Creación de una interfaz gráfica para visualizar árboles genealógicos

María González Palacios
Máster en Bioinformática y Bioestadística
Bioinformática clínica

Director: Guerau Fernández Isern
Profesor responsable de la asignatura: Javier Luis Cánovas Izquierdo

Enero de 2020

Título del trabajo:	Creación de una interfaz gráfica para visualizar árboles genealógicos
Nombre del autor:	María González Palacios
Nombre del director:	Guerau Fernández Isern
Nombre del PRA:	Javier Luis Cánovas Izquierdo
Fecha de entrega (mm/aaaa):	01/2020
Titulación:	Máster en Bioinformática y Bioestadística
Área del Trabajo Final:	Área 1. Bioinformática clínica
Idioma del trabajo:	Castellano
Palabras clave:	Aplicación web, Árbol genealógico, Historial médico

Resumen del Trabajo:

La idea de realizar este proyecto surge de la necesidad de recoger en un mismo sitio todos los datos genealógicos de una familia, tanto personales como clínicos, y mostrarlos. Para ello, se ha creado una aplicación web en la que es posible, de una manera sencilla y amigable, introducir información de cada miembro del árbol y visualizar su contenido, así como realizar búsquedas sobre el historial médico. Las herramientas que utilizadas han sido el *framework* Spring y el motor de base de datos MySQL, entre otras; y se ha seguido una metodología ágil a través del encadenamiento de tareas, ya que tener un control de versiones es importante para conocer los posibles errores que vayan surgiendo en el desarrollo. Si se piensa en el futuro, este proyecto puede abarcar mucho recorrido, tanto en la parte de diseño de la interfaz de usuario como en la funcionalidad. Gracias a la estructura base que se ha montado, se pueden agregar nuevos módulos muy fácilmente sin que afecte al código ya generado.

Abstract:

The leading idea for this project comes from the necessity of grouping in the same place all the genalogical data of a family, both personal and clinic, and visualizing them. For this purpose a web application has been created in which, in a simple and friendly manner, each member's information can be introduced and visualized; and searches can be made in the medical history. The tools employed are Spring framework and MySQL database, among others, and a agile methodology has been followed for task chaining, since a version control is crucial to track the possible errors that can occur during development. For future developments this project is very expandable, both in user interface design and functionality because of the employed architecture new modules can be added easily without affecting the already generated code.

Índice

Lista de figuras

1. Introducción

1.1 Contexto y justificación

El Trabajo Final de Máster consiste en diseñar y desarrollar una aplicación web que proporcione al usuario una interfaz donde visualizar árboles genealógicos basados en los datos introducidos.

La aplicación web permitirá al usuario crearse una cuenta y acceder a ella con una contraseña, dar de alta miembros del árbol, modificar su información, añadir datos adicionales y dar de baja otros miembros; además de generar el árbol genealógico resultante y visualizar todos sus datos. La aplicación también dará la posibilidad de realizar búsquedas personalizadas con la información que haya guardada sobre el historial médico de cada miembro.

La razón por la que se realiza este proyecto es la necesidad de recoger en un mismo sitio todos los datos genealógicos de una familia, tanto personales como clínicos, y mostrarlos.

El sitio elegido es una aplicación web porque es una forma sencilla, amigable y accesible para cualquier usuario de introducir información de cada miembro de su familia y posteriormente ver esos datos gráficamente. Además, el usuario podrá tener actualizados sus datos personales y consultarlos siempre que quiera.

A la hora de elegir la herramienta más adecuada para realizar un proyecto de este tipo, surgen muchas opciones, lo cual puede suponer un problema de decisión. Se barajó la posibilidad de utilizar MongoDB [1] ya que ofrece flexibilidad y una optimización de consultas para grandes cantidades de datos, pero no contemplan la integridad de los datos, así que finalmente se ha preferido usar MySQL [2] junto con phpMyAdmin [3]. Como *framework* de desarrollo se ha elegido Spring [4] por su gran contenedor de dependencias.

1.2 Objetivos

Los objetivos generales del proyecto son:

1. Desarrollar y diseñar íntegramente una aplicación web comprensible, visual y fácil de usar.
2. Adquirir las capacidades necesarias y la solvencia técnica para desarrollar una aplicación web utilizando el lenguaje de programación Java, el *framework* Spring [5], y el motor de base de datos MySQL.

Por cada uno de los objetivos generales se pueden considerar los siguientes objetivos específicos:

1. Desarrollar y diseñar íntegramente la aplicación web.
   1.1. Diseñar la base de datos para poder guardar toda la información introducida por el usuario, así como para poder consultarla.
   1.2. Implementar y diseñar cada pantalla de la aplicación web.
   1.3. Generar el código indispensable para cumplir cada una de las funcionalidades que permitirán hacer uso de la aplicación.

2. Adquirir las capacidades necesarias y la solvencia técnica para desarrollar la aplicación web.
   2.1. Utilizar el lenguaje Java para crear código optimizado siguiendo las normas básicas de programación.
   2.2. Seguir la arquitectura del *framework* Spring para estructurar todas las clases del proyecto.
   2.3. Hacer uso de MySQL para crear la base de datos con ayuda de la herramienta phpMyAdmin.

1.3 Enfoque y metodología

Se puede enfocar este proyecto de varias maneras:

a. Encadenamiento de tareas. Sin haber terminado y probado una tarea, no se podrá continuar con la siguiente. Esta forma sería desarrollada mediante un control de versiones, el cual garantiza que se pueda volver a una versión anterior si la actual fallase.
b. Tareas simultáneas. Se pueden realizar tareas al mismo tiempo sin necesidad de terminar una para empezar la siguiente. Esto evitaría el estancamiento y permitiría avanzar, aunque sin la certeza de poder volver hacía atrás.

El método más adecuado sería el encadenamiento de tareas, ya que tener un control de versiones es muy importante para conocer y hacer un seguimiento de los posibles errores. Da la seguridad de poder volver hacía atrás si algo no funciona y encontrar una versión que sea correcta a partir de la cual seguir avanzando. El enfoque de las tareas simultáneas dificultaría este proyecto por no tener tareas terminadas en cada fase.

Este enfoque está destinado al desarrollo del producto final, es decir de la aplicación, lo cual no quiere decir que no se pueda solapar subtareas, como por ejemplo las pruebas de cada módulo que se vaya desarrollando o incluir nueva documentación en la memoria según se vaya avanzando.

1.4 Planificación

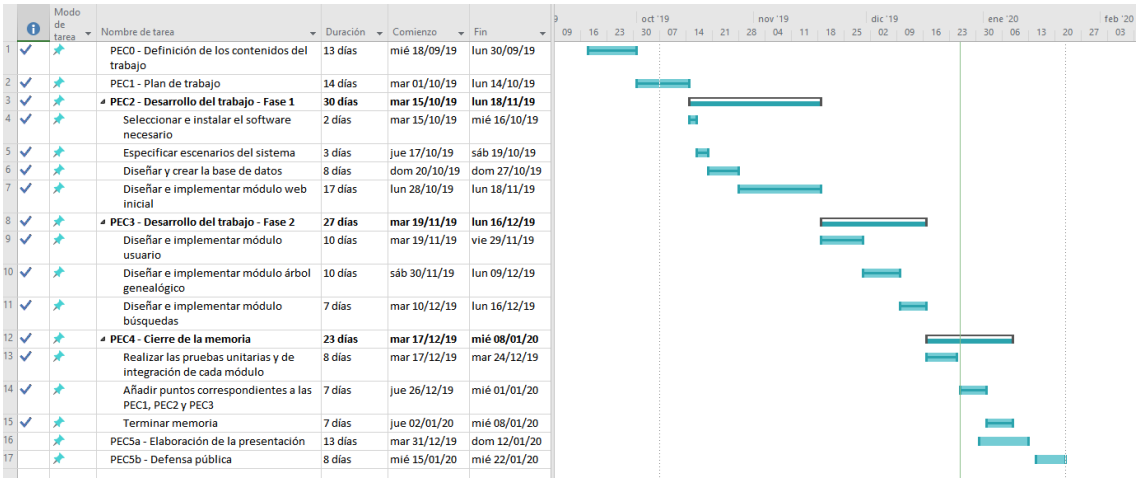
1.4.1 Tareas

Según los objetivos descritos anteriormente, el proyecto se divide en las siguientes tareas:

- Seleccionar e instalar el software necesario.
- Especificar los escenarios del sistema.
- Diseñar y crear la base de datos.
- Diseñar e implementar el módulo web inicial.
- Diseñar e implementar el módulo usuario.
- Diseñar e implementar el módulo árbol genealógico.
- Diseñar e implementar el módulo búsquedas.
- Realizar las pruebas unitarias y de integración de cada módulo.
- Elaboración de la memoria del proyecto.

1.4.2 Calendario

El tiempo para cada una de las tareas se cuenta en días, tal y como se muestra en éste Diagrama de Gantt [6] siguiendo el enfoque y método elegido:



	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin
1		PEC0 - Definición de los contenidos del trabajo	13 días	mié 18/09/19	lun 30/09/19
2		PEC1 - Plan de trabajo	14 días	mar 01/10/19	lun 14/10/19
3		**PEC2 - Desarrollo del trabajo - Fase 1**	**30 días**	**mar 15/10/19**	**lun 18/11/19**
4		Seleccionar e instalar el software necesario	2 días	mar 15/10/19	mié 16/10/19
5		Especificar escenarios del sistema	3 días	jue 17/10/19	sáb 19/10/19
6		Diseñar y crear la base de datos	8 días	dom 20/10/19	dom 27/10/19
7		Diseñar e implementar módulo web inicial	17 días	lun 28/10/19	lun 18/11/19
8		**PEC3 - Desarrollo del trabajo - Fase 2**	**27 días**	**mar 19/11/19**	**lun 16/12/19**
9		Diseñar e implementar módulo usuario	10 días	mar 19/11/19	vie 29/11/19
10		Diseñar e implementar módulo árbol genealógico	10 días	sáb 30/11/19	lun 09/12/19
11		Diseñar e implementar módulo búsquedas	7 días	mar 10/12/19	lun 16/12/19
12		**PEC4 - Cierre de la memoria**	**23 días**	**mar 17/12/19**	**mié 08/01/20**
13		Realizar las pruebas unitarias y de integración de cada módulo	8 días	mar 17/12/19	mar 24/12/19
14		Añadir puntos correspondientes a las PEC1, PEC2 y PEC3	7 días	jue 26/12/19	mié 01/01/20
15		Terminar memoria	7 días	jue 02/01/20	mié 08/01/20
16		PEC5a - Elaboración de la presentación	13 días	mar 31/12/19	dom 12/01/20
17		PEC5b - Defensa pública	8 días	mié 15/01/20	mié 22/01/20

Figura 1. Diagrama de Gantt

Cada día de trabajo es de 3 horas para poder compaginarlo con un trabajo a jornada completa, y no se distingue entre días laborales, festivos o fines de semana. Por asistencia a un viaje de empresa, en el mes de noviembre no se han dedicado horas todas las semanas a realizar el proyecto.

1.4.3 Hitos

Los hitos son aquellos momentos importantes y decisivos del proyecto en los que si las tareas previstas hasta ese punto no están terminadas, retrasaría los plazos de entrega de las demás tareas.

Hito	Fecha límite
Plan de trabajo	14/10/2019
Fase 1 del desarrollo del trabajo	18/11/2019
Fase 2 del desarrollo del trabajo	16/12/2019
Cierre de la memoria	08/01/2020
Elaboración de la presentación	12/01/2020

Defensa pública	22/01/2020

1.4.4 Análisis de riesgos

Todo proyecto tiene una serie de riesgos que vienen dados por la mala planificación, por imprevistos o por agentes externos. Algunos de ellos pueden ser los siguientes:

a. El tiempo estimado en la planificación no es suficiente para completar el trabajo.
b. Las tareas definidas no son adecuadas para cumplir todos los objetivos especificados.
c. No haber instalado bien el software elegido y provocar el retraso de una o varias tareas planeadas.
d. Falta de conocimiento del lenguaje Java y/o del motor de base de datos phpMyAdmin.
e. No conseguir resolver todas las funcionalidades necesarias para el correcto funcionamiento de la aplicación web.

Enumerados los posibles riesgos, se puede pensar en sus soluciones para que no haya ningún punto crítico que impida avanzar y terminar el proyecto.

Si el tiempo no fuese sufciente se puede reducir el trabajo de cada tarea proporcionalmente y adaptarlas para seguir cumpliendo los objetivos finales, aunque haya que renunciar a ciertas partes. En caso de que el software haga que se retrase el proyecto, ya sea por falta de compresión del mismo o por no estar bien instalado, se procedería a buscar una solución alternativa con una herramienta más conocida y ya instalada para completar esa tarea.

1.5 Breve sumario de productos obtenidos

Producto	Formato de entrega
Código fuente de la aplicación web	Zip con todos los archivos .java .jsp .js .css .xml .sql, y un readme.txt que incluye una guia básica de importación del proyecto y un link al repositorio GitHub con el histórico de *commits*
Memoria	Documentos .doc y .pdf
Presentación	Archivos .pptx .ppsx y .mp4

1.6 Breve descripción de los otros capítulos de la memoria

- Especificación de requisitos.
  Se especificarán los requisitos funcionales y no funcionales de cada módulo en que se divide el proyecto. Tanto los funcionales como los no

funcinalidades se basan en los objetivos descritos en capítulos anteriores.

- Entorno de trabajo.
  Se describirá el hardware y el software utilizado y se explicará en qué consisten esas tecnologías, además de mencionar el motivo por el que se han escogido para realizar este proyecto.

- Especificación del diseño.
  En este capítulo se hablará de todo lo referente al diseño de la aplicación web.
  - La estructura del sistema: organización de carpetas del *framework* Spring.
  - El modelo de datos: estructura de la base de datos con la que cuenta la aplicación web y sus relaciones entre tablas.
  - Características funcionales: todo lo que la aplicación puede y no puede ofrecer al usuario.

- Pruebas y resultados.
  Se explicará la estrategia que se ha seguido para la realización de todas las pruebas unitarias y funcionales del código del proyecto, así como la muestra de los resultados obtenidos.

- Conclusiones.
  Se expondrán las lecciones que se han aprendido en este trabajo, junto con una reflexión crítica sobre el logro de los objetivos planteados inicialmente y el seguimiento de la planificación. También se incluirán unas líneas de trabajo futuro que no se han podido explorar en este proyecto y que han quedado pendientes.

- Glosario.
  Se definirán los términos y acrónimos más relevantes utilizados dentro de la memoria.

- Bibliografía.
  Se hará una lista numerada de las referencias bibliográficas utilizadas dentro de la memoria.

2. Especificación de requisitos

2.1 Requisitos funcionales

Los requisitos funcionales están divididos en 2 grupos, según los módulos más importantes de la aplicación: usuario y árbol.

Usuario

- **RFU1** La aplicación web debe ser capaz de almacenar en su base de datos todos los datos del usuario que se ha registrado.
- **RFU2** La aplicación tiene que ser capaz de saber qué usuario ha iniciado sesión.
- **RFU3** La aplicación debe ser capaz de modificar los datos de la cuenta del usuario que lo solicite.
- **RFU4** La aplicación tiene que ser capaz de mostrar al usuario todos los datos de su cuenta.

Árbol

- **RFA1** La aplicación debe permitir al usuario incluir nuevos miembros en su árbol, modificarlos y eliminarlos siempre que quiera.
- **RFA2** La aplicación debe ser capaz de mostrar el árbol completo con todos los miembros que haya añadido el usuario.
- **RFA3** La aplicación debe ser capaz de realizar búsquedas por el historial médico de cada miembro y mostrar los resultados.

2.2 Requisitos no funcionales

En este apartado se diferencian varios tipos de requisitos no funcionales, los cuales describen las condiciones del sistema.

Usabilidad

- **RNFU1** La aplicación tiene que ser comprensible y fácil de usar.
- **RNFU2** El usuario tiene que ser capaz de visualizar su árbol genealógico actualizado y completo.
- **RNFU3** La aplicación ha de cargar todo su contenido lo más rápido posible.

Entorno

- **RNFE1** La aplicación debe permitir el acceso al usuario desde cualquier navegador sin problemas de visualización.

Escalabilidad

- **RNFES1** El sistema debe permitir desarrollar nuevas funcionalidades a partir de las ya existentes.

3. Entorno de trabajo

En este apartado se describe el hardware y el software que se utiliza en el proyecto.

Hardware

- PC Sony Vaio equipado con Windows 8.1

Software

- Entorno de desarrollo Eclipse Luna [7]: es una herramienta *open-source* que proporciona al usuario un entorno donde desarrollar aplicaciones, sobre todo mediante el lenguaje de programación Java, con plugins disponibles a través de instalación externa para adaptarse a las necesidades del desarrollador y poder integrarse con cualquier *framework*.
- *Framework* Spring v4.3: está construido para la plataforma Java y consiste en un modelo de estructura de carpetas basada en MVC. Su conexión con base de datos la hace el módulo de controladores de cada clase por medio de anotaciones, desde la vista se pulsa un botón y la información recogida viaja a la interfaz de servicio que conecta con el controlador, y éste se encarga de llamar a base de datos para guardar dicha información. Se ha elegido este *framework* porque tiene una estructura muy definida, la cual te permite añadir cuantos módulos nuevos quieras sin interferir con los anteriores.
- Contenedor de distribución de aplicaciones XAMPP [8] v3.2.1: es una herramienta de desarrollo que permite probar el trabajo en tu propio ordenador sin necesidad de acceder a internet. Está formado por un servidor web Apache y una base de datos MySQL [9].
- Sistema de controlador de versiones Git [10] v2.6.3: permite gestionar repositorios, alojados en GitHub [11] en este caso, desde cualquier lugar. También permite tener un control sobre cómo ha cambiado nuestro proyecto con el paso del tiempo, y volver a otra versión más antigua de una manera realmente simple.
- Herramientas de ofimática Microsoft Office 365.
- Herramienta de diseño gráfico Adobe Photoshop CS6 [12].

4. Especificación del diseño

4.1 Estructura del sistema

Este proyecto se basa en una estructura de directorios y ficheros impuesta por el *framework* Spring, la cual se desglosa como se indica a continuación:

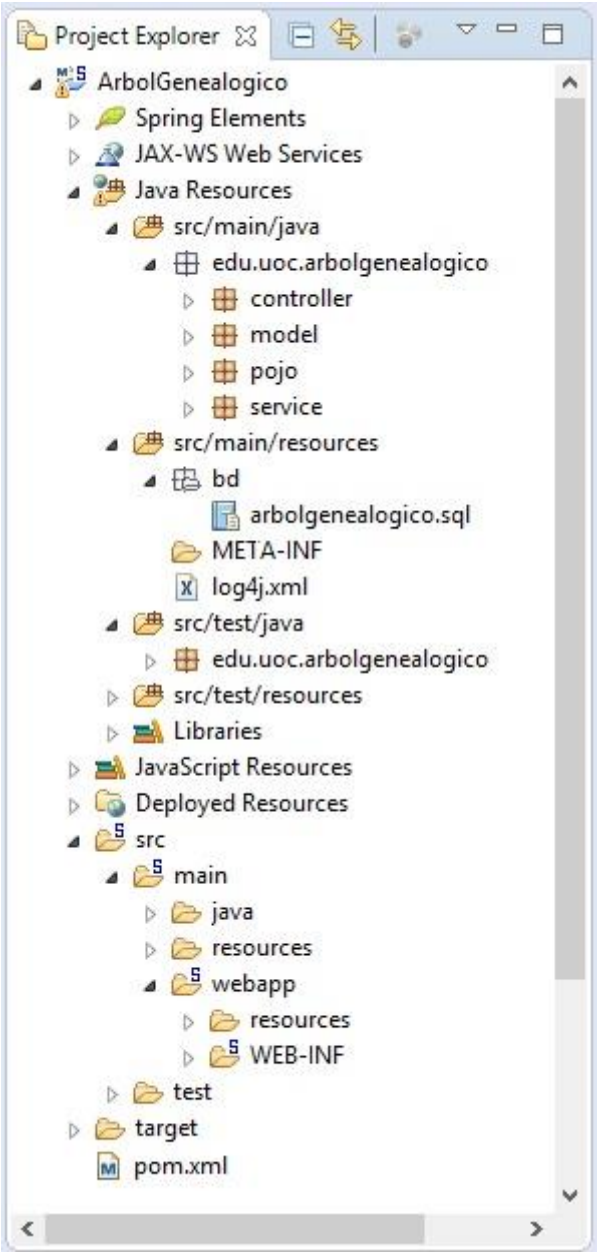


Figura 2. Estructura del proyecto

Dentro de Java Resources se encuentra todo el código de la aplicación, dividido en diferentes paquetes o carpetas.

- Controller: todas las clases a las cuales se les llama cada vez que se quiere realizar una acción dentro de la aplicación; las que actuarán de conexión entre los demás módulos.

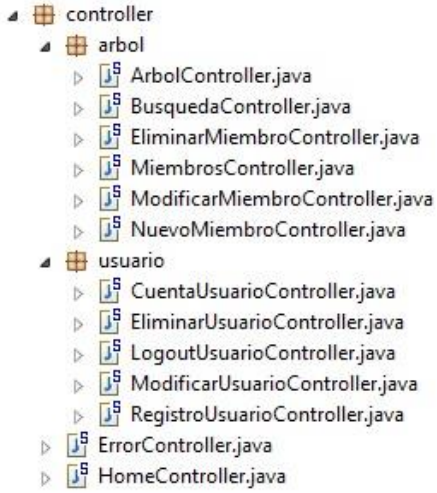


Figura 3. Paquete Controller

- Model: paquete referente a la base de datos; cada clase implementará los métodos que permitirán conectarse con las tablas de base de datos.

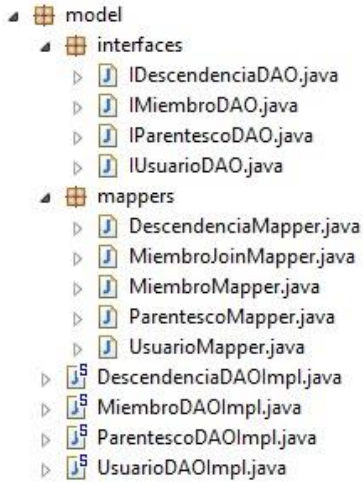


Figura 4. Paquete Model

- Pojo: aquí se alojan las clases básicas, con todos sus atributos y métodos.

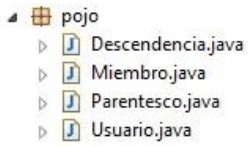


Figura 5. Paquete Pojo

- Service: todas las clases de este módulo actuarán de intermediario entre los controladores y los modelos.

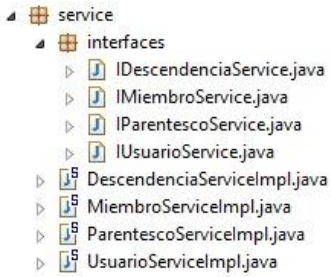


Figura 6. Paquete Service

Además, al tratarse de una aplicación web, en la carpeta webapp se pueden encontrar todos los archivos relacionadas con el diseño de las pantallas [13]:

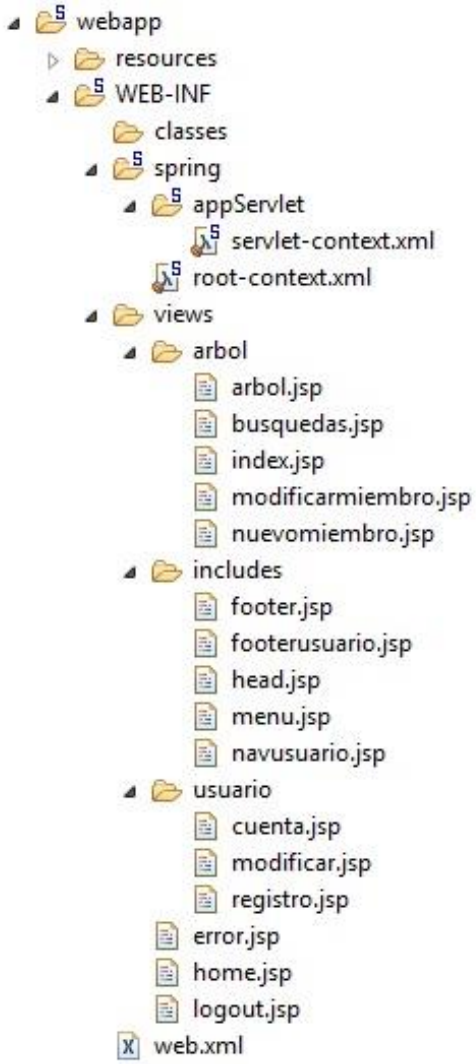


Figura 7. Estructura Webapp

4.2 Modelo de datos

A continuación se muestra el modelo de datos entidad-relación con el que cuenta la base de datos de este proyecto:

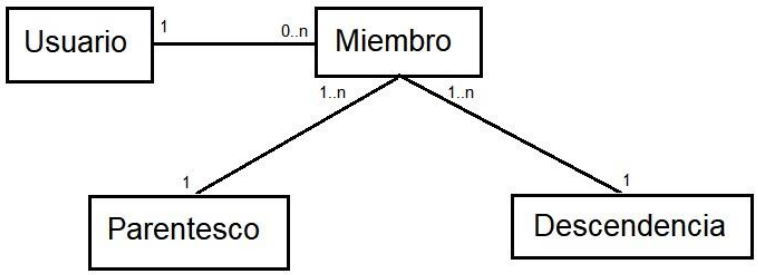


Figura 8. Diagrama entidad-relación

Las entidades indicadas en el diagrama corresponden a las tablas de la base de datos.

- Usuario
  - Id: identificativo único generado automáticamente
  - *Username*: nombre de usuario identificativo en el inicio de sesión
  - Nombre: nombre del usuario
  - Apellidos: apellidos del usuario
  - Contraseña: contraseña del usuario
  - Imagen: ruta de la foto correspondiente al usuario
  - Año nacimiento: necesario para calcular la edad
  - Online: 1 o 0 según haya iniciado sesión el usuario o no
  - Última conexión: fecha de la última vez que el usuario ha iniciado sesión

- Parentesco
  - Id: identificativo único generado automáticamente
  - Descripción: nombre dentro de la familia (hijo, padre, abuelo…)
  - Rama: rama a la que pertenece el miembro según la descripción

- Descendencia
  - Id: identificativo único generado automáticamente
  - Tipo rama: rama paterna o materna descendiente del árbol según el parentesco

- Miembro
  - Id: identificativo único generado automáticamente
  - Nombre: nombre del miembro
  - Apellido: apellido del miembro
  - Imagen: ruta de la foto correspondiente al miembro asignada de un color diferente según el parentesco
  - Año nacimiento: necesario para calcular la edad
  - Año defunción: necesario para calcular la edad, vacío si no procede

- Historial médico: patologías conocidas del miembro
- Usuario: identificativo único del usuario al que pertenece el árbol
- Parentesco: identificativo único del parentesco
- Descendencia: identificativo único de la descendencia

4.3 Características funcionales

Cuando el usuario entre en la aplicación web se encontrará con la página principal, en la que se verá, de una forma muy visual y simple, los pasos que hay que seguir para utilizar dicha aplicación. Lo que se pretende con esto es que el usuario encuentre todo lo que busca en un solo movimiento por la aplicación.

Lo primero será iniciar sesión o registrarse.

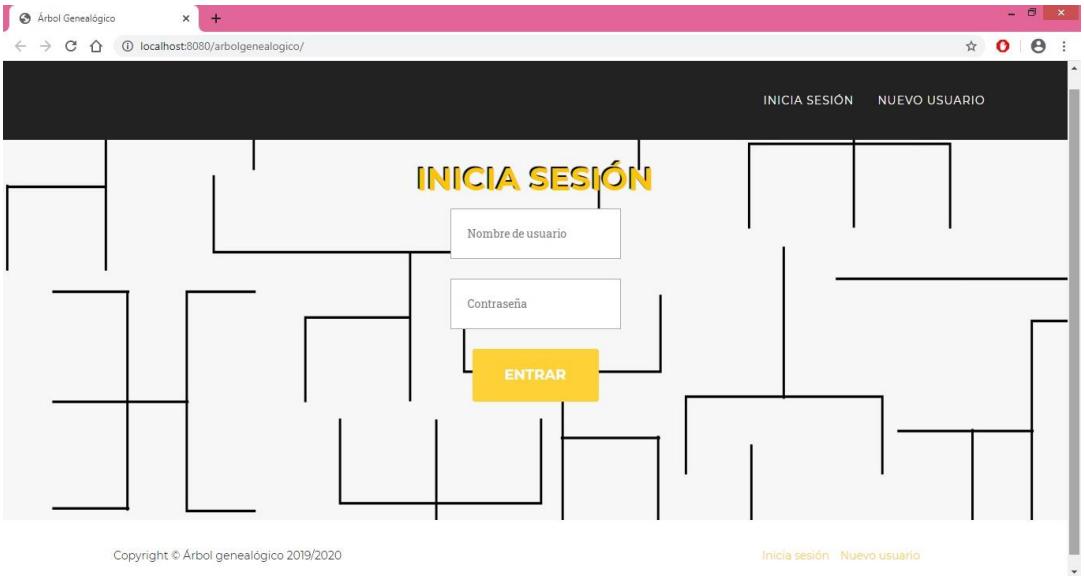


Figura 9. Pantalla Inicio sesión

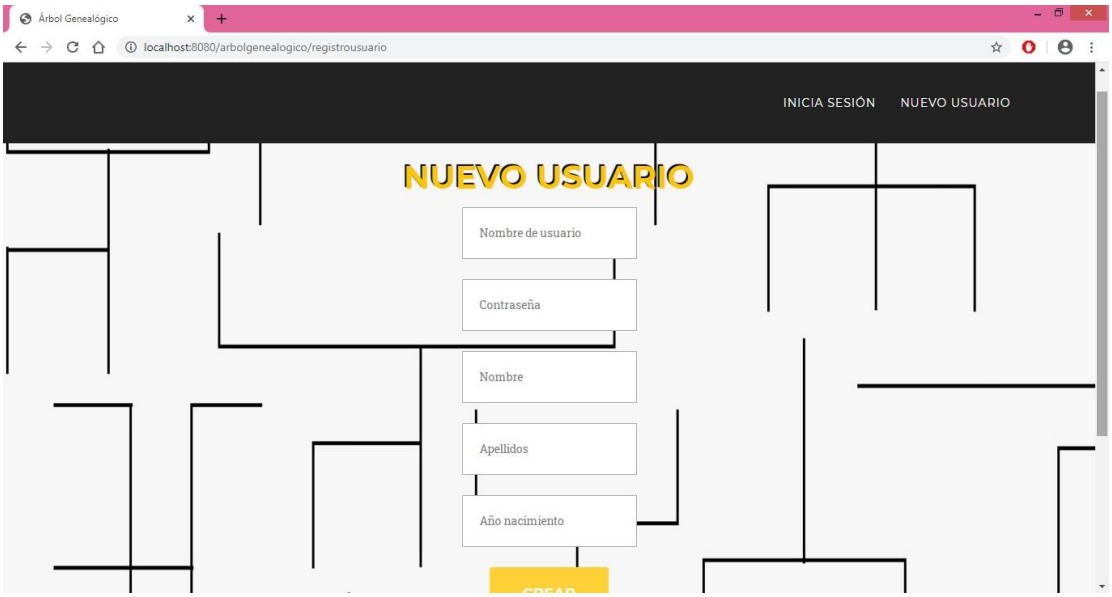


Figura 10. Pantalla Registro

El menú principal de un usuario que ha iniciado sesión o se ha registrado consta de dos opciones útiles para poder navegar por la aplicación.

La primera es “Mi cuenta”, en la que se puede consultar los datos del usuario, modificarlos y borrarlos, así como ver la lista de miembros que ha añadido a su árbol genealógico.

Figura 11. Pantalla Cuenta de usuario inicial

Inicialmente, la lista de miembros estará vacía. A medida que el usuario añada nuevos, irán apareciendo en la lista de miembros de la cuenta.

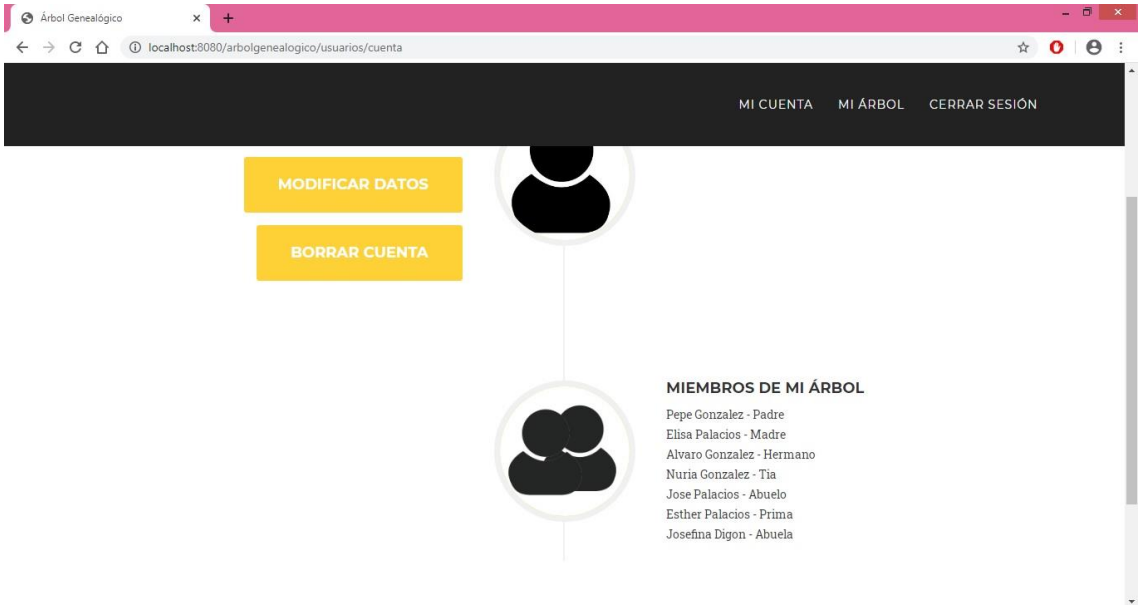


Figura 12. Pantalla Cuenta de usuario con lista de miembros

La segunda opción es “Mi árbol”, en la que se puede gestionar todo lo referente al árbol genealógico del usuario y sus miembros; además de realizar búsquedas por historial médico.

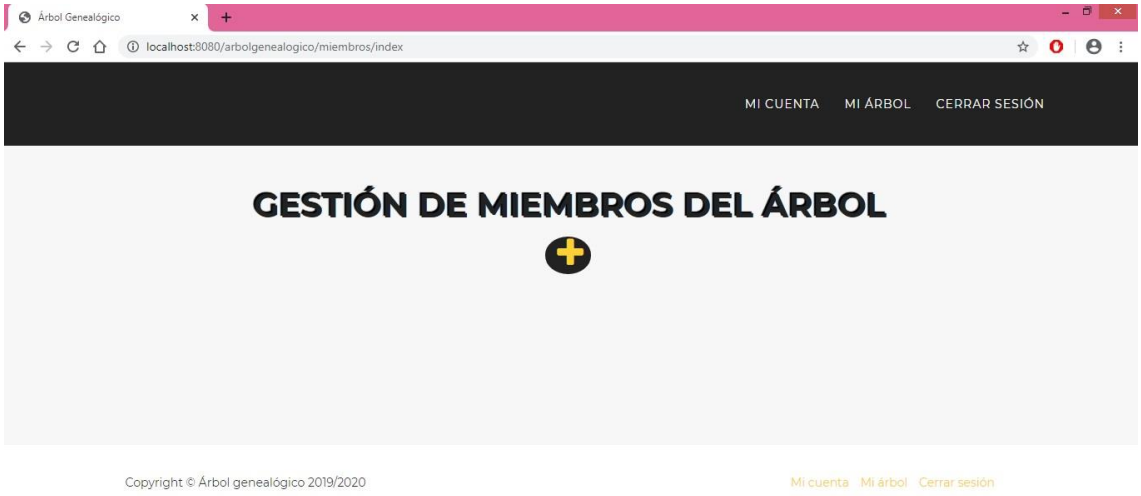


Figura 13. Pantalla Gestión de miembros vacía

El usuario podrá añadir todos los miembros que desee para formar su árbol, así como poder modificar sus datos y eliminarlos. Si el usuario no hay añadido ningún miembro todavía, el botón de añadir será lo único que pueda ver en ese apartado.

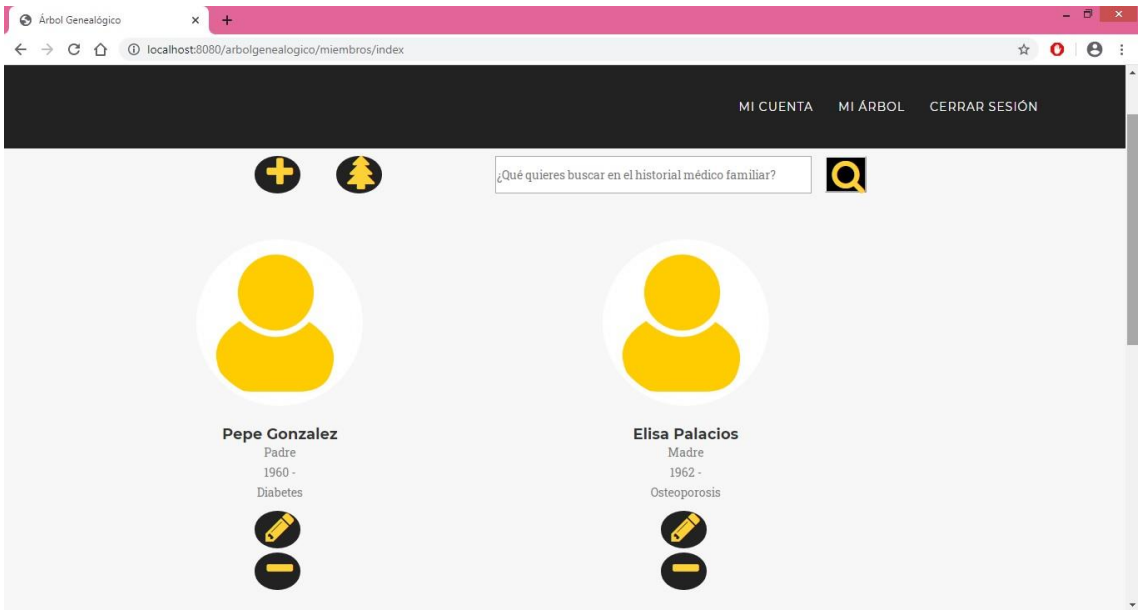


Figura 14. Pantalla Gestión de miembros

Una vez haya al menos un miembro, el usuario tendrá también la opción de poder visualizar el árbol completo. Este árbol personalizado se irá actualizando con toda la información cada vez que se añada, modifique o elimine un miembro. La edad de cada persona será calculada según el año actual con la fecha de nacimiento aportada, ya que es un valor variable.

El usuario podrá visualizar el árbol siempre que quiera, no hace falta que esté completo, se irá actualizando según añada, modifique o elimine miembros.

Figura 15. Pantalla Árbol simple

La clase de árbol elegido para hacer esta interfaz se basa en que el usuario siempre será la base de la que surgen generaciones anteriores.

Se ha separado en dos figuras el árbol con generaciones de ejemplo (véase Figura 16 y Figura 17) para evitar aplicar zoom a la pantalla y poderse leer mejor la información.

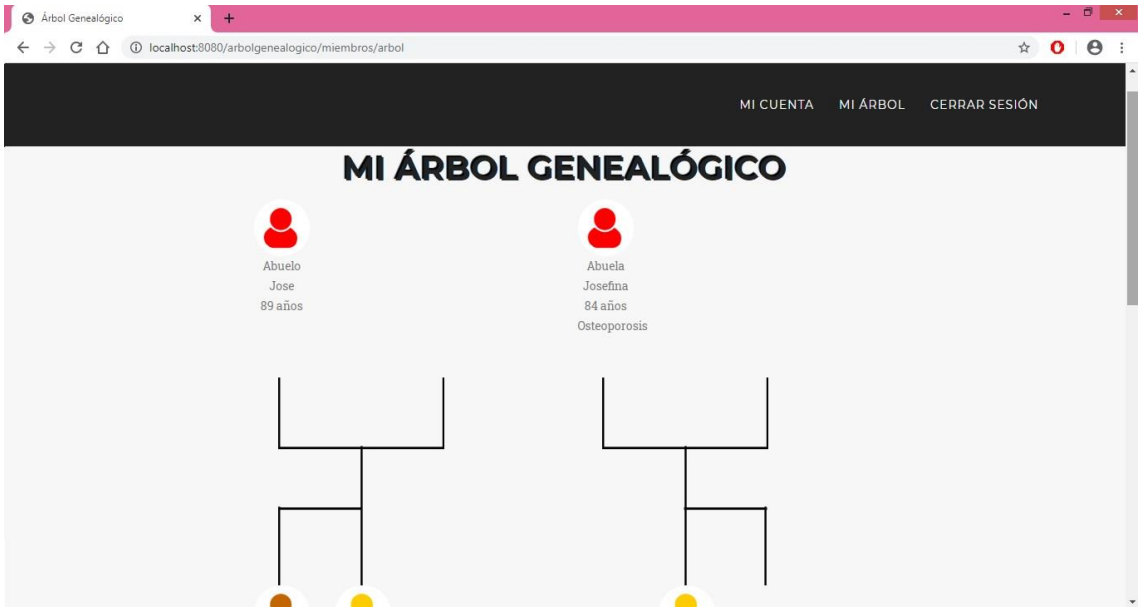


Figura 16. Pantalla Árbol con generaciones 1

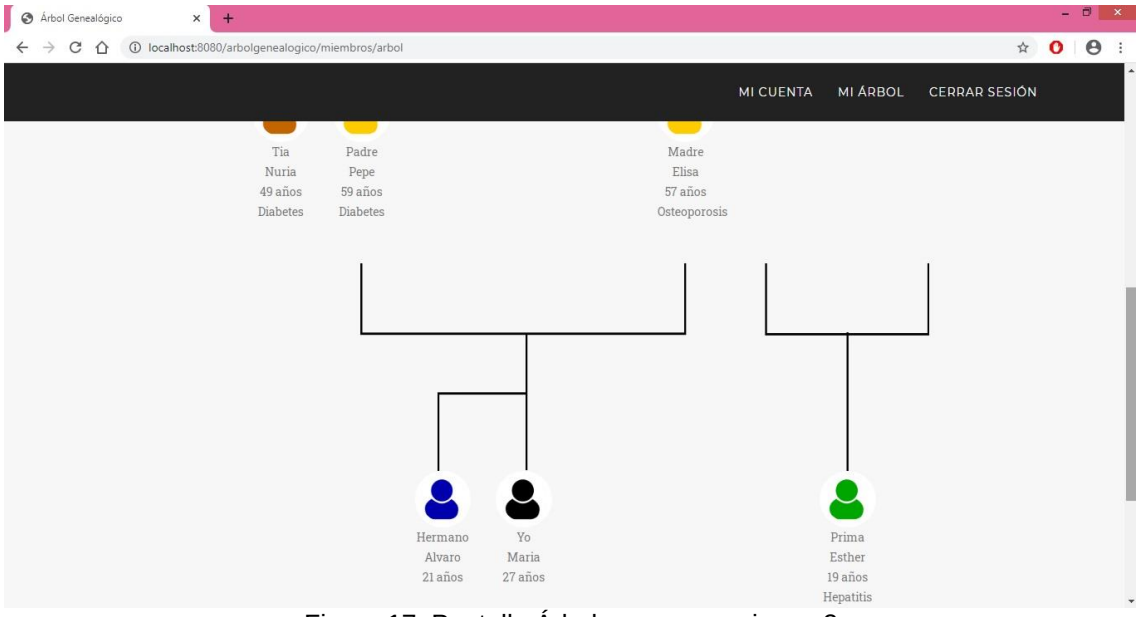


Figura 17. Pantalla Árbol con generaciones 2

Para realizar búsquedas por patologías, según las registradas en el historial médico de cada miembro, basta con introducir la palabra deseada y mostrará los miembros que en su historial se encuentra alguna coincidencia.

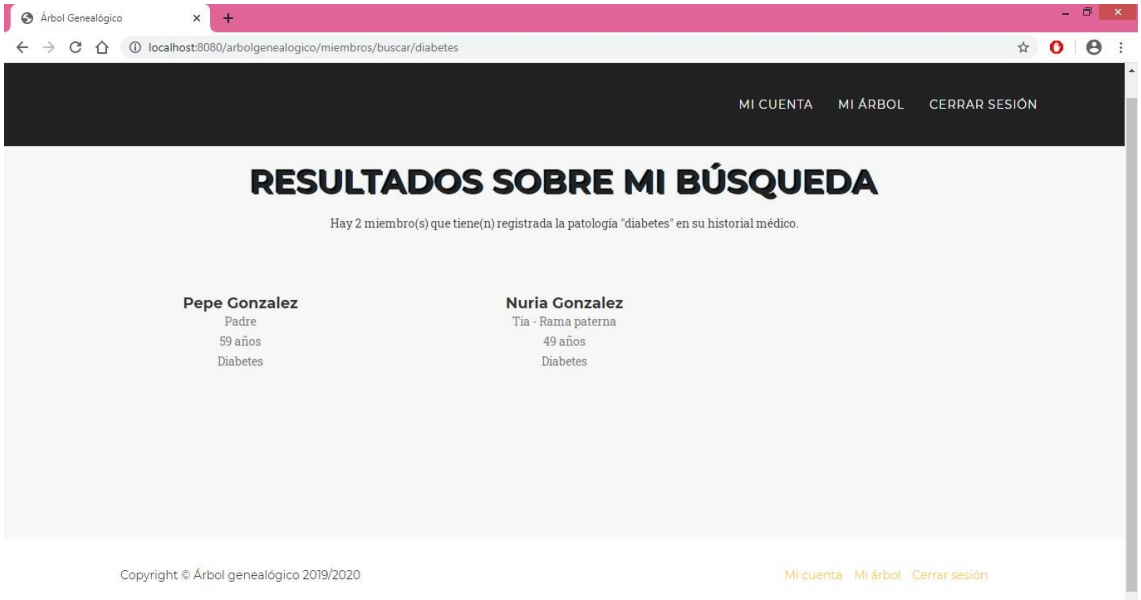


Figura 18. Pantalla Búsqueda Diabetes

5. Pruebas y resultados

Para comprobar que toda la aplicación funciona correctamente, tal y como se ha programado, se han hecho pruebas unitarias y funcionales mediante Junit [14], además de elegir a una persona externa al proyecto para que probase como si de un usuario se tratase. Tanto las pruebas de código como las personales son esenciales para poder obtener resultados válidos.

5.1 Estrategia

Los test son imprescindibles para controlar la calidad del código de la aplicación. Éstos test se dividen en dos tipos: unitarios y funcionales.

Los unitarios prueban pequeñas parte del código. El objetivo de estas pruebas es aislar cada parte del programa y demostrar que las partes de forma individual son correctas. Una prueba unitaria proporciona un contrato escrito que la pieza de código debe satisfacer. Como resultado, las pruebas unitarias encuentran problemas en las fases iniciales del desarrollado.

Un ejemplo de prueba unitaria para esta aplicación es validar que el usuario que se cree siempre tenga imagen asociada.

Figura 19. Test unitario Usuario

Las funcionales, en cambio, prueban partes enteras de la aplicación.

5.2 Resultados

Mediante las pruebas unitarias y funcionales se obtienen diferentes resultados que, tanto los negativos como los positivos, puedes ayudar a mejorar el código de la aplicación. En el caso de esta, las pruebas unitarias son más importantes que las funcionales, ya que todo está hecho por módulos y es más fácil captar los errores en trozos de código más reducido.

Como todos los frameworks de pruebas unitarias, JUnit utiliza *assertions* para verificar que el comportamiento de una unidad de código es el esperado.

En la siguiente figura se puede ver que el test ha dado positivo, esto quiere decir que la aserción ha devuelvo “verdadero” a la comparación que se indicó.

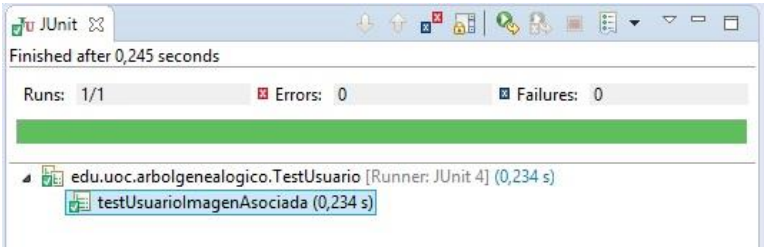


Figura 20. Resultado test unitario Usuario

6. Conclusiones

La idea de este proyecto fue propuesta por el director y desarrollada siguiendo esa idea con aportaciones propias del autor.

6.1 Retos superados

Todos los objetivos, tanto principales como secundarios, se han cumplido dentro de los plazos especificados al iniciar el proyecto.

Se ha desarrollado y diseñado una aplicación web comprensible y fácil de usar, haciendo posible la visualización de árboles genealógicos personalizados por el usuario, además de haber proporcionado al usuario un alojamiento web para todos sus datos y los de sus miembros familiares.

Se han adquirido las capacidades necesarias y la solvencia técnica adecuada para crear un proyecto desde cero utilizando el *framework* Spring y también para generar documentación a partir de su código y de su situación.

La primera fase del desarrollo del proyecto ha sido la más complicada y decisiva, ya que, al haber dudas sobre qué tecnología era mejor utilizar, se retrasó más de lo esperado; aunque se lograse al final cumplir con las fechas y objetivos.

6.2 Líneas de trabajo futuro

Un proyecto como este es difícil completarlo en 3 meses y siempre queda mucho trabajo futuro pendiente; podría abarcar mucho recorrido más.

Por un lado, estaría el tema del diseño de la interfaz de usuario de la aplicación, que puede ser modificado y adaptado en cualquier momento para satisfacer las tendencias del momento.

Por otro lado, en cuanto a funcionalidad, se pueden añadir más características, ya que todo el proyecto se ha hecho de manera que cualquier programador pueda seguir con él sin tener que empezar de nuevo o tener que modificar mucho código para introducir nuevas funcionalidades. Se puede empezar con la posibilidad de descargar la imagen del árbol genealógico completo; seguir con nuevas funcionalidades como poder subir la fotografía del miembro que se añada o como ver estadísticas de las patologías familiares según las incluidas en el historial médico de cada miembro; y terminar con la ampliación de la visualización del árbol de una forma más detallada y compleja. Por supuesto, cabe mencionar también, el despliegue de la aplicación en un servidor, ya que el alcance de este proyecto solo contempla el funcionamiento en un servidor local.

7. Glosario

Framework

Arquitectura de software que modela las relaciones generales de las entidades del dominio, y provee una estructura y una metodología especial de trabajo.

Open-source

Software de código abierto que se desarrolla de manera pública y colaborativa. Los usuarios pueden estudiar, cambiar y distribuir este software a cualquier persona y para cualquier propósito.

Commit

Se refiere, en el contexto de la informática, a la idea de confirmar un conjunto de cambios provisionales de forma permanente. Se suele usar cuando se quieren subir a un repositorio los cambios hechos en los archivos del código de un proyecto o al final de una transacción de base de datos.

MVC

Modelo-vista-controlador. Es un patrón de diseño de software comúnmente utilizado para desarrollar interfaces de usuario que divide la lógica del programa relacionada en tres elementos interconectados.

Assertion

Una aserción es un predicado (una sentencia verdadero-falso) incluido en un programa como indicación de que el programador piensa que dicho predicado siempre se cumple en ese punto de flujo de programa. Las aserciones suelen ser útiles para especificar programas y para razonar la corrección de los mismos.

8. Bibliografía

[1] “La base de datos líder del mercado para aplicaciones modernas | MongoDB”, https://www.mongodb.com/es (consultado el 20/09/19).

[2] “NoSQL vs SQL: principales diferencias y cuándo elegir cada una de ellas”,https://pandorafms.com/blog/es/nosql-vs-sql-diferencias-y-cuando-elegir-cada-una/ (consultado el 15/10/19).

[3] “phpMyAdmin”, https://www.phpmyadmin.net/ (consultado el 15/10/19).

[4] “Spring”, https://spring.io/ (consultado el 18/09/19).

[5] “¿Qué es Spring Framework?”, https://www.genbeta.com/desarrollo/que-es-spring-framework (consultado el 19/09/19).

[6] “La mejor herramienta para el Diagrama de Gantt”, https://obsbusiness.school/es/blog-project-management/planificacion-de-las-actividades-y-tiempo-de-un-proyecto/la-mejor-herramienta-para-el-diagrama-gantt-de-excel-ms-project (consultado el 01/10/19).

[7] “Eclipse Luna | The Eclipse Foundation”, https://www.eclipse.org/luna/ (consultado el 15/10/19).

[8] “XAMPP. Apache Friends”, https://www.apachefriends.org/es/index.html (consultado el 15/10/19).

[9] ELMASRI / NAVATHE, “Fundamentos de Sistemas de Bases de Datos, 3ª edición”, *Addison-Wesley*, 2002.

[10] “Git”, http://git-scm.com/about (consultado el 15/10/19).

[11] “GitHub - Build software better, together”, https://github.com (consultado el 15/10/19).

[12] JOSE MARÍA DELGADO, “Photoshop CS6”, *Anaya Multimedia*, 2012.

[13] IAN LLOYD, “Build your own web site the right way using HTML & CSS”, *Sitepoint (Collingwood, Australia)*, 2006.

[14] “Testing Spring con JUnit 4 – danielme.com”, https://danielme.com/2017/07/17/tutorial-spring-testing-junit-4/ (consultado el 18/12/19).