Escuela Politécnica Superior de Linares

Grado en Ingeniería Telemática

Universidad de Jaén

Escuela Politécnica Superior de Linares

Trabajo Fin de Grado

Control de acceso a dependencias seguro basado en NFC para terminales Android con sistema de gestión centralizado

Alumno: Eva Rebollo Calvo

Tutor: Prof. D. Juan Carlos Cuevas Martínez.

Depto.: Ingeniería de Telecomunicación.

Septiembre, 2017

INDICE

Índice de Figuras

ÍNDICE DE TABLAS

Resumen

En esta memoria se describe como se ha llevado a cabo el desarrollo e implementación de la aplicación para el control de acceso a dependencias seguro para dispositivos Android utilizando la tecnología NFC.

Haciendo uso de la aplicación desarrollada en este proyecto, es posible disponer de un control de acceso de usuarios, no solo en una infraestructura fija, sino que como se ha desarrollado haciendo uso de los terminales móviles, se puede utilizar para garantizar el control de accesos en cualquier emplazamiento que no disponga de infraestructura previa. Para la identificación de los usuarios se utilizan tarjetas NFC.

Solamente es necesario contar con acceso a Internet en el terminal para hacer uso de ella. Este acceso puede ser a través de la red de telefonía móvil, infraestructura WIFI o Ethernet

Junto con esta aplicación Android, se ha desarrollado una aplicación servidora, en la que se sustenta el funcionamiento del control de acceso a las dependencias

En este desarrollo se ha tenido muy en cuenta las medidas de seguridad necesarias, para que los datos de los usuarios en todo momento estén protegidos durante las comunicaciones y las peticiones a la aplicación servidora.

1 MEMORIA

1.1 Introducción

En este proyecto se ha creado un servicio que permite el control de acceso de los usuarios a las diferentes estancias de un edifico o recinto de forma segura. Para identificar a los usuarios se usan tarjetas identificativas NFC pasivas, que será leídas por un dispositivo móvil compatible con esta tecnología. A su vez se ha desarrollado una aplicación web que permite la administración y el diseño de las políticas de control de acceso en las que se basa la aplicación móvil.

En la actualidad los nuevos teléfonos móviles ofrecen multitud de aplicaciones que nos permiten tener en un dispositivo de pequeño tamaño gran cantidad de funcionalidades. Por este motivo para llevar a cabo este proyecto, se ha elegido crear una aplicación para dispositivos móviles con sistema operativo Android, que funcione a modo de “cerradura” para el acceso a cada una de las estancias.

Conjuntamente se ha llevado a cabo una aplicación servidora, con la que se comunica la aplicación Android, para interactuar en la base de datos y administrar los permisos y registros de cada una de las estancias y usuarios.

Esta aplicación servidora se podrá consultar en cualquier momento vía web y permitirá establecer la normativa de acceso en el interior de la estancia, la administración de las salas, usuarios, permisos, registros, aforo y terminales móviles que conforman la estancia a controlar.

El servicio de identificación se realiza con el uso de distintas tecnología:

- Android es el sistema operativo en el que se ha programado la aplicación.
- Etiquetas NFC que almacenan los datos que identifican a las personas que quieren acceder al edificio.
- `HTTPS` es el protocolo de comunicación segura entre la aplicación Android y la aplicación servidora.
- Python es el lenguaje en el que se ha programado la aplicación servidora.
- AES es el algoritmo de cifrado empleado para cifrar los datos que se envían desde los dispositivos móviles y se descifran en la aplicación servidora.
- SQLite como gestor de base de datos.

Hoy en día en cualquier momento por medidas de seguridad, es necesario llevar un control de las personas que entran o salen de cualquier estancia. Para ello se hace necesario un sistema que permita identificar, controlar aforo y dar accesos a cada persona que permanecen en las diferentes estancias del recinto.

Debido a que esta aplicación no tiene la necesidad de disponer de una infraestructura fija para su implementación, su uso es ideal para controlar el acceso en eventos que se realicen al aire libre, recintos donde no haya una infraestructura previa o incluso en eventos deportivos donde es necesario contar con dispositivos que permitan gran movilidad.

1.2 Objetivos

El principal objetivo de este proyecto es diseñar un sistema seguro que permita controlar el acceso a las diferentes dependencias de un recinto a través de la tecnología inalámbrica NFC y utilizando para su gestión una aplicación web.

Las funciones principales que debe tener el sistema son las siguientes:

- Autenticación en la aplicación móvil a través de tarjetas NFC.
- Registro de cada acceso en una aplicación servidora remoto que podrá otorgar acceso o no en función de políticas o permisos concretos.
- Control de aforo.
- Gestión de la aplicación servidora para controlar el acceso vía web.
- Comunicación entre la aplicación móvil y la aplicación servidora debe estar cifrada.

Objetivos secundarios:

- La aplicación móvil desarrollada deberá estar preparada para adaptarse a diferentes tipos de terminales móviles que existen actualmente en el mercado.
- La aplicación servidora debe ser funcional en diferentes tipos de dispositivos y resoluciones de pantalla.

1.3 Estado del Arte

El uso del NFC está transformando la forma de relacionarnos con el entorno, ya que el teléfono móvil se ha convertido en la tarjeta de crédito, en llavero o en mando a distancia entre otros muchísimos usos.

La tecnología NFC [1] en los últimos años ha desarrollado un gran auge y hoy en día se usa en muchos nuevos desarrollos. Esta tecnología nos permite la interacción entre dispositivos de forma intuitiva, fácil y segura.

Los usos más comunes de esta tecnología son la identificación y el intercambio de datos a través de las aplicaciones, como por ejemplo el pago a través de teléfonos móviles.

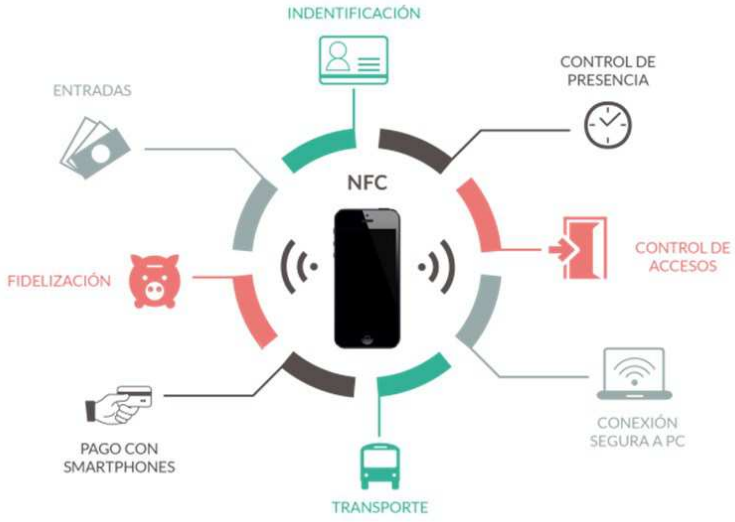

Figura 1.Usos de la tecnología NFC[1].

- **Identificación vía NFC:** El nuevo DNI 3.0, incorpora la conexión NFC, que evita en muchos casos la necesidad de disponer de un lector de tarjetas especial. Así, se abre la posibilidad de utilizar el DNI a través de teléfonos móviles o tabletas que dispongan de tecnología NFC, de tal manera que, si alguna entidad creara una aplicación en la que fuera necesario identificarse, pueda hacerse simplemente acercando el DNI 3.0 a la antena NFC de dichos dispositivos móviles.
- **Control de Accesos:** También es utilizada en el control de accesos, permitiendo la comunicación entre los dispositivos con NFC y los lectores con esta tecnología incorporada. Su funcionamiento es sencillo, lo único que hay que hacer es acercar el dispositivo móvil al lector NFC para que intercambien los datos y dependiendo de la configuración y de los permisos que tengamos configurados en el sistema podremos acceder a unas zonas u otras. Por ejemplo la Universidad Pontificia de Salamanca ha diseñado un sistema que automatiza el seguimiento de la asistencia a clase gracias a la tecnología NFC.
- **Recogida de datos:** Como ejemplo práctico del uso de la tecnología NFC, la organización ,conocida por sus siglas, ***AAC***[2] (del inglés, *Automated Assembly Corporation*) está en negociaciones con varias empresas para

[1] Disponible en : https://www.by.com.es/wp-content/uploads/2015/02/Screenshot_12-768x529.jpg.

[2] AAC: http://www.autoassembly.com/

empezar a producir en masa el sistema ***InfoSkin***[3], una nueva línea de etiquetas NFC que se adhieren al cuerpo humano. Estas etiquetas NFC están pensadas para su uso en la industria de la salud y se utilizarían para mejorar la atención al paciente, haciendo que sea más fácil para el personal médico actualizar los registros de tratamiento de cada paciente.

- **Pago con móvil gracias al NFC:** En los teléfonos inteligentes que usan el sistema operativo Android la tecnología NFC está disponible desde hace ya un tiempo.

1.3.1 Aplicaciones prácticas de NFC

Para conocer mejor la tecnología NFC se van a mencionar algunas de sus aplicaciones prácticas [2], que muestran hasta donde se puede llegar con la utilización de dicha tecnología.

- **Transacciones:** Una de las aplicaciones que existe en algunos países es el hecho de poder realizar transacciones en cuestiones del día a día. Por ejemplo, a la hora de realizar el pago de un ticket del metro. Con la utilización de una tarjeta y simplemente acercándola al dispositivo de validación se puede ejecutar la transacción, descontar el saldo que ha costado el viaje y abrir la barrera para que el usuario pueda acceder al metro. La tecnología NFC también se está utilizando en otros medios de transporte para comprobar que cada pasajero ha pagado su pasaje, esto ocurre actualmente en ciudades como Los Ángeles. Otro tipo de transacciones podría ser el hecho de realizar un pedido en un restaurante, con una aplicación que contenga los menús de los restaurantes y de camino al restaurante ir realizando la selección de lo que se quiere comer. A la entrada del restaurante se haría la transmisión de la información y una vez en la mesa sólo habría que esperar a que el camarero sirva los platos. Otra sencilla opción sería la de realizar una transferencia de dinero de una persona a otra. Simplemente acercando el teléfono al de un amigo y validando la transacción, sin duda una forma rápida y fácil de prestar o devolver dinero.
- **Folletos digitales:** Para evitar utilizar papel y tener un coste por cada uno de los folletos o periódicos, se podría transmitir el contenido de forma digital. De esta forma cada persona que pasa puede recoger el folleto con su dispositivo móvil o puede leer todo el periódico desde su teléfono

[3] InfoSkin : http://www.autoassembly.com/infoskin-wearable-nfc-tag/

inteligente. Desde otro punto de vista, también se puede utilizar en carteles que anuncien un concierto (en los que se podría obtener información y enviar a una web para comprar una entrada), un espectáculo e incluso para caridad (que a través del cartel se pueda hacer una donación, vía ***paypal***[4]).

- **Control de pacientes en un hospital:** El control de pacientes en un hospital requiere una importante cantidad de información a la que deben poder acceder los médicos. Gracias a NFC es posible que el médico llegue a la habitación del paciente y mediante un dispositivo pueda conocer la situación del paciente y su identificación.
- **Control de identificación de usuarios:** NFC está siendo ya utilizando como sistema de identificación en edificios de oficinas, donde los empleados acceden a la oficina con validación directa desde su teléfono inteligente.
- **Utilización en vehículos:** Cuando se combina toda la potencia de NFC en un lugar como es el coche, se pueden conseguir efectos realmente impactantes. Por ejemplo, ya es posible abrir y cerrar un coche con NFC, algunas empresas como Hyundai, ya lo han logrado implementar. Es posible imaginar ciertas cuestiones como el hecho de adaptar el coche a la persona que conduce. Esto es de la siguiente manera: una persona entra al coche y a través de su teléfono inteligente transmite al vehículo una posición del sillón, de los espejos retrovisores y también la posición del volante. Son los teléfonos móviles los que se ocupan de mandar la orden al coche para que se adapte al usuario.
- **Datos de localización:** En lugares abiertos, la utilización de Internet y un GPS consiguen unos resultados realmente impactantes (saber dónde estamos, qué lugares hay cerca, información en tiempo real, etc.). Gracias a NFC se podrían mejorar algunos aspectos, sobre todo en lugares en los que no se tiene visión directa al cielo. El mejor ejemplo es el de un museo de arte, en el que se puede ir caminando y con la utilización del teléfono móvil o teléfono inteligente se podría ir conociendo cada uno de los cuadros, su historia, su explicación e incluso NFC podría reemplazar al auricular tradicional. Actualmente algunos museos en el mundo están utilizando esta tecnología como es el caso del museo de Londres con una iniciativa promovida por el propio museo en colaboración con Nokia.

[4] PayPal: https://www.paypal.com/es/home

1.4 Tecnologías utilizadas

El uso de los teléfonos móviles se ha generalizado de tal manera en nuestra sociedad que hoy en día la gran mayoría de la población dispone de uno o más teléfonos móviles de última generación.

1.4.1 Android

Android es un sistema operativo que se emplea en dispositivos móviles basado en el núcleo de Linux.

Este sistema operativo es código libre, y para su desarrollo se utiliza el lenguaje de programación que tiene el mismo nombre. Este lenguaje está basado en el lenguaje de programación de Java y tiene una serie de herramientas de desarrollo llamadas kit de desarrollo software, más conocidas por sus siglas SDK (del inglés *Software Development Kit*).

1.4.1.1 Aplicaciones Android

La mayoría de las aplicaciones Android [3] se encuentran disponibles en ***Google Play***[5], aunque también existen desarrolladores que crean aplicaciones que se pueden encontrar en sitios web independientes. En este último caso, es recomendable estar seguro de que el sitio web en cuestión es confiable para evitar la instalación de virus o malware en general en los dispositivos Android.

En la actualidad existe un gran número de desarrolladores que escriben aplicaciones específicas para la plataforma Android hasta el punto de que el número de aplicaciones que puedes encontrar actualmente supera las 400.000.

1.4.1.2 Características de Android

Los móviles Android son baratos, si buscas un dispositivo barato seguramente lo encuentres en el mundo de Android, pero si buscas un dispositivo caro con más prestaciones también lo encontrarás.

Android puede personalizarse ampliamente por lo que cada fabricante puede adaptarlo como mejor le parezca y que los programadores tienen bastante libertad de movimiento a la hora de personalizarlo.

Android es fácilmente transportable. Puedes encontrar dispositivos Android de todo tipo, desde teléfonos inteligentes y tabletas hasta reproductores multimedia.

1.4.1.3 Características de técnicas Android

Las características técnicas [4] Android son las siguientes:

- Núcleo basado en Linux.
- Máquina virtual basada en Java para la ejecución de aplicaciones.

[5] Google Play: https://play.google.com/store .

- Soporte para formatos multimedia, audio y vídeo: WebM, H.263, H.264, MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, MP3, MIDI, OggVorbis, WAV, JPEG, PNG, GIF y BMP.
- Gráficos optimizados 2D/3D.
- Soporte para Bluetooth, WIFI, NFC, 4G, LTE y 3G entre otros.
- Soporte para GPS, brújulas y acelerómetros.

1.4.2 Python

Python es un lenguaje de programación [5] independiente de plataforma y orientado a objetos, preparado para realizar cualquier tipo de programa, desde aplicaciones Windows a servidores de red o incluso, páginas web. Es un lenguaje interpretado, lo que significa que no se necesita compilar el código fuente para poder ejecutarlo, lo que ofrece ventajas como la rapidez de desarrollo e inconvenientes como una menor velocidad.

En los últimos años el lenguaje se ha hecho muy popular, gracias a varias razones como:

- La cantidad de librerías que contiene, tipos de datos y funciones incorporadas en el propio lenguaje, que ayudan a realizar muchas tareas habituales sin necesidad de tener que programarlas desde cero.
- La sencillez y velocidad con la que se crean los programas. Un programa en Python puede tener de 3 a 5 líneas de código menos que su equivalente en Java o C.
- La cantidad de plataformas en las que podemos desarrollar, como Unix, Windows, OS/2, Mac entre otros.
- Además, Python es gratuito, incluso para propósitos empresariales.

1.4.2.1 Características del lenguaje

Es de propósito general, es decir se pueden crear todo tipo de programas. Es multiplataforma ya que existen versiones disponibles de Python en muchos sistemas. Originalmente se desarrolló para Unix, aunque cualquier sistema es compatible con el lenguaje siempre y cuando exista un intérprete programado para él.

La programación orientada a objetos está soportada en Python y ofrece en muchos casos una manera sencilla de crear programas con componentes reutilizables. Dispone de muchas funciones incorporadas en el propio lenguaje, además, existen muchas librerías que podemos importar en los programas para tratar temas específicos.

1.4.3 Django.

Django es un entorno de desarrollo web [6] de código abierto, escrito en Python, que respeta el patrón de diseño conocido como **Modelo–vista–controlador**, y fue liberada al público bajo una licencia BSD en julio de 2005.

La meta fundamental de Django es facilitar la creación de sitios web complejos. Django pone énfasis la reutilización, la conectividad y extensibilidad de componentes, el desarrollo rápido y el principio No te repitas.

Python es usado en todas las partes del entorno, incluso en configuraciones, archivos y en los modelos de datos.

Proporciona una serie de características que facilitan el desarrollo rápido de páginas orientadas a contenidos. Por ejemplo, en lugar de requerir que los desarrolladores escriban controladores y vistas para las áreas de administración de la página, Django proporciona una aplicación incorporada para administrar los contenidos, que puede incluirse como parte de cualquier página hecha con Django y que puede administrar varias páginas hechas con Django a partir de una misma instalación.

La aplicación administrativa permite la creación, actualización y eliminación de objetos de contenido, llevando un registro de todas las acciones realizadas sobre cada uno, y proporciona una interfaz para administrar los usuarios y los grupos de usuarios.

La distribución principal de Django también aglutina aplicaciones que proporcionan un sistema de comentarios, herramientas para publicar contenido vía RSS y/o Atom, "páginas planas" que permiten gestionar páginas de contenido sin necesidad de escribir controladores o vistas para esas páginas, y un sistema de redirección de URLs.

1.4.3.1 Características de Django

De las características que nos brinda el entorno Django, podemos encontrar:

- Una interfaz de administración la cual nos permite administrar los modelos de datos sin tener que programarlos antes.

- Dispone de una API[6] (en inglés, *Application Programming Interface*) que nos permite trabajar con distintas bases de datos

- Incorpora un modelo objeto-relacional más conocido por sus siglas ORM, (en inglés, *Oriented Relational Model*).

- Dispone de un sistema de plantillas el cual esta implementado en un lenguaje de etiquetas con herencia.

[6] API: son un conjunto de comandos, funciones y protocolos informáticos [7] que permiten a los desarrolladores crear programas específicos para ciertos sistemas operativos. Permiten usar funciones predefinidas para interactuar con el sistema operativo o con otro programa.

• Un ***dispatcher***[7] de URLs el cual está basado en expresiones regulares.

•Nos permite añadir funcionalidades adicionales ya que dispone de un sistema de middleware.

•Soporta muchos lenguajes.

•Dispone de documentación extensa y fácilmente accesible.

1.4.4 NFC

Las siglas de NFC hacen referencia a su nombre en inglés *Near Field Comunication* que traducido al español significa comunicación de campo cercano

Es una tecnología de comunicación inalámbrica, de corto alcance, ya que los dispositivos pueden estar como máximo a como unos 4 cm, para permitir el intercambio de datos entre dispositivos.

Funciona en la banda [8] de los 13.56 MHz y no es necesario la solicitud de licencia para usarla, nos proporciona unas velocidades de transmisión de 106 Kbit/s, 212 Kbit/s y424Kbit/s. En el año 2004 se fundó el **NFC Fórum**[8] quienes fueron los encargados de establecer las bases de esta tecnología inalámbrica. Las especificaciones técnicas están estandarizadas en la ISO/IEC 14443 e ISO/IEC 18092.

NFC Forum define dos modos de funcionamiento que deben soportar todos los dispositivos NFC:

- **Activo**: los dos dispositivos que intervienen en la comunicación generan un campo electromagnético, que usaran para transmitir los datos.
- **Pasivo**: el campo electromagnético lo crea únicamente el dispositivo receptor y el dispositivo emisor utiliza ese campo electromagnético para enviarle sus datos.

En este proyecto el modo de funcionamiento que se ha utilizado es el pasivo, ya que es el teléfono móvil el que va a crear ese campo electromagnético y la tarjeta NFC la que va enviar sus datos a través del campo electromagnético anteriormente creado.

1.4.4.1 Formato de datos NDEF

Los mensajes NDEF [10] proporcionan un método estandarizado para que un lector se comunique con un dispositivo NFC. El mensaje NDEF contiene varios registros, como se muestra. Recibe soporte NDEF sólo cuando trabaja con etiquetas estandarizadas - las etiquetas propietarias típicamente no proporcionan este soporte. El estándar NFC admite cinco tipos de etiquetas, todas las cuales soportan el mismo formato de mensaje NDEF.

---

[7]Dispatcher: es una parte de un programa [9] la cual se encarga de lanzar un proceso en el servidor cuando se está utilizando un entorno cliente/servidor.

[8]NFC FORUM: http://www.nfc-forum.org/home .

Cada registro contiene un encabezado y una carga útil. El encabezado contiene información útil para el lector, como el ID de registro, su longitud y tipo. El tipo define el tipo de carga útil que contiene el registro. La carga útil es simplemente datos.

1.4.4.2 Tipos de etiquetas NFC

El **NFC Forum** ha redactado una serie de especificaciones [11] para asegurar que los fabricantes de etiquetas y lectores mantengan la interoperabilidad.

El foro ha resumido especificaciones para cuatro tipos diferentes de etiquetas, cada una con diferentes características y tamaños de carga útil:

- NFC Forum Tipo 1 Etiqueta: Los usuarios pueden configurar la etiqueta para que sea de sólo lectura. La disponibilidad de memoria es de 96 bytes y se puede ampliar a 2 Kbyte.
- NFC Forum Tipo 2 Etiqueta: Los usuarios pueden configurar la etiqueta para que sea de sólo lectura. La disponibilidad de memoria es de 48 bytes y se puede ampliar a 2 Kbyte.
- NFC Forum Tipo 3 Etiqueta: El fabricante puede configurar la etiqueta para que sea de sólo lectura. Disponibilidad de memoria de hasta 1MByte.
- NFC Forum Tipo 4 Etiqueta: El fabricante puede configurar la etiqueta para que sea de sólo lectura. Disponibilidad de memoria de hasta 1MByte.
- NFC Forum Tipo 5 Etiqueta: El fabricante puede configurar la etiqueta para que sea de sólo lectura. Disponibilidad de memoria de hasta 1MByte.

Cualquier dispositivo certificado por **NFC Fórum** está garantizado para trabajar con estos cinco tipos de etiquetas.

2 MATERIALES Y METODOS

El servicio desarrollado en el presente trabajo fin de grado consta de dos partes, una aplicación Android, que se ejecutaría en el terminal que haría las veces de cerradura virtual o control de acceso de cada estancia del edificio y por otra parte, la aplicación servidora que es la encargada de almacenar y gestionar la información del sistema. Desde esta última se podrá establecer las medidas de control de acceso del sistema.

Cada usuario que tenga acceso a una o varias estancias del recinto debe de disponer de una tarjeta NFC para poder identificarse.

2.1 Arquitectura del sistema

La infraestructura que ha sido necesaria implementar para el despliegue del sistema está compuesta de los siguientes elementos:

- **Aplicación servidora**: está desarrollada en Python, haciendo uso de su entorno Django. Es el centro de control de todo el sistema. Recibe peticiones de acceso mediante `HTTPS` y según la configuración que tenga cada usuario responderá con un código 200 (ok) u otro código distinto denegando el acceso.
- **Base de datos**: se ha utilizado el sistema gestor de base de datos relacionales *SQLite*. Es la encargada de almacenar los datos personales de cada empleado y las dependencias en las que cada uno de ellos tiene acceso. Para los accesos a la aplicación se podrá definir el perfil de cada uno de los usuarios autorizados. Se ha decidió utilizar este sistema gestor de base de datos, porque para desarrollar el proyecto no es necesario almacenar gran cantidad de datos.
- **Conexión a Internet**: para la comunicación entre los teléfonos móviles y la aplicación servidora. Debido a que la aplicación servidora está alojada en un **alojamiento web** (en inglés, *hosting*) se podrá acceder a él desde cualquier dispositivo que disponga de conexión a Internet. Esto nos garantiza que el sistema no tiene por qué ser un sistema fijo, sino que tiene movilidad, ya que cualquier teléfono móvil con acceso a Internet y con la aplicación instalada podrá hacer uso de él.
- **Dispositivo Android:** es necesario que el terminal de cada dependencia disponga de sistema operativo Android y que lleve incorporada la tecnología NFC.

A continuación, se presenta una representación grafica de la arquitectura del sistema en la *Figura 2. Arquitectura del Sistema.*.

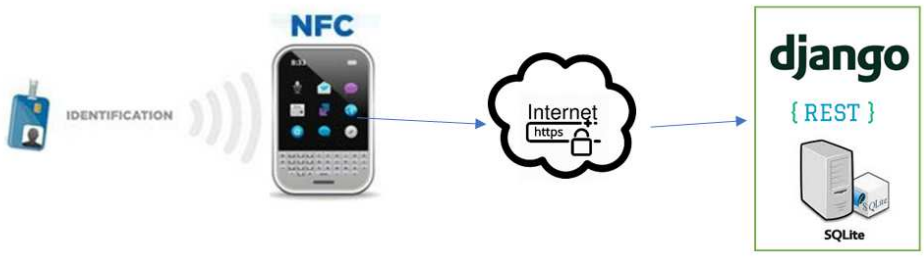

Figura 2. Arquitectura del Sistema.

2.1.1 Base de datos

El gestor de base de datos que se ha utilizado, como anteriormente se ha indicado, es *SQLite*, este gestor lo incorpora de forma nativa Django.

La composición de la base de datos del proyecto es la siguiente:

- La implementación del sistema ha requerido crear 4 tablas (ver *Figura 3.Modelo de datos creado para el proyecto NFC*).
- La APP ***[9]Admin View Permission***[10] de Django se ha utilizado para administrar los permisos, crea 6 tablas.
- Django consta de 6 tablas para gestionar el sistema gestor de base de datos.

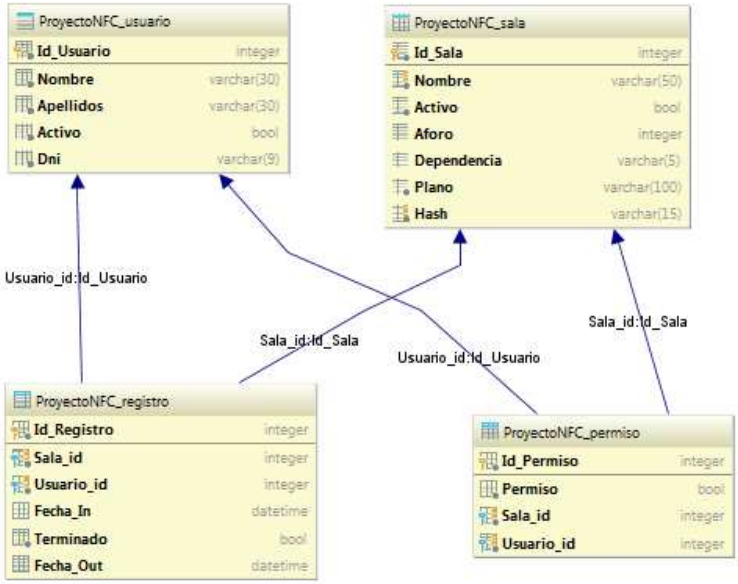

Figura 3.Modelo de datos creado para el proyecto NFC.

[9] APP: es un programa informático ligero [12], descargable que permite unas funciones determinadas, normalmente diseñado para equipos móviles: teléfonos móviles y tabletas.

[10] Documentación: http://django-admin-view-permission.readthedocs.io/en/latest/.

Las tablas que se han creado para establecer el modelo de datos de la aplicación son las siguientes:

- **Usuario**: almacena los datos personales de los usuarios que van a identificarse en la aplicación móvil utilizando las tarjetas NFC. También permite almacenar si un usuario esta activo o no.
- **Sala**: almacena los datos de identificación de la sala y el aforo del que dispone cada una, así como otros datos descriptivos de las salas. Permite almacenar el estado de una sala, para saber si esta activa o no.
- **Permiso**: almacena las salas en la que los diferentes usuarios tienen permiso para entrar.
- **Registro**: almacena los registros de entrada y salida de todos los usuarios,

Las tablas que almacenan los datos de autentificación de la aplicación servidora son:

- **Auth_user:** almacena los datos personales de los usuarios que pueden acceder a la aplicación servidora y pueden iniciar y configurar la aplicación móvil.
- **Auth_group:** almacena el nombre de los grupos de usuarios creados en la aplicación servidora.
- **Auth_group_permissions:** almacena los permisos de los grupos que hay creados en la aplicación servidora.
- **Auth_permissions:** almacena los tipos de permisos que se pueden establecer.
- **Auth_user_groups:** almacena los usuarios que pertenecen a los distintos grupos.
- **Auth_user_user_ permissions:** almacena los permisos que tiene cada usuario.

Entre las que crea Django cabe destacar las siguientes:

- **Django_admin_log:** en la que contiene un log de todas las modificaciones que se han producido en la base de datos.
- **Django_migrations**: en la cual se registra cualquier modificación del modelo de datos que se realice.

Al usar *SQLite* no es necesario crear una base de datos, porque *SQLite* usa un archivo autónomo sobre el sistema de archivos para guardar los datos.

La configuración de la base de datos se encuentra en el archivo de configuración de Django, llamado, `settings.py`. Hay que editar la variable `DATABASES`, en la opción

de ENGINE se indica cual es el gestor de base de datos a utilizar y en la opción NAME se indica la ruta y el nombre de nuestra base de datos.

Django dispone de la librería models [13] la cual corresponde a la capa de datos de la aplicación.

En el archivo models.py y haciendo uso de la librería de Django models, se modela los datos en la base de datos, representada como código de Python.

Cada clase definida en models.py corresponde a una tabla de la base de datos y cada atributo corresponde a una columna de esa tabla.

Cada atributo contiene el nombre de la columna, el tipo de campo que almacena y las características que tiene ese campo. Por ejemplo, si se quisiera crear una columna que se llame nombre, que sea de tipo cadena de caracteres, con una longitud máxima de 50 caracteres y que sea única en mi modelo se hace de la siguiente forma:

Nombre = models.CharField (max_length=50, unique=True).

Para crear relaciones entre las tablas solo hay que poner como tipo de campo de esa columna ForeignKey e indicar de qué tabla se hereda.

Por último, para que Django cree o modifique el modelo de datos que se ha definido se debe ejecutar en la terminal los siguientes comandos:

- python manage.py makemigrations: para construir los modelos.
- python manage.py migration: para guardar los modelos en la base de datos.

Para trabajar con la capa de datos Django se dispone de una API Python. Para utilizar esta API es necesario importar la clase del modelo que se va a utilizar y se trabaja con los modelos como si fueran objetos.

A continuación, se detallan algunos ejemplos de cómo utilizar este API:

- Para consultar todos los datos de una tabla:
  - Nombre del modelo.objects.all ().
- Consultar los datos que dispongan algún atributo en concreto:
  - Nombre del modelo.objects.get (atributo=datos a consultar).
- Para guardar datos en la base de datos se utiliza la sentencia save(),de la siguiente forma:
  - Nombre del modelo (Nombre de la columna=nuevos datos)
  - Nombre del modelo save ().

2.1.2 Aplicación servidora

En la construcción de la aplicación servidora se ha desarrollado con Django como sea indicado anteriormente.

En este desarrollo se han utilizado las librerías y las API que Django ofrece de manera gratuita. Para la implementación de estas funcionalidades se ha utilizado `pip`[11], Django trabaja con el patrón de diseño *MVC* (*Modelo-Vista-Controlador*), solo que en Django este modelo se denomina *MTV* (*Modelo-Template-Vista*), el esquema que sigue este modelo se puede observar en la *Figura 4 Esquena MVT*.

Las siglas MTV hacen referencia a las siguientes capas de diseño:

- **M** hace referencia a "Model" (*Modelo*), corresponde con la capa de acceso a la base de datos. Esta capa contiene toda la información sobre los datos: cómo acceder a ellos, cómo validarlos, cuál es el comportamiento que tiene, y las relaciones entre ellos.
- **T** hace referencia a las plantillas (en inglés, *Template*), que corresponde con la capa de presentación. Esta capa contiene los archivos que pertenecen a la presentación de nuestros datos, como van a ser mostrados los datos de nuestra aplicación.
- **V** significa "View" (*Vista*), corresponde a la capa de la lógica de negocios. Contiene la lógica que accede al modelo y la delega a la plantilla apropiada.

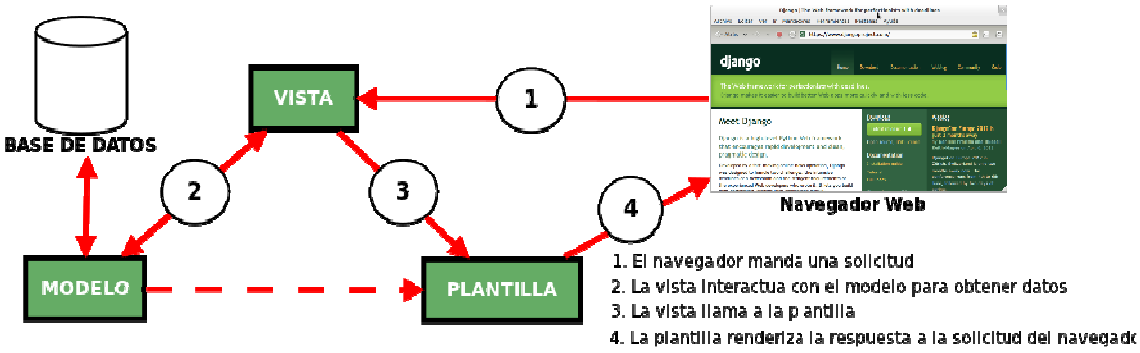

Figura 4 Esquena MVT[12].

Cuando se hace la instalación de Django se crea un directorio de trabajo compuesto por varios archivos (ver *Figura 5.Directorio de trabajo*).

[11] PIP es un sistema de gestión de paquetes que permite instalar paquetes desarrollados en Python.

[12] Disponible en: http://www.maestrosdelweb.com/images/2012/04/esquema-mtv.png.

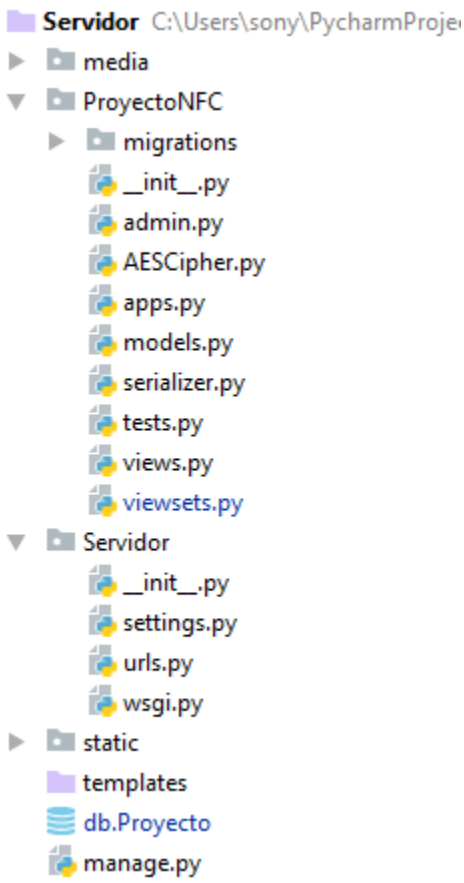

Figura 5.Directorio de trabajo.

La funcionalidad que tienen algunos de estos archivos es la siguiente:

- `Servidor/`: es el directorio que contiene nuestro proyecto.
- `manage.py`: sirve para interactuar con los comandos administrativos de nuestro proyecto.
- `Servidor/Servidor/`: es el directorio que contiene el código Python del proyecto.
- `_init__.py`: es un archivo que se requiere para que Python utilice el directorio `Servidor` como un paquete. En este archivo habitualmente no tiene que ser modificado.
- `settings.py`: es el archivo de configuración del proyecto.
- `urls.py`: es el archivo donde se declaran las direcciones que apuntan a los distintos recursos del proyecto.
- `wsgi.py`: este archivo es el punto de entrada *WSGI*[13] para el alojamiento web, donde está desplegada la aplicación servidora del proyecto.

[13] Documentación en : https://docs.djangoproject.com/en/1.11/howto/deployment/wsgi/.

- `view.py`: es el archivo que por defecto crea Django para crear las vistas de la aplicación.
- `models.py`: es el archivo donde se definen los modelos de la base de datos.
- `admin.py`: es el archivo donde se controla los modelos que van a formar parte de la interfaz de administración.

2.1.2.1 Desarrollo de la aplicación servidora

Para el desarrollo la aplicación servidora se ha utilizado la plantilla de administración que Django incorpora.

Esta interfaz está basada en HTML y permite a los usuarios autorizados agregar, editar y modificar los parámetros de configuración del sistema de control de acceso.

La aplicación servidora es el núcleo de control del sistema de acceso, para que esté funcione correctamente es necesario configurar los siguientes parámetros en el sistema:

-**Modelo de salas:** hay que indicar el nombre de la sala, el número de IMEI del teléfono móvil que va a controlar esta sala, el aforo máximo que admite, el aforo actual, si está activa o no, el plano de las medias de seguridad y el número de dependencia al que pertenece la sala (ver *Figura 58.Vista para añadir y modificar datos de salas*).

La sala solo se puede identificar de forma inequívoca a través del número de IMEI, esta medida añade seguridad al sistema, ya que en todo momento se controla que el dispositivo es el que pertenece a cada sala.

El campo de activo permite que, en cualquier momento de forma sencilla, se pueda interrumpir la actividad en una sala.

Con el campo de aforo en todo momento se puede controlar el número de personas que hay en el interior de una sala. Su valor inicial debe ser 0 cuando la sala está vacía.

El plano de la sala debe mostrar las indicaciones de seguridad (extintores, salidas de emergencia, teléfonos, vías de escape rápido), de esta forma se contribuye a la aplicación de medidas de seguridad dentro de las estancias, ya que podrá ser consultado por el usuario en todo momento en la aplicación móvil.

- **Modelo de usuarios:** hay que indicar el nombre, apellidos y el DNI de cada uno de los usuarios que se quiera identificar en la aplicación. También se puede elegir el estado del usuario, si esta activo o no. El DNI es el dato que se utiliza para identificar a los usuarios de forman inequívoca.

- **Modelo permisos:** en este apartado se puede configurar que usuarios pueden acceder a una determinada sala. Con el campo permiso se puede elegir fácilmente si ese permiso está vigente o no.

- **Modelo Registros:** se ha creado para llevar el control del sistema. Refleja la fecha de entrada y salida de los usuarios a las distintas salas (ver Figura 55.Vista resumen de registros.).

La interfaz es muy intuitiva y permite de forma sencilla administrar el sistema creando, consultando o modificando los datos de la misma.

En todos los modelos está habilitado el servicio para que se puedan hacer búsquedas de los datos que contienen.

La interfaz de administración de Django es solo una parte del paquete de funcionalidades llamado **`django.contrib`** que Django incorpora.

Para activar esta interfaz es necesario añadir las siguientes dependencias en la variable `INSTALLED_APPS` del archivo de configuración `settings.py`:

- `'django.contrib.admin'`
- `'django.contrib.auth'`
- `'django.contrib.contenttypes'`
- `'django.contrib.sessions'`
- `'django.contrib.messages'`

En la variable `TEMPLATES`, del mismo archivo, hay que incluir en el apartado de `OPTIONS` la librería `'django.contrib.messages.context_processors.messages'`.

También hay que añadir en la variable `MIDDLEWARE` las librerías `'django.contrib.auth.middleware.AuthenticationMiddleware'y'django.contrib.messages.middleware.MessageMiddleware'`.

Para poder crear el usuario administrador hay que ejecutar el siguiente comando en la terminal desde la raíz de nuestro programa:

```
python manage.py createsuperuser
```

El sistema pide un nombre de usuario, un email y una contraseña para dar de alta el usuario administrador.

Para cambiar el idioma de esta plantilla al español, en la variable `LANGUAGE_CODE`, del archivo `settings.py`, se debe indicar el código del idioma correspondiente al español:

`LANGUAGE_CODE` = "es-ES".

También hay que incluir la librería `'django.middleware.locale.LocaleMiddleware'` en a la variable `MIDDLEWARE_CLASSES`.

Para que esta plantilla sea la página principal de la aplicación servidora, en el archivo de configuración `URL.py`, se debe hacer una redirección de la página principal a la página de administración de la siguiente forma:

url (**r'^'**, admin.site.urls).

Para que en la plantilla de administración se muestren los modelos que se han creado en el archivo `MODEL.py`, estos se deben de definir en el archivo `ADMIN.py`. Para hacerlo el método que se utiliza es: `admin.site.register` (Nombre del modelo).

Para personalizar la plantilla que se muestra en la aplicación servidora (ver *Figura 6.Interface de Administración.*), Django nos ofrece herramientas que nos permiten añadir funcionalidades especiales, se han utilizado son las siguientes:

- `list_display`: permite elegir cuales son las columnas del modelo que deseas mostrar en la interfaz.
- `search_field`: permite hacer búsquedas sobre las columnas que se incluyan en esta variable
- `list_filter`: permite hacer filtros de búsqueda en los campos de fechas, para buscar los cambios que se hayan producido hoy, hace 7 días, etc.
- `list_editable`: permite elegir las columnas que quieres que sean modificables desde la interface.

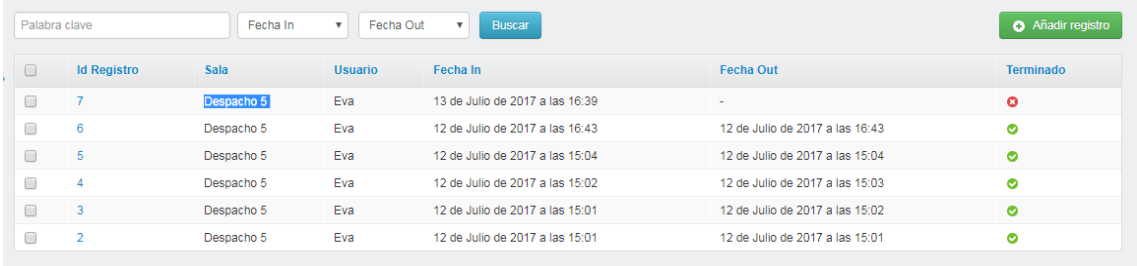

Figura 6.Interface de Administración.

Con el fin de separar los usuarios que intentan identificarse en la aplicación móvil de los usuarios que van administrar la aplicación, se ha optado por la instalación de la APP de autentificación de usuarios que Django ofrece, denominada ***Admin View Permission***. Esta herramienta permite administrar usuarios, grupos, permisos y sesiones de usuario.

Con esta herramienta se puede dar de alta y gestionar a los usuarios de seguridad y en función de su perfil, se pueden determinar que funciones van llevar a cabo o no en la aplicación servidora.

Este sistema de autentificación está formado por:

Usuarios: en este apartado se dan de alta a los usuarios de seguridad del sistema, los cuales llevaran a cabo la administración la aplicación servidora y la puesta en marcha de la aplicación móvil.

Cuando se crea un usuario nuevo hay que definir su nombre de usuario y una contraseña. Esta contraseña es automáticamente cifrada usando el algoritmo `pbkdf2_sha256`.

Se pueden añadir datos adicionales de los usuarios como son, nombre, apellidos o email.

Dentro de la opción de permisos, se pueden personalizar los permisos para cada uno de los usuarios de seguridad que pertenecen al sistema. Las opciones que se pueden personalizar son las siguientes:

- Activo: este campo permite activar o desactivar un usuario, facilita la gestión de usuarios, ya que si un usuario deja de pertenecer a la organización únicamente se debe desactivar y dejara de tener acceso.
- Staff: si está activado este campo el usuario podrá acceder al sitio web, si no es así no podrá acceder. Si esta seleccionado puede acceder a la aplicación servidora, sino lo está solo podrá administrar la aplicación móvil.
- Superusuario: cuando un usuario es marcado como supe usuario se le otorga automáticamente acceso a que realice cualquier modificación en la aplicación servidora.
- Grupos: permite introducir a los usuarios dentro de grupos con perfiles de permisos ya establecidos, para facilitar la gestión de los usuarios de seguridad.
- Permisos: se puede elegir de forma manual y muy detalladamente a que va a tener permiso dentro de la aplicación servidora cada usuario. Los permisos que se pueden otorgar son de creación, visualización, edición y borrado de cada uno de los datos de los modelos que conforman la aplicación servidora. Aparte permite establecer permisos referentes a los usuarios y grupos de usuarios de seguridad, gestión de log del sistema, capacidad de establecer permisos a otros usuarios y por último permisos referentes al manejo de sesiones que controla Django.
- Por último se pueden fijar las fechas de creación y de último inicio de sesión del usuario.

Grupos: en este apartado se pueden crear grupos y definirles unos permisos determinados. Esta opción facilita la gestión de permisos, ya que se puede crear perfiles

predeterminados de acceso a la aplicación servidora e ir incluyendo a los distintos usuarios dentro de esos grupos.

Para la configuración de API en Django se debe verificar que, en el archivo de configuración `settings.py`, la variable `INSTALLED_APP` contiene las siguientes dependencias: `'django.contrib.auth'` y `'django.contrib.contenttypes'`

Por último en la variable de `MIDDLEWARE` deben estar los valores `SessionMiddleware` y `AuthenticationMiddleware`

2.1.2.2 Diseño de la aplicación servidora

Para el diseño de la aplicación servidora se ha hecho uso de la plantilla gratuita **Django Suit**[14], que permite modificar el diseño que la plantilla de administración de Django incorpora por defecto.

La implementación de esta plantilla se lleva a cabo de la siguiente forma:

- Instalación de la versión 0.2.25 que es la más estable al día de hoy.

  ```
  pip installdjango-suit==0.2.25
  ```

- En el archivo de configuración `settings.py` en la variable `INSTALLED_APP` hay que incluir la dependencia `'suit'`, siempre después de la dependencia que hacer referencia a la plantilla de administración que se llama `'django.contrib.admin'`.
- En el apartado `TEMPLATES` hay que añadir para la variable de `OPTIONS` los siguientes contextos:
  - 'django.template.context_processors.debug',
  - 'django.template.context_processors.request'
  - 'django.contrib.auth.context_processors.auth',
  - 'django.contrib.messages.context_processors.messages'

2.1.2.3 Comunicación entre aplicaciones

Para la comunicación entre la aplicación servidora y la aplicación móvil se ha implementado un servicio `REST`.

Para el desarrollo de este servicio se ha utilizado el entorno de trabajo desarrollado para Django denominado **Rest-Framework**[15].

La instalación de este entorno de trabajo se realiza de la siguiente forma:

- Instalación del entorno de trabajo utilizando el comando para pip:

  ```
  pip install djangorestframework
  ```

- En el archivo de configuración `settings.py` en el apartado de `INSTALLED_APP` hay que incluir la dependencia `'rest_framework'`.

[14] Django Suit: http://djangosuit.com/ .
[15] Rest-framework: http://www.django-rest-framework.org/ .

- En el mismo archivo `settings.py` hay que crear la variable `REST_FRAMEWORK`. En esta variable se deben definir los permisos de los usuarios que pueden consumir el servicio `REST`, en este caso como lo consume una aplicación móvil y no es necesaria autenticación previa para utilizarla, se ha optado por que lo pueda consumir cualquier persona.
- la variable `REST_FRAMEWORK` también hay que determinar qué tipo de peticiones acepta el servicio, en este caso las peticiones usan el formato JSON.
- Por último en el archivo `URL.py`, hay que crear la redirección a cada método, esto se hacer indicando el nombre que se haya elegido para llamar al servicio `REST` y se le redirige al método que se haya creado en el archivo `wiewsets.py`.

  Por ejemplo la llamada al método `UsuarioViewSet`, se realiza de la siguiente forma:

  ```
  url (r'^rest_usuario/$', viewsets.UsuarioViewSet)
  ```

La configuración de la variable `REST_FRAMEWORK` se puede observar en la Figura 7.Configuración Variable REST_FRAMEWORK.

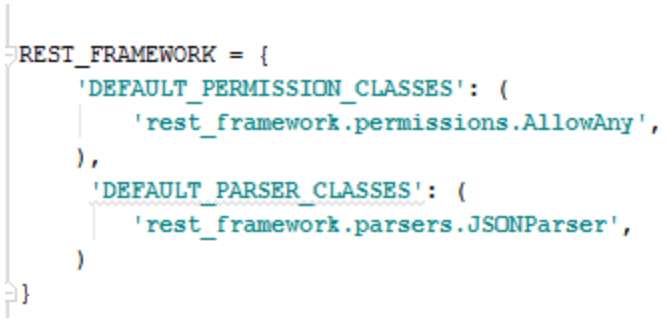

Figura 7.Configuración Variable REST_FRAMEWORK.

La implementación del servicio `REST` se ha realizado en el archivo `wiewsets.py`. Para poder responder todas las peticiones que necesita la aplicación móvil, se han programado 5 peticiones `REST`, las cuales están implementadas en los siguientes métodos:

`UsuarioViewSet:` este método soporta peticiones tanto `GET` como `POST`. La petición `GET` únicamente va a devolver todos campos de la tabla Usuarios. Las peticiones que se van a recibir desde la aplicación móvil van a ser de tipo `POST`.

La aplicación móvil utiliza este método cuando desea verificar la identidad de un usuario que intenta identificarse con su tarjeta NFC.

La aplicación móvil hace una petición de tipo `POST` a este método, en la cual le envía cifrados los datos del id de la sala a la que pertenece y el DNI del usuario que ha hecho la lectura de su tarjeta identificativa.

En todo momento la aplicación servidora devuelve a la aplicación móvil, un mensaje JSON que incluye el código de respuesta y un mensaje con el resultado de la petición, para que en la aplicación móvil se pueda mostrar al usuario el estado de su solicitud.

A partir de estos datos va a realizar las siguientes comprobaciones en la base de datos de la aplicación servidora:

1. Comprueba si existe el usuario en la tabla Usuarios. Si no existe el usuario en la tabla se le envía un mensaje JSON con el código de respuesta 401 y el mensaje de error "Usuario no existe".
2. Si el usuario existe comprueba si ese usuario esta marcado como activo en la tabla Usuarios. Si no está activo el mensaje de respuesta contendrá el código de respuesta 401 y el mensaje de error "El usuario esta desactivado".
3. Si el usuario esta activo se comprueba si en la tabla Permisos esta dado de alta ese usuario en esa sala. Si no está dado de alta en la tabla Permisos la aplicación servidora contestara a la aplicación móvil con un mensaje que incluye el código de respuesta 404 y el mensaje de error "Usuario no tiene permiso para entrar en esta Sala".
4. Si el usuario tiene permiso para entrar en esa sala, se comprueba en la tabla Permisos que el permiso de ese usuario en esa sala está activo. Si no está activo el permiso la aplicación servidora responde con un mensaje JSON que contendrá el código de respuesta 403 y el mensaje de error "Permiso del usuario desactivado".
5. Si el permiso esta activos se comprueba en la tabla Salas que la sala este activa. Si la sala no está activa se le devuelve a la aplicación móvil el código de error 404 y el mensaje de error "Sala no está activa".
6. Si la sala esta activa se comprueba en la tabla de registros si hay un registro anterior de ese usuario en esa sala.
7. Según lo que devuelva la consulta en la tabla de registros se pueden dar 2 situaciones:

    a) Hay un registro anterior pero con el valor de la columna de Fecha Out vacio, por lo que se trata de un registro de salida ya que el usuario está intentando salir de la dependencia. En este caso se completa la entrada creada previamente en la tabla Registro,

incluyendo la fecha de salida y terminando el registro. También registra la salida en la tabla Salas disminuyendo el aforo. Por último responde a la aplicación móvil con el código de respuesta 200 y el mensaje 'Hasta Pronto'

b) Puede que haya o no registros anteriores, si hay registros anteriores estos ya están terminados. Este caso indica que el usuario está intentado entrar en la dependencia, para permitirle el paso o no, primero hay que consultar el aforo de la sala.

8. Si se trata de una solicitud de entrada se hace una consulta a la tabla Salas para confirmar si el aforo supera el aforo máximo de la dependencia. Si el aforo no permite la entrada de mas usuarios se responde a la aplicación móvil con el código de respuesta 403 y el mensaje "Sala completa".
9. Si el aforo no está completo se registra la entrada en la tabla Registros, se actualiza el aforo de la sala y se envía un mensaje con el código de respuesta 200 y el mensaje 'Recuerde pasar su credencial por este punto al salir'.

SalaViewSet: este método también soporta peticiones del tipo `GET` y `POST`. Cuando este método recibe una petición `GET` devuelve todos los valores que están almacenados en la tabla Salas.

La aplicación móvil, para identificar la sala, hace una petición `POST` a este método, en esta petición envía cifrado el IMEI del terminal donde está instalada la aplicación. Una vez descifra el dato, si es posible descifrarlo, se consulta en la tabla Salas si ese IMEI está almacenado.

Si el IMEI no está almacenado en la tabla, la aplicación servidora responde con un mensaje de error a la aplicación móvil con el código de error 404 y el mensaje "El IMEI no existe en la base de datos".

Si el IMEI está almacenado, se pasa a comprobar si la sala tiene activo o no el campo activo en la tabla Salas.

Si la sala esta activa el mensaje de respuesta de la aplicación servidora incluye todos los datos que contienen la sala identificada y el código de respuesta 200.

Si no la sala no está activa el mensaje de respuesta de la aplicación servidora contendrá el código de error 404 y el mensaje "La sala no está activa."

LoginViewSet: este método solo soporta peticiones `POST`, va a recibir una petición `POST` por parte de la aplicación móvil para identificar al usuario en la pantalla de registro.

Los datos que la aplicación servidora debe recibir son el `username` y el `password` del usuario, estos datos como en todas las peticiones llegan cifrados a la aplicación servidora.

Este método va intentar descifrar los datos y si puede, realiza una consulta al sistema de autentificación de usuarios que incorpora Django. Para hacer esta consulta se usa el método **`authenticate`** y se le pasan como parámetros el `username` y el `password` del usuario. También se consulta si el usuario de seguridad está activo o no.

Si el usuario o la contraseña no son correctos o el usuario no está activo, se envía un mensaje de respuesta a la aplicación móvil con el código de error 404 y el mensaje "Datos de usuario incorrectos".

Si el usuario y contraseña son correctos se verifica si el usuario esta activo o no, solo si el usuario esta activo podrá entrar en la aplicación móvil.

`ModPassViewSet`: este método recibe únicamente peticiones de tipo `POST`. Con este método desde el menú de configuración de la aplicación móvil, se podrá modificar la contraseña del usuario de seguridad.

En esta petición la aplicación servidora recibe nuevamente cifrado el `username` y el `password` que quiere modificar. Antes de modificar la contraseña se hace una consulta a la tabla User para obtener todos los datos del `username` recibido y comprobar si el usuario existe.

Si el usuario no existe la aplicación servidora responde con el código de respuesta 404 y el mensaje "Usuario no existe en la Base de Datos"

Si el usuario de seguridad existe, para modificar el `password` como se encuentra cifrado en la base de datos, para modificarlo hay que utilizar el método **`set_password`**, propio del sistema de gestión de usuarios que incorpora Django.

Si se modifica el password se envía un mensaje a la aplicación móvil que incluye el código de respuesta 200 y el mensaje "Password modificado".

.**`ModUserViewSet`**: este método se utiliza cuando se quiere cambiar desde el menú de configuración de la aplicación móvil el nombre de usuario de seguridad. Este método solo responde a peticiones `POST`.

Los datos que se reciben cifrados son el antiguo nombre de usuario y el nuevo nombre de usuario. Para realizar el cambio de nombre se vuelve hacer una consulta al sistema de gestión de usuarios que incluye Django para confirmar que existe el `username` que hemos recibido en la petición.

Si no existe ese nombre de usuario en la base de datos, la aplicación servidora informa a la aplicación móvil de este error con un mensaje de respuesta que incluye el código de respuesta 404 y el mensaje "Usuario no existe en la Base de Datos".

Si existen esos datos, comprueba que el nuevo `username` no corresponde con el de otro usuario. Si este `username` ya existe en la base de datos se envía un mensaje de respuesta a la aplicación móvil que incluye el código de respuesta 500 y el mensaje "Nombre de Usuario ya existe en la Base de Datos".

Si no existe un `username` igual en la base de datos se modifica ese dato por el nuevo nombre de usuario, que se ha introducido en el formulario del menú de configuración y se le envía como respuesta a la aplicación móvil un mensaje que incluye el código de respuesta 200 y el mensaje "Usuario modificado".

2.1.2.4 Seguridad implementada

En el desarrollo de este proyecto se ha utilizado el algoritmo de cifrado AES para cifrar los datos que se intercambian en las comunicaciones entre el servidor y la aplicación móvil.

Se han cifrado aquellos datos que son más sensibles, como son el IMEI del teléfono, que se envía durante la petición de identificar la sala, el número de DNI del usuario, que se intercambia durante el proceso de identificación de un usuario, y los datos de usuario, que hay que enviar durante la ejecución del menú de configuración y de la pantalla de inicial registro en la aplicación móvil.

En la Figura 8.Cifrado AES. se puede ver un ejemplo del cifrado de datos que se realiza, cuando un terminal solicita identificar su sala, en una petición HTTPS.

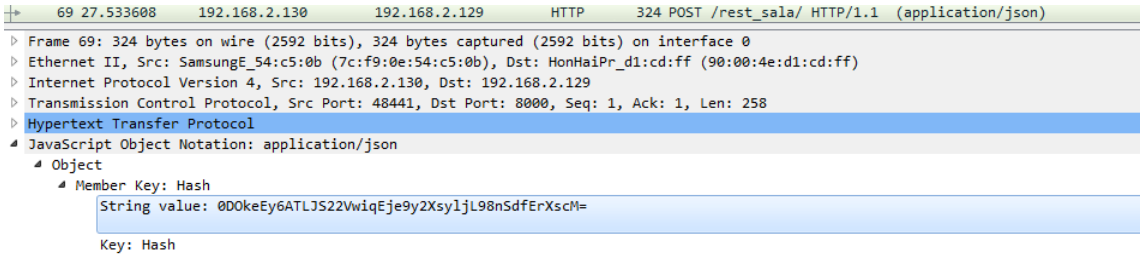

Figura 8.Cifrado AES.

Otra medida de seguridad que se ha implementado, es el uso del numero de IMEI del terminal para identificar la sala, de esta forma se han implementado una medida de seguridad a nivel físico, ya que se garantiza, que solo con un dispositivo móvil se puede hacer el registro para esa sala.

Otro aspecto que se ha tenido en cuenta es que las peticiones entre la aplicación móvil y la aplicación web se realizan bajo el protocolo `HTTPS`, que se ocupa de cifrar esa comunicación.

2.1.3 Aplicación Android

En la puerta de acceso a cada recinto, debe existir un dispositivo móvil con la aplicación desarrollada previamente instalada, para poder realizar el control de acceso a usuario de forma correcta.

La aplicación Android consta de las siguientes partes:

1. Registro: para iniciar la aplicación es necesario que un usuario de seguridad configure los parámetros de configuración y se identifique correctamente.
2. Llamada a un servicio `REST` que envía los parámetros de configuración de la aplicación y la identidad del usuario de seguridad.
3. Llamada a un servicio `REST` para identificar el terminal con la Sala correspondiente.
4. Menú de configuración: que permite modificar los parámetros de configuración de la aplicación móvil.
5. Lectura de la tarjeta NFC que identifica al usuario que intenta acceder.
6. Llamada a un servicio `REST` para validar los datos del usuario que intenta acceder.
7. La comunicación se realizará mediante el protocolo `HTTPS`.
8. Los datos que se envían en las comunicaciones se cifran mediante el algoritmo AES.

La aplicación consta de 4 actividades y 3 clases auxiliares, la función de cada una de ellas es la siguiente:

- **LoginActivity:** en esta actividad se configuran y almacenan los parámetros de configuración y se identifica a los usuarios de seguridad en la aplicación móvil
- **MainActivity:** en esta actividad se identifica la sala, se muestran los datos de la sala y contiene el menú de configuración de los parámetros de configuración de la aplicación.
- **NfcActivity**: en esta actividad se lee la tarjeta NFC del usuario y se valida si puede entrar o no.
- **UserActivity**: en esta actividad se gestionan los cambios de los parámetros de configuración que se realizan desde el menú de configuración.
- **Cypt**: en esta clase auxiliar se implementa el cifrado de los datos usando el algoritmo AES

- **Http**:en esta clase auxiliar se centralizan las peticiones `REST` a la aplicación servidora.
- **Prefern**: en esta clase auxiliar se centralizan las preferencias administrativas.

2.1.3.1 Implementación

2.1.3.1.1 Identificación de la aplicación

Para dar seguridad a la aplicación móvil es necesario que solo la puedan utilizar los usuarios de seguridad, con este fin se ha creado una pantalla inicial que permite identificar a los usuarios que ponen en marcha la aplicación. Esta funcionalidad se ha desarrollado en la clase `LoginActivity.java.`

En esta primera pantalla hay implementadas dos medidas de seguridad para poder acceder a la aplicación móvil. Una es la identificación mediante el nombre y la contraseña del usuario de seguridad.

El usuario y la contraseña están almacenados en la base de datos de la aplicación servidora y se emplea el sistema de autentificación integrado en Django para almacenar los datos de este tipo de usuarios.

Por otra parte para acceder es necesario introducir un pin. Este pin es una medida de seguridad que se implementa únicamente en la aplicación móvil. Para desarrollar este sistema de autentificación se ha hecho uso de las preferencias compartidas más conocidas por su nombre en inglés, *Shared Preferences.* Las preferencias compartidas [14] permiten guardar y recuperar datos en forma de llave, par de valores, el nombre del campo y el valor de la variable.

Para poder utilizar las preferencias hay que llamar a al método **`getSharedPreferences ()`**, este método devuelve una instancia `SharedPreference`, que apunta al archivo que contiene los valores de las preferencias.

A este método hay que pasarle dos parámetros, el primero es el nombre que se le quiere asignar a la preferencia y el segundo parámetro es el modo, en este caso, el modo elegido es el modo privado para que solo esta aplicación pueda acceder a estos datos.

Para poder editar la preferencia hay que utilizar el constructor `SharedPreferences.Editor.`

Para poder manipular los datos se usa el método **`putString ()`**.Este método tiene una sintaxis definida, en el primer parámetro se indica el nombre que se le va a signar al dato y como segundo parámetro el dato que se va almacenar en la preferencia.

Para leer los datos que están almacenados en las preferencias se utiliza el método **`putString (),`** al cual hay que indicarle el nombre del dato que se quiere obtener y el valor de ese dato.

La primera vez que se instale la aplicación en el terminal, se le pedirá al usuario que introduzca un pin, este pin se almacena en la preferencia compartida denominada Pin, que se ha creado para tal fin.

Las siguientes veces que se ejecute la aplicación en ese dispositivo se consultara la preferencia compartida indicada, para verificar que el pin introducido es correcto o no. Un usuario solo podrá poner en marcha esta aplicación si conoce el Pin.

Una vez confirmado el pin, la aplicación hará una petición post a la aplicación servidora, redirigiendo la petición a la URL que el usuario haya indicado.

Con el fin de identificar la identidad del usuario que pone en marcha la aplicación, se le envía en la petición el nombre de usuario y la contraseña, para su verificación por parte de la aplicación servidora.

Únicamente será válida la petición, si la URL indicada, corresponde a la dirección donde está alojada la aplicación servidora, y el usuario y la contraseña corresponden a un usuario previamente dado de alta y que se encuentre activo en el momento de hacer la petición.

Si la autentificación es correcta se guardan los datos introducidos en la preferencia denominada Preferencias, para que posteriormente si el usuario de seguridad lo desea, desde el menú de configuración, se puedan modificar estos datos.

La aplicación se ha programado de tal forma, que si un usuario pulsa el botón vuelta atrás de Android, no pueda acceder a esta clase de nuevo, la aplicación se detendrá. Si se quiere volver a activar la aplicación, el usuario deberá autentificarse de nuevo.

2.1.3.1.2 Menú de configuración

Para que un usuario de seguridad pueda realizar cambios en los parámetros de configuración del sistema, se ha incluido un menú en la barra de acciones donde se pueden realizar estos cambios.

Para definir las distintas opciones que va a contener el menú hay que crear el archivo `configuración.xml` en la carpeta **res > menú.** En este archivo para cada opción del menú se debe definir un ítem que contenga el id de ese ítem, el titulo y como se va a comporta dentro de la barra de acciones. Estos ítem deben estar definidos dentro de un elemento menú.

Este menú ha sido invocado desde la clase `MainActivity.java` mediante el método **`onCreateOptionsMenu ().`**

En el método **`onOptionsItemSelected (),`** también incluido en esta clase, se ha implementado el comportamiento que va a tener el menú. Se ha incluido la petición del cogido Pin para poder acceder a las distintas opciones del menú, ya que como este menú

se visualiza en la pantalla de bienvenida, cualquier tipo de usuario podría acceder a cambiar las opciones de configuración si el menú no estuviera protegido.

La petición del código pin se ha implementado en un dialogo cuyo diseño está definido en el Layout denominado `dialog.xml`.

Este dialogo se ha implementando usando la subclase `AlertDialog`. En el área de contenido de esta clase únicamente se ha puesto el mensaje “Introduzca Pin “y se han definido dos Botones de acción.

El botón Cancelar, que su única funcionalidad es cancelar el dialogo y regresar a la pantalla anterior.

El en botón OK se realiza la comprobación del código Pin. Para comprobar si el pin es correcto o no se ha creado el método **`comprobarPin`** en la clase `Prefern.java`.

Este método hace la lectura del campo del formulario y según lo que el usuario haya introducido llevara a cabo una función u otra.

Si el usuario no ha introducido datos en el formulario, le muestra el mensaje “Introduzca Pin”.

Si el usuario introduce datos en el formulario, se hace la lectura del valor del Pin que hay almacenado en la preferencia denominada “Pin” usando el método **`getVariablePin`** creado para que lea ese valor en la preferencia.

Si el valor introducido por el usuario y el leído de la preferencia no es igual, se muestra al usuario el mensaje “Pin Incorrecto “y la aplicación muestra de nuevo la pantalla de bienvenida.

Si el valor introducido coincide con el que hay almacenado en la preferencia, la aplicación se redirige a la clase `UserActivity.java`, esta clase es la encargada de llevar a cabo las peticiones de modificación de el parámetro de configuración elegido por el usuario ya identificado.

En esta clase se ocultaran, por defecto, todos los campos del formulario creado en el Layout denominado `activity_user.xml`. Solo se mostrara el formulario correspondiente al valor de configuración que el usuario haya elegido modificar, también se personaliza el mensaje que se muestra como titulo del formulario según la selección realizada por parte usuario.

Para cada modificación el sistema tiene que guardar los valores modificados introducidos en las preferencias compartidas implementadas. Para guardar los datos modificados se ha diseñado el método **`saveEditText`** en la clase `Prefern.java`.

Para modificar el nombre de usuario hay que realizar una petición `REST` al método **`rest_modUsuario`**, creado en la aplicación servidora, esta llamada esta implementada en el método **`llamadaRest1`**. Para hacer esta modificación hay que enviar en la petición el

nombre de usuario almacenado en las preferencias compartidas y el nuevo nombre de usuario que quiere usar el usuario de control. Si esta petición realiza los cambios correctamente en la base de datos de la aplicación servidora, se guardara el nuevo nombre de usuario en la preferencia usando el método **`saveEditText`** creado para tal fin.

Para modificar la contraseña del usuario de control hay que hacer una petición `REST` al método **`rest_modPassword`** desarrollado en la aplicación servidora. Esta petición se realiza en el método `llamadaRest`, esta petición envía el nombre de usuario y la nueva contraseña que quiere tener el usuario. Si la respuesta a esta petición es correcta, la aplicación móvil guardara el valor de la nueva contraseña en las preferencias usando el método **`saveEditText.`**

Si el usuario lo que quiere modificar es la URL donde está alojada la aplicación servidora, lo único que se va a realizar es guardar este dato en la preferencia compartida correspondiente usando también el método **`saveEditText.`** Las siguientes peticiones que se realicen a la aplicación servidora se realizaran usando esta URL almacenada.

Si el usuario lo que quiere es cambiar el Pin de acceso a la aplicación móvil, para realizar este cambio se llama al método **`modificarPin`**, creado también en la clase `Prefern.java.` Este método tras comprobar los datos introducidos por el usuario guarda el nuevo valor del pin en la preferencia compartida denominada Pin, creada únicamente para almacenar este dato.

2.1.3.1.3 Identificar sala

El usuario de seguridad una vez logado en la aplicación debe de identificar la sala que va a controlar. La lógica que sigue la aplicación para realizar la identificación de la sala se puede observar en la *Figura 9. Diagrama de flujo identificar sala*.

Para hacer una identificación segura se ha utilizado el número de IMEI que va asociado al dispositivo que se va a utilizar en la sala. De esta forma se garantiza que solamente un terminal puede identificar de forma inequívoca una sala.

Esta identificación se ha implementado en la clase `MainActivity.java.`

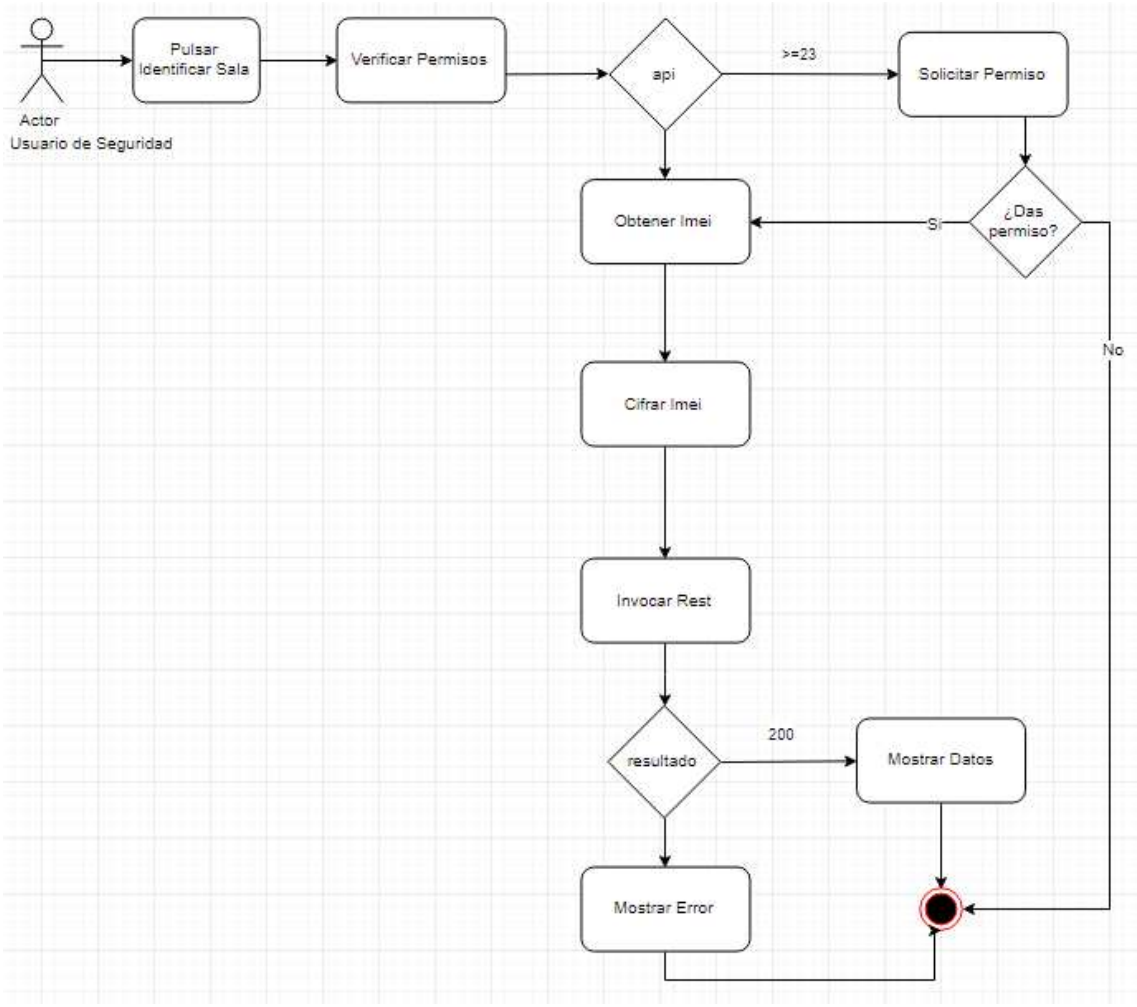

Figura 9. Diagrama de flujo identificar sala.

Lo primero que hay que hacer es ver qué tipo de versión de Android tiene el dispositivo móvil, si es una versión igual o superior a Android 6.0, para poder leer el número de IMEI del terminal, la aplicación debe solicitar permisos al usuario para poder leer este dato. El tipo de permiso que se necesita para leer el IMEI en un dispositivo Android se denomina `READ_PHONE_STATE`.

Es importante que en el archivo de configuración `AndroidManifest.xml` previamente se hayan añadido los permisos de usuarios necesarios

```
<!-- Permisos imei-->
<!--WRITE_EXTERNAL_STORAGE tiene implícito READ_EXTERNAL_STORAGE porque pertenecen al mismo grupo de permisos-->
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
```

Figura 10. Permiso lectura IMEI

En los casos, que por la versión que tiene instalada el dispositivo, sea necesario pedir permisos a los usuarios, se usara la función **`requestPermission ().`**

Esta función comprueba si ya se le ha solicitado el permiso o no anteriormente al usuario, en el caso que se le haya pedido y el usuario lo haya denegado, se le recuerda que es necesario leer el IMEI del teléfono móvil para poder utilizar la aplicación.

Si el usuario hubiera marcado anteriormente el cuadro de comprobación de que no se le vuelva a preguntar, no podrá hacer uso de la aplicación en ese dispositivo móvil, hasta que desinstale y vuelva a instalar la aplicación móvil.

El mensaje se muestra al usuario usando un nuevo método de notificaciones llamado *snackbar*[16] . Se utiliza un *snackbar* para mostrar un mensaje, el usuario debe validar, para que la aplicación pueda hacer la lectura del IMEI del dispositivo. Este mensaje se muestra en la *Figura 11. Snackbar de permisos* y contiene un botón de texto de acción que registrara la decisión del usuario.

La función **`showSnackBar ()`** es la que lanza el evento de preguntarle al usuario si acepta el permiso o no.

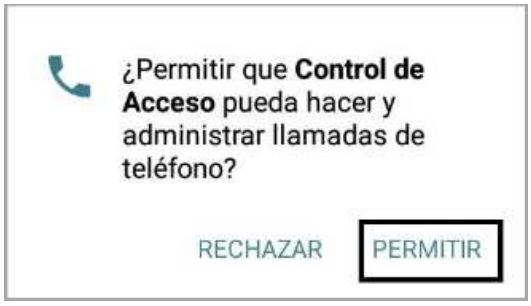

Figura 11. Snackbar de permisos.

Cuando el usuario haya contestado hay que procesar su respuesta, esto se lleva a cabo en la función onRequestPermissionsResult ().

Si el usuario no acepta los permisos se le vuelven a solicitar. Si no permite que se pueda leer el IMEI del teléfono, no podrá hacer uso de la aplicación.

Si el usuario acepta los permisos se procederá a leer el IMEI, esto se hará en la función **`obtenerIMEI ().`**

Esta función usa la *API* `android.telephony`. El método `getDeviceID ()` de esta *API* es la que devuelve el número de IMEI del terminal.

Una vez se sabe el IMEI hay que enviarlo a la aplicación servidora para poder identificar la sala. Para que la comunicación entre la aplicación servidora y la aplicación móvil sea segura se ha optado por cifrar este dato antes de enviarlo, haciendo uso del algoritmo `AES`. Este algoritmo esta implementado en la clase `Crypt.java` y hace uso de la librería de java denominada `javax.crypto`.

[16]Mensajes *snackbar*:aparecen por encima de todos los demás elementos de la pantalla y desaparecen automáticamente después de un tiempo de espera o después de la interacción del usuario

Una vez cifrados los datos, para poder enviarlos a la aplicación servidora se ha optado por hacer una petición `HTTPS` al servicio `REST`. Para la implementación se ha realizado en la clase Http, utilizando el paquete `org.apache` para poder hacer este tipo de comunicación.

Para el intercambio de datos entre las dos aplicaciones, se ha utilizado el formato conocido por las siglas JSON, que hacen referencia a su nombre en inglés *Java Script Object Notation*, que traducido al español significa notación de objetos JavaScript. Para utilizar este formato es necesario el uso de la librería `json.JSONObject`.

Un objeto JSON se delimita entre llaves y para asignar a una variable un valor se usa los dos puntos. Un ejemplo de este formato es el siguiente:

{"Hash": "Zl5oR1bza3d6P523e/yy2DD4VvjRLB15Dj36CGUb5Ks="}.

Una vez se recibe la respuesta de la aplicación servidora a la petición, se comprueba si la ejecución ha sido correcta o no.

Si es correcta, se muestran los datos de la sala que se acaba de identificar y se visualizara el botón de leer NFC.

Si la ejecución es errónea, la aplicación mostrara el mensaje de error que ha recibido de la aplicación servidora y permanecerá en la pantalla principal para que se vuelva a intentar la identificación.

Si la comunicación entre ambas aplicaciones no se puede establecer se muestra una pantalla que muestra el mensaje que se ha producido y se muestra activo el botón de Identificar Sala para que se vuelva a realizar la solicitud sin tener que salir de la aplicación.

2.1.3.1.4 Lectura de tarjetas NFC

Cuando un usuario quiere acceder a una sala, primero debe pulsar el botón “leer NFC”, para poder acceder a la pantalla donde podrá hacer la lectura de su tarjeta identificativa. Al pulsar el botón lo que el sistema muestra es la vista contenida en la clase `NfcActivity`. Cuando la aplicación se encuentra en esta vista, esta espera que un usuario se acerque y haga la lectura de sus credenciales.

Para poder programar el uso del NFC es necesario utilizar la librería `android.nfc`.

En el archivo `AndroidManifest.xtml` es imprescindible incluir el permiso necesario para que el dispositivo pueda hacer uso del NFC, esto permiso se muestra en la *Figura 12.Permisos NFC*.

```
<!-- Permisos nfc-->
<uses-permission android:name="android.permission.NFC" />
```

Figura 12.Permisos NFC

En la clase `NfcActivity` lo primero que se hace es comprobar si el dispositivo móvil tiene NFC y si este está activado, sino esta activado se solicitara al usuario que lo active el NFC en el dispositivo.

Si se trata de un dispositivo que no tiene la tecnología NFC integrada, no podrá hacer uso de la aplicación.

Estas comprobaciones se realizan en el método `OnCreate ()`.Para saber cuál es el adaptador por defecto del terminal Android se usa el método getDefaultAdapter () que contiene la librería `android.nfc`.

Si esta correcta la configuración de NFC, la lectura de la tarjeta se lleva a cabo en la clase **`handleIntent ()`.**Este método se ejecuta al acceder a la actividad y se queda esperando en segundo plano a que un usuario realice la lectura de su tarjeta. Este intent se lanza cuando al dispositivo se le acerca una tarjeta NFC.

Para programar un `intent` es necesario que en el archivo `AndroidManifest.xtml` se defina el tipo de intent y el tipo de datos que va utilizar, los tipos de intent que se deben declara para la lectura de NFC son NDEF_DISCOVERED y TECH_DISCOVERED y el tipo de dato que se utiliza es texto plano. La definición en el archivo que se ha realizado AndroidManifest.xtml se puede ver en la Figura 13. Definición intent NFC.

```
<activity

    android:name=".NfcActivity"
    android:configChanges="keyboardHidden|orientation|screenSize">

    <intent-filter>
        <action android:name="android.nfc.action.NDEF_DISCOVERED"/>
        <action android:name="android.nfc.action.TECH_DISCOVERED" />

        <data android:mimeType="text/plain" />
    </intent-filter>
    <meta-data
        android:name="android.nfc.action.TECH_DISCOVERED"
        android:resource="@xml/nfc_tech_filter" />
</activity>
```

Figura 13. Definición intent NFC.

Cuando se lee una tarjeta NFC lo primero que se debe comprobar es que tipo de datos que utiliza. Si los datos siguen el patrón del formato NDEF, lo cual se sabrá por la variable `ACTION_NDEF_DISCOVERED`, se puede leer la tarjeta.

Hay comprobar también si el tipo de mime que contiene es el que se ha definido previamente, lo indicara la variable `MIME_TEXT_PLAIN`. Cuando estas dos condiciones se den, se llamara a la clase **NdefReaderTask ().**

En algunos casos puede pasar que cuando se escanea una etiqueta esta contenga datos formateados en NDEF, pero no se pueden asignar a un tipo MIME. En estos casos, debe abrir la comunicación directamente con la etiqueta y leerla y escribirla con su propio protocolo (en bytes sin procesar). En este caso se lanza un intent `ACTION_TECH_DISCOVERED`. Para hacer uso de este intent es necesario que previamente se declare en el archivo `AndroidManifest.xtml` el intent y los metadatos como se puede observar en la *Figura 13. Definición intent NFC.*

Se ha evitado que cada vez que se lleve a cabo una lectura la aplicación habrá una nueva actividad haciendo que las lecturas se hagan en segundo plano, para ello se usa `ForegroundDispatch` dando prioridad a la aplicación que esta activa será esta, para que sea esta aplicación la que realice la lectura de los datos por NFC y no cualquier otra aplicación que tenga el dispositivo móvil instalado. Para controlar esto se usan estos tres métodos:

- `onResume ():` activa la actividad en un segundo plano y si no se produce una excepción.
- `onPause ():` pone en pausa la actividad.
- `onNewIntent ():` activa la actividad cuando un usuario acerca una tarjeta.

En la clase **`NdefReaderTask`** se lee los datos de la tarjeta y se envía a la aplicación servidora. La lectura se lleva a cabo en el método `doInBackground ()`, se hace en una tarea asíncrona, para ello se define como `AsyncTask`. Solo se permite la lectura de los datos que estén formateados en NDEF y los datos se leen haciendo uso del método `getRecords ().`

Una vez se hace la lectura de esos datos es necesario formatearlos según las especificaciones técnicas del modelo de datos NDEF, esto se lleva a cabo en el método `readText ()`.

En el método `onPostExecute ()` recoge los datos de leídos por NFC, se cifran y se envían por `HTTPS` a la aplicación servidora.

Si el usuario tiene permisos mostrara en pantalla la vista de que el acceso del usuario está permitido y en caso contrario mostrara en pantalla el motivo por el que no permite el acceso al usuario.

El diagrama que describe como se hace la lectura de las tarjetas NFC se puede observar en la *Figura 14.Diagrama de flujos de la lectura de NFC.*

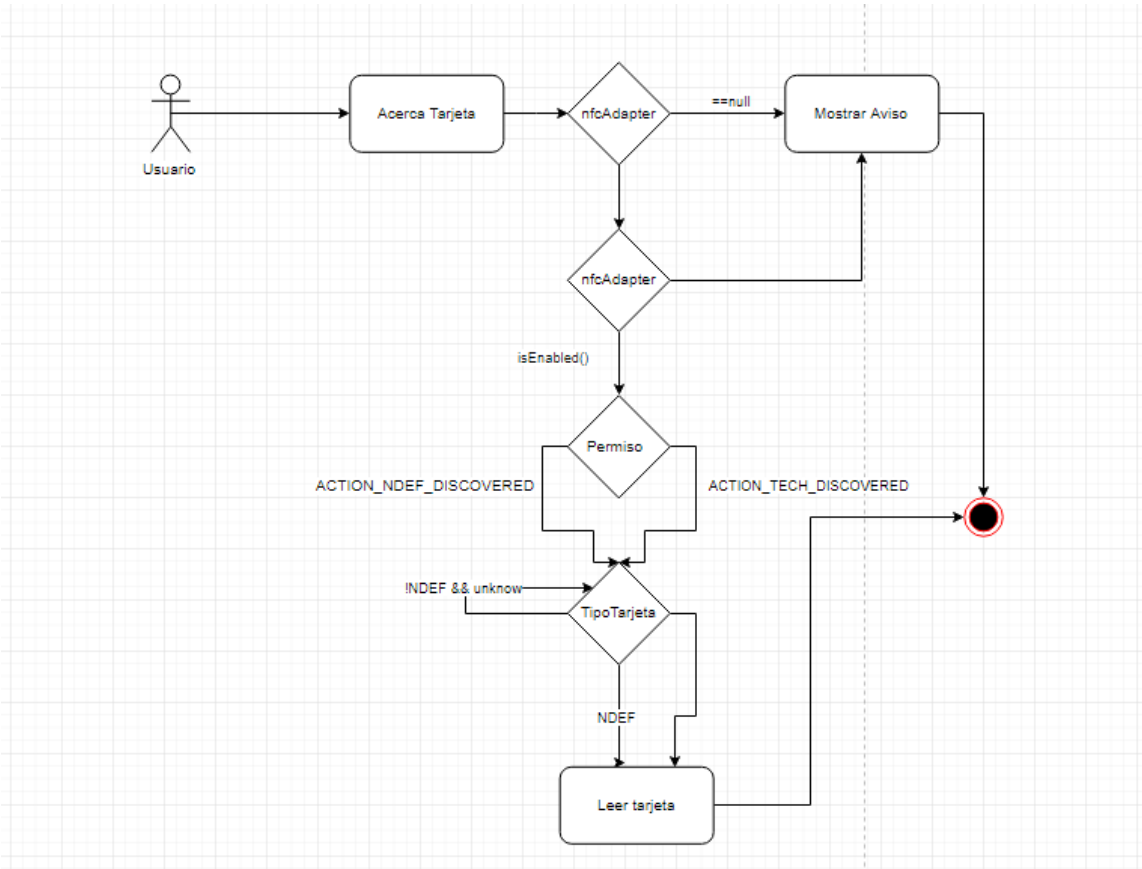

Figura 14.Diagrama de flujos de la lectura de NFC.

3 PLAN DE PRUEBAS

Para comprobar el funcionamiento correcto del sistema de control de acceso, se ha elaborado un plan de pruebas en el cual se han evaluado todos los posibles casos de uso de la aplicación.

3.1 Pantalla de acceso

Crear Pin		
Descripción:	Cuando el usuario instala por primera vez la aplicación en el terminal debe crear un pin de acceso.	
Prerrequisitos:	Tener la aplicación instalada.	
Pasos:	1	El usuario de seguridad inicia a la aplicación.
	2	El sistema muestra la pantalla de acceso.
		2.1 El usuario introduce un pin de 4 dígitos en el campo Pin de la pantalla acceso.
		2.2 El usuario no introduce datos en el campo Pin de la pantalla acceso.
		2.3 El usuario introduce un pin que no tiene 4 dígitos en el campo Pin de la pantalla acceso.
	3	El usuario pulsa el botón “Hacer Login”.
Resultados	2.1. El sistema muestra el mensaje “Pin creado”, almacena el valor en las preferencias compartidas en el campo “Pin” y muestra la pantalla de bienvenida (ver *Figura 20. Vista creación Pin.*). 2.2: El sistema muestra el mensaje “Introduzca Pin”. 2.3. El sistema muestra el mensaje “Introduzca un pin de 4 dígitos”.	

Tabla 1.Prueba Crear Pin.

Introducir Pin		
Descripción:	Cuando el usuario quiere iniciar la aplicación debe introducir en el campo Pin, el pin antes creado para la aplicación móvil.	
Prerrequisitos:	Tener la aplicación instalada.	
Pasos:	1	El usuario de seguridad accede a la aplicación.
	2	El sistema muestra la pantalla de acceso.
		2.1 El usuario introduce un pin correcto.

		2.2	El usuario no introduce datos en el campo Pin
		2.3	El usuario introduce un pin incorrecto en el campo pin de la pantalla de acceso.
	3	El usuario pulsa el botón “Hacer Login”.	
Resultados	2.1. El sistema muestra la pantalla de bienvenida, si el resto de los datos introducidos en el formulario de registro son correctos (ver la *Figura 22.Vista pantalla de Bienvenida.*). 2.2: El sistema muestra el mensaje “Introduzca Pin”. 2.3. El sistema muestra el mensaje “Pin Incorrecto” (ver la *Figura 21. Vista pantalla Pin Incorrecto.*)		

Tabla 2.Prueba Introducir Pin

Datos de Usuario de seguridad			
Descripción:	En la pantalla de acceso el usuario de seguridad debe identificarse indicando su usuario y contraseña asignada.		
Prerrequisitos:	Tener la aplicación instalada		
Pasos:	1	El usuario de seguridad accede a la aplicación.	
	2	El sistema muestra la pantalla de acceso.	
		2.1	El usuario introduce el nombre y password correctos.
		2.2	El usuario introduce el nombre y password erróneos.
		2.3	El usuario deja vacios los campos de usuario y password.
		2.4	El usuario no está activo en la aplicación servidora.
	3	El usuario pulsa el botón “Hacer Login”.	
Resultados	2.1. El sistema muestra la pantalla de bienvenida de la aplicación (ver *Figura 22.Vista pantalla de Bienvenida.*). 2.2. El sistema muestra el mensaje “Datos de usuario incorrectos”. 2.3: El sistema muestra el mensaje “Introduzca todos los datos”. 2.4. El sistema muestra el mensaje “Datos de usuario incorrectos”.		

Tabla 3.Prueba de los datos de los usuarios de seguridad.

URL de la aplicación servidora	
Descripción:	En la pantalla de acceso el usuario de seguridad debe indicar la url donde está alojada la aplicación servidora.

Prerrequisitos:	Tener la aplicación instalada		
Pasos:	1	El usuario de seguridad accede a la aplicación.	
	2	El sistema muestra la pantalla de acceso.	
		2.1	El usuario introduce la url correcta.
		2.2	El usuario introduce la url incorrecta.
		2.3	El usuario deja vacio el campo url.
		2.4	La aplicación servidora no responde a la petición.
	3	El usuario pulsa el botón “Hacer Login”.	
Resultados	2.1. El sistema muestra la pantalla de Bienvenida de la aplicación. 2.2. El sistema muestra el mensaje “Petición Incorrecta”. 2.3: El sistema muestra el mensaje “Introduzca todos los datos”. 2.4. El sistema muestra el mensaje “Petición Incorrecta”.		

Tabla 4.Pruebas de la URL de la aplicación servidora

3.2 Pantalla identificar sala

Identificar sala			
Descripción:	En la pantalla de bienvenida se debe identificar a que sala pertenece el terminal móvil		
Prerrequisitos:	Tener la aplicación instalada y el usuario debe estar identificado.		
Pasos:	1	El usuario de seguridad accede a la aplicación.	
	2	El usuario de seguridad se ha identificado.	
	3	El sistema muestra la pantalla de bienvenida	
	4	El usuario de seguridad pulsa el botón “Identificar Sala”	
		4.1	El IMEI del terminal de la sala está registrado en la aplicación servidora.
		4.2	El IMEI del terminal de la sala está registrado en la aplicación servidora.
		4.3	Ocurre un error en la petición a la aplicación servidora.
Resultados	4.1. El sistema muestra la pantalla con todos los datos de la sala (ver *Figura 29. Vista identificación de sala.*).La aplicación servidora verifica que el IMEI es correcto y responde a la petición con todos los datos de la sala. 4.2. El sistema muestra la pantalla de error que se puede		

	observar en la *Figura 30. Vista error de identificación.* 4.3: El sistema muestra la pantalla que se muestra en la *Figura 31.Vista error de la aplicación servidora.*

Tabla 5.Pruebas opción Identificar Sala.

3.3 Menú de configuración

Pin de acceso al menú de configuración		
Descripción:	En la pantalla de bienvenida, en la parte superior izquierda hay un menú de configuración, desde el cual el usuario puede cambiar los datos de configuración de la aplicación móvil.	
Prerrequisitos:	Tener la aplicación instalada y el usuario debe estar identificado.	
Pasos:	1	El usuario de seguridad accede a la aplicación.
	2	El usuario de seguridad se ha identificado.
	3	El sistema muestra la pantalla de bienvenida
	4	El usuario de seguridad pulsa el botón de acceso al menú
	5	El usuario elige una opción del menú de configuración.
		5.1 El usuario introduce el pin correctamente y pulsa el botón “Ok”
		5.2 El usuario introduce el pin incorrecto y pulsa el botón “Ok”
		5.3 El usuario deja vacio el campo de pin y pulsa el botón “Ok”
		5.4 El usuario pulsa el botón “Cancelar”
Resultados	5.1. El sistema muestra la pantalla correspondiente a la opción del menú de configuración que el usuario haya elegido (ver *Figura 26. Vista formulario configuración.*) 5.2. El sistema muestra el mensaje “Pin Incorrecto” y vuelve a la pantalla de bienvenida (ver *Figura 25. Vista mensaje Pin Incorrecto.*) 5.3. El sistema muestra el mensaje “Introduzca Pin” y vuelve a la pantalla de bienvenida 5.4 El sistema vuelve a la pantalla de bienvenida (ver *Figura 22.Vista pantalla de Bienvenida.*).	

Tabla 6.Pruebas del Pin de acceso al menú de configuración

Menú de configuración Modificar Usuario

Descripción:	En la pantalla de bienvenida, en la parte superior izquierda hay un menú de configuración, desde el cual el usuario puede cambiar su nombre de usuario de seguridad.		
Prerrequisitos:	Tener la aplicación instalada y el usuario debe estar identificado.		
Pasos:	1	El usuario de seguridad accede a la aplicación.	
	2	El usuario de seguridad se ha identificado.	
	3	El sistema muestra la pantalla de bienvenida	
	4	El usuario de seguridad pulsa el botón de acceso al menú y elige la opción “Modificar Usuario”.	
	5	El usuario introduce el pin correcto.	
	6	El sistema muestra la pantalla de “Modificar Usuario”.	
		6.1	Introduce un nuevo nombre de usuario en el formulario.
		6.2	Deja el formulario vacio.
		6.3	El nuevo nombre que introduce el usuario ya existe en la base de datos.
		6.4	Ocurre un error en la petición a la aplicación servidora.
	7	El usuario pulsar el botón “Guardar Datos”.	
Resultados	6.1. El sistema muestra el mensaje “Usuario Modificado” y regresa a la pantalla anterior. En la base de datos se modifica el nombre del usuario. 6.2. El sistema muestra el mensaje “Introduzca nuevo nombre de usuario”. 6.3. El sistema muestra el mensaje “Nombre de Usuario ya Existe en la Base de Datos” y regresa a la pantalla anterior. 6.4. El sistema muestra el mensaje “Petición Incorrecta” y regresa a la pantalla anterior.		

Tabla 7.Pruebas menú de configuración Modificar Usuario.

Menú de configuración Modificar Password	
Descripción:	En la pantalla de bienvenida, en la parte superior izquierda hay un menú de configuración, desde el cual el usuario puede cambiar su contraseña de usuario de seguridad.
Prerrequisitos:	Tener la aplicación instalada y el usuario debe estar logado.

Pasos:	1	El usuario de seguridad accede a la aplicación.	
	2	El usuario de seguridad se ha identificado.	
	3	El sistema muestra la pantalla de bienvenida	
	4	El usuario de seguridad pulsa el botón de acceso al menú e elige la opción “Modificar Password”.	
	5	El usuario introduce el pin correcto.	
	6	El usuario elige la opción de “Modificar Password”.	
		6.1	Introduce una nueva contraseña en el formulario
		6.2	Deja el formulario vacio.
		6.3	Ocurre un error en la petición a la aplicación servidora.
	7	El usuario pulsa el botón “Guardar Datos”.	
Resultados	6.1. El sistema muestra el mensaje “Password Modificado” y regresa a la pantalla anterior (ver *Figura 27. Vista mensaje de modificación.*). 6.2. El sistema muestra el mensaje “Introduzca nueva contraseña para el usuario” 6.3. El sistema muestra el mensaje “Petición Incorrecta” y regresa a la pantalla anterior.		

Tabla 8. Pruebas menú de configuración Modificación Password.

Menú de configuración Modificar Url			
Descripción:	En la pantalla de bienvenida, en la parte superior izquierda hay un menú de configuración, desde el cual el usuario puede cambiar la url donde está alojada la aplicación servidora.		
Prerrequisitos:	Tener la aplicación instalada y el usuario debe estar identificado.		
Pasos:	1	El usuario de seguridad accede a la aplicación.	
	2	El usuario de seguridad se ha identificado.	
	3	El sistema muestra la pantalla de bienvenida	
	4	El usuario de seguridad pulsa el botón de acceso al menú y elige la opción “Modificar Url”.	
	5	El usuario introduce el pin correcto.	
	6	El sistema muestra la pantalla de “Modificar Url”.	
		6.1	Introduce un nuevo valor para la url de la aplicación

			servidora.
		6.2	Deja el formulario vacio.
	7	El usuario pulsar el botón “Guardar Datos”.	
Resultados	6.1. El sistema muestra el mensaje “URL Modificada” y regresa a la pantalla anterior. 6.2. El sistema muestra el mensaje “Introduzca nueva url del servidor”		

Tabla 9. Pruebas menú de configuración Modificar URL.

Menú de configuración Modificar Pin			
Descripción:	En la pantalla de bienvenida, en la parte superior izquierda hay un menú de configuración, desde el cual el usuario puede cambiar el pin con el que accede a la aplicación móvil.		
Prerrequisitos:	Tener la aplicación instalada y el usuario debe estar identificado.		
Pasos:	1	El usuario de seguridad accede a la aplicación.	
	2	El usuario de seguridad se ha identificado.	
	3	El sistema muestra la pantalla de bienvenida	
	4	El usuario de seguridad pulsa el botón de acceso al menú y elige la opción “Modificar Pin”.	
	5	El usuario introduce el pin correcto.	
	6	El sistema muestra la pantalla de “Modificar Pin”.	
		6.1	Introduce un nuevo el pin de 4 dígitos para la de la aplicación.
		6.2	Deja el formulario vacio.
		6.3	Introduce un pin de una longitud distinta a 4 dígitos para la aplicación.
	7	El usuario pulsa el botón “Guardar Datos.	
Resultados	6.1. El sistema muestra el mensaje “Pin Modificado” y regresa a la pantalla anterior. 6.2. El sistema muestra el mensaje “Introduzca nuevo pin” 6.3. El sistema muestra el mensaje “Debe introducir un pin de 4 dígitos”.		

Tabla 10. Pruebas menú de configuración Modificar Pin.

3.4 Pantalla de lectura NFC

Lectura de la tarjeta de identificación de usuarios NFC		
Descripción:	En la pantalla de identificación de sala hay un botón denominado Leer NFC Cuando llega un usuario para identificarse debe pulsar dicho botón La aplicación se redirige a la pantalla de lectura de tarjetas, esperando que el usuario haga la lectura de sus credenciales.	
Prerrequisitos:	Tener la aplicación instalada, el usuario de seguridad debe estar identificado y tiene que estar la sala identificada.	
Pasos:	1	El usuario de seguridad accede a la aplicación.
	2	El usuario de seguridad se ha registrado.
	3	El sistema muestra la pantalla de bienvenida
	4	El usuario de seguridad ha identificado la sala.
	4	El usuario que quiere acceder a la sala ha pulsado el botón “Leer NFC”.
	5	El sistema muestra la pantalla de NFC.
	6	El usuario hace la lectura de su tarjeta NFC.
		6.1 Datos del usuario correctos.
		6.2 Datos del usuario incorrectos.
		6.3 El usuario esta activo.
		6.4 El usuario esta desactivado.
		6.5 La sala esta activada.
		6.6 La sala esta desactivada.
		6.7 No hay registro anterior de ese usuario en esa sala
		6.8 Hay registro anterior de ese usuario en esa sala
		6.9 El registro anterior tiene fecha de salida.
		6.10 El registro anterior no tiene fecha de salida.
		6.11 Hay aforo en la sala
		6.12 No hay aforo en la sala
	7	El sistema muestra el resultado de la lectura de credenciales.
Resultados	6.1. El sistema pasa a comprobar si el usuario esta activo o no	

	(ver pasos 6.3 y 6.4) 6.2. La aplicación le muestra al usuario la pantalla que se puede ver en la *Figura 33: Vista de error de usuario.* 6.3. El sistema va a comprobar si la sala esta activada o no. (ver pasos 6.5 y 6.6) 6.4. La aplicación le muestra al usuario la pantalla que se puede ver en la *Figura 38.Vista de error de usuario desactivado.* 6.5. El sistema pasa a comprobar si hay registro previo del usuario en esa sala(ver pasos 6.7 y 6.8) 6.6. La aplicación le muestra al usuario la pantalla que se puede ver la *Figura 39.Vista de error de sala desactivada.* 6.7.El sistema pasa a comprobar si hay aforo o no en la sala (ver pasos 6.11 y 6.12) 6.8 El sistema pasa a comprobar si es un registro de salida o un nuevo registro de entrada (ver pasos 6.9 y 6.10). 6.9. El sistema comprueba que es un registro nuevo y pasa a verificar si hay aforo o no en la sala (ver pasos 6.11 y 6.12). 6.10. El sistema comprueba que es un registro de salida, por lo tanto, guarda la fecha de salida, actualiza el aforo y muestra al usuario la pantalla que se puede observar en la *Figura 35.Vista de registro de salida.* 6.11. El sistema guarda el nuevo registro, actualiza el aforo y muestra al usuario la pantalla que se puede ver en la *Figura 34.Vista de registro de entrada.* 6.12. El sistema muestra al usuario un mensaje indicando que el aforo esta completo (ver *Figura 41.Vista error de aforo* completo.)

Tabla 11. Pruebas de lectura de la tarjeta de identificación de usuarios NFC

4 CONCLUSIONES

En el mercado actual existen multitud de aplicaciones que controlan el acceso a recintos y la identidad del usuario que desea acceder, pero no se han encontrado aplicaciones que ofrezcan la movilidad y flexibilidad que ofrece la aplicación que se ha desarrollado en este proyecto.

Esta aplicación no necesita contar con el respaldo de una infraestructura previa para ser usada, únicamente requiere un dispositivo con sistema operativo Android que tenga incorporado la tecnología NFC.

Para hacer uso de esta aplicación lo único necesario es que nuestro dispositivo disponga de una conexión a Internet, ya que tanto el acceso a la aplicación servidora para su gestión, como la comunicación entre las dos aplicaciones se realiza vía web.

Una vez se ha finalizado el desarrollo es hora de valorar si se han cumplido los objetivos que se habían marcado para este proyecto:

- Autenticación en la aplicación móvil a través de tarjetas NFC: la aplicación permite leer los datos que contienen las tarjetas NFC de los usuarios .Al acceder a esos datos se pueden hacer las comprobaciones necesarias para dar acceso o no a los usuarios.
- Registro de cada acceso en una aplicación servidora remota que podrá otorgar acceso o no en función de políticas o permisos concretos: el control y diseño de la aplicación móvil está basado en las configuraciones que estén registradas en la aplicación servidora. Como se ha indicado anteriormente es el centro de coordinación de nuestro proyecto. En aplicación servidora se pueden otorgar o anular permiso de acceso a los usuarios a las distintas salas y también se pueden activar o desactivar salas y usuarios. Se ha implementado un registro en el cual se sabe con exactitud la fecha y horas de entrada y salida de cada usuario en una sala.
- Control de aforo: en la aplicación servidora en todo momento se puede llevar a cabo el control de los usuarios que hay en cada sala. Se establece un control de entrada para tener controlado, en todo momento, que no se supere el aforo máximo que tiene la sala, restringiendo la entrada siempre que sea necesario.
- La aplicación móvil desarrollada deberá estar preparada para adaptarse a diferentes tipos de terminales móviles que existen actualmente en el mercado. La aplicación está preparada para ser utilizada desde una versión de Android 4.0 (Ice CreamSandwich), ya que la API mínima en la que se ha creado el proyecto es la

API 15, hasta la versión Android 7.1 (Nougat), ya que el proyecto se ejecuta por defecto en una API 25.

Siguiendo los parámetros que nos proporciona el IDE Android Studio, podría funcionar en el 100 por cien de los dispositivos actuales que llevan incorporados el sistema operativo Android (ver *Figura 15. Sistema Operativo y API.*)

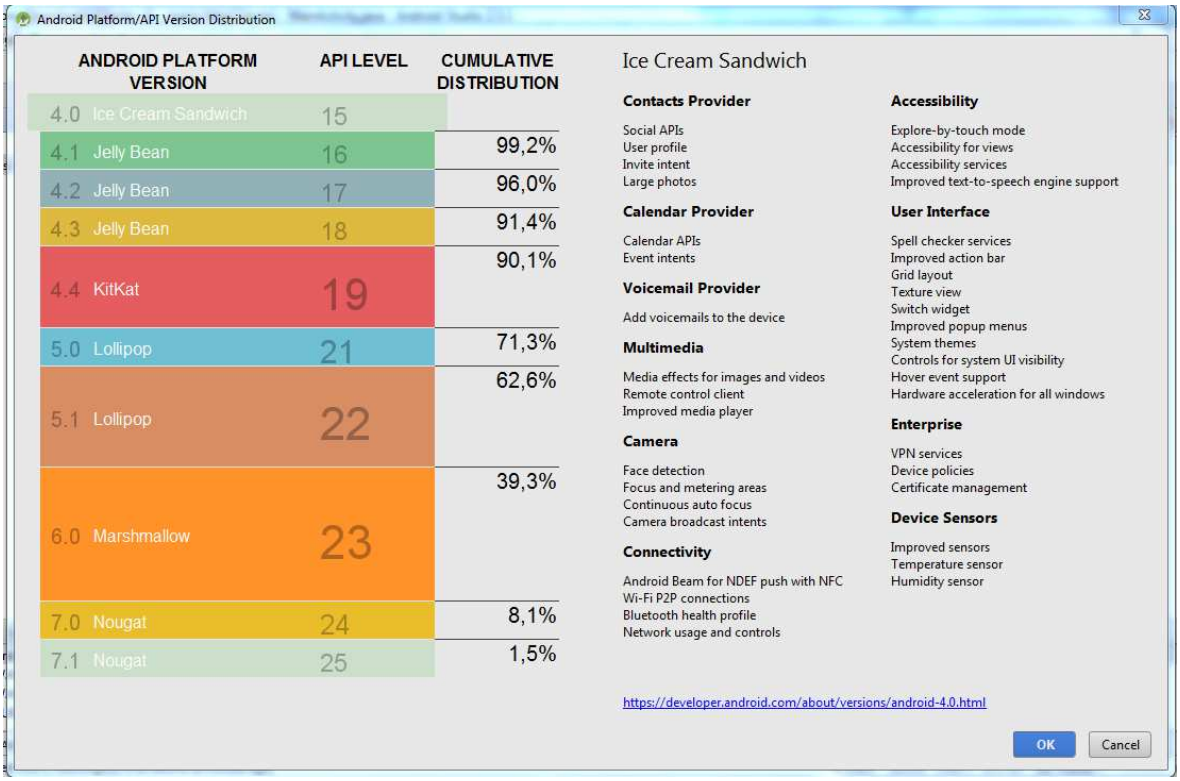

Figura 15. Sistema Operativo y API. [17]

- La aplicación servidora debe ser funcional en diferentes tipos de dispositivos y resoluciones de pantalla. Se ha conseguido gracias al uso de la plantilla de administración de Django e implementando el diseño facilitado por la API *Django-Suit* .Esta plantilla genera las vistas de la aplicación ,con el entorno de desarrollo de CSS ,conocido como *Bootstrap*, el cual permite diseñar aplicaciones web que se adapten a distintas resoluciones de pantalla y dispositivos.

[17] Disponible al hacer clip en el enlace **Help me choose**, que se encuentra en la pantalla de selección del tipo de SDK, cuando se están configurando los parámetros de un nuevo proyecto en Android Studio.

5 LINEAS DE FUTURO.

La aplicación desarrollada en este proyecto cumple con todos los objetivos que se habían establecido al inicio del desarrollo, pero hay un gran abanico de nuevas funcionalidades que se pueden agregar, debido a que los teléfonos móviles inteligentes tienen incorporados multitud de tecnologías, de las cuales se pueden hacer uso a través de futuras ampliaciones de esta aplicación móvil.

Sería interesante mejorar el diseño estético de esta aplicación, haciendo un diseño escalable que permitiera fácilmente adaptar la visualización a la imagen corporativa de la empresa que vaya a utilizarla.

Para darle mayor seguridad, en un desarrollo posterior sería conveniente que la palabra 'semilla' para el cifrado AES no fuera la misma en todos los caso. De tal forma que se pudiera modificar y cada sala perteneciente a la aplicación tuviera una distinta.

Otra ampliación que se podría llevar a cabo es que los usuarios no necesitaran disponer de una tarjeta NFC para identificarse, si no que con su teléfono móvil personal se pudiera llevar a cabo la identificación.

Además se podría incorporar una mejora en la aplicación móvil para que cuando estuviera activo el perfil de administración, se pudiera hacer también la escritura de estas tarjetas identificativas. De tal forma que con una única aplicación se podría leer y escribir las tarjetas identificativas.

Otro punto que se podría incorporar al proyecto es mejorar la visualización de las imágenes con las medidas de seguridad de la sala. Estas imágenes, en vez de permanecer estáticas en la pantalla principal, tendrían la posibilidad de ampliarse para facilitar su mejor visualización, guardarse en el teléfono móvil o enviarse a otros dispositivos para su posterior visionado.

Sería interesante, en un desarrollo posterior, incorporar la tecnología GPS a la aplicación móvil. De esta forma se podrá saber en todo momento la situación geográfica de las salas a controlar. De tal forma se podrían identificar las distintas salas, no solo por el terminal a la que va asociada, sino también limitar el área geográfica en el que se puede utilizar.

También se podría ampliar creando un sistema de reservas de sala, de tal forma que se podría hacer una reserva de una sala para un evento en una fecha determinada y en un horario fijado. Junto a esta ampliación sería interesante incorporar un sistema de avisos de eventos en la sala, de tal forma que cuando se desarrolle alguna actividad especial en la sala se muestre en la pantalla principal los detalles de ese evento, como puede ser el horario de duración, la temática o el ponente. Esto supondría incorporar en

la aplicación servidora la opción de gestión de avisos que nos permita establecer el control de los eventos que se desarrollan en las distintas salas.

También en un desarrollo posterior sería conveniente implementar un sistema de cifrado que permita que el usuario pueda modificar la clave de cifrado para cada sala, esta clave se almacenaría, junto al resto de los datos de las salas, en la base de datos de la aplicación servidora, y el usuario de control debería introducir esta clave, en la pantalla de registro, para poder acceder a la aplicación móvil.

6 ANEXOS

6.1 Manual de usuario de la aplicación

Se debe instalar la aplicación en el dispositivo móvil, una vez instalada, se localiza el icono de la aplicación y se ejecuta (ver *Figura 16.Vista icono* aplicación).

Cuando se ha ejecutado la aplicación se muestra la pantalla de registro (ver *Figura 17 .Vista pantalla de registro de la aplicación*).

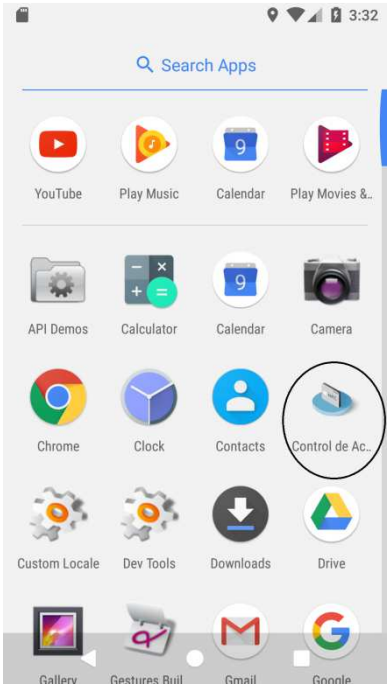

Figura 16.Vista icono aplicación.

Figura 17 .Vista pantalla de registro de la aplicación.

En esta pantalla el usuario de control debe introducir su nombre, su contraseña, la URL de la aplicación servidora y el pin del teléfono.

Si deja alguno de los campos vacíos se mostrará el mensaje “Introduzca todos los campos”, este mensaje se puede observar en la *Figura 18.Vista de error datos Vacios.*

Si el usuario de seguridad introduce su usuario o contraseña de forma errónea, la aplicación mostrara a este usuario el mensaje “Datos de usuario incorrectos”, para volver a intentar registrarse el usuario deberá introducir sus datos correctamente. Este mensaje de error se puede ver en la *Figura 19 .Vista error datos de Usuario Incorrectos.*

Figura 18.Vista de error datos Vacios.

Figura 19 .Vista error datos de Usuario Incorrectos.

La primera vez que se instala la aplicación en el teléfono, se debe crear el pin que permite acceder a la aplicación. La aplicación mostrara el mensaje “Pin creado” al crear el nuevo pin (ver *Figura 20. Vista creación Pin*).

Figura 20. Vista creación Pin.

Figura 21. Vista pantalla Pin Incorrecto.

Este pin debe contener únicamente 4 dígitos, si se escriben mas dígitos durante el proceso de creación del pin la aplicación mostrara un mensaje de este error.

Si se intenta acceder introduciendo un pin incorrecto, la aplicación mostrara en la pantalla el mensaje “Pin incorrecto” como se puede observar en la *Figura 21. Vista pantalla Pin Incorrecto.*

Una ver introducido el pin correctamente la aplicación se redirige a la pantalla de bienvenida. Este pin, salvo modificaciones por parte del usuario de seguridad en el menú de configuración, será el que se utilice de ahora en adelante para los siguientes accesos.

En la pantalla de bienvenida se pueden realizar dos funciones. Se puede identificar la sala o se puede acceder al menú de configuración, este menú se localiza en la barra superior (ver *Figura 22.Vista pantalla de Bienvenida*).

Figura 22.Vista pantalla de Bienvenida.

Figura 23. Vista opciones menú de Configuración.

Si el usuario selecciona el menú de configuración, podrá desplegar un menú que le muestra todas las posibles modificaciones que el usuario puede hacer en los parámetros de configuración (ver *Figura 23. Vista opciones menú de Configuración.*).

Solo pueden hacer modificaciones los usuarios de seguridad, por este motivo cuando se accede al menú de configuración se solicitará el pin de la aplicación, como se muestra en la *Figura 24 Vista solicitud de Pin.*

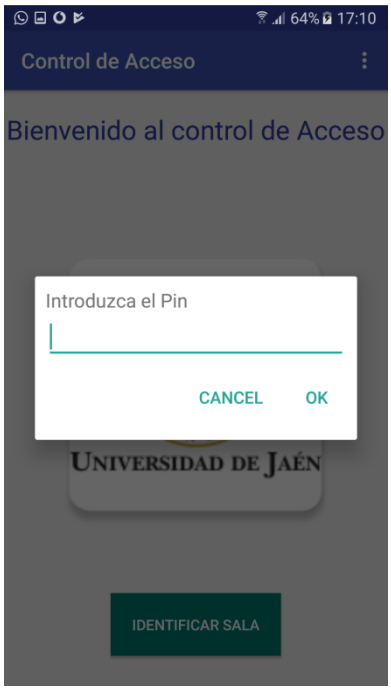

Figura 24 Vista solicitud de Pin.

Figura 25. Vista mensaje Pin Incorrecto.

Si el usuario introduce mal el pin, la aplicación no le permitirá hacer cambios ya que regresa a la página de bienvenida de la aplicación (ver *Figura 25. Vista mensaje Pin Incorrecto.*).

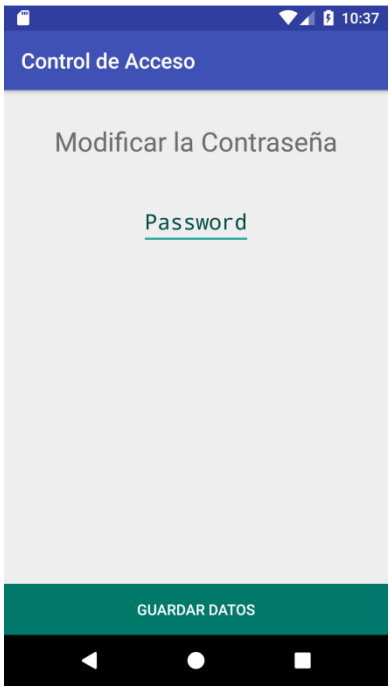

Figura 26. Vista formulario configuración.

Figura 27. Vista mensaje de modificación.

Cuando se seleccione una opción del menú de configuración y se introduzca el pin correctamente, se mostrará un formulario donde el usuario puede introducir el nuevo valor para el campo que ha solicitado modificar, como se muestra en la *Figura 26. Vista formulario configuración.*

Se puede modificar el usuario y la contraseña del usuario de seguridad, la url donde está alojada la aplicación servidora y el código pin de la aplicación.

Para que los datos se modifiquen una vez escritos los datos nuevos en el formulario de debe pulsar el botón “Guardar Datos”.

Si los datos se cambian correctamente se muestra el mensaje que se puede observar en la *Figura 27. Vista mensaje de modificación.*

Para que los usuarios puedan acceder a la sala, lo primero que se debe de hacer es identificar la sala, para ello hay que pulsar el botón “Identificar sala” de la pantalla de bienvenida.

Si es la primera vez que se intenta identificar una sala en la aplicación y el sistema operativo Android tiene una versión igual o superior a 6.0, la aplicación mostrara un mensaje en el cual el usuario debe otorgar los permisos necesarios a la aplicación para la lectura del IMEI del teléfono. El mensaje que se muestra se puede ver en la *Figura 28. Vista mensaje de permisos.* Si el usuario no otorga estos permisos no se podrá seguir utilizando la aplicación.

Figura 28. Vista mensaje de permisos.

Figura 29. Vista identificación de sala.

Si la sala se identifica correctamente, se muestra la pantalla de identificación de sala, en la cual se muestran los datos de la sala a la que el usuario intenta acceder. Los datos que se muestra son: el número de dependencia, la capacidad, el aforo actual y un plano con la información de seguridad de la sala (ver *Figura 29. Vista identificación de sala.*).

Esta es la pantalla que por defecto mostrara a los usuarios que lleguen a identificarse.

Si la sala no está registrada en la base de datos, la aplicación mostrara una imagen de error (ver *Figura 30. Vista error de identificación.*). Si se muestra este error la aplicación se detendrá, ya que solo aquellos dispositivos móviles cuyo IMEI este registrado en la base de datos de la aplicación servidora pueden utilizar esta aplicación.

Si la aplicación servidora no está disponible por cualquier motivo, le mostrara la pantalla que se puede ver en la *Figura 31.Vista error de la aplicación servidora.* Desde esta pantalla se puede volver a intentar hacer la identificación de la sala de nuevo.

Figura 30. Vista error de identificación.

Figura 31.Vista error de la aplicación servidora.

Si llega un usuario nuevo intentando acceder a la sala, debe pulsar el botón “Leer NFC”, la aplicación le mostrara la pantalla de lectura NFC, en esa pantalla la aplicación estará a la espera de que un usuario pase su tarjeta para leerla. (Ver Figura 32. Vista de pantalla *de lectura NFC.*).

Una vez el usuario acerque su tarjeta la aplicación le mostrara diferentes mensajes dependiendo de los permisos del usuario, el aforo de la sala y el estado de la aplicación servidora, estos mensajes son los que van a informar si un usuario puede o no acceder a la dependencia.

Los mensajes que se muestran son los siguientes:

1. Si el usuario acerca una tarjeta NFC con sus credenciales y estas credenciales no están registradas en la base de datos se mostrará la imagen de error que se observa en la *Figura 33: Vista de error de usuario.* Si se muestra este error es porque este usuario no está registrado en la aplicación servidora.

2. Si el usuario intenta entrar en la sala y se ha comprobado que tiene permisos y hay aforo disponible, el sistema mostrara la pantalla que se puede ver en la *Figura 34.Vista de registro de entrada.*
3. Si el usuario hace una petición para salir de la sala, y no se produce ningún error, se mostrará la *Figura 35.Vista de registro de salida.*
4. Si un usuario no tiene el permiso activado para entrar en la sala, el mensaje de error que mostrara la aplicación es el que se muestra en la *Figura 36.Vista de error de permisos de usuario.* En este caso para solucionar este error se debería activar el permiso de acceso en la aplicación servidora.
5. Si no hay creado en el sistema un permiso para que este usuario pueda entrar en la sala, se mostrara el mensaje de error que se puede ver en la *Figura 37.Vista de error de permisos en la sala*.
6. Si el usuario esta desactivado en el sistema se mostrara la pantalla de la *Figura 38.Vista de error de usuario desactivado.* Este error quiere decir que el usuario esta dado de baja del sistema.
7. Si la sala se desactiva y un usuario intenta acceder se mostrará el mensaje de error al que hace referencia la *Figura 39.Vista de error de sala desactivada.*
8. Si el aforo de la sala está completo se mostrará la pantalla de la Figura 41.Vista error de aforo completo.
9. Si la aplicación servidora no responde a las peticiones de la aplicación móvil se mostrará la pantalla de error de la Figura 40.Vista error la aplicación servidora. Si se encuentra ante este caso hay que pedir al departamento técnico que reinicie la aplicación servidora.

Pasados 5 segundos de la lectura de la tarjeta identificativa, la aplicación regresa a la pantalla donde se muestran los datos identificativos de la sala hasta que llegue otro usuario y pida hacer una nueva lectura.

Figura 32. Vista de pantalla de lectura NFC.

Figura 33: Vista de error de usuario.

Figura 34.Vista de registro de entrada.

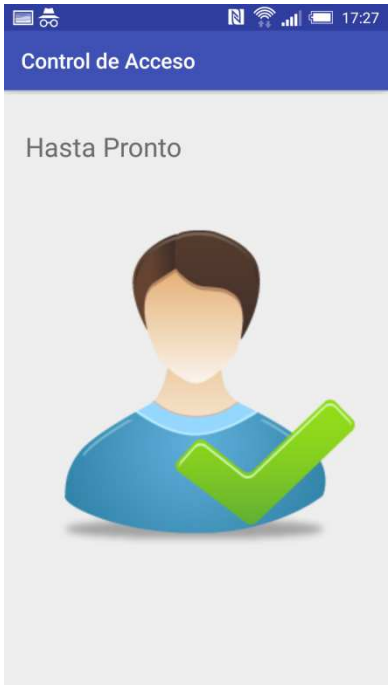

Figura 35.Vista de registro de salida.

Figura 36.Vista de error de permisos de usuario.

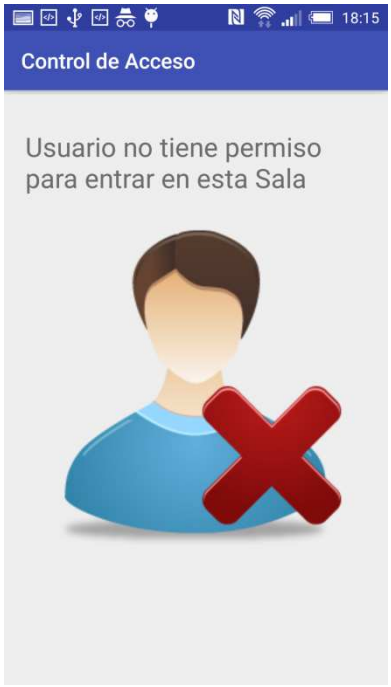

Figura 37.Vista de error de permisos en la sala

Figura 38.Vista de error de usuario desactivado.

Figura 39.Vista de error de sala desactivada.

Figura 40.Vista error la aplicación servidora.

Figura 41.Vista error de aforo completo.

6.2 Manual de administración de la aplicación servidora

Para acceder a la aplicación servidora los usuarios deberán poner en su navegador el dominio de la aplicación, en este caso el dominio es:

https://proyectoepsl.pythonanywhere.com/login/?next=/

Cuando intenten acceder a la aplicación lo primero que se va a mostrar es la pantalla de petición de autentificación, donde el usuario debe introducir el nombre y la contraseña que tenga asignada (ver *Figura 42.Vista formulario de autentificación.*)

Figura 42.Vista formulario de autentificación.

En la aplicación servidora hay gran cantidad de posibilidades de visualización según el perfil de permisos que tenga cada usuario.

En la *Figura 43.Vista usuario administrador* y en la *Figura 44.Vista usuario sin permisos.*, se pueden ver dos ejemplos de cómo puede cambiar la vista de la aplicación según los permisos que tenga un usuario u otro.

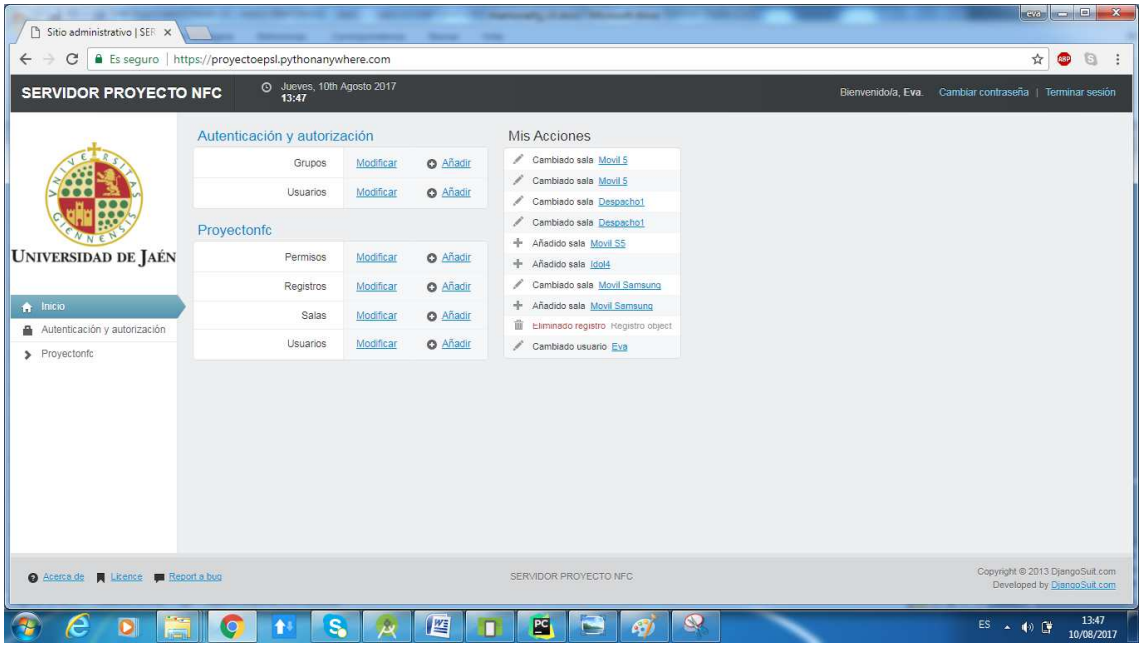

Figura 43.Vista usuario administrador.

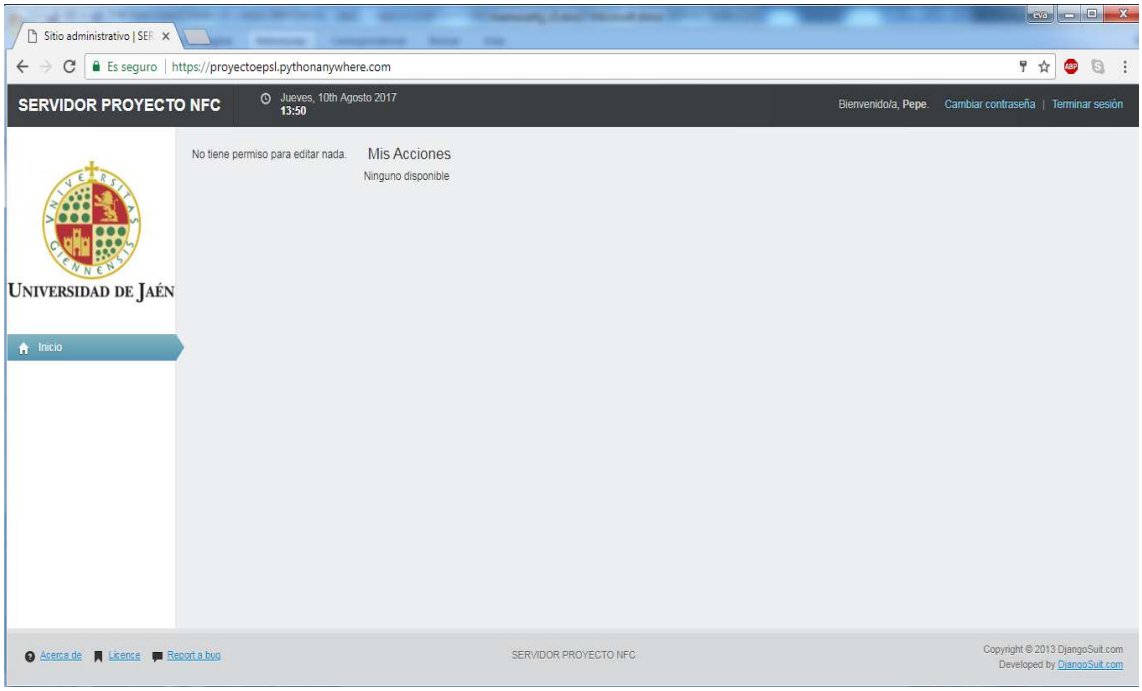

Figura 44.Vista usuario sin permisos.

En este manual se van a detallar todas las opciones que ofrece la aplicación servidora, se va a suponer que el acceso a la aplicación la realiza un usuario administrador para poder mostrar todas las opciones que ofrece la aplicación.

Según el perfil del usuario que accede a la aplicación tendrá que consultar una parte u otra de este manual.

6.2.1 Gestión de usuarios aplicación servidora

La parte de autentificación solo se mostrara a aquellos usuarios que tengan perfil de administración.

Para la gestión de usuarios, en la pantalla de inicio en el menú lateral, se debe elegir el apartado de **Autentificación y autorización**.

Dentro de este menú se encuentran dos opciones, la administración de los usuarios y la administración de los grupos.

6.2.1.1 Administración de usuarios

Si se pulsa la opción de Usuarios se ve un resumen de todos los usuarios que hay en este momento dados de alta en la aplicación servidora y sus datos personales (ver *Figura 45.Vista resumen de la administración de usuarios.*).

También se pueden ver los usuarios que pueden entrar en la plataforma, ya que solo aquellos que tengan permiso podrán acceder a la aplicación servidora.

En la parte superior hay un buscador que permite buscar a los usuarios por su nombre de usuario, el email, nombre y apellidos. También permite establecer filtros de búsqueda para el usuario, como por ejemplo si esta activo o no.

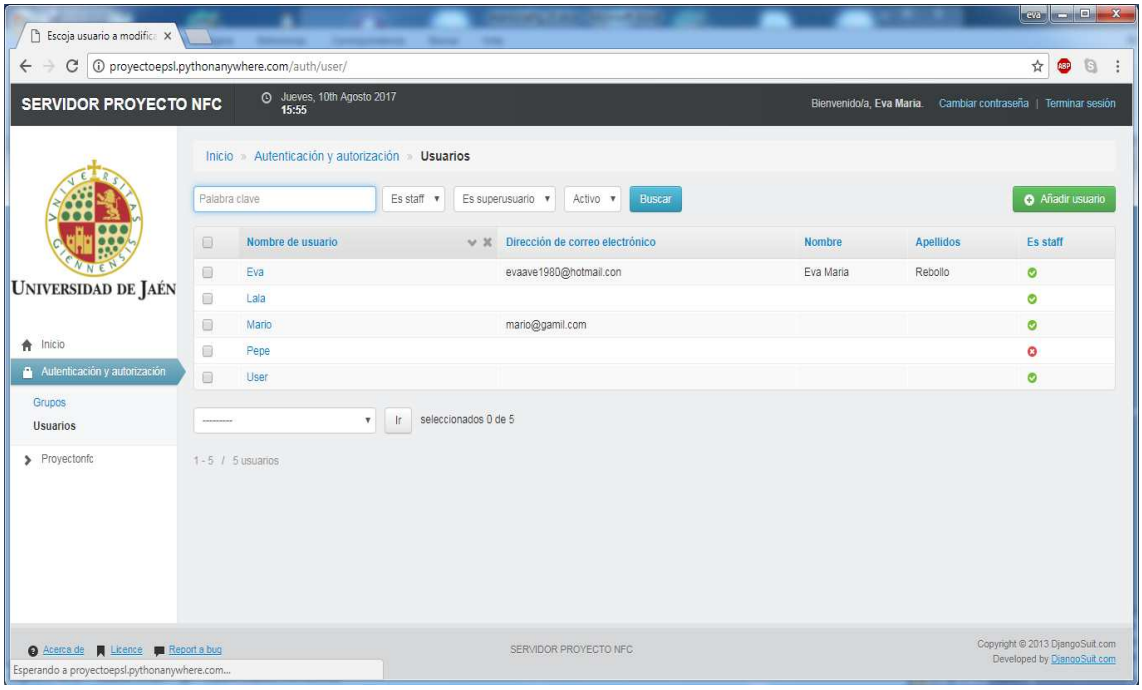

Figura 45.Vista resumen de la administración de usuarios.

Si se desea crear uno nuevo, se debe pulsar el botón de color verde que se encuentra en la parte superior derecha de la *Figura 45.Vista resumen de la administración de usuarios.*

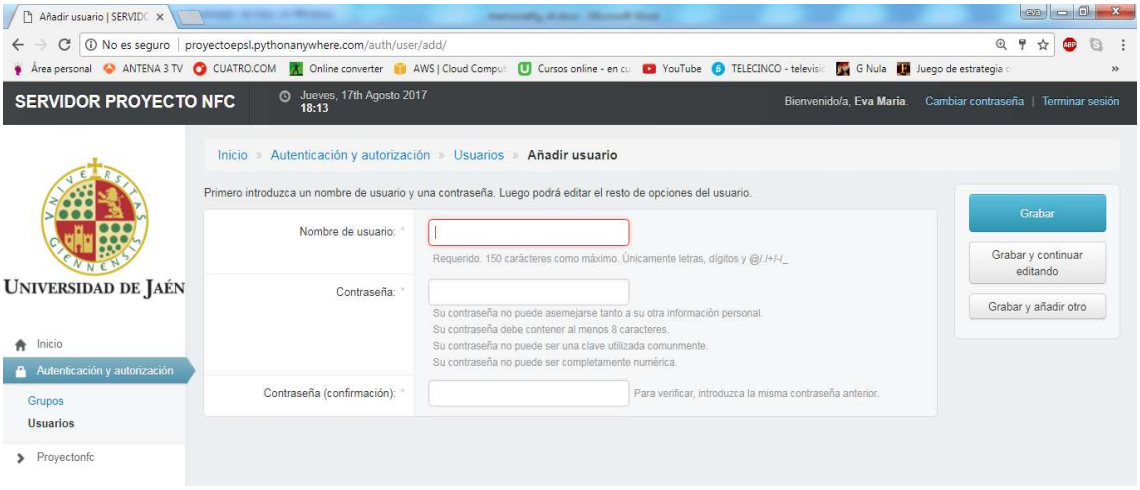

Figura 46.Vista de botón añadir usuario.

Para crear un usuario se deben definir el nombre de usuario y la contraseña (ver *Figura 46.Vista de botón añadir usuario*).

El nombre de usuario puede contener como máximo 150 caracteres y puede estar compuesto por letras y/o números. Solo se admiten los siguientes caracteres especiales: @, punto, guion medio o bajo y los signos más y menos.

La contraseña tiene que tener al menos 8 caracteres y tiene que ser alfanumérica.

Una vez creado el usuario se muestra la pantalla que se puede ver en la *Figura 47.Vista de datos personales del usuario.*

En esta pantalla se puede introducir información personal del usuario como su nombre, apellidos o email y también se pueden definir los permisos del usuario.

Figura 47.Vista de datos personales del usuario.

El apartado de los permisos que se le pude asignar a cada usuario se muestra en la *Figura 48.Vista de permisos de usuario.*

Si se desea que el usuario este activo, se debe marcar el campo “activo”.

Si se desea que el usuario pueda entrar en el sitio administrativo se debe marcar el campo “Es Staff”.

Si se desea otorgar al usuario todos los permisos como administrador se debe marcar el campo “Superusuario”.

Si se quiere asignar a un usuario a un grupo, se debe elegir a qué grupo se le quiere asignar en el campo grupos disponibles y para incluirlo, con las flechas que aparecen, hay que pasarlo de la opción de grupos disponibles a la opción de grupos elegidos. Se puede asignar uno o varios grupos a un mismo usuario. Un usuario tendrá todos los permisos asignados a cada uno de los grupos al que pertenezca.

También se puede elegir asignar permisos particulares a cada usuario, esto se realiza en el campo permisos de usuarios. Se hace igual que para seleccionar a que grupos pertenece un usuario, se selecciona el permiso que se quiere asignar, del campo permisos de usuarios disponibles, se arrastra, con ayuda de las flechas, a la casilla de permisos de usuario elegidos.

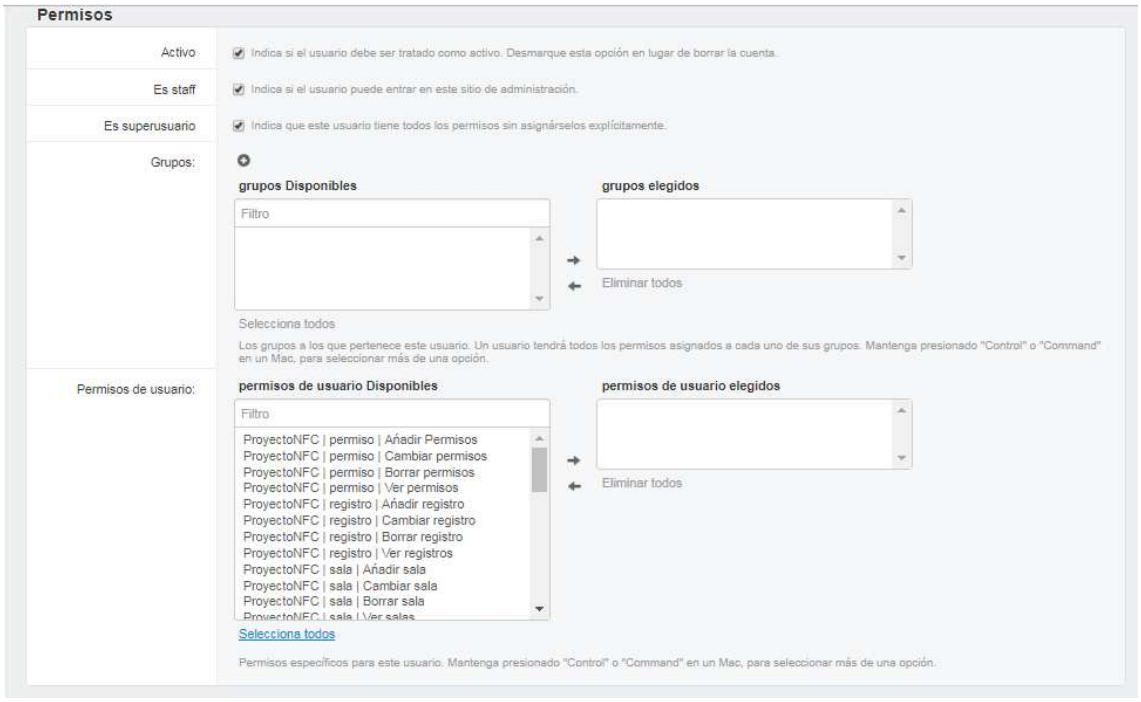

Figura 48.Vista de permisos de usuario.

Por último, para guardar los datos se deben pulsar uno de los tres botones que aparecen en el lateral izquierdo (ver *Figura 49.Vista de botones para guardar y eliminar.*).

Cada botón va a permitir llevar a cabo una acción:

- Guardar: guarda los datos y pasa a la vista resumen de los usuarios.
- Guardar y continuar editando: permite ir guardando datos a medida que se van introduciendo y luego poder seguir editando.
- Guardar y añadir: permite guardar los datos del usuario que se está editando y añadir un nuevo usuario a continuación.

También está la opción de eliminar un usuario previamente creado.

Figura 49.Vista de botones para guardar y eliminar.

Si se desea modificar la contraseña debe seleccionar el enlace que aparece en la parte de abajo del apartado de Contraseña. En la pantalla de cambiar contraseña el usuario debe introducir la nueva contraseña que quiere establecer (ver *Figura 50.Vista de cambiar contraseña.*)

Figura 50.Vista de cambiar contraseña.

6.2.1.2 Administración de grupos

Se pueden crear grupos de usuarios para determinar el perfil de acceso de varios usuarios a la vez (ver *Figura 51.Vista para la creación de grupos.*)

Para crear un grupo de usuarios hay que indicar un nombre de grupo y los permisos específicos que va a tener ese grupo. Para elegir que permisos tendrá ese grupo, se debe seleccionar el permiso del apartado permisos disponibles y pasarlo, con ayuda de las flechas, al apartado de permisos elegidos.

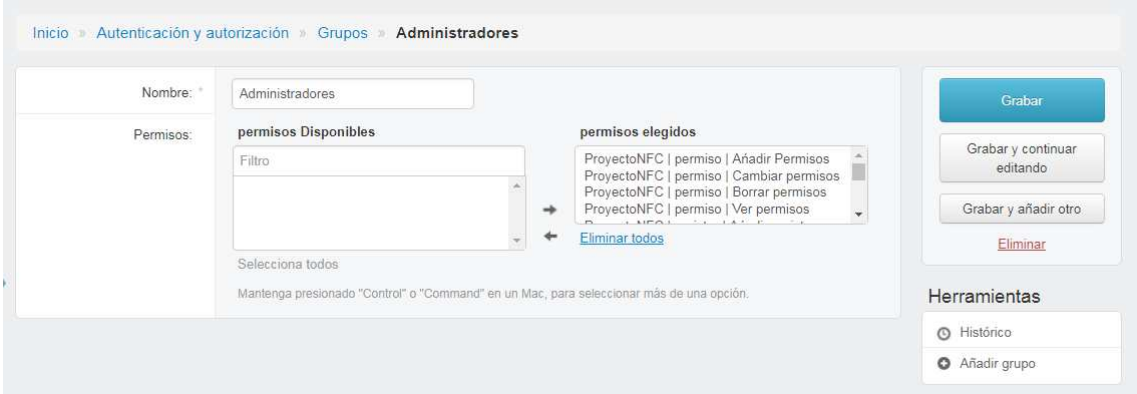

Figura 51.Vista para la creación de grupos.

6.2.2 Gestión de configuración de la aplicación servidora

En la pestaña Proyectonfc que se muestra en el menú lateral de la aplicación, se pueden gestionar todos los valores de configuración que va a tener la aplicación móvil de control de acceso (ver *Figura 52.Vista de menú de Proyectonfc.*).

Figura 52.Vista de menú de Proyectonfc.

6.2.2.1 Configuración de permisos

En este apartado se tienen que definir los usuarios que tendrán acceso a cada sala.

Cuando se selecciona la pestaña permisos, aparecerá un resumen de los permisos otorgados hasta el momento (ver *Figura 53.Vista de resumen de permisos.*).

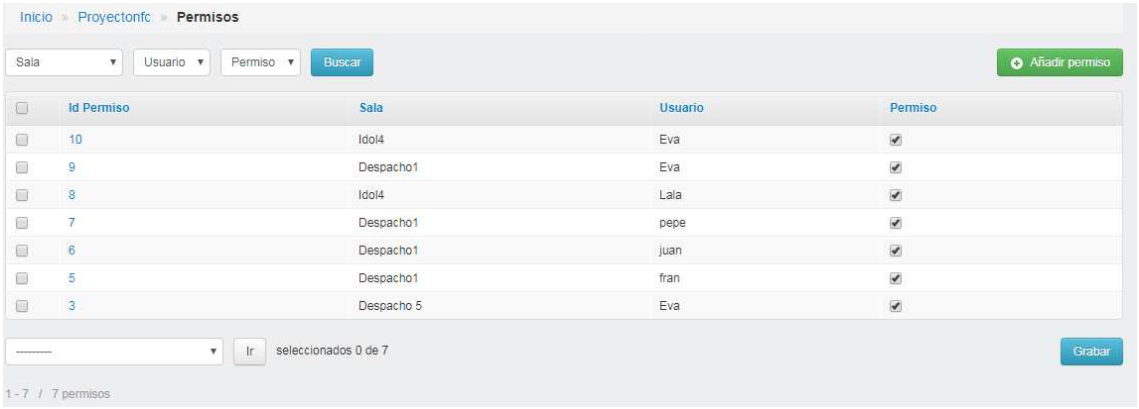

Figura 53.Vista de resumen de permisos.

En esta pantalla resumen en la parte superior también se incorpora un buscador que permite realizar búsquedas por el nombre de sala o usuario y por el valor del campo permiso.

Desde esta pantalla se pude seleccionar si un permiso va a estar activo o no, marcando o dejando sin marcar la columna permiso.

Si se desea dar de alta un nuevo permiso, se deberá pulsar el botón verde que aparece en el lateral derecho denominado “Añadir Permiso”. La pantalla para dar de alta un nuevo permiso se puede observar en la *Figura 54.Vista para añadir permisos.*

Para definir un permiso se debe seleccionar en el campo “Sala” el nombre de la sala que se quiere tratar. Se puede crear o editar una sala pulsando en los iconos de lápiz y más, que se muestran al final del formulario.

Se debe seleccionar el usuario al que se quiere otorgar el permiso, se elige del desplegable que aparece en el campo “Usuario”, también se puede crear o editar un usuario directamente desde esta pantalla.

El campo permiso determinara si el permiso otorgado esta activo o no, si esta marcado este campo el permiso esta activado.

Por último se debe guardar el permiso creado utilizando los botones que se muestran en la columna de la derecha.

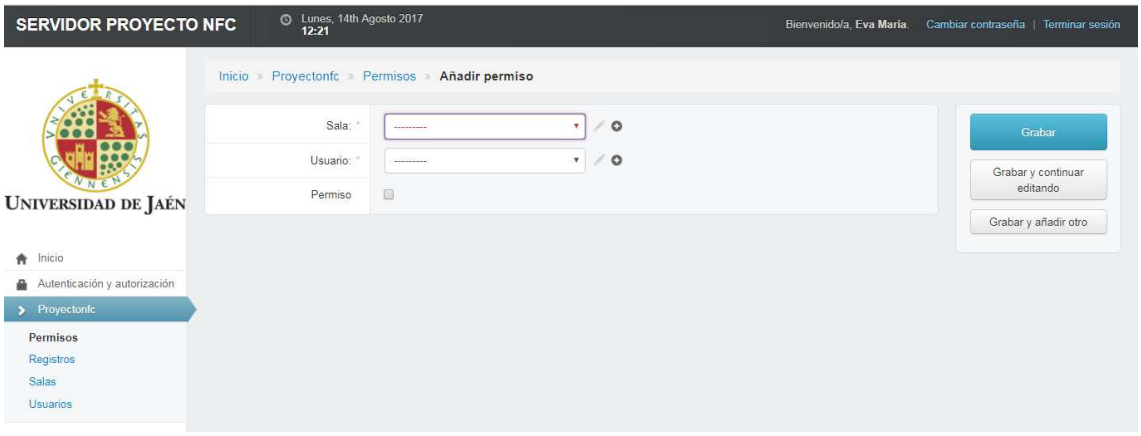

Figura 54.Vista para añadir permisos.

6.2.2.2 Configuración de registros

Para acceder a la pantalla de resumen de todos los registros, que se han realizado en el sistema, se debe seleccionar el enlace denominado “Registro”, este enlace se encuentra en el menú de la izquierda de la pantalla inicial de la aplicación.

En la vista de resumen de todos los registros que se han contabilizado en la aplicación se pueden hacer filtros de búsquedas por sala, usuarios, fechas de entrada, salida y si el registro está terminado o no. Este buscador está en la parte superior de la vista resumen (ver Figura 55.Vista resumen de registros.)

Los registros finalizados en la columna “Terminado” muestran el cuadro de comprobación marcado, si este campo aparece sin marcar quiere decir que el usuario aun permanece en el interior de la sala.

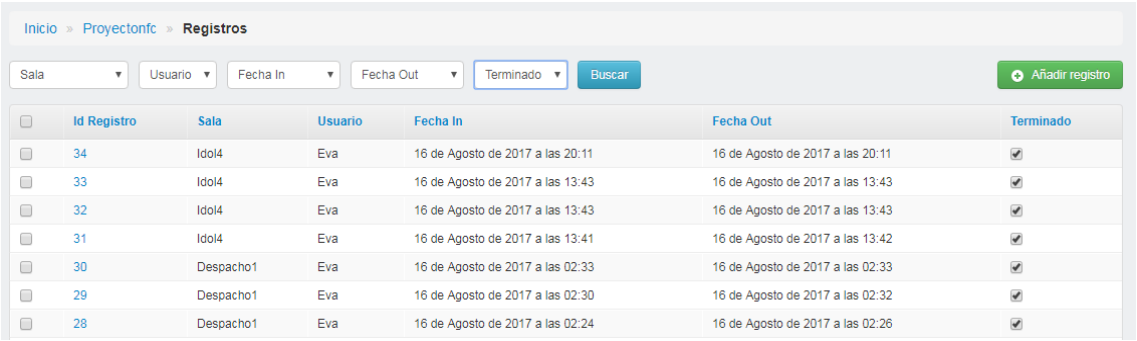

Figura 55.Vista resumen de registros.

Para poder modificar los parámetros de cualquier registro hay que acceder a la pantalla de modificación, para ello hay que pulsar el enlace que aparece en el valor de la columna “IdRegistro”.

Esta pantalla es similar a la de añadir registro, en ella se puede modificar el nombre de la sala, del usuario, establecer el valor de la variable Fecha Out y la variable terminado.

El valor de la variable Fecha In se almacena automáticamente en el momento que se crea un registro y no se puede modificar (ver Figura 56.Vista para añadir y modificar registros.).

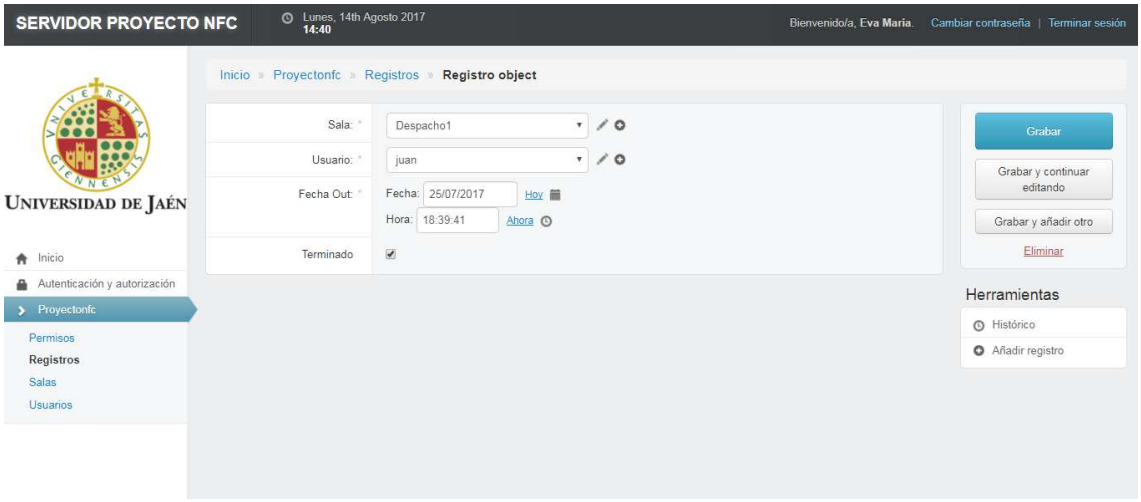

Figura 56.Vista para añadir y modificar registros.

También existe la posibilidad de añadir un nuevo registro seleccionando el botón verde que aparece en la pantalla de resumen.

6.2.2.3 Configuración de salas

Para acceder a la configuración de las salas que conforma el recinto de acceso a controlar, es necesario seleccionar el enlace de la columna de la izquierda denominado Salas.

La primera pantalla que aparece es el resumen de toda la información relativa a las salas que hay ya grabadas en el sistema (ver *Figura 57.Vista resumen de salas.*).

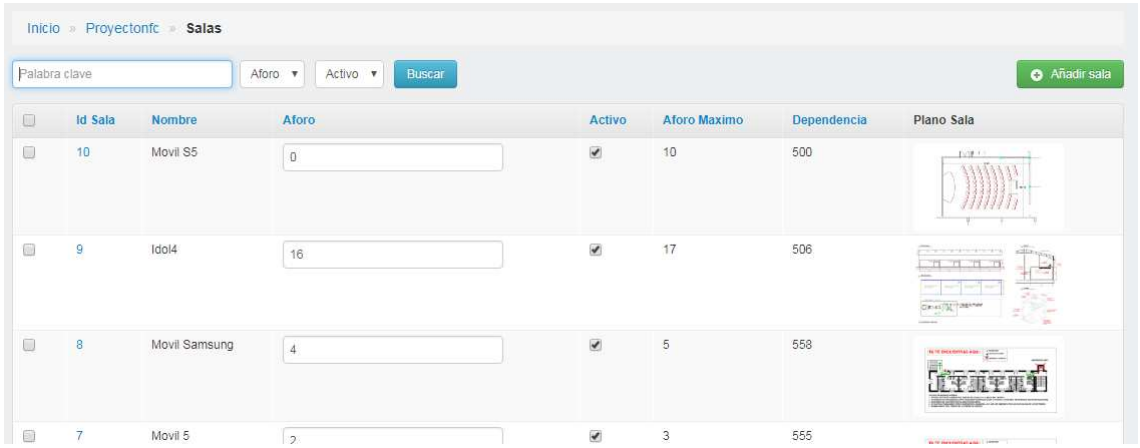

Figura 57.Vista resumen de salas.

En la parte superior hay un buscador que permite buscar salas por el nombre de una sala, el IMEI o el número de dependencia al que corresponde.

También se pueden crear filtros de búsqueda por el aforo y el valor de la columna "activo".

En esta pantalla se pude modificar también el estado de una sala, si esta activa el cuadro de comprobación de la columna “Activo” estará seleccionado, si esta desactivada el cuadro de comprobación estará vacio.

Desde esta pantalla resumen se podrá activar o desactivar una sala o modificar el aforo actual de la sala.

Si se quiere modificar algún dato de una sala se puede acceder a la pantalla que permite modificar los datos de la sala pulsando sobre el enlace que hay en el valor de la columna “IdSala”. La pantalla donde puedo modificar los datos de la sala se puede ver en *Figura 58.Vista para añadir y modificar datos de salas.* Esta pantalla es la misma que se utiliza para añadir una sala nueva.

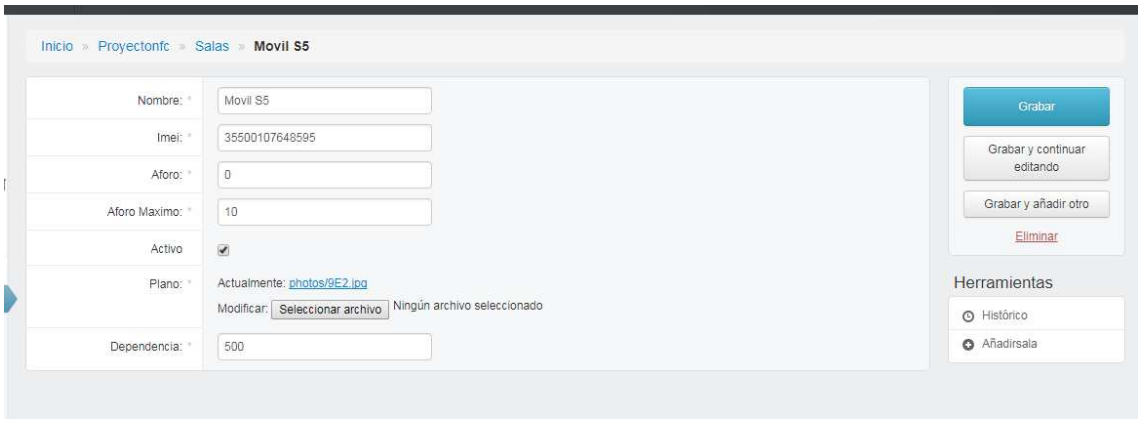

Figura 58.Vista para añadir y modificar datos de salas.

Los datos que hay que introducir son el nombre de la sala, el IMEI del teléfono, el aforo actual de la sala, el aforo máximo de la sala, la imagen del plano de emergencia y el número de dependencia de la sala.

El aforo actual cuando se da de alta una sala se debe de establecer a 0 ya que no hay nadie dentro de la sala.

No conviene que se modifique este parámetro ya que es el que va a controlar el número de personas que hay en el interior de la sala.

El tipo de formato de imagen que se pueden subir a la aplicación servidora es jpg o png. Se puede ver la imagen de la sala en su tamaño original pulsando en el enlace que hay encima de la opción de seleccionar imagen.

6.2.2.4 Configuración de usuarios

Esta opción se selecciona en la columna de la izquierda de la vista principal de la aplicación. Una vez es seleccionada muestra un resumen de todos los usuarios que pueden acceder a la aplicación móvil identificándose con sus tarjetas NFC (ver *Figura 59.Vista resumen de usuarios.*).

Desde la pantalla resumen se pueden hacer búsquedas por el nombre del usuario, el apellido o el DNI, también se puede hacer un filtro de búsqueda para ver que usuarios están activos o no.

Figura 59.Vista resumen de usuarios.

Desde la pantalla resumen se puede modificar si un usuario esta activo o no en el sistema. Para activar o desactivar un usuario, se utiliza el campo “activo”, para que un usuario este activo debe estar seleccionado el cuadro de comprobación de la columna activo.

La pantalla para modificar los datos de usuario coincide con la pantalla que se utiliza para dar de alta un nuevo usuario y se puede observar esta vista en la *Figura 60.Vista para añadir o modificar un usuario.* Los campos que hay que rellenar son el nombre y apellidos del usuario, el número de DNI y seleccionar si esta activo o no.

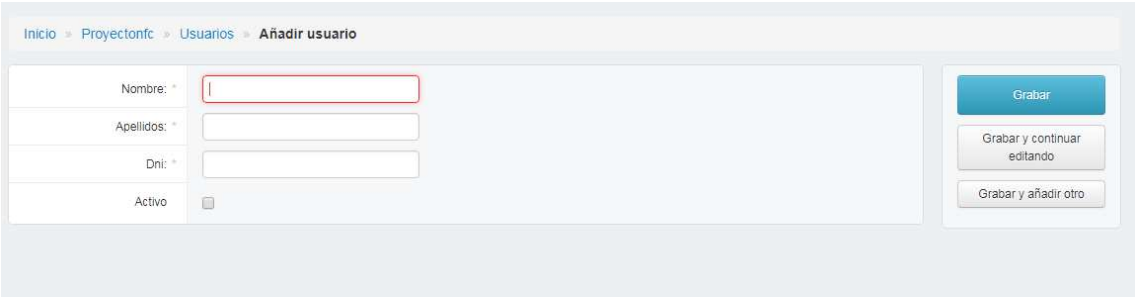

Figura 60.Vista para añadir o modificar un usuario.

El número de DNI hay que escribirlo con la letra en mayúsculas para evitar posibles errores de identificación.

6.3 Manual de configuración tarjetas NFC de los usuarios

Los usuarios para identificarse deberán de disponer de una tarjeta NFC con sus datos almacenados en ella.

Las tarjetas que se utilicen deben tener las especificaciones técnicas indicadas en el apartado de este proyecto denominado Características técnicas de la tarjeta NFC.

Para hacer la escritura de la tarjeta, se ha utilizado la aplicación **NFC TagWriter**[18] la cual está disponible, en su versión gratuita, en la aplicación Play Store disponible para los teléfonos Android.

Una vez se inicia la aplicación la pantalla inicial muestra todas las opciones que esta aplicación permite llevar acabo (ver la *Figura 61. Vista opciones de TagWriter.*). Esta aplicación permite obtener elementos externos como son un archivo CSV o copiar los datos de otra tarjeta.

Para la escritura de las tarjetas NFC que utiliza la aplicación móvil, hay que seleccionar la opción de crear un Nuevo Elemento, esta opción se puede observar en la *Figura 62. Vista de opciones de escribir.*

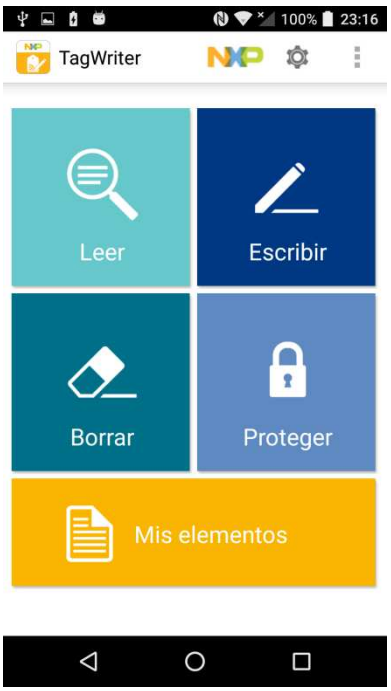

Figura 61. Vista opciones de TagWriter.

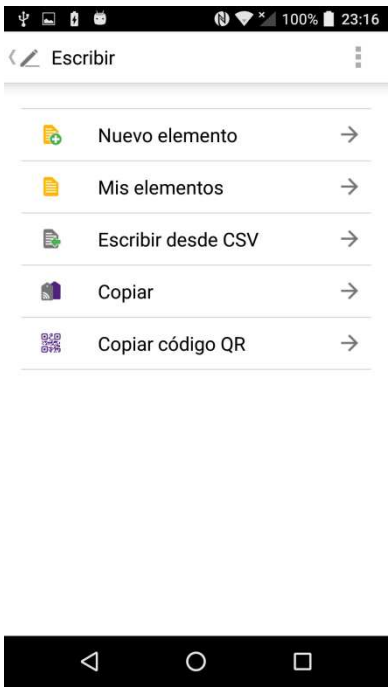

Figura 62. Vista de opciones de escribir.

El formato usado para escribir las tarjetas identificativas es tipo texto, para ello hay que seleccionar el tipo de elemento “Texto plano”,el cual esta disponible entre los elementos que ofrece dicha aplicación. Todos los tipos de elementos que pueden escribir con esta aplicacion se pueden ver en la *Figura 63. Vista opciones de tipo de elemento.*

Para que no se produzca ningún problema en la lectura de los datos desde la aplicación móvil, es necesario que se tenga especial cuidado del formato de texto que hay que utilizar para escribir los datos.

[18]Disponible:https://play.google.com/store/apps/details?id=com.nxp.nfc.tagwriter&hl=es .

En las tarjetas hay que escribir el nombre, los apellidos y el DNI del usuario al que pertenece la tarjeta, cada dato en una línea tal y como se puede observar que se ha realizado en la *Figura 64. Vista de formato de escritura de datos.* Es necesario que no se introduzca ningún espacio al final de la línea y que la letra del DNI del usuario se escriba en mayúscula.

Para escribir la tarjeta hay que pulsar el botón de guardar y escribir.

El programa ofrece varias opciones, las cuales se pueden observar en la Figura 65. Vista de petición de escritura.

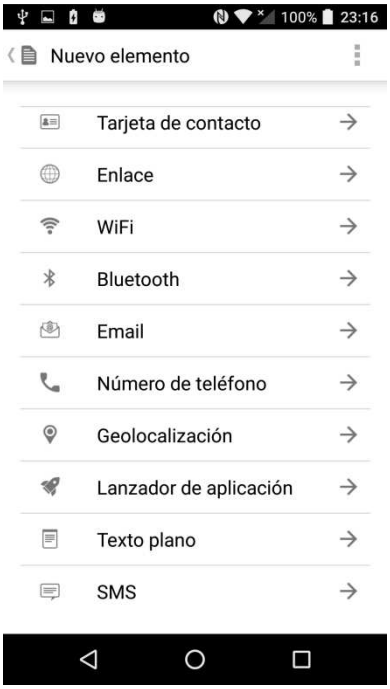

Figura 63. Vista opciones de tipo de elemento.

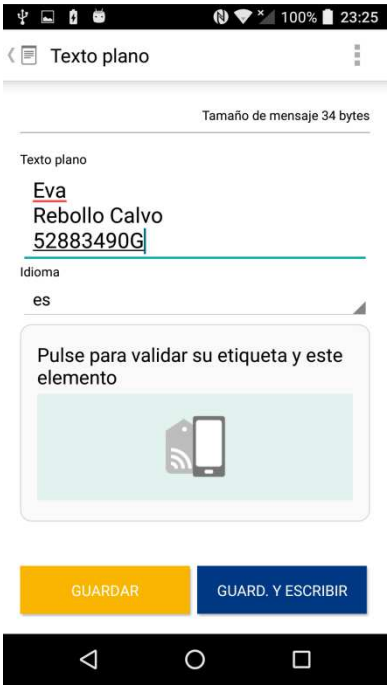

Figura 64. Vista de formato de escritura de datos.

Si la escritura se ha realizado correctamente en la tarjeta, el programa en el campo Resultado mostrara el mensaje “Escritura Correcta” (ver la Figura 66. Vista de confirmación de escritura.).

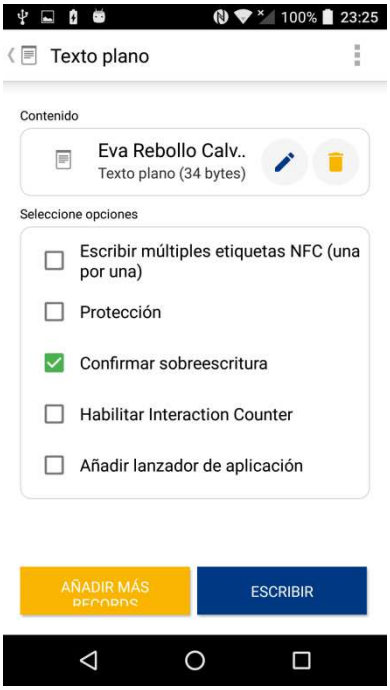

Figura 65. Vista de petición de escritura.

Figura 66. Vista de confirmación de escritura.

6.4 Manual de mantenimiento de la aplicación servidora

La aplicación servidora durante el proceso de desarrollo estaba disponible para hacer pruebas dentro de la red local, pero esta opción no es válida para el contexto en el que se ha desarrollado esta aplicación, cuyo fin es que pueda ser utilizada en cualquier recinto que no disponga de infraestructura para instalar un control de acceso.

Para que este objetivo se pueda cumplir, se ha decido desplegar la aplicación servidora en un alojamiento web.

El alojamiento web que se ha escogido es ***PhytonAnywhere***[19], ya que es un proveedor gratuito que cumple con las características técnicas detalladas en el apartado 7.1.3 y porque el despliegue de una aplicación Django en este alojamiento es un proceso relativamente sencillo, ya que cuanta con mucha documentación.

El otro servicio externo que se ha empleado es ***GitHub***[20] que es un servicio de alojamiento de código gratuito. Este servicio se ha utilizado tanto para implementar un control de versiones durante el desarrollo, para evitar la pérdida de información, como para desplegar las diferentes versiones al alojamiento web.

6.4.1 Implementación de GitHub

Para utilizar una cuenta en GitHub lo primero que hay que realizar es darse de alta, para ello, desde la pantalla de inicio de la pagina web https://github.com/, se puede

[19] Disponible en : https://www.pythonanywhere.com/.
[20] Disponible en: https://github.com/.

crear una nueva cuenta indicando, un nombre de usuario, una contraseña y un email de usuario.

El segundo paso es seleccionar un tipo de plan, en este caso se ha escogido la opción gratuita por que no es necesaria la opción de pago.

Por último hay que responder a varias preguntas en relación al motivo del por qué se crea esta cuenta.

Una vez se hayan registrado todos los datos para poder usar la cuenta hay que validarla, a través del enlace que se recibe en el email.

Para poder empezar a desplegar el código en el servidor es necesario crear un repositorio nuevo (ver Figura 67.Vista de creación de repositorio en GitHub.), el nombre del repositorio será el elegido para el proyecto. En este caso sería el repositorio de la aplicación servidora. Cuando el repositorio se creer estará vacio, no contendrá ningún dato.

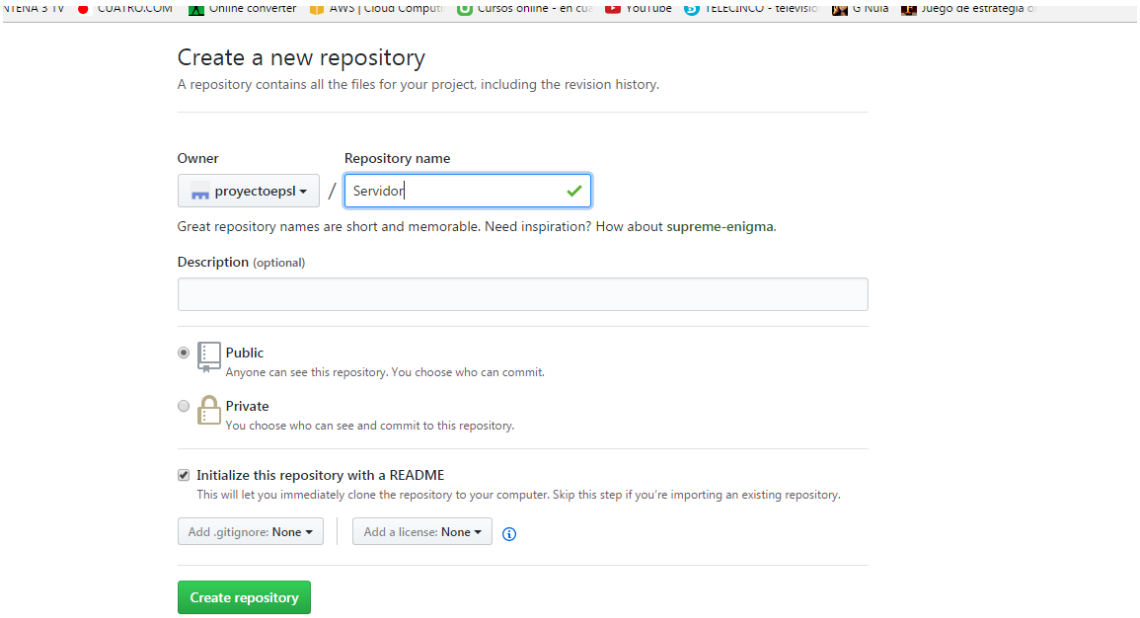

Figura 67.Vista de creación de repositorio en GitHub.

Para poder subir el código desde el entorno de desarrollo utilizado hasta el repositorio es necesario configurar, en las opciones del entorno de desarrollo, el servidor *GitHub*.

Para el desarrollo de esta aplicación servidora el entorno de desarrollo que se ha utilizado es ***PyCharm***[21], que es un entorno que permite trabajar con Django.

[21] Disponible en : https://www.jetbrains.com/pycharm/download/#section=windows

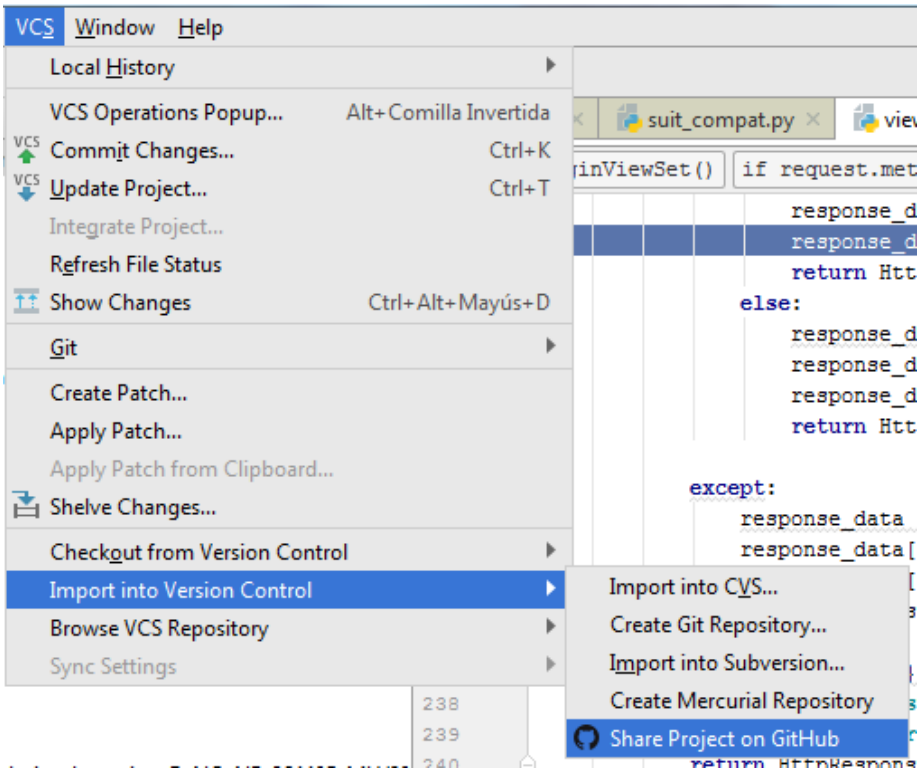

Figura 68.Vista de opción de configuración de GitHub en PyCharm

En este entorno se configura el control de versiones elegido en la opción `VCS>Import into Version Control >Share Project on GitHub`.

En esta opción es necesario indicar un nombre para el repositorio con el que vamos a sincronizar nuestro código, la ruta del repositorio (Host) y las credenciales para entrar en el repositorio.

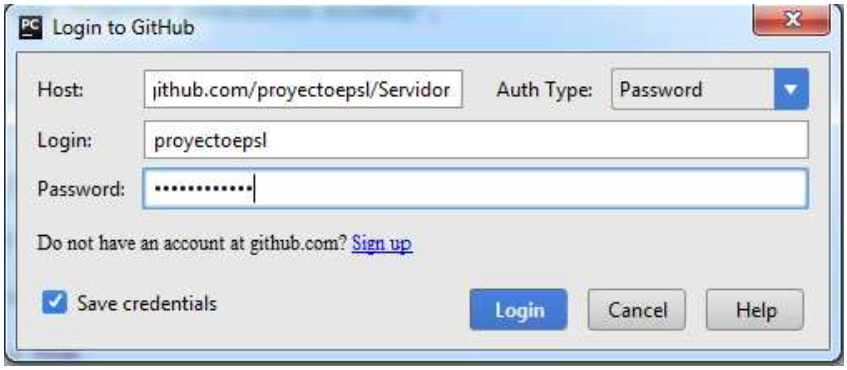

Figura 69. Vista de peticion de credenciales.

Una vez este configurado el servicio, para poder subir las nuevas versiones del codigo que se vayan generando durante el desarrollo, hay que pulsar el boton derecho y seleccionar la opcion `Git>Commit Directory` (ver en la *Figura 70.Vista opción de Commit desde el entorno de desarrollo.*). Hay que ejecutar `Commit` y despues `Push` para que la informacion se suba a nuestro repositorio en internet.

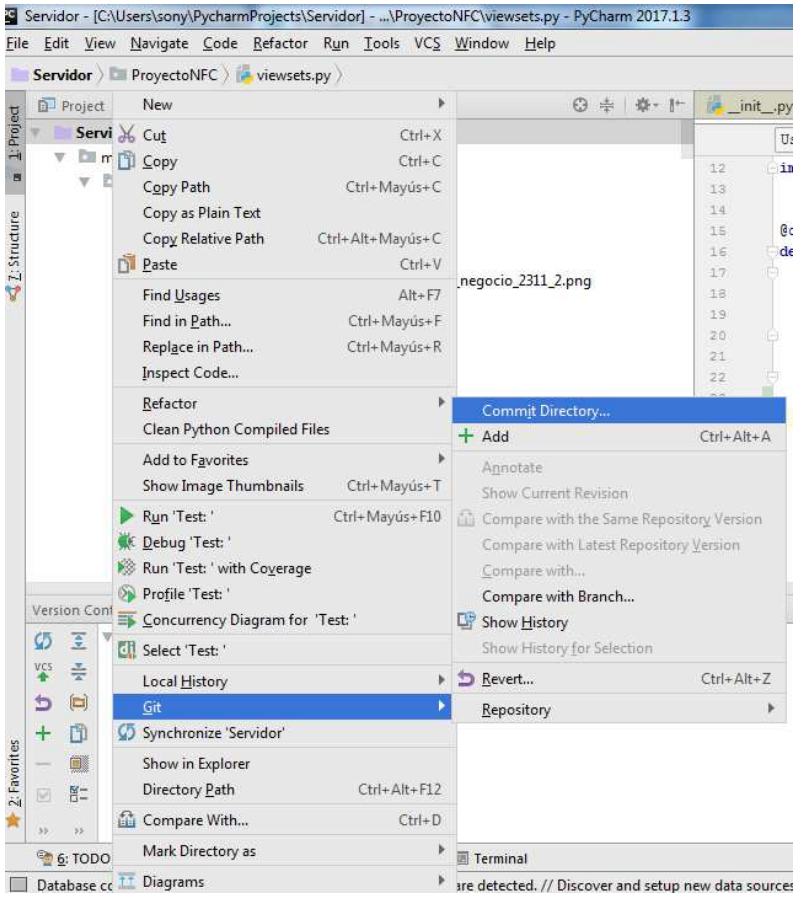

Figura 70.Vista opción de Commit desde el entorno de desarrollo.

6.4.2 Implementación del alojamiento Web

Para poder desplegar la aplicación en el alojamiento web, lo primero que hay que hacer es dar de alta una cuenta en dicho alojamiento.

Para crear esta nueva cuenta los pasos a seguir son los siguientes.

1. Pulsar el enlace “Start running Python online in less than a minute” que aparece en la pantalla inicial de la pagina web de este servicio.
2. En la siguiente pantalla para crear una cuenta gratuita se debe seleccionar el enlace que se denomina “Create a Beginner account”.
3. Los datos requeridos para dar de alta la cuenta son: un nombre de usuario, una dirección de email y una contraseña.

Una vez creada la cuenta el sistema mostrara la pantalla de configuración del alojamiento web que se quiere crear. Esta pantalla se puede ver en la Figura 71.Vista de la pantalla inicial de configuración de pythonanywhere.).

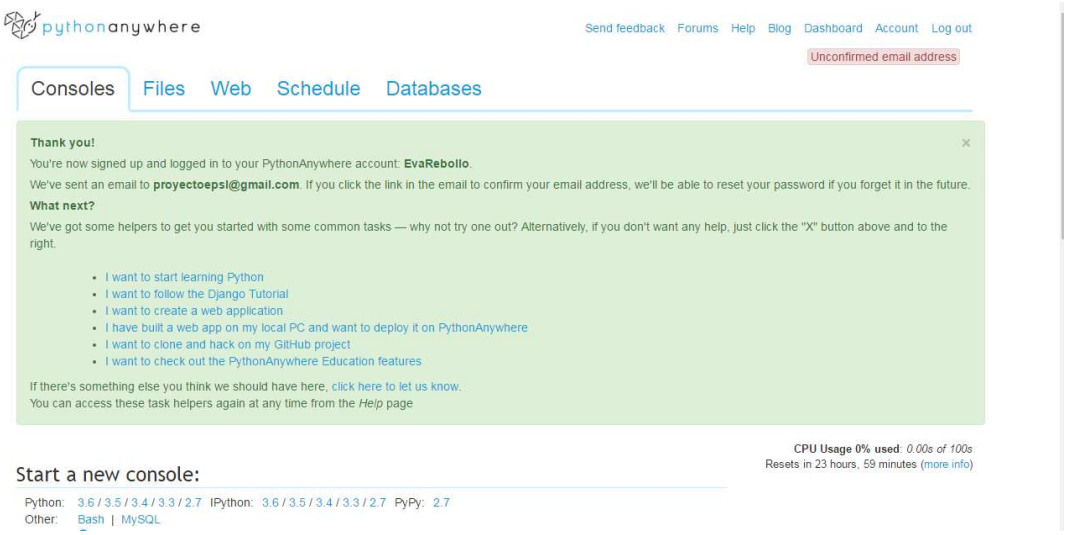

Figura 71.Vista de la pantalla inicial de configuración de pythonanywhere.

Para llevar acabo los siguientes pasos de configuración del alojamiento web es necesario que se inicie una consola de administración en el alojamiento web, para ello hay que seleccionar la pestaña “Console”. En esta pantalla para ejecutar la consola hay que pulsar la opción “Bash” que se encuentra en la sección “Star a new console” (ver Figura 72.Acceso a la consola del alojamiento web.)

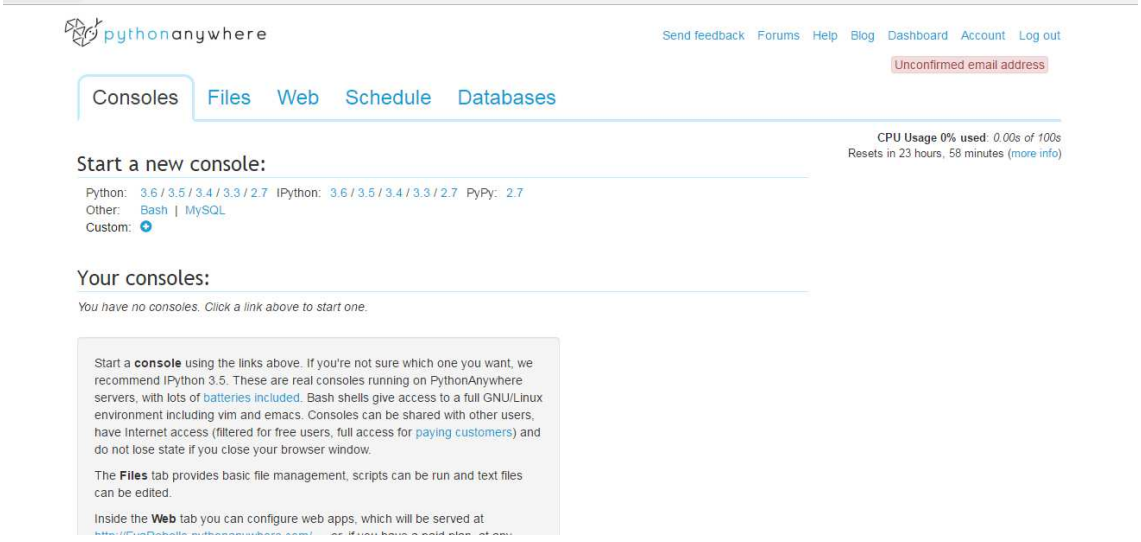

Figura 72.Acceso a la consola del alojamiento web.

El siguiente paso es clonar la copia de la aplicación servidora que se encuentra guardada en el repositorio GitHub. Esta copia se debe clonar al directorio del alojamiento web, haciendo uso de su consola de administración. Para hacer la clonación se debe ejecutar en la consola el comando git clone seguido de la URL del repositorio, la ejecución de este comando se puede ver en la Figura 73.Vista de la clonación del repositorio.

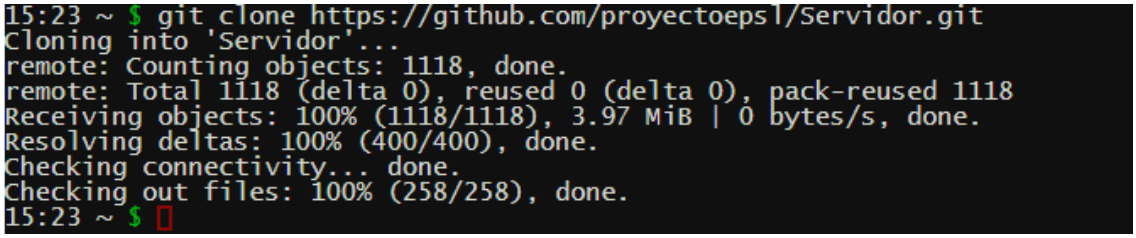

Figura 73.Vista de la clonación del repositorio.

Para la creación del dominio del sitio web se debe seleccionar la pestaña “Web” de la pantalla inicial del alojamiento web y pulsar el botón “Add a new web app”. La dirección del dominio estará formada, en la versión gratuita, por el nombre del usuario seguido de pythonanywhere.com.

1. En la pantalla que permite elegir el tipo de configuración que se va a llevar a cabo en el sitio web, hay que elegir la configuración manual. Esta pantalla tiene por título “Select a Python Web Framenword”.
2. En la siguiente pantalla titulada “Select a Python Version”, se debe de seleccionar la versión 3.6 de Python, ya que es esta versión la que se ha elegido para crear el proyecto.

Una vez creado el dominio hay que configurar todos los parámetros de la aplicación en la pestaña “Web” (ver Figura 74.Vista pantalla Web del alojamiento web.). En esta vista se puede observar el nombre del dominio en la parte superior de la pantalla en el apartado “Configuration for”.

También es importante indicar que en la pestaña “Web” es donde se va a poder reiniciar el dominio, cuando se realice cualquier cambio, en el apartado “Reload”. Para reiniciar el servicio hay que pulsar el botón de color verde, cada vez que se quiera forzar un reinicio del dominio.

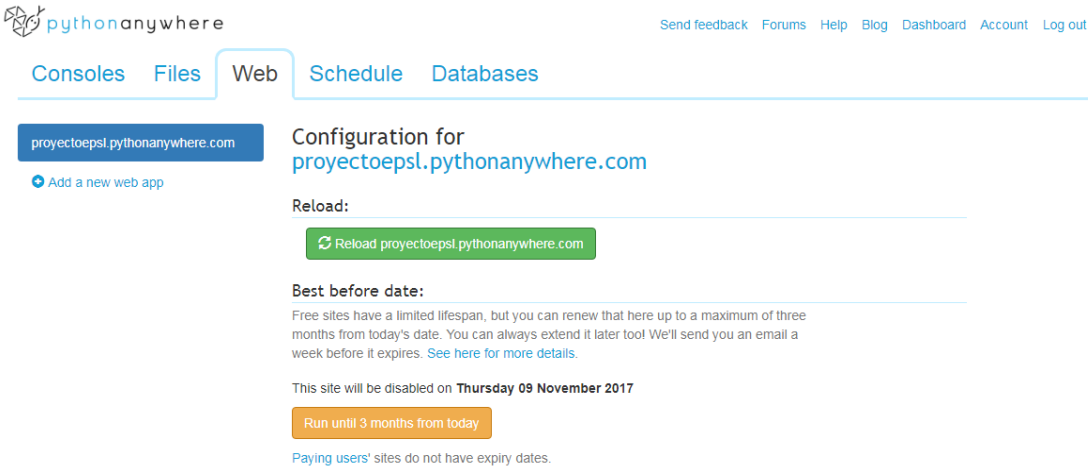

Figura 74.Vista pantalla Web del alojamiento web.

En esta misma pestaña en el apartado “Code”, se debe configurar las rutas de los directorios que conforman el proyecto. En el apartado “Source Code” se indica la carpeta

raíz donde se encuentra el proyecto y en el apartado “Working Directory” la carpeta donde se encuentran los archivos de configuración del proyecto (ver Figura 75.Vista de los parámetros de configuración del directorio de trabajo).

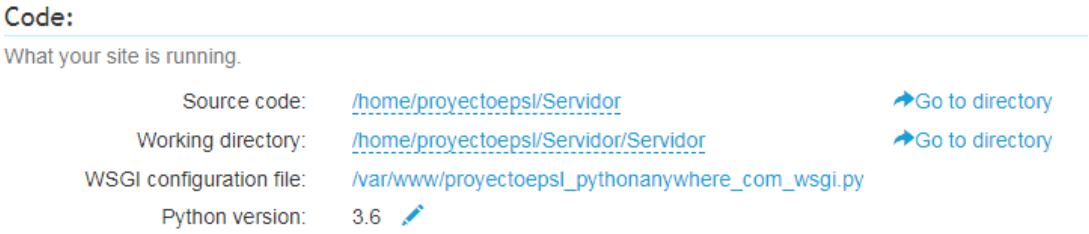

Figura 75.Vista de los parámetros de configuración del directorio de trabajo.

En el apartado “Code” se encuentra también el enlace al archivo de configuración WSGI.py del proyecto en la opción “WSGI configuration file”. En este archivo se debe de indicar en el apartado DJANGO el directorio donde se aloja la carpeta raíz del proyecto y donde se encuentra el archivo de configuración settings.py (ver Figura 76.Vista de modificaciones en el archivo WSGI.)

```
# ++++++++++++ DJANGO ++++++++++++
# To use your own Django app use code like this:
import os
import sys

# assuming your Django settings file is at '/home/myusername/mysite/mysite/settings.py'
path = '/home/proyectoepsl/Servidor'
if path not in sys.path:
    sys.path.append(path)

os.environ['DJANGO_SETTINGS_MODULE'] = 'Servidor.settings'

## Uncomment the lines below depending on your Django version
###### then, for Django >=1.5:
from django.core.wsgi import get_wsgi_application
application = get_wsgi_application()
###### or, for older Django <=1.4
#import django.core.handlers.wsgi
#application = django.core.handlers.wsgi.WSGIHandler()
```

Figura 76.Vista de modificaciones en el archivo WSGI.

En el apartado “Static Files” de esta misma pestaña, se deben definir las rutas estáticas que necesita el proyecto, para cargar correctamente las plantillas web que utiliza (ver Figura 72.Acceso a la consola del alojamiento web.).

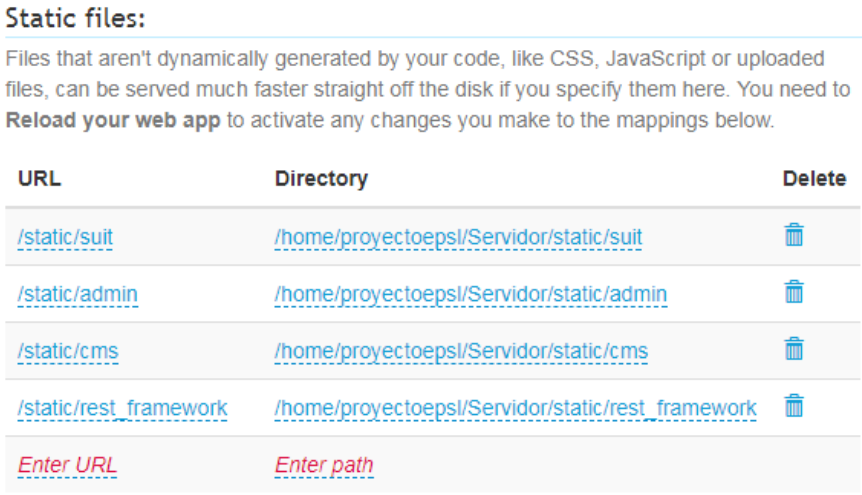

Figura 77.Vista de definición de las rutas estáticas.

Para poder instalar Django y todas las plantillas que se han utilizado en el desarrollo del proyecto es necesario crear una maquina virtual dentro del proyecto. Para crear una maquina virtual se debe escribir el comando mkvirtualenv --python=/usr/bin/python3.6 proyectoepsl-virtualenv en la consola de administración del alojamiento web (ver Figura 78.Vista de instalación de maquina virtual.).

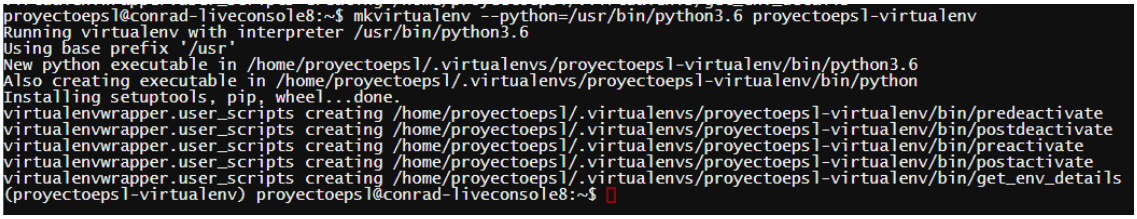

Figura 78.Vista de instalación de maquina virtual.

Para terminar de configurar la máquina virtual en la pestaña “Web” en el apartado “Virtualenv” se debe indicar la ruta del directorio de la maquina virtual (ver Figura 79.Vista de la definición del directorio de la maquina virtual.)

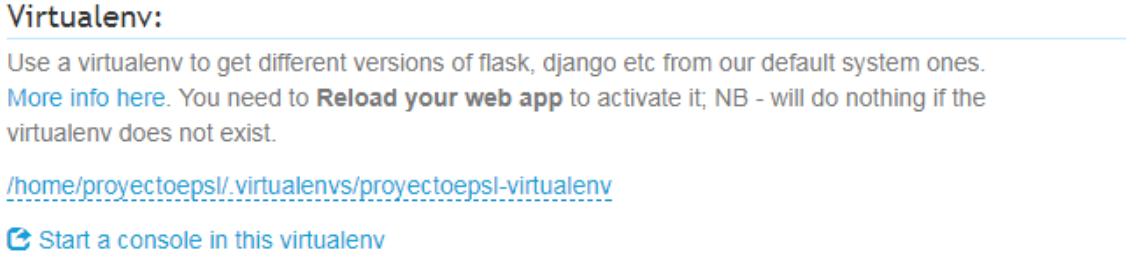

Figura 79.Vista de la definición del directorio de la maquina virtual.

Para instalar las herramientas que se necesitan para la implementación del proyecto, es necesario ejecutar una consola en la maquina virtual, esto se puede hacer pinchando en el enlace denominado “Star a console in this virtualenv” que aparece debajo de la definición de la ruta en la Figura 79.Vista de la definición del directorio de la maquina virtual.).

Las herramientas que hay que instalar en la consola de la maquina virtual y sus comandos de instalación son los siguientes:

- Instalación de Django: el comando que se utiliza es pip install django y su instalación se puede ver en la Figura 80.Vista de la instalación de Django.)

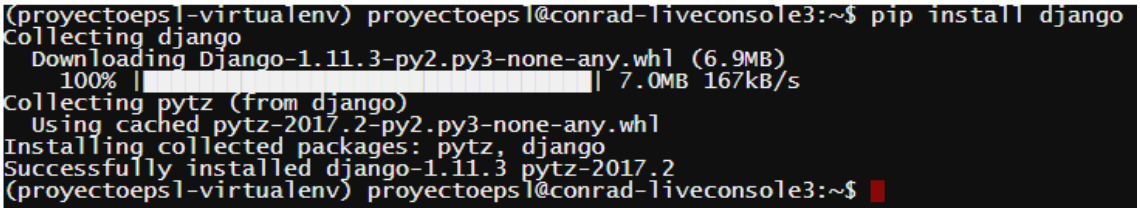

Figura 80.Vista de la instalación de Django.

- Instalación de Django-Suit: el comando que instala este paquete es pip install django-suit ==0.2.25.
- Instalación de la librería Pycrytodome22: esta es la librería que se usa para el cifrado AES y el comando que se usa para su instalación es pip install pip install pycryptodome.
- Instalación del paquete Django-restframework: para instalar este paquete se debe ejecutar el comando pip install djangorestframework.

Por último para dar permisos para que se pueda ejecutar el proyecto en el alojamiento web, es necesario que en el archivo de configuración settings.py, de la aplicación servidora, en la variable ALLOWED_HOST se indique el nombre del dominio de la aplicación, tal y como se muestra en la Figura 81.Vista de permiso de ejecución en el dominio.)

```
ALLOWED_HOSTS = ['proyectoepsl.pythonanywhere.com']
```

Figura 81.Vista de permiso de ejecución en el dominio.

6.5 Códigos Fuentes.

Junto a esta memoria se incluye la carpeta denominada Codigos_Fuentes,en ella se incluyen los códigos fuentes de las aplicaciones para el terminal Android y para la aplicacion servidora.

En esta carpeta se incluye tambien incluye el achivo db.Proyecto correspondiente a la base de datos utilizada en el proyecto.

[22] Documentación: http://pycryptodome.readthedocs.io/en/latest/src/introduction.html.

7 PLIEGO DE CONDICIONES

7.1 Características hardware y software

7.1.1 Características teléfono móvil Inteligente

Para poder hacer uso de la aplicación Android es necesario que el teléfono móvil inteligente tenga las siguientes características:

- Sistema operativo Android en su versión 4.0.3 o superior
- 4 MB de espacio libre
- Acceso a internet
- Chip NFC integrado

7.1.2 Características técnicas de la tarjeta NFC

Las tarjetas NFC a utilizar tienen que ser compatibles con los chips que normalmente tienen instalados los teléfonos inteligentes Android, por lo que tienen que tener las siguientes características:

- Tipo de tarjeta con especificaciones de **NFC Forum** Type2.
- Basada en la ISO/IEC 14443-A.
- Tarjetas NFC Anti metal, para que no tenga problemas a la hora de leerlas el teléfono móvil.
- Frecuencia de funcionamiento a 13,56 MHz
- Pasiva, es decir, que no precise de alimentación.
- Se recomienda una capacidad de 164 bytes como mínimo. Aunque según la normativa la memoria disponible mínima es 48 bytes y la máxima 2 Kbyte.

7.1.3 Características aplicación servidora

El alojamiento en donde se publique la Aplicación servidora puede tener un sistema operativo Windows o Linux con las siguientes características:

- Soporte al lenguaje de programación Python con versión 3.6.
- Soporte al administrador de paquetes pip.
- Soporte para el entorno de programación Django en su versión 1.11.2.
- Soporte para las librerías:
    - Django REST framework.
    - PyCryptodome.
    - Django contrib.
    - Django Suit 0.2.25.

Servidor web compatible con aplicaciones web de Python, recomendable Apache2 configurado para que la comunicación sea por `HTTPS`.

8 PRESUPUESTO

A continuación, se presenta un desglose de las tareas necesarias para la realización de este proyecto y su coste asociado:

8.1 Diseño del proyecto

Descripción	Cantidad	Precio Unitario	Precio Total
Diseño de funcionalidades	10h	17,00€	170,00€
Diseño de la arquitectura del sistema	5h	17,00€	85,00€
Diseño de la Aplicación Servidora	10h	17,00€	170,00€
Diseño de la Aplicación Android	5h	17,00€	85,00€
		Total	510,00€

Tabla 12.Presupuesto de Diseño

8.2 Desarrollo del proyecto

Descripción	Cantidad	Precio Unitario	Precio Total
Desarrollo de la Aplicación Servidora	100h	17,00€	1700,00€
Desarrollo de la Aplicación Android	150h	17,00€	2550,00€
		Total	4250,00€

Tabla 13.Presupuesto Desarrollo

8.3 Pruebas del proyecto

Descripción	Cantidad	Precio Unitario	Precio Total
Pruebas de la Aplicación Servidora	10h	17,00€	170,00€
Pruebas de la Aplicación Android	10h	17,00€	170,00€
		Total	340,00€

Tabla 14.Presupuesto Despliegue

8.4 Resumen

Diseño del Proyecto	510,00€
Desarrollo del Proyecto	4250,00€
Despliegue del Proyecto	340,00€
SUBTOTALES	**5100,00€**
IVA 21%	**1071,00€**
TOTAL	**6171,00€**

El coste total del proyecto asciende a **seis mil ciento setenta y un euros (6171,00€)**

9 REFERENCIAS BIBLIOGRÁFICAS

[1] RUBEN ANDRES. 08 Febrero 2017. Qué es NFC Móvil, cómo funciona y qué puedes hacer con él. ComputerHoy [online].[Consultado 01 Mayo 2017] http://computerhoy.com/noticias/life/que-es-nfc-movil-para-que-sirve-como-funciona-24207

[2] JUAN SEGUÍ MORENO. 28 Febrero 2012 .Aplicaciones prácticas de nfc. Área de Innovación y Desarrollo, S.L [pdf online].[Consultado 26 Junio 2017] https://www.3ciencias.com/wp-content/uploads/2013/01/NFC.pdf

[3] ¿Qué es Android y para qué sirve? WebGenio [online]. 16 Abril 2014. [Consultado 13 Mayo 2017] http://webgenio.com/2012/04/24/que-es-android-y-que-es-un-telefono-movil-android/

[4] Android. Wikipedia [online].[Consultado 1 Agosto 2017]. https://es.wikipedia.org/wiki/Android

[5] DesarrolloWeb.com. Qué es Python. DesarrolloWeb.com [online]. [Consultado 12 Julio 2017] https://desarrolloweb.com/articulos/1325.php

[6] 5.2. El patrón de diseño MTV (El libro de Django 1.0). LibrosWeb.es [online]. [Consultado 27 Julio 2017] http://librosweb.es/libro/django_1_0/capitulo_5/el_patron_de_diseno_mtv.html

[7] TECNOLOGÍA, ABC. ¿Qué es una API y para qué sirve? ABC [online]. 16 Febrero 2015. [Consultado 13 Agosto 2017]

[8] LUIS ALEXIS. Tecnología NFC.Monografias.com [online]. [Consultado 17 Agosto 2017] http://www.monografias.com/trabajos101/tecnologia-nfc/tecnologianfc.shtml#ixzz4mRbxg25G

[9] Definición de Dispatcher .Máster Magazine [online]. [Consultado 22 Julio 2017] https://www.mastermagazine.info/termino/4684.php

[10] ROBERT R. SABELLA. The NFC Data Exchange Format (NDEF). dummies [online]. [Consultado 28 Julio 2017] http://www.dummies.com/consumer-electronics/nfc-data-exchange-format-ndef/

[11] The asics of NDEF. Flomio [online]. 13 de Mayo 2012. [Consultado 15 Agosto 2017] https://flomio.com/2012/05/ndef-basics/

[12] Apps .Que significa [online] [Consultado 14 Agosto 2017] http://www.quesignificala.com/2015/10/apps-aplicacion.html

[13] Capítulo 5. Interactuar con una base de datos: Modelos (El libro de Django 1.0). LibrosWeb.es [online]. [Consultado 28 de Julio 2017]
http://librosweb.es/libro/django_1_0/capitulo_5.html

[14] Android tutorial preferencias compartidas. w3ii.com [online]. [Consultado 8 Agosto 2017]
http://www.w3ii.com/es/android/android_shared_preferences.html