Universidad Zaragoza
1542

Trabajo de Fin de Grado

Generación e intercambio de imágenes médicas con perfil DICOM entre una aplicación diseñada en Android y una red social privada alojada en una instancia EC2.

Autor

Alejandro Alquézar Ibáñez

Director

José García Moros

Escuela de Ingeniería y Arquitectura / Universidad de Zaragoza

2017

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

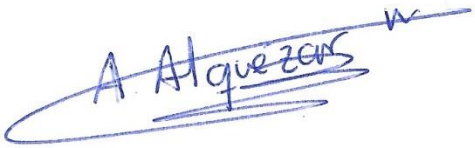
D./Dª. ALEJANDRO ALQUÉZAR IBÁÑEZ,

con nº de DNI 73158267-M en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Grado, (Título del Trabajo) Generación e intercambio de imágenes médicas con perfil DICOM entre una aplicación diseñada en Android y una red social privada alojada en una instancia EC2.,

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 19 de Junio de 2017



Fdo: Alejandro Alquézar Ibáñez

Agradecimientos

En primer lugar, dar las gracias a mis padres, por enseñarme a ser quien soy, y por su comprensión y apoyo durante estos años, al igual que a mi director de proyecto por guiarme para hacer de este proyecto, un trabajo menos complejo.

Quisiera incluir en este agradecimiento a todas y cada una de las personas que he conocido durante mis años en el grado, los cuales han hecho muy llevaderos estos años, aprendiendo de ellos una infinidad de valores y compartiendo momentos que no se pueden olvidar, muchas gracias.

También quisiera hacer una mención especial a Santiago, Daniel, Laura y María José, por ser ese apoyo que siempre se necesita y estar, tanto en los buenos, como en los malos momentos. Este trabajo es por vosotros.

Muchas gracias a todos.

Alejandro Alquézar Ibáñez

Resumen

A día de hoy, no se puede imaginar una vida sin tecnología a nuestro alrededor. Todo está interconectado y se hace uso de ella para facilitar la vida en la medida de lo posible. Cualquier campo imaginable, tiene tecnología aplicada a su entorno, y uno de ellos es la medicina, donde la tecnología ha hecho posibles grandes avances. Tanta es la implicación de la tecnología en la medicina, que es posible realizar intervenciones con ayuda de otros profesionales a distancia y en tiempo real, o como en el caso de este proyecto, es posible la transmisión de archivos y ficheros entre puntos muy distantes manteniendo la confidencialidad y autenticidad, en periodos muy breves de tiempo.

Es por ello que, en este trabajo se ha realizado una aplicación mediante la cual, a partir de una fotografía tomada desde un dispositivo móvil donde estará instalada dicha aplicación, se procede al encapsulado de la imagen y una información básica del paciente.

Para ello, utilizaremos el estándar DICOM para la compresión de imágenes, ya que permite la transmisión, tratamiento e impresión de archivos DICOM, es decir, de imágenes biomédicas con un informe del estudio realizado. El estándar DICOM (Digital Imaging and Communications in Medicine) nace en el año 1993, a partir de un rediseñado completo de la publicación normalizada No 33-1988 de ACR-NEMA, y pertenece al campo de la informática médica.

Posteriormente, la aplicación se conecta con una red social, alojada en la nube, a la cual podrá transferir estos archivos y publicarlos a modo de blog. Dicha red social ha sido instalada y configurada también para este trabajo.

Para la realización de la aplicación, se ha trabajado con Android Studio, mientras que, para la instalación de la red social en la nube, se ha utilizado un entorno Linux, donde se conecta a una instancia EC2 de Amazon a través de SSH.

Índice de contenidos

Índice de figuras

Listado de acrónimos

ACR – American Collegue of Radiology

APK – Android Application Package

AWS – Amazon Web Services

DICOM – Digital Imaging and Communications in Medicine

EC2 – Elastic Compute Cloud

IDE – Integrated Development Environment

IOS – IPhone Operative System

JPEG – Joint Photographic Experts Group

NEMA – National Electrical Manufacturers Association

PHP – Hypertext PreProcessor

RDS – Relational Database Service

RLE – Run Length Encoding

S3 – Simple Storage Service

SDK – Software Development Kit

SO – Sistema Operativo

SSH – Secure Shell

SSL – Secure Socket Layer

TCP/IP – Transmission Control Protocol / Internet Protocol

UID – Unique Identifier

VR – Value Representation

Capítulo 1 – Introducción

En este primer capítulo se presenta, tanto la motivación de este proyecto, como un breve resumen acerca de los principales objetivos y la organización de la memoria del mismo.

1.1 Motivación

Durante los últimos años, se puede apreciar una clara evolución en lo referente a la tecnología, y podemos notarlo en casi todos los campos que podamos imaginar. Uno de ellos es la medicina, cuyo término técnico, la telemedicina, hace referencia a la prestación de servicios médicos a distancia. Podemos citar dos ejemplos, uno tan sencillo como pudiera ser una discusión entre dos profesionales médicos a distancia, y otro algo más complejo, como pudiera ser el uso de tecnología en comunicaciones e informática para la realización de consultas, diagnósticos o cirugías a distancia y en tiempo real.

La telemedicina incluye tanto diagnóstico y tratamiento, como la educación médica. Actualmente, podemos encontrar que su uso se enfoca básicamente en dos áreas de trabajo: la práctica clínica y la educación sanitaria. Dentro de la práctica clínica, la telemedicina se centra principalmente en la telediagnósis, teleconsulta, monitorización remota y una de las principales, almacenamiento digital de datos o fichas médicas. Por otro lado, dentro del área educativa, destaca las clases a distancia desde centros médicos, haciendo uso de las técnicas de videoconferencia, sacando mayor provecho a los recursos educativos.

En este proyecto, trataremos de abordar el campo del telediagnóstico, desarrollando una red social privada donde podremos alojar imágenes de carácter médico para el diagnóstico por parte de profesionales que también utilicen dicha red social. Además, se podrá acceder a dicha red social desde una aplicación móvil que también generará las imágenes con formato médico .dcm basado en el estándar DICOM.

1.2 Objetivos

En este apartado, se ha realizado una clasificación de los objetivos que se van a tratar de abordar.

El primero de ellos, consiste en entender el estándar DICOM para la implementación del software necesario para la compresión de una imagen con formato JPEG y la generación de una imagen con formato DICOM. Para ello estudiaremos los puntos más importantes dentro del estándar completo de DICOM [1].

El siguiente objetivo, es la creación de una aplicación móvil, mediante la cual se puede realizar una fotografía y procesarla para su compresión y posterior generación de una imagen con formato médico. La aplicación será diseñada para el sistema operativo Android.

El último de los objetivos, será la implementación de un motor de red social al cual se podrá conectar desde la aplicación móvil o desde cualquier navegador instalado en un dispositivo con conexión a internet. También habrá que dotar a dicha red social de seguridad, mediante la instalación de un certificado de seguridad SSL.

1.3 Organización de la memoria

Una vez presentados los principales objetivos, este documento abarca los siguientes capítulos:

Capítulo 2 - Escenario de trabajo: En este capítulo se tratan de manera breve y clara los principales recursos que se han utilizado en el proyecto. Se describe de manera breve el motor de red social utilizado, una comparativa con otros motores de red social existentes, una breve descripción de la aplicación diseñada para Android y una explicación sobre el estándar en el que se basa para la compresión de imágenes.

Capítulo 3 – Implementación de Elgg alojada en Cloud: En este capítulo se describe de manera extensa y detallada todos los elementos que componen la red social creada mediante Elgg, así como los plugins utilizados y cualquier detalle relativo a este motor de red social.

Capítulo 4 – DicomApp: Desarrollo de la aplicación móvil: En este capítulo se abordan de manera completa todos y cada uno de los detalles de la aplicación móvil realizada para este proyecto, así como una descripción del software utilizado, remarcando los posibles conceptos y características importantes. De manera explicativa, se introducen también fragmentos de código en este apartado para explicar algunas de las funcionalidades de la aplicación.

Capítulo 5 – Conclusiones y líneas futuras: En este apartado se presenta una opinión sobre la experiencia del proyecto llevado a cabo, y una explicación de las posibles líneas futuras que pudieran surgir a partir del trabajo ya realizado aquí.

Anexos – Este apartado contiene los anexos necesarios donde se detallan algunos de los pasos más importantes para la configuración de la red social.

1.4 Distribución del tiempo

Como se puede apreciar en la figura 1.1, se muestra una tabla con las fechas aproximadas de inicio y fin de las distintas etapas de las que está formado este trabajo.

Actividad	Fecha de inicio	Fecha de fin	Duración (horas)
Aprendizaje	**01/09/2016**	**05/03/2017**	**145**
Análisis y diseño	**28/01/2017**	**10/03/2017**	**15**
Configuración e implementación	**20/02/2017**	**25/03/2017**	**65**
Comprobación del funcionamiento	**15/03/2017**	**16/04/2017**	**50**
Mejoras	**14/04/2017**	**22/05/2017**	**25**
Redacción de la memoria	**16/04/2017**	**30/06/2017**	**60**

Figura 1.1: Representación de la distribución del tiempo

La etapa de aprendizaje es la que más tiempo se ha invertido, ya que ha sido necesaria la comprensión del estándar DICOM, no en su totalidad, puesto que se trata de un estándar complejo y del cual no necesitamos comprender todas sus variantes, y, por otro lado, ha sido necesario profundizar en la programación en Android, ya que los conocimientos adquiridos durante el Grado de Telecomunicaciones no bastaban para satisfacer los requisitos que se necesitan. Posteriormente, la etapa de análisis y diseño, ha tenido una duración aproximada de unos 24 días donde se le ha dado forma a los requisitos necesarios en la red social, así como las características principales que necesita la aplicación móvil.

En último lugar, las etapas de mejoras y redacción de la memoria, se han realizado de forma paralela.

En la siguiente figura (Figura 1.2), se representa la distribución temporal del trabajo mediante un diagrama de Gantt.

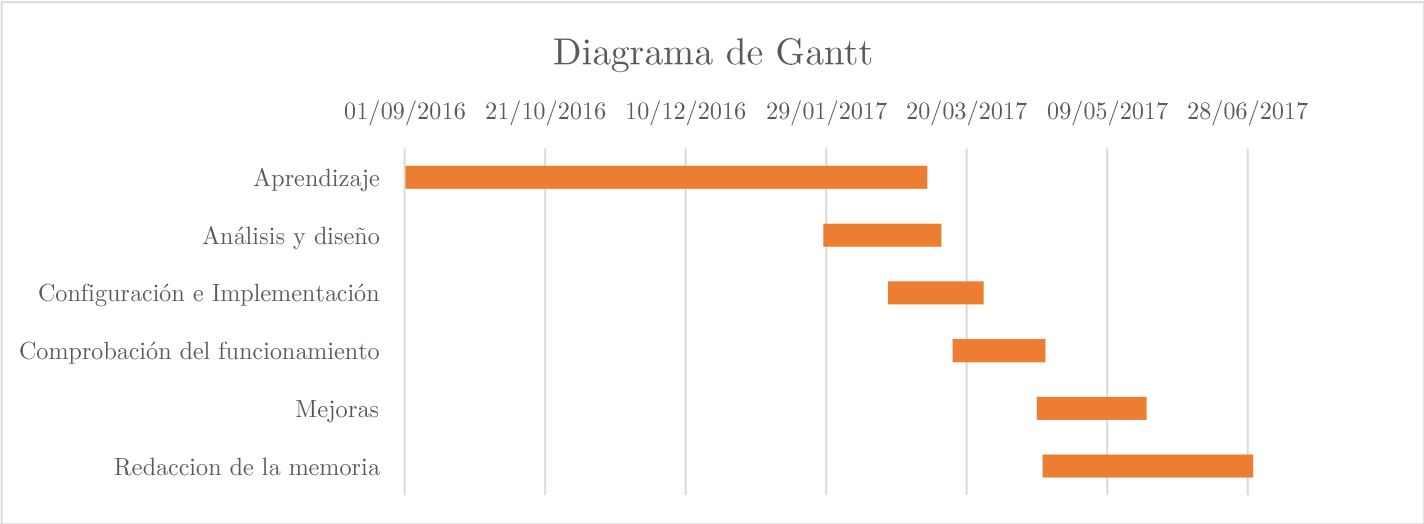


Figura 1.2: Diagrama de Gantt de la distribución temporal del trabajo.

Capítulo 2 – Escenario del trabajo

2.1 Red social: Elgg

Como sabemos, existe un gran abanico de posibilidades si queremos crear una red social propia. Entre este gran abanico, podemos diferenciar dos grandes grupos: redes sociales gratuitas, y redes sociales de pago.

Para la realización de este proyecto, se realiza un estudio sobre las redes sociales gratuitas, entre las que podemos destacar las siguientes: HumHub, BuddyPress y el motor de red social que utilizaremos, que es Elgg.

De manera muy breve, se resumen las principales características de estas redes sociales gratuitas:

En primer lugar, BuddyPress [3] (figura 2.1) es una extensión o plugin de WordPress. Es una característica que posibilita añadir una red social a la instalación ya existente. Podemos distinguir dos modos diferentes de funcionamiento de BuddyPress, el primero de ellos consiste en la posibilidad de crear una completa red social desde cero, y el otro se basa en la adición de características para ampliar una red de blogs ya creada anteriormente. Al igual que el resto de redes sociales que se citan en este proyecto, permite la creación de grupos y el intercambio de mensajes entre usuarios de manera pública y privada.

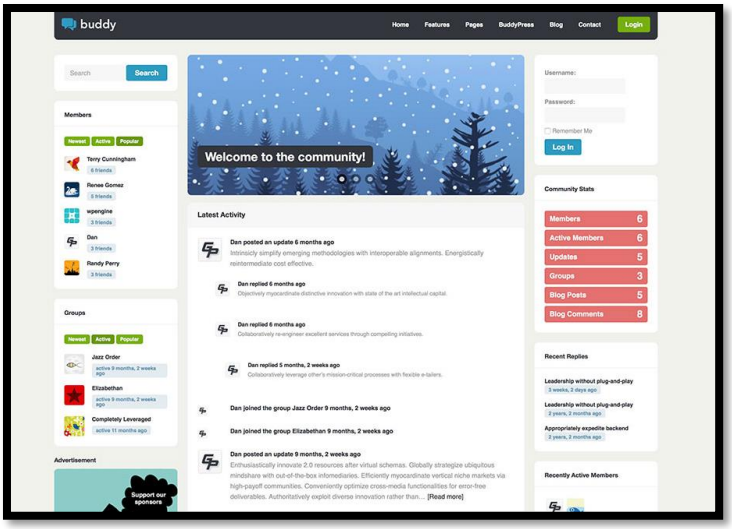


Figura 2.1: Vista general de una red social creada con la herramienta BuddyPress

Por otro lado, HumHub [4] (figura 2.2) se trata de una plataforma libre y de código abierto, desarrollado en PHP, compatible con temas y módulos que aumentan la funcionalidad del software para cumplir los requisitos necesarios. Puede ser utilizada para la comunicación y la colaboración interna, permitiendo el uso en diferentes tamaños, como puede ser desde unos pocos usuarios hasta grandes Intranets en empresas. También tiene otras características importantes como la posibilidad de habilitar notificaciones, gráficos del tráfico de actividad en la red, creación de grupos de usuarios y de directorios de archivos, búsqueda de usuarios de la red y de archivos de manera simple y sencilla, y una versión adaptada para dispositivos móviles.

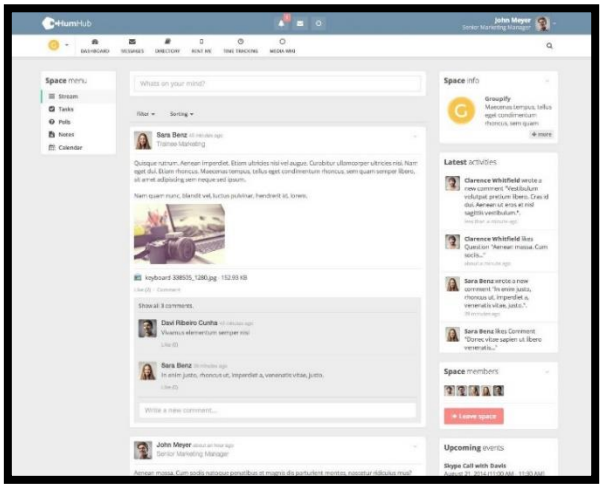


Figura 2.2: Vista general de una red social creada con HumHub

Para concluir con el repaso de las principales redes sociales gratuitas que se han encontrado, hablaremos de Elgg. Se trata de una herramienta de código abierto, mediante la cual podemos crear redes sociales con una complejidad relativamente baja. Elgg está diseñada para promover el aprendizaje a través del conocimiento compartido, aportando herramientas que facilitan las tareas con la idea de compartir información, como puede ser mediante blogs, contactos, grupos de trabajo…

Además de las características ya mencionadas, cuenta con una gran comunidad que desarrolla plugins y los comparte de manera libre para el resto de la comunidad. En los siguientes capítulos se detallarán las características, funcionalidades, y los plugins utilizados con Elgg.

Tras conocer las principales características de los motores de red social gratuitos, y ver que, en su mayoría, comparten muchas de sus funcionalidades y

características, en este proyecto se optará por utilizar Elgg, ya que se trata de una herramienta sencilla, mediante la cual podremos crear una red social nueva de forma simple y flexible.

Además, Elgg nos permite su completa personalización mediante temas y plugins para poder tener una red tan completa como se desee.

Para la implementación de la red social, utilizaremos un motor que ya existe como es ELGG [2]. Dado que ELGG ya es una red social desarrollada, nos permitirá instalarla sin necesidad de tener que llevar a cabo ningún esfuerzo para el desarrollo de su programación.

Aunque se trate de una red social ya programada, será necesario que realicemos un trabajo de instalación, configuración, y puesta en marcha para el correcto funcionamiento de la red.

El motor utilizado en este proyecto, ELGG, es una plataforma de código abierto, para el trabajo en red, recolección de noticias vía fuentes web e intercambio de archivos. Toda esta información puede ser compartida por los usuarios a través de los controles de acceso, y puede ser catalogada mediante etiquetas.

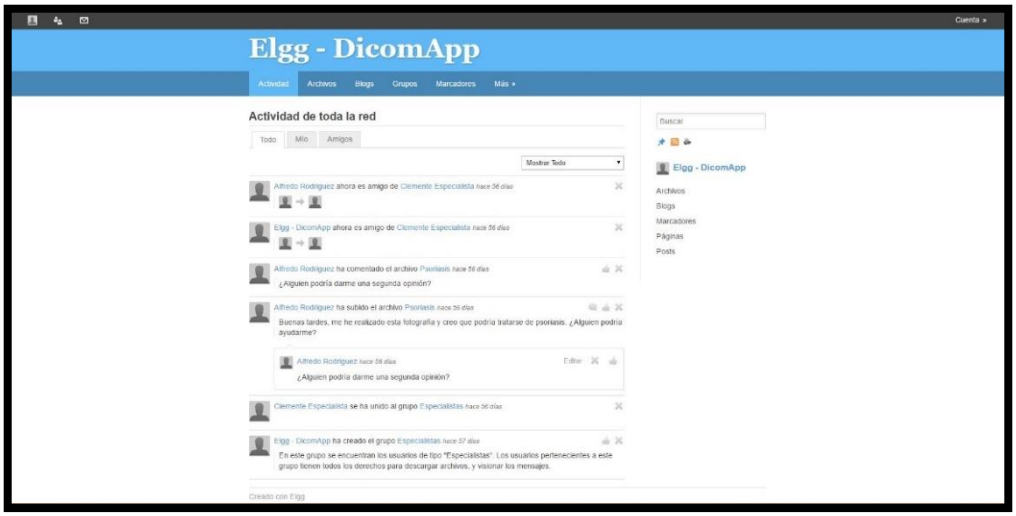


Figura 2.3: Vista general del panel de inicio de Elgg

Para un acceso remoto a la red social, será necesaria la instalación de la misma en un servidor. En nuestro caso, dicho servidor estará ubicado en la nube.

Se ha elegido ubicar el servidor en Amazon Web Services (AWS), ya que ofrece una capa gratuita de recursos que satisface las necesidades requeridas para este proyecto.

2.2 Aplicación Android: DicomApp

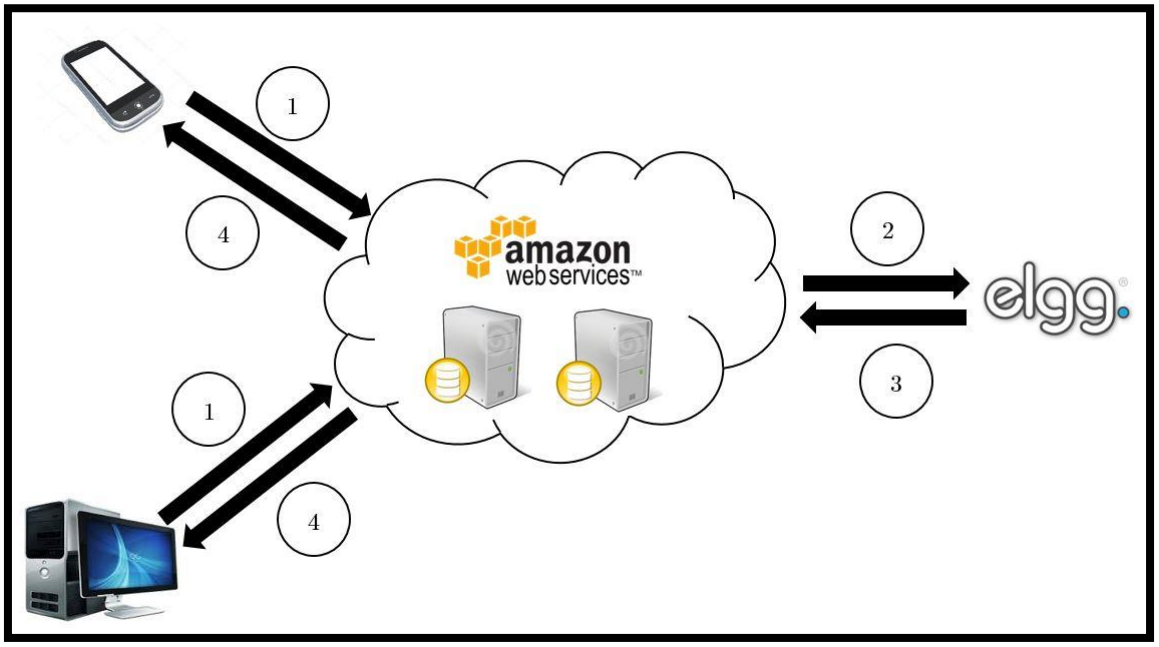


Figura 2.4 Esquema general del funcionamiento del proyecto

En la figura 2.4, se muestra un esquema del funcionamiento del trabajo. En los pasos 1 y 2, se conecta, el dispositivo móvil o el ordenador con la red social Elgg, a través de Amazon Web Services, lugar donde la red social está ubicada. En los pasos 3 y 4, una vez establecida la conexión, se envían los datos de la red social al dispositivo u ordenador para su posterior visualización.

Para dar un grado de movilidad al proyecto, se ha optado por la creación de una aplicación móvil, mediante la cual se pueda realizar una fotografía, subirla a la red social e interactuar por la red social.

De manera adicional, se ha decidido implementar un sencillo código en Java, mediante el cual a partir de una imagen en formato JPEG, se le añadirá una cabecera y se generará una imagen con formato *.dcm*, que podremos visualizar siempre y cuando tengamos un visor para imágenes con este formato.

2.3 Estándar DICOM

2.3.1 ¿Qué es DICOM?

Hoy en día, con el uso de las nuevas tecnologías, es muy común que nos encontremos con el uso de la informática en aplicaciones médicas, más concretamente en el campo de diagnóstico por imagen. Con estos avances, el uso de la imagen digital se ha terminado imponiendo, suponiendo una mejora en cuanto a calidad se refiere, y a la transmisión de dichas imágenes a cualquier punto de manera inmediata.

Este estándar, se ha desarrollado para encontrar las necesidades que fabricantes y usuarios tienen con el equipamiento de imagen médica para la interconexión de dispositivos entre sí y la transmisión de estos documentos y ficheros.

Un punto importante a remarcar, es que DICOM 3.0 es aplicable a toda la esfera de imágenes médicas, desde la transmisión hasta el tratamiento e impresión, independientemente de la especialidad médica que la exporte. DICOM es un estándar reconocido en todo el mundo utilizado en el intercambio de pruebas médicas, y su correspondiente manejo, visualización, almacenamiento, impresión y transmisión. Este estándar está formado por un formato de fichero, y por un protocolo de comunicación de red, que será TCP/IP.

2.3.2 ¿Por qué usar DICOM?

Podríamos preguntarnos, ¿Por qué queremos utilizar DICOM, si ya tenemos otros formatos como pueden ser JPEG o XML?

El principal motivo por el que usaremos el estándar DICOM, es que, gracias a este estándar, somos capaces de unificar en un mismo formato de imagen, una imagen que puede ser de una radiografía, por ejemplo, junto con información de dicho paciente, como podría ser su número de historial, información clínica del paciente…

Este tipo de ficheros consisten en una cabecera con campos estandarizados, y un cuerpo con datos de imagen. Además, cualquier objeto DICOM puede contener desde una imagen simple hasta una imagen con varios frames, que además estarán comprimidos usando una gran variedad de estándares, incluidos los más conocidos como JPEG, o RLE.

2.3.3 Estructura de datos

Si analizamos un fichero DICOM, podemos comprobar como dicho fichero, está compuesto por una cabecera, que almacena información sobre el nombre del paciente, el tipo de escáner, las dimensiones de la imagen… y toda la información relativa de la imagen, que puede ser, por ejemplo, información sobre una imagen en tres dimensiones.

Los primeros 794 bytes del fichero con extensión ".dcm" se utilizan para almacenar la cabecera DICOM, que contendrá la descripción de las dimensiones de la imagen así como otra información sobre el registro.

El tamaño de la cabecera varía en función de la información almacenada. Esta cabecera, requiere un preámbulo de 128 bytes, los cuales normalmente están puestos a 0.

Seguidamente, tenemos un prefijo, de 4 bytes, donde se sitúan los caracteres 'D', 'I', 'C', 'M'. Esta cadena debe estar codificada siempre con las letras en mayúscula. El propósito de este prefijo es permitir a las implementaciones diferenciar cuándo un fichero es DICOM o no.

A continuación, hallamos la información de la cabecera, que está organizada por grupos. Estos grupos, se utilizan para organizar la información dentro del fichero, y a su vez, está dividido en varios campos:

- **Número de grupo:** identifica al elemento dentro del fichero DICOM; pueden existir más de un elemento con el mismo valor.
- **Número de elemento:** identifica al elemento dentro del grupo.
- **Valor de Representación**: indica de qué forma está almacenado el contenido correspondiente al elemento.

Estos valores están predefinidos, y los podemos encontrar en el capítulo 6 del estándar DICOM.

- **Longitud:** Ya sea de 16 o 32 bits (dependiendo si el valor de representación es explícito o implícito) entero sin signo que contiene la longitud explícita del campo de Valor.
- **Valor:** Es el valor del elemento de datos, codificado según el campo VR y con la longitud que indica el campo Longitud del Valor, y que puede contener información vinculada al fichero como por ejemplo información del paciente, información sobre la imagen…

La información que se especifica en el campo Valor, es codificada teniendo en cuenta los siguientes criterios: Little endian y Big endian. cuando los datos son codificados en Little endian, siempre se tiene en cuenta el primer bit menos significante y viceversa en caso de ser big endian.

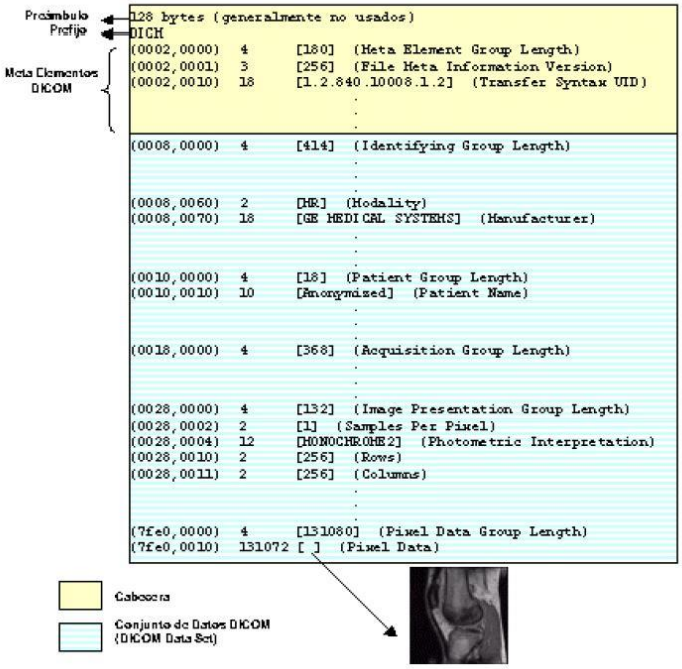


Figura 2.5: Esquema de un fichero DICOM

Como se aprecia en la figura 2.5, se muestra el esquema de un fichero DICOM, donde la primera sección, de color amarillo, representa la cabecera, que almacena información sobre el nombre del paciente, el tipo de escáner, las dimensiones de la imagen, etc. La siguiente sección, coloreada en azul, contiene toda la información relativa a la imagen, que puede ser, por ejemplo, información de una imagen en tres dimensiones.

Dentro de los campos de la cabecera, resulta de especial importancia el elemento 0002:0010, que define el "*Transfer Syntax Unique Identification*". Este valor hace referencia a la estructura de los datos de la imagen, indicando si la imagen ha sido comprimida. En la parte 5 del estándar de DICOM se tratan los temas de compresión de imagen.

Además del campo "*Transfer Syntax UID*", la imagen también se caracteriza por el **número de muestras por píxel** (0028:0002), la **interpretación fotométrica** (0028:0004) y el **número de bits reservados** (0028:0100).

Capítulo 3 – Implementación de Elgg alojada en cloud

En este capítulo se trata todo el contenido relacionado con la plataforma Elgg. Para la instalación de Elgg, es necesario un servidor instalado en la nube y encargado de alojar toda la plataforma web. Dado que se ha utilizado el motor de Elgg, no es necesaria la programación de dicha plataforma para su uso, lo que no implica que no haya que hacer nada, dado que, para su funcionamiento, es necesaria la instalación y configuración de todos sus componentes y plugins en el servidor de Amazon Web Services. Además, se han instalado y configurado los certificados SSL necesarios para realizar una conexión segura mediante el protocolo HTTPS.

Para este proyecto, el servidor ha sido instalado en una instancia de Amazon EC2, utilizando una capa gratuita que cubre las necesidades y requisitos para llevar a cabo este trabajo. Además de la instancia, ha sido necesaria la creación de una base de datos mediante Amazon RDS. A continuación, se describen brevemente los servicios utilizados de Amazon:

- Amazon EC2: Amazon Elastic Compute Cloud, es un servicio web que proporciona capacidad informática, permitiendo obtener un espacio de tamaño variable en la nube. La ventaja de usar este servicio es que permite pagar únicamente por la capacidad que se utiliza. Además, Amazon EC2 permite aumentar o disminuir la capacidad contratada en minutos.
  La sencilla interfaz de servicios web de Amazon EC2 proporciona un control completo sobre los recursos informáticos. Este servicio está diseñado para trabajar conjuntamente con Amazon S3 y Amazon RDS.
- Amazon S3: Amazon Simple Storage Service es un servicio que ofrece un almacenamiento de objetos sencillo, duradero y altamente escalable. S3 es un servicio muy útil y sencillo, ya que permite, mediante una sencilla interfaz de servicios web, almacenar y recuperar la cantidad de datos que desee desde cualquier ubicación de la red. Los clientes utilizan S3 como almacenamiento principal para el contenido generado por los usuarios. Además, al igual que Amazon EC2, S3 permite pagar por la capacidad utilizada y añade la posibilidad de activar notificaciones y aletas.

- Amazon RDS: Amazon Relational Database Service es un servicio administrado de base de datos relacional que pone a su disposición seis motores de base de datos conocidos como son: *Amazon Aurora, MySQL, MariaDB, Oracle, Microsoft SQL Server y Postgre SQL.*
  Con Amazon RDS, es sencillo configurar, utilizar y escalar una base de datos relacional en la nube. Proporciona capacidad rentable y de tamaño modificable y, al mismo tiempo, administra las tareas de administración de la base de datos.

3.1 Red social Elgg: configuración e instalación.

Elgg se define como una herramienta open source, la cual permite crear redes sociales sin necesidad de programar la plataforma completa. De igual manera, se pueden programar tantos plugins como se desee para su incorporación en la red social.

Aunque Elgg es una herramienta enfocada al aprendizaje, en el contexto de este proyecto se utiliza como plataforma donde compartir archivos con expertos de un campo concreto del conocimiento como es la medicina.

El uso de esta red social, aporta numerosas características como la gestión avanzada de usuarios, el soporte de varios idiomas, una versión disponible para dispositivos móviles, además de la posibilidad de añadir plugins ya desarrollados o desarrollados por uno mismo, ya que el código es abierto.

En la figura 3.1, se muestra un esquema del funcionamiento de un servidor privado alojado en una instancia de AWS. Mediante un dispositivo móvil o PC, se conecta a través de un navegador a una dirección IP pública que es la proporcionada por la instancia EC2 de Amazon. El servidor, está instalado en la instancia EC2.

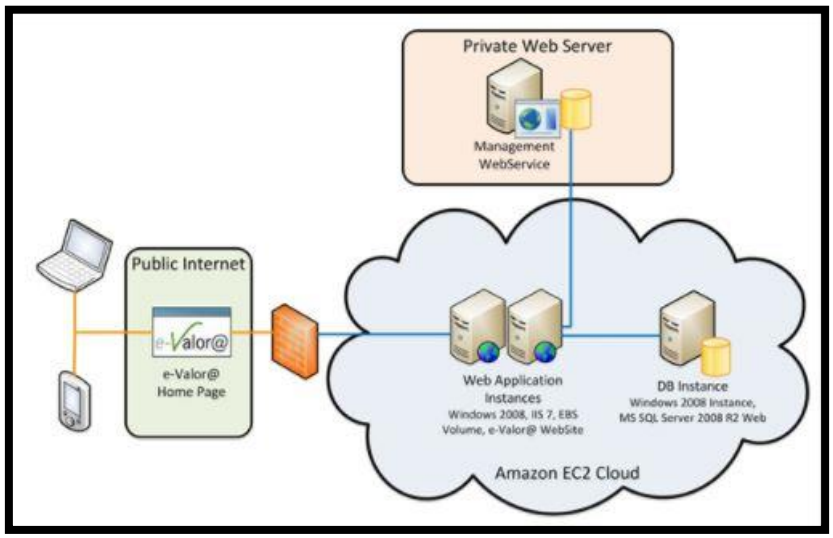


Figura 3.1: Esquema general del funcionamiento de AWS

Para la configuración del servidor de Amazon, se ha accedido a la instancia mediante una conexión SSH a través de una máquina virtual instalada en el PC.

Para iniciar la conexión SSH, debemos introducir la siguiente línea de código en la consola de la máquina virtual:

```
ssh -i "t2microTFG.pem" ubuntu@ec2-52-57-17-241.eu-central-1.compute.amazonaws.com
```

3.2 Plugins

En este capítulo se muestra una breve introducción de los archivos que forman parte de un plugin instalado en Elgg, y posteriormente los plugins que contendrá la red social creada para este proyecto.

En primer lugar, un plugin es una aplicación, o programa informático, que se relaciona con otra para agregarle una función nueva y de carácter específico.

Los principales archivos de los que un plugin está compuesto son:

- *start.php*: En este fichero se deben indicar tanto los eventos como el código que se ejecutará cuando el plugin esté operativo.
- *manifest.xml:* Se incluye información sobre el plugin, requisitos que necesita para funcionar y alguna información opcional. Además, podemos distinguir varios campos dentro de un fichero *manifest.xml*:
    - *id*: Nombre de la carpeta que usa el plugin.
    - *Name*: Nombre con el que el plugin va a ser visualizado.
    - *author:* Nombre del autor que ha desarrollado el plugin.

    - *description:* Descripción general acerca del plugin, información importante y características principales del mismo.
    - *version:* Versión actual del plugin.
    - *requires:* Requisitos de la versión de Elgg para la que el plugin ha sido desarrollado.
- *activate.php* y *deactivate.php:* Contienen el código que será ejecutado cuando se active o se desactive el plugin.

Dentro del plugin también se encuentran los ficheros *changes.txt*, donde se puede encontrar información acerca de los cambios producidos en cada versión del plugin, *install.txt*, donde se localiza información de carácter importante para la instalación del plugin, *readme.txt*, el cual contiene información adicional sobre el plugin en cuestión, *license.txt*, que proporciona información acerca de la licencia, y *copyright.txt*, que ofrece una descripción acerca del copyright del plugin.

Además, Elgg tiene una jerarquía de directorios la cual debemos respetar. Los directorios que podemos utilizar son los siguientes:

- **Classes:** En este directorio *classes/* se encuentran todas las clases que han sido programadas para dicho plugin. Todas las clases que se encuentren en este directorio serán ejecutadas de forma automática.
- **Languages:** En el directorio *languages/* se especifica un fichero por cada idioma disponible. Esto nos permite que la red social llegue a una mayor cantidad de usuarios.
- **Views:** En esta carpeta se encuentran todos los diseños de configuración de las páginas de Elgg, así como su presentación. El formato de los archivos contenidos en esta carpeta es PHP. Además, existen varios niveles dentro de esta carpeta, que permiten que Elgg cree las vistas correctamente.
- **Lib:** En este directorio, se debe introducir el código que se haya creado para el correcto funcionamiento del plugin. El código que se encuentra en esta carpeta no se carga automáticamente, ha de ser cargado configurando anteriormente el fichero *start.php*. El código de esta carpeta también puede ser utilizado por otros plugins, mediante el uso del comando *elgg_load_library*.
- **Pages:** Directorio donde se encuentra el contenido de las páginas que los usuarios después visualizarán.
- **Graphics:** En el directorio *graphics/* se encuentran todas las imágenes y gráficos que compongan el plugin.
- **Javascript:** Aquí se sitúan todos los ficheros javascript, dentro del directorio *js/*.

En el caso de Elgg, existen unos plugins ya activados por defecto, mientras que existe otra colección de plugins que, de manera sencilla, pueden ser añadidos a la red:

- **Blog:** Plugin que habilita un blog para todos los usuarios del sistema. Dentro del blog, el usuario puede publicar entradas de blog, así como ver los blogs de sus usuarios añadidos como amigos.
- **Bookmarks:** Permite a los usuarios marcar como favorito, tanto información, como entradas de blog, grupos, etc. Se puede visualizar en la pantalla cuando un usuario marca algún contenido como favorito.
- **Categorías:** Funcionalidad para hacer una clasificación del contenido de la red.
- **Custom índex:** Plugin para personalizar la página de inicio de cada usuario de Elgg, permite agregar widgets o módulos que indican la información reciente.
- **Files:** Plugin para subir archivos a la red, permitiendo la subida de imágenes, archivos en pdf, y para el caso en el que se está trabajando, archivos con formato .dcm
- **Friends:** Provee un widget en el que se muestran los amigos en la página de inicio del usuario.
- **Perfil:** Permite crear y administrar los perfiles de usuario, configurando la visibilidad de los campos. Se trata de un plugin muy básico en el que se guardan los campos comunes, como es el nombre, email, información personal, y foto de perfil.
- **Mensajes:** Permite enviar mensajes entre los usuarios. En la página de perfil del usuario, aparece un enlace, para poder enviarle un mensaje de forma privada. El plugin también provee una *bandeja de entrada* y *elementos enviados.*
- **Páginas:** Permite la creación de páginas similares a *wiki*. Estas páginas pueden ser editadas por los usuarios a los que se les permita el acceso necesario.

En la figura 3.2, se muestra el aspecto que tiene dentro de la red, el perfil de un usuario.

Figura 3.2: Vista del perfil de un usuario de Elgg.

La información mostrada en la figura está obtenida desde el perfil de administrador de la red social. Esta misma información puede ser visualizada tanto por el propio usuario, como por los usuarios con los que está conectado. Elgg permite que cada usuario determine los límites de privacidad de sus contenidos publicados en 3 categorías diferentes: *público, sólo para mis contactos, privado (sólo para mi).*

Capítulo 4 – DicomApp: Desarrollo de la aplicación móvil

En el mundo en el que vivimos, la evolución de la tecnología es descomunal. Hoy en día, prácticamente cualquier persona dispone de un dispositivo móvil con aplicaciones cuyo objetivo es facilitar la vida de las personas. Estas aplicaciones, son desarrolladas para diversos sistemas operativos, donde se encuentra el S.O utilizado para el desarrollo de la aplicación móvil utilizada en este proyecto.

Se trata de Android, que a día de hoy es una de las tecnologías que más avances ha sufrido. Android, es un sistema operativo inicialmente pensado para su uso en dispositivos móviles, al igual que otros S.O como podrían ser iOS, Symbian o BlackBerry OS. La principal característica que lo diferencia del resto de sistemas operativos, es que está basado en Linux, un núcleo de sistema operativo libre, gratuito y multiplataforma.

Este sistema operativo proporciona todas las interfaces necesarias para desarrollar aplicaciones que accedan a las funciones del teléfono, como puede ser el GPS, llamadas, agenda... Todo esto es posible a través de una forma sencilla a través de un lenguaje de programación muy conocido como es Java.

En la figura 4.1 se presenta la estructura de Android, donde se pueden apreciar distintas capas:

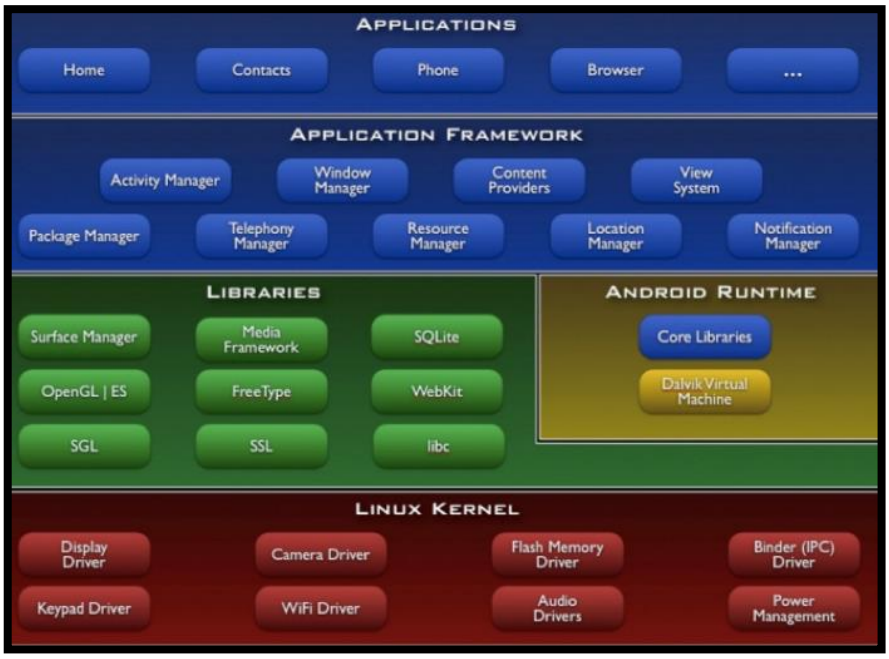


Figura 4.1: Estructura del sistema operativo Android.

Como se menciona anteriormente, Android está basado en Linux. Para ser más específicos, Android utiliza como base el kernel de Linux, ya que es open-source, lo que quiere decir que puede ser usado por otros sistemas para desarrollar aplicaciones.

En la figura 4.1, tenemos una estructura basada en diferentes capas. En la capa roja, denominada **capa del kernel**, se sitúa el kernel de Linux, es decir, el manejo de memoria, procesos, drivers… Aquí es donde se establece una comunicación con el hardware. Esto sirve para evitar la necesidad de tener un conocimiento específico sobre cada fabricante, por ejemplo, para usar la cámara, no es necesario saber cómo funciona la cámara de cada fabricante. En esta capa también se administran los recursos del celular, memoria, energía…

La siguiente capa, marcada con un color verde, es la **capa de librerías**. Esta capa contiene las librerías de Android, las cuales están escritas en C o C++ y contienen tareas específicas:

- **Surface Manager:** Utilizada para la gestión del acceso a la pantalla.
- **Media Framework:** Reproducción de imágenes, audio y vídeo.
- **SQLite:** Utilizada para las bases de datos.
- **Webkit:** Librería utilizada para el navegador.
- **SGL y OpenGL:** Estas librerías se utilizan para gráficos 2D y 3D respectivamente.
- **Freetype:** Librería utilizada para la renderización de vectores o imágenes.

La capa de color amarillo, es la **capa runtime.** En esta capa, se sitúa la máquina virtual de Android, también denominada *Dalvik*.

La capa azul, llamada **entorno de trabajo de las aplicaciones (application framework**), es la capa más visible para el desarrollador, ya que la mayoría de los componentes que forman parte del desarrollo se localizan en esta capa. Podemos encontrar diferentes componentes:

- **Activity Manager:** Administra las actividades de nuestra aplicación y el ciclo de vida.
- **Windows Manager:** Administra el contenido que se visualiza a través de la pantalla.
- **View:** Contiene las vistas de elementos que son parte de la interfaz gráfica, como mapas, cuadros de texto, etc.
- **Notification Manager:** Administra las notificaciones.

- **Package Manager:** Administra los paquetes y nos permite el uso de archivos en otros paquetes.
- **Location Manager:** Gestiona la posición geográfica.
- **Multimedia:** Administra el contenido referente a audio, video y fotos.

Por último, se encuentra la **capa de aplicaciones**, donde se localizan las aplicaciones que vienen en el dispositivo, como por ejemplo el gestor de correo, los mensajes, market, etc.

Una vez explicada la estructura de Android, podría surgir la siguiente pregunta, ¿Por qué desarrollar la aplicación para Android?

Para responder a esta cuestión, una de las principales ventajas del desarrollo de aplicaciones en Android, es que la base para la mayoría de aplicaciones es Java, ya que es uno de los lenguajes más universales y extendidos en el mundo del desarrollo de software, lo que hace más cómodo el desarrollo.

A diferencia de iOS, donde solo hay un fabricante, que es Apple, en Android hay una mayor cantidad de fabricantes con una gran cantidad de terminales disponibles. A simple vista, podría parecer una desventaja, pero desde el nacimiento de Android, se solucionó esta necesidad adaptando sus diseños a un tipo de diseño responsivo, lo que quiere decir que utilizaron una técnica de diseño que permite redimensionar y colocar los elementos que aparecen en las aplicaciones para que se adapten al tamaño de la pantalla de cada dispositivo para que se visualicen de una forma correcta.

Un punto importante a destacar de la elección de Android, es la facilidad de aprendizaje, ya que únicamente son necesarios los conocimientos sobre Java, y la obtención del SDK de Android, de manera gratuita.

4.1 Android Studio

En este apartado, se trata de mostrar una vista general del entorno de desarrollo donde vamos a crear la aplicación DicomApp. Android Studio, es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones para Android, basado en IntelliJ IDEA. Además del editor de códigos y de las herramientas que IntelliJ ofrece, Android Studio ofrece más funciones como son las siguientes:

- Sistema de compilación basado en Gradle flexible
- Emulador rápido con varias funciones

- Instant Run, para aplicar cambios mientras tu app se ejecuta sin la necesidad de compilar un nuevo APK.
- Compatibilidad con C++ y Android NDK

4.1.1 Estructura de un proyecto en Android Studio

En Android Studio, cada proyecto contiene uno o más módulos con archivos de código fuente y archivos de recursos. Los módulos más remarcables son los módulos de apps para Android, módulos de bibliotecas, y módulos de Google App Engine.

De una forma predeterminada, Android Studio permite visualizar los archivos en la vista de proyectos de Android. En la figura 4.2 podemos ver como el proyecto, se organiza en módulos para tener un acceso más rápido a los archivos. Dentro del módulo Application, tenemos diferentes niveles de carpetas en las cuales se organizan los distintos recursos de la aplicación.

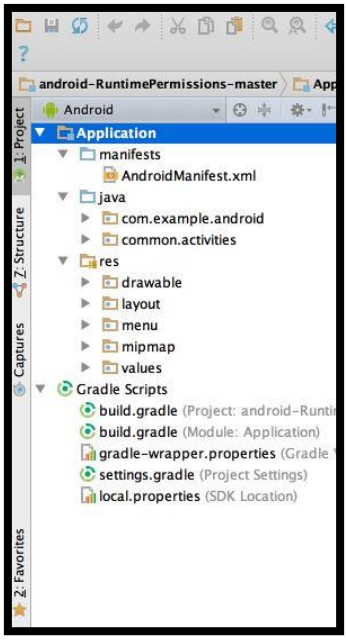


Figura 4.2: Archivos del proyecto en vista de Android.

4.1.2 Interfaz de usuario

En la figura 4.3, se muestra una vista de la interfaz de usuario de Android Studio.

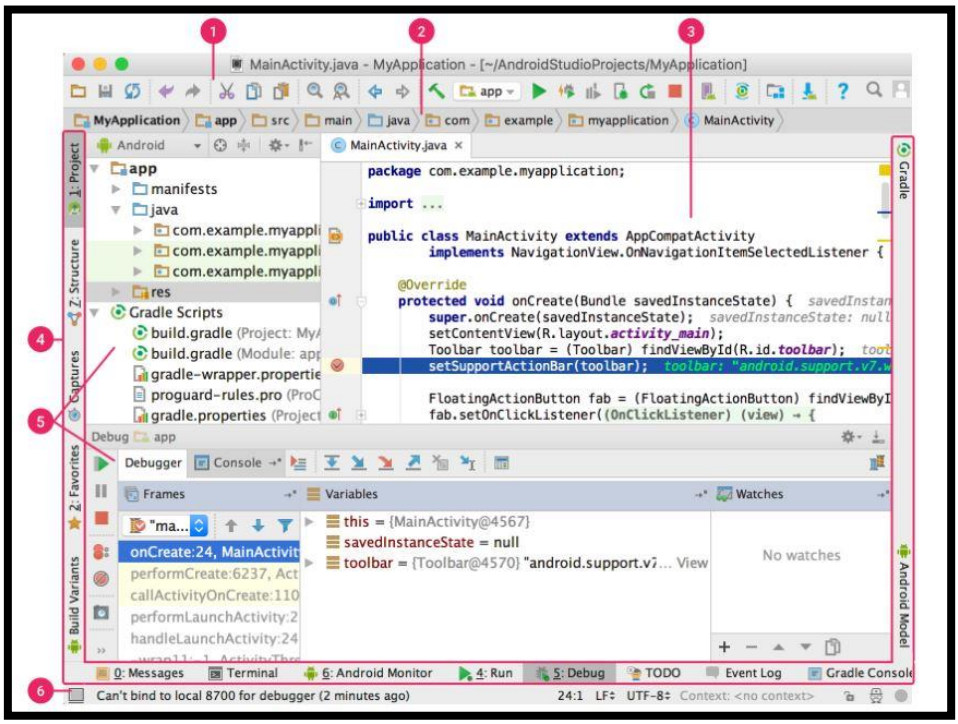


Figura 4.3: Ventana principal de Android Studio.

Dentro de la interfaz de Android Studio, podemos encontrar diferentes secciones:

1. **Barra de herramientas:** Permite realizar una gran variedad de acciones, como la ejecución de la aplicación y el inicio de herramientas de Android.
2. **Barra de navegación:** Sirve de ayuda para explorar el proyecto y abrir archivos para su edición. Proporciona una vista más compacta de la estructura visible en la ventana Project.
3. **Ventana del editor:** Área donde crear y modificar código. El editor puede cambiar en función del tipo de archivo, por ejemplo, al visualizar un archivo de diseño.
4. **Barra de la ventana de herramientas:** Se localiza alrededor de la parte externa de la ventana del IDE y contiene los botones que permiten expandir o contraer ventanas de herramientas individuales.
5. **Ventana de herramientas:** Permite acceder a tareas específicas, como la administración de proyectos, búsquedas, controles de versión...
6. **Barra de estado:** Muestra el estado del proyecto. También puede contener cualquier advertencia o mensaje.

4.2 Implementación de DicomApp

Una vez descrito, en el anterior punto, el software que se ha utilizado para la creación de la aplicación, se procede a explicar los conceptos más relevantes de la programación de la aplicación móvil. En este apartado, se incluyen fragmentos de código de las secciones más importantes de la aplicación, para una mejor comprensión de la misma.

Esta aplicación, ha sido desarrollada para que trabaje en una versión de Android 7.0 dado que la mayoría de los terminales se encuentran en esta versión o versiones anteriores.

En primer lugar, representado en la figura 3.4, se ha seleccionado un icono para representar dicha aplicación:

Figura 4.4: Imagen del icono de la aplicación móvil DicomApp

La aplicación consta de 3 vistas o pantallas en las que el usuario puede interactuar con la aplicación. La primera de ellas, es la vista de inicio, la cual se visualizará al abrir la aplicación. En la figura 4.5 se muestra un pantallazo de la primera vista de la aplicación:

Figura 4.5: Vista inicial de DicomApp

En la primera vista de la aplicación, se pueden realizar dos tareas principales, la primera de ellas, *hacer fotografía*, acciona la cámara del dispositivo móvil, para tomar una fotografía. La identificación del paciente se produce una vez nos conectamos a Elgg, ya que la red social exige la creación de una cuenta.

Para poder utilizar la cámara y guardar imágenes tomadas por ella, en primer lugar, se necesitan los permisos de escritura para guardar dichas imágenes. Añadiendo las siguientes líneas de código en el archivo *AndroidManifest.xml* obtenemos los permisos necesarios:

```
<uses-permission
android:name="android.permission.WRITE_INTERNAL_STORAGE" />
<uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.CAMERA" />
<uses-feature android:name="android.hardware.camera" />
```

Mediante el siguiente fragmento de código se puede accionar la cámara, y guardar una fotografía tomada por ella en un lugar específico del almacenamiento del dispositivo móvil:

```
    //Añadimos el Listener Boton
    bt_hacerfoto.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {

            //Creamos el Intent para llamar a la Camara
            Intent cameraIntent = new
Intent(MediaStore.ACTION_IMAGE_CAPTURE);

            String path =
Environment.getDataDirectory().getAbsolutePath().toString()+
"/storage/emulated/0/imagenesTFG";
            String path2 = "/storage/emulated/0/imagenesTFG/";
            File mFolder = new File(path2);
            if (!mFolder.exists()) {
                mFolder.mkdir();
            }
            //Añadimos nombre a la imagen
            String nameImg = "imagenJPG.jpg";
            File image = new   File(mFolder, nameImg);

            Uri uriSavedImage = Uri.fromFile(image);

            //Le decimos al Intent que queremos grabar la imagen
            cameraIntent.putExtra(MediaStore.EXTRA_OUTPUT,
uriSavedImage);
            //Lanzamos la aplicacion de la camara con retorno
(forResult)
            startActivityForResult(cameraIntent, TAKE_PICTURE);
        }});

}
```

Posteriormente, tras guardar la foto realizada, la aplicación muestra una vista donde se encuentra un formulario que rellenar con unos datos básicos del paciente, como son el nombre, los apellidos, el número de historial, y el sexo. Una vez rellenados los datos, al pulsar el botón *Guardar*, se genera un archivo con extensión .dcm junto con los datos introducidos en el formulario, y la aplicación regresa a la pantalla inicial.

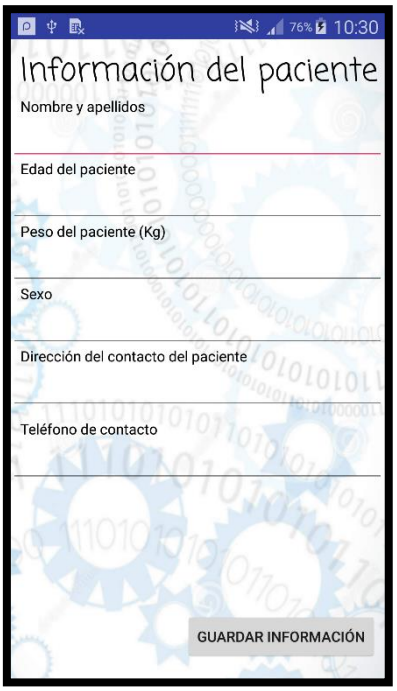


Figura 4.6: Vista del formulario de información del paciente.

Para que el botón *Guardar* genere el archivo con extensión .dcm con la fotografía tomada, y la información aportada, se ha optado por la utilización de una librería externa muy conocida en este ámbito, como es la librería "*dcm4che2*". Esta librería, contiene todos los métodos necesarios, ya desarrollados para la creación de imágenes con formato DICOM, además de un gran número de funcionalidades para el trabajo con imágenes médicas. Mediante el siguiente código, creamos un objeto DICOM y le introducimos algunos de los tags necesarios:

```
DicomObject dicom = new BasicDicomObject();
//Insertamos los TAGS necesarios para la formación de la imagen.
dicom.putString(Tag.PatientName, VR.PN, nombre);
dicom.putString(Tag.PatientAge, VR.AS, edad);
dicom.putString(Tag.PatientSex, VR.CS, sexo);
dicom.putString(Tag.PatientAddress, VR.LO, address);
dicom.putString(Tag.PatientWeight, VR.DS, peso);
dicom.putString(Tag.PatientTelephoneNumbers, VR.SH, telefono);
dicom.putString(Tag.PhotometricInterpretation, VR.CS, samplesPerPixel
== 3 ? "YBR_FULL_422" : "MONOCHROME2");

dicom.putInt(Tag.SamplesPerPixel, VR.US, samplesPerPixel);
```

```
dicom.putInt(Tag.Rows, VR.US, jpegImage.getHeight());
dicom.putInt(Tag.Columns, VR.US, jpegImage.getWidth());

dicom.putDate(Tag.InstanceCreationDate, VR.DA, new Date());
dicom.putDate(Tag.InstanceCreationTime, VR.TM, new Date());
```

Como se puede observar en el fragmento de código anterior, se han elegido los tags más relevantes para la información del paciente, como son el nombre completo, edad, sexo, dirección de contacto, peso, y teléfono de contacto. Existe una gran cantidad de tags disponible para completar la información de un usuario, pero se han elegido los más relevantes.

Una vez rellenada la información necesaria, y al hacer click en el botón *Guardar*, se ha programado un mensaje emergente que se muestre en la pantalla para indicar la creación de la imagen con éxito, tal y como se observa en la figura 4.7.

Figura 4.7: Vista de la confirmación de la imagen DICOM creada.

En caso de seleccionar el botón *Conectarse a Elgg*, la aplicación redirige a una nueva vista, donde, a través de *WebView* para una vista más cómoda, el dispositivo móvil se conectará a la dirección IP (52.57.17.241) donde está alojada la red social. En la siguiente figura, se muestra la vista del navegador al conectarse a la dirección IP de la red social.

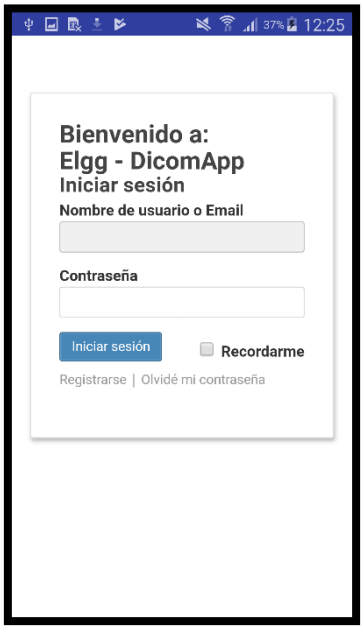


Figura 4.8: Vista del navegador, página inicial de Elgg – DicomApp

En este caso, el fragmento de código para poder iniciar el navegador integrado en la aplicación (WebView) es el siguiente:

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_elggbrowser);

    WebView myWebView = (WebView)
this.findViewById(R.id.webView);
    myWebView.setWebViewClient(new WebViewClient());
    myWebView.loadUrl("https://52.57.17.241");

}
```

Capítulo 5 – Conclusiones y Líneas futuras

5.1 Conclusiones

La realización de este trabajo ha supuesto una gran oportunidad para la profundización y comprensión de un estándar, importante en el campo de la medicina, pero desconocido para un gran sector de los alumnos de Telecomunicaciones. Gracias a este trabajo se puede apreciar la dimensión que tiene el trabajar con un estándar de este tipo. A su vez, también ha permitido profundizar tanto en protocolos de comunicación, para la conexión segura mediante HTTPS, como indagar en la programación Android a un nivel más complejo. Cabe mencionar que la experiencia con los entornos de trabajo de Amazon y de Elgg ha sido muy positiva, ya que son entornos intuitivos y amenos a la hora de trabajar con ellos.

Como resultado, se han alcanzado todos los objetivos propuestos para este trabajo, de manera que se ha realizado, por un lado, tanto la configuración como la implementación de una red social de carácter privado en una instancia EC2 de Amazon Web Services, configurada para ello en un sistema operativo Linux. Por otro lado, mediante un software de desarrollo de aplicaciones móviles, en este caso Android Studio, se ha desarrollado una aplicación móvil para la generación de archivos con extensión .dcm donde se encuentra tanto la imagen previamente realizada, cómo información acerca del paciente. De manera adicional, también se ha añadido una función extra a la aplicación para la conexión entre el dispositivo móvil y la plataforma alojada en la nube, de manera que se conecte utilizando el protocolo HTTPS.

En este Trabajo Fin de Grado, quedan reflejados los conocimientos impartidos durante los 4 años de duración que tiene el Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación.

Como opinión personal del autor de este trabajo, la realización del mismo ha supuesto una experiencia muy positiva, al poder tener la oportunidad de trabajar en un campo en plena proyección y por el que siento una gran devoción. También quedo muy satisfecho por el resultado obtenido y los objetivos alcanzados en este proyecto.

5.2 Líneas futuras

Como líneas futuras para la ampliación de este trabajo, se han propuesto las siguientes ideas o funcionalidades que mejorarían la calidad del trabajo:

- Crear un chat personalizado dentro de la red social, que permita ponerse en contacto en tiempo real al paciente con el especialista.
- Añadir un sistema que envíe una notificación al correo electrónico del paciente cuando éste ha obtenido una respuesta por parte de un profesional.
- Añadir el software necesario en la red social, de manera que exista una pestaña para poder realizar el proceso de generación de imágenes con formato .dcm desde la propia red social, sin necesidad de tener la aplicación instalada en el dispositivo móvil.
- En concordancia con el punto anterior, añadir el software necesario para la creación de un visor de documentos con formato DICOM, ya que se necesita de un visor especial. Esto permitiría no tener que descargar un visor específico para el visionado de los documentos.
- Respecto a la aplicación móvil, añadir la posibilidad de dónde guardar, tanto las fotografías como los documentos con formato .dcm dentro del dispositivo. Actualmente se genera una carpeta específica para el guardado de dichos documentos.
- En relación al punto anterior, la creación de una sección, tanto en la aplicación móvil como en la red social, que permita generar una imagen con formato .dcm a partir de una imagen ya existente, sin la necesidad de tener que realizar una fotografía.
- Realizar el cifrado de los datos almacenados en la red social ubicada en la nube, para disponer de mayor seguridad a la hora de guardar estos documentos.

Capítulo 6 – Bibliografía

[1] Estándar DICOM. Accedido en 2016-2017.

http://dicom.nema.org/standard.html

[2] Elgg – Página de inicio.

https://elgg.org

[3] BuddyPress – Página de inicio. Accedido en 2017

https://es.buddypress.org

[4] HumHub – Página de inicio. Accedido en 2017

https://www.humhub.org/en

[5] Android – Wikipedia Android. Accedido en 2017

https://en.wikipedia.org/wiki/Android_(operating_system)

[6] Android Studio. Accedido en 2017

https://developer.android.com/studio/index.html?

[7] Android Programming Tutorials. 2009. Mark Murphy

[8] Amazon Web Services. Accedido durante 2016-2017.

https://aws.amazon.com/es/

[9] Freier, A.; Karlton, P.; Kocher, P. (Agosto de 2011) RFC 6101 - The Secure Sockets Layer (SSL) Protocol Version 3.0.
https://tools.ietf.org/html/rfc6101

Anexo A – Configuración del servidor

En este anexo, explicaremos de manera breve y sencilla, cómo configurar todos los elementos necesarios para poner en funcionamiento la red social Elgg.

Para ello, necesitaremos configurar la base de datos localizada en la instancia de Amazon RDS, instalar la red social Elgg en la instancia de Amazon EC2, y configurar tanto Apache como los certificados SSL también en la instancia de Amazon EC2. A continuación, indicaremos todos los pasos necesarios para dicha configuración.

A.1 Configuración de la base de datos en la instancia de Amazon RDS.

*** En caso de no tener instalado mysql, debemos ejecutar el siguiente comando.*

```
sudo apt-get install mysql-server mysql-common mysql-client
```

En primer lugar, para realizar dicha configuración, accedemos a la base de datos desde un ordenador que tenga instalado mysql.

```
$ mysql –u [tu.nombre.de.usuario] –p –h
nombre.de.la.instancia.rds.amazonaws.com
```

Una vez que hemos accedido a la base de datos en la instancia, se crea la base de datos y el usuario.

```
CREATE DATABASE elgg;

CREATE USER usuario.de.la.base.de.datos IDENTIFIED BY 'password';

GRANT ALL ON elgg.* TO usuario.de.la.base.de.datos;
```

***Mediante el comando "GRANT ALL ON", se permite el acceso a la base de datos al usuario indicado.*

A.2 Instalación de Elgg en la instancia de Amazon EC2.

1. Comprobar y realizar las actualizaciones pertinentes del sistema.

   ```
   $ sudo apt-get update
   $ sudo apt-get upgrade
   ```

2. Instalación de Apache2.

   ```
   $ sudo apt-get install apache2
   ```

3. Habilitar el modo *rewrite* de Apache2.

   ```
   $ sudo a2enmod rewrite
   ```

4. Modificar el fichero *"apache2.conf"* ubicado en el directorio *"/etc/apache2/"* para que Apache tenga en cuenta el fichero *.htaccess* que necesita Elgg para trabajar.

   ```
   $ sudo nano /etc/apache2/apache2.conf
   ```

   Cambiar `AllowOverride None` por `AllowOverride All` en la sección 'Directory /var/www/' de manera que quedará la siguiente configuración final en esa sección:

   ```
   <Directory /var/www/>
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
   </Directory>
   ```

5. Reiniciar el servidor Apache2.

   ```
   $ sudo /etc/init.d/apache2 restart
   ```

   También se puede realizarlo en dos pasos (a veces la orden restart no se ejecuta bien)

   ```
   $ sudo /etc/init.d/apache2 stop
   $ sudo /etc/init.d/apache2 start
   ```

6. Descargar la versión actual de Elgg en la carpeta /var/www/html con el comando *wget.*

```
$ sudo wget getelgg.php\?forward\=elgg-2.2.2.zip
```

7. Descomprimir el fichero con extension ".zip" que hemos descargado, y lo copiamos a la carpeta /var/www/html

```
$ sudo unzip elgg-2.2.2.zip

$ sudo rm elgg-2.2.2.zip

$ sudo mv elgg-2.2.2/* /var/www/html/

$ sudo rm –r –f elgg-2.2.2
```

8. Crear una carpeta para el almacenamiento de datos de Elgg, fuera del directorio raíz del servidor web, es decir, en el directorio /var/

```
$ sudo mkdir /var/elggdata
```

9. Dar permisos de escritura exclusivamente al servidor web sobre la carpeta recientemente creada.

```
$ sudo chown –R www-data:www-data /var/elggdata
```

10. Mover el archivo que se llama *htaccess.dist,* cuya ruta es /var/www/html/install/config/ a la siguiente ruta: /var/www/html/ con el nombre ".htaccess".

```
$ sudo mv /var/www/html/install/config/htaccess.dist
/var/www/html/.htacess
```

11. Mover el fichero que se llama *settings.example.php,* cuya ruta es /var/www/html/engine/ a la siguiente ruta: /var/www/html/engine/ con el nombre "settings.php".

```
$ sudo mv /var/www/html/engine/settings.example.php
/var/www/html/engine/settings.php
```

12. Abrir el fichero settings.php y cambiar los valores de configuración.

```
$ sudo nano /var/www/html/engine/settings.php
```

Tendrá que quedar la siguiente configuración:

```
$CONFIG->dbuser = 'usuario.de.la.base.de.datos';

$CONFIG->dbpass = 'password';

$CONFIG->dbname = 'elgg';
```

```
$CONFIG->dbhost                                                    =
‘nombre.de.la.instancia.rds.amazonaws.com:3306’;
$CONFIG->dbprefix = ‘elgg_’;
```

13. Instalar la red social Elgg desde un navegador, introduciendo la dirección:

    https://52.57.17.241/install.php

14. Para la instalación, sólo hay que seguir los pasos indicando los datos necesarios para dicha configuración. Cuando la red social Elgg esté completamente instalada, se mostrará una pantalla de bienvenida.

Anexo B – Instalación de apache y de un certificado SSL en la instancia EC2

1. Instalar Apache2.

   ```
   $ sudo aptitude install apache2
   ```

2. Habilitar el módulo SSL.

   ```
   $ sudo a2enmod ssl
   ```

3. Habilitar la configuración por defecto de SSL.

   ```
   $ sudo a2ensite default-ssl
   ```

4. Reiniciar el servidor Apache2.

   ```
   $ sudo /etc/init.d/apache2 restart
   ```

5. Crear las llaves en el directorio /etc/apache2/

   ```
   $ cd /etc/apache2/
   $ sudo openssl genrsa -des3 -out server.key 1024
   ```

6. A partir de la llave, crear el certificado.

   ```
   $ -sudo openssl req -new -key server.key -out server.csr
   ```

7. Crear el certificado autofirmado, contestando a las preguntas según corresponda.

   ```
   $ sudo openssl x509 -req -days 365 -in server.csr -signkey
   ```

```
server.key -out server.crt
```

8. Mover la llave y el certificado a las carpetas /etc/ssl/private/ y /etc/ssl/certs/ respectivamente.

```
$ sudo cp server.key /etc/ssl/private/
$ sudo cp server.crt /etc/ssl/certs/
```

9. Abrir el archivo de configuración default-ssl.conf ubicado en el directorio /etc/apache2/sites-available/

```
$ cd /etc/apache2/sites-available/
$ sudo nano default-ssl.conf
```

10. Reemplazar y habilitar la siguiente configuración en el archivo.

```
SSLOptions +FakeBasicAuth +ExportCertData +StrictRequire
SSLCertificateFile /etc/ssl/certs/server.crt
SSLCertificateKeyFile /etc/ssl/private/server.key
```

11. Habilitar los cambios realizados en el archivo.

```
$ sudo a2ensite default-ssl
```

12. Reiniciar el servidor Apache2.

```
$ sudo /etc/init.d/apache2 restart
```

Para redireccionar todo el tráfico que el usuario genere mediante el protocolo HTTP a este servidor, se realiza de la siguiente manera:

1. Modificar el fichero 000-default.conf localizado en /etc/apache2/sites-available/

```
$ cd /etc/apache2/sites-available/
$ sudo nano 000-default.conf
```

2. Añadir la siguiente configuración al archivo.

```
Redirect permanent / https://52.57.17.241/
```

3. Reiniciar el servidor Apache2.

```
$ sudo /etc/init.d/apache2 restart
```