INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Sistema de controlo de acessos e segurança

DANIEL FILIPE COSTA
Outubro de 2021

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Sistema de controlo de acessos e segurança

Daniel Filipe Costa

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Engenharia de Software

Orientador: Jorge Pinto Leite

Coorientador: Constantino Martins

Supervisor: Nuno Rodrigues

Porto, outubro 2021

Agradecimentos

O meu muito obrigado à minha família, aos meus amigos e a todos que me acompanharam ao longo do caminho percorrido até este momento.

Deixo um agradecimento especial ao meu orientador, Jorge Pinto Leite, que desde o início me motivou bastante a alcançar este objetivo, ajudando-me sempre com os seus comentários e opiniões ao longo do desenvolvimento deste projeto.

Deixo também um agradecimento especial à minha namorada, Joana Pereira, pelo apoio incondicional desde o momento em que iniciei a caminhada neste mestrado, por nunca ter duvidado de mim, mesmo quando eu duvidava.

Resumo

O presente documento tem como objetivo descrever todo o processo de desenvolvimento do projeto realizado no âmbito da unidade curricular Tese/Dissertação/Estágio, do Mestrado em Engenharia Informática, no ramo de Engenharia de Software, do Instituto Superior de Engenharia do Porto.

O projeto, desenvolvido para a empresa Actuasys, enquadra-se na área de atividade de Recursos Humanos, mais concretamente na gestão de Recursos Humanos, a principal área de atividade da empresa onde este foi desenvolvido.

Este projeto, denominado "Sistema de Controlo de Acessos e Segurança", consiste no desenvolvimento de uma solução de software, num ambiente web, que permitirá a gestão e controlo de acessos a zonas pré-definidas, destinado a empresas de todas as dimensões, com o intuito destas conseguirem controlar toda a afluência de pessoas dentro e fora das suas instalações.

Ao longo deste documento, o leitor irá ser contextualizado gradualmente do desenvolvimento do projeto, conhecendo as circunstâncias bem como os indicadores de sucesso e consequentes resultados do sistema desenvolvido.

O sucesso deste projeto implica que este apresente resultados positivos ao que se comprometeu e seja reconhecido como uma mais-valia na ótica da empresa, assim como dos clientes. Já o insucesso representa uma tentativa falhada do desenvolvimento de um sistema que visa o controlo de acessos.

Na avaliação à prova de conceito conclui-se que o desenvolvimento deste projeto apresenta resultados maioritariamente positivos, representando o sucesso no desenvolvimento de um produto que respeita as políticas de qualidade da empresa.

Palavras-chave: Gestão de Recursos Humanos; Controlo de acessos; Ambiente web

Abstract

This dissertation aims to describe the entire development process of the project held by the course unit “Tese/Dissertação/Estágio” in Computer Engineering master’s degree, in software engineering specifically, at Instituto Superior de Engenharia do Porto.

The project was produced for the company Actuasys, it fits in the Human Resources activity, more specifically in Human Resources management, this being the main area of activity in the company where this project has been held.

This project called “Sistema de Controlo de Acessos e Segurança” consists in the development of a web-based software solution that will allow the management and access control in predefined areas, destined for companies of any dimension, to control the affluence of people inside and outside of its facilities.

Throughout this document, the reader will gradually be contextualized in the development of the project, knowing the circumstances of this, as well as its success indicators and consequent results of the developed system.

The achievement of this project implies positive results, to what it is committed. It also must be considered an asset in this company´s perspective and to its customers. As for the underachievement, it represents a failed attempt to develop a system aimed for access control.

In the conducted proof of concept, it is concluded that the development of this project is predominantly positive. Thus, meaning the success of the development of a product that respects the company's quality policies.

Keywords: Human Resources Management; Access Control; Web-based software solution

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos e Símbolos

Lista de Acrónimos

CLS	*Cumulative Layout Shift*
DAL	*Data Access Logic*
FCP	*First Contentful Paint*
HTTP	*HyperText Transfer Protocol*
IP	*Internet Protocol*
JSON	*JavaScript Object Notation*
LCP	*Largest Contentful Paint*
MAC	*Media Access Control*
PIN	*Personal Identification Number*
PME	Pequenas e médias empresas
QA	*Quality Assurance*
RGPD	Regulamento Geral sobre Proteção de Dados
Snap-in	Interface/componente modular
SQL	*Structured Query Language*
TBT	*Total Blocking Time*
TCP	*Transmission Control Protocol*
TTI	*Total Blocking Time*
UML	*Unified Modeling Language*
VPN	*Virtual Private Network*
WCF	*Windows Communication Foundation*

1 Introdução

Neste capítulo é feita uma introdução sobre o contexto do projeto, com o intuito de fornecer ao leitor o envolvimento que levou ao seu desenvolvimento. Inicialmente é dada uma breve apresentação da empresa e apresentas as suas principais áreas da atividade. De seguida, é apresentado o contexto sobre a área da atividade em que se insere o tema do projeto, enumerando-se os principais problemas e objetivos a endereçar no desenvolvimento do sistema. Será ainda, efetuada uma breve síntese da análise de valor a endereçar no segundo capítulo, Contexto e Estado da arte, finalizando com a apresentação da organização do documento, oferecendo um resumo dos restantes capítulos.

É de salientar que ao longo do documento são apresentados alguns termos em inglês, constituindo parte integrante do léxico do documento. Devido a serem um termo generalizado e comum na área de especialização da dissertação, como por exemplo as palavras software e web, não aparecem em itálico.

1.1 Empresa Actuasys

A Actuasys (Figura 1), fundada em 1995 com o seu primeiro nome "Milénio 3", sofreu um *rebranding*[1] em 2019 para a marca atual, Actuasys. É uma empresa que se insere numa área de engenharia e tecnologia muito específica, tendo como seu principal foco tecnologias e sistemas para a da gestão de recursos humanos, assim como indica o seu slogan: "Acreditamos que o sucesso das empresas resulta do compromisso e do bem-estar dos seus colaboradores" (Actuasys 2020).

Esta destaca-se por ser uma empresa de engenharia dedicada não só ao design, conceção, industrialização e comercialização de produtos e sistemas de software, mas também equipamentos de primeira linha, que complementam e suportam com toda a eficácia as soluções por si desenvolvidas (Actuasys 2020). A garantia de qualidade, competência e profissionalismo pode ser provada pela sua carteira de clientes, contando já com mais de 700, alguns deles acompanhando o progresso da empresa desde o início. Atuando em áreas de atividade distintas, desde administração pública a turismo, a empresa conseguiu conquistar PME's e grandes empresas, desenvolvendo sempre relações sólidas para que as necessidades dos clientes sejam sempre satisfeitas (Actuasys 2020).

Através do estabelecimento de parcerias com empresas, como SAP e Microsoft, e ajudas de financiamento de programas como o Norte 2020 e Portugal 2020, a Actuasys consegue focar-se no desenvolvimento de novos produtos e constante manutenção dos produtos já

[1] *Rebranding* (sugestão de tradução: renovação da marca): processo de alteração da imagem de uma empresa ou organização

existentes, tendo sempre como principal objetivo garantir que os produtos lançados para o mercado estejam desenvolvidos nas tecnologias mais emergentes, respeitando as políticas de qualidade da empresa.

Figura 1 – Logotipo da empresa (Actuasys 2020)

1.2 Contextualização

O controlo de acessos pretende assegurar a restrição ao acesso a algum lugar ou recurso, garantindo que nenhuma entidade não autorizada aceda a algo ao qual não é suposto ter acesso. Existem dois tipos de controlo de acessos: o controlo físico, onde se pretende restringir o acesso a salas, edifícios e parques de estacionamento, entre outros espaços; e o controlo lógico ou virtual, onde se pretende limitar o acesso a redes privadas e determinados ficheiros, entre outros recursos. Este projeto enquadra-se no âmbito do primeiro, controlo físico.

Assim, de modo a garantir a segurança física, isto é, o controlo de acessos de entidades a zonas restringidas, deverá existir alguma barreira que bloqueie a entrada ou saída de pessoas destas zonas. Neste projeto, vamos ficar a conhecer que estas barreiras poderão ser controladas por sistemas eletrónicos, que dependem de algum tipo de identificação do utilizador e, através da utilização de processos designados de autenticação e autorização, se determine a entidade que está a requisitar o acesso e se a mesma tem acesso à zona controlada ou não.

Estes sistemas têm como principal objetivo minimizar o risco de acesso não autorizado a recursos ou zonas por parte das pessoas. Sendo assim um componente essencial de segurança para as empresas e organizações que pretendem assegurar e controlar o acesso aos seus edifícios.

1.3 Problema

A gestão de acesso a zonas de espaço físico controladas para qualquer empresa ou organização é essencial, mas para tal ser possível, é necessário que exista um sistema capaz de garantir que estes espaços sejam intransponíveis e/ou de controlar o acesso a zonas pré-determinadas, garantindo assim que entidades não autorizadas sejam negadas e barradas do acesso às mesmas. Caso essa tentativa seja detetada, deverá ainda esse sistema emitir alertas

e interagir com a entidade bloqueadora de acesso, realizando-se assim uma monitorização constante e em tempo-real das tentativas de acesso.

O problema prende-se com a otimização da solução de controlo de acessos atual da empresa, que se encontra como um módulo "*snap-in*" de uma aplicação Windows, revendo os processos de determinação de acesso a zonas seguras e controladas conhecidos pelo sistema, desde a autenticação, autorização, validação de acesso e a confiabilidade do sistema, ou seja, uma otimização e revisão do sistema a nível funcional, mas também a modernização na componente tecnológica e arquitetural.

Assim, o principal problema prende-se com a confiabilidade e desempenho do sistema, garantindo que todos os processos enumerados sejam executados com a máxima segurança e, simultaneamente, com tempos de resposta mínimos, certificando que a aferição de permissão de acesso seja corretamente avaliada, impedindo presenças indesejadas nas zonas definidas.

1.4 Objetivos

Os objetivos desta dissertação são múltiplos, resumindo-se da seguinte forma:

- Apresentar e desenvolver uma solução de software que seja capaz de garantir o controlo de acessos de entidades a espaços físicos controlados;
- Desenvolver um sistema que cumpra o objetivo anterior aplicando sempre boas práticas de engenharia de requisitos, programação e padrões de software;
- Analisar e comparar diferentes perspetivas e metodologias que poderão ser aplicadas para o desenvolvimento do sistema, documentado neste documento o levantamento de requisitos, os testes ao protótipo final, a análise e a resposta às hipóteses identificadas.

Pretende-se assim que seja desenvolvido um sistema de controlo de acessos, que permita garantir as necessidades tanto dos clientes como da empresa, oferecendo uma solução que deverá assegurar os padrões e políticas de qualidade da empresa. Os *outcomes* esperados do projeto são:

- Portal web de configuração e parametrização do sistema de controlo de acessos;
- Solução e algoritmo de determinação de acesso;
- Comunicações e integração com as unidades responsáveis pelo controlo de barreiras;
- Documentação e de todo o processo de criação e experimentação deste sistema.

Nesse sentido, de modo a cumprir todos os objetivos pretendidos nesta dissertação e projeto, foram definidos um conjunto de tarefas a serem desempenhadas ao longo do desenvolvimento deste, assim como:

- Contextualização e análise sobre o tema do projeto;

- Levantamento e especificação de requisitos;
- Documentação das decisões tomadas a nível arquitetural;
- Desenvolvimento de um protótipo do sistema capaz de:
    - Registar e gerir entidades para acessos (colaboradores, externos e visitantes);
    - Criar e parametrizar horários;
    - Definir zonas a serem controladas;
    - Definir perfis de acessos;
    - Registar unidades de registo;
    - Determinação de autorização do acesso de uma entidade a uma zona;
    - Registo das tentativas de acesso a uma zona (com e sem sucesso);
    - Emissão de alarmes;
    - Monitorização em tempo-real das tentativas de acessos.
- Comparação e avaliação de melhorias comparando com a atual solução de acessos, a níveis arquiteturais e de desempenho;
- Documentação do desenvolvimento do protótipo, bem como todos os diferentes tipos de testes efetuados.

1.5 Análise de valor

A análise de valor é uma metodologia que permite avaliar qual a relação entre o custo e benefício de um produto ou serviço, identificando quais os pontos positivos e negativos deste. Uma das etapas mais importantes da análise de valor, e que pode fornecer uma síntese desta, é a elaboração da proposta de valor. Para a elaboração da proposta de valor, é necessário conhecer os pontos fortes do produto que estamos a desenvolver, assim como o que o distingue dos demais produtos semelhantes no mercado, caso existam. Após realizada uma análise a como responder a estas questões, obteve-se uma proposta de valor que vai de encontro com os princípios da empresa e os seus produtos:

“Desenvolvemos soluções para a área de dos Recursos Humanos que promovem a eficiência, o rigor e a transparência no interior das organizações, reforçando a relação entre colaborador e empresa, promovendo sempre a produtividade e bem-estar dia após dia.”

1.6 Estrutura do documento

A seguinte dissertação apresenta-se dividida em 7 capítulos, cada um deles apresentando os diferentes estados do desenvolvimento. De seguida, são listados os respetivos capítulos, acompanhados com uma breve descrição.

1. **Introdução**: Presente capítulo onde se ficou a conhecer a origem do projeto a desenvolver, através da contextualização, identificação do problema e objetivos a endereçar nos restantes capítulos;

2. **Contexto e Estado da Arte**: Neste segundo capítulo ficam-se a conhecer os tópicos, que contextualizam um sistema de controlo de acessos, apresentando o estado da arte em todos estes tópicos, enquadrando o leitor para os conceitos que são relevantes para o projeto, tanto a nível contextual como tecnológico;
3. **Análise de valor**: No terceiro capítulo será efetuada a análise de valor do produto a ser desenvolvido, adaptando a mesma com o modelo *New Concept Development*, apresentando de seguida a proposta de valor, a análise de sacrifícios e benefícios associados, terminando com o modelo Canvas;
4. **Análise e design**: No quarto capítulo, o leitor ficará a conhecer todos os requisitos funcionais e não funcionais associados ao produto, todos os conceitos associados ao tema, bem como as diferentes abordagens possíveis e pensadas para o seu desenvolvimento;
5. **Implementação**: No capítulo de implementação é dado a conhecer ao leitor as escolhas efetuadas para o desenvolvimento da solução, a implementação do algoritmo de avaliação de acesso, finalizando com a apresentação das páginas web desenvolvidas para configuração do sistema;
6. **Experimentação e avaliação**: Neste sexto capítulo são levantadas as hipóteses de investigação definindo a base de como a solução é avaliada. São apresentadas todas metodologias e ferramentas utilizadas com o intuito de testar o produto, assim como a análise aos resultados obtidos na experimentação realizada;
7. **Conclusão**: No sétimo e último capítulo são apresentadas as conclusões retiradas do desenvolvimento do projeto, bem como as limitações encontradas e trabalho futuro a realizar.

2 Contexto e Estado da arte

Neste capítulo são detalhados alguns tópicos que darão ao leitor uma maior contextualização ao tema e problema do projeto, descrevendo assim qual o estado da arte envolvente dos sistemas de controlo de acessos. Para finalizar, será ainda efetuada uma análise de valor à solução proposta.

2.1 Equipamentos no controlo de acessos

A criação de um sistema de controlo de acessos assume a existência de equipamentos que controlem a autenticação e o acesso a determinadas zonas ou espaços, através da determinação por parte das mesmas se a entidade terá de facto acesso ou não. Em alguns cenários, fará sentido e deverá existir mesmo pessoal, como seguranças, supervisores ou rececionistas, a controlar as entradas e saídas de certas zonas. Cada vez mais, a substituição de pessoal por maquinaria e sistemas automáticos é uma realidade ("A máquina no lugar do homem", 2018), isto prende-se com o facto dos sistemas informáticos desenvolvidos serem cada vez mais eficientes e seguros.

Neste momento, existem equipamentos de todo o tipo, sejam estes destinados para abertura de portas, desbloquear barreiras de acessos ou até mesmo abertura de cancelas. Um equipamento para ser capaz de efetuar o controlo de acessos, deverá possuir determinadas capacidades, assim como:

1. Leitura e autenticação de um ou múltiplos fatores;
2. Comunicação com algum equipamento, seja este um painel de controlo ou um computador, para determinar a concessão ou não o acesso ao espaço;
3. Comunicar com a entidade bloqueadora de acesso, como por exemplo, uma porta ou um torniquete, desbloqueando, se o acesso for concedido, a mesma.

Ao longo do tempo, podemos ver uma notória evolução nos equipamentos para controlo de acessos (Houlis 2018), desde os simples teclados alfanuméricos aos leitores de porta por *Internet Protocol* (IP), acompanhando sempre as necessidades dos consumidores, públicos ou privados, assim como, a necessidade de ultrapassar potenciais brechas de segurança, garantindo sempre que os sistemas sejam o mais seguro possíveis. Esta evolução dos equipamentos é acompanhada pela evolução constante do *hardware* e capacidade, mas também pela descoberta de novas alternativas de autenticação para além do tradicional PIN ou palavra-passe. Estes tipos de autenticação tradicionais não foram completamente esquecidos ou ultrapassados, como vamos poder constatar no subcapítulo seguinte "Autenticação no sistema".

Figura 2 – Evolução dos tipos de equipamentos (Access Control Systems, 2019)

Na Figura 2, é possível analisar a evolução dos tipos de equipamentos, desde equipamentos com a simples e única funcionalidade de leitura de cartões, até leitores com múltiplos níveis de autenticação, como é o caso do leitor biométrico para leitura de impressões digitais com teclado numérico para inserção de um código PIN. Esta figura retrata a evolução do estado da arte destes equipamentos, mas prevê-se que a evolução dos equipamentos continue a um nível cada vez mais elevado (E. Barnhart 2019), acompanhado sempre o rigor e as dificuldades a nível de segurança, bem como os diferentes tipos de autenticação que continuam a surgir, nomeadamente a nível biométrico e de cartões inteligentes.

Os equipamentos utilizados para a leitura de credenciais, podem ser classificados de acordo com as funções que conseguem desempenhar. Assim podem ser divididos em três grupos distintos (HS Tech Group 2017):

- Leitores não inteligentes, como por exemplo, leitores de cartões ou teclado numérico simples. Estes necessitarão de comunicar com um painel de controlo, transmitindo a informação necessária para este tomar a decisão e comunicar com a entidade bloqueadora de acesso. São menos eficientes e dependentes da existência de rede e de um painel de controlo.

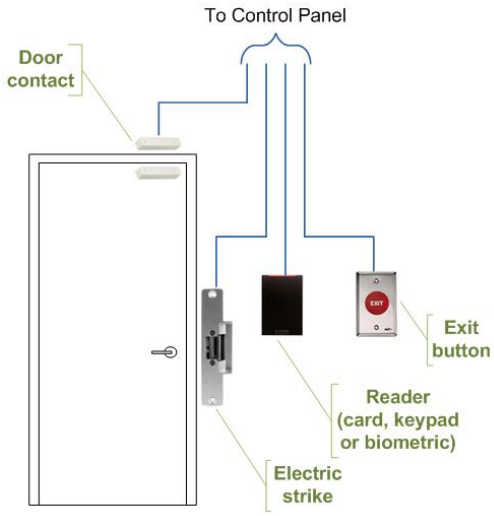

Figura 3 – Sistema não inteligente (Fair 2018)

- Leitores semi-inteligentes, estes são bastante semelhantes aos leitores não inteligentes, contêm toda a informação necessária para desbloquear a entrada e permitir o acesso mas não conseguem determinar se as credenciais inseridas têm permissão de acesso. Para isso, terão de comunicar com o painel de controlo principal e só depois poderão dar acesso, informando a entidade bloqueadora que o acesso é permitido. Estes sistemas são bastante semelhantes aos sistemas não inteligentes, sendo que, caso a comunicação com o painel de controlo falhe ou tenha interrupções, não vão funcionar devidamente.
- Leitores inteligentes, semelhante ao anterior, estes também têm a capacidade de comunicar com a entidade bloqueadora com o intuito de desbloquear e permitir o acesso. A principal diferença prende-se com a capacidade que estes equipamentos têm para tomar decisões isoladamente, ou seja, têm memória e poder de processamento necessário para conseguir garantir a permissão de acesso sem a necessidade de comunicar com o painel de controlo. Assim, a comunicação com o painel de controlo pode ser efetuada, mesmo depois da permissão de acesso ter sido garantida, com o intuito de registar toda a informação necessário de quem pediu o acesso a determinada zona ou espaço.

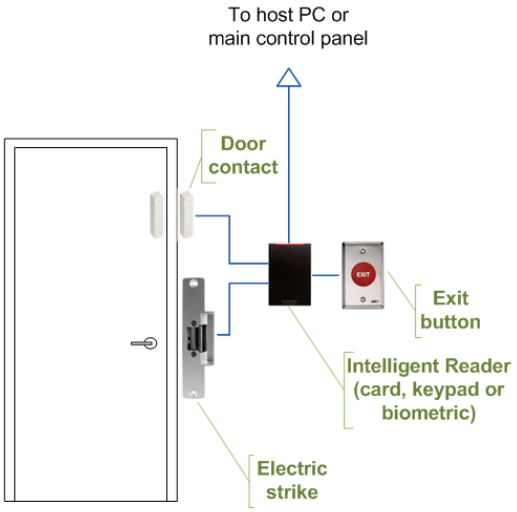

Figura 4 – Sistema inteligente (Fair 2018)

Para além desta distinção dos leitores baseado nas suas capacidades, surge um novo tipo de sistema de controlo de acessos, que é baseado no estabelecimento e comunicação através de IP, designado sistema de IP ou mesmo sistema baseado na *cloud*[2]. Estes sistemas requerem que o equipamento de leitura esteja diretamente conectado a uma rede interna, normalmente através de um cabo de rede ou através da sua capacidade de conexão sem fios. Ao invés da comunicação normal com um painel de controlo, estes sistemas funcionam através da conexão à rede da empresa, necessitando apensas de estabelecer essa ligação, sendo assim mais fáceis de instalar.

Estes sistemas através de IP, têm vindo a ganhar mais popularidade acompanhando também o aumento de popularidade, tanto a nível empresarial como pessoal, das soluções baseadas na *cloud*. Apesar de muitas empresas e organizações escolherem este tipo de sistemas devido à sua facilidade de instalação, utilização e acessibilidade, há ainda uma grande contenção do mercado da segurança em relação a qual é o melhor sistema, devido ao facto de estando estes sistemas conectados a uma rede, estão sempre sujeitos a falhas na rede e/ou ataques informáticos.

2.2 Autenticação no sistema

Todos os sistemas de controlo de acessos, para funcionar devidamente, devem ser capazes de autenticar credenciais, que existem no sistema através do registo das mesmas, aquando ocorre uma tentativa de acesso. Neste subcapítulo será dada uma definição de autenticação, seguida da explicação de todos os fatores de autenticação conhecidos, exemplificando cada

[2] *Cloud* (sugestão de tradução: armazenamento em nuvem): armazenamento de dados em ambiente digital, chamado de "nuvem".

um destes com casos práticos. No final é apresentado o ponto de situação da autenticação nos sistemas modernos, no ponto de vista da segurança e confiabilidade.

Autenticação é o momento de validação de credenciais de uma pessoa, com o intuito de validar e provar a sua identidade, isto é, quando uma entidade apresenta algum tipo de credenciais, seja esta um cartão ou um código PIN, o sistema vai verificar se a pessoa é quem realmente tenta provar que é, através da validação das mesmas (P. Christensson 2018). Muitas vezes este processo é confundido com o processo de autorização, que acontece depois de o primeiro se verificar, ou seja, quando a identidade da pessoa foi autenticada com sucesso no sistema. *Autorização* é o processo de verificação sobre o tipo de acesso que a pessoa autenticada no sistema tem sobre um recurso, seja este um ficheiro, uma base de dados, ou até mesmo um espaço fisíco, o que é o principal foco deste projeto (Siddiqui 2018).

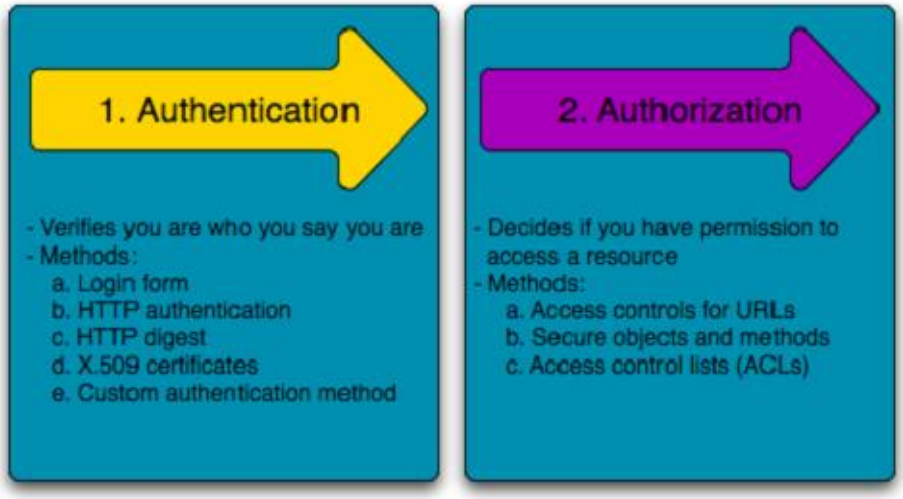

Figura 5 – Autenticação e autorização (Siddiqui 2018)

Autenticação verifica e valida a própria identidade de quem está na tentativa de autenticação, enquanto autorização é um processo de validação ao que o autenticado tem acesso no sistema.

Posto isto, pode-se concluir que estes dois conceitos coexistirão na maior parte dos sistemas, sendo que, num sistema de controlo de acessos são um ponto crítico, no que toca à questão de segurança. Em conjunto, garantem a autenticidade e permissão da pessoa na tentativa de acesso, protegendo o sistema de potenciais tentativas de fraude, sendo assim, um ponto-chave no controlo de acessos.

Atualmente, quando se fala em autenticação é difícil não pensar em termos como "Autenticação em 2 fatores" ou "Autenticação em vários fatores", que são termos que surgem da necessidade de melhorar a segurança no acesso a algum recurso ou lugar. Para continuarmos a falar sobre estes termos, é necessário perceber primeiro o que é um fator de autenticação e quais são os diferentes fatores conhecidos.

Um fator de autenticação não é nada mais que um tipo único de autenticação, que é utilizado no momento de provar ao sistema a nossa identidade. Por exemplo, quando estamos a tentar ter acesso a algo, é requisitado sempre algum tipo de informação que a pessoa pode fornecer, seja esta informação, uma palavra-passe, um PIN ou uma impressão digital. Esta informação enquadra-se num tipo de autenticação, sendo denominado fator de autenticação (Dias 2017).

A razão pela qual os fatores são múltiplos tem a ver com a evolução dos tempos, pois à medida que se sente a evolução das tecnologias, acrescem também os perigos e debilidades do mundo *online*, devido à maior quantidade de informação exposta e que necessita de uma maior proteção. Posto isto, assegurar-se apenas com aquilo que acreditamos ser uma "palavra-passe forte" de uma combinação de letras, números e símbolos, deixa de ser possível, pois estudos datados de 2018 revelam que 81% de atividade de pirataria informática está ligada ao roubo de autenticação de palavras-passes (Trace Security blog). Desta forma, com objetivo de fortalecer a segurança na autenticação, alguns fatores (tipos de autenticação, como acima explicado) emergiram.

Antes de avançar com maior detalhe sobre os conceitos em questão, é necessário compreender quais são os fatores de autenticação conhecidos. Neste momento, existem cinco fatores conhecidos, apresentados por ordem cronológica, sendo eles (Sarmah 2019):

1. **Conhecimento**

Este é o mais básico, mas também o mais usado dos fatores, o "Conhecimento" refere-se a tudo que pode ser memorizado, dando origem a conhecimento do indivíduo, podendo ser usado para garantir o acesso a algo. Um exemplo de um fator de conhecimento, e provavelmente o mais utilizado e conhecido, é a palavra-passe ou PIN. Estes são exemplos que podem ser memorizados e utilizados sempre que necessário e se tenta provar a identidade num sistema. Outro exemplo deste tipo de autenticação, pode ser uma pergunta de segurança, que muitas vezes é utilizado para recuperar acesso a uma conta, quando se esquece, precisamente, da palavra-passe ou do nome de utilizador. Este fator pode ser caraterizado pela expressão "Algo que sabe".

2. **Posse**

Assim como o seu nome indica, este fator refere-se a algo que a pessoa tem em sua posse, ou seja, informação que leva consigo. Por exemplo, antes de enviar dinheiro para alguém utilizando um aparelho eletrónico como telemóvel, *tablet* ou computador, alguns sistemas de segurança dos bancos pedem um código (normalmente um código de utilização única com 6 a 8 dígitos) que expira após a utilização ou num certo período, normalmente muito curto. Existem duas formas de gerar estes códigos: HOPT (HMAC- based One-Time Password) que gera uma palavra passe para ser utilizada e expira após a sua utilização; TOPT (Time- based One-Time Password) que gera um código de utilização a cada 30 segundos, até o código ser utilizado, deixando depois de gerar. Outro exemplo, mais no contexto do projeto, é a posse de um cartão-chave que pode ser lido no momento da tentativa de acesso. Este fator pode ser caraterizado pela expressão "Algo que possui".

3. Biometria

Posto de forma simples, é a informação contida na própria pessoa, ou seja, autenticação por sistema biométrico. É uma caraterística física única e próprio da pessoa em tentativa de autenticação que mais ninguém possui igual, ou pelo menos, a possibilidade de isso acontecer é muito ínfima. Inclui, não sendo limitado, a impressão digital de algum dedo da mão, da palma da mão, incluindo também a autenticação por voz, cara, ou verificação de retina e iris do olho. Este fator pode ser caraterizado pela expressão "Algo que é".

4. Localização

Este fator tal como o seu nome indica é um processo de autenticação através da localização geográfica. Uma das formas mais comuns de detetar a localização de um utilizador é através do IP do seu dispositivo, sendo que a utilização de um sistema ou aplicação de alteração de endereço de rede, como é o caso de uma *Virtual Private Network* (VPN), faz com que este método não seja tão eficaz. Por exemplo, supondo que o utilizador necessita de dar permissões a uma certa aplicação móvel para poder usufruir dela, sendo que esta apenas funciona se o utilizador estiver em Portugal, quando este estiver a tentar registar-se no sistema, a aplicação poderá recusar o registo deste utilizador caso se encontre fora do país, informando-o que não poderá utilizar a aplicação. Se alguém tentar entrar na conta com uma localização IP na Alemanha, o serviço avisará que a tentativa foi feita num sítio diferente da localização indicada como corrente. As localizações de IP não são as únicas uteis no processo deste fator, também é possível através de endereço *Media Access Control* (MAC), por outras palavras, controlo de acesso aos meios de comunicação. Uma organização ou empresa poderá implementar o acesso à sua rede exclusivamente em computadores específicos, assim, qualquer tentativa de acesso num computador diferente será recusada. Este fator pode ser caraterizado por "Um lugar onde se encontra"

5. Gestos

É um tipo de autenticação que funciona através de observação de ações ou movimentos. Estes movimentos podem ser gestos ou toques. Uma boa forma de explicar através de um exemplo é a utilização de imagens ao efetuar a autenticação. O usuário escolhe uma imagem para servir como acesso e escolhe como autenticação, por exemplo, dois movimentos circulares no centro da imagem e um toque no canto superior esquerdo. Sempre que o acesso é concedido, significa que na tentativa de autenticação está alguém que sabe o sítio exato da imagem onde tocar para o desbloqueio, através de gestos.

Baseado em diferentes níveis de segurança, a autenticação pode ser feita através da utilização de apenas um fator, chamado Fator de Autenticação Único, em que se opta apenas pela utilização de um dos fatores acima referidos. A autenticação pode também ser feita através da escolha de dois fatores, chamando-se assim Autenticação de dois fatores. Pode ainda ser feita através do método mais avançado de autenticação que é a Autenticação de múltiplos fatores, que requer a escolha de três ou mais fatores para garantir acesso ao utilizador, tornando-o assim o mais complexo, mas também o mais seguro (Ohio State University 2019), não sendo completamente infalível contra certos ataques informáticos, nomeadamente *phishing*, onde a inserção das credenciais numa página não legítima podem ser captadas.

2.3 Regulamento Geral sobre Proteção de dados

A privacidade e proteção de dados é algo que, cada vez mais, tem de ser tido em consideração, devido à potencialidade de fuga, roubo ou divulgação de informação, principalmente dados pessoais, que podem ser usados para crimes como roubo de identidade. Baseado em incidentes de fuga de informação, registados entre 2005 e 2015, dados pessoais e dados fiscais/financeiros foram os principais alvos destes ataques (Trend Micro 2018). Sendo que, os métodos mais utilizados na divulgação de informação são: pirataria, equipamentos perdidos (*flash drive*, telemóvel, computador), falha de segurança, *insider leak*[3], publicação por engano, entre outros (Trend Micro 2018).

Figura 6 – Principais métodos de divulgação de informação (Trend Micro 2018)

Juntando informação de relatórios de empresas, dos *media*, notícias governamentais e populares, estima-se que em 2021 o custo médio de uma fuga de informação a nível global terá ultrapassará os 150 milhões de dólares americanos (cerca de 138 milhões de euros), com uma previsão de 2.1 triliões de dólares americanos (cerca de 1.2 biliões de euros) no final de 2021 (Wikipedia 2019). Sendo que, apenas na primeira metade de 2018, estima-se uma fuga de 4.5 mil milhões de registos, resultado de fugas de informação (The Citizen 2018). Em 2019, um conjunto de 2.7 mil milhões de registos pessoais, consistindo em 774 milhões endereços de email e 21 milhões de palavras-passes, foram expostos na Internet para venda (Song 2019).

Tabela 1 – Exemplos conhecidos de fuga de informação (Wikipedia 2019)

ENTIDADE	ANO	REGISTOS	MÉTODO
Yahoo	2013	3 mil milhões	Pirataria
First american corporation	2019	885 milhões	Falha de segurança
Facebook	2019	540 milhões	Falha de segurança

[3] *insider leak* (sugestão de tradução: fuga interna de informação): roubo de informação através da divulgação ou partilha, a partir de um dos colaboradores da empresa

Marriott international	2018	500 milhões	Pirataria
Yahoo	2014	500 milhões	Pirataria

Endereçando agora um caso particular de fuga de informação, numa empresa que é uma referência, a nível internacional, no setor de controlo de acessos, com o seu sistema implementado em bancos mundiais, na polícia Britânica, entre muitos outros clientes, a Suprema (Suprema 2020). Este caso retrata uma brecha de segurança no sistema biométrico da empresa que implicou a partilha de mais de 1 milhão de registos de impressões digitais, reconhecimento facial e informação pessoal dos colaboradores das diferentes empresas onde o sistema está implementado (Taylor 2019).

Numa pesquisa realizada a um dos produtos mais recentes da empresa, o Biostar 2[4], foi descoberto que as base de dados internas das unidades estavam demasiado desprotegidas, com muita falta de encriptação dos dados (Taylor 2019). Isto foi possível analisar através da manipulação do endereço URL, modificando os parâmetros de pesquisa no acesso à plataforma Elasticsearch (Elasticsearch 2020), para obtenção de dados sensíveis, que não deveriam ser obtidos tão facilmente. Através desta pesquisa, os investigadores conseguirão analisar mais de 27 milhões de registos, correspondendo a cerca de 23 gigabytes de informação, desde códigos e palavras-chave de utilizador desencriptados a detalhes pessoais de colaboradores (Taylor 2019).

Com o objetivo de proteger a privacidade dos titulares, prevenindo e impedindo que os seus dados pessoais sejam alvos destes ataques e fugas de informação, foi lançado o Regulamento Geral sobre Proteção de Dados (GDPR.eu 2016), RGPD no acrónimo português, em abril de 2016, substituindo assim a antiga diretiva 95/46/CE (Parlamento Europeu e do Conselho 1995), trazendo maior responsabilidade para as empresas em qualquer lado, desde que utilizem ou armazenem informação de pessoas singulares pertencentes à União Europeia, mas não só.

O principal foco deste regulamento, são os dados pessoais, o tratamento e utilização dos mesmos. Dados pessoais, são todos aqueles que permitam a identificação direta ou indireta de uma pessoa. Estes são sem dúvida o principal tópico do RGPD, com uma definição mais alargada daquilo que podem constituir dados pessoais, sendo necessário uma proteção mais severa e encriptação dos mesmo quando do seu tratamento. Na tabela seguinte, podemos comparar as diferenças entre o que são considerados dados pessoais na diretiva 95/46/CE e o RGPD.

[4] Biostar 2: https://www.supremainc.com/en/platform/hybrid-security-platform-biostar-2.asp

Tabela 2 – Comparação da definição de dados pessoais na diretiva 95/46/CE e no RGPD

DADOS PESSOAIS	DIRETIVA 95/46/CE	RGPD
Identificáveis	Nome Números de identificação (cidadania, fiscal, utente)	Nome Números de identificação (cidadania, fiscal, utente) Localização Via eletrónica (telemóvel)
Específicos	Atributos: • Físicos • Fisiológicos • Psíquicos • Económicos • Sociais • Culturais	Atributos: • Físicos • Fisiológicos • Genéticos/biométricos • Mentais • Económicos • Sociais • Culturais

O tratamento destes dados deve seguir os princípios acordados pela lei em vigor sobre os dados pessoais, estes princípios encontram-se enumerados no 2º capítulo do RGPD. Sendo que, o tratamento deve ser feito de forma leal e lícita, seguindo sempre as normas definidas, garantindo sempre a integridade e confidencialidade dos dados.

O consentimento do tratamento dos dados pessoais é um dos aspetos legais a ter em consideração, quando se pretende aceder ou manipular estes dados. Os direitos existentes no antiga diretiva 95/45/CE, propagaram-se para o RGPD, suplementando duas novas definições: o direito de recusar que informação pessoal seja utilizada para ações de marketing e o direito à portabilidade dos dados (Klekovic 2017).

Existem severas penalizações para o incumprimento do RGPD, dividindo-se em duas partes de acordo com a gravidade da situação:

- Não cumprimento – violação de obrigações no processamento e armazenamento dos dados pessoais. Pode resultar em coimas que podem atingir os 10 milhões de euros, ou 2% da faturação do último exercício.
- Negligência – violação das condições de consentimento ou dos direitos do titular dos dados. Pode resultar em coimas que podem atingir os 20 milhões de euros, ou 4% da faturação do último exercício.

Simultaneamente às coimas, vinte vezes maiores que as retribuições máximas dos estados-membros da União Europeia, atendendo à diretiva 95/46/EC, um relevante momento judicial acompanha-se por um elemento de compensação individual. Este elemento concede aos titulares dos dados a possibilidade de intentar uma ação judicial que caso seja atendida e favorável, dá o direito de receber uma indemnização, se sofrerem danos devido a um processamento que não respeitou as regras, e dá-lhes também o direito de registar uma

reivindicação conjunta (uma prática introduzida nos princípios de “Lei Comum” norte-americana e britânica). Há também espaço para a legislação nacional elaborar medidas adicionais, tais como a detenção de responsáveis identificados de entidades legais em violação do RGPD e regulações subsequentes (Klekovic 2017).

2.4 Segurança e encriptação de dados

A segurança e encriptação de dados é um dos tópicos que é relevante abordar, visto que, um sistema de controlo de acessos necessita de armazenar informação considerada pessoal de entidades, por consequente, dados pessoais de titulares. De forma a respeitar e cumprir o RGPD, abordado previamente, é necessário considerar a utilização de um ou vários algoritmos de encriptação.

Neste subcapítulo, é abordado de forma abstrata e generalizada os princípios da encriptação de dados, descrevendo sucintamente alguns dos algoritmos que são utilizados para atingir esse objetivo.

Encriptação é a conversão de informação para uma forma de leitura inacessível para todos exceto quem autorizado. O principal objetivo é que duas partes consigam comunicar entre si, sem que mais ninguém consiga ler e/ou entender a mensagem (Buchanan B. 1999). A informação legível é convertida em informação encriptada, através de algoritmos capazes de encriptação utilizando uma chave, podendo depois ser desencriptada do lado do recetor através de um algoritmo de desencriptação, que deve corresponder ao mesmo utilizado no momento da encriptação, convertendo novamente para a informação original, tal como é possível analisar na Figura 7.

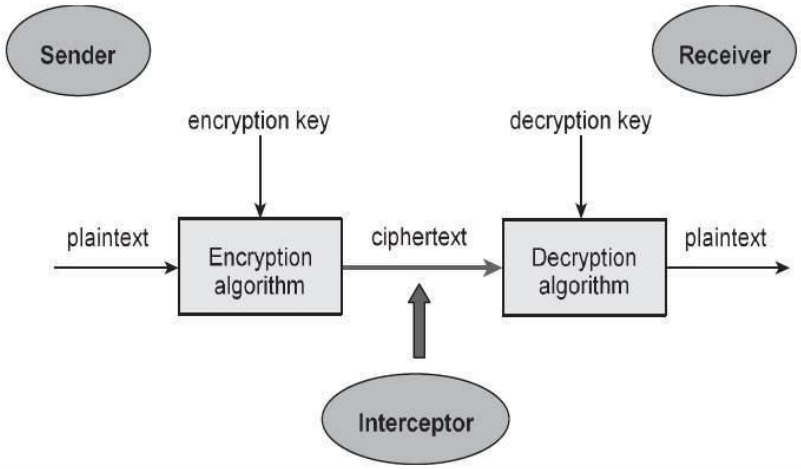

Figura 7 – Processo de encriptação e desencriptação de informação (Tutorials Point 2020)

Na mesma Figura 7, podemos constatar que na encriptação de informação existem dois momentos em que é necessária a utilização de uma chave, primeiro para transformar o texto em texto ilegível e segundo para voltar a transformar esse texto para a sua forma original.

Estas chaves podem ser iguais, isto é, a mesma chave é partilhada tanto para encriptar, mas também para desencriptar, ou então podem ser diferentes. Posto isto, baseado neste mesmo critério surgem dois métodos de encriptação: a encriptação simétrica e a encriptação assimétrica (Almeida 2017). No primeiro método, a mesma chave é utilizada tanto para encriptar como para desencriptar o texto, sendo que ambas as partes partilham uma chave para conseguir comunicar entre si. No segundo método, a encriptação assimétrica, pode não ser usado apenas uma chave, mas também um par de chaves, sendo que existem três métodos de encriptação assimétrica:

- A utilização de uma chave pública no momento de encriptação e uma chave privada no momento da desencriptação;
- A utilização de uma chave privada na encriptação e chave pública na desencriptação;
- A utilização de um par de chaves pública e privada em ambos momentos.

De seguida, realiza-se uma análise a cinco algoritmos de encriptação comuns (Bradford 2019), utilizados atualmente na maior parte das organizações, para combater as constantes ameaças que se sentem de obtenção de informação de forma ilícita, conhecido por todos como ataques informáticos piratas, tradução do termo original *hacking*. Sendo cada um dos subcapítulos descritos de seguida um algoritmo de encriptação.

2.4.1 Triple DES (Triple Data Encryption Standard)

O Padrão Triplo de Criptografia de Dados (sugestão de tradução) é um tipo de criptografia computorizada que tem por base um outro algoritmo de criptografia – DES, Data Encryption Standard, pois este foi considerado como padrão múltiplas vezes ao longo da história, porém os chamados piratas informáticos (hackers) desenvolveram modos eficazes de desencriptar os dados encriptados através deste algoritmo. Desta forma, com maior rentabilidade, o Triple DES usufrui 3 chaves de 64 bits, embora apenas 56 bits de cada chave são efetivamente usados, pois os restantes podem ser utilizados para verificar a paridade. Posto isto, perfaz-se assim um tamanho máximo de chave 168 bits (56 bits x 3 chaves). Os dados são encriptados com a primeira chave, desencriptados com a segunda e finalmente encriptados novamente com a terceira. Apesar de ser considerado um processo lento, a sua complexidade confere-lhe uma maior segurança na proteção de dados.

2.4.2 RSA (Rivest-Shamir Adleman)

RSA é um tipo de criptografia computorizada com nome em homenagem aos seus criadores, compondo a inicial de cada um deles: Ron Rivest, Adi Shamir e Leonar Adleman. O RSA é um algoritmo de encriptação que utiliza chaves públicas, sendo o algoritmo padronizado para comunicação para transmissão de informação para a *Internet* (5 common encryption algorithms, 2019). RSA é considerado um algoritmo assimétrico porque usufrui de um par de chaves, pois existe a chave pública, que é utilizada para encriptar a mensagem, e a chave privada para a desencriptar. O resultado deste método traduz-se em registos bastante

complicados de ler, sendo muito difícil saber qual o par de chaves utilizado, e como consequência é mais difícil para conseguir aceder através de pirataria informática.

2.4.3 Blowfish

É um tipo de encriptação, que tal como o Triple DES, serve para aperfeiçoar e melhorar em termos de eficácia o método DES – Data Encryption Standard. Blowfish é uma cifra simétrica que divide mensagens em blocos de 64 bits e as encripta individualmente. É um método conhecido pela sua rapidez e eficácia, com uma rede de Fiestel de 16 interações com tamanho de bloco de 64 bits, onde uma chave pode variar entre 32 e 448 bits independentes.

2.4.4 Twofish

Este é um algoritmo de encriptação proveniente do método anteriormente explicado, em Blowfish. As chaves utilizadas neste algoritmo podem ir até 256 bits de extensão e assim como nos métodos simétricos, apenas uma chave é necessária. É um dos métodos mais rápidos dentro deste tipo de métodos, superando o anterior (D. Rane 2016). O criador do método Blowfish, recomenda o seu abandono em favor deste, devido à sua enorme eficiência e maior segurança (Common Lounge 2018).

2.4.5 AES (Advanced Encryption Standard)

Este é talvez o método mais comum, padronizado pelo governo dos Estados Unidos da América e por várias e numerosas organizações (5 common encryption algorithms, 2019). Apesar de ser extremamente eficiente e eficaz com a utilização normal de chaves de 128 bits, o AES também pode ser utilizado com chaves de 192 e 256 bits para propósitos de sistemas mais robustos. É extensivamente considerado impenetrável a qualquer ataque, podendo vir assim a ser aclamado o método padrão para encriptação de dados no setor privado (5 common encryption algorithms, 2019).

2.5 Sistemas de gestão de identidade

A gestão de identidade é um dos conceitos mais importantes e críticos quando se aborda um tema como o controlo de acessos, visto que se pressupõe a necessidade de identificar e aferir as responsabilidades e permissões de uma entidade no momento da tentativa de acesso a um determinado espaço ou recurso da organização.

Neste subcapítulo, vão ser abordados e descritos conceitos relevantes que são necessários e utilizados em sistemas de gestão de identidade.

Um sistema de gestão de identidade pode ser implementado de múltiplas formas, utilizando diferentes mecanismos e padrões. São exemplos destes sistemas:

- Oracle Identity Manager
- WSO2 Identity Server
- Keycloak

De seguida, são apresentados alguns dos conceitos mais relevantes utilizados nas diferentes implementações destes sistemas.

2.5.1 Single Sign-On (SSO)

O Single Sign-On[5] é uma das propriedades de um sistema de gestão de identidade, que permite aos utilizadores a possibilidade de efetuar uma autenticação segura e única entre diferentes sistemas e aplicações, através da utilização de informação que identifica o utilizador, armazenada num local seguro, sendo que cada interação entre os diferentes sistemas e o mecanismo de autenticação é efetuada através de uma ligação segura e confiável (One Login 2020).

A utilização de um mecanismo de autenticação única e central, como o Single Sign-On, é cada vez mais necessária, visto que a utilização, por parte das organizações, de diferentes sistemas e aplicações para controlar as suas operações e tarefas diárias é cada vez maior, sendo também necessário conseguir restringir o acesso aos conteúdos e funcionalidades de cada um destes sistemas através das capacidades do utilizador autenticado (Peyrott 2015).

Como referido previamente, a autenticação utilizando SSO necessita de um elo, que seja seguro, entre os diferentes domínios das aplicações, onde a informação do utilizador está armazenada, normalmente designado por *Identity Provider* ou servidor de autenticação, sendo este o meio de ligação central capaz de garantir com que a sessão seja partilhada pelos múltiplos domínios.

[5] *Single Sign-On* (sugestão de tradução: Autenticação única ou singular) - Mecanismo de autenticação partilhada entre sistemas

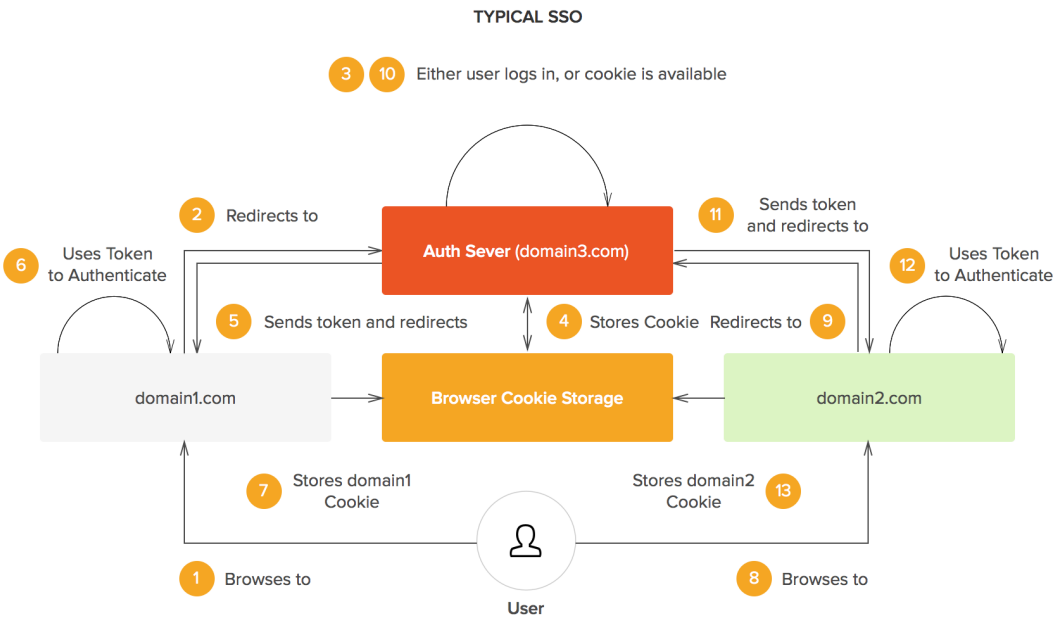

Figura 8 – Funcionamento típico de autenticação single sign-on (Peyrott 2015)

Na Figura 8, também disponível para consulta no anexo A, podemos verificar o funcionamento típico do mecanismo SSO num ambiente web com múltiplos sistemas/domínios. Na tentativa de aceder a um dos domínios por parte de um utilizador, o servidor de autenticação providenciará um identificador de sessão assinado e encriptado designado *JSON Web Token* (JWT), que será partilhado para os diferentes sistemas, onde será utilizado para garantir a autenticidade de sessão. Este *token* pode ser armazenado pelos diferentes domínios de forma a garantir a autenticação, sempre que o utilizador tenta aceder ao mesmo. O *token* poderá ser configurado para ter validade, sendo que caso a validade deste expire, deixará de ser válido para qualquer sistema que o tenha armazenado. Neste caso, o processo de autenticação e geração deste recurso terá de se repetir, mas simplificando bastante o processo de autenticação, visto não ser necessário a introdução de credenciais do utilizador a cada tentativa de acesso aos diferentes sistemas.

2.5.2 Autenticação federada

A autenticação federada ou gestão de identidade federada requer uma relação de confiança entre os diferentes sistemas, onde as credenciais do utilizador são armazenadas, na qual o mesmo pode ser autenticado nos múltiplos domínios sem sequer necessitar de introduzir ou conhecer as credenciais utilizadas (Broeckelmann 2017).

A federação baseia-se na utilização de um conjunto de padrões comuns e protocolos com o intuito de gerir, identificar e mapear o utilizador, através de informação armazenada num

domínio capaz de providenciar identificação, partilhando uma relação de confiança com os diferentes sistemas através de métodos como assinaturas digitais ou utilização de infraestrutura de chaves públicas.

Este tipo de autenticação pode ser atingido de múltiplas formas e implementações, como por exemplo a utilização de padrões formais como Security Assertion Markup Language (SAML) e tecnologias como é o caso do OpenID Connect (uma das camadas de autenticação sobre a *framework open source* OAuth 2.0), que permite a utilização dos seus mecanismos de autenticação em implementações nas mais variadas linguagens e tecnologias de programação. Estes padrões e mecanismos autorizam a utilização de autenticação federada e fornecem uma ligação segura entre os diferentes sistemas e o utilizador, na tentativa de autenticar e autorizar o acesso. Por sua vez, necessitam de uma identidade federada, que armazena a identificação dos utilizadores e permite o acesso/autenticação em sistemas externos, tendo o utilizador uma conta registada na identidade federada em questão e autorizado o acesso aos dados da sua conta, para se autenticar noutros domínios. Exemplos de plataformas digitais federadas que permitem a utilização da informação dos seus utilizadores, mediante autorização dos próprios, no acesso a sistemas externos são (Okta 2021):

- Microsoft Account
- Google Account
- Amazon
- Github

2.5.3 Auto autenticação

Apesar da existência de diferentes mecanismos de Single Sign-on abordados previamente, algumas organizações necessitam de saber mais informações sobre os utilizadores quando estes são contratados e necessitam de aceder ao sistemas e aplicações das mesmas. Para que o processo de autenticação funcione, com ou sem a utilização de Single Sign-on, a requisição de informação dos utilizadores é necessária para que os mesmos sejam registados e reconhecidos pelo servidor de autenticação, mas este processo pode ser realizado sem a necessidade de interação com os membros da organização.

O processo de auto autenticação acelera o registo de novos utilizadores aquando da chegada de novas contratações às organizações, visto que estes têm a possibilidade de se registar sem a necessidade dos membros da organização requisitar qualquer informação, fazendo com que o processo de registo de novas entidades seja eficiente.

2.6 Corrente tecnológica

Neste subcapítulo pretende-se apresentar, assim como o nome indica, a corrente tecnológica que se estabelece atualmente, que deverá ser analisada a partir do momento que se pretende desenvolver um sistema estado da arte, num ambiente web.

Apresenta-se algumas tecnologias, *frameworks* e linguagens de programação, em termos de popularidade, suportabilidade e prosperidade, que devem ser consideradas, no momento da escolha da arquitetura e tecnologias a utilizar no desenvolvimento e implementação de uma solução de software. São apresentados alguns padrões de *design* arquitetural e de software, também a ser tidos em consideração aquando da escolha da arquitetura a implementar num sistema de software de origem e natureza web.

2.6.1 *Frameworks,* tecnologias e linguagens

Nesta secção são apresentadas as principais *frameworks*, tecnologias e linguagens de programação utilizados no desenvolvimento de soluções, nesta área de inovação digital e de crescimento e expansão do ambiente web.

De forma a facilitar a separação destas tecnologias e linguagens, será feita a divisão entre três áreas de desenvolvimento web distintas, sendo elas: o Front-end, o Back-end e os motores de base de dados.

2.6.1.1 Front-End

Relativamente ao front-end, existem algumas linguagens e *frameworks* que se destacam pela sua versatilidade, ganhando popularidade entre os desenvolvedores de sistemas web. De seguida, apresenta-se as respetivas linguagens de programação mais populares, baseado num inquérito realizado em 2019, aos desenvolvedores registados numa das maiores plataformas de partilha de conhecimento, o Stack Overflow[6] (Stack Overflow 2019).

1. JavaScript

Juntamente com as linguagens HTML e CSS, o JavaScript é a linguagem de programação web mais comum, sendo a mais popular e geralmente utilizada pelo maior parte dos websites, nos quais se navega diariamente (Yang 2020).

Devido à sua enorme flexibilidade na sintaxe e suporte para a maior parte, senão todos, os *browsers* da atualidade, esta é uma linguagem normalmente aconselhada para iniciantes no mundo da programação (Yang 2020).

2. React Native *Framework*

Esta *framework* é baseada na linguagem de programação JavaScript, apresentada anteriormente, é uma das *frameworks* com maior emergência no desenvolvimento tanto de aplicações web como para aplicações nativas para *mobile*. Tendo sido disponibilizada pela organização Facebook em 2015, sendo utilizada atualmente por várias aplicações com grande popularidade, desde o próprio Facebook ao Skype, sendo também utilizado por outras organizações no ramo automóvel, como por exemplo, a Tesla (Facebook Open Source 2020).

3. Typescript e Angular *Framework*

[6] Stackoverflow - https://stackoverflow.com/

Outra linguagem com crescente popularidade no desenvolvimento de aplicações web é o TypeScript, propriedade da Microsoft. Esta é uma variação da linguagem JavaScript, fornecendo ao programador uma interface e estrutura mais amigável ao desenvolvimento, sendo compilada posteriormente para JavaScript, conseguindo correr qualquer aplicação desenvolvida com esta linguagem em qualquer *browser* (Microsoft - Typescript 2020).

A *framework* Angular, propriedade da Google, utiliza esta linguagem que através da utilização conjunta com as linguagens HTML e CSS, formam esta *framework* que pode ser utilizada tanto para aplicações web desktop, como para aplicações de natureza Mobile (Google 2020).

2.6.1.2 Back-End

Relativamente a tecnologias e linguagens para desenvolvimento de aplicações de *back-end*, destacam-se as seguintes:

1. .NET Core e C#

Esta tecnologia/*framework open-source*, propriedade e desenvolvido principalmente pelo Microsoft, é o sucessor do .NET Framework, tendo sido lançado em 2014 através da sua primeira versão .NET Core 1.0. Atualmente, já se encontra na 3ª versão, designada .NET Core 3.1 (Microsoft - .NET Core 2020).

Esta *framework* suporta totalmente as linguagens de programação C# e F#, suportando também de forma parcial a linguagem Visual Basic. Sendo que, entre estas linguagens o foco principal vai para o C#, sendo esta a mais popular no desenvolvimento web em conjunto com a *framework* .NET Core.

2. Node.js

Node.js é um *runtime* baseado em JavaScript, que pode ser utilizado para o desenvolvimento de aplicações *back-end*. Devido ao seu único modelo de I/O (*Inputs/Outputs*), esta é bastante escalável, eficiente e leve, sendo uma excelente alternativa quando se procura desenvolver uma aplicação de *back-end* simples (Rachowicz 2017).

3. Java e Scala

O Java é uma linguagem de programação que dispensa introduções, estando no topo das linguagens de programação durante os últimos anos. Esta linguagem é bastante versátil e geralmente é utilizada em aplicações, como uma linguagem orientada a objetos, sendo bastante relacional. A linguagem Java é utilizada para o desenvolvimento de aplicações para diferentes plataformas, como aplicações desktop, aplicações web e mobile. O Scala é uma linguagem de programação de alto-nível baseada na anterior, combinando dois paradigmas: desenvolvimento de software orientado a objetos e programação funcional.

2.6.1.3 Base de dados

Quanto aos motores de base de dados, existem alguns aspetos que devem ser considerados, nomeadamente se a solução de software a desenvolver necessita de um motor de base de dados relacional. Assim, são apresentadas as diferentes bases de dados, relacionais (SQL) e não relacionais (NoSQL), existentes e que devem ser consideradas.

Figura 9 – Logótipos de bases de dados relacionais (Data Science Academy 2016)

Na Figura 9 apresenta-se os logótipos de algumas bases de dados relacionais[7], tipicamente utilizadas, quando se pretende garantir que o motor de base de dados cumpre alguns requisitos do sistema, como por exemplo, a existência de propriedades ACID (*Atomic, Consistency, Isolation e Durability*): atomicidade, consistência, isolamento e durabilidade (Wenzel 2020).

Em contraste a estas bases de dados apresentadas, existem as bases de dados não relacionais, apresentando cada uma delas as suas capacidades e as suas complicações. As principais bases de dados NoSQL, ou seja, não relacionais, atualmente mais proficientes são:

- MongoDB
- Firebase
- Apache Cassandra
- Redis

2.6.2 Padrões de *design* arquitetural e de software

Em termos arquiteturais, deverão ser observados e considerados alguns fatores e características de software, como por exemplo, o desempenho, a baixa tolerância a falhas, a escalabilidade e confiabilidade (Rabelo 2019).

Atualmente, são cada vez mais comuns termos como "*MicroServices*" e "*Software as a Service*" (Rabelo 2019), estes são ambos padrões arquiteturas, consistindo em implementações de software distintas.

[7] Oracle – https://www.oracle.com/pt/database/
Microsoft SQL Server – https://www.microsoft.com/pt-pt/sql-server/sql-server-2019
IBM DB2 – https://www.ibm.com/analytics/db2
PostgreSQL – https://www.postgresql.org/
SQLite – https://www.sqlite.org/index.html
MySQL – https://www.mysql.com/

De seguida, são apresentados alguns padrões de *design* arquitetural, bastante populares nas soluções de software de ambiente web desenvolvidas atualmente.

1. Arquitetura de microserviços

Esta arquitetura tem-se tornado na mais popular nos últimos anos (Rabelo 2019), devido às suas características modulares fornecendo assim, um carácter modular, altamente manutenível e escalável à solução arquitetural.

Esta consiste no desenvolvimento de pequenos serviços, responsável especificamente pela funcionalidade atribuída, comunicando entre si através de uma rede bem definida garantindo a integridade dos dados.

2. Arquitetura *Serverless*

Esta arquitetura baseia-se no fornecimento de serviços, que dependem de sistemas externas para manter e gerir os processos e complexidade do *back-end*. Esta arquitetura pode ser traduzida através de termos como: *Back-end as a Service* (BaaS) ou *Software as a Service* (SaaS) (Rabelo 2019).

3. Arquitetura baseada em eventos

Esta arquitetura depende da existência de dois elementos: os produtores de eventos e os consumidores de eventos. Esta arquitetura pode ser entendida através do seguinte exemplo, imaginando um sistema de comércio online, onde um cliente compra algum item existente na loja, esta ação irá despoletar um evento "compraPendente", sendo que, qualquer serviço que esteja à escuta deste evento, irá consumi-lo podendo despoletar um novo evento ou finalizar a ação (Rabelo 2019).

No momento da decisão da arquitetura a implementar, deverá também ser considerado alguns padrões de software, que influenciam direta ou indiretamente a escolha do *design* arquitetura. Um dos padrões mais conhecidos no desenvolvimento de software, maioritariamente em soluções relacionais e orientada a objetos, é o padrão **SOLID** (Rabelo 2019):

- ***S**ingle Responsability Principle*
- ***O**pen Closed Principle*
- ***L**iskov Substitution Principle*
- ***I**nterface Segregation Principle*
- ***D**ependency Inversion Principle*

Outros padrões de software, de diferentes tipos como estrutural, comportamental e de concorrência, que são utilizados e devem ser considerados durante o desenvolvimento de soluções são:

- *Factory Pattern*
- *Repository Pattern*
- *Dependecy Injection*

- *Singleton*
- *Adapter*
- *Mediator*
- *Strategy*

3 Análise de valor

Neste capítulo será realizada a análise de valor do projeto, adaptando e relacionando a mesma com o modelo NCD (*New Concept Development*) de Peter Koen (Peter Koen 2014). Será ainda elaborado o valor da solução para o cliente, apresentando os benefícios e sacrifícios relativos ao mesmo, enunciando a proposta de valor da solução. Para finalizar, será ainda apresentado o modelo Canvas, onde se poderá analisar factores como parcerias e relação com os clientes.

3.1 Modelo New Concept Development

De acordo com o modelo *New Concept Development*, de Peter Koen, são distinguidas três secções, sendo elas:

- Motor (traduzido de "engine"): fornecendo força criativa para a geração e enriquecimento da ideia, fortalecendo o processo de inovação;
- Roda (traduzido de "wheel"): designação para as cinco principais atividades no processo de inovação e desenvolvimento;
- Borda (traduzido de "rim"): designação para os fatores externos não controláveis.

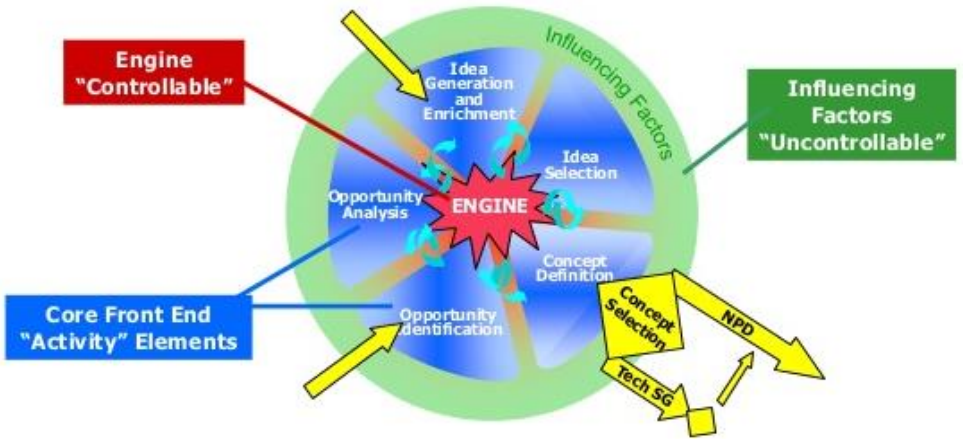

Figura 10 – Modelo *New Concept Development* (Peter Koen 2014)

Segundo Peter Koen, o motor que é o elemento central deste modelo, promove a liderança e estratégia, dando força ao processo criativo e de inovação, que se *design* de "roda" no modelo, focando-se na ideia a desenvolver, desde a análise de oportunidade ao enriquecimento e todo o processo de definição, nunca esquecendo e ignorando os fatores não controláveis caracterizados por "borda" a verde na Figura 10 (Peter Koen 2014).

De seguida, explora-se as cinco atividades chaves, as quais se encontram na "roda" do modelo, que impulsionam a ideia e tratam da sua definição de conceito. Na Figura10, podemos ver

duas setas a amarelo sem texto, que apontam para os pontos de início onde o modelo pode começar. Neste caso, a ideia surgiu através da identificação da oportunidade por parte da empresa, portanto vamos começar por enquadrar a ideia do projeto por esta atividade.

3.1.1 Identificação e análise da oportunidade

Vamos juntar a primeira e segunda atividade num único ponto de modo a facilitar ao leitor a compreensão da oportunidade. A primeira atividade deste modelo é a identificação da oportunidade que, tal como o próprio nome indica, consiste em identificar oportunidades de negócio, sejam estas ideias para produção de produtos tangíveis ou serviços no âmbito tecnológico. A segunda atividade consiste numa análise mais detalhada da ideia/oportunidade, de forma a conseguir-se determinar se a oportunidade pode representar uma mais-valia para o mercado.

A ideia identificada como oportunidade, que resultou no tema da presente dissertação, é obtida através da preocupação da empresa em reformular e melhorar o sistema de controlo de acessos existente na empresa. Após ser identificada a oportunidade no desenvolvimento de uma nova solução de acessos, é necessário analisar se esta oportunidade representa uma mais-valia para a empresa e seus clientes. Para isto, a empresa decidiu fazer uma análise de mercado, através de inquéritos e apresentações desta nova ideia de modo a receber *feedback* e perceber se esta oportunidade apresenta alguma viabilidade. Através da avaliação dos resultados dos inquéritos e do *feedback* obtido, determinou-se que o desenvolvimento de uma nova solução de controlo de acessos pode representar uma mais-valia.

3.1.2 Génese e enriquecimento da ideia

Esta fase representa o amadurecimento da ideia, através de processos iterativos, com recurso a várias reuniões de *brainstorming*[8], onde vão ser apresentadas diferentes sugestões de como se poderá desenvolver a ideia.

Após a identificação e análise da oportunidade, de onde surgiu a ideia do desenvolvimento de um sistema de controlo de acessos, foram surgindo algumas questões e alternativas de como a oportunidade ia ser explorada e como a ideia seria abordada e desenvolvida.

A principal questão identificada foi, se deveria este sistema ser desenvolvido num ambiente web ou não? Esta questão prende-se com o facto de os equipamentos prévios à nova TUX 500, não possuírem a capacidade de comunicar através do protocolo HTTP. O sistema operativo deste terminal é baseado no projeto de *browser open source* Chromium (Google - Chromium OS 2020), que lhe confere a possibilidade de correr aplicações web sobre um navegador de Internet, conseguindo comunicar com outros recursos através de HTTP.

[8] *Brainstorming* (sugestão de tradução: tempestade de cérberos) – termo utilizado para representar reuniões com vários "cérberos" juntos, cujo intuito é obter ideias e tirar conclusões

Outras questões também foram colocadas em cima da mesa, como por exemplo, seria possível desenvolver uma camada de comunicação intermédia, que tratava da comunicação das unidades anteriores com os serviços desenvolvidos em ambiente web, sem afetar a *performance* das mesmas e a experiência de utilizador.

3.1.3 Seleção da ideia

Através da fase anterior de criação e enriquecimento de ideias, são levantadas algumas questões que se prendem com as mesmas, influenciando assim esta fase de decisão. Compete à fase de seleção da ideia, identificar e avaliar quais as ideias que vão ser mais benéficas no desenvolvimento do produto, analisando como se pode maximizar a relação benefício/custo para a empresa.

Nesta fase, cabe aos líderes da empresa e de produto, definirem qual o caminho a tomar no desenvolvimento do produto, tendo sempre em consideração todas as sugestões e ideias discutidas previamente pelas equipas de desenvolvimento de software.

Tendo em conta todo o debate entre diferentes intervenientes nas fases anteriores, decidiu-se seguir em frente pelo desenvolvimento do produto num ambiente web, devido a todas as vantagens e todo o estado da arte de produção de software, assim existem várias alternativas quanto à sua implementação e design arquitetural, sendo que a experiência do utilizador deverá ser sempre garantida e a infraestrutura sólida, tendo sempre em conta a política de qualidade da empresa.

3.1.4 Definição de conceito

Chegamos agora à fase final do modelo NCD, onde se define como será desenvolvido o produto, quais os caminhos a seguir e o que se pretende num protótipo final de um sistema de controlo de acessos, a ser desenvolvido.

Deve ser bem definido quais as tecnologias e abordagens a utilizar, uma vez que a solução se vai enquadrar num ambiente web, podendo ser desenvolvida através de várias *frameworks,* diferentes padrões e desenhos arquiteturais diferentes.

Independentemente das tecnologias a serem utilizadas no desenvolvimento da solução, o sistema deverá garantir a integração e comunicação com as unidades sempre seguras, respeitando as normas de segurança e o RGPD. Deverá também, assegurar que o produto seja completamente parametrizável por parte do cliente, sempre com grandes opções de personalização. Não esquecendo a parte mais importante da solução, que é assegurar o controlo de acessos, negando o acesso não autorizado a zonas, com tempos de resposta mínimos.

3.2 Benefícios e sacrifícios

Relativamente aos benefícios e sacrifícios pretende-se, naturalmente, que as vantagens na utilização do produto, serviço e na relação empresa-cliente sejam mais vantajosas do que prejudiciais, isto é, no âmbito do produto final como um todo é de esperar que os benefícios se superem aos sacrifícios para o consumidor final.

Posto isto, na Tabela 3 são apresentados os principais pontos positivos e negativos do produto final no âmbito do cliente.

Tabela 3 – Apresentação dos benefícios e sacrifícios para o cliente

Benefícios	**Sacrifícios**
• Produto de acordo com as políticas de qualidade da empresa; • Qualidade e eficiência nos serviços fornecidos pela empresa; • Técnicos com elevadas competências técnicas; • Confiabilidade e responsividade altas do produto; • Relação próxima e personalizada empresa-cliente; • Personalização do produto.	• Preço pode ser considerado *premium* mediante personalização; • Tempo de aprendizagem pode ser elevado; • Tempo de configuração da aplicação aumenta mediante complexidade da parametrização.

3.3 Proposta de valor

Este projeto visa o desenvolvimento de um sistema de controlo de acessos, para realizar a gestão de acessos a zonas condicionadas em pequenas, médias ou grandes empresas e superfícies. Este sistema vai permitir conectar-se com as unidades de marcações, desenvolvidas pela empresa, no âmbito de utilizar os algoritmos de autenticação já existentes para controlo o acesso de entidades a certas zonas.

Pretende-se assim, que o sistema desenvolvido permita a parametrização e gestão de zonas controladas, configuração de unidades, entidades, viaturas e visitas de acessos, consulta de marcações, geração de alarmes e envios de alertas, abertura de portas mediante autorização, entre outras funcionalidades ainda por determinar ou a serem solicitadas via personalização.

Com base na análise a produtos da concorrência com objetivos semelhantes, naquilo que o nosso produto pode oferecer e considerando as normas e políticas de qualidade da empresa, a propostas de valor para esta solução pode ser descrita através da seguinte expressão:

“Desenvolvemos soluções para a área de dos Recursos Humanos que promovem a eficiência, o rigor e a transparência no interior das organizações, reforçando a relação entre colaborador e empresa, promovendo sempre a produtividade e bem-estar dia após dia.”

3.4 Modelo Canvas

Com o intuito de resumir o plano de negócios da empresa relativamente ao produto, foi elaborado um modelo de negócios Canvas, apresentado na Figura 11.

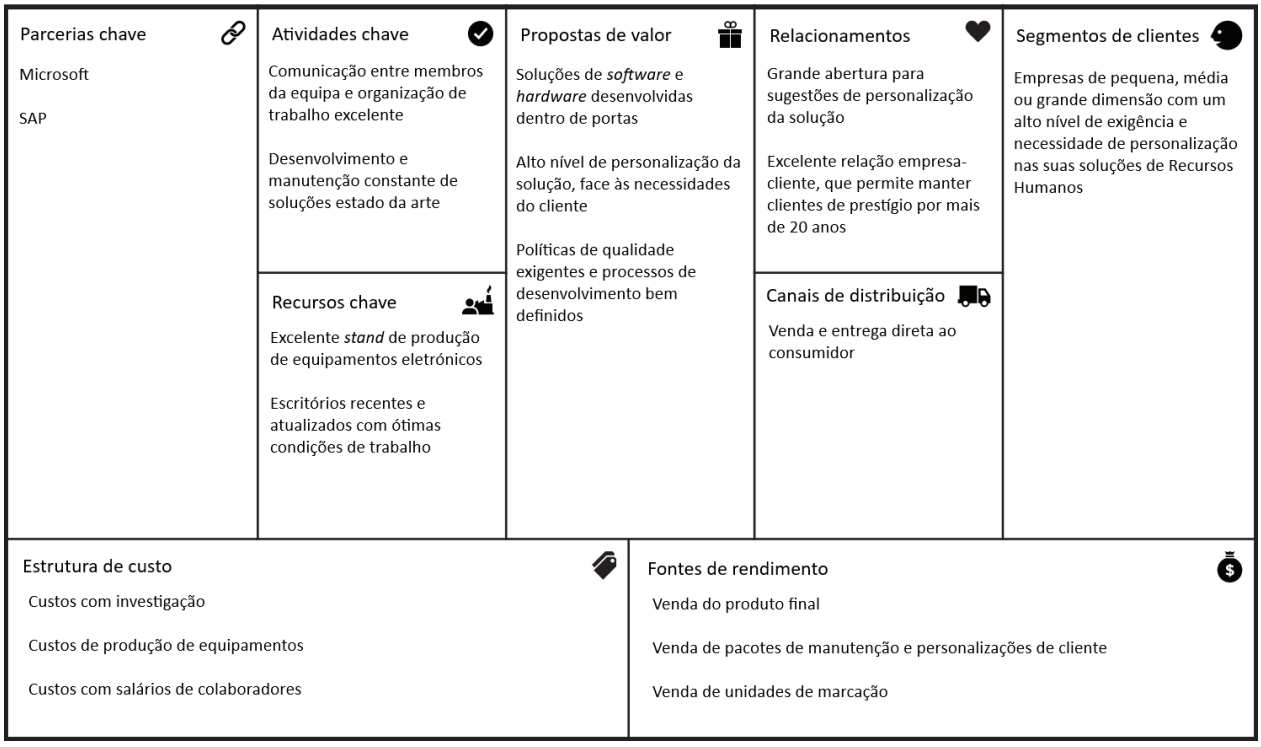

Figura 11 – Modelo Canvas

4 Análise e Design

No presente capítulo, descreve-se a análise e *design* realizado, enunciando algumas alternativas de implementação, para dar resposta ao problema descrito.

Será elaborada a modelação de negócio, com o intuito de fornecer ao leitor uma contextualização de todos os conceitos da solução, definindo o modelo de domínio. De seguida, é descrita a engenharia de requisitos realizada para a solução, começando por descrever os atores do sistema, seguido da análise dos requisitos funcionais e dos requisitos não funcionais com a ajuda do modelo FURPS+.

Ao longo do capítulo serão apresentados diagramas UML[9], representado a arquitetura proposta para o desenvolvimento da solução, através das vistas lógica e de implantação selecionadas, assim como alternativas de *design* para cada uma delas.

4.1 Modelação de negócio

A modelação de negócio pretende fornecer ao leitor uma maior compreensão do domínio do sistema, contextualizando o mesmo ao tema, através da descrição dos conceitos principais do sistema de controlo de acessos desenvolvido. Este trabalho é importante porque permite estabelecer, através de uma linguagem comum, uma melhor compreensão do tema para todas as partes interessadas.

Para a realização deste trabalho de modelação do negócio, foi desenvolvido um modelo de domínio, com a ajuda de um diagrama UML de classes. Este permite representar todas as classes conceptuais existentes no domínio do sistema, representando também todas as relações entre as mesmas. Através da análise de todos os conceitos e do desenho deste diagrama, será possível ficar a conhecer todos os conceitos que darão origem às classes de software, que são necessárias para a definição da solução.

Assim, baseado em toda a definição e análise inicial do projeto, foi possível elaborar e desenhar o modelo de domínio, visível na Figura 12, contemplando todos os conceitos relacionados com o tema do projeto, que são necessários para o desenvolvimento da solução e de todas as funcionalidades existentes na mesma.

[9] UML (Unified Modeling Language) – para auxílio na leitura e compreensão dos artefactos apresentados, consultar https://www.javatpoint.com/uml

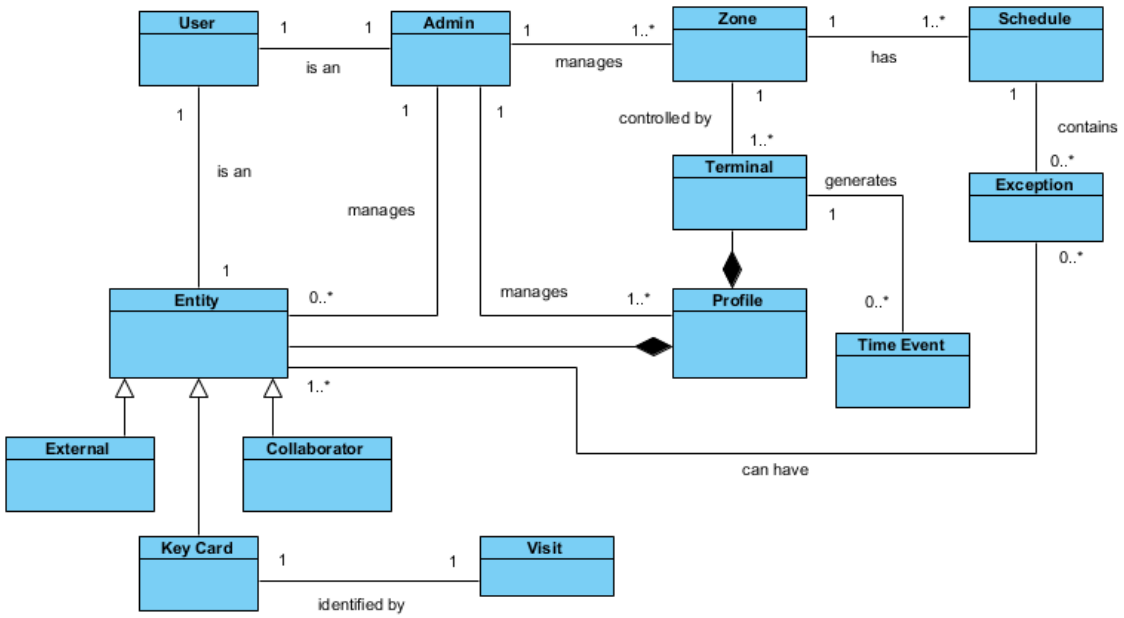

Figura 12 – Modelo de domínio da solução

Na Figura 12 é possível verificar as classes conceptuais que pertencem ao domínio da solução, bem como a relação que existe entre elas, através da análise do diagrama de classes UML elaborado. De forma a fornecer ao leitor uma melhor contextualização ao tema do projeto, são descritas as classes de domínio, referindo o seu propósito e enquadramento na solução.

1. Site (Zona)

Zona é o conceito base de todo o sistema, surgindo através deste conceito a necessidade criar um sistema capaz de garantir e controlar o acesso a espaços fisícos. O controlo de acessos pressupõe a configuração e existência de zonas, que em termos reais podem ser salas de aulas ou reuniões, *data centres*, escritórios, parques de estacionamento, entre muitas outras.

2. Terminal (Terminal de marcação)

O terminal de marcação, ou unidade de marcação, é o elemento que trata de controlar o acesso às zonas, como pode ser visto através da relação que existe entre ambos no diagrama. Uma zona pode ser controlada por uma ou várias unidades, normalmente existem duas unidades, uma a controlar as entradas e outra a controlar as saídas, mas pode ser uma zona muito grande, como exemplo um parque de estacionamento com várias entradas, onde será necessário existirem vários terminais a controlar o acesso.

3. Profile (Perfil)

O perfil de acesso, estabelece a ligação entre todas as entidades, conceito descrito posteriormente, e os terminais a quais estas têm acesso e são reconhecidas. Posto isto, qualquer entidade para ter acesso a uma zona, necessita de estar ligada a um perfil de acesso, conseguindo através deste conceito controlar quais entidades possuem acesso e são identificados numa zona.

4. Schedule (Horário)

O horário é um conceito que através do nome é possível perceber a necessidade deste no controlo de acessos. Uma zona necessita de ter um horário de acesso, capaz de através do tempo exato da tentativa de acesso conseguir determinar se é ou não possível aceder aquela zona em questão.

5. Exception (Exceção)

Uma exceção é algo que permite realizar uma operação excecional ao funcionamento do sistema. Como se pode ver no modelo de domínio, é algo que garante a um conjunto de entidades selecionadas acesso ou não a uma zona num horário especificado, sobrepondo-se à configuração prévia da zona.

6. Key Card (Cartão-chave)

Um cartão chave é um conceito que irá existir como uma classe de software da solução, sendo este um objeto físico (um cartão), que controlará através dos perfis atribuídos ao mesmo, o acesso a determinadas zonas.

7. Visit (Visita)

Uma visita, tal como o nome indica, é algo que será temporário no sistema. Pode ser, por exemplo, uma reunião de trabalho com um parceiro, onde as pessoas visitantes terão acesso aos escritórios através da atribuição de um cartão chave a essa visita, que como já foi referido previamente, um cartão chave terá um perfil atribuído que determinará os terminais aos quais terão acesso.

8. Time Event (Marcação)

A marcação é um conceito virtual que consiste no registo de uma tentativa de acesso a uma zona, registando o terminal, a entidade e a hora na qual foi realizada a tentativa de acesso à zona.

9. User (Utilizador)

O utilizador é uma classe abstrata nesta solução de software, sendo qualquer pessoa que interage com o sistema, independentemente das permissões que esta possua na utilização do sistema. Sendo esta solução direcionada para a área de atividade de Recursos Humanos, é imprescindível que existam utilizadores do sistema.

10. Admin (Administrador)

O administrador é um tipo de utilizador do sistema, que tratará da parametrização e gestão de todos os conceitos do sistema, definição de zonas, gestão de entidades e perfis, como é possível verificar através das relações existentes entre os conceitos na Figura 12.

11. Entity (Entidade)

Entidade, como o próprio nome indica, é o conceito que referencia uma entidade, que será registada no sistema com o intuito de controlar o acesso desta às zonas existentes. Estas entidades são geridas pelo administrador do sistema, sendo o seu acesso determinado pelo perfil associado.

12. External Entity (Entidade externa), Collaborator (Colaborador) e Vehicle (Veículo)
Entidade externa, colaborador e veículo são tipos de entidades registados no sistema, como é possível verificar através da análise da figura 12, cada um deles com uma identificação única, sendo o seu acesso gerido pelo administrador do sistema.

4.2 Engenharia de requisitos

A engenharia de requisitos é a disciplina responsável por fornecer toda a documentação dos requisitos do sistema, envolvendo todas as atividades em volta destes, como por exemplo, a definição inicial dos atores do sistema e dos requisitos funcionais e não funcionais.

Neste subcapítulo vão ser apresentados, inicialmente, os atores principais do sistema, seguido de todos os requisitos funcionais e não funcionais. Começando com os requisitos funcionais, com a ajuda de um diagrama de casos de uso, seguido de uma breve explicação dos mesmos. Terminando com apresentação dos requisitos não funcionais, através do modelo FURPS+.

4.2.1 Atores do sistema

Um ator do sistema é qualquer um que interaja com o mesmo, estes podem ser pessoas, organizações ou sistemas externos. Desde que tenham influência e interajam com o sistema são considerados atores do sistema, desde que usufruam de qualquer funcionalidade desenvolvida, o que significa que um *stakeholder*[10] pode não ser considerado ator do sistema. Num sistema de controlo de acessos, a interação humana com o sistema é imprescindível, daí que os atores apresentados de seguida são ambos de origem humana, sendo eles:

1. Administrador do sistema – como foi referido previamente na explicação da classe conceptual no domínio do sistema, o administrador de sistema trata de toda a gestão, desde parametrização e definição de zonas à gestão de entidades que terão acesso às mesmas zonas.
2. Entidade – uma entidade é qualquer interveniente no sistema, este pode ser classificado como colaborador, entidade externa. Interage com o sistema através das unidades de marcação, que tratam do controlo de acessos na entrada e saída das zonas definidas. Aqui, apesar da representação no modelo de domínio, vamos também considerar uma visita como sendo uma entidade, visto que uma visita não terá, de uma perspetiva de requisitos, nenhuma diferença das entidades referidas previamente.

[10] *Stakeholder* (sugestão de tradução: parte interessada): entidade com algum interesse na solução desenvolvida

4.2.2 Requisitos funcionais

Os requisitos funcionais representam todos as funcionalidades do sistema, ou seja, aquilo que o sistema consegue fazer através da interação com o mesmo (Guru99 – Functional Requirements 2020). Estes podem ser representados como casos de uso, sendo que estes ajudam a entender as ações e como estas se passam na interação de um utilizador ou um sistema externo com o nosso sistema.

Assim, com a ajuda de um diagrama UML de casos de uso, vão ser apresentados todos os requisitos funcionais através da sua representação como casos de uso, seguido de uma breve explicação dos mesmos, bem como alguns exemplos da interação do ator do caso de uso com o sistema, através da utilização de diagramas UML de sequência. Todos os diagramas UML são apresentados em português, ao contrário do resto do diagrama de classes apresentado, devido a serem textos bastante extensos, fazendo com que seja desnecessário fazer a tradução dos mesmos no documento.

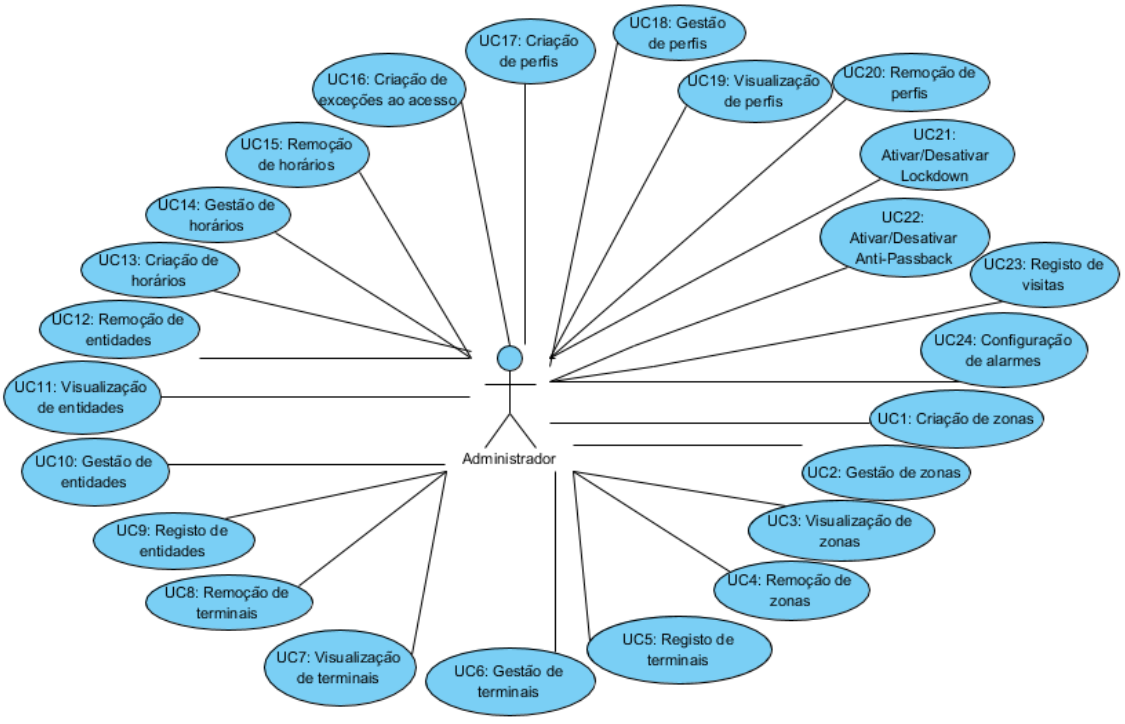

Figura 13 – Diagrama de casos de uso do administrador

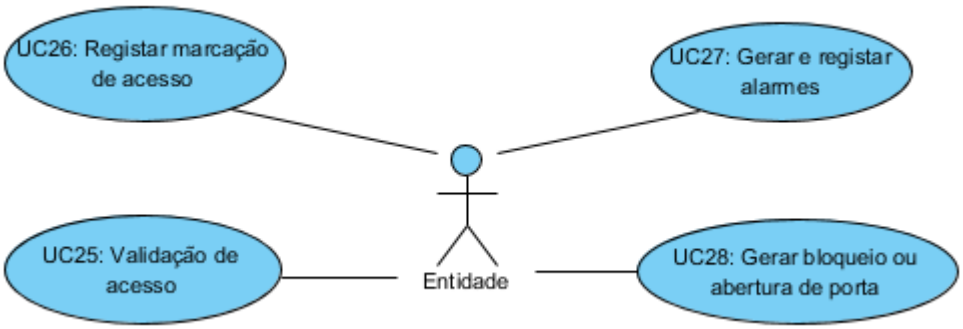

Figura 14 – Diagrama de de casos de uso de entidades

Nas figuras 13 e 14 estão apresentados todos os casos de uso para os atores do sistema, o administrador e a entidade respetivamente, representando assim todas as funcionalidades do sistema.

De seguida, será apresentada uma breve descrição para cada um dos casos de uso identificados, bem como uma representação gráfica, com a ajuda de um diagrama de sequência de sistema (da tradução de SSD: *System Sequence Diagram*), com o intuito de demonstrar a interação de cada um dos atores com o sistema. Sendo que, de forma a abreviar a descrição e explicação de todos os casos de uso, agrupam-se casos de uso devido às semelhanças existentes na sua funcionalidade e fluxo.

UC1 até ao UC20: Criar, visualizar, gerir e remover entidades de sistema

Os respetivos casos de uso foram agrupados, como foi referido previamente, devido à semelhança das funcionalidades, correspondendo à criação, visualização, gestão e remoção entidades existentes no sistema, mais concretamente as entidades que representam os conceitos base, como: zonas, terminais, entidades (colaboradores, externos, veículos), horários e perfis.

A funcionalidade destes casos de uso corresponde ao típico CRUD (*Create, Read, Update, Delete*) de classes numa aplicação de software, mais especificamente numa API, que é um dos componentes que irá existir na nossa solução, como se vai poder verificar posteriormente, no capítulo de Design arquitetural.

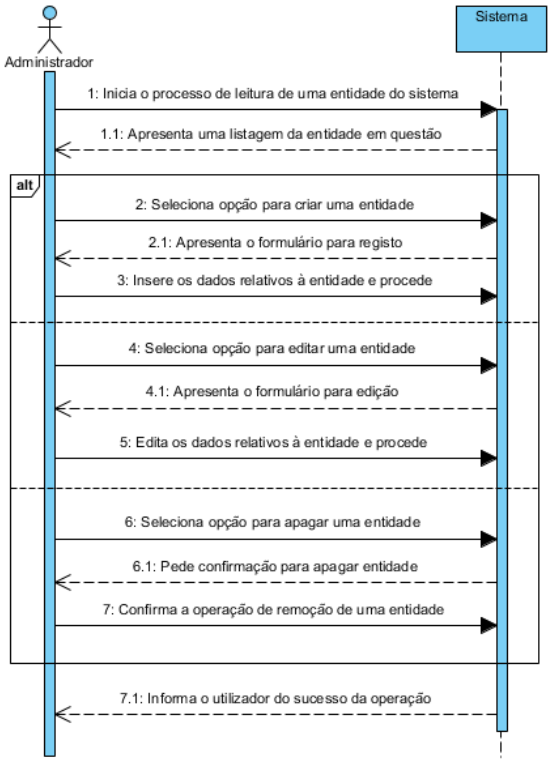

Figura 15 – Diagrama de sequência de sistema de CRUD de entidades

Na Figura 15 está caracterizado o comportamento típico de um CRUD de entidades, representado através da interação de um ator do sistema, neste caso o administrador, com o mesmo. Através da análise do diagrama, é possível verificar que o administrador inicia o processo de listagem de uma entidade, e a partir daí terá acesso a todas as outras funcionalidades: criação, edição e remoção.

UC21 e UC22: Ativar/Desativar *Lockdown* e Ativar/Desativar *Anti-Passback*

Estes casos de uso são idênticos na sua funcionalidade, visto que, ambos tratam de casos excecionais no momento da avaliação do acesso.

No caso de uso de ativar/desativar *Lockdown,* ou modo confinamento, o administrador sinaliza a unidade de que as zonas se encontram com acesso restrito, tanto para entradas como para saídas, daí o nosso confinamento. O administrador ativa esta opção selecionando um conjunto de pessoas que possam aceder aos espaços abrindo as entidades bloqueadoras, ou seja, uma vez ativa esta opção ninguém consegue desbloquear a entrada a não ser as entidades selecionadas. Esta funcionalidade surge com o intuito de proteger e encerrar os espaços em caso de roubo ou semelhante, impossibilitando que qualquer entidade não autorizada mude de zona.

A funcionalidade de ativar/desativar *Lockdown* no que diz respeito ao administrador de sistema é simplesmente sinalizar o sistema ativando ou desativando esta opção. O *anti-passback* é uma ferramenta que permite no momento da avaliação de acesso garantir que

certa entidade necessita de estar no limite de saída identificado pela unidade para conseguir entrar na zona à qual esta dá acesso. Por exemplo, para aceder a uma sala reservada dentro de um edifício, é necessário que a entidade tenha dado entrada neste edifício, caso contrário não poderá aceder à sala. Esta funcionalidade impede que se rompa barreiras para aceder a espaços reservados.

UC23: Registo de visitas

Este caso de uso, assim como o próprio nome indica, trata do registo de visitas esperadas ou não esperadas no sistema, isto é, um registo de uma visita pode ser no momento no qual a visita ocorre, ou então quando a visita é planeada e se sabe previamente quando é que esta se vai realizar.

UC24: Configuração de alarmes

O administrador do sistema pode realizar a configuração de alarmes para os terminais selecionados, sendo que, se o terminal estiver localizado numa zona crítica, o administrador vai querer se notificado da ocorrência de uma tentativa de acesso negada e neste caso, ser alarmado da possível tentativa de intrusão na zona.

UC25 ao UC28: Interação de uma entidade com o sistema

Repetidamente, estes casos de uso foram agrupados devido à semelhança da sua funcionalidade, na qual a interação de uma entidade com o sistema se realiza através de um terminal de marcação. Isto é, uma entidade irá realizar alguma operação no terminal, que normalmente será a tentativa de autenticação e autorização no sistema com o intuito de aceder a uma determinada zona.

De seguida, o sistema irá tentar identificar a entidade no sistema e decidir entre alguns cenários, sendo eles:

- Permitir ou negar o acesso;
- Registar tentativa de acesso;
- Gerar alarmes, como abertura de porta forçada;
- Gerar o bloqueio ou abertura permanente da entidade bloqueadora;

A análise da interação entre uma entidade e o sistema, pode ser complementada com a ajuda do SSD, representado na Figura 16.

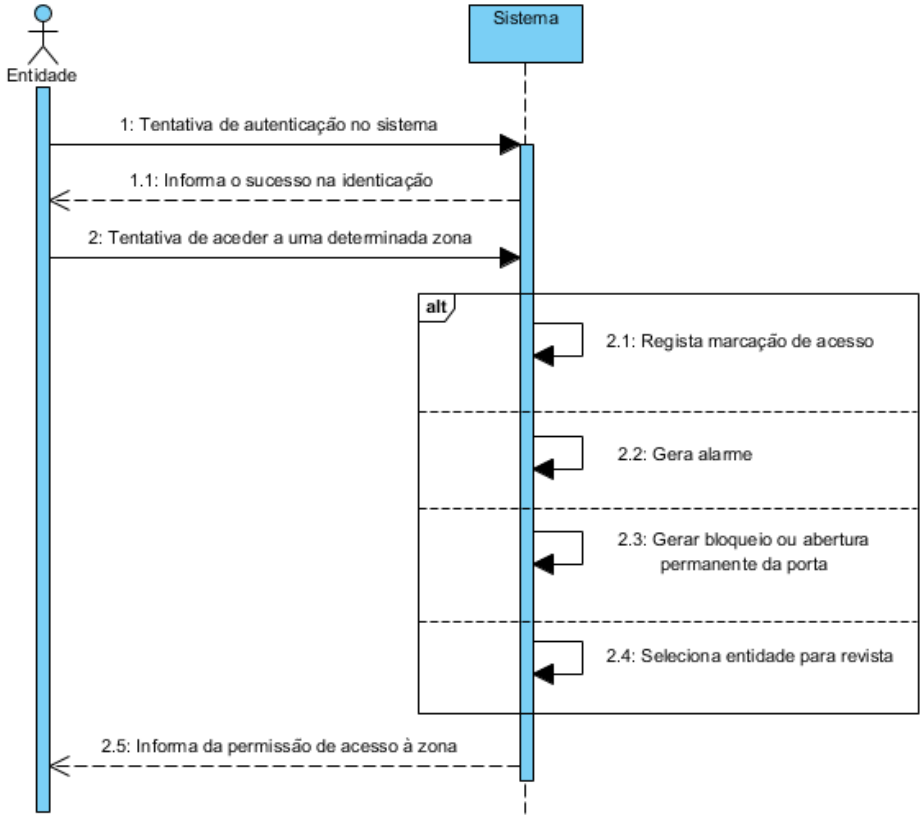

Figura 16 – Diagrama de sequência de sistema da interação de uma entidade com o sistema

4.2.3 Requisitos não funcionais

Um requisito não funcional define o atributo qualidade de um sistema de software (Guru99 - Non-Functional Requirement, 2020). A identificação destes requisitos é uma mais-valia para o produto, pois moldam aquilo que vai ser a qualidade do produto final, porque aquilo que está disponível e visível para o cliente pode ser muito apelativo, mas muito pouco funcional e *performant*, ou seja, pouco eficiente.

A identificação destes requisitos vai ser apresentada com a ajuda do modelo FURPS+, sendo este um acrónimo de: "F" de *Functionality* (Funcionalidade), "U" de *Usability* (Usabilidade), "R" de *Reliability* (Confiabilidade), "P" de *Performance* (Desempenho) e o "S" de *Suportability* (Suportabilidade). Sendo que, neste modelo existe ainda um "+", que representa a especificação de limitação no *design* da solução.

De seguida, são indicados alguns requisitos não funcionais que foram identificados como necessários para a solução a desenvolver, com o âmbito de garantir os padrões e políticas de qualidade ao sistema de software.

1. Autenticação e autorização

O sistema deve garantir a autenticação, impedindo o uso indesejado, através da leitura de diferentes tipos de credenciais, sejam estas de origem biométrica, física ou virtual. O sistema deve ser capaz de, a partir da autenticação de uma entidade no sistema, determinar se este tem autorização ou não para aceder ao recurso em questão, neste caso, a uma determinada zona.

2. Interfaces apelativas, simples e organizadas

O utilizador do sistema deve conseguir configurar e parametrizar o sistema, não sendo demasiado complicado e sem dar a entender que está a criar classes de software, mas sim a criar todo o contexto da aplicação de uma forma lógica.

3. Disponibilidade máxima do sistema

O sistema deverá fornecer uma disponibilidade próxima de 100%, não apresentando falhas e quebras de sistema, evitando causar constrangimentos na sua utilização. Sendo que, uma quebra no sistema, pode significar uma falha grande, consequentemente um problema grave, na produção de qualquer cliente.

4. Rapidez nos processos de interação com o utilizador

Independente do utilizador da aplicação, todos os processos devem ser otimizados, com o intuito de fornecer ao mesmo uma melhor experiência, sem causar qualquer constrangimento ao utilizado da solução, através de atrasos indesejados nos tempos de resposta.

5. *Browser independent*

Deve ser possível utilizar qualquer *browser* para aceder à aplicação, fornecendo assim uma maior cobertura para todos ou quase todos os *browsers*.

6. Base de dados relacional e única

Devido ao modelo conceptual do sistema ser bastante relacional, a escolha e utilização de uma base de dados deste tipo é praticamente obrigatória, conseguindo aproveitar funcionalidades, normalmente fornecidas por motores de base de dados relacional, permitindo uma melhor gestão e manter a integridade e consistência dos dados. O requisito de ser única deve-se ao facto de ser prática na empresa com as aplicações e sistemas existentes, pretendendo-se que este sistema seja abrangente tanto aos clientes existentes, como a novos clientes.

7. Segurança do sistema

Sendo este um sistema que trata de informação sensível das organizações clientes da solução, recomenda-se que todos os terminais estabeleçam comunicação com servidores internos e seguros, que estarão dentro de portas da organização.

4.3 Design arquitetural e de software

Através da elaboração dos subcapítulos anteriores, Modelação de negócio e Engenharia de requisitos, é agora possível continuar com o desenvolvimento do *design* da solução, apresentando diagramas alternativas que podem representar a sua arquitetura.

A arquitetura do software tem como principal objetivo apresentar as características e as expectativas da solução nos níveis operacionais e técnicos, sendo que, devem ser considerados e respeitados vários aspetos e características, como por exemplo, o desempenho, a tolerância a falhas, crescimento, alto nível de confiança e facilidade na manutenção do sistema (Rabelo 2019).

Para continuar, é necessário saber distinguir padrões arquiteturais de padrões de software. Os primeiros tratam de uma vista superior e lógica da arquitetura da solução, são exemplos destes padrões: Arquitetura de microserviços, Arquitetura *serverless* e Arquitetura orientada a eventos. Relativamente aos padrões de software, estes orientam e devem ser considerados no momento da escolha da arquitetura ou arquiteturas a utilizar na solução, são exemplos destes: padrão **SOLID** (***S****ingle Reponsibility Principle*, ***O****pen Closed Principle*, ***L****iskov Substituiton Principle*, ***I****nterface Segregation Principle*, ***D****ependency Inversion Principle*) e o padrão *Factory*.

De seguida, são apresentadas as vistas lógica e de implantação pensadas para a solução a desenvolver, considerando algumas alternativas e a respetiva explicação da arquitetura selecionada.

4.3.1 Vista lógica

A vista lógica será apresentada através de um diagrama de componentes UML, com o objetivo de demonstrar os componentes necessários para o sistema funcionar, bem como, todas as relações entre os mesmos. Nesta fase, será apenas apresentado um diagrama de componentes de nível 2, apresentando apenas os componentes base da solução e as respetivas relações.

De seguida, será apresentado o diagrama de componentes desenvolvido que representa melhor a arquitetura pretendida para o sistema, tendo em consideração alguns aspetos, como a necessidade de ser um sistema num ambiente web, assim como todas as restrições de *design* apresentadas previamente nos requisitos não funcionais, na disciplina de Engenharia de requisitos.

São apresentadas diferentes alternativas, que foram pensadas para a arquitetura do nosso sistema, apresentando qual a solução escolhida considerando todas as vantagens e desvantagens de cada uma, bem como todos os padrões de *design* e software que estas respeitam.

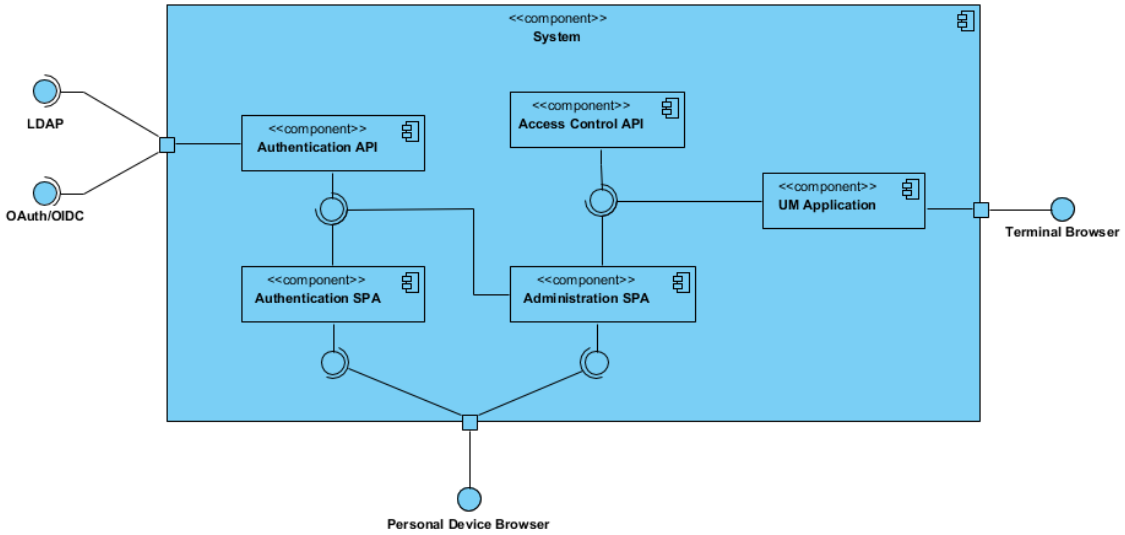

Figura 17 – Diagrama de componentes – Alternativa 1

Na Figura 17, temos representado, através de um diagrama de componentes UML, a primeira alternativa proposta de uma vista lógica da arquitetura da solução.

Nesta solução, assim como em todas as soluções apresentadas, irá existir alguns componentes base, que são os componentes relativos à autenticação (Authentication API e Authentication SPA), bem como a página de administração, onde o administrador (ator do sistema) irá realizar toda a configuração e parametrização do sistema.

Será de seguida apresentada uma nova alternativa, sendo feita a comparação entre ambas e dizendo qual será a mais benéfica para ser a solução, em termos arquiteturais, do nosso sistema.

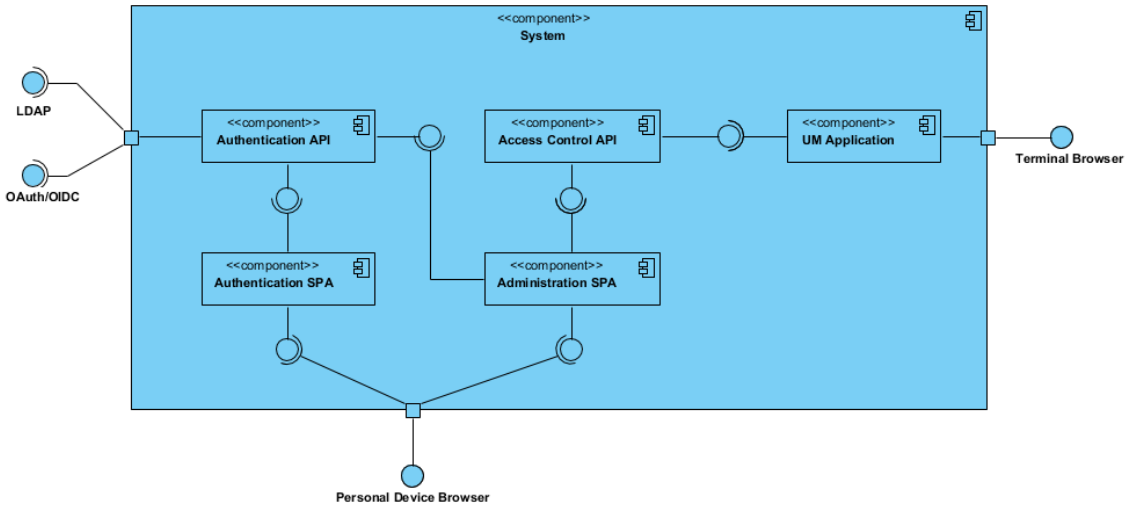

Figura 18 – Diagrama de componentes – Alternativa 2

Comparando as alternativas de arquitetura, apresentadas nas figuras 17 e 18, podemos analisar à primeira vista que ambas as soluções parecem bastante semelhantes, sendo que

possuem uma pequena diferença, que no ponto de vista arquitetural e dos padrões de *design* e software, pode ter bastante impacto.

Falamos aqui do padrão ISP (*Interface Segregation Principle*), que num ambiente de vista lógica de segundo nível, representa a separação das interfaces de comunicação entre as API's (Authentication API e Access Control API) com as respetivas aplicações que consumem os serviços fornecidas por estas API's. Isto pode ter um impacto muito grande em termos de desempenho do sistema e de confiabilidade, permitindo uma maior modularidade do sistema e, consequente estabilidade. Sendo que a suscetibilidade a falhas reduz drasticamente. Sendo que, entre ambas as alternativas apresentadas, a que oferece maior segurança, estabilidade e confiabilidade num nível lógico é a segunda alternativa, sendo esta a selecionada para ser implementada durante o desenvolvimento da prova de conceito do sistema em causa.

De seguida, será apresentada outra alternativa que foi considerada para implementação do nosso sistema, mas acabando por ser descartada nesta fase inicial devido a algumas restrições de *design*, que foram referidas previamente no subcapítulo Requisitos não funcionais.

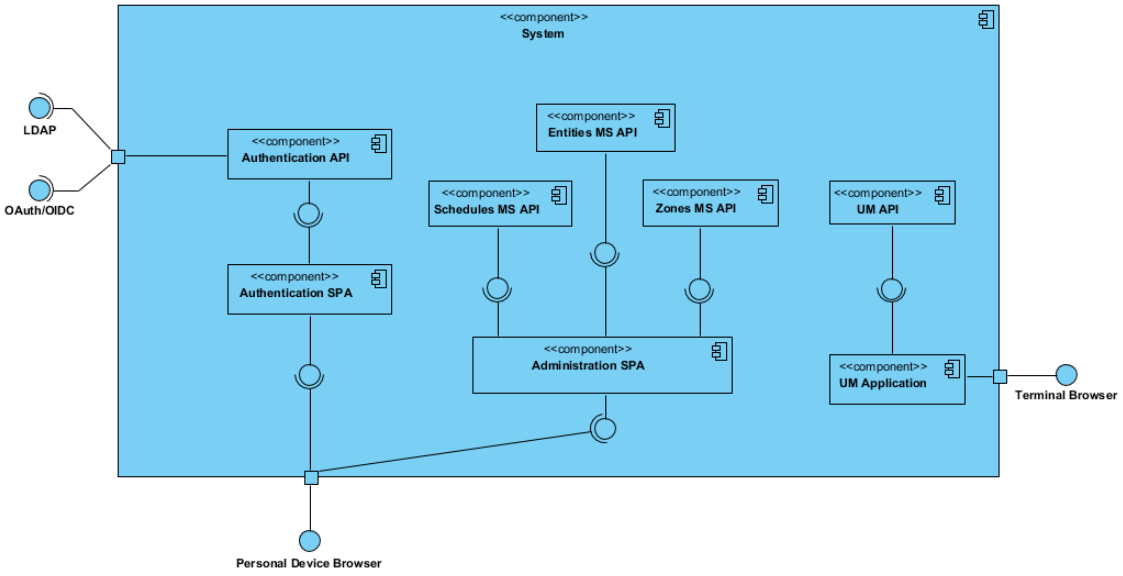

Figura 19 – Diagrama de componentes – Alternativa 3

Na Figura 19 está representada uma terceira alternativa, considerada para vista lógica da solução, onde é possível analisar a representação de uma arquitetura baseada em microserviços, dividindo o que era uma só API nas alternativas anteriores (Access Control API), em quatro API's distintas, cada uma com a sua responsabilidade única respeitando assim um dos padrões SOLID, *Single Responsability Principle*, bem como um dos padrões também respeitados pela alternativa 2, o *Interface Segregation Principle,* através do fornecimento de interfaces distintas, para os diferentes clientes que vão consumir os seus serviços HTTP. Esta alternativa é cada vez mais falada e abordada por diferentes autores, por fornecer uma arquitetura extremamente modular e manutenível, apesar de aumentar consideravelmente a complexidade do *design* arquitetural da solução.

Comparando esta alternativa com a arquitetura monolítica apresentada na segunda alternativa (Figura 18), esta arquitetura de microserviços, apesar do acrescente de complexidade, fornece uma maior modularidade e relativa facilidade em manter e suportar o sistema, mas esta modularidade assenta no princípio de que cada microserviço possui uma responsabilidade única, a sua base de dados dedicada e é responsável apenas pela gestão desta. Remetendo para um dos requisitos não funcionais, da existência e manutenção de uma base de dados relacional devido ao grande número de relações existentes entre os conceitos do sistema.

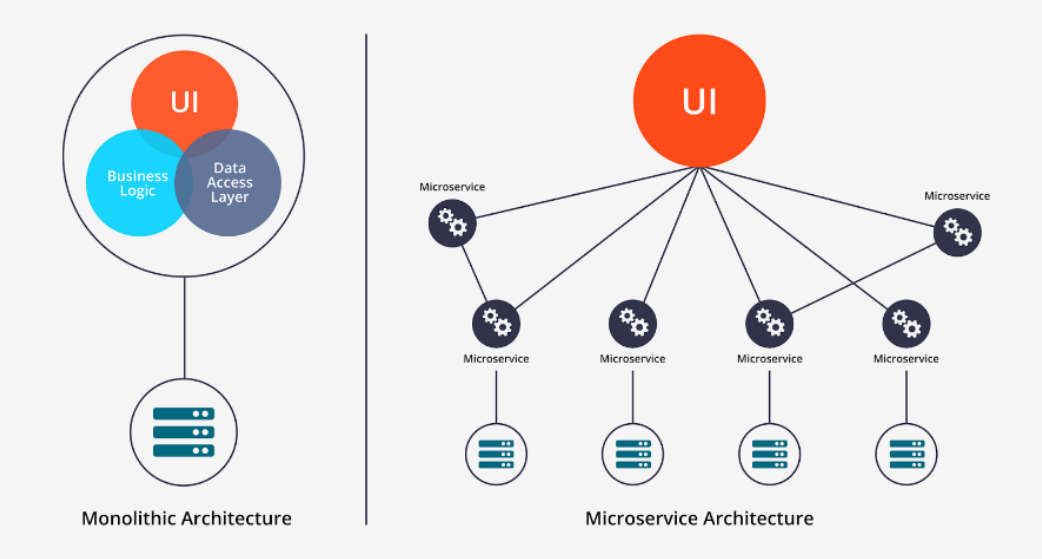

Figura 20 – Comparação entre arquitetura monolítica e arquitetura de microserviços (Rabelo 2019)

Posto isto, o *design* arquitetural em termos de vista lógica, será representado pela alternativa 2, visível na Figura 18. Sendo esta a arquitetura a ser apresentada com prova de conceito no desenvolvimento e implementação do sistema, e que melhor responde aos requisitos não funcionais do sistema, mais propriamente ao requisito Base de dados relacional e única, visto que a arquitetura de microserviços pressupõe a existência de múltiplas instâncias de bases de dados para a sua correta utilização, que como já foi referido não pode ser utilizado neste projeto.

4.3.2 Vista de implantação

A vista de implantação será apresentada através do diagrama de implantação UML. Este diagrama representa uma vista estrutural, responsável por estabelecer a ligação entre os componentes/artefactos do sistema e os recursos de infraestrutura, necessários para a implementação da solução nos clientes.

O diagrama de implantação consiste em três principais elementos, os artefactos, os "nós" e a relação entre estes. Os artefactos representam os componentes, os executáveis necessários

para o bom funcionamento do sistema, os “nós” são utilizados neste diagrama de UML para representar um recurso da infraestrutura, ou seja, um equipamento de *hardware* onde os artefactos são alojados, estes “nós” podem representar alguns elementos como, uma máquina, um sistema operativo, entre outros. Por fim, temos o estabelecimento de ligação entre os “nós”, representando as comunicações que deverão existir entre os diferentes elementos da infraestrutura para garantir que esta seja implantada devidamente.

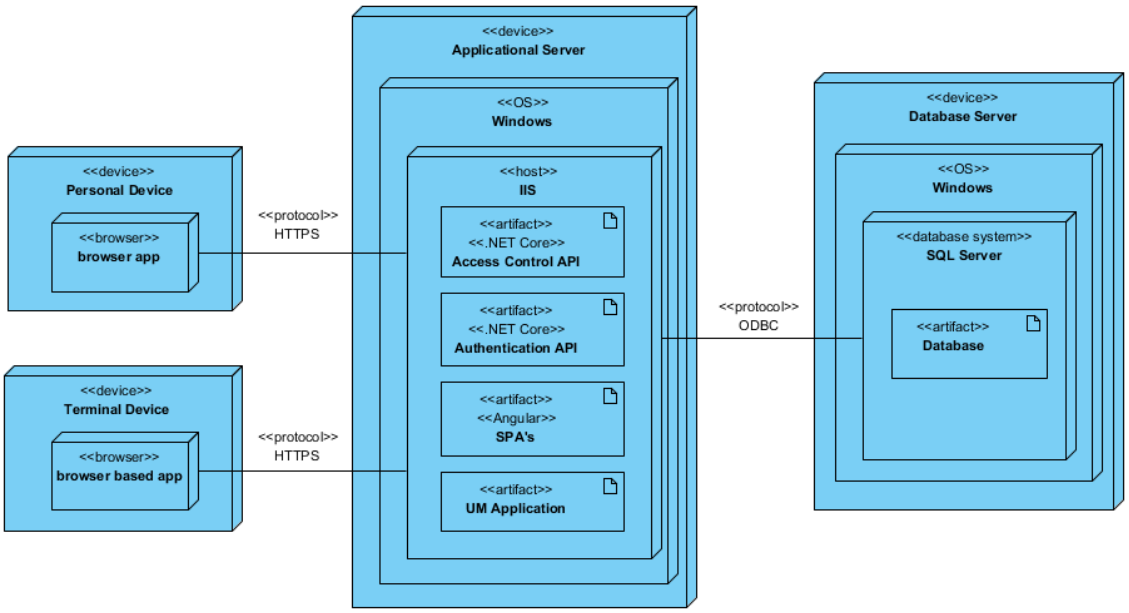

Figura 21 – Diagrama de implantação – Alternativa 1

Na Figura 21 temos apresentado a primeira alternativa para a implantação do sistema, onde estão representados 4 “nós” principais: dispositivo pessoal (*Personal Device*), terminal de marcação (*Terminal Device*), servidor aplicacional (*Application Server*) e servidor de base de dados (*Database Server*).

Nesta implantação, como podemos concluir através da análise do diagrama apresentado, todos os artefactos necessários para o funcionamento do sistema estão implantados no mesmo servidor aplicacional, fazendo com que este esteja encarregue de toda a gestão de recursos funcionais do sistema, podendo levar a um *overload* dos pedidos, através do protocolo HTTP, provenientes dos diferentes “nós” onde é consumido os serviços fornecidos pelas diferentes API’s, bem como os serviços e fornecimento das interfaces a partir das aplicações cliente, também implantados no servidor aplicacional, por parte dos diferentes SPA’s.

De seguida, será apresentada um novo diagrama representante de uma alternativa de implantação da solução, juntamente com a comparação entre a nova alternativa e a primeira alternativa, apresentando qual será a escolha para implantação do sistema acompanhado da sua justificação.

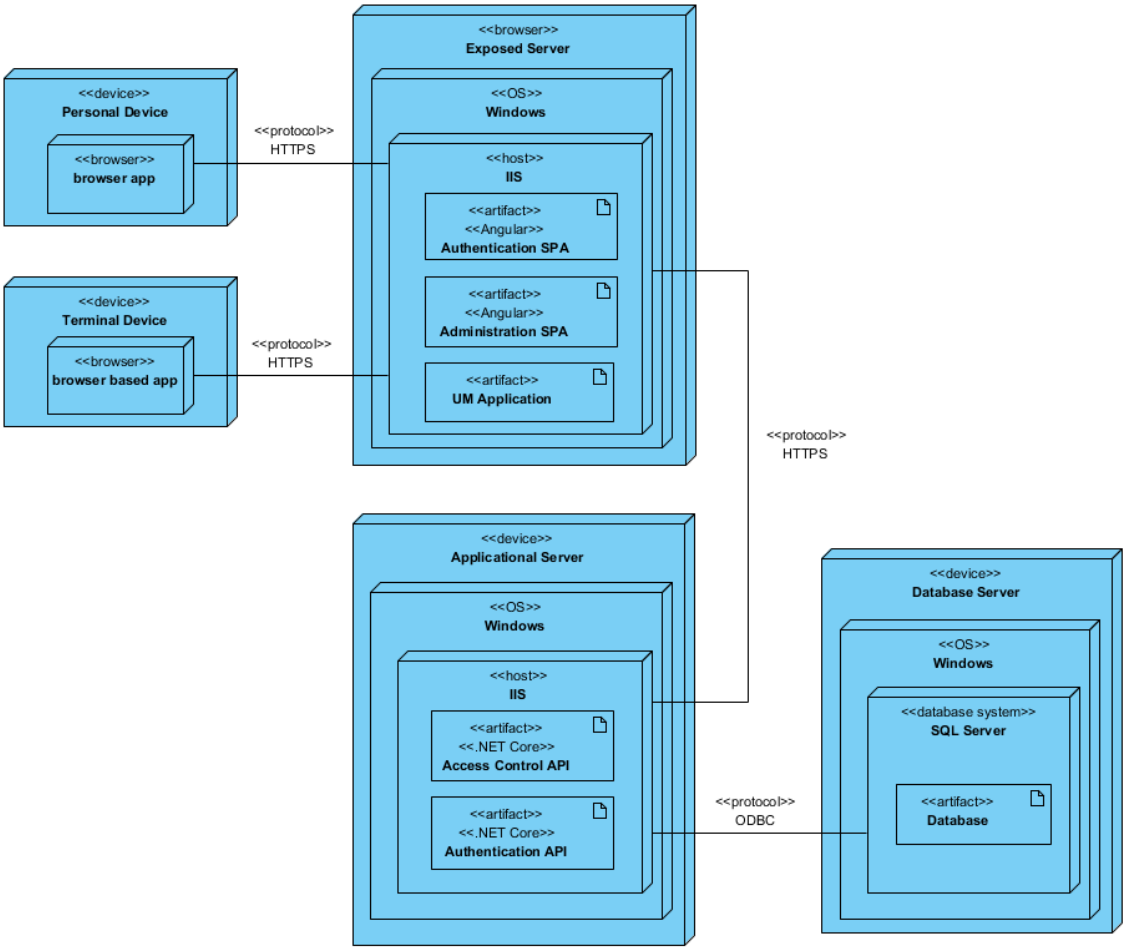

Figura 22 – Diagrama de implantação – Alternativa 2

Na Figura 22 temos apresentado uma nova alternativa de implantação do sistema onde, à primeira vista, podemos verificar a existência de um novo "nó", designado servidor exposto (*Exposed Server*).

A introdução deste novo "nó", em comparação com a primeira alternativa, simboliza a separação de conceitos, ou seja, os artefactos que estabelecem a ligação direta com os "nós" dos dispositivos, fornecendo as *interfaces* para a apresentação das aplicações nos diferentes dispositivos, são separados dos artefactos que tratam da gestão dos dados do sistema, estabelecendo a ligação com um servidor de base de dados, assim como na primeira alternativa. Assim, existe uma separação entre os componentes que estabelecem e fornecem serviços para os dispositivos externos, e os componentes que tratam de todos os processos funcionais e não funcionais da solução, como autenticação, autorização e validação de acesso.

Assim, a alternativa mais válida para ser implementada no nosso sistema é a segunda, devido à maior modularidade e confiabilidade que fornece ao nosso sistema, assim como à redução da carga adicional de possuir os artefactos "cliente" e "servidor" na mesma máquina. De salientar ainda o fator segurança que foi referido como um requisito não funcional do sistema

(Segurança do sistema), ficando todos os serviços que tratam de lógica e gestão de dados da aplicação num servidor interno da empresa.

5 Implementação

No capítulo de implementação da solução são descritas as decisões tomadas no desenvolvimento da solução, fazendo sempre que possível uma análise comparativa com a solução existente em termos de tecnológicos, bem como em termos conceptuais.

É, de seguida, apresentado a partir de imagens acompanhadas de uma breve descrição o protótipo final do portal de administração desenvolvido. Mostrando as interfaces que o administrador de sistema irá interagir para configurar e parametrizar todo o sistema.

Sabendo que o foco deste sistema é controlar o acesso aos espaços físicos garantindo que estes sejam intransponíveis, são ainda apresentados os passos e as condições que estão por detrás do algoritmo de validação do acesso de uma entidade.

5.1 Tecnologias e estrutura

As tecnologias escolhidas para o desenvolvimento deste projeto baseiam-se em dois principais fatores, o ambiente de desenvolvimento e o *stack* tecnológico da empresa e dos seus colaboradores.

Posto isto, a solução e seus componentes encontra-se desenvolvida recorrendo a duas *frameworks open source*. A *framework* .NET Core, desenvolvida e mantida pela Microsoft, para os componentes envolvidos no *back-end* e a *framework* Angular, desenvolvida e mantida pela Google, para o desenvolvimento do cliente web (*front-end*).

5.1.1 Back-end

Recorrendo à *framework* .NET Core e utilizando a ferramenta/ambiente de desenvolvimento Microsoft Visual Studio[11], foi desenvolvido a lógica de negócio de todo o sistema, o chamado Back-end.

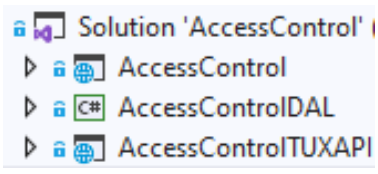

Figura 23 – Componentes e projetos relativos ao back-end

[11] Microsoft Visual Studio – https://visualstudio.microsoft.com/

Na figura 23 está apresentado os três componentes criados para o desenvolvimento da solução de controlo de acessos, no que diz respeito à camada lógica de negócio da aplicação, o *back-end*. Sendo eles:

1. **AccessControl** – Projeto web que contém toda a lógica de parametrização e configuração do sistema de controlo de acessos. Este fornece uma interface HTTP, que será consumida pela aplicação web de gestão do sistema de controlo de acessos.
2. **AccessControlTUXAPI** – Projeto web com função particular de comunicação com os terminais de marcação, tratando das interações das entidades com o sistema, desde a validação da tentativa de acesso ao registo da marcação. Este projeto fornece uma interface HTTP que será consumida pelos terminais.
3. **AccessControlDAL** – Projeto partilhado pelas duas aplicações, responsável pelo acesso à informação, tratando de todas as interações do sistema com a base de dados.

Esta solução encontra-se desenvolvida com base numa arquitetura monolítica dividida em 3 camadas com funções e responsabilidade únicas: *User Interface*, *Business Logic* e *Data Access Logic*.

Utilizando esta arquitetura os utilizadores realizam os pedidos e comunicam com o sistema através das *interfaces* disponibilizadas pela camada de *User Interface*, que por sua vez comunica com a camada de *Business Logic,* capaz e destinada ao tratamento e validação dos dados fornecidos pelo utilizador. Esta, muitas das vezes, pode necessitar de aceder a informação da base de dados para as suas operações e para isso terá de comunicar com a camada designada para tal, a *Data Access Logic*.

A camada de *User interface* nunca deve comunicar diretamente com a camada de informação e persistência. Para tal deve passar pela camada que trata da lógica de negócio e tratamento de dados, mantendo assim um fluxo único e as responsabilidades nas diferentes camadas bem definidas.

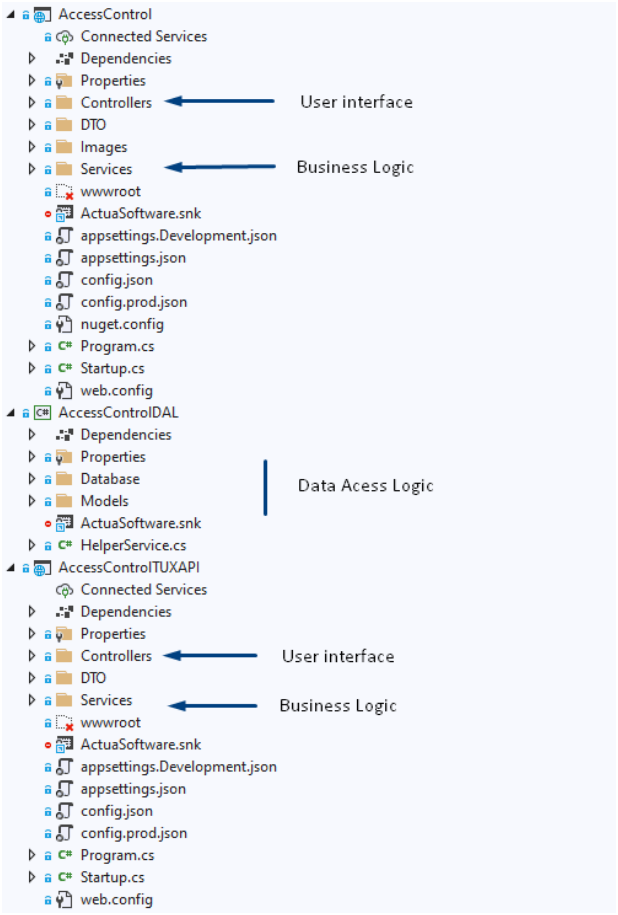

Figura 24 – Estrutura detalhada da solução

Na figura 24 está representada a estrutura detalhada da solução desenvolvida apresentando os diferentes componentes e suas característica e camadas. Ambos os projetos web, AccessControl e AccessControlTUXAPI, contêm as camadas *User Interface* e *Business Logic* oferecendo uma interface HTTP a ser consumida pelos seus destinatários, administradores de sistema e terminais respetivamente, bem como todo o tratamento de dados adequado à sua função e responsabilidade. O projeto AccessControlDAL trata da camada de persistência e controlo de acesso à informação, a *Data Access Layer*.

Em termos comparativos com a solução de controlo de acessos existente na empresa, no que diz respeito ao *back-end*, a principal diferença está na comunicação dos terminais com o painel de controlo onde esta fica mais facilitada, isto porque a comunicação passa a ser realizada através do protocolo HTTP, ao invés de ser TCP/IP. Esta mudança facilita todo o processo de atualização e manutenção do software, fazendo com que as alterações realizadas à lógica de negócio se possam refletir rapidamente e em poucos passos no ambiente de produção dos clientes.

5.1.2 Front-end

No que diz respeito ao front-end, camada visual e de interação com o utilizador de uma aplicação, este é desenvolvido recorrendo à *framework* Angular, versão 9. Para tal, foi criado um projeto único que vai conter todas as páginas e ações necessárias para a gestão do sistema de controlo de acessos.

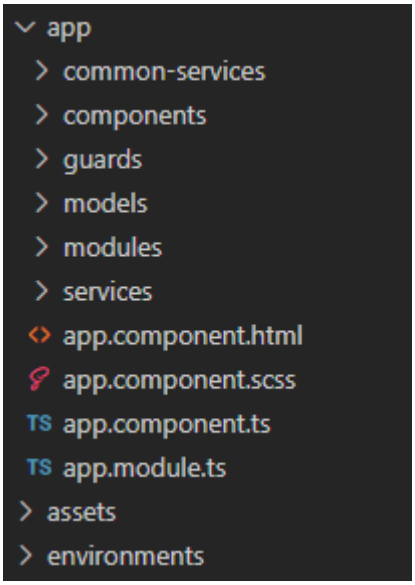

Figura 25 – Arquitetura projeto front-end

Na figura 25 é possível ver a estrutura do projeto desenvolvido, onde toda a parte aplicacional e programática se encontra na diretoria "app" do projeto. Dentro desta diretoria, encontram-se subpastas que caracterizam partes distintas da aplicação:

- Common-services: diretoria onde se encontram os ficheiros cuja função é partilhar os seus métodos e funcionalidades para serem utilizados por toda a aplicação. Por exemplo, função cujo objetivo é converter uma hora especificada para o número total de segundos;
- Components: diretoria onde se encontram todos os componentes da aplicação. Um componente retrata uma página/interface da aplicação, sendo composto por um ficheiro Typescript, contendo as funções desencadeadas pela interação do utilizador com a interface, um ficheiro HTML e CSS para o desenvolvimento das interfaces;
- Guards: diretoria contendo os ficheiros "seguranças" da aplicação, isto é, quando se tenta aceder a uma página da aplicação sem ter realizado autenticação ou com a sessão expirada, a aplicação redireciona para a página de autenticação;
- Models: diretoria contendo todos as classes de domínio/modelos da aplicação;
- Modules: diretoria que contém todos os módulos da aplicação. A *framework* Angular é bastante modular, um módulo contém parte da aplicação e esta divisão é feita pois todos os componentes podem não necessitar de ser carregados inicialmente no

primeiro ecrã, fazendo com que esta divisão tenha bastante impacto na velocidade de carregamento das páginas;

- Services: diretoria contendo os ficheiros que tratam da ligação da aplicação/cliente web com as interfaces e serviços HTTP disponibilizados pela API, desenvolvida para tratar toda a lógica de negócio e persistência dos dados.

Esta aplicação fornece todos os meios necessários para que o administrador de sistema interaja para configurar e parametrizar todo o sistema de controlo de acessos.

Em comparação com a solução existente, no que diz respeito ao *front-end*, as diferenças notórias do sistema podem ser sentidas em 2 fatores:

1. Compatibilidade, porque falamos de uma página web que pode ser acedida a partir de qualquer parte do mundo, desde que tenha acesso ao servidor onde o sistema está instalado, bem como poder-se utilizar qualquer navegador de Internet à preferência do administrador para interagir com a aplicação. Enquanto na solução atual o sistema tem que ser parametrizado numa aplicação exclusiva para Windows, com interfaces menos apelativos;
2. Usabilidade, devido principalmente à simplicidade e facilidade nos processos de parametrização e configuração do sistema, bem como às *interfaces* modernas e mais apelativas para o utilizador.

5.1.3 Base de dados

A escolha e implementação da solução, relativamente à escolha do sistema de base de dados, foi bastante orientada à carteira de clientes existentes da empresa, uma vez que esta solução de controlo de acessos tem que ser suportada em clientes existentes.

Posto isto, a solução foi implementada e testada em dois grandes sistemas de base de dados, Microsoft SQL Server e Oracle. Apesar disso, a camada de acesso à base de dados (*Data Access Layer*) está preparada para aceder a outros sistemas de base de dados, desde que esta seja relacional e suporte a linguagem estrutural designada *Structured Query Language* (SQL).

5.2 Portal de administração

O portal de administração é uma aplicação/página web onde o administrador de sistema irá aceder para parametrizar e configurar todo o sistema de controlo de acessos numa primeira instância, assim como sempre que necessitar de realizar alguma alteração à configuração do sistema.

O protótipo final do portal de administração onde se parametriza todo o sistema de controlo de acessos é apresentado no Anexo B, de modo a garantir uma melhor apresentação das imagens e leitura para o respetivo leitor.

No Anexo B são apresentadas as *interfaces* desenvolvidas com base no *Look and Feel* apelativo utilizado noutras aplicações da empresa, desde a página de autenticação às páginas de parametrização. Para reforçar o entendimento dos conceitos envolvidos do sistema, a apresentação destas *interfaces* é também acompanhada por uma breve descrição e explicação da utilização prática destes.

5.3 Avaliação de acesso

Mais uma vez, o principal desafio deste projeto passa pela correta avaliação das entidades que têm acesso a determinados espaços físicos, designados neste sistema como zonas, permitindo ou negando o acesso baseado em toda a configuração e parametrização realizada pelos administradores de sistema.

Antes de se explicar o algoritmo de avaliação de acesso, é necessário salientar que a empresa produz e trabalha com equipamentos e sistemas inteligentes, ou seja, os equipamentos não necessitam de comunicar com o painel de controlo/servidor para determinar e avaliar o acesso aos espaços, possuindo memória e capacidade de processamento para o fazer. Posto isto foi desenvolvido, com base no modelo e sistema de controlo de acessos atual, um algoritmo de envio de dados para os terminais que é invocado periodicamente. No entanto, o serviço de sincronização dos dados para os terminais está fora do âmbito deste projeto, pois este já existia previamente sendo apenas desenvolvido um algoritmo que, através da configuração do sistema, avalie e envie os dados das entidades e o seu horário de acesso ao espaço em questão para os terminais.

A avaliação do acesso é baseada em prioridades, ou seja, existem certas parametrizações que se vão sobrepor à definição e configuração base do sistema de controlo de acessos. Para se perceber melhor o que isto quer dizer, explica-se de seguida as diferentes etapas de avaliação de acesso, apresentando também as razões do acesso não autorizado, sendo elas:

1. Verificação se a entidade se encontra ativa

A inativação de uma entidade pode acontecer por várias razões, rescisão do contrato, suspensão de trabalho, entre outras. Para isso, é necessário verificar se esta se encontra ativa no momento da tentativa de acesso.

2. Verificar modo confinamento (*Lockdown*)

O modo confinamento surge como uma funcionalidade nova em relação à solução de controlo de acessos atual. O intuito desta funcionalidade é proteger os espaços, por exemplo em caso de roubo, encerrando todos os acessos às pessoas não autorizadas pelo administrador de sistema. Assim, caso este modo se encontre ativo e a pessoa não tenha sido dada como autorizada a desbloquear as portas pelo administrador, o acesso será negado.

3. Verificar *anti-passback*

O *anti-passback* é uma funcionalidade que existe assentando na necessidade de garantir que não se rompam barreiras, consiste na avaliação se a entidade em questão está a entrar para

uma zona saindo da zona limite definida pela unidade, ou seja, quando se instala uma unidade define-se quais são as zonas limite desta, indicando a zona para a qual esta dá entrada e a zona de saída.

Estando o *anti-passback* ativo e caso a entidade não se encontre, no momento da avaliação, na zona limite de saída identificada pela unidade o acesso será negado.

4. Verificar existência de exceções

As exceções, tal como o próprio nome indica, são utilizadas excecionalmente quando necessárias e têm uma prioridade alta no momento da avaliação. Estas podem ser de três tipos:

- Exceção ao horário: indicação do horário (24 horas) a utilizar entre duas datas;
- Exceção ao período: indicação do tipo de acesso (permitido, negado ou condicionado) entre um período;
- Exceção ao perfil: associação temporário de uma entidade a um perfil.

Caso uma destas exceções se verifique, o acesso será determinado com base no horário ou tipo de acesso definido pela exceção em causa e a avaliação de acesso será terminada.

5. Verificar capacidade máxima da zona

A definição da capacidade máxima de uma zona surge também como uma novidade em relação à solução existente. A avaliação será feita com base nas entidades que se encontram na zona à qual se tenta aceder, sendo negado o acesso caso a capacidade máxima da zona tenha sido atingida.

6. Validar se a entidade faz parte de um perfil com a unidade associada

Como já foi referido, a ligação entre quais entidades têm acesso a determinadas zonas/unidade é feita através dos perfis, atribuindo-se perfis com base nas permissões que se pretende dar às entidades.

Posto isto, é necessário determinar se a entidade está afeta a um perfil com acesso a esta zona/unidade e qual é este perfil. Caso seja determinado que a entidade em causa não se encontra em nenhum perfil que garanta acesso a esta unidade, o acesso é negado e a avaliação terminada.

7. Verificar qual o horário afeto à zona com base no perfil

Após se garantir que a entidade se encontra num perfil, pode-se determinar qual o horário que está afeto a esta zona. Isto é, no momento de definição da zona é possível indicar qual o horário a ser usado por defeito na mesma, sendo que este horário pode ser sobreposto no momento da definição dos perfis e associação das zonas que fazem parte deste.

Posto isto, após determinar qual o horário afeto à zona com base no perfil associado à entidade, é possível determinar qual o tipo de acesso dando como terminado o processo de avaliação.

6 Experimentação e avaliação

No capítulo de experimentação e avaliação é inicialmente apresentada a hipótese de investigação que foi definida para avaliação deste projeto e determinará o sucesso ou insucesso do desenvolvimento do sistema.

Para isso, foram identificadas algumas métricas que indicam valores relevantes para a avaliação do desempenho dos diferentes componentes desenvolvidos neste projeto. Considerando as métricas definidas e através da análise dos resultados obtidos nas metodologias de avaliação utilizadas, é possível avaliar e concluir se o sistema apresenta ou não resultados positivos quanto à hipótese de investigação, permitindo determinar o sucesso deste projeto.

6.1 Hipótese de investigação

Nesta secção é apresentada a hipótese de investigação definida para esta dissertação, com o intuito de determinar e concluir se o desenvolvimento deste projeto representa uma mais-valia considerando os resultados obtidos.

Para isso, é necessário determinar se o produto desenvolvido respeita as políticas de qualidade da empresa e qual o valor acrescentado que este representará tanto no âmbito da empresa como no âmbito do cliente.

De forma a concluir quais as mais-valias no investimento do desenvolvimento desta solução de controlo de acessos, foram definidos os seguintes objetivos que permitirão determinar o sucesso do desempenho do sistema:

- Garantir tempos de resposta mínimos nas operações relacionadas com o portal de administração;
- Garantir tempos de resposta mínimos no momento de avaliação do acesso;
- Fornecer excelente disponibilidade dos serviços, respondendo a momentos de maior afluência ao sistema;
- Ótima gestão de conflitos, evitando erros na configuração do sistema.

Deste modo, a hipótese de investigação que determinará em última instância o sucesso no desenvolvimento deste projeto é o valor acrescentado que este produto representará tanto na ótica da empresa, como dos clientes. Esta hipótese será avaliada atendendo os objetivos definidos e considerando os resultados obtidos nas métricas apresentadas no seguimento deste capítulo.

6.2 Métricas

Neste subcapítulo serão apresentadas as métricas utilizadas no desenvolvimento do projeto que vão permitir avaliar e determinar o sucesso da hipótese de investigação.

Estas métricas vão basear-se naquilo que se pretende avaliar e testar, neste caso pretende-se analisar o desempenho e o grau de satisfação do sistema em dois principais momentos de interação críticos:

- Interação do administrador de sistema com o portal de administração;
- Interação das entidades com o terminal no momento da tentativa de acesso;

De seguida, são apresentadas as métricas consideradas para a avaliação do sistema, sendo que as primeiras seis métricas são utilizadas na avaliação do desempenho da página web (portal de administração), e as restantes utilizadas para a avaliação das duas API's desenvolvidas, a primeira para a parametrização e configuração do sistema e a segunda para a comunicação com os terminais no momento de avaliação do acesso.

1. *First Contentful Paint* (FCP)

Esta métrica corresponde ao momento em que o utilizador navega para uma página web, representando o tempo que o *browser* demora a apresentar o primeiro conteúdo, seja este um pedaço de texto, imagem ou qualquer elemento HTML visível na página (Walton – FCP 2021).

2. *Time to Interactive* (TTI)

TTI calcula quanto tempo a página demora desde o início do carregamento até ficar totalmente operacional e utilizável (Walton – TTI 2020). Isto acontece quando:

- A página carregou todo o conteúdo considerado útil;
- O evento de iniciação dos elementos da página é finalizado;
- A página responde a interações do utilizador em menos de 50 milissegundos.

3. Índice de velocidade

Esta métrica permite perceber o quão rápido o conteúdo apresentado na página passa a ser percetível no momento de carregamento da mesma (Web.Dev – Speed Index 2021).

4. *Total blocking time* (TBT)

O TTI é calculado através da soma de todas as diferenças temporais entre as métricas FCP e tempo para interação, quando o tempo de execução de uma tarefa é superior a 50 milissegundos, designada como tarefa longa. O tempo máximo de 50 milissegundos para uma tarefa ser finalizada foi definido com base que a partir desse tempo o utilizador poderá notar o atraso no carregamento da página (Walton – TBT 2020).

5. *Largest Contentful Paint* (LCP)

O LCP é uma métrica que reporta o tempo que o maior componente, em termos de tamanho (bytes), demora a renderizar e a aparecer visível e utilizável na página. Esta métrica é muito relevante, porque muitas vezes os resultados de outras métricas como a FCP não é suficiente, devido a considerem todos os elementos como por exemplo um simples texto (Walton – LCP 2020).

6. *Cumulative Layout Shift* (CLS)

Esta métrica calcula a quantidade de movimento dos componentes apresentados na página desde o momento inicial ao final de carregamento. O movimento dos elementos na página pode muitas das vezes ser inesperado e causar constrangimentos ao utilizador, uma vez que um elemento pode aparecer inicialmente visível numa posição e no milissegundo a seguir ser deslocado 1 pixel, que pode ser suficiente para causar uma ação indesejada por parte do utilizador (Walton et al. 2021).

7. Número de falsos positivos

Esta métrica indica o número de resultados que não correspondem ao expectável. Este valor é quantitativo e permite perceber até que ponto é que o sistema pode ser falível.

8. Tempo mínimo e máximo de resposta

Esta métrica, tal como o próprio nome indica, refere-se aos tempos mínimo e máximo de resposta do servidor aos pedidos efetuados. Estes valores podem variar consoante o caso de teste aplicado, pretendendo-se que sejam sempre o mais baixo possível.

9. Tempo médio de resposta

O tempo médio de resposta permite uma melhor avaliação em relação à responsividade do sistema. Assim como na métrica anterior, este pode variar pretendendo-se que seja sempre o mais baixo possível.

10. Percentagem de pedidos falhados

Esta métrica é calculada através da relação do número de pedidos falhados com o número total de pedidos efetuados. Um pedido falhado no contexto do projeto é qualquer código de resposta a um pedido HTTP fora da família de códigos de sucesso (2XX).

11. Desvio padrão dos tempos de resposta

O desvio padrão nos tempos de resposta permite perceber o desfasamento no tempo de espera dos utilizadores por um pedido. Pretende-se que o desvio seja o mínimo possível, sendo que um valor muito alto no desfasamento pode causar constrangimentos entre os utilizadores do sistema.

6.3 Metodologias de avaliação e análise de resultados

No presente subcapítulo são apresentadas as várias metodologias utilizadas para testar o nosso sistema, assim como as experiências planeadas e elaboradas atendendo às métricas

definidas. É dividido em subsecções que representam os diferentes componentes web em análise, de maneira a ser mais percetível para o leitor quais as experiências realizadas sobre cada um.

A quantidade de metodologias e tipos de testes que se podem realizar numa aplicação de cariz web é muita vasta, sendo que nesta secção são apresentadas aquelas experiências que se consideram mais relevantes para as métricas definidas, e consequente avaliação da hipótese de investigação.

Assim, recorrendo a diferentes ferramentas e plataformas que permitiram testar o sistema de controlo de acessos desenvolvido, apresenta-se de seguida as experiências e testes realizados aos diferentes componentes em avaliação, bem como os resultados obtidos e a análise a estes resultados. Sendo importante salientar que todas as experiências são realizadas num ambiente interno, controlado e sobre as mesmas condições.

6.3.1 Página web (Portal de administração)

Começando por apresentar os resultados obtidos nos testes ao portal de administração utilizando o Lighthouse[12]. O Lighthouse é uma ferramenta *open-source* integrada no DevTools (ferramentas do programador) do *browser* Chrome, que permite a avaliação da qualidade de uma página web, submetendo-a a vários testes e calculando métricas relevantes para este projeto.

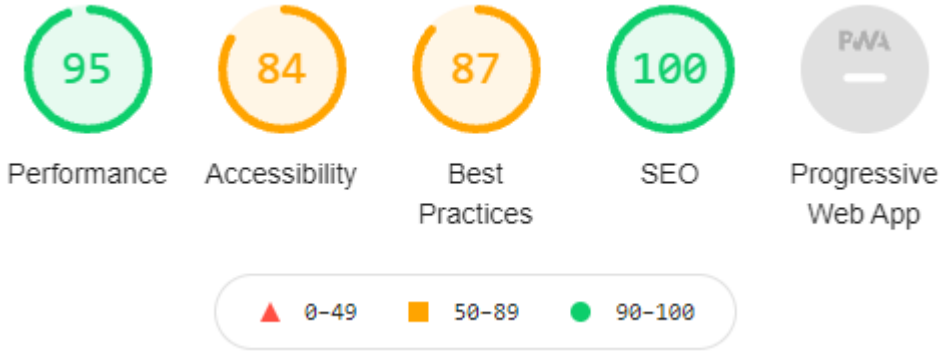

Figura 26 – Resultados gerais da qualidade da página

Na Figura 26 estão apresentados os resultados gerais obtidos e atribuídos pelo Lighthouse à qualidade das páginas desenvolvidas para o portal de administração. Estes resultados mostram-se bastante positivos, sendo o indicador de desempenho, traduzido de *performance*, o mais relevante apresentando um valor bastante perto do máximo possível. Para a atribuição deste resultado foram consideradas várias métricas relevantes, que foram abordadas e explicadas na secção 6.2.

[12] Lighthouse – https://developers.google.com/web/tools/lighthouse

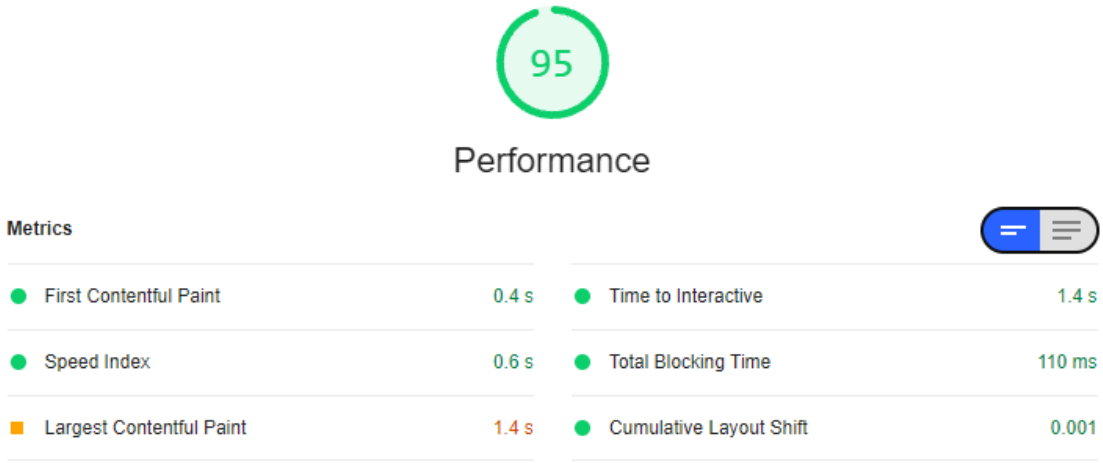

Figura 27 – Métricas de avaliação de desempenho

Como se pode ver através da análise da Figura 27, os resultados obtidos nas métricas utilizadas para o cálculo do resultado final de *performance* foram todos muito positivos, dentro da escala de avaliação do Lighthouse, com a exceção da métrica LCP. Esta métrica, como já foi explicada previamente, reporta o tempo que o maior elemento da página, considerando o tamanho em bytes, demora a renderizar na página.

Acredita-se que todas as aplicações são passíveis de melhoria, sendo que esta não é exceção. O Lighthouse fornece um relatório que disponibiliza e identifica as oportunidades de melhoria nos diferentes parâmetros avaliados.

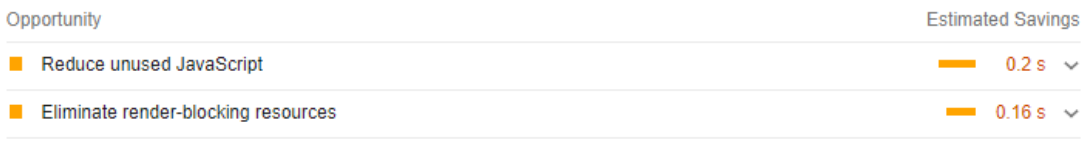

Figura 28 – Oportunidades de melhoria do desempenho da página

Em termos de desempenho, foram identificados pela ferramenta dois principais pontos sujeitos a melhoria:

1. Redução de código JavaScript não utilizado, que prevê a melhoria de 0.2 segundos;
2. Eliminação de recursos que estão a impedir o carregamento da página, que prevê a melhoria de 0.16 segundos.

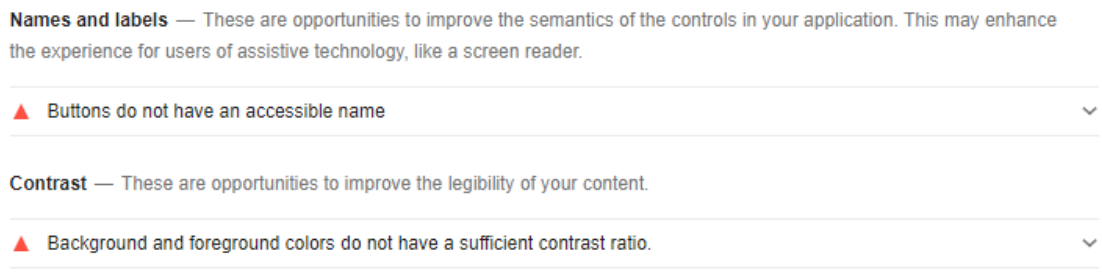

Figura 29 – Oportunidades de melhoria da acessibilidade da página

No que diz respeito à acessibilidade da página, traduzido de *accessibility*, as oportunidades identificadas foram:

1. Inclusão do atributo *name* nos elementos da página, principalmente nos botões. Isto pode melhorar a experiência de utilizadores com deficiências visuais, sendo este atributo utilizado em algumas tecnologias de assistência auditiva;
2. Analisar relação de contraste das cores utilizadas nas páginas.

Relativamente ao parâmetro *best practices*, que representa as práticas que devem ser consideradas e utilizadas quando a página está em fase de produção em tópicos como segurança e memória. No entanto, o resultado obtido neste parâmetro não corresponde à realidade, pois a falha detetada foi a falta de utilização de um certificado válido nas comunicações HTTP com o servidor, que não é um fator crítico nesta avaliação uma vez que os testes são realizados em ambiente interno e controlado.

Assim, considerando os resultados obtidos nos testes realizados sobre a página web do portal de administração é possível afirmar que apesar da existência de possíveis melhorias, os resultados nas métricas definidas confirmaram a fluidez, usabilidade e funcionalidade que se sente na utilização da página.

6.3.2 API para configuração do sistema

O desenvolvimento desta API é acompanhado por testes constantes utilizando a ferramenta Postman, que permitiram validar a implementação da solução, no que diz respeito aos diferentes pontos de entrada da API e algoritmos desenvolvidos para gestão dos diferentes conceitos envolvidos no sistema, testando vários cenários de configuração possíveis.

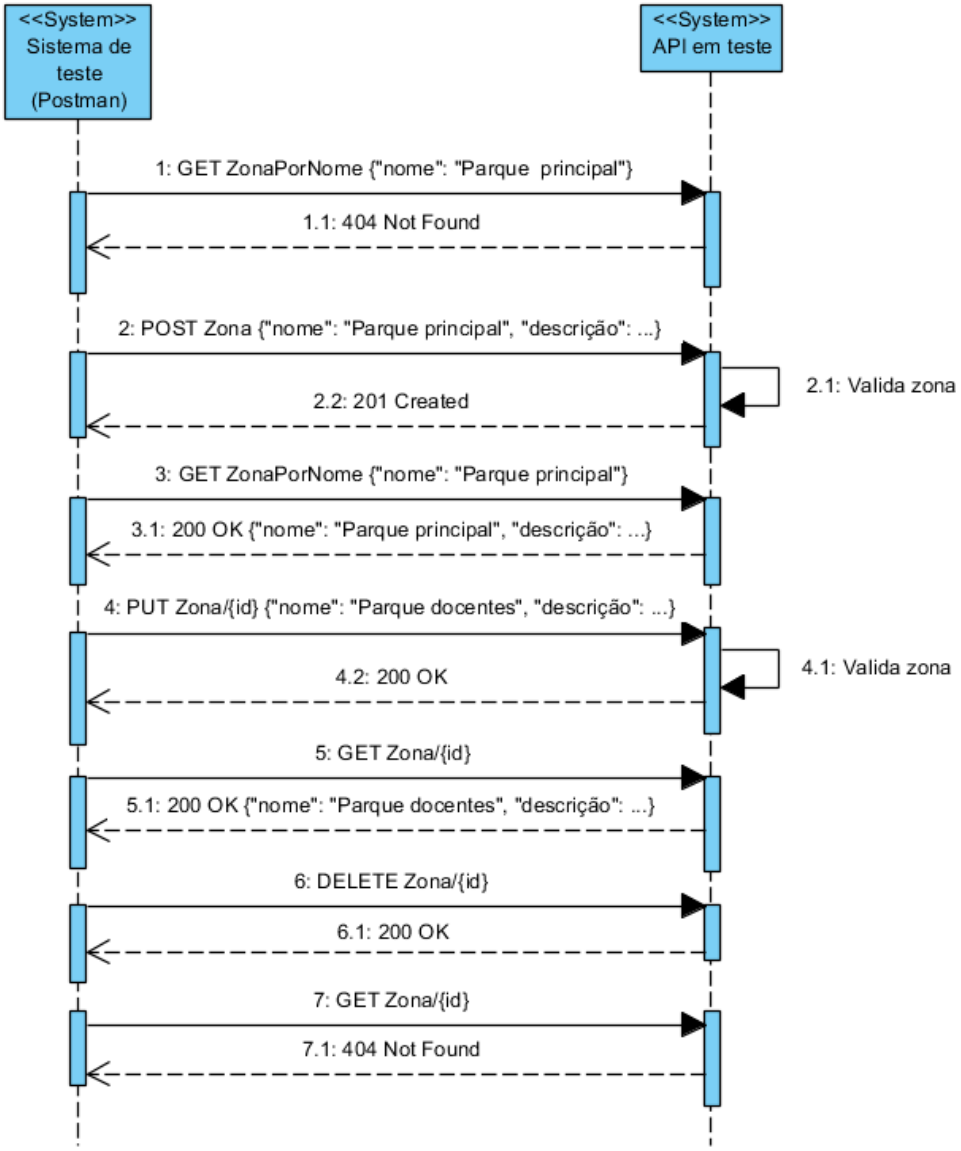

Figura 30 – Padrão de teste realizado à API para configuração do sistema

Na Figura 30 está apresentado um diagrama de sequência que permite representar um exemplo de cenário de teste executado sobre a API para configuração do sistema, neste caso o cenário de gestão de zonas. Uma vez que esta API trata principalmente da gestão dos conceitos envolvidos no sistema é importante testar aquilo que neste âmbito se designa por CRUD:

- (C) *Create* – Criação de novos elementos no sistema;
- (R) *Read* – Leitura de objetos do sistema;
- (U) *Update* – Atualizar elementos no sistema;
- (D) *Delete* – Eliminar elementos do sistema.

De seguida, é apresentada uma breve explicação do fluxo destes testes aplicado ao caso prática do conceito zona:

1. Verificação se a zona existe no sistema;
2. Teste à criação e respetiva verificação do algoritmo de validação de zona;
3. Testar a leitura de uma zona específica e ao mesmo tempo garantir que a zona foi criada com sucesso;
4. Testar a edição de uma zona alterando o nome;
5. Verificar se a alteração foi sucedida através da leitura da zona alterada;
6. Teste à eliminação de uma zona do sistema;
7. Através de um pedido de leitura da zona que se eliminou, verificar que a mesma não se encontra no sistema.

Este tipo de testes permite também a validação da integração entre componentes, nomeadamente a integração entre a API desenvolvida e a base de dados, simulando um ambiente real de utilização do sistema, sendo que ao longo da execução destes testes é possível consultar diretamente na base de dados a informação dos dados utilizados nos testes, recorrendo a aplicações como o SQL Server Management Studio[13].

De modo a facilitar a execução deste tipo de testes, pretende-se automatizar a sua execução permitindo testar múltiplos cenários diferentes com relativa facilidade. Para isto, foi utilizada novamente a ferramenta Postman, que permite a definição de um conjunto de pedidos executados sequencialmente à API de modo a simular o fluxo de teste apresentado na Figura 30.

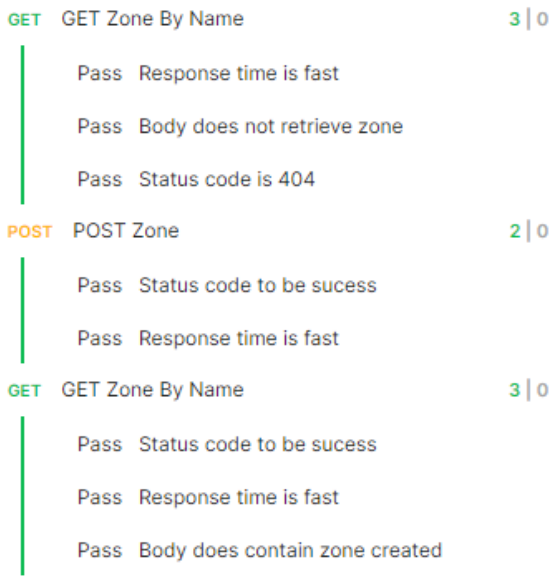

Figura 31 – Resultados dos testes ao CRUD de zonas (parcial)

[13] SQL Server Management Studio – https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15

Na Figura 31 são apresentados os resultados obtidos, de forma parcial (resultados completos no Anexo C), no teste que permite validar a leitura, criação, edição e eliminação de uma zona no sistema.

A execução automatizada deste tipo de testes e consequente análise dos resultados apresenta uma mais-valia no desenvolvimento de software, uma vez que permite muitas vezes detetar erros inesperados, corrigindo de imediato as falhas detetadas. Assim, esta prática oferece à solução desenvolvida uma maior solidez na implementação, indo de encontra com as políticas de qualidade e boas práticas de produção de software praticadas na empresa.

6.3.3 API para avaliação de acesso

Esta API é desenvolvida com a responsabilidade única de comunicação com os terminais, e consequente interação com as entidades do sistema no momento da tentativa de acesso a uma determinada zona. Este componente acaba por ser o que mais impacto tem na decisão final de aceitação da hipótese, uma vez que se a avaliação de acesso de uma entidade no momento da tentativa de acesso a um determinado espaço físico não for correta, todo o sistema pode ser posto em causa.

Com o intuito de testar o sistema na avaliação da tentativa de acesso é realizado uma carga de testes funcionais, ajustando a parametrização para testar e cobrir o maior número de cenários possíveis.

Para a realização destes é utilizada a plataforma e ferramenta Postman[14] onde é possível simular o pedido HTTP que é efetuado pelo terminal à nossa API no momento da tentativa de acesso (Figura 32).

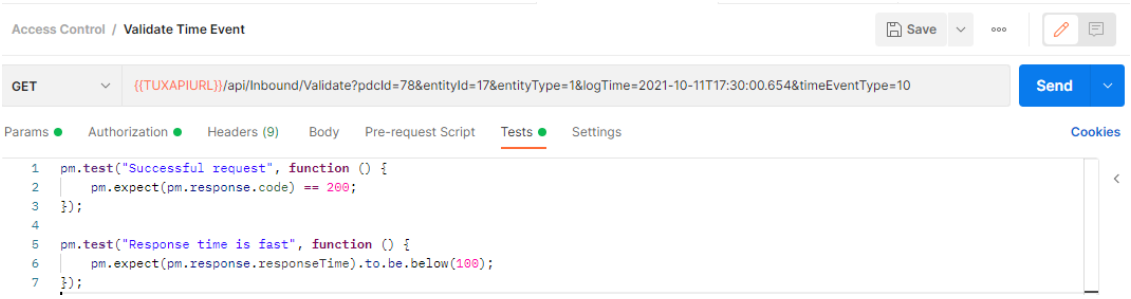

Figura 32 – Simulação da tentativa de acesso via Postman

Esta ferramenta permite também executar ao mesmo tempo um conjunto de testes em relação ao pedido efetuado, através da escrita de testes no tabulador disponibilizado para o efeito, como se pode ver na Figura 32. Neste caso, os testes escritos pretendem validar se o pedido responde com código de sucesso e se o tempo de resposta do pedido é inferior de 100 milissegundos.

[14] Postman – https://www.postman.com/

De seguida, são apresentados alguns dos cenários de testes funcionais mais relevantes realizados sobre as condições limite do sistema, ou seja, casos de teste em que se pressupõe que a entidade reúne inicialmente todas as condições para ter acesso ao espaço físico, mas com a adição de novas condicionantes.

Tabela 4 – Testes funcionais a cenários limite do sistema

CENÁRIO	**RESULTADO PRETENDIDO**	**RESULTADO OBTIDO**
Entidade é inativada	Acesso negado	Acesso negado
Sistema entra em modo de confinamento e entidade não tem permissão para abertura de portas	Acesso negado	Acesso negado
Sistema entra em modo de confinamento e entidade tem permissão para abertura de portas	Acesso permitido	Acesso permitido
Sistema com *anti-passback* ativo e entidade não se encontra na zona limite do terminal	Acesso negado	Acesso negado
Sistema com *anti-passback* ativo e entidade encontra-se na zona limite do terminal	Acesso permitido	Acesso permitido
Entidade associada a uma exceção com horário sem acesso	Acesso negado	Acesso negado
Entidade retirada do perfil que a associava com terminal	Acesso negado	Acesso negado
Horário associado ao perfil da entidade alterado para sem acesso	Acesso negado	Acesso negado
Zona de entrada encontra-se com a capacidade máxima permitida	Acesso negado	Acesso negado

Através da análise dos testes funcionais realizados visíveis na Tabela 4, é possível concluir que os resultados obtidos em comparação ao que era expectável foram muito positivos, sendo que todos os resultados obtidos corresponderam ao que era pretendido. Isto é um fator bastante positivo naquilo que é a segurança do sistema, uma vez que o número de falsos positivos nestes testes funcionais foi 0.

Foram também realizados testes de desempenho com o intuito de perceber e avaliar métricas como os tempos mínimo, máximo e médio de resposta, no momento da avaliação de acesso. Para isso, é utilizada a ferramenta *open-source* Apache JMeter[15] que permite a definição de múltiplos casos de uso de teste, variando as condicionantes sobre as quais os pedidos são efetuados ao servidor.

[15] Apache JMeter – https://jmeter.apache.org/

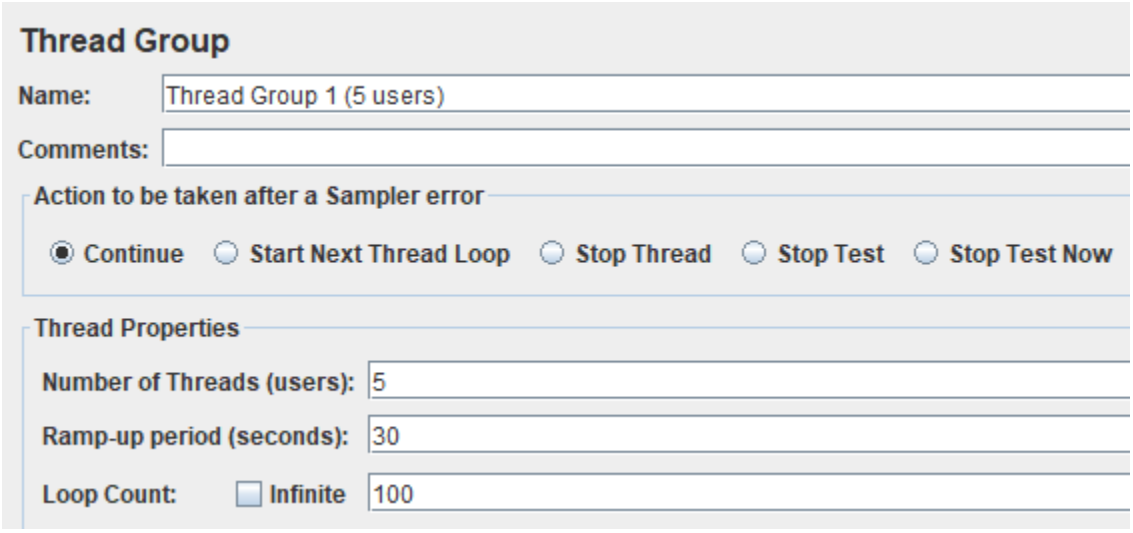

Figura 33 – Exemplo de teste de desempenho utilizando JMeter

Na Figura 33 está representado o primeiro caso de teste de desempenho executado ao momento de avaliação do acesso, sendo que a primeira condicionante definida foi:

- *Number of threads* – 5 utilizadores. Este número simboliza, em termos reais, o número de entidades que estão a fazer uma tentativa de acesso em simultâneo. Um caso prático seria, por exemplo, a tentativa de entrada de 5 alunos em diferentes salas ao mesmo tempo;
- *Ramp-up period* – 30 segundos. Este valor representa o tempo máximo de espera no qual se pretende que o servidor responda a todas as *threads*, sendo que após este período são executadas todas as *threads* que ainda não foram lançadas;
- *Loop count* – 100 vezes. Este valor significa o número de vezes que serão lançadas as *threads* (número de pedidos) definidas, uma vez que a anterior terminar. Permite avaliar um número maior de casos e ter uma melhor perceção de um cenário onde existem vários pedidos a serem realizados de seguida. Um caso prático deste cenário é, por exemplo, uma empresa que tenha múltiplos torniquetes à entrada e num período de maior afluência são feitos múltiplos pedidos atrás de múltiplos pedidos.

# Samples	Average	Min	Max	Std. Dev.	Error %
500	29	22	53	4.16	0.00%
500	29	22	53	4.16	0.00%

Figura 34 – Resultados obtidos no primeiro teste de desempenho

Na Figura 34 são apresentados os resultados obtidos no primeiro teste realizado em formato de tabela, retirado da própria ferramenta JMeter, que é mais percetível para o leitor. Nestes resultados podemos analisar e avaliar várias métricas, nomeadamente:

- Número de amostra – 500 pedidos. Considerando que são efetuados 5 pedidos em simultâneo 100 vezes seguidas;

- Tempo médio de resposta – 29 milissegundos;
- Tempo mínimo de resposta – 22 milissegundos;
- Tempo máximo de resposta – 53 milissegundos;
- Desvio padrão – 4.16 milissegundos;
- Percentagem de pedidos falhados – 0.00%;

Neste primeiro cenário de teste são descritos todos os valores visíveis na tabela, de modo a contextualizar o leitor do significado das métricas e das unidades de medida, uma vez que a tabela está em inglês e os tempos não têm identificação da unidade utilizada.

Os tempos mínimo, máximo e médio obtidos neste teste mostraram-se bastante positivos, sendo todos muito baixos, o que na ótica do utilizador representa um tempo de espera pela resposta e avaliação de acesso quase impercetível. O desvio padrão de 4.16 milissegundos representa o tempo de espera que um utilizador poderá esperar, em média, mais do que outro utilizador na tentativa de acesso, sendo mais uma vez impercetível para o utilizador comum. Relativamente à percentagem de pedidos falhados, uma vez que os testes são realizados num ambiente controlado, era expectável que o valor obtido fosse de 0.

# Samples	Average	Min	Max	Std. Dev.	Error %
2000	28	20	89	3.94	0.00%
2000	28	20	89	3.94	0.00%

Figura 35 – Resultados obtidos no segundo teste de desempenho

Aumentando o valor da variável *number of* threads, que representa o número de utilizadores a realizar a tentativa de acesso em simultâneo, para 20 foram obtidos os resultados que se podem analisar na Figura 35. Se compararmos os resultados com o primeiro cenário de teste, à primeira vista podemos ver valores muito semelhantes, o que é um indicador excelente no que diz respeito à estabilidade do sistema. Sendo que, os tempos mínimo, máximo e médio não apresentaram diferenças significativas, apesar do tempo máximo apresentar uma subida de 36 milissegundos. Esta diferença não será percetível pelo utilizador que naquela tentativa de acesso esperou mais 36 milissegundos que das outras vezes.

Utilizando a ferramenta JMeter também é possível efetuar testes de carga sobre esta API, modificando as variáveis de modo a executar múltiplos pedidos em simultâneo ao servidor, simulando casos de afluência extrema em empresas que possuem instalações com bastantes pontos de controlo, que vai permitir avaliar grandezas como a confiabilidade e disponibilidade do sistema. Assim, foi aumentada, mais uma vez, a variável que simula o número de utilizadores a efetuar tentativa de entrada em simultâneo para 100.

# Samples	Average	Min	Max	Std. Dev.	Error %
10000	23	15	59	4.45	0.00%
10000	23	15	59	4.45	0.00%

Figura 36 – Resultados obtidos no teste de carga

Como se pode verificar na Figura 36, os resultados obtidos foram surpreendentes. Apesar do aumento significativo do número de utilizadores a requisitar o sistema em simultâneo, os valores obtidos não desviaram muito dos valores obtidos nos testes de desempenho onde a carga era muito menor. Numa amostra de 10000 pedidos, o tempo médio de resposta foi inferior aos tempos obtidos para amostras de menos dimensão, o que reflete no valor incrível obtido no desvio padrão de 4.45 milissegundos. Este valor representa que a diferença sentida quando o servidor está sobre imensa pressão será insignificante e impercetível para o utilizador.

Os resultados obtidos nos testes de desempenho e carga realizados sobre a API de interação com os utilizadores no momento da avaliação de acesso mostraram-se bastante positivos. Pretende-se saber se os valores obtidos nas métricas utilizadas apresentam uma melhoria significativa em relação à solução existente de acessos.

Infelizmente, não é possível obter de forma fácil uma amostra quantitativa grande dos tempos de resposta do momento de avaliação de acesso utilizando esta solução, uma vez que a comunicação efetuada entre os terminais e o servidor não é através do protocolo HTTP, sendo feita através de *endpoints* disponibilizados a partir de serviços *Windows Communication Foundation* (WCF).

```
14:47:00.834 : [Actuasys Expert Base App.registerTimeEvents] Central validation succeeded and took 305 ms
```

Figura 37 – Tempo de resposta da avaliação de acesso na solução existente de acessos

Na Figura 37 é apresentado um excerto de texto retirado dos *logs* gerados pelos terminais quando detetam qualquer tipo de ação ou evento, neste caso o evento de tentativa de acesso. Esta linha de *log* apresenta o tempo (305 milissegundos) que o servidor demorou a responder ao pedido na avaliação de acesso.

De modo a conseguir uma amostra minimamente aceitável, para comparar com os resultados obtidos nos testes ao sistema de controlo de acessos desenvolvido, foram realizadas 10 tentativas de acesso nas mesmas condições dos testes de desempenho executados anteriormente. Os resultados obtidos com base nas métricas definidas foram:

- Tempo médio de resposta – 330 milissegundos;
- Tempo mínimo de resposta – 305 milissegundos;
- Tempo máximo de resposta – 378 milissegundos.

Considerando o primeiro teste de desempenho efetuado, onde o tempo médio de espera foi de 29 milissegundos, podemos verificar que a diferença, quando comparado com o valor obtido neste último teste, é de 301 milissegundos. Esta diferença apesar de se tratar de frações de segundos pode-se mostrar significativa em alguns cenários, sendo a afluência o mais significativo, ou seja, quando o número de tentativas de acesso seguidas no mesmo terminal é maior. Um caso prático desta situação é, por exemplo, quando para aceder às instalações é necessário passar por uma barreira de torniquetes, sendo que com base nestes

resultados afluência maior pode resultar em filas maiores na solução existente do que na solução desenvolvida.

Apesar que a comparação com a solução existente de acessos possa ser insuficiente para se tirar conclusões com um grau de certeza elevado, devido à dificuldade no processo de obtenção de uma amostra maior de resultados, estes apresentaram diferenças significativas quando comparados com os resultados obtidos nos testes à solução desenvolvida. Assim, considerando todos os resultados obtidos nos testes executados à API para validação de acesso, tendo sido bastante positivos, é possível concluir, considerando a hipótese de investigação em análise, que o desenvolvimento desta solução pode representar uma mais-valia significativa para a empresa.

7 Conclusão

Neste capítulo são apresentadas as conclusões retiradas do desenvolvimento desta dissertação. Começando por fazer uma síntese do problema endereçado, os objetivos que se pretendiam alcançar e o que foi efetivamente possível de cumprir com o desenvolvimento deste projeto.

Para finalizar, são apresentadas as principais limitações encontradas ao longo do desenvolvimento deste projeto e o trabalho futuro a desempenhar.

7.1 Síntese

O controlo de acessos é ao mesmo tempo o tema e problema deste projeto, sendo um assunto que as empresas têm que pensar e considerar quando pretendem gerir e limitar o acesso às suas instalações.

Para auxiliar as empresas neste processo de gestão de acesso, surgiu a necessidade de desenvolver um sistema de controlo de acessos capaz de satisfazer estas necessidades, garantindo que entidades sem autorização não consigam aceder a zonas que não deviam à partida ter acesso, garantindo que estes espaços sejam intransponíveis a acessos indesejados.

Assim, com base nesta necessidade, foi desenvolvido um sistema de controlo de acessos totalmente em ambiente web com o intuito de colmatar as lacunas existentes na solução de controlo de acessos atual. Estas lacunas foram identificadas através do *feedback* dos clientes que utilizam esta solução ao longo dos anos, concluindo-se que esta é difícil de gerir e limitada nas suas funções.

A definição deste sistema de acessos foi repensada revendo os conceitos envolvidos de modo a garantir uma solução estável, com vista a otimizar todos os processos de interação com os utilizadores, desde a configuração do sistema por parte do administrador ao momento que as entidades interagem com os terminais na tentativa de aceder às zonas controladas.

Com base nos objetivos definidos inicialmente, foi realizada uma análise do contexto e estado da arte, que permitiram obter melhor entendimento dos conceitos envolvidos, das tecnologias e arquiteturas relevantes e mais utilizadas, de modo a fornecer a base para a decisão da escolha da arquitetura do sistema e das tecnologias a utilizar para o desenvolvimento do projeto. De forma resumida, a solução foi desenvolvida com base numa arquitetura monolítica, utilizando a *framework* .NET Core para o desenvolvimento dos componentes de *back-end* e a *framework* Angular para o desenvolvimento do cliente web, onde o administrador de sistema irá configurar o sistema.

Tendo este projeto nascido da necessidade de melhorar e modernizar a solução de controlo de acessos atual da empresa, o investimento e desenvolvimento deste projeto parte do

pressuposto que este constituirá uma mais-valia tanto para a empresa como para os seus clientes, sendo esta a hipótese de investigação definida e que irá determinar o sucesso do desenvolvimento deste projeto. Com o intuito de responder e avaliar esta hipótese, foram identificadas as métricas a ter em conta no momento da avaliação e de seguida realizados um conjunto de testes aos diferentes componentes do sistema.

As análises efetuadas aos resultados obtidos nos diferentes tipos de testes efetuados permitiram concluir as mais-valias do desenvolvimento desta solução de controlo de acessos, uma vez que os resultados destes testes foram predominantemente positivos. A comparação com a solução atual permitiu concluir que a adoção de tecnologias mais recentes no desenvolvimento desta aplicação foram um fator relevante que proporcionou com que esta se apresente mais eficiente, apelativa e funcional. Concluindo-se assim, que esta aplicação poderá representar uma mais-valia como substituição da solução atual tanto para a empresa, em termos de negócio, como para os clientes existentes e potenciais, em termos de usabilidade, eficiência e confiabilidade.

7.2 Limitações e trabalho futuro

No decorrer do desenvolvimento do projeto foram encontradas algumas limitações que acabaram por impedir a execução de alguns testes. Uma das limitações está relacionada com os testes de comparação de desempenho no momento da avaliação de acesso, uma vez que não era possível de forma fácil e direta obter uma amostra grande dos tempos de resposta com a solução existente, fazendo com que a análise destas métricas não fosse tão relevante na comparação com a solução desenvolvida como poderia ser.

Relativamente ao trabalho futuro a desenvolver para consolidar e aumentar o valor da solução passará por dois principais pontos:

1. Realização de uma carga de testes por parte da equipa de *Quality Assurance* (QA) em ambientes mais próximos daquilo que será a realidade dos clientes, de maneira a determinar e detetar falhas no sistema de controlo de acessos desenvolvido. Sendo que, de uma forma geral, os elementos que participam no desenvolvimento do projeto não são capazes de prever alguns cenários reais e por essa mesma razão, a importância do sistema ser testado por uma equipa externa ao desenvolvimento é muito significativa.
2. Desenvolvimento de uma API que fará a integração com a aplicação de portarias existente na empresa. A aplicação de portarias é um módulo adicional da solução de acessos que não fez parte do âmbito deste projeto. Esta aplicação permite a monitorização em tempo real do sistema de controlo de acessos, controlando as entradas e saídas das instalações, capacidades das zonas, entre outras funcionalidades.

Referências

(Actuasys 2020)	Actuasys, acedido a 9 de fevereiro de 2021, https:// actuasys.com/pt/
(Almeida 2017)	Symmetric and Assymetric Encryption, Hackernoon, Rafael Almeida, acedido em 2 de fevereiro de 2021, https://hackernoon.com/symmetric-and-asymmetric-encryption-5122f9ec65b1
(Bradford 2019)	5 common encryption algorithms, Contel Bradford, Storage Craft, acedido a 2 de fevereiro de 2021, https://blog.storagecraft.com/5-common-encryption-algorithms/
(Broeckelmann 2017)	Authentication vs Federation vs SSO, Medium, Robert Broeckelmann, acedido em 2 de outubro de 2021, https://medium.com/@robert.broeckelmann/authentication-vs-federation-vs-sso-9586b06b1380
(Buchanan B. 1999)	Buchanan B. (1999) Data Encryption Principles. In: Handbook of Data Communications and Networks. Springer, Boston, MA
(Common Lounge 2018)	Blowfish: The first well-known encryption algorithm in public domain, Common Lounge, acedido em 3 de fevereiro de 2021, https://www.commonlounge.com/discussion/d95616beecc148daaa23f35178691c35
(D. Rane 2016)	Superiority of Twofish over Blowfish, Deepali D. Rane, Assistant Professor Department of Information Technology, D.Y. Patil College of Engineering, novembro de 2016
(Data Science Academy 2016)	Bases de dados relacionais, Data Science Academy, acedido em 7 de março de 2021
(Dias 2017)	The 5 Factors of Authentication, Renan Dias, acedido em 31 de janeiro 2021, https://medium.com/@renansdias/the-5-factors-of-authentication-bcb79d354c13
(E. Barnhart 2019)	Trends & Technology dictate 'Smart' Future for Access Control, Security Magazine, Jeffrey E. Barnhart, acedido em 26 de janeiro de 2021, https://www.securitymagazine.com/articles/91343-trends-technology-dictate-smart-future-for-access-control
(Elasticsearch 2020)	Elasticsearch, acedido em 2 de fevereiro de 2021, https://www.elastic.co/
(Facebook Open Source 2020)	React Native, Facebook Open Source, acedido em 4 de fevereiro de 2021, https://facebook.github.io/react-native/
(Fair 2018)	The Evolution of Access Control, Kintronics, Virginia Fair, acedido em 26 de janeiro de 2021, https://www.isonas.com/news-education/the-evolution-of-access-control/
(GDPR.eu 2018)	GDPR.eu, Programme of the Europe Union, acedido em 2 de fevereiro de 2021, https://gdpr.eu/tag/gdpr/
(Google - Chromium OS 2020)	Cromium Operative System, Google, acedido em 16 de fevereiro de 2021, https://www.chromium.org/chromium-os
(Google 2020)	Angular, Google, acedido em 4 de fevereiro de 2021, https://angular.io/
(Guru99 - Functional Requirement 2020)	Functional Requirement, Guru99, acedido em 22 de fevereiro de 2021, https://www.guru99.com/functional-requirement-specification-example.html
(Guru99 - Non-Functional Requirement 2020)	Non-Functional Requirement, Guru99, acedido em 22 de fevereiro de 2021, https://www.guru99.com/non-functional-requirement-type-example.html
(Houlis 2018)	The history and future of access control, IFSEC Global, Peter Houlis, acedido a 26 de janeiro de 2021,

https://www.ifsecglobal.com/global/history-future-access-control-credentials/

(HS Tech Group 2017) Access Control Devices, HS Tech Group, Stuart Forchheimer, acedido em 26 de janeiro de 2021, https://hstechgroup.com/blog/access-control-101-what-you-need-to-know-about-security/

(Klekovic 2017) EU GDPR Blog, EU GDPR vs. European data protection directive, Ivan Klekovic, acedido em 2 de fevereiro de 2021, https://advisera.com/eugdpracademy/blog/2017/10/30/eu-gdpr-vs-european-data-protection-directive/

(Microsoft - .NET Core 2020) .NET Core, Microsoft, acedido em 4 de fevereiro de 2021, https://github.com/dotnet/core

(Microsoft - Typescript 2020) TypeScript, Microsoft, acedido em 4 de fevereiro de 2021, https://www.typescriptlang.org/

(Ohio State University 2019) Cybersecurity, Ohio State University, acedido em 2 de fevereiro de 2021, https://cybersecurity.osu.edu/cybersecurity-you/passwords-authentication/multifactor-authentication

(Okta 2021) What Is Federated Identity?, Okta, acedido em 2 de outubro de 2021, https://www.okta.com/identity-101/what-is-federated-identity/

(One Login 2020) What is single-sign on?, OneLogin, acedido em 02 de outubro de 2021, https://www.onelogin.com/learn/how-single-sign-on-works

(P. Christensson 2018) Authentication definition, Tech Terms, P. Christensson, acedido em 29 de janeiro de 2021, https://techterms.com/definition/authentication

(Parlamento Europeu e do Conselho 1995) Parlamento Europeu e do Conselho, EUR-lex, acedido em 2 de fevereiro de 2021, https://eur-lex.europa.eu/legal-content/PT/TXT/?uri=celex%3A31995L0046

(Peter Koen 2014) Front End Innovation, Peter Koen, acedido em 11 de fevereiro de 2021, http://frontendinnovation.com/fei

(Peyrott 2015) What is and how does single sign-on authentication work?, Auth0, Sebastian Peyrott, acedido em 2 de outubro de 2021, https://auth0.com/blog/what-is-and-how-does-single-sign-on-work/

(Walton – FCP 2021) First Contentful Paint, Web.Dev, Philip Walton, acedido em 12 de outubro de 2021, https://web.dev/fcp/

(Walton – LCP 2020) Largest Contentful Paint, Web.Dev, Philip Walton, acedido em 12 de outubro de 2021, https://web.dev/lcp/

(Walton – TBT 2020) Total Blocking Time, Web.Dev, Philip Walton, acedido em 12 de outubro de 2021, https://web.dev/tbt/

(Walton – TTI 2021) Time to Interactive, Web.Dev, acedido em 12 de outubro de 2021, https://web.dev/interactive/?utm_source=lighthouse&utm_medium=devtools

(Walton et al. 2021) Cumulative Layout Shift, Web.Dev, Philip Walton and Milica Mihajlija, acedido em 12 de outubro de 2021, https://web.dev/cls

(Rabelo 2019) Arquitetura de software, Medium, Eduardo Rabelo, acedido em 22 de fevereiro de 2021, https://codeburst.io/software-architecture-the-difference-between-architecture-and-design-7936abdd5830

(Rachowicz 2017) When, How And Why Use Node.js as Backend, netguru, Justyna Rachowicz, acedido em 4 de fevereiro de 2021, https://www.netguru.com/blog/use-node-js-backend

(Sarmah 2019) Understanding the 5 factors of multi-factor authentication, Harshajit Sarmah, acedido em 2 de fevereiro 2021, https://analyticsindiamag.com/understanding-the-5-factors-of-multi-factor-authentication/

(Siddiqui 2018) Authentication vs Authorization, Data Driven Investor, Anum Siddiqui, acedido em 29 de janeiro de 2021, https://medium.com/datadriveninvestor/authentication-vs-authorization-716fea914d55

(Song 2019) Mother of all breaches, Victoria Song, Gizmodo, acedido em 1 de fevereiro de 2021, https://gizmodo.com/mother-of-all-breaches-exposes-773-million-emails-21-m-1831833456

(Stack Overflow 2019) Stack Overflow Survey, Stack Overflow, acedido em 4 de fevereiro de 2021, https://insights.stackoverflow.com/survey/2019

(Suprema 2020) Suprema, acedido em 1 de fevereiro de 2021, https://www.supremainc.com/en/main.asp

(Taylor 2019) Major breach found in biometrics system, Josh Taylor, The Guardian, acedido em 1 de fevereiro de 2021, https://www.theguardian.com/technology/2019/aug/14/major-breach-found-in-biometrics-system-used-by-banks-uk-police-and-defence-firms

(The Citizen 2018) Data breaches, Gemalto, The Citizen, acedido em 1 de fevereiro de 2021, https://thecitizenng.com/analysis-data-breaches-compromised-4-5bn-records-in-half-year-2018-gemalto/

(Trend Micro 2018) Data breach, Trend Micro, acedido em 1 de fevereiro de 2021, https://www.trendmicro.com/vinfo/us/security/definition/data-breach

(Trindade 2018) "A máquina no lugar do homem", UOL Brasil, Rodrigo Trindade, acedido em 26 de janeiro 2021, https://www.uol/tecnologia/especiais/inteligencia-artificial-vai-acabar-com-empregos-.htm

(Tutorials Point 2020) Cryptosystems, Tutorials Point, acedido em 2 de fevereiro de 2021, https://www.tutorialspoint.com/cryptography/cryptosystems.htm

(Web.Dev – Speed Index 2021) Speed Index, Web.Dev, acedido em 12 de outubro de 2021, https://web.dev/speed-index/?utm_source=lighthouse&utm_medium=devtools

(Wenzel 2020) SQL ACID Explained, Essential SQL, Kris Wenzel, acedido em 4 de fevereiro de 2021, https://www.essentialsql.com/what-is-meant-by-acid/

(Wikipedia 2019) List of data breaches, Wikipedia, acedido em 1 de fevereiro de 2021, https://en.wikipedia.org/wiki/List_of_data_breaches

(Yang 2020) Best Programming Languages to Learn, Fullstack Academy, David Yang, acedido em 4 de fevereiro de 2021, fullstackacademy.com/blog/nine-best-programming-languages-to-learn

(Zafatech 2019) Access Control Systems, Zafatech, acedido em 26 de janeiro de 2021, http://zafatech.com/our-solutions/access-control-systems/

Anexos

Anexo A – Caso de uso típico Single Sign-on

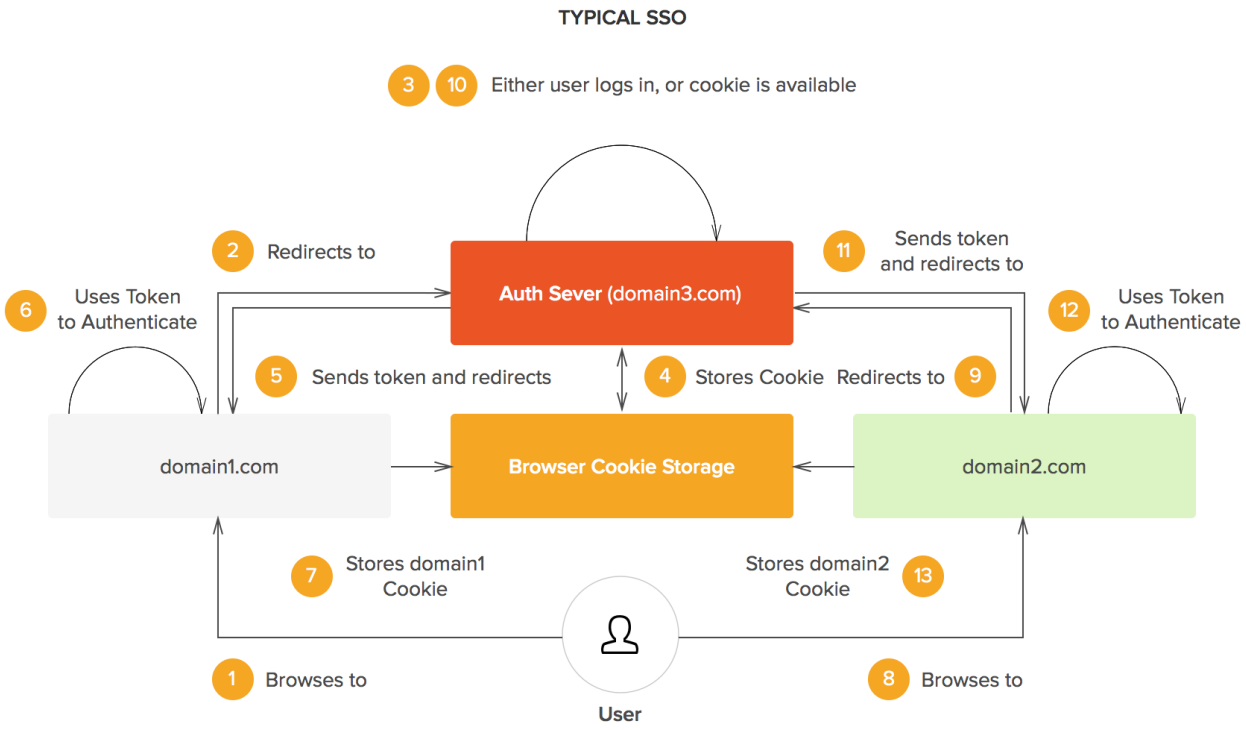

Figura 38 – Funcionamento típico de autenticação single sign-on (Peyrott 2015)

Anexo B – Protótipo final: Portal de administração

Neste anexo é apresentado o protótipo final do portal de administração desenvolvido, através da demonstração das interfaces com as quais o utilizador interage para configurar e parametrizar o sistema de controlo de acessos.

Autenticação

Figura 39 – Página de autenticação

Na Figura 39 podemos ver a página de autenticação, onde é possível autenticar no portal introduzindo-se as credenciais código e palavra-passe ou entrar através de uma conta externa numa entidade federada suportada, neste caso conta Microsoft, Google ou Facebook. Para utilizar autenticação *Single Sign-On* com uma destas contas externas, é necessário que o utilizador tenha um destes e-mails associados ao seu cadastro na plataforma.

Página inicial (*Dashboard*)

Após a autenticação no sistema, a aplicação poderá apresentar duas páginas distintas com base no sistema do cliente. Isto é, caso o sistema ainda não tenha nada parametrizado será apresentada uma página com uma explicação dos passos a seguir no menu lateral esquerdo, de modo a facilitar a definição e configuração do sistema de controlo de acessos (Figura 40). Se a aplicação detetar que o sistema já se encontra parametrizado será apresentada uma página inicial mais avançada com todos os conceitos visíveis no menu lateral esquerdo (Figura 41).

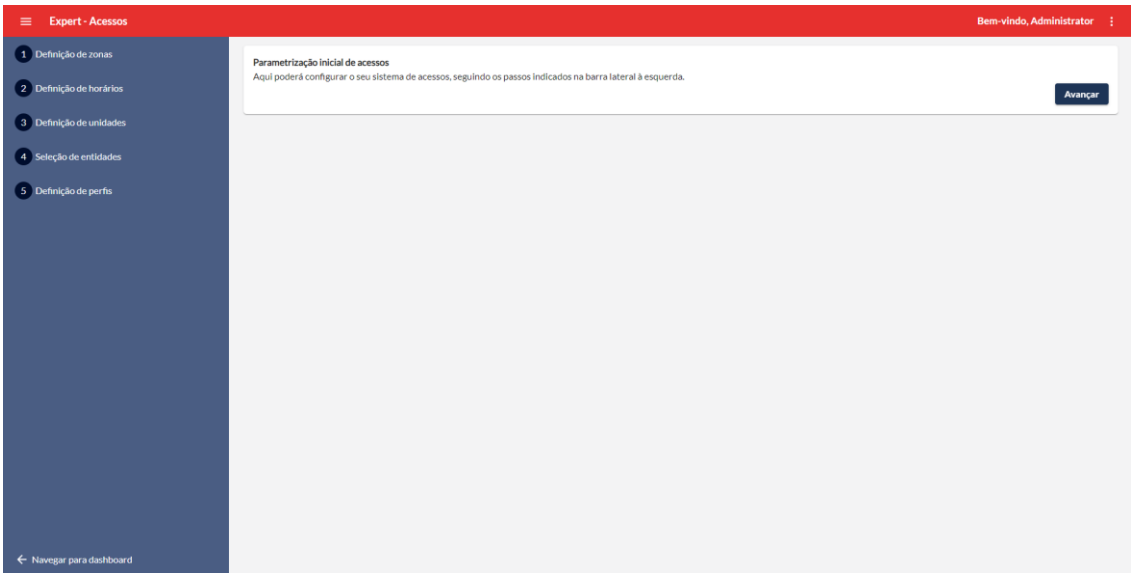

Figura 40 – Página de parametrização inicial de acessso

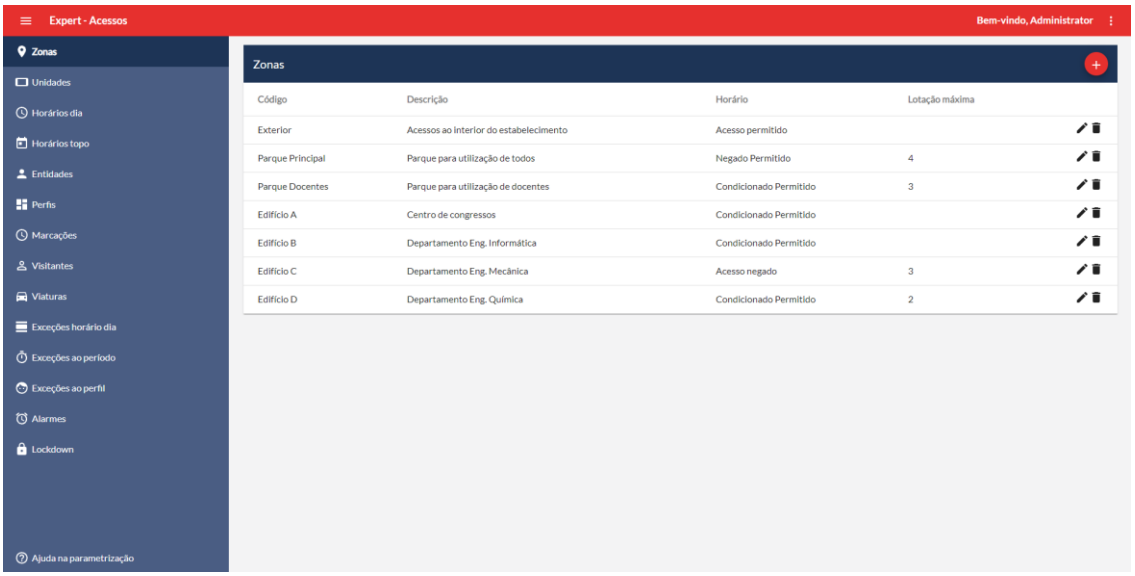

Figura 41 – *Dashboard* principal do sistema

Definição de zonas

Zona é o conceito base desta aplicação, representando o espaço físico onde se pretende controlar os acessos. Posto isto, o primeiro passo na parametrização do sistema de controlo de acessos é precisamente definir as zonas que se pretende controlar.

Figura 42 – Listagem de zonas

Na Figura 42 podemos ver a listagem de zonas já inseridas no sistema pelo administrador, com a possibilidade de criar, editar e apagar uma zona. Estes processos de criar, editar e apagar verificam-se em todos os conceitos do sistema e por consequente não será referenciado novamente ao longo da demonstração do protótipo, a possibilidade de efetuar-se estas operações.

Uma zona representa-se através de código e descrição, contendo a indicação do horário a ser utilizado por defeito e a lotação máxima desta. A criação de uma nova zona obriga apenas ao preenchimento do código.

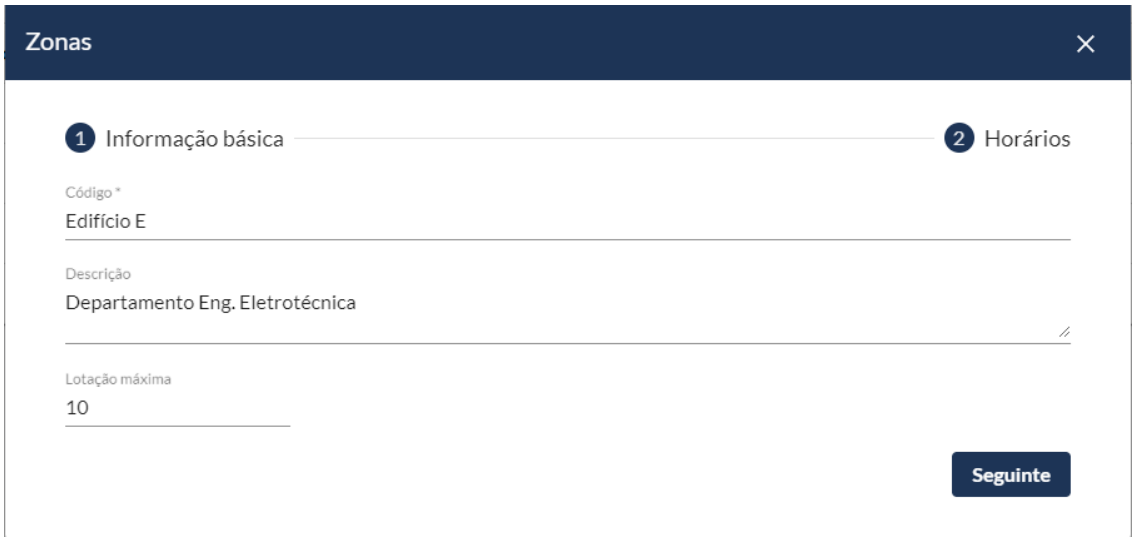

Figura 43 – Criação de uma zona

Definição de horários

O horário é o que permite determinar por último o acesso a uma determinada zona. Para tal, é necessário criar horários para que no momento da definição dos perfis se possa associar qual o horário a ser utilizado para aceder a uma determinada zona.

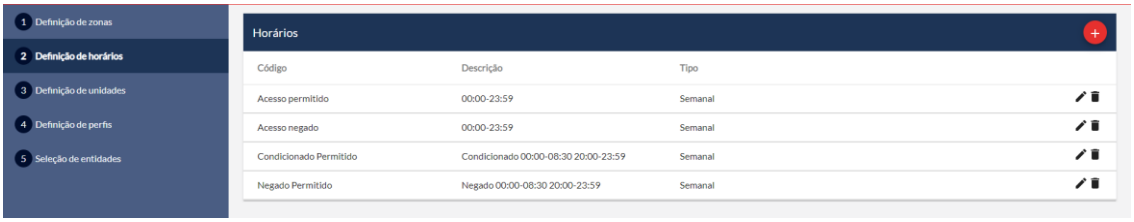

Figura 44 – Listagem de horários

Na Figura 44 é possível ver a listagem de horários existentes no sistema, contendo código, descrição e tipo de horário. Este tipo pode ser semanal ou rotativo.

O conceito do horário é algo complexo, sendo que um horário é constituído por vários "horário dia". Um "horário dia" é a definição do tipo de acesso (permitido, negado ou condicionado) a ser utilizado em um ou mais períodos nas 24 horas do dia.

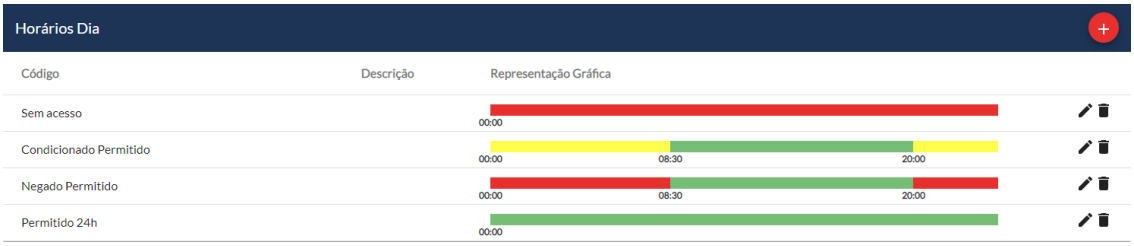

Figura 45 – Listagem de horários dia

Para ajudar na compreensão deste conceito, na Figura 45 estão apresentados os horários dia parametrizados no sistema onde é possível analisar, através da representação gráfica, os períodos e o seu tipo de acesso que fazem parte de um horário. Por exemplo, no horário "Negado Permitido" é possível analisar que existem três períodos, sendo que o acesso a uma zona com este horário só será permitido no intervalo das 08:30 às 20:00.

Posto isto, a criação de um horário passa por definir que horários serão utilizados em cada dia da semana no caso de ser semanal, ou definir a rotação indicando a data de referência do início da rotação. Na Figura 46 está apresentado um exemplo da criação de um horário do tipo semanal, sendo necessário indicar o código, horário a ser utilizado no feriado e os diferentes horários para cada dia da semana. De forma a auxiliar na criação, a aplicação sugere que o horário a ser utilizado no dia seguinte seja o mesmo do dia anterior.

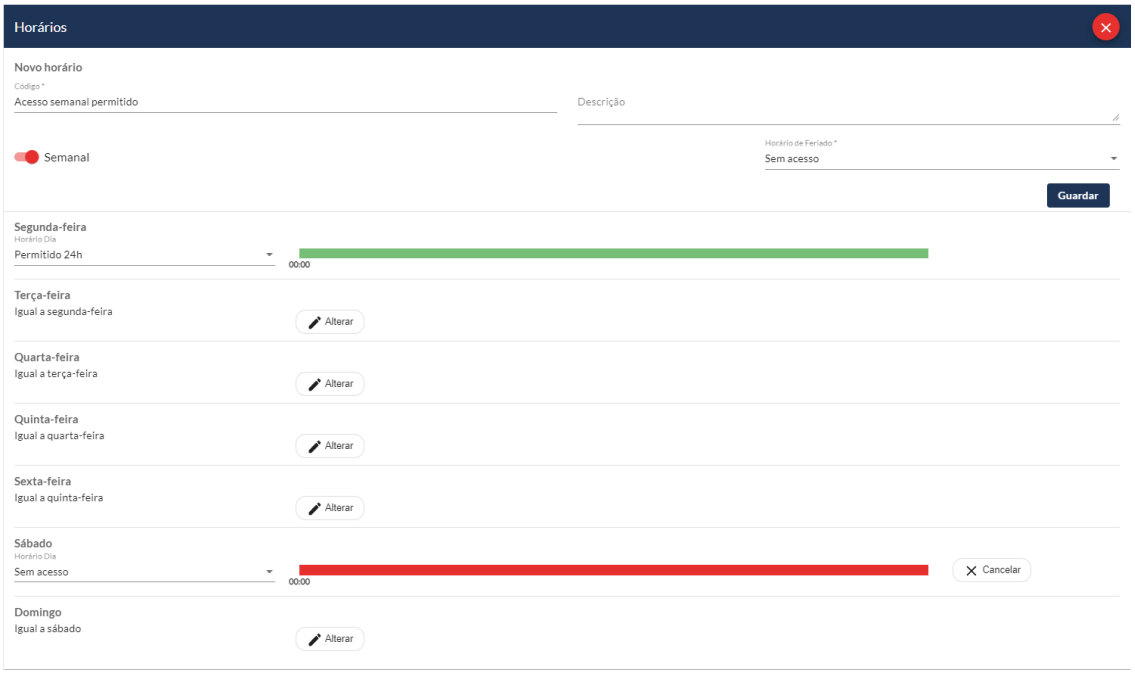

Figura 46 – Criação de horário

Na listagem de horários é ainda possível consultar o detalhe pressionando em cima de um dos horários da lista, sendo apresentado como se pode ver na Figura 47.

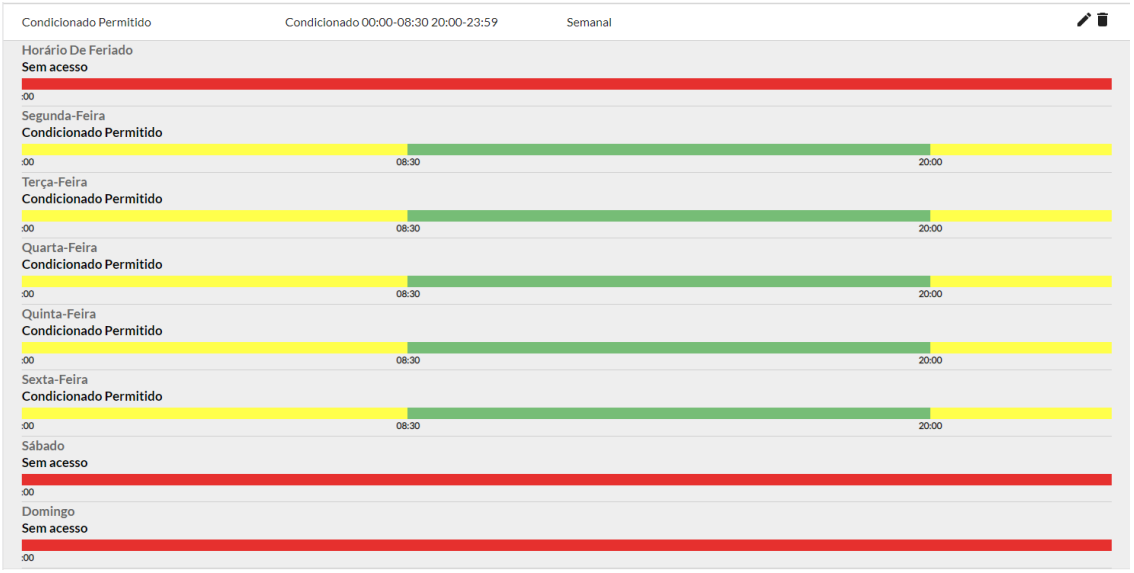

Figura 47 – Detalhe de um horário

Definição de unidades

A unidade é o conceito na aplicação que representa o equipamento físico que estará presente junto da entidade bloqueadora para desbloquear o acesso desta.

Figura 48 – Listagem de unidades

Estas unidades necessitam existir previamente no sistema, sendo que toda a configuração de rede e de encriptação da unidade é realizada na aplicação Windows. Neste portal de administração apenas se configura o que diz respeita ao sistema de controlo de acessos, neste caso as zonas limites (obrigatoriamente), os botões a aparecer no terminal, o estado da unidade e o sentido da marcação (Figura 49).

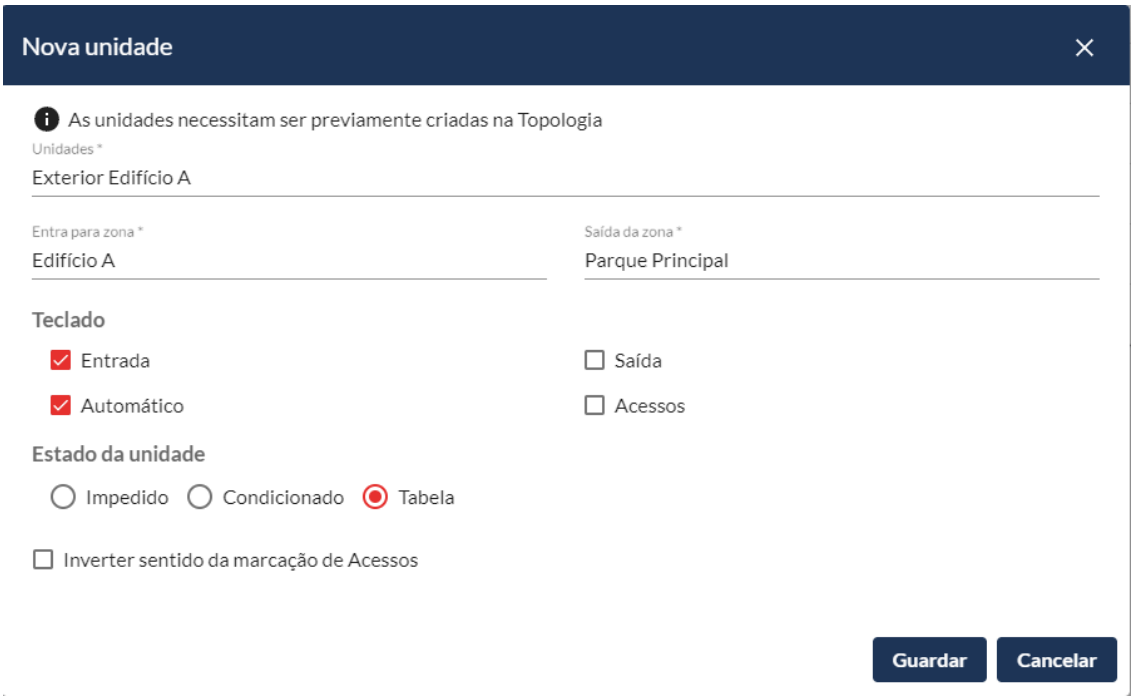

Figura 49 – Configuração de unidade para acessos

Definição de perfis

O perfil é o elo de ligação entre as entidades e as zonas/unidades nas quais estas vão ser reconhecidas. Nesta definição pretende-se que o administrador crie associações lógicas identificando as necessidades da existência de múltiplos perfis.

Figura 50 – Listagem de perfis

A listagem de perfis é básica, apresentando inicialmente apenas o código e descrição. No entanto, assim como na definição dos horários, também é possível consultar o detalhe de um perfil pressionando num que se pretenda analisar (Figura 51).

Figura 51 – Detalhe de um perfil

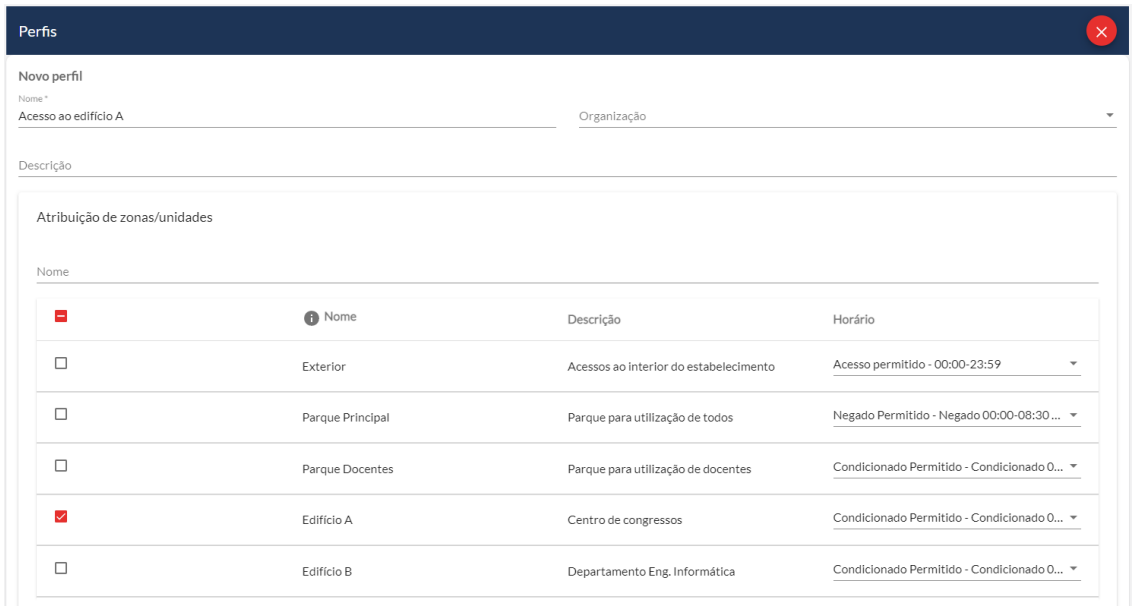

Figura 52 – Criação de perfil

Na Figura 52 está representado o ecrã para criação de um novo perfil, sendo necessário indicar o nome e pelo menos uma zona com o respetivo horário a utilizar neste perfil. O horário preenchido inicialmente será aquele que foi selecionado por defeito no momento de

criação da zona, todavia pode ser alterado neste momento de associação a um perfil, podendo uma zona ter diferentes horários de acesso para perfis diferentes.

Seleção de entidades

Uma entidade é um conceito abstrato da aplicação que representa um colaborador, uma entidade externa ou um cartão-chave (cartão físico atribuído às visitas para que estas sejam identificadas). De salientar que as entidades não são criadas/cadastradas nesta aplicação, mas sim no módulo de entidades da aplicação *core*, estando fora do âmbito do projeto e assumindo-se aqui que estas entidades existem previamente.

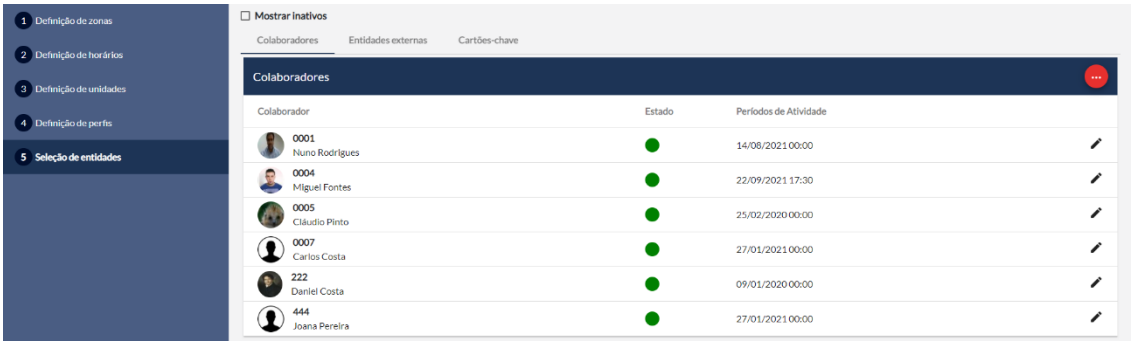

Figura 53 – Listagem de entidades (colaboradores)

Na Figura 53 é visível a listagem de entidades, neste caso de colaboradores, onde é possível ver a fotografia (caso possua no seu cadastro), o código, nome, representação gráfico do estado e o último período de atividade.

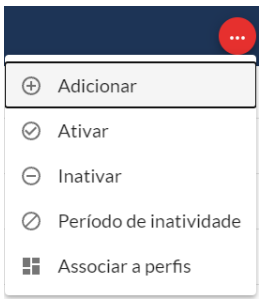

Figura 54 – Ações a realizar na página de entidades

Pressionando o botão com reticências, localizado no canto superior direito da tabela, será aberto um menu com todas as ações que é possível realizar nesta página de entidades (Figura 54):

1. Adicionar: adicionar novas entidades, que nunca fizeram parte do sistema de controlo de acessos;
2. Ativar: ativar entidades que se encontram inativas a partir de uma data especificada;
3. Inativar: inativar entidades que se encontram ativas a partir de uma data especificada;

4. Período de inatividade: inativar entidades que se encontram ativas num intervalo temporal especificado;
5. Associar a perfis: associação de entidades selecionadas a um ou mais perfis especificados;

Anexo C – Resultado completo do fluxo de teste CRUD de zonas

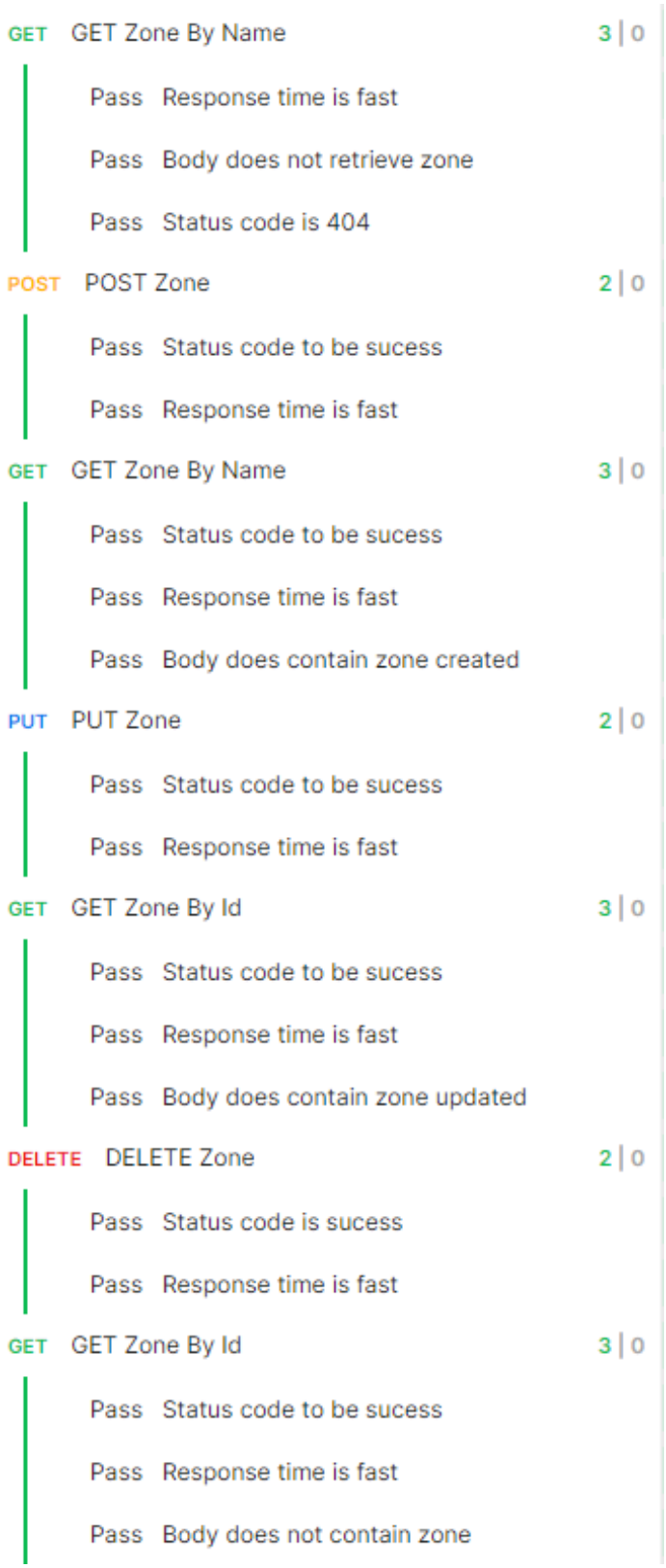

Figura 55 – Resultados dos testes ao CRUD de zonas (completo)