INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA

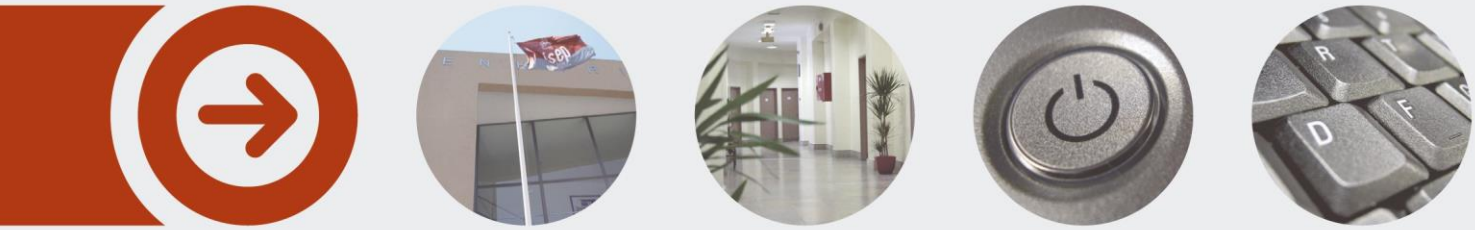



Framework para controlo de acessos

VASCO CHOUZAL FREIRE
Outubro de 2021

POLITÉCNICO
DO PORTO

INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA



Framework para controlo de acessos

VASCO CHOUZAL FREIRE

Outubro de 2021

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Framework para controlo de acessos

Vasco Chouzal Freire

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

Orientador: Paulo Baltarejo Sousa

Porto, outubro 2021

Dedicatória

Aos meu pais, irmãos, avós e namorada…

Vasco Freire

Resumo

As organizações têm de se preocupar cada vez mais com a segurança dos serviços e recursos que disponibilizam. Para tal, implementam uma camada de segurança nas suas aplicações que obriga a que os utilizadores tenham credenciais de acesso e permissão para aceder aos seus recursos.

O objetivo desta dissertação é elaborar um estudo sobre as soluções de Identity and Access Management existente no mercado e perceber as suas principais características e potencialidades. Será também elaborado um estudo referente à arquitetura implementada nas aplicações da organização i2S, bem como perceber qual é a melhor maneira de integrar uma das soluções, previamente estudadas, na arquitetura em causa.

Ao longo da análise conclui-se que existem duas alternativas para integrar a *framework* de Identity and Access Management com as aplicações desenvolvidas pela organização. A primeira alternativa consiste na utilização de ferramentas e *software* disponibilizado pela *framework* e implementá-las nas aplicações. A segunda alternativa consiste num maior desenvolvimento do lado das aplicações de modo a utilizarem os serviços *web* disponibilizados pela *framework* para executar as funcionalidades de autenticação e validação de autorização.

Após uma análise das alternativas, concluiu-se que se iria optar pela utilização dos serviços *web* por permitir uma maior agilidade na implementação e, através de desenvolvimento, permitir a utilização dos modos de autenticação e autorização já implementados pela organização.

Os testes executados sobre a implementação final demonstram que esta solução apresenta uma boa velocidade e capacidade de resposta para a autenticação e autorização sem falhas quando sobrecarregada.

Palavras-chave: Segurança, Identity and Access Management, Autorização, Autenticação

Abstract

Organizations have been more and more concerned with the security of their services and the resources they provide. To this end, they implement a layer of security in their applications that requires users to have access credentials and permission to access their resources.

The objective of this dissertation is to prepare a study on the Identity and Access Management solutions existing in the market and to understand their main characteristics and potentialities. A study on the architecture implemented on i2S organization applications will also be elaborated to understand what the best way is to integrate one of the solutions studied in this architecture.

Throughout the analysis it is concluded that there would be two alternatives to integrate the Identity and Access Management framework with the applications developed by the organization. The first alternative consists of using tools and software provided by the framework and implement them in the applications. The second alternative consists of further development on the application side to use the web services provided by the framework to perform the authentication and authorization validation features.

After an analysis of the alternatives, it is concluded that the use of web services would be opted for allowing greater agility in the implementation and, through development, allowing the use of the authentication and authorization modes already implemented by the organization.

Tests performed on the final implementation demonstrate that this solution has good speed and responsiveness for flawless authentication and authorization when overloaded.

Keywords: Security, Identity and Access Management, Authorization, Authentication

Agradecimentos

Em primeiro lugar, agradeço ao professor Paulo Baltarejo Sousa por todos os conselhos que me deu e que permitiram a escrita deste documento.

De seguida agradeço à minha equipa de trabalho na i2S, em especial ao António Pinto, supervisor deste projeto, que me orientou no desenvolvimento do mesmo.

Por fim, um agradecimento especial aos meus pais, avós e namorada que me apoiaram durante esta etapa da minha carreira académica.

A todos, muito obrigado.

Índice

Lista de Figuras

Lista de Tabelas

Lista de Excertos de Código

Acrónimos e Símbolos

Lista de Acrónimos

AHP	*Analytic Hierarchy Process*
AOP	*Aspect Oriented Programming*
API	*Application Programming Interface*
B2B	*Business to Business*
B2C	*Business to Customer*
BYOI	*Bring Your Own Identity*
CIAM	*Custom Identity and Access Management*
CORBA	*Common Object Request Broker Architecture*
DTM	*Decentralized Trust Management*
HTTPS	*Hyper Text Transfer Protocol Secure*
IAM	*Identity and Access Management*
IDP	*Identity Provider*
IGA	*Identity Governance and Administration*
IP	*Internet Protocol*
IT	*Information Technology*
JAX-WS	*Java API for XML Web Services*
JAX-RS	*Java API for RESTful Web Services*
JMS	*Java Message Service*
JMX	*Java Management Extensions*
JVM	*Java Virtual Machine*
JWS	*JSON Web Signature*
JWT	*JSON Web Token*
LCM	*Life Cycle Management*
LDAP	*Lightweight Directory Access Protocol*

MFA	*Multi-factor Authentication*
MSC	*Modular Service Container*
POC	*Proof of Concept*
RADIUS	*Remote Authentication Dial In User Service*
REST	*Representation State Transfer*
RBAC	*Role-based Access Control*
RDBMS	*Relational Database Management System*
SaaS	*Software as a Service*
SAML	*Security Assertion Markup Language*
SEA	*Secure Entitlement and Authentication*
SSO	*Single Sign On*
SOAP	*Simple Object Access Protocol*
SWOT	*Strengths, Weaknesses, Opportunities, and Threats*
UAI	*User Authentication Information*
UEBA	*User and Entity Behavior Analytics*
UML	*Unified Modeling Language*
XML	*Extensible Markup Language*

1 Introdução

Este capítulo apresenta uma breve descrição do projeto desenvolvido no âmbito da unidade curricular de Tese de Mestrado do Departamento de Engenharia Informática (TMDEI) do segundo ano do Mestrado de Engenharia Informática (MEI) de Sistemas Computacionais, do Instituto Superior de Engenharia do Porto (ISEP).

1.1 Enquadramento

Um dos maiores desafios dos dias de hoje no mundo da informática é a segurança. Como o envio de informação através da Internet se tornou crucial para o funcionamento e sucesso das empresas, também se tornou importante que os métodos e estratégias para o controlo e acesso a estes recursos fossem melhorados.

Muitas vezes, as empresas implementam diferentes soluções de gestão de segurança para responder a problemas específicos e cada uma destas tecnologias operam de forma independente, o que pode causar uma ineficiência no seu uso[1].

Neste sentido, a i2S – Insurance Knowledge pretende implementar uma solução de Identity and Access Management (IAM), que permita gerir os acessos às suas aplicações e respetivos recursos.

1.2 Problema

As *frameworks* de IAM têm como objetivo identificar, autenticar e gerir autorizações de utilizadores a recursos, como aplicações e os seus serviços, assim como a computadores e outros dispositivos [2]. Tipicamente, esta gestão de permissões é feita agregando-as em papéis que por sua vez são atribuídos aos utilizadores.

Os sistemas que mais frequentemente implementam as funcionalidades de IAM, recorrem a serviços externos para identificação e autenticação como Windows Active Diretory, IBM Tivoli Directory, entre outros [3]. A gestão de autorizações destes sistemas é baseada em papéis e/ou grupos combinado com as permissões definidas por cada recurso.

Na implementação de uma nova *framework* de IAM pretende-se que todas estas funcionalidades sejam mantidas e centralizadas assegurando níveis elevados de alta disponibilidade, tolerância a falhas e suporte a vários Relational Database Management System (RDBMS) como Oracle, SQL Server e DB2 [4].

1.3 Objetivos

A intenção de implementação de uma *framework* IAM vem com a necessidade de proteger os recursos da empresa. Para tal, os objetivos propostos podem ser sintetizados nos seguintes pontos:

- Analisar as *frameworks* de IAM existentes no mercado e identificar as suas especificidades e funcionalidades para obter uma visão global das soluções;
- Analisar o processo de IAM existente na solução atual;
- Definir os requisitos que a solução implementada terá de ter;
- Elaborar um estudo para selecionar qual das *frameworks* de IAM analisadas anteriormente completa melhor os requisitos definidos;
- Analisar as alterações que serão necessárias a fazer para implementar a *framework* selecionada;
- Desenhar uma possível solução final, tendo em conta a necessidade de alta disponibilidade do sistema;
- Implementação do novo processo IAM utilizando a solução selecionada;
- Avaliar a solução implementada, tendo em conta a alta disponibilidade, performance e tolerância a falhas.

No final deste projeto, a solução implementada deverá responder a todos os objetivos pretendidos.

1.4 Abordagem preconizada

Com o estudo realizado sobre as *frameworks* IAM, pretende-se desenhar uma solução onde, utilizando esta nova ferramenta, se consiga centralizar todas as ações de gestão de acessos e permissões das aplicações i2S, assegurando as funcionalidades e performance dos serviços destas mesmas.

Para tal, foi elaborado um plano de trabalhos para o desenvolvimento deste projeto, excluindo a escrita da dissertação. Neste plano existem quatro fases importantes:

1. Análise das *frameworks* IAM existentes no mercado;
2. Avaliação da implementação da nova *framework* IAM;
3. Implementação da *framework* IAM;
4. Avaliação da solução implementada.

A Figura 1 mostra o plano de trabalhos para este projeto.

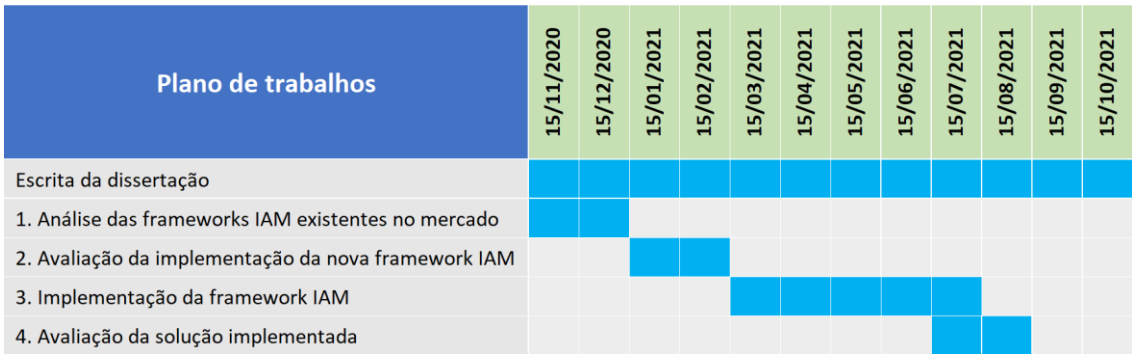


Figura 1 – Diagrama de Gantt com o plano de trabalhos

Inicialmente foi elaborada uma análise das *frameworks* de IAM existentes no mercado, selecionando as mais relevantes para se poder fazer uma avaliação das mesmas e seleção de qual será a melhor opção tendo em conta as suas funcionalidades. Após a seleção da *framework*, será elaborada uma avaliação para a sua implementação e do que se terá de desenvolver para que se possa fazer a integração com o sistema atual.

Segue-se a implementação e integração da solução, cumprindo com o desenho anteriormente definido e com as boas práticas de engenharia e desenvolvimento de *software*. Por fim, o sistema será testado e validado de modo a se perceber se cumpre com todos os requisitos definidos.

Durante todas estas fases, a escrita da dissertação decorreu em paralelo.

1.5 Estrutura do documento

Este documento está dividido em capítulos, onde cada um se refere a uma parte da análise e desenvolvimento feito no projeto. Cada capítulo é constituído por secções que permitem que o conteúdo seja mais bem entendido. Os capítulos existentes são os seguintes:

- Contextualização;
- Estado de arte;
- Análise;
- Desenho da solução;
- Avaliação;
- Conclusão.

No capítulo Contextualização, é feito uma breve apresentação da empresa onde o projeto foi desenvolvido e descrito o problema de uma maneira mais aprofundada onde são apresentadas as questões e dúvidas que levaram à necessidade do desenvolvimento deste projeto. É também apresentado o tema de IAM e que

características as *frameworks* deste tipo podem ter. Por fim, é feita uma análise de valor onde é apresentado o valor que este projeto traz ao cliente.

No capítulo Estado de arte, são apresentadas as tecnologias que se enquadram no tema e que poderão resolver o problema em causa. Para cada tecnologia são apresentados os pontos positivos e negativos de modo a poder fazer uma avaliação e escolher a mais apropriada ao problema. São também apresentadas algumas tecnologias relevantes que foram utilizadas ao longo do desenvolvimento deste projeto.

No capítulo Análise, são descritos os requisitos funcionais e não funcionais necessários para que os objetivos sejam cumpridos. É também feita a decisão de qual das *frameworks* analisadas foi utilizada para o desenvolvimento do projeto. Por fim, são apresentadas as possíveis maneiras de implementar e migrar a solução atual para a utilização da *framework* escolhida anteriormente e é feita uma análise para se escolher qual das abordagens apresentadas será utilizada.

No capítulo Desenho da solução, é apresentada uma análise técnica do problema com a utilização de diagramas Unified Modeling Language (UML) e a respetiva descrição para apresentar a nova infraestrutura, novos módulos e padrões utilizados.

No capítulo da Avaliação, são descritos os testes efetuados à solução desenvolvida de modo a perceber a validade e fiabilidade da solução implementada. São também apresentados os resultados destes testes.

Por fim, o capítulo da Conclusão, onde são apresentadas as conclusões relativas ao projeto desenvolvido, descrevendo os pontos atingidos e não atingidos, se o produto final corresponde às expectativas apresentadas inicialmente e se existem possíveis melhorias.

2 Contextualização

Para se compreender um projeto na sua totalidade, é importante entender os conceitos associados ao mesmo e em que contexto se aplica, pois só assim é possível fundamentar decisões a tomar.

Para tal, neste capítulo é feita uma contextualização do projeto apresentando a empresa onde foi desenvolvido e descrevendo o tema de IAM. É também apresentado o problema que levou ao desenvolvimento do projeto e é feita uma análise de valor do trabalho realizado no contexto desta dissertação.

Ao longo deste capítulo será dada a resposta às seguintes questões:

1. O que são *frameworks* IAM?
2. Como é a arquitetura de um sistema IAM?
3. Que mais valia traz um sistema IAM a uma empresa?
4. Qual a mais valia do trabalho apresentado neste documento para a implementação de uma *framework* IAM?

2.1 i2S Insurance Knowledge

Fundada há mais de trinta anos, a i2S *Insurance Knowledge* é uma empresa que se dedica ao desenvolvimento de soluções de *software* para a indústria seguradora e que consolidou a sua posição de liderança nos mercados de seguros de Portugal, Angola e Moçambique, com implementações de *software* em vários outros países, num total de mais de quarenta companhias de seguros, apresentando assim um volume de negócios anual de mais de treze milhões de euros [5].

Como empresa com mais de trinta anos, e como muitas empresas, passaram por lá muitos colaboradores que contribuíram com o seu conhecimento e experiência para o desenvolvimento do *software* que ainda existe nos dias de hoje.

A longa história da i2S tem uma face que é considerada um problema pois com o passar dos anos e com o passar do conhecimento de colaborador para colaborador, muito do código existente, as *frameworks* e processos existentes não estão devidamente atualizados o que pode levar a alguns problemas de segurança que, nos dias de hoje, é considerado como um dos tópicos mais importantes.

Apesar de tudo isto, em 2016 recebeu o prémio PME Inovação COTEC 2016 [6] o que mostra que se tem feito um grande esforço para que os produtos da i2S sejam cada vez mais inovadores. Uma prova disto é a capacidade de adaptação das soluções i2S às necessidades dos clientes.

Em 2019, o grupo Inetum, comprou a i2S com o objetivo de, em conjunto com a empresa Cleva, construir uma solução completa para as companhias de seguros. Deste modo, a i2S passou a pertencer a um grupo com mais de 19500 colaboradores com presença em mais de vinte países e que em 2018 gerou um volume de negócio de 1395 milhões de euros.

Pertencendo a grupo deste tamanho pode ter consequências maiores quando falando de problemas de segurança, pois possíveis sanções podem chegar aos vinte milhões de euros ou a um valor equivalente a 4% do volume de negócios mundial da empresa. Deste modo, este projeto tem também como objetivo corrigir possíveis problemas segurança tal como a implementação de uma solução IAM para a gestão centralizada dos acessos às soluções i2S.

Tendo agora como objetivo construir uma solução mais completa para as companhias de seguros em conjunto com a empresa Cleva, tornou-se necessário que a i2S adapta-se parte dos seus produtos para utilizar ferramentas externas utilizadas pelos produtos da Cleva. Deste modo surgiu a necessidade deste projeto e de implementar uma *framework* IAM para fazer a gestão dos utilizadores e das suas permissões.

2.2 Identity and Access Management

Cientistas e administradores de Information Technology (IT) têm vindo a trabalhar para desenvolver um método para a proteção dos dados e recursos das organizações. Mesmo com todo este trabalho, as pessoas ainda não se sentem seguras ao utilizarem muitas plataformas na Internet que lhes pedem por dados pessoais [7].

Para promover a segurança e tentar retirar a insegurança do lado do utilizador, muitas empresas estão a implementar sistemas de IAM para proteger o ambiente da organização.

Utilizando estes sistemas de IAM e implementando acessos com base em privilégios de utilizadores individuais é possível obter o melhor resultado de segurança de um sistema. Neste modelo, os utilizadores recebem, ou não, permissão com base na sua função e respetivos privilégios de acesso.

O principal objetivo dos sistemas IAM é que o utilizador possua apenas uma identidade digital, e, uma vez que esta é criada no sistema é mantida, monitorizada e pode ser modificada quando exigido pelos utilizadores por meio do ciclo de vida de gestão do utilizador.

É com este tipo de sistemas que é possível atingir o objetivo principal de gerir as identidades, a autenticação e autorização dos utilizadores e, assim, aumentar a sua segurança [7].

Um sistema IAM tem uma estrutura flexível e escalonável que fornece uma arquitetura utilizada para fornecer segurança. Esta estrutura pode ter vários componentes como os apresentados na Figura 2.

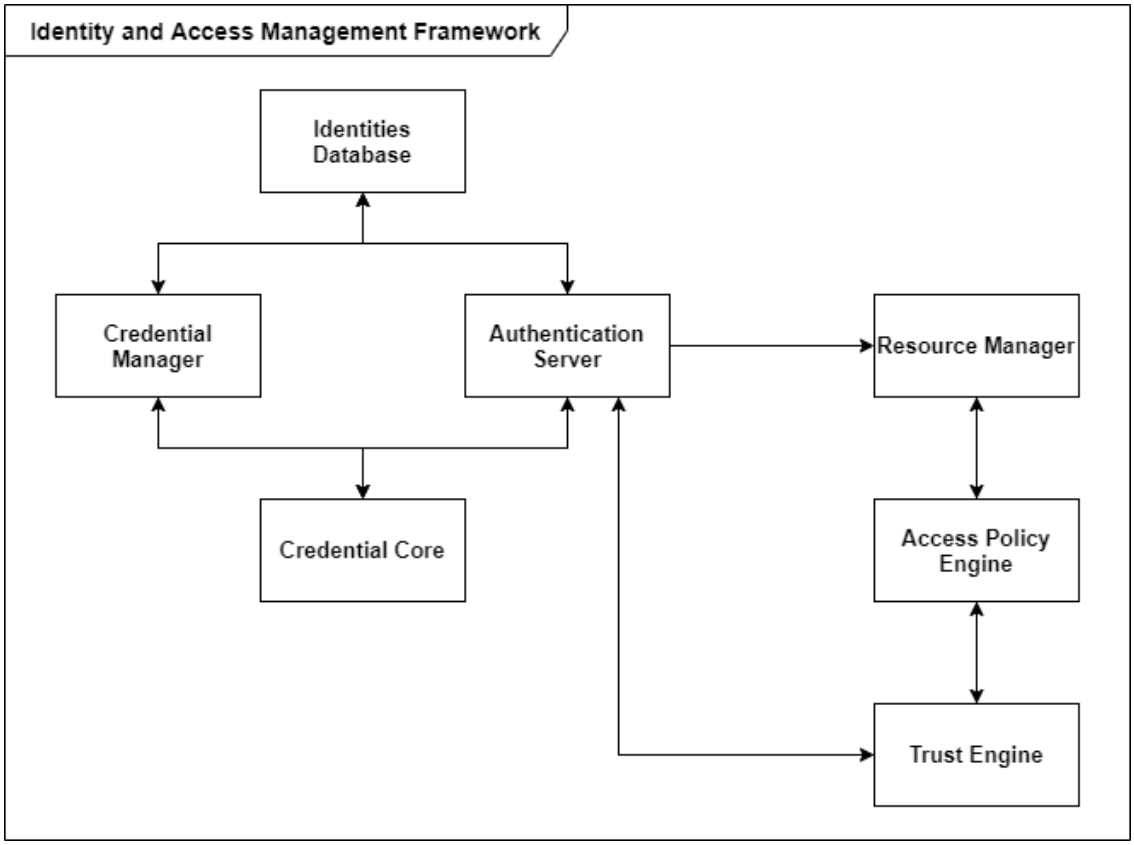


Figura 2 – Componentes de uma arquitetura IAM (adaptado de [1])

A base de dados de identidades, identificado na Figura 2 com a etiqueta *Identities Database,* tem como objetivo armazenar os perfis associados aos utilizadores que têm acessos aos recursos. Normalmente, a base de dados de identidades está conectada ao *Credential Manager* e ao *Authentication Server* que a utilizam para conseguirem funcionar e persistirem os dados que produzem.

O gestor de credenciais (*Credential Manager*) armazena e gere os diferentes tipos de credenciais que podem ser fornecidos por um utilizador. Estas credenciais fornecidas podem incluir nomes de utilizador e *passwords, passwords* de uso único ou qualquer outro tipo de informação de autenticação fornecida por um utilizador e que o sistema esteja configurado para receber. O gestor de credenciais pode estar conectado com o núcleo de credenciais (*Credential Core*).

O núcleo de credenciais, identificado na Figura 2 com a etiqueta *Credential Core,* contém um conjunto de componentes de serviços de rede que permitem gerir o ciclo de vida dos diferentes tipos de credenciais. Este ciclo de vida inclui o período de validade das credenciais para que esta componente possa gerir a inicialização e expiração das credenciais. O núcleo de credenciais pode ser conectado à base de dados de credenciais que armazena a informação necessária de cada tipo de credencial.

O servidor de autenticação (*Authentication Server*) tem como principal objetivo autenticar credenciais fornecidas por um utilizador para aceder a recursos protegido pela estrutura de IAM. Este componente utiliza um modelo de confiança fornecido pelo

mecanismo de confiança para permitir o acesso do utilizador aos recursos protegidos. As credenciais fornecidas pelo utilizador ao servidor de autenticação têm de ser compatíveis com o que é configurado nos tipos de credenciais armazenados no gestor de credenciais. O servidor de autenticação também tem a capacidade de auditoria que lhe permite determinar, por exemplo, quantas vezes um determinado tipo de credencial é solicitado pelo utilizador ou o número de vezes que o utilizador é solicitado a fornecer as credenciais antes destas serem validadas com sucesso. Esta componente é uma das mais importantes na estrutura de IAM pois, para além de ser a que gere os acessos, permite a integração de qualquer servidor de autenticação que atenda aos requisitos de segurança da empresa.

O gestor de recursos, identificado na Figura 2 com a etiqueta *Resource Manager,* tem como objetivo permitir que o utilizador visualize os recursos que este tem permissão para aceder. A conexão com um recurso em específico é facilitada pela utilização de um protocolo de asserção exigido por esse recurso em específico. Estes protocolos de asserção suportados pelos recursos podem incluir Kerberos, Security Assertion Markup Language (SAML), Site Minder R, Windows Integrated Authentication, Secure Entitlement and Authentication (SEA), OpenID Connect e etc.. O gestor de recursos tem também como objetivo traduzir a User Authentication Information (UAI) fornecida pelo servidor de autenticação para ser utilizada pelo protocolo de asserção exigido pelo recurso e, para isso, constrói dinamicamente o formato de asserção correto a partir da UAI para autenticar automaticamente o utilizador com o recurso. Este componente em também a capacidade de *Single Sign On* (SSO) para que outros recursos que suportem protocolos de asserção comuns. Este mecanismo, normalmente, utiliza *tokens* que são dados seguros utilizados para transmitir informações confidenciais entre duas partes de maneira compacta e independente [8].

O mecanismo de política de acessos, identificado na Figura 2 com a etiqueta *Access Policy Engine,* tem como objetivo determinar se um determinado utilizador tem acesso ao recurso solicitado. Para tal, é configurado para receber um nível de confiança do mecanismo de confiança e a UAI do gestor de recursos para determinar se o utilizador tem, ou não, permissão para aceder ao recurso solicitado.

Por último, o mecanismo de confiança, identificado na Figura 2 com a etiqueta *Trust Engine* é utilizado para determinar um nível de confiança necessário para o utilizador ou uma sessão do utilizador. Este componente é integrado com o servidor de autenticação para um serviço de autenticação mais eficaz. É com base na UAI gerado pelo servidor de autenticação que o mecanismo de confiança atribui às sessões dos utilizadores o nível de confiança apropriado. As regras para a atribuição dos níveis de confiança são definidas por um conjunto de regras de negócio definidas pela empresa que utiliza a estrutura de IAM.

Esta estrutura descrita pode ser utilizada pelas empresas para se basearem na arquitetura de segurança que pretendem seguir, pois estes componentes não limitam a implementação ou desenvolvimento de uma estrutura diferente da apresentada. A estrutura de IAM pode incluir outros componentes adicionais ou pode integrar componentes juntos, mas que no fim ofereça as funcionalidades descritas acima [1].

2.3 Contexto e problema

IAM é um dos principais desafios para as empresas para fornecer acesso seguro aos seus recursos [9]. Para que as infraestruturas IAM e os seus processos sejam implementados com sucesso é necessário que sejam cumpridos os requisitos de conformidade.

No início, havia um pequeno número de padrões e regulamentos que deveriam ser adotados na implementação deste tipo de soluções. Mas, atualmente, os governos e entidades reguladoras de cada país, impõem cada vez mais requisitos de conformidade que só podem ser implementadas através de processos, diretrizes e tecnologias padronizadas de IAM [9].

Atualmente as soluções de IAM permitem que a maioria dos fluxos de administração de utilizadores sejam automatizados com sucesso fornecendo funcionalidades para análise de segurança e também permitindo a integração de serviços de federação e integração em nuvem.

A i2S como empresa de *software*, pretende que os seus produtos tenham uma camada de segurança que permita a gestão de quem pode aceder a determinado recurso de cada produto.

Atualmente a i2S já tem implementado um módulo de segurança desenvolvido internamente. Infelizmente a solução existente não permite ter uma plataforma centralizada que funcione como uma solução IAM para gerir os acessos às aplicações e recursos existentes.

Apesar de não permitir esta gestão centralizada, a solução atual tem algumas funcionalidades que se deverão manter após a implementação ou desenvolvimento de uma nova *framework*. Algumas das funcionalidades são:

- Permite gerar um *token* através do nome de utilizador e respetiva *password*;
- Permite aceder a recursos protegidos utilizando um *token* válido;
- Permite SSO entre aplicações;
- Permite revogar o *token*.

Todas as funcionalidades apresentadas acima deverão ser mantidas. Dependendo da análise efetuada e da solução selecionada para implementar, poderão ser adicionadas novas e/ou alteradas algumas existentes.

Para além destas restrições funcionais, existem algumas restrições de infraestrutura tais como a necessidade de:

- A *framework* tem de suportar vários RDBMS como Oracle, SQL Server e DB2 for i;

- A *framework* tem de permitir a autenticação utilizando serviços externos como Windows Active Diretory, IBM Tivoli Directory *e etc*.

É através da resolução destes desafios anteriormente apresentados que contribuem para que os produtos da i2S sejam mais reconhecidos no mundo dos seguros por oferecerem uma maior segurança e proteção dos dados e uma maior capacidade para gerir os acessos dos utilizadores aos seus recursos.

2.4 Análise de valor

Nas secções anteriores foram apresentados os motivos e o contexto que levaram à necessidade da implementação de uma *framework* que permita a gestão de autorizações para o acesso às aplicações.

Nesta secção será apresentado o modelo New Concept Development (NCD), de Peter Koen [10], identificando qual o valor gerado e percebido pela conclusão deste projeto.

2.4.1 Modelo NCD

O modelo NCD, de Peter Koen, é apresentado na Figura 3.

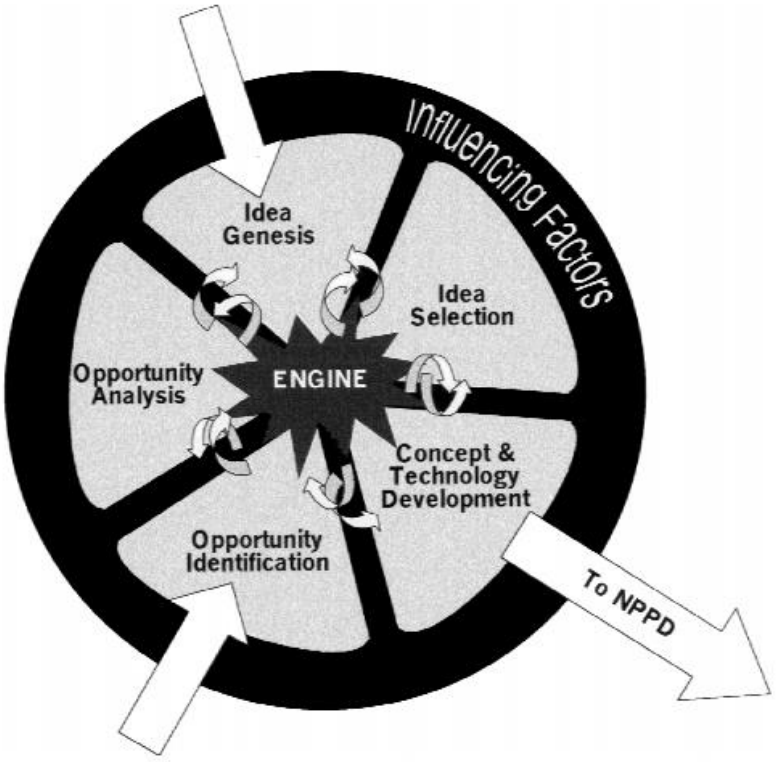


Figura 3 – Diagrama do modelo NCD [10]

O diagrama de NCD apresenta cinco elementos chave que compõem o modelo Front End Innovation (FEI), identificação de oportunidade (*Opportunity Identification*), análise

da oportunidade (*Opportunity Analysis*), geração de ideias (*Idea Genesis*), seleção de ideias (*Idea Selection*) e definição de conceito (*Concept & Technology Development*). O motor deste diagrama, identificado na figura pela etiqueta "*Engine*", impulsiona através da liderança e cultura da organização os cinco elementos do modelo FEI. Os fatores influenciadores consistem nas capacidades organizacionais, na estratégia de negócios, nos canais de distribuição, clientes e concorrentes e na ciência capacitadora que é utilizada. Estes fatores influenciam todo o processo de inovação que inclui o FEI [10].

De seguida serão apresentados os cinco elementos que compões o modelo FEI.

2.4.1.1 Identificação de Oportunidade

A identificação de oportunidade tem como objetivo identificar um negócio ou uma tecnologia que possa trazer algo de positivo para a empresa. Estas oportunidades deverão ser concretas e bem definidas para que, eventualmente, sejam alocados recursos para trabalhar nesses projetos. Existem vários métodos que as empresas utilizam para identificarem estas oportunidades. Podem utilizar ferramentas ou técnicas como *brainstorming,* sessões *ad hoc,* entre outras [10].

Os problemas apresentados anteriormente representam uma oportunidade para implementar uma solução sólida de IAM utilizando uma *framework* externa. Esta oportunidade por si só não representa qualquer acréscimo de valor à empresa, o que torna necessário efetuar uma análise de como se irá implementar e qual *framework* se irá utilizar.

2.4.1.2 Análise da oportunidade

A análise de oportunidade, tal como o nome indica tem como objetivo analisar a oportunidade identificada no ponto anterior de modo a identificar os pontos fortes e pontos fracos da mesma, podendo tirar algumas conclusões relativamente ao valor que esta acrescenta ao produto e ao custo do seu desenvolvimento. Uma ferramenta utilizada frequentemente para fazer esta análise é a utilização da análise Strengths, Weaknesses, Opportunities, and Threats (SWOT).

A análise SWOT é uma abordagem que permite considerar as fraquezas e as potencialidades do desempenho que uma organização encontra nos seus ambientes internos e externos. As forças apresentam o desempenho pretendido, as fraquezas são os inibidores do desempenho desejado, as oportunidades são o que podem potencializadores que podem contribuir para o sucesso da ideia e as ameaças identificam os fatores que podem fazer com que a ideia não se concretize na totalidade [11].

De seguida será apresentada a análise SWOT, na Tabela 1, para a oportunidade identificada.

Tabela 1 – Análise SWOT

Forças	**Oportunidades**
• Plataforma robusta para a gestão de acessos; • Delegação da gestão de utilizadores a um sistema externo; • Simplicidade para o gestor do sistema;	• Melhorar a funcionalidade da gestão de utilizadores; • Centralizar as funcionalidades numa plataforma.
Ameaças	**Fraquezas**
• Várias *frameworks* de IAM no mercado; • Suporte para a *framework*.	• Dificuldade de integração com o sistema.

A implementação de uma *framework* IAM robusta representa um ponto forte pois permite que não haja a necessidade de ser feito um desenvolvimento das mesmas funcionalidades que uma *framework* externa já o permite fazer. A delegação da gestão de utilizadores numa plataforma externa também é um ponto positivo pois, tal como o ponto anterior, não existe a necessidade de ser feito nenhum desenvolvimento e o número de possíveis *bugs* nestas funcionalidades é inferior. Por fim, esta plataforma fará com que o trabalhado do gestor do sistema tenha a sua tarefa mais simplificada pois a *framework* trará uma agilidade para a gestão dos utilizadores.

Relativamente às oportunidades, foram identificadas duas grandes oportunidades. A implementação de uma nova *framework* IAM permitirá melhorar as funcionalidades de gestão de utilizadores relativamente à implementação existente atualmente. Esta implementação permitirá também simplificar todas as funcionalidades relativas gestão de autorizações, acesso e utilizadores pois estas estarão todas centralizadas numa única plataforma.

As ameaças identificadas estão relacionadas com a procura da *framework* final, pois existem várias no mercado o que pode dificultar a avaliação de todas as soluções encontradas. Outra ameaça encontrada é relativa ao suporte futuro que poderá ser necessário no caso de haver algum problema com a *framework*. Este suporte poderá ser demorado o que poderá trazer problemas a implementações futuras em clientes.

Por fim, a fraqueza encontrada é relativa à integração da *framework* com o sistema da empresa. Esta integração poderá ser complicada dependendo do tipo de implementação e que utilização será feita da *framework* IAM.

2.4.1.3 Geração de ideias

A geração de ideias é o desenvolvimento e a maturação da oportunidade identificada em uma ideia concreta. Esta fase pode ser um processo formal, incluindo sessões de *brainstorming* de modo a provocar a organização a gerar ideias novas para a oportunidade [10].

No caso deste projeto esta geração de ideias está dependente da análise, avaliação e seleção da *framework* de IAM a ser utilizada.

2.4.1.4 Seleção de ideias

Na seleção de ideias é feita uma análise das ideias geradas no ponto anterior de modo a se perceber as suas vantagens e desvantagens para que se possa tomar uma decisão. É necessário perceber se o retorno da implementação de cada ideia compensa o esforço realizado.

A seleção de ideias foi efetuada tendo em conta a *framework* de IAM selecionada para implementar e os requisitos funcionais e não funcionais identificados.

2.4.1.5 Definição de conceito

A definição de conceito tem como objetivo avaliar se a ideia trabalhada se deverá desenvolver. Deste modo, deverá se fazer uma avaliação dos objetivos e elaborar um plano, a apresentar à organização, que descreva em que medida esta ideia pode afetar positivamente o produto final.

Assim, a implementação desta ideia permite que a gestão dos acessos às aplicações desenvolvidas pela organização seja simplificada e centralizada numa única plataforma de forma que a sua utilização seja mais intuitiva para o gestor de utilizadores e que as aplicações estejam protegidas por uma infraestrutura robusta.

2.4.2 Definição de valor

Para se entender a importância de um produto ou serviço é importante definir o seu próprio valor. O valor de um produto consiste na sua relevância que este tem para o cliente. É importante que as organizações se preocupem com o valor dos seus produtos e serviços pois a maioria das suas soluções advêm dos seus clientes o que se traduz na sua continua atualização para que se consigam adaptar aos requisitos os seus clientes lhes vão colocando.

Aplicando este conceito a este projeto, o produto final pretende oferecer uma solução mais simples, intuitiva e segura ao utilizador final de forma a não perder as funcionalidades anteriormente implementadas e oferecendo outras para melhorar a experiência do utilizador. Pode-se então concluir que o principal cliente é a i2S, pois são os recursos da organização que vão usufruir da implementação desta nova solução

de IAM. Deste modo, os produtos da i2S ganham valor e é expectável que os clientes sintam este aumento no valor do produto que lhes é oferecido pela organização.

2.4.3 Valor percecionado

O valor percecionado consiste no resultado da diferença entre os benefícios e os sacrifícios para a organização. Na Tabela 2, é possível visualizar a relação dos benefícios e os sacrifícios existentes no projeto, relativamente ao produto da organização, à relação que terá com os restantes colaboradores ou utilizadores que utilizaram esta solução.

Tabela 2 – Relação Sacrifício/Beneficio

Propósito / Domínio	**Produto**	**Relação**
Benefício	• Serviço de autenticação externo; • Implementação de uma solução capaz de satisfazer os requisitos atuais.	• Utilização intuitiva na gestão dos utilizadores; • Aumento de segurança para as restantes aplicações
Sacrifício	• Custo de integração com o sistema atual.	• Formação de utilizadores para configurar a *framework* de IAM.

Deste modo, pode-se concluir que o valor percecionado deste projeto é elevado uma vez é uma mais-valia para o produto final da organização. As mais valias consistem na implementação de uma solução capaz de satisfazer todos os requisitos avaliados e pretendidos pela organização, a implementação de uma *framework* externa que tenha um serviço de autenticação, a utilização desta seja mais intuitiva para quem faz a gestão dos utilizadores e o aumento de segurança para os restantes produtos e serviços disponibilizados pela organização. Por outro lado, é possível considerar alguns fatores negativos no desenvolvimento deste projeto, como por exemplo o custo da integração da *framework* IAM com o sistema existente e a formação de utilizadores para a utilização desta *framework*.

2.4.4 Proposta de valor

A proposta de valor pretende apresentar os motivos pelos quais o cliente deverá adquirir o produto final e quais os benefícios que este lhe trará.

No âmbito deste projeto, a proposta de valor consiste na implementação de uma *framework* externa, onde seja possível gerir as autorizações dos utilizadores que pretendem aceder aos recursos e serviços disponibilizados pela organização. Deste

modo, a organização terá os seus produtos mais robustos e protegidos a acessos indevidos.

Esta solução, para além do melhoramento relativo à segurança, irá melhorar a usabilidade das funcionalidades de gestão de utilizadores relativamente à solução atualmente existente, como por exemplo a gestão dos grupos de um utilizador e das permissões de cada utilizador.

2.4.5 Modelo de Negócio Canvas

A utilização do modelo de negócio Canvas, tem como objetivo auxiliar o negócio a perceber como é que os diversos aspetos relativos à venda de um produto ou serviço que se interligam.

O modelo de negócio Canvas é constituído pelos seguintes tópicos:

- Parceiros
    - Os parceiros chave seriam as empresas que desenvolvem *frameworks* de IAM e *software* para a geração e validação de *tokens*;
- Atividades Chave;
    - As atividades chave são a análise das *frameworks* de IAM existentes para perceber as suas funcionalidades e como se poderá integrar estas no sistema da organização e o desenvolvimento para a sua integração.
- Recursos Chave;
    - Os recursos chave são a documentação existente das *frameworks* de IAM, o conhecimento da arquitetura do sistema de autenticação e de autorização atualmente utilizado e conhecimento tecnológico para fazer a integração da *framework* IAM.
- Proposta de Valor;
    - A proposta de valor consiste na implementação de uma *framework* externa, onde seja possível gerir as autorizações dos utilizadores que pretendem aceder aos recursos e serviços disponibilizados pela organização
- Relação com os Clientes;
    - Sendo os clientes os desenvolvedores e utilizadores das aplicações da organização, espera-se que a comunicação seja feita através de assistência técnica quando necessário.
- Canais;

    - O canal de venda é a própria venda dos produtos da organização, uma vez que as funcionalidades de autenticação e autorização estão embutidas nas aplicações.

- Segmentos de Clientes;
    - Os clientes esperados são os utilizadores das aplicações que se autenticam, os gestores de sistema que gerem os utilizadores e as suas permissões e os desenvolvedores da organização.
- Custos;
    - Os custos existentes são relativos a recursos humanos para possíveis desenvolvimentos e uma possível compra de uma licença para a utilização de uma *framework* de IAM.
- Fontes de Retorno.
    - O retorno produzido pelos desenvolvimentos deste projeto é indireto, uma vez que apenas acrescenta valor aos produtos disponibilizados pela organização.

2.4.6 Analytic Hierarchy Process

O modelo Analytic Hierarchy Process (AHP) foi desenvolvido por Thomas L. Saaty na década de 1970. Desde então tem sido utilizado no auxílio para a tomada de decisões para cenários complexos.

O modelo AHP decompõe um problema numa hierarquia de critérios para se atingir um objetivo através da comparação entre cada um dos critérios. O método transforma estas comparações em números que são processados e comparados. São também definidos pesos para cada fator para, no fim, se calcular a probabilidade numérica de cada alternativa. Quanto maior a probabilidade, maior são as chances de a alternativa cumprir os requisitos e atingir o objetivo [12]. A aplicação deste modelo é feita na secção de avaliação das abordagens.

2.5 Sumário

Neste capítulo foi apresentado o tema de IAM e qual o objetivo destas *frameworks* e que problemas tentam resolver. Foi também apresentada a arquitetura mais comum de um sistema IAM e o trabalho realizado por cada um dos seus componentes. Foi também feita uma contextualização do problema a resolver e o que se pretende com este projeto. Por fim, foi apresentada a análise de valor deste projeto para que se pudesse perceber o real valor que trará ao cliente e à organização.

No início foi proposto que fossem respondidas às seguintes questões:

1. O que são *frameworks* IAM?

R: As *frameworks* IAM são sistemas que têm como objetivo assegurar a segurança dos recursos e serviços de uma organização. Têm também a possibilidade de fazer a gestão dos utilizadores que têm acesso a estes recursos e fazer a atribuição de permissões a estes utilizadores.

2. Como é a arquitetura de um sistema IAM?

R: A arquitetura de um sistema IAM pode ser composta por vários componentes. A arquitetura normal consiste numa base de dados de identidades, um gestor de credenciais, um servidor de autenticação, um núcleo de credenciais, um gestor de recursos, um motor de políticas de acesso e um motor de confiança. Um sistema IAM não precisa de conter, obrigatoriamente, todos estes componentes, desde que consiga realizar as mesmas tarefas respondidas na questão anterior.

3. Que mais valia traz um sistema IAM a uma empresa?

R: Um sistema IAM traz mais segurança e simplicidade a uma organização. Segurança, pois, consegue restringir o acesso aos recursos e serviços disponibilizados pela organização e simplicidade pois todas as funcionalidades para a gestão dos utilizadores estão centralizadas numa única plataforma.

4. Qual a mais valia do trabalho apresentado neste documento para a implementação de uma *framework* IAM?

R: A análise de valor realizada concluiu que este projeto trará muito valor à organização pois consegue resolver os problemas descritos anteriormente, apesar do custo de trabalho necessário realizar para a integração da *framework* IAM com o sistema existente.

3 Estado da arte

Após uma análise pormenorizada do problema, é necessário aprofundar o conhecimento relativamente à área em específico e perceber os padrões utilizados e as soluções existentes no mercado.

Neste capítulo serão apresentados algumas *frameworks* de IAM que se destacam no mercado. As *frameworks* foram selecionadas de acordo com a Gartner que é uma empresa de pesquisa e consultoria que fornece informações, conselhos e ferramentas para a tomada de decisões. Serão descritas algumas das suas funcionalidades e alguns pontos positivos e negativos destas *frameworks*. No final do capítulo serão descritas algumas tecnologias relevantes que foram utilizadas para o desenvolvimento deste projeto.

Ao longo deste capítulo será dada a resposta às seguintes questões:

1. Existem *frameworks* eficazes para a gestão de acessos?
2. Quais as funcionalidades mais e menos trabalhadas nas soluções existentes no mercado?

3.1 Frameworks existentes

Nesta secção serão abordadas algumas *frameworks* de IAM existentes no mercado. Este estudo tem como objetivo principal identificar os seus pontos positivos e negativos baseados nas suas funcionalidades.

A seleção das *frameworks* para se elaborar uma análise, foi baseada na análise feita pela conceituada empresa Gartner. Na Figura 5 é possível ver o Magic Quadrant que apresenta o grau de visão da empresa em termos de inovação tecnológica e de abrangência sobre as necessidades do mercado e grau da habilidade de executar e implementar o que prometem.

Serão então apresentadas as soluções das empresas que, de acordo com a Figura 4, são classificadas como líderes no tema de gestão de acessos, como:

- Auth0;
- ForgeRock;
- IBM;
- Microsoft;
- Okta;

- OneLogin;
- Ping Identity.

Figura 4 – Gartner *Magic Quadrant* para *Access Management* [13]

Será também apresentada uma outra *framework,* o Keycloak, que é utilizada pela empresa Cleva que é um dos parceiros da i2S.

3.1.1 Auth0

A Auth0, apesar de na Figura 5 ser considerado como desafiador, está perto de ser considerada como líder, segundo a Gartner [13]. Apesar de ser conhecida pelos seus produtos focados para desenvolvedores, vende alguns produtos de IAM. A plataforma fornecida pela OAuth0 consiste em três módulos: autenticação, gestão de acessos e gestão de utilizadores.

O módulo de autenticação permite autenticar os utilizadores nas aplicações configuradas. A autenticação é feita através de servidores de provedores de identidade que utilizem protocolos como OpenID Connect, SAML, WS-Federation, entre outras. Oferece também SSO para que o utilizador apenas tenha de se autenticar uma única vez e navegar por todas os recursos [14].

O módulo de gestão de acessos permite que ocorra uma validação para avaliar se o utilizador tem permissão para aceder ao recurso que esta a tentar aceder. Esta autorização é derterminada por meio do uso de políticas e regras, baseadas em *Role-based Access Control* (RBAC) [15].

O módulo de gestão de utilizadores permite armazenar os perfis dos utilizadores, utilizando fontes externas como provedores de identidades e fazer a gestão destes perfis. Permite alterar ou adicionar novos dados ao perfil do utilizador como a cor favorita, número de telefone e adicionar ou remover permissões [16].

Na Tabela 3 são apresentados os pontos fortes e pontos fracos da plataforma fornecida pela OAuth0, segundo a Gartner.

Tabela 3 – Avaliação da OAuth0 feita pela Gartner [13]

Pontos fortes	• Tem um forte foco e especialização em *Custom Identity and Access Management* (CIAM); • Fornece recursos para o controlo de acessos e incorpora um conjunto de recursos de segurança para as APIs; • Boa capacidade de gerir regras e fornece a habilidade de criar fluxos complexos de autenticação e autorização; • Tem preços competitivos, com quase todos os produtos a estarem abaixo da média do mercado.
Pontos fracos	• Não fornece uma tecnologia para integrar aplicações que não utilizem protocolos de autenticação e autorização; • Fornece um catálogo limitado de integrações de aplicações *Software as a Service* (SaaS), o que exige às organizações que configurem mais as suas próprias integrações; • Funcionalidade de geração de relatórios são limitadas.

3.1.2 ForgeRock

A ForgeRock, segundo a Gartner, é um líder no mercado das plataformas de IAM. Esta organização oferece uma plataforma de identidade que inclui módulos como de acesso inteligente, autorização, SSO, federação, Social Sign-On e gestão de sessões [17].

A solução desta organização, oferece uma plataforma que permite a gestão de acessos a recursos que estão configurados. Também fornece uma infraestrutura para gerir os utilizadores, as suas funções e os seus acessos aos recursos.

A autenticação fornecida utiliza uma variedade de módulos que se conectam aos repositórios de identidade que armazenam os utilizadores e fornecem serviços de autenticação. Estes repositórios podem ser implementados como diretórios Lightweight Directory Access Protocol (LDAP), bases de dados relacionais, Remote Authentication Dial In User Service (RADIUS), autenticação do Windows, serviços de senha única e outros serviços de gestão de acessos [18].

A autorização fornecida, permite a gestão de políticas de acesso para os recursos configurados. Esta solução permite instalar um agente na aplicação *web* para solicitar as políticas de decisões à plataforma. Deste modo, no desenvolvimento das aplicações é possível incorporar estas políticas e se houver algum problema ou alguma alteração destas políticas de acesso, não é preciso atualizar a aplicação pois basta alterar a definição da política na plataforma [18].

Na Tabela 4 são apresentados os pontos fortes e pontos fracos da plataforma fornecida pela ForgeRock, segundo a Gartner.

Tabela 4 – Avaliação da ForgeRock feita pela Gartner [13]

Pontos fortes	• Tem se envolvido nos esforços de padronização para novos protocolos, incluindo UMA e OAuth; • Tinha produtos de CIAM e *Open Banking* disponíveis como SaaS e no último ano lançou opções baseadas em SaaS para os seus produtos.
Pontos fracos	• As opções fornecidas como serviços são recentes o que necessita que potenciais compradores devem avaliar atentamente estes serviços e aproveitar os *Proof of Concept* (POC) para validar se o produto cumpre com as suas expectativas; • As funcionalidades de *User and Entity Behavior Analytics* (UEBA) são recentes e não estão tão trabalhadas como as restantes do mercado; • Os preços dos produtos são desiguais onde os preços para *software* são abaixo da média e os produtos de SaaS e CIAM são acima da média.

3.1.3 IBM

A IBM, apesar de ser considerada desafiadora pela Gartner está perto de ser considerada uma organização líder nos produtos de gestão de acessos, segundo a Figura 5. A IBM tem dois produtos de IAM, o IBM Security Verify Access que é fornecido como *software* e o IBM Security Verify que é fornecido como SaaS.

Ambas as soluções oferecem funcionalidades idênticas como SSO para o utilizador apenas se autenticar uma vez, Multi-factor Authentication (MFA) que oferece uma maior segurança na autenticação e UEBA para análise de estatísticas dos utilizadores. Oferece também funcionalidades de gestão de utilizadores e grupos e permite utilizar provedores de identidade externos para fazer a autenticação dos utilizadores como Active Directory ou Google. Por fim, oferece uma plataforma para os desenvolvedores criarem fluxos de autenticação e incorporarem as funcionalidades de MFA nas suas aplicações [19].

Na Tabela 5 são apresentados os pontos fortes e pontos fracos da plataforma fornecida pela IBM, segundo a Gartner.

Tabela 5 – Avaliação da IBM feita pela Gartner [13]

Pontos fortes	• Oferece bons recursos de *Life Cycle Management* (LCM) de identidade integrada no produto IBM *Security Verify,* mas em casos de uso mais complexos é-lhes oferecido, sem custos extra, um conjunto de recursos *Identity Governance and Administration* (IGA) pelo produto IBM *Identity Governance and Intelligence*; • Oferece recursos de proteção da API das aplicações acima da média; • O preço do produto IBM SaaS supera os restantes concorrentes se for feita uma análise relação qualidade preço;
Pontos fracos	• No último ano não foram feitas grandes melhorias a estes produtos; • O suporte Business to Business (B2B) e Business to Customer (B2C) é limitado.

3.1.4 Microsoft

A Microsoft é considerada líder nos produtos de gestão de acessos, pela Gartner, e oferece o produto Microsoft Azure Active Directory como SaaS.

Este produto oferece SSO para um único início de sessão, MFA e acesso condicional baseado em políticas de autenticação para proteger e gerir os acessos às aplicações. Todas estas funcionalidades estão centralizadas numa única plataforma [20].

A Microsoft oferece também ferramentas aos desenvolvedores para integrarem as suas aplicações com o Azure Active Directory como o Microsoft Graph. Esta solução também suporta normas como o OAuth 2.0, SAML e SCIM para uma mais fácil integração das aplicações dos clientes [21].

Na Tabela 6 são apresentados os pontos fortes e pontos fracos da plataforma fornecida pela Microsoft, segundo a Gartner.

Tabela 6 – Avaliação da Microsoft feita pela Gartner [13]

Pontos fortes	• A ferramenta de acesso condicional tem uma funcionalidade de auditoria; • O preço da oferta B2C foi simplificado fazendo com que o preço geral do produto esteja abaixo da média do mercado;
Pontos fracos	• O licenciamento deste produto não permite que os módulos sejam vendidos individualmente; • Falta de experiência em CIAM relativamente aos outros líderes do mercado; • Funcionalidade de gestão de sessões é menos madura que as restantes no mercado; • As interfaces programáticas são limitadas à API do Microsoft Graph.

3.1.5 Okta

A Okta é considerada líder nos produtos de gestão de acessos, segundo a Gartner. Os produtos oferecidos, como SaaS, são Workforce Identity, Customer Identity e Plataform Services.

O Workforce Identity oferece SSO para um único *login* de sessão, MFA para melhor proteção das aplicações e um diretório universal para a gestão dos utilizadores, grupos e aplicações. Permite também a gestão do ciclo de vida dos processos de negócio para utilizadores externos e internos, uma API para gestão de acessos, gestão de acesso Zero Trust para a infraestrutura. Por fim, oferece um *gateway* de acesso para proteger as aplicações sem alterar o seu comportamento.

O Customer Identity oferece *login* seguro e SSO para as aplicações, gestão de utilizadores e de clientes através do uso de uma API ou da consola de administração

fornecida. Oferece MFA para aumentar a segurança das aplicações, gestão dos fluxos de trabalho, integração com diretórios corporativos ou provedores de identidade e um *gateway* de acesso sem ter de alterar código.

O Plataform Services oferece automatização de processos de identificação utilizando lógica condicional, um módulo para gestão de acessos, para gestão das aplicações. Oferece um módulo para o armazenamento das informações dos utilizadores, ferramentas para facilitar a integração das aplicações para utilizarem a plataforma e dados estatísticos.

Na Tabela 7 são apresentados os pontos fortes e pontos fracos da plataforma fornecida pela Okta, segundo a Gartner.

Tabela 7 – Avaliação da Okta feita pela Gartner [13]

Pontos fortes	• Melhorou as funcionalidades de UEBA através de um servidor que fornece dados analíticos através dos acessos a partir da utilização de padrões; • Adição de novos recursos de fluxos de trabalho permitem criar cenários complexos de autenticação e autorização sem alterar as aplicações;
Pontos fracos	• A integração de diretórios externos são limitados relativamente aos outros fornecedores; • As funcionalidades de CIAM são básicas; • Os preços dos produtos são bem acima da média.

3.1.6 OneLogin

A OneLogin é considerada pela Gartner uma organização líder nos produtos de gestão de acesso. O produto que oferece é uma solução de IAM fornecida por SaaS. Os benefícios da solução da OneLogin são o reforço da segurança, redução dos custos de infraestrutura de identidade, aumento da produtividade facilitando a integração das aplicações e a capacidade de os utilizadores começarem a utilizar rapidamente as aplicações integradas utilizando políticas de acesso definidas [22].

O produto de IAM da OneLogin tem cinco funcionalidades que se destacam. SSO para os utilizadores só necessitarem de se autenticarem uma única vez, a automatização dos ciclos de vida dos utilizadores, ou seja, as alterações feitas nos diretórios ou provedores de identidade são automaticamente sincronizadas, MFA para uma maior proteção dos dados das organizações, a utilização de um diretório ou provedor de identidade externo como Active Directory ou Google para fazer a autenticação e armazenamento dos utilizadores e a geração de relatórios que fornecem informações

relativas à atividade de *login,* utilização das aplicações, utilizadores inativos, entre outras [22].

Na Tabela 8 são apresentados os pontos fortes e pontos fracos da plataforma fornecida pela OneLogin, segundo a Gartner.

Tabela 8 – Avaliação da OneLogin feita pela Gartner [13]

Pontos fortes	• Tem melhorado significativamente as ferramentas para os desenvolvedores utilizarem nas suas aplicações para acederem, programaticamente, à plataforma da OneLogin; • O preço do produto é competitivo e está alinhado com a média dos restantes produtos de IAM no mercado;
Pontos fracos	• As funcionalidades de proteção das API ainda não são maduras relativamente às do mercado em geral; • As funcionalidades de CIAM necessitam de trabalho extra com as interfaces da API; • As funcionalidades de *Bring Your Own Identity* (BYOI) não estão tão trabalhadas como as restantes do mercado.

3.1.7 Ping Identity

A Ping Identity é considerada pela Gartner a empresa líder no que toca a produtos de gestão de acessos. Os serviços SaaS que fornece incluem PingID, PingIntelligence para APIs, Ping One para Enterprise e Ping One para Clientes.

O produto fornecido tem funcionalidade de SSO para uma única autenticação, permite a gestão das identidades e aplicar políticas de acesso, utilização de padrões de federação como SAML, WS-Federation, OAuth, OpenID Connect, entre outros, a vinculação de contas do Facebook, Google, entre outros e MFA para maior proteção das aplicações dos clientes. Permite também a integração das aplicações em servidores aplicacionais como IBM WebSphere, Oracle Weblogic, Apache Tomcat, entre outos, a utilização de sistemas de autenticação externos como Active Directory, LDAP, Azure AD, entre outros e a utilização do Servidor Microsoft IIS e Apache HTTP [23].

Na Tabela 9 são apresentados os pontos fortes e pontos fracos da plataforma fornecida pela Ping Identity, segundo a Gartner.

Tabela 9 – Avaliação da Ping Identity feita pela Gartner [13]

Pontos fortes	• A integração do produto é flexível e a implementação e administração é fácil; • É líder em integração de aplicações que não utilizam padrões; • Introdução de um recuso de auto-atendimento para desenvolvedores e proprietários de aplicações;
Pontos fracos	• Faltam recursos do ciclo de vida das identidades; • O preço é variável dependendo do cenário de implementação necessário, o que pode levar ao preço ser acima ou abaixo da média do mercado.

3.1.8 Keycloak

O Keycloak apesar de não ter sido avaliado pela Gartner, é utilizada pela empresa Cleva que é um dos parceiros da i2S. O Keycloak é uma solução *open-source* de IAM e tem como objetivo proteger as aplicações e os seus serviços com pouca ou nenhuma alteração nas aplicações.

Esta plataforma fornece SSO às aplicações para que os utilizadores não precisem de fazer *login* mais do que uma vez para acederem a uma aplicação diferente e o mesmo se aplica ao *logout* o que significa que só precisa de fazer *logoff* uma vez e é desconectado de todas as aplicações que utilizam a plataforma. Fornece uma consola para fazer a gestão de vários recursos como utilizadores, aplicações clientes, permissões, entre outras. O Keycloak permite também a utilização de provedores de identidade externos que utilizem os padrões OpenID Connect ou SAML 2.0 para fazer a autenticação ou permite a integração para se conectar com servidores LDAP ou Active Directory para autenticação e sincronização de utilizadores. Fornece também adaptadores para as aplicações clientes para a integração das aplicações com o Keycloak ou, se não houver a necessidade de fazer alterações às aplicações, fornece uma API REST para ser utilizada como *proxy* para as proteger [24].

Na Tabela 10 são apresentados os pontos fortes e pontos fracos da plataforma Keycloak. Apesar da Gartner não ter efetuado uma avaliação sobre esta plataforma, estes pontos serão efetuados com base nos restantes produtos avaliados.

Tabela 10 – Avaliação do Keycloak

Pontos fortes	• É uma plataforma *open-source,* ou seja, não terá custos monetários para os clientes; • Tem tido atualizações constantes e correção de problemas;
Pontos fracos	• Não tem funcionalidades de UEBA; • Não é compatível com todas as versões dos servidores aplicacionais; • Sendo uma plataforma *open-source* a probabilidade de serem encontrados problemas é maior.

3.1.9 Avaliação das *frameworks*

Para a avaliação das *frameworks* analisadas, decidiu-se fazer uma lista das funcionalidades mais relevantes que um sistema IAM deveria ter e avaliar cada *framework* tendo em conta a sua maturidade para essa funcionalidade. A avaliação foi feita tendo em conta a análise disponibilizada pela Gartner e a informação existente na documentação de cada *framework*.

As funcionalidades a serem utilizadas para efetuar a avaliação são as seguintes:

- Gestão de utilizadores;
- Gestão de autorizações;
- Flexibilidade no protocolo de segurança;
- Análise de dados analíticos.

Os níveis utilizados para a avaliação são:

- 0 → Não tem a funcionalidade;
- 1 → Funcionalidade recente ou pouco desenvolvida;
- 2 → Funcionalidade relativamente desenvolvida;
- 3 → Funcionalidade bastante desenvolvida.

Analisando a Tabela 11, pode-se verificar que as *frameworks* fornecidas pela Okta e pela Ping Identity são as que fornecem melhor suporte às principais funcionalidades dos sistemas IAM. Sendo assim, estas *frameworks* serão tidas em conta quando for feita a implementação.

Tabela 11 – Avaliação das *frameworks*

	Gestão de utilizadores	Gestão de autorizações	Flexibilidade no protocolo de segurança	Análise de dados analíticos	**Total**
Auth0	2	2	3	1	8
ForgeRock	2	1	3	1	7
IBM	2	2	2	2	8
Microsoft	2	2	2	2	8
Okta	2	3	2	3	10
OneLogin	3	1	2	1	7
Ping Identity	3	1	3	2	9
Keycloak	2	2	2	0	6

3.2 Tecnologia relevante

Nesta secção são apresentadas algumas tecnologias que estão relacionadas com a implementação deste projeto e que estão em uso na i2S. As tecnologias a apresentar são:

- Wildfly;
- WebSphere Application Server;
- Spring Framework;
- Apache CXF;
- Json Web Token;
- OAuth 2.0;
- Security Assertion Markup Language;
- OpenID Connect.

Nestas tecnologias há os servidores aplicacionais, *software* utilizado pela i2S e padrões e protocolos de segurança considerados relevantes para se fazer um estudo. Estas tecnologias e metodologias serão apresentadas por serem relevantes para o desenvolvimento do projeto.

3.2.1 Wildfly

O Wildfly é uma solução que permite implementações de plataformas de *Java Enterprise Edition*. É um projeto licenciado como *Open Source* e apoiado pela comunidade *JBoss* mas existe uma versão comercial, *JBoss Enterprise Application Platform* que é licenciada e suportada pela Red Hat.

A estrutura do Wildlfy tem vindo a mudar nos últimos anos devido aos requisitos de manutenção e de desempenho. No entanto, o Wildfly é composto pelo seu núcleo e por um conjunto de subsistemas. Estes subsistemas são responsáveis por uma funcionalidade especifica e são definidos como serviços *on-demand,* que são chamados por aplicações instaladas no servidor.

O núcleo do servidor, consiste em duas partes:

- Módulos *JBoss*;
- *Modular Service Container* (MSC).

Os módulos do *JBoss* são responsáveis pelo carregamento dos *class loaders* dos recursos no servidor e desta forma fornece uma maneira eficaz de fazer este carregamento para o *class path*.

O MSC fornece uma maneira de instalar, desinstalar e gerir os serviços utilizados pelo servidor, permitindo assim a injeção de recursos nos serviços e fazendo a gestão das dependências entre serviços [25].

A Figura 5, apresenta a arquitetura básica do *core* do Wildfly.

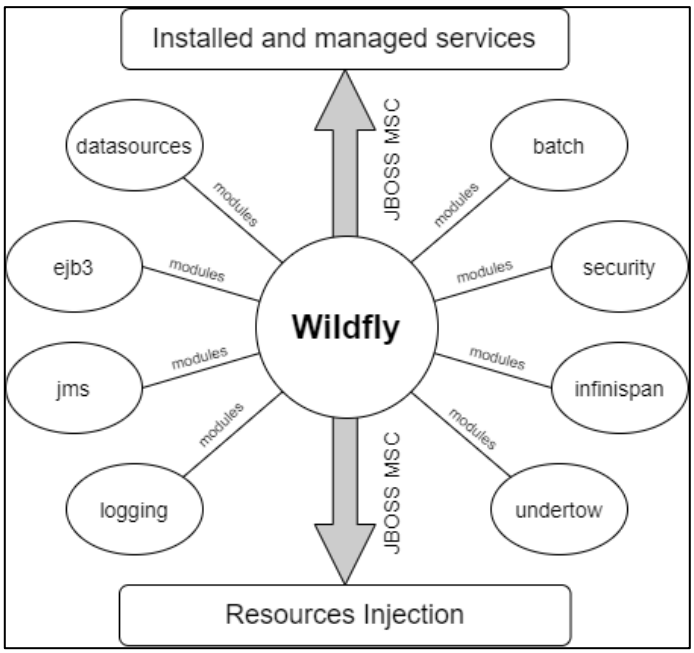


Figura 5 – Arquitetura do *core* do Wildfly (adaptado de [25])

Os módulos que o Wildly suporta são as *datasources* que são as conexões às bases de dados, *ejb3* que são os *Enterprise Java Beans,* os módulos de JMS e os de *logging* para fazer as configurações para a escrita *log*. Também permite a configurações de processos *batch,* configurações de segurança como conexões de LDAP, e módulos de *infinispan* e *undertow*. O Wildlfy permite também a injeção de recursos e a instalação e gestão de serviços como aplicações dos clientes.

3.2.2 WebSphere Application Server

O WebSphere Application Server é um servidor aplicacional que permite acelerar a entrega de aplicações com um ambiente em tempo de execução baseado em Java Enterprise Edition.

Este servidor aplicacional oferece suporte a modelos de programação baseados em padrões para ajudar os utilizadores a modernizar as suas soluções, permitindo obter maior visibilidade entre cargas de trabalho e analisando as aplicações corporativas [26].

O WebSphere Application Server é composto por vários componentes apresentados na Figura 6.

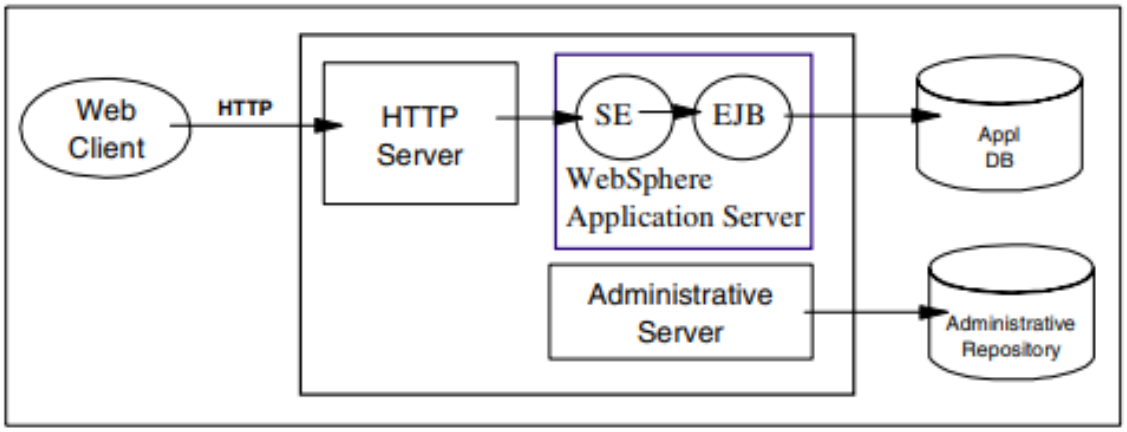


Figura 6 – Componentes do WebSphere Application Server [27]

O servidor administrativo, *Administative Server*, fornece, tal como o nome indica, as funções administrativas para todos os nós que executam no domínio do WebSphere. Este é responsável por manter a configuração de um domínio WebSphere, fornecendo serviços de segurança, gestão de carga de trabalho e monitorização do estado de execução.

O repositório administrativo, *Administrative Repository*, armazena a configuração de todos os componentes do servidor aplicacional dentro do domínio do WebSphere. Cada servidor administrativo comunica as mudanças na configuração para todos os servidores administrativos através deste componente. Este repositório consiste numa série de tabelas que são persistidas como *Entity Enterprise Java Beans* do WebSphere.

Um servidor aplicacional do WebSphere é o processo utilizado para executar as aplicações. Cada WebSphere Application Server é executado na sua própria *Java Virtual Machine* (JVM). A capacidade de criar vários processos do servidor aplicacional em tempo de execução do WebSphere fornece um meio par aumentar o rendimento através de uma escalabilidade vertical. Este tipo de clonagem, também pode ser empregada no WebSphere para suportar escalabilidade horizontal onde vários processos são distribuídos em várias máquinas físicas.

A consola administrativa fornece uma janela gráfica fácil de utilizar para gerir um domínio/*cluster* do WebSphere. Esta consola, liga-se a um dos servidores administrativos num *cluster* e pode ser utilizado para alterar a configuração ou o estado de execução em qualquer máquina no domínio [27].

3.2.3 Spring Framework

A Spring Framework é uma plataforma de Java que fornece suporte de infraestrutura para o desenvolvimento de aplicações Java. Este *software* lida com a infraestrutura de modo que o utilizador se posso concentrar no desenvolvimento da sua aplicação.

O Spring permite que as aplicações sejam criadas a partir de objetos Java simples e aplicar serviços de forma não invasiva a estes objetos. Esta capacidade aplica-se ao modelo de programação Java SE e ao Java EE [28].

Alguns exemplos em que a plataforma possa ser utilizada de forma vantajosa são fazendo com que:

- Um método Java, ao executar uma transação numa base de dados não tem de se preocupar em lidar com APIs de transação;
- Um método Java local seja transformado num procedimento remoto sem ter de lidar com APIs remotas;
- Um método Java local seja transformado numa operação de gestão sem ter de lidar com APIs Java Management Extensions (JMX);
- Um método Java local seja transformado num manipulador de mensagens sem ter de lidar com APIs Java Message Service (JMS).

A Spring Framework tem recursos organizados em cerca de vinte módulos que são agrupados em Core Container, Data Access/Integration, Web, Aspect Oriented Programming (AOP), Instrumentation e Test. Estes componentes podem ser visualizados na Figura 7.

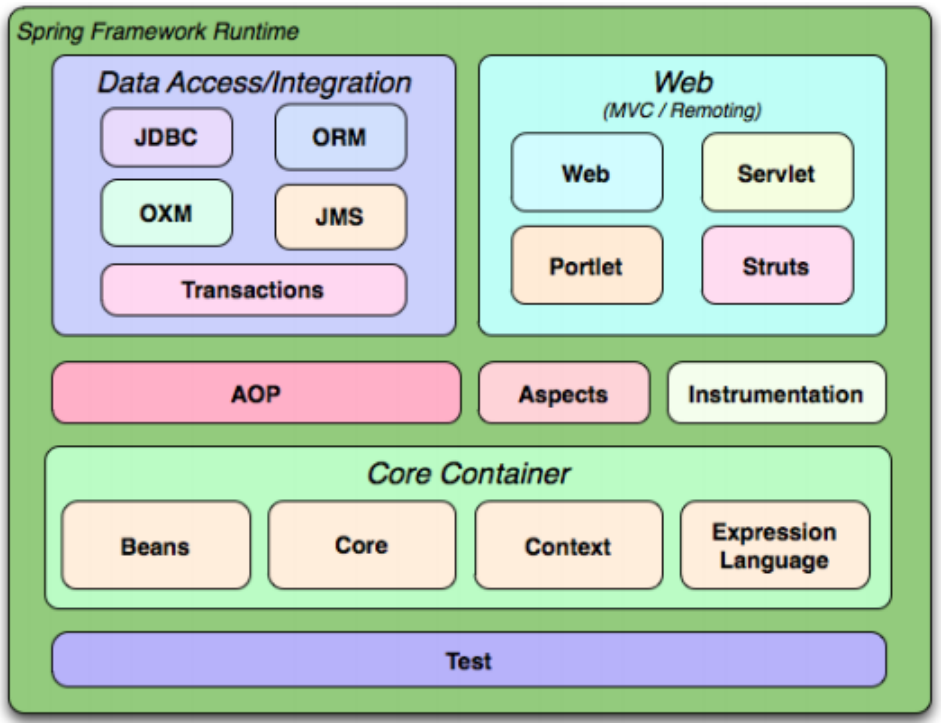


Figura 7 – Visão geral da Spring Framework [28]

3.2.4 Apache Cxf

O projeto Apache Cxf é uma estrutura modular e extensível escrita em Java. Esta implementa várias especificações de serviços *web*, fornece ferramentas para facilitar a programação de uma aplicação de *front-end* e oferece suporte a muitos protocolos e ligações de dados. Cada módulo pode ser substituído, podendo-se criar aplicações que comunicam através de protocolos compatíveis com a *framework* como HTTP, JMS ou *Common Object Request Broker Architecture* (CORBA) [29].

Na Figura 8 pode-se ver os módulos da arquitetura do Cxf. O *Bus* fornece os recursos partilhados em tempo de execução do Cxf. Os *Front-ends* fornecem um modelo de programação para interagir com o Cxf através das APIs Java API for XML Web Services (JAX-WS), Java API for RESTful Web Services (JAX-RS), Simple e Javascript. O módulo de *Messaging and Interceptors* é a principal camada do Cxf. Estes *interceptores* são a unidade fundamental do Cxf pois permite dividir como as mensagens são processadas e enviadas. O *Service Model* é a representação de um serviço dentro da arquitetura Cxf. Os *Protocol Bindings* fornecem maneiras de mapear formatos e protocolos na camada de transporte. Por fim, os *Transports* é a camada de transporte dos dados, por exemplo HTTP, HTTPs, HTTP-Jetty, HTTP-OSGI, Servlet, local, JMS, In-VM [30].

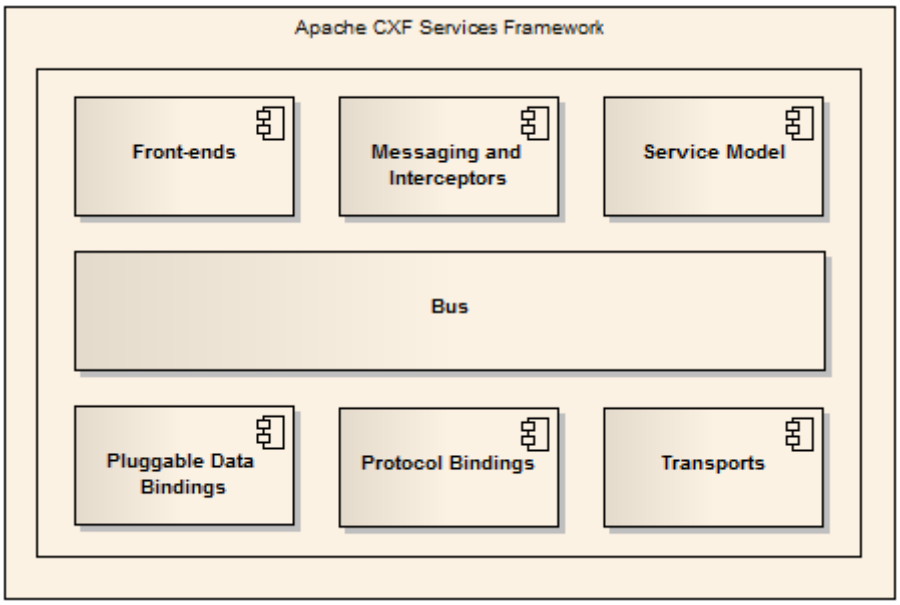


Figura 8 – Módulos da arquitetura do CXF [30]

3.2.5 Json Web Token

Um Json Web Token (JWT) é um meio de representar intenções a serem transferidas entre duas partes. Estas intenções são codificadas num objeto JSON que é assinado digitalmente utilizando uma JSON Web Signature (JWS) e, opcionalmente, encriptada utilizando uma JSON Web Encryption. Uma JWS é um meio de representar conteúdo assinado utilizando uma estrutura de dados JSON. Diretamente relacionado, existe a JWE que é um meio de representar conteúdo encriptado utilizando estrutura de dados JSON.

Os JWT têm sido utilizados nas *frameworks* de OAuth para conceder autorizações a um utilizador na utilização de um serviço de terceiros. Isto permite que estas aplicações tenham acesso aos recursos dos utilizadores no serviço.

Um JWT está dividido em três partes. O cabeçalho que contém o conteúdo de identificação como o algoritmo de assinatura e a identificação da chave utilizada. O conteúdo onde é descrita a informação do essencial do utilizador, das suas autorizações e da data de expiração do *token*. Por fim, a assinatura ou código de autenticação codificado em Base64 do cabeçalho e conteúdo. Cada uma destas partes está codificada em Base64 separadas por "." [31].

3.2.6 OAuth 2.0

O OAuth é uma camada de autorização que se encontra nas aplicações. No OAuth, o cliente solicita acesso aos recursos controlados por uma aplicação e é emitido um conjunto de credenciais fornecidas pelo servidor de autorização com a aprovação da

aplicação a ser acedida. Esta credencial obtida é um *token* de acesso que permite ao utilizador aceder aos recursos protegidos [32].

Um servidor de autorizações é responsável por identificar e autenticar as aplicações clientes e os utilizadores, gerir as permissões dos utilizadores, emitir os *tokens* de acesso e é responsável pelo SSO.

Existem dois tipos de aplicações clientes, os confidenciais e os públicos. Os públicos são clientes onde os seus recursos estão expostos a qualquer utilizador sem que lhe tenha de ser pedido qualquer credencial. Os clientes confidencias têm os seus serviços protegidos e só com credenciais válidas fornecidas pelo utilizador é que este os poderá aceder.

Um dos tipos de credenciais de *tokens* de acesso mais utilizado são os *Bearer token*. Estes permitem aceder aos recursos protegidos através do cabeçalho *Authorization* dos pedidos. Este *token* representa uma autorização emitida por um cliente através do servidor de autorização. Atualmente o formato padrão mais adotado pelas implementações de OAuth 2.0 é o JWT apesar deste padrão apenas especificar os critérios que o *Bearer token* deverá ter [33].

Na figura 9 é possível ver o fluxo de autenticação padrão do protocolo OAuth.

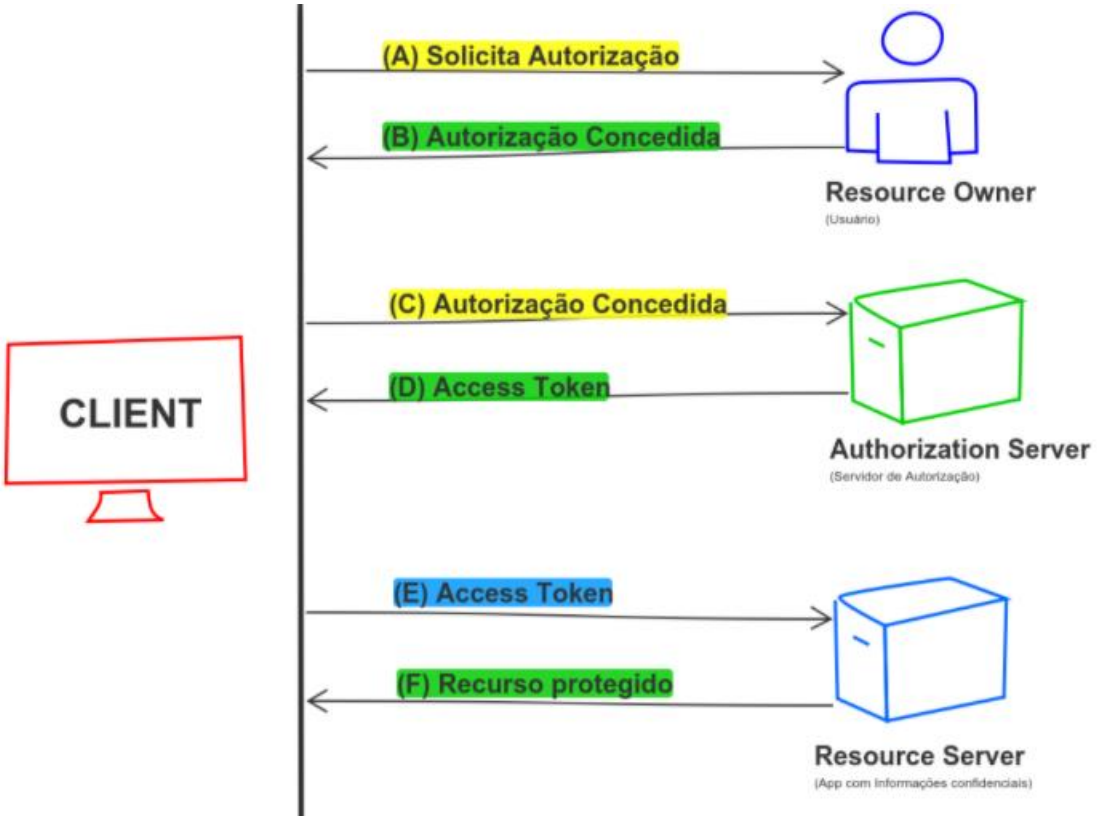


Figura 9 – Fluxo de autenticação padrão do protocolo OAuth [33]

Inicialmente, o utilizador acede a uma aplicação cliente para ter acesso a conteúdo protegido. Este solicita autorização à qual o utilizador fornece as suas credenciais,

como por exemplo o par nome de utilizador e *password*. O cliente solicita ao servidor de autorização um *token* de acesso através das credenciais fornecidas pelo utilizador. Este servidor valida a identidade do utilizador utilizando um Identity Server, por exemplo uma Windows Active Directory. Se a identidade for validada, o *token* de acesso é criado e devolvido para o cliente. Por fim, a aplicação cliente informa ao utilizador que o acesso lhe foi concedido e fornece o conteúdo protegido.

3.2.7 Security Assertion Markup Language

O SAML é um padrão utilizado para fazer a autenticação de utilizadores. Este padrão de SSO tem vantagens relativamente à autenticação que utiliza o nome do utilizador e a password pois não há a necessidade de escrever credenciais.

O SSO do SAML utiliza documentos Extensible Markup Language (XML) assinados digitalmente para transferir a identidade de um utilizador de um provedor de identidade para outro provedor de serviços [34].

Na Figura 10, é possível visualizar um possível fluxo deste padrão.

Figura 10 – Fluxo de SSO do padrão SAML [34]

Inicialmente o utilizador acede a uma aplicação e esta identifica o utilizador através do seu subdomínio ou do endereço de Internet Protocol (IP) ou outro e redireciona o utilizador para o provedor de identidade para solicitar a sua autenticação. De seguida, se o utilizador não tiver uma sessão ativa com o Identity Provider (IDP), efetua a sua autenticação. O provedor de identidade cria a resposta de autenticação na forma de um documento XML assinado utilizando um certificado X.509, que é um certificado digital que contem o mínimo de informação sobre a entidade certificada [35], e publica essa informação para o provedor de serviços. O provedor de serviços, recupera a resposta de autenticação efetuada pelo utilizador e valida a impressão digital do certificado. Se for válido, é dada autorização ao utilizador para aceder à aplicação [34].

3.2.8 OpenID Connect

O OpenID Connect é construído como uma camada de identificação em cima do protocolo OAuth 2.0. A funcionalidade que adiciona a este protocolo é a capacidade de verificar a identidade de um utilizador, confiando no processo de autenticação feito pelo

OpenID Provider. Ou seja, adiciona a funcionalidade de gestão de identidades ao sistema de OAuth 2.0 [36].

De maneira a verificar a identidade do utilizador, o OpenID Connect adiciona um novo tipo de token ao OAuth 2.0, chamado “id_token”. Este *token* pode conter a identidade do OpenID Provider que emitiu o *token*, o identificador do utilizador, a identidade do destinatário pretendido, a hora que foi emitido e o seu tempo de expiração. O *token* é do tipo JWT e é assinado digitalmente pelo OpenID Provider.

3.3 Sumário

Neste capítulo foram apresentadas várias plataformas IAM recomendadas pela Gartner. Pode-se concluir que existem várias soluções conseguem responder às necessidades impostas, apesar de nenhuma ser perfeita pois depende do cenário que se pretende implementar. Foram também apresentadas algumas tecnologias e padrões relevantes para o desenvolvimento deste projeto.

No início deste capítulo foi proposto que fossem respondidas às seguintes questões:

1. Existem *frameworks* eficazes para a gestão de acessos?

R: No estudo efetuado, foram encontradas várias soluções que conseguem responder às necessidades impostas, mas cada solução tem as suas especificidades e características para facilitar a integração da plataforma com as aplicações dos clientes. Pode se concluir que a escolha da *framework* a implementar depende muito dos requisitos e dos cenários que os clientes pretendem.

2. Quais as funcionalidades mais e menos trabalhadas nas soluções existentes no mercado?

R: No geral, quase todas as *framework* analisadas têm as mesmas funcionalidades, umas mais desenvolvidas e maduras que outras. As funcionalidades que se verifica que são priorizadas são as de gestão de utilizadores e as suas permissões. No entanto, uma funcionalidade que se verifica que ainda necessita de maior trabalho é a geração de relatórios de auditorias e estatísticas que fornecem dados às organizações que podem ser valiosos.

4 Análise

Após terem sido analisadas as soluções de IAM existentes no mercado e as tecnologias e padrões relevantes para a resolução do problema, é necessário definir que funcionalidades se pretende que a solução final tenha e de que maneira se pode resolver o problema.

Neste capítulo serão apresentados os requisitos funcionais e não funcionais que se pretende que a solução final concretize. Serão também descritos os casos de uso mais relevantes da solução. Por fim, serão apresentadas as duas abordagens que se poderão utilizar para resolver o problema e quais as vantagens e desvantagens do uso de cada uma delas.

Ao longo deste capítulo será dada a resposta à seguinte questão:

1. Quais as abordagens para a resolução do problema?

2. Qual a abordagem escolhida?

4.1 Requisitos

Nesta secção são identificados os requisitos funcionais e não funcionais e os respetivos casos de uso para a implementação e desenvolvimento desta solução.

Os requisitos serão classificados de acordo com o modelo FURPS. Este modelo classifica os requisitos em funcionais, usabilidade, confiabilidade, performance e suportabilidade. Os funcionais representam as principais funcionalidades que representam o produto. A usabilidade tem a ver com as características de estética e consistência na interface do utilizador. A confiabilidade preocupa-se com a disponibilidade da solução e a capacidade de recuperação de falhas. A performance preocupa-se com o rendimento, tempo de resposta e tempo de recuperação. A suportabilidade está relacionada com adaptabilidade, manutenção, compatibilidade, configurabilidade e escalabilidade da solução [37].

Na Tabela 12, são apresentados os requisitos funcionais.

Tabela 12 – Requisitos funcionais

Requisito
1.1. Registo de aplicações
1.2. Registo de perfis aplicacionais
1.3. Atribuição de perfis aplicacionais a utilizadores e ou grupos

Requisito
1.4. Gestão de *tokens*
1.5. Implementação do *Refresh Token*

Na Tabela 13 são apresentados os requisitos não funcionais.

Tabela 13 – Requisitos não funcionais

Requisito	Classificação FURPS
1.6. Interface simplificado e intuitivo apenas para a gestão dos utilizadores que não exponha a complexidade da *framework* e das suas restantes funcionalidades	Usabilidade
1.7. Implementação em *cluster* Ativo/Ativo	Confiabilidade
1.8. As tarefas de validação de autorização têm de ser impercetíveis	Performance
1.9. O número de *tokens* a ser gerido num momento pode ser de vários milhares	Performance
1.10. Integração via LDAP com Windows Active Directory e ou IBM Tivoli Directory Server em AS400 (*Read Only*)	Suportabilidade
1.11. *Web* API para ser acedido pelas aplicações para gestão das identidades e dos *tokens*	Suportabilidade
1.12. Manter os modos de autenticação e autorização já utilizados na organização	Suportabilidade
1.13. Suporte a servidores aplicacionais, de preferência WebSphere e Wildfly	Suportabilidade
1.14. Suporte a base de dados RDBMS como Oracle, SQLServer e DB2 for i	Suportabilidade

4.1.1 Casos de uso

Após terem sido definidos os requisitos pretendidos para a solução, é necessário a esquematização dos casos de uso pretendidos. Um caso de uso é uma representação em forma de diagrama de um requisito de sistema ou *software* para uma nova funcionalidade do *software* desenvolvido [38].

Na Figura 11, é apresentado o diagrama dos casos de uso que serão descritos.

Os casos de uso a descrever são:

- *Autenticate*;

- *Access Resouce*;
- *Manage Users and Groups*;
- *Configure client applications*.

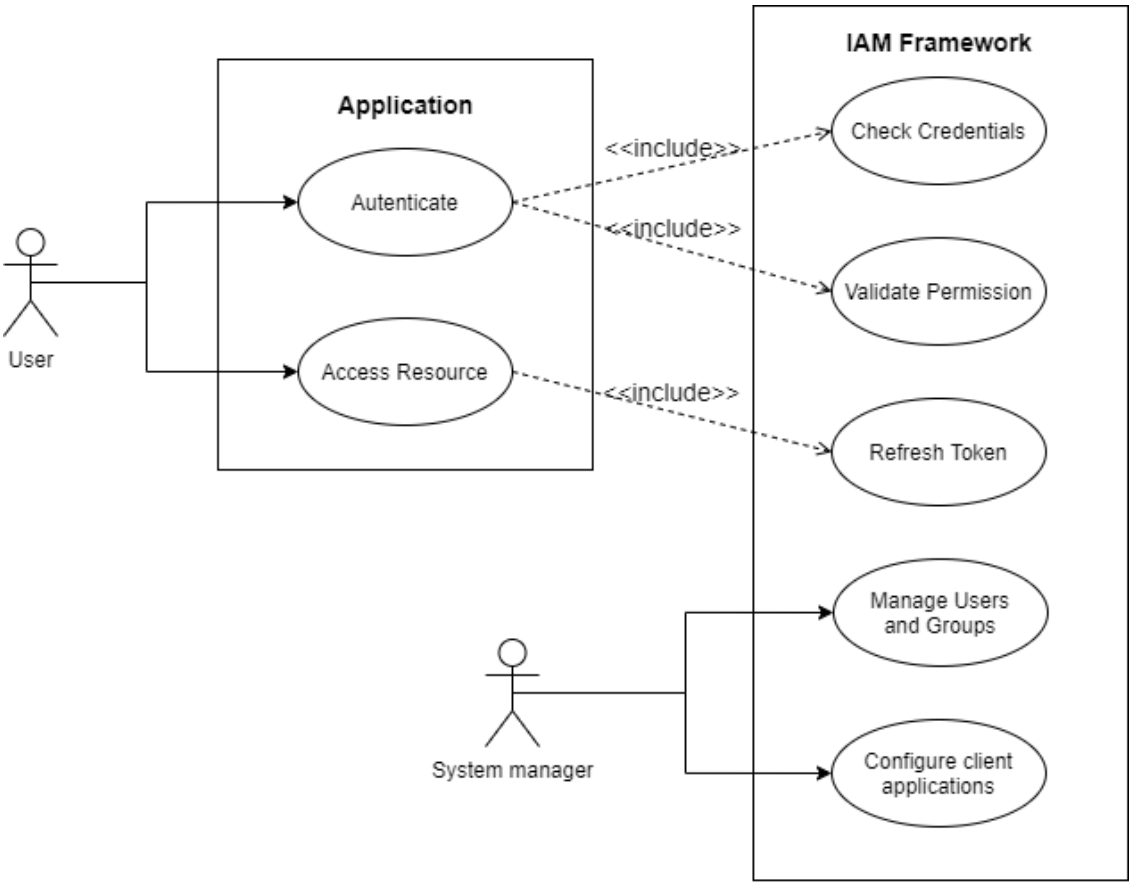


Figura 11 – Diagrama de casos de uso

4.1.1.1 Caso de uso: *Authenticate*

Um utilizador, pretende autenticar-se numa aplicação para ter acesso aos seus recursos. Na Figura 12, é descrito o cenário de sucesso deste caso de uso.

Para este caso de uso ter sucesso, o utilizador tem de fornecer um nome de utilizador e respetiva password válidos. O sistema valida estes dados através de um diretório de utilizadores como Windows Active Directory ou IBM Tivoli Directory Server. De seguida, o sistema valida se o utilizador tem acesso à aplicação que está a autenticar. Se estas duas validações tiverem sucesso, o sistema cria um *token* para o utilizador navegar na aplicação.

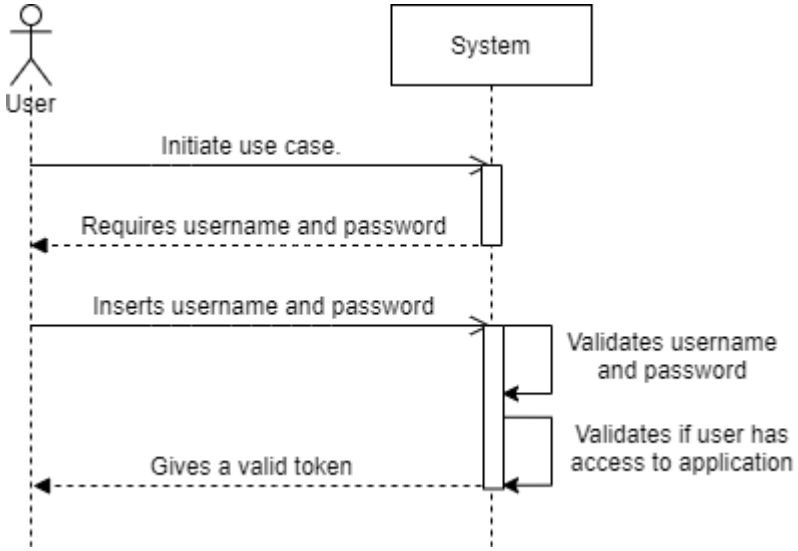


Figura 12 – Cenário de sucesso do caso de uso autenticar

4.1.1.2 Caso de uso: *Access Resource*

O utilizador, pretende aceder a um recurso disponibilizado por uma aplicação. Na Figura 13, é descrito o cenário de sucesso deste caso de uso.

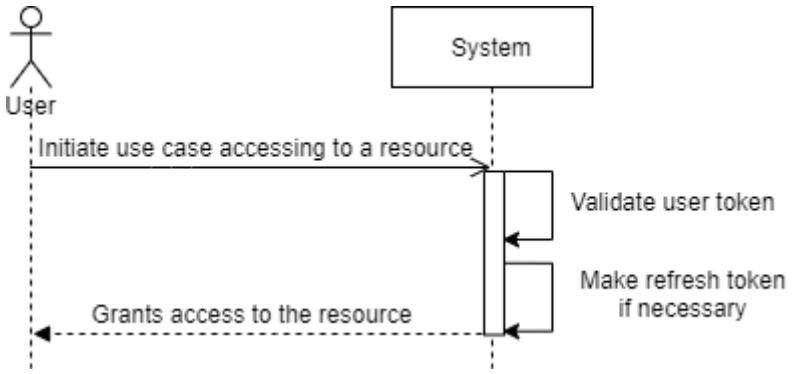


Figura 13 – Cenário de sucesso do caso de uso aceder a recurso

Para este caso de uso ter sucesso, o utilizador tem de possuir um *token* válido. Ao aceder a um recurso de uma aplicação, este *token* é enviado para o sistema para este o validar. O sistema valida o *token* validando a sua assinatura com uma chave publica fornecida pela *framework* IAM e valida a sua data de expiração. Se o *token* estiver expirado, é necessário atualizá-lo utilizando o *refresh token*. O *refresh token* é passado para a *framework* IAM que devolve um novo *token.* Se o *token*, passado pelo utilizador estiver válido ou se a atualização do *token* tiver sucesso, é permitido o acesso ao recurso pretendido.

4.1.1.3 Caso de uso: *Manage Users and Groups*

O gestor do sistema, pretende gerir os utilizadores e grupos editando os seus atributos e atribuindo permissões de acesso. Na Figura 14, é descrito o cenário de sucesso deste caso de uso.

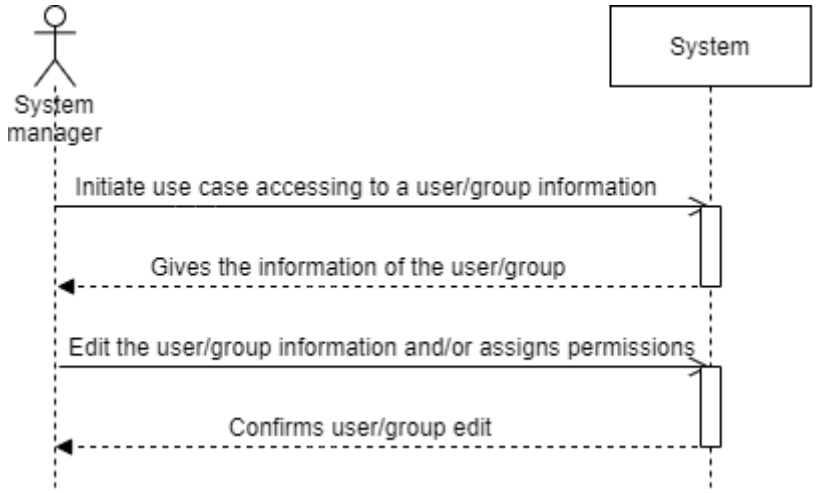


Figura 14 – Cenário de sucesso do caso de uso gerir utilizadores e grupos

Para este caso de uso ter sucesso, o gestor de sistema necessita de selecionar um utilizador ou grupo existente para que o sistema possa apresentar a sua informação. O utilizador pode alterar dados como o email, o nome, entre outros. O utilizador também pode gerir as permissões que cada utilizador tem ou do grupo. O sistema valida os dados introduzidos e confirma as alterações ao gestor de sistema.

4.1.1.4 Caso de uso: *Configure cliente applications*

Como gestor do sistema, quero configurar aplicações cliente para que estas consigam fazer a sua autenticação e autorizações na *framework* IAM. Na Figura 15, é descrito o cenário de sucesso deste caso de uso.

Para este caso de uso ter sucesso, o gestor de sistema tem de fornecer informação válida para que o registo da aplicação seja validado. O sistema valida os dados inseridos e regista a aplicação. Ao registar a aplicação, é gerado o *cliente secret*. O sistema confirma o registo da aplicação. O gestor de sistema, após a aplicação ser registada, pode adicionar as permissões que os utilizadores necessitam de ter para puderem aceder à aplicação e aos seus recursos. O sistema guarda as alterações e confirma ao gestor a sua alteração.

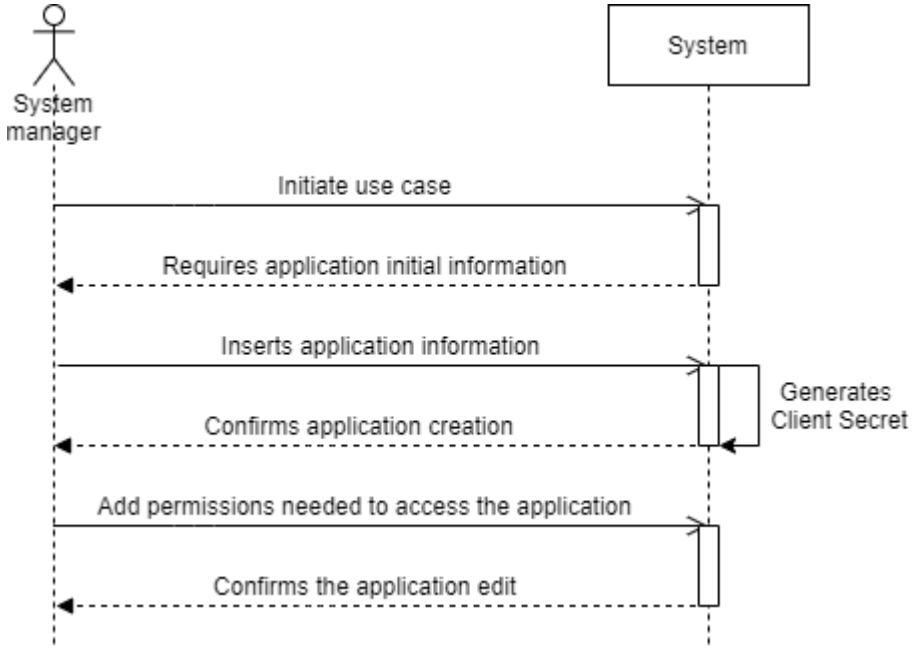


Figura 15 – Cenário de sucesso do caso de uso configurar aplicações cliente

4.2 Arquitetura atual

Nesta secção será descrita a arquitetura que as aplicações da organização utilizam para se autenticarem e validarem as autorizações. É possível visualizar a arquitetura na Figura 16.

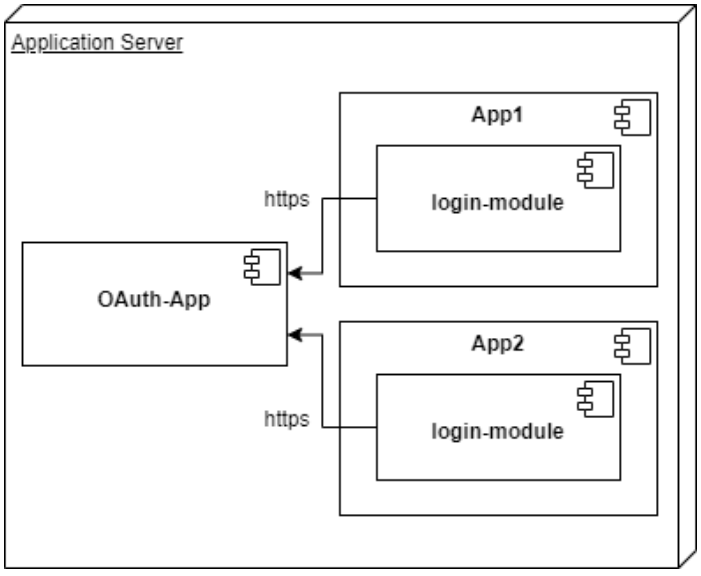


Figura 16 – Arquitetura atual

As aplicações estão instaladas num ou mais servidores aplicacionais e comunicam entre si através do protocolo HTTPS. Os servidores aplicacionais utilizados são o Wildfly e o WebSphere.

Para as funcionalidades de autenticação, as aplicações recorrem ao módulo “login-module” que comunica com a aplicação “OAuth-App” para, no caso de autenticação, validar as credenciais do utilizador e gerar um *token* válido para o utilizador.

Para as funcionalidades de autorização, o processo é o mesmo, as aplicações recorrem ao módulo “login-module” que pede à aplicação “OAuth-App” para verificar se o *token* está válido.

4.3 Análise das abordagens

Nesta secção serão apresentadas as alternativas para a integração da *framework* IAM com a arquitetura existente na organização. A primeira alternativa será a utilização dos adaptadores fornecidos pela *framework* para as aplicações puderem fazer a comunicação de autenticação e autorização com o servidor de autenticação. A segunda alternativa é a utilização dos serviços *web* fornecidos pela *framework* IAM para as aplicações fazerem as funcionalidades de autenticação e autorização através de serviços Simple Object Access Protocol (SOAP) ou Representation State Tranfer (REST). Nestas abordagens foi utilizada a *framework* Keycloak por ser uma solução *open-source* e não ter custos para a empresa para esta fase de análise. A última abordagem é manter a estrutura atual, mas melhorar as funcionalidades já existentes de autenticação e autorização e as funcionalidades de gestão dos utilizadores e os seus acessos.

4.3.1 Utilização dos adaptadores

A primeira abordagem consiste na utilização de adaptadores, que terão de ser implementados nas aplicações clientes, para fazerem a comunicação com a *framework* IAM. Para tal, a arquitetura do sistema, com a integração da *framework* IAM ficaria como é apresentado na Figura 17.

Com a utilização dos adaptadores seria necessária a sua implementação em cada uma das aplicações. Deste modo, a autenticação nas aplicações e o acesso a cada recurso disponibilizado pelas aplicações, passaria pelo adaptador que comunicaria com a *framework* IAM para permitir o acesso ao utilizador. Assim, a aplicação ficaria protegida de acordo com as políticas de acesso configuradas na *framework* para cada uma das aplicações.

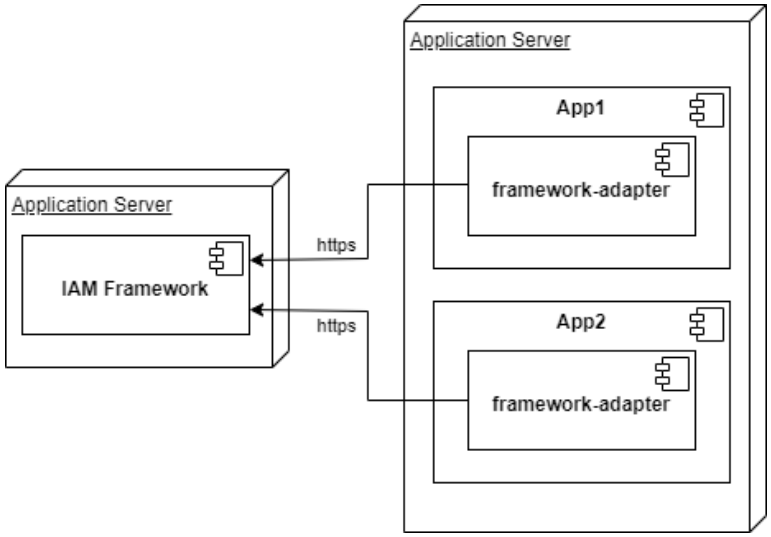


Figura 17 – Arquitetura com a utilização de adaptadores

De seguida são apresentadas as vantagens e desvantagens para a utilização desta abordagem na Tabela 14.

Tabela 14 – Vantagens e desvantagens da utilização dos adaptadores

Vantagens	• Funcionalidades de autenticação e autorização são delegadas ao *software* fornecido pela *framework*; • A gestão dos *token* é feita pelos adaptadores; • A funcionalidade de SSO é delegado aos adaptadores; • A atualização do *token* é gerida pelos adaptadores; • Pouca alteração nas aplicações da organização.
Desvantagens	• Os adaptadores são diferentes em cada uma das *frameworks*, ou seja, só se conseguia utilizar uma *framework*; • Os modos de autenticação e autorização anteriormente existentes já não seriam utilizáveis.

4.3.2 Utilização dos serviços *web*

A segunda abordagem consiste na utilização dos serviços *web* fornecidos pela *framework* IAM. Assim a comunicação com o servidor de autenticação teria de ser feita através de um módulo implementado dentro da organização. Para tal, a arquitetura do sistema ficaria como é apresentado na Figura 18.

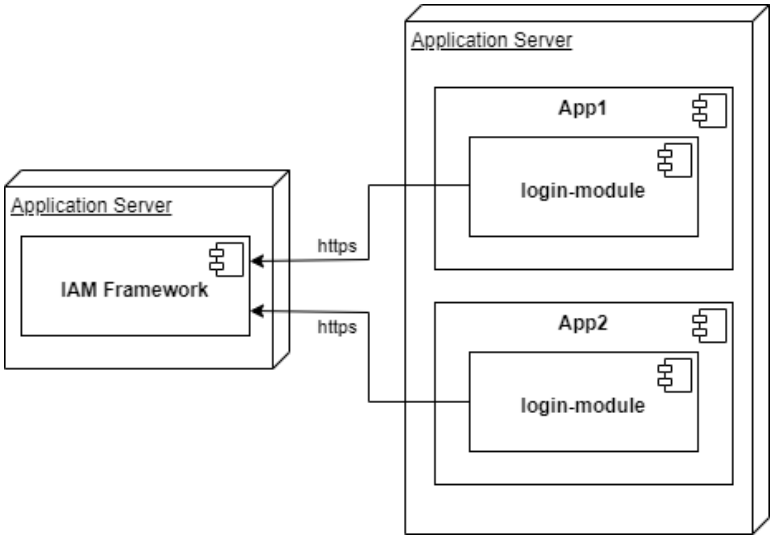


Figura 18 – Arquitetura com a utilização dos serviços *web*

Com a utilização dos serviços *web* seria necessário a implementação de um módulo que ficaria responsável pela autenticação e autorização para o acesso aos recursos das aplicações. Tendo em conta que este módulo já existe, mas não está configurado para comunicar com um servidor de autenticação externo, seria necessário adicionar essa funcionalidade a esse módulo. Este módulo teria a funcionalidade de autenticação que comunicaria com a *framework* IAM para validar a identidade do utilizador e teria a funcionalidade de intercetar o acesso aos recursos, disponibilizados pela aplicação, e validar se o utilizador teria acesso a esse recurso. Deste modo, a aplicação estaria protegida de acordo com as políticas de acesso configuradas na *framework* IAM implementada.

De seguida são apresentadas as vantagens e desvantagens para a utilização desta abordagem na Tabela 15.

Tabela 15 – Vantagens e desvantagens da utilização dos serviços *web*

Vantagens	• Permite a utilização de qualquer *framework* IAM que forneça serviços *web* de OAuth; • Os modos autenticação e autorização existentes continuam a ser utilizáveis através de configuração; • Funcionalidades de autenticação e autorização são geridas pelos serviços *web* disponibilizados pela *framework*.
Desvantagens	• Os serviços *web* têm de obedecer ao protocolo OAuth para que o formato das respostas aos pedidos seja idêntico de *framework* para *framework*; • É necessário algum esforço de implementação.

4.3.3 Desenvolver solução atual

Esta última abordagem consiste em manter a arquitetura atual e utilizar uma aplicação, já existente, que gere os utilizadores e os seus grupos e papeis. Como esta ainda não possui as funcionalidades pretendidas, seria necessário fazer alguns desenvolvimentos para que esta abordagem atingisse todos os requisitos pretendidos. Para tal, arquitetura do sistema ficaria como é apresentado na Figura 19.

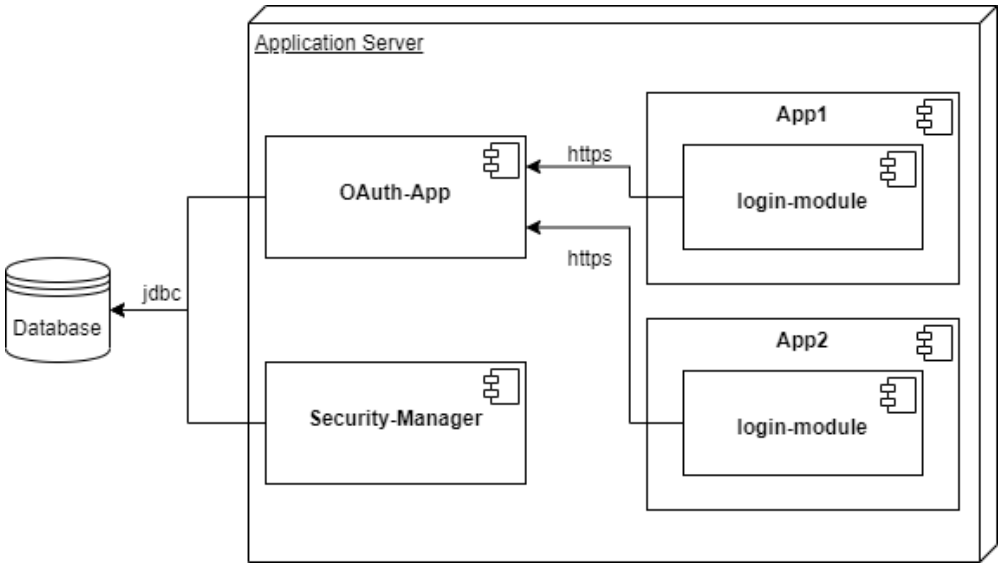


Figura 19 – Arquitetura fazendo desenvolvimentos na solução atual

Nesta arquitetura, a aplicação *Security-Manager* teria as mesmas funcionalidades que uma *framework* IAM, ou seja, gestão de utilizadores, grupos e papeis e gerir os privilégios necessários para um determinado utilizador aceder a uma determinada aplicação. Esta informação seria armazenada numa base de dados partilhada com a aplicação *OAuth-App* que tem como objetivo fazer a autenticação e validar a autorização do utilizador para aceder a uma aplicação. Com esta arquitetura, não seria necessário a utilização de uma *framework* IAM externa.

De seguida são apresentadas as vantagens e desvantagens para a utilização desta abordagem na Tabela 16.

Tabela 16 – Vantagens e desvantagens da utilização dos serviços *web*

Vantagens	• Os modos autenticação e autorização existentes continuam a ser utilizáveis através de configuração; • Maior liberdade para desenvolver apenas as funcionalidades necessárias para atingir os requisitos;
Desvantagens	• É necessário um grande esforço de desenvolvimento e implementação.

4.3.4 Avaliação das abordagens

Para fazer a avaliação das abordagens existentes, aplicou-se o modelo AHP onde se vai comparar as três soluções encontradas para fazer a implementação de uma *framework* IAM.

4.3.4.1 Estrutura hierárquica

Na Figura 20 é possível visualizar a estrutura hierárquica para o desenvolvimento deste projeto.

Os critérios identificados foram o tempo de desenvolvimento, o custo da implementação, ou seja, o que será necessário alterar para atingir os requisitos e manter as funcionalidades de autenticação e autorização onde algumas são especificas de cliente e deverão ser mantidas.

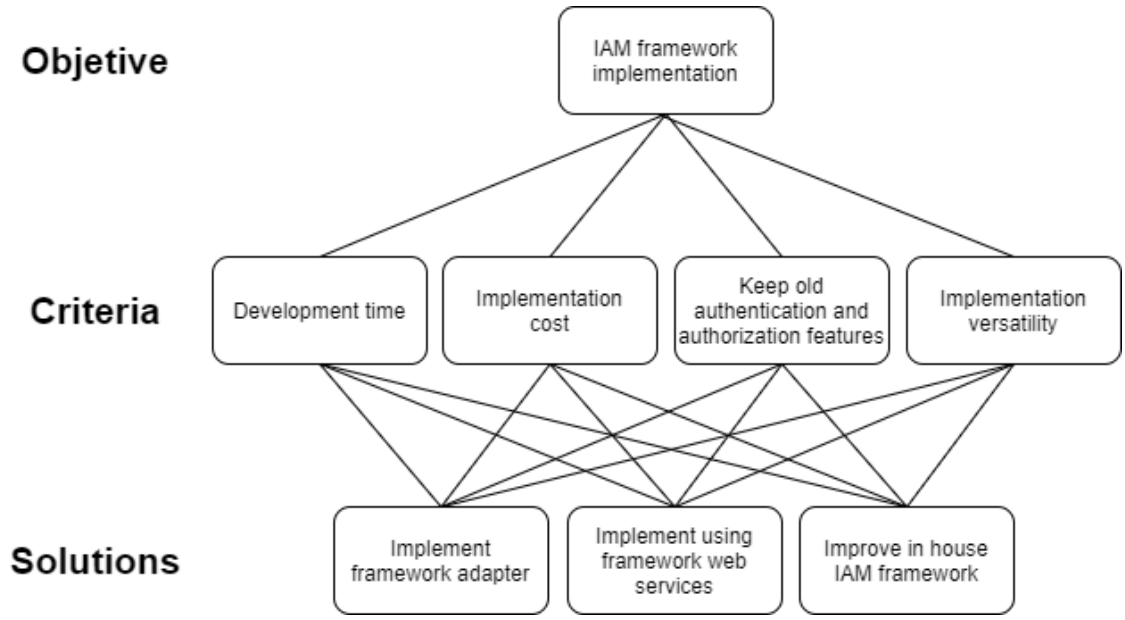


Figura 20 – Estrutura hierárquica do modelo AHP

4.3.4.2 Matriz de importância de critérios

Para quantificar os critérios, Saaty criou uma escala de valores que identifica o peso ou importância para o relacionamento entre critérios. É possível visualizar esta escala na Tabela 17.

Com a ajuda da escala, definiu-se que o critério mais importante seria a manter as funcionalidades antigas de autenticação e autorização, sendo identificado com extrema importância. Identificado com forte importância foi o critério do custo de implementação e de importância moderada o tempo de implementação.

Tabela 17 – Escala de Saaty (adaptado de [12])

Escala	Classificação Numérica	Recíproca (decimal)
Extrema importância	9	1/9 (0.111)
Muita a extrema importância	8	1/8 (0.125)
Muita importância	7	1/7 (0.143)
Forte a muita importância	6	1/6 (0.167)
Forte importância	5	1/5 (0.200)
Moderada a forte importância	4	1/4 (0.250)
Moderada importância	3	1/3 (0.333)
Igual a moderada importância	2	1/2 (0.500)
Igual importância	1	1 (1.000)

A Tabela 18 apresenta a matriz de importância dos critérios definidos anteriormente. O critério “A” é manter as funcionalidades antigas, o “B” é a versatilidade de implementação, o “C” é o custo de implementação e o “D” é o tempo de implementação.

Tabela 18 – Matriz de importância de critérios

	A	B	C	D
A	1	5	3	7
B	1/5	1	1/3	5
C	1/3	3	1	5
D	1/7	1/3	1/5	1

Com esta matriz, é possível definir o vetor de prioridades e que irá indicar o peso de cada critério. O cálculo é feito através da divisão de cada elemento de cada coluna da matriz com a soma dos valores dessa coluna mais o cálculo da média de cada nova linha da matriz [12]. Estes resultados são apresentados na Tabela 19.

Tabela 19 – Vetor de prioridades

A	0,546
B	0,144
C	0,255
D	0,055

Com os valores do vetor de prioridades, consegues se verificar que o critério de manter as funcionalidades antigas tem aproximadamente 54,6% de prioridade, o critério de versatilidade de implementação 14,4%, o critério de custo de implementação 25,5% e o critério de tempo de desenvolvimento 5,5%. Com estes valores é possível aplicar o peso de cada critério às diferentes alternativas.

4.3.4.3 Matrizes de comparação

O próximo passo é comparar as diferentes alternativas para cada um dos critérios. As alternativas são o uso de adaptadores fornecidos pelas *frameworks* de IAM ou o uso dos serviços *web* ou o desenvolvimento interno, também fornecidos por estas.

Na Tabela 20 pode-se visualizar a matriz de comparação para o critério de manter as funcionalidades antigas de autenticação e autorização.

Tabela 20 – Matriz de comparação de manter as funcionalidades antigas

	Adaptadores	**Serviços *web***	**Desenvolvimento Interno**
Adaptadores	1	1/3	1/3
Serviços *web*	3	1	3
Desenvolvimento Interno	3	1/3	1

Apesar de em todos os casos haverá a necessidade de desenvolvimentos, um para a implementação dos adaptadores, outro para a utilização de serviços *web* externos e no último para melhorar a plataforma criada na companhia, existe a possibilidade de que com a utilização dos adaptadores não ser possível utilizar as funcionalidades de autenticação e autorização existentes atualmente. Assim, concluiu-se que a utilização dos serviços *web* e desenvolvimento interno seria melhor para este critério.

Na Tabela 21 é possível visualizar a matriz de comparação para o critério de versatilidade de implementação.

Tabela 21 – Matriz de comparação para a versatilidade de implementação

	Adaptadores	**Serviços *web***	**Desenvolvimento Interno**
Adaptadores	9	1/5	1
Serviços *web*	5	1	5
Desenvolvimento Interno	1	1/5	1/9

Tendo em conta que seria necessário desenvolvimento para melhorar a plataforma de IAM existente, a versatilidade seria muito maior pois só se implementaria as funcionalidades pretendidas, conclui-se que para este critério este desenvolvimento teria vantagem relativamente à utilização dos serviços *web* e dos adaptadores.

Na Tabela 22 é possível visualizar a matriz de comparação para o critério de custo de implementação.

Tabela 22 – Matriz de comparação para o custo de implementação

	Adaptadores	**Serviços *web***	**Desenvolvimento Interno**
Adaptadores	1	1/3	1/7
Serviços *web*	3	1	1/3
Desenvolvimento Interno	7	3	1

Apesar de não se saber que complexidade existe na implementação dos adaptadores, verificou-se que, normalmente, estas *frameworks* fornecem boa documentação e bons exemplos para uma possível integração. Por outro lado, a utilização dos serviços *web* poderá trazer mais trabalho pois é necessário um maior desenvolvimento e a solução de desenvolvimento interno teria ainda um maior custo do que a opção anterior. Deste modo, conclui-se que a utilização dos adaptadores teria uma vantagem relativamente a este critério.

Na Tabela 23 é possível visualizar a matriz de comparação para o critério de tempo de implementação

Tabela 23 – Matriz de comparação para o tempo de implementação

	Adaptadores	**Serviços *web***	**Desenvolvimento Interno**
Adaptadores	1	1/2	1/9
Serviços *web*	2	1	1/2
Desenvolvimento Interno	9	2	1

Tendo em conta que para a utilização dos serviços *web* ou o desenvolvimento interno para melhorar a plataforma atual de IAM existirá uma necessidade de uma maior análise e um maior desenvolvimento, conclui-se que a utilização dos adaptadores seria a melhor alternativa para este critério.

4.3.4.4 Classificação das alternativas

Após uma análise detalhada de cada critério quando comparado com cada alternativa, é necessário calcular o vetor prioridade para cada um dos critérios. A Tabela 24 apresenta estes valores.

Tendo estes valores, consegue-se classificar as diferentes alternativas através da multiplicação de cada elemento da Tabela 24 com o peso dos critérios calculados na Tabela 19, somando a linha para obter o resultado.

Tabela 24 – Matriz de prioridades dos critérios para cada alternativa

	A	B	C	D
Adaptadores	0,142	0,067	0,636	0,750
Serviços *web*	0,429	0,333	0,273	0,167
Desenvolvimento Interno	0,429	0,600	0,091	0,083

O resultado é possível ser visualizado na Tabela 25.

Tabela 25 – Classificação das alternativas

	A	B	C	D	Resultado
Adaptadores	0,078	0,009	0,162	0,041	0,29
Serviços *web*	0,234	0,048	0,069	0,009	0,36
Desenvolvimento Interno	0,234	0,086	0,023	0,004	0,35

Pode-se então confirmar que a utilização dos serviços *web* tem um resultado favorável de aproximadamente 36% enquanto o desenvolvimento para melhorar a plataforma existente teve 35% e a utilização dos adaptadores apenas de 29%

4.3.5 Decisão da abordagem

Após terem sido apresentadas as abordagens existentes, foi necessário decidir qual delas seria implementada. Para tal foi feito uma análise aplicando o método estatístico AHP para decidir qual das abordagens daria mais garantias para atingir os resultados pretendidos.

Os resultados da aplicação do método AHP decidiu que a implementação dos serviços *web* seria a melhor abordagem para resolver o problema. Analisando os resultados, é também possível ver que a abordagem de desenvolver a solução atual também seria uma boa opção a seguir.

Tendo em conta os resultados desta análise e discussões com o supervisor na empresa, decidiu-se assim que se iria avançar com a abordagem dos serviços *web* e fazer a sua implementação. Para a implementação e testes, será utilizada a *framework* Keycloak por não ter custos para a organização.

4.4 Sumário

Neste capítulo foi possível perceber, com maior detalhe, o que se pretende que a solução final realize. Ao longo do capítulo foram apresentados os requisitos funcionais e não funcionais, caracterizados de acordo com o modelo FURPS, e os principais casos de uso que a solução deverá disponibilizar aos utilizadores.

Após a descrição dos requisitos e dos casos de uso, fez-se uma análise às possíveis abordagens que se poderia seguir e quais as suas vantagens e desvantagens. A primeira abordagem consiste na utilização dos adaptadores fornecidos pela *framework* e a segunda abordagem consiste na utilização dos serviços *web* disponibilizados pela *framework*.

No início deste capítulo foi proposto que fosse respondida à seguinte questão:

1. Quais as abordagens para a resolução do problema?

R: Foram analisadas três abordagens, a utilização dos adaptadores fornecidos pela *framework*, a utilização dos serviços *web* também disponibilizados pela *framework* e o desenvolvimento das funcionalidades pretendidas numa aplicação já existente. A utilização dos adaptadores é uma abordagem mais limpa, ou seja, não é necessário grandes alterações nas aplicações da organização, no entanto, os modos de autenticação e autorização existentes deixariam de ser utilizados. A utilização dos serviços *web*, apesar de ser necessário um pouco de mais esforço na implementação, permite que os modos de autenticação existentes também sejam utilizados através de configuração. O desenvolvimento da solução atual é uma abordagem com maior custo devido à necessidade de fazer um maior desenvolvimento para a implementação das funcionalidades pretendidas, mas permite ter uma maior liberdade para o desenvolvimento.

2. Qual a abordagem escolhida?

R: Das três abordagens escolhidas a que melhor se adequou ao problema foi a de utilização dos serviços *web* devido à versatilidade de implementação e da sua compatibilidade com os modos de autenticação já existentes.

5 Desenho da solução

Após ter sido feito a análise dos requisitos, dos casos de uso pretendidos e das abordagens existente para resolver o problema, é necessário definir e desenhar a solução proposta por esta dissertação.

Tendo sido decidido pela organização a utilização da abordagem que utiliza os serviços *web* da *framework* de IAM, neste capítulo será apresentada a arquitetura, o modelo de dados e os processos mais relevantes com base na abordagem selecionada.

Ao longo deste capítulo será dada a resposta à seguinte questão:

1. Qual a solução proposta para combater o problema estudado?

5.1 Arquitetura

Na Figura 21, é apresentada a primeira proposta para a arquitetura da solução.

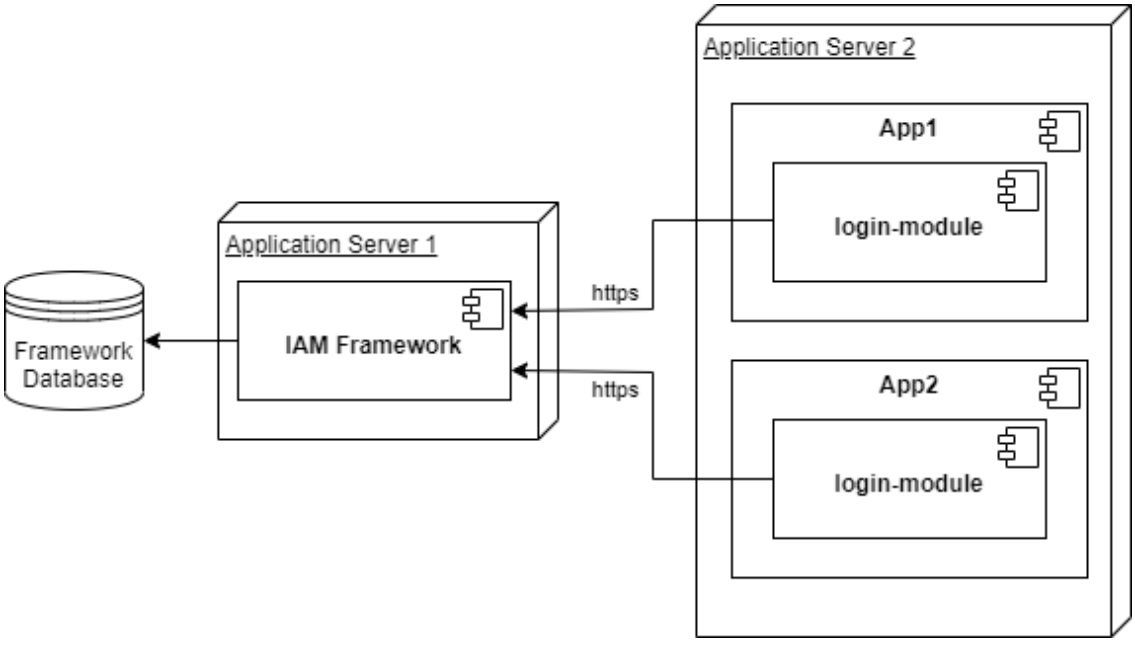


Figura 21 – Arquitetura inicial da solução

Inicialmente haveria um componente, o “login-module”, que seria responsável pela autenticação e gestão das autorizações dos utilizadores. Este componente faz a comunicação com a *framework* IAM através dos serviços que esta disponibiliza. Mas foi encontrado um problema com esta arquitetura.

Sendo o componente “login-module” responsável pelas funcionalidades de autenticação e gestão de autorizações, teria de fazer a gestão do *token* e do respetivo

refresh token. A maneira como este faria esta gestão era guardando o *token* como *cookie* e o *refresh token*, não pudendo guardá-lo como *cookie* por motivos de segurança, seria guardado como atributo na sessão de Java. Sendo a sessão de Java dependente da JVM onde o servidor aplicacional está a ser executado, deparou-se com o problema de havendo duas aplicações instaladas em servidores aplicacionais diferentes, a sessão de Java criada uma aplicação não é acessível através da outra aplicação. Assim, no caso do utilizador se autenticar numa das aplicações e navegar através do *browser* para a outra aplicação, esta não teria acesso ao *refresh token* e no caso do *token* expirar não será possível atualizá-lo.

Para resolver este problema, desenhou-se uma nova arquitetura apresentada na Figura 22.

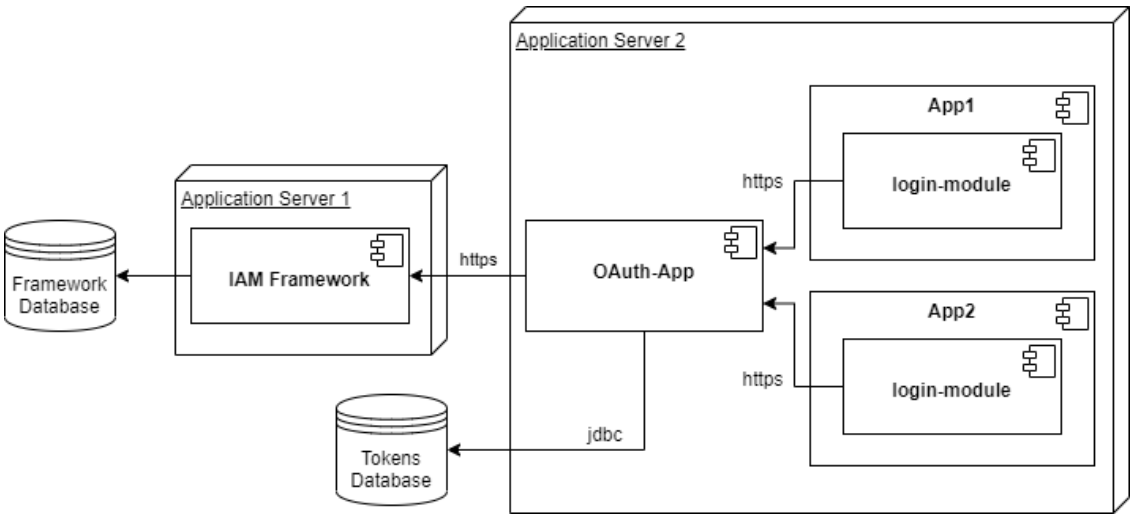


Figura 22 – Arquitetura final da solução

Foi necessário acrescentar uma aplicação, a “OAuth-App” que já era utilizada pelos outros modos de autenticação e autorização, que fará de *middleman* para as comunicações entre as aplicações e a *framework* de IAM e todos os *tokens* e respetivos e *refresh tokens* serão guardados numa base de dados.

Assim, utilizando o mesmo exemplo anteriormente descrito, se um utilizador se autenticar numa aplicação, a aplicação “OAuth-App” guarda o seu *refresh token* e *token* na base de dados, é criado a *cookie* com o *token.* Se o utilizador navegar através do *browser* para a outra aplicação, instalada noutro servidor aplicacional, quando o seu *token* expirar, esta segunda aplicação pede à aplicação “OAuth-App” para atualizar o *token*. Esta sabendo qual o *refresh token* relacionado com o *token* do utilizador, através da base de dados, consegue fazer o pedido à *framework* de IAM para atualizar o *token*. Assim, é possível que a funcionalidade de *refresh* do *token* esteja sempre disponível.

5.2 Base de dados

Na Figura 23, é apresentado o modelo de dados a ser utilizado. As bases de dados que utilizarão este modelo serão base de dados relacionais como Oracle, Microsoft SQLServer e DB2 for i.

TOKEN		
PK	ID	NUMBER(18)
	TOKENHASH	VARCHAR(256)
	TOKENKEY	VARCHAR(4000)
	EXPIRESIN	NUMBER(19)
	ISSUEDATE	NUMBER(19)
	ISSUER	VARCHAR(256)
	REFRESHTOKEN	VARCHAR(4000)
	TOKENTYPE	VARCHAR(256)
	GRANTTYPE	VARCHAR(256)
	CLIENTID	VARCHAR(256)

Figura 23 – Modelo de dados

O modelo de dados, é apenas constituído por uma tabela onde serão armazenados os *tokens*. Os atributos relevantes do *token* a serem guardados são o próprio *token* (campo “TOKENKEY”), o respetivo *refresh token* (campo “REFRESHTOKEN”), quando é que o *token* expira (campo “EXPIRESIN”), quando é que o *token* foi gerado (campo “ISSUEDATE”), quem é que gerou o *token* (campo “ISSUER”), ou seja, qual foi o servidor de autenticação. É também armazenado o tipo do *token* (campo “TOKENTYPE”), o tipo do pedido que fez a geração do *token* (campo “GRANTTYPE”), se foi através de credenciais ou se foi através de um pedido de *refresh* do *token* e a identificação da aplicação que fez o pedido para a geração do *token* (campo “CLIENTID”).

Para além do armazenamento destes campos, também é necessário armazenar uma *hash* do *token*, o campo “TOKENHASH”. Esta *hash* irá ter um índice para acelerar o processo de procura do *token*. Este índice não é colocado no campo do *token* pois o tamanho do campo é maior do que o permitido para a criação de índices nos motores de base de dados utilizados pela organização. É também utilizado um campo numérico como chave primária que irá utilizar uma sequência, o campo “ID”.

5.3 Processos de execução

Existem dois momentos imprescindíveis que devem ser implementados para que todo o processo de autenticação e autorização corra com sucesso. O momento da autenticação e a validação do acesso do utilizador a um recurso de uma aplicação.

Na Figura 24 é possível visualizar o processo de autenticação a implementar.

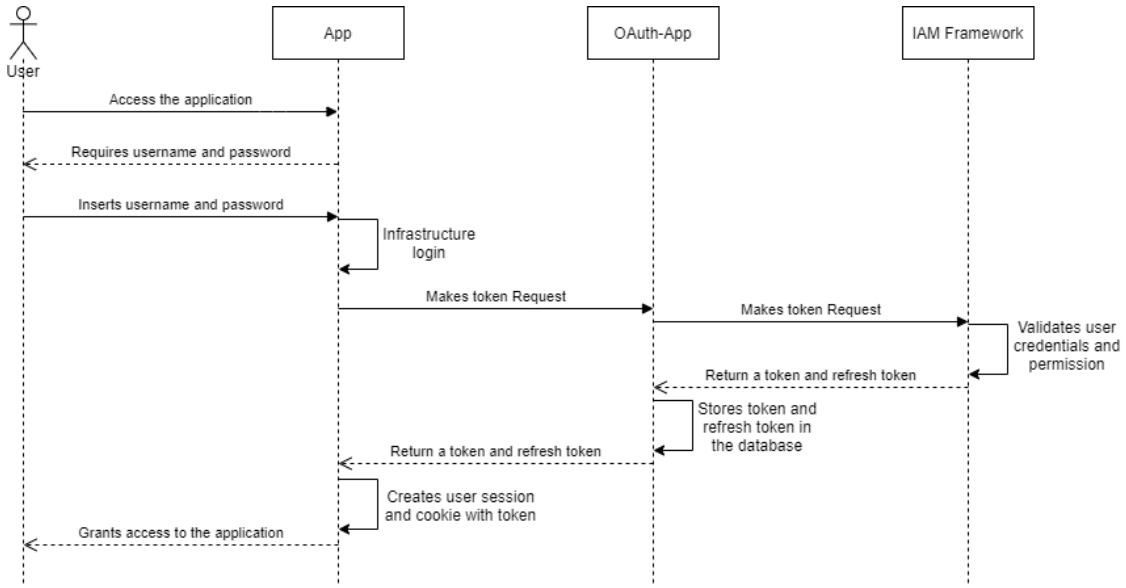


Figura 24 – Processo de autenticação

No processo de autenticação, o utilizador tem como intenção aceder a uma aplicação. Para tal, a aplicação pede ao utilizador as suas credenciais para validar a sua identidade. Quando o utilizador insere as suas credenciais, a aplicação executa o *login* de infraestrutura, normalmente feito através de um servidor LDAP, e faz o pedido de um novo *token* à aplicação "OAuth-App". Esta pede à *framework* de IAM para gerar um novo *token* e respetivo e respetivo *refresh token*. A "OAuth-App", após receber o *token*, armazena-o na base de dados, com o *refresh token* e devolve o token à aplicação inicial. Por fim, a aplicação, que o utilizador pretende aceder, cria a sessão de Java e guarda a *cookie* com o *token* nos seus atributos e fornece acesso para o utilizador navegar na aplicação.

Na Figura 25 é possível visualizar o processo de autorização quando o utilizador tenta aceder a um recurso de uma aplicação.

Neste processo o utilizador tem intenção de aceder a um recurso de uma aplicação protegida. Ao fazer o pedido, a aplicação vai validar o *token* do utilizar. Esta validação tem três fases. Inicialmente é validada a assinatura do *token* através da chave pública emitida pela *framework* de IAM, que utilizou a respetiva chave privada para assinar o *token*.

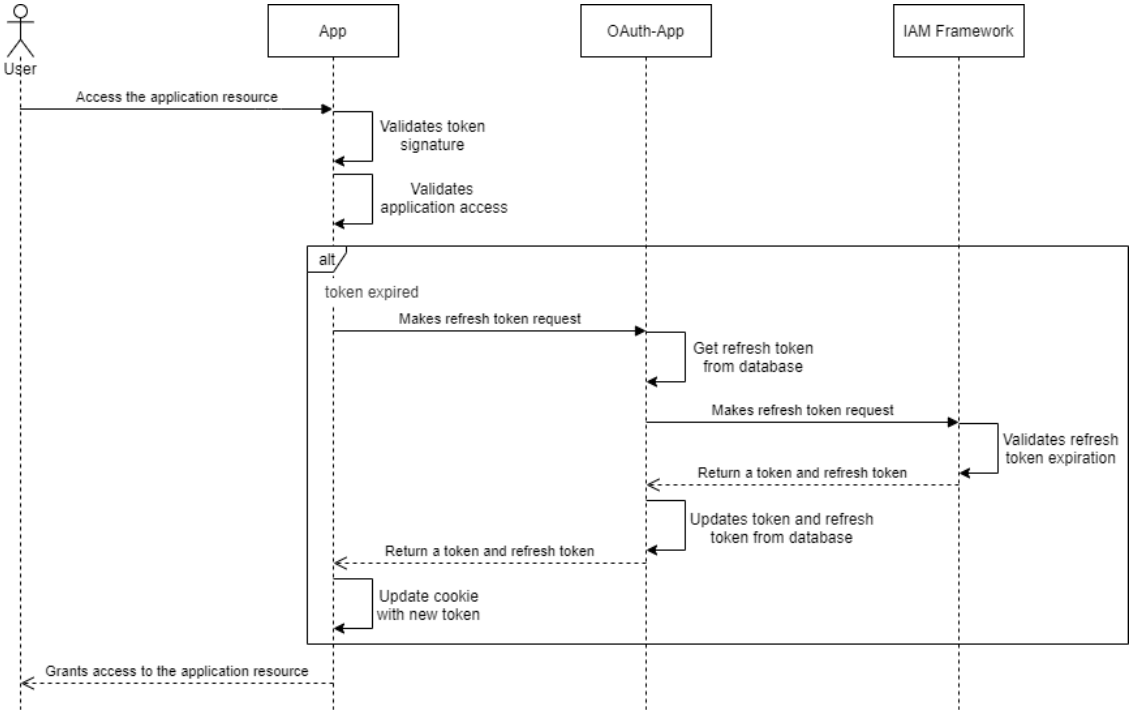


Figura 25 – Processo de autorização

Depois da assinatura ser validada, a aplicação verifica se o utilizador tem permissão para aceder aos seus recursos através dos *scopes* que o *token* transporta. Por fim, é validado se o *token* está expirado. No caso deste estiver expirado, a aplicação faz um pedido à aplicação "OAuth-App" para fazer a sua atualização. Esta aplicação vai buscar o *refresh token*, armazenado na base de dados, do *token* expirado e faz o pedido à *framework* de IAM para gerar um novo *token*. Quando recebe o novo *token* e respetivo *refresh token*, a aplicação "OAuth-App" atualiza os *token* na base de dados e devolve o novo *token* à aplicação inicial. Esta, atualiza a *cookie* com o novo *token* e fornece acesso ao utilizador ao recurso da aplicação.

5.4 Implementação

Nesta secção é apresentado como foi feita a implementação dos dois processos de execução apresentados anteriormente, olhando para o código relevante desenvolvido. Serão também apresentadas as configurações necessárias para a execução correta do código.

5.4.1 Autenticação

O processo de autenticação inicia-se quando um utilizador introduz as suas credenciais para aceder a uma aplicação. Esta envia através de um pedido *Hyper Text Transfer Protocol Secure* (HTTPS) as informações do utilizador para a aplicação responsável pela comunicação com a *framework* IAM.

```
1  String clientId = params.getFirst(OAuthConstants.CLIENT_ID);
2  String clientSecret = params.getFirst(OAuthConstants.CLIENT_SECRET);
3  String grantType = params.getFirst(OAuthConstants.GRANT_TYPE);
4  String[] pathParams = getPathParams(params);
```

Excerto de Código 1 – Excerto de código do serviço de autenticação (1/5)

No Excerto de Código 1, é possível visualizar os primeiros passos do serviço de autenticação. Inicialmente é necessário verificar a identificação da aplicação, que é feita do lado da *framework* IAM. Assim, o serviço procura no pedido o *client id* e *client secret* (linha 1 e 2) e o *grant type* (linha 3) para verificar o motivo do pedido. Os únicos motivos válidos neste serviço são o pedido de um novo *token* ou o pedido de *refresh token*. No caso da autenticação, o motivo é o pedido de um novo *token*. Também se procura por uns parâmetros que podem ser necessários colocar no URL para a comunicação com a *framework* IAM. Após esta validação é possível avançar para o próximo passo.

```
1  String ownerName = params.getFirst(OAuthConstants.RESOURCE_OWNER_NAME);
2  String ownerPassword =
       params.getFirst(OAuthConstants.RESOURCE_OWNER_PASSWORD);
3
4  if (ownerName == null || ownerPassword == null) {
5      getLog().warn("Resource owner name or password is null.");
6      throw new OAuthServiceException(new
             OAuthError(OAuthConstants.INVALID_REQUEST));
7  }
```

Excerto de Código 2 – Excerto de código do serviço de autenticação (2/5)

De seguida, é necessário saber quais são as credenciais do utilizador, que pode-se visualizar no Excerto de Código 2. Assim, vai-se procurar no pedido pelo nome de utilizador e pela sua *password* (linha 1 e 2) e de seguida validar que ambas as credencias são encontradas no pedido (linha 4). No caso de não serem encontradas, é lançada uma exceção a avisar que o pedido feito foi inválido.

```
1  Secret password = new Secret(ownerPassword);
2  AuthorizationGrant passwordGrant = new
       ResourceOwnerPasswordCredentialsGrant(ownerName, password);
3
4  TokenResponse tokenResponse = makeTokenRequest(clientId, clientSecret,
       pathParams, passwordGrant);
```

Excerto de Código 3 – Excerto de código do serviço de autenticação (3/5)

Após as validações iniciais, é necessário preparar o pedido para a geração de *token* à *framework* IAM, iniciada no Excerto de Código 3. Para tal, utilizou se uma biblioteca externa, o Nimbus JOSE + JWT [39], que permite trabalhar com JWT e o protocolo OAuth em Java. Para criar o pedido, é necessário criar o objeto da autorização que transporta o nome do utilizador e respetiva password (linha 1 e 2). De seguida executou-se o método “makeTokenRequest” (linha 4) que é responsável pela criação e

envio do pedido do *token* à *framework* IAM, que é possível visualizar no Excerto de Código 4.

```
1  ClientID clientID = new ClientID(clientId);
2  Secret secret = new Secret(clientSecret);
3  ClientAuthentication clientAuth = new ClientSecretBasic(clientID, secret);
4
5  URI tokenEndpoint = new URI(buildExternalOauthEndpoint(pathParams));
6
7  TokenRequest request = new TokenRequest(tokenEndpoint, clientAuth, grant);
8
9  HTTPResponse httpResponse = request.toHTTPRequest().send();
10
11 return TokenResponse.parse(httpResponse);
```

Excerto de Código 4 – Excerto de código do serviço de autenticação (4/5)

Neste método, cria-se o objeto que leva o *client id* (linha 1), o objeto que leva o *client secret* (linha 2) da aplicação e o objeto (linha 3) que leva estes anteriores. De seguida, é criado o objeto do URL (linha 5) onde podem ser utilizados os parâmetros recebidos no pedido. É criado o objeto para fazer o pedido (linha 7) que leva o objeto do URL do *client id* e *client secret* da aplicação e o objeto que leva o nome do utilizador e a password. Por fim, é executado o método para enviar o pedido (linha 9) que devolve o objeto retornado pelo pedido à *framework* IAM e feito o tratamento desse objeto (linha 11). O próximo e último passo é validar a resposta e devolver o resultado à aplicação inicial, este código é possível ser visualizado no Excerto de Código 5.

Para finalizar o serviço, é necessário fazer a validação da resposta obtida à *framework* IAM (linha 1). No caso de a resposta não ter tido sucesso, é feito o tratamento dessa resposta (linha 2) e é criado o objeto para responder à aplicação inicial (linha 4). Se a resposta tiver sucesso, é feito o tratamento dessa resposta (linha 6), são retirados os objetos do *token* e do *refresh token* e são armazenados na base de dados com o *client id* da aplicação inicial. Por fim, é feito o tratamento da resposta (linha 13 a 21) onde são colocados o *token*, o tempo de expiração deste, o *refresh token* e o tipo de *token*, que será sempre do tipo “BEARER”.

```
1 if (!tokenResponse.indicatesSuccess()) {
2      TokenErrorResponse errorResponse = tokenResponse.toErrorResponse();
3
4      response = Response.status(Status.OK).entity(new
              OAuthResponse<TokenErrorResponse>(new
                    I2SStatus(I2SMessageType.ERROR),
              errorResponse)).build();
5   } else {
6      AccessTokenResponse successResponse =
              tokenResponse.toSuccessResponse();
7
8      AccessToken accessToken =
successResponse.getTokens().getAccessToken();
9      RefreshToken refreshToken =
              successResponse.getTokens().getRefreshToken();
10
11     externalDatabaseUtils.saveTokenAndRefreshToken(accessToken,
              refreshToken, clientId);
12
13     JwtTokenResponse jwtResponse = new JwtTokenResponse();
14     jwtResponse.setAccess_token(accessToken.getValue());
15     jwtResponse.setExpires_in(accessToken.getLifetime());
16     jwtResponse.setRefresh_token(refreshToken.getValue());
17     jwtResponse.setToken_type(OAuthConstants.BEARER_TOKEN_TYPE);
18
19     OAuthResponse<JwtTokenResponse> oauthRes = new
              OAuthResponse<JwtTokenResponse>(new
       I2SStatus(I2SMessageType.INFO),
                     jwtResponse);
20
21     response = Response.status(Status.OK).entity(oauthRes).build();
22 }
```

Excerto de Código 5 – Excerto de código do serviço de autenticação (5/5)

5.4.2 Autorização

O processo de validação da autorização acontece quando um serviço de uma aplicação é acedido. Esta validação é feita antes de se aceder ao serviço e é feita do lado da aplicação cliente. No Excerto de Código 6 é possível visualizar o início da validação da autorização.

```
1  JWKSet keys = getKeySet();
2  if (keys != null) {
3      JWSObject jwsObject = JWSObject.parse(token);
4      JWK key = keys.getKeyByKeyId(jwsObject.getHeader().getKeyID());
5      if (key != null) {
6            JWSVerifier verifier = getJWSVerifier(key.getKeyType(), key);
7            if (verifier != null && jwsObject.verify(verifier)) {
8                  return true;
9            }
10     }
11 }
```

Excerto de Código 6 – Excerto de código de autorização 1/3

Inicialmente, é feita a validação da assinatura do *token*. Para tal, é necessário ir buscar as chaves publicas utilizadas para encriptar os *tokens* que são fornecidas pela aplicação que comunica com a *framework* IAM (linha 1). Após ter recebido a chave publica, é feito o tratamento do *token* utilizando objetos da biblioteca Nimbus JOSE + JWT (linha 3) e verificado se a chave utilizada para encriptar o *token* fornecido está entre as chaves recebidas anteriormente (linhas 4 e 5). Se a chave existir, é criado o objeto que verifica a assinatura do *token* (linha 6) e é feita a validação da assinatura (linha 7).

```
1  JWT jwt = JWTParser.parse(authSchemeData);
2  long expTime = jwt.getJWTClaimsSet().getExpirationTime().getTime() –
       System.currentTimeMillis();
```

Excerto de Código 7 – Excerto de código de autorização 2/3

Após a validação da assinatura do *token*, é necessário verificar se este está expirado. Esta verificação é possível ser visualizada no Excerto de Código 7, onde é feito o tratamento do *token* para um objeto de Java (linha 1) para se puder fazer o cálculo para verificar se está expirado (linha 2). No caso do *token* estar expirado, é pedido um novo *token* utilizando o *refresh token* armazenado na base de dados. Para tal, é feito um pedido à aplicação responsável pela comunicação com a *framework* IAM que pedirá seja gerado um novo *token*. No Excerto de Código 8, é possível visualizar o início desse serviço.

```
1  String token = params.getFirst("token");
2  String refreshTokenValue =
       externalDatabaseUtils.selectRefreshTokenByToken(token);
```

Excerto de Código 8 – Excerto de código do serviço de *refresh token* 1/3

No serviço de *refresh token*, inicialmente é necessário ir buscar o *token* aos parâmetros do pedido recebido (linha 1) para ir buscar o *refresh token* associado a este à base de dados (linha 2). No caso de não se encontrar o *refresh token* na base de dados não é possível gerar um novo *token* e é enviada a resposta com erro. Se o *refresh token* for

encontrado, passa-se para o próximo passo que é possível visualizar no Excerto de Código 9.

```
1  RefreshToken refreshToken = new RefreshToken(refreshTokenValue);
2  RefreshTokenGrant refreshTokenGrant = new RefreshTokenGrant(refreshToken);
3
4  TokenResponse tokenResponse = makeTokenRequest(clientId, clientSecret,
       pathParams, refreshTokenGrant);
```

Excerto de Código 9 – Excerto de código do serviço de *refresh token* 2/3

Após ter o *refresh token*, é iniciado o processo para pedir à *framework* IAM que gere um novo *token*. Para tal, é utilizado novamente a biblioteca externa Nimbus JOSE + JWT para criar este pedido. É criado o objeto do *refresh token* (linha 1 e 2) e é feito o pedido à *framework* IAM (linha 4). O método "makeTokenRequest" é o mesmo utilizado no pedido de autenticação no Código 4. De seguida é feito o tratamento da resposta que é possível visualizar no Excerto de Código 10.

```
1  if (!tokenResponse.indicatesSuccess()) {
2      TokenErrorResponse errorResponse = tokenResponse.toErrorResponse();
3
4      response = Response.status(Status.OK).entity(new
               OAuthResponse<TokenErrorResponse>(new
                     I2SStatus(I2SMessageType.ERROR), errorResponse)).build();
5  } else {
6      AccessTokenResponse successResponse =
               tokenResponse.toSuccessResponse();
7
8      AccessToken accessToken = successResponse.getTokens().getAccessToken();
9      refreshToken = successResponse.getTokens().getRefreshToken();
10
11     externalDatabaseUtils.updateTokenAndRefreshToken(token, accessToken,
               refreshToken, clientId);
12
13     JwtTokenResponse jwtResponse = new JwtTokenResponse();
14     jwtResponse.setAccess_token(accessToken.getValue());
15     jwtResponse.setExpires_in(accessToken.getLifetime());
16     jwtResponse.setRefresh_token(refreshToken.getValue());
17
18     jwtResponse.setToken_type(OAuthConstants.BEARER_TOKEN_TYPE);
19
20     OAuthResponse<JwtTokenResponse> oauthRes = new
               OAuthResponse<JwtTokenResponse>(new
                     I2SStatus(I2SMessageType.INFO),
                           jwtResponse);
21
22     response = Response.status(Status.OK).entity(oauthRes).build();
23 }
```

Excerto de Código 10 – Excerto de código do serviço de *refresh token* 3/3

Para finalizar o serviço, é necessário fazer o tratamento da resposta obtida à *framework* IAM (linha 1). No caso de a resposta não ter tido sucesso, é feito o tratamento dessa

resposta (linha 2) e é criado o objeto para responder à aplicação inicial (linha 4). Se a resposta tiver sucesso, é feito o tratamento dessa resposta (linha 6), são retirados os objetos do *token* e do *refresh token* e é feita a sua atualização na base de dados (linha 11). Por fim, é feito o tratamento da resposta (linha 13 a 21) onde são colocados o *token*, o tempo de expiração deste, o *refresh token* e o tipo de *token*, que será sempre do tipo "BEARER".

```
1  if (tokenResp != null) {
2      jwt = JWTParser.parse(tokenResp.getAccess_token());
3      expTime = jwt.getJWTClaimsSet().getExpirationTime().getTime() -
              System.currentTimeMillis();
4      if (response != null) {
5              Cookie tokenCookie = new Cookie("i2stoken",
                      tokenResp.getAccess_token());
6              if (request.isSecure()) {
7                      tokenCookie.setSecure(true);
8              }
9              tokenCookie.setPath("/");
10             tokenCookie.setHttpOnly(true);
11             tokenCookie.setMaxAge(-1);
12             response.addCookie(tokenCookie);
13     }
14 }
```

Excerto de Código 11 – Excerto de código de autorização 3/3

Após ter recebido a resposta do pedido de *refresh token*, iniciado no Excerto de Código 11. É necessário validar se o pedido teve sucesso (linha 1) para de seguida atualizar a *cookie* que transporta o *token*. Para isso, é feito o tratamento da resposta para obter o *token* (linha 2) e o seu tempo de expiração (linha 3) para criar a nova *cookie* com este novo *token* (linha 5 a 12)

5.5 Sumário

Neste capítulo foi possível perceber o que será implementado para resolver o problema estudado. Ao longo do capítulo foi apresentada a arquitetura, desde a infraestrutura utilizada, o modelo de base de dados e alguns processos importantes para a solução final.

A solução final está dividida em duas partes, uma que tem o servidor aplicacional onde estará instalada a *framework* de IAM e outra que terá os servidores aplicacionais onde estarão instaladas as aplicações da organização. As aplicações instaladas terão um módulo que será responsável pelas funcionalidades de autenticação e autorização dentro da aplicação. Este módulo comunica com uma outra aplicação responsável pela gestão dos *tokens* com a ajuda da base de dados e pela comunicação com a *framework* IAM. Com esta arquitetura todos os requisitos anteriormente definidos poderão ser atingidos.

No início deste capítulo foi proposto que fosse respondida à seguinte questão:

1. Qual a solução proposta para combater o problema estudado?

R: A solução proposta utiliza os serviços *web* disponibilizados pela *framework* IAM para gerar, atualizar e revogar os *tokens.* Deste modo, as aplicações da organização é que serão responsáveis por fazer a validação dos *tokens*.

6 Avaliação

Após a implementação da solução é necessário fazer uma avaliação com o objetivo de verificar se a solução é fiável em ambientes reais. Neste capítulo serão apresentados os testes que serão executados após a implementação da solução e qual será a sua metodologia de avaliação.

Ao longo deste capítulo será dada a resposta às seguintes questões:

1. Que testes serão realizados?
2. Que metodologia a adotar para avaliar a solução?
3. Quais os critérios a utilizar na avaliação?
4. Quais os resultados dos testes?

6.1 Testes

Nesta dissertação foi estudada a integração de uma *framework* de IAM no sistema da organização, com a intenção de utilizar esta *framework* para fazer a gestão dos utilizadores e das suas permissões de acessos. O objetivo desta secção especificar os testes que serão executados para avaliar a fiabilidade do sistema implementado, tendo em conta que os testes serão executados num ambiente o mais próximo de um ambiente real.

6.1.1 Grandezas a Avaliar

Nesta subsecção são apresentadas as grandezas a avaliar para validar a solução implementada. Foram então definidas as seguintes grandezas:

- Tempo de geração do *token*;
- Tempo de validação do *token*;
- Escalabilidade da solução implementada.

A grandeza do tempo da geração do *token* tem como objetivo avaliar o tempo que demora o *token* a ser gerado desde o pedido de autenticação do utilizador até este receber o *token*.

A grandeza do tempo de validação do *token* tem como objetivo comparar o tempo que demora um pedido a um serviço *web* de uma aplicação sem validação do *token* comparado com o mesmo pedido, mas com a validação do *token*.

A grandeza da escalabilidade da solução implementado tem como objetivo perceber a capacidade de o sistema suportar um grande número de interações na geração e validação dos *token* e que impacto tem no número de respostas com sucesso e no tempo de resposta.

6.1.2 Hipóteses

No âmbito deste projeto, foi definido que se pretende testar a eficiência das funcionalidades implementadas de geração e validação dos *tokens*, ou seja, pretende-se verificar que estas funcionalidades não interferem com o funcionamento geral das aplicações.

Para além de testar a eficiência, é necessário testar a fiabilidade da solução, ou seja, pretende-se verificar se a solução é capaz de suportar uma grande quantidade de interações num curto espaço de tempo, mantendo sempre a consistência dos dados e a capacidade de responder a todos os pedidos realizados pelos utilizadores com sucesso.

Para tal, foram identificadas as seguintes hipóteses:

- Tempo médio de geração do *token* tem de ser menor que 1 segundo;
- Tempo de validação do *token* não pode acrescentar mais do que 100 milissegundos, em média, ao pedido;
- A solução tem de suportar até 100 interações em simultâneo com 100% de respostas com sucesso.

6.1.3 Metodologia de Avaliação

A metodologia de avaliação tem como objetivo validar se as hipóteses definidas se verificam. Para tal, existem várias metodologias que dependem das hipóteses e das grandezas definidas. É então necessário escolher as melhores metodologias para a obtenção dos melhores resultados.

Para avaliar o tempo de geração dos *tokens*, serão elaborados vários testes onde será cronometrado o tempo de resposta a cada um dos pedidos. Quanto mais rápido for o tempo de resposta, melhor é o resultado.

Para avaliar o tempo de validação dos *tokens*, serão elaborados vários testes onde será cronometrado o tempo médio de resposta de pedidos a um recurso de uma aplicação sem a validação do *token* e o tempo médio de resposta ao mesmo recurso, mas com a validação do *token*. Por fim, será avaliada a diferença entre estes tempos.

No caso da escalabilidade da solução, será avaliado a sustentabilidade e tolerância a falhas da solução implementada com recurso a uma ferramenta que permita realizar um número elevado de pedidos consecutivos a uma aplicação.

É preciso ter em conta que a recolha destes dados será feita em ambientes controlados, recorrendo a máquinas com algum poder computacional e que apenas estejam a ser utilizadas para a execução destes testes.

6.1.4 Testes de hipóteses

Os testes de hipóteses consistem na utilização de testes estatísticos para facilitar a tomada de decisão entre duas ou mais hipóteses existentes.

No caso da avaliação do tempo de geração do *token*, pretende-se avaliar se o tempo médio de geração do *token* é inferior a 1 segundo. Para tal, a hipótese 1 (H0) é a hipótese que se pretende rejeitar, ou seja, o tempo médio ser igual ou superior a 1 segundo. A hipótese 2 (H1) é a hipótese que se quer que se verifique, ou seja, o tempo médio seja inferior a 1 segundo.

$$H0 : \mu >= 1seg \text{ (Hipótese 1)}$$

$$H1 : \mu < 1seg \text{ (Hipótese 2)}$$

No caso da avaliação do tempo de validação do *token*, pretende-se avaliar se o tempo acrescentado a um serviço por esta validação, em média, não é superior a 100 milissegundos. Para tal, a hipótese 1 (H0) é a hipótese que pretende-se rejeitar, ou seja, o tempo médio ser superior a 100 milissegundos. A hipótese 2 (H1) é a hipótese que se quer que se verifique, ou seja, o tempo médio seja igual ou inferior a 100 milissegundos.

$$H0 : \mu > 100ms \text{ (Hipótese 1)}$$

$$H1 : \mu <= 100ms \text{ (Hipótese 2)}$$

Por último, no caso da escalabilidade da solução implementada, pretende-se avaliar se o número de resposta com sucesso no teste seja de 100%. Para tal, a hipótese 1 (H0) é a hipótese que se pretende rejeitar, ou seja, existam alguns erros nas respostas aos pedidos e, portanto, a percentagem seja inferior a 100%. A hipótese 2 (H1) é a hipótese que se quer que se verifique, ou seja, que todas as respostas tenham sucesso e, portanto, a percentagem seja igual a 100%.

$$H0 : \mu < 100\% \text{ (Hipótese 1)}$$

$$H1 : \mu = 100\% \text{ (Hipótese 2)}$$

6.2 Resultados

Nesta secção serão apresentados das experiências realizadas para avaliar as hipóteses definidas. Através da análise dos resultados, é possível validar a solução apresentada e averiguar se os objetivos foram atingidos ou a necessidade de mais trabalho futuro. Na execução das experiências, foi utilizado a ferramenta SoapUI [40]

que permitiu a execução de vários serviços *web* em simultâneo e o cálculo do tempo de resposta.

6.2.1 Tempo geração do *token*

O pretendido do teste de geração do *token* é averiguar o tempo, em média, que demora todo o processo de autenticação. Desde o pedido de *login*, validação das credenciais, geração do *token* e devolução do *token*.

Para tal, foram executados cem pedidos sequenciais de geração do *token*. Os resultados podem ser visualizados na Tabela 26.

Tabela 26 – Resultados de cem execuções de geração do *token*

Mínimo (ms)	Máximo (ms)	Média (ms)
125	176	163,37

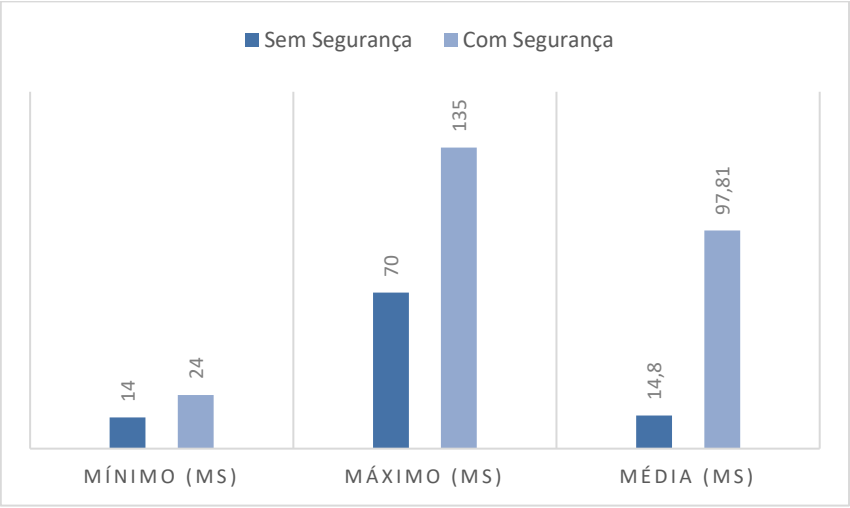
Com base os resultados da Tabela 26 e segundo o teste de hipóteses definido na Secção 6.1.4, a hipótese que se verifica é a hipótese 2 (H1), que significa que o tempo médio de todo o processo de autenticação é feito em menos de um segundo.

6.2.2 Tempo validação do *token*

No teste de validação do *token*, pretende-se averiguar a diferença existente no tempo de resposta de um serviço *web* com a validação do *token* e de um serviço *web* sem qualquer tipo de validação de autorização.

Para tal, foram executados cem pedidos sequencias para cada serviço *web* e foi feita a recolha dos seus tempos de resposta. Os resultados podem ser visualizados na Tabela 27.

Tabela 27 – Resultados de cem execuções da validação do *token*

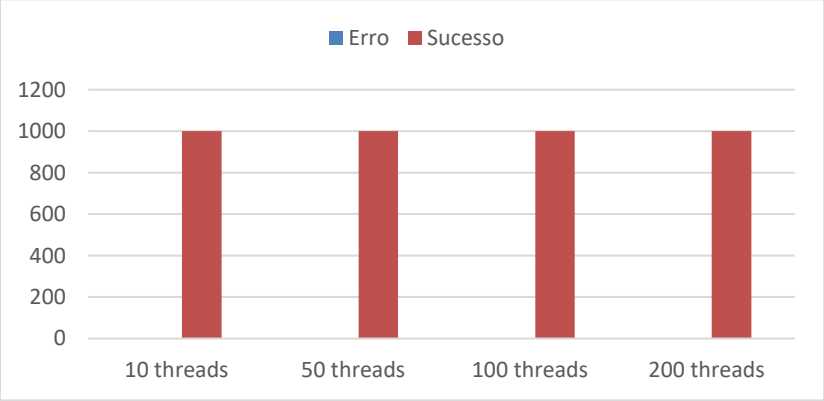


Como é possível visualizar na Tabela 27, os tempos de resposta do serviço sem segurança são bastante inferiores aos do serviço com segurança. No entanto, segundo o teste de hipóteses definido na Secção 6.1.4, a hipótese que se verifica é a hipótese 2 (H1), que significa que o tempo médio da validação do *token* não é superior a 100 milissegundos relativamente ao serviço sem a validação do *token*.

6.2.3 Escalabilidade da solução

No teste de escalabilidade da solução pretende-se perceber se a solução implementada aguenta uma grande carga de pedidos em simultâneo. Para tal, foram executados quatro testes para cada um dos serviços a testar, o de geração do *token* e o de validação do *token*. Os quatro testes são a execução de 1000 pedidos com 10, 50, 100 e 200 *threads* em execução.

Tabela 28 – Resultados de 1000 execuções da geração do *token*



Analisando a Tabela 28, verifica-se que em todos os testes executados houve 100% de sucesso na resposta dos pedidos para geração do *token*. Assim, segundo o teste de hipótese definido na Secção 6.1.4, a hipótese que se verifica é a hipótese 2 (H1).

Tabela 29 – Resultados de 1000 execuções da validação do *token*

Tal como no serviço de geração do *token*, os testes executados no serviço onde há validação do *token*, houve 100% de sucesso na resposta aos pedidos executados.

Concluí-se então que, segundo o teste de hipótese definido na Secção 6.1.4, a hipótese que se verifica é a hipótese 2 (H1).

6.3 Sumário

Neste capítulo foram descritos os testes que serão efetuados após a implementação da solução. Definiram-se duas grandezas a avaliar que permitiram validar a fiabilidade, sustentabilidade e tolerância a falhas deste desenvolvimento. Ficou também esclarecido que estes testes serão executados num ambiente controlado o mais próximo possível de um ambiente real.

No início deste capítulo foi proposto que fosse respondida à seguinte questão:

1. Que testes serão realizados?

R: Os testes realizados têm com base validar a fiabilidade e sustentabilidade da solução e para tal serão executados testes que permitam recolher dados relativos à velocidade de resposta das funcionalidades implementadas e da tolerância a falhas.

2. Que metodologia a adotar para avaliar a solução?

R: Os testes realizados serão executados num ambiente controlado e serão feitos três testes diferentes. Um para averiguar a média do tempo de geração dos *token* desde o pedido do utilizador até este receber a resposta. Um para averiguar o tempo médio acrescentado ao pedido com a adição da validação do *token*. Por último, validar a tolerância a falhas do sistema de modo a validar se com um elevado número de interações com sistema, este consegue responder com sucesso a todos os pedidos.

3. Quais os critérios a utilizar na avaliação?

R: Os critérios utilizados para cada teste são diferentes. Para o teste de validar o tempo de geração dos *tokens*, pretende-se que o tempo médio de geração seja inferior a 1 segundo. Para o teste validar o tempo de validação do *token*, pretende-se que o tempo médio de validação seja igual ou inferior a 100 milissegundos. Por fim, para o teste de escalabilidade da solução, pretende-se que o sistema consiga responder com sucesso a 100% dos pedidos efetuados.

4. Quais os resultados dos testes?

R: Em todos os testes obteve-se resultados com sucesso. O tempo de geração do *token* foi inferior ao tempo máximo pré-definido. O tempo de validação do *token* é praticamente impercetível na chamada de serviços *web*. Por fim, nos testes de escalabilidade provaram que a solução aguenta várias chamadas em simultâneo ao serviço de geração e de validação do *token*.

7 Conclusão

Neste capítulo é apresentada uma síntese do documento, uma análise aos objetivos e requisitos inicialmente identificados, as possíveis melhorias e limitações da solução implementada e é feita uma apreciação do trabalho realizado e quais os benefícios que o desenvolvimento deste projeto trouxe à i2S.

7.1 Síntese

Este documento tem como objetivo descrever o processo conducente à implementação de uma *framework* IAM ao produto já existente. As etapas passadas incluem a identificação e descrição do problema, análise e avaliação das *frameworks*, mais conhecidas, existentes no mercado, o relato das possíveis abordagens e todas as etapas que conduziram à implementação da solução produzida.

O problema abordado neste documento, surge na necessidade da i2S de implementar um sistema externo que permita fazer a gestão dos utilizadores e das suas permissões. Consequentemente permitir a autenticação dos utilizadores e da validação de autorização ao acesso aos serviços disponibilizados pelas aplicações.

Para chegar à solução final, foi feita uma análise entre três abordagens encontradas. A primeira tinha como base a utilização de adaptadores fornecidos pelas *framework* IAM, no entanto, esta abordagem trazia o problema que a solução ficaria presa à *framework* escolhida e a sua mudança implicaria desenvolvimento extra. A segunda abordagem era o desenvolvimento e melhoria da ferramenta já existente, mas esta necessitaria de um grande esforço que não era possível. A terceira e última abordagem é a utilização dos serviços *web* disponibilizados pela *framework* IAM para fazer a gestão das autenticações e autorizações.

Esta última abordagem foi a escolhida, com o apoio do modelo AHP através de vários critérios por permitir uma maior agilidade na implementação e, através de desenvolvimento, permitir a utilização dos modos de autenticação e autorização já implementados pela organização.

7.2 Objetivos

A solução final, é assim capaz de fazer a gestão das autenticação e autorizações utilizando uma *framework* IAM externa que seja compatível com o protocolo OAuth para a geração, validação e atualização do *token*. Após os testes realizados sobre a solução implementada, conseguiu-se também concluir que a solução implementada consegue aguentar um número elevado de sobrecarga mantendo-se responsivo sem falhas.

Após a conclusão deste projeto foi possível a realização de uma análise aos objetivos definidos anteriormente. Esta análise engloba os objetivos definidos na Secção 1.3 e os requisitos funcionais e não funcionais definidos na Secção 4.1.

A metodologia de trabalho utilizada ao longo do desenvolvimento deste projeto permitiu criar um sistema que permite a autenticação e validar a autorização utilizando uma *framework* IAM externa. Com este sistema foi possível a execução de teste que ajudaram na validação dos objetivos.

Através da Tabela 30 é possível visualizar os requisitos definidos e os seus graus de tangibilidade.

Tabela 30 – Nível de tangibilidade dos requisitos

Objetivo	**Nível de tangibilidade**
1.1. Registo de aplicações	Atingido
1.2. Registo de perfis aplicacionais	Parcialmente atingido
1.3. Atribuição de perfis aplicacionais a utilizadores e ou grupos	Atingido
1.4. Gestão de *tokens*	Atingido
1.5. Implementação do *Refresh Token*	Atingido
1.6. Interface simplificado e intuitivo apenas para a gestão dos utilizadores que não exponha a complexidade da *framework* e das suas restantes funcionalidades	Não atingido
1.7. Implementação em *cluster* Ativo/Ativo	Atingido
1.8. As tarefas de validação de autorização têm de ser impercetíveis	Atingido
1.9. O número de *tokens* a ser gerido num momento pode ser de vários milhares	Parcialmente atingido
1.10. Integração via LDAP com Windows Active Directory e ou IBM Tivoli Directory Server em AS400 (*Read Only*)	Atingido
1.11. *Web* API para ser acedido pelas aplicações para gestão das identidades e dos *tokens*	Atingido
1.12. Manter os modos de autenticação e autorização já utilizados na organização	Atingido
1.13. Suporte a servidores aplicacionais, de preferência WebSphere e Wildfly	Atingido
1.14. Suporte a base de dados RDBMS como Oracle, SQLServer e DB2 for i	Atingido

Pode-se visualizar que praticamente todos os requisitos foram atingidos com sucesso à exceção de três. Inicialmente, o pretendido seria o registo automático das aplicações, mas atualmente é um processo manual e pretendia-se a gestão simultânea de milhares de *token* o que só foi possível testar algumas centenas. Relativamente ao requisito da criação de um interface mais simplificado para a gestão dos utilizadores, não foi possível executar esta tarefa devido a atrasos no projeto.

Relativamente aos objetivos definidos, é possível visualizá-los através da Tabela 31 e os seus respetivos graus de tangibilidade.

Tabela 31 – Nível de tangibilidade dos objetivos

Objetivo	Nível de tangibilidade
Analisar as *frameworks* de IAM existentes no mercado e identificar as suas especificidades e funcionalidades para obter uma visão global das soluções	Atingido
Analisar o processo de IAM existente na solução atual	Atingido
Definir os requisitos que a solução implementada terá de ter	Atingido
Elaborar um estudo para selecionar qual das frameworks de IAM analisadas anteriormente se completa melhor os requisitos definidos	Atingido
Analisar as alterações que serão necessárias fazer para implementar a *framework* selecionada	Atingido
Desenhar uma possível solução final, tendo em conta a necessidade de alta disponibilidade do sistema	Atingido
Implementação do novo processo IAM utilizando a solução selecionada	Atingido
Avaliar a solução implementada, tendo em conta a alta disponibilidade, performance e tolerância a falhas	Atingido

Pode-se concluir que todos os objetivos foram atingidos com sucesso tendo assim analisado as *frameworks* IAM mais conhecidas no mercado, analisada a solução existente e os requisitos necessários para a implementação do novo processo IAM e a sua implementação executando testes para validar a sua alta disponibilidade, performance e tolerância a falhas.

7.3 Limitações e trabalho futuro

Apesar do esforço feito ao longo do desenvolvimento deste projeto para atingir os objetivos e requisitos inicialmente definidos, houve algumas limitações que fizeram com

que algumas das tarefas que estavam a ser executadas tivessem que ser suspensas, o que levou a que estas excedessem o tempo pré-definido.

Foi durante a fase de implementação da solução que se originaram maiores desvios de tempo. Estes deveram-se à mudança de prioridades nos desenvolvimentos a decorrer na i2S, o que levou a que este projeto ficasse para segundo plano e que levou a que alguns dos requisitos definidos não ficassem completos ou por completar. No entanto, foi possível desenvolver uma solução que atingiu com sucesso os principais objetivos definidos inicialmente.

Apesar do sucesso geral do projeto e da sua implementação, algumas funcionalidades não foram desenvolvidas de modo a concluí-lo definitivamente.

Como já referido, uma das funcionalidades não desenvolvidas foi a implementação do interface simplificado para a gestão dos utilizadores para não expor a complexidade da *framework*. Esta funcionalidade seria uma mais-valia para o utilizador final pois a interface que lhe seria apresentada apenas continha as funcionalidades necessárias e utilizadas pela nossa solução.

Outras funcionalidades já pensadas e que trariam uma mais valia seriam a gestão de permissões mais finas, ou seja, permissões que permitam restringir o acesso a funcionalidades das diferentes aplicações e a restrição da visualização de dados específicos de negócio. Esta funcionalidade é importante para o produto pois permite adicionar mais uma camada de segurança às aplicações e assim protegendo ainda mais os dados que elas contêm.

7.4 Crítica

Com o desenvolvimento deste projeto, verificou-se que várias empresas se dedicam a desenvolver *frameworks* IAM. No entanto, estas soluções têm especificidades diferentes, funcionalidades mais desenvolvidas que outras e maneiras de implementações diferentes. Com a análise efetuada, verificou-se que para este projeto a melhor implementação seria a utilização da API disponibilizada pelas *frameworks* devido às necessidades impostas inicialmente.

Relativamente ao planeamento inicialmente definido, ocorreram alguns desvios anteriormente descritos, que levaram a que o desenvolvimento da solução se atrasasse. No entanto, foi possível terminar o projeto com sucesso apesar do atraso.

Considerando o trabalho executado ao longo deste projeto, a análise das ferramentas e dos requisitos e posterior desenvolvimento e testagem, considera-se que o trabalho cumpriu todos os objetivos. A experiência foi considerada positiva, não só pelo ganho de conhecimento na área de IAM, mas também pelo que trouxe à i2S.

Bibliografia

[1] T. Nguyen, S. Cuttill, T. T. Nguyen e M. Mahdavi, “IDENTITY AND ACCESS MANAGEMENT,” pp. 1-6, 31 Jan 2008.

[2] M. Gaedke, J. Meinecke e M. Nussbaumer, “A modeling Approach to Federated Identity and Access Management,” pp. 1156-1157, 2005.

[3] K. Tracy, “Identity management systems,” pp. 34-37, 2006.

[4] I. Indu, P. R. Anand e V. Bhaskar, “Identity and access management in cloud environment: Mechanisms and challenges,” pp. 574-588, Agosto 2018.

[5] i2S, “Grupo Gfi compra i2S para construir solução completa para companhias de seguros,” [Online]. Available: https://i2s.pt/pt-pt/recursos/comunicados-de-imprensa/grupo-gfi-compra-i2s-para-construir-solucao-completa-para-companhias-de-seguros/. [Acedido em 11 Janeiro 2021].

[6] Cleva Insurance Solution, “i2S vence prémio PME Inovação COTEC,” [Online]. Available: https://i2s.pt/pt-pt/noticias/i2s-vence-premio-pme-inovacao-cotec/. [Acedido em 2021 Fevereiro 22].

[7] M. A. Thakur, T. J. Parvat e V. S. Walunj, “Data Security Using Directory Server,” pp. 73-84, 2020.

[8] auth0, “Token Based Authentication Made Easy,” [Online]. Available: Token Based Authentication Made Easy. [Acedido em 9 Fevereiro 2021].

[9] M. Kunz, A. Puchtaa, S. Groll, L. Fuchs e G. Pernul, “Attribute Quality Management for Dynamic Identity,” pp. 1-36, 13 Novembro 2018.

[10] P. Koen, G. Ajamian, R. Burkart, A. Clamen, J. Davidson, R. D’Amore, C. Elkins, K. Herald, M. Incorvia, A. Johnson, R. Karol, R. Seibert, A. Slavejkov e K. Wagner, “PROVIDING CLARITY AND A COMMON LANGUAGE TOTHE “FUZZY FRONT END”,” pp. 46-51, 27 Janeiro 2016.

[11] Pershing e J. A., “Handbook of Improving Performance in the Workplace,” pp. 1089-1090, 2009.

[12] R. Vargas, “Using the analytic hierarchy process (ahp) to select and prioritize projects in a portfolio,” Outubro 2010. [Online]. Available:

https://www.pmi.org/learning/library/analytic-hierarchy-process-prioritize-projects-6608. [Acedido em 26 Fevereiro 2021].

[13] M. Kelley, A. Data e H. Teixeira, “Magic Quadrant for Access Management,” 17 Novembro 2020. [Online]. Available: https://www.gartner.com/doc/reprints?id=1-24MIGFGK&ct=201119&st=sb. [Acedido em 9 Fevereiro 2021].

[14] OAuth0, “Make Authentication Even More Personal,” OAuth0, [Online]. Available: https://auth0.com/authentication. [Acedido em 10 Fevereiro 2021].

[15] OAuth0, “Access Management,” OAuth0, [Online]. Available: https://auth0.com/access-management. [Acedido em 10 Fevereiro 2021].

[16] OAuth0, “Frictionless experiences for all,” OAuth0, [Online]. Available: https://auth0.com/user-management. [Acedido em 10 Fevereiro 2021].

[17] ForgeRock, “Gestão de Acesso,” ForgeRock, [Online]. Available: https://www.forgerock.com/platform/access-management. [Acedido em 10 Fevereiro 2021].

[18] ForgeRock, “Evaluation Guid - ForgeRock Access Management 7.0.1,” 7 Janeiro 2021. [Online]. Available: https://backstage.forgerock.com/docs/am/7/AM-7-Eval-Guide.pdf. [Acedido em 10 Fevereiro 2021].

[19] IBM, “Securely connect any user to any resource,” IBM, [Online]. Available: https://www.ibm.com/security/digital-assets/iam/verify-demo-trial/#/. [Acedido em 13 Fevereiro 2021].

[20] Microsoft, “Azure Active Directory,” Microsoft, [Online]. Available: https://azure.microsoft.com/pt-pt/services/active-directory/#features. [Acedido em 13 Fevereiro 2020].

[21] Microsoft, “Integração do Azure Active Directory,” Microsoft, [Online]. Available: https://azure.microsoft.com/pt-pt/services/active-directory/integrate/. [Acedido em 13 Fevereiro 2020].

[22] onelogin, “OneLogin ™ SSO Identity Management,” onelogin, [Online]. Available: https://www.onelogin.com/lp/demo?v=af&headline=OneLogin%E2%84%A2+SSO+Identity+Management&_bt=381997188390&_bk=onelogin&_bm=e&_bn=g&utm_source=GOOGLE&utm_medium=cpc&gclid=CjwKCAiAsaOBBhA4EiwAo0_AnHHx3aUWzISpBxVXrVFZI2dmb4TqURLVmFo3rLT1oiXxyBh2q3ku0RoCx7kQA. [Acedido em 14 Fevereiro 2021].

[23] Ping Identity, “PingFederate,” Ping Identity, [Online]. Available: https://www.pingidentity.com/en/resources/client-library/data-sheets/pingfederate-data-sheet.html. [Acedido em 14 Fevereiro 2021].

[24] Keycloak, “About,” Keycloak, [Online]. Available: https://www.keycloak.org/about. [Acedido em 14 Fevereiro 2021].

[25] B. R. Koubský, “Extend WildFly Application Server logging capabilities,” 2016.

[26] “IBM WebSphere Application Server,” IBM, [Online]. Available: https://www.ibm.com/cloud/websphere-application-server. [Acedido em 14 Janeiro 2021].

[27] K. Ueno, T. Alcott, J. Blight, J. Dekelver, D. Julin, C. Pfannkuch e T. Shieh, “WebSphere Scalability,” *WebSphere Scalability: WLM and Clustering,* 2000.

[28] R. Johnson, J. Hoeller, K. Donald, C. Sampaleanu, R. Harrop, T. Risberg, A. Arendsen, D. Davison, D. Kopylenko, M. Pollack, T. Templier, E. Vervaet, P. Tung, B. Hale, A. Colyer, J. Lewis e Co, “Spring Framework Reference Documentation,” 2004.

[29] Duda e B. Erich, “Implement WS-Transfer specification for Apache CXF framework,” 2015.

[30] Apache CXF, “Apache CXF Software Architecture Guide,” Apache, [Online]. Available: http://cxf.apache.org/docs/cxf-architecture.html. [Acedido em 1 Fevereiro 2021].

[31] K. Shingala, “JSON Web Token (JWT) based client authentication in Message Queuing Telemetry Transport (MQTT),” 2019.

[32] D. Hardt, “A estrutura de autorização OAuth 2.0,” Outubro 2012. [Online]. Available: https://tools.ietf.org/html/rfc6749. [Acedido em 2 Fevereiro 2021].

[33] B. Brito, “OAuth 2.0 - Guia do Iniciante,” 9 Janeiro 2019. [Online]. Available: https://www.brunobrito.net.br/oauth2/. [Acedido em 2 Fevereiro 2021].

[34] onelogin Developers, “Overview of SAML,” onelogin Developers, [Online]. Available: https://developers.onelogin.com/saml. [Acedido em 13 Fevereiro 2020].

[35] IBM, “O Que é um Certificado Digital,” [Online]. Available: https://www.ibm.com/docs/pt-br/ibm-mq/9.0?topic=ssfksj-9-0-0-com-ibm-mq-sec-doc-q009830--htm. [Acedido em 3 Junho 2021].

[36] Mitchell, W. Li e C. J, “Analysing the Security of Google’s implementation of OpenID Connect,” em *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment.*, Springer, Cham, 2016.

[37] P. Eeles, “Capturing Architectural Requirements,” IBM Rational Developer Works, 2001.

[38] Visual Paradigm, “What is Use Case Diagram?,” Visual Paradigm, [Online]. Available: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/. [Acedido em 16 Fevereiro 2021].

[39] connect2id, “Nimbus JOSE + JWT,” connect2id, [Online]. Available: https://connect2id.com/products/nimbus-jose-jwt. [Acedido em 8 Setembro 2021].

[40] SoapUI, “The Industry’s #1 API Testing Tool,” SoapUI, 2021. [Online]. Available: https://www.soapui.org/tools/soapui/. [Acedido em 21 9 2021].

[41] C. A. Gunter, D. Liebovitz e B. Malin, “Experience-Based Access Management: A Life-Cycle Framework for Identity and Access Management Systems,” *IEEE Security & Privacy,* vol. 9, nº 5, pp. 48-55, 2011.