados todos os "*Chats*" do utilizador, sendo que após a seleção de determinada conversa, serão listadas todas as mensagens trocadas tendo em conta esse *chat*. Isto é possível graças à chave estrangeira presente na tabela mensagens, onde cada mensagem enviada por um utilizador, possui o identificador do *chat* que pertence. Na figura 45, é possível encontrar a página *Web* responsável por apresentar todas as mensagens trocados pelo utilizador.

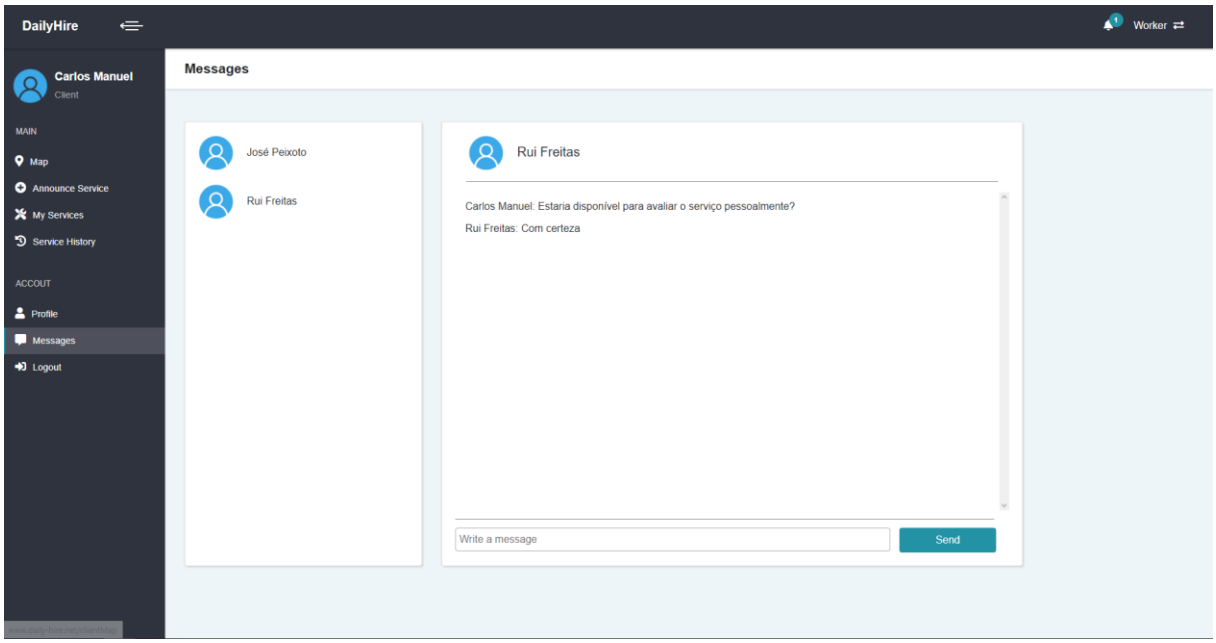

Figura 45 - Funcionalidades: Visualizar Mensagens - Página *Web*

5.1.12. Enviar Notificações

Esta funcionalidade divide-se em duas áreas. Estas áreas correspondem a alterações num serviço que um utilizador esteja associado e novas mensagens que um utilizador receba. Face à primeira área, um utilizador é notificado enquanto trabalhador, quando um serviço que se candidatou é cancelado, a sua candidatura à realização de um serviço é aprovada ou quando o serviço é terminado pelo cliente, o que permite ao trabalhador avaliar o respetivo cliente. Enquanto cliente, este é notificado sempre que seja criada uma candidatura para um serviço que tenha anunciado. Na figura 46, é possível visualizar o diagrama de sequência responsável por enviar notificações.

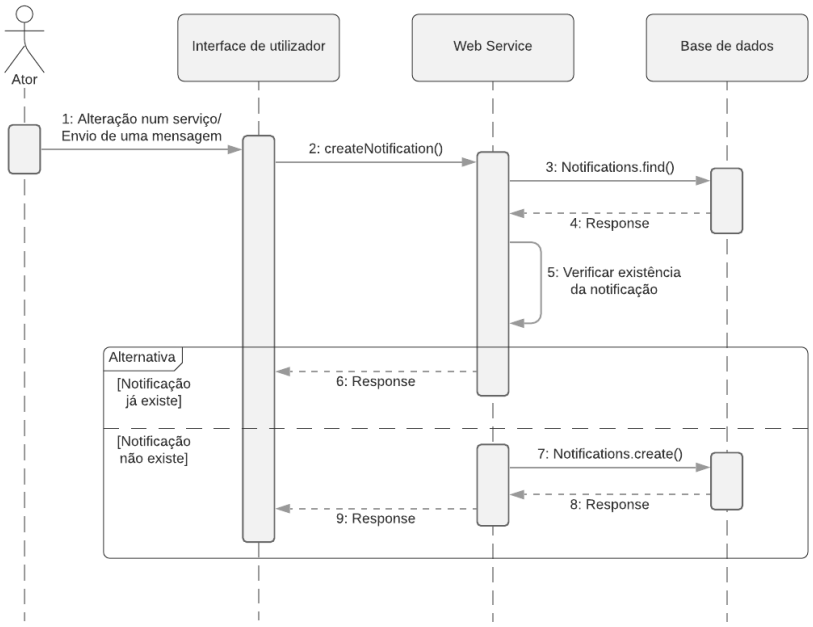

Figura 46 - Diagrama de Sequência: Enviar Notificações

5.1.13. Visualizar Notificações

Como referido na secção 5.11, as notificações são enviadas para o utilizador quando um serviço que esteja associado sofre qualquer alteração ou quando recebem uma nova mensagem de outro utilizador. Esta funcionalidade é responsável por possibilitar a visualização das notificações que o utilizador possui no momento. Na figura 47, encontra-se o diagrama de sequência desta mesma funcionalidade.

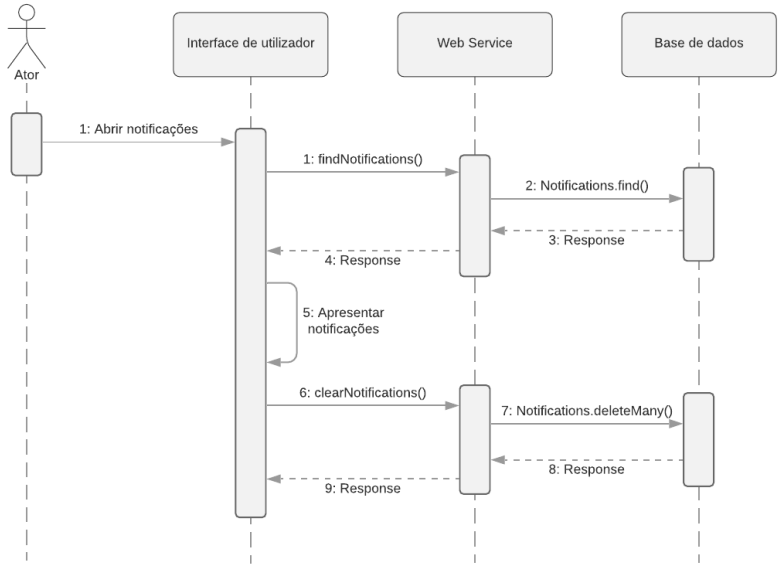

Figura 47 - Diagrama de Sequência: Visualizar Notificações

Sempre que uma página é carregada, é requisitado ao *Web Service* por eventuais notificações. Caso estas existam, é apresentado o número notificações existentes junto ao ícone de notificações, tanto na página *Web* como na *App*. Assim que, o utilizador visualize as suas notificações, estas são removidas do sistema de forma a não voltarem a ser apresentadas. Nas figuras 48 e 49, podemos encontrar a área responsável por apresentar as notificações ao utilizador, na página *Web* e *App*, respetivamente.

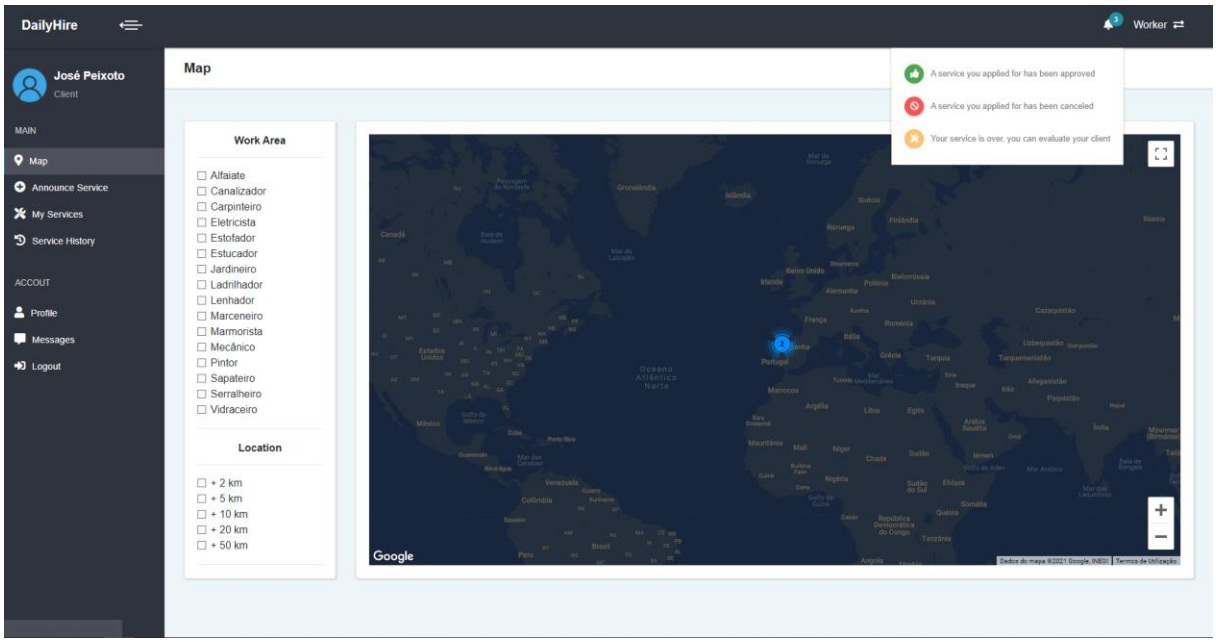

Figura 48 - Funcionalidade: Visualizar Notificações – Página *Web*

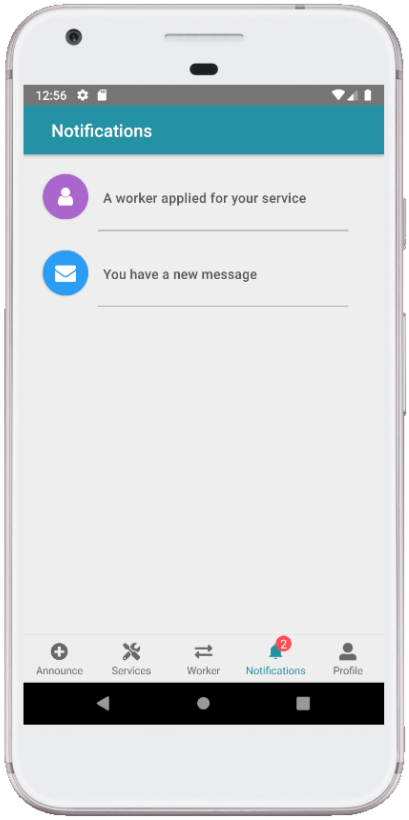

Figura 49 - Funcionalidade: Visualizar Notificações - *App*

5.1.14. Apresentar Mapa

Nesta funcionalidade é apresentado o mapa, utilizando a API da *Google*. Neste mapa é possível encontrar todos os trabalhadores registados na plataforma, através de um marcador assinalado no mapa, tendo em conta a morada fornecida pelo trabalhador no ato de registo da sua conta. De seguida, é possível encontrar o diagrama de sequência a explicar o processo.

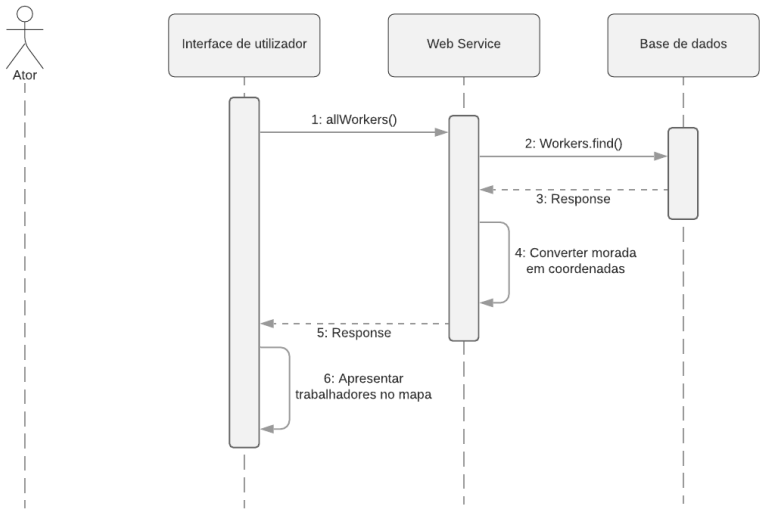

Figura 50 - Diagrama de Sequência: Apresentar Mapa

Na Figura 51, é possível encontrar a interface mapa, estando disponível na página *Web*. Neste mapa, como é possível visualizar, o utilizador tem a opção de selecionar filtros como a profissão pretendida ou a distância em relação à localização do utilizador.

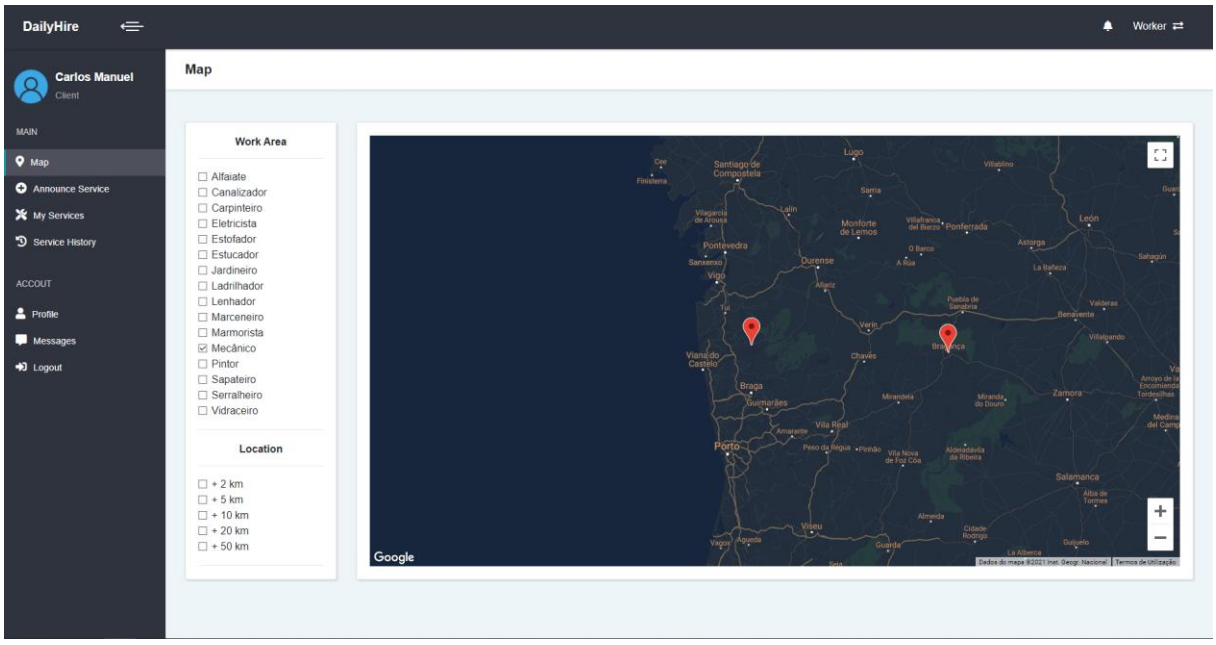

Figura 51 - Funcionalidade: Apresentar Mapa – Página *Web*

5.1.15. Exportar *Linked Data*

Esta funcionalidade permite ao utilizador obter os dados definidos como públicos, presentes na base de dados, sob a forma de *Linked Data*. Para tal, é fornecido ao utilizador um ficheiro RDF-XML, tendo como base para a sua elaboração a matriz de restrições definida, apresentada nos apêndices (Matriz de Restrições). De seguida, é possível encontrar na Figura 52, o diagrama de sequência a explicar o funcionamento te todo o processo.

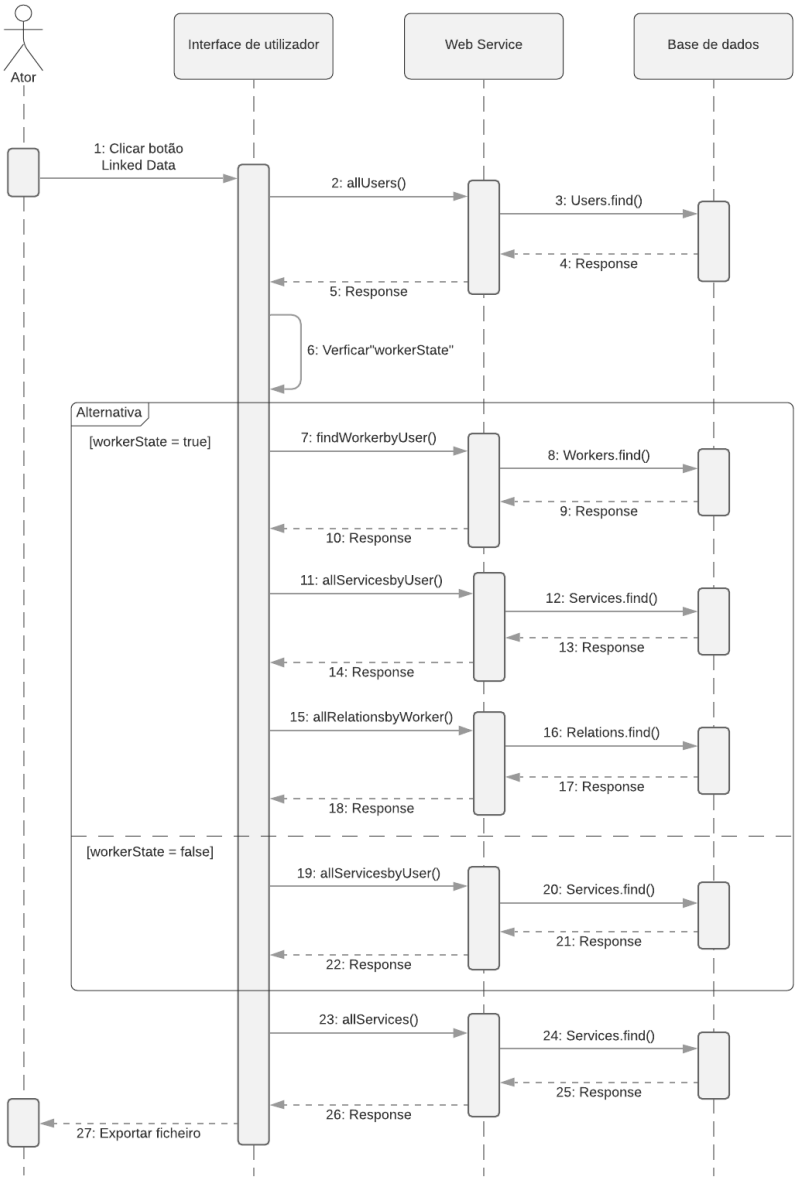

Figura 52 - Diagrama de Sequência: Exportar Ficheiro RDF-XML

A informação presente no ficheiro, é dividida em duas partes. Inicialmente, é apresentada a informação relativa a todos os utilizadores registados no sistema, onde é requisitado à base dados a informação destes mesmos utilizadores. Para além dos seus dados, é também requisitado os serviços anunciados por cada utilizador, a informação de trabalhador caso o utilizador se tenha registado como tal e, por último, todas as candidaturas efetuadas enquanto trabalhador. Na segunda parte, é apresentada a informação relativa a todos os serviços anunciados.

Os dados, são apresentados sob a forma de triplos RDF, como é possível visualizar na Figura 53. Os triplos, como mencionado na secção 2.2, correspondem a uma combinação de recursos, propriedades e literais na forma de sujeito, predicado e objeto.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
xmlns:dct="http://purl.org/dc/terms/"
xmlns:foaf="http://xmlns.com/foaf/0.1/"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:schema="http://schema.org/"
xmlns:vcard="http://www.w3.org/2006/vcard/ns#"
xmlns:frapo="http://purl.org/cerif/frapo/"
xmlns:empower="http://purl.org/empower/vocab/1.0/">

<rdf:Description rdf:about="http://www.daily-hire.net/user/5fa415b7204f642410cbf16c">
<rdf:type rdf:resource="http://purl.org/empower/vocab/1.0/User"/>
<dct:identifier rdf:resource="5fa415b7204f642410cbf16c"/>
<foaf:name rdf:datatype="http://www.w3.org/TR/xmlschema11-2/#string">Carlos Manuel</foaf:name>
<dct:type rdf:datatype="http://www.w3.org/TR/xmlschema11-2/#boolean">0</dct:type>
<empower:ratingCountAsClient rdf:datatype="http://www.w3.org/TR/xmlschema11-2/#integer">1</empower:ratingCountAsClient>
<empower:ratingValueAsClient rdf:datatype="http://www.w3.org/TR/xmlschema11-2/#float">3</empower:ratingValueAsClient>
<schema:offers rdf:resource="http://www.daily-hire.net/service/5fa41aa3204f642410cbf175"/>
<schema:offers rdf:resource="http://www.daily-hire.net/service/5fa41af6204f642410cbf176"/>
<schema:offers rdf:resource="http://www.daily-hire.net/service/5fa41b54204f642410cbf177"/>
<schema:offers rdf:resource="http://www.daily-hire.net/service/5fa481c52b69900a182ef53c"/>
<schema:offers rdf:resource="http://www.daily-hire.net/service/5fa4821c2b69900a182ef53e"/>
<schema:offers rdf:resource="http://www.daily-hire.net/service/5fa96059406e050f685639b9"/>
<schema:offers rdf:resource="http://www.daily-hire.net/service/5faeee431ca8aa302c8884ae"/>
<schema:offers rdf:resource="http://www.daily-hire.net/service/5ffba13a113e200017a8cc4d"/>
<schema:offers rdf:resource="http://www.daily-hire.net/service/5ffba144113e200017a8cc4e"/>
<schema:offers rdf:resource="http://www.daily-hire.net/service/5ffba1d9113e200017a8cc59"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.daily-hire.net/user/5fa41620204f642410cbf16d">
<rdf:type rdf:resource="http://purl.org/empower/vocab/1.0/User"/>
<dct:identifier rdf:resource="5fa41620204f642410cbf16d"/>
<foaf:name rdf:datatype="http://www.w3.org/TR/xmlschema11-2/#string">Sara Costa</foaf:name>
<dct:type rdf:datatype="http://www.w3.org/TR/xmlschema11-2/#boolean">0</dct:type>
</rdf:Description>

<rdf:Description rdf:about="http://www.daily-hire.net/user/5fa41684204f642410cbf16e">
<rdf:type rdf:resource="http://purl.org/empower/vocab/1.0/User"/>
<dct:identifier rdf:resource="5fa41684204f642410cbf16e"/>
<foaf:name rdf:datatype="http://www.w3.org/TR/xmlschema11-2/#string">Mario Alberto</foaf:name>
<dct:type rdf:datatype="http://www.w3.org/TR/xmlschema11-2/#boolean">0</dct:type>
<schema:offers rdf:resource="http://www.daily-hire.net/service/5ffb31466a24570017938e6c"/>
<schema:offers rdf:resource="http://www.daily-hire.net/service/5ffb552771d3b00017b453c8"/>
</rdf:Description>
```

Figura 53 - Segmento de um ficheiro RDF-XML

Trabalhador

Como mencionado anteriormente, este capítulo divide-se em duas secções. Esta secção, corresponde às funcionalidades de trabalhador. Apesar de um utilizador se registar como trabalhador, o mesmo pode executar as funções de cliente apresentadas na secção anterior. Nem todas as funcionalidades de trabalhador estão apresentadas nesta secção, uma vez que, algumas destas, se assemelhavam com funcionalidades de cliente, sendo assim, apresentadas em conjunto.

5.2.1. Alternar Modo Trabalhador

Esta funcionalidade é responsável por alternar entre as duas áreas, cliente e trabalhador. Por sua vez, dá a possibilidade ao utilizador de executar funcionalidades diferentes consoante a área que se encontre. Caso o utilizador ainda não se tenha registado como trabalhador, será requisitado ao mesmo que o faça, preenchendo os campos com a sua informação de trabalhador. Na figura 54, é possível encontrar o diagrama de sequência a explicar como é efetuado todo o processo.

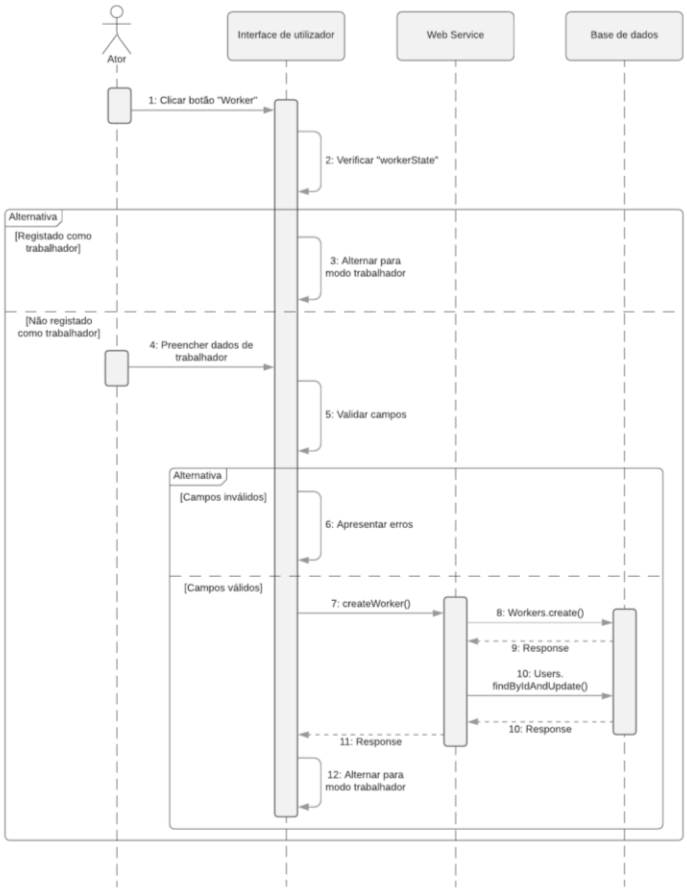

Figura 54 - Diagrama de Sequência: Alternar Modo Trabalhador

Após o utilizador pressionar o botão "*worker*", visível na Figura 55 no canto superior direito para a página *Web* e na parte inferior na Figura 56 para a *App*, é verificado se o mesmo já se encontra registado como trabalhador, através do atributo "*workerState*". Caso já o tenha feito, será reencaminhado para a respetiva área de trabalhador, caso contrário, é requisitado ao utilizador o preenchimento de campos. Como é possível visualizar nas figuras seguintes.

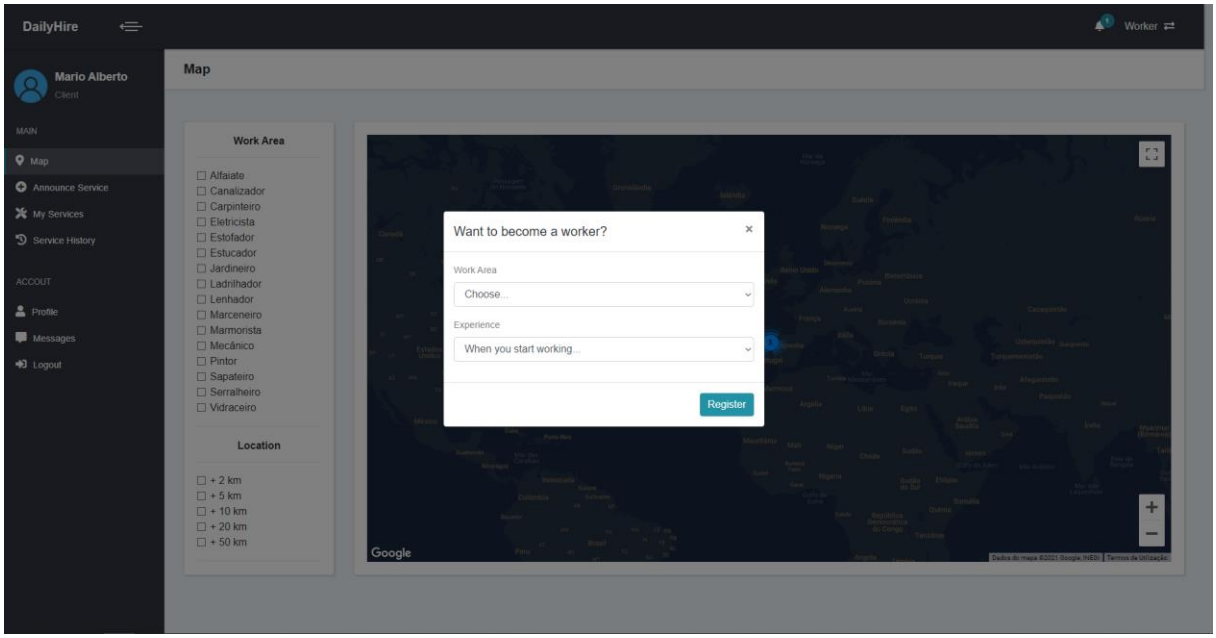

Figura 55 - Funcionalidade: Alternar Modo Trabalhador – Página *Web*

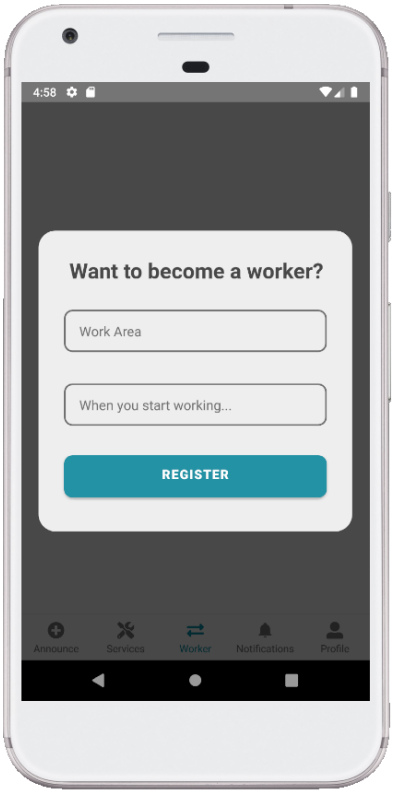

Figura 56 - Funcionalidade: Alternar Modo Trabalhador – *App*

5.2.2. Listar Serviços Disponíveis

Como mencionado anteriormente na secção 5.7, é possível um trabalhador se candidatar a um serviço. Para tal, é necessário apresentar todos os serviços disponíveis para o trabalhador se candidatar. Na Figura 57, podemos encontrar o diagrama de sequência que explica todo este processo.

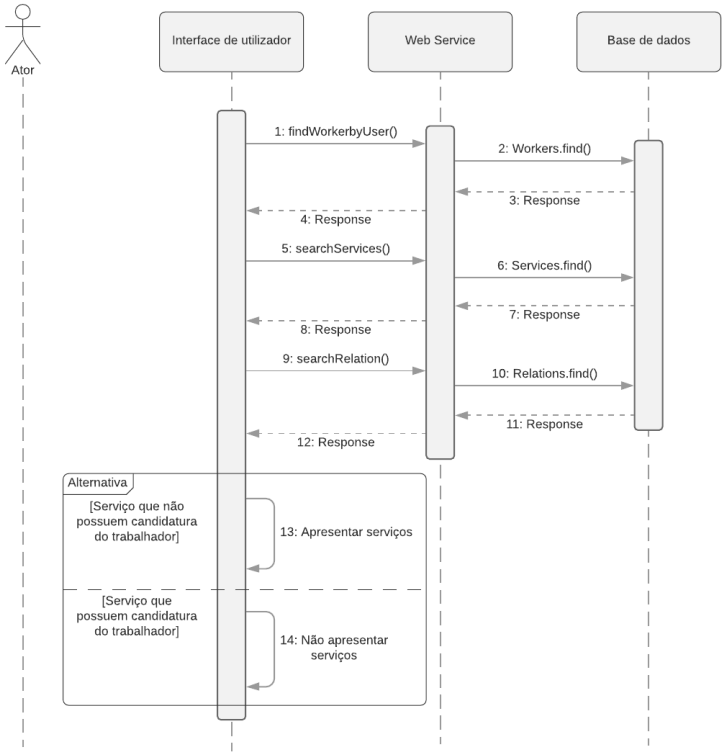

Figura 57 - Diagrama de Sequência: Listar Serviços Disponíveis

De forma, a evitar serem apresentar serviços anunciados pelo próprio utilizador, inicialmente é verificado em todos os serviços pendentes, se o utilizador que anunciou o serviço é o utilizador a executar esta funcionalidade. Uma vez removidos possíveis serviços do utilizador, é necessário verificar se o utilizador já efetuou alguma candidatura a algum dos serviços. Para tal, é pesquisado na tabela "*Relation*" por relações que possuam o identificador do trabalhador e o identificador do serviço em questão. Na eventualidade de já existir uma relação entre esse trabalhador e o serviço, significa que o mesmo já efetuou uma candidatura. Para finalizar, os restantes serviços que não se enquadrem nestes parâmetros, serão apresentados ao trabalhador. Nas figuras 58 e 59, podemos encontrar a interface *Web* e *App*, desta mesma funcionalidade.

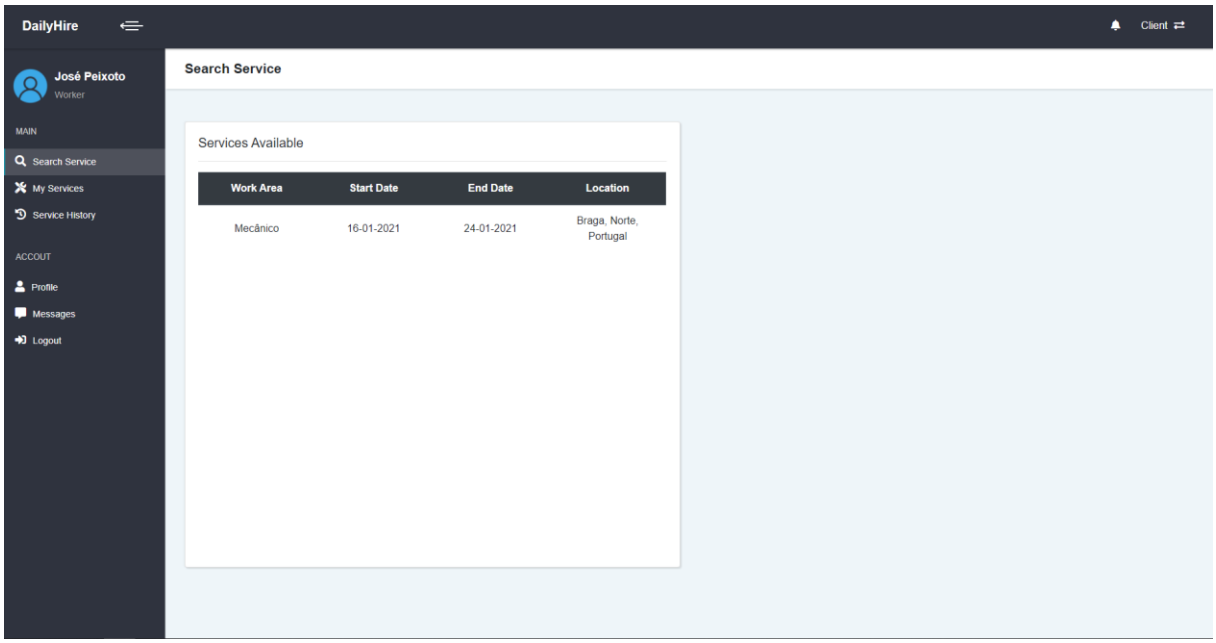

Figura 58 - Funcionalidade: Listar Serviços Disponíveis – Página *Web*

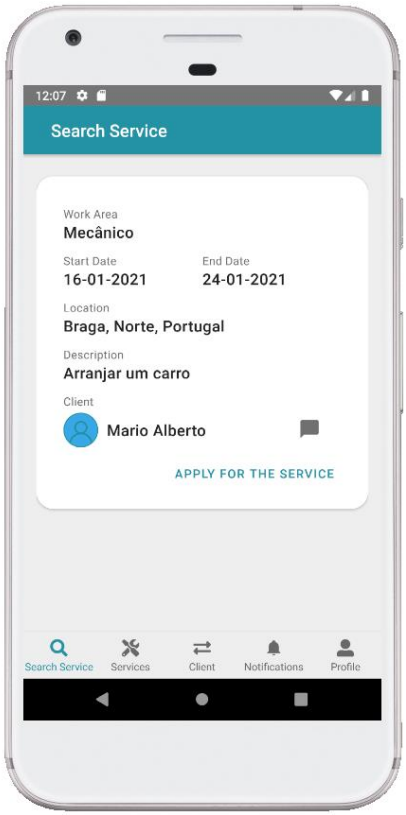

Figura 59 - Funcionalidade: Listar Serviços Disponíveis – *App*

5.2.3. Candidatar-se a um Serviço

Esta funcionalidade permite aos trabalhadores candidatarem-se a um serviço que pretendam. Uma vez selecionado o serviço, e posteriormente a opção "*Apply for the Service*" surgirá, irá ser criada uma nova "*Relation*". Esta relação, como referido anteriormente, corresponde à candidatura de um trabalhador a um serviço, sendo inicialmente definida no atributo "*status*" como falsa. De seguida, é possível encontrar o diagrama de sequência a explicar todo este processo.

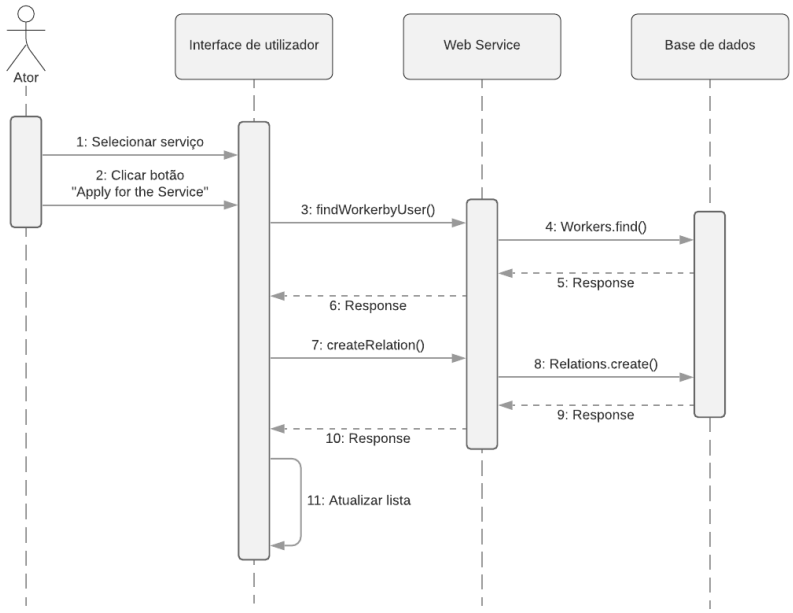

Figura 60 - Diagrama de Sequência: Candidatar-se a um Serviço

Uma vez que um trabalhador se candidata ao serviço, este desaparece da lista de serviços disponíveis, de forma a evitar que o trabalhador se candidate duas vezes ao mesmo serviço. É possível o trabalhador avaliar o estado da sua candidatura na página "Meus serviços" na área de serviços pendentes. Na figura 61 e 62, é possível encontrar a interface da página *Web* e *App*, responsáveis por possibilitar que um trabalhador se candidate a um serviço.

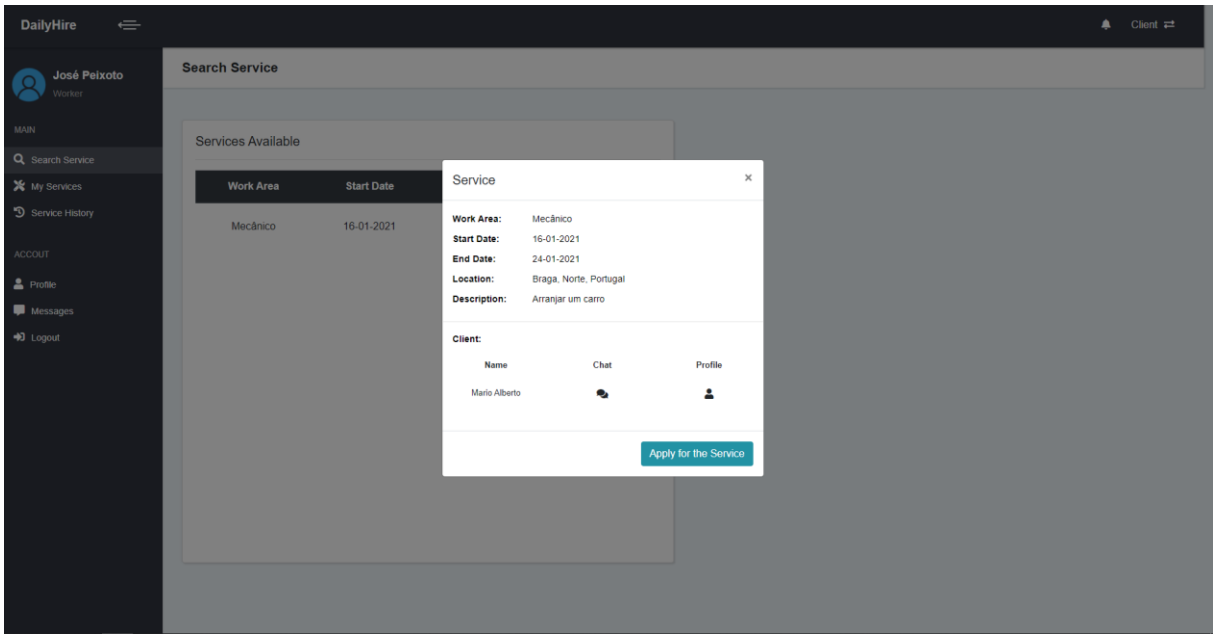

Figura 61 - Funcionalidade: Candidatar-se a um Serviço - Página *Web*

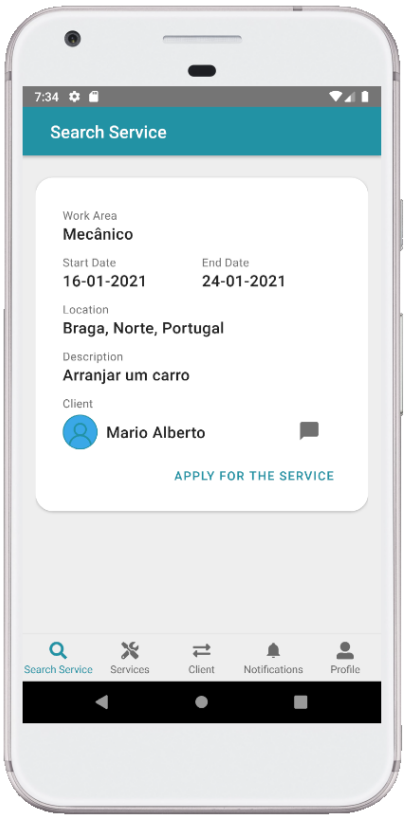

Figura 62 - Funcionalidade: Candidatar-se a um Serviço - *App*

5.2.4. Avaliar Cliente

Uma vez terminado o serviço por parte do cliente, o trabalhador associado a esse serviço tem a capacidade de avaliar o cliente que o contratou. Esta avaliação é efetuada numa escala de 1 a 5, sendo que, o trabalhador também tem a possibilidade de não avaliar o cliente, se assim o pretender. Na figura 63, é possível encontrar o diagrama de sequência desta funcionalidade.

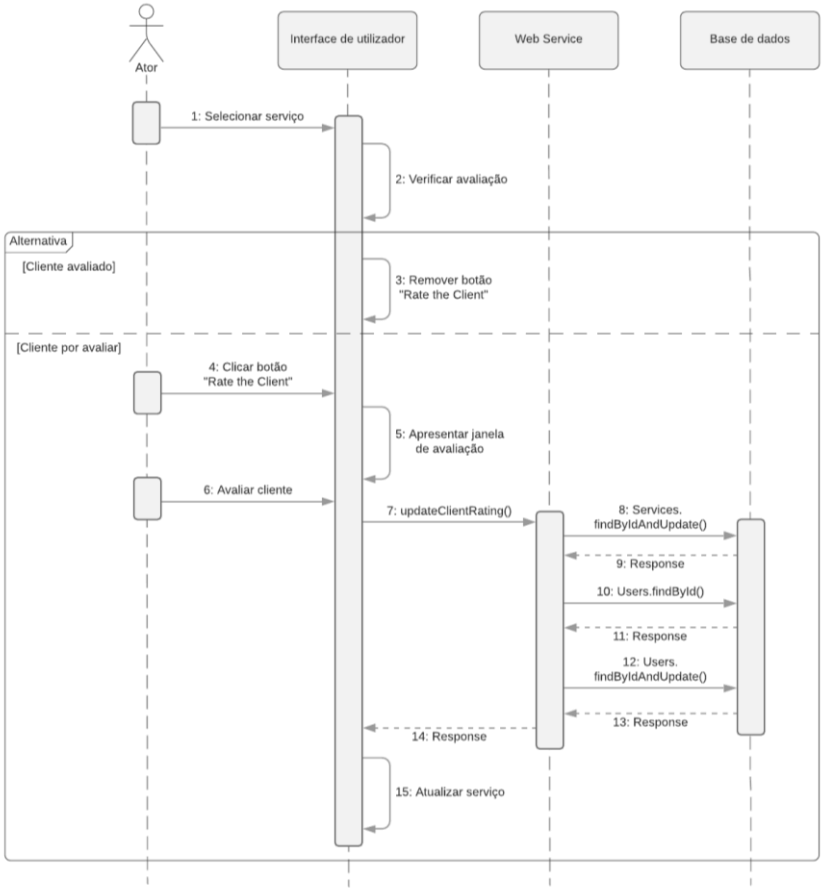

Figura 63 - Diagrama de Sequência: Avaliar Cliente

Caso o trabalhador já tenha efetuado a avaliação do cliente nesse serviço, a opção de avaliação não será apresentada. Caso contrário, o trabalhador poderá avaliar o cliente, contando para a avaliação de cliente desse utilizador. Esta avaliação corresponde à divisão do valor total de avaliações pelo número de avaliações efetuadas, estando disponível essa informação no perfil do cliente. Este parâmetro poderá influenciar um trabalhador no ato de se candidatar a um serviço, uma vez que, experiências passadas de outros trabalhadores podem ditar se se trata de um bom ou mau cliente. Na figura 64 e 65, é possível encontrar as interfaces da página *Web* e *App*, respetivamente.

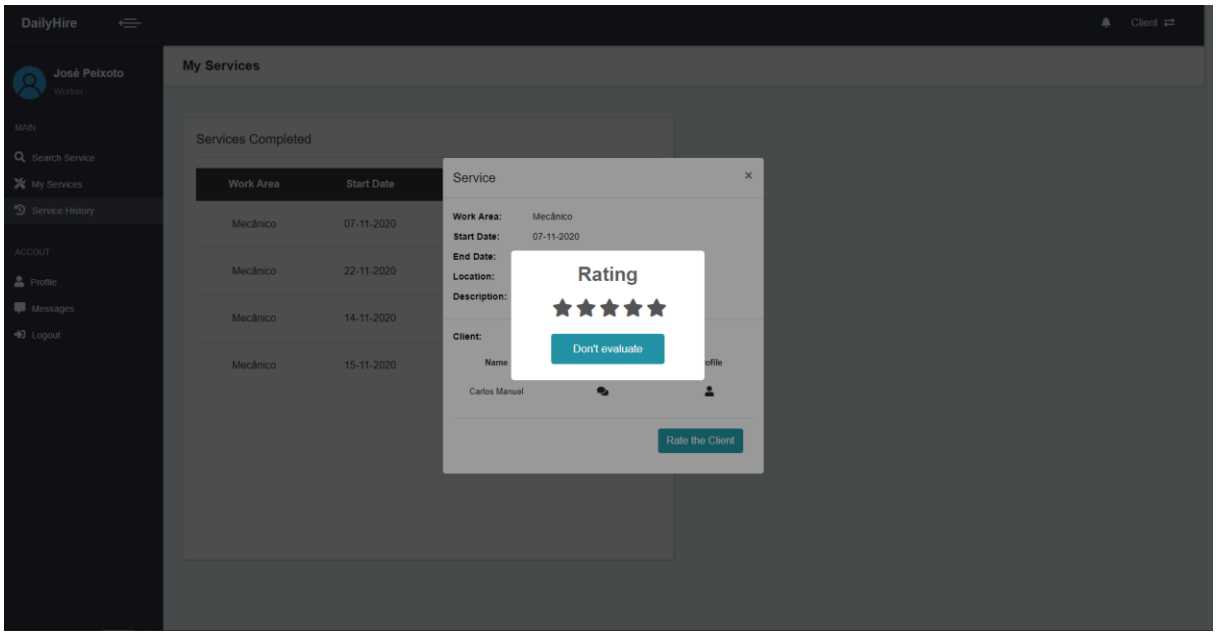

Figura 64 - Funcionalidade: Avaliar Cliente – Página *Web*

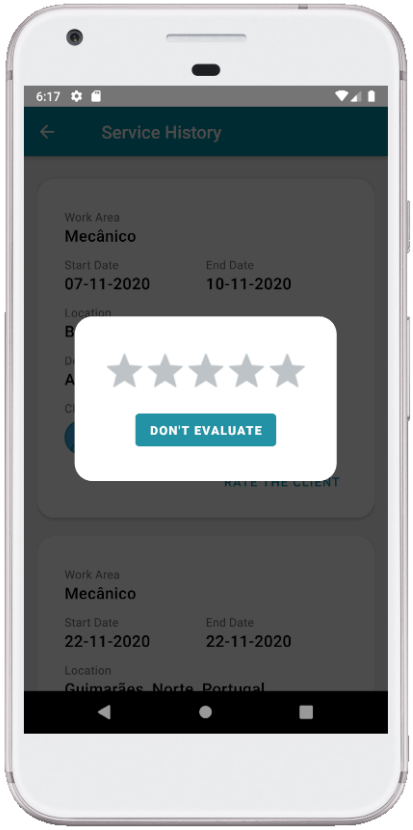

Figura 65 - Funcionalidade: Avaliar Cliente – *App*

6. TESTES DE USABILIDADE

Uma vez desenvolvidas as plataformas, é necessário testar a sua usabilidade. Para tal, é realizado os testes de usabilidade utilizando o método *Goal-Directed Design*, descrito na subsecção 3.2.4. Estes testes, têm como objetivo identificar dificuldades que os utilizadores encontrem na utilização do sistema, assim como, possíveis erros que impossibilitem o bom funcionamento das plataformas. Com isto, é possível reconhecer que funcionalidades necessitam de melhorias. Inicialmente, é necessário a elaboração de uma *persona*, responsável por contextualizar o participante do utilizador que irá representar. Uma vez elaboradas as *personas*, é necessário elaborar cenários. Os cenários correspondem a histórias, sobre determinada *persona*, e como esta se comportaria na execução de uma tarefa, tendo em conta o contexto apresentado.

Na secção 6.1, é possível encontrar as *personas* desenvolvidas, seguidas pelos cenários na secção 6.2. Por último, é apresentado na secção 6.3, os resultados obtidos dos testes de usabilidade efetuados.

6.1. *Personas*

Posto isto, é possível encontrar, de seguida, as *personas* desenvolvidas, utilizadas posteriormente para a contextualização dos participantes.

***Persona* Trabalhador:**

José Peixoto

38 anos

Braga, Portugal

Casado com dois filhos

Figura 66– Demonstração de uma Persona para um trabalhador. Retirado de (Random User Generator, 2021)

José Peixoto, é mecânico na zona de Braga há mais de 8 anos. Vive com a sua esposa e dois filhos. A sua rotina passa por chegar à oficina por volta das 8:30h e terminando o seu dia de trabalho por volta das 20:00h. José Peixoto é o dono da oficina, empregando mais 2 mecânicos. Ele adora o seu trabalho, uma vez que desde a sua adolescência desenvolveu o interesse por automóveis. Uma das vantagens da sua profissão, é a possibilidade de conduzir uma grande variedade de veículos, incluindo carros de luxo.

Com o avanço tecnológico, José Peixoto pretendia aumentar a divulgação da sua oficina, mas isto por sua vez, implicaria custos. Ficou reticente, pois, não saberia até que ponto o investimento na divulgação *Web* da oficina, resultaria em um aumento de clientes. Outra alternativa, passa por possibilitar a organizações, serem o seu intermediário para o cliente final, o que por outro lado, também apresenta pontos negativos, pois esta organizações pretendem obter uma percentagem do valor do serviço.

Tendo em conta o mencionado anteriormente, José Peixoto utiliza os serviços da DailyHire, pois possibilita a grátis divulgação dos seus serviços, sem necessitar de despender uma parte do valor dos serviços efetuados.

***Persona* Cliente:**

Figura 67 - Demonstração de uma Persona para um cliente. Retirado de (Random User Generator, 2021)

Sara Costa, trabalha para um banco como analista financeira. O seu emprego ocupa-lhe bastante do seu dia, tendo por vezes, trabalho excedente que traz para casa. O computador está presente na maior parte do seu dia, seja para o trabalho, ou para outros interesses. Sara decidiu que todo o seu esforço, deveria ser recompensado. Para tal, decidiu que merecia arranjar o seu automóvel e até mesmo efetuar alterações na sua casa, que até então não o tinha feito pois estava a economizar para o seu futuro.

Uma das dificuldades encontradas, passaria por encontrar um profissional adequado, pois não tem contacto com ninguém da área, nem conhecimento sobre a qualidade dos serviços. Outra dificuldade que pode ser encontrada, corresponde à falta de disponibilidade por parte da Sara, para realizar uma pesquisa detalhada de profissionais. Necessita de encontrar toda esta informação numa plataforma com o objetivo de a ajudar a escolher quem contratar.

6.2. Cenários

De seguida, são apresentados os cenários desenvolvidos, utilizados posteriormente com os participantes.

- Planeia se registar na plataforma DailyHire como trabalhador, uma vez que isto, poderá aumentar o seu número de clientes.
- Pretende renovar a pintura do seu quarto, no início do próximo mês. Para tal, necessita de requisitar o serviço de um pintor. Dirige-se à plataforma DailyHire com o objetivo de resolver este problema.
- Devido à pandemia atual, o seu volume de vendas diminuiu. Decidiu que a solução passaria por procurar serviços na plataforma DailyHire.
- Um serviço que anunciou já se encontra realizado, sendo que pretende atualizar essa informação na plataforma e avaliar o trabalhador responsável. Com isto, pretende informar futuros clientes do trabalhador em questão, da qualidade do seu serviço.
- Necessita dos serviços de um mecânico urgentemente, sendo que não pretende esperar por eventuais candidaturas de trabalhadores. Para tal, resolve pesquisar por mecânicos na sua área de residência e entrar em contacto com os mesmos.

6.3. Resultados dos Testes

Uma vez definidos os cenários e as *personas*, a próxima etapa é responsável por analisar a interação dos participantes com a plataforma. Para isso, foi utilizada uma amostra de cinco participantes, em janeiro de 2021, sendo a interação avaliada através da ferramenta zoom, que possibilita a partilha de ecrã e comunicação com os participantes, permitindo analisar os seus comportamentos na plataforma e a sugestões que indiquem. Cada participante, foi responsável por executar um determinado cenário, personificando a *persona* adequada. Após a avaliação dos resultados, objetivou-se a identificação de melhorias, assim como a correção de eventuais falhas que existissem no sistema. Outra questão a ter em atenção, é o número de etapas necessárias para o utilizador executar as tarefas necessárias. Ou seja, se o utilizador executou o percurso correto ou se cometeu erros, dirigindo-se a outras áreas da plataforma até eventualmente alcançar o objetivo. De seguida, são apresentadas as análises efetuadas aos cinco participantes.

No primeiro cenário, o participante personificou a *persona* José Peixoto, tendo como objetivo registar-se na plataforma DailyHire enquanto trabalhador. Inicialmente, houve resistência em se registar na plataforma, uma vez que a mesma não possuía uma opção que

identificasse o tipo de registo que estaria a efetuar, indicado pelo participante como "Não consigo encontrar a opção para me registar como trabalhador. Se me registar como sei se estou a fazê-lo como cliente ou trabalhador". Uma vez ultrapassada esta barreira, o participante, ao fim de algumas tentativas conseguiu alcançar o objetivo. Analisando as sugestões e dificuldades encontradas, a plataforma deveria diferenciar trabalhador e cliente na página inicial. Estes dois perfis, possuem objetivos diferentes, sendo que apresentar a mesma interface de registo, sem informação do que terá acesso, poderá suscitar duvidas ao utilizador. Outra melhoria mencionada, passa pela possibilidade de pesquisar por determinado serviço, sem a necessidade do utilizador enquanto cliente, se registar na plataforma.

No segundo cenário, o participante responsável pela sua execução, representou a *persona* Sara costa. Objetivou-se o anúncio de um serviço a requisitar um pintor. Indo de encontro ao mencionado no cenário anterior, houve dificuldade por parte do utilizador em diferenciar o perfil que se encontrava. A identificação *Announce Service* suscitou a ideia, que se encontrava como trabalhador, uma área responsável por permitir a trabalhadores anunciar serviços, referido pelo participante como "O nome anunciar serviço dá a sensação que sou eu que estou a anunciar um serviço, ou seja, que sou eu que estou disponível a realizar algo". Analisando os resultados desta interação, as plataformas deveriam possuir uma maior diferenciação entre os dois perfis. Com isto, seria possível o utilizador, facilmente identificar que perfil se encontrava e que funcionalidades seria capaz de executar. Para finalizar, a sugestão apresentada pelo participante passaria pela modificação da opção anunciar serviço para requisitar serviço.

Relativamente ao terceiro cenário, o participante deveria candidatar-se a um serviço, representando a *persona* trabalhador. Inicialmente, a questão de anunciar serviço voltou a suscitar confusão. Levou o participante a tentar anunciar um serviço. A conclusão obtida desta ação, vai de encontro ao mencionado anteriormente, de que esta funcionalidade dá a sensação aos utilizadores, que possibilita aos trabalhadores anunciar os seus serviços. Tal como referido na análise anterior, a melhoria encontrada seria a modificação do nome desta funcionalidade. Por último, a sugestão apresentada pelo participante, seria a alteração da cor de fundo do botão "*Worker*", com o objetivo de apelar o olhar do utilizador, no momento que este efetua o *login*. A conclusão obtida da interação deste participante com a plataforma, tal como o mesmo indicou "Ao início pensei que a área de anunciar serviço, seria para eu dizer que tipo de serviços faço, mas depois de entender o funcionamento, percebo qual o sentido", ou seja, o contacto inicial dos participantes com o sistema suscita confusão na diferenciação dos perfis, mas após uma maior utilização da plataforma, é possível entender o objetivo das funcionalidades.

Em relação ao quarto cenário, o participante deveria efetuar o login e posteriormente terminar um serviço, sendo que para efetuar esta operação seria necessário avaliar o trabalhador responsável pela execução do serviço. Uma vez que o participante, após iniciar sessão na plataforma já se encontrava na área de cliente, facilmente encontrou a área dos serviços abertos. Inicialmente, houve hesitação em terminar o serviço, pois a denominação do botão *Completed Service*, não seria a mais adequada. Face à dificuldade encontrada, a melhoria selecionada para o sistema, seria a modificação de *Completed Service* para *Finish Service*.

No último cenário, o participante tinha como objetivo encontrar um mecânico na sua área de residência. Para tal, deveria entrar em contacto com os mecânicos apresentados no mapa e que se enquadrassem com os requisitos, não esperando por eventuais candidaturas para o serviço. Uma vez que o participante após iniciar sessão na plataforma é deparado com o mapa, este facilmente selecionou a profissão pretendida de entre as várias opções disponíveis nos filtros. Uma vez apresentados no mapa todos os trabalhadores registados no sistema como mecânicos, o participante decidiu entrar em contacto com um mecânico localizado na zona de Braga. Analisando a interação do participante com o sistema, houve uma fácil interpretação da interface e dos elementos que a constituem, que levou a uma rápida execução do cenário. Apesar do resultado positivo desta interação, o participante mencionou uma melhoria que poderia ser implementada, em relação ao filtro da distância entre o utilizador e os trabalhadores apresentados no mapa. Atualmente, este filtro disponibiliza cinco intervalos, sendo que o participante sugeriu substituir por uma barra com intervalos variáveis, onde o utilizador poderia selecionar qualquer distância, e assim, aumentar a precisão dos resultados.

Uma vez implementados todos os cenários desenvolvidos, foi possível obter o conjunto melhorias sugeridas pelos participantes, assim como, possíveis alterações, que resolvessem as dificuldades encontradas. De seguida, é possível encontrar todas as melhorias para o sistema, resultantes dos testes de usabilidade.

- Diferenciar o tipo de registo na página inicial, entre trabalhador e cliente;
- Possibilitar o anúncio de serviços sem a necessidade de se registar na plataforma, permitindo verificar se o serviço pretendido possui os trabalhadores necessários registados no sistema;
- Alterar a opção *Announce Service* para *Request Service*;
- Modificar o aspeto do botão *Worker*, com o objetivo de apelar ao olhar dos utilizadores na sua primeira interação com a plataforma;
- Alterar a opção *Completed Service* para *Finish Service*;

〉 Substituir filtro do mapa com os intervalos de distância entre utilizador e trabalhadores, por uma barra com intervalos variáveis;

7. CONCLUSÃO

O objetivo principal desta dissertação, como apresentado na secção 1.2, é o desenvolvimento de uma solução capaz de aproximar trabalhadores não qualificados dos seus potenciais clientes, suportada no modelo de dados DailyHire.

Numa fase inicial desta dissertação, foram abordadas as temáticas e aplicações semelhantes, relacionadas com a dissertação. Face à primeira temática, Economia Social e Solidária, perspetivou-se os conceitos abordados e os seus objetivos. Em relação à segunda temática, *Linked Data*, foram apresentadas as suas principais características, assim como, uma contextualização sobre RDF, futuramente usado na fase de desenvolvimento. De seguida, foi efetuada a contextualização do Projeto EMPOWER-SSE, sendo o projeto no qual esta dissertação se enquadra. Aqui também é possível visualizar o trabalho anteriormente desenvolvido, que sofreu alterações de modo a adaptar o mesmo a esta dissertação. Para finalizar a contextualização desta dissertação, foram apresentadas as plataformas e aplicações existentes, com o objetivo de avaliar e reunir ideias, que diferenciassem a aplicação desenvolvida das restantes.

Numa segunda fase, foram abordados os materiais e métodos utilizados ao longo do desenvolvimento desta dissertação. Este divide-se em duas secções, sendo inicialmente apresentadas as plataformas e ferramentas utilizadas devidamente justificadas, seguido dos procedimentos metodológicos.

Tendo em conta a metodologia FDD utilizada nesta dissertação, é apresentada a *Initial Modeling*, responsável pela elaboração do modelo detalhado do sistema e da lista de funcionalidades deste. Relativamente ao modelo anteriormente mencionado, foi apresentada inicialmente a arquitetura tecnológica, detalhando o funcionamento de todo o sistema e interações existentes entre as várias ferramentas. Após a apresentação da arquitetura, incluiu-se o modelo entidade relacionamento, representando a organização dos dados do sistema. Em conclusão da *Initial Modeling*, foi também apresentada a validação dos requisitos funcionais, responsável pela elaboração final da lista de funcionalidades.

Pretendendo demonstrar as aplicações desenvolvidas, apresentou-se as funcionalidades existentes no sistema e o seu funcionamento. Por último, é efetuada uma avaliação aos resultados obtidos, através de testes de usabilidade. Nesta avaliação, desenvolveu-se *personas* e cenários, utilizados por uma amostra de participantes, com o objetivo de identificar erros e eventuais melhorias a ser implementadas.

7.1. Resultados Obtidos

Nesta dissertação, pode-se concluir que os seus resultados obtidos foram significativamente satisfatórios, uma vez que foi alcançado o objetivo e resultados esperados. O desenvolvimento de uma página *Web* e uma *App*, com funcionalidades inovadoras capazes de diferençar de aplicações já existentes. Estas plataformas permitem assim satisfazer o objetivo definido, aproximando trabalhadores não qualificados dos seus potenciais clientes. Para tal, as plataformas possibilitam aos seus utilizadores anunciar serviços, troca de mensagens entre trabalhador e cliente, aprovar trabalhadores, candidatar-se a executar um serviço, entre outros. De salientar que a utilização da plataforma DailyHire apresenta vantagens desde, não possuir qualquer interação com questões monetárias, ou seja, não retira qualquer valor aos seus utilizadores pela utilização das plataformas, o que ao contrário, de várias aplicações já existentes isso não acontece. Outra vantagem e também um resultado esperado nesta dissertação, corresponde à possibilidade de obter os dados sob a forma de *Linked Data*, sendo para tal atualizado e adaptado o perfil de aplicação elaborado pelos estudantes indianos do *National Institute of Technology* em colaboração com os docentes de investigadores da Universidade do Minho.

Foi ainda definido nos resultados esperados, a disponibilização do código desenvolvido nesta dissertação, estando assim disponível em repositórios na plataforma GitHub, de forma, a possibilitar a sua utilização por outras pessoas.

Por último, era pretendido disponibilizar a plataforma *Web* para utilizadores externos e não apenas localmente, ou seja, divulgar o trabalho desenvolvido permitindo a qualquer pessoa utilizar o sistema. Este resultado esperado também foi alcançado, estando a plataforma *Web* disponível no endereço www.daily-hire.net, o que viabiliza o trabalho desta dissertação, uma vez que, a plataforma encontra-se neste momento operacional.

7.2. Trabalho Futuro

Apesar dos objetivos definidos terem sido alcançados, existe a possibilidade de implementar melhorias ao sistema. Posto isto, de seguida são detalhadas algumas sugestões de possíveis melhorias:

- Aumentar a segurança da plataforma. Esta segurança pode ser efetuada de diversas maneiras: encriptação dos dados dos utilizadores, utilização de *tokens* para restringir o acesso às funcionalidades do *Web Service*, entre outras possibilidades.
- Completar o desenvolvimento do perfil de aplicação, utilizados na criação dos triplos RDF, visível nos apêndices (Matriz de Restrições).
- Implementação das sugestões mencionadas na secção 6.3, consideradas como benéficas para o sistema.
- Realizar mais testes de usabilidade.
- Utilizar um *triplestore* em vez de uma base de dados relacional.
- Atualmente a informação exportada como *Linked Data* é pré-definida pelo sistema. No futuro, seria interessante implementar uma funcionalidade que permita aos utilizadores selecionar que dados pretendem que sejam públicos e privados.

REFERÊNCIAS BIBLIOGRÁFICAS

Abramova, V., & Bernardino, J. (2013, July). NoSQL databases: MongoDB vs cassandra. In Proceedings of the international C* conference on computer science and software engineering (pp. 14-22).

Allemang, D., & Hendler, J. (2011). Semantic web for the working ontologist: effective modeling in RDFS and OWL. Elsevier.

Bauer, F., & Kaltenböck, M. (2011). Linked open data: The essentials. Edition mono/monochrom, Vienna, 710.

Bauer, M. (2005). Successful Web Development Methodologies Article. https://www.sitepoint.com/successful-development/.

Berners-Lee, T., & Hendler, J. (2001). Publishing on the semantic web. Nature, 410(6832), 1023-1024.

Bizer, C. (2009). The emerging web of linked data. IEEE intelligent systems, 24(5), 87-92.

Bizer , C., Cyganiak, R., & Heath, T. (2007). How to Publish Linked Data on the Web? http://wifo5-03.informatik.uni-mannheim.de/bizer/HowtoPublishLinkedData.htm.

Bizer, C., Heath, T., & Berners-Lee, T. (2011). Linked data: The story so far. In Semantic services, interoperability and web applications: emerging concepts (pp. 205-227). IGI Global.

Carvalho, A. A. A. (2002). Testes de Usabilidade: exigência supérflua ou necessidade. In Actas do 5º Congresso da Sociedade Portuguesa de Ciências da Educação (pp. 235-242).

Chodorow, K. (2013). MongoDB: the definitive guide: powerful and scalable data storage. " O'Reilly Media, Inc.".

Chowdhury, A. F., & Huda, M. N. (2011, December). Comparison between adaptive software development and feature driven development. In Proceedings of 2011 International Conference on Computer Science and Network Technology (Vol. 1, pp. 363-367). IEEE.

Courage, C., & Baxter, K. (2005). Understanding your users: A practical guide to user requirements methods, tools, and techniques. Gulf Professional Publishing.

Danielsson, W. (2016). React Native application development. Linköpings universitet, Swedia, 10, 4.

Devedzic, V. (2006) Semantic Web and Education. Springer.

Drechsler, A., & Hevner, A. (2016). A four-cycle model of IS design science research: capturing the dynamic nature of IS artifact design. DESRIST 2016.

Economia social. (2019). In Wikipedia. Retrieved from https://pt.wikipedia.org/w/index.php?title=Economia_social&oldid=55268127.

EJS -- Embedded JavaScript templates. (2021). EJS. https://ejs.co/#about

Firmino, H. (2013) Organização e Publicação dos Termos do Website da ANACOM sob uma Perspetiva Linked Open Data. 135.

Gruber, T. (2008). Collective knowledge systems: Where the social web meets the semantic web. Journal of web semantics, 6(1), 4-13.

Heath, T., & Bizer, C. (2011). Linked data: Evolving the web into a global data space. Morgan & Claypool Publishers.

Hendler, J. A. (2009). Tonight's Dessert: Semantic Web Layer Cakes. In European Semantic Web Conference (pp. 1-1). Springer, Berlin, Heidelberg.

Henriques, J. M. (2010). Crise, economia social e solidária e 'integração económica na acção contra a pobreza. Rev Econ Solid, 2, 83-114.

Heroku. (2021). Heroku. https://www.heroku.com/about

Hevner, A. R. (2007). A three cycle view of design science research. Scandinavian journal of information systems, 19(2), 4.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. MIS quarterly, 75-105.

Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P. F., & Rudolph, S. (2009). OWL 2 web ontology language primer. W3C recommendation, 27(1), 123.

Isotani, S., Mizoguchi, R., Bittencourt, I. I., & Costa, E. (2008). Web 3.0-Os rumos da Web semântica e da Web 2.0 nos ambientes educacionais. In Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE) (Vol. 1, No. 1, pp. 785-795).

Jovanovic, J., Torniai, C., Gasevic, D., Bateman, S. & Hatala, M. (2008) Leveraging the Social Semantic Web in Intelligent Tutoring Systems. In Proceedings of the International Conference on Intelligent Tutoring Systems, LNCS 5091, 563-572.

Lechat, N. M. P. (2002). Economia social, economia solidária, terceiro setor: do que se trata?. Civitas-Revista de Ciências Sociais, 2(1), 123-140.

Lenon (2018). Node.js – O que é, como funciona e quais as vantagens. Opus Software. https://www.opus-software.com.br/node-js/

Malta, M (2014). Contributo metodológico para o desenvolvimento de perfis de aplicação no contexto da Web Semântica.

Malta, M (2020). Me4MAP: Um método para o desenvolvimento de perfis de aplicação de metadados. Association for Information Science and Technology | ASIS&T. https://www.asist.org/meetings-events/webinars/me4map-um-metodo-para-o-desenvolvimento-de-perfis-de-aplicacao-de-metadados/

Marques, F. (2017). Economia Social e Solidária em afirmação na União Europeia. [online] euronews Available at: https://pt.euronews.com/2017/02/10/economia-social-e-solidaria-em-afirmacao-na-uniao-europeia.

McCloskey, M. (2014). Task Scenarios for Usability Testing. Nielsen Norman Group. https://www.nngroup.com/articles/task-scenarios-usability-testing/

Mikroyannidis, A. (2007). Toward a social semantic web. Computer, 40(11), 113-115.

Mizoguchi, R., Hayashi, Y. and Bourdeau, J. (2007) Inside Theory-Aware Authoring System. In Proceedings of the Int. Workshop on Ontologies and Semantic Web for E-Learning (SWEL), 1- 18.

Murugesan, S. (2007) Understanding Web 2.0. IEEE IT Professional, 9(4), 34-41.

Novick, V. (2017). React Native-Building Mobile Apps with JavaScript. Packt Publishing Ltd.

OWL (2008) Web Ontology Language. http://www.w3.org/TR/owl-features/.

Paule, N. M. L. (2002). Economia social, economia solidária, terceiro setor: do que se trata?. Civitas-Revista de Ciências Sociais, 2(1), 123-140.

Paulheim, H., & Bizer, C. (2014). Improving the quality of linked data using statistical distributions. International Journal on Semantic Web and Information Systems (IJSWIS), 10(2), 63-86.

Random User Generator (2021). Random User Generator. https://randomuser.me/photos

RIPESS and Emily Kawano (2012). Differences and Convergences in Social Solidarity Economy Definitions, Concepts and Frameworks. RIPESS Working Paper.

Rodrigues, J. (2021). MER e DER: Modelagem de Bancos de Dados. DevMedia. https://www.devmedia.com.br/modelo-entidade-relacionamento-mer-e-diagrama-entidade-relacionamento-der/14332

Sen, S., Rasa, N., Curado Malta, M., Baptista, A. A.. (2018). Developing a Metadata Application Profile for the Daily Hire Labour. In M. Curado Malta and K. Eckert (Eds), Proceedings of the 2018 International Conference on Dublin Core and Metadata Applications (pp.81-83).

Souza, R. R., & Alvarenga, L. (2004). A Web Semântica e suas contribuições para a ciência da informação. Ciência da Informação, 33(1), 132-141.

Tilkov, S., & Vinoski, S. (2010). Node. js: Using JavaScript to build high-performance network programs. IEEE Internet Computing, 14(6), 80-83.

Validação de Requisitos (2020). In Wikipedia. Retrieved from https://pt.wikipedia.org/wiki/Validação_de_requisitos.

Vieira, J. (2019). Teste de usabilidade: tudo o que você precisa saber! Medium. https://medium.com/aela/teste-de-usabilidade-o-que-voc%C3%AA-precisa-saber-39a36343d9a6

Volpato, E. (2014). Teste de usabilidade: o que é e para que serve? Medium. https://brasil.uxdesign.cc/teste-de-usabilidade-o-que-%C3%A9-e-para-que-serve-de3622e4298b

Why (and How) You Should Use Feature-Driven Development. (2021). [online] Lucidchart Available at: https://www.lucidchart.com/blog/why-use-feature-driven-development.

APÊNDICES

Atributos do Modelo Entidade Relacionamento

A Tabela 2 apresenta a descrição dos atributos expostos no modelo entidade relacionamento.

Tabela	Coluna	Tipo dos dados	Descrição
User	_id	ObjectId	Identificador único
	name	String	Nome do utilizador
	address	String	Morada do utilizador
	email	String	Email do utilizador, sendo um atributo único
	phoneNumber	Number	Contacto telefónico do utilizador
	gender	String	Género do utilizador
	birthdate	Date	Data de nascimento do utilizador
	password	String	*Password* do utilizador
	workerState	Boolean	Identificador se o utilizador se encontra registado como trabalhador
	ratingCounter	Number	Número de avaliações efetuadas ao utilizador, enquanto cliente
	ratingTotal	Number	Soma de todas as avaliações efetuadas ao utilizador, enquanto cliente
	photo	String	Endereço correspondente da fotografia do utilizador
Worker	_id	ObjectId	Identificador único
	user	ObjectId	Identificador do utilizador associado
	area	String	Profissão do trabalhador
	experience	Number	Número de anos que exerce a profissão
	ratingCounter	Number	Número de avaliações efetuadas ao utilizador, enquanto trabalhador
	ratingTotal	Number	Soma de todas as avaliações efetuadas ao utilizador, enquanto trabalhador
Service	_id	ObjectId	Identificador único

	workArea	String	Profissão requisitada para o serviço
	startDate	Date	Data de início do serviço
	endDate	Date	Data de término do serviço
	address	String	Localização onde o serviço irá ocorrer
	description	String	Descrição sobre o serviço
	rating	Number	Avaliação numa escala de 1 a 5, ao trabalhador responsável pela execução do serviço
	state	String	Estado do serviço, variando entre "*Pending*", "Open", "*Canceled*" e "*Finished*"
	clientRating	Number	Avaliação numa escala de 1 a 5, ao cliente que anunciou o serviço
	user	ObjectId	Identificador do utilizador associado
Relation	_id	ObjectId	Identificador único
	service	ObjectId	Identificador do serviço associado
	worker	ObjectId	Identificador do trabalhador associado
	state	Boolean	Estado da relação. Identifica se a candidatura do trabalhador a um serviço foi provada
Message	_id	ObjectId	Identificador único
	content	String	Conteúdo da mensagem
	chat	ObjectId	Identificador do *chat* associado
Chat	_id	ObjectId	Identificador único
	room	ObjectId	Identificador da sala que os utilizadores em questão utilizam para a comunicação em tempo real
	participant1	ObjectId	Identificador do utilizador associado, enquanto cliente
	Participant2	ObjectId	Identificador do utilizador associado, enquanto trabalhador
Notification	_id	ObjectId	Identificador único

	content	String	Mensagem que irá ser apresentada ao utilizador
	user	ObjectId	Identificador do utilizador associado

Tabela 3 - Descrição dos atributos do DER

Matriz de Restrições

Na Tabela 3 é apresentada a matriz de restrições, para os atributos considerados como públicos, relativos ao utilizador, e na Tabela 4, os atributos relativos aos serviços.

Namespaces	
dct	http://purl.org/dc/terms/
foaf	http://xmlns.com/foaf/0.1/
vcard	http://www.w3.org/2006/vcard/ns#
schema	https://schema.org/
frapo	http://purl.org/cerif/frapo/
empower	http://purl.org/empower-sse/vocab/

Domínio no MAP	empower:User rdfs:subclassOf foaf:Person	to describe resources rdf:type empower:User

Nome do Termo	Descrição	Propriedade	Valores Permitidos	Cardinalidade	Modelo Relacional Entidade.Atributo
identifier	Um identificador do utilizador, de preferência um URI	dct:identifier	xsd:anyURI	1	User._id
name	Nome completo do utilizador	foaf:name	xsd:string	1	User.name

serviceType	O tipo de serviço oferecido, p.e. Eletricidade, Mecânico, Pintor	schema:serviceType	Vocabulário controlado de profissões / xsd:string	0…1	Worker.area
typeOfUser	Por padrão, um utilizador é considerado cliente (valor=0). Adicionalmente, pode-se tornar trabalhador (valor=1). Este termo indica se o utilizador é também considerado trabalhador, em adição ao tipo cliente	dct:type	xsd:boolean	1	User.workerState
ratingCountAsClient	Número total de avaliações efetuadas por todos os trabalhadores que realizaram serviços a este utilizador-cliente	Nova propriedade empower:ratingCountAsClient que será rdfs:subPropertyOf https://schema.org/ratingCount	xsd:integer	0…1	User.ratingCounter
ratingValueAsClient	Média das avaliações efetuadas por todos os trabalhadores que realizaram serviços a este utilizador-cliente	Nova propriedade empower:ratingValueAsClient que será rdfs:subPropertyOf https://schema.org/ratingValue	xsd:float	0…1	User.ratingTotal

ratingCountAsWorker	Número total de avaliações efetuadas por todos os clientes que requisitaram serviços a este utilizador-trabalhador	Nova propriedade empower:ratingCountAsClient que será rdfs:subPropertyOf https://schema.org/ratingCount	xsd:integer	0…1	Worker.ratingCounter
ratingValueAsWorker	Média das avaliações efetuadas por todos os clientes que requisitaram serviços a este utilizador-trabalhador	Nova propriedade empower:ratingValueAsWorker que será rdfs:subPropertyOf https://schema.org/ratingValue	xsd:float	0…1	Worker.ratingTotal
yearsOfExperience	Número de anos de experiência, que o trabalhador possui	Nova propriedade empower:yearsOfExperience que será rdfs:subPropertyOf http://dbpedia.org/ontology/years e de http://www.w3.org/2006/time#years	xsd:integer	0…1	Worker.experience
offers	Identificador do serviço anunciado pelo utilizador-cliente	schema:offers	schema:Service	0…n	Relação User->Service
appliesFor	Identificador do serviço que o utilizador-trabalhador se candidatou	frapo:appliesFor	schema:Service	0…n	Relação User->Service (Através da entidade Relation)

Tabela 4 - Matriz de restrições – Utilizadores

Domínio no MAP	schema:Service	to describe resources rdf:type schema:Service

Nome do Termo	Descrição	Propriedade	Valores Permitidos	Cardinalidade	Modelo Relacional Entidade.Atributo
identifier	Um identificador do serviço, de preferência um URI	dct:identifier	xsd:anyURI	1	Service._id
serviceType	O tipo de serviço oferecido, p.e. Eletricidade, Mecânico, Pintor	schema:serviceType	Vocabulário controlado de profissões / xsd:string	0…1	Service.workArea
startDate	Data em que o serviço está definido para iniciar	schema:startDate	xsd:date	0…1	Service.startDate
endDate	Data em que o serviço está definido para terminar	schema:endDate	xsd:date	0…1	Service.endDate
description	Descrição do serviço pelo utilizador-cliente	dct:description	xsd:string	1	Service.description
status	Estado do serviço. O serviço pode ter os seguintes estados: “Pending”, “Open”, “Canceled”, “Completed”	schema:status	Vocabulário controlado: “Pending”, “Open”, “Canceled”, “Completed”	1	Service.status

Tabela 5 - Matriz de restrições – Serviço

ANEXO

Matriz de Restrições Obsoleta

Namespaces	
foaf	http://xmlns.com/foaf/0.1/
vard	http://www.w3.org/2006/vcard/ns
time	http://www.w3.org/2006/time
acrt	http://privatealpha.com/ontology/certification/1#
dcterms	http://purl.org/dc/terms/
schema	http://schema.org/
juso	http://rdfs.co/juso/
essglobal	http://purl.org/essglobal/vocab/
dbpedia-owl	http://dbpedia.org/ontology/

Entidade: Trabalhador	Rdfs:subClassOf foaf:Person

Nome do Atributo	*Label*	Prefixo *namespace*	Propriedade	Descrição	*Linked/Linked Open Data*	Domínio Original	Alcance Original	Domínio no MAP	Alcance no MAP	Cardinalidade
Name	name	foaf	name	Nome do trabalhador	LOD	owl:Thing	Literal	dh:Person	xsd:string	1
Gender	gender	foaf	gender	Especificação Masculino/ Feminino	LOD	foaf:Agent	Inespecífico	dh:Person	xsd:string	1
Birthday	Birth date	vcard	bday	Data de nascimento do trabalhador	LOD	Inespecífico	xsd:dateTime; xsd:dateTime Stamp;	dh:Person	xsd:date	1

								xsd:gYear		
Image	image	foaf	img	Fotografia do trabalhador	LOD	foaf:Person	foaf:Image	dh:Person	Não literal	1
Has Default Pay Visit	Has Default Pay Visit	dh	HasDefaultPayVisit	O pagamento padrão do pagamento diário do trabalhador de acordo com a sua habilidade	LOD	Inespecífico	Inespecífico	dh:Worker	Literal (ISSO 4217)	0..1
Experience in Years	years duration	time	years	Experiência em anos do trabalhador	LOD	GeneralDurationDescription	xsd:Decimal	dh:Worker	Literal	1
has-certification	Has certification	acrt	acrt:has-certification	Os detalhes da certificação de habilidade, caso o trabalhador possua	LOD	foaf:Agent	acrt:Certification	dh:Worker	acrt:Certification	0..1
Has place preference	has Place Preference	vcard	locality	A preferência do trabalhador, do local para executar os seus serviços	LOD	Inespecífico	Não literal	dh:Worker	xsd:String; http://purl.org/dc/terms/TGN	1..n
Language Preference	Language	dcterms	language	A preferência de idioma do trabalhador	LOD	Inespecífico	dcterms:LinguisticSystem	dh:Person	ISO 639-3 (http://www-01.sil.org/iso639-3/codes.asp)	1..n
Fax	Fax Number	schema	faxNumber	Especificação do número do Fax	LOD	Inespecífico	Inespecífico	dh:Person	Literal	1..0
Mobile-Phone	Mobile-Phone	schema	telephone	Contacto telefónico do trabalhador	LOD	Inespecífico	Literal	dh:Person	Literal	1..n
Mail	mail-ID	foaf	mbox	Especificação do *Mail* ID	LOD	schema:ContactPoint; schema:Organization; schema:Person; schema:Place	schema:Text	dh:Person	Literal	1..0

Skill	Skill	dh	SkillType	A especialização/ habilidade do trabalhador	LOD	Inespecífico	Inespecífico	dh:Worker	Vocabulário controlado de SkillType	1..n
Pay Amount	Pay Amount	dh	PayAmount	Pagamento do trabalhador consoante a sua habilidade	LOD	Inespecífico	Inespecífico	dh:Worker	Literal (ISO 4117)	0..1
has type of Pay Amount	has type of Pay Amount	dh	hasTypeofPay Amount	Tipo de pagamento do trabalhador, de acordo com o estabelecido por este (hora, dia, mês, etc.)		Inespecífico	Inespecífico	dh:Worker	Vocabulário controlado de PayAmountType	0..1
is Member Of	isPartOf	dcterms	isPartOf	Esta propriedade ajuda a identificar o utilizador como membro de uma organização ou grupo		Inespecífico	Inespecífico	dh:Worker	dh:WorkerGroup	1..n
Has Home Address	Address	essglobal	hasAddress	A morada do escritório do grupo de trabalhadores		-	-	dh:Worker	dh:Address	1

- *Object Property*

Tabela 6 - Matriz de restrições obsoleta - Trabalhador

Entidade: Cliente	Rdfs:subClassOf foaf:Person									
Nome do Atributo	***Label***	**Prefixo *namespace***	**Propriedade**	**Descrição**	***Linked/Linked Open Data***	**Domínio Original**	**Alcance Original**	**Domínio no MAP**	**Alcance no MAP**	**Cardinalidade**
Name	name	foaf	name	Nome do cliente		owl:Thing	Literal	dh:Person	xsd:string	1
Gender	gender	vcard	hasGender	Especificação Masculino/ Feminino		vcard	Inespecífico	dh:Person	xsd:string	1
Birthday	Birth date	vcard	bday	Data de nascimento do cliente		Inespecífico	xsd:dateTime; xsd:dateTimeStamp; xsd:gYear	dh:Person	xsd:dateTime	1
Language Preference	Language	dcterms	language	A preferência de idioma do cliente		Inespecífico	dcterms:LinguisticSystem	dh:Person	ISO 639-3 (http://www-01.sil.org/iso639-3/codes.asp)	1..n
Image	image	foaf	img	Fotografia do cliente		foaf:Person	foaf:Image	dh:Person	Não literal	1
Fax	Fax Number	schema	faxNumber	Especificação do número do Fax		Inespecífico	Inespecífico	dh:Person	Literal	1..0
Mobile-Phone	Mobile-Phone	schema	telephone	Contacto telefónico do cliente		Inespecífico	Literal	dh:Person	Literal	1..n
Mail	mail-ID	foaf	mbox	Especificação do *Mail* ID		schema:ContactPoint; schema:Organization; schema:Person; schema:Place	schema:Text	dh:Person	Literal	1..0
Has Home Address	Address	essglobal	hasAddress	A morada cliente		-	-	dh:Employer	dh:Address	1
Books Appointment	Books	dh	BooksAppointment	A propriedade que permite o cliente agendar um encontro		-	-	dh:Employer	Dh:Appointment	1

- *Object Property*

Tabela 7 - Matriz de restrições obsoleta - Cliente

Entidade: WorkerGroup	Rdfs:subClassOf essglobal:SSEInitiative									
Nome do Atributo	***Label***	**Prefixo *namespace***	**Propriedade**	**Descrição**	***Linked/Linked Open Data***	**Domínio Original**	**Alcance Original**	**Domínio no MAP**	**Alcance no MAP**	**Cardinalidade**
Group Name	Group Name	gr	name	Nome do grupo da organização		Inespecífico	Literal	dh:WorkerGroup	xsd:String	1
TotalOfMembers	Total No Of Members	essglobal	totalOfMembers	Número total de membros da organização		Inespecífico	Literal	dh:WorkerGroup	Literal	1
Total of Women	Total no of Women	essglobal	totalWomen	Número total de mulheres na organização		Inespecífico	Literal	dh:WorkerGroup	Literal	1
Total of Men	Total of Men	essglobal	totalMen	Número total de homens na organização		Inespecífico	Literal	dh:WorkerGroup	Literal	1
Type of the Group	Group type	dh	GroupType	Tipo do grupo de acordo com a habilidade dos membros		Inespecífico	Inespecífico	dh:WorkerGroup	xsd: String	
Home PPage	Home Page	foaf	homepage	A página da organização		foaf:Online Account	foaf:Document	dh:WorkerGroup	Não literal	1
Group logo	logo	dbpedia-owl	logo	Logotipo que identifica o grupo de trabalho		Inespecífico		dh:WorkerGroup	xsd:String	0..1
Date Of Constitution	Date Of Constitution	dh	DateOrgCreated	Data na qual a organização foi constituída		-	-	dh:WorkerGroup	xsd:Date	1
Has place preference	has Place Preference	vcard	locality	O local de preferência do grupo para executar trabalhos		Inespecífico	Não literal	dh:WorkerGroup	xsd:String; http://purl.org/dc/terms/TGN	1..n
Language Preference	Language	dcterms	language	A preferência de idioma do grupo de trabalho		Inespecífico	dcterms:LinguisticSystem	dh:WorkerGroup	ISO 639-3 (http://www-01.sil.org/iso6	1..n

								39-3/codes.asp)	
Default Pay Visit	Has Default Pay Visit	dh	HasDefaultPayVisit	O pagamento padrão do pagamento diário do grupo	Inespecífico	Inespecífico	dh:WorkerGroup	Literal (ISO 4217)	0..1
Pay Amount	Pay Amount	dh	PayAmount	O pagamento do trabalhador de acordo com a sua habilidade	Inespecífico	Inespecífico	dh:WorkerGroup	Literal (ISO 4217)	0..1
Skill	Skill	dh	SkillType	A especialização/habilidade do trabalhador	Inespecífico	Inespecífico	dh:Worker	Vocabulário controlado de SkillType	1..n
has type of Pay Amount	has type of Pay Amount	dh	hasTypeofPayAmount	Tipo de pagamento do trabalhador, de acordo com o estabelecido por este (hora, dia, mês, etc.)	Inespecífico	Inespecífico	dh:WorkerGroup	Vocabulário controlado de PayamountType	0..1
Fax	Fax Number	schema	faxnumber	Especificação do número do Fax	Inespecífico	Inespecífico	dh:WorkerGroup	Literal	1..0
Mobile-Phone	Mobile-Phone	schema	telephone	Contacto telefónico	Inespecífico	Literal	dh:WorkerGroup	Literal	1..n
Mail	mail-ID	foaf	mbox	Especificação do *Mail* ID	schema:Conta	schema:Text	dh:WorkerGroup	Literal	1..0
form With	isFormwith	dh	isFormwith	Um grupo de trabalho é formado com alguns trabalhadores	-	-	dh:WorkerGroup	dh:Worker	1..n
Has Office Address	Address	essglobal	hasAddress	A morada do escritório do grupo de trabalhadores	-	-	dh:WorkerGroup	dh:Address	1

- *Object Property*

Tabela 8 - Matriz de restrições obsoleta - Grupo de trabalhadores

Entidade: Address	subClassOf juso:Address									
Nome do Atributo	***Label***	**Prefixo *namespace***	**Propriedade**	**Descrição**	***Linked/Linked Open Data***	**Domínio Original**	**Alcance Original**	**Domínio no MAP**	**Alcance no MAP**	**Cardinalidade**
Postal Code	Postal Code	vcard	postal-code	Código postal da morada		Inespecífico	xsd:literal	dh:Address	Literal	1
Building Name	Building Name	dh	buildingName	Nome do edifício correspondente à morada		-	-	dh:Address	Literal	1
Landmark	Landmark	dh	landmark	A *landmark* correspondente, para fácil identificação da morada		-	-	dh:Address	Literal	1
Street	Street Address	juso	full_address	Nome da rua do local ao qual a morada corresponde		juso:Address	xsd:String	dh:Address	xsd:string , http://purl.org/dc/terms/TGN	1
City	City	vcard	locality	Localidade, cidade da morada indicada		Inespecífico	Não literal	dh:Address	xsd:string , http://purl.org/dc/terms/TGN	1
State	State	essglobal	state	Estado do país, da morada indicada		Inespecífico	Literal	dh:Address	Literal	1
Country	Country	vcard	country-name	O país da morada indicada		Inespecífico	xsd:String	dh:Address	Não literal	1

Tabela 9 - Matriz de restrições obsoleta - Morada

Entidade: Appointment										
Nome do Atributo	***Label***	**Prefixo *namespace***	**Propriedade**	**Descrição**	***Linked/Linked Open Data***	**Domínio Original**	**Alcance Original**	**Domínio no MAP**	**Alcance no MAP**	**Cardinalidade**
Has Start DateTime	Start Date Time	dh	StartDateTime	A data e hora de início do encontro		-	-	dh:Appointment	xs:DateTime	1
Has End DateTime	End Date Time	dh	EndDateTime	A data e hora de término do encontro		-	-	dh:Appointment	xs:DateTime	1
Has Appointment description	Has Appointment description	dcterms	description	A descrição do encontro		Inespecífico	Inespecífico	dh:Appointment	Literal	1
Has Payment Method	Has Payment Method	dh	PaymentMethod	O método de pagamento do encontro		-	-	dh:Appointment	Novo vocabulário controlado de PaymentMethod	1
Having PaymentType	Payment Type	dh	PaymentType	O tipo de pagamento ao trablhador		-	-	dh:Appointment	Novo vocabulário controlado de PaymentType	1..n
Has PieceType Order Of	Has PieceType Order Of	dh	HasPieceType OrderOf	A peça de trabalho, caso exista, que o trabalhador está a trabalhar		-	-	dh:Appointment	Vocabulário controlado de Piece Type	0..n
books No Of Pieces	books No Of Pieces	dh	BooksNoOfPieces	Número de peças para o qual o trabalhador, pago por peça, foi contratado		-	-	dh:Appointment	xsd:Integer	0..1
No of worker needed	No of worker needed	dh	#noOfWoker Needed	Identificador do número de trabalhadores necessário para o serviço		-	-	dh:Appointment	xsd: Integer	0..n
Has Report	Has Report	dh	HasReport	O relatório associado ao cliente, fornecido por um		-	-	dh:Employer	Literal	1

				trabalhador contratado, após a conclusão do encontro					
Has Rating	Work Rating	dh	Rating	Avaliação numa escala de 1 a 5, para o trabalhador	-	-	dh:Worker, dh:WorkerGroup	xsd:Decimal	1
Appointment holder	Appointment Holder	dh	HasAppointmentHolder	A pessoa que iniciou o agendamento do encontro	-	-	dh:Appointment	dh:Worker , dh:WorkerGroup	1
Job location	Job location	dh	jobLocation	Localização do serviço	-	-	dh:Appointment	dh:Address	1
Team Formation	Team formation	dh	team	Grupo de trabalhadores com habilidades distintas	-	-	dh:Appointment	dh:Worker	1

- *Object Property*

Tabela 10 - Matriz de restrições obsoleta - Avaliação presencial

Lista de Funcionalidades Obsoleta

Na Tabela 5 é apresentada a lista de funcionalidades obsoleta.

Nome	Descrição	Tipo de Utilizador
Login	Qualquer utilizador que aceda nas plataformas (*App* ou *Web*), necessita de um e-mail e *password*. No momento do *login*, o utilizador escolhe o tipo de perfil que pretende utilizar durante a sua sessão. Uma vez efetuado o registo de um utilizador, este não necessita de efetuar o *login*, sendo diretamente reencaminhado para o seu perfil.	Trabalhador e Cliente
Registo como cliente	O registo é efetuado para o utilizador possuir o acesso à plataforma. Um utilizador pode ter ambos os perfis (cliente e trabalhador), ou apenas um dos dois. No caso do utilizador escolher ambos, todos os campos necessários são apresentados. O Perfil do trabalhador não é público. As informações do cliente, só são apresentadas ao trabalhador, caso este seja contratado a realizar o serviço pelo cliente.	Cliente
Registo como trabalhador	A localização do trabalhador pode ser obtida através do reconhecimento, pelo sistema, da localização do trabalhador no momento do seu registo ou através dos dados fornecidos por este. O trabalhador, têm a hipótese de decidir que informação é publicada abertamente. O trabalhador é que decide, até que distância está disponível a responder a um serviço. Durante o registo na plataforma, é considerado a categoria/categorias do trabalhador, ou seja, se possui uma ou várias habilidades.	Trabalhador
Editar Perfil	O utilizador deve ter a possibilidade de alterar o seu perfil, a qualquer momento, após o seu registo.	Trabalhador e Cliente

Criar uma oferta de serviço	Um cliente cria uma oferta de serviço, especificando o tipo de serviço, duração, datas, avaliação ou experiência mínima, etc. O serviço pode requisitar um ou mais trabalhadores da mesma profissão. Neste caso, apenas grupos de trabalhadores podem responder à oferta. O sistema utilizará a localização do serviço, para localizar trabalhadores na área, apresentando-os ao cliente. O cliente deve ainda, definir um prazo no qual o serviço está suscetível a receber propostas.	Cliente
Lista de serviços passados publicados pelo cliente	Lista de serviços, publicados pelo cliente no passado e serviços abertos. Nesta lista deve estar visível a avaliação do trabalhador. Apenas o cliente pode visualizar esta lista.	Cliente
Página de perfil	Uma página que apresenta toda a informação do utilizador.	Trabalhador e Cliente
Serviço disponível para um trabalhador específico	A partir do momento que existem uma ou várias ofertas num serviço, e o cliente aprova o trabalhador/grupo, um alerta é emitido para o perfil do trabalhador, fornecendo a este a opção de aceitar ou recursar o serviço. Esta decisão deve ser tomada num pequeno intervalo de tempo. O trabalhador associado, deve possuir todos uma lista com todos os serviços a executar juntamente com os alertas temporários e lembretes.	Trabalhador
Aceitar a oferta de um serviço	Um trabalhador aceita uma oferta, baseando-se no tipo de oferta apresentada.	Trabalhador
Trabalhos entregues	Deve ser estabelecido um limite de trabalhos entregues a um único trabalhador, de modo, a possibilitar que outros trabalhadores que correspondam ao perfil do serviço, tenham a possibilidade de serem empregues.	Trabalhador

Oferta de serviço – Escolher o trabalhador	O cliente analisa os trabalhadores disponíveis para o serviço, escolhendo o que o irá executar.	Cliente
Alerta para informar trabalhadores *offline* de um novo serviço	Uma vez gerado um pedido de serviço, automaticamente é emitido um alerta para os trabalhadores.	Trabalhador
Logout	O utilizador termina a sua sessão na plataforma.	Trabalhador e Cliente
Avaliação do cliente	Quando um trabalhador termina o serviço, o seu empregador procede ao pagamento. Posto isto, o trabalhador tem a possibilidade de avaliar o cliente, entre 1-5 estrelas. Esta possibilidade, estará disponível durante um número de dias, sendo que, o trabalhador não consegue ver a avaliação do cliente antes de executar a sua avaliação.	Trabalhador
Avaliação do trabalhador	O cliente tem a possibilidade de avaliar o trabalhador, entre 1-5 estrelas. Esta possibilidade, estará disponível durante um número de dias, sendo que, o cliente não consegue ver a avaliação do trabalhador antes de executar a sua avaliação.	Cliente
Comentários sobre o trabalhador	Para além da avaliação, é possível ao cliente adicionar um comentário sobre o serviço prestado pelo trabalhador. Esta possibilidade também estará disponível durante um número de dias, sendo que, o cliente não consegue ver os comentários do trabalhador antes de comentar algo sobre o mesmo.	Cliente
Comentários sobre o cliente	Para além da avaliação, é possível ao trabalhador adicionar um comentário sobre a sua interação com cliente. Esta possibilidade também estará disponível durante um número de dias, sendo que, o trabalhador não consegue ver os comentários do cliente antes de comentar algo sobre mesmo.	Trabalhador

Criar um grupo de trabalhadores	Um grupo de trabalhadores é composto apenas por uma profissão. Este grupo é iniciado por um trabalhador com interesse em formar um grupo.	Trabalhador
Aceitar convite de grupo	Um trabalhador pode aceitar convites para ingressar um grupo.	Trabalhador
Enviar convite para pertencer a um grupo	O líder do grupo, envia um convite para um ou vários trabalhadores. Após os convites serem aceites, um alerta é emitido para o líder do grupo, informando-o que um novo trabalhador ingressou ao grupo.	Trabalhador
Deixar grupo	Um trabalhador tem a possibilidade de deixar o grupo. O líder não consegue deixar o grupo, para tal, necessita de apagar o grupo por completo ou mudar a liderança do mesmo.	Trabalhador
Apagar grupo	Possibilidade de apagar o grupo. Apenas o líder pode executar esta funcionalidade.	Trabalhador
Alterar líder do grupo	Apenas o líder pode alterar a liderança do grupo. Quando um novo líder é escolhido pelo antigo, este deve aceitar o cargo, caso contrário, a liderança não é alterada.	Trabalhador
Aceitar a liderança do grupo	Um membro do grupo deve aceitar o pedido para ser o líder. No seu perfil, são apresentadas todas as lideranças passadas e atuais.	Trabalhador
Verificar lista de trabalhadores pertencentes a um grupo	Um trabalhador pertencente a um grupo, pode verificar os restantes elementos do mesmo.	Trabalhador
SPARQL *query endpoint*	SPARQL *endpoint* (acesso por máquinas ou leitura *Web*), apenas para leitura, sem escrita.	n.d
Dataset dump* para *download	*Dump* completo do *dataset* pode ser obitdo em N3 & Turtle.	n.d
Alerta de um serviço não poder ser processado	No caso de um serviço não conseguir ser processado, é emitido um alerta para o cliente. Este deve escolher	Cliente

	entre estender o prazo para receber propostas de trabalhadores ou retirar o pedido de serviço da plataforma.	
Trabalhador indisponível	Um trabalhador pode estar indisponível em certo dia. Esse trabalhador não deve ser apresentado para serviços nessa data.	Trabalhador

Tabela 11 - Lista de funcionalidades obsoleta