UNIVERSIDAD DE VALLADOLID

E.T.S.I TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA DE TECNOLOGÍAS ESPECÍFICAS DE TELECOMUNICACIÓN, ESPECIALIDAD TELEMÁTICA

Diseño y Desarrollo de una aplicación iOS para el control y seguimiento de enfermedades.

Autor:

Adrián Benito Valverde

Tutora:

Dña. Miriam Antón Rodríguez

TÍTULO:	**Diseño y Desarrollo de una aplicación iOS para el control y seguimiento de enfermedades.**
AUTOR:	**Adrián Benito Valverde**
TUTORA:	**Dña. Miriam Antón Rodríguez**
DEPARTAMENTO:	**Teoría de la Señal y Comunicaciones e Ingeniería Telemática**

TRIBUNAL

PRESIDENTA:	**Dña. Miriam Antón Rodríguez**
VOCAL:	**D. Mario Martínez Zarzuela**
SECRETARIO:	**D. David González Ortega**
SUPLENTE:	**D. Francisco Javier Díaz Pernas**
SUPLENTE:	**Dña. Mª Ángeles Pérez Juárez**

FECHA:	25-11-2014
CALIFICACIÓN:	

Resumen del TFG

El objetivo principal de este proyecto es desarrollar una aplicación eHealth con soporte multilenguaje y para dispositivos iOS, cuyo fin es, además de aportar información sobre una enfermedad como es en este caso, el hepatocarcinoma celular o cáncer hepático, llevar un seguimiento sobre esta en los pacientes mediante el aporte de datos. Por otra parte, toda esta información queda registrada en una base de datos pudiendo acceder a ella a través de una aplicación Web.

Otro de los hitos a cumplir con este trabajo es desarrollar una aplicación cuyo contenido fuese en su mayor parte dinámico, no siendo necesario tocar la aplicación para modificarlo y por consiguiente que soporte cualquier idioma, y escalable en el sentido de facilitar la inclusión de nuevas enfermedades dentro de la propia aplicación.

Como consecuencia de la necesidad de desarrollar la aplicación para esta plataforma, ha hecho posible el aprendizaje de un lenguaje de programación como es *Objective-C*.

Palabras Clave
eHealth, iOS, app, Objective-C, IPhone, CHC.

Abstract

The main purpose of this project is to develop an eHealth app that support multilanguage and particularly to iOS devices, whose purpose is, besides giving information about an illness such as hepatocellular carcinoma, keep track of a disease on patients through the provision of data. On the other hand, all of this information keep stored into a database to Access with a Web application.

In addition, others purposes to accomplish with this work were to develop an application whose content was largely dynamic been not necessary change the app to modify it and therefore, support any language, and scalable in the sense of facilitate the inclusion of new diseases into the app.

As a result of the need to develop an app to this platform, it has become possible to learn a programming language such as *Objective-C*.

Keywords

eHealth, iOS, app, Objective-C, IPhone, HCC.

Agradecimientos

Agradecer a mis padres Cesar B. y Montserrat V. todo el apoyo durante la realización de este trabajo y mi paso por la universidad.

A mi tía M. Carmen B. por ayudarme y aconsejarme durante el transcurso de estos años.

A mi tutora Dña. Míriam Antón Rodríguez por darme la oportunidad de desarrollar este trabajo y por confiar en mí para ello.

Índice de contenidos

Índice de tablas

Índice de ilustraciones

1 Introducción

1.1 Introducción al proyecto

Hoy en día, el cáncer es una de las principales causas de mortalidad en todo el mundo. Según estudios de la Organización Mundial de la Salud, en el año 2012, esta enfermedad causó un total de 8,2 millones de defunciones y se prevé que para las dos próximas décadas, el número de casos anuales aumente de 14 a 22 millones [1].

Entre los numerosos tipos de cáncer que existen, es necesario destacar el cáncer pulmonar como principal causa de muerte dentro de esta enfermedad [2]. Sin embargo, en segundo lugar, nos encontramos con un tipo sobre el que trabajaremos en este proyecto, el cáncer hepático o cáncer de hígado.

Aunque lo deseable para reducir el índice de mortalidad de esta enfermedad sería que disminuyera el número de casos que aparecen, en la actualidad este hecho es poco asumible debido a que los principales factores de riesgo vienen dados por la propia actividad humana como son el consumo de tabaco y alcohol, inactividad física o dietas malsanas, y por otros factores externos de los que no se espera un decremento, como la contaminación del aire de las ciudades y la de combustibles sólidos [3].

Es por ello que, a esperas de que la medicina dé con un mecanismo capaz de prevenir esta enfermedad, ya sea una vacuna por ejemplo, una de las opciones más viables junto con el uso de medicamentos, a la hora de reducir este número de defunciones es el uso de la tecnología para el diagnóstico y seguimiento del cáncer. Una aplicación eHealth que permita esto es un claro ejemplo.

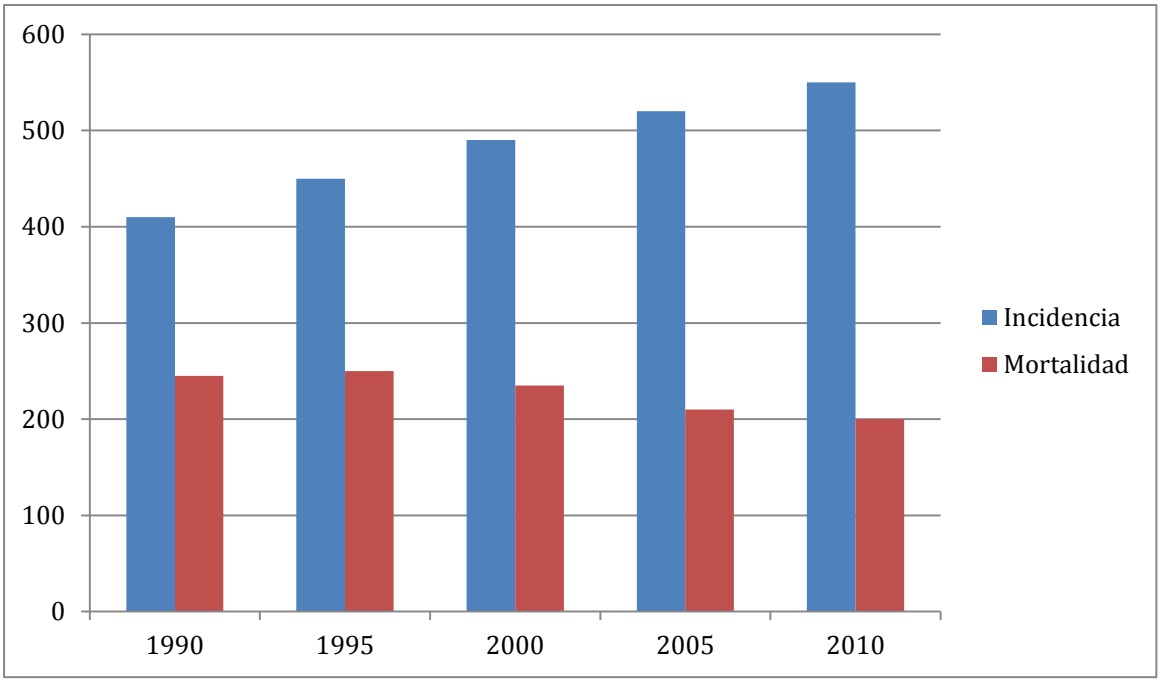

Tabla 1. Número de incidencias y muertes (miles) en España en los 20 años

Fuente: Las cifras del cáncer en España 2014, SEOM, Sociedad Española de Oncología Médica. [4]

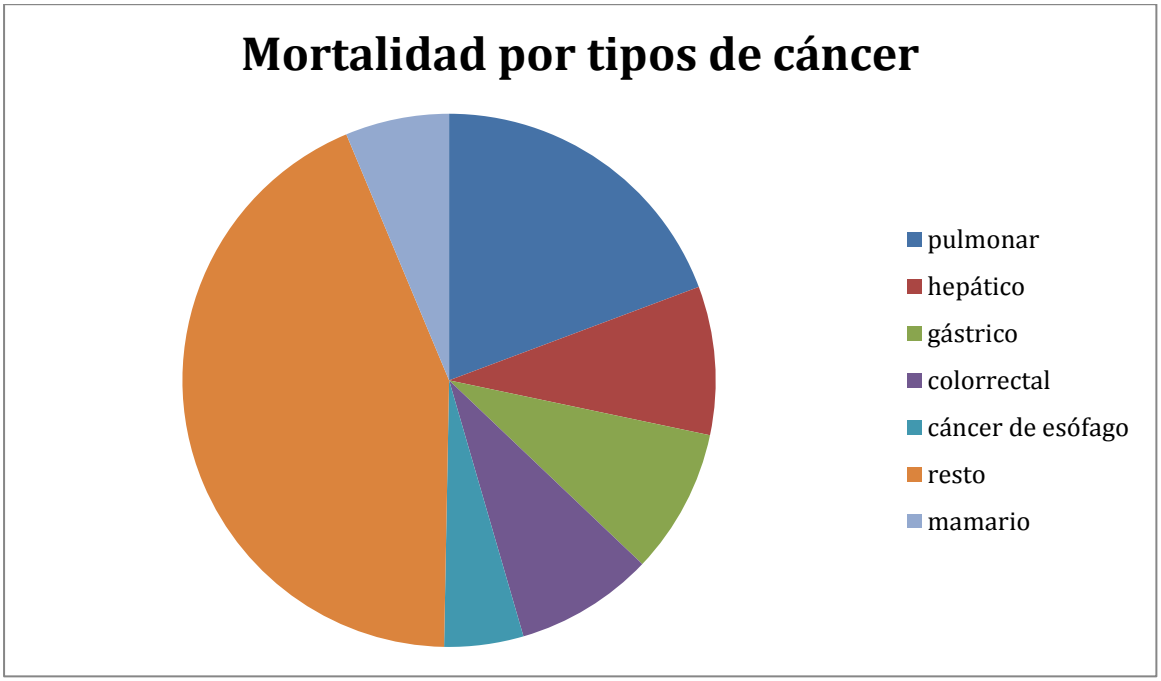

Tabla 2. Mortalidad por tipos de cáncer

Fuente: Las cifras del cáncer en España 2014, SEOM, Sociedad Española de Oncología Médica. [4]

1.2 Motivación y objetivos

Viendo que la tecnología está cada vez más presente en nuestras vidas no solo con el fin de divertirnos, sino también permitiéndonos tener una vida más cómoda y longeva, me siento con ganas de asumir el cargo de este trabajo. Por una parte aprovechar la oportunidad de aprender un nuevo lenguaje de programación empleado en el desarrollo de aplicaciones móviles como es *Objective-C*, y por otro lado, desarrollar una aplicación cuyos aspectos a destacar son la escalabilidad y el dinamismo de su contenido y que permita a profesionales médicos diagnosticar y controlar la evolución de una enfermedad a través de distintos algoritmos y criterios que no serían prácticos de aplicar sin la ayuda de un sistema informático. La enfermedad que se ha elegido y sobre la que se centrará el proyecto y este documento es el hepatocarcinoma celular o cáncer hepático.

Por otro lado, se desarrollará en paralelo una aplicación web que permita a pacientes y profesionales consultar registros y resultados.

1.3 Descripción del problema

Se plantea la necesidad de desarrollar una aplicación móvil nativa para iOS que permita:

- Obtener información básica sobre una enfermedad; Causas, síntomas, diagnostico, posible tratamiento…
- Obtener información específica sobre ella como documentos externos o contenido multimedia.
- Obtener un conjunto de resultados a través de la introducción de datos por parte del profesional médico y el procesamiento de estos que permitan diagnosticar o evaluar el proceso de una enfermedad sobre un paciente.
- Tener un registro de resultados así como los pacientes en una base de datos.

Desarrollar una aplicación para navegador web que permita:

- Gestionar el alta de profesionales médicos a la app.
- Gestionar el listado de pacientes así como las pruebas realizadas sobre ellos cuando accedemos como un profesional médico.
- Gestionar el listado de pruebas realizadas cuando accedemos como un paciente.

1.4 Metodología

Para el desarrollo de este proyecto se ha establecido un plan de trabajo de acuerdo al número total de horas a realizar, a la fecha prevista de cierre y a una estimación de la carga de trabajo que va a suponer cada fase del proyecto.

La primera fase del proyecto consiste en el proceso de **captura de requisitos.** Se lleva a cabo un planteamiento del problema teniendo en cuenta los aspectos más importantes que será necesario implementar en la aplicación. Se valora la viabilidad de estos, su carga de trabajo, la posibilidad de cumplir estos requisitos dentro de los plazos establecidos y las posibles mejoras que podrían asumirse.

En segundo lugar es necesario **documentarse.** Por una parte sobre la propia enfermedad elegida como ejemplo, que es el hepatocarcinoma celular, sistemas de diagnóstico, criterios para trasplantes, escalas…
Para esta información se ha hecho uso de un documento elaborado por el doctor K. Burak de la Universidad de Calgary en Canadá [5] donde quedan reflejados los distintos algoritmos que hará uso esta aplicación, la obtención de los resultados.
Por otro lado, ha sido necesario documentarse sobre el entorno de desarrollo con el que se va a trabajar*, Xcode* 5 [6], y sobre el proceso de desarrollo de aplicaciones iOS con el lenguaje de programación *objective-C* [7][8][9].

El siguiente paso que se ha seguido se basa en el desarrollo de la aplicación, tanto de la parte de **programación** de la lógica de negocio como el diseño de las interfaces que se mostrará al usuario. Bien es cierto que esta fase del proyecto es la que requiere más tiempo y dedicación debido a la falta de experiencia en la programación del lenguaje utilizado y a la necesidad del continuo aprendizaje y documentación que estará presente durante el transcurso de esta fase. Por otra parte, una vez implementadas todas las funcionalidades de las que se desea dotar a la aplicación, es necesario realizar un conjunto de pruebas que verifiquen el correcto funcionamiento de estas, así como un repaso del código fuente en busca de posibles mejoras que pudiesen aportar robustez a la aplicación.

La siguiente y última fase de todo este proceso se basa en la elaboración de una **memoria**, este documento, que estará formado por un conjunto de capítulos como se definen en la siguiente sección.

1.5 Introducción al documento

El documento está estructurado en 5 secciones cada una de ellas formadas por varios capítulos cuyo contenido se describe en la siguiente lista:

1. **Introducción**

   1.1 Introducción al proyecto

   Se pretende introducir el tema abordado en el proyecto, contextualizarlo dentro de un marco actual con el fin de que el lector sienta curiosidad o atracción sobre el resto del contenido. Se describe el cáncer como una enfermedad con alto índice de mortalidad y como el uso de la tecnología, en nuestro caso, una aplicación eHealth para dispositivos móviles, puede resultar beneficiosa.

   1.2 Motivación y objetivos

   Se exponen las diferentes razones por las cuales se ha decidido llevar a cabo este proyecto, así como cuales son los principales objetivos que se pretenden alcanzar con su desarrollo.

   1.3 Descripción del problema

   Tiene como objetivo analizar la situación actual que se describe en la introducción del proyecto e identificar cuáles son los principales aspectos sobre los que sería necesario cubrir o tratar en el desarrollo del trabajo.

1.4 Metodología

Se recogen y exponen las distintas fases que se han dado durante el desarrollo de todo el proyecto, desde la creación de un plan de trabajo y captura de requisitos hasta elaboración de este documento.

2 Tecnologías y conocimientos previos

2.1 Tecnologías para aplicaciones web.

Detalla un conjunto de tecnologías válidas para la realización de dicha tarea mostrando sus principales ventajas e inconvenientes y analizando el por qué o no de su elección.

2.2 Conexión con base de datos

Se expone la necesidad de una base de datos, así como el lenguaje de consulta a ella o los principales sistemas de gestión que existen para acabar con un análisis de la solución más adecuada.

2.3 Tecnologías para aplicaciones móviles

Se introduce y abarcan los distintos tipos de aplicaciones móviles además de las plataformas móviles actuales más importantes con el objetivo de determinar qué tipo se ajusta más a nuestras necesidades.

2.4 Herramientas de control de versiones

Se analizan distintas alternativas en el uso de un sistema de control de versiones.

3 Desarrollo de la aplicación

3.1 Contextualización

Se pretende situar la aplicación en un contexto hablando sobre la hepatología e eHealth.

3.2 Beneficios de la aplicación

Se recogen los distintos beneficios que pueden aportar las aplicaciones.

3.3 Descripción técnica

En este apartado se pretenden detallar todos los aspectos técnicos a tener en cuenta de las aplicaciones. Desde las distintas forma en cómo se han alcanzado las características de la aplicaciones, a como se han implementado las funcionalidades de estas.

3.4 Manual de usuario

Con este punto se pretende que sirva de guía a la hora de hacer uso de las aplicaciones.

3.5 Pruebas

Se recogen distintos casos de prueba con el fin de verificar el correcto funcionamiento de las aplicaciones.

4 Presupuesto económico

En este apartado se pretende realizar una estimación del coste económico que implica el desarrollo de la aplicación.

5 Conclusión y líneas futuras

Se recogen las distintas conclusiones fruto de la elaboración de este proyecto así como valorar las distintas líneas futuras que pudiesen acontecer.

2 Tecnologías y conocimientos previos

2.1 Tecnologías para aplicaciones Web. Lado del cliente.

Una parte del proyecto se basa en el desarrollo de una aplicación Web desde donde los usuarios, ya sean profesionales médicos o pacientes, puedan llevar un control sobre los resultados obtenidos en la aplicación móvil.

Para su implementación, se han valorado distintas tecnologías que puedan cumplir ampliamente su cometido. Sí es cierto, que dada las características que debe tener, es necesario hacer previamente una distinción entre tecnologías del lado cliente y tecnologías del lado del servidor.

Las Tecnologías del lado del cliente [10] son aquellas que se ejecutan en el navegador del usuario donde toda la carga de procesamiento de los efectos y funcionalidades las soporta este. Permiten crear interfaces de usuario atractivas pero restándole dinamismo.

2.1.1 HTML

Es el lenguaje básico que constituye casi todo el contenido web, con él se escribe la estructura y semántica que va a aparecer en el documento.

HTML (HyperText Markup Language) [11] está basado en SGML (Standard Generalized Markup Language) un sistema que permite definir lenguajes para dar formato a documentos.

A día de hoy es el la tecnología por excelencia para la maquetación de páginas Web. En la aplicación se hará uso de HTML 4.01.

2.1.2 CSS

CSS (Cascading Style Sheets) [12] se encarga de describir cómo se va a mostrar un documento en el navegador. Es usado para dar estilos a documentos HTML o XML separando el contenido de la presentación, permitiendo detallar la forma en la que se muestran los distintos elementos HTML en una página.

En la actualidad, emplear esta tecnología para darle atractivo visual a la aplicación es la opción más viable.

2.1.3 JavaScript

Es un lenguaje de programación que permite crear páginas web dinámicas sin la necesidad de que intervenga un servidor. Se trata de un lenguaje de programación interpretado, por lo que es necesaria la compilación para su ejecución. Esto supone que los programas escritos con este lenguaje no requieran de procesos intermediarios a la hora de probarlo en un navegador.

2.1.4 AJAX

Es un conjunto de técnicas para desarrollo web usado en el lado del cliente que permite crear aplicaciones asíncronas. Con AJAX, estas aplicaciones pueden enviar datos y recibirlos de un servidor de tal forma que esto no afecte con el comportamiento de la página.

Las tecnologías que forman AJAX [13] son:

- XHTML y CSS para la creación de una presentación basada en estándares.
- DOM (Document Object Model) como interfaz de programación de aplicaciones para la manipulación e interacción dinámica de la presentación.

- XML, XSLT (Extensible Stylesheet Language Transformations) y JSON (JavaScript Object Notation) para el intercambio y la manipulación de información.
- XMLHttpRequest para el intercambio asíncrono de información.
- JavaScript para unir todas las demás tecnologías.

2.2 Tecnologías para aplicaciones Web. Lado del servidor

Un lenguaje del lado del servidor es aquel que se ejecuta en el lado del servidor web justo antes de que se envíe la página a través de Internet al cliente.

A diferencia del anterior tipo de tecnologías, en este punto, disponemos de distintos lenguajes de programación que nos permitirían cumplir nuestro cometido [14].
A continuación se valoran distintos lenguajes ampliamente utilizados para el desarrollo de páginas dinámicas destacando sus principales características así como sus ventajas e inconvenientes y el porqué de su elección o no.

Tabla 3. Lenguajes del lado del servidor más usados. Octubre 2014 (W3Techs, 2014)

Fuente: W3techs.com World Wide Web Technology surveys [15]

2.2.1 ASP.NET

Este lenguaje comercializado por Microsoft y usado para desarrollar entre otros, sitios web, es el sucesor de la tecnología ASP [16]. Es posible utilizar lenguajes como C Sharp, Virtual Basic o J Sharp para su desarrollo siendo necesario tener instalado el servidor web del sistema operativo de Microsoft Windows, IIS y el framework de .Net.

Entre las principales ventajas se encuentran:

- Completamente orientado a objetos.
- División entre la capa de aplicación y el código.
- Facilita el mantenimiento de grandes aplicaciones.
- Elevada velocidad de respuesta del servidor.
- Elevada seguridad.

Como desventajas tenemos:

- Tecnología propietaria de Microsoft
- Alto consumo de recursos
- Elevado coste en el hospedaje de sitios web.

Debido a la falta de experiencia en el uso de este lenguaje de programación y a la falta de recursos a la hora de alojarlo en un Hosting se ha desechado la idea de emplearlo como lenguaje del lado del servidor.

2.2.2 JSP y JSF

JSP (Java Server Pages) es un lenguaje para la creación de sitios web basándose en Java. Por otro lado, JSF (Java Server Faces) es un framework para crear aplicaciones Java J2EE haciendo uso de páginas JSP para generar las vistas. Esta tecnología nos permite desarrollar aplicaciones dinámicas donde toda la lógica de negocio se implementa con Java.

Las principales ventajas de usar JSF son:

- Código JSF basado en etiquetas jsp muy similares al estándar de HTML.
- Está integrado dentro del código jsp y se encarga de la recogida y generación de los valores de los elementos de la página.
- Dispone de funcionalidades para la resolución de validaciones, conversiones o mensajes de error.
- Es totalmente compatible con JavaScript y Ajax permitiendo incluir este código.
- Es escalable, permitiendo añadir nuevos componentes a medida.

Como principal desventaja cabe destacar la complejidad de la API y lo que conlleva su correcto aprendizaje.

A pesar de haber obtenido experiencia haber trabajado con él, se ha decidido no emplearlo como lenguaje de programación del lado del servidor por los siguientes motivos:

- Desarrollar una aplicación móvil no basada en Java y por consiguiente no pudiendo consumir directamente sus servicios web.
- El tiempo de desarrollo que conllevaría utilizar este lenguaje para la funcionalidad que pretende cubrir.
- La necesidad de tener lanzado un servidor Tomcat frente a otras alternativas más económicas y viables.

2.2.3 Perl

Es un lenguaje de programación basado en scripts portable a cualquier plataforma que no sigue ninguna filosofía de programación concreta [17]. Aunque soporta varios paradigmas de programación, no se puede decir que sea orientado a objetos, modular o estructurado siendo su punto fuerte las labores de procesamiento de textos y archivos.

Como ventajas hay que destacar:

- Velocidad a la hora de programar.
- Soporte en multitud de plataformas y sistemas operativos.
- Es un software libre, gratuito.
- Le otorga al programador mucha libertad a la hora de programar.
- No obliga a seguir o elegir un paradigma de programación

Su principal desventaja es la falta de ordenación del lenguaje debido a la libertad que le hace destacar.

No se ha elegido Perl como lenguaje de programación del lado del servidor debido a la falta de experiencia con él y a la necesidad de tiempo extra para documentarse.

2.2.4 PHP

En la actualidad se ha convertido en uno de los lenguajes de programación por excelencia a la hora de desarrollar aplicaciones web dinámicas que trabajen del lado del servidor [18]. Desde su comienzo en 1994 y gracias a su política de código abierto, ha conseguido a lo largo de su historia que muchos desarrolladores hagan sus propias contribuciones a este lenguaje.

Las ventajas que tiene son numerosas:

- Es un lenguaje fácil de aprender debido a su semántica.
- Es un lenguaje rápido donde el cliente solo recibe una página HTML resultado de la ejecución de la PHP.
- Es compatible con todos los navegadores siendo un lenguaje multiplataforma.
- Posibilita la conexión con la mayoría de manejadores de base de datos. MySQL, Oracle, SQL Server…
- Uso de una librería de funciones muy extendida.
- No requiere definición de tipos de variables.
- Incorpora módulos de seguridad.

Por otra parte las desventajas que presentan son:

- La programación orientada a objetos no es muy eficiente para aplicaciones grandes.
- Dificulta la organización por capas de la aplicación.

En este caso, debido a las ventajas que posee y a experiencia adquirida en este lenguaje durante la realización de otros proyectos, PHP ha sido el lenguaje del lado del servidor elegido para el desarrollo de la aplicación.

2.3 Conexión a base de datos

Uno de los objetivos de la aplicación y de la necesidad de desarrollar una aplicación web, es el registro en una base de datos de todos los resultados obtenidos y capturados por la aplicación móvil.

Para ello, es necesario por una parte un lenguaje que nos permita realizar una consulta contra una base de datos y por otro lado, un sistema para la gestión de esta.

2.3.1 Lenguaje de consulta estructurado, SQL

Es el lenguaje estándar para la consulta a base de datos relacionales. Este lenguaje de acceso a base de datos permite realizar diversas operaciones sobre ella a través de sentencias permitiendo recuperar de forma sencilla información contenida en la base de datos así como hacer cambios sobre ellas.

Dentro de este lenguaje, se pueden distinguir varios tipos de sentencias [19]. En primer lugar, el lenguaje de definición de datos o DDL permite definir la estructura de los datos. También tenemos el lenguaje de manipulación de datos o DML que permite llevar a cabo tareas de consulta y modificación de los datos contenidos en una base de datos. Por otro lado se encuentra el lenguaje de control de datos y el de control de transiciones permitiendo realizar tareas como conceder o revocar permisos o hacer un Commit o Rollback de la base de datos.

SQL es el lenguaje por excelencia empleado en la mayoría de los sistemas de gestión de bases de datos. Estos últimos, son necesarios para poder almacenar, extraer o modificar la información que contendrá la base de datos. En los siguientes puntos se analizan distintas herramientas que nos permite realizar esta función**.**

2.3.2 Microsoft SQL Server

Es un sistema de gestión de base de datos relacionales desarrollado por Microsoft. A día de hoy está entre los tres sistemas más populares a la hora de administrar una base de datos [20]. Tanto es así que existen numerosas ediciones en función de la carga de trabajo o el tamaño de la aplicación desde donde se quiere dar uso.

En nuestro caso, debido a la complejidad de nuestra base de datos y a la imposibilidad de tener un servidor dedicado con este sistema de gestión, no es una opción válida.

2.3.3 Oracle SQL

Oracle SQL Developer [21] es un entorno de desarrollo integrado gratuito que proporciona Oracle para gestionar o administrar una base de datos.

Puede trabajar con multitud de ellas, entre las que se encuentra, IBM DB2, Microsoft SQL Server, MySQL o Microsoft Access. Por la dificultad de su implementación debido a la necesidad de tener un servidor dedicado con este IDE no se ha considerado viable en este proyecto

2.3.4 MySQL y phpMyAdmin

MySQL es uno de los sistemas de gestión de base de datos relacionales más conocidos. Un ejemplo de ello es que grandes empresas y organizaciones como Facebook, Wikipedia, YouTube, Google, Adobe o Alcatel Lucent confían en MySQL como sistema para administrar sus datos [22].

Este sistema de software libre desarrollado y distribuido por Oracle Corporation, es muy utilizado en aplicaciones web y frecuentemente su popularidad se ha ligado con PHP. Dispone de un motor no transaccional MyISAM que aporta una rápida lectura del contenido de la base de datos haciéndola ideal para aplicaciones con baja concurrencia en la modificación de datos. Además, es soportado por la mayoría de plataformas [23].

Parte de la popularidad que tiene se debe a su vinculación con phpMyAdmin. Una herramienta libre desarrollada en PHP cuyo fin es manejar la gestión de MySQL a través de una página web. Esto resulta una ventaja importante ya que para su funcionamiento solo es necesaria una máquina con un servidor web que soporte PHP y MySQL.

Bien es cierto que para entornos corporativos de grandes dimensiones, puede no resultar una opción no viable debido a que no cubre todo el rango de posibilidades que permite MySQL [24], en nuestra situación encaja perfectamente.

2.4 Tecnologías para aplicaciones móviles

2.4.1 Introducción.

A la hora de desarrollar una aplicación que correrá sobre un dispositivo móvil, es necesario tener en cuenta varios factores. Uno de ellos, quizás el más importante, es valorar sobre qué sistema operativo o plataforma queremos que funcione. Tener este punto claro es de vital importancia ya que, cambiar de idea una vez iniciado el proceso de desarrollo puede suponer un cambio trascendental en los plazos fijados y en el número total de horas que dure el proyecto.

Por otro lado, y teniendo en cuenta el punto anterior, es necesario valorar otros aspectos como son el rendimiento, tiempo de desarrollo, la disponibilidad o las funcionalidades que se desean implementar en la aplicación. Es aquí cuando es necesario decidir qué tipo de aplicación queremos crear. Se pueden distinguir tres tipos de aplicaciones cada una con sus ventajas e inconvenientes [25].

2.4.2 Tipos de aplicaciones

2.4.2.1 Aplicaciones nativas

Son aquellas desarrolladas en el lenguaje especifico del sistema operativo o la plataforma, Esto significa que si queremos que nuestra aplicación funcione en varias plataformas como iOS, Android y Windows Mobile será necesario desarrollar la misma aplicación en tres versiones distintas.

Aunque esto puede parecer un trabajo laborioso, analizando las principales ventajas y desventajas y teniendo claro los factores anteriormente expuestos, la decisión debería estar más clara.

Entre las ventajas más relevantes se encuentran, por una parte, la posibilidad de acceder a todas las funcionalidades del teléfono como son la cámara, los contactos, GPS, acelerómetro… Además, al tratarse de una aplicación

desarrollada para una sola plataforma, permite sacar más partido a los recursos del teléfono como la CPU, memoria o consumo de batería.
Por otro lado, al instalarse en el sistema de almacenamiento local del dispositivo, le aporta seguridad y la posibilidad de acceder a ella sin disponer de conexión a internet.

Como desventajas, destaca el coste y tiempo de desarrollo si es necesario crear varias versiones de la aplicación o el mantenimiento de esas versiones.

2.4.2.2 Aplicaciones Web Móvil

El desarrollo de aplicaciones web para dispositivos móviles se basa en el desarrollo de páginas web que son optimizadas para ser visualizadas en las pantallas de estos dispositivos. Para acceder a ellas, basta con disponer de un navegador instalado.

Aunque no debería tratarse como una aplicación móvil, en algunos casos puede ser una buena opción a tener en cuenta en función de los requisitos que se contemplen. Se trata de aplicaciones desarrolladas en HTML, CSS y JavaScript que se visualizarán de forma idéntica en todos los dispositivos dependiendo de la resolución de la pantalla. Principalmente requieren de conectividad a internet aunque pueden contar con una cache local y almacenamiento de datos para el funcionamiento offline, no obstante, están pensadas para terminales con acceso a internet.

En cuanto a su coste y al tiempo de desarrollo es bastante reducido al reutilizar el código fuente para todos los sistemas operativos. Otro aspecto a tener en cuenta es su forma de distribución, a diferencia de una aplicación nativa que necesita de un tiempo de aprobación tras cualquier modificación o actualización, en este caso es inmediato. No obstante, no es posible hacerlo a través de los canales oficiales como Apple Store o Android Market

Los inconvenientes principales con los que nos encontramos en este tipo de aplicaciones seria la integración con los componentes nativos al venir determinados por el navegador web, la fluidez de la aplicación y la necesidad de acceso a Internet.

2.4.2.3 Aplicaciones híbridas

Una aplicación híbrida, como su nombre indica es aquella que mezcla características de una aplicación web móvil y una aplicación nativa.
Este tipo se empezó a hacer popular gracias al framework Apache Cordova que permite desarrollar aplicaciones móviles que, por un lado utilizan las APIs nativas que permitan acceder a las distintas funcionalidades propias de cada sistema operativo y que por otro lado emplea tecnología web para el desarrollo de la interfaz de la aplicación [26]. Un ejemplo de estas aplicaciones son Instagram, LinkedIn o Facebook.

Esta opción puede resultar muy interesante cuando queremos crear una aplicación que soporte varias plataformas pero que también necesite hacer uso de las características del sistema operativo y el terminal ya que nos permitiría reutilizar el código empleado en el diseño de la interfaz, limitándonos a la lógica de la aplicación al consumo de los componentes de la plataforma a través de sus APIs.

También es cierto que el rendimiento y la experiencia de usuario no pueden alcanzar los niveles de una aplicación nativa.

En las siguientes tablas se muestra una comparativa entre una aplicación híbrida y una nativa destacando los principales factores a tener en cuenta.

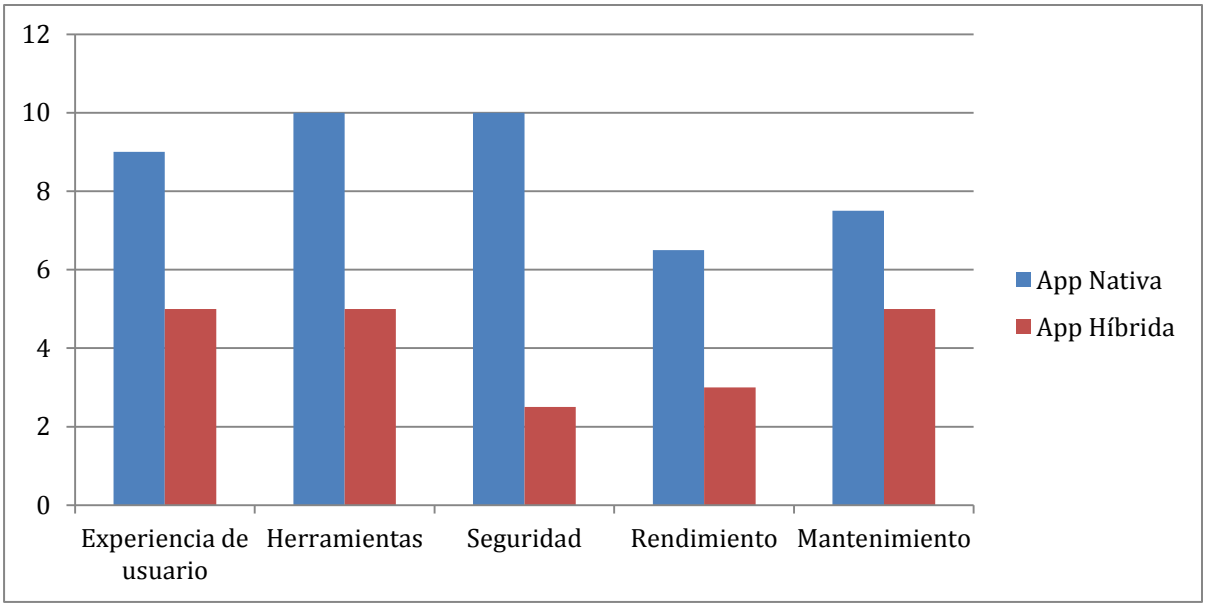

Tabla 4. Comparativa entre aplicaciones nativas e híbridas para dispositivos móviles. Factores 1

Fuente: Comparativa de aplicaciones para dispositivos móviles. Accensit. [25]

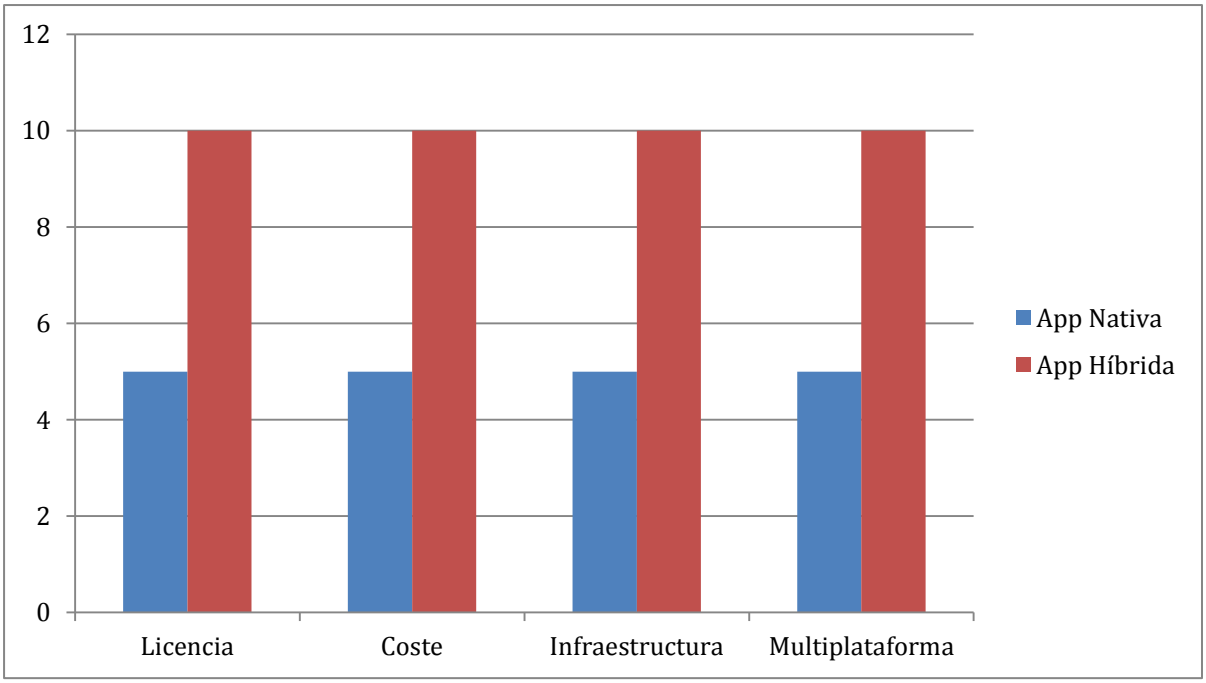

Tabla 5. Comparativa entre aplicaciones nativas e híbridas para dispositivos móviles. Factores 2

Fuente: Comparativa de aplicaciones para dispositivos móviles. Accensit. [25]

2.4.3 Plataformas

2.4.3.1 Android

Android es el sistema operativo más usado a día de hoy con una cuota de mercado que supera en algo más del doble a sus dos competidores más cercanos, Windows Mobile e iOS [27].

El sistema operativo de Google que se caracteriza por ser abierto y estar disponible para cualquier fabricante interesado en utilizarlo en sus dispositivos, añade la posibilidad de que cada uno de ellos incorpore su propia capa sobre el original de tal forma que se le dote cierta experiencia de usuario que no tenga que ver con la deseada por Google.

Esta versatilidad a la hora de tener distintos terminales con el mismo sistema operativo conlleva un decremento en la robustez de la plataforma cuando lo comparamos con otras como iOS.

Bien es cierto que cuando disponemos de un Smartphone de buenas características, este punto se hace menos notorio destacando aspectos como la posibilidad de personalizar el dispositivo o no tener que ceñirte a un grupo reducido de terminales, o a la amplia gama de aplicaciones que se encuentran en el Android Market al tratarse de la plataforma cuya comunidad cuenta con el mayor número de desarrolladores.

Por otro lado, a la hora de desarrollar una aplicación para Android es posible hacerlo programando con distintos lenguajes como C/C++, Java o C#.

2.4.3.2 Windows Phone

Windows Phone es el nuevo sistema operativo para dispositivos móviles desarrollado por Microsoft. A pesar de haber creado anteriormente Windows Mobile, tras su lanzamiento en 2010, nada tenía que ver ya con su predecesor. Hasta hace unos años, a la hora de comprarte un terminal móvil, la decisión estaba entre un sistema operativo Android o iOS, pero desde el retroceso de BlackBerry y el acuerdo entre Microsoft y Nokia con el objetivo de reinventarse, ha llevado a esta plataforma a situarse entre los 3 primeros puestos en el número de dispositivos vendidos [27].

Para el desarrollo de aplicaciones nativas para Windows Phone están soportados los lenguajes de programación C# y Visual Basic .NET.

2.4.3.3 iOS

IOS es el sistema operativo que la empresa Apple monta en sus terminales desde su lanzamiento en 2007. Aunque en un principio era desarrollado solamente para IPhone, poco después se incluyó en el resto de dispositivos móviles de la compañía como son el iPod Touch, el IPad y el Apple TV [28].
A diferencia de otras plataformas como Android, Apple no permite la instalación de su sistema operativo en hardware de terceros lo que le otorga cierta exclusividad y le permite desarrollar un sistema operativo que encaje perfectamente con todos sus productos. Esto hace que en principio su sistema operativo sea más estable.

También es necesario destacar, que el elevado precio de sus terminales y ese grado de exclusividad hace que el desarrollo de aplicaciones para esta plataforma sea inferior a la de sus competidores.

Para el desarrollo de estas, es necesario disponer de un equipo de la marca con el entorno de desarrollo *Xcode* y tener conocimiento de alguno de los lenguajes de programación que soporta. *Objective-C*, *Cocoa*, o el reciente Swift.

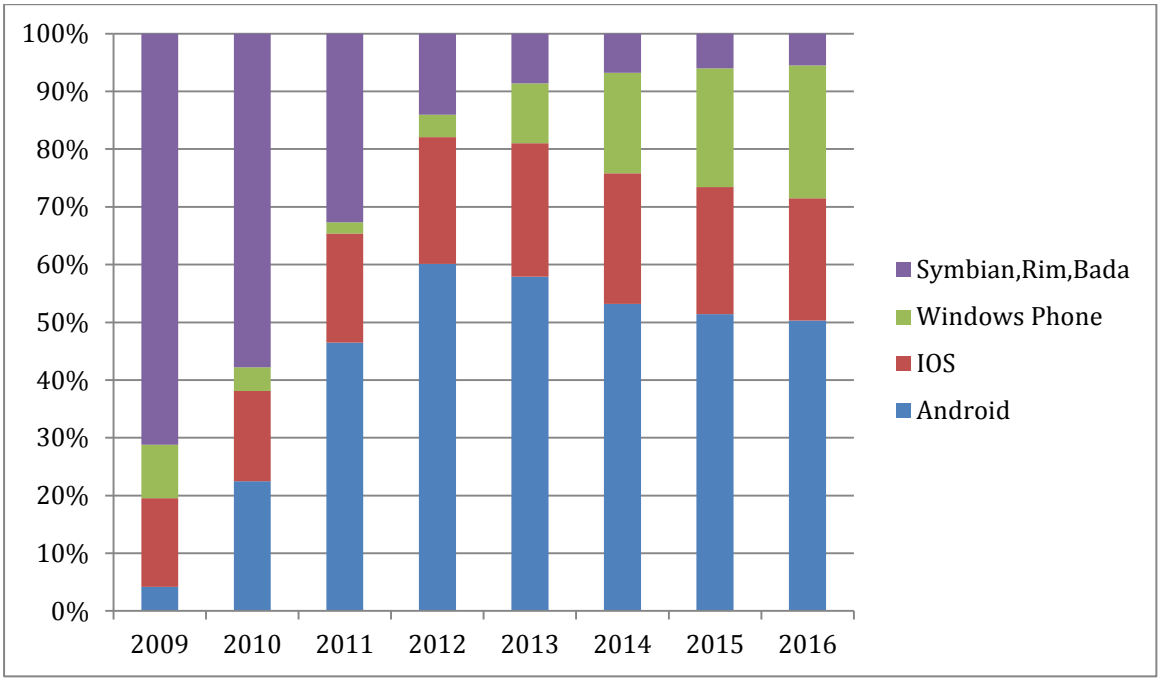

Tabla 6. Predicción cuota de mercado para ventas de Smartphone por sistema operativo.

Fuente: Gartner, Mobile Devices by Open Operating System, Worlwide, 2009-2016, 2012 Update. [27]

2.4.8 Solución, Aplicación nativa para iOS 7.1

Tras identificar las posibles soluciones que el mercado nos ofrece, estamos en disposición de elegir la que más se ajuste a nuestros requisitos.

El proyecto, en un principio estaba planificado para desarrollarse sobre iOS existiendo ya una aplicación similar para Android y sin la necesidad de soporte para Windows Phone, por lo tanto desechamos la idea de multiplataforma. Por otra parte, si a esto le sumamos que uno de los objetivos de este trabajo era aprender un nuevo lenguaje de programación con la posibilidad de hacer un aplicación que destaque por su rendimiento, la mejor solución es desarrollar una aplicación nativa, donde todo el código sea el propio de la plataforma, y que sea soportada por dispositivos móviles con sistema operativo iOS. La versión sobre la que se trabajará es iOS 7.1 al ser la más reciente durante el desarrollo de la aplicación.

2.5 Herramientas de control de versiones

Cuando desarrollamos una aplicación, independientemente de si está destinada para dispositivos móviles, ordenadores, o cualquier otro dispositivo electrónico, resulta muy aconsejable disponer de un mecanismo que nos permita llevar un control sobre los distintos cambios que se llevan a cabo en los elementos que forman parte de la aplicación, entiéndase ficheros de código, de configuración, documentos …
Estas herramientas se conocen como sistemas de control de versiones, y nos permiten entre otras muchas cosas:

- Almacenar, ya sea en local o un servidor, el proyecto o parte de él y tener una copia de seguridad en caso de pérdida de datos. Aunque esto no quita que se utilicen otras medidas de respaldo.
- La posibilidad de trabajar con otros desarrolladores en paralelo sobre un mismo proyecto si se trata de un sistema centralizado o distribuido.
- Realizar un histórico sobre ficheros comparando distintas versiones.
- Recuperar código que se ha perdido tras realizar modificaciones.

Entre los sistemas de control de versión se pueden diferenciar tres tipos en función de donde residan los repositorios [29]. Estos repositorios son los lugares donde se almacenan los ficheros y se registran los cambios de la aplicación.

Una de las opciones, quizás de las más básicas, se basa en tener el repositorio en la misma máquina donde se está desarrollando la aplicación. Un sistema de control de versiones local.
La principal ventaja reside en la falta de necesidad de conexión a internet a la hora de acceder a este. Sin embargo, esta opción hace imposible la colaboración dentro de un equipo o la imposibilidad de trabajar en máquinas diferentes.

Otra alternativa, consiste en la disposición de un solo repositorio situado en un servidor. Un sistema de control de versiones centralizado. Esto hace posible que varios programadores trabajen sobre el mismo proyecto compartiendo ficheros, unificando código a través de “merges” o descargarse las últimas versiones. Por otro parte, al estar todo centralizado, por parte, de los administradores es más cómodo llevar un registro de las tareas asignadas a cada participante. Sin embargo no todos son ventajas ya que la caída de este servidor, supone la imposibilidad de acceso al repositorio.

Por otro lado, en la actualidad se ha vuelto muy popular otro tipo de sistema donde, además del repositorio que reside en el servidor, cada participante dispone en local de uno propio. Esto hace posible que si se produce una pérdida de datos en el servidor, cualquiera de los otros repositorios pueda replicar su contenido en él. Para ello, cuando un usuario se descarga una versión, esta se replica de tal forma que tanto el repositorio local como el entorno de trabaja disponen del mismo contenido.
Al igual, un desarrollador puede subir varias versiones de código antes de que todo sea subido al repositorio remoto.

A continuación se entra en detalle en dos de los sistemas de control de versiones más utilizados. *Subversion* y *Git* [30].

2.5.1 CVS, Apache Subversion

Subversion o SVN es un software libre desarrollado por Apache que cubre las principales características que un sistema de control de versiones podría tener.

Se basa en un sistema centralizado donde desde un principio se promueve la idea de trabajo en equipo siendo hasta hace poco tiempo, una de las herramientas más usadas en su campo.

A pesar de haber trabajo con él, y estar disponible para Mac y *Xcode*, me he decantado por la siguiente alternativa debido a su reciente alzamiento en el mercado y al sistema que emplea a la hora de gestionar los cambios.

2.5.2 Git

Git, es un sistema de control de versiones distribuido, libre y de código abierto que se ha impuesto fuertemente en el mercado.
Debido a este impulso, y a su desarrollo más reciente, cuenta con alguna ventaja importante frente a Subversion. Entre ellas, como se ha comentado anteriormente, se encuentra la posibilidad de tener un control total sobre el repositorio al disponer una réplica de él en la máquina loca. Es más rápido que su competidor e incorpora medidas de seguridad como por ejemplo el uso de funciones has SHA.

Por otra parte, cuando se comienza con esta aplicación después de haber usado otras, si se decide trabajar a través de un cliente como puede ser *SourceTree*, el aprendizaje y el dominio sobre la aplicación puede ser costoso debido a las nuevas funcionalidades que incorpora y a la idea de disponer de distintos repositorios y ramas donde subir los ficheros.

No obstante, tras probar su funcionamiento en Xcode y ver su sencillez y comodidad, será la herramienta utilizada durante el desarrollo de este proyecto.

2.6 Servicios web

Para implementar algunas funcionalidades como son la autenticación de usuarios o el registro de datos desde la aplicación móvil, es necesario que esta consuma dos servicios web. Para ello se ha optado por servicios Rest donde todos los parámetros van contenidos en el método POST de HTTP. Posteriormente, la respuesta se recogerá en la propia aplicación en formato JSON.

3 Desarrollo del proyecto: HepAPPtology

3.1 Contextualización, la hepatología y aplicaciones eHealth

Entre los diferentes tipos de cáncer que se pueden presentar hoy en día, uno de ellos, y objeto de trabajo en este proyecto es el cáncer de hígado, en particular el hepatocarcinoma celular, representando este el 80-90% de los tumores hepáticos malignos [31].

Hasta hace pocos años, el diagnóstico de esta enfermedad se realizaba ya en fases avanzadas cuando el paciente presentaba claros síntomas relacionados con el tumor, y habiendo alcanzado este gran tamaño impidiendo así la aplicación de un tratamiento curativo. Sin embargo, la introducción de técnicas clínicas como la ecografía u otras de imagen, sumado al seguimiento de los pacientes a través de diversos programas, ha permitido el diagnóstico en un estado más temprano y la posible aplicación de un tratamiento de carácter curativo.

Es en todo este marco donde se sitúa la **hepatología** como subespecialidad médica que se dedica al estudio y tratamiento de enfermedades del hígado [32].

Por otro lado, y ya centrándonos más en el contenido desarrollado de este trabajo, es necesario hablar sobre la **eHealth**. La Organización Mundial de la Salud, (OMS) lo define como “El empleo de la información y tecnologías de la comunicación para un mejor control de la salud” [33], y es que, este término puede englobar muchos aspectos. Desde un servicio telefónico para dar asistencia médica, dispositivos electrónicos como medidores para evaluar la tensión arterial hasta sistemas incorporados en Smartphone capaces de mejorar la adherencia del paciente a los programas de evaluación y control facilitados por centros para poder así ayudar a la prevención de futuros acontecimientos que pudieran

desencadenar sucesos innecesarios como el ingreso hospitalario o fatales como una posible defunción. Un ejemplo de esta aplicación es **HepAPPtology**.

3.2 Beneficios de la aplicación

HepAPPtology es una aplicación eHealth para dispositivos móviles, en particular para IPhone y IPad, cuyo objetivo es realizar un control o seguimiento sobre una enfermedad recientemente diagnosticada a través de diversas técnicas médicas como son analíticas, ecografías…

Esta aplicación, en concreto sobre el hepatocarcinoma celular, no es capaz de diagnosticar la enfermedad sin resultados previos obtenidos con otras técnicas sin que ya evidencien la presencia de este tumor.
No obstante, el uso de estos datos por parte de la aplicación permite obtener a raíz del empleo de diversos algoritmos, resultados que no serían fácilmente realizables sin un sistema computarizado. Un ejemplo de estos datos son una escala de severidad de la enfermedad, un posible tratamiento, la esperanza de vida del paciente, el volumen total de tumoración, o diversos sistemas que permiten evaluar el grado de seriedad de la enfermedad.

3.3 Descripción técnica

Una vez situada la aplicación en un marco contextual y expuesto los principales beneficios que se esperan con el desarrollo de ella, se detallarán en esta sección los distintos aspectos técnicos que se han tenido en cuenta y llevado a cabo durante el desarrollo de la aplicación.

3.3.1 Dinamismo de la aplicación

Uno de los objetivos secundarios que se querían cumplir con el desarrollo de esta aplicación, era la idea de crear una cuyo contenido sea lo más dinámico posible.

Para ello se optó por definir todo el contenido, textos, posibles imágenes, títulos, listas, en un fichero o ficheros HTML que se encontrará alojado en un servidor web para desde la aplicación, descargarse este contenido, parsearlo e insertarlo en su lugar correspondiente.

Esta funcionalidad podría conllevar a la necesidad de tener acceso a internet cada vez que se inicia la aplicación. Sin embargo, esto puede llegar a tener sentido dado que con los resultados obtenidos se espera que queden registrados en una base de datos. No obstante, si esta opción no se requiere, es posible acceder a ella en modo offline siempre que se haya accedido una primera vez con conexión.

Esto es así gracias a que una vez descargado el fichero con el contenido, este se almacena en local de forma permanente hasta que se detecte que la versión que se encuentra hospedada en el hosting es más reciente. En este caso se descarga la nueva y se reemplazada por la anterior.

En el caso en el que no volvamos a acceder con conexión a internet, siempre se hará uso de la misma.

Para hacerse una idea de este funcionamiento, se muestra a continuación unas líneas del fichero con parte del contenido para posteriormente comentar brevemente el framework empleado para su parseo.

```
<div class="CalculatorView">
    <p des="titleView">RX Calculator</p>
<p>Pacient:</p>
<p>Clinical cuestions</p>
<p>Laboratory values</p>
<p>Diagnostic imaging</p>

<p des="titleAlert">Atention</p>
<p des="messageAlert">You must fill all the inputs</p>
<p des="cancelButtonAlert">OK</p>
<p des="sendButton">Send</p>

</div>
```

Ilustración 1. Contenido dinámico de la aplicación

En la captura anterior se muestra el contenido que aparecerá en la vista correspondiente al menú con las opciones de la calculadora.

Para el parseo de este código hacemos uso del framework Hpple que nos permite parsear contenido HTML y XML para Objective-C.

Su funcionamiento se basa, en nuestro caso, en buscar dentro del fichero la etiqueta “div” con la clase “CalculatorView” y recoger todas las cadenas contenidas entre etiquetas “ ”.

Los atributos “des” no tienen ninguna utilidad de código, únicamente nos aportan una descripción del contenido asociado. Por ejemplo, la descripción “titleView” nos indica que “RX Calculator” será el contenido que se mostrará como título de la

vista, “messageAlert” hace referencia al mensaje que se mostrará en la posible ventana de alert que pudiera aparecer y “sendButton” al título que aparecerá en el botón que nos permite realizar los cálculos algorítmicos.

3.3.2 Escalabilidad

El disponer de una aplicación cuyo contenido se cargue de forma dinámica le aporta cierta escalabilidad. Esto se debe a que, sería posible ampliar la aplicación, por una parte, añadiendo nuevas entradas a las posibles vistas que aparecen y por otro lado, aprovechando la arquitectura y la estructura de la aplicación ya creada. Se podría desarrollar una aplicación con soporte para varias enfermedades, bastando con añadir una nueva vista de menús que permita seleccionar la enfermedad, la simple gestión del fichero con contenido a descargar que dependerá de esta, y la modificación de la lógica con los algoritmos médicos así como la necesidad de añadir o eliminar entradas de datos.

3.3.3 Soporte multilenguaje

Otra de las ventajas de obtener el contenido de forma dinámica, es la posibilidad de dar un soporte multidioma.

Para ello, partiendo de la premisa de que todo el texto que aparecerá se encuentra en un fichero, gestionando el idioma en el que se desea mostrar la aplicación, es posible en función de este, descarga un fichero u otro o parsear uno u otro a la hora de insertar el contenido.

Actualmente la aplicación soporta dos idiomas, español e inglés. Inicialmente, en la ventana de login de la aplicación es necesario seleccionarlo siendo por defecto el de la configuración establecida del sistema operativo. En el supuesto de que ese idioma no sea compatible el idioma por defecto será el inglés.

Es por ello, que a la hora de incluir nuevos lenguajes, solo es necesario por una parte crear un nuevo fichero de contenido con la misma estructura que el resto

pero con el texto en ese lenguaje, y gestionar en el inicio de la aplicación el fichero a descargar definiendo únicamente su nombre e incluyendo la nueva opción en el display que permite seleccionarlo.

En las siguientes ilustraciones se muestran dos ficheros de contenido para los idiomas inglés y español.

```
<div class="labView">
    <p des="titleView">Laboratory values</p>
<p>Bilirrubin:</p>
<p>Albumin:</p>
<p>Creatinine:</p>
<p>INR:</p>
<p>International Normalized Ratio</p>
<p>Platelets:</p>
<p>AFP:</p>

<p des="titleAlert">Atention</p>
<p des="messageAlert">You must fill all the inputs</p>
<p des="cancelButtonAlert">OK</p>
<p des="sendButton">Send</p>
</div>
```

Ilustración 2. Ejemplo de contenido dinámico en inglés

```
<div class="labView">
    <p des="titleView">Resultados de laboratorio</p>
<p>Bilirrubina:</p>
<p>Albúmina:</p>
<p>Creatinina:</p>
<p>INR:</p>
<p>Relación normalizada internacional</p>
<p>Plaquetas:</p>
<p>AFP:</p>

<p des="titleAlert">Atención</p>
<p des="messageAlert">Debe rellenar todos los campos</p>
<p des="cancelButtonAlert">OK</p>
<p des="sendButton">Enviar</p>
</div>
```

Ilustración 3. Ejemplo de contenido dinámico en español

3.3.4 Gestión de la base de datos

El sistema de gestión de base de datos empleado en esta práctica es *MySQL* usando la interfaz web de *phpMyAdmin* para administrar su contenido.

Se emplea un motor de búsqueda *MyISAM* con cotejamiento de caracteres UTF8 y collation spanish_ci con el fin de hacerlo compatible con acentos y otros caracteres como la “ñ”.

3.3.5 Base de datos, tablas

Para poder almacenar todos los resultados obtenidos así como información sobre los pacientes y profesionales médicos, es necesario disponer de un conjunto de tablas relacionadas entre sí como se muestra en la ilustración. Por otra parte, el esquema actual de tablas está pensado para una posible ampliación de la aplicación de tal forma que si se desean añadir nuevas enfermedades para su control y seguimiento, bastaría con añadir una nueva tabla por cada una de ellas relacionándola con otra de ellas, RES_GEN.

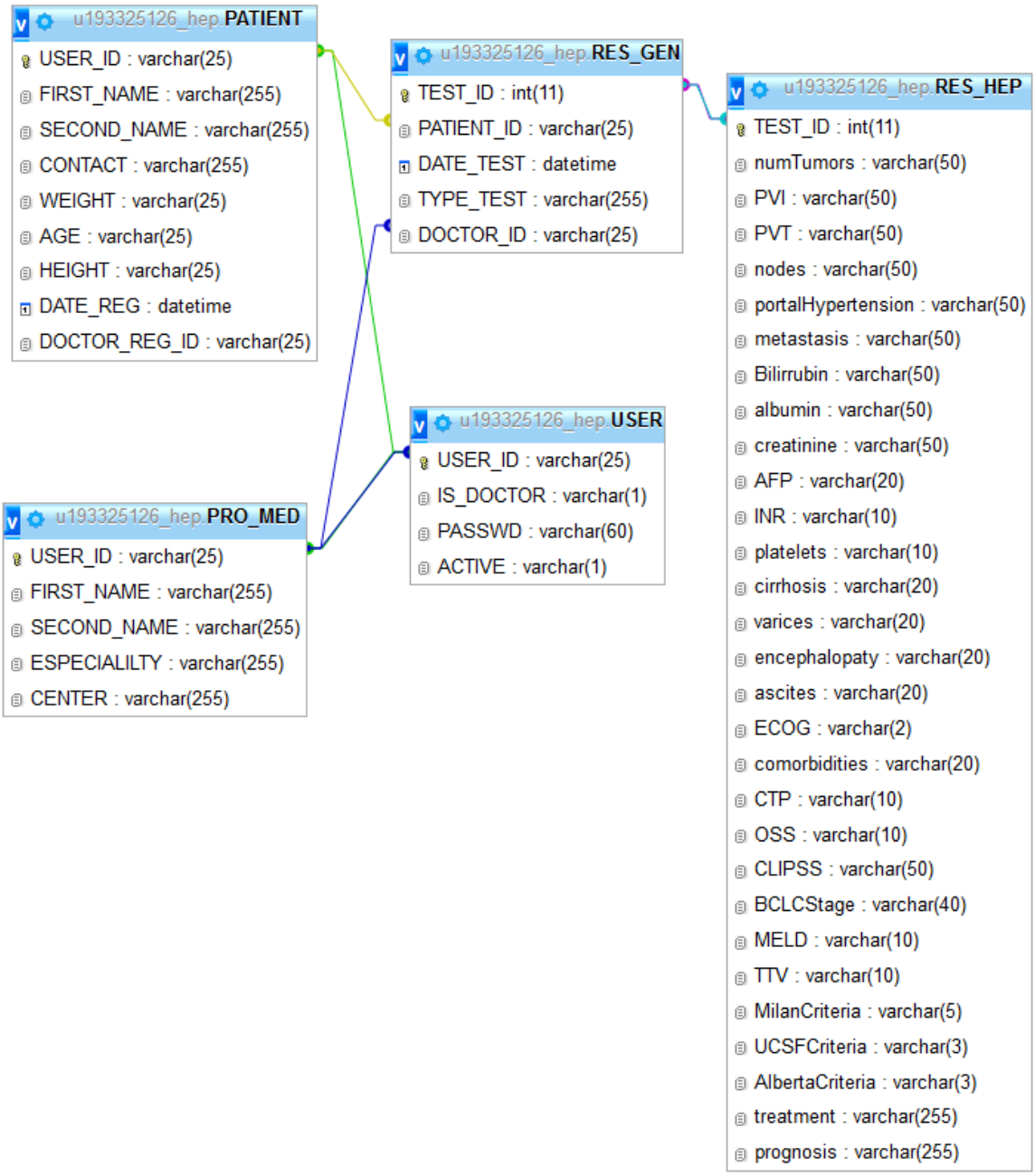

Ilustración 4. Relación de tablas de la base de datos

3.3.5.1 Tabla PRO_MED

Contiene toda la información asociada al profesional médico de la siguiente forma:

- USER_ID: Identificador de usuario. Estará formado por la inicial de su nombre seguido de las tres primeras letras de sus dos apellidos.
  Es una foreign key a USER.USER_ID.
- FIRST_NAME: Se corresponde con el nombre de la persona.
- SECOND_NAME: Se corresponde con los apellidos de la persona.
- CENTER: Centro o clínica donde trabaja el profesional médico.
- SPECIALITY: Especialidad sobre la que trabaja el profesional médico.
- CONTACT: Forma de contacto con el usuario, teléfono, email...

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado
1	USER_ID	varchar(25)	utf8_spanish_ci		No	
2	FIRST_NAME	varchar(255)	utf8_spanish_ci		No	*Ninguna*
3	SECOND_NAME	varchar(255)	utf8_spanish_ci		No	*Ninguna*
4	ESPECIALILTY	varchar(255)	utf8_spanish_ci		No	*Ninguna*
5	CENTER	varchar(255)	utf8_spanish_ci		No	*Ninguna*

Ilustración 5. Estructura de la tabla PRO_MED.

3.3.5.2 Tabla PATIENT

Contiene la información relacionada con los pacientes sobre los que se ha realizado alguna prueba

Se almacena de la siguiente forma:

- USER_ID: Identificador de usuario. El contenido de este campo lo definirá el profesional médico durante el registro del paciente en la aplicación móvil. En el caso de que no se introduzca ningún valor, se generará un número aleatorio. Es una foreign key a USER.USER_ID.
- FIRST_NAME: Nombre del usuario.
- SECOND_NAME: Apellidos del usuario.
- CONTACT: Forma de contacto con el usuario, teléfono, email...
- WEIGHT: Peso del usuario. Siempre será nulo para el profesional médico.
- AGE: Edad del usuario. Siempre será nulo para profesional médico.
- HEIGHT: Altura del usuario. Siempre será nulo para el profesional médico.
- DATE_REG: Fecha de alta del usuario.
- DOCTOR_REG_ID: Identificador del usuario. Contendrá el identificador del profesional médico, PRO_MED.USER_ID.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado
1	USER_ID	varchar(25)	utf8_spanish_ci		No	
2	FIRST_NAME	varchar(255)	utf8_spanish_ci		Sí	*NULL*
3	SECOND_NAME	varchar(255)	utf8_spanish_ci		Sí	*NULL*
4	CONTACT	varchar(255)	utf8_spanish_ci		Sí	*NULL*
5	WEIGHT	varchar(25)	utf8_spanish_ci		Sí	*NULL*
6	AGE	varchar(25)	utf8_spanish_ci		Sí	*NULL*
7	HEIGHT	varchar(25)	utf8_spanish_ci		Sí	*NULL*
8	DATE_REG	datetime			Sí	*NULL*
9	DOCTOR_REG_ID	varchar(25)	utf8_spanish_ci		Sí	*NULL*

Ilustración 6. Estructura de la tabla PATIENT

3.3.5.3 Tabla USER

Contiene la información relacionada con los usuarios que pueden acceder tanto a la aplicación móvil como al sistema de gestión de pacientes y resultados ya sean pacientes o profesionales médicos.
Se almacena de la siguiente forma:

- USER_ID: Identificador de usuario.
    - Si el usuario se corresponde con un profesional médico el contenido de este campo estará formado por la inicial de su nombre seguido de las tres primeras letras de sus dos apellidos.
    - Si el usuario se corresponde con un paciente, el contenido de este campo lo definirá el profesional médico durante el registro del paciente en la aplicación móvil. En el caso de que no se introduzca ningún valor, se generará un número aleatorio.
- IS_DOCTOR: Flag para controlar si el usuario es un paciente o un profesional médico.
    - N: paciente.
    - S: Profesional médico.
- PSSWD: Contraseña del usuario.
- ACTIVE: Flag para controlar si el usuario tiene acceso a cualquiera de los sistemas. Un paciente siempre tendrá acceso a cualquier de las dos aplicaciones una vez registrados sus resultados. Un profesional médico tendrá acceso a cualquiera de las dos aplicaciones una vez confirmado su registro por parte del administrador.
    - N: no tiene acceso.
    - S: tiene acceso.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado
1	USER_ID	varchar(25)	utf8_spanish_ci		No	*Ninguna*
2	IS_DOCTOR	varchar(1)	utf8_spanish_ci		No	N
3	PASSWD	varchar(60)	utf8_spanish_ci		No	*Ninguna*
4	ACTIVE	varchar(1)	utf8_spanish_ci		No	N

Ilustración 7. Estructura de la tabla USER

3.3.5.4 Tabla RES_GEN

Contiene un listado sobre las diferentes pruebas realizadas.

Se almacena de la siguiente forma:

- TEST_ID: Identificador de la prueba realiza, se corresponde con un número aleatorio generado del lado del servidor.
- PATIENT_ID: Identificador del paciente sobre el que se ha realizado dicha prueba. Es una foreign key a PATIENT.USER_ID.
- DATE_TEST: Fecha de realización de la prueba.
- TYPE_TEST: Tipo de prueba. Este campo está pensado para posibles ampliaciones de la aplicación. En nuestro caso solo contendrá la cadena “hepática”.
- DOCTOR_ID: Identificador del profesional médico que realizo la prueba. Es una foreign key a PRO_MED.USER_ID.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado
1	TEST_ID	int(11)			No	*Ninguna*
2	PATIENT_ID	varchar(25)	utf8_spanish_ci		No	*Ninguna*
3	DATE_TEST	datetime			Sí	*NULL*
4	TYPE_TEST	varchar(255)	utf8_spanish_ci		Sí	*NULL*
5	DOCTOR_ID	varchar(25)	utf8_spanish_ci		Sí	*NULL*

Ilustración 8. Estructura de la tabla RES_GEN

3.3.5.5 Tabla RES_HEP

Contiene toda la resultados tanto aportados a la aplicación como generados por esta sobre una prueba en concreto.

Se almacena de la siguiente forma:

- TEST_ID: Identificador de la prueba. Es una foreign key a RES_GEN.TEST_ID.
- numTumors: Número de tumores. Número entero.
- PVI: Campo PVI. Posibles valores: “Sí” o “No”.
- PVT: Campo PVT. Posibles valores: “Sí” o “No”.
- nodes: Campo nodos. Posibles valores: “Sí” o “No”.
- portalHypertension: Campo Hipertensión por vena porta. Posibles valores: “Sí” o “No”.
- metástasis: Campo metástasis. Posibles valores: “Sí” o “No”.
- bilirrubin: Campo bilirrubina. Número decimal.
- albumin: Campo albúmina. Número decimal.
- creatinine: Campo creatinina. Número decimal.
- AFP: Campo AFP. Número decimal.
- INR: Campo INR. Número decimal.
- platelets: Campo plaquetas. Número decimal.
- cirrhosis: Campo cirrosis. Posibles valores: “Sí” o “No”.
- varices: Campo varices. Posibles valores: “Sí” o “No”.
- encephalopaty: Campo encefalopatía. Posibles valores: “Ninguna”, “Grado 1-2” o “Grado 3-4”.
- ascites: Campo ascitis. Posibles valores: “Ninguna”, “Controlada” o “Resistente”.
- ECOG: Campo ECOG. Numérico del 1 al 6.
- comorbidities: Campo comorbilidad. Posibles valores: “Sí” o “No”.
- CTP: Campo CTP. Posibles valores: “A”, “B” o “C”.
- OSS: Campo OSS. Posibles valores: “I”, “II” o “III”.
- CLIPSS: Campo CLIPSS. Número entero.

- BCLCStage: Campo estado BCLC. Posibles valores: “A”, “B”, “C” o “D”.
- MELD: Campo escala MELD. Número decimal.
- TTV: Campo TTV. Número decimal.
- MilanCriteria: Campo Criterio de Milán. Posibles valores: “Sí” o “No”.
- UCSFCriteria: Campo Criterio de UCSF. Posibles valores: “Sí” o “No”.
- AlbertaCriteria: Campo Criterio de Alberta. Posibles valores: “Si” o “No”.
- treatment: Campo tratamiento. Cadena de texto.
- prognosis: Campo prognosis. Cadena de texto.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado
1	TEST_ID	int(11)			No	*Ninguna*
2	numTumors	varchar(50)	utf8_unicode_ci		No	*Ninguna*
3	PVI	varchar(50)	utf8_unicode_ci		No	*Ninguna*
4	PVT	varchar(50)	utf8_unicode_ci		No	*Ninguna*
5	nodes	varchar(50)	utf8_unicode_ci		No	*Ninguna*
6	portalHypertension	varchar(50)	utf8_unicode_ci		No	*Ninguna*
7	metastasis	varchar(50)	utf8_unicode_ci		No	*Ninguna*
8	Bilirrubin	varchar(50)	utf8_unicode_ci		No	*Ninguna*
9	albumin	varchar(50)	utf8_unicode_ci		No	*Ninguna*
10	creatinine	varchar(50)	utf8_unicode_ci		No	*Ninguna*
11	AFP	varchar(20)	utf8_unicode_ci		No	*Ninguna*
12	INR	varchar(10)	utf8_unicode_ci		No	*Ninguna*
13	platelets	varchar(10)	utf8_unicode_ci		No	*Ninguna*
14	cirrhosis	varchar(20)	utf8_unicode_ci		No	*Ninguna*
15	varices	varchar(20)	utf8_unicode_ci		No	*Ninguna*
16	encephalopaty	varchar(20)	utf8_unicode_ci		No	*Ninguna*
17	ascites	varchar(20)	utf8_unicode_ci		No	*Ninguna*
18	ECOG	varchar(2)	utf8_unicode_ci		No	*Ninguna*
19	comorbidities	varchar(20)	utf8_unicode_ci		No	*Ninguna*
20	CTP	varchar(10)	utf8_unicode_ci		No	*Ninguna*
21	OSS	varchar(10)	utf8_unicode_ci		No	*Ninguna*
22	CLIPSS	varchar(50)	utf8_unicode_ci		No	*Ninguna*
23	BCLCStage	varchar(40)	utf8_unicode_ci		No	*Ninguna*
24	MELD	varchar(10)	utf8_unicode_ci		No	*Ninguna*
25	TTV	varchar(10)	utf8_unicode_ci		No	*Ninguna*
26	MilanCriteria	varchar(5)	utf8_unicode_ci		No	*Ninguna*
27	UCSFCriteria	varchar(3)	utf8_unicode_ci		No	*Ninguna*
28	AlbertaCriteria	varchar(3)	utf8_unicode_ci		No	*Ninguna*
29	treatment	varchar(255)	utf8_unicode_ci		No	*Ninguna*
30	prognosis	varchar(255)	utf8_unicode_ci		No	*Ninguna*

Ilustración 9. Estructura de la tabla RES_HEP

3.3.6 Entorno de desarrollo, Git y pautas de programación

El entorno de desarrollo sobre el que se trabaja es Xcode en su versión 5.1

Se crea un proyecto denominado “HepAPPtology” cuyo directorio principal contendrá un conjunto de carpetas correspondientes a los distintos frameworks empleados: *AFNetworking*, *JSON*, *Reachability* y *Hpple*.

Por otro parte, de esta raíz principal colgarán también todas las clases e interfaces de las que consta la aplicación. En cuanto a la nomenclatura de estas estarán compuestas de un conjunto de letras que describen la vista a la que están vinculadas seguidas de “ViewController” como la clase que hereda.

Además, existirán dos clases, *Data* y *LogicCalculator* que guardaran datos usados en la aplicación para posteriormente registrarlos en la base de datos.

Por otra parte, para hacer uso de la herramienta que nos permite llevar un control sobre las versiones de la aplicación, Git, se dispondrá de un repositorio alojado en un servidor gratuito, *BitBucket*, mantenido por la empresa *Atlassian.*

https://abenitov@bitbucket.org/abenitov/hepapp.git

En este repositorio se crearan dos ramas, *developing* y *deploy,* manteniendo en la primera las versiones iniciales de la aplicación antes de que se le realice el proceso de *restyling* mientras que en la segunda se registrarán las últimas fases previas a la finalización de ella.

Por último, es necesario en *Xcode* añadir este repositorio y aportar los datos de acceso para poder trabajar con él.

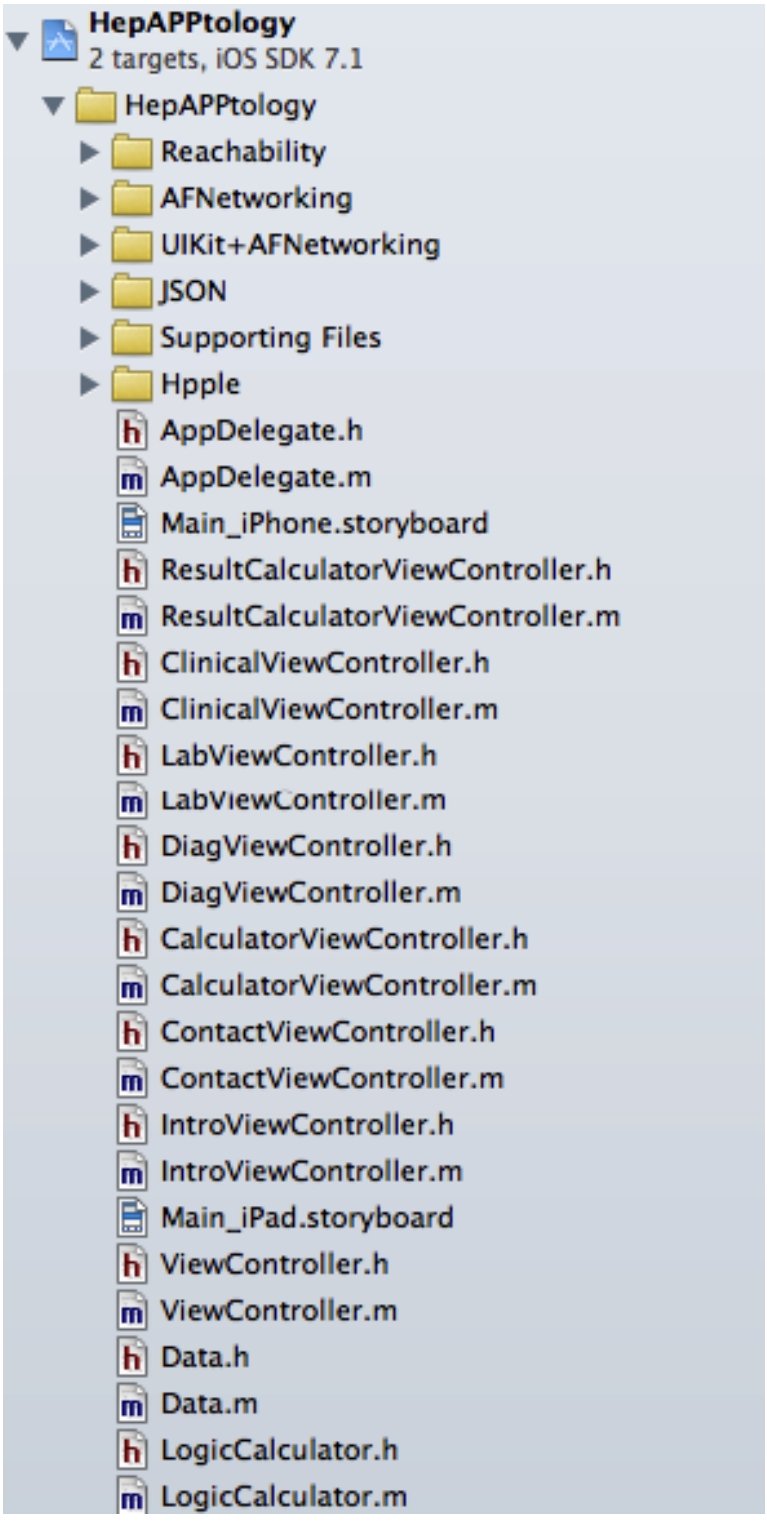

Ilustración 10. Directorio de trabajo en *Xcode*.

3.3.7 Inicio de sesiones y registro de usuarios.

A la hora de acceder al sistema de gestión de pacientes y resultados es posible hacerlo con dos roles, como usuario o como profesional médico.
Si se accede como el primero es necesario que previamente se hayan registrado sus datos en la BD. Esta acción solo se produce cuando un doctor decide registrar los resultados obtenidos a través de la aplicación móvil al no existir un formulario de registro para ellos.

Desde la aplicación móvil, el registro de pacientes se lleva a cabo de la siguiente manera:

1. El doctor introduce los datos del paciente.
2. Introduce los distintos valores médicos en las diferentes pantallas.
3. Pulsa sobre el botón que calcula los resultados.
4. Pulsa sobre el botón que registra los resultados.
5. El paciente recibe un correo electrónico con los datos de acceso a la aplicación web.

Todos estos pasos están mejor detallados en la Ilustración 11. Diagrama de secuencia, Registro de pacientes.

Por otro lado, un profesional médico solo puede acceder a cualquier de las dos aplicaciones tras haber enviado un formulario de registro y después de que un administrador valide su cuenta.

Desde la aplicación web, el registro de doctores se realiza de la siguiente forma:

1. El doctor rellena un formulario de datos.
2. El sistema valida los datos y verifica que el usuario no exista.
3. El sistema registra los datos del doctor.
4. El doctor recibe un correo electrónico indicándole que es necesario que un administrador valide su cuenta.

Todos estos pasos están mejor detallados en Ilustración 12. Diagrama de secuencia, Registro de doctores.

La validación de la cuenta por parte del administrador se basa en la modificación del flag ACTIVE de la tabla USER.

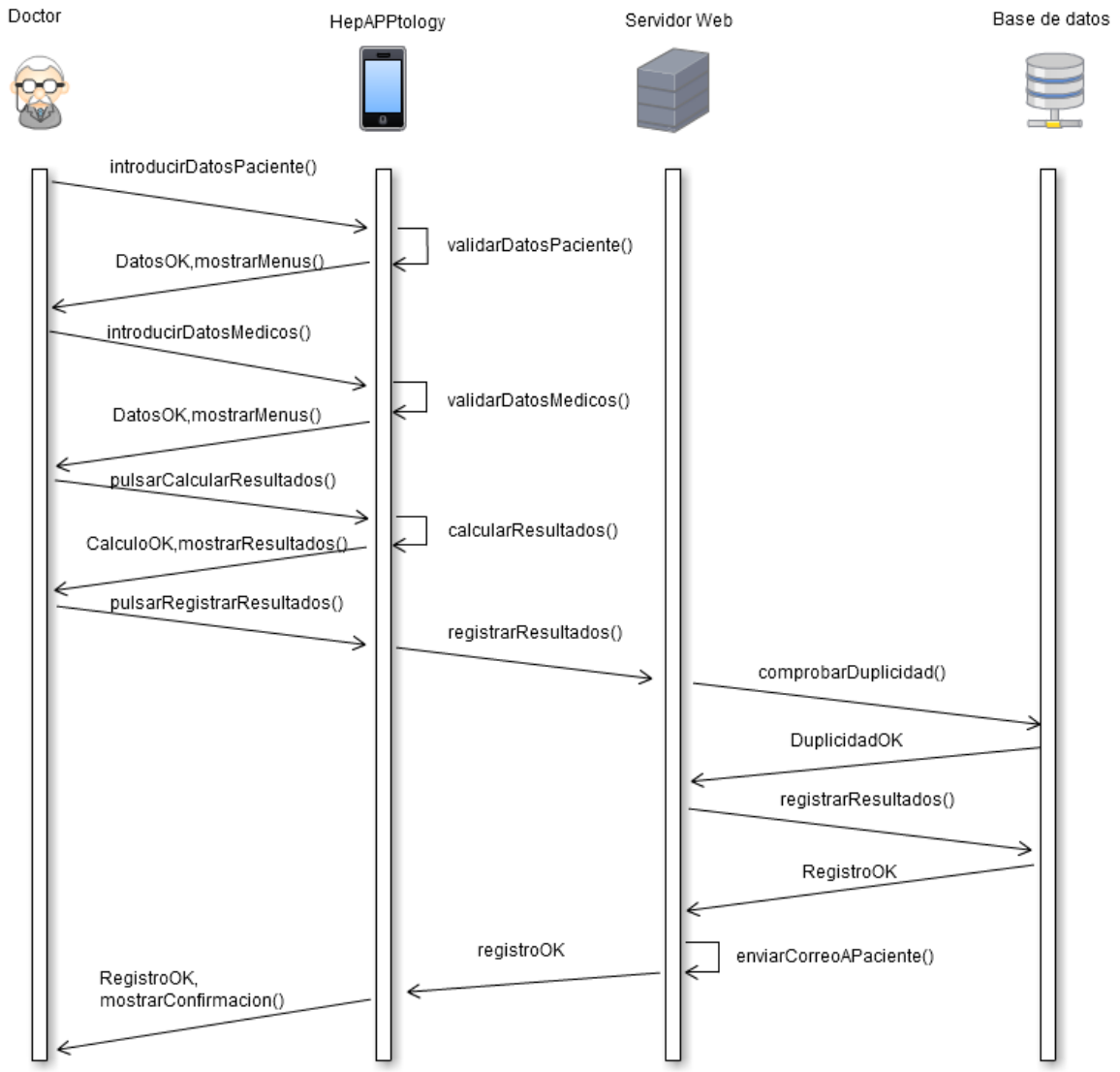

Ilustración 11. Diagrama de secuencia, Registro de pacientes.

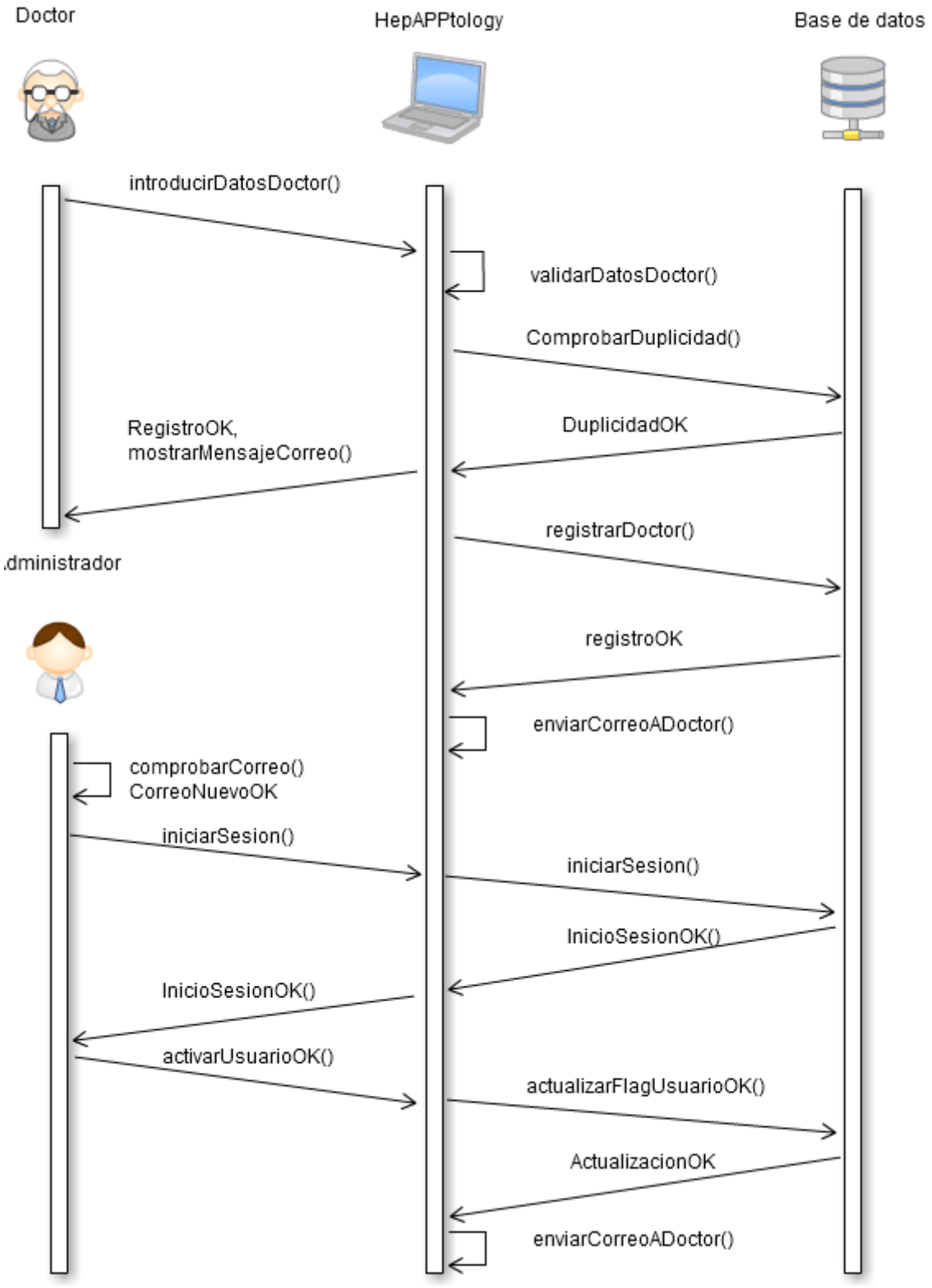

Ilustración 12. Diagrama de secuencia, Registro de doctores.

A continuación se muestran las consultas realizadas en estos pasos:

Comprobación de duplicidad de pacientes.

*"select * from USER where USER_ID = ? and IS_DOCTOR = 'N'"*

Donde "?" es el identificador de usuario recogido en el POST.

Comprobación de duplicidad de doctores.

"select USER_ID from PRO_MED where FIRST_NAME = "strtolower(trim(?))" and SECOND_NAME= = "strtolower(trim(?))""

Donde "?" son el nombre del doctor y sus apellidos recogidos en el POST.

Alta de pacientes.

"*Insert intro PATIENT values('?', '?', '?', '?', '?', '?', '?', '?', '?')"*

Donde "?" son los datos del paciente recogidos en el POST.

"*Insert intro USER values('?', '?', '?', '?')"*

Donde "?" son el identificador del paciente, el flag "N" que indica que no es un doctor, la contraseña recogida en el POST y el flag "S" indicando que la cuenta esta activa.

Alta de doctores.

"Insert intro PRO_MED values('?', '?', '?', '?', '?', '?')"

Donde "?" son los datos del paciente recogidos en el POST.

"Insert intro USER values('?', '?', '?', '?', '?')"

Donde "?" son el identificador del doctor, el flag "S" que indica que lo es, la contraseña recogida en el POST y el flag "N" indicando que la cuenta no está activa.

Una vez que el usuario, tanto el paciente como el profesional médico tiene su entrada correspondiente en la base de datos, es posible iniciar sesión.

A la hora de acceder a la aplicación web no existe ningún tipo de restricción que afecte a estos roles. El proceso es el mismo:

1. El usuario introduce sus datos de acceso.
2. Se validan y se verifica que el par usuario contraseña coincide con alguna de las entradas de la tabla USER y se verifica que el usuario tiene el flag USER.ACTIVE a “S”.
3. Se concede o deniega el acceso.

Cuando se quiere acceder a la aplicación móvil, se añado una restricción más al proceso:

1. El usuario introduce sus datos de acceso
2. Se verifica que el par usuario contraseña coincide con alguna de las entradas de la tabla USER, que el usuario tiene el flag USER.ACTIVE a “S” y que se trata de un profesional médico. USER.IS_DOCTOR coincide con “S”.

Las consultas que se generan para este inicio de sesión son las siguientes:

En la aplicación móvil:

“*Select USER_ID from USER where USER_ID = ? and PASSWD = ? and ACTIVE = ’S’ and IS_DOCTOR = ‘S’”*

Donde “?” son el identificador de usuario y contraseña recogidos en el POST.

En la aplicación web:

“*Select USER_ID from USER where USER_ID = ? and PASSWD = ? and ACTIVE = ’S’”*

Donde “?” son el identificador de usuario y contraseña recogidos en el POST.

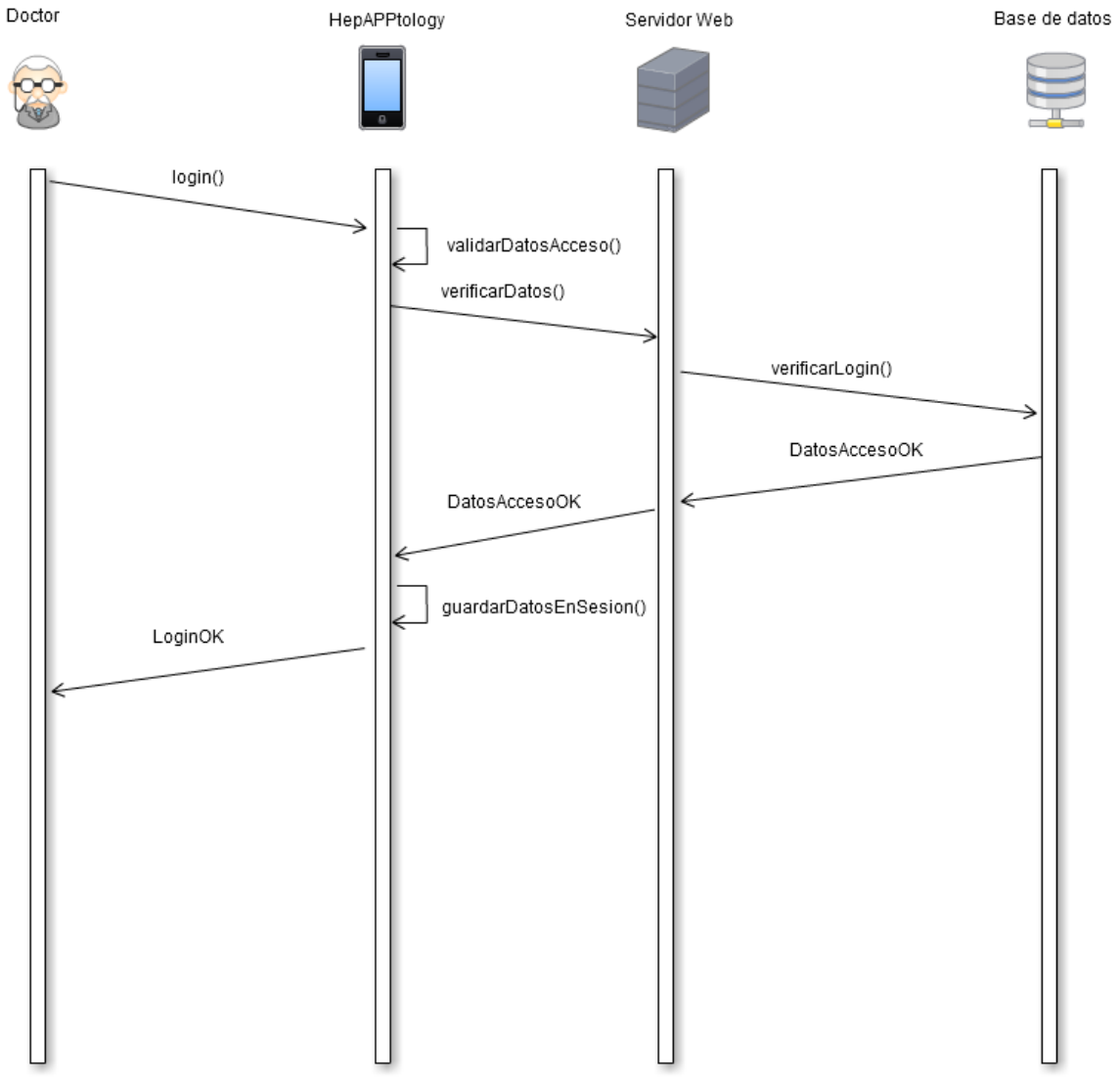

Ilustración 13. Diagrama de secuencia, Inicio de sesión desde terminal móvil.

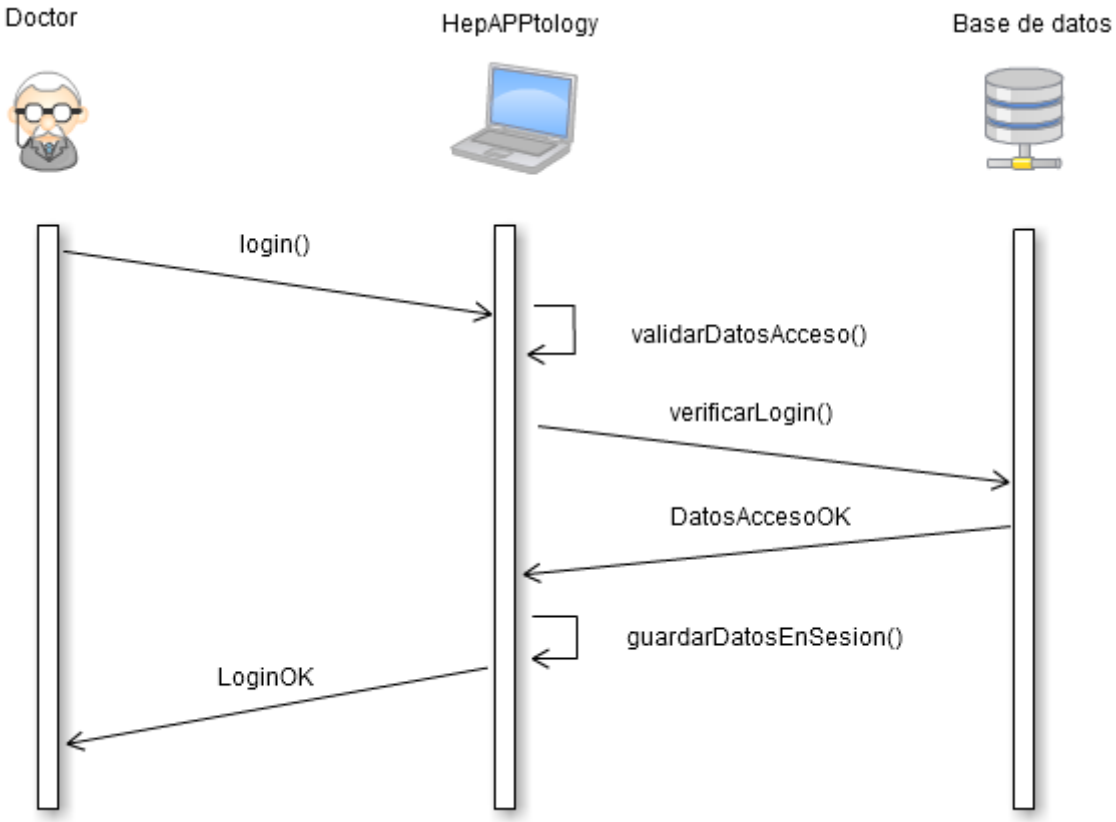

Ilustración 14. Diagrama de secuencia, Inicio de sesión desde web

La principal diferencia entre iniciar sesión desde una aplicación u otra, sin tener en cuenta los aspectos comentados anteriormente, es la necesidad de consumir un servicio web cuando accedemos con el móvil.

Para ello empleamos el framework *AFNetworking* que nos permite enviar mensajes http cuya cabecera contendrá nuestros datos de acceso.

3.3.8 Aplicación Web. Gestión de usuarios y resultados

Esta aplicación, desarrollada en PHP, HTML y CSS, permite a través de consultas contra una base de datos, mostrar tablas con la información que contienen.

En función del rol con el que accedemos, nos muestra distintas listas producto de las siguientes consultas SQL.

Si accedemos como paciente, listará todas las pruebas realizadas ordenadas por fecha permitiendo ver en detalle aquella seleccionada al cargar de forma dinámica su contenido con Ajax. El fichero donde se realiza esto es *detallePaciente.php*

La consulta que lista las pruebas realizadas es la siguiente:
*"Select * from RES_GEN where USER_ID = ? ORDER BY DATE_TEST DESC LIMIT ? , 15"*

Donde "?" son el identificador de usuario recogidos en el POST y el elemento inicial que se mostrará cuando hacemos uso de paginación. Solo se mostrarán 15 resultados por página.

La consulta que carga los datos de la prueba es la siguiente:
*"select * from ? where TEST_ID = ?"*

Donde "?" son el nombre de la tabla que contendrán los datos de la prueba y el identificador de la prueba. Ambos parámetros se envían con javascript y Ajax desde la lista de pruebas recogiéndolos en el GET.

Si accedemos como profesional médico, se listaran todas los pacientes a los que el propio doctor ha realizado una prueba. El fichero donde se realiza esto es *detalleDoctor.php.*

Para obtener estos datos se realiza la siguiente la consulta:
*"select * from PATIENT where USER_ID in*
(select distinct USER_ID from RES_GEN where DOCTOR_ID = ?)"

Donde "?" es el identificador del doctor que se ha logueado.

Una vez desplegada la lista, al pulsar sobre el paciente, se redireccionará a *detallePaciente.php* para mostrar los mismos resultados que se detallaron anteriormente.

Otro tipo de acción que se puede llevar a cabo desde esta aplicación es el acceso como administrador para la activación de usuarios.

En este caso, es necesario comentar que cuando se verifica el usuario y contraseña, antes de redireccionar se comprueba si el usuario y contraseña coinciden con dos cadenas. Si esto es así, se redireccionará a la página *detalleAdmin.php* donde se listaran todos los profesionales médicos inactivos, para, a través de un enlace, modificar su estado.

Las consultas que obtienen la lista de profesionales médicos inactivos y permiten su activación son las siguientes:

*"select * from PRO_MED where USER_ID in*
(select distinct USER_ID from USER where IS_DOCTOR = 'S' and ACTIVE = 'N')"

"update USER set ACTIVE = 'S' where USER_ID = ?

Donde en este último caso, "?" es el identificador del doctor cuyo registro se quiere activar.

En las siguientes tres ilustraciones se muestran en detalles los distintos procesos que se pueden llevar a cabo en esta aplicación una vez iniciada sesión.

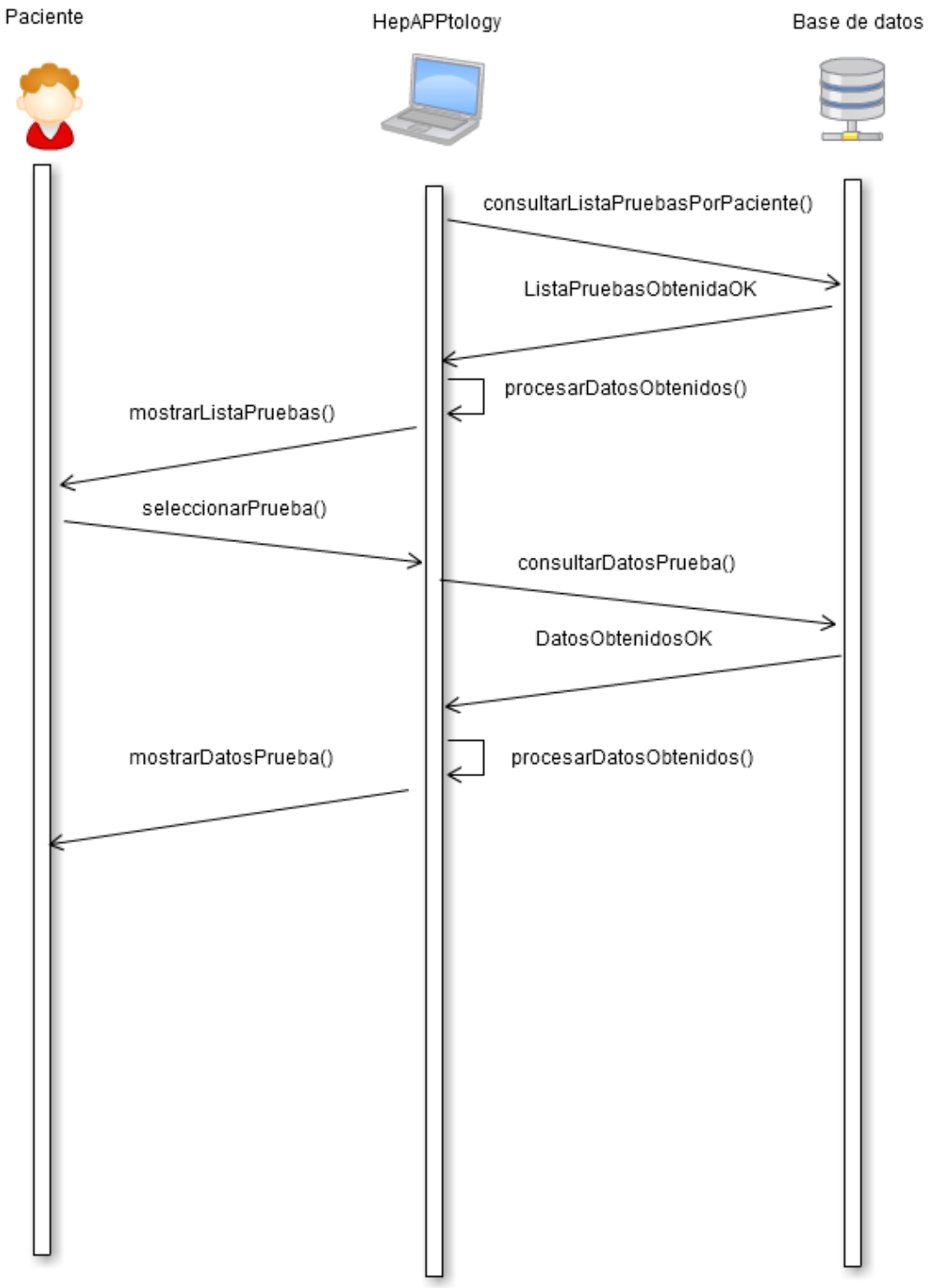

Ilustración 15. Diagrama de secuencia, gestión de resultados por paciente.

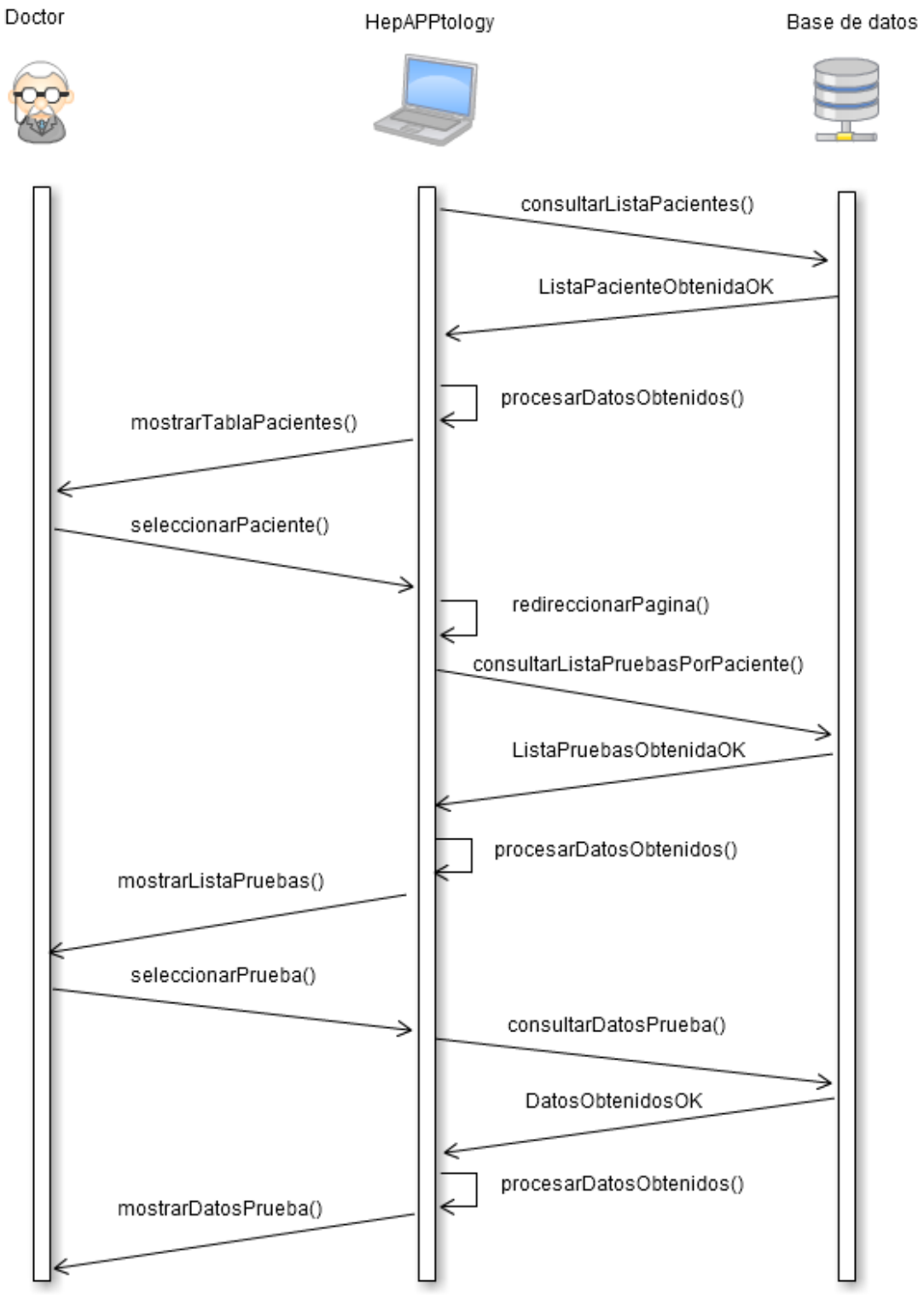

Ilustración 16. Diagrama de secuencia, gestión de pacientes por doctor.

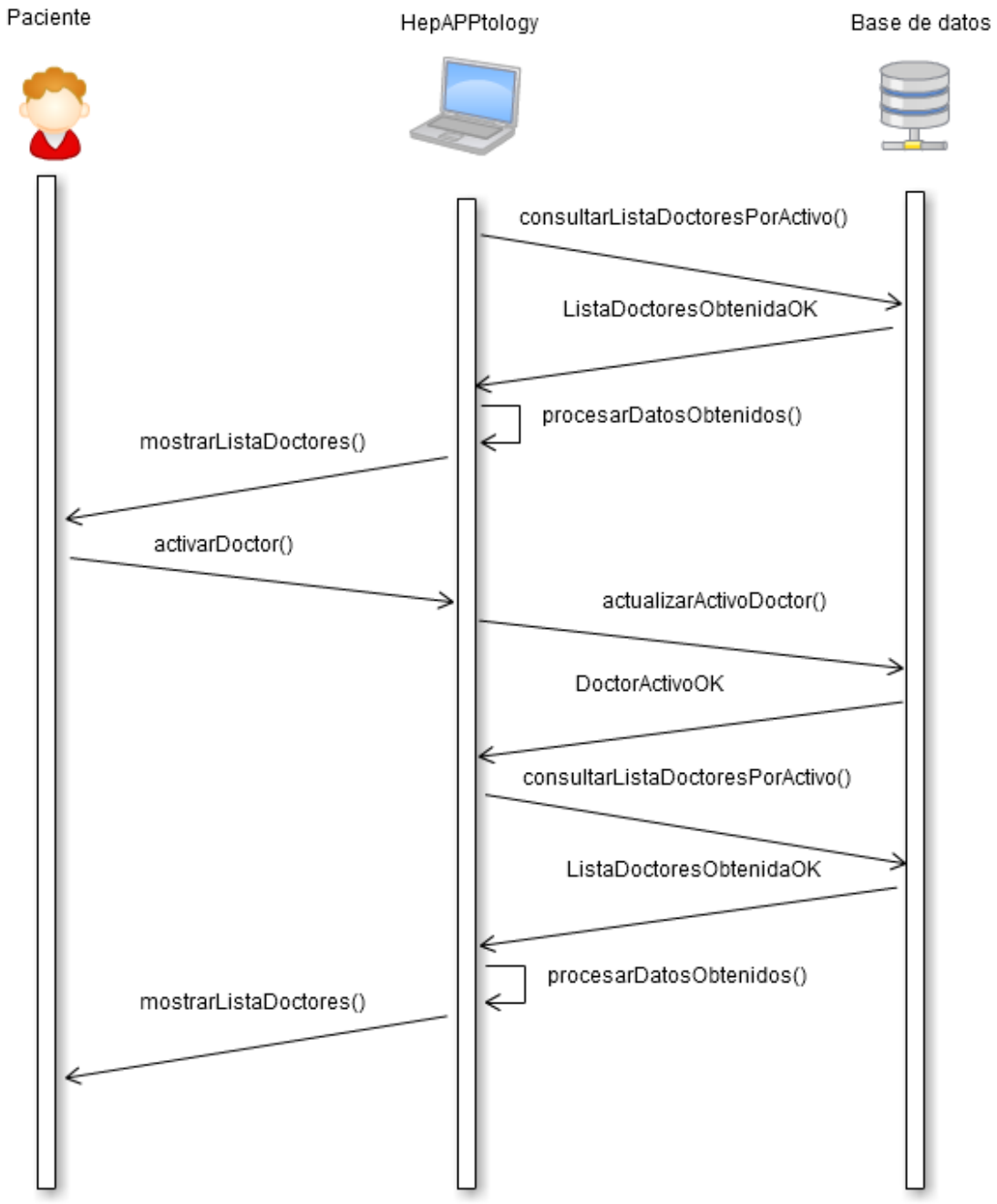

Ilustración 17. Diagrama de secuencia, gestión del registro de doctores.

3.3.9 Aplicación móvil. Gestión del contenido y el idioma

Como ya se ha comentado en los primeros capítulos de esta sección, todo el contenido de la aplicación se obtiene de un fichero html alojado en un servidor web.

Tras arrancar la aplicación, y antes de que se muestre la vista que contendrá un formulario para el acceso al resto de la aplicación, en primer lugar se comprobará que existe conexión a Internet para, en el caso de que no sea así y de que no residan los ficheros de contenido en el almacenamiento local del dispositivo, se muestre un mensaje por pantalla en forma de "alert" que nos indique lo sucedido.
Además, el botón de login no aparecerá ya que no podremos iniciar sesión sin estar conectados a Internet obligándonos así a no continuar con la aplicación. Si no disponemos de conexión a internet pero si se encuentran los ficheros en nuestro terminal, se podrá continuar con la aplicación entrando como invitado.

Si por el contrario tenemos acceso a Internet, se comprobará si los ficheros se encuentran en el almacenamiento local del dispositivo, si esto es así, se compara el tamaño de estos con los alojados en el servidor para que en caso sean distintos, actualizarlos. Si es la primera vez que se inicia la aplicación y estos ficheros no se encuentran almacenados, se descargarán todos ellos con el contenido a mostrar en todos los idiomas soportados, permitiendo que de este instante en adelante, sea posible el acceso a la aplicación sin disponer de conectividad.
Después de esto, se comprueba el idioma configurado del sistema operativo para, en función de este, mostrar los textos de "Usuario" y "Contraseña" en un idioma u otro tras parsear su código correspondiente.

Una vez hecho todo esto, se mostrará la vista con los distintos inputs y botones.

El botón de "Acceso como invitado" se mostrará en todos los casos.
El botón de "Acceso como médico" se mostrará solo cuando exista conectividad.

Acceder como invitado implica no tener que validar los datos de acceso así como la omisión del formulario con los datos del paciente y el registro en base de datos.

Todas las vistas o pantallas que se pueden ver en la aplicación se gestionan de la misma forma. Antes de que estas se muestren, en función del idioma seleccionado durante el inicio de la aplicación, se parseará un fichero u otro para después introducir su contenido en los distintos elementos que formaran la vista.

3.3.10 Aplicación móvil. Clases Data y LogicCalculator

Durante la ejecución de toda la aplicación, es necesario que algunos datos queden almacenados con el objetivo de poder gestionar algunas funcionalidades o controlar la navegabilidad de esta. Para ello, disponemos de dos clases, *Data* y *LogicCalculator* que obteniendo la instancia de su objeto en las vistas, nos permitirá hacer uso de sus atributos, acceder o registrar estos datos en la aplicación.

Por una parte, *LogicCalculator* contendrá todos los datos que el usuario introduce en los tres formularios que implementa la aplicación para el cálculo de los algoritmos. Además contendrá el método encargado del cálculo de estos algoritmos para al final, guardar también como atributos de esta clase, los distintos resultados obtenidos.

Por otra parte, la clase Data contendrá todos aquellos atributos que serán usados internamente por la aplicación para gestionar funcionalidades o permitir ciertos flujos de navegación. Además, registrará la información sobre el paciente.
A continuación se muestra una tabla con todos los atributos de esta clase y el uso que se le da a cada uno:

direccion	Contendrá el nombre del fichero con el contenido. Este campo se setea en función del idioma establecido por el usuario.
Language	Contendrá dos letras asociadas al idioma establecido por el usuario. Por ejemplo “en” o “es”.
patientID	Contendrá el ID del paciente recogido en el formulación de registro del paciente.
patientName	Contendrá el nombre del paciente recogido en el formulación de registro del paciente.
patientSecondName	Contendrá los apellidos del paciente recogido en el formulación de registro del paciente.
email	Contendrá el email del paciente recogido en el formulación de registro del paciente.
age	Contendrá la edad del paciente recogido en el formulación de registro del paciente.
Weight	Contendrá el peso del paciente recogido en el formulación de registro del paciente.
Height	Contendrá la altura del paciente recogido en el formulación de registro del paciente.
diagDone	Variable para comprar si el formulario de diagnóstico por imagen se ha completado con éxito.
labDone	Variable para comprar si el formulario de resultados de laboratorio se ha completado con éxito.
clinicalDone	Variable para comprar si el formulario de cuestiones clínicas se ha completado con éxito.
wikiTitle	Titulo asociado la vista con la información sobre un capítulo.
wikiText	Contenido del capítulo que se mostrará.
Doctor	Usuario con el que el doctor ha iniciado sesión. Contendrá “Invitado” si se accede con este perfil.

Tabla 7. Atributos de la clase Data

3.3.10 Aplicación móvil. Calculadora y algoritmos.

Una de las partes más importantes de la aplicación es el conjunto de algoritmos y cálculos de los que hace uso a la hora de obtener los distintos resultados.

Es necesario remarcar que debido a la forma de modelar estos, es necesario que los datos introducidos en la calculadora sean coherentes y consistentes. De no ser así no se podrán obtener algunos resultados como son el posible tratamiento o la prognosis mostrando entonces un mensaje en su lugar de datos no disponibles.

Por otra parte, esta parte de la aplicación, que contiene la lógica de negocio, es la única que sería necesario modificar íntegramente en el caso de que queramos evaluar cualquier otra enfermedad. Esto conllevaría la necesidad de por ejemplo de disponer de varias clases, como la comentada en el capítulo anterior "LogicCalculator" donde en cada una de ellas estén recogidos los distintos parámetros de la enfermedad asi como la gestión de los distintos algoritmos empleados.

Lo mismo sucedería a la hora de registrar estos datos en la base de datos.
A la hora de consumir el servicio web que conectará con ella, será necesario especificar una llamada por enfermedad pasándole los parámetros relacionados en cada una de ellas.

3.4 Manual de usuario

Este capítulo pretende que sirva de guía para el uso de las aplicaciones. Por una parte la aplicación web *www.hepapp.esy.es* y la aplicación móvil *HepAPPtology*.

A continuación se describen las distintas características y funcionalidades de estas sin entrar en aspectos técnicos.

3.4.1 Aplicación web. Registro de usuarios e inicio de sesión.

Si accedemos a la url que detallamos anteriormente nos conducirá a la siguiente vista:

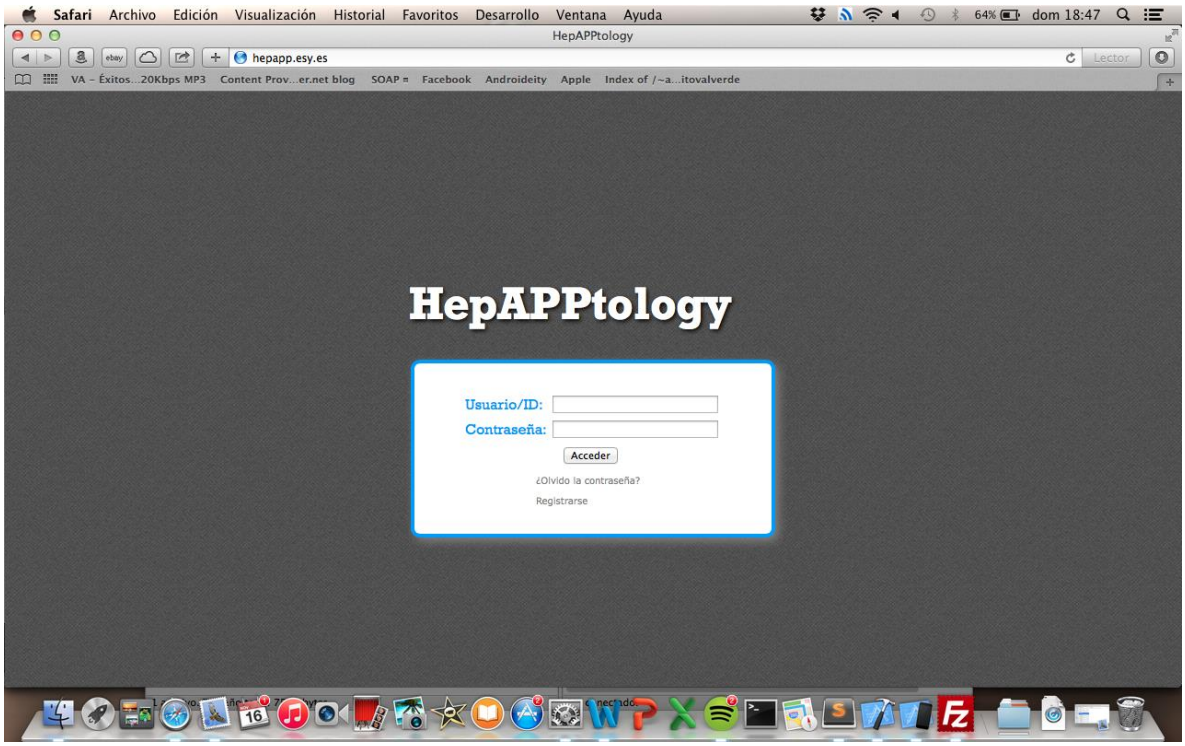

Ilustración 18. Aplicación web: pantalla de inicio de sesión

Desde ella es posible autenticarse ya sea como paciente o como profesional médico teniendo en cuenta varios aspectos:

- Para acceder como médico, es necesario previamente haberse registrado como tal en su formulario correspondiente y por otra parte, que un administrador active su cuenta.
- Para acceder como paciente, es necesario que a este se le haya realizado alguna prueba con la aplicación móvil y sus resultados se hayan registrado en la base de datos. Cada vez que se realice este proceso, el paciente recibirá un correo electrónico con sus datos de acceso a la aplicación

Si el usuario ha ingresado sus datos de forma errónea o la cuenta del médico no ha sido activada, se mostrará un mensaje indicando lo sucedido como se muestra a continuación.

Ilustración 19. Aplicación web: vista con usuarios y clave incorrectos

Si el profesional médico no se ha registrado podrá hacerlo a través del enlace “Registrarse” que le conducirá al formulario que se muestra en la siguiente captura.

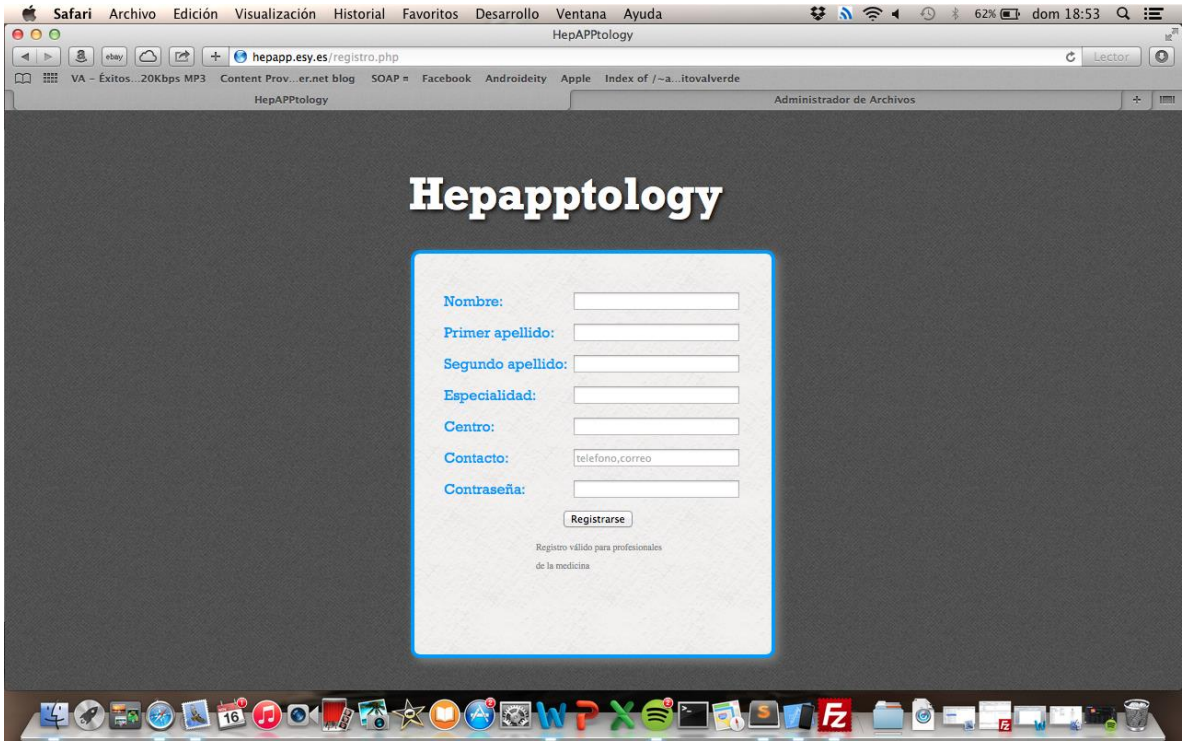

Ilustración 20. Aplicación web: pantalla de registro

Es necesario apuntar que el registro de usuarios solo está permitido para profesionales médicos. Aunque cualquier usuario podría rellenar esta información y registrarse, durante el proceso de activación, un personal autorizado se podrá poner en contacto con este para verificar sus datos a través del medio de contacto aportado.

Todos estos datos son obligatorios mostrando un mensaje de error si alguno no se ha informado, y tendrán que tener sentido. De no ser así, es posible que el administrador cancele la cuenta.

Cuando un usuario se registra, este recibirá un correo electrónico donde estarán recogidos todos los puntos anteriormente comentados.

3.4.2 Aplicación web. Gestión de pacientes.

Una vez activado el usuario e autenticado como médico, se mostrará la siguiente vista.

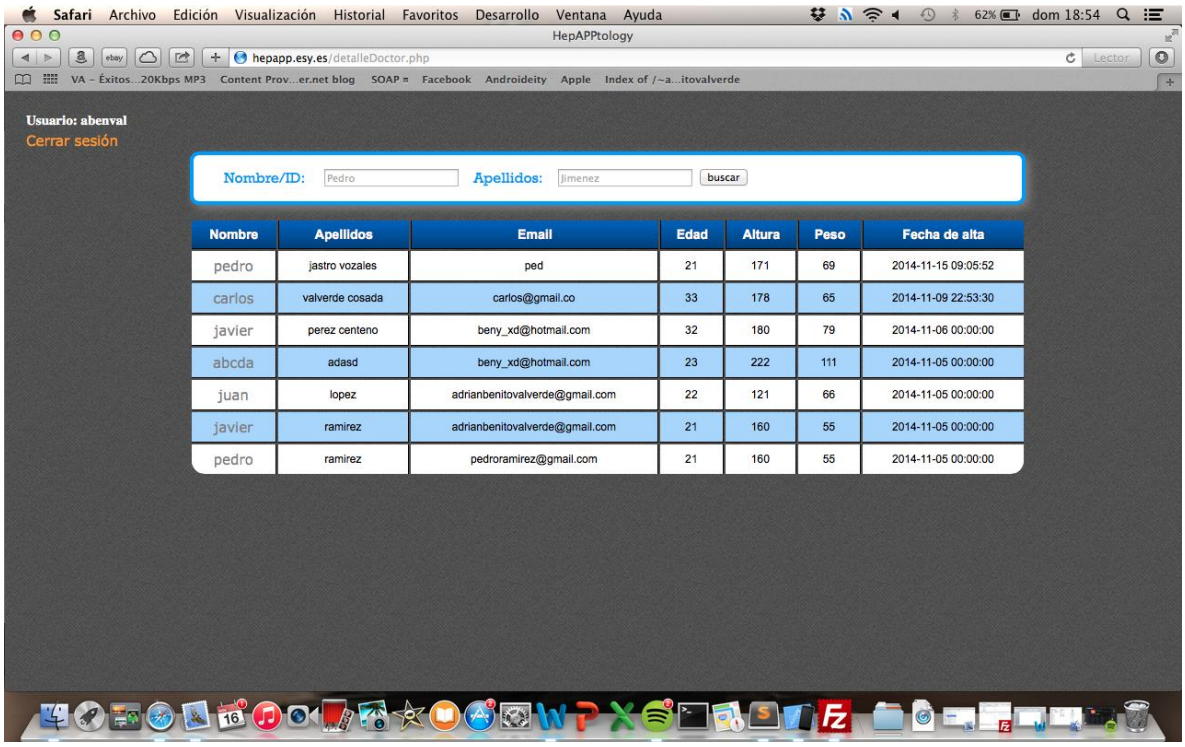

Ilustración 21. Aplicación web: pantalla con listado de pacientes

Desde ella, el profesional médico podrá buscar un paciente al que ha realizado alguna prueba a través del formulario de búsqueda. En el caso de que no querer filtrar esta información, se mostrará una lista con todos los pacientes a los que ha prestado servicio.

Si se desea consultar las pruebas realizadas sobre este paciente, basta con pulsar sobre el nombre de usuario para acceder a la información de él.

3.4.3 Aplicación web. Gestión de pruebas.

Al acceder a la información de cierto paciente a través del listado de pacientes, o accediendo como paciente a la aplicación, se mostrará la siguiente lista.

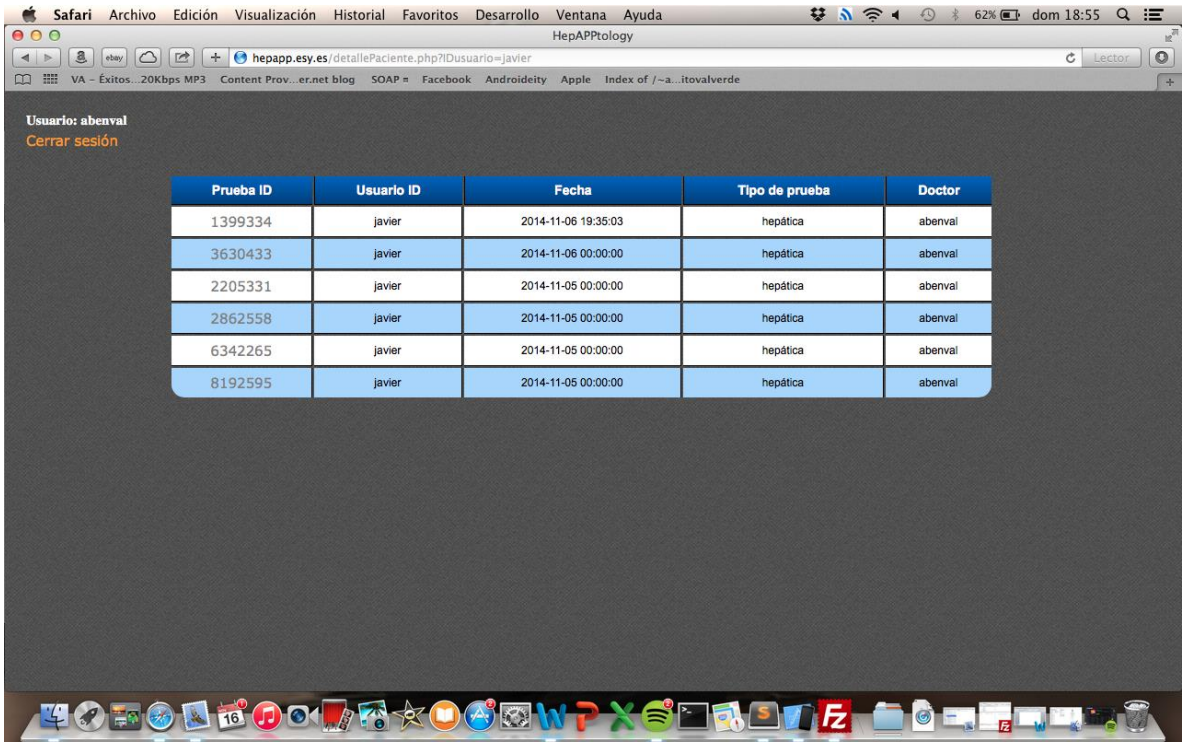

Ilustración 22. Aplicación web: pantalla con listado de resultados

En ella, aparecerá una tabla con todas las pruebas que se han realizado sobre esa persona indicando el ID de la prueba, la fecha de la prueba, el tipo de la prueba, y el médico que la realizó.

Si se quiere obtener más información sobre esta, bastará con pulsar sobre el ID de la prueba mostrándose todos los datos aportados y resultados de la misma.

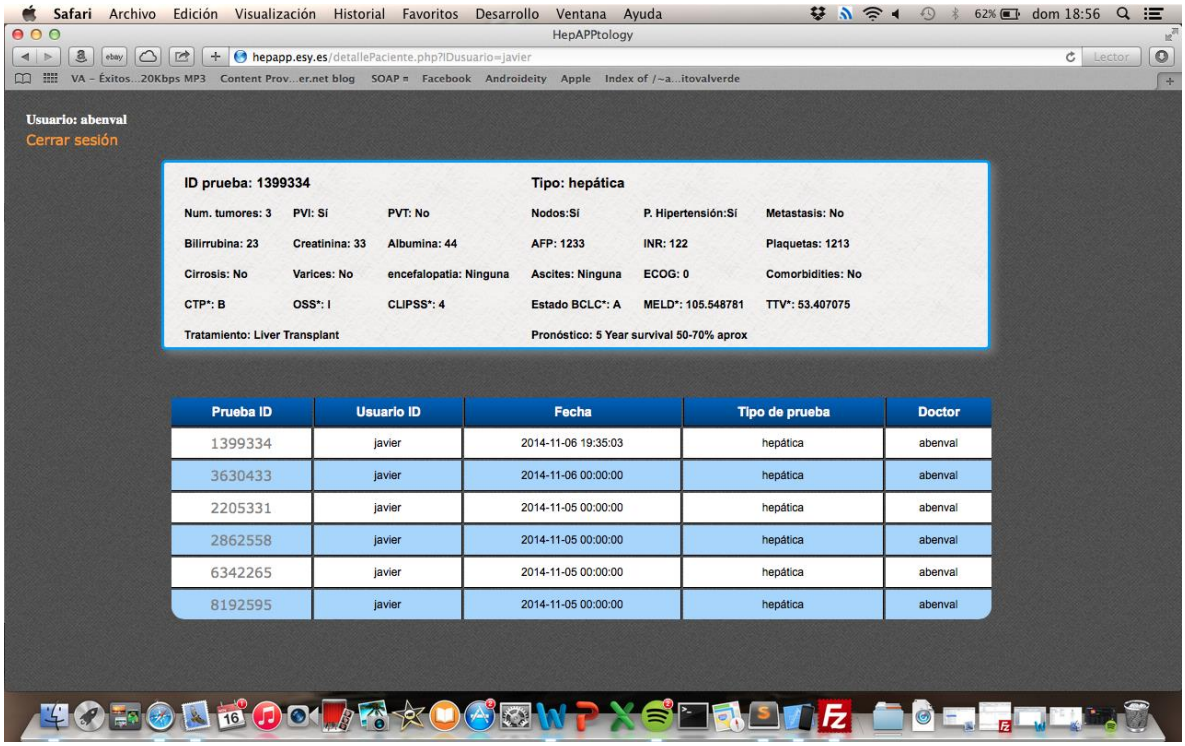

Ilustración 23. Aplicación web: pantalla con los datos de una prueba

3.4.4 Aplicación web. Panel de Administración.

Si somos el administrador y accedemos como tal a la aplicación se nos mostrará la siguiente vista.

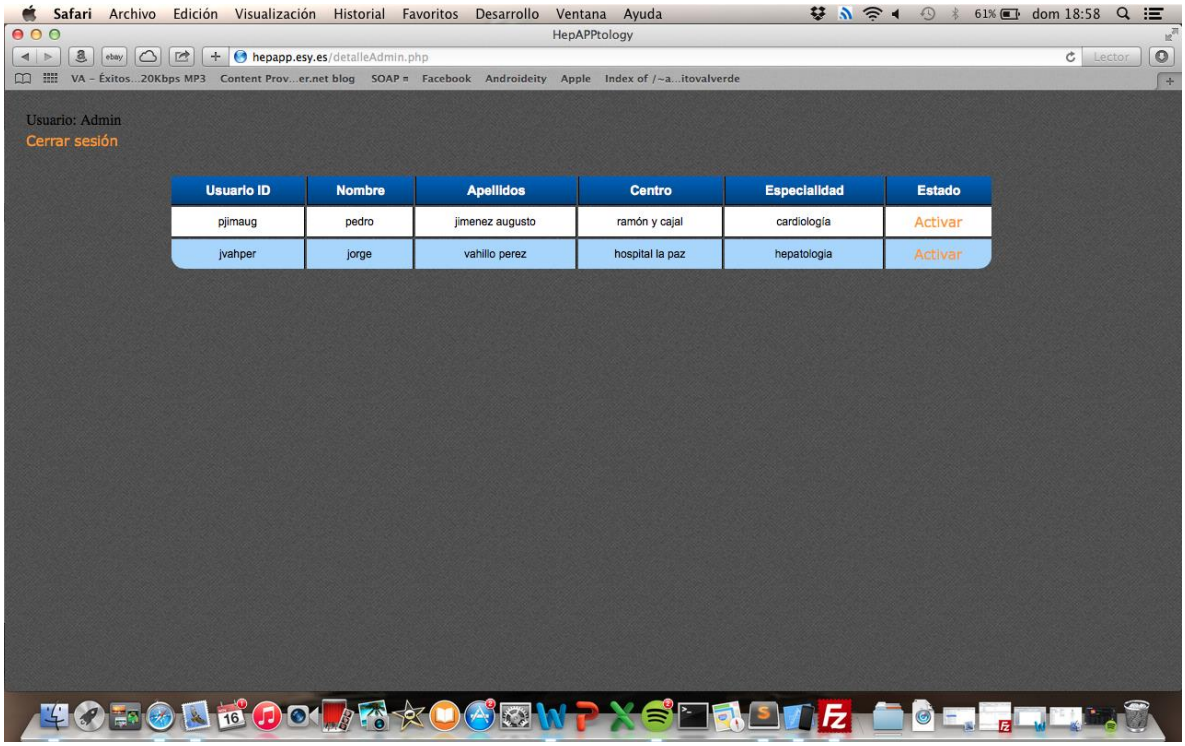

Ilustración 24. Aplicación web: panel de administración

Desde ella, se podrá gestionar la activación de los usuarios que estén pendientes de registrarse. Para ello, solo es necesario pulsar sobre el enlace de “Activar” asociado a cada usuario. Una vez hecho esto, este desaparecerá de la lista.

3.4.5 Aplicación móvil. Inicio de sesión

Nada más iniciar la *HepAPPtology* nos aparece la vista de login como se muestra a continuación.

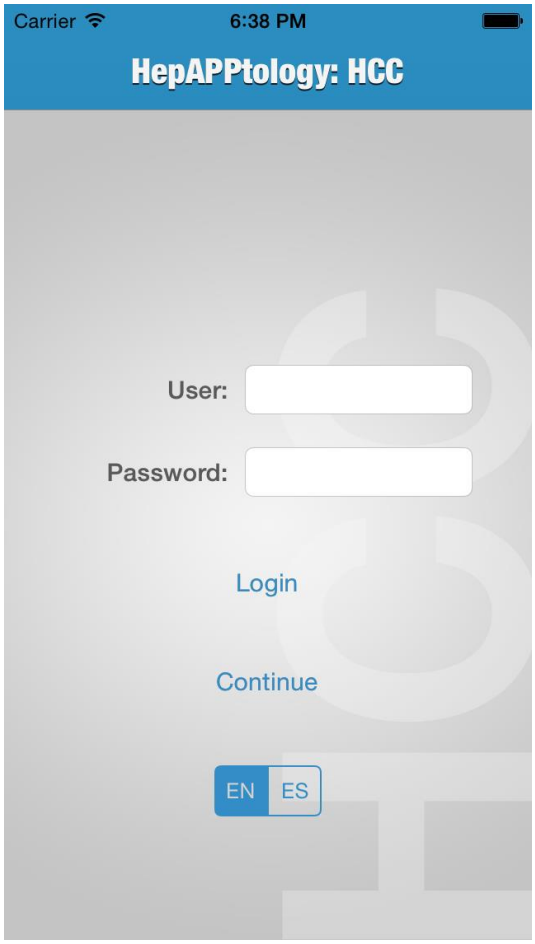

Ilustración 25. Aplicación móvil: vista login

En ella deberemos rellenar los datos de acceso si disponemos de ellos y podremos seleccionar el idioma en el que queremos que se muestre el contenido de nuestra aplicación. Una vez hecho esto, y pulsando sobre el botón acceso doctores, accederemos a todo el contenido de la aplicación.

Si por el contrario no disponemos datos de acceso, podremos saltarnos este paso de autenticación pulsando sobre el botón de acceso invitados. La diferencia reside en que si accedemos como doctores, podremos registrar tanto los datos del paciente como los resultados obtenidos de la aplicación de los algoritmo pudiendo verlos reflejados más tarde en la aplicación web.

También destacar en esta vista, que si accedemos a la aplicación sin conexión a internet, solo podremos acceder como invitados, y además, si es la primera vez

que la lanzamos, no será posible continuar con ella al ser necesario una primera descarga de los contenidos a mostrar.

3.4.6 Aplicación móvil. Introducción, capítulos y enlaces.

Una vez autenticados, se mostrará un menú principal con las distintas opciones que nos permite la aplicación.

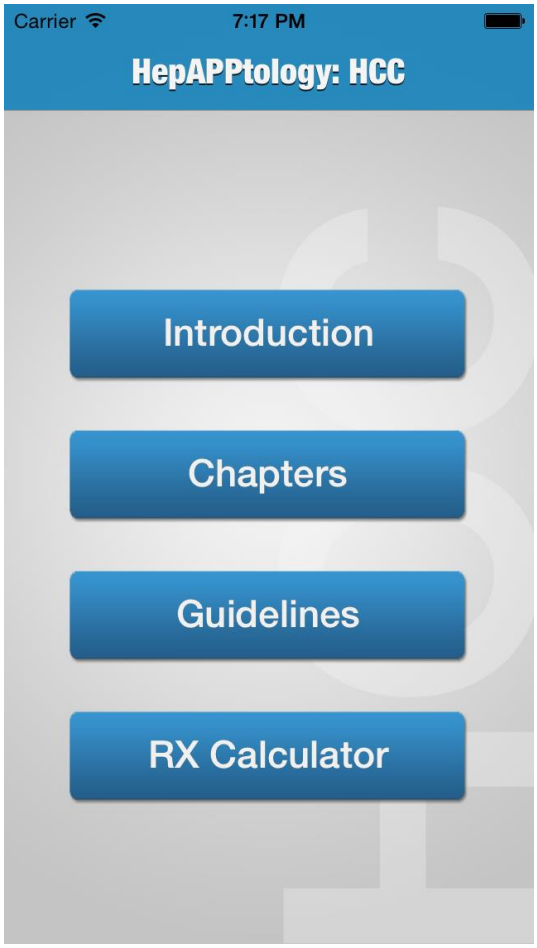

Ilustración 26. Aplicación móvil: Menú principal

Entre ellas se encuentran en primer lugar la introducción. Accediendo a esta nos mostrará una breve descripción además de una sección de contacto donde nos

aparecerá el teléfono de contacto, una dirección, y la localización de esta sobre un mapa.

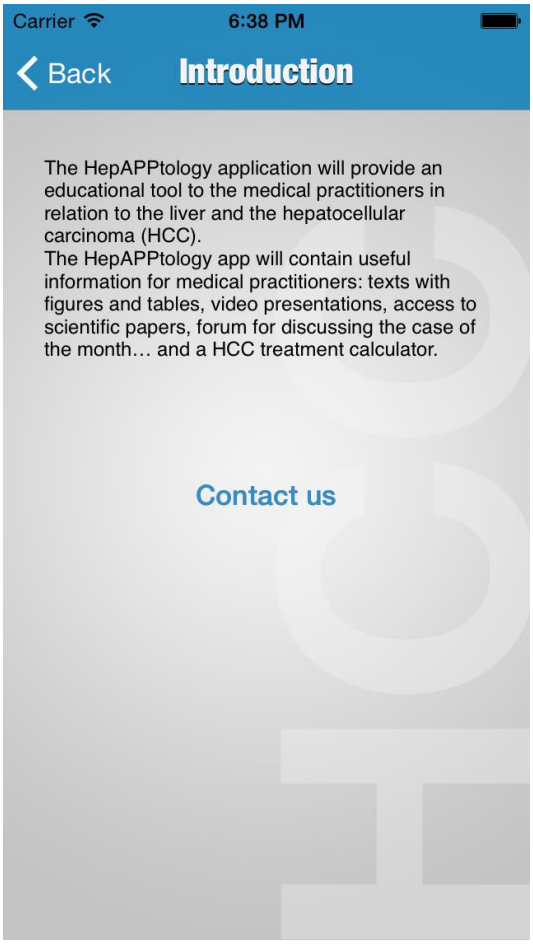

Ilustración 27. Aplicación móvil: Vista introducción

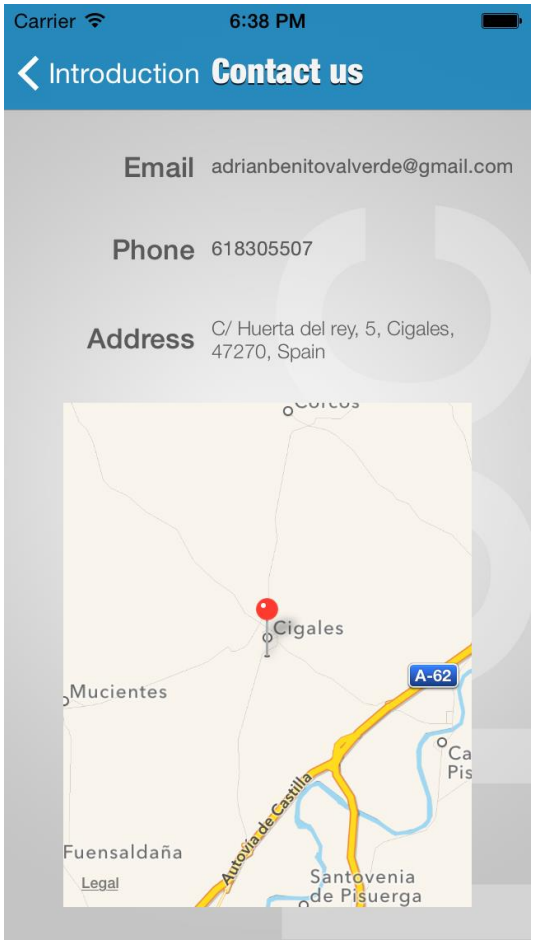

Ilustración 28. Aplicación móvil: Vista contacto

Por otra parte tenemos una opción de capítulos con el fin de que sirva de wiki sobre la propia enfermedad. Nos mostrará una lista con varias secciones relacionadas con la enfermedad o la propia aplicación. Pulsando sobre una de ellas, nos mostrará información en detalle.

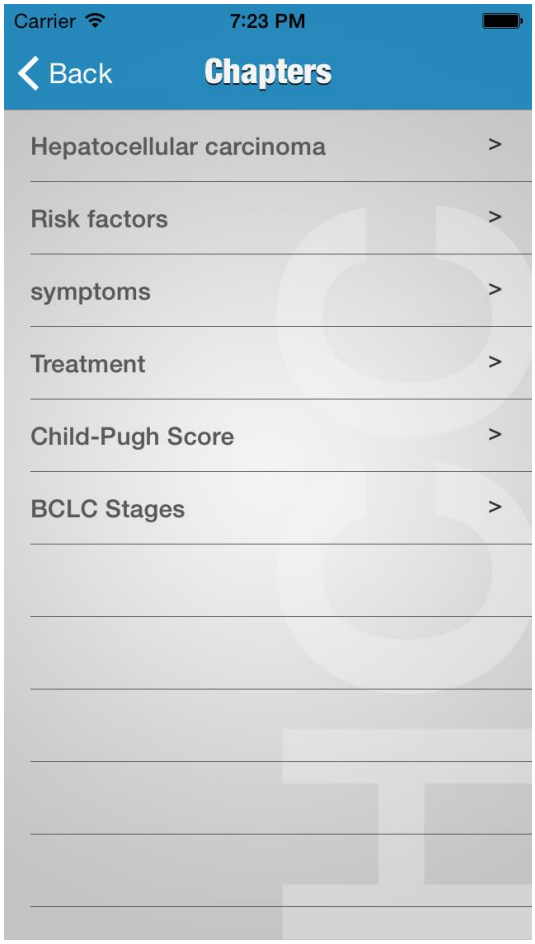

Ilustración 29. Aplicación móvil: Vista capítulos

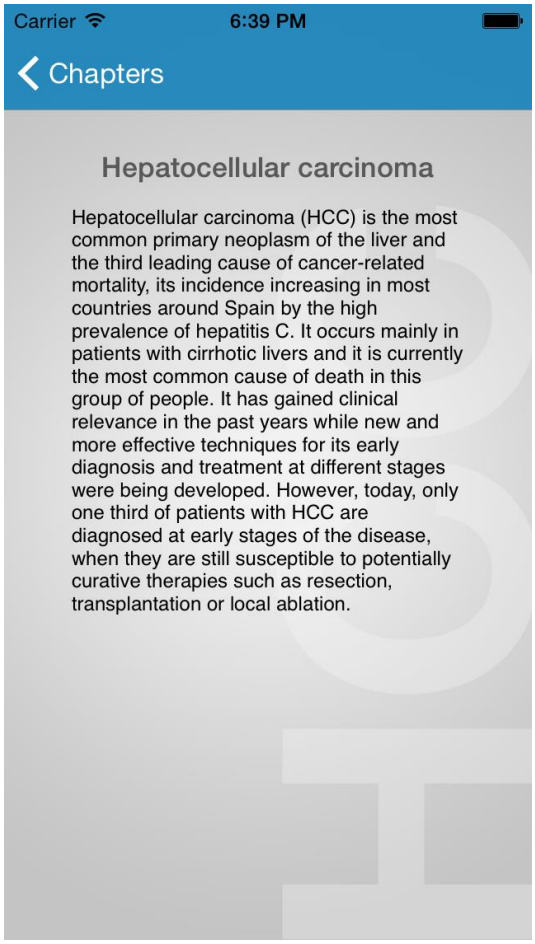

Ilustración 30. Aplicación móvil: Vista detalle capitulo

Otra de las opciones que aporta es Enlaces, una lista de links a documentos relacionados con el tema tratado.

Ilustración 31. Aplicación móvil: Vista enlaces

3.4.7 Aplicación móvil. Calculadora.

La última opción y también la más interesante, es la calculadora. El objetivo de esta es, tras introducir un conjunto de datos sobre el paciente, se aplicarán distintos algoritmos médicos que nos aportarán los resultados esperados por la aplicación.

Uno de los primeros pasos para obtenerlos es, si hemos accedido como profesionales médicos, cumplimentar todos los datos del paciente. Entre ellos, es

posible no aportar el usuario, nombre y apellidos con el fin de guardar la privacidad del paciente, sin embargo otros como la edad, peso, estatura o correo de contacto si serán obligatorios.

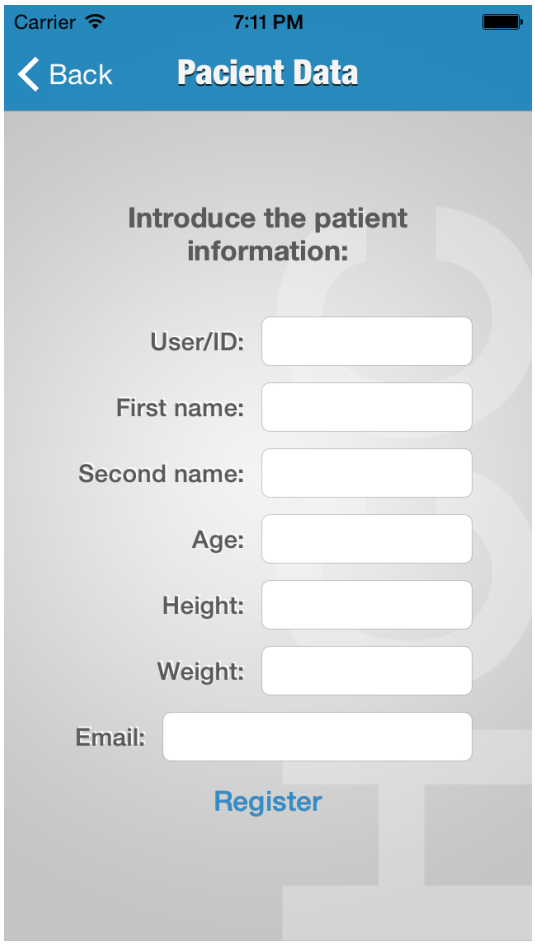

Ilustración 32. Aplicación móvil: Vista registro de pacientes

Una vez registrado. Nos aparecerá un menú con tres opciones además del identificador de usuario del paciente en la parte superior. Si hemos decidido no aportarlo, se generará un ID automático con el fin de que pueda ser usado por el paciente para consultar sus pruebas a través de la aplicación web.

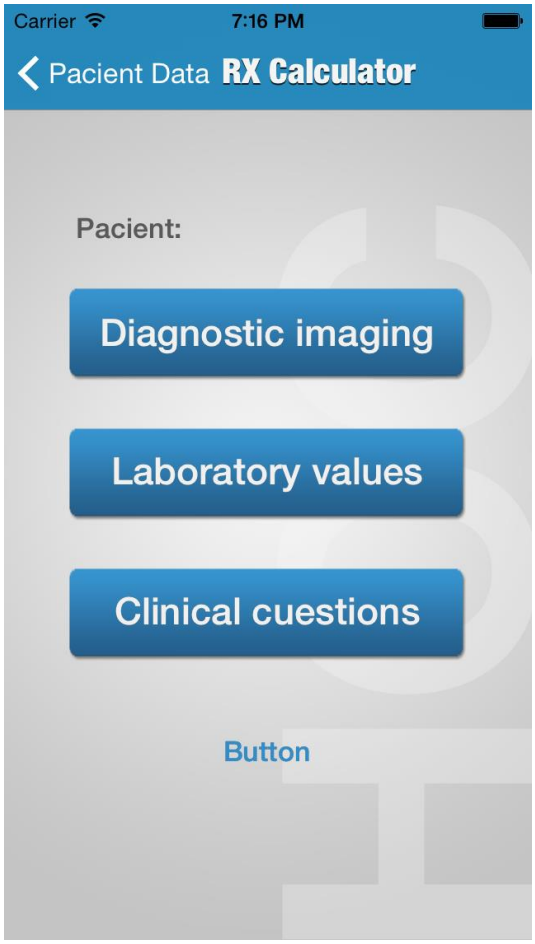

Ilustración 33. Aplicación móvil: Vista calculadora

Las tres opciones que nos aparecen en esta vista son, Diagnóstico por imagen, Resultados de laboratorio y Cuestiones clínicas. Cada una de ellas contendrá un formulario específico sobre su tema.

Para poder obtener los resultados esperados por la aplicación, es necesario rellenar todos los campos que constan en los formularios, así como cumplimentar todos ellos. Si algún campo se deja en blanco este será validado por la aplicación y no permitirá avanzar mostrando un mensaje de error.

Una vez rellenados todos los formularios, podremos pulsar sobre el botón de calcular para obtener los distintos.

Hay que tener en cuenta que los datos aportados a la aplicación tienen que ser coherentes. De no ser así, es posible que algunos valores no se lleguen a calcular mostrando en su lugar un “No disponible”.

Por último, siempre que hayamos accedido como profesional médico y teniendo acceso a internet. Podremos registrar tanto los datos del paciente como los resultados obtenidos pulsando sobre el botón de registrar datos. Una vez hecho esto, el paciente recibirá un correo electrónico con sus datos de acceso y los pasos necesarios para consultarlos.

3.5 Pruebas

En este capítulo se recogen diversas pruebas unitarias realizadas a ambas aplicaciones, tanto la web como la móvil, con el fin de evaluar y verificar el correcto funcionamiento de cada una de ellas.

3.5.1 Casos de prueba para la aplicación web

Código	C001
Escenario	Acceso de usuarios, Fallo de autenticación
Instrucciones de ejecución	1. El usuario accede a la aplicación. 2. El usuario introduce unos datos de acceso en la ventana de login que no se encuentran en la base de datos.
Resultados esperados	Se muestra un mensaje de error.
Resultados obtenidos	Se muestra un mensaje de error, "Usuario o contraseña incorrectos".
Estado	OK

Tabla 8. Caso de prueba C001: Fallo de autenticación

Código	C002
Escenario	Acceso de usuarios, Usuario no activo
Instrucciones de ejecución	1. El usuario accede a la aplicación. 2. El usuario introduce unos datos de acceso en la ventana de login pertenecientes a un usuario que aún no ha sido activado.
Resultados esperados	Se muestra un mensaje de error.
Resultados obtenidos	Se muestra un mensaje de error indicando que el usuario no ha sido activado todavía.
Estado	OK

Tabla 9. Caso de prueba C002: Usuario no activo

Código	C003
Escenario	Acceso de usuarios, Autenticación correcta
Instrucciones de ejecución	1. El usuario accede a la aplicación 2. El usuario introduce unos datos de acceso en la ventana de login existen en la base de datos.
Resultados esperados	Se muestra la siguiente vista.
Resultados obtenidos	Se muestra la vista con la lista de pacientes o pruebas.
Estado	OK

Tabla 10. Caso de prueba C003: Autenticación correcta

Código	C004
Escenario	Registro de usuarios, Campos vacíos o nulos
Instrucciones de ejecución	1. EL usuario accede a la aplicación. 2. El usuario pulsa sobre el enlace de registrarse. 3. Se rellenan algunos campos o se dejan todos en blanco. 4. Se pulsa sobre el botón de registrarse.
Resultados esperados	Se muestra un mensaje de error
Resultados obtenidos	Se muestra un mensaje de error indicando que todos los campos son obligatorios.
Estado	OK

Tabla 11. Caso de prueba C004: Registro de usuarios fallido 1

Código	C005
Escenario	Registro de usuarios, Usuario ya existe
Instrucciones de ejecución	1. EL usuario accede a la aplicación 2. El usuario pulsa sobre el enlace de registrarse. 3. Se rellenan todos los campos con nombre y apellidos ya existentes. 4. Se pulsa sobre el botón de registrarse.
Resultados esperados	Se muestra un mensaje de error
Resultados obtenidos	Se muestra un mensaje de error indicando que ya existe un usuario con ese nombre.
Estado	OK

Tabla 12. Caso de prueba C005: Registro de usuarios fallido 2

Código	C006
Escenario	Registro de usuarios, Registro válido
Instrucciones de ejecución	1. EL usuario accede a la aplicación 2. El usuario pulsa sobre el enlace de registrarse. 3. Se rellenan todos los campos con datos nuevos. 4. Se pulsa sobre el botón de registrarse.
Resultados esperados	Se muestra un mensaje de confirmación
Resultados obtenidos	Se muestra un mensaje de error indicando que el usuario está a la espera de confirmación por parte de un administrador
Estado	OK

Tabla 13. Caso de prueba C006: Registro de usuarios válido

Código	C007
Escenario	Listado pacientes
Instrucciones de ejecución	1. EL usuario accede a la aplicación. 2. El usuario rellena los datos de acceso como doctor. 3. Se pulsa sobre el botón de acceder.
Resultados esperados	Se muestra un buscador y la lista con los pacientes.
Resultados obtenidos	Se muestra un buscador y la lista con los pacientes.
Estado	OK

Tabla 14. Caso de prueba C007: Listado pacientes

Código	C008
Escenario	Búsqueda de pacientes, usuario no encontrado
Instrucciones de ejecución	1. EL usuario accede a la aplicación. 2. El usuario rellena los datos de acceso como doctor. 3. Se pulsa sobre el botón de acceder. 4. Se rellenan los campos Nombre/ID y Apellidos con datos no existentes en la base de datos. 5. Se pulsa sobre el botón “buscar”
Resultados esperados	Se muestra un mensaje de error.
Resultados obtenidos	Se muestra un mensaje indicando que no se encontraron resultados.
Estado	OK

Tabla 15. Caso de prueba C008: Búsqueda de pacientes fallida

Código	C009
Escenario	Búsqueda de pacientes, usuario encontrado
Instrucciones de ejecución	1. EL usuario accede a la aplicación. 2. El usuario rellena los datos de acceso como doctor. 3. Se pulsa sobre el botón de acceder. 4. Se rellenan los campos Nombre/ID y Apellidos con datos existentes en la base de datos. 5. Se pulsa sobre el botón "buscar"
Resultados esperados	Se muestra una lista con las coincidencias.
Resultados obtenidos	Se muestra una lista con todos los pacientes que cumplan esa condición
Estado	OK

Tabla 16. Caso de prueba C009: Búsqueda de pacientes correcta

Código	C010
Escenario	Acceso a pruebas desde listado de pacientes
Instrucciones de ejecución	1. EL usuario accede a la aplicación. 2. El usuario rellena los datos de acceso como doctor. 3. Se pulsa sobre el botón de acceder. 4. Se pulsa sobre el nombre un paciente. 5. Se pulsa sobre el ID de prueba de un paciente.
Resultados esperados	Se muestra una tabla con los resultados
Resultados obtenidos	Se muestra una tabla con todos los resultados asociados a esa prueba.
Estado	OK

Tabla 17. Caso de prueba C010: Acceso a pruebas como doctor

Código	C011
Escenario	Acceso a pruebas desde listado de pruebas
Instrucciones de ejecución	1. EL usuario accede a la aplicación. 2. El usuario rellena los datos de acceso como paciente. 3. Se pulsa sobre el botón de acceder. 4. Se pulsa sobre el ID de prueba de un paciente.
Resultados esperados	Se muestra una tabla con los resultados
Resultados obtenidos	Se muestra una tabla con todos los resultados asociados a esa prueba.
Estado	OK

Tabla 18. Caso de prueba C011: Acceso a pruebas como paciente

Código	C012
Escenario	Acceso al panel de administrador.
Instrucciones de ejecución	1. EL usuario accede a la aplicación. 2. El usuario rellena los datos de acceso como administrador. 3. Se pulsa sobre el botón de acceder.
Resultados esperados	Se muestra una lista con usuarios.
Resultados obtenidos	Se muestra una lista con todos los usuarios inactivos.
Estado	OK

Tabla 19. Caso de prueba C012: Acceso a panel de administrador

Código	C013
Escenario	Panel de administrador, activar usuario.
Instrucciones de ejecución	1. EL usuario accede a la aplicación. 2. El usuario rellena los datos de acceso como administrador. 3. Se pulsa sobre el botón de acceder. 4. Pulsa sobre el botón activar asociado a un usuario
Resultados esperados	Se actualiza la entrada de ese usuario con ACTIVE=S, Se elimina de la lista.
Resultados obtenidos	Se actualiza la entrada de ese usuario con ACTIVE=S, Se elimina de la lista.
Estado	OK

Tabla 20. Caso de prueba C013: Panel de administrador, Activar usuario

3.5.2 Casos de prueba para la aplicación móvil

Código	C014
Escenario	Inicio de la aplicación sin acceso a internet
Instrucciones de ejecución	1. EL usuario accede a la aplicación sin acceso a internet.
Resultados esperados	No da la pasibilidad de iniciar sesión como doctor.
Resultados obtenidos	No se muestra el botón de loguearse como doctor.
Estado	OK

Tabla 21. Caso de prueba C014: Inicio de aplicación móvil sin internet.

Código	C015
Escenario	Inicio de la aplicación acceso a internet
Instrucciones de ejecución	1. EL usuario accede a la aplicación con acceso a internet.
Resultados esperados	Da la posibilidad de iniciar sesión como doctor o invitado.
Resultados obtenidos	Se muestran los botones de acceso como doctor e invitado.
Estado	OK

Tabla 22. Caso de prueba C015: Inicio de aplicación con internet.

Código	C016
Escenario	Iniciar sesión, datos incompletos
Instrucciones de ejecución	1. EL usuario accede a la aplicación con acceso a internet. 2. Se dejan los datos de acceso en blanco o incompletos. 3. Se pulsa sobre el botón de iniciar sesión como doctor.
Resultados esperados	Muestra un error.
Resultados obtenidos	Muestra un mensaje indicando que los datos de acceso no pueden ir en blanco.
Estado	OK

Tabla 23. Caso de prueba C016: Iniciar sesión con datos incompletos

Código	C017
Escenario	Iniciar sesión, datos erróneos
Instrucciones de ejecución	1. EL usuario accede a la aplicación con acceso a internet. 2. Se rellenan los datos de acceso con información no registrada en la base de datos. 3. Se pulsa sobre el botón de iniciar sesión como doctor
Resultados esperados	Muestra un mensaje de error
Resultados obtenidos	Muestra un mensaje indicando que el usuario o contraseña son erróneos.
Estado	OK

Tabla 24. Caso de prueba C017: Inicio de sesión con datos erróneos

Código	C018
Escenario	Iniciar sesión, autenticación correcta
Instrucciones de ejecución	1. EL usuario accede a la aplicación con acceso a internet. 2. Se rellenan los datos de acceso con información registrada en la base de datos. 3. Se pulsa sobre el botón de iniciar sesión como doctor
Resultados esperados	Se avanza de vista.
Resultados obtenidos	Se muestra el menú principal de la aplicación
Estado	OK

Tabla 25. Caso de prueba C018: Autenticación correcta

Código	C019
Escenario	Selección de idioma
Instrucciones de ejecución	1. EL usuario accede a la aplicación. 2. Se rellenan los datos de acceso con información registrada en la base de datos. 3. Se selecciona el lenguaje deseado. 4. Se pulsa sobre el botón de iniciar sesión como doctor.
Resultados esperados	Se avanza de vista.
Resultados obtenidos	Se muestra el menú principal de la aplicación en el idioma deseado. El resto de vistas también estarán en el mismo idioma.
Estado	OK

Tabla 26. Caso de prueba C019: Selección de idioma

Código	C020
Escenario	Inicio de sesión como invitado
Instrucciones de ejecución	1. EL usuario accede a la aplicación. 2. Se pulsa sobre el botón de iniciar sesión como invitado.
Resultados esperados	Se avanza de vista. Se limita las funcionalidades de la aplicación.
Resultados obtenidos	Se muestra el menú principal de la aplicación, no se mostrará la vista de registro de pacientes ni se permitirá registrar los resultados en base de datos.
Estado	OK

Tabla 27. Casi de prueba C020: Inicio de sesión como invitado

Código	C021
Escenario	Calculadora, no se completan todos los datos de algún formulario
Instrucciones de ejecución	1. EL usuario accede a la aplicación. 2. Se pulsa sobre el botón de iniciar sesión. 3. Se selecciona la opción de calculadora del menú principal. 4. Se selecciona algún formulario de pruebas. 5. No se rellenan todos los datos. 6. Se pulsa sobre el botón de enviar.
Resultados esperados	Muestra un mensaje de error.
Resultados obtenidos	Muestra un mensaje indicando que todas las entradas son obligatorias.
Estado	OK

Tabla 28. Caso de prueba C021: formularios incompletos

Código	C022
Escenario	Calculadora, no se completan inician todos los formularios
Instrucciones de ejecución	1. EL usuario accede a la aplicación. 2. Se pulsa sobre el botón de iniciar sesión. 3. Se selecciona la opción de calculadora del menú principal. 4. Se pulsa sobre el botón de calcular.
Resultados esperados	Muestra un mensaje de error.
Resultados obtenidos	Muestra un mensaje indicando que es necesario completar el resto de formularios.
Estado	OK

Tabla 29. Caso de prueba C022: Faltan formularios

Código	C023
Escenario	Calculadora, Calculo de resultados.
Instrucciones de ejecución	1. EL usuario accede a la aplicación. 2. Se pulsa sobre el botón de iniciar sesión. 3. Se selecciona la opción de calculadora del menú principal. 4. Se rellenan todos los formularios. 5. Se pulsa sobre el botón de calcular.
Resultados esperados	Se muestra una vista con los resultados.
Resultados obtenidos	Se muestra una vista con los resultados.
Estado	OK

Tabla 30. Caso de prueba C023: Calculo resultados

Código	C024
Escenario	Calculadora, no se completan todos los datos de algún formulario
Instrucciones de ejecución	1. EL usuario accede a la aplicación con internet. 2. Se rellenan los datos de acceso 3. Se pulsa sobre el botón de iniciar sesión como doctor. 4. Se selecciona la opción de calculadora del menú principal. 5. Se selecciona algún formulario de pruebas. 6. se rellenan todos los datos. 7. Se pulsa sobre el botón de calcular. 8. Se pulsa sobre el botón de registrar datos
Resultados esperados	Muestra un mensaje de confirmación
Resultados obtenidos	Muestra un mensaje confirmando que los datos se han registrado en base de datos
Estado	OK

Tabla 31. Caso de prueba C024: registro de datos en BD

4 Presupuesto económico

Hoy en día, el coste económico que supone la creación de una aplicación para dispositivos móviles puede depender de muchos factores. Entre ellos, es necesario destacar por una parte las funcionalidades de las que hará uso. No conlleva el mismo trabajo una aplicación que hace uso de la cámara, el GPS y el acelerómetro para "buscar" a otras personas, o edificios, que una aplicación que permita hacer fotos y subirlas a un servidor como era de esperar.

Otro de los puntos aspectos a destacar a la hora de dar un presupuesto para el desarrollo de una aplicación, es el tamaño de esta, el número de vistas que contendrá. Bien es cierto que no es un aspecto muy fiable a la hora de estimar el tiempo que supondrá la creación de la aplicación debido a la posible complejidad de algunas vistas frente a otras.

Por último, otro de los aspectos a tener en cuenta es el tipo de aplicación que se quiere desarrollar así como las plataformas sobre las que funcionará.

Muchas empresas de diseño y desarrollo al igual que *freelancers* se comprometen a tener lista una aplicación móvil en menos de 20 días a un precio bastante reducido. Esto puede ser muy interesante si el objetivo es abaratar costes y no exiges mucho de la aplicación ya que probablemente obtendrás una aplicación web móvil basada en plantillas de estructura muy similar a otras del mercado. Sin embargo, si lo que se quiere es una aplicación más personal y priorizando el rendimiento. Esto conllevará más carga de trabajo y por lo tanto un coste más elevado.

En cuanto al aspecto de la aplicación, un diseño completo y personalizado puede rondar los 500-800 euros, en función también de las dimensiones y exigencias de esta.

Para el caso de *HepAPPtology* el *restyling* ha supuesto alrededor de 200 euros dado que aproximadamente el 70-80% del diseño de la aplicación ya se encontraba hecho.

Es por ello que, una aplicación móvil, nativa para iOS, desarrollada íntegramente en Objective-C y con un aspecto personalizado, destacando como características o funcionalidades, por una parte, la posibilidad de modificar su contenido fácilmente y el soporte multilenguaje, el uso de datos de entrada para la generación de resultados a partir de varios algoritmos y la interacción con una base de datos, y el desarrollo de una pequeña aplicación web donde poder consultarlos, el coste económico de una aplicación así teniendo también en cuenta que el público al que está orientado son hospitales o centros médicos, puede rondar los 1500-2000 euros.

5 Conclusión y líneas futuras

5.1 Conclusión

Con la elaboración de este documento se ha conseguido en primer lugar contextualizar la necesidad de desarrollar una aplicación para dispositivos móviles que permita realizar un seguimiento sobre enfermedades que hayan sido diagnosticadas en un paciente.

Se han analizado distintas tecnologías que nos permiten por una parte, desarrollar una aplicación web para la consulta de resultados y gestión de pacientes haciendo una distinción entre aquellas que trabajan del lado cliente como son HTML, CSS, JavaScript y Ajax, y las que trabajan del lado del servidor como ASP.NET, Perl o php. Además, para cada una se ha terminado valorando la idea de hacer uso de ella para el desarrollo de la aplicación.

Por otro lado, para el desarrollo de la aplicación móvil, se ha realizado una comparativa entre los distintos tipos de aplicaciones que existen, aplicaciones nativas, aplicaciones web móvil y aplicaciones hibridas. Se ha hablado sobre las principales plataformas o sistemas operativos móviles que el mercado ofrece como son Android, iOS y Windows Phone para terminar justificando cual es la opción que más se ajusta a nuestros objetivos o necesidades. Una aplicación nativa para dispositivos iOS.

Terminando con el análisis de estas tecnologías, se ha cuestionado la necesidad de usar un sistema de control de versiones como Git y phpMyAdmin como herramienta para la administración de bases de datos relacionales a través de un interfaz web.

Otro de los puntos recogidos en este documento y entrando ya a hablar sobre la propia aplicación, es la identificación de los distintos beneficios que esta aportará además de las características principales de ella.

La posibilidad de disponer de una aplicación cuyo contenido este alojado en un servidor web y por consiguiente, que aporte dinamismo a esta. La idea de que este hecho, a mayores nos facilite la actualización de la información que se mostrará así como el soporte en varios idiomas.

Se ha realizado un análisis técnico de la aplicación, desde el proceso de inicio de sesión o registro de profesionales médicos en la interfaz web hasta el proceso de registro de información de todos los datos del paciente contando con los resultados obtenidos tras la aplicación de los algoritmos médicos.

Posteriormente se ha elaborado un manual de uso con el fin de que pueda servir de guía a los posibles usuarios de la aplicación.

Se han detallado un conjunto de casos de pruebas donde se recogen las posibles situaciones o escenarios que se pueden dar y verificando el correcto funcionamiento de las aplicaciones.

En definitiva, destacar que se han alcanzado de forma satisfactoria los objetivos que se pretendían con el desarrollo de este trabajo fin de grado. Por una lado, el aprendizaje de un lenguaje de programación ampliamente utilizado en el desarrollo de aplicaciones móviles como es *Objective-C*, el cumplimiento de todos los plazos y fases de los que consta un proyecto, y por otro, el desarrollo de una aplicación *eHealth* que en un futuro pueda servir tanto a médicos como a pacientes, a llevar un control más preciso y temprano sobre ciertas enfermedades pudiendo permitir esto, reducir el índice de mortalidad asociado a una de las asuntos que más preocupa a la población, la salud.

5.2 Líneas futuras

Por una parte, en cuanto a las posibles tecnologías, a pesar de haber elegido *Objective-C* para el desarrollo de la aplicación móvil y *php* y *phpMyAdmin* como lenguajes o herramientas del lado del servidor, habría sido muy interesante desarrollar este mismo proyecto empleando tecnologías basadas en Java. Bien es cierto que aunque desde un principio la idea era desarrollar una aplicación para dispositivos móviles únicamente, en concreto iOS, se ha visto la posibilidad de crear una pequeña aplicación web para la consulta de datos sin afectar a los plazos marcados. No obstante, a pesar de esto me habría gustado aplicar los conocimientos adquiridos en el mundo laboral para desarrollar una aplicación web con fuerte carga de Java que reúna las características de ambas. Por una parte un sistema de gestión de pacientes y por otro lado un site que permita además de aportar información sobre varias enfermedades, obtener diversos resultados a través del aporta de datos. Para ello, por un lado la interfaz web se basaría en JSP y JSF y la parte de la lógica de la aplicación y conexión con base de datos se apoyaría en Java y en frameworks o APIs como Hibernate, Spring o Maven.

Respecto a la aplicación actual, sería interesante ampliar el número de enfermedades que soporta teniendo en cuenta que, en este aspecto y dado las características de la aplicación, el trabajo más complicado residiría en dar con el conjunto de algoritmos asociados a la enfermedad.

Otro de los aspectos que nos aporta la posibilidad de registrar todos los datos clínicos de los pacientes es el uso de estos con fines estadísticos y es que, dado los datos proporcionados a la hora de realizar una prueba, sería posible estudiar a medio-largo plazo la incidencia de cierta enfermedad y valores como el grado de severidad de ella en pacientes varones, de cierta estatura y peso, o en rango de edades. Así, no yendo tan lejos, sería posible desde la propia aplicación web,

realizar comparativas entre distintos resultados de un mismo paciente o mostrar las evoluciones de ciertos valores en gráficas.

Para finalizar, y hablando sobre el uso que se le puede dar a la aplicación, llegar a un acuerdo con algún centro médico que estuviera interesado en explotarla sería lo más factible. Por supuesto, realizando las modificaciones oportunas de acuerdo a los intereses del propio centro. Otra opción sería sacarla al mercado a través del Apple Store pero dado el público reducido al que se dirige, sería complicado sacarla a delante.

Bibliografía

[1] Cancer, I. A. (2012). *globocan.iarc.fr*.

[2] Organización Mundial de la Salud. (2014). *OMS*. Recuperado en Noviembre de 2014, de www.who.int

[3] Asociación Española Contra el Cáncer. (9 de Julio de 2014). *Posibles causas del cáncer*. Recuperado en 2014, de www.aecc.es

[4] Sociedad Española de Oncología Médica. (2014). *Las Cifras del Cáncer en España 2014*. Obtenido de www.seom.org

[5] Saunders, D. K. (2011). *HepAPPtology - Application Definition Document.* Calgary.

[6] Apple Inc. (s.f.). *Xcode Overview*. Obtenido de www.developer.apple.com

[7] Bucanek, J. (s.f.). *Learn iOS 7 App Development.* Apress.

[8] Hoffman, J. (s.f.). *iOS 7 Development recipes.* Apress.

[9] Kochan, S. G. (s.f.). *Programming in Objective-C.* Addison Wesley.

[10] Fariña, A. d. (s.f.). *Tecnología WEB. Programación en el lado del cliente*. Recuperado en 2014, de tecsanmartinisc.galeon.com

[11] Mozilla Corporation. (Julio de 2014). *HTML*. Recuperado en Noviembre de 2014, de www.developer.mozilla.org

[12] World Wide Web Consortium (W3C). (s.f.). *Guía Breve de CSS*. Recuperado en Noviembre de 2014, de www.w3c.es

[13] Librosweb.es. (s.f.). *Introcucción a AJAX*. Recuperado en Noviembre de 2014, de www.librosweb.es

[14] Villalobos, J. (Septiembre de 2010). *Comparación php, jsp y asp.net.* Recuperado en Noviembre de 2014, de www.codigoprogramacion.com

[15] W3Techs. (Octubre de 2014). *W3Techs - World Wide Web Technology Surveys.* Recuperado en Noviembre de 2014, de www.w3techs.com

[16] Microsoft. (s.f.). *Get Started with ASP.NET.* Recuperado en Noviembre de 2014, de www.asp.net

[17] Álvares, M. A. (29 de Septiembre de 2001). *DesarrolloWeb, Qué es Perl.* Recuperado en Noviembre de 2014, de www.desarrolloweb.com

[18] Ángel Cobo, P. G. (s.f.). *PHP y MySQL, Tecnologías para el desarrollo de aplicaciones web.*

[19] Oracle Corporation. (s.f.). *Types of SQL Statements.* Recuperado en Noviembre de 1014, de www.docs.oracle.com

[20] db-engines. (Noviembre de 2014). *Popularity ranking of database management systems*. Recuperado en Noviembre de 2014, de www.db-engines.com

[21] Oracle Corporation. (s.f.). *Oracle SQL Developer*. Obtenido de www.oss.oracle.com

[22] Oracle Corporation. (s.f.). *Why MySQL?* Recuperado en Noviembre de 2014, de www.mysql.com

[23] Rackspace Support. (Mayo de 2013). *MySQL Engines - MyISAM vs Innodb.* Recuperado en Noviembre de 2014, de www.rackspace.com

[24] *phpMyAdmin*. (s.f.). Obtenido de www.phpmyadmin.net

[25] Accensit. (s.f.). *Comparativa de tecnologías para el desarrollo de aplicaciones móviles*. Obtenido de www.accensit.com

[26] Apache Software Foundation. (s.f.). *Apache Cordova API Documentacion, Overview*. Recuperado en Noviembre de 2014, de www.cordova.apache.org

[27] Gartner Inc. (2012). *Forecast: Mobile Communications Devices by Open Operating System.* Recuperado en Noviembe de 2014, de www.gartner.com

[28] Williams, R. (2 de Septiembre de 2014). Apple iOS: a brief history. *The Telegraph*.

[29] Git-scm. (s.f.). *Acerca del control de versiones*. Recuperado en Noviembre de 2014, de www.git.scm.com

[30] btilford. (Noviembre de 2010). *Comparison of Git with CVS and Git Overview*. Recuperado en Noviembre de 2014

[31] Plazas, D. J. (14 de Marzo de 2013). *Cáncer de hígado.* Recuperado en Noviembre de 2014, de www.seom.org

[32] Soza, D. A. (s.f.). *Hepatología*. Obtenido de www.hepatitis.cl

[33] Gámiz, J. L. (s.f.). *¿Qué es la eHealth?* Recuperado en Noviembre de 2014, de www.ediagnostic.es