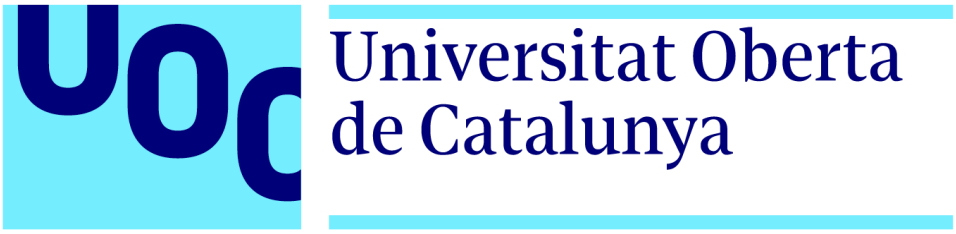

Implantación de un SSO en un entorno empresarial

TRABAJO FINAL DEL MÁSTER UNIVERSITARIO EN CIBERSEGURIDAD Y PRIVACIDAD

Seguridad empresarial

Jesús Gutiérrez de la Vega

Tutor del TFM
Antoni González Ciria

Profesor responsable de la asignatura
Víctor García Font

Fecha entrega
13/06/2023

Agradecimientos:

Quisiera agradecer y dedicar este trabajo a todas aquellas personas que durante mi carrera académica y profesional han estado enseñándome, apoyándome, guiándome y compartiendo su experiencia para llegar a ser el profesional que soy hoy en día. También agradecer a todas aquellas personas que, sin cuya paciencia, dedicación y apoyo recibido, no podría haber conseguido esta meta.

FICHA DEL TRABAJO FINAL

Título del trabajo	Implantación de un SSO en un entorno empresarial
Nombre del autor	Jesús Gutiérrez de la Vega
Nombre del consultor/a	Antoni González Ciria
Nombre del PRA	Víctor García Font
Fecha de entrega (mm/aaaa):	06/2023
Titulación o programa	Máster Universitario en Ciberseguridad y Privacidad
Área del Trabajo Final	Seguridad empresarial
Idioma del trabajo	Español
Palabras clave	SSO, Keycloak, Docker

Resumen del trabajo

El presente trabajo de fin de máster analiza la implantación de un sistema de autenticación centralizada en el ámbito de una empresa del sector TIC. Este trabajo viene dado tras la detección de problemas en la productividad de la plantilla de la empresa debido el gran número de autenticaciones que tiene que hacer el personal durante su jornada, lo cual disminuye notablemente su eficiencia. Del mismo modo, se necesita una solución que garantice y mejore la seguridad de los accesos, cumpliendo así las directrices legales vigentes. Por todo ello, se determinó que la mejor solución era el desarrollo de un proyecto para la implantación de un SSO Corporativo, para el cual se siguió un modelo de desarrollo tradicional en cascada, el cual ha permitido obtener en su primera fase un listado de requisitos del sistema de las personas implicadas y posteriormente la implementación de la solución. Tras la implantación y configuración de la solución se ha comprobado la mejoría de la productividad y seguridad de los accesos en la compañía.

Abstract

The present Master's thesis analyzes the implementation of a centralized authentication system within a company of the ICT sector. This work is motivated by the detection of productivity problems among the company's staff due to the large number of authentications they have to perform during their workday, which significantly decreases their efficiency. Similarly, a solution is needed to ensure and improve access security, thus complying with current legal guidelines. Therefore, it was determined that the best solution was to develop a project for the implementation of a Corporate SSO system, following a traditional waterfall development model. This approach allowed obtaining, in its first phase, a list of system requirements from the individuals involved, followed by the implementation of the solution. After the implementation and configuration of the solution, the improvement in productivity and access security in the company has been verified.

Índice

Lista de figuras

Lista de tablas

1. Introducción

1.1. Contexto y justificación del trabajo

El presente TFM aborda la implantación de un sistema de autenticación centralizado y único, en adelante SSO, en un entorno empresarial.

Actualmente trabajo como coordinador de equipos en una importante empresa andaluza dedicada a la consultoría y desarrollo de aplicaciones en el sector de las energías renovables y de las Administraciones Públicas.

En el día a día, trabajamos con una decena de aplicaciones, tanto de desarrollo propio como desarrolladas por terceras empresas. Las características de estas aplicaciones son bastante heterogéneas, dado que se tienen aplicaciones en la nube (SaaS), servidores dedicados en el CPD propio, así como aplicaciones basadas tanto en Software Libre como en Software Comercial.

Tras algunas reuniones de la Dirección, se decidió que, dentro del plan estratégico de la empresa, se implantaría un SSO corporativo con el objetivo de centralizar los inicios de sesión en las aplicaciones por parte de todo el personal y, al mismo tiempo, reducir las veces que el mismo gasta tiempo en dicha operación.

La implementación de esta medida fue aceptada por la Dirección al comprobarse que cada persona empleada realizaba en su jornada laboral entre 8 y 10 autenticaciones en las diferentes aplicaciones. Estimando que en cada autenticación se tarda entre 15 y 20 segundos en introducir su identificador de acceso y su contraseña (suponiendo que las personas usuarias no cometen errores en el proceso) y que actualmente la empresa cuenta con algo más de 300 personas en la plantilla, al día se gastan aproximadamente 13 horas y media en tareas repetitivas, pudiendo extrapolarse a casi 300 horas mensuales.

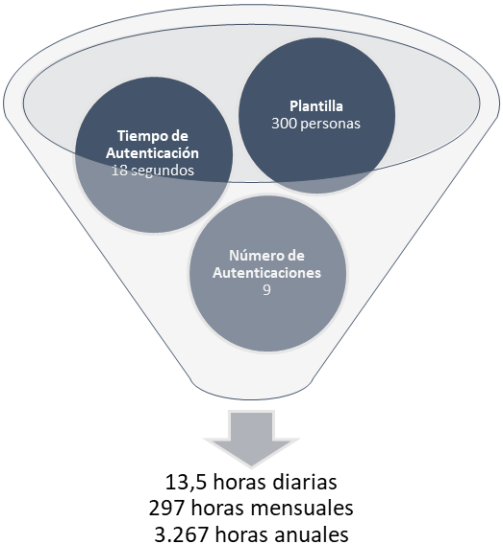

Figura 1: Estimación de tiempo medio empleado

1.2. Objetivos del trabajo

A continuación, se enumeran los objetivos principales del TFM:

- Realización de un análisis de requisitos que detalle las necesidades de los diferentes estamentos de la empresa:
  - Personas usuarias finales.
  - Personal TIC.
  - Dirección de la empresa.
- Desarrollo del estado del arte, donde se contemplen las ventajas e inconvenientes de las principales soluciones. Del mismo modo plantear la adaptación y personalización de plataformas existentes, así como la realización de alguna plataforma a medida.
- Desarrollo del plan de viabilidad, incluyendo un análisis de riesgos, de costes de desarrollo e implantación, así como de retorno de la inversión.
- Desarrollo, adaptación e implantación del SSO elegido en un entorno de laboratorio.
- Desarrollo de un "*Kit de Integración*" para la futura adaptación de las herramientas corporativas.
- Integración de una o varias aplicaciones piloto con el SSO previamente desplegado.
- Documentar todo el proceso en la memoria del propio TFM.

1.3. Impacto en sostenibilidad, ético-social y de diversidad

A continuación, se detallan los impactos que tiene el TFM en relación con las competencias transversales del "Compromiso ético y global" [1].

Figura 2: Objetivos de desarrollo sostenible.

1.3.1. Dimensión sostenibilidad

El desarrollo de un SSO, y más concretamente la implantación y puesta a disposición de las personas usuarias, implica que el mismo esté desplegado en algún servidor de

aplicaciones (físico o virtual). Como norma general, se ubican en pequeños o grandes centros de datos (dependiendo la capacidad, infraestructura de la empresa y/o delegación de recursos que esta haga).

Los centros, aunque necesarios hoy por hoy para la sociedad, están desde algunos años en el punto de mira del mundo debido a que tradicionalmente han sido muy ineficientes y contribuidores del cambio climático [2] [3] [4]. Los principales problemas que tienen son:

- El equipamiento de los centros de datos genera con su funcionamiento una gran cantidad de calor, el cual debe ser refrigerado dado que estas máquinas están pensadas para funcionar de forma óptima a unos 24 grados centígrados de media.
- El equipamiento y los sistemas de refrigeración consumen una gran cantidad de electricidad, la cual a su vez, dependiendo de su fuente de obtención, puede tener un gran impacto.
- Los residuos que se generan en el sector, tanto durante la creación de los componentes usados en los equipos informáticos, como en la fase de desuso de los mismo.

Para mitigar este problema, en el caso de utilizar centros de datos externos, se debe escoger una compañía que esté comprometida con la sostenibilidad de sus CPD. De igual forma, si se cuenta con CDP propio, este se debe adecuar a las buenas prácticas que garanticen un consumo sostenible.

En el caso de la empresa donde se instalará el software, cuenta con un centro de datos propio, el cual está certificado bajo la norma ISO 50600:2016. Por otro lado, dado que el principal sector al que se decida la empresa son las energías renovables, desde hace algunos años todo el techo de sus edificios se cubrió de placas solares que aportan la mayor parte de la energía consumida en los mismos, incluyendo el CPD (durante las horas de sol, apoyándose en el exceso acumulado en baterías y consumiendo lo imprescindible de la propia red eléctrica).

Objetivos de desarrollo sostenible relacionados:

- *ODS 9 – Industria, innovación e infraestructura.*
- *ODS 12 – Producción y consumo responsables.*

1.3.2. Dimensión comportamiento ético y de responsabilidad social (RS)

Desde el punto de vista de los aspectos éticos y sociales, el software que se utilice para la implantación de la solución SSO elegida tendrá como factor clave en su elección la licencia que utilice el mismo. Esto es debido a que, desde hace muchos años, la empresa está concienciada con el uso de tecnologías de código abierto, siempre que las posibilidades y su clientela permitan su utilización.

La preferencia de este tipo de licencias de software libre no exime del uso de software comercial en algunas ocasiones, siempre que el mismo se haga de una forma racional y justa, valorando las capacidades y comparándolas con las opciones de software libre existentes.

Del mismo modo, la empresa procura ser patrocinadora del software libre, organizando y liderando eventos, realizando donaciones y siendo abanderada de su uso en toda la organización.

Objetivo de desarrollo sostenible relacionado:

- *ODS 8 – Trabajo decente y crecimiento económico.*

1.3.3. Dimensión diversidad, género y derechos humanos

Desde el punto de vista de la diversidad, género y derechos humanos, dado que la empresa está comprometida con sectores tradicionalmente discriminados, como las personas con discapacidad, el software que se realice debe cumplir con los requisitos necesarios para que cualquier persona de la organización, sea cual sean sus condiciones físicas, psíquicas o psicológicas pueda utilizar el mismo, o en el caso de que este no pudiera adaptarse, habilitar una opción accesible [5].

Objetivo de desarrollo sostenible relacionado:

- *ODS 10 – Reducción de las desigualdades.*

1.4. Enfoque y método seguido

El ciclo de trabajo que se seguirá durante el TFM será un modelo tradicional en cascada [6] [7] [8]. A continuación, se exponen las ventajas y desventajas de este modelo:

Ventajas

- **Estructura clara**: el modelo de cascada tiene una estructura clara y fácil de seguir, ya que cada etapa se completa antes de pasar a la siguiente, lo que ayuda a la planificación y el control del proyecto.

- **Documentación**: el modelo requiere una documentación exhaustiva en cada etapa del proyecto, lo que garantiza la trazabilidad y la transparencia del proceso de desarrollo.

- **Adecuado para proyectos pequeños**: dado que en los proyectos pequeños y bien definidos los requisitos no cambian con frecuencia, este modelo es adecuado estos casos.

- **Facilita la detección de errores en una fase**: cada etapa del modelo de cascada se centra en una actividad específica, lo que facilita la detección temprana de errores y su corrección antes de pasar a la siguiente etapa.

- **Mejora la calidad del software**: se centra en la calidad y la documentación del propio software, lo que garantiza que el software generado tenga una alta calidad y mantenibilidad.

Desventajas

- **Falta de flexibilidad**: el modelo de cascada es muy poco flexible en sus etapas, por lo que no se adapta bien a los cambios en los requisitos del proyecto una vez iniciado.

- **Retrasos en la detección de errores de desarrollo**: dado que las fases no se empiezan hasta que termina la anterior, errores de software o de arquitectura pueden aparecer de forma muy tardía, lo que podría ser más difíciles y costosos de corregir.

- **No es adecuado para proyectos complejos**: este modelo no es adecuado para proyectos complejos que requieren un enfoque iterativo o incremental.

- **Poca participación de las personas responsables**: el modelo de cascada no fomenta la participación de las personas responsables del proyecto en el proceso de desarrollo, dado que principalmente interviene solo en la toma de requisitos. Esto puede dar lugar que el software no se adapte a las necesidades reales de las personas usuarias.

1.5. Planificación del trabajo

Para la planificación del proyecto se ha realizado un diagrama de Gantt donde se puede ver la división de tareas por entregas:

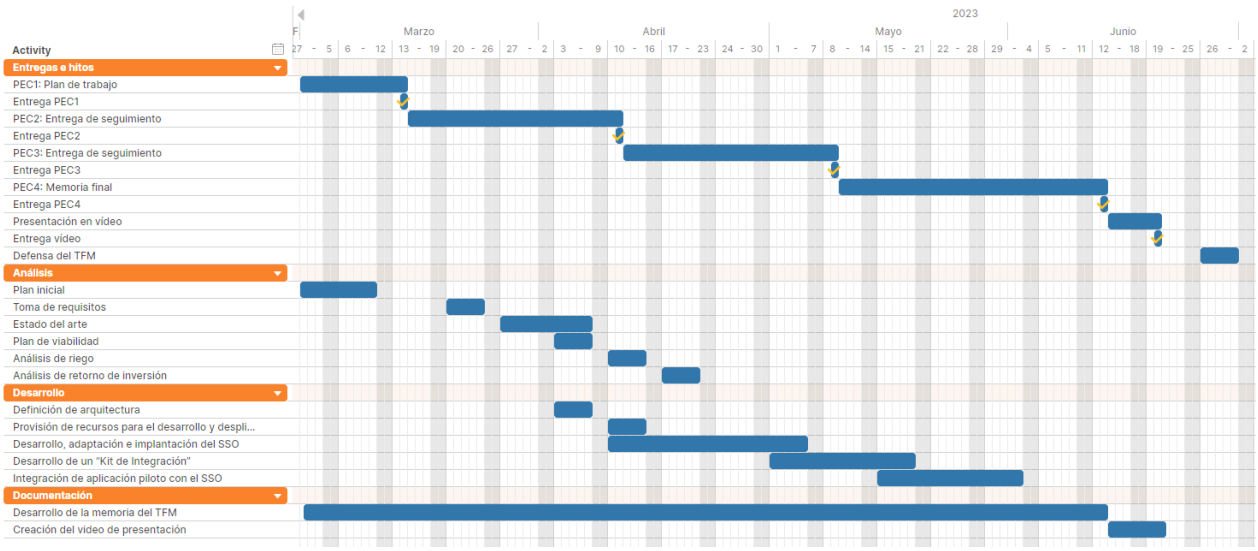

Figura 3: Planificación temporal del proyecto

A continuación, se detalla el calendario de hitos y el contenido de estos:

PEC1. Plan de Trabajo: 14 de marzo de 2023

En la primera entrega se presenta el plan de trabajo del proyecto, incluyendo las necesidades detectadas, la calendarización y resultados esperados en el TFM.

Figura 4: Planificación temporal PEC1

PEC2. Entrega de seguimiento: 11 de abril de 2023

En la primera entrega de seguimiento se recogerá la mayor parte del análisis de la solución, donde se incluirá los siguientes puntos:

- Toma de requisitos.
- Estado del arte.
- Plan de viabilidad.

Durante esta entrega también se realizará la definición de la arquitectura del sistema.

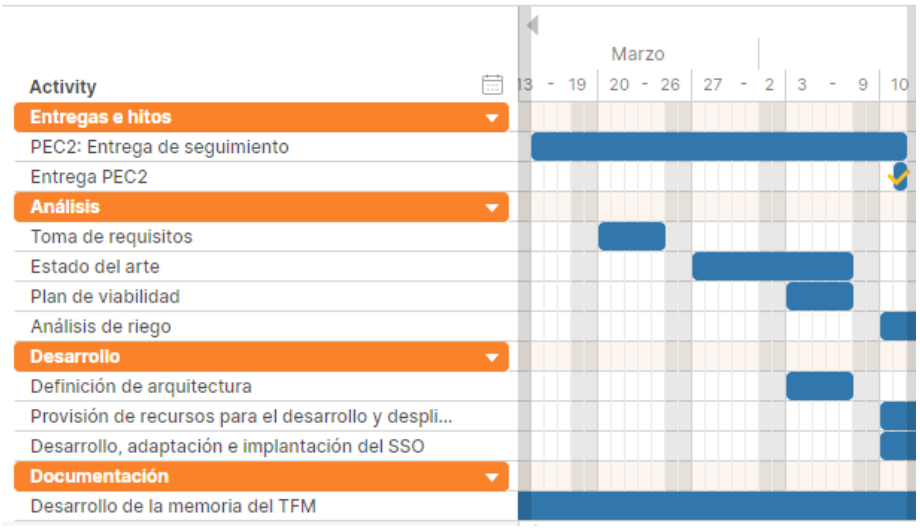

Figura 5: Planificación temporal PEC2

PEC3. Entrega de seguimiento: 9 de mayo de 2023

En la segunda entrega de seguimiento, además de refinar los puntos anteriores, se añadirá el resto de documentación relacionada con el análisis (riesgos y retorno de inversión).

Se realizará la provisión de recursos para el desarrollo y despliegue y se realizará el desarrollo, adaptación e implantación del SSO.

Figura 6: Planificación temporal PEC3

PEC4. Entrega de memoria final: 13 de junio de 2023

Como en las entregas anteriores, además del refinamiento, se realizará el resto de puntos relacionadas con el desarrollo (kit de integración e interacción de aplicación piloto).

Además, se integrará toda la documentación en la memoria del proyecto revisando todos los aspectos formales de cara a la entrega de esta.

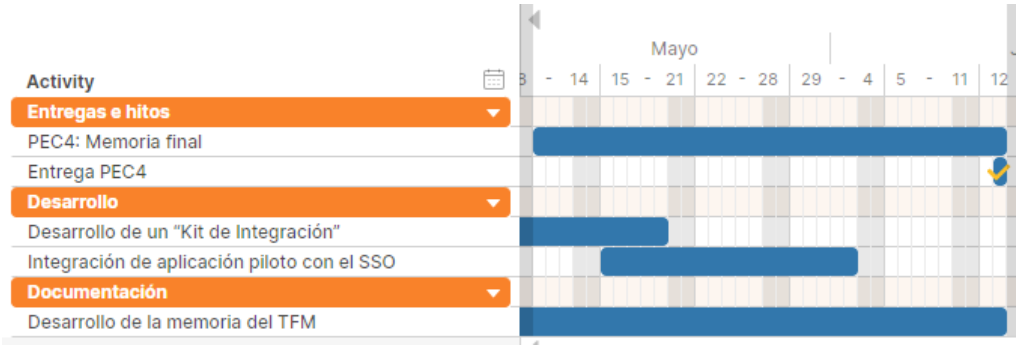

Figura 7: Planificación temporal entrega final

Entrega de video de presentación: 20 de junio de 2023

En esta entrega se realizará la grabación de la presentación, mostrando el software implementado en funcionamiento.

2. Análisis de la solución

2.1. Estado del arte

Los sistemas centralizados de autenticación son una tecnología madura y ampliamente utilizada en la actualidad para la gestión de identidades y accesos en todos los ámbitos empresariales.

Estos sistemas se pueden inicialmente clasificar entre:

- **Cloud** donde el sistema de autenticación se ejecuta en servidores y redes de computadoras que son propiedad y están administrados por proveedores de servicios en la nube. En este tipo de clasificación incluiremos los SaaS (*Software as a Service*) y los PaaS (*Platform as a Service*).

- **On-Premise** donde el sistema de autenticación se ejecuta en servidores y redes de que son propiedad y/o están administrados por la organización que utiliza el sistema.

Por otro lado, podemos clasificarlos entre **sistemas propietarios** y **software libre**.

Las principales diferencias entre ambos son:

- **Accesibilidad del código fuente**: en los sistemas propietarios, el código fuente no suele estar disponible para los equipos de desarrollo, mientras que en los sistemas de software libre el código fuente si está disponible y puede ser modificado y distribuido.

- **Licenciamiento**: los sistemas propietarios suelen tener una propiedad exclusiva y las personas o empresas que lo usen deben pagar para obtener licencias de uso, mientras que los sistemas de software libre, como norma general, son gratuitos y suelen tener licencias de código abierto.

- **Flexibilidad**: los sistemas de software libre, al tener el código publicado y disponible para los equipos de desarrollo, son más flexibles ya que pueden ser modificados para satisfacer las necesidades específicas, mientras que los sistemas privativos tienen características predefinidas y a priori no suelen poder modificarse.

- **Comunidad de desarrollo**: los sistemas de software libre suelen contar con una amplia comunidad, así como equipos de desarrollo, que pueden contribuir al implementación y mejora del sistema. En cambio, en los sistemas propietarios, su comunidad suele ser más reducida y limitada, o más orientada al soporte en vez del desarrollo.

- **Soporte técnico**: en los sistemas de software libre, el soporte técnico suele ser proporcionado por la propia comunidad y equipos de desarrollo implicados, mientras que en los sistemas privativos el soporte técnico suele ser proporcionado por el propio fabricante del software y puede tener un costo adicional.

A continuación, se detallan algunas de las soluciones comerciales y de software libre más utilizadas en la actualidad:

Sistemas propietarios

Okta es una plataforma de identidad y acceso en la nube que proporciona soluciones de SSO y gestión de identidad para empresas. Utiliza varios protocolos de intercambio de credenciales, como SAML, OAuth 2.0, OpenID Connect, WS-Federation y SCIM, para permitir la autenticación y autorización de accesos a través de aplicaciones web y móviles. Aunque Okta no es un software libre, ofrece una prueba gratuita y planes de suscripción para adaptarse a diferentes necesidades empresariales.

Microsoft Azure Active Directory es una solución de gestión de identidades y accesos basada en la nube desarrollada por Microsoft. Azure AD proporciona servicios de autenticación y autorización a aplicaciones web y móviles, así como a servicios en la nube de Microsoft, como Office 365, Dynamics CRM Online y Azure. Además, también permite la gestión de identidades para aplicaciones SaaS y personalizadas tanto la nube como en on-premise. Azure AD admite varios protocolos de autenticación y autorización, como SAML, OAuth, OpenID Connect y WS-Federation.

Google Cloud Identity es una solución de gestión de identidad y acceso en la nube ofrecida por Google. Se utiliza para autenticar y autorizar personas usuarias en aplicaciones y servicios en la nube. Este SSO admite una variedad de protocolos de autenticación, entre los que están OpenID Connect, OAuth 2.0, SAML y LDAP. Además, Google Cloud Identity ofrece una amplia gama de funciones de gestión de identidad, como la asignación de permisos, la creación de grupos, el seguimiento de personas usuarias y la autenticación multifactor.

Onelogin es una solución de SSO basada en la nube que proporciona autenticación segura y gestión de identidades para entidades y empresas. Este SSO es compatible con una variedad de protocolos de autenticación, como SAML, OAuth y OpenID Connect, lo que permite la integración con aplicaciones empresariales y de terceros. Además, Onelogin ofrece funciones de gestión de acceso y seguridad, como autenticación multifactor, control de acceso basado en roles y auditoría de registros.

ForgeRock es una solución de identidad y acceso que ofrece una amplia variedad de herramientas de gestión de identidades y acceso para las personas, dispositivos y aplicaciones. ForgeRock ofrece una solución integral y modular que permite a las organizaciones personalizar y escalar fácilmente su infraestructura de autenticación y autorización. La plataforma de ForgeRock soporta una amplia variedad de protocolos de autenticación y autorización, incluyendo SAML, OAuth, OpenID Connect, LDAP y más. Además, ForgeRock también cuenta con un enfoque en la seguridad y privacidad, lo que permite a las personas mantener el control de su información privada y su privacidad.

Sistemas de software libre

Keycloak es una solución de Single Sign-On de código abierto, lo que significa que es libre de usar, modificar y distribuir. Proporciona una plataforma para autenticar y autorizar a las personas usuarias en aplicaciones web y móviles. Con Keycloak, las personas pueden acceder a varias aplicaciones usando un conjunto único de credenciales, lo que elimina la necesidad de recordar múltiples contraseñas. Keycloak admite una amplia variedad de protocolos de autenticación y autorización, incluyendo SAML, OpenID Connect y OAuth2, lo que lo hace altamente interoperable con otros sistemas y aplicaciones. Además, Keycloak ofrece una amplia gama de características, incluyendo autenticación multifactor, registro de eventos de seguridad y gestión de personas usuarias y roles. Keycloak está escrito en lenguaje Java y se despliega habitualmente sobre un contenedor de aplicaciones Wildfly.

Shibboleth es una aplicación comúnmente utilizada en entornos académicos y gubernamentales como SSO codificada en Java. Shibboleth utiliza el protocolo SAML para la autenticación y autorización de las personas. Además, también es compatible con otros protocolos de autenticación, como CAS y OAuth. Shibboleth es una solución de SSO altamente configurable y personalizable que se puede integrar con diferentes tipos de aplicaciones web. Además, Shibboleth también proporciona una gestión de identidades y acceso a recursos, permitiendo la administración de las identidades de las personas usuarias en un entorno federado.

Central Authentication Service (CAS) es una aplicación de código abierto escrita principalmente en Java, que proporciona servicios de autenticación única para aplicaciones web. Se basa en el protocolo de autenticación y autorización JA-SIG Central Authentication Service (CAS) y es compatible con múltiples protocolos de autenticación, incluyendo SAML y OAuth. CAS funciona como un intermediario entre los proveedores de identidad y las aplicaciones, permitiendo a las personas acceder a múltiples aplicaciones con una sola identidad y contraseña. Además, CAS incluye

características de seguridad como la encriptación de tokens y el bloqueo de sesiones, lo que garantiza una experiencia de autenticación segura y confiable.

Gluu es una plataforma de gestión de identidad y acceso que proporciona soluciones de SSO para aplicaciones web y móviles. Utiliza varios protocolos de interacción, como OpenID Connect, OAuth 2.0 y SAML 2.0, para permitir la autenticación y autorización segura. Además de los servicios de SSO, Gluu también ofrece funciones de gestión de identidades, como la creación de perfiles de acceso, la gestión de contraseñas y la integración con múltiples proveedores de identidad externos. Con su arquitectura modular y de código abierto, Gluu se puede personalizar y extender para adaptarse a las necesidades específicas de una organización. La plataforma Gluu está escrita principalmente en Python, aunque también utiliza otros lenguajes de programación como Java y Javascript. Además, hace uso de diversos frameworks y tecnologías de código abierto, como Flask, Pyramid, jQuery y AngularJS, entre otros.

LemonLDAP::NG es una aplicación multiplataforma, diseñada para integrarse en una gran variedad de arquitecturas de autenticación y autorización. Esta aplicación está escrita en el lenguaje de programación Perl y utiliza varios protocolos de interacción, entre ellos SAML, OAuth, OpenID Connect, CAS y Radius, para permitir la integración con diferentes servicios de identidad y proveedores de autenticación. LemonLDAP::NG proporciona una interfaz de usuaria intuitiva para gestionar y configurar los diversos aspectos de la autenticación, como la autenticación multifactor, la autorización, el cifrado de sesión y la protección contra ataques de fuerza bruta. Además, permite a los grupos de administración crear políticas de acceso personalizadas para los diferentes roles de acceso, lo que facilita la administración de las aplicaciones y servicios que requieren autenticación. Gracias a su código abierto, LemonLDAP::NG es una opción ideal para aquellas organizaciones que deseen utilizar una solución SSO de alta calidad y personalizable sin tener que pagar altas licencias de software.

OpenAM es una solución de gestión de identidades que permite a las organizaciones gestionar la autenticación, autorización y federación de accesos en sus aplicaciones web y móviles. OpenAM utiliza una variedad de protocolos de intercambio de credenciales, incluyendo SAML, OAuth, OpenID Connect, CAS y LDAP, lo que proporciona una gran flexibilidad a la hora de integrar con diferentes sistemas y aplicaciones. Además, OpenAM es multilenguaje de forma nativa, lo que la hace accesible a una audiencia global. OpenAM está escrito en Java y ofrece características avanzadas de seguridad, como la autenticación multifactor, la gestión de contraseñas y la auditoría, lo que la convierte en una solución robusta y completa para la gestión de identidades y accesos en una amplia variedad de entornos empresariales.

Para una mayor claridad, se resumen los sistemas mencionados en la siguiente tabla:

	Software libre	Cloud	On-premise	Protocolos de comunicación
Okta	☒	☑	☒	SAML, OpenID Connect, OAuth 2.0, WS-Fed, SWA
Microsoft Azure Active Directory	☒	☑	☒	SAML 2.0, OpenID Connect, OAuth 2.0, WS-Fed, Kerberos
Google Cloud Identity	☒	☑	☒	SAML, OpenID Connect, OAuth 2.0, Active Directory
Onelogin	☒	☑	☒	SAML, OAuth 2.0, OpenID Connect, WS-Fed,
ForgeRock	☒	☑	☒	SAML 2.0, OAuth 2.0, OpenID Connect, CAS Protocol, Kerberos
Keycloak	☑	☑	☑	OpenID Connect, OAuth 2.0, SAML 2.0, CAS Protocol, Kerberos, JWT
Shibboleth	☑	☑	☑	SAML, OpenID Connect, OAuth 2.0, WS-Fed, CAS Protocol, Kerberos
CAS	☑	☑	☑	CAS Protocol, SAML, OAuth, OpenID
Gluu	☑	☑	☑	OpenID Connect, OAuth 2.0, SAML, CAS Protocol y UMA
LemonLDAP::NG	☑	☑	☑	SAML, OpenID Connect, OAuth 2.0, CAS Protocol, Kerberos, Radius
OpenAM	☑	☑	☑	SAML 2.0, OAuth 2.0, OpenID Connect, RESTful APIs

Tabla 1: Comparativa de SSO

2.2. Análisis de requisitos

El análisis de requisitos es un paso crucial en el desarrollo de software y de sistemas, ya que permite comprender las necesidades y expectativas de las personas usuarias y establecer una base sólida para la planificación y el diseño del proyecto [9].

Para llevar a cabo el análisis de requisitos, se ha empleado una metodología donde se ha involucrado a las partes interesadas relevantes, incluyendo a la dirección de la empresa, una representación del área TIC y a una selección de la plantilla como personas usuarias finales del sistema.

Para la confección de los requisitos, se han utilizado entrevistas personales, cuestionarios y análisis documental preexistente, para recopilar y analizar la información determinar las necesidades y requisitos del sistema.

2.2.1. Requisitos funcionales

Identificador	Descripción
SSO-FUN-01	Las personas usuarias de la organización deben poder autenticarse de forma centralizada en un SSO.
SSO-FUN-02	El SSO dispondrá de una pantalla de autenticación, donde la persona usuaria utilizará sus credenciales de acceso a los sistemas corporativos.
SSO-FUN-03	El SSO mantendrá activo el acceso de la persona usuaria por un tiempo limitado, el cual quedará definido por los requisitos de seguridad correspondientes.
SSO-FUN-04	Las aplicaciones de la organización deben comprobar en el SSO si la persona se ha autenticado previamente: • Si lo hubiera realizado ya, recuperará sus credenciales. • Si no lo hubiera hecho, redirigirá a la persona a la pantalla de autenticación del SSO.
SSO-FUN-05	Las aplicaciones corporativas informarán al SSO de forma periódica si la persona usuaria aún se encuentra trabajando en dicha aplicación.
SSO-FUN-06	El SSO debe contar con un kit de integración para las diferentes aplicaciones corporativas, teniendo en cuenta que la organización cuenta con un parque de aplicaciones heterogéneo.

Tabla 2: Requisitos funcionales

2.2.2. Requisitos no funcionales

En el ámbito del desarrollo de software, los requisitos no funcionales son tan importantes como los requisitos funcionales, ya que definen la calidad, el rendimiento y la seguridad de un sistema. Los requisitos no funcionales se refieren a las características y cualidades del software que no están relacionadas directamente con las funcionalidades principales del sistema, sino con su comportamiento, calidad y rendimiento.

El objetivo es garantizar que el software cumpla satisfaga las necesidades de la organización y de las personas usuarias finales en cuestiones tan diversas como la calidad, el rendimiento o la seguridad.

Ian Sommerville [10], un reconocido experto en ingeniería de software clasifica los requisitos no funcionales en seis categorías principales:

- **Requisitos de calidad**: estos requisitos se refieren a las características que definen la calidad del software, como la confiabilidad, la seguridad, la facilidad de mantenimiento y la usabilidad.

- **Requisitos de rendimiento**: estos requisitos se refieren a las medidas de rendimiento que deben cumplir el software, como la velocidad, la capacidad, la escalabilidad y la eficiencia.

- **Requisitos de compatibilidad**: estos requisitos se refieren a la capacidad del software para funcionar en diferentes entornos, sistemas operativos, navegadores y dispositivos.

- **Requisitos de interoperabilidad**: estos requisitos se refieren a la capacidad del software para interoperar con otros sistemas y aplicaciones, a través de protocolos y estándares.

- **Requisitos de seguridad**: estos requisitos se refieren a las medidas de seguridad que deben adoptarse para proteger la información y los datos del software, garantizando la confidencialidad, integridad y disponibilidad.

- **Requisitos legales y normativos**: estos requisitos se refieren a los requisitos legales y normativos que deben cumplir el software, como las leyes de privacidad de datos, la accesibilidad y las regulaciones de la industria.

Esta clasificación proporciona un marco útil para identificar y definir los requisitos no funcionales de un sistema de software y garantizar que se satisfagan las necesidades y expectativas de las personas usuarias finales y de las partes interesadas.

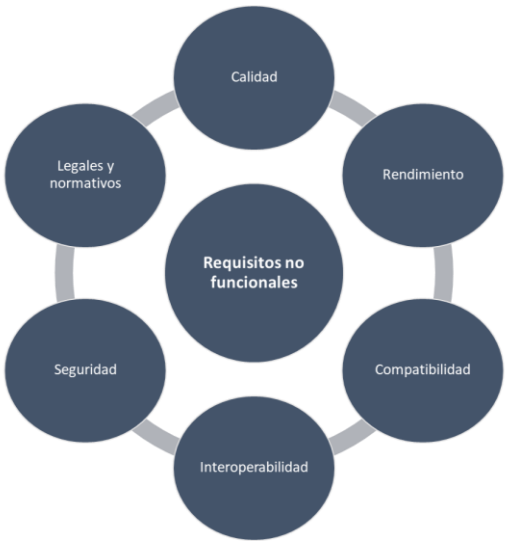

Figura 8: Clasificación de requisitos no funcionales

2.2.2.1. Requisitos de usabilidad e interfaz

Identificador	Descripción
SSO-USA-01	El SSO debe ser fácil de usar y comprender para las personas usuarias, siendo su interfaz intuitiva y fácil de usar.
SSO-USA-02	El SSO debe ser accesible para personas con discapacidad, cumpliendo para ello con el estándar de accesibilidad "*Web Content Accessibility Guidelines (WCAG)*", al menos en su nivel AA.
SSO-USA-03	La interfaz de persona usuaria debe ser consistente en su diseño, utilizando para ello los elementos y estilos corporativos.

SSO-USA-04	El SSO debe ser adaptable a diferentes dispositivos (responsive), como ordenadores personales, tabletas digitales o teléfonos móviles.
SSO-USA-05	El SSO debe proporcionar una retroalimentación adecuada a las personas usuarias, indicando el estado de la autenticación de una forma clara.

Tabla 3: Requisitos no funcionales de usabilidad

2.2.2.2. Requisitos de seguridad

Identificador	Descripción
SSO-SEG-01	El SSO debe ser seguro y debe proteger la información privada de las personas usuarias.
SSO-SEG-02	El SSO debe utilizar protocolos de autenticación y cifrado que actualmente se consideren seguros.
SSO-SEG-03	El sistema debe contar con un registro de seguridad y de auditoría, que permita monitorizar las acciones de las personas usuarias y detectar posibles violaciones de seguridad.
SSO-SEG-04	El sistema debe contar con un mantenimiento periódico que garantice contar con las actualizaciones y parches de seguridad que corrijan las vulnerabilidades conocidas.

Tabla 4: Requisitos no funcionales de seguridad

2.2.2.3. Requisitos de entorno

Identificador	Descripción
SSO-ENT-01	El sistema SSO debe proporcionar un tiempo de carga mínimo que garantice que las personas usuarias puedan realizar su trabajo de una forma óptima.
SSO-ENT-02	El sistema debe ser escalable, teniendo la capacidad para manejar un creciente número de autenticaciones por segundo.
SSO-ENT-03	El sistema debe estar accesible en todo momento, permitiendo con ellos una configuración de alta disponibilidad.
SSO-ENT-04	El sistema debe funcionar correctamente y sin fallos.

Tabla 5: Requisitos no funcionales de entorno

2.2.2.4. Requisitos de legales

Identificador	Descripción
SSO-LEG-01	Los sistemas de software implicados deben cumplir con la regulación legislativa española y de la Unión Europea. Entre otras, se debe tratar las siguientes:

	• Reglamento (UE) 2016/679 del Parlamento Europeo y del Consejo, de 27 de abril de 2016, relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos. • Ley Orgánica 3/2018 de Protección de Datos Personales y garantía de derechos digitales.

Tabla 6: Requisitos no funcionales legales

2.3. Plan de gestión del cambio

La implantación del SSO en la organización significará un cambio significativo en la forma en que la plantilla accede a los sistemas y aplicaciones corporativas. Por lo tanto, es importante abordar la gestión del cambio de manera efectiva para garantizar una transición fluida y una adopción exitosa.

Para ello se seguirán los siguientes pasos:

1. **Identificar el impacto del cambio**: es necesario identificar y comunicar el impacto del cambio en la organización y en las personas usuarias individuales. Esto implica establecer claramente los objetivos, beneficios y limitaciones del SSO, así como los cambios que experimentarán el personal en la forma de acceder a las aplicaciones.

2. **Identificar a las personas interesadas**: es importante identificar a todos las personas interesadas del cambio, incluyendo al personal de la plantilla, el personal TIC y la propia dirección de la empresa. Cada grupo puede tener diferentes necesidades y preocupaciones, por lo que es importante abordarlos de manera individualizada.

3. **Desarrollar un plan de comunicación**: se debe desarrollar un plan de comunicación que aborde los objetivos marcados, los beneficios que aporta y cómo las personas se verán afectadas. Este plan debe incluir la frecuencia, el canal y los mensajes de comunicación, y debe ser comunicado a todas las personas interesadas.

4. **Proporcionar capacitación y soporte**: es clave proporcionar capacitación y soporte a las personas afectadas para que comprendan cómo utilizar el nuevo sistema y puedan solucionar los posibles problemas si surgiesen. Esto puede incluir la creación de materiales de formación y soporte, así como la realización de sesiones de divulgación.

5. **Medir y evaluar el éxito**: se debe medir y evaluar el éxito de la implementación del SSO para identificar oportunidades de mejora y garantizar que se logren los objetivos. Esto puede incluir la realización de encuestas de satisfacción, el seguimiento del uso y el monitoreo de los problemas informados durante su uso.

2.4. Análisis de riesgos del desarrollo

El análisis de riesgos es un proceso crítico en la implantación de cualquier sistema software, dado que permite anticiparse a posibles problemas. Por ello, es importante identificar los riesgos potenciales que pueden afectar a la seguridad y la operación del sistema y desarrollar un plan de mitigación adecuado.

Algunos riesgos que pueden surgir durante la implantación de este sistema son:

- **Vulnerabilidades de seguridad**: los sistemas de autenticación son objetivos interesantes para los hackers, dado que un fallo en la seguridad de estos podría permitirles acceder a todos los sistemas con los que esté conectado. Por ello, es importante realizar una evaluación exhaustiva de las vulnerabilidades y debilidades potenciales del software, así como el desarrollo de planes de mitigación adecuados.

- **Problemas de integración**: la integración del sistema con el resto del parque software existente puede ser complicada y puede generar problemas de compatibilidad. Es importante identificar los riesgos potenciales asociados con la integración y desarrollar planes para abordar cualquier problema que surja.

- **Problemas de escalabilidad**: el sistema debe ser capaz de manejar el posible crecimiento futuro la interconexión con nuevas aplicaciones. Es importante evaluar la escalabilidad del sistema diseñando medidas adecuadas para ello.

- **Problemas de disponibilidad**: se debe evitar que el sistema sea un punto de fallo único (SPOF), garantizando que el mismo esté disponible en todo momento, dado que la falta de disponibilidad podría interrumpir el trabajo diario y crear un problema de productividad en la plantilla.

Por ello se propone la siguiente tabla de riesgos con su criticidad y plan de mitigación inicial:

Riesgo	Probabilidad	Criticidad	Plan de mitigación
Vulnerabilidades de seguridad	Media	Alta	Análisis de vulnerabilidades del sistema y test de penetración.
Problemas de integración	Baja	Baja	Análisis del parque software a integrar y maximizar las opciones de conectividad.
Problemas de escalabilidad	Baja	Alta	Test de rendimiento y estrés durante el desarrollo y una vez se implante en el entorno preproductivo.
Problemas de disponibilidad	Media	Alta	Habilitar el sistema en alta disponibilidad.

Tabla 7: Criticidad de riesgos del desarrollo

2.5. Evaluación de viabilidad económica

Uno de los puntos más importantes de cualquier proyecto es evaluar si económicamente es viable el mismo. Para ello, se debe estudiar varios puntos claves:

- **Costes de desarrollo**: en este punto se debe identificar y calcular todos los costes asociados con el análisis, desarrollo e implantación del sistema, incluyendo hardware, licencias de software, personal de desarrollo, capacitación y consultoría. Además, se deben considerar los posibles costes de migración de datos y de la integración con los sistemas existentes.

- **Escalabilidad y mantenimiento**: es importante evaluar la escalabilidad del sistema, estudiando el crecimiento y posibles cambios futuros para identificar como repercutirá en el coste mensual/anual del sistema. Además, se deben considerar los gastos de mantenimiento continuo, incluyendo actualizaciones de software, soporte técnico y posibles mejoras o expansiones del sistema.

- **Beneficios económicos**: quizás uno de los puntos más importantes es conseguir identificar y cuantificar los beneficios económicos que se obtendrán con la implementación del sistema. Estos beneficios pueden incluir ahorro de costes operativos, aumento de la productividad, reducción de errores, mejora en la toma de decisiones o aumento de la satisfacción del cliente.

- **Retorno de la inversión**: un indicador económico muy interesante es el ROI, el cual permite calcular el período de tiempo requerido para recuperar la inversión inicial en el sistema. Esto se logra comparando los costes totales del desarrollo y mantenimiento del sistema con los beneficios económicos esperados.

- **Análisis de riesgos económicos**: de cara a prevenir posibles problemas, se debe realizar un análisis de riesgos potenciales que podrían afectar la viabilidad económica del sistema. Esto incluye riesgos como cambios en las condiciones del mercado, fluctuaciones en los costes de hardware o software, cambios en los requisitos del sistema y riesgos de seguridad. Se deben desarrollar estrategias de mitigación de riesgos para minimizar su impacto.

2.5.1. Análisis de costes de desarrollo

Los principales puntos a tener en cuenta para poder realizar adecuadamente el análisis de costes de implementación son los siguientes:

- **Hardware y software**: se deben evaluar los costes relacionados con la adquisición de hardware, como servidores, equipos de red y dispositivos periféricos. Asimismo, es necesario tener en cuenta las licencias de software requeridas para la instalación y configuración del sistema.

- **Personal y capacitación**: se deben estimar los costes asociados con el personal necesario para la implementación, incluyendo analistas, desarrolladores y personal de soporte. Además, es fundamental considerar la capacitación requerida para que el personal adquiera las habilidades necesarias para trabajar con el sistema.

- **Consultoría externa**: en algunos casos, puede ser necesario contratar servicios de consultoría externa para ayudar en la implementación del sistema. Esto implica costes adicionales que deben ser considerados en el análisis.

- **Migración de datos**: si el sistema requiere la migración de datos desde sistemas existentes, es importante evaluar los costes asociados con la extracción, transformación y carga de datos en el nuevo sistema.

- **Infraestructura**: además del hardware, se deben considerar los costes relacionados con la infraestructura necesaria para soportar el sistema, como sistemas de refrigeración, energía y seguridad.

- **Personalización y configuración**: dependiendo de los requisitos específicos del sistema, pueden surgir costes adicionales relacionados con la personalización y configuración del software para adaptarlo a las necesidades de la organización.

Tipo	Detalle	Importe
Hardware	Dado que el sistema se alojará en el CPD de la organización, los precios que se pagan por el alojamiento de los servicios son internos a la propia organización. El coste por máquina está establecido en 50 € y se estima necesario 3 máquinas costeadas durante un mes.	150 €
Licencia software	Debido al compromiso que tiene la organización con el software libre, se asume que los costes en este punto será 0 €.	0 €
Personal de desarrollo	Se estima que será necesario que haya un equipo de desarrollo y mantenimiento del sistema, el cual inicialmente tendrá una dedicación alta y posteriormente solo de forma residual. El equipo inicialmente se dimensionará con una persona especialista en sistemas (con dedicación del 20%) y dos personas especialistas en desarrollo (con dedicación del 75%). El coste de este equipo se calculará según el convenio laboral vigente usando perfiles A1 (retribución persona trabajadora → 26.790 €, coste empresa → 35.403 €).	5.015 €
Migración de datos	No se prevé inicialmente la necesidad de migración de datos.	0 €
Infraestructura	Los costes de infraestructura están incluidos en los costes mensuales del hardware.	0 €

Tabla 8: Costes de desarrollo

Según la tabla anterior, se prevé un coste para el desarrollo e implantación inicial del sistema de 5.165 €. Hay que tener en cuenta que en este importe no está incluido el coste de explotación final del sistema, el cual se añade en el siguiente apartado

2.5.2. Análisis de costes de escalabilidad y mantenimiento del sistema

El análisis de los costes de escalabilidad y mantenimiento del sistema SSO es fundamental para evaluar la viabilidad económica a largo plazo. Estos costes están directamente relacionados con la capacidad del sistema de adaptarse al crecimiento y evolución de la organización, así como con los gastos asociados al mantenimiento continuo del sistema.

En primer lugar, es importante evaluar los costes de **escalabilidad del sistema**, lo cual implica considerar los recursos necesarios para aumentar la capacidad del sistema y garantizar un rendimiento óptimo a medida que crece la cantidad de personas usuarias, aplicaciones y transacciones que interaccionarán con el sistema. Los costes de escalabilidad pueden incluir la adquisición de nuevos servidores, el aumento de almacenamiento, la ampliación de la infraestructura de red y la contratación de personal adicional para gestionar el crecimiento.

En cuanto a los **costes de mantenimiento del sistema** se deben considerar los gastos continuos asociados con la operación y gestión del sistema. Esto incluye los costes relacionados con el funcionamiento continuo del sistema, el soporte técnico, las actualizaciones de software, el monitoreo del sistema, las labores de mantenimiento preventivo y correctivo, así como los posibles costes de formación y capacitación del personal encargado de administrar el sistema. Además, es importante tener en cuenta los costes relacionados con la seguridad, como las auditorías y las medidas de protección de datos.

Para este apartado se consideran los siguientes puntos:

Tipo	Detalle	Importe
Hardware	Se estima que el anualmente el importe hardware seguirá siendo el mismo.	1.800 €
Personal de mantenimiento	Se estima que será necesario que haya un equipo de mantenimiento con una dedicación residual (se establece una dedicación del 15% mensual para compensar posibles picos de trabajo).	5.310 €

Tabla 9: Costes de escalabilidad y mantenimiento

2.5.3. Análisis de beneficios económicos

A continuación, se detallan los aspectos clave a considerar en el análisis de beneficios económicos:

- **Ahorro de costes operativos**: el sistema puede generar ahorros significativos en los costes operativos de la organización, como puede ser por ejemplo que, al centralizar la gestión de accesos y autenticación en un solo punto, se reduce la necesidad de mantener múltiples sistemas de credenciales, lo que implica un ahorro en costes de licencias, mantenimiento y soporte.

- **Aumento de la productividad**: uno de los objetivos de un SSO es crear eficiencia al agilizar el proceso de autenticación y acceso a múltiples

aplicaciones, lo que se traduce en un aumento de la productividad de las personas usuarias.

- **Mejora en la experiencia de la persona usuaria**: un SSO bien implementado proporciona una experiencia de personas usuaria más fluida y conveniente al permitir el acceso rápido y seguro a múltiples aplicaciones con una sola autenticación.

- **Reducción del riesgo de seguridad**: al implementar un sistema de SSO, se pueden establecer políticas de seguridad consistentes y reforzar los controles de acceso. Esto reduce el riesgo de violaciones de seguridad, robos de identidad y acceso no autorizado, lo que puede resultar en ahorros significativos en términos de costes de recuperación y reparación.

- **Facilitación de la auditoría y cumplimiento normativo**: un sistema de SSO bien diseñado proporciona un mejor control y trazabilidad de los accesos a las aplicaciones, lo que facilita las auditorías internas y externas.

La fórmula del beneficio económico puede variar dependiendo del enfoque y los factores considerados en el análisis. Sin embargo, una fórmula comúnmente utilizada para calcular el beneficio económico es la siguiente:

Beneficio económico = Ingresos totales - Costes totales

Donde podemos definir los términos de la siguiente forma:

- **Ingresos totales**: incluyen los ahorros, incrementos de ingresos o cualquier otro beneficio económico que se haya identificado en el análisis. Estos beneficios pueden ser expresados en términos monetarios o cualitativos.

- **Costes totales**: incluyen los costes directos e indirectos asociados con el desarrollo, operación y mantenimiento del sistema. Estos costes pueden incluir licencias de software, hardware, personal, capacitación, infraestructura, entre otros.

Si hacemos un ejercicio de estimación basándonos únicamente en el ahorro de costes operativos y mejora de productividad, y teniendo en cuenta la estimación inicial que se hacía en el punto 1.1, se utilizan una media de 3.267 horas anuales en la compañía en la autenticación del personal en las diferentes aplicaciones de la compañía.

Siguiendo el mismo ejercicio del punto anterior, calcularemos el coste de estas horas basándonos en el convenio colectivo, y asignando una media al importe que cobra anualmente cada persona de la plantilla a 24.640€ (categoría C1), lo cual se transforma en 32.561€ incluyendo los costes que paga la empresa por ella.

Por otro lado, se estima que cada mes se trabaja una media de 160 horas lo que da un total de 1.760 horas anuales (11 meses de trabajo anuales). Con estos datos, vemos que las 3.267 horas anuales destinadas a autenticación, cuestan aproximadamente 60.441€ al año a la empresa. Este importe es el que se estima como potencialmente ahorrable en el mejor de los casos (ingresos totales).

Con los datos obtenidos previamente podemos resumir en los siguientes datos:

Ingresos totales: 60.441 €
Costes totales primer año: 10.063 €

- Desarrollo:
  - Hardware: 150 €
  - Personal: 5.015 €
- Mantenimiento
  - Hardware: 1.800 €
  - Personal: 5.310 €

Beneficio económico = 60.441 - 10.063 = 50.378 €

2.5.4. Análisis de retorno de inversión

El ROI es una herramienta importante para evaluar la rentabilidad y viabilidad económica de cualquier proyecto. En el contexto en el que estamos trabajando el ROI permite determinar si la inversión realizada en el proyecto generará beneficios financieros a largo plazo o no lo hará.

El análisis del retorno de inversión implica comparar los costes totales de desarrollo, operación y mantenimiento del sistema con los beneficios esperados durante un período de tiempo específico. Para calcular el ROI, se utiliza la siguiente fórmula:

ROI = ((Ingresos totales - Costes totales) / Costes totales) x 100

El resultado del cálculo del ROI se expresa como un porcentaje, que indica el rendimiento económico relativo de la inversión. Un ROI positivo indica que los beneficios superan los costes y se considera una inversión rentable. Por otro lado, un ROI negativo indica que los costes superan los beneficios y puede requerir una revisión adicional del proyecto.

Si aplicamos la formula usando los datos anteriormente obtenidos, los cuales se han extrapolados a una anualidad, tenemos lo siguiente:

Ingresos totales: 60.441 €
Costes totales primer año: 10.063 €
ROI primer año = ((60.441 - 10.063) / 10.063) x 100 = 392%

Si tenemos en cuenta que en los años posteriores el coste de desarrollo en principio tendería a 0, podemos obtener los siguientes datos (suponiendo que no ha habido cambio en el resto de las magnitudes):

Ingresos totales: 60.441 €
Costes totales años posteriores: 7.110 €
ROI años posteriores: = ((60.441 - 7.110) / 7.110) x 100 = 750%

Según los datos obtenidos en este apartado y los anteriores ya vistos, el proyecto sería viable económicamente.

2.5.5. Análisis de riesgos económicos

El análisis de riesgos económicos es la última etapa del análisis de viabilidad económica, donde se busca pode identificar y evaluar los posibles riesgos que podrían

afectar a la economía del proyecto y tomar las medidas necesarias para prevenirlos o mitigarlos. Entre los riesgos más importantes se encuentran:

- **Fluctuaciones en los costes**: los costes asociados con el desarrollo e implementación, como podrían ser el coste de hardware, personal o la necesidad de soporte técnico, pueden verse afectados por fluctuaciones en el mercado. Es importante considerar las posibles variaciones de precios y tener un margen de contingencia para evitar desviaciones significativas en el presupuesto.

- **Cambios en los requisitos y alcance del proyecto**: los cambios en los requisitos o el alcance del proyecto pueden tener implicaciones económicas, como la necesidad de capacitar a personal adicional o realizar modificaciones en la infraestructura existente. Es esencial contar con un proceso de gestión del cambio adecuado que permita controlar y evaluar el impacto económico de los mismos.

- **Rendimiento y escalabilidad del sistema**: una implementación eficiente y escalable es fundamental para garantizar el funcionamiento óptimo del sistema y evitar así problemas de rendimiento. Si el sistema no puede manejar la carga de personas usuarias o presenta deficiencias en su rendimiento, podría ser necesario realizar inversiones adicionales en desarrollo, hardware o infraestructura, lo que afectaría los costes del proyecto.

- **Adopción y aceptación de la plantilla**: la adopción y aceptación del sistema por parte del resto de la plantilla es fundamental para el éxito del proyecto. Si estos no se adaptan o rechazan el nuevo sistema, podría haber una disminución en la eficiencia y productividad esperada, lo que afectaría sensiblemente los beneficios económicos proyectados. Es esencial implementar estrategias de comunicación, capacitación y cambio organizacional para fomentar la adopción y aceptación del nuevo sistema.

- **Cambios regulatorios y legales**: los cambios en las regulaciones y leyes relacionadas con la seguridad de la información y la protección de datos pueden tener un impacto económico significativo. Es importante estar al tanto de los posibles cambios en las regulaciones que afecten al sistema y a la organización asegurándose así que el nuevo sistema cumpla con los requisitos legales necesarios.

Riesgo	Probabilidad	Criticidad	Plan de mitigación
Fluctuaciones en los costes	Media	Media	Se reevaluará el plan de viabilidad si los costes de personal o materiales se ven incrementados.
Cambios en los requisitos y alcance del proyecto	Alta	Alta	Los cambios en el sistema deberán estar acotados a lo especificado en el análisis de requisitos. Cualquier cambio que desvíe sensiblemente el alcance (siempre que venga autorizado por la dirección de la organización), implicará la reestimación del proyecto.
Rendimiento y escalabilidad del sistema	Baja	Alta	Realización de pruebas de carga y adopción de alta disponibilidad en todos los recursos software afectados.

Adopción y aceptación de la plantilla	Baja	Alta	Desarrollo y aplicación del plan de gestión del cambio.
Cambios regulatorios y legales	Baja	Media	Realización de análisis temprano de los cambios legislativos que pudieran afectar al proyecto para ver su impacto, y reestimar si fuera necesario.

Tabla 10: Criticidad de riesgos económicos

3. Desarrollo e implementación

3.1. Solución elegida

Tras el estudio del estado del arte realizado previamente donde se pudo comparar las diferentes opciones de Single Sign-On, se ha llegado a la conclusión de que Keycloak era la mejor solución para implementar en la organización. Para esta elección, se han evaluado diferentes criterios como son la facilidad de uso, la escalabilidad, la seguridad, la interoperabilidad, el soporte y la disponibilidad de características y funcionalidades específicas para las necesidades de este proyecto.

Keycloak se destaca por su simplicidad y facilidad de uso, lo que permite a las personas acceder a los diferentes sistemas con un único inicio de sesión, mejorando la experiencia de personas usuaria. Además, también ofrece una arquitectura escalable y flexible que permite el crecimiento y la integración con diferentes plataformas. En términos de seguridad proporciona características avanzadas de autenticación y autorización, incluyendo soporte para autenticación multifactor y diferentes protocolos de autenticación, como OpenID Connect y SAML. Además, esta solución de software libre cuenta con una comunidad activa de desarrollo que ofrece soporte y continúa mejorando la plataforma.

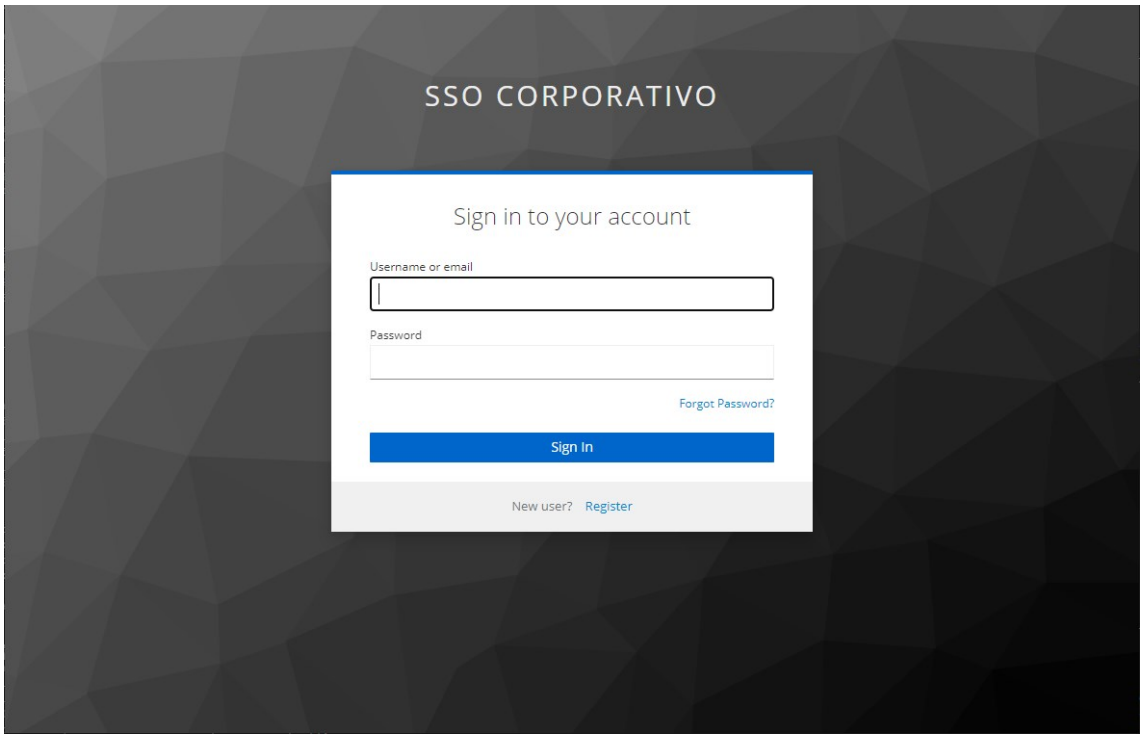

Figura 9: Pantalla de acceso del SSO

3.2. Arquitectura del sistema

Tal como se ha comentado anteriormente, el SSO a desplegar debido a su criticidad necesita una configuración en alta disponibilidad, por lo que cada uno de sus componentes debe configurarse para que permita esta configuración.

A continuación, se exponen los elementos más importantes que tendrá el sistema:

- **Balanceador de carga**: es el componente inicial de la arquitectura y es responsable de distribuir el tráfico de las personas usuarias entre los diversos servidores del sistema. Aparte una de las funciones principales es la detección de actividad de cada uno de los servidores para en su caso, redirigir el tráfico al resto de servidores, evitando así la indisponibilidad.
- **Proveedores de identidad**: estos son los encargados responsables de verificar las credenciales de las personas. Dependiendo del número de tipos de autenticación posible, deberán existir servicios acordes a los mismos, como podrían ser, autenticación por LDAP, certificado, base de datos propia de credenciales, etc.
- **Servicio de persistencia de sesión**: este servicio es el responsable de mantener la sesión de las personas una vez que las mismas se hayan autenticado en el sistema.
- **Registros de control y auditoria**: de cara a mantener un correcto control del sistema se hace imprescindible que todas las operaciones queden auditadas y almacenadas de cara a posibles verificaciones.
- **Base de datos**: habitualmente y como apoyo a otros servicios son requeridas bases de datos para poder consultar y almacenar información en el sistema, como pude ser los datos de auditoria o los tokens de sesiones.

La interacción de estos componentes se muestra en la siguiente figura:

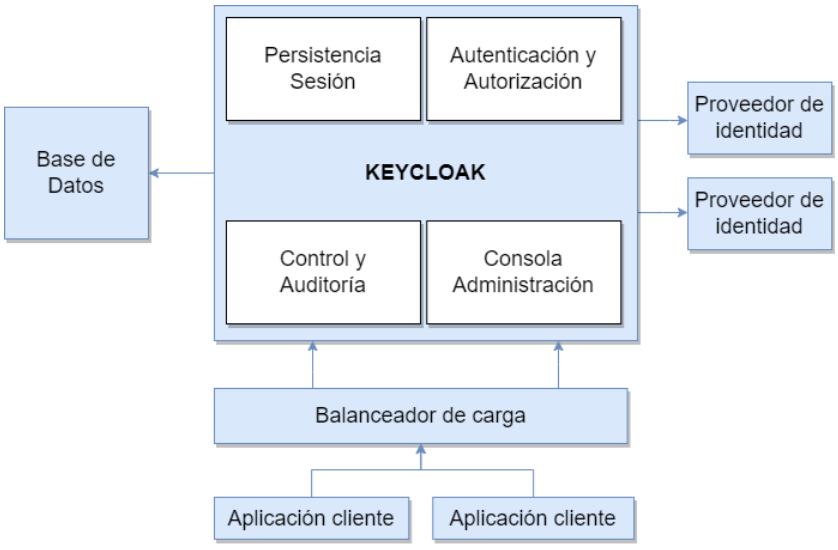

Figura 10: Diagrama de componentes del sistema

Tras tener una primera visión de la arquitecta lógica del sistema, se muestra un diagrama de alto nivel de la arquitectura física de los sistemas de la organización, donde se puede observar donde quedaría ubicado el SSO.

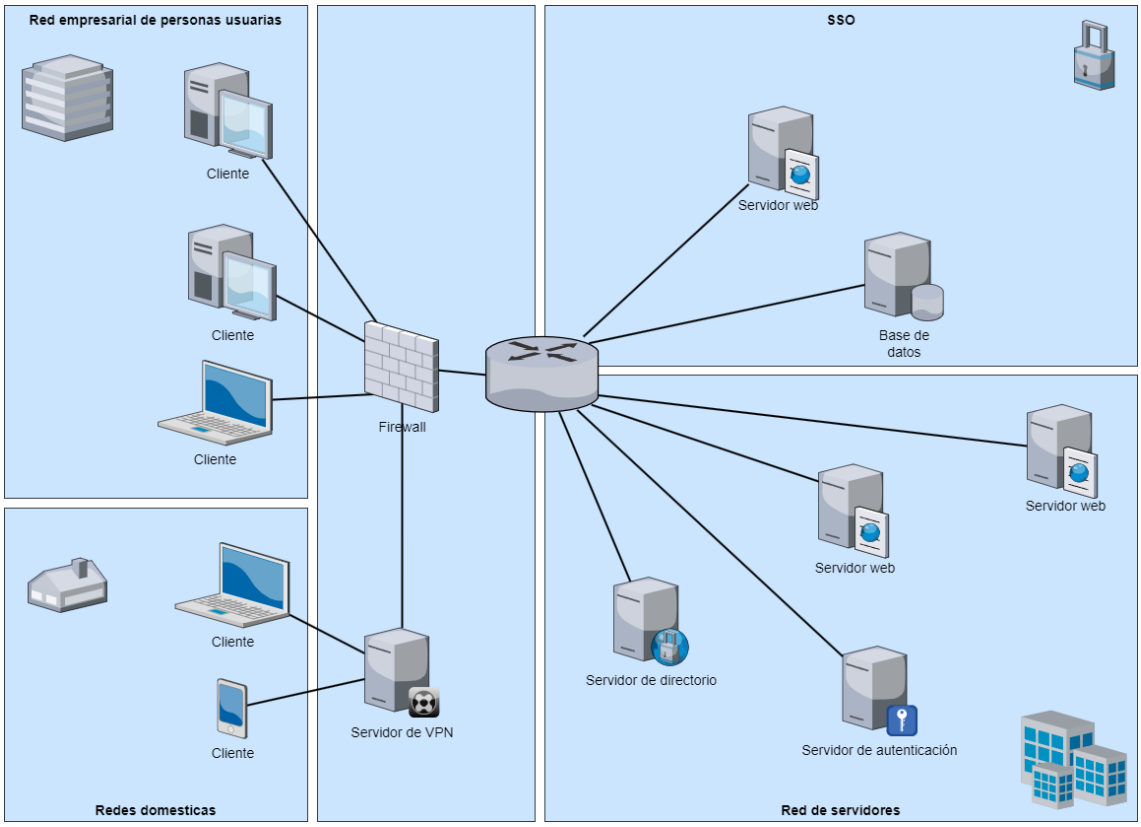

Figura 11: Arquitectura física de alto nivel

3.3. Autenticación y autorización de personas usuarias

Los tres protocolos más importantes utilizados en aplicaciones web y sistemas de identidad para la autenticación y autorización de personas usuarias son actualmente SAML, OAuth y OIDC. Aunque comparten muchos conceptos y funcionalidades, hay algunas grandes diferencias entre ellos:

- **SAML** (Security Assertion Markup Language): se trata de un protocolo basado en XML que permite el intercambio seguro de información de autenticación y autorización entre proveedores de identidad (IdP) y los proveedores de servicios (SP). SAML se centra en la federación de identidad, donde el IdP autentica a la persona y emite un token de seguridad, que luego se envía al SP para permitir el acceso. SAML se utiliza principalmente en entornos empresariales y federaciones de identidad [11].

- **OAuth** (Open Authorization): es un protocolo de autorización utilizado para permitir que una aplicación obtenga acceso limitado a recursos protegidos en nombre de una persona sin compartir sus credenciales de inicio de sesión. OAuth se basa en tokens de acceso y permite que una aplicación obtenga un token de acceso después de que la persona otorgue su consentimiento. Este token se utiliza para acceder a los recursos protegidos en el servidor de recursos. OAuth es ampliamente utilizado en aplicaciones de terceros que requieren acceso a recursos de servicios web, como redes sociales y proveedores de correo electrónico [12] [13].

- **OIDC** (OpenID Connect): se trata de un protocolo de autenticación basado en OAuth 2.0 que proporciona una capa de autenticación sobre OAuth para permitir la autenticación de las personas usuarias utilizando identidades federadas. OIDC combina la autenticación basada en tokens de OAuth con un perfil de identidad basado en JSON para proporcionar información de identidad estructurada de la persona. OIDC proporciona una forma segura y estándar de autenticar personas usuarias en aplicaciones cliente finales [14].

Por ello y en resumen podemos decir que SAML se centra en la federación de identidad mientras que OAuth se utiliza para la autorización de aplicaciones y OIDC agrega una capa de autenticación sobre OAuth, permitiendo la autenticación federada y la obtención de información de identidad estructurada. Cada protocolo tiene sus casos de uso específicos y es importante seleccionar el protocolo adecuado según los requisitos y características de la aplicación o sistema en cuestión.

3.3.1. Secuencia de operaciones de autenticación y autorización

A alto nivel y de forma genérica, los protocolos de autenticación de los SSO funcionan bajo la siguiente secuencia de operaciones:

1. La persona usuaria intenta acceder a una aplicación que necesita autenticación.
2. La aplicación comprueba en el SSO si la persona tenía una sesión previamente iniciada.
3. En caso de que no tuviese, la aplicación redirige a la persona usuaria al servidor de SSO para que se autentique.
4. El SSO solicita a la persona usuaria sus credenciales (como podría ser su identificador de acceso y contraseña o su certificado digital).
5. El SSO valida las credenciales de la persona contra un proveedor de identidad y genera un token de seguridad.
6. La aplicación inicial recibe el token previamente generado y verifica su validez contra el SSO.
7. El SSO valida el token de autenticación y devuelve la información personal a la aplicación solicitante.
8. La aplicación utiliza la información recibida para autorizar el acceso a sus recursos y servicios.

Esto se puede ver representado el siguiente diagrama:

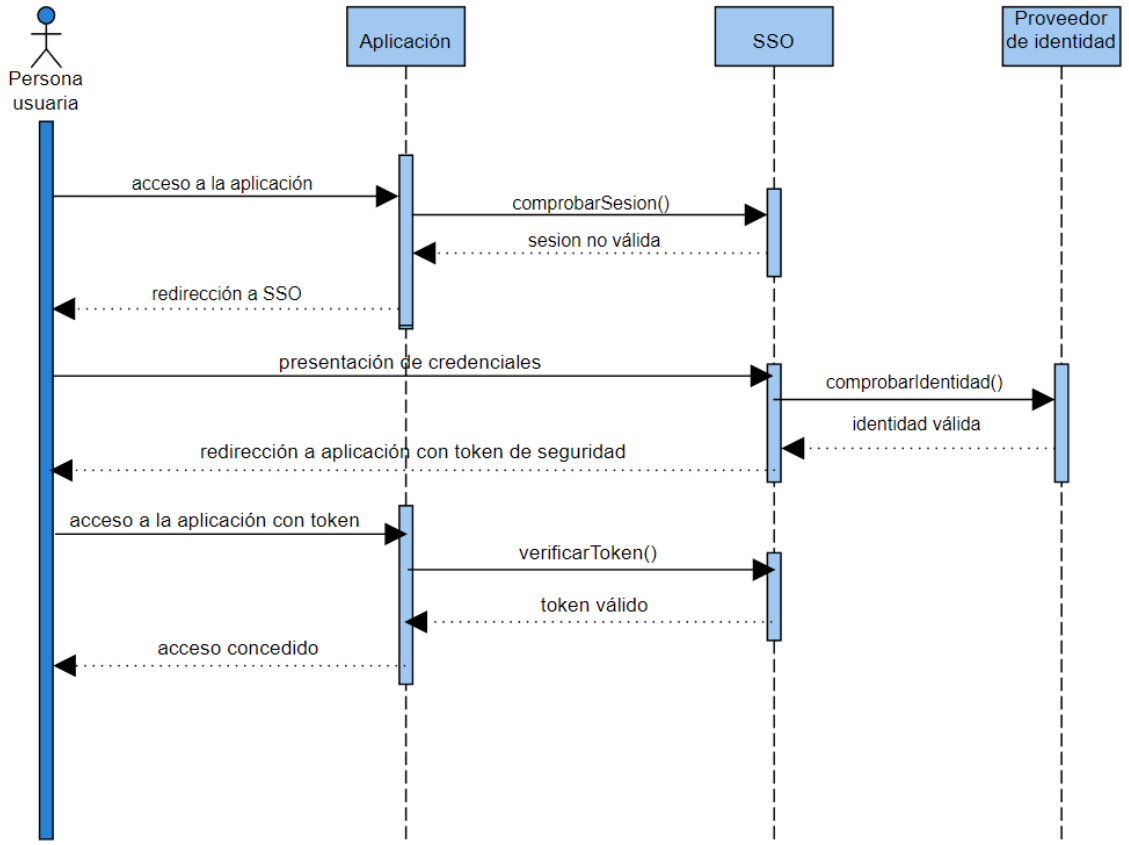

Figura 12: Diagrama de secuencia de autenticación

En este diagrama se puede ver claramente los servicios principales que intervienen en la autenticación. De entre ellos el “Proveedor de Identidad” tiene una especial mención dado que será el encargado de validar las credenciales de la persona usuaria dentro del ámbito del SSO.

En el siguiente apartado veremos que en Keycloak, el Proveedor de identidad puede ser interno o externo, delegando así esta función en terceros sistemas.

3.4. Instalación del sistema

3.4.1. Despliegue en contenedor vs Standalone

Para la instalación de Keycloak, al igual que la mayoría de los sistemas SSO, se puede realizar principalmente de dos formas cuando se despliega “on-premise”: utilizando contenedores Docker o en modo standalone (existe una tercera opción que sería un “hibrido” entre ambas, y es usando imágenes de máquinas virtuales o contenedores preconfigurados, como puede ser el caso de “Bitnami”).

Cada enfoque tiene sus propias ventajas e inconvenientes, las cuales se deben estudiar adecuadamente para decidir así qué opción es la más adecuada para el proyecto y para cada uno de los entornos (por ejemplo, para el entorno de desarrollo puede ser aconsejable una forma mientras que para el de producción otra forma diferente).

A continuación, se presentan las principales ventajas e inconvenientes de cada uno de los modos de despliegue:

Instalación usando **Contenedores Docker**

- Principales ventajas
    - **Portabilidad**: Docker permite empaquetar la aplicación junto con y sus dependencias en un contenedor aislado, lo que facilita la portabilidad y la ejecución en diferentes entornos.
    - **Facilidad de despliegue**: la instalación y el despliegue de la aplicación suele ser bastante rápida y sencilla.
    - **Aislamiento**: cada contenedor se ejecuta de forma aislada, lo que garantiza que los cambios o actualizaciones no afecten a otros servicios o aplicaciones en el sistema. También permite, por ejemplo, tener desplegadas varias instancias sin que compartan información entre ellas, pudiendo tener incluso distintas versiones del producto.
    - **Escalabilidad**: Docker permite escalar los contenedores de forma horizontal y vertical, lo que facilita la gestión del crecimiento de la carga de trabajo según las necesidades de cada servicio.
- Principales inconvenientes
    - **Configuración adicional**: al utilizar Docker, se requiere una configuración adicional para enlazar los contenedores con otros servicios o bases de datos que pudieran ser necesarios para el funcionamiento.
    - **Uso de recursos**: Docker introduce una capa adicional de virtualización, lo que puede hacer consumir más recursos de hardware y afectar al rendimiento en comparación con una instalación standalone.

Instalación en modo **Standalone**

- Principales ventajas
    - **Mayor control y/o configuraciones más detalladas**: al instalar la aplicación en modo standalone, se puede llegar a tener un mayor control sobre la configuración y personalización del entorno.
    - **Acceso directo al sistema operativo**: en el modo standalone, se puede acceder directamente al sistema operativo y software base, pudiendo realizar ajustes específicos según sea necesario.
- Principales inconvenientes
    - **Mayor complejidad de instalación**: la instalación en modo standalone puede requerir más pasos y configuraciones adicionales en comparación con la instalación en Docker al tener que instalar individualmente cada componente junto con sus dependencias.
    - **Mayor acople del sistema**: en modo standalone, es necesario asegurarse de que todas las dependencias y requisitos del sistema estén correctamente instalados y configurados.

Por ello, inicialmente el sistema se desplegará usando la imagen Docker del repositorio "quay.io".

Nota: se puede consultar el método de despliegue básico usando Docker en la web oficial de Keycloak: https://www.keycloak.org/getting-started/getting-started-docker

3.4.2. Instalación con Docker y Docker Compose

Un Dockerfile es un archivo de texto que contiene instrucciones para construir una imagen de Docker. Es utilizado en conjunto con Docker, una plataforma de virtualización ligera que permite empaquetar aplicaciones y sus dependencias en contenedores.

El Dockerfile define una receta reproducible y automatizada para la creación de una imagen de Docker. Cada instrucción del fichero agrega una capa a la imagen resultante, lo que permite que el proceso de construcción sea rápido y eficiente. Al construir una imagen a partir de este fichero, se crea una imagen inmutable y portable que puede ser ejecutada en cualquier entorno de Docker compatible.

A continuación, se muestra el fichero usado para la construcción de la imagen de Keycloak:

```
FROM quay.io/keycloak/keycloak:latest as builder

ENV KC_HEALTH_ENABLED=true
ENV KC_METRICS_ENABLED=true
ENV KC_FEATURES=token-exchange
ENV KC_DB=postgres

RUN curl -sL https:
//github.com/aerogear/keycloak-metrics-spi/releases/download/2.5.3/keycloak-metrics-spi-2.5.3.jar -o
/opt/keycloak/providers/keycloak-metrics-spi-2.5.3.jar
RUN /opt/keycloak/bin/kc.sh build

FROM quay.io/keycloak/keycloak:latest
COPY --from=builder /opt/keycloak/ /opt/keycloak/
WORKDIR /opt/keycloak

ENV KC_DB_URL="jdbc:postgresql://postgres-keycloak:5432/keycloak"
ENV KC_DB_USERNAME=keycloak
ENV KC_DB_PASSWORD=password
ENV KC_HOSTNAME=sso.uoc.edu
ENTRYPOINT ["/opt/keycloak/bin/kc.sh"]
```

Figura 13: Código de Dockerfile

Por otro lado, Docker Compose es una herramienta que simplifica la administración de aplicaciones multi-contenedores en Docker, la cual permite definir y coordinar la configuración de varios contenedores en un único archivo YAML llamado "*docker-compose.yml*". Este archivo describe entre otros, los servicios, redes, volúmenes y otras configuraciones necesarias para ejecutar una aplicación compuesta por múltiples contenedores.

La principal diferencia entre Docker Compose y un Dockerfile radica en su propósito y su nivel de abstracción. Mientras que el Dockerfile se utiliza para definir la construcción de una imagen de un contenedor individual, el archivo *docker-compose.yml* se enfoca en describir, coordinar y orquestar la ejecución de múltiples contenedores interconectados, definiendo la relación entre ellos, las variables de entorno, los puertos expuestos y otras configuraciones necesarias para el despliegue y funcionamiento de la aplicación en su conjunto.

En el caso de Keycloak, para su ejecución es necesaria una base de datos, la cual se ha añadido como servicio externo (en nuestro caso usando PostgreSQL). A continuación, se muestra el fichero *docker-compose.yml* usado:

```yaml
version: '3.8'
services:
  postgres:
      container_name: postgres-keycloak
      image: postgres:14.4
      volumes:
        - postgres_data:/var/lib/postgresql/data
      environment:
        POSTGRES_DB: keycloak
        POSTGRES_USER: keycloak
        POSTGRES_PASSWORD: password
  keycloak:
      container_name: keycloak
      build: .
      environment:
        KC_HEALTH_ENABLED: true
        KC_METRICS_ENABLED: true
        KC_FEATURES: token-exchange
        KC_DB: postgres
        KC_DB_USERNAME: keycloak
        KC_DB_PASSWORD: password
        KEYCLOAK_ADMIN: admin
        KEYCLOAK_ADMIN_PASSWORD: uoc.2023
        PROXY_ADDRESS_FORWARDING: true
        REDIRECT_SOCKET: "https"
        KC_DB_URL: "jdbc:postgresql://postgres-keycloak:5432/keycloak"
      command: start --hostname sso.uoc.edu --optimized --proxy edge
      --spi-login-protocol-openid-connect-legacy-logout-redirect-uri=true
      ports:
        - 8080:8080
      depends_on:
        - postgres
volumes:
  postgres_data:
      driver: local
```

Figura 14: Código de Docker Compose utilizado

3.4.3. Alta disponibilidad

Para lograr alta disponibilidad en Keycloak, se debe asegurar que no existen SPO (puntos únicos de fallo). Para ello se deben implementar varias medidas:

- **Configuración de clúster**: utilizar una arquitectura de clúster con múltiples instancias de Keycloak distribuidas en diferentes servidores. Esto asegura que si una instancia o servidor de Keycloak falla, las otras instancias puedan continuar dando el servicio de forma adecuada. Para esto se puede utilizar dos estrategias principalmente: el balanceo activo/activo y activo/pasivo

    - En la configuración de servidores activo/activo, se tienen varios servidores en funcionamiento simultáneamente, compartiendo la carga de trabajo y aprovechando así los recursos puestos a disposición. Cada servidor activo maneja un porcentaje de las solicitudes de autenticación, lo que permite distribuir la carga y mejorar el rendimiento. Esta configuración requiere la sincronización de datos entre los servidores, ya sea mediante la replicación de la base de datos o mediante el uso de sistemas de caché distribuida. Además, se deben tener en cuenta mecanismos de coordinación para evitar conflictos en la gestión de sesiones y otros aspectos críticos del sistema.

    - En la configuración de servidores activo/pasivo, se establece un servidor principal (activo) y uno o varios servidores secundarios (pasivos) en

espera. El servidor activo es el encargado de gestionar las solicitudes de autenticación y autorización de las personas usuarias. En caso de que el servidor activo falle, uno de los servidores secundarios toma su lugar automáticamente para garantizar la continuidad del servicio. Para implementar esta configuración, se requiere la utilización de un mecanismo de detección de fallos, como un balanceador de carga o un sistema de conmutación por error (heartbeat). Este tipo de HA solo se debe utilizar cuando no es posible utilizar el método activo/activo dado que a priori se desperdicia recursos físicos y lógicos (energía / cpu / memoria / disco) a la espera de que se produzca un fallo en el sistema.

- **Réplica de base de datos**: es recomendable utilizar una base de datos externa y escalable y configurada en modo replicación. Esto permite tener varias copias de la base de datos distribuidas en diferentes servidores, evitando que un fallo en el servidor de base de datos afecte la disponibilidad del sistema. La replicación asegura que los datos estén sincronizados y disponibles en caso de que una réplica falle.

- **Almacenamiento de archivos externos**: algunas configuraciones requieren almacenar archivos externos, como imágenes o ficheros de configuración. Para evitar un SPO en el almacenamiento de estos archivos, se puede utilizar un sistema de almacenamiento distribuido como puede ser un NFS. Esto permite que los archivos estén disponibles en diferentes ubicaciones y garantiza la disponibilidad incluso si un servidor o sistema de almacenamiento falla.

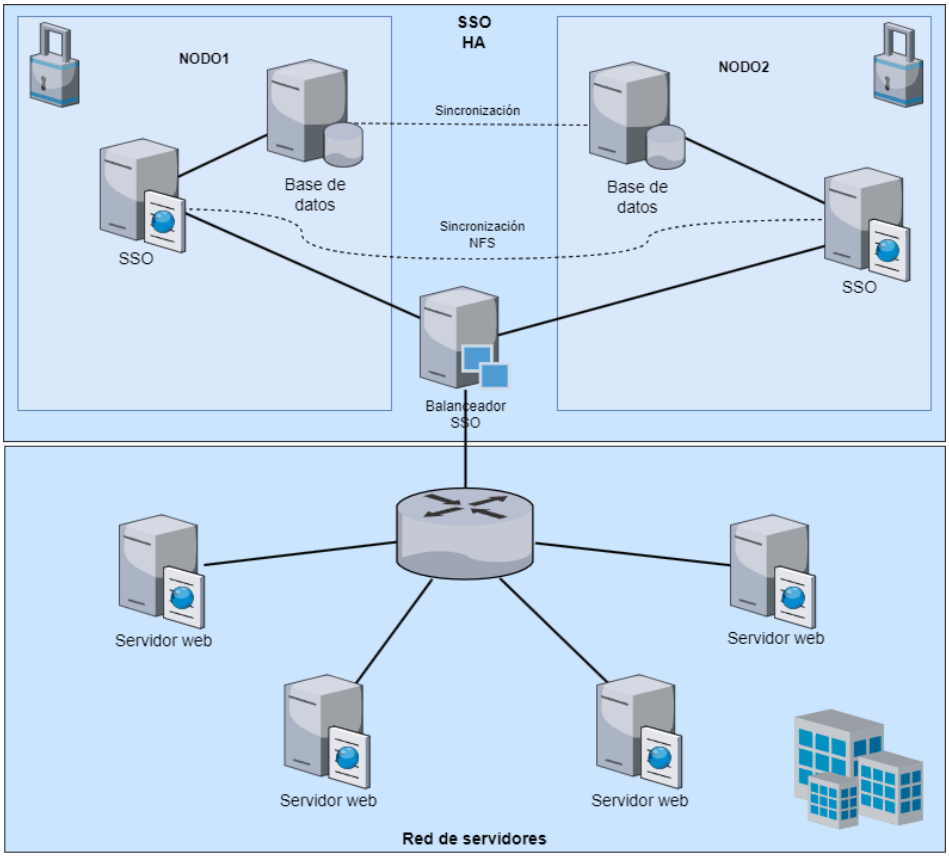

Figura 15: Diagrama de alta disponibilidad

3.4.3.1. Alta disponibilidad basada en la nube

Además de las soluciones tradicionales On-Premise, existen otras alternativas para lograr la alta disponibilidad en Keycloak como pueden ser los operadores de Kubernetes y los charts de Helm.

Estas soluciones son ofrecidas tanto de forma oficial por los desarrolladores del software como por la propia comunidad o de terceras empresas.

- Los **operadores de Kubernetes** son controladores personalizados que extienden las capacidades de Kubernetes para administrar aplicaciones complejas. Los operadores de Keycloak permiten automatizar y simplificar la implementación, configuración y gestión de Keycloak en un clúster de Kubernetes. Proporcionan funcionalidades específicas de alta disponibilidad, como la configuración de múltiples réplicas de Keycloak y la distribución del tráfico entre ellas. Para la instalación de estos operadores se puede consultar la documentación oficial en https://www.keycloak.org/operator/installation.

- Los **charts de Helm** son paquetes preconfigurados que contienen archivos de configuración para desplegar aplicaciones en Kubernetes. Los charts de Helm de Keycloak facilitan la implementación y configuración de Keycloak con alta disponibilidad, ya que incluyen valores y opciones recomendadas para garantizar la redundancia y la escalabilidad. Con Helm, es posible desplegar rápidamente instancias de Keycloak y gestionar su configuración de forma sencilla. Estos paquetes preconfigurados se pueden descargar terceras fuentes:
  - *Codecentric*: https://artifacthub.io/packages/helm/codecentric/keycloak
  - *Bitnami*: https://bitnami.com/stack/keycloak/helm.

Estas alternativas basadas en operadores de Kubernetes y charts de Helm ofrecen una forma automatizada y simplificada de lograr alta disponibilidad en Keycloak, aprovechando las capacidades y la flexibilidad de Kubernetes como plataforma de orquestación de contenedores.

3.5. Configuración del sistema

De cara a la configuración del sistema, una de las primeras decisiones que se debe de tomar al implantar un SSO, es el proveedor de identidad que se utilizará en el sistema. Según su naturaleza, podemos hablar de tres grupos diferenciados:

- **Interno:** es el tipo por defecto con el que las personas usuarias se autentican en el sistema, y está integrado en la propia aplicación.

- **Externo federado**: delegando la autenticación de las personas en sistemas de Active Directory o LDAP.

- **Externo usando otro sistema (IdP)**: delegando la autenticación de las personas en sistemas externos a la compañía ya sean locales o en la nube (Google, Microsoft, Github, Facebook, Twitter, etc.).

En la configuración que se realizará del sistema, inicialmente se utilizará la configuración de identificación interna, quedando para futuros trabajos la federación y el uso sistemas externos.

3.5.1. Configuración de dominios

Los dominios de Keycloak, también conocidos como *Realms*, son una característica fundamental de la plataforma. Un dominio en Keycloak actúa como un contenedor lógico y aislado que agrupa credenciales, aplicaciones y configuraciones relacionadas, la cual proporciona una forma organizada y segura de administrar la autenticación y autorización de las personas usuarias en un entorno de múltiples aplicaciones.

Cada dominio en Keycloak tiene su espacio de datos independiente, lo que permite separar y gestionar de manera eficiente las identidades y las políticas de acceso para diferentes proyectos o áreas funcionales dentro de la organización. Los dominios también ofrecen la capacidad de definir reglas y políticas de seguridad específicas, lo que permite adaptar la experiencia de autenticación y autorización según los requisitos únicos de cada dominio.

Además, los dominios de Keycloak brindan un alto nivel de escalabilidad y flexibilidad, pudiendo ser utilizados por ejemplo para crear entornos de prueba y producción separados, lo que facilita el desarrollo y la implementación de nuevas funcionalidades sin afectar el funcionamiento de los sistemas en producción.

3.5.2. Aplicaciones clientes

Las aplicaciones clientes son aquellas aplicaciones o servicios que se integran con el SSO proporcionado por Keycloak. Estas aplicaciones como consumidores de la autenticación y autorización proporcionadas por Keycloak, permitiendo a las personas usuarias acceder a sus funcionalidades utilizando sus credenciales de inicio de sesión.

Keycloak proporciona diferentes mecanismos de integración para las aplicaciones clientes:

- **SAML**: Keycloak admite el protocolo para la integración con aplicaciones cliente que utilizan este estándar. Las aplicaciones cliente pueden configurarse como proveedores de servicios SAML y comunicarse con Keycloak como proveedor de identidad para la autenticación y autorización de las personas.

- **OIDC**: la plataforma también utiliza este protocolo para la autenticación y autorización de las personas. Las aplicaciones clientes pueden implementar el flujo de autenticación basado en OIDC para interactuar con Keycloak y obtener los tokens de acceso necesarios para acceder a los recursos protegidos.

A continuación, veremos la configuración de estos.

3.5.2.1. SAML

Para la configuración de un SAML se tiene establecer una configuración en la aplicación cliente (aplicación que se integrará con el SSO) como en el servidor de Keycloak (o en cualquier otro sistema de Single Sign-On) los puntos más importantes son los siguientes:

Configuración de la aplicación cliente de SAML

- **Nombre cliente**: se debe proporcionar el nombre del proveedor de identidad SAML con el que se integrará Keycloak, el cual será único (dentro del dominio) e identificará a la aplicación en Keycloak.

- **URL de metadatos SAML**: Keycloak también debe proporcionar a las aplicaciones clientes la URL de los metadatos de SAML. Estos metadatos contienen información sobre el proveedor de identidad, como la clave pública, la URL de inicio de sesión, etc. La aplicación cliente utilizará estos metadatos para establecer una conexión segura con el SSO.

- **Certificado X.509**: aunque no es obligatorio, es muy recomendable utilizar un un certificado X.509 para establecer la confianza entre la aplicación cliente y Keycloak. Este certificado debe generarlo la aplicación cliente y posteriormente añadir la clave pública en la configuración del cliente de Keycloak.

Configuración en Keycloak:

Dentro de Keycloak, y una vez elegido el dominio donde crearemos los clientes, en el menú homónimo, se dan de alta los mismos.

Para dar de alta un cliente, se debe utilizar información común que se ha utilizado en la configuración del cliente como puede ser el Cliente ID (que debe coincidir con el establecido anteriormente), las URL's de vuelta de la aplicación (una vez autenticado) así como la configuración de los claves de autenticación y cifrado si las hubiera dentro de la pestaña *keys*.

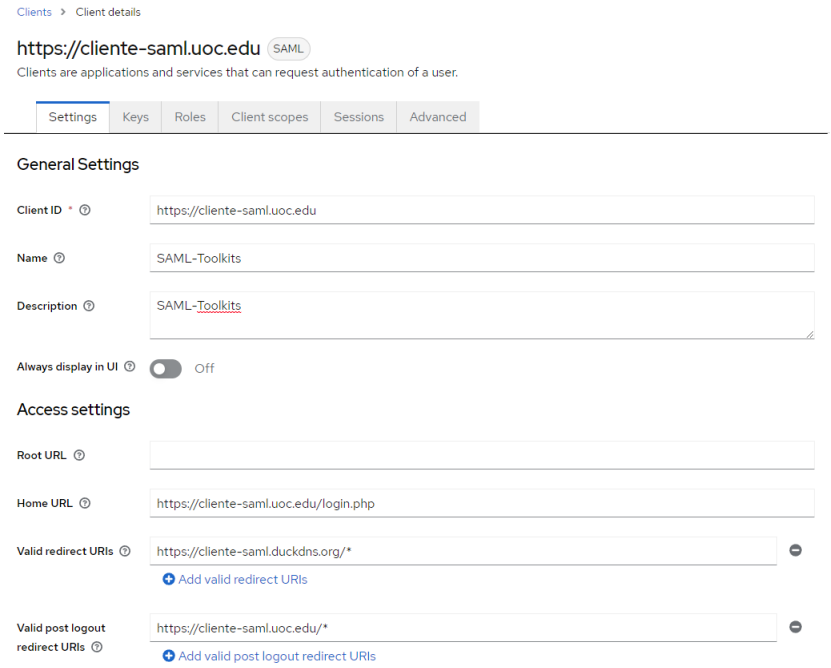

Figura 16: Configuración de aplicación cliente con SAML

Para obtener la URL de los metadatos SAML del SSO, se realiza desde el menú de configuración del dominio:

https://sso.uoc.edu/realms/SSO-Corporativo/protocol/saml/descriptor

```
<md:EntityDescriptor xmlns="urn:oasis:names:tc:SAML:2.0:metadata" xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata" xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion" xmlns:ds=
"http://www.w3.org/2000/09/xmldsig#" entityID="https://sso.uoc.edu/realms/SSO-Corporativo">
    <md:IDPSSODescriptor WantAuthnRequestsSigned="true" protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
        <md:KeyDescriptor use="signing">
            <ds:KeyInfo>
                <ds:KeyName>o7sI5uKu9w7lqbUxntNtDF1qXCy9rb0_RrSng1lNVrs</ds:KeyName>
                <ds:X509Data>
                    <ds:X509Certificate>
                    MIIClzCCAX8CBgGHvz/fajANBgkqhkiG9w0BAQsFADAPMQ0wCwYDVQQDDARUZXN0MB4XDTIzMDQyNjIwMjU0NloXDTMzMDQyNjIwMjcyNlowDzENMAsGA1UEAwwEVGVzdDCCASIwDQYJKoZIhvcNAQEBBQA
                    DggEPADCCAQoCggEBAOXiBpCUKsp5caCNJAJI9Xv/tRMLPRj2bFL4qXeEKZPCJwgE2LENcsbfyP1dgW8o9dlcHdfXlXH19Uq6UCEc2VHR+q9jSvEpZQL3ZicF1Rslk3q2/FNj3oEw2IdAG3VrZNfZBoECMU
                    X2STJ/7PFpVIv77beLs1dyLc+hz6TwVyhOF/TQEHPLsCcl4AU7ETdDbDHLbz3aWy+NpjmE5Z+ZguK1qQzwGgMguL4rTPQuRmYaYrfmtE8jllyqESWaZOF2j+4cxaDdE502Hu/OeuSBilWGceoqQ+oNP8fRt
                    JUuQhSGAK6fnKtZo6tkY0/F6UYqvQiZUd7mkrewrJ4sA7f3JgkCAwEAATANBgkqhkiG9w0BAQsFAAOCAQEAliC8Mc00S9f+CtTGOHnDfwLJqzpxO2FmG+cwFA5E+ody7j69ZNHnno0ZO8jJ3kOw0yrYYoDQ
                    m0TUuuh3E6TUgoLjnbyA3GDoaXpv5gu9WsHTi1OhW4N7xxH4JZHc3E+G0YboORRJpPaSbCB80Ec+2OAe+4M78s79XbgpTRZB05M9enf87utEJDtOsNuHt7cXAAPeSLNKuYOHkMn+brjElxRbzu45hclUIq
                    PqWDNAa1nlm56N3LDpOwfJzQ+0LKWgtb0Ex4owcL60jfO+uqi26tVGjzYYkZ+fGiEleRDY+uQP2022O2RkWtCj1HcR/sy2VHLUB0pocrmclE/xY0bRA==</ds:X509Certificate>
                </ds:X509Data>
            </ds:KeyInfo>
        </md:KeyDescriptor>
        <md:ArtifactResolutionService Binding="urn:oasis:names:tc:SAML:2.0:bindings:SOAP" Location="https://sso.uoc.edu/realms/SSO-Corporativo/protocol/saml/resolve" index="0"
        />
        <md:SingleLogoutService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="https://sso.uoc.edu/realms/SSO-Corporativo/protocol/saml"/>
        <md:SingleLogoutService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect" Location="https://sso.uoc.edu/realms/SSO-Corporativo/protocol/saml"/>
        <md:SingleLogoutService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Artifact" Location="https://sso.uoc.edu/realms/SSO-Corporativo/protocol/saml"/>
        <md:NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</md:NameIDFormat>
        <md:NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:transient</md:NameIDFormat>
        <md:NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified</md:NameIDFormat>
        <md:NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</md:NameIDFormat>
        <md:SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="https://sso.uoc.edu/realms/SSO-Corporativo/protocol/saml"/>
        <md:SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect" Location="https://sso.uoc.edu/realms/SSO-Corporativo/protocol/saml"/>
        <md:SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:SOAP" Location="https://sso.uoc.edu/realms/SSO-Corporativo/protocol/saml"/>
        <md:SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Artifact" Location="https://sso.uoc.edu/realms/SSO-Corporativo/protocol/saml"/>
    </md:IDPSSODescriptor>
</md:EntityDescriptor>
```


Figura 17: SAML 2.0 Identity Provider Metadata

3.5.2.2. OIDC

De igual forma que en los clientes SAML, para la configuración de los clientes OIDC las propiedades claves son las siguientes:

Configuración de la aplicación cliente de OIDC

- El **Client ID** es un identificador único asignado a tu cliente OIDC por el SSO. Este valor se utiliza para identificar de manera únivoca el cliente en las interacciones con el proveedor de identidad.

- El **Client Secret** es una clave secreta compartida entre el cliente OIDC y el proveedor de identidad. Esta clave se utiliza para autenticar securizar las comunicaciones entre ambos extremos. El Client Secret es una medida de seguridad adicional para garantizar que solo el cliente autorizado pueda comunicarse con el proveedor de identidad, por lo que es muy importante mantener dicha clave de forma confidencial, ya que su compromiso podría permitir a un tercero no autorizado acceder al cliente y a sus recursos protegidos.

- La **URL del proveedor de identidad** es la dirección específica del dominio del SSO donde se implementa el protocolo OpenID Connect. Esta URL se utiliza para realizar la configuración del cliente y denotas las URLS donde se realizarán las solicitudes de autenticación, autorización y obtención de tokens.

- De forma opcional, en OIDC se pueden usar los **JSON Web Key Set** los cuales son un componente clave para la validación y verificación de tokens JWT. El JWKS es una representación JSON que contiene un conjunto de claves públicas utilizadas para verificar la autenticidad y firma de los tokens emitidos por el proveedor de identidad.

Configuración en Keycloak:

De forma análoga a SAML, una vez elegido el dominio donde crearemos los clientes, en el menú “Clients”, creando uno de tipo OpenID Connect:

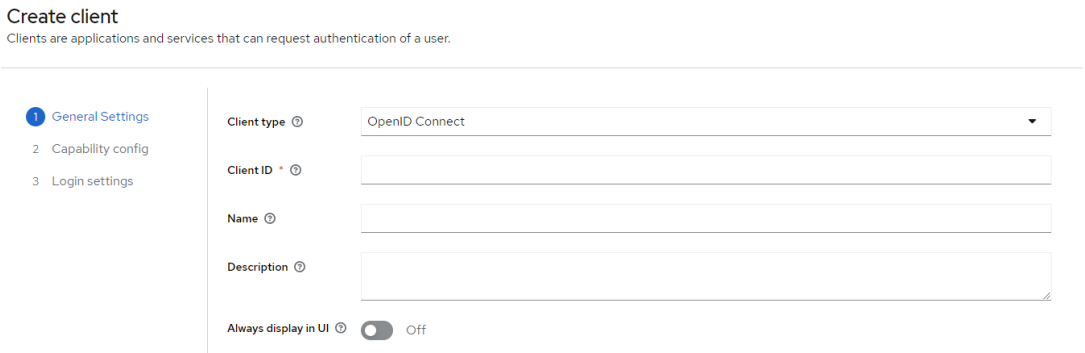

Figura 18: Pantalla de creación de cliente OIDC

Una vez rellenados los datos concordantes con los que se han introducido en el cliente como son el Client ID, se terminará de configurar el cliente añadiendo las claves, y URL’s de vuelta al cliente:

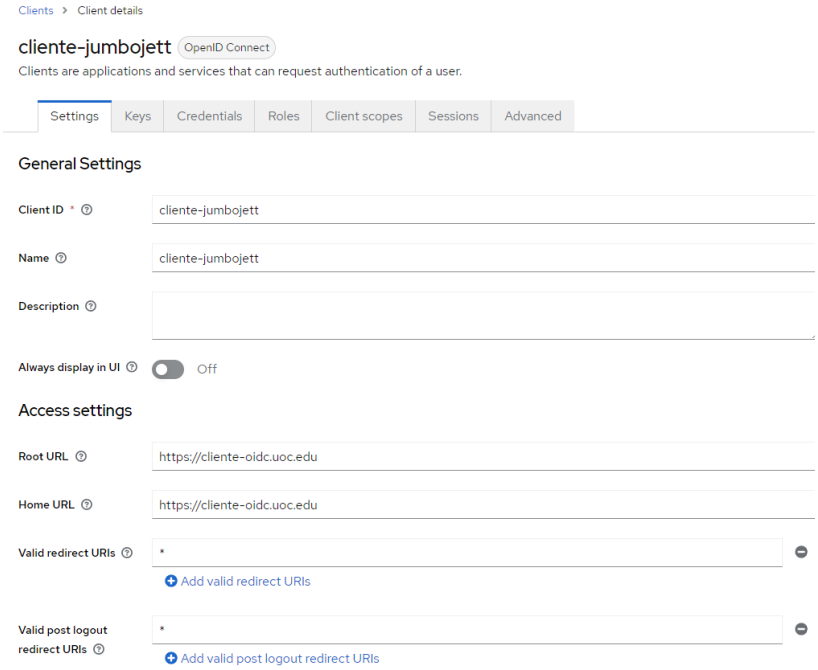

Figura 19: Configuración de aplicación cliente con OIDC

De igual forma que se hacía en SAML, para obtener la URL de los metadatos de OIDC del SSO, se realiza desde el menú de configuración del dominio:

https://sso.uoc.edu/realms/SSO-Corporativo/.well-known/openid-configuration


```
{
    "issuer": "https://sso.uoc.edu/realms/SSO-Corporativo",
    "authorization_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/auth",
    "token_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/token",
    "introspection_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/token/introspect",
    "userinfo_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/userinfo",
    "end_session_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/logout",
    "frontchannel_logout_session_supported": true,
    "frontchannel_logout_supported": true,
    "jwks_uri": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/certs",
    "check_session_iframe": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/login-status-iframe.html",
    "grant_types_supported": ["authorization_code", "implicit", "refresh_token", "password", "client_credentials", "urn:ietf:params:oauth:grant-type:device_code", "urn:openid:params:grant-type:ciba"],
    "acr_values_supported": ["0", "1"],
    "response_types_supported": ["code", "none", "id_token", "token", "id_token token", "code id_token", "code token", "code id_token token"],
    "subject_types_supported": ["public", "pairwise"],
    "id_token_signing_alg_values_supported": ["PS384", "ES384", "RS384", "HS256", "HS512", "ES256", "RS256", "HS384", "ES512", "PS256", "PS512", "RS512"],
    "id_token_encryption_alg_values_supported": ["RSA-OAEP", "RSA-OAEP-256", "RSA1_5"],
    "id_token_encryption_enc_values_supported": ["A256GCM", "A192GCM", "A128GCM", "A128CBC-HS256", "A192CBC-HS384", "A256CBC-HS512"],
    "userinfo_signing_alg_values_supported": ["PS384", "ES384", "RS384", "HS256", "HS512", "ES256", "RS256", "HS384", "ES512", "PS256", "PS512", "RS512", "none"],
    "userinfo_encryption_alg_values_supported": ["RSA-OAEP", "RSA-OAEP-256", "RSA1_5"],
    "userinfo_encryption_enc_values_supported": ["A256GCM", "A192GCM", "A128GCM", "A128CBC-HS256", "A192CBC-HS384", "A256CBC-HS512"],
    "request_object_signing_alg_values_supported": ["PS384", "ES384", "RS384", "HS256", "HS512", "ES256", "RS256", "HS384", "ES512", "PS256", "PS512", "RS512", "none"],
    "request_object_encryption_alg_values_supported": ["RSA-OAEP", "RSA-OAEP-256", "RSA1_5"],
    "request_object_encryption_enc_values_supported": ["A256GCM", "A192GCM", "A128GCM", "A128CBC-HS256", "A192CBC-HS384", "A256CBC-HS512"],
    "response_modes_supported": ["query", "fragment", "form_post", "query.jwt", "fragment.jwt", "form_post.jwt", "jwt"],
    "registration_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/clients-registrations/openid-connect",
    "token_endpoint_auth_methods_supported": ["private_key_jwt", "client_secret_basic", "client_secret_post", "tls_client_auth", "client_secret_jwt"],
    "token_endpoint_auth_signing_alg_values_supported": ["PS384", "ES384", "RS384", "HS256", "HS512", "ES256", "RS256", "HS384", "ES512", "PS256", "PS512", "RS512"],
    "introspection_endpoint_auth_methods_supported": ["private_key_jwt", "client_secret_basic", "client_secret_post", "tls_client_auth", "client_secret_jwt"],
    "introspection_endpoint_auth_signing_alg_values_supported": ["PS384", "ES384", "RS384", "HS256", "HS512", "ES256", "RS256", "HS384", "ES512", "PS256", "PS512", "RS512"],
    "authorization_signing_alg_values_supported": ["PS384", "ES384", "RS384", "HS256", "HS512", "ES256", "RS256", "HS384", "ES512", "PS256", "PS512", "RS512"],
    "authorization_encryption_alg_values_supported": ["RSA-OAEP", "RSA-OAEP-256", "RSA1_5"],
    "authorization_encryption_enc_values_supported": ["A256GCM", "A192GCM", "A128GCM", "A128CBC-HS256", "A192CBC-HS384", "A256CBC-HS512"],
    "claims_supported": ["aud", "sub", "iss", "auth_time", "name", "given_name", "family_name", "preferred_username", "email", "acr"],
    "claim_types_supported": ["normal"],
    "claims_parameter_supported": true,
    "scopes_supported": ["openid", "web-origins", "acr", "roles", "address", "phone", "offline_access", "profile", "email", "microprofile-jwt"],
    "request_parameter_supported": true,
    "request_uri_parameter_supported": true,
    "require_request_uri_registration": true,
    "code_challenge_methods_supported": ["plain", "S256"],
    "tls_client_certificate_bound_access_tokens": true,
    "revocation_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/revoke",
    "revocation_endpoint_auth_methods_supported": ["private_key_jwt", "client_secret_basic", "client_secret_post", "tls_client_auth", "client_secret_jwt"],
    "revocation_endpoint_auth_signing_alg_values_supported": ["PS384", "ES384", "RS384", "HS256", "HS512", "ES256", "RS256", "HS384", "ES512", "PS256", "PS512", "RS512"],
    "backchannel_logout_supported": true,
    "backchannel_logout_session_supported": true,
    "device_authorization_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/auth/device",
    "backchannel_token_delivery_modes_supported": ["poll", "ping"],
    "backchannel_authentication_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/ext/ciba/auth",
    "backchannel_authentication_request_signing_alg_values_supported": ["PS384", "ES384", "RS384", "ES256", "RS256", "ES512", "PS256", "PS512", "RS512"],
    "require_pushed_authorization_requests": false,
    "pushed_authorization_request_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/ext/par/request",
    "mtls_endpoint_aliases": {
        "token_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/token",
        "revocation_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/revoke",
        "introspection_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/token/introspect",
        "device_authorization_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/auth/device",
        "registration_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/clients-registrations/openid-connect",
        "userinfo_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/userinfo",
        "pushed_authorization_request_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/ext/par/request",
        "backchannel_authentication_endpoint": "https://sso.uoc.edu/realms/SSO-Corporativo/protocol/openid-connect/ext/ciba/auth"
    }
}
```


Figura 20: OpenID Endpoint Configuration

3.5.3. Múltiples factores de autenticación

Los múltiples factores de autenticación (MFA, por sus siglas en inglés) son una forma de fortalecer la seguridad del proceso de autenticación al requerir que las personas usuarias proporcionen más de un elemento para verificar su identidad. Estos elementos de se puede clasificar en tres grupos principalmente:

- **Algo que se sabe**: en este grupo se aglutinan la información que solo la persona usuaria debe conocer, como pueden ser las contraseñas, los códigos PIN o las respuestas a preguntas de seguridad.

- **Algo que se tiene**: este segundo grupo hace referencia a la posesión física de un dispositivo o una tarjeta, como podría ser los tokens de seguridad, tarjetas inteligentes o teléfonos móviles.

- **Algo que se es**: por último, este grupo utiliza características inherentes de las personas, como puede ser sus huellas dactilares, su biometría facial o patrones de voz.

La implementación de MFA ofrece varios beneficios en términos de seguridad:

- **Mayor protección contra ataques de suplantación de identidad**: al requerir más de un factor para la autenticación, los atacantes enfrentan un obstáculo adicional para obtener acceso no autorizado a una cuenta.

- **Reforzamiento de la autenticación débil**: si una contraseña o credencial de autenticación única se ve comprometida, el MFA proporciona una capa adicional de seguridad al requerir un segundo factor.

- **Cumplimiento normativo**: muchos estándares de seguridad y regulaciones gubernamentales exigen el uso de MFA para ciertos sistemas y aplicaciones que almacenan o acceden a datos sensibles.

Keycloak de forma nativa usa credenciales tipo login y contraseña con la posibilidad de añadir como segundo factor un OTP (contraseña de un solo uso).

Para activarlo, simplemente hay que marcar los toggles correspondientes en el menú Autenticación, dentro de la pestaña "*Required actions*":

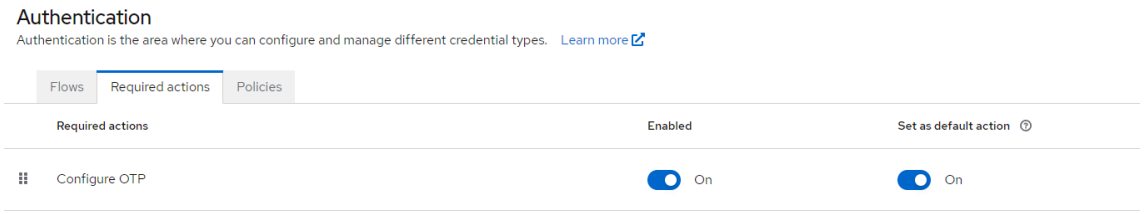

Figura 21: Activación de OTP

Tras activarlo, todas las personas usuarias, al hacer login en la plataforma tendrán que configurar un sistema OTP (en este caso la propia plataforma sugiere 3 aplicaciones), tras instarla y seguir las indicaciones, el doble factor quedará vinculado correctamente a la persona usuaria (ver Figura 22: Configuración de sistema OTP).

En las siguientes autenticaciones que realice la persona en la plataforma, a las personas se les solicitará el código OTP generado por la aplicación software (ver Figura 23: Solicitud de código OTP)

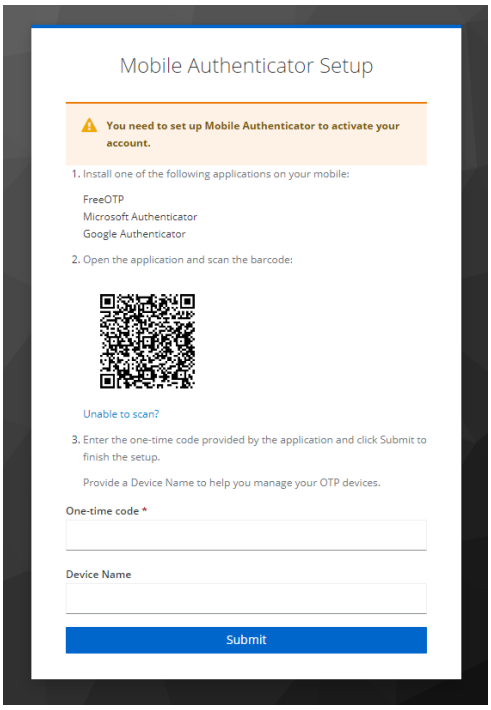

Figura 22: Configuración de sistema OTP

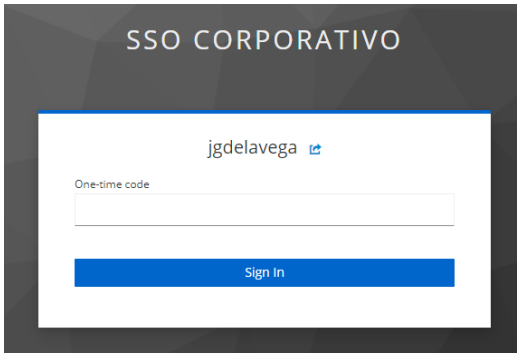

Figura 23: Solicitud de código OTP

4. Conclusiones y trabajos futuros

4.1. Resultado del proyecto

La implantación Keycloak como sistema centralizado de autenticación en la organización ha demostrado proporcionar mejoras significativas en varios aspectos clave:

- Se ha observado una notable mejora en los tiempos que consumirá la plantilla en el acceso a las herramientas corporativas. Gracias al mecanismo de autenticación centralizada y la capacidad de inicio de sesión único, las personas usuarias podrán acceder rápidamente a todas aplicaciones y servicios de la organizativos sin tener que recordar múltiples credenciales, agilizando con ello los procedimientos operativos y aumentando así su productividad.

- En términos de seguridad, Keycloak ha demostrado ser una solución robusta y confiable dado que implementa protocolos de seguridad como SAML, OAuth y OIDC, consiguiendo así establecer mecanismos sólidos para la autenticación y autorización de las personas usuarias. Además, las características de gestión de identidad y acceso de Keycloak, como la autenticación multifactor y la gestión centralizada de políticas de seguridad, permiten incrementar la seguridad de los accesos a los sistemas.

- En cuanto al aspecto económico, se ha comprobado que la implantación de Keycloak tendrá una repercusión muy positiva en los márgenes económicos de la organización al evitar el gasto innecesario de tiempo de las personas trabajadoras, siendo el coste de implantación muy por debajo al beneficio obtenido. Además, la alta compatibilidad del sistema y su capacidad de escalabilidad, unido a los mecanismos de actualizaciones de la solución, consiguen garantizar que no existirán gastos sobrevenidos en el futuro por la necesidad de realizar cambios funcionales o de ampliación de la infraestructura.

4.2. Verificación del sistema

Una vez realizada la implementación y despliegue del sistema se deberá realizar una lista de comprobaciones y de verificación que asegure la correcta integridad de los productos desarrollados e implementados. Esta lista constará al menos de los siguientes elementos:

1. **Verificación de requisitos**: asegurar que todos los requisitos del proyecto están claramente definidos y comprensibles.

2. **Configuración del entorno**: configurar y verificar que el entorno de pruebas / productivo esté funcionando de forma adecuada y óptima, incluyendo servidores, servicios de directorio, aplicaciones, etc.

3. **Pruebas unitarias**: realizar pruebas para comprobar que todos los componentes funcionen adecuadamente, verificando entre otros la autenticación, autorización, gestión de sesiones y el acceso a los diferentes recursos.

4. **Pruebas de integración**: verificar que los sistemas de terceros se integran correctamente con el SSO. Esto conlleva asegurar que las personas usuarias

puedan autenticarse adecuadamente y tener acceso a los recursos y aplicaciones.

5. **Pruebas de seguridad**: realizar pruebas de seguridad para garantizar que el sistema sea seguro y resista ataques, como los de denegación de servicio.

6. **Pruebas de escalabilidad**: comprobar que la aplicación sea escalable y pueda manejar un número de accesos simultáneamente acorde a las necesidades de la organización.

7. **Pruebas de rendimiento**: realizar comprobaciones de rendimiento para verificar que el sistema sea capaz de manejar cargas pesadas sin interrupciones.

8. **Pruebas de recuperación ante desastres**: verificar que la solución sea capaz de recuperarse rápidamente de fallos o desastres, y que los datos estén seguros y disponibles aplicando en caso de ser necesarios los protocolos de recuperación.

9. **Revisión de la documentación**: asegurar que toda la documentación, manuales de uso, guías de configuración y otros recursos estén actualizados y sean comprensibles.

5. Glosario

Término	Definición
CAS Protocol	Protocolo de comunicaciones mantenido por la Fundación Apereo.
CPD	Centro de proceso de datos.
IdP	Identity Provider; Proveedor de identidad.
JWKS	JSON Web Key Set.
JWT	JSON Web Token.
Kerberos	Protocolo de autenticación creado por el MIT.
MIT	Instituto Tecnológico de Massachusetts.
OAuth	Protocolo de autenticación “Open Authorization”.
OpenID Connect	OpenID Connect es un protocolo de identidad basado en el protocolo OAuth 2.0.
PaaS	*Platform as a Service;* Plataforma como servicio.
PEC	Prueba de evaluación continua.
Radius	Protocolo de autenticación de acceso a red IP.
Realm	Dominio de aplicación.
RESTful APIs	Interfaz de sistemas de computación basada en HTTP.
ROI	*Return On Investment*; Retorno de la inversión.
SaaS	*Software as a Service;* Software como servicio.
SP	*Service Provider; Proveedor de servicio.*
SAML	*Security Assertion Markup Language;* Lenguaje de Marcado para Confirmaciones de Seguridad.
SPOF	*Single Point Of Fail;* Punto único de fallo.
SSO	*Single Sign-On;* Sistema de autenticación único.
TFM	Trabajo de fin de máster.
TIC	Tecnologías de la información y las comunicaciones.
WCAG	*Content Accessibility Guidelines;* Pautas para la accesibilidad del contenido.
WS-Federation	Federación de servicios web.

Tabla 11: Glosario

6. Bibliografía

[1] Naciones Unidas. *ONU.* (septiembre de 2015). https://www.un.org/sustainabledevelopment/es/2015/09/la-asamblea-general-adopta-la-agenda-2030-para-el-desarrollo-sostenible

[2] Aenor. (2023). *Certificación Sostenibilidad energética en Centros de Proceso de Datos.* Obtenido de Aenor: https://www.aenor.com/certificacion/tecnologias-de-la-informacion/sostenibilidad-energetica-cpd

[3] Gómez-Cornejo Gilpérez, I. (octubre de 2020). *Interempresas*. Obtenido de https://www.interempresas.net/Instaladores/Articulos/262317-Climatizacion-y-eficiencia-energetica-en-la-industria-CPD.html

[4] Laurent, A., & Dal Maso, M. (marzo de 2020). *The Copenhagen Centre on Energy Efficiency*. Obtenido de https://c2e2.unepccc.org/wp-content/uploads/sites/3/2020/02/environmental-sustainability-of-data-centres-a-need-for-a-multi-impact-and-life-cycle-approach-brief-1-es.pdf

[5] Menéndez Menéndez, Mª Isabel. Lenguaje administrativo no sexista. Instituto Andaluz de la Mujer, 2006

[6] Lucidchart. (agosto de 2018). Lucidchart.com. Obtenido de https://www.lucidchart.com/blog/es/pros-y-contras-de-la-metodologia-de-cascada

[7] Sharma, P. (mayo de 2022). *Top software developemtn models to choose from.* Obtenido de https://cynoteck.com/es/blog-post/top-software-development-models-to-choose-from/

[8] Solera, S. (abril de 2022). *Las mejores metodologías para un correcto desarrollo de software*. Obtenido de occamagenciadigital: https://www.occamagenciadigital.com/blog/las-mejores-metodologias-para-un-correcto-desarrollo-de-software

[9] PMOinformatica.com (abril de 2015). Obtenido de http://www.pmoinformatica.com/2015/04/requerimientos-no-funcionales-una.html.

[10] Sommerville, Ian. *Ingeniería de software.* Pearson, 2012.

[11] Campbell B. & Mortimore C. & Jones M. (mayo de 2015). *Security Assertion Markup Language (SAML) 2.0 Profile for OAuth 2.0 Client Authentication and Authorization Grants*. Obtenido de https://datatracker.ietf.org/doc/html/rfc7522

[12] Hardt Ed. (octubre 2012). *The OAuth 2.0 Authorization Framework.* Obtenido de https://datatracker.ietf.org/doc/html/rfc6749

[13] Jones, M. & Hardt, D. (octubre 2012) *The OAuth 2.0 Authorization Framework: Bearer Token Usage*. Obtenido de https://datatracker.ietf.org/doc/html/rfc6750

[14] Sakimura N. & Bradley J. & and others (8 de noviembre de 2014). *OpenID Connect Core 1.0*. Obtenido de https://openid.net/specs/openid-connect-core-1_0.html

7. Recursos y herramientas

7.1. Software del estado del arte

- Okta: https://www.okta.com/
- Microsoft Azure Active Directory: https://www.microsoft.com/es-es/security/business/identity-access/azure-active-directory-single-sign-on
- Google Cloud Identity: https://cloud.google.com/architecture/identity/single-sign-on?hl=es-419
- Onelogin: https://www.onelogin.com/product/sso
- ForgeRock: https://www.forgerock.com/es/platform/access-management/sso
- Keycloak: https://www.keycloak.org/
- Shibboleth: https://www.shibboleth.net/
- Central Authentication Service: https://www.apereo.org/projects/cas
- Gluu: https://gluu.org/
- LemonLDAP::NG: https://lemonldap-ng.org/
- OpenAM: https://www.openidentityplatform.org/openam

7.2. Herramientas usadas en el TFM

- UOC. Herramientas para elaborar tu trabajo final: https://biblioteca.uoc.edu/export/sites/biblio/.galleries/documents/tabla-periodica-tf-es.pdf
- Sistema de virtualización: https://www.virtualbox.org/
- Imágenes de máquinas virtuales; https://www.osboxes.org/
- Generador de cronogramas: https://www.tomsplanner.es/
- Editor de texto: https://www.microsoft.com/es-es/microsoft-365/
- Sistema de despliegue de contenedores: https://www.docker.com/
- Repositorio de contenedores: https://hub.docker.com/

8. Anexos

8.1. Implementación de SP de SAML en PHP

A continuación, se muestra un ejemplo de implementación de un proveedor de servicios SAML en PHP utilizando la librería SAML-Toolkits

Código del fichero *acs.php*

```php
<?php
require __DIR__.'/vendor/autoload.php';

use OneLogin\Saml2\Auth;

try {
    $auth = new Auth(require __DIR__.'/settings.php');

    $auth->processResponse($_SESSION['AuthNRequestID'] ?? null);
    unset($_SESSION['AuthNRequestID']);
} catch (Exception $e) {
    echo $e->getMessage();
}
$errors = $auth->getErrors();

if (!empty($errors)) {
    echo '<p>', implode(', ', $errors), '</p>';
    exit();
}

if (!$auth->isAuthenticated()) {
    echo "<p>Not authenticated</p>";
    exit();
}

$attributes = $auth->getAttributes();
$nameId = $auth->getNameId();

echo '<h1>Identified user: '.htmlentities($nameId).'</h1>';

if (!empty($attributes)) {
    echo '<h2>User attributes:</h2>';
    echo '<table><thead><th>Name</th><th>Values</th></thead><tbody>';
    foreach ($attributes as $attributeName => $attributeValues) {
        echo '<tr><td>'.htmlentities($attributeName).'</td><td><ul>';
        foreach ($attributeValues as $attributeValue) {
            echo '<li>'.htmlentities($attributeValue).'</li>';
        }
        echo '</ul></td></tr>';
    }
    echo '</tbody></table>';
} else {
    echo 'No attributes found.';
}
```

Figura 24: Ejemplo de implementación de SAML en PHP. Fichero ACS

Código del fichero *settings.php*

Figura 25: Ejemplo de implementación de SAML en PHP. Fichero de configuración

8.2. Implementación de OpenID Connect en PHP

A continuación, se muestra un ejemplo de implementación de un cliente OIDC en PHP utilizando la librería [OpenID-Connect-PHP]

Código del fichero *index.php*

```php
<?php
require 'vendor/autoload.php';

$issuer = 'https://sso.uoc.edu/realms/SSO-Corporativo/';
$cid = 'cliente-jumbojett';
$secret = 'A2rv6vCv9MHOKhVpDYUyAcqQOQv0istv';
$oidc = new Jumbojett\OpenIDConnectClient($issuer, $cid, $secret);

$oidc->authenticate();
$nombre = $oidc->requestUserInfo('given_name');
$apellidos = $oidc->requestUserInfo('family_name');
$correo = $oidc->requestUserInfo('email');

?>

<html>
<head>
    <title>Ejemplo de OpenID Connect Client</title>
</head>
<body>

    <div>
        Hola <?= $nombre ?> - <?= $apellidos ?> (<?= $correo ?>)
    </div>

</body>
</html>
```

Figura 26: Ejemplo de implementación de OIDC en PHP