Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

INURSE – APLICACIÓN ANDROID PARA LA MEJORA DE COMUNICACIÓN ENTRE PACIENTES HOSPITALIZADOS Y ENFERMEROS

Trabajo Fin de Máster

Máster Universitario en Ingeniería Informática

Autor: Alejandro Lledó Viciano

Tutor: Vicente Pelechano Ferragud

2016-2017

Resumen

Este Trabajo Fin de Master pretende ofrecer una solución informática para el sector de la sanidad. Se trata de una aplicación para dispositivos móviles destinada a ser usada dentro de los hospitales, y por ello la solución propuesta ofrecerá una aplicación con dos *front-end*, uno para el personal sanitario y otro para los pacientes hospitalizados. Cada *front-end* ofrecerá diferentes servicios, todos ellos orientados a facilitar la actividad clínica del personal sanitario, incrementar la eficiencia del proceso asistencial y mejorar la atención de los pacientes.

La aplicación se desarrollará para el sistema operativo Android, el entorno de programación utilizado será Android Studio y el lenguaje de programación empleado será Java. La parte servidora atenderá las peticiones realizadas por la aplicación mediante el uso de scripts escritos en PHP, los cuales realizarán consultas a una base de datos MySQL. La información será transportada entre cliente y servidor codificada en el formato de intercambio de información JSON. Además, se realizará una integración con el servicio Firebase Cloud Messaging para la recepción de notificaciones instantáneas que informen a los usuarios de determinados eventos en tiempo real.

Palabras clave: iNurse, Android, Java, Hospital, Sanidad, Atención Hospitalaria, Aplicación Móvil, PHP, JSON, MySQL.

Abstract

The aim of this final Master work is to offer a computer solution for the Health System. It's about a mobile application to be used inside the hospitals and for that reason this suggested solution will offer one application with two front-end; one of them destined to health workers and the other one for the inpatient. Each front-end will offer different services all of them directed to make the clinic activity easier for the health workers for increasing the efficiency during the ancillary tasks and also to improve the inpatient assistance.

The application will be developed to Android operating system. The programming surroundings used will be Android Studio and the programming language used will be Java. The servant part will attend petitions made by the application through the use of scripts written in PHP which will make requests to a database MySQL. The information will be carried between costumer and server encrypted by the information exchange format JSON. In addition, it will be made an integration with the Firebase Cloud Messaging service to the immediate notification reception which will inform to the users about specific events in real time.

Keywords: iNurse, Android, Java, Hospital, Health, Hospital Care, Mobile Application, PHP, JSON, MySQL.

Tabla de contenidos

Tabla de ilustraciones

1. Introducción

1.1. Motivación

Una de las principales motivaciones que han llevado a escoger un TFM de esta temática surge como consecuencia de mi actual trabajo. Como desarrollador en el equipo de Orion Clinic, sistema de información clínico-asistencial para centros hospitalarios, cualquier aportación al mundo de la sanidad que derive en una mejora de la atención al ciudadano y que facilite el trabajo del personal sanitario se convierte en una gran motivación.

Por una parte, el principal motivo que me han llevado a desarrollar una aplicación para smartphones es que en los últimos años los avances tecnológicos han permitido que prácticamente en cualquier dispositivo, ya sea de gama baja o alta, pueda ejecutarse una aplicación que consuma una cantidad moderada de recursos. Además, hoy en día prácticamente cualquier persona adulta dispone de un smartphone. Por ello, la aplicación propuesta en el presente proyecto está pensada para que cualquier paciente hospitalizado pueda utilizarla desde su smartphone en cualquier momento, aprovechando así la movilidad que estos dispositivos ofrecen.

Por otra parte, se ha escogido el sistema operativo Android principalmente por dos razones. La primera es que se trata del sistema operativo con mayor *market share* o participación en el mercado y por lo tanto la aplicación será accesible a un público notablemente mayor que si se implementara para otro sistema operativo. En la ilustración 1 podemos observar la comparativa entre las participaciones del mercado de los diferentes sistemas operativos, donde claramente Android es superior a los demás sistemas operativos.

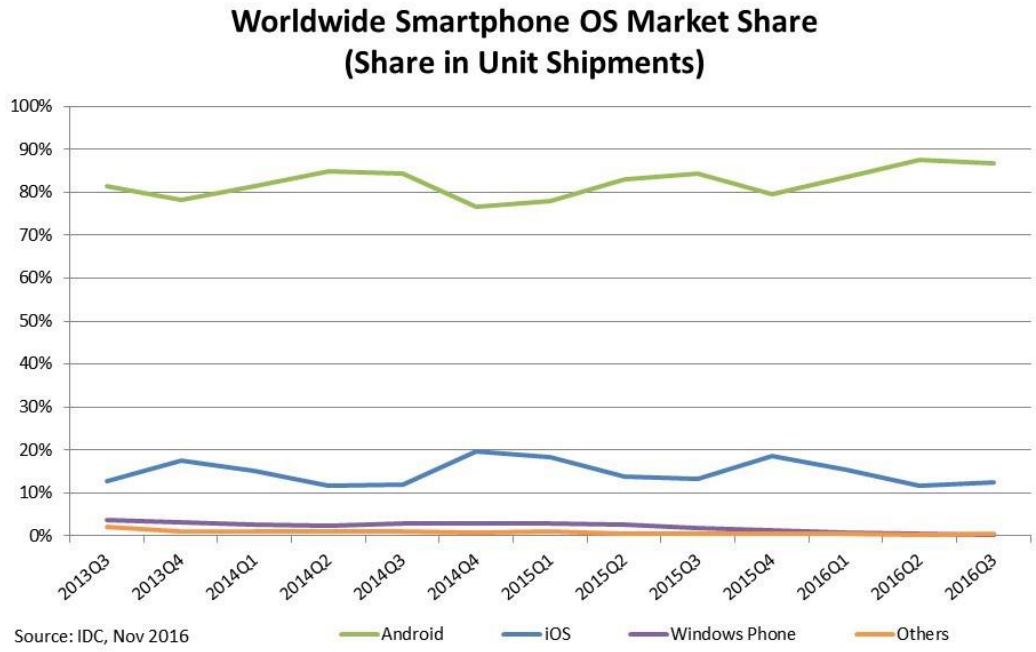

Ilustración 1. Participación de mercado de los sistemas operativos

La segunda razón es que recibe actualizaciones constantemente por lo que a largo plazo la utilización de este sistema operativo es una buena elección ya que, no se convertirá en una tecnología obsoleta y contará con las ultimas funcionalidades disponibles, que en algún momento podrían ser aprovechadas por iNurse.

Por último, la realización de este proyecto me va a permitir conocer en mayor profundidad dos lenguajes de programación y un entorno de desarrollo con los que no tengo mucha experiencia, estos son Android, PHP y Android Studio, respectivamente. Actualmente, en el mundo de la informática es de gran importancia tener un conocimiento muy variado y tener dichos conocimientos actualizados puesto que, en un sector en continuo cambio y crecimiento, centrarse en un solo lenguaje o en un único entorno de programación no es nada recomendable.

1.2. Objetivos

El objetivo principal del proyecto es desarrollar una aplicación Android orientada a ayudar en la actividad clínica del personal sanitario, incrementar la eficiencia del proceso asistencial y mejorar la atención de los pacientes.

Para ello, la solución propuesta ofrecerá una aplicación con dos *front-end*, uno para los enfermeros y otro para los pacientes hospitalizados. El *front-end* al que tendrán acceso los pacientes hospitalizados tendrá las siguientes funcionalidades:

- Solicitar un servicio al personal sanitario.
- Visualizar los servicios solicitados y el estado en el que se encuentran.
- Mostrar los diferentes menús del día y poder escoger uno de ellos.
- Visualización y recepción de avisos establecidos por los enfermeros.
- Visualizar información de interés referente al hospital.
- Recepción de notificaciones.

Por otra parte, el *front-end* de los enfermeros ofrecerá las siguientes funcionalidades:

- Visualizar las solicitudes de servicios realizadas por los pacientes, ordenadas por prioridad.
- Cambiar el estado de una solicitud.
- Mapa de camas que muestre información básica acerca de los pacientes.
- Mostrar información detallada sobre los pacientes.
- Visualizar los menús escogidos por los pacientes.
- Programación de avisos para los pacientes.
- Mostrar los avisos activos programados para los pacientes.
- Recepción de notificaciones.

2. Estado del arte

En este segundo capítulo se hará un estudio de la situación actual del mercado realizando una búsqueda para encontrar aplicaciones cuyo ámbito de aplicación sea el sanitario y que realicen funciones similares a iNurse. Se analizarán dichas aplicaciones y las funcionalidades que ofrecen frente a iNurse con la finalidad de determinar si lo que ofrece nuestra aplicación es lo suficientemente novedoso como para tener éxito en el mercado.

2.1. Situación actual de la tecnología

2.1.1. Orion Clinic

Orion Clinic es un proyecto de la Agencia Valenciana de Salud (AVS) basado en un sistema de información clínico-asistencial para los centros hospitalarios.

La aplicación está orientada a ayudar a la actividad clínica de los profesionales, a incrementar la eficiencia del proceso asistencial en su conjunto y a facilitar la continuidad asistencial de los cuidados, todo ello con el fin último de mejorar la atención al paciente. Entre sus funcionalidades, destaca la visualización de la historia clínica del paciente, las opciones de prescripción, informes, lista de espera quirúrgica o gestor de solicitudes (Everis, 2014).

Además, destaca la mayor disponibilidad de la información, el mayor control y seguridad para el paciente o la integración de las nuevas tecnologías de la comunicación en el sistema de seguimiento por parte del profesional.

El ámbito de aplicación de Orion Clinic es la atención especializada y cubre todas sus áreas, como por ejemplo admisión, consultas externas, hospitalización, urgencias, farmacia y bloque quirúrgico. En la ilustración 2 podemos ver la pantalla principal de Orion Clinic, desde la cual el profesional puede acceder a las áreas que tenga configuradas.

Orion Clinic está implantado en la mayoría de los hospitales de la Comunidad Valencia con mayor volumen de pacientes. Algunos de estos hospitales son: el Hospital Universitari i Politècnic La Fe, Hospital Francesc de Borja, Hospital Doctor Peset y el Hospital General Universitario de Alicante. Además, actualmente se va a poner en marcha un proceso de implantación con el que Orion Clinic se pondrá en marcha en más hospitales.

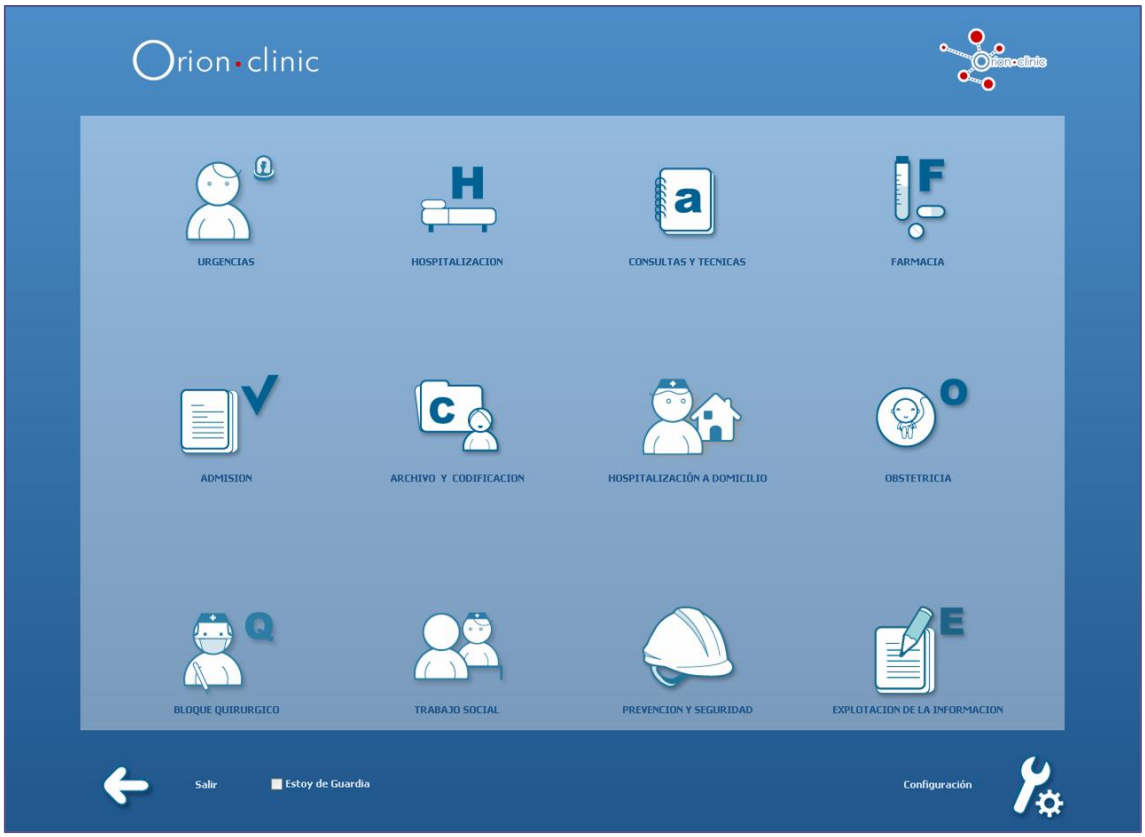

Ilustración 2. Pantalla principal de Orion Clinic

2.1.2. IMED PACIENTES

Se trata de una aplicación para los pacientes pertenecientes al grupo privado hospitalario IMED Hospitales (ver ilustración 3). Los usuarios finales de esta aplicación son los pacientes de dichos hospitales y su propósito es mejorar la atención de sus pacientes con una aplicación para smartphones que puedan usar en su día a día (IMED Hospitales, 2017). Está disponible para los sistemas operativos iOS y Android.

El conjunto de servicios que ofrece son las siguientes:

- Solicitar citas pudiendo acceder a la agenda de especialidades y consultar la disponibilidad de horarios en tiempo real.
- Consultar las citas concertadas, pudiendo anularlas, y sincronizarlas en el calendario del smartphone.
- Solicitar que un operador telefónico de IMED Hospitales se ponga en contacto.
- Pedir una ambulancia mediante el uso de los servicios de geoposicionamiento del smartphone de forma que la ambulancia acudirá a nuestra ubicación sin necesidad de introducir la dirección manualmente.
- Control y gestión de la medicación con la posibilidad de establecer el plan de tomas y añadirlo a la agenda para que recuerde las tomas.
- Recepción de notificaciones cuando sea el momento de tomar una medicación la cual haya sido recetada por los profesionales de IMED Hospitales.
- Activar/Desactivar el recordatorio de citas por email.

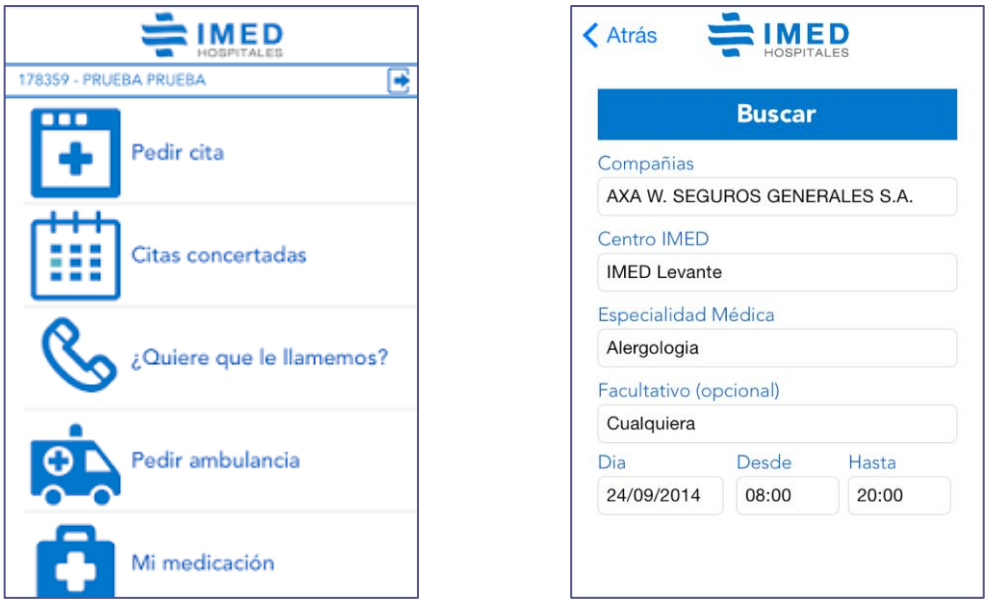

Ilustración 3. Interfaces de la aplicación IMED

2.1.3. Hospital Universitario Rey Juan Carlos

Esta aplicación está destinada exclusivamente a los pacientes del Hospital Universitario Rey Juan Carlos. Se ha diseñado con el objetivo de facilitar a los pacientes el acceso a su historia clínica y a la gestión de la información médica facilitando el acceso al portal del paciente (ver ilustración 4) desde un dispositivo móvil. El portal del paciente es un espacio personal desde el que se puede acceder al historial médico, pruebas diagnósticas, informes clínicos, gestión de citas médicas y tener un seguimiento (IDCSALUD, S.L., 2017).

Esta aplicación está disponible tanto para Android como para iOS y su finalidad es ofrecer un conjunto de servicios para estrechar la relación entre el paciente y el médico para ofrecerle un servicio de mayor calidad, sin esperas ni desplazamientos innecesarios. Las principales funciones que ofrece esta aplicación son:

- Consultar las citas concertadas, pudiendo cancelarlas e incluso modificar la fecha y hora de la cita.
- Consultar y realizar las posibles tareas pendientes que el médico haya programado.
- Descargar y consultar los resultados de sus pruebas sin necesidad de desplazarse al centro médico para recogerlos.
- Consultar informes clínicos e históricos de formularios realizados pudiendo filtrar los resultados por fecha o especialidad entre otros.

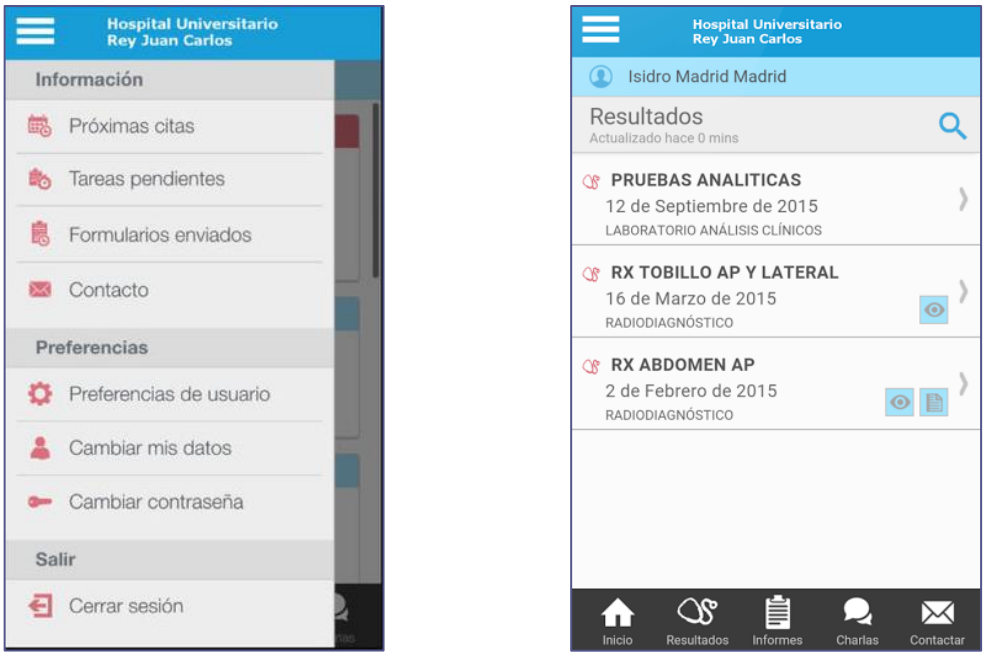

Ilustración 4. Interfaces de la aplicación Hospital Universitario Rey Juan Carlos

2.1.4. ehCOS My Hospital

Se trata de una herramienta que proporciona un conjunto de servicios que proveen al profesional médico la capacidad de realizar una monitorización continua de sus pacientes, acceso inmediato a la información clínica y administrativa de un paciente para una mejor toma de decisiones y mejorar la eficiencia del hospital a través de la explotación de la información (Everis, 2017). En la ilustración 5, podemos ver un ejemplo de las interfaces de ehCOS My Hospital.

Entre las funcionalidades que ofrece, podemos destacar:

- Seguimiento y monitorización de los pacientes.
- Acceso al historial clínico de los pacientes en tiempo real.
- Visualizar la lista de pacientes hospitalizados.
- Consultar el detalle de los episodios de los pacientes, así como las atenciones previas a un episodio.
- Obtener estadísticas del centro hospitalario como indicadores de admisión, indicadores de ocupación de camas o indicadores de tiempo de espera.
- Consultar el resumen del estado de las urgencias y emergencias hospitalarias.

Ilustración 5. Interfaces de la aplicación ehCOS My Hospital

2.1.5. Mi Sanitas

Se trata de una aplicación móvil donde los clientes de Sanitas pueden realizar un conjunto acciones para gestionar su salud en cualquier momento y desde cualquier lugar (ver ilustración 6). Dispone de una interfaz muy intuitiva, sencilla e innovadora y está desarrollada con el objetivo optimizar la velocidad y el tiempo de carga para ofrecerle al usuario una buena experiencia de uso (Sanitas, 2017). Se encuentra disponible para las plataformas Android e iOS.

Los servicios que tiene disponibles Mi Sanitas son los siguientes:

- Solicitar citas médicas, gestionarlas, reprogramarlas y cancelarlas.
- Buscar médicos, hospitales, centros médicos o clínicas dentales.
- Encontrar el centro de urgencia y especialista más cercano.
- Consultar el histórico de visitas al médico y las citas pendientes.
- Guardar médicos favoritos.
- Consultar informes médicos, analíticas, radiografías, etc.
- Subir informes de otros centros sanitarios para tener toda la información de salud en un mismo lugar.
- Acceder a la Tarjeta de salud digital.
- Hacer gestiones de la póliza, como solicitar reembolsos y autorizaciones.
- Consultar los recibos y copagos.
- Consultar, modificar y personalizar el perfil de cada beneficiario.
- Cambiar la cuenta bancaria y periodicidad.
- Acceder a la biblioteca de consejos de salud.

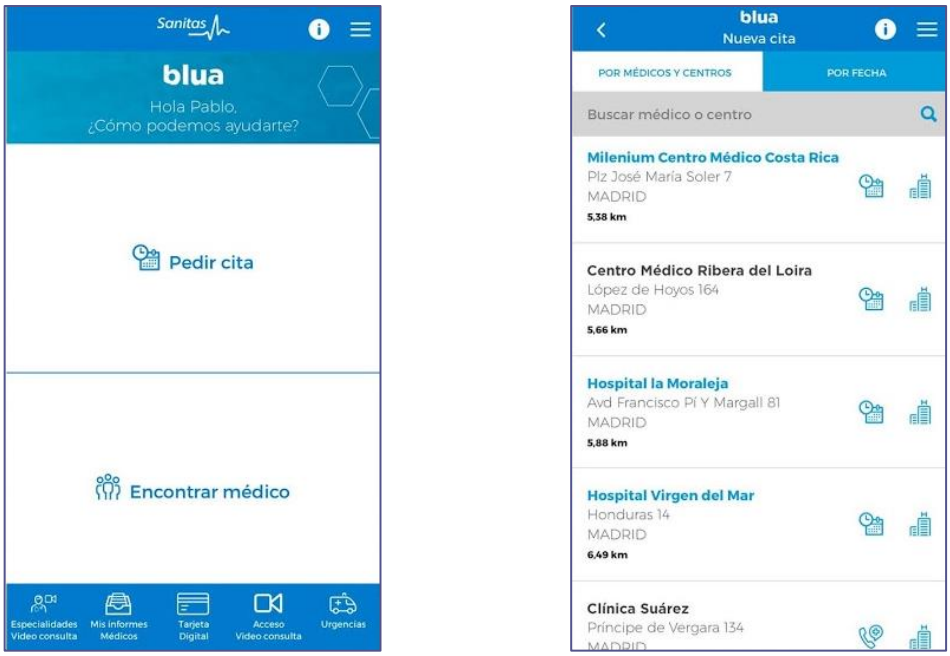

Ilustración 6. Interfaces de la aplicación Mi Sanitas

2.2. Crítica al estado del arte

Las aplicaciones analizadas en el apartado anterior son todas muy completas, ofrecen un conjunto de funcionalidades que, a su modo, facilitan la actividad clínica de los profesionales y mejoran la atención del paciente, adaptándose perfectamente a las necesidades del ámbito de uso en el que son utilizadas. Sin embargo, cada aplicación está enfocada a un rol de usuario en concreto, o bien al profesional sanitario, o bien al paciente, pero nunca a los roles.

Con iNurse se pretende introducir un conjunto de novedades respecto a las aplicaciones presentes en el mercado. La primera novedad es que se trata de una aplicación destinada a ser usada tanto por el profesional sanitario como por los pacientes, lo que nos lleva a la segunda novedad introducida la cual es su ámbito de uso. iNurse está destinada a ser utilizada durante la estancia de un paciente en el hospital, a diferencia de las demás aplicaciones que están pensadas para ser usadas en el día a día de los pacientes.

Por último, la principal característica de iNurse con la que se pretende proporcionarle un valor añadido que le permita destacar entre las aplicaciones existentes es la interacción directa y continua que ofrece entre el personal sanitario y los pacientes durante su estancia en el hospital.

Con todas las novedades que introduce iNurse, se pretende tapar el hueco funcional al cual no dan soporte las aplicaciones actuales con la finalidad de aportar nuevas herramientas que mejoren la atención de las personas.

3. Análisis del problema

En el tercer capítulo se va a realizar un análisis de los requerimientos necesarios para llevar a cabo nuestro proyecto. En primer lugar, se obtendrán los requisitos que debe cumplir iNurse para satisfacer sus objetivos y, en segundo lugar, se estudiarán las necesidades de privacidad y seguridad que debe cumplir una aplicación de estas características.

3.1. Análisis de requisitos

Uno de los primeros pasos del desarrollo del proyecto fue obtener los casos de uso de la aplicación. Un caso de uso describe una secuencia de interacciones que se llevarán a cabo entre el sistema y sus actores en respuesta a un evento que inicia el actor principal sobre el sistema. Un actor se trata de una entidad externa al sistema el cual le solicita una funcionalidad. En caso de que se trate personas, se les puede ver como definiciones de rol. En nuestro caso tenemos dos roles diferentes: paciente y enfermero.

También se ha creado un diagrama de casos de uso cuyo objetivo es representar el comportamiento del sistema y las interacciones del usuario con el sistema. La ilustración 7 nos muestra el diagrama de casos de uso de iNurse.

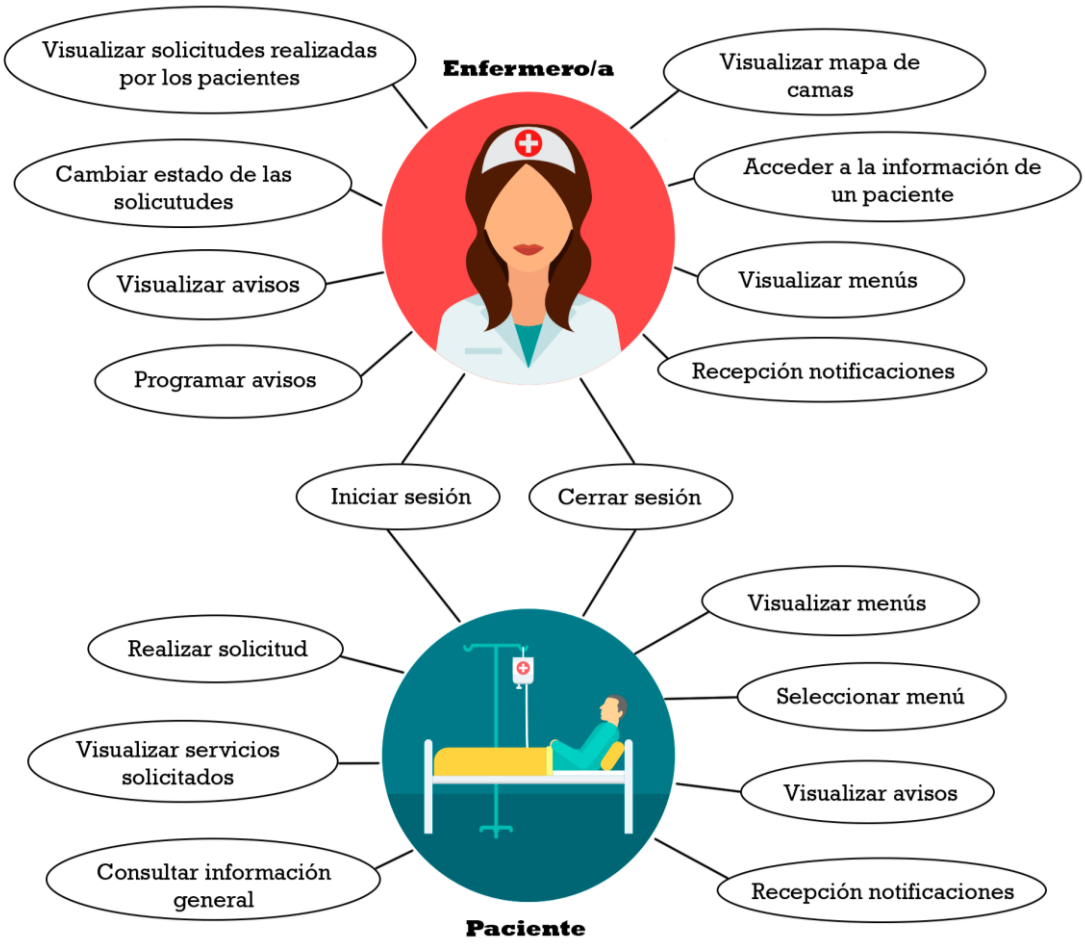

Ilustración 7. Diagrama de casos de uso de iNurse

A continuación, se detalla la funcionalidad de los casos de usos expuestos en el diagrama. Se dividirán en tres bloques según el rol del actor que realiza el evento.

Enfermero y paciente

Iniciar sesión: se debe poder acceder a la aplicación autenticándose mediante las credenciales proporcionadas por el hospital. Las credenciales de los pacientes se generarán en el momento que sean hospitalizados, en cambio, el enfermero será dado de alta por el responsable del sistema.

Los datos de sesión se deben almacenar de forma que el usuario no tenga que introducirlas la próxima vez que acceda a la aplicación.

Cerrar sesión: el usuario debe poder salir de la aplicación borrando los datos de sesión. La próxima vez que se acceda a la aplicación se deberán introducir las credenciales de acceso.

Paciente

Realizar solicitud: el paciente puede realizar solicitudes al hospital, previamente definidas por el hospital. Además, cada solicitud tiene una prioridad asignada, la cual no puede ser editada.

Visualizar servicios solicitados: visualizar todas las solicitudes que ha realizado el paciente y conocer en qué estado se encuentran.

Consultar información general: el paciente puede consultar información ofrecida por el hospital.

Visualizar menús: visualizar los menús ofrecidos por el hospital en un determinado momento del día.

Seleccionar menú: el paciente puede seleccionar un menú de los que se encuentren disponibles en un determinado momento del día.

Visualizar avisos: el paciente puede visualizar todos los avisos que los enfermeros le han programado.

Recepción notificaciones: el paciente tiene la capacidad de recibir notificaciones que le proporcionen información sobre los avisos que le han sido programados y del cambio de estado de sus solicitudes realizadas.

Enfermero

Visualizar solicitudes realizadas por los pacientes: los enfermeros pueden ver todas las solicitudes realizadas por los pacientes, las cuales aparecen ordenas por prioridad de mayor a menor importancia. Las solicitudes pueden tener uno de los siguientes estados: pendiente, en progreso, rechazada y finalizada.

Cambiar estado de las solicitudes: según el estado en el que se encuentre una solicitud, los enfermeros podrán realizar una determinada acción sobre ella. Las acciones disponibles según el estado de la solicitud son:

- Si una solicitud se encuentra en estado pendiente podemos realizar dos acciones: empezar progreso o rechazar solicitud. Cuando una solicitud es rechazada, el enfermero debe especificar el motivo del rechazo.
- Si la solicitud está en progreso, esta se puede finalizar o rechazar.
- En el caso de que la solicitud tenga el estado de rechazada, no se puede realizar ninguna acción.
- Sobre las solicitudes que se encuentren finalizadas tampoco se puede realizar ninguna acción.

Visualizar menús: un enfermero puede ver que menús han sido escogidos por los pacientes, pudiendo filtrar los resultados por planta. Dependiendo de la hora del día, se recuperará la información correspondiente al servicio de comida que deban cubrir los enfermeros.

Visualizar mapa de camas: el enfermero puede visualizar en una lista la información básica sobre los pacientes hospitalizados, teniendo la posibilidad de filtrar los resultados por planta.

Acceder a la información de un paciente: si se pulsa sobre un ítem del mapa de camas, el enfermero puede consultar la información que contiene la ficha del paciente.

Ver avisos: el enfermero puede ver los avisos que se encuentran actualmente activos. Los avisos activos son aquellos cuya fecha de finalización es posterior al momento actual.

Programar avisos: un enfermero puede crear avisos para un determinado paciente.

Recepción notificaciones: los enfermeros recibirán una notificación cada vez que un paciente realice una solicitud.

3.2. Requisitos no funcionales

Los requisitos no funcionales son aquellos que establecen una serie de criterios para evaluar el comportamiento de un sistema, en contraste con los requisitos funcionales que especifican comportamientos más concretos. En nuestro caso, los requisitos funcionales se corresponden con los casos de uso expuestos en el apartado anterior.

Los requisitos no funcionales que debe cumplir iNurse son los siguientes:

- **Usabilidad**: la aplicación debe ser intuitiva y fácil de usar. Cualquier usuario debe ser capaz de aprender a usar la aplicación de forma rápida.
- **Escalabilidad**: la aplicación debe ser capaz de reaccionar y adaptarse a un posible crecimiento de la aplicación debido al incremento de usuarios activos sin perder la calidad de los servicios ofrecidos.
- **Eficiencia**: la aplicación debe ser capaz de realizar sus funciones sin cometer errores.
- **Mantenibilidad**: el esfuerzo requerido para conservar el funcionamiento correcto de la aplicación o para ser reparada ante fallos producidos debe ser bajo.

- **Disponibilidad**: la aplicación tiene que estar disponible para ser descargada y en funcionamiento durante al menos el 95% del tiempo.
- **Estabilidad**: la aplicación debe ser capaz de responder y recuperarse ante posibles errores de ejecución o errores de red, avisando al usuario en los casos que corresponda.
- **Seguridad**: el sistema debe cumplir las normas, procedimientos y los métodos necesarios para asegurar que el sistema sea seguro y confiable.

3.3. Análisis de la protección de datos

El derecho a la protección de datos personales es un derecho fundamental de todas las personas que se traduce en la autoridad de control sobre el uso que se hace de sus datos personales. Este control permite evitar que, a través del tratamiento de nuestros datos, se pueda llegar a disponer de información sobre nosotros que afecte a nuestra intimidad y demás derechos fundamentales y libertades públicas (lopd-proteccion-datos.com, 2017).

Los datos sanitarios, junto con datos sobre ideología, religión, afiliación sindical, creencias, vida sexual, origen racial y comisión de delitos penales o administrativos, son considerados por la LOPD, Ley Orgánica de Protección de Datos, como “datos especialmente protegidos” por afectar especialmente a la intimidad, los derechos fundamentales y las libertades públicas de las personas y, por tanto, se exige una mayor protección que para el resto de datos personales (González, 2016).

Debido a la LOPD, los datos que puede contener iNurse sobre los pacientes debe ser tratados con precaución. Por una parte, se deben establecer políticas de seguridad para autentificar al usuario que accede a información sensible y auditar cuando se ha accedido a dicha información.

Por otra parte, se deben establecer canales de comunicación seguros para asegurar la confidencialidad de la información durante las comunicaciones realizadas por la aplicación. En el siguiente apartado, se realiza un análisis de como enviar información cifrada para proteger la información sensible y respetar la LOPD.

3.4. Análisis de seguridad

Hoy en día, la seguridad en Internet es un aspecto realmente importante que no se debe ignorar. En un proyecto de estas características donde viaja información sensible de personas, tal y como se ha comentado en el apartado anterior, se deben usar protocolos de seguridad con la finalidad de que los datos viajen de forma segura, totalmente encriptada y cifrada de forma que nadie pueda robar la información y darle un uso malintencionado.

Una solución para proteger nuestros datos es la utilización del protocolo SSL (*Secure Sockets Layer*). Se trata de un protocolo criptográfico que proporciona privacidad e integridad entre dos aplicaciones de comunicaciones utilizando HTTP.

Para conocer el funcionamiento de este protocolo, primero hay que conocer que es la criptografía simétrica y asimétrica. La criptografía simétrica utiliza una misma clave para cifrar y descifrar los mensajes. Esta clave debe ser secreta y solo la deben conocer el

emisor y el receptor. Una de las grandes ventajas de este mecanismo es que el cifrado y descifrado de mensajes es muy rápido, sin embargo, el gran problema reside en el intercambio de dicha clave. Remitente y destinatario deben encontrar un canal de comunicación seguro para transmitirse las claves ya que, para un atacante sería más fácil interceptar la clave que probar las posibles combinaciones del espacio de claves posibles (ver ilustración 8).

Los algoritmos de criptografía simétrica más modernos, como por ejemplo IDEA, generan claves de 128 bits lo cual corresponde a 2^{128} posibles claves. A pesar de que todas las computadoras que existen cooperasen para descifrarla, tardarían más tiempo en encontrar la clave que la edad del universo. Por esta razón, la principal vulnerabilidad de este método es el intercambio de la clave.

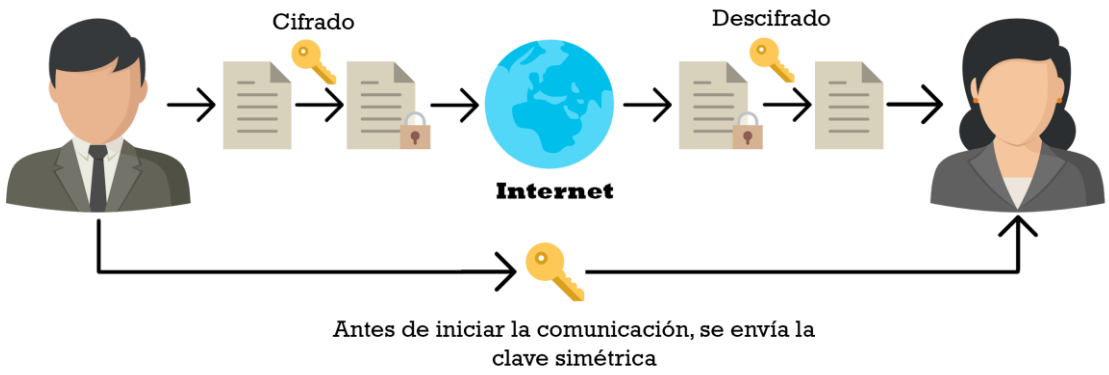

Ilustración 8. Cifrado simétrico

Por otra parte, está la criptografía asimétrica. Este sistema de cifrado utiliza dos claves diferentes, una pública que conoce todo el mundo y una privada que debe ser totalmente secreta. En este caso, el remitente cifra el mensaje utilizando la clave pública del destinatario y solamente el destinatario podrá descifrar el mensaje haciendo uso de la clave privada (ver ilustración 9). Gracias a esto, se logra la confidencialidad del mensaje ya que, únicamente el destinatario puede descifrarlo.

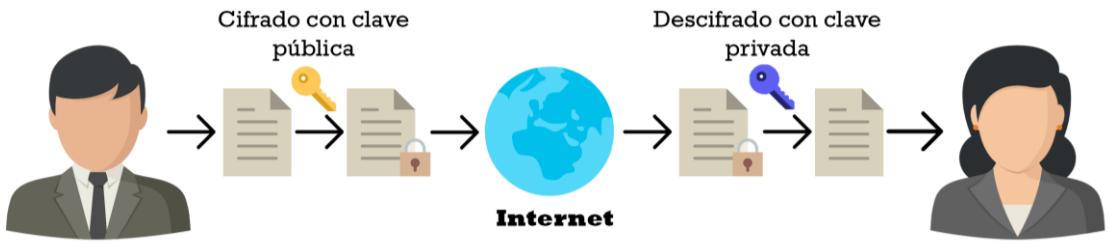

Ilustración 9. Confidencialidad con cifrado asimétrico

Además, si el propietario del par de claves hace uso de su clave privada para cifrar el mensaje, cualquiera puede descifrarlo utilizando su clave pública (ver ilustración 10). De este modo se consigue la autentificación del remitente ya que solo él pudo haber empleado la clave privada. En este fundamento se basa la firma electrónica.

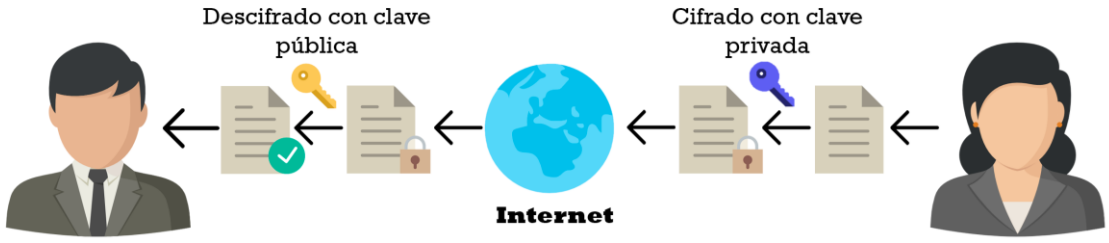

Ilustración 10. Autentificación mediante cifrado simétrico

La principal desventaja frente a la criptografía simétrica es que este mecanismo es poco eficiente y, por lo tanto, más lento. Además, una utilización repetitiva de las claves privadas puede ser peligroso debido a que hay ataques criptográficos que se basan en el análisis de paquetes encriptados y se podría llegar a obtener la clave privada. Por otra parte, la principal ventaja es que no necesita canales seguros para mandar la clave ya que esta se distribuye públicamente manteniéndose la privada para el uso exclusivo del propietario.

Pues bien, el protocolo SSL utiliza un mecanismo de criptografía hibrido que combina el mecanismo simétrico con el asimétrico aprovechando de este modo las ventajas que ofrecen ambas opciones.

Para establecer una canal de comunicación SSL ocurre lo siguiente:

1. El emisor envía al receptor una petición para iniciar una sesión de comunicación segura
2. El receptor le responde al emisor con un certificado que contiene su la clave pública.
3. El emisor debe autentificar el certificado con una lista de CA conocidas.
4. El emisor genera una clave simétrica aleatoria, la cifra utilizando la clave pública del emisor y se la envía.
5. El receptor descifra el mensaje con su clave privada y obtiene la clava simétrica.
6. Emisor y receptor ya conocen la clave simétrica por lo que pueden enviarse mensajes cifrados. La clave simétrica únicamente tiene validez durante esta sesión de comunicación.

La ilustración 11 muestra de forma resumida el proceso usado por el protocolo SSL.

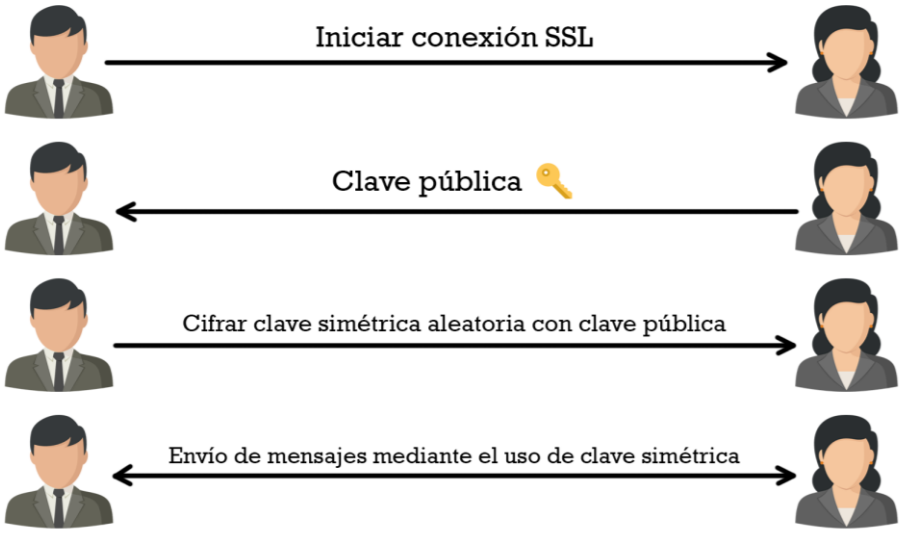

Ilustración 11. Proceso del protocolo SSL

En conclusión, si queremos conseguir un canal de comunicación que asegure la confidencialidad e integridad de los datos enviados debemos utilizar protocolos de seguridad y sin duda el protocolo SSL es una excelente elección.

4. Contexto tecnológico

En este cuarto capítulo, se va a llevar a cabo un estudio acerca de las tecnologías que van a ser necesarias para el desarrollo del proyecto. Se analizarán aspectos como el sistema operativo empleado, lenguajes de programación, plataformas de desarrollo y gestores de bases de datos.

4.1. Sistema operativo Android

Según (Gironés, 2016) se trata de un sistema operativo escrito en C, basado en el *kernel* de Linux y diseñado principalmente para dispositivos con pantalla táctil. Android es libre y de código abierto, lo que significa que cualquier desarrollador software puede crear aplicaciones y publicarlas.

Los principales componentes que componen su arquitectura (ver ilustración 12) son:

- **Aplicaciones**: esta capa contiene las aplicaciones instaladas en el dispositivo, las cuales utilizan los servicios, APIs y librerías de los niveles inferiores. Las aplicaciones están escritas bajo el lenguaje Java.
- **Framework de aplicaciones**: contiene el conjunto de APIs de desarrollo básicas para cualquier aplicación.
- **Librerías nativas**: este nivel incluye el conjunto de librerías C/C++ utilizadas por Android. Junto al *kernel* de Linux, estas librerías constituyen un pilar fundamental para las capacidades de Android.
- **Android Runtime**: set de bibliotecas que proporcionan la mayor parte de las funcionalidades disponibles en las bibliotecas base de Java. Incluye también la máquina virtual Dalvik sobre la cual se ejecutan las aplicaciones.
- **Kernel Linux**: actúa como capa de abstracción entre el hardware y el resto de la pila de software. Android depende del *kernel* para los servicios base del sistema tales como la seguridad, la gestión de la memoria o la gestión de procesos.

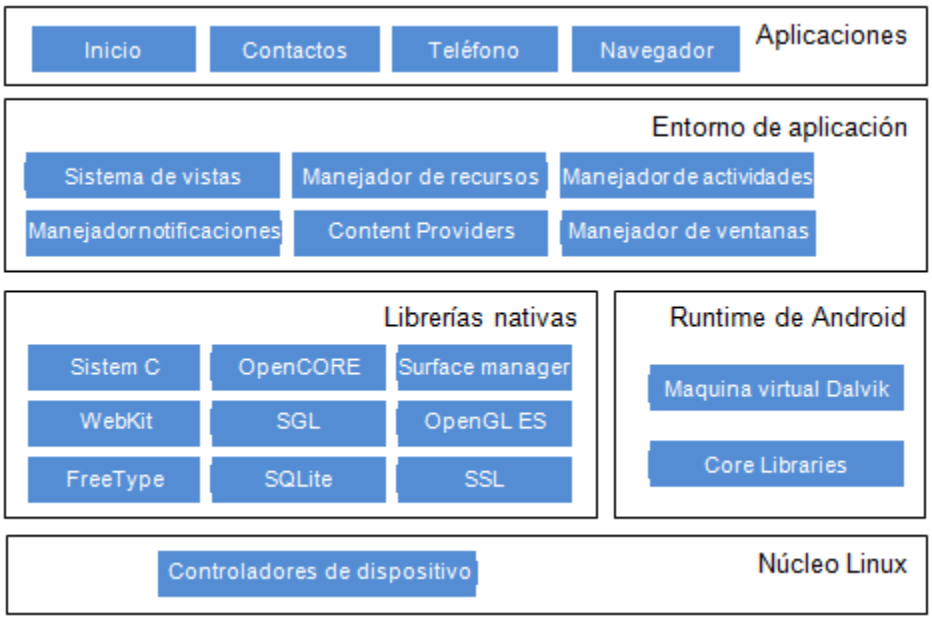

Ilustración 12. Arquitectura Android

Actualmente, la última versión estable de Android es la versión 7.x, también conocida como Nougat. iNurse ha sido desarrollada con el nivel 24 de API lo que quiere decir que puede aprovechar las funcionalidades más novedosas que ofrece la versión Android Nougat.

4.2. Lenguajes de programación

Java

Según (Schildt, 2007) es un lenguaje de programación, desarrollado por Sun Microsystems, de propósito general, concurrente y orientado a objetos diseñado específicamente para tener el número mínimo de dependencias con la implementación. Gracias a esto, los programadores pueden desarrollar una aplicación y una vez compilada, puede ser ejecutada sin necesidad de ser recompilada en cualquier dispositivo equipada con una Máquina Virtual Java (en inglés *Java Virtual Machine*, JVM) sin importar la arquitectura de la computadora subyacente, tanto hardware como software.

Algunas de las características más destacables de este lenguaje son:

- **Amplio conjunto de bibliotecas**: el gran número de librerías que pone Sun a la disposición del programador le ofrecen la posibilidad de realizar cualquier tipo de aplicación. Además, existen también una gran cantidad de librerías realizadas por terceros.
- **Distribuido**: dispone de una gran colección de clases que facilitan la creación de aplicaciones distribuidas.
- **Interpretado y compilado a la vez**. Java es compilado a bytecodes, semejantes a las instrucciones de ensamblador. Por otra parte, es interpretado, ya que los

bytecodes se pueden ejecutar directamente sobre cualquier máquina a la cual se hayan portado el intérprete y el sistema de ejecución en tiempo real (*run-time*).

- **Robusto**: proporciona numerosas comprobaciones sobre el código durante la compilación y en tiempo de ejecución. Además, sus características de memoria libran a los programadores de la aritmética de punteros, causante de muchos errores, y el recolector de basura exime al programador de la responsabilidad de liberar memoria de forma explícita.
- **Indiferente a la arquitectura.**
- **Portable**. La indiferencia a la arquitectura representa sólo una parte de su portabilidad. Java especifica los tamaños de sus tipos de datos básicos y el comportamiento de sus operadores aritméticos, de manera que los programas son iguales en todas las plataformas.
- **Alto rendimiento.**
- **Multithreading**. Soporta sincronización de múltiples hilos de ejecución a nivel de lenguaje. De esta forma, mientras un hilo se encarga de recuperar información de la base de datos, otro puede encargarse de establecer comunicaciones y otro presentarle una animación al usuario.

Actualmente, la versión estable más reciente de Java es la versión 8 sin embargo, para el desarrollo del proyecto se ha tenido que utilizar Java 7. Esto es debido a que Android no admite todas las características de Java 8, solo un subconjunto de ellas. Además, para utilizar dicho subconjunto de Java 8 hay que realizar una serie de configuraciones que puede causar una serie de problemas, como por ejemplo que la función Instant Run de Android Studio, comentada en siguientes apartados, no funciona correctamente y debe permanecer deshabilitada. Por ello, se ha utilizado Java 7 que es totalmente compatible y así evitar problemas de compatibilidad.

PHP

Se trata de un lenguaje de programación de propósito general que puede ser usado, entre otros, para la comunicación entre maquinas, diseño de páginas web o acceso a bases de datos. Se trata de un código del lado del servidor, lo que quiere decir que se encarga de procesar peticiones mediante la interpretación de un script en el servidor web, con el objetivo de generar dinámicamente páginas HTML como respuesta. Además, puede ser utilizado en la gran mayoría de servidores y en prácticamente todos los sistemas operativos.

En nuestro proyecto, PHP ha sido usado para implementar la lógica de nuestro servidor, por lo tanto, es el responsable de la comunicación entre cliente y servidor. Para ello, se han creado un conjunto de scripts que responden ante las posibles peticiones que puede realizar iNurse de modo que, dado un grupo de parámetros de entrada, cada script realizará la correspondiente consulta a base de datos y generará una respuesta en formato JSON, que será interpretada por iNurse para extraer la información solicitada.

Los motivos por los cuales se ha decidido utilizar PHP para realizar esta función son:

- Capacidad de conexión con la mayoría de gestores de bases de datos, entre ellos MySQL el cual ha sido utilizado en este proyecto.
- Proporciona el acceso a bases de datos de una forma muy sencilla.
- Es libre y abierto, por lo que puede ser utilizado sin coste alguno.
- Su curva de aprendizaje es baja por lo que su uso no ha supuesto una gran inversión de tiempo.

4.3. Entorno programación

Android Studio ha sido el entorno de desarrollo integrado o IDE (*Integrated Development Environment*) usado para el desarrollo de iNurse. Está basado en IntelliJ IDEA y posee una serie de características que lo convierten actualmente en el mejor entorno de desarrollo Android.

Dispone de una función novedosa llamada Instant Run la cual permite aplicar cambios en el código y los recursos de nuestra aplicación mientras se ejecuta. Instant Run interpreta de forma inteligente los cambios realizados y a menudo los entrega sin reiniciar la aplicación ni volver a compilar el código, de forma que podemos ver nuestros cambios de manera inmediata (Google, 2016).

También dispone de un emulador denominado Android Emulator, que permite instalar e iniciar las aplicaciones más rápidamente que en un dispositivo real. Nos permite probar nuestros prototipos en cualquier configuración de dispositivo Android y simular todo tipo de funciones como la localización GPS o las funciones multitáctiles.

Además, proporciona herramientas GUI que facilitan mucho las tareas del programador, una de las más útiles es el editor de diseño que ofrece. Las interfaces se generan a través de ficheros XML los cuales definen la posición y características de los elementos que componen dicha interfaz. Android Studio ofrece un editor visual con la función de arrastrar y colocar, lo que supone una gran facilidad a la hora de la creación de diseños, pudiendo ver en tiempo real los resultados de nuestro diseño en cualquier configuración de pantalla.

Finalmente comentar que también ofrece integraciones con herramientas de control de versiones tales como GitHub y Subversion. En este proyecto se ha hecho uso de GitHub y gracias a su integración en Android Studio su configuración y uso han sido triviales.

Todo esto y muchas más funciones interesantes que ofrece han hecho que este proyecto se decante por utilizar este IDE y no otro, tal como podría haber sido Eclipse.

4.4. Plataforma de desarrollo

Como se ha mencionado anteriormente, en iNurse se ha utilizado la plataforma de desarrollo colaborativo GitHub para alojar el proyecto. El código es almacenado de forma pública, aunque también permite hacerlo de forma privada mediante la utilización de cuentas de pago.

GitHub utiliza el sistema de control de versiones Git, el cual es gratis y de código abierto. Está diseñado para manejar tanto proyectos pequeños como proyectos grandes con eficiencia y rapidez. En iNurse se han realizado un total de 25 commits tal y como podemos observar en la ilustración 13.

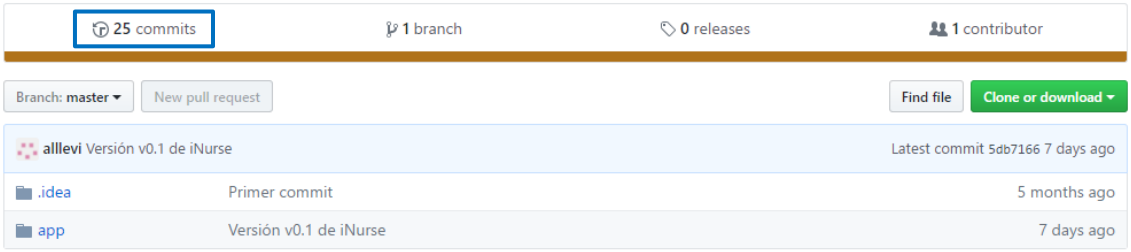

Ilustración 13. Uso de GitHub en iNurse

La principal característica de Git es que se trata de un sistema distribuido por lo que no necesita de un servidor central donde resida el proyecto. Cada programador tiene una copia en su máquina del repositorio entero de forma que, al realizar un cambio este se propaga a todos los demás nodos. Esto le proporciona un gran rendimiento y que las búsquedas sean muy eficaces, lo que se traduce en una gran rapidez para detectar diferencias entre ficheros. Esta característica es la que más le diferencia de otros sistemas de control de versiones, como por ejemplo Subversion el cual es centralizado.

4.5. Formato de intercambio de información

El formato elegido para el intercambio de información entre cliente y servidor es JSON. Se trata de un formato ligero de intercambio de datos el cual resulta fácil de leer y escribir para los humanos y simple de generar e interpretar por las maquinas.

Según (Ecma International, 2013) JSON es un formato de texto que es completamente independiente del lenguaje de programación, pero utiliza convenciones que son ampliamente conocidas por los programadores. JSON está formado por dos tipos de estructuras básicas:

- Una colección de pares de nombre-valor. En varios lenguajes esto es conocido como un objeto, registro, estructura o tabla hash.
- Una lista ordenada de valores. En la mayoría de los lenguajes, esto se implementa como arrays, vectores, listas o secuencias.

Las estructuras mencionadas son estructuras universales, virtualmente todos los lenguajes de programación las soportan de una forma u otra. Es razonable que un formato de intercambio de datos que es independiente del lenguaje de programación se base en estas estructuras.

En JSON, las estructuras mencionadas anteriormente se presentan de las siguientes formas:

- **Objetos**: conjunto desordenado de pares nombre-valor. Un objeto comienza con una llave de apertura representada por el carácter “{” y termina con una llave de cierre “}”. Cada nombre es seguido por “:” y los pares nombre-valor se separan mediante comas. En la ilustración 14 podemos ver con mayor claridad cómo se compone un objeto.

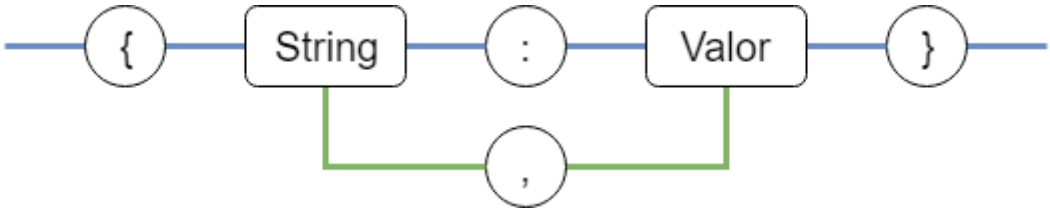

Ilustración 14. Objetos JSON

- **Arrays**: un array es una colección de valores. Un array comienza con un corchete izquierdo “[“ y acaba con un corchete derecho “]” (ver ilustración 15). Los valores se separan mediante comas.

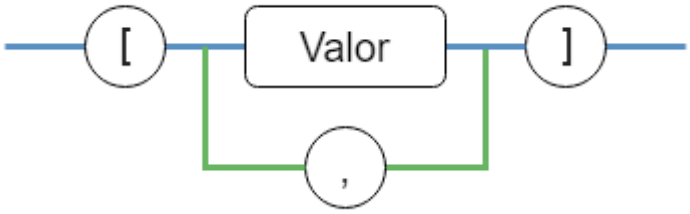

Ilustración 15. Arrays JSON

- **Valor**: un valor puede ser una cadena de caracteres con comillas dobles, un número, un valor booleano (true o false), null, un objeto o un array.

En la ilustración 16 podernos ver un fragmento JSON extraído de un mensaje mandado por el servidor a la aplicación iNurse, el cual devuelve información sobre los pacientes ingresados.

```
{
   "mapa_hospitalario":[
      {
         "mapa_key":"1",
         "mapa_planta":"1",
         "mapa_habitacion":"01A",
         "paci_nombre":"Alejandro",
         "paci_primer_apellido":"Lledó",
         "paci_segundo_apellido":"Viciano",
         "paci_sexo":"Hombre",
         "paci_motivo_ingreso":"Fractura de brazo",
         "paci_key":"1"
      },
      {
         "mapa_key":"2",
         "mapa_planta":"1",
         "mapa_habitacion":"01B",
         "paci_nombre":"Marta",
         "paci_primer_apellido":"Fernández",
         "paci_segundo_apellido":"Martí",
         "paci_sexo":"Mujer",
         "paci_motivo_ingreso":"Neumonía leve",
         "paci_key":"2"
      }
   ]
}
```


Ilustración 16. Ejemplo de mensaje JSON

4.6. Gestor de base de datos

En este proyecto se ha escogido MySQL como sistema de gestión de bases de datos relacionales. Actualmente, se trata de una de las bases de datos más populares del mercado y es utilizada por grandes aplicaciones tales como Twitter, Facebook o YouTube.

Es desarrollada por Oracle y se distribuye bajo licencia dual GPL/Licencia comercial. La licencia GPL (*General Public License*) permite que la utilización de MySQL sea gratuita y que además se pueda modificar con total libertad. En este proyecto no se ha realizado ninguna modificación sobre MySQL, pero el hecho de que se sea software libre era una característica indispensable para este proyecto.

Además, ofrece una serie de características que hacen que MySQL sea uno de los gestores de bases de datos más rápidos del mercado. Según (Tangient LLC, 2011) las propiedades más destacadas son las siguientes:

- **Gran escalabilidad y flexibilidad**, siendo capaz de soportar diferentes versiones de Windows, Linux, y UNIX.
- **Alto rendimiento**. Permite configurar el servidor MySQL para obtener el máximo rendimiento para aplicaciones específicas.
- **Alta disponibilidad**, ofrece un gran abanico de soluciones tales como la replicación o la utilización de clústeres de alta disponibilidad.
- **Transaccionalidad robusta**, ya que soporta todas las características transaccionales que puede disponer una base de datos, las cuales son: atomicidad, consistencia, aislamiento y durabilidad.

- **Protección de datos**. MySQL ofrece potentes mecanismos de autentificación y tiene soporte SSH y SSL para asegurar conexiones seguras. Además, dispone de una estructura de privilegios que permite que un usuario solo pueda acceder a los datos que se le permite, así como potentes algoritmos de cifrado y descifrado de datos. Este punto es crítico en el caso concreto del sector de la sanidad, ya que como se ha comentado anteriormente, los datos de un paciente son muy sensibles.
- Dispone de **APIs** para gran cantidad de lenguajes tales como PHP, Java, C, C++, etc.

Por otro lado, como administrador de base de datos para nuestro proyecto se ha escogido phpMyAdmin. Se trata de una herramienta software de código abierto escrita en PHP diseñada con el propósito de administrar y gestionar bases de datos MySQL a través de navegadores web. Soporta la ejecución de una gran cantidad de operaciones a través de una interfaz gráfica intuitiva. También ofrece la posibilidad de ejecutar directamente sentencias SQL.

5. Diseño y arquitectura

En este capítulo, se va a mostrar el diseño y los patrones arquitectónicos que van a ser usados por iNurse. Desde el principio se ha pretendido que iNurse disponga de una estructura correctamente definida con la que poder cumplir sus funciones de una forma organizada y eficiente. Además, se va a mostrar un mapa de navegabilidad que nos proporcione en un golpe de vista las navegaciones entre interfaces.

5.1. Arquitectura software

En este apartado se va a explicar la arquitectura diseñada para la aplicación iNurse. Definir una buena arquitectura que se adapte a nuestras necesidades en las etapas tempranas del desarrollo es de gran importancia ya que, define la estructura general del sistema y las interacciones entre los diferentes componentes del sistema.

5.1.1. Arquitectura cliente - servidor

iNurse ha seguido el modelo de programación por capas, el propósito principal de este modelo es el desacoplamiento de las diferentes partes que componen un sistema software separando la lógica de la aplicación en diferentes componentes, reduciendo de este modo el número de dependencias entre los componentes de la aplicación. Las capas que forman una aplicación software son las siguientes:

- **Capa de presentación**: es la encargada de presentarle el sistema al usuario, capturar la información del usuario con un nivel de procesamiento mínimo para mandarla a la capa de negocio y comunicar la información recibida desde la lógica de negocio.
- **Capa de negocio**: se ocupa de gestionar los datos a nivel de procesamiento y es donde se establecen todas las reglas de negocio que debe cumplir la aplicación. Actúa de intermediario entre la capa de presentación y la capa de datos.
- **Capa de datos**: es donde se almacenan los datos y es el responsable de acceder a ellos. Está formada por al menos un gestor de bases de datos que se encarga de realizar todo el almacenamiento y atender a las peticiones de la capa de negocio.

Para iNurse, se ha decidido utilizar una arquitectura en dos capas, también conocida como arquitectura cliente-servidor, la cual distribuye la aplicación en dos componentes lógicos. Existen diferentes variantes en la arquitectura cliente-servidor dependiendo de cómo se distribuyan las 3 capas lógicas. En el caso de iNurse, nos hemos decantado por la variante que retira el manejo de datos de la aplicación (ver ilustración 17), de forma que en el servidor residirá la capa de datos y en la aplicación móvil las capas restantes.

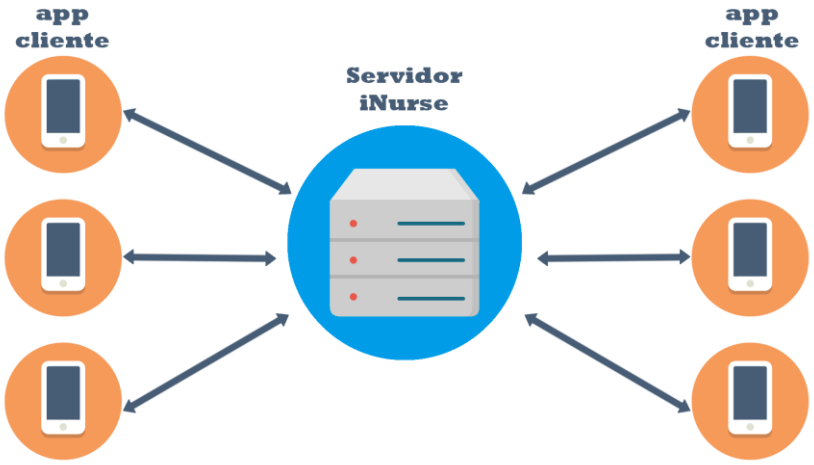

Ilustración 17. Arquitectura cliente - servidor

5.1.2. Patrón DAO - DTO

Un Objeto de Transferencia de Datos (DTO por sus siglas en inglés) es un objeto usado para la encapsulación de datos entre procesos. Los DTOs son objetos simples que no deben contener lógica de negocio por lo que no tiene más comportamiento que almacenar y entregar sus propios datos.

Un Objeto de Acceso a Datos (DAO por sus siglas en inglés) es un componente software que encapsula el acceso a la base de datos, proporcionando una interfaz común entre la aplicación y el almacenamiento de datos. Una de las principales ventajas de este patrón es que, al aislar una aplicación de la tecnología de persistencia, dicha tecnología podría ser actualizada o modificada sin la necesidad de modificar otras partes de la aplicación.

La utilización de estos patrones en iNurse es bien simple. Los DAO se encargan de implementar los accesos a la base de datos de modo que cuando la aplicación solicita determinados datos que se encuentran en la base de datos, realiza una invocación al método correspondiente de un determinado DAO, pero sin conocer como el DAO interactúa con la base de datos. Una vez el DAO ha recuperado la información de la base de datos, construye un objeto DTO con toda la información obtenida y este será el objeto devuelto por el DAO.

5.1.3. Base de datos

Una vez definida la arquitectura de la aplicación y los patrones software a utilizar, hay que diseñar el modelo de base de datos. Realizar un buen diseño es importante ya que, nos permite recuperar y actualizar la información almacenada con mayor facilidad y garantiza que la base de datos sea sencilla de mantener. En la ilustración 18, podemos observar un diagrama simplificado de la base de datos de iNurse donde podemos ver las relaciones entre las tablas.

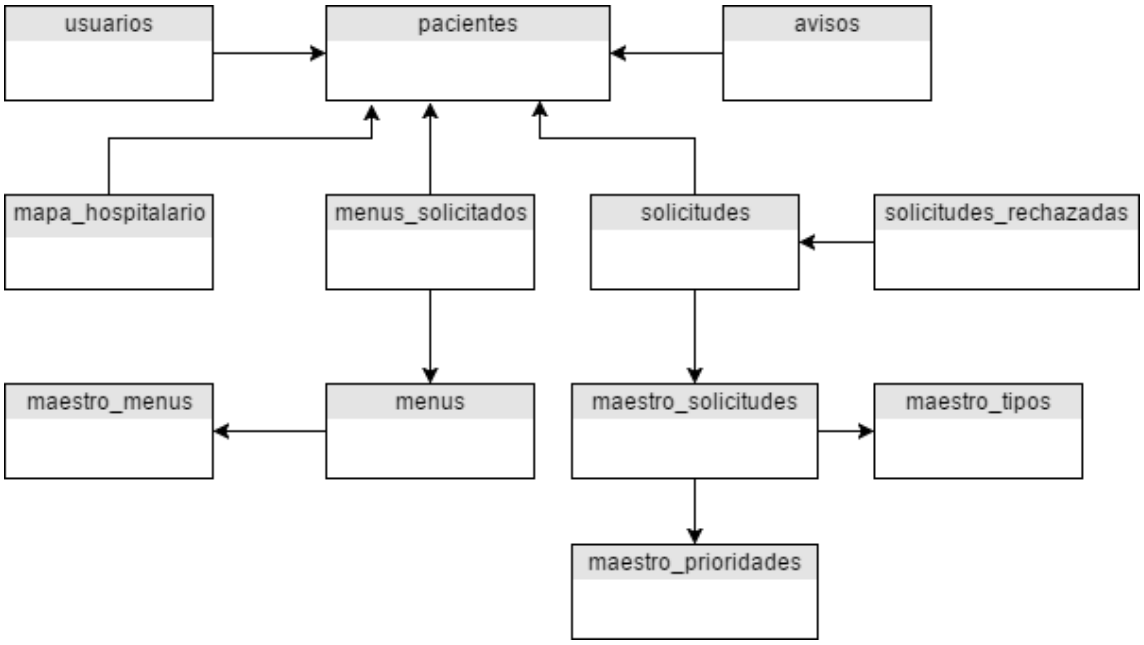

Ilustración 18. Tablas relacionales de iNurse

A continuación, se explicará de forma breve y concisa las tablas y relaciones que forman la base de datos:

- **usuarios**: contiene la información de inicio de sesión y el rol de todos los usuarios de la aplicación.
- **pacientes**: almacena toda la información relativa a un paciente (nombre, SIP, fecha nacimiento, sexo, etc.).
- **avisos**: reside la información sobre los avisos programados a los pacientes.
- **mapa_hospitalario**: tabla que establece la relación entre un paciente y la habitación donde reside.
- **maestro_menus**: contiene todas las descripciones de los menús que puede ofrecer un centro.
- **maestro_solicitudes**: almacena todas las descripciones de las solicitudes que tiene disponibles el hospital, así como su prioridad.
- **maestro_tipos**: contiene los estados en los que se puede encontrar una solicitud.
- **maestro_prioridades**: guarda el conjunto de prioridades disponibles para las solicitudes.
- **menus**: cada registro de esta tabla conforma un menú que podrá ser elegido por el paciente. Un menú se forma a partir de 3 ítems de la tabla “maestro_menus”. Además, se distingue si se trata de un menú desayuno, comida, merienda o cena.
- **menus_solicitados**: relaciona un menú con un paciente.
- **solicitudes**: contiene la información de una solicitud realizada por un paciente.
- **solicitudes_rechazadas**: registra las solicitudes rechazadas y el motivo de rechazo.

5.1.4. Firebase Cloud Messaging

Firebase Cloud Messaging (FCM) es una solución multiplataforma que permite enviar mensajes y notificaciones de forma segura. Esto nos permite enviar notificaciones *push* a nuestros dispositivos Android, siendo el siendo el servidor el que inicia el proceso de notificación cuando se produce un determinado evento, evitando de este modo que la aplicación cliente esté continuamente realizando consultas al servidor, ahorrando así batería y recursos. Además, las notificaciones *push* disponen de una gran ventaja y es que, aunque la aplicación se encuentre apagada o en segundo plano, nuestro dispositivo recibirá la notificación, por lo que evitamos realizar operaciones de sincronización con el servidor para recuperar posibles notificaciones perdidas mientras la aplicación no estaba en ejecución.

Entre las funcionalidades que ofrece FCM, la que más interesa a este proyecto es la que ofrece una gran versatilidad a la hora de enviar los mensajes ya que permite distribuirlos de tres formas diferentes:

- A dispositivos individuales: permite enviar notificaciones personalizadas a cada usuario de la aplicación.
- A grupos de dispositivos: esta opción posibilita la difusión de notificaciones a grupos de pacientes.
- A dispositivos suscritos a temas: esta funcionalidad permite que los usuarios de la aplicación decidan qué tipo de información desean recibir suscribiéndose a los tópicos que le resulten de interés.

Además, dispone de la consola administrativa Firebase Analytics (ver ilustración 19) la cual nos permite obtener información sobre el uso de la app y la interacción del usuario. Las funciones más destacadas son la generación de reportes sobre 500 tipos diferentes de eventos con 25 atributos cada uno de ellos y su *dashboard* que permite monitorizar el comportamiento del usuario y conocer la segmentación demográfica de edad, género y localización (Google, 2016).

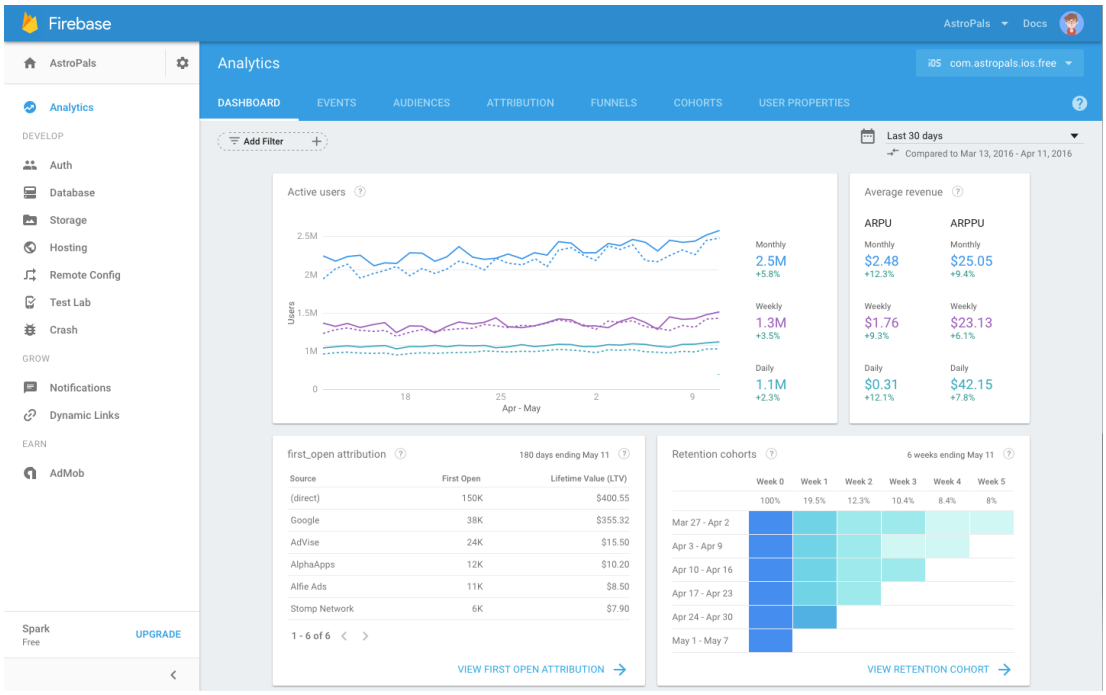

Ilustración 19. Firebase Analytics

El servicio de FCM dispone de un plan gratuito (ver ilustración 20), ideal para las primeras versiones de la aplicación. La limitación de este plan gratuito que podría condicionar el crecimiento de iNurse es el número limitado de conexiones simultaneas, ya que, si la aplicación llegase a ser implantada en una buena cantidad de hospitales, podrían darse más de 100 conexiones simultaneas. En el caso de llegar a este punto, debería estudiarse la contratación de un plan de pago con coste o implementar un servicio propio de difusión de notificaciones utilizando una tecnología alternativa que posibilite el envío de notificaciones *push*, como por ejemplo Node.js.

		SPARK Free Generous limits for hobbyists	FLAME $25/month Predictable pricing for growing apps	BLAZE Pay as you go Commodity pricing for apps at scale
Included Free Analytics, App Indexing, Authentication, Dynamic Links, Invites, Notifications, Crash Reporting, & Remote Config		✓	✓	✓
Realtime Database	Simultaneous connections	100	Unlimited [1]	Unlimited [1]
	GB stored	1 GB	2.5 GB	$5/GB
	GB transferred	10 GB	20 GB	$1/GB
	Daily private backups	✗	✓	✓

Ilustración 20. Planes de pago de FCM

Con la inclusión de la plataforma FCM a nuestra aplicación, la arquitectura final de la aplicación quedaría tal y como muestra la ilustración 21.

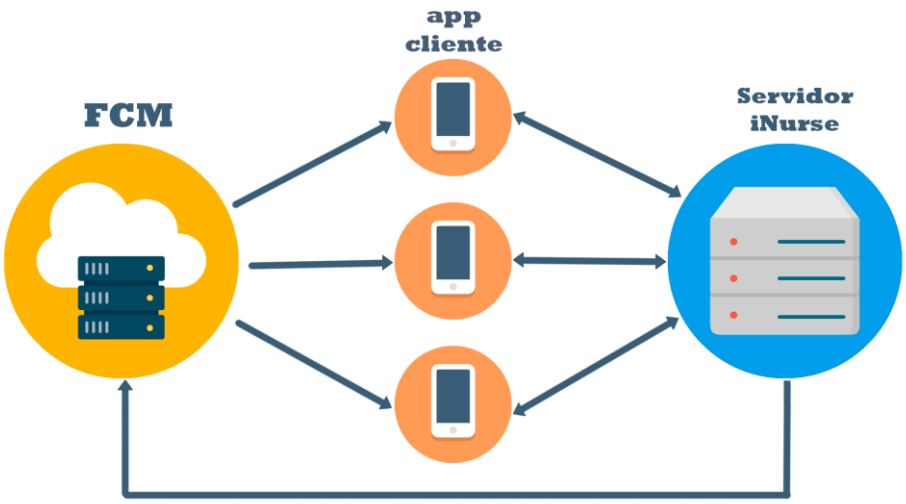

Ilustración 21. Arquitectura de iNurse

Como podemos observar, la plataforma FCM únicamente se encarga de entregar las notificaciones a los dispositivos que corresponda en cada caso, eximiendo al servidor iNurse de la responsabilidad de gestionar el envío de notificaciones. El servidor únicamente se limita a proporcionarle los datos necesarios a FCM sobre las notificaciones que desea enviar. Los datos que necesita conocer FCM son: la información que va a contener la notificación y los usuarios que debe recibir dicha notificación.

5.2. Mapa de navegabilidad

Con la finalidad de obtener una imagen que facilite la comprensión de iNurse y que ofrezca una visión global de las interacciones entre las interfaces, se ha elaborado el mapa de navegabilidad que podemos ver en la ilustración 22.

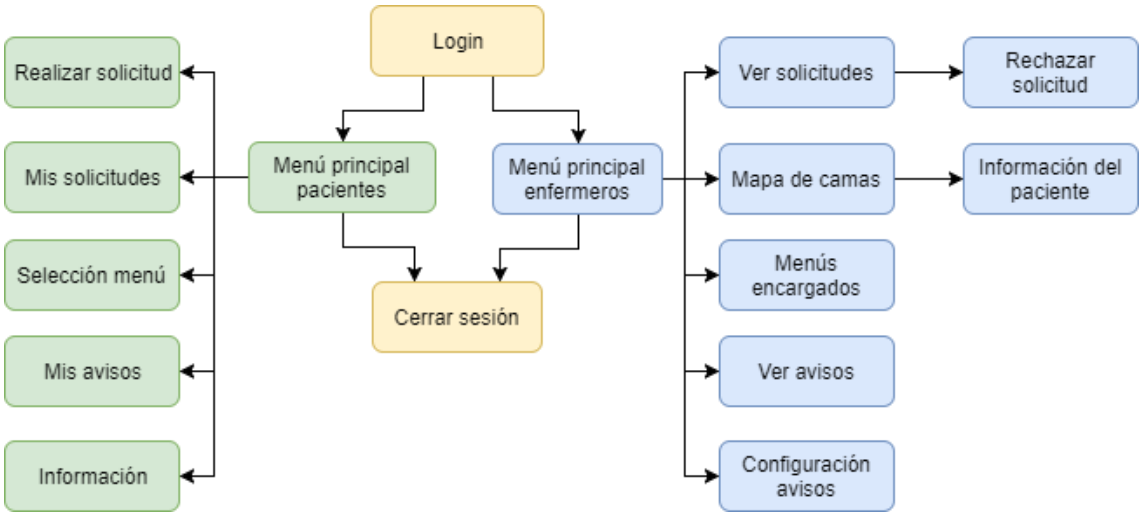

Ilustración 22. Mapa navegabilidad

Gracias al mapa de navegabilidad, podemos en primer lugar observar fácilmente la navegación entre las diferentes pantallas y, en segundo lugar, identificar a que pantallas puede acceder un determinado rol.

Los colores verde y azul separan claramente la aplicación en dos partes, que se corresponden con los dos *front-end* de iNurse, comentados en apartados anteriores. Por una parte, el color verde se corresponde con aquellas interfaces a las que tienen acceso los pacientes, aquí tenemos el *front-end* del paciente. Por otra parte, aquellas interfaces representadas de azul conforman el *front-end* de los enfermeros.

Por último, en color amarillo tenemos aquellas interfaces a las que tiene acceso ambos roles, estas son la pantalla de “Login” y la pantalla “Cerrar sesión”. Estas dos pantallas proporcionan el acceso y la salida de la aplicación a los usuarios y por ello, son interfaces compartidas entre ambos roles.

5.3. Estructura del proyecto

En la ilustración 23, podemos observar la estructura de nuestro proyecto dentro del entorno de desarrollo Android Studio.

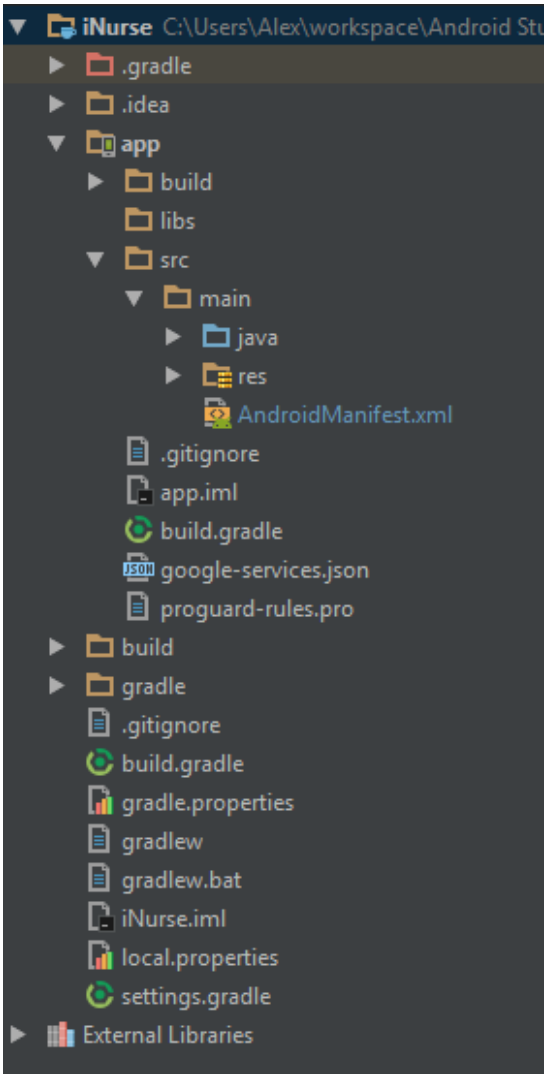

Ilustración 23. Estructura proyecto iNurse

Tal y como podemos observar, nuestro proyecto se compone de un gran número de paquetes y ficheros, la mayoría autogeneradas por el propio entorno. A continuación, se van a describir las partes más importantes del proyecto.

Paquete app/src/main/java

Este paquete contiene todo el código fuente de la aplicación. Dentro de este paquete, se han organizado las clases Java de una forma estructurada, siendo agrupadas según su funcionalidad. Podemos observar la jerarquía de paquetes de iNurse en la ilustración 24.

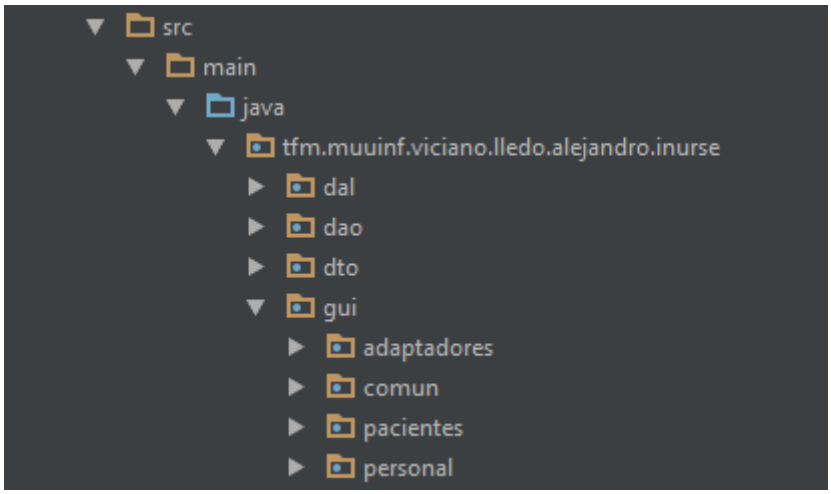

Ilustración 24. Subestructura paquete app/src/main/java

En la siguiente tabla, se describen brevemente los paquetes mostrados en la anterior ilustración.

Nombre paquete	Descripción
dal	Contiene el objeto DAL
dao	Contiene todos los objetos DAO
dto	Contiene todos los objetos DTO
gui	Paquete padre que engloba los subpaquetes que contienen las Activity de la aplicación
gui/adaptadores	Contiene los adaptadores necesarios para construir las CardView disponibles
gui/comun	Contiene las Activity comunes a los roles paciente y enfermero
gui/pacientes	Contiene las Activity propias al rol paciente
gui/personal	Contiene las Activity empleadas por el rol de enfermero

Paquete app/src/main/res

Este paquete contiene todos los ficheros de recursos necesarios para el proyecto, entre los cuales podemos encontrar *layouts*, cadenas de texto, imágenes, animaciones, etc. Los diversos tipos de recursos se organizan jerárquicamente en diferentes subpaquetes. En la ilustración 25, podemos observar los recursos de los que dispone iNurse.

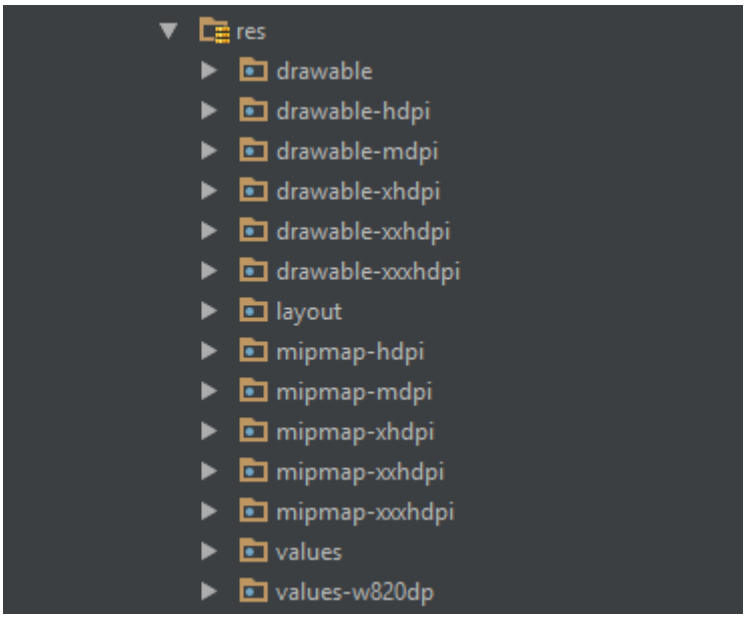

Ilustración 25. Subestructura paquete app/src/main/res

Como podemos observar cada paquete contiene un conjunto de recursos, agrupados por su tipo. Hay ciertos tipos de recursos que no son utilizados por iNurse y por lo tanto no aparecen en la ilustración. En la siguiente tabla, se realiza una breve descripción de los paquetes en los que se pueden distribuir los recursos de una aplicación.

Paquete	Contiene
animator	Animaciones
color	Ficheros XML para la definición de colores
drawable	Elementos gráficos. Se divide en subcarpetas para definir recursos dependiendo de la densidad y resolución de pantalla del dispositivo: ○ drawable (independientes de la densidad) ○ drawable-ldpi (densidad baja) ~120 dpi ○ drawable-mdpi (densidad media) ~160 dpi ○ drawable-hdpi (densidad alta) ~240 dpi ○ drawable-xhdpi (densidad extra alta) ~320 dpi ○ drawable-xxhdpi (densidad extra extra alta) ~480 dpi ○ drawable-xxxhdpi (densidad extra extra extra alta) ~640 dpi
layout	Ficheros XML que definen las interfaces gráficas de usuario. Se puede dividir en subcapetas dependiendo de la densidad de pantalla y de la orientación del dispositivo. Por ejemplo: ○ layout-sw600dp (para tablets de 7”) ○ layout (orientación vertical) ○ layout-land (orientación horizontal)

menu	Definición XML de los menús de la aplicación
mipmap	Iconos de lanzamiento de la aplicación. Al igual que los recursos drawable, se divide en subcarpetas para adaptarse de forma óptima a la densidad de pantalla
raw	Recursos que no se incluyan en el resto de carpetas
values	Ficheros XML de recursos tales como cadenas de texto, tamaños, colores, arrays de valores, etc.
xml	Ficheros XML que contiene datos empleados por la aplicación

Fichero AndroidManifest.xml

Se trata de un fichero que deben contener todas las aplicaciones Android. Este archivo suministra información importante sobre la aplicación al sistema Android, información esencial que debe conocer para poder ejecutar el código de la app.

Entre otras, el archivo AndroidManifest.xml hace lo siguiente:

- Nombra el paquete de la aplicación. Dicho nombre sirve como identificador único para la aplicación.
- Describe los componentes que forman la aplicación tales como las actividades, los servicios o los receptores de mensajes.
- Declara los permisos con los que debe contar la aplicación para poder realizar determinadas acciones.
- Declara el nivel mínimo de Android API que requiere la aplicación.
- Enumera las bibliotecas con las que debe estar asociada la aplicación.

Fichero build.gradle

Para entender la función de este archivo hay que explicar en primer lugar que función realiza la herramienta Gradle. Su finalidad es la automatización de la construcción y compilación de un proyecto Android.

Dentro del archivo build.gradle, se especifica información como la versión del SDK de Android usada para compilar el código, la versión mínima de Android que soporta la aplicación o las librerías externas empleadas. En un proyecto pueden existir varios ficheros de este tipo, cada uno englobando un determinado nivel. Por ejemplo, un fichero a nivel de proyecto y otro a nivel de módulo. El primero de ellos fija parámetros globales a todos los módulos del proyecto, y el segundo únicamente actuará para cada módulo en concreto. En la ilustración 23, podemos observar como nuestro proyecto posee un build.gradle a nivel de proyecto y a nivel de módulo. En la ilustración 26, tenemos el fichero build.gradle que se encuentra a nivel de módulo en iNurse, destacando algunas de sus características.

```
apply plugin: 'com.android.application'
apply plugin: 'com.google.gms.google-services' ----> Plugin FCM

android {
    compileSdkVersion 25 ----> Versión SDK usada para compilar
    buildToolsVersion '25.0.2'
    defaultConfig {
        applicationId "tfm.muuinf.viciano.lledo.alejandro.inurse"
        minSdkVersion 24 ----> Versión mínima de API de Android
        targetSdkVersion 25
        versionCode 1
        versionName "1.0" ----> Versión de la aplicación
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'),
'proguard-rules.pro'
        }
    }
}

dependencies { ----> Dependencias con librerías externas
    compile fileTree(include: ['*.jar'], dir: 'libs')
    compile 'com.android.support:appcompat-v7:25.1.1'
    compile 'com.android.support:design:25.1.1'
    compile 'com.android.support:recyclerview-v7:25.1.1'
    compile 'com.android.support:cardview-v7:25.1.1'
    compile 'org.apache.commons:commons-lang3:3.5'
    compile 'org.apache.commons:commons-collections4:4.1'
    compile 'commons-io:commons-io:2.5'
    compile 'joda-time:joda-time:2.9.4'
    compile 'com.android.support.constraint:constraint-layout:+'

}
```

Ilustración 26. Fichero build.gradle iNurse

6. Implementación

En el presente capitulo, se van a explicar un conjunto de aspectos relacionados con los detalles de implementación de este proyecto. Se expondrán detalles técnicos acerca de la herencia de clases empleada, los componentes Android más usados, las distintas capas lógicas que componen la aplicación, la tecnología FCM y las consultas a base de datos a través de PHP.

6.1. Herencia

Una gran ventaja de Java es que soporta la herencia permitiendo que una clase incorpore a otra en su declaración gracias a la palabra clave *extends*. Gracias al uso de la herencia, podemos crear una clase general que defina rasgos comunes a un conjunto de clases relacionadas. Luego, otras clases, pueden heredar los rasgos de esta clase y además añadir funcionalidades propias, lo que las hace únicas.

Este principio se ha aplicado de forma personalizada en iNurse en dos ocasiones. La primera es mediante la creación de InurseActivity. Todas las Activity de iNurse extienden de InurseActivity. Una Activity es un componente de la aplicación que contiene una pantalla con la que los usuarios pueden interactuar para realizar una acción.

Los métodos más empleados que ofrece InurseActivity son los siguientes:

Método	Parámetros	Objeto retorno	Descripción
checkInternet	-	boolean	Comprueba si el dispositivo dispone de conexión a internet
getSharedPreferences	-	SharedPreferences	Devuelve el fichero de preferencias del dispositivo
onBackPressed	-	void	Ofrece una implementación al evento generado al pulsar la tecla “Atrás” del dispositivo
crearNotificación	String título String mensaje	void	Muestra una notificación al usuario con el título y el mensaje pasados como parámetros

Por otra parte, también se ha creado la clase InurseDAO. De ella extienden todas las clases DAO de iNurse, evitando tener que reimplementar los siguientes métodos en cada uno de los objetos DAO:

Método	Parámetros	Objeto retorno	Descripción
getHTTP	URL	JSONObject	Inicia una conexión HTTP a partir del objeto URL recibido como parámetro y devuelve un objeto JSON con la respuesta recibida
insertHTTP	URL	boolean	Inicia una conexión HTTP a partir del objeto URL recibido como parámetro y devuelve un boolean que indica si la operación se ha realizado con éxito
inputStreamToJSON	InputStream	JSONObject	Convierte un objeto InputStream en un objeto JSON

6.2. RecyclerView and CardView

Los widgets RecyclerView y CardView son los componentes visuales más empleados en la construcción de las interfaces de iNurse. Un RecyclerView es un contenedor empleado para mostrar en forma de lista un conjunto de datos grande de una forma muy eficiente, ya que internamente gestiona las vistas mostradas de forma que solo mantiene un número limitado de ellas. Es un componente perfecto para mostrar conjuntos de datos que pueden cambiar en tiempo de ejecución en respuesta a un evento producido por un usuario o por un evento de red. Por otra parte, un CardView es un componente que permite mostrar de forma organizada un conjunto de datos dentro de una tarjeta.

Por ello, las CardView constituyen los ítems a mostrar y el RecyclerView es el componente que contiene dichos ítems y, por tanto, el que se encarga de gestionar los recursos necesarios para mostrarlos eficientemente. Para el uso del RecyclerView es necesario implementar un adaptador. Principalmente, este adaptador proporciona acceso a los elementos del RecyclerView y especifica cómo crear el contenido de las vistas, en este caso las CardView. En la ilustración 27, podemos observar un ejemplo de cómo son usados estos componentes en iNurse.

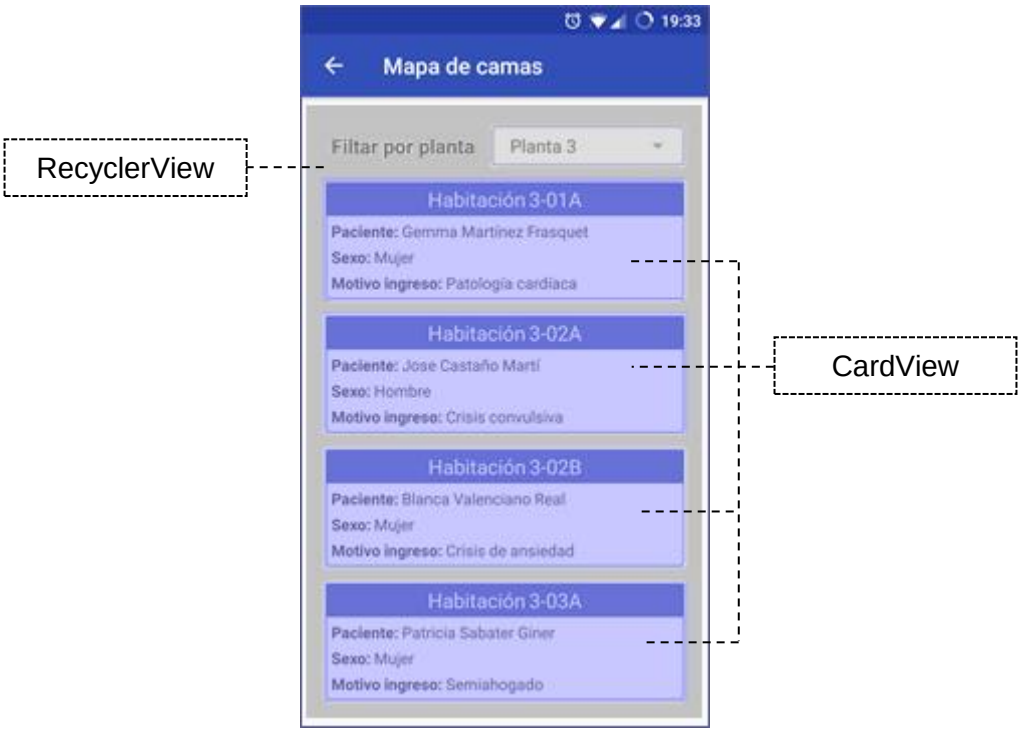

Ilustración 27. RecyclerView y CardView

6.3. Firecloud

Para poder hacer uso de los servicios de FCM desde iNurse, se ha tenido que realizar una configuración previa. En primer lugar, se ha tenido que crear un proyecto desde la consola de Firebase, tal y como muestra la ilustración 28.

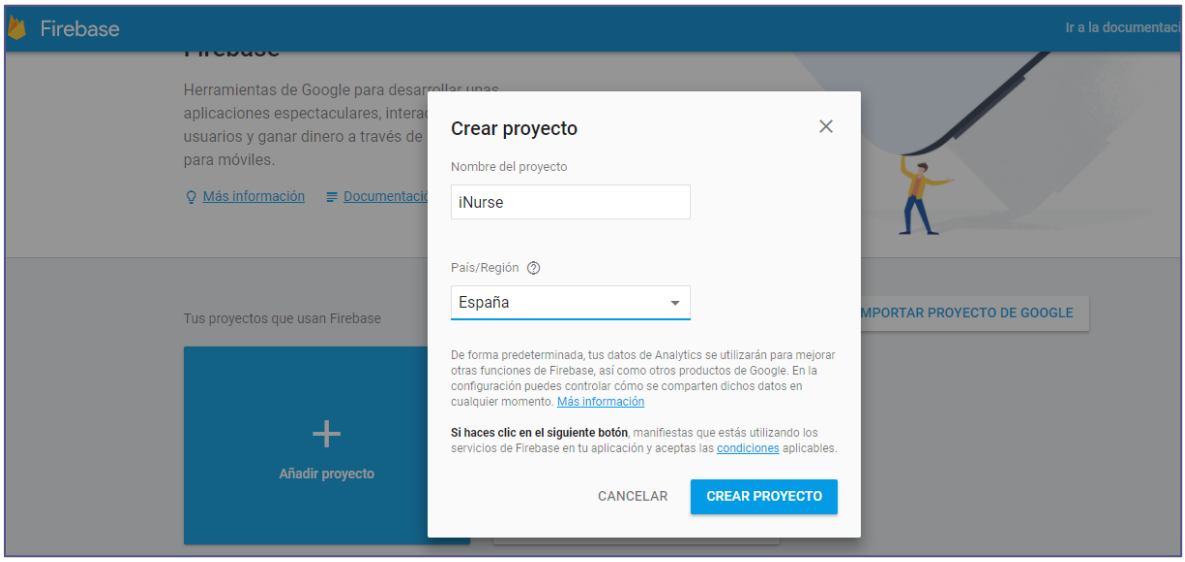

Ilustración 28. Creación proyecto FCM

Una vez creado el proyecto, se ha tenido que asociar una aplicación Android a dicho proyecto (ver ilustración 29), en nuestro caso iNurse. Para ello, se han seguido los pasos

de un sencillo asistente en el que es suficiente con indicar el nombre del paquete de la aplicación.

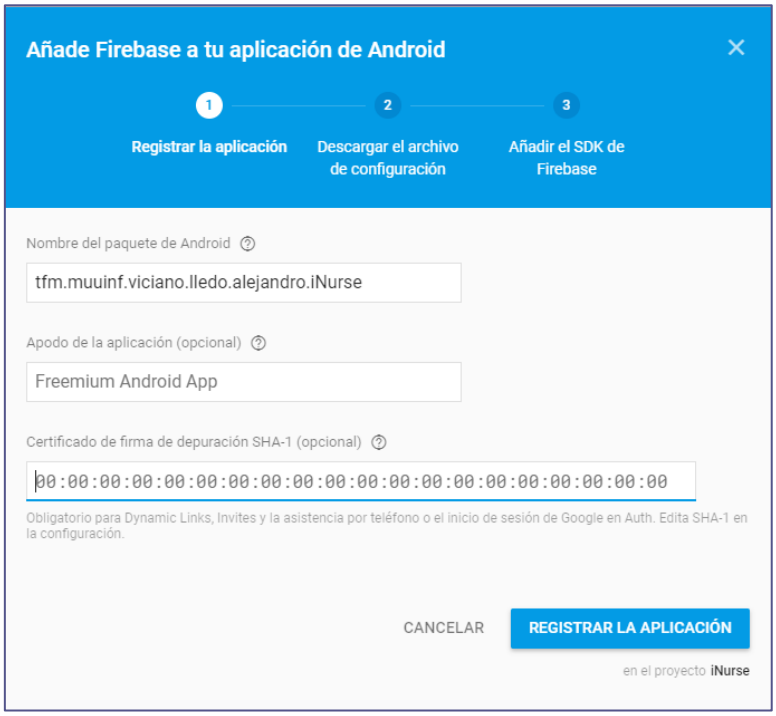

Ilustración 29. Registrar app FCM

Finalizado el paso anterior, se nos descargará un fichero de configuración llamado **google-services.json**, el cual hay que añadir en el directorio root del proyecto (ver ilustración 23). Por último, hay que configurar las dependencias necesarias para el correcto funcionamiento de FCM (ver ilustración 30), tal y como nos indica el asistente.

Ilustración 30. Añadir SDK de FCM

El siguiente paso es obtener un token de registro para cada instancia de la app iNurse. Este token sirve como identificador único de un dispositivo por lo que nos permitirá enviar notificaciones a dispositivos individuales o a grupos de dispositivos. Para obtenerlo, en el arranque inicial de la aplicación hay que realizar una llamada a los servicios de FCM para generar dicho token. En la ilustración 31, se muestra cómo realizar una llamada a FCM para obtener dicho token.

```
String refreshedToken = FirebaseInstanceId.getInstance().getToken();
```

Ilustración 31. Obtener token de registro FCM

Una vez obtenido el token de registro, debemos enviarlo al servidor y almacenarlo. En iNurse esta información se almacena en la tabla 'usuarios' de forma que el token queda asociado al usuario correspondiente. Si el usuario inicia sesión en más de un dispositivo, los tokens se almacenan de forma que el servidor es capaz de identificar que existe más de un token, notificando de este modo a todos los dispositivos de un usuario. En el caso de que el usuario cierre la sesión en un dispositivo, este token será borrado de forma que no recibirá más notificaciones en dicho dispositivo hasta que vuelva a iniciar sesión y se genere un nuevo token.

Por otra parte, hay que enviar las notificaciones a los dispositivos desde el servidor. Para ello, hay que generar una solicitud POST que debe constar de dos partes: encabezado HTTP y cuerpo HTTP.

El encabezado HTTP está formado por dos campos:

- **Content-Type**: identificador del formato del contenido transmitido en la petición HTTP.
- **Authorization**: permite que nuestro servidor se autentique frente a los servidores de FCM.

Para conocer el token de identificación de nuestro servidor, debemos entrar en la consola administrativa de Firebase (ver ilustración 32).

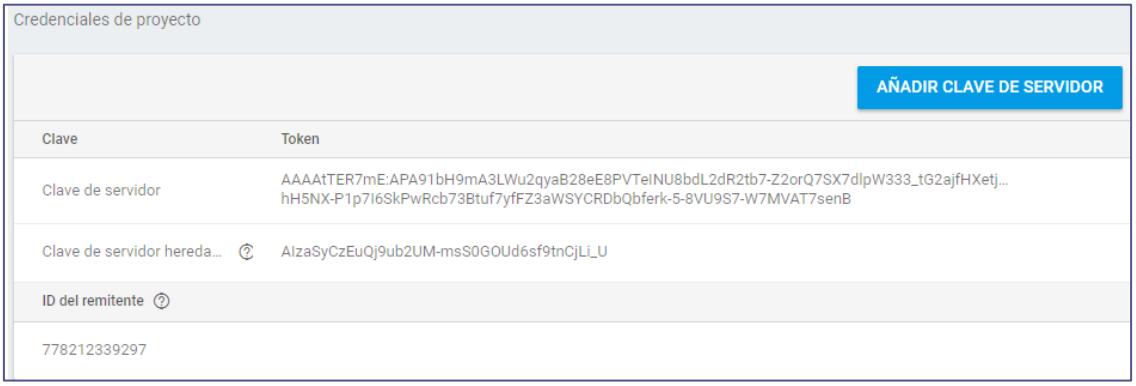

Ilustración 32. Token FCM del servidor iNurse

El mensaje HTTP debe estar compuesto por al menos dos campos:

- **to**: Este parámetro especifica el destinatario de un mensaje, el valor debe ser un token de registro. Se pueden especificar varios destinatarios.
- **data**: especifica los pares de clave-valor personalizados de la carga del mensaje.

En la ilustración 33, se muestra un ejemplo de un mensaje enviado por el servidor de iNurse.

```
https://fcm.googleapis.com/fcm/send
Content-Type:application/json
Authorization:key= AAAAtTER7mE:APA91bH9mA3LWu2qyaB28eE8... ---> Token servidor

{
  "to": "bk3RNwTe3H0:CI2k_HHwgIpoDKCIZvvDMExUdFQ...",  ---> Token dispositivo

"data": { ---> Mensaje
       "titulo": "¡Una solicitud ha cambiado de estado!",
       "descripcion": "Su solicitud \"Fuga de agua\" ha sido finalizada"
   }
}
```

Ilustración 33. Mensaje enviado por el servidor iNurse a FCM

Cuando una notificación es enviada por el servidor, todos los dispositivos destino la reciben y leen los datos contenidos en el mensaje JSON para ser mostrados en una notificación.

6.4. Capa DAL

La capa de acceso a datos o DAL (del inglés *Data Access Layer*) es una capa cuya finalidad es proporcionar un acceso simplificado a los objetos DAO. Su función es devolver una referencia del objeto DAO, proporcionando de este modo un nivel adicional de abstracción, ya que todas las llamadas a los DAO se realizan desde esta clase de modo que, cualquier cambio solo afectaría a la capa DAL. En la ilustración 34, podemos observar la implementación del objeto DAL.

```
public class ServiciosDAL {

    private final MenusDAO menusDAO;
    private final PacienteDAO pacienteDAO;
    private final SolicitudDAO solicitudDAO;
    private final MaestrosDAO maestrosDAO;
    private final AvisosDAO avisosDAO;
    private final MapaHospitalarioDAO mapaHospitalarioDAO;

    public ServiciosDAL() {
        menusDAO = new MenusDAO();
        pacienteDAO = new PacienteDAO();
        solicitudDAO = new SolicitudDAO();
        maestrosDAO = new MaestrosDAO();
        avisosDAO = new AvisosDAO();
        mapaHospitalarioDAO = new MapaHospitalarioDAO();
    }

    public MenusDAO getMenusDAO() {
        return menusDAO;
    }

    public MenusDAO getMenuDAO() {
        return menusDAO;
    }

    public PacienteDAO getPacienteDAO() {
        return pacienteDAO;
    }

    public SolicitudDAO getSolicitudDAO() {
        return solicitudDAO;
    }

    public MaestrosDAO getMaestrosDAO() {
        return maestrosDAO;
    }

    public AvisosDAO getAvisosDAO() {
        return avisosDAO;
    }

    public MapaHospitalarioDAO getMapaHospitalarioDAO() {
        return mapaHospitalarioDAO;
    }
}
```

Ilustración 34. Capa DAL

Como podemos observar, esta capa cumple la simple función de encapsular todas las llamadas a los objetos DAO devolviendo una referencia del objeto. Una vez obtenida la referencia al objeto DAO, ya se puede trabajar con él y llamar a los métodos correspondientes. En la ilustración 35, se muestra como su uso es muy sencillo.

```
{...}
// Incializar el objeto DAL
ServiciosDAL dal = new ServiciosDAL();

// Obtener la referencia al DAO correspondiente y acceder
// al método del objeto DAO
List<SolicitudDTO> listaSolicitudes = dal.getSolicitudDAO().getAllSolicitudes();

{...}
```

Ilustración 35. Ejemplo de uso del objeto DAL

6.5. Patrón DTO

Tal y como se ha mencionado anteriormente, un objeto DTO es usado para la encapsulación de datos para su transferencia. En iNurse se ha creado un conjunto de objetos DTO que almacenan los datos recibidos desde el servidor, donde cada DTO contiene los datos referentes a un determinado tipo de objeto. De esta forma, la aplicación tiene acceso a un conjunto de datos mediante una única invocación al servidor. En la ilustración 36, se muestra un ejemplo sencillo de cómo se implementa un objeto DTO.

```
public class UsuarioDTO {

    private Integer key, pacienteKey;
    private String usuario, password, tipo;

    public UsuarioDTO (Integer key, String usuario, String password, String tipo,
    Integer pacienteKey) {
        this.key = key;
        this.usuario = usuario;
        this.password = password;
        this.tipo = tipo;
        this.pacienteKey = pacienteKey;
    }

    public Integer getKey() {
        return this.key;
    }

    public String getUsuario() {
        return usuario;
    }

    public String getPassword() {
        return password;
    }

    public String getTipo() {
        return this.tipo;
    }

    public Integer getPacienteKey() {
        return this.pacienteKey;
    }
}
```

Ilustración 36. UsuarioDTO

Como podemos observar, el objeto UsuarioDTO solo almacena información y no implementa ningún tipo de lógica. En la siguiente tabla se describen de forma breve los DTO existentes en iNurse.

Nombre	Descripción
AvisoConfiguracionDTO	Contiene los datos necesarios para la configuración de un aviso.
AvisosDTO	Almacena toda la información acerca de un aviso
MaestroPrioridadesDTO	Encapsula los datos de un maestro prioridad
MaestroSolicitudDTO	Guarda la información de un maestro solicitud
MaestroTiposDTO	Almacena los datos referentes a un maestro tipos
MapaHospitalarioDTO	Guarda toda la información relativa al mapa de camas
MenuDTO	Encapsula los datos sobre un menú
PacienteDTO	Almacena toda la información sobre un paciente
SolicitudDTO	Guarda los datos que conforman una solicitud
UsuarioDTO	Contiene la información básica sobre un usuario

6.6. Patrón DAO

Como se ha explicado anteriormente, para abstraer a la aplicación de la tecnología empleada para el acceso a base de datos, se ha hecho uso de los objetos DAO. Este patrón evita que se incluyan llamadas al servidor en cualquier parte del código, de forma que, si se tiene que realizar alguna modificación, como por ejemplo un cambio de tecnología, solo sería necesario modificar la clase DAO. En la ilustración 37, podemos ver un fragmento de una clase DAO extraída de iNurse.

```
public class SolicitudDAO extends InurseDAO {

//Constructor de la clase
public SolicitudDAO() {}

// Método para recuperar las solicitudes de un paciente a partir de su key
public List<SolicitudDTO> getSolicitudesByPacienteKey(Integer pacienteKey) throws  Exception
{
  // Construcción de la URL
  URL url = new URL(ConstantesDAO.SOLICITUDES_BY_PACI_KEY + "paciKey=" + pacienteKey);

  // Realizar petición HTTP
  JSONObject jsonObject = getHTTP(url);

  // Extraer resultados
  JSONArray jsonArrayUsuarios = jsonObject.getJSONArray("solicitudes");
  List<SolicitudDTO> listaSolicitudDTO = new ArrayList<>();

  for (int i = 0; i < jsonArrayUsuarios.length(); i++) {
     JSONObject jsonObjectUsuario = jsonArrayUsuarios.getJSONObject(i);
     Integer soliKey = Integer.parseInt(jsonObjectUsuario.get("soli_key").toString());
     String masoliDesc = jsonObjectUsuario.get("masoli_desc").toString();
     String soliDesc = jsonObjectUsuario.get("soli_desc").toString();
     // Lectura del resto de campos del JSON
     {...}

    // Añadir una nueva solicitud a la lista de resultados
    listaSolicitudDTO.add(new SolicitudDTO(soliKey, masoliDesc, soliDesc, maprioPrioridad,
    maprioDesc, matiCod, matiDesc, soliFecha, motivoRechazo));
  }

  // Devolver la lista de resultados
  return listaSolicitudDTO;
}
```

Ilustración 37. Fragmento SolicitudDAO

En primer lugar, podemos observar como la clase SolicitudDAO extiende de la clase padre InurseDAO, heredando de este modo sus métodos y atributos. Gracias a esta herencia, estamos evitando tener que volver a implementar determinados métodos y duplicar código de forma innecesaria.

En la siguiente tabla se muestran los DAO existentes en iNurse describiendo brevemente el cometido que desempeñan.

Nombre	Descripción
AvisosDAO	Recopila todas las llamadas que realizan operaciones sobre los avisos
InurseDAO	Clase padre de la que extienden el resto de objetos DAO de iNurse
MaestrosDAO	Todas las llamadas para recuperar algún maestro se realizan desde este DAO
MapaHospitalariDAO	Agrupa los servicios para recuperar información acerca del mapa de camas
MenusDAO	Encapsula las peticiones que recuperan, insertan o recuperan información acerca de los menús

PacienteDAO	Desde este DAO se realizan las peticiones para recuperar información sobre un determinado paciente
SolicitudDAO	Todas las operaciones que interactúan con las solicitudes se realizan desde este DAO

6.7. Scripts PHP

Como se ha comentado anteriormente, los scripts PHP son los responsables de realizar las consultas a base de datos y devolver la información solicitada por la aplicación. Los scripts pueden aceptar un conjunto de parámetros de entrada y como respuesta generan un objeto JSON, el cual será interpretado por iNurse para extraer la información solicitada. Algunos de estos scripts también son los responsables de enviar las peticiones a FCM para que determinados dispositivos reciban una notificación.

En la ilustración 38, podemos ver uno de los scripts PHP empleados por iNurse.

```
<?php
// ************** CONEXION A BASE DE DATOS **************
$hostname="localhost";
$username="root";
$password="";
$dbname="iNurse";
$usertable="menus_solicitados";

//Connect to the database
$db = mysqli_connect($hostname,$username,$password,$dbname);

if($db->connect_errno > 0){
        die('Unable to connect to database [' . $db->connect_error . ']');
    }

//*************** PARAMETROS ENTRADA **************
$codigo = $_GET['codigo'];
$paciKey = $_GET['paciKey'];

//*************** QUERY A EJECUTAR **************
$query = "SELECT m.menu_key FROM $usertable m
                WHERE m.menu_cod='$codigo'
                        AND m.menusoli_fecha=CURDATE()
                        AND m.paci_key=$paciKey";

// ************** LANZAR QUERY **************
$result = mysqli_query($db, $query);

// ************** RESULTADOS EN JSON **************
echo '{"'.$usertable.'":[';
$firstRow = true;
while($row = mysqli_fetch_assoc($result)) {

//Iteramos los resultados y generamos salida JSON
{...}

echo ']}';

// ************** CERRAR CONEXION **************
 mysqli_close($db);
?>
```

Ilustración 38. PrecargaMenu.php

Para que iNurse sea capaz de ofrecer todas las funcionalidades de las que dispone, ha sido necesario la implementación de un total de 20 scripts. Cada script cumple con un objetivo en concreto y le devuelve una determinada información a la aplicación. En la siguiente tabla, podemos ver de forma resumida cuales son los scripts implementados, los parámetros que pueden recibir y una breve descripción de su cometido.

Nombre	Parámetros	Descripción
autenticarUsuario	usuario password	Comprueba las credenciales de un usuario y password
getSolicitudesByPaciKey	pacienteKey	Obtiene todas las solicitudes de un paciente
getAllSolicitudes	-	Recupera las solicitudes de todos los pacientes
getMaestroSolicitudes	-	Recupera todos los maestros solicitudes activos
updateSolicitud	solicitudKey codigo	Cambia el estado de una solicitud. El nuevo estado es el código pasado como parámetro. Además, se encarga de enviar notificaciones a los usuarios a través de FCM cuando una solicitud es modificada
insertarSolicitud	pacienteKey maestroSoliKey descripcion	Crea una solicitud para un determinado paciente. Cuando una solicitud es creada, se envía una notificación a través de FCM a los enfermeros que corresponda
getMaestroPrioridades	-	Obtiene todos los maestro prioridades activos
getMaestroTipos	-	Recupera todos los maestro tipos activos
getAvisosByPaciKey	pacienteKey	Obtiene todos los avisos de un paciente
getMapaCamas	-	Devuelve la información necesaria sobre el mapa de camas
getMaxPlanta	-	Obtiene el número total de plantas del centro hospitalario
getDetallePaciente	pacienteKey	Recupera la información sobre un paciente
getAvisosConfiguracion	-	Recupera la información necesaria para la configuración de un aviso
insertarAviso	fechaInicio fechaFin horasRepeticion descripción pacienteKey	Inserta un nuevo aviso a un paciente. Además, manda una solicitud a FCM para que el paciente sea notificado del nuevo aviso.
getMenusByCodigo	codigo	Devuelve todos los menús con un determinado código
insertOrUpdateMenu	codigo pacienteKey menuKey	Inserta un menú a un determinado paciente y, en caso de que ya exista, lo actualiza
deleteMenu	codigo pacienteKey	Borra el menú de un paciente, que tenga el código recibido como

		parámetro y que, además, sea un menú creado durante el día actual
precargaMenu	codigo pacienteKey	Obtiene el menú escogido, durante el día actual, por un paciente con el código pasado como parámetro
getListMenus	codigo	Recupera la lista de menús solicitados por los pacientes del centro durante el día actual y que tengan el código pasado como parámetro
getAvisos	-	Recupera todos los avisos cuya fecha de finalización sea mayor o igual a la fecha actual

7. Manual de usuario

En este capítulo se van a mostrar los diseños finales de las interfaces de iNurse, mediante las cuales los usuarios pueden interactuar con la aplicación. Desde el principio, se ha buscado conseguir interfaces con un diseño intuitivo, amigable y fácil de usar, de forma que el usuario no requiera de ningún tipo de formación previa para poder hacer un correcto uso de iNurse. Las interfaces vendrán acompañadas de una explicación para comprender las funcionalidades que desempeñan, todo ello sin entrar en detalles técnicos ni de implementación.

7.1. Pantalla “Login”

La primera interfaz que nos encontramos al abrir iNurse por primera vez es la pantalla que nos permite autenticarnos (ver ilustración 39) y entrar en la aplicación. Dependiendo de las credenciales introducidas, la aplicación sabrá de qué tipo de usuario se trata y mostrará la interfaz correspondiente. Una vez el usuario se ha autenticado, se almacenan las credenciales de modo que, si no cerramos la sesión, la próxima vez que se abra la aplicación no tendremos que volver a iniciar sesión.

En el caso del personal sanitario, cuando sean dados de alta en la aplicación podrán elegir nombre de usuario y contraseña. Por otra parte, las credenciales de los pacientes se generarán automáticamente en el momento que sean hospitalizados con la finalidad de agilizar el acceso a iNurse. De forma que, el usuario de un paciente será su DNI (con letra) y la contraseña será un número de 4 dígitos generado aleatoriamente, el cual se le proporcionará en el momento de la hospitalización.

Ilustración 39. Pantalla “Login”

7.2. Interfaces de los pacientes

Si han sido introducidas las credenciales de un paciente ingresado, la aplicación comprobará consultando en la base de datos la validez de las credenciales y en el caso de validar el acceso mostrará el menú principal de los pacientes. Desde el menú principal (ver ilustración 40), el paciente puede realizar todas las gestiones que iNurse ofrece.

Ilustración 40. Pantalla "Menú principal pacientes"

7.2.1. Pantalla "Realizar solicitud"

La primera funcionalidad que va a ser descrita es la de realizar una solicitud, para ello el usuario debe pulsar sobre el botón "Realizar solicitud".

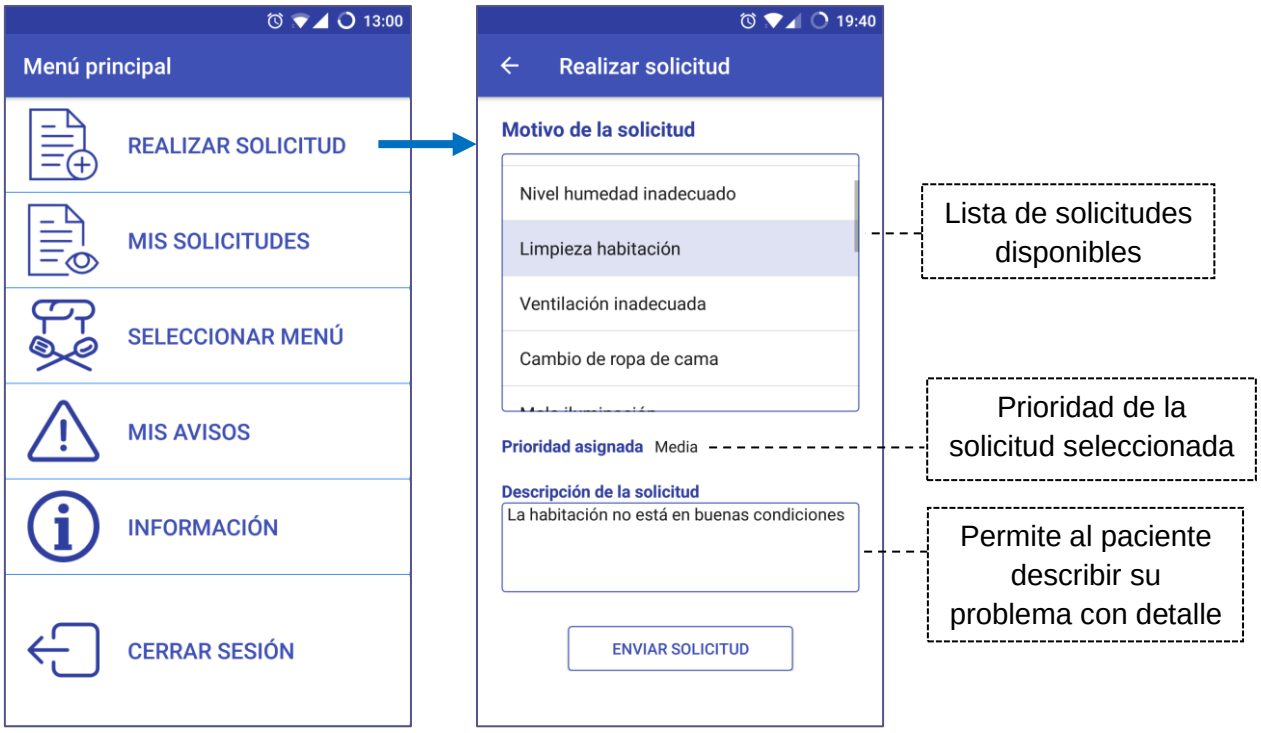

Ilustración 41. Pantalla "Realizar solicitud"

A través de esta interfaz (ver ilustración 41), el paciente puede realizar solicitudes. Las solicitudes vienen predefinidas por el hospital y según su importancia están etiquetadas con una prioridad, la cual no puede ser modificada. Dado que las solicitudes vienen predefinidas, se puede dar el caso que el paciente quiera solicitar algo que no se encuentra en dicha lista, por lo tanto, existe una entrada con la descripción "Otros" de forma que el paciente puede indicar en el campo de la descripción de la solicitud lo que desea.

7.2.2. Pantalla "Mis solicitudes"

Si el paciente desea ver las solicitudes que ha realizado y conocer en qué estado se encuentran, desde el menú principal debe acceder al apartado de "Mis solicitudes".

Ilustración 42. Pantalla "Mis solicitudes"

Esta interfaz (ver ilustración 42) mostrará toda la información relativa a las solicitudes realizadas. El paciente podrá ver que solicitudes ha realizado, la fecha de la solicitud, su prioridad, su descripción, su estado y en el caso de haber sido rechazada, aparecerá el motivo del rechazo.

Como se ha mencionado en apartados anteriores, los estados de una solicitud en iNurse pueden ser los siguientes: pendiente, en progreso, finalizada o rechazada. El *comboBox* situado en la parte superior de la interfaz nos permite seleccionar un estado para filtrar la lista de resultados.

7.2.3. Pantalla "Selección menú"

El paciente también tiene la posibilidad de elegir el menú deseado, entre los que el hospital tenga disponibles. Para ello, debe acceder a la interfaz de selección de menús (ver ilustración 43) pulsando el botón "Seleccionar menú" de la pantalla principal.

Ilustración 43. Pantalla "Selección menú"

Dependiendo de la hora del día, se le mostrarán al paciente los menús correspondientes al desayuno, comida, merienda o cena, siempre y cuando el hospital lo tenga configurado de ese modo. De los menús disponibles, el paciente únicamente podrá seleccionar un menú.

7.2.4. Pantalla "Mis avisos"

Por otra parte, el paciente puede visualizar todos los avisos los cuales le han sido programados. Para ello, debe pulsar sobre el botón "Mis avisos" y desde dicha interfaz (ver ilustración 44) podrá recordar todo lo que el personal sanitario le ha encomendado.

Ilustración 44. Pantalla "Mis avisos"

7.2.5. Pantalla "Información"

Por último, el paciente puede visualizar información general sobre el hospital desde la interfaz de "Información" (ver ilustración 45). En la primera versión de iNurse se ha decidido que se abra la página web del hospital en el que se utilice la aplicación. Sin embargo, está pensada para que en un futuro fuese personalizable y cada hospital pudiese mostrar la información que crea de interés para los pacientes.

Ilustración 45. Pantalla "Información"

7.3. Interfaces del enfermero

En el caso de que las credenciales introducidas en la pantalla de "Login" sean las de un miembro del personal sanitario, se mostrará el menú principal del personal sanitario el cual podemos ver en la ilustración 46.

Ilustración 46. Pantalla "Menú principal enfermero"

7.3.1. Pantalla “Ver solicitudes”

La primera funcionalidad disponible para los enfermeros es la de visualizar las solicitudes de los pacientes (ver ilustración 47), las cuales aparecen ordenadas por prioridad y pudiendo ser filtradas por su estado, y realizar acciones sobre ellas para cambiar su estado. Como se ha mencionado anteriormente, el flujo de acciones es el siguiente:

- Sobre las solicitudes cuyo estado es pendiente se pueden realizar dos acciones, empezar progreso o rechazar. Cuando una solicitud es rechazada, se debe especificar el motivo del rechazo.
- Si una solicitud está en progreso, puede ser finalizada o rechazada.
- Sobre las solicitudes rechazadas o finalizadas (ver ilustración 48) no se puede realizar ninguna acción.

Ilustración 47. Pantalla “Ver solicitudes” con filtro "Pendiente"

Ilustración 48. Pantalla "Ver solicitudes" con filtro "Finalizado"

Cuando una solicitud es rechazada, el enfermero debe indicarle al paciente el motivo por el cual su solicitud no puede ser llevada a cabo, tal y como podemos ver en la ilustración 49.

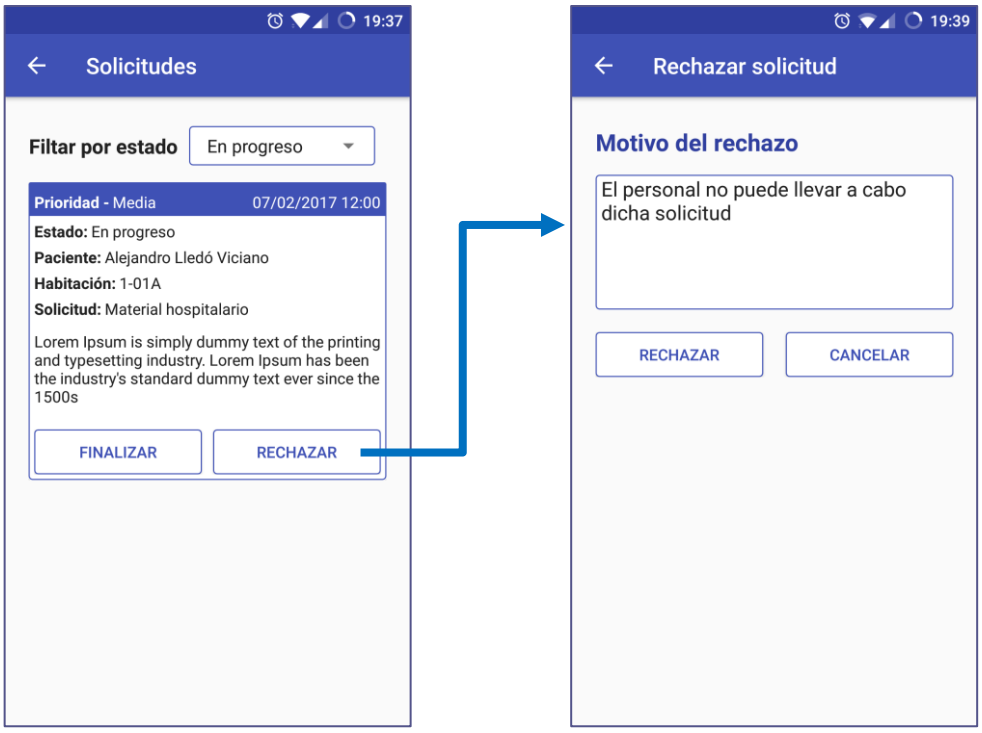

Ilustración 49. Pantalla "Rechazar solicitud"

7.3.2. Pantalla “Mapa de camas”

La siguiente funcionalidad nos permite visualizar la información básica sobre los pacientes hospitalizados, pudiendo filtrar los resultados por planta. Esto le permite al enfermero conocer en que habitación se encuentra un paciente de una forma rápida (ver ilustración 50).

Ilustración 50. Pantalla “Mapa de camas”

Además, pulsando sobre un ítem de la lista del mapa de camas, el enfermero puede consultar información detallada sobre el paciente (ver ilustración 51).

Ilustración 51. Pantalla "Información del paciente"

En cuanto a la imagen del paciente, iNurse permite la inclusión de la foto de un paciente de forma que el enfermero podría recordar con mayor facilidad de que paciente se trata. Sin embargo, es algo opcional y por ello, existen un conjunto de imágenes por defecto (ver ilustración 52) que se mostrarán dependiendo de la edad y sexo del paciente. Las imágenes disponibles son las siguientes:

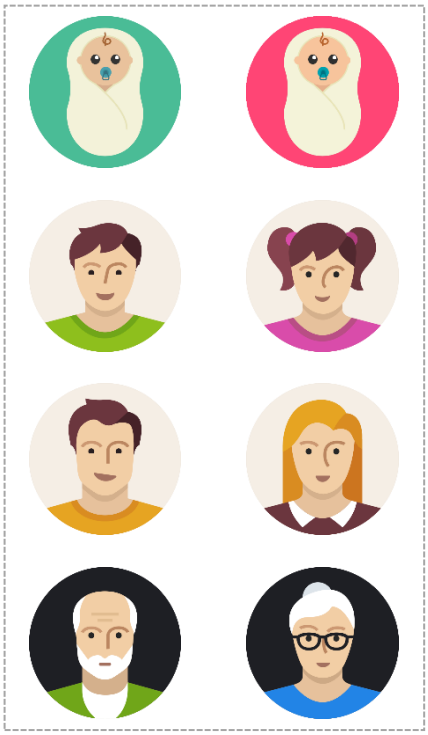

Ilustración 52. Imágenes por defecto de un paciente

7.3.3. Pantalla “Menús encargados”

Si el hospital ofrece variedad en los menús, desde esta pantalla (ver ilustración 53) los enfermeros pueden ver con antelación que menús han elegido los pacientes y poder hacer una mejor provisión. Debido a que hay desayuno, comida, merienda y cena, dependiendo de la hora del día la aplicación recupera los datos correspondientes. Además, se dispone de la posibilidad de filtrar los resultados por planta.

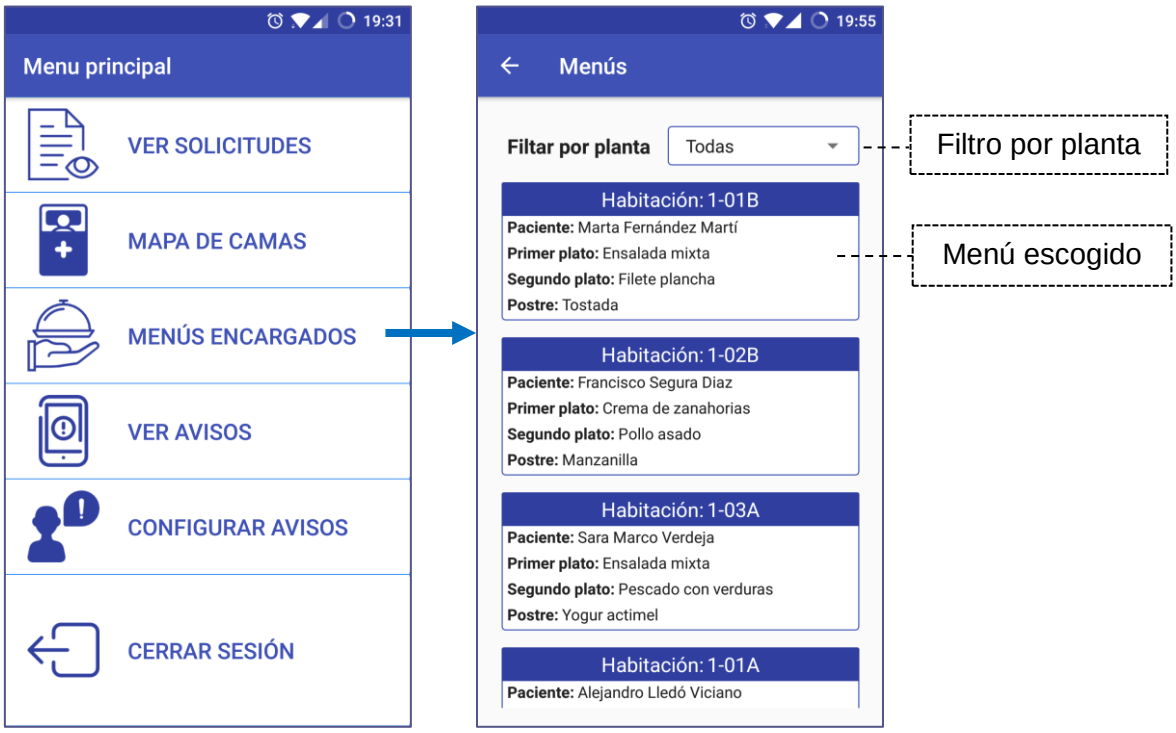

Ilustración 53. Pantalla “Menús encargados”

7.3.4. Pantalla “Ver avisos”

La pantalla mostrada en la ilustración 54, permite a los enfermeros visualizar los avisos que han sido programados a los pacientes hospitalizados. Solo se mostrarán los avisos activos, es decir, aquellos cuya fecha de finalización sea posterior al momento en el que se consulten los avisos. Además, se permite filtrar los resultados por planta.

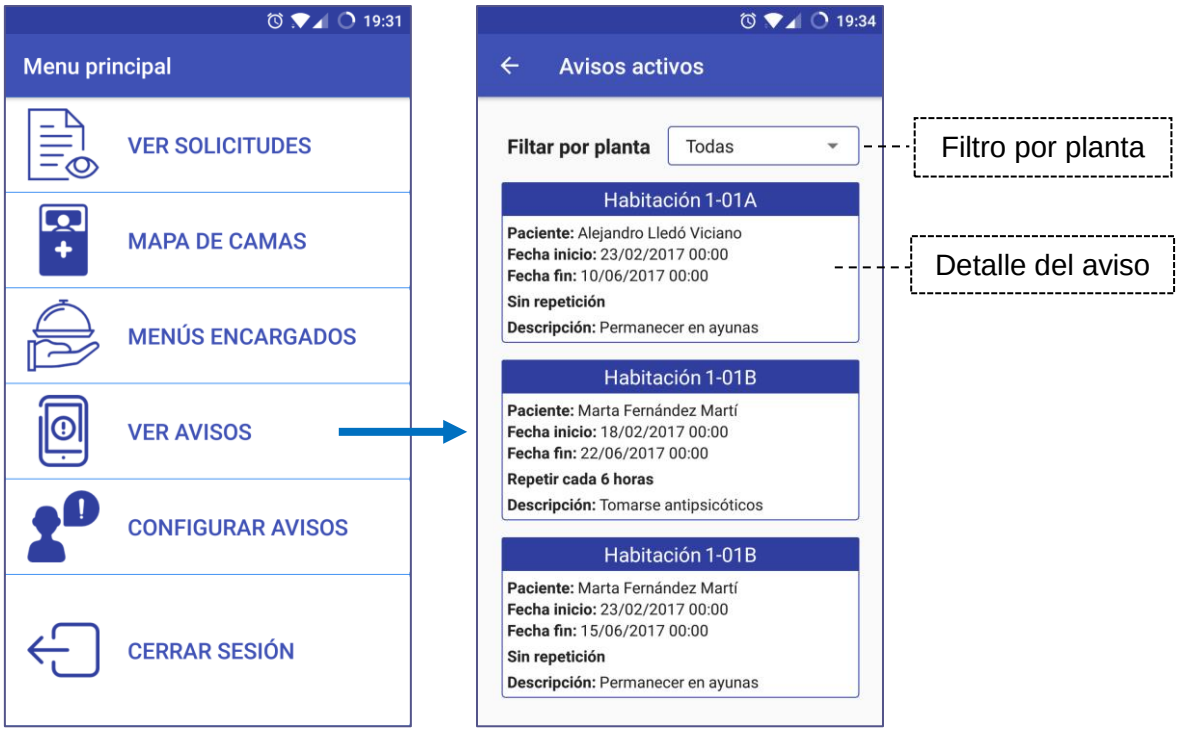

Ilustración 54. Pantalla "Ver avisos"

7.3.5. Pantalla "Configurar avisos"

Desde esta interfaz (ver ilustración 55) un enfermero puede programar un aviso a un determinado paciente. El enfermero deberá establecer la fecha de inicio y fin (ver ilustraciones 56 y 57), la descripción del aviso y si se debe repetir el aviso. En el caso de que el aviso sea recurrente, se debe especificar el intervalor en horas.

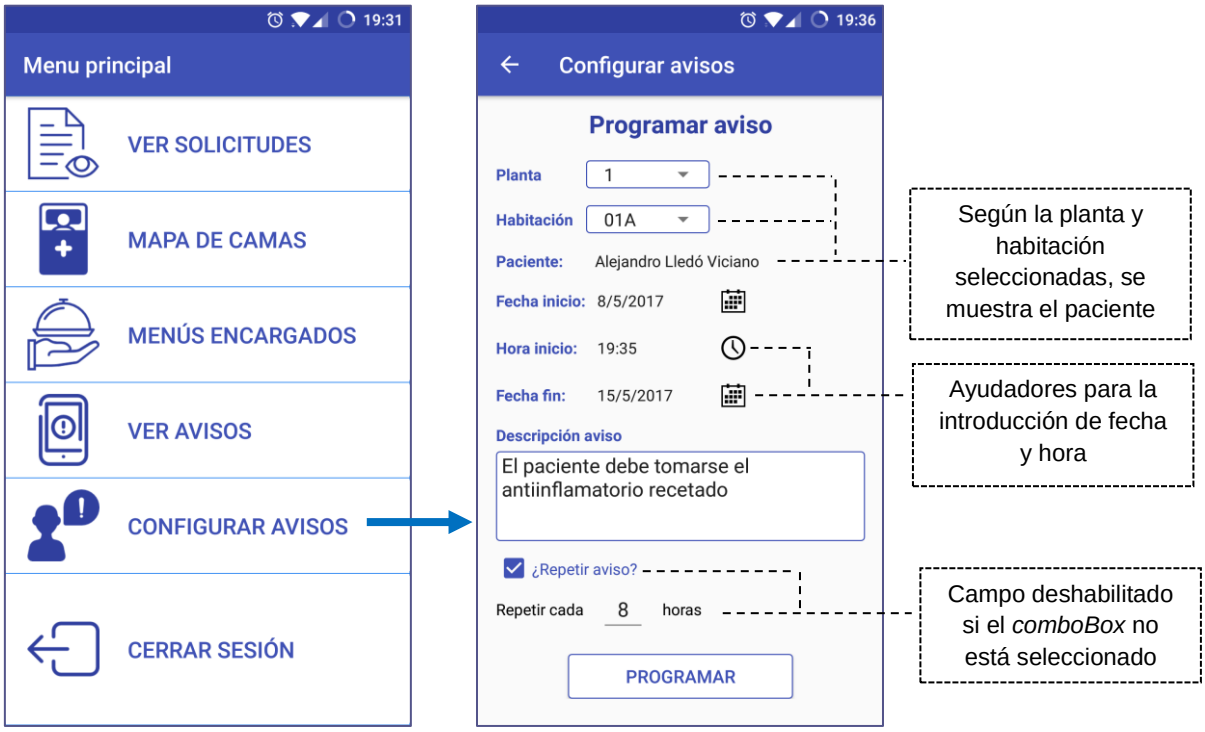

Ilustración 55. Pantalla "Configurar avisos"

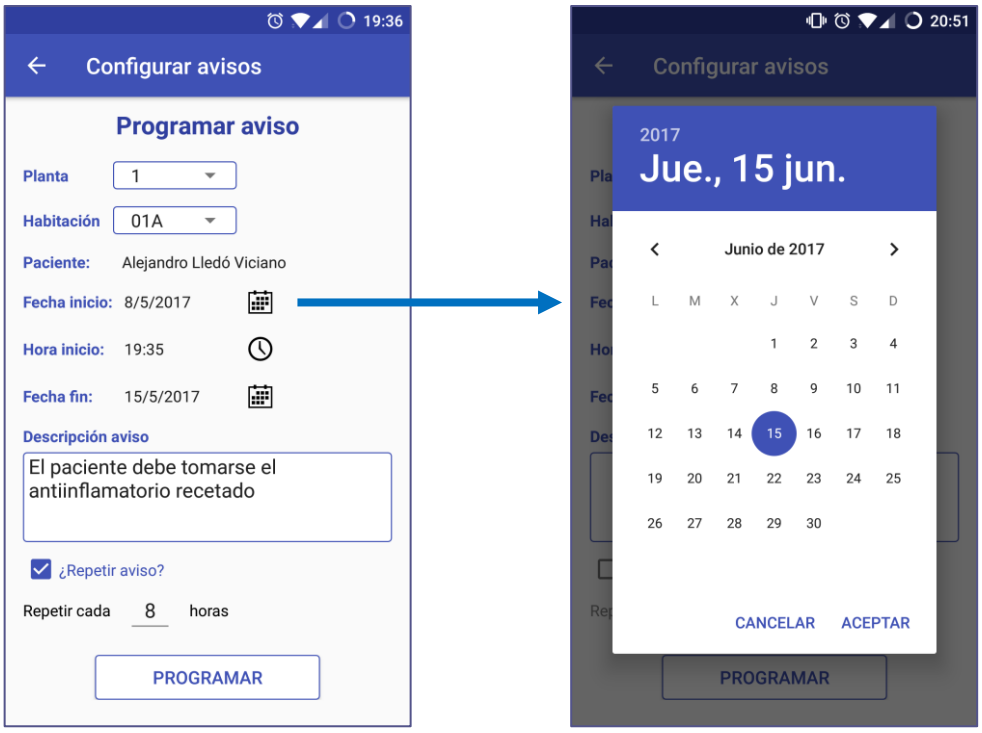

Ilustración 56. Ayudador de introducción de fecha

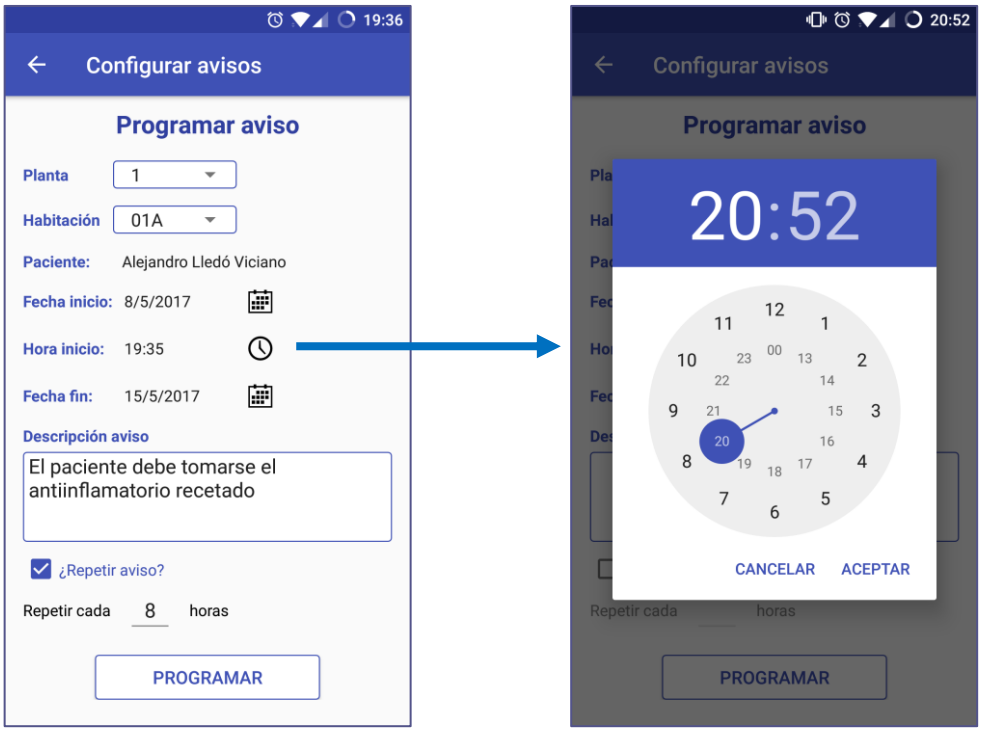

Ilustración 57. Ayudador de introducción de hora

7.4. Pantalla “Cerrar sesión”

Para cerrar la sesión y borrar las credenciales de acceso almacenadas en el dispositivo, tanto desde el menú de los pacientes como desde el menú de los enfermeros, basta con pulsar el botón “Cerrar sesión” y aceptar el mensaje de confirmación. En la ilustración 58 podemos observar cómo un usuario cerraría la sesión.

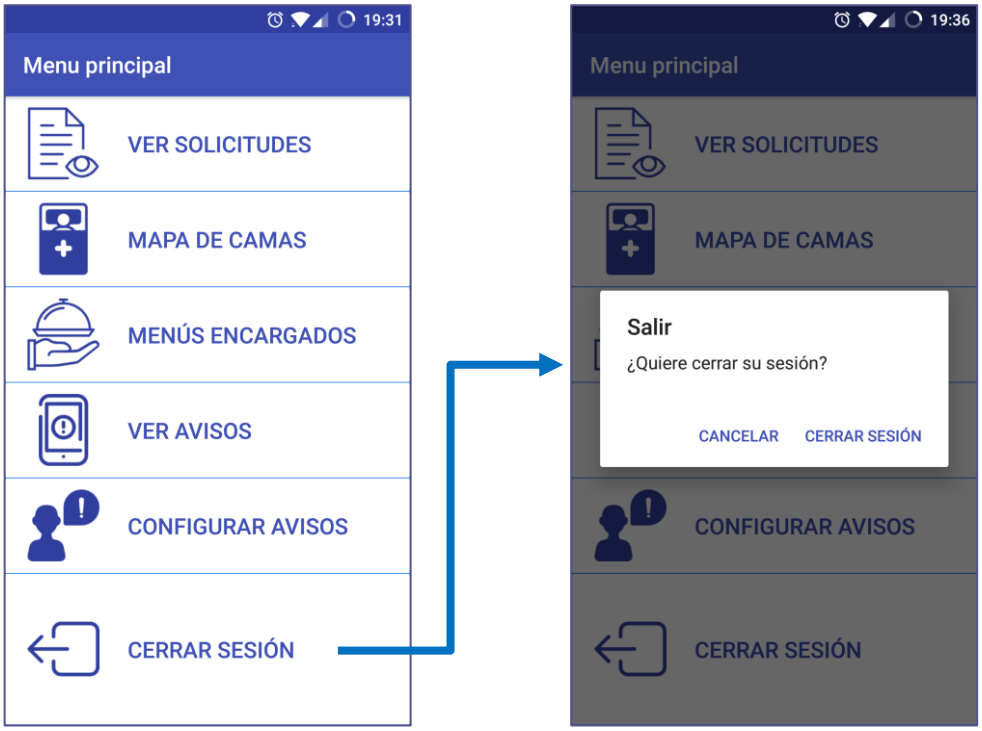

Ilustración 58. Pantalla “Cerrar sesión”

7.5. Notificaciones

Como se ha comentado anteriormente, la integración con Firebase Cloud Messaging permite que nuestro servidor envíe notificaciones *push* a iNurse, de modo que los usuarios de iNurse no tengan que entrar continuamente a la aplicación para saber si ha habido algún cambio de interés. Además, cuando se recibe la notificación y se pulsa sobre ella, se abre la pantalla de iNurse correspondiente en la cual podemos visualizar la información de dicha notificación con mayor detalle.

Los pacientes pueden recibir notificaciones en dos casos (ver ilustración 59), el primero de ellos es cuando una solicitud que ha realizado cambia de estado y el segundo caso es cuando se le ha programado un aviso.

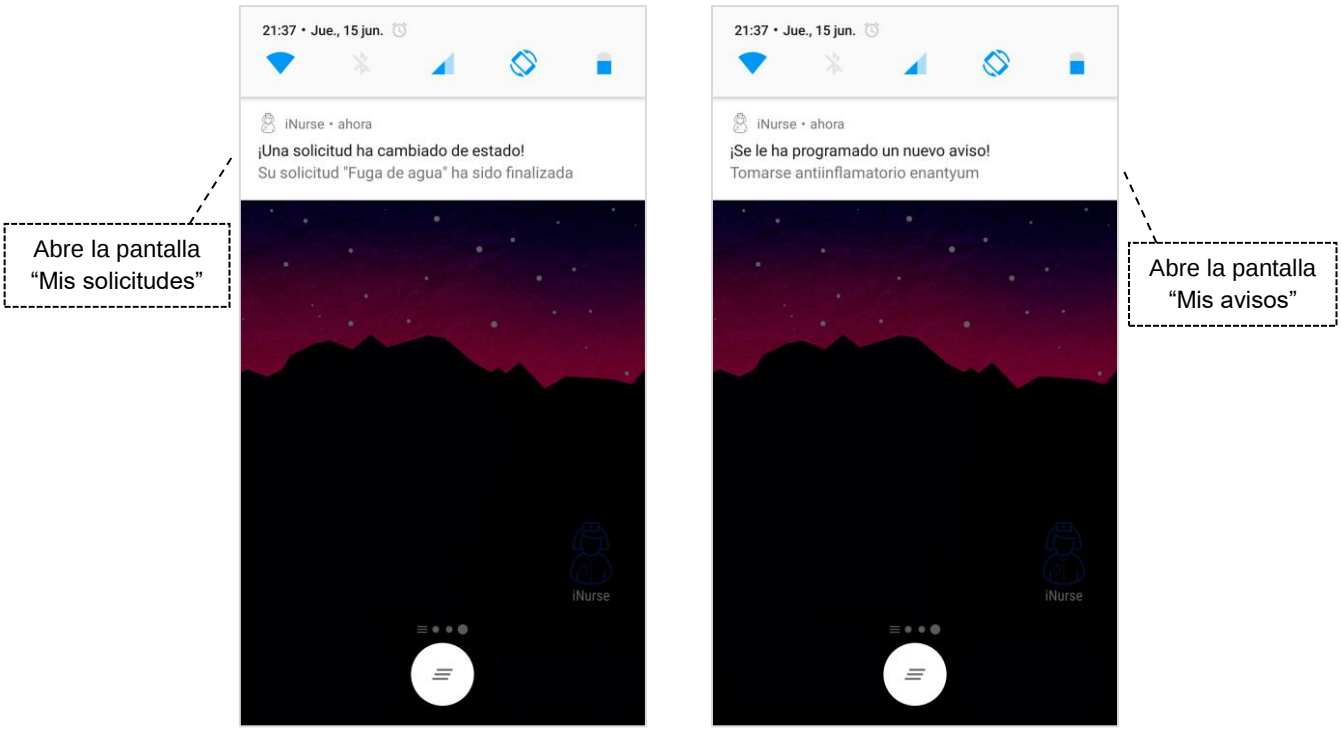

Ilustración 59. Notificaciones de los pacientes

Por otra parte, los enfermeros reciben notificaciones cada vez que el paciente realiza una solicitud, tal y como podemos ver en la ilustración 60.

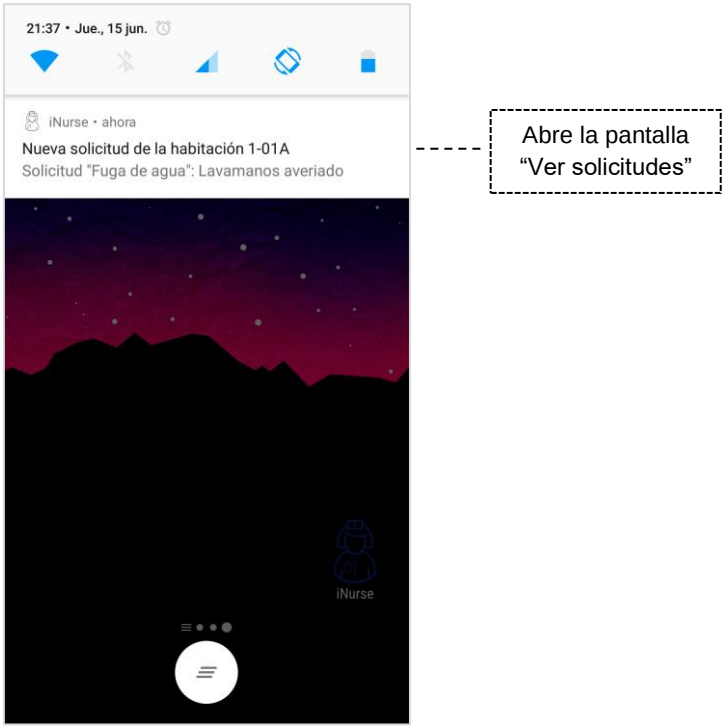

Ilustración 60. Notificaciones de los enfermeros

8. Escenarios de uso

8.1. Escenario de uso de un paciente

Andrea llega al hospital acompañado por su hijo Fernando, se dispone a ser ingresada debido a una intervención quirúrgica que tiene programada para ese mismo día. Después de un largo rato de espera, le llega su turno en el Servicio de Admisión donde facilita sus datos para realizar los trámites administrativos correspondientes. Una vez formalizado el ingreso, Andrea recibe el informe de ingreso hospitalario y es acompañada a su habitación. Una vez allí, la enfermera les transmite la información básica que deben conocer y, además les comenta que recientemente el hospital cuenta con una aplicación llamada iNurse con la cual pueden realizar de forma cómoda un conjunto de gestiones, como por ejemplo realizar una solicitud, y que disponen de las credenciales de acceso en el informe de ingreso. Como a Fernando le gusta curiosear mucho, se descarga la aplicación, se autentica con las credenciales de acceso de su madre (ver ilustración 61) y se pone a investigar que ofrece la aplicación.

Ilustración 61. Escenario de uso paciente - login

Media hora antes de la operación, Fernando recibe una notificación informándole que se le ha programado un aviso. Fernando pulsa sobre la notificación y iNurse se abre mostrándole la pantalla de “Mis avisos” (ver ilustración 62) donde puede ver de forma más detallada los preparativos que debe llevar a cabo su madre. Una vez llegado el momento, la enfermera entra en la habitación y se lleva a Andrea, quien ya estaba totalmente preparada, a quirófano para comenzar la intervención.

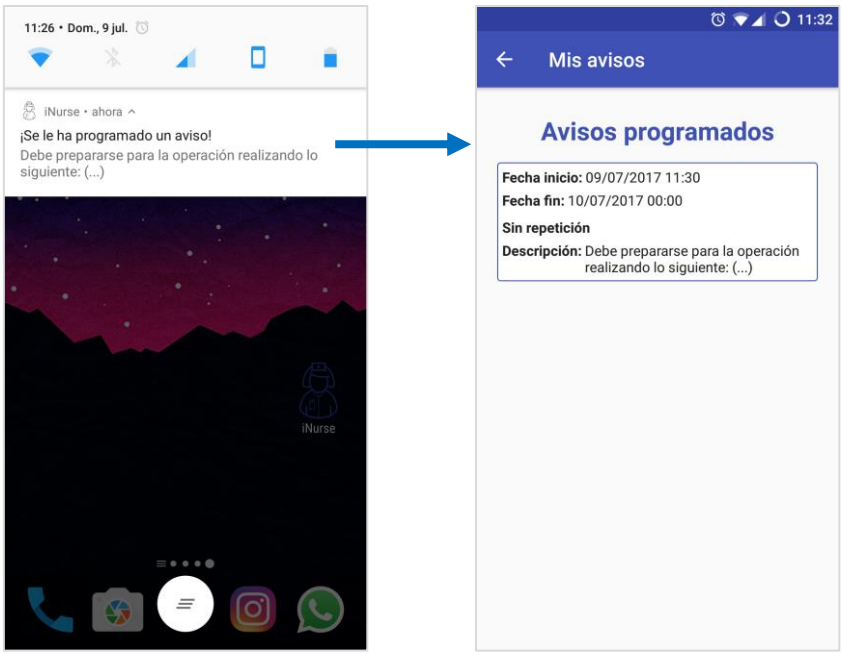

Ilustración 62. Escenario de uso paciente - recepción de aviso

Una vez finalizada la operación, Andrea es llevada de nuevo a la habitación junto a su hijo. Pasada una hora, Andrea comienza a notar que tiene fiebre y se lo dice a Fernando. Entonces, este decide usar iNurse para avisar las enfermeras. Desde la pantalla de "Realizar solicitud", Fernando busca en la lista de solicitudes predeterminadas la opción "Fiebre" y la selecciona. Luego, en el campo de descripción de la solicitud expone brevemente lo que le ocurre a su madre y pulsa el botón "Enviar solicitud". A continuación, accede a la pantalla "Mis solicitudes" donde comprueba que su incidencia ha quedado registrada. En la ilustración 63, podemos ver la navegación realizada por el usuario.

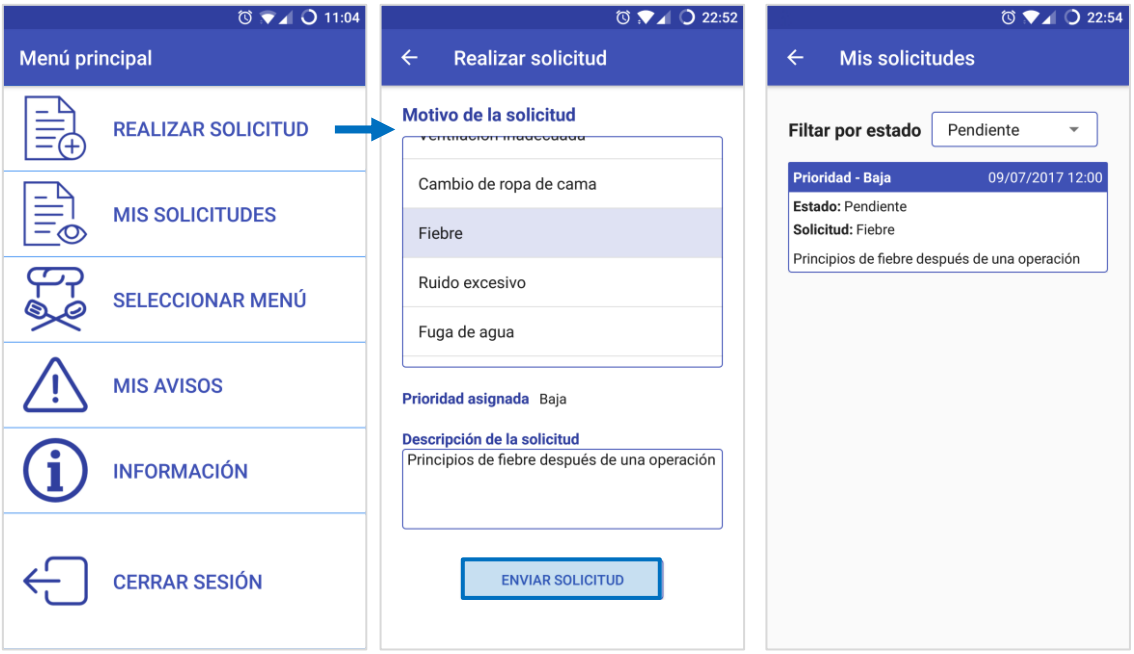

Ilustración 63. Escenario de uso paciente - realizar solicitud

A los pocos minutos, Fernando recibe una notificación en su dispositivo en la que se le informa que su solicitud ha cambiado de estado (ver ilustración 64), ahora su solicitud está en progreso.

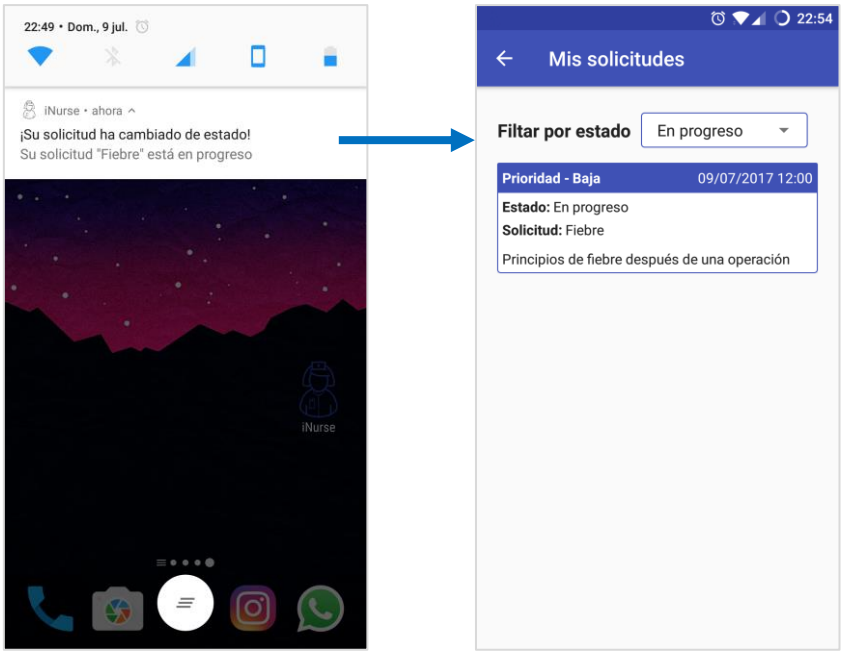

Ilustración 64. Escenario de uso paciente - solicitud en progreso

Poco después, aparece en la habitación la enfermera y le realiza una toma de temperatura a Andrea. Efectivamente, comprueba que tiene una fiebre leve y como la enfermera venía preparada por si realmente tenía fiebre, le suministra inmediatamente la medicación correspondiente. Una vez suministrada, la enfermera saca su dispositivo móvil, marca como finalizada la solicitud y les dice que si la fiebre continua, que vuelvan a enviar una solicitud indicando que la fiebre persiste. A los pocos segundos de irse la enfermera, Fernando recibe de nuevo una notificación (ver ilustración 65) indicando que la solicitud ha sido finalizada.

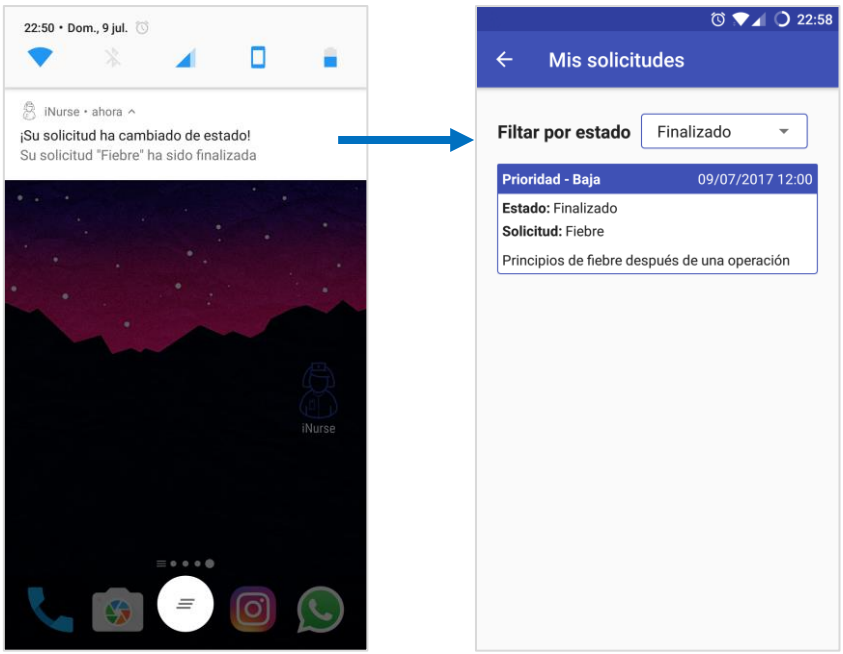

Ilustración 65. Escenario de uso paciente - solicitud finalizada

8.2. Escenario de uso de un enfermero

Un paciente ha sido ingresado recientemente y la enfermera de turno debe ir a la habitación a realizar unas tareas. La enfermera acaba de comenzar su turno y aún no conoce al paciente por lo que accede a iNurse y abre la pantalla “Mapa de camas”. Una vez abierto el mapa de camas, busca la habitación del paciente y pulsa sobre el ítem de la lista, correspondiente a dicha habitación, para obtener información detallada sobre el paciente (ver ilustración 66). Una vez hecho esto se dirige a la habitación a realizar su trabajo.

Ilustración 66. Escenario de uso enfermero - mapa de camas

Más tarde, la enfermera recibe una notificación indicándole que una habitación ha realizado una nueva solicitud y pulsa sobre ella para abrir la pantalla de “Ver solicitudes”. Una vez abierta, consulta los detalles de la notificación, informa a la persona que se debe responsabilizar de atender la solicitud y cambia su estado pulsando sobre el botón “Empezar”, tal y como podemos ver en la ilustración 67.

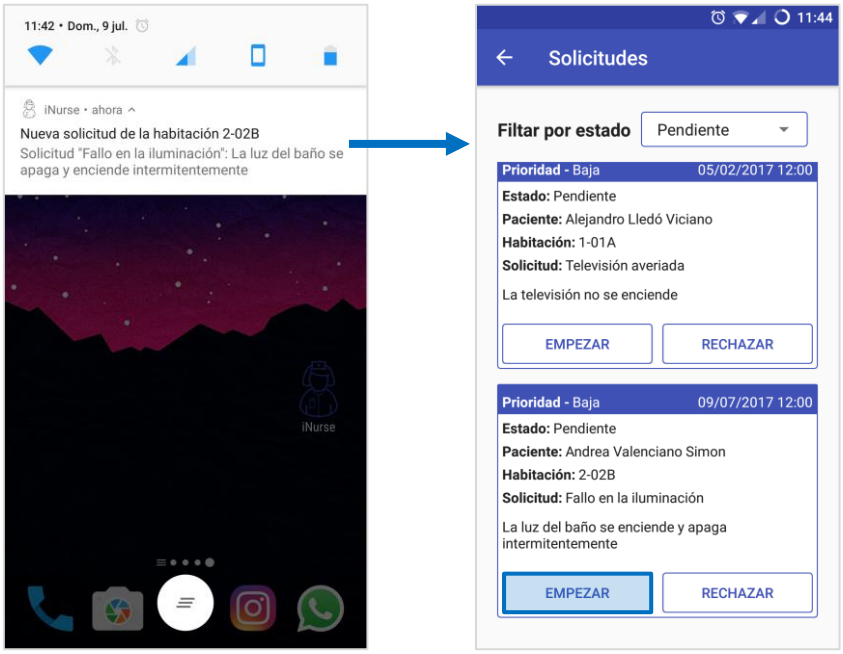

Ilustración 67. Escenario de uso enfermero – nueva solicitud

Una vez el responsable de atender la solicitud finaliza su tarea, informa a la enfermera y esta cambia el estado de la solicitud a finalizada pulsando sobre “Finalizar” (ver ilustración 68).

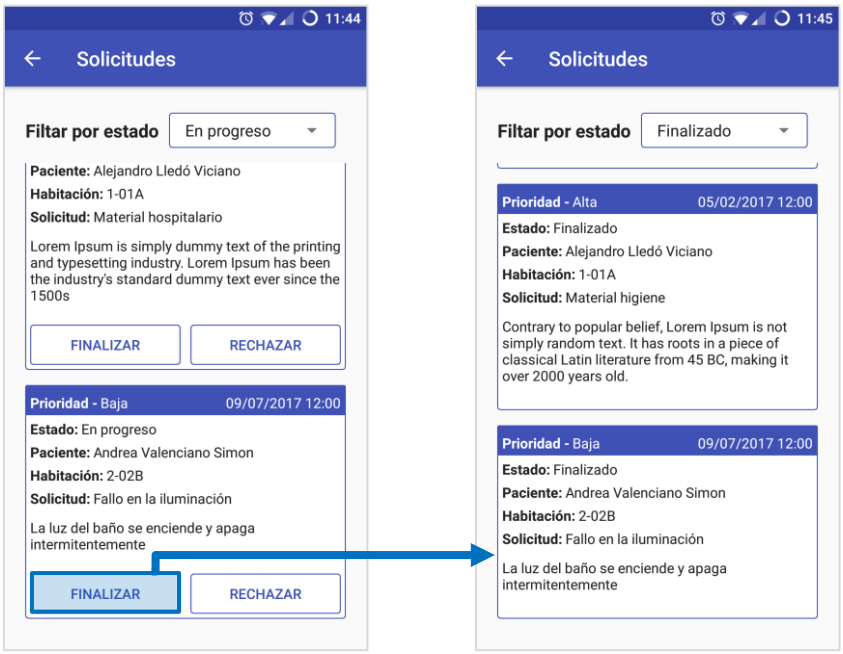

Ilustración 68. Escenario de uso enfermero - solicitud finalizada

Antes de finalizar su turno, debe configurar un aviso a un paciente. Para ello, entra en la pantalla de “Configurar avisos” (ver ilustración 69) y realiza las siguientes acciones:

1. Seleccionar la planta.
2. Seleccionar la habitación.
3. Indicar la fecha de inicio.
4. Introducir la hora de inicio.
5. Especificar la fecha de finalización.
6. Introducir una descripción que explique en que consiste el aviso.
7. Indicar la frecuencia de repetición del aviso, en el caso de que deba repetirse.
8. Pulsar sobre el botón “Programar”.

Una vez programado el aviso, la enfermera entra en la pantalla de “Ver avisos” para asegurarse de que ha quedado correctamente registrado.

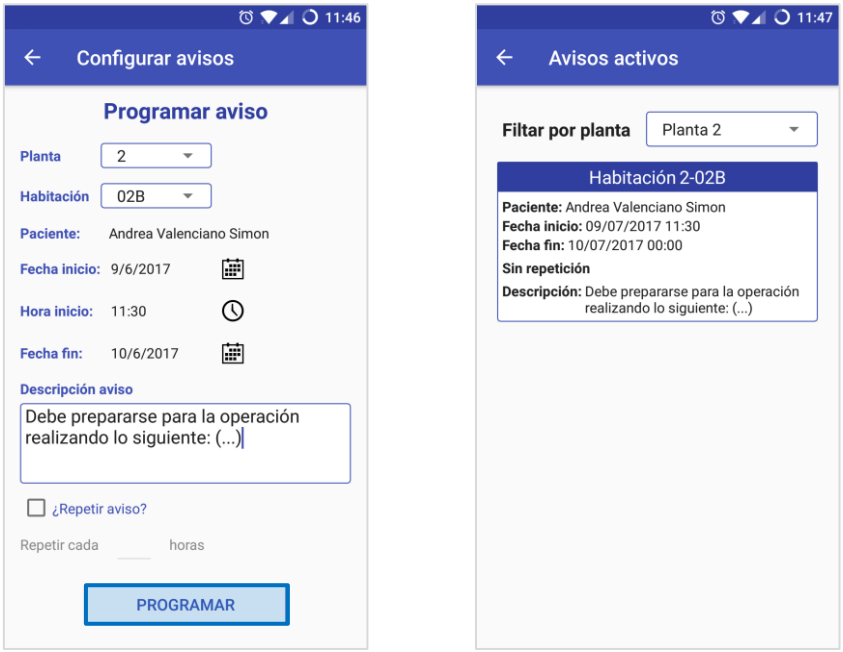

Ilustración 69. Escenario de uso enfermero - programación aviso

9. Trabajos futuros

A causa de las limitaciones de tiempo de entrega del proyecto, no ha sido posible el desarrollo de un conjunto de funcionalidades que harían de iNurse una aplicación mucho más completa y segura. Por esa razón, en este capítulo se plantearán una serie de funcionalidades cuyo desarrollo mejorarían la experiencia de usuario.

Disponibilidad en iOS

A pesar de que la gran mayoría de smartphones hacen uso del sistema operativo Android, también resulta necesario ofrecer nuestra aplicación a los usuarios que disponen de iOS. Estar únicamente disponible para Android resulta una gran desventaja frente a las aplicaciones analizadas durante el segundo capítulo, las cuales están disponibles para ambos sistemas operativos. Por ello, para aumentar las posibilidades de éxito de iNurse debería estar disponible mínimo para los dos sistemas operativos dominantes de hoy en día, de forma que pudiese ser utilizada por el máximo número de usuarios.

Seguridad SSL

Implementar un mecanismo de seguridad para proteger las comunicaciones de iNurse es estrictamente necesario. Como se ha comentado en anteriores apartados, asegurar la confidencialidad de la información sensible sobre las personas es imprescindible, ya que en el caso de no hacerlo podría infringirse la Ley Orgánica de Protección de Datos lo que conllevaría sanciones graves.

La solución propuesta anteriormente en este proyecto sería emplear SSL, un protocolo eficiente y seguro que permite comunicarse asegurando la privacidad de la información.

Suscripción a tópicos

Una funcionalidad muy interesante y beneficiosa para los pacientes sería la posibilidad de suscribirse a temas de interés. Cada hospital sería libre de configurar los tópicos que considerase necesarios de forma que el paciente podría suscribirse para recibir información. De este modo, cada vez que el hospital publicase información sobre un tópico, el paciente recibiría de forma instantánea una notificación avisándole de ello. Si el paciente lo desea, accedería a iNurse para ver en detalle la publicación del hospital. El paciente también debe ser capaz de cancelar sus suscripciones.

La infraestructura necesaria para implementar el sistema de notificaciones mediante suscripción ya está disponible gracias a Firebase Cloud Messaging, por lo que desarrollar está funcionalidad no supondría un gran coste de tiempo.

Integración con otro sistema de información

Realizar una integración con otro sistema de información ampliaría mucho las limitaciones de nuestra aplicación. Esto le permitiría a iNurse desarrollar una gran cantidad de funcionalidades nuevas y mostrar mucha más información a los usuarios.

Ejemplos de funcionalidades potenciales gracias a una integración podrían ser, entre muchas otras, las siguientes:

- Los pacientes podrían consultar desde iNurse los resultados de las pruebas que se le han realizado durante su estancia actual en el hospital e incluso tener la posibilidad de acceder a su historia clínica.
- El personal sanitario podría tener acceso a la historia clínica de los pacientes hospitalizados de forma que podrían consultar las alergias de un paciente desde la misma habitación, por ejemplo.

Como desarrollador de Orion Clinic, se intentó convencer al cliente de que permitiese realizar algún tipo de prueba de integración para la realización de este proyecto, sin embargo, la respuesta fue negativa por lo que esta posibilidad quedó totalmente descartada.

10. Conclusiones

En los inicios del proyecto, se partía con unos conocimientos básicos sobre las tecnologías empleadas, pero gracias al presente proyecto se ha adquirido conocimientos más avanzados sobre todas ellas.

En primero lugar comentar que, a pesar de que Android se desarrolla con el lenguaje de programación Java, existen un gran número de peculiaridades a la hora de desarrollar una aplicación para Android. Por una parte, la estructura del proyecto de una aplicación Android dista bastante del que puede tener una aplicación desarrollada para ser ejecutada sobre una máquina virtual estándar de Java y, por otra parte, la forma de trabajar con las interfaces de usuario es también totalmente diferente. Por lo que a pesar de estar habituado a la programación de aplicaciones Java, desarrollar iNurse ha supuesto realizar un cambio de mentalidad para adaptarse a la forma de trabajar de Android.

En lo que respecta al entorno de desarrollo la experiencia ha sido muy buena. Android Studio ofrece herramientas que facilitan mucho el trabajo del desarrollador. Podemos destacar su editor de diseño para las interfaces gráficas, su función Instant Run que ha permitido probar los cambios realizados durante el proceso de implementación con gran rapidez y el emulador del que dispone, que nos ha permitido ejecutar nuestra aplicación en cualquier versión de Android y cualquier tamaño de pantalla.

En cuanto al lenguaje de programación PHP, se ha tenido que dedicar tiempo adquirir los conocimientos necesarios para implementar la lógica que necesita nuestro servidor para establecer las comunicaciones con el cliente, ya que durante mis estudios no había cursado ninguna asignatura donde se hiciera uso de PHP. Sin embargo, el tiempo de aprendizaje ha sido bastante rápido y no ha supuesto grandes dificultades familiarizarse con dicho lenguaje. PHP ha demostrado ser un lenguaje muy potente y versátil, que sin duda volvería a utilizar en situaciones similares.

En lo que respecta a la integración con Firebase Cloud Messaging, ha sido un punto fundamental para la implementación de un sistema de notificaciones muy eficiente que permite mantener al paciente informado de forma instantánea. Al principio resultó algo complejo debido a las configuraciones que se deben realizar para que todo funcione correctamente, pero gracias a la excelente documentación que ofrece Google se solventaron las dificultades con éxito.

Finalmente, a nivel personal supone una gran satisfacción haber realizado con éxito una primera versión de la aplicación iNurse partiendo totalmente desde cero. Ha supuesto un reto interesante y mediante el cual he adquirido un buen conjunto de conocimientos que pueden ser muy útiles a lo largo de mi carrera profesional. Además, los conocimientos adquiridos, especialmente los referentes a Android pueden suponer una motivación para que en un futuro comience a desarrollar nuevas aplicaciones Android por mi propia cuenta.

Bibliografía

Ecma International. (2013). The JSON Data Interchange Format.

Everis. (2014, Abril 10). *Everis*. Retrieved from https://www.everis.com/spain/es/news/newsroom/el-programa-orion-clinic-desarrollado-por-everis-galardonado-por-su-excelencia-en-el

Everis. (2017). *ehCOS by Everis health*. Retrieved from http://www.ehcos.com/productos/my-hospital/

Gironés, J. T. (2016). *El Gran Libro de Android.* Marcombo.

González, A. (2016, Abril 24). *Ayuda ley protección de datos*. Retrieved from https://ayudaleyprotecciondatos.es/2016/04/24/datos-sanitarios-seguridad/

Google. (2016). *Android Developers*. Retrieved from https://developer.android.com/studio/features.html?hl=es

Google. (2016). *Firebase*. Retrieved from https://firebase.google.com/docs/cloud-messaging/

IDCSALUD, S.L. (2017, Junio 14). *iTunes*. Retrieved from https://itunes.apple.com/es/app/hospital-universitario-rey-juan-carlos/id1050079598

IMED Hospitales. (2017). *IMED Hospitales*. Retrieved from http://www.imedhospitales.com/es/pagina/app-iphone-imed/

lopd-proteccion-datos.com. (2017). Retrieved from http://www.lopd-proteccion-datos.com/ley-proteccion-datos.php

Sanitas. (2017, Junio 22). *Google Play*. Retrieved from https://play.google.com/store/apps/details?id=com.sanitas.misanitas&hl=es

Schildt, H. (2007). *Fundamentos de JAVA.* McGRAW-HILL INTERAMERICANA EDITORES.

Tangient LLC. (2011, Agosto 30). *Wikispaces*. Retrieved from https://packo.wikispaces.com/Caracteristicas+de+MYSQL