Universidade do Minho
Escola de Engenharia

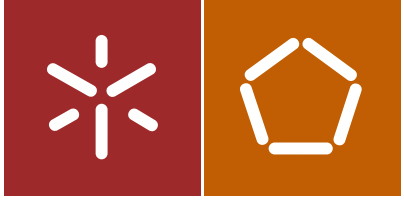
Kevin Ribeiro da Costa

EcoWise – Sistema de Monotorização de Ecopontos baseado em Cloud

outubro de 2016

Kevin Ribeiro da Costa EcoWise – Sistema de Monotorização de Ecopontos baseado em Cloud

UMinho | 2016



Universidade do Minho
Escola de Engenharia

Kevin Ribeiro da Costa

EcoWise – Sistema de Monotorização de Ecopontos baseado em Cloud

Dissertação de Mestrado
Ciclo de Estudos Integrados Conducentes ao Grau de
Mestre em Engenharia Eletrónica Industrial e Computadores

Trabalho efetuado sob a orientação do
Professor Doutor Nuno Filipe Gomes Cardoso

outubro de 2016

AGRADECIMENTOS

Em primeiro lugar, o maior agradecimento é, sem dúvida, destinado à minha mãe Maria Jacinta, ao meu pai Agostinho Costa, ao meu irmão Dino Mickael Costa por todo o apoio e suporte durante a minha vida pessoal e académica.

Ao meu orientador, Professor Doutor Nuno Cardoso pelo apoio e confiança depositada em mim para a realização deste projeto e pelas oportunidades proporcionadas. Aos meus colegas de curso e amigos, principalmente os que partilhei o laboratório Pedro Leite, Bruno Mendes, Artur Faria, António Ribeiro, Diogo Cruz, Rui Paixão e Áurea Salgado pela amizade, confiança e acompanhamento do meu percurso académico e por terem proporcionado todas as condições para a realização desta dissertação e acima de tudo pelos momentos bem passados.

Por fim, um grande agradecimento ao Alexis Oliveira pela orientação prestada durante o desenvolvimento desta dissertação e por todos os conhecimentos disponibilizados.

A todos, muito obrigado!

RESUMO

Um dos maiores problemas da nossa sociedade é a quantidade de recursos existentes no nosso planeta, mais concretamente a sua escassez que pode atingir níveis críticos no futuro. A reciclagem tem como objetivo utilizar os detritos produzidos pelas pessoas em novos produtos e consequentemente reduzir a quantidade de recursos utilizados.

No entanto a recolha de detritos filtrados necessita de mais recursos(veículos e mão-de-obra) comparando com a recolha de detritos não-filtrados tornando o processo consideravelmente mais caro ,o processo de recolha não é o mais eficiente devido a falta de informação relativa ao estado de ocupação dos ecopontos.

O sistema Ecowise procura resolver este problema recorrendo à intervenção da população. Através de uma aplicação para dispositivos móveis que permitirá a qualquer cidadão comum relatar o estado dos ecopontos na sua localização, de uma forma rápida e acessível.

Esta informação é armazenada numa base de dados que contém informação dos ecopontos que estão a ser monitorizados e todas as atualizações sobre o seu estado.

Para permitir ás empresas que tratam da manutenção dos ecopontos o acesso a esta informação, será desenvolvido um *website*. Através dele é possível determinar quais os ecopontos que necessitam de intervenção e assim eliminar gastos desnecessários em combustível e pessoal . Para além disso, os dados recolhidos permitem tomar decisões que tornam o processo de recolha mais eficiente, por exemplo, na modificação/alteração das rotas de recolha existente e averiguar a necessidade de mais ecopontos em certos locais.

ABSTRACT

One of the biggest problems in our society is the quantity of resources still left in our planet, more specifically their fast shortage that may reach critical levels in the future. The purpose of recycling is to take advantage of debris generated by people into new products and at the same time reduce the quantity of resources spent. However, the colleting of filtered debris is considerably more expensive and the collection process is not the most efficient because of the lack of information on the occupation state of the Eco points.

The Eco wise system seeks to solve this problem with the involvement of the population. An application for mobile device will allow for any ordinary citizen to report the state of the Eco points in a quick and accessible way.

This information is then stored in a database which will have information about the Eco points that are being monitored and all of the updates about their state from the app users.

To allow the companies responsible for the maintenance of Eco points to access this information, a website will be developed. With it, they will be able to determine which Eco points are in need of attention and thus eliminate unnecessary expenses in fuel and staff caused by unnecessary travels. Furthermore the collected data will allow to make decision, that make the collection process more efficient, for example, it can be used to modify existing collection routes and assess the need for more ecopoints in certain locations.

INDICE

INDICE DE FIGURAS

LISTA DE ALGORITMOS

Acrónimos e abreviaturas

UE União Europeia

RFID *RadioFrequency Identification*

RAM *Random Access Memory*

ROM *Read Only Memory*

HTML *HyperText Markup Language*

DOM *Document Object Model*

P2P *Peer-to-Peer*

USB *Universal Serial Bus*

SDK *Software Development Kit*

IDE *Integrated Development Environment*

ADT *Android Developer Tools*

GPS *Global Position System*

NFC *Near Field Communication*

HTTP *Hypertext Transfer Protocol*

XML *EXtensible Markup Language*

JSON *JavaScript Object Notation*

WSDL *Web Services Description Language*

UDDI *Universal Description, Discovery and Integration*

SOAP *Simple Object Access Protocol*

REST *Representational state transfer*

URL *Uniform Resource Locator*

SDK *Software Development Kit*

ODM *Object Modeling Tool*

1

INTRODUÇÃO

Nos dias de hoje, existe uma crescente preocupação com o planeta e a contínua escassez de recursos. O constante crescimento da população mundial significa um correspondente aumento dos detritos produzido, principalmente por parte dos países mais desenvolvidos. Uma das soluções para este problema é a reciclagem, que permite o aproveitamento de detritos em novos produtos, reduzindo significativamente a utilização de novos recursos. No entanto, apesar de esta realidade ser conhecida há bastante tempo, não é uma prática comum em muitos países, existindo um elevado número de pessoas que não recicla o seu lixo e coloca todos os detritos no lixo comum[1].

No caso de Portugal, a situação não é muito animadora. Apesar de se verificar um aumento da reciclagem nos últimos anos, a taxa apresenta baixos valores (cerca de 22%), aquém do que foi estabelecido como meta pela EU para 2020 (50%). Por isso, é necessário um esforço bastante relevante para se atingir essa meta[2].

É necessário ter em conta que a colheita seletiva é mais dispendiosa, desde a recolha até á triagem [3]. Apesar de vivemos num país em que a poupança é uma prioridade, ainda existem muitos gastos desnecessários na recolha. Um exemplo disso é recolha dos contentores, sendo que as empresas de gestão dos ecopontos não têm maneira de detetar se existe lixo suficiente para ser recolhido e transportado para um centro de triagem. Caso não haja lixo suficiente, significa um desperdício de recursos que podia ser evitado se houvesse um conhecimento prévio da situação dos contentores. O inverso também pode ocorrer, a ultrapassagem da capacidade dos contentores provoca a deterioração da via pública e aversão à utilização dos ecopontos pelo utilizador[4].

O sistema EcoWise procura resolver este problema recorrendo à população para fornecer a informação sobre o estado de ocupação dos ecopontos que é necessária para às empresas de gestão de ecopontos. O sistema consiste na utilização de uma aplicação para *smartphone* que permite ao utilizador comum, no seu dia-a-

dia ,enviar informações relacionadas com os ecopontos da sua região de uma forma simples e eficaz. A informação é depois processada, estando disponível através de um *Website*, ás empresas de gestão de ecoponto.

Com a utilização do EcoWise é possível otimizar todo o processo de gestão dos ecopontos, evitando deslocações desnecessárias em situações em que a recolha não é necessária e mais, avisando que a recolha é necessária antes do período normalmente previsto para a recolha do lixo.

1.1 MOTIVAÇÃO

Nas sociedades modernas, o aumento do poder de compra levou a que os níveis de consumo atingissem valores históricos. Com a quantidade limitada de recursos existentes, a reciclagem começou a tornar-se uma prioridade. Como a recolha seletiva é bastante dispendiosa, na sociedade atual foram tomadas um conjunto de medidas, de forma a evitar o desperdício dos recursos existentes. Apesar do esforço, ainda não existe um sistema que seja capaz de monitorizar, de forma eficaz, o nível dos ecopontos e informar as empresas de gestão dos ecopontos, da necessidade do seu tratamento.

Com este sistema pretende-se: (i) minimizar os custos associados ao processo de gestão dos vários ecopontos, diminuíndo o risco de poluição associado ao facto de estes estarem cheios, ao longo de vários dias e (ii) aumentar a atenção da população relativamente à reciclagem. Como referido anteriormente, a percentagem da população que pratica a reciclagem é muito baixa, relativamente à pretendida, e sendo assim necessário que as pessoas tomem a consciência sobre a importância da reciclagem.

A possibilidade de trabalhar num projeto que promove a reciclagem e tornar as pessoas mais envolvidas no processo, foram os pontos mais importantes para a motivação da realização desta dissertação. Mais, este projeto permite desenvolver novas competências de aprendizagem que são abordadas durante o curso, nomeadamente as relacionadas com o desenvolvimento de aplicações para *smartphones* , e o desenvolvimento de aplicação *web* baseados em *Cloud*.

Palavra-Chave: Sistemas Embebidos, Programação Android, Programação Web, Desenvolvimento de Software.

1.2 OBJETIVOS

Com esta dissertação pretende-se desenvolver duas aplicações: (i) uma que permita ao utilizador comum registar eventos relacionados com os Ecopontos utilizando um smartphone e (ii) uma aplicação web com um conjunto de serviços alojados na *Cloud* que permitem às empresas de ecopontos fazer a gestão dos mesmos.

O primeiro objetivo é o desenvolvimento de uma aplicação para Android que possibilita a um utilizador comum registar e relatar informações relativas aos ecopontos. A recolha de informação é realizada através do envio de fotos e/ou informações relativos ao ecopontos e à sua condição. Depois, essa mesma informação é transmitida para *Cloud* de forma a ser processada.

Esta aplicação poderá ser utilizada pelas pessoas durante o seu dia-a-dia, por isso deve ser o mais acessível possível, de forma a que o utilizador não perca muito tempo a usá-la, e incentivar a sua utilização frequente.

Adicionalmente pretende-se a criação de uma aplicação *Web* que permita a identificação de casos que precisem de resolução, e informação estatística que pode ser utilizada para múltiplos fins, como por exemplo as rotas que devem ser realizadas ou as zonas em que se devem focar.

Finalmente, procura-se uma gestão de recursos mais eficiente e organizada, sensibilizando ao mesmo tempo a população com o tema da reciclagem.

Requisitos Funcionais:

- Desenvolver processos de recolha de informação dos ecopontos utilizando a aplicação móvel;
- Estabelecer interoperabilidade entre as aplicações cliente e a *Cloud*;
- Armazenar e processar a informação obtida pela aplicação;
- Identificar no mapa as localizações e os estados de ocupação dos ecopontos;

Requisitos Não-Funcionais:

- Ser robusto;
- Ser fiável;
- *User-friendly*;

- A aplicação deve ser desenvolvida para dispositivos móveis com o sistema operativo Android;
- Para a implementação *front-end* do *Website*, deve ser utilizada a *framework* Javascript AngularJS;
- O *Backend* deverá ser desenvolvido utilizando o NodeJs;

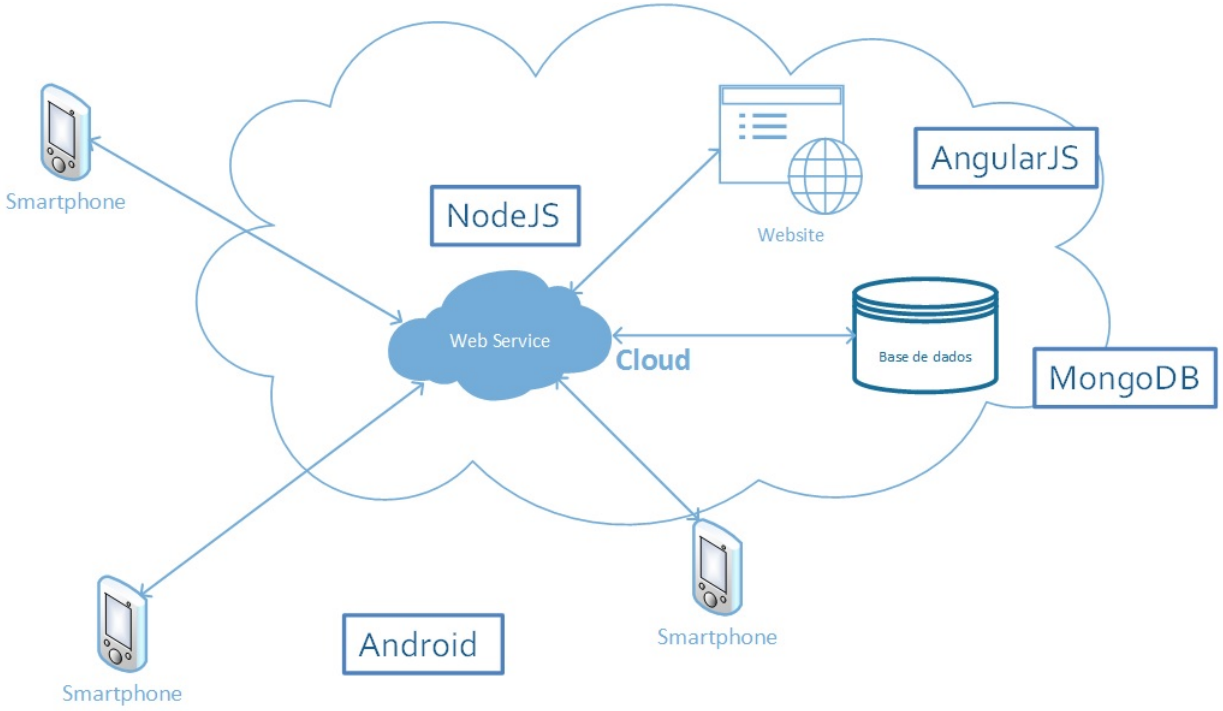


Figura 1: Visão geral do sistema.

1.3 METODOLOGIA DE PESQUISA

Pretende-se incentivar a população a prestar mais atenção ao tema da reciclagem e ao mesmo tempo interagir diretamente no seu processo. Assim, é necessário ter em conta a existência de três componentes diferentes: a aplicação móvel, a base de dados e *website*. Cada um deles devera ser capaz de realizar as suas atividades e de comunicar com as restantes de maneira eficiente.

Dessa forma na escolha da plataforma de desenvolvimento da aplicação devem ser considerados certos requisitos. Deve permitir a criação de uma aplicação acessível ao maior número de pessoas possíveis já que se pretende que seja utilizada regularmente sendo que a eficácia do sistema dependerá essencialmente da sua utilização pelo utilizador comum. Permita a criação de um *design* simples e

acessível que possibilite o registo de informações com o menor número de passos e informações necessárias de forma a aumentar a sua usabilidade. Possibilite a utilização da câmara já que esta é essencial tanto para a captura de imagens como leitura de códigos QR sendo que serão estas as principais fontes de obtenção de informação para o utilizador e também tenha a capacidade de utilizar *GPS* de forma a reconhecer corretamente a localização do utilizador aquando da captura de imagem sendo a precisão fundamental para que seja possível determinar qual ecoponto necessita de manutenção. Para a conexão é crítico que a aplicação seja capaz de se conectar á *internet* já que irá ser necessário conectar ao serviço Web de forma a enviar informações relativas á estes e também obter dados sobre a conta do utilizador.

Será necessário construir uma base de dados capaz de guardar a informação relativa ao ecopontos que estão a ser monitorizados e outras informações consideradas relevantes como por exemplo a informação relativa dos utilizadores e das situações que necessitam de atenção. Para isso a base de dados terá de ser capaz guardar a informação de uma maneira organizada de forma a ser obtida de uma forma rápida e eficiente. O serviço Web deve interpretar apropriadamente os pedidos provenientes das aplicações cliente e realizar as ações necessárias.

Por fim temos o *website* que deverá ser capaz de disponibilizar de uma forma compreensível e acessível a informação relativa aos ecopontos que estão a ser monitorizados. Deverá apresentar um mapa que permita ao utilizador determinar as situações em que é necessário intervir, a sua localização e o necessário para a sua resolução. Pretende-se que o *website* só seja acessível por pessoas que se encontrem envolvidas na gestão e monitorização de ecopontos, por isso terão de ser implementas medidas de seguranças que previnam o acesso não autorizado á essa informação. Permitir obter e enviar dados para a base de dados rapidamente de forma a manter a informação o mais atualizada e fiável possível.

1.4 ESTRUTURA DA DISSERTAÇÃO

Esta dissertação encontra-se dividida em cinco capítulos. No capítulo 1 é apresentado uma breve descrição do trabalho que ira ser desenvolvido como também os objetivos que se pretende cumprir.

No capítulo 2 é apresentado o estado da arte e o enquadramento teórico do tema da reciclagem relativos ao estado da reciclagem e da recolha de detritos em Portugal, das diferentes formas existentes de monitorização de ecopontos e tam-

bém um estudo das distintas tecnologias e abordagens que poderão ser utilizadas para o desenvolvimento do sistema.

No capítulo 3 é apresentada a arquitetura e a especificação do sistema do sistema EcoWise e o seu enquadramento no sistema de monitorização e recolha de ecopontos existente em Portugal. De seguida são descritos os três módulos que constituem o sistema e a sua importância para a viabilidade do sistema.

O capítulo 4 trata-se da implementação da arquitetura e módulos apresentados no capítulo 3, e também das modificações que tiveram de ser efetuadas para que o sistema funcione como planeado.

No capítulo 5 são analisados os resultados obtidos com a implementação do sistema e será possível determinar as principais funcionalidades do sistema sendo dessa forma possível a avaliação da sua viabilidade.

No capítulo 6 são apresentadas a principais conclusões que foram possíveis obter e também sugestões que poderão ser utilizadas para melhorar o sistema no futuro.

2

ESTADO DE ARTE

2.1 RECICLAGEM EM PORTUGAL

Como já foi referido anteriormente o crescimento exponencial da população levou a um aumento do consumo recursos utilizados. Como muitos desses recursos existem em quantidades reduzidas, a reciclagem começou a ser encarada como um princípio que todos os cidadãos devem por em prática.

A reciclagem de resíduos ao tornar-se uma prática comum, permite reduzir o consumo de matéria primas e consequentemente aumentar o tempo de vida de um aterro sanitário. Ao mesmo tempo diminui a quantidade de resíduos que são depositados em aterros e os custos associados a esse processos. Do ponto de vista ambiental a redução da quantidade de resíduos em aterros ou incineração dos mesmos provoca uma menor contaminação da terra, ar e agua. No entanto a reciclagem não apresenta só vantagens. Os custos associados a este processo são mais elevados se, quando comparados com a recolha de resíduos não-sorteada. A criação de um sistema eficaz requer investimentos em novas estruturas específicas para recolha seletiva e separação de resíduos e custos adicionais de transportes.

Em Portugal, o processo de recolha de resíduos dos ecopontos tem-se mantido praticamente o mesmo desde o seu aparecimento. Isto é, existe uma viatura específica para cada um dos ecopontos, que de forma regular passa pelos ecopontos, não existindo uma forma concreta de se saber quando e onde um certo ecoponto deve ser recolhido. Assim, as viaturas de recolha de resíduos têm que passar pela localização de cada um dos ecopontos, de forma a poderem determinar se a recolha é ou não necessária. Facilmente se conclui, que o processo de recolha de resíduos dos ecopontos atualmente implementado, bastante ineficiente e dispendioso.

Outro aspeto que se deve ter em conta é facto de não existir um incentivo

para reciclar em Portugal. É possível verificar casos na Europa em que a população é incentivada a reciclar, por exemplo o caso do Reino Unido que atribuiu um imposto às autoridades locais por cada tonelada depositada no aterro sanitário[5] e da Alemanha em que existem cidades que reduzem a taxa local caso reciclem[6][7].

2.2 PLATAFORMA MÓVEIS

Um dispositivo móvel tem as seguintes características é portátil, pessoal, rápido, fácil de se usar e com conexão de rede. Estas características tornam estes dispositivos apelativos para o público em geral. As grandes empresas têm visto os dispositivos móveis como boas oportunidades de negócios, por isso é possível verificar um investimento cada vez maior nesta área. A evolução da tecnologia permitiu a incorporação de cada vez mais periféricos nestes dispositivos, como por exemplo a câmara fotográfica, GPS, NFC, Bluetooth, Wi-Fi e ligação de dados. Permitindo assim que telemóveis, PDAs e outros dispositivos eletrónicos começassem gradualmente convergir para um só: o smartphone. Assim o smarthphone consegue apresentar funcionalidades de dispositivos de entretimento móvel e de outros dispositivos eletrónicos sem-fios.

Devido a um forte crescimento do mercado de *smartphone* e *tablets* o desenvolvimento de software para dispositivos móveis tem aumentado de forma significativa nos últimos anos. Existe uma vasta variedade de aplicações para *smartphone* e *tablets* devido à interface gráfica do sistema operativo que torna o seu acessível e intuitivo. Para além disso conseguem incorporar elementos que normalmente não se encontram em portáteis ou *desktops*[8].

2.2.1 *Arquitetura do dispositivo móvel*

Um dispositivo móvel é constituído por um conjunto de componentes de hardware e software. Estes componentes de hardware incluem um microprocessador, ROM, RAM, armazenamento de expansão, interfaces de rede, bateria e um *display*. O dispositivo é controlado por um sistema embebido que realiza uma serie de funções que controlam qualquer dispositivo eletrónico ou equipamento desde sistema indústrias ou dispositivos de uso doméstico.

Ao desenvolver de uma aplicação é necessário ter em consideração os atributos do *display* como: tamanho, resolução, profundidade de cor, retroiluminação e consumo energético. O *display* do dispositivo móvel representa uma significativa quantidade do consumo energético. Então os sistemas operativos devem empregar técnicas que permitam um uso baixo do *display* com o objetivo de reduzir o consumo energético. A figura 2 mostra uma representação da arquitetura dos dispositivos móveis[8].

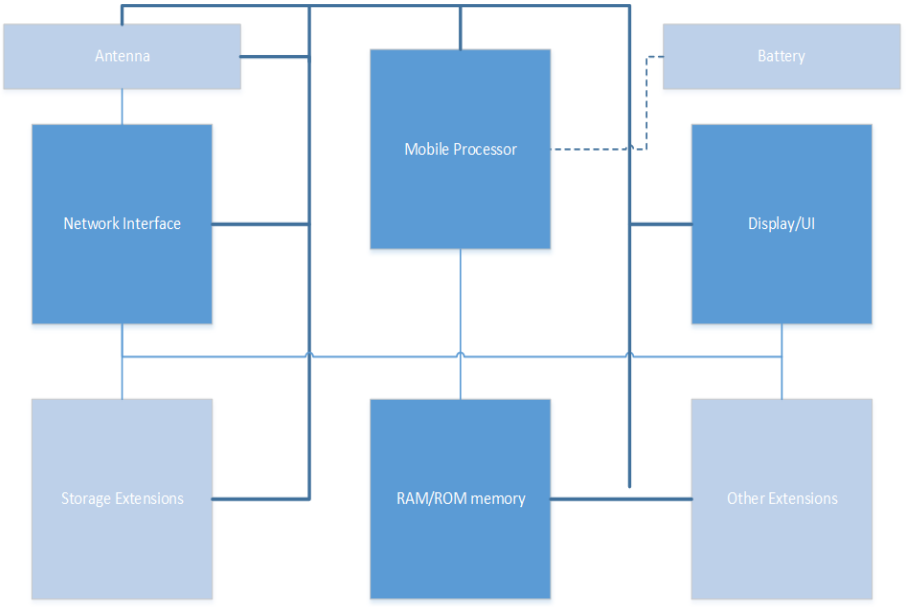


Figura 2: Arquitetura dos dispositivos móveis.

2.3 QR CODE

O código de barras sendo um símbolo unidimensional encontra-se bastante limitado na informação que é capaz de armazenar. Os dados no código de barras são representados por linhas pretas, com diferentes espessuras com espaços entre eles. A segurança do código de barras pode ser considerada questionável já que é facilmente obtida. No entanto depende de um dispositivo específico que não se encontra na maioria dos telemóveis ou *tablets*.

O código de resposta rápida (QR Code) é um símbolo bidimensional inventado pela Denso, uma das companhias do grupo da Toyota em 1994. Inicialmente com o objetivo de ser usado pela indústria automóvel. Os códigos QR representam uma evolução do tradicional código de barras, os dados existentes nestes não

são tão facilmente compreensíveis sem conhecimento prévio das características da sua formação. Dessa forma acrescentando uma camada de segurança e impedem o pessoal não autorizado de obter informação importante. Uma maior densidade permite um maior número de caracteres especiais sendo que suporta até 7089 caracteres numéricos e 4296 alfanuméricos. O facto de a patente ter sido libertada para domínio público juntamente com o seu baixo custo de produção, visto que podem ser impressos em qualquer suporte em papel, permitiu uma expansão da sua utilização como é possível verificar pelo seu uso em revistas, livros, embalagens e muitos outros produtos. O código QR apresenta-se assim como uma solução mais portável e com melhor disponibilidade e segurança, já que o único hardware adicional necessário é a câmara. Por estas razões os códigos QR tem ganho popularidade e aceites pelo público geral e praticamente usados como forma de comunicação entre o mundo físico e digital ao traduzir informação de um para o outro[9].

Figura 3: QR Code[10]

Como é possível verificar na figura 3, o código QR tem a estrutura de uma matriz quadrada. Consiste em duas partes principais: funcionalidade e áreas de informação. Sendo também dívido em padrões:

- Padrão de descoberta: permite a deteção da posição do código QR e encontra-se localizado em três cantos do símbolo, de forma a encontrar detetar corretamente a posição, tamanho e ângulo dele.
- Padrão de alinhamento: usado para a correção de distorções principalmente as distorções não lineares.

- Padrão de tempo: consiste em padrões branco e preto arranjado alternadamente. É usado para a correção da célula de informação quando distorcido ou quando existe um erro.
- Zona Limpa: espaço branco fora do símbolo que torna a deteção do símbolo mais fácil entre a restante informação quando a fazer *scan* com a câmara.
- Área de dados: os dados do QR codes são codificados em binários 0 e 1. Estes números são convertidos em células pretas e brancas sendo depois organizadas.

Figura 4: Padrões do QR Code.[11]

2.4 WEB SERVICES

O paradigma SOA (*Service-Oriented Architecture*) descreve um conjunto de instruções de desenho que atuam como um serviço intermediário. Este componentes implementam uma funcionalidade especifica e publicam uma interface de comunicação pela qual podem ser acedidas.

Um serviço Web trata-se de uma serviço oferecido por um dispositivo eletrónico para outro, cuja comunicação é feita através da *internet*. A sua tecnologia foi construída pelos princípios do paradigma do SOA ao serem aplicados diretamente em aplicações da *internet*. Num serviço Web, o protocolo HTTP, originalmente direcionado para a comunicação homem-máquina, é utilizado para comunicação máquina-máquina, mais especificamente para a transferência de ficheiros de formato de leitura, como por exemplo XML e JSON. Em prática, o serviço Web normalmente providência uma interface baseada na *internet* para base de dados-servidor, utilizando por exemplo outro servidor de *internet*, ou uma aplicação móvel, que oferece uma interface para o utilizador. Dessa forma fornecendo intercomunicação e interoperabilidade entre aplicações diferentes que se podem encontrar

a correr em plataformas diferentes com a utilização de protocolos padrão para a troca de informação desde que o software seja capaz de interpretar o mesmo tipo de formato, i.e. XML,JSON. Na figura 5 encontra-se representado à arquitetura de um *web service*[12][13] .

Para a comunicação ser possível é necessário estabelecer um conjunto de regras:

- Processo de requisição de dados de um sistema para outro.
- Especificar os parâmetros necessários no pedido de informação.
- A estrutura dos dados produzida.

Estas regras são definidas num ficheiro chamado WSDL. Uma diretoria chamada UDDI define qual o sistema de software que deve ser contactado para qual tipo de dados. Quando um sistema de software necessita de uma informação, terá que ir à UDDI e descobrir que outro sistema pode contactar para receber essa informação. Uma vez identificado qual o sistema que se deve contactar, então contacta esse sistema usando um protocolo especial denominado SOAP(*Simple Object Access Protocol*). O serviço distribuidor do sistema irá primeiro validar o pedido ao verificar o ficheiro WSDL e depois executa o pedido e envia a informação utilizando o protocolo SOAP.

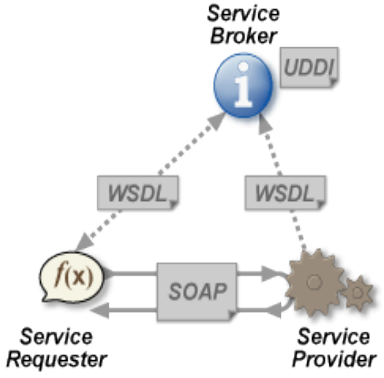


Figura 5: Arquitetura de um *web service* [14]

2.4.1 *SOAP*

SOAP é um protocolo especificado para troca de informação estruturada em implementação de serviços Web em rede. Para tal as mensagens são codificadas de acordo com o protocolo utilizado, normalmente o protocolo HTTP e para descrever o serviço é utilizado o WSDL.

Este protocolo é baseado em XML e consiste em três partes:

- O envelope, que define a estrutura da mensagem e a forma para o seu processamento.
- Conjunto de regras de codificação para expressar instâncias dos tipos de dados definidos pela aplicação.
- A convenção que representa os procedimentos de chamadas e respostas.

Uma mensagem SOAP é um documento XML constituído pelos seguintes elementos: envelope, cabeçalho, corpo e pela falha. Sendo possível verificar um exemplo destas mensagens na figura 6.

O envelope define onde a mensagem começa e acaba. O cabeçalho é um recipiente extensível para a infraestrutura da informação da camada da mensagem e pode ser usado para *routing* e configuração da qualidade do serviço. O corpo da mensagem contém o a informação XML correspondente a mensagem a ser enviada. A falha disponibiliza a informação relativa a erros que ocorreram durante o processamento da mensagem. Estes elementos são declarados no *namespace* padrão sendo obrigatório enviar apenas o cabeçalho e o corpo[15].

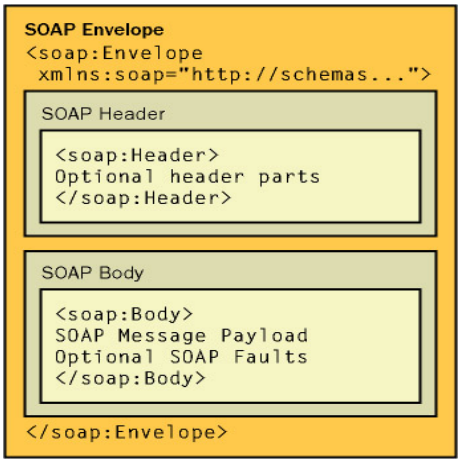


Figura 6: Mensagem SOAP [16]

2.4.2 *REST*

REST é um estilo de arquitetura que incorpora um conjunto de princípios que determinam como os recursos da rede devem ser definidos e abordados. Estas aplicações ou arquiteturas são referidas como RESTful ou REST-style. Sistemas RESTful comunicam através de operações HTTP (GET,POST,PUT ou DELETE) semelhantes aos utilizados pelos navegadores de *internet*, para enviar dados para servidores remotos. REST também permite diferentes tipos de dados sendo que o formato de dados mais comum é o JSON. JSON é a representação textual dos objetos javascript, distribuindo este em listas ordenadas de valores, onde cada um é um pares nome/valor, representando um formato de dados mais compactos e leve, comparativamente com o formato de dados XML[17]. A figura 7 demonstra o processo de comunicação utilizado pela arquitetura REST.

Uma aplicação que utilize a arquitetura REST é caracterizada pelo seguintes pontos:

- Interface uniforme- Recursos individuais são identificados pelos URLs. Cada mensagem enviada entre o cliente e o servidor e contém informação suficiente para descrever a forma em que é processada.
- Interações sem argumentos – nenhum do contexto dos clientes é armazenado no lado do servidor durante o pedido. Toda a informação necessária para cumprir o pedido é contida no URL.
- *Cacheable* - Clientes podem guardar as respostas prevenindo o cliente de enviar informação não apropriada em resposta a pedidos seguintes.
- Cliente- Servidor- O cliente e servidor encontram-se separados um dos outros, permitindo que o cliente não se preocupe com o armazenamento de dados e dessa maneira aumentando a portabilidade do código do cliente. Já o servidor não ser preocupa com interferências do cliente e assim tornando o servidor mais simples e fácil de crescer.
- Camada Intermediária – A qualquer altura o cliente não se apercebe se se encontra conectado o servidor final ou a um intermediário. A camada intermediária ajuda a executar as políticas de segurança e melhora a capacidade de crescimento do sistema.

- Código a pedido - Uma opção que permite ao servidor temporariamente estender a funcionalidade do cliente ao transferir código executável.

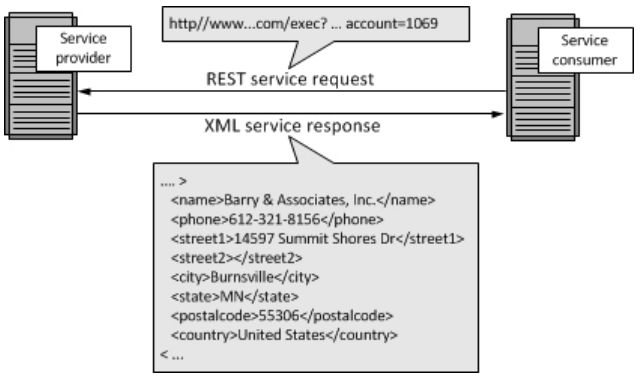


Figura 7: Mensagem REST[18]

2.5 EXEMPLOS DO MERCADO

Atualmente, existem vários projetos que tentam resolver este problema. De seguida são descritos cada um deles, bem como as suas vantagens e desvantagens.

2.5.1 *GePOR*

O GePOR é uma aplicação informática que permite a gestão de pontos de recolha de resíduos e o cálculo de rotas otimizadas. A recolha é feita com o auxílio de uma aplicação Android que permite registar o grau de ocupação de um certo ecoponto. A informação registada pode ser vista através de um *website*, que permite analisar os ecopontos, fornecendo gráficos com os dados da recolha e cálculo de rotas. No entanto, este sistema apresenta algumas desvantagens: não consegue resolver o problema dos gastos referentes à determinação do estado dos ecopontos, o registo da taxa de ocupação do ecoponto é feito no local, sendo que a única maneira de manter a informação atualizada continua a ser a passagem de um empregado em cada um desses ecopontos, de forma a verificar a sua condição, sendo que os gastos de transporte e mão-de-obra se continuam a verificar. A aplicação Android não é acessível, já que o utilizador padrão é um empregado da empresa[19].

2.5.2 *iEcoSYs*

Este sistema permite identificar o lixo produzido individualmente utilizando etiquetas RFID. As etiquetas encontram-se presentes nas sacas de lixo (*iBags*). Aquando da deposição do lixo, o centro de reciclagem identifica e pesa cada saco e envia os dados coletados para um servidor, utilizando a comunicação ZigBee *standard*.

Este sistema permite um controlo preciso das quantidades de lixo depositado e de quem o depositou. No entanto, os sacos têm que ser comprados individualmente, o que aumenta o custo e têm de ser depositados em ecopontos próprios (*iEcoPoint*) para o seu rastreio[20].

O sistema EcoWise apresenta algumas vantagens relativamente às propostas:

- Baixo Custo – Tanto para o utilizador, como para a empresa.
- Acessibilidade – O utilizador só tem que fazer download da aplicação e os funcionários só precisam de aceder ao site.
- Universalidade – Funciona para qualquer ecoponto.
- Rapidez – Recolha da informação é rápida

O sistema é composto por três partes principais: a aplicação, a comunicação e o *Website*.

2.6 APLICAÇÃO MÓVEL

Para o desenvolvimento de aplicações para dispositivos móveis existem duas plataformas de desenvolvimento mais popularmente utilizadas que são:

- Android – Um sistema operativo baseado em Linux do Google;
- iOS – Plataforma da Apple;

2.6.1 *Android*

O Android, é um sistema operativo móvel de código aberto baseado no *kernel* do Linux ,que facilita o desenvolvimento de aplicações para sistemas móveis utilizando as biblioteca Java desenvolvidos pela Google. A plataforma Android não só providencia o próprio sistema operativo móvel, incluído no ambiente de desenvolvimento, como também providencia uma máquina virtual customizada (*Dalvik Virtual Machine*), para que as aplicações possam correr e também atuar como intermediário entre o código e o sistema operativo[21]. A arquitetura do Android encontra-se apresentada na figura 8.

Normalmente os aparelhos Android funcionam por bateria, por isso uma das preocupações no desenvolvimento de uma aplicação é o consumo energético. O Android está desenhado de modo a manter o gasto enérgico a um mínimo, sendo isto conseguido através do controlo da memória RAM. Quando uma aplicação não se encontra em uso, ela entra em suspensão na memória e caso a memória chegue o seu limite, começa a terminar as aplicações que estejam suspensas à mais tempo[22].

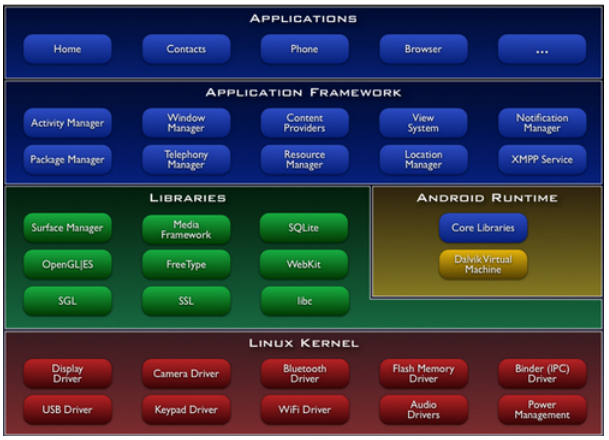


Figura 8: Arquitetura do sistema Android[23]

2.6.2 *iOS*

iOS é o sistema operativo utilizado por diversos aparelhos da Apple, sendo um dos mais importantes o iPhone. O sistema operativo controla o *hardware* do dispositivo e providência as tecnologias necessárias para o desenvolvimento de aplicações nativas. O kit de desenvolvimento de *software* do iOS (SDK) contém

as ferramentas necessárias para o desenvolvimento, instalação, teste e execução de aplicações nativas.

As aplicações são escritas em Objective-C utilizando a biblioteca Cocoa Touch. O Objective-C é a extensão da linguagem C, enquanto o Cocoa Touch é uma *framework* de interface para o utilizador, facilitando o desenvolvimento de aplicações em iOS. O desenvolvimento para iOS requer um computador a correr Mac OSX. A arquitetura do iOS é visível na figura 9[24].

Objective-C apresenta-se como uma alternativa ao C# e ao Java que são bastante semelhantes em sintaxe [25].

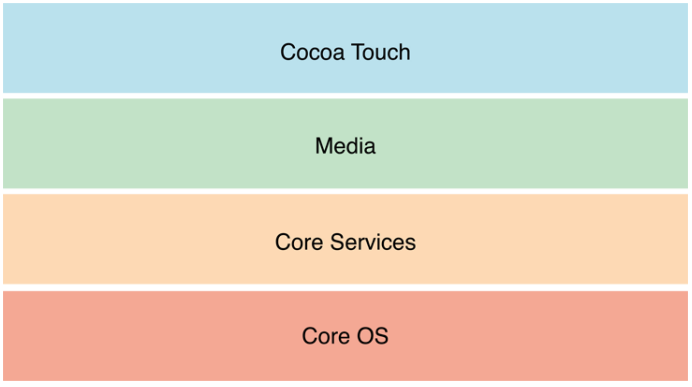


Figura 9: Arquitetura do sistema iOS[26]

2.7 COMUNICAÇÃO

2.7.1 *Cloud*

Computação em nuvem é um modelo que permite conveniência no acesso à rede *on-demand* a um grupo de recursos computacionais configuráveis, que podem ser rapidamente fornecidos e libertados com um esforço mínimo de gestão ou interação com o fornecedor do serviço. Computação em nuvem refere-se tanto às aplicações que fornecem o serviço através da *internet* e ao hardware e software de sistemas nos centros de dados que prestam esses serviços [21]. A figura 10 apresenta uma representação da arquitetura da Cloud. Os serviços fornecidos são variados desde bases de dados, armazenamento, plataformas, segurança, entre outros, sendo que podem ser geridos de forma dinâmica conforme as necessidades do utilizador.

O *Cloud computing* apresenta diversas vantagens:

- Remove a necessidade de planeamento dos recursos necessários já que existe disponibilidade de recursos na *Cloud*.
- Permite a alteração dos recursos de hardware conforme as necessidades dessa forma permitindo as empresas não realizar compromissos de longo termo.
- Permite pagar os recursos em função do tempo de utilização

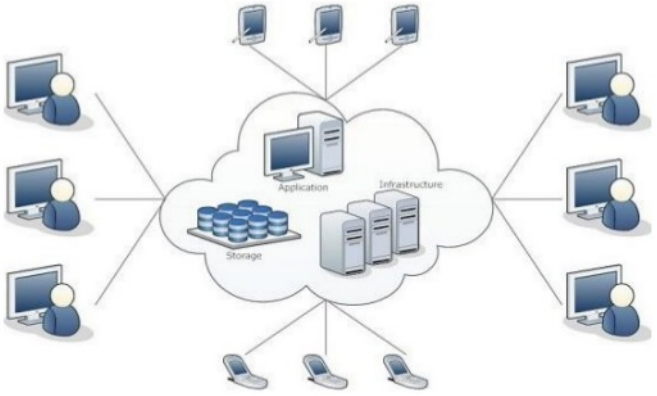


Figura 10: Arquitetura da Cloud[27]

2.8 WEBSITE

2.8.1 *AngularJS*

Para desenvolver o website é utilizada a *framework* Angular.js. Esta é uma *framework* JavaScript *open-source* mantida pela Google, que procura resolver muitas das dificuldades existentes no desenvolvimento de aplicações web de uma única página, utilizando o padrão de arquitetura de software *Model View Controller* (MVC).

Assim, a *framework* disponibiliza funcionalidades que permitem extender o tradicional HTML para apresentar conteúdo dinâmico, através da ligação bidirecional de dados (*Two-way data-biding*), que permite a sincronização automática dos modelos e das vistas através de controladores. A figura 11 mostra o seu funcionamento. Esta ferramenta é a mais notável já que permite dispensar o *backend* do servidor das responsabilidades de modelação. Em vez disso os modelos são

rendarizados em pleno HTML de acordo com os dados contidos no *scope* definidos no modelo. O serviço *scope* deteta mudanças na secção do modelo e modifica as expressões HTML nas vistas através do controlador. Da mesmo maneira qualquer alteração na vista é refletida no modelo.

Como resultado o Angular.js abstrai-se da manipulação explícita do DOM (Modelo de objeto de documentos) com o objetivo de melhorar capacidade de teste e performance[28][29].

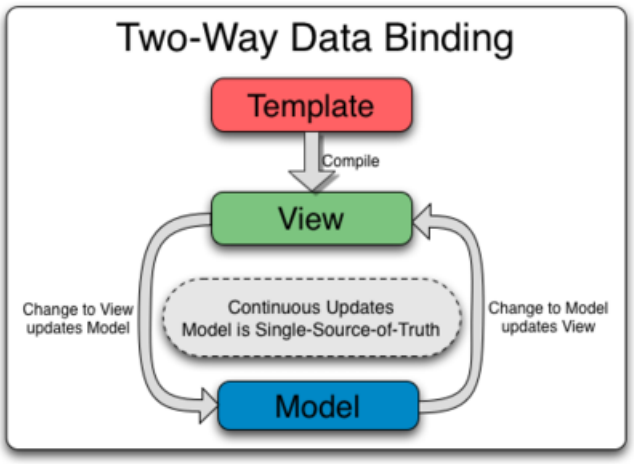


Figura 11: Funcionamento do *Two-way data binding[30]*

2.9 BASE DE DADOS

2.9.1 *PostgreSQL*

O PostgreSQL é um sistema de manuseamento de base de dados relacionados ao objeto cujo objetivo é ser *standards-compliant* e extensível. Difere-se com o suporte para a funcionalidade de base de dados relacionais e/ou orientados ao objeto que são altamente requisitados e essenciais, como o suporte completo para transições fiáveis. Devido à sua tecnologia o PostgreSQL é capaz de lidar com muitas tarefas eficientemente. É altamente programável e extensível com procedimentos personalizados denominados "procedimentos armazenados". Estas funções podem ser criada para simplificar a execução de operações complexas, repetitivas e necessárias da base de dados [31].

2.9.2 *MongoDB*

MongoDB é uma base de dados *open-source* desenvolvida pela MongoDB,Inc. MongoDB armazena os seus dados em documentos semelhantes a JSON que variam em estrutura. Informação relacionada é armazenada junto de forma a permitir um acesso mais rápido através da linguagem *query* utilizada pelo MongoDB. O uso de *schemas* dinâmicos permite a criação de registo sem primeiro definir a estrutura, como os campos que contém ou o tipo de valores utilizados. Um exemplo destes *schemas* é exibido na figura 12. É possível a mudança de uma estrutura dos registos (denominados de documentos) simplesmente ao adicionar novos campos ou eliminando aqueles já existentes. O modelo dos dados apresentam a capacidade de representar relacionamentos hieráticos, de armazenar *arrays* e outras estruturas mais complexas facilmente.Os documentos de uma coleção não necessitam de conter um conjunto de campos identifico e a desnormalização dos dados é comum.O MongoDB foi desenvolvido com alta disponibilidade e escalabilidade em mente[32].

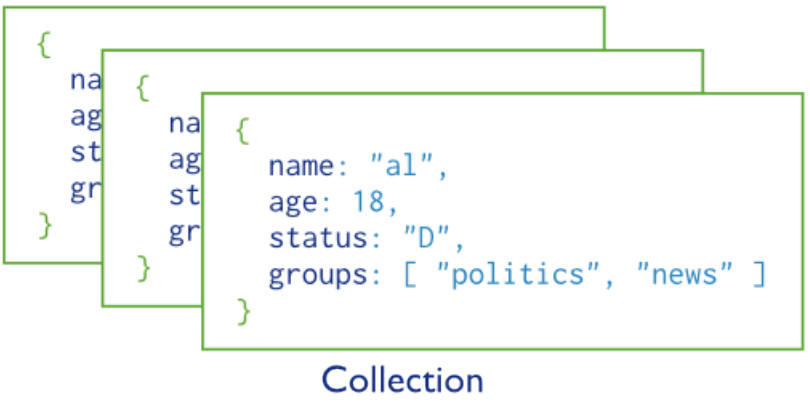


Figura 12: Exemplo de um Schema[33]

2.9.3 *MySQL*

O MySQL trata-se de um sistema de manuseamento de base de dados relacional *open-source* que é desenvolvido, distribuído e suportado pela Oracle Corporation. Como os outros sistemas relacionais, o MySQL armazena os seus dados em tabelas e usa linguagem query estruturada(SQL) para acesso a base de dados. O *schema* da base de dados é predefinido tendo em conta os requerimentos necessários e são atribuídas as regras que organizam as relações entre os campos nas tabelas.

No MySQL a informação relacionada pode ser armazenada em tabelas diferentes mas é associadas através do uso de juntas dessa maneira evitando a duplicação dos dados, como é possível verificar na figura 13[34].

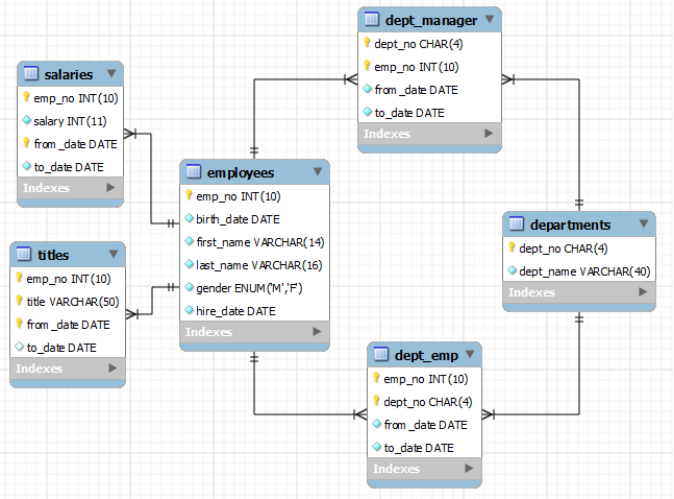


Figura 13: Exemplo de uma base de dados relacional[35]

2.9.4 *Comparação entre o MongoDB e o MySQL*

Uma comparação entre MySQL e o MongoDB é em muitas maneiras a a distinção entre base de dados relacional e a não-relacional, que se encontra representada na figura 14. Cada um apresenta vantagens em certos cenários e desvantagens para outras. Ambas apresentam características especificas que que as tornam mais indicadas para o tratamento de certos tipos de dados.

O MongoDB apresenta uma performance bastante boa em situações em que a pressão de escrita é bastante elevada mas a integridade dos dados não é importante. O modelo de dados flexível permite evoluir o *schema* da base de dados dependente dos requerimentos necessários. Em termos de escalabilidade o MongoDB apresenta vantagem já que o permite fazer com relativa facilidade enquanto que no MySQL é normalmente necessário trabalho significativo e customizado.

Em casos de aplicações que requerem complexas, multi-filas transações o MySQL é a melhor opção. Para além disso apresentam uma maneira fácil de representar modelos de dados, uma linguagem fácil de usar que permite uma boa manipulação de dados e apresenta uma boa integridade e segurança nos dados. As

limitações são na sua dificuldade de escalar e no elevado tempo de carregamento de dados que provoca uma diminuição na performance [36][34].

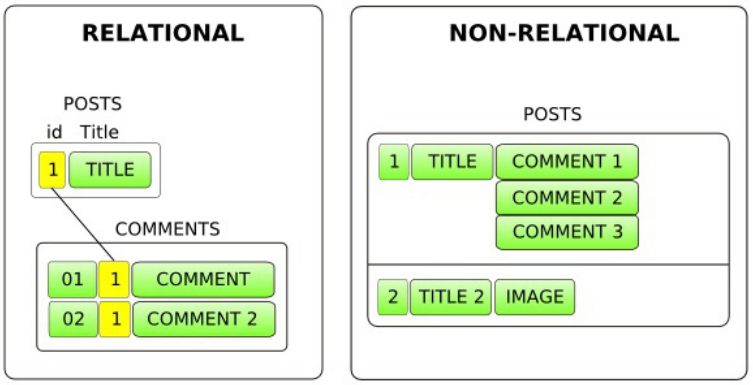


Figura 14: Comparação entre base de dados relacional e não-relacional[37]

2.10 WEBSERVICE

2.10.1 *NodeJs*

Node.js é um ambiente de execução *cross-plataform* para aplicação de rede, do lado do servidor. Programação baseada em eventos para servidores de *internet*, possibilitando o desenvolvimento de servidores *web* em Javascript[38]. O Node.js corre *javascript* utilizando o motor V8 criado pela google. O uso deste motor permite a criação de um ambiente de execução que puxa o Javascript do servidor para o cliente rapidamente. O V8 traduz *javascript* para linguagem maquina nativa,em vez de trabalhar em *overtime* para o traduzir em *bytecode*. Trabalha de uma forma assíncrona utilizando para tal uma função denominada de Event Loop que garante que os dados não estão constantemente a ser requisitados mas simplesmente transmite quando eles existem. Enquanto que o *Event Loop* trata de todas as tarefas assíncronas, Node.js pode continuar a correr o programa normalmente e dessa forma deixando o trabalho pesado de parte. A velocidade providenciada pelo motor V8 combinado com a programação assíncrona são as principais funcionalidades que dão ao Node.js a capacidade de respostas que tem[24].

2.10.2 *PHP*

PHP também conhecido como *PHP:HyperText Preprocessor* é uma linguagem de escrita do lado do servidor usado principalmente para o desenvolvimento em web no entanto verifica-se o seu uso como linguagem de programação de uso genérico. O código PHP poderá ser encontrado embebido em código HTML, ou usado em combinação com vários sistemas de *web templates*, sistemas de manuseamento de conteúdo *web* e *web frameworks*. Código PHP é normalmente processado por um interpretador de PHP implementado como um modulo no servidor *web* ou como um executável CGI. O servidor *web* combina os resultados do código PHP interpretado e executado, que pode ser qualquer tipo de dados, incluindo imagens, com a pagina *web* gerada. Este código pode ser executado através da interface da linha de comandos e pode ser usado para implementar aplicação gráfica autónoma. Na figura 15 é possível verificar essa diferença no tratamento de tarefas[39].

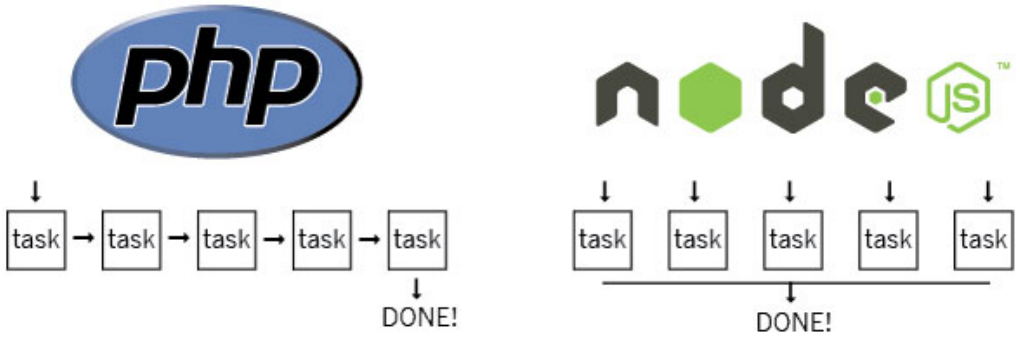


Figura 15: Comparação do tratamento de tarefas entre PHP e Node.js[40]

2.10.3 *Comparação entre o PHP e o Node.js*

O PHP é conceptualmente mais simples e acessível para novos desenvolvedores devido ao seu grande suporte. O código pode ser escrito em qualquer ficheiro desde que este tenha uma extensão .php sendo que o URL é utilizado para mapear o ficheiro para o *browser*.

A implementação do código num servidor web é mais fácil com PHP já que existem diversas empresas de hospedagem web que apresentam suporte para esta linguagem. Com o Node.js é diferente já que as aplicações no lado do servidor correm permanentemente sendo necessário um servidor cloud/virtual para ser im-

plementado. Quanto às ferramentas de desenvolvimento ambos apresentam um bom conjunto de IDEs, debuggers e outras ferramentas no entanto o Node.js apresenta uma vantagem: o npm.

Esta ferramenta permite a instalação e manuseamento de dependências, definição de variáveis configuráveis, definição de scripts. Em termos de performance o Node.js apresenta vantagem devido a apresentar menos dependências, um interpretador mais pequeno e rápido, um I/O não bloqueador e baseado em eventos que aumenta consideravelmente a velocidade da suas aplicações[41].

A figura 16 apresenta uma comparação mais detalhada dos componentes existentes no Node.js e no PHP.

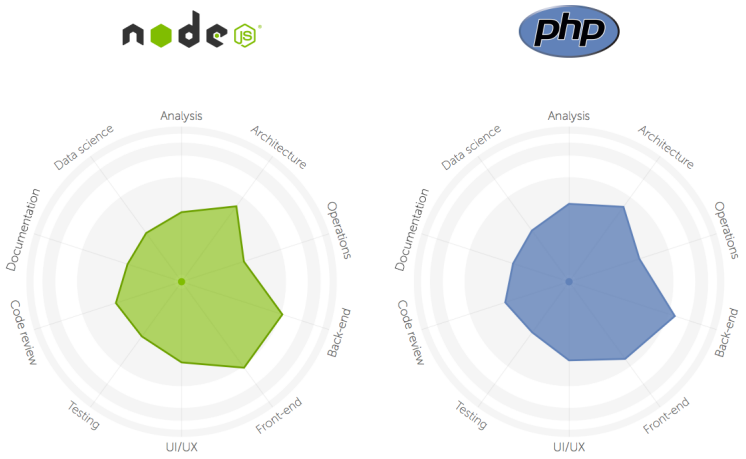


Figura 16: Comparação de componentes entre PHP e Node.js[42]

3

ANALISE DOS REQUISITOS

No capítulo anterior foram descritos os principais conceitos relacionados com o tema e, dessa forma, foi possível determinar os diferentes métodos existentes e as diferentes direções que podiam ser tomadas de forma a desenvolver o sistema como planeado. Neste capítulo é descrita a arquitetura do sistema e como cada módulo vai ser desenvolvido individualmente. Mais especificamente, quais foram os métodos escolhidos e os motivos por detrás dessa escolha, bem como o papel e consequente importância para o sistema em geral.

O sistema EcoWise, como referido anteriormente, é constituído por três módulos principais:

- Aplicação Móvel
- *Web service*
- *Website*

3.1 SISTEMA ECOWISE

Esta dissertação visa o desenvolvimento de um sistema que permita a um utilizador comum relatar os estados dos ecopontos na sua região, de forma a providenciar às empresas responsáveis pela sua recolha mais informações de forma a tornar todo processo mais eficiente. Para tal é utilizada uma aplicação Android que permite a qualquer pessoa utilizando um dispositivo móvel submeter a informação relacionada com os ecopontos utilizando as funcionalidades existentes. O *website* permite gerir a informação recolhida, disponibilizando às empresas responsáveis

pela manutenção dos ecopontos, o estado dos mesmos para uma monitorização mais apta e estruturada. Por fim, o sistema disponibiliza um serviço Web que é responsável pela interface entre a aplicação Android e o *website*. Uma visão geral do sistema encontra-se representado na figura 17.

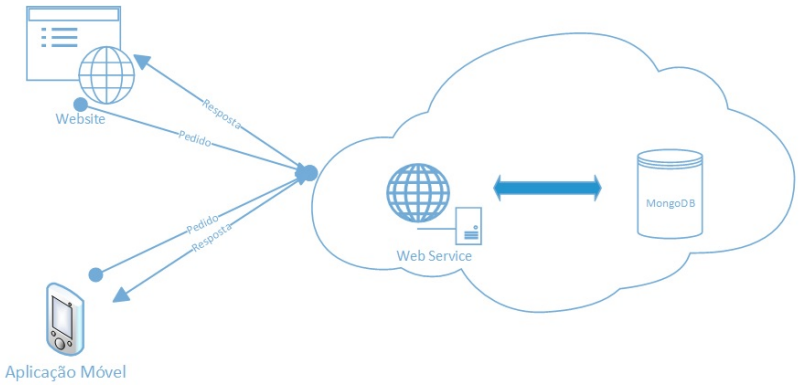


Figura 17: Visão geral do Sistema

3.1.1 *Requisitos do sistema*

Como determinado no Capítulo 1, a arquitetura do sistema e os seus módulos devem cumprir os seguintes requisitos:

Requisitos funcionais

- A Aplicação Android deve disponibilizar as funcionalidades de captura de informação sobre o estados dos ecoponto e consequentemente comunicar com a base de dados através do serviço web, de forma a possibilitar o armazenamento da informação da informação recebida.
- O *Web Service* deve satisfazer as necessidades das aplicações web e Android através da requisição de recurso da base de dados recorrendo à utilização do protocolo REST.
- O *Website* deve permitir a recolha, organização dos dados pretendidos pelo utilizador e disponibiliza-los de uma forma fácil e compreensível.

Requisitos não funcionais

- A Aplicação móvel deve ser desenvolvida para a plataforma Android utilizando o IDE do Android Studio .
- Para a implementação do *website* deve ser utilizada a *framework AngularJS*.
- O *Web sevice* deve ser implementado utilizando o Nodejs com o MongoDB como sistema de gestão de base de dados.

3.2 CASOS DE USO E DESCRIÇÃO

O sistema é composto por um conjunto de ações divididas por vários módulos que se encontram associadas a dois tipo de atores: Utilizadores e Administradores. No diagrama de casos presente na figura 18 é apresentada uma visão geral das funcionalidades do sistema.

O nível de autenticação "Utilizador"refere-se às pessoas que tem acesso à aplicação móvel e às suas funcionalidades. Estas tem como objetivo a atualização do estado dos ecopontos de uma forma rápida e acessível através dos processos existentes. Os dados são enviados posteriormente para o *web service* que determina quais as ações que são necessárias realizar e a resposta pretendida.

O nível de autenticação "Administrador"permite o acesso ao *website* e às suas funcionalidades. Através dele pode-se obter informações relacionadas com os ecopontos que se encontram registados e realizar a manutenção do sistema de uma forma flexível e fácil.

3.3 APLICAÇÃO MÓVEL

Tanto o iOS como o Android uma plataforma de desenvolvimento estável e flexível que disponibilizam as funcionalidades necessárias para a criação de uma aplicação que cumpra com os requerimentos definidos. Embora esteja definido nos requisitos não funcionais do sistema, que a plataforma a usar é o Android, um dos fatores que levam a que muitas empresas optem por desenvolver primeiro aplicações móveis para Android, é o fator de mercado, comparativamente com o IOS. Neste ponto é que se verifica a maior diferença entre os dois sendo que é possível

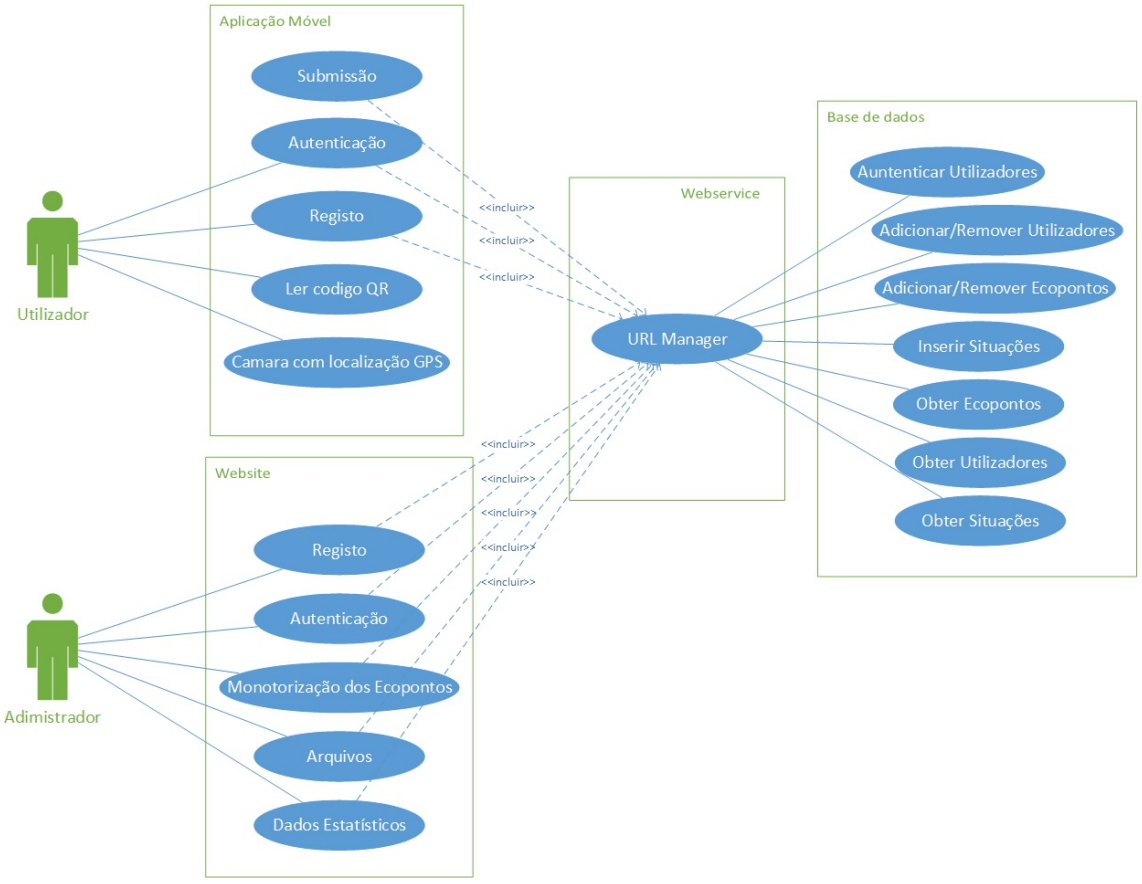


Figura 18: Diagrama de casos de uso do sistema

verificar que um domínio por parte do Android (cerca de 82%) relativamente a iOS (Cerca de 15%). Esta liderança do mercado deve-se principalmente à sua flexibilidade, *smarthphones* mais baratos e pelo facto de ser *open-source*. Uma vez que se pretende que a aplicação seja utilizada pelo maior número de pessoas a escolha da plataforma móvel Android torna-se a opção mais lógica.

A aplicação será desenvolvida utilizando o Android Studio, o IDE principal utilizado pela Google, para o desenvolvimento de aplicações Android nativas, e apresenta mais funcionalidades que o Eclipse ADT. Entre elas uma maior estabilidade, organização do projeto, rapidez e uma interface de utilizador mais fácil e customizável.

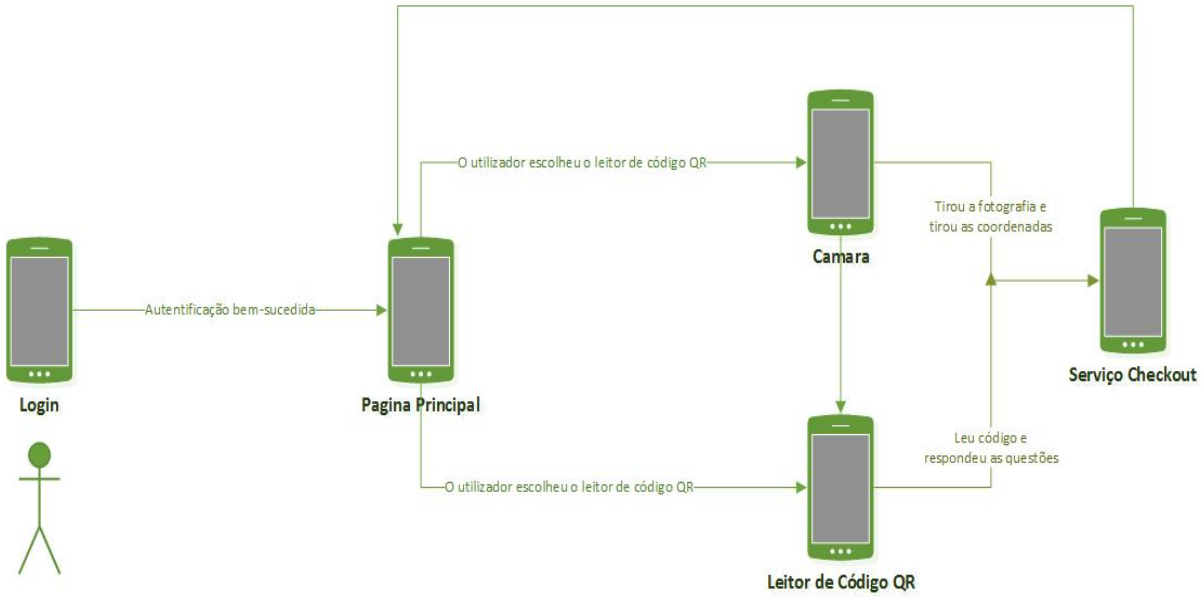


Figura 19: Visão geral da Aplicação Móvel.

3.3.1 *Visão geral da Aplicação*

Como é possível visualizar na figura 19, a aplicação *Android* é estruturada da seguinte maneira:

- Página Inicial – Permite o registo de novos utilizadores e a autenticação dos já existentes.
- Página Principal – Permite aceder às funcionalidades da aplicação: o código QR e a câmara. A Página Principal permite também visualizar outros tipos de informações , tais como: o perfil e as definições do sistema.
- Leitor de código QR – Acede a câmara dando ao utilizador a possibilidade de ler código QR que pretende. Aquando de uma leitura positiva aparece um breve questionário de forma a providenciar a informação relativa ao ecoponto que se encontra a ler.
- Câmara - Acede a câmara dando ao utilizador a possibilidade de capturar uma fotografia. Depois de capturar a imagem é verificada a posição GPS atual do utilizador.
- Serviço *Checkout* – Uma vez feita a leitura do código QR ou a captura de uma imagem , é apresentado o serviço de *checkout*. No *checkout* é preparada a mensagem para ser enviada para o servidor. Este processo é feito em

background de forma a permitir ao utilizador continuar a utilizar aplicação de forma livre.

3.3.2 *Página Inicial*

Na figura 19, é possível verificar que a atividade principal da aplicação móvel é a página principal. O propósito desta página é permitir controlar o acesso à aplicação. Desta forma é possível determinar quem é que usou a aplicação e, assim permitir obter mais informações que podem ser consideradas importantes. A página inicial contém duas opções: o registo de novos utilizadores e a autenticação de utilizadores já registados no sistema.

A opção de registo permite a um utilizador se registar no sistema, providenciando a informação necessária (a palavra-passe e o nome) sendo a informação enviada para o servidor onde se verifica se já existe um utilizador com os dados introduzidos na base de dados. Caso a operação seja bem-sucedida, e o utilizador é registado com sucesso, sendo possível a sua autenticação no sistema.

Por fim, na opção de autenticação o utilizador pode aceder a aplicação com as suas credenciais onde são enviados os dados para o servidor, que consequentemente verificara a existência dos dados de autenticação na base de dados. Se tal ser verificar o utilizador será reencaminhado para a página principal da aplicação e terá acesso às suas funcionalidades. A partir desse ponto a aplicação terá conhecimento do utilizador atual e as ações efetuadas serão atribuídas a esse utilizador. O fluxograma da figura 20 demonstra o processo de autenticação na aplicação móvel.

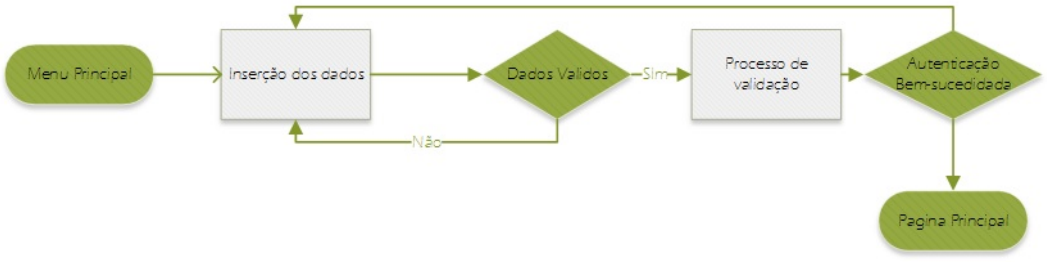


Figura 20: Fluxograma do processo de autenticação

3.3.3 *Código QR*

A utilização do código QR como método de identificação dos ecopontos apresenta várias vantagens: o fato de ser barato, apenas sendo necessário imprimir o código correspondente ao ecoponto num papel e cola-lo, e o fato de se conseguir armazenar e aceder a grandes quantidades de informações de uma forma relativamente fácil e rápida. No entanto obriga o utilizador a descrever o estado dos ecopontos, visto que é vulnerável ao vandalismo já que simples danificações ao papel irá dificultar a sua leitura.

O envio de informação através da leitura de um código QR é bastante simples. Ao selecionar esta opção a aplicação inicia a câmara, que permite a leitura do código. De seguida bastará ao utilizador apontar a câmara ao ecoponto com o código. Em caso de uma leitura bem-sucedida a aplicação mostra o conteúdo lido e irá questiona o utilizador sobre os ecopontos e o seu estado atual. Depois de completar o questionário, é conduzido o processo de criação e envio da mensagem e o utilizador é conduzido de volta para a página principal. O fluxograma da figura 21 demonstra o processo de leitura do QR Code na aplicação móvel.

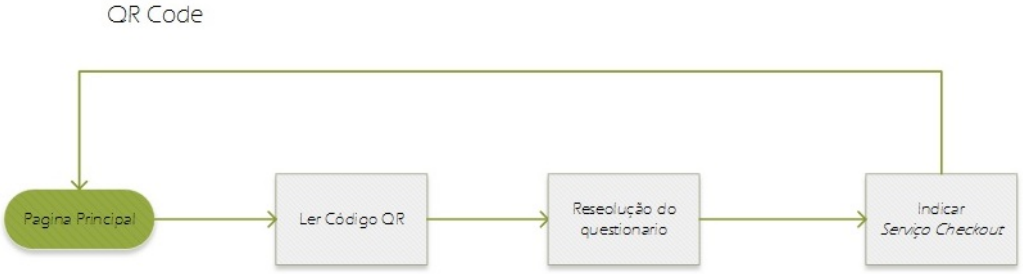


Figura 21: Fluxograma do processo de leitura do QR Code

3.3.4 *Câmara*

A escolha de utilizar a câmara como método de recolha de informações relacionadas com os ecopontos apresenta algumas vantagens: permite uma confirmação visual do estado dos ecopontos e a utilização do GPS para obter à localização da captura de forma precisa , tornando o questionário desnecessário ,comparativamente com o código QR. No entanto apresenta algumas desvantagens: o processo

de triangulação do sinal GPS pode ser bastante demorado dependo do aparelho utilizado e a qualidade da imagem dependerá da câmara e das condições atmosféricas.

Este processo tal como no código QR é iniciado com a preparação da câmara, em modo de captura de imagem. O utilizador poderá tirar a fotografia e visualizar o resultado. Quando se encontrar satisfeito e decidir enviar a aplicação, é obtida a posição GPS em que o utilizador realizou a captura. Caso não seja capaz, a aplicação notifica o utilizador da impossibilidade de obter a informação e serão apresentadas duas opções: utilizar o leitor de código QR ou voltar ao menu principal e de novo. Se conseguir obter as coordenadas GPS então o processo de criação da mensagem é iniciado e o utilizador é conduzido para o menu principal. A figura 22 apresenta o fluxograma do processo da utilização da câmara na aplicação móvel.

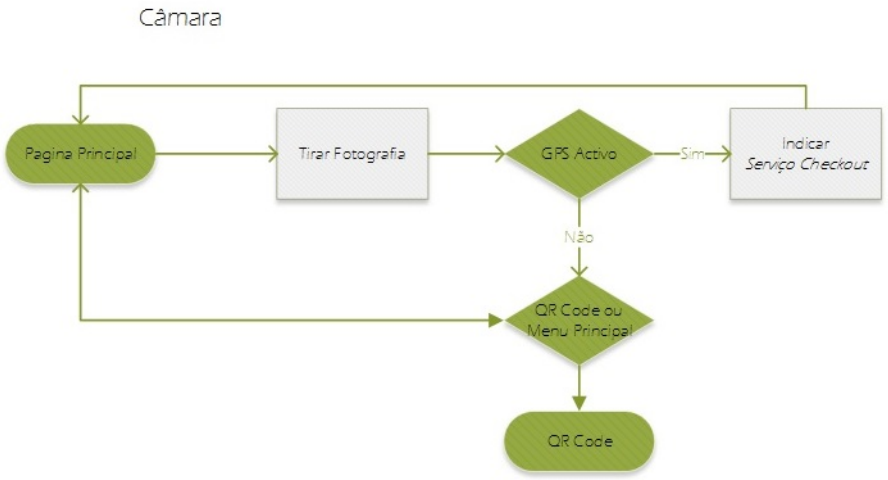


Figura 22: Fluxograma do processo da utilização da câmara

3.3.5 *Serviço Checkout*

Um serviço permite a execução de operações de longa duração e em *background*, não disponibilizando uma interface gráfica ao utilizador. Este pode ser iniciado pela própria aplicação, por outra aplicação, e pode continuar em execução em background mesmo que se mude de aplicação. Desta forma um serviço permite realizar diversas ações sem o conhecimento do utilizador, tais como: comunicação por rede, reprodução de música, leitura/escrita de ficheiros, entre outros.

No caso desta aplicação o papel principal do serviço será a preparação das mensagens e o envio das mesmas para o servidor, estabelecendo assim a comunicação entre a aplicação e o servidor. A escolha de realizar a comunicação num serviço em vez de uma atividade como os restantes módulos, é que desta forma o utilizador não se encontra preso a espera de uma resposta, fornecendo assim uma maior fluidez à aplicação.

O serviço é iniciado quando o utilizador escolhe a câmara ou o leitor de código QR. No entanto as ações que irá realizar serão diferentes dependendo de quem iniciou o serviço.

Assim, no inicio é necessário verificar se o *smartphone* tem acesso à *internet*, sendo absolutamente necessário para a comunicação. Caso contrário a mensagem é armazenada sendo iniciado um *timer* que permite verificar a periodicamente a ligação com a rede e proceder como planeado. É neste ponto que se verificam as diferenças entre os dois métodos.

Isto é, quando a mensagem é proveniente do leitor QR a mensagem é criada a partir, tanto da informação recebida do próprio código QR, como das respostas fornecidas pelo utilizador após a sua leitura. Como esta mensagem é essencialmente constituída por *strings* o processo de criação da mensagem é relativamente simples.

No caso de a mensagem ser proveniente da câmara serão necessários mais alguns passos. Nomeadamente a latitude e longitude obtidos pelo GPS aquando da captura da fotografia. Estes dados são utilizados para realizar geocodificação reversa e obter desta forma a localização onde a fotografia foi tirada. Por fim a imagem é codificada numa *string* de base 64, de forma a facilitar a sua transferência através da rede e através de sistemas. O fluxograma da figura 23 demonstra as ações realizadas pelo serviço *checkout* na aplicação móvel.

A estrutura da mensagem é a seguinte:

- Local – Informação essencial que permite às empresas de manutenção de ecopontos ter conhecimento do local onde devem intervir. Esta informação é obtida de diferentes formas: no caso do leitor QR encontra-se no próprio conteúdo lido e no caso da câmara é obtido através da geocodificação reversa que permite obter a morada a partir das coordenadas GPS obtidas no local da captura da fotografia.

- Condição (Código QR) – Este campo é constituído pelas respostas feitas no questionário aquando da leitura do código QR. Este campo não é necessário se a mensagem for proveniente da câmara.

- Imagem (Fotografia) – Neste campo encontra-se a fotografia codificada em base64. Este campo não é necessário se a mensagem for proveniente do leitor de código QR.
- Dados – Este campo contém os dados em que ocorreu a submissão sendo retirada diretamente do *smartphone*.

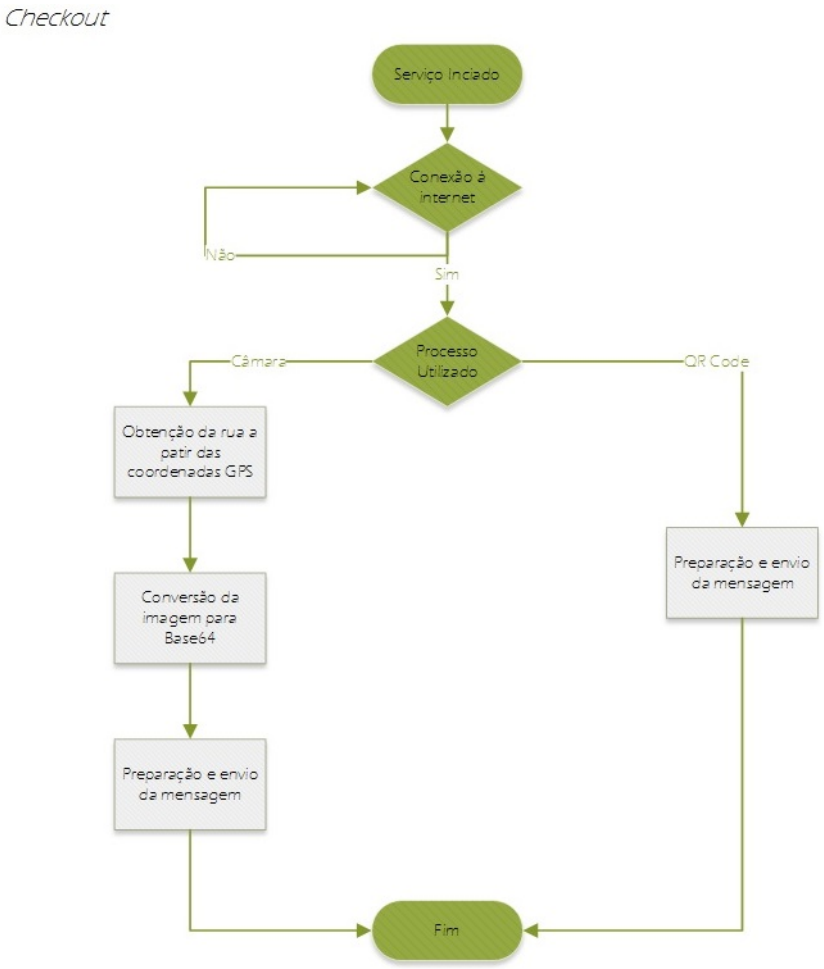


Figura 23: Fluxograma do serviço *checkout*

3.3.6 *Estrutura e Navegação*

Ao iniciar a aplicação o utilizador é apresentado com uma página onde só pode realizar duas ações: registar-se ou autenticar-se. Essa página encontra-se representada na primeira imagem da figura 24, onde se pode verificar que para poder aceder às funcionalidades da aplicação é necessário iniciar sessão com sucesso.

A segunda imagem da figura 24 apresenta a página principal da aplicação onde o utilizador se encontra com a sessão iniciada e já pode submeter a informação do estado dos ecopontos com a utilização da câmara ou do leitor de código QR.

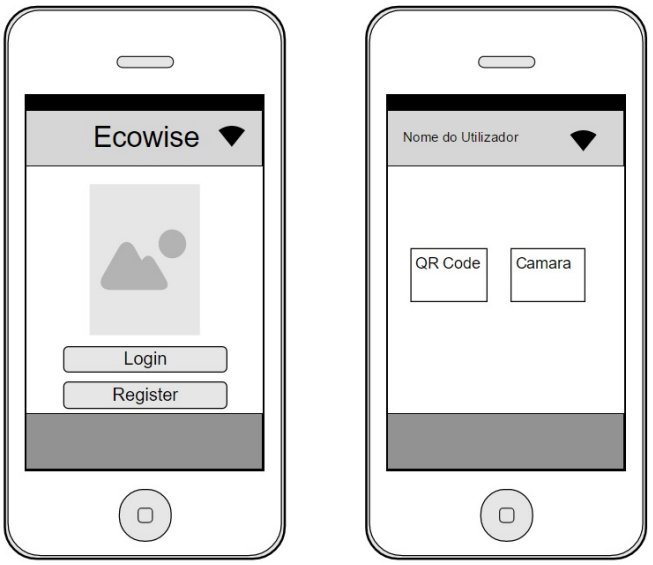


Figura 24: *Mockups* da página de autenticação e da página principal

3.4 SERVIDOR

3.4.1 *MEAN Stack*

Para o desenvolvimento desta aplicação será utilizado o MEAN *stack*. Este termo refere-se a agrupamento de tecnologias baseadas em Javascript que apresentam boa sinergia entre elas, sendo utilizadas para o desenvolvimento de sites de *internet* dinâmicos e aplicações de *internet*. A principal vantagem é que permite desenvolver todos os módulos necessários utilizando uma única linguagem e ao mesmo tempo permite de aplicações escaláveis. O MEAN *Stack* é constituído:

- MongoDB
- Express
- AngularJS
- Node.js

3.4.2 *Web Service*

Para o desenvolvimento dos serviços Web é utilizado o Node.js. Tal como o PHP o Node.js é usado principalmente no desenvolvimento de aplicações de rede, como um servidor de *internet*. No entanto, a maior diferença é que a maioria das funções em PHP bloqueiam até a sua finalização, enquanto as funções em Node.js foram desenvolvidas para serem não-bloqueadoras dessa forma, permitindo um aumento da capacidade de respostas e de crescimento em aplicações de *internet*. Para além disso existem muitas bibliotecas *open-souce*, sendo que a maior parte delas se encontra, hospedadas no site NPM, dessa forma acelerando o desenvolvimento das aplicações. Outras vantagens incluem a sua rapidez devido a utilização do motor de execução V8 do Google que compila o javascript diretamente em código máquina e a sua arquitetura orientada ao evento que torna o Node.js apropriado para aplicações em tempo real.

Para a base de dados é utilizado o MongoDB. A principal razão para esta escolha é a flexibilidade existente e o facto de gravar a informação em formato JSON que torna o processo de transferência de informação entre o cliente e o servidor mais fácil. A particularidade do MongoDB é de ser tratar de uma base de dados NoSQL. Estas bases de dados não forçam um esquema específico, permitindo a existência de diversos documentos dentro de uma coleção que apresentem estruturas e campos diferentes, permitindo assim o armazenamento de diversos tipos de dados. Em bases de dados do tipo SQL as tabelas têm de ser definidas previamente à inserção de dados e qualquer modificação que seja necessário realizar a longo prazo é consideravelmente mais difícil. Para além disso é suportado oficialmente pelo Node.js e através de ODMs, como o Mongoose, onde é possível criar e manipular a base de dados de uma maneira mais rápida e eficiente.

Do lado do servidor tem a principal função de manutenção e manuseamento da base de dados. Assim, é capaz de receber os pedidos provenientes tanto da aplicação móvel como pela aplicação Web e responder de forma apropriada.

A base de dados é constituída por três coleções: o dos utilizadores, o dos ecopontos e das situações.

A coleção dos Utilizadores contém toda a informação considerada importante tanto dos utilizadores da aplicação móvel como dos utilizadores da aplicação web. Este documento é acedido prioritariamente para o processo de autenticação do utilizador e a identificação dos emissários das situações.

A coleção dos Ecopontos contém a informação correspondente aos ecopon-

tos que estão registados no sistema. Esta coleção será acedido prioritariamente para a identificação dos ecopontos correspondentes às situações recebidas e para o controlo do seu estado. Mais especificamente a sua localização (nome da rua e coordenadas GPS), quais os ecopontos existentes nessa localização e as situações que foram enviadas correspondentes a esses mesmos ecopontos.

A coleção das Situações contém, as situações submetidas pelos utilizadores da aplicação móvel no seu correspondente ecoponto. Mais especificamente o estado em que se encontrava o ecoponto na altura da submissão, a imagem que foi captura, a data e se a situação já se encontra resolvida.

Como se trata de uma arquitetura REST irá receber pedidos HTTP com diferentes métodos (GET,POST,DELETE e PUT) e o *web service* terá de ser capaz de traduzir esses pedidos de forma a determinar qual coleção que é necessário aceder e quais são as ações que devem ser realizadas. Para isso é utilizado o express, uma *framework* utilizada em Node.js para aplicações web que possibilita a criação de uma tabela de *routing*. Cada uma das entradas da tabela de *routing*, que se encontram definidas na tabela da figura 25, permitem à aplicação determinar a resposta apropriada aos pedidos provenientes do cliente baseados no método HTTP utilizado no pedido e do seu URL.

URI	Método	Ação	Content-type
/ecopoint	GET	Listar ecopontos	JSON
/ecopoint	POST	Criar ecopontos	JSON
/ecopoint/:ecopoint_id	GET	Obter um ecoponto	JSON
/ecopoint/:ecopoint_id	POST	Criar um ecoponto	JSON
/ecopoint/:ecopoint_id	DELETE	Apagar um ecoponto	JSON
/situation	POST	Criar situação	JSON
/:ecopoint_id/:situation_id	GET	Obter uma situação	JSON
/:ecopoint_id/:situation_id	PUT	Modificar uma situação	JSON
/:ecopoint_id/:situation_id	DELETE	Apagar uma situação	JSON
/resolved /:situation_id	POST	Definir uma situação como resolvida	JSON
/users	POST	Criar utilizador	JSON
/users	GET	Listar utilizadores	JSON
/users/:user_id	GET	Obter um utilizador	JSON
/users/:user_id	PUT	Modificar um utilizador	JSON
/users/:user_id	DELETE	Apagar um utilizador	JSON
/users/authenticate	POST	Procedimento de autenticação	JSON

Figura 25: Diagrama Sequencial com representação de um pedido REST para listagem de recursos

Como podemos verificar as ações realizadas divergem de acordo com o URL recebido. Quando o URL é identificado com "/recurso"corresponde normalmente ao processo de listagem de uma coleção ou da criação de um novo objeto. A figura 26 mostra o exemplo de um pedido GET entre uma aplicação cliente e o *web Service*.

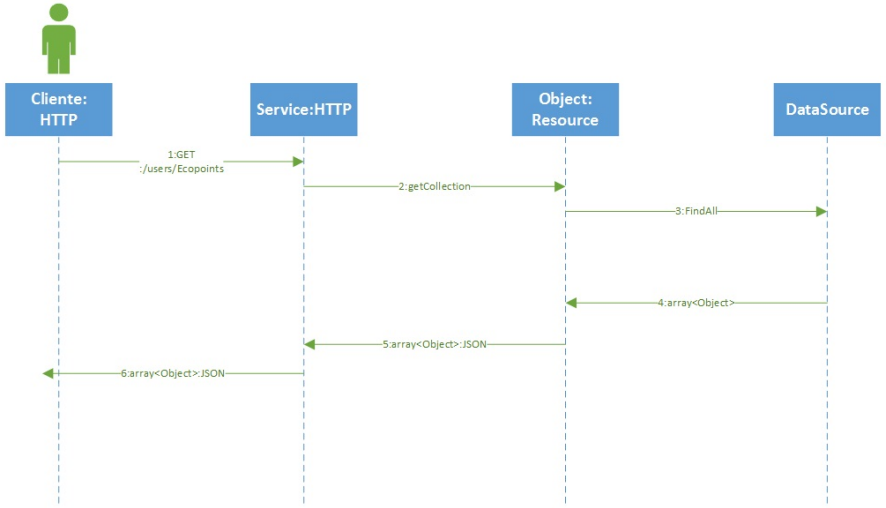


Figura 26: Tabela de ações generalizadas dos serviços Web utilizando os métodos

Caso se pretende aceder a um objeto em particular é necessário indicar a coleção em que se encontra e o seu número identificador. Nestas situações os processos a realizar são diversos já que incluem: listagem, modificação e remoção de um objeto específico. Um exemplo é o processo de autenticação que procura um objeto que contenha os dados introduzidos pelo utilizador, para que possa assim confirmar a sua presença na base de dados. Este processo encontra-se representado na figura 27.

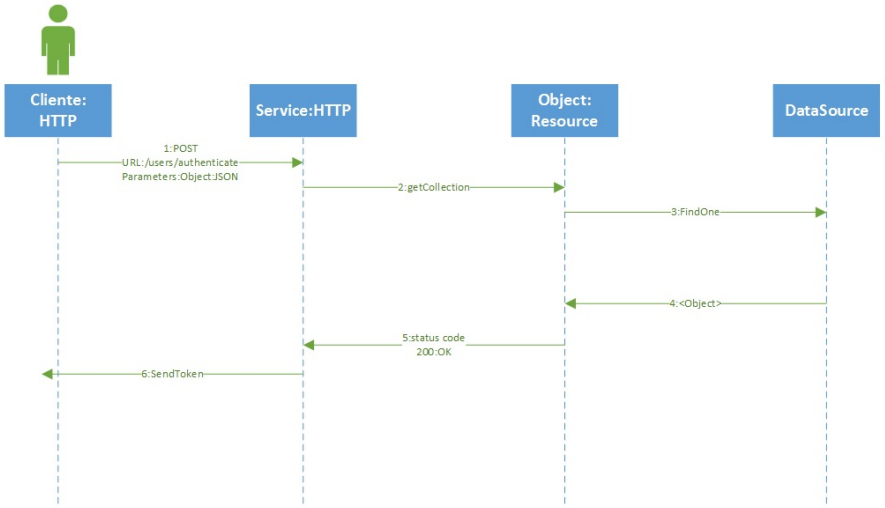


Figura 27: Diagrama Sequencial com representação de um pedido REST para a autenticação de um utilizador

3.5 *website*

Para o desenvolvimento do *website* é utilizado o Angular.js. Como foi referido anteriormente o Angular.js trata-se de uma *framework* de aplicações web baseado em JavaScript que adapta e estende o tradicional HTML para apresentar conteúdo dinâmico. As razões para esta escolha foram diversas: o facto de ser desenvolvido em JavaScript que permite uma maior sinergia e rapidez de desenvolvimento, o serviço scope que permite detetar nas secção do *model* e modificar as expressões da *view* através do *controller* e vice-versa. Este permite uma experiência mais rica e responsiva para o utilizador. Este é suportado pela Google e apresenta um bom suporte por parte da comunidade.

3.5.1 *Visão geral da Aplicação*

Como é possível verificar na figura 28, o *website* será estruturado da seguinte maneira:

- Página Principal – Contém a informação relacionada com o ao *website* e permite o registo de novos utilizadores e a entrada dos já existentes.

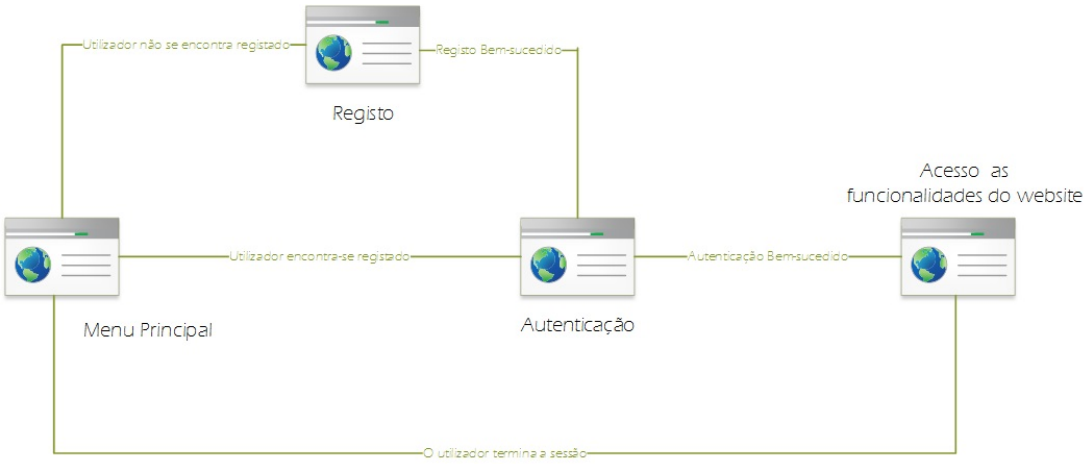


Figura 28: Visão geral do *Website*

- Autenticação – Permite a introdução dos dados relativos ao utilizador com correspondente autenticação.
- Registo – Registo de um novo utilizador de forma a permitir o acesso às funcionalidades do *website*.
- Calendário – Apresenta um calendário ao utilizador com as submissões ocorridas organizadas a partir da data da sua submissão.
- Mapa – Apresentação do mapa corresponde à área que a empresa se encontra a monitorizar com marcadores correspondentes à localização dos ecopontos.
- Relatórios - Permite o acesso aos dados estatísticos dos ecopontos que se encontram a ser monitorizados.

3.5.2 *Página Principal*

A página inicial apresentada ao utilizador, apresenta a informação sobre o sistema EcoWise com um explicação relativa as suas funcionalidades e objetivos. As funcionalidades disponíveis ao utilizador variam dependendo do seu estado de autenticação.

3.5.3 *Autenticação*

Esta página é onde o utilizador realiza o processo de autenticação. Para tal introduz os dados utilizados no seu registo, sendo que de seguida é preparada uma mensagem com esses mesmos dados para serem envidados para o servidor, de forma a confirmar a existência desse utilizador na base de dados. Caso seja bem-sucedido terá acesso às funcionalidades do *website* que se encontram limitadas a utilizadores registados. Este processo encontra-se demonstrado na figura 29.

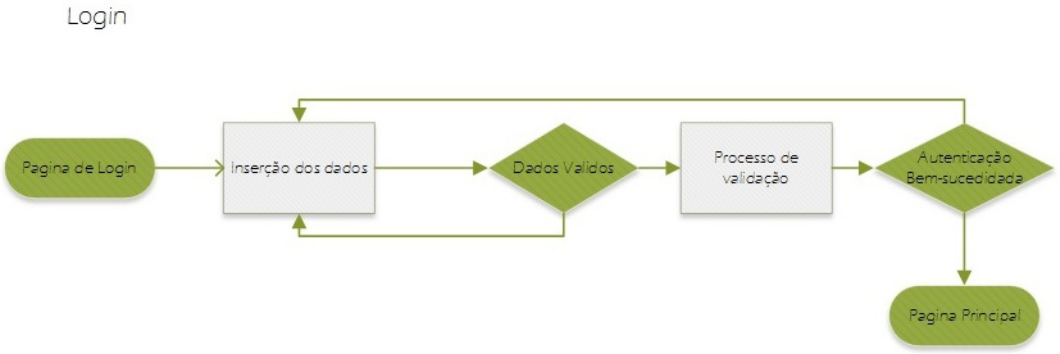


Figura 29: Fluxograma do processo de autenticação

3.5.4 *Registo*

Será aqui que o utilizador irá efetuar o seu registo. Tal como acontecia na aplicação móvel consiste na inserção dos dados que pretende utilizar para se registar com consequente processo de autenticação de forma a confirmar a possibilidade de registo.

3.5.5 *Calendário*

Apresentação de um calendário com as situações submetidas pelos utilizadores da aplicação móvel organizadas de acordo com a sua data de submissão. Dessa forma permite aos gestores dos ecopontos terem um controlo mais preciso das submissões e possibilita a utilização dessa informação para estatística. O processo pode ser visualizado na figura 30.

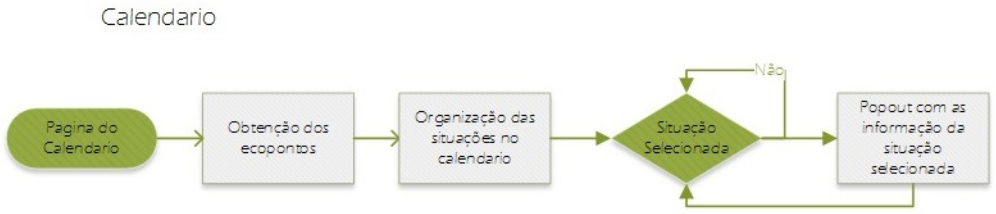


Figura 30: Fluxograma do processo da organização das situações no calendário

3.5.6 *Mapa*

Esta página apresentará um mapa contendo os ecopontos que a empresa se encontra a monitorizar, exposto através de marcadores. O fluxograma da figura 30 mostra esse processo. Ao pressionar este marcadores o utilizador tem acesso às situações que foram submetidas relativamente a esse ecoponto, podendo de seguida visualizar cada uma destas situações ao pormenor.

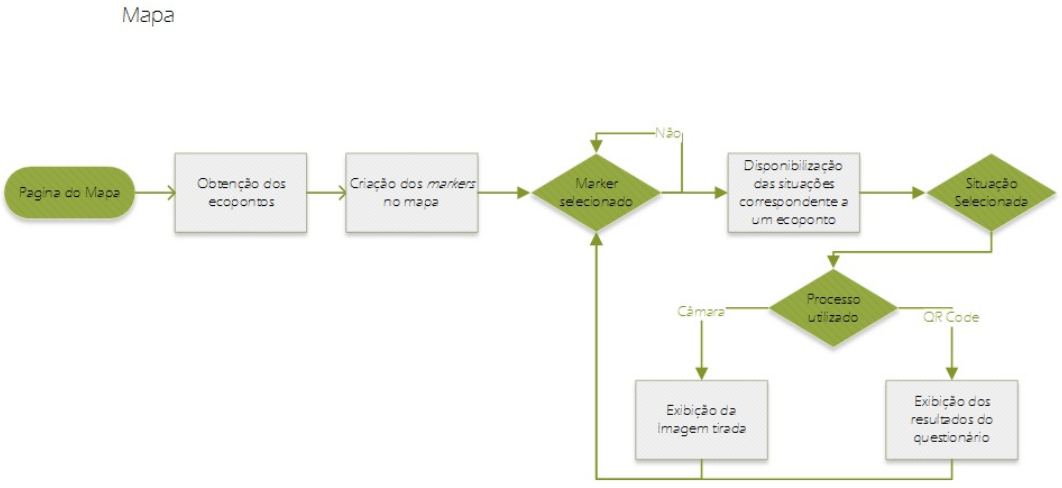


Figura 31: Fluxograma do processo de visualização dos ecopontos no mapa

3.5.7 *Relatórios*

Nesta página o utilizador pode escolher visualizar informações estatísticas, em forma de gráficos e dos dados, dos ecopontos existentes na base de dados. Os dados necessários para a criação desses mesmos gráficos são calculados quando o

utilizador seleciona a rua do ecoponto que pretende. A obtenção dos relatórios é apresentada na figura 32.

Figura 32: Fluxograma do processo de obtenção dos relatórios

3.5.8 *Estrutura e Navegação*

O *layout* do *website* é composto por três componentes: o *Header*, o *Body* e o *Footer*. O *Header* corresponde ao cabeçalho da página que contém os links para as diversas páginas a que o utilizador tem acesso. Os links apresentados dependem do facto de o utilizador se encontrar ou não autenticado no sistema. Se estiver autenticado, é apresentado com todas as funcionalidades, caso contrário apenas poderá realizar o registo e a autenticação. Na figura 33 é possível visualizar os *mockups* do *Website*. A primeira imagem corresponde à página de autenticação onde as únicas opções que lhe são possíveis são efetuar o registo ou proceder à autenticação. Na segunda imagem, o utilizador já se encontra autenticado e é possível verificar que os *links* no *header* foram alterados, disponibilizando um completo acesso às funcionalidades.

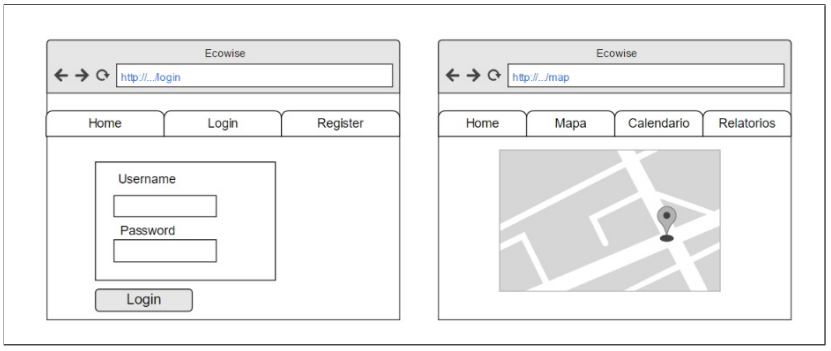


Figura 33: Mockups da estrutura das páginas de autenticação e do Mapa do *Website*

4

IMPLEMENTAÇÃO DO SISTEMA

No capítulo anterior foram apresentados os módulos do sistema de maneira a permitir compreender a arquitetura do sistema e o seu funcionamento. Foi descrita a estrutura que é utilizada para a implementação do *Website* e o desenvolvimento da aplicação Android, as funcionalidades necessárias para a sua viabilidade, a organização da base de dados e o processo de comunicação entre as aplicações clientes e o servidor.

Neste capítulo é apresentado o trabalho para a realização dos diferentes módulos, mais concretamente, o que foi possível cumprir relativamente ao planeado no inicio planeado e o que foi necessário alterar de maneira a que fosse possível cumprir com os objetivos pretendidos. Tal como no capítulo anterior é abordado em primeiro o desenvolvimento da aplicação Android, seguido pela base de dados e o código do lado do servidor, e por fim a implementação do *website*.

4.1 APLICAÇÃO ANDROID

A implementação do sistema foi iniciada com o desenvolvimento da aplicação cliente móvel. Como já foi referido, esta aplicação foi realizada em Android já que permitiu cumprir com as funcionalidades necessárias e ao mesmo tempo disponibilizar a aplicação a um grupo de pessoas mais abrangente.

4.1.1 *Página Inicial*

A primeira interação é com a página de inicial que apresenta ao utilizador duas opções: realizar a autenticação com o seus dados ou efetuar o registo do utilizador.

O processo de inicio de sessão e o do registo apresentam bastantes semelhanças, o utilizador introduz os seus dados nos respetivos campos sendo este constituídos por nome, palavra-chave, localização e email, sendo os dois últimos exclusivamente necessários para o registo. Estes dados são utilizados para a preparação de uma mensagem que e enviada num pedido de autenticação ao servidor. Para a comunicação é utilizado o *Volley*. Este trata-se de uma biblioteca HTTP com o propósito de transferir informação através da rede. A sua compatibilidade com strings, imagens e JSON fazem deste o ideal para a realização desta tarefa.

No caso da realização do início da sessão o pedido é enviado para a *route* /api/users/authenticate, que permite pesquisar na base de dados pela existência do utilizador. Aquando da resposta do servidor, e caso esta se verifique, os seus dados são guardados de forma a permitir obtê-los quando necessário, sendo de seguida reencaminhado para a página principal.

No caso do registo. o pedido é enviado para a *route* /api/users que, irá pesquisar na base de dados pela existência do utilizador, e caso não exista procede ao seu registo. Caso o registo seja possível, o utilizador é reencaminhado de volta para a página inicial para que se possa autenticar.

```
b1.setOnClickListener(new View.OnClickListener() {
          @Override
          public void onClick(View v) {

              if (!Username.getText().toString().equals("") && !Password.getText().toString().equals("")  )
              {
                  authenticateUser();
                  saveLogindata();
              }
              else{
                  MyUtils.MakeToast(Act,"Error! Password or Username are empty.");
                  }
          }
      });
```

Algoritmo 4.1: Código do processo de autenticação

4.1.2 *Página Principal*

Depois do utilizador ter iniciado a sessão com sucesso é encaminhado para esta página. Aqui o utilizador tem acesso às funcionalidades da aplicação.

Tal como pretendido o utilizador pode decidir entre as duas opções: QR Code ou Câmara. A escolha é feita através de um ecrã deslizante, desta forma com um simples movimento, é capaz de alternar entre os processos disponíveis na aplicação e decidir qual deles pretende utilizar para registar a situação. Com um simples premir de um botão este é iniciado, de forma a ser a mais interativo. Apresenta também uma barra de tarefas que exibe o nome do utilizador que se encontra em sessão e também dá acesso às opções e às ferramentas existentes na aplicação. Qualquer que seja o processo utilizado no final da submissão, o utilizador é reencaminhado para esta página.

```
        mPager = (ViewPager) findViewById(R.id.pager);
        mPagerAdapter = new ScreenSlidePagerAdapter(getSupportFragmentManager());
        mPager.setAdapter(mPagerAdapter);
    }
```

Algoritmo 4.2: Código da inicialização do ecrã deslizante

4.1.3 *QR Code*

Para a leitura dos código QR é utilizada ZXing, que se trata de uma biblioteca *open-source* de processamento de imagem e permite a leitura de códigos QR de uma forma rápida e eficiente. O processo começa por iniciar a câmara em modo de leitura, o utilizador passa a câmara por qualquer superfície que contenha o código e após algum tempo, é apresentado o resultado da leitura. De seguida é lhe mostrado um questionário constituído por *checkboxes* que permite ao utilizador informar relativamente aos ecopontos existentes e os seus estados atuais. Estas informações são guardadas para mais tarde serem utilizadas para a submissão da situação. Por fim inicia o serviço de envio de mensagem e volta para a página principal.

```
    public void onActivityResult(int requestCode, int resultCode, Intent intent) {
       if (requestCode == 0) {
           if (resultCode == RESULT_OK) {
               //get the extras that are returned from the intent
               String contents = intent.getStringExtra("SCAN_RESULT");
               String format = intent.getStringExtra("SCAN_RESULT_FORMAT");
               Toast toast = Toast.makeText(this, "Content:" + contents +
```

```
                                                    " Format:" + format, Toast.LENGTH_LONG);
            toast.show();
        }
    }
}
```

Algoritmo 4.3: Código da obtenção dos resultados da leitura do código QR

4.1.4 *Câmara*

A captura de imagens é realizada utilizando a própria biblioteca Android para o efeito. A câmara é inicializada e a partir daí basta ao utilizador premir o botão de captura. Após guardar a imagem, a aplicação obtém as informações GPS existentes na altura em que a foto foi tirada. Caso não seja possível é apresentado um aviso sobre a impossibilidade de obtenção da localização geográfica e apresentada ao utilizador duas opções: voltar á página principal e tentar outra vez ou mudar para o leitor de QR code. Caso os dados GPS sejam obtidos com sucesso o serviço Checkout é iniciado e o utilizador reencaminhado para a página principal.

```
public void showCurrentLocation() {


    Geocoder geocoder= new Geocoder(this, Locale.ENGLISH);
    Location location=null;
    int permissionCheck = ContextCompat.checkSelfPermission(this,
            Manifest.permission.ACCESS_FINE_LOCATION);

    if(permissionCheck==0){
        location = manager.getLastKnownLocation(LocationManager.GPS_PROVIDER);
        }
    if ( location!= null) {

        String message = String.format(" Location \n Longitude: %1$s \n Latitude: %2$s",
                location.getLongitude(), location.getLatitude());
                  Toast.makeText(this, message, Toast.LENGTH_LONG).show();
```

Algoritmo 4.4: Código da obtenção das coordenadas GPS

4.1.5 *Serviço Checkout*

Este serviço é iniciado independentemente da escolha do processo por parte do utilizador. É responsável pela organização da informação recolhida e preparação da submissão da mensagem. Ao iniciar o serviço, é necessário saber qual o processo que o invocou de forma a ser possível determinar o que é necessário fazer. De seguida é iniciado o *timer*. Para ser possível enviar informação para o

servidor é necessário conexão à *internet*, sendo que a função do *timer* é verificar se a aplicação contém acesso a esta. Por essa razão, é que se utiliza o serviço já que permite ao utilizador continuar a realizar as suas tarefas enquanto a aplicação se encontra a espera da conexão. Caso tal se verifique, o *timer* é desligado e as operações correspondentes aos processos começam a realizar-se. Caso o serviço tenha sido iniciado pelo processo de leitura do código QR, é fundamental recolher a informação obtida e proceder ao registo da situação. A informação necessária é o resultado do questionário que o utilizador submeteu e a localização do ecoponto que deverá, corresponder ao texto obtida aquando da leitura do código QR. Para o registo, é obtida a data diretamente do sistema e juntamente com a restante informação é enviada para o servidor.

Caso o serviço tenha sido iniciado pela câmara, é necessário obter a informação relativamente a imagem e proceder ao registo da situação. Neste caso é obtida a latitude e a longitude em que o utilizador se encontrava e a imagem obtida é convertida em base64, de forma a tornar o seu armazenamento mais fácil e conveniente na base de dados. Utilizando as coordenadas. é iniciado o processo de geo codificação reversa que permite a obtenção da rua. Por fim é retirada a data e juntamente com os restantes dados é criada uma mensagem que será enviada ao servidor.

Em ambos os processos a base de dados é revista a procura da rua existente na mensagem submetida e caso seja bem-sucedido esta será armazenada no local apropriado sendo a aplicação informada do sucedido. A quando da receção da resposta proveniente do servidor, o utilizador é notificado do resultado da submissão.

```
private Runnable runnable = new Runnable() {
       @Override
       public void run() {

           if(started && CheckConnection()) {

               if(appName.equals("QRC"))
               {
                   getQRCData();
                   registerUserQRC();
               }

               if(appName.equals("GPS"))
               {
                   getImageData();
                   registerUserImage();
               }

               started = false;
               handler.removeCallbacks(runnable);
               stopSelf(serviceID);
           }
           else handler.postDelayed(runnable, 2000);
```

```
    }
};
```

Algoritmo 4.5: Código que determina quais as funções que são necessárias realizar de acordo com o processo utilizado

4.2 WEB SERVICE

Como já foi referido, o código do lado do servidor foi desenvolvido em Node.js utilizando o Jetbrains WebStorm como IDE. Este código tem como objetivo tratar do lado do servidor correspondente da aplicação Web. A sua principal tarefa é receber os pedidos provenientes da aplicação Android e do lado cliente da aplicação web e determinar os processos que necessitam de ser efetuados e responder devidamente. Como está a ser utilizada uma arquitetura RESTful é necessário não só analisar o URL utilizado, como o método HTTP utilizado. Inicialmente foi necessário determinar os *schemas* que serão utilizados para guardar a informação considerada necessária na base de dados. Foi preciso definir quais as *keys* que o constituem e também qual o seu tipo. No caso do sistema EcoWise foram construídas os três *schemas* seguintes:

- Ecopontos - Contém as informações relativas aos ecopontos que se encontram a ser monitorizados. Inclui a sua localização (GPS e nome da rua) de forma a ser possível identificar de forma mais rápida, quais os ecopontos existentes nesse local e a situações que foram submetidas sobre esse mesmo ecoponto.
- Utilizadores – Contém os dados dos utilizadores, da aplicação web e Android e o seus dados pessoais como por exemplo: nome, e-mail e password. Os utilizadores da aplicação devem ser separados já que os únicos que devem ter acesso ás funcionalidades do *website* são os trabalhadores da empresas de gestão dos ecopontos , daí a existência de um campo denominado de *role*. Este campo dependerá se o utilizador se registou no *website* ou pela aplicação móvel dessa forma permitindo a distinção entre os dois grupos.
- Situações - Este *schema* disponibiliza as informações consideradas relevantes sobre uma determinada situação, como por exemplo: o seu estado, a data de submissão, a fotografia correspondente e o seu estado de resolução. Encontra-se embebido no *schema* dos utilizadores já que um ecoponto contará com diversas submissões independentemente de estarem resolvidas ou não.

Com o *schemas* definidos, é necessário preparar a base de dados. Utilizando o MongoDB é criada e inicializada para que esteja pronto a armazenar. Para facilitar o seu manuseamento, é utilizado uma biblioteca Nodejs chamada de mongosse. Para o desenvolvimento do código também foi utilizado o Express, que como já foi referido, trata-se de uma *framework* para aplicações web que permite preparar mediadores de forma a responder aos pedidos HTTP. Com o Express também se define as tabelas de *routing* que são utilizadas para determinar as diferentes ações a executar, dependendo dos métodos desses pedidos e do URL.

A routes foram que foram definidas são as seguintes:

- /users – Processos em que é necessário o acesso à informação relativa aos utilizadores das aplicações. Mais concretamente a inserção, remoção, modificação dos dados relativos aos utilizadores e autenticação das suas credenciais.
- /ecopoint - Processos em que é necessário acesso aos dados referentes aos ecopontos que a empresa se encontra a monitorizar e a sua manipulação.
- /situation – Aos processos que correspondem ao manuseamento das situações relativas à um determinado ecoponto, a *route* utilizada é esta. Essas ações incluem a remoção ou inserção, a recolha das situações de um determinado ecoponto ou de todos e também a classificação delas como resolvidas.

Figura 34: Coleções existente na base de dados

4.3 *website*

O *website* foi desenvolvido usando Angular.js utilizando Jetbrains WebStorm como IDE. O *website* tem como principal objetivo fornecer uma interface ao utilizador que permita um acesso às funcionalidades existentes no sistema de forma mais intuitiva possível. O *website* tem de ser capaz de fazer pedidos de informação à base de dados a quando considerar necessário, fazer o tratamento da informação existente e disponibilizar ao utilizador de forma acessível e conveniente.

Inicialmente foi necessário definir a estrutura que o *website* iria tomar. Com isso já definido, foi necessário desenvolver as páginas apropriadamente ditas. Como já foi referido, o *website* é constituído pelas seguintes páginas:

- Página Principal
- Autenticação
- Registo
- Calendário
- Mapa
- Data

Para além disso foi, necessário determinar o método de autenticação que permitisse o acesso único aos utilizadores da empresa às funcionalidades do *website* e às informações relativas aos ecopontos.

Com estes aspetos já definidos, o próximo passo é o desenvolvimento das páginas individuais e implementação das funcionalidades necessárias. De seguida é demonstrado o desenvolvimento do *website*.

4.3.1 *Página Inicial*

O desenvolvimento do *website* começou pela criação da página introdutória que contém a informação relativa ao sistema EcoWise e as funcionalidades. Antes da autenticação, só é apresentado ao utilizador duas opções: iniciar sessão ou registar-se. O propósito é limitar o acesso a funcionalidades do sistema de forma a prevenir o acesso não-autorizado a utilizadores que não tem permissão para o

fazer. As funcionalidades disponíveis aos utilizadores são demonstradas na barra de navegação. Esta barra encontra-se presente em todas as páginas e muda dependo da existência de uma sessão iniciada por um utilizador ou não.

Os processos de início de sessão e registo foram os primeiros a serem trabalhados. Em ambas as páginas o utilizador insere os dados em que se pretende registar ou iniciar sessão, sendo esses dados transferidos para o servidor para se proceder a autenticação. No processo de autenticação, os dados são recolhidos e é criada uma estrutura com essa informação, sendo depois enviada para servidor de forma a prosseguir com o processo.

Quando o início de sessão é bem-sucedido é retornado um *token* que é necessário para qualquer tipo de acesso e manipulação dos dados correspondentes aos ecopontos. Dessa forma, é possível restringir o acesso a esta informação unicamente a utilizadores registados. Este *token* encontra-se armazenado durante o decorrer da sessão e está sempre disponível para ser integrado em qualquer pedido em que seja necessário. Para além do *token* a barra de navegação é modificada de forma a dar acesso ao utilizador às páginas que anteriormente estavam restringidas.

```
<div class="collapse navbar-collapse" id="myNavbar">
    <ul  ng-if="isAuthenticated()" class="nav navbar-nav">
         <li class="active"><a href="#">Home</a></li>

        <li><a href="#/ecopointLocation">Mapa</a></li>
        <li><a href="#/ecopointGraphs">Relatorios</a></li>
        <li><a href="#/ecopointCalendar">Calendario</a></li>
    </ul>
    <ul ng-if="!isAuthenticated()" class="nav navbar-nav  navbar-right style=">
        <li><a  href="#/ecopointSign"><span class="glyphicon glyphicon-user">
                                </span> Sign Up</a></li>
       <li><a href="#/ecopointLogin"><span class="glyphicon glyphicon-log-in">
                                </span> Login</a></li>
       </ul>
    <ul ng-if="isAuthenticated()" class="nav navbar-nav  navbar-right style=">
       <li><a href="#/ecopointData"><span class="glyphicon glyphicon-user">
                                </span>{{nameUser}}</a></li>
         <li><a  ng-click="logOut()">Logout</a></li>
       </ul>
   </div>
```

Algoritmo 4.6: Código da barra de navegação da página inicial do *website*

4.3.2 *Mapa*

De seguida, a página desenvolvida foi a do Mapa. Esta página é o núcleo do *website* já que é esta página que permite ao utilizador monitorizar os ecopontos e as situações correspondentes a cada uma deles. Um das preocupações na imple-

mentação do mapa foi tornar o processo de monitorização dos ecopontos o menos complicado possível. Para tal foi utilizado a API do *Google Maps* que permite a visualização do mapa da zona que se encontra a vigiar e com comandos simples que permitem a sua manipulação. A sua configuração consiste em definir a localização predefinida em que o mapa estaria ao carregar a página e as suas características. Os ecopontos são acessíveis através dos símbolos no mapa denominados de *markers* existentes na localização dos ecopontos. A partir desses *markers* que é possível ter acesso às situações correspondentes e as suas propriedades. Ao carregar a página a primeira coisa a ser feita é adicionar os *markers* ao mapa, para tal é feito um pedido (juntamente com o *token*) para obter os ecopontos presentes na base de dados e as suas informações. Essas incluem a latitude e longitude dos ecopontos, permitindo assim um posicionamento no mapa de forma mais correto possível, e com um identificador única para esse ponto.

Com os pontos já a serem visíveis a partir do mapa, o foco passou a ser como obter as situações e disponibilizar as mesmas aos para os utilizadores. Ao clicar em qualquer um dos *markers* é feito um pedido ao servidor para obter as situações relacionadas com esse ecoponto. A disponibilização dos dados, depende do método de submissão da situação aplicado pelo utilizador da aplicação móvel. Caso tenha decidido utilizar a câmara é necessário descodificar a string *Base64* de volta para uma imagem para poder ser visível no *website*. Caso contrário, se tiver decidido utilizar o código QR é necessário apresentar as respostas realizadas ao questionário. Ambas apresentam um botão de resolução de forma a distinguir as situações que já foram resolvidas e quais as que ainda estão por resolver. Qualquer mudança à situação realizada é atualizada na base de dados.

```
<div id="two">
      <p ng-if="showSituations==true && errorOcorred==true">

         {{chosenSituation.status=null}}
         {{chosenSituation.image=null}}
         {{chosenSituation.date=null}}
      </p>
       <p ng-if="chosenSituation.status=='Camera'">
             <panzoom  config="config" model="model" align="center" style="width:500px; height: 400px">
          <img   ng-class="{rotateImg: true}" ng-src="{{getImage(chosenSituation._id)}}"
                                              border="1" width="400px;" height="300px;"/>
          </panzoom>
       </p>
       <p ng-if="chosenSituation.image=='QR CODE'" >
          {{chosenSituation.status}} {{chosenSituation.image}} {{chosenSituation.date}}
       </p>
                  <p ng-if="chosenSituation.image=='QR CODE' ||chosenSituation.status=='Camera' " >
                     <button ng-click="ClearSituation(chosenSituation._id)">Resolvido</button>
        </div>
                </section>
```

Algoritmo 4.7: Código da disponibilização dos dados ao utilizador

4.3.3 *Calendário*

Um dos problemas que o mapa apresenta, é o facto de a certo ponto o número de situações de um determinado ecoponto ser bastante elevado o que tornaria todo o processo mais complicado e pouco acessível. Para resolver este problema e ao mesmo tempo providenciar um registo temporal das situações, decidiu-se desenvolver uma página que contém um calendário onde estariam presentes as situações submetidas num determinado mês. Inicialmente as situações são retiradas e são criados eventos a partir das informações existentes. Neste ponto o calendário deverá conter as situações do mês que o utilizador se encontra a visualizar. Tal como no mapa, também é possível analisar uma situação específica com mais detalhe, para isso é pressionado o evento que se pretende visualizar e é criado um *popout* com as informações relativas a essa situação, como por exemplo a localização, o estado e a sua imagem.

```
$scope.setEvent = function () {
       console.log($scope.Calendar);
       for (i = 0; i <  $scope.Calendar.length; i++) {

           if($scope.Calendar[i].date!=null)
           {
               var event=new Object();
               event.id=$scope.Calendar[i].id;
               day=$scope.Calendar[i].date.substring(8,10);
               month=$scope.Calendar[i].date.substring(5,7);
               year= $scope.Calendar[i].date.substring(0,4);
               event.title=$scope.Calendar[i].location;
               event.start=new Date(year,month-1,day);
               \$scope.events.push(event);
           }
       }
       $scope.showEvents=true;
   };
```

Algoritmo 4.8: Calendário com eventos

4.3.4 *Relatórios*

Um dos objetivos do sistema EcoWise, é a obtenção de dados estatísticos relativamente aos ecopontos e ao processo de recolha destes. Com isso em mente decidiu-se desenvolver uma página que permitisse ao utilizador aceder aos dados estatísticos sobre um ecoponto específico. O utilizador ao escolher um determinado ecoponto são criados gráficos utilizando os dados existentes na base de dados.

Para tal são recolhidos os dados referentes a cada um dos ecopontos e as variáveis calculadas de forma a ser utilizadas para a geração dos gráficos.

```
function addEco(result) {
                id=0;
                for (i = 0; i < result.length; i++) {
                        var c=new Object();
                        num_sit=0;
                        suc_sit=0;
                        unr_sit=0;

                        for (x = 0; x < result[i].situation.length; x++) {
                                num_sit++;
                                if(result[i].situation[x].resolved=="False")
                                {
                                        unr_sit++;
                                }
                                else  suc_sit++;
                                console.log(unr_sit);
                                c.data=result[i].situation[x].date;
                        }
                        id++;
                        c.nome=result[i].location;
                        c.num_sit=num_sit;
                        c.suc_sit=suc_sit;
                        c.unr_sit=unr_sit;
                        c.id=id;

                        $scope.ecopontos.push(c);

                }

                console.log($scope.ecopontos);
        }
    };
```

Algoritmo 4.9: Código de obtenção dos ecopontos e dos dados necessários para a formação dos gráficos

5

RESULTADOS

Neste capítulo são descritos os resultados obtidos após a implementação dos módulos que foram descritos no capítulo anterior, e da sua integração.

Inicialmente são necessários testes em cada um dos módulos. É preciso verificar se o seu funcionamento e comportamento se encontra de acordo como pretendido. Caso sejam bem-sucedidos, significa que os módulos trabalham da maneira pretendida e que a integração é possível. De seguida, a viabilidade do sistema é analisada de forma a conferir se apresenta as ferramentas e funcionalidades necessárias de forma a que seja capaz de cumprir os objetivos propostos.

Após a realização destes testes, chega a altura de examinar o seu desempenho em situação reais. Os objetivos do sistema EcoWise são de apresentar a capacidade, de em qualquer lugar, o utilizador comum submeter uma situação relativa a um ecoponto através de uma aplicação móvel, para que essa mesma situação seja enviada e armazenada numa base de dados de forma a que o *website* seja capaz de processar essa informação e apresenta-la no mapa de controlo dos ecopontos. Por estas razões, é que existe a necessidade de realizar testes no terreno de forma a validar a integridade do sistema em situações do dia-a-dia.

5.1 TESTE DE INTEGRAÇÃO

Antes de começar a testar o sistema em situações reais é necessário verificar e analisar o comportamento de cada unidade existente nos módulos que constituem o sistema EcoWise. Como já foi referido o sistema é constituído por três módulos: a aplicação Android, o *webservice* e o *website*. De seguida serão descritos os testes realizados em cada um dos módulos, o processo utilizado e o motivo da sua realização.

5.1.1 *Aplicação Android*

Como já foi referido anteriormente o IDE utilizado para o desenvolvimento da aplicação Android foi o Android Studio. Este, apresenta a capacidade de criar uma máquina virtual onde é possível testar as funcionalidades da aplicação. No entanto o seu elevado consumo de memoria, que causa uma maior lentidão na execução da aplicação, funcionalidades e portabilidade limitada torna-o inadequado para testar as suas funcionalidades.

Por estas razões é que se decidiu-se realizar os testes às unidades da aplicação num *smartphone*. O tempo de implementação e execução da aplicação é muito mais rápido, permite testar as funcionalidades em diversas situações mais próximas das pretendidas e também permite analisar o desempenho da aplicação em tempo real através do *debug* do IDE.

As unidades que se pretendem testar são as seguintes:

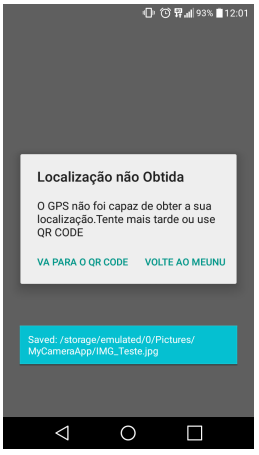
- Câmara
- GPS
- QR Code
- Comunicação com o servidor

5.1.1.1 *Câmara*

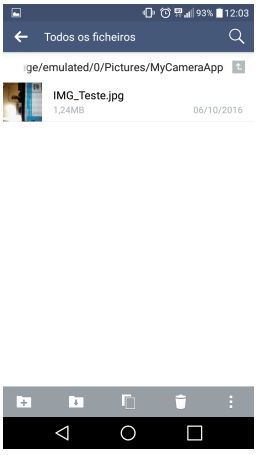
A câmara é utilizada para a recolha de fotografias dos ecopontos de forma a servir de validação da situação correspondente a determinado ecoponto sendo que por isso o seu bom funcionamento é essencial. Quando iniciado o processo a câmara é apresentada num *frame* onde o utilizador a direciona para onde pretende realizar a captura da imagem.

Teste: Realizar uma captura de uma fotografia e armazená-la numa localização que permita o seu acesso quando pretendido.

Resultado: A captura de imagem foi bem-sucedida, no caso da primeira captura realizada pela aplicação, sendo criada uma diretoria onde a imagem é guardada. A imagem foi armazenada nessa diretoria em formato JPEG. Este teste foi realizado várias vezes com sucesso e em diferentes ambientes.



(a) Leitor de código QR



(b) Resultado da descodificação de código

Figura 35: Resultados do testes realizados na câmara

5.1.1.2 *GPS*

O GPS é utilizado juntamente com a captura de imagem sendo necessário para obter a localização e dessa forma conectar a foto capturada com um ecoponto existente na base de dados. Aquando da captura da imagem é feita uma recolha dos dados da localização (latitude e longitude) que o utilizador se encontrava durante esse processo. Pretende-se que essas coordenadas sejam as mais precisas possíveis de forma a facilitar a obtenção da rua e evitar erros de submissão.

Teste: Obtenção das coordenadas GPS o mais preciso possível após a captura da imagem.

Resultado: De forma geral o teste foi bem-sucedido. A precisão das coordenadas vai depender da qualidade do recetor GPS. Foram realizados vários testes com diversos *smartphones* em que se verifica um pequena diferenças entre eles. No entanto em todos eles foi possível obter o dados de forma a poder determinar de forma confiável. Na figura 37 é possível verificar um exemplo dos testes realizados em que é possível verificar a precisão das coordenadas obtidas.

Casas decimais	Graus decimais	Escala Quantitativa
0	1.0	País ou grande região
1	0.1	Distrito ou grande cidade
2	0.01	Cidade ou Vila
3	0.001	Bairro ou Rua
4	0.0001	Rua Individual
5	0.00001	Árvores individuais
6	0.000001	Humanos individuais
7	0.0000001	Limitado praticado.
8	0.00000001	Avaliação especializada

Figura 36: Precisão das casas decimais

Na tabela da figura 36 é possível verificar que a precisão das coordenadas GPS obtidas é dependente das casas decimais que contém. Dessa forma é possível concluir que os resultados dos testes são satisfatórios, e que permitem uma boa identificação do local em que a captura de imagem ocorreu e assim facilitando a identificação do ecoponto a que a situação submetida corresponde.

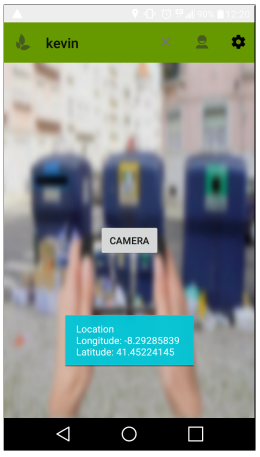


Figura 37: Dados GPS obtidos

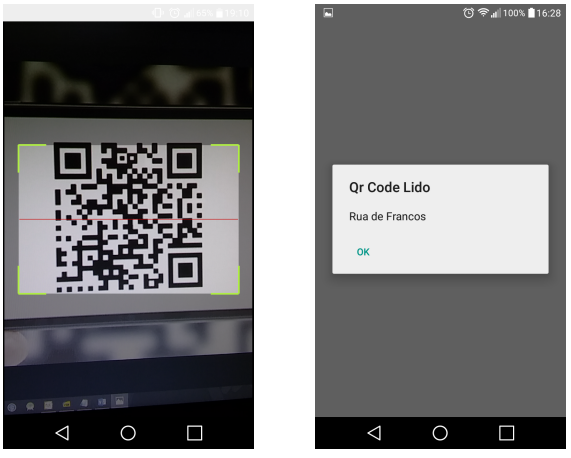
5.1.1.3 *QR Code*

A leitura do código QR é o segundo processo utilizado para a submissão de situações. O objetivo deste método é a descodificação de um código de barras bidimensional utilizando a câmara do dispositivo que o utilizador possui. Caso a descodificação seja bem sucedida então o utilizador deverá ser apresentado com um pequeno questionário onde ele indica as condições do ecoponto que o utilizador pretende submeter. Os dados que se encontram encriptados no código QR correspondem à rua em que o ecoponto se encontra e são utilizados de forma a corresponder às respostas do questionário de um ecoponto existente na base de dados.

Teste: Leitura do código QR e com posterior preenchimento de um questionário relativo ao estado dos ecopontos.

Resultado: Os testes foram executados com sucesso. A aplicação foi capaz de rapidamente e eficazmente descodificar os diversos código QR a que foi apresentado.

Nas figura 38 estão representados alguns dos testes efetuados. Como é possível verificar os resultados da leitura são apresentados após o direcionamento da câmara para a localização do código QR. Depois como previsto apareceu o questionário para preenchimento.



(a) Leitor de código QR

(b) Resultado da descodificação de código

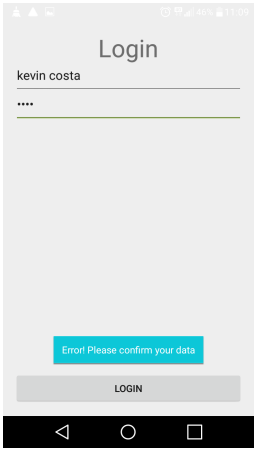
Figura 38: Resultados do testes realizados do leitor de código QR

5.1.1.4 *Comunicação com o servidor*

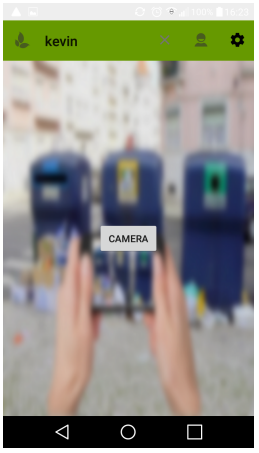
Uma comunicação eficaz com o servidor é um aspeto essencial para o bom funcionamento do sistema. Múltiplos processos necessitam de interagir com o servidor de forma a efetuarem as suas funções eficazmente. Estes processos incluem a autenticação e registo dos utilizadores da aplicação e a submissão de novas situações para a base de dados. Para tal é utilizado o *volley* que se trata de uma biblioteca HTTP que apresenta suporte para JSON e permite a requisição de informação ao servidor.

Teste: Serão realizados testes tanto ao processo de registo/autenticação como a submissão de situações provenientes dos dois processos de recolha.

Resultado: Na figura 39 mostram alguns dos testes que foram realizados. No primeiro conjunto são apresentados os resultados relativos ao processo de registo/autenticação. O utilizador foi capaz de se registar e de seguida autentificar com sucesso de forma a conseguir o acesso as funcionalidades da aplicação.



(a) Erro durante a autenticação



(b) Autenticação bem sucedida

Figura 39: Resultados do testes a comunicação ao servidor

A figura 40 representa um dos testes referentes a submissão de situações para a base de dados. Ambos os processos foram utilizados de forma a testar o maior número de condicionastes possíveis. Após a criação da mensagem é certificado que o dispositivo tenha conexão a *internet*. Caso não tenha a mensagem é preparada e é iniciado um *timer* que terá como função a análise do estado de conexão do dispositivo de forma a proceder ao envio aquando disponível.

Figura 40: Resultado da transmissão

5.1.2 *Web Service*

O código do lado do servidor foi desenvolvido em Node.js recorrendo ao Jetbrains Webstorm como IDE. Os testes foram realizados com recurso ao Postman, que se trata de um cliente HTTP, que permite a realização de testes à arquitetura REST. Para tal esta ferramenta utiliza pedidos HTTP customizados de forma a simular os que são efetuados pela aplicação móvel e pelo *website*, permitindo assim testar as diferentes condições que o sistema está sujeito.

Tirando partido das suas funcionalidades é possível testar cada uma da *routes* definidas e verificar se a receção e descodificação dos pedidos recebidos é bem sucedida e consequentemente analisar se os procedimentos realizados são os esperados para o pedido efetuado.

Teste: Utilizando o Postman serão efetuados pedidos iguais aos que o servidor deve estar preparado a receber. Dessa maneira é possível confirmar se a *routing table* foi bem desenvolvida e a descodificação corre como planeado.

Resultado: A figura 41 apresenta um dos testes que foram realizados. Os testes foram efetuados nas diferentes *routes* existentes (ecopontos, situações e utilizadores) e aos diferentes métodos que serão utilizadas para cada uma delas (GET,POST,PUT,DELETE).

Nas imagens verificamos os testes realizados onde é possível concluir que os resultados são os esperados. Foi possível obter as informação relativa a cada uma das coleções, inserir novas entradas e modificar/apagar as já existentes.

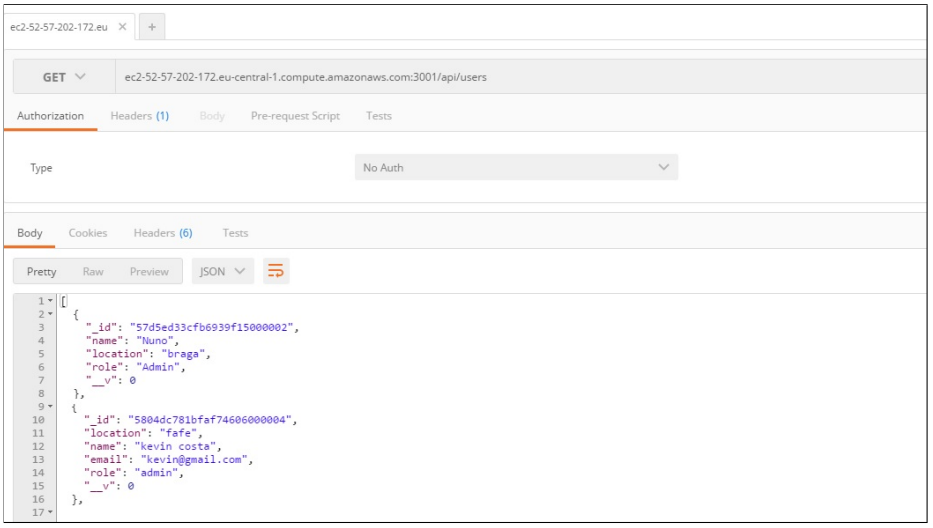


Figura 41: Listagem dos utilizadores registados

5.1.3 *Website*

O código para o *website* foi desenvolvido em Angular.js recorrendo ao Jetbrains Webstorm como IDE. O IDE apresenta a capacidade de monitorizar e validar o código diretamente num navegador de *internet* dessa forma aumentando a viabilidade dos teste realizados. De seguida são apresentadas as componentes que foram testadas:

- Registo/Autenticação
- Mapa
- Calendário
- Relatórios

Como nesta altura ainda a base de dados ainda se encontrava bastante limitada em termos de conteúdo, foi utilizado o Postman para introduzir ecopontos e situações de forma a simular as condições normais de utilização do *Website*.

5.1.3.1 *Registo/Autenticação*

O *website* deve ser utilizado unicamente pelos trabalhadores das empresas de gestão de ecopontos para controlar a sua recolha mais eficazmente. Para garantir

isso foi criada uma componente de Registo/Autenticação de forma a garantir o acesso exclusivo às funcionalidades do *website* pelas pessoas apropriadas.

Teste: Consistira em replicar o processo comum de um utilizador a efetuar o registo ao *website*. Com o registo bem-sucedido é realizado de seguida autenticação de forma a ter acesso ao *website*.

Resultado: Durante o processo de registo foi necessário indicar os dados que que se pretendem que estejam associado á nossa conta sendo esse dados de seguida utilizados para o processo de autenticação. Foram inseridas irregularidades propositadas de forma a verificar a resposta providenciada. Verificou-se que em caso de alguma irregularidade o utilizador foi consequentemente avisado do erro ocorrido.

Na figura 42 verificou-se como o processo decorreu. Inicialmente é feito o registo onde ocorre uma comparação dos dados fornecidos pelo utilizador de forma a evitar dados duplicados. De seguida foi feita a autenticação onde é confirmada a existência do utilizador na base de dados. Com esse processo concluído com sucesso o utilizador recebe um *token*. Com ele pode ter acesso a informação relativa aos ecopontos e as situações. O *token* é mantido enquanto o utilizador manter a sessão sendo removido quando a sessão termina.

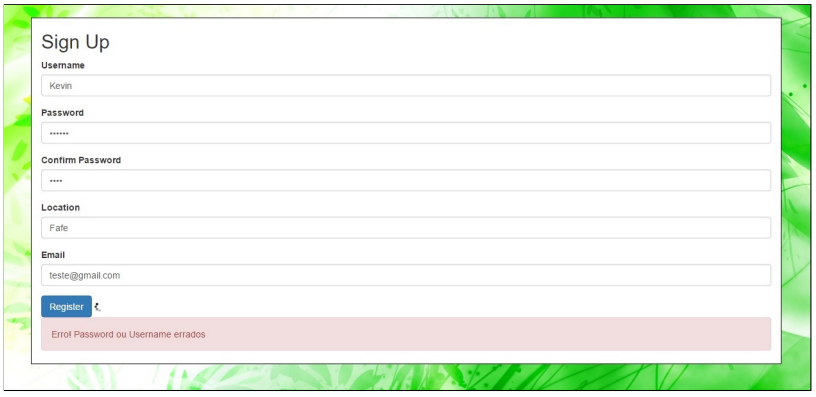


Figura 42: Erro durante o processo de registo

5.1.3.2 *Mapa*

O Mapa é uma das ferramentas essenciais para o funcionamento do *website*. É a partir daqui que é possível monitorizar os ecopontos da região e visualizar as situações existentes de cada um deles. Foram inseridos na base de dados ecopontos

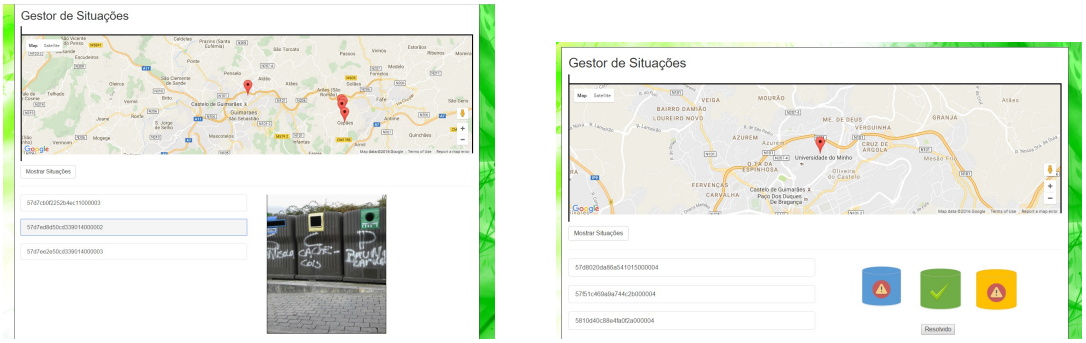
e situações para razões de teste de forma a simular o mais próximo possível a pretendia.

O *website* deve ser utilizado unicamente pelos trabalhadores das empresas de gestão de ecopontos para controlar a sua recolha mais eficazmente. Para garantir isso foi criada uma componente de Registo/Autenticação de forma a garantir o acesso exclusivo às funcionalidades do *website* pelas pessoas apropriadas.

Teste: Consistira em replicar o processo comum de um utilizador a efetuar o registo ao *website*. Com o registo bem-sucedido é realizado de seguida autenticação de forma a ter acesso ao *website*.

Resultado: Ao carregar o mapa é possível verificar os marcadores espalhados. Estes representam os ecopontos e como é possível verificar pelas imagens, ao pressionar um deles são apresentados as situações recebidas e não resolvidas desse ecoponto. As situações são obtidas diretamente da base de dados sendo por isso necessário o *token* obtido durante a autenticação de forma a pode-los recolhê-las.

Na figura 43 é possível verificar os dois cenários que podem ocorrer. Caso o processo de submissão tenha sido a câmara é exibida a imagem captura pela aplicação, se o escolhido foi o leitor de código QR então é exibido o resultado do questionário realizado pelo utilizador aquando da descodificação da matriz.



(a) Câmara (b) Leitor de código QR

Figura 43: Resultados do testes realizados ao mapa

5.1.3.3 *Calendário*

O calendário tem como objetivo de ser um arquivo da situações já resolvidas e dessa forma permitir ao utilizador analisar as ocorrências submetida durante um específico mês. É possível aceder, tal como no mapa, aos detalhes relativos

a situação incluindo a imagem capturada e o estado do ecoponto submetido pelo utilizador.

Teste: Aceder ao calendário e visualizar às situações ocorridas durante um mês especifico.

Resultado: Ao abrir a página é possível verificar as situações resolvidas organizadas de acordo com a data de submissão. Estas são apresentadas como anotação definidas pela rua em que se encontra o ecoponto. Ao clicar aparece um *popout* com as informações relativas as submissão e ao ecoponto na altura da submissão. A figura 44 demonstra um exemplo do que o calendário pode apresentar.

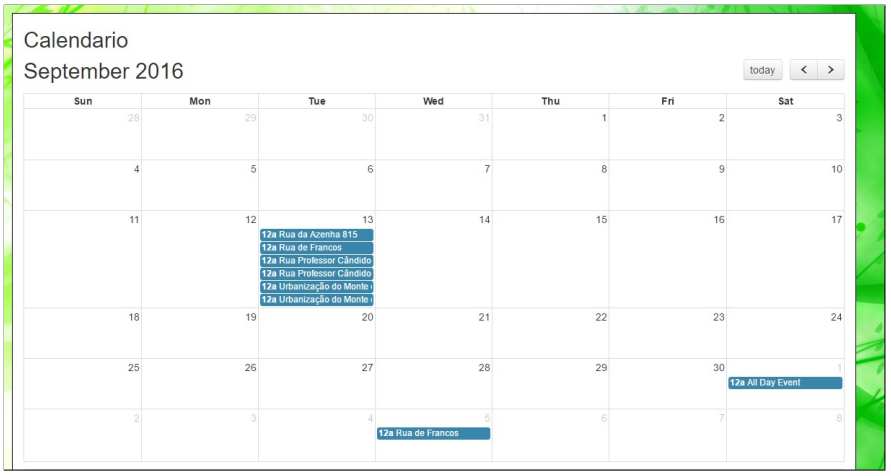


Figura 44: Calendário com as situações submetidas organizadas por data

5.1.3.4 *Relatórios*

A partir da página dos relatórios é possível verificar o dados estatísticos relativos a um determinado ecoponto. Estes dados são calculados a partir da informação obtida a partir dos ecopontos existentes na base de dados.

Teste: Aceder ao página dos relatórios e visualizar os dados estatísticos de um determinado ecoponto.

Resultado: Como é possível verificar na figura 45 a página contem uma *list-view* que contém os ecopontos existentes na base de dados. Ao clicar em qualquer um deles é exibido na mesma página a informação estatística em texto e em forma de gráficos.

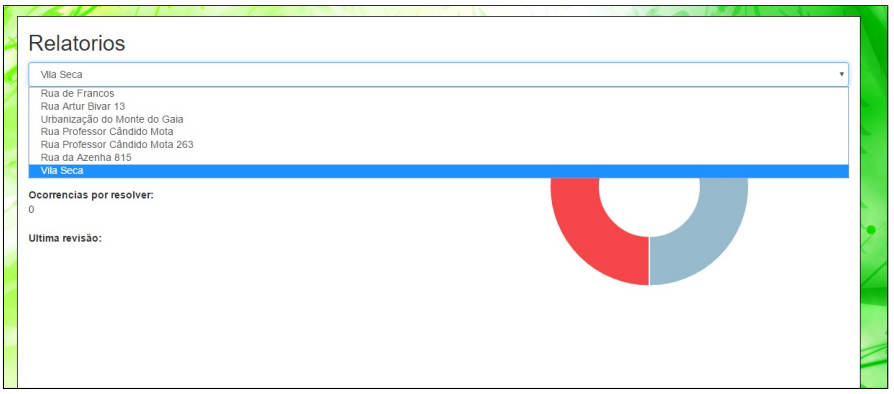


Figura 45: Página de relatórios ainda sem situações submetidas

5.2 TESTE REAIS

Com o testes de integração bem sucedidos chega a altura de verificar integridade do sistema em situações reais. Até agora todos os testes realizados as funcionalidades do sistema foram realizadas localmente, limitando severamente a área da sua viabilidade já que a aplicação móvel tem de ter acesso à uma rede cuja a comunicação com o servidor seja possivel. Como já foi referenciado um dos objetivos do sistema Ecowise é apresentar a capacidade de em qualquer localização o utilizador poder submeter situações referentes ao ecopontos. Para tal ser possível é necessário disponibilizar o servidor na *internet* de forma permitir o seu acesso e consequentemente à base de dados em qualquer local.

Os testes realizados são semelhantes aos que foram realizados anteriormente. Esses tinham como objetivo testar o funcionamento de cada um dos modelos individualmente de formar a tornar possível a sua integração num sistema viável. Os que serão realizados tem como propósito analisar o comportamento do sistema, em situações reais, como um todo de forma a concluir se o seu desempenho corresponde ao pretendido. Por essas razões deverão simular situações no nosso dia-a-dia em que a aplicação será mais utilizada.

5.2.1 *Realização dos Testes*

Qualquer submissão efetuada até agora foi efetuada nos limites de acesso a rede em que o servidor se encontrava a funcionar. Com a sua implementação online existe a possibilidade de acesso em qualquer localização. Assim é possível testar

as funcionalidades do sistema em situações em que o seu funcionamento correto é pretendido e tido em conta durante o seu desenvolvimento.

A aplicação será distribuída para 10 pessoas de forma a replicar o máximo possível o cenário pretendido. Estas pessoas efetuarão o processo típico pretendido para um utilizador comum. Os passos que os utilizadores devem realizar de forma a testar o bom funcionamento do são o seguintes:

- Registo/Autenticação - Estes passos são essenciais tanto para o acesso as funcionalidades da aplicação móvel como para a identificação do utilizador responsável por uma determinada submissão. Os utilizadores deverão inicialmente de se registarem através da aplicação e de seguida procederem a autenticação de forma a poder ter acesso as ferramentas da aplicação.

- Câmara/GPS - Um dos dois processos que o utilizador pode utilizar para a submissão de situações relativos aos ecopontos. Para ser bem sucedido o *smarthphone* necessita de conseguir os dados da sua localização. Caso tal não seja possível terá que tentar outra vez ou utilizar o leitor de código QR.

- Leitor de código QR - O segundo processo que o utilizador pode escolher utilizar. Ao iniciar o processo a câmara é iniciada e preparada para ler códigos QR. É apenas necessário uma passagem rápida para ocorrer a descodificação. De seguida um pequeno questionário tem de ser resolvido de a ter informação sobre o estado do ecoponto.

- Serviço de envio de mensagens - Ambos os processos preparam mensagens para serem enviados para o servidor. Um serviço é iniciado que examina a existência de conexão a *internet*. Caso não haja é iniciado um *timer* que verifica num determinado intervalo de tempo o estado da conexão para que as condições necessárias para o envio sejam atingidas.

Os utilizadores que estão a utilizar a aplicação estão todos em locais diferentes de forma a demonstrar a capacidade de aceder ao servidor é dependente unicamente da conexão a *internet*. Um dos objetivos principais com estes testes é averiguar as respostas provenientes do sistema ao maior número de situações possíveis.

5.2.2 *Resultados dos Testes*

Após uma recolha substancial de dados é agora possível efetuar uma analise mais pormenorizada e concluir se estes se encontram dentro dos valores esperados. Na figura 46 são apresentados alguns dos utilizadores que se encontram atualmente registados na base de dados.

É possível verificar que um dos campos nos documentos dos utilizador é o "role", sendo que existem dois tipos: "user"e "admin". Este campo existe de forma a diferenciar os utilizadores que efetuaram o seu registo adicionando mais uma camada de segurança ao sistema. Os utilizadores que efetuaram o registo pela aplicação móvel só conseguem fazer a autenticação nessa plataforma já que não apresentam autorização para aceder a informação relativa aos ecopontos nem as funcionalidades do *website*. As pessoas capazes de aceder a esse conteúdo ficam limitadas aos utilizadores que efetuaram o seu registo no *website* e que se encontram autorizadas para tal.


```
{
  "_id": "5804d60d1bfaf74606000002",
  "email": "artfaria@gmail.com",
  "name": "Artur Faria ",
  "location": "Barcelos",
  "role": "user",
  "__v": 0
},
{
  "_id": "5804dc781bfaf74606000004",
  "location": "fafe",
  "name": "kevin costa",
  "email": "kevin@gmail.com",
  "role": "admin",
  "__v": 0
},
{
  "_id": "5804dcfb1bfaf74606000005",
  "email": "bruno_esdm04@outlook.pt",
  "name": "Bruno Mendes",
  "location": "Guimarães",
  "role": "user",
  "__v": 0
}
```


Figura 46: Utilizadores registados na base de dados

Um dos objetivos essências que era necessário comprovar, era a capacidade de a aplicação de realizar ambos processos de registo/autenticação e de submissão de situações independente da rede e do local que o utilizador se encontra. Os registo que podemos verifica na imagem foram todos feitos na aplicação, com os dados inseridos pelos utilizadores a partir da suas localizações. Dessa forma comprovando que o servidor se encontra a funcionar corretamente e é capaz de receber e interpretar os pedidos provenientes a possibilidade de aceder ao servidor desde que a aplicação tenha acesso a *internet*.

Agora são analisadas as submissões efetuadas pelos utilizadores. Como é possível verificar na figura 47, o campo "status"indica o processo utilizado pelo o utilizador para a submissão dessa situação. Existem dois processos que podem utilizar para esse fim: a câmara com obtenção da localização GPS no ato de captura e o leitor de código QR.

O primeiro permite uma confirmação visual da situação dessa forma aumento a viabilidade da submissão e verifica-se que nas condições ideais é mais rápido e fiável no entanto é dependente da qualidade do recetor GPS do dispositivo que se encontra a utilizar. Caso a latitude e longitude recebidos não sejam razoavelmente próximos ao real pode comprometer uma identificação viável da rua e assim impedir a submissão da situação.

O segundo apresenta um processo de descodificação rápido e eficiente sendo que basta uma rápida passagem do leitor pelo símbolo para se obter um resultado e encontra-se unicamente dependente da câmara do dispositivo para um conseguir um bom funcionamento. Porem é necessário a introdução dos estado do ecoponto por parte do utilizador e é necessário que o símbolo esteja completamente legível para o leitor, o que muitas vezes não se verifica devidos aos atos de vandalismo que muitas vezes nos ecopontos.

Podemos assim concluir que ambos os processos apresentam vantagens e desvantagens no seu uso. Foi com estas características em mente que se decidiu desenvolver a aplicação móvel com estes dois processos. Quando um processo se revelar inadequado para ser utilizado existirá sempre outra opção disponível ao utilizador para realizar a sua submissão. Independentemente do processo utilizado é essencial a obtenção da localização em que o utilizador se encontra de foram a determinar qual o ecoponto a que uma determinada situação deve ser inserida.


```
"_id": "57d6b11ed101b24e09000002",
"log": -8.288094,
"lat": 41.453149,
"ecopoint_Red": false,
"ecopoint_Blue": true,
"ecopoint_Yellow": true,
"ecopoint_Green": true,
"location": "Rua de Francos",
"name": "Ecoponto da Universidade",
"id": 23467,
"__v": 3,
"situation": [
 {
   "_id": "57d8020da86a541015000004",
   "status": "Camera",
   "image": "/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDABALDA4MChAODQ4SERATGCgaGBYWGDEjJR0oOjM9PDkz\nODdASFxOQERXRTc4L
       /2wBDARESEhgVGC8aGi9jQjhCY2NjY2Nj\nY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2P/wAARCA
       /8QAtRAAAgEDAwIEAwUFBAQA\nAAF9AQIDAAQRBRIhMUEGE1FhByJxFDKBkaEII0KxwRVS0fAkM2JyggkKFhcYGRolJicoKSo0NTY
       5usLDxMXGx8jJytLT1NXW19jZ2uHi4+Tl5ufo6erx8vP09fb3+Pn6/8QAHwEA\nAwEBAQEBAQEBAQAAAAAAAAECAwQFBgcICQoL
       /8QAtREAAgECBAQDBAcFBAQAAQJ3AAECAxEEBSEx\nBhJBUQdhcRMiMoEIFEKRobHBCSMzUvAVYnLRChYkNOEl8RcYGRomJygpKjU
       nuLm6wsPExcbHyMnK0tPU1dbX2Nna4uPk5ebn6Onq8vP09fb3+Pn6
```


Figura 47: Listagem de um ecoponto com as correspondes situações

Com a submissão de situações bem-sucedida, vamos agora verificar se o *website* consegue tirar partido desses dados de forma a disponibilizar a informação necessária para o cumprimento dos objetivos pretendidos. Para isso é analisar as seguintes funcionalidades que utilizam dados provenientes do servidor o mapa, o calendário e os relatórios.

O mapa, como referido anteriormente, contém marcadores que correspondem aos ecopontos que se estão a monitorizar. Os ecopontos são inseridos manualmente pelos responsáveis pela sua manutenção sendo que sua inserção na base de dados é essencial para que o sistema funcione eficazmente, já que seria impossível determinar a qual ecoponto corresponde a uma situação submetida. Os dados necessários incluem obrigatoriamente a sua localização (localização GPS e nome da rua) e quais ecopontos é que existem nesse local porque apesar de atualmente o padrão existente ser o verde, amarelo e azul existem situações em que tal não ser verifica. Antigamente a reciclagem exclusiva unicamente ao vidro, que continua a ser o material mais reciclado nos dias de hoje, por isso ainda existem muitos pontos em que existe só o ecoponto verde e recentemente tem aparecido ecopontos vermelhos juntamente com os tradicionais chamados de Pilhão com o objetivo de reciclar pilhas usadas. Na recolha dos ecopontos é utilizado um camião diferente para cada material a reciclar por isso esta informação é importante para tornar o sistema mais eficiente.

Ao pressionar em qualquer um dos marcadores são exibido os casos não resolvidos existentes para aquele ecoponto. O conteúdo exposto depende do processo utilizado para a submissão dessa mesma situação. Se tiver sido utilizado a câmara é exposta a fotografia tirada. Caso tenha utilizado o leitor de códigos QR são apresentados os resultado do questionário realizado aquando da leitura. Ambos contém um botão que lhes permite determinar a situação como resolvida sendo desaparece do mapa e irà diretamente para o arquivo sendo depois unicamente acessível através do calendário. Na figura 48 é possível ver um dos exemplos das submissões efetuadas.

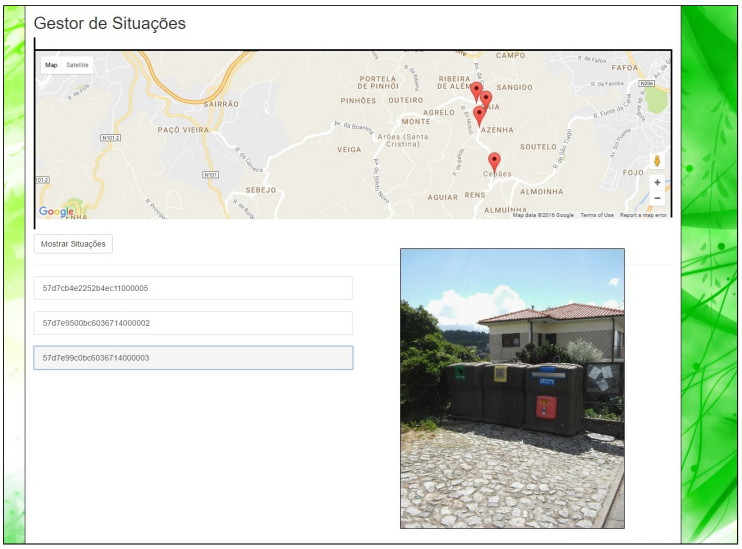


Figura 48: Situações submetidas disponibilizadas no mapa

O calendário funciona como um arquivo onde contém os casos já resolvidos organizados de acordo com a sua data de submissão. Podemos observar que os casos são apresentados como eventos com os nomes da rua em que o ecoponto se encontra. Ao clicar em alguns deles é exibido um *popout*, como está representado na figura 49. Este apresenta a informação relativo a essa situação de forma semelhante á do mapa com informação do estado do ecoponto naquela altura e outras informações consideradas importantes.

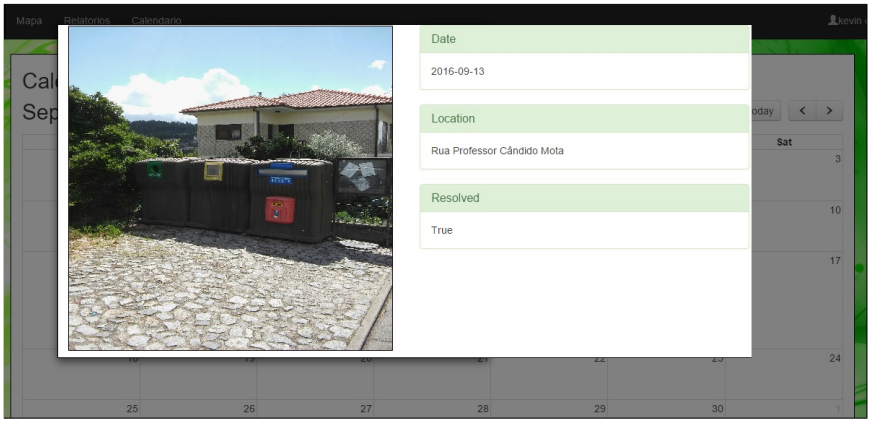
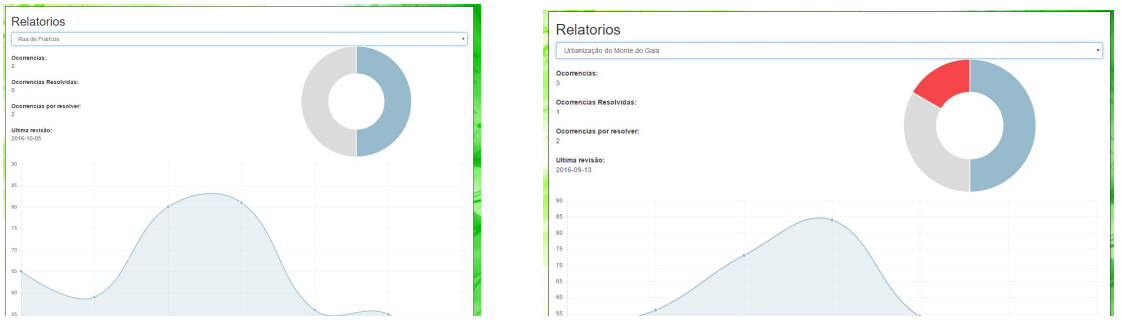


Figura 49: Situação acedida no calendário

Por fim temos os relatórios onde é organizada a informação relativa as situações obtidas relativas a um determinado e disponibilizada em forma de gráficos e dados estatísticos de forma a permitir uma analise sobre a atividade em volta de determinado ecoponto, isto é por exemplo, o numero de situações submetidas e quantas delas foram resolvidas ou ainda se encontram por resolver. A *listview*

contém todos os ecopontos existentes na base de dados. Escolher cada um deles permite exibir as estatísticas associadas a esse ecoponto. Esta informação permite determinar se uma rota encontra-se bem definida ou necessita de mudanças de forma a se tornar mais eficaz e a necessidade de mais ecopontos caso se verifique que os já existentes atingem o seu máximo de capacidade com uma velocidade insustentável. Ela é calculada durante o carregamento dos gráficos, dessa forma permitindo assim uma atualização constante (figura 50).



(a) Ecopontos sem situações resolvidas (b) Ecopontos com situações resolvidas

Figura 50: Exemplos da página dos relatórios

6

CONCLUSÕES E TRABALHO FUTURO

Neste capítulo é efetuada uma reflexão relativa ao desenvolvimento do projeto da dissertação. Também serão definidas propostas que podem ser implementadas no futuro, de forma a aumentar a rentabilidade e eficiência do sistema, bem como expandir as funcionalidades existentes e a sua fiabilidade.

6.1 CONCLUSÃO

No âmbito da presente dissertação foi implementado um sistema com o objetivo de tornar mais eficiente o processo de recolha de ecopontos através de uma monitorização mais rigorosa. Para isso tirou-se partido das tecnologias existentes nas aplicações móveis e da submissão de atualizações. relativamente ao estado dos ecopontos, por parte da população geral. O desenvolvimento de uma aplicação para plataformas móveis tornou possível a qualquer cidadão comum, com recurso a um leitor de códigos QR ou câmara com localização GPS, relatar o estado de um determinado ecoponto de uma forma fácil e acessível.

O *webservice* por sua vez foi desenvolvido de forma a apresentar a capacidade de receber e interpretar os pedidos recebidos independentemente de vir da aplicação móvel ou do *website*, e de controlar o acesso à base de dados. Na receção é realizada, uma analise de forma a descodificar a intenção do pedido e a resposta pretendida. Desta forma é possível restringir o acesso a informação existente na base de dados aos utilizadores pretendidos.

Com a informação pretendida já devidamente armazenada, é necessário disponibilizar a mesma às empresas de gestão dos Ecopontos. Através do *website* é possível visualizar os ecopontos que se encontram a monitorizar e quais necessitam

de atenção, estabelecer um registo das situações que necessitaram de intervenção o que já foram resolvidas e obter dados estatísticos que podem ser utilizados de forma a tornar mais eficiente o processo de recolha naquela região.

O sistema é baseado numa arquitetura REST que providência interoperabilidade entre a aplicação móvel, o servidor e o *website*. Permitindo assim uma comunicação rápida, eficaz e robusta entre os diferentes módulos existentes no sistema.

O serviço *EC2* da Amazon permitiu a criação de um servidor virtual com o sistema operativo Linux para o alojamento do *webservice* e da base de dados. Dessa forma permitiu acesso completo às funcionalidades do sistema independentemente da localização, desde que tenha acesso a *internet*.

Podemos concluir que os objetivos do projeto que foram inicialmente definidos foram cumpridos. O sistema criado permite às empresas de gestão de ecopontos uma vigilância mais eficaz através participação por parte dos cidadãos comuns que podem, através de dispositivos que são utilizados durante o nosso dia-a-dia, de uma forma simples e acessível relatar informações relativos aos ecopontos da sua região. Com o seu contributo será possível tornar o processo de recolha mais eficiente, e consequentemente melhorar o ambiente em que se vive.

6.2 TRABALHO FUTURO

Os objetivos propostos foram cumpridos, no entanto existem melhorias que podem ser aplicadas ao sistema através da adição de várias funcionalidades e aperfeiçoar a eficácia do processo.

Tornar o processo ainda mais seguro com o implementação de mais camadas de segurança, como por exemplo, restringir a inscrição no *website* unicamente aos trabalhadores da empresa, com métodos de autenticação mais rigorosos e diminuir a carga imposta sobre a base de dados com o armazenamento das imagens em repositórios, de forma a poupar espaço e tornar o processo de recolha de informação ainda mais rápido.

A criação de um mapa que contenha as rotas utilizadas pelos condutores responsáveis pela recolha, que providenciando a capacidade aos responsável pela

motorização de modifica-las de acordo com as informações obtidas pela aplicação e assim tornar-la mais eficiente.

Finalmente, o desenvolvimento da aplicação móvel para dispositivos com sistema operativo *iOS*, dessa forma abrangendo um maior numero de utilizadores e, consequentemente, aumentar a informação obtida.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] N. McCarthy, "The Countries Winning The Recycling Race," 2016. [Online]. Available: http://www.forbes.com/sites/niallmccarthy/2016/03/04/the-countries-winning-the-recycling-race-infographic/{#}6386790e4795

[2] I. B. Etc, S. C. P. February, and A. Reichel, "Municipal waste management in Portugal," no. February, 2013.

[3] N. F. da Cruz, P. Simões, and R. C. Marques, "Economic cost recovery in the recycling of packaging waste: the case of Portugal," *Journal of Cleaner Production*, vol. 37, pp. 8–18, 2012. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0959652612002764

[4] Z. Zsigraiova, V. Semiao, and F. Beijoco, "Operation costs and pollutant emissions reduction by definition of new collection scheduling and optimization of MSW collection routes using GIS. The case study of Barreiro, Portugal," *Waste Management*, vol. 33, no. 4, pp. 793–806, 2013.

[5] GOV.UK, "Landfill Tax rates." [Online]. Available: https://www.gov.uk/government/publications/rates-and-allowances-landfill-tax/landfill-tax-rates-from-1-april-2013

[6] L. Hickman, "A small town in Germany where recycling pays," 2011. [Online]. Available: http://www.theguardian.com/environment/2011/mar/18/recycling-waste

[7] "Reciclagem: Portugal longe das metas para 2020." [Online]. Available: http://www.tvi24.iol.pt/sociedade/lixo/reciclagem-portugal-longe-das-metas-para-2020

[8] I. Hernández, A. Viveros, and E. Rubio, "Analysis for the design of open applications on mobile devices," *Ieeexplore.Ieee.Org*, pp. 126–131, 2013. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=6525772

[9] D. Jagodi, D. Vuji, and S. Ran, "Android system for identification of objects based on QR code," vol. 7, pp. 922–925.

[10] "No Title." [Online]. Available: http://cdnqrcgde.s3-eu-west-1.amazonaws.com/wp-content/uploads/2013/11/jpeg.jpg

[11] "No Title." [Online]. Available: http://datagenetics.com/blog/november12013/anat.png

[12] H. Haas and A. Brown, "Web Services Glossary." [Online]. Available: https://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/{#}webservice

[13] "Outbound REST Web Service." [Online]. Available: http://wiki.servicenow.com/index.php?title=Outbound{_}REST{_}Web{_}Service{#}gsc.tab=0

[14] "No Title." [Online]. Available: https://upload.wikimedia.org/wikipedia/commons/4/4a/Webservices.png

[15] Tutorialspoint, "SOAP - Message Structure." [Online]. Available: http://www.tutorialspoint.com/soap/soap{_}message{_}structure.htm

[16] "No Title." [Online]. Available: http://flylib.com/books/2/439/1/html/2/images/f02mp01.jpg

[17] B. Burke, "Introduction to REST," in *RESTful Java with JAX-RS*. O'Reilly Media, Inc., 2009, ch. 1. [Online]. Available: https://www.safaribooksonline.com/library/view/restful-java-with/9780596809300/ch01.html

[18] "No Title." [Online]. Available: http://service-architecture.static-barryandassociates.com/images/web{_}services/rest{_}messages.jpg

[19] NivelBase, "GEPOR – GESTÃO DE ECOPONTOS." [Online]. Available: http://www.nivelbase.pt/?page{_}id=33

[20] P. Reis, R. Pitarma, C. Goncalves, and F. Caetano, "Intelligent system for valorizing solid urban waste," *2014 9th Iberian Conference on Information Systems and Technologies (CISTI)*, pp. 1–4, 2014. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6876904

[21] I. Hernández, A. Viveros, and E. Rubio, "Analysis for the design of open applications on mobile devices," *Ieeexplore.Ieee.Org*, pp. 126–131, 2013. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=6525772

[22] F. Maker and Y.-h. Chan, "A survey on android vs. linux," *University of California*, pp. 1–10, 2009.

[23] "No Title." [Online]. Available: http://cdn.edureka.co/blog/wp-content/uploads/2013/01/Android-Stack1.jpg

[24] Diffen, "Android vs. iOS." [Online]. Available: http://www.diffen.com/difference/Android{_}vs{_}iOS

[25] T.-M. Gronli, J. Hansen, G. Ghinea, and M. Younas, "Mobile Application Platform Heterogeneity: Android vs Windows Phone vs iOS vs Firefox OS," *2014 IEEE 28th International Conference on Advanced Information Networking and Applications*, pp. 635–641, 2014. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6838724

[26] "No Title." [Online]. Available: https://developer.apple.com/library/ios/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/Art/SystemLayers{_}2x.png

[27] "No Title." [Online]. Available: http://image.slidesharecdn.com/cloudcomputingsimpleppt-141114085742-conversion-gate01/95/cloud-computing-simple-ppt-5-638.jpg?cb=1443063081

[28] L. Ruebbelke, "5 Awesome AngularJS Features," 2012. [Online]. Available: http://code.tutsplus.com/tutorials/5-awesome-angularjs-features--net-25651

[29] "What Is Angular?" [Online]. Available: https://docs.angularjs.org/guide/introduction

[30] "No Title." [Online]. Available: http://vojtajina.github.io/ng-slides/2011-09-23-web-expo/img/two-way-db.png

[31] DigitalOcean, "SQLite vs MySQL vs PostgreSQL: A Comparison Of Relational Database Management Systems," 2014. [Online]. Available: https://www.digitalocean.com

[32] N. Leite, "Building your first application with MongoDB: Creating a REST API using the MEAN Stack," 2015. [Online]. Available: https://www.mongodb.com/blog/post/building-your-first-application-mongodb-creating-rest-api-using-mean-stack-part-1

[33] "No Title." [Online]. Available: https://docs.mongodb.com/v3.0/{_}images/crud-annotated-collection.png

[34] “MySQL vs MongoDB.” [Online]. Available: https://www.upguard.com/articles/mysql-vs-mongodb

[35] “No Title.” [Online]. Available: https://www.ntu.edu.sg/home/ehchua/programming/sql/images/SampleEmployees.png

[36] “MongoDB and MySQL Compared.” [Online]. Available: https://www.mongodb.com/compare/mongodb-mysql

[37] “No Title.” [Online]. Available: https://bytesforlunch.files.wordpress.com/2011/01/relational-vs-non-relational.jpg?w=700

[38] K. Peeples, “What are the Benefits of Node.js?” [Online]. Available: https://dzone.com/articles/what-are-benefits-nodejs

[39] “What can PHP do?” [Online]. Available: https://secure.php.net/manual/en/intro-whatcando.php

[40] “No Title.” [Online]. Available: https://servercheck.in/sites/servercheck.in/files/node-drupal-queue-comparison.jpg

[41] L. Orsini, “What You Need To Know About Node.js,” 2013. [Online]. Available: http://readwrite.com/2013/11/07/what-you-need-to-know-about-nodejs/

[42] “No Title.” [Online]. Available: http://devzone.co.in/wp-content/uploads/2015/09/01-avg-shapes657be.png