UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

DISEÑO E IMPLEMENTACIÓN DE UNA HERRAMIENTA EDUCATIVA SOBRE LA GESTIÓN DE DIABETES TIPO 1 EN NIÑOS EMPLEANDO UN DISPOSITIVO ROBÓTICO DE BAJO COSTE

AUTOR: CARLOS VISERAS RUIZ

TUTOR: JOSÉ LUÍS DIEZ RUANO

COTUTOR: JORGE BONDIA COMPANY

Curso Académico: 2017-18

RESUMEN

En el presente proyecto se ha desarrollado una herramienta educativa para ayudar a los niños con diabetes mellitus tipo 1 a gestionarla y tratarla. Para ello se ha desarrollado una aplicación móvil en Android que interactúa con un dispositivo robótico basado en Arduino. El proyecto consta de diferentes juegos lúdico-educativos, además de incorporar un sistema de monitorización a tiempo real. Con esta herramienta se ha tratado de transmitir de una manera más empática, adaptando toda la interfaz al público al que va dirigido, los conocimientos que un niño diabético debe conocer para gestionar su enfermedad.

Palabras Clave: diabetes mellitus, niño, Android, Arduino, robot.

RESUM

En aquest projecte s'ha desenvolupat una eina educativa per ajudar als xiquets que pateixen diabetes mellitus tipus 1 a gestionar-la i tractar-la. Per tal cosa s'ha desenvolupat una aplicació mòvil a Android que interactúa amb un dispositiu robótic basat en Arduino. El projecte consta de diferents jocs lúdic-educatius, a més d'incorporar un sistema de monitorització a temps reial. Amb aquesta eina s'ha tractat transmetre d'una forma més enpatica, adaptant tota l'interfaç al públic al qual va dirigit, les coneixements que un xiquet diabètic a de coneixer per gestionar la seua malaltia.

Paraules clau: diabetes mellitus, xiquet, Android, Arduino, robot.

ABSTRACT

In the current project an educational tool has been developed to help children with type 1 diabetes mellitus to manage and treat it. To do this, a mobile application has been developed on Android, that interacts with a robotic device based on Arduino. The project consists of different recreational-educational games, in addition it incorporates a monitoring system in real time. With this tool it has been treated to transmit, in a more empathetic way and adapting the whole interface to the target audience, the knowledge that a diabetic child must know to manage their disease.

Keywords: diabetes mellitus, child, Android, Arduino, robot.

ÍNDICE

Capítulo 1. Introducción

1.1. Objetivo del Proyecto 1

1.2. motivación 1

1.3. Interés académico 2

1.4. Estructura del documento y del proyecto 2

Capítulo 2. La diabetes

2.1. Definición 5

2.2. Epidemiología 5

2.3. Mecanismo de producción 5

2.4. Tipos 6

2.5. SÍNTOMAS 6

2.6. Consecuencias 7

2.7. Prevención 7

2.8. tratamiento 8

2.9 La alimentación en la diabetes 11

2.10. La actividad física 12

2.11. Educación diabetológica 13

2.12. La robótica y la educación 14

Capítulo 3. El problema y sus posibles soluciones

3.1. Planteamiento del problema 15

3.2. Alternativas de solución 15

3.2.1. Sistema operativo móvil de desarrollo 16

3.2.2. Dispositivo robótico 16

3.3. Justificación de la elección 17

3.3.1. Android. El sistema operativo elegido 17

3.3.2. MBot. El dispositivo robótico elegido 17

Capítulo 4. GlucoEduca, la aplicación móvil

4.1. Definición 19
4.2. Android Studio, entorno de programación en JAVA 19
4.3. Firebase 20
4.4. GIMP, manipulador de imágenes 23
4.5. Audacity, editor de audio 23
4.6. Representación de gráficos. Librerías utilizadas 24
4.7. Animaciones 24
4.8. Control de tiempos y fechas 25
4.9. Variables Globales 26
4.10. La conexión Bluetooth, y su efecto en la transición entre actividades 26
4.11. Estructura de la aplicación y sus diferentes pantallas 28
4.11.1. Pantalla de inicio de sesión 28
4.11.2. Pantalla de selección de modo 29
4.11.3. Modo de conducción libre 33
4.11.4. Modo de simulación de estados de glucosa 34
4.11.4.1. Actividad lotería de raciones de carbohidratos 34
4.11.4.2. Actividad calculadora de insulina 35
4.11.4.3. Actividad de simulación interactiva 37
4.11.4.4. Simulador de glucosa 38
4.11.5. Modo preguntas 40
4.11.6. Modo recompensas 41
4.11.7. Modo de simulación continua 42
4.11.8. Modo juego de memoria 43
4.11.9. Modo Estadísticas 44
4.11.10. Modo resetear 47
4.11.11. Modo historia interactiva 48
4.11.12. Modo estadísticas con acceso remoto 51
4.12. Manual de Usuario 52

Capítulo 5. MBot el dispositivo robótico

5.1. Descripción 53

5.2. EL movimiento del robot 54
5.3. La programación de Mbot 55
5.3.1. MBlock, programación gráfica 55
5.3.2. Arduino IDE, programación desde código 56
5.4. Conexión entre la aplicación móvil y el robot 56
5.5. La estructura del código 57
5.6. Integración del robot en la aplicación 59
5.6.1. Modo principal de selección de modo 59
5.6.2. Modo de conducción libre 59
5.6.3. Modo de simulación interactiva 59
5.6.4. Modo de preguntas 60
5.6.5. Modo de recompensas 60
5.6.6. Modo de simulación continua 60
5.6.7. Modo de juego de memoria 60
5.6.8. Modo de estadísticas 60
5.6.9. Modo de historia 61
5.6.10. Modo de desconexión 61
5.6.11. Modo de alarma 61

Capítulo 6. Resultados y conclusiones 62

Capítulo 7. Trabajo futuro y mejoras 63

Capítulo 8. Bibliografía

8.1. Bibliografía de información 67
8.2. Bibliografía de recursos 69

Presupuesto

1. Mano de obra 2
2. Materiales 2
3. Precios unitarios 3
4. Precios descompuestos 3
5. Presupuesto de ejecución por contrata 6

Anexo I. Código simulador de glucosa.....2

Anexo II. Manual de usuario.....6

Anexo III. Manual del programador

1. Introducción 8

2. Estructura manual 8

3. La aplicación móvil 8

3.1. Firebase 8

3.2. Monitorización continua 9

3.3. Conexión Bluetooth 9

3.4. Personalización de la base de datos a cada usuario 9

3.5. Progreso y recompensas 9

3.6. El simulador 9

4. EL robot 10

4.1. Conexión con la aplicación 10

4.2. Movimiento del robot 10

4.3. Programación del robot 10

4.4. Estructura del código 10

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

DISEÑO E IMPLEMENTACIÓN DE UNA HERRAMIENTA EDUCATIVA SOBRE LA GESTIÓN DE DIABETES TIPO 1 EN NIÑOS EMPLEANDO UN DISPOSITIVO ROBÓTICO DE BAJO COSTE

MEMORIA

Curso Académico: 2017-18

CAPÍTULO 1. INTRODUCCIÓN

1.1. OBJETIVO DEL PROYECTO

El objetivo de este proyecto es el de crear una plataforma educativa para el tratamiento y gestión de la diabetes 1 en niños. Para ello se desarrollará un software educativo junto a la programación de un dispositivo robótico que potencie la interacción y empatía del niño hacia la plataforma. Este objetivo integra varios subobjetivos cuyo cumplimiento se presenta crucial para el desarrollo de una plataforma educativa útil.

- Crear un software que transmita de forma adecuada los conocimientos que un niño con diabetes tipo 1 deba adquirir.
- Diseñar una interfaz atractiva para el niño con el fin de captar su interés.
- Integrar un simulador de la evolución de la glucosa en sangre, basado en el modelo de un paciente, en el software.
- Desarrollar un sistema de monitorización continua de la glucosa en sangre del niño.
- Tratamiento y visualización de las estadísticas extraídas de la monitorización continua.
- Crear juegos puramente lúdicos que mantengan el interés del niño en utilizar la plataforma.
- Diseñar un sistema de recompensas que motiven al usuario a continuar su progreso en la plataforma.
- Establecimiento de un sistema de conexión e integración entre el software y el dispositivo robótico.
- Integrar todos los datos del usuario en una base de datos en la nube para independizar el dispositivo que ejecute el software de la plataforma en sí.

1.2. MOTIVACIÓN

El proyecto se motiva en la creación de una herramienta educativa que ayude a los niños pacientes de diabetes tipo 1 ha aprender y poder llevar mejor su enfermedad. Con esta motivación se intenta crear una plataforma lúdico-educativa que atraiga al usuario y lo anime a seguir utilizándola para de forma indirecta adquirir los conocimientos necesarios para la correcta gestión de su tratamiento.

Con ello se intenta mejorar la vida del niño proporcionándole un sistema de monitorización continua evadiéndole del proceso de toma de muestras. También se busca la seguridad del niño

al controlar y alertar en el momento que su estado glucémico pueda resultar peligroso. Así como, facilitarle el arduo proceso de aprendizaje de la gestión de comidas e insulina gracias al simulador integrado en la herramienta.

Este proyecto se enmarca en un proyecto de educación diabetológica iniciado por el Instituto de Automática e Informática Industrial de la UPV.

1.3. INTERÉS ACADÉMICO

Este proyecto se aborda desde el ámbito de un Trabajo Final de Grado, requisito necesario para completar los estudios de Grado en Ingeniería en Tecnologías Industriales cursados en la Universidad Politécnica de Valencia.

Al ser un proyecto que aborda tanto la implementación de un software educativo como la integración de un dispositivo robótico en él, deriva en varios motivos que justifican su interés académico:

- Uso y aprendizaje de varios lenguajes de programación introducidos durante el grado.
- Programación de un dispositivo robótico, donde no solo se aplica el conocimiento de un determinado lenguaje de programación, sino que también se aplican los conceptos de robótica y cinemática adquiridos en los estudios.
- Desarrollo y entendimiento de la importancia de establecer un protocolo de comunicación entre dos dispositivos tan dispares con el fin de dotar de robustez.
- Instrumentación y entendimiento de los diferentes componentes que integra el robot, así como su utilización y tratamiento. Aspectos para los cuales los conocimientos aprendidos en las asignaturas de electrónica han resultado útiles.
- Uso y combinación de software muy diverso pero que combinados complementan el desarrollo de la plataforma y por ende su resultado final.
- Entendimiento e integración de un modelo de paciente donde se simula la evolución de la glucosa en sangre. Para ello se han aplicado las nociones adquiridas durante el grado en áreas como los métodos numéricos o la modelización.

1.4. ESTRUCTURA DEL DOCUMENTO Y DEL PROYECTO

Este documento se ha estructurado siguiendo el mismo recorrido seguido para el desarrollo del proyecto.

Primero se inició un proceso de recogida de información y entendimiento sobre la diabetes en general, específicamente en la de tipo 1 en niños, para luego centrar la búsqueda en los precedentes existentes en cuanto a educación diabetológica se trata.

En segundo lugar, se inició el desarrollo del software que conforma la plataforma educativa creando diferentes modos que proporcionen un producto útil para la educación en el tratamiento y gestión de la diabetes en niños.

Luego se abordó la integración como interfaz de un dispositivo robótico en la plataforma con el fin de potenciar la empatía e interacción entre el producto y el niño.

Por último, se centraron los esfuerzos en el diseño visual y composición de la interfaz que ofrece tanto el software como el dispositivo robótico haciéndolo atractivo para un niño.

Después de completar todos estos pasos, se crea un producto bastante completo en cuanto a educación diabetológica se refiere que puede dar el siguiente paso para empezar a realizar pruebas con el colectivo al que va destinado.

CAPÍTULO 2. LA DIABETES

2.1. DEFINICIÓN

La diabetes es una enfermedad crónica que se produce porque el páncreas no regula adecuadamente los niveles de glucosa en sangre, esto puede ser causado por tres motivos, el primero es porque el páncreas no produce insulina de forma adecuada, el segundo porque la insulina se produce correctamente, pero el organismo no la puede utilizar de forma eficaz haciéndose resistente a la misma, el tercero y último se trata de una mezcla de los dos anteriores. Como resultado de lo anterior se produce hiperglucemia, es decir, un aumento de los niveles de glucosa en sangre. En la actualidad esta enfermedad no tiene cura alguna, únicamente se pueden prevenir las posibles complicaciones. [(1)]

2.2. EPIDEMIOLOGÍA

En el primer Informe mundial publicado por la OMS sobre diabetes ya se estimaba que en el año 2014 422 millones de adultos en todo el mundo sufrirían esta enfermedad, frente a los 108 millones de 1980. [(2)]

En el año 2014 más de 5 millones de personas en España sufrían esta enfermedad crónica, formando el colectivo más grande de pacientes crónicos del país. En ese momento más de 5.301.314 personas sufrían diabetes tipo 2, y la Federación Española de Diabetes estimaba que más de 2 millones en ese momento estaban aún sin diagnosticar. Por otro lado, la diabetes tipo 1 representaba un 13% del total de los diabéticos, de los cuales 29.000 eran niños menores de 15 años, con una incidencia anual de 1.100 casos. [(3)]

Por otro lado, el coste anual de este problema en nuestro país es de 23.077 millones de euros, de los cuales 17.630 son ocasionados por costes indirectos como el absentismo laboral, las jubilaciones anticipadas y los gastos sociales, y 5.447 son costes directos ocasionados por tratamientos y hospitalizaciones. [(3)]

En un ranking de 30 países europeos España se sitúa en el puesto número 18. En el mundo 382 millones de personas tienen esta enfermedad, lo que supone un 8,3% de la población adulta, y se prevé que esta cifra aumente a 592 millones en el año 2035, lo cual supone un aumento del 55% en dos décadas. [(3)]

2.3. MECANISMO DE PRODUCCIÓN

En el momento en el que ingerimos un alimento este se descompone en los elementos que lo forman, muchos de los cuales son moléculas de glucosa, comúnmente conocidos como hidratos de carbono, estas moléculas son absorbidas en el intestino delgado, a través del cual

entran al torrente sanguíneo, después de lo cual el páncreas, un órgano tanto endocrino como exocrino, produce insulina, que se encargará de que esas moléculas de glucosa anteriormente nombradas sean utilizadas por el organismo para la producción de energía. [1]

2.4. TIPOS

Existen principalmente dos tipos de diabetes, detallados en la siguiente tabla (ver Tabla 1). [1]

Tabla 1. Tipos de diabetes.

Diabetes tipo 1	También conocida como diabetes insulino dependiente, este tipo de diabetes se debe a que el páncreas no produce insulina de forma total o parcial, debido a lo cual las personas que la sufren han de inyectarse insulina de forma diaria. Este tipo de diabetes se da con mayor frecuencia en niños. La causa que produce esta enfermedad es todavía desconocida.
Diabetes tipo 2	Este tipo de diabetes se debe a que, aunque el páncreas produce de forma eficaz la insulina, esta no es utilizada de forma adecuada por el organismo, este tipo de diabetes es más frecuente en adultos, la mayoría de los cuales desconocen que la padecen.

Fuente: elaboración propia a partir de la bibliografía consultada. [1]

2.5. SÍNTOMAS

Los síntomas más frecuentes causados por la diabetes descompensada son los siguientes (ver Figura 1). [1]

Figura 1. Síntomas de la diabetes.

Fuente: Elaboración propia a partir de la bibliografía consultada. [1]

Mientras que la diabetes tipo 1 debuta de forma temprana, la diabetes tipo 2 o hace de forma insidiosa, por lo que muchas de las personas que la sufren pueden no tener síntomas en un principio. [1]

2.6. CONSECUENCIAS

Si esta enfermedad de no se detecta y se trata de forma precoz puede acarrear otros problemas más graves a largo plazo, como, por ejemplo: retinopatía diabética, úlceras e infecciones, neuropatía, insuficiencia renal, arteriopatías, etc. [1]

2.7. PREVENCIÓN

Actualmente la diabetes tipo 1 no tiene ningún tipo de prevención, pero si existen investigaciones que están tratando de averiguar cómo retrasar la aparición de esta enfermedad en personas de alto riesgo. [1]

Por otra parte, la prevención de la diabetes tipo 2 consiste en mantener un estilo de vida saludable, manteniendo un peso corporal adecuado junto a actividad física regular. [1]

2.8. TRATAMIENTO

El tratamiento de la diabetes tipo 1 consiste fundamentalmente en seguir una dieta equilibrada, ejercicio regular y el tratamiento con insulina. Mientras que el tratamiento de la diabetes tipo 2 consiste en dieta y ejercicio físico, dependiendo de la gravedad de la enfermedad se puede necesitar o no tratamiento farmacológico y/o insulina. Todo ello junto a una adecuada educación en salud y seguimiento profesional en cualquiera de ambos casos. [(1)]

Hay que tener en cuenta que poco después de haber hecho el diagnóstico de diabetes tipo 1 en niños se produce un periodo conocido como "luna de miel", el cual consiste en una fase en la que el páncreas vuelve a secretar insulina, de forma que en muchos casos hay que disminuir la dosis de insulina o incluso anularla, sin embargo, hay que tener en cuenta que esta fase dura poco tiempo y es necesario prevenir a los niños y a la familia de que esta fase de remisión es transitoria y por tanto una vez haya pasado se volverán a requerir mayores cantidades de insulina. [(4)]

Existen varios tipos de insulina que se pueden combinar para el tratamiento de la diabetes, estos tipos se diferencian en el momento en el que empieza el efecto, el momento en el que se produce el pico de máximo efecto y la duración de la insulina en el organismo. Actualmente en el mercado existen insulinas ya mezcladas, sin embargo, estas no son adecuadas para la edad pediátrica puesto que las necesidades de los niños son muy variables. Además, hay que tener en cuenta que existen factores que aceleran o disminuyen la velocidad de la insulina una vez administrada (ver Tabla 2). [(5)]

Tabla 2. Factores que modifican la velocidad del efecto de la insulina.

Factores que aceleran la acción	- Actividad física en la zona de administración. - Calor en la zona de punción. - Masaje sobre la zona de administración. - Administrar la inyección a mayor profundidad de la adecuada.
Factores que retrasan la acción	- El humo del tabaco. - Frío sobre la zona de punción. - Administrar la inyección a menor profundidad de la adecuada.

Fuente: elaboración propia a partir de la bibliografía consultada. [(5)]

Para el correcto uso de la insulina hay que tener en cuenta varios factores: mirar la fecha de caducidad antes de su administración, si la presentación normal se vuelve turbia no debe

inyectarse, es importante tener insulina de reserva y conservarla en la nevera, el vial que este en uso en ese momento puede conservarse a temperatura ambiente, pero sin superar los 24ºC, además una vez abierto es importante anotar la fecha de la apertura puesto que este no debe usarse pasadas tres semanas. Si por el contrario se guarda el frasco en uso en la nevera hay que calentarlo con las manos antes de administrar la insulina, ya que el niño sentirá más dolor en su administración y se absorberá peor una vez puesta. Por otra parte, si se van a hacer viajes largos es importante que el fármaco se mantenga a una temperatura más o menos constante, evitando exponerla a temperaturas muy frías o muy cálidas. [5]

A partir de los 8 años es importante que los niños con diabetes comiencen a aprender a administrarse la insulina de forma autónoma. Para la correcta administración los niños deben de seguir una serie de pasos (ver Figura 2). [5]

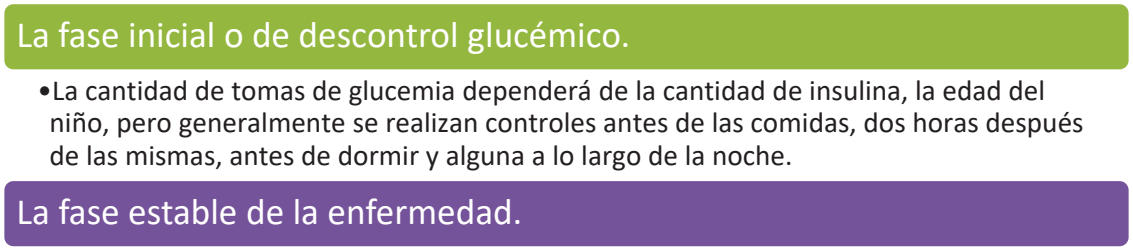
Figura 2. Pasos para a seguir para la administración de la insulina.

Fuente: Elaboración propia a partir de la bibliografía consultada. [5]

Es importante que si se observan anomalías en la zona de punción se consulte con el equipo sanitario, y se vayan alternando los puntos de inyección para que estas no lleguen a producirse. Existen varios lugares en los que se puede puncionar: en el muslo tanto en la parte anterior como lateral externa, en el cuadrante externo superior de las nalgas, en el abdomen dejando una zona libre de aproximadamente dos dedos alrededor del ombligo, y en la parte superior externa de ambos brazos en caso de que los niños ya no sean muy pequeños. Es importante seleccionar una de las zonas y puncionar siempre en la misma, puesto que la velocidad de absorción varía entre ellas. [5]

En la realización de los controles de glucemia tenemos que tener en cuenta dos fases (ver Figura 3). [5]

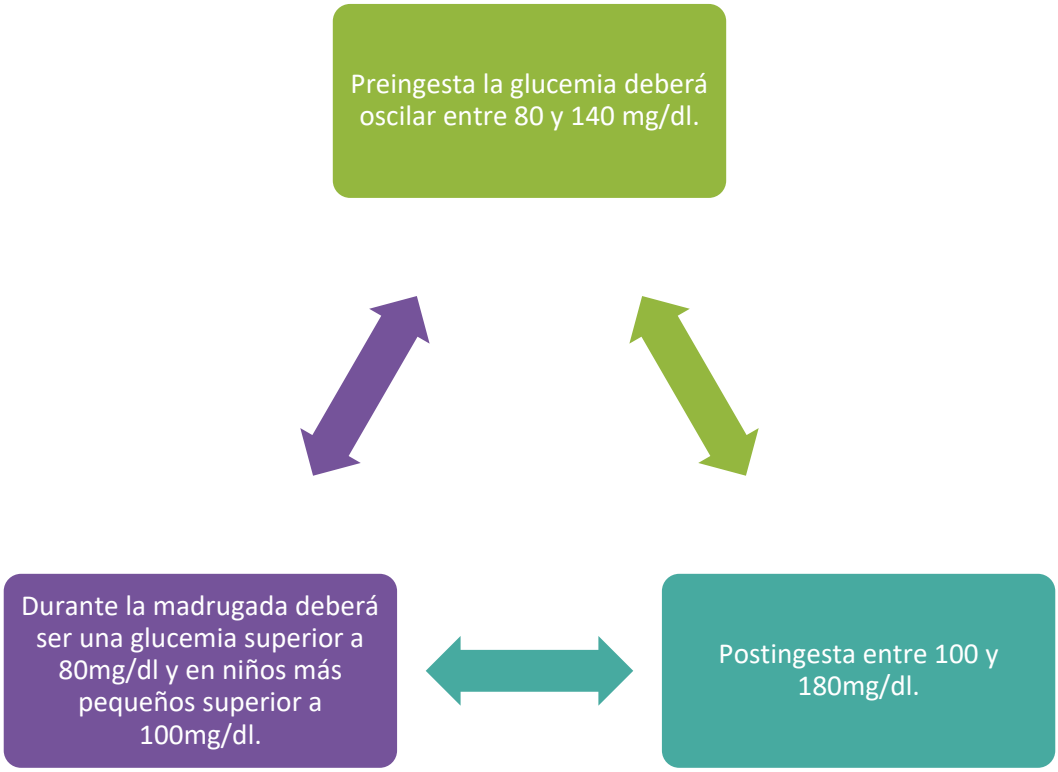
Figura 3. Fases de la diabetes.



Fuente: Elaboración propia a partir de la bibliografía consultada. [5]

Todo ello con el fin de llegar a unos niveles ideales de glucemia en distintos momentos (ver Figura 4). [5]

Figura 4. Niveles de glucemia en función del momento del día.



Fuente: Elaboración propia a partir de la bibliografía consultada. [5]

Actualmente existen unos dispositivos que se colocan a nivel subcutáneo de forma invasiva para lograr una monitorización continua de la glucosa con el fin de conocer como fluctúan los niveles de esta a lo largo del día dependiendo de la ingesta y el ejercicio. [5]

2.9 LA ALIMENTACIÓN EN LA DIABETES

La finalidad de la alimentación en el niño es cubrir todas sus necesidades energéticas para conseguir un correcto desarrollo tanto psicológico como físico. Para tal fin hay que diferenciar entre los diferentes grupos de alimentos. [5]

Por un lado, los hidratos de carbono o glúcidos popularmente conocidos como azúcares podemos separarlos en dos grupos: simples y complejos. Los hidratos de carbono simples están compuestos por una o dos moléculas de glucosa, es por ello que su absorción es muy rápida y aumentan muy rápido los niveles de glucemia; como ejemplo entramos el azúcar, la miel, las frutas y la leche. Los hidratos de carbono complejos están formados por muchas moléculas de glucosa por lo que pasarán más lentamente a sangre ya que previamente deberán descomponerse en sus formas más simples y por tanto subirán la glucemia más lentamente; como ejemplo encontramos el arroz, las pastas, el pan, las legumbres, las verduras, las patatas, etc. Por todo ello se recomienda que los niños tomen preferentemente hidratos de carbono complejos frente a los simples. [5]

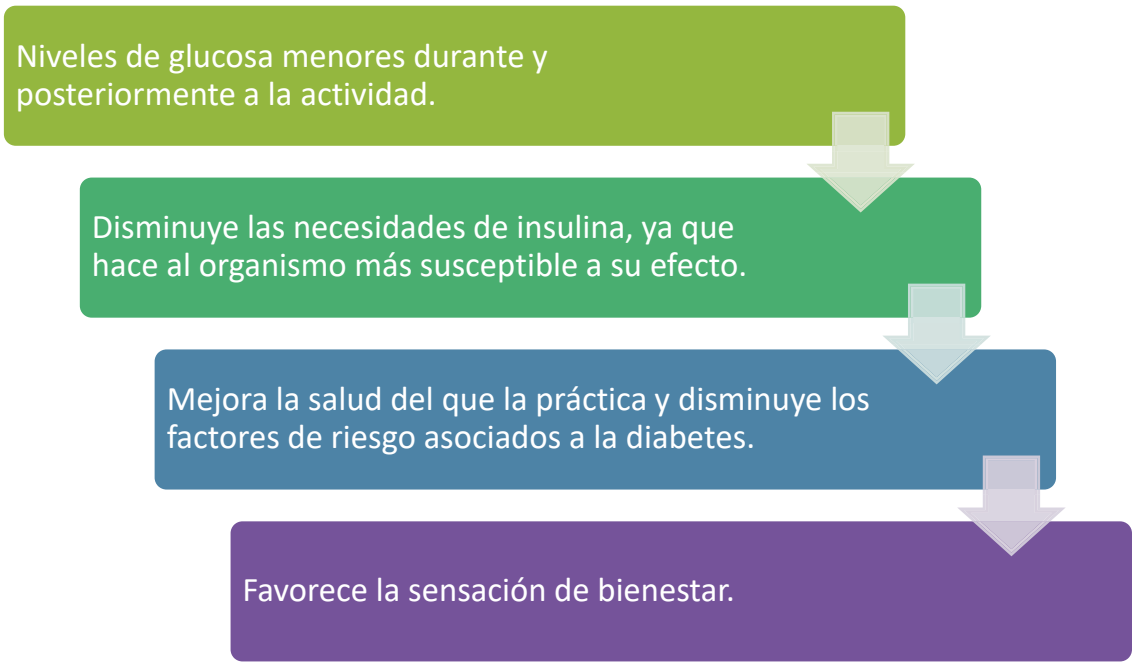
Los lípidos, conocidos como grasas son los alimentos que mayor aporte energético aportan al organismo. Dado que las personas diabéticas tienen un mayor riesgo de sufrir arterioesclerosis es importante prevenir esta mediante una alimentación saludable, por lo que las grasas saturadas presentes en embutidos, carnes grasas, mantequilla, nata, etc. están desaconsejadas de la dieta. Sin embargo, sí que se recomiendan grasas insaturadas como el aceite de oliva, las presentes en los frutos secos, en los pescados azules, etc. [5]

Por último, las proteínas son las encargadas de formar estructuralmente el organismo, las encontramos en alimentos como la carne, el pescado, los huevos, la leche, etc. [5]

2.10. LA ACTIVIDAD FÍSICA

Junto a la dieta y el tratamiento con insulina es el tercer pilar fundamental en el tratamiento de la diabetes, ya que ayuda a controlar los niveles de glucosa en sangre, por lo que es aconsejable realizar al menos treinta minutos diarios de actividad física. Entre los beneficios podemos encontrar (ver Figura 5). [5]

Figura 5. Beneficios de la actividad física en la diabetes.



Fuente: Elaboración propia a partir de la bibliografía consultada. [5]

Existen dos tipos de actividad física, la aeróbica y la anaeróbica. La aeróbica se trata de una actividad mantenida en el tiempo de moderada o baja intensidad, de manera que mantiene una correcta oxigenación, entre este tipo de ejercicio encontramos el ciclismo, la natación, el futbol, etc. Sin embargo, la actividad física anaeróbica se trata de una actividad física intensa y de duración corta, como el levantamiento de pesas, que disminuye la oxigenación celular,

además de aumentar la presión arterial, es por ello que este tipo de actividades no son recomendables. [5]

Previa a la realización de cualquier actividad física hay que medir el nivel de glucosa plasmática, adecuar las necesidades energéticas y de insulina, no inyectar la insulina en los principales grupos musculares que vayan a intervenir en el ejercicio, y estar atento a las fluctuaciones de glucemia entre las doce y veinticuatro horas posteriores a la actividad física. [5]

2.11. EDUCACIÓN DIABETOLÓGICA

Numerosos estudios desarrollados a nivel internacional recalcan que la importancia de la educación en diabetes es fundamental para poder prevenir las complicaciones tanto a corto como a largo plazo que podría sufrir el niño. También son fundamentales la implantación de programas educativos que enseñen al niño y a la familia cómo actuar ante una hiperglucemia o hipoglucemia, cómo administrarse la insulina, conocer qué tipo de dieta deben llevar y dominar cuantas raciones supone un determinado plato, qué aspectos son importantes a la hora de realizar actividad física. [6]

Se ha demostrado que una correcta educación reduce el número de episodios agudos y por tanto de ingresos hospitalarios relacionados con un mal control de la enfermedad, así como la cantidad de días de ingreso si este llega a producirse. Es por ello importante proporcionar una adecuada educación diabetológica desde el primer momento en el que se detecta un debut diabético. [6]

Una parte fundamental de la educación diabetológica es la dietética ya que en gran medida va a determinar junto con la insulina y el ejercicio físico los niveles de glucosa en el organismo. A la hora de planificara la alimentación de una persona que padece diabetes hay que tener en cuenta una serie de aspectos como son la cultura, la dieta familiar y el contexto social. La dieta por raciones consiste en diferenciar los alimentos según son frutas y verduras, hidratos de carbono y proteínas, los cuales pueden intercambiarse dentro del mismo grupo, cada ración es aquella que contiene 10g de hidratos de carbono, que dependiendo del alimento se corresponderá con una cantidad u otra, ejemplo: una ración de arroz son 15g, mientras que una ración de pan se corresponden con 20g del alimento. Dependiendo de la edad del niño su dieta se compondrá de un número de raciones u otro, que se irán ajustando según sus necesidades y momento del desarrollo. A continuación, se muestra una tabla con algunos alimentos en los que se puede ver representado a qué cantidad de alimento equivale una ración (ver Tabla 3). [7]

Tabla 3. Tabla de equivalencias entre raciones y alimentos.

ALIMENTO	CANTIDAD
Arroz	15g
Pasta	15g
Galleta tipo María	15g
Garbanzos cocidos	50g
Lentejas cocidas	50g
Pan blanco	20g
Patatas	50g
Melocotón	100g
Manzana	100g
Naranja	100g
Fresas	200g
Melón	200g
Plátano	50g
Uva	50g
Leche	200g

Fuente: Elaboración propia a partir de la bibliografía consultada. [7]

2.12. LA ROBÓTICA Y LA EDUCACIÓN

Hoy en día la robótica forma parte de la vida de los individuos de muchas formas diferentes, llegando a ser parte en su día a día. También la medicina ha apostado por las nuevas tecnologías con todos los beneficios que ello supone como por ejemplo la disminución de costos, ya que evita muchos desplazamientos por parte del personal sanitario, también puede servir como apoyo educativo en numerosos procesos, etc. Sin embargo, también es muy importante que las personas que los utilicen reciban una correcta formación para poder darles un uso adecuado y sacar el máximo provecho, por esto mismo es importante tener en cuenta que hay que adaptar los robots al contexto de las personas que lo van a utilizar. [8]

CAPÍTULO 3. EL PROBLEMA Y SUS POSIBLES SOLUCIONES

3.1. PLANTEAMIENTO DEL PROBLEMA

Como se ha especificado en el capítulo anterior la diabetes es una enfermedad común en España y en el resto del mundo. Esto no deja exento a un colectivo como los niños, los cuales desde una temprana edad deben lidiar con la enfermedad. La gestión de la diabetes requiere un continuo control y conocimiento sobre lo que ocurre, lo cual supone dificultades para el paciente a su corta edad.

Uno de los problemas más visibles, en general en las enfermedades que afectan a los niños y en específico en la diabetes, dado su alto impacto en el estilo de vida del paciente, es el hecho de que el menor pueda sentirse diferente o especial.

Otro problema que afecta a la diabetes es la gran cantidad de conocimientos y situaciones que hay que conocer y tener muy presentes para lograr una adecuada gestión de la enfermedad.

Muy relacionado con lo anterior se encuentra el hecho de que, para adquirir todos esos conocimientos necesarios, se hace imprescindible una exhaustiva educación en el tratamiento y gestión de la diabetes. Este problema se acrecienta por el hecho de ser niños, donde la educación requiere una mayor focalización y sobre todo no centrarse solo en los conocimientos estrictamente si no en forma y la interfaz con la que se pretenden transmitir.

El último problema muy relacionado con la educación en general y más aún con la educación de una enfermedad, tema mucho menos atractivo para un niño, es el reto de mantener el interés del menor en seguir aprendiendo. Esto en los niños, por naturaleza inquietos y con ganas de descubrir el mundo y probar cosas nuevas, supone un gran desafío. Para ello es necesario crear no solo una plataforma educativa, sino también atractiva para el niño y aportarle conocimientos aun cuando este piense que está jugando y divirtiéndose.

3.2. ALTERNATIVAS DE SOLUCIÓN

Como solución a los múltiplos problemas detallados en el apartado anterior se opta por una plataforma educativa consistente en la combinación de una aplicación móvil junto con un robot que complemente la interfaz y aporte el factor empático al proyecto. Para abordar el desarrollo de la plataforma se plantean diversas alternativas de solución enmarcadas en estos dos campos principales:

3.2.1. Sistema operativo móvil de desarrollo

El mercado de la telefonía móvil lo dominan básicamente dos sistemas operativos iOS y Android. El primero es el sistema desarrollado por Apple para sus teléfonos iPhone y tablets iPad. Es un sistema que solo utilizan sus propios productos por lo que está muy optimizado e integrado en el ecosistema creado por esta compañía. Por ello da una gran experiencia de uso y fluidez al usuario. En su contra se encuentra el precio de los terminales que lo integran, sobre los 800€, lo que provoca junto con la poca oferta existente, que tengan una relativa baja cuota de mercado suponiendo menos del 15%.[(36)]

Por otro lado, se encuentra Android un sistema operativo desarrollado por Google, el cual, es el más extendido y que más móviles integran. Debido a esta heterogeneidad es un sistema menos cerrado y menos optimizado pero que últimamente ha avanzado mucho. Android acapara casi el 85% de la cuota de mercado, esto permite que los terminales más económicos opten por este sistema. [(36)]

Por último, existen otros sistemas como Windows Phone, Blackeberry 6, Symbian o Firefox OS, entre todos suponen menos del 1% del mercado por lo que muy pocos terminales lo incluyen, así como muy pocos desarrolladores invierten su esfuerzo en crear aplicaciones para ellos. [(36)]

3.2.2. Dispositivo robótico

En el mercado existen multitud de alternativas de robots programables. Aquí se seleccionan 3 diferentes por concordar bien con el proyecto tanto a nivel técnico como económico.

Lego MindStorm EV3: Lego lleva desarrollando sus robots desde hace 20 años y ha creado una plataforma muy potente de desarrollo y programación de sus robots mediante SCRATCH, gráficamente o código. Aunque existen diversos lenguajes (.NET, LeJOS, RobotC, etc) en el que se pueden programar la mayoría son propios y se usan principalmente para la programación de estos dispositivos. Su morfología es la de un ladrillo con entradas y salidas para sus sensores y actuadores de los cuales hay una gran variedad, pudiendo crear multitud de robots diferentes. Esta alternativa es la que mayores opciones brinda en cuanto a su personalización y mejoras. Su precio oscila alrededor de los 350€. [(37)]

Zowi: este robot ha sido diseñado por la empresa española BQ con el objetivo de acercar la programación a los más pequeños. Es un robot bípedo que como sensores incluye un sensor de proximidad mediante ultrasonidos y un micrófono. En cuanto actuadores Zowi viene de serie con un zumbador, cuatro motores de corriente continua, y una matriz led de 5x6. También dispone de un módulo Bluetooth que permite su conexión inalámbrica. Su programación puede realizarse mediante Scratch o en lenguaje Arduino directamente. Esta alternativa tiene un coste de 100€. [(38)]

MBot: este dispositivo ha sido desarrollado por la compañía Makeblock surgida a partir de la plataforma de financiación Kickstarter. Mbot es el robot más bajo de la gama que ofrece esta empresa, pero con un precio muy ajustado ofrece una gran cantidad de funcionalidades. En cuanto a su forma, mBot es un robot de 3 ruedas, dos atrás movidas por dos motores DC y una delante que gira libremente pero solo en una dirección. En cuanto a sensores cuenta con multitud de sensores y actuadores en el que destaca su matriz led de 8x16, la cual da libertad para crear multitud de caras y expresiones. La conexión que ofrece también es Bluetooth. Está

basado en la placa Arduino UNO y por tanto su programación puede realizarse mediante SCRATCH o directamente en lenguaje Arduino. El precio de este dispositivo es de 98€. [31]

3.3. JUSTIFICACIÓN DE LA ELECCIÓN

Para la elección de tanto el sistema operativo como el dispositivo robótico utilizado se han seguido dos criterios fundamentales, el económico y el técnico.

3.3.1. Android. El sistema operativo elegido

Como se ha mencionado con anterioridad Android es un sistema operativo móvil que integran la mayoría de los teléfonos del mercado, incluyendo también los de menor coste. Por tanto, siguiendo el objetivo de que la aplicación pueda llegar al porcentaje de público mayor posible, así como buscando en todo momento que esta opción educativa sea lo más accesible económicamente, se ha elegido Android como plataforma de desarrollo.

Otro criterio muy determinante en la elección es el hecho de que, durante el grado cursado de Ingeniería en Tecnologías Industriales en la UPV, una de las asignaturas optativas escogidas haya sido Desarrollo de Aplicaciones para Dispositivos Móviles. Asignatura en la cual, se reciben los conocimientos básicos que hacen posible crear aplicaciones sencillas y adentrarse en el mundo de la programación en Android. De esta manera la asignatura da la base necesaria para poder ampliar conocimientos sin toparse con un muro insalvable como sería comenzar desde cero.

3.3.2. MBot. El dispositivo robótico elegido

En primera instancia se descartó los robots de la casa Lego por tener un precio demasiado elevado para el fin que tenía el proyecto. Este fin no es otro que desarrollar una plataforma educativa para niños con diabetes tipo I lo más accesible posible tanto en el mercado como económicamente se refiere. Por ello, la elección se redujo al dispositivo de BQ Zowi y al mBot de Makeblock. Ambos muy parecidos a priori ya que su corazón es el mismo, una placa Arduino, y cuentan con sensores similares. Pero lo que ha decantado la balanza es la interfaz que ofrece mBot frente al robot Zowi. Esto es sumamente importante pues con el dispositivo robótico se busca dotar al proyecto de una interfaz que permita una interacción más natural entre el niño y la aplicación. Así como, acercar los aspectos educativos que constituyen la base del proyecto de una forma más amigable y lúdica. Con este objetivo en mente, mBot, con un precio prácticamente idéntico a su competidor, ofrece una matriz de leds notablemente más grande, lo que permite crear expresiones más reales y crear una mayor empatía con el menor. Por otro lado, el hecho de que su sistema de movimiento sea mediante ruedas permite mucha más libertad de movimiento e interacción que el sistema antropomórfico de Zowi que lo hace mucho más estático y torpe. Por último, el hecho de incorporar una serie de leds que potencian la interfaz que ofrece este robot frente al dispositivo de BQ hacen que la elección se decante a favor del mBot.

Por otro lado, atendiendo a un criterio más técnico y menos de relación funcionalidades-precio destaca el hecho de que el corazón del mBot se base en una placa Arduino. Esto facilita mucho su programación ya que es un lenguaje muy similar a C, además de permitir una conexión relativamente sencilla desde la aplicación desarrollada en Android. Esta característica también

influyó en que los robots de Lego no fueran los más adecuados para el proyecto. Otro aspecto para tener en cuenta es el hecho de haber cursado otra asignatura optativa llamada Laboratorio de Automatización y Control, donde en una de las cuatro partes que componen la asignatura, específicamente el en el seminario de robótica, se sentaron las bases del control y programación de robots con sistema motriz basado en ruedas con configuración diferencial. Esto contribuyó aún más para que finalmente mBot se posicionase como la mejor alternativa.

CAPÍTULO 4. GLUCOEDUCA, LA APLICACIÓN MÓVIL

4.1. DEFINICIÓN

GlucoEduca se presenta como una aplicación desarrollada para dispositivos móviles Android que busca educar a los niños acerca de cómo tratar la diabetes tipo 1, además de ofrecer una monitorización continua y un tratamiento de las lecturas por medio de estadísticas y demás funcionalidades como se detallará más adelante.

Para el desarrollo de esta aplicación se han utilizado diversas funcionalidades que incluye la programación en Android, así como software adicional para enriquecer la aplicación como se explicará a continuación.

4.2. ANDROID STUDIO, ENTORNO DE PROGRAMACIÓN EN JAVA

Android Studio es el entorno de desarrollo integrado (IDE) oficial para la programación de aplicaciones en el sistema operativo Android. Este software sustituyó a Eclipse como IDE oficial de Android. Está basado en Intellij IDEA desarrollado por JetBrains y disponible gratuitamente bajo la licencia Apache 2.0. Entre sus características destaca su separación, pero manteniendo la relación, entre los dos pilares de la programación en Android, el desarrollo de la interfaz gráfica y el código en lenguaje JAVA que ejecuta dicha interfaz. Además, incluye soporte para la construcción basada en Gradle, una herramienta de automatización en el proceso de construcción de un proyecto en Java que agiliza la compilación del código fuente a tiempo real. Por otro lado, destaca su potente y estable emulador muy útil para probar las aplicaciones durante su desarrollo sin necesidad de usar un dispositivo externo. [(10)]

La programación en Android Studio se basa en parte, como se ha introducido en el párrafo anterior, en un archivo en formato XML que constituye la interfaz gráfica que verá el usuario y con la que podrá interactuar. A esta interfaz se le da vida a través del código en lenguaje Java, el cual se ejecuta junto a la interfaz en función de la interacción que el usuario desarrolle durante su uso de la aplicación. De ahí, la importancia de un buen diseño para crear una buena aplicación, que al fin y al cabo es un producto que va a ser utilizado por personas, y en este caso niños los cuales agradecerán una interfaz ordenada e intuitiva aparte de que el código programado haga lo que pretenda hacer. [(10)]

Java se define como un lenguaje de programación de propósito general orientado a objetos. Está basado en gran medida en C y C++, pero incorpora menos utilidades en bajo nivel que estos. El objetivo que busca este lenguaje es el de permitir a los desarrolladores escribir una vez el código y poder ejecutarlo en cualquier dispositivo a través de lo que se conoce como máquina virtual de Java (JVM) sin importar la arquitectura del dispositivo. Esta propiedad lo hace ideal para el desarrollo de aplicaciones para Android donde existe una gran heterogeneidad en cuanto a hardware integrado en los dispositivos. [(11)]

4.3. FIREBASE

Firebase es un conjunto de herramientas diseñadas por Google para permitir a los desarrolladores incluir en sus aplicaciones tratamiento de datos en la nube y crear aplicaciones de mayor calidad. Algunos de las funciones que incluye son:

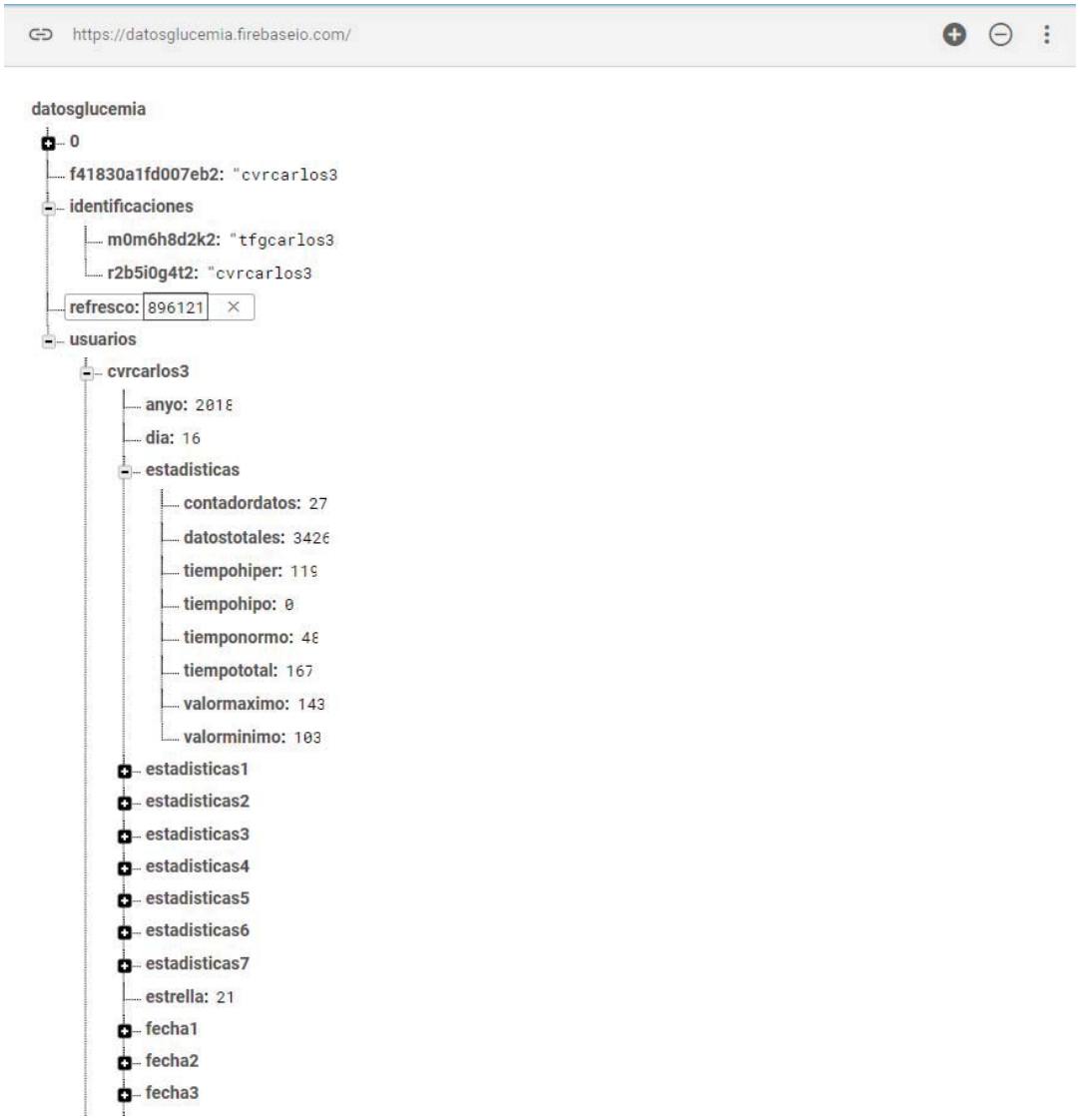
- Base de datos a tiempo real: esta función permite tener una base de datos en la nube que se actualiza cada vez que hay un cambio en ella ya sea de lectura o escritura. En este proyecto la base de datos es el eje fundamental de la aplicación, ya que en ella se almacena toda la información y datos referentes a cada usuario, de los cuales la aplicación necesitará disponer. De esta forma el usuario no liga la aplicación a un dispositivo móvil en concreto, sino que, al disponer de todos sus datos en la nube, simplemente iniciando sesión con su cuenta en otro dispositivo, puede continuar usando GlucoEduca con el mismo progreso que en el móvil anterior. Como su nombre indica es una base de datos que se actualiza a tiempo real esto en el entorno de desarrollo se traduce en que la forma de comunicarse con ella es mediante un "listener" o escuchador que solo se activa cuando se produce una variación en alguna variable de la base de datos. Esto implica que durante toda la implementación de la aplicación para leer algún valor de la base de datos sea necesario haber anteriormente modificado algún otro de esta. Como solución a este problema se ha optado por la inclusión de una variable de refresco en la base de datos, que mediante código se actualice cada vez que la aplicación precise leer algún dato. Para evitar duplicidades y solapes en accesos simultáneos, cada usuario cuenta con su propia variable de refresco. [13]

- Autentificación: Firebase ofrece una gestión de usuarios bastante completa donde el usuario puede registrarse en la aplicación mediante diferentes métodos como con una cuenta de Google, Facebook o Twitter o mediante un correo y contraseña. Esta última ha sido la opción implementada en la aplicación, ya que como va destinada a niños cabe más la posibilidad de que tengan un correo electrónico que una cuenta en las demás plataformas anteriormente mencionadas. Este acceso mediante correo y contraseña es un acceso seguro, que evita la creación de cuentas falsas, debido a que permite la implementación en la aplicación de un sistema de verificación de identidades mediante el envío de un correo electrónico. Este se envía a la dirección de correo que se desea registrar en la aplicación junto a un enlace que debe pulsar el usuario para activar su cuenta. De esta forma se evita una sobrecarga o saturación en el sistema por el intento de creación de cuentas falsas o duplicadas. Esta función es muy utilizada en este proyecto para realizar la gestión de usuarios y sus datos como se comprobará más adelante. [12]

- Firebase también cuenta con otras funcionalidades como análisis y estadísticas de uso, así como un sistema de detección de errores en la nube. Estas funciones no han sido utilizadas en este proyecto ya que no se ha llegado a esa fase del desarrollo. Pero en futuras mejoras donde la aplicación sea lanzada para su testeo estas funciones que ofrece Firebase serán de gran utilidad. [13]

En esta aplicación en concreto a parte de para tener una sincronización online de datos relativos al progreso del usuario en la aplicación, la base de datos a tiempo real tiene otro importante uso, la monitorización. Para lograr esta lectura de datos continua, lo ideal hubiera sido conectar la aplicación con un sensor de glucosa real. Pero debido a las limitaciones lógicas que esto ofrece en el ámbito en el que se desarrolla un trabajo final de grado esta opción fue descartada. Como alternativa se optó por cargar en la base de datos un estudio que recoge más de 7500 muestras del nivel de glucosa de un paciente de real. Este estudio fue proporcionado por los tutores del proyecto en formato de hoja de cálculo. Debido a que la base de datos solo admite la importación de información en formato JSON fue necesario una previa conversión de la hoja de cálculo a formato separado por comas (CSV). Una vez el archivo tuvo este formato con la ayuda de un conversor online se transformó a formato JSON, pudiéndose así importar sin problemas a la base de datos y comenzar la lectura continua en la aplicación. La metodología utilizada para la lectura ha sido la de asignar a cada muestra un nombre constituido por la cadena de caracteres “Dato”, seguida de la posición que ocupa en el estudio. De este modo la muestra numero 52 quedará almacenada en la base de datos en la variable “Dato52”. Con este sistema de identificación resulta sencillo la lectura desde la aplicación simplemente utilizando la suma de cadenas de caracteres entre la cadena “Dato” y una variable que vaya aumentando en una unidad cada vez que se produce una lectura.

Figura 6. Estructura de la base de datos a tiempo real



Fuente: Captura de pantalla de la base de datos a tiempo real de Firebase

Como se observa en la Figura 6, la estructura de la base de datos tiene forma de árbol, donde como campos principales se han incluido un apartado en el que se asocia el código de identificación de cada dispositivo móvil con el del usuario que está conectado a él. Otro campo de identificaciones, el cual permite, como se verá en el apartado donde se describe la pantalla de estadísticas con acceso remoto, la posibilidad de realizar una consulta del estado de la monitorización de la glucosa en sangre del niño desde cualquier dispositivo. Luego se encuentra, personalizado para cada usuario registrado en la aplicación, todos los campos necesarios para el funcionamiento de esta. Como por ejemplo todos los relativos a las estadísticas y al control de tiempos y fechas. También se almacena todo el progreso que el usuario ha ido acumulando en la aplicación, permitiendo así independizar la cuenta del usuario del dispositivo móvil. De esta forma se ofrece la posibilidad de sincronizar los datos del usuario en cualquier teléfono que tenga descargada la aplicación.

4.4. GIMP, MANIPULADOR DE IMÁGENES

La aplicación va destinada a un público infantil, por ello la interfaz gráfica debe ser muy visual y llamativa con el objetivo de crear una cierta empatía entre el usuario y la aplicación. Por ello GlucoEduca cuenta con muchas imágenes dispuestas en fondos, botones y animaciones. En muchas ocasiones en la creación de imágenes una sola no era suficiente por ello ha sido necesario el retoque y composición de varias de ellas con el objetivo de crear una imagen atractiva para el usuario. Para esta tarea se ha utilizado GIMP un software libre y público de manipulación de imágenes bajo la licencia pública general de GNU. Este software no ha sido solo utilizado para la creación de iconos, sino que también se han creado con él los escenarios y personajes que conforman el modo de historia interactiva como se detallará en el capítulo 4.11.11. Como ejemplo de esto, la creación del personaje que interpreta a la madre del protagonista resulta de la conjunción y manipulación de fragmentos de 6 imágenes distintas. [15]

Otro incentivo para el uso de este software ha venido impuesto por la limitada capacidad de memoria RAM que presentan los dispositivos móviles actuales. Esta memoria es la más utilizada por el dispositivo cuando ejecuta una aplicación. Debido a este motivo, la creación de una aplicación con una carga de imágenes importante supondría graves problemas en el espacio de memoria, así como en la fluidez y experiencia de uso por parte del usuario. A raíz de esta limitación, ha sido necesario la reducción del tamaño de prácticamente todos los recursos gráficos incluidos en GlucoEduca. Para esta operación el software GIMP ha sido de gran ayuda. [14]

4.5. AUDACITY, EDITOR DE AUDIO

En este proyecto con el objetivo de dar protagonismo al dispositivo robótico frente a la aplicación como interfaz de comunicación con el usuario no se han incluido muchos recursos de audio. En concreto solo se ha incluido música y sonidos en la historia interactiva como se explicará en el apartado 4.11.11. En el caso de la música en concreto como en toda la aplicación en general se ha tratado de solo incluir recursos sin derechos de autor con el objetivo de crear un proyecto sin ataduras legales. Esto condiciona mucho la oferta disponible de ,en este caso, recursos de audio. Debido a ello la música seleccionada no disponía de la duración suficiente como para ser utilizada como música de fondo de la historia. A raíz de esto, usando el software Audacity, un editor de audio de código libre bajo la licencia pública general de GNU, se ha triplicado su duración juntando el mismo archivo en tres ocasiones incluyendo un efecto de transición entre las diferentes pistas de audio. [16]

Como se verá en el apartado donde se relata la historia interactiva (4.11.11) los personajes emiten un sonido simulando su voz cada vez que hablan. Este efecto se ha logrado mediante el uso del software Audacity. Para ello, se ha realizado grabaciones personales de unos 10-15 segundos para posteriormente, alterando el tono y sobre todo la velocidad de estas pistas de audio, crear la simulación de la voz. De esta forma se consigue un sonido agradable y reconocible de dibujos animados infantiles, donde no se articulan palabras como tal, permitiendo una interacción sensorial mayor entre el usuario y la historia. Así como, un mejor

control del tiempo de avance de la historia ya que el sonido alerta de que el personaje ha cambiado su diálogo y está diciendo algo diferente. [15]

4.6. REPRESENTACIÓN DE GRÁFICOS. LIBRERÍAS UTILIZADAS

En este proyecto donde el tratamiento de las estadísticas recabadas a partir de la monitorización continua tiene un papel tan fundamental, se hace indispensable el uso de gráficos para la representación visual y por tanto más intuitiva de dichas estadísticas. En este caso en concreto se hace uso de dos de las librerías gratuitas más conocidas en el desarrollo de aplicaciones Android como son GraphView y MPAndroidChart. Ambas librerías se hallan publicadas bajo la licencia Apache versión 2.0., la cual da derecho a su uso sin necesidad de reconocimiento ni pago de royalties. [17] [18]

En cuanto al uso de dos librerías diferentes en el proyecto se debe a las diferentes prestaciones que ofrece cada una con respecto a la otra.

GraphView permite crear, sin necesidad de modificar la propia librería, gráficos que se actualicen a tiempo real. Lo cual será utilizado en los modos de simulación de glucosa detallados en los apartados 4.11.4 y 4.11.7. Esta función de gráfico a tiempo real ha sido implementado mediante la superposición de gráficos cada vez más completos encima del anterior. De este modo un gráfico con un vector de valores tanto para el eje de abscisas como el de ordenadas de tamaño una unidad mayor crea un gráfico que se superpone al anterior. Esto da la sensación de que el gráfico se va generando valor a valor con una cierta cadencia de tiempo. A pesar de ser una librería muy completa no permite la creación de gráficos sectoriales ni de dibujar líneas límite muy útiles para poder ver de un vistazo si los valores graficados se encuentran dentro de los márgenes establecidos. Por ello se recurrió a otra librería que da más capacidad de personalización y permite las dos premisas anteriormente mencionadas. Esta ha sido MPAndroidChart la cual ha sido utilizada para la creación de todos los gráficos del modo estadísticas como se verá más adelante en el apartado 4.11.9. Esta librería a pesar de no permitir la creación de gráficos a tiempo real permite dibujar gráficos muy vistosos y personalizables, pero con el inconveniente de que se debe proporcionar todos los puntos que conforman el gráfico nada más iniciar la actividad que los contiene. Esto conlleva que sea necesario el uso de variables globales y de una pantalla de transición para transmitir todos los datos a los gráficos en el momento de su creación (ver apartado 4.11.9). [17] [18]

4.7. ANIMACIONES

Con el objeto de dotar a la aplicación de una mayor vistosidad y ser más atractiva, se han incluido varias animaciones en su desarrollo. Todas estas animaciones han sido descargadas a través de la web LottieFiles y usan la librería Lottie para su implementación. Dicha librería y animaciones se encuentran bajo la licencia Creative Commons versión 4.0. [19]

Se hace uso de este tipo de animaciones y no de otras como podría ser el formato GIF debido a que estas últimas son muchos más pesadas y requieren muchos más recursos que las anteriores para ser ejecutadas. Las animaciones de Lottie tienen formato JSON, el mismo que utilizaba la base de datos, el cual las hace muy livianas tanto para su almacenamiento como ejecución. Para su implementación en la aplicación es necesario en primer lugar cargar la

librería, luego crear un objeto animación mediante dicha librería y referenciarlo a la animación anteriormente incluida en el directorio de recursos de la aplicación. Tras todo esto diversos métodos incluidos en la librería permiten su reproducción y pausa cuando se desee. [19]

4.8. CONTROL DE TIEMPOS Y FECHAS

En este proyecto al tener la necesidad de almacenar y tratar datos relativos a un número de días concreto se hace necesario una gestión de las fechas y de conocer en todo momento cual es la fecha actual y que grupo de estadísticas pertenecen a ese día. Para lograr este control, en la base de datos en la nube se guardan las fechas relativas a los últimos 7 días. Cuando se inicia la aplicación, mediante la clase "Date" se obtiene del dispositivo la fecha actual para posteriormente comenzar un proceso de comparación con las diferentes fechas almacenadas en la base de datos, con el fin de conocer por ejemplo si se está en la fecha numero 2 o 5. Esta comparación se realiza mediante la utilización del método getTime que devuelve la diferencia en milisegundos entre la fecha introducida y la fecha correspondiente a enero 1, 1970, 00:00:00 GMT. Conociendo este dato y mediante comparación con las demás fechas se es capaz de ordenarlas y saber el número de fecha correspondiente al día actual. De este modo se puede acceder a las estadísticas correspondientes a dicho número de fecha ya que estas están numeradas por la siguiente cadena: "Estadisticas1, …, Estadisticas7". Una vez resuelto el problema de saber en qué número de fecha se está viene el problema de actualizar dichas fechas y estadísticas relacionadas cuando hayan pasado esos 7 días que se almacenan. Para ello se utiliza otra variable de tipo fecha que sería la correspondiente a la fecha numero 8 mediante la cual una vez se supera significa que se han superado los 7 días y deben actualizarse todos los campos para almacenar los datos de los siguientes 7 días. Un ejemplo de este proceso sería el siguiente: el usuario se registra en la aplicación el día 12 de junio, en ese momento se crea en la base de datos los campos de fechas y estadísticas relativos a los días del 12 al 18 de junio y la variable de fecha tope que representaría al 19 junio. Todas estas variables se inicializan a la hora 00:00:00, es decir cuando comienza el día. En el momento que el día 18 finalice y comience el 19 la aplicación detectará que se ha sobrepasado la fecha tope y actualizará los campos para almacenar las estadísticas y fechas de los días del 19 al 25 de junio, así como la siguiente fecha tope correspondiente al día 26. [20]

Por otro lado, se encuentra el control de tiempos que se realiza en el modo preguntas (ver apartado 4.11.5). En él se muestra en todo momento una cuenta atrás actualizada a tiempo real de 24 horas, la cual se inicia cada vez que el usuario responde una pregunta. Esto se consigue de la siguiente forma: cuando el usuario responde una pregunta, en la base de datos se carga una variable de tipo fecha, resultado de obtener la fecha actual con la clase Date y sumarle un día mediante la clase Calendar. El hecho de hacer uso de estas clases y no hacerlo manualmente facilita la gestión de si los meses tienen 30 o 31 días o si el año es bisiesto ya que estas clases lo tienen en cuenta. Una vez cargada dicha fecha la aplicación puede leerla y compararla con la fecha actual que va leyendo constantemente del dispositivo móvil. De esta forma en todo momento se obtiene la diferencia en milisegundos entre ambas fechas como se ha comentado anteriormente. Luego esta diferencia se separa en días, horas, minutos y segundos y se muestra en un texto que se actualiza cada segundo. [20]

Para la representación de las fechas en los modos de estadísticas es necesario darle un formato diferente al que proporciona la clase "Date". En concreto en estos modos se han representado con el formato "dd/m/aaaa/". Esto se consigue, en primer lugar, guardando en tres variables distintas el número correspondiente al día, mes y año a través del método "SimpleDateFormat". Luego se genera una cadena de caracteres con la suma de las variables anteriores y las barras inclinadas. Esta cadena ya se transmite mediante variables globales a las actividades que lo precisen. [20]

4.9. VARIABLES GLOBALES

En el entorno de desarrollo de aplicaciones para el sistema operativo móvil Android existen básicamente dos formas de pasar información entre dos actividades de una misma aplicación. El primero de ellos y más intuitivo es el "Bundle", básicamente esto es un contenedor de información heterogénea en distintos formatos que se crea antes de finalizar una actividad y se vuelca la información al comienzo de la siguiente. Este método implica varias limitaciones, la primera y más importante es que la transmisión del "Bundle" solo se puede hacer entre dos actividades y se destruye cuando esta transmisión finaliza. Esto deriva en la necesidad de crear un "Bundle" cada vez que se pase de actividad, si es necesario compartir una cierta variable como es el caso. La segunda limitación afecta a la fluidez en la transición entre actividades o pantallas. Como se ha explicado anteriormente el "Bundle" se crea antes de finalizar una actividad y este se transmite mientras se produce la transición a la siguiente actividad. Esto implica que el tiempo de transición aumente considerablemente. La última limitación que ofrece este método y que hace que sea descartado para este proyecto es la imposibilidad de enviar información a otra actividad durante la ejecución de esta primera, es decir antes de que se produzca la transición. Esto hace que sea inadecuado para el sistema de alarma sobre la monitorización continua que está implementado en todas las actividades como se detallará más adelante en el apartado 4.11.2. [21]

Por todos estos motivos para este proyecto se ha optado por el uso de variables globales, es decir variables que pueden ser leídas y sobrescritas desde cualquier actividad de la aplicación. El manejo de estas variables está implementado mediante una clase Java donde se declara cada variable con su tipo y nombre además de un método de escritura y otro de lectura para cada una de ellas, como se puede ver en la figura "xxgt" donde se encuentra un fragmento del código implementado. Gracias al uso de este método la actualización y lectura esta se puede hacer en el momento oportuno durante la ejecución de cada actividad sin sobrecargar la transición entre estas y permitiendo la monitorización continua de los valores de la glucosa en sangre.

4.10. LA CONEXIÓN BLUETOOTH, Y SU EFECTO EN LA TRANSICIÓN ENTRE ACTIVIDADES

Como se viene desarrollando a lo largo de esta memoria, GlucoEduca busca educar a los niños a través de una interacción constante entre la aplicación móvil y el dispositivo robótico. Para ello la conexión elegida ha sido el enlace mediante la tecnología Bluetooth. Tecnología que permite mediante radiofrecuencia en la banda de los 2,4 GHz conectar dos dispositivos entre sí. A pesar de las limitaciones que ofrece esta tecnología como el corto alcance o la imposibilidad de transmitir muchos datos, resulta ideal para este proyecto. Esto es así porque se busca que en

todo momento el robot esté junto al niño como si de su mascota se tratara interactuando con él, interacción la cual se realiza a través de la aplicación móvil. Otro aspecto decantador ha sido su precio y sencillez respecto a otras tecnologías de comunicación como podría haber sido la conexión WiFi. [9]

Una vez seleccionada la tecnología de comunicación a utilizar el siguiente paso es el de programar la conexión en la aplicación móvil. Para ello el primer paso es comprobar si el dispositivo cuenta con adaptador Bluetooth y este se encuentra activado. Una vez realizada dicha comprobación el siguiente paso es el de localizar el dispositivo con el que se desea realizar la conexión, en este caso el robot mBot. Tras su localización se dispone a la apertura de un puerto de comunicación a través del cual transmitir la información. El último paso es el de enviar información que a través de dicho puerto esto se realiza mediante una función que recibe un dato tipo entero y lo envía al robot a través del puerto para que este lo reciba en su puerto serie tal como se explicará con más detalle en el capítulo 5.4. Equivalente a la apertura del puerto se ha implementado una función que cierra este y cesa la conexión entre ambos dispositivos. [9]

En una aplicación móvil el tener activo y en funcionamiento una conexión como la Bluetooth acapara muchos recursos valiosos del sistema, más aún si esta conexión perdura en el tiempo sin reinicializarla pudiendo dar errores que originen un cierre inesperado de la aplicación o “crash”. Para evitar este problema que fue descubierto de una forma totalmente empírica se ha optado por realizar un cese de la conexión y por ende una destrucción del puerto de comunicación en cada transición entre actividades. Para posteriormente volver a inicializarlo al lanzar la nueva actividad. Esto evita que el puerto se sature, evitando así problemas en la conexión. Evidentemente este método se puede aplicar ya que tanto el cese como la reapertura de los puertos se hace de una forma casi instantánea la cual él usuario no llega a percibir creyendo que la conexión no se corta nunca.

Por otro lado, como se verá más adelante en el desarrollo de la aplicación, no todos los modos de esta requieren una interacción con el robot y por tanto tener establecida una conexión con este. Esto implica que sea necesario diferenciar entre dos casos, la entrada a dichos modos con una conexión establecida o sin ella. Es básico tenerlo bien diferenciado porque el intento de cerrar un puerto de comunicación que no existe o el de abrir uno que ya está abierto puede conllevar el “crash” de la aplicación. Este problema se soluciona nuevamente con una variable global la cual informa en todo momento a las actividades si existe una conexión Bluetooth activa pudiendo así realizar la llamada o no a las funciones encargadas de su gestión.

Además de esto, como se verá a continuación en el desarrollo de la aplicación, el usuario es notificado de que cuando decide iniciar la conexión bluetooth con el robot esta se ha establecido correctamente, pero debido a que como se ha mencionado anteriormente esta conexión se establece en cada transición entre actividades esta notificación se mostraría más veces de las necesarias. Para ello haciendo uso nuevamente de una variable global (“primera vez”) se informa a la aplicación que ya se ha notificado al usuario la primera vez y que no se vuelva a realizar al menos que este decida cesar voluntariamente la conexión mediante el botón del que dispone para ello.

4.11. ESTRUCTURA DE LA APLICACIÓN Y SUS DIFERENTES PANTALLAS

GlucoEduca se compone de más de 5500 líneas de código conformando los 25 archivos relativos a la interfaz gráfica ("layout") de las actividades, junto a 10 "fragments" y 6 archivos gráficos que conforman los menús desplegables. Por otro lado, se han implementado más de 16000 líneas de código en lenguaje Java separadas en las 25 actividades que conforman la aplicación, junto a tres clases más relativas al manejo de las variables globales y a los adaptadores de los menús y gráficas.

Todo ello deriva en una aplicación distribuidas en 8 modos de juegos educativos y monitorización de la glucosa en sangre. Cada uno de los cuales se describirán detalladamente a continuación.

4.11.1. Pantalla de inicio de sesión

Esta es la actividad que primero se lanza al abrir la aplicación. Se basa en una identificación mediante correo electrónico y contraseña implementada a través del sistema se autentificación de Google en Firebase. Cuando el usuario se encuentra en esta pantalla tiene ante él un campo para introducir su correo electrónico y una contraseña con el fin de proceder a realizar su registro. Para ello, cuenta con dos botones uno de registro y otro para entrar a la aplicación una vez registrado. Ambos botones cuentan con código que supervisa si los dos campos de texto han sido rellenados, así como si el correo introducido tiene el formato adecuado. Una vez el usuario pulsa el botón de registro el sistema de autentificación de Firebase implementado envía un e-mail a la cuenta de correo introducida con una breve descripción del proceso de registro y un enlace, para que cuando se pulse sobre el poder verificar esa dirección de correo y así evitar la creación masiva de cuentas falsas, sobrecargando la base de datos. Una vez completada la verificación, por pantalla se informa mediante un "Toast" (mensaje que desaparece en un breve periodo de tiempo cuya función es informar al usuario sin interferir en ningún momento con las acciones que este estuviera realizando) que el proceso se ha realizado correctamente. Simultáneamente al lanzamiento del "Toast" se almacenan los datos de acceso en el registro de usuarios con acceso a la aplicación. Por otro lado, también se realiza la creación e inicialización de todos los campos y variables, mencionados anteriormente en el apartado 4.3 relativo a Firebase, con los que cuenta el usuario en la base de datos a tiempo real. (ver Figura 7) [(22)]

Una vez el registro ha concluido, el usuario pulsará el botón de entrar. En el caso de que volviese a pulsar el botón de registro un "Toast" le informaría de que ya se ha registrado esa cuenta. Al hacer clic sobre el botón de entrar se muestra un diálogo informándole de que se está comprobando que se ha producido la verificación del correo electrónico y en caso negativo se informa por pantalla de que no se ha realizado. Si todo está correcto se da acceso a la aplicación y se realiza la llamada a la pantalla de selección de modo.

Esta pantalla de inicio de sesión también tiene implementado lo que se conoce como un "Listener" o escuchador, que en este caso se encarga de que, si ya se ha producido la entrada a la aplicación y no se ha cerrado la sesión, al volver a entrar en la aplicación esta pantalla no llega a mostrarse y se lanza directamente la pantalla de selección de modo. Esto hace mucho más agradable el uso de la aplicación eliminando trabas que perjudiquen el interés del usuario por usar GlucoEduca.

Figura 7. Pantalla de inicio de sesión

Fuente: Captura de pantalla de la aplicación.

4.11.2. PANTALLA DE SELECCIÓN DE MODO

Es la pantalla principal y desde ella se tiene acceso a todos los aspectos de la aplicación y a la que se regresa al salir de los diferentes modos que se incluyen.

Al ver esta actividad lo primero que llama la atención es un gran "ToggleButton" personalizado en forma de interruptor (botón que puede estar en dos estados bien diferenciados, normalmente encendido y apagado, y en función de donde se encuentre se ejecuta un fragmento de código u otro). Colgando de este botón se encuentra un cartel que indica la función de este interruptor, activar y desactivar la monitorización (ver Figura 8). Esto se refiere a la lectura de datos continua de glucosa desde la base de datos simulándose la lectura desde un sensor. Este proceso se realiza mediante dos hilos de ejecución diferentes en segundo plano ("Threads"). [23]

El primero de ellos se encarga de leer consecutivamente cada 5 segundos un dato de glucosa en sangre desde Firebase. El punto de partida de la lectura se realiza de manera aleatoria con el fin de simular una mayor variabilidad en la monitorización. Esta lectura se fundamenta en la conjunción de un "Listener" que se ejecuta cuando se produce una variación en algún campo de la base de datos junto con un refresco realizado en el "Thread" que produce esta variación. Este refresco no es más que la escritura de un valor aleatorio en un campo de refresco de la base de datos propio a cada usuario para no producir conflictos cuando haya varias personas conectadas haciendo uso simultáneo de la base de datos. Cabe destacar que en este "Listener" también se realiza, una vez se inicia la actividad, el manejo de las fechas y la selección del número de fecha (ver apartado 4.8), así como, la actualización de las fechas y estadísticas almacenadas en "Firebase". En este hilo también se realiza la actualización de las diferentes estadísticas en la base de datos. El motivo de realizar este proceso en este hilo y no en el segundo hilo, donde se conforman las estadísticas, se debe a que el segundo hilo tiene una cadencia de

1 segundo lo que provocaría demasiada carga de trabajo para la aplicación, base de datos y conexión a internet.

El segundo hilo como se ha comentado anteriormente se ejecuta cada segundo, el fin de esto se basa en que es el hilo donde se van conformando las estadísticas del tiempo que está en hiperglucemia, normoglucemia e hipoglucemia. Como se busca una monitorización continua esta se realiza con una cadencia igual al paso del tiempo para de esta forma, si el usuario pasa poco tiempo utilizando la aplicación poder haber obtenido todos los valores estadísticos posibles. Además, en este hilo de ejecución se realiza la comprobación de si los valores de glucosa leídos están en los márgenes correctos, para que en el momento en el que se produzca una hipoglucemia o una hiperglucemia se desencadene una serie de avisos y alarmas haciendo saber al usuario que algo va mal y hay que actuar. Esta alarma consiste en el lanzamiento de una animación por pantalla con forma de triángulo de peligro que sigue un efecto de empequeñecimiento y agrandamiento. Simultáneamente se le envía al robot la orden de entrar en modo de alarma, el cual se detallará más adelante en el apartado 5.6.11. En cuanto al manejo de la alarma en la aplicación esta se desactiva automáticamente cuando se vuelven a estabilizar los valores de glucosa y se lea uno que derive en una normoglucemia. En el caso de querer detener la alarma antes de tiempo, el usuario tiene la opción de mediante la pulsación prolongada en la animación que se está mostrando lanzar un diálogo donde se da la posibilidad, en función de la opción que se elija, de posponer la alarma durante 20 segundos, 45 segundos o desactivarla definitivamente. En cualquier caso, por seguridad este veto a se levanta cada vez que se produzca la transición a otra pantalla de la aplicación. La única forma de desactivarla por completo es desactivando la monitorización continua mediante el "Toggle Button" detallado anteriormente.

Lo siguiente que llama la atención son los botones representados por diferentes iconos situados en la parte inferior de la pantalla. Estos iconos se encuentran embebidos en lo que se conoce como un "Scroll View", lo cual consiste en un "contenedor" de elementos con la peculiaridad de que, mediante un gesto táctil de deslizamiento, permite el desplazamiento entre los diferentes iconos haciendo aparecer en pantalla unos u otros. Esta funcionalidad nos permite incluir en la actividad todos los iconos de los diferentes modos manteniendo un tamaño agradable y funcional. Es notable que todos los iconos excepto uno tienen un aspecto de semitransparencia que indica inequívocamente que están deshabilitados. Esta semitransparencia, así como la composición de las diferentes imágenes han sido creados mediante el software libre de manipulación de imágenes GIMP. Los botones de selección de modo se encuentran deshabilitados pues aún no se ha establecido la conexión bluetooth entre el robot y la aplicación, la cual es necesaria pues estos modos hacen uso de la interfaz que proporciona el dispositivo robótico. En cambio, hay un icono que no está deshabilitado y este es el que lanza la pantalla de estadísticas la cual será detallada más adelante en el apartado 4.11.9. Esto se debe a que este modo no precisa del uso del robot permitiendo hacer la aplicación independiente de este en todo lo relativo a la monitorización continua (ver Figura 8 y Figura 9). (24)

Por otro lado, en la parte superior central de la pantalla se encuentra un texto que indica cual es el usuario que actualmente tiene la sesión activa en la aplicación. Esto es muy importante pues saber que usuario esta activo en el dispositivo permite tanto leer como actualizar los valores correctos en la base de datos. Esto se consigue en primer lugar obteniendo el código de

identificación del dispositivo móvil, el cual es único para cada teléfono. Luego en la base de datos, asociado a ese código, establecemos desde la pantalla anterior de inicio de sesión, el usuario el cual se ha verificado en su cuenta. Pudiendo de esta forma simplemente leyendo ese campo asociado al código de identificación saber en ese dispositivo que cuenta de usuario está activa. Esto permite tanto la posibilidad de iniciar sesión desde diferentes cuentas en un mismo dispositivo como poder manejar múltiples sesiones activas de diferentes usuarios en la base de datos. Junto a este texto se encuentra un botón representado con una imagen indicativa de "apagar" el cual tras su pulsación cerrará la sesión activa en ese momento tanto en la aplicación como en la base de datos y poder así lanzar la pantalla de inicio de sesión para dar la posibilidad al usuario de poder entrar o registrarse desde una cuenta diferente (ver Figura 8).

En la esquina superior izquierda se encuentra un botón representado mediante un icono indicativo de una sección de estadísticas. Este botón tampoco está deshabilitado pues su uso no implica la utilización del robot. Al pulsar sobre él, la aplicación se traslada a la pantalla de estadísticas con acceso remoto (ver apartado 4.11.12).

Por último, en la esquina superior izquierda se observa un botón en color rojo con el logo de la conexión Bluetooth. Al hacer clic sobre este, la aplicación establece la conexión Bluetooth con el dispositivo robótico, evidentemente previamente encendido, y le manda la orden de entrar en el modo de selección de modo (ver apartado 5.6.1). Así mismo se habilitan los demás botones embebidos en el "Scroll View" tanto funcionalmente como en apariencia ya que ahora no presentan la semitransparencia anterior. Además de todo esto para hacer claramente visible al usuario que la conexión se ha establecido correctamente aparece un texto informativo indicando que el proceso ha resultado con éxito, así como el botón de conexión que hemos pulsado anteriormente se vuelve verde como símbolo inequívoco de que la conexión bluetooth ha sido establecida. En caso de no poder realizarse dicha conexión por cualquier motivo aparecerá por pantalla un "Toast" haciéndoselo saber al usuario (ver Figura 9).

Una vez la conexión aplicación-robot se ha configurado, se está en disposición de mediante los diferentes iconos situados en la parte inferior de la pantalla ir accediendo hacia los distintos modos con los que cuenta la aplicación.

Figura 8. Pantalla de selección de modo con conexión Bluetooth desactivada

Fuente: Captura de pantalla de la aplicación

Figura 9. Pantalla de selección de modo con conexión Bluetooth activada

Fuente: Captura de pantalla de la aplicación

4.11.3. MODO DE CONDUCCIÓN LIBRE

Al hacer clic sobre el primer botón representado mediante un icono de un volante y un velocímetro se lanza la pantalla que se va a describir a continuación.

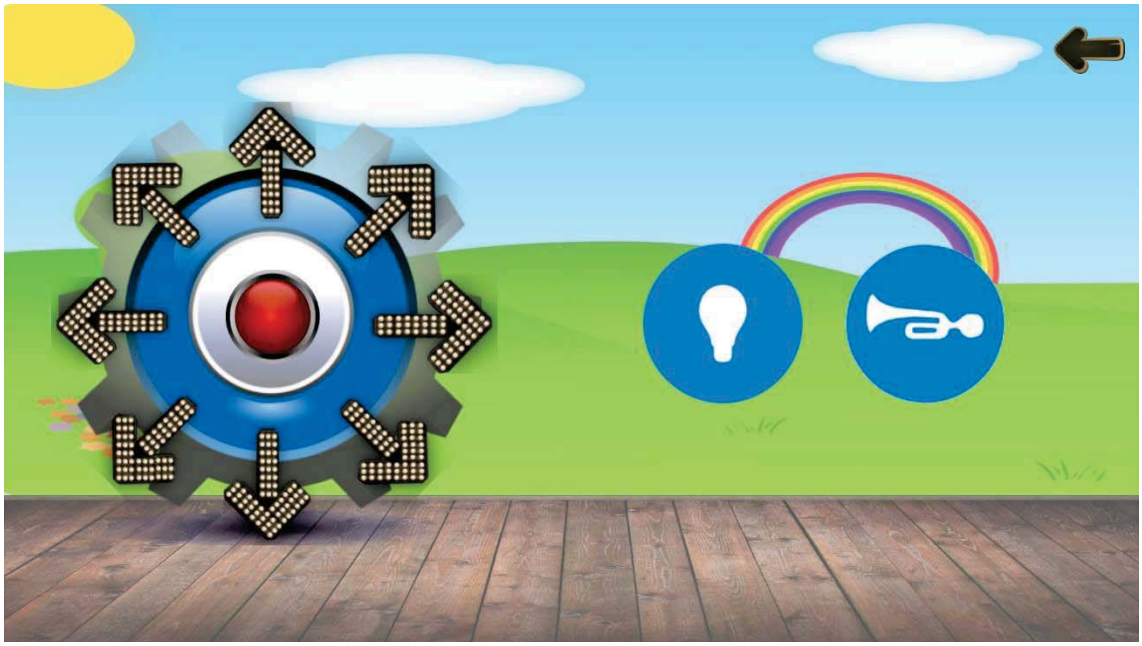
En primer lugar, llama la atención un gran "joystick" con ocho flechas sobreimpresionadas en un patrón circular sobre él. Estas flechas son las diferentes direcciones que el robot puede tomar. Desde el punto de vista del código este "joystick" ha sido implementado mediante lo que se conoce como "Drag and Drop Listener" en conjunción con un "Shadow Builder". Al pulsar sobre el botón central del "joystick" gracias al constructor se crea la sombra de este botón la cual puede ser arrastrada por la pantalla. Aquí entran en juego los "listeners" anteriormente mencionados, ya que, si esa sombra que se arrastra entra en alguno de ellos, colocados en cada una de las flechas, se ejecuta el fragmento de código relativo al envío de la instrucción que el robot debe interpretar como la dirección en la que debe moverse. Una vez se deja de arrastrar dicho botón central, este vuelve a su sitio y se envía al robot la orden de detenerse y entrar de nuevo en modo de escucha. Aparte de esto la imagen del botón central tiene implementado un "On Touch Listener" que se encarga de cuando se pulse sobre ella lanzar el fragmento de código que en este caso hace invisible dicho botón y crea la sombra para dar la sensación de que se puede arrastrar por la pantalla. De la misma forma cuando cesa dicho arrastre la sombra se destruye y la imagen vuelve a hacerse visible haciendo la ilusión de que el botón vuelve a su punto de partida. [25]

Por otro lado, a la derecha de dicho "joystick" se sitúan dos botones, uno que representa una bocina. La cual al pulsarse envía al robot la orden de hacer sonar un sonido con el zumbador. Y el otro botón hace referencia a las luces o faros del robot, ya que cuando se acciona se le envía al este la acción de encender los leds interiores de los que dispone. Si se vuelve a pulsar dicho botón se enviará la orden de apagar dichos leds y así sucesivamente (ver Figura 10).

Por otra parte, tanto esta pantalla como todas las demás tienen implementado el mismo sistema de alarma de los niveles de glucosa detallado en el apartado anterior, pero con unas ligeras modificaciones. La primera de ellas es que tanto la lectura de datos como la supervisión de que los valores leídos están dentro de márgenes aceptables se realiza en los hilos abiertos en la pantalla de selección de modo, y estos hilos no son accesibles desde otras actividades. Es por lo que se hace necesario el uso de variables globales en la aplicación, tal como se ha detallado anteriormente en el apartado 4.9. De esta forma cuando un valor de glucosa leído se encuentre fuera de los márgenes establecidos se actualizará una determinada variable global, la cuál es leída continuamente mediante un "Thread" con una cadencia de un segundo, local a cada actividad abierta. De modo cuando dicha variable se encuentre a nivel alto se desencadenará la misma situación de alerta descrita en el apartado anterior. De manera análoga cuando se desactiva dicha alarma se vuelve a poner la variable global a cero.

Por último, en la esquina superior derecha se encuentra un botón representado por una flecha que al hacer clic sobre ella cierra el hilo de comprobación de la variable de alarma, así como desencadena la transición de la conexión bluetooth y la actualización de la variable global de "primera vez", ambas explicadas en el apartado 4.10. Tras este proceso se vuelve a lanzar la actividad de selección de modo.

Figura 10. Pantalla de conducción libre



Fuente: Captura de pantalla de la aplicación

4.11.4. MODO DE SIMULACIÓN DE ESTADOS DE GLUCOSA

Este modo se compone de tres pantallas o actividades bien diferenciadas pero muy relacionadas entre sí, las cuales se detallarán a continuación:

4.11.4.1. Actividad lotería de raciones de carbohidratos

Esta es la primera pantalla que se lanza cuando se pulsa el botón en la pantalla principal. En ella se muestra un gran dado sobre un tapete verde y un "Toast" que invita a agitar el teléfono para que de manera aleatoria se muestre una cara del dado relativa al número de raciones que se van a establecer en el simulador (ver Figura 11). La captación de esta agitación se consigue haciendo uso del acelerómetro que incorpora el propio dispositivo móvil. El código que hace posible esto es en primer lugar la inicialización de las variables de aceleración a la aceleración de la gravedad (9.80665 m/s^2) para así independizarlas de esta magnitud presente en todo momento, luego mediante un "listener", que se activa cuando se produce una variación en los valores recogidos por el acelerómetro en sus tres ejes, se registran dichos datos en un vector tridimensional. Seguidamente, se realiza el módulo de dicho vector y se obtiene la aceleración en ese momento. A continuación, se realiza la resta de dicha aceleración respecto a la aceleración anterior tomada y si la diferencia es superior a un valor asignado se establece que se ha producido la agitación y se desencadena el código correspondiente. Por último, se realiza la actualización de la variable aceleración anterior a la aceleración leída en ese momento. [(26)]

Por código solo se permite que la agitación del teléfono tenga efecto una sola vez, para evitar que el usuario pueda estar indefinidamente probando hasta conseguir las raciones de carbohidratos que le interesara introducir en el simulador, perdiéndose así el efecto de evaluación y aprendizaje que se busca con este modo.

Una vez, aleatoriamente, se establece el número de raciones, se envía al robot la orden de mostrar mediante su matriz de leds y con el efecto de "scroll" el mensaje de que tantas raciones equivalen a tantos gramos de carbohidratos, siguiendo la ratio 1/10 tal como se establece normalmente. Tras concluir dicho mensaje aparece en pantalla un botón que al pulsarse inicia la actividad de calculadora de insulina.

Figura 11. Pantalla lotería de raciones

Fuente: Captura de pantalla de la aplicación

4.11.4.2. Actividad calculadora de insulina

Esta actividad junto con otras dos forma parte de un "ViewPager" que es una librería propia de Android que mediante el uso de "Fragments" (cada una de las actividades que componen un "ViewPager") permite una transición entre pantallas con un simple deslizamiento táctil sobre esta. De esta forma se integra de una forma más agradable y eficaz diversas pantallas que guardan una fuerte relación entre ellas. [(27)]

Nada más abrirse la actividad un texto invita a introducir la ratio de bolo de insulina por gramos de carbohidratos personal del usuario. Esta introducción se realiza mediante unas flechas hacia arriba y abajo que aumentan o disminuyen el divisor o el dividendo una unidad en cada caso. Una vez seleccionado la ratio al pulsar el botón de establecer aparecerá un texto mostrando la ratio seleccionada, y cambiará la zona de selección a una donde por el mismo sistema se podrá configurar tanto la parte entera como decimal de las unidades de insulina que el usuario cree necesarias para las raciones de comidas que le han tocado en la pantalla anterior. Al mismo tiempo otro texto le recordará de cuantos gramos de carbohidratos es la comida que va a ingerir. Del mismo modo al volver a pulsar en el botón establecer se mostrará por pantalla un texto indicando los parámetros seleccionados, también aparecerá un botón de modificar que permitirá volver a introducir los parámetros, simultáneamente se ejecutará el código del simulador con las especificaciones introducidas por el usuario (ver Figura 12).

Este código fue proporcionado por los tutores del proyecto en lenguaje C++, y tras "traducirlo" a lenguaje Java e integrarlo en la aplicación, es capaz de proporcionar mediante una serie de ecuaciones diferenciales la respuesta de la glucosa en sangre durante un intervalo de tiempo a partir de una entrada de gramos de carbohidratos y bolo de insulina. Todo esto será ampliado a continuación en el apartado dedicado al simulador (4.11.4.4). Una vez se ha ejecutado el simulador y deslizándose hacia la derecha aparece una pantalla donde se puede ver en una gráfica, implementada mediante la librería "MPAndroidChart" (ver apartado 4.6), la respuesta de la glucosa durante 6 horas que es para lo que está programado el simulador (ver Figura 13).

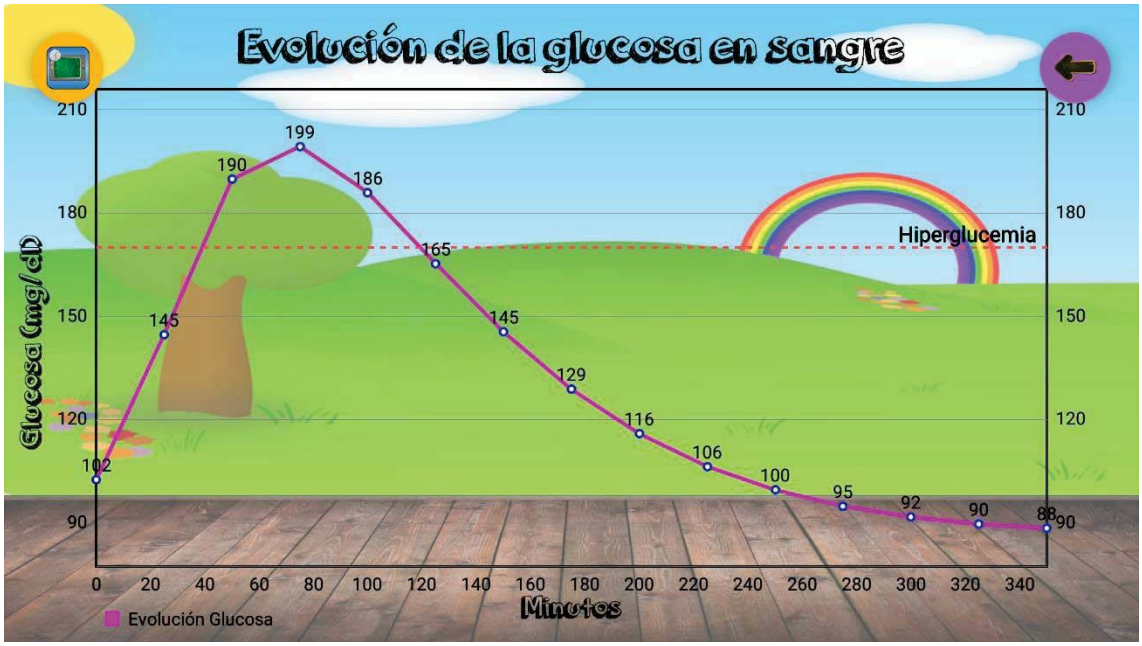
Deslizándose nuevamente hacia la derecha da lugar a la última pantalla del "ViewPager". La cual muestra todas las estadísticas de la simulación, tanto los porcentajes de tiempo como los minutos que ha pasado en cada estado de glucemia (hiperglucemia severa, hiper glucemia normal, normoglucemia, hipoglucemia e hipoglucemia severa). Además de estos datos se da información al usuario sobre los valores máximos y mínimos alcanzados y la media de los valores obtenida. A partir de estos datos, los cuales dependen directamente de lo bien o mal que el usuario haya estimado el bolo de insulina necesario, este es recompensado con un número determinado de estrellas en función de las estadísticas obtenidas. Estas estrellas permitirán desbloquear funcionalidades en la aplicación como se verá más adelante en el apartado 4.11.6.

Cabe destacar que en el "ViewPager" y por consiguiente en todos los "Fragments" que lo componen se han introducido dos "Floating Buttons". Uno de ellos hace que la aplicación regrese a la pantalla principal de selección de modo de la misma forma que se ha mostrado anteriormente. Y el otro situado en la esquina superior derecha representado por un osciloscopio lanza la siguiente actividad de simulación interactiva.

Figura 12. "ViewPager" calculadora de insulina (Fragment 1)

Fuente: Captura de pantalla de la aplicación

Figura 13. "ViewPager" calculadora de insulina (Fragment 2)



Fuente: Captura de pantalla de la aplicación

4.11.4.3. Actividad de simulación interactiva

En esta pantalla, previamente habiendo recibido mediante variables globales el vector de estados de glucosa obtenido en la actividad anterior, se comienza a dibujar una gráfica donde cada segundo se añade un nuevo valor. Este intervalo de tiempo corresponde en realidad a 5 minutos en la escala de tiempos del simulador, por lo tanto, tras 72 segundos de simulación interactiva se completarán las 6 horas de simulación. A la misma vez que se va dibujando la gráfica se van mostrando a tiempo real las mismas estadísticas que en la pantalla anteriormente descrita pudiendo de esta forma llevar un control paso a paso de cómo se comportará la glucosa en el individuo durante las 6 horas para las que se ha simulado.

A parte de todo esto meramente estadístico se encuentra implementado lo que hace especial a esta actividad y es su integración con el dispositivo robótico el cual hace de interfaz de lo que está ocurriendo. De esta forma en función del estado de glucemia en el que se encuentre el individuo según la simulación, así como si mantiene una tendencia ascendente o descendente el robot mostrará y ejemplificará mediante una serie de acciones todas estas variaciones. Tanto las acciones como toda la interfaz que muestra el robot, en relación con esta simulación, serán tratados con más detalle en el apartado 5.6.3.

Una vez finalizada la simulación interactiva, o en cualquier momento, el usuario puede detenerla y volver a la pantalla principal de selección de modo manteniendo el dedo pulsado sobre la gráfica y haciendo clic sobre la flecha que aparecerá en pantalla.

4.11.4.4. Simulador de glucosa

El simulador consiste en la modelización de un subsistema de la glucosa que recoge las fases de absorción, distribución y eliminación. Equivalentemente se modela el subsistema de la insulina. Además, el modelo tiene en cuenta la acción que la insulina produce en el transporte de la glucosa, así como en su eliminación como producción endógena. [28]

El modelo se fundamenta en dos ecuaciones diferenciales que representan la cinética de la glucosa en el cuerpo humano: [28]

Ecuación 1. Modelo de la cinética de la glucosa

$$\frac{dQ_1(t)}{dt} = -\left[\frac{F_{01}^c}{V_G G(t)} + x_1(t)\right] Q_1(t) + k_{12} Q_2(t) - F_R + U_G(t) + EGP_0[1 - x_3(t)]$$

$$\frac{dQ_1(t)}{dt} = x_1(t) Q_1(t) - [k_{12} + x_2(t)] Q_2(t) y(t) G(t) = Q_1(t)/V_G$$

Fuente: Bibliografía [28]

Q_1 y Q_2 representa, respectivamente, las masas de glucosa en la zona de medida (accesible) y en la zona donde no se realiza medida alguna (no accesible). K_{12} representa la tasa de transferencia constante entre la zona no accesible y la accesible, V_G simboliza el volumen distribuido de la zona accesible y G es la concentración de glucosa medible. EGP_0 representa la producción endógena de insulina extrapolada al cero de la concentración de insulina. F^c_{01} es el flujo de glucosa no dependiente de la insulina. Y F_R representa a la glucosa renal. [28]

La absorción de la insulina se modela de la siguiente forma: [28]

Ecuación 2. Modelo de la absorción de la insulina

$$\frac{dS_1(t)}{dt} = u(t) - \frac{S_1(t)}{t_{max,I}} \qquad \frac{dS_2(t)}{dt} = \frac{S_1(t)}{t_{max,I}} - \frac{S_2(t)}{t_{max,I}}$$

Fuente: Bibliografía [28]

S_1 y S_2 representan la absorción de la insulina de corta acción administrada de forma subcutánea, u(t) modela la administración de la insulina (bolo e infusión). Y $t_{max,I}$ es el tiempo para que se produzca la absorción completa de la insulina. [28]

La concentración de insulina en plasma se modela: [28]

Ecuación 3. Modelo de la concentración de la insulina

$$\frac{dI(t)}{dt} = \frac{U_1(t)}{V_1} - k_e I(t)$$

Fuente: Bibliografía [28]

Donde K_e es la tasa de eliminación fraccional y V_I es el volumen de distribución. Por su parte U_I se define como $U_I = S_2(t)/ t_{max,I}$. [28]

El modelo define tres acciones de la insulina sobre la cinética de la glucosa: [28]

Ecuación 4. Acciones de la insulina sobre la cinética de la glucosa

$$\frac{dx_1}{dt} = -k_{a1}x_1(t) + k_{b1}I(t) \qquad \frac{dx_2}{dt} = -k_{a2}x_2(t) + k_{b2}I(t) \qquad \frac{dx_3}{dt} = -k_{a3}x_3(t) + k_{b3}I(t)$$

Fuente: Bibliografía [28]

X_1 y X_2 representa el efecto de la insulina sobre el transporte de la glucosa, su eliminación y sobre la creación endógena. El factor K_{ai} con i= 1,2,3, representan tasas constantes de desactivación, así como K_{bi} representa las tasas de activación constantes. [28]

La implementación de estas ecuaciones en un código ejecutable requiere la utilización de un algoritmo de integración numérica. El utilizado en este simulador ha sido el método de Dormand-Prince. Este método es una adaptación del conocido método de Runge-Kutta. El algoritmo de Dormand-Prince mejora la exactitud con respecto a su algoritmo de base, se categoriza como un método de paso adaptativo. El algoritmo se basa en evaluar en cada paso la ecuación: [29]

Ecuación 5. Ecuación algoritmo Dormand-Prince quinto orden

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i k_i$$

Fuente: Bibliografía [29]

Donde:

Ecuación 6. Parámetros ecuación algoritmo Dormand-Prince

$$k_1 = f(t_n\,, y_n),$$
$$k_2 = f(t_n + \ c_2 h\,, y_n + h(a_{21}k_1))\,,$$
$$k_3 = f(t_n + \ c_3 h\,, y_n + h(a_{31}k_1 + a_{32}k_2))\,,$$
$$\vdots$$
$$k_s = f\left(t_n + \ c_s h\,, y_n + h\big(a_{s1}k_1 + a_{s2}k_2 + \cdots + a_{s,s-1}k_{s-1}\big)\right).$$

Fuente: Bibliografía [29]

y los parámetros b_i, c_i, a_{ij}, se obtienen de la tabla de Butcher correspondiente a este método. [29]

A continuación, se evalua la ecuación:

Ecuación 7. Ecuación algoritmo Dormand-Prince cuarto orden

$$y^*_{n+1} = y_n + h\sum_{i=1}^{s} b^*_i k_i$$

Fuente: Bibliografía [29]

Donde b_i corresponde al método de Runge-Kutta de quinto orden mientras que los términos b^*_i son del de orden cuatro. [29]

Por último, se calcula el error como la diferencia entre ambas ecuaciones: [29]

Ecuación 8. Error algoritmo Dormand-Prince

$$e_{n+1} = y_{n+1} - y^*_{n+1} = h\sum_{i=1}^{s}(b_i - b^*_i)k_i$$

Fuente: Bibliografía [29]

y se define el paso utilizado como: [29]

Ecuación 9. Paso algoritmo Dormand-Prince

$$s = \left(\frac{\varepsilon h}{2|y_{n+1} - y^*_{n+1}|}\right)^{\frac{1}{5}}$$

Fuente: Bibliografía [29]

Donde épsilon hace referencia a la tolerancia utilizada y h es el intervalo de tiempo anterior. [29]

Para más información acerca de cómo se ha implementado este simulador en la aplicación se encontrar en el Anexo I el código escrito en lenguaje JAVA.

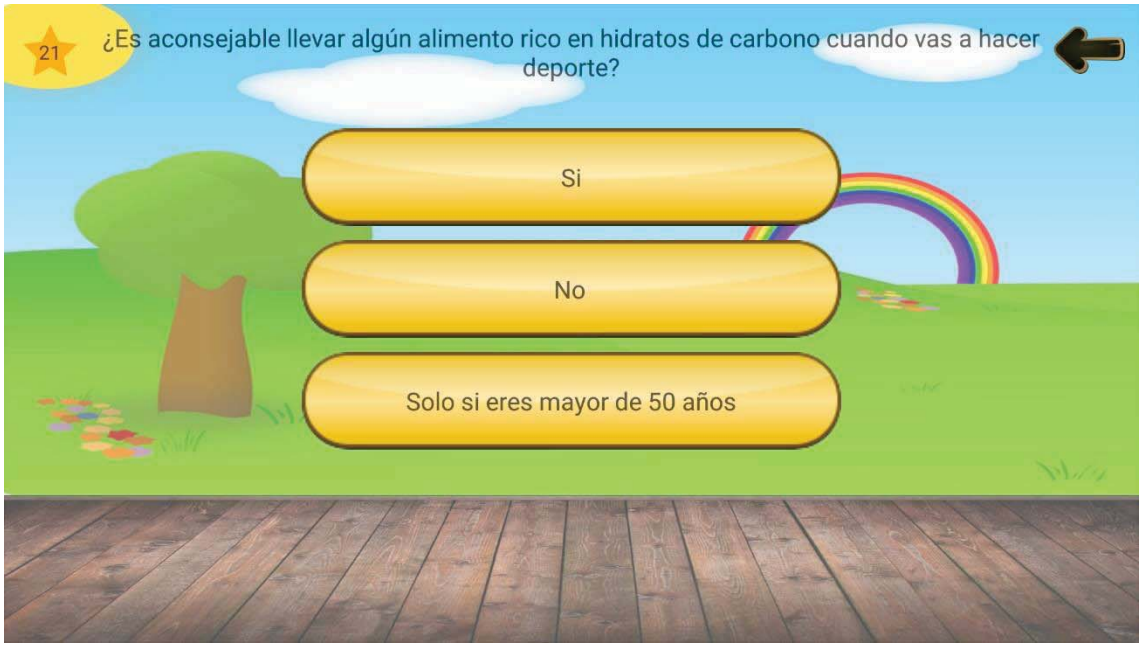
4.11.5. MODO PREGUNTAS

Tras pulsar en el tercer botón de la pantalla principal la aplicación se traslada a la pantalla de preguntas. En ella al usuario se le hará una pregunta sobre la diabetes y su gestión y se le darán tres opciones de las cuales el deberá elegir pulsando sobre una de ellas. En caso de acertar aparecerá una animación de un "tick" haciéndole saber que ha acertado, así como un texto con una breve explicación sobre la respuesta. Además de esto se sumará una estrella a su progreso lo cual quedará evidenciado en el marcador de estrella situado en la esquina superior izquierda. Este marcador se animará cada vez que se obtenga una recompensa. En caso de que el usuario seleccione la opción incorrecta saltará una animación en forma de equis roja, así como la misma explicación de la respuesta como en el caso de acertar. Lógicamente, en el caso de respuesta errónea el usuario no es recompensado con ninguna estrella.

Como novedad esta actividad cuenta con un sistema de control de tiempos haciendo que entre que se muestre una pregunta y la siguiente deban pasar al menos 24 horas. Esto es bastante importante tanto para motivar al usuario a utilizar la aplicación a diario, así como, para tener la posibilidad, de con una batería de preguntas no excesivamente grande, poder abastecer el modo durante bastante tiempo. Este control del tiempo se consigue mediante el uso conjunto de la clase "Date" junto a la base de datos tal como se ha explicado anteriormente en el apartado "control de tiempo y fechas" (4.8).

Además de la flecha para volver a la pantalla principal como todas las demás actividades, este modo en concreto tiene la posibilidad de acceder al modo de recompensas haciendo clic en el icono de la estrella donde se encuentra el marcador de estas.

Figura 14. Pantalla de preguntas



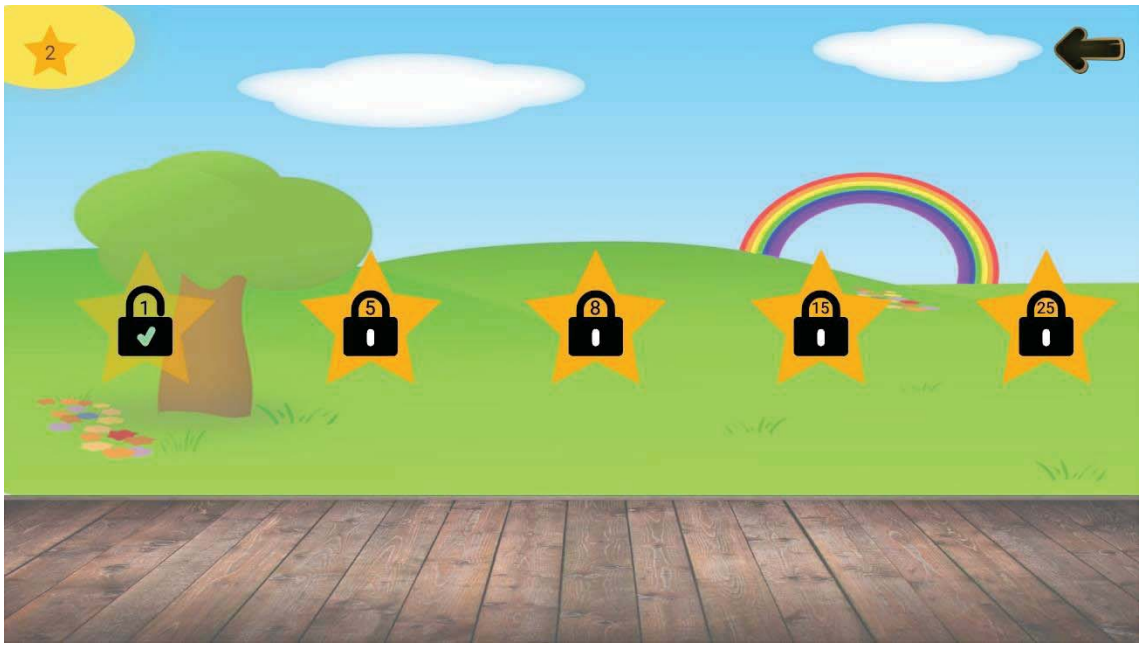
Fuente: Captura de pantalla de la aplicación

4.11.6. Modo recompensas

En este modo el usuario puede, en función de las estrellas que tenga acumuladas, desbloquear diferentes funcionalidades pulsando sobre las estrellas de mayor tamaño que aparecen en la pantalla. Cada estrella está bloqueada por un candado y un número que indica la cantidad de estrellas necesarias para ser desbloqueada. Cuando el usuario pulsa sobre una de estas estrellas si no tiene suficiente para desbloquearla se mostrará un "Toast" advirtiéndole de ello y de cuantas estrellas le faltan para lograrlo. En el caso de tener suficientes estrellas y pulsar sobre una de las grandes se reproducirá la animación de apertura del candado, así como, la estrella cambiará a un estado de apariencia semitransparente haciendo indicativo que ya ha sido desbloqueada. Simultáneamente a todo lo anterior un "Toast" será mostrado indicando al usuario que funcionalidad de la aplicación acaba de desbloquear al mismo tiempo que son actualizadas las variables de mejoras desbloqueadas en la base de datos. Esto es importante para que no solo la próxima vez que entre en esta pantalla se muestren las estrellas correspondientes ya desbloqueadas y las funcionalidades sean patentes en la aplicación, sino para que sí el usuario cambia de dispositivo, simplemente iniciando sesión con su cuenta pueda continuar con todo su progreso por donde lo dejó (ver Figura 15).

Entre algunas de las funcionalidades que se presentan están las de cambiar la velocidad del robot en el modo de conducción libre, así como que los leds que representen los faros se enciendan con diferentes colores. El motivo de todo este sistema de logros y recompensas busca que el usuario, en este caso un niño, desee jugar con la aplicación y entrar en modos educativos que de otro modo no tendría ningún interés en entrar.

Figura 15. Pantalla de recompensas



Fuente: Captura de pantalla de la aplicación

En la Figura 15 se puede observar un instante de la animación automática que se produce al entrar a la pantalla en la estrella de valor la unidad ya que se tienen suficientes estrellas y anteriormente se ha realizado su desbloqueo.

4.11.7. Modo de simulación continua

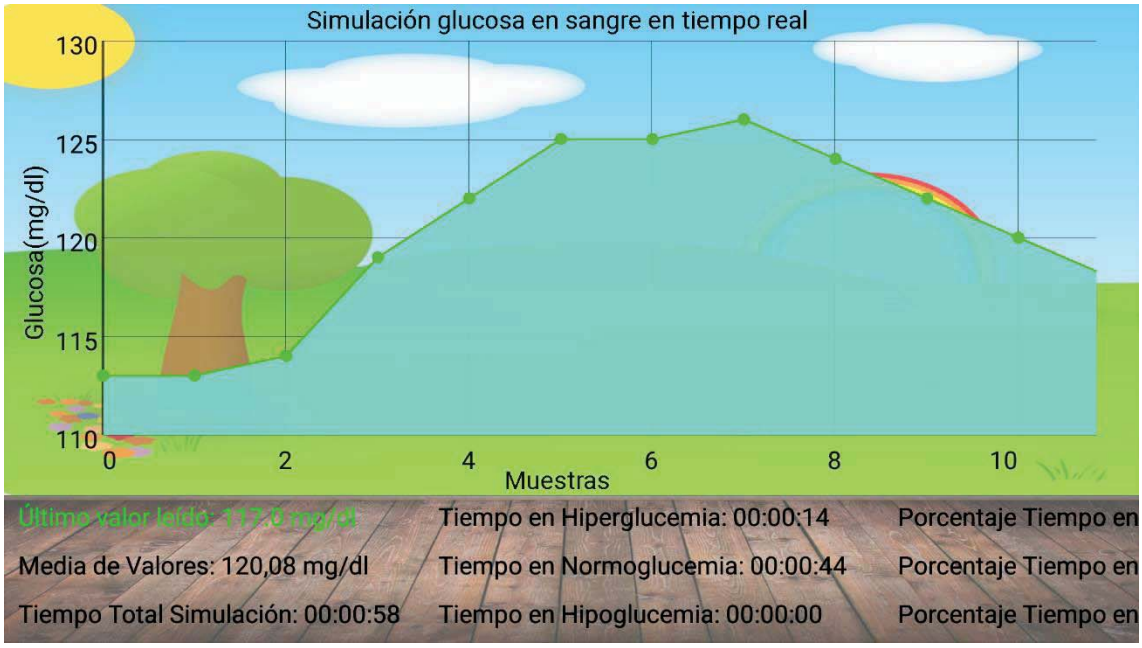
Este modo al igual que el modo de simulación interactiva dibuja una gráfica a tiempo real mediante la librería "GraphView" tal como se ha explicado en el apartado 4.6. A diferencia del otro modo este no representa los valores de glucosa procedentes de una simulación, sino que crea la gráfica a partir de los valores almacenados en la base de datos relativos a estudios sobre pacientes reales. La creación de esta gráfica a tiempo real se implementa con el mismo sistema de doble "Thread" utilizado en la pantalla principal de selección de modo (ver apartado 4.11.2). Al ser a tiempo real permite ver al usuario la evolución de su glucosa de una forma muy intuitiva. Además, embebidos en un "Scroll View" se encuentran todas las estadísticas relativas a la simulación que se van actualizando a tiempo real (ver Figura 16).

Para no perder la monitorización común a toda la aplicación, la simulación continua tomo como valor de partida el último valor leído por el "thread" de la pantalla principal, el cual se termina al iniciar este modo. Este se consigue mediante la actualización continua de una variable global en dicho "thread" así como la lectura de ese valor al iniciar este modo de simulación a tiempo real. Del mismo modo cuando se sale de este modo y se vuelve a la pantalla principal mediante la flecha que aparece al mantener pulsado sobre la gráfica se actualizará dicha variable global para que la monitorización pueda continuar por el punto donde se quedó.

Al igual que todas las demás actividades este modo tiene implementado el sistema de alarma con la peculiaridad de que a diferencia de los demás localmente se produce la lectura de datos y la supervisión de que se encuentran en los márgenes establecidos. De esta forma el lanzamiento del sistema de alarma se realiza directamente en esta actividad sin necesidad de

pasar por la lectura y actualización de una variable global. De esta forma cuando el usuario entre en este modo la aplicación aprovecha para reiniciar los hilos de ejecución encargados de la monitorización minimizando los riesgos de saturación de estos.

Figura 16. Pantalla de simulación continua



Fuente: Captura de pantalla de la aplicación

4.11.8. Modo juego de memoria

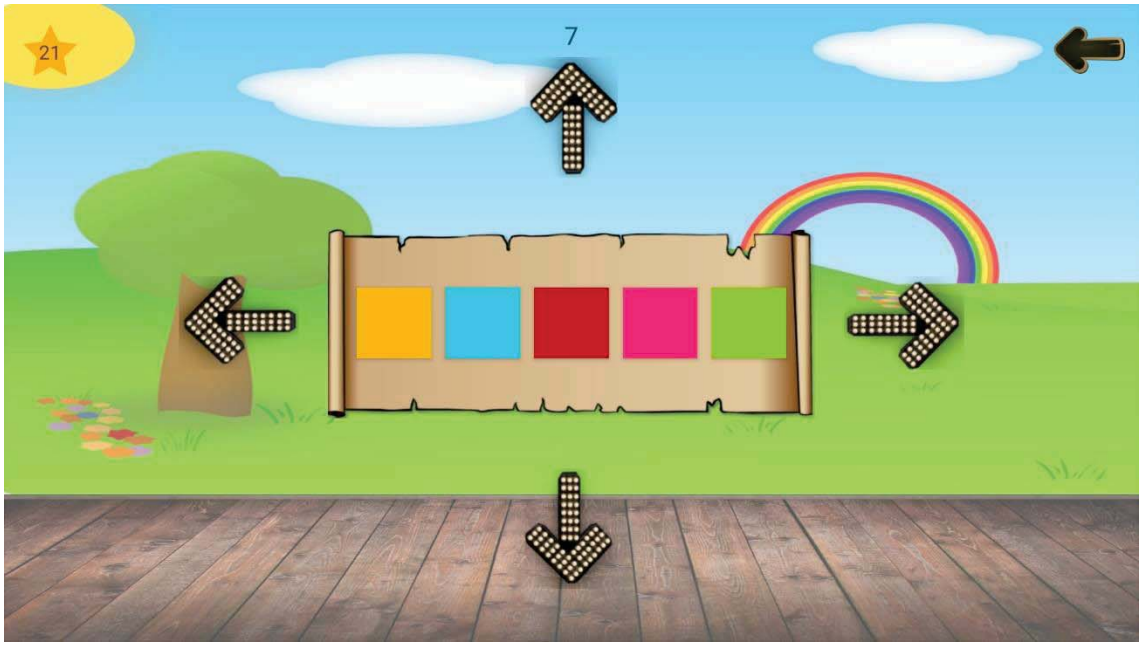
A este modo se llega mediante la pulsación del quinto icono del "Scroll View" de la pantalla principal representado por unos dados y unos "gamepads" (ver Figura 9). Este gesto conduce hasta una pantalla protagonizada por un gran botón que invita al usuario a pulsarlo para comenzar el juego. Tras esta pulsación desparece dicho botón y se muestra en pantalla una cuenta atrás de tres segundos. Seguidamente aparecen en pantalla los diferentes controles del juego y se inicia este. (ver Figura 17)

El juego consiste en primero, aleatoriamente, se establece en el intervalo de 3 a 15 el número de acciones que el robot va a llevar a cabo. A continuación, el robot ejecuta la serie de acciones aleatorias marcadas por el juego consistente en movimientos y diferentes colores mostrados a través de sus leds internos. Una vez finaliza la secuencia el robot hace sonar su zumbador indicando al usuario que ha terminado. En este momento el jugador debe replicar las acciones en el mismo orden pulsando en los botones correctos en la pantalla. Si va acertando se mostrará una animación haciéndole saber que lo está haciendo bien. En caso contrario, una animación le indicará que ha fallado y el juego habrá finalizado. En este momento se activa un texto parpadeante en la parte inferior de la pantalla indicando al usuario que pulse sobre la pantalla para volver a la pantalla principal. En cambio, si el jugador es capaz de completar la secuencia sin fallar un texto se lo hará saber, así como, de nuevo un texto parpadeante le invitará a pulsar para continuar. Cuando el jugador, tras completar la secuencia, pulsa sobre el botón invisible pero habilitado que inunda la pantalla el juego pasa a una nueva actividad llamada "La Lotería de las Estrellas".

En esta pantalla el usuario tiene la oportunidad de ganar estrellas con las que desbloquear mejoras en la aplicación como se ha detallado anteriormente. Para conseguirlo en primer lugar se le pide al usuario que elija dos números del 2 al 12 mediante unas flechas que aumentan y disminuyen el número en una unidad. Una vez el jugador pulsa sobre el botón de confirmar y establece sus apuestas aparece en pantalla dos dados y un "toast" que le dice al usuario que agite el teléfono para lanzar los dados. Esta acción solo se podrá realizar una vez.

Una vez, de forma aleatoria, los dados muestran una de sus caras el sistema evalúa si alguno de los números establecidos por el usuario corresponde con la suma de ambas caras de los dados. En el caso de que el jugador resulte afortunado un texto le dará la enhorabuena y se sumará una estrella a su marcador. En caso contrario, se informará nuevamente de que no ha resultado premiado. En ambos casos se visibilizará un texto parpadeante y un botón invisible que ocupa toda la pantalla, para cuando el usuario pulse volver a la pantalla de selección de modo.

Figura 17. Pantalla juego de memoria



Fuente: Captura de pantalla de la aplicación

4.11.9. MODO ESTADÍSTICAS

Este es el único modo al que se puede acceder sin previamente haber establecido la conexión bluetooth entre la aplicación y el robot. Tras pulsar sobre el icono representado por una serie de gráficos estadísticos, la aplicación abre una pantalla de transición con varias animaciones que hacen un poco más amena la espera de 5 segundos que dura esta. El motivo de la inclusión de esta pantalla de transición se debe a que al finalizar esta pantalla se abre un "ViewPager" con 5 "Fragments", donde en cada uno de ellos se representa un gráfico distinto. La peculiaridad del "ViewPager" es que ejecuta el código relativo a todos los "Fragments" en el momento que se inicia la actividad que los contiene. Por ello si se intentase hacer la lectura de todas las variables estadísticas almacenadas en Firebase en la misma actividad donde se crean los gráficos, estos en primera instancia aparecerían vacíos sin ninguna información, siendo necesario que el usuario pulsara en algún botón para recargarlos y volver a crearlos con la nueva información

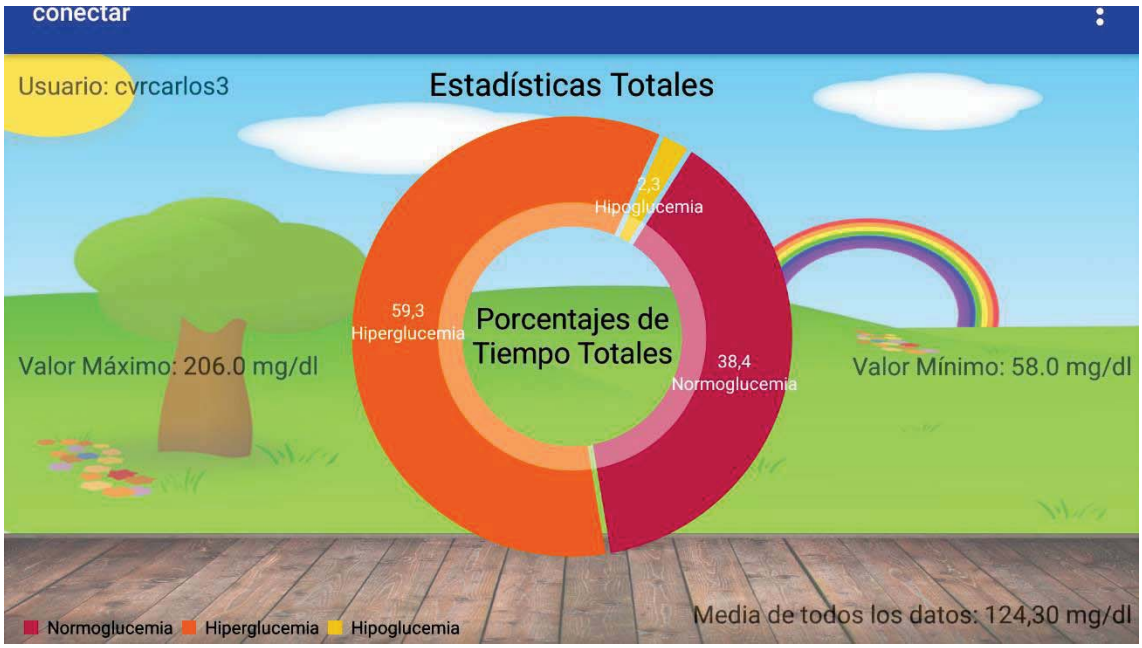
leída, suponiendo una duplicidad en todo el proceso. Para evitar esto y posibles problemas a la hora de la visualización de gráficos sin datos, se ha optado por la solución de incluir una breve pantalla de transición, donde se lean todos los datos necesarios y después mediante variables globales se ceda toda la información a la actividad coordinadora de los "Fragments". Este método se hace posible gracias a que la lectura de variables globales internas de la aplicación es significativamente más rápida que la lectura de valores desde la base de datos, haciendo posible que la actividad recabe toda la información antes de crearse los gráficos. De esta forma cuando la aplicación finalice la pantalla de transición y abra el "ViewPager" el usuario tendrá la posibilidad de pasar rápidamente de gráfico en gráfico sin esperas estando todos ellos creados de antemano.

Como se ha comentado anteriormente en esta actividad se representan 5 gráficos repartidos en 5 "Fragments" distintos. El primero de ellos, el cual se ve nada más iniciarse la actividad, es el relativo a las estadísticas del día actual. En este "Fragment" se encuentra un gráfico sectorial donde se representan los porcentajes de tiempo que el usuario ha estado en cada estado de glucemia. También en esta parte del "ViewPager" se muestra tanto el valor máximo como el mínimo alcanzado, como la media de los valores monitorizados. Además de esto, se informa al usuario mediante otro texto cual es la fecha de la cual se está representando los datos, la cual corresponderá con la actual, así como de que usuario son los datos aquí mostrados. En este caso evidentemente corresponderán al usuario que ha iniciado sesión en la aplicación, pero como se verá más adelante en el modo "Estadísticas con Acceso Remoto" (ver apartado 4.11.12) esto no siempre será así. El segundo "Fragment" es completamente equivalente al anterior, pero en este caso se representan las estadísticas recabadas a partir de los valores totales o acumulados leídos durante todos los momentos que el usuario ha tenido activada la monitorización. El tercero representa, mediante un gráfico de línea, la media de los valores leídos durante cada uno de los últimos 7 días. Pudiéndose observar que en el eje de abscisas se encuentran las fechas relativas a cada día representadas en formato dd/m/aaaa. Tanto la implementación del método para saber cuáles son los últimos 7 días como para establecer el formato utilizado en su representación se encuentra desarrollado en el apartado anterior de "control de tiempo y fechas" (ver apartado 4.8). Por último, los dos "Fragments" restantes son totalmente equivalentes a excepción de que en uno se representan los valores máximos de cada día durante los últimos siete días y en el otro los valores mínimos. Para dicha representación se hace uso de un gráfico de barras implementado como todos los anteriores mediante la librería MPAndroidChart, así como la representación de las fechas en el eje de abscisas sigue el mismo método que en los gráficos anteriores (ver Figura 18 y Figura 19).

Por otro lado, la actividad coordinadora de todos los "Fragments" y por tanto común a todos ellos, tiene implementado lo que se conoce como "Toolbar" junto a un menú en ella. Esto no es más que una barra situada en la parte superior de la pantalla que muestra el nombre de la aplicación, así como un botón que despliega un menú personalizado. Además, esta barra tiene la funcionalidad de ser arrastrada hacia arriba escondiéndose dejando más espacio en pantalla. En cuanto al menú, este requiere de un archivo gráfico en formato XML junto con un método implementado por código que se conoce como "inflador" que es el que hace posible superponer dicho menú en la barra de tareas y mostrarlo cuando se pulsa en el botón. Además, por código se ha implementado la función que realizará cada una de las tres secciones que componen el menú. Estás son, ordenadas de arriba hacia abajo, "actualizar", "salir" y "resetear". Si se pulsa

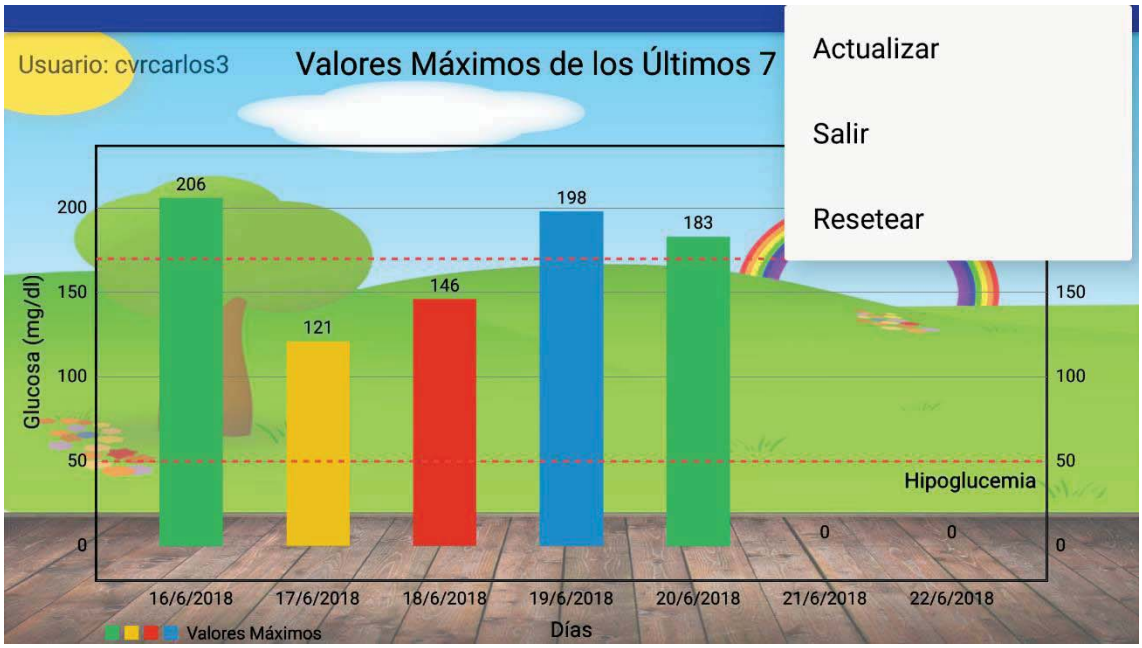
sobre “actualizar” la aplicación vuelve a lanzar la pantalla de transición con el fin de volver a leer los datos actualizados de la monitorización y poder volver a mostrarlos. En el caso de hacer clic sobre “salir” la aplicación vuelve a la pantalla principal de selección de modo. Y, por último, si se pulsa sobre “resetear” se lanza un nuevo modo que permitirá al usuario tener un control más preciso de sus estadísticas como se verá a continuación.

Figura 18. “ViewPager” de estadísticas



Fuente: Captura de pantalla de la aplicación

Figura 19. “ViewPager” de estadísticas



Fuente: Captura de pantalla de la aplicación

4.11.10. Modo resetear

Esto modo, como, su nombre indica permite resetear diferentes campos relativos a las estadísticas del usuario. Para ello la actividad cuenta con una lista expandible con dos niveles. Para la creación de esta lista ha sido necesaria la conjunción de dos archivos en formato XML adicionales relativos a cada nivel de la lista, así como un "adaptador" que es una clase java que se encarga de manejar los desplegables en función de en cual se pulse. [30]

En el primer nivel hay tres categorías, la primera es la de resetear estadísticas totales, donde en su segundo nivel, se puede resetear todas las estadísticas o solo resetear las estadísticas acumuladas no alterando las relativas a cada día guardado. La segunda categoría del primer nivel es la que da opción de resetear todas las estadísticas relativas a un día completo. Para ello se muestran en el segundo nivel una sección para cada uno de los días almacenados en la base de datos. Por último, en la tercera categoría se da la opción de resetear un campo específico de un día concreto. Para ello tras seleccionar el día del que queremos resetear alguna variable aparece en pantalla un texto informando de la selección y lo que se conoce como "Seekbar", la cual es una barra sobre la que se puede desplazar un botón, pudiendo seleccionar en este caso en función del punto en el que se deje un campo u otro de las estadísticas. En todos los casos cuando se ha establecido la variable o variables que se quieren resetear, lo cual será mostrado en un texto informativo, el usuario podrá pulsar el botón de resetear para confirmar su acción. En ese momento por seguridad saltará un diálogo preguntando al usuario si está seguro de lo que va a hacer para que en función de lo que elija ejecutar el reseteo o no. Dicho reseteo no es más que sobrescribir en la base de datos las variables correspondientes con su valor de inicialización (ver Figura 20).

Esta pantalla también cuenta con la misma "toolbar" junto con un menú como la actividad anterior con la única salvedad de que en este caso el menú solo da la posibilidad de ejecutar el código relativo a "actualizar" o "salir" tal como se vio anteriormente.

Figura 20. Pantalla resetear

Fuente: Captura de pantalla de la aplicación

4.11.11. Modo historia interactiva

Este es el último modo accesible mediante los iconos embebidos en el "Scroll View" de la pantalla principal. El icono que inicia la historia muestra a los protagonistas de esta, un osito llamado Glucosito y su mascota, un robot, de apariencia parecida al mBot utilizado en este proyecto, llamado Boti. La historia interactiva cuenta con cuatro escenarios principales (calle, clase, casa de Glucosito y parque) donde de alguno de los cuales deriva algún escenario secundario relacionado. Como su nombre indica la historia es interactiva y por tanto da la posibilidad de que el niño tenga cierto poder de decisión sobre ella, aparte de poder interactuar con ella de manera lúdica a la vez que aprende. A raíz de este poder de decisión con el que cuenta el usuario, la historia puede adoptar cuatro hilos diferentes, cada uno de los cuales termina contando todos los escenarios en un orden u otro adecuando las situaciones y diálogos a los diferentes hilos establecidos.

La historia cuenta con 9 pantallas y unas 120 acciones diferentes. Dichas acciones se estructuran dentro de una sentencia "Switch" dentro de una función que es llamada cada medio segundo. De esta forma la aplicación evalúa con esa cadencia en qué punto se encuentra la historia en cada momento. La historia tiene tres métodos de avance básicos el primero y mayoritario es el avance automático entre una acción y otra que se realiza cada 3 segundos. Dicho control del tiempo se realiza con contadores implementados en cada caso de la sentencia "Switch". El otro método que se puede presentar a lo largo de la historia es que al niño se le haga una pregunta cuya respuesta sea sí o no, y en función de su respuesta la historia avanzará por un lugar u otro. Y el último método de avance es en el que el niño debe interactuar con diversos objetos de la pantalla desde arrastrar un administrador de insulina al osito hasta pulsar sobre una nevera para abrirla. Aparte de todo esto la historia cuenta con varios botones que permiten su control. Existe un botón para pausar la historia el cual incidirá sobre el "Thread" que se encarga de la llamada continua a la función principal. Equivalente a este, se encuentra otro botón para reanudarla realizando el proceso contrario que en el pausado. También está un botón de "stop" que finaliza la historia y devuelve la aplicación a su pantalla principal de selección de modo (ver Figura 23).

Por otro lado, para hacer la historia más lúdica y llevadera se ha incluido tanto una música de fondo consistente en una melodía tocada con un ukelele y una serie de sonidos que simulan la voz de los distintos personajes cada vez que hablan. Estos sonidos son creados a partir de grabaciones personales retocadas tanto su velocidad como su tono mediante el software de edición de audio Audacity tal como se detalla en apartados anteriores (ver apartado 4.5). Ambos sonidos pueden ser pausados y reanudados mediante dos botones situados a sendos lados de los botones de control anteriormente mencionados. (ver Figura 21)

Además de todo esto, la historia cuenta con una integración notable con el dispositivo robótico, ya que cada vez que el osito mide sus niveles de glucosa, como se verá a continuación en la breve explicación del desarrollo de la historia, el robot reaccionará en función del estado en el que se encuentre Glucosito tanto en la pantalla como en el dispositivo real (ver apartado 5.6.9).

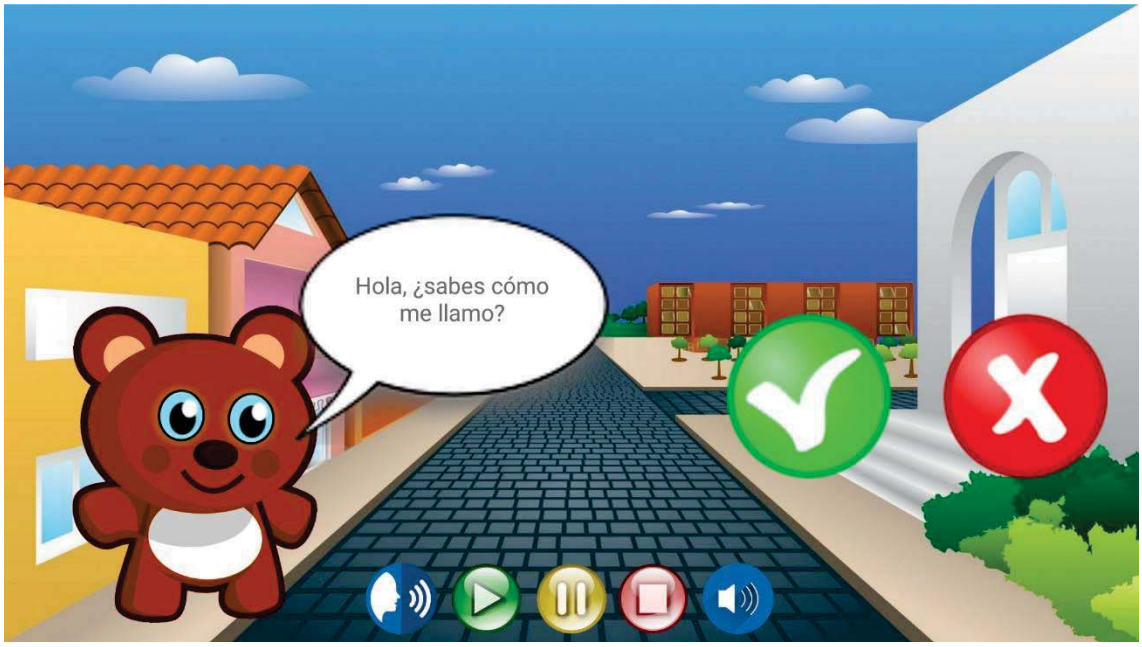
La historia comienza con una pantalla donde el osito Glucosito se encuentra en la calle y se presenta. En un momento dado este empieza a encontrarse mal y decide medir sus niveles de glucosa. El resultado de la medición es que se encuentra en hiperglucemia y el robot actúa

en consecuencia. Para avanzar el niño debe arrastrar hasta el osito el administrador de insulina. Después de este percance el osito le da las gracias y le cuenta su situación y que es diabético tipo I. Siguiendo con el primer hilo que puede adoptar la historia, el osito y por ende el niño van a clase donde el profesor les da unas pautas sobre cómo llevar una vida sana (ver Figura 22). En un momento dado del desarrollo de la historia se dibuja un gráfico sectorial en la pizarra donde se muestran los porcentajes de cada grupo de alimentos recomendados para llevar una vida saludable. Al pulsar sobre la pizarra la aplicación salta a una nueva pantalla donde aparece dicho gráfico más grande. Si se pulsa sobre cada uno de los sectores un dialogo da información adicional sobre ese grupo de alimentos. Mediante una flecha se vuelve a la actividad anterior donde a través de un control realizado mediante variables globales esta sigue su desarrollo por el punto en que se quedó. A continuación, el profesor les hace varias preguntas a las que el usuario debe responder pulsando sobre una de las cuatro opciones que aparecen en pantalla. En función de sus respuestas los diálogos del profesor variaran en concordancia. Tras terminar la clase con el sonido de una bocina, y siguiendo el primer hilo de la historia, esta se traslada a la casa de Glucosito. Allí este recibe al niño en el pasillo donde se encuentra un cuadro colgado cuyo estilo recuerda al de Van Gogh para que, tras previa invitación del protagonista a pulsar sobre él, un diálogo de información al niño sobre este famoso pintor. Tras esta escena la historia avanza hacia el baño donde el osito enseña paso a paso como se debe medir la glucosa en sangre. Para ello el niño deberá replicar arrastrando los objetos sobre la pantalla todos los movimientos necesarios en esta acción como el pinchazo para obtener la gota de sangre, su posterior aplicación en una tira de medición e introducción en el glucómetro para que haga la medida. En este caso el resultado de la medición desvela una hipoglucemia, así que, alertado por el robot, la historia se traslada a la cocina para comer y normalizar los valores de glucosa. En dicha estancia espera la madre de Glucosito que les comenta que aún queda un poco para que la comida esté preparada por lo que deciden buscar algún tentempié en la nevera. Cuando el usuario pulsa sobre la nevera esta se abre mostrando varios alimentos poco saludables y un zumo de naranja. Si el niño pulsa en algún alimento poco saludable un texto se lo hará saber indicándole que busque otro (ver Figura 23). Cuando este haga clic sobre el zumo la historia avanzará y todos juntos irán a comer. Este momento de la comida se simula con una espera de 10 segundos junto con una animación de un reloj y un texto que informa al usuario de qué está ocurriendo. Tras terminar de comer y decidir que van a ir al parque a jugar se repite la misma situación de espera, pero en este caso para reposar la comida. Con este gesto se intenta hacer énfasis en el niño de que las cosas conllevan un tiempo y una espera que hay que saber gestionar y no todo lo que se quiere se consigue de manera inmediata. Tras finalizar estos últimos 10 segundos la historia se traslada al parque donde el osito vuelve a medir su glucosa antes de hacer ejercicio. En este caso el glucómetro desvela unos niveles de glucosa normales. Sin embargo, en vista de que se va a realizar una actividad física y esta disminuirá sus niveles de glucosa, aparece en pantalla una manzana para que el usuario la arrastre hasta Glucosito. Después de esta enseñanza se lanza un minijuego en otra pantalla que simula que juegan con el balón. Este minijuego consiste en un campo de futbol con una pelota de fútbol en el centro y donde una portería aparece aleatoriamente en distintas localizaciones durante un muy breve periodo de tiempo. El reto es que el usuario arrastre el balón hasta la portería antes de que esta desaparezca y vuelva a aparecer en otro lugar. Cuando el usuario complete correctamente esta acción 10 veces la historia volverá a la pantalla anterior. En ella los protagonistas se hidratarán

después del ejercicio y se despiden dando lugar a una pantalla de despedida, donde al pulsar un botón la historia finalizará y se volverá a la pantalla principal de selección de modo.

En todo momento en la historia se busca no solo educar en el tratamiento de la diabetes sino aprovechando el entorno lúdico se pretende hacer llegar al niño valores como la amistad, la solidaridad, que todos somos iguales independientemente de nuestra situación y que todo problema se puede solventar si se cree en ello y se busca una solución.

Figura 21. Pantalla historia interactiva (presentación)



Fuente: Captura de pantalla de la aplicación

Figura 22. Pantalla historia interactiva (clase)

Fuente: Captura de pantalla de la aplicación

Figura 23. Pantalla historia interactiva (cocina)

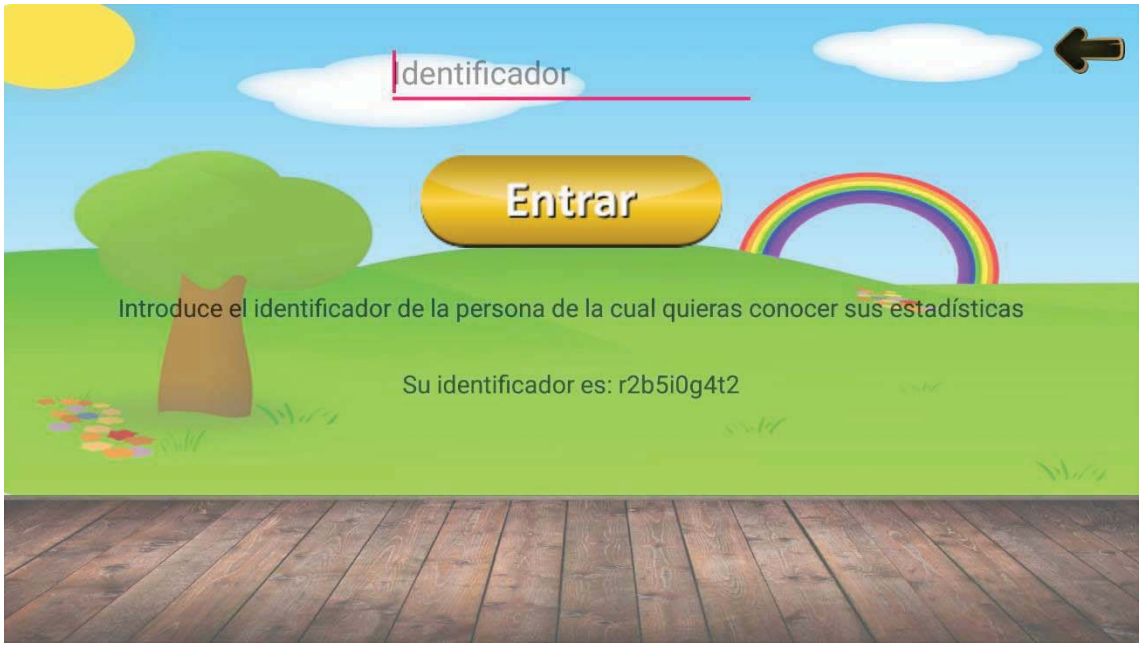
Fuente: Captura de pantalla de la aplicación

4.11.12. MODO ESTADÍSTICAS CON ACCESO REMOTO

La forma de acceder a este modo es mediante la pulsación del botón situado en la esquina superior izquierda de la pantalla principal. Al igual que el modo de estadísticas convencional tampoco precisa de haber establecido la conexión bluetooth entre la aplicación y el robot con anterioridad. Una vez se accede, se muestra en pantalla un campo para introducir texto con un "hint" (texto identificativo que se borra al pulsar sobre el campo) que pone "identificador". También hay un gran botón de entrar y dos textos más informativos. En el primero de ellos se dan instrucciones al usuario de que introduzca el identificador de la persona, la cual quiere conocer sus estadísticas. Y el otro muestra el identificador del propio usuario que tiene la sesión iniciada en la aplicación con el fin de que pueda compartirlo. El objeto de este sistema es que tanto padres como profesionales sanitarios puedan acceder a las estadísticas relativas al usuario, en este caso el niño con diabetes, desde sus propios dispositivos móviles simplemente teniendo la aplicación instalada. De esta forma el menor puede ser monitorizado a distancia sin necesidad de acceder al dispositivo de este con los problemas que esto pudiese acarrear. Todo este sistema se gestiona a través de un identificador único que la aplicación se encarga de asignar a cada usuario que se registra en ella. Este proceso ocurre en la pantalla de inicio de sesión donde mediante un bucle FOR se crea un código de 10 letras y números alternativamente. Estas letras y números se asignan de manera aleatoria utilizándose en el caso de las letras su correspondencia numérica en la tabla ASCII para su asignación aleatoria. Una vez el código está creado se almacena en la base de datos una variable de tipo "string" cuyo nombre será el identificador y su valor el nombre de usuario registrado. Este nombre de usuario no es más que la cadena de caracteres que se encuentra antes del símbolo "@" de la dirección de correo electrónico introducida. De este modo si un padre o profesional médico introduce el identificador del menor y pulsa el botón de entrar este será redirigido a la pantalla de transición

o espera previa a la de estadísticas, para del mismo modo que en su equivalente, mostrar las estadísticas, en este caso del menor, a monitorizar. En este momento cobra sentido el texto donde se informa de que usuario están siendo revelados sus registros (ver Figura 24).

Figura 24. Pantalla de estadísticas con acceso remoto



Fuente: Captura de pantalla de la aplicación

4.12. MANUAL DE USUARIO

Para que el niño tenga una primera idea acerca de cómo funciona la aplicación y sus diferentes modos y funcionalidades se ha elaborado un tríptico informativo. Este formato permite de una manera concentrada y focalizada transmitir al usuario las ideas básicas que debe conocer para entender el funcionamiento de GlucoEduca. Como el proyecto está destinado a ser usado por niños un tríptico colorido y bien estructurado resultará mucho más interesante y atractivo que un manual de instrucciones al uso. Además, que este tríptico podría ser entregado directamente cuando se le proporcione al menor todos los componentes necesarios para empezar a utilizar GlucoEduca en su totalidad.

El tríptico diseñado mediante el software Microsoft Word puede ser visualizado en el Anexo II.

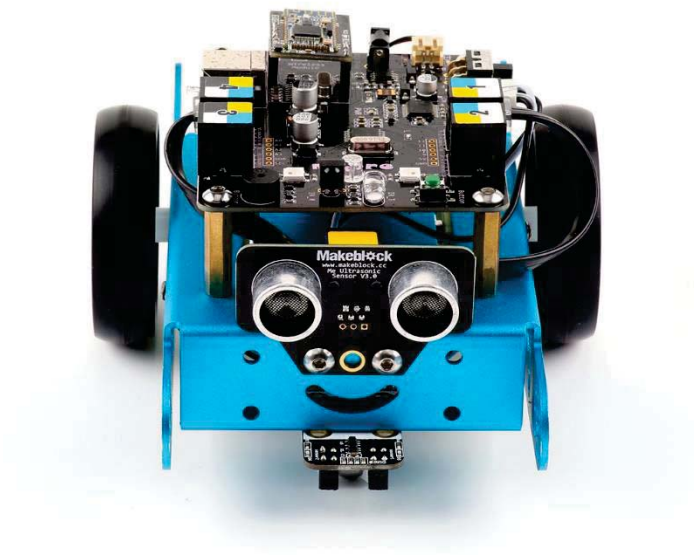
CAPÍTULO 5. MBOT, EL DISPOSITIVO ROBÓTICO

5.1. DESCRIPCIÓN

MBot se define como un kit de robótica pensado para la educación en programación de los más jóvenes. Este robot surge del proyecto Makeblock iniciado con una campaña de financiación en Kickstarter. Sus productos se caracterizan por estar construidos con materiales de gran calidad (acero reforzado y tornillería de acero inoxidable) ofreciendo un gran acabado tanto visual como de robustez. En este caso mBot se trata del dispositivo de gama más baja de la firma, pero el más adecuado para la educación. Como veremos a continuación mediante un enfoque distinto y sabiendo sacar el potencial que tiene puede ser usado no solo para educar en programación sino en muchas otras áreas como en este caso la diabetes. [31]

Este robot se compone de un armazón principal construido en acero reforzado donde mediante tornillos se van ensamblando los demás componentes que conforman el dispositivo. El principal de ellos y aquel que le da vida es la placa basada en Arduino Uno que incorpora. Esta placa está ligeramente modificada sustituyendo las conexiones de los sensores y actuadores directamente en los pines por conectores RJ25. Este tipo de conexión es una variante del RJ11 utilizado en las líneas de teléfono, pero en vez de incorporar 4 conectores cuenta con 6. Este tipo de conexión implica que la programación del robot no sea la habitual de Arduino de establecer a nivel alto o bajo un determinado pin de la placa, sino que habrá que utilizar las distintas funciones que Makeblock ofrece, a través de su software mBlock, para controlar los diferentes componentes. Por su parte, el robot cuenta con un sistema motriz mediante ruedas en configuración diferencial, aspecto que se detallará en el próximo apartado (5.2), para ello cuenta con dos motores DC cada uno de los cuales acciona una de las ruedas traseras. Además, mBot cuenta con un zumbador, dos leds interiores y un receptor y emisor de infrarrojos los cuales potencian su capacidad de interacción. Uno de los aspectos más diferenciadores de este dispositivo frente a la competencia es su matriz led 8x16 totalmente configurable, la cual es muy utilizada para informar al usuario tanto de en qué modo se encuentra, como si su monitorización está transcurriendo con normalidad o si en algún juego está obteniendo resultados positivos o negativos. Por ello la matriz leds se ha convertido en el módulo más utilizado para dotar al robot de una interfaz cara al usuario amigable y cercana. Por último, el kit incorpora una serie de sensores como el sigue líneas o el de proximidad por ultrasonidos los cuales no han sido utilizados en este proyecto. Eso no quita que estos módulos, dada su sencilla programación, puedan ser integrados en diferentes juegos en actualizaciones futuras del proyecto (ver Figura 25). [31]

Figura 25. Dispositivo robótico mBot



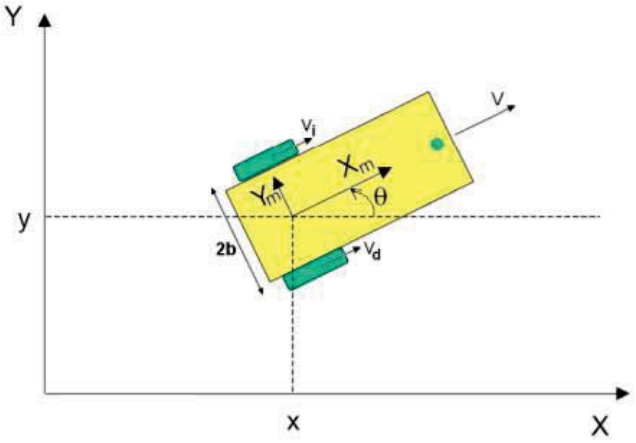
Fuente: http://www.mblock.cc/materialcat/image/

5.2. EL MOVIMIENTO DEL ROBOT

Como se ha comentada en apartados anteriores, mBot integra un sistema de movimiento basada en dos ruedas traseras de tracción y una rueda delante que solo proporciona apoyo. A esta configuración se le domina diferencial, ya que basa su movimiento en la diferencia de velocidades entre ambas ruedas motrices. Esta configuración permite una fácil implementación y programación de su movimiento, pero dificulta su control mediante algoritmos tipo PID, aspecto que no se aborda en este proyecto. Otro problema que presenta esta configuración es su dificultad para mantener trayectorias rectas ya que requeriría que los dos motores dotaran a las ruedas de exactamente la misma velocidad. Así también este sistema está más predispuesto a sufrir deslizamientos que otros como la configuración Ackerman o en triciclo (ver Figura 26) [(32)]

La trayectoria que adoptará el robot dependerá principalmente de la diferencia de velocidades entre sus dos ruedas motrices y su velocidad lineal será la media de estas: [(32)]

Figura 26. Cinemática del robot, configuración diferencial



$$V_L = \frac{v_d + v_i}{2}$$

$$\dot{\theta} = \omega_a = \frac{v_d - v_i}{2b}$$

Fuente: Diapositivas asignatura Laboratorio de Automatización y Control [32]

Estos conceptos se tienen muy en cuenta a la hora de la programación del movimiento del robot en Arduino. Así como el hecho de que esta configuración sea muy dada a deslizamientos, evitando en todo momento que el arranque del robot se produzca a velocidades altas. [32]

Los dispositivos que permiten el movimiento son dos motores de corriente continua de 6v que proporcionan una velocidad máxima de 200 rpm y una reducción de 1/48 lo cual proporciona un par mayor y la posibilidad de que el robot circule a velocidades bajas. [31]

5.3. LA PROGRAMACIÓN DE MBOT

Mbot permite su programación a través de mBlock, un software basado en Scratch, o directamente desde el IDE de Arduino. Ambos han sido utilizados en el proyecto y se detallarán a continuación. [31]

5.3.1. MBlock, programación gráfica

MBlock es un software desarrollado por el propio proyecto de Makeblock, este está diseñado para permitir la programación de los distintos robots de su gama con relativa sencillez a través de su sistema de programación gráfica basada en Scratch (creado por el MIT para permitir acceder al mundo de la programación sin unos conocimientos profundos de este). Este software al estar específicamente diseñado para los robots de la gama permite una conexión muy sencilla ya sea por bluetooth o cable, así como permite la restauración del código base o actualizar el firmware del robot. No solo sirve para programar los robots de la casa, sino que también permite programar mediante este lenguaje gráfico placas de Arduino sin modificar. En este proyecto, mBlock ha sido utilizado básicamente para conocer la definición y estructura de las distintas funciones que controlan los componentes del robot. Una de las funciones más interesantes de este software es su modo Arduino. Este permite visualizar en todo momento la traducción a Arduino de lo que se está programando con Scratch. Esta función ha sido muy utilizada para conocer en detalle las distintas funciones, tanto su definición como estructura, que controlan los diferentes componentes del mBot. Este software también ha sido

ampliamente utilizado para crear las distintas expresiones que se representan en su matriz de leds gracias a su asistente que simplifica mucho la tarea. [33]

5.3.2. Arduino IDE, programación desde código

Arduino es un ecosistema abierto que engloba tanto la parte de hardware como la de software. Fue desarrollado en la primera década del siglo XXI como un proyecto de investigación en el Interaction Design Institute de Ivrea una ciudad al norte de Italia. El proyecto nació con la intención de acercar a los estudiantes el mundo de la electrónica y los microcontroladores a un precio significativamente menor que las opciones que se encontraban en ese momento en el mercado. De esta forma en 2005 se crea la primera placa Arduino con el objetivo de ayudar a los estudiantes de diseño, sin previos conocimientos en programación ni electrónica, a crear prototipos que conectasen el mundo físico y el digital. Arduino se define a sí mismo como un proyecto integrador para crear una comunidad que pueda ayudar a expandir el uso de la herramienta y crecer a través de la contribución de miles de personas alrededor del mundo. Personas que aporten su experiencia a la hora de crear tutoriales, depurar códigos o mantener la gran cantidad de foros que existen en la web. Gracias a todo esto Arduino se sitúa como la herramienta preferida por el movimiento “Maker” y todos aquellos que desean introducirse en el apasionante mundo de la electrónica y la programación de microcontroladores. [34]

Arduino cuenta con su propio ambiente integrado de desarrollo (IDE), el cual es un software que permite tanto la conexión con las distintas placas para la transferencia de programas como una serie de librerías que permiten la programación de estas. En este software se puede programar en lenguaje Arduino basado en C/C++ cualquier placa del proyecto incluida la que incluye mBot, una placa Arduino Uno ligeramente modificada. De esta forma el Arduino IDE integra todas las etapas de la programación: creación del programa, depuración y compilación, conexión con la placa y transferencia del programa. En este proyecto el programa escrito en la placa del robot ha sido programado con este software ya que con cierto conocimiento de programación resulta mucho más eficiente y menos limitado que la programación gráfica que ofrecía mBlock. [34]

5.4. CONEXIÓN ENTRE LA APLICACIÓN MÓVIL Y EL ROBOT

Como se ha comentado en apartados anteriores la conexión entre ambos dispositivos se ha establecido mediante la tecnología bluetooth dotando al robot de libertad de movimientos y haciendo más sencillo su manejo e interacción por parte del niño. [35]

El robot cuenta con un módulo Bluetooth Dual (compatible con Bluetooth 2.0 y 4.0) que se conecta directamente a la placa con los pines que integra el propio módulo. Este está alimentado por 5v en sus pines Vcc y GND y conectado en la configuración típica donde el pin Tx (transmisión) del módulo se conecta al pin Rx (recepción) de la placa y viceversa. Este módulo es de tipo esclavo por lo que solo puede recibir comunicaciones y no iniciarlas. Para este proyecto no es necesario transmitir ningún dato desde el robot a la aplicación por lo que el módulo incorporado de serie es suficiente. [35]

Una vez el dato es recibido por el módulo bluetooth este lo escribe en el puerto serie de la placa. Este puerto es la forma genérica de llamar a la interfaz de comunicación entre dos dispositivos. El puerto serie, a diferencia del paralelo, solo necesita obligatoriamente dos

conectores uno para la transmisión y otro para la recepción por medio de los cuales se envía la información en bits. Por ello la implementación de la comunicación en el código que da vida al robot se basa en la lectura de este puerto serie constantemente. Esta acción se realiza mediante la instrucción de Arduino "Serial.read()". La velocidad de transmisión establecida (Baud rate) es de 115200 bits por segundo, una velocidad de transmisión catalogada como media-alta la cual se ha comprobado empíricamente que es la más adecuada para este proyecto. [35]

5.5. LA ESTRUCTURA DEL CÓDIGO

El código programado en el IDE de Arduino y escrito en la placa que da vida al robot se estructura como la mayoría de los códigos en Arduino en sus dos funciones básicas "setup" y "loop".

La función "setup" es la primera que se ejecuta al iniciar el programa y solo se ejecuta una vez. Esta función está destinada a una inicialización del programa y en el caso del programa que se ha implementado en el robot la función "setup" se encarga de avisar al usuario de que el sistema está esperando para realizarse la conexión a través de un parpadeo de leds, una serie de sonidos emitidos por el zumbador y una expresión mostrada mediante la matriz de leds. Previamente a la programación de esta función se ha implementado la declaración e inicialización de las diferentes variables que forman el código, así como la declaración de los diferentes sensores y actuadores utilizados. A parte de todo esto también se ha implementado una función que se encarga del movimiento del robot.

Tras ejecutarse la función "setup" es llamada la función "loop" que como su nombre indica es una función que se ejecuta constantemente en bucle. En ella se configura el programa en sí que va a ejecutar el microcontrolador y en este caso el robot. Al mBot se le ha introducido un programa que se basa en la lectura constante del valor que recibe en su puerto serie y en función de ese valor entrar en una estructura o modo donde seguirá leyendo ese puerto serial, pero este será interpretado de forma diferente. En código esto se traduce en que en cada iteración de la función "loop" una variable se actualiza con el valor que lee en el puerto serial, después mediante una serie de sentencias condicionales "if", cada una de las cuales corresponde a un modo, el programa entra en una sentencia o en otra pudiéndose decir que ha cambiado de modo. Cada fragmento de código embebido en cada sentencia condicional sigue la misma estructura. En primer lugar, una serie de instrucciones meramente de interfaz para informar al usuario de en qué modo ha entrado el robot. Luego se encuentra un bucle "while" del cual no se sale mientras no se lea el valor que corresponda a un nuevo modo o a un estado de alarma. De esta forma el robot nunca sale del modo en el que se encuentra salvo que se le ordene específicamente. Con esto se consigue que el robot ejecute un fragmento de código relativamente pequeño a gran velocidad pudiendo comprobar si se recibe una nueva orden. Dentro de este bucle "while" se implementa una sentencia "switch" donde dependiendo del valor que se lea por el puerto serie el robot ejecutará una acción u otra, la cual dependerá del modo en el que se encuentre. En los modos donde pueda haber una rápida actualización de las ordenes enviadas al robot, se ha implementado una sentencia condicional que no ejecuta la sentencia switch si no se lee un valor distinto al anterior leído. Esto permite que el bucle se ejecute de forma mucho más rápida pudiendo existir una cadencia mucho menor entre lectura y lectura de una nueva orden. De este modo también se evita que se puedan repetir instrucciones sin desearlo (ver Figura 27).

Figura 27. Fragmento de código del modo juego de memoria

```
if(val==102)
     {buzzer.tone(262, 500);
     rgbled_7.setColor(0,0,0,0);
     ledMtx_3.setColorIndex(1);
   ledMtx_3.setBrightness(6);
   drawTemp = new unsigned char[16]{127,65,85,65,85,65,127,0,0,254,130,162,146,138,130,254};
   memcpy(drawBuffer,drawTemp,16);
   memcpy(drawBuffer,drawTemp,16);
   free(drawTemp);
   ledMtx_3.drawBitmap(0,0,16,drawBuffer);
       rgbled_7.show();
       while(val!=245){
         delay(5);
         val = Serial.read();
         delay(5);
         if(val!=valant){
              switch(val){
         case 0:move(1,0);
         valant=val;
         break;
        case 1: move(1,100);
        delay(800);
        move(1,0);
        valant=val;
        break;

        case 4: move(4,120);
        delay(800);
        move(1,0);
        valant=val;
         break;

        case 2: move(2,100);
        delay(800);
        move(1,0);
        valant=val;
        break;

        case 3: move(3,120);
        delay(800);
        move(1,0);
        valant=val;
         break;
```

Fuente: Captura de pantalla del programa implementado en el Arduino IDE para el robot

Como se puede ver en la Figura 27 el modo juego de memoria corresponde al modo 102, en el momento que la aplicación escribe por Bluetooth ese valor en el puerto serie del robot, este lo lee, lo almacena en la variable "val" y entra en la sentencia condicional correspondiente. Nada más entrar se hace sonar al zumbador y se dibuja en la matriz de leds unos dados simulando el azar con el que transcurre el juego. Luego el programa entra en el bucle "while" del que no saldrá hasta que se le dé la orden desde la aplicación de volver al modo principal designado por el valor 245. En este bucle el robot va leyendo continuamente el valor recibido en su puerto serial y ejecuta la instrucción relativa al valor leído. En la Figura 27, se encuentra el fragmento relativo a los movimientos que el robot hará en la secuencia aleatoria que se creará en la aplicación (ver apartado 4.11.8). Cabe destacar varios aspectos, en cuanto al movimiento del robot. La función que implementa el movimiento de los motores recibe un valor entre -255 y 255, siendo el valor negativo el correspondiente a la marcha atrás a máxima potencia y el positivo a la marcha adelante a máxima potencia también. Por ello nunca se exigen por código valores próximos a esos límites cuando el robot parte del reposo para evitar posibles

deslizamientos. A esta función se le llama a través de la función propia “move” que tiene como entradas el movimiento en sí (avance recto, giro brusco a la derecha, giro suave a la izquierda, etc.) y el valor que define la potencia del motor como se ha detallado anteriormente. Como se desea que el robot ejecute un movimiento corto con el fin de que el jugador lo memorice rápidamente y dotar de dificultad al juego, se ejecuta la función de movimiento durante 800 milisegundos a lo que se suma el tiempo de cómputo y transmisión. Luego se llama a la función para que detenga el robot poniendo en el apartado de la potencia del motor un 0.

5.6. INTEGRACIÓN DEL ROBOT EN LA APLICACIÓN

Como se lleva incidiendo durante toda la memoria este proyecto se basa en la integración de dos interfaces como una aplicación móvil y un dispositivo robótico. Dicha integración se consigue con la escritura en el módulo Bluetooth de un valor por la aplicación e interpretado por el robot tal como se le ha programado. Siguiendo esta premisa a continuación se va a describir la interacción y la interfaz que el robot ofrece en cada modo seleccionado o en el que puede encontrarse mBot:

5.6.1. Modo principal de selección de modo

- Matriz de leds: se muestra la palabra “MODO” en ella.
- Zumbador: se emite un sonido correspondiente a la nota musical Sol sostenido, octava cuarta, durante 500 ms una vez.
- Leds: se apagan los leds en el caso de estar encendidos.
- Movimiento: se detiene el robot si estuviese en movimiento.

5.6.2. Modo de conducción libre

- Matriz de leds: se muestra la silueta de un coche.
- Zumbador: se emite un sonido correspondiente a la nota musical Do, octava cuarta, durante 500 ms cada vez que se pulsa el botón de la bocina.
- Leds: se encienden y apagan los leds en color morado (RGB: 255,0,255) cuando se pulsa el botón de los faros.
- Movimiento: se realiza uno de los ocho movimientos según se ordene mediante el joystick (avance adelante, avance atrás, giro brusco a izquierda, giro brusco a derecha, giro suave a izquierda y adelante, giro suave a derecha y adelante, giro suave a la izquierda y atrás y giro suave a la derecha y atrás).

5.6.3. Modo de simulación interactiva

- Matriz de leds: se muestra una expresión anímica u otra en función del estado glucémico que atraviese el paciente cuya simulación se está visualizando. También se mostrará una flecha parpadeante hacia arriba o hacia abajo en función de la tendencia que adopten los valores de glucosa simulados.
- Zumbador: se emite un sonido correspondiente a la nota musical Do, octava cuarta, durante 500 ms alternativamente cuando se produce una situación peligrosa de glucemia.

- Leds: se encienden y apagan los leds en color rojo (RGB: 255,0,0) durante la situación de peligro.
- Movimiento: se realiza movimientos circulares variando la velocidad y trayectoria en función del estado glucémico.

5.6.4. Modo de preguntas

- Matriz de leds: se muestra un símbolo de interrogación.
- Zumbador: se emite un sonido correspondiente a la nota musical Do, octava cuarta, durante 500 ms una vez.
- Leds: se encienden los leds en color rojo (RGB: 255,0,0) si se falla o verde (RGB: 0,255,0) si se acierta la pregunta.
- Movimiento: no se realiza ningún movimiento en este modo.

5.6.5. Modo de recompensas

- Matriz de leds: se muestra una expresión anímica de felicidad.
- Zumbador: se emite un sonido correspondiente a la nota musical Do, octava cuarta, durante 500 ms una vez.
- Leds: se encienden los leds en color rojo (RGB: 255,0,0) si no se tiene suficientes estrellas para desbloquear la mejora o verde (RGB: 0,255,0) en caso contrario.
- Movimiento: no se realiza ningún movimiento en este modo.

5.6.6. Modo de simulación continua

- Matriz de leds: se muestra una gráfica dibujada en la matriz.
- Zumbador: se emite un sonido correspondiente a la nota musical Do, octava cuarta, durante 500 ms una vez.
- Leds: se apagan los leds en el caso de estar encendidos.
- Movimiento: no se realiza ningún movimiento en este modo.

5.6.7. Modo de juego de memoria

- Matriz de leds: se muestra unos dados de juego dibujados en la matriz
- Zumbador: se emite un sonido correspondiente a la nota musical Do, octava cuarta, durante 500ms dos veces cuando el robot completa la secuencia de acciones.
- Leds: se encienden con el color (amarillo, azul, rojo, rosa y verde) que aleatoriamente haya establecido la aplicación en la secuencia de acciones.
- Movimiento: se realiza el movimiento (avance adelante, avance atrás, giro brusco a izquierda, giro brusco a derecha) que aleatoriamente haya establecido la aplicación en la secuencia de acciones.

5.6.8. Modo de estadísticas

Para acceder a este modo no es necesario tener establecida la conexión Bluetooth, en el caso de estar establecida dicha conexión, la interfaz que mostrará el robot será:

- Matriz de leds: se muestra una gráfica dibujada en la matriz.

- Zumbador: se emite un sonido correspondiente a la nota musical Do, octava cuarta, durante 500 ms una vez.
- Leds: se apagan los leds en el caso de estar encendidos.
- Movimiento: no se realiza ningún movimiento en este modo.

5.6.9. Modo de historia

- Matriz de leds: se muestra una expresión anímica u otra en función del estado glucémico que atraviese el protagonista de la historia cuando se realiza una medición de su glucosa en sangre.
- Zumbador: se emite un sonido correspondiente a la nota musical Do, octava cuarta, durante 500 ms alternativamente cuando se produce una situación peligrosa de glucemia.
- Leds: se encienden y apagan los leds en color rojo (RGB: 255,0,0) cuando se dan unos niveles de glucosa inaceptables en las medidas que se realiza el protagonista.
- Movimiento: se realiza movimientos circulares variando la velocidad y trayectoria en función del estado glucémico del osito.

5.6.10. Modo de desconexión

A este modo se accede cuando la conexión bluetooth entre el robot y la aplicación no está establecida.

- Matriz de leds: se muestra una expresión de felicidad dibujada en la matriz.
- Zumbador: se emite un sonido correspondiente a la nota música Sol sostenido, octava cuarta, durante 500 ms una vez.
- Leds: se apagan los leds en el caso de estar encendidos.
- Movimiento: se detiene el robot si estuviese en movimiento.

5.6.11. Modo de alarma

A este modo se puede acceder en cualquier momento de la ejecución del programa independientemente del modo en el que se encuentre el robot siempre y cuando esté la conexión inalámbrica establecida y la aplicación mande la orden por haber leído un valor de glucosa que no se encuentra en los márgenes seguros.

- Matriz de leds: se muestra un triángulo de peligro dibujado en la matriz.
- Zumbador: se emite un sonido correspondiente a la nota música Do, octava cuarta, durante 500 ms alternativamente durante todo el tiempo que se permanezca en el modo.
- Leds: se encienden y apagan los leds en color rojo (RGB: 255,0,0) alternativamente durante todo el tiempo que se permanezca en el modo.
- Movimiento: no se realiza ningún movimiento en este modo ya que el robot puede entrar en él en cualquier momento no sabiendo se está situado en una superficie libre de caídas.

CAPÍTULO 6. RESULTADOS Y CONCLUSIONES

La realización de este proyecto culmina en un producto llamado GlucoEduca, una aplicación para dispositivos móviles Android destinada para la educación de niños en el tratamiento de su diabetes. Para complementar y potenciar la interacción y empatía del niño con el entorno que crea GlucoEduca se ha utilizado un dispositivo robótico que emula una especie de mascota que vela por la seguridad de su dueño (ver Figura 28).

Figura 28. Icono de la aplicación

Fuente: creación propia a partir de imágenes libres de derechos con el software GIMP

Como se abordó en el apartado 1.1 “objetivo del proyecto” y en apartado 3.1 “planteamiento del problema” se definieron una serie de objetivos, subobjetivos y problemas a abordar, de los cuales se puede constatar que se han cumplido en gran medida todos. GlucoEduca es una plataforma educativa integrada por una aplicación móvil y un robot los cuales conjuntamente logran educar al niño en la gestión de la diabetes, proporcionarle un sistema de monitorización continua y tratamiento de las estadísticas derivadas, una serie de juegos y modos lúdico-educativos, así como, diversos sistemas para intentar potenciar el uso de la aplicación y mantenimiento del interés por parte del usuario. También se ha creado la infraestructura en la base de datos en la nube que permite traspasar el progreso del usuario con independencia del teléfono móvil utilizado.

Con todo ello se puede constatar que el proyecto ha derivado en un producto completo, que abarca un gran abanico en cuanto a educación en diabetes se trata. Un producto que cumple con su premisa de educar con el menor coste posible, e intentar hacer la vida de los más pequeños un poco más llevadera.

CAPÍTULO 7. TRABAJO FUTURO Y MEJORAS

- **Realizar la monitorización con un sensor real:** actualmente GlucoEduca integra la implementación de un sistema de monitorización basado en la lectura de valores de glucosa de un estudio de un paciente real incluidos en la base de datos. Una mejora que acercaría la aplicación más al mundo real sería establecer esta monitorización a través de un sensor de glucosa real.
- **Crear más juegos y modos en la aplicación:** una aplicación móvil tiene la funcionalidad de poder actualizarse fácilmente y hacerla llegar a todos los usuarios a través del lanzamiento de una nueva versión. Esto permite agregar nuevos juegos lúdico-educativos que hagan más completa la aplicación.
- **Potenciar el uso de los sensores del robot:** en este proyecto debido a la limitación de tiempo, así como la concepción del dispositivo robótico como mera interfaz muchos de sus sensores no se han usado. Sería una mejora relativamente fácil de implementar el incluir en los modos y juegos la lectura de sensores como el sigue-líneas o el de proximidad por ultrasonidos.
- **Potenciar el uso de los sensores del teléfono móvil:** los smartphones actuales incorporan gran cantidad de sensores cuyo uso podría enriquecer al proyecto de manera significativa. El uso del giroscopio o incluso de la cámara permitiría implementar nuevas funcionalidades a la aplicación.
- **Implementar un sistema de seguimiento por parte del robot:** la idea de que el robot se convirtiera en tu mascota que te sigue a todas partes y te alerta si hay algún problema sería una mejora que llevaría a GlucoEduca a otro nivel. Una opción para implementar este sistema podría ser colocar el móvil con un soporte sobre el robot y utilizar su cámara, para por medio de técnicas de inteligencia artificial, lograr que este realice el seguimiento.
- **Realizar una experiencia con niños para probar la aplicación:** GlucoEduca se muestra como un producto con la intención de salir al mercado y estar disponible para los niños a los que pueda ayudar. Debido a esto antes de su lanzamiento sería muy interesante que una muestra de este colectivo la pruebe y de su punto de vista y posibles mejoras.

CAPÍTULO 8. BIBLIOGRAFÍA

8.1. BIBLIOGRAFÍA DE INFORMACIÓN

1. Medline Plus [Internet]. Bethesda (MD): U.S. National Library of Medicine; c2017. Diabetes; 04 Abril 2017 [consultado 15 Mayo 2018]; [aproximadamente 7 pantallas]. Disponible en: https://medlineplus.gov/spanish/ency/article/001214.htm

2. OMS. Informe mundial sobre la diabetes [Internet]. Ginebra: Organización Mundial de la Salud; 2016. [consultado 15 Mayo 2018]. Disponible en: http://apps.who.int/iris/bitstream/handle/10665/254649/9789243565255-spa.pdf?sequence=1

3. FEDE. Las cifras de diabetes en España [Internet]. Madrid: Federación Española de Diabetes; 2014. [consultado 15 de Mayo 2018]. Disponible en: https://www.fedesp.es/portal/1/main_noticias.aspx?idnoticia=2306&idportal=1

4. Garrido R, Torres M. Protocolos de urgencias pediátricas. 2ª ed. Barcelona: Ergón S.A.; 2010. 75-81. Capítulo 8. Urgencias endocrinas: Diabetes. [consultado 28 de Mayo 2018]. Disponible en: https://www.aeped.es/sites/default/files/documentos/diabetes.pdf

5. Barrio Castellanos R, García Cuartero B, Gómez Gila AL, González Casado I, Hermoso López F, López García MJ, et al. Lo que debes saber sobre la diabetes en la edad pediátrica. 3ª ed. Sociedad Española de Endocrinología Pediátrica; 2008. [consultado 30 de Mayo 2018]. Disponible en: http://www.seep.es/privado/gdiabetes/libro_diabetes_infantil.pdf

6. Montilla-Pérez Manuel, Mena-López Natalia, López-de-Andrés Ana. Efectividad de la educación diabetológica sistematizada en niños que debutan con Diabetes Mellitus tipo 1. Index Enferm [Internet]. 2012 Jun [citado 2018 Jun 19]; 21 (1-2): 18-22. Disponible en: http://scielo.isciii.es/scielo.php?script=sci_arttext&pid=S1132-12962012000100005&lng=es. http://dx.doi.org/10.4321/S1132-12962012000100005.

7. Vidal M, Jansà M. Entrenamiento del paciente y de la familia en el cálculo de raciones de hidratos de carbono. Av Diabetol. 2006; 22(4): 262-268. [consultado 20 de junio 2018]. Disponible en: http://www.avancesendiabetologia.org/gestor/upload/revistaAvances/22-4-4.pdf

8. López Ramírez, Pedro Antonio, Andrade Sosa, Hugo, Aprendizaje con robótica, algunas experiencias. Revista Educación [en línea] 2013, 37 (Enero-Junio): [Fecha de consulta: 19 de junio de 2018] Disponible en: http://ucsj.redalyc.org/articulo.oa?id=44028564003> ISSN 0379-7082

9. Android.bluetooth [Fecha de consulta: 2 de junio de 2018] Disponible en: https://developer.android.com/reference/android/bluetooth/package-summary

10. Android Studio [Fecha de consulta: 25 de mayo de 2018] Disponible en: https://developer.android.com/studio/

11. Java [Fecha de consulta: 25 de mayo de 2018] Disponible en: https://www.java.com/es/download/faq/whatis_java.xml

12. Firebase Authentication [Fecha de consulta: 8 de junio de 2018] Disponible en: https://developer.android.com/distribute/best-practices/develop/firebase-authentication

13. Firebase Realtime Database [Fecha de consulta: 8 de junio de 2018] Disponible en: https://firebase.google.com/docs/database/

14. About GIMP [Fecha de consulta: 22 de junio de 2018] Disponible en: https://www.gimp.org/about/

15. About Audacity [Fecha de consulta: 23 de junio de 2018] Disponible en: https://www.audacityteam.org/about/

16. Java [Fecha de consulta: 25 de mayo de 2018] Disponible en: https://www.java.com/es/download/faq/whatis_java.xml

17. GraphView [Fecha de consulta: 5 de junio de 2018] Disponible en: https:// http://www.android-graphview.org/

18. PhilJay. MPAndroidChart [Fecha de consulta: 6 de junio de 2018] Disponible en: https://github.com/PhilJay/MPAndroidChart

19. LottieFiles [Fecha de consulta: 12 de junio de 2018] Disponible en: https://www.lottiefiles.com/

20. Date [Fecha de consulta: 15 de junio de 2018] Disponible en: https://developer.android.com/reference/java/util/Date

21. Variables Globales [Fecha de consulta: 13 de mayo de 2018] Disponible en: https://stackoverflow.com/questions/1944656/android-global-variable

22. Toasts overview [Fecha de consulta: 2 de junio de 2018] Disponible en: https://developer.android.com/guide/topics/ui/notifiers/toasts

23. Thread [Fecha de consulta: 2 de junio de 2018] Disponible en: https://developer.android.com/reference/java/lang/Thread

24. ScrollView [Fecha de consulta: 5 de junio de 2018] Disponible en: https://developer.android.com/reference/android/widget/ScrollView

25. Drag and Drop [Fecha de consulta: 8 de junio de 2018] Disponible en: https://developer.android.com/guide/topics/ui/drag-drop

26. Motion Sensors [Fecha de consulta: 10 de junio de 2018] Disponible en: https://developer.android.com/guide/topics/sensors/sensors_motion

27. ViewPager [Fecha de consulta: 12 de junio de 2018] Disponible en: https://developer.android.com/reference/android/support/v4/view/ViewPager

28. Roman Hovorka et al. Nonlinear model predictive control of glucose concentration in subjects with type 1 diabetes [Fecha de consulta: 18 de junio de 2018] Disponible en: http://www.stat.yale.edu/~jtc5/diabetes/NonlinearModelPredictiveControl_Hovorka_04.pdf

29. Toshinori Kimura. September 24,2009. On Dormand-Prince Method [Fecha de consulta: 18 de junio de 2018] Disponible en: http://depa.fquim.unam.mx/amyd/archivero/DormandPrince_19856.pdf

30. ExpandableListView [Fecha de consulta: 15 de junio de 2018] Disponible en: https://developer.android.com/reference/android/widget/ExpandableListView

31. mBot Bluetooth [Fecha de consulta: 26 de mayo de 2018] Disponible en: https://makeblock.es/productos/robot_educativo_mbot/

32. Ángel Valera Fernández. Transparencias: Seminario: Introducción al Control de Robots [Fecha de consulta: 18 de junio de 2018] Disponible en: https://poliformat.upv.es/access/content/group/GRA_13237_2017/Rob%C3%B3tica/Seminario/Intro%20a%20rob%C3%B3ticav2_alumnos.pdf

33. mBlock [Fecha de consulta: 19 de junio de 2018] Disponible en: http://www.mblock.cc/

34. What ii Arduino? [Fecha de consulta: 19 de junio de 2018] Disponible en: https://www.arduino.cc/en/Guide/Introduction

35. Módulo Bluetooth HC-06 [Fecha de consulta: 8 de junio de 2018] Disponible en: https://www.prometec.net/bt-hc06/

36. Comparativa sistemas operativos [Fecha de consulta: 8 de junio de 2018] Disponible en: http://www.pcactual.com/noticias/actualidad/ponemos-nota-sistemas-operativos-moviles-populares-2_12412

37. About EV3 [Fecha de consulta: 26 de mayo de 2018] Disponible en: https://www.lego.com/es-es/mindstorms/about-ev3

38. BQ Zowi [Fecha de consulta: 26 de mayo de 2018] Disponible en: https://www.bq.com/es/zowi

8.2. BIBLIOGRAFÍA DE RECURSOS

Todas las imágenes utilizadas en GlucoEduca provienen de https://pixabay.com, un repositorio de recursos gráficos bajo la licencia CC0 Creative Commons. Esta licencia permite gratuitamente el uso comercial, así como declara que no es necesario el reconocimiento del autor que crea el recurso.

Los recursos de audio, que no han sido de creación propia, han sido obtenidos de la página web https://www.bensound.com la cual establece para la música escogida la licencia FREE Creative Commons la cual permite el uso gratuito del proyecto siempre que se muestre la fuente en este caso la página web.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

DISEÑO E IMPLEMENTACIÓN DE UNA HERRAMIENTA EDUCATIVA SOBRE LA GESTIÓN DE DIABETES TIPO 1 EN NIÑOS EMPLEANDO UN DISPOSITIVO ROBÓTICO DE BAJO COSTE

PRESUPUESTO

Curso Académico: 2017-18

PRESUPUESTO

1. MANO DE OBRA

Se toma como referencia el salario medio de un ingeniero industrial recién graduado correspondiente a unos 21.000€ anuales. Tomando la cifra de 223 días laborables al año y una jornada completa de 8 horas se puede estimar un coste de unos 12€ por hora trabajada.

2. MATERIALES

En este apartado se detalla el coste de los materiales utilizados para la realización del proyecto:

- Ordenador portátil: 1050€
  - Procesador Intel Core i7 7th Gen a 3.5 Ghz
  - Memoria RAM 8 Gb
  - Disco Duro 1TB (HDD)+ 128GB (SSD)
  - Gráfica dedicada NVIDIA GeForce GTX1050
- Robot mBot Bluetooth + matriz de leds de Makeblock 98€
- Alimentación Robot 3€
- Sistema operativo Windows 10 145€
- Software mBlock 0€
- Software Arduino IDE 0€
- Software GIMP 0€
- Software AUDACITY 0€
- Software Android Studio 0€
- Licencia Spark de Firebase 0€
- Licencia MS Word (1 año) 69€

3. PRECIOS UNITARIOS

- **Unidad de obra 1**
  Ordenador portátil (Windows 10)
  Instalación MS Office
  Diseño y creación tríptico informativo

- **Unidad de obra 2**
  Instalación de todo el software utilizado en el proyecto:
  Arduino IDE, mBlock, Android Studio, GIMP, Audacity
- **Unidad de obra 3**
  Documentación y desarrollo de una aplicación móvil en Android Studio para educar a niños con diabetes tipo 1

- **Unidad de obra 4**
  Documentación y uso de software adicional (GIMP, Audacity, Firebase) para mejorar la aplicación

- **Unidad de obra 5**
  Documentación y desarrollo del código en Arduino que ejecuta el robot y uso de software mBlock

4. PRECIOS DESCOMPUESTOS

- **Unidad de obra 1**
  Ordenador portátil, instalación MS Office, diseño tríptico informativo:

	Importe (€)	Total (€)
1 Ordenador portátil	1050	1050
1 Sistema operativo Windows 10	145	135
1 MS Office (licencia 1 año)	69	69
4h Graduado Ingeniería Industrial	12	48
0.02 Medios auxiliares	1302	26.04
0.03 Costes Indirectos	1328.04	39.84
TOTAL		1367.88

- **Unidad de obra 2:**
  Instalación de todo el software utilizado en el proyecto

	Importe (€)	Total (€)
1 Arduino IDE	0	0
1 mBlock	0	0
1 Android Studio	0	0
1 GIMP	0	0
1 Audacity	0	0
2.5h Graduado Ingeniería Industrial	12	30
0.02 Medios auxiliares	30	0.6
0.03 Costes Indirectos	1328.04	0.92
TOTAL		31.52

- **Unidad de obra 3:**
  Documentación y desarrollo de una aplicación móvil en Android Studio para educar a niños con diabetes tipo 1

	Importe (€)	Total (€)
180h Graduado Ingeniería Industrial	12	2160
0.02 Medios auxiliares	2160	43.2
0.03 Costes Indirectos	2203.2	66.1
TOTAL		2269.3

- **Unidad de obra 4:**
  Documentación y uso de software adicional (GIMP, Audacity, Firebase) para mejorar la aplicación

	Importe (€)	Total (€)
60h Graduado Ingeniería Industrial	12	720
0.02 Medios auxiliares	720	14.4
0.03 Costes Indirectos	734.4	22.03
TOTAL		756.43

- **Unidad de obra 5:**
  Documentación y desarrollo del código en Arduino que ejecuta el robot y uso de software mBlock

	Importe (€)	Total (€)
45h Graduado Ingeniería Industrial	12	540
0.02 Medios auxiliares	540	10.8
0.03 Costes Indirectos	550.8	16.52
TOTAL		567.32

5. PRESUPUESTO DE EJECUCIÓN POR CONTRATA

Descripción	Importe (€)
1. Ordenador portátil y tríptico	1367.88
2. Instalación de software adicional	31.52
3. Documentación y desarrollo de la aplicación móvil	2269.3
4. Documentación y uso de software adicional	756.43
5. Documentación y desarrollo del código en Arduino	567.32
Presupuesto de ejecución material	4992.45
Gastos Generales 13%	649.02
Beneficio Industrial 6%	299.55
Presupuesto Total	5941.02
IVA 21%	1247.61
Presupuesto de Ejecución por Contrata	7188.63

Asciende el presente presupuesto a la expresada cantidad de:

SIETE MIL CIENTO OCHENTA Y OCHO EUROS CON SESENTA Y TRES CÉNTIMOS

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

DISEÑO E IMPLEMENTACIÓN DE UNA HERRAMIENTA EDUCATIVA SOBRE LA GESTIÓN DE DIABETES TIPO 1 EN NIÑOS EMPLEANDO UN DISPOSITIVO ROBÓTICO DE BAJO COSTE

ANEXOS

Curso Académico: 2017-18

ANEXO I. CÓDIGO SIMULADOR DE GLUCOSA

```
protected int pasosSimulacion=360/5, Nstates=10;
double[] dx = new double[Nstates];
double [] x={0,0, 916.6667,916.6667, 14.3777,0.0287,0.0046,0.2916,63.7738, 25.9337};
double t[]=new double[pasosSimulacion];
double[]SimVolGlucosaTotal=new double[pasosSimulacion];
double gamma = 0.39, basal = 1d/60d, bw=70, tmaxG = 40, vg = 0.16 * bw, egp0 = 0.0161 *bw, f01 = 0.0097 *bw,
bolus, cho;
boolean ejercicio = false;
double[] exercise =new double[3];
double [][]xtbl;
double[] ttbl;
void ConfigurarSimulador(){
   int valort=0;
   t=new double[pasosSimulacion];
   for(int i=0;i<pasosSimulacion;i++){
      t[i]=valort;
      valort+=5;
   }
Simulador();
}
void Simulador(){
   bolus=bolo;
   cho=(raciones+1)*10.0;
   ode45solver(x,t,0.00001,0);
}
void ode45solver(double []x0, double []tspan, double tol, double hmax){
   double mi_pow = 1f/6f;
   double [][]a=new double[7][6];
   a[0][0] = (double)0;
   a[1][0] = (double)1 / (double)5;
   a[2][0] = (double)3 / (double)40; a[2][1] = (double)9 / (double)40;
   a[3][0] = (double)44 / (double)45; a[3][1] = (double)-56 / (double)15; a[3][2] = (double)32 / (double)9;
   a[4][0] = (double)19372 / (double)6561; a[4][1] = (double)-25360 / (double)2187; a[4][2] = (double)64448 /
(double)6561; a[4][3] = (double)-212 / (double)729;
   a[5][0] = (double)9017 / (double)3168; a[5][1] = (double)-355 / (double)33; a[5][2] = (double)46732 /
(double)5247; a[5][3] = (double)49 / (double)176; a[5][4] = (double)-5103 / (double)18656;
   a[6][0] = (double)35 / (double)384; a[6][1] = (double)0; a[6][2] = (double)500 / (double)1113; a[6][3] =
(double)125 / (double)192; a[6][4] = (double)-2187 / (double)6784; a[6][5] = (double)11 / (double)84;
   double []b4=new double[7];
   b4[0] = (double)5179 / (double)57600; b4[1] = (double)0; b4[2] = (double)7571 / (double)16695; b4[3] =
(double)393 / (double)640; b4[4] = (double)-92097 / (double)339200; b4[5] = (double)187 / (double)2100; b4[6] =
(double)1 / (double)40;
   double []b5=new double[7];
   b5[0] = (double)35 / (double)384; b5[1] = (double)0; b5[2] = (double)500 / (double)1113; b5[3] = (double)125 /
(double)192; b5[4] = (double)-2187 / (double)6784; b5[5] = (double)11 / (double)84; b5[6] = (double)0;
   double[] c = new double[7];
   for (int i=0; i<7;i++) {
      c[i] = 0;
      for (int j=0;j<6;j++)
      {
         c[i] += a[i][j]; }
   }
```

```
double t0
 double tfinal;
   double tp;
   double hmin;
   double h;
   double delta = 0;
   double tau = 0;
   double hinf_x;

   double[][] k_ = new double[Nstates][7];
   double[] x4 = new double[Nstates];
   double[] x5 = new double[Nstates];
   double[] gamma1 = new double[Nstates];
   ttbl[0] = tspan[0];
   for (int i=0; i<Nstates;i++) {
      xtbl[0][i] = x0[i];
   }
   int nelem_t = pasosSimulacion;
   double tout[];
   double xout[][];

   for (int idx_t=0; idx_t<(nelem_t-1);idx_t++)
   {
      t0 = tspan[idx_t];
      tfinal = tspan[idx_t + 1];
      tp = t0;
      hmin = (tfinal - tp) / 1e20;
      h = (tfinal - tp) / 100; // initial step size guess
      x = x0;
      int N_est_acc_steps=(int)(tfinal-t0)*1000;
      if (hmax==0) {hmax = (int)((tfinal - t0) / 2.5);}
      tout=new double[N_est_acc_steps];
      xout=new double[N_est_acc_steps][Nstates];
      int Nsteps_rej = 0;
      int Nsteps_acc = 1;
      tout[0] = tspan[0];
      for (int i=0;i<Nstates;i++)
      {
         xout[0][i] = x0[i];
      }
      ode_function_1_diff(x, tp, dx);
      for (int i = 0; i < Nstates; i++)
      {
         k_[i][0] = dx[i];
      }
      double tt;
      double xx[];
      xx = new double[Nstates];
      double tmp;
      while ((tp < tfinal) && (h >= hmin))
      {
         if ((tp + h) > tfinal) {h = tfinal - tp;}
         for (int j = 0; j < 6; j++) // 1:6
         {

            tt = tp + (c[j+1]*h);
            for (int zz=0; zz<Nstates; zz++)
            {
               tmp = 0;
               for (int ww=0; ww<j+1; ww++)
               {
                  tmp = tmp + k_[zz][ww] * a[j+1][ww];
               }
               xx[zz] = x[zz] + h * tmp;
            }

            ode_function_1_diff(xx,tt,dx);
            for (int ii = 0; ii < Nstates; ii++)
            {
               k_[ii][ j+1] = dx[ii];
            }
         }
         for (int ii = 0; ii < Nstates; ii++)
         {
            tmp = 0;
            for (int jj=0; jj<7; jj++)
            {
               tmp = tmp + k_[ii][jj] * b4[jj];
            }
            x4[ii] = x[ii] + h * tmp;
         }
         for (int ii = 0; ii < Nstates; ii++)
         {
            tmp = 0;
            for (int jj = 0; jj < 7; jj++)
            {
               tmp = tmp + k_[ii][jj] * b5[jj];
            }
            x5[ii] = x[ii] + h * tmp;
         }
         delta = Double.NEGATIVE_INFINITY;
         hinf_x = Double.NEGATIVE_INFINITY;
         for (int i = 0; i < Nstates; i++)
         {
            gamma1[i] = x5[i] - x4[i];
            if (Math.abs(gamma1[i]) > delta)
                delta = Math.abs(gamma1[i]);
            if (Math.abs(x[i]) > hinf_x)
                hinf_x = Math.abs(x[i]);
         }
         tau = tol * Math.max(1, hinf_x);
         if (delta <= tau)
         {
            tp = tp + h;
            for (int i=0; i<Nstates;i++)
            {
               x[i] = x5[i];
            }

            Nsteps_acc = Nsteps_acc + 1;
            tout[Nsteps_acc - 1] = tp;
            for (int i = 0; i < Nstates; i++)
            {
               xout[Nsteps_acc - 1][i] = x[i];
            }

            for (int i = 0; i < Nstates; i++)
            {
               k_[i][0] = k_[i][6];
            }
         }
         else
         {
```

```
            Nsteps_rej = Nsteps_rej + 1;
          }
          if (delta == 0.0)
          {
            delta = 1e-16;
          }
          h = Math.min(hmax,((0.8 * h) *
        (Math.pow((tau / delta), mi_pow))));
        }//FIN WHILE
        ttbl[idx_t+1] = tout[Nsteps_acc - 1];
        for (int i = 0; i < Nstates; i++)
        {
          xtbl[idx_t+1][i] = xout[Nsteps_acc - 1][i];
          x0[i] = xtbl[idx_t+1][i];
        }
      }//FIN FOR
      calcularGlucosa();
      double2float();
      estadisticas();

    }//FIN ODE45SOLVER
    void ode_function_1_diff(double []x, double tp,
    double[] dx){
      double gamma2 = 0.39;
    // factor to change patient's insulin sensitivity
    (to make the patient more insulin
    sensitive or insulin resistant)
      //  Model parameters (Hovorka)
      double ag = 0.8;              // unitless
      double ka1 = 0.006;           // 1/min
      double ka2 = 0.06;            // 1/min
      double ka3 = 0.03;            // 1/min
      double sit = gamma2 * 0.00512; // min^(-1) per
    mU/L
      double sid = gamma2 * 0.00082; // min^(-1) per
    mU/L
      double sie = gamma2 * 0.052;  // per mU/L
      double k12 = 0.066;           // 1/min
      double ke = 0.138;            // 1/min
      double vi = 0.12 * bw;        // L
      double tmaxI = 55;            // min
      //  BASAL

      double g1 = x[0];
      double g2 = x[1];
      double s1 = x[2];
      double s2 = x[3];
      double I = x[4];
      double X1 = x[5];
      double X2 = x[6];
      double X3 = x[7];
      double Q1 = x[8];
      double Q2 = x[9];
      double meal_duration = 1;
      double u_cho = (cho / meal_duration) * 1000 /
    180;
      if (tp > meal_duration) {u_cho = 0;}
      double u_ins_basal = basal;
      double u_ins_bolus = bolus;
      if (tp > 1) {u_ins_bolus = 0;}
      double u_ins = 1000 * (u_ins_basal + u_ins_bolus);
      //Log.d("creacionx","u_ins: "+u_ins);
      if (ejercicio ==true){
        double exercise_factor=1;
     // standard sensitivity
        double alpha = 1.25*3.29;
    // according to Schiavon paper; it may
     require tuning for Hovorka model
        double texercise;
        double dexercise;
        texercise=exercise[0];
        dexercise=exercise[2];
        if ((tp>=texercise) &&
    (tp<(texercise+dexercise))){
          exercise_factor=alpha;   // factor for a 50%
    VO2max.... for different intensity we could
    change proportionally, but not validated
        }else if ((tp>=(texercise+dexercise)) &&
     (tp<(texercise+dexercise+180))){
          // generate slope: line equation
     (y-y0)=m*(x-x0);  y=y0+m*(x-x0)
          exercise_factor=alpha+((1-alpha)/180)
    *(tp-(texercise+dexercise));
    // factor will return to 1 after 180 minutes
        }
        sid = sid*exercise_factor;
      }
      // glucose rate of appearance
      double Ug = g2 /tmaxG;
      // insulin rate of appearance
      double Ui = s2 /tmaxI;
      // non - insulin - dependent consumption
      double F01c;
      if (Q1 / vg >= 4.5) {F01c = f01;}
      else {F01c =(f01 * Q1) / (vg * 4.5);}
      // renal execretion
      double Fr;
      if (Q1 / vg >= 9){
        Fr = 0.003 * (Q1 - 9 * vg);
      }else{
        Fr = 0;
      }
      dx[0] = ag * u_cho - (1 /tmaxG) * g1;
      dx[1] = (1 /tmaxG) * g1 - (1 /tmaxG) * g2;
      dx[2] = u_ins - s1 / tmaxI;
      dx[3] = s1 / tmaxI - s2 / tmaxI;
      dx[4] = Ui / vi - ke * I;
      dx[5] = -ka1 * X1 + ka1 * sit * I;
      dx[6] = -ka2 * X2 + ka2 * sid * I;
      dx[7] = -ka3 * X3 + ka3 * sie * I;
      dx[8] = -F01c - X1 * Q1 + k12 * Q2 –
    Fr + Ug + Math.max((egp0 * (1 - X3)),0.0);
      dx[9] = X1 * Q1-(k12 + X2) * Q2;

    }
    void calcularGlucosa(){
      for (int i = 0; i < xtbl.length; i++)
      {
        SimVolGlucosaTotal[i]=(18 * xtbl[i][8] / vg);
      }
    }
```

ANEXO II. MANUAL DE USUARIO

Primeros pasos con la aplicación

Activa y desactiva la monitorización con este botón

Pulsa en el símbolo Bluetooth para conectarte con Boti

Si pulsas en el botón rojo podrás cerrar la sesión

Pulsa en este botón y entra en el modo de conducción libre En él puedes mover a Boti con el joystick, y encender y apagar las luces. Y también puedes pitar ¡piiii, piiii!

Pulsa en este botón y entra en el modo de simulación donde aleatoriamente se elegirá un numero de carbohidratos y tu deberás establecer tu ratio y decir cuanta insulina necesitas utilizando las flechas.

Si te desplazas por la pantalla puedes ver una gráfica de tu simulación y muchas estadísticas.

Pulsa ahora en este botón y verás como Boti se mueve en función de los niveles de glucosa

Pulsa en este botón y entra en el modo de preguntas, en él puedes ganar estrellitas si respondes bien. Saldrá una nueva pregunta cada día.

Pulsa sobre la estrellita y ve a desbloquear mejoras.

Pulsa en este botón y entra en el modo de simulación a tiempo real. Pulsa sobre textos y cambiarán para mostrar más estadísticas. Para salir mantén pulsado la gráfica y aparecerá una flechita.

Pulsa en este botón y entra en el modo de juego de memoria. Boti hará varias acciones y luego tu deberás reproducir la secuencia en el mismo orden. Si lo logras podrás ir a apostar para ganar estrellas.

ANEXO III. MANUAL DEL PROGRAMADOR

1. INTRODUCCIÓN

Este manual se define como un documento de carácter formativo cuya función es la de orientar al programador sobre cómo se ha estructurado el proyecto. En él se detallan las funciones y código implementado con la idea de que este pueda modificarlo y continuar el proyecto.

2. ESTRUCTURA MANUAL

El manual se organiza diferenciando claramente los dos pilares que componen el proyecto: la aplicación móvil y el robot. Este manual trata de resumir la información detallada anteriormente en la memoria incidiendo en los aspectos claves del proyecto y en los puramente relacionados con el código, pasados por alto en esta.

3. LA APLICACIÓN MÓVIL

La aplicación ha sido desarrollado mediante el software Android Studio. Este software diferencia claramente entre el código que constituye la interfaz gráfica y el código principal donde se ejecutan las diferentes funciones que conforman la aplicación. Atendiendo a esto cada pantalla de la aplicación tiene asociados como mínimo dos archivos. El primero es el llamado "layout", está en formato XML aunque la interfaz que proporciona el software permite crearlo arrastrando diferentes "widgets", de los cuales pueden modificarse sus opciones y características. El segundo es el código principal de la pantalla o actividad donde se asocia el código a ejecutar con los distintos elementos que conforman la interfaz gráfica. Ambos archivos que componen cada actividad están designados con nombres identificativos de la pantalla que integran.

3.1. Firebase

GlucoEduca basa gran parte de su actividad en la base de datos a tiempo real de Firebase. En ella se encuentra incluido una base de datos de valores de glucosa de un paciente real. Esto será utilizado para establecer la monitorización continua. Además de esto, en ella se guardan todos los datos que necesita la aplicación sobre el usuario, así como, los campos que conforman su progreso en la aplicación. También se utiliza esta plataforma para implementar el sistema de acceso de usuario por medio de sus cuentas de correo.

La comunicación con la base de datos se realiza mediante un Database Listener, el cual mediante un refresco continuo realizado por un "thread" implementado en cada actividad, ejecuta el código de lectura que contiene.

3.2. Monitorización continua

Leyendo las muestras de la glucosa en sangre del paciente almacenados en la base de datos se simula la monitorización que se realizaría desde un sensor real colocado en el paciente. Relativo al código implementado, esta función se basa en dos hilos en segundo plano ("threads") donde en uno se ejecuta la lectura consecutiva de datos partiendo de un valor aleatorio. Y en el otro cada segundo se va actualizando las diferentes estadísticas almacenadas en la base de datos. Estos hilos se inician en la pantalla principal y no se cortan hasta que se ejecute el código asociado al "toogle button" de la monitorización.

3.3. Conexión Bluetooth

El puerto de transferencia de datos Bluetooth se abre al pulsar sobre el icono de la pantalla principal y se cierra y se reabre cada vez que se pasa de pantalla. Esto se realiza porque experimentalmente se comprobó que este puerto se saturaba si se mantenía mucho tiempo abierto sin resetearse.

3.4. Personalización de la base de datos a cada usuario

Como se ha comentado en el apartado 3.2 de este manual las estadísticas procedentes de la monitorización se asocian al usuario mediante el siguiente sistema. Mediante la utilización del código de identificación único del dispositivo móvil se le asocia el nombre de usuario con sesión activa en la aplicación en ese dispositivo. De esta forma leyendo este nombre de la base de datos se personaliza la ruta de los diferentes campos con ese nombre.

Además, mediante un sistema de control de fechas basado en conocer la diferencia entre la fecha actual y siete fechas almacenadas en la base de datos se dilucida que campo de las estadísticas se actualiza.

3.5. Progreso y recompensas

Para incentivar el uso de la aplicación por parte del niño se ha establecido un sistema de recompensas que el usuario irá recibiendo en función de su buen hacer en los diferentes modos de la aplicación. Este sistema se basa en la actualización en la base de datos y su posterior lectura en las diferentes actividades de una variable que almacena el número de estrellas, con ese mismo nombre, que ha ganado el usuario.

3.6. El simulador

En el modo de simulación interactiva se ha integrado un simulador de glucosa en sangre basado en la en la resolución de una serie de ecuaciones diferenciales mediante un algoritmo de integración numérica basado en el algoritmo de Dormand-Prince. El código implementado en la aplicación se puede encontrar en el Anexo 1.

4. EL ROBOT

El dispositivo robótico utilizado ha sido el mBot de Makeblock. Un robot de tres ruedas basado en Arduino que cuenta con multitud de sensores y actuadores. Destaca su matriz de leds programable desde el asistente que ofrece el software de la marca mBlock.

4.1. CONEXIÓN CON LA APLICACIÓN

La conexión es vía Bluetooth y esta se realiza mediante la apertura de los puertos de transmisión mediante la función implementada “openBT”, luego se inicializa la conexión con la función “startBT” o cerrarla mediante “endBT”. Para finalizar se realiza el envío de la información a través “sendBT”.

4.2. MOVIMIENTO DEL ROBOT

El movimiento de este se basa en un sistema de tres ruedas en configuración diferencial por lo que la trayectoria se marca por la diferencia de velocidades entre ambas ruedas traseras. Hay que tener muy en cuenta el posible deslizamiento de estas evitando arranques bruscos. Todo ello se controla con una función propia llamada “move” que ejecuta en función de los parámetros introducidos la función del mBot que mueve los motores a la potencia y sentido indicados (ver Figura 27).

4.3. PROGRAMACIÓN DEL ROBOT

El robot se puede programar tanto en el software mBlock como en el IDE de Arduino. La combinación utilizada en este proyecto ha sido la de realizar la programación directamente en Arduino, pero usando mBlock para conocer la estructura y definición de las funciones encargadas de controlar los actuadores y sensores del mBot.

4.4. ESTRUCTURA DEL CÓDIGO

El código se estructura básicamente en una división en modos donde se accede mediante sentencias “if” y se sale cuando se vuelve al modo principal o a una situación de alarma. En todo momento a veces en varias ocasiones dentro del modo se realiza la lectura del puerto serie de la placa y se almacena en la variable “val”. Esta variable es la que se evalúa en todo momento para decidir en qué modo entrar o que acción ejecutar.