Universidade de Évora - Escola de Ciências e Tecnologia

Mestrado em Engenharia Informática

Dissertação

SISTEMA DE INFORMAÇÃO GEORREFERENCIADO PARA CONTADORES DE ELECTRICIDADE NOS PONTOS DE CONSUMO

Filipe Lima Paixão Pereira

Orientador(es) | José Saias

Évora 2024

Universidade de Évora - Escola de Ciências e Tecnologia

Mestrado em Engenharia Informática

Dissertação

SISTEMA DE INFORMAÇÃO GEORREFERENCIADO PARA CONTADORES DE ELECTRICIDADE NOS PONTOS DE CONSUMO

Filipe Lima Paixão Pereira

Orientador(es) | José Saias

Évora 2024

A dissertação foi objeto de apreciação e discussão pública pelo seguinte júri nomeado pelo Diretor da Escola de Ciências e Tecnologia:

Presidente	|	Teresa Gonçalves (Universidade de Évora)
Vogais	|	José Saias (Universidade de Évora) (Orientador) Pedro Salgueiro (Universidade de Évora) (Arguente)

Évora 2024

À Deus, aos meus parentes e amigos.

Prefácio

Filipe Lima Paixão Pereira, casado, duas filhas, formado em Sistemas de Informação pela Universidade FANOR/DEVRY — Brasil em 2013, certificado cisco CCNA, Windows Server 2008 e 2012, reside na cidade de São Tomé, mais concretamente no Bairro Satón — Aeroporto. Sou aluno do Mestrado em Engenharia Informática na Universidade de Évora. A minha motivação para desenvolver o trabalho com esse tema, surge da necessidade de facilitar o trabalho de localização de contadores de eletricidade instalados no país, por ser uma tarefa muito complicada para os operadores de terreno em definir com exatidão os pontos que os mesmos encontram-se instalados.

Agradecimentos

Primeiramente a DEUS por conceder-me à luz do dia e o oxigénio que respiro. Ao ilustre professor José Miguel Gomes Saias pela sapiência, paciência, disponibilidade, humildade e apoio pedagógico na orientação da Dissertação, a ilustre professora Teresa Gonçalves por ajudar de forma paciente, incansável e insistentemente no processo do meu reingresso ao Mestrado em Engenharia Informática. A minha esposa Alzira Xavier Garçês Paixão Pereira e às minhas filhas, Lisandra Barros Paixão Pereira e Lavínea Garçês Paixão Pereira, por estar sempre perto nessa caminhada. Aos meus pais Filipe Paixão Pereira e Deolinda dos Santos Lima e amigos, Eneio Lima Afrodite Baptista, Chris Allen Barroso, Ary Sanches e Inácio Carvalho.

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Acrónimos

URL *Uniform Resource Locator*
POC Pontos de Consumo
TI Tecnologia de Informação
VPN *Virtual Private Network*
OSI *Open Systems Interconnection*
HTTP *Hypertext Transfer Protocol*
IoT *Internet of Things*
STP São Tomé e Príncipe
API *Application Programming Interface*
JSON *JavaScript Object Notation*
REST *Representational State Transfer*
XML *Extensible Markup Language*
SOAP *Simple Object Access Protocol*
UI *User Interface*
UAT *User Acceptance Testing*
MVC *Model-View-Controller*
UML *Unified Modeling Language*
BD Base de Dados
GPS *Global Position System*
JVM *Java Virtual Machine*
JDK *Java Development Kit*
JDBC *Java Database Connectivity*
JPA *Java Persistence API*
POM *Project Object Model*

STS *Spring Tool Suite*

GPL *General Public License*

OS *Open Source*

HTML *Hyper Text Markup Language*

PWA *Progressive Web App*

DER Diagrama de Entidade Relacional

DMS *Degrees Minutes Seconds*

DD *Decimal Degrees*

GIS Sistema de Informação Geográfica

RE *Requirements Engineering*

RF Requisitos Funcionais

RNF Requisitos não Funcionais

V e V Verificação e Validação

TA *Test Automation*

Resumo

Neste trabalho, pretende-se desenvolver um sistema distribuído, com armazenamento persistente e uma interface Web, para dar resposta ao fornecimento de energia, uma vez que tem sido um problema identificar e localizar contadores instalados em diversos pontos de São Tomé e Príncipe. Para resolver a situação, será necessário georreferenciar contadores de eletricidade instalados nos locais de consumo em todo o território do país. A solução desenhada neste trabalho proporciona a recolha de informações dos contadores georreferenciados além de disponibilizar uma interface Web de utilização intuitiva capaz de devolver aos utilizadores conhecimentos de locais de instalação de contadores de energia por meio de solicitações HTTP. Foi desenvolvida uma aplicação Web com o framework Spring Boot, e com o motor de templates Thymeleaf para a componente de apresentação. A aplicação mostra mapas interativos para apresentação de dados georreferenciados. Toda a solução foi avaliada com três categorias de testes, de onde se concluiu que responde ao objetivo de forma eficaz.

Palavras chave: Contador, Sistema Distribuído, Serviços, Georreferenciação, Rest, Leaflet.

Abstract

Georeferenced Information System for Electricity Meters in Consumption Location

In this project, the aim is to develop a distributed system with persistent storage and a Web interface to address the issue of energy supply, as it has been a challenge to identify and locate meters installed at various points in São Tomé and Príncipe. To solve this situation, it will be necessary to georeference electricity meters installed at consumption locations throughout the country. The solution designed in this project provides for the collection of information from georeferenced meters, as well as offering an intuitive Web interface capable of providing users with knowledge of energy meter installation locations through HTTP requests. A Web application was developed using the Spring Boot framework, and the Thymeleaf template engine for the presentation component. The application displays interactive maps for presenting georeferenced data. The entire solution was evaluated with three categories of tests, from which it was concluded that it effectively achieves the objective.

Keywords: Distributed System, Service, Rest, Georreferenced, Meter, Leaflet.

1

Introdução

A questão de georreferenciação tem sido problema significativo para muitos países do continente africano e São Tomé e Príncipe não foge a regra.

A localização dos contadores de eletricidade tem sido motivadas por diversas razões no quotidiano da empresa responsável pelo tratamento e produção de eletricidade em todo o território. O fornecimento de energia para os diversos pontos do país não é uma tarefa fácil, quando se fala de identificação dos contadores instalados. Tudo isto é possível através do avanço tecnológico nesta área, que tem sido cada vez mais significativo [Ant24].

É com base nesse problema, que surge a necessidade de desenvolver a aplicação com uso de tecnologias REST como ponto fulcral para georreferenciar contadores de eletricidade instalados nos pontos de consumo (*consumption location - POC*) pela empresa responsável pelo fornecimento de eletricidade em São Tomé e Príncipe. Será um sistema capaz, de identificar os locais de fornecimento de energia e consequentemente devolver aos utilizadores, informações sobre locais definidos na *API* com uso de solicitações *HTTP*.

Por ser um projeto inovador no ramo de produção e comercialização de energia, disponibiliza relatórios, informações dos contadores, clientes e utilizadores do sistema, além de permitir visualizar informações dos dados persistidos na base de dados através do mapa possível com a integração da biblioteca *Leaflet*.

1.1 Motivação

Com a implementação do sistema pré-pagamento de energia em São Tomé e Príncipe, chegou-se a conclusão que ouve menos custo com os recursos humanos, sobre tudo na coordenação de recolha de leituras dos contadores pós-pago. Contudo, o incentivo maior desse trabalho é associar os recursos as novas tecnologias, incluindo REST e Spring Boot, para renderizar dados no formato JSON numa Aplicação Web que consiga dar resposta ao problema de identificação da localização de contadores instalados. A relevância desse projeto prende-se, sobre tudo, em algo inovador, pois se considera um novo paradigma de implementação de microsserviços através da arquitetura REST, que justifique a georreferenciação de contadores instalados nos pontos de consumo (POC).

A necessidade deste sistema pode ser explicada por crescente importância no acompanhamento dos processos de produção, tratamento e fornecimento de eletricidade aos consumidores, para melhorar facilitar e estabelecer a tarefa de recolha de dados a partir de contadores instalados nos pontos de consumo.

A Instituição responsável pelo fornecimento de água e eletricidade, é uma entidade pública, dotada de autonomia administrativa e financeira, sob tutela do Organismo da Administração Central do Estado, responsável pelo sector de água e eletricidade. Criada juridicamente ao abrigo da alínea a) do Artigo 1.º do Decreto-Lei n.º 34/79 de 21 de junho de 1979, foi formalmente constituída em 31 de dezembro de 1991 através da publicação dos seus Estatutos pelo Decreto n.º 59/91, de 19 de novembro. A Instituição tem por objeto principal a prestação de serviços públicos de produção, transporte de energia em todo o território de São Tomé e Príncipe. Nesses moldes, faz-se necessário dispor de infraestrutura tecnológica que possa dar resposta aos serviços prestados aos clientes.

1.2 Objetivos

1.2.1 Objetivo Geral

Desenvolver uma Aplicação Web que georreferencia informações de contadores de eletricidade, instalados nos POCs com auxílio das requisições HTTP.

1.2.2 Objetivos Específicos

- Identificar com exatidão os contadores de eletricidade instalados;
- Desenvolver uma ferramenta que proporciona e otimiza a recolha de dados dos contadores georreferenciados;
- Disponibilizar uma interface de interação baseada na Web, que permita ao utilizador influenciar diretamente a experiência através da execução das tarefas com funcionalidades disponíveis de forma eficaz, onde a comunicação com o servidor tolera falhas pontuais de ligação à rede;
- Produzir base de dados estrutural para guardar e gerir as informações dos clientes, contadores, coordenadas georreferenciados e utilizadores.

1.3 Estrutura da Dissertação

A composição da dissertação está estruturada por seis capítulos que abrangem diferentes secções do trabalho realizado em torno da implementação da solução de software proposto, uma Aplicação Web, para

visualizar, coletar, tratar e disponibilizar informações mediante API com recursos a coordenadas geográficas, para empresa responsável pela produção e tratamento de Eletricidade em São Tomé e Príncipe. Fazem parte dos seis capítulos a introdução, onde é realizado uma abordagem geral do que se trata o trabalho, assim como os objetivos, motivação, justificativa, estrutura da dissertação e um contexto de como a empresa regula o setor elétrico no país.

O Capítulo 2 descreve o estado de arte e alguns conceitos fulcrais no campo da ciência da computação quando aplicados para georreferenciação de pontos geográficos. No Capítulo 3, propõe-se uma Aplicação Web para apoiar a transição de informações de um local geograficamente identificado com auxílio de API REST. A definição de um protótipo seguido da definição dos requisitos funcionais e não funcionais, o processo de recolha de dados geográficos e finaliza com a identificação e seleção de tecnologias e softwares utilizados na implementação da aplicação que subjuga com o capítulo 4, onde é escrito o procedimento para a visualização de dados georreferenciados. O uso do recurso Unified Modeling Language para modelação de diagramas de caso de uso, de sequência e de classe respetivamente para cada fluxos de processo, definição de MER (Modelo de Entidade Relacional) e DER (Diagrama de Entidade Relacional) além dos filtros para a segurança da plataforma que, por hoje se torna um dos pilares mais relevantes para minimizar as vulnerabilidades de qualquer aplicação. Dando sequência a implementação, é descrita a arquitetura do sistema com evidências de algumas operações, e diferentes cenários de uso de diferentes telas da aplicação, detalhados com identificação de alguns trechos de código que demonstram como a solução foi implementada tecnicamente.

Por hierarquia segue-se o capítulo 5, onde são apresentados cenários de testes de verificação e validação de aplicação, para apoiar algumas análises e comportamento do sistema em tempo de execução. Foram desenhadas experiências para três categorias de testes nesta secção, onde se pretende avaliar o sistema tecnicamente, por teste unitário, de aceitação e desempenho com forte argumento para ilustração dos valores de cada teste feito. O último capítulo 6 conclui esta dissertação com apresentação dos resultados obtidos na aplicação ora desenvolvida, os aspetos do trabalho são discutidos para avaliar se todos os objetivos inicialmente propostos foram alcançados, recomendações e melhorias são enumeradas para trabalhos futuros.

O próximo capítulo apresenta relatos de trabalhos de pesquisa bibliográfica que forneça uma imagem adequada do estado da arte, conceitos e evidências relevantes determinantes para contextualizar e fornecer uma visão geral do trabalho relacionado com tecnologias de georreferenciação de locais analogamente ao uso de REST no processamento de pedidos HTTP, nas aplicações Web.

2

Estado de Arte

Neste capítulo são apresentadas algumas aplicações que direta ou indiretamente possuem algumas similaridades em termos funcionais, ou com o grau de pertinência às desta dissertação. Projetos, desde gestão energética, monitorização de pontos de consumo até georreferenciação de locais através do mapa com marcados são apresentados. Este capítulo é crucial para o entendimento da importância de um sistema de informação georreferenciado, com utilização de recursos e tecnologias que facilitam a coleta e tratamento de dados a partir de uma Aplicação Web num local remotamente identificado com auxílio da API de suporte a coordenadas geográficas.

A ***georreferenciação*** é o processo de associar informações geográficas a um dado ou objeto, posicionando-o em um sistema de coordenadas conhecido, como as de latitude e longitude. Esse processo permite que objetos, imagens, mapas ou dados sejam relacionados a um lugar específico na superfície da Terra [Yao20].

2.1 Abordagem de Sistema de Informação Georreferenciado

Ao medir coordenadas geográficas para encontrar um local, duas das maneiras de mostrar a localização, uma por grau, minutos e segundos, definido por DMS, e por graus decimais, ou DD.

A maioria dos dispositivos GPS fornece coordenadas em graus, minutos e segundo formato (DMS) ou, mais comummente, no formato de graus decimais (DD). O Google Maps popular fornece as suas coordenadas nos formatos DMS e DD [RHW+24].

1. **DD** — é usado para expressar coordenadas geográficas em graus decimais com objetivo de representar a posição de um objeto, local ou mesmo um país, exemplo 41.40338, 2.17403.
2. **DMS** — é usado graus-minutos-segundos para representar um objeto, local ou mesmo um país, onde minutos e segundos variam de 0 a 60 exemplo 41°24'12.2"N 2°10'26.5"E [AHAS21] [MBM01] [FMK+23].

Um Sistema de Informação Geográfica (SIG) refere-se a um sistema de computador que permite analisar e visualizar informações geograficamente referenciadas, transforma dados em mapas interativos, que podem ser analisados e visualizados em diversas perspetivas espaciais [USG24] [LGMR05]

2.1.1 WattWater

Pensando na eficiência energética, e proteção ambiental a Empresa Portuguesa de Água Livres (EPAL), cria a aplicação WattWater ilustrado na figura 1, com objetivo fundamental para apoiar a gestão energética em entidades gestoras de abastecimento de água e saneamento numa perspetiva de otimização operacional e da eficiência energética e económica. Dentre os benefícios da aplicação destaca-se a georreferenciação como ponto fulcral para análise contextual dos valores obtidos através da área privada de monitoramento da aplicação [CDB15][dP24a].

Figura 1: WattWater, fonte (EPAL)

2.1.2 LiteMe

Ilustrado na figura 2, é uma tecnologia de ponta que ajuda indústrias, comércios e residências a monitorar e identificar os potenciais de economia de consumo.

O software é solução na nuvem que monitora diversos pontos de consumo e apresenta tudo num único dashboard sofisticado. Através dessa tela, é possível personalizar que informações irão aparecer e com isso criar um ambiente de monitoramento especial. No entanto, a customização permite criação de novos componentes, integração com outras fontes de dados, criação de novas categorias de indicadores,

dashboards para gestão de vários pontos de medição. Dispõe de uma ferramenta de visualização de consumo georreferenciada, ideal para clientes que precisam monitorar vários prédios ou unidades numa única tela.

Medidores instalados em unidades ou prédios coletam dados de consumo, de energia e água, enviados para a nuvem onde são processados para reduzir custos de operação por meio de mecanismos de segurança e proteção de privacidade dos utilizadores [Let23].

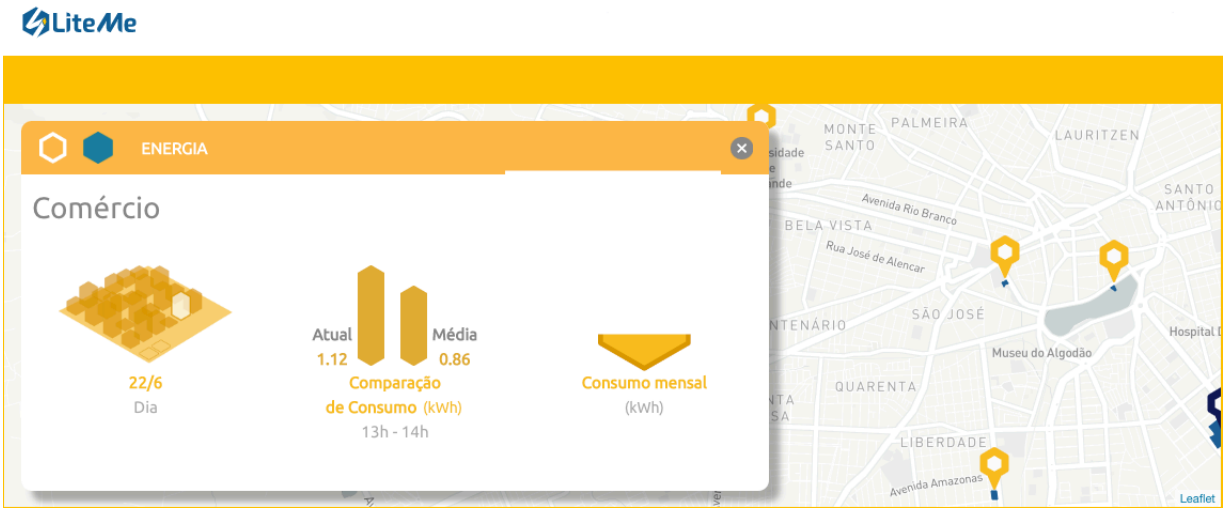

Figura 2: LiteMe, fonte (LiteMe)

2.1.3 REJA

Sistema de recomendação híbrido para restaurantes, colaborativos e baseados no conhecimento, capaz de fornecer recomendações em qualquer situação requerida pelos utilizadores / clientes. Além disso, fornece informações encaminhadas pelo Google Maps, relativamente às recomendações. Esse sistema foi desenvolvido para a província, Jaén (Espanha), mas pode ser facilmente adaptado para qualquer outra área geográfica. Os sistemas de recomendação surgiram neste campo porque o objetivo de tais sistemas é personalizar a informação que recebe os utilizadores conforme as suas preferências, necessidades ou gostos. Devido ao seu sucesso, existe uma ampla gama de aplicações para sistemas, principalmente para o comércio eletrónico de lazer [MRE09].

2.1.4 PASEV's Platform

A plataforma apresenta um mapa centrado na cidade de Évora, com vários marcadores, cada um representando um local. Esses locais possuem detalhes e informações que são mostrados quando o utilizador interage com o marcador. No primeiro clique, um nome, foto, a sua origem e endereço são mostrados ao utilizador, conforme ilustrado na figura 3. Ao clicar num botão para ver mais conteúdo revela várias guias que contêm informações, sons, imagens, vídeos e uma bibliografia. A seleção padrão é a aba de informações que, além das informações mencionadas anteriormente, possuem uma descrição. A bibliografia mostra uma lista de referências e as demais abas dão acesso aos diferentes tipos de conteúdo multimédia [Fer21].

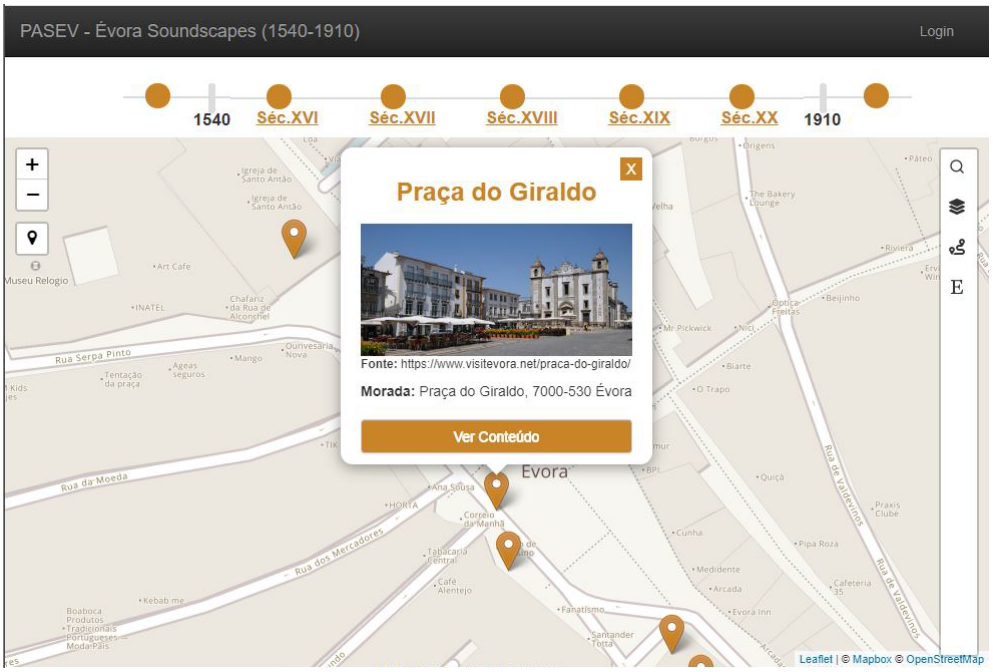

Figura 3: Plataforma PASEV, fonte (PASEV)

2.1.5 H2O Quality

H2O Quality ilustrado na figura 4, é um aplicativo gratuito de georreferenciação de água, a génese no setor de água em todo o mundo, permite a qualquer cidadão ou turista visualizar resultados de qualidade da água no ponto em que encontra, com informações atualizadas diariamente. A aplicação disponibiliza informações de como está a qualidade da água distribuída num local geograficamente georreferenciado, com informação sobre os parâmetros que envolve, cálcio, magnésio, cloro, cor, dureza e pH [dP24b] [MUI+23].

2.1.6 BUPI

Aplicação projetada e suportada para sistemas operativos iOS e Android, com objetivo de auxiliar os proprietários de imóveis e terras. Dispõe de funcionalidades que, permite desenhar áreas em forma de polígonos, mapear e salvar o esboço visual. A filosofia por detrás do BUPI (Balcão Único do Prédio), figura 5, é de garantir o registo dos terrenos na conservatória do registo predial além de disponibilizar ao proprietário a localização da sua propriedade através do desenho cadastral [Fer23] [Sou23].

2.1.7 eTourism

Berger et al. (2007) descreveram um ambiente de eTourism baseado numa abordagem voltada para a comunidade para promover uma sociedade ativa de viajantes que trocam experiências de viagem e recomendam destinos turísticos.

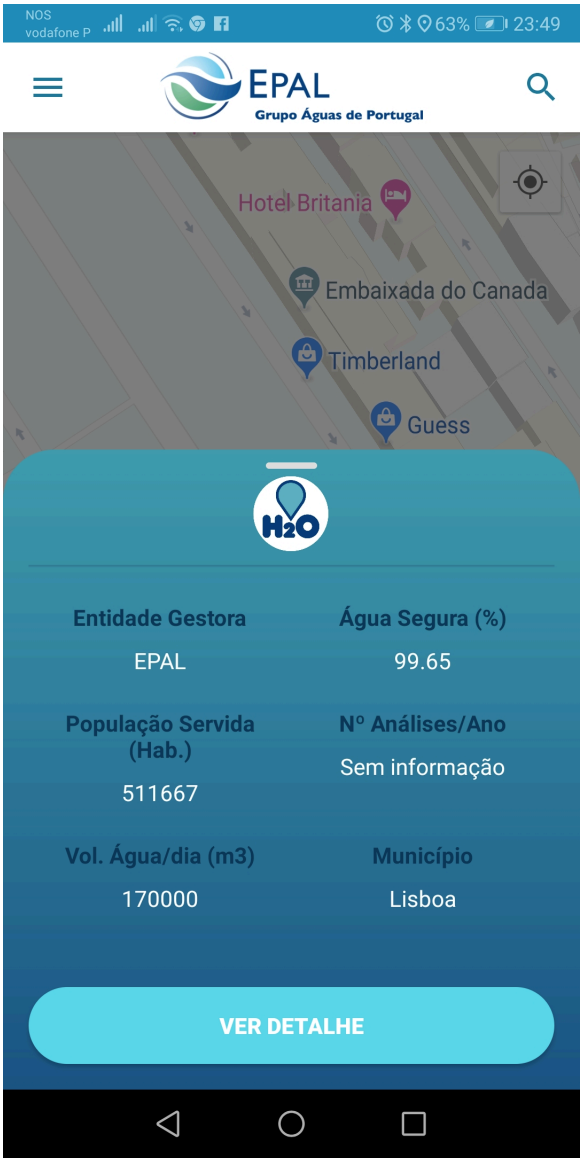

Figura 4: H2O Quality, fonte (EPAL)

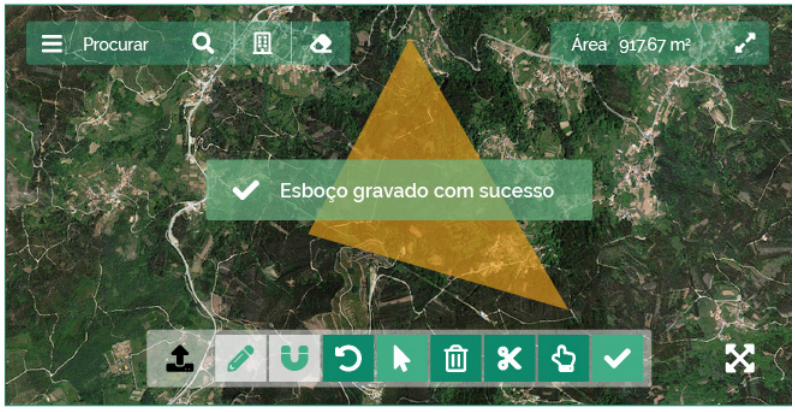

Figura 5: BUPI, fonte (BUPI)

2.1.8 Idris e Yahaya (2009)

Discutiram o projeto e a implementação de um sistema de informações de turismo na Web usando a agregação da Web como motor central.

2.1.9 Anderson e Souleyrette (2002)

Recomendaram uma abordagem de modelação dinâmica aproximada para avaliar a implantação da tecnologia da informação do viajante integrando um sistema regional modelo de demanda de viagens e pacote de microssimulação num ambiente GIS.

2.1.10 Minagawa e Nami (1999)

Aplicaram o GIS para localizar áreas adequadas para o desenvolvimento do turismo na Ilha de Lombok, na Indonésia. Turk e Gumusay (2004) realizaram o projeto de GIS e a análise de rede tirando proveito das possibilidades de GIS para o turismo. O estudo foi realizado no distrito de Eminonu, onde existem diversos locais históricos e turísticos.

2.1.11 Othman et al. (2010)

Abordaram o uso da tecnologia GIS no desenvolvimento de um banco de dados e na análise de dados associados às acomodações turísticas nos estados da costa leste da Malásia: Pahang, Terengganu e Kelantan.

2.1.12 Feick e Hall (2000)

Se concentraram na aplicação de tecnologia de Sistema de Informação Geográfica num design de software customizado para permitir múltiplos participantes de vários setores no distrito de West Bay em Grand Cayman, Índias Ocidentais britânicas para designar parcelas de terra apropriadas para o desenvolvimento relacionado ao turismo ou para um uso competitivo da terra.

2.1.13 Dye e Shaw (2007)

Apresentaram um aplicativo de sistema de apoio à decisão espacial baseado em GIS (SDSS) integrando funções de GIS e designs de SDSS com interfaces gráficas de utilizador easytouse para ajudar os visitantes do Parque Nacional das Grandes Montanhas Fumegantes (GSMNP) a escolher e planear as suas atividades eficazmente para corresponder as suas preferências e restrições pessoais.

2.1.14 Olafsdottir e Runnstrom (2009)

Desenvolveram uma metodologia para gerar um Sistema de Apoio à Decisão em Turismo (TDSS) para auxiliar no planeamento do turismo sustentável. Um modelo GIS foi desenvolvido com base na classificação de fatores e variáveis de impacto identificados, bem como algoritmos de classificação selecionados.

2.1.15 Bertazzon et al. (1997)

Discutiram as possibilidades teóricas de usar TGIS para fins de marketing turístico tradicional e também forneceram um relato detalhado do uso da Internet para incorporar modelos e gráficos baseados em GIS como uma ferramenta de marketing para a indústria de Alberta Ski Resort.

2.1.16 Yolanda e Hernndez (2011)

Usam um modelo de preços hedônicos, estimaram a influência de algumas variáveis espaciais e ambientais nos preços de aluguéis de casas rurais na Gran Canaria, Espanha e recomendou que este modelo ajude a orientar as políticas de diversificação turística aplicadas à ilha. Os recursos de correio do GIS são para integrar grandes volumes de dados espaciais e não espaciais e melhorar a compreensão do problema por meio da visualização de dados na forma de mapas.

2.1.17 Deviana (2011)

Implementar serviços da Web para oferecer suporte à integração entre plataformas de sistema e aplicativos. O conjunto de dados usa estoque de farmácias centrais e pontos de venda. O método usa SOAP para arquitetura de serviço da Web. A avaliação dos dados da transação pode ser vista porque, usando o esquema XML, as categorias de dados podem ser rastreados entre o cliente e o servidor [Dev11].

2.1.18 Sibagariang (2016)

Discute que a comunicação entre servidor e cliente tende a ser vulnerável como segurança, aceleração, eficácia e eficiência de um sistema. O conjunto de dados usa livros da biblioteca da Universidade Católica de Santo Thomas. A arquitetura do serviço da Web usa a API REST. Na avaliação desta dissertação, a utilização de serviços Web, cada cliente pode acessar os mesmos dados de plataformas diferentes. A limitação desta dissertação é que o sistema de segurança para serviços da Web não é descrito [Sib16].

2.1.19 Moodle

Aziz, Wiharto, e Wicaksono (2013), constataram que o sistema Moodle com outros sistemas externos que desejam ser integrados ao sistema Moodle. Portanto, esse serviço da Web basicamente não é adequado se usado para dispositivos móveis. A arquitetura do serviço da Web usa a API REST. Com o serviço da Web REST, o serviço da Web Moodle que foi originalmente criado para as necessidades de sistema também pode ser usado para sistema para utilizador. Portanto, é necessário criar um modelo de aplicação para que os smart clientes possam acessar funções de serviço Web que atendam às necessidades dos utilizadores móveis [Wic14].

2.1.20 Wagh e Thool (2012)

Constataram que fornecer serviços, Web em dispositivos móveis porque o número de utilizadores é móvel e está sempre aumentando. Portanto, o autor deseja fazer uma comparação entre SOAP e REST em serviços da Web. O conjunto de dados usa várias estruturas que podem ser usadas em dispositivos

móveis. A avaliação é a estrutura usada para teste pode funcionar bem com arquiteturas SOAP e REST. No entanto, o autor não inclui um sistema de segurança no seu artigo e existem vários frameworks que nunca foram testados [WT12].

2.1.21 Naskah (2010)

Sugere como integrar diferentes programas aplicativos e entre plataformas, mas neste estudo limita-se a levar funções php num servidor Web para serem utilizadas em outro servidor Web, ou seja, gerir sítios Web. O conjunto de dados usa várias partições em que cada partição recebe uma entrada e compara a saída com os requisitos fornecidos. A arquitetura do serviço da Web usa a API SOAP. A avaliação efetuada é que o serviço Web construído consegue proporcionar uma relação eficaz entre empregadores e candidatos, isto devesse à facilidade de ligação ao sítio Web do candidato [Nas11].

2.1.22 GEO API PT

João Pimentel Ferreira, Engenheiro Eletrotécnico e de Computadores (2022), no seu mais recente trabalho na área de Tecnologia de Informação, desenvolveu uma API RESTful com integração a biblioteca Leaflet para dar resposta aos problemas de regiões administrativas oficiais de Portugal, não é exatamente uma aplicação direcionada para representação de informações relacionada com contadores de eletricidade, mas sim, para disponibilizar através dos endpoints informações oficiais sobre as regiões administrativas oficiais de Portugal (com base na Carta Administrativa Oficial de Portugal, 2022). Ela Inclui informação sobre Portugal Continental, Açores e Madeira. Também fornece informações sobre os códigos postais (obtido e atualizado do sítio dos CTT e do INE em dezembro de 2023) e os censos estatísticos (CENSOS 2021) [Joa24]. A partir de um ficheiro JSON configurado com os dados figura 7, é possível capturar informações do Município através do endpoint e devolver respostas por meio operações que ocorrem no servidor. Os dados são processados conforme a lógica de negócio e manipulação de solicitações por meio do cliente.

2.1.23 Mapping Corona Virus

Herman Milton Maleiane em Moçambique 2020, desenvolve uma aplicação com a tecnologia Spring Boot com integração da biblioteca Leaflet, onde as pessoas podem visualizar informações relacionadas ao Corona vírus por via de dados alfanuméricos e espaciais utilizando Spring Boot para criar serviços em Java, Thymeleaf para a camada de visualização de dados alfanuméricos e o Leaflet para renderizar os mapas (Informações espaciais). Através do ficheiro csv com os dados de COVID-19 disponível no site [1], é possível criar o serviço que realiza uma solicitação HTTP para a URL especificada onde os dados de casos de infeções são atualizados, reportados e marcados diariamente no mapa conforme é ilustrado na figura 8 [Mal24].

2.1.24 Netherlands Mapa OpenStreetMap LEAFLET + GeoJSON

Lucas Jellema, CTO e arquiteto de TI, Diretor do Oracle ACE, cria um mapa da Holanda (a partir de um arquivo GeoJSON com províncias e grandes cidades), os mapas são personalizados com marcadores no topo do mapa para os locais dos escritórios do ecossistema com a chamada a partir de outro arquivo

[1] `https://github.com/CSSEGISandData/COVID-19`

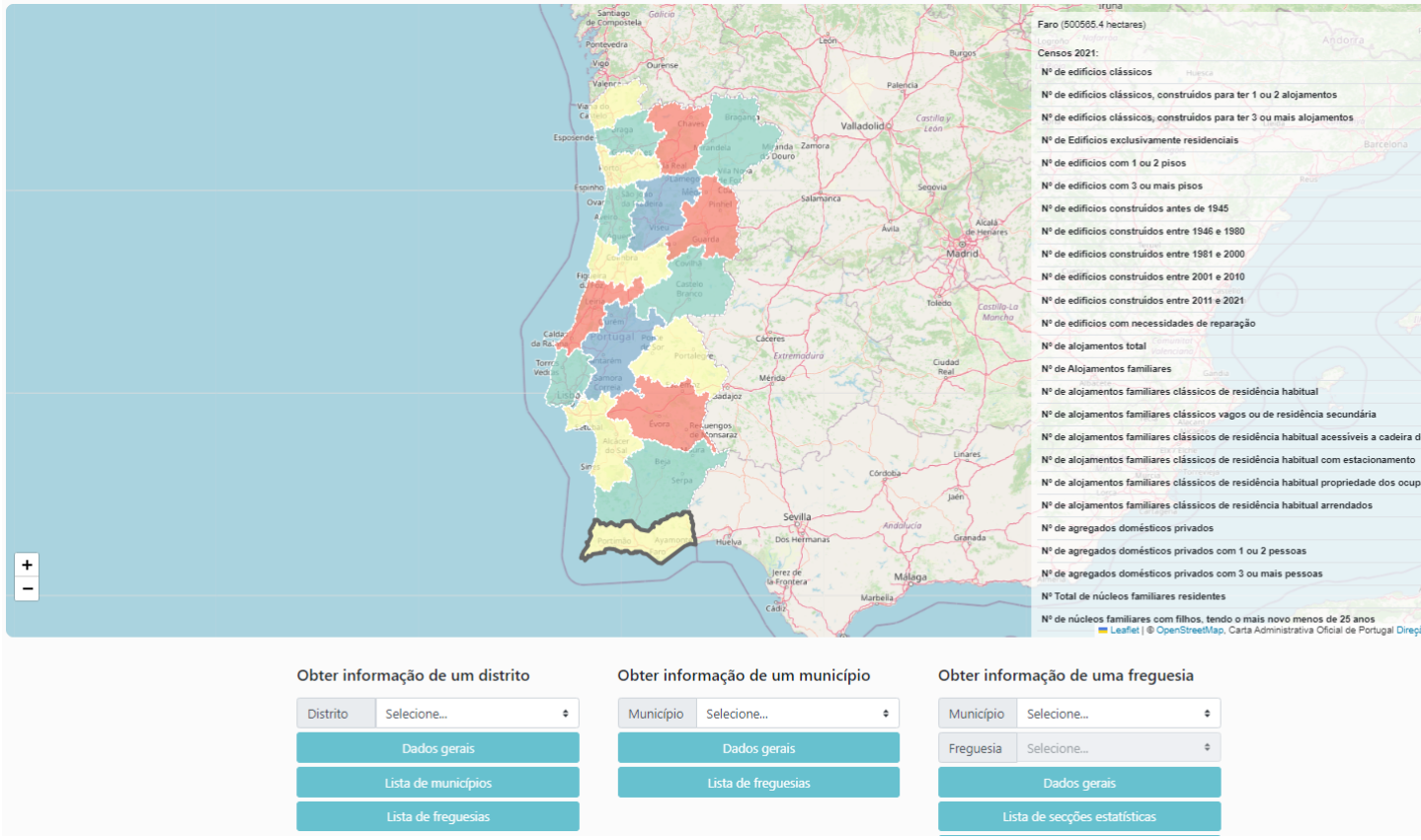

Figura 6: Regiões Administrativas oficiais de Portugal

GeoJSON. Usa o Leaflet e OpenStreetMap para criar uma camada sobreposta que possui marcadores baseados nas feições do arquivo GeoJSON com definição de estilos e também um Popup que aparecerá quando o marcador for clicado conforme é apresentado na figura 8 [Jel24].

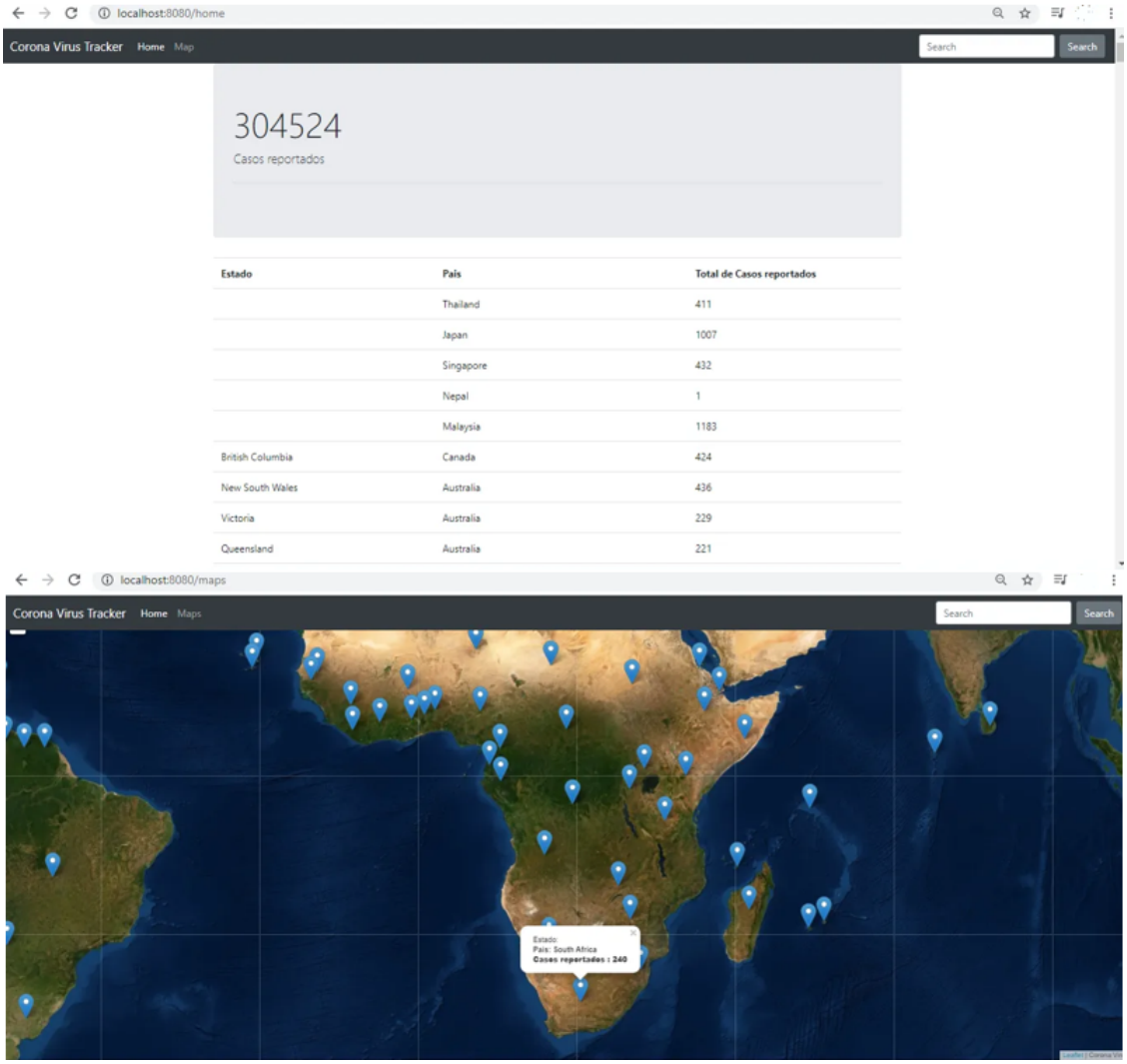

Figura 7: Casos Reportados e Áreas de Infeções, fonte (Mapping Corona Virus)

2.2 Síntese

A aplicação desenvolvida neste trabalho que se designa GeoMeter, é uma sistema de informação georreferenciado capaz de auxiliar na resolução de problemas relacionados com localização dos contadores instalados nos pontos de consumo, como estratégia para monitorar e controlar a eficácia da gestão de fornecimento de eletricidade em São Tomé e Príncipe.

A georreferenciação dos contadores de eletricidade nos pontos de consumo é o aspeto fundamental deste projeto. Porém, a localização destes deve ser de tal forma precisa para melhor interesse do projeto, certamente para que, isso seja materializado necessário se torna incluir estudos e análise meticulosamente dos elementos envolventes e utilizadores que participam na interação com o mesmo.

Uma vez que não foram encontrados sistemas que dessem resposta conveniente à necessidade existente, especificamente voltado para georreferenciação de contadores de eletricidade anteriormente, esse projeto é a base de informações de grande valia para empresa responsável pelo tratamento e produção de eletricidade no país. Dito isto, em termos de representação de contadores no mapa, fica claro que o projeto é ambicioso e se torna a opção ideal para referência dos trabalhos futuros na área de georreferenciação.

Figura 8: Mapa da Holanda, fonte (Mapping Corona Virus)

Conforme mencionado na secção (2.1.22, 2.1.23 e 2.1.24), o uso da Leaflet tem se popularizado e tornado uma das principais bibliotecas do código aberto para visualização de mapas dinâmicos.

Devido às suas características leves, possui boa compatibilidade com o Spring Boot para carregar várias camadas e a velocidade de carregamento dos dados no mapa e integração é muito rápida. O principal desafio é fazer a configuração correta da mesma para explorar ao máximo os benefícios a qualidade do conteúdo. As vantagens são inúmeras com destaque para simplicidade, desempenho e usabilidade, além de ser compatível com *HTML5, CSS3* e os navegadores modernos. O capítulo 3, é reservado a exposição a mais alto nível da implementação de um Sistema de Informação Georreferenciado para Eletricidade, onde é verificado minuciosamente todos os aspetos que envolve funcionalidade, design, arquitetura e segurança de um protótipo de resposta ao problema de georreferenciação de contadores instalados nos pontos de consumo.

3

Sistema de Informação Georreferenciado Para Eletricidade

Neste capítulo, é apresentado o protótipo inicial alinhado aos objetivos e requisitos para sua implementação. Arquitetura do sistema proposto, os requisitos da solução e alguns aspetos relevantes dada a característica da aplicação em causa. A identificação das funcionalidades do sistema, definição dos componentes e como ocorre o fluxo de decisões para o bom funcionamento dos principais módulos do sistema também são detalhados.

3.1 Protótipo

Esta dissertação cumpre os fundamentos mais importantes da etapa de prototipagem, sob os aspetos relacionados com o esboço inicial do sistema usado para demonstrar conceitos, design e descobrir problemas e as suas possíveis soluções. Para isso, é preciso cumprir as etapas do desenvolvimento. O protótipo da aplicação é usado no processo de desenvolvimento de software para ajudar acionar a pro-atividade nas mudanças que podem ser necessárias, obviamente esta dissertação cumpre os requisitos para tal, nomeadamente: [Som11]

- **Mapa** - correspondência gráfica de um espaço real da ilha de S. Tomé e Príncipe;
- **Nomes de lugares** - identificação dos locais com informações geográficas descriminados em latitude e longitude para diversos pontos do país;
- **Marcadores** - são etiquetas que contêm informações relacionados com os clientes, contadores e georreferenciação;
- **Eventos** - *link* que contém detalhes do cliente associado as informações do contador e do ponto onde o mesmo encontra-se instalado.

Para a distribuição, produção e comercialização de energia, é essencial planear como os medidores estão mapeados nos pontos de consumo em todo o território nacional. Isso é fundamental para que, a informação seja centralizada e efetiva. Assim, são estabelecidos os requisitos para a formalização e implementação do sistema.

3.2 Definição de Requisitos

Os requisitos de um sistema são contextos ou tarefas que um sistema deve alcançar ou realizar, com objetivo único de satisfazer um ou mais documento que envolve a regra de negócio de uma organização, por outras palavras, são condições imprescindíveis para o funcionamento de um sistema de informação [SOM].

3.2.1 Requisitos Funcionais

Para a aplicação proposta, são declarados algumas funções de carácter funcional que o sistema deve cumprir para que, possa ocorrer a melhor iteratividade com o utilizador. Em alguns casos, os requisitos funcionais também podem explicitar o que o sistema deve fazer [SOM].

Os requisitos funcionais para o sistema a desenvolver são:

- [RF001] A aplicação deve permitir adicionar novos clientes;
- [RF002] Deve ser possível adicionar contador;
- [RF003] Deve ser possível pesquisar por palavras-chave qualquer contador, cliente ou utilizador armazenado na base de dados;
- [RF004] Um utilizador deve conseguir associar um cliente e uma localização a um contador;
- [RF005] É condição necessária gerar relatório da lista dos contadores, clientes e utilizadores;
- [RF006] A aplicação deve mostrar de forma visual no mapa, todos os contadores existentes na base de dados relacional;
- [RF007] O sistema deve oferecer condições de acesso rápido aos detalhes dos clientes associado a um contador.

3.2.2 Requisitos não Funcionais

Para garantir a qualidade da aplicação proposta, foi necessário prevê e envolver algumas características de funcionalidades e têm origem no fluxo de processos e tratamento de dados de componentes:

- Autenticação;
- Desempenho;
- Segurança;
- Usabilidade;
- Tratamento de comunicação de rede

Assim sendo, foram identificados alguns requisitos não funcionais para o sistema proposto:

- [RNF001] Após o login do utilizador, será apresentado a página as funcionalidades conforme o seu perfil;
- [RNF002] Os relatórios de clientes, contadores e utilizadores persistidos na base de dados, são gerados em pdf;
- [RNF003] O acesso à aplicação não pode demorar mais do que 5 minutos;
- [RNF004] O sistema deve ser implementado em Spring Boot com motor de templates, para produção da componente de vista/apresentação Thymeleaf;
- [RN005] Para garantir a visualização dos contadores no mapa através dos marcadores, é condição usar a biblioteca Leaflet.

3.3 Abordagem

GeoMeter é uma aplicação com modelo padrão MVC que inclui pastas separadas para garantir de forma independente os acessos e tratamento dos métodos distribuídos em vários pacotes. Nessa disposição, ocorre o processo de subdivisão em 3 camadas possíveis a considerar que explico com mais detalhes no parágrafo a seguir.

Arquitetura MVC (*Model, View, Control*), onde cada componente deste modelo desempenha uma função no sistema, camada VIEW (com interface para utilizadores), camada lógica no que lhe concerne é responsável pelo tratamento das regras de negócio do sistema, e por fim a camada de Controlo ou Serviços. A camada VIEW simplifica as tarefas dos técnicos de terreno com o sistema para realização de *inputs* de dados, leitura e análise de informações relacionados com a produção e comercialização de energia nos pontos de consumo. O processo de execução, esta é a primeira camada que opera tarefas por ações ou pedidos que decorrem de utilizadores por eventos de *clicks*, tornando assim a interatividade com às demais camada para ocorrer o *output* em forma de resultados que envolve às camadas 3 e 4 do modelo OSI (*Open Systems Interconnection*), nomeadamente *network* e *transport*.

Este padrão divide o aplicativo em três camadas diferentes: modelo, visualização e controlador, que atuam como componentes independentes e permitem que os desenvolvedores trabalhem em partes diferentes sem afetar todo o sistema [KSBW16].

A arquitetura em camadas nos aplicativos estabelece regras sobre a comunicação entre diferentes níveis. Este modelo de arquitetura facilita o isolamento, permitindo que apenas as camadas diretamente relacionadas sofram impactos quando uma camada específica é modificada ou substituída. Restringindo as dependências entre as camadas, é possível minimizar o efeito das mudanças, garantindo que uma alteração isolada não comprometa o funcionamento integral do aplicativo [Bas12].

A estrutura em camadas, associada ao isolamento, facilita significativamente a substituição de funcionalidades num aplicativo. Por exemplo, um aplicativo pode inicialmente utilizar o seu próprio banco de dados SQL Server para persistência [Bas12].

A camada lógica controla o funcionamento da aplicação processando comandos, avaliando e calculando dados específicos. As informações da camada de interface do utilizador são posteriormente armazenadas nos servidores de banco de dados da camada de dados e essas informações armazenadas são então transferidas de volta para a camada lógica para processamento posterior e, finalmente, para o sistema de computador do utilizador [Mic23a].

3.4 Processo de Recolha de Dados Georreferenciados

Para a coleta de dados georreferenciados, foi utilizada a ferramenta Google Maps, acessível através do sítio Web. "https://maps.google.com", conforme o exemplo disposto na figura 9, para recolha dos valores de latitude e longitude em diversos pontos de São Tomé e Príncipe e persistido na base de dados na tabela georreferencia conforme a figura 10. Esses mesmos valores previamente cadastrados na tabela

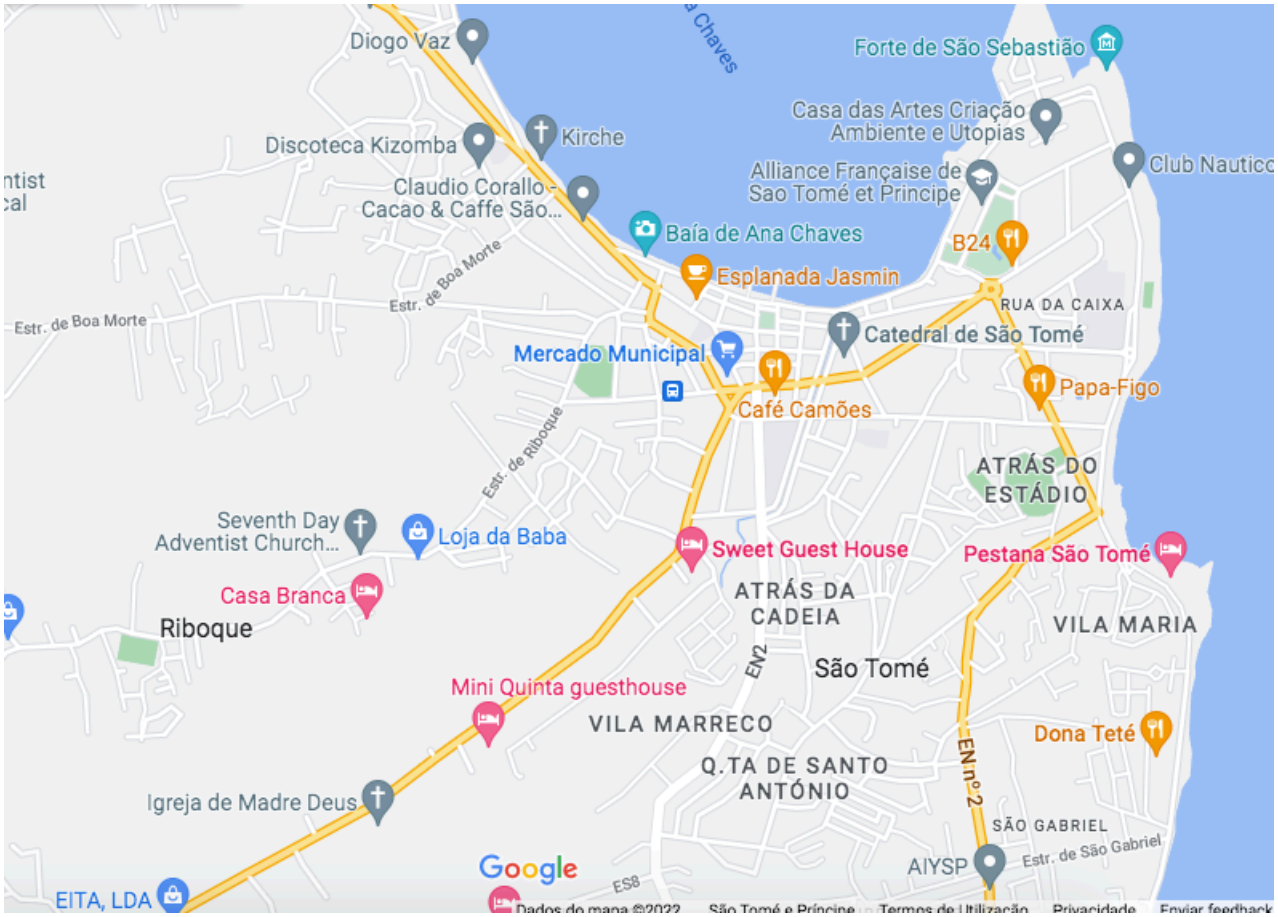

Figura 9: Imagem da Google Map, fonte: Autor

georreferencia com as devidas descrições, será objeto de correspondência para localizar os contadores elétricos por todo o país, todavia os mesmos ficam disponíveis para ser vinculado de forma dinâmica aquando se cogita efetuar a inserção de um novo contador como se apresenta no exemplo da figura 11.

idgeorreferencia	latitude	longitude	descricao
1	0.3259966	6.7247009	T
2	0.3217278	6.7292763	L
3	0.322511	6.7247015	M
4	0.3262042	6.7291684	U
5	0.3276909	6.7303989	O
6	0.3276909	6.7303989	C
7	0.3320704	6.7409314	P
8	0.3399808	6.7375332	R
9	0.3337227	6.7373283	A
10	0.3351384	6.7312433	P
11	0.337359	6.7373311	P
12	0.3385161	6.7386468	L
13	0.3355888	6.7263235	M
14	0.344320	6.722941	D
15	0.338282	6.730795	C
16	0.328405	6.741577	C
17	0.340327	6.738619	U
18	0.338685	6.739692	M
19	0.337881	6.739778	E

Figura 10: Tabela Georreferencia, Fonte: Autor

Figura 11: Lista de Contadores, Fonte: Autor

3.5 Tecnologias e Softwares Utilizados

3.5.1 Spring Framework

O Spring Framework é uma ferramenta de código aberto dividida em módulos que suporta uma ampla variedade de cenários de aplicativo Java para garantir a flexibilidade de criar várias categorias de arquiteturas [Spr23c].

Relativamente à filosofia de divisão dos aplicativos em camadas, é importante observar a necessidade de comunicação entre os mesmos. Na região central, é visível a camada principal que, fornece os fundamentos de inversão de controle o (IoC) para injeção de dependência no Beans. O Spring Framework dispõe de diferentes arquiteturas de aplicativos que inclui, mensagens, dados transacionais e persistência e Web.

Outra observação importante sobre os módulos, recai na ideia de ter os jars do Framework do Spring, que permitem a implantação no caminho do módulo do JDK 9 (“Jigsaw”). Para uso em aplicativos habilitados para Jigsaw, os jars do Spring Framework 5 vêm com entradas de manifesto “Automatic-Module-Name” que

definem nomes de módulos de nível de linguagem estável, ("Spring.core", "Spring.context", etc.) nomes de artefactos jar (os jars seguem o mesmo padrão de nomenclatura com "-" em vez de ".", por exemplo, "Spring-core" e "Spring-context").

O Spring Framework, também suporta as especificações Dependency Injection (JSR 330) e Common Annotations (JSR 250), que os desenvolvedores de aplicativos podem optar por usar em vez dos mecanismos específicos do Spring fornecidos pelo Spring Framework.

Dada a evolução constante do framework Spring e pelas características do projeto de dissertação, houve necessidade de adicionar outros módulos como: Spring Security, Spring Data JPA Hibernate, Thymeleaf, Maven, Spring Boot DevpTools, MySQL JDCB, entre outros, para o bom funcionamento do sistema GeoMeter. É importante lembrar que cada módulo tem o seu próprio repositório de código-fonte, rastreador de problemas e cadência de lançamento [Wal22] [PRPR17].

3.5.2 Spring Security

Spring Security é uma ferramenta de segurança aplicacional com características de promover a autenticação, autorização e proteção contra ataques comuns dos sistemas baseados em Spring. A sua principal finalidade é gerir autenticação e autorização dos níveis de requisições Web por via de chamadas métodos. [Spi20] [WM12] [Sca13].

3.5.3 Spring Data JPA Hibernate

Spring Data JPA fornece o elo para persistir dados entre objetos do Java e Bancos de Dados Relacional por meio de técnicas ORM (*Object Relational Mapping*) Hibernate para mapear as classes Java específicas para tabelas de banco de dados. Para que o processo da persistência de dados seja efetivo, é necessário adicionar o recurso *javax.persistence* que contém as classes e interfaces JPA [KSN18].

3.5.4 Thymeleaf

É um motor de templates, para produção da componente de vista/apresentação em Frameworks Java Server-Side para desenvolvimento Web e que seguem a abordagem *Model-View-Controller* (MVC). Moderno mecanismo capaz de processar *HTML, XML, JavaScript, CSS* e até mesmo texto simples. O principal objetivo do *Thymeleaf* é disponibilizar de forma mais simples a criação e manutenção de protótipo. Isto tudo é possível com base no conceito de injeção da lógica em arquivos de modelo de uma forma que não afete o uso do modelo como protótipo de design. Isso melhora a comunicação do design e preenche a lacuna entre as equipas de design e desenvolvimento. O *Thymeleaf* também foi projetado desde o início com os padrões da *Web* em mente, especialmente *HTML5* que permite criar modelos totalmente validados [Thy23].

3.5.5 Maven

O Apache Maven é uma poderosa ferramenta de gerenciamento de projetos de software, amplamente utilizada para automatizar o processo de construção, relatório e documentação de projetos. Baseado no conceito de Project Object Model (POM), o Maven centraliza informações do projeto, permitindo uma gestão eficiente de dependências, configurações e ciclos de desenvolvimento [Mav24].

3.5.6 DevTools

É uma dependência ou módulo disponível no projeto Spring Boot que, tem objetivo, melhorar a experiência de tempo de desenvolvimento dos aplicativos Spring Boot. Também é conhecido como a ferramenta responsável por executar o reinício automático a cada alteração realizada no projeto [Dev24b].

3.5.7 MySQL JDBC Driver

MySQL JDBC Driver, é uma interface de programação que permite que aplicativo Java aceda a um banco de dados relacional. O GeoMeter, a aplicação a ser desenvolvida, precisa de um driver JDBC para aceder, consultar e persistir dados na base de dados relacional. Originalmente executar o GeoMeter, precisa adicionar a dependência *MySQL-connector-java*, conforme a figura 12, no caso particular para o uso do framework Spring o driver JDBC apropriado [FEEB03].

```
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <scope>runtime</scope>
</dependency>
```

Figura 12: Dependência MySQL JDBC Driver, Fonte (Autor)

Pelas características das tarefas a ser desenvolvido nesta dissertação, adotou-se a utilização dos seguintes softwares:

3.5.8 Spring Tool Suite (STS)

Spring Tool Suite é um Ambiente de Desenvolvimento Integrado para desenvolver aplicativos Spring, baseado em Eclipse. Fornece um ambiente pronto para usar, implementar, executar, implantar e depurar o aplicativo. Valida e fornece correções rápidas para os aplicativos. [HH12] [CHSH17].

3.5.9 MySQL

MySQL é um sistema para gestão Base de dados Relacional de código aberto, com foco para armazenar dados de forma estruturada e organizado em linhas e colunas que permite os utilizadores criar, gerir e manipular bancos de dados e o seu uso é garantido pelos níveis de segurança da confiabilidade, escalabilidade e facilidade e pode ser consultado no site [1] [SZT12] [bU22].

3.5.10 Workbench

Neste projeto de dissertação, a escolha da ferramenta visual para manipulação dos dados persistidos é o *Workbench*, pela facilidade da administração e abstração de utilizador da base de dados na totalidade e ser multiplataforma [bU22].

[1] `https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html`

3.5.11 HeidiSQL

HeidiSQL é uma ferramenta gráfica de gestão de bancos de dados, especialmente popular entre os desenvolvedores e administradores de sistemas que trabalham com MySQL, MariaDB, PostgreSQL e MS SQL Server. Desenvolvida para ambientes Windows, ela oferece uma interface amigável e intuitiva para executar uma ampla gama de tarefas de banco de dados, desde operações simples de consulta até a administração complexa de esquemas e dados.

3.5.12 Java Development Kit (JDK)

O JDK conforme a disposição na figura 13, permite escrever aplicativos desenvolvidos uma vez e executados em qualquer lugar em qualquer máquina virtual Java. Os aplicativos Java desenvolvido com o JDK num sistema podem empregar as técnicas de refatoração de códigos para beneficiar outros sistemas sem alterar ou recompilar [IBM23a].

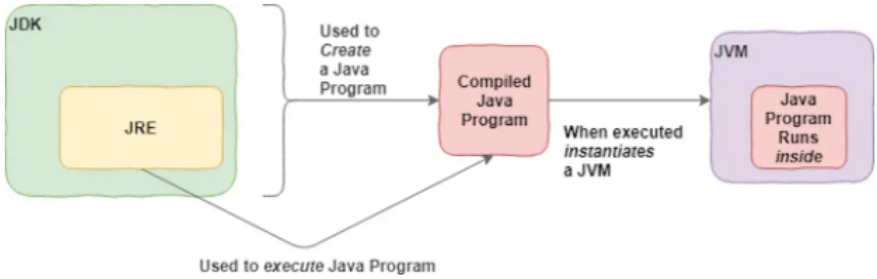

Figura 13: Arquitetura JDK, Fonte IBM

3.5.13 Leaflet

O projeto GeoMeter incorpora funcionalidades de visualização espacial, representando localizações registadas na base de dados sobre mapas exibidos com a biblioteca *Leaflet*. Por ter características multiplataforma e apresentar um bom desempenho, a tecnologia foi selecionada para a visualização de dados georreferenciados no presente trabalho [CI14] [Lew16].

3.5.14 Componentes do Sistema

Este capítulo aborda os componentes do sistema a desenvolver com ênfase para quatro itens principais a destacar:

1. **Sistema de Georreferenciação**, aplicação capaz de permitir a recolha de coordenadas geográficas por via normas de representação Latitude e Longitude, no formato **DD** ou **DMS** por intermédio do posicionamento do utilizador.
2. **Estação de Trabalho**, repositório onde é processado a aplicação Spring Java com objectivo de realizar operações de leitura, inserção, consulta, escrita e exclusão de dados persistidos na base de dados do GeoMeter;

3. **Servidor de Banco de Dados**, armazena todas as informações relativas ao cliente, contadores de electricidade, localização e utilizadores;
4. **Servidor Web**, garante a comunicação por via de solicitações *HTTP* (protocolo de transferência de hipertexto) do utilizador e o sistema.

3.6 Estrutura da Aplicação

O sistema de informação georreferenciado é uma ferramenta que facilitará os técnicos na recolha, identificação e centralização de informações dos contadores instalados exatamente nos pontos de consumo bem como servir de guia de recurso a visualização no mapa.

Para isso, adota às seguintes funcionalidades para o desenvolvimento das diferentes camadas estruturais para aplicação seguindo a lógica MVC ilustrado na figura 14, podemos observar como é distribuído os pacotes e respetivas camadas para execução de todos os processos afetos a mesma.

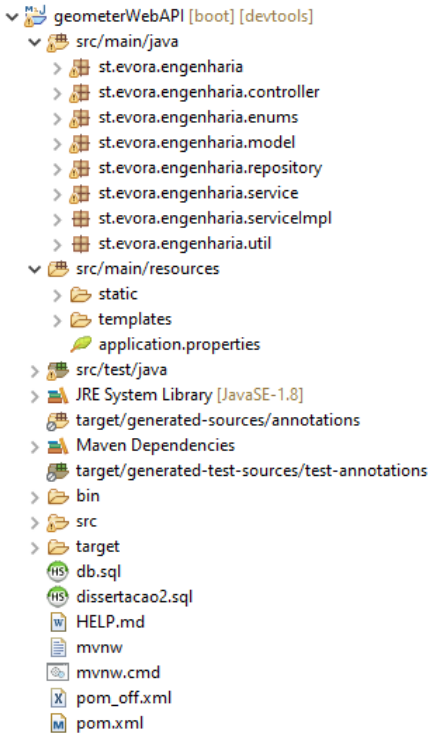

Figura 14: Disposição de Pacotes do Sistema GeoMeter, Fonte: (Autor)

3.6.1 Modelação dos Casos de Usos e Diagrama Classe

Face à identificação dos casos de uso e os seus respetivos diagramas, é importante definir o conceito chave pela qual a Modelação se centra, sobre tudo quando se trata de construção de artefactos de softwares complexos.

Em Engenharia de software, a UML é um padrão de Modelação que outorga representar um sistema de forma padronizada (com intuito de simplificar a abstração da sua implementação). É adequada para a Modelação de aplicações de grande volume de dados no seu fluxo de processos [RJB04].

O caso de uso desenha o mapa mental, da forma como o utilizador interage com a aplicação sob perspetiva de delinear etapas necessárias para o alcance de tarefas, em virtude do cumprimento rigoroso dos requisitos funcionais e não funcionais e devolver resposta ao utilizador da forma de mensagens, notificações ou imagens na tela do mesmo, por exemplo (autenticar, listar clientes, listar contadores, listar coordenadas, listar utilizadores, etc.).

Assim sendo, para aplicação GeoMeter, recorreu-se aos seguintes fluxo e diagramas para verificar os comportamentos da aplicação:

1. Diagrama Caso de Uso;
2. Diagrama de Classe;
3. Diagrama de Sequência.

Fora considerado o ator utilizador, que efetivamente será o responsável pela utilização do sistema. Diferentes níveis de abstração foram identificados nos diagramas ilustrados nas figuras 15 e 16 [Pre05].

- **Caso de Uso 1: Autenticar - Atores (Admin $->$ Utilizador)**;
    1. Para ter acesso ao sistema, deve informar o nome de utilizador e a senha e clicar "validar". A inclusão do validar, deve-se sobretudo pelo facto de o sistema dispor de classe responsável, pela verificação e validação das credenciais do utilizador, para o mesmo poder efetuar a autenticação no sistema com sucesso. Classe essa que será discutido mais para frente na secção 3.8 relativo a segurança.
- **Caso de Uso 2: Listar Clientes - Ator (Utilizador)**;
    1. O utilizador faz o "clique"no menu cliente, a partir da seleção, ocorre o processo de listagem de todos os clientes existentes na base de dados;
        - Conforme os requisitos funcionais do sistema, o mesmo poderá efetuar às seguintes operações (adicionar novo cliente, editar, excluir, gerar pdf e ordenar os clientes.);
        - Caso contrário, às funcionalidades estarão indisponíveis.
- **Caso de Uso 3: Listar Contadores - Ator (Utilizador)**;
    1. O utilizador faz o "clique"no menu contador, a partir da seleção, ocorre o processo de listagem de todos os contadores existentes na base de dados.
        - Conforme os requisitos funcionais do sistema, o mesmo poderá efetuar às seguintes operações (adicionar novo contador, editar, excluir, gerar pdf e ordenar os contadores.);
        - Caso contrário, às funcionalidades estarão indisponíveis.
- **Caso de Uso 4: Listar Coordenadas - Ator (Utilizador)**;
    1. O utilizador faz o "clique"no menu coordenada, a partir da seleção, ocorre o processo de listagem de todas as coordenadas existentes na base de dados.
        - Conforme os requisitos funcionais do sistema, o mesmo poderá efetuar às seguintes operações (adicionar nova coordenada, editar, excluir, gerar pdf e ordenar as coordenadas.);
        - Caso contrário, às funcionalidades estarão indisponíveis.
- **Caso de Uso 5: Listar Utilizadores - Ator (Utilizador)**.
    1. O utilizador faz o "clique"no menu utilizador, a partir da seleção, ocorre o processo de listagem de todos os utilizadores existentes na base de dados.

- Conforme os requisitos funcionais do sistema, o mesmo poderá efetuar às seguintes operações (adicionar novo utilizador, editar, excluir, gerar pdf e ordenar os utilizadores.);
- Caso contrário, às funcionalidades estarão indisponíveis.

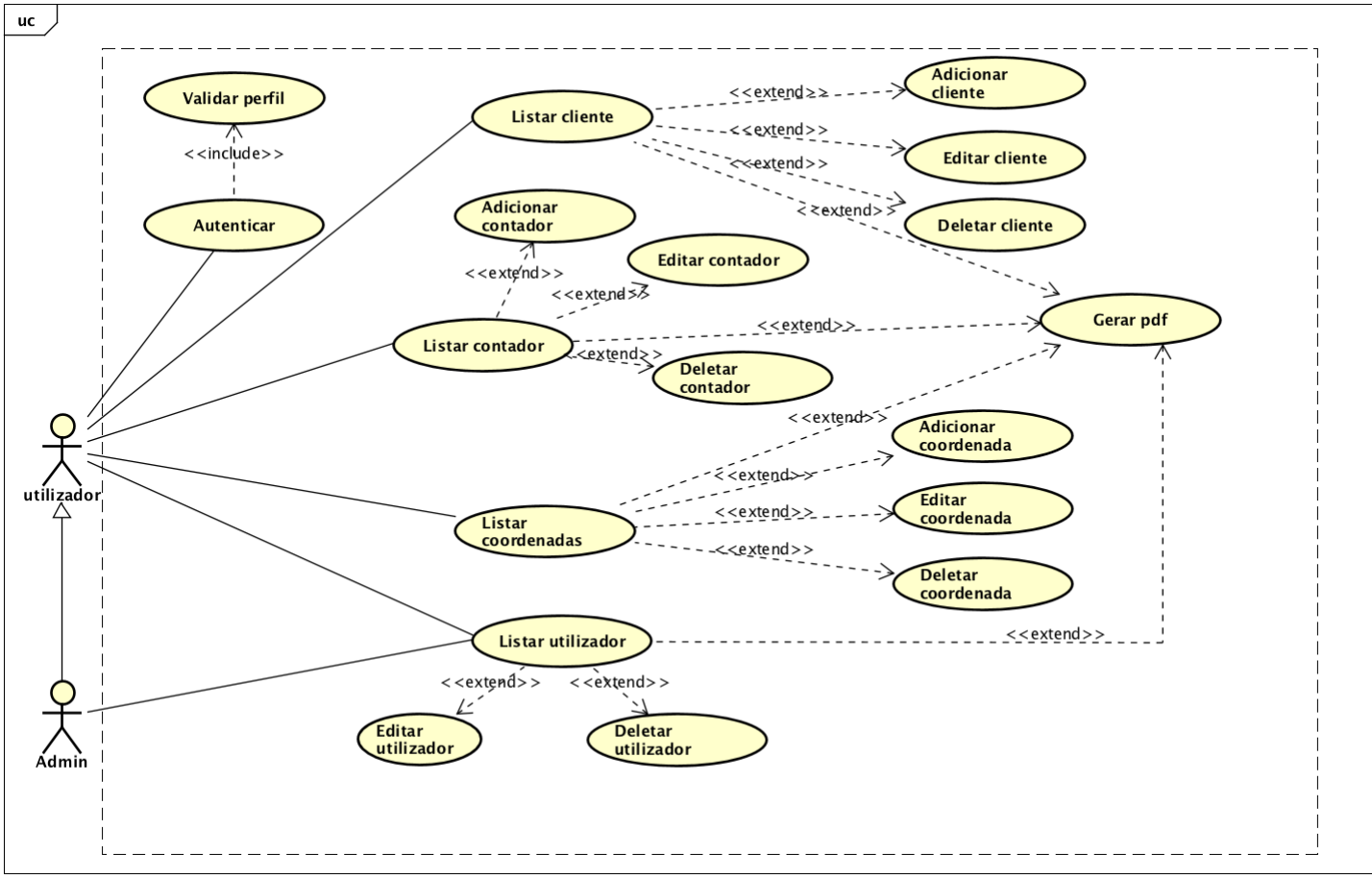

Figura 15: Diagrama Caso de Uso GeoMeter, Fonte: (Autor)

3.6.2 Diagrama de Sequência

O diagrama fornece ilustração a sequência de mensagens trocadas entre objectos do sistemas. São usados principalmente para representar a estrutura de controle entre os objectos e Modelação de interação de várias categorias de processos no ciclo de vida de um sistema conforme é ilustrado na figura 17.

3.7 Síntese

A análise do protótipo e a sua avaliação foram cruciais para desenhar o mapa mental dos requisitos desta dissertação e as iterações nela contida. Doravante, após entender os componentes do sistema, a sua estrutura e os vários diagramas que dele fazem parte, podemos nos preparar para expandir tal sistema. Da mesma forma, saber quais tecnologias, softwares e bibliotecas será adotada para ajudar nesse contexto.

O capítulo seguinte é dedicado à fase de implementação com detalhes mais ínfimos do processo de coleta de dados georreferenciados que, por sua vez, é contínuo e impactam nos requisitos. Com recurso ao Google Maps é possível realizar recolha de referência geográficas que, ao mesmo tempo, representa um macro importante para persistir os valores na base de dados por meio de representação dos pontos em latitude e longitude manualmente.

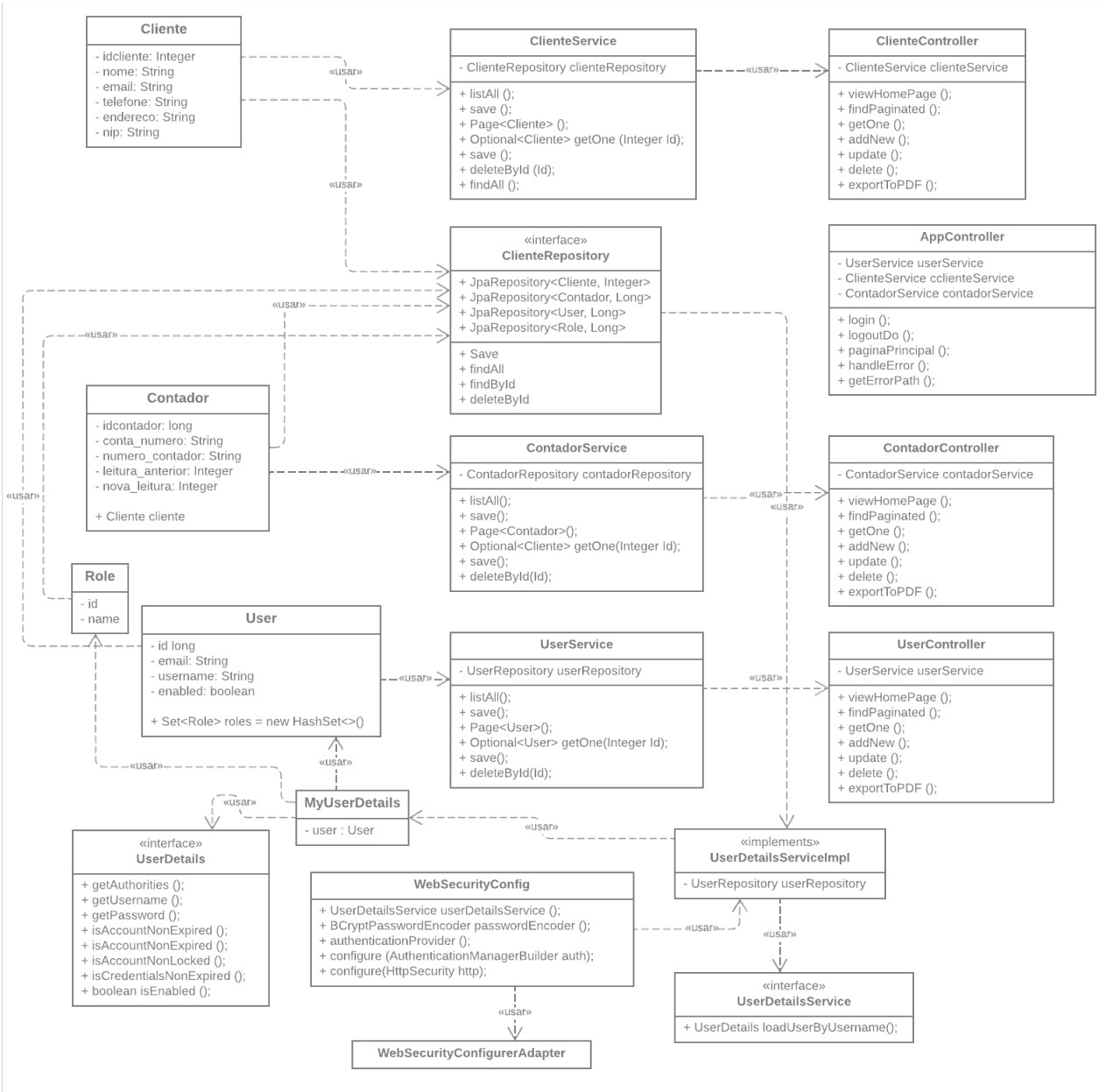

Figura 16: Diagrama Classe GeoMeter, adaptado codejava.net

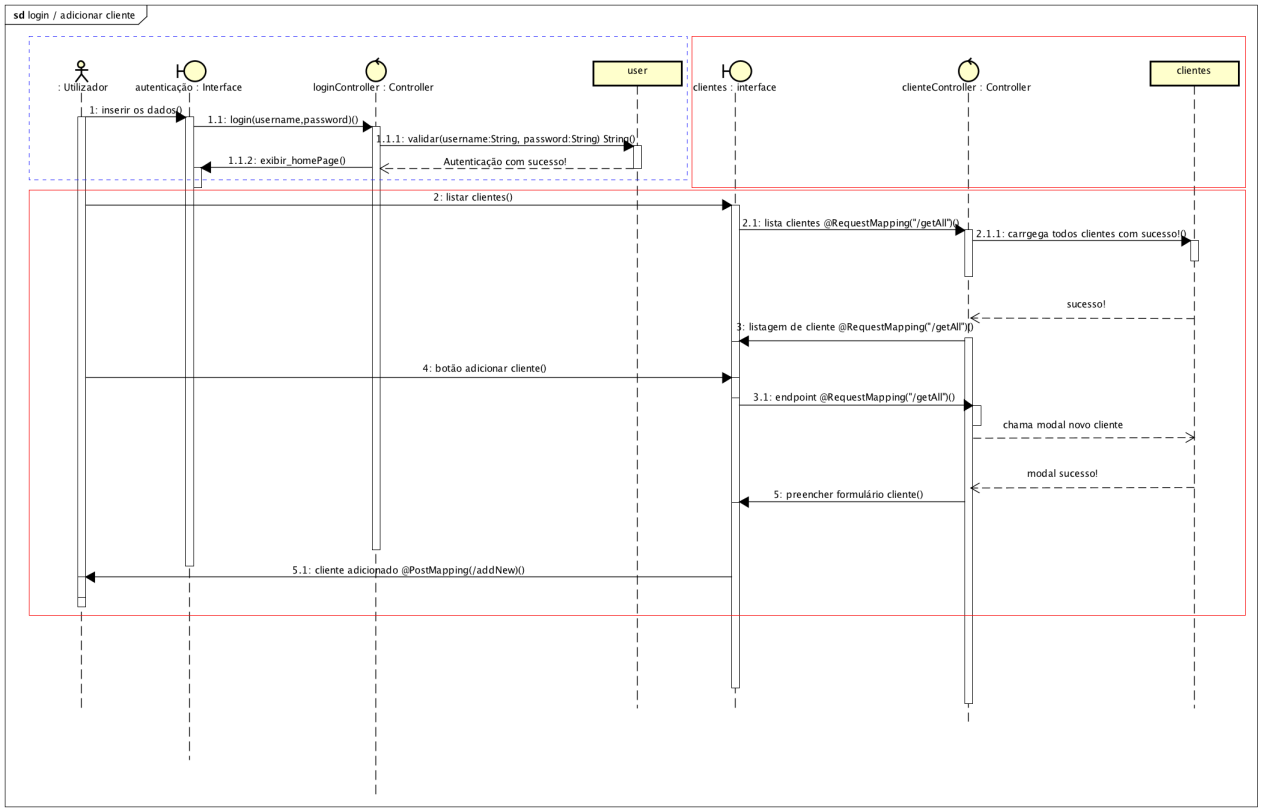

Figura 17: Diagrama de Sequência GeoMeter, Fonte (Autor)

4

Implementação

Ao final do desenho inicial do protótipo exposto no capítulo anterior, além de lapidar os requisitos e identificar as tecnologias a serem utilizadas, eis a hora de expandir a dissertação, com novas funcionalidades e alterações, para obter melhor interação do utilizador com a aplicação, que visa propiciar uma experiência agradável cogitado com a usabilidade na realização das tarefas. Este capítulo descreve o processo e vários aspetos diferentes envolvidos na conceção e desenvolvimento desta plataforma.

4.1 Visualização dos Dados Georreferenciados

Do ponto de vista de visualização dos valores persistidos na base de dados, é imprescindível reter aspetos de visualização e distribuição dos contadores no mapa. Porém, para que esse efeito tenha resultado, é necessário alguns ajustes que passo a enumerar:

- Criar o arquivo com extensão ***.html*** na VIEW conforme ilustrado na figura 18;
- Adicionar a tag e atributo *XML* ao arquivo que definem a execução lógica do *template Thymeleaf*;

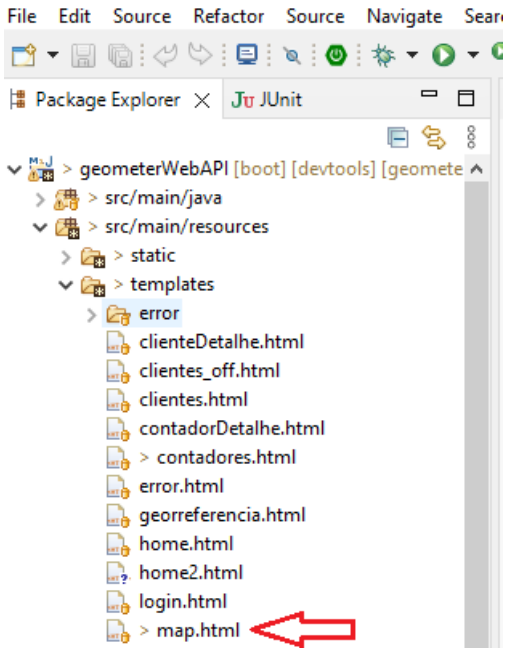

Figura 18: Ficheiro Thymeleaf do Mapa, Fonte (Autor)

```
<html xmlns:th="http://www.thymeleaf.org"
xmlns:sec="https://www.thymeleaf.org/thymeleaf-extras-springsecurity5">
```

- Incluir o pacote CSS responsável pela organização de estilos figura 19;

```
<style>
      #map {
         position: relative;
         top: -2em;
         left: 6em;
         width: 235em;
         height: 108em;
         margin-left:5em;
         margin-top:6em;
      }
      .leaflet-popup-content-wrapper {
         font-family: "Helvetica Neue", Arial, Helvetica, sans-serif;
         font-size: 1.09rem;
         line-height: 1.5;
      }

 </style>
```

Figura 19: Ficheiro CSS do Mapa, Fonte (Autor)

- Por fim, o Script da figura 20 contém todas as informações para a localização que se aspira apresentar no mapa, tendo sido formada pela seguinte estrutura:
  - API OpenStreetMap com método L.tileLayer() para aceitar URL da API e obter automaticamente o mapa de blocos necessário para a posição atual do utilizador;
  - Método addTo() que, por sua vez, é o atributo que representa a utilização do bloco API OpenStreetMap;

– Método setView() que, tem o par de latitude e longitude e um valor de zoom determinado por **[0.2249, 6.6179], 11**, representado pela superfície de São Tomé e Príncipe.

```
<script>
    var map = L.map('map').setView([0.2249, 6.6179], 11);
    L.tileLayer('https://tile.openstreetmap.org/{z}/{x}/{y}.png', {
        maxZoom: 19,
        attribution: '&copy; <a href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>'
    }).addTo(map);
    //map.dragging.disable();
    $.get("/contador_1/lista", function (data) {
        for (var i = 0; i < data.length; i++) {
            var marker = L.marker([data[i].georreferencia.latitude, data[i].georreferencia.longitude], {
                title: data[i].georreferencia.descricao
            }).addTo(map);
            var popupContent = "<span class='popup' > <b>Cliente: </b>  <a href=\"/clientes/detalhes/"
                                                + data[i].cliente.idcliente + "\"> "
                                                + data[i].cliente.nome + "</a><br>" +
                    "<b>Email :</b>            " + data[i].cliente.email + "<br>" +
                    "<b>Nº Conta:</b>          " + data[i].conta_numero + "<br>" +
                    "<b>Nº Contador:</b>       " + data[i].numero_contador + "<br>" +
                    "<b>Leitura Anterior:</b>  " + data[i].leitura_anterior + "<br>" +
                    "<b>Nova Leitura :</b>     " + data[i].nova_leitura + "<br>" +
                    "<b>Amperagem :</b>        " + data[i].amperagem + "<br>" +
                    "<b>Ramal :</b>            " + data[i].ramal + "<br>" +
                    "<b>Latitude:</b>          " + data[i].georreferencia.latitude + "<br>" +
                    "<b>Longitude:</b>         " + data[i].georreferencia.longitude + "<br>" +
                    "<b>Descrição:</b>         " + data[i].georreferencia.descricao + "</span>";
            marker.bindPopup(popupContent);
            marker.on('mouseover', function (e) {
                this.openPopup();
            });
        }
    });
    // L.circle([0.30693,6.65394], {radius: 10000}).addTo(map);
</script>
```

Figura 20: Trecho de Código JavaScript da Biblioteca LEAFLET, Fonte (Autor)

Para o mapa estar disponível e exibir fielmente os valores da Base de Dados para o utilizador, é necessário primeiramente definir um elemento "div" e informar o local onde se pretende que o mesmo seja apresentado. Porém, a figura 20 reflete o que é mais crucial para efetivação da visualização do mapa, ao traduzir a ideia genuína de como é processado a integração e carregamento dos valores georreferenciados mediante técnicas do JavaScript disponíveis na página oficial do *Leaflet*.

De forma não ordenada, mas obrigatória, será necessário criar as classes Controlo e Serviços que será responsável por receber, processar e responder às solicitações HTTP provenientes da página HTML conforme a figura 21. Para comprovar todas as configurações feitas a nível de Back-end, executamos o

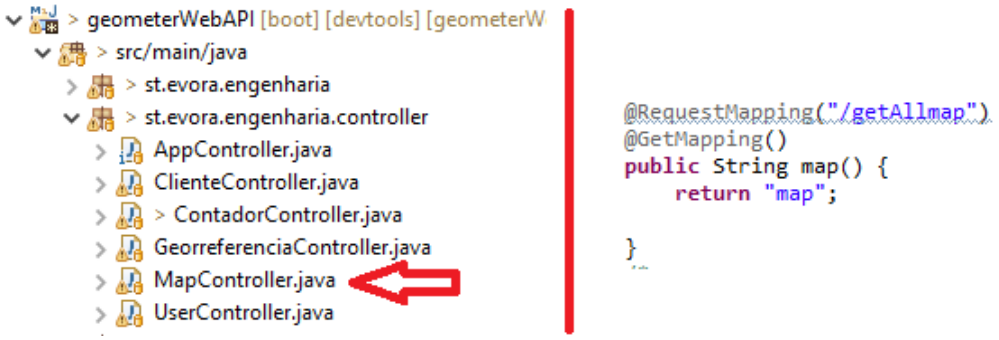

Figura 21: Classe MapController.java, Fonte (Autor)

Spring Boot e passamos a URL parametrizada no Controlo "/getAllmap" associado ao ficheiro Thymeleaf de nome "map" sem a extensão, pois já é reconhecido pelo compilador Spring. Teremos a seguinte saída conforme a figura 22.

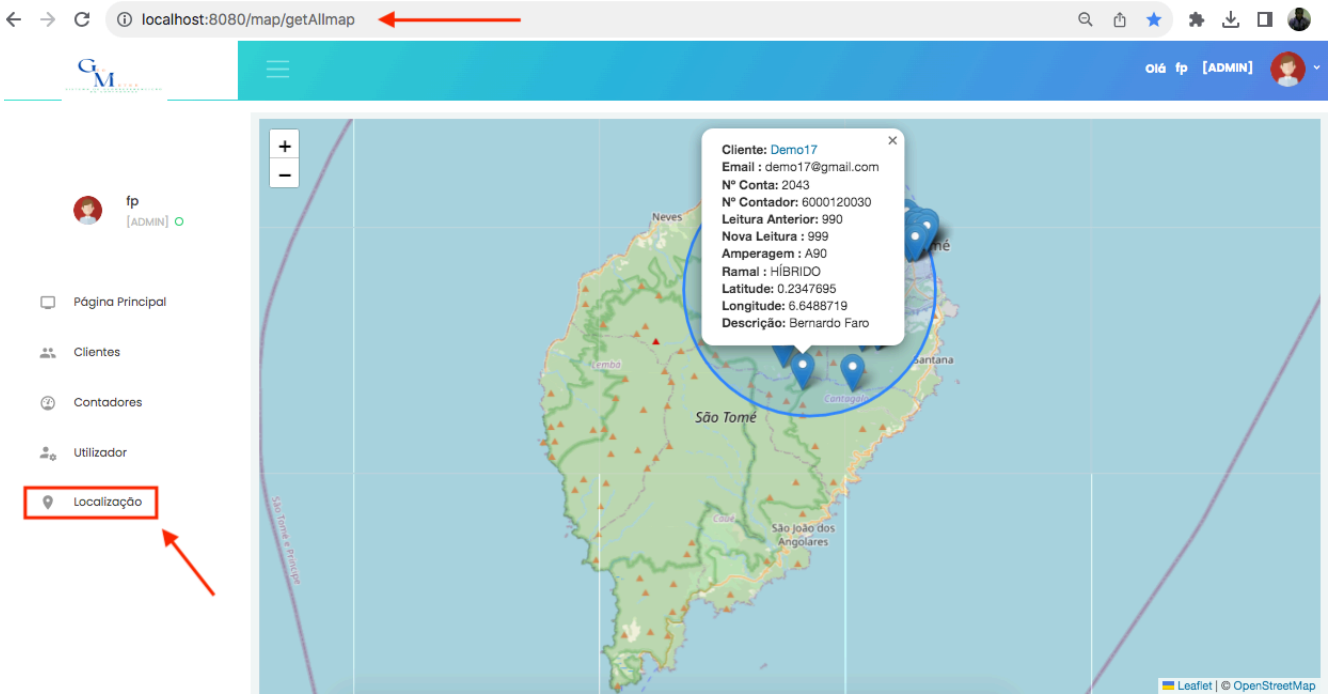

Figura 22: Resultado do Mapa Leaflet dos Dados Georreferenciados, Fonte (Autor)

4.2 Camada UI/Métodos

Responsável por acelerar o processo de usabilidade dos utilizadores com a aplicação, por via de adoção do modelo *Thymeleaf*, e fundamentalmente garantir maior experiência da parte do utilizador. A exibição em navegadores é um dos aspetos cruciais do bom funcionamento da estrutura da aplicação conforme a figura 23. Com módulos para Spring Framework, uma série de integrações com as suas ferramentas favoritas e a capacidade de conectar a sua própria funcionalidade, o *Thymeleaf* é ideal para o desenvolvimento Web moderna em HTML5 JVM [Thy23]. Na sequência, ilustro as figuras 24, 25, 26 e 27, alguns métodos que fazem parte da construção, bem como de resposta de todas as requisições ao ***endpoint*** /clientes.

4.3 Camada APIs / Métodos

4.3.1 Metodologia de Desenvolvimento GeoMeter

A metodologia *Extreme Programming* (XP) foi assumida para o desenvolvimento da aplicação GeoMeter devido à sua leveza e facilidade de adaptação a diferentes níveis de desenvolvimento. Com o XP, várias novas versões de um sistema podem ser desenvolvidas, integradas e testadas em um único dia, independentemente do tamanho da equipe. Essa metodologia se destaca por sua excelente capacidade de adaptar rapidamente novos requisitos e alterações à aplicação. Ela é sustentada pelo desenvolvimento iterativo, permitindo que

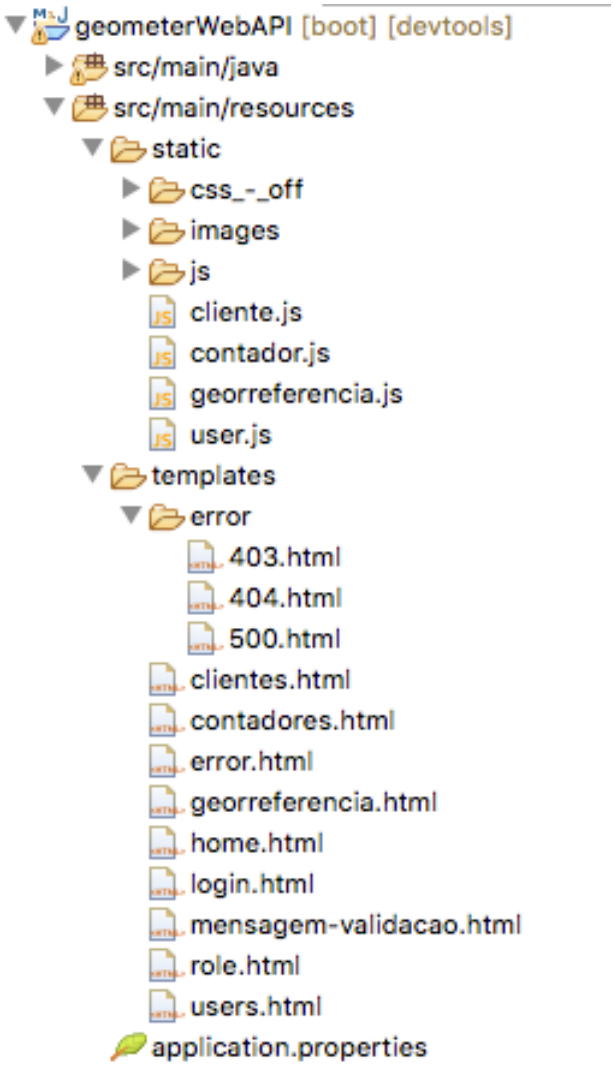

Figura 23: Estrutura da Camada VIEW, Fonte (Autor)

```
/* BEGIN MÉTODO LISTAR TODOS CLIENTES */

@RequestMapping("/getAll")
@GetMapping()
public String viewHomePage(Model model, @Param("keyword") String keyword) {
    List<Cliente> listClientes = clienteservice.listAll(keyword);
    model.addAttribute("listClientes", listClientes);
    model.addAttribute("keyword", keyword);

    return findPaginated(1, "nome", "asc", model);

}

/* END MÉTODO LISTAR TODOS CLIENTES */
```

Figura 24: Método Listar todos os Clientes na Base de Dados, Fonte (Autor)

```java
/* BEGIN endPoints adicionarNovoCliente */

@PostMapping("/addNew")
public String addNew(Cliente cliente, BindingResult result, RedirectAttributes attributes) {
    clienteservice.addNew(cliente);

    if (result.hasErrors()) {
        attributes.addFlashAttribute("mensagem", "Verifique os campos");
        return "redirect:/clientes/getAll";
    }
    attributes.addFlashAttribute("mensagem", "Cliente Adicionado com sucesso!");
    return "redirect:/clientes/getAll";
}

/* END endPoints adicionarNovoCliente */
```

Figura 25: Método Adicionar Clientes na Base de Dados, Fonte (Autor)

```java
/* BEGIN endPoints updateCliente */

@RequestMapping(value = "/update", method = { RequestMethod.PUT, RequestMethod.GET })
public String update(Cliente cliente, BindingResult result, RedirectAttributes attributes) {
    clienteservice.update(cliente);

    if (result.hasErrors()) {
        attributes.addFlashAttribute("mensagem", "Verifique os campos");
        return "redirect:/clientes/getAll";
    }
    attributes.addFlashAttribute("mensagem", "Cliente " + cliente.getNome() + " Cliente modificado com
    return "redirect:/clientes/getAll";

}

/* END endPoints updateCliente */
```

Figura 26: Método atualizar Clientes na Base de Dados, Fonte (Autor)

```java
/* BEGIN endPoints deleteCliente */

@RequestMapping(value = "/delete", method = { RequestMethod.DELETE, RequestMethod.GET })
public String delete(Integer Id) {
    clienteservice.delete(Id);
    return "redirect:/clientes/getAll";
}

/* BEGIN endPoints deleteCliente */
```

Figura 27: Método Excluir Clientes na Base de Dados, Fonte (Autor)

múltiplas novas versões sejam desenvolvidas, testadas, incrementadas e ajustadas em um curto espaço de tempo [Bec00] [Bec99] [AASW17].

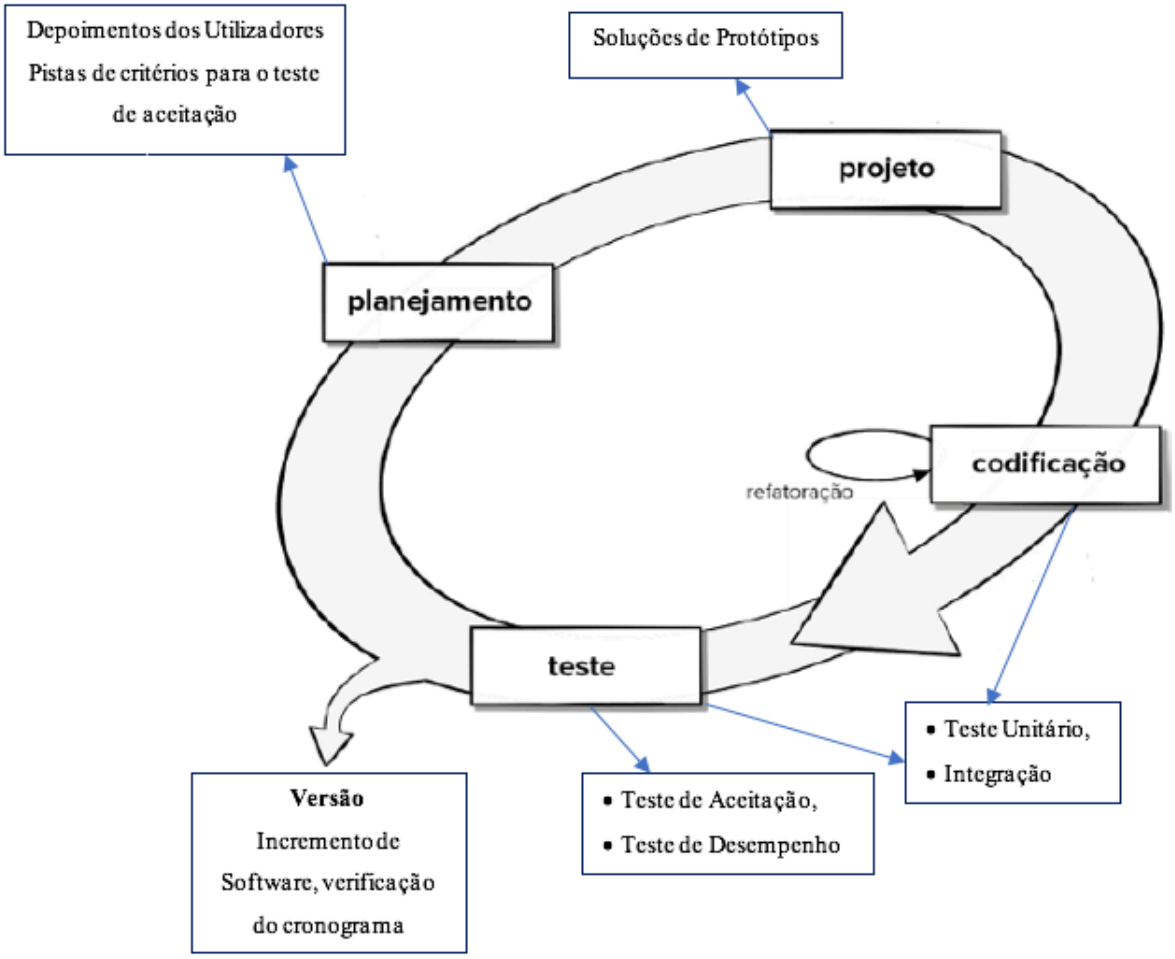

Figura 28: Processo de Construção da aplicação GeoMeter, Fonte (Adaptado [SOM])

A conceção da aplicação GeoMeter, surge do esforço na componente lidar com mudanças que é considerado um dos princípios fundamentais do XP, construído a partir da materialização e solicitações de alterações em diferentes etapas do projeto. Portanto, a abordagem XP aceita que, as mudanças acontecem em função da organização do sistema quando efetivamente essas mudanças devem acontecer.

Dessa maneira, para implementação criteriosa da aplicação GeoMeter foi essencial o cumprimento de algumas estratégias para sua melhoria, que se inicia, com um plano de atividade por via de entrevistas aos utilizadores, para promover um conjunto de "histórias" que conduz ao resultado. Cada depoimento é anotado com objetivo de gerar valor a regra de negócio do sistema. Através dos depoimentos foi possível criar um protótipo operacional da aplicação com compromisso básico que inclui, *deadline* das entregas do projeto e estimula projeto e a fase de transformar as histórias em solução de problemas relatados pelos utilizadores como garantia de qualidade e integração contínua a diversos testes com foco para disponibilização da versão do artefacto do software.

4.3.2 Base de Dados

Para armazenar dados relativos aos processos relacionados com contadores, clientes e a sua localização, optou-se por utilizar a Base de Dados MySQL para armazenar os dados relacionados aos processos envolvendo contadores, clientes e as suas localizações, devido à sua robustez, alta performance, escalabilidade e amplo suporte da comunidade. O estudo bibliográfico realizado a propósito do armazenamento, permitiu não apenas a aquisição de conhecimentos sobre MySQL, como também o desenvolvimento de competências e insights relevantes sobre o desenho e implementação de APIs RESTful, essenciais para qualquer desenvolvedor buscando criar aplicações Web eficientes e escaláveis [RAR13] [SZT12] [bU22].

A figura 29 ilustra o diagrama de entidade-relacionamento estendidos gerado a partir da ferramenta Workbench, de modo a obter uma visão mais nítida de como é feito a lógica do relacionamento entre as tabelas na base de dados relacional, assim como está configurado os atributos nas tabelas para armazenar os dados processados pela aplicação. Para realização da conexão com a base de dados relacional, no caso

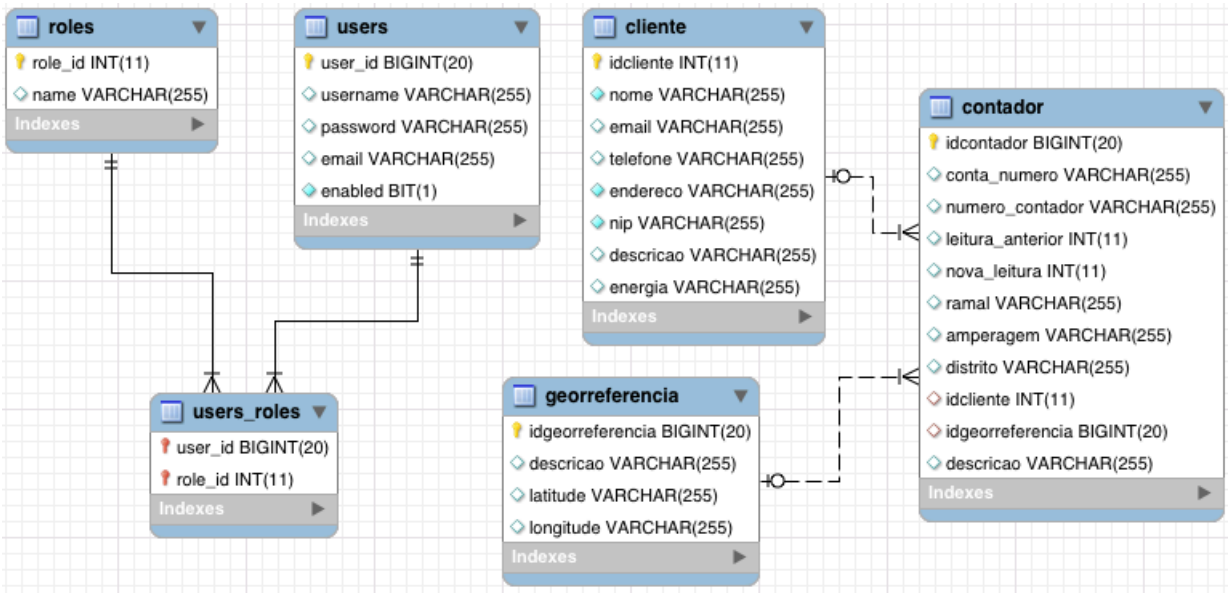

Figura 29: Diagrama EER Base de Dados GeoMeter, Fonte (Autor)

Mysql, recorreu-se à configuração do *application.properties*, conforme a figura 30.

```
# Database Details
spring.datasource.url = jdbc:mysql://localhost/dissertacao?useSSL=false&serverTimezone=UTC&useLegacyDatetimeCode=false
spring.datasource.username = root
spring.datasource.password = password
spring.datasource.driver-class-name = com.mysql.jdbc.Driver
spring.jpa.database-platform = org.hibernate.dialect.MySQL8Dialect

# Hibernate

#Tratamento de Erro
server.error.whitelabel.enabled=false

# Hibernate ddl auto (create, create-drop, validate, update)
spring.jpa.hibernate.ddl-auto = update

logging.level.org.hibernate.SQL=DEBUG
logging.level.org.hibernate.type=TRACE

# The SQL dialect makes Hibernate generate better SQL for the chosen database
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5InnoDBDialect
```

Figura 30: Mapeamento de Conexão Base de Dados, Fonte (Autor)

4.4 Segurança

O controle e tratamento do processo de autenticação, adota a implementação da classe nativa do *Spring Boot WebSecurityConfigurerAdapter* que fornece uma classe base conveniente para criar uma instância para configuração de segurança padrão do projeto Maven no *pom.xml* conforme ilustrado na figura 31. O modelo padrão do *WebSecurityConfigurerAdapter*, ilustrado na figura 32, com o uso de anotações

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-security</artifactId>
</dependency>

<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-test</artifactId>
    <scope>test</scope>
</dependency>
```

Figura 31: Dependência Spring-Security, Fonte (Autor)

@Configuration, @EnableWebSecurity mostra como promover o *WebSecurityConfig* através de requisição *HttpSecurity* [SS23] [Spi20] [Mul10] [Wal22]. A implementação da classe *WebSecurityConfigurerAdapter*, permite controlar acesso ao sistema, assim como, determinar a manipulação das funcionalidades dos módulos/menus existentes na aplicação, baseado na lógica do negócio, pois se trata de um aspeto crucial para manter o sistema seguro e confiável. Além disso, recorreu-se à classe nativa *BCryptPasswordEncoder*, para proteger as senhas dos utilizadores de acordo com técnicas criptográficas e conformidade com as boas práticas nesta área através interface *PasswordEncoder* [Bae23c].

A *BCryptPasswordEncoder* usa o algoritmo *bcrypt* amplamente suportado para fazer o hash das senhas. Para torná-lo mais resistente à quebra de senha, o *bcrypt* é deliberadamente lento. Como outras funções unidirecionais adaptáveis, ela deve ser ajustada para levar cerca de 1 segundo para verificar uma senha no seu sistema. A implementação padrão de *BCryptPasswordEncoder* usa força 10 conforme mencionado no Javadoc de *BCryptPasswordEncoder* [Bae23a] [Spr24].

A classe Java *PasswordGenerator* realiza esse processo, conforme podemos observar na figura 33.

Os requisitos para acessos e manipulação das funcionalidades do sistema por parte dos utilizadores, faz referência, a regra de negócio definida nos requisitos funcionais que envolve o processo de vincular níveis de permissões para cada utilizador do sistema conforme identificado na figura 32. No sistema, a tabela é mapeada na classe Java que segue a lógica do padrão MVC para o pacote (Modelo/Entidade), possa referenciar esses mesmos campos na forma de propriedades do objeto. O resultado da implementação de um mecanismo de requisição para uma lista de utilizadores é apresentado na figura 35. Nesta tela, o utilizador pode visualizar e realizar ações sobre os botões disponíveis na interface da listagem de utilizadores. Numa única tabela pode também filtrar os utilizadores por username, correio eletrónico, senha.

```java
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.authentication.dao.DaoAuthenticationProvider;
import org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBui
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

import st.evora.engenharia.serviceImpl.UserDetailsServiceImpl;

@Configuration
@EnableWebSecurity
public class WebSecurityConfig extends WebSecurityConfigurerAdapter {

    @Override
    protected void configure(HttpSecurity http) throws Exception {
        http.authorizeRequests()
                .antMatchers("/").hasAnyAuthority("USER", "CREATOR", "EDITOR", "ADMIN")
                .antMatchers("/new")
                .hasAnyAuthority("ADMIN", "CREATOR")
                .antMatchers("/edit/**")
                .hasAnyAuthority("ADMIN", "EDITOR")
                .antMatchers("/delete/**")
                .hasAuthority("ADMIN")
                .anyRequest()
                .authenticated()
                .and()
                .formLogin()
                .permitAll()
                .loginPage("/login")
                .and()
                .logout()
                .permitAll()
                .and()
                .exceptionHandling()
                .accessDeniedPage("/403");
    }
```

Figura 32: Configuração, filtros de autorização e autenticação HTTPSecurity, Fonte (Autor)

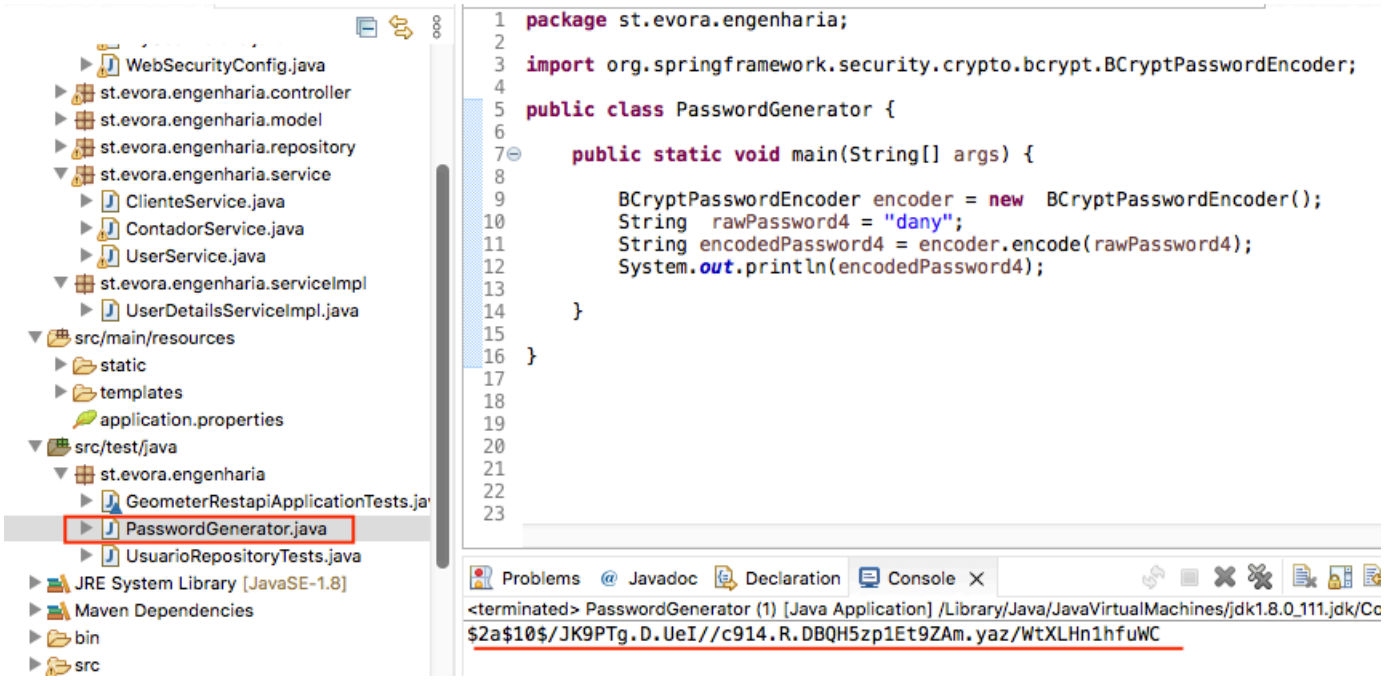

Figura 33: Classe PasswordGenerator, Fonte (Autor)

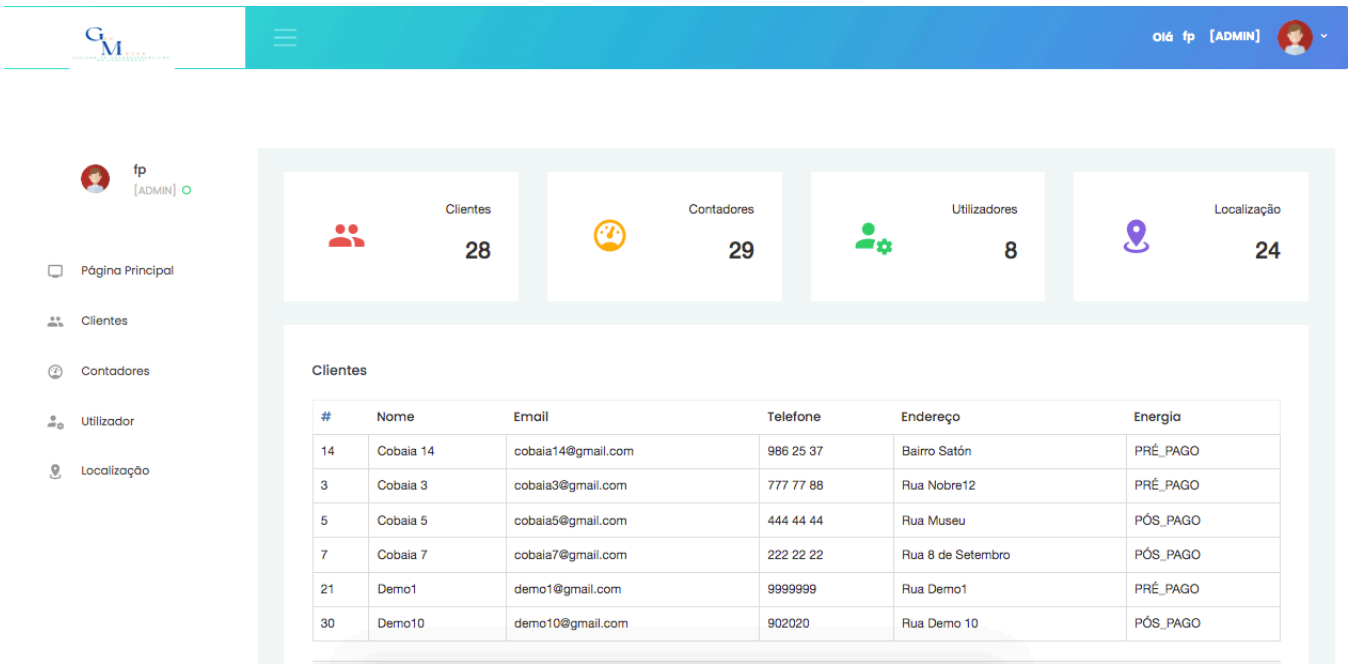

Figura 34: Página Inicial GeoMeter, Fonte (Autor)

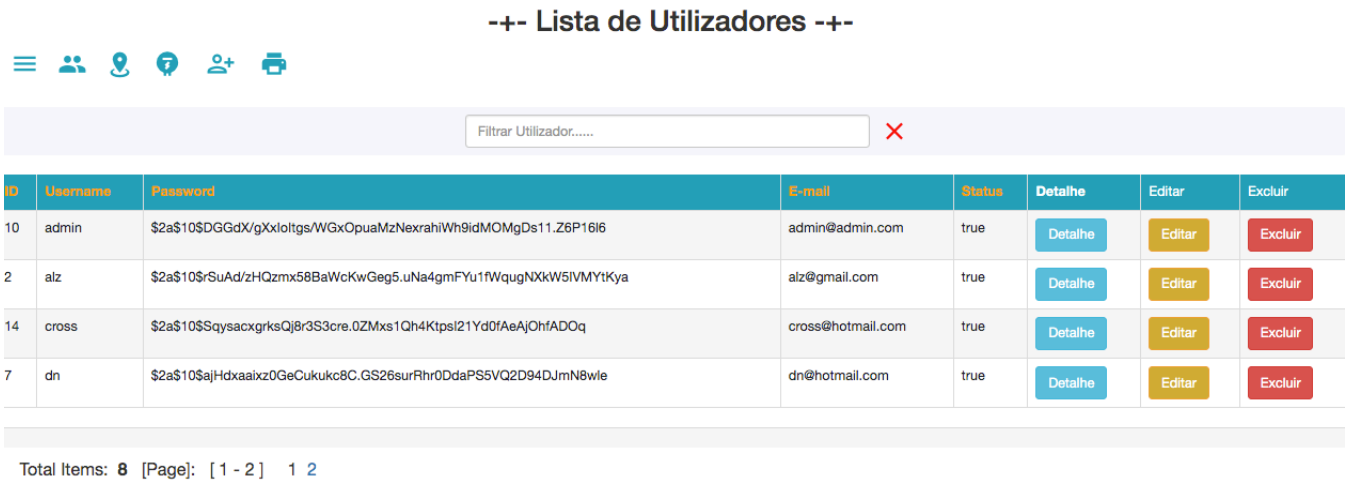

Figura 35: Lista de Utilizadores, Fonte (Autor)

A classe responsável pela renderização na exibição da busca por meio de *keyword* é a interface *UserRepository* figura 36, que estende ao *JpaRepository <User, Long>*, fornece métodos relacionados ao JPA, como liberar o contexto de persistência e excluir registos num lote, é possível usar recursos de organização e filtro conforme a interface ilustrada na figura 37 e recorre a métodos nativos como *List<User> listAll(String keyword)userRepository.findById(id)* [Bae23b].

```
public interface UserRepository extends JpaRepository<User, Long> {
    User findByUsernameAndPassword(String username, String password);

    @Query("SELECT u FROM User u WHERE u.username = :username")
    public User getUserByUsername(@Param("username") String username);

    @Query("SELECT u FROM User u WHERE CONCAT " + "(u.username, " + " u.email, " + "u.enabled) LIKE %?1%")
    public List<User> search(String keyword);

}
```

Figura 36: Disposição da interface UserRepository, Fonte (Autor)

Relativamente ao processo de adicionar um novo utilizador, os requisitos afirmavam que, um utilizador, com a correta permissão, deve conseguir adicionar e vincular o estado (ativo / inativo) outro possível utilizador.

Figura 37: Disposição Resultado do filtro

4.5 Síntese

O conteúdo unidimensional foi consolidado principalmente durante a dissertação. O processo de visualização dos dados georreferenciados na base de dados foi momento alto para implementação do sistema que, culmina com a configuração do arquivo HTML com execução lógica do motor de templates para produção da componente de vista/apresentação em frameworks Java server-side. Seguidamente ocorre de forma íntegra a integração do Leaflet com adoção de diferentes técnicas disponíveis da sua biblioteca para tornar a experiência com utilizador mais agradável. Exposição de métodos, camadas e metodologia de desenvolvimento da aplicação foram expostas com forte similaridade, a evidência de como ocorre o ciclo de entregas do GeoMeter. Escolha da base de dados, justificativa do seu uso e ilustração do diagrama de entidade e relação da base de dados, também foram aspetos tratados nessa secção, além dos filtros e autenticação HTTPSecurity feito a defesa a segurança da plataforma que, por hoje se torna um dos pilares mais relevantes de qualquer aplicação.

Componentes do sistema e modelação com a *Unified Modeling Language* foram elementos para validação dos diagramas de caso de uso, de sequência e de classe respetivamente com margem para identificar como ocorrem diferentes fluxos de processos. Dito isto, segue-se o capítulo 5, responsável por teste, verificação e validação.

5

Teste Verificação e Validação

*O teste de software é um elemento de um tópico mais amplo, muitas vezes conhecido como verificação e validação (**V&V**). Verificação refere-se ao conjunto de tarefas que garantem que o software implementa corretamente uma função específica. Validação refere-se a um conjunto de tarefas que asseguram que o software foi criado e pode ser rastreado segundo os requisitos do cliente.*

O capítulo 5, é responsável pela apresentação de qualidade estimada do software ora desenvolvido de forma pragmática com ilustração dos erros descobertos. Os resultados de diversos tipos de testes ao sistema serão apresentados e discutidos para entender se a solução proposta funciona conforme o esperado e atende aos objetivos específicos e requisitos definidos. Este processo de avaliação é dividido em três testes:

1. Teste unitário;
2. Teste de aceitação / regressão (UAT);
3. Teste de desempenho;

5.1 Teste Unitário

O teste unitário é o processo de testar os componentes de programa, como métodos ou classes de objeto. As funções individuais ou métodos são mais simples de componente. Os seus testes devem ser chamadas para essas rotinas com parâmetros diferentes de entrada [SOM].

Focaliza o esforço de verificação na menor unidade de projeto do software, ou seja, componente ou módulo. Usa guia e a descrição do projeto no nível de componente, caminhos de controle importantes são testados para descobrir erros nos limites do módulo. Esse teste tem a lógica interna de processamento e as estruturas de dados nos limites de um componente, também pode ser conduzido em paralelo para diversos componentes [Pre05].

O teste de unidade é uma categoria de teste de desenvolvimento e é normalmente usado pela equipa de programação, internamente, para encontrar problemas lógicos e melhorar a confiabilidade da implementação do sistema. A principal característica deste processo de teste é que, todos são focados em unidades individuais do sistema. Tem significado para componentes como classes, funções ou métodos, casos de teste isolados são projetados e executados para se certificar de que a funcionalidade do software funciona conforme o esperado [MSB11].

É um aspeto fundamental do teste de software, onde componentes ou funções individuais de um aplicativo de software são testados de forma isolada. Este método garante que cada unidade do software tenha o desempenho esperado.

Os testes unitários são normalmente automatizados e escritos por desenvolvedores usando vários Frameworks, como JUnit, NUnit ou pytest . Esses testes validam a correção do código, verificando se cada função ou método retorna os resultados esperados com base em entradas específicas. Fornecem classes de teste genéricas que pode estender para criar casos de teste específicos. Podem, então, executar todos os testes que se implementou e informar, muitas vezes por alguma interface gráfica, sobre o sucesso ou o fracasso dos testes. Um conjunto inteiro de testes frequentemente pode ser executado em poucos segundos; assim, é possível executar todos os testes cada vez que é feita uma alteração no programa.

A plataforma JUnit serve como base para lançar estruturas de teste na máquina virtual JVM. Também define a API *TestEngine* para desenvolver uma estrutura de teste executada na plataforma. Além disso, a plataforma fornece uma Console Launcher para iniciar a plataforma a partir da linha de comando e o JUnit Platform Suite Engine para executar um conjunto de testes personalizado usando um ou mais mecanismos de teste na plataforma. O suporte de primeira classe para a plataforma JUnit também existe em IDEs populares (IntelliJ IDEA, Eclipse, NetBeans, Spring Tool Suit e Visual Studio Code) e ferramentas de compilação como (Gradle, Maven e Ant)[Bec09] [Gar20] [Bin17].

Foi realizado vários testes, um deles é o teste para criar e popular a base de dados com um novo utilizador, com referência para declaração do método *Assertions*.

O teste para criar o novo utilizador na base de dados relacional ilustrado na figura 38, é visível um exemplo prático de implementação do teste unitário para execução da lógica que faz a chamada do verbo HTTP **POST** da camada *controller*. Este método de teste valida a resposta de inclusão a base de dados um ou vários utilizadores, conforme a parametrização de números de objetos que deseja incluir.

Figura 38: Disposição de Teste de Inclusão de Utilizador, Fonte (Autor)

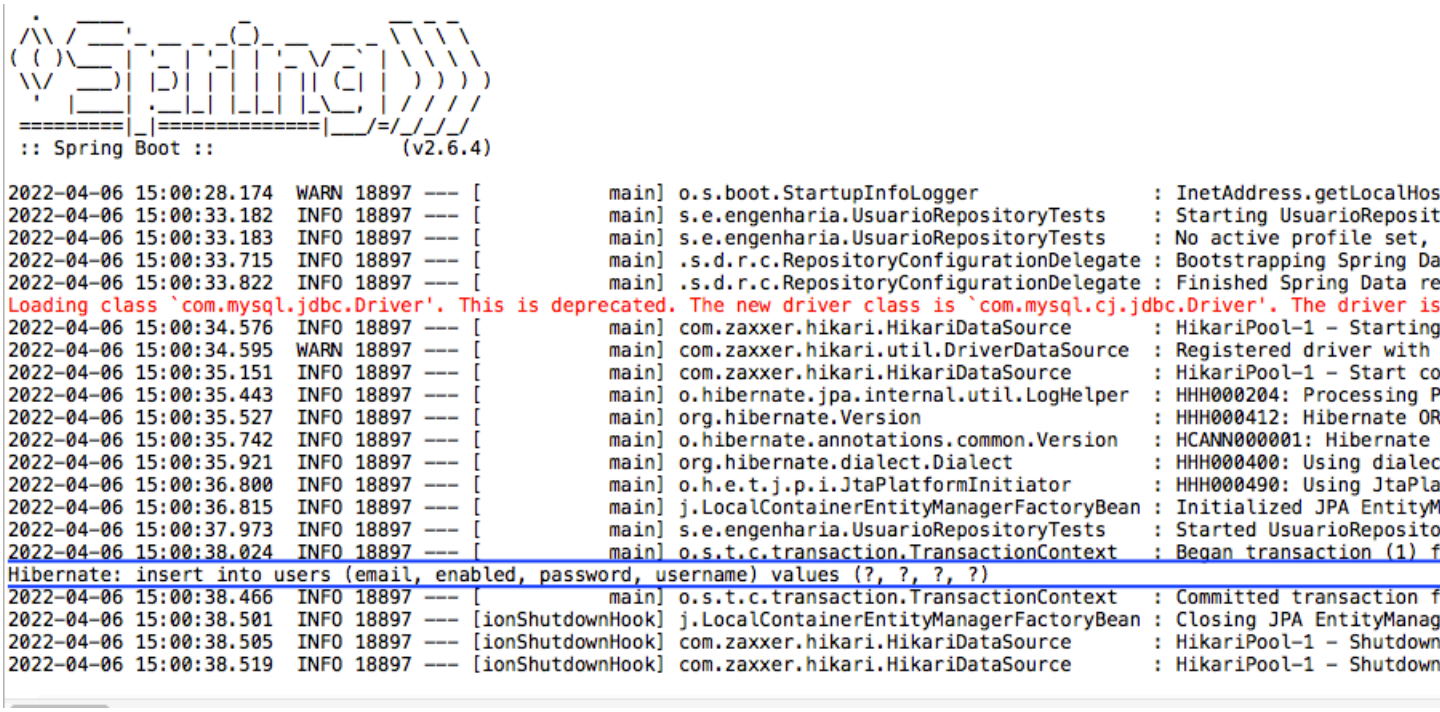

Figura 39: Resultado da Consola, Fonte (Autor)

user_id	username	password	email	enabled
1	[redacted]	$2a$10$[redacted]k...	[redacted]@gmail.com	1
2	[redacted]a	$2a$10$[redacted]I...	[redacted]@gmail.com	1
3	[redacted]	$2a$10$[redacted]...	[redacted]@gmail.com	1
4	[redacted]ea	$2a$10$[redacted]....	[redacted]@gmail.com	1
6	[redacted]r	$2a$10$[redacted]...	[redacted]@gmail.com	1
7	[redacted]	$2a$10$[redacted]K...	[redacted]@hotmail.com	0
10	[redacted]in	[redacted]min	[redacted]@admin.com	1
11	[redacted]aia	cobaia	cobaia@cobaia.com	0
12	[redacted]n	nolan	[redacted]@hotmail.com	0
13	cross	cross	cross@hotmail.com	1

Figura 40: Resultado da Tabela Utilizador da Base de dados GeoMeter, Fonte (Autor)

Também é possível verificar o resultado do teste *JUnit* ilustrado na figura 39, através do *output* da consola do *Spring Tool IDE*, assim como o estado da coluna afetada na Base de Dados Relacional observado na figura 40.

Em relação aos testes efetuado ao ***endpoint*** http://localhost:8080/users, responsável por renderizar um modelo com a listagem de todos os utilizadores da base de dados, é necessário a chamada JSON correto para se obter sucesso. Em cada chamada do teste, também é necessário definir algumas configurações importante, como o caso da instância da classe *MockMvc*, para poder executar solicitações *HTTP* de teste.

A figura 41, representa a captura dos registos dos utilizadores na base de dados, para isso houve necessidade de implementar a injeção da anotação *@MockBean*.

Figura 41: Pré-Teste da Classe UserControllerTests, Fonte (Autor)

Podemos observar alteração na estrutura do diretório através do pré-teste e pós-teste realizado ao endpoint /users, para obter impressão da lista de todos os utilizadores persistido na base de dados relacional, para isso, foi necessário recorrer à instância do *mockMvc* e o método *perform* passando como parâmetro *get* vinculado ao *path* da URL conforme se pode visualizar nas figuras 42 e 43. Ao analisarmos o resultado da execução do teste, é notório a geração do ficheiro com extensão **.pdf**, arquivo este, que contem a lista de todos os utilizadores adicionados na base de dados relacional, valores esses que pode ser justificados através da figura 44 na interface Web.

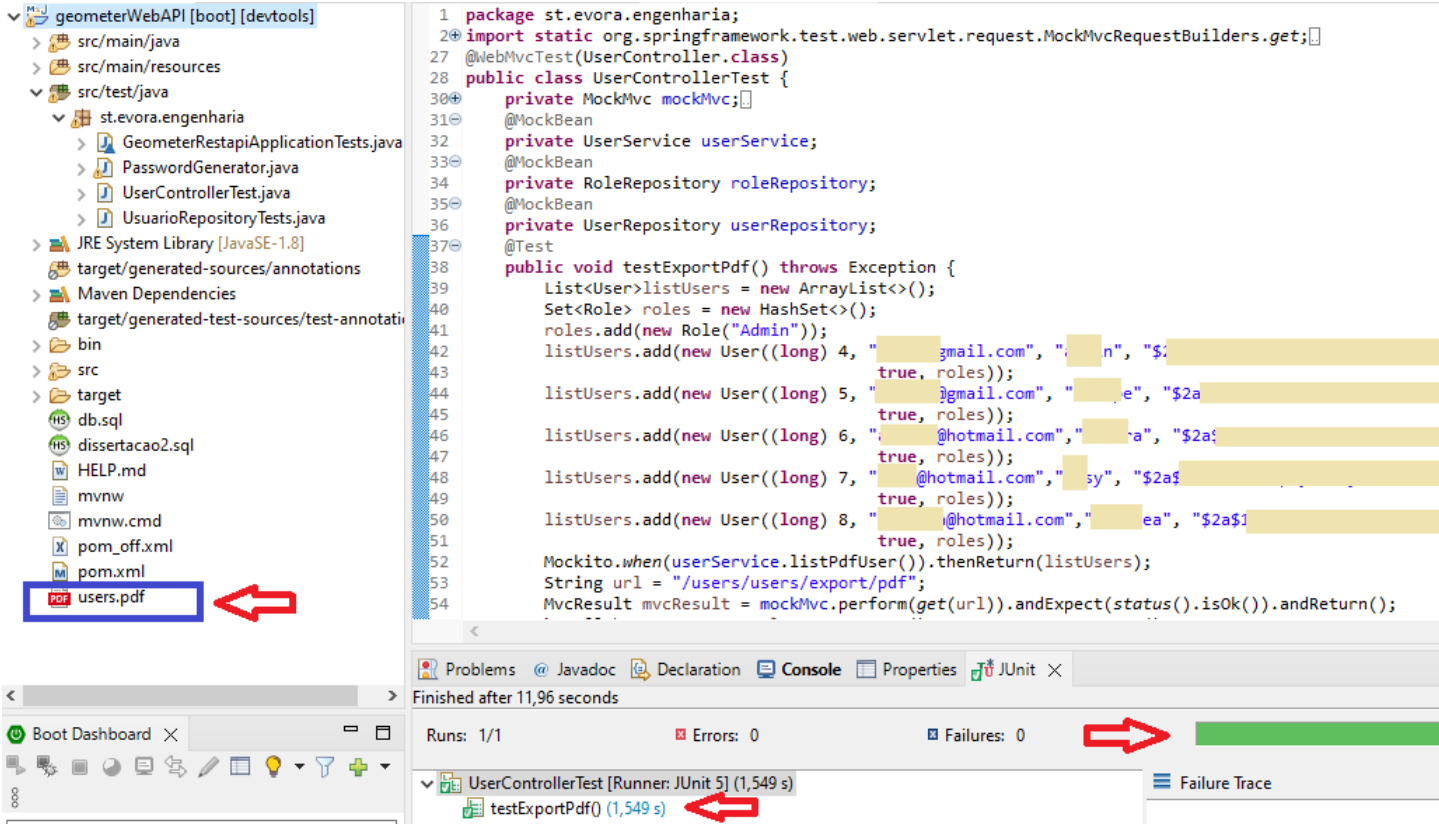

Figura 42: Pós-Teste da Classe UserControllerTests

Lista de Utilizadores

ID	E-mail	Username	Enabled
4	a	a	true
5	f	f	true
6	a	a	true
7	l	l	true
8	l	l	true

Figura 43: Saída do Ficheiro PDF dos Utilizadores.

user_id	username	password	email	enabled
4	n	yryHuJwLS31rbmR2	@gmail.com	1
5		Fo6iOrXtne7HIb9tea	@gmail.com	1
6	a	7ZGw6.Lx11P/efD3Dxa	@hotmail.com	1
7		r2nddxI9hf.09IQbgq36	@hotmail.com	1
8	e	?iWkSS3sYE00zCOBb56	@hotmail.com	1

Figura 44: Informações Web da Tabela Utilizadores

5.2 Teste de Aceitação

Os testes de aceitação são uma etapa fundamental no ciclo de vida do desenvolvimento de software. O seu principal objetivo é assegurar que, o sistema atenda aos requisitos e expectativas dos utilizadores finais e *stakeholders*. São um tipo de teste realizados para determinar, se uma aplicação satisfaz os critérios de aceitação especificados e para permitir que os utilizadores finais ou clientes validem se o produto de software atende às suas necessidades e requisitos antes produto final esteja alinhado com os requisitos e pronto para colocar em produção num ambiente real de produção [SOM] [Med24] [Gee24b].

Os testes de aceitação representam geralmente os testes funcionais de ponta a ponta, mantendo o produto ou uma aplicação no escopo Os testes de aceitação são orientados pelo requisito do utilizador final do sistema, ou aplicativo. Estes são então revistos por analistas de negócios designados, consultor de processos e designers de ponta a ponta. Isso ajuda a estimar o uso do requisito do utilizador do ponto de vista estratégico e definir o requisito de nível granular do sistema ou aplicativo a ser desenvolvido [Jac23]. Isso garante:

- O desenvolvimento satisfaz os critérios de testabilidade;
- Todas as expectativas dos clientes são consideradas e atendidas conforme os padrões.

Assim sendo, o teste de API envolve teste de interfaces de programação de aplicativos diretamente como parte da integração teste que determina se a lógica da API corresponde às expectativas de segurança, usabilidade, confiabilidade, testabilidade e escalabilidade. Como as APIs não possuem uma GUI, ela é testada na camada de mensagem. Para obter o tal desempenho esperado pela API, torna-se necessário envolver desafios de *Test Automation* (TA), especialmente ao adotar a metodologia Ágil, porque APIs próprios servem como a interface primária para o serviço ou lógica do aplicativo [Ani22].

5.2.1 Desafios do teste de API

Os desafios do teste de API incluem:

- Os principais desafios no teste de API da Web são a combinação de parâmetros, a seleção de parâmetros e o sequenciamento de chamadas;
- Verificar e validar a saída num sistema diferente é um pouco difícil para os testadores;
- A seleção e a categorização dos parâmetros devem ser conhecidas pelos testadores;
- A função de tratamento de exceção precisa ser testada;
- Conhecimento de codificação é necessário para testadores.

Até agora, existem muitas categorias de ambientes e dispositivos que usam soluções baseadas em API. Eles são aplicativos móveis, aplicativos da Web, aplicativos em nuvem, aplicativos de TV e dispositivos da Internet das Coisas (IoT). Eles conectam-se, integram e estendem sistemas de software numa arquitetura de API complexa [Ly18].

Os cenários da tabela 1, espelha um exemplo de implementação simples do teste realizado para solução proposta na dissertação. Foram realizados vários cenários de carga de solicitação para servidor e resposta para cliente para identificar o fluxo de comportamento da aplicação.

Foram testados 5 utilizadores, sendo 3 com níveis e perceção de informática avançada e 2 com níveis básicos, para, enfim, medir o grau de satisfação e funcionalidades das operações disponíveis na aplicação GeoMeter. Ficou assim verificado que, houve margem mínima relativamente ao tempo de execução de uma operação de utilizadores avançados para os básicos, resultados esses que, em nada nos remete inferir que existe alguma lacuna de satisfatibilidade por parte dos utilizadores conforme a tabela 1.

Resultado do Teste de Aceitação.			
Descrição Cenários	**Resultado**	**Tempo Execução**	**Comentários.**
O utilizador logado com perfil ***#admin*** realiza login no sistema para realizar operações de inserir um novo Cliente e novo Contador.	O Sistema deve, conforme os requisitos funcionais, permitir que o utilizador possa incluir um novo Cliente e Contador.	Foi programado no cronómetro, o tempo para execução da mesma tarefa.	Sucesso em ambas às operações!
O utilizador logado com perfil ***#admin***, faz login no sistema para realizar operações de, **edição** e **exclusão** de um ou mais Cliente, ou Contador.	O Sistema deve, conforme os requisitos funcionais, permitir que o utilizador possa realizar às mesmas operações.	Foi programado no cronómetro, o tempo para execução da mesma tarefa.	Sucesso em ambas às operações!
O utilizador logado com perfil ***#admin***, faz login no sistema para criar utilizadores.	Novo utilizador é criado.	É disponibilizado a lista com o novo utilizador criado.	Sucesso nas operações criar utilizador!
O utilizador logado com perfil ***#admin***, faz login no sistema para exportar ficheiro ***.pdf*** que contem a lista de utilizadores.	O *endpoint* /users/users/export/pdf, deve responder à requisição do utilizador para realizar a operação.	Tempo de resposta para essa operação, foi satisfatória.	Ficheiro ***users.pdf***, é disponibilizado para o utilizador.

Tabela 1: Tabela do Teste de Aceitação, Fonte (Autor)

5.3 Teste de desempenho

Os testes de desempenho precisam ser projetados para assegurar que o sistema processe a carga a que se destina. Isso normalmente envolve a execução de uma série de testes em que aumenta a carga até que o desempenho do sistema se torne inaceitável [SOM].

O teste de desempenho é projetado para testar o desempenho em tempo de execução do software no contexto do sistema integrado [Pre05].

O último estágio de teste realizado na aplicação, é o teste de desempenho, este permite classificar o grau de eficiência, operacionalidade e resposta que o sistema oferece aos utilizadores finais. Esta fase, é normalmente executada quando a aplicação está pronta para entrar em produção, com objetivo de medir algumas métricas de desempenho com base em cargas de solicitações produzidas pelo utilizador, de modo a obter os valores considerados positivos no horizonte de taxa de transferências de dados relativamente ao volume de trabalho. Assim sendo, é listado a tabela 2 de desempenho realizados em função dos cenários estabelecidos para execução de determinada tarefa por parte do utilizador, com objetivo de verificar se

o comportamento de sistema manterá de acordo ao esperado pelo utilizador. Conforme os testes de

Tabela de Desempenho	
Cenários	**Resultados**
Login no sistema GeoMeter, independentemente do ***perfil***, a tela de login do sistema GeoMeter deve carregar e apresentar de igual forma para todos os utilizadores.	O endpoint **/login** é executado no servidor local, e carrega todos os ficheiros necessários para apresentar ao utilizador a tela de login, está tarefa demora cerca de **13,062** segundos.
Página principal, para acesso aos menus e às operações disponíveis na aplicação.	O endpoint **/home**, é apresentado na saída da consola Spring Tool Suite, o valor da execução em **14,39** segundos.
O utilizador, após 15 minutos de inatividade, é pedido que faça de novo o login.	O sistema deve devolver e renderizar a tela de login para que o utilizador volte a logar no sistema.
O utilizador logado com perfil ***#admin***, faz login no sistema para exportar ficheiro ***.pdf*** que contem a lista de utilizadores.	O *endpoint* /users/users/export/pdf, deve responder à requisição do utilizador para realizar a operação.

Tabela 2: Tabela do Teste de Desempenho, Fonte (Autor)

desempenho realizado para diferentes cenários, deve-se tomar em conta os seguintes fatores:

- Quantidade de registos armazenados na Base Dados;
- Caraterística do Hardware responsável pela execução da aplicação.

O primeiro cenário realizado com a ferramenta JMeter, foi para o endpoint /login, com objetivo de recolher informações de tempo e resposta a solicitação para o sistema. A tabela 3, representa alguns valores extraídos do teste. Podemos sintetizar algumas informações conforme o quadro resumo do teste efetuado ao *endpoint*

Quadro de Desempenho Endpoint /login						
N.º Utilizador	**Tempo de Carga**	**Min/ms**	**Max/ms**	**Avg/ms**	**Latência/ms**	**Req/seg**
1	1348	1348	1348	1348	645	0,24
10	419	247	611	476	341	2,49
50	5478	1024	5576	4103	4850	2,87
100	10233	848	10233	6784	10018	3,15

Tabela 3: Tabela do Teste de Desempenho Login, Fonte (Autor)

/login. Existe alguma discrepância de valores apresentados para diferente parâmetro de utilizador, destaque maior para o tempo de carregamento e a latência, a variação de 1 utilizador para 100, é expressivo com um horizonte de 1348 para 10233 e latência varia entre 645 e 10018ms. Esses valores podem ser enquadrado como falso/positivo, mediante os vários testes, podemos obter valores mais abaixo do que apresentado inicialmente, por outro lado, podemos justificar que, esses valores surgem devido existência de códigos css, js até mesmo métodos desnecessários, que exigem o carregamento para apresentar a página de login.

A fase seguinte do teste de desempenho, envolve os cenários internos da aplicação, chamados testes de "stresse" inclui avaliação de números de acessos ao serviço REST para diferentes operações como o número de utilizadores que executam as mesmas operações. A fase de testes de desempenho realizados incluiu a avaliação da resposta do sistema quando submetido a uma demanda acima do normal. Esses testes, são geralmente projetados para medir a resposta do sistema em tais situações e detetar quaisquer anomalias

que possam ser causadas por a falta de recursos disponíveis para processar os pedidos. Para suportar a execução de casos de teste com diferentes cenários, foi utilizada a ferramenta *JMeter*, sendo uma aplicação Java de código aberto capaz de simular carga de trabalho num sistema ao avaliar a sua resposta [RDM19]. A tabela 4, representa os testes de carga realizado ao ***endpoint*** /clientes/getAll com diferente número

Quadro de Desempenho Endpoint /clientes/getAll						
N.º Utilizador	**Tempo de Carga**	**Min/ms**	**Max/ms**	**Avg/ms**	**Latência/ms**	**Req/seg**
1	9	34	34	34	9	7,41
10	31	27	55	36	5	2,76
50	982	58	1078	506	954	7,94
100	2142	85	2369	1149	1828	8,88

Tabela 4: Tabela do Teste de Desempenho listar Clientes, Fonte (Autor)

de utilizadores. Pode-se verificar o balanceamento de carga obtidos para os valores de latência, tempo de carregamento, tempo mínimo, tempo máximo, médio e requisição por segundo obtidos a partir dos registos da base de dados, ou seja, quanto maior for o volume de registos na base de dados, maior será o tempo de carregamento, podemos observar na tabela entre (1-100) número de utilizador, a variação de valores de tempo de carregamento e latência, para 1 utilizador conectado, o tempo foi de 9 milissegundos e igual para a latência, o mesmo não acontece quando temos 100 utilizadores conectado, o valor de carregamento dispara para 2142 milissegundos tanto quando e a latência para 1828. Esse comportamento está associado a vários fatores de construção do resultado da árvore de resposta da requisição da API.

Contudo, deve-se interpretar bem os valores da latência, é representado pelo valor em milissegundos, que decorre entre o processo do framework JMeter envia a solicitação http e obtém uma resposta inicial da API, ou seja, o tempo de carregamento é alternativamente chamado do tempo de amostra, não é o número de milissegundos que o servidor levou a resolver completamente a solicitação, logo, o tempo de carregamento é o tempo total que cada requisição leva para concluir todo o processo, faz a chamada do Threads, executa a lógica e devolve a resposta. Isso varia conforme o método responsável para carregar os registos na Base de Dados. Dado o baixo volume de registo de contadores na Base de dados, podemos

Tabela de Desempenho Endpoint /contadores/getAll						
N.º Utilizador	**Tempo de Carga**	**Min/ms**	**Max/ms**	**Avg/ms**	**Latência/ms**	**Req/seg**
1	44	44	44	44	19	5,77
10	27	25	69	36	4	2,68
50	215	27	439	151	147	11,19
100	1919	26	1997	1044	481	10,68

Tabela 5: Tabela do Teste de Desempenho listar Contadores, Fonte (Autor)

verificar o baixo valor obtido no tempo de carregamento e latência, resultado estes que tendem alterar conforme a quantidade de utilizadores conectados na aplicação conforme a ilustração da tabela 5.

Podemos concluir que, dos três testes realizados para diferentes ***endpoints***, obteve-se o resultado satisfatório, pois não foi detetado nenhum erro operacional e todas as solicitações foram concluídos com sucesso. Relativamente aos valores de velocidade, para solicitações, foram registados os valores mínimo e tempo máximo para uma solicitação e carregamento de registos, bem como os valores médios. O maior tempo registado foi de 11,19 segundos para uma solicitação de contadores/getAll, com a parametrização de 50 utilizadores, valor esse que teve uma redução para 10,68 segundos quando ocorreu o acesso de 100 utilizadores tentando usar a aplicação em simultâneo, e embora pareça um pouco longo, pode acontecer situação desse género, pois o número alvo de utilizadores parece normal para o comportamento do sistema.

Diversos fatores podem estar envolvidos nesse comportamento, recursos de hardware, recursos de rede, categoria de equipamento que realiza a solicitação, etc. Por fim, foi registado o número de solicitações atendidas por segundo e com essa informação foi possível concluir que quando o número de utilizadores chega a 50, o desempenho do aplicativo tende a manter constante, porque o valor de solicitações por segundo permanece entre 10 e 11 respetivamente.

Em resumo, os testes de desempenho são aqueles em que submetemos o sistema a uma avaliação de carga para "stress" ou desempenho para avaliar se os resultados correspondem com o esperado, garantindo assim a qualidade dos sistemas. Contudo, existem algumas estratégias para alcançar o desempenho das aplicações sob condições normais de uso, tempo de resposta, número de transações por minutos, utilizadores logados em simultâneos, etc. Todo teste é iniciado com carga baixa e vai aumentado gradualmente de acordo com diferentes ambientes e variáveis e surgem algumas questões como: quantas transações serão suportadas por minutos quando aumentarmos o número de utilizadores logados simultaneamente, exemplo comum de 50 para 100.

O objetivo, é garantir que o sistema permaneça estável ou a funcionar de maneira satisfatória após o período de uso. Serve para o engenheiro de teste e a equipa de desenvolvimento medir o grau de requisições que a aplicação suporta. Pela norma, a equipa de teste deve executar com uma carga constante e manter por horas a mesma, para obter o resultado que será o objeto de análise através da variável tempo vs resposta dada pelo sistema.

Já, no que concerne ao teste de "stress", serve para verificar o software sob condições extremas de uso. Grande volume de transações, muitos registos armazenados na base de dados para carregar, utilizadores simultâneos a usar recursos do servidor para requisitar algum serviço, enfim, picos excessivos de carga em curto tempo.

Ao longo do teste da aplicação hora desenvolvida, foi notória essa fase, sinal esse que nos remete a classificar a ferramenta JMeter como, uma ótima para implementar o teste de desempenho. Vale lembrar que apenas foi realizado o teste com recurso HTTP, mas existem outros testes suportado pela ferramenta JMeter como: testes para servidores Web, HTTP, SOAP, Base de Dados via JDBC, LDAP, JMS, MAIL (POP , SMTP e IMAP), etc.

Destacamos algumas melhorias detetadas nos testes realizados na máquina local, pois a mesma dispõe de poucos recursos físicos, assim sendo, para obter um maior desempenho na aplicação deve-se ter em conta os seguintes valores como atributos-chave para maximização da resposta a requisição de qualquer ***endpoints***:

- Diminuir tamanho ou número de ficheiro como CSS, imagens, JavaScript, ícones, pois os mesmos podem causar o aumento de tempo para tratar e disponibilizar ao utilizador;
- Otimizar o algoritmo, ou seja, implementar poucas linhas de códigos e/ou métodos para resolver uma operação.

5.4 Síntese

Um total de três testes foram feitos a diversos *endpoints* com diferentes finalidades. O primeiro designado por teste unitário que, recaiu na seleção do *framework JUnit* na sua versão 5 e instância da classe *MockMvc* com anotação *@MockBean* para verificar a criação de um novo utilizador na base de dados relacional através da chamada do verbo HTTP POST da camada *UserController* ao *endpoint http://localhost:8080/users* e geração do ficheiro *users.pdf* sem fazer o uso da *GUI* (*graphical user interface*). O teste foi executado com sucesso, com 2 (duas) evidências respetivamente ilustradas na figura 4.1

e 4.3, podemos observar a saída da consola *Spring* o resultado da criação do utilizador em **0,442s** e o valor persistido na base de dados.

O segundo teste decorre por meio de simulados de usabilidade para verificar as funcionalidades e o comportamento do sistema. Para tal, os testes de aceitação representam geralmente os testes funcionais de ponta a ponta, mantendo produto ou uma aplicação no escopo orientados pelos requisitos do utilizador final do sistema. Isso ajuda a identificar e formular heurística de usabilidade. A validação do teste de aceitação teve sucesso de acordo relatório elaborado aos diferentes tipos de utilizadores.

Por fim, foi realizado o teste de desempenho com enfoque para exposição do tempo de execução de cada funcionalidades do sistema. O teste de desempenho permite classificar o grau de eficiência, operacionalidade e resposta que o sistema oferece aos utilizadores finais. Esta fase, é normalmente executada quando a aplicação está pronta para entrar em produção, com objetivo de medir algumas métricas de desempenho com base em cargas de solicitações produzidas pelo utilizador, de modo a obter os valores considerados positivos no horizonte de taxa de transferências de dados relativamente ao volume de trabalho. Assim sendo, é listado a quadro geral 4.2 de desempenho realizados em função dos cenários estabelecidos para execução de determinada tarefa por parte do utilizador, com objetivo de verificar se o comportamento de sistema manterá de acordo ao esperado.

O capítulo seguinte, é o último desta dissertação, apresenta algumas considerações gerais sobre o trabalho realizado onde é enumerado alguns pontos de melhorias futuras do projeto.

6

Conclusão

Este capítulo apresenta o silogismo da conceção de todo o trabalho realizado na dissertação com identificação de vários resultados obtidos. O capítulo 6 conclui este documento com o delineamento de trabalhos futuro a ser desenvolvido, de modo a incrementar melhorias a proposta sistema.

A tecnologia REST Spring Boot, mostrou-se eficiência para construir uma solução de Sistema de Informação Georreferenciado, para dar resposta ao problema de georreferenciação de contadores instalados nos pontos de consumo pela empresa responsável pelo tratamento, produção e distribuição da eletricidade em todo o território de São Tomé e Príncipe.

Os recursos da Google Maps serviram para recolher informações relativas às coordenadas geográficas, com referência para o uso de graus decimais, que permitem guardar localizações na base de dados para serem manipuladas a partir de uma interface Web.

Foi possível incorporar funcionalidades de visualização espacial, representando localizações registadas na base de dados sobre mapas exibidos com a biblioteca Leaflet. Por fim, o sistema de informação georreferenciado desenvolvido, mostrou-se eficiente para dar resposta ao problema de localização de contadores instalados nos pontos de consumo.

6.1 Síntese Geral da Dissertação

A dissertação começa com uma breve explanação da empresa e o ramo de negócio em que a mesma se enquadra. Após isso, fez-se a introdução com destaque para o ponto central a que se destina o desenvolvimento do projeto, no caso prático para criar uma Aplicação Web com uso de tecnologia REST para dar resposta a questão de georreferenciação de contadores de eletricidade instalados em todo o território santomense e dispor informações numa interface amigável de manipulação, como em formato JSON para ser interpretado e consumido em forma de serviços para futuras aplicações. Diversas abordagens de sistemas que utilizam técnicas para georreferenciar objetos no espaço foram mencionados no capítulo estado de arte, onde se procurou identificar e frisar especialmente esse aspeto, por ser o ponto fulcral pela qual a dissertação se enquadra. Por conseguinte, foi importante descrever como é feito a recolha de dados de coordenadas geográficas no sistema GeoMeter para localizar contadores instalados.

Os principais objetivos e os requisitos delineados no início deste trabalho foram cumpridos. Um protótipo funcional da Aplicação Web proposta, contendo recursos de requisição e resposta de diversos *endpoints* e acesso à base de dados, foi disponibilizado à empresa responsável pela produção e fornecimento de eletricidade do país.

O protótipo desenvolvido contou com linguagem de modelação unificada para identificar diferentes cenários de funcionalidades em caso de usos, assim como, abstração de alguns diagramas representativos de várias operações desde administrar clientes, contadores, coordenadas, utilizadores onde se recorreu à arquitetura *MVC* para organizar, separar e delegar responsabilidades em diferentes camadas.

Para fins de segurança recorreu-se à classe nativa do Spring Boot o WebSecurityConfigurerAdapter com auxílio do algoritmo BcryptPasswordEncoder para codificar e proteger a password com objetivo de aprovar os acessos e conceder permissão para realizar determinada ação no sistema.

Com esses recursos, os operadores podem realizar uma vasta gama de ações na aplicação, a começar com adicionar um cliente, coordenada no sistema e continua a fazer a associação dos mesmo a um contador, por fim pode gerar o pdf e perceber qual cliente pertence o mesmo contador está instalado. Nesse desenvolvimento, um dos principais requisitos para a solução foi sanada, pois, os operadores constataram que o trabalho de recolha de leitura dos contadores mais facilitado devido à exatidão do ponto de instalação dos mesmos, de acordo com critérios de negócio definidos.

Por fim, foi realizada uma avaliação do sistema com realização de testes para mensurar a qualidade do sistema através da ferramenta Jmeter para diferentes cenários de testes. O teste unitário, serviu para testar os componentes de programa com métodos e classes de objeto com foco para verificação na menor unidade de projeto do software, ou seja, componente ou módulo. Foi testado a inserção de um novo utilizador na base de dados com declaração do método *assertions* assim como para gerar um ficheiro **pdf** com todos os utilizadores no diretório raiz da aplicação.

O teste de aceitação/alfa ou UAT, auxiliou para verificar funcionalidades da aplicação da parte do cliente ou *front-end*, com objetivo de eliminar erros e omissões dos requisitos do sistema além de mensurar ações necessárias para verificar se todas as expetativas dos clientes foram cumpridas na perspetiva da interface do utilizador, e por fim foi feito o teste de desempenho, com objetivo de assegurar o comportamento e o fluxo de carga a que se destina o sistema.

Os resultados foram positivos, pois não foram encontrados defeitos significativos ao testar as atividades simuladas em produção. A secção seguinte aponta linhas de continuação possíveis para trabalho futuro.

6.2 Trabalho Futuro

- Desenvolver o módulo *Progressive Web App (PWA)*, que será responsável pela entrega Web que visa fornecer uma experiência de utilizador como uma aplicação nativa num dispositivo móvel Offline com suporte a notificações push [San16];
- Desenvolver uma aplicação em Android para a localização dos contadores de eletricidade com um dispositivo Smartphone, para os utilizadores poderem visualizar remotamente a posição do contador no mapa;
- Implementar o módulo para obter de forma automática as coordenadas do ponto de instalação dos contadores;
- Reforçar o canal de segurança de comunicação dos dados nos tablet/PC cliente e permitir que os mesmos percorram numa VPN com objectivo de garantir proteção dos dados em trânsito, com utilização de algoritmo criptográfico simétrico (ou chave secreta), que se traduz em técnica de criptografia onde a mesma chave é utilizada para encriptar e desencriptar os dados [Net23].

Em última análise, este projeto se revelou um desafio gratificante no campo da engenharia de software, com ênfase na construção de uma aplicação Web adotando o modelo de desenvolvimento de microserviços. Foi focado em criar e disponibilizar diferentes modelos de apresentação, oferecendo ainda vários modelos de apresentação na visualização de dados espaciais para os utilizadores, com destaque para flexibilidade e escalabilidade dos serviços oferecidos.

Ao longo de todo o trabalho, as várias etapas referentes ao estudo, definição de requisitos, desenvolvimento de protótipo, implementação e teste da solução foram cumpridas. Esses passos foram pertinentes para construir e entregar uma solução de software estável e de qualidade.

Bibliografia

[AASW17] Faiza Anwer, SSSM Aftab, S Shah Muhammad Shah, and Usman Waheed. Comparative analysis of two popular agile process models: extreme programming and scrum. *International Journal of Computer Science and Telecommunications*, 8(2):1–7, 2017.

[AHAS21] Tariq N Ataiwe, Israa Hatem, and Hisham MJ Al Sharaa. Digital model in close-range photogrammetry using a smartphone camera. In *E3S Web of Conferences*, volume 318, page 04005. EDP Sciences, 2021.

[aJHU24] CSSE at Johns Hopkins University. CSSEGISandData COVID-19. `https://github.com/CSSEGISandData/COVID-19`, 2024. [Online; accessed 01-janeiro-2024].

[Ani22] Maurício Aniche. *Effective Software Testing: A developer's guide*. Simon and Schuster, 2022.

[Ant24] Sebastian Anthony. Think gps is cool? ips will blow your mind. *an Extreme Tech digital news article dated*, 2024.

[App18] Karthik Appigatla. *MySQL 8 Cookbook: Over 150 recipes for high-performance database querying and administration*. Packt Publishing Ltd, 2018.

[Arc19] Andrea Arcuri. Restful api automated test case generation with evomaster. *ACM Trans. Softw. Eng. Methodol.*, 28(1), jan 2019.

[Bae23a] Baeldung. HashingaPasswordinJava. `https://www.baeldung.com/java-password-hashing`, 2023. [Online; accessed 11-April-2023].

[Bae23b] Baeldung. JpaRepository. `https://www.baeldung.com/spring-data-repositories`, 2023. [Online; accessed 30-March-2023].

[Bae23c] Baeldung. PSpring Security Form Login. `https://www.baeldung.com/spring-security-login`, 2023. [Online; accessed 29-March-2023].

[Bas12] Len Bass. *Software architecture in practice*. Pearson Education India, 2012.

[BBBN12] Sunil L Bangare, Seema Borse, Pallavi S Bangare, and Shital Nandedkar. Automated api testing approach. *International Journal of Engineering Science and Technology*, 4(2), 2012.

[Bec99] Kent Beck. Embracing change with extreme programming. *Computer*, 32(10):70–77, 1999.

[Bec00] Kent Beck. *Extreme programming explained: embrace change*. addison-wesley professional, 2000.

[Bec04] Kent Beck. *Programação Extrema (XP) explicada*. Bookman; 1ª edição, 02 2004.

[Bec09] Kent Beck. Junit: A cook's tour. *IEEE Software*, 16(5):87–89, 2009.

[Bin17] Andrew Binstock. Junit 5 architecture and testengine api. *Java Magazine*, 2017.

[BJ22] Michael Brown and Emily Johnson. Optimizing restful apis with mysql workbench and spring boot. In *Proceedings of the 2022 International Conference on Software Engineering*, pages 102–113. IEEE, 2022.

[BR21] A. Brown and M. Rodriguez. Thymeleaf: Integrating templating and business logic in modern java applications. *Computer Science Review*, 34(1):78–92, 2021.

[BT21] Silvia Botros and Jeremy Tinley. *High Performance MySQL*. "O'Reilly Media, Inc.", 2021.

[bU22] Sufyan bin Uzayr. *Mastering MySQL for Web: A Beginner's Guide*. CRC Press, 2022.

[CDB15] RN Carneiro, S Damiao, and MJ Benoliel. Water safety plans at epal's water supply system-tool to prioritize investments and mitigation actions. *Water Science and Technology: Water Supply*, 15(5):1106–1114, 2015.

[CDPEV19] Gerardo Canfora, Massimiliano Di Penta, Roberto Esposito, and Maria Luisa Villani. Testing web apis in agile development: Challenges and approaches. In *Proceedings of the 19th International Conference on Web Engineering*, pages 45–60. Springer, 2019.

[CHSH17] Iuliana Cosmina, Rob Harrop, Chris Schaefer, and Clarence Ho. *Pro Spring 5: An in-depth guide to the Spring framework and its tools*. Apress, 2017.

[CI14] Paul Crickard III. *Leaflet. js essentials*. Packt Publishing Ltd, 2014.

[Cor23a] Oracle Corporation. MySQL Documentation. `https://dev.mysql.com/doc/`, 2023. [Online; accessed 21-November-2023].

[Cor23b] Oracle Corporation. MySQL Documentation. `https://www.mysql.com/products/workbench/`, 2023. [Online; accessed 01-janeiro-2023].

[Cor23c] Oracle Corporation. Mysql workbench. `https://www.mysql.com/products/workbench/`, 2023. [Online; accessed 25-November-2023].

[CV22] Vasco Fitas da Cruz Carla Varanda, Patrick Materatski. Normas para elaboração de Dissertação Mestrado em Engenharia Agronómica. `https://dspace.uevora.pt/rdpc/bitstream/10174/31803/1/Manual_Dissertacao_CV_18032022.pdf`, 2022. [Online; accessed 25-march-2024].

[CZ23] Li Chen and Wei Zhang. Advanced orm techniques with hibernate and spring data jpa. In *Proceedings of the 2023 International Conference on Software Engineering*, pages 215–223. ACM, 2023.

[Dev11] Hartati Deviana. Penerapan xml web service pada sistem distribusi barang. *Generic*, 6(2):61–70, 2011.

[DEV23] DEVMEDIA. Guia de ASP.NET Web API. `https://www.devmedia.com.br/guia/asp-net-web-api/38182`, 2023. [Online; accessed 22-November-2023].

[Dev24a] HeidiSQL Developers. Releases · heidisql/heidisql · github. *GitHub*, 2024.

[Dev24b] DevTools. Developer Tools. `https://docs.spring.io/spring-boot/reference/using/devtools.html#using.devtools`, 2024. [Online; accessed 11-august-2024].

[DSEB12] György Dán, Henrik Sandberg, Mathias Ekstedt, and Gunnar Björkman. Challenges in power system information security. *IEEE Security & Privacy Magazine*, 10(4):62–70, 2012.

[Ecl09] IDE Eclipse. Eclipse ide. *Website www. eclipse. org Last visited: July*, 2009.

[FB23] L. Fritz and M. Black. Modern java templating with thymeleaf: Best practices and performance considerations. *Journal of Software Engineering*, 12(3):45–67, 2023.

[FEEB03] Maydene Fisher, Jonathan Ellis, Jon Ellis, and Jonathan Bruce. *JDBC API tutorial and reference*. Addison-Wesley Professional, 2003.

[Fer21] A Ferreira. *Extending the Auditory Atlas of the City of Évora*. PhD thesis, Tese de mestrado. Faculdade de Ciências e Tecnologias da Universidade Nova …, 2021.

[Fer23] Correia J. Ferreira, A. Bupi: Digital transformation in property registration. 2023.

[FMK+23] C. R. Fol, A. Murtiyoso, D. Kükenbrink, F. Remondino, and V. C. Griess. Terrestrial 3d mapping of forests: Georeferencing challenges and sensors comparisons. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLVIII-1/W3-2023:55–61, 2023.

[Gar20] Dushyant Garg. Modern software development using junit 5, gradle, and continuous integration. *IEEE Software*, 37(3):26–31, 2020.

[Gee24a] Geeksforgeeks. Introduction to Spring Security and its Features. `https://www.geeksforgeeks.org/introduction-to-spring-security-and-its-features/`, 2024. [Online; accessed 17-julho-2024].

[Gee24b] Geeksforgeeks. Software Testing. `https://www.geeksforgeeks.org/software-testing-tutorial/`, 2024. [Online; accessed 15-julho-2024].

[Goo23] Google. Pesquisa por latitude e longitude. `https://support.google.com/maps/answer/18539?hl=pt-BR0&ref_topic=3092444`, 2023. [Online; accessed 01-janeiro-2023].

[Gur23] Guru99. Testing REST API Manually. `https://www.guru99.com/testing-rest-api-manually.html`, 2023. [Online; accessed 25-November-2023].

[HB08] Jomi F Hübner and Rafael H Bordini. Developing a team of gold miners using jason. In *Programming Multi-Agent Systems*, pages 241–245. Springer, 2008.

[HG23] K. Harris and P. Greene. Exploring thymeleaf in contemporary web application development. *International Journal of Web Technologies*, 15(2):123–135, 2023.

[HH12] Rob Harrop and Clarence Ho. *Pro Spring 3*. Apress, 2012.

[HS18] Maciej Haras and Thomas Skotnicki. Thermoelectricity for iot–a review. *Nano Energy*, 54:461–476, 2018.

[IBM23a] IBM. Java Development Kit. `https://www.ibm.com`, 2023. [Online; accessed 22-june-2023].

[IBM23b] IBM. Java Spring Boot. `https://www.ibm.com/cloud/learn/java-spring-boot`, 2023. [Online; accessed 19-November-2023].

[Jac23] Jacotech. Acceptance Test Driven Development. `http://www.jacotech.org/index.php/paper/paper/paperDetails/22`, 2023. [Online; accessed 29-March-2023].

[Jav24] Javatpoint. IDE do Spring Tool Suite (STS). `https://spring.io/tools`, 2024. [Online; accessed 11-august-2024].

[Jel24] Lucas Jellema. Adding GeoJSON features to a OpenStreetMap in Leaflet. `https://lucasjellema.medium.com/adding-geojson-features-to-a-openstreetmap-in-leaflet-b4b42db674d4`, 2024. [Online; accessed 01-janeiro-2024].

[JM22] Alice Johnson and Robert Miller. Mysql: An in-depth analysis of its evolution and current capabilities. In *Proceedings of the 2022 International Conference on Database Technology*, pages 50–65. ACM, 2022.

[Joa24] Joaopimentel1980. Geoapi Overview. `https://geoapi.pt/`, 2024. [Online; accessed 01-janeiro-2024].

[JW22] Alice Johnson and Mark Wilson. Integrating maven with modern ci/cd pipelines. In *Proceedings of the 2022 International Conference on Software Development*, pages 102–115. Springer, 2022.

[Kar21] Yavuz Selim Kart. Connecting to sql server using heidisql. *MSSQL Query*, 2021.

[KRTS13] Kari Karhu, Altti Repo, Ossi Taipale, and Kari Smolander. Experiences of using test automation tools in agile software development. *Proceedings of the 2009 ICSE Workshop on Automation of Software Test*, pages 20–25, 2013.

[KSBW16] Shameer Kunjumohamed, Hamidreza Sattari, Alex Bretet, and Geoffroy Warin. *Spring MVC: Designing Real-World Web Applications*. Packt Publishing Ltd, 2016.

[KSN18] Mike Keith, Merrick Schincariol, and Massimo Nardone. *Pro JPA 2 in Java EE 8: An In-Depth Guide to Java Persistence APIs*. Apress, 2018.

[lar]

[Let23] LetMe. LetMeMaps. `https://liteme.com.br/`, 2023. [Online; accessed 22-june-2023].

[Lew16] Mark Lewin. Leaflet. js succinctly. *Syncfusion, Inc*, 2016.

[LGMR05] Paul A. Longley, Michael F. Goodchild, David J. Maguire, and David W. Rhind. *Geographic Information Systems and Science*. Wiley & Sons, West Sussex, UK, 2nd edition, 2005.

[LW21] Alice Lee and David Wilson. Leveraging mysql workbench for effective database management in spring boot applications. *Database Systems Journal*, 16(4):85–99, 2021.

[Ly18] Minh Ly. Creating api test automation of a service for company x. 2018.

[LZ23] Qing Liu and Anna Zhang. Advanced dependency management with maven: Techniques and tools. *Software Development Journal*, 16(2):87–100, 2023.

[LZS20] Xiaofeng Li, Yilun Zhu, and Hao Sun. Api testing using test automation in agile software development: A systematic review. *Journal of Systems and Software*, 164:110569, 2020.

[Mad23] Ansgar Becker Madhon. Heidisql 12.5.0.6680 openssl 3.1.0 vulnerability cve-2022-4203. *GitHub*, 2023.

[Mal24] Herman Milton Maleiane. Spring Boot+Thymeleaf+ Leaflet Js Mapping Corona Virus. https://medium.com/@hermanmaleiane/spring-boot-thymeleaf-leaflet-js-mapping-corona-virus-a8309c5a0b6d, 2024. [Online; accessed 01-janeiro-2024].

[Mas11] Mark Masse. *REST API design rulebook*. "O'Reilly Media, Inc.", 2011.

[Mav24] Maven. Apache Maven Project. https://maven.apache.org/, 2024. [Online; accessed 11-august-2024].

[May23] Matthew S Mayernik. Toward stronger coupling between technical infrastructures and institutional processes in data-intensive science. 2023.

[MBM01] Edward M Mikhail, James S Bethel, and J Chris McGlone. *Introduction to modern photogrammetry*. John Wiley & Sons, 2001.

[Med24] Medium. Testes de aceitação. https://medium.com/@celionormando/testes-de-aceita%C3%A7%C3%A3o-cc2440894aab, 2024. [Online; accessed 15-julho-2024].

[MG24] Carlos Martinez and Sofia Garcia. Building scalable rest apis with spring boot and mysql workbench. In *2024 IEEE Conference on Cloud and Big Data*, pages 145–159. IEEE, 2024.

[Mic23a] Microsoft. Arquiteturas comuns de aplicativo Web. https://docs.microsoft.com/pt-br/dotnet/architecture/modern-web-apps-azure/common-web-application-architectures, 2023. [Online; accessed 22-November-2023].

[Mic23b] Microsoft. Extension API. https://code.visualstudio.com/api, 2023. [Online; accessed 23-November-2023].

[Mic23c] Microsoft. Getting Started. https://code.visualstudio.com/docs, 2023. [Online; accessed 25-November-2023].

[MNS+23] J Meyer, S Nebiker, S Schürmann, E Ferrari, and M Ammann. Exploiting pole-like objects from cadastres for sub-metre accurate integrated georeferencing of low-cost mobile mapping systems. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 48:303–310, 2023.

[MRE09] Luis Martinez, Rosa Rodríguez, and Macarena Espinilla. Reja: A georeferenced hybrid recommender system for restaurants. volume 3, pages 187–190, 09 2009.

[MSB11] Glenford J Myers, Corey Sandler, and Tom Badgett. *The art of software testing*. John Wiley & Sons, 2011.

[MUI+23] Hamza Ahmad Madni, Muhammad Umer, Abid Ishaq, Nihal Abuzinadah, Oumaima Saidani, Shtwai Alsubai, Monia Hamdi, and Imran Ashraf. Water-quality prediction based on h2o automl and explainable ai techniques. *Water*, 15(3):475, 2023.

[Mul10] Peter Mularien. *Spring Security 3*. Number 3. Packt Publishing Birmingham,, England, 2010.

[Nas11] Ihsan Naskah. Memanfaatkan web services untuk layanan informasi pekerjaan online. *Inspiration: Jurnal Teknologi Informasi dan Komunikasi*, 1(2), 2011.

[Net23] Virtual Private Networks. Virtual Private Networks. https://www.gta.ufrj.br/ensino/eel879/trabalhos_vf_2012_2/vpn/SeguranaemredesVPN.html, 2023. [Online; accessed 19-November-2023].

[New15] S Newman. Building microservices: Designing fine-grained systems, 1ra edición. sebastopol, ciudad de california, 2015.

[NP22] T. Nguyen and R. Patel. Enhancing user experience with thymeleaf: A case study of web application frameworks. In *Proceedings of the 2022 International Conference on Software Engineering*, pages 50–60. IEEE, 2022.

[Ora23] Oracle. What is MySQL. `https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html`, 2023. [Online; accessed 09-april-2023].

[PHP23] PHP. PHP Documentations. `https://www.php.net/manual/en/intro.pdo.php`, 2023. [Online; accessed 23-November-2023].

[PI23] Mihai Popovici and Elena Ionescu. Persistence layer implementation using spring data and hibernate in enterprise applications. *Journal of Software Engineering and Applications*, 16(4):123–134, 2023.

[Pre05] Roger S Pressman. *Software engineering: a practitioner's approach*. Palgrave macmillan, 2005.

[PRPR17] K Siva Prasad Reddy and K Siva Prasad Reddy. Getting started with spring boot. *Beginning Spring Boot 2: Applications and Microservices with the Spring Framework*, pages 21–33, 2017.

[PYR20] Stanislao Pinzón, Stefany Yánez, and Milton Ruiz. Optimal location of transformers in electrical distribution networks using geographic information systems. *Enfoque Ute*, 11(1):84–95, 2020.

[RAR13] Leonard Richardson, Mike Amundsen, and Sam Ruby. *RESTful Web APIs: Services for a Changing World*. "O'Reilly Media, Inc.", 2013.

[RDM19] Antonio Gomes Rodrigues, Bruno Demion, and Philippe Mouawad. *Master Apache JMeter-From Load Testing to DevOps: Master performance testing with JMeter*. Packt Publishing Ltd, 2019.

[RHW+24] Alejandro Román, Sergio Heredia, Anna E Windle, Antonio Tovar-Sánchez, and Gabriel Navarro. Enhancing georeferencing and mosaicking techniques over water surfaces with high-resolution unmanned aerial vehicle (uav) imagery. *Remote Sensing*, 16(2):290, 2024.

[RJB04] James Rumbaugh, Ivar Jacobson, and Grady Booch. *The Unified Modeling Language Reference Manual*. Object Technology Series. Addison-Wesley, 2 edition, 2004.

[RW21] David Roberts and Emily White. A technical review of leafletjs: Features, performance, and best practices. Technical Report TR-2021-14, Tech University of Example, 2021.

[San14] Nuno Silva Santos. Georreferenciação em dispositivos móveis num contexto industrial. 2014.

[San16] João Miguel Martins Marques dos Santos. *Push notifications*. PhD thesis, 2016.

[Sca13] Carlo Scarioni. *Pro Spring Security*. Apress, 2013.

[Sib16] Swono Sibagariang. Penerapan web service pada perpustakaan berbasis android. *Jurnal Mahajana Informasi*, 1(1):28–32, 2016.

[SMC+23] Lorenz Schmid, Tomislav Medic, Brian D Collins, Lorenz Meier, and Andreas Wieser. Georeferencing of terrestrial radar images in geomonitoring using kernel correlation. *International Journal of Remote Sensing*, 44(21):6736–6761, 2023.

[SML+23] Dexuan Sha, Anusha Srirenganathan Malarvizhi, Hai Lan, Xin Miao, Hongie Xie, Daler Khamidov, Kevin Wang, Seren Smith, Katherine Howell, and Chaowei Yang. Arcci: A high-resolution aerial image management and processing platform for sea ice. 2023.

[SOM] Ian SOMMERVILLE. Engenharia de software/ian sommerville. *Tradução Ivan Bosnic e karlinka G. de O. Gonçalves*.

[Som11] Ian Sommerville. Software engineering 9th edition. *ISBN-10*, 137035152:18, 2011.

[Sou23] Nogueira T. Sousa, R. Bupi: Enhancing transparency and efficiency in property management in portugal. 2023.

[Spi20] Laurentiu Spilca. *Spring security in action*. Simon and Schuster, 2020.

[Spr23a] Spring. Provide a WebSecurityConfigurerAdapter. `https://docs.spring.io/spring-security/reference/servlet/oauth2/login/core.html#oauth2login-provide-websecurityconfigureradapter`, 2023. [Online; accessed 29-March-2023].

[Spr23b] Spring. Spring Data. `https://spring.io/projects/spring-data`, 2023. [Online; accessed 22-june-2023].

[Spr23c] Spring. Spring Framework Overview. `https://docs.spring.io/spring-framework/docs/current/reference/html/overview.html#overview`, 2023. [Online; accessed 22-june-2023].

[Spr23d] SpringVMware. Building REST services with Spring. `https://spring.io/guides/tutorials/rest/`, 2023. [Online; accessed 23-November-2023].

[Spr24] Spring. Authentication. `https://docs.spring.io/spring-security/reference/features/authentication/index.html`, 2024. [Online; accessed 29-March-2024].

[SS17] J Sharma and Ashish Sarin. Getting started with spring framework: Covers spring 5, 2017.

[SS23] Spring-Security. Provide a WebSecurityConfigurerAdapter. `https://docs.spring.io/spring-security/site/docs/current/api/org/springframework/security/config/annotation/web/configuration/WebSecurityConfigurerAdapter.html`, 2023. [Online; accessed 29-March-2023].

[SSS11] Sunil Pratap Singh, Jitendra Sharma, and Preetvanti Singh. A geo-referenced information system for tourism (georist). *International Journal of Geomatics and Geosciences*, 2(2):456–464, 2011.

[SW22] J. Smith and L. Wang. Comparing thymeleaf with other java templating engines in enterprise applications. *Journal of Computing and Technology*, 18(4):89–102, 2022.

[Sys23] TRX Systems. neon. `https://www.trxsystems.com/pt-mil.html`, 2023. [Online; accessed 22-june-2023].

[SZT12] Baron Schwartz, Peter Zaitsev, and Vadim Tkachenko. *High performance MySQL: optimization, backups, and replication*. "O'Reilly Media, Inc.", 2012.

[Tec23a] Techiediaries. API with PHP (Complete Beginner's Guide. `https://rapidapi.com/blog/how-to-use-an-api-with-php/`, 2023. [Online; accessed 22-june-2023].

[Tec23b] Techiediaries. PHP 7 Tutorial with MySQL: CRUD REST API. `https://www.techiediaries.com/php-rest-api/`, 2023. [Online; accessed 23-November-2023].

[Thy23] Thymeleaf. Thymeleaf Overview. `https://www.thymeleaf.org/documentation.html`, 2023. [Online; accessed 11-august-2024].

[TW07] Seyed Tahaghoghi and Hugh E Williams. *Learning MySQL*. "O'Reilly Media, Inc.", 2007.

[USG24] USGS. What is a geographic information system (GIS). `https://www.usgs.gov/faqs/what-geographic-information-system-gis`, 2024. [Online; accessed 21-august-2024].

[UW+20] Arief Umarjati, Arief Wibowo, et al. Implementasi jwt pada aplikasi presensi dengan validasi fingerprint, geotagging dan device checker. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, 4(6):1085–1091, 2020.

[VIS19] Alex Valenzuela, Esteban Inga, and Silvio Simani. Planning of a resilient underground distribution network using georeferenced data. *Energies*, 12(4):644, 2019.

[VMI19] Alex Valenzuela, Iván Montalvo, and Esteban Inga. A decision-making tool for electric distribution network planning based on heuristics and georeferenced data. *Energies*, 12(21):4065, 2019.

[Wal22] Craig Walls. *Spring in action*. Simon and Schuster, 2022.

[Wic14] Bayu Wicaksono. Pemanfaatan web service moodle berbasis rest-json untuk membangun moodle online learning extension berbasis android. 2014.

[WM12] Robert Winch and Peter Mularien. *Spring Security 3.1*. Packt Pub., 2012.

[WT12] Kishor Wagh and Ravindra Thool. A comparative study of soap vs rest web services provisioning techniques for mobile host. *Journal of Information Engineering and Applications*, 2(5):12–16, 2012.

[Yao20] Xiaobai A. Yao. Georeferencing and geocoding. In Audrey Kobayashi, editor, *International Encyclopedia of Human Geography (Second Edition)*, pages 111–117. Elsevier, Oxford, 2020.

[dP24a] Águas de Portugal. Águas de Portugal. `https://www.adp.pt/`, 2024. [Online; accessed 01-janeiro-2023].

[dP24b] Águas de Portugal. Águas de Portugal. `https://www.epal.pt/EPAL/en/menu/products-and-services/h2o-quality`, 2024. [Online; accessed 13-julho-2024].

Contatos:
Universidade de Évora
Escola de Ciências e Tecnologia
Colégio Luis António Verney, Rua Romão Ramalho, nº59
7000 - 671 Évora | Portugal
Tel: (+351) 266 745 371
email: geral@ect.uevora.pt

Bairro Satón, Aeroporto - São Tomé, nº 1980
Tel: (+239) 991 68 03 | 909 68 28
WhatsApp: (+239) 991 68 03
email: m38980@alunos.uevora.pt
filipelpaixao@hotmail.com