Universidade de Aveiro
Departamento de Eletrónica, Telecomunicações e Informática
2018

Luís Pedro Martins de Almeida

Plataforma de Software para Sistemas PAYT em Cidades Inteligentes

Software Platform enabling PAYT Systems in Smart Cities

Universidade de Aveiro
2018

Departamento de Eletrónica,
Telecomunicações e Informática

Luís Pedro Martins de Almeida

Plataforma de Software para Sistemas PAYT em Cidades Inteligentes

Software Platform enabling PAYT Systems in Smart Cities

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Doutor João Paulo Silva Barraca, Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro, e do Doutor Diogo Gomes, Professor Auxiliar do Departamento de Electrónica e Telecomunicações e Informática da Universidade de Aveiro.

Esta dissertação foi financiada pelo programa LIFE Ambiente da União Europeia através do projeto LIFE 15 ENV/PT/000609.

Dedico este trabalho à minha família e amigos pelo incansável apoio durante este período tão importante na minha vida.

o júri / the jury

presidente / president

Professor Doutor André Ventura da Cruz Marnoto Zúquete
Professor Auxiliar da Universidade de Aveiro

vogais / examiners committee

Professora Doutora Ana Cristina Costa Aguiar
Professora Auxiliar do Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto

Professor Doutor João Paulo Silva Barraca
Professor Auxiliar da Universidade de Aveiro

agradecimentos /
acknowledgements

Gostaria de deixar um agradecimento a todos os que contribuíram para que finalizasse esta etapa tão importante na minha vida.
A todos os meus amigos que tiveram a paciência para lidar comigo nos momentos de maior stress e que sempre me apoiaram e ajudaram.
Aos meus pais que me deram a possibilidade de frequentar o ensino superior.
Ao Hélder e ao André, colegas de projeto, por em todos os momentos que tinha algum problema na realização deste trabalho se prestaram a ajudar-me.
Por fim, um especial agradecimento ao meu orientador, Professor Doutor João Paulo Barraca, por todo o conhecimento transmitido durante a realização desta dissertação bem como toda a sua disponibilidade.

Palavras Chave

Cidades Inteligentes, Internet das Coisas, Gestão de Resíduos, Sistemas PAYT, Dados Abertos

Resumo

À medida que o mundo avança, a quantidade de resíduos municipais está a aumentar a uma taxa superior à taxa de urbanização. É preciso, por isso, reduzir essa mesma quantidade produzida. Esta dissertação, numa primeira fase, irá apresentar o estado atual no que toca a soluções para redução de resíduos ligadas a cidades inteligentes. De seguida, é apresentada uma solução que demonstre a eficácia das metodologias PAYT("Pague-o-que-deita-fora") em ambiente de cidade inteligente. A solução desenvolvida contempla a existência de uma interface para os utilzadores para servir de incentivo à redução de produção de resíduos, bem como uma forma de tornar toda a informação recolhida em dados abertos, ainda que devidamente agregados e anonimizados.

Keywords

Smart Cities, Internet of Things, Waste Management, PAYT Systems, Open Data

Abstract

Following the world's progress, the amount of waste produced is growing even faster than the rate of urbanization. Therefore, it's necessary to reduce it. In a first stage, this dissertation will present the current solutions to reduce the amount of waste produced regarding smart cities environment. Then, a solution is presented, aiming to demonstrate the effectiveness of PAYT (Pay-As-You-Throw) methodologies in a smart environment. The implemented solution involves the existence of an user interface, to motivate the citizens to reduce the waste production, and a way of making all the information collected in open data, although properly aggregated and anonymised.

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Acrónimos

RSU	resíduos sólidos urbanos
PAYT	Pay-As-You-Throw
IoT	Internet of Things
EDP	Energias de Portugal
API	Application Programming Interface
PPP	princípio poluidor pagador
RFID	Radio Frequency Identification
AdRA	Águas da Região de Aveiro
SARA	Sistemas de Águas da Região de Aveiro
CNPD	Comissão Nacional de Proteção de Dados
STOMP	Simple Text Oriented Messaging Protocol
HTML	Hypertext Markup Language
HTML5	Hypertext Markup Language v5
CSS	Cascading Style Sheets
JS	Javascript
MVC	Modelo-Vista-Controlador
DOM	Document Object Model
XML	eXtensible Markup Language
JSON	JavaScript Object Notation
CSV	Comma-separated values
XLS	Microsoft Excel File
REST	Representation State Transfer
SOAP	Simple Object Access Protocol
HTTP	Hypertext Transfer Protocol
URI	Identificador Uniforme de Recurso
W3C	World Wide Web Consortium
TCP	Transmission Control Protocol
IP	Internet Protocol
PLC	Power-line Communication
JWT	JSON Web Tokens

capítulo 1

Introdução

À medida que o mundo avança, a quantidade de resíduos municipais está a aumentar a uma taxa superior à taxa de urbanização. Em 2012 era estimado que existissem cerca de 3 mil milhões de residentes a produzir 1,2kg de resíduos por pessoa por dia. Em 2025 prevê-se que estes números sejam ainda maiores, chegando aos 4,3 mil milhões de residentes atingindo os 1,42kg de resíduos produzidos por pessoa por dia [1].

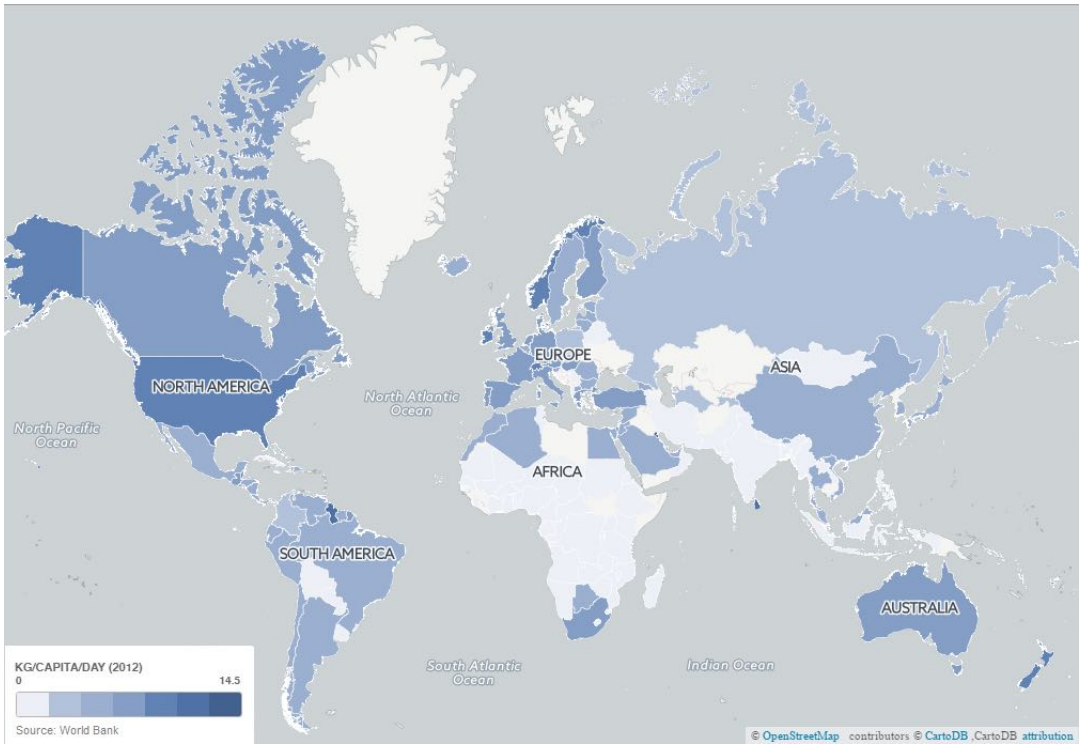

Figura 1.1: Quantidade de resíduos produzidos em cada país (2012) [2]

A gestão dos resíduos sólidos urbanos (RSU) é um dos serviços mais importantes prestado pelos municípios e em alguns casos até é o serviço com maior orçamento e

dos que mais trabalhadores emprega. Uma má gestão destes resíduos tem um impacto enorme, tanto na saúde como no ambiente, bem como acaba por se traduzir num maior custo para os municípios [3].

Tendo isto em consideração, torna-se importante baixar a produção de RSU, de modo a proporcionar uma melhor qualidade de vida aos residentes dos municípios. O uso de instrumentos económicos está a crescer como uma opção para criar incentivos com o objetivo de aumentar a recolha seletiva dos resíduos e de ajustar as tarifas cobradas aos residentes por este serviço.

Os sistemas Pay-As-You-Throw (PAYT) permitem estabelecer uma conexão entre a quantidade de resíduos produzidos e as tarifas pagas às autoridades locais. O PAYT atualmente é usado como um mecanismo em que o cidadão que usufrui do serviço de recolha de resíduos paga uma taxa de acordo com a quantidade produzida e o serviço que é usado. Assim, os sistemas PAYT promovem a redução dos resíduos produzidos, a reciclagem e facilitam a aplicação do princípio poluidor pagador. Consequentemente, o esforço em reduzir a produção de resíduos e aumentar a sua separação, por parte dos residentes ou comércios, é devidamente recompensado com a redução do valor a pagar pelo serviço de recolha.

1.1 MOTIVAÇÃO

Os sistemas PAYT já são usados no norte e centro da Europa, estando já implementados em países como Suíça, Alemanha, Dinamarca, Holanda, entre outros. No entanto, no sul da Europa ainda continuam a não ser muito populares.

A implementação das metodologias PAYT no sul da Europa irá permitir uma redução substancial na produção de RSU, tornando o ambiente bem mais agradável para os seus residentes. Na maioria dos casos, os sistemas PAYT foram implementados em contexto de recolha seletiva porta a porta com identificação do contentor ou saco. A implementação do novo sistema irá recorrer ao uso de métodos inteligentes de identificação dos produtores e de software desenvolvido para recolher dados de várias fontes e tornar esse processo transparente a todos os stakeholders.

A abertura dos dados recolhidos, para um ecossistema de cidades inteligentes, representa um enorme valor para eventuais melhorias nos seus serviços. Este e o facto da solução desenvolvida ser aplicada num cenário do mundo real torna-se um grande fator de motivação para a realização desta dissertação.

1.2 OBJETIVOS

Com o trabalho realizado nesta dissertação pretende-se obter uma solução que ajude a provar a eficácia das metodologias PAYT em alguns municípios, nomeadamente Aveiro, Condeixa, Lisboa, Larnaka, and Vrilissia.

Na mesma linha do projeto LIFE PAYT, a implementação desta solução deve contemplar um portal onde os cidadãos possam consultar os resíduos que estão a produzir para que estes se sintam motivados a reduzir a sua produção e a aumentar a separação dos mesmos.

Para além do portal para consulta dos cidadãos, no final desta dissertação também deverá existir uma integração com os portais de dados abertos dos municípios, caso estes existam, bem como um portal do mesmo tipo, mas exclusivo do projeto.

1.3 CONTRIBUIÇÕES

O trabalho desenvolvido no âmbito desta dissertação contribui com uma solução para implementar sistemas PAYT em ambiente de cidades inteligentes. Como parte do projeto LIFE-PAYT, esta solução foi testada e validada com dados e utilizadores reais. A solução proposta segue uma arquitetura modular com o objetivo de tornar a integração com outros municípios/cidades mais fácil de ser efetuada. A solução será posteriormente disponibilizada num repositório de código aberto, de forma a que potenciais interessados possam reutilizar ou adaptar ao seu caso de uso. Neste momento a solução desenvolvida já se encontra em funcionamento para os cidadãos da zona piloto em Aveiro.

Para além disso, a vertente dos dados abertos permite aos municípios serem mais transparentes, aumentando assim a confiança por parte dos seus habitantes. A disponibilização dos dados, previamente anonimizados, também benéfico para os investigadores já que irão ter acesso a dados que de outra forma não teriam.

1.4 ESTRUTURA DA DISSERTAÇÃO

Este documento está dividido em 6 capítulos dos quais, o capítulo 1, a Introdução, já foi apresentado. Os restantes capítulos estão organizados da seguinte maneira:

- **Capítulo 2:** apresentação do estado de arte. Neste capítulo alguns conceitos

referentes a cidades inteligentes, modelos de gestão de resíduos e dados abertos serão apresentados, juntamente com soluções já implementadas sobre esses mesmos conceitos.

- **Capítulo 3:** uma breve introdução ao projeto Life Payt, apresentando os objetivos principais e os benefícios para os stakeholders. Serão apresentados os principais requisitos do sistema bem como a arquitetura proposta para a solução seguida de uma explicação de cada componente.
- **Capítulo 4:** apresentação das tecnologias adotadas, juntamente com a razão dessas escolhas, seguidas da descrição da implementação da arquitetura proposta para a solução. É também apresentada a implementação de algumas das funcionalidades do sistema.
- **Capítulo 5:** apresentação e análise dos resultados obtidos dos testes realizados à solução implementada. A metodologia dos testes é descrita e os resultados analisados com foco nas métricas temporais que são cruciais nas interfaces com os utilizadores.
- **Capítulo 6:** conclusões finais sobre o trabalho realizado e apresentação de potenciais melhorias a serem realizados em trabalhos futuros.

CAPÍTULO 2

Estado de Arte

2.1 SMART CITY

Na literatura encontram-se muitas e diferentes definições para *smart city*. De um ponto de vista geral, uma cidade inteligente pode ser entendida como uma cidade que faz uso de tecnologias e análise de dados para tornar os seus processos mais eficientes, levar a um crescimento da sua economia e poder oferecer uma melhor qualidade de vida aos seus habitantes. Um dos objetivos das cidades inteligentes é recolher dados e torná-los disponíveis para outros usarem. Estes dados podem ser informação relativos ao clima, nível da água, trânsito, entre outros. A informação recolhida, maioritariamente por sensores estrategicamente posicionados nas cidades é bastante útil para programadores criarem aplicações. Após a recolha dos dados por parte dos sensores, esta tem de ser enviada para uma entidade central, conhecida como *broker*. Um *broker* serve como intermediário entre o produtor e consumidor dos dados, neste caso os sensores e aplicações respetivamente. O principal objetivo das cidades quando disponibilizam este tipo de dados publicamente é para incentivar os programadores a desenvolver *software* capaz de ajudar os seus cidadãos no seu quotidiano, melhorando a qualidade dos serviços fornecidos pelos municípios.

O aparecimento de novas tendências como a automação, *machine learning* e Internet of Things (IoT) está a fazer com que as cidades inteligentes sejam cada vez mais populares. Qualquer área de uma cidade pode ser incluída numa iniciativa *smart city*, havendo alguns exemplos clássicos deste tipo de iniciativas (fig. 2.1).

Uma das áreas que mais beneficia da incorporação de tecnologia e de análise de dados é a área dos transportes. Existem soluções que permitem o controlo do fluxo de trânsito com base na zona e na hora do dia, controlando os semáforos para evitar

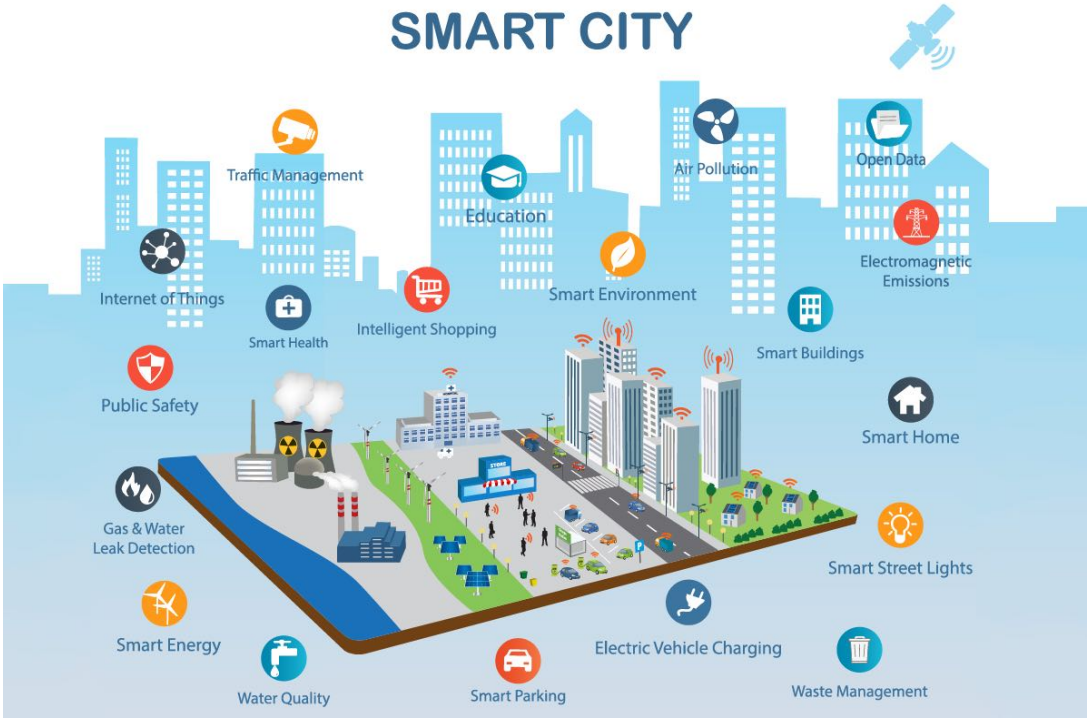

Figura 2.1: Áreas de intervenção de uma Smart City [4]

congestionamentos em algumas zonas críticas da cidade. Existe outro tipo de soluções na área dos transportes públicos que estão a começar a aparecer, permitindo a partilha de boleias de forma a satisfazer as necessidades de todos os cidadãos.

Outra das áreas bastante afetadas por estas iniciativas é a parte energética e ambiental. A presença de sensores nas ruas pode ser bastante vantajosa para obter uma maior eficiência e conservação energética. Com o uso destes sensores e uma integração com os candeeiros públicos, estes podem ajustar a intensidade luminosa consoante a presença ou não de carros ou peões na rua. As redes energéticas inteligentes podem ser usadas para aperfeiçoar operações, manutenção e planeamento, mas também para oferecer a possibilidade de fornecimento de potência energética adequada à necessidade do cliente. Em Portugal, a Energias de Portugal (EDP) (InovGrid) já possui uma destas redes inteligentes [5]. Nos dias de hoje, com toda a poluição e alterações climáticas, existe uma grande preocupação com as questões ambientais por parte das cidades. Cada vez mais estas alterações afetam a qualidade de vida dos habitantes, o que faz com que haja a necessidade de melhorar a qualidade do ar e da água, através do uso de sensores para efetuar medições constantes aos parâmetros da água, garantindo assim a sua qualidade.

A área da gestão de resíduos também beneficia deste tipo de tecnologias, tirando partido da ligação dos caixotes do lixo com a entidade que gere a recolha dos resíduos.

Com esta ligação é possível a troca de informação relativa à ocupação do contentor que é útil para otimizar a recolha dos resíduos numa determinada área melhorando a eficiência da frota das entidades de recolha. Outro sistema que pode tirar proveito da existência de mais tecnologia é o Pay-As-You-Throw (PAYT), o qual vou explicar mais à frente.

2.1.1 SANTANDER – ESPANHA

Em 2010, a cidade Santander foi escolhida para se tornar o primeiro campo de testes de sensores inteligentes. Nos últimos quatro anos, mais de 12.000 sensores foram instalados em toda a cidade, medindo uma grande variedade de fatores, incluindo a poluição do ar, os níveis de trânsito, o número de lugares de estacionamento e muito mais [6]. Esta informação alimenta em tempo real os servidores centrais que, de seguida, fornecem informações para os funcionários da cidade. Por exemplo, as autoridades locais podem decidir com que frequência enviam veículos de recolha de resíduos com base nos sensores presentes nos caixotes que permitem identificar quão cheias estão as unidades. Os dados recolhidos pelos sensores também são fornecidos ao público para que, por exemplo, o setor privado possa criar aplicações que permitam aos cidadãos fazer bom uso dos dados.

2.1.2 CASCAIS – SMARTBIN

Em março de 2015, Cascais decidiu dar um grande passo em frente no que toca à gestão de resíduos, implementando a tecnologia SmartBin nos seus contentores subterrâneos. A partir desta data, a região começou a monitorizar os contentores através dos sensores de nível de preenchimento. O município, que serve cerca de 200.000 cidadãos, tornou-se uma das mais defensoras regiões de cidades inteligentes, com vários projetos inovadores e vários prémios ganhos [7].

O objetivo de Cascais ao adotarem esta solução era arranjar uma forma de otimizar as suas rotas de recolha de resíduos, reduzir os custos do município ao mesmo tempo que melhoravam o serviço e, por fim, manter uma cidade limpa e os seus cidadãos felizes ao conseguir evitar o enchimento excessivo dos contentores [8].

Através da cooperação entre a Sotkon Waste Systems e a entidade responsável pela frota da empresa Cascais Ambiente, a solução SmartBin foi facilmente integrada no município de Cascais. Foram colocados sensores nos contentores subterrâneos localizados por todo o município. Os dados recolhidos pelos sensores são enviados através de redes

de móveis para o serviço SmartBin Live, onde o centro de controlo da Cascais Ambiente monitoriza o nível de preenchimento dos seus 400 contentores e planeia rotas otimizadas para a recolha dos resíduos. Usando a API disponibilizada pela solução, as informações sobre as rotas são diretamente enviadas para os camiões de recolha, permitindo aos condutores que apenas se foquem nas rotas com os contentores que realmente necessitam de recolha.

2.2 PAY-AS-YOU-THROW (PAYT)

Nos dias de hoje, os (RSU) são um dos principais problemas ambientais a nível mundial, verificando-se um enorme crescimento na sua produção. Embora a separação destes materiais por parte das pessoas seja cada vez maior, ainda não é suficientemente grande para que este deixe de ser um problema.

Atualmente, em Portugal as tarifas cobradas aos cidadãos para recolha e tratamento de resíduos não estão diretamente relacionadas com a sua produção. Esta pode ser uma das causas para as estratégias adotadas para redução da produção de RSU não obterem os efeitos desejados.

Uma forma mais eficaz de se reduzir a produção de resíduos pode passar pela aplicação de incentivos ou penalizações financeiras, recorrendo à implementação de um novo sistema que permita o cálculo de uma tarifa mais justa, com base nas quantidades de resíduos produzidos. Este sistema já existe e denomina-se PAYT, ou seja, "paga o que produzes".

2.2.1 MODELOS DE SISTEMAS PAYT

Segundo Marta e Hanf [9], um sistema PAYT assenta em duas diretrizes de planeamento ambiental, o princípio poluidor pagador (PPP) e o conceito de responsabilidade partilhada. De acordo com o PPP, os cidadãos devem pagar os custos da sua parte de responsabilidade no que toca à produção de resíduos.

Os sistemas PAYT são aplicados sob forma de um incentivo financeiro, já que com este sistema o cidadão apenas paga a porção de resíduos indiferenciados que produz, enquanto que a deposição seletiva não entra para o cálculo da tarifa. Esta forma de cálculo pode-se traduzir num incentivo, mas também pode servir como penalização já que quanto maior for a quantidade de resíduos produzida, mais o cidadão terá de pagar. Com a aplicação destes sistemas espera-se promover a separação na origem e aumentar

as taxas de recolha seletiva.

Vários países já têm implementados sistemas PAYT. Como existem vários modelos deste sistema, a sua implementação pode adotar várias formas. Estes modelos dependem da forma como a identificação do produtor de resíduos é feita bem como da medição dos mesmos resíduos. Na figura 2.2 pode-se verificar as variantes que o PAYT pode assumir.

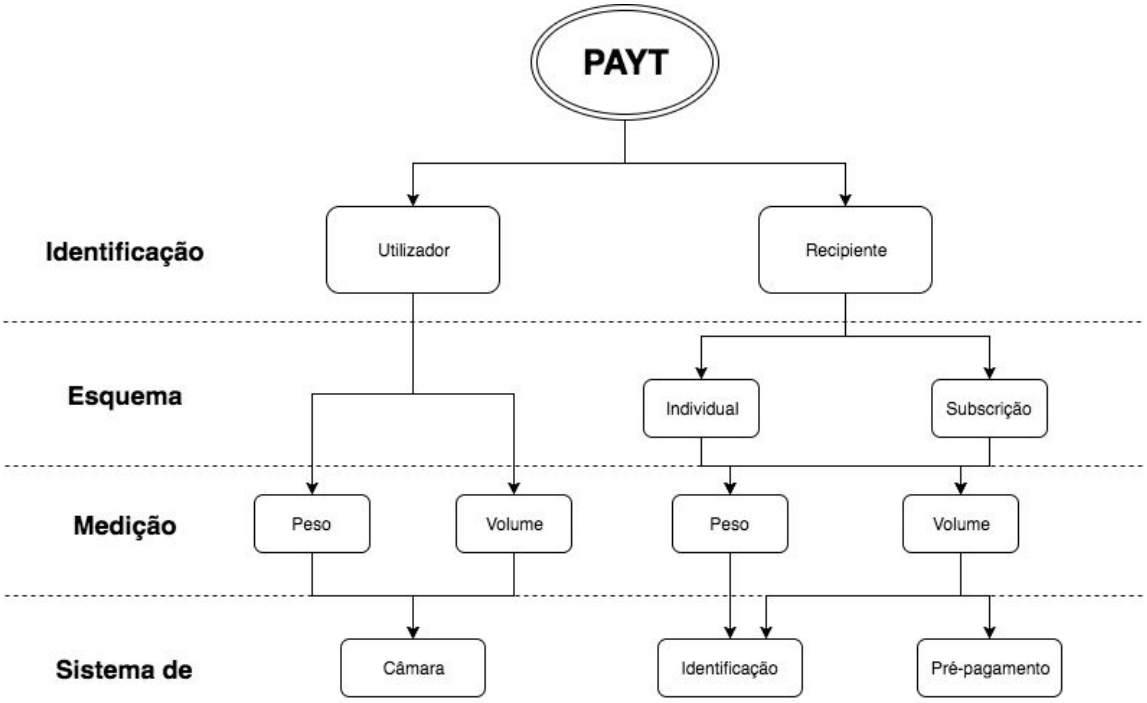

Figura 2.2: Modelos de implementação de um sistema PAYT (Adptado de Reichenbach [10], 2008).

Inicialmente pode-se dividir as variantes em dois grandes grupos, com base na identificação do utilizador ou com base na identificação do contentor.

Identificação do utilizador / Sistema de câmara

Os utilizadores possuem um método de identificação, podendo este ser um cartão RFID ou uma chave eletrónica, que permite que a utilização do serviço seja registada no momento em que procede à deposição dos resíduos.

Medição por volume - o contentor tem uma tampa que abre e permite que o utilizador deposite os resíduos numa câmara, com um volume máximo por cada utilização, após ter-se identificado usando o cartão/chave eletrónica.

Medição por peso - o contentor tem uma balança que regista o peso que cada utilizador deposita em cada utilização.

Identificação do recipiente

Nesta variante é feita a identificação do recipiente, que regista a utilização do produtor por subscrição, atribuição ou compra. O recipiente pode ser um contentor ou mesmo um saco (identificado por uma etiqueta).

Esquema individual - é atribuído um recipiente a cada utilizador ou grupo de utilizadores com um volume conhecido que é identificado com um chip ou etiqueta. A identificação é feita com recurso a equipamento instalado no camião de recolha ou através de um leitor portátil na posse do operador. Neste sistema é assumido que o contentor está cheio a cada recolha, sendo usado o seu volume para o cálculo do valor a cobrar. Para maximizar o uso do contentor, o utilizador pode decidir quando o coloca para remoção.

Esquema por subscrição - neste caso a recolha dos recipientes é feita regularmente de acordo com o estabelecido com o produtor. O produtor pode estabelecer a frequência com que os recipientes são recolhidos e os seus volumes. A recolha é sempre feita e cobrada mesmo que os recipientes estejam vazios. Com isto pretende-se que cada produtor subscreva os recipientes de acordo com as suas necessidades e paguem o que realmente produzem.

Identificação e pesagem do recipiente - o valor cobrado é calculado através do peso do recipiente que é identificado através de um chip ou etiqueta. O recipiente é pesado através de um mecanismo presente no camião de recolha.

Pré-pagamento - neste caso o produtor paga a taxa no ato de compra do saco destinado à deposição dos resíduos. A entidade responsável pela recolha apenas aceita sacos deste tipo. Estes sacos são distribuídos pelas câmaras municipal ou por colaboradores e são diferentes para cada tipo de resíduos.

2.2.2 SOLUÇÕES PAYT

Os sistemas PAYT já são bastante usados no norte e centro da Europa, nomeadamente na Suíça, Áustria, Alemanha, Itália, Dinamarca e Holanda. Nos Estados Unidos o PAYT já é usado desde o final dos anos 90, sendo que atualmente, mais de 7000 municípios já têm esquemas PAYT implementados. Nos municípios com maiores dimensões predominam os esquemas de pagamento por recolha de contentor sem subscrição enquanto que nos municípios mais pequenos são mais usados os esquemas de pagamento por saco ou por contentor com sistemas de identificação dos produtores.

Mais recentemente surgiram novas implementações, nomeadamente em vários municípios de Barcelona e também no centro histórico de Guimarães.

Em Janeiro de 2010, o município de Canet de Mar [11], que possui cerca de 14.000 habitantes, introduziu um sistema PAYT baseado nos contentores, apenas para os grandes produtores de resíduos. Dos 700 comércios existentes na região, 100 satisfaziam os requisitos para serem classificados como grandes produtores. Anteriormente ao sistema ser implementado era cobrada uma taxa fixa. Mas com o aparecimento do PAYT, a fatura passou a ser dividida numa pequena taxa fixa e uma variável baseada no volume do contentor e no número de recolhas. As recolhas são registadas automaticamente usando uma etiqueta RFID em cada contentor que posteriormente é registada pelos leitores presentes nos veículos de recolha.

Em dezembro de 2015 arrancou, no Centro Histórico de Guimarães [12], um projeto inovador cujo objetivo era implementar um sistema PAYT para que os cidadãos apenas pagassem pelos resíduos que produzem. O projeto é o resultado do trabalho desenvolvido entre a Câmara Municipal, a Vitrus Ambiente e a Resinorte. Foram distribuídos mini ecopontos por cada habitação para a separação dos resíduos recicláveis e cada habitante ou comerciante terá de usar sacos adequados à deposição do lixo doméstico ou indiferenciado. Os sacos são comprados nas instalações da Vitrus ou na viatura que irá efetuar a recolha. A viatura de recolha efetua passagens por todas as ruas do centro histórico várias vezes por dia para recolher os sacos, enquanto que os resíduos recicláveis devem ser na mesma colocados nos ecopontos. O valor a pagar pelos resíduos produzidos reflete a quantidade de sacos usados, ou seja, quantos menos usar, menos irá pagar. Este sistema assemelha-se ao já praticado na Bélgica.

2.3 TECNOLOGIAS RELEVANTES EM AMBIENTES DE CIDADES INTELIGENTES

De acordo com o W3C, um serviço web é um meio de obter interoperabilidade na interação através da rede entre diferentes aplicações de software, independentemente da plataforma ou *framework* onde estão a correr [13]. Os serviços web são um ponto fundamental de integração para aplicações pertencentes a diferentes plataformas, linguagens e sistemas, já que estes serviços são baseados num conjunto de normas que os tornam independentes das tecnologias usadas para a sua conceção [14].

Na figura 2.3 está representado o processo de interação com um serviço web, no qual existem duas entidades, o fornecedor do serviço e o seu consumidor. Este processo

decorre em 4 fases distintas: (1) o fornecedor e o consumidor tomam conhecimento da existência um do outro (ou pelo menos um toma conhecimento da existência do outro); (2) as duas entidades efetuam, de alguma maneira, um acordo sobre a descrição do serviço e da forma como interação vai ser feita entre ambos os agentes; (3) as implementações acordadas na fase 2 são desenvolvidas em cada entidade; (4) os agentes fornecedor e consumidor trocam mensagens que na realidade representam as interações entre ambas as partes.

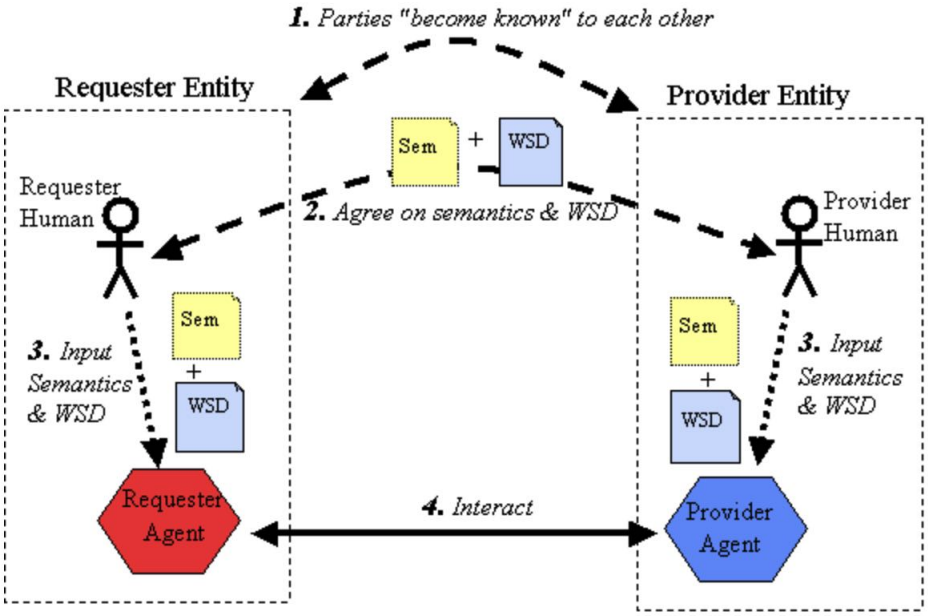

Figura 2.3: Processo de interação com um serviço web [13]

Nas subsecções 2.3.1 e 2.3.2 vão ser descritas as arquiteturas mais comuns para a implementação de serviços web, a arquitetura baseada em SOAP e a baseada em REST respetivamente.

2.3.1 Simple Object Access Protocol (SOAP)

SOAP é um protocolo de comunicação usado para troca de informação num ambiente descentralizado e distribuído [15]. A arquitetura de serviço web baseado em SOAP define 3 entidades: o fornecedor do serviço, o consumidor e o registo dos serviços [16]. O fornecedor do serviço é uma entidade presente na rede que recebe e executa o pedido do consumidor. O consumidor do serviço é uma aplicação ou outro qualquer módulo de software que necessita do serviço. O registo dos serviços é um diretório presente na rede, onde se encontra a informação referente a todos os serviços disponíveis. O consumidor encontra a descrição de cada serviço, publicada pelo fornecedor neste registo, e usa-a

para começar a interagir com o serviço escolhido.

A comunicação entre as várias entidades envolvidas é baseada em XML e no protocolo SOAP. A estrutura das mensagens trocadas, designadas por envelopes SOAP, pode ser visualizada na figura 2.4, sendo constituída por dois blocos principais, o cabeçalho e o corpo. O cabeçalho (campo opcional) é utilizado para fornecer informação adicional específica para o uso do serviço, tal como dados de autenticação, definições de transação, etc. O corpo é um campo obrigatório e contém a informação destinada ao recetor da mensagem, como por exemplo o método a ser executado e os seus argumentos, ou então o valor de retorno de um certo método.

Figura 2.4: Estrutura de uma mensagem SOAP [17]

2.3.2 Representation State Transfer (REST)

Alguns anos após a criação dos serviços web baseados em SOAP, foi desenvolvida uma alternativa, a criação dos serviços baseados numa arquitetura REST [14]. Esta arquitetura, introduzida por Roy Fielding em 2000, resume-se a uma arquitetura cliente-servidor (fig. 2.5) em que o cliente envia um pedido para o servidor, este processa-o e devolve a resposta a esse pedido.

Existem alguns princípios para a criação de serviços web REST, sendo eles os seguintes: um sistema universal de endereçamento e uma interface uniforme e sem conservação de estado. Quanto ao sistema de endereçamento, a arquitetura REST modela os conjuntos de dados como recursos e marca-os com um URI. É usada uma interface uniforme, ou seja, é utilizado um conjunto fixo de métodos Hypertext Transfer

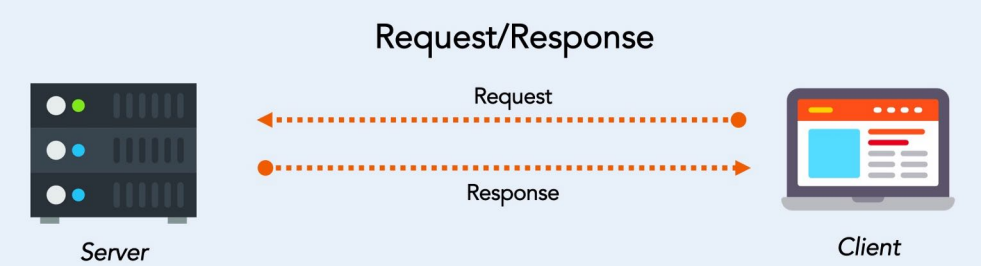

Figura 2.5: Processo de interação com um serviço web baseado numa arquitetura REST [18]

Protocol (HTTP) para aceder e/ou modificar os recursos. Cada transação efetuada é considerada independente da anterior já que toda a informação necessária para o processamento do pedido está contida apenas nesse pedido. Como a informação relativa à sessão do cliente não é mantida no lado do servidor, as respostas do servidor também são independentes. Estes princípios fazem com que a arquitetura REST seja leve e simples [16].

2.3.3 COMPARAÇÃO ENTRE SOAP E REST

Apesar das vantagens e desvantagens que estas duas arquiteturas possuem, ambas são adequadas para construção de sistemas distribuídos. A grande vantagem dos serviços web baseados numa arquitetura SOAP é o facto de não dependerem de um protocolo específico enquanto que a arquitetura REST implica o uso do protocolo HTTP.

Outro ponto forte do SOAP é a forma de como lida com os erros [19] pois a sua especificação define um padrão para descrever os erros através do elemento SOAP-Fault. A presença deste elemento no corpo da mensagem de resposta indica que ocorreu um erro que vem descrito nos subelementos nele contidos. São eles o faultcode, elemento obrigatório, que indica especificamente o erro ocorrido. Para além deste elemento, também estão presentes os elementos *faultstring* e *faultactor*. O conteúdo do primeiro é uma mensagem possível de ser entendida por um humano, que descreve o erro ocorrido enquanto que o segundo serve para fornecer informação de onde é que ocorreu a falha. Ao contrário da arquitetura SOAP, a arquitetura REST não tem um padrão bem definido para retornar lidar com erros, sendo responsabilidade do programador o modo como estes são processados e reportados. As 3 formas mais usuais para informar que um erro ocorreu são as seguintes: através do uso dos códigos de erro definidos para o protocolo HTTP (ex: 2xx – *Success*, 4xx – *Client Error*, 5xx – *Server Error*), através de um campo com a descrição do erro no corpo da mensagem de retorno, e uma junção das duas anteriores.

No que toca à complexidade de desenvolvimento, os serviços com arquitetura REST são consideravelmente mais simples tanto para o fornecedor do serviço bem como para o consumidor [14]. Quanto ao desempenho, a arquitetura SOAP é bastante inferior já que os seus pacotes introduzem uma carga na rede mais elevada que a arquitetura REST, o que implica um uso bastante maior de largura de banda para o envio e receção de mensagens bem como um maior consumo de recursos para codificar e descodificar as mensagens XML [16].

2.3.4 WEBSOCKETS

No passado, a criação de aplicações web que necessitam de comunicação bidirecional entre cliente e servidor (ex. serviços de mensagens instantâneas ou jogos) leva ao abuso do protocolo HTTP para verificar junto do servidor se existem atualizações que tenham de ser reportadas ao cliente [20]. Esta constante verificação traz consigo alguns problemas:

- o servidor é forçado a usar diferentes ligações TCP para cada cliente, uma para enviar a informação para o cliente e outra para cada mensagem nova que é recebida;
- existe uma grande sobrecarga no canal de comunicação já que cada mensagem cliente-servidor tem um cabeçalho HTTP;
- o cliente é forçado a manter um mapeamento entre cada ligação de saída e a respetiva ligação de entrada para obter as respostas a cada pedido.

Para poder atenuar os problemas anteriormente referidos foi então criado o protocolo Websocket, que permite usar uma única ligação Transmission Control Protocol (TCP) sobre HTTP para tráfego nas duas direções [20].

O protocolo Websocket foi desenvolvido como parte da iniciativa HTML5 para facilitar as comunicações sobre canais TCP e não se baseia no modelo pedido/resposta como os serviços SOAP e RESTful nem no modelo publicar/subscrever [21]. O cliente estabelece uma ligação Websocket através de um processo conhecido como handshake. Este processo inicia-se com o cliente a enviar um pedido HTTP para o servidor. Um cabeçalho Upgrade é incluído neste pedido (fig. 2.6) informando o servidor que o cliente pretende estabelecer uma ligação Websocket.

Se o servidor suportar este protocolo, aceita este pedido incluindo na resposta o mesmo cabeçalho Upgrade (fig. 2.7).

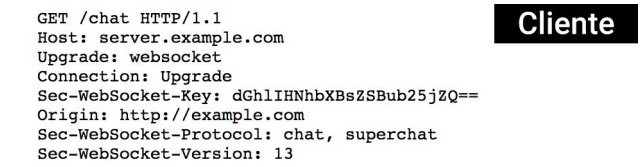

Figura 2.6: Cabeçalhos websocket handshake (cliente)

Figura 2.7: Cabeçalhos websocket handshake (servidor)

Depois deste processo de handshake terminar, a ligação HTTP inicial é substituída por uma ligação Websocket que utiliza a ligação TCP/IP já existente. Neste momento tanto o cliente como o servidor podem começar a enviar informação.

Com websockets é possível transferir a quantidade de informação desejada sem que isso traga a sobrecarga adicional associada aos tradicionais pedidos HTTP. A informação é transferida como uma mensagem, que consiste num ou mais frames que contêm a informação a ser transferida.

2.3.5 MICROSERVIÇOS

Até aos dias de hoje, a forma mais comum para desenvolver aplicações de software era através de uma abordagem monolítica onde um módulo, na maioria dos casos, é responsável por todo o funcionamento do sistema. Para pequenos projetos esta continua a ser uma boa abordagem. Para outro tipo de projetos onde a escalabilidade, flexibilidade e rapidez de desenvolvimento sejam fatores críticos, uma abordagem monolítica pode-se tornar numa barreira para atingir o sucesso [22]. À medida que a aplicação cresce e mais funcionalidades são implementadas, alterações e/ou correções tornam-se mais difíceis de fazer, atrasando o ciclo de lançamento da aplicação.

Para solucionar os problemas de abordagem monolíticas foi introduzida a arquitetura baseada em microserviços. Aplicações construídas com base nesta arquitetura são compostas por um ou mais serviços que podem ser lançados de forma independente. Cada um desses microserviços estão focados apenas em desempenhar uma única função [23].

Uma arquitetura de microserviços possui um conjunto de características [23] que a definem e que serão abordadas nesta secção.

Pequenos e focados

Os microserviços devem-se focar apenas numa tarefa logo, devem ser pequenos. Isto também faz com que seja mais fácil manter ou mesmo reformular na totalidade o microserviço. Este também deve ser tratado como se fosse uma aplicação ou um produto, isto é, também deve ter associado a si um repositório para gestão de código base bem como um ciclo de lançamento.

Desacoplados

Uma característica essencial dos microserviços é eles serem desacoplados. Cada um dos serviços deve poder ser lançado sem este depender de outros. Este desacoplamento faz com que os lançamentos de novas funcionalidades sejam mais rápidos e frequentes.

Linguagem neutra

Cada microserviço deve ser desenvolvido usando a linguagem de programação que seja mais adequada à tarefa que lhe está associada. A comunicação entre serviços deve ser feita através de protocolos padrão, que não dependam da linguagem de programação, tais como o HTTP ou protocolos baseados em mensagens.

Contexto limitado

Um determinado microserviço não deve "saber" nada sobre a implementação de outros microserviços que o rodeiam. Um serviço deve apenas ter conhecimento de como comunica com outros microserviços.

Como foi dito anteriormente, os microserviços apareceram para colmatar alguns dos pontos fracos de arquiteturas monolíticas, trazendo consigo vários benefícios. Graças a uma divisão da aplicação em vários serviços, cada um deles pode ser alterado e lançado outra vez sem que isto comprometa a integridade da aplicação. Um dos grandes benefícios está relacionado com a tolerância a falhas, isto é, se um serviço falhar os outros continuam a trabalhar.

Como a aplicação está mais dividida e organizada por funcionalidades é mais fácil para fazer alterações bem como acrescentar mais funcionalidades ao sistema. Para além disso o facto de cada serviço ser independente permite que estejam espalhados por vários servidores e/ou datacenters. Este estilo de arquitetura também traz benefícios ao nível da segurança já que as diferentes partes da aplicação estão isoladas. Um problema de segurança pode acontecer numa secção sem que isso afete outras áreas do projeto [24].

2.4 SOLUÇÕES EXISTENTES – DASHBOARDS

Hoje em dia é muito importante uma boa relação entre o consumidor e a entidade prestadora do serviço. Para melhorar essa relação, as empresas cada vez mais apostam em disponibilizar aos cidadãos portais para consulta dos serviços contratados. Na ótica do consumidor, torna-se bastante útil consultar os últimos consumos, assim como fazer comparações com outros meses para poder ter uma perceção da evolução dos seus consumos. Para além disso, em muitos destes portais já é possível alterar condições dos contratos, dados de faturação e muitas outras opções. De seguida irei dar alguns exemplos destes portais especificando algumas das suas funcionalidades.

Bee2Waste – Gestão de Resíduos Sólidos Urbanos

Para os administradores dos serviços municipais também se torna bastante útil ter interfaces para gerir cada serviço de uma forma mais eficiente. É neste âmbito que a Compta decidiu criar um leque de soluções para diferentes áreas de gestão de uma cidade. Relativamente à gestão de resíduos construiu a solução Bee2Waste, que já serve mais de 40 cidades da Europa e América do Sul. Esta solução é capaz de interagir com uma grande variedade de sensores IoT e métodos de recolha de resíduos, para que os métodos mais eficientes possam ser implementados e adaptados às necessidades específicas de cada cidade.

Figura 2.8: Componentes Bee2Waste [25]

O software foi criado para poder interagir com todos os elementos operacionais no processo de recolha (figura 2.8). No que toca às rotas de efetuadas pelos veículos de recolha, esta solução sugere ao operador a melhor rota consoante os diferentes fatores selecionados (hora, tipo de resíduos, veículos usados, etc.). Através da análise dos dados

referentes ao passado, é possível ao operador planear novas rotas, por exemplo, planear uma rota que recolha todos os contentores que estejam 30% preenchidos nas próximas 12 horas.

Com recurso a esta solução, é possível manter um catálogo com todas as informações sobre os pontos de recolha (tipo de contentor, tipo de resíduos, etc.), que podem ser adicionadas ao sistema de forma manual ou automática através de um leitor RFID. É também possível obter vários indicadores de desempenho, automaticamente atualizados, tais como valores totais de resíduos recolhidos (geral e por tipo), incidentes registados, custos de recolha por tonelada de resíduos recolhidos ou por habitante, entre outros.

A integração com sistemas externos é feita através da API onde os dados de entrada e de saídas podem ser trocados com outros sistemas que fazem parte do processo de recolha. Este sistema é capaz de interagir com uma grande variedade de dispositivos e sensores independentemente da tecnologia de comunicação que usam como LoRa ou SigFox.

EDP Online

O Grupo EDP está entre os grandes operadores europeus do sector da energia, sendo um dos maiores operadores energéticos da Península Ibérica e o maior grupo industrial português. Para quem é cliente EDP, além da fatura mensal, existe também um portal que permite ao consumidor realizar as mais diversas tarefas.

No EDP Online é possível fazer a gestão de todos os contratos relativos a casas e/ou negócios que possua, de uma forma simples e rápida. Neste portal, o consumidor pode comunicar leituras, consultar faturas e pagamentos.

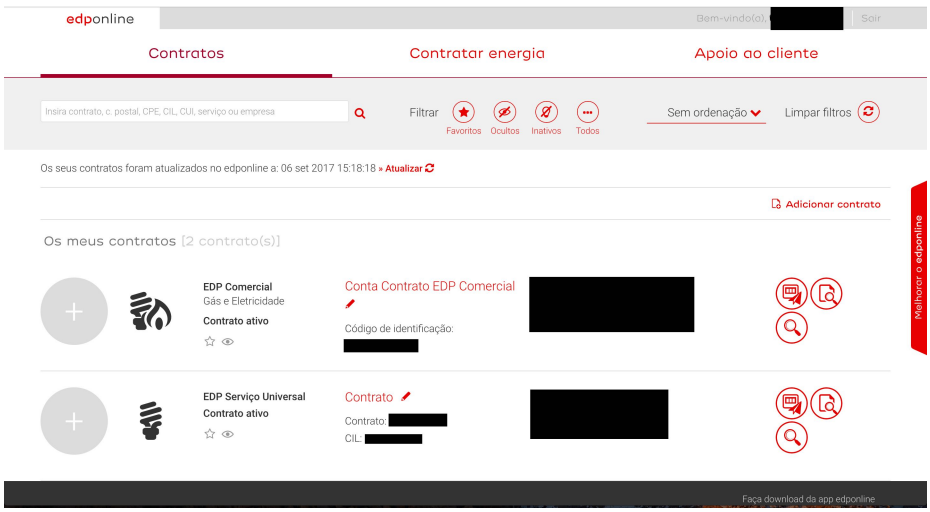

Figura 2.9: Vista geral sobre os contratos ativos

Na vista geral do portal (fig. 2.9) é exibida uma lista dos contratos associados ao consumidor, bem como informações gerais sobre os mesmos. É possível pesquisar contratos através de vários parâmetros. Nesta vista o cliente também pode executar várias ações sobre cada contrato, como por exemplo comunicar leitura, verificar a última fatura ou dirigir-se para a vista sobre os detalhes do contrato. Todas estas ações podem ser executadas através de uma simples clique.

No que toca à gestão dos contratos, o consumidor tem a possibilidade de celebrar novos contratos, alterar os dados contratuais, como a potência ou periodicidade da fatura, e personalizar os nomes dos contratos para facilitar a distinção. Também é possível ao cliente efetuar pedidos de apoio e de informação bem como aderir à fatura eletrónica e débito direto.

Se o consumidor pretender também pode visualizar os seus consumos sob a forma de gráfico de barras (fig. 2.10). Com este tipo de visualização o cliente tem uma ideia mais clara de como os seus consumos estão a variar ao longo do tempo.

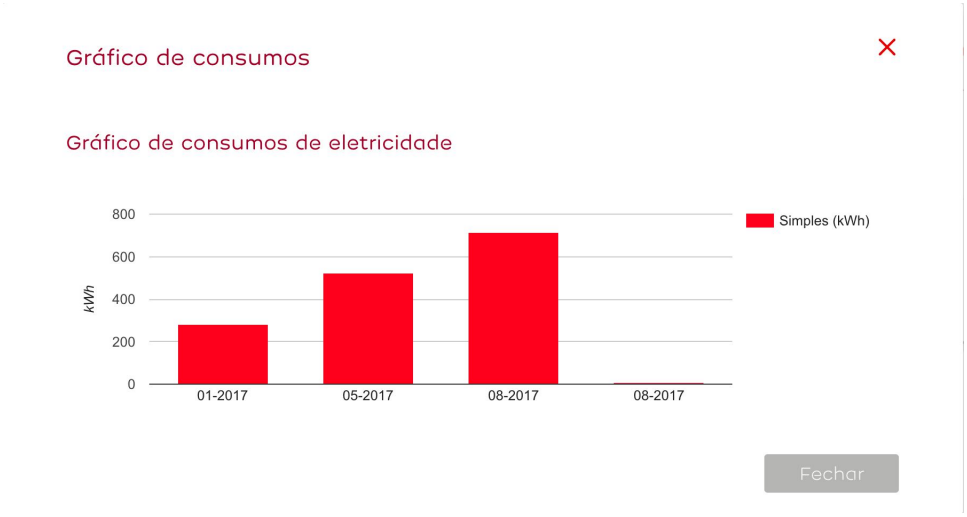

Figura 2.10: Vista sobre os consumos de eletricidade

Tal como acontece com os consumos, o cliente pode consultar as suas faturas de forma detalhada, bem como o estado da mesma (paga ou ainda a aguardar pagamento). Também é possível alterar tanto o tipo de periodicidade da fatura, como a morada de envio da mesma.

Para concluir, esta é uma plataforma que serve para aproximar mais os clientes da entidade prestadora do serviço de eletricidade. Com ela, os clientes conseguem visualizar tudo o que está relacionado com os seus contratos. Graças à sua interface elegante, torna bastante simples a navegação e é possível obter-se a informação pretendida em poucos cliques, aumentando a satisfação do cliente.

Para além do serviço EDP Online, o grupo EDP desenvolveu uma solução que permite aos consumidores gerirem o seu consumo energético remotamente, a EDP RE:DY [26]. Esta solução está dependente de dois tipos de equipamentos: a re:dy box e os periféricos re:dy. A re:dy box, o equipamento central desta solução recebe informação diretamente do contador inteligente da EDP Distribuição através do protocolo sobre a rede elétrica (PLC) e comunica com router via TCP/IP através do cabo fornecido. Para além disso integra todos os equipamentos ligados aos periféricos re:dy (fig. 2.11) utilizando o protocolos de comunicação ZigBee e processa a informação recebida. Toda esta informação é apresentada ao utilizador na aplicação móvel ou no website do serviço. Através destes dois pontos de gestão o utilizador pode consultar os seus consumos, controlar remotamente os equipamentos de sua casa ou até mesmo ser alertado em caso de anomalias ou consumos inesperados.

Figura 2.11: Integração dos equipamentos no serviço EDP RE:DY [26]

AdRAnet

A Águas da Região de Aveiro (AdRA) é a entidade responsável pelos serviços de água e saneamento relativos ao Sistemas de Águas da Região de Aveiro (SARA). Para facilitar a interação com os clientes a AdRA lançou o AdRAnet, um portal integrado no seu *website* que permite a consulta da sua conta cliente de uma forma cómoda e simples.

Com este novo serviço disponibilizado pela Águas da Região de Aveiro o consumidor pode acompanhar os seus consumos, consultar as suas faturas, gerir os seus dados de cliente, comunicar leituras e anomalias (fig. 2.12).

Uma das funcionalidades também presentes neste portal é a possibilidade de efetuar pedidos de assistência, onde o cliente seleciona o tipo de assistência que pretende, escolhe a data e horário pretendidos e se necessitar pode prestar informações complementares.

Inicio Periodo	Fim Periodo	Consumo m3	Nº Dias	Tipo Consumo
2017-02-02	2017-04-04	11	62	R
2017-04-05	2017-06-02	13	59	R
2017-06-03	2017-08-02	13	61	R
2017-08-03	2017-08-04	0	2	E
2017-08-05	2017-09-04	6	31	E

Figura 2.12: Vista sobre a faturação e consumos

Portal de Transparência Municipal

De forma a aumentar a transparência entre a administração local e os cidadãos, o governo português decidiu avançar com a criação do Portal de Transparência Municipal. Nele são apresentados vários indicadores de desempenho relativos à gestão dos 308 municípios portugueses. O portal faz uso do Pentaho [1], uma plataforma de fonte aberta ligada ao *business intelligence*, e da biblioteca *JavaScript FusionCharts* que é responsável pela representação gráfica dos indicadores. Todos os dados apresentados no portal estão também disponíveis na plataforma Dados.gov[2], a plataforma de dados abertos do governo português. Os indicadores disponíveis no portal foram organizados em 6 grupos: gestão financeira, gestão administrativa, decisões fiscais do município, dinâmica económica do município, serviços municipais e participação eleitoral autárquica.

[1] https://www.hitachivantara.com/go/pentaho.html
[2] www.dados.gov.pt

Figura 2.13: Grupo de indicadores referentes ao município de Aveiro [27]

Para além da consulta por município (fig. 2.13) também é possível a consulta por indicador onde o cidadão seleciona um indicador e um município e obtém a informação sobre esse indicador referente ao município selecionado.

Outra das funcionalidades deste portal é a possibilidade de fazer comparações entre municípios ou mesmo com médias, máximos e mínimos nacionais como se pode ver na fig. 2.14.

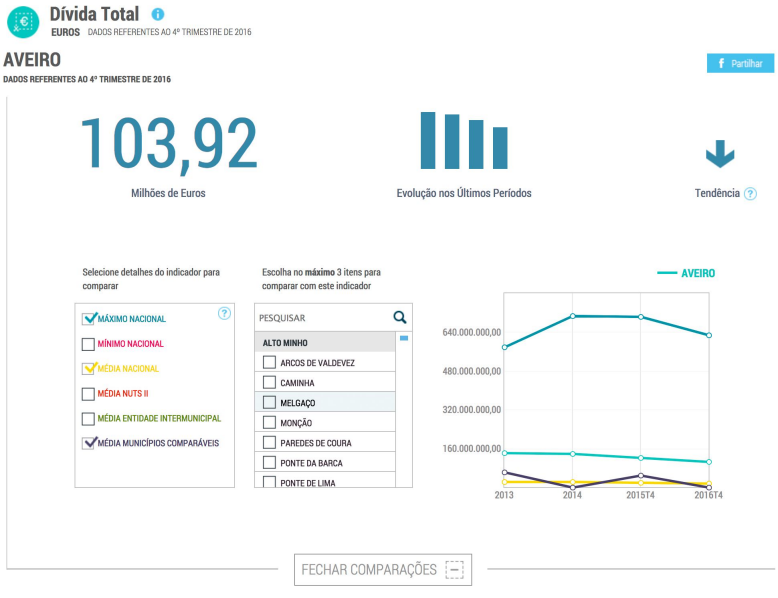

Figura 2.14: Vista sobre comparações [27]

2.5 DADOS ABERTOS

Segundo Thierauf [28], dados representam factos e figuras sem qualquer tipo de organização ou processamento e que não fornecem mais nenhuma informação acerca de padrões ou contexto. Para que os dados passem a ser entendidos como informação, têm de ser tratados, contextualizados, categorizados [29] de forma a poderem fazer sentido para quem os está a consultar.

A procura por um governo cada vez mais transparente é um tema, que apesar de não ser recente, continua a ser uma preocupação bastante atual devido, em grande parte, ao aparecimento de novas tecnologias. A partir de 2008 vários governos começaram a tornar os seus dados públicos para consulta [30].

Atualmente é extremamente importante a disponibilização de informação do sector público, de uma forma que seja fácil para os cidadãos aceder aos dados produzidos pelos Governos e Entidades Públicas. Com os avanços tecnológicos ocorridos nestes últimos tempos é possível aceder a estes dados de uma forma mais direta e, para além disso, conseguir reutilizar essa informação com outras finalidades ou contextos. O crescimento dos movimentos *Open* (*open source, open software*, etc) fez com que os Governos disponibilizassem cada vez mais a informação que produzem, facilitando e incentivando o acesso à mesma por parte da sociedade.

A nível europeu, a reutilização de informação pública foi impulsionada devido à revisão da Diretiva Europeia da Informação do Setor Público em 2013, que encoraja os países europeus a tornarem disponíveis mais dados em formato aberto, levando à criação de novos produtos baseados nessa informação. Em Portugal, o acesso aos documentos administrativos e sua reutilização é regulado pela Lei de Acesso aos Documentos Administrativos [31].

O Estado produz uma enorme quantidade de dados, em que grande maioria já são considerados públicos. Estes conjuntos de dados podem ser extremamente úteis para vários setores da sociedade, mas para isto acontecer, os diferentes grupos têm de ter facilidade no seu acesso e reutilização. A disponibilização dos dados públicos beneficia vários grupos da sociedade. Os cidadãos passam a ter a possibilidade de acesso aos dados que por direito já lhes pertencem. O Governo, ao disponibilizar os dados, dá uma imagem bastante mais transparente, para além de se tornarem mais eficientes. O setor privado passa a poder criar aplicações e serviços de grande valor comercial reutilizando e combinando os dados públicos.

2.5.1 DEFINIÇÃO

Numa perspetiva mais técnica, o termo dados abertos refere-se a quaisquer conjuntos de dados que possam ser reutilizados sem restrições por parte de licenças ou patentes, dados esses que devem estar bem estruturados e possam ser facilmente usados por instituições, cientistas ou pela comunidade web [32].

Segundo o website Open Definition [3], dados abertos são "dados que podem ser livremente acedidos, modificados e partilhados por qualquer um, em qualquer contexto". De uma forma mais específica, os dados para serem considerados abertos têm de cumprir com os seguintes requisitos na sua distribuição:

Os dados têm de ser disponibilizados sob uma licença aberta. Quaisquer termos adicionais (tais como os termos de uso ou patentes do licenciador) não podem contradizer os termos da licença. Para além disso, devem estar disponíveis como um todo e por um valor não superior ao custo da sua produção, preferencialmente devem poder ser descarregados via Internet gratuitamente. Qualquer informação adicional necessária para o cumprimento da licença deve acompanhar os dados. Para finalizar, os dados devem ser disponibilizados tal como colhidos da fonte, sem agregação ou modificação, e o mais rápido possível, quanto mais recentes e atuais, mais úteis. Especificamente, os dados devem poder ser lidos por máquinas, disponíveis em massa/bruto, e num formato aberto ou que pelo menos possam ser processados por uma ferramenta de código aberto.

2.5.2 PRIVACIDADE

Apesar de todos os benefícios que advêm da disponibilização dos dados do setor público em formato aberto, existem questões importantes relativas à privacidade. Nem toda a informação deve ser tornada pública, alguma deve continuar com acesso restrito à Administração Pública. As diretrizes relativas a forma como é feita a abertura dos dados são responsabilidade das entidades que os produzem em conjunto com a Comissão Nacional de Proteção de Dados (CNPD), no caso de Portugal.

2.5.3 COMO ABRIR OS DADOS

Para haver a possibilidade de consulta dos dados, é necessário que estes sejam publicados por alguém. Para que os dados sejam efetivamente abertos é importante seguir alguns padrões. Quando uma entidade decide abrir os seus dados, existe um

[3] http://opendefinition.org

conjunto de regras que deve seguir: decidir que dados irá abrir, aplicar uma licença aberta adequada, disponibilizar os dados de acordo com alguns princípios e normas e, por fim, definir as estratégias de divulgação.

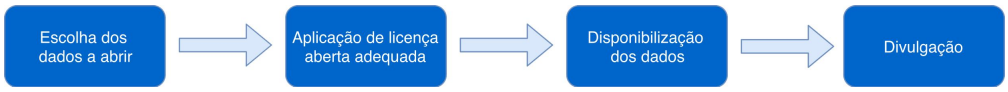

Figura 2.15: Fases do processo de abertura de dados (Adaptado de [30])

Escolha dos dados a abrir

Devido ao vasto conjunto de dados produzidos pelas entidades públicas, é difícil estabelecer um conjunto de regras, que cubram todos os casos, para determinar que dados abrir. Para tomarem esta decisão deverão primeiro perceber para quem estão a disponibilizar os dados e que interesse é que os mesmos possam suscitar nos potenciais consumidores.

De uma maneira geral, os dados que mais interesse suscitam nos utilizadores estão relacionados com as seguintes temáticas [30]:

- Acesso às entidades e serviços disponíveis;
- Transparência das entidades.

Quando possível, a informação a disponibilizar deve ser georreferenciada já que torna mais provável a sua utilização, tornando possível a sua visualização num mapa. Uma forma de tentar perceber que conjuntos de dados devem ser disponibilizados, é integrar os utilizadores, ou os potenciais interessados na informação, desde início no processo de escolha. É importante procurar saber as suas necessidades e as possíveis utilizações dos dados, através de questionários ou correio eletrónico, por exemplo. Outra forma de obter informação sobre que dados abrir é fornecer alguns conjuntos de dados e tentar perceber qual o potencial de estes serem utilizados.

Aplicação de licença adequada

Associada ao processo de abertura de dados está a escolha da sua licença. Como estamos a lidar com dados que pretendemos que sejam abertos, faz sentido que a licença escolhida também tenha um perfil aberto.

Quando alguém cria algo original, obtém os direitos sobre esse mesmo trabalho, ou seja, tem o poder para decidir se o seu trabalho pode ou não ser usado por terceiros e de que forma é que essa utilização pode ser feita. É aqui que entra o papel das licenças. Quando um autor aplica uma licença a um trabalho seu, está a declarar explicitamente

que direitos abdica ou mantém bem como a forma como os outros podem reutilizá-lo. Existem vários tipos de licenças, umas mais restritivas que outras como se pode ver na figura 2.16.

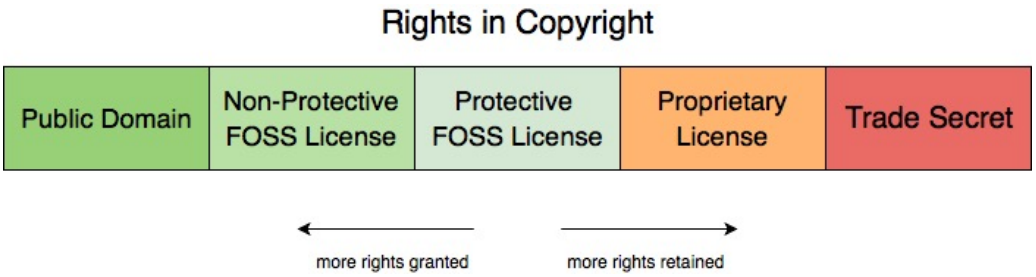

Figura 2.16: Espectro das licenças (adaptado de [33])

O espectro de licenças vai desde o software proprietário, que proíbe toda a cópia, redistribuição ou modificação, até às licenças de domínio público que permitem o uso, distribuição e até o sublicenciamento do software [34].

Quando um autor pretende disponibilizar o seu trabalho de forma aberta é sempre necessário aplicar uma licença aberta, abdicando dos direitos de propriedade originais, de forma a que terceiros possam reutilizar, modificar e partilhar o seu trabalho. A escolha da licença depende dos direitos que o autor está disposto a abdicar, sendo que num projeto de dados abertos, deve ser tido em conta a exploração dos dados e o interesse do público alvo.

Entre as licenças abertas destacam-se a Licença Creative Commons (CC) e a Open Data Commons (ODC), que estão divididas em vários níveis. A tabela 2.1 apresenta os vários níveis definidos em ambas as licenças.

Nível da Licença	**Licença Creative Commons**	**Licença Open Data Commons**
Domínio Público	CC0	PDDL
Atribuição	CC-by	ODC-by
Atribuição - Partilha	CC-by-sa	ODbL

Tabela 2.1: Níveis das licenças Creative Commons e Open Data Commons

As licenças de domínio público significam que o autor renuncia a todos os seus direitos sobre o seu trabalho, em todo o mundo. As de atribuição significam que a utilização do trabalho é livre, podendo ser utilizada para fins comerciais ou para criação de novos trabalhos, sendo que é essencial que seja dado o devido crédito ao seu autor. Por fim, as licenças de atribuição – partilha, significam que o autor, para além de pretender que lhe seja dado o crédito pela autoria do trabalho, também obriga a que os trabalhos derivados também sejam licenciados da mesma forma.

Para projetos relacionados com dados abertos é de extrema importância que a licença escolhida respeite um dos princípios chave da abertura de dados: garantir que os dados possam ser utilizados sem qualquer restrição. Dentro das licenças faladas anteriormente, a que mais garantias dá aos utilizadores que querem explorar os dados é a de domínio público, onde o autor renuncia a todos os seus direitos de propriedade.

Disponibilização dos dados

Esta é a etapa mais importante do processo de abertura porque de nada vale a abertura dos dados se estes não forem disponibilizados devidamente. Numa primeira fase, caso a entidade que quer abrir os dados não tenha muitos recursos, pode optar por uma solução mais simples e apenas acrescentar uma zona na sua página institucional, com referências para os dados disponibilizados junto com as ligações para baixar os ficheiros com os conjuntos de dados, nos formatos habitualmente mais usados (CSV, XLS, JSON).

Caso a entidade já tenha condições e mais recursos disponíveis, pode apostar numa solução mais complexa, baseada numa plataforma de catálogo e manutenção de dados abertos. Este tipo de plataformas consistem em ferramentas web que dão a possibilidade aos utilizadores de terem acesso a conjuntos de dados sob licenças abertas.

As plataformas de catálogo de dados permitem aos utilizadores várias funcionalidades, como pesquisar conjuntos de dados, filtrá-los por temas e descarregar os dados em formatos abertos. Para além das funcionalidades básicas referidas anteriormente, também é possível usar APIs para criar aplicações que necessitem dos dados, ver os metadados referentes aos conjuntos de dados e ter outro tipo de visualização dos dados (gráficos, mapas, etc). Já existem várias soluções que permitem às entidades ter uma destas plataformas, entre as quais se destacam as soluções abaixo descritas.

Open Data Soft [4]

O OpenDataSoft é uma solução que permite interação e visualização através de APIs geradas automaticamente. Para além de ser uma plataforma fácil de utilizar, tem um bom desempenho com conjuntos de dados bastante extensos e tem suporte a formatos geo espaciais. Tira partido do motor de pesquisa Elasticsearch e garante uma pesquisa e análise quase em tempo real. Esta solução está construída para também ser consultada em dispositivos móveis [35].

[4] https://www.opendatasoft.com/

Socrata [5]

Esta solução permite armazenar grandes conjuntos de dados. A publicação dos conjuntos de dados pode ser feita através de APIs ou de uma ferramenta para desktop. Os dados podem ser adicionados numa grande variedade de formatos, desde CSV, XLS até mesmo GeoJSON. São disponibilizadas ferramentas para atuar ao nível dos metadados que filtram a informação, exportam dados e criam visualizações. Uma funcionalidade interessante presente nesta solução é a possibilidade dos utilizadores terem acesso ao mesmo tipo de análises sobre conjuntos de dados sem terem de os publicar. Este tipo de solução já se encontra implementado em vários locais, nomeadamente em Chicago onde já existem 5.8 milhões de registos de crimes desde 2001. O Departamento de Polícia de Nova Iorque também usa esta solução para publicar informações sobre acidentes rodoviários [35].

CKAN [6]

Esta é a solução mais popular no que toca a portais para dados abertos. É uma ferramenta que possibilita ao utilizador gerir e publicar conjuntos de dados. É usada por governos, tanto a nível nacional como local, institutos de investigação e outras organizações que produzem grandes quantidades de informação.

Após a publicação dos conjuntos de dados, os utilizadores têm a possibilidade de usar os seus mecanismos de pesquisa para consultar os dados que necessitam. Estes dados podem ser visualizados de várias formas, através de tabelas, gráficos ou mapas.

O CKAN foi desenvolvido com base numa arquitetura modular o que permite que extensões sejam desenvolvidas para acrescentar funcionalidades às já existentes. O CKAN usa o seu modelo interno para armazenar metadados sobre os diferentes registos que depois vão ser usados na interface web para efeitos de pesquisa. Para além da interface web também fornece uma poderosa API que permite a aplicações externas comunicarem com a plataforma.

Estratégias de divulgação

Após uma entidade ter escolhido os dados a abrir, ter aplicado uma licença aberta e ter finalmente disponibilizado os conjuntos de dados, segue-se a última etapa, a definição de estratégias de divulgação. De nada serve os dados estarem abertos se ninguém sabe da sua existência. É então importante que a instituição invista algum tempo e recursos no processo de divulgação de forma a informar os potenciais utilizadores que estão a

[5] https://socrata.com/
[6] https://ckan.org

disponibilizar os dados em formatos abertos.

Existem várias estratégias de divulgação da abertura de dados. Numa primeira fase é de extrema importância que a entidade disponibilize uma forma de qualquer cidadão ter acesso às atualizações, como por exemplo através de um mecanismo de RSS onde o cidadão subscreve um determinado tópico e recebe automaticamente as atualizações relacionadas com esse tópico.

Outra forma bastante eficaz é a divulgação em comunidades de programadores, já que possivelmente serão estes que mais valor irão retirar dos dados através da criação de novas aplicações que relacionem dados provenientes de várias fontes, produzindo informação mais rica e com mais utilidade para os restantes cidadãos. O contato direto com instituições ou pessoas da mesma área dos dados pode também ser uma boa ideia para a divulgação, sendo importante ter um bom conhecimento do público-alvo.

Para os governos, a melhor forma de divulgação de novos conjuntos de dados disponíveis é através de um catálogo central onde possam ter diferentes áreas para diferentes departamentos. Atualmente, vários governos já adotaram soluções abertas para gerir os seus catálogos de dados. Através delas é mais fácil para os vários departamentos manterem os seus dados atualizados e para os potenciais utilizadores terem acesso aos dados de uma forma mais centralizada [36].

capítulo 3

Life Payt - Solução Proposta

O objetivo deste capítulo é apresentar uma solução que demonstre a eficiência das metodologias PAYT em ambiente de cidade inteligente, através da criação de uma plataforma capaz de integrar dados provenientes de várias fontes, tornando o processo transparente para todos os *stakeholders*. A implementação, que será aplicada num cenário real está integrada no projeto LIFE PAYT e será apresentada no próximo capítulo.

Seguido desta breve introdução, irá ser apresentada, na secção 3.1, a descrição do projeto LIFE PAYT e os seus objetivos ao desenvolver uma plataforma para gestão eficiente dos resíduos em três municípios nacionais (Aveiro, Lisboa e Condeixa) e dois internacionais (Vrilissia e Larnaka).

No processo de criação de uma plataforma deste tipo vários aspetos têm de ser tidos em conta para assegurar fiabilidade. Nesse sentido existe a necessidade de identificar os requisitos e capacidades que o sistema tem de possuir para garantir a satisfação de todos os *stakeholders*. Estes serão descritos na secção 3.2 e alguns exemplos de casos de uso serão identificados na secção 3.3. Finalmente, na secção 3.4 será apresentada a arquitetura do sistema sob a forma de diagrama, bem como a descrição de cada componente que foi desenvolvido e as interações entre eles.

3.1 PROJETO LIFE PAYT

3.1.1 VISÃO GERAL E OBJETIVOS

O grande objetivo do projeto LIFE PAYT é definir um novo modelo para gestão de resíduos em ambiente de cidades inteligentes. De modo a atingir o objetivo, o projeto vai seguir uma estratégia altamente dependente da existência de uma plataforma de software. Esta plataforma é responsável por fornecer aos cidadãos informação clara sobre a sua produção de resíduos, interfaces capazes de motivar os utilizadores a entender melhor o processo a nível local e possibilidade de integração com os sistemas de software já existentes nos municípios abrangidos pelo projeto.

O novo modelo de gestão de resíduos vai ter como base o modelo PAYT, que consiste na cobrança, por parte da entidade responsável, de uma tarifa associada apenas à quantidade de resíduos gerada pelo produtor. O modelo vai ser suportado por uma plataforma de gestão criada para processar e atuar conforme a informação ambiental recebida.

O sistema irá receber informação referente à produção de resíduos em cada local, nomeadamente, quantidade, data de leitura e informação relativa a que produtor corresponde essa leitura. Depois de recebida essa informação, o sistema irá estruturá-la e armazená-la, podendo esta depois ser consultada através dos portais criados para o efeito. Nestes portais poderá ser possível observar gráficos, tabelas, valores absolutos, valores médios, entre outro tipo de indicadores.

O projeto contempla também um espaço reservado para os dados abertos, porque apesar de tudo, a informação recolhida, desde que devidamente anonimizada, é considerada informação pública. Relativamente aos dados abertos, o sistema vai ter capacidade para se integrar com sistemas já existentes nos municípios, nomeadamente, portais para dados abertos. Essa integração é realizada através de envios periódicos de ficheiros com informação agregada e anonimizada para os portais, caso estes existam.

A interação entre o utilizador e o sistema será feito através de uma interface web que será capaz de responder às necessidades dos utilizadores. A interface pode ser acedida através de um dispositivo móvel, com ligação à internet, já que esta está construída de forma a que todos os componentes tenham um tamanho e um comportamento adequado ao tamanho do dispositivo que esteja a aceder.

No que toca aos cidadãos/produtores, estes poderão consultar o seu histórico de produção, comparar as suas tarifas com produtores semelhantes (mesma área de comércio, agregado familiar semelhante, etc), simular tarifas com base no número de contentores contratualizados, no caso de produtores não domésticos.

Graças à informação que os municípios têm ao seu dispor, poderão tornar mais eficientes as rotas de recolha dos resíduos. Com isto, deixam de efetuar rotas que não fazem sentido porque os contentores não estão cheios, e estas passam só a ocorrer quando realmente são necessárias. Estes desenvolvimentos permitem aos municípios baixarem os seus orçamentos e ao mesmo tempo manter os espaços limpos. Para além disto, os municípios têm também acesso a informação mais pormenorizada e de uma forma mais rápida relativo a cada produtor na sua área de intervenção.

3.1.2 STAKEHOLDERS

As partes envolventes em qualquer projeto desempenham um papel importantíssimo no que toca ao seu sucesso, ou por terem uma influência direta na execução do projeto ou simplesmente por serem o utilizador final do produto. Por isto tudo, é crucial identificar todos os stakeholders e as suas expetativas para garantir que todas as partes envolvidas estão motivadas e agradadas com o avançar do projeto, para assim obter melhores resultados.

No projeto LIFE PAYT as partes envolvidas são essencialmente 3, nomeadamente:

- Produtores domésticos e não domésticos
- Municípios
- Administradores do sistema

Os municípios e os produtores (tanto os domésticos como os grandes produtores) são os principais stakeholders pois são eles que vão tirar partido das funcionalidades e usabilidade do sistema. Com a ajuda do sistema, os municípios vão ter grandes benefícios económicos já que vão poder ter rotas de recolha bem mais eficientes, fazendo com que os custos de recolha baixem de forma substancial. Os produtores tiram partido do sistema para se manterem sempre informados sobre as tarifas que lhes são cobradas entre outros indicadores importantes. Os administradores da plataforma, graças às ferramentas de monitorização presentes no sistema, vão poder atuar mais cedo em caso de falhas, já que vão ser alertados caso elas ocorram.

3.2 REQUISITOS DO SISTEMA

A solução proposta, como já foi dito anteriormente, vai ser aplicada num cenário do mundo real, integrado no projeto LIFE PAYT que já foi descrito na secção 3.1.

Os requisitos expostos nesta seção deverão estar de acordo com os objetivos descritos também na secção 3.1. Como o produto final vai ser usado num cenário real, foi feito um levantamento de requisitos junto de alguns stakeholders, nomeadamente dos municípios. Após várias etapas chegou-se a uma lista final de requisitos não funcionais que vai ser apresentada de seguida.

Acesso

A aplicação web deve possuir uma página que não necessita de autenticação contendo dados gerais do projeto (contadores, parceiros, estado, etc). Para além desta página, deve disponibilizar uma parte em que o utilizador tenha que estar autenticado. Esta parte da aplicação é onde o utilizador poderá observar tudo o que está relacionado consigo.

Desempenho

A aplicação deve possuir a capacidade de lidar com uma grande quantidade de dados para produção de gráficos, tabelas e médias, sem que o utilizador sinta atrasos na sua visualização mesmo que as condições de conexão à internet não sejam as melhores, i.e. baixa largura de banda ou sinal fraco. A plataforma deverá suportar um bom funcionamento num município/cidade, o que implica ter capacidade de lidar com informação de milhares de utilizadores.

Usabilidade

O utilizador deverá poder facilmente e sem necessidade prévia de conhecimento da aplicação encontrar forma de efetuar todas as ações pretendidas. Todas as funcionalidades da aplicação devem estar claramente identificadas para que o utilizador consiga saber logo à partida o que cada ação irá fazer no sistema. Para além disto a interface deve ter um design responsivo, i.e. a interface deve adaptar-se ao tamanho do ecrã do dispositivo de forma a facilitar a visualização por parte do utilizador.

Robustez

Caso ocorra uma falha no sistema, este deve voltar a estar disponível num curto espaço de tempo e, se possível, sem que o utilizador perceba que ocorreu uma falha. Os dados mostrados ao utilizador devem ser os mais recentes na base de dados e caso ocorram falhas, estas não se devem refletir nos dados.

Internacionalização

A aplicação deve estar disponível em várias línguas e com auxílio de símbolos nas

ações principais do sistema. A interface deverá permitir a inserção de novos idiomas sem ser necessário alterar o código.

Open Source

Como requisito do projeto e consequentemente da aplicação, todo o código desenvolvido deverá ser disponibilizado num repositório de código aberto de modo a poder ser reutilizado/adaptado por potenciais interessados.

Tolerância a falhas

A aplicação deve possuir capacidade de lidar com falhas, quer sejam falhas do servidor ou falhas de conectividade no lado do cliente. No caso de ocorrerem este tipo de falhas, a aplicação deve possuir mecanismos de cache para apresentar sempre os últimos valores obtidos da base de dados, dando informação ao utilizador que os dados apresentados podem não ser os dados mais recentes.

Interface de fácil utilização

A solução descrita neste capítulo tem como principal objetivo definir um novo modelo mais eficiente de gestão de resíduos, reduzindo a sua produção e efetuar rotas de recolha mais eficientes. Para que este objetivo se concretize, é extremamente importante a visualização dos dados referentes à produção de resíduos. Então, é necessário que a aplicação tenha uma interface gráfica amigável e de fácil compreensão. Para além dos utilizadores normais da aplicação, também existe uma equipa de manutenção que deve ter uma interface amigável destinada a gestão da plataforma.

3.3 CASOS DE USO

A interface desenvolvida procura satisfazer os requisitos de três tipos de utilizadores: os produtores de resíduos, os municípios e os gestores da plataforma. Cada um dos tipos de utilizador mencionados anteriormente pode tirar partido da interface para executar diferentes ações que vão ser apresentados de seguida, divididas de acordo com o tipo de utilizador a que se referem.

3.3.1 PRODUTORES DE RESÍDUOS

A solução apresentada contempla dois tipos de produtores de resíduos, os grandes produtores (modelo de identificação do recipiente – esquema por subscrição) e os

pequenos produtores, que correspondem aos clientes domésticos ou pequenos comércios.

Relativamente às interações dos grandes produtores com o sistema, a figura 3.1 mostra os principais casos de uso a serem abordados nesta solução, cada um a fazer o seguinte:

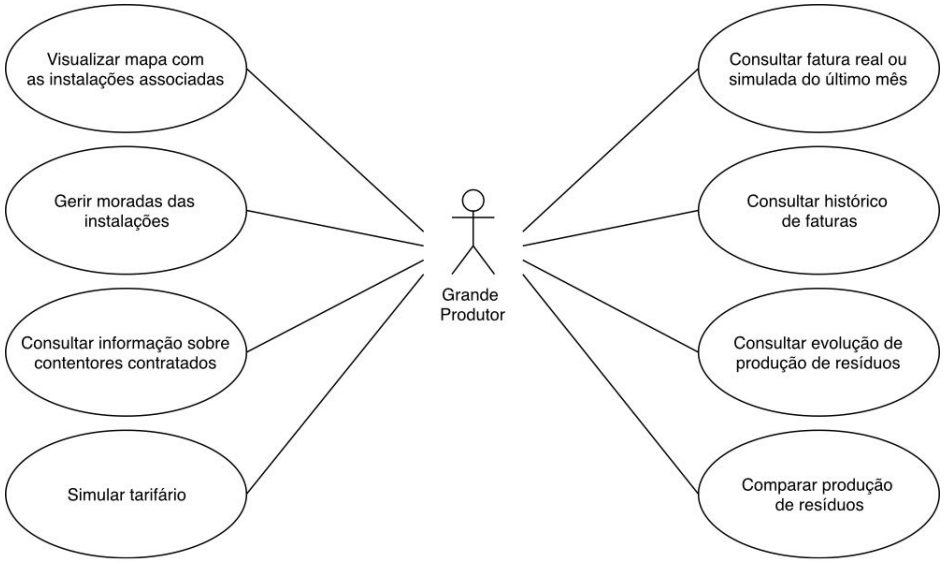

Figura 3.1: Diagrama de caso de uso: interação com grandes produtores

- Consultar faturas: consultar a fatura real ou simulada do último mês. Para além disso, o utilizador terá acesso ao histórico de faturas (reais e simuladas).
- Consultar produção de resíduos: aqui, o utilizador poderá observar a quantidade de resíduos que está a produzir no mês atual, bem como um histórico das quantidades produzidas nos meses anteriores.
- Comparar produção de resíduos: com esta ação o utilizador tem a possibilidade de comparar a quantidade de resíduos que está a produzir com os produtores do mesmo setor de atividade.
- Visualizar mapa com instalações associadas: neste mapa o utilizador tem acesso a todas as localizações das suas instalações.
- Gerir morada das instalações: adicionar, editar ou remover a morada de uma instalação em específico.
- Consultar informação sobre contentores instalados: aqui, o utilizador pode visualizar quantos contentores tem instalados em cada instalação, o seu tipo, e a data da última recolha dos mesmos.
- Simular tarifário: com esta funcionalidade o utilizador pode simular a tarifa que seria cobrada de acordo com o número e tipo de contentor instalado, caso pretenda alterar o seu contrato.

Relativamente às interações dos pequenos produtores com o sistema, a figura 3.2 mostra os principais casos de uso, alguns que são comuns aos grandes produtores, apenas com a seguinte diferença:

- Visualizar mapa com as localizações dos contentores: neste mapa, o utilizador tem acesso às localizações dos contentores presentes na sua área.

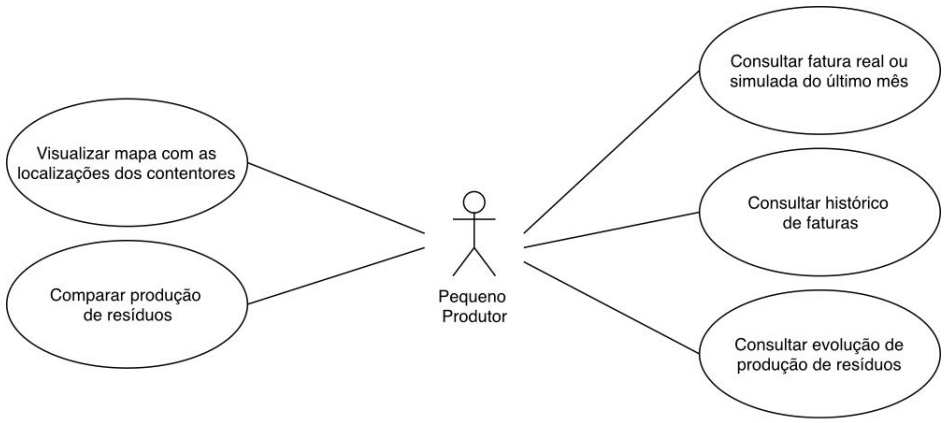

Figura 3.2: Diagrama de caso de uso: interação com pequenos produtores

3.3.2 MUNICÍPIOS

Para além dos produtores de resíduos existem outros utilizadores do sistema, nomeadamente os municípios (os seus responsáveis) que desempenham um importante papel na plataforma. São eles que fornecem a lista de clientes e os valores atualmente cobrados aos produtores.

Relativamente às interações dos municípios (associados a pequenos produtores) com o sistema, a figura 3.3 mostra os principais casos de uso a serem abordados nesta solução, cada um a fazer o seguinte:

- Enviar ficheiro com lista de clientes: processo de enviar um ficheiro com a lista de todos os produtores na sua área.
- Enviar ficheiro com faturas: processo de enviar um ficheiro com os valores das tarifas de todos os produtores que depois será processado e armazenados todos os valores na base de dados.
- Consultar preferências dos produtores: consultar horários e semanas preferenciais para deposição de resíduos por parte dos produtores.
- Consultar valores totais das faturas: consultar o valor total da tarifa real ou simulada do município no último mês.

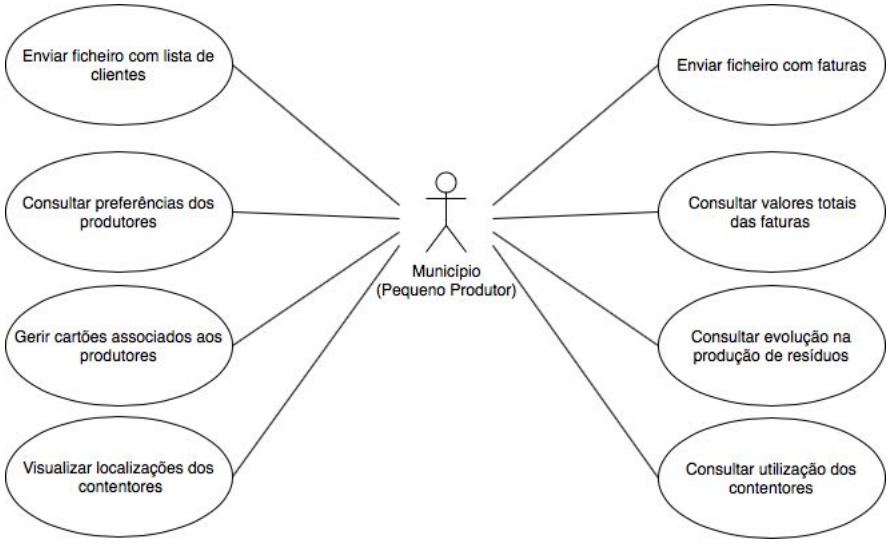

Figura 3.3: Diagrama de caso de uso: interação com municípios (pequenos produtores)

- Gerir cartões associados aos produtores: aqui o município poderá adicionar/remover cartões do sistema, bem como associar um cartão a um produtor.
- Consultar evolução na produção de resíduos: é aqui que o município pode observar a evolução da produção de resíduos.
- Visualizar localizações dos contentores: o município terá acesso a um mapa onde consta a localização de todos os contentores posicionados na sua área.
- Consultar utilização dos contentores: Verificar o estado de ocupação de cada contentor.

Relativamente às interações dos municípios (associados a grandes produtores) com o sistema, a figura 3.4 mostra os principais casos de uso a serem abordados nesta solução, cada um a fazer o seguinte:

- Enviar ficheiro com lista de clientes: processo de enviar um ficheiro com a lista de todos os produtores na sua área.
- Enviar ficheiro com faturas: processo de enviar um ficheiro com os valores das tarifas de todos os produtores que depois será processado e armazenados todos os valores na base de dados.
- Verificar produtor com maior/menor produção de resíduos: é aqui que o município poderá verificar que em empresa produziu mais/menos resíduos.
- Consultar valores totais das faturas: consultar o valor total da tarifa real ou simulada do município no último mês.
- Verificar produtor com maior/menor taxa de separação: é aqui que o município poderá verificar que produtor apresenta uma maior/menor taxa de separação.

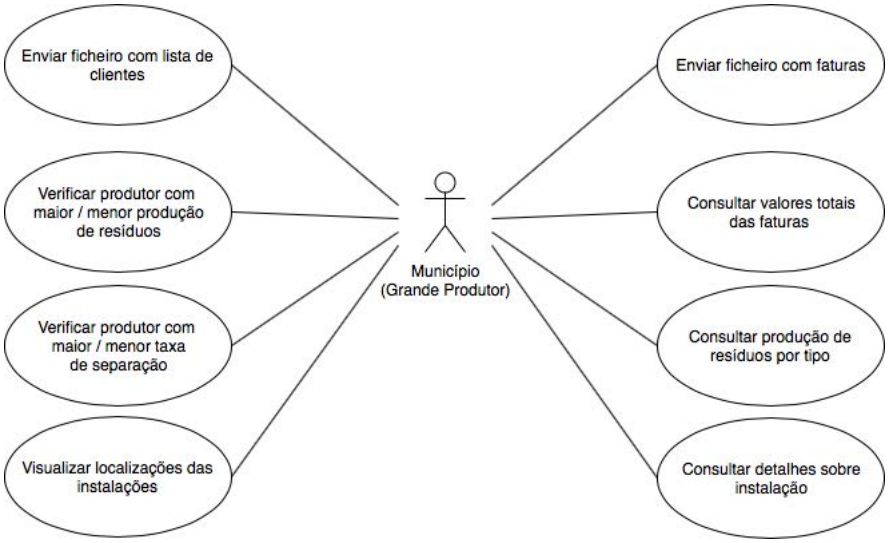

Figura 3.4: Diagrama de caso de uso: interação com municípios (grandes produtores)

- Consultar produção de resíduos por tipo: é aqui que o município pode observar a evolução da produção de resíduos por tipo de material recolhido.
- Visualizar localizações das instalações: o município terá acesso a um mapa onde consta a localização de todos as instalações da sua área.
- Consultar detalhes sobre instalação: é aqui que será possível obter mais detalhes sobre cada instalação, nomeadamente o número e tipo de contentores contratualizados entre o município e o produtor.

3.3.3 ADMINISTRADOR

Relativamente às interações do administrador com o sistema, a figura 3.5 mostra os principais casos de uso a serem abordados nesta solução, cada um a fazer o seguinte:

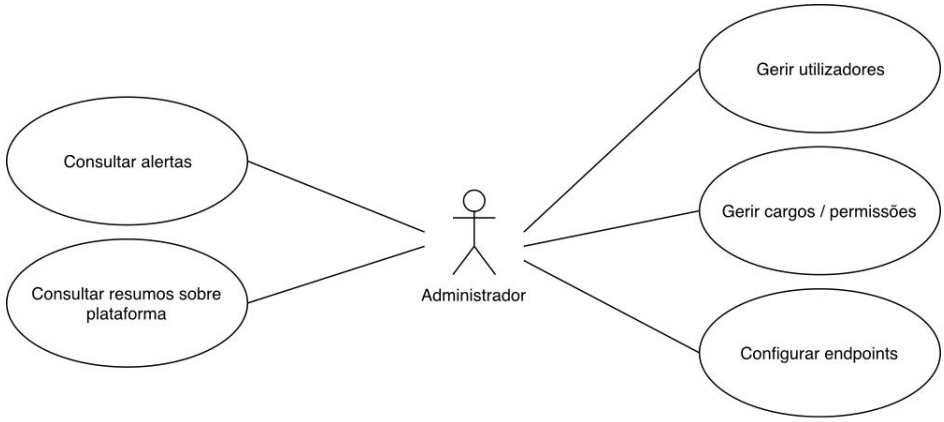

Figura 3.5: Diagrama de caso de uso: interação com administrador

- Consultar alertas: é aqui que o administrador pode consultar, caso existam, os alertas sobre o uso da plataforma, i.e. elevado número de tentativas de início de sessão falhadas, falhas no sistema, etc.
- Consultar resumos sobre plataforma: grande parte das interações do utilizador com o sistema são guardadas, sob a forma de logs que serão posteriormente processados e darão origem a relatórios que poderão ser consultados pelo administrador.
- Gerir utilizadores: adicionar, editar, remover utilizadores do sistema.
- Gerir cargos / permissões: aqui o administrador pode executar todas as ações relativas aos cargos e permissões, i.e. atribuir um cargo (com permissões associadas) a cada utilizador, alterar o cargo atribuído.
- Configurar *endpoints*: aqui o administrador poderá configurar tudo relativo aos *endpoints* existentes para comunicação com sistemas externos.

3.4 SOLUÇÃO LIFE PAYT – ARQUITETURA

Tendo em conta os requisitos do projeto LIFE PAYT, a arquitetura proposta pretende criar uma plataforma não só robusta, que recupere das falhas, mas também com elevado desempenho já que irá lidar com grandes quantidades de dados. Este capítulo tem como objetivo apresentar a arquitetura, especificando cada um dos componentes que a constituem e descrevendo os seus objetivos e interações.

Como este trabalho incidiu mais sobre a parte da apresentação dos dados e exportação dos mesmos para a plataforma de dados abertos (CKAN), irá ser feita uma descrição mais detalhada destes enquanto que relativamente aos módulos restantes apenas irá ser feita uma breve descrição.

3.4.1 ARQUITETURA GERAL

Na subsecção 2.3.5 foram enunciados os benefícios que uma arquitetura baseada em microserviços apresenta quando comparada com uma arquitetura monolítica. Tendo isso em conta a arquitetura proposta pretende seguir esse mesmo estilo. Um dos problemas associados a este tipo de arquiteturas é a necessidade, por parte da equipa de programadores, de gerir vários ambientes de desenvolvimento, caso os vários serviços sejam construídos com recurso a tecnologias e linguagens diferentes. A solução para o problema anterior seria encapsular cada microserviço num *container*.

Para ajudar a gerir esses mesmos *containers* existe o Docker [1], uma ferramenta

[1] https://www.docker.com/

de código aberto de desenvolvimento e execução de aplicações que tem como objetivo facilitar a criação e gestão de ambientes isolados. A vantagem de usar esta ferramenta é o facto da configuração do *container* só ter de ser feita uma vez e este poder ser posteriormente instanciado em qualquer máquina que tenha o Docker instalado.

Em termos de utilização de recursos, esta é uma solução bastante económica já que num sistema de virtualização tradicional existe um sistema operativo isolado e no caso do Docker, temos recursos isolados que fazem uso de bibliotecas do próprio *kernel* do sistema operativo nativo. Com isto, consegue-se obter um melhor desempenho comparativamente à virtualização [37].

O diagrama da arquitetura apresentado na 3.6 mostra todos os módulos nos quais a solução está dividida.

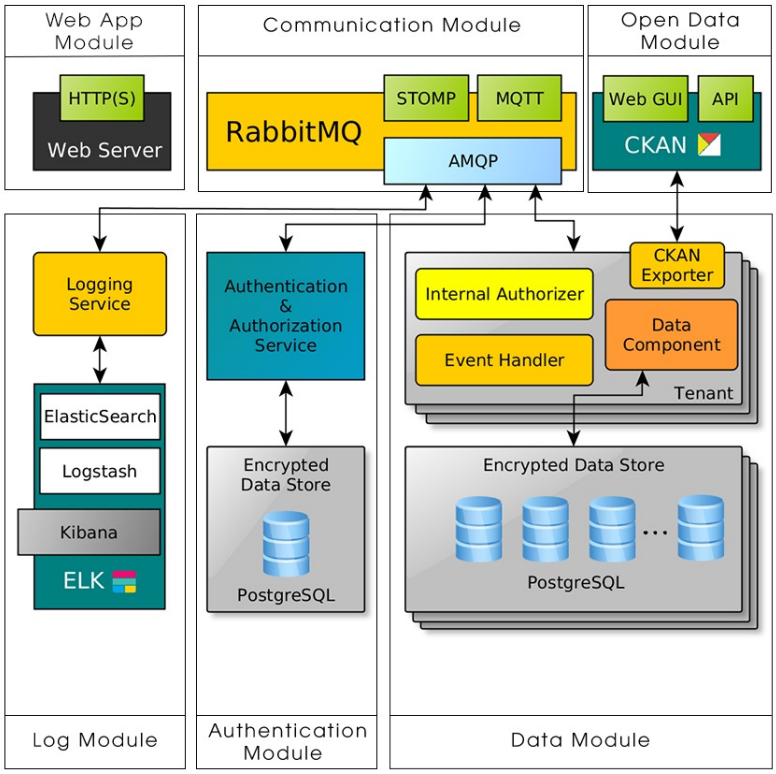

Figura 3.6: Vista geral da arquitetura

Authentication Module

Este é o módulo responsável pela parte de autenticação no sistema. Quando o utilizador submete as suas credenciais para poder aceder à plataforma, estas são enviadas para este módulo que irá confrontar as mesmas com a base de dados de utilizadores. Caso as credenciais sejam válidas o utilizador terá acesso às funcionalidades disponíveis de acordo com o seu papel no sistema.

Communication Module

É neste módulo que é gerida toda a parte da comunicação entre a aplicação web e os restantes módulos (base de dados, autenticação e módulo responsável pela parte de *logging*). Toda a comunicação, exceto a comunicação com o módulo de dados abertos, é feita através deste módulo que implementará um sistema de comunicação com base em filas de mensagens. A comunicação entre o módulo de dados e o de dados abertos é feita diretamente através da API disponibilizada pelo CKAN.

Data Module

Este é o módulo onde se encontram as bases de dados com a informação armazenada pela plataforma. É aqui que irá ser guardada toda a informação relativa a moradas associadas a cada utilizador, a quantidades de resíduos produzidos, tarifas cobradas. Toda esta informação irá estar dividida por municípios e devidamente isolada dos restantes. Internamente, existe um módulo próprio de autorização para que só utilizadores com as devidas permissões possam efetuar qualquer tipo de ação nos municípios.

Log Module

Todas as interações do utilizador com o sistema são guardadas sob a forma de logs. Toda essa informação é enviada para o serviço de *logging* que irá posteriormente processá-la e armazená-la numa estrutura bem definida. Por exemplo, antes de um utilizador começar a utilizar a plataforma terá de se autenticar. O resultado desse processo de autenticação será enviado para este módulo que irá processar essa mensagem e registar um acesso à plataforma bem ou mal sucedido.

Devido à existência deste tipo de registos, é possível obter as mais variadas estatísticas, como por exemplo o número de acessos num determinado intervalo de tempo ou o número de falhas que ocorreram no sistema e o seu motivo. Todos estes indicadores estão disponíveis para consulta, apenas para o administrador, numa interface personalizável, sob a forma de gráficos, tabelas ou outro tipo de visualização.

É graças a este módulo que o administrador tem possibilidade de observar o estado

do sistema, se está ou não disponível, a utilização dos recursos, entre outras coisas. É também aqui que vão ser gerados os relatórios/resumos sobre a plataforma que depois estarão disponíveis para uma vista mais rápida na aplicação web.

Open Data Module

Como já foi descrito na secção 3.1.1, o projeto contempla um módulo relativamente aos dados abertos. Este módulo é responsável por criar relatórios com informação agregada e anonimizada e enviar os mesmos para os respetivos portais de dados abertos dos municípios.

Os relatórios irão ser criados de forma periódica e autónoma, juntando a informação relativa a cada a utilizador, como por exemplo o valor da tarifa cobrada e a quantidade de resíduos produzidos. Antes dos relatórios serem submetidos para os respetivos portais, têm de ser anonimizados pois apesar da informação ser de interesse público, esta não pode ser associada a nenhum cidadão.

Na interface web dos portais referidos anteriormente, é possível ver os relatórios enviados pelo sistema, bem como outras informações, entre elas a data de criação e o intervalo de tempo a que o relatório se refere.

Web App Module

É com recurso a este módulo que o utilizador consegue interagir com a plataforma. Relativamente ao administrador, é aqui que irá poder consultar a informação relativa aos utilizadores do sistema bem como resumos da plataforma de forma a poder monitorizar e acompanhar o seu estado. Aqui, a aplicação web deve providenciar uma interface de fácil utilização juntamente com interfaces intuitivas para fácil monitorização.

Os cidadãos têm a possibilidade de consultar toda a informação sobre a sua produção de resíduos e tarifas que lhe foram cobradas a partir da mesma aplicação web. Graças a ela, os municípios também podem acompanhar o comportamento dos cidadãos da sua área relativamente à produção de resíduos e assim tornar a gestão de resíduos mais eficiente. Esta interface também funciona como uma ligação para outro tipo de interfaces, tais como a interface responsável pela monitorização mais detalhada e a interface no portal de dados abertos. A comunicação com os restantes módulos é feita com recurso a um cliente Web-STOMP que atua como uma ponte que expõe o protocolo STOMP em websockets de forma direta ou emulada.

Resumo

Nesta secção foi apresentado um modelo de arquitetura que satisfaz todos os requisitos especificados bem como os casos de uso para cada utilizador. O encapsulamento dos módulos em *containers* Docker permite que estes apenas utilizem os recursos que realmente necessitam, diminuindo assim a carga sobre a máquina onde estão a correr. Para além disso, esta arquitetura permite que cada módulo possa estar alojado em locais diferentes. Isto torna-se bastante importante caso os municípios pretendam que as suas instâncias estejam alojadas nos seus servidores.

Graças ao módulo de comunicação, que permite que os módulos troquem informação com base num sistema de filas de mensagens, novos módulos podem ser adicionados ao sistema sem grandes restrições ao nível das tecnologias utilizadas.

CAPÍTULO 4

LIFE PAYT - IMPLEMENTAÇÃO

No capítulo 3 foi apresentada uma arquitetura para uma solução, que satisfaz todos os requisitos necessários para criar uma plataforma que demonstre eficiência das metodologias PAYT, em ambiente de cidade inteligentes.

O objetivo deste capítulo é descrever a implementação tendo por base este modelo, explicando todas as etapas e a razão pela qual cada decisão foi tomada ao longo da implementação. Sempre que for pertinente, partes de código e diagramas irão ser usados para justificar a escolhas feitas ao longo deste processo.

Como já foi anteriormente dito, esta implementação foi aplicada num cenário do mundo real, de acordo com o projeto LIFE PAYT. Dentro deste mesmo projeto, estão a ser desenvolvidos outros trabalhos, focados nos módulos de comunicação, gestão de dados e utilizadores. Esta dissertação está então focada na implementação dos restantes módulos (dados abertos e aplicação web).

4.1 OBJECTIVOS

Apesar da implementação da solução por completo, apresentada no capítulo 3 ser um resultado ideal, considerando o tempo disponível, isto não seria realista. Devido a este facto, como a implementação é para ser aplicada num cenário real, nos 5 municípios enunciados no capítulo 3, foram definidos alguns objetivos para garantir que as funcionalidades mais importantes fossem implementadas.

Tendo em conta que os principais utilizadores desta plataforma são os cidadãos e os municípios, foi dada uma maior importância ao módulo responsável pela aplicação web.

Tomando isto em consideração, todos os casos de uso da interação dos municípios e dos cidadãos foram implementados, de forma a possibilitar a consulta dos dados relativos a cada utilizador.

Dos requisitos definidos, destacam-se dois como os mais importantes, a existência de uma interface de fácil utilização e o desempenho da mesma. O primeiro foi abordado nesta implementação recorrendo a técnicas de apresentação de dados próprias para o efeito, fazendo com que o utilizador não sinta nenhuma dificuldade quando pretende executar qualquer tipo de ação na plataforma. O segundo requisito foi cumprido recorrendo ao uso de linguagens e bibliotecas de programação com elevado desempenho.

4.2 APLICAÇÃO WEB

Nesta secção irão ser apresentadas todas as tecnologias adotadas no decorrer do processo de implementação juntamente com as razões que levaram a ser feita a escolha sobre essa tecnologia em detrimento de outras.

4.2.1 TECNOLOGIAS ADOTADAS

Um dos componentes principais da plataforma é a aplicação web pois é através desta que os vários tipos de utilizadores irão interagir com o sistema. Tendo isso em conta, no processo de implementação pretende-se obter uma aplicação fácil de utilizar e ao mesmo tempo com elevado desempenho, tanto em termos de rapidez, como de capacidade de lidar com grandes quantidades de dados.

Para ir de encontro com os objetivos definidos acima é preciso então escolher a tecnologia que mais se adequa aos mesmos. Para a criação da interface foram escolhidas as habituais tecnologias de desenvolvimento de interfaces de utilizador na web, sendo elas o HTML, CSS e JS. Resta então fazer a escolha entre que framework/biblioteca de JS usar. Podia ter sido usado JS puro, mas iria tornar o desenvolvimento muito mais complexo, sendo preferível usar uma tecnologia que permitisse que a atenção fosse mais virada para a forma como estruturar a aplicação. Entre as tecnologias mais populares no momento destacam-se o AngularJS, desenvolvido e mantido pela Google e o ReactJS, criado pelo Facebook (fig. 4.1).

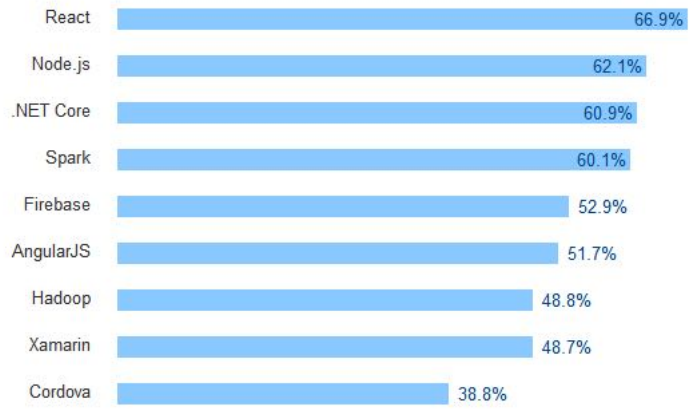

Figura 4.1: As frameworks JS mais votadas no inquérito do Stack Overflow (2017) [38]

AngularJS [1]

O AngularJS é uma framework de código aberto usada para construção de aplicações Web. Assenta numa arquitetura Modelo-Vista-Controlador (MVC) e conta com uma grande comunidade de apoio. O AngularJS vem resolver o problema de desenvolver aplicações de página única, i.e. aplicações capazes de mostrar conteúdo novo sem sair da mesma página, trazendo novas funcionalidades ao HTML. O foco principal do Angular é criar e ter uma aplicação a correr rapidamente.

A sua principal funcionalidade é a ligação bidirecional dos dados, que se refere a *event listeners* que detetam qualquer alteração de dados na aplicação, seja esta desencadeada pelo utilizador ou pela base de dados [39]. Com recurso a esta tecnologia é possível criar aplicações responsivas e com baixos tempos de carregamento.

Mas como qualquer outra tecnologia esta também tem os seus pontos fracos. O desempenho das aplicações que tenham que lidar com grandes quantidades de dados acaba por não ser o expectável. E, num mundo onde o desempenho das aplicações cada vez se torna mais importante, qualquer atraso no carregamento de páginas acaba por levar ao descontentamento do utilizador. Para além dessa perda no desempenho, é uma tecnologia extremamente estruturada hierarquicamente, tornando-se nalguns casos bastante complexa nos primeiros tempos em que se começa a desenvolver em AngularJS. Por último, é uma framework que ocupa um tamanho considerável.

[1] https://angularjs.org/

ReactJS [2]

Apesar de em vários locais ser dito que ReactJS é uma framework, isso não é verdade. ReactJS é apenas uma biblioteca JS que facilita a gestão da Document Object Model (DOM), o que significa que na grande maioria dos casos o ReactJS seja usado em conjunto com mais bibliotecas para formar uma solução completa.

Na documentação oficial [40], o ReactJS é definido como uma biblioteca JS para construção de interfaces de utilizador. É aqui que esta tecnologia é realmente boa, na criação de interfaces, já que o programador apenas precisa de construir simples visualizações para cada estado e a própria biblioteca se encarrega de atualizar e voltar a carregar apenas os componentes que alterem o seu estado. É uma biblioteca baseada em componentes, ou seja, é possível criar componentes que gerem o seu próprio estado e, posteriormente, juntar esses componentes criados para construir interfaces de utilizador mais complexas.

A principal funcionalidade desta biblioteca é a existência de uma DOM virtual que é guardada em memória juntamente com a DOM real. Sempre que ocorre alguma alteração no estado de um componente, o ReactJS compara a nova árvore HTML com a atual e apenas atualiza os componentes que foram afetados com a alteração, tornando a atualização do conteúdo apresentado na página mais eficiente e rápida.

Apesar dos pontos fortes apresentados anteriormente também existem alguns pontos fracos, nomeadamente a existência de alguns conflitos com outras bibliotecas de manipulação da DOM e o facto da sua documentação não ser suficientemente profunda.

Conclusão

Como um dos principais requisitos desta interface é o desempenho, e tendo em conta que a interface terá de lidar com grandes quantidades de dados nomeadamente para construção de gráficos e tabelas, a escolha da tecnologia a usar para a sua construção foi o ReactJS. Esta biblioteca garante tempos de carregamento mais baixos relativamente ao AngularJS juntamente com facilidade de estruturação de código, já que não é tão complexa. Para além destes fatores, os tamanhos das duas é bastante diferente, com substancial vantagem para o ReactJS em relação ao AngularJS como pode ser visto na tabela 4.1. Junto com ReactJS está incluído o tamanho da biblioteca ReactDOM pois é necessária para a manipulação da DOM com mais facilidade e é habitual as duas serem usadas em conjunto.

[2] https://reactjs.org/

Tecnologia	Tamanho
ReactJS + ReactDOM	133K
AngularJS v2	566K

Tabela 4.1: Tamanhos dos ficheiros referentes à tecnologia. Valores retirados de [41]

4.2.2 IMPLEMENTAÇÃO

Na secção 3.2 um dos requisitos definidos foi a existência de uma página, onde qualquer utilizador não autenticado, pudesse ter acesso a alguma informação geral sobre o projeto e de uma área onde aí sim, fosse preciso autenticar-se para ter acesso à informação lá presente. Para tal efeito foi criada a página que se pode ver na figura 4.2, onde estão presentes alguns indicadores. Na mesma página encontra-se um botão que dá acesso a um formulário para efetuar o login na área privada. Após o utilizador submeter os dados referentes à sua conta é feito um pedido ao sistema para confirmar se os dados estão corretos e pertencem a uma conta com acesso à plataforma. Em caso afirmativo, o utilizador é redirecionado para a sua área privada. Caso contrário, continuará na mesma página.

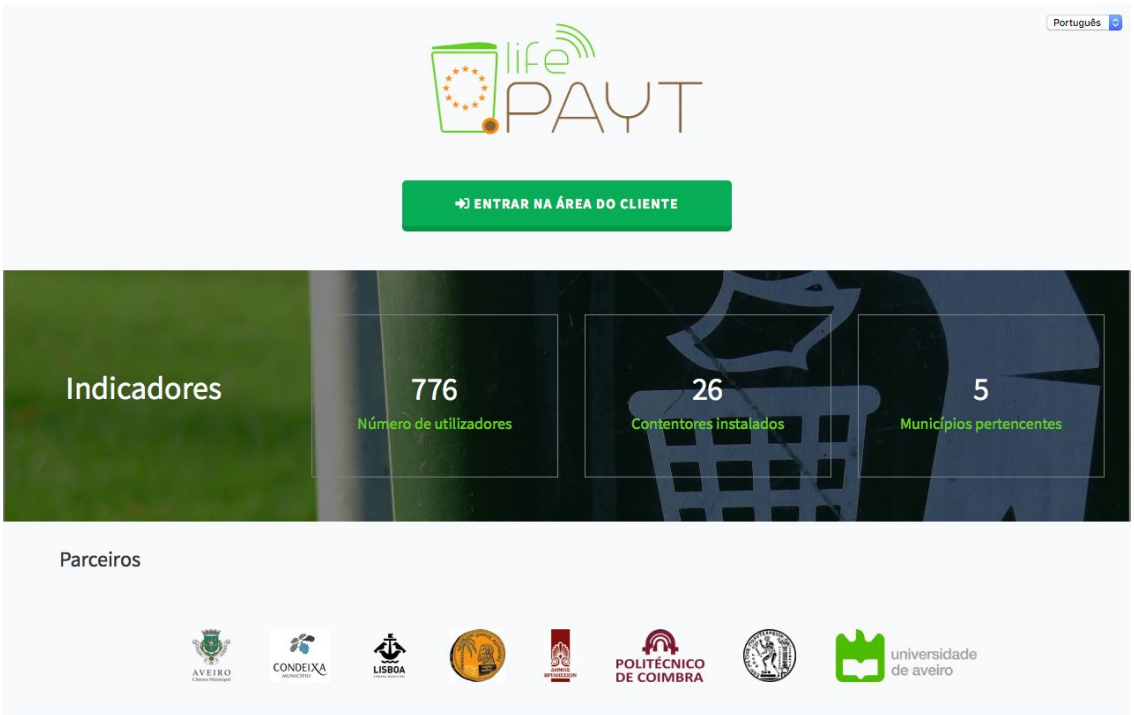

Figura 4.2: Página com acesso a utilizadores não autenticados

Outro dos requisitos era que a interface tivesse um design responsivo para que os utilizadores tivessem sempre uma boa visualização do conteúdo em qualquer dispositivo,

sem que o tamanho do seu ecrã tivesse importância.

Para satisfazer este requisito foi utilizada a biblioteca Bootstrap [3] que possui um conjunto de ferramentas para construção de interfaces responsivas. O Bootstrap já inclui vários modelos para os mais diversos componentes que possam existir numa página web, nomeadamente botões, formulários, tabelas.

A interface desenvolvida tira partido do sistema de grelha que o Bootstrap utiliza para que todos os seus conteúdos se adaptem quando o tamanho do ecrã é alterado. A partir da terceira versão, o Bootstrap inclui classes predefinidas para construção de layouts para diferentes tipos de dispositivos tais como telemóveis, tablets, computadores portáteis, etc. Por exemplo, é possível usar a classe .col-xs-* para criar colunas para dispositivos pequenos como o telemóvel, ou então a classe .col-sm-* para dispositivos um pouco maiores como tablets. Para ecrãs maiores pode-se usar as classes .col-md-* ou .col-lg-* [42].

Para além do sistema de grelha, o Bootstrap inclui um extenso catálogo de ícones para ajudar à usabilidade. Por exemplo, num botão, a presença de um ícone ajuda à compreensão da ação que esse botão irá desencadear. De uma forma semelhante, a presença de um ícone junto de um valor ajuda o utilizador a perceber a que é que esse valor se refere.

Todas as interfaces referentes à área pessoal têm como base uma estrutura como a da fig. 4.3, sendo constituída por um cabeçalho, uma barra lateral e um rodapé. O cabeçalho é composto pelo nome do projeto, que remete o utilizador para a página oficial do projeto caso o utilizador clique nele, por um botão que faz com que a barra lateral colapse, por um menu que permite a troca do idioma apresentado na página e finalmente, por um botão utilizado para o utilizador sair da sua área pessoal. A barra lateral é composta por dois elementos comuns a todos os utilizadores, um botão que permite aceder a uma área onde o utilizador pode alterar algumas informações relativas à sua conta, nomeadamente o seu e-mail e a sua palavra-passe. O outro elemento é um botão que redireciona o utilizador para um manual de ajuda da aplicação onde pode consultar as ações que pode fazer e o que tem de fazer para atingir essa mesma ação. O último bloco desta estrutura base é o rodapé que apenas apresenta a licença a que está sujeita a aplicação, neste caso a CC-BY (Atribuição). A descrição em detalhe de cada tipo de interface irá ser feita de seguida.

[3] https://getbootstrap.com/

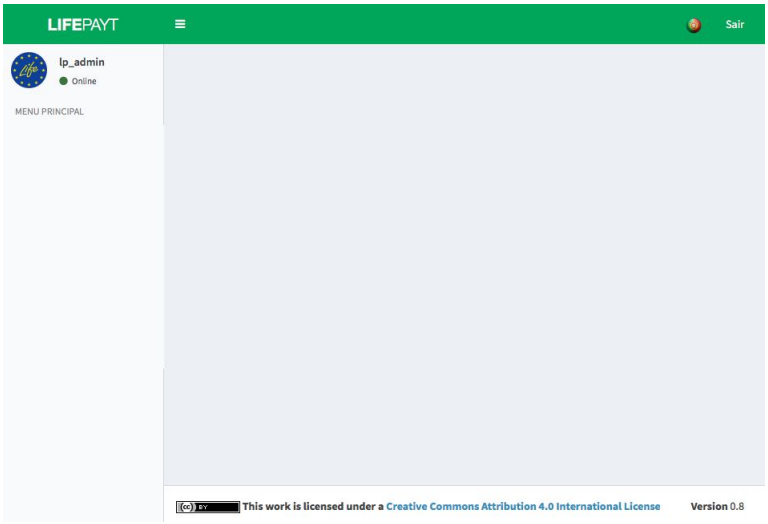

Figura 4.3: Estrutura base da interface

Existem 3 tipos de interface, dependendo do papel do utilizador no sistema: a interface do administrador do sistema, do responsável do município, e do cliente. Dentro das categorias de responsável pelo município e do cliente também existe distinção entre as interfaces apresentadas, i.e., os dados que são apresentados dependem do município a que estão associados.

Interface do Cliente – Pequenos Produtores

São considerados pequenos produtores aqueles que colocam os resíduos produzidos em contentores fora das suas casas/negócios e que têm um cartão RFID associado a si. Depois de ser redirecionado para a sua interface, o utilizador terá acesso a várias informações acerca da morada associada ao seu perfil.

A fig. 4.4 representa a interface que o utilizador terá acesso. Nela poderá observar alguma da sua informação pessoal, tal como o seu nome, morada, entidade responsável pela recolha de resíduos e algumas das suas conquistas. Relativamente às faturas, o utilizador terá, de imediato, acesso aos valores da fatura real e simulada do último mês. Caso o utilizador clique no botão "Mais informação", terá acesso a informação mais detalhada sobre as faturas. Para além de informação acerca dos valores das faturas, o utilizador tem a possibilidade de comparar a sua produção de resíduos do último mês com a média da sua região e perceber como é que esse valor é qualificado em termos ambientais. No final da página o utilizador tem acesso a um gráfico que apresenta os valores correspondentes à produção de resíduos ao longo dos últimos 12 meses.

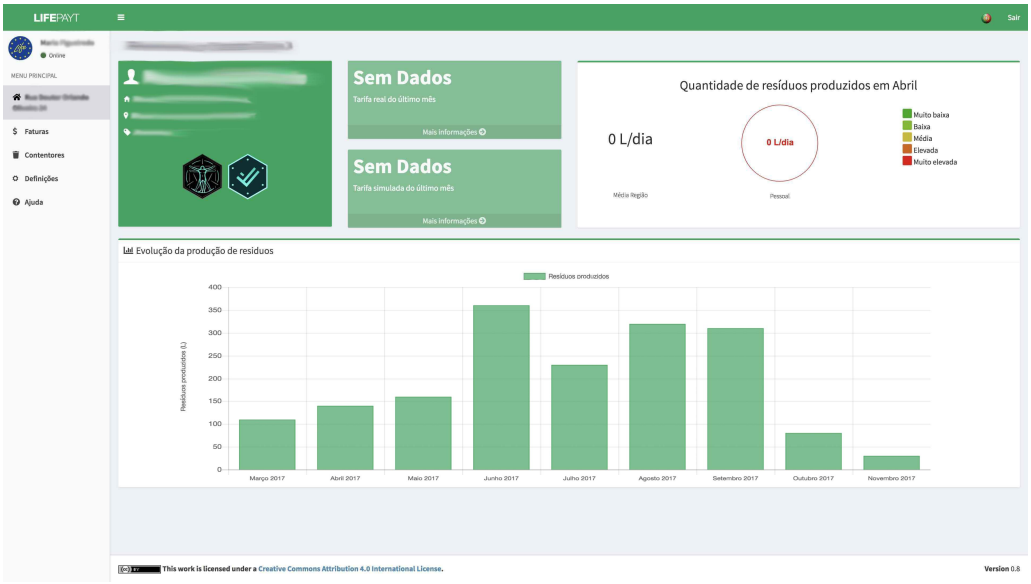

Figura 4.4: Página inicial da interface do utilizador (pequeno produtor)

Quando o utilizador clica no botão “Mais informação” ou no separador "Faturas"será apresentada a informação presente na fig. 4.5. Aqui, o utilizador tem a possibilidade de comparar os valores das faturas reais com as simuladas. No topo, o utilizador tem de imediato uma vista sobre os valores das faturas do último mês. Logo abaixo desses valores são apresentadas as fórmulas usadas para calcular essas mesmas faturas. Abaixo existem duas vistas para a mesma informação, a relação entre o valor das faturas nos últimos meses. Tanto a tabela como o gráfico apresentam os meses referentes ao intervalo que pode ser selecionado pelo utilizador no botão que se encontra no topo da página.

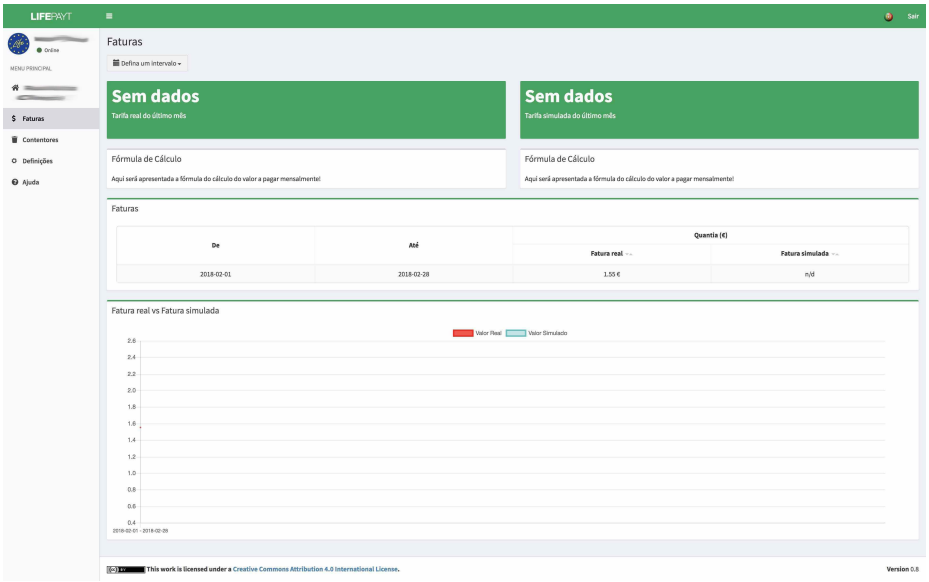

Figura 4.5: Separador referente à informação detalhada sobre as faturas

Para além deste separador referente a uma morada do utilizador, este pode navegar para o separador "Contentores" (fig. 4.6) onde tem acesso a um mapa interativo onde se encontram as localizações de todos os contentores do seu município.

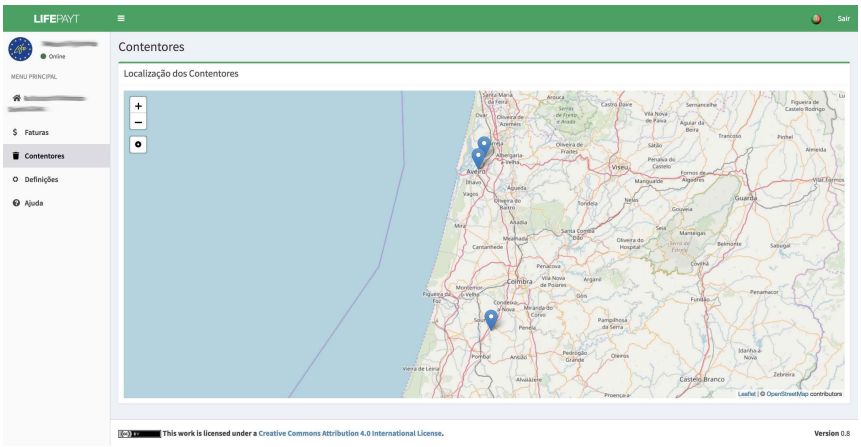

Figura 4.6: Separador referente ao mapa de localizações dos contentores

Por último, o utilizador tem a possibilidade de aceder ao separador das definições (fig. 4.7) já apresentado anteriormente aquando da descrição da estrutura base da aplicação. Para além das funcionalidades aí descritas, o utilizador pode visualizar uma tabela com as moradas a si associadas, podendo executar duas ações sobre essas mesmas moradas. Uma delas é a possibilidade de alterar o nome dessa morada, ou seja, caso o utilizador pretenda pode dar um nome que lhe facilite a identificação de que morada se trata (ex: Casa de férias) sendo que depois este irá ser o nome que vai aparecer na barra lateral. Para além disso o utilizador pode esconder uma morada, i.e. fazer com que uma certa morada não apareça na barra lateral nem seja utilizada nos cálculos dos totais das faturas e quantidade de resíduos produzidos. Ao esconder uma morada, não faz com que ela deixe de estar associada a si nem que a quantidade de resíduos nela produzidos não seja contabilizada, apenas terá impacto na interface visual.

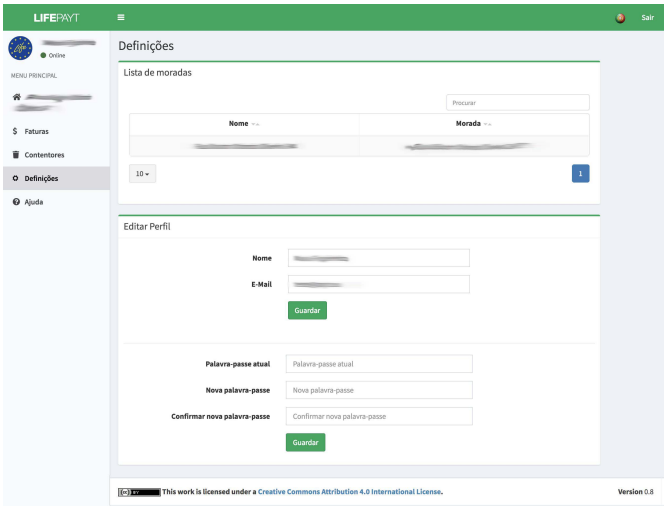

Figura 4.7: Separador referente às definições de conta do utilizador

Interface do Cliente – Grandes Produtores

São considerados grandes produtores aqueles que tenham contratado um serviço de recolha de resíduos nas suas instalações. O primeiro separador a que o utilizador tem acesso é muito semelhante ao que os pequenos produtores têm acesso, apenas com duas diferenças. O gráfico da evolução (fig. 4.8) de resíduos passa a apresentar a quantidade de resíduos produzidos de acordo com o seu tipo (indiferenciados, orgânicos, papel/cartão, plástico e vidro) ao longo dos meses.

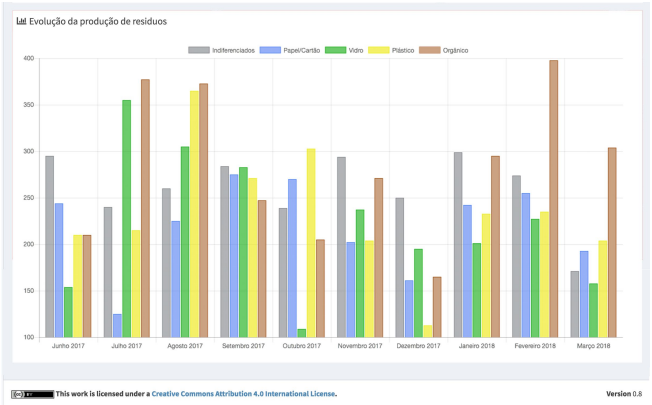

Figura 4.8: Gráfico da evolução de resíduos por tipo

Para além desta diferença, o componente onde o utilizador pode fazer comparações com a média da região tem um botão que apresenta ao utilizador um novo conjunto de informação detalhada (fig. 4.9) acerca dessas comparações. Deste conjunto de informações destacam-se três novos componentes: um que apresenta comparações consigo próprio num determinado período de tempo (selecionável dentro de uma lista

predefinida), um gráfico de barras sobrepostas que representa a taxa de apresentação à remoção dos contentores de cada tipo de resíduos e um gráfico circular relacionado com a taxa de separação de cada tipo de resíduos e que no centro apresenta a taxa de separação geral.

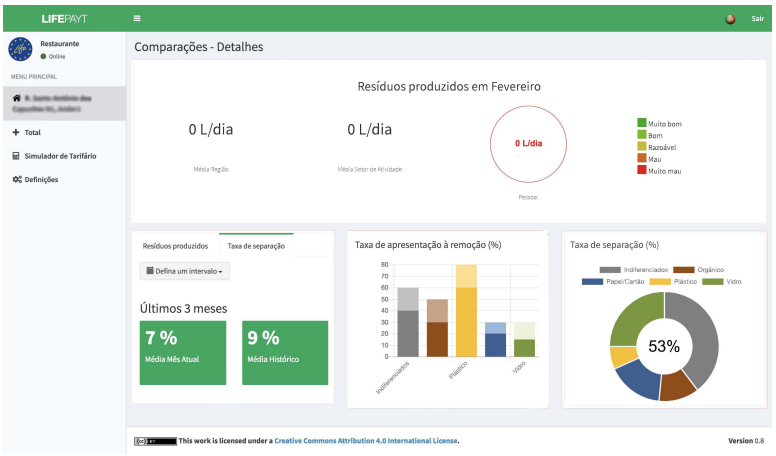

Figura 4.9: Vista detalhada sobre comparações

A interface destinada aos grandes produtores tem um separador (fig. 4.10) destinado a informações acerca de todas as moradas associadas a si. Nesse separador o utilizador poderá consultar o valor total das tarifas reais e simuladas do último mês, bem como o gráfico que mostra a relação entre o valor real e o simulado ao longo dos últimos 6 meses. Por fim, do lado esquerdo, tem à disposição um mapa interativo onde se encontram as moradas das suas instalações.

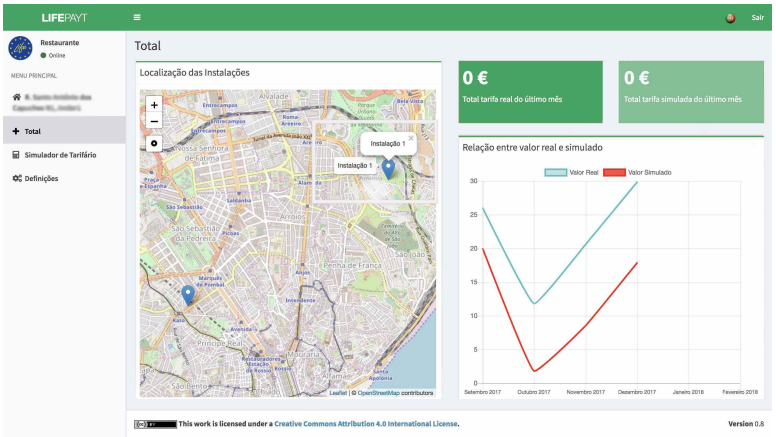

Figura 4.10: Vista sobre separador Total

Caso pretenda obter mais informações sobre uma das suas instalações, o utilizador pode selecionar no mapa a instalação pretendida. Após clicar na instalação, irão ser apresentados dois novos componentes (fig. 4.11) Um deles serve para informar o

utilizador do número total de contentores de cada tipo de resíduos que estão presentes nessa instalação, bem como a data da última recolha. O outro componente é um gráfico que representa o número de contentores atribuídos de acordo com a sua capacidade e com o tipo de resíduos. Este gráfico é interativo e permite esconder os tipos de resíduos bastando para isso que clique em cima da respetiva legenda presente no topo do componente. Os dados associados ao gráfico podem ser descarregados em formato JSON clicando no botão "Download" que se encontra no canto superior direito.

Figura 4.11: Vista sobre detalhes no separador Total

Para além das funcionalidades descritas anteriormente, o utilizador tem acesso a um separador (fig. 4.12) onde está presente um simulador de tarifas que permite simular vários cenários e obter o valor que iria pagar nesse cenário. Graças a este simulador o utilizador pode ajustar o número de contentores de cada capacidade e tipo de resíduos de forma a pagar menos sem que isto afete o seu negócio. Após preencher o formulário basta clicar no botão "Calcular" e o valor simulado irá aparecer por baixo do simulador.

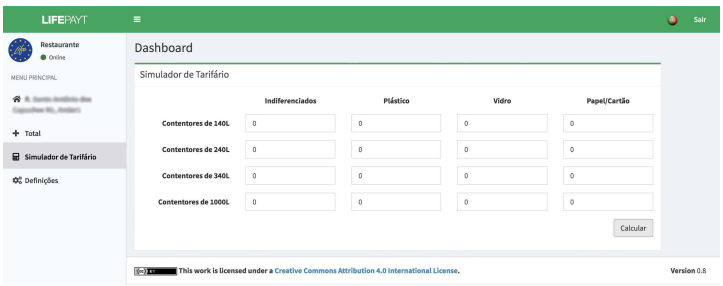

Figura 4.12: Vista sobre simulador de tarifas

Tal como a interface para os pequenos produtores também esta apresenta um separador de definições com as mesmas funcionalidades.

Interface do Município – Pequenos Produtores

Assim com os munícipes, também os responsáveis pelo município têm acesso a uma interface personalizada para poderem observar alguns indicadores úteis para a monitorização do seu serviço de recolha de resíduos. No separador "Estatísticas Gerais" (fig. 4.13) um utilizador com o cargo de responsável do município tem a possibilidade de observar um conjunto de indicadores relativos ao seu município. Dos quais destacam-se os valores totais das tarifas reais e simuladas do último mês, o número de utilizadores ativos no sistema (munícipes que já tenham utilizado esta aplicação), dois indicadores das preferências dos clientes e por último, o mês com maior produção de resíduos. Por baixo desses indicadores é possível visualizar um gráfico de barras que representa a evolução da produção de resíduos no município ao longo dos últimos 12 meses.

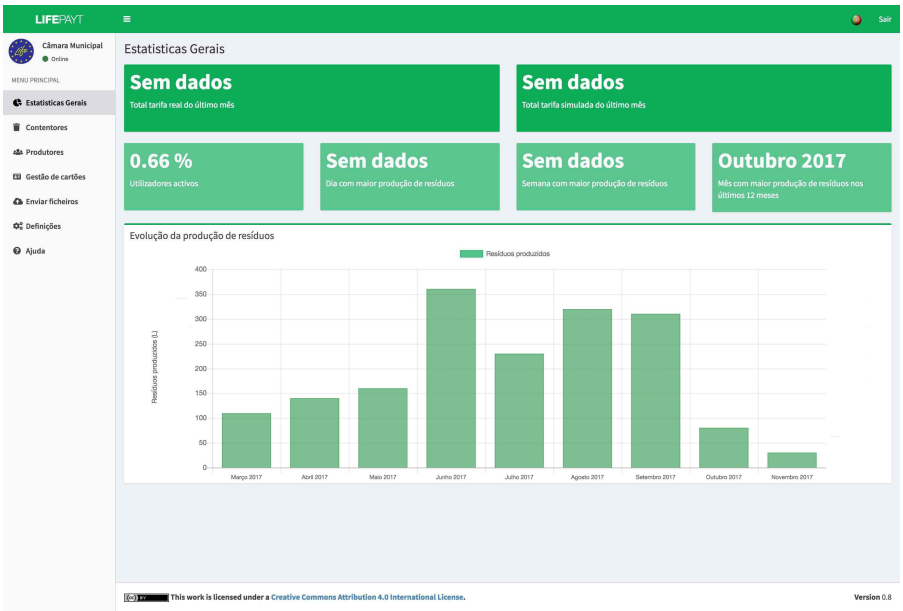

Figura 4.13: Vista do separador com as estatísticas gerais do município

Caso o utilizador queira visualizar informações relativas aos contentores instalados no município pode aceder ao separador "Contentores" (fig. 4.14). Aqui, terá acesso a um mapa interativo com as localizações de todos os contentores. Ao clicar numa dessas localizações é apresentada informação sobre a utilização desse mesmo contentor sob a forma de gráfico. Mais a baixo, é apresentada a informação acerca da utilização de cada um dos contentores sob a forma de tabela. É possível, para cada contentor, observar a sua utilização em diferentes períodos de tempo.

Na lista de separadores segue-se a área reservada à apresentação de informação sobre os produtores do município (fig. 4.15), sob a forma de tabela. Para além do nome, e-mail e morada também é possível verificar o estado desse produtor na plataforma

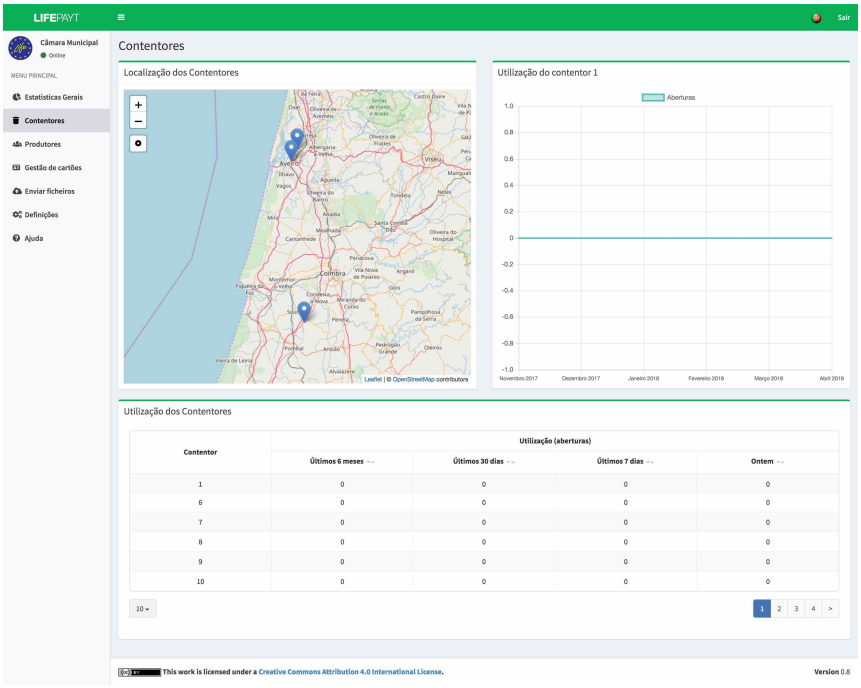

Figura 4.14: Separador com as estatísticas sobre os contentores instalados no município

(ativo/desativo/banido), a data do último acesso à plataforma e os cartões RFID que tem associados a si. Caso o utilizador pretenda encontrar um produtor em específico pode tirar partido da caixa de pesquisa que se encontrar por cima da tabela. Para ter acesso à informação relativa aos cartões é necessário clicar em "Gerir cartões".

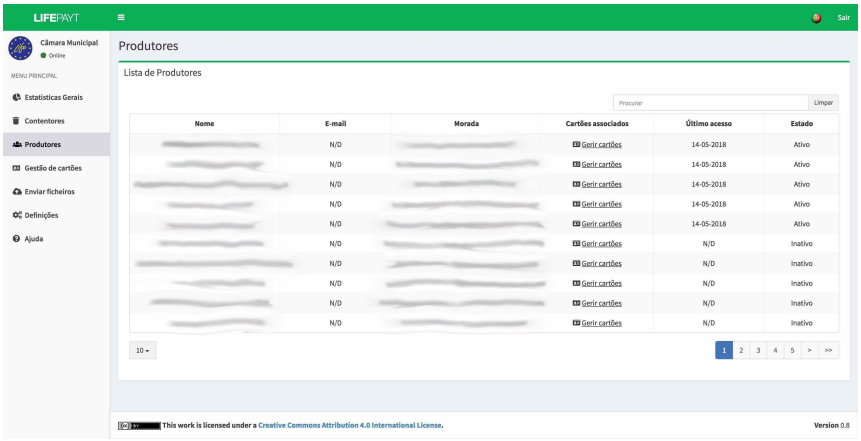

Figura 4.15: Separador com informação dos produtores de municípios

Após clicar em "Gerir cartões" é apresentado o ecrã representado na fig. 4.16. Aqui pode ser feita toda a gestão dos cartões associados a um produtor. É possível adicionar mais cartões, alterar os já existentes ou mesmo remover cartões a si associados.

Figura 4.16: Informação relativa aos cartões associados a um produtor

Para que o utilizador possa gerir os cartões presentes no sistema existe o separador "Gestão de cartões" (fig. 4.17), composto por uma tabela com a listagem de todos os cartões do sistema. Para além do número do cartão é possível ver o utilizador associado ao mesmo e um histórico de todas as ações relacionadas com esse cartão e o momento em que foram realizadas.

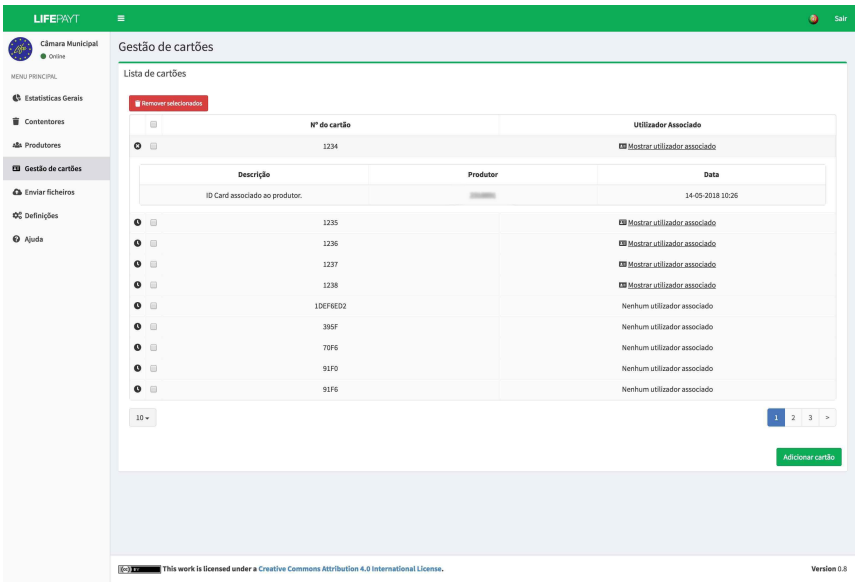

Figura 4.17: Vista sobre o separador responsável pela gestão dos cartões

Também é possível adicionar um novo cartão ao clicar em "Adicionar cartão" que se encontra abaixo da tabela. Ao clicar em "Mostrar utilizador associado" é apresentada a informação sobre o utilizador que possui aquele cartão (fig. 4.18).

Figura 4.18: Informação relativa ao produtor associado a um cartão

O separador "Envio de ficheiros" é o espaço onde é possível carregar ficheiros para a plataforma (fig. 4.19). Existem dois componentes distintos para carregar os ficheiros, ambos no formato XLS. Um está reservado ao envio de um ficheiro com a lista atual dos clientes e o outro está reservado ao envio de ficheiros com a lista dos valores cobrados no mês anterior a cada cliente. Estes ficheiros serão processados pela plataforma e irão servir para atualizar as bases de dados refletindo-se na atualização dos dados que podem ser visualizados nas interfaces dos utilizadores.

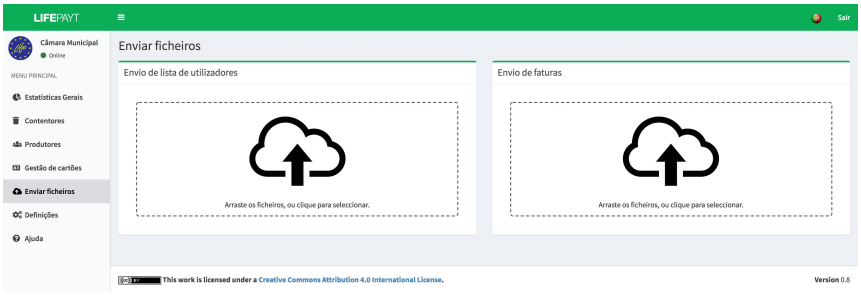

Figura 4.19: Espaço para carregar ficheiros

Interface do Município – Grandes Produtores

No separador "Estatísticas Gerais" (fig. 4.20) o utilizador tem acesso a um conjunto de indicadores relativos ao seu município, tais como valores totais das tarifas reais e simuladas do último mês, clientes com maior/menor produção de resíduos e clientes com maior/menor taxa de separação. Para além destes indicadores é possível visualizar um gráfico de barras que representa a evolução da produção de resíduos por tipo de material recolhido no município ao longo dos últimos 12 meses.

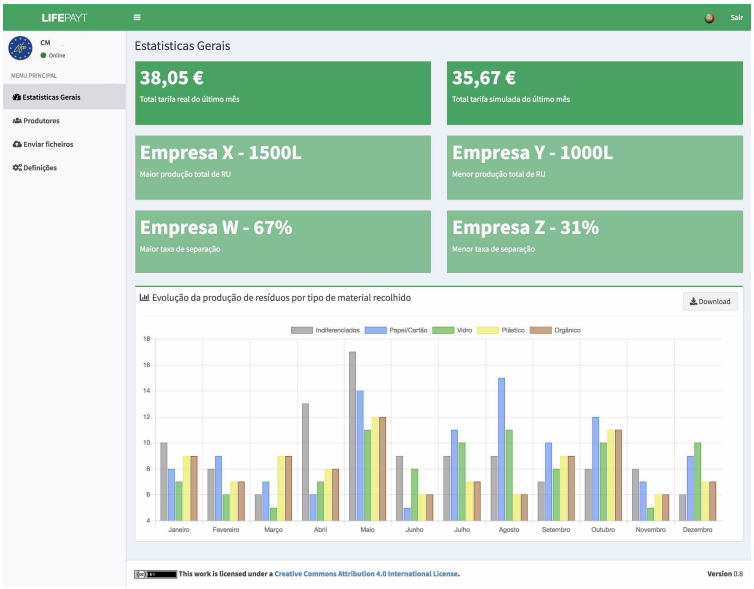

Figura 4.20: Vista sobre as estatísticas gerais do município

No separador seguinte (fig. 4.21) encontra-se um mapa com as localizações de todos os produtores do município. Caso pretenda obter mais informações sobre um dos produtores, o utilizador pode selecionar no mapa o produtor pretendido. Após clicar no produtor, irão ser apresentados dois novos componentes. Um deles informa o número total de contentores de cada tipo de resíduos que estão presentes nesse produtor, bem como a data da última recolha. O outro componente representa o número de contentores atribuídos de acordo com a sua capacidade e com o tipo de resíduos. Este gráfico é interativo e permite esconder os tipos de resíduos bastando para isso que clique em cima da respetiva legenda, presente no topo do componente. Os dados associados ao gráfico podem ser descarregados em formato JSON clicando no botão "Download" que se encontra no canto superior direito.

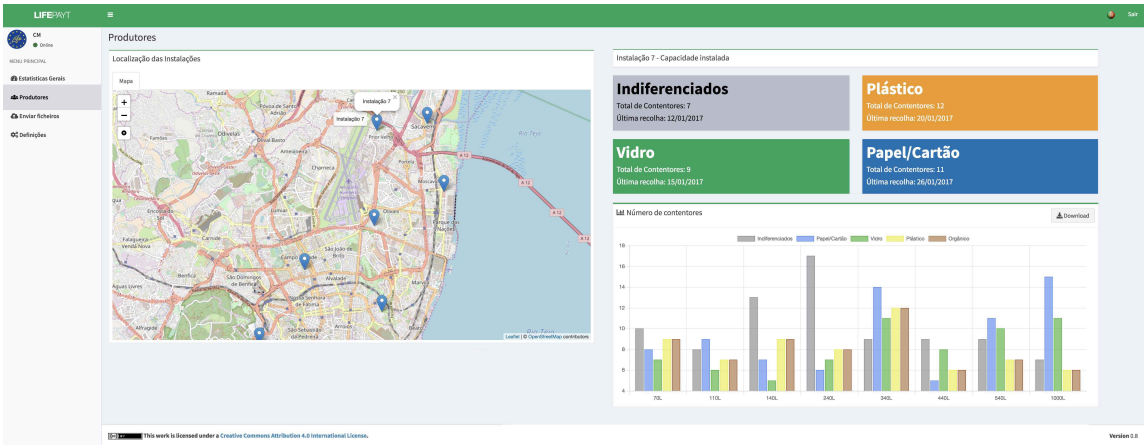

Figura 4.21: Vista sobre a informação relativa aos produtores

Assim como na interface dos municípios relativos aos pequenos produtores, também esta tem um separador dedicado ao envio de ficheiros relativos à lista de clientes e às faturas cobradas aos utilizadores do sistema.

Interface do Administrador

Por último, existe uma interface reservada apenas aos responsáveis pela gestão de todo o sistema. Através dela o administrador tem a possibilidade de visualizar o estado do sistema. No primeiro separador (fig. 4.22) é possível observar as últimas tentativas de login na plataforma bem como se estas foram aprovadas ou recusadas.

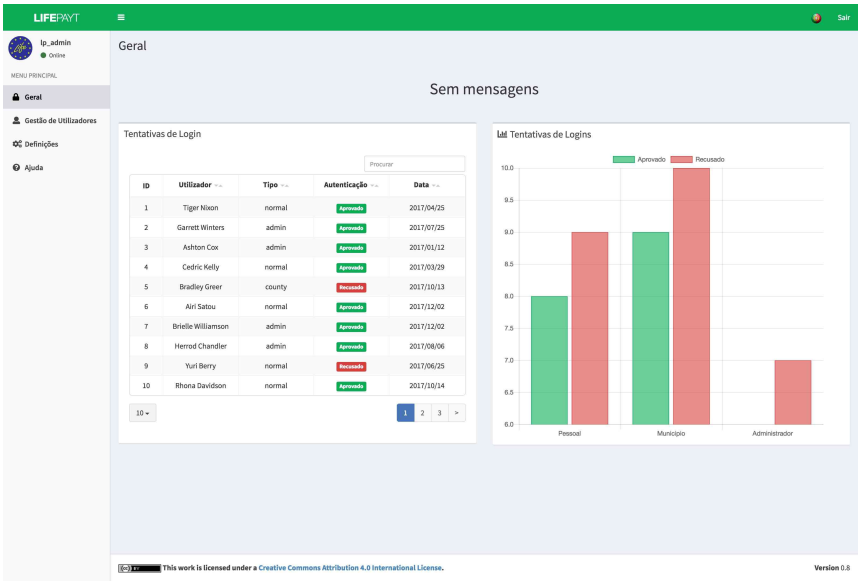

Figura 4.22: Vista sobre as últimas tentativas de login

No separador "Gestão de Utilizadores" (fig. 4.23) são apresentadas ao administrador duas tabelas, uma relativa aos utilizadores e outra com os atuais cargos existentes internamente no sistema e número de utilizadores com esse mesmo cargo.

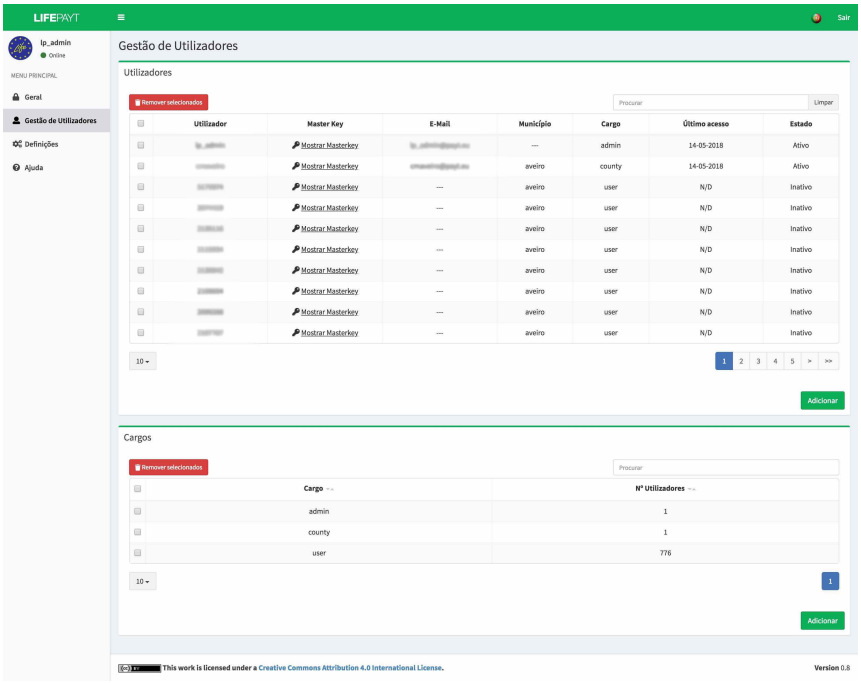

Figura 4.23: Vista sobre o separador responsável pela gestão dos utilizadores

A tabela dos utilizadores permite observar informações relativas a cada um dos utilizadores registados. Dessa informação destaca-se a chave-mestra que é utilizada para repor a palavra-passe em caso de esquecimento da mesma por parte de um utilizador. Para poder visualizar a chave-mestra (fig. 4.24) basta clicar em "Mostrar Masterkey". É também nesta tabela que o administrador pode adicionar/remover utilizadores do sistema. Caso pretenda remover utilizadores terá que selecionar os que pretende e clicar em "Remover selecionados", presente no topo da tabela. Para adicionar, tem à sua disponibilidade o botão "Adicionar" que abre um formulário que terá de ser preenchido com algumas informações necessárias para o bom funcionamento do resto do sistema.

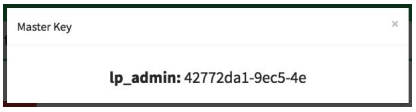

Figura 4.24: Vista sobre a chave-mestra associada a um utilizador

Processo de registo e início de sessão

Como já referido anteriormente, existe uma página inicial sem qualquer tipo de controlo de acesso e uma página pessoal à qual só utilizadores autenticados podem aceder. É importante então perceber de que forma é que um utilizador é registado no sistema e de que forma se pode autenticar. Como esta plataforma apenas é destinada a cidadãos pertencentes às áreas onde o projeto irá ser implementado, o processo de

registo não pode ser feito por qualquer pessoa. A solução implementada possui dois mecanismos de registo de utilizadores, um deles manual e outro automático com recurso ao processamento de um ficheiro XLS com uma lista de clientes.

O mecanismo manual é apenas utilizado pelo administrador do sistema, recorrendo à área de gestão de utilizadores presente na sua interface dedicada (fig. 4.25). Este tipo de registo é feito apenas em situações muito específicas, nomeadamente para registo de outros utilizadores com o cargo de administrador ou para utilizadores responsáveis pelo município que também têm associada a si uma interface diferente da dos clientes/cidadãos.

Figura 4.25: Processo de registo de novos utilizadores através da interface do administrador

O segundo mecanismo é o utilizado pelos responsáveis do município que para isso, terão de carregar, através da sua interface, um ficheiro no formato XLS contendo a lista atual de clientes (fig. 4.26). Este foi o formato escolhido depois de saber que alguns dos *stakeholders* já possuem a informação necessária neste mesmo formato. Após o carregamento do ficheiro para a plataforma, este é processado do lado do servidor que extrai a informação necessária ao registo, ignorando toda a informação adicional. Tendo em conta que a lista enviada contém todos os clientes e sendo que alguns já possam existir no sistema, antes de um utilizador ser registado é feita uma verificação se este já se encontra registado. Caso exista é descartado, caso não exista procede-se ao seu registo. Depois de processado, o ficheiro é eliminado do servidor e o utilizador que o carregou é notificado de que o processo de registo foi concluído.

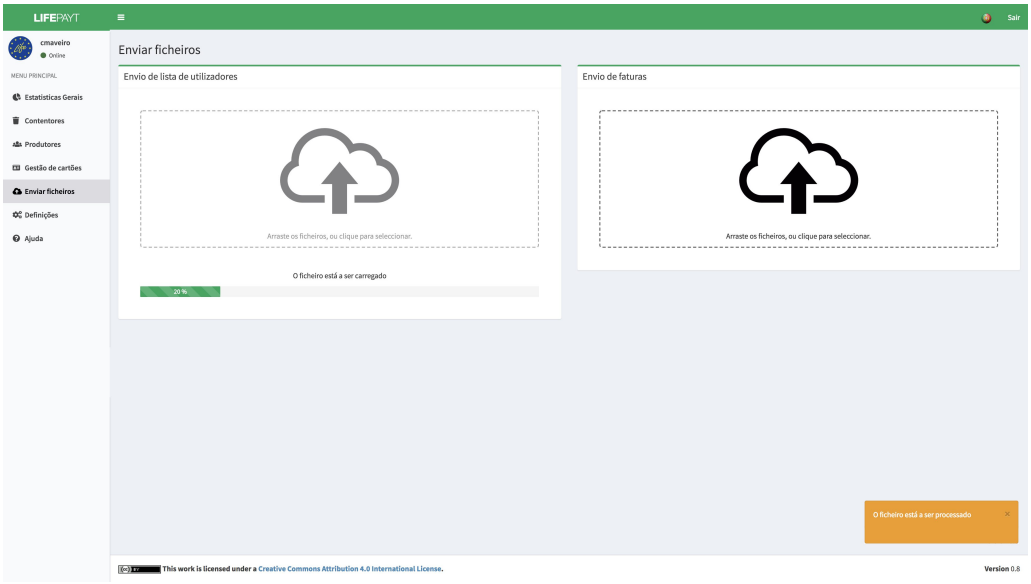

Figura 4.26: Processo de registo de novos utilizadores através de um ficheiro

Depois de registados, os utilizadores serão informados que a sua conta já se encontra disponível juntamente com as credenciais de acesso. Após este momento cada utilizador já tem a possibilidade de se autenticar na aplicação e poder aceder à sua interface pessoal. Na primeira vez que aceder ser-lhe-á apresentado um formulário para introdução de informação adicional (ex: e-mail). Nesse momento também será obrigatório alterar a sua palavra-passe por razões de segurança.

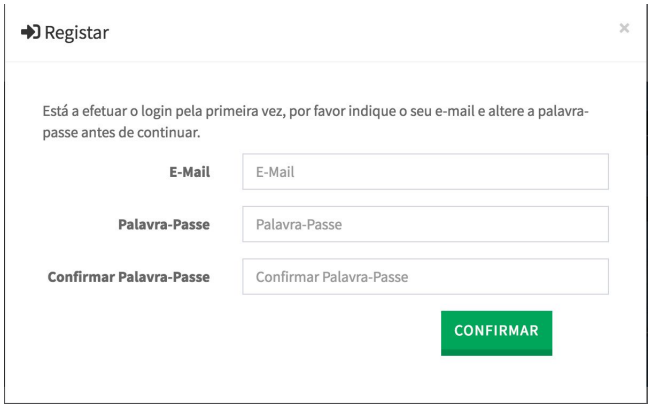

Figura 4.27: Formulário para adicionar informação adicional

Após o formulário da fig. 4.27 ser preenchido o utilizador é redirecionado para a sua interface pessoal.

Processo de obtenção de dados do lado do servidor

A comunicação com a base de dados para obter os valores que irão ser apresentados na interface é feita através de *websockets* juntamente com o protocolo Simple Text Oriented

Messaging Protocol (STOMP). Este é um protocolo baseado em texto que permite que um cliente STOMP possa comunicar com um *broker* de mensagens que suporte este protocolo não dependendo da linguagem para a qual o broker foi desenvolvido. Neste caso em particular a mensagem é estruturada da seguinte forma:

```
payt.<serviço>.db.<método>, <argumentos>
```

Dependendo de vários fatores, nomeadamente as condições de rede, o valor proveniente da base de dados pode demorar algum tempo até estar disponível. Para que a interface não fique bloqueada enquanto o valor não está disponível, a comunicação com a base de dados é feita de forma assíncrona. Para dar a entender ao utilizador que o valor ainda está a ser obtido da base de dados, existe uma animação que fica a rodar até o valor ser obtido. Quando o valor é obtido, o estado interno do componente é atualizado, parando a animação e apresentando o valor obtido.

Processo de reposição de palavra-passe

Caso o utilizador se esqueça da sua palavra-passe atual tem a possibilidade de a repor. Para isso, no formulário para entrar na plataforma tem uma opção "Repor palavra-passe" que apresentará um formulário diferente (fig. 4.28) onde são pedidos o nome de utilizador e a sua respetiva chave-mestra. Caso os dados preenchidos coincidam com os presentes na base de dados de autenticação, a palavra-passe associada a esse utilizador voltará a ser a chave-mestra. Após a reposição ser feita, na primeira vez que o utilizador iniciar sessão ser-lhe-á pedido novamente que altere a palavra-passe.

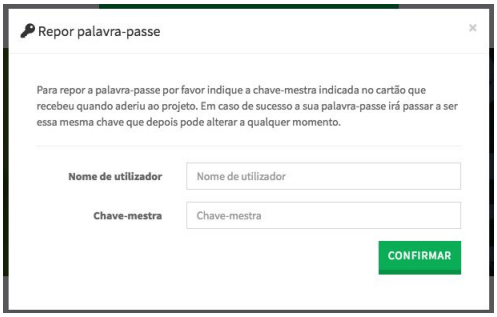

Figura 4.28: Formulário para repor a palavra-passe

Internacionalização

Tratando-se este de um projeto com dois municípios não portugueses, torna-se de extrema importância a existência de uma forma de consultar a interface noutros idiomas.

Para além do Português, Grego e Inglês, a interface deve suportar a inserção de novos idiomas sem ter de alterar o código para que isso aconteça.

Para satisfazer este requisito, recorreu-se a uma forma de internacionalização com base em objetos JSON, cada um definindo um dicionário no respetivo idioma. Graças ao padrão Flux é possível a atualização de todo o texto quando o utilizador altera o idioma da interface. Este padrão [43] carateriza-se pelo fluxo de dados unidirecional (fig. 4.29), ou seja, os dados fluem na aplicação apenas num sentido.

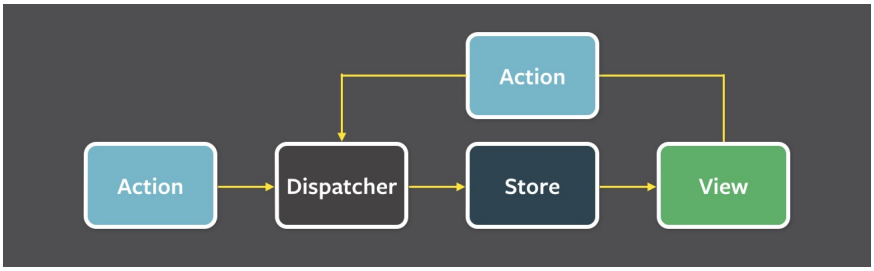

Figura 4.29: Fluxo de dados referente ao padrão Flux [43]

Toda a informação flui através do *Dispatcher* que funciona como um ponto central de ligação. As ações são passadas para o *Dispatcher* com a ajuda de um método criador das mesmas, maioritariamente originadas de interações do utilizador com a interface. De seguida o *Dispatcher* invoca todos os *callbacks* que foram registados pelas *Stores*, originando ações dentro das mesmas. Consoante a ação, as *Stores* procedem às alterações que a si dizem respeito, se elas existirem. Após as alterações é emitido um evento sinalizando que ocorreram alterações no seu estado. Os controladores que estão associados a essas *Stores*, ou seja, utilizam os dados nelas presentes, invocam os próprios métodos de atualização e assim causam a sua própria atualização e de todos os seus descendentes na árvore de componentes.

O padrão acima descrito vem resolver o problema de mudança de idioma na interface desenvolvida. Sempre que o utilizador altera o idioma recorrendo ao menu que se encontra no topo da interface, desencadeia uma ação que é passada para o *Dispatcher* sinalizando que ocorreu uma troca de idioma. Este, por sua vez, faz com que a *Store* responsável pelo armazenamento de todos os blocos de texto utilizados na interface proceda à alteração desses mesmos blocos pelos referentes ao idioma selecionado pelo utilizador. Essa alteração faz-se através do carregamento do objeto JSON referente ao idioma escolhido. Após o carregamento ser finalizado, a *Store* emite um evento sinalizando que possui os novos blocos de texto e consequentemente os componentes que os utilizam atualizam o seu estado interno e isso reflete-se na atualização visual do texto apresentado.

```
export default {
   AdminSideBar: {
      mainMenu:'MENU PRINCIPAL',
      general:'Geral',
      userManagement:'Gestão de Utilizadores'
   },
   RowAddress: {
      realTitle:'Tarifa real do último mês',
      simulTitle:'Tarifa simulada do último mês',
      labelKnob:'Pessoal',
      unit:' €/mês'
   }
}
```

```
export default {
   AdminSideBar: {
      mainMenu: 'MAIN MENU',
      general: 'General',
      userManagement: 'User Management'
   },
   RowAddress: {
      realTitle:"Last month's real bill",
      simulTitle:"Last month's simulated bill",
      labelKnob:'Personal',
      unit:' €/month'
   }
}
```

Bloco de código 1: Excerto dos ficheiros referentes aos idiomas português e inglês

De acordo com o explicado anteriormente torna-se fácil de perceber que caso seja preciso adicionar outro qualquer idioma, só é necessário criar um novo ficheiro JavaScript Object Notation (JSON) com a estrutura igual aos já presentes, apenas substituindo o texto lá presente pelo texto no idioma pretendido. O bloco de código 1 representa um excerto dos ficheiros referentes aos idiomas português e inglês, dando assim uma ideia do que foi descrito em cima.

Processo de obtenção de dados de fontes externas

No caso específico do município de Aveiro existem duas fontes externas que fornecem dados para a nossa plataforma: os ficheiros XLS enviados pelos responsáveis do município e a API desenvolvida pela empresa Citibrain [4]. O modo como estas fontes interagem com a plataforma será descrito de seguida.

Para além do ficheiro com a lista de clientes (utilizado para o processo de registo de novos clientes), os responsáveis pelo município tem a possibilidade de carregar ficheiros referentes às tarifas reais cobradas, no mês anterior, aos cidadãos. Este ficheiro é carregado através da interface criada para o efeito, sendo posteriormente processado e eliminado do servidor. Após o seu processamento, os novos valores das tarifas já estarão disponíveis para consulta na sua interface, bem como na área pessoal de cada cidadão.

Para além da lista de clientes e dos valores cobrados a eles em cada mês, é necessário saber os seus consumos atuais agora no sistema PAYT, a informação sobre os cartões associados aos utilizadores e sobre os contentores instalados na zona piloto. Esta informação chega à plataforma através da API REST disponibilizada pela Citibrain. Para poder obter essa informação foi criado um módulo que diariamente sincroniza as bases de dados internas com a informação proveniente da API.

[4] http://www.citibrain.com/pt/

A autenticação na API é efetuada com base em JSON Web Tokens (JWT), gerando um *token* que tem de ser enviado nos cabeçalhos de cada pedido. Este *token* tem um tempo de expiração de 30 minutos, mas pode ser atualizado recorrendo a um pedido efetuado ao *endpoint* criado para o efeito, com o *token* antigo. Depois de obtido o *token*, podem ser efetuados todos os pedidos para sincronizar a informação.

Um pedido efetuado à API segue a seguinte estrutura, onde é possível filtrar os resultados de acordo com alguns parâmetros.

```
GET /api/usage/?container=5 HTTP/1.1
Host: payt-api.citibrain.com
Authorization: JWT <token>
Cache-Control: no-cache
```

O exemplo acima representa o pedido efetuado para obter todas as utilizações do contentor 5. Em resposta ao pedido recebemos um objeto JSON com a seguinte estrutura.

```
{
   "count": 3,
   "next": null,
   "previous": null,
   "results": [{
         "count": 1,
         "container": 5
         "date": "2017-10-09",
         "consumer": "1234"
      },
      {
         "count": 5,
         "container": 5,
         "date": "2017-10-10",
         "consumer": "1234"
      },
      {
         "count": 25,
         "container": 5,
         "date": "2017-10-10",
         "consumer": "1235"
      }]
}
```

4.3 PORTAL DE DADOS ABERTOS

Outra das vertentes deste projeto é o portal para dados abertos. É através deste portal que a informação produzida pela plataforma poderá ser consultada pelos cidadãos em geral. A informação vai ser apresentada neste portal de forma anonimizada e agregada para que não seja possível relacionar um perfil de produção de resíduos com um produtor em específico. Com este portal pretende-se disponibilizar os dados recolhidos e processados pela plataforma, de uma forma aberta e de fácil visualização, para que outras pessoas possam reutilizar os dados nos seus trabalhos/investigações e deste modo tirar o máximo proveito destes mesmos dados.

As soluções já existentes de catálogos de dados descritas na secção 2.5.3 são muito semelhantes, pelo que a escolha de utilizar o CKAN em detrimento das outras soluções se prendeu pelo fato de já ser utilizado pelos países envolvidos no projeto, o que facilita a integração entre o nosso sistema e os portais já existentes em alguns municípios. Para além disso o CKAN é uma solução de código aberto, um dos requisitos obrigatórios neste projeto, a utilização apenas de tecnologias de código aberto. Para concluir, o CKAN permite ter um portal a funcionar corretamente em pouco tempo e as APIs fornecidas são bastante intuitivas e de fácil utilização. Foi por todos estes motivos que a escolha foi o CKAN.

4.3.1 IMPLEMENTAÇÃO

De acordo com a fig. 3.6 existem 2 componentes relacionados com o portal de dados abertos: o exportador dos conjuntos de dados e a instância CKAN para a visualização dos mesmos. Seguindo a mesma ideia dos restantes componentes do sistema, a instância CKAN também foi encapsulada num *container* Docker de modo a ter um ambiente isolado apenas responsável pela sua execução.

Para efeitos de simplificação e maior celeridade na construção da plataforma foi utilizado o ficheiro Docker Compose fornecido pelos criadores do CKAN que oferece a possibilidade de ter uma instância limpa, i.e sem extensões, mas que continua a permitir a adição das mesmas. Para ter a instância pronta a ser lançada apenas são precisas algumas configurações que podem ser feitas com recurso a dois ficheiros: o docker-compose.yml e o .env. O primeiro tem as configurações necessárias para a

construção dos *containers* e que por sua vez utiliza variáveis de ambiente presentes no segundo ficheiro. Apenas foi necessário alterar o ficheiro das variáveis de ambiente já que o docker-compose.yml já estava adequado às necessidades do projeto.

Neste momento, é possível encontrar a interface web no URL indicado no ficheiro de configuração anteriormente referido. Na fig. 4.30 é possível visualizar a página inicial da interface sem esta estar personalizada.

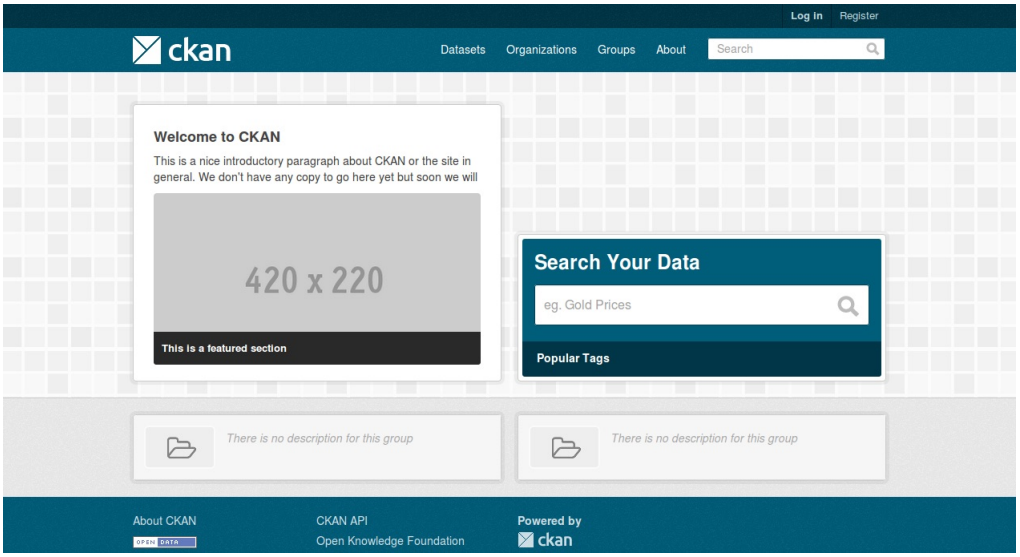

Figura 4.30: Interface por defeito

Através da interface de gestão a que o administrador tem acesso procedeu-se à alteração de várias configurações: o título, o logo, o esquema de cores da interface, e o modo como as várias secções estão organizadas. Após as mudanças referidas anteriormente é possível agora ver o esquema de cores concordante com as restantes interfaces relacionadas com o projeto (fig. 4.31) e uma secção onde é apresentado o projeto.

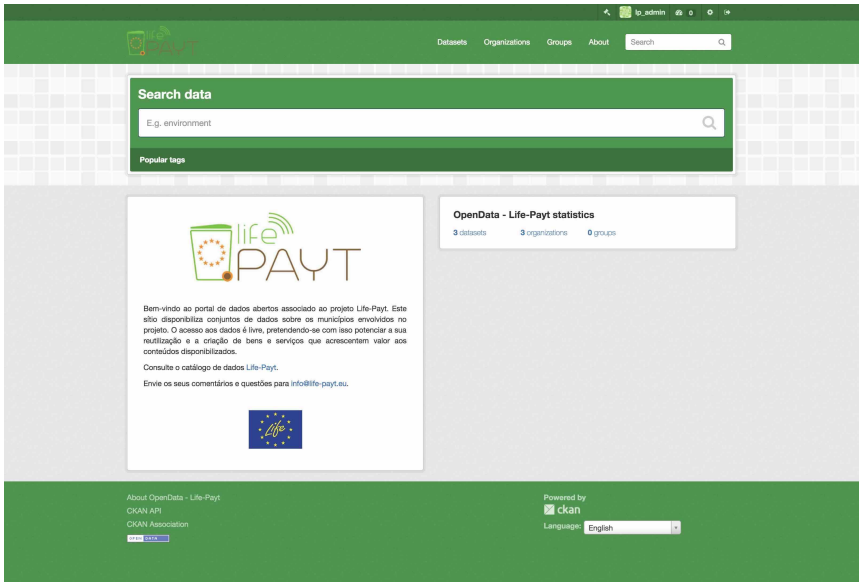

Figura 4.31: Interface personalizada para o projeto

Agora que a instância está adaptada ao contexto do projeto faltam então os dados para serem lá apresentados. No CKAN cada conjunto de dados é propriedade de uma organização e apenas utilizadores pertencentes a essa mesma organização podem publicá-los. Uma instância pode ter várias organizações. Por exemplo, no caso do CKAN ser usado como um portal de dados por um governo, as organizações podem ser departamentos diferentes, onde cada um dos quais publica dados relacionados com as suas atividades. No caso do projeto LIFE-PAYT cada organização está associada com um município envolvido. Para tal, foram criadas 5 organizações, uma para cada município. Após a sua criação foram registados 5 utilizadores responsáveis pela gestão de cada uma das organizações.

Como já foi referido anteriormente, o CKAN disponibiliza uma poderosa API que expõe todas funcionalidades (as que estão disponíveis na interface web e mais) para que código externo consiga interagir com a plataforma. É através desta API e com recurso aos utilizadores criados para cada município que o serviço responsável por exportar os dados vai interagir com a plataforma e enviar para lá a informação. Algumas das funções da API necessitam de autorização. Quando uma função que requer autorização é chamada, tem que ser fornecida uma chave de autenticação junto com o pedido, chave essa que pode ser encontrada na página de perfil na interface web.

Depois da instância do portal estar preparada e as contas de utilizador criadas resta então fazer com que o componente responsável pelos dados exporte os mesmos para o CKAN. O diagrama presente na fig. 4.32 representa o fluxo de dados no sistema no processo de exportação.

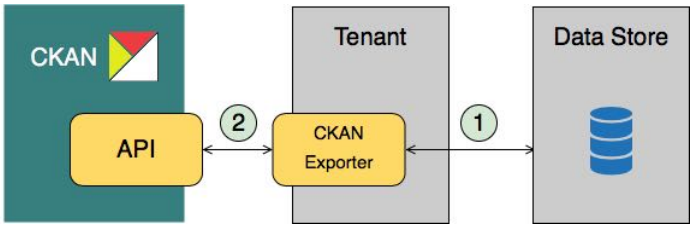

Figura 4.32: Processo de exportação

Antes de descrever as etapas do processo de exportação é importante perceber o modo como este componente foi construído. Este componente é um módulo escrito em Python que funciona como uma extensão ao componente principal que é responsável por todas as interações com a base de dados. Na inicialização deste módulo são definidas algumas propriedades internas (bloco de código 3 presente no anexo A) necessárias para a comunicação com o portal CKAN. Algumas dessas propriedades, por forma a tornar este módulo genérico para qualquer município, vão ser atribuídas com valores provenientes de variáveis de ambiente definidas aquando da criação do *container* Docker. Entre elas destacam-se o URL onde está disponível o portal, e a chave de autorização. Para além da API, os criadores do CKAN fornecem um módulo Python para aceder a ela.

Depois do módulo estar corretamente configurado é então possível exportar os conjuntos de dados pretendidos para o portal. Este é um processo automático e composto por 3 etapas que se repetem por cada conjunto de dados que se pretenda exportar: obtenção da informação vinda da base de dados, criação dos ficheiros e envio desses mesmos ficheiros para o portal.

Obtenção da informação

Durante o tempo de execução deste componente, este tem ao seu dispor o módulo responsável pela comunicação com a base de dados podendo chamar qualquer função lá definida. Através de funções já lá existentes ou criando outras para o efeito é extraída da base de dados a informação pretendida. Antes de passar para a próxima etapa é retirada qualquer informação que permita identificar alguém.

Criação dos ficheiros

Após a informação ser obtida é necessário organizá-la em ficheiros para serem enviados para o portal. De acordo com a secção 2.5.3 os dados devem ser disponibilizados nos formatos mais comuns. Tendo isto em conta todos os conjuntos de dados serão encapsulados em 3 formatos: XLS, CSV e JSON. Para a criação do ficheiro XLS foi usado o módulo Xlsxwriter. Foi adotada esta solução porque é um módulo com uma

excelente documentação, suporta muitas das funcionalidades do Excel e tem baixos consumos de memória associados. Depois de construído, o ficheiro é guardado em disco de forma temporária até ser exportado.

Após o ficheiro estar disponível no formato XLS é feita a conversão para os formatos CSV e JSON tirando partido da biblioteca de análise de dados Pandas. Graças a esta biblioteca é possível carregar o ficheiro anteriormente criado e transformá-lo em ficheiros com diferentes formatos como pode ser visto no bloco de código 2.

```
data_xls = pd.read_excel(fpath_xls)
data_xls.to_csv(fpath_csv, encoding='utf-8', index=False)
data_xls.to_json(path_or_buf=fpath_json, orient='table')
```

Bloco de código 2: Conversão de XLSX para outros formatos

Envio dos ficheiros para o portal

O último passo no processo de exportação passa por enviar os ficheiros já criados para o CKAN. Os ficheiros criados têm de estar associados a um conjunto de dados. Isto significa que, caso o conjunto de dados ainda não exista na organização, este tem de ser criado antes do envio dos ficheiros.

Caso o conjunto de dados já exista, os ficheiros apenas têm de substituir os já existentes ou, caso ainda não exista nenhum, podem simplesmente ser adicionados. Tanto a função de criar um novo conjunto de dados como a de adição/substituição de ficheiros são funções que requerem autorização, daí este componente necessitar de ser configurado com uma chave de autorização. Após os ficheiros serem adicionados/substituídos no portal, estes são removidos do disco e passam a só existir na instância do CKAN.

capítulo 5

Resultados

O objetivo deste capítulo é avaliar a implementação da solução proposta no capítulo 4 através de vários testes de desempenho. Em primeiro lugar irão ser apresentados os testes efetuados para obter os tempos de carregamento da página, com uma breve descrição dos mesmos seguida dos resultados obtidos. Posteriormente serão apresentados os resultados dos testes de carga efetuados na plataforma.

5.1 testes de desempenho

Nesta secção são apresentados os testes de desempenho efetuados à aplicação web, começando pela especificação do ambiente dos testes, nomeadamente as condições da ligação e os recursos do dispositivo.

O tempo de carregamento é um dos fatores que mais contribui para o utilizador deixar de visitar uma página. Segundo [44], existem 3 limites no que toca aos tempos de resposta de uma página web:

- **0.1 segundos -** ações que demorem até 100ms parecem instantâneas para o utilizador.
- **1 segundo -** o utilizador sente o atraso nas operações que demorem 1 segundo, no entanto continuam-se a sentir no controlo.
- **10 segundos -** num intervalo de 1s-10s sentem definitivamente o atraso, mas mantêm a atenção. Mais que 10s o utilizador começa a pensar noutras coisas e acaba por deixar de visitar a página.

Tendo isto em conta espera-se que o tempo inicial de carregamento da aplicação não se aproxime dos 10s, mantendo-se o mais baixo possível. Quanto às ações dentro da aplicação como por exemplo mudar de separador, espera-se que estas não excedam o 1s.

Foram realizados testes para avaliar o desempenho da aplicação web desenvolvida. Foram consideradas duas métricas relativas ao tempo de carregamento da página: o momento em que o browser emite o evento onLoad e o momento em que após este ter sido emitido não ocorra qualquer atividade de rede durante 2s. Para além das métricas anteriormente referidas foram também medidos: o tempo de obtenção de todos os recursos necessários ao carregamento da página bem como o tempo de obtenção dos dados do lado do servidor por websockets.

No que toca aos testes para obtenção das métricas referentes aos tempos de carregamento foram considerados dois cenários: a primeira visita e a visita repetida. A primeira visita significa que tanto os cookies como a cache foram limpos e representa a experiência vivida pelo utilizador na primeira vez que visitar a aplicação. Na visita repetida os cookies e a cache permanecem intactos e representa o que o utilizador irá ver se visitar a aplicação algum tempo após visitá-la pela primeira vez. Estes testes foram efetuados com recurso a ferramentas de teste e automação de web browsers. Para tal, foram escolhidos o Selenium [1] e o Mocha [2].

Para a medir os tempos de obtenção de recursos e dos dados recorreu-se a um *script* em NodeJS. Este simulava uma aproximação ao fluxo normal de um *browser*, começando por reproduzir os pedidos para cada recurso. Após obter todos os recursos é aberta uma conexão por *websocket* usada para comunicar com o servidor para obtenção dos dados necessários à apresentação da página visitada. No fim são apresentados os dois tempos parciais (obtenção de recursos e obtenção de dados) e o tempo total. A este tempo total acresce ainda o tempo gasto pelo browser para renderização do HTML correspondente.

Os testes foram realizados utilizando uma máquina com as seguintes especificações: processador com 4 cores a uma frequência de 2.9GHz e memória RAM de 8GB. Todos os testes foram realizados para três velocidades de conexão diferentes de forma a tentar cobrir a maioria das experiências vividas pelo utilizador real:

- **3G -** $\approx$ 1.6mbps / 400ms RTT
- **DSL -** $\approx$ 5mbps / 28ms RTT
- **Fibra ótica -** $\approx$ 20mbps / 6ms RTT

[1] https://www.seleniumhq.org
[2] https://mochajs.org

Todos os pedidos ao servidor onde está alojada a aplicação passam pelo serviço Cloudflare [3] (fig. 5.1). De seguida serão apresentados os resultados referentes aos testes efetuados sendo que os valores apresentados foram calculados após 100 repetições de cada teste.

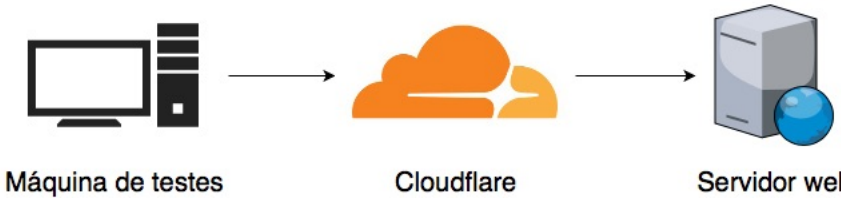

Figura 5.1: Fluxo de dados referente aos pedidos ao servidor que aloja a aplicação web

5.1.1 TEMPO DE CARREGAMENTO

Os resultados obtidos para dispositivos com baixa velocidade de conexão (3G) à internet podem ser observados na tabela 5.1.

	Carregamento (s)	**Total(s)**
Primeira visita	9,86 ± 0,49	14,612 ± 0,5
Visita repetida	2,65 ± 0,09	4,03 ± 0,11

Tabela 5.1: Tempos de carregamento com ligação 3G

Para dispositivos com conexão à internet semelhantes a DSL, os resultados podem ser observados na tabela 5.2.

	Carregamento (s)	**Total(s)**
Primeira visita	3,44 ± 0,19	4,97 ± 0,37
Visita repetida	1,35 ± 0,05	1,89 ± 0,31

Tabela 5.2: Tempos de carregamento com ligação DSL

Por fim, para dispositivos com elevada velocidade de conexão (fibra ótica) à internet os resultados podem ser observados na tabela 5.3.

	Carregamento (s)	**Total(s)**
Primeira visita	2,11 ± 0,3	3,00 ± 0,45
Visita repetida	1,21 ± 0,05	1,72 ± 0,48

Tabela 5.3: Tempos de carregamento com ligação fibra ótica

[3] https://www.cloudflare.com

Análise de resultados

Ao analisar os resultados obtidos pode constatar-se que o tempo médio de carregamento apenas depende das condições da ligação caso os mecanismos de cache não estejam ativos ou se for a primeira vez que a página é carregada, pois significa que os recursos têm de ser todos descarregados para o dispositivo. Caso os mecanismos de cache estejam ativos, os tempos médios não sofrem grandes alterações com o condicionamento da ligação pois os recursos já se encontram em memória e, assim, o tempo de carregamento apenas depende da capacidade de processamento do dispositivo.

Comparando os valores obtidos com os considerados aceitáveis segundo [44], verifica-se que quando a ligação está condicionada e a cache desativada ou vazia, os tempos médios de carregamento ultrapassam o limite máximo aumentando a probabilidade do utilizador perder a atenção. Caso a cache esteja ativa e a ligação não esteja condicionada, os tempos são bastante bons.

5.1.2 TEMPOS DE OBTENÇÃO DE RECURSOS E DADOS

Para além da avaliação dos tempos de carregamento da página para diferentes condições de rede e recursos, foram também realizados testes para avaliar o tempo de obtenção dos recursos e dados necessários. Os resultados podem ser observados na tabela 5.4.

Ligação	**Obtenção de recursos (s)**	**Obtenção de dados (s)**	**Total(s)**
3G	12,60 ± 0,78	3,71 ± 0,02	16,31 ± 0,77
DSL	2,89 ± 0,24	0,73 ± 0,04	3,62 ± 0,25
Fibra Ótica	1,66 ± 0,13	0,73 ± 0,03	2,39 ± 0,13

Tabela 5.4: Tempos de obtenção de recursos/dados para diferentes condições

Análise de resultados

A análise dos resultados obtidos permite concluir que a grande fatia do tempo despendido no carregamento da página se deve à obtenção dos recursos necessários. Este já era um resultado esperado, já que os dados vindos do lado do servidor são apenas em formato de texto enquanto que os recursos incluem imagens e várias bibliotecas JS necessárias ao correto carregamento da página.

Com a variação da velocidade de ligação verifica-se que para baixas velocidades (3G) o tempo de obtenção de recursos é bastante alto afetando negativamente o carregamento

total da página. Para velocidades médias e altas (DSL e fibra ótica) esse tempo é substancialmente mais baixo estando dentro dos valores aceitáveis referidos em [44].

5.2 TESTES DE CARGA

De modo a validar a solução proposta e mostrar que a plataforma desenvolvida é capaz de suportar as necessidades de um município foram efetuados testes de carga. Para a realização destes testes foi desenvolvida uma ferramenta que simula acessos à interface destinada aos cidadãos. Cada acesso implica a obtenção de informação que está no lado do servidor. De maneira a sobrecarregar o servidor e perceber qual o seu comportamento em casos de elevado número de acessos num curto espaço de tempo, foi elaborado o teste a seguir descrito. Procedeu-se à criação de um município de teste, chamado Caoli, com dez mil utilizadores registados, cada um com cinco anos de faturas/consumos associados a si. Foram recolhidas informações acerca do servidor, nomeadamente a carga média do sistema (no último minuto) e o uso de memória, durante um período de duas horas. Neste intervalo de tempo, a cada 5 minutos eram lançados 100 acessos à plataforma, representando um cenário de baixa carga. Para além deste cenário foram simulados outros dois, mais intensivos: um cenário de carga média em que foram lançados 1000 acessos (espaçados no tempo durante um período de 10 minutos) e um de carga elevada em que foram lançados 20000 acessos (espaçados no tempo durante um período de cerca de 30 minutos). Na figura 5.2 pode-se observar a carga média do sistema (4 CPUs), que representa o número de processos que estão a ser executados ou à espera de o ser, ao longo do período monitorizado.

No gráfico da carga média torna-se fácil de identificar os momentos em que os dois cenários mais intensivos estavam a decorrer. No momento em que a carga imposta pelos testes é média, identificado no gráfico pela letra A, verifica-se que os valores da carga do sistema sobem, mas nunca chegam a atingir valores muito elevados, sendo 1 o seu valor máximo. Este valor encontra-se abaixo do limite máximo (cerca de 70% do número de CPUs) indicado para testes deste tipo. O momento identificado no gráfico pela letra B representa o período em que o sistema sofreu um maio número de pedidos. Durante este período a carga média atingiu valores mais elevados, chegando mesmo a passar o valor 4 (limite máximo, em relação ao número de CPUs, para testes deste tipo).

Quanto à memória RAM utilizada pelo sistema, foi monitorizada a sua variação ao longo do tempo em que decorreram os testes. Em períodos de baixa carga o sistema utiliza cerca de 359MB, repartidos pelos vários módulos. Foi possível verificar que este valor, mesmo aumentando substancialmente a carga imposta, se manteve relativamente

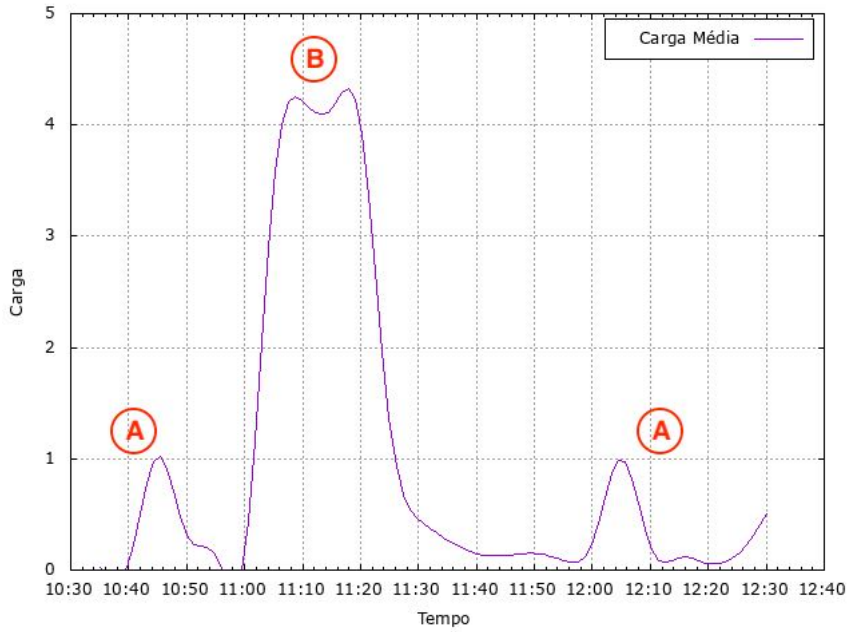

Figura 5.2: Carga média do sistema no período monitorizado

constante.

Para além desta monitorização, foi recolhida informação acerca dos tempos de resposta obtidos em cada um dos cenários. Estes tempos apenas correspondem ao tempo que a plataforma demora a processar o pedido e devolver a informação pedida, estando excluído o tempo de carregamento de recursos por parte do browser para efeitos visuais. Os resultados das medições do tempo de resposta podem ser observados na tabela 5.5.

	Carga baixa	**Carga média**	**Carga elevada**
Desvio padrão (s)	0,78	0,97	2,19
Tempo médio (s)	1,42	4,98	4,82
Tempo mínimo (s)	0,24	0,27	0,42
Tempo máximo (s)	2,43	6,35	10,64
Erros	0	0	13

Tabela 5.5: Tempos de resposta para diferentes cargas

Quanto ao tempo médio verifica-se que tanto o cenário de carga média como o de carga elevada apresentam valores semelhantes. No entanto, o tempo máximo observado num cenário de carga elevada é bastante mais alto que no cenário de carga média. Para além disso, pode-se observar que no período em que o sistema esteve sujeito a uma maior carga ocorreram alguns erros que impediram que esses pedidos pudessem ter uma resposta.

capítulo 6

Conclusões e Trabalho Futuro

Antes deste trabalho, o modelo de cobrança de tarifas referentes à produção de resíduos implementado em Portugal era um modelo que assentava em estimativas. A entidade responsável por cobrar a tarifa estimava, através do consumo de água, a quantidade de resíduos que cada cidadão ou comércio produzia e cobrava uma tarifa associada a essa estimativa. Com o trabalho realizado nesta dissertação, é agora possível a essa mesma entidade saber o real valor, ou pelo menos um valor muito mais próximo do real, da quantidade de resíduos produzidos, e assim, cobrar um valor mais justo. Com isto, os cidadãos que neste momento estão a pagar montantes acima do que deveriam, ficam mais satisfeitos pois deixam de se sentir injustiçados.

Nesta dissertação foi criada uma interface para os diversos utilizadores, que permite a consulta das quantidades de resíduos que estão a produzir, bem como o valor das tarifas que estão a ser cobradas. Para além destes valores a interface fornece uma grande variedade de indicadores interessantes que podem ser consultados pelos utilizadores.

Com o trabalho realizado nesta dissertação é agora possível a outras pessoas utilizarem os dados que são exportados para o catálogo de dados abertos para trabalhos de investigação ou para qualquer outro tipo de finalidades.

Depois da avaliação do desempenho que foi efetuada no capítulo 5, pode-se concluir que mesmo sob uma carga elevada durante um grande período de tempo, o sistema é capaz de responder à grande maioria dos pedidos em menos de 5 segundos. Este fato vem comprovar que a solução proposta nesta dissertação e a sua implementação são capazes de satisfazer as necessidades de um município.

Como trabalho futuro, a implementação da interface deveria passar na totalidade a seguir a arquitetura Flux que foi descrita no capítulo 4, de modo a que a interface possua uma arquitetura uniforme. Para além disso, com uma arquitetura Flux iria haver uma melhor separação entre a lógica, que iria ficar toda nas *Stores*, e a visualização, que ficaria nos componentes. Outra melhoria seria tirar mais partido da cache dos browsers, armazenando nela os valores obtidos pelas chamadas à API para que nas próximas vezes que fosse preciso, caso o valor fosse o mesmo, o valor já se encontraria em cache, tornando a interface mais rápida.

Como trabalho futuro seria interessante adicionar novas funcionalidades na interface do administrador como por exemplo a configuração e definição de novos alertas, bem como a possibilidade de adicionar novos municípios ao sistema com recurso a um formulário que serviria para definir as configurações necessárias para o lançamento do *container* Docker associado.

Referências

[1] D. Hoornweg, P. Bhada-Tata e C. Kennedy, «Environment: Waste production must peak this century», *Nature News*, 2013.

[2] World Economic Forum. (2017). Which countries produce the most wasteper person?, [Online] Disponível: https://www.weforum.org/agenda/2015/08/which-countries-produce-the-most-waste/ (acedido em 28/08/2017).

[3] (2012). What a Waste: A Global Review of Solid Waste Management, [Online] Disponível: https://siteresources.worldbank.org/INTURBANDEVELOPMENT/Resources/336387-1334852610766/What_a_Waste2012_Final.pdf.

[4] Adtell Integration. (2017). Smart Cities Infrastructure, [Online] Disponível: http://adtellintegration.com/smart-cities-infrastructure/ (acedido em 23/05/2017).

[5] E. de Portugal (EDP). (2017). InovGrid, [Online] Disponível: https://www.edpdistribuicao.pt/pt/rede/InovGrid/Pages/InovGrid.aspx.

[6] (2014). Santander: the smartest smart city, [Online] Disponível: http://www.governing.com/topics/urban/gov-santander-spain-smart-city.html.

[7] (2015). SmartBin Announce Deal with Portuguese Municipality, [Online] Disponível: https://www.smartbin.com/smartbin-announce-deal-with-portuguese-municipality/.

[8] (2017). Cascais – SmartBin Solution, [Online] Disponível: https://www.smartbin.com/clients/cascais-municipality/.

[9] M. Batllevell e K. Hanf, «The fairness of PAYT systems: Some guidelines for decision-makers», *Waste Management*, vol. 28, n.º 12, pp. 2793–2800, 2008, Pay as you throw: a tool fo urban waste management, ISSN: 0956-053X. DOI: https://doi.org/10.1016/j.wasman.2008.02.031.

[10] J. Reichenbach, «Status and prospects of pay-as-you-throw in Europe – A review of pilot research and implementation studies», *Waste Management*, vol. 28, n.º 12, pp. 2809–2814, 2008, Pay as you throw: a tool fo urban waste management, ISSN: 0956-053X. DOI: https://doi.org/10.1016/j.wasman.2008.07.008.

[11] (2011). Pay-As-You-Throw in Spain – Waste360, [Online] Disponível: http://www.waste360.com/pay-you-throw-payt/pay-you-throw-spain.

[12] (2015). Projeto inovador em Portugal muda recolha de lixo no Centro Histórico de Guimarães, [Online] Disponível: http://www.cm-guimaraes.pt/frontoffice/pages/991?news_id=2363.

[13] D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris e D. Orchard. (2004). Web Services Architecture. W3C, [Online] Disponível: http://www.w3.org/TR/ws-arch/.

[14] P. A. Castillo, J. L. Bernier, M. G. Arenas, J. Merelo e P. Garcia-Sanchez, «SOAP vs REST: Comparing a master-slave GA implementation», *arXiv preprint arXiv:1105.4978*, 2011.

[15] D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mandelsohn, H. F. Nielsen, S. Thatte e D. Winer, «Simple Object Access Protocol (SOAP) 1.1», W3C, W3C Recommendation, 2000, https://www.w3.org/TR/2000/NOTE-SOAP-20000508/.

[16] S. Mumbaikar, P. Padiya et al., «Web services based on soap and rest principles», *International Journal of Scientific and Research Publications*, vol. 3, n.º 5, pp. 1–4, 2013.

[17] IBM. (2017). The structure of a SOAP message, [Online] Disponível: `https://www.ibm.com/support/knowledgecenter/en/SSMKHH_10.0.0/com.ibm.etools.mft.doc/ac55780_.htm` (acedido em 03/10/2017).

[18] Realtime API Hub. (2017). Request-Response APIs, [Online] Disponível: `https://realtimeapi.io/hub/requestresponse-apis/` (acedido em 03/10/2017).

[19] S. Graham, *Building Web Services with Java: Making Sense of XML, SOAP, WSDL, and UDDI*, sér. Java (Sams). Sams, 2002, ISBN: 9780672321818. [Online] Disponível: `https://books.google.pt/books?id=5TPEHhuIum8C`.

[20] I. Fette e A. Melnikov, «The WebSocket Protocol», RFC 6455, 2011.

[21] V. Karagiannis, P. Chatzimisios, F. Vazquez-Gallego e J. Alonso-Zarate, «A survey on application layer protocols for the internet of things», *Transaction on IoT and Cloud Computing*, vol. 3, n.º 1, pp. 11–17, 2015.

[22] D. Jaramillo, D. V. Nguyen e R. Smart, «Leveraging microservices architecture by using Docker technology», em *SoutheastCon, 2016*, IEEE, 2016, pp. 1–5.

[23] S. Daya, N. Van Duy, K. Eati, C. Ferreira, D. Glozic, V. Gucer, M. Gupta, S. Joshi, V. Lampkin, M. Martins et al., *Microservices from Theory to Practice: Creating Applications in IBM Bluemix Using the Microservices Approach*. IBM Redbooks, 2016, ISBN: 9780738440811. [Online] Disponível: `https://books.google.pt/books?id=eOZyCgAAQBAJ`.

[24] (2017). Benefits & Examples of Microservices Architecture Implementation, [Online] Disponível: `https://apiumhub.com/tech-blog-barcelona/microservices-architecture-implementation/`.

[25] Compta Emerging Business. (2017). Which countries produce the most wasteper person?, [Online] Disponível: `https://www.weforum.org/agenda/2015/08/which-countries-produce-the-most-waste/` (acedido em 23/05/2017).

[26] EDP. (2018). EDP RE:DY, [Online] Disponível: `https://www.edp.pt/particulares/servicos/redy/5` (acedido em 02/06/2018).

[27] Governo de Portugal. (2017). Portal da Transparência Municipal, [Online] Disponível: `https://www.portalmunicipal.gov.pt/municipio?locale=pt&param_municipio=105` (acedido em 19/10/2017).

[28] R. Thierauf, *Knowledge Management Systems for Business*. Quorum Book, 1999, ISBN: 9781567202182. [Online] Disponível: `https://books.google.pt/books?id=5Dv82411J-AC`.

[29] T. Davenport e L. Prusak, *Working Knowledge: How Organizations Manage What They Know*. Harvard Business Review Press, 2000, ISBN: 9781422160688. [Online] Disponível: `https://books.google.pt/books?id=RkQ3zMKQvOYC`.

[30] (2016). Guia dos Dados Abertos, [Online] Disponível: `https://www.ama.gov.pt/documents/24077/24804/guia_dados_abertos_ama.pdf`.

[31] (2007). Lei de Acesso aos Documentos Administrativos, [Online] Disponível: `http://www.cada.pt/modules/smartsection/item.php?itemid=41`.

[32] J. Hoxha e A. Brahaj, «Open Government Data on the Web: A Semantic Approach», em *2011 International Conference on Emerging Intelligent Data and Web Technologies*, 2011, pp. 107–113. DOI: 10.1109/EIDWT.2011.24.

[33] Mark Webbink. (2017). Free Software License, [Online] Disponível: https://en.wikipedia.org/wiki/Free_software_license (acedido em 19/10/2017).

[34] B. S. Kristoffersen, «An open-source approach to Integrated Operations», tese de mestrado, NTNU, 2017.

[35] A. Ziadeh. (2015). Picking the right portal to make open data accessible, [Online] Disponível: https://gcn.com/articles/2015/07/10/open-data-portal.aspx.

[36] (2017). How to Open Data, [Online] Disponível: http://opendatahandbook.org/guide/en/how-to-open-up-data/.

[37] (2018). Docker – Porque é esta uma tecnologia tão popular?, [Online] Disponível: https://pplware.sapo.pt/linux/docker-tecnologia-tao-popular/.

[38] Stack Overflow. (2017). Most Loved, Dreaded, and Wanted Frameworks, Libraries and Other Technologies, [Online] Disponível: https://insights.stackoverflow.com/survey/2017#most-loved-dreaded-and-wanted (acedido em 28/10/2017).

[39] E. team. (2017). Angular vs. react: Choosing the right framework, [Online] Disponível: https://effectiveinc.com/thought-leadership/blogs/angular-vs-react-choosing-right-framework/.

[40] F. Inc. (2017). ReactJS: Documentação Oficial, [Online] Disponível: https://reactjs.org/docs/getting-started.html.

[41] (2016). Sizes of JS frameworks, just minified + minified and gzipped, (React, Angular 2, Vue, Ember), [Online] Disponível: https://gist.github.com/Restuta/cda69e50a853aa64912d.

[42] T. Republic. (2017). Bootstrap Grid System, [Online] Disponível: https://www.tutorialrepublic.com/twitter-bootstrap-tutorial/bootstrap-grid-system.php.

[43] F. Inc. (2017). Flux - In Depth Overview, [Online] Disponível: https://facebook.github.io/flux/docs/in-depth-overview.html (acedido em 28/10/2017).

[44] J. Nielsen. (2010). Website Response Times, [Online] Disponível: https://www.nngroup.com/articles/website-response-times/.

Anexo A: Exemplo de configuração do módulo de exportação

```
class e_ckan(object):
        def __init__(self, *args, **kwargs):
        self._name = kwargs.get('e_name', None)
        self._county = kwargs.get('e_county', None)
        self._refresh_every= 300

        # CKAN API
        self._url_ckan = kwargs.get('url_ckan', None)
        self._api_key = kwargs.get('api_key', None)
        self._mysite = RemoteCKAN(self._url_ckan,
        apikey=self._api_key,
        user_agent='CKAN Exporter')
        self._org = 'cm-' + self._county
        self._license = 'cc-zero'
        self._username = 'gis' + self._county
        self._user = self._mysite.action.user_show(id=self._username)
        self._author = self._user['display_name']
        self._author_email = self._user['email']
```

Bloco de código 3: Configuração do módulo