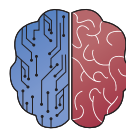




UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería de la Salud



INGENIERÍA
DE LA SALUD

**TFG del Grado en Ingeniería de la
Salud**

**título del TFG
Documentación Técnica**

Presentado por Naiara Gadea Rodríguez Gómez
en la Universidad de Burgos

25 de junio de 2023

Tutor: Pedro Luis Sánchez Ortega

Índice general

Índice general	i
Índice de figuras	iii
Índice de tablas	iv
Apéndice A Plan de Proyecto Software	1
A.1. Introducción	1
A.2. Planificación temporal	1
A.3. Planificación económica	2
A.4. Viabilidad legal	4
Apéndice B Documentación de usuario	7
B.1. Requisitos software y hardware para ejecutar el proyecto.	7
B.2. Instalación / Puesta en marcha	11
B.3. Manuales y/o Demostraciones prácticas	28
Apéndice C Manual del desarrollador / programador / investigador.	31
C.1. Estructura de directorios	31
C.2. Compilación, instalación y ejecución del proyecto	33
C.3. Pruebas del sistema	33
C.4. Instrucciones para la modificación o mejora del proyecto.	33
Apéndice D Descripción de adquisición y tratamiento de datos	35
D.1. Descripción formal de los datos	35
D.2. Descripción clínica de los datos.	35
Apéndice E Manual de especificación de diseño	37

E.1. Planos	37
E.2. Diseño arquitectónico	37
Apéndice F Especificación de Requisitos	39
F.1. Diagrama de casos de uso	39
F.2. Explicación casos de uso.	39
F.3. Prototipos de interfaz o interacción con el proyecto.	51
Apéndice G Estudio experimental	57
G.1. Cuaderno de trabajo.	57
G.2. Configuración y parametrización de las técnicas.	57
G.3. Detalle de resultados.	57
Bibliografía	59

Índice de figuras

A.1. Planificación temporal seguida para la realización de este proyecto. . .	1
B.1. Diagrama de la primera versión del prototipo, empleando el sensor SW520D	12
B.2. Diagrama de las realizadas para la implementación de la primera versión del protoripo.	13
B.3. Diagrama de la segunda versión del prototipo, empleando el módulo MPU-6050	17
B.4. Diagrama de las conexiones realizadas para la implementación de la segunda versión, siendo el módulo MPU-6050, U2.	18
F.1. Diagrama de casos de uso	39
F.2. Pantalla de inicio de sesión.	51
F.3. Pantalla de inicio con información en tiempo real.	52
F.4. Pantalla con las estadísticas en forma de gráfica de distintos periodos de tiempo.	53
F.5. Pantalla con juegos y ejercicios de mejora de la postura.	54
F.6. Pantalla de ajustes del dispositivo conectado.	55
F.7. Pantalla del perfil del usuario.	56

Índice de tablas

A.1. Resumen de gastos y precio total del producto	2
B.1. Requisito Funcional 1 'Aplicación'	7
B.2. Requisito Funcional 2 'Iniciar grabación'	8
B.3. Requisito Funcional 3 'Identificación de perfiles'	8
B.4. Requisito Funcional 4 'Detección postural'	8
B.5. Requisito Funcional 5 'Comunicar una postura incorrecta'	9
B.6. Requisito Funcional 6 'Realizar seguimiento'	9
B.7. Requisito Funcional 7 'Manual de usuario'	9
B.8. Requisito Funcional 8 'Batería'	10
F.1. CU-01. Encender dispositivo.	40
F.2. CU-02. Apagar dispositivo.	41
F.3. CU-03. Cargar dispositivo.	42
F.4. CU-04. Calibrar dispositivo.	43
F.5. CU-05. Dar de alta usuario.	44
F.6. CU-06. Iniciar sesión.	45
F.7. CU-07. Realizar monitoreo de la postura.	46
F.8. CU-08. Guardar datos.	47
F.9. CU-09. Generar informe.	48
F.10. CU-10. Ejercitar la musculatura de la postura.	49
F.11. CU-11. Consultar instrucciones de uso.	50

Apéndice A

Plan de Proyecto Software

A.1. Introducción

Para el correcto desarrollo del proyecto se seguirá una planificacion temporal y se desarrollara una planificacion economica para ver el coste economico aproximado del producto desarrollado a lo largo del proyecto. Además, se dispondrá de un apartado donde consultar la viabilidad legal del desarrollo del proyecto.

A.2. Planificación temporal

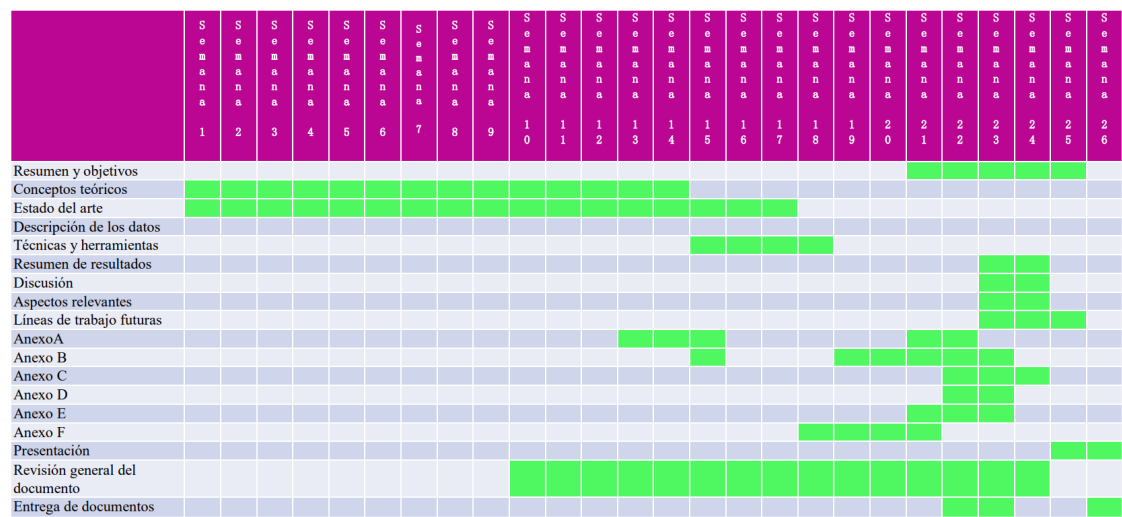


Figura A.1: Planificación temporal seguida para la realización de este proyecto.

En la planificación temporal se puede ver cada apartado de la memoria y los anexos desarrollados a lo largo de las semanas.

A.3. Planificación económica

falta incluir tiempo de desarrollo del ingeniero. SIEMPRE, porque sino estás diciendo que tu trabajo NO VALE NADA ...

también se incluyen los equipos de desarrollo amortizados en 4 años, pero solo aplicando el tiempo de uso para el proyecto, es decir si son 4 o 6 meses esa parte proporcional de la amortización

quizás te pueden decir que los desgloses mejor en tablas... pero no es importante.

y en los precios siempre hay que decir si con impuestos o sin impuestos. También te preguntarán por el proveedor...

en algunos casos en los costes se incluye el reciclado, aunque al no ser una producción industrial yo indicaría que se considera PROTOTIPO o PRE.serie pero nunca producción donde si hay que contemplarlo por legislación, creo.

Para obtener una buena planificación económica se deberán identificar los gastos y los ingresos relacionados con el desarrollo del producto.

Como primeros gastos incluiremos el precio de los componentes y los gastos de producción, mientras que para obtener los ingresos se tendrá en cuenta el beneficio, gastos imprevistos e I+D del producto, para posibles mejoras en el futuro. La suma de gastos e ingresos nos devolverá el precio final del dispositivo.

	Cálculos	Versión 1	Versión 2
Gastos de los componentes	Suma de los precios de cada componente del producto	27€	32.5€
Gastos de producción	10 % del precio de los componentes	2.7€	3.25€
Ingresos destinados a beneficio	5 % del total de gastos	1.48€	1.79€
Gastos imprevistos e I+D	10 % del total de gastos	2.97€	3.58€
Precio Total	Suma de los gastos e ingresos	34.15€	41.10€

Tabla A.1: Resumen de gastos y precio total del producto

El prototipo de la versión 1, en la que se emplea el sensor SW520D, tiene un coste final aproximado¹ de unos 34.15€, precio que podría ser menor al crear nuestro propio microcontrolador o utilizar una alternativa similar a arduino, puesto que es el elemento que más aumenta el precio de la solución, siendo el precio del resto de los componentes aproximadamente unos 3€.

El prototipo de la versión 2, en la que se emplea el módulo MPU6050, tiene un coste final aproximado de unos 41.10€, precio que también podría disminuir al crear nuestro propio microcontrolador o utilizar una alternativa a arduino, ya que sin el microcontrolador Arduino el precio ronda los 8.5€.

Desglose de los precios de los componentes del prototipo Versión 1

Se han tenido en cuenta los precios más bajos encontrados de cada componente necesario.

- Arduino UNO R3: 24€
- Resistencias (330 Ω , 2x220 Ω , 33 Ω , 1000 Ω): 0.05€
- Zumbador pasivo: 0.25€
- Motor de vibración: 1€
- Transistor: 0.05€
- SW520D: 0.5€
- Led azul: 0.02€
- Pulsador: 0.05€
- Otros elementos variados: 1€

Desglose de los precios de los componentes del prototipo Versión 2

Se han tenido en cuenta los precios más bajos encontrados de cada componente necesario.

¹La planificación económica será variable en el tiempo y durante el desarrollo del producto, por lo que se trata de precios totales aproximados. Igual para el prototipo versión 2.

- Arduino UNO R3: 24€
- Resistencias (2x330 Ω , 2x220 Ω , 33 Ω , 1000 Ω): 0.06€
- Zumbador pasivo: 0.25€
- Motor de vibración: 1€
- Transistor NPN: 0.05€
- MPU-6050: 6€
- Led azul: 0.02€
- 2xPulsador: 0.10€
- Otros elementos variados: 1€

A.4. Viabilidad legal

Se debe tener en cuenta en todo momento que el dispositivo sea completamente seguro y no afecte negativamente al usuario. Para ello, existen legislaciones específicas a cada fase del desarrollo y comercialización del producto que se deben cumplir para obtener un dispositivo seguro y regulado.

Se pueden diferenciar 3 fases, una primera fase de creación de la idea, diseño y desarrollo y realización de pruebas, una segunda fase de comercialización y la última fase de posventa, donde se incluyen las demandas y la gestión de los datos de los usuarios.

Durante la primera fase de creación de la idea, diseño y desarrollo del producto y realización de pruebas, todos los movimientos que se realicen se deberán ajustar a las siguientes legislaciones:

- Ley 24/2015[1], Ley de Patentes, donde se regula todo lo relacionado con la protección de invenciones empleando patentes, desde el registro de las patentes, invenciones patentables, el derecho a la patente y los procedimientos para pedir una patente.
- Real Decreto Legislativo 1/1996[2] relativo Ley de Propiedad Intelectual que regulariza la protección del derecho de autor y de derechos similares.

- Los productos sanitarios se rigen por la Agencia española de medicamentos y productos sanitarios (AEMPS)[3]. En este proyecto nos interesan especialmente el Real Decreto 1591/2009[4] que regula todo lo relativo a los productos sanitarios, desde su desarrollo a su venta, y el Real Decreto 437/2002[5] establece las pautas para la concesión de licencias de fabricación y desarrollo de productos sanitarios.
- Reglamento de la UE 2017/745[6] de Productos Sanitarios de la Unión Europea este reglamento establece requisitos y regula la comercialización de productos sanitarios en la Unión Europea, con el fin de garantizar un dispositivo de calidad, eficaz y completamente seguro.
- Además, durante todo el desarrollo del producto se deberá cumplir con la normativa laboral española[7], que incluye leyes y reglamentos como pueden ser el Estatuto de los Trabajadores, la Ley de Prevención de Riesgos Laborales o la Ley de Igualdad.

Si se consigue crear el dispositivo en base a todas las leyes anteriores y se quisiera sacar a mercado se deberán cumplir también con los siguientes requisitos legales:

- La Ley 34/2002[8] de Servicios de la Sociedad de la Información y de Comercio Electrónico, en caso de que se realice una tienda web oficial de comercialización del dispositivo.
- Además, se deberán tener en cuenta otras leyes[9] como la Ley 7/1996[10] de Ordenación del Comercio Minorista.

Por último, si el dispositivo se ha puesto a la venta se debe pensar en los requisitos legales que se necesitarán cumplir a partir del momento de la primera venta. Alguno de estos requisitos serán:

- Ley Orgánica 3/2018[11] de Protección de Datos Personales y garantía de los derechos digitales. Para poder proteger cualquier información que identifique a una persona, de forma confidencial. Además, el usuario debe estar correctamente informado del tratamiento de sus datos, además el acceso al tratamiento de sus datos debe ser claro y accesible.

El usuario tendrá derecho al acceso de sus datos, derecho de rectificación y supresión de sus datos, derecho a la limitación del tratamiento de sus datos, derecho a la portabilidad de sus datos y el derecho a oponerse al tratamiento de sus datos. Por todo ello el tratamiento de sus datos debe ser

tras la confirmación clara del consentimiento informado del tratamiento de sus datos.

- Reglamento UE 2016/679[12] relativo a Protección de las Personas Físicas en lo que respecta al tratamiento de datos personales y circulación de estos Datos. Donde se define que se debe garantizar la protección de los datos con los que se trabaja, además de notificar brechas de seguridad o exposición de datos al usuario.

Además el dispositivo deberá contar con un certificado CE[13], que garantizará que el dispositivo cumple con los requisitos de seguridad, protección y sanidad europeos. Una vez se obtenga el certificado el dispositivo podrá ser comercializado legalmente en la Unión Europea.

Apéndice B

Documentación de usuario

B.1. Requisitos software y hardware para ejecutar el proyecto.

En esta sección del anexo se definen varios requisitos tanto de software como de hardware que requiere el dispositivo objeto de este trabajo. Cuantos más requisitos cumpla el prototipo más exitoso resultará el dispositivo.

Requisitos funcionales

RF-01	Aplicación
Descripción	La solución deberá contar con una aplicación, ya sea una aplicación de escritorio, web o móvil, para simplificar la experiencia de uso y la visualización de resultados por parte del usuario.
Importancia	Alta, es la base de la visualización del seguimiento de la persona que utiliza el dispositivo.
Prioridad	Alta

Tabla B.1: Requisito Funcional 1 'Aplicación'

RF-02	Iniciar grabación
Descripción	La solución deberá contar con una opción de grabación, con la cual el profesional o el usuario tendrán la posibilidad de comenzar y finalizar el registro de la postura. Los resultados durante la grabación se almacenarán en la plataforma para su análisis.
Importancia	Alta, ya que la grabación de las respuestas permitirá al profesional analizarlas de forma detallada con el objetivo de obtener conclusiones y determinar el grado y evolución de la afectación.
Prioridad	Alta

Tabla B.2: Requisito Funcional 2 'Iniciar grabación'

RF-03	Identificación de perfiles
Descripción	La aplicación debe ser capaz de diferenciar a diferentes perfiles, en el caso de uso de una organización o un profesional, y una única identificación en el caso de que se trate de un usuario particular.
Importancia	Media, una vez se obtenga la base del dispositivo y su funcionamiento se puede dividir a los usuarios entre profesionales o particulares, con distintas funciones para cada uno de ellos.
Prioridad	Media

Tabla B.3: Requisito Funcional 3 'Identificación de perfiles'

RF-04	Detección de la postura
Descripción	La solución deberá ser capaz de detectar los cambios en la postura. Para ello se deberá implementar un algoritmo que filtre en función de los datos en crudo recogidos, una postura correcta o incorrecta. Esta medición se podría obtener en forma de 'porcentaje de buena postura'.
Importancia	Alta, dado que es la base que permitirá definir si la persona lleva una buena postura o no, y en base a ello, realizar la comunicación correspondiente y obtener las estadísticas necesarias para la toma de decisiones.
Prioridad	Alta

Tabla B.4: Requisito Funcional 4 'Detección postural'

RF-05	Comunicar una postura incorrecta
Descripción	La solución debe poder comunicar mediante, vibración, sonido u otra manera una mala postura continuada durante un periodo de tiempo definido.
Importancia	Alta, es necesario que el usuario conozca en todo momento su situación, para poder corregir su postura cuando sea necesario.
Prioridad	Media

Tabla B.5: Requisito Funcional 5 'Comunicar una postura incorrecta'

RF-06	Realizar seguimiento
Descripción	La información registrada por el dispositivo debe quedar almacenada para valorar y evaluar la postura del paciente, con el fin de modificar o no el tratamiento o fisioterapia o tomar otro tipo de decisiones. La visualización de la información recogida se reflejará en forma de gráficos y tablas. Esto permitirá analizar la información de manera clara y sencilla
Importancia	Alta, ya que será clave para la toma de decisiones por parte del especialista en cuanto a la personalización del tratamiento y rehabilitación.
Prioridad	Alta

Tabla B.6: Requisito Funcional 6 'Realizar seguimiento'

RF-07	Manual de usuario
Descripción	La solución deberá incluir unas instrucciones que se entreguen al usuario que lo vaya a utilizar. Esto supone un apoyo durante todo el proceso de uso del dispositivo y de la aplicación por parte del usuario.
Importancia	Media, puesto que supone un apoyo para el usuario que lo utilice.
Prioridad	Baja

Tabla B.7: Requisito Funcional 7 'Manual de usuario'

RF-08	Batería
Descripción	El dispositivo debe disponer de una batería para poder utilizarlo de forma telemática. Además, la batería del dispositivo debe ser suficiente para el uso previsto.
Importancia	Media, se debe incluir para mayor comodidad y libertad del paciente al utilizar el dispositivo.
Prioridad	Media

Tabla B.8: Requisito Funcional 8 'Batería'

Requisitos no funcionales

- **Accesibilidad:** la aplicación debe ser accesible para el mayor grupo de personas posible, tengan o no algún tipo de discapacidad.
- **Seguridad:** el dispositivo electrónico debe ser seguro y la información manejada en la aplicación debe estar protegida.
- **Compatibilidad:** la aplicación debe ser compatible con distintos dispositivos.
- **Eficiencia:** la aplicación debe permitir al usuario lograr sus objetivos, con un coste computacional y temporal bajo.
- **Efectividad:** la aplicación debe cumplir con exactitud los requisitos funcionales.
- **Errores:** la aplicación debe presentar una tasa de error baja, además debe mostrar posibles soluciones en caso de anomalías.
- **Usabilidad:** tanto el uso del dispositivo electrónico como de la aplicación debe ser sencillo, es decir, se debe poder usar de forma intuitiva.
- **Memorabilidad:** tanto el funcionamiento del dispositivo electrónico como de la aplicación debe ser fácil de recordar, tras no haberlos utilizado durante un tiempo.
- **Satisfacción:** el usuario debe estar satisfecho con el dispositivo electrónico y la aplicación, tanto por su comodidad, estética y usabilidad.

B.2. Instalación / Puesta en marcha

Como se han creado dos versiones los requisitos de instalación y puesta en marcha serán diferentes.

Para la primera versión del prototipo donde se emplea únicamente el sensor SW520D, es necesario tener instalado el software de Arduino IDE [Citar la página donde se instala el software](#), sin necesidad de instalar cualquier librería adicional.

Mientras que para la segunda versión se ha de tener instalado el software de Arduino IDE[14, 15], además de tener instaladas las siguientes librerías:

- I2Cdev:
- MPU6050:
- Wire:

Una vez tenemos instalado el software y las librerías necesarias el siguiente paso será el de montar el prototipo siguiendo los esquemas especificados.

Si se ha montado la primera versión del prototipo se deberá descargar el programa que se encuentra en el repositorio de GitHub bajo el nombre de 'ProgramaV1'. En el caso de haber montado la segunda versión, se deberá descargar el programa del repositorio de GitHub con el nombre de 'ProgramaV2'. Estos programas son los que abriremos y se enviarán al microprocesador Arduino UNO R3, el prototipo montado, que se encontrará conectado con el cable USB al ordenador. Una vez cargados los programas se podrá realizar el control postural, de forma que se consiga una ejecución exitosa.

Versión 1, empleando el sensor SW520D

Como primera versión se ha creado un prototipo empleando arduino y el sensor SW520D[16].

Se ha incluido un botón de encendido, un led de color azul que indica que el dispositivo se encuentra encendido, un zumbador pasivo que actuará como señal sonora, un motor de vibración que actuará como aviso vibratorio y el sensor de inclinación SW520D.

Cuando el sensor detecta que la persona se ha inclinado, por lo tanto detecta una mala postura y salta una alerta de sonora (melodía modificable empleando el zumbador) y vibratoria (motor de vibración).

Se pueden observar los componentes y las conexiones realizadas en las siguientes imágenes:

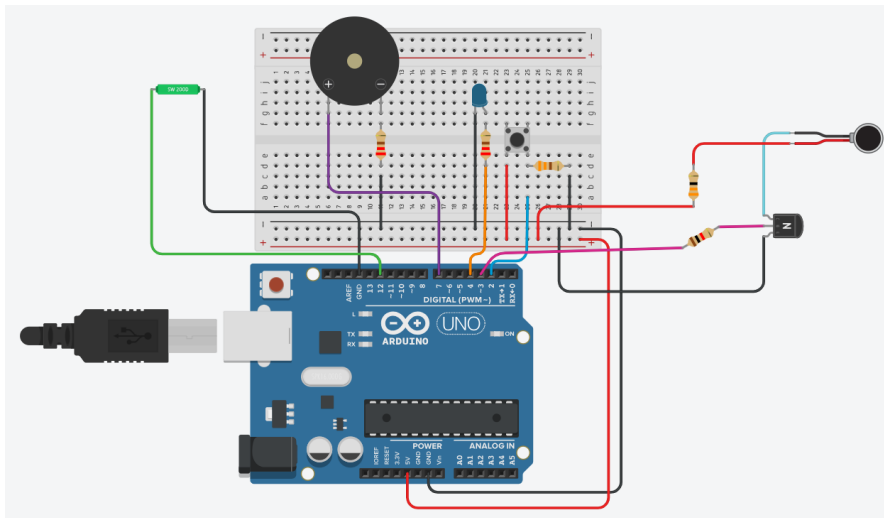


Figura B.1: Diagrama de la primera versión del prototipo, empleando el sensor SW520D

El código empleado para el funcionamiento de esta primera versión ha sido el siguiente:

```

1 // Led simple + Boton + Zumbador + Tilt + Motor Vibracion
2 // Naiara Gadea Rodriguez Gomez
3 //
4
5 int led = 4; // Seleccin del pin del led (pin digital)
6 int boton = 2; // Seleccin del pin del boton (pin digital
7 int zum = 7; // Seleccin del pin del zumbador
8 int tilt = 12; // Seleccin del pin del sensor SW250D
9 int motor = 3; // Seleccin del pin del motor de vibracion
10
11 int estado; // Estado del boton
12
13 void setup() {
14     // put your setup code here, to run once:
15     Serial.begin(9600);
16     pinMode(led, OUTPUT); // inicializacion del pin led.
17     pinMode(boton, INPUT); // inicializacion del pin del
        boton.

```

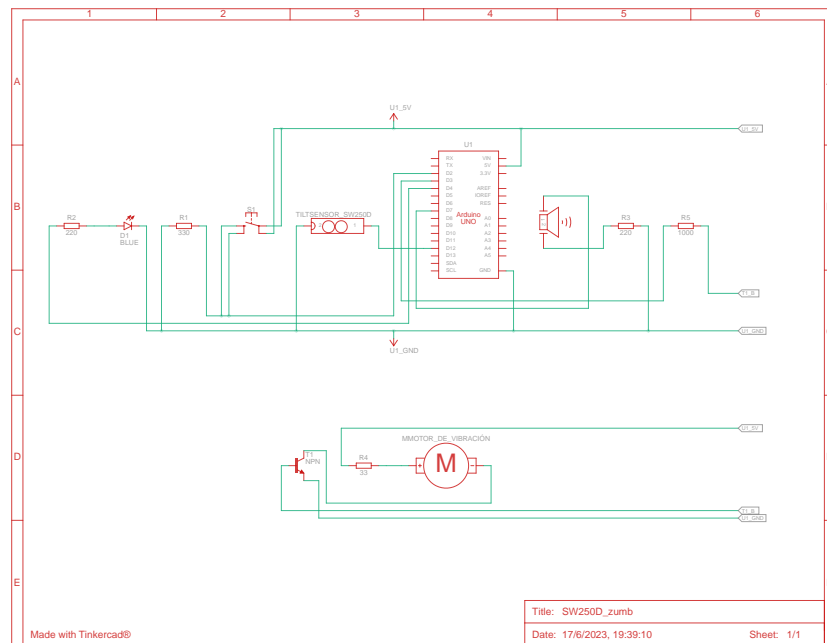


Figura B.2: Diagrama de las realizadas para la implementación de la primera versión del prototipo.

```

18  pinMode(zum, OUTPUT); // inicializacion del pin del
    zumbador
19  pinMode(tilt , INPUT); // inicializacion del pin del
    sensor tilt
20  digitalWrite(tilt , HIGH); // Sensor tilt
21  pinMode(motor, OUTPUT); // inicializacion del pin del
    motor de vibracion.
22 }
23
24 void loop() {
25     // put your main code here, to run repeatedly:
26     Serial.println(digitalRead(tilt)); // Comprobar en el
        Serial Monitor.
27     if (estado == LOW && digitalRead(boton)){
28         // Si se presiona el boton se enciende el dispositivo
29         digitalWrite(led , HIGH); // Encendido

```

```
30     delay(1000); // Durante 1 segundo (1000 ms)
31     estado = HIGH; // Cambia el estado del boton a
    encendido.
32
33 }
34 if(estado == HIGH){
35     // Si el sensor tilt hace contacto, el usuario tiene
    una mala postura y el dispositivo manda un aviso,
    musical o de vibracion.
36     if (digitalRead(tilt)) {
37         // Vibracion intermitente
38         digitalWrite(motor, HIGH); //vibracion
39         delay(500); // delay 0.5 seconds
40         //digitalWrite(motor, LOW); //stop vibrating
41         //delay(500); //wait 0.5 seconds.
42         // si vemos que durante la musica no se enciende el
    motor
43
44         melodia(); // En este caso es un aviso sonoro, pero
    teniendo un motor de vibracion se puede utilizar un
    aviso vibratorio.
45
46     } else {
47         // Si no hay contacto con el sensor tilt, no suena la
    melodia
48         noTone(zum); // El zumbador ya no emite ruido
49         //delay(3000);
50         digitalWrite(motor, LOW); // Paramos el motor
51     }
52
53     // Si se presiona el boton se apaga el dispositivo
54     if (digitalRead(boton)){
55         digitalWrite(led, LOW); // Apagado
56         delay(1000); // Durante 1 segundo (1000ms)
57         estado = LOW; // Cambia el estado del boton a apagado
    .
58     }
59
60 }
61
```

```
62 }
63
64 // Definimos las notas
65 int Do = 261;
66 int Re = 293;
67 int Mi = 329;
68 int Fa = 349;
69 int Sol = 392;
70 int La = 440;
71 int Si = 493;
72
73 void melodia(){
74     // Escala de musica con el zumbador
75     tone(zum, Fa, 500);
76     delay(700);
77     tone(zum, Sol, 500);
78     delay(700);
79     tone(zum, Sol, 500);
80     delay(700);
81     tone(zum, La, 1000);
82     delay(1700);
83     tone(zum, Sol, 500);
84     delay(700);
85     tone(zum, Fa, 500);
86     delay(700);
87     tone(zum, Sol, 500);
88     delay(700);
89     //tone(zum, Do, 1000);
90     //delay(1700);
91     //tone(zum, Fa, 500);
92     //delay(700);
93     //tone(zum, La, 500);
94     //delay(700);
95     //tone(zum, Fa, 500);
96     //delay(700);
97     //tone(zum, Re, 1000);
98     //delay(1700);
99
100 }
```

Durante el desarrollo de esta primera versión del proyecto donde se empleaba el sensor SW520D, el más sencillo, se observó que sí que es capaz de realizar un control postural, pero no se trata de un dispositivo de gran precisión, ya es muy sensible a vibraciones.

Aunque este sensor no cumple con el requisito de la precisión esta primera versión ha permitido analizar las posibilidades de mejora y las ventajas que supone trabajar con un sensor de mayor precisión que se emplea en la fase 2.

Versión 2, empleando el sensor MPU-6050

La segunda versión se ha creado en base al microprocesador arduino y el módulo MPU6050[17, 18].

Esta segunda versión mantiene el botón de encendido, el led de color azul que indica el estado del dispositivo, el zumbador pasivo y el motor de vibración que realizarán el feedback a través de un aviso sonoro o vibratorio.

Además, esta versión cuenta con un botón de calibración para mayor precisión. Este botón deberá presionarse cuando el dispositivo se encienda cuando se coloque por primera vez en la espalda del usuario con una postura correcta o en el caso de que el usuario sea consciente de que el dispositivo no esté funcionando correctamente.

El sensor detecta que la persona se ha inclinado más de 15° de su posición, se detecta una mala postura y se ofrece el feedback sonoro (melodía modificable empleando el zumbador) y vibratorio (motor de vibración) para que el usuario modifique su postura, una vez la persona recupera su postura normal correcta el dispositivo deja de sonar y vibrar, indicando que se encuentra en una postura correcta. Inicialmente se ha creado un umbral de 15°, pero se trata de un umbral modificable.

Se pueden observar los componentes y las conexiones realizadas en las siguientes imágenes:

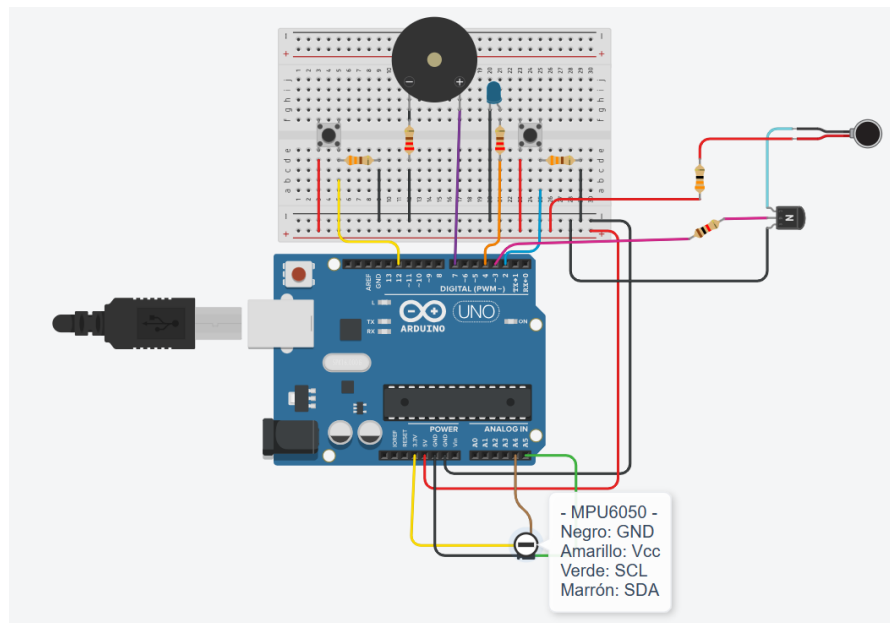


Figura B.3: Diagrama de la segunda versión del prototipo, empleando el módulo MPU-6050

El código empleado para el funcionamiento de esta segunda versión es el siguiente:

```

1 // Led simple + Boton + Zumbador + motor + MPU6050
2 // Naiara Gadea Rodriguez Gomez
3 //
4
5 // Librerias I2C para controlar el mpu6050
6 // las librerias MPU6050.h, I2Cdev.h y Wire.h
7 #include "I2Cdev.h"
8 #include "MPU6050.h"
9 #include "Wire.h"
10
11 // La direccion del MPU6050 puede ser 0x68 o 0x69,
12 // dependiendo
13 // del estado de AD0. Si no se especifica , 0x68 estara
14 // implicito
15
16 // Se crea la variable del sensor. En este caso se esta
17 // trabajando con 0x68. Si se quiere trabajar con 0x69 hay
18 // que poner MPU6050 sensor(0x69)

```

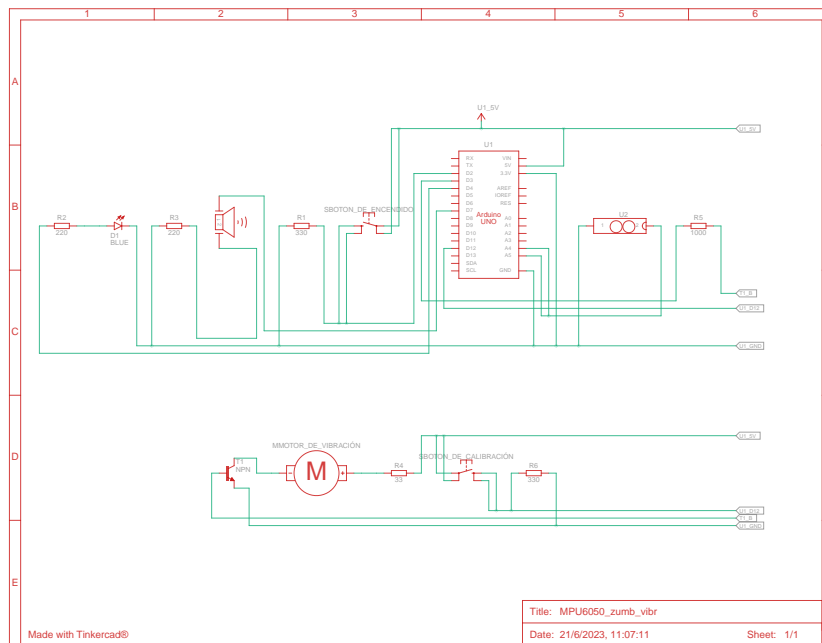


Figura B.4: Diagrama de las conexiones realizadas para la implementación de la segunda versión, siendo el módulo MPU-6050, U2.

```

15 MPU6050 sensor(0x68);
16
17 // Valores RAW (sin procesar) del acelerometro y giroscopio
   en los ejes x,y,z
18 int ax, ay, az; // acelerometro
19 int gx, gy, gz; // giroscopio
20
21 float accel_ang_x, accel_ang_y, accel_ang_z; // Variables
   correspondientes a los angulos de inclinacion. Son los
   que nos interesan principalmente.
22
23 int boton1 = 2; // Seleccin del pin del boton (pin
   digital)
24 int estado1; // Estado del boton ON/OFF
25 int led = 4; // Seleccin del pin del led (pin digital)
26 int zum = 7; // Seleccin del pin del zumbador

```



```
27 int motor = 3; //Selección del pin del motor —Cuando se
    tenga el motor—
28 int boton2 = 12; //Botón de calibrado
29 int estado2; //Estado del botón de calibrado
30
31 // Otras variables
32 bool calibrar = false; // Variable que indica cuando seguir
    o no calibrando.
33 bool origen = true; // Para guardar los datos del primer
    valor tras la calibración.
34
35 //Variables usadas por el filtro pasa bajos
36 long f_ax, f_ay, f_az;
37 int p_ax, p_ay, p_az;
38 long f_gx, f_gy, f_gz;
39 int p_gx, p_gy, p_gz;
40 int counter=0;
41
42 //Valor de los offsets
43 int ax_o, ay_o, az_o;
44 int gx_o, gy_o, gz_o;
45
46 float first_x, first_y, first_z; // Si no funciona
    correctamente desde el principio, hay que calibrar, o
    inicializar desde el principio.
47
48 float sum_ax, sum_ay, sum_az;
49 float sum_gx, sum_gy, sum_gz;
50 float media_ax, media_ay, media_az;
51 float media_gx, media_gy, media_gz;
52 int cont = 0;
53
54
55 void setup() {
56     // put your setup code here, to run once:
57     Serial.begin(9600);
58     pinMode(led, OUTPUT); // inicialización del pin digital
        led.
59     pinMode(boton1, INPUT); // inicialización del pin digital
        del botón.
```

```
60 pinMode(boton2, INPUT); // init del pin boton de calibrado
61 pinMode(zum, OUTPUT); // inicializacion del pin digital
    del zumbador
62 pinMode(motor, OUTPUT); // inicializacion del pin digital
    del motor de vibracion
63
64 Wire.begin(); //Iniciando I2C
65 sensor.initialize(); //Iniciando el sensor
66
67 // Al inicializar el sensor los rangos seran:
68 // Acelerometro: -2g a +2g
69 // Giroscopio: -250deg/sec a +250deg/sec
70
71 // Se comprueba que se ha inicializado correctamente.
72 if (sensor.testConnection()) Serial.println("Sensor
    iniciado correctamente");
73 else Serial.println("Error al iniciar el sensor");
74
75 // Leer los offset los offsets anteriores (iniciales)
76 ax_o=sensor.getXAccelOffset();
77 ay_o=sensor.getYAccelOffset();
78 az_o=sensor.getZAccelOffset();
79 gx_o=sensor.getXGyroOffset();
80 gy_o=sensor.getYGyroOffset();
81 gz_o=sensor.getZGyroOffset();
82
83 Serial.println("Offsets Originales:");
84 Serial.print(ax_o); Serial.print("\t");
85 Serial.print(ay_o); Serial.print("\t");
86 Serial.print(az_o); Serial.print("\t");
87 Serial.print(gx_o); Serial.print("\t");
88 Serial.print(gy_o); Serial.print("\t");
89 Serial.print(gz_o); Serial.println("\t");
90 }
91
92 void loop() {
93     // put your main code here, to run repeatedly:
94
95     if (estado1 == LOW && digitalRead(boton1)){
```

```
196 // Si se presiona el boton se ENCIENDE EL DISPOSITIVO
197 digitalWrite(led, HIGH); // Encendido
198 delay(1000); // Durante 1 segundo (1000 ms)
199 estado1 = HIGH; // Cambia el estado del boton a
    encendido.
100
101 }
102 if(estado1 == HIGH){
103     // Si el sensor tilt hace contacto, el usuario tiene
    una mala postura y el dispositivo manda un aviso,
    musical o de vibracion.
104
105     // Leer las aceleraciones y velocidades angulares y se
    guardan en sus variables
106     sensor.getAcceleration(&ax, &ay, &az);
107     sensor.getRotation(&gx, &gy, &gz);
108
109     if (digitalRead(boton2) && estado2==LOW){
110         calibrar = true;
111     }
112
113     if (calibrar){
114         //Serial.println("Se ha dado el boton, calibrar");
115         calibracion();
116         origen = true;
117     }else{
118         if(origen){
119             first_x=accel_ang_x;
120             first_y = accel_ang_y;
121             first_z = accel_ang_z;
122
123             origen = false;
124         }
125
126         if (cont < 10){
127             lecturas();
128             sum_ax = sum_ax + accel_ang_x; // queremos el
    angulo de inclinacion.
129             sum_ay = sum_ay + accel_ang_y; // queremos el
    angulo de inclinacion.
```

```
130         sum_az = sum_az + accel_ang_z; // queremos el
        angulo de inclinacion.
131
132         cont++;
133
134     } else {
135         media_ax = sum_ax/10;
136         Serial.print("Valor medio de inclinacion en X cada
5 s: "); Serial.println(media_ax);
137         media_ay = sum_ay/10;
138         Serial.print("Valor medio de inclinacion en Y cada
5 s: "); Serial.println(media_ay);
139         media_az = sum_az/10;
140         Serial.print("Valor medio de inclinacion en Z cada
5 s: "); Serial.println(media_az);
141         // reestablecemos los sumatorios
142         sum_ax = 0;
143         sum_ay = 0;
144         sum_az = 0;
145         cont = 0;
146
147         if(abs(first_x - media_ax)<15 & abs(first_y -
media_ay)<15 & abs(first_z - media_az)<15){
148             Serial.println("Buena postura");
149             noTone(zum); // NO se produce alerta
150             digitalWrite(motor, LOW); // Paramos el motor —
Cuando se tenga motor.
151         } else {
152             Serial.println("Mala postura, ponte recto");
153             digitalWrite(motor, HIGH); // vibracion — Cuando se
tenga motor.
154             //delay(500); // delay one second — Cuando se
tenga motor.
155             melodía(); // Se produce alerta
156
157         }
158
159     }
160
161 }
```

```
162
163 // Si se presiona el boton se apaga el dispositivo
164 if (digitalRead(boton1)){
165     digitalWrite(led , LOW); // Apagado
166     delay(1000); // Durante 1 segundo (1000ms)
167     estado1 = LOW; // Cambia el estado del boton a
        apagado.
168 }
169
170 }
171
172 }
173
174 void lecturas(){
175 // Si queremos pasar las lecturas del acelerometro a m/s
    ^2 hay que multiplicar las lecturas por (9.81/16384.0).
176 // En la componente Z se deben encontrar mediciones
    aproximadas a los 9.8 m/s^2
177 // Si queremos pasar las lecturas del giroscopio a deg/s
    (grados/s) hay que multiplicar las lecturas por
    (250.0/32768.0)
178 //Mostrar las lecturas separadas por un [tab]
179 Serial.print("a[x y z]    Incl X    Incl Y  g[x y z]:\t");
180 Serial.print(ax*(9.81/16384.0)); Serial.print("\t"); //
    En m/s^2
181 Serial.print(ay*(9.81/16384.0)); Serial.print("\t"); //
    En m/s^2
182 Serial.print(az*(9.81/16384.0)); Serial.print("\t"); //
    En m/s^2
183
184 accel_ang_x=atan(ax/sqrt(pow(ay,2) + pow(az,2)))
    *(180.0/3.14); // En angulos de inclinacion
185 Serial.print(accel_ang_x); Serial.print("\t");
186 accel_ang_y=atan(ay/sqrt(pow(ax,2) + pow(az,2)))
    *(180.0/3.14); // En angulos de inclinacion
187 Serial.print(accel_ang_y); Serial.print("\t");
188 accel_ang_z=atan(az/sqrt(pow(ax,2) + pow(ay,2)))
    *(180.0/3.14); // En angulos de inclinacion
189 Serial.print(accel_ang_z); Serial.print("\t");
190
```

```

191 // Esto no es necesario
192 Serial.print(gx*(250.0/32768.0)); Serial.print("\t"); //
    En grados/s
193 Serial.print(gy*(250.0/32768.0)); Serial.print("\t"); //
    En grados/s
194 Serial.println(gz*(250.0/32768.0)); // En grados/s
195
196 delay(500); // Mide cada 0,5 segundos, la media para
    comprobar que se encuentra en una buena postura sera
    cada 10 medidas (cada 5 segundos)
197
198 }
199
200 void calibracion(){
201     // Filtrar las lecturas
202     f_ax = f_ax-(f_ax>>5)+ax;
203     p_ax = f_ax>>5;
204
205     f_ay = f_ay-(f_ay>>5)+ay;
206     p_ay = f_ay>>5;
207
208     f_az = f_az-(f_az>>5)+az;
209     p_az = f_az>>5;
210
211     f_gx = f_gx-(f_gx>>3)+gx;
212     p_gx = f_gx>>3;
213
214     f_gy = f_gy-(f_gy>>3)+gy;
215     p_gy = f_gy>>3;
216
217     f_gz = f_gz-(f_gz>>3)+gz;
218     p_gz = f_gz>>3;
219
220     //Cada 100 lecturas corregir el offset
221     if (counter==100){
222         //Serial.print("Calibrando...");
223         //Mostrar las lecturas separadas por un [tab]
224         Serial.print("promedio:"); Serial.print("\t");
225         Serial.print(p_ax); Serial.print("\t");
226         Serial.print(p_ay); Serial.print("\t");

```

```
227 Serial.print(p_az); Serial.print("\t");
228 Serial.print(p_gx); Serial.print("\t");
229 Serial.print(p_gy); Serial.print("\t");
230 Serial.println(p_gz);
231 //Basicamente, se modifica constantemente el offset
para que sea 0, como medida real.
232 // Una vez se observen valores cercanos a 0 se debe
desconectar o reiniciar el Arduino, y asi ya se
encuentra calibrado el dispositivo.
233 //Calibrar el acelerometro a 1g en el eje z (ajustar el
offset)
234 if (p_ax>0) ax_o--;
235 else {ax_o++;}
236
237 if (p_ay>0) ay_o--;
238 else {ay_o++;}
239
240 if (p_az-16384>0) az_o--;
241 else {az_o++;}
242
243 sensor.setXAccelOffset(ax_o);
244 sensor.setYAccelOffset(ay_o);
245 sensor.setZAccelOffset(az_o);
246
247 //Calibrar el giroscopio a 0deg/s en todos los ejes (
ajustar el offset)
248 if (p_gx>0) gx_o--;
249 else {gx_o++;}
250
251 if (p_gy>0) gy_o--;
252 else {gy_o++;}
253
254 if (p_gz>0) gz_o--;
255 else {gz_o++;}
256
257 sensor.setXGyroOffset(gx_o);
258 sensor.setYGyroOffset(gy_o);
259 sensor.setZGyroOffset(gz_o);
260
261 counter=0;
```

```

262
263     if (p_ax>-10 & p_ax<10 & p_ay>-10 & p_ay<10 & p_az
        >16374 & p_az<16394 & p_gx>-10 & p_gx<10 & p_gy>-10 &
        p_gy<10 & p_gz>-10 & p_gz<10){
264         Serial.println("DISPOSITIVO CALIBRADO!!!");
265         calibrar = false;
266     }else{
267         Serial.println("Calibrando...");
268     }
269 }
270 counter++;
271
272 }
273
274
275 // Definimos las notas
276 int Do = 261;
277 int Re = 293;
278 int Mi = 329;
279 int Fa = 349;
280 int Sol = 392;
281 int La = 440;
282 int Si = 493;
283
284 void melodia(){
285     // Escala de musica con el zumbador
286     tone(zum, Fa, 500);
287     delay(700);
288     tone(zum, Sol, 500);
289     delay(700);
290
291 }

```

Esta versión proporciona un resultado satisfactorio, porque realiza su función correctamente, aunque hay que calibrarlo cada vez que se enciende el dispositivo y la calibración puede llevar varios minutos. Además, habría que modificar el ángulo de aviso que se quiere tener de umbral de correcta o incorrecta postura. También se puede modificar el tiempo de espera que se quiere tener para dar el aviso de una postura incorrecta, en este caso se encuentra a 5 segundos.

Por otro lado, hay que pensar en cómo posicionar de forma correcta el dispositivo con el sensor en la espalda del usuario, ya que el sensor debe estar en horizontal. Si no se coloca correctamente el sensor habría que modificar los ejes en el código, ya que no detectara de forma correcta los ángulos y esto puede ser algo complejo.

IDEAS: una base para la calibración, ya que indicas que debe estar en una superficie plana, igual algún tipo de superficie con indicadoresde nivel

B.3. Manuales y/o Demostraciones prácticas

si a la vez que preparas la demostración grabas los vídeos y los incluyes puede ser bastante valorado .. además de al tribunal puedes decir que le servirá al usuario para entender como debe realizar las acciones.

Las demostraciones prácticas se realizarán en base a una serie de pasos que se deberán realizar. Si estos pasos transcurren sin ningún ontratiempo se obtendrá una ejecución exitosa y observarán y analizarán los resultados.

Incluir imágenes de cada paso al utilizar cada una de las versiones.

Se incluirá la dirección a los videos de ejemplo.

Demostración de la primera versión del prototipo

En primer lugar el usuario se deberá colocar el prototipo, si es necesario puede solicitar ayuda a otra persona.

Una vez se encuentre colocado el dispositivo se encenderá presionando el botón de encendido, tras encender el prototipo se encenderá un led azul. El dispositivo empezará a controlar la postura del usuario.

Para poder comprobar su correcto funcionamiento el usuario se inclinará hacia delante y el dispositivo al detectar un postura incorrecta dará feedback a través de sonido, y en el caso de tener añadido en el dispositivo un motor de vibración, también vibrará. Tras unos segundos el usuario volverá a una posición correcta, al corregir su postura el dispositivo dejará de sonar y de vibrar.

Se volverá a comprobar que funciona correctamente haciendo que el usuario se vuelva a inclinar hacia delante, el dispositivo volverá a indicar que se ha establecido una mala postura y el usuario gracias al aviso sabrá que deberá corregir su postura, una vez el usuario haya corregido su postura el dispositivo dejará de pitar.

Se realizarán pruebas en las que el usuario se agache o salte o ande para ver cuando da problemas este dispositivo, debido a su sensibilidad a vibraciones. Los problemas detectados son aquellos que se intentarán solucionar en la segunda version donde se empleará un sensor más complejo.

Por último, el dispositivo se apagará empleando el botón de encendido/ apagado.

Demostración de la primera versión del prototipo

Para la demostración de esta segunda versión en la que se emplea un sensor más complejo y preciso, el sensor MPU6050, el primer paso es que el usuario se coloque el prototipo.

Una vez el dispositivo se encuentre correctamente colocado el sistema pedirá el ángulo de aviso y el tiempo de aviso, si el usuario no indica se toma como valor por defecto 15º y 5 segundos. Se debe encender el dispositivo presionando el botón de encendido. Al encender el prototipo se encenderá un led de color azul y el dispositivo comenzará a controlar la postura. Si es la primera vez que se conecta al portatil el dispositivo pitará por falta de calibración. En ese momento el usuario deberá presionar el botón de calibración para calibrar el dispositivo. Durante la calibración el dispositivo no debe moverse y puede tardar varios minutos una vez calibrado, el dispositivo pitará 3 veces o vibrará tres veces.

Tras la calibración el prototipo comenzará el control postural de forma correcta.

Para comprobar su funcionamiento la persona se inclinará ligeramente hacia delante, si supera el umbral, en este caso 15º, el dispositivo pitará y vibrará a modo de feedback. La persona gracias al biofeedback corregirá su postura, una vez adoptada una postura correcta el dispositivo dejará de emitir sonido y vibración. Se volverá a realizar una medición de prueba pero esta vez el usuario se encontrará andando.

El usuario se inclinará de nuevo ligeramente y el dispositivo indicará una mala postura, la persona corregirá su postura tras conocer su estado gracias al dispositivo, y una vez corregida la postura el dispositivo dejará de emitir la alerta.

Se puede probar la mayor precisión de esta versión realizando los movimientos que perturbaban en la versión anterior, como agacharse o saltar donde el sensor deberá no deberá emitir alerta.

Queda comprobado que con esta versión se obtiene un dispositivo más preciso y personalizado y, además, se puede emplear tanto para uso en personas sentadas o en movimiento (andando).

Por último, se deberá apagar el dispositivo con el botón de encendido/apagado una vez ya no se requiera el control postural.

Apéndice C

Manual del desarrollador / programador / investigador.

C.1. Estructura de directorios

Eliminar aquellos anexos que no se hayan realizado o entregado en la memoria final

En el repositorio de GitHub encontraremos los siguientes ficheros y directorios:

- **Carpeta img:** carpeta dónde se incluyen todas las imágenes que se han empleado en el desarrollo del proyecto.
- **Carpeta tex:**
 - **1__objetivos.tex:** documento LaTeX que contiene la información acerca de los objetivos.
 - **2__introduccion.tex:** documento LaTeX que contiene la información acerca de la introducción del trabajo, se incluyen conceptos teóricos básicos y el estado del arte.
 - **3__metodologia.tex:** documento LaTeX que contiene la información acerca de la metodología empleada, dónde se incluyen descripción de los datos con los que se trabajan y técnicas y herramientas.
 - **4__conclusiones.tex:** documento LaTeX que contiene las conclusiones del proyecto, se puede encontrar un resumen de resultados, discusión y aspectos relevantes.

- **5_lineas_futuras.tex**: documento LaTeX que contiene la información acerca de las líneas de trabajo futuras.
- **A_planificacion.tex**: documento LaTeX que contiene información del anexo A, donde se incluye la planificación temporal, la planificación económica y la viabilidad legal en España del trabajo.
- **B_manual_usuario.tex**: documento LaTeX que contiene la información del anexo B que incluye los requisitos funcionales y no funcionales, la instalación y puesta en marcha y manuales o demostraciones prácticas.
- **C_manual_programador.tex**: documento LaTeX que contiene la información del anexo C que contiene la estructura de los directorios entregados, la información acerca de la ejecución del proyecto y las instrucciones de mejora del proyecto.
- **D_datos.tex**: documento LaTeX que contiene la información acerca de los datos utilizados en el proyecto.
- **E_diseno.tex**: documento LaTeX que contiene la información acerca del diseño del prototipo realizado.
- **F_requisitos.tex**: documento LaTeX que contiene la información acerca los casos de uso definidos.
- **G_experimental.tex**: documento LaTeX que contiene la información acerca del estudio experimental realizado.
- **readme.txt**:
- **Carpeta videos**: carpeta dónde se encuentran los vídeos de demostración del proyecto.
- **Carpeta Arduino**: carpeta dónde se encuentran los programas de arduino empleados en las distintas versiones del prototipo del proyecto.
 - **ProgramaV1.ino**: Programa que se ha de cargar en la placa de Arduino de la primera versión del prototipo del dispositivo de control postural, dónde se emplea el sensor SW520D.
 - **ProgramaV2.ino**: Programa que se ha de cargar en la placa de Arduino de la segunda versión del prototipo del dispositivo de control postural, dónde se emplea el módulo MPU-6050.
- **README.md**: archivo de presentación del repositorio de GitHub.
- **anexos.pdf**: documento PDF que contiene los anexos completos.
- **anexos.tex**: archivo LaTeX que contiene la estructura del documento pdf de los anexos.

- **bibliografia.bib**: archivo que recoge la bibliografía de la memoria.
- **bibliografiaAnexos.bib**: archivo que recoge la bibliografía de los anexos.
- **memoria.pdf**: documento PDF que contiene la memoria completa.
- **memoria.tex**: archivo LaTeX que contiene la estructura del documento de la memoria.

C.2. Compilación, instalación y ejecución del proyecto

En caso de ser necesaria esta sección, porque la compilación o ejecución no sea directa.

Para utilizar los ficheros de código de arduino se deberá utilizar el Arduino IDE y tener montado el prototipo del dispositivo e introducir en el arduino el código necesario para que el dispositivo funcione.

YA SE HA EXPLICADO EN EL B2.INSTALACIÓN Y PUESTA EN MARCHA.
VER SI MOVER

C.3. Pruebas del sistema

Esta sección puede ser opcional.

Se ha realizado una encuesta de validación por parte del usuario.

Puede tratarse de validación de la interfaz por parte de los usuarios, mediante encuestas o similar o validación del funcionamiento mediante pruebas unitarias.

C.4. Instrucciones para la modificación o mejora del proyecto.

Instrucciones y consejos para que el trabajo pueda ser mejorado en futuras ediciones.

Se puede crear un prototipo más robusto, utilizar un módulo Bluetooth, crear la aplicación del dispositivo para la mejora de la interacción del usuario con el dispositivo.

Durante el desarrollo de la primera versión del proyecto con el sensor más sencillo, el sensor SW250D, se observó que no tiene una gran precisión para esta aplicación, además es necesario crear una carcasa donde colocar el sensor de una determinada forma para que cumpla de manera correcta su función.

Mejora haciendo que la versión 2 del prototipo con el módulo MPU6050 se calibre automáticamente, sin tener que clicar ningún botón.

Realizar un prototipo con el módulo MPU9250 o similar porque será similar que el MPU6050 pero más preciso y sin problemas de deriva durante el uso continuado.

Este proyecto es básicamente un prototipo, por lo que hay bastantes aspectos en los que mejorar en el futuro.

Durante el desarrollo y las pruebas de la primera fase, donde se empleaba el sensor SW520D, se pudo observar que, aunque no se trata de un sensor de gran precisión se pudieron analizar las posibilidades de mejora y las ventajas que suponía trabajar con el sensor de mayor precisión en la fase 2.

Apéndice D

Descripción de adquisición y tratamiento de datos

Va fuera creo simplemente con la información del punto de descripción de los datos de la memoria será suficiente.

cuando hables de los datos, como los habrás generado y recogido tu habla de ELABORACIÓN PROPIA.

D.1. Descripción formal de los datos

Tablas, imágenes, señales, secuencias de ADN...

D.2. Descripción clínica de los datos.

Descripción y explicaciones clínicas del significado o interpretación de los datos.

Apéndice E

Manual de especificación de diseño

Si es necesario. Se incluirán los planos de las distintas versiones de arduino.

Planos (Si procede) Diseño arquitectónico (Si procede) Diagrama de clases, diagrama de despliegue

E.1. Planos

Si procede

E.2. Diseño arquitectónico

Si procede.

Diagramas de clases, diagramas de despliegue ...

Especificación de Requisitos

F.1. Diagrama de casos de uso

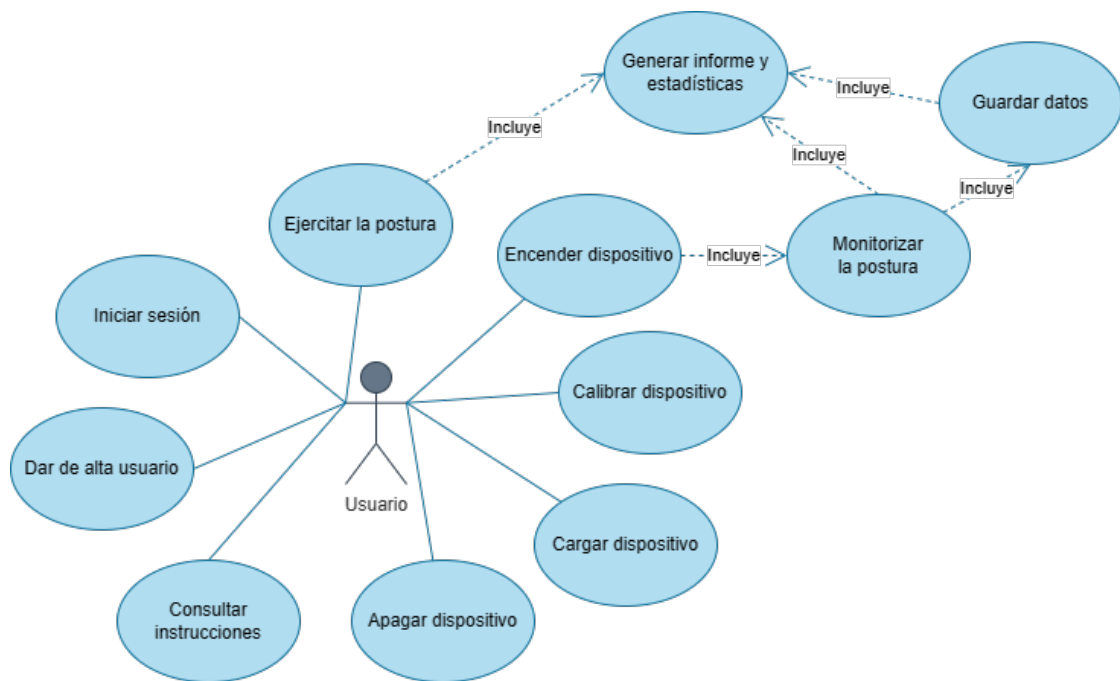


Figura F.1: Diagrama de casos de uso

F.2. Explicación casos de uso.

CS-01	<Encender dispositivo>
Versión	1.0
Autor	Naiara Gadea Rodríguez Gómez
Descripción	Puesta en marcha del dispositivo hardware y su conexión con la aplicación. El dispositivo no deberá estar siempre conectado con la aplicación, pero si se conecta a la aplicación se puede obtener más información.
Secuencia Normal	<ol style="list-style-type: none"> 1. Se coloca el dispositivo en contacto sobre la piel, como se indica en las instrucciones. 2. Se enciende el dispositivo con el botón ON/OFF. 3. Cuando se encienda un led verde indicará que el dispositivo está encendido. 4. Se conecta el dispositivo al dispositivo que tiene instalada la aplicación vía Bluetooth o vía WIFI. 5. El usuario accede a la aplicación software instalada en el dispositivo móvil o en el ordenador, que deberán estar conectados a la misma red WIFI o Bluetooth para permitir la comunicación. 6. Una vez se realiza la conexión, el led verde del dispositivo cambia a color a azul. Además, en la aplicación aparece el dispositivo como conectado.
Frecuencia	Alta
Importancia	Alta
Urgencia	Alta
Requisitos Funcionales Relacionados	RF-02
Casos de uso relacionados	CS-02, CS-03, CS-04, CS-07, CS-08, CS-10

Tabla F.1: CU-01. Encender dispositivo.

CS-02	<Apagar dispositivo>
Versión	1.0
Autor	Naiara Gadea Rodríguez Gómez
Descripción	Apagado del dispositivo.
Secuencia Normal	<ol style="list-style-type: none"> 1. Se apaga el dispositivo con el botón ON/OFF. 2. Una vez el dispositivo se encuentre apagado el led azul o verde se apagará. 3. El usuario se puede quitar el dispositivo y puede cargarlo.
Frecuencia	Alta
Importancia	Alta
Urgencia	Media
Requisitos Funcionales Relacionados	RF-02
Casos de uso relacionados	CS-01, CS-03
Comentarios	El dispositivo también se apaga cuando se acaba la batería, en cuyo caso el caso de uso comienza en el paso 2.

Tabla F.2: CU-02. Apagar dispositivo.

CS-03	<Cargar dispositivo>
Versión	1.0
Autor	Naiara Gadea Rodríguez Gómez
Descripción	Carga de la batería del dispositivo empleando la estación de carga.
Secuencia Normal	<ol style="list-style-type: none"> 1. Actúa el CS-02. 2. Una vez que el dispositivo se encuentre apagado el usuario coloca el dispositivo sobre la estación de carga enchufada a la corriente. 3. Se enciende un led rojo intermitente. 4. Cuando la batería del dispositivo se encuentre completamente cargada, el led rojo intermitente deja de ser intermitente. 5. El dispositivo está completamente cargado y puede desconectar la estación de carga de la corriente o desconectar el dispositivo. 6. El dispositivo estará listo para su uso.
Frecuencia	Alta
Importancia	Alta
Urgencia	Media
Requisitos Funcionales Relacionados	RF-08
Casos de uso relacionados	CS-02

Tabla F.3: CU-03. Cargar dispositivo.

CS-04	<Calibrar dispositivo>
Versión	1.0
Autor	Naiara Gadea Rodríguez Gómez
Descripción	Se recalculan los valores en reposo para que el dispositivo devuelva medidas lo más precisas posible. Se debe calibrar el dispositivo cada cierto tiempo para que los valores medidos sean lo más precisos posible.
Secuencia Normal	<ol style="list-style-type: none"> 1. El dispositivo debe estar encendido y sobre una superficie plana. 2. Se presionará el botón de calibrado. Sobre el dispositivo o de la propia aplicación. 3. El sistema recalibra los valores medidos por el dispositivo. 4. Una vez el dispositivo se haya calibrado correctamente se encenderá 3 veces el led verde o azul, en función si la calibración se ha realizado a través de la aplicación o directamente sobre el dispositivo. 5. El dispositivo se encuentra correctamente calibrado y listo para su uso.
Frecuencia	Baja
Importancia	Alta
Urgencia	Baja
Requisitos Funcionales Relacionados	RF-01, RF-04
Casos de uso relacionados	CS-01, CS-07

Tabla F.4: CU-04. Calibrar dispositivo.

CS-05	<Dar de alta usuario>
Versión	1.0
Autor	Naiara Gadea Rodríguez Gómez
Descripción	El usuario crea una cuenta para cada recopilar los datos recopilados por el dispositivo. Se creará una cuenta por persona que vaya a utilizar el dispositivo.
Secuencia Normal	<ol style="list-style-type: none"> 1. El caso de uso comienza cuando el facultativo clica en registrar. 2. El sistema solicita el nombre de usuario. 3. El usuario introduce el nombre de usuario. 4. El sistema solicita los nombre y apellidos del usuario. 5. El usuario introduce su nombre y apellidos. 6. El sistema solicita una contraseña. 7. El usuario introduce una contraseña. 8. El sistema guarda y almacena los datos introducidos. Se ha creado la cuenta del usuario.
Frecuencia	Baja
Importancia	Media
Urgencia	Media
Requisitos Funcionales Relacionados	RF-01, RF-03
Casos de uso relacionados	CS-06

Tabla F.5: CU-05. Dar de alta usuario.

CS-06	<Iniciar sesión>
Versión	1.0
Autor	Naiara Gadea Rodríguez Gómez
Descripción	El usuario inicia sesión en la aplicación para poder acceder a sus datos, informes y ejercicios. El usuario tiene la opción de mantener la sesión iniciada.
Secuencia Normal	<ol style="list-style-type: none"> 1. El sistema solicita nombre de usuario y contraseña. 2. El usuario introduce su nombre de usuario y contraseña. 3. El sistema pregunta si desea mantener la sesión abierta. 4. El usuario introduce si desea mantener la sesión abierta. 5. El sistema compara con su base de datos y si encuentra coincidencia accede a los datos del usuario. En caso de que no encuentre al usuario el sistema imprime por pantalla 'Contraseña o usuario incorrectos'. 6. El sistema muestra las estadísticas y datos del paciente.
Frecuencia	Media
Importancia	Alta
Urgencia	Media
Requisitos Funcionales Relacionados	RF-01, RF-03, RF-06
Casos de uso relacionados	CS-05, CS-07, CS-08, CS-09, CS-10
Comentarios	Si el usuario ha indicado que desea mantener la sesión abierta, la sesión se mantiene abierta, aunque se cierre la aplicación. El usuario deberá cerrar manualmente su sesión.

Tabla F.6: CU-06. Iniciar sesión.

CS-07	<Realizar monitoreo de la postura>
Versión	1.0
Autor	Naiara Gadea Rodríguez Gómez
Descripción	Una vez el dispositivo se encuentre encendido se realiza la monitorización de la postura. Si se detecta una mala postura el dispositivo emitirá una vibración. En el caso de que el dispositivo se encuentre conectado con la aplicación los datos.
Secuencia Normal	<ol style="list-style-type: none"> 1. El caso de uso comienza cuando se enciende el usuario enciende el dispositivo. 2. Pasos del CU El sistema obtiene los datos proporcionados por el sensor del dispositivo. 3. El sistema identifica entre una buena o mala postura, gracias al algoritmo. 4. Si se detecta mala postura el dispositivo emita una señal vibratoria para que el usuario modifique su postura. 5. Interviene el CS-08. 6. El caso de uso finaliza cuando se apaga el dispositivo o cuando se gasta la batería del dispositivo.
Frecuencia	Alta
Importancia	Alta
Urgencia	Alta
Requisitos Funcionales Relacionados	RF-02, RF-04, RF-05
Casos de uso relacionados	CS- 01, CS-04, CS-06, CS-08, CS-09, CS-10

Tabla F.7: CU-07. Realizar monitoreo de la postura.

CS-08	<Guardar datos>
Versión	1.0
Autor	Naiara Gadea Rodríguez Gómez
Descripción	Se archivan los datos recopilados a través del dispositivo. Los datos se van guardando en tiempo real si el dispositivo se encuentra conectado a la aplicación. dices que 'puede ser en tiempo real' deberá de explicar en que casos no es en tiempo real, o que pasa con el almacenamiento.
Secuencia Normal	<ol style="list-style-type: none"> 1. El sistema registra los datos enviados vía Bluetooth o WIFI por el dispositivo. 2. Interviene el CS-07. 3. El sistema guarda los datos y el informe generado en la base de datos integrada. Que se podrán consultar posteriormente o en tiempo real.
Frecuencia	Alta
Importancia	Alta
Urgencia	Alta
Requisitos Funcionales Relacionados	RF-01, RF-02, RF-04, RF-06
Casos de uso relacionados	CS-06, CS-07, CS-09

Tabla F.8: CU-08. Guardar datos.

CS-09	<Generar informe>
Versión	1.0
Autor	Naiara Gadea Rodríguez Gómez
Descripción	A partir de los datos recopilados del sensor, se realiza un informe que puede ser en tiempo real que incluye toda la información relevante y estadísticas calculadas en función de los datos. CS-09 dices que 'puede ser en tiempo real' deberá de explicar en que casos no es en tiempo real, o que pasa con el almacenamiento.
Secuencia Normal	<ol style="list-style-type: none"> 1. El caso de uso comienza tras el CS-08. 2. El sistema incluye los datos obtenidos. 3. El sistema crea varias estadísticas utilizando gráficas que resumen visualmente los datos recogidos y la evolución del paciente. 4. Se puede obtener un informe en formato pdf.Incluir donde se crea, como se crea..
Frecuencia	Alta
Importancia	Alta
Urgencia	Alta
Requisitos Funcionales Relacionados	RF-01, RF-02, RF-04, RF-06
Casos de uso relacionados	CS-06, CS-07, CS-08, CS-10

Tabla F.9: CU-09. Generar informe.

CS-10	<Ejercitar la musculatura de la postura>
Versión	1.0
Autor	Naiara Gadea Rodríguez Gómez
Descripción	Se realizan distintos juegos o ejercicios incluidos en la aplicación software, que, mediante la interacción con el dispositivo hardware, permitirán al usuario aumentar y mejorar la musculatura que se necesita para una buena postura. en realizan distintos juegos o ejercicios incluidos en la aplicación softwarezo subdividiría en 1. y 2. con una breve descripción por lo menos de los mas relevantes. auqneu puedes indicar que hay más si no describes todos.
Secuencia Normal	<ol style="list-style-type: none"> 1. El usuario selecciona dentro de la pestaña de juegos y ejercicios, el juego o el ejercicio que desee realizar. 2. El sistema muestra el juego o ejercicio seleccionado. 3. El usuario realiza el juego o ejercicio seleccionado. 4. Una vez finalizado el juego o el ejercicio el sistema vuelve a la pantalla donde se incluyen los ejercicios y juegos disponibles para mejorar la musculatura o la postura.
Frecuencia	Media
Importancia	Baja
Urgencia	Baja
Requisitos Funcionales Relacionados	RF-01, RF-02, RF-04, RF-05
Casos de uso relacionados	CS-01, CS-04, CS-06, CS-07, CS-09, CS-11

Tabla F.10: CU-10. Ejercitar la musculatura de la postura.

CS-11	<Consultar instrucciones de uso>
Versión	1.0
Autor	Naiara Gadea Rodríguez Gómez
Descripción	El usuario puede revisar las instrucciones de uso del dispositivo, y de esa forma puede obtener información para la puesta en marcha para poder monitorizar su postura.
Secuencia Normal	<ol style="list-style-type: none"> 1. El usuario accede a su perfil y clicca sobre el icono ‘?’ donde se puede consultar las instrucciones. 2. El sistema abre una pestaña con las instrucciones de uso que puede seguir el usuario. Las instrucciones serán sencillas y deberán redirigir a un vídeo explicativo de la puesta en marcha y uso del dispositivo. 3. El usuario cierra la ventana de instrucciones cuando deje de necesitar su consulta.
Frecuencia	Baja
Importancia	Baja
Urgencia	Baja
Requisitos Funcionales Relacionados	RF-02, RF-07
Casos de uso relacionados	CS-01, CS-02, CS-03, CS-04, CS-05, CS-06, CS-07, CS-09, CS-10

Tabla F.11: CU-11. Consultar instrucciones de uso.

F.3. Prototipos de interfaz o interacción con el proyecto.

Se ha realizado un prototipo de interfaz de aplicación móvil, en base a las aplicaciones de los dispositivos existentes.

La interfaz creada consta de 6 pantallas principales:

1. Pantalla de inicio de sesión.
2. Pantalla de inicio con información en tiempo real.
3. Pantalla con las estadísticas en forma de gráfica de distintos periodos de tiempo.
4. Pantalla con juegos y ejercicios de mejora de la postura.
5. Pantalla de ajustes del dispositivo conectado.
6. Pantalla del perfil del usuario.

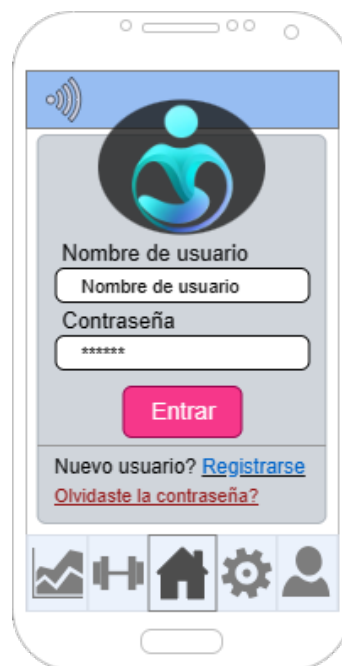


Figura F.2: Pantalla de inicio de sesión.



Figura F.3: Pantalla de inicio con información en tiempo real.

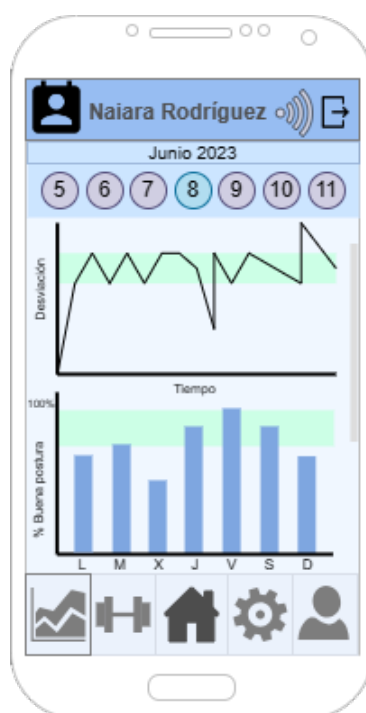


Figura F.4: Pantalla con las estadísticas en forma de gráfica de distintos periodos de tiempo.



Figura F.5: Pantalla con juegos y ejercicios de mejora de la postura.

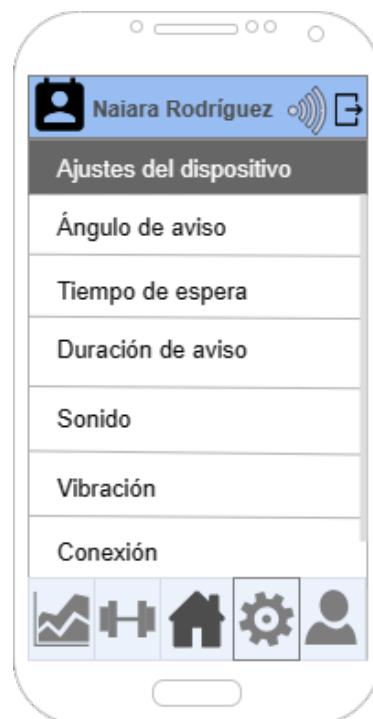


Figura F.6: Pantalla de ajustes del dispositivo conectado.



Figura F.7: Pantalla del perfil del usuario.

Apéndice G

Estudio experimental

FUERA, este trabajo no es un estudio experimental.

podrías indicar las pruebas y los resultados de los usuarios... pero si no lo tienes pensado así puedes decir que podría ir en trabajos futuros, para el seguimiento de resultados del uso que hayan hecho los usuarios y la explotación estadística de los datos...

G.1. Cuaderno de trabajo.

Enumeración de todos los métodos probados con resultados positivos o no.

G.2. Configuración y parametrización de las técnicas.

G.3. Detalle de resultados.

[19]

Bibliografía

- [1] BOE. “Ley 24/2015, de 24 de julio, de Patentes (BOE-A-2015-8328).” En: *Boletín Oficial del Estado* 177 (2015). URL: <https://www.boe.es/buscar/act.php?id=BOE-A-2015-8328>.
- [2] BOE. “Real Decreto Legislativo 1/1996, de 12 de abril, por el que se aprueba el texto refundido de la Ley de Propiedad Intelectual, regularizando, aclarando y armonizando las disposiciones legales vigentes sobre la materia. (BOE-A-1996-8930).” En: *Boletín Oficial del Estado* 97 (1996). URL: <https://boe.es/buscar/act.php?id=BOE-A-1996-8930>.
- [3] AEMPS. *Legislación sobre productos sanitarios*. 2022. URL: <https://www.aemps.gob.es/productos-sanitarios/legislacion-sobre-productos-sanitarios/>.
- [4] BOE. “Real Decreto 1591/2009, de 16 de octubre, por el que se regulan los productos sanitarios. (BOE-A-2009-17606).” En: *Boletín Oficial del Estado* 268 (2009). URL: <https://www.boe.es/buscar/doc.php?id=BOE-A-2009-17606>.
- [5] BOE. “Real Decreto 437/2002, de 10 de mayo, por el que se establecen los criterios para la concesión de licencias de funcionamiento a los fabricantes de productos sanitarios a medida.(BOE-A-2002-10228).” En: *Boletín Oficial del Estado* 128 (2002). URL: <https://www.boe.es/buscar/doc.php?id=BOE-A-2002-10228>.
- [6] EUR-Lex. “Reglamento (UE) 2017/745 del Parlamento Europeo y del Consejo, de 5 de abril de 2017, sobre los productos sanitarios, por el que se modifican la Directiva 2001/83/CE, el Reglamento (CE) n.º 178/2002 y el Reglamento (CE) n.º 1223/2009 y por el que se derogan las Directivas 90/385/CEE y 93/42/CEE del Consejo (Texto pertinente a efectos del EEE.)” En: *Diario*

- Oficial de la Unión Europea* (2017). URL: <https://www.boe.es/buscar/act.php?id=BOE-A-2015-8328>.
- [7] Agencia Estatal Boletín Oficial del Estado. *Código Laboral y de la Seguridad Social*. URL: https://www.boe.es/biblioteca_juridica/codigos/codigo.php?id=93&modo=2¬a=0&tab=2.
- [8] BOE. “Ley 34/2002, de 11 de julio, de servicios de la sociedad de la información y de comercio electrónico.(BOE-A-2002-13758).” En: *Boletín Oficial del Estado* 166 (2002). URL: <https://www.boe.es/buscar/act.php?id=BOE-A-2002-13758>.
- [9] Agencia Estatal Boletín Oficial del Estado. *Código de Comercio y legislación complementaria*. URL: https://www.boe.es/biblioteca_juridica/codigos/codigo.php?id=35&modo=2¬a=0&tab=2.
- [10] BOE. “Ley 7/1996, de 15 de enero, de Ordenación del Comercio Minorista.(BOE-A-1996-1072).” En: *Boletín Oficial del Estado* 15 (1996). URL: <https://www.boe.es/buscar/act.php?id=BOE-A-1996-1072>.
- [11] BOE. “Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales. (BOE-A-2018-16673).” En: *Boletín Oficial del Estado* 294 (2018). URL: <https://www.boe.es/buscar/doc.php?id=BOE-A-2018-16673>.
- [12] EUR-Lex. “Reglamento (UE) 2016/679 del Parlamento Europeo y del Consejo, de 27 de abril de 2016, relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos y por el que se deroga la Directiva 95/46/CE (Reglamento general de protección de datos)”. En: *Diario Oficial de la Unión Europea* (2016). URL: <https://eur-lex.europa.eu/legal-content/ES/TXT/?uri=celex%5C%3A32016R0679>.
- [13] Your Europe. *Marcado CE*. URL: https://europa.eu/youreurope/business/product-requirements/labels-markings/ce-marking/index_es.htm.
- [14] Arduino. *What is Arduino?* URL: <https://www.arduino.cc/en/Guide/Introduction>. (accessed: 17.03.2023).
- [15] Arduino. *UNO R3, Arduino documentation*. URL: <https://docs.arduino.cc/hardware/uno-rev3>. (accessed: 17.03.2023).
- [16] Luis Llamas. *Medir inclinación con Arduino y sensor tilt SW-520d*. 2015. URL: <https://www.luisllamas.es/medir-inclinacion-con-arduino-y-sensor-tilt-sw-520d>. (accessed: 17.05.2023).
- [17] Naylamp Mechatronics. *Tutorial MPU6050, Acelerómetro y Giroscopio*. URL: https://naylampmechatronics.com/blog/45_tutorial-mpu6050-acelerometro-y-giroscopio.html.

- [18] Luis Llamas. *Determinar la orientación con Arduino y el IMU MPU-6050*. URL: <https://www.luisllamas.es/arduino-orientacion-imu-mpu-6050/>.
- [19] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.