Informe de prácticas de ISDCM

Entrega 3

Alumno/a: Albert Bausili Fernández

Alumno/a: Noa Yu Ventura Vila

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Facultat d'Informàtica de Barcelona

FIB

Contenido

1. Introducción

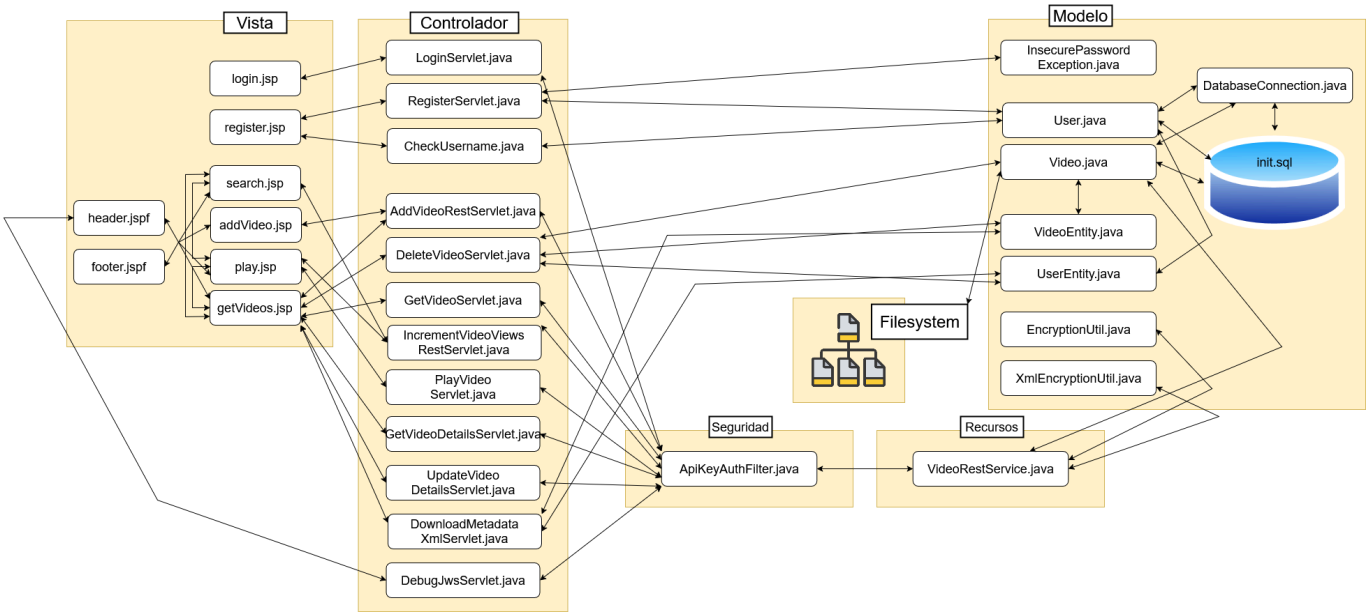
Este informe contiene toda la documentación del código hecho para esta tercera entrega. Primero de todo veremos un esquema para tener una idea de la arquitectura de la aplicación y ver cómo se comunican las distintas partes. Como no hemos cambiado la mayoría de los métodos en sí, sino la seguridad de éstos, la tabla con todos los endpoints disponibles y los métodos usados para cada uno no se incluye en este informe, ya que está disponible en el de la entrega 2. Luego habrá una breve y resumida explicación de las funcionalidades de la aplicación con especial énfasis en las mejoras hechas para esta entrega en específico. También habrá un apartado de aclaraciones para justificar ciertas decisiones en la arquitectura de la aplicación y después un apartado para recalcar todos los extras que hemos hecho. Finalmente podremos ver los repositorios de código y bibliografía consultados.

El propósito principal de esta entrega es modificar el código de la entrega 2 para que sea más seguro, y para hacerlo hemos añadido tres *features*: usar HTTPS añadiendo un certificado, encriptar y desencriptar ciertos contenidos, y usar JWT para manejar la sesión del usuario cuando éste hace login. Además, ahora se pueden descargar los metadatos del vídeo que escojas.

2. Decisiones de diseño

2.1. Arquitectura de la Aplicación

Las vistas para esta entrega son las mismas que las de la entrega anterior (solo hemos añadido un par de botones). En cuanto a la arquitectura de la aplicación, no ha cambiado demasiado ya que las funcionalidades son las mismas. Lo que sí hemos añadido es un endpoint en el backend REST para hacer el login con REST usando JWT y otro endpoint que permite descargar el JWT sin la encriptación del JWE para poder verificarlo en jwt.io. También hemos añadido un servlet para poder descargar los metadatos del vídeo que quieras de manera fácil con un solo click en el botón y otro servlet que hace de "puente" para descargar el JWT desencriptado.



Respecto a la figura 5 del enunciado que se muestra en la imagen de abajo, nosotros lo hemos hecho de la misma manera que se indica en ésta. El XmlEncryptionUtil.java es el que cifra y descifra los metadatos de los vídeos que vienen en formato XML, y el EncryptionUtil.java es el que encripta el contenido de los vídeos en sí.

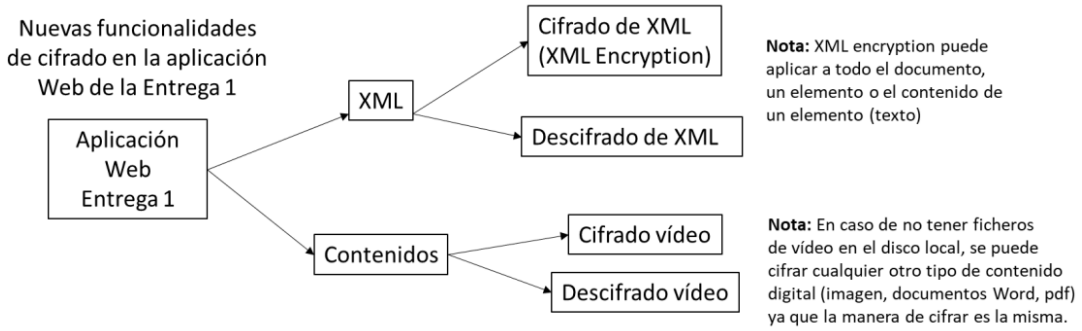


Figura 5. Integración aspectos de seguridad en la Entrega 1

2.2. Explicación de Ficheros

Los ficheros que no aparezcan en esta lista pero que estén presentes en el código significa que ya no se usan, ya que son funcionalidades deprecadas hechas sin REST de la entrega 1. Los ficheros de configuración como los que hay dentro de WEB-INF tampoco se especifican.

Página	**Breve descripción**
Archivos JSP	
login.jsp	Interfaz para iniciar sesión. Las credenciales ingresadas por el usuario se envían al LoginServlet. Se puede acceder sin sesión.
register.jsp	Proporciona el formulario de registro. Incluye estilos y scripts para validaciones en tiempo real, como la llamada a CheckUsername para verificar la disponibilidad del nombre de usuario (mostrando mensajes de error si ese nombre de usuario ya existe). Se puede acceder sin sesión.
addVideo.jsp	Contiene la tag style y bootstrap para el diseño de la página. Vista para colgar un nuevo vídeo después de darle al botón "Upload File" en la vista getVideos.jsp. Contiene un script que envía el vídeo y la información introducida por el usuario (Título y descripción) al servlet y automáticamente obtiene la duración del vídeo. El envío del formulario puede dirigirse a AddVideoServlet (subida directa con cifrado y generación XML local) o a AddVideoRestServlet (que llama al API REST). Requiere sesión.
getVideos.jsp	Contiene la tag style y bootstrap para el diseño de la página. Comprueba que haya un usuario que se haya logueado para mostrar el icono de borrar video solo en los que el usuario haya subido. Cada video tiene un botón para reproducir mediante streaming de un servlet y al reproducirlo llama a

	otro servlet que aumenta el número de vistas en 1. Requiere sesión.
search.jsp	Muy parecido al getVideos.jsp con la peculiaridad que permite buscar imágenes según ciertos filtros establecidos por el usuario. Requiere sesión.
play.jsp	Reproducir los vídeos al darle al botón de play. Es el que incrementa el número de visitas del vídeo.
Servlets	
LoginServlet.java	Tiene las mismas comprobaciones de errores que en el JSP y llama al endpoint login en el servicio REST para comprobar que el nombre de usuario y la contraseña son correctos. Si son correctos inicia la sesión del usuario. Usa JWT.
RegisterServlet.java	Tiene las mismas comprobaciones de errores que en el JSP para ver si la contraseña es segura (en caso negativo lanza un InsecurePasswordException). Si los datos son correctos los envía a la capa modelo para guardarlos en la DB y redirige al usuario al login. No requiere sesión.
CheckUsername.java	Llama a User.java para comprobar que el nombre de usuario no existe aún (y si no impide proseguir con el registro). Se usa cada vez que el usuario deja de teclear durante un tiempo específico en el campo de Usuario en el formulario de registro. No requiere sesión.
AddVideoRest Servlet.java	Crea un nuevo vídeo con las características que recibe con la request y llama al servicio REST para que éste lo añada a la base de datos y lo guarde también en el filesystem para que después se pueda recuperar. Requiere sesión.
DeleteVideoServlet.java	Cuando el servlet recibe la petición lo envía al servicio REST, y éste comprueba que el vídeo exista, que quien quiere borrarlo sea el autor de ese vídeo y que la operación haya sido exitosa. Requiere sesión.
GetVideoServlet.java	Devuelve los vídeos que están subidos en la plataforma según unos filtros establecidos por el usuario (y si no lo están, pues devuelve los que haya). Para hacerlo se lo pide al servicio REST. Requiere sesión.
GetVideoDetails Servlet.java	Obtiene los detalles de un vídeo específico llamando al endpoint GET del API REST /rest/resources/videos/details/{videoId}. Envía el JWT de la cookie en la cabecera Authorization. Pasa los detalles del vídeo a play.jsp. Requiere sesión.
IncrementVideo ViewsRestServlet.java	Cuando el servlet recibe la petición envía un POST para incrementar el número de visitas del vídeo que dice el parámetro en 1. Requiere sesión.
PlayVideoServlet.java	Sirve el contenido de un vídeo para su reproducción. Llama al endpoint GET del API REST /rest/resources/videos/stream/{videoId} para obtener el flujo de vídeo (que el API REST se encarga de descifrar). Envía el JWT de la cookie en la cabecera Authorization. Requiere sesión.

UpdateVideoDetails Servlet.java	Actualiza el título y la descripción de un vídeo. Llama al endpoint PUT del API REST /rest/resources/videos/update/{videoId}. Envía el JWT de la cookie en la cabecera Authorization. Se espera que el API REST actualice la BD y el archivo XML de metadatos. Requiere sesión.
DownloadMetadataXml Servlet.java	Permite descargar los metadatos de un vídeo en formato XML. Si el solicitante es el autor del vídeo, se descarga el XML totalmente descifrado (obtenido via Video.getDecryptedMetadataXmlStream()). Si no es el autor, se descarga el archivo XML tal como está almacenado (es decir, con su contenido ya XML-cifrado). Requiere sesión.
LogoutServlet.java	Cierra la sesión del usuario invalidando la sesión HTTP y eliminando la cookie jwt_token. Redirige a login.jsp.
Clases de Modelo	
UserEntity.java	Clase usuario con sus atributos y métodos. Requiere sesión.
VideoEntity.java	Clase vídeo con sus atributos y métodos. Requiere sesión.
User.java	Gestiona todas las llamadas a la base de datos relacionadas con el usuario: registro, autenticación, hashear contraseña y comprobar si el nombre de usuario existe. Requiere sesión.
Video.java	Gestiona todas las llamadas a la base de datos relacionadas con los vídeos: añadir uno nuevo, obtener todos los vídeos, incrementar las visitas en 1 de un vídeo en concreto, obtener un vídeo por ID y borrar vídeos. Requiere sesión.
DatabaseConnection.java	Crea la conexión con la base de datos. Requiere sesión.
EncryptionUtil.java	Clase de utilidad para el cifrado y descifrado de flujos de datos (utilizada para los archivos de vídeo) usando AES/CBC/PKCS5Padding. Utiliza una clave secreta y un vector de inicialización fijos.
XmlEncryptionUtil.java	Clase de utilidad para el cifrado y descifrado de documentos XML (utilizada para los archivos de metadatos) siguiendo el estándar XML Encryption, usando AES de 128 bits. Cifra un elemento específico dentro del documento XML.
InsecurePassword Exception.java	La comprobación de que la contraseña que quiere crear el usuario en la vista de registro es segura solamente está implementada en el servlet de registrar usuario (no ha dado tiempo a hacerlo en el JSP, pero en un futuro lo implementaremos). Cuando la contraseña no es segura se lanza esta excepción explicándole al usuario los requerimientos para que la contraseña sea segura. Requiere sesión.

2.3. Endpoints y Métodos

Hemos mantenido los endpoints del servicio REST de la entrega anterior tal y como estaban, y hemos añadido uno nuevo para hacer el login con JWT. Antes hacíamos el login llamando

directamente a las clases de modelo, pero ahora lo hacemos a través del servicio REST y además usando la cookie con el JWT.

Endpoint	Método	Retorno	Descripción	Parámetros
/login	POST	application /json	Verificar las credenciales ingresadas por el usuario en el login.	FormParam: username (string)
				FormParam: password (string)
/debug-jws	GET	Text-plain	Desencriptar el JWE y retornar el JWS que está "dentro" del JWE.	Authorization header with the JWE

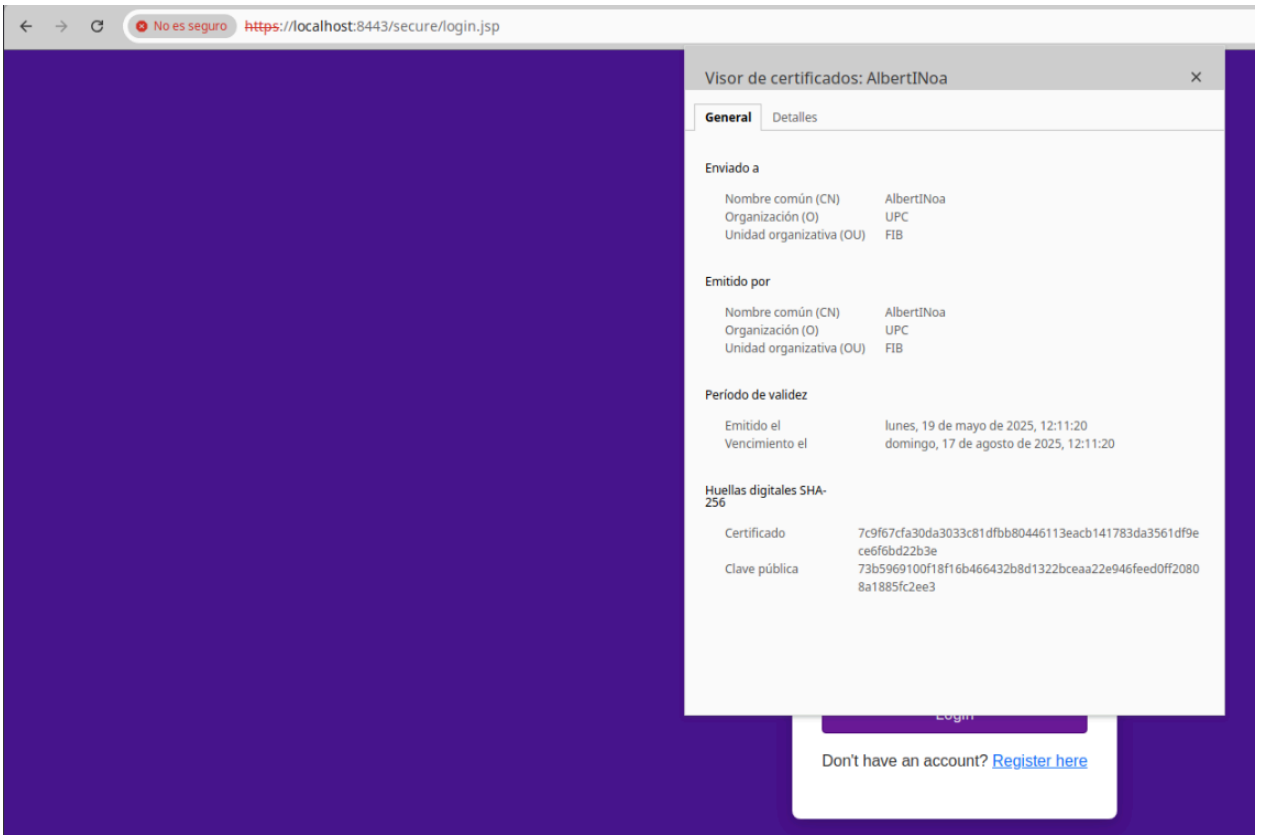
2.4. Funcionalidades

- Gestión de usuarios:
  - Inicio de sesión (REST y JWT)
  - Registro de usuarios
- Gestión de vídeos:
  - Reproducir vídeo (REST)
  - Subir vídeo (REST y XML Encryption)
  - Eliminar vídeo (REST)
  - Obtener vídeo(s) (con filtros) (REST)
  - Actualizar campos del vídeo (REST y XML Encryption)
  - Incrementar número de visitas del vídeo (REST y XML Encryption)
  - Descargar metadatos de un vídeo elegido

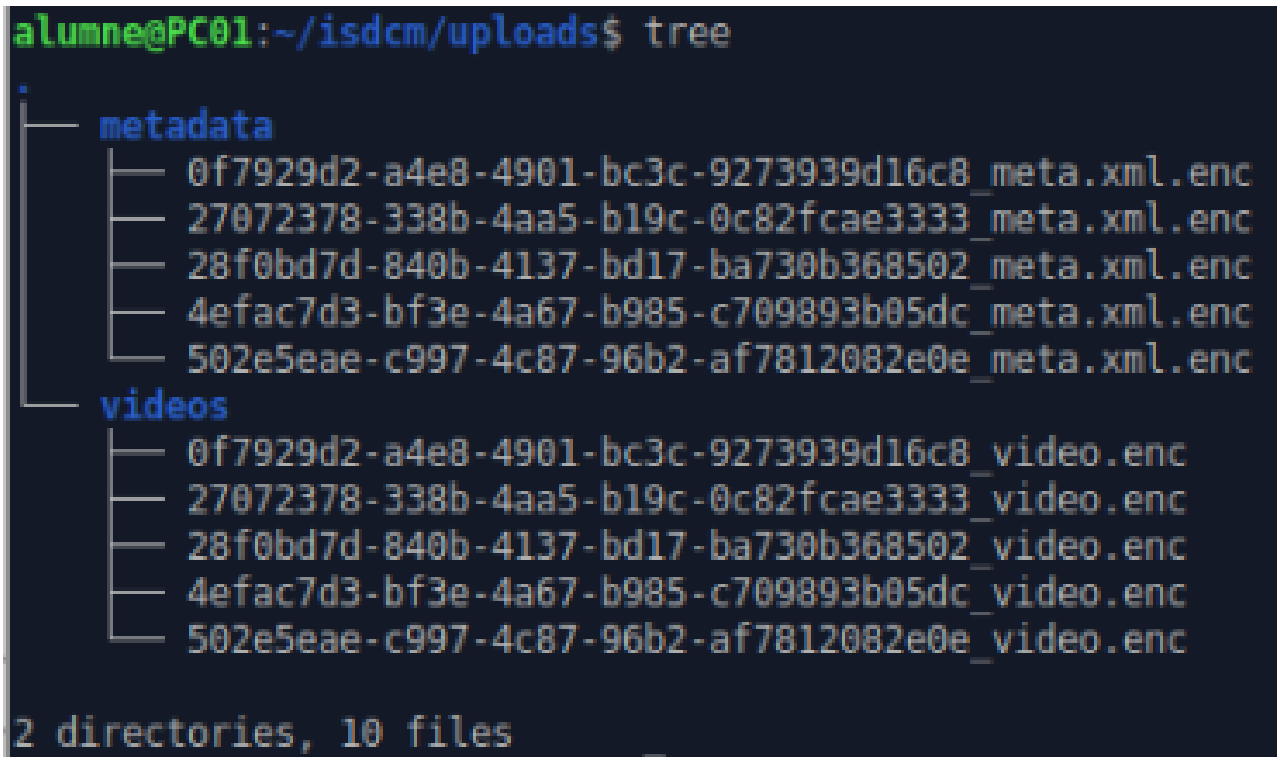
2.5. Demostraciones

Este apartado sirve para demostrar que los 3 apartados de esta entrega funcionan correctamente.

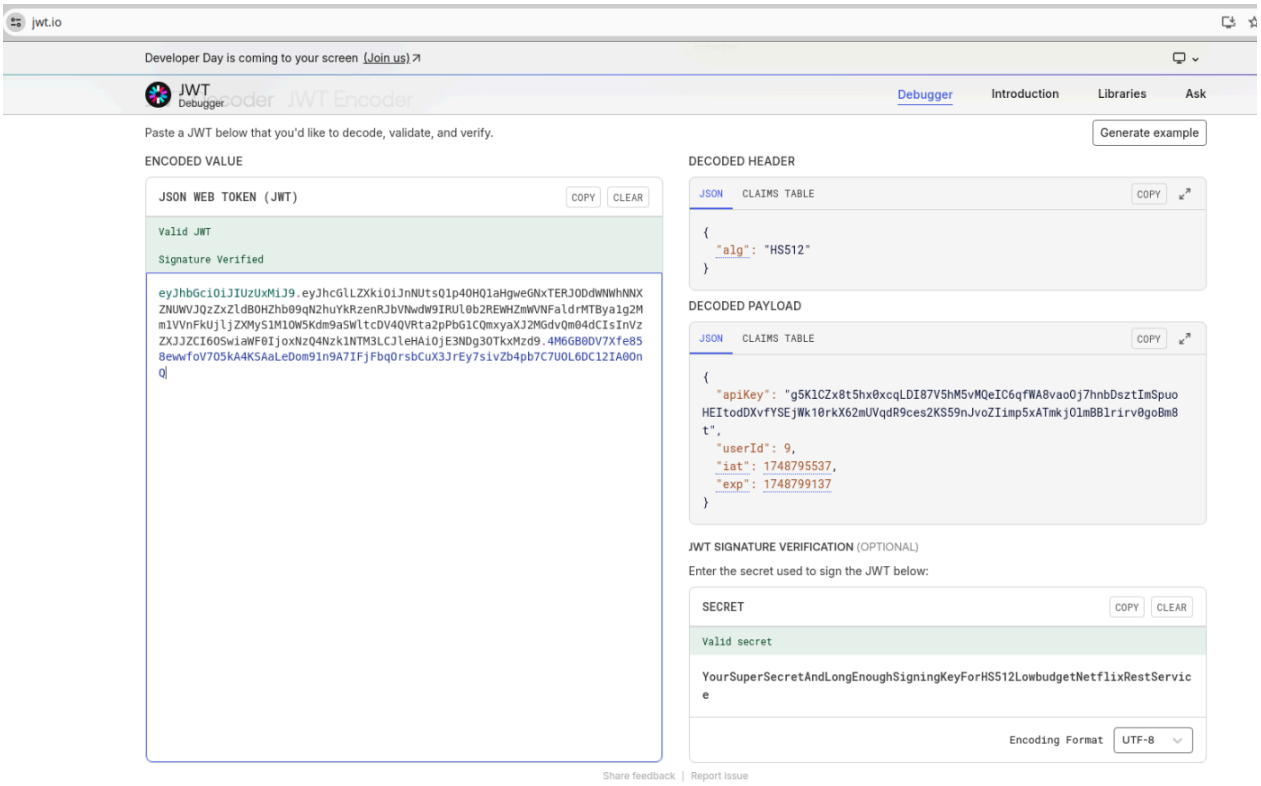
- Uso de un certificado para tener HTTPS:



- Encriptación y desencriptación de ficheros XML:



- Validación del JWT:



2.6. Aclaraciones

Para obtener esta configuración de Tomcat hemos seguido los pasos que hay en el enunciado de la entrega 3 paso a paso cambiando los nombres correspondientes (CN) en cada uno de los comandos a ejecutar. Previamente hemos eliminado la configuración existente como se pidió en el aviso del racó.

En nuestra implementación de XML Encryption encriptamos todo el documento entero, y luego antes de guardarlo en disco lo encriptamos con AES. De este modo tenemos dos métodos de seguridad, y además no permite ver que es un fichero de tipo XML.Cuando un usuario se descarga un fichero XML que no le corresponde, es descarga un fichero XML encriptado, por lo que no puede ver qué hay dentro. Solamente saber que es un XML.

Hay un botón para poder descargar el JWS generado con jose4j tal y como se nos pide en el enunciado. El JWT está encriptado con JWE (AES-256) para que sea seguro (sign-then-encrypt).

2.7. Extras realizados

- **Uso de JWT para una segura gestión de accesos:** además de poder descargar un JWT (desencriptando el JWE) con un botón y poder validarlo en jwt.io, hemos implementado un sistema funcional y seguro para el login de usuario y la gestión de su sesión que impide hacer peticiones a la api REST a cualquiera que no se haya logueado en la aplicación "principal".

- **JWT dentro de una cookie segura:** este JWT se sitúa en una cookie segura, lo que significa que la cookie tiene una configuración segura:

Atributo	Valor
secure	true
httponly	true
duration	session

3. Repositorios de código consultados

No hemos necesitado utilizar ningún repositorio de código.

4. Bibliografía consultada

Con el enunciado nos ha bastado. Hemos seguido las instrucciones paso por paso del enunciado para generar el certificado a añadir en HTTPS con la configuración de Tomcat. También para añadir el Encryption y Decryption de contenidos y XML, y la generación de un JWT desde el servidor. Muchos conceptos como el uso de JWT, gestión de cookies, entre otros, los hemos sacado de nuestra experiencia laboral.