Universitat Oberta de Catalunya

uoc.edu

CloudDocs Signature Platform

Joan Font Rosillo
Màster universitari en Seguretat de les Tecnologies de la Informació i de les Comunicacions

Sistemes d'autenticació i autorització

Juan Carlos Fernández Jara
Víctor Garcia Font

31/12/2018


Aquesta obra està subjecta a una llicència de Reconeixement-NoComercial-SenseObraDerivada 3.0 Espanya de Creative Commons

Llicències alternatives (triar alguna de les següents i substituir la de la pàgina anterior)

A) Creative Commons:

Aquesta obra està subjecta a una llicència de Reconeixement-NoComercial-SenseObraDerivada 3.0 Espanya de Creative Commons

Aquesta obra està subjecta a una llicència de Reconeixement-NoComercial-CompartirIgual 3.0 Espanya de Creative Commons

Aquesta obra està subjecta a una llicència de Reconeixement-NoComercial 3.0 Espanya de Creative Commons

Aquesta obra està subjecta a una llicència de Reconeixement-SenseObraDerivada 3.0 Espanya de Creative Commons

Aquesta obra està subjecta a una llicència de Reconeixement-CompartirIgual 3.0 Espanya de Creative Commons

Aquesta obra està subjecta a una llicència de Reconeixement 3.0 Espanya de Creative Commons

B) GNU Free Documentation License (GNU FDL)

Copyright © 2019 Joan Font Rosillo

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free

Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
A copy of the license is included in the section entitled "GNU Free Documentation License".

C) Copyright

© Joan Font Rosillo
Reservats tots els drets. Està prohibit la reproducció total o parcial d'aquesta obra per qualsevol mitjà o procediment, compresos la impressió, la reprografia, el microfilm, el tractament informàtic o qualsevol altre sistema, així com la distribució d'exemplars mitjançant lloguer i préstec, sense l'autorització escrita de l'autor o dels límits que autoritzi la Llei de Propietat Intel•lectual.

FITXA DEL TREBALL FINAL

Títol del treball:	*CloudDocs Signature Platform*
Nom de l'autor:	*Joan Font Rosillo*
Nom del consultor/a:	*Juan Carlos Fernández Jara*
Nom del PRA:	*Víctor García Font*
Data de lliurament (mm/aaaa):	*01/2019*
Titulació o programa:	*Màster universitari en Seguretat de les Tecnologies de la Informació i de les Comunicacions*
Àrea del Treball Final:	*Sistemes d'autenticació i autorització*
Idioma del treball:	*Català*
Paraules clau	*signatura electrònica, oauth2, dropbox, google drive, eidas, cloud.*

Resum del Treball (màxim 250 paraules): *Amb la finalitat, context d'aplicació, metodologia, resultats i conclusions del treball*

El Treball Final de Màster CloudDocsSign tracta d'una aplicació demostrador que integra diversos proveïdors de documents *cloud* (Google Drive i Dropbox) i permet realitzar la signatura electrònica a aquests documents utilitzant el servei TrustedX de l'empresa Safelayer.

La necessitat d'aquesta aplicació sorgeix de la baixa penetració dels sistemes de signatura electrònica que hi ha en l'actualitat, ja que es requereix de dispositius addicionals per llegir els certificats qualificats que hi ha al DNIe o DNI 3.0.

La metodologia a seguir de projecte ha estat la següent: s'ha introduït el concepte, s'ha justificat la necessitat del projecte. A continuació s'ha definit l'àmbit i els requisits de l'aplicació demostrador i s'ha avaluat l'estat de totes les tecnologies que usa el projecte .La part programàtica ha centrat la major part del temps dedicat al projecte D'aquesta part programàtica ha sorgit l'aplicació demostrador. Al finalitzar s'ha avaluat si s'han complert els requisits i s'han plantejat línies de treball futur.

Com a resultat d'aquest TFM hem obtingut una aplicació totalment funcional que permet a un usuari realitzar la signatura electrònica sobre un document que tingui allotjat al *cloud.* També s'han sembrat les bases per a que es puguin integrar més proveïdors de documents.

La conclusió d'aquest projecte és que usant diferents serveis que podem trobar a Internet, com Google Drive, Dropbox i TrustedX aconseguim una "signatura qualificada" sobre un document. El procés de signatura és senzill i no requereix que l'usuari disposi de cap dispositiu addicional.

Abstract (in English, 250 words or less):

This project, CloudDocsSign, consists in a demo application that integrates múltiple document cloud providers (such as Google Drive o Dropbox) and allows the user to perform an electronic signature to this documents using TrustedX service, owned by Safelayer.

The need for this application arises from the low penetration of electronic signature systems. These systems require of physical devices for reading certificates that are built in DNIe or DNI 3.0.

The metodology that has driven the project development is the following: we have introduced the concept, we have justified the need of the project. Also we have defined the scope and the requirements that our demo application has to met. We have analyzed the status of all the tecnologie that our project uses. The programmatic part has focused most of the time spent on the project. In this programmatic part, the demo application has emerged. At the end of the project we have evaluated whether the requirements have been accomplished and we have proposed future work.

The result of this project is our demo application that is able to sign a document from a cloud provider, whatever the document provider is. The application has been prepared to integrate more cloud document providers.

The conclusion of this project is that using several services that we can found on the Internet, such as Google Drive, Dropbox or TrustedX we can perform a "qualified electronic signature" on a document. The signature process is simple and does not require the user to have any additional device.

Índex

Llista de figures

1. Introducció

1.1 Context i justificació del Treball

Avui en dia sempre que es vol fer una signatura avançada o qualificada, s'ha de fer amb un dispositiu qualificat de signatura o en local, és a dir, a l'ordinador de l'usuari.

A l'estat espanyol, la manera que té un ciutadà per usar una firma avançada o qualificada és fent ús del DNIe [1] o DNI 3.0 [2]. En el cas del primer suport (s'expedeix a totes les Oficines d'Expedició fins al desembre de 2015) és necessari tenir un dispositiu addicional per llegir el certificat reconegut que porta el xip electrònic. El fet de precisar d'un dispositiu addicional per a poder usar el certificat ha provocat que no s'hagi estès l'ús del DNIe. Amb l'entrada en vigor del DNI 3.0 s'elimina aquesta barrera d'haver de disposar d'un dispositiu de lectura específic, ja que aquest nou format de document permet la lectura mitjançant el protocol NFC [3]. Però encara és necessari tenir un dispositiu de lectura d'NFC (molt més estès que el lector del DNIe) i de portar el DNI al moment de voler realitzar una firma electrònica.

Amb l'auge d'internet són més les accions quotidianes que traslladem del mon analògic al món digital. Una d'aquestes accions és la firma de documents. El DNIe o DNI 3.0 incorporen certificats electrònics reconeguts que permeten realitzar aquesta acció. És a dir, que mitjançant el document d'identitat es pot signar digitalment un document.

Una manera d'esmenar aquesta barrera és portar la firma al núvol. Els documents els podem tenir situats al núvol. Existeixen diversos proveïdors que ens permeten emmagatzemar qualsevol tipus de document al núvol sense cost algun. Ens queda solucionar l'altra branca: l'emmagatzemat de les claus al núvol. Amb productes com TrustedX de Safelayer podem custodiar les claus al núvol com si de recursos es tractessin. Aquest servei compleix amb la normativa europea eIDAS [4], fet que permet reconèixer les signatures de tots els estats membres de la Unió als diferents països, però sobretot, avança en la possibilitat de la signatura avançada i qualificada al núvol.

1.2 Objectius del Treball

L'objectiu principal del treball és obtenir una aplicació demostrador, anomenada CloudDocs Signature Platform (abreviat CloudDocsSign), que permeti signar un document resident al núvol amb una clau emmagatzemada també al núvol. Aquest és l'objectiu final del projecte. Juntament amb aquest objectiu global, també hem de tenir en compte els següents objectius:

- Permetre a l'usuari gestionar els documents que vol signar des d'una única plataforma.

- Obtenir un servei que permeti signar documents de manera confidencial, que el document a signar no s'enviï a cap tercer. Tot el procés es farà a la plataforma CloudDocs Signature Platform.

- Obtenir una plataforma que permeti signar documents al núvol, independentment del proveïdor d'aquests mitjançant un proveïdor de claus. Dissenyar una interfície el més genèrica possible per a futures integracions de proveïdors de documents.
- Integrar-se almenys amb dos proveïdors de documents al núvol. En aquest cas Google Drive i Dropbox

- Integrar-se amb TrustedX de Safelayer per a la gestió de les claus.

- Disposar d'una interfície web que ens faciliti la signatura de documents.

1.3 Enfocament i mètode seguit

La metodologia a seguir passa per definir els requisits de l'aplicació demostrador des d'un punt de vista que no es tingui en compte quin proveïdor de documents es vol usar. Partint d'aquesta premissa, el treball es desenvoluparà de la següent manera:

- **Definició del pla de treball:** en aquesta primera fase es defineixen els objectius i es justifica el perquè d'aquest treball. A més s'exposa la metodologia utilitzada i es mostra el cronograma d'execució del projecte que marcarà una fita temporal per a cada objectiu definit. Per a finalitzar també es defineixen els entregables del projecte.

- **Anàlisi de requisits:** en aquesta segona fase es definiran els requisits de l'aplicació. És a dir, què ha de fer la nostra aplicació. És aquí on es detallarà també l'abast, fins on ha de cobrir l'aplicació. El que no es pugui cobrir es pot plantejar com a treball futur. Aquest punt serà el que regirà principalment la fase d'execució del projecte.

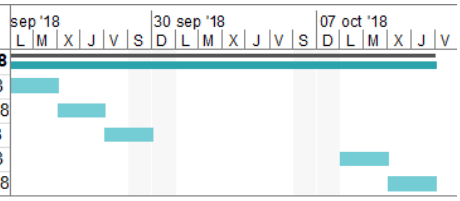
- **Estat de l'art:** en aquesta tercera fase s'avaluarà en quin estat es troba el tema del treball, és a dir, la signatura de documents electrònics al núvol. Es mostraran quines solucions hi pot haver i quines mancances podrien tenir que justifiquin aquest projecte.

- **Desenvolupament de l'aplicació demostrador:** aquesta fase contindrà el bloc de feina més gros del projecte. Comprendrà tota la part programàtica de l'aplicació demostrador sempre tenint en compte els requisits i objectius definits en la fase de definició del pla de treball. El desenvolupament de l'aplicació s'anirà fent per fases seguint un sistema incremental d'entregables marcat per fites temporals.

- **Conclusions i treball futur:** en aquesta darrera fase, com a tancament del projecte, es presenten les conclusions que es deriven de tots els apartats anteriors i s'identifiquen possibles línies de treball futur que no s'han pogut abastar en aquest projecte, per a que pugui ser utilitzat com a treball futur.

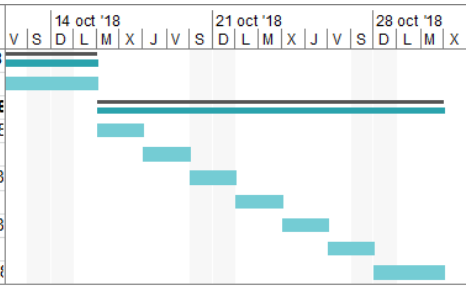
1.4 Planificació del Treball

Id	Nombre de tarea	Duración	Comienzo
1	**Definició del pla de treball**	**18 días**	**lun 24/09/18**
2	Contextualització del projecte	2 días	lun 24/09/18
3	Justificació del projecte	2 días	mié 26/09/18
4	Definició dels objectius	2 días	vie 28/09/18
5	Definició de la metodologia de trbeall	2 días	lun 08/10/18
6	Definició del pla temporal	2 días	mié 10/10/18

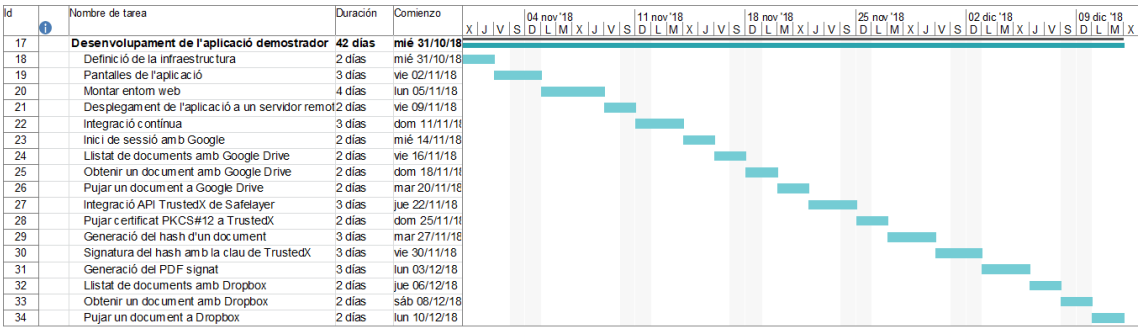


Il·lustració 1: Pla del treball

Id	Nombre de tarea	Duración	Comienzo
7	**Anàlisi de requisits**	**4 días**	**vie 12/10/18**
8	Anàlisi de requisits	4 días	vie 12/10/18
9	**Estat de l'art**	**15 días**	**mar 16/10/18**
10	Anàlisi de sol·lucions semblants	2 días	mar 16/10/18
11	Anàlisi de la plataforma TrustedX	2 días	jue 18/10/18
12	Signatura de documents PDF	2 días	sáb 20/10/18
13	Normativa europea eIDAS	2 días	lun 22/10/18
14	Ús de l'API de Dropbox	2 días	mié 24/10/18
15	Ús de l'API de Google Drive	2 días	vie 26/10/18
16	Anàlisi i elecció de framework web per Java	3 días	dom 28/10/18

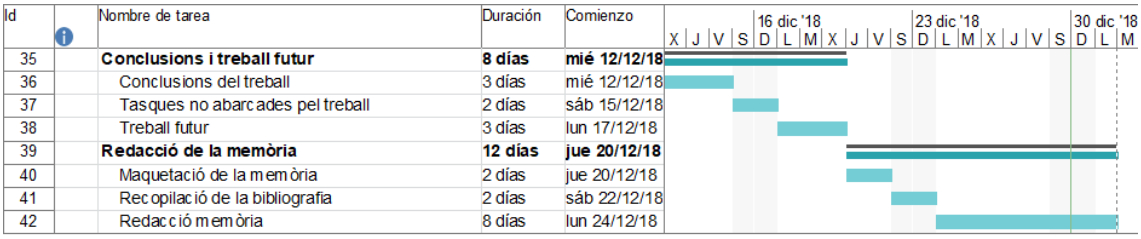


Il·lustració 2: Anàlisi de requisits i estat de l'art



Id	Nombre de tarea	Duración	Comienzo
17	**Desenvolupament de l'aplicació demostrador**	**42 días**	**mié 31/10/18**
18	Definició de la infraestructura	2 días	mié 31/10/18
19	Pantalles de l'aplicació	3 días	vie 02/11/18
20	Montar entorn web	4 días	lun 05/11/18
21	Desplegament de l'aplicació a un servidor remot	2 días	vie 09/11/18
22	Integració contínua	3 días	dom 11/11/18
23	Inici de sessió amb Google	2 días	mié 14/11/18
24	Llistat de documents amb Google Drive	2 días	vie 16/11/18
25	Obtenir un document amb Google Drive	2 días	dom 18/11/18
26	Pujar un document a Google Drive	2 días	mar 20/11/18
27	Integració API TrustedX de Safelayer	3 días	jue 22/11/18
28	Pujar certificat PKCS#12 a TrustedX	2 días	dom 25/11/18
29	Generació del hash d'un document	3 días	mar 27/11/18
30	Signatura del hash amb la clau de TrustedX	3 días	vie 30/11/18
31	Generació del PDF signat	3 días	lun 03/12/18
32	Llistat de documents amb Dropbox	2 días	jue 06/12/18
33	Obtenir un document amb Dropbox	2 días	sáb 08/12/18
34	Pujar un document a Dropbox	2 días	lun 10/12/18

Il·lustració 3: Desenvolupament del projecte



Id	Nombre de tarea	Duración	Comienzo
35	**Conclusions i treball futur**	**8 días**	**mié 12/12/18**
36	Conclusions del treball	3 días	mié 12/12/18
37	Tasques no abarcades pel treball	2 días	sáb 15/12/18
38	Treball futur	3 días	lun 17/12/18
39	**Redacció de la memòria**	**12 días**	**jue 20/12/18**
40	Maquetació de la memòria	2 días	jue 20/12/18
41	Recopilació de la bibliografia	2 días	sáb 22/12/18
42	Redacció memòria	8 días	lun 24/12/18

Il·lustració 4: Conslusions i treball futur i redacció de la memòria

1.5 Breu sumari de productes obtinguts

El producte obtingut en aquest treball és una aplicació demostrador totalment funcional que permetrà a un usuari signar documents provinents de serveis d'allotjament d'arxius al núvol usant les claus que ell mateix ha proporcionat.

2. Anàlisi de requisits

En aquest capítol de la memòria ens centrarem en analitzar els requisits que ha de complir l'aplicació demostrador. En primer lloc es farà una descripció a alt nivell del que ha de fer el servei i posteriorment s'enumeraran de manera detallada els requisits que ha de complir.

2.1 Anàlisi a alt nivell

L'aplicació demostrador que esdevé com a resultat d'aquest treball consisteix en una aplicació web, que ha proporcionar a un usuari la capacitat de signar documents al núvol. El procés s'ha de dur a terme al núvol, sense que sigui necessari per a l'usuari haver-se d'instal·lar cap programari addicional al seu ordinador, telèfon mòbil o tauleta digital.

El servei ha de permetre a l'usuari signar els documents allotjats a Google Drive o Dropbox, sense cap necessitat d'haver de carregar un document des del dispositiu. Per a la gestió de les claus s'usarà el servei TrustedX de Safelayer. En aquest cas sí que s'ha de permetre a l'usuari carregar un certificat en format PKCS#12 [5] des del dispositiu en el que es troba. Una vegada la clau estigui carregada al servei TrustedX, l'usuari ha de poder seleccionar quin document vol signar i amb quina clau ho desitja fer. Com a resultat del procés, es genera un document signat amb el certificat escollit i es diposita al mateix proveïdor del qual s'ha triat el document.

A més dels requisits funcionals que ha de complir l'aplicació, s'estableixen uns requisits més tècnics que ha de cobrir. Un dels objectius del projecte és que l'aplicació ha de ser capaç de gestionar documents de la mateixa manera, sigui quin sigui el proveïdor d'arxius al núvol. D'aquest objectiu en sorgeix un requisit: l'aplicació ha de ser capaç de signar un document del núvol sigui quin sigui el seu origen.

2.2 Anàlisi detallada

A nivell funcional tenim els següents requisits:

- L'usuari ha de poder iniciar sessió al servei amb el seu compte de Google.
- L'usuari ha de poder pujar una clau en format PCKS#12 al servei TrustedX de Safelayer.
- L'usuari ha de poder eliminar una clau allotjada al servei TrustedX de Safelayer.
- L'usuari ha de poder llistar els documents PDF que té allotjats al servei Google Drive.
- L'usuari ha de poder llistar els documents PDF que té allotjats al servei Dropbox.
- L'usuari ha de poder signar amb les claus custodiades al servei TrustedX de Safelayer qualsevol document allotjat a un dels dos proveïdors que s'ofereixen.

A nivell de sistema tenim els següents requisits:

- L'aplicació ha de gestionar la firma d'un document independentment de la seva provinença.
- L'aplicació ha de ser compatible amb la normativa europea eIDAS.

2.3 Fluxos de l'aplicació

L'aplicació es pot dividir en tres fluxos diferents: inici de sessió a la plataforma, gestió de claus i signatura de documents. En cada un d'aquests fluxos interactuen diferents subsistemes.

A continuació es descriuran els tres fluxos i es veurà quines parts interaccionen entre elles.

Inici de sessió

- L'usuari entra a la plana d'inici de sessió de l'aplicació demostrador
- L'usuari selecciona iniciar sessió amb Google
- L'usuari autoritza a Google a cedir-nos les dades necessàries per a poder iniciar-li la sessió: el seu nom i la seva direcció de correu electrònic.
- L'usuari retorna a la nostra aplicació amb una sessió iniciada

Internament dins l'aplicació quan l'usuari ha seleccionat iniciar sessió amb Google s'ha fet una petició a Google per obtenir un *token* per a l'usuari. La interacció amb aquest sistema es pot resumir amb el flux d'autenticació i autorització amb l'API de Google. Posteriorment l'usuari pot seleccionar Google com a proveïdor de documents al núvol. En aquest cas es tornaria a demanar accés al Google Drive de l'usuari i aquest ens hauria de tornar a donar autorització; en cap cas demanarem accés a Google Drive si l'usuari al final vol signar documents provinents d'un altre proveïdor.

Gestió de claus

La gestió de claus inclou varis fluxos. Com analitzarem posteriorment amb més detall, l'API de TrustedX diferencia la gestió de claus ja existents del registre de noves claus. En la descripció dels següents fluxos suposarem que l'usuari ja ha iniciat sessió amb Google i els passos necessaris s'ometran de la descripció.

Registre d'una nova clau

- L'usuari es dirigeix a l'apartat de gestió d'identitats.
- L'usuari selecciona l'acció de registrar una nova clau a la pantalla principal de la plana de gestió de claus.
- L'usuari és redirigit a la plana d'autorització de TrustedX on se li demana autorització per a l'aplicació demostrador per a registrar una nova clau.
- L'usuari selecciona un fitxer PCKS#12 mitjançant un selector de fitxers.
- L'usuari especifica la contrasenya del fitxer PCKS#12 seleccionat a un camp de tipus contrasenya.
- L'usuari selecciona el rol de la clau (estudiant o professor)

- L’usuari envia el formulari.
- Es realitza la petició de registrar una nova clau. En funció del resultat:
    - Si la petició resulta correcta es redirigeix l’usuari al llistat de claus on es pot veure la clau nova.
    - Si la petició no resulta correcta es descriu l’error que ha succeït.

Eliminació d’una clau

- L’usuari es dirigeix a l’apartat de gestió de claus.
- L’usuari selecciona eliminar una clau en concret
- Si no havia autoritzat prèviament l’aplicació demostrador a gestionar les seves claus, se li demana autorització.
- Es realitza una petició per eliminar la clau. En funció del resultat:
    - Si la petició resulta correcta es mostra un missatge d’èxit enunciant que s’ha eliminat la clau i aquesta s’elimina de la llista.
    - Si la petició no resulta correcta es descriu l’error que ha succeït.

En aquests dos fluxos s’ha descrit el comportament de l’API de TrustedX. Aquest està analitzat amb més detall a l’apartat d’Integració amb l’API de TrustedX.

Signatura de documents

El flux de signatura de documents inclou dos subsistemes diferents. L’ API de Google Drive o Dropbox i l’API de TrustedX. Encara que es tracta d’un únic flux, el dividirem en dos: la selecció del document a signar i la signatura del document.

Selecció del document a signar

- L'usuari selecciona quin proveïdor vol usar per obtenir els documents. Google Drive o Dropbox.

- En ambdós casos es demana autorització per usar l'API del proveïdor seleccionat.

- L'usuari és redirigit a una plana on pot veure els documents PDF que té al seu proveïdor de documents al núvol.

Signatura del document

- L'usuari selecciona el document que desitgi signar.

- L'usuari selecciona quina clau vol usar per a signar el document. S'ha de recordar que per llistar les claus l'usuari ha d'haver autoritzat prèviament l'aplicació demostrador a gestionar les seves claus de TrustedX.

- L'usuari selecciona l'acció de signar.

- Se li demana autorització a l'usuari per utilitzar la clau escollida.

- Es realitza la petició de signatura del document. En funció del resultat d'aquesta petició:

    - Si es satisfactòria es mostra el nom del document signat i un missatge enunciant l'èxit de la operació.

    - Si resulta incorrecta es descriu el resultat de l'operació mostrant el missatge d'error que ens ha retornat l'API.

3. Estat de l'art

En aquest apartat de la memòria s'analitzarà l'estat de l'art que envolta el treball. En concret es veuràn quines solucions semblants hi ha al mercat actualment, si existeixen serveis que ofereixin el mateix que la nostra plataforma Cloud Docs Signature Platform i com es ve fent aquest procés. També s'analitzarà l'estat actual de les signatures de documents PDF i quines solucions programàtiques s'ofereixen.

Un punt fort d'aquest treball és que la plataforma TrustedX de Safelayer compleix amb la normativa europea eIDAS. Analitzarem què comporta aquesta normativa. Les tres API que integrarem es basen en OAuth 2.0 [6] per oferir autenticació i autorització, veurem en què consisteix aquest protocol i quina modalitat s'usa en cada API que consumirem. També es farà una breu anàlisi del que ofereixen les API de Google Drive i Dropbox per integrar-les al nostre projecte. Per finalitzar veurem quin és l'estat dels *framework* web de Java i n'elegirem un com a base del nostre projecte.

3.1 Anàlisi de solucions semblants

En l'actualitat, els processadors de text amb més quota d'ús incorporen mecanismes de signatura electrònica de documents. Avui en dia és possible signar un document amb Microsoft Word o LibreOffice.

A més a més existeixen solucions que permeten signar directament documents PDF. Entre aquestes solucions que hem anomenat solucions *offline* podem trobar: PDFCreator [7], DigiSigner [8], Sinadura [9], JSignPdf [10], iSafePDF [11], XolidoSign [12], EcoFirma [13], etc... Aquestes solucions requereixen tenir tant el document PDF com el certificat a l'hora de fer la firma. És a dir, que quan es desitgi signar el document s'haurà de disposar del certificat amb el que es vol signar-lo.

A banda d'aplicacions d'escriptori també existeixen aplicacions web que permeten realitzar una signatura d'un document PDF amb un certificat electrònic. Un exemple d'aquests és Signaturit [14]. Aquest servei ofereix signatura electrònica de documents, però no amb un certificat PCKS#12. A més, la integració amb Google Drive o Dropbox que ofereix Signaturit, entre d'altres, requereix seleccionar el document al portal del proveïdor de documents al núvol i no a la mateixa aplicació Signaturit. Aquesta plataforma és compatible amb la normativa europea eIDAS.

A més de Signaturit també existeix DocuSign [15]. Aquest servei ofereix característiques semblants a Signaturit. Ofereix integracions amb serveis d'emmagatzemament de documents al núvol però d'una manera diferent. A més ofereix una API per a què es puguin desenvolupar solucions personalitzades sobre la seva plataforma. Com Signaturit, DocuSign també és compatible amb la normativa europea eIDAS.

3.2 Anàlisi de la plataforma TrustedX de Safelayer

Per a facilitar la gestió de claus dels usuaris i signar els documents PDF usarem el servei TrustedX de Safelayer. Aquest servei de l'empresa Safelayer permet realitzar firmes de documents amb claus custodiades al núvol.

TrustedX ens ofereix una API REST amb la que podem realitzar la gestió de claus custodiades al seus servidors. Aquesta gestió inclou les següents accions: registrar una clau, administrar les ja existents (llistar-les, modificar-les i eliminar-les) i signar un *hash* amb una clau registrada.

Per interactuar amb l'API de TrustedX és necessari estar registrat prèviament a la plataforma TrustedX. El procés de registre es realitza de forma presencial amb un nivell LoA (*Level of Assurance* [16]) alt. Cada acció que es desitgi realitzar s'ha d'acompanyar d'un *token* d'autenticació prèviament obtingut per l'usuari. És a dir, les peticions HTTP han de dur a la capçalera *Authorization* el *token* que identifica a l'usuari. Aquest *token* s'obté usant el protocol OAuth2 amb el flux *Authorization Code Grant* [17].

Cada acció de l'API requereix un permís associat. Quan es demana un *token* d'autenticació, es demana amb uns permisos i l'usuari final és el responsable d'autoritzar l'aplicació a usar el seu compte o no. Si el *token* associat a la petició no té els permisos necessaris que l'acció que es vol realitzar requereix, l'acció no es pot realitzar.

En la nostra aplicació demostrador haurem de realitzar les accions enunciades anteriorment. Cada una d'aquestes tres accions té un permís associat. Els permisos associats a cada acció que permet fer l'API són els següents:

Scope	**Accions que permet realitzar**
urn:safelayer:eidas:sign:identity:register	Registrar una clau al sistema
urn:safelayer:eidas:sign:identity:manage	Llistar, actualitzar o eliminar claus del sistema.
urn:safelayer:eidas:sign:identity:use:server	Permet usar una clau per a una firma. Un fet important és que cada vegada que es vulgui realitzar una nova firma s'ha de renovar el *token*, ja que en aquest cas es tracta d'un token d'un sol ús.

Aquests *tokens* tenen una caducitat, és a dir, tenen una durada determinada. Si quan es vol realitzar una petició – per exemple de registrar una clau – el *token* caduca, aquest s'haurà de tornar a demanar a l'usuari final.

Un aspecte important a considerar sobre l'API de TrustedX és que aquesta compleix amb la normativa europea eIDAS que veurem a continuació.

3.3 Normativa europea eIDAS

La normativa europea eIDAS és un nou Reglament Europeu que regula la identificació electrònica i estableix les portes per als serveis de confiança relatius a les transaccions electròniques. La normativa va entrar en vigor l'1 de juliol de 2016. Aquest reglament és prou important perquè opera sobre les següents àrees: firma electrònica, establiment de segells i marques de temps, documents electrònics i serveis d'entrega electrònica registrada o correu electrònic certificat i serveis de certificat per a autenticació de llocs web. En el nostre cas l'escenari d'estudi és la firma electrònica.

Com a antecedent d'aquesta nova normativa tenim la Directiva 1999/93/CE [18] que fou la primera que regulava els serveis de la firma electrònica a la Unió Europea. El problema d'aquesta directiva era que cada estat de la Unió l'interpretava a la seva manera. Això va suposar una complicació per al reconeixement de la validesa de les firmes electròniques entre països de la Unió i els seus respectius tribunals. D'aquest problema sorgeix la normativa eIDAS que estableix un nou marc jurídic comunitari per a la firma electrònica.

Respecte a la firma electrònica, el nou reglament manté els tres tipus de firma electrònica que figuraven a la Directiva 1999/93/CE: firma electrònica, firma electrònica avançada i firma electrònica avançada basada en un certificat reconegut.

El que ha canviat és el que en la Directiva anterior s'anomenava firma electrònica avançada basada en un certificat reconegut, amb el nou reglament s'anomena firma electrònica qualificada. Respecte a la Directiva, aquesta firma electrònica qualificada serà reconeguda per tots els estats membres de la Unió, independentment de la seva provinença.

A l'article 25 [19] de la nova normativa europea eIDAS s'especifica que no es denegaran efectes jurídics a una firma electrònica pel fet de ser electrònica o perquè no compleixi els requisits d'una firma electrònica qualificada. També s'esmenta que una firma electrònica qualificada tindrà el mateix efecte que una firma manuscrita.

A l'article 26 [20] del mateix reglament s'estableixen els requisits legals per a una firma electrònica avançada:

- Ha d'estar vinculada al firmat de manera única
- Ha de permetre la identificació del firmant
- La firma ha d'haver estat creada utilitzant mitjans de creació que el firmant pugui usar amb alt nivell de confiança i sota el seu control exclusiu (que no pugui controlar-se per tercers)
- Ha d'estar vinculada amb les dades firmades de manera que qualsevol alteració sobre aquestes pugui ser detectada.

3.4 Signatura de documents PDF

La signatura de documents PDF es basa en l'estàndard ISO 32000-1 [22]. PAdES [21] (*PDF Advanced Electronic Signature*) estableix unes restriccions sobre l'estàndard ISO-32000-1 i PDF fent-lo compatible amb els requisits de la signatura electrònica avançada.

Mentre que el PDF i la ISO 32000-1 estableixen un marc de treball per firmar els documents, PAdES especifica perfils concrets fent la signatura compatible amb la normativa europea eIDAS vista en el punt anterior. Una signatura PAdES té la consideració d'una signatura electrònica qualificada. Les signatures a documents PDF estan incorporades dins el fitxer PDF signat. Així com una signatura manuscrita està incorporada dins el propi paper signat, una

signatura en un fitxer PDF emula el mateix escenari. Un avantatge de PAdES és que està incorporat en la majoria dels lectors PDF.

En la part programàtica trobem la llibreria iText per a Java i .NET que ens permet realitzar la signatura PAdES sobre el document PDF. En el nostre cas, com que la signatura del *hash* del document es farà usant la plataforma TrustedX de Safelayer, el que farem serà usar la llibreria iText per aplicar la signatura ja feta al document PDF. Aquesta implementació la veurem en l'apartat 4 d'aquesta memòria.

3.5 OAuth 2.0

Les tres APIs que integrarem en la nostra aplicació demostrador tenen un factor en comú. Les tres usen OAuth 2.0 com a protocol d'autenticació i autorització.

OAuth és un estàndard obert per delegar l'accés a les dades. El seu ús està estès a Internet per permetre als llocs web o aplicacions accedir a la informació de l'usuari sense necessitat de compartir contrasenyes. És a dir, mitjançant aquest protocol, podem autoritzar un tercer a accedir a les nostres dades sense necessitat de proporcionar-li les credencials d'accés.

OAuth contempla diferents fluxos per atorgar l'accés a les dades a un tercer. En el nostre cas el flux que ens aplica és *Authorization Code Grant*. Podem trobar més detalls d'aquest flux al punt 4.1 de l'RFC-6749 [1]. Per obtenir autorització es redirigeix l'usuari al portal d'on es vol obtenir informació i se li demana el consentiment per accedir a unes dades específiques (en OAuth s'anomena *scope*). Un cop l'usuari ens ha concedit permís per accedir a aquestes dades es realitza una cridada *callback* a una URL que l'aplicació ha proporcionat i aquest *callback* conté un codi que s'intercanvia per un *token* d'accés (*access token*) que identifica l'usuari.

Tant Google Drive, com Dropbox, com TrustedX implementen aquest protocol per concedir permís per usar les seves dades. Com veurem a continuació cada una d'aquestes tres API té les seves particularitats a l'hora de de demanar el *token* d'autorització, però les tres es basen en el flux que acabem de comentar en l'anterior paràgraf.

3.6 Ús de l'API de Google Drive

La nostra aplicació demostrador es basa en dos pilars: l'aplicació d'una signatura a un document PDF i l'ús de proveïdors d'emmagatzemament de documents al núvol. En aquest cas ens centrarem en com podem fer ús de l'API de Google Drive [23] per obtenir els documents que l'usuari té allotjats al servei.

A més, per poder operar amb les dades de l'usuari que estigui usant la nostra aplicació demostrador haurem d'obtenir un *token* d'autenticació que ens permeti usar l'API en nom seu. Com succeeix amb TrustedX també usarem el protocol OAuth2 per obtenir un *token* de l'usuari. Les accions que hem de realitzar amb l'API de Google Drive es limiten a llistar fitxers, descarregar-ne i pujar-ne de nous. Donats aquests requisits i comparant-los amb els *scopes* d'OAuth2 disponibles per a l'API de Google Drive veiem que els permisos que hem de demanar a l'usuari per a operar amb el seu compte són els següents:

- **`https://www.googleapis.com/auth/drive:`** permet accés complet a totes les funcionalitats de l'API de Drive.

Si bé és cert que Google recomana demanar a l'usuari els *scopes* necessaris per realitzar les accions que es volen fer, en el nostre cas hem de demanar accés complet ja que és l'única manera de tenir accés de lectura i escriptura sobre el Google Drive de l'usuari.

Una alternativa a aquest model de permisos és demanar en primer lloc el permís `https://www.googleapis.com/auth/drive.readonly` i en cada operació d'escriptura (cada vegada que es signi un document s'ha de pujar al Drive) sol·licitar a l'usuari el permís `https://www.googleapis.com/auth/drive.file.` La diferència que hi trobem entre la primera opció és que aquesta segona alternativa pot resultar més molesta per a l'usuari ja que ha de concedir permís a l'aplicació demostrador dos cops per signar un document en lloc d'un cop. L'avantatge que té aquesta segona opció sobre la primera és que es garanteix que l'aplicació no podrà alterar en cap cas el contingut de l'usuari, sinó que només podrà crear fitxers que ell hagi autoritzat específicament.

Per finalitzar s'ha comentar que l'API de Google Drive disposa d'un SDK per Java per facilitar les operacions amb aquestes.

3.7 Ús de l'API de Dropbox

Un altre avantatge de la nostra aplicació demostrador és que ha de ser capaç de manejar documents independentment de la seva provinença. Per aquest motiu també ens integrarem amb l'API de Dropbox [25] per obtenir documents PDF.

Al igual que amb Google Drive, l'API de Dropbox també usa el mecanisme OAuth2 per autenticar les peticions. El flux usat també és el de *Authorization Code Grant*, per la qual cosa les tres API que hem d'integrar al nostre projecte contemplen el mateix flux de l'estàndard.

El que sí que varia respecte a l'API de Google és el sistema de permisos. Enlloc d'especificar els permisos que es demanen a la petició d'autenticació aquests estan implícits a l'aplicació OAuth de Dropbox que usarà la nostra aplicació demostrador. És a dir, s'estableixen un cop creada l'aplicació i no es poden modificar en funció de les necessitats de la petició que s'hagi de realitzar. Dropbox ofereix diferents permisos entre els quals hi podem trobar aquests dos:

- ***App Folder*:** ens dona accés complet a una carpeta específica per a la nostra aplicació. Per exemple podríem crear el directori CloudDocsSign i l'aplicació tindria accés complet a tot aquest directori i subdirectoris.

- ***Full Dropbox*:** accés complet a tots els fitxers de l'usuari.

Ens trobem en el mateix escenari que amb Google Drive. Si s'opta per donar més privacitat a l'usuari, la tasca d'aquest es complica perquè en aquest cas és ell el responsable de situar els fitxers que vulgui signar dins el directori de l'aplicació. Per altra banda, si s'opta per donar accés complet a tots els fitxers de Dropbox l'ús de l'aplicació es simplifica però la privacitat de l'usuari queda compromesa, ja que l'aplicació demostrador té accés a tots els fitxers. En ambdós casos se'ns permet accés de lectura i escriptura.

Com Google Drive, l'API de Dropbox també compta amb un SDK per facilitar les operacions amb aquesta.

3.8 Anàlisi i elecció d'un framework web Java

La nostra aplicació demostrador es basa en un entorn on l'usuari interactua amb una interfície web. Per facilitar el disseny d'aquesta aplicació web ens recolzarem en un *framework*. En cap cas l'objectiu d'aquest projecte és seleccionar el millor *framework* sinó que aquest ens ha de permetre centrar-nos en les tasques que sí que entren dins els objectius del projecte, com: signatura de documents PDF i interacció amb les API de Google Drive, Dropbox i TrustedX. Per aquest motiu el *framework* no ha de ser en cap cas una barrera, sinó que ens ha de proporcionar solucions fàcils a problemes complexes.

Per aquest motiu, els criteris en els que ens basarem per elegir un *framework* web de Java seran la facilitat d'us, la mida de la comunitat de desenvolupadors que hi ha a darrera i l'extensibilitat. Per aquest darrer concepte s'ha d'entendre extensibilitat com a poder afegir característiques sense que esdevingui una tasca complexa.

S'han analitzat els següents *frameworks:* Grails [25], Spring [26], Play! [27], i Vert.x [28] . Finalment s'ha elegit Spring. Els motius pels quals s'ha elegit aquest *framework* és la corba d'aprenentatge. No requereix de molts coneixements sobre el *framework* per fer aplicacions web senzilles, i disposa d'una gran comunitat on resoldre els dubtes sobre el seu funcionament. Per això el nostre projecte es basarà en Spring.

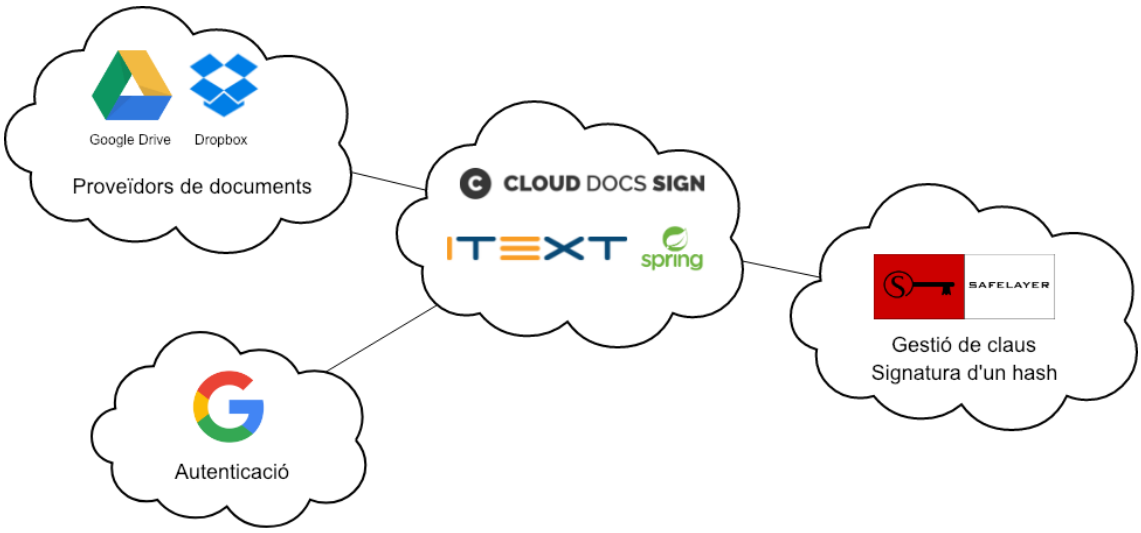
Els altres *frameworks* s'han descartat per voler cobrir massa aspectes (com Grails), per l'escassa documentació i *plugins* (Play!) per la mida de la comunitat (Vert.x).

4. Desenvolupament de l'aplicació demostrador

En aquest apartat veurem el més important de tot el procés de desenvolupament de l'aplicació demostrador. No s'entrarà en detall de cada mòdul o cada característica desenvolupada, sinó que es comentaran els aspectes més destacats de cada un d'aquests. Cal recordar que l'aplicació demostrador es compon de tres parts. En primer lloc tenim tota la gestió de documents de diferents proveïdors *cloud* (Google Drive i Dropbox). En segon lloc tenim tot el que comporta la gestió de claus amb TrustedX (llistat, inserció i eliminació). Per finalitzar tenim la signatura d'un document d'un proveïdor integrat amb una clau allotjada a TrustedX.

Totes aquetes característiques les hem d'englobar dins el que és el desenvolupament d'una aplicació web que inclou com a mecanisme d'autenticació l'inici de sessió amb Google. A més a més, l'aplicació està desenvolupada mitjançant un *framework* que proporciona eines per fer aquesta tasca web més àgil.

Una visió del projecte, sense entrar en detalls, és la que es pot observar a continuació. CloudDocsSign es compon de tres components principals: els proveïdors de documents *cloud*, el proveïdor de claus i signatura de *hashes* i l'autenticació. El punt de connexió d'aquests tres components és la nostra aplicació demostrador. Aquesta es basa en *Spring* com a *framework* web i iText per a la signatura de documents PDF.



Il·lustració 5: Components de CloudDocsSign

A continuació s'explicaran cada un d'aquests components que interactuen amb el sistema i com s'han integrat amb la plataforma.

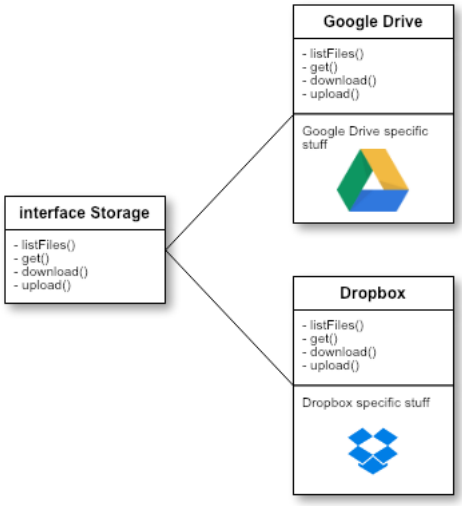
4.1 Proveïdors de documents

Una de les principals característiques de l'aplicació demostrador és que pugui funcionar amb qualsevol proveïdor de documents *cloud.* L'aplicatiu té integrats Google Drive i Dropbox, però tal i com està dissenyat, afegir-ne d'altres és quasi trivial.

S'ha creat el concepte d'emmagatzemament, que no és més que una capa d'abstracció per damunt de qualsevol proveïdor de documents que permet l'aplicació demostrador interactuar de la mateixa manera amb Google Drive que amb Dropbox. Així la nostra aplicació no interactua directament amb els SDK de Google Drive o Dropbox, sinó que ho fa a través de la interfície d'emmagatzemament. Aquesta té els mètodes necessaris per a què la nostra aplicació pugui fer la signatura d'un document:

- Llistar els documents del proveïdor
- Descarregar un document del proveïdor
- Pujar un nou document al proveïdor

Internament existeixen dues implementacions d'aquests magatzems de dades. Un per Google Drive i un altre per Dropbox. Cada un d'aquests implementa l'accés als fitxers mitjançant les seves corresponents API REST. Com que aquestes API no són idèntiques, cada una té les seves particularitats i mètodes d'accés, no disposar d'aquesta manera homogènia d'accés a les dades incrementaria la complexitat de l'aplicació. Però la nostra aplicació accedeix a aquests mètodes de la mateixa manera, independentment de la seva implementació, a través d'aquestes interfícies.



Il·lustració 6: Interfície d'emmagatzemament

En el cas que es desitgi incorporar-hi més proveïdors només s'ha d'implementar l'accés a aquests proveïdors mitjançant les seves corresponents

API. L'aplicació serà capaç de funcionar a través de la interfície d'emmagatzemament.

Per a cada un dels proveïdors integrats hem hagut d'implementar per una banda les funcions d'accés als fitxers i per l'altra banda el mecanisme per obtenir autorització de l'usuari per operar amb els seus fitxers.

Ambdós proveïdors contemplen OAuth2 com a protocol per obtenir autorització de l'usuari per operar amb els seus fitxers. En concret implementen el flux *Authorization Code Grant* d'aquest protocol. La implementació de l'obtenció de l'autorització es veurà amb més detall en cada implementació.

Per finalitzar, les implementacions de cada proveïdor usen els *tokens* obtinguts del procés d'autorització per interactuar amb les API de gestió de fitxers. A continuació s'exposen les particularitats d'implementació de cada un dels dos proveïdors.

4.1.1 Google Drive

Google proporciona un SDK (*Software Development Kit*) per facilitar la interacció amb l'API de Google Drive. A més també hem d'usar el client d'OAuth2 [29] que proporciona Google per obtenir les credencials de l'usuari que ens serviran per interactuar amb l'API de Google Drive. D'aquesta manera, la interacció amb Google Drive queda de la següent manera:

Il·lustració 7: Components interacció amb Google Drive

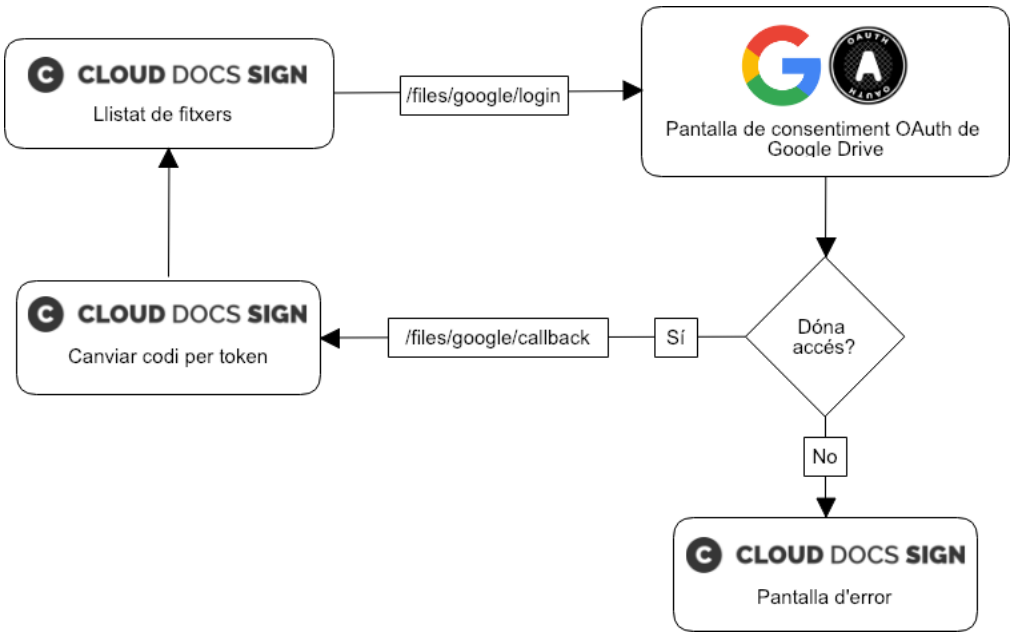
Mitjançant el client d'OAuth de Google obtenim l'autorització de l'usuari en forma de *token* d'OAuth. Aquestes credencials obtingudes són usades per l'SDK de Google Drive, que operarà en nom de l'usuari que ens ha autoritzat.

Per obtenir les credencials d'OAuth s'han habilitat dues URL necessàries per cobrir tot el procés. En primer lloc la URL que redirigeix l'usuari a la pantalla de consentiment d'OAuth de Google.

Un cop l'usuari autoritza l'aplicació a tractar amb les seves dades de Google Drive, aquest és redirigit a una URL intermèdia que serveix per intercanviar el codi rebut per Google per un *token* d'OAuth vàlid. Aquest pas és transparent per a l'usuari. Al final, quan des de CloudDocsSign es té el *token* d'OAuth vàlid es redirigeix l'usuari al llistat de fitxers on es poden veure els documents que aquest té allotjats a Google Drive.

Il·lustració 8: Pantalla de consentiment d'OAuth de Google Drive

Llavors, el flux que ha de seguir l'usuari per llistar els seus fitxers de Google Drive és el següent:



Il·lustració 9: Flux llistat de documents amb Google Drive

Un cop s'ha obtingut la credencial, aquesta es guardada en memòria, i és el propi client d'OAuth de Google l'encarregat de gestionar-la. Quan aquesta credencial caduca, de manera automàtica i totalment transparent per a l'usuari, el client d'OAuth de Google usa el *token* de refresc per obtenir un nou *token* que ens permeti seguir usant l'API de Google Drive sense necessitat de tornar a dirigir l'usuari al flux que acabem de veure.

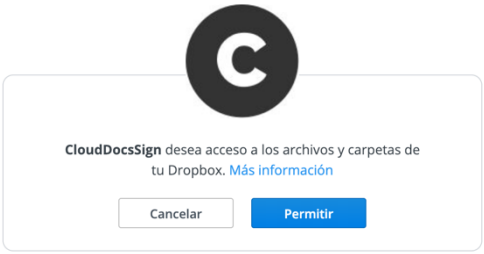
4.1.2 Dropbox

A banda de Google Drive, l'aplicació demostrador també funciona amb Dropbox com a magatzem de fitxers. Com hem comentat a apartats anteriors, el fet què l'usuari seleccioni Dropbox o un altre proveïdor de fitxers no afecta la resta de mòduls de l'aplicació, ja que aquesta treballa amb el concepte d'emmagatzemament i no directament amb Dropbox, Google Drive o altres proveïdors de documents *cloud*.

Pel que fa a com s'integra Dropbox amb CloudDocsSign hi podem remarcar una diferència amb Google Drive. La gestió de les credencials per usar l'API de Dropbox en nom de l'usuari la fa la pròpia llibreria de Dropbox. Així com amb Google Drive aquesta responsabilitat estava delegada al client d'OAuth de Google per Java, en aquest cas és el propi SDK de Dropbox que s'encarrega de la gestió dels *tokens*.

Dropbox també usa el protocol OAuth2 per a l'obtenció de les credencials d'ús. Com Google Drive també podem fer uns del flux *Authorization Code Grant*.

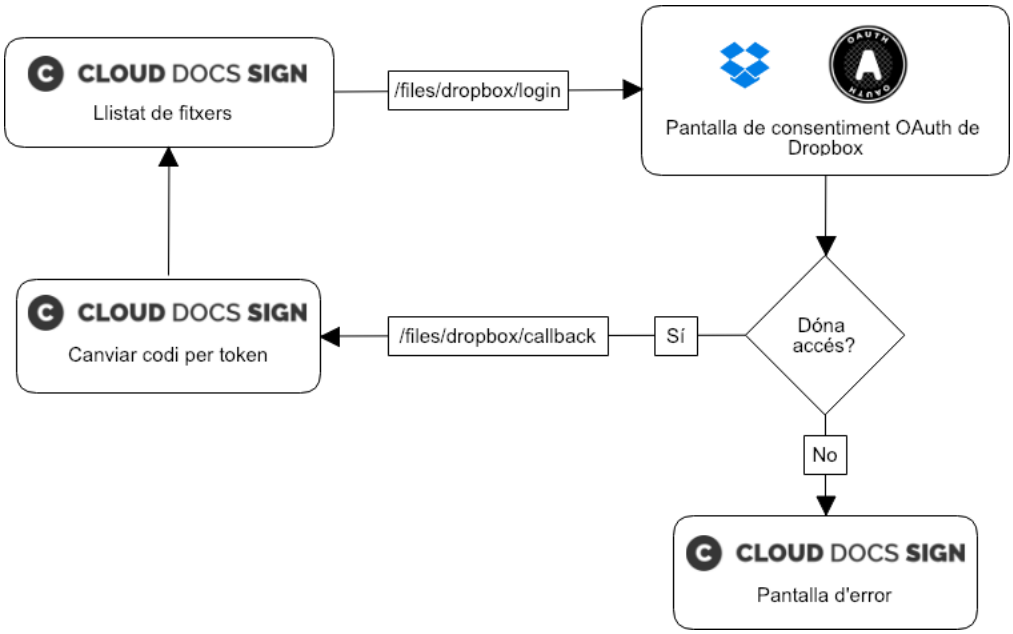
Quan sol·licitem permís a l'usuari per usar el seu compte de Dropbox és redirigit a la plana de consentiment d'OAuth de Dropbox.



Il·lustració 10: Pantalla de consentiment d'OAuth amb Dropbox

Un cop l'usuari ens ha donat accés se'l redirigeix a la mateixa plana intermèdia que intercanvia el codi rebut per Dropbox per un *token* d'OAuth vàlid. Un altre cop, com succeeix amb Google Drive aquest pas és transparent per a l'usuari que no és conscient d'aquesta redirecció. Si l'operació resulta satisfactòria, l'usuari és redirigit al llistat de fitxers on pot veure els documents que té allotjats a Dropbox.

El flux d'obtenció de credencials de Dropbox es pot observar a la següent figura:

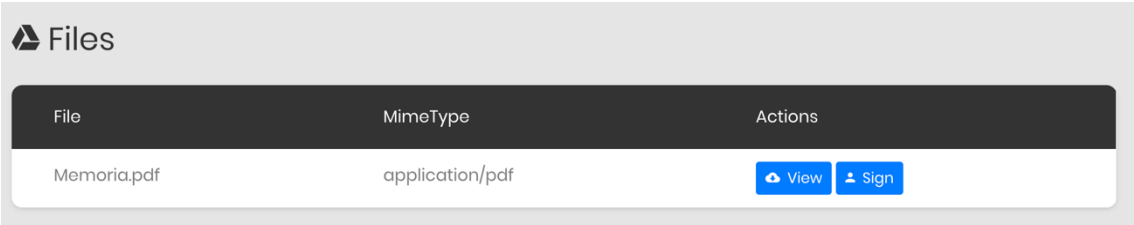


Il·lustració 11: Flux de llistat de fitxers amb Dropbox

4.1.3 Fitxers a CloudDocsSign

Una vegada tenim la integració amb les diferents plataformes de documents *cloud* que desitgem. El mòdul de gestió de fitxers de CloudDocsSign interactua amb les classes d'emmagatzemament. Concretament usa els tres mètodes que implementa: llistar fitxers, descarregar un fitxer i pujar-ne un de nou.

Quan l'usuari ha seleccionat el proveïdor de fitxers que desitja usar, és redirigit al llistat de fitxers. En aquest apartat l'usuari pot escollir o bé signar un fitxer amb una clau que escollirà posteriorment, o bé visualitzar-lo.



Il·lustració 12: Llistat de documents a CloudDocsSign

4.2 Gestió de claus i signatura de *hashes*

Tota la gestió de claus de signatura i firmat de documents està delegada a la plataforma TrustedX de Safelayer. Això té vàries implicacions. Per una banda TrustedX ens ofereix serveis de custòdia de claus. L'usuari allotja les seves claus de firma al servei de TrustedX i delega la responsabilitat de la guarda a aquest servei. D'aquesta manera la nostra aplicació demostrador queda lliure d'aquesta responsabilitat i l'únic que ha de fer és comunicar-se amb TrustedX per pujar o eliminar claus.

Per altra banda també s'ha delegat a TrustedX la signatura de documents PDF. Com veurem més endavant, el document no es signa a CloudDocsSign, sinó que s'envien les dades a signar a TrustedX juntament amb l'identificador de la clau que ha d'usar per realitzar aquesta signatura.

En aquest cas també es tracta d'integrar una API REST que incorpora OAuth com a mecanisme d'autenticació i autorització. Amb la diferència que no tenim cap SDK de TrustedX i hem de fer nosaltres les peticions HTTP per interactuar amb el sistema. Pel que fa OAuth també usa el flux *Authorization Code Grant* però amb la particularitat que per utilitzar una identitat de signatura s'ha de demanar un *token* específic que només serveix per usar la identitat especificada.

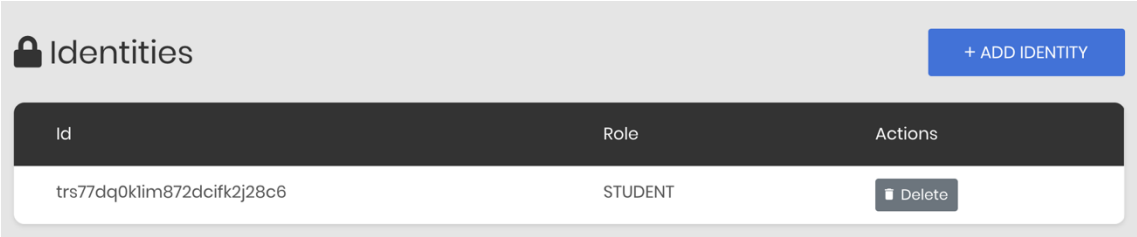
A continuació es veu amb més detall cada un d'aquests tres punts: gestió de claus, signatura d'un *hash* i interacció amb l'API de TrustedX.

4.2.1 Gestió de claus

Com hem comentat anteriorment, tota la gestió de claus la fa TrustedX. L'usuari només és responsable de pujar la seva clau de firma mitjançant l'aplicació demostrador a TrustedX. Des d'aquesta mateixa aplicació l'usuari pot visualitzar quines claus té allotjades al servei TrustedX i eliminar-les si escau.

Cal comentar que és necessari que l'usuari estigui registrat prèviament a la plataforma TrustedX de Safelayer. Aquest registre només es pot realitzar de manera presencial amb un *LoA (Level of Assurance)* alt.

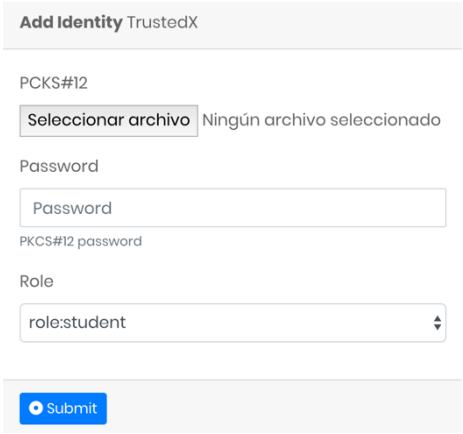
Al costat de l'aplicació demostrador tenim un apartat d'identitats. Des d'aquest apartat l'usuari pot gestionar les seves claus i afegir-ne una de nova. En el primer accés a aquest apartat, l'aplicació demana permís a l'usuari per gestionar les seves claus de TrustedX (mitjançant el protocol OAuth). Una vegada tenim el permís de l'usuari es mostra una llista amb les claus de l'usuari i les accions que pot realitzar.



Il·lustració 13: Llistat d'identitats a CloudDocsSign

Com es pot observar, cada clau té un identificador i un rol associat. L'acció que pot realitzar l'usuari sobre cada clau és eliminar-la.

Per altra banda l'usuari també pot afegir claus noves a TrustedX. En afegir una nova clau se li demana permís per a registrar noves claus. Com hem vist a l'apartat d'anàlisi de la plataforma TrustedX, registrar una clau o identitat és un *scope* d'OAuth. En aquest cas, si l'usuari no ens havia autoritzat prèviament a afegir noves identitats, en primer lloc li demanarem permís per realitzar aquesta acció i posteriorment es mostra un formulari per afegir una nova clau.



Il·lustració 14: Formulari d'inserció d'una clau a TrustedX amb CloudDocsSign

L'usuari ha d'especificar el fitxer en format PKCS#12, la contrasenya d'aquest arxiu de claus i el rol que vol associar a la clau. Un cop l'usuari envia el formulari s'envia la clau a TrustedX i l'usuari és redirigit al llistat de claus que hem vist anteriorment on pot veure la clau recentment afegida.

La darrera operació que manca per explicar és l'eliminació de claus. Quan l'usuari desitja eliminar una identitat del seu perfil i l'operació resulta satisfactòria, és redirigit un altre cop al llistat de claus on pot observar que la seva clau ja no es troba disponible.

4.2.2 Signatura de *hashes*

L'altra part imprescindible de la integració amb TrustedX és que aquesta plataforma ens brinda la possibilitat de signar el *digest* del document PDF al *cloud*. D'aquesta manera, quan l'usuari selecciona el document que vol signar i quina clau desitja usar, s'envia el *hash* del document PDF a TrustedX juntament amb l'identificador de la clau que l'usuari ha elegit prèviament i l'algorisme de xifratge a utilitzar a la signatura.

Igual que amb les peticions a TrustedX per llistar signatures, afegir-ne una o eliminar-ne, la petició de signatura d'un *hash* també ha d'anar autenticada amb un *token* d'OAuth. Però aquest *token* que s'ha de proporcionar ha d'estar associat a una identitat en particular. És a dir, quan signem un document se li ha de demanar permís a l'usuari a fi que la plataforma CloudDocsSign pugui usar la clau que aquest ha seleccionat. El *token* que s'obtindrà només serà vàlid per a la clau especificada i no ho serà per cap altra.

Un cop s'ha fet la petició, si aquesta resulta satisfactòria, TrustedX ens retorna els bytes de la signatura en format PKCS#1 [30] que hem d'incrustar al document PDF. Un cop tenim els bytes de la signatura d'aquest procés se n'encarrega la llibreria iText que explicarem més endevant.

4.2.3 Interacció amb l'API de TrustedX

Com s'ha comentat anteriorment, per aquesta API REST no tenim un SDK o client que ens ajudi a interactuar amb la API. S'ha desenvolupat un client d'API que ens proporciona una interfície que contempla les operacions que ens brinda l'API de TrustedX. Aquesta interfície ens ofereix els següents mètodes:

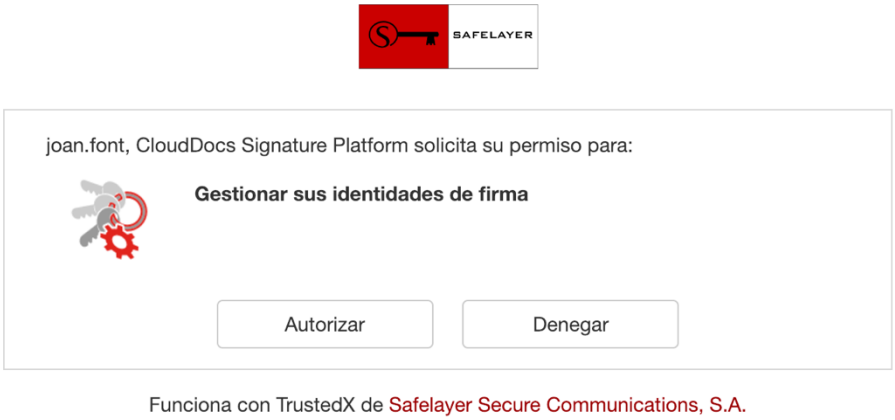
- Llistat d'identitats
- Detall d'una identitat
- Borrat d'una identitat
- Signatura d'un *hash* amb una identitat

Aquestes quatre operacions es veuen envoltades d'altres mètodes que gestionen els *tokens* d'OAuth necessaris per realitzar-les. És a dir, el client d'API de TrustedX es veu envoltat d'un altre client que a més de realitzar la petició, decora els mètodes amb el *token* d'OAuth necessari.

Com s'ha comentat a l'apartat d'anàlisi de la plataforma TrustedX, cada petició requereix un permís que l'usuari ha d'haver concedit prèviament. Això es coneix com a *scope*. Per exemple, per realitzar la petició de llistat d'identitats és necessari l'*scope urn:safelayer:eidas:sign:identity:manage,* en canvi per registrar una nova identitat és necessari l'*scope urn:safelayer:eidas:sign:identity:register.* Aquests *tokens* tenen una caducitat que ve especificada en la resposta de petició d'un *token*. És a dir, podem usar el *token* tantes vegades com desitgem mentre aquest no estigui expirat.

Quan l'usuari registra una identitat, en primer lloc s'ha de dirigir al llistat d'identitats. En aquest cas, si és la primera vegada que hi accedeix se li demana permís per gestionar les seves identitats (amb l'*scope urn:safelayer:eidas:sign:identity:manage*). Un cop és al llistat, es pot dirigir a l'apartat d'afegir una nova identitat. Com que per realitzar aquesta acció és necessari un altre permís, se li torna a demanar permís per registrar una nova identitat (*scope urn:safelayer:eidas:sign:identity:register*). Un cop ha registrat la nova identitat, segons el flux de l'aplicació, l'usuari és redirigit al llistat d'identitats. En aquest cas ja no se li demanarà permís per llistar les identitats ja que ens l'ha concedit prèviament i aquest permís encara no ha caducat.

Això s'aconsegueix usant el que hem anomenat registre de *tokens*. Per a cada usuari es guarda en sessió (gestionat pel *framework* web) un *token* vàlid per a cada *scope*. Aquest registre de *tokens* s'usa juntament amb el client de TrustedX que hem desenvolupat. D'aquesta manera, quan es realitza una petició que requereix l'*scope* X, si no existeix *token* per aquest *scope*, es redirigeix l'usuari a la pantalla de consentiment d'OAuth de TrustedX on l'usuari d'autoritzar l'aplicació a realitzar l'acció associada a l'*scope* X.



Il·lustració 15: Pantalla de consentiment d'OAuth de TrustedX

Un cop es té el consentiment de l'usuari, aquest *token* associat a l'*scope* es guarda al registre de *tokens* de l'usuari. D'aquesta manera, quan es torni a requerir un *token* amb aquest *scope* es podrà reutilitzar el que s'ha obtingut prèviament mentre no hagi expirat.

Un cas especial d'aquest registre de *tokens* és quan demanem un token associat a l'*scope urn:safelayer:eidas:sign:identity:use:server.* Aquests *tokens* van associats a més de a l'*scope* a una identitat en concret. En el nostre registre de tokens, en lloc d'associar-los a un *scope* els associem a un identificador d'identitat. D'aquesta manera, quan l'usuari vol signar un document amb una identitat en concret se li demana permís per usar aquesta identitat i s'associa dins el registre de *tokens* el *token* obtingut amb la identitat associada.

4.2.4 Signatura qualificada i TrustedX

La signatura que realitza TrustedX sobre el *hash* que enviem a signar no és una signatura qualificada real. Tot i que TrustedX és un producte capaç de donar signatura qualificada, en la nostra aplicació demostrador hi ha alguns aspectes que fan que no ho sigui a efectes legals. El certificat que es proporciona no és un certificat reconegut. En el moment d'usar la clau, quan demanem autorització a l'usuari per usar la firma, la plataforma hauria de sol·licitar les credencials a l'usuari per tornar activar la clau. A més, les claus s'haurien d'haver generat al servidor qualificat i mai s'hauria d'haver importat la clau en format PKCS#12.

Aquests punts que es comenten són importants per tenir una signatura qualificada a efectes legals, però en cap cas depèn de l'aplicació CloudDocsSign. La responsabilitat que la signatura sigui qualificada a efectes legals recau sobre TrustedX. El procés de signatura no es veuria afectat en una hipotètica adaptació a la legalitat.

L'únic canvi que hauríem de realitzar a la nostra aplicació seria que desapareixeria el formulari de pujada d'una clau i seria substituït per una acció que indicaria al servidor de TrustedX que generi les claus i el certificat qualificat. Només amb aquest petit canvi ja tindríem una signatura qualificada real i vàlida a efectes legals.

4.3 Autenticació

L'autenticació també és part fonamental de l'aplicació demostrador. En el cas de CloudDocsSign s'ha optat per delegar aquesta responsabilitat a Google. Per accedir a qualsevol mòdul de l'aplicació és necessari haver-se autenticat contra Google.

L'autenticació es realitza mitjançant el protocol OAuth2. En concret es demana a l'usuari que ens concedeixi permís per consultar el seu nom d'usuari i la direcció de correu electrònic a Google. La integració de l'autenticació és feta completament pel mòdul *spring-security-oauth2-jose* [31] del *framework* web

que hem usat. D'aquesta manera només és necessari especificar el *client id* i *client secret* de l'aplicació donada d'alta a Google.

Quan l'usuari entra a CloudDocsSign el que se li demana és que s'autentiqui amb Google per seguir usant l'aplicació. Quan s'autentica amb Google el *framework* s'encarrega de realitzar tot el fluxe d'autorització d'OAuth2 (parescut al que hem vist anteriorment amb Google Drive). Un cop s'ha autenticat i ens ha autoritzat a consultar la informació sol·licitada, es crea una sessió per a l'usuari amb aquesta informació. Per la part de configuració, només hem hagut d'especificar quin proveïdor d'OAuth2 es desitja usar i la seva configuració:

```
security:
  oauth2:
    client:
      registration:
        google:
          client-id: ${GOOGLE_LOGIN_CLIENT_ID}
          client-secret: ${GOOGLE_LOGIN_CLIENT_SECRET}
```

Com podem observar, només hem configurat les credencials de l'aplicació d'OAuth creada a la consola de Google. Tota la resta de configuració està implícitament configurada dins el paquet *spring-security-oauth2-jose*. Es segueix l'estàndard OpenID [32]. Actualment a CloudDocsSign només tenim configurat Google com a principal proveïdor d'identitat. Afegir altres proveïdors com Facebook, GitHub, Twitter entre d'altres només requreix el registre d'una aplicació d'OAuth a la seva plataforma de desenvolupadors, i la configuració de nous proveïdors d'identitat per part de *Spring Framework*.

És important comentar la separació de l'IdP (proveïdor d'identitat) de CloudDocsSign i dels documents de l'IdP del proveïdor de claus. Es tracta de dos proveïdors d'identitat diferents. En el cas de CloudDocsSign i documents usem Google i Dropbox (Google és usat com a sistema d'autenticació per a CloudDocsSign i Google i Dropbox com a IdP pels documents). L'IdP del proveïdor de claus qualificades és TrustedX. Fet aquest aclariment cal remarcar que mai es protegiran les claus qualificades amb un compte de Google o Dropbox. Com hem comentat anteriorment, aquesta protecció es fa amb TrustedX que compta amb un procés de registre amb un *LoA (Level of Assurance)* alt.

4.4 iText PDF

iText PDF [33] és el darrer pilar de la plataforma CloudDocsSign. La llibreria de manipulació de documents PDF és l'encarregada d'aplicar els bytes de la signatura obtinguts de TrustedX al PDF que es troba en algun proveïdor de documents *cloud*.

La signatura de documents PDF amb iText pot funcionar de dues formes. Proporcionant el certificat del firmant i que iText s'encarregui de realitzar la signatura o delegant aquesta acció a un tercer que se n'encarregui. Aquest darrer és el cas que aplica a CloudDocsSign, on la responsabilitat de signatura del document PDF és delegada a TrustedX.

Aquest mètode de signatura s'anomena "signatura separada" [34] [35]. iText PDF proporciona una interfície de signatura separada que obliga a implementar al programador un mètode de signatura que rep per paràmetre el missatge a signar i dos altres mètodes per especificar l'algorisme de *hash* de les dades i el de xifratge.

```
public interface ExternalSignature {

    public String getHashAlgorithm();

    public String getEncryptionAlgorithm();

    public byte[] sign(byte[] message) throws GeneralSecurityException;
}
```

Per a la signatura amb TrustedX l'algorisme de *hashing* és SHA-256, l'algorisme d'encriptació és RSA i el mètode *sign* és l'encarregat de cridar a l'API de TrustedX que s'encarrega de rebre el missatge a signar i retornar els bytes de la signatura que s'ha d'aplicar al document PDF.

Els passos a seguir dins el mètode *sign* són:

- Aplicar la funció de resum especificada al mètode *getHashAlgorithm* al missatge rebut per paràmetre.
- Cridar l'API de TrustedX amb el missatge a signar, la identitat a usar i l'algorisme de signatura a usar (*rsa-sha256*).
- Agafar del cos de la resposta els bytes, que representen la signatura en format PKCS#1.

Un cop s'ha realitzat la signatura, iText PDF s'encarrega d'aplicar el que ha rebut de la signatura separada al document PDF i s'obté com a resultat el document signat.

4.5 Spring Framework

Per finalitzar l'apartat de desenvolupament del projecte es comentarà com s'ha usat *Spring Framework* per contribuir a un desenvolupament més àgil.

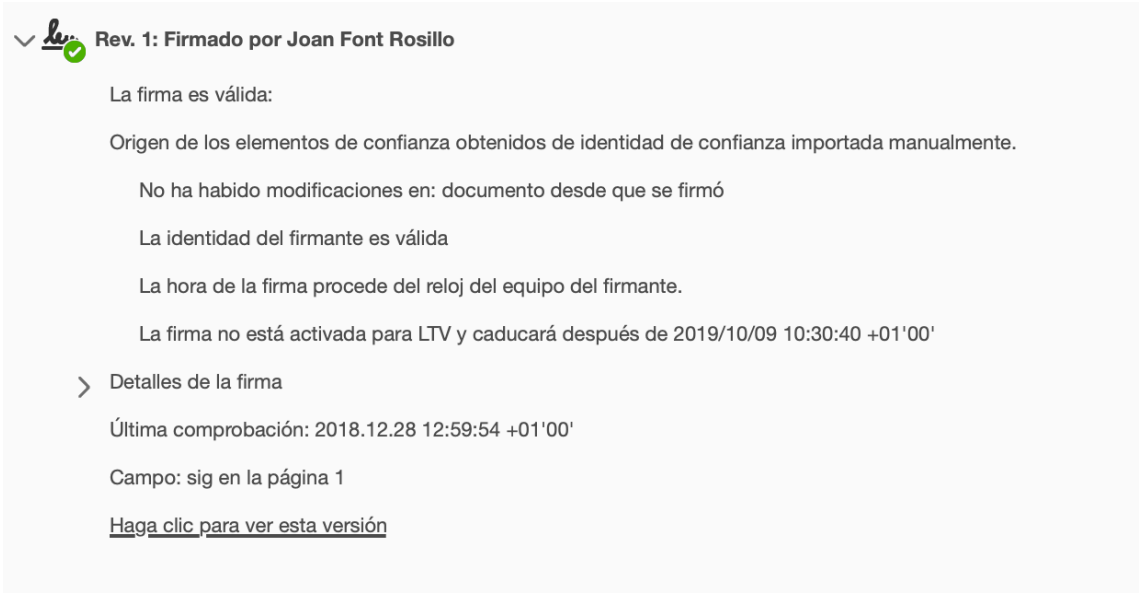
Un dels punts on ha contribuït *Spring Framework* és en el procés d'autenticació. Com ja hem comentat anteriorment, només amb configuració del projecte hem pogut delegar l'autenticació a un proveïdor d'identitat extern i centrar els esforços en el que requereix el projecte, la signatura electrònica de documents PDF al *cloud*.

Tot el que no requereix d'interacció amb l'usuari, per exemple el procés de signatura de documents, la comunicació amb els diferents proveïdors de documents o la interacció amb l'API de TrustedX s'ha desenvolupat de tal manera que la seva dependència amb el *framework Spring* és nul·la. D'aquesta manera fer el canvi de *framework* a un altre, no seria una complicació ja que la majoria de components de la plataforma estan deslligats del *framework*.

Al cap i a la fi, el *framework* només ens ha servit de capa de comunicació entre l'usuari final i la lògica de la nostra aplicació. L'hem fet servir per a la capa de presentació usant el motor de plantilles que incorpora, per proporcionar autenticació i com a mètode de suport per a què el desenvolupament sigui més àgil, ja que proporciona un marc de treball que facilita la recepció de peticions HTTP i l'enviament de les corresponents respostes.

4.6 Signatura verificable amb Adobe Acrobat

Un cop realitzat el procés de signatura, podem veure com aquesta és reconeguda pel software Adobe Acrobat [36]. Es tracta d'un dels lectors de documents més usats. Com hem comentat en l'apartat de la integració amb TrustedX, el certificat de signatura no és un certificat reconegut. En primer lloc haurem d'afegir a Adobe Reader el *Root CA* de TrustedX. Un cop el tenim afegit i obrim un document signat amb la plataforma CloudDocsSign podem inspeccionar la firma.



Il·lustració 16: Sumari de la signatura electrònica a un document PDF amb Adobe Acrobat

Com podem observar, es tracta d'una signatura vàlida i reconeguda per Adobe. També podem veure que els elements de confiança (els certificats) s'han importat manualment. Com hem comentat anteriorment, hem indicat a Adobe que els certificats de TrustedX que hem importat són de confiança.

5. Conclusions i treball futur

5.1 Conclusions

Un cop desenvolupada l'aplicació demostrador és el moment de analitzar si s'han assolit els requisits que havíem marcat a l'inici del projecte. Recordem breument quins eren aquests objectius. Ens havíem plantejat tenir una aplicació que fos capaç de signar documents provinents del *cloud* usant la plataforma TrustedX. A més, l'usuari havia de ser capaç de gestionar les seves identitats amb TrustedX. Per finalitzar, l'usuari s'ha d'autenticar amb Google per poder entrar a l'aplicació i fer-ne ús. Com a requisits de sistema ens havíem plantejat que l'aplicació fos capaç de gestionar documents de la mateixa manera fos quin fos el seu proveïdor. A més l'aplicació havia de complir amb la normativa europea eIDAS.

L'aplicació demostrador que hem obtingut del procés de desenvolupament és capaç de gestionar documents d'una mateixa manera independentment de la seva provinença. Com hem explicat en l'apartat de desenvolupament s'ha creat una interfície que permet tractar qualsevol document de la mateixa forma sigui quin sigui el seu proveïdor.

A més s'ha integrat per complet el procés de signatura d'un *hash* amb TrustedX. També, amb TrustedX, s'ha integrat la gestió d'identitats. Per autenticar-se a l'aplicació és necessari identificar-se davant Google; en cas contrari no es pot usar.

Podem dir que s'han complert tots els objectius que ens havíem marcat a l'inici del projecte. Prova d'això és que l'aplicació demostrador és del tot funcional: realitza la signatura sobre un document i el puja al proveïdor d'origen del document.

Per altra banda, i com hem comentat anteriorment, hem de remarcar que la nostra signatura no és una signatura electrònica qualificada real. Tot i que TrustedX és un producte capaç de donar una signatura qualificada, en la nostra aplicació demostrador hi ha alguns aspectes que fan que no sigui així a efectes legals. Fer que la signatura que es realitza sigui qualificada es pot plantejar com un punt del treball futur que veurem a continuació.

5.2 Treball futur

Com hem anat comentant al llarg d'aquesta memòria, el TFM es tracta del desenvolupament d'una aplicació demostrador, que integri almenys dos proveïdors de documents *cloud* i que sigui capaç de realitzar la signatura amb TrustedX. L'aplicació és de tot funcional i compleix amb els requisits que ens havíem plantejat a l'inici del projecte. Però no té per què acabar aquí: s'hi poden afegir més característiques o millorar les ja existents.

En primer lloc i com hem comentat en el punt anterior de conclusions, la signatura que realitzem amb TrustedX no és una signatura qualificada real per múltiples factors que hem vist a l'etapa de desenvolupament. Un punt a treballar en el futur seria fer els pertinents ajusts per a què aquesta signatura sigui qualificada. La major part d'aquest desenvolupament recau és responsabilitat de TrustedX però hauríem de fer alguns canvis a la nostra interfície per complir amb els nous requisits.

Per altra banda, un pilar fonamental de CloudDocsSign és que l'aplicació funcioni amb diferents proveïdors de documents *cloud.* La nostra aplicació demostrador funciona amb Google Drive i Dropbox, però es podria integrar també amb altres proveïdors com Amazon S3 [37], Box [38], OwnCloud [39], Mega [40] o OneDrive de Microsoft [41]. Només caldria implementar les corresponents classes que es comuniquin amb aquests serveis. L'aplicació disposa de la capa d'abstracció necessària per tractar un fitxer de la mateixa manera sigui quina sigui la seva provinença.

Amb aquests dos nous desenvolupaments tindríem una millor aplicació, capaç de realitzar una signatura qualificada real i integrada amb molts més proveïdors. A banda d'això també es podria millorar la interfície de la plataforma per a què sigui més usable. Es podria implementar la signatura massiva de documents, permetent l'usuari seleccionar múltiples documents que desitgi signar.

Bibliografia

[1] DNIe. Dirección General de la Policía. Consultat el 30 de desembre de 2018. https://www.dnielectronico.es/PortalDNIe/PRF1_Cons02.action?pag=REF_106&id_menu=1

[2] DNI 3.0. Dirección General de la Policía. Consultat el 30 de desembre de 2018. https://www.dnielectronico.es/PortalDNIe/PRF1_Cons02.action?pag=REF_103&id_menu=1

[3] NFC. Widipedia. Consultat el 30 de desembre de 2018. https://en.wikipedia.org/wiki/Near-field_communication

[4] eIDAS. Wikipedia. Consultat el 30 de desembre de 2018. https://en.wikipedia.org/wiki/EIDAS

[5] PKCS#12. Wikipedia. Consultat el 30 de desembre de 2018. https://en.wikipedia.org/wiki/PKCS_12

[6] OAuth. Wikipedia. Consultat el 30 de desembre de 2018. https://en.wikipedia.org/wiki/OAuth

[7] PDFCreator. PDFForge. Consultat el 30 de desembre de 2018. https://www.pdfforge.org/pdfcreator

[8] DigiSigner. Consultat el 30 de desembre de 2018. https://www.digisigner.com/

[9] Sinadura. Consultat el 30 de desembre de 2018. http://www.sinadura.net/es/

[10] JSignPdf. SourceForge. Consultat el 30 de desembre de 2018. http://jsignpdf.sourceforge.net/

[11] iSafePDF. CodePlex. Consultat el 30 de desembre de 2018. https://archive.codeplex.com/?p=isafepdf

[12] XolidoSign. Xolido. Consultat el 30 de desembre de 2018. https://www.xolido.com/lang/xolidosign/

[13] eCoFirma. Ministerio de Industria, Comercio y Turismo. Consultat el 30 de desembre de 2018. https://sedeaplicaciones.minetur.gob.es/ecofirma/

[14] Signaturit. Consultat el 30 de desembre de 2018. https://www.signaturit.com/es

[15] DocuSign. Consultat el 30 de desembre de 2018. https://www.docusign.com/

[16] Level of Assurance. ISO/IEC 29115. Consultat el 30 de desembre de 2018. https://www.iso.org/standard/45138.html

[17] Authorization Code Grant. RFC 6749. Consultat el 30 de desembre de 2018. https://tools.ietf.org/html/rfc6749#section-4.1

[18] Directiva 1999/93/CE. Boletín Oficial del Estado. Ministerio de la Presidencia, Relaciones con las Cortes e Igualdad. Consultat el 30 de desembre de 2018. https://www.boe.es/buscar/doc.php?id=DOUE-L-2000-80059

[19] Artícle 25 eIDAS. Reglamento 910/2014. Unió Europea. Consultat el 30 de desembre de 2018. https://eur-lex.europa.eu/legal-content/ES/TXT/?uri=celex%3A32014R0910

[20] Artícle 26 eIDAS. Reglamento 910/2014. Unió Europea. Consultat el 30 de desembre de 2018. https://eur-lex.europa.eu/legal-content/ES/TXT/?uri=celex%3A32014R0910

[21] PAdES. Wikipedia. Consultat el 30 de desembre de 2018. https://en.wikipedia.org/wiki/PAdES

[22] ISO 32000-1. IEO/IEC. Consultat el 30 de desembre de 2018. https://www.iso.org/standard/51502.html

[23] Google Drive API. Google Developers. Consultat el 30 de desembre de 2018. https://developers.google.com/drive/

[24] Dropbox API. Dropbox Developers. Consultat el 30 de desembre de 2018. https://developers.google.com/drive/

[25] Grails Frameowrk. Grails. Consultat el 30 de desembre de 2018. https://grails.org/

[26] Spring. Consultat el 30 de desembre de 2018. https://spring.io/

[27] Play! Famework. Consultat el 30 de desembre de 2018. https://www.playframework.com/

[28] Vert.x. Consultat el 30 de desembre de 2018. https://vertx.io/

[29] OAuth Client for Java. Google Developers. Consultat el 30 de desembre de 2018. https://developers.google.com/api-client-library/java/google-oauth-java-client/

[30] PKCS#1. Wikipedia. Consultat el 30 de desembre de 2018. https://en.wikipedia.org/wiki/PKCS_1

[31] Spring Security OAuth JOSE. Spring. Consultat el 30 de desembre de 2018. https://spring.io/blog/2018/01/30/next-generation-oauth-2-0-support-with-spring-security

[32] OpenID. Wikipedia. Consultat el 30 de desembre de 2018. https://es.wikipedia.org/wiki/OpenID_Connect

[33] iText. Consultat el 30 de desembre de 2018. https://itextpdf.com/

[34] iText External Signature Sample. GitHub. Consultat el 30 de desembre de 2018. https://github.com/itext/i5js-tutorial/blob/master/signatures/src/main/java/signatures/chapter4/C4_07_ClientServerSigning.java

[35] iText External Signature Interface. iText. Consultat el 30 de desembre de 2018. http://itextsupport.com/apidocs/itext5/5.5.9/com/itextpdf/text/pdf/security/ExternalSignature.html

[36] Acrobat DC. Adobe. Consultat el 30 de desembre de 2018. https://acrobat.adobe.com/es/es/acrobat.html

[37] Amazon S3. Amazon Web Services. Consultat el 30 de desembre de 2018. https://aws.amazon.com/es/s3/

[38] Box. Consultat el 30 de desembre de 2018. https://www.box.com/es-es/home

[39] ownCloud. Consultat el 30 de desembre de 2018. https://owncloud.org/

[40] MEGA. Consultat el 30 de desembre de 2018. https://mega.nz/

[41] OneDrive. Microsoft. Consultat el 30 de desembre de 2018. https://onedrive.live.com/about/es-es/