Instituto Politécnico de Tomar

Escola Superior de Tecnologia de Tomar

Miguel Filipe Redol Cotralha Inácio Coelho

Analytics Applied to Tourism

Trabalho Final de Mestrado

Orientado por:

Professor Luís Miguel Lopes de Oliveira, Instituto Politécnico de Tomar

Professor Renato Eduardo Silva Panda, Instituto Politécnico de Tomar

Trabalho Final de Mestrado

apresentado ao Instituto Politécnico de Tomar

para cumprimento dos requisitos necessários

à obtenção do grau de

Mestre em Engenharia Informática – Internet das Coisas

Resumo

Nos últimos anos tem-se assistido a uma enorme influência das novas tecnologias sobre os diversos sectores da sociedade. O sector do turismo não é exceção. Nos dias que correm o turista procura melhores ferramentas que tornem a sua visita numa experiência inesquecível, seja pela fotografia, vídeo ou experiências de realidade virtual.

O turismo de hoje procura acompanhar as inovações tecnológicas e ao longo dos últimos anos têm sido utilizadas diversas tecnologias que permitem uma maior interação entre o turista e os seus locais de visita. Este tipo de soluções oferece dois grandes benefícios: permitem ao turista aceder, de forma facilitada, a informação personalizada e, por outro lado, permitem a recolha de informação estatística sobre os próprios turistas, podendo esta servir para melhorar a própria oferta turística. No entanto, a maioria das soluções atuais tem-se focado sobretudo no primeiro ponto, não havendo ainda grandes soluções em funcionamento para a analítica de dados ao serviço do turismo.

Este projeto aborda o tema da analítica de dados sobre turismo. Para tal foram estudadas e implementadas soluções tecnológicas que permitam ao operador turístico extrair informações e conhecimento sobre os seus visitantes, auxiliando-o em decisões futuras.

Neste caminho, foram abordadas as diversas fases de um projeto de inteligência de negócio, desde a recolha de dados, criando mecanismos para garantir o correto funcionamento e maximizar a capacidade do sistema, até à fase final, fornecendo ferramentas que permitem explorar dados históricos e em tempo real. Estes mecanismos foram testados, estando implementados em cenário real no projeto “MovTour”.

São ainda sugeridas possíveis soluções e demonstrados alguns conceitos para a previsão de dados futuros com base no conjunto de dados históricos.

Palavras-chave: Turismo, Analítica, *BigData*, *Análise em tempo real*.

Abstract

In recent years there has been a huge influence of new technologies on the various sectors of society. The tourism sector is no exception. Nowadays tourists look for better tools that make their visit an unforgettable experience, whether by photo, video or virtual reality experiences.

Tourism today seeks to keep up with technological innovations and over the past few years a number of technologies have been used to allow greater interaction between tourists and their places of visit. This type of solution offers two major benefits: it allows the tourist to easily access personalized information and, on the other hand, allows the collection of statistical information about the tourists themselves, which can be used to improve their tourism offer. However, most of the current solutions have focused mainly on the first point, and there are not yet great solutions in operation for data analytics at the service of tourism.

This project addresses the topic of data analytics on tourism. To this purpose, technological solutions have been studied and implemented, which allow the tour operator to extract information and knowledge about their visitors, helping them to make future decisions.

In this way, the various phases of a business intelligence project, from data collection, creating mechanisms to ensure the correct operation and to maximize the capacity of the system to the final stage were presented, providing tools to explore historical data and real time. These mechanisms were tested, being implemented in real scenario in the project "MovTour".

Possible solutions are also suggested and some concepts for forecasting future data based on the historical data set are demonstrated.

Keywords: Tourism, Analytics, BigData, Real-time analysis.

Agradecimentos

A realização deste trabalho final de Mestrado contou com o apoio de diversas pessoas, às quais quero deixar o meu profundo agradecimento.

Em primeiro lugar, deixo o meu agradecimento aos orientadores deste projeto, professores Luís Miguel Lopes de Oliveira e Renato Eduardo Silva Panda.

A todos os meus amigos e colegas de mestrado, pelo apoio e colaboração, em especial ao comparsa Pedro Nunes.

Um especial e enorme agradecimento à minha namorada, pais e sogros, pelo apoio incondicional, amor, amizade e confiança que depositaram em mim.

A todos, o meu enorme obrigado!

Índice

Índice de figuras

Índice de tabelas

Índice de gráficos

Lista de abreviaturas e siglas

API	*Application Programming Interface*
APP	*Application*
BI	*Business Intelligence*
BLE	*Bluetooth Low Energy*
CSV	*Comma-separated values*
DDOS	*Distributed Denial of Service*
HTTP	*Hyper Text Transfer Protocol*
JSON	*JavaScript Object Notation*
NoSQL	*Not Only Structured Query Language*
SQL	*Structured Query Language*
SVM	*Support vector machine*
XLS	*Excel Spreadsheet*

Capítulo 1 – Introdução

O mundo digital tem revolucionado o dia-a-dia tanto de pessoas como de empresas. Estas transformações ocorrem constantemente, a grandes velocidades e das mais diversas formas.

Os avanços tecnológicos a que temos assistido nas últimas décadas, baseados nas novas tecnologias impõem um novo ritmo à sociedade, conduzindo-a a novos modos de produzir, comunicar e gerir dados. O fácil acesso à *internet*, a rápida adaptação às novas tecnologias e o uso de diversos sensores presentes nos dispositivos móveis potenciam evoluções tecnológicas cada vez mais dinâmicas.

O mundo em que hoje vivemos é cada vez mais global, moderno e dependente das novas tecnologias. Todos nós produzimos diariamente uma pegada digital, muitas vezes de forma inconsciente, através do uso de *internet* e computadores, *smartphones*, *smartwatches*, mas também ao fazer pagamentos eletrónicos, registar a presença no trabalho, entre outros. Esta informação é hoje extremamente valiosa, sendo usada para extrair conhecimento e criar novos negócios.

1.1. Enquadramento tecnológico

O século XXI será para sempre um marco histórico no Turismo [1]. As inovações tecnológicas têm vindo a transformar de tal forma este sector que existe um antes e um depois do segundo milénio na forma de viajar, não só ao nível do próprio turista, mas também ao nível do negócio.

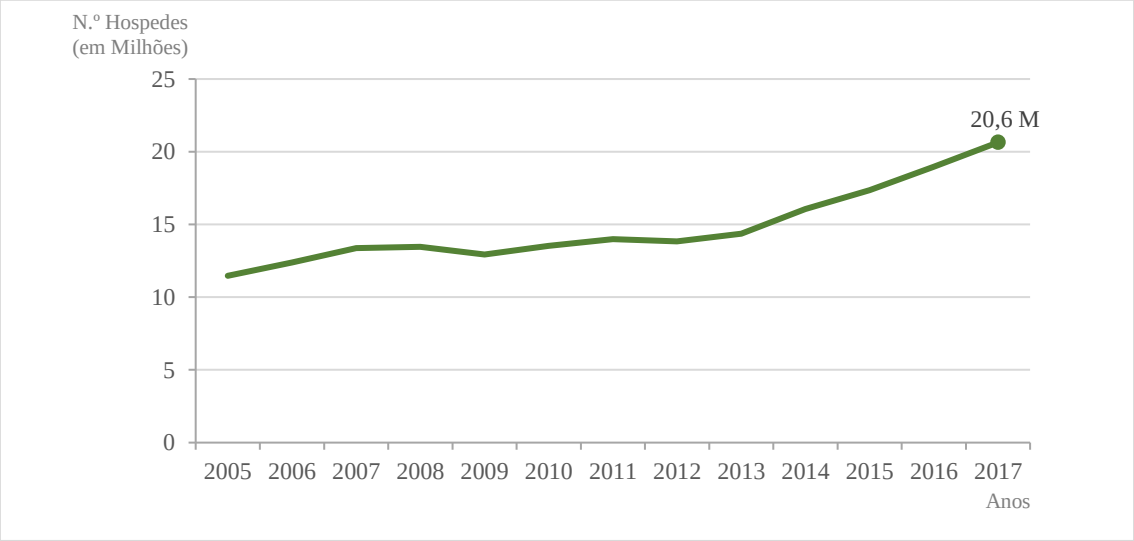
Nos dias de hoje o turista viaja sempre acompanhado de tecnologias, principalmente de dispositivos móveis. Os *smartphones* são os seus dispositivos preferidos, uma vez que na sua grande maioria se encontram equipados com câmaras fotográficas, rápidos processadores, diversos sensores e ecrãs a cores, dispõem de uma enorme capacidade de armazenamento e funcionam, em simultâneo, como guia virtual, tradutor, localizador de hotéis e mapas. Estas caraterísticas e funcionalidades associadas à dependência destes equipamentos por parte do ser humano, proporcionam uma aposta cada vez maior e mais forte no desenvolvimento de aplicações para o sector turístico.

A evolução tecnológica está a modificar os hábitos turísticos, transformando-os numa experiência tecnológica inesquecível. A realidade virtual, a realidade aumentada e a geolocalização através de *beacons* são cruciais na evolução tecnológica deste sector [2].

A substituição dos suportes tradicionais e habituais pelas novas tecnologias de conteúdos multimédia em aplicações móveis são uma ferramenta cada vez mais eficiente na divulgação da informação em qualquer momento e em qualquer lugar. As novas tecnologias permitem melhorar os serviços prestados, enriquecer a experiência do turista e estimular o *marketing*. No entanto, este tipo de tecnologias está ainda pouco disponível aos agentes locais e gestores de monumentos. Existe a possibilidade de usar os mesmos para extrair informação sobre os visitantes e com isto obter algum conhecimento que permita otimizar os seus serviços. Esta área de *data mining* e análise de grandes volumes de dados está ainda por explorar. Embora exista um número crescente de visitantes em Portugal são poucos os dados recolhidos sobre os mesmos e ainda menos informação é extraída destes.

Portugal é um destino turístico que está na moda e o tem sido divulgado em todo o mundo. Em 2018 foi eleito o melhor destino turístico europeu, pelo segundo ano consecutivo, pelos *World Travel Awards* [3].

Gráfico 1: Evolução do número de hóspedes em Portugal (2005-2017)



Fonte: ECO [4]

A atividade turística em Portugal tem vindo a crescer de forma gradual, afirmando-se como um dos motores da economia nacional. O ano de 2017 foi um ano recorde neste sector, onde mais de 20 milhões de visitantes ficaram hospedados em Portugal [4]. O

turismo no território nacional tem como plano aumentar significativamente estes números até 2027.

De uma forma geral, a utilização das novas tecnologias não está a ser feita na sua plenitude pela grande maioria da população [5]. O seu uso não se pode cingir apenas ao acesso às redes sociais e ao entretenimento. A revolução tecnológica vai muito mais além.

As tecnologias digitais são inúmeras e com limites inimagináveis. As conjunturas oferecidas pela conectividade, pela realidade aumentada, pelos robôs e pela inteligência artificial irão revolucionar ainda mais o sector turístico na próxima década. Cabe a cada um de nós tornar estes avanços tecnológicos em grandes aliados do nosso cotidiano.

1.2. Motivação e objetivos

O mercado turístico encontra-se em constante mudança, não só pela tendência da procura e diversificação da oferta, mas principalmente pela evolução incomparável em tempo real das novas tecnologias.

A tecnológica é a linha de qualificação fundamental para a gestão integrada dos principais componentes do turismo como viagens, hospedagem, alimentação e bebidas, e entretenimento e lazer. Esta componente não contribui apenas para responder à questão da qualidade da informação para a tomada de decisão, mas também para o conhecimento que contribui cumulativamente para o desenvolvimento de processos e produtos turísticos. O modelo tradicional de turismo é condicionado por essas transformações que absorvem a diversificação da oferta, novos produtos e novos mercados emergentes e, portanto, há a necessidade de componentes tecnológicos que garantam a sua continuidade como um sector forte da economia.

A necessidade de mais conhecimento sobre a cultura turística é evidente, de forma a valorizar a experiência turística e a preservação do património. Esse conhecimento ainda é escasso e limitado. Assim, o uso de tecnologias de monitorização e avaliação em tempo real é relevante para determinar o “uso” do patrimônio cultural em termos turísticos.

A utilização de componentes tecnológicas como deteção de *wi-fi*, *beacons Bluetooth Low Energy* (BLE), aplicações móveis e soluções de análise, podem conduzir há obtenção de mais informações sobre os visitantes e, portanto, melhorar os modelos de negócios.

O projeto aborda esta lacuna e visa construir soluções tecnológicas para fornecer informações valiosas para os tomadores de decisão do turismo.

Os objetivos específicos deste trabalho são:

- Garantir a correta recolha de dados e funcionamento do sistema de acordo com o volume de visitantes esperados, desenvolvendo testes de estabilidade e conformidade e aplicando as otimizações adequadas;
- Implementação de soluções de *BigData e* analítica de dados, para facilitar o armazenamento e a análise dos dados recolhidos;
- Implementação de soluções dinâmicas que permitam recolher e analisar os dados em tempo real.

O trabalho de projeto foi desenvolvido no âmbito do "Turismo Digital". Neste projeto todos os dados tratados e analisados são provenientes do projeto "MovTour".

1.3. Descrição da solução proposta

A solução proposta consiste no desenvolvimento de uma componente de inteligência de negócios para fazer a análise dos dados obtidos pelo projeto "MovTour". O projeto "MovTour" consiste no desenvolvimento de um sistema para a gestão de pontos de interesse e monumentos de forma a recolher informação sobre os seus visitantes e lhes fornecer informações pertinentes sobre os locais que visitam. Desta forma a solução proposta passa por:

- Projetar e desenvolver as soluções para fornecer informações sobre a recolha de dados pelo projeto "MovTour";
- Apresentar soluções de *design* e implementação para representar as informações recuperadas no formato de painel em tempo real;
- Desenvolver uma ferramenta de criação de relatórios personalizada;
- Implementar mecanismos de otimização de desempenho no *backoffice* e servidor.

As funcionalidades das ferramentas propostas a desenvolver são:

- Consulta de dados dos visitantes de forma dinâmica, definida de acordo com os parâmetros de interesse do administrador do *backoffice*. O principal objetivo é permitir construir por si gráficos com a informação pretendida;
- Fornecer informação em tempo real, do tipo "*real time dashboard*", no painel de administração, onde se visualize em tempo real os acessos aos pontos de interesse e monumentos. Com isso, permitir ao administrador visualizar informações como: número de pessoas que se encontram em determinado ponto de interesse ou monumento, número de pessoas que saíram e entraram nos últimos minutos, entre outras;
- Aumento do desempenho, com a implementação de mecanismos de otimização como sistemas de *caching* eficientes (fragmentos) e otimização de *queries* de pesquisa;

1.4. Organização do relatório

Este relatório está divido em cinco capítulos.

No capítulo 1 é elaborada uma introdução, composto por um enquadramento tecnológico, onde:

- São referidas as motivações e objetivos deste trabalho
- É descrita a solução proposta; e
- É apresentada a estrutura do relatório.

Seguidamente, no capítulo 2, é apresentado o estado de arte e as soluções existentes para atingir os objetivos definidos.

O capítulo 3 aborda a arquitetura do sistema enquanto o capítulo 4 apresenta os resultados obtidos sobre as soluções implementadas no projeto.

Por último, no capítulo 5, é elaborada a conclusão, onde são apresentadas as principais conclusões, as limitações ao estudo e pistas para futuras investigações.

Capítulo 2 – Estado de arte e Background

Conhecer por exemplo o número de visitas de dado monumento é bastante conveniente. No entanto, saber de onde estas provêm, que pontos visitaram e durante quanto tempo (entre outras informações) é bastante mais significativo. Para este fim existem diversas soluções, tais como mecanismos de *BigData* e ferramentas de analítica de dados. Estas soluções permitem ter uma melhor visão sobre os dados recolhidos. Desta forma, é possível extrair conhecimentos, que de grosso modo tentam identificar padrões nos dados recolhidos, com o intuito de serem usados para previsões futuras.

De seguida são abordadas cada uma das duas variantes citadas.

2.1. Soluções para *BigData* e análise de grandes volumes de dados

O processo de armazenamento, pesquisa e receção de informação foi dos que sofreu uma maior transformação ao longo dos últimos anos, acabando por aumentar as expectativas no que diz respeito à informação digital e aos conteúdos relevantes para determinado tema ou contexto.

Inicialmente o conceito de *BigData* foi contemplado por 3V's: Volume (quantidade de dados disponíveis), Velocidade (necessidade de gerar informação com a maior agilidade possível para as tomadas de decisão serem efetivas) e Variedade (quantos mais dados e fontes existirem, maior é a possibilidade de se gerar informação útil). Alguns anos mais tarde, foram adicionados: a Veracidade (o quanto a informação é verdadeira) e o Valor (a informação terá de ter valor para o caso em que está a ser utilizada, pois gerar informação sem utilidade não tem interesse) [6].

O objetivo do *BigData* no turismo é fornecer dados sobre a atividade e o comportamento do ser humano, podendo, assim, aumentar os benefícios, não só para os clientes do turismo, mas também para todos que usufruem do mesmo.

Até há pouco tempo, todos os dados recolhidos acerca dos turistas eram baseados em inquéritos. Isto quer dizer que apenas era conseguido armazenar uma amostra mínima de dados/informações.

Atualmente, o que se pretende com o *BigData* no turismo é a possibilidade de guardar e analisar um maior volume de dados de forma automática.

Estes novos dados gerados pelos turistas têm uma maior confiabilidade do que os dados gerados através de inquéritos. Isto acontece uma vez que o utilizador guarda de forma automática, por exemplo, os percursos que realizou em determinado local e período. Esta informação tem mais valor do que a informação que o próprio turista descreveria num inquérito.

A informação gerada, muitas vezes de forma automática, pelo turista ao visitar uma região utilizando o seu dispositivo móvel é ainda hoje muito pouco aproveitada. No entanto, esta pode permitir às autoridades competentes pelo turismo verificar não só os monumentos mais visitado num determinado local, mas também as horas a que são visitados, entre outras informações.

Devido à enorme quantidade de dados armazenados, a analítica de dados é necessária para retirar informação dos mesmos.

Nos dias de hoje, antes de marcar uma viagem, o visitante procura obter vários tipos de informações, tais como: hotéis, pontos turísticos a visitar, passeios, entre outros. É nestas alturas que a analítica de dados entra para o ajudar a decidir o que fazer.

Ao se considerar que o objetivo da análise de grandes volumes de dados é conseguir obter informação e conhecimento destes, então existem vários tipos de análises. Dentro destes diversos tipos destacam-se quatro áreas: análise descritiva, análise diagnóstica, análise preditiva e análise prescritiva

A análise descritiva, tem como base a compreensão em tempo real dos acontecimentos. Esta análise é muito usada no sector financeiro, nomeadamente nos Bancos. Estas instituições analisam todo o histórico de um indivíduo, de uma empresa ou de um grupo social para então compreenderem os riscos envolvidos na concessão, por exemplo, de um crédito.

Enquanto a análise descritiva procura detalhar os dados, a análise diagnóstica tem como objetivo compreender de forma pontual todas as suas possibilidades (“Quem?”, “Quando?”, “Como?”, “Onde?”). Esta análise tem o caminho mais curto e eficiente para avaliar os impactos de uma ação.

A partir da identificação de padrões, a análise preditiva permite a identificação de tendências futuras. O seu grande objetivo é ajudar o ser humano a deixar de tomar decisões baseadas apenas na sua intuição, obtendo assim um prognóstico sólido para cada ação efetuada. Este tipo de análise utiliza *data mining*, onde são obtidos dados estatísticos e dados históricos, de forma a conhecer futuras tendências.

Por fim, a análise prescritiva, que por vezes é confundida com a análise preditiva por trabalhar com a mesma lógica, traça possíveis consequências das ações. Esta, é uma forma de definir qual a escolha mais acertada em determinada situação [7].

Ao trabalhar com grandes volumes de dados é necessário um sistema de dados que consiga suportar tanto o armazenamento como a leitura de tamanha quantidade de informação. Existem diversas soluções para este efeito, algumas comerciais enquanto outras são tecnologias livres e de código-fonte aberto.

De seguida, são apresentadas algumas destas soluções.

2.1.1. Base de dados

2.1.1.1. *Hadoop*

Hadoop é uma estrutura de *software* de código-fonte aberto (*open source*) para armazenamento distribuído de grandes volumes de dados em *cluster* de computadores [8]. *Cluster* é o nome dado ao conjunto de computadores que trabalham de forma sincronizada para funcionar como um único computador. Quando o volume de dados é enorme, um computador não consegue processar toda essa informação, para isso é utilizado os *clusters*. Isto significa que os dados podem ser escalados sem a preocupação de falhas com o *hardware*.

O *Hadoop* permite o armazenamento para quaisquer tipos de dados e a capacidade de lidar com tarefas concorrentes praticamente ilimitadas. A sua capacidade de armazenamento, gestão e análise de grandes quantidades de dados, tanto estruturados como não-estruturados, de forma rápida, confiável e flexível faz com que o *Hadoop* seja facilmente uma escolha para este tipo de cenário [9].

2.1.1.2. *Apache Cassandra*

Apache Cassandra é, atualmente, uma das base de dados NoSQL mais eficientes. Esta é um sistema de base de dados distribuído de código-fonte aberto extremamente poderoso que funciona para lidar com grandes volumes de dados. É a escolha ideal para implementação em sistemas que necessitem de escalabilidade e alta disponibilidade sem comprometer o seu desempenho [10]. A *Apache Cassandra* reparte a informação entre *clusters.* Cada um tem a sua função, onde é capaz de reconhecer dados replicados, conseguindo executar determinadas ações por redundância dos mesmos. É também escalável e tolerante a falhas, pois replica o mesmo dado em diferentes nós. Permite ainda a integração com outras soluções de *BigData,* como por exemplo o *Hadoop* [11].

A utilização da *Apache Cassandra* não é aconselhada se for implementada numa solução que exija operações transacionais, registos primários ou financeiros, consultas dinâmicas dos dados ou latência baixa [12].

Aplicações como *Netflix, Reddit* e *Sound Cloud* são exemplos de aplicações que utilizam este sistema de base de dados [13].

2.1.1.3. *MongoDB*

MongoDB é a abordagem moderna que oferece o melhor dos dois mundos: relacional e NoSQL. As vantagens das base de dados relacionais permanecem, assim como a sua fácil configuração, permitindo a criação de soluções mais rápidas. Esta base de dados é orientada a documentos, no formato JSON, não possuindo como restrição a necessidade de ter as tabelas e colunas criadas previamente. É usada quando os dados mudam com frequência ou quando os mesmos são semiestruturados ou não estruturados. Os documentos são agrupados em coleções e um conjunto de coleções forma uma base de dados.

Não esquecendo a análise em tempo real, o registo em alta velocidade, o armazenamento em *cache* a alta escalabilidade, o grande ponto forte do *MongoDB* é a sua flexibilidade, pois a estrutura orientada a documentos permite com facilidade gravar os dadas da forma que for melhor para a solução [11].

Caso a solução exija um sistema altamente transacional ou requisitos de base de dados tradicionais, como por exemplo restrições de chave estrangeira, não deve ser utilizado.

Empresas como o “ebay”, “Facebook”, “Google” e “SAP”, são exemplos que utilizam esta base de dados [14].

2.1.1.4. *PostgreSqL*

O *PostgreSQL* é um poderoso e dos mais avançados sistemas tradicionais de gestão de base de dados relacional de objetos, de código-fonte aberto. Ele usa e estende a linguagem SQL combinando recursos que armazena e dimensiona com segurança.

Este ganhou uma forte reputação pela sua arquitetura, comprovando confiabilidade, integridade de dados, conjunto robusto de recursos e extensibilidade. Este sistema contém vários recursos destinados a ajudar na criação de aplicações, na proteção da integridade dos dados, na criação de ambientes tolerantes a falhas e na gestão dos dados, independentemente do tamanho do conjunto (de dados). O *PostgreSQL* suporta um modelo de dados que consiste numa coleção de relações nomeadas, contendo atributos de um tipo específico [15].

Podem existir situações em que o *Hadoop* ou a *Cassandra* sejam os melhores locais para armazenar determinados dados, provenientes de *Internet Of Things*, dependendo do volume.

Não são impostos limites para o tamanho máximo da base de dados, existindo base de dados de 32TB. O tamanho máximo de uma tabela são de 32TB, o tamanho máximo de uma linha 400GB e o tamanho máximo de um campo é de 1GB [16].

2.1.2. Ferramentas de análise de dados

O *IBM Cognos analytics* contem um conjunto de ferramentas integrado de inteligência de negócios que fornece uma ampla variedade de funcionalidades para ajudar a entender os dados de um conjunto. As suas principais funcionalidades são: a criação de relatórios, análise e monotorização de eventos e criação de *dashboards*.

Esta ferramenta combina duas grandes áreas, *Business Intelligence* (BI) e análise exploratória numa única solução.

Business Intelligence é a recolha e análise de grandes volumes de dados estruturados para fornecer visualizações históricas, atuais e preditivas. Conjuntos de dados, esquemas e consultas pré-determinados compõem a arquitetura do BI.

A análise exploratória é projetada para utilizadores que não se encontram familiarizados com estas novas ferramentas. Por exemplo, num departamento de vendas e *marketing*, é utilizada para explorar dados a serem usados nas tomadas de decisão [17].

Também a empresa "Google" tem ferramentas de análise de dados. *Google Analytics* é um serviço grátis com o objetivo de gerar relatórios que mostram, entre outras coisas, a página mais visitada, o número de visitantes únicos, o número de utilizadores que visitaram a página num determinado dia, análise de redes sociais e até mesmo análise de publicidade.

Esta ferramenta pode ajudar nas tomadas de decisão baseadas nos dados que recolhe. Estes dados são categorizados em 5 grupos: Aquisição, Comportamento, Conversão, Audiência e Tempo real [18].

Começando pela aquisição, esta refere como é recebido o tráfego na plataforma, mais precisamente, por onde entraram os utilizadores. Os seus relatórios fornecem dados dos melhores caminhos para trazer o tráfego para a plataforma, podendo observar qual o canal com mais utilização, seja ele por referência em alguma plataforma, pesquisa num motor de pesquisa, por entrada direta ou entrada através da rede social.

Já o comportamento refere como os utilizadores utilizam a plataforma. Como é possível otimizar o seu desempenho e determinar se os utilizadores estão a executar as ações que são pretendidas, recolher dados referentes, por exemplo, ao total de páginas visitadas, tempo médio numa página, recolher valores sobre como os visitantes interagem com as várias páginas, entre outras.

Relativamente às conversões, estas ocorrem sempre que um utilizador realiza uma ação desejada na plataforma. Isso pode ser feito por meio do preenchimento de um formulário, da conclusão de uma compra ou da simples exibição de alguma página específica.

Para se saber quem compõe o tráfego da plataforma, é utilizada a seção da Audiência. Isto quer dizer que é possível saber se a plataforma está a ser acedida por um dispositivo móvel ou por um *desktop*. Para complementar esta informação, existe a parte do tempo

real que fornece dados sobre quem está e o que está a fazer na plataforma no momento da consulta.

2.1.3. Conclusão

O mercado de ferramentas baseadas no *BigData* está a crescer cada vez mais. O elevado número de pessoas que possui um *smartphone* aumenta de ano para ano. Com este aumento de dispositivos na rede, também aumenta a quantidade de dados que circulam na *internet*.

O *BigData* é visto como uma das principais tecnologias para empresas que procuram formas de melhorar as suas campanhas de venda e estratégias comerciais. Apesar da análise de dados já fazer parte do dia-a-dia de vários negócios, foi com o auxílio dessa tecnologia que negócios se tornaram capazes de cruzar registos encontrados em meios digitais (como redes sociais e aplicações moveis), com informações obtidas em pesquisas para a criação de campanhas de *marketing* de sucesso.

Este também pode ser visto como uma forma de otimizar a estrutura interna de empresas e organizações de outras áreas. Serviços de saúde, processos de tomada de decisão e planos de investimento passam a ter um nível de precisão mais elevado.

Das várias soluções de armazenamento apresentadas, a que foi selecionada para implementação no projeto foi o *PostgreSQL*. Todas elas são capazes de resolver os problemas atuais no cenário do projeto. Tendo em conta a quantidade de dados a gerar, possivelmente todas as outras são soluções sobredimensionadas para o problema em causa. É mais exequível desenvolver mecanismos de análise sobre modelo de dados *PostgreSQL* do que para os outros sistemas de base de dados, uma vez que são mais complexos.

Das ferramentas de análise de dados, foi escolhido o *IBM Cognos Analyrics* como uma das soluções para análise dos dados gerados pelos turistas. Os resultados poderão ser exportados para esta plataforma para assim se obter relatórios e gráficos das soluções ou tendências.

2.2. Aprendizagem automática ou computacional

Tendo um conjunto grande de dados podem ser utilizadas ferramentas de *BigData* para análise e extração de informação do conjunto. Outra vertente, cada vez mais importante hoje em dia, é a utilização de técnicas de aprendizagem automática, tanto para deteção de padrões como para previsão de eventos futuros. Este tipo de soluções está presente em grande parte dos serviços que utilizamos, desde filtros de spam, processamento de linguagem natural, deteção de caras e outros objetos em imagens, entre outros. Um exemplo deste tipo de abordagem na área do turismo pelo sistema descrito no artigo científico "*Tweet-mapping method for tourist spots based on now-tweets and spot-photos*" [19]. Este utiliza a rede social "Twitter" para, através da junção da geolocalização dos *tweets* publicados juntamente com as fotos associadas a cada uma dessas publicações, classificar pontos de turístico, utilizando para isso algoritmos de *clustering*.

2.2.1. Algoritmos de classificação de dados

O processo de aprendizagem computacional, de um modo geral, consiste na extração de estatísticas de um conjunto de dados, que depois são utilizados com um dos vários algoritmos existentes para o efeito. Estes algoritmos procuram padrões nos dados fornecidos, de maneira a que possam, por exemplo, separar os mesmos em grupos distintos (*clustering*) ou classificar novos dados futuros com base nos padrões encontrados.

Os algoritmos de classificação dividem-se em dois grandes grupos: algoritmos de classificação supervisionada e não supervisionada. A sua principal diferença é que os algoritmos de classificação supervisionada necessitam de um "professor" para avaliar a resposta do algoritmo e nos algoritmos de classificação não supervisionada não existe esta figura, pelo que a resposta terá de vir inteiramente do resultado do algoritmo.

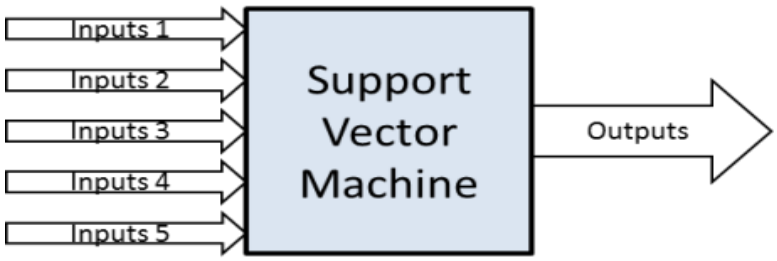
Existem vários tipos de algoritmos de classificação que se englobam nos dois grupos mencionados anteriormente, tais como o algoritmo de classificação (previsão de classes ou categorias) e regressão (previsão de valores numéricos), algoritmo de *clustering* (agrupar dados semelhantes) e algoritmo de associação (descoberta de elementos que ocorrem em comum dentro de um determinado conjunto de dados).

2.2.1.1. Algoritmos de classificação supervisionada

2.2.1.1.1. *Support Vector Machine* (SVM)

As *Support Vector Machines* são uma das técnicas mais utilizadas em *Data Mining* quando é necessário construir um classificador para dados constituídos maioritariamente por variáveis numéricas contínuas/discretas [20]. No entanto, dada a sua complexidade existem várias variáveis que podem e devem ser ajustadas aquando da construção do classificador e a compreensão das mesmas necessita de um estudo aprofundado. Muitas das vezes, torna-se difícil obter uma justificação clara para o classificador obtido. Contudo, ao serem utilizadas as ferramentas certas e os valores por defeito, é possível construir um classificador de forma rápida e utilizá-lo, por exemplo, através da plataforma WEKA2.

Figura 1: Funcionamento básico de uma SVM



Fonte: Readhead [21]

Na imagem acima é possível observar que uma SVM necessita de vários *inputs* para produzir um *output*.

2.2.1.1.2. Árvores de decisão

Uma árvore de decisão pode ser vista como um fluxograma que representa de forma gráfica o processo de tomada de decisão. As empresas podem utilizá-la diante de situações do dia-a-dia, como conceder um empréstimo a um potencial cliente ou realizar análises financeiras [22].

Por outro lado, os indivíduos comuns podem necessitar delas para serem ajudados em algum processo do quotidiano. O processo de decisão faz parte da vida do ser humano.

Para analisar grandes volumes de dados são necessários modelos computacionais que possam fornecer uma determinada precisão e agilidade.

2.2.1.1.3. Classificação *Naïve Bayes*

A classificação *Naïve Bayes* é uma técnica baseada no teorema de *Bayes* com uma suposição de independência entre os preditores. Em termos simples, um classificador *Naive Bayes* assume que a presença de uma característica particular numa classe não está relacionada com a presença de qualquer outro recurso. Por exemplo, um fruto pode ser considerado como uma maçã se for vermelho, redondo e tiver cerca de 3 polegadas de diâmetro. Mesmo que estes recursos dependam uns dos outros ou da existência de outras características, todas estas propriedades contribuem de forma independente para a probabilidade de que o fruto possa ser uma maçã, e é por isso que é conhecido como '*Naive*' [23].

O modelo *Naive Bayes* é fácil de construir e particularmente útil para grandes conjuntos de dados. Além de simples, *Naive Bayes* é conhecido por ganhar a métodos de classificação altamente sofisticados.

2.2.1.2. Algoritmos de classificação não supervisionada

2.2.1.2.1. *K-means*

O algoritmo *K-means* é do grupo de algoritmos de *clustering*. O seu objetivo é separar os objetos de teste em grupos. Para isso é necessário ter-se em conta as características dos objetos. Este algoritmo tem como ideia a colocação de objetos similares no mesmo grupo. Cada *cluster* tem o seu centro.

O *K-means* tem como base uma técnica de agrupamento não hierárquico. Este procura minimizar a distância dos elementos a um conjunto de K centros iterativamente, em que K é o número de *clusters*. Dado um objeto, é calculado a distância desse mesmo objeto ao centro de cada *cluster* para se determinar a qual o que pertence a esse objeto [24].

Ao longo do tempo, o centro de cada grupo vai mudando. Para calcular o centro de cada grupo, apenas é necessário calcular a média (*means*) dos valores dos objetos que se encontram naquele grupo.

2.2.2. Ferramentas computacionais

O *IBM Watson* é um sistema baseado em computação cognitiva que inicialmente foi criado como um sistema de pergunta e resposta. Este é um sistema que responde às perguntas que lhe são colocadas. A computação cognitiva é uma mistura de diferentes técnicas que se dividem em, principalmente, cinco grupos, *machine learning*, processamento de linguagem natural, inteligência artificial, interação humana e raciocínio [25].

O *machine learning* é uma técnica de analise de dados que automatiza a construção de modelos analíticos e que se foca na ideia de que os sistemas podem aprender com os dados que lhe são fornecidos, identificar padrões e até mesmo tomar decisões sem a intervenção humana.

O processamento da linguagem natural é uma maneira pela qual o ser humano pode interagir com computadores com uma linguagem que falamos diariamente. A inteligência artificial é a forma pela qual os computadores executam algumas tarefas que realmente precisam da inteligência humana. Já a interação humana é a maneira pela qual os computadores podem interagir com os humanos. O raciocínio é a parte em que o sistema realmente pensa como humanos e produz respostas para as perguntas.

A principal inovação do *Watson* não foi na criação de um novo algoritmo para analisar questões em diferentes palavras-chave ou em fragmentos de frase, mas sim na sua capacidade de executar rapidamente centenas de algoritmos de análise de linguagem simultaneamente. Quanto mais algoritmos encontrarem a mesma resposta, maior será a probabilidade de o *Watson* estar correto. Uma vez que este tenha um pequeno número de possíveis soluções, ele é capaz de verificar na sua base de dados se a solução faz sentido ou não.

Tabela 1: Serviços disponibilizados pelo IBM Watson

Serviço	**Descrição**
Discovery	Descoberta de conexões profundas em todos os dados
Watson Assistant	Construção e implementação de agentes virtuais numa variedade de canais, incluindo dispositivos moveis e plataformas de mensagens
Tone Analyzer	Compreensão de emoções e estilo de comunicação em texto
Speech to Text	Conversão de áudio e voz em texto escrito
Text to Speech	Conversão de texto escrito em som natural numa variedade de idiomas e vozes
Language Translator	Conversor de texto entre idiomas
Visual Recognition	Busca de conteúdo em imagens
Natural Language Undestanding	Processamento de linguagem natural para texto
Personality Insights	Recolha de características de personalidade, necessidade e valores em textos
Natural Language Classifier	Atribuição de categorias personalizadas a texto
Watson Studio	*Framework* de análise de dados para implementação num projeto
Machine Learning	Conjunto de *Application Programming Interface* (API) para ajuda a tomada de decisões e resolução de problemas complexos
Knowledge Catalog	Criação de uma visão global dos dados não importando onde eles estejam armazenados
Compare Comply	Coleção de API's que permitem melhor e mais rápida compreensão de documentos
Knowledge Studio	Ferramenta de construção de modelos de *machine learning* num ambiente colaborativo

Fonte: IBM Cloud [26]

A tabela acima resume os serviços que são disponibilizados pelo *IBM Watson*.

Outro sistema bastante conhecido é o *Google AI* que é um ramo de pesquisa e desenvolvimento de inteligência artificial da "Google". O *Google AI* continua a evoluir no ramo do *machine learning* com a construção de modelos para reconhecimento de imagens. Existe uma parte denominada de *DeepMind* que planeia criar um conjunto de poderosos algoritmos de aprendizagem que combinados possam criar um sistema de inteligência artificial. Este utiliza dados em bruto como entrada e aprende com a experiência.

Capítulo 3 – Arquitetura do Sistema

O projeto "MovTour" consiste num sistema para apoio ao turismo composto por 3 sistemas distintos, tal como ilustrado na figura 5:

1. *Beacons Bluetooth* – pequenos dispositivos que são instalados em cada ponto de interesse para identificação do mesmo;
2. Aplicação móvel – a ser utilizada pelo turista;
3. Sistema de *backoffice* – servindo de suporte para a gestão e recolha de toda a informação turística do sistema.

Figura 2: Ecrã inicial da aplicação móvel

Fonte: MovTour

Figura 3: Lista de monumentos da aplicação móvel

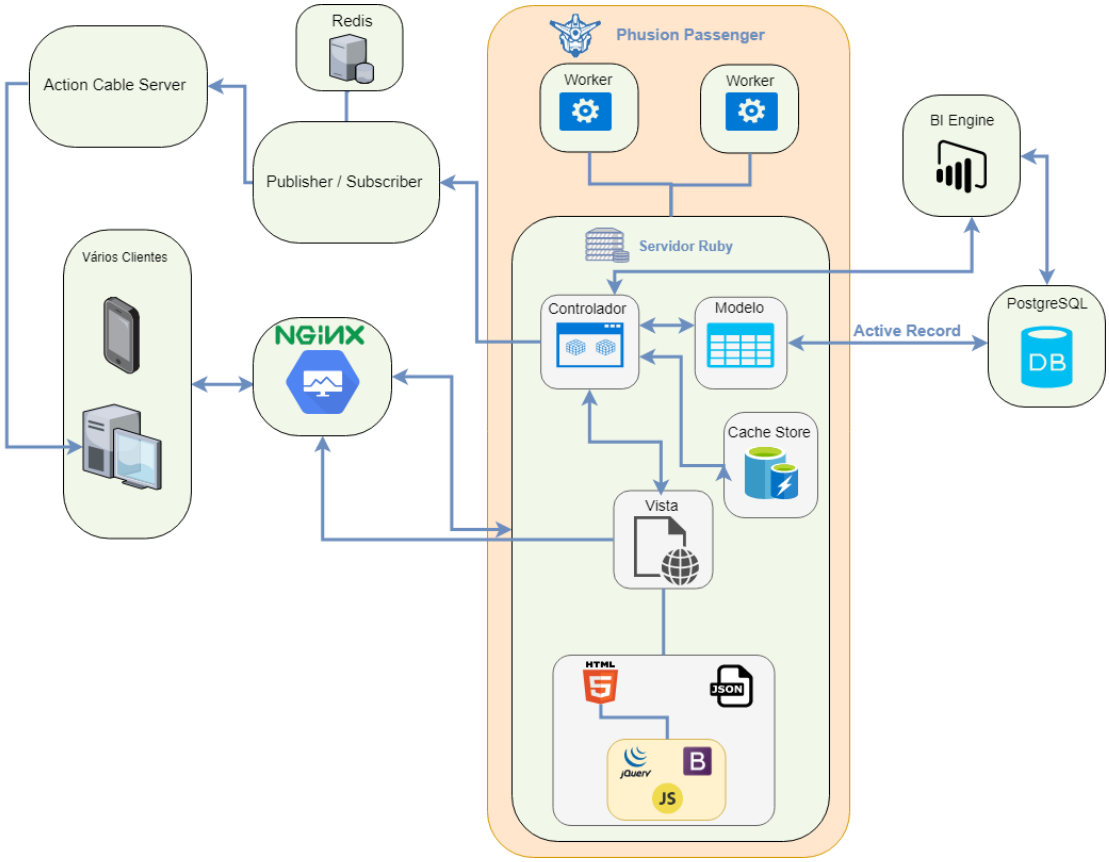
Fonte: MovTour

A primeira componente é a utilização de *beacons*. Estes são dispositivos *bluetooth* que estão distribuídos em determinados pontos de interesse de um monumento. O seu objetivo é avisar/informar o visitante que está perante um ponto de interesse. Assim que o utilizador recebe a notificação de que está perto do ponto de interesse é enviada informação não sensível sobre o dispositivo móvel para a terceira componente.

A segunda componente é uma aplicação móvel, desenvolvida numa tecnologia denominada de *react-native*. Esta permite disponibilizar informações sobre pontos de interesse e monumentos ao utilizador. Quando o visitante se encontra dentro do raio de um *beacon* é alertado, por via de uma notificação, que se encontra perto de um ponto de interesse ou monumento. Após este alerta, o utilizador é direcionado, dentro da aplicação, para a informação detalhada do local.

A última componente é o *backoffice*, implementado usando a linguagem *Ruby* e a *framework Rails* (*Ruby on Rails*). Além de outras funcionalidades, esta permite uma gestão de informação a nível dos pontos de interesse, monumentos e suas categorias. É possível efetuar a criação, remoção e edição tanto do local como das suas informações (endereço, coordenadas, texto descritivo, imagens, etc.) fornecendo-a em vários idiomas.

Figura 4: Arquitetura do backoffice

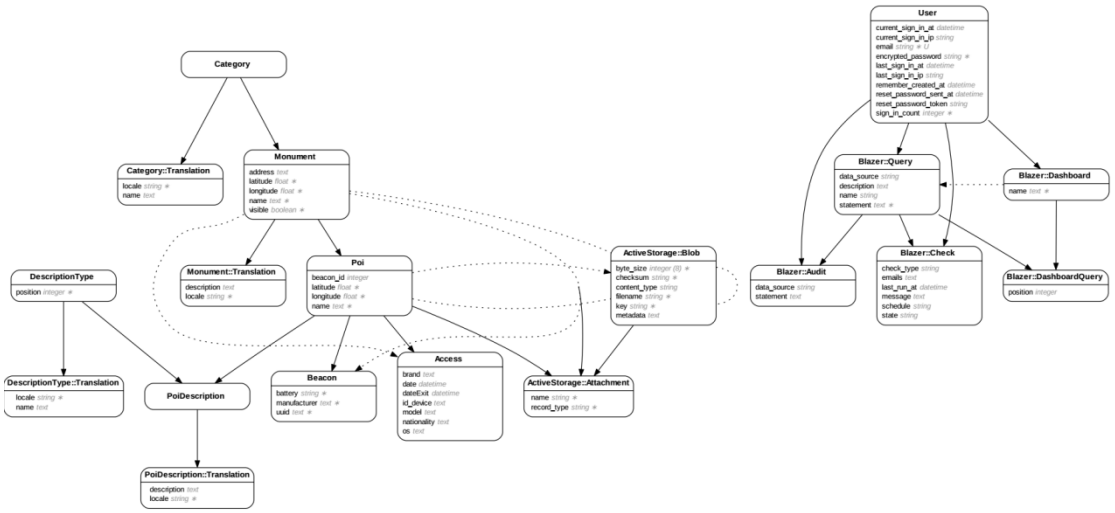


Fonte: Elaboração própria

O *backoffice* disponibiliza informação para que seja visualizada na aplicação móvel. A disponibilização é efetuada através de uma API que permite também o envio de informação para o *backoffice* sobre as visitas para registo de acessos.

É no *backoffice* que fica guardada toda a informação referente às visitas, guardando vários tipos de informações não sensíveis: o idioma, o modelo e o sistema operativo do dispositivo móvel, o identificador único do dispositivo, a data e hora tanto de entrada como de saída do raio do *beacon*.

Figura 5: Diagrama de Entidade e Relacionamento



Fonte: Ferramenta Rails

Foi implementado uma ferramenta de BI, que permite a criação de *dashboards*, gráficos e alertas à cerca de erros nos modelos de dados. Existe também a possibilidade de obter a informação dos visitantes em tempo real por via de uma tabela ou de um gráfico atualizado sempre que um visitante entra ou sai de um raio do *beacon*. Esta implementação foi graças à tecnologia de *websockets* que permite a comunicação bidirecional entre o *browser* e o servidor.

Foi também implementado um sistema de *caching* para garantir que o servidor não se torna lento com a gestão e as pesquisas efetuadas.

Capítulo 4 – Resultados

Tal como anteriormente descrito, uma solução de BI contempla duas grandes áreas: a recolha e a análise de grandes volumes de dados. Nas secções seguintes são abordados os mecanismos de otimização, os testes realizados, a solução de análise proposta, a visualização de dados em tempo real e o mecanismo implementado para reprimir possíveis ataques DDoS.

4.1. Mecanismos de otimização

Para tornar o projeto o mais otimizado possível foram implementadas técnicas de melhoramento de desempenho como a implementação de *cache* fragmentada e *queries* simplificados para que as pesquisas sejam feitas mais rapidamente.

4.1.1. Implementação de *cache*

Um dos mecanismos de otimização que foi implementado foi a *cache* fragmentada. A *cache* é um mecanismo de *software* ou *hardware* que é usado para armazenar os dados temporariamente. Este mecanismo, é mais rápido, mas também mais dispendioso em comparação com memórias tradicionais. A *cache* é utilizada porque o armazenamento em massa não consegue acompanhar os pedidos dos clientes.

A utilização da *cache* permite a redução dos tempos de acesso aos dados, logo o armazenamento em *cache* melhora o desempenho da operação.

Quando uma operação necessita de aceder a determinados dados, primeiro procura na *cache*. Se a informação desejada for encontrada (chamado de *Hit*), é devolvida imediatamente. Caso a informação desejada não seja encontrada (denominado de *Miss*), então é copiada da memória principal para a *cache*. Os *browsers* de *internet* utilizam a denominada "*browser cache*" para melhorar o desempenho das páginas *web* acedidas. Quando uma página é visitada, esta fica guardada na *cache* do *browser* para que na próxima vez o carregamento da página ser mais rápido.

O armazenamento em *cache* ajuda a tornar as aplicações mais rápidas e mais eficientes, pois os dados são armazenados o mais localmente possível. Para *websites* ou *browsers* os dados em *cache* carregam mais rápido, uma vez que podem aceder aos dados mais rapidamente a partir de uma pasta local. As imagens, por exemplo, podem ser relativamente grandes. Assim, o armazenamento em *cache* desses elementos significa que apenas é necessário fazer o *download* uma vez num determinado período.

Se um *website* demorar muito tempo a carregar é natural que o utilizador deixe de o utilizar. Usar o armazenamento em *cache* é uma medida importante para que isso não aconteça.

Embora o armazenamento em *cache* possa melhorar o desempenho da navegação nas páginas de *internet*, também pode deixá-lo vulnerável a ataques. As páginas de *internet* são armazenadas em *cache* no *browser*, o que significa que provavelmente contêm informações confidenciais e sensíveis. É possível que a exposição dos dados de autenticação, *tokens* de sessão ou informações confidenciais de clientes possa ocorrer.

Existe também uma questão importante, a privacidade. Um utilizador pode ter acesso ao histórico a partir da *cache* do *browser*, de outro utilizador.

A implementação de *cache* fragmentada no projeto deveu-se ao facto de quando o utilizador visitava o *backoffice* demorava muito tempo a disponibilizar a informação (texto, imagens, ...), o que tornava a experiência do utilizador bastante depreciativa. A implementação da *cache* pode ser dividida em dois grupos: a implementação no *backoffice* e a implementação na API.

No *backoffice*, onde fazia sentido implementar a *cache* seria quando o servidor teria de carregar listas ou várias informações numa única página. Desta forma, foi implementado na lista de monumentos, na visualização da informação de um monumento (pode conter vários pontos de interesse afetos), na visualização individual do ponto de interesse e na lista de *beacons*.

A nível da API, foi a disponibilização do ficheiro JSON (que contem toda a informação dos pontos de interesse, monumentos, etc.) que sofreu otimização. Este ficheiro é enviado para o dispositivo móvel quando a aplicação é instalada no mesmo.

4.2. Testes ao Sistema

Existem dois tipos de testes: os testes funcionais e os testes não funcionais.

Os testes funcionais avaliam o comportamento do sistema. Este tipo de testes tem como objetivo uma comparação no final da sua execução. Isto quer dizer que com é possível verificar se uma determinada ação tem como *output* o que é esperado.

Os testes não funcionais incidem essencialmente sobre o desempenho e usabilidade. Este tipo de testes tem como objetivo medir os tempos de resposta, a escalabilidade do sistema e verificar quando o sistema é comprometido.

4.2.1. Testes funcionais

Existem dois grupos dentro dos testes funcionas: os testes estáticos e os dinâmicos.

Os testes estáticos incidem diretamente com o código do sistema. Os componentes do sistema são verificados sem que o código seja executado, seja de forma manual ou automática. O objetivo principal é identificar erros de programação.

Neste âmbito, e para tornar o sistema o mais automatizado possível, foram utilizadas várias ferramentas para a execução deste tipo de testes:

- *CodeClimate* – Ferramenta que analisa a manutenção, a cobertura de teste e se existe código duplicado do sistema. O seu objetivo é correr o código todo do sistema e mediante algumas métricas, atribuir determinada avaliação;
- *Snyk* – Esta ferramenta disponibiliza informações a cerca de falhas de segurança e vulnerabilidades do sistema;
- Hakiri – Esta ferramenta informa-nos sobre novas vulnerabilidades do sistema.

Para que fosse possível o controlo e a revisão de código, foram utilizadas duas ferramentas: o *Git* e o *GitHub*. O *Git* é um sistema de controlo de versões. Através dele é possível desenvolver projetos onde diversas pessoas podem contribuir simultaneamente para o mesmo. O *GitHub* é um serviço *web* que oferece diversas funcionalidades extras aplicadas ao *Git*. Com ela é possível ter uma equipa a trabalhar sobre o mesmo projeto.

Figura 6: Repositório do projeto no GitHub

Fonte: Repositório GitHub

As alterações são efetuadas num repositório local e, quando necessário, podem ser enviadas para o serviço na *web* que irá atualizar o repositório partilhado. A qualquer momento é possível reverter o projeto dependendo dos envios para o repositório partilhado. Todas estas ferramentas estão representadas no repositório do projeto no *Github*.

Os testes dinâmicos normalmente são utilizados/efetuados para complementação dos testes estáticos. Estes testes necessitam de valores de *input* para assim facultar o seu *output*.

O *Minitest* é uma ferramenta de teste para o *Ruby* que fornece um conjunto completo de instalações de teste. A ferramenta foi adicionada à biblioteca padrão do *Ruby*, que aumentou a sua popularidade. Mesmo que no início o *Minitest* dê a impressão de uma ferramenta de teste muito pequena, este tem um conjunto muito rico de recursos que o tornam numa ferramenta poderosa para se ter em mãos. É com esta ferramenta que é possível codificar os tipos de teste que queremos executar.

Figura 7: Percentagem de cobertura de testes

```
Finished in 13.754519s, 2.6173 runs/s, 3.5625 assertions/s.
36 runs, 49 assertions, 0 failures, 0 errors, 0 skips
Coverage report generated for MiniTest to /vagrant/coverage. 575 / 699 LOC (82.26%) covered.
```

Fonte: Vagrant

No caso deste projeto, foram efetuados vários destes tipos de testes a quase todos os controladores, cobrindo cerca de 80% das funcionalidades do projeto.

Figura 8: Código de teste do monumento

```
test "should create monument" do
  sign_in users(:one)
  assert_difference('Monument.count') do
    post monuments_url("pt"), params: {
      monument: {
        address: @monument.address,
        image_file_name: @monument.image_file_name,
        image_content_type: @monument.image_content_type,
        image_file_size: @monument.image_file_size,
        image_updated_at: @monument.image_updated_at,
        latitude: @monument.latitude,
        longitude: @monument.longitude,
        name: @monument.name,
        visible: @monument.visible,
        category: @monument.category,
        description_pt: @monument.description,
      }
    }
  end
  assert_redirected_to monument_url(Monument.last)
end
```

Fonte: Elaboração própria

Um dos testes criados serviu para verificar se era possível a criação de um monumento com determinados parâmetros. Na figura anterior, é demonstrado um exemplo em código para saber se é possível criar um monumento com os parâmetros.

Para conseguir efetuar testes, é necessário ter dados. Esta ferramenta (*minitest*) contem um local próprio para a criação de dados de testes. Este local é denominado de *fixtures*.

Figura 9: Fixture de monumento

```
one_mon:
  name: Convento de Cristo
  # description: "O Castelo Templário/Convento de Cristo foi sede da Ordem do Templo, até 1314, e da Ordem de Cristo, a partir de 1357
  longitude: -8.41902494
  latitude: 39.60349765
  address: Rua do Convento
  category: one_cat
  image_file_name: temp_file.jpg
  image_content_type: image/jpeg
  image_file_size: 223312
  image_updated_at: 2015-02-29 10:30:19 Z
  visible: true
```

Fonte: Elaboração própria

As *fixtures* são uma forma de organizar os dados que se pretende testar, definindo atributos que irão ser testados quando os testes foram criados. Executar estes testes manualmente sempre que haja uma modificação no código não é prático. Assim, foi utilizada uma ferramenta chamada *travis*.

Figura 10: Ficheiro travis.yml

```
env:
  global:
    - DB=postgresql
    - CC_TEST_REPORTER_ID=191bea95d04b78e28e30235f83828eb583d7dfc736d97aa1a5e4581d40825894
    - GIT_COMMITTED_AT=$(if [ "$TRAVIS_PULL_REQUEST" == "false" ]; then git log -1 --pretty=format:%ct; else git log -1 --skip 1
language: ruby
rvm:
  - "2.4.0"
services:
  - postgresql
script:
  - RAILS_ENV=test bundle exec rake db:migrate --trace
  - bundle exec rake db:test:prepare
  - bundle exec rake test test/
  - if [ "$TRAVIS_PULL_REQUEST" == "false" ]; then ./cc-test-reporter after-build --exit-code $TRAVIS_TEST_RESULT; fi
before_script:
  - psql -c 'create database travis_ci_test;' -U postgres
  - cp config/database.yml.travis config/database.yml
  - curl -L https://codeclimate.com/downloads/test-reporter/test-reporter-latest-linux-amd64 > ./cc-test-reporter
  - chmod +x ./cc-test-reporter
bundler_args: --binstubs=./bundler_stubs
notifications:
  slack: movtour:KWhf3Ah0baQLUxy9Zopi3D5A
```

Fonte: Elaboração própria

Esta é uma ferramenta de integração continua, usada para testar projetos hospedados no *GitHub*. Um ficheiro (*travis.yml*) é parametrizado e executado a cada nova iteração (*commit*) do *GitHub*. É neste ficheiro que é configurado todas as instruções que se deseja que a ferramenta execute, como por exemplo criar a base de dados e executar os testes.

A cada nova versão da aplicação enviada para o *GitHub*, são executadas todas as instruções configuradas no ficheiro, informando no final se a versão do projeto está sem erros.

4.2.2. Testes não funcionais

No ramo dos testes não funcionais foram executados testes de carga e stress.

Os testes de carga são realizados para verificar o volume de dados que um sistema consegue processar, ou seja, permite monitorizar o comportamento do sistema diante de uma situação onde há um grande número de acessos simultâneos. Este teste permite avaliar a atuação do *software* frente a um fluxo intenso de dados, avaliar a estabilidade do sistema num período de grande carga para estabelecer a sua margem de operação com segurança, identificar os pontos críticos que são de grande impacto quando o sistema está submetido a grande carga e monitorizar o desempenho do sistema frente a um aumento significativo de dados processados simultaneamente.

Este teste é também utilizado para verificar o desempenho do *software* num cenário onde existe um grande volume de utilizadores com acessos em simultâneo ao sistema, num horário de pico, por exemplo.

Os testes de stress são utilizados para verificar o limite máximo do sistema submetido a um grande volume de acessos em simultâneo. Portanto, visa identificar o ponto em que o sistema deixa de atender ao mínimo especificado. A execução deste tipo de testes é vantajosa, pois fornece informações que permitem evoluir a arquitetura do *software*. Os indícios de que o sistema não atende mais ao esperado podem manifestar-se de diversas formas, tais como: os dados deixarem de ser guardados ou serem corrompidos; alguns recursos do sistema, quando este regressa ao normal, permanecem danificados; o sistema não responde, trava ou deixa de funcionar definitivamente.

Para ser possível verificar se a implementação de *cache* no projeto foi uma mais valia, foi utilizado uma ferramenta denominada *Postman*. Este é um poderoso cliente *Hyper Text Transfer Protocol* (HTTP) com o objetivo de testar serviços *web*. Este facilita o teste, o desenvolvimento e a documentação de API's, permitindo que os utilizadores executem rapidamente solicitações HTTP simples e complexas. Através desta ferramenta, foi possível medir os tempos de carregamento tanto da parte do servidor como da parte da API.

Tabela 2: Tempos de resposta do backoffice

Página	Vista	Tempo (s)	*Cache*
Monumentos	Index	0,304	Não
Monumentos	Index	0,205	Sim
Monumentos	Show	0,230	Não
Monumentos	Show	0,093	Sim
Ponto de Interesse	Show	0,145	Não
Ponto de Interesse	Show	0,096	Sim
Beacons	Show	0,082	Não
Beacons	Show	0,072	Sim

Fonte: Elaboração própria

A tabela anterior mostra os tempos de carregamento com e sem *cache* de algumas páginas do lado do *backoffice*. Para estes tempos não estão comtemplados o *download* das imagens, apenas o carregamento da informação.

Tabela 3: Tempos de transferência do ficheiro JSON

Lista	Tempo (s)	*Cache*
Monumentos	0,588	Não
Monumentos	0,118	Sim
Beacons	0,117	Não
Beacons	0,046	Sim

Fonte: Elaboração própria

A tabela supra mostra os tempos que o servidor demorou a disponibilizar o ficheiro *JSON* para a aplicação móvel. Foram medidos para a lista de monumentos e para a lista de *beacons*.

Foi utilizado também uma ferramenta, denominada de *Vegeta* que permite a execução de testes parametrizados. Esta é uma ferramenta versátil de teste de carga HTTP criada a partir da necessidade de detalhar serviços HTTP com uma taxa de solicitação constante. Os testes foram efetuados em dois momentos diferentes, antes e depois da implementação de *cache* e mecanismos de otimização. Esta ferramenta foi utilizada para medir várias métricas nos métodos do sistema (GET e POST). O método GET é executado quando o utilizador abre pela primeira vez a aplicação móvel. O que este método faz é descarregar para o dispositivo móvel a lista completa de monumentos e seus dependentes (categorias, monumentos, pontos de interesse, etc.). O método POST é executado quando o utilizador entra dentro do raio de um *beacon*, onde é enviada informação não sensível para o servidor.

Para que os testes fossem otimizados, foi criado um *scrypt* na linguagem *Ruby*, de forma que os testes fossem executados com taxas e durações diferentes. Foi testada a capacidade de resposta que o servidor tinha em disponibilizar o ficheiro JSON para a aplicação. Este ficheiro (*monuments.json*) é descarregado após a primeira iniciação da aplicação no dispositivo móvel.

Tabela 4: Tempos de download do ficheiro JSON (duração 10s)

Taxa de Pedidos	Duração (s)	Total de Pedidos	Tempo de resposta médio (s)	Tempo de resposta 50% (s)	Tempo de resposta 95% (s)	Tempo de resposta máximo (s)	Rácio de Sucesso	*Cache*
5	10	50	3,120	3,210	4,520	4,760	100%	Não
10	10	100	6,900	7,110	11,400	11,700	100%	Não
15	10	150	10,800	10,830	19,470	20,100	100%	Não
18	10	180	10,150	10,100	21,300	21,700	84%	Não
20	10	200	9,180	8,550	21,300	21,960	75%	Não
5	10	50	0,060	0,059	0,0650	0,106	100%	Sim
10	10	100	0,050	0,045	0,0610	0,232	100%	Sim
15	10	150	0,050	0,045	0,0610	0,104	100%	Sim
18	10	180	0,054	0,056	0,0740	0,168	100%	Sim
20	10	200	0,048	0,043	0,0620	0,222	100%	Sim

Fonte: Elaboração própria

Este teste foi reproduzido tanto para cenários prováveis como para cenários improváveis. Este foi efetuado com 5, 10, 15, 18 e 20 pedidos por segundo durante 10 segundos.

Observa-se que, sem *cache*, durante 10 segundos, se consegui-o ter uma taxa de sucesso de 100% para 5, 10 e 15 pedidos por segundo, mas observando os tempos de resposta, é possível observar que apenas os 5 pedidos por segundo têm um tempo de resposta minimamente aceitável (aproximadamente 3 segundos). Isto quer dizer que se 5 utilizadores abrirem a aplicação ao mesmo tempo, o tempo que o utilizador está á espera para ver alguma informação é de aproximadamente 3 segundos, assim que abre a aplicação móvel pela primeira vez.

Assim que a *cache* foi implementada, os tempos de resposta foram bastante melhorados tendo sido obtido uma taxa de sucesso de 100% para todos os tipos de pedidos, com médias entre os 40 e os 60ms de tempo de resposta.

Tabela 5: Tempos de download do ficheiro JSON (duração 30s)

Taxa de Pedidos	Duração (s)	Total de Pedidos	Tempo de resposta médio (s)	Tempo de resposta 50% (s)	Tempo de resposta 95% (s)	Tempo de resposta máximo (s)	Rácio de Sucesso	*Cache*
5	30	150	5,000	5,700	6,200	6,500	100%	Não
10	30	300	10,900	11,000	21,500	21,700	84%	Não
15	30	450	8,500	4,300	21,700	23,300	55%	Não
18	30	540	7,340	0,049	22,000	22,400	46%	Não
20	30	600	6,650	0,047	21,900	22,430	41%	Não
5	30	150	0,055	0,056	0,062	0,177	100%	Sim
10	30	300	0,046	0,043	0,061	0,106	100%	Sim
15	30	450	0,047	0,043	0,061	0,107	100%	Sim
18	30	540	0,047	0,043	0,063	0,229	100%	Sim
20	30	600	0,060	0,054	0,144	0,304	100%	Sim

Fonte: Elaboração própria

Para uma duração maior (30 segundos) efetuou-se um teste com as mesmas variantes.

Para o teste sem *cache* apenas o primeiro teste obteve 100% de sucesso, considerando também que no total foram feitos 150 pedidos em 30 segundos.

Verifica-se que para 18 e 20 pedidos por segundo, para uma resposta de 50% dos pedidos, o tempo foi extremamente baixo face às outras métricas. Isto deve-se ao facto de como o servidor já não tem capacidade de responder a todos os pedidos, existem pedidos que são perdidos, o número de respostas é inferior ao das outras métricas e assim o cálculo dos tempos vai ser diferente.

Mais uma vez, o teste com a *cache* revela que tanto os tempos como a taxa de sucesso foram melhorados a um nível surpreendente.

Tabela 6: Tempos de download do ficheiro JSON (duração 60s)

Taxa de Pedidos	Duração (s)	Total de Pedidos	Tempo de resposta médio (s)	Tempo de resposta 50% (s)	Tempo de resposta 95% (s)	Tempo de resposta máximo (s)	Rácio de Sucesso	*Cache*
5	60	300	5,630	6,000	6,500	7,800	100%	Não
10	60	600	10,700	10,900	21,500	21,700	66%	Não
15	60	900	7,700	49,000	22,000	22,400	43%	Não
18	60	1080	6,600	46,000	21,900	22,500	36%	Não
20	60	1200	5,900	46,600	21,800	22,400	32%	Não
5	60	300	0,059	0,058	0,063	0,135	100%	Sim
10	60	600	0,047	0,044	0,060	0,105	100%	Sim
15	60	900	0,047	0,043	0,062	0,234	100%	Sim
18	60	1080	0,045	0,042	0,059	0,106	100%	Sim
20	60	1200	0,046	0,042	0,059	0,104	100%	Sim

Fonte: Elaboração própria

Neste terceiro teste, foi aumentada a duração (de 30 para 60 segundos).

Neste, apenas a taxa de 5 pedidos por segundo, sem *cache*, consegue obter uma taxa de 100% de sucesso. Em contrário, quando aplicado a *cache*, observa-se uma melhoria significativa dos tempos de resposta. Com a *cache* implementada, é possível ver que os tempos de resposta são bastante próximos para diversas métricas.

Quando o utilizador entra no raio de um *beacon*, a aplicação móvel insere no servidor (por meio da API através do método POST) um novo registo com várias informações. Para esta ação, de guardar informação no servidor, foram efetuados 3 testes. A taxa para a qual foram efetuados os testes foi de 5, 10 e 25 pedidos por segundo durante 10, 30, 60 e, para verificarmos a viabilidade do servidor, 100 segundos.

Tabela 7: Tempos de inserção de dados através da API no backoffice (10s)

Taxa de Pedidos	Duração (s)	Total de Pedidos	Tempo de resposta médio (s)	Tempo de resposta 50% (s)	Tempo de resposta 95% (s)	Tempo de resposta máximo (s)	Rácio de Sucesso
5	10	50	0,029	0,028	0,040	0,057	100%
10	10	100	0,019	0,017	0,037	0,041	100%
25	10	250	0,026	0,025	0,031	0,056	100%
50	10	500	0,027	0,025	0,032	0,146	100%
100	10	1000	0,026	0,023	0,042	0,176	100%

Fonte: Elaboração própria

Através da tabela 7 observar-se que a média do tempo de resposta do servidor está entre os 20 e os 30ms. Para as métricas testadas, obteve-se um rácio de sucesso de 100%.

Tabela 8: Tempos de inserção de dados através da API no backoffice (30s)

Taxa de Pedidos	Duração (s)	Total de Pedidos	Tempo de resposta médio (s)	Tempo de resposta 50% (s)	Tempo de resposta 95% (s)	Tempo de resposta máximo (s)	Rácio de Sucesso
5	30	150	0,026	0,025	0,032	0,050	100%
10	30	300	0,030	0,027	0,040	0,142	100%
25	30	750	0,026	0,025	0,030	0,052	100%
50	30	1500	0,025	0,025	0,030	0,060	100%
100	30	1000	0,023	0,023	0,027	0,028	100%

Fonte: Elaboração própria

Tabela 9: Tempos de inserção de dados através da API no backoffice (60s)

Taxa de Pedidos	Duração (s)	Total de Pedidos	Tempo de resposta médio (s)	Tempo de resposta 50% (s)	Tempo de resposta 95% (s)	Tempo de resposta máximo (s)	Rácio de Sucesso
5	60	300	0,026	0,025	0,032	0,063	100%
10	60	600	0,030	0,027	0,040	0,142	100%
25	60	1500	0,030	0,027	0,047	0,174	100%
50	60	3000	0,030	0,027	0,050	0,193	100%
100	60	6000	0,025	0,023	0,051	0,236	100%

Fonte: Elaboração própria

Quando aumentado o tempo de teste para 30 segundos – tabela 8 – e 60 segundos – tabela 9, é possível observar que tanto os tempos médios de resposta como o rácio de sucesso se mantém praticamente os mesmos.

Através destas tabelas, é possível afirmar que o servidor suporta várias inserções na base de dados em simultâneo. Como esta ação de inserção da base de dados e os tempos são bastante rápidos, não foram implementados nenhuns mecanismos de otimização.

Quando o utilizador acede ao *backoffice*, é carregada a lista de monumentos. Esta lista contém inúmeras informações que podem tornar o seu carregamento lento. Para testar esta lacuna, foram medidos 5, 10, 15, 18 e 20 acessos por segundo, durante 10, 30 e 60 segundos. Os 5 acessos por segundo quer dizer que 5 pessoas acederam ao *backoffice* e pediram ao mesmo tempo a lista de monumentos.

Tabela 10: Tempo de disponibilização da lista de monumentos (10s)

Taxa de Pedidos	Duração (s)	Total de Pedidos	Tempo de resposta médio (s)	Tempo de resposta 50% (s)	Tempo de resposta 95% (s)	Tempo de resposta máximo (s)	Rácio de Sucesso	*Cache*
5	10	50	0,217	0,221	0,243	0,269	100%	Sim
10	10	100	0,228	0,223	0,279	0,345	100%	Sim
15	10	150	0,861	0,870	1,200	1,300	100%	Sim
18	10	180	2,000	2,100	3,430	3,500	100%	Sim
20	10	200	2,600	2,700	4,700	4,900	100%	Sim
5	10	50	0,260	0,250	0,305	0,364	100%	Não
10	10	100	0,395	0,305	0,764	1,150	100%	Não
15	10	150	1,800	1,770	2,940	3,100	100%	Não
18	10	180	3,100	3,300	5,500	5,700	100%	Não
20	10	200	4,000	4,100	3,100	7,550	100%	Não

Fonte: Elaboração própria

A tabela acima compara as métricas anteriormente ditas, com e sem *cache* implementadas. Pode-se observar que com a *cache* os tempos de resposta melhoram bastante face aos tempos de resposta sem *cache*.

Tabela 11: Tempo de disponibilização da lista de monumentos (30s)

Taxa de Pedidos	Duração (s)	Total de Pedidos	Tempo de resposta médio (s)	Tempo de resposta 50% (s)	Tempo de resposta 95% (s)	Tempo de resposta máximo (s)	Rácio de Sucesso	*Cache*
5	30	150	0,217	0,213	0,236	0,294	100%	Sim
10	30	100	0,227	0,220	0,278	0,343	100%	Sim
15	30	250	1,700	1,740	2,880	3,100	100%	Sim
18	30	540	4,300	5,600	7,500	7,900	95%	Sim
20	30	600	4,600	5,300	7,700	8,000	86%	Sim
5	30	150	0,226	0,251	0,300	0,370	100%	Não
10	30	100	0,250	0,255	0,327	0,565	100%	Não
15	30	250	4,700	4,700	8,000	9,000	100%	Não
18	30	540	5,200	5,800	9,100	9,200	83%	Não
20	30	600	5,300	5,800	9,100	9,400	75%	Não

Fonte: Elaboração própria

Aumentando a duração dos pedidos para 30 segundos, constata-se que tanto para a taxa de 18 como para a taxa de 20 pedidos por segundo, com e sem *cache*, o rácio de sucesso não é de 100%, apesar de que com *cache* os tempos de resposta são melhores.

Tabela 12: Tempo de disponibilização da lista de monumentos (60s)

Taxa de Pedidos	Duração (s)	Total de Pedidos	Tempo de resposta médio (s)	Tempo de resposta 50% (s)	Tempo de resposta 95% (s)	Tempo de resposta máximo (s)	Rácio de Sucesso	*Cache*
5	60	300	0,217	0,211	0,240	0,300	100%	Sim
10	60	600	0,229	0,223	0,283	0,581	100%	Sim
15	60	900	3,000	3,100	5,400	5,700	100%	Sim
18	60	1080	5,100	7,000	7,700	8,000	86%	Sim
20	60	1200	5,300	7,400	7,600	7,900	78%	Sim
5	60	300	0,257	0,251	0,291	0,391	100%	Não
10	60	600	0,259	0,255	0,310	0,450	100%	Não
15	60	900	5,800	7,100	9,100	9,400	88%	Não
18	60	1080	5,900	8,200	9,100	9,350	74%	Não
20	60	1200	6,100	7,550	9,100	9,400	66%	Não

Fonte: Elaboração própria

Por fim, foi aumentada a duração para 60 segundos. Observar-se que o rácio de sucesso se manteve em 100% para a taxa de 5, 10 e 15 pedidos com *cache* e 5 e 10 para pedidos sem *cache*.

Com o registo destes tempos, observa-se que a implementação de mecanismos de otimização é uma mais valia para o projeto, pois aumenta o desempenho, estabilidade e capacidade de resposta do servidor.

4.3. *Dashboards*

Independentemente do projeto, existem inúmeros dados que são armazenados. O problema é que dados armazenados não tratados, não têm propósito. Não é fácil ver tendências e fazer suposições com base na exibição de uma tabela que contem dados brutos.

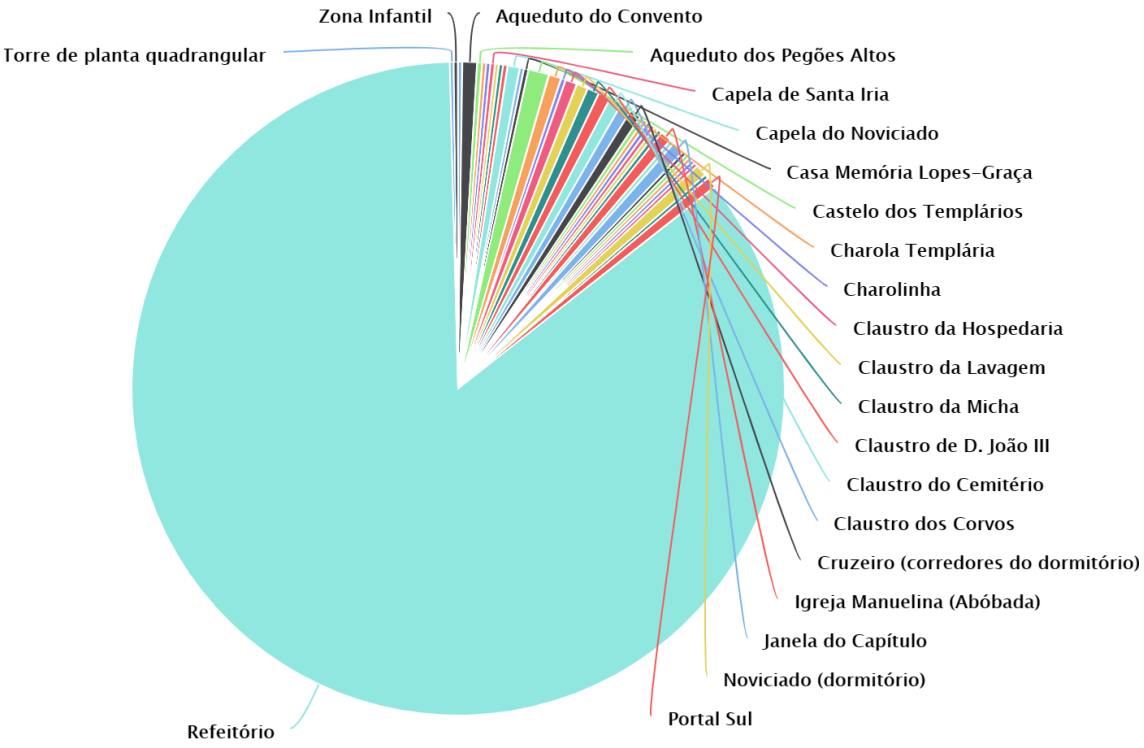
Com a quantidade enorme de dados que é possível armazenar, é necessário utilizar uma ou mais ferramentas para os ajudar a estruturar estes dados, de forma a ser possível ter uma visão mais clara. Para isso é utilizado os denominados *dashboards*.

A definição mais precisa, de *dashboards*, seria a de Stephen Few, fundador da "Perceptual Edge" e renomeado consultor e educador nas áreas de inteligência de negócios e design de informações: "um *dashboard* é a apresentação visual das informações mais importantes e necessárias para alcançar um ou mais objetivos de negócio, consolidadas e ajustadas numa única tela para que a informação possa ser monitorizada de forma ágil" [27].

Assim, um *dashboard* é um painel visual que apresenta, de maneira centralizada, um conjunto de informações (indicadores e suas métricas). Este recurso também pode auxiliar no apoio à decisão. A sua utilização é uma forma interessante de dinamizar o trabalho de gestão e o seu principal objetivo, enquanto painel de informações é facilitar o acompanhamento eficiente das operações.

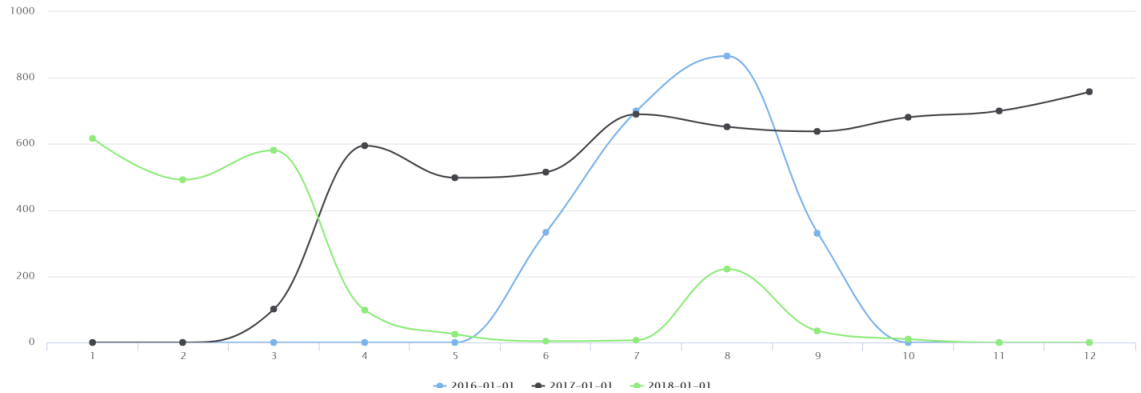
A primeira abordagem foi a criação de gráficos estáticos. Estes gráficos foram codificados diretamente no código, impossibilitando assim o utilizador a visualizar outros dados ou outros tipos de gráficos.

Figura 11: Visitas por ponto de interesse



Fonte: Elaboração própria

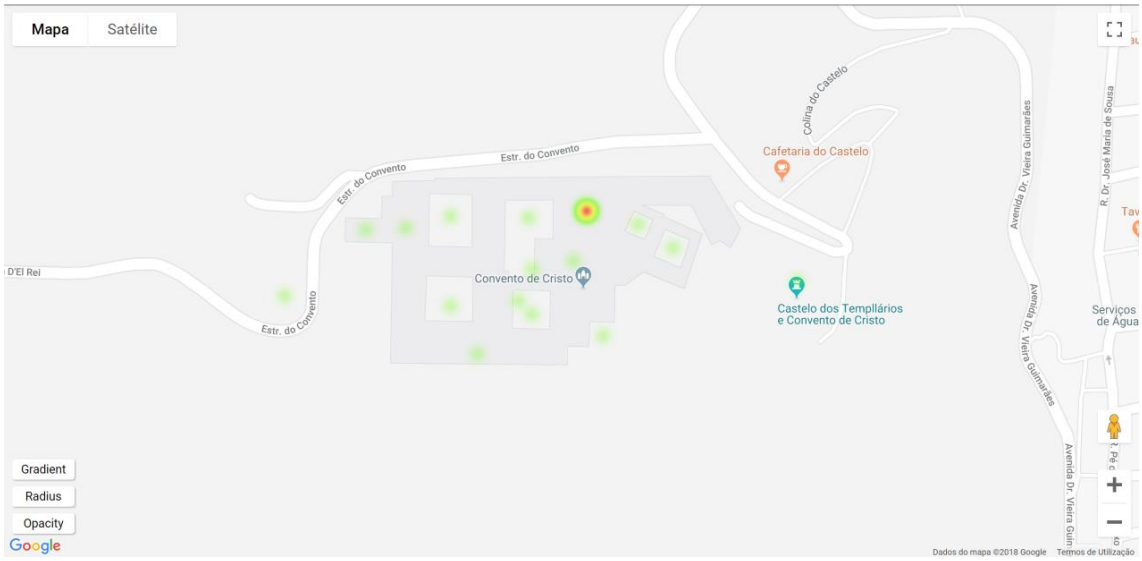
Figura 12: Visitantes por mês e por ano



Fonte: Elaboração própria

Foram utilizadas bibliotecas *JavaScript* para criação de gráficos circulares que mostram as visitas por pontos de interesse, gráficos de linhas que mostram o número de visitantes por mês e por ano, um *heatmap* (gráfico de dispersão) com as visitas representadas, entre outros.

Figura 13: Heatmap



Fonte: API Google

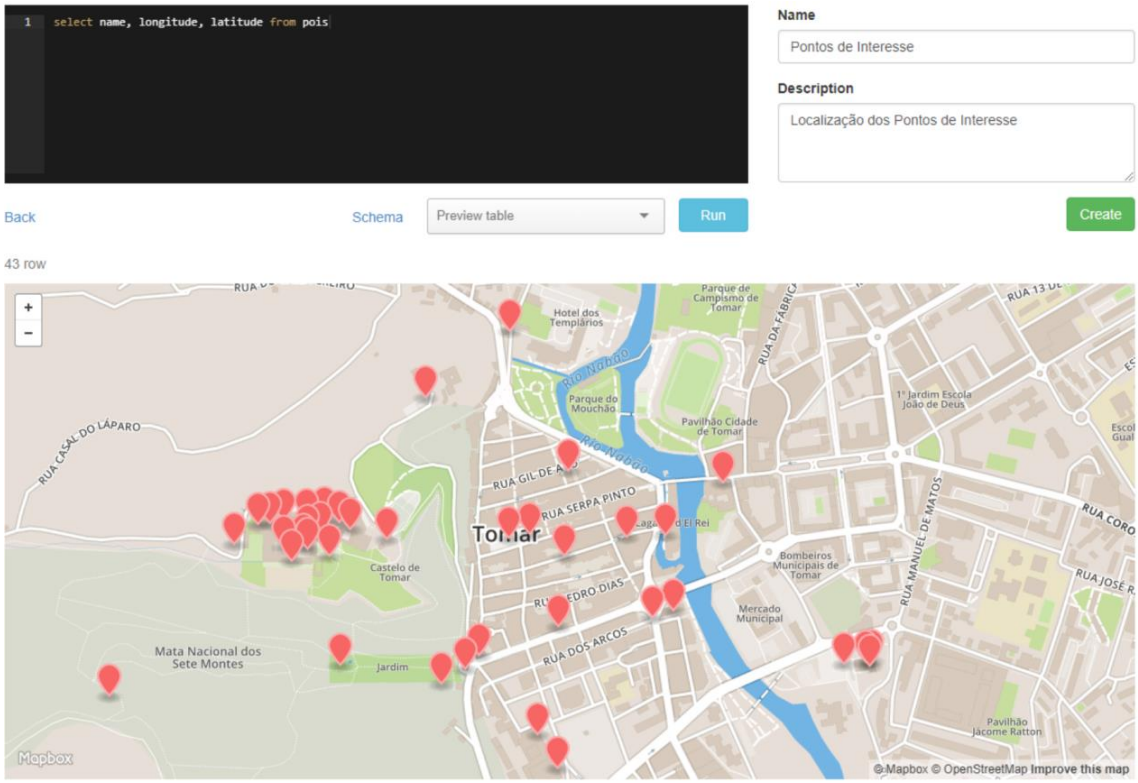
Na imagem anterior é possível observar o mapa que foi criado com a dispersão das visitas. Quanto mais vermelho mais visitas ocorrem no ponto.

O problema destes gráficos, é que são *hardcoded*. Isto que dizer que são escritos diretamente no código do sistema. Esta não é uma abordagem particularmente correta porque se for necessário ver outros tipos de gráficos, é necessário que estes sejam

programados diretamente no código. Foram, então, utilizadas duas abordagens para melhorar a parte dos *dashboards* e para assim torná-los mais dinâmicos.

A primeira abordagem foi a utilização de um sistema dinâmico de exploração e consulta de dados via *Blazer*.

Figura 14: Localização dos pontos de interesse



name	longitude	latitude
Castelo dos Templários	-8.41765165	39.60355138
Janela do Capítulo	-8.41936558	39.60360511

Fonte: API Mapbox

Este sistema permite a criação de gráficos por meio de consultas em SQL, criação de *dashboard* que alojam os vários gráficos criados, e permite também enviar *e-mails* caso exista alguma anomalia nos dados. Esta ferramenta tem uma interface em inglês e requer comandos SQL para a criação dos gráficos.

É possível criar vários tipos de gráficos e adicioná-los a diferentes *dashboards*. Como os gráficos são criados na *interface* do *browser*, é possível gerar vários tipos de gráficos, com informações variáveis, o que na solução anterior era impossível. Os gráficos que são possíveis de criar são: gráficos de barras, linhas, dispersão e mapas.

A segunda abordagem foi a implementação de um live *dashboard*. Para isso foram implementados *websockets*. O protocolo HTTP é o que guia atualmente a comunicação entre *browsers* e servidores nas principais aplicações de *internet*. A questão é que o protocolo HTTP exige que uma requisição seja executada da parte do *browser* para que então esta possa ter uma resposta do servidor. Desta maneira, assim que o servidor envia uma resposta, uma nova mensagem só pode ser enviada após uma nova requisição por parte do *browser*. Este modelo funciona bem para a maioria dos *websites* que habitualmente são utilizados. Requisitados os conteúdos, como por exemplo textos e imagens, são recebidos assim que possível levando-se em conta o tempo de latência da conexão e a velocidade de *download*. O seu funcionamento é relativamente simples.

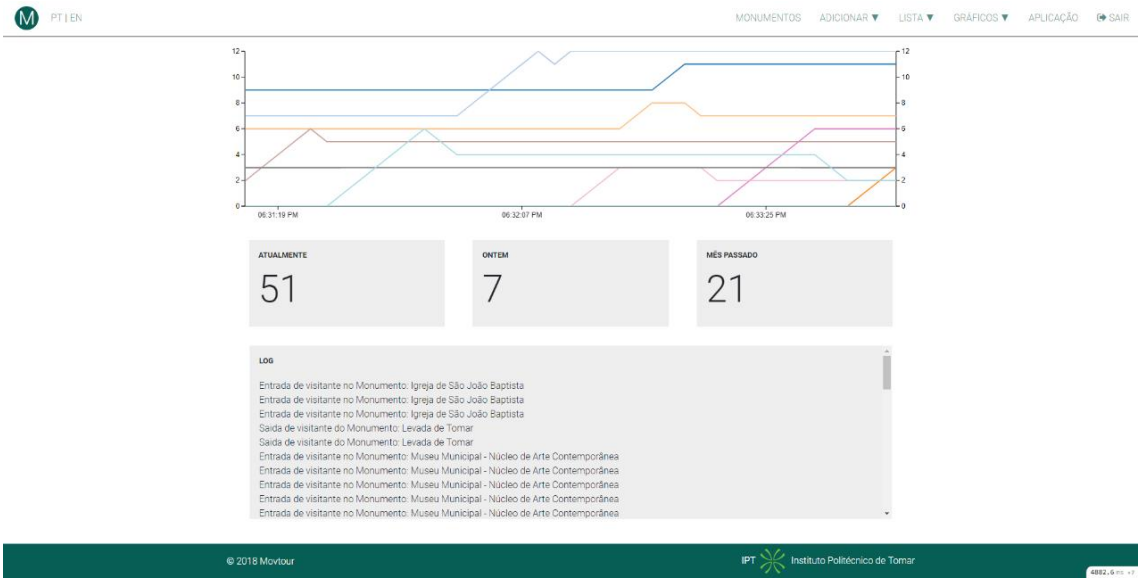
O cenário começa a mudar a partir do momento em que o servidor não tem disponível toda a informação que é útil, rapidamente. Diversas técnicas tentam mitigar este problema. A mais simples consiste no *browser* executar uma série de requisições com o intuito de verificar se o servidor possui uma nova informação disponível. Fazendo o uso da tecnologia de *websockets*, que é uma extensão do protocolo HTTP, um canal de comunicação é estabelecido entre *browser* e o servidor. Assim, a troca de mensagens não fica condicionada a uma requisição da parte do *browser*. Por outras palavras, durante o período de conexão, o servidor pode enviar quaisquer mensagens diretamente para o *browser* e vice-e-versa. Com isto, é possível saber sempre quantas pessoas se encontram por ponto de interesse ou por monumento no exato momento.

Figura 15: Nova visita no ponto de interesse

Fonte: Elaboração própria

Numa primeira fase, para que este conceito fosse testado, foram atualizadas as visitas totais da página de visitas por ponto de interesse. Quando o utilizador entra dentro do raio de um *beacon* a aplicação móvel envia registos (pelo método POST da API) para o servidor. Através dos canais de comunicação (*websockets*), obteve-se a informação do aumento ou diminuição do número de visitas totais e assim modificar a página incrementando ou decrementando as visitas totais do determinado ponto de interesse. Quando o número de visitas totais aumenta, o número pisca verde, quando diminui, o número pisca vermelho.

Figura 16: Live Dashboard



Fonte: Elaboração própria

Com esta tecnologia dos *websockets* a funcionar e após ter sido verificado que era uma mais valia, foi criado também um gráfico para demonstração do número de visitantes por monumento em tempo real. Nesta página em que é mostrado este gráfico em tempo real, também é possível obter-se a informação dos visitantes no dia anterior e no mês anterior à consulta atual. É possível ainda obter um histórico dos últimos monumentos visitados, sabendo se o visitante entrou ou saiu do monumento.

Sempre que o um visitante entra ou sai de dentro do raio de um *beacon* este gráfico é atualizado, permitindo assim, obter em direto a informação de quantas pessoas estão concretamente a visitar os monumentos.

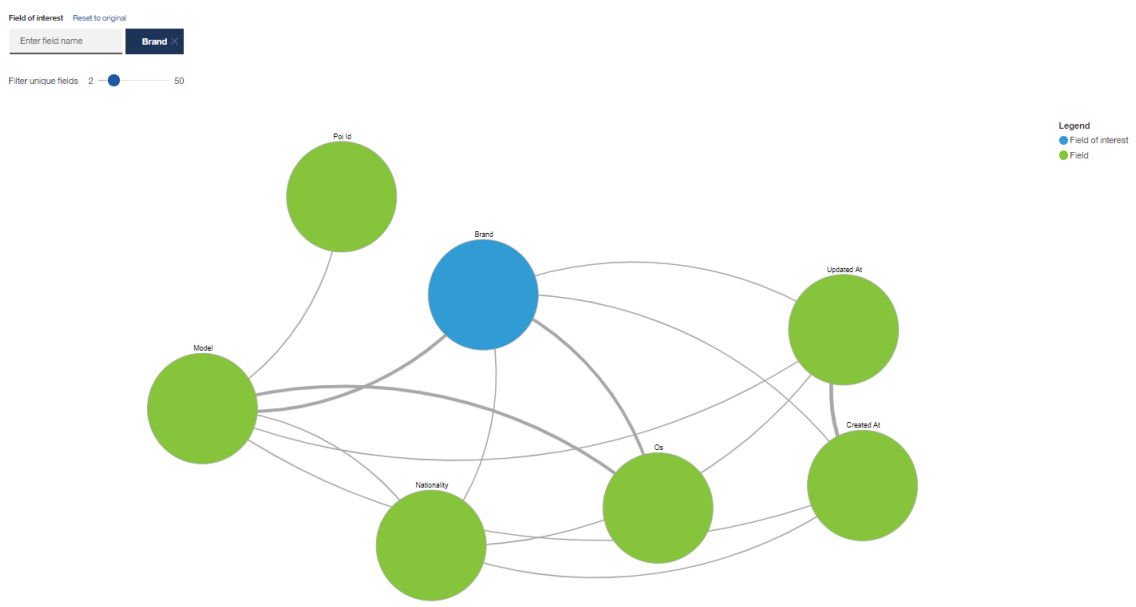
O facto de se ter um gráfico que diz em tempo real quantas pessoas estão a visitar os monumentos, ter uma lista em que é mostrado as visitas totais nos pontos de interesse e as visitas atuais dos mesmos possibilita que por exemplo o prazo de encerramento do mesmo seja alargado visto que ainda contem visitantes dentro dele. Este foi um de inúmeros exemplos de informações que foi possível retirar com estes mecanismos.

Para que toda esta tecnologia funcione, foi implementado um servidor (*Redis*). Este é um servidor específico de armazenamento de estruturas de dados na memória. Ele é usado como base de dados, *cache* e intermediário de mensagens. Este tipo de servidor armazena inúmeros e diferentes tipos de estruturas de dados.

4.4. Integração com *IBM Cognos Analytics*

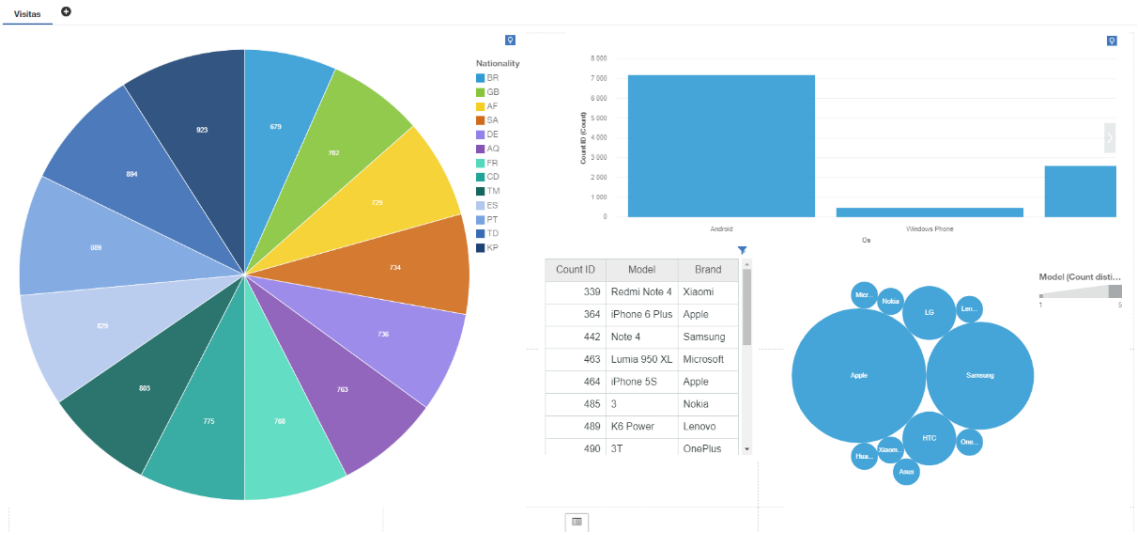
Como referido inicialmente, o *IBM Cognos Analytics* é uma poderosa ferramenta de análise de dados. Para que fosse possível alimentar esta ferramenta com os dados das visitas, foi implementado uma solução que permite a exportação da lista das visitas para um ficheiro de formato XLS ou CSV. Com esta lista é possível a integração no *IBM Cognos Analytics* e obter diversas informações.

Figura 17: IBM Cognos Exploration



Fonte: IBM Cognos Analytics

Figura 18: IBM Cognos Dashboard



Fonte: IBM Cognos Analytics

É possível construir d*ashboards* interativos em que quando um item é pressionado, todos os outros são filtrados com base no que foi selecionado. É possível também utilizar um método simples de relacionamento em que adicionamos um ficheiro de dados e escolhemos qual o atributo que queremos ver e de seguida é apresentado um gráfico que mostra os relacionamentos (no modelo de dados) entre os atributos. Esta opção é relevante quando o modelo de dados é muito grande.

4.5. Segurança ao nível da API

Hoje em dia estamos numa era em que cada vez é necessário mais segurança. Para que as estatísticas dos gráficos sejam as mais fidedignas possíveis, implementou-se um mecanismo de segurança/proteção ao nível da comunicação entre a aplicação móvel e o *backoffice*, para assim evitar ataques de DDoS. Estes ataques são de negação de serviço, ou seja, são uma tentativa de tornar os recursos de um sistema indisponíveis para os seus utilizadores. Para evitar que o servidor seja atacado/sobrecarregado com as comunicações e que seja impossibilitado a disponibilização de recursos entre a aplicação móvel e o *backoffice*, foi implementado um mecanismo de segurança.

Este mecanismo faz com que seja limitado o número de comunicações entre a aplicação móvel e o *backoffice*, ou seja, definiu-se um limite de pedidos que podem ser executados ao mesmo tempo pela API, impedindo assim que o servidor seja sobrecarregado.

Capítulo 5 – Conclusão

Hoje em dia, com a modernização das tecnologias, é cada vez mais fácil e comum obter dados estatísticos dos mais diversos dispositivos. Com as ferramentas adequadas é possível extrair padrões e observar tendências sobre estes dados, que depois permitem otimizar processos ou gerar novas formas de negócio.

Com o desenvolvimento deste projeto, foram adquiridos diversos conhecimentos na área de *Business Inteligence* e analítica de dados, na área das comunicações entre as plataformas, e de ferramentas de testes e integração de dados. Estes permitiram a implementação de soluções robustas e implementadas em cenário real, que ajudam os agentes do turismo a dinamizar as visitas aos monumentos, podendo retirar informação pertinente sobre os seus visitantes.

O sistema "MovTour" estava até agora focado na disponibilização de informação personalizada ao turista que encontra um monumento específico, através do seu *smartphone*. Como fruto deste trabalho, o sistema passa a dispor de toda uma componente de análise da informação recolhida, permitindo às entidades que gerem os monumentos a extração de conhecimento e otimização da sua oferta.

5.1. Objetivos alcançados

Com o desenvolvimento deste projeto, o objetivo de implementar soluções de *BigData* e análise de dados bem como a implementação de *dashboard* dinâmicos e em tempo real foi cumprida, criando uma melhor interação para o utilizador do *backoffice*.

Com estas soluções, foi dada a oportunidade de criação de gráficos personalizados para registo de vários tipos de informações, como por exemplo saber qual o tipo de dispositivos móveis mais comum que visita um determinado monumento. É possível também, saber o número total de visitantes por ponto de interesse e o número total de visitante no geral e em tempo real.

Para garantir que o servidor suporta um fluxo elevado de comunicações foram aplicados dois tipos de mecanismos de otimização, utilização de *cache* e simplificação de *queries*. Foram efetuados também testes de carga e *stress* para, não só, comprovar que os

mecanismos de otimização foram corretamente implementados, mas também demostrar que o servidor tem uma taxa de eficiência bastante alta.

Com a funcionalidade de exportação das visitas para um ficheiro XLS ou CSV, foi possível efetuar a sua importação na plataforma *IBM Cognos Analytics* e criar *dashboards* com essa informação.

Por último foi implementado um sistema de proteção contra ataques de negação de serviço no servidor e de tentativa de introdução de dados inválidos ou incorretos.

5.2. Limitações e trabalho futuro

As limitações encontradas ao longo do período de desenvolvimento deste trabalho foram: a restrição de tempo para a sua realização, uma vez que seria necessário um período mais alargado para o seu desenvolvimento; o facto de ainda não existirem dados reais que permitam uma exploração de soluções reais para a previsão de tendências, uma vez que com dados fictícios é difícil validar tais abordagens.

Como pistas para uma investigação futura sugere-se a implementação de uma base de dados NoSQL, caso o volume de dados aumente, embora o sistema de base de dados utilizado suporte o volume de dados expectável. Uma outra sugestão passa por implementar *machine learning*, quando existir um conjunto de dados reais, que permita prever as soluções e/ou tendências com base em padrões dos dias anteriores. Por fim, poderia constituir um desafio a utilização de algoritmos de classificação supervisionados e não supervisionados nos dados obtidos através das visitas, para ser possível observar a diferença destes dois tipos de algoritmos de classificação.

Referências bibliográficas

[1] Unidad Editorial, “The Daily Prosper,” 29 agosto 2018. [Online]. Available: https://thedailyprosper.com/pt/a/turismo-e-tecnologia-mais-recente-inovacao-no-setor-turistico. [Acedido em 10 novembro 2018].

[2] A. Beleza, “TecheNet,” 7 fevereiro 2014. [Online]. Available: https://www.techenet.com/2014/02/novas-tecnologias-levam-a-um-turismo-digital/. [Acedido em 10 novembro 2018].

[3] Turismo de Portugal, “Turismo de Portugal,” 1 julho 2018. [Online]. Available: https://www.turismodeportugal.pt/pt/Noticias/Paginas/wta-2018-portugal-e-o-melhor-destino-europeu-pela-segunda-vez.aspx. [Acedido em 2018 novembro 2018].

[4] J. N. Santos, “ECO,” 14 fevereiro 2018. [Online]. Available: https://eco.pt/2018/02/14/turismo-bate-recordes-em-2017-hospedes-aumentam-89/. [Acedido em 2018 novembro 2018].

[5] F. E. Anunciação, “DireitoNet,” 24 fevereiro 2019. [Online]. Available: https://www.direitonet.com.br/artigos/exibir/8962/O-mundo-digital-e-sua-importancia-no-cotidiano. [Acedido em 10 novembro 2018].

[6] B. Marr, “Linkedin,” 6 março 2014. [Online]. Available: https://www.linkedin.com/pulse/20140306073407-64875646-big-data-the-5-vs-everyone-must-know/. [Acedido em 10 novembro 2018].

[7] BigData Business, “BigData Business,” [Online]. Available: http://www.bigdatabusiness.com.br/conheca-os-4-tipos-de-analises-de-big-data-analytics/. [Acedido em 10 novembro 2018].

[8] D. Godoi, “Apache Hadoop: Tudo o que você precisa saber,” CETAX | DEAL GROUP, [Online]. Available: https://www.cetax.com.br/apache-hadoop-tudo-o-que-voce-precisa-saber/. [Acedido em 10 novembro 2018].

[9] M. Pfeil, “DataStax,” 18 outubro 2010. [Online]. Available: https://www.datastax.com/dev/blog/what-cassandra-good. [Acedido em 10 novembro 2018].

[10] Cetax, “Cetax,” [Online]. Available: https://www.cetax.com.br/blog/tudo-sobre-o-apache-cassandra/. [Acedido em 10 novembro 2018].

[11] C. Wodehouse, “Upwork,” 14 junho 2016. [Online]. Available: https://www.upwork.com/hiring/development/mongodb-vs-cassandra/. [Acedido em 10 novembro 2018].

[12] M. Pfeil, “DataStax,” 18 outubro 2010. [Online]. Available: https://www.datastax.com/dev/blog/what-cassandra-good. [Acedido em 10 novembro 2018].

[13] Dta Flair Team, “Data Flair,” 13 setembro 2018. [Online]. Available: https://data-flair.training/blogs/cassandra-applications/. [Acedido em 10 novembro 2018].

[14] mongoDB, “mongoDB,” [Online]. Available: https://www.mongodb.com/who-uses-mongodb. [Acedido em 10 novembro 2018].

[15] S. Blitz, “Sisense,” 8 outubro 2018. [Online]. Available: https://www.sisense.com/blog/decide-postgresql-reporting-right/. [Acedido em 10 novembro 2018].

[16] Postgresql, “Postgresql,” [Online]. Available: https://wiki.postgresql.org/wiki/FAQ#What_is_the_maximum_size_for_a_row.2C_a_table.2C_and_a_database.3F. [Acedido em 10 novembro 2018].

[17] IT World Canada, “IT World Canada,” 28 abril 2017. [Online]. Available: https://www.itworldcanada.com/sponsored/ibm-cognos-advanced-analytics-for-the-whole-enterprise. [Acedido em 10 novembro 2018].

[18] R. Godinho, “Good Intelligence,” [Online]. Available: https://goodi.pt/google-analytics/. [Acedido em 10 novembro 2018].

[19] K. Oku, F. Hattori e K. Kawagoe, “Tweet-mapping method for tourist spots based on now-tweets,” p. 1318 – 1327, 2015.

[20] S. Ray, “Analytics Vidhya,” 13 setembro 2017. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/. [Acedido em 10 novembro 2018].

[21] J. Readhead, “MLQ5,” 17 dezembro 2012. [Online]. Available: https://www.mql5.com/en/articles/584. [Acedido em 10 novembro 2018].

[22] A. Freitas, “Web Técnico Universidades Lisboa,” [Online]. Available: http://web.tecnico.ulisboa.pt/ana.freitas/bioinformatics.ath.cx/bioinformatics.ath.cx/indexf23d.html?id. [Acedido em 10 nvembro 2018].

[23] R. Gandhi, “Towards Data Science,” 5 maio 2018. [Online]. Available: https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c. [Acedido em 10 novembro 2018].

[24] M. J. Garbade, “Towards Data Science,” 12 setembro 2018. [Online]. Available: https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1. [Acedido em 10 novembro 2018].

[25] M. Rouse, “Search Enterprise AI,” junho 2018. [Online]. Available: https://searchenterpriseai.techtarget.com/definition/IBM-Watson-supercomputer. [Acedido em 10 novembro 2018].

[26] https://console.bluemix.net/developer/watson/services, “IBM Cloud,” [Online]. Available: https://console.bluemix.net/developer/watson/services. [Acedido em 10 novembro 2018].

[27] L. Sanroma, “Dashboard Design,” 19 outubro 2017. [Online]. Available: https://www.dashboarddesign.com.br/o-que-e-dashboard/. [Acedido em 10 novembro 2018].