Escuela Politécnica Superior de Linares

Grado en Ingeniería Telemática

UNIVERSIDAD DE JAÉN

Escuela Politécnica Superior de Linares

Trabajo Fin de Grado

SERVICIO DE PRESENTACIÓN VIRTUAL CONTROLADO DESDE TERMINAL MÓVIL

Alumno: **Sergio Caballero Garrido**

Tutor: Prof. D. Juan Carlos Cuevas Martínez

Depto.: Ingeniería de Telecomunicación

Noviembre, 2021

UNIVERSIDAD DE JAÉN

Escuela Politécnica Superior de Linares

Trabajo Fin de Grado

SERVICIO DE PRESENTACIÓN VIRTUAL CONTROLADO DESDE TERMINAL MÓVIL

Alumno: Sergio Caballero Garrido

Tutor: Prof. D. Juan Carlos Cuevas Martínez

Depto.: Ingeniería de Telecomunicación

Firma del autor

CABALLERO GARRIDO SERGIO - 71360191T

Firmado digitalmente por CABALLERO GARRIDO SERGIO - 71360191T
Fecha: 2021.11.12 20:07:24 +01'00'

Firma del tutor

CUEVAS MARTINEZ JUAN CARLOS - 26038239P

Firmado digitalmente por CUEVAS MARTINEZ JUAN CARLOS - 26038239P
Fecha: 2021.11.12 20:25:04 +01'00'

TABLA DE CONTENIDO

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

RESUMEN

El presente documento describe el trabajo fin de grado (TFG) el cual consiste en el desarrollo de un servicio de presentaciones controlado desde un terminal móvil. Con este sistema se consiguen mejoras y facilidad de acceso para la exposición de un documento, puesto que, desde un dispositivo móvil Android se puede preparar y dirigir la presentación de un documento.

El sistema está compuesto por el servidor, la aplicación móvil y el portal web. En el servidor se almacenan los datos y presentaciones que el usuario utiliza y realiza operaciones con estos a partir de peticiones HTTPS o *websocket*. Desde la aplicación móvil, el usuario podrá enviar o eliminar presentaciones en el servidor, así como preparar una sesión para exponer un documento ya almacenado. El portal web, por su parte, da acceso a la sesión que ha definido el usuario previamente. Al acceder al portal, se mostrará un código QR que será escaneado por la aplicación, desde la que se establecerá la conexión con el portal para comenzar la exposición y el control de la presentación.

El estudio y desarrollo del proyecto, ha permitido afianzar y poner en práctica el conjunto de conocimientos adquiridos en las asignaturas del grado, así como el aprendizaje autónomo y uso de nuevas tecnologías.

ABSTRACT

This document describes the final degree project, which consists in the development of a presentation service controlled from a mobile device. With this system, improvements and ease of access for the exhibition of a document are achieved, since, from an Android mobile device, the presentation of a document can be prepared and directed.

The system consists of the server, the mobile application and the website. The server stores the data and presentations that the user uses and performs operations with them based on HTTPS or websocket requests. From the mobile application, the user can send or delete presentations on the server, as well as prepare a session to expose a document already stored. The web portal, on the other hand, gives access to the session previously defined by the user. When accessing the portal, a QR code will be displayed and scanned by the application, from which the connection with the portal will be established to start the exhibition and the control of the presentation.

The study and development of the project has allowed to consolidate and put into practice the set of knowledge acquired in the subjects of the degree, as well as autonomous learning and the use of new technologies.

1 INTRODUCCIÓN

En la actualidad, la mayor parte de las lecciones en el aula se imparten apoyándose en una presentación que es elaborada por el presentador, bien sea docente o alumno, a modo de resumen de los conceptos a explicar. El uso de éstas, ayuda al asistente a comprender mejor el tema a tratar y de una forma más dinámica, puesto que en muchas ocasiones son utilizadas como esquema siendo completado con las explicaciones del instructor.

Cuando el personal docente o un alumno desea presentar diapositivas en el aula, generalmente, invierte un tiempo en iniciar la exposición. Para ello, es necesario abrir un documento desde medios como:

- PLATEA (platea.ujaen.es)
- Dispositivo de memoria USB.
- Presentaciones de Google.
- Aplicación de presentaciones Prezi.
- Microsoft Sway.
- Entre otros …

Cada uno de ellos cuenta con sus propias características con ventajas e inconvenientes. Para poder utilizar cualquiera de las plataformas online y mostrar los documentos, es imprescindible contar con las credenciales de acceso a la misma.

En el caso de la plataforma de enseñanza y aprendizaje de la Universidad de Jaén, PLATEA, cuyo aspecto es el que se muestra en la Ilustración 1.1, de forma general, su uso sería el siguiente: una vez que se ha accedido a la cuenta personal, navegará por la web hasta encontrar el espacio de la asignatura y posteriormente al documento deseado, donde se podrá descargar o abrir en una nueva ventana del navegador, en el caso de que sea compatible. Como ventajas, se tiene acceso a todas las funcionalidades de la plataforma como, por ejemplo, control de asistencia, configuración de fechas de entregas, acceso a otros documentos. Aunque su uso también puede presentar algunos de los inconvenientes tales como caídas de la plataforma, retrasos por sobrecarga de peticiones a la misma, así como olvidar cerrar sesión al finalizar la exposición.

Ilustración 1.1: PLATEA

Para las plataformas online como presentaciones de Google, Prezi o Sway, es posible acceder al documento a presentar mediante un enlace o accediendo a la cuenta personal. Los documentos desarrollados se pueden establecer como públicos o privados, en el caso de que sean públicos, es posible generar una URL con la que podrá acceder cualquier usuario para ver o editar el mismo. Cuando se va a exponer, si el presentador accede a su cuenta, podrá realizar modificaciones en cualquier momento de la lección.

Otro recurso es utilizar un dispositivo de memoria USB donde llevar almacenadas las presentaciones. Este método tiene como ventaja poder realizar modificaciones de la presentación en el momento de la exposición, si se abre con el programa de edición como PowerPoint, o realizar anotaciones en los documentos PDF. Algunos detalles a tener en cuenta, son el reconocimiento del dispositivo en el equipo, que en ocasiones puede fallar y la incompatibilidad de versiones del programa necesario para abrir el documento, como por ejemplo PowerPoint, Adobe Reader, etc., con respecto al programa utilizado para la creación del mismo.

Con el presente proyecto se facilita el acceso a documentos y su exposición mediante el uso de una aplicación móvil y un portal web, distribuido como se muestra en Ilustración 1.2. Para preparar una exposición desde la aplicación móvil, una vez el usuario se ha autenticado, podrá subir o eliminar las presentaciones, definir las sesiones o manejar el control del documento. Las presentaciones se podrán enviar desde la aplicación al servidor, seleccionándolos desde el explorador de archivos del dispositivo. Posteriormente, se creará una sesión a partir de las presentaciones registradas y un nombre identificativo. Por otro lado, desde un navegador, se accederá al portal web de la sesión, generando un

código QR que será interpretado desde la aplicación para iniciar la exposición, a partir de entonces, se utilizará el dispositivo móvil a modo de mando a distancia de la presentación. El control de la presentación cuenta con las opciones de cambiar de página, control de zoom y movimientos, y envío de notas.

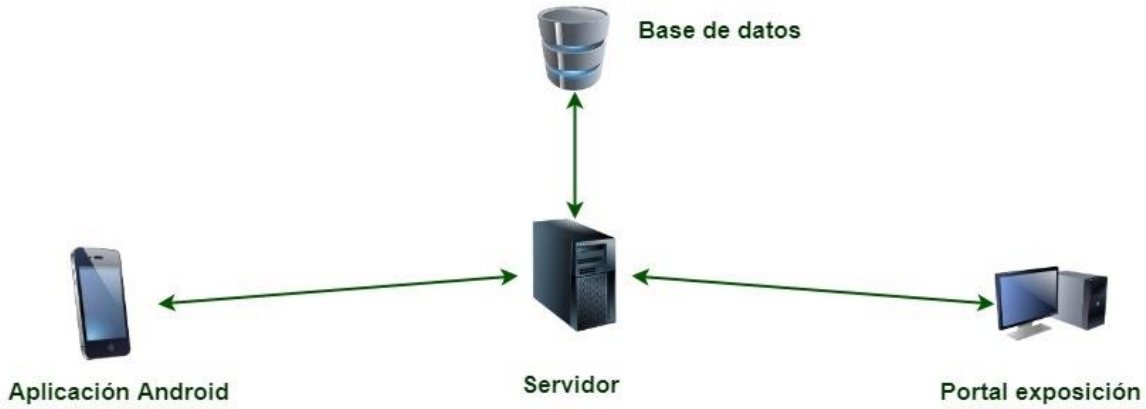

Ilustración 1.2: Esquema del sistema

1.1 Objetivos

Este trabajo tiene como objetivo diseñar y desarrollar un servicio telemático que permita a un cliente (navegador) acceder a un servicio remoto de presentación de diapositivas que esté controlado a su vez por otro cliente que funcionará a modo de control remoto (terminal móvil o navegador en terminal móvil) que aporte las funciones típicas para el control de una presentación, las cuales, como mínimo serán las siguientes:

- Lanzar una nueva presentación
- Cerrar una presentación
- Avanzar y retroceder las diapositivas.

El sistema debe aportar un mecanismo flexible de autenticación basado en la lectura de un código QR en la página web del servicio siguiendo el siguiente proceso:

- Autenticación en la web o aplicación del terminal móvil que ejercerá como control remoto (App Remota)
- Entrada en la web del portal que representará las diapositivas e iniciará una nueva presentación, lo cual mostrará un código QR identificativo.
- Captura del código QR por parte de la aplicación remota y envío de los datos al servidor del portal.
- Selección de la presentación a mostrar desde la aplicación remota.
- Si los datos tienen validez, comenzará la presentación según comande la App Remota.

Objetivos opcionales:

- Control del puntero del Portal por parte de la App Remota.
- Envío de texto desde la aplicación remota a la pantalla del Portal
- Envío de dibujos a mano alzada desde la App Remota al Portal.

1.2 Estado del arte

En la actualidad, no existe ningún servicio similar al del presente trabajo, no obstante, hay diversas plataformas en línea de presentación de documentos. Algunas de las más conocidas de almacenamiento, creación y exposición de presentaciones son *Slideshare*, Presentaciones de Google (*Slides*), *Sway* de *Microsoft* o *Prezi*, entre otras.

1.2.1 SlideShare

El portal *SlideShare*, fue creado en 2006 con el fin de intercambiar conocimientos hasta convertirse en la actualidad en un recurso bastante utilizado en el ámbito profesional, estando entre las 100 webs más frecuentadas del mundo [1].

Esta web se clasifica como un sitio web dónde se comparte información y está diseñado para el usuario, esto es conocido como sitio web 2.0. Los usuarios podrán compartir contenido de forma pública o privada. Mencionados contenidos pueden tratarse desde documentos en formato PDF, textos con formato como *Microsoft W*ord u *OpenOffice*, textos sin formato (con extensión .txt), o hasta ficheros de audio o vídeo. En primera instancia, el sitio web se diseñó para el ámbito empresarial y educativo, pero posteriormente se amplió a ofrecer entretenimiento.

En 2012 *LinkedIn* compró esta plataforma evolucionando posteriormente la web orientándola más hacia el ámbito profesional, implementando nuevas herramientas y opciones como mejoras de organización o búsquedas de expertos por categorías. En septiembre de 2020, *LinkedIn* vende SlideShare a la plataforma *Scribd* [2].

SlideShare cuenta con servicios adicionales complementarios como son:

- ***Zipcast hd***: presentación de documentos, a través de una conferencia de audio y vídeo, mientras que el presentador controla la presentación por internet. También dispone de chat. Los espectadores tienen permitido navegar entre las diapositivas.
- ***Clipping***: soporte de descarga de diapositivas individuales de presentaciones.
- ***Slidecast***: mejora las presentaciones de la plataforma permitiendo incluir voz o música de forma sincronizada con las diapositivas.

1.2.2 *Presentaciones de Google*

La herramienta de Presentaciones de Google (en inglés *Slides*), permite crear y editar presentaciones directamente desde el navegador sin ser necesaria la instalación de algún programa adicional. Una de las características a destacar de esta herramienta es la posibilidad de crear un documento de forma colaborativa, permitiendo a los colaboradores ver, editar o incluir comentarios [3]. Está disponible de forma gratuita para los usuarios de una cuenta de Google.

Las presentaciones son fáciles de diseñar, se puede elegir un diseño de plantilla, añadir imágenes, vídeos y transiciones. Los cambios que se van realizando se almacenan automáticamente en línea, manteniendo las versiones de las modificaciones de manera indefinida. El formato del documento no es un problema puesto que se puede convertir desde otros formatos al propio de presentaciones de Google y viceversa.

Es multiplataforma, se puede utilizar en ordenadores de cualquier sistema operativo, así como en teléfonos móviles y tabletas. Además, es compatible con dispositivos Chromecast, permitiendo mostrar las presentaciones de forma inalámbrica a una pantalla compatible [4].

Algunas herramientas útiles de Presentaciones de Google son:

- Vincular diapositivas entre presentaciones, de forma que, modificando la diapositiva en una presentación, automáticamente se modifica en la otra.
- Durante una exposición, el presentador puede abrir una sesión de preguntas a la que los espectadores accederán a un enlace y podrán formular preguntas, así como valorarlas para posteriormente el presentador las muestre en la presentación y pueda responderlas [5].

1.2.3 *Sway Microsoft*

Sway, la utilidad de la familia de productos de *Microsoft* creada en 2014, permite crear boletines, presentaciones y comunicaciones en línea. Es multiplataforma, pudiendo utilizarse desde la web o desde sus aplicaciones. Los documentos se pueden compartir mediante un enlace, con el que cualquier usuario podrá acceder para ver o editar, en función de la configuración al generarlo. Cualquier usuario con una cuenta *Microsoft* tiene acceso a este programa de forma gratuita [6].

Las presentaciones cuentan con un diseño sencillo, el cual se puede personalizar al estilo propio del usuario y se adaptan automáticamente a cualquier tamaño de pantalla. Para aumentar la interactividad de los documentos es posible agregar:

- Imágenes, audio y vídeo.
- Cualquier contenido de fuentes como Bing, OneDrive o redes sociales.

- Un cuestionario y poder hacer un seguimiento de las respuestas de los espectadores.
- Una comparación dinámica de imágenes unidas por un divisor que los usuarios podrán desplazar con el fin de ver los detalles de la comparativa.
- Conjunto de imágenes agrupadas en galerías que el espectador podrá verlas individualmente.

1.2.4 Prezi

Otra de las plataformas web más conocidas para crear y mostrar presentaciones es *Prezi*. La diferencia respecto a otras aplicaciones es el estilo y la forma de las presentaciones. Prezi expone un lienzo abierto dónde se pueden organizar los elementos y animaciones que serán visualizados en conjunto. Su interfaz de creación es sencilla, se pueden agregar elementos arrastrando el contenido hasta la posición deseada.

Al presentar un documento, consta de llamativos efectos visuales como zoom y movimientos, evitando saltar entre diapositivas. Ofrece plantillas prediseñadas para agilizar la creación de las presentaciones.

También es multiplataforma, se puede utilizar desde un navegador o descargando un *software*. Existen diferentes modalidades de licencia, gratuitas y de pago [7].

2 DESCRIPCIÓN DEL SISTEMA DESARROLLADO

2.1 Introducción

El sistema consta de tres partes: servidor, aplicación móvil y portal web. El servidor, proporciona los mecanismos necesarios para visualizar y dirigir las presentaciones. La aplicación móvil o cliente, que hará uso de dichos mecanismos del servidor para establecer sesión, subir y eliminar presentaciones y exponer las diapositivas. Por último, el portal web, donde se muestra la presentación.

La principal funcionalidad de este servicio consiste en mostrar presentaciones en un navegador, sin necesidad de acceder a una plataforma web o usar una memoria USB. Dichas presentaciones deberán estar cargadas previamente en el servidor. Posteriormente, una vez que el usuario acceda a la URL de la sesión que ha definido, procederá a la lectura de un código QR mediante la aplicación que le dará acceso a la exposición del documento y conexión de la aplicación con la presentación. La carga del documento en el navegador se realizará en segundo plano descargándose en la caché. Para poder utilizar la aplicación móvil es necesario contar con unas credenciales para autenticarse.

2.2 Tecnologías utilizadas

En primer lugar, el elemento central del sistema, el servidor o *back-end*, ha sido desarrollado sobre *NodeJS* utilizando el framework *express* con el que se manejan las peticiones recibidas. Además, se han utilizado las librerías *Socket.IO* (para la comunicación *websocket*), *JWT* (utilizada en la autenticación y autorización de peticiones) o *PDFJS-dist* (representación de la presentación), entre otras, para el correcto funcionamiento en conjunto.

Los datos que se manejan durante el uso del sistema se registran en la base de datos que utiliza el gestor de base de datos MySQL.

La aplicación móvil se ha desarrollado para Android utilizando el lenguaje Java y las clases propias de Android como son *Activity o Fragment.* Las solicitudes hacia el servidor se realizan mediante la librería *Volley* excepto para el envío de archivos, en cuyo caso se utilizará *Upload Service*. La comunicación *websocket* con el portal hace uso de la librería *Socket.IO*.

La descripción de las tecnologías y conexión entre cada uno de los elementos, se resume en la Ilustración 2.1.

Ilustración 2.1: Esquema del sistema con tecnologías

2.2.1 Servidor

La parte del servidor o *back-end*, se encarga de las funcionalidades propias del sistema, de forma que, atiende y resuelve las peticiones que se realicen desde la aplicación o desde el navegador, así como gestiona los datos y documentos pertenecientes al mismo. En concreto, estas funcionalidades implementadas, son la gestión de peticiones web; lectura, escritura y borrado de ficheros; lectura y representación de documentos PDF en el navegador; y por último comunicación vía *websocket* (apartado 2.2.7) del terminal móvil con la exposición de documentos.

El servidor utiliza dos tecnologías: *Node.JS* (apartado 2.2.3) para la implementación del sistema y una base de datos MySQL (apartado 2.2.6) dónde se almacenan los usuarios, los nombres de las presentaciones almacenadas en el servidor y las sesiones que se han definido para el usuario.

Las peticiones web que se han implementado con el módulo *express* (apartado 2.2.4) de *Node.JS*, desarrolladas en un estilo de arquitectura REST, son:

- **Registro de usuario**: a partir de los datos de usuario recibidas se registrará en la base de datos.
- **Acceso de usuario:** mediante las credenciales del usuario y una consulta a la base de datos se da acceso a la aplicación, en el caso de que sean correctas.
- **Crear y eliminar sesión:** permite definir o anular una sesión. Dicha sesión está caracterizada por los nombres de usuario, de la presentación y de la sesión.

- **Solicitud de listado de presentaciones o sesiones para un usuario:** realiza una consulta a la base de datos y devuelve el listado correspondiente.
- **Cargar y eliminar documentos:** almacena en el servidor documentos en formato PDF y a su vez, se registra en la base de datos.
- **Página de exposición de presentaciones:** acceso al portal indicando el usuario y el nombre de la sesión que debe estar definida con antelación. Si los datos son correctos mostrará un código QR para establecer la conexión con la aplicación móvil, en caso contrario se mostrará una página en la que informa que la sesión no está definida.

Para representar las diapositivas en el navegador se ha utilizado un módulo de *Node.JS*, *pdfjs-dist* [21], que facilita el control de páginas y zoom, mediante código *JavaScript*. Esta representación permite, además, la selección de texto y acceso a enlaces integrados en el documento. La presentación se descargará en la memoria caché del navegador cuando se accede a la página de la sesión mientras que se muestra el código QR, por lo que no será necesario volver a cargar el documento al cambiar de páginas y se evita el tiempo de carga prolongado inicial.

La tecnología *websocket* posibilita la conexión entre navegador, servidor y aplicación cliente. Define una sesión de comunicación bidireccional, sin ser necesario realizar consultas al servidor para obtener una respuesta. Por ello, se ha utilizado para el control de la presentación, mediante el envío y recepción de mensajes.

La exposición de un documento estará disponible en la dirección URL de una sesión previamente creada a partir de una presentación ya existente. Al acceder al portal, se mostrará un código QR generado con los datos identificativos de la sesión. El código QR será escaneado desde la aplicación móvil creando una conexión *websocket* que permite el control de la presentación. En los mensajes intercambiados en la conexión se indican los comandos con los que se permite el cambio de página, control de tamaño y posición de la presentación, así como el envío de notas.

2.2.2 Aplicación Android, cliente

En cuanto a la aplicación móvil (cliente), es la herramienta que utilizará el usuario para poder interactuar con el servicio. En ella se pueden diferenciar cuatro secciones o pantallas, cada una con un fin determinado:

- **Pantalla de bienvenida** (conocida en inglés como *splash screen*): muestra información básica del proyecto, concretamente, el título del trabajo, titulación, logo de la escuela, autor y tutor.

- **Acceso de usuario:** con el que se permite acceder a las funcionalidades propias de la aplicación, introduciendo y validando en el servidor las credenciales, nombre de usuario y contraseña.
- **Actividad principal:** consta de un menú con acceso a las siguientes funcionalidades, divididas en fragmentos:
    - **Crear sesiones**: permite definir una nueva sesión, eligiendo el documento a presentar de la lista obtenida y estableciendo un nombre. Si la sesión ya está creada en el servidor se actualizará.
    - **Subida de presentaciones al servidor**: consiste en la búsqueda de un documento PDF almacenado en el dispositivo y posteriormente cargarlo en el servidor.
    - **Eliminar presentaciones**: selección de una presentación ya almacenada para eliminarla del servidor.
    - **Lectura del código QR**: permite acceder al control de la presentación si la sesión que se interpreta del código se ha creado por el usuario autenticado en la aplicación.
    - **"Acerca de"**: muestra la información de la pantalla de bienvenida, así como la descripción de las funcionalidades de la aplicación.
- **Control de presentación:** proporciona los controles de la presentación como son cambiar de diapositiva, adaptación de tamaño (zoom) y posición de la presentación y envío de notas. Al acceder a esta vista se realiza la conexión con el servidor vía *websocket* y se inicia la presentación en el navegador.

Para realizar las peticiones al servidor, se ha hecho uso de una serie de librerías, las cuales facilitan tanto el establecimiento y conexión con el servidor, como el desarrollo de la aplicación. Dichas librerías son, *Volley, Upload Service y SocketIO*.

La librería *Volley* [8] se utiliza para realizar las peticiones HTTPS al servidor. Mediante esta utilidad se facilita la definición de solicitudes al servidor, es decir, el envío de las cabeceras HTTPS y de los parámetros necesarios en cada petición. Las peticiones en las que se hace uso de esta librería son: registro de usuario, envío de credenciales para autenticarse, petición del listado de documentos y sesiones disponibles en el servidor, crear y eliminar sesión y por último eliminar presentación.

Se ha elegido la librería *Upload Service* (de Aleksandr Gotev [9]) para permitir la subida de los documentos PDF al servidor de forma asíncrona y mostrando el progreso en una notificación.

Para la comunicación vía *websocket* de la aplicación con el servidor se ha hecho uso de la librería *SocketIO-client* [10]. Esta librería permite el establecimiento de conexión,

envío y recepción de mensajes de texto. Los mensajes que se intercambian están establecidos por un protocolo diseñado para este fin, desarrollado en el apartado 2.4.8.

2.2.3 NodeJS

Node.JS, de la fundación *OpenJS* (https://openjsf.org), es un entorno de ejecución de *JavaScript*, multiplataforma, el cual, es utilizado en gran medida para desarrollar aplicaciones web que puedan crecer fácilmente. Internamente, *Node.JS* está programado en C++, es de código abierto y utiliza el motor para *JavaScript* V8 de Google, por lo que cuenta con buen soporte y alto rendimiento. Al tratarse de un interpretador de código JavaScript unifica el lenguaje de programación, pudiendo utilizarse este tanto en el servidor (*back-end*) como en el cliente (*front-end*). Este entorno está orientado a eventos asíncronos de forma que no se realiza un bloqueo del servidor ni realiza Entrada/Salida directamente, mencionadas características permiten simplificar la programación sin asumir problemas de concurrencia. Esto es conocido como el ciclo de eventos (*Event Loop*), solo permite la ejecución de un evento al mismo tiempo en el hilo principal, aunque admite simultaneidad de eventos y devolución de llamadas (*callbacks*). Otra de las ventajas de *Node.JS* es que admite múltiples conexiones, respondiendo para cada una de ellas el resultado correspondiente en la devolución de llamada, cuando no recibe peticiones se quedará en estado inactivo. El uso de *Node.JS* es recomendable para conexiones persistentes, procesamiento de datos en tiempo real y sockets [11].

Este entorno de ejecución está orientado para crear aplicaciones de red escalables, por ello se ha elegido para el desarrollo de la parte del servidor del presente sistema. También dispone de amplia variedad de librerías públicas, algunas desarrolladas por la comunidad, con las que manejar funcionalidades necesarias.

En los rankings anuales publicados por la web *StackOverflow* (https://insights.stackoverflow.com/survey/2021), para el año 2021, el lenguaje de programación más utilizado según el 64,96% de los encuestados es *JavaScript* y *Node.JS* se encuentra como la sexta tecnología más utilizada, con un 33,91%.

Algunas de las plataformas web más reconocidas desarrolladas con este entorno son *eBay, PayPal, Netflix o Linkedin* [12].

El servidor se ha desarrollado en *Node.JS* gracias a las características propias descritas anteriormente, ya que permite adaptarse en gran medida al sistema del presente proyecto. Entre las características que han facilitado su elección para el desarrollo del servidor destacan, la posibilidad de usarlo tanto para *back-end* como *front-end* (para el caso del portal de exposición) y la amplia variedad de librerías públicas que facilitan el progreso del sistema, como son la de lectura y representación del documento o la conexión a la base de datos.

2.2.4 Framework Express

Express es un *framework* de código abierto con licencia MIT (licencia de software del Instituto Tecnológico de Massachusetts) utilizada para crear aplicaciones en *back-end* como APIs (Interfaz de programación de Aplicaciones) y aplicaciones web. Fue creado en noviembre de 2010 y su última versión estable es la 4.17.1 publicada en mayo de 2019, en la actualidad se está trabajando en la versión 5.0.0 alpha. El *framework Express* es el más conocido de *NodeJS*, por ello, asegura el mantenimiento y actualizaciones, así como, encontrar documentación y soluciones de problemas más fácilmente [13].

El *framework* ofrece mecanismos para el control de peticiones mediante distintos métodos HTTP (GET, POST, PUT, HEAD, DELETE) y rutas o URLs, del mismo modo, motores de renderizado de vistas con inyección de datos en plantillas, configuración de ajustes tales como la localización de dichas vistas, ficheros necesarios o el puerto de uso, y, además, permite añadir procesado de peticiones mediante middleware. Esta última característica permite ampliar las posibilidades de resolución de problemas con las librerías compatibles o metodologías propias, como por ejemplo operar con cookies o sesiones, parámetros URL, cabeceras de seguridad entre otras. Los middlewares, pueden acceder a los parámetros de la solicitud, los de la respuesta y continuar con el ciclo de la petición, permitiendo la ejecución de tareas, así como modificar el ciclo de la solicitud [14].

Para este sistema se ha elegido el *framework Express* por ser uno de los más conocidos para *NodeJS* y por la facilidad de implementación tanto para la API como para el manejo de vistas *HTML*.

2.2.5 JSON Web Token

Como medida de seguridad, las peticiones deberán estar reconocidas mediante la cabecera "Authorization" que incluirá un *token* con el que se identifica al usuario. Este *token* se generará mediante *JWT* en la autenticación del cliente y será comprobado mediante el middleware de *express* en cada petición.

Un *token* se define como la transformación de datos denominados sensibles, que pueden leerse con facilidad, a una cadena de caracteres no sensibles y sin significado de forma que a partir del *token* no es sencillo obtener los datos sensibles [15].

JWT (*JSON* Web *Token* [16]) es un estándar abierto el cual especifica una configuración compacta y autónoma para la transferencia de información entre las partes comunicantes de forma segura y cuyo formato es un objeto *JSON*. Además, la información que se transmite está firmada, de forma que puede ser verificada aportando confiabilidad e integridad evitando que sea modificada por terceras partes. La firma puede ser utilizando una clave secreta, mediante el algoritmo *HMAC*, o el par de claves pública y privada mediante *RSA* o *ECDSA*. En el caso de que el *token* se firme mediante el par de claves

pública y privada, permite garantizar el emisor puesto que las claves son únicas para este. *JWT*, además de los *tokens* firmados, permite cifrar los mensajes entre ambas partes.

El ciclo de uso de este *token* para la autenticación, comienza cuando el usuario realiza una petición de inicio de sesión en la API con sus credenciales, generando ésta el *token* que será enviado en la respuesta de la petición. Cuando el usuario quiera realizar una petición a un recurso protegido de la API es necesario que envíe junto con el resto de cabeceras, el encabezado de autorización indicando el *token*, generalmente tiene el aspecto que se muestra a continuación: `Authorization: Bearer <token>` Una vez el servidor ha recibido la cabecera A*uthorization* en los recursos protegidos, comprueba la validez del *token*, extrae los datos incluidos en el mismo y por último se ejecutan los procesos de la propia petición.

Es imprescindible tener cuidado con los *tokens* ya que están formados a partir de credenciales hay que procurar evitar posibles problemas de seguridad, los *tokens* no deberán permanecer activos más de lo fundamental. Del mismo modo, para los *tokens* firmados, es necesario considerar que la información incluida en la carga útil puede ser decodificada por terceras partes, aunque no se podrán realizar modificaciones ya que la firma quedaría inválida, por lo tanto, no deben contener datos privados [17].

2.2.6 MySQL

Como sistema gestor de bases de datos SQL (*Structured Query Language*) para el presente trabajo, se ha elegido MySQL, de código abierto y desarrollado por Oracle Corporation.

Se define base de datos como un conjunto estructurado de datos, desde listas simples, imágenes o grandes cantidades de datos pertenecientes a una organización. Las bases de datos gestionadas por MySQL son de tipo relacionales, es decir, los datos se encuentran en tablas separadas y organizados en ficheros diseñados para tener mayores velocidades de lectura y escritura. En cuanto al modelo lógico, cuenta con objetos tales como bases de datos, vistas, tablas, filas y columnas. Las tablas pueden estar relacionadas entre sí por sus campos, de uno a uno o de uno a muchos y en cada tabla, los datos en las filas pueden ser opcionales, obligatorios o únicos. Todas estas características, configurando de forma correcta las estructuras de las tablas hacen que la base de datos sea consistente de forma que no puedan existir datos duplicados, referencias a datos inexistentes o desactualizados.

El servidor MySQL, se define como rápido, escalable, confiable y de uso sencillo. Por ello, puede alojarse tanto en un ordenador personal como en servidores dedicados, en este último caso, es posible configurarlo de forma que se utilice toda la capacidad del equipo de memoria, procesamiento y conexiones.

MySQL fue desarrollado con el fin de gestionar grandes cantidades de datos de forma más rápida que las alternativas disponibles en el momento de su creación, llegando a ser en la actualidad, de las más utilizadas en el ámbito de producción. Debido a su éxito se ha continuado su desarrollo, mejorando la conectividad, seguridad y velocidad [18].

2.2.7 Websocket

Tradicionalmente, las aplicaciones web que requerían comunicación entre el cliente y el servidor, era necesario un alto uso de peticiones HTTP con el fin de buscar actualizaciones en el servidor. Como solución a este problema se diseñó el protocolo *WebSocket*, mediante el que es necesario una única conexión TCP para el tráfico en ambos sentidos. Este protocolo combinado con *Websocket* API ofrece otra posibilidad al sondeo HTTP en las comunicaciones bidireccionales desde la aplicación cliente al servidor remoto. Las aplicaciones cliente pueden ser diversas, tales como: aplicaciones multiusuario, juegos, interfaces de usuario con actualizaciones en tiempo real, etc. El protocolo está definido en RFC 6455: *The WebSocket Protocol* [19].

El protocolo *WebSocket* reemplaza las tecnologías de conexión bidireccionales soportadas sobre la capa de transporte HTTP, utilizando la infraestructura existente, añadiendo eficiencia y confiabilidad ya que el protocolo de transporte no estaba inicialmente diseñado para comunicaciones en ambos sentidos. *WebSocket* está diseñado para poder operar en los puertos 80 y 443 e incluso admite intermediarios y proxies HTTP. Una vez que se ha establecido la conexión entre cliente y servidor de forma exitosa, mediante una petición de actualización HTTP, comienza la transferencia de datos. Esta transferencia se realiza en un canal de comunicación bidireccional donde ambos extremos pueden enviar datos (mensajes) de forma independiente.

Por estas características se ha elegido este protocolo para el control de la presentación durante la exposición con el envío de comandos interpretados por el sistema al completo (servidor, aplicación móvil y portal web).

2.2.8 Android

Las aplicaciones Android pueden ser desarrolladas sobre los lenguajes de programación Java, Kotlin y C++. Mediante las herramientas Android SDK, la aplicación se compila añadiendo los recursos y archivos necesarios en un paquete con la extensión APK. Este archivo APK incluye todo lo necesario para la instalación de la aplicación, los dispositivos Android lo utilizan para la instalación de la aplicación.

Por las características del sistema operativo Android, siguiendo “el principio de mínimo privilegio” las aplicaciones, por defecto, únicamente pueden tener acceso a sus propios componentes. De esta forma, se genera un entorno seguro ya que no es posible acceder a entornos del sistema donde no se le ha concedido acceso. A pesar de ello, una

aplicación puede acceder a los datos de otra o del sistema. Para que una aplicación tenga acceso a información del dispositivo, como puede ser los contactos, almacenamiento, cámara, es necesario que solicite expresamente al usuario los permisos correspondientes.

Las aplicaciones Android están estructuradas en diversos bloques o componentes, cada uno de ellos permite interacción entre el sistema y el usuario. Estos componentes son actividades (clase *Activity*), fragmentos (clase *Fragment*), servicios, receptores de emisiones y proveedores de contenido, cada uno de ellos tiene una finalidad específica, así como un ciclo de vida propio. Los componentes de una aplicación y los permisos necesarios de la misma, deben estar declarados en el archivo de manifiesto (AndroidManifest.xml) a fin de que el sistema Android pueda reconocer e iniciar los componentes de la propia aplicación [20].

2.3 Base de datos

La base de datos, denominada "presentacionvirtual", diseñada para el presente trabajo, cuenta con un total de tres tablas o entidades en las que se registran los datos de los usuarios, propiedades de las presentaciones y las definiciones de cada sesión.

2.3.1 Entidad Usuarios

En la entidad usuarios, se declaran los datos pertenecientes al usuario, así como un *token* temporal generado al iniciar sesión, que contiene algunos de estos datos, la fecha y hora de la autenticación. En la Tabla 2.1 se describen en detalle los campos y su uso.

Tabla 2.1: Campos entidad usuario

Nombre del campo	Tipo	Descripción
idusuario	Numérico (INT)	Identifica unívocamente al usuario en la base de datos. No se permite dejar vacío el campo
nombreusuario	Cadena de texto (VARCHAR(45))	Identifica unívocamente el usuario en el sistema. No se permite dejar vacío el campo
password	Cadena de texto (VARCHAR(200))	Contraseña de acceso para el usuario. No se permite dejar vacío el campo
nombre	Cadena de texto (VARCHAR(45))	Nombre del usuario
apellidos	Cadena de texto (VARCHAR(45))	Apellidos del usuario
email	Cadena de texto (VARCHAR(45))	Dirección de correo electrónico del usuario
token	Cadena de texto (VARCHAR(255))	*Token* identificativo generado al iniciar sesión

2.3.2 Entidad Presentaciones

Para mantener un registro de las presentaciones que los usuarios publican desde la aplicación, se define la entidad presentaciones. En la entidad, se almacena la información de las mismas, como se muestra en la Tabla 2.2.

Tabla 2.2: Campos entidad presentaciones

Nombre del campo	Tipo	Descripción
idpresentacion	Numérico (INT)	Identifica unívocamente la presentación en la base de datos
presentacion	Cadena de texto (VARCHAR(100))	Nombre completo del documento
paginas	Numérico (INT)	Número de páginas de la presentación. Se permite dejar vacío el campo
nombreusuario	Cadena de texto (VARCHAR(45))	Usuario propietario del documento

2.3.3 Entidad Sesiones

Por último, en la entidad sesiones se inscriben las sesiones que genera el usuario a partir de una presentación y un nombre identificativo. La entidad contiene los campos que se detallan en la Tabla 2.3.

Tabla 2.3: Campos entidad sesiones

Nombre del campo	Tipo	Descripción
idsesion	Numérico (INT)	Identifica unívocamente la sesión en la base de datos
sesion	Cadena de texto (VARCHAR(50))	Nombre identificativo de la sesión
presentacion	Cadena de texto (VARCHAR(100))	Nombre completo del documento
nombreusuario	Cadena de texto (VARCHAR(45))	Usuario creador de la sesión

2.3.4 Relaciones entre entidades

Las entidades descritas en el apartado anterior están relacionadas entre sí de la siguiente forma:

- **Usuarios - presentaciones (1:N):** un usuario puede tener registradas varias presentaciones y un documento únicamente puede tener un usuario.
- **Usuarios - sesiones (1:N):** una sesión puede pertenecer únicamente a un usuario, pero a su vez, un usuario puede tener registradas varias sesiones.
- **Presentaciones - sesiones (N:N):** una presentación puede aparecer en varias sesiones y de igual forma en el sentido contrario.

Estas relaciones están representadas en la Ilustración *2.2*. Cada una de las entidades tiene establecidas una clave primaria, representadas en la imagen como PK, correspondiendo con el índice de cada una de ellas, con la que se identifica unívocamente la fila de datos. Adicionalmente, se encuentran las claves únicas que evitan el duplicado de datos que puedan inducir a errores en el funcionamiento del sistema, marcadas en la imagen como UK. Las claves únicas corresponden, en la entidad usuarios al nombre de usuario, en la entidad presentaciones al nombre del documento y el usuario que lo ha registrado, y, por último, la entidad sesiones, el nombre de la sesión y el del usuario.

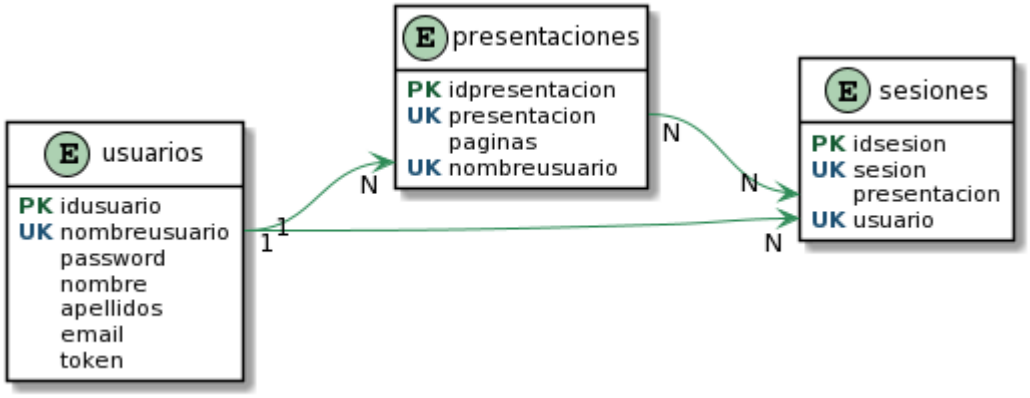

Ilustración 2.2: Relaciones entre tablas

2.4 API REST

La API que define las operaciones del sistema se puede diferenciar diversas secciones en función del recurso al que hagan referencia la acción, contando así con tres secciones como son, usuario, presentaciones y sesiones, tal como se muestra en la Ilustración 2.3. Todas las peticiones se realizarán mediante HTTPS al servidor donde se encuentra alojado y cuya ruta base es "/virtualpresentation/". Al mismo tiempo, las solicitudes podrán ser únicamente con los métodos GET, POST, PUT y DELETE, si se realiza una petición con un método distinto, se producirá un error y el servidor responderá con el código de estado 405 (*Method Not Allowed*).

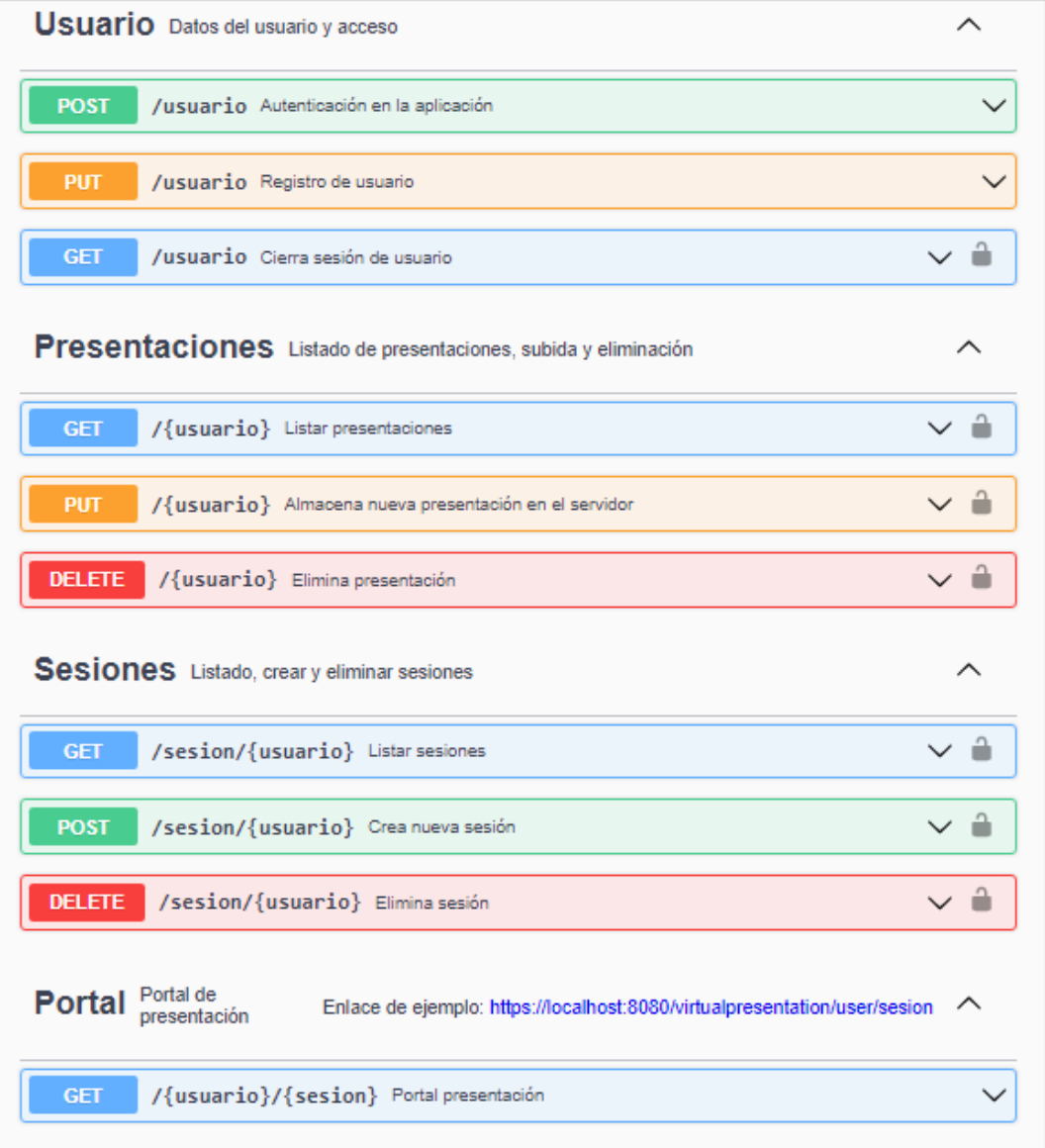

Ilustración 2.3: Documentación servidor, resumen de recursos y acciones

Para el uso de cualquiera de los recursos de la API, exceptuando la autenticación, registro de usuario y acceso al portal, es necesario incluir una nueva cabecera en la petición con el nombre "Authorization" cuyo valor será un *token* identificativo de la sesión del usuario. Este *token* de usuario se obtendrá mediante la petición de autenticación y tendrá una validez de 2 días. Los recursos de la API que lo requieran, en primer lugar, realizará una comprobación a la base de datos para verificar que el *token* es correcto, posteriormente comprobará la fecha de expiración y, por último, da paso al procesado de la petición en sí. La comprobación del *token* puede detener la solicitud y devolver uno de los siguientes errores que se muestran a continuación junto con el motivo:

- **401**: La cabecera "Authorization" no se encuentra en la solicitud o el *token* enviado no es válido.
- **419**: El *token* recibido está caducado.

2.4.1 Registro de usuario en el sistema

En primer lugar, para poder utilizar el servicio, el interesado deberá contar con un perfil de usuario registrado en el sistema. Por ello se ha dispuesto la opción de registro mediante el uso de una petición con el método PUT a la URL: */virtualpresentation/usuario*. En el cuerpo de la petición se incluirá un objeto *JSON* con las propiedades: nombre de usuario ("nombreusuario"), contraseña ("password") y nombre ("nombre"), opcionalmente se podrá indicar también los apellidos ("apellidos") y la dirección de correo electrónico ("email"). Cuando el servidor recibe la solicitud comprueba que el nombre de usuario no está siendo utilizado en cuyo caso responderá con el código de respuesta HTTP 409. En caso contrario se procederá al registro del usuario en la base de datos, una vez finalizado, resuelve la petición informando del resultado de la operación. Este proceso se encuentra representado gráficamente en la Ilustración 2.4.

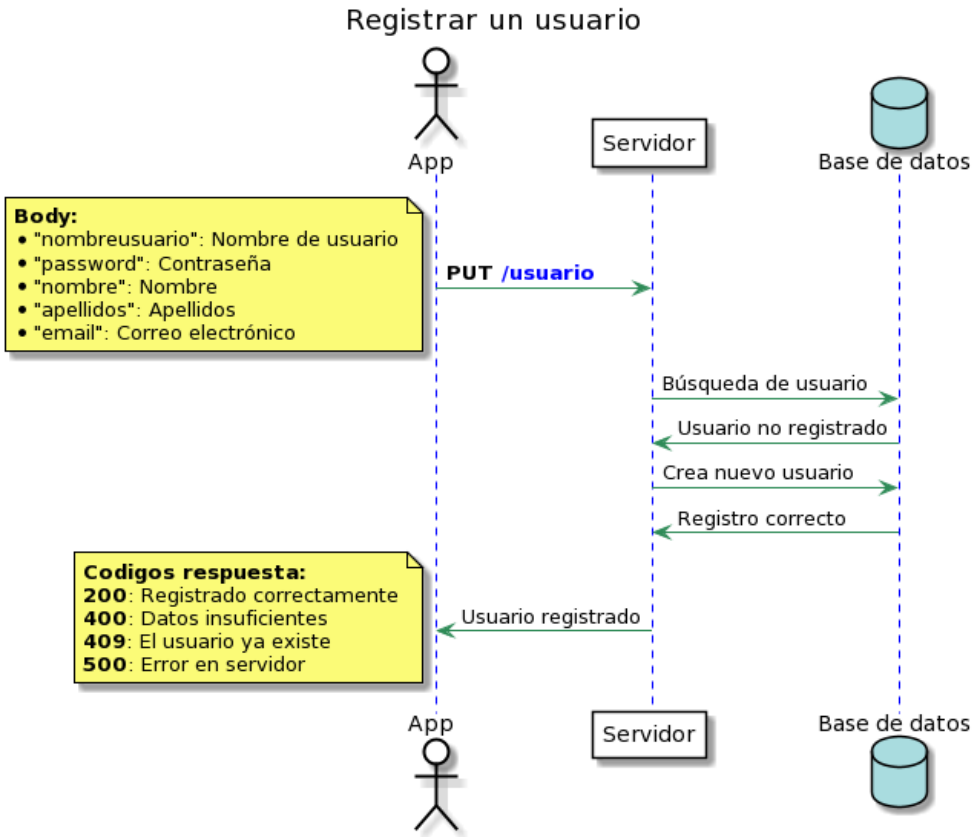

Ilustración 2.4: Diagrama registro de usuario

2.4.2 Autenticación y cierre de sesión de usuario

Como se muestra en Ilustración 2.5, con el fin de autenticarse el usuario en la aplicación, se realizará una petición con el método *POST*, donde como parámetros en el cuerpo de la misma (*body*) se enviarán el usuario ("user") y la contraseña ("password") en formato *JSON*. Una vez que el servidor recibe estos datos, consulta en la base de datos que sean correctos. En el caso de que sea correcta la autenticación, el servidor generará

un *token* encriptado y firmado con clave privada mediante *JWT* (como está definido en el apartado 2.2.5 JSON Web Token) a partir de los valores: nombre de usuario, nombre completo, id, fechas de creación y expiración. Esta cadena de caracteres, se registra en la base de datos para las posteriores consultas. Como respuesta, se envía un objeto *JSON* de usuario con las propiedades que lo identifican y el *token JWT* ("id", "nombreusuario", "nombre", "apellidos", "*token*"), además del código de estado HTTP 200. En el caso de que no sea correcto el usuario o la contraseña, enviará un código de respuesta 401 y un mensaje informando de que no son correctos los datos.

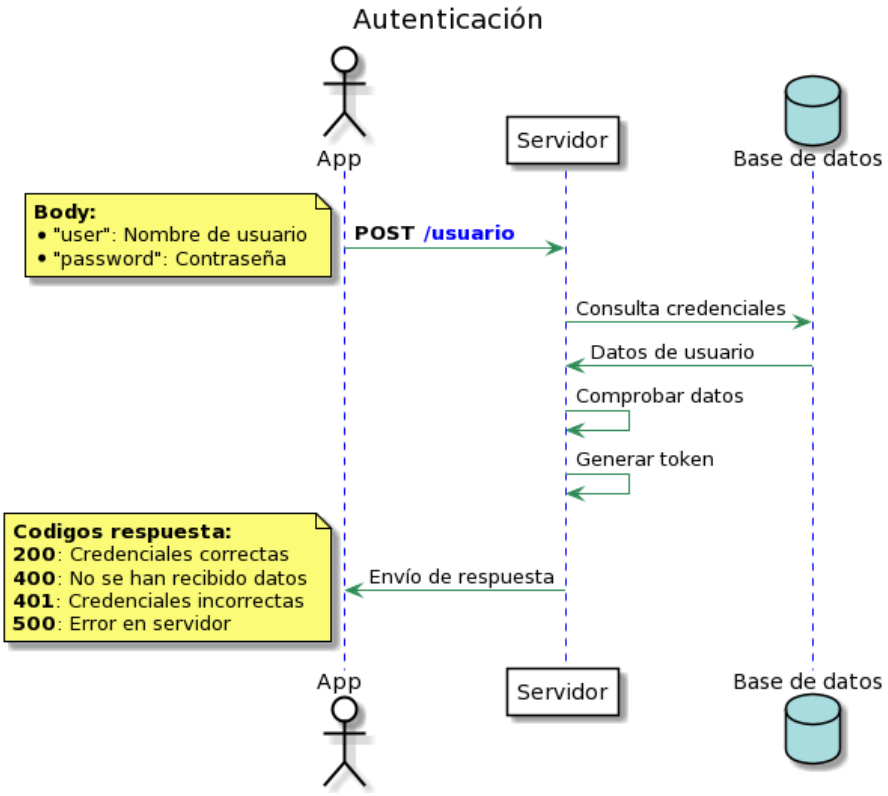

Ilustración 2.5: Diagrama autenticación de usuario

El usuario podrá cerrar sesión y eliminar su *token* de autorización realizando una llamada a la URL */usuario* mediante el método GET incluyendo como cabecera el *token* de usuario. Cuando el servidor reciba la petición, una vez validado el *token*, se procede a eliminar el *token* registrado para el usuario que ha realizado la petición. Como respuesta si no se produce ningún error, el cliente recibirá el código 200 junto con la descripción "Sesión cerrada". A continuación, en la Ilustración 2.6 se describe de forma de esquema este proceso.

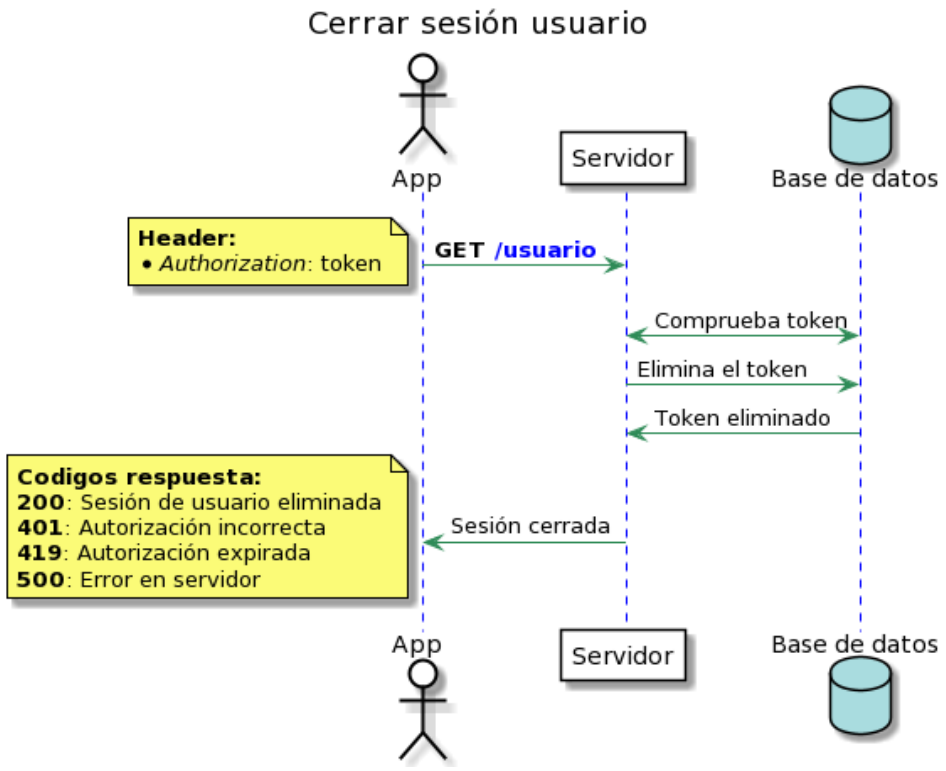

Ilustración 2.6: Diagrama cierre de sesión

2.4.3 Subir presentación

Para el uso dinámico del portal, se ha definido una ruta específica con la que se permite la subida de documentos al portal. Se trata de una petición con el método PUT, que debe incluir la cabecera de autenticación y en el cuerpo de la misma el archivo a almacenar. El usuario incluido en la URL deberá corresponder al usuario del *token*, en caso contrario no se podrá completar la petición respondiendo con el código 403.

El proceso de esta petición consta de varias partes, tal como se muestra en la Ilustración 2.7. Al recibir la solicitud, se realiza en primer lugar la comprobación del *token* de usuario y posteriormente se comprueba que se ha recibido el documento en *form-data* y además el tipo mime es “application/pdf”. A continuación, se realiza una nueva consulta a la base de datos para comprobar si la presentación ya está registrada, en caso afirmativo resolverá la petición con el código 409 notificando que ya se encuentra en el sistema. Si la presentación no está registrada, se copia el archivo recibido en el directorio de ficheros del servidor para el usuario, mediante el gestor de archivos de *NodeJS*, *fs* (*File System*). Posteriormente, se registra los datos del documento en la base de datos, siendo estos el nombre de la presentación, número de páginas y nombre de usuario que ha realizado el envío. Por último, se crea un fichero *JSON* en el mismo directorio que la presentación, con el cual se guardarán las notas que el usuario desee conservar durante la exposición. Cuando el proceso se completa correctamente sin producirse errores, como respuesta al usuario se envía el código 200 de HTTP informando que la presentación se ha almacenado.

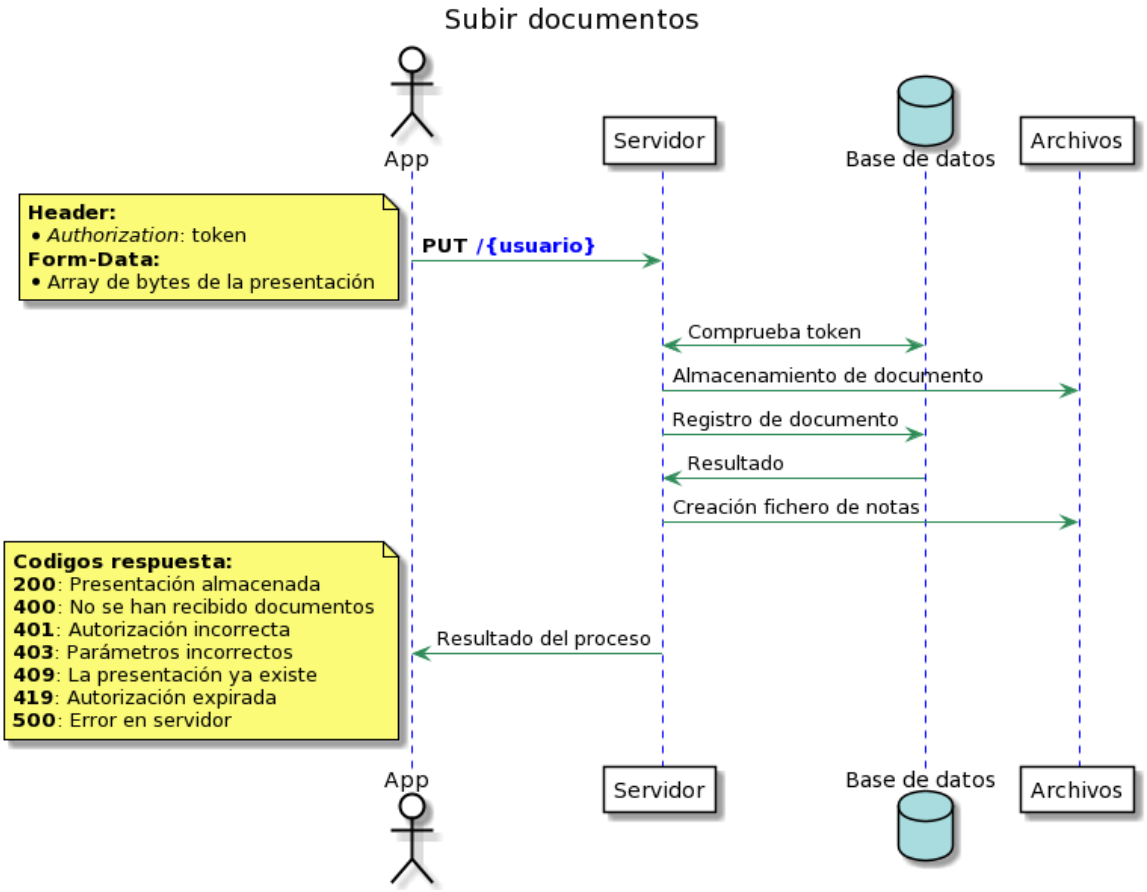

Ilustración 2.7: Diagrama subida presentación

2.4.4 Listar presentaciones y sesiones

Con el objetivo de que la aplicación, y por tanto el usuario, conozca las presentaciones que tiene registradas en el sistema, así como las sesiones que ha creado previamente se han definido dos recursos en la API para su listado. Ambas peticiones siguen un procedimiento similar, se realizan mediante el método HTTP GET y si no se produce ningún error devolverá un array con el listado. Cuando el servidor recibe la petición, realizará una consulta a la base de datos para obtener el listado correspondiente. Esta consulta y las propiedades de cada elemento cambiarán dependiendo de si se ha solicitado el listado de presentaciones o de sesiones. La lista será devuelta en un array de objetos *JSON* hacia el cliente, para el caso de las presentaciones contará con las propiedades: “idpresentacion”, “presentacion”, “paginas” y “nombreusuario”; y para las sesiones: “idsesion”, “sesion”, “presentacion”, “paginas” y “usuario”. Todo el proceso se puede observar en la Ilustración 2.8.

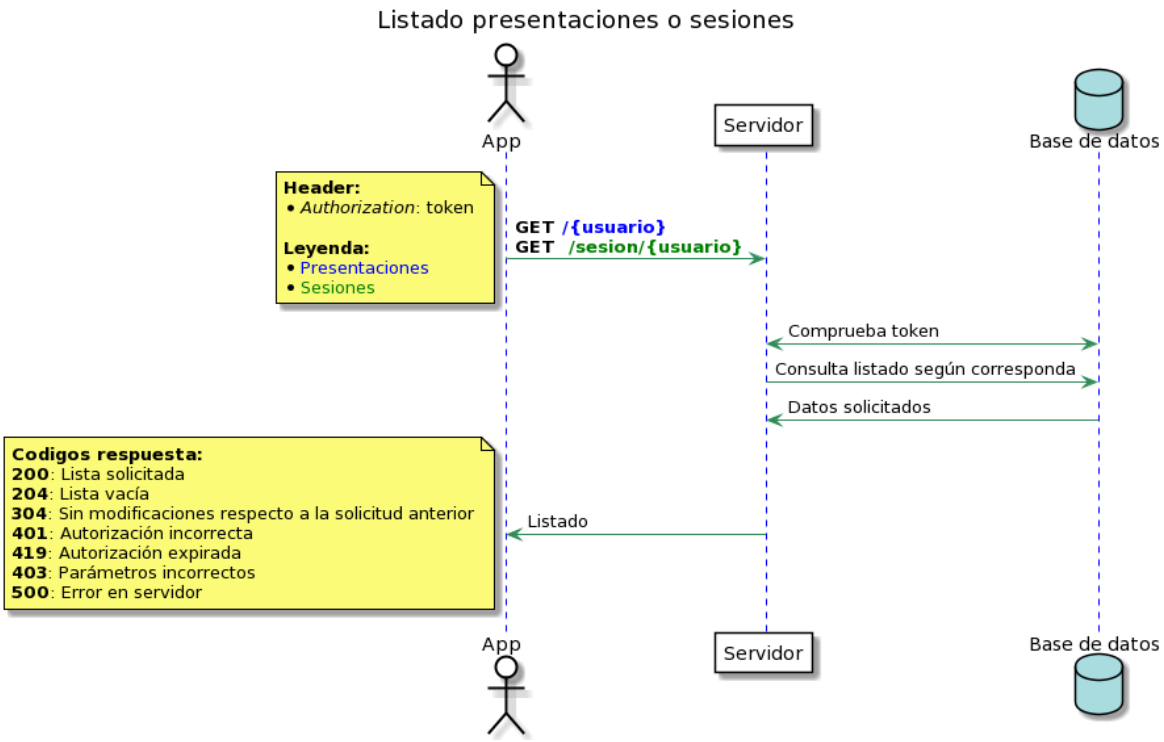

Ilustración 2.8: Diagrama listados

2.4.5 Eliminar presentación

Con el fin de eliminar una presentación, se ha dispuesto un recurso al que se accederá con el método POST y será necesario incluir en las cabeceras de la petición un nuevo elemento, “presentacion", en el que se incluirá el nombre del documento a eliminar. El procesado de la petición sigue el esquema de la Ilustración 2.9. En primer lugar, realiza una comprobación del *token*, a continuación, elimina el documento del sistema, así como el fichero de notas asociado. Por último, elimina el registro del documento en la base de datos y en consecuencia se elimina a la vez las sesiones asociadas al mismo. El cliente que realizó la petición recibirá el código 200 con el texto “Eliminado correctamente” cuando se han completado las operaciones sin errores.

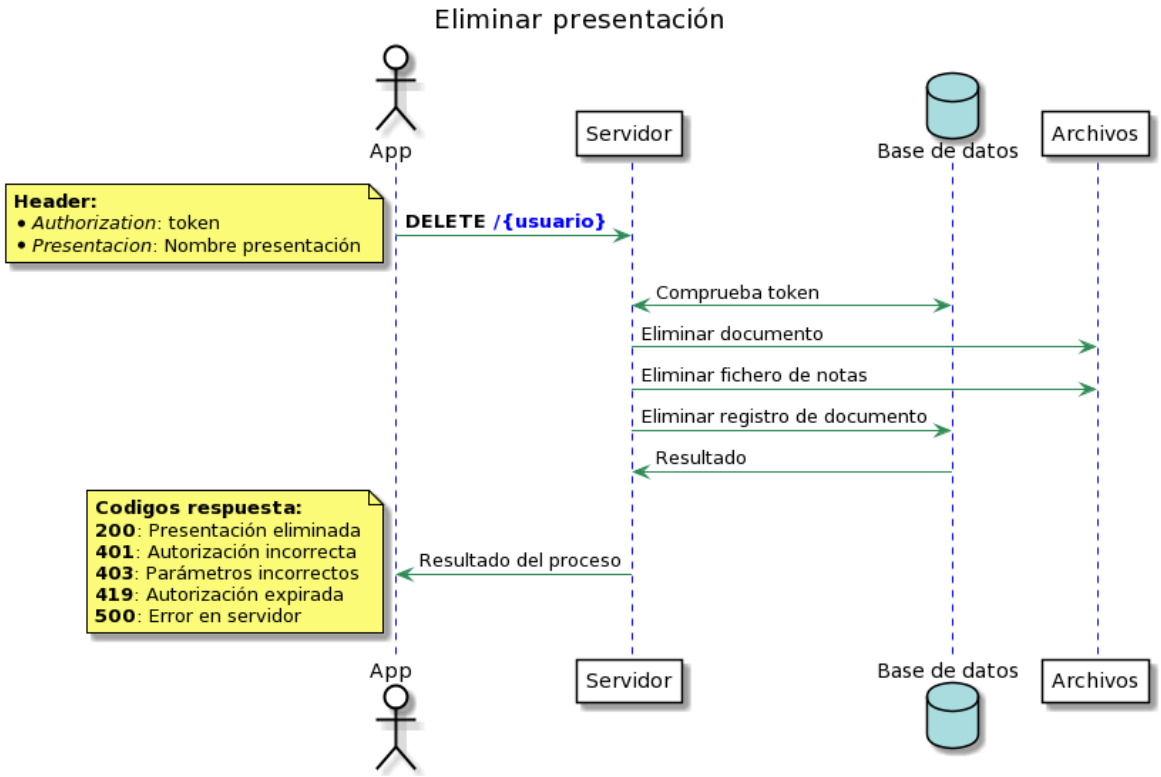

Ilustración 2.9: Diagrama eliminar documento

2.4.6 Crear y eliminar sesión

Antes de comenzar una exposición será necesario contar con una sesión, que estará determinada con el nombre, el documento a presentar y el usuario que la ha creado. Para definirla, se realizará una llamada POST al servidor, incluyendo en el cuerpo de la misma un objeto *JSON* con el nombre de la presentación ("presentación") y de la sesión ("sesion"). El proceso de la petición se describe en la Ilustración 2.10, al registrar la sesión inserta un nuevo registro, si ya existe el nombre de sesión para el usuario se actualizará el documento asociado. En ambos casos, devuelve un código 200 informando del resultado de la operación.

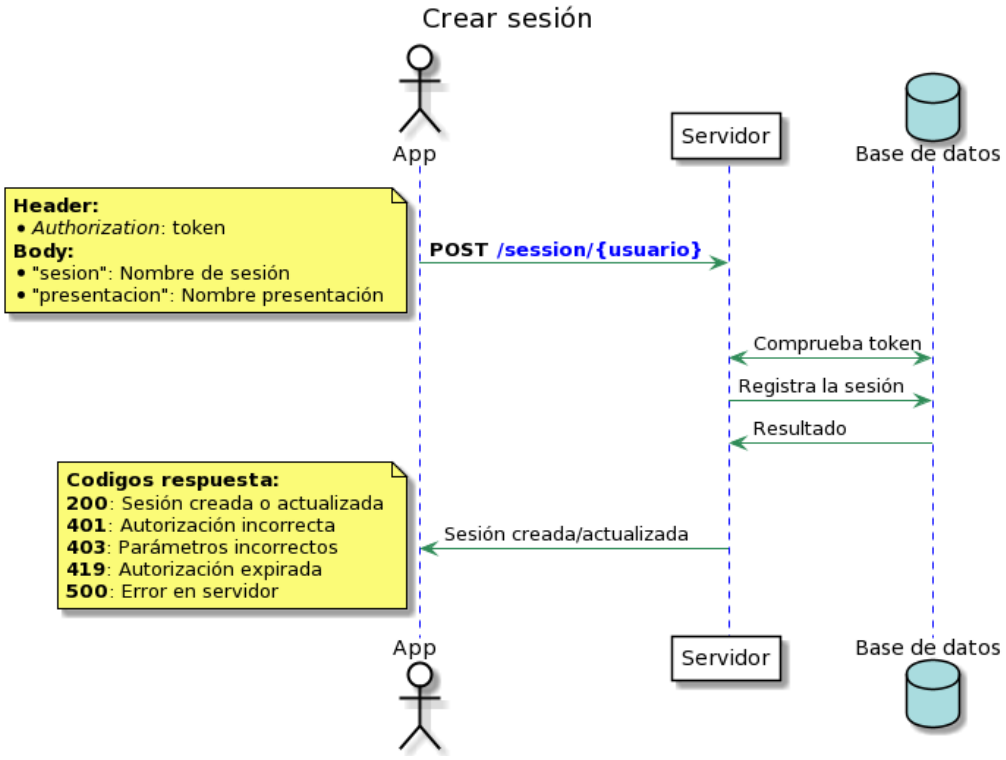

Ilustración 2.10: Diagrama creación de sesión

De forma similar, si el usuario desea eliminar la sesión, se realizará una petición tipo DELETE incluyendo en las cabeceras de la misma, el nombre de la sesión ("sesion") a eliminar. El servidor realizará una solicitud de eliminar el registro en la base de datos, si ha sido exitoso, el resultado de la operación será código 200 acompañado del texto que informa del hecho.

Para ambas operaciones, en la solicitud, el cliente incluirá la cabecera de autorización con el *token* de sesión de usuario, de forma que el servidor pueda comprobar que sea válido.

2.4.7 Portal

Por último, el presentador de la exposición, deberá acceder mediante un navegador al portal, cuya dirección URL está formada por el nombre de usuario seguido del nombre de la sesión, por ejemplo, para el usuario "sergio" y la sesión "mi-sesion", la URL quedaría https://localhost/virtualpresentation/sergio/mi-sesion. La respuesta será de contenido HTML, permitiendo diferenciarse dos vistas, una para mostrar error y otra para la sesión correcta. El servidor comprobará la existencia del par nombre de sesión y usuario en la base de datos, si no obtiene resultados de la consulta devolverá la vista indicando que no existe o a expirado, como se muestra en la Ilustración 2.11.

Ilustración 2.11: Vista sesión caducada

Si la sesión es correcta, el servidor generará un código QR con la información de la misma (Ilustración 2.12), además de un código aleatorio de 5 caracteres necesario para asegurar la unicidad de la sesión. Posteriormente, se realizan una serie de peticiones entre el navegador y el servidor solicitando los ficheros necesarios, así como la presentación correspondiente.

Ilustración 2.12: Vista portal QR

Ambos casos de vistas se muestra su proceso de comunicaciones en la Ilustración 2.13.

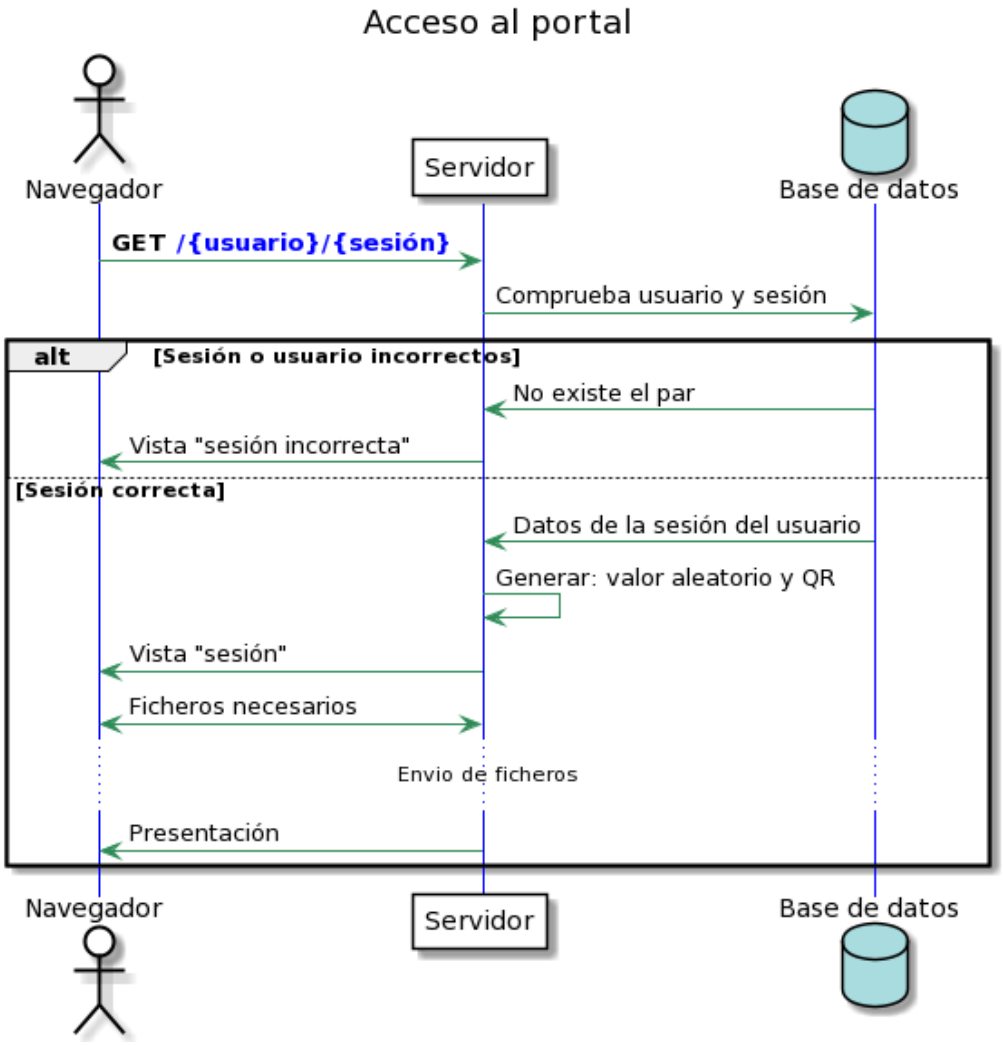

Ilustración 2.13: Diagrama portal

2.4.8 Protocolo de comunicación del sistema

Durante la exposición de una presentación, se lleva a cabo un intercambio de mensajes entre la aplicación móvil y el portal para el control de la misma, los cuales son enviados mediante el protocolo *WebSocket*. Estos mensajes se describen en este protocolo de comunicación, tienen formato de objetos *JSON* y permiten el envío de comandos, notas de texto y mensajes informativos. En la comunicación, el servidor actuará de intermediario, recibiendo y redirigiendo el contenido al otro extremo de la misma. Hay definidos cuatro objetos de mensajes con diferentes propiedades, uno para el envío de comandos y el resto para el manejo de notas.

El servidor *WebSocket* cuenta con un canal común (denominado "virtualPresentations") en el que se reciben todos los mensajes que se intercambian en la comunicación. Posteriormente, a partir de la propiedad "sesion" del objeto mensaje, se

enviará al canal con el nombre de la sesión y será recibido e interpretado por la aplicación o por el portal.

Para comenzar la exposición, una vez se ha establecido la conexión *WebSocket* en ambos extremos, en primer lugar, se enviará desde la aplicación un mensaje incluyendo el comando “OK”. Cuando el portal web recibe dicho mensaje, dejará de mostrar el código QR, expondrá la presentación y por último enviará un mensaje de confirmación a la aplicación indicando el número de página en la que se encuentra. A continuación, se intercambiarán mensajes entre ambos extremos para el control de la presentación. Para finalizar la exposición, la aplicación enviará el comando “FIN” lo que ocasiona la recarga del portal volviendo a generar un código QR o indicando que la sesión ha finalizado. Este proceso se describe gráficamente en Ilustración 2.14.

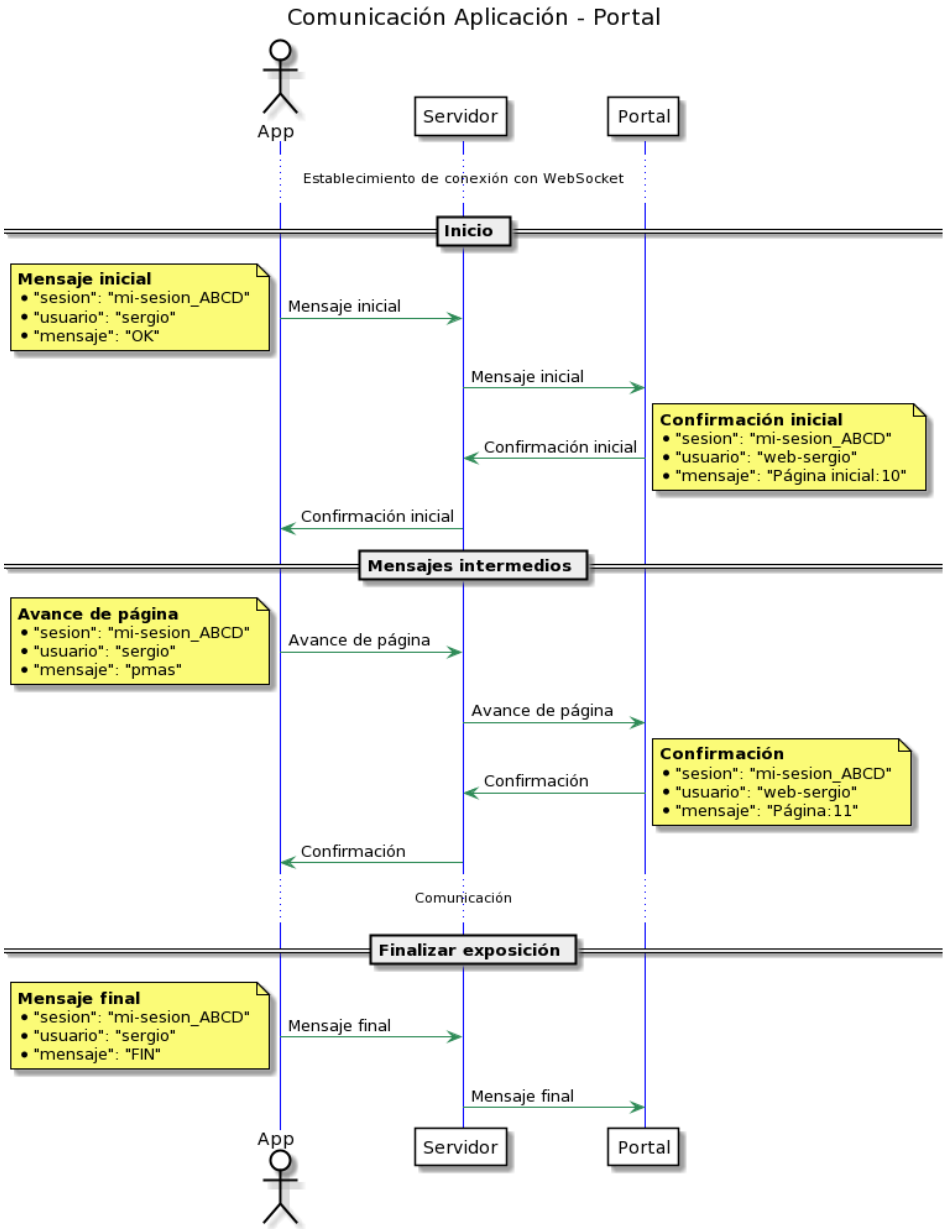

Ilustración 2.14: Ejemplo comunicación aplicación-portal

La estructura de los mensajes de comandos se utiliza para el envío general de comandos desde la aplicación, así como los mensajes enviados desde el portal a la aplicación. Este tipo de mensajes, se trata de un objeto *JSON* formado por tres propiedades, como se detalla en la Tabla 2.4.

Tabla 2.4: Estructura mensajes de comandos, websocket

Propiedad	Valor	Descripción
sesión	NOMBRE_SESION	Nombre del canal de escucha de la sesión
usuario	USUARIO	Nombre de usuario que envía el mensaje
mensaje	COMANDO	Mensaje para realizar alguna acción en el sistema

El envío de notas se realizará mediante el objeto *JSON* definido para este fin, de uso único desde la app hacia el servidor y de éste al portal web, e incluye las propiedades que se muestran en la Tabla 2.5

Tabla 2.5: Estructura mensajes de notas, websocket

Propiedad	Valor	Descripción
sesión	NOMBRE_SESION	Nombre del canal de escucha de la sesión
usuario	USUARIO	Nombre de usuario que envía el mensaje
fijar	BOOLEAN	Determina si la nota permanecerá a lo largo del uso de la presentación
nota	NOTA	Texto a mostrar en la sección de notas del portal
pagina	DIGIT	Número de página en el que se encuentra la presentación en el momento del envío de la nota

Para enviar las notas que están marcadas como fijas (el valor de la propiedad “fijar” con el valor “true”) hacia el portal, una vez procesadas por el servidor, éste añadirá dos propiedades más al objeto *JSON* con el fin de poder identificarlas y representarlas correctamente en el portal. Estas propiedades se describen en la Tabla 2.6.

Tabla 2.6: Propiedades adicionales mensajes notas

Propiedad	Valor	Descripción
id	DIGIT	Identificador de la nota, numérico definido automáticamente
fecha	FECHA	Fecha en la que se envía la nota

Por último, para poder eliminar una nota del fichero correspondiente de la presentación, se ha definido un objeto *JSON* con las propiedades que se muestran a continuación, Tabla 2.7. Su uso está limitado al portal, de forma que será enviado desde el mismo hacia el servidor sin recibir respuesta ni implicar a la aplicación.

Tabla 2.7: Estructura mensaje eliminar nota específica

Propiedad	Valor	Descripción
sesión	NOMBRE_SESION	Nombre del canal de escucha de la sesión
usuario	USUARIO	Nombre de usuario que envía el mensaje
eliminar	DIGIT	Identificador de la nota

Como tanto la aplicación como el portal web reciben los mensajes en el mismo canal, el específico de la sesión, para diferenciar si un mensaje es procedente de la app o el portal, el valor <USUARIO> de los mensajes del portal será el nombre del usuario de la sesión precedido del texto “web-”.

Los valores adoptados por cada una de las propiedades deben seguir la estructura que se detalla a continuación en formato ABNF [22]:

```
NOMBRE_SESION = 1 * ALPHANUMERIC "_" 5 * 5 ALPHANUMERIC ;
USUARIO = 1*ALPHANUMERIC ;
BOOLEAN = "true" / "false" ;
COMANDO = 1 * ALPHANUMERIC ;
FECHA = DD "/" MM "/" AAAA ;
ALPHANUMERIC = ALPHA / DIGIT ; puede contener valores alfabéticos
                             ; y/o numéricos
DD = 2*2 DIGIT ;
MM = 2*2 DIGIT ;
AAAA = 4*4 DIGIT ;
```

Por último, se ha definido un listado de comandos de texto, estarán incluidos en la propiedad “mensaje” de los objetos JSON que se comunican entre portal y la aplicación, definidos en la Tabla 2.8 y serán utilizados para el valor <COMANDO>.

Tabla 2.8: Comandos websocket

Comando	Descripción
OK	Comienzo de sesión
FIN	Finaliza la sesión desde la aplicación
Página inicial:num	Página inicial en la presentación. "num": número de página
Página:num	Página actual (num)
pmas	Página siguiente, si no está en la última
pmenos	Página anterior, si no está en la primera
pnum-num	Página específica, donde "num" representa el número de la página
zmas	Aumenta el zoom
zmenos	Disminuye zoom
zinicial	Vuelve al zoom predeterminado y centra la presentación
subir	Mueve la diapositiva en la dirección indicada por el comando
bajar	
izquierda	
derecha	
muestraNotas	Muestra la lista de notas
eliminaNotas	Elimina todas las notas

2.5 Aplicación

2.5.1 Aspectos generales

Durante el uso de la aplicación se realizan intercambios de datos entre ésta y el servidor, para tener un acceso a los mismos más ordenado en la programación de la aplicación, se han definido una serie de clases que modelan los objetos de datos como son el usuario, las sesiones y las presentaciones. Cada clase tiene los métodos apropiados para establecer u obtener los valores concretos de cada propiedad, así como algunos métodos para obtener los datos a partir de un objeto *JSON*.

La clase "User" para los datos de usuario cuenta con las siguientes propiedades:

- **"id":** identificador numérico único del usuario.
- **"nombreusuario":** apodo (nick).
- **"nombre":** nombre real.
- **"apellidos":** apellidos.
- **"token":** *token* generado en la autenticación.

- **“listaPresentaciones”:** listado de las presentaciones que tiene registradas en el sistema.

Para las presentaciones se define la clase “Presentations” cuyas propiedades son:

- **“idpresentacion”:** identificador numérico de la presentación.
- **“presentacion":** nombre completo de la presentación.
- **“paginas”:** número total de páginas del documento.

Por último, la clase “Session” en la que se describe completamente una sesión:

- **“nombreUsuario”:** nombre de usuario propietario de la sesión.
- **“nombreSesion”:** nombre identificativo de la sesión.
- **“presentacion”:** nombre de la presentación a exponer.
- **“paginas”:** número total de páginas de la presentación.
- **“paginalnicio”:** página inicial por la que comenzará la exposición.
- **“código”:** cadena de 5 caracteres generado antes del código QR de la sesión.

2.5.2 Pantalla de bienvenida

En primer lugar, al iniciar la aplicación, se muestra una pantalla de bienvenida denominada en inglés *Splash Screen*, como se representa en la Ilustración 2.15. Esta actividad de la clase *Activity*, tiene una duración de 2 segundos y muestra los datos principales del presente trabajo, como son el logo de la Escuela Politécnica Superior de Linares, nombre del trabajo, autor y tutor del mismo. Al mismo tiempo que se muestra esta pantalla, se realiza una lectura de los datos almacenados en la memoria, si están almacenados los datos de la última autenticación correcta y el usuario marcó la opción para recordar las credenciales, se iniciará la actividad principal (apartado 2.5.4), en caso contrario comenzará la actividad para la autenticación (2.5.3).

Ilustración 2.15: Pantalla bienvenida

El almacenamiento en memoria se realiza mediante *SharedPreferences*, para el caso de los datos del usuario, las preferencias compartidas están identificadas con el nombre "default" y en ellas se registran las propiedades del usuario: "id", "nombreusuario", "nombre", "apellidos", "permanente" (de tipo booleano, registra si en la autenticación previa, el usuario marcó la opción para recordar las credenciales) y "*token"* (para las peticiones hacia la API).

2.5.3 Autenticación

A fin de acceder al uso de la aplicación y del sistema, es necesario autenticarse frente al servidor mediante esta actividad (de la clase *Activity*), que se muestra en la Ilustración 2.16. El usuario deberá indicar su nombre de usuario (*nick*) y su contraseña, en los cuadros de entrada de texto correspondientes. Además, el usuario podrá marcar la casilla de verificación "Recuérdame" de forma que la siguiente vez que inicie la aplicación no sea necesario volver a introducir las credenciales y podrá acceder directamente a la actividad principal de la aplicación.

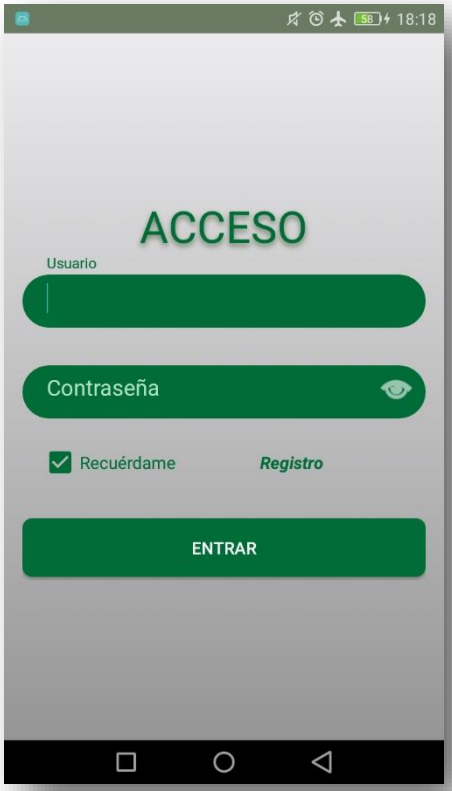

Ilustración 2.16: Autenticación

Al pulsar sobre el botón “Acceder” se realiza la petición de autenticación enviando las credenciales hacia el servidor, esta petición está descrita en el Apartado 2.4.2. La contraseña introducida por el usuario será transformada a una cadena hash SHA256, de forma previa al envío. Cuando se recibe la respuesta del servidor indicando mediante el código de respuesta HTTP 200 que son correctas las credenciales, a partir del cuerpo (*body*) de la respuesta se almacenará cada una de las propiedades del objeto *JSON* recibido de la clase *User* (“id”, “nombreusuario”, “nombre”, “apellidos”, “*token*”) en las preferencias compartidas (*SharedPreferences*) “default” para su posterior uso. También, se registra la propiedad “permanente” la cual, mediante un valor booleano, se registra si ha sido marcada la opción “Recuérdame”. Por su parte, el *token* almacenado será consultado de forma previa a realizar cualquiera de las peticiones al servidor. Una vez almacenados los datos, se lanzará la siguiente *Activity*, actividad principal (apartado 2.5.4).

El ciclo de uso de esta actividad se representa de forma esquemática en la Ilustración 2.17.

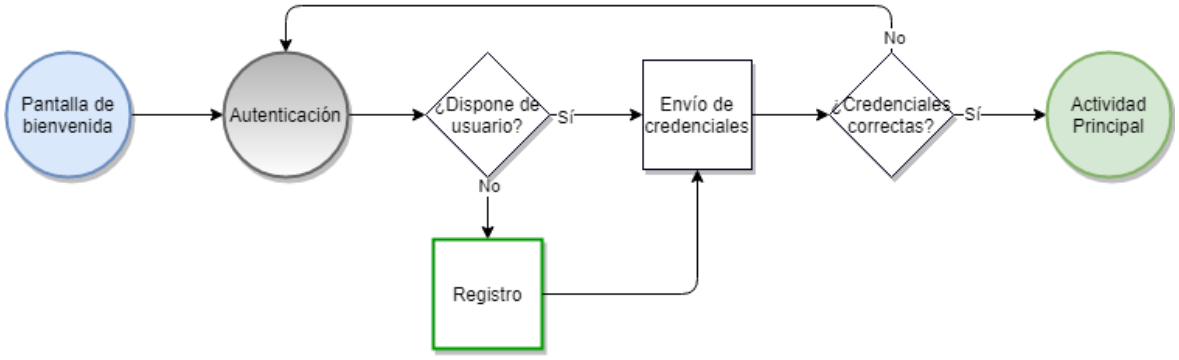

Ilustración 2.17: Ciclo autenticación

Adicionalmente a la autenticación, desde esta actividad un usuario podrá registrarse en el sistema mediante un formulario que es accesible pulsando el texto “Registro”.

2.5.3.1 Registro

El formulario de registro se encuentra en una ventana emergente superpuesta a la interfaz de autenticación, como se observa en la Ilustración 2.18. Esta nueva ventana de estilo modal, se implementa mediante la subclase *AlertDialog* (utilizadas para mostrar alertas o solicitar nueva información), en ella se establece la vista creada para este fin, el formulario de registro.

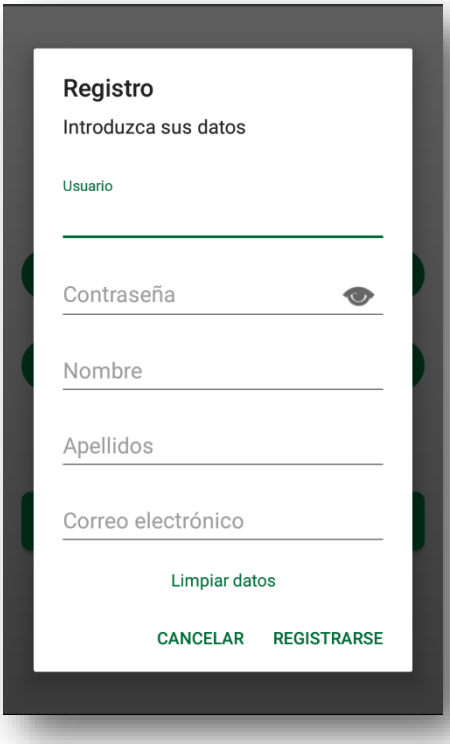

Ilustración 2.18: Registro

En el formulario, el usuario deberá incluir de forma obligatoria y un tamaño mínimo de dos caracteres, el nombre de usuario, la contraseña y el nombre. De forma opcional podrá especificar sus apellidos y dirección de correo electrónico. Los datos introducidos por el usuario se podrán eliminar y volver a introducirlos de forma correcta pulsando sobre el texto “Limpiar datos”. Una vez completado el formulario, al pulsar sobre “Registro” se realizará una petición PUT a la API como se desarrolla en el Apartado 2.4.1, enviando los datos introducidos por el usuario transformando la contraseña a un hash SHA256. Una vez completado el registro de forma correcta, el formulario se cierra manteniendo el *nick* en el campo propio de las credenciales para proceder a la autenticación.

2.5.4 Actividad principal

En la actividad principal, el usuario podrá preparar lo necesario para realizar la exposición de una presentación, como son la declaración de las sesiones, publicar y eliminar documentos. Esta *Activity* cuenta con un menú desplegable en el que se muestran los ítems de las secciones, un contenedor donde se incluirán las secciones y un menú de opciones. El contenedor se podrá completar con una de las cinco secciones, de la clase *Fragment*, en las cuales se podrán realizar las acciones listadas, además de la lectura del código QR identificativo de la sesión y una breve descripción de la aplicación.

El menú desplegable que permanece oculto, el cual, se puede acceder pulsando sobre el botón de la cabecera en el lateral izquierdo o bien, realizando el toque de izquierda a derecha en cualquier sitio de la pantalla. Está formado por dos partes, la superior en la cual se muestran los datos del usuario como son el nombre y el nick, y la parte inferior, donde se listan y son accesibles los diferentes fragmentos de la actividad, como se muestra en la Ilustración 2.19. Las secciones disponibles en la presente actividad son:

1. Crear sesión.
2. Lectura de código QR.
3. Subir presentación.
4. Eliminar presentación.
5. Información del proyecto.

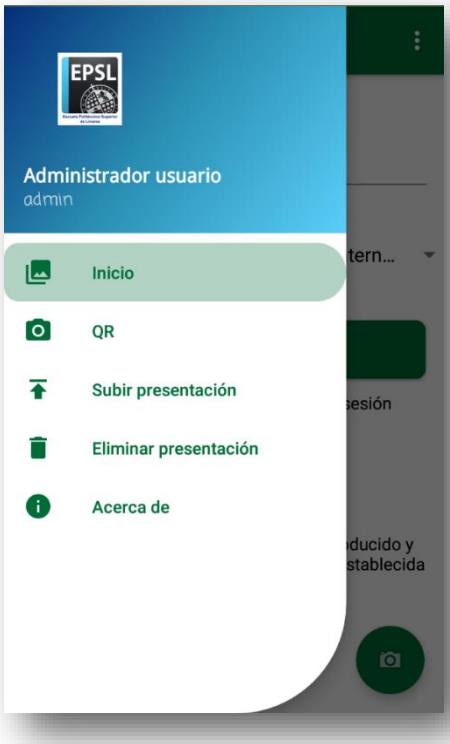

Ilustración 2.19: Menú

En el menú de opciones, situado en la parte superior derecha, con un botón para cerrar sesión, como se muestra en la Ilustración 2.20. Esta opción, permite realizar una petición a la API para anular la validez del *token* generado en la autenticación, tal como se describió en el Apartado 2.4.2. Además, elimina los datos de la memoria de la aplicación, que el usuario ha generado durante el uso de la misma, eliminando en las preferencias compartidas los datos del usuario almacenados en "default", cada una de las sesiones creadas y el listado "all_Sessions" donde se almacena el nombre de todas ellas. Por último, una vez finalizado lo anterior, la aplicación regresa a la actividad de autenticación (A*ctivity* descrita en 2.5.3).

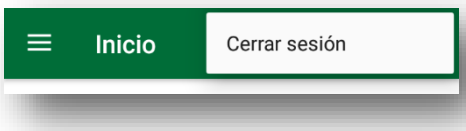

Ilustración 2.20: Menú cierre de sesión

En esta actividad, excepto cuando se muestran los fragmentos de lectura del código QR y de "acerca de", se muestra un botón flotante con icono de una cámara, como se muestra en la Ilustración 2.21, situado en la zona inferior derecha, que facilita el acceso directo para escanear el código QR de la sesión.

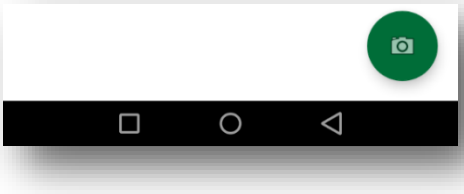

Ilustración 2.21: Botón flotante lectura QR

En primer lugar, al iniciar esta actividad, se realiza una petición hacia el servidor para obtener el listado de las sesiones que se han creado previamente (apartado 2.4.4), permitiendo recuperar las sesiones que el usuario ha creado anteriormente y que aún están activas.

Durante el uso de la aplicación una vez autenticado el usuario, se realizarán diversas solicitudes a la API, siendo estas las peticiones para obtener los listados de sesiones y presentaciones, almacenar y eliminar un documento, crear y eliminar una sesión. Cuando una solicitud sea resuelta con el código de respuesta HTTP 419, indicando que ha expirado el *token* (*JWT*) de usuario, o el código HTTP 401, indicando que es incorrecto, la aplicación mostrará una alerta solicitando al usuario la contraseña, como se muestra en la Ilustración 2.22. Esta alerta, creada con la subclase *AlertDialog* y una vista con un cuadro de texto para la introducción de la contraseña, ofrece dos opciones: cerrar la sesión, o bien enviar de nuevo la contraseña. Con la primera opción, la aplicación elimina los datos del usuario almacenados en memoria (todas las preferencias compartidas) e inicia la actividad de autenticación (*Activity* descrita en 2.5.3). En la segunda opción, será necesario escribir la contraseña en el cuadro dispuesto para ello y se enviará al servidor mediante la petición de autenticación (apartado 2.4.2). Una vez ha procesado el servidor las credenciales y genere la respuesta, se actualizan las preferencias compartidas (*SharedPreferences*) "default" con el nuevo *token* de usuario recibido.

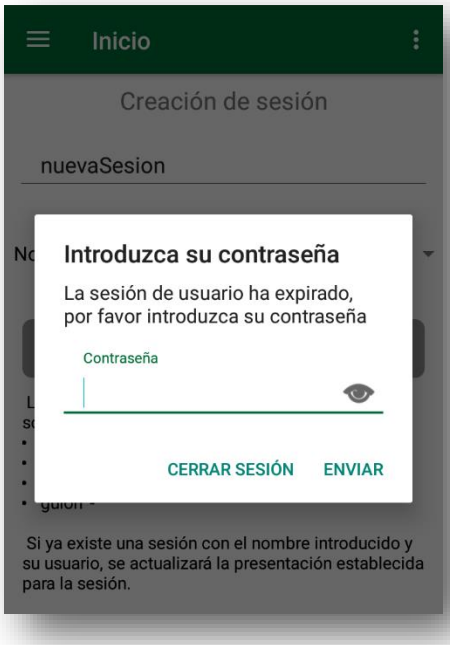

Ilustración 2.22: Alerta contraseña

Al iniciar esta actividad se realizará una petición al servidor solicitando el listado de sesiones (Apartado 2.4.4). A partir del *array* de sesiones recibido se comprobará si está en memoria cada una de las sesiones en las preferencias compartidas, identificadas por el nombre de la sesión. Cuando ya existe la sesión en memoria, se comprueba que la presentación sea la misma, en caso contrario actualiza la sesión en la memoria. En caso de que no esté almacenada, se creará una nueva preferencia compartida con el nombre de la sesión y los datos de la misma. De esta forma se podrá proceder a la lectura del código QR de una sesión, aunque previamente no haya sido creada en el mismo dispositivo o no se encuentre almacenada en la memoria de la aplicación.

De forma general, esta actividad se utilizará siguiendo el modelo esquemático de la Ilustración 2.23.

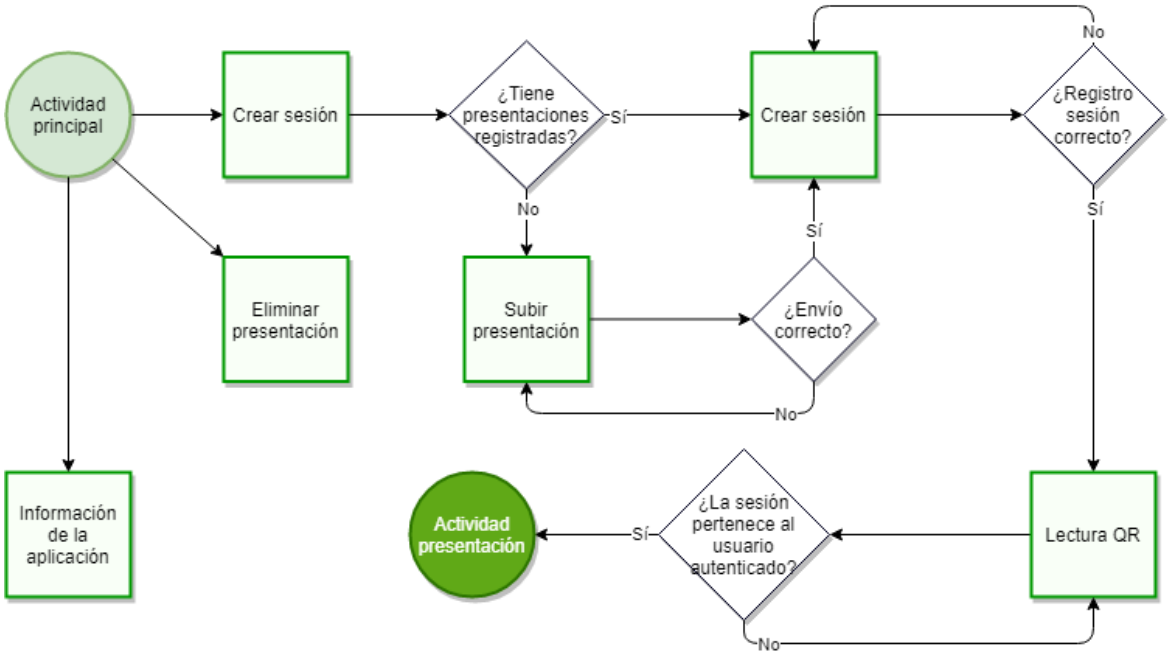

Ilustración 2.23: Ciclo actividad principal

2.5.4.1 Crear sesión

Una vez que el usuario se ha autenticado correctamente, la primera sección que puede visualizar de la actividad principal es la creación de una sesión, Ilustración 2.24. Cuando se inicia este fragmento (de clase *Fragment*), se realiza una petición con el método GET al servidor solicitando la lista de las presentaciones almacenadas (Apartado 2.4.4). Los datos recibidos son utilizados para completar la lista desplegable con la que el usuario podrá seleccionar el documento para la sesión.

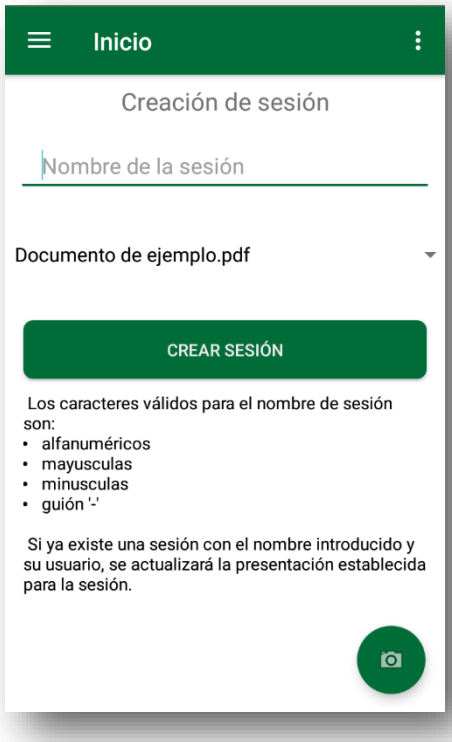

Ilustración 2.24: Vista crear de sesión

Cuando el usuario ha introducido el nombre de la sesión, así como seleccionado el documento deseado, mediante el botón “Crear sesión”, se enviarán los datos en formato *JSON* a la API de la forma que se describe en el Apartado 2.4.6, dónde se validan y registra la sesión. Si los datos que se han enviado son correctos, el servidor generará una respuesta informado si se ha creado una nueva o modificado una existente. Finalmente, cuando la respuesta ha sido procesada, el texto mostrado bajo el botón de enviar, cambiará indicando la dirección URL de la sesión que acaba de crearse y, además, se almacenan los datos de la sesión (objeto de clase *Session*) en la memoria de la aplicación, mediante las preferencias compartidas identificados con el nombre de la sesión, para su posterior uso en la lectura del código QR.

2.5.4.2 Lectura de código QR

El fragmento de lectura del código QR hace uso de la cámara del dispositivo, por ello es necesario que el usuario conceda el permiso de acceso a la cámara. Al iniciarse, mediante la clase *ActivityCompat* se comprueba si tiene el permiso concedido de acceso a la cámara y en caso contrario se muestra una solicitud de acceso que el usuario podrá confirmar, como se muestra en la Ilustración 2.25. En el archivo de manifiesto “AndroidManifest.xml” donde se configuran los componentes de la aplicación, se indicará además el requisito de acceso a la cámara para poder solicitar el permiso posteriormente.

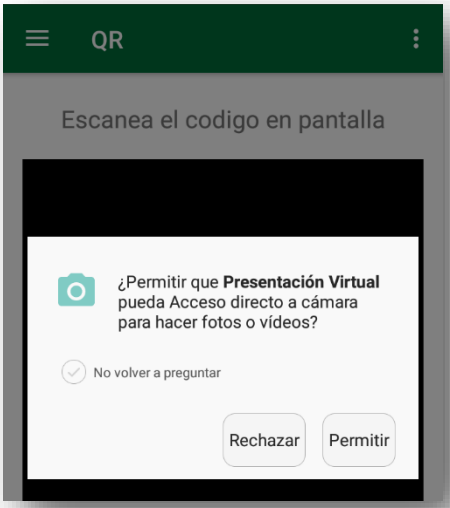

Ilustración 2.25: Solicitud acceso a cámara

Este fragmento está compuesto únicamente por un marco en el que se puede visualizar la cámara para escanear el código QR, como se puede observar en Ilustración 2.26. Este código será mostrado en el navegador una vez el usuario ha accedido a la dirección URL de la sesión. Cuando la aplicación detecta un código QR legible, convierte la cadena de texto que ha extraído del mismo a un objeto *JSON* a partir del cual se crea un objeto de la clase *Session*. Posteriormente se comprueba en las preferencias compartidas el nombre de la sesión y si existe, se comprueba que el título de la presentación coincida con el leído en el código QR, en este caso se iniciará la actividad de control remoto de la presentación (apartado 2.5.5).

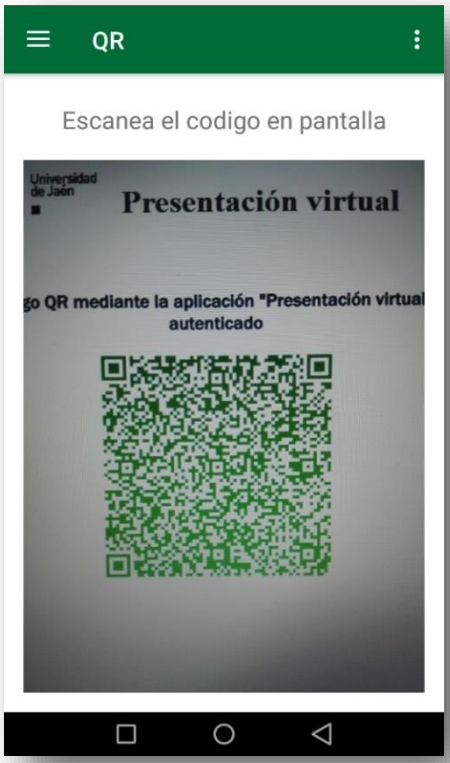

Ilustración 2.26: Lectura código QR

2.5.4.3 Subir presentación

En el apartado de subir presentación (de clase *Fragment*), de forma similar como sucede en la sección anterior, es necesario contar con los permisos, en este caso de acceso a documentos almacenados en el dispositivo, la primera vez que se ingrese en la sección, solicitará los permisos, como se muestra en la Ilustración 2.27.

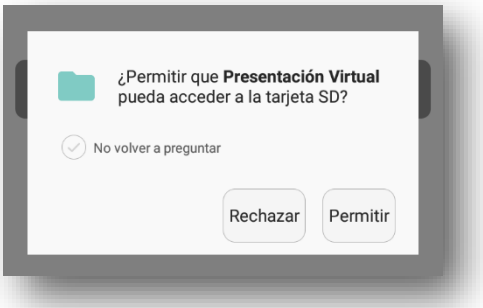

Ilustración 2.27: Solicitud acceso a memoria del dispositivo

En el fragmento se disponen de dos botones, el primero a partir del cual se podrá elegir el documento y el segundo enviará el documento elegido al servidor, como se muestra en la Ilustración 2.28. Para seleccionar el documento se utiliza el administrador de

ficheros propio del dispositivo, permitiendo seleccionar el documento desde la memoria del dispositivo o almacenamiento de archivos online como Google Drive. Para seleccionar un documento deberá ser en formato PDF ya que está restringido al tipo mime "application/pdf", en caso contrario no serán visibles en el momento de su selección. Por último, una vez seleccionado el documento, al pulsar el botón enviar, se realizará la petición PUT a la API enviándolo, como se explica en el Apartado 2.4.3

Ilustración 2.28: Vista subir presentación

El proceso de envío del documento al servidor se informará mediante una notificación en el dispositivo, generada de forma automática, por la librería *Upload Service* [9]. En primer lugar, durante la subida, se mostrará una barra de progreso con el tanto por ciento de datos enviados al servidor. Una vez completada la carga y el servidor haya generado una respuesta, la notificación cambiará informando si se ha enviado correctamente la presentación o por el contrario si se ha producido algún error y no ha podido registrarse.

2.5.4.4 Eliminar presentación

De forma similar a crear una sesión, esta sección (de clase *Fragment*) contiene el listado de las presentaciones y un botón para el envío de la petición, como se muestra en la Ilustración 2.29. La lista seleccionable se completa realizando nuevamente una petición al servidor para solicitar el array de presentaciones registradas (apartado 2.4.4).

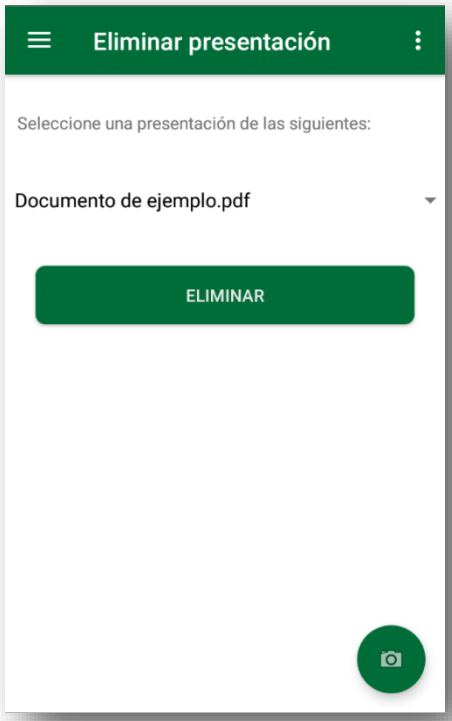

Ilustración 2.29: Vista eliminar documento

Para eliminar la presentación seleccionada, una vez pulsado el botón "Eliminar" se muestra una ventana emergente (*AlertDialog*) para que el usuario confirme si realmente desea eliminar el documento, con el aspecto que se puede observar en la Ilustración 2.30. Cuando se ha confirmado el borrado por parte del usuario, se realiza la petición a la API con el método HTTP DELETE, enviando el nombre de la presentación en las cabeceras, como se describe en el Apartado 2.4.5.

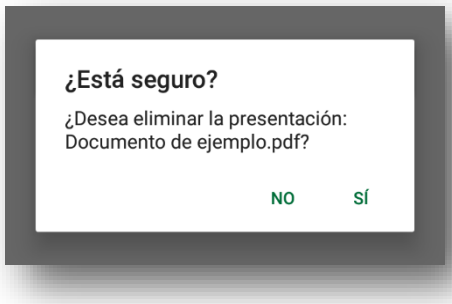

Ilustración 2.30: Confirmación eliminar documento

2.5.4.5 *Acerca de*

Por último, se dispone de un fragmento (clase *Fragment*) que resume brevemente las utilidades de la aplicación e información principal del trabajo, como es el título, autor, tutor y una breve descripción. Esta sección no tiene funcionalidad más allá de un resumen, tiene el aspecto que se muestra en la Ilustración *2.31*.

Ilustración 2.31: Acerca de

2.5.5 *Presentación*

Continuando con el ciclo de la aplicación, de forma consecutiva a la lectura correcta del código QR de la sesión, la aplicación dará paso a la actividad de presentación (clase *Activity*). Para iniciar la actividad será necesario indicar el nombre de la sesión y el código aleatorio generado por el servidor al acceder a la dirección URL, estos datos se especifican durante el cambio de actividad. Una vez que se inicia la actividad, se crea una instancia de *SocketIO*, si el servidor está disponible y activo, se realizará la conexión y se envía un primer mensaje de confirmación para comenzar la exposición. En el caso de que el servidor se encuentre activo, pero no se ha podido realizar correctamente el inicio de exposición, se mostrará un mensaje para volver a intentar realizar la conexión.

La actividad tiene un ciclo de uso sencillo, como se muestra en Ilustración 2.32.

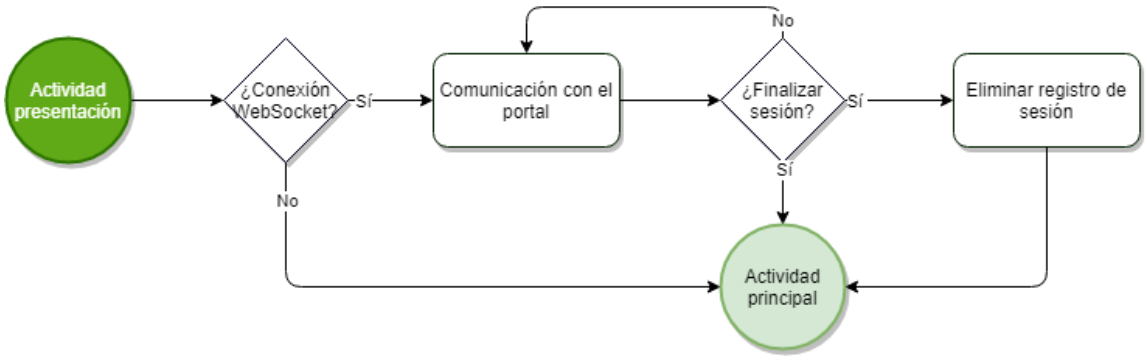

Ilustración 2.32: Ciclo actividad presentación

La interfaz de usuario de esta actividad, que actúa y simula el control remoto de la presentación, está dividida en tres secciones:

1. **Cambio de páginas:** botones de avance y retroceso e ir a una página específica.
2. **Controles de visualización:** ajustes de zoom y controles de movimiento del documento.
3. **Envío de notas:** un cuadro de texto donde introducir la nota, casilla de verificación para mantener la nota en la próxima exposición y botones de enviar, eliminar notas y mostrar la sección de las notas en el navegador.

Después de la recepción del primer mensaje enviado desde el portal, que corresponde al número de página en la que se encuentra, se habilitará la interfaz de usuario, como se representa en la Ilustración 2.33.

Ilustración 2.33: Control remoto presentación

Para concluir la exposición, desde la aplicación, deberá pulsarse el botón atrás del dispositivo Android, y a continuación, se mostrará un cuadro de diálogo, como se ilustra en la Ilustración 2.34. En él se solicita confirmación al usuario con las opciones: salir, salir y eliminar la sesión, y cancelar.

Ilustración 2.34: Confirmación salir de la exposición

Si el usuario desea mantener la sesión habilitada, pulsará “salir”, con ello envía un mensaje para finalizar la exposición en el portal. También, se almacena en la memoria de la aplicación, el número de página en la que se encuentra, de esta forma, cuando el usuario

vuelva a utilizar el dispositivo para la exposición de la sesión, la presentación comenzará desde esta página que está almacenada en la memoria.

Utilizando la opción “salir y eliminar sesión”, se realizará una petición hacia el servidor con el método HTTP DELETE, como se explicó en el Apartado 2.4.6, una vez que se ha completado la petición correctamente, se eliminará de la memoria de la aplicación los datos de la sesión.

Con la opción “cancelar”, continuará el control de la presentación.

Al término de la exposición, se finalizará la conexión *websocket* y posteriormente, la aplicación regresará a la actividad principal en la sección de lectura de QR.

2.5.5.1 Cambio de páginas

En cada cambio de página, bien mediante los botones o bien introduciendo el número de página mediante teclado, el portal devuelve, como confirmación, el número de página en el que se encuentra la presentación en el navegador.

Por otra parte, la aplicación de forma automática, guarda el número de página en la que se encuentra con el fin de poder habilitar o deshabilitar los botones de cambio de página cuando corresponda. Es decir, si se encuentra en la primera diapositiva, el botón de página anterior estará deshabilitado y del mismo modo, si es la última página se deshabilitará el botón de avanzar.

También, puesto que se almacena el número máximo de páginas de la presentación, el cuadro de texto para introducir el número de página está limitado al rango numérico entre 1 y dicho número máximo. Si el usuario introduce un número fuera de ese rango, no se permitirá el envío y, además, el diseño del cuadro de introducción numérica cambia de verde a rojo como se muestra en la Ilustración *2.35*

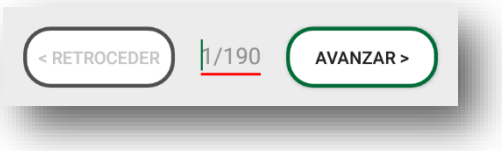

Ilustración 2.35: Vista número incorrecto

2.5.5.2 Controles de visualización

Respecto a la zona intermedia de control de visualización, cada uno de los botones corresponde con el envío de un comando específico que interpretará el portal y realizará la acción indicada. Estos controles son los siguientes:

- **Zoom +:** aumenta el tamaño de la presentación.
- **Zoom -:** disminuye el tamaño de la presentación.
- **Inicial:** restablece el tamaño de la presentación y centra la diapositiva en la pantalla.

- **Subir, bajar, izquierda, derecha:** mueve la diapositiva en el sentido indicado.

2.5.5.3 Notas

En último lugar, la sección de notas, cuenta con un apartado para introducir texto, con la opción de dejar de forma permanente el texto para las próximas exposiciones. Las notas se mostrarán en el portal en la sección que corresponda dependiendo de si se ha marcado o no la opción de mantener la nota.

También se ha incluido la opción de enviar una nota cuyo contenido es un enlace. Para ello es necesario marcar el cuadro de selección “Enlace”, en ese instante se muestra un nuevo cuadro de texto en el que poder introducir el título visible del enlace. La entrada de texto, tanto del título como del enlace, es necesario completarlo, en caso contrario no se podrá enviar la nota. Cuando se ha enviado el enlace, se limpian los cuadros de introducción de texto y se desmarca la opción de enlace ocultando a su vez el cuadro de introducción del título.

Por otro lado, se disponen dos botones con los que se podrá abrir o cerrar el panel de notas y eliminar el listado de notas de la presente exposición, enviando cada una su correspondiente comando hacia el portal.

3 RESULTADOS

A lo largo del desarrollo del sistema, así como en la posterior realización de pruebas, se han implementado mejoras y corregidos los fallos detectados. Entre las mejoras destacan el cambio de librería utilizada en el servidor para representar el documento, lo que conlleva cambios en la metodología utilizada. Esta nueva librería permite la selección del texto, así como el acceso a las direcciones URL integradas en la presentación. Otra de las mejoras importantes, es la posibilidad de enviar una presentación al servidor, ya que no estaba preparado en los pasos iniciales.

3.1 Resumen del sistema

En este apartado se describe un resumen del ciclo de vida del sistema completo, tanto la aplicación como el servidor.

En primer lugar, una vez que el usuario ha instalado la aplicación en su dispositivo móvil, al arrancarla, se mostrará en pantalla completa una animación de bienvenida durante dos segundos. Seguidamente, dará lugar a la actividad de inicio de sesión, mediante el cual, el usuario podrá introducir sus credenciales como son el usuario y la contraseña, para ello deberá estar previamente registrado en la base de datos del sistema. El usuario también podrá realizar su registro desde la misma actividad mediante el uso de un formulario en el que introducirá sus datos.

A continuación, la actividad posterior al registro permite realizar las acciones necesarias para definir una exposición. La primera operación necesaria para la exposición, es enviar una presentación al servidor mediante la sección preparada para este fin. Desde dicha sección es posible seleccionar el documento desde el explorador propio del dispositivo con el cual se elige el documento de formato PDF de la memoria o desde los servicios de almacenamiento en nube como Google Drive, si el dispositivo es compatible para ello.

El siguiente paso es la creación de la sesión, esta sección se encuentra en primer lugar de la actividad. En ella se proporciona un listado con el que indicar el documento para la exposición, así como introducir un nombre identificativo de la sesión, estos datos son enviados al servidor para su registro.

La sesión será accesible con la dirección URL, que se muestra en la aplicación una vez registrada en el servidor, en cualquier navegador (Chrome, Mozilla Firefox, Edge...). Al acceder a la sesión en el portal, se genera un código QR que es mostrado en la vista, además establece la conexión *websocket* con el servidor. Desde la aplicación móvil, en la sección de lectura del código se procederá a escanear el QR para acceder a la exposición. Tras el establecimiento de conexión con *websocket* por parte de la aplicación, esta realizará el envío del mensaje “OK” con el que se oculta el código QR y la

cabecera del portal y en su lugar se muestra el documento de la sesión, quedando como se muestra en la Ilustración 3.1.

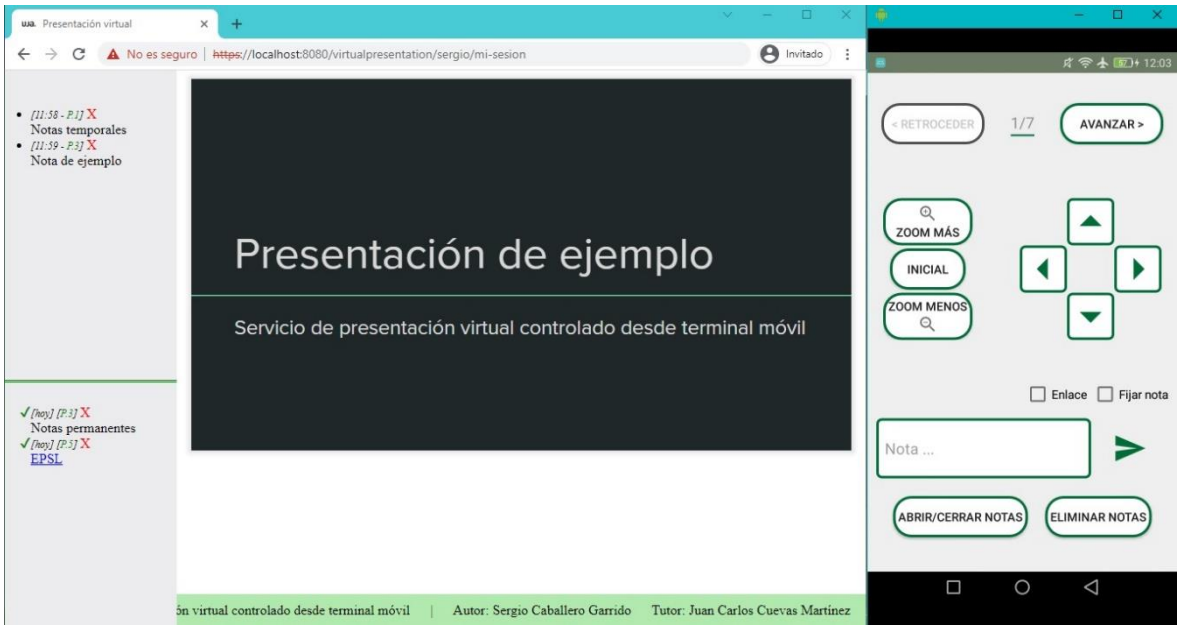

Ilustración 3.1: Portal exposición y aplicación

Durante la exposición, se podrán realizar diversas acciones englobadas en tres grupos: cambio de página, controles de visualización y manejo de notas.

El cambio de páginas se podrá realizar tanto desde la aplicación como desde las notas del portal. En el lado de la aplicación se diferencian dos botones, de avance y retroceso, y cuadro de entrada numérico en el que se introduce de forma manual. De forma alternativa, puesto que los objetos de las notas enviadas desde la aplicación llevan la propiedad de la página en la que se encuentra en ese momento, esta se mostrará con el formato de un enlace en la sección de notas del portal. Mediante el enlace con el número de la página la presentación cambiará al número correspondiente e informará a la app de dicho número.

En el lado del control de movimiento, se realizará mediante los controles de zoom que aumentan o disminuyen, según corresponda, únicamente el tamaño de la presentación. También están disponibles botones de movimiento (subir, bajar, izquierda, derecha) desde la aplicación o bien las barras de *scroll* desde el propio portal.

Por último, el manejo de notas desde la aplicación con la que se presentan los botones necesarios para mostrar u ocultar el listado, eliminar las notas enviadas y enviar una nota. Las notas se enviarán a partir del texto introducido en el campo correspondiente, adicionalmente se encuentran las opciones de fijar la nota y enviar un enlace. Con la primera opción, una vez enviada la nota, el servidor la registrará con sus respectivas propiedades en el fichero asociado a la presentación y se mostrará en el portal en la sección inferior del listado de notas. Estas notas volverán a representarse la próxima vez

que se exponga el documento. Con el uso de la opción “enlace”, una vez marcado se mostrará un nuevo campo donde el usuario introduce el título que se muestra en el enlace y la dirección URL deberá ir en el cuadro de introducción de texto inferior.

Para concluir la exposición el usuario pulsará el botón atrás del dispositivo y se mostrará una ventana ofreciendo las opciones de salir, continuar o salir y eliminar la sesión. Mediante la última opción, la aplicación realiza una petición al servidor para eliminar el registro de la sesión. Con las opciones de salir se enviará un mensaje al portal indicando que la sesión ha finalizado, en el momento que este lo recibe se realizará una nueva carga de la página volviendo a generar un nuevo código QR para la sesión o informando que la sesión no existe. Una vez ha cerrado la actividad de la presentación, la aplicación regresa a la actividad principal, en concreto a la sección donde se escanea el código QR.

Como último paso, cuando el documento ha cumplido su función y el usuario desea eliminarlo podrá hacerlo mediante la sección de la actividad principal desarrollada para este fin. En mencionada vista, mediante la selección de un documento entre la lista de los pertenecientes al usuario y el botón eliminar, se realizará una petición al servidor para realizar dicha acción. Previamente al envío de la solicitud, se mostrará una alerta preguntando si se desea eliminar la presentación seleccionada, una vez confirmado, se realizará la petición. En el servidor, se eliminará tanto el registro de la presentación y las sesiones asociadas como los ficheros, el documento en sí y el listado de notas asociadas.

3.2 Control de errores

A continuación, se explican algunos de los posibles errores ocurridos en la aplicación, por parte de la misma o provocados por el usuario.

Uno de los errores más comunes por parte del usuario, es la introducción de credenciales (usuario y contraseña) de forma incorrecta. De esta forma, una vez consultado el combo de datos en el servidor, este dará como respuesta el código de estado 401 y en la aplicación se mostrará una alerta con la que se informa al usuario que las credenciales no son correctas, como se muestra en la Ilustración 3.2.

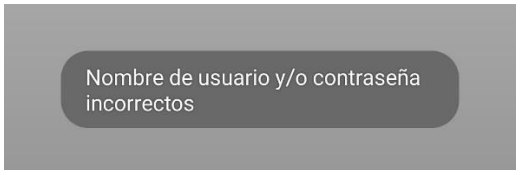

Ilustración 3.2: Credenciales incorrectas

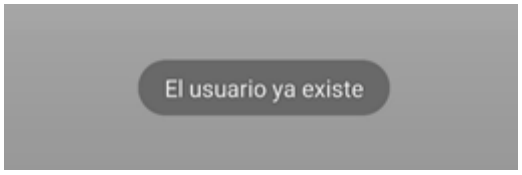

Ilustración 3.3: Usuario incorrecto

Otro caso de fallo similar al anterior, cuando un cliente pretende registrarse en el sistema, pero introduciendo un nombre de usuario que ya está siendo utilizado por otro cliente. Al realizar las comprobaciones en el servidor y verificar que ya existe, este responderá con el código 409 y, en la aplicación, se mostrará un mensaje informando del error (Ilustración 3.3).

En el momento de la autenticación del usuario, el servidor genera un *token* con algunos datos del mismo, así como marcas temporales de creación y expiración. Este *token* tiene establecida una caducidad transcurridos dos días, por lo que las posteriores peticiones realizadas no serán completadas ya que se comprueba previamente la validez del *token* y se rechaza la petición. Cuando la aplicación interpreta el código de estado, se mostrará una ventana emergente en la que se solicita nuevamente la contraseña y es enviada para generar un nuevo *token*. De esta forma, se podrá continuar utilizando la aplicación sin necesidad de pasar por la pantalla de autenticación.

En la selección para enviar y registrar un documento en el sistema no es posible seleccionar un archivo cuya extensión sea distinta de ".PDF". No obstante, en algunos dispositivos dichos archivos, es posible seleccionarlos, el explorador de archivos cambia la extensión a PDF, provocando un fallo a la hora de acceder al fichero para proceder a su envío, lo que provoca el cierre inesperado de la aplicación. Para evitar cualquier problema por el tipo de fichero, en el servidor se comprueba que sea del tipo mime "application/pdf", en caso contrario se rechazará la petición informando del error en la aplicación.

Para el acceso a la exposición se procederá a la lectura de código QR de la sesión. En el momento de la lectura, si por cualquier motivo el servidor no se encuentra disponible o el dispositivo no tiene conexión, una vez que se ha iniciado la vista del control de la presentación en la app, los botones se mostrarán deshabilitados y al pie de la vista se muestra un diálogo informando que no se ha podido conectar y se ofrece la posibilidad de volver a intentarlo.

Durante la exposición, el sistema no está preparado para la pérdida de conexión, por lo que, si por cualquier motivo se produce, no es posible volver a conectar. Para continuar con la exposición, será necesario volver a generar el código QR cargando de nuevo la página del portal y en la aplicación salir de la exposición y volver a escanear el código.

4 PRESUPUESTO

En el presente proyecto, se han invertido un total de 300 horas, con un importe por hora de 25€. A continuación, se muestra la distribución de tiempo por cada una de las etapas del mismo:

Tabla 4.1: Presupuesto

Concepto	Tiempo (horas)	Importe (€)
Evaluación tecnologías y pruebas	15	375
Montar estructura del sistema	15	375
Cambio de tecnologías y/o versiones	10	250
Desarrollo	200	5000
Testeo	40	1000
Corrección de fallos detectados en test	20	500
Base imponible	300	7.500,00 €
IVA (21%)		1.575,00 €
TOTAL		**9.075,00 €**

El importe total del proyecto asciende a la cantidad de **nueve mil setenta y cinco euros** (9.075,00 €)

5 CONCLUSIONES Y LÍNEAS FUTURAS

Una vez realizado el desarrollo y pruebas del sistema, se ha logrado la obtención de los objetivos básicos planteados al inicio del proyecto. En concreto, se ha definido un sistema mediante el cual poder realizar la presentación de diapositivas desde un dispositivo móvil que cuenta con una autenticación previa y lectura de código QR. Adicionalmente, se han implementado mejoras, como son la posibilidad de envío de notas hacia el portal y la facilidad de envío de un documento mediante la aplicación.

El presente proyecto cuenta con las funcionalidades básicas para su correcto desempeño, aunque existen varias líneas de mejora que podrán aportar un valor añadido. Entre las posibles implementaciones, se encuentran:

A nivel de seguridad, el sistema, cuenta con HTTPS y encriptado de las contraseñas, como mejora se podría implementar opciones de seguridad para *WebSocket* de forma que un usuario externo no pueda operar sobre la presentación en exposición. También sería interesante encriptar el texto utilizado para generar el código QR de acceso a la exposición.

Como se describe en los objetivos opcionales del presente proyecto, para la exposición, desarrollar un puntero controlado desde la aplicación móvil, y, de forma similar, permitir realizar dibujos sobre la presentación.

Para la autenticación, además de la forma tradicional mediante usuario y contraseña, permitir el acceso a partir de una cuenta de Google u otros servicios similares, como se muestra en Ilustración 5.1, extraída como ejemplo de acceso del portal prezi.com.

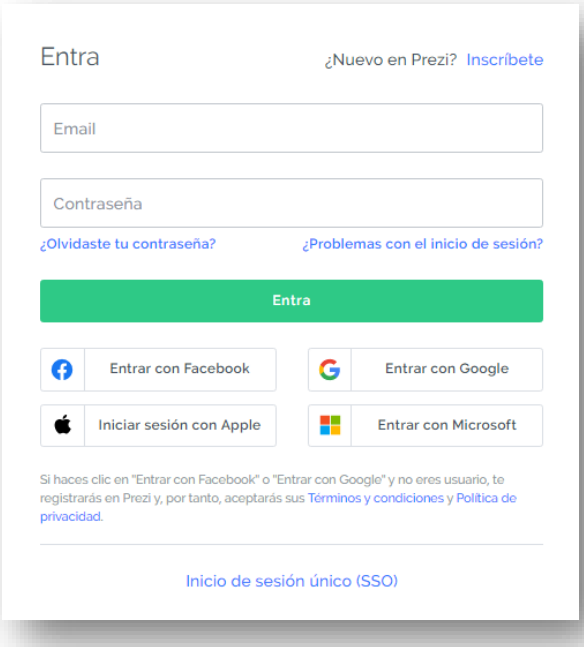

Ilustración 5.1: Autenticación ejemplo, prezi.com

Por último, realizar los ajustes y conversiones necesarias para poder generar la aplicación en modo multiplataforma, para Android, iOS u otros sistemas operativos, permitiendo ampliar el público objetivo de uso de la misma.

6 REFERENCIAS BIBLIOGRÁFICAS

[1] About Us. Share and Discover Knowledge on SlideShare. [Consulta: febrero 2021]. Disponible en: https://es.slideshare.net/about

[2] SlideShare – Wikipedia, la enciclopedia libre [Consulta: febrero 2021] Disponible en: https://es.wikipedia.org/wiki/SlideShare

[3] Presentaciones de Google [Consulta: 1 febrero 2021] Disponible en: https://www.google.com/slides/about/

[4] Presentaciones de Google: presentaciones online [Consulta: febrero 2021] Disponible en: https://workspace.google.com/products/slides/

[5] Novedades de presentaciones de Google [Consulta: febrero 2021] Disponible en: https://support.google.com/docs/answer/9233149?hl=es&ref_topic=9052636

[6] Introducción a Sway [Consulta: febrero 2021] Disponible en: https://support.microsoft.com/es-es/office/introducci%C3%B3n-a-sway-2076c468-63f4-4a89-ae5f-424796714a8a

[7] ¿Qué es Prezi y para qué sirve? [consulta: febrero 2021] Disponible en: https://www.appf.edu.es/que-es-prezi-para-que-sirve/

[8] Descripción general de Volley. (s. f.). Android Developers. [consulta: noviembre 2021] https://developer.android.com/training/volley?hl=es

[9] Gotev, A. (s. f.). Android Upload Service. [consulta: noviembre 2021] GitHub https://github.com/gotev/android-upload-service/wiki

[10] Introduction. Socket.IO. [consulta: noviembre 2021] https://socket.io/docs/v4/

[11] 10 motivos para usar Node.js para desarrollar aplicaciones web. Universia [Consulta: abril 2021] Disponible en: https://www.universia.net/es/actualidad/habilidades/10-motivos-usar-nodejs-desarrollar-aplicaciones-web-1154033.html

[12] 5 mejores ejemplos de aplicaciones Node.js para empresas. Hack a boss. [consulta: abril 2021]. Disponible en: https://hackaboss.com/blog/5-mejores-ejemplos-de-aplicaciones-node-js-para-empresas/

[13] Express.js – Wikipedia, la enciclopedia libre [Consulta: abril 2021] Disponible en: https://en.wikipedia.org/wiki/Express.js

[14] Introducción a Express/Node. MDN Web Docs, mozilla [Consulta: abril 2021] Disponible en: https://developer.mozilla.org/es/docs/Learn/Server-side/Express_Nodejs/Introduction

[15] *Token* – Wikipedia, la enciclopedia libre. [Consulta: abril 2021] Disponible en: https://es.wikipedia.org/wiki/*Token*_(inform%C3%A1tica)

[16] RFC 7519: *JSON* Web *Token* (*JWT*). IETF Datatracker [Consulta: noviembre 2021] Disponible en: https://datatracker.ietf.org/doc/html/rfc7519

[17] Introduction to *JSON* Web *Tokens*. *JWT* [Consulta: abril 2021] Disponible en: https://jwt.io/introduction

[18] MySQL 8.0 Reference Manual, what is MySQL? [Consulta: junio 2021] Disponible en: https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html

[19] The WebSocket Protocol. IETF Datatracker [Consulta: febrero 2021] Disponible en: https://datatracker.ietf.org/doc/html/rfc6455

[20] Aspectos fundamentales de la aplicación. Android developers [Consulta: julio 2021] Disponible en: https://developer.android.com/guide/components/fundamentals?hl=es-419

[21] PDFJS-DIST. NPM [Consulta: noviembre 2021] Disponible en: https://www.npmjs.com/package/pdfjs-dist

[22] Augmented BNF for Syntax Specifications: ABNF. IETF Datatracker [Consulta: noviembre 2021] Disponible en: https://datatracker.ietf.org/doc/html/rfc5234

7 ANEXOS

7.1 Manual de instalación del servidor

El proyecto se encuentra alojado en el sistema de control de versiones GitHub de forma pública en la dirección URL: https://github.com/scg00046/virtualPresentations_Server. Los ficheros se podrán descargar mediante las opciones que ofrece GitHub en su página web (Ilustración 7.1) o mediante línea de comandos. Para obtener el proyecto mediante comandos, se abrirá una terminal de línea de comandos como es CMD en Windows, desde la misma, se ubica en la ruta de directorios donde se desee alojar los archivos y se ejecuta el siguiente comando:

```
git clone https://github.com/scg00046/virtualPresentations_Server.git
```

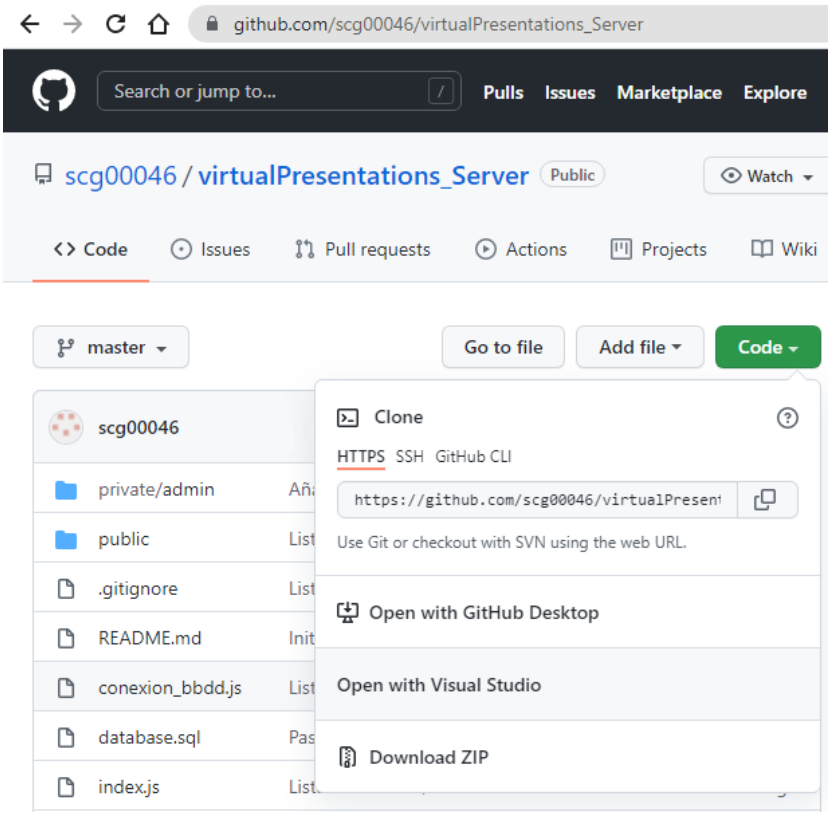

Ilustración 7.1: Descargar proyecto de GitHub

Cuando esté completa la descarga, en primer lugar, será necesario preparar la base de datos. Para ello, deberá instalarse un servidor SQL o utilizar uno ya existente. Posteriormente, mediante un gestor de bases de datos relacionales, como por ejemplo MySQLWorkbench (https://www.mysql.com/products/workbench/), Heidi (https://www.heidisql.com/), entre otros, se iniciará la conexión a dicho servidor y se ejecutará el script “database.sql” que se encuentra en la raíz del proyecto obtenido previamente.

Antes de poder iniciar el proyecto, es imprescindible que el equipo disponga de *NodeJS* y el gestor de paquetes *NPM*, se puede obtener en la página web oficial https://nodejs.org/es/, Ilustración 7.2. A continuación, nuevamente en la raíz del proyecto y desde el terminal de línea de comandos, se ejecutará el comando "`npm install`" con el que se instalan los paquetes necesarios para el servidor.

Ilustración 7.2: Descarga NodeJS

Cuando se han realizado las anteriores acciones, se modificará el archivo "conexion_bbdd.js" en el objeto de configuración de la conexión, cambiando el host ("localhost") por la dirección IP donde se encuentre el servidor SQL, así como el usuario (user) y contraseña (password) establecidos en la instalación del servidor SQL, Ilustración 7.3.

```
// Conexión a la base de datos
//parámetros de la conexión
var con = mysql.createConnection({
    host: "localhost",
    port: 3306,
    user: "root",
    password: "root",
    database: "presentacionvirtual"
});
```

Ilustración 7.3: Configuración base de datos

El servidor podrá iniciarse una vez se han configurado los parámetros mencionados anteriormente, mediante alguno de los comandos: "`node index.js`" o "`npm start`", mostrando a continuación las líneas que se muestran en la Ilustración 7.4. Si la base de

datos no se encuentra disponible o no se puede acceder a ella, el servidor no podrá iniciarse.

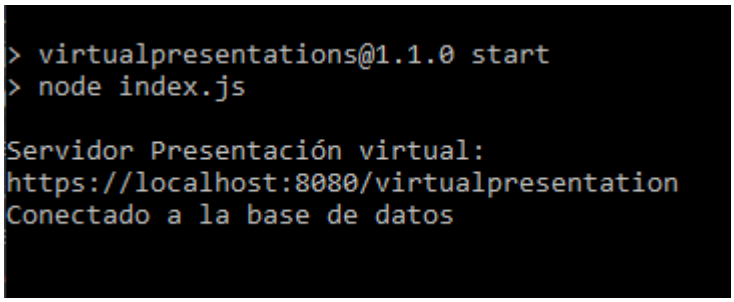

Ilustración 7.4: Servidor iniciado

Como prueba del correcto funcionamiento, desde un navegador (Edge, Chrome, etc.) se accederá a la dirección URL donde se encuentra la documentación de la API https://localhost:8080/virtualpresentation/docs/

7.2 Manual generar archivo instalación aplicación

El proyecto de la aplicación para dispositivos Android se encuentra alojado en GitHub en la dirección URL https://github.com/scg00046/VirtualPresentation_app con la posibilidad de obtenerlo a partir del portal GitHub o bien mediante línea de comandos con: `git clone https://github.com/scg00046/VirtualPresentation_app.git`

Desde el programa AndroidStudio (disponible para su descarga en: https://developer.android.com/studio), se abrirá el proyecto para poder configurar la dirección del servidor mediante el archivo JAVA "Constants" que se encuentra en el árbol de directorios "app/src/main/java/es/ujaen/virtualpresentation/data", modificando las constantes "IP" y "PORT" según donde esté alojado el servidor (Ilustración 7.5). Para generar el archivo de instalación para Android, se realizará desde el menú "Build", "Build Bundle / APK" y la opción "Build APK" (Ilustración 7.6), una vez compilado el proyecto se generará el archivo APK en el directorio "app/build/outputs/apk/debug". El fichero generado se deberá copiar en el dispositivo móvil desde el que se abrirá para instalar la aplicación

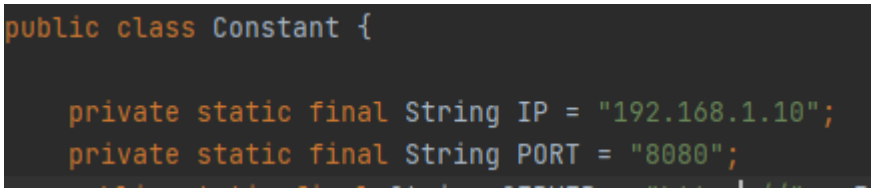

Ilustración 7.5: Configuración del servidor en la aplicación

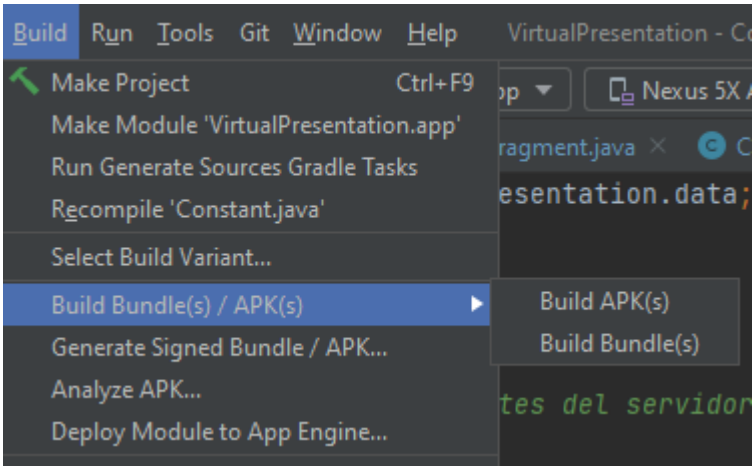

Ilustración 7.6: Generar APK

7.3 Manual de aplicación

7.3.1 Registro y autenticación

En primer lugar, el usuario deberá registrarse en el sistema. Para proceder al registro, desde la pantalla de autenticación, pulsando sobre “Registro” (Ilustración 7.7), se abrirá un formulario donde el usuario introducirá sus datos. Los campos obligatorios que debe completar son usuario, contraseña y nombre y opcionalmente se podrán indicar los apellidos y el correo electrónico. Posteriormente, una vez completado el registro, con las credenciales introducidas, se autenticará en el sistema en la pantalla de acceso.

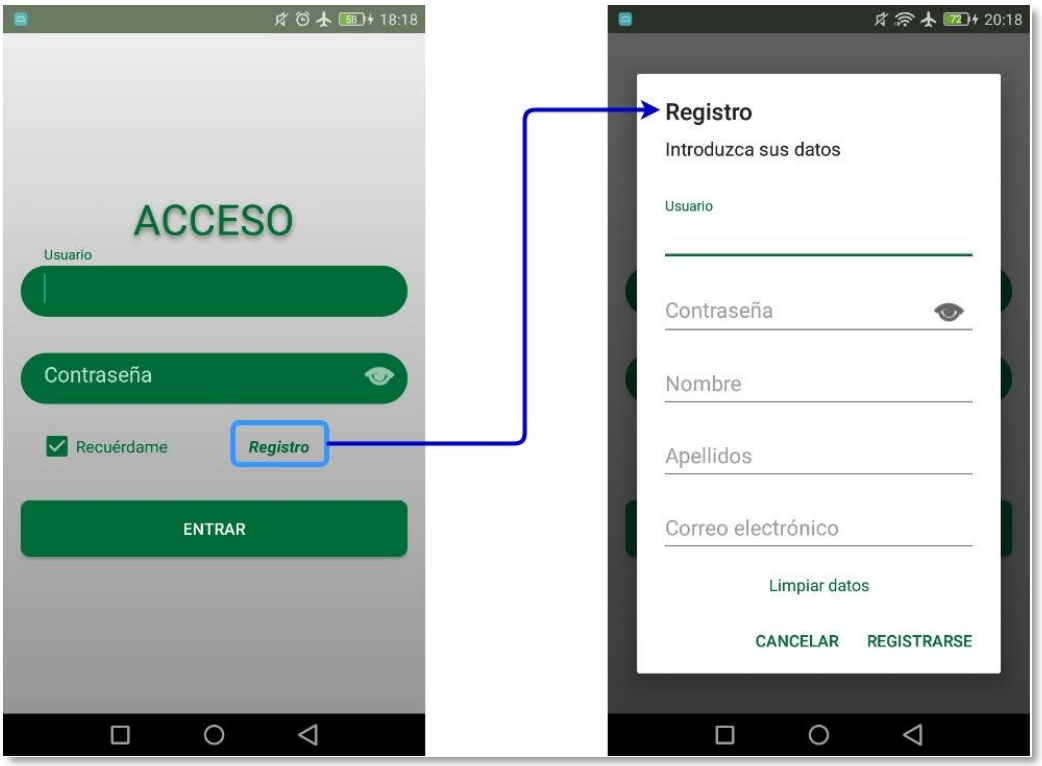

Ilustración 7.7: Autenticación y registro

7.3.2 Pantalla principal

Cuando se ha completado el proceso de autenticación, el usuario, se encontrará en la sección de crear sesión, para cambiar de sección deberá pulsar el menú donde se listan las diferentes opciones de uso de la aplicación, botón superior izquierda, Ilustración 7.8.

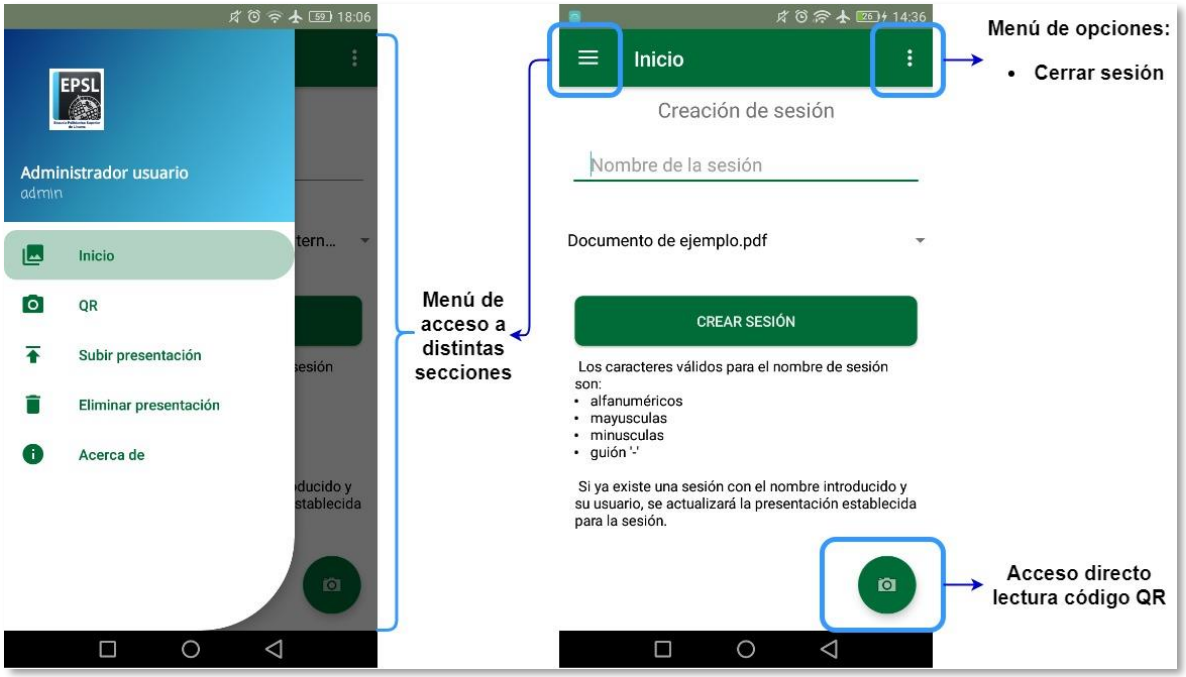

Ilustración 7.8: Menú e inicio actividad principal

Para utilizar el sistema, el primer paso será subir una presentación, para ello se seleccionará en el menú el ítem “Subir presentación”. Desde esta sección que se muestra en Ilustración 7.9, pulsando el botón “Seleccione la presentación”, en primer lugar, se solicita el permiso de acceso a la memoria, si no estaba concedido previamente. Posteriormente, se abrirá el gestor de archivos propio de Android, se navegará entre las carpetas hasta la ubicación del documento deseado, permitiendo la elección de documentos también desde Google Drive. El documento seleccionado deberá ser en formato PDF, una vez elegido, se habilitará el botón “Subir presentación”, cambiando el color a verde y mostrando el nombre del documento seleccionado entre ambos botones. Pulsando este último botón, se enviará la presentación al servidor, mostrando una notificación con el proceso de carga y finalización.

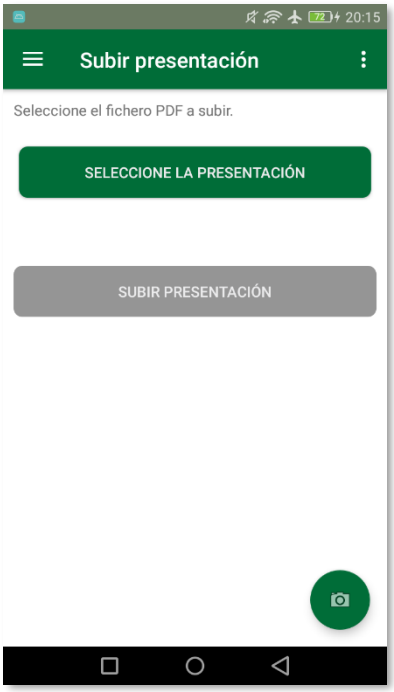

Ilustración 7.9: Subir presentación

El segundo paso es la creación de la sesión, desde el ítem “Inicio” del menú. En la sección que se muestra en Ilustración 7.10, se introducirá el nombre deseado para la sesión, se marcará el documento a exponer de la lista desplegable que se muestra y se pulsará el botón “Crear sesión” para registrarla en el sistema. El nombre de la sesión está limitado en cuanto a tipo de caracteres, únicamente se permiten alfanuméricos distinguidos por mayúsculas y minúsculas, y el carácter guion “-“.

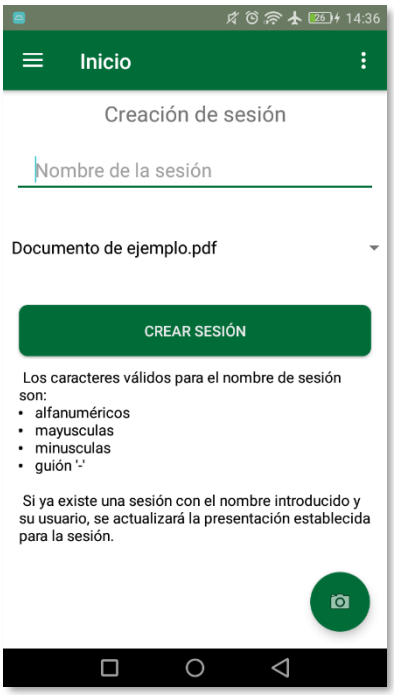

Ilustración 7.10: Crear sesión

Por último, en cualquier momento se podrá eliminar una presentación, para ello abrirá la sección “eliminar presentación” desde el menú. En la pantalla que se muestra en Ilustración 7.11, se seleccionará de la lista de presentaciones, el documento que se desee eliminar y se pulsará el botón “Eliminar”. Para completar el proceso, se mostrará una alerta para confirmar la acción.

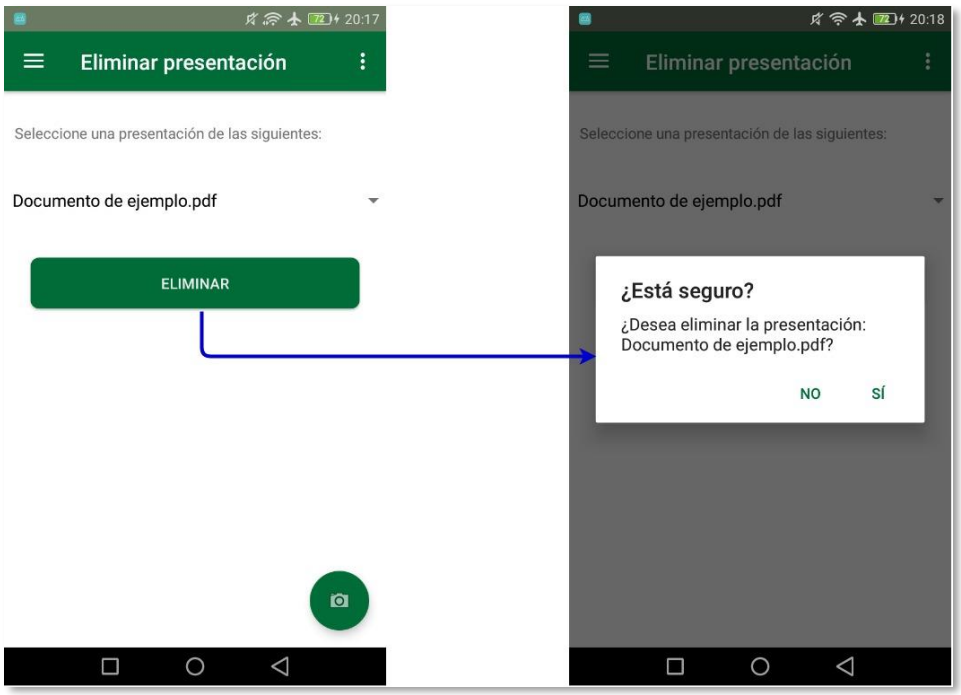

Ilustración 7.11: Eliminar presentación

Cuando el usuario quiera cambiar de cuenta o bien simplemente cerrar la suya, en el menú de opciones situado en la parte superior izquierda (Ilustración 7.8) se pulsará sobre “Cerrar sesión” lo que permite volver de nuevo a la pantalla de autenticación.

Una vez que se autentica el usuario, podrá utilizar la cuenta durante 2 días, trascurrido ese tiempo, por seguridad, se volverá a solicitar la contraseña como se muestra en Ilustración 7.12. En la alerta se muestran dos opciones, el usuario podrá cerrar la sesión y volver a la pantalla de autenticación o bien introducir la contraseña y pulsar enviar para continuar utilizando la misma cuenta de usuario.

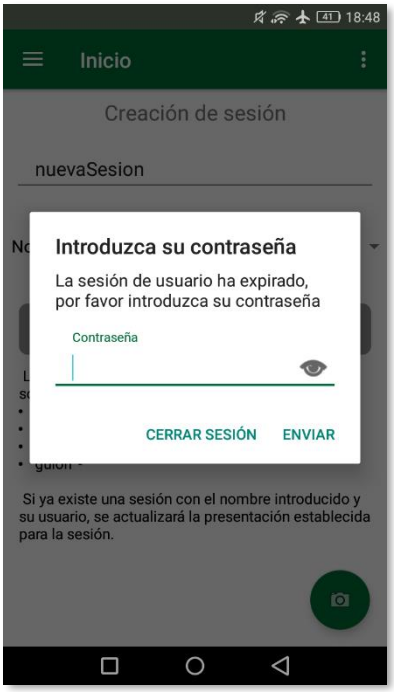

Ilustración 7.12: Solicitud contraseña

7.3.3 Exposición

Desde el navegador donde se quiera realizar la exposición, se accederá a la dirección URL generada al crear la sesión (se muestra en la misma pantalla de la creación). En dicha página se mostrará un código QR el cual será escaneado desde la aplicación, en el ítem "QR" del menú de la aplicación. La sección de lectura del código, Ilustración 7.13, solicitará acceso a la cámara si no se ha permitido anteriormente, con la lectura de un código válido dará lugar a la pantalla de control de la presentación en la aplicación y al documento en el navegador (Ilustración 7.14).

Ilustración 7.13: Lectura código QR

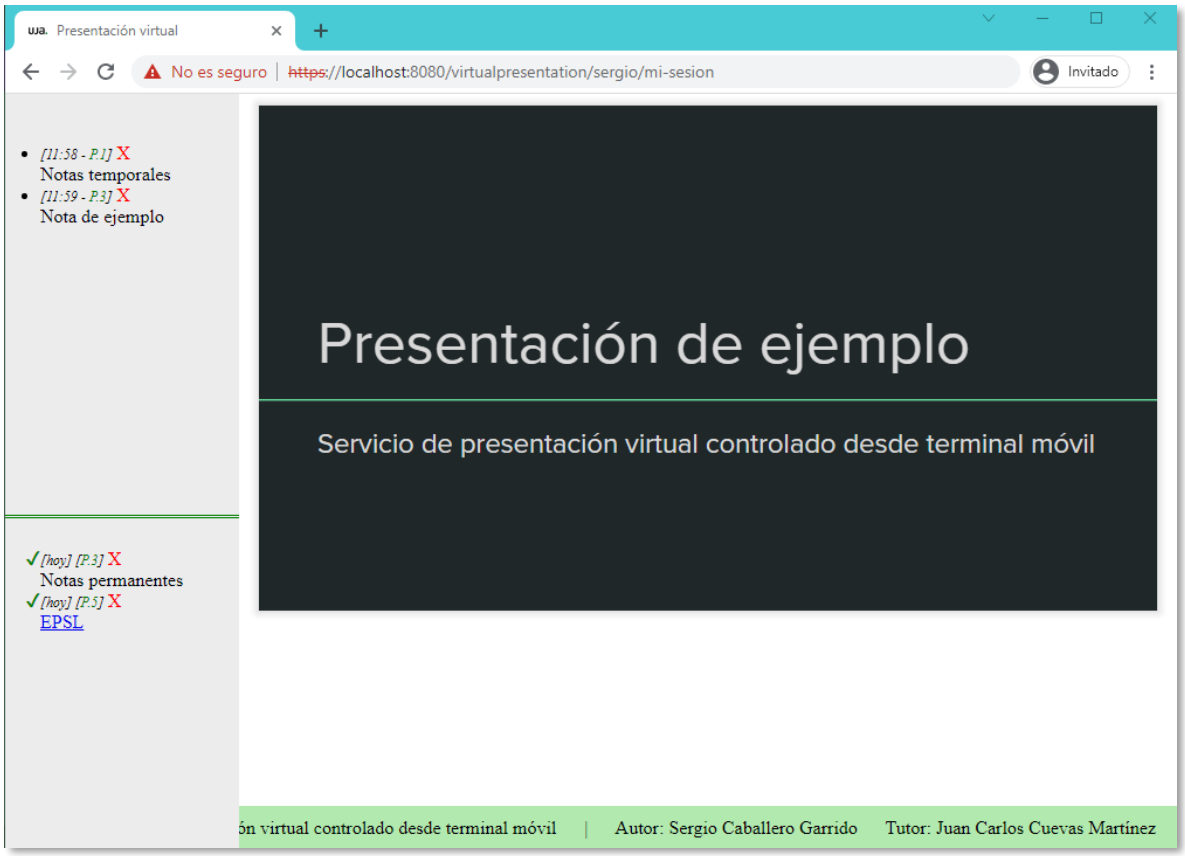

Ilustración 7.14: Portal web

Los controles de la presentación están distribuidos como se muestra en Ilustración 7.15.

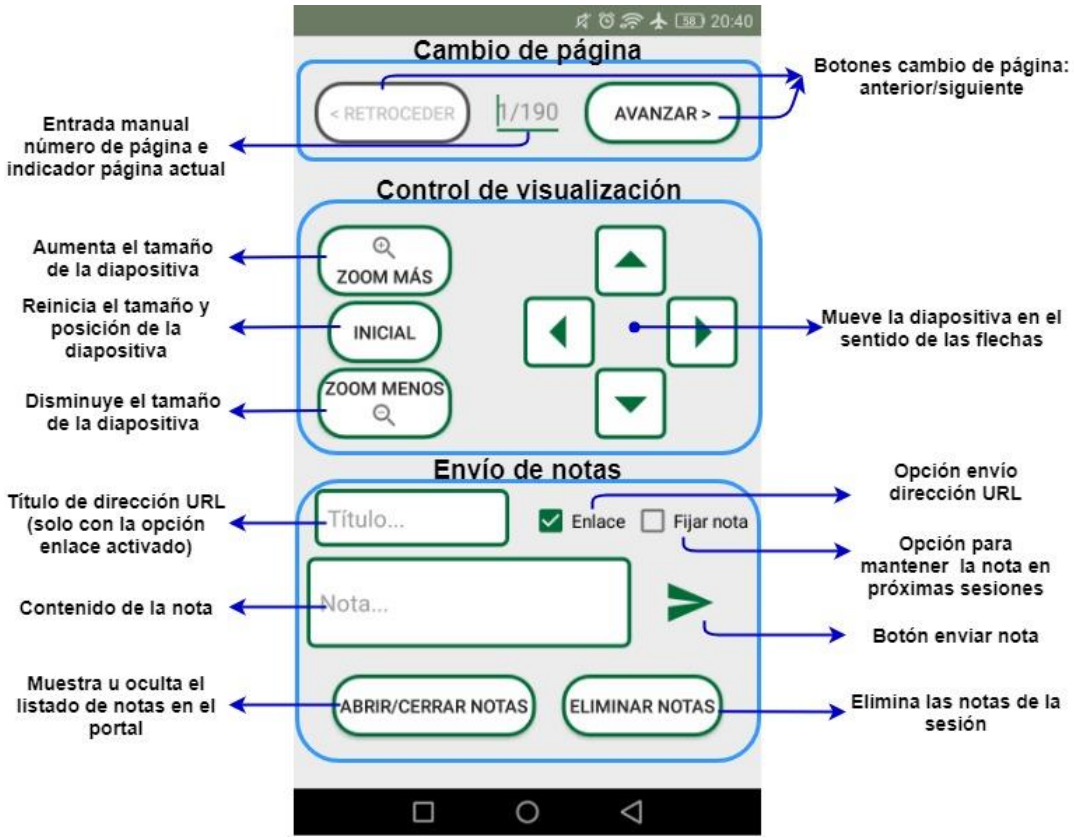

Ilustración 7.15: Controles presentación

En la entrada del número de página se mostrará el número de la página actual y el número de páginas totales que tiene el documento, a su vez se podrá introducir el número de página deseado, no siendo posible introducir un valor que no se encuentre en el rango de páginas. También se disponen dos botones para cambiar de página a la anterior o posterior, quedando deshabilitados en el momento que se encuentre en la primera o última página respectivamente.

En los controles de movimiento, los botones de zoom permiten aumentar o disminuir el tamaño de la presentación, permitiendo el desplazamiento mediante los botones situados a la derecha. Además, el botón “inicial”, centrará la presentación y volverá al tamaño inicial del documento.

Para el envío de notas se disponen de dos opciones (casillas de selección):

- **Enlace**: marcando esta opción se muestra una nueva entrada de texto en la que se introducirá el título de la dirección URL y ésta se escribirá en el cuadro de entrada inferior al título.
- **Fijar**: permite registrar la nota para mostrarla también en las sucesivas sesiones para el documento que se está exponiendo.

En la sección de notas del portal, por cada una de las notas (Ilustración 7.16), junto a la fecha u hora, en color verde se muestra el número de página (ej P.5), haciendo *click* en él se dirigirá a la página indicada. Además, pulsando sobre la "X" en color rojo junto cada nota, se eliminará esta de forma individual.

Ilustración 7.16: Ejemplo notas

Todos los enlaces mostrados en el portal, tanto en las notas como los presentes en el documento, se abrirán en una nueva pestaña del navegador.

Cuando se desee concluir la exposición, se pulsará sobre el botón atrás del dispositivo a través del cual se mostrará una ventana de confirmación (Ilustración 7.17) con las siguientes opciones:

- **Sí**: cierra la exposición y regresa a la sección de lectura del código QR.
- **Salir y eliminar la sesión**: cierra la exposición, elimina el registro de la sesión en el servidor y regresa a la sección QR.
- **Cancelar**: continua con la exposición.

Ilustración 7.17: Finalizar sesión