

UNIVERSIDAD DE CASTILLA LA MANCHA
ESCUELA SUPERIOR DE INFORMÁTICA

Calculadora de Riesgo Cardiovascular

Gestión de Sistemas de Información
Curso 2018/2019

Sergio González Velázquez
Antonio Rubio Menchero



ÍNDICE GENERAL

Resumen	I
1. Introducción	1
1.1. Propuesta de Proyecto	1
1.2. Descripción y justificación del trabajo	1
1.3. Herramientas utilizadas	2
2. Fase I: Análisis	4
2.1. Factores de Riesgo cardiovascular	4
2.2. Estimación del Riesgo cardiovascular	5
2.3. Implementando el Algoritmo de Framingham	5
3. Fase II: Diseño	7
3.1. Presentación. Prototipo de la GUI	7
3.2. Dominio	9
3.3. Persistencia. BBDD Local	11
4. Fase III: Implementación	13
4.1. Inicio de sesión y registro de un nuevo usuario	14
4.2. Barra de navegación inferior	15
4.2.1. Ventana de Estado	15
4.2.2. Ventana de nuevo Cálculo	15
4.2.3. Ventana de Perfil	16
4.3. Action Bar y Acerca De...	16
A. Tablas de Framingham en Java	17
Bibliografía	21

CAPÍTULO 1

INTRODUCCIÓN

Si algo valoran especialmente en su trabajo cotidiano los médicos de familia es la aproximación a la incertidumbre. Cuando se encuentran a su alcance las pruebas diagnósticas en cada caso y disponen de guías de tratamiento actualizadas, basadas en evidencias y adaptadas a su quehacer cotidiano, experimentan cómo disminuye la incertidumbre propia de su trabajo. Esta situación queda destacada con especial relevancia cuando se trata de atender a los pacientes que sufren las enfermedades cardiovasculares. En España, las enfermedades del aparato circulatorio constituyeron la principal causa de muerte para el conjunto de la población, representando el 36 % de todas las defunciones.

1.1. PROPUESTA DE PROYECTO

Según diversos estudios presentados en el Congreso de eSalud, la Inteligencia Artificial puede ayudar al diagnóstico y tratamiento de enfermedades cardiovasculares, la primera causa de morbilidad y gasto sanitario.

En este sentido, nos gustaría realizar un prototipo de una aplicación para detectar el riesgo de enfermedad cardiovascular en función del peso, edad, alimentación, frecuencia con que se práctica deporte y otros factores a estudiar. El programa deberá analizar la información introducida por el paciente y, comparando con su base de datos, establecer las probabilidades de que sufra una enfermedad cardiovascular. Es decir, el objetivo de este trabajo es hacer medicina predictiva identificando signos que preceden a algunas enfermedades cardíacas concretas.

La idea ha sido tomada del desafío social propuesto por Mary Luz Mouronte López para la edición 2019 del programa HackForGood.

<https://hackforgood.net/reto-ufv-1-riesgo-de-enfermedad-cardiovascular/>

1.2. DESCRIPCIÓN Y JUSTIFICACIÓN DEL TRABAJO

En el trabajo teórico de la asignatura Gestión de Sistemas de Información, hemos investigado sobre la Inteligencia Artificial y su aplicación en Medicina. Hablábamos de los Sistemas Expertos como uno de los primeros resultados de la Inteligencia Artificial, porque logran resolver problemas a través del conocimiento, de forma similar a como lo hace un ser humano. Decíamos también, que una de las aplicaciones más importantes de los sistemas expertos tiene lugar en Medicina, donde pueden utilizarse principalmente para diagnóstico médico.

Por otro lado, concluimos nuestro trabajo haciendo una pequeña reflexión sobre el futuro que nos espera gracias a los avances de la Inteligencia Artificial. En este sentido, proponíamos un cambio de chip que nos conduzca a marcarnos otros objetivos más ambiciosos: hacer **medicina preventiva**

intentando evitar la aparición del mayor número de enfermedades, basándonos en actuaciones y consejos médicos que se pueden anticipar gracias a los avances de la tecnología.

La combinación de Sistemas Expertos con la idea de medicina preventiva constituye la raíz del origen de nuestro proyecto de laboratorio. De manera más concreta, hemos intentado desarrollar una aproximación de Sistema Experto en forma de aplicación Android que se pueda utilizar para prevenir y reducir el número de enfermedades cardiovasculares. Decimos que se trata de una aproximación de Sistema Experto, porque hemos desarrollado un algoritmo (**motor de inferencia**) que permita relacionar información concreta sobre el estilo de vida de un usuario (**base de hechos**), con una serie de conocimiento general sobre los factores que influyen en el riesgo de sufrir una enfermedad cardiovascular (**base de conocimiento**).

Tal y como explicábamos en el trabajo teórico, la base de conocimiento de un SE está formada por todo el conocimiento disponible sobre el campo en el que se desarrolla la aplicación. En nuestro caso particular, este conocimiento lo encontramos en el **modelo para el cálculo de la probabilidad de riesgo cardiovascular de Framingham**.

1.3. HERRAMIENTAS UTILIZADAS

Entorno de Desarrollo

Hemos decidido implementar nuestra aplicación en Android haciendo uso del entorno de desarrollo **Android Studio** y del lenguaje de programación **Java**. La principal motivación de esta decisión es que nos gustaría aprender a programar en esta plataforma, pues creemos que es bastante versátil, compatible con una inmensidad de dispositivos y que nos ofrece la posibilidad de desarrollar aplicaciones que estén disponibles para millones de usuarios en todo el mundo. Por tanto, consideramos que es muy recomendable aprender a programar en Android y nos puede resultar de utilidad en el futuro.

Además, otra de las razones que nos conducen a decantarnos por Android es que, al tratarse de una plataforma de código abierto, podemos encontrar una gran cantidad de publicaciones que nos ayudarán a desarrollar nuestro trabajo.

Repositorio de código

Hemos utilizado un repositorio de código privado en Github que nos ha permitido de tener un control de las versiones de nuestra práctica, además de ofrecer la posibilidad de trabajar de forma paralela todos los miembros del grupo de trabajo.

<https://github.com/SergioGonzalezVelazquez/ProyectoGSI>

Base de Datos

Para añadir persistencia a los datos de nuestra aplicación Android, hemos utilizado un motor ligero de bases de datos de código abierto como es **SQLite**, una tecnología muy cómoda para los dispositivos móviles. Su simplicidad, rapidez y usabilidad permiten un desarrollo muy amigable. El conector que hemos utilizado para que nos proporcione los mecanismos básicos para la relación entre la aplicación Android y la información, es **SQLiteOpenHelper**.

Librerías de terceros

- **HelloCharts.** Librería para implementar gráficos estadísticos compatible con API 8+(Android 2.2).

<https://github.com/lecho/hellocharts-android>

- **JustifiedTextView.** Librería que permite justificar texto en una aplicación Android.

<https://github.com/amilcar-sr/JustifiedTextView>

- **CircleImageView.** Librería que permite introducir imágenes con forma circular.

<https://github.com/hdodenhof/CircleImageView>

CAPÍTULO 2

FASE I: ANÁLISIS

Hasta hace algo más de una década, las recomendaciones clínicas en la prevención cardiovascular iban dirigidas fundamentalmente al manejo independiente de sus factores de riesgo. Con este enfoque, y de forma sistemática, se fueron elaborando guías para el abordaje de cada uno de los factores de riesgo. Sin embargo, nuestro trabajo se apoya en estudios que toman los factores de riesgo como un conjunto para realizar una estimación precisa de la probabilidad de sufrir una enfermedad cardiovascular.

2.1. FACTORES DE RIESGO CARDIOVASCULAR

Los factores de riesgo cardiovascular son condicionantes ligados a estilos de vida que incrementan la probabilidad de padecer o morir por enfermedad vascular en aquellas personas en las que inciden. Se catalogan como tales cuando cumplen unos requisitos que permiten establecer una relación de causa-efecto con respecto a la enfermedad vascular. Se pueden clasificar como se muestra en la siguiente figura. Subrayados en amarillo se muestran los que utiliza el algoritmo de Framingham, tal y como se explicará más adelante.

Factores de Riesgo Cardiovascular Mayores	Factores de Riesgo Cardiovascular Bien Validados	Otros Factores de Riesgo Cardiovascular
Antecedentes Personales de ECV	Sobrepeso / Obesidad	Factores lipídicos:
Tabaquismo	Inactividad física	- Triglicéridos
Hipertensión Arterial	Síndrome Metabólico	- Apolipoproteínas
Diabetes Mellitus	GBA / TAG	- Lipoproteína a
LDLc elevado	Historia familiar de EVA prematura	- Subfracciones lipoproteicas
HDLc disminuido	Aterosclerosis subclínica	Factores no lipídicos:
Edad / Sexo	Microalbuminuria / ERC	- Resistencia a la insulina
	Hipertrofia Ventricular Izda.	- Marcadores protombóticos
	Fibrilación Auricular	- Marcadores proinflamatorios
	Factores genéticos y raciales	- Alcohol
		- Score Cálcico
		Dieta aterogénica
		Estrés socioeconómico / psicosocial
		SHAOs

Figura 2.1: Factores de Riesgo Cardiovascular (FRCV) [11]

Estos factores de riesgo cardiovasculares, ya sea de forma aislada o como sucede con mucha mayor frecuencia, en combinación, explican la mayoría de casos de enfermedad vascular o de muerte que ocurre en individuos de alto riesgo y una proporción considerable de casos en la población general.

2.2. ESTIMACIÓN DEL RIESGO CARDIOVASCULAR

El Riesgo Cardiovascular expresa la probabilidad de sufrir un evento de la enfermedad vascular en un determinado período de tiempo, generalmente 5 o 10 años. La explicación en la práctica de este hecho es que de 100 personas con un mismo porcentaje de riesgo cardiovascular (Ej. RCV de 21 %), de esas 100 personas, 21 desarrollaran una Enfermedad Vascular en los próximos 5 o 10 años.

Para la prevención cardiovascular es necesaria una valoración conjunta de los factores de riesgo mediante la estimación del riesgo cardiovascular del individuo. Esto, permite definir niveles de riesgo con los que el personal sanitario y los propios pacientes pueden valorar cómo se modifica el mismo a medida que se van logrando los objetivos pactados.

Existen diversos métodos para estimar el RCV y ninguno de ellos es perfecto. Sin embargo, se ha comprobado que son las **tablas de Framingham** las que mejor se comportan a la hora de predecir la aparición de eventos cardiovasculares:

- Tienen **riesgo cardiovascular alto** los pacientes que reúnen una puntuación de Framingham de 22 o superior, lo que supone una probabilidad de padecer un evento cardiovascular en los próximos 10 años superior al 20 %. También consideraremos en riesgo alto a aquellos que ya han sufrido un episodio cardiovascular o presentan diabetes.
- Se califican de **riesgo cardiovascular moderado** aquellos pacientes con factor de riesgo (hipertensión arterial o tabaquismo) y una puntuación inferior al 22, y consiguientemente una probabilidad de episodio isquémico inferior al 20 % en los próximos 10 años.
- Son de **riesgo cardiovascular bajo** los pacientes, independientemente de la edad y el sexo, sin factores de riesgos reconocidos.

El cálculo de valoración al que nos referimos se reproduce en las tablas 3.3(a) y ???. Es importante tener en cuenta que existen factores de riesgo cardiovascular no incluidos en la tabla, como el sedentarismo, obesidad, y sobre todo el antecedente familiar aparecido en edad precoz (antes de 55 años en familiares varones y de 65 en mujeres), que deben ser considerados para establecer un plan personalizado de control de riesgo en cada paciente.

2.3. IMPLEMENTANDO EL ALGORITMO DE FRAMINGHAM

Una vez estudiado qué es y cómo se estima el riesgo cardiovascular, el objetivo que nos proponemos en este trabajo es el desarrollo de una aplicación que implemente el algoritmo de Framingham, de forma que, para la información particular introducida por un sujeto, sea capaz de calcular el porcentaje de riesgo de sufrir una enfermedad cardiovascular en un período de 10 años.

La utilidad práctica del uso de la tabla, y consecuentemente de nuestra aplicación, radica en que permite:

- Priorizar los cuidados, controles y seguimientos en aquellas personas que presentan un mayor riesgo.
- Apoyar en las decisiones de tratamiento farmacológico en cuanto a hipolipemiantes, antihipertensivos y antiagregantes.
- Monitorizar la evolución del RCV
- Constituir una herramienta educativa y motivacional para el paciente en cuanto a la obtención de objetivos para la reducción de su riesgo.

Mujeres (edad)	Puntos	Varones (edad)	Puntos	cHDL (mg/dl)	Puntos	Colesterol (mg/dl)	Puntos	PAS	Puntos	Otros factores	Puntos
30	-12	30	-2	25-26	7	139-151	-3	98-104	-2	Tabaquismo	4
31	-11	31	-1	27-29	6	152-166	-2	105-112	-1	Diabetes:	
32	-9	32-33	0	30-32	5	167-182	-1	113-120	0	- varones	3
33	-8	34	1	33-35	4	183-199	0	121-129	1	- mujeres	6
34	-6	35-36	2	36-38	3	200-219	1	130-139	2	HVI	9
35	-5	37-38	3	39-42	2	220-239	2	140-149	3		
36	-4	39	4	43-46	1	240-262	3	150-160	4		
37	-3	40-41	5	47-50	0	263-288	4	161-172	5		
38	-2	42-43	6	51-55	-1	289-315	5	173-185	6		
39	-1	44-45	7	56-60	-2	316-330	6				
40	0	46-47	8	61-66	-3						
41	1	48-49	9	67-73	-4						
42-43	2	50-51	10	74-80	-5						
44	3	52-54	11	81-87	-6						
45-46	4	55-56	12	88-96	-7						
47-48	5	57-59	13								
49-50	6	60-61	14								
51-52	7	62-64	15								
53-55	8	65-67	16								
56-60	9	68-70	17								
61-67	10	71-73	18								
68-74	11	74	19								

Figura 2.2: Tablas de cálculos de RCV del estidoo de Framingham (Anderson, 1991) [11]

Puntos	Riesgos	Puntos	Riesgos	Puntos	Riesgos	Puntos	Riesgos
<1	<2	9	5	17	13	25	27
2	2	10	6	18	14	26	29
3	2	11	6	19	16	27	31
4	2	12	7	20	18	28	33
5	3	13	8	21	19	29	36
6	3	14	9	22	21	30	38
7	4	15	10	23	23	31	40
8	4	16	12	24	25	32	42

Figura 2.3: Tabla de puntuación y porcentaje de riesgo en los próximos 10 años [11]

CAPÍTULO 3

FASE II: DISEÑO

En el diseño de sistemas informáticos se suelen usar las arquitecturas multinivel o programación por capas, en la que a cada nivel se le confía una misión simple. Para el desarrollo de nuestra aplicación nos hemos marcado como objetivo el desacoplamiento de las diferentes partes que componen el sistema: capa de presentación, capa de datos y capa de lógica de negocios. En este sentido, comenzamos el proyecto diseñando la interfaz de usuario, continuamos modelando la parte lógica y terminamos añadiendo persistencia de datos.

3.1. PRESENTACIÓN. PROTOTIPO DE LA GUI

Una vez identificados los requisitos de la aplicación, nos hemos centrado en la creación de bocetos de baja fidelidad de la aplicación. En esta parte, nos hemos preocupado por el diseño de ventanas y posicionamiento de los controles, así como en el diseño de formularios y listados de información. Para ello, nos hemos basado en los conceptos teóricos vistos en la asignatura **Interacción Persona-Ordenador**, como empleo de metáforas, selección adecuada de colores y, sobre todo los aspectos relacionados con principios de usabilidad (flexibilidad, adaptabilidad, consistencia, etc.) y Leyes de Gestalt.

En este sentido, hemos utilizado la aplicación Balsamiq Mockups para realizar un prototipado horizontal que incluye la interfaz de todas las características del sistema, pero sin funcionalidad subyacente. En las capturas que se adjuntan a continuación, se incluyen los bocetos de baja fidelidad que creamos en una primera fase del proyecto, y que consecuentemente, pueden haber sufrido modificaciones con respecto al aspecto final de la aplicación.

Ventana de Inicio



Para que la aplicación permita monitorizar la evolución del RCV y pueda utilizarse como una herramienta motivacional para el paciente, hemos considerado obligatoria la necesidad de diseñar una ventana de *login*, como paso previo a la utilización de las funciones de la aplicación.

De esta forma, se podrá registrar cada uno de los cálculos realizados por el usuario y mostrar un gráfico con la evolución del RCV. Además, al disponer de cierta información personal sobre el usuario, no será necesario que introduzca algunos datos necesarios para realizar el cálculo.

Una vez autenticado, la navegación por la aplicación se basará en una barra de navegación situada en la parte inferior de la pantalla y

Figura 3.1: Ventana de Inicio

que ofrecerá la posibilidad de desplazarse por tres ventanas: estado, nuevo cálculo y perfil. Una de las razones que nos llevan a tomar esta decisión de diseño, es que la barra de navegación inferior se encuentra dentro de la ‘thumb zone’, que es el área que puede alcanzar el usuario cuando sujetá el móvil con una sola mano utilizando el pulgar para navegar. De esta forma, permite navegar rápidamente entre las principales vistas del menú de la app.

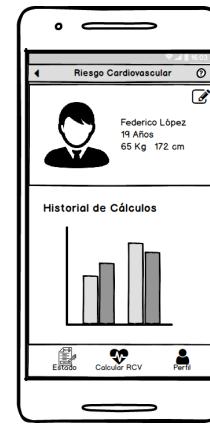
Ventana de Estado

La primera de las opciones seleccionables en el menú de navegación será la denominada Estado. En ella, se mostrará información relativa al último cálculo de RCV realizado por el usuario que ha iniciado sesión. Para ello, en la parte superior de la pantalla se mostrará un gráfico circular con el porcentaje de riesgo calculado y la fecha del mismo. Horizontalmente alineado con esta información aparecerá una etiqueta en la que se clasifique ese último cálculo en riesgo alto, moderado o bajo, de acuerdo con la tabla 2.3.

Por otro lado, se incluirá un listado dinámico con información sobre los factores de riesgo utilizados para realizar el cálculo. En este sentido, aparecerá información sobre la abstención o no de tabaco por parte del usuario, actividad física, tensión arterial, colesterol y si el usuario está en su peso ideal de acuerdo basándose en un cálculo del índice de masa corporal (IMC).



(a) Ventana de Estado



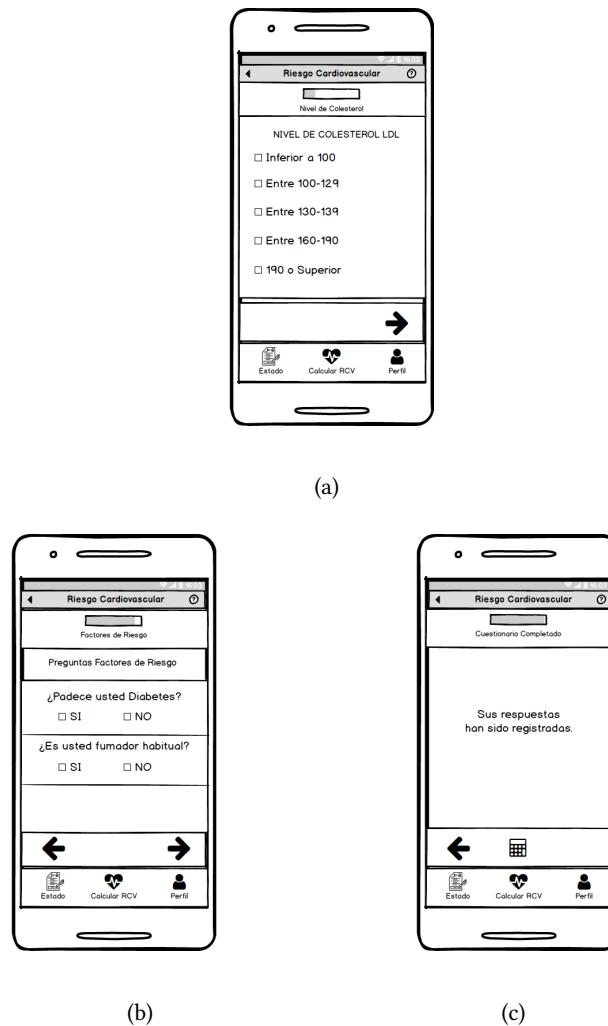
(b) Ventana de Perfil

Figura 3.2: Ventana de Perfil

Ventana de Cálculo del RCV

Seleccionando la opción Cálculo del menú de navegación de la aplicación el usuario podrá realizar una estimación del riesgo cardiovascular. Como sabemos, para poder aplicar el algoritmo de Framingham, es necesario haber obtenido previamente una serie de información concreta relevante al usuario. Estamos haciendo referencia a los factores de riesgos de los que hablamos en la sección de Análisis (2).

Esta ventana mostrará un formulario en el que se solicitará al usuario que introduzca su peso, altura, tensión arterial, si es o no fumador, colesterol (HDL y total), nivel de actividad física (un día por semana, dos días por semana, nada o una vez al mes) y enfermedades que puedan afectar al cálculo del RCV como la diabetes, hipertensión arterial o hipertrofia del ventrículo izquierdo.

**Figura 3.3:** Ventana Cálculo del RCV

3.2. DOMINIO

Dentro de la lógica de negocio, los usuarios se representan como objetos de la clase **Usuario** con una serie de propiedades para realizar la autenticación: correo electrónico y pass. Además, para permitir el control de la evolución del RCV, cada objeto usuario almacena un *array* con el historial de cálculos asociados a ese usuario. Un cálculo está definido mediante un objeto de la clase **CalculoRCV**, la cual contiene atributos y métodos necesarios para encapsular una estimación de riesgo cardiovascular. Esta es la razón de la unión mediante una relación de asociación entre la clase Usuario y CalculoRCV.

Al mismo tiempo, la clase CalculoRCV contiene como atributo un objeto de la clase **FactoresRCV**, la cual se utiliza para encapsular los factores de riesgo que se mostrarán en el listado dinámico que explicamos en la parte de presentación.

Por otro lado, hemos definido una *interface* **ConstantesFactores** para definir todas las propiedades constantes utilizadas para la estimación de RCV y recomendaciones de actuación. En este sentido, en esta interfaz definidos como constantes los valores límites para el colesterol, tensión y resultados del cálculo. Esta interfaz es implementada por las clases *FactoresRCV* y *CalculoRCV*.

Finalmente la clase **FraminghamRiskScore** contiene los métodos necesarios para calcular el porcentaje de riesgo cardiovascular según las tablas de Framingham. Adjuntamos el código Java en el anexo A.

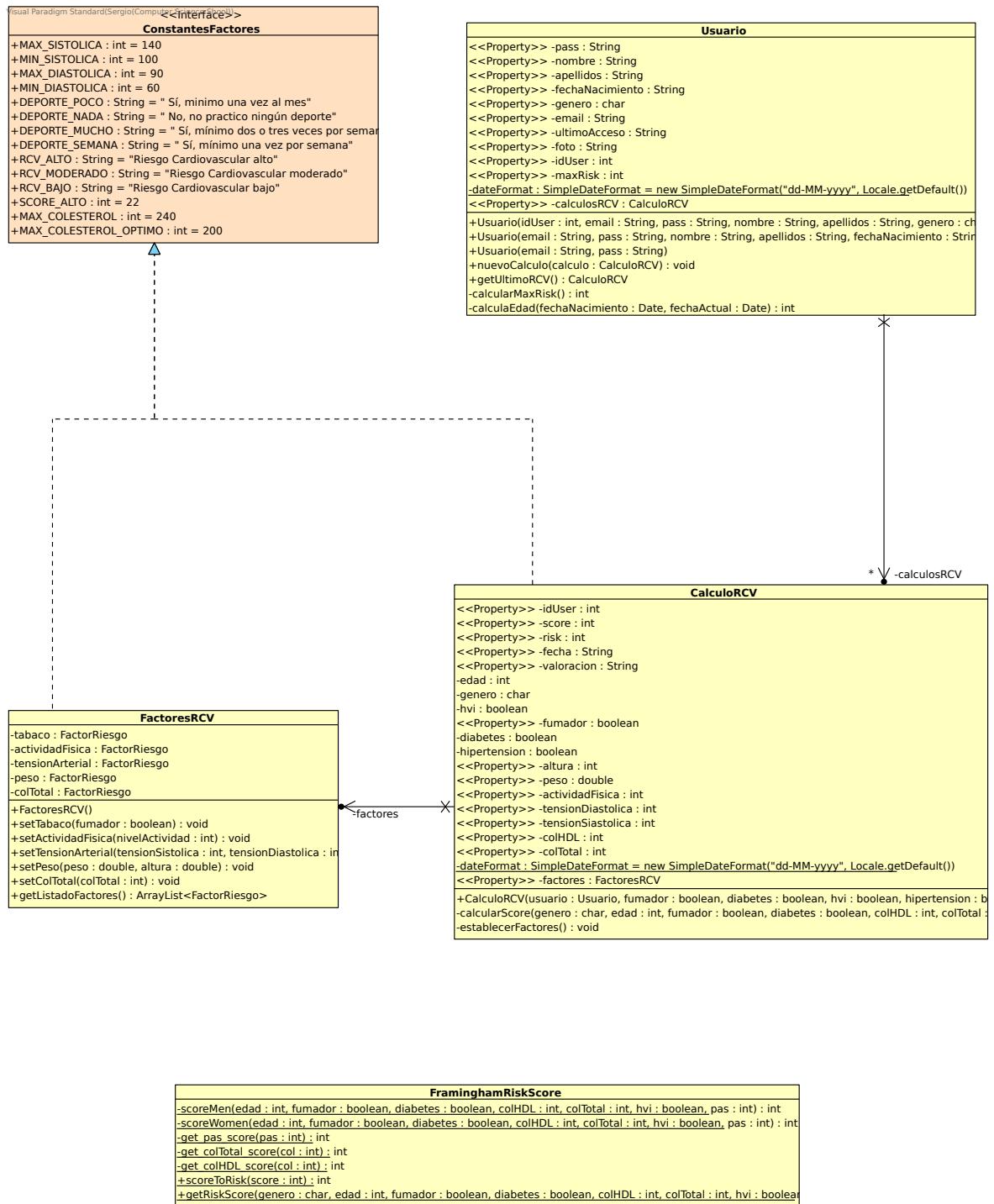


Figura 3.4: Diagrama de clases UML de la capa de dominio

3.3. PERSISTENCIA. BBDD LOCAL

Dentro de los distintos sistemas de bases de datos tanto privativos como libres/open source (Oracle, SQLServer, MySQL, etc) existe uno que se adapta perfectamente a las aplicaciones móviles: **SQLite**. El principal motivo es que SQLite no requiere más que un simple fichero para almacenar los datos, ya que la lógica de funcionamiento debe ser implementada por la plataforma que deseé interactuar con los datos.

Una vez decida la plataforma de BBDD que ibamos a utilizar en nuestra aplicación, era el momento de diseñar la estructura de la misma. Simplemente necesitamos modelar dos tablas: una para almacenar la información de los usuarios registrados en el sistema, indexada por un identificador único de usuario; y otra tabla en la que se almacenen todos los cálculos realizados en el sistema. Las dos tablas están relacionadas a través del identificador de usuario.

Las siguientes imágenes muestran el código SQL de creación de las dos tablas que forman la base de datos de nuestra aplicación.

```
CREATE TABLE `Usuario` (
    `idUser`      INTEGER PRIMARY KEY AUTOINCREMENT,
    `email`        TEXT UNIQUE,
    `pass`         TEXT NOT NULL,
    `nombre`       TEXT NOT NULL,
    `apellidos`   TEXT NOT NULL,
    `genero`       TEXT NOT NULL,
    `fechaNacimiento` TEXT NOT NULL,
    `ultimoAcceso` TEXT,
    `foto`          TEXT
);
```

(a) Tabla Usuario

```
CREATE TABLE `CalculosRCV` (
    `idCalculo`    INTEGER,
    `idUser`       INTEGER NOT NULL,
    `fumador`      INTEGER NOT NULL,
    `diabetes`     INTEGER NOT NULL,
    `hvi`          INTEGER NOT NULL,
    `hipertension` INTEGER NOT NULL,
    `peso`          REAL NOT NULL,
    `altura`       INTEGER NOT NULL,
    `actividadFisica` INTEGER NOT NULL,
    `tas`           INTEGER NOT NULL,
    `tad`           INTEGER NOT NULL,
    `colHDL`        INTEGER NOT NULL,
    `colTotal`      INTEGER NOT NULL,
    `fecha`         TEXT,
    PRIMARY KEY(`idCalculo`),
    FOREIGN KEY(`idUser`) REFERENCES Usuario
);
```

(b) Tabla CalculosRCV

Figura 3.5: Código creación de tablas SQLite

Implementación en Android

El procedimiento recomendado para crear una nueva base de datos SQLite en Android, se resume de forma esquemática:

- Crear una **clase** que extienda de **SQLiteOpenHelper**. En nuestro caso, la hemos llamado **DB_Helper**.

- Sobreescribir en ella el método ***onCreate()***, donde se ejecutará un comando SQLite para crear las tablas de la base de datos.
- También es necesario sobrescribir el método ***onUpgrade***, el cual se ejecutará cada vez que cambiamos la versión de la base de datos, y se usará para migrar los datos de la base de datos anterior a la nueva versión.

El siguiente listado de código muestra los métodos *onCreate* y *onUpgrade* de la clase DB_Helper, utilizada como conector en nuestra aplicación.

Listing 3.1: Clase DB_Helper.java

```
1  public DB_Helper(Context contexto , String nombreBD, ↪
2      ↪ SQLiteDatabase.CursorFactory factory , int versionBD) {
3      super(contexto , nombreBD , factory , versionBD);
4  }
5
6  @Override
7  public void onCreate(SQLiteDatabase db) {
8      try {
9          /* Se ejecuta la sentencia SQL de creación de la tabla */
10         db.execSQL(sqlCrearTablaUsuario);
11         db.execSQL(sqlCrearTablaCalculo);
12     } catch (SQLException e){
13         e.printStackTrace();
14     }
15 }
16
17 @Override
18 public void onUpgrade(SQLiteDatabase db, int oldVersion, int ↪
19      ↪ newVersion) {
20     try {
21         /* Se elimina la versión anterior de la table */
22         db.execSQL("DROP TABLE IF EXISTS Usuario");
23         db.execSQL("DROP TABLE IF EXISTS CalculosRCV");
24         /* Se crea la nueva versión de la table */
25         db.execSQL(sqlCrearTablaUsuario);
26         db.execSQL(sqlCrearTablaCalculo);
27     } catch (SQLException e){
28         e.printStackTrace();
29     }
30 }
31 }
```

CAPÍTULO 4

FASE III: IMPLEMENTACIÓN

4.1. INICIO DE SESIÓN Y REGISTRO DE UN NUEVO USUARIO

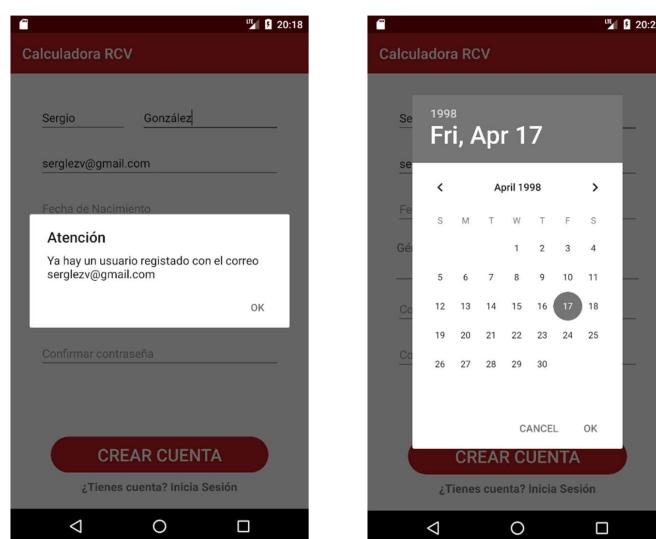
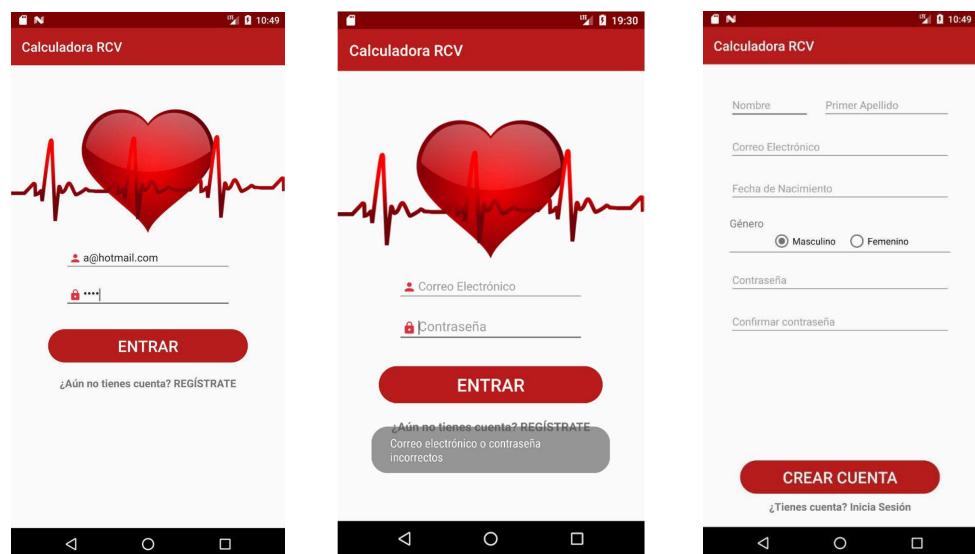


Figura 4.1: Implementación de la ventana para iniciar sesión y registrarse.

4.2. BARRA DE NAVEGACIÓN INFERIOR

4.2.1. Ventana de Estado



Figura 4.2: Implementación de la ventana que informa sobre la última estimación de RCV realizada

4.2.2. Ventana de nuevo Cálculo

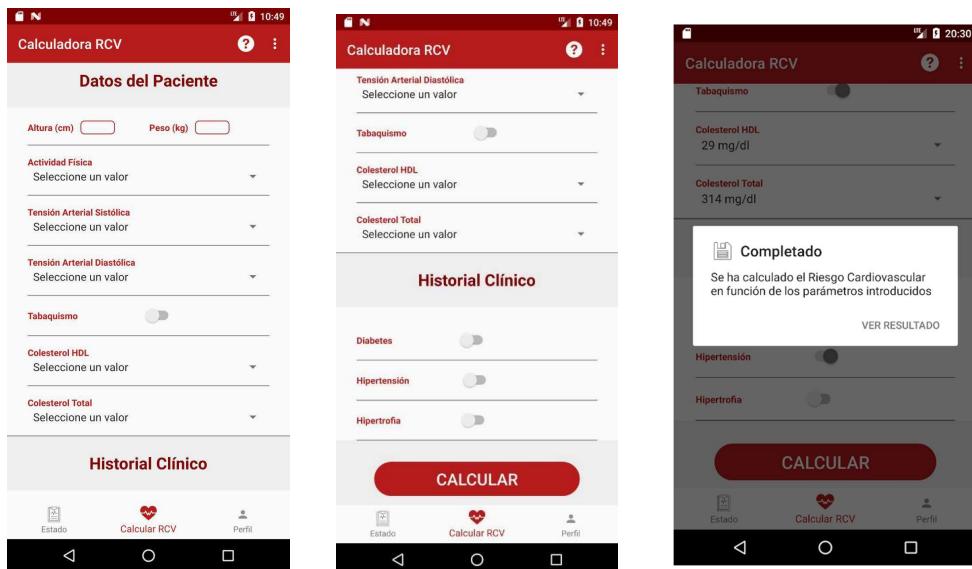


Figura 4.3: Implementación del formulario para realizar un nuevo cálculo de estimación RCV

4.2.3. Ventana de Perfil

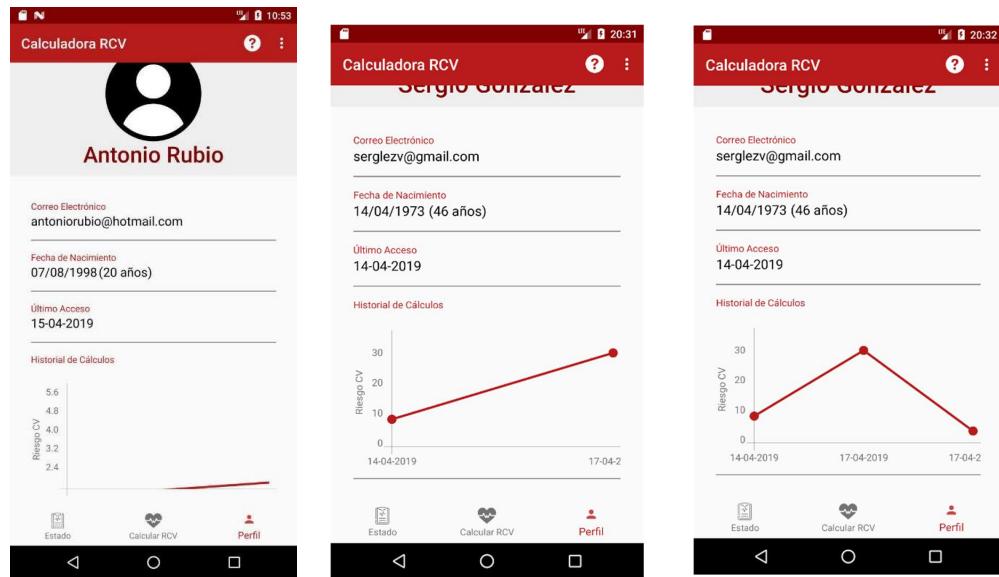


Figura 4.4: Ventana de perfil en la que se muestra un gráfico que permite monitorizar la evolución del RCV

4.3. ACTION BAR Y ACERCA DE...

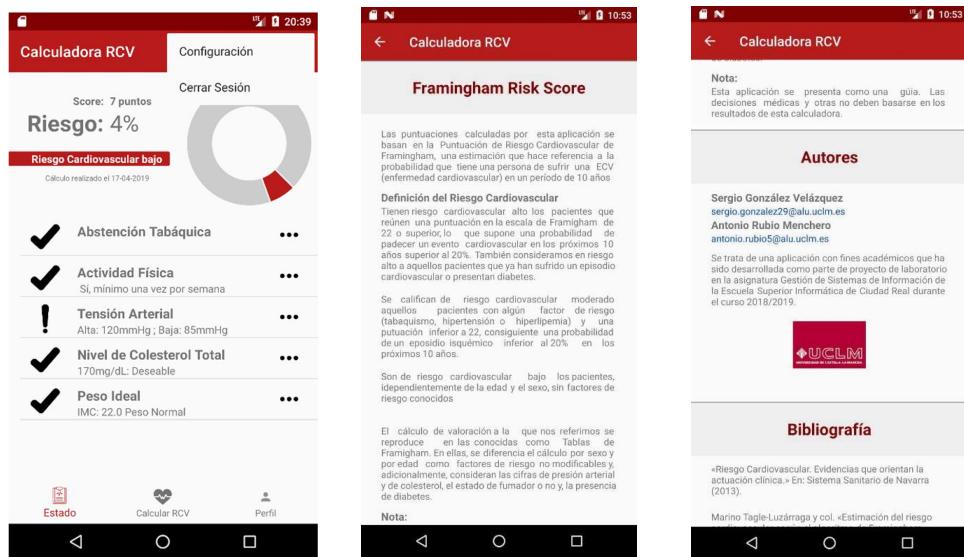


Figura 4.5: Ventana de información y ayuda sobre la aplicación

APÉNDICE A

TABLAS DE FRAMINGHAM EN JAVA

Consideramos interesante incluir en la documentación de nuestra aplicación la implementación del algoritmo de Framingham en Java. Para su desarrollo nos hemos basado en las tablas de estimación de riesgo cardiovascular explicadas en el capítulo 2. Entrando un poco en detalle en la estructura de la clase **FraminghamRiskScore**, cabe destacar la existencia de los métodos públicos **getRiskScore** y **scoreToRisk**. El primero de ellos es el encargado de estimar la valoración de Framingham a partir de los factores de riesgo que recibe como parámetros de entrada. Por su parte, la segunda función mencionada es la encargada de proporcionar el porcentaje de riesgo cardiovascular en función de la valoración de Framingham.

Listing A.1: Implementación del algoritmo de Framingham en Java

```
1 package ipo2.es.calculadorarcv.dominio;
2
3
4 import java.io.Serializable;
5
6 public final class FraminghamRiskScore implements Serializable {
7
8     private static int scoreMen(int edad, boolean fumador, boolean diabetes,
9                               int colHDL, int colTotal, boolean hvi, int pas) {
10        int points = 0;
11
12        /* -----
13         /          Edad hombres
14         ----- */
15        if(edad < 30) points = points - 2;
16        else if(edad == 31) points = points - 1;
17        else if(edad == 32 || edad == 33) points = points + 0;
18        else if(edad == 34) points = points + 1;
19        else if(edad == 35 || edad == 36 ) points = points + 2;
20        else if(edad == 37 || edad == 38 ) points = points + 3;
21        else if(edad == 39 ) points = points + 4;
22        else if(edad == 40 || edad == 41 ) points = points + 5;
23        else if(edad == 42 || edad == 43 ) points = points + 6;
24        else if(edad == 44 || edad == 45 ) points = points + 7;
25        else if(edad == 46 || edad == 47 ) points = points + 8;
26        else if(edad == 48 || edad == 49 ) points = points + 9;
27        else if(edad == 50 || edad == 51 ) points = points + 10;
28        else if(edad >= 52 && edad <= 54 ) points = points + 11;
29        else if(edad == 55 || edad == 56 ) points = points + 12;
30        else if(edad >= 57 && edad <= 59 ) points = points + 13;
31        else if(edad == 60 || edad == 61 ) points = points + 14;
32        else if(edad >= 62 && edad <= 64 ) points = points + 15;
33        else if(edad >= 65 && edad <= 67 ) points = points + 16;
34        else if(edad >= 68 && edad <= 70 ) points = points + 17;
35        else if(edad >= 71 && edad <= 73 ) points = points + 18;
36        else if(edad >= 74) points = points +19;
37
38
39        /* -----
40         /          PRESIÓN ARTERIAL SISTÓLICA
41         ----- */
42        points += get_pas_score(pas);
```

```

43
44     /* -----
45     /      COLESTEROL Total (malo)(mg/ dl)
46     /----- */
47     points += get_colTotal_score(colTotal);
48
49     /* -----
50     /      COLESTEROL HDL (bueno)(mg/ dl)
51     /----- */
52     if (colHDL == 0) points+=1;
53     else points += get_colHDL_score(colHDL);
54
55     /* -----
56     /          OTROS FACTORES
57     /----- */
58     if(fumador) points+=4;
59
60     if(diabetes) points += 3;
61
62     if(hvi) points += 9;
63
64
65
66     return points;
67 }
68
69 private static int scoreWomen(int edad, boolean fumador, boolean diabetes,
70                               int colHDL, int colTotal, boolean hvi, ↵
71                               ↵ int pas){
72
73     int points = 0;
74
75     /* -----
76     /          Edad mujeres
77     /----- */
78     if(edad <= 41){
79         if(edad < 30) points = points - 12;
80         else if(edad == 31) points = points - 11;
81         else if(edad == 32) points = points - 9;
82         else if(edad == 33) points = points - 8;
83         else if(edad == 34) points = points - 6;
84         else if(edad == 35) points = points - 5;
85         else if(edad == 36) points = points - 4;
86         else if(edad == 37) points = points - 3;
87         else if(edad == 38) points = points - 2;
88         else if(edad == 39) points = points - 1;
89         else if(edad == 40) points = points - 0;
90         else if(edad == 41) points = points + 1;
91     } else{
92         if(edad == 42 || edad == 43 ) points = points + 2;
93         else if(edad == 44) points = points + 3;
94         else if(edad == 45 || edad == 46 ) points = points + 4;
95         else if(edad == 47 || edad == 48 ) points = points + 5;
96         else if(edad == 49 || edad == 50 ) points = points + 6;
97         else if(edad == 51 || edad == 52 ) points = points + 7;
98         else if(edad <= 55) points = points + 8;
99         else if(edad <= 60) points = points + 9;
100        else if(edad <= 67) points = points + 10;
101        else points = points + 11;
102    }
103
104    /* -----
105    /          PRESIÓN ARTERIAL SISTÓLICA
106    /----- */
107    points += get_pas_score(pas);
108
109    /* -----
110    /      COLESTEROL LDL (malo)(mg/ dl)
111    /----- */
112    points += get_colTotal_score(colTotal);
113
114    /* ----- */

```

```

115     /      COLESTEROL HDL (bueno) (mg/ dl)
116     /----- */
117     if (colHDL == 0) points +=1;
118     else points += get_colHDL_score(colHDL);
119
120     /* -----
121     /      OTROS FACTORES
122     /----- */
123     if (fumador) points +=4;
124
125     if (diabetes) points += 6;
126
127     if (hvi) points += 9;
128
129
130     return points;
131
132 }
133
134
135     private static int get_pas_score(int pas){
136         //Presión arterial sistólica
137         int score = 0;
138         if (pas <=104) score +=-2;
139         else if (pas <=112) score +=-1;
140         else if (pas <=120) score +=0;
141         else if (pas <=129) score +=1;
142         else if (pas <=139) score +=2;
143         else if (pas <=149) score +=3;
144         else if (pas <=160) score +=4;
145         else if (pas <=172) score +=5;
146         else score += 6;
147         return score;
148     }
149
150
151     private static int get_colTotal_score(int col){
152         int score = 0;
153         if (col <=151) score +=-3;
154         else if (col <=166) score +=-2;
155         else if (col <=182) score +=-1;
156         else if (col <=199) score +=0;
157         else if (col <=219) score +=1;
158         else if (col <=239) score +=2;
159         else if (col <=262) score +=3;
160         else if (col <=288) score +=4;
161         else if (col <=315) score +=5;
162         else score += 6;
163
164         return score;
165     }
166
167     private static int get_colHDL_score(int col){
168         int score = 0;
169         if (col <=26) score +=7;
170         else if (col <=29) score +=6;
171         else if (col <=32) score +=5;
172         else if (col <=35) score +=4;
173         else if (col <=38) score +=3;
174         else if (col <=42) score +=2;
175         else if (col <=46) score +=1;
176         else if (col <=50) score +=0;
177         else if (col <=55) score -=1;
178         else if (col <=60) score -=2;
179         else if (col <=66) score -=3;
180         else if (col <=73) score -=4;
181         else if (col <=80) score -=5;
182         else if (col <=87) score -=6;
183         else score -=7 ;
184         return score;
185     }
186
187     public static int scoreToRisk(int score){
188         int risk = 0;

```

```

188
189     if(score <=1) risk = 1;
190     else if(score <= 4) risk = 2;
191     else if(score <= 6) risk = 3;
192     else if(score <= 8) risk = 4;
193     else{
194         if(score < 17){
195             if(score ==9) risk = 5;
196             else if(score <= 11) risk = 6;
197             else if(score ==12) risk = 7;
198             else if(score ==13) risk = 8;
199             else if(score ==14) risk = 9;
200             else if(score ==15) risk = 10;
201             else risk = 12;
202         }
203         else if (score <=24){
204             if(score ==17) risk = 13;
205             else if(score == 18) risk = 14;
206             else if(score ==19) risk = 16;
207             else if(score ==20) risk = 18;
208             else if(score ==21) risk = 19;
209             else if(score ==22) risk = 21;
210             else if(score ==23) risk = 23;
211             else risk = 25;
212         }
213     }
214     if(score ==25) risk = 27;
215     else if(score == 26) risk = 29;
216     else if(score ==27) risk = 31;
217     else if(score ==28) risk = 33;
218     else if(score ==29) risk = 36;
219     else if(score ==30) risk = 38;
220     else if(score ==31) risk = 40;
221     else risk = 42;
222     }
223 }
224     return risk;
225 }
226
227 public static int getRiskScore(char genero, int edad, boolean fumador, ↵
228     ↵ boolean diabetes, ↵
229     ↵ int colHDL, int colTotal, boolean hvi, ↵
230     ↵ ↵ int pas){
231     int score= 0;
232     if(genero == 'm'){ //mujer
233         score = scoreWomen(edad, fumador, diabetes, colHDL,colTotal, ↵
234             ↵ ↵ hvi, pas);
235     } else{ //hombre
236         score = scoreMen(edad, fumador, diabetes, colHDL,colTotal, hvi, ↵
237             ↵ ↵ pas);
238     }
239     return score;
240 }
```

BIBLIOGRAFÍA

FUENTES NO ONLINE

- [3] María Grau y Jaume Marrugat. «Funciones de riesgo en la prevención primaria de las enfermedades cardiovasculares». En: *Revista española de cardiología* 61.4 (2008), págs. 404-416.
- [7] «Riesgo Cardiovascular. Evidencias que orientan la actuación clínica.» En: *Sistema Sanitario de Navarra* (2013).
- [10] José Enrique Amaro Soriano. *Android: Programación de dispositivos móviles a través de ejemplos*. Marcombo, 2011.
- [11] Marino Tagle-Luzárraga y col. «Estimación del riesgo cardiovascular según el algoritmo de Framingham en sujetos con síndrome metabólico, definido por los criterios del NCEP-ATP-III». En: *Endocrinología y Nutrición* 54.4 (2007), págs. 211-215.

FUENTES ONLINE

- [1] codedmin. *Android Line Chart – How to Draw Line Chart in Android*. URL: <https://www.codingdemos.com/draw-android-line-chart/>.
- [2] codedmin. *Android Pie Chart – How to Create Pie Chart in Android Studio*. URL: <https://www.codingdemos.com/android-pie-chart-tutorial/>.
- [4] Valentin Hinov. *Scroll your Bottom Navigation View away with 10 lines of code*. 2018. URL: <https://android.jlelse.eu/scroll-your-bottom-navigation-view-away-with-10-lines-of-code-346f1ed40e9e>.
- [5] Santi Martínez. *Buenas prácticas usando Fragments en Android*. 2014. URL: <http://gpmess.com/blog/2014/04/16/buenas-practicas-usando-fragments-en-android/>.
- [6] James Revelo. *Tutorial De Bases De Datos SQLite En Aplicaciones Android*. URL: <http://www.hermosaprogramacion.com/2014/10/android-sqlite-bases-de-datos/>.
- [8] Shaon. *Android Line Chart Example*. URL: <http://shaoniuc.com/android/android-line-chart-example/>.
- [9] Nipun De Silva. *HelloCharts for Android Example*. Sunday, June 12, 2016. URL: <http://nipuns writings.blogspot.com/2016/06/hellocharts-for-android-example.html>.