

DIRECCIÓN DE PROYECTOS DE DESARROLLO DE SOFTWARE

StraightUp

Fecha: 02/10/2025

Grupo: G5

Salvador Ayala Iglesias, Iciar García Izquierdo, Anastasiia
Sadova, Nicola Stefania Cindea, Aisling Florencio Pimentel

Cliente: Universidad Carlos III de Madrid

Contenido

1. Descripción de la idea creativa	6
1.1. Dispositivo Hardware	6
1.2. Aplicación software	6
2. Datos generales de la empresa que ofrece el proyecto	8
3. Definiciones y acrónimos	9
4. Oferta inicial y presupuesto	10
4.1 Oferta	10
Figura 1. Tabla de perfiles y su participación en cada fase del proyecto	
12	
4.2 Presupuesto	12
Figura 2. Casos de uso de la tira	13
Figura 3. Casos de uso del anillo	13
Figura 4. Casos de uso de BeLine App	14
5. Plan de gestión de la configuración del software (CMP)	15
5.1 Finalidad del Plan	15
5.2 Alcance	15
5.3 Organización y Responsabilidades	16
5.4 Identificación de la configuración	17
5.4.1 Jerarquía preliminar de productos	17
5.4.2 Elementos de configuración	18
5.4.3 Esquema de identificación	24
5.4.4 Definición de las relaciones	24
5.4.5 Definición y establecimiento de líneas de base	26
5.4.6 Definición y creación de bibliotecas de software	27
5.5 Control de cambios	30
5.6 Contabilidad de estado	30
5.7 Auditoría de la configuración	31
6. Plan de calidad	32
6.1 Revisiones previstas	32
6.1.1 Revisión del Estudio de Viabilidad del Sistema	33
6.1.2 Revisión de los Casos de Uso	33
6.1.2.1 Revisión del Diagrama de Casos de Uso	33
6.1.2.2 Revisión de Casos de Uso de Alto Nivel	34
6.1.3 Revisión del Plan de Gestión de la Configuración	35
6.1.4 Revisión de la Estimación	35
6.1.5 Revisión de la Planificación	36
6.1.6 Revisión del Plan de Pruebas	37
6.1.7 Revisión de los Casos de Uso en formato Expandido	38
6.1.8 Revisión del Modelo Conceptual del Análisis	38
6.1.9 Revisión de los Contratos de Operación	39
6.1.10 Revisión del Diagrama de Clases	39
6.1.11 Revisión de los Diagramas de Secuencia	41
6.1.12 Revisión de los Diagramas de Estado	41

6.2 Gestión de Riesgos	42
7. Estimación	46
8. Planificación	47
9. Viabilidad y especificación de requisitos	48
9.1 Estudio de Viabilidad	48
9.1.1 Definición de Requisitos	48
REQUISITOS FUNCIONALES	49
REQUISITOS NO FUNCIONALES	53
9.1.2 Estudio de Alternativas posibles para abordar la solución	55
9.1.3 Valoración de Alternativas	56
9.1.4 Selección de soluciones	56
9.2 Modelo de casos de uso y matriz de trazabilidad	57
9.3 Descripción de alto nivel de los casos de uso	63
9.4 Priorización de casos de uso	63
10. Construcción	65
10.1 Primera iteración	65
10.1.1 Análisis de la primera iteración	65
Modelo conceptual	65
Descripción de casos de uso con formato ampliado	65
Contratos de operación	65
10.1.2 Diseño de la primera iteración	65
Diagramas de secuencia o Wireframe	65
Diagrama de clases o Modelo de datos	65
10.2 Segunda iteración	65
10.2.1 Análisis de la segunda iteración	65
Descripción de casos de uso con formato ampliado	65
Contratos de operación	65
10.2.2 Diseño de la segunda iteración	65
Diagramas de secuencia o Wireframe	65
Diagrama de clases o Modelo de datos	65
11. Ejecución del plan de calidad	66
12. Ejecución del plan de gestión de la configuración	69
13. Conclusiones generales	69
14. Conclusiones individuales de cada miembro	69
15. Declaración de uso de IA generativa	69
15.1. Entregable 1	69

Índice de tablas

1. Descripción de la idea creativa	5
1.1. Dispositivo Hardware	5
1.2. Aplicación software	5
2. Datos generales de la empresa que ofrece el proyecto	7

3. Definiciones y acrónimos	8
4. Oferta inicial y presupuesto	9
4.1 Oferta	9
4.2 Presupuesto	10
5. Plan de gestión de la configuración del software (CMP)	13
5.1 Finalidad del Plan	13
5.2 Alcance	14
5.3 Organización y Responsabilidades	14
5.4 Identificación de la configuración	15
5.4.1 Jerarquía preliminar de productos	15
5.4.2 Elementos de configuración	15
5.4.3 Esquema de identificación	15
5.4.4 Definición de las relaciones	15
5.4.5 Definición y establecimiento de líneas de base	15
5.4.6 Definición y creación de bibliotecas de software	15
5.5 Control de cambios	15
5.6 Contabilidad de estado	15
5.7 Auditoría de la configuración	15
6. Plan de calidad	16
6.1 Revisiones previstas	16
6.1.1 Revisión del Estudio de Viabilidad del Sistema	17
6.1.2 Revisión de los Casos de Uso	17
6.1.2.1 Revisión del Diagrama de Casos de Uso	17
6.1.2.2 Revisión de Casos de Uso de Alto Nivel	18
6.1.3 Revisión del Plan de Gestión de la Configuración	18
6.1.4 Revisión de la Estimación	19
6.1.5 Revisión de la Planificación	20
6.1.6 Revisión del Plan de Pruebas	21
6.1.7 Revisión de los Casos de Uso en formato Expandido	21
6.1.8 Revisión del Modelo Conceptual del Análisis	22
6.1.9 Revisión de los Contratos de Operación	23
6.1.10 Revisión del Diagrama de Clases	23
6.1.11 Revisión de los Diagramas de Secuencia	24
6.1.12 Revisión de los Diagramas de Estado	25
6.2 Gestión de Riesgos	25
7. Estimación	28
8. Planificación	29
9. Viabilidad y especificación de requisitos	30
9.1 Estudio de Viabilidad	30
9.1.1 Definición de Requisitos	30
REQUISITOS FUNCIONALES	31
REQUISITOS NO FUNCIONALES	31
9.1.2 Estudio de Alternativas posibles para abordar la solución	31
9.1.3 Valoración de Alternativas	31

9.1.4 Selección de soluciones	32
9.2 Modelo de casos de uso y matriz de trazabilidad	32
9.3 Descripción de alto nivel de los casos de uso	32
9.4 Priorización de casos de uso	32
10. Construcción	33
10.1 Primera iteración	33
10.1.1 Análisis de la primera iteración	33
Modelo conceptual	33
Descripción de casos de uso con formato ampliado	33
Contratos de operación	33
10.1.2 Diseño de la primera iteración	33
Diagramas de secuencia o Wireframe	33
Diagrama de clases o Modelo de datos	33
10.2 Segunda iteración	33
10.2.1 Análisis de la segunda iteración	33
Descripción de casos de uso con formato ampliado	33
Contratos de operación	33
10.2.2 Diseño de la segunda iteración	33
Diagramas de secuencia o Wireframe	33
Diagrama de clases o Modelo de datos	33
11. Ejecución del plan de calidad	34
12. Ejecución del plan de gestión de la configuración	35
13. Conclusiones generales	35
14. Conclusiones individuales de cada miembro	35
15. Declaración de uso de IA generativa	35

Índice de figuras

Figura 1. Tabla de perfiles y su participación en cada fase del proyecto	11
Figura 2. Casos de uso de la tira	12
Figura 3. Casos de uso del anillo	13
Figura 4. Casos de uso de BeLine App	14

1. Descripción de la idea creativa

Se desea desarrollar un producto hardware y software que mejore el nivel de conciencia del usuario con respecto a su postura corporal en un trabajo de oficina. De esta manera, se pretende minimizar los riesgos músculo-esqueléticos al dotar al usuario de una correcta propiocepción.

1.1. Dispositivo Hardware

El dispositivo se basará en una tira adhesiva dotada de sensores interconectados, la cual se colocará verticalmente en la espalda, siguiendo la columna vertebral desde el cuello hasta la dorsal. De esta forma, se podrán obtener datos que permitirán saber la postura exacta de la espalda. La cinta será delgada, ligera y flexible para reducir cualquier molestia que pudiera causar al usuario al llevarla puesta.

Asimismo, se contará con un anillo inteligente que permitirá medir las constantes vitales del usuario. Este anillo actuará de enlace entre la tira adhesiva y la aplicación alojada en el smartphone del usuario.

Además, el anillo podrá calcular su posición relativa con respecto de los distintos sensores de la tira para estimar la posición de los brazos. De esta manera, se podrá mejorar la postura en su conjunto sin atender únicamente a la espalda.

Ambos dispositivos hardware contarán con una batería interna recargable que permitirá mantener los sensores activos. El anillo se comunicará con la tira por radiofrecuencia, mientras que contará con un módulo bluetooth para transmitir los datos al smartphone.

1.2. Aplicación software

Los datos recibidos del anillo (y de los sensores de la tira, consecuentemente) se procesarán en la aplicación *BeLine*. Dicha aplicación podrá hacer uso de los datos de la tira para:

- Almacenar los datos recibidos en una base de datos alojada en el dispositivo móvil.
- Consultar las constantes vitales a petición del usuario, pudiendo este visualizarlas en la interfaz de la aplicación.
- Procesar los datos de la tira para trazar la postura de la espalda del usuario, pudiendo el usuario ver su postura actual.
- Ofrecer una comparativa entre la postura actual de la espalda y la posición ideal de esta.
- Analizar patrones de postura haciendo uso de los datos posturales y las constantes vitales con respecto del tiempo, pudiendo el usuario consultar dichos patrones.
- Sugerir ejercicios correctivos en función de los patrones detectados que permitan compensar las malas posturas adoptadas. Se hará uso de la posición de la espalda para guiar al usuario durante los ejercicios.

- Detectar la adopción de una mala postura, notificando al usuario con una vibración del anillo y con una notificación en el teléfono en caso de prolongarse la postura.
- Detectar daños severos en la espalda causados por una caída o un golpe y realizar una llamada de emergencia a las autoridades.

Además, la aplicación podrá hacer uso de los datos registrados por el anillo para:

- Registrar datos de entrenamientos y ejercicios realizados.
- Sincronizar los datos con otras aplicaciones de salud.
- Procesar la posición del anillo con respecto de los sensores de la tira para trazar la postura de los brazos con respecto de la espalda, pudiendo el usuario ver dicha postura.

Por último, con la intención de incentivar la adopción de una correcta postura, la aplicación otorgará puntos en función de la postura adoptada y el tiempo que se mantenga. Estos puntos se compartirán con otros usuarios para formar un ranking de mejores posturas.

Además, se enviarán periódicamente mensajes al usuario a través de las notificaciones con datos o consejos para convencer al usuario de la importancia de su postura. Este comportamiento podrá ser ajustado a través de la configuración, en la aplicación.

2.Datos generales de la empresa que ofrece el proyecto

- **Nombre:** *StraightUp*
- **Descripción:** *StraightUp* es una empresa especializada en ofrecer productos y soluciones software orientadas a mejorar la salud física y mental de sus usuarios. La empresa desarrolla aplicaciones, herramientas digitales y productos diseñados para el seguimiento de la actividad física y el bienestar emocional.
- **Misión:** Potenciar la salud de las personas a través de la tecnología, creando soluciones digitales innovadoras y accesibles que faciliten la adopción de estilos de vida más saludables y apoyen proactivamente el cuidado de la salud física y mental.

3.Definiciones y acrónimos

QA Tester (Quality Assurance Tester): profesional responsable de garantizar la calidad del producto mediante pruebas. Su objetivo es detectar defectos y posibles mejoras y verificar que el sistema cumple con los requisitos y las necesidades del usuario.

UX Designer (User Experience Designer): diseñador de la interfaz gráfica de la aplicación que sea fácil de usar, intuitiva y que optimice y mejore la interacción entre el usuario y el sistema.

SQA (Software Quality Assurance): actividades para garantizar la calidad de software a lo largo de todo el ciclo de vida del proyecto. Incluye definición de estándares, realización de auditorías y verificación de entregables.

SCM (Software Configuration Management): proceso de gestión y control de las versiones del software y sus componentes.

MVP (Minimum Viable Product): versión inicial del producto que incluye solo funcionalidades esenciales para validar su funcionamiento y obtener feedback temprano de los usuarios.

SRS (Software Requirements Specification): documento de especificación de los requisitos del software que recoge toda la información detallada de las funcionalidades, restricciones y características técnicas del sistema.

CME (Configuration Management Engineer): responsable de custodiar, mantener y distribuir los elementos de configuración del proyecto.

PR (Pull Request): solicitud de modificación del código fuente a través de la plataforma Github que detalla los cambios con respecto a la versión original.

CCB (Change Control Board): responsable de evaluar y aprobar solicitudes de cambios que puedan afectar al alcance, coste o calidad del proyecto.

4.Oferta inicial y presupuesto

La oferta económica del proyecto BeLine presenta la estimación de esfuerzo y coste asociada al desarrollo de un sistema hardware–software orientado a la mejora de la postura corporal en entornos de oficina.

El proyecto se estructura en cinco fases principales:

- **F1 – Análisis y planificación inicial**
- **F2 – Especificación de requisitos y arquitectura**
- **F3 – Iteración de construcción I**
- **F4 – Iteración de construcción II y validación**
- **F5 – Instalación, entrega y soporte.**

Cada fase cuenta con una dedicación estimada en jornadas equivalentes (FTE) y un coste proporcional al número de horas/hombre necesarias.

El equipo de trabajo incluye perfiles especializados como Project Manager, UX Designer, Hardware y Firmware Engineers, Pentester, QA Tester, Data Algorithm Specialist, Legal Advisor, Backend Developer y Mobile App Developer.

4.1 Oferta

El desarrollo del proyecto BeLine se llevará a cabo considerando la siguiente estructura de perfiles profesionales y niveles de dedicación estimada por fase.

El equipo estará conformado por los siguientes roles:

- **Project Manager:** coordinación y organización general de todo el proyecto y el equipo. Facilita la comunicación entre las distintas áreas y lleva el control del tiempo y los costes.
- **UX Designer:** se encarga principalmente del diseño de interfaz de la app (frontend) y de que la experiencia del usuario con la aplicación sea favorable.
- **Hardware Engineer:** desarrolla los circuitos, sensores y estructura del dispositivo físico, tanto para el anillo como para la tira de espalda.
- **Firmware Engineers (tira y anillo):** cada uno se encarga de los microcontroladores de lectura y transmisión de datos en cada dispositivo físico. También se encarga de que ambos dispositivos tengan una optimización del uso de la batería para una duración más larga.
- **Pentester:** asegura la integridad y la seguridad de los datos dentro del sistema, así como las posibles vulnerabilidades y su solución más óptima y segura.
- **QA Tester:** es necesario para realizar evaluaciones y pruebas del producto final, tanto con usuarios como a nivel interno.

- **Data Algorithm Specialist:** implementa algoritmos para analizar las posturas y calcular las respectivas puntuaciones. Uno de los requisitos sería que sepa sobre machine learning para poder aplicar el mejor algoritmo para los cálculos y las estimaciones de las mejores posiciones con un conjunto de datos obtenido a través de los usuarios.
- **Legal Advisor:** BeLine necesita un asesor financiero y legal, además de asesoramiento en materia de protección de datos y normativa de dispositivos médicos.
- **Backend Developer:** es el encargado de crear y mantener el servidor y la base de datos sobre la cual se trabajará.
- **Mobile App Developer:** desarrolla la aplicación y se encarga de que sea compatible con diferentes dispositivos. También se asegura de que haya una comunicación efectiva entre la aplicación y el anillo para notificar al usuario de una mala postura a través de una ligera vibración.

Cada perfil cuenta con una tarifa horaria (€/h) y una dedicación estimada (FTE) por fase, distribuidas de la siguiente manera:

- **F1 – Análisis y planificación inicial:** es la fase 0 del proyecto en la que se definen los fundamentos, los casos de uso y la planificación global. También se define el plan de calidad de software (SQA) y el de configuración (SCM). Es necesario entregar una documentación clara de los requisitos y del alcance del proyecto BeLine.
- **F2 – Especificación de requisitos y arquitectura:** en esta fase pasamos a detallar requisitos funcionales y no funcionales del sistema. Se define también la arquitectura del hardware y del software y se finaliza el modelo de casos de uso. El documento de requisitos (SRS) y el esquema de la arquitectura del sistema se deben entregar en esta fase.
- **F3 – Iteración de construcción I:** la fase 3, junto con la fase 4, se encarga de la parte de desarrollo del sistema. Se construye el prototipo mínimo funcional (MVP) desarrollando el firmware del anillo y la tira, implementando el backend y la base de datos y desarrollando la versión inicial de la app de BeLine.
- **F4 – Iteración de construcción II y validación:** en la segunda iteración del desarrollo se completan las funcionalidades finales tanto de los objetos físicos como de la app y se optimiza el rendimiento energético. También se valida la calidad final del sistema y se hacen pruebas de seguridad y tests con y sin usuarios. Se revisa cuidadosamente todo el sistema y se realizan los ajustes de algoritmos de detección postural necesarios. Finalmente se entrega un informe de QA y pentesting y la versión casi final del proyecto.
- **F5 – Instalación, entrega y soporte:** esta es la última fase y está centrada en la documentación y transferencia del sistema al cliente. Se instala el sistema en el entorno de destino, se documenta todo el proceso y se le ofrece soporte al cliente post-entrega inicial.

PERFIL	DESCRIPCIÓN	F1	F2	F3	F4
Project Manager	Coordinación general, comunicación entre áreas, control de plazos y costes.	✓	✓	✓	✓
UX Designer	Diseño de interfaz y experiencia de usuario de la app	✓	✓	✓	✓
Hardware Engineer	Diseño de sensores, circuitos y estructura física del anillo y tira.	✓	✓	✓	✓
Firmware Engineers (tira y anillo)	Programación de microcontroladores y transmisión de datos-	✓	✓	✓	✓
Backend Developer	Implementación del servidor y base de datos			✓	✓
Mobile App Developer	Desarrollo y mantenimiento de la aplicación móvil, comunicación con el anillo		✓	✓	
Data Algorithm Specialist	Creación de algoritmos de análisis postural mediante aprendizaje automático		✓	✓	
QA Tester	Pruebas funcionales y de usuario del sistema final.		✓	✓	
Pentester	Verificación de seguridad e integridad de datos			✓	
Legal Advisor	Asesoría legal, protección de datos y normativa médica				✓

Figura 1. Tabla de perfiles y su participación en cada fase del proyecto

4.2 Presupuesto

El presupuesto global se ha determinado a partir de los casos de uso definidos en los diagramas, los roles necesarios en la empresa para llevar a cabo cada uno de ellos, y los costes de cada rol en cada fase. El coste total asociado al personal asciende a 619.620,00€, correspondiente a un esfuerzo total de 24.308,00€ horas distribuidas a lo largo de 8 meses y una semana de ejecución del proyecto. Los cálculos se realizaron con una dedicación media de 160 h/mes por persona, y los costes se expresan en euros, sin impuestos aplicables.

Los importes incluyen únicamente los costes de personal, sin considerar gastos adicionales de infraestructura, licencias o viajes.

Las cifras presentadas son orientativas y podrán ajustarse durante la planificación final del proyecto según los avances técnicos y las necesidades del cliente.

Para poder realizar un presupuesto más cercano a la realidad se decidió poner cifras orientativas a ciertos gastos, como infraestructura, licencias de software, viajes, y además se calculó el presupuesto final con un margen del 15% y un riesgo del 12%. La cifra final asciende a 933.742,86€, sin impuestos aplicables

Cada componente representa un módulo funcional con sus propios procesos de captura, envío y análisis de datos. La aplicación es el centro del proyecto y es la que se encarga de recopilar todos los datos provenientes de los dos dispositivos físicos y procesar la información para poder mostrarla de manera efectiva al usuario.

- **Tira:** es un dispositivo físico flexible y ligero que se encarga de recopilar la información de la postura del usuario a través de sensores internos y enviar los datos instantáneamente y mediante conexión inalámbrica al anillo. La tira opera de manera continua, pero incluye una optimización del consumo energético → sólo envía datos cuando detecta cambios significativos en la postura.

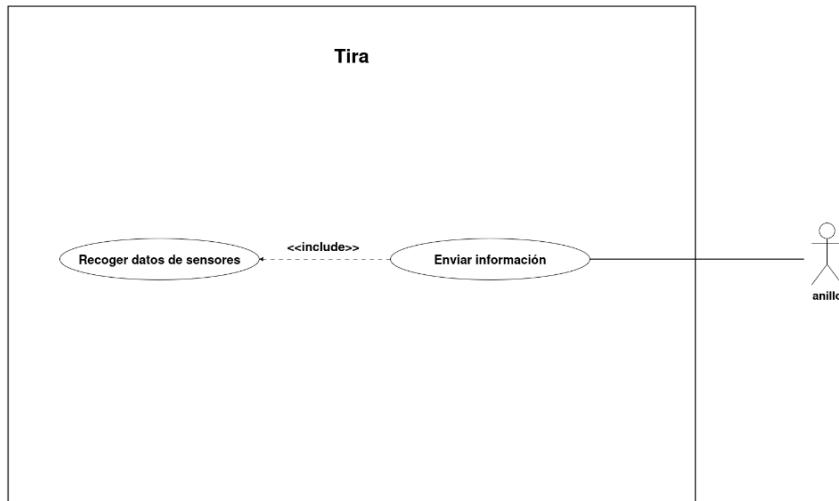


Figura 2. Casos de uso de la tira

- **Anillo:** este dispositivo es el punto central de comunicación que conecta la tira con la aplicación, ya que retransmite la información que le llega de la tira a la app. Sirve como puente entre el usuario y el software. El anillo mide las constantes vitales del usuario (frecuencia cardíaca, temperatura, etc) y detecta movimientos bruscos o posiciones prolongadas. El dispositivo emite una ligera vibración solo en los casos en los que la mala postura sea prolongada y se conecta con la aplicación para enviar también una notificación. Gracias a la detección de movimientos bruscos y a su conexión inalámbrica con la app, puede activar automáticamente una alerta o llamada de emergencia en caso de caída.

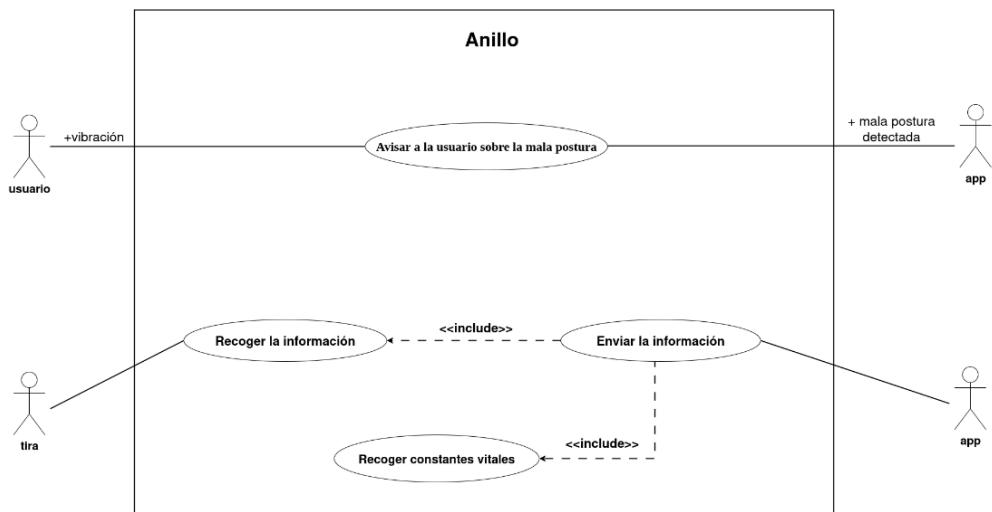


Figura 3. Casos de uso del anillo

- **Aplicación móvil:** es el núcleo de inteligencia del sistema y crea una conexión directa e interactiva con el usuario. La app cuenta con diferentes funcionalidades, entre ellas:
 - Recepción de datos: recibe los datos recopilados por el anillo y la tira.

- Triangulación entre los dispositivos: gracias a la posición del anillo y de los sensores de la tira, puede triangular la posición exacta del usuario en cada momento determinado.
- Almacenamiento y sincronización de datos con otras apps: almacena toda la información en una base de datos.
- Procesos internos de la app:
 - Calcular una puntuación según la postura del usuario. Cuanto mejor se acerque a una correcta postura a lo largo del día, mayor puntuación obtendrá el usuario.
 - Analizar los patrones de postura muy repetidos por el usuario.
 - Generar mensajes motivacionales y personalizados.
 - Elegir los mejores ejercicios para sugerir al usuario según el análisis del patrón de postura.

Gracias a estas funcionalidades, el usuario puede interactuar con la aplicación y obtener una experiencia de usuario más agradable y eficiente. El usuario puede:

- Ver su ejercicio diario realizado.
- Ver la información sobre su postura corporal a lo largo del día.
- Cambiar el modo de los mensajes según si se quiere un tono motivacional o uno estricto.
- Ver su puntuación y su ranking y compartirlo con amigos.
- Visualizar las posiciones correctas de la espalda junto con su postura actual para poder corregirla.
- Podrá ver ejercicios correctivos sugeridos por la app.

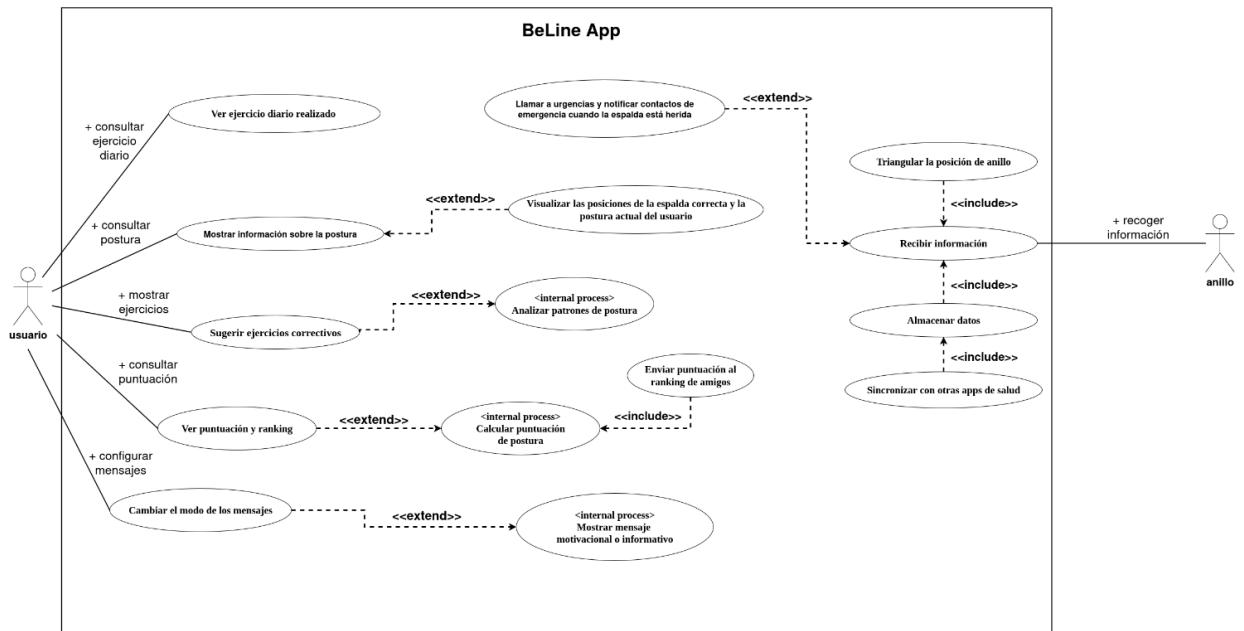


Figura 4. Casos de uso de BeLine App

5. Plan de gestión de la configuración del software (CMP)

5.1 Finalidad del Plan

La finalidad fundamental de este Plan de Gestión de la Configuración (CMP) es establecer y mantener la integridad de los productos de trabajo a lo largo de todo el ciclo de vida del proyecto. Este plan actuará como "guardián" de todos los artefactos generados: desde los requisitos iniciales y el código fuente, hasta la documentación de diseño, los planes de prueba y los ejecutables finales, etc.. es decir, cualquier elemento de la configuración. Precisamente la primera parte del plan se centra en identificar adecuadamente todos los elementos de la configuración (Cis) del proyecto.

La idea es que el CMP garantizará que, en cualquier momento, el equipo pueda identificar la versión exacta de cada componente, entender su estado, rastrear los cambios realizados y saber quién los hizo. Esto es crucial no solo para el desarrollo y la depuración, sino también para el mantenimiento futuro del software, las auditorías de calidad y el cumplimiento normativo.

Este plan busca por lo tanto definir y controlar cómo se realizarán los cambios que se producen en los elementos de la configuración (Cis), asegurando que solo las modificaciones autorizadas y debidamente documentadas se incorporen al proyecto.

En definitiva, el Plan de Gestión de la Configuración será el marco que permite que el proyecto de desarrollo de software, sea predecible, controlable y auditabile, minimizando riesgos y maximizando la probabilidad de entregar un producto que satisfaga las expectativas y funcione correctamente.

5.2 Alcance

Este plan SCM se aplicará al proyecto BeLine, cuyo objetivo es desarrollar un sistema integral hardware-software que pretende mejorar la postura corporal de los usuarios. El alcance de este plan comprende todos los elementos de nuestro proyecto que requieran control de versiones y retroalimentación continua. Se debe asegurar la integridad de cada componente y por ello, en particular, el CMP se aplicará tanto a la aplicación móvil como a los componentes físicos del proyecto, asegurando que cada versión pueda identificarse, reproducirse y auditarse adecuadamente.

Los componentes no incluidos en este plan son los componentes de software o hardware de terceros que se utilicen únicamente como soporte externo, salvo en los casos en los que se modifiquen o integren en el proyecto de manera directa.

El plan define los procedimientos, responsabilidades, herramientas y políticas de control de cambios que garanticen que únicamente las versiones autorizadas sean incorporadas.

5.3 Organización y Responsabilidades

Debe existir un contacto permanente y directo entre el personal de desarrollo y el comité de control de cambios, para que los retrasos en la tramitación de un cambio sean los menores posibles, de modo que tanto los procesos de mejora como los de corrección no sean un trabajo tedioso.

Tanto el comité de control de cambios como el resto del personal de desarrollo deben prestar especial atención a los puntos en los que se ha estipulado que se establecerán líneas de base dentro del desarrollo. Para más información, véase la sección sobre Definición y establecimiento de líneas de base así como la sección con el procedimiento de control de cambios.

Estructura y responsabilidades

- **Responsable de la Gestión de la Configuración (Configuration Manager):** es la persona responsable de implantar, definir y mantener las políticas de gestión de configuración del proyecto. Se encarga de garantizar la integridad de los elementos, mantener los repositorios y la documentación actualizados, aprobar nuevas versiones con el CCB y emitir informes del estado de las versiones del proyecto. **Encargado:** Salvador Ayala Iglesias
- **Comité de control de cambios (Change Control Board):** es el responsable de evaluar y aprobar solicitudes de cambios que puedan afectar al alcance, coste o calidad del proyecto. Sus funciones son revisar esas solicitudes y evaluar el impacto de los cambios si se aceptan las solicitudes. También se encarga de documentar todas las decisiones tomadas y los impactos sobre los requisitos y la arquitectura del proyecto. **Encargado:** Anastasiia Sadova y Aisling Florencio Pimentel
- **Bibliotecario (Configuration Management Engineer / Software Librarian):** Es el responsable de custodiar, mantener y distribuir los elementos de configuración del proyecto. Se encarga de garantizar que las versiones aprobadas estén correctamente almacenadas en los repositorios correspondientes, mantener la trazabilidad entre los diferentes artefactos del sistema y asegurar que los miembros del equipo trabajen siempre con las versiones más recientes. Además, supervisa el control de acceso a los repositorios y apoya al Configuration Manager en tareas de auditoría. **Encargado:** Nicola Cindea e Iciar García Izquierdo
- **Equipo de desarrollo:** Es el grupo encargado de implementar, probar y documentar los componentes de software del proyecto, siguiendo las políticas definidas por la gestión de configuración. Entre sus responsabilidades se incluyen la notificación de incidencias, la solicitud de cambios al CCB, la integración de nuevas versiones y la verificación de la correcta aplicación de las líneas base. El equipo debe mantener una comunicación constante con el Configuration Manager y el CCB para asegurar la coherencia entre código, documentación y entregables. **Encargado:** todos los integrantes del equipo.

5.4 Identificación de la configuración

5.4.1 Jerarquía preliminar de productos

La jerarquía preliminar de productos del sistema BeLine se organiza en tres niveles principales: producto global, subsistemas y componentes.

Esta jerarquía permite identificar la estructura modular del sistema y facilita la gestión de la configuración, el control de versiones y la trazabilidad de los cambios.

Nivel 1 – Producto global

Sistema BeLine: solución integral hardware–software para la monitorización postural y de constantes vitales del usuario.

Nivel 2 – Subsistemas

1. Subsistema Hardware

1.1 Tira sensora postural

- 1.1.1 Sensores de flexión y movimiento
- 1.1.2 Microcontrolador principal
- 1.1.3 Módulo de comunicación RF
- 1.1.4 Batería recargable y circuito de carga

1.2 Anillo inteligente

- 1.2.1 Sensores biométricos (frecuencia cardíaca, temperatura)
- 1.2.2 Módulo de comunicación Bluetooth
- 1.2.3 Motor de vibración y batería
- 1.2.4 Firmware embebido

2. Subsistema Software

2.1 Aplicación móvil BeLine

- 2.1.1 Módulo de recepción y sincronización de datos
- 2.1.2 Módulo de análisis postural y cálculo de puntuaciones
- 2.1.3 Módulo de notificaciones y alertas
- 2.1.4 Módulo de interfaz de usuario (UI/UX)
- 2.1.5 Base de datos local del usuario

2.2 Backend y servidor

- 2.2.1 API de comunicación
- 2.2.2 Almacenamiento y gestión de usuarios
- 2.2.3 Integración con aplicaciones de salud externas

Nivel 3 – Componentes de soporte

- 3.1 Documentación técnica (SRS, manuales, plan de pruebas, plan de configuración)
- 3.2 Repositorios de código fuente y firmwares
- 3.3 Librerías de software y dependencias externas

3.4 Herramientas de desarrollo (Git, Android Studio, etc.)

5.4.2 Elementos de configuración

Los elementos de configuración del sistema BeLine incluyen todos los componentes de hardware, software y documentación que requieren control de versiones, trazabilidad y gestión de cambios.

Se clasifican en tres grupos principales:

1. Elementos de hardware

Código CI	Nombre del CI	Descripción del CI	Fecha de creación	Proyecto al que pertenece	Línea base a la que pertenece	Tipo de CI
H-TIRA	Tira sensora postural	Dispositivo flexible con sensores que registran la postura de la espalda y envían datos al sistema.	19/10/2025	Sistema BeLine Posture	Design BL	Componente de hardware
H-ANILLO	Anillo inteligente	Dispositivo portátil con sensores biométricos, vibración y conectividad Bluetooth para retroalim	19/10/2025	Sistema BeLine Posture	Design BL	Componente de hardware

		entación postural.				
H-FIRMWARE-TIRA	Firmware de la tira sensora	Código embebido que controla la captura y transmisión de datos de la tira sensora.	19/10/2025	Sistema BeLine Posture	Product BL	Software embebido
H-FIRMWARE-ANILLO	Firmware del anillo inteligente	Software embebido que gestiona sensores biométricos y controla las alertas hápticas.	19/10/2025	Sistema BeLine Posture	Product BL	Software embebido
H-BATERÍA	Sistema de batería recargable	Módulo de energía encargado de alimentar los dispositivos mediante carga inalámbrica.	19/10/2025	Sistema BeLine Posture	Design BL	Componente de hardware

2. Elementos de software

Código CI	Nombre del CI	Descripción del CI	Fecha de creación	Proyecto al que pertenece	Línea base a la que pertenece	Tipo de CI
S-APP-MÓVIL	Aplicación móvil BeLine	Aplicación principal de interacción con el usuario para visualización de datos, configuración y seguimiento postural.	19/10/2025	Sistema BeLine Posture	Product BL	Software de aplicación
S-BACKEND	Servidor y API	Gestiona la comunicación entre la aplicación y la base de datos, incluyendo autenticación y sincronización de datos.	19/10/2025	Sistema BeLine Posture	Product BL	Software de servidor
S-BASE-DATOS	Base de datos	Sistema de almacenamiento que guarda métricas, historial y	19/10/2025	Sistema BeLine Posture	Design BL	Base de datos

		datos del usuario, tanto local como en la nube.				
S-ALGORITMOS-POSTURA	Algoritmos de postura	Módulo que analiza los datos de los sensores y calcula las puntuaciones posturales en tiempo real.	19/10/2025	Sistema BeLine Posture	Design BL	Módulo analítico
S-UI-UX	Interfaz de usuario y experiencia	Conjunto de pantallas, flujos de interacción y diseño visual de la aplicación BeLine.	19/10/2025	Sistema BeLine Posture	Design BL	Módulo de diseño
S-MOD-ALERTAS	Módulo de alertas	Sistema que genera vibraciones, notificaciones y alertas en tiempo real ante detección de mala postura o	19/10/2025	Sistema BeLine Posture	Product BL	Módulo funcional

		fatiga.				
--	--	---------	--	--	--	--

3. Elementos de documentación y gestión

Código Ci	Nombre del CI	Descripción del CI	Fecha de creación	Proyecto al que pertenece	Línea base a la que pertenece	Tipo de CI
D-SRS	Documento de requisitos	Especifica las funcionalidades, restricciones y requisitos técnicos del sistema BeLine.	19/10/2025	Sistema BeLine Posture	Functionality BL	Documento
D-SDD	Documento de diseño del sistema	Describe la arquitectura general y los componentes principales del sistema.	19/10/2025	Sistema BeLine Posture	Design BL	Documento
D-PLAN-CALIDAD	Plan de calidad	Define los estándares y procedimientos de verificación, validación y control de	19/10/2025	Sistema BeLine Posture	Functionality BL	Documento

		calidad del proyecto.				
D-PLAN-C ONFIG	Plan de gestión de la configuración	Establece políticas, identificadores y procesos de control de cambios dentro del sistema.	19/10/2025	Sistema BeLine Posture	Functiona l BL	Documento
D-PLAN-P RUEBAS	Plan de pruebas	Detalla los tipos de pruebas, criterios de aceptación y resultados esperados.	19/10/2025	Sistema BeLine Posture	Product BL	Documento
D-MANUAL-USUARIO	Manual de usuario	Guía dirigida al cliente final para la instalación, uso y configuración del sistema.	19/10/2025	Sistema BeLine Posture	Operation BL	Manual de usuario
D-MANUAL-MANTENIMIENTO	Manual de mantenimiento	Describe los procedimientos de calibración,	19/10/2025	Sistema BeLine Posture	Operation BL	Manual de mantenimiento

		mantenimiento y actualización del sistema.				
--	--	--	--	--	--	--

5.4.3 Esquema de identificación

Para la identificación de cada CI se ha utilizado un esquema significativo, puesto que los códigos empleados aportan información de la naturaleza del componente. Siguen la siguiente estructura: <Tipo>-<Nombre>-<Continuación nombre>

El <Tipo> está relacionado con el grupo al que pertenece el CI:

Prefijo	Significado
H	Hardware
S	Software
D	Documentación y gestión

La información registrada para cada CI es la siguiente:

- Código del CI: identificador único de cada CI. Explica información adicional.
- Nombre del CI: nombre descriptivo que permite reconocer al CI de forma clara y directa. Corresponde con el campo <Nombre>-<Continuación nombre> del identificador.
- Descripción: breve resumen de las funciones y contenido del CI para comprender su rol en el sistema.
- Fecha de creación: fecha en la que se consideró por primera vez el CI parte del proyecto. Permite mantener una cronología de la evolución del sistema.
- Proyecto al que pertenece: identifica al proyecto o subproyecto del que forma parte.
- Línea base asignada: Indica a qué línea base pertenece el CI, señalando la versión aprobada que está controlada oficialmente.
- Tipo de CI: clasifica al CI según su naturaleza (software, hardware o documentación),

5.4.4 Definición de las relaciones

En esta sección se definirán las relaciones entre los CIs que permiten comprender cómo afectan los cambios en las CIs y su impacto sobre otras CIs.

Primero explicaremos las relaciones:

- Composición (C): El CI de origen está compuesto por el CI de destino.

- Dependencia (D): El CI de origen necesita el CI de destino para funcionar o ser desarrollado. Si el destino cambia o no existe, el origen se ve directamente afectado.
- Equivalencia (E): El CI de origen es una copia idéntica del CI de destino.
- Provee (P): El CI de origen proporciona datos o funcionalidad al CI de destino.
- Derivación (Der): El CI de origen es generado o deriva del CI de destino.

Para aprovechar los códigos de identificación de los CIs (CI, H-TIRA, S-APP-MÓVIL, etc.) y simplificar la gestión de las interdependencias, se utilizará una matriz de doble entrada. En las intersecciones se indicará el tipo de relación que existe entre el CI de la fila (Origen) y el CI de la columna (Destino). Estas son las relaciones que existen en el proyecto BeLine:

CI Origen ↓ / CI Destino →	H-TIRA	H-ANILLO	S-APP- MÓVIL	S-BACKEND	S-ALGORITMO S-POSTURA	D-SRS
H-TIRA	-	D	P	P	P	D
H-ANILLO	-	-	P	P	P	D
S-APP-MÓVIL	D	D	-	D	C	D
S-BACKEND	-	-	P	-	D	D
S-ALGORITM OS-POSTURA	D/P	D/P	P	P	-	D
D-SRS	Der	Der	Der	Der	Der	-

5.4.5 Definición y establecimiento de líneas de base

Las líneas base serán una consecuencia directa de la metodología Craig Larman que hemos elegido. Decidimos usar cuatro líneas base (BL), que se establecerán al final de las Fases 1, 2.1 y 2.2. A continuación, las desglosamos:

- Functional BL (or Definition BL) - se establece al final de la fase de análisis y especificación. Incluye todos aquellos documentos en los que se define el problema a resolver, las herramientas del proyecto, los costes del proyecto y la planificación, así como los diferentes requisitos funcionales, de interoperabilidad y de interfaz de sistema entre los dispositivos (tira sensora, anillo inteligente y aplicación móvil)..

Se implementará al final de la Fase 1 (Planificación y especificación de requisitos) e incluirá los siguientes documentos:

- Casos de uso y modelo de priorización
- Oferta y Propuesta de Presupuesto

- Arquitectura preliminar del sistema (hardware y software)
 - Plan de Gestión de la Configuración del Software (SCM)
 - Plan de Aseguramiento de la Calidad del Software (SQA)
 - Estudio de viabilidad y de requisitos
 - Propuesta técnica y económica aprobada
- Design BL - se establece al final de la fase de diseño detallado. Incluye todos aquellos documentos que contienen el diseño detallado del software, el plan de implementación y la descripción de los casos de prueba. Se implementará al final de la Fase 2.1 (Primera iteración de Construcción).

Incluirá los siguientes documentos:

- Diseño detallado del software (módulos, clases, canales de comunicación)
 - Diseño del hardware para la tira sensora y el anillo inteligente
 - Diagramas de flujo de datos y estructura de la base de datos móvil
 - Plan de implementación y estrategia de integración
 - Casos de prueba
- Product BL - se establece al final de la fase de pruebas. Incluye los programas creados y todos aquellos documentos que contienen la información relativa a las pruebas realizadas. Se implementará al final de la Fase 2.1 (Primera iteración de Construcción).

Incluirá los siguientes documentos:

- Pruebas unitarias y de integración detalladas
 - Código fuente y su documentación
 - Resultados de las pruebas funcionales que se llevaron a cabo
- Operation BL - se establece al final de la fase de implementación. Incluye manuales de usuario, guías de operación y mantenimiento, manuales de capacitación, etc. Se implementará al final de la Fase 2.2 (Segunda iteración de Construcción).

Incluirá los siguientes documentos:

- Versión final del sistema (anillo, tira y aplicación)
- Manual de usuario y guía de instalación
- Manuales de mantenimiento
- Capacitación del usuario y su documentación

5.4.6 Definición y creación de bibliotecas de software

En este subpunto, debemos establecer las librerías de software y/o documentación relacionada cuyo propósito es asistir en el desarrollo y mejorar la visibilidad del sistema.

Tipos de librerías:

- 1) Librería de trabajo (CIs en desarrollo).**

Esta librería contiene elementos de configuración (Clis) en desarrollo activo y proporciona un espacio controlado donde los desarrolladores pueden modificar y probar componentes antes de su integración.

Estará alojada en un servidor de desarrolladores (**DevServer**) en la ruta **/BeLine/WorkLib**, y contendrá diferentes módulos de software, scripts y documentación asociada.

Protocolo y control de acceso:

- *Read access:*
 - 1) Los project managers tienen acceso completo de lectura para controlar el flujo de trabajo actual y registrar el progreso.
 - 2) El tester/pentester de QA tiene acceso de lectura para consultar el código en caso de que se encuentren inconsistencias o vulnerabilidades durante las pruebas.
 - 3) Todos los ingenieros de firmware y hardware podrían necesitar acceso de lectura para verificar las dependencias de firmware/software relacionadas.
- *Read and Write access:*
 - 1) El especialista en algoritmos de datos tiene acceso completo de lectura y escritura para trabajar en la algoritmización y modificar fallos lógicos, así como asegurar la eficiencia.
 - 2) Todos los desarrolladores backend y de aplicación móvil tienen ambos tipos de acceso para sus respectivos bloques de desarrollo.
 - 3) El diseñador UX tiene ambos accesos debido a que codifica directamente los elementos de la interfaz de usuario (UI).
- Todos los cambios deben realizarse a través de pull requests para su posterior revisión y para evitar que se dañe el proyecto.
- Todos los Cls deben tener descriptores que identifiquen su versión, fecha de modificación, autor y descripción para asegurar la claridad y la trazabilidad.

2) Integración (fusión de Cls en el desarrollo)

Esta librería contiene elementos de configuración (Clis) que están a la espera de ser integrados en la rama principal de desarrollo. Es una librería esencial, siguiendo nuestra metodología.

Estará ubicada en un servidor de desarrolladores (**DevServer**), en la ruta: **/BeLine/IntegrationLib**.

El protocolo y el control de acceso para la Integración siguen, en su mayoría, la misma estructura que los de la librería de trabajo, pero se añaden cambios menores:

- *Read access:*

Idéntico a la librería de trabajo, excepto:

- 1) Un asesor legal podría necesitar acceso de lectura para asegurar el cumplimiento de los requisitos legales.
- *Read and Write access:*
Idéntico a la librería de trabajo, excepto:
 - 1) Solo los desarrolladores backend y de aplicación móvil escogidos tienen ambos accesos para sus bloques de desarrollo.
- Solo los Cls verificados y aprobados de la librería de trabajo pueden ser fusionados.
- Todos los cambios deben realizarse a través de pull requests para su posterior revisión y para evitar que se dañe el proyecto.
- Todos los Cls deben tener descriptores para identificar su versión, fecha de modificación, autor y descripción, asegurando la claridad y la trazabilidad.
- Los conflictos o problemas de integración deben ser documentados y resueltos antes de la integración final.

3) Proyecto/Soporte (Cls finales que conforman las BL)

Esta librería contiene los elementos de configuración que han conformado los entregables de las líneas base.

Estará ubicada en un servidor SCM (**SCMServer**), en la ruta: **/BeLine/Project**, conteniendo las líneas base y los documentos aprobados.

El protocolo y el control de acceso para Proyecto/Soporte siguen la misma estructura que la librería de trabajo, con los siguientes cambios menores:

- *Read access:*
 - 1) Todos los roles descritos previamente tienen permisos de acceso de lectura para verificar las líneas base, revisar los módulos aprobados y asegurar el cumplimiento.
- *Read and Write access:*
 - 1) Solo el Project Manager y el Release Manager pueden añadir Cls desde la librería de integración para mantener la integridad de la línea base.
- Solo los Cls verificados y aprobados de la librería de integración pueden ser añadidos.
- Se deben realizar solicitudes formales para cualquier cambio entrante.
- Todos los Cls deben tener descriptores para identificar su versión, fecha de modificación, autor y descripción, asegurando la claridad y la trazabilidad.

4) Master (versiones entregadas al cliente)

Esta librería contiene una versión maestra de los entregables (software y documentación que se entrega al cliente). Alberga versiones estables y consistentes de los entregables para futura referencia, mantenimiento y actualizaciones.

Estará ubicada en un servidor de versiones (**ReleaseServer**), en la ruta: **/BeLine/Master**, conteniendo las líneas base y los documentos aprobados.

El protocolo y el control de acceso para Máster siguen la misma estructura que la Librería de Trabajo, con los siguientes cambios menores:

- *Read access:*
 - 1) Todos los roles descritos previamente tienen permisos de acceso de lectura para consultar las versiones entregadas.
- *Read and Write access:*
 - 1) Solo el Project Manager y el Release Manager pueden añadir nuevas versiones maestras.
- Las versiones maestras son inmutables y solo pueden modificarse mediante una solicitud formal.
- Las versiones entregadas deben ser rastreadas.
- Todos los Cls deben tener descriptores para identificar su versión, fecha de modificación, autor y descripción, asegurando la claridad y la trazabilidad.

5) Backup

Esta librería contiene versiones de respaldo de todas las demás librerías en caso de pérdida o corrupción de las mismas.

Estará ubicada en un servidor en la nube (**CloudServer**), en la ruta: **/BeLine/Backup**, conteniendo copias de las otras librerías.

Protocolo y control de acceso

- *Read access:*
 - 1) El Project Manager tiene permiso de lectura completo para verificación y restauración.
- *Read and Write access:*
 - 1) Solo las copias de seguridad automatizadas pueden escribir en esta librería durante un respaldo.
- Se deben realizar copias de seguridad diarias.
- Las versiones respaldadas deben ser rastreadas.
- Todos los Cls deben tener descriptores para identificar su versión, fecha de modificación, autor y descripción, asegurando la claridad y la trazabilidad.

5.5 Control de cambios

Para hacer un correcto control y seguimiento de los cambios, ya sea a los documentos o al código desarrollado, se usará la herramienta Hojas de Cálculo para asegurar la trazabilidad de estos.

Además de esta plataforma, se deberá usar una herramienta de control de versiones. Para el código se usará *Git* y *Github*, mientras que para los documentos se usará el historial de versiones de *Google Docs*.

Para aplicar cualquier cambio de forma definitiva, se deberá seguir un procedimiento dividido en 4 fases:

1. **Identificación de tareas:** usando Hojas de Cálculo, el CME identificará y asignará tareas a los miembros del equipo de desarrollo. Estos también podrán identificar tareas como resultado del proceso de desarrollo del código (p.e. corregir un error).
2. **Realización de tareas:** una vez identificadas, el equipo de desarrollo se encargará de realizar dichas tareas hasta finalizarlas.
3. **Solicitud de cambio:** una vez realizada la tarea, se realizará una solicitud de cambio (p.e. una *Pull Request* si se tratase de código) aportando el trabajo realizado y detallando todos los cambios.
4. **Evaluación y registro:** la solicitud es revisada por el CCB junto con el Configuration Manager para determinar si se debe aprobar o no la solicitud:
 - a. En caso de ser aprobada, el Configuration Manager registra las modificaciones y el CCB documenta los cambios realizados para asegurar la trazabilidad.
 - b. En caso de ser denegada, se notifica al solicitante de la decisión tomada y los motivos de su denegación. De esta forma, el solicitante puede implementar cambios y volver a solicitar una vez el contenido cumpla los requisitos del CCB.

5.6 Contabilidad de estado

Para registrar la contabilidad del estado (cambios a los Cis) lo haremos en el apartado 12 de este documento.

5.7 Auditoría de la configuración

La auditoría de la configuración es la responsable de verificar la integridad, coherencia y conformidad de todos los elementos del proyecto. Garantiza que todos los componentes coincidan con las versiones aprobadas y que los cambios realizados estén bien registrados y autorizados. Durante este proceso se revisa lo siguiente:

- Todos los elementos de configuración identificados están presentes y actualizados.
- Las últimas versiones corresponden con las versiones aprobadas por el CCB.
- Cada cambio aplicado está respaldado por una solicitud de cambio documentada y aprobada.
- La documentación está actualizada y refleja el estado actual del sistema.
- Los repositorios del proyecto están organizados, sin duplicados y con control de acceso adecuado.

Los resultados de la auditoría sirven como entrada para el proceso de aseguramiento de la calidad, garantizando que los hallazgos detectados se documenten y se tomen las acciones correctivas necesarias.

El Plan de Gestión de la Configuración (CMP) está directamente vinculado con el plan de calidad del proyecto, ya que ambos buscan garantizar la fiabilidad, trazabilidad y control del producto desarrollado. El Plan de Calidad define los estándares y procedimientos que aseguran la calidad del producto mientras que el CMP establece los métodos y herramientas necesarias para ello sin que la calidad se vea afectada con los diferentes cambios realizados. El aseguramiento de la calidad en el ámbito de la Gestión de Configuración se realiza mediante el procedimiento general de verificación:

1. **Revisión específica del Plan de Gestión de la Configuración:** el responsable de calidad realizará esta revisión verificando que se cumplan las normas IEEE Std. 828-2005 y 1042-1987.
2. **Comprobación del CMP y todas las actividades, mecanismos y responsabilidades:** se comprobará que el CMP define claramente todo lo necesario para el control y la gestión de los cambios en todas las fases del proyecto.
3. **Mecanismo de control de versiones:** durante el proyecto se debe cumplir que el CMP incluya mecanismos de control de versiones, al igual que toda la documentación de cambios para así poder garantizar la trazabilidad y la generación de un software de calidad.
4. **Informe de Auditoría:** aquí se deben reflejar todos los resultados de esta revisión donde se registrará la aprobación o las posibles correcciones del CMP.

6.Plan de calidad

En este apartado se recogen las tareas a realizar para garantizar la calidad de los productos obtenidos durante el desarrollo del proyecto. El objetivo fundamental del Plan de Calidad es poder cumplir con las expectativas del cliente de manera más efectiva y lograr el objetivo esperado del proyecto.

También se indican las personas encargadas de llevar a cabo cada una de estas tareas, así como la normativa a seguir para informar de los defectos encontrados y hacer un seguimiento de los mismos hasta su corrección.

Para verificar que el Plan que se detalla a continuación se cumple, se elaborarán informes de auditoría en los que se expondrán los resultados de todas las revisiones realizadas en el marco de este plan. Esto se incorporará al punto 11 del presente documento.

6.1 Revisiones previstas

En los sucesivos puntos del documento, se expondrán las tareas detalladas que se van a llevar a cabo en el cumplimiento del Plan de Garantía de Calidad para comprobar que todo el proyecto cumple con los criterios de calidad necesarios y que han sido considerados como imprescindibles para la correcta realización del proyecto.

Las revisiones se harán a medida que se vayan completando las fases del proyecto hasta llegar al diseño final y completo del producto.

Los responsables de realizar las revisiones y aceptar la validez de los productos serán Iciar García Izquierdo como Responsable de Calidad y Salvador Ayala Iglesias como Jefe de Proyecto. Además, todos los miembros del equipo de trabajo deberán realizar las revisiones asignadas por el Jefe de Proyecto y comunicar a los dos responsables del Plan de Aseguramiento de la Calidad en caso de encontrar algún fallo.

El establecimiento de este plan de garantía de calidad comenzará en el Estudio de Viabilidad del Sistema y se aplicará en adelante durante todo el desarrollo del proyecto de software (análisis, diseño, implementación...).

Para cada una de las revisiones, deberá añadirse un Informe de Auditoría que recoja la aprobación o rechazo del producto revisado, indicando, en su caso, las causas de rechazo de dicho producto. Esto se llevará a cabo en el apartado 11 del documento.

6.1.1 Revisión del Estudio de Viabilidad del Sistema

Iciar García Izquierdo, como Responsable de Calidad, confirmará que los requisitos se han especificado de forma estructurada, con un contenido preciso y completo, tal y como se establece en el Plan de Garantía de Calidad. Nuestro Responsable de Calidad se asegurará de que el documento de especificación de requisitos ofrezca las siguientes características:

- Identificación de absolutamente todos los requisitos del usuario.
- Coherencia entre el contenido del documento y su objetivo.
- Cada requisito describe la funcionalidad que le corresponde.
- Correspondencia entre los requisitos del documento y los requisitos obtenidos del usuario, por lo que la especificación de requisitos es completa.
- Descripción de los requisitos en un lenguaje claro, sin ambigüedades y, por tanto, preciso.
- El estudio de viabilidad es autodescriptivo, ya que se describen su estructura y contenido.
- Se llevará a cabo una matriz de trazabilidad de requisitos para comprobar que todos los requisitos de usuario tienen asociado al menos un requisito de software y, por tanto, están presentes en el diseño del sistema.

6.1.2 Revisión de los Casos de Uso

6.1.2.1 Revisión del Diagrama de Casos de Uso

Los casos de uso son una herramienta muy importante en el proceso de desarrollo de software y los utilizamos para estimar las actividades antes de modelar o construir un proceso de desarrollo de software.

Con los casos de uso tenemos las funcionalidades y características o requisitos básicos del sistema. No se basan en ningún lenguaje, por lo que son independientes de ellos.

A partir de los casos de uso, utilizando el método de casos de uso, se estimará el tamaño del software. El requisito para poder utilizar esta herramienta es definir un modelo de casos de uso que represente bien el dominio del problema que se va a abordar.

Iciar García Izquierdo, como Responsable de Calidad, debe realizar la revisión del Diagrama de Casos de Uso, para ello debe verificar que el diagrama de casos de uso cumpla con lo siguiente:

- El diagrama de casos de uso describe el comportamiento del sistema, es decir, la funcionalidad completa del proyecto de software que se va a desarrollar.
- El diagrama de casos de uso incluye todos los casos de uso identificados que representan todas las funcionalidades del sistema.
- El diagrama de casos de uso incluye a todos los actores identificados e implicados en el sistema.
- El diagrama de casos de uso incluye todas las dependencias y relaciones entre actores y casos de uso.
- El diagrama de casos de uso se ajusta a la notación gráfica definida en el lenguaje de modelado UML.
- El modelo de casos de uso incluye un glosario de términos que describe la terminología utilizada.

6.1.2.2 Revisión de Casos de Uso de Alto Nivel

Iciar García Izquierdo, como Responsable de Calidad, debe llevar a cabo la revisión de los Casos de Uso de alto nivel, para ello debe verificar que cumplen con lo siguiente

- Los casos de uso de alto nivel contienen el nombre, los actores, la descripción y el tipo de caso de uso.
- Cada caso de uso describe cómo alcanzar un único objetivo, es decir, describe una característica del sistema.
- Cada caso de uso contiene una descripción textual de la funcionalidad asociada con el nivel de detalle adecuado, incluidas las formas en que los actores previstos podrían trabajar con el sistema. La descripción utilizará el lenguaje del usuario final.
- Los casos de uso no describen la funcionalidad interna del sistema ni explican cómo se implementará. No incluyen jerga técnica.
- Cada caso de uso muestra los pasos que sigue el actor para realizar una operación.
- Los casos de uso se ajustan a la notación gráfica definida en el lenguaje de modelado UML.

6.1.3 Revisión del Plan de Gestión de la Configuración

Iciar García Izquierdo, como Responsable de Calidad, debe llevar a cabo la revisión del Plan de Gestión de la Configuración, para ello debe comprobar que cumple con lo siguiente:

- El proyecto incluye un Plan de Gestión de la Configuración para el control y gestión de los cambios en el que se establecen las actividades a realizar que permiten el control y gestión de los cambios en el proyecto.
- El Plan de Gestión de la Configuración cumple con IEEE Std. 828 - 2005: "IEEE Standard for Software Configuration Management Plans" y ANSI/IEEE Std. 1042 - 1987: "IEEE Guide to Software Configuration Management".
- La gestión de la configuración definida en el SCM se lleva a cabo durante todas las fases de desarrollo del proyecto de software, incluidos el mantenimiento y el control de cambios.
- El SCM describe un mecanismo de control de cambios y versiones que garantiza la producción de software de calidad.
- El SCM incluye el procedimiento para generar la documentación necesaria para el registro y seguimiento de los cambios que se produzcan durante el desarrollo del proyecto.

6.1.4 Revisión de la Estimación

Cuando se planifica un proyecto, hay que obtener una estimación del coste y el esfuerzo humano necesarios. La estimación es una de las actividades cruciales del proceso de gestión de proyectos de software, necesaria para la planificación del proyecto.

Iciar García Izquierdo, como Responsable de Calidad, debe realizar la revisión de la estimación realizada para el proyecto de desarrollo de software, para ello debe revisar lo siguiente:

- El método utilizado para estimar el esfuerzo de desarrollo del proyecto de software utiliza métricas orientadas al tamaño basadas en puntos de casos de uso.
- Antes de cada iteración, verifique que la estimación se ha realizado teniendo en cuenta los casos de uso incluidos en la estimación.
- Los puntos de uso para cada una de las iteraciones se han calculado siguiendo el procedimiento establecido para este método de estimación, que incluye los siguientes pasos:

- o Clasificar cada iteración entre actor y caos de uso en función de su complejidad y asignarle un peso en función de la misma.
- o Calcule la complejidad de cada caso de uso en función del número de transacciones o pasos del caso.
- o Calcular los puntos de caso de uso no ajustados de la iteración.
- o Calcular los factores de complejidad técnica y medioambiental.
- o Calcular los puntos de casos de uso ajustados.

● Una vez obtenidos los puntos de casos de uso para una iteración, compruebe que se ha calculado a partir de ellos el esfuerzo correspondiente necesario para llevarlos a cabo en esa iteración.

6.1.5 Revisión de la Planificación

La planificación es el proceso de establecer objetivos temporales y elegir los medios para alcanzarlos. Es fundamental realizar un análisis del proyecto para prever desde el principio y durante el desarrollo del mismo las situaciones que puedan surgir y crear las condiciones necesarias para poder resolverlas o minimizar las consecuencias que puedan tener sobre el desarrollo del proyecto y la consecución de los objetivos.

Iciar García Izquierdo, como Responsable de Calidad, debe llevar a cabo la revisión de la planificación realizada para el proyecto de desarrollo de software, para ello debe verificar lo siguiente:

- Se ha realizado una priorización de los casos de uso a desarrollar y se han definido las iteraciones que conformarán el desarrollo completo del software y los casos de uso incluidos en cada una de ellas.
- Se ha realizado una estimación de cada iteración basada en casos de uso. A partir de esta estimación, se llevará a cabo la planificación.
- Antes de iniciar una iteración, se realizará una planificación de la misma basada en la estimación del esfuerzo necesario en función de los puntos de casos de uso.
- La planificación prevista para el desarrollo del proyecto de software se adaptará y actualizará a medida que avance el proyecto.
- La planificación incluye cuántas personas deben participar en el equipo del proyecto, qué conocimientos técnicos se necesitan, cuándo aumentar el número de personas y quién participará.

- La planificación realizada define cómo se organizará el equipo que trabajará en el proyecto de desarrollo de software.
- La planificación sigue la metodología aplicada al proyecto de desarrollo de software que es, en este caso, incremental iterativa basada en casos de uso.
- Se incluye un diagrama de Gantt que representa todas las actividades que deben realizarse a lo largo del periodo de desarrollo del proyecto. El diagrama conecta las distintas actividades en función de sus relaciones de precedencia y define los recursos y tiempos estimados para cada una de ellas.
- El diagrama de Gantt refleja las tareas y las fechas clave, los hitos y la dependencia entre tareas.
- Las métricas de calidad que se aplicarán a la planificación realizada serán
 - Velocidad a la que se completan los objetivos o requisitos en cada iteración
 - Urgencia y prioridad de los requisitos cumplimentados, para comprobar si hay alguna desalineación con los objetivos del proyecto y la estrategia de la organización.
 - Requisitos completados en iteración.
 - Cambios incorporados y requisitos añadidos en el alcance inicial de la iteración
 - Número de requisitos completados del total de requisitos.
 - Desviación de los resultados del proyecto respecto a la planificación inicial
 - Presupuesto disponible, presupuesto gastado y desviación financiera con respecto a la planificación inicial.
 - Satisfacción del cliente con respecto a los resultados obtenidos.

6.1.6 Revisión del Plan de Pruebas

Iciar García Izquierdo, como Responsable de Calidad, debe llevar a cabo la revisión del Plan de Pruebas, para ello debe hacer lo siguiente:

- Debe comprobarse que existen normas de realización de las pruebas que permitan verificar que éstas se han llevado a cabo, así como indicar cómo actuar en caso de diferencias entre el resultado esperado y el obtenido.

- Debe realizarse una matriz de trazabilidad para garantizar que existen pruebas para verificar todos los requisitos del software.

6.1.7 Revisión de los Casos de Uso en formato Expandido

Iciar García Izquierdo, como Responsable de Calidad, debe realizar la revisión de los Casos de Uso en formato expandido, para ello debe hacer lo siguiente:

- A partir de cada caso de uso de alto nivel, se ha construido un caso de uso ampliado, en cada iteración.
- Cada caso de uso ampliado se compone de dos secciones, la cabecera que incluye el nombre, los actores, la descripción y el tipo de caso de uso, y el cuerpo que describe los sucesos típicos y las alternativas a los sucesos típicos.
- Los casos de uso ampliados definen el iniciador del caso de uso.
- El cuerpo del caso de uso consta de dos columnas que describen las acciones del actor y las respuestas del sistema a las mismas.

6.1.8 Revisión del Modelo Conceptual del Análisis

Iciar García Izquierdo, como Responsable de Calidad, debe realizar la revisión del Modelo Conceptual, para ello se debe verificar lo siguiente:

- El modelo de análisis representa los aspectos del problema de forma cercana a los conceptos del dominio del problema y describe las principales características del sistema. Se validará el modelo de análisis realizado en cada una de las iteraciones que componen el proyecto.
- El modelo conceptual no incluye las decisiones de aplicación. También se comprobará que es independiente de la aplicación.
- El modelo conceptual se ajusta a la notación gráfica del lenguaje de modelado UML. También debe comprobar que la notación tiene el nivel de detalle necesario para representar el problema, sin estar sobrecargada.
- El modelo conceptual se ha realizado mediante un modelo de objetos o diagrama de clases (sin métodos) que define las propiedades del sistema. Las entidades y las relaciones entre ellas se han identificado para cada iteración.
- Las métricas de calidad que deben aplicarse al modelo conceptual resultante del análisis en cada iteración son las siguientes:
 - Calidad semántica: correspondencia entre el modelo y el dominio, es decir, el modelo refleja el dominio. Se comprobará la validez del modelo, es decir, que

todos los hechos incluidos en el modelo son correctos y pertinentes para el dominio.

- o Completitud: el modelo se comprobará para garantizar que todos los hechos son correctos y pertinentes para el dominio.
- o Calidad del lenguaje: el lenguaje de modelado utilizado para capturar el dominio es un lenguaje fácil de entender por todos los participantes. La formalización del lenguaje permite la ejecución del sistema.
- o Calidad sintáctica: existe una correspondencia entre la externalización del modelo y la extensión del lenguaje en el que está escrito el modelo.

6.1.9 Revisión de los Contratos de Operación

Iciar García Izquierdo, como Responsable de Calidad, deberá realizar la revisión de los contratos de operación que se generen, para ello se deberá verificar lo siguiente:

- Para cada caso de uso, debe existir un contrato de operación para cada acción del actor.
- Cada contrato de explotación constará de los siguientes campos: nombre, responsabilidades, referencias cruzadas, notas, excepciones, salida, condiciones previas y condiciones posteriores.
- Las referencias cruzadas en el contrato corresponderán a referencias a los requisitos definidos en el proyecto que se resuelven con el caso de uso al que pertenece el contrato de explotación.

6.1.10 Revisión del Diagrama de Clases

Evaluar si el diseño obtenido cumple el nivel de calidad exigido es importante para conocer la eficacia de los procesos que se han modelado y si requieren o no un gran esfuerzo para su implantación.

La evaluación de modelos de clases de diseño mediante la aplicación de métricas permite detectar deficiencias y posibles mejoras desde las primeras fases del desarrollo del producto, evitando que se extiendan a fases posteriores y posibilitando la creación de un sistema robusto desde su concepción.

Iciar García Izquierdo, como Responsable de Calidad, deberá realizar la revisión de los Diagramas de Clases, para ello deberá comprobar lo siguiente:

- Se realizarán diagramas de clases para cada iteración con UML y el diseño será totalmente independiente de la implementación.

- Se medirá la comprensibilidad del modelo o facilidad con la que se puede entender el diagrama de clases, la analizabilidad del modelo o facilidad que ofrece el diagrama de clases para descubrir sus deficiencias o errores, y la modificabilidad del diagrama o facilidad que ofrece el diagrama para realizar una modificación especificada, ya sea por error, por un concepto no tenido en cuenta o por un cambio en los requisitos.
- Se utilizarán las siguientes métricas para medir la complejidad estructural de los diagramas de clases:
 - Número de clases: número total de clases.
 - Número de atributos: número total de atributos.
 - Número de métodos: número total de métodos.
 - Número de asociaciones: número total de asociaciones.
 - Número de agregaciones: número total de relaciones de agregación.
 - Número de dependencias: número total de relaciones de dependencia.
 - Número de generalizaciones: número total de relaciones de generalización.
 - Número de jerarquías de generalización: número total de jerarquías de generalización
 - Número de agregaciones: número total de relaciones de agregación.
 - WMC: métodos ponderados por clase, según su complejidad.
 - DIT máximo: es el valor máximo de DIT obtenido para cada clase en un diagrama de clases. Para una clase dentro de una jerarquía de generalización, es la longitud del camino más largo desde la clase hasta la raíz de la jerarquía.
 - HAgg máximo: es el valor HAgg máximo obtenido para cada clase en el diagrama de clases. Para una clase dentro de una jerarquía de agregación es la longitud del camino más largo desde la clase hasta las hojas.
- Las métricas propuestas están muy relacionadas tanto con el tiempo de mantenimiento como con la comprensibilidad, analizabilidad y modificabilidad del diagrama de clases diseñado.

6.1.11 Revisión de los Diagramas de Secuencia

Iciar García Izquierdo, como Responsable de Calidad, debe realizar la revisión de los diagramas de secuencia generados en el proyecto durante la fase de diseño de cada iteración, para ello se debe verificar lo siguiente:

- Para cada caso de uso, se han diseñado diagramas de secuencia que definen tanto el curso típico como los cursos atípicos de los eventos definidos en ellos.
- Los diagramas de secuencia muestran la interacción representada por la secuencia de mensajes entre las instancias de clase y los actores. Los diagramas muestran instancias y eventos que describen la interacción entre las clases.
- El tiempo fluye por los diagramas y muestra el flujo de control de un participante a otro.
- En la definición de los diagramas se sigue la notación UML. Los elementos incluidos en el diagrama de secuencia son:
 - Nombre del diagrama de secuencia.
 - Líneas de vida para actores e instancias de clase.
 - Mensajes entre instancias que definen el método al que llama el mensaje en la línea de vida receptora. Además, la línea receptora está vinculada a una interfaz o clase.
 - Los bucles indican el número de veces que se ejecuta el bucle, si se conoce.

6.1.12 Revisión de los Diagramas de Estado

Iciar García Izquierdo, como Responsable de Calidad, debe realizar la revisión de los diagramas de estado generados en el proyecto durante la fase de diseño de cada iteración, para ello se debe verificar lo siguiente:

- Los diagramas de estado definidos describen el comportamiento del sistema, y cada diagrama muestra el comportamiento de un único objeto durante todo su ciclo de vida.
- Los diagramas de estado contienen estados y transiciones, y las transiciones entre ellos incluyen los eventos o acciones correspondientes.
- El diagrama de estados muestra todos los posibles estados por los que pasa el objeto durante su vida en la aplicación como resultado de los eventos que le llegan.
- Hay un estado inicial y un estado final y todos los estados representados en el diagrama son accesibles.

6.2 Gestión de Riesgos

La Gestión de Riesgos es una parte muy importante del aseguramiento de la calidad de un proyecto y por ello, aunque podría ser un apartado con entidad propia, se ha situado dentro del Plan de Aseguramiento de la Calidad (SQA).

La gestión de riesgos de los proyectos de desarrollo de software, también conocida como risk management, es la práctica de identificar, analizar y responder de manera proactiva a diferentes tipos de riesgos potenciales de un proyecto. Un riesgo de un proyecto es todo aquello que pueda afectar al éxito del proyecto, puede ser algo que cause retrasos en el cronograma del proyecto, que haga que se exceda el presupuesto previsto o cualquier cosa que derive en la disminución del rendimiento del equipo de un modo u otro.

Con una gestión de riesgos efectiva, podremos detectar cualquier riesgo en potencia que pueda surgir durante el ciclo de vida de un proyecto y mitigarlo para que el proyecto se mantenga en curso, dentro del presupuesto y bien orientado.

En el curso de este proyecto se llevará a cabo un seguimiento simplificado de riesgos. Normalmente se tendrían que identificar los potenciales riesgos con un número significativo de atributos asociados, para a continuación cuantificar el impacto en costes/plazos/etc... de cada riesgo, junto a asignar una probabilidad de ocurrencia todos los posibles riesgos. Seguidamente estableceríamos un plan de mitigación para cada uno de los riesgos para ya pasar a realizar un seguimiento periódico de los mismos.

Sin embargo, lo que incorporamos a este Plan de la Calidad va a ser únicamente un seguimiento simplificado de los riesgos más potencialmente relevantes para el proyecto.

Se detallarán a continuación 2 tablas:

- La primera de ellas es la identificación uno por uno de los riesgos más relevantes junto a su estado actual, que inicialmente será siempre no materializado y que según avance el proyecto irá potencialmente, cambiando de estado. Cada vez que se realice un seguimiento (entrega) del proyecto se comprobará si hay algún cambio en la situación o se están siguiendo correctamente los aspectos de prevención y mitigación de riesgos.
- La segunda tabla contiene un pequeño plan de mitigación para aquellos riesgos identificados, con la descripción de posibles acciones a realizar para minimizar o eliminar el impacto de la ocurrencia del riesgo.

Si aparecen nuevos riesgos, se debe solicitar que se incluyan en este documento con una solicitud de cambio en el documento indicando que se ha detectado un nuevo riesgo en el informe de seguimiento y desea agregarlo.

Tabla 1 - Identificación de Riesgos:

- **Identificador de Riesgo:** Formado por "Riesgo" seguido de un guión y el número de identificación del riesgo.
- **Nombre:** Nombre del riesgo.
- **Origen del riesgo:** Origen del riesgo (Equipo, Natural, Tecnológico, Externo).
- **Probabilidad de ocurrencia:** Porcentaje de posibilidad de ocurrencia del riesgo.
- **Impacto del riesgo:** (muy bajo, bajo, moderado, alto, muy alto)
- **Descripción:** Explicación detallada del riesgo.
- **Consecuencias:** Explicación de los efectos que produciría dicho riesgo.

Riesgo-01	
Nombre	Abandono del equipo de desarrollo
Origen del Riesgo	Equipo
Probabilidad de Ocurrencia	Medio
Impacto	Muy alto
Description	Ante malas condiciones laborales o una alta demanda de trabajo en cortos plazos de tiempo, los integrantes del equipo de desarrollo podrían abandonar el proyecto.
Consecuencias	Con un número reducido de integrantes, el trabajo se retrasaría considerablemente, poniendo más presión sobre los miembros restantes del equipo. Este riesgo, por tanto, se retroalimenta.
Estado	No materializado

Riesgo-02	
Nombre	Problemas de firmware con los componentes hardware
Origen del Riesgo	Tecnológico
Probabilidad de Ocurrencia	Bajo
Impacto	Medio
Description	El hardware utilizado, a pesar de ser simple, puede presentar problemas como incompatibilidades entre componentes o diferentes entornos de desarrollo. Es crucial comunicar todos los componentes correctamente.
Consecuencias	El desarrollo del software de la aplicación móvil podría encontrar errores inesperados derivados de problemas con el hardware o su firmware, retrasando el desarrollo.
Estado	No materializado

Riesgo-03	
Nombre	Retrasos con respecto a las fechas de entrega
Origen del Riesgo	Equipo
Probabilidad de Ocurrencia	Alto
Impacto	Medio
Description	Ante cualquier problema no esperado, los tiempos de desarrollo pueden verse alterados.

Consecuencias	El equipo no será capaz de cumplir con los plazos establecidos y el cliente no tendrá sus entregas según lo acordado en el contrato inicial.
Estado	No materializado

Riesgo-04	
Nombre	Problemas de especificación de requisitos
Origen del Riesgo	Externo
Probabilidad de Ocurrencia	Medio
Impacto	Alto
Description	El cliente puede no tener claro inicialmente, o no comunicar correctamente, lo que desea del producto final.
Consecuencias	Los requisitos establecidos no satisfacen lo deseado por el cliente, resultando en la necesidad de revisarlos y, posiblemente, tener que rehacer componentes ya desarrollados.
Estado	No materializado

Riesgo-05	
Nombre	Falta de financiación para el proyecto
Origen del Riesgo	Externo
Probabilidad de Ocurrencia	Medio
Impacto	Muy Alto
Description	El cliente podría no proporcionar los pagos acordados a tiempo, ya sea por un retraso o una falta de fondos interna.
Consecuencias	El equipo no tendrá fondos para pagar a los integrantes del proyecto a tiempo. Esto aumenta directamente la probabilidad de ocurrencia del riesgo 01.
Estado	No materializado

Riesgo-06	
Nombre	Cambios en las tecnologías o herramientas
Origen del Riesgo	Tecnológico
Probabilidad de Ocurrencia	Medio
Impacto	Medio
Description	Las librerías o APIs externas utilizadas en el desarrollo podrían recibir actualizaciones o descontinuarse durante el ciclo de vida del proyecto.
Consecuencias	El equipo tendría que reajustar el código, evaluando los cambios y actualizando las dependencias. En caso extremo, se tendría que sustituir la dependencia en su totalidad.
Estado	No materializado

Riesgo-07	
Nombre	Falta de precisión en los sensores escogidos
Origen del Riesgo	Tecnológico

Probabilidad de Ocurrencia	Medio
Impacto	Alto
Description	Los sensores utilizados para los distintos componentes hardware podrían no devolver resultados precisos. Por ejemplo, la distancia medida entre nodos de la tira podría no estar dentro de la precisión esperada.
Consecuencias	Se tendría que evaluar sustituir los sensores o suplir la falta de precisión con un algoritmo más preciso, retrasando el desarrollo.
Estado	No materializado

Riesgo-08	
Nombre	Fallos del algoritmo de cálculo de postura ante casos atípicos
Origen del Riesgo	Tecnológico
Probabilidad de Ocurrencia	Medio
Impacto	Medio
Description	Los algoritmos teorizados e implementados podrían no adaptarse correctamente a un usuario con un físico no normativo (por ejemplo, con obesidad o con una altura extremadamente elevada).
Consecuencias	Se debería evaluar si el algoritmo tendría que ser rediseñado para adaptarse a dichos casos, ocasionando un posible retraso.
Estado	No materializado

Tabla 2 - Plan de Mitigación de Riesgos:

Nombre	Plan de Mitigación
Riesgo-01	<ul style="list-style-type: none"> - Mejorar el ambiente laboral ofreciendo buenas condiciones, reconociendo los méritos individuales y tratando las inquietudes de cada miembro del equipo a nivel personal. - Asegurar el entendimiento de cada componente por más de un miembro del equipo para, en caso de abandono de un integrante, no perder el conocimiento.
Riesgo-02	<ul style="list-style-type: none"> - Realizar una prueba de concepto temprana con todo el equipamiento con la intención de detectar incompatibilidades hardware. - Definir y documentar una interfaz de comunicación con el hardware que permita abstraer la comunicación y aislar los errores hardware.
Riesgo-03	<ul style="list-style-type: none"> - Incluir márgenes de tiempo extensos que permitan disponer de más tiempo del calculado inicialmente. - Hacer uso de metodologías ágiles que permitan reuniones diarias y semanales para monitorizar los avances. - Priorizar requisitos de tal manera que se minimice el tiempo perdido y se asegure la entrega de los componentes esenciales para el producto mínimo viable.
Riesgo-04	<ul style="list-style-type: none"> - Realizar prototipos interactivos iniciales que permitan al cliente visualizar el producto que ha especificado.

	<ul style="list-style-type: none"> - Reunirse con el cliente periódicamente para revisar y aprobar requisitos bajo firma de conformidad con lo entregado. - Reunirse con el cliente ante cualquier indicio de un requisito pobremente definido.
Riesgo-05	<ul style="list-style-type: none"> - División de pagos vinculados a entregables u otros hitos a lo largo del tiempo de vida del proyecto. - Reservar un fondo de seguridad para poder pagar a los empleados en caso de retrasos. - Solicitud de un primer pago cuya cantidad permita al equipo operar con un margen de seguridad reservado desde el principio. - Recordar las fechas de pago próximas con antelación al cliente.
Riesgo-06	<ul style="list-style-type: none"> - Realizar un seguimiento del estado de las tecnologías en uso por el equipo de desarrollo. - Evaluar tecnologías alternativas de reserva para facilitar el cambio en caso de descontinuación de estas. - Aislamiento de dependencias para minimizar el impacto de una actualización o descontinuación. - Definir políticas de actualización de dependencias para minimizar los cambios a realizar sobre el proyecto.
Riesgo-07	<ul style="list-style-type: none"> - Realizar un seguimiento del estado de las tecnologías en uso por el equipo de desarrollo, revisando las fichas técnicas y comparando con los resultados mostrados por el sensor. - Evaluar sensores alternativos de reserva para facilitar el cambio en caso de encontrar problemas. - Definir y ejecutar casos de prueba que validen la precisión de los sensores, asegurando así que se encuentran dentro del rango de tolerancia especificados. - Diseñar un algoritmo de calibración y corrección de los datos para suplir pequeñas faltas de precisión del hardware dentro de un rango aceptable.
Riesgo-08	<ul style="list-style-type: none"> - Muestreo de usuarios diversos, al incluir a un pequeño grupo de usuarios con físicos no normativos en las pruebas tempranas para obtener datos reales que permitan entrenar o ajustar el algoritmo en fases tempranas. - Documentar y mantener versiones anteriores del algoritmo para facilitar la reversión a una versión estable si una nueva implementación falla con nuevos casos. - Diseñar un algoritmo suplementario opcional que se pueda activar para ajustar los cálculos en casos concretos que se salgan de los valores esperados. - Evaluar el abandono de ese público atípico temporalmente, desarrollando una alternativa para los usuarios cuando el proyecto haya triunfado.

7. Estimación

En esta sección aclararemos y explicaremos los puntos más importantes elaborados en la hoja excel “UCP Estimation Template”.

Tipo de actor:

- Simple → Another system that interacts with the system to be developed through an API (Application Programming Interface)
- Average → Another system that interacts with the system to be developed through a protocol or a person interacting through a text mode interface.
- Complex → A person who interacts with the system through a graphic interface

Solo tenemos 3 actores:

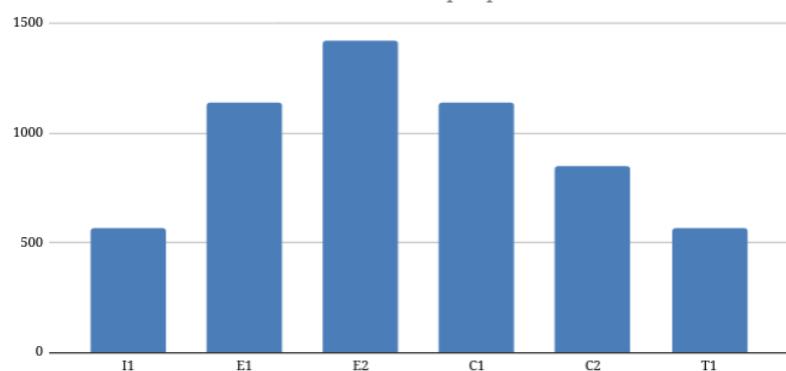
- Usuario. Lo catalogamos como complex porque es humano.
- Anillo. Catalogado como average porque es un sistema externo.
- Tira. Se le considera simple ya que solo se comunica con el anillo

Los casos de uso se clasifican con el siguiente criterio:

- Simple → 1-3 Transactions (*) Or Less than 3 classes
- Average → 4-7 Transactions (*) Or Between 5 and 10 classes
- Complex → More than 8 Transactions (*) Or More than 10 classes

Categoría	Cantidad (Number)	Justificación (Basada en la funcionalidad de BeLine)
Simples (4)	UC-04, UC-11, UC-13, UC-15	Funcionalidades de baja interacción o visualización directa (p. ej., "Comparación Postura Actual vs. Ideal", "Exportar Datos", "Visualización de Ranking").
Promedio (6)	UC-02, UC-03, UC-05, UC-08, UC-12, UC-14	Funcionalidades estándar de un CRUD (Creación, Lectura, Actualización, Borrado) o de notificación (p. ej., "Almacenamiento Seguro", "Consulta en Tiempo Real", "Sincronizar con Otras Apps").
Complejos (4)	UC-01, UC-06, UC-09, UC-10	Funcionalidades críticas o con alta complejidad algorítmica y/o integración de sistemas (p. ej., UC-01: Recepción automática de datos biométricos en tiempo real, UC-10: Detección de caída y llamada de emergencia, Sugerencia/Guía de ejercicios).

Effort Distribution per phase



Grupo de Trabajo	I1	E1	E2	C1	C2	T1	Total
Total Horas Persona	568	1.135	1.419	1.135	851	568	5.676
Gestión del Proyecto	45	91	114	91	68	45	
Gestión de la Calidad	45	57	71	57	43	28	
Gestión de la Configuración	28	57	71	57	43	28	
Gestión de Requerimientos	335	114	71	57	43	-	
Análisis y Diseño	-	533	667	284	213	-	
Implementación y Pruebas	114	284	426	590	357	153	
Implantación	-	-	-	-	85	312	
Total	568	1.135	1.419	1.135	851	568	5.676

8. Planificación

En este apartado sólo se detallarán los aspectos más importantes relativos al diagrama de Gant elaborado con MS Project. Para consultarlos en su totalidad, se debe consultar el archivo adjunto “Gant.mpp”.

- **Días no laborables:** se han establecido cómo días festivos todos aquellos incluidos en el calendario de festivos regionales. Además, se han incluido los festivos específicos del municipio de Madrid. Por último, se ha decidido dar dos semanas a los trabajadores con motivo de la celebración de la Navidad.
- **Personal contratado:** para no alargar en exceso el proyecto, se ha decidido contratar a un desarrollador de backend y a un desarrollador de aplicación móvil adicionales con respecto al personal detallado inicialmente en el presupuesto.
- **Fechas de inicio y finalización:** se ha establecido como fecha de inicio el 15 de septiembre del 2025 (el año actual). La fecha de finalización se ha calculado automáticamente en base a los esfuerzos calculados en la estimación de costes del apartado anterior.
- **Distribución de esfuerzo con respecto a los casos de uso:** el esfuerzo total por caso de uso se ha calculado siguiendo la priorización realizada en el apartado 9 de este documento. Tomando el total de horas calculadas para las fases de “Análisis y Diseño” e “Implementación y Pruebas”, se han repartido proporcionalmente a la complejidad calculada.
- **Línea base y el estado del proyecto:** se ha establecido la línea base y se ha avanzado a la fecha actual (14/11/2025). Aún no se ha logrado completar el primer hito pero está a un 76% de finalización, según las estimaciones realizadas.
- **Dependencias entre tareas:** se han establecido todas las dependencias que se han considerado necesarias (tanto entre fases del proyecto como entre casos de uso). La mayoría de casos de uso se han considerado independientes, siendo el principal factor limitante la disponibilidad del personal contratado.

9. Viabilidad y especificación de requisitos

9.1 Estudio de Viabilidad

9.1.1 Definición de Requisitos

Los requisitos se describirán a continuación con el siguiente formato:

Identificador:	
Nombre:	
Prioridad:	Fuente:
Necesidad:	
Claridad:	Verificabilidad:
Estabilidad:	
Descripción:	

Donde:

- La identificación de los requisitos se hará de la siguiente manera:
 - Identificador: UG-nnn, donde
 - U: indica que se trata de un requisito del usuario
 - G: Requisito general
 - nnn: Números consecutivos para identificar un requisito
- El campo nombre resume el requisito
- La prioridad tendrá uno de los siguientes valores:
 - Alta
 - Medio
 - Bajo
- El campo fuente puede tener uno de los siguientes valores:
 - Cliente
 - Analistas
- El campo necesidad tendrá uno de los siguientes valores:
 - Alta
 - Medio
 - Bajo
- Al campo claridad se le asignará uno de los siguientes valores:
 - Alta
 - Medio
 - Bajo
- El campo verificabilidad puede tener uno de los siguientes valores:
 - Alta

- Medio
- Bajo
- La estabilidad describe la duración del requisito a lo largo de la vida del software.
 - Alta
 - Medio
 - Bajo
- El campo de descripción sirve para explicar el requisito.

REQUISITOS FUNCIONALES

Identificador: UG-001

Nombre:	Recepción de datos biométricos		
Prioridad:	Alta	Fuente:	Analistas / Diseño
Necesidad:	Alta		
Claridad:	Alta	Verificabilidad:	Alta
Estabilidad:	Alta		
Descripción:	<p>El sistema debe recibir automáticamente los datos biométricos capturados por el anillo y las tiras, incluyendo: frecuencia cardíaca (en bpm), postura (coordenadas articulares en 3D), movimiento (acelerómetro y giroscopio). Si el usuario se desconecta, el sistema deberá detener la recepción y mostrar el estado sin conexión. Si los datos son correctos, el sistema debe actualizar todas las variables internas en tiempo real y guardar los datos en memoria (UG-002).</p> <p>Condiciones de fallo: si hay una pérdida de conexión durante más de 5 segundos, el sistema deberá intentar reconnectarse automáticamente. Los paquetes corruptos o incompletos deberán ser descartados y notificados.</p>		

Identificador: UG-002

Nombre:	Validación y preprocesado de los datos recibidos.		
Prioridad:	Alta	Fuente:	Analistas / Equipo técnico
Necesidad:	Alta		
Claridad:	Alta	Verificabilidad:	Alta
Estabilidad:	Alta		
Descripción:	<p>El sistema debe comprobar que cada paquete recibido contiene un timestamp válido, una estructura completa de los datos y valores numéricos en un rango lógico. Deberá ser posible detectar valores imposibles o inconsistentes como: aceleración > +-16 (límite de IMU), rotaciones > 2000º, valores nulos o una frecuencia cardíaca menor a 30 o mayor que 240 bpm. Todos los datos deberán ser normalizados (ajustados al rango [-1,1] para que puedan ser interpretados por el algoritmo SVM).</p>		

Condición de fallo: si tras 10 segundos no se validan los datos, se notificará al usuario para que vuelva a colocar (o revise) su tira y su anillo.

Identificador: UG-003

Nombre:	Almacenamiento de datos de salud		
Prioridad:	Alta	Fuente:	Analistas / Diseño/Normativa
Necesidad:	Alta		
Claridad:	Alta	Verificabilidad:	Alta
Estabilidad:	Media		
Descripción:	El sistema debe almacenar de forma segura los datos biométricos (frecuencia cardíaca, postura y movimiento) recibidos, manteniendo un historial de mediciones diarias para su análisis posterior. Los datos deben guardarse en una base de datos cifrada con AES-256 y debe aceptar únicamente datos que ya hayan sido validados. Se debe mantener un historial diario con marcas de tiempo para cada medición. Condiciones de fallo: Si la base de datos no está disponible o hay error de escritura, se debe registrar el fallo y reintentar cada 10 segundos. Si no hay memoria suficiente, se eliminarán registros más antiguos de 30 días con previo aviso al usuario para que pueda aceptar o rechazar.		

Identificador: UG-004

Nombre:	Análisis de postura		
Prioridad:	Alta	Fuente:	Cliente / Casos de uso /Analistas
Necesidad:	Alta		
Claridad:	Alta	Verificabilidad:	Alta
Estabilidad:	Media		
Descripción:	El sistema debe analizar las coordenadas articulares (dedos, muñeca, codo, hombro) para determinar si la postura es correcta. Se utilizará un modelo de clasificación (SVM o Random Forest) entrenado con datos etiquetados. Se considera mala postura si: <ul style="list-style-type: none">○ Inclinación de columna > 15°○ Rotación de muñeca > 10°○ Desviación mantenida > 30 segundos Se generará una alerta si se detecta mala postura. (UG-009) Condiciones de fallo: Datos incompletos o sensores desconectados suspenden el análisis registrando el tipo de fallo. En caso de que el modelo no esté disponible, se dispondrá de un modelo de reserva basado en reglas y se registra una advertencia interna.		

Identificador: UG-005

Nombre:	Generación de puntuación de salud		
Prioridad:	Media	Fuente:	Analistas / Casos de uso
Necesidad:	Media		
Claridad:	Alta	Verificabilidad:	Media
Estabilidad:	Media		
Descripción:	El sistema debe calcular una puntuación diaria entre 0 y 100 basada en: la postura (60%) y su actividad física (40%). Estos datos serán recopilados de los módulos de cálculo de cada parámetro. Se aplicará una fórmula ponderada y se almacenará junto con la fecha. $\text{Puntuación} = (\text{Postura} * 0,6) + (\text{Actividad_física} * 0,4)$ Condiciones de fallo: Si falta alguna métrica, se debe calcular la puntuación parcial y marcarla como incompleta.		

Identificador: UG-006

Nombre:	Ranking entre amigos		
Prioridad:	Media	Fuente:	Usuario
Necesidad:	Media		
Claridad:	Alta	Verificabilidad:	Media
Estabilidad:	Media		
Descripción:	El sistema debe permitir compartir la puntuación diaria con amigos registrados en la app y para ello se necesita la puntuación del usuario y la de sus amigos. Se mostrará un ranking ordenado por puntuación, actualizado cada 24 horas. El usuario sólo puede ver el ranking de sus amigos y no puede ver las métricas biométricas brutas. Condiciones de fallo: Si no hay conexión a internet, se debe mostrar el ranking local y sincronizar cuando sea posible. En caso de empate se mostrará al usuario que haya mantenido la postura correcta durante más tiempo.		

Identificador: UG-007

Nombre:	Sugerencia de ejercicios correctivos		
Prioridad:	Media	Fuente:	Analistas
Necesidad:	Media		
Claridad:	Alta	Verificabilidad:	Alta
Estabilidad:	Media		
Descripción:	El sistema debe sugerir ejercicios correctivos personalizados según el tipo de desviación postural detectada durante el análisis postural (UG-004). El módulo de sugerencias recibe como entrada la zona afectada, el tipo de desviación, la severidad y el tiempo en el que el usuario ha mantenido esa postura. Se seleccionarán los ejercicios de una base de datos según el tipo de postura, nivel de dificultad elegida por el usuario, duración máxima de la que el usuario dispone y zona del cuerpo implicada.		

Condiciones de fallo: Si no se detecta mala postura, no se mostrarán sugerencias.

Identificador: UG-008

Nombre: Configuración de alertas y mensajes

Prioridad: Media **Fuente:** Cliente

Necesidad: Media

Claridad: Alta **Verificabilidad:** Alta

Estabilidad: Alta

Descripción: El sistema debe permitir al usuario configurar:

- Tipo de alerta: vibración, notificación, mensaje motivacional
- Frecuencia de alertas (5, 10, 15...min)
- Horario de silencio: un rango en el que no se enviarán alertas y el sistema deberá asegurarse de que la hora del inicio < hora final.

Todas estas preferencias deberán ser comprobadas antes de realizar las alertas.

Condiciones de fallo: Si no se puede guardar la configuración, se debe informar al usuario y mantener la anterior.

Identificador: UG-009

Nombre: Sincronización con otras apps de salud

Prioridad: Baja **Fuente:** Analistas

Necesidad: Baja

Claridad: Media **Verificabilidad:** Media

Estabilidad: Media

Descripción: El sistema debe permitir sincronizar los datos con apps compatibles (Google Fit, Apple Health) mediante sus respectivas APIs. Se debe solicitar permiso explícito del usuario.

Condiciones de fallo: Si la app externa no responde, se debe reintentar cada 30 minutos y registrar el error.

Identificador: UG-0010

Nombre: Generación de alertas automáticas

Prioridad: Alta **Fuente:** Analistas / Diseño

Necesidad: Alta

Claridad: Alta **Verificabilidad:** Alta

Estabilidad: Alta

Descripción: El sistema debe generar alertas automáticas cuando los sensores detecten métricas que superen los umbrales definidos para la salud y postura:

- Frecuencia cardíaca > 120 bpm en reposo durante más de 25 segundos consecutivos. Debe ser un umbral configurable para cada usuario.
- Postura incorrecta detectada más de 3 veces en 1 hora. Las alertas deben ser inmediatas y visibles en la interfaz.

Cada alerta tendrá una prioridad, un origen y un mensaje asociado.

Condiciones de fallo: Si el sistema no puede enviar la alerta, se debe guardar el evento y mostrarlo en el historial.

Identificador: UG-0011

Nombre:	Alerta de emergencia		
Prioridad:	Alta	Fuente:	Analistas / Diseño
Necesidad:	Alta		
Claridad:	Alta	Verificabilidad:	Alta
Estabilidad:	Alta		
Descripción:	El sistema debe poder detectar cambios repentinos y bruscos en la posición del cuerpo que podrían relacionarse con caídas repentinas. Estos cambios se detectarán a través de los sensores y cambios en las métricas del usuario que podrían considerarse anormales para ese usuario específico. En caso de detectar este evento, el sistema deberá hacer uso de su conexión con otras aplicaciones para enviar una alerta a una autoridad mayor. También deberá poder detectar si las métricas y la postura del usuario vuelven a la normalidad.		
Condiciones de fallo:	Si el sistema no puede enviar la alerta, se debe volver a intentarlo cada 30 segundos y guardarla en el historial. Si los sensores fallan, se deberá registrar el estado de los sensores.		

Identificador: UG-012

Nombre:	Visualización de historial de métricas		
Prioridad:	Alta	Fuente:	Analistas
Necesidad:	Alta		
Claridad:	Alta	Verificabilidad:	Alta
Estabilidad:	Alta		
Descripción:	El sistema debe permitir visualizar el historial de: frecuencia cardíaca, actividad física, postura y puntuación de salud. Para ello debe existir un menú historial donde el usuario podrá visualizar sus datos diarios incluso sin conexión (modo offline). Condiciones de fallo: Si no hay datos disponibles, se debe mostrar un mensaje explicativo. Si no hay conexión, deberá mostrar la información local y marcar datos no sincronizados.		

Identificador: UG-013

Nombre:	Visualización de ejercicios correctivos		
Prioridad:	Alta	Fuente:	Cliente/Diseño

Necesidad: Alta	
Claridad: Alta	Verificabilidad: Alta
Estabilidad: Media	
Descripción:	Tras la elección de 3 ejercicios sugeridos para mejorar la postura corporal según las diferentes métricas descritas anteriormente, el sistema debe mostrar la lista de los ejercicios correctivos recomendados junto con su descripción, ilustración (opcionalmente con instrucciones básicas para realizarlo), el nivel de dificultad y la zona del cuerpo trabajada.
	Condiciones de fallo: Si no hay ejercicios disponibles, se debe mostrar una sugerencia genérica de estiramiento.

Identificador: UG-014

Nombre:	Registro de usuario
Prioridad: Alta	Fuente: Cliente
Necesidad: Alta	
Claridad: Alta	Verificabilidad: Alta
Estabilidad: Media	
Descripción:	El sistema debe permitirle al usuario la creación de una nueva cuenta introduciendo obligatoriamente el nombre, un correo electrónico, una contraseña, la confirmación de la contraseña y aceptación de su política de privacidad. La contraseña deberá tener al menos 8 caracteres, entre ellos una mayúscula, un carácter especial y al menos un dígito. La contraseña se deberá almacenar encriptada y con salt. El correo electrónico deberá ser uno válido y no existir ya en la base de datos del sistema.
	Condiciones de fallo o excepciones: si hay algún problema durante el proceso de registro, se deberá notificar al usuario con la aclaración exacta, para que sea posible corregir el error.

Identificador: UG-015

Nombre:	Inicio de sesión
Prioridad: Alta	Fuente: Cliente
Necesidad: Alta	
Claridad: Alta	Verificabilidad: Alta
Estabilidad: Media	
Descripción:	El sistema debe autenticar al usuario mediante su correo y su contraseña y deberá registrar la hora, el modelo del dispositivo, la IP y la ubicación aproximada. Se deberá notificar al usuario si se detecta un inicio de sesión sospechoso desde una ubicación no común o más de 5 intentos fallidos al introducir la contraseña. Opcional: se podrá generar un token temporal SHA-256 válido por 24 horas que sea válido para confirmar la cuenta o iniciar sesión.
	Condiciones de fallo: si hay más de 5 intentos fallidos → bloquear temporalmente la cuenta hasta que el usuario confirme por correo si quiere cambiar la contraseña. Si no hay conexión, permitir el modo offline si el usuario ya tenía la sesión guardada previamente.

Identificador: UG-016

Nombre:	Sincronización entre la tira, el anillo y la app móvil		
Prioridad:	Alta	Fuente:	Cliente/Equipo técnico
Necesidad:	Alta		
Claridad:	Alta	Verificabilidad:	Alta
Estabilidad:	Media		
Descripción:	El sistema se debe emparejar con los dispositivos BeLine (anillo y tira) mediante BLE, escaneando únicamente los dispositivos cercanos y mostrando únicamente los dispositivos BeLine. El sistema enviará una solicitud de conexión, realizará una autenticación inicial mediante intercambio de claves (pairing BLE seguro) estableciendo un canal cifrado (AES-128) y le mostrará al usuario la confirmación o denegación visual.		

Identificador: UG-017

Nombre:	Calibrar la postura inicial del usuario		
Prioridad:	Alta	Fuente:	Cliente/Equipo técnico/Analistas
Necesidad:	Alta		
Claridad:	Alta	Verificabilidad:	Alta
Estabilidad:	Media		
Descripción:	Se deberá realizar una calibración de postura inicial del usuario que servirá como referencia para detectar desviaciones posteriores. Se debe solicitar al usuario colocarse en una postura neutral y se le deberán mostrar instrucciones visuales de cómo mantener la postura durante la calibración. El sistema registrará datos IMU del anillo y tiras, ángulos, coordenadas articulares estimadas mediante MediaPipe Pose y curvatura de la espalda. Deberá ser posible detectar anomalías en la posición corporal a través de los sensores. Todos estos datos se almacenarán cifrados. Condiciones de fallo: repetir la calibración ante cualquier posible movimiento del usuario que impida la correcta recopilación de los datos. Cancelar el proceso si algún sensor no responde y pedir al usuario que compruebe la sincronización de los dispositivos.		

Identificador: UG-018

Nombre:	Exportar datos		
Prioridad:	Media	Fuente:	Cliente/Equipo técnico
Necesidad:	Media		
Claridad:	Alta	Verificabilidad:	Alta
Estabilidad:	Media		
Descripción:	El usuario deberá poder acceder a sus datos y exportarlos garantizando la integridad, privacidad y trazabilidad de los mismos. Los datos a exportar serán: historial de métricas (diario, semanal, mensual o anual) que incluirá también los gráficos del progreso; puntuaciones de salud; ejercicios recomendados; alertas de cualquier tipo (inicio de sesión, fallos de algún dispositivo, etc) y perfil de postura inicial calibrada. Los formatos		

permitidos serán PD, CSV y JSON. El sistema debe extraer los datos desde la base local, validar su integridad, formatearlos según el formato y solicitar la confirmación del usuario. **Condiciones de fallo:** si existe un error al generar el archivo a exportar, se aborta la operación. Cualquier otro error se le deberá ser notificado al usuario.

Identificador: UG-019

Nombre:	Recuperación de datos ante fallos		
Prioridad:	Alta	Fuente:	Equipo técnico/QA Tester
Necesidad:	Alta		
Claridad:	Alta	Verificabilidad:	Alta
Estabilidad:	Media		
Descripción:	El sistema debe garantizar la continuidad del servicio y la preservación de los datos durante fallos inesperados, reinicios o cierres forzados. Para ello se deberá disponer de un guardado automático de los datos biométricos recibidos, último análisis postural, estado de emparejamiento con sensores y última sesión activa dentro de la aplicación. Si la aplicación se cierra inesperadamente, al volver a iniciar se debe comprobar si existe un estado guardado, validar la integridad mediante el hash, mostrar un mensaje al usuario y restaurar automáticamente los datos. No será posible guardar automáticamente datos que requieran validaciones previas antes de ser almacenadas (calibración inicial, cambios de contraseña, etc).		

Identificador: UG-020

Nombre:	Modo offline		
Prioridad:	Media	Fuente:	Equipo técnico/QA Tester
Necesidad:	Media		
Claridad:	Alta	Verificabilidad:	Alta
Estabilidad:	Media		
Descripción:	El sistema debe permitir que la aplicación funcione parcialmente sin conexión, garantizando que los datos esenciales se sigan recogiendo y almacenando de forma segura. La sincronización se volverá a reanudar cuando vuelva la conexión. Los módulos que deben mantenerse operativos son: captura de datos biométricos, procesamiento local, generación de alertas de emergencia y alertas normales. También se deberá mantener activo el guardado de la frecuencia cardíaca, la actividad, la configuración del usuario y las posturas detectadas. Las puntuaciones y los rankings se actualizarán cuando se recupere la conexión con los datos guardados durante el modo offline, y las exportaciones o sincronizaciones con apps no estarán disponibles. Los datos se almacenarán con estructura FIFO y metadatos (hash, estado, tipo de dato)		
Condiciones de fallo:	si la cola local supera el límite de almacenamiento, el sistema debe pedir al usuario que libere espacio. Si los datos se corrompen, la app deberá descartar solo los registros corruptos, no toda la cola.		

Identificador: UG-021

Nombre:	Cálculo de puntuación postural		
Prioridad:	Media	Fuente:	Analistas
Necesidad:	Media		
Claridad:	Alta	Verificabilidad:	Alta
Estabilidad:	Media		
Descripción:	Para calcular la puntuación postural, se debe hacer uso de los datos de postura recopilados durante el día y obtener el porcentaje en el que el usuario mantuvo la postura correcta para poder asignarle una puntuación. >= 90% postura correcta → 40 puntos. 70-89% → 30 puntos. 50-69% → 20 puntos 30-49% → 10 puntos <30% → 0 puntos. Esta información será enviada al módulo del cálculo definitivo de la puntuación. Si no existe registro completo, se marcará como incompleta.		

Identificador: UG-022

Nombre:	Cálculo de puntuación de la actividad física		
Prioridad:	Media	Fuente:	Analistas
Necesidad:	Media		
Claridad:	Alta	Verificabilidad:	Alta
Estabilidad:	Media		
Descripción:	El sistema debe calcular la puntuación de la actividad física según los pasos estimados. >= 10.000 pasos → 30 puntos. 5.000 - 9.999 → 15 puntos. 1.000 - 4.999 → 5 puntos. < 1.000 → 0 puntos. Esta puntuación será enviada al módulo de cálculo total de la puntuación.		

Identificador: UG-023

Nombre:	Filtros, navegación temporal y gráficos		
Prioridad:	Media	Fuente:	Analistas/UX Designer
Necesidad:	Media		
Claridad:	Alta	Verificabilidad:	Alta
Estabilidad:	Media		
Descripción:	El sistema debe disponer de filtros para visualizar los datos del historial de métricas y ejercicios sugeridos. Los datos se podrán visualizar en función de día, semana, mes, año y progreso visual. Estos filtros se podrán aplicar a cada una de las métricas. Se deberán mostrar indicadores circulares para comprobar si el usuario ha cumplido con sus objetivos de postura o de actividad física. También se mostrarán gráficos visuales de progresos mensuales o anuales en el apartado progreso visual. Condiciones de fallo: si el rango seleccionado no contiene datos, se mostrará un mensaje claro al usuario.		

Identificador: UG-024

Nombre:	Compartir datos exportados
----------------	----------------------------

Prioridad: Media	Fuente: Cliente
Necesidad: Media	
Claridad: Alta	Verificabilidad: Alta
Estabilidad: Media	
Descripción:	El archivo generado se podrá compartir a través de las aplicaciones móviles de las que el usuario disponga (email, whatsapp, instagram, etc). Se deberá añadir un mensaje de advertencia de privacidad antes de compartir y registrar esa acción en el log del usuario.
	Condiciones de fallo: si el dispositivo no permite compartir el archivo, se mostrará un mensaje indicando la causa.

REQUISITOS NO FUNCIONALES

Identificador: UG-025	
Nombre:	Rendimiento del sistema
Prioridad: Alta	Fuente: Equipo de desarrollo/Cliente
Necesidad: Media	
Claridad: Alta	Verificabilidad: Alta
Estabilidad: Alta	
Descripción:	La pantalla principal de la app se debe cargar en menos de 3 segundos y las operaciones básicas no deben superar los dos segundos de respuesta. Esto incluye operaciones como visualizar los datos, log in, envío de datos, etc.

Identificador: UG-026	
Nombre:	Seguridad y protección de datos
Prioridad: Alta	Fuente: Cliente/Pentester
Necesidad: Alta	
Claridad: Alta	Verificabilidad: Alta
Estabilidad: Alta	
Descripción:	Todos los datos y la información de los usuarios se debe almacenar cifrada (AES-256) y todas las comunicaciones con el servidor deben usar HTTPS. Sólo los usuarios autenticados deben poder acceder a datos sensibles. Los datos deben disponer también de una firma digital para evitar la corrupción/manipulación de dichos datos.

Identificador: UG-027	
Nombre:	Usabilidad
Prioridad: Media	Fuente: Cliente/Equipo UX
Necesidad: Media	
Claridad: Alta	Verificabilidad: Alta

Estabilidad:	Media
Descripción:	Toda la aplicación debe ser intuitiva y constar del uso de las Heurísticas de Nielsen y los patrones de Van Duyne para mantener el equilibrio entre un diseño bonito, limpio y fácil de entender.

Identificador: UG-028

Nombre:	Autenticación segura	
Prioridad:	Alta	Fuente: Cliente/Pentester
Necesidad:	Alta	
Claridad:	Alta	Verificabilidad: Alta
Estabilidad:	Alta	
Descripción:	El sistema deberá requerir autenticación mediante contraseñas robustas: con longitud mínima, números, caracteres especiales y mayúsculas. También deberá permitir la autenticación multifactor (MFA) y registrar los intentos de acceso fallidos.	

Identificador: UG-29

Nombre:	Denegación de accesos simultáneos	
Prioridad:	Media	Fuente: Equipo técnico
Necesidad:	Media	
Claridad:	Alta	Verificabilidad: Alta
Estabilidad:	Alta	
Descripción:	No se permitirá más de una sesión activa por usuario al mismo tiempo. Si un usuario inicia sesión desde un nuevo dispositivo, la sesión anterior deberá cerrarse automáticamente para evitar accesos no autorizados.	

Identificador: UG-030

Nombre:	Uso de salt en el cifrado de contraseñas.	
Prioridad:	Muy alta	Fuente: Pentester/Cliente
Necesidad:	Alta	
Claridad:	Alta	Verificabilidad: Alta
Estabilidad:	Alta	
Descripción:	Las contraseñas deberán almacenarse utilizando algoritmos de hashing con sal (bcrypt, Argon2 o equivalente).	

Identificador: UG-031

Nombre:	Alertas de seguridad	
Prioridad:	Alta	Fuente: Pentester/Cliente
Necesidad:	Alta	
Claridad:	Alta	Verificabilidad: Media
Estabilidad:	Media	
Descripción:	La aplicación debe poder registrar y notificar de forma automática al usuario de intentos de acceso fallidos repetidos y actividades inusuales o	

Identificador: UG-031

sospechosas (ej: acceso desde múltiples zonas diferentes) en un periodo corto de tiempo.

Identificador: UG-032

Nombre: Rendimiento**Prioridad:** Media**Fuente:** Equipo técnico**Necesidad:** Media**Claridad:** Alta**Verificabilidad:** Alta**Estabilidad:** Media**Descripción:** La CPU se debe mantener en un uso inferior al 60% y un consumo de memoria por debajo del 70% en condiciones normales.

Identificador: UG-033

Nombre: Accesibilidad y control por voz**Prioridad:** Media**Fuente:** Equipo UX/Cliente**Necesidad:** Media**Claridad:** Alta**Verificabilidad:** Alta**Estabilidad:** Media**Descripción:** La aplicación deberá cumplir las pautas WCAG 2.1 nivel AA, incluyendo contraste de color suficiente, textos ampliables, compatibilidad con lectores de pantalla, y, a ser posible, control por voz para acciones básicas.

9.1.2 Estudio de Alternativas posibles para abordar la solución

El foco principal de nuestro proyecto es diseñar un dispositivo hardware-software que sea capaz de analizar y corregir la postura corporal. Para abordar esta solución, analizaremos diferentes aspectos del mercado actual.

- Dispositivos de monitorización directa: actualmente existen dispositivos que utilizan sensores iniciales (IMU) o acelerómetros como Upright Go y ALEX+.
 - Los acelerómetros miden la aceleración lineal, mientras que los giroscopios registran la velocidad angular, lo cual permite conocer mejor la postura corporal con buena precisión y bajo consumo.
- Seguimiento mediante IA o visión: se usan cámaras RGB con buena precisión junto con algoritmos de visión por computador para detectar la posición corporal. Requieren mayor potencia y pueden comprometer la privacidad del usuario, pero son muy precisas.
- Finalmente, también existen aplicaciones móviles sin hardware externo, que se basan en el uso de sensores del móvil (acelerómetro), pero no tienen buena precisión.

BeLine pretende adaptar todas estas opciones para combinar eficiencia, precisión, comodidad, accesibilidad económica y privacidad.

Desde el punto de vista tecnológico, se analizaron también las siguientes alternativas:

- Sensores de presión: son un tipo de sensores que miden la fuerza ejercida sobre una superficie. Cuando la presión aumenta, la resistencia eléctrica disminuye generando una señal. Podría detectar cuando el usuario apoya la espalda en el respaldo.
- Sensores de flexión: son tiras conductoras que cambian sus resistencia cuando se doblan. Podrían detectar la curvatura de la columna.

A nivel de software, se han evaluado librerías y frameworks para facilitar el desarrollo:

- TensorFlow Lite: framework diseñado para dispositivos móviles o embebidos que permite entrenar modelos de IA y ejecutarlos localmente sin necesidad de conexión a internet. Se podría usar para entrenar un modelo que detecte posturas incorrectas a partir de unos datos de entrenamiento extraídos de los sensores.
- MediaPipe: framework de visión para reconocer posturas, gestos o rostros a partir de imágenes o datos de los sensores. Requiere una cámara.
- OpenPose: librería de código abierto que se usa para detectar la pose humana completa mediante redes neuronales profundas para generar un esqueleto del cuerpo en 2D o 3D. Se podría usar para entrenar modelos de postura y para las fases de validación, pero no en el producto final ya que requiere mayor capacidad de procesamiento.
- Madwick Filter y Kalman Filter: algoritmos matemáticos para microcontroladores y para predecir y corregir errores en los datos sensoriales. Calculan la orientación en tiempo real con bajo consumo.

Aspecto	Alternativa 1	Alternativa 2	Alternativa 3
Tipo de sensor	Acelerómetro / giroscopio (IMU)	Sensor de presión o flexión	Cámara + IA
Plataforma de desarrollo	Android/IOS	Multiplataforma	
Almacenamiento	Local	En la nube	Local+nube
Alerta al usuario	Vibración/sonido	feedback visual en pantalla.	

9.1.3 Valoración de Alternativas

Cada alternativa se evalúa en función de criterios como el coste, precisión, consumo energético, facilidad de uso y privacidad del usuario.

- **Tipo de sensor:** el sensor IMU cuenta con un coste medio, una privacidad y precisión muy alta y un bajo consumo energético. En cambio, el sensor de flexión, a pesar de ser similar en cuanto a privacidad y consumo, tiene un coste más bajo y una menor precisión. Por último, la alternativa de la cámara y la IA consta de una precisión muy alta al igual que su coste, una privacidad baja y un consumo energético elevado.

- **Almacenamiento:** nos interesa que el almacenamiento sea lo más seguro posible. Analizando las tres alternativas obtenemos que el coste del almacenamiento en la nube y el mixto es medio, y la privacidad de cada uno es media y alta, respectivamente. En cambio, el almacenamiento local ofrece una privacidad mayor, un coste bajo y una alta velocidad de acceso.
- **Plataforma de desarrollo:** en cuanto al coste, al implementarlo en Android/IOS, el coste se eleva bastante frente a la multiplataforma, pero ofrece un mejor rendimiento y acceso total al hardware (BLE, sensores).
- **Alerta al usuario:** si queremos ofrecerle el feedback en pantalla, corremos el riesgo de comprometer la privacidad del usuario, al igual que el coste será más elevado. Notificaciones mediante mensajes sutiles con vibración mantienen un mejor control del usuario con respecto a lo que quiere mostrar y lo que no.

9.1.4 Selección de soluciones

Tras analizar detalladamente las diferentes alternativas, seleccionamos las siguientes soluciones para el desarrollo del sistema BeLine:

- En primer lugar, contaremos con sensores IMU (acelerómetro y giroscopio), ya que mantiene un balance entre el bajo consumo, coste medio y alta precisión y privacidad. Además, también permiten detectar movimientos y desviaciones posturales sin requerir visión artificial ni almacenamiento de imágenes.
- En cuanto al almacenamiento, hemos elegido la opción del almacenamiento mixto para poder garantizar máxima privacidad a los usuarios almacenando datos críticos en el dispositivo. Las métricas podrán sincronizarse con el backend cuando haya conexión. Gracias al almacenamiento mixto garantizamos tanto privacidad como disponibilidad.
- Para poder mantener una sola base de código sin comprometer el rendimiento, hemos decidido elegir el desarrollo multiplataforma.
- Finalmente, está claro que la mejor opción para las notificaciones es enviar mensajes cortos sin feedback por pantalla para mejorar la experiencia del usuario y mantener su privacidad, así como evitar posibles distracciones.

9.2 Modelo de casos de uso y matriz de trazabilidad

9.2.1 Modelo de casos de uso

El modelo inicial de casos de uso fue refinado a partir de los requisitos funcionales definidos en la Sección 9.1.1. Este proceso de refinamiento implicó un análisis de los requisitos para garantizar que todos los comportamientos necesarios del sistema y los agentes externos correspondientes fueran identificados y representados con precisión en el modelo. También hemos añadido más requisitos funcionales.

Los siguientes actores están involucrados en la funcionalidad del sistema:

- Usuario: El usuario principal que interactúa con la aplicación móvil BeLine.

- Anillo BeLine: El dispositivo hardware encargado de medir signos vitales, detectar movimientos o caídas repentinas y actuar como un centro de comunicación.
- Tira sensora: El dispositivo hardware responsable de capturar y transmitir datos de la postura de vuelta al sistema.

Para consultar los diagramas de casos de uso UML, puede referirse a la Sección 4.2 de este documento.

9.2.2 Matriz de trazabilidad requisitos

ID -UC/ Requisit	01	02	03	04	05	06	08	09	10	11	12	13	14	15
UG-01				X										
UG-02				X										
UG-03				X										
UG-04					X									
UG-05					X						X			
UG-06											X			
UG-07								X						
UG-08									X					
UG-09										X				
UG-10						X								
UG-11						X		X						
UG-12							X							
UG-13							X	X						
UG-14	X													
UG-15	X													
UG-16		X												
UG-17			X											
UG-18												X		
UG-19													X	X

UG-20													X
UG-21											X		
UG-22											X		
UG-23							X						
UG-24											X		

9.3 Descripción de alto nivel de los casos de uso

Use case: UC-01 Registrarse e iniciar sesión

Actors: Usuario

Type: Primary

Description: El usuario abre la aplicación BeLine y elige registrarse/iniciar sesión. Durante este proceso, el usuario proporciona información personal y configura sus credenciales. Tras una autenticación exitosa, el usuario obtiene acceso al sistema.

References: UG-014, UG-015

Use case: UC-02 Emparejar dispositivo BeLine

Actors: Usuario, Anillo BeLine, Tira sensora

Type: Primary

Description: El usuario abre la aplicación BeLine e inicia el emparejamiento entre la app y el anillo inteligente y la tira dorsal mediante Bluetooth. El sistema busca los dispositivos y establece una conexión segura, permitiendo el intercambio continuo de datos biométricos y de postura. El usuario ve una confirmación de que el emparejamiento se ha realizado correctamente.

References: UG-016

Use case: UC-03 Calibrar postura inicial

Actors: Usuario

Type: Primary

Description: El usuario abre la aplicación BeLine y adopta una postura correcta siguiendo las indicaciones de la app. El sistema registra las posiciones de referencia de la columna y los brazos mediante los sensores, creando un punto de calibración para evaluar futuras desviaciones posturales. Cuando la calibración finaliza, el usuario ve una confirmación de que su postura actual es la correcta.

References: UG-017

Use case: UC-04 Capturar y almacenar datos biométricos

Actors: Anillo BeLine, Tira sensora

Type: Primary

Description: El sistema captura continuamente los datos biométricos de los dispositivos y los almacena de forma segura en la base de datos de la aplicación para su análisis

References: UG-001, UG-002

Use case: UC-05 Analizar y evaluar la postura

Actors: Usuario

Type: Primary

Description: El sistema captura los datos de los sensores y, tras sincronizarse con la app, los analiza para detectar desviaciones respecto a la postura ideal y realiza una evaluación postural. El usuario puede ver en la aplicación la evaluación de su postura actual.

References: UG-004, UG-005

Use case: UC-06 Generar y enviar alertas automáticas

Actors: Usuario

Type: Primary

Description: El sistema monitoriza continuamente la postura y los signos vitales del usuario. Cuando detecta una postura anómala o valores vitales anormales, envía alertas al usuario mediante la vibración del anillo y notificaciones en la aplicación. El usuario ve la alerta y puede tomar medidas inmediatas.

References: UG-010, UG-011

Use case: UC-08 Visualizar historial de métricas

Actors: Usuario

Type: Secondary

Description: El usuario abre la aplicación BeLine y accede a la sección de datos históricos. Allí puede ver el historial de su ritmo cardíaco, postura, actividad y puntuaciones diarias de salud. La información se muestra en tablas y gráficos de líneas, filtrables por día, semana o mes.

References: UG-012, UG-013, UG-023

Use case: UC-09 Recomendar ejercicios correctivos

Actors: Usuario

Type: Primary

Description: El usuario abre la aplicación BeLine y va a la sección de ejercicios correctivos. Allí puede ver los ejercicios correctivos propuestos según el último análisis de postura. Puede ver descripciones, ilustraciones o animaciones, así como el nivel de dificultad de los ejercicios. Selecciona un ejercicio y sigue las instrucciones, mientras el sistema confirma la finalización y actualiza las recomendaciones según sea necesario.

References: UG-007, UG-011, UG-013

Use case: UC-10 Configurar preferencias y alertas

Actors: Usuario

Type: Secondary

Description: El usuario abre la aplicación BeLine y accede a las preferencias. Allí configura las preferencias generales de la aplicación y ajusta las preferencias para el tipo de notificaciones (vibración del anillo, notificación en la app, manera de hablar en la notificación), así como los períodos de no molestar. Esas preferencias se guardan únicamente en el dispositivo local que está conectado al anillo y a la tira.

References: UG-008

Use case: UC-11 Sincronizar con apps externas

Actors: Usuario

Type: Secondary

Description: El usuario decide sincronizar sus datos de BeLine con aplicaciones externas autorizadas. El sistema solicita permiso, se conecta mediante API y transfiere los datos. El usuario recibe una confirmación una vez que la sincronización se completa.

References: UG-009

Use case: UC-12 Compartir puntuaciones / ranking social

Actors: Usuario

Type: Optional

Description: El usuario abre la función de ranking social en la aplicación BeLine y elige compartir sus puntuaciones diarias de postura y compararlas en un ranking social con amigos registrados en la app. Puede ver su posición en el ranking y compararla con la de sus amigos.

References: UG-005, UG-006, UG-021, UG-022

Use case: UC-13 Exportar o compartir datos personales

Actors: Usuario

Type: Optional

Description: El usuario abre la aplicación BeLine y selecciona la opción de exportar o compartir datos personales. El sistema muestra los formatos disponibles, como CSV o PDF. El usuario elige el formato deseado y confirma la acción. El sistema genera el archivo y lo guarda localmente o lo comparte a través del canal externo seleccionado. El usuario recibe una confirmación una vez que la exportación o el envío se completan.

References: UG-018, UG-024

Use case: UC-14 Recuperación ante fallos y guardado automático

Actors: Anillo BeLine, Tira sensora

Type: Secondary

Description: El sistema guarda continuamente toda la información biométrica. En caso de que un error ocurra, el sistema recupera la información no guardada automáticamente, de esta manera, se asegura que la información no se pierde.

References: UG-019

Use case: UC-15 Modo sin conexión y sincronización posterior

Actors: Anillo BeLine, Tira sensora

Type: Primary

Description: El sistema almacena información biométrica incluso si no está conectado. Una vez se vuelve a conectar, automáticamente se sincroniza en la base de datos toda la información recogida. El usuario puede observar los resultados una vez el proceso ha sido completado.

References: UG-019, UG-020

9.4 Priorización de casos de uso

Primero, se definen los criterios que se seguirán para la priorización. Las características de un caso de uso específico (elegidas del 6 dimension set) que convierten a un caso en uno de alto nivel son:

Index	Characteristic	Explanation	Impact
A	Architectural design	Enseña cuánto afectan los casos de uso en la estructura del sistema o	0.45

		componentes principales.	
B	Design understanding	Enseña cuánto ayudan los casos de uso a clarificar la manera en la que el sistema funciona e interacciona.	0.25
C	Technical complexity / risk	Enseña el grado de dificultad o riesgo que supone implementarlo.	0.3

Ahora, en la siguiente tabla se muestra la priorización de casos de uso (se han separado en dos iteraciones):

Use Case	A(0.45)	B(0.25)	C(0.3)	Total	Order	Iteración
UC-01. Registrarse e iniciar sesión	8	7	3	6.25	9	2
UC-02. Emparejar dispositivo BeLine	10	7	8	8.65	3	1
UC-03. Calibrar postura inicial	10	9	8	9.15	2	1
UC-04. Capturar y almacenar datos biométricos	10	8	9	9.2	1	1
UC-05. Analizar y evaluar la postura	9	7	8	8.2	5	1
UC-06. Generar y enviar alertas automáticas	7	7	6	6.7	7	2
UC-08. Visualizar historial de métricas	5	7	5	5.5	11	2
UC-09. Recomendar ejercicios correctivos	6	7	6	6.25	10	2
UC-10. Configurar preferencias y alertas	3	6	4	4.05	12	2

UC-11. Sincronizar con apps externas	6	5	8	6.35	8	2
UC-12. Compartir puntuaciones / ranking social	3	4	5	3.85	13	2
UC-13. Exportar o compartir datos personales	3	4	2	2.95	14	2
UC-14. Recuperación ante fallos y guardado automático	9	5	8	7.7	6	1
UC-15. Modo sin conexión y sincronización posterior	9	6	9	8.25	4	1

Iteración 1

Casos de uso con puntuación alta, considerados prioritarios para la funcionalidad principal del sistema:

- UC-02 Emparejar dispositivo BeLine
- UC-03 Calibrar postura inicial
- UC-04 Capturar y almacenar datos biométricos
- UC-05 Analizar y evaluar la posturas
- UC-14 Recuperación ante fallos y guardado automático
- UC-15 Modo sin conexión y sincronización posterior

Iteración 2

Casos de uso complementarios, que enriquecen la experiencia del usuario y aportan funcionalidades adicionales:

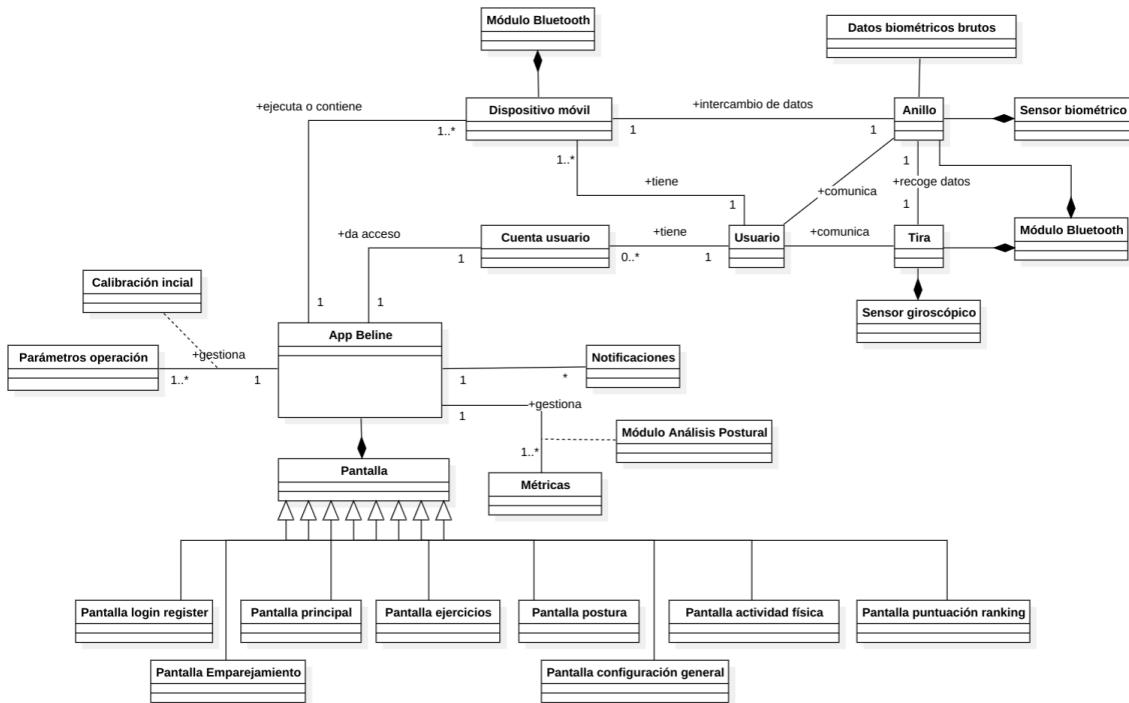
- UC-01 Registrarse e iniciar sesión
- UC-06 Generar y enviar alertas automáticas
- UC-08 Visualizar historial de métricas
- UC-09 Recomendar ejercicios correctivos
- UC-10 Configurar preferencias y alertas
- UC-11 Sincronizar con apps externas
- UC-12 Compartir puntuaciones / ranking social

10. Construcción

10.1 Primera iteración

10.1.1 Análisis de la primera iteración

Modelo conceptual



El modelo conceptual representa la arquitectura funcional del sistema BeLine, formado por dispositivos físicos, el núcleo central (la app) con sus respectivas pantallas y servicio internos de análisis. El anillo es el dispositivo principal que actúa como punto central de captura de datos biométricos puros provenientes desde los sensores de cada dispositivo. Recibe la información de la tira y la transmite hacia el dispositivo móvil del usuario y posteriormente hacia la App para que pueda ser procesada por el módulo de análisis postural. La función principal del dispositivo móvil es servir como plataforma de ejecución de la App BeLine y como puente entre los wearables y el usuario final.

El usuario representa la persona física y la cuenta del usuario representa la identidad digital del usuario dentro de nuestra aplicación que permite gestionar el acceso y personalizar la experiencia.

El núcleo del sistema es la app funcional que recibe y procesa todos los datos para ofrecer servicios a través de diferentes tipos pantallas. Todos los datos brutos recibidos, son procesados y almacenados como métricas e indicadores interpretables por el usuario.

La calibración inicial se encarga de determinar una postura correcta de referencia del usuario, guardar los datos en parámetros de operación y usarlos para futuros análisis.

El usuario también dispone de notificaciones personalizadas y de diferentes pantallas para visualizar datos de postura y actividad física, mostrar puntuaciones y ranking con amigos conectados, sugerir ejercicios correctivos, mostrar el emparejamiento de los dispositivos, modificar la configuración general y autenticar al usuario.

Descripción de casos de uso con formato ampliado

CASO DE USO: UC-02 – Emparejar dispositivo BeLine

- **Actores:** Usuario (iniciador), Anillo BeLine, Tira sensora
- **Propósito:** Establecer una conexión segura entre la aplicación BeLine y los dispositivos físicos para habilitar el intercambio continuo de datos biométricos y posturales.
- **Descripción:** El usuario abre la aplicación BeLine e inicia el emparejamiento entre la app y el anillo inteligente y la tira dorsal mediante Bluetooth. El sistema busca los dispositivos y establece una conexión segura, permitiendo el intercambio continuo de datos biométricos y de postura. El usuario ve una confirmación de que el emparejamiento se ha realizado correctamente.
- **Tipo:** Primary, Actual
- **Referencias:** UG-016
- **Curso típico de eventos:**
 1. El usuario abre la aplicación BeLine y selecciona la opción “Emparejar dispositivo”.
 2. La aplicación activa el módulo de emparejamiento y escanea dispositivos Bluetooth cercanos compatibles con BeLine.
 3. La aplicación muestra en pantalla los dispositivos detectados.
 4. El usuario selecciona dispositivos para iniciar el emparejamiento.
 5. Cada dispositivo confirma la conexión Bluetooth con la app.
 6. La aplicación verifica que ambos dispositivos transmiten correctamente paquetes de prueba.
 7. La aplicación registra los dispositivos como “vinculados” y habilita las funciones que dependen de ellos.
 8. La aplicación muestra un mensaje indicando que el emparejamiento se realizó con éxito.
- **Cursos alternativos:**
 - Paso 4. Pérdida de conexión: Si se pierde la señal de la tira o del anillo, el sistema detiene la captura, muestra un aviso para reconnectar y reinicia el proceso.
 - Paso 5. Solo un dispositivo se empareja correctamente: Si el anillo o la tira se empareja pero el otro no, la aplicación indica qué dispositivo falta por conectar y reintenta la conexión con ese dispositivo.

CASO DE USO: UC-03 – Calibrar postura inicial

- **Actores:** Usuario (iniciador)
- **Propósito:** Registrar la postura de referencia del usuario para permitir evaluaciones posturales precisas en el futuro.

- **Descripción:** El usuario abre la aplicación BeLine y adopta una postura correcta siguiendo las indicaciones de la app. El sistema registra las posiciones de referencia de la columna y los brazos mediante los sensores, creando un punto de calibración para evaluar futuras desviaciones posturales. Cuando la calibración finaliza, el usuario ve una confirmación de que su postura actual es la correcta.
- **Tipo:** Primary, Actual
- **Referencias:** UG-017
- **Curso típico de eventos:**
 1. El usuario abre la aplicación y selecciona “Calibración inicial”. La aplicación carga el módulo de calibración y verifica que los dispositivos estén conectados.
 2. La aplicación muestra instrucciones visuales y auditivas sobre cómo adoptar la postura correcta. Incluye indicaciones de pies, hombros, cuello y alineación general, junto con una animación del modelo postural ideal.
 3. El usuario adopta la postura indicada. La aplicación detecta que el usuario está en posición neutral gracias al sensor de proximidad/inmovilidad del anillo.
 4. La tira sensora y el anillo inician la captura continua de datos (ángulos, inclinación, rotación, estabilidad). El sistema establece un intervalo de muestreo y comienza a registrar valores en tiempo real.
 5. La aplicación analiza la estabilidad de las mediciones. El sistema verifica variaciones mínimas entre muestras consecutivas y descarta picos producidos por micro-movimientos. También valida que no haya interferencias (Bluetooth, latencia, señal débil).
 6. Si los datos permanecen estables durante la ventana de tiempo requerida, la aplicación registra los valores como postura de referencia. Se genera un perfil postural con parámetros normalizados y asociado al ID del usuario.
 7. La aplicación confirma la calibración. Muestra un mensaje de éxito, sugiere repetir si la persona desea mejorar la precisión y habilita las funciones de análisis postural.
- **Cursos alternativos:**
 - Paso 4. Pérdida de conexión: Si se pierde la señal de la tira o del anillo, el sistema detiene la captura, muestra un aviso para reconnectar y reinicia el proceso.
 - Paso 5. Movimiento detectado: Si las muestras muestran variaciones superiores al umbral, la aplicación muestra un aviso: “*Movimiento detectado. Mantente inmóvil unos segundos más.*” El temporizador de estabilidad se reinicia.

CASO DE USO: UC-04 – Capturar y almacenar datos biométricos

- **Actores:** Tira sensora BeLine (iniciador), Anillo BeLine (iniciador)
- **Propósito:** Registrar y almacenar de forma segura los datos biométricos y posturales capturados por los dispositivos BeLine.

- **Descripción:** El sistema captura continuamente los datos biométricos de los dispositivos y los almacena de forma segura en la base de datos de la aplicación para su análisis.
- **Tipo:** Primary, Actual
- **Referencias:** UG-001, UG-002
- **Curso típico de eventos:**
 1. La tira sensora y el anillo capturan continuamente datos biométricos y posturales. Incluyen inclinación, curvatura, presión, aceleración, frecuencia cardíaca, variabilidad, etc.
 2. Los dispositivos envían los datos a la aplicación usando el protocolo de comunicación definido (BLE de bajo consumo). Se envían en paquetes periódicos con checksum para validar integridad.
 3. La aplicación recibe los datos y verifica consistencia y validez. Filtra valores extremos, picos, interferencias y mide latencia para asegurar calidad del dato.
 4. Los datos se almacenan en la base de datos con sellos de tiempo, metadatos del dispositivo, usuario y contexto. La base garantiza atomicidad y evita duplicados.
 5. La aplicación confirma que los datos ya están disponibles para análisis posteriores. Actualiza el historial postural y envía los registros al módulo de evaluación si está activo.
- **Cursos alternativos:**
 - Paso 2. Conexión interrumpida: Los dispositivos guardan datos temporalmente en su memoria hasta restaurar la conexión.
 - Paso 3. Datos incompletos/corruptos: El sistema detecta el error, rechaza el paquete y solicita retransmisión inmediata.

CASO DE USO: UC-05 – Analizar y evaluar la postura

- **Actores:** Usuario (iniciador)
- **Propósito:** Evaluar el estado postural del usuario comparando los datos biométricos actuales con su postura de referencia.
- **Descripción:** El sistema captura los datos de los sensores y, tras sincronizarse con la app, los analiza para detectar desviaciones respecto a la postura ideal y realiza una evaluación postural. El usuario puede ver en la aplicación la evaluación de su postura actual.
- **Tipo:** Primary, Actual
- **Referencias:** UG-004, UG-005
- **Curso típico de eventos:**
 1. La aplicación recibe datos de postura en tiempo real. Se activa el módulo de cálculo postural.
 2. El sistema compara los datos con la referencia registrada en la calibración inicial. Normaliza ambos conjuntos y alinea las dimensiones correspondientes.
 3. La aplicación calcula desviaciones en ángulos, alineación, curvatura y rotación. Evalúa hombros, columna, pelvis y cuello, aplicando algoritmos de detección de asimetrías.

4. El sistema determina el nivel de corrección requerido según umbrales predefinidos. Clasifica en leve, moderado o severo e identifica las zonas específicas del problema.
 5. La aplicación muestra al usuario una evaluación con gráficos y explicaciones. Puede mostrar una figura 3D, mediciones cuantitativas y sugerencias rápidas.
- **Cursos alternativos:**
 - Paso 1. Lectura inestable: Si hay movimientos bruscos, el sistema lo marca como “dato no confiable” y pide al usuario permanecer quieto.
 - Paso 3. Falta de información: Si un sensor entrega datos incompletos, la aplicación realiza un análisis parcial y notifica al usuario.

CASO DE USO: UC-14 – Recuperación ante fallos y guardado automático

- **Actores:** Tira sensora BeLine (iniciador), Anillo BeLine (iniciador)
- **Propósito:** Garantizar que los dispositivos BeLine continúen capturando datos durante una falla de conexión y que la información se sincronice correctamente cuando se restablezca la comunicación.
- **Descripción:** El sistema guarda continuamente toda la información biométrica. En caso de que un error ocurra, el sistema recupera la información no guardada automáticamente, de esta manera, se asegura que la información no se pierde.
- **Tipo:** Secondary, Actual
- **Referencias:** UG-019
- **Curso típico de eventos:**
 1. La conexión entre los dispositivos y la aplicación se interrumpe.
 2. El sistema cambia automáticamente al modo offline.
 3. Los dispositivos continúan capturando datos y los almacenan en su memoria interna con sello de tiempo.
 4. La aplicación detecta la reconexión cuando el usuario vuelve al rango o activa Bluetooth.
 5. La aplicación solicita a los dispositivos los datos pendientes para sincronización.
 6. Los dispositivos envían los datos cronológicamente en bloques.
 7. La aplicación valida los bloques recibidos, descarta duplicados y los guarda en la base de datos.
 8. La aplicación informa al usuario que la sincronización se completó con éxito.

- **Cursos alternativos:**

- Paso 3. Memoria casi llena: El dispositivo prioriza datos esenciales y marca una alerta para mostrar en cuanto vuelva la conexión.
- Paso 6. Bloque corrupto: La aplicación detecta inconsistencias en un bloque de datos y solicita su retransmisión.

CASO DE USO: UC-15 – Modo sin conexión y sincronización posterior

- **Actores:** Tira sensora BeLine (iniciador), Anillo BeLine (iniciador)
- **Propósito:** Garantizar la continuidad de captura de datos cuando no existe conexión y sincronizar la información posteriormente.

- **Descripción:** El sistema almacena información biométrica incluso si no está conectado. Una vez se vuelve a conectar, automáticamente se sincroniza en la base de datos toda la información recogida. El usuario puede observar los resultados una vez el proceso ha sido completado.
- **Tipo:** Primary, Actual
- **Referencias:** UG-019, UG-020
- **Curso típico de eventos:**
 9. La conexión entre dispositivos y aplicación se interrumpe. El sistema cambia automáticamente a modo offline.
 10. Los dispositivos continúan capturando y almacenando datos en memoria interna. Etiquetan cada registro con su propio sello de tiempo.
 11. La aplicación detecta la reconexión cuando el usuario vuelve al rango o activa Bluetooth.
 12. La aplicación solicita a los dispositivos los datos pendientes. Envía una solicitud de recuperación ordenada.
 13. Los dispositivos transmiten los datos en bloques cronológicos. Verifican integridad con checksum y marcan bloques ya enviados.
 14. La aplicación valida los bloques recibidos, elimina duplicados y los almacena en la base de datos.
 15. La aplicación informa al usuario que la sincronización se completó con éxito.
- **Cursos alternativos:**
 - Paso 5. Bloque corrupto: La aplicación detecta inconsistencias y solicita retransmisión.
 - Paso 2. Memoria casi llena: El dispositivo prioriza datos esenciales y marca una alerta para mostrarla cuando vuelva la conexión.

Contratos de operación

CONTRATO CO-1:

- **Nombre:** Iniciar_Escaneo_BLE()
- **Responsabilidad:** Iniciar un escaneo Bluetooth y activar el componente de hardware correspondiente para comenzar la búsqueda activa de señales de dispositivos BeLine cercanos.
- **Referencias:**
 - **Requerimientos:** UG-016
 - **Casos de Uso:** UC-02
 - **Diagramas de secuencia:** SD UC-02
- **Notas:**
 - **Excepciones:**
 - Fallo de hardware. El módulo BLE no puede iniciar el escaneo debido a un fallo de hardware. En ese caso, se muestra el mensaje de error.
 - **Salida:** N/A
 - **Pre-condiciones:**
 - El Bluetooth del dispositivo móvil está activado.
 - La aplicación BeLine está abierta y en ejecución.

- El Módulo de Emparejamiento BLE ha recibido correctamente el comando para iniciar el escaneo.
- **Post-condiciones:**
- El Módulo BLE está ejecutando activamente el escaneo de dispositivos y está preparado para detectar y notificar al sistema cualquier dispositivo BeLine compatible que se encuentre.

CONTRATO CO-2:

- **Nombre:** Detectar_Dispositivo(id_dispositivo: string, tipo: string)
- **Responsabilidad:** Identificar y mapear todos los dispositivos BeLine disponibles dentro de los resultados actuales del escaneo BLE
- **Referencias:**
 - **Requerimientos:** UG-016
 - **Casos de Uso:** UC-02
 - **Diagramas de secuencia:** SD UC-02
- **Notas:**
 - **Excepciones:**
 - No hay ningún dispositivo BeLine detectado.
 - **Salida:** N/A.
 - **Pre-condiciones:**
 - El Bluetooth del dispositivo móvil está activado.
 - La aplicación BeLine está abierta y en ejecución.
 - El Módulo Emparejamiento BLE ha finalizado el escaneo.
 - **Post-condiciones:**
 - Se crearon instancias temporales para los dispositivos BeLine detectados.
 - Los atributos de las instancias fueron inicializados:
 - ❖ id_dispositivo: {id_anillo, id_tira}
 - ❖ tipo: {Anillo BeLine, Tira BeLine}

CONTRATO CO-3:

- **Nombre:** Solicitar_Pairing_Seguro(id_dispositivo: string)
- **Responsabilidad:** Iniciar el protocolo seguro de emparejamiento Bluetooth para el dispositivo BeLine seleccionado, que implica un intercambio de claves públicas y la confirmación de un canal seguro.
- **Referencias:**
 - **Requerimientos:** UG-016
 - **Casos de Uso:** UC-02
 - **Diagramas de secuencia:** SD UC-02
- **Notas:**
 - **Excepciones:**
 - Error en el intercambio de claves. Las claves públicas no se están intercambiando correctamente.
 - Fallo de Cifrado. La confirmación del cifrado del canal ha fallado.
 - **Salida:** N/A
 - **Pre-condiciones:**
 - El Bluetooth del dispositivo móvil está activado.

- La aplicación BeLine está abierta y en ejecución.
- El Bluetooth no está conectado a otros dispositivos BeLine (ej. otro anillo)
- El usuario ha elegido *id_dispositivo* para emparejar.
- **Post-condiciones:**
- Se ha enviado una solicitud de emparejamiento seguro al módulo de emparejamiento BLE.
- El módulo de emparejamiento BLE ha iniciado la comunicación e intercambio de claves con un dispositivo.

CONTRATO CO-4:

- **Nombre:** Actualizar_Estado_Dispositivo(*id_dispositivo*: string, "emparejado")
- **Responsabilidad:** Registrar el resultado del emparejamiento del dispositivo y actualizar su estado en la base de datos de la aplicación.
- **Referencias:**
 - **Requerimientos:** UG-016
 - **Casos de Uso:** UC-02
 - **Diagramas de secuencia:** SD UC-02
- **Notas:**
 - **Excepciones:**
 - Error en el acceso a la base de datos. No se puede acceder a la base de datos.
 - ID de dispositivo incorrecto. No se encontró ningún dispositivo con *id_dispositivo*.
 - **Salida:** N/A
 - **Pre-condiciones:**
 - El resultado del emparejamiento (estado) ha sido notificado.
 - **Post-condiciones:**
 - Si estado = "aceptado": El dispositivo se marca como "emparejado" de forma persistente.
 - Si estado = "error" o "denegado": El estado del dispositivo se registra como {fallido}.

CONTRATO CO-5:

- **Nombre:** Solicitar_Datos_IMU_Anillo()
- **Responsabilidad:** Solicitar a un BeLine Ring que envíe los datos del IMU recopilados al aplicativo BeLine.
- **Referencias:**
 - **Requerimientos:** UG-017, UG-001, UG-002
 - **Casos de Uso:** UC-03, UC-04
 - **Diagramas de secuencia:** SD UC-03, SD UC-04
- **Notas:**
 - **Excepciones:**
 - El anillo no está conectado. En ese caso, se muestra un mensaje de error.
 - Error de transmisión. Se muestra el mensaje de error y los datos se están retransmitiendo.

- **Salida:** N/A
- **Pre-condiciones:**
 - El anillo BeLine está conectado y activo.
- **Post-condiciones:**
 - El anillo comenzó a transmitir los datos del IMU a la aplicación.

CONTRATO CO-6:

- **Nombre:** Solicitar_Datos_Tira()
- **Responsabilidad:** Solicitar a una Tira BeLine que envíe los datos del IMU recopilados al aplicativo BeLine.
- **Referencias:**
 - **Requerimientos:** UG-017, UG-001, UG-002
 - **Casos de Uso:** UC-03, UC-04
 - **Diagramas de secuencia:** SD UC-03, SD UC-04
- **Notas:**
 - **Excepciones:**
 - La Tira no está conectada. En ese caso, se muestra un mensaje de error.
 - Error de transmisión. Se muestra el mensaje de error y los datos se están retransmitiendo.
 - **Salida:** N/A
 - **Pre-condiciones:**
 - La Tira BeLine está conectada y activa.
 - **Post-condiciones:**
 - La Tira comenzó a transmitir los datos del IMU a la aplicación.

CONTRATO CO-7:

- **Nombre:** Guardar_Postura_Refencia(postura_inicial)
- **Responsabilidad:** Almacenar la postura calculada en la base de datos del sistema, estableciendo la posición de referencia inicial del usuario.
- **Referencias:**
 - **Requerimientos:** UG-017
 - **Casos de Uso:** UC-03
 - **Diagramas de secuencia:** SD UC-03
- **Notas:**
 - **Excepciones:**
 - Error en el acceso a la base de datos. No se puede acceder a la base de datos.
 - **Salida:** N/A.
 - **Pre-condiciones:**
 - El ciclo de recopilación de datos se ha terminado correctamente y los datos finales de *postura_inicial* están disponibles.
 - **Post-condiciones:**
 - La postura calculada se está almacenando correctamente en la base de datos del sistema.

CONTRATO CO-8:

- **Nombre:** Guardar_Datos_Biometricos(datos_imu_anillo, datos_tira)
- **Responsabilidad:** Almacenar los datos IMU en crudo en la base de datos del sistema.
- **Referencias:**
 - **Requerimientos:** UG-001, UG-002
 - **Casos de Uso:** UC-04
 - **Diagramas de secuencia:** SD UC-04
- **Notas:**
 - **Excepciones:**
 - Error en el acceso a la base de datos. No se puede acceder a la base de datos.
 - **Salida:** N/A.
 - **Pre-condiciones:**
 - Los datos en crudo fueron recibidos correctamente tanto del Anillo como de la Tira BeLine (*datos_recibidos = TRUE*).
 - **Post-condiciones:**
 - Se agregó correctamente un nuevo registro que contiene los datos biométricos.

CONTRATO CO-9:

- **Nombre:** Solicitar_Evaluación(datos_sensores_actuales)
- **Responsabilidad:** Iniciar el análisis de los últimos datos del IMU por el Módulo de Análisis Postural.
- **Referencias:**
 - **Requerimientos:** UG-004, 005
 - **Casos de Uso:** UC-05
 - **Diagramas de secuencia:** SD UC-05
- **Notas:**
 - **Excepciones:**
 - Datos inválidos. *datos_sensores_actuales* estaban corruptos o ausentes.
 - **Salida:** N/A.
 - **Pre-condiciones:**
 - El ciclo de evaluación continua está activo.
 - Se han recopilado datos IMU del Anillo y la Tira.
 - **Post-condiciones:**
 - Se inició la recuperación de los datos de referencia.

CONTRATO CO-10:

- **Nombre:** Devolver_Evaluación_Postural(resultado, nivel_riesgo)
- **Responsabilidad:** Devolver el resultado del análisis postural calculado (incluyendo el resultado cualitativo y el nivel de riesgo) desde el Módulo de Análisis Postural a la App BeLine.
- **Referencias:**
 - **Requerimientos:** UG-004, 005
 - **Casos de Uso:** UC-05
 - **Diagramas de secuencia:** SD UC-05
- **Notas:**

- **Excepciones:** N/A.
- **Salida:** Una tupla que contiene el *resultado* y el correspondiente *nivel_riesgo*.
- **Pre-condiciones:**
 - La evaluación ha sido calculada correctamente (con salida) por el Módulo de Análisis Postural..
- **Post-condiciones:**
 - Los resultados de la evaluación fueron recibidos por la App BeLine, lo que activa la visualización de la pantalla de postura correcta o la pantalla de correcciones recomendadas.

CONTRATO CO-11:

- **Nombre:** Obtener_Datos_Pendientes()
- **Responsabilidad:** Devolver los datos biométricos (*datos_pendientes*) que fueron almacenados temporalmente tras un fallo de almacenamiento previo.
- **Referencias:**
 - **Requerimientos:** UG-019
 - **Casos de Uso:** UC-14
 - **Diagramas de secuencia:** SD UC-14
- **Notas:**
 - **Excepciones:**
 - Error de acceso al almacenamiento. El sistema no puede conectarse a la base de datos ni escribir datos.
 - No se encontraron datos pendientes.
 - **Salida:** Una lista de datos pendientes (*datos_pendientes*)
 - **Pre-condiciones:**
 - El ciclo (loop [mientras *datos_pendientes* = true]) está activo.
 - **Post-condiciones:**
 - Se devolvió correctamente una lista de datos pendientes (*datos_pendientes*) a la App BeLine.

CONTRATO CO-12:

- **Nombre:** Guardar_Datos_Recuperados(*datos_pendientes*)
- **Responsabilidad:** Almacenar los datos biométricos pendientes (*datos_pendientes*) recuperados de la fuente de datos temporal en el almacenamiento permanente de Métricas.
- **Referencias:**
 - **Requerimientos:** UG-019
 - **Casos de Uso:** UC-14
 - **Diagramas de secuencia:** SD UC-14
- **Notas:**
 - **Excepciones:**
 - Error de acceso al almacenamiento. El sistema no puede conectarse a la base de datos ni escribir datos.
 - **Salida:** N/A.
 - **Pre-condiciones:**

- Los datos pendientes del módulo de Datos Biométricos Brutos fueron recuperados correctamente.
- **Post-condiciones:**
 - Los datos pendientes se almacenaron correctamente en el almacenamiento de Métricas.
 - Se recibió confirmación del almacenamiento.

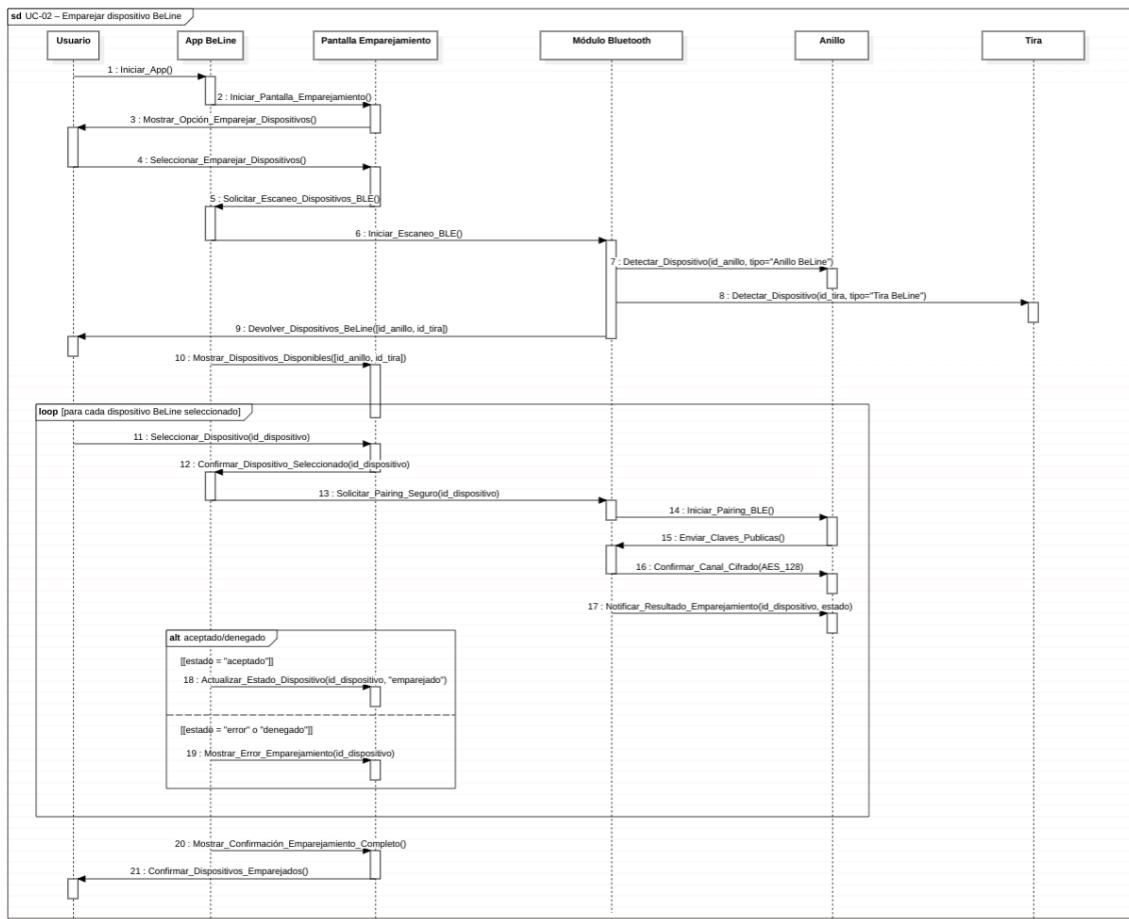
CONTRATO CO-13:

- **Nombre:** Encolar_Datos_Offline(id_dispositivo: id_tira/id_anillo, datos, timestamp)
- **Responsabilidad:** Recibir datos biométricos en crudo del Anillo y la Tira BeLine y encollarlos en el almacenamiento temporal de Datos Biométricos Brutos para su posterior sincronización.
- **Referencias:**
 - **Requerimientos:** UG-019, UG-020
 - **Casos de Uso:** UC-15
 - **Diagramas de secuencia:** SD UC-15
- **Notas:**
 - **Excepciones:**
 - Error de volumen de almacenamiento. No hay suficiente espacio en el almacenamiento temporal para guardar todos los datos en cola.
 - **Salida:** N/A.
 - **Pre-condiciones:**
 - La Tira y el Anillo BeLine están conectados.
 - El sistema está en modo offline (*modo_offline = true*).
 - **Post-condiciones:**
 - Se creó y guardó un nuevo registro que contiene los datos y la marca de tiempo en el almacenamiento temporal de Datos Biométricos Brutos.

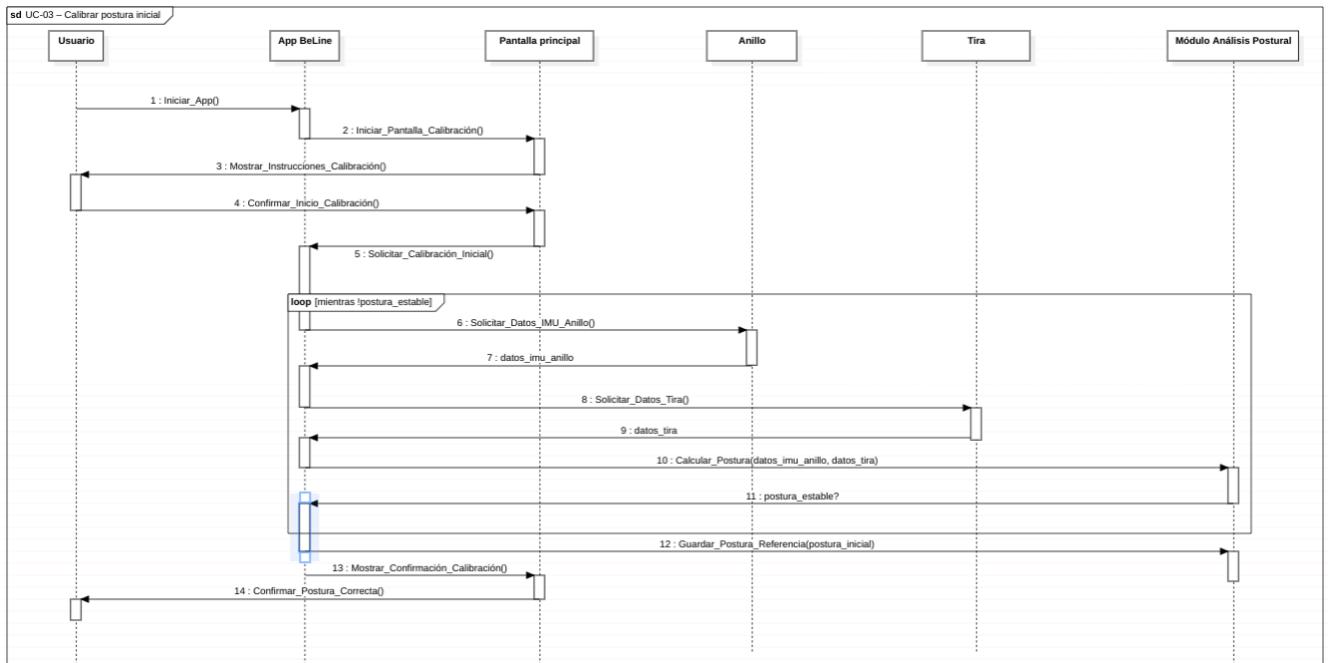
10.1.2 Diseño de la primera iteración

Diagramas de secuencia o Wireframe

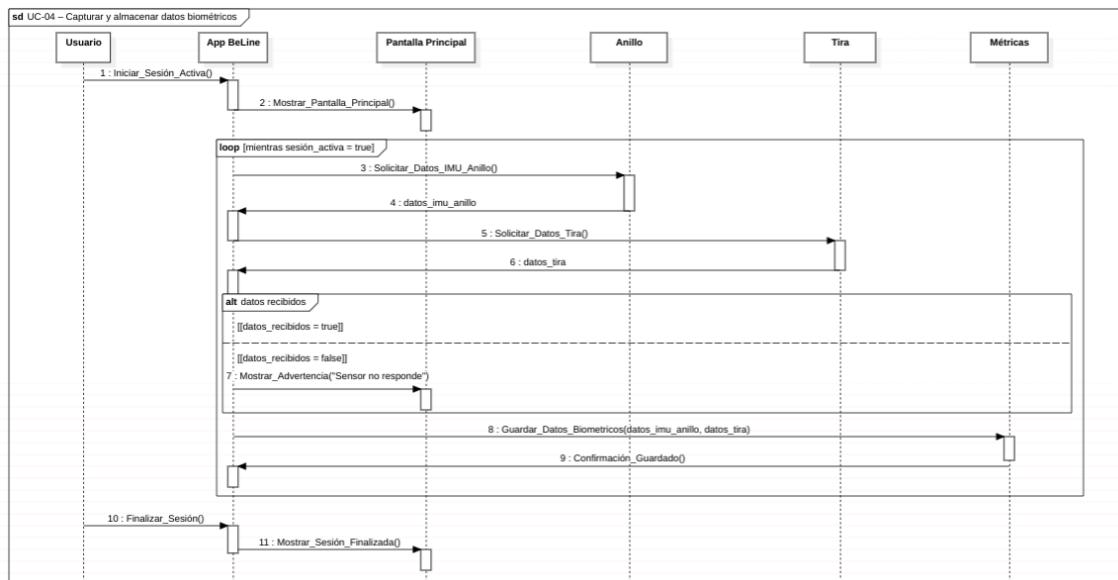
UC-02 – Emparejar dispositivo BeLine



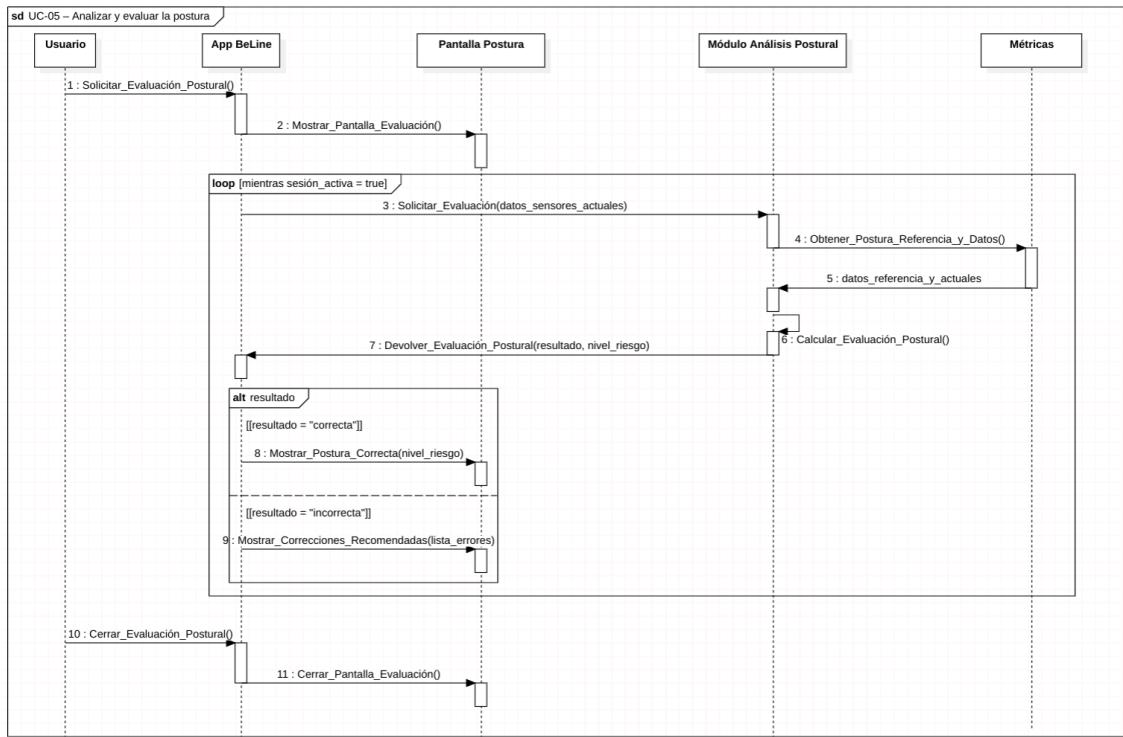
UC-03 – Calibrar postura inicial



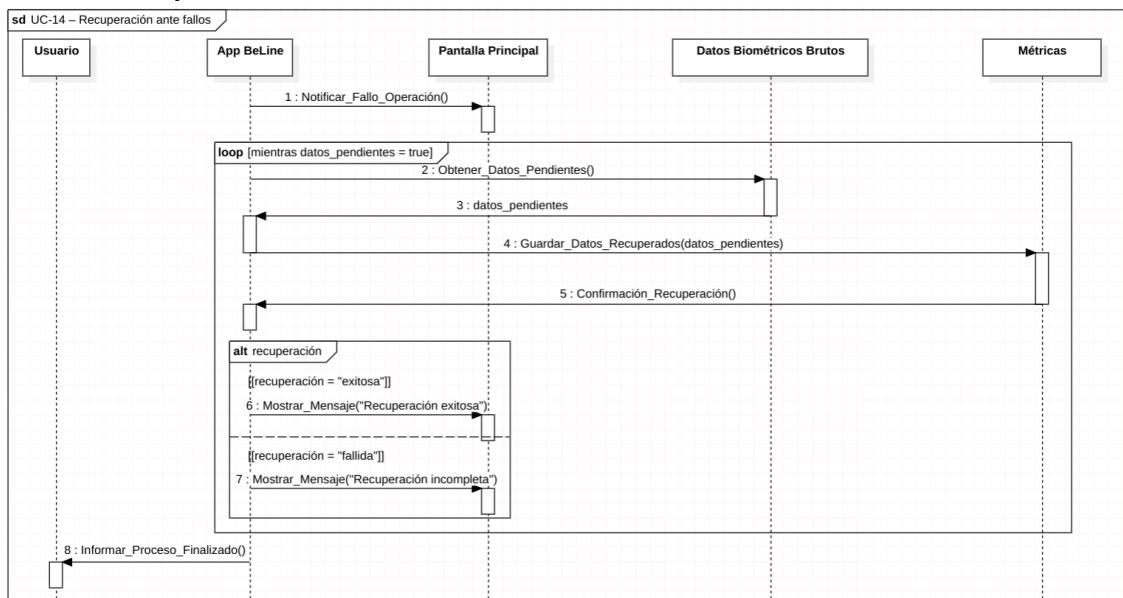
UC-04 – Capturar y almacenar datos biométricos



UC-05 – Analizar y evaluar la postura



UC-14 – Recuperación ante fallos



UC-15 – Modo sin conexión y sincronización posterior

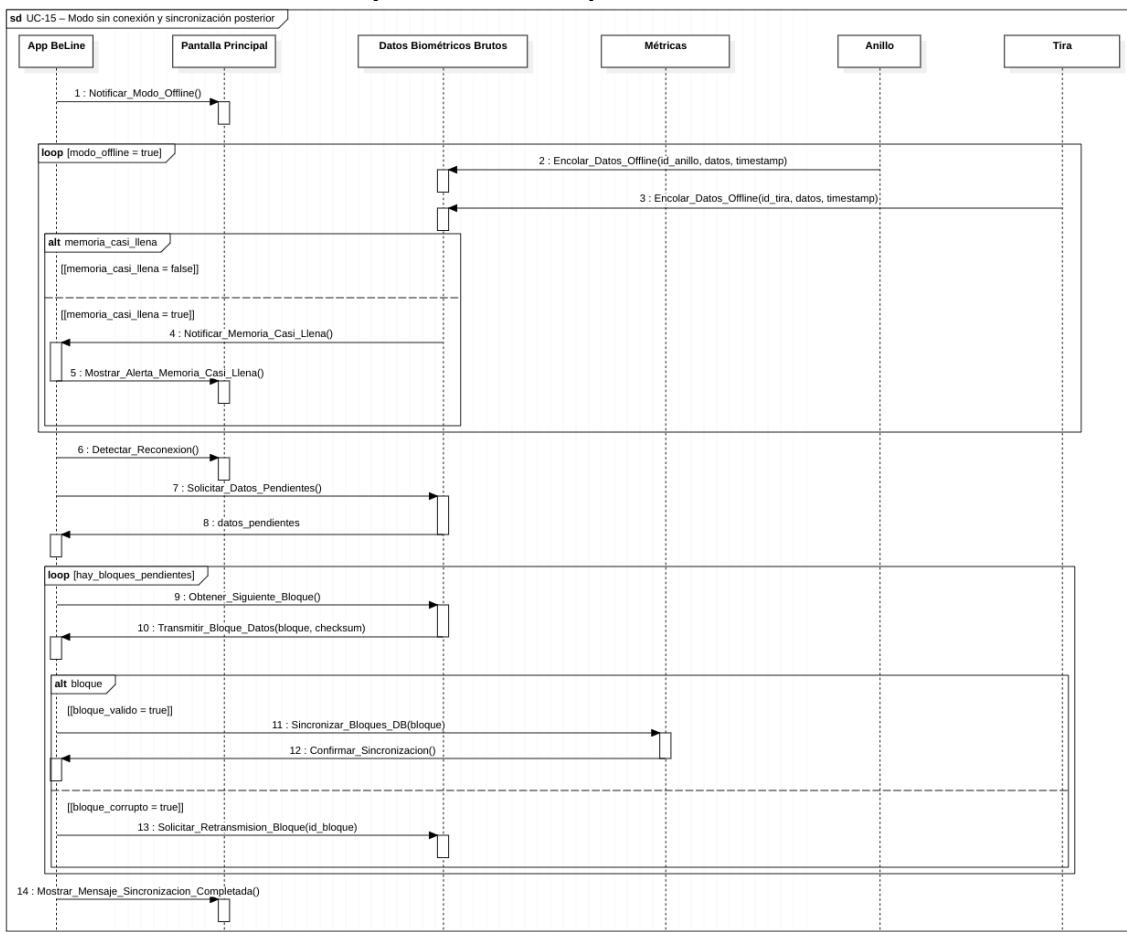
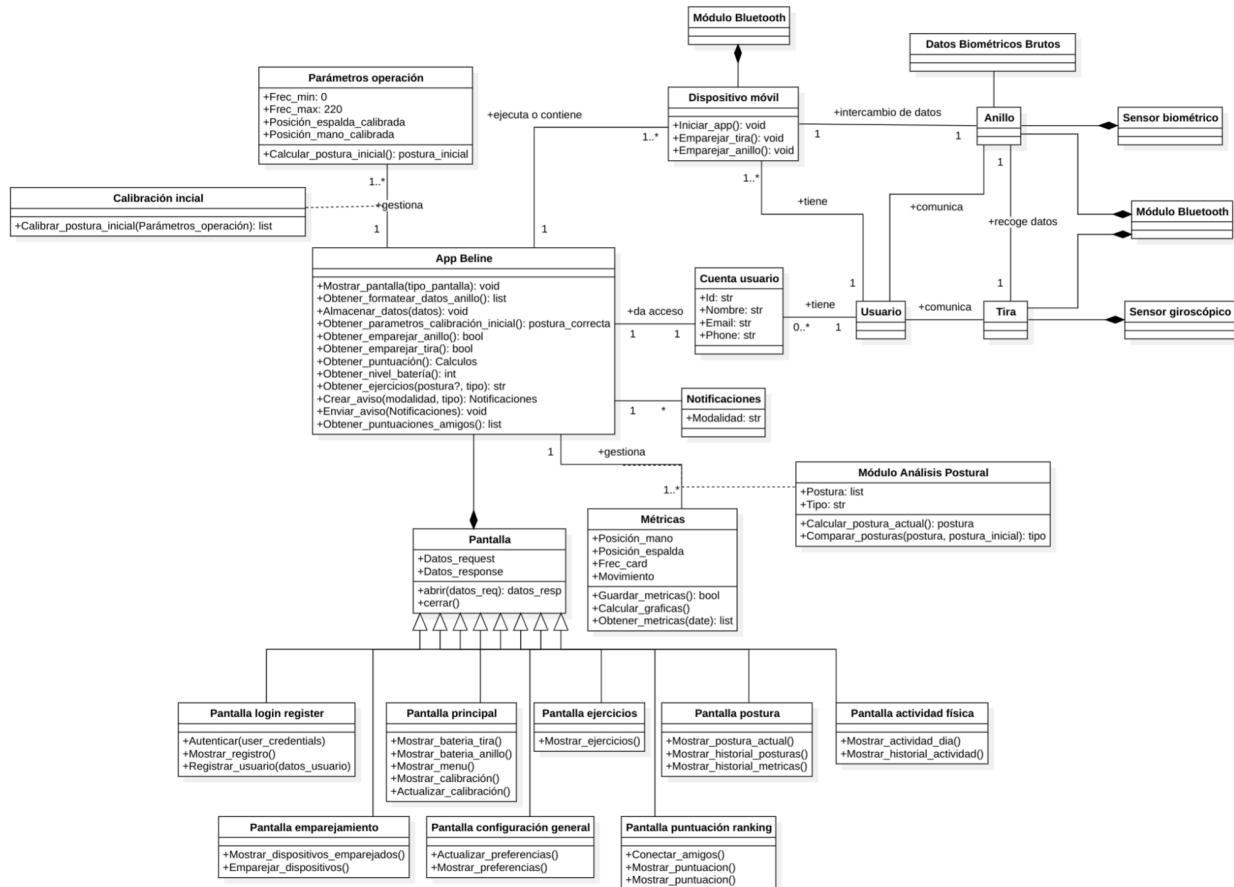


Diagrama de clases o Modelo de datos

Refinamiento del modelo conceptual



Componentes físicos:

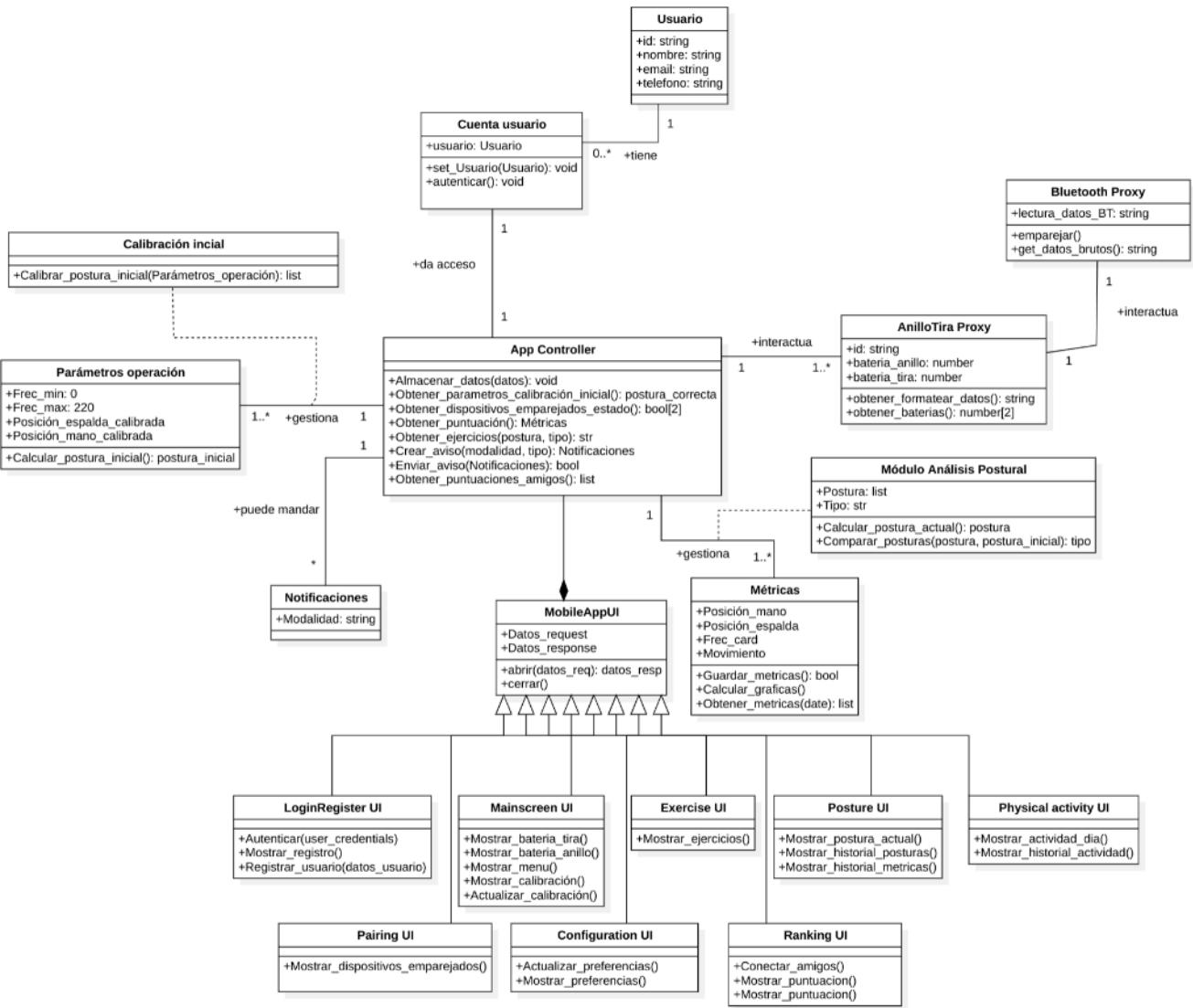
- El **anillo** incluye su propio sensor biométrico y un módulo bluetooth para poder conectarse con la tira y recopilar los datos brutos de ambos sensores (el del anillo y el de la tira a través de ella). Esta capacidad permite reunir tanto datos del anillo como de la tira.
- La **tira** también incorpora su propio sensor giroscópico encargado de detectar variaciones en la postura y el movimiento. También se conecta a través de bluetooth al anillo para poder transmitir los datos del sensor.

El **dispositivo móvil** recibe los datos y posturales biométricos brutos directamente desde el anillo mediante bluetooth para ponerlos a disposición de la app. En nuestro caso el usuario hace referencia a la persona física que utiliza el sistema a través de su propia (y única) cuenta de usuario.

App Beline: es el núcleo del sistema que recibe y procesa todos los datos brutos a través del módulo de Análisis Postural para convertirlos en métricas. También se encarga de gestionar la visualización de la información dentro de diferentes pantallas, permitir configuraciones personalizadas por parte del usuario, hacer usos de unos parámetros fijos de operación y mantener un control de las notificaciones.

- **Calibración inicial y parámetros de operación:** la calibración inicial le permite al usuario establecer una postura base de referencia desde la pantalla general de la App que se usará de referencia para evaluar si la postura del usuario a lo largo del día está alineada o desviada. Estos parámetros de operación se vuelven fijos (aunque modificables) para ese usuario una vez finalizada la calibración y almacenará todos los parámetros necesarios para realizar futuras comparaciones. Además también contienen algunos parámetros más generales como serían: heart rate entre 0-220 bpm.
- **Pantallas:**
 - **Pantalla general:** en la pantalla general es donde se muestra la posibilidad de realizar o modificar la calibración inicial, así como obtener acceso a la visualización de datos y conexiones entre pantallas.
 - **Pantalla login/registro:** permite crear una cuenta nueva de usuario o acceder a una existente. Gestiona autenticación y recuperación de acceso de forma cifrada y segura.
 - **Pantalla de emparejamiento:** guía al usuario en el proceso de conexión Bluetooth entre el móvil, el anillo y la tira. Confirma la correcta vinculación entre los tres (un sólo anillo y una sola tira por usuario).
 - **Pantalla postura:** muestra el estado postural del usuario en ese determinado momento, así como la posibilidad de visualizar su historial. Indica las desviaciones respecto a la postura calibrada y la opción de ver ejercicios correctivos.
 - **Pantalla ejercicios correctivos:** proporciona acceso a ejercicios correctivos con las indicaciones para realizarlos.
 - **Pantalla actividad física:** presenta métricas de movimiento como los pasos realizados por el usuario cada día y grafos circulares mostrando la progresión.
 - **Pantalla de puntuación/ranking:** ofrece un sistema basado en puntuaciones según el tiempo que el usuario ha mantenido su postura correcta durante el día y rankings con amigos conectados.
 - **Pantalla de configuración general:** incluye ajustes respecto a notificaciones que el usuario quiere recibir y preferencias del usuario.

Diagrama de clases final



Para realizar el diagrama de clases final, hemos aplicado el patrón Vista-Controlador sobre el modelo conceptual refinado, adaptando las pantallas para ser interfaces de usuario (UI) y la aplicación para ser el controlador.

Por otro lado, hemos aplicado un proxy a los dispositivos hardware, encapsulando la funcionalidad dentro de **AnilloTira Proxy**. De esta manera, las acciones que trabajan con los datos relacionados con el dispositivo pueden estar contenidos en su propio componente.

Para simplificar aún más, hemos aplicado otro proxy sobre las comunicaciones vía Bluetooth, de tal manera que la recepción de los datos brutos captados por el hardware, así como la responsabilidad del emparejamiento, ahora están contenidos en **Bluetooth Proxy**.

Por último, hemos eliminado algunas funciones que se realizan de forma implícita, (cómo podía ser **Mostrar_pantalla()**), así como mover ciertas funciones que corresponden dentro de los nuevos proxys (**obtener_formatear_datos()**). También hemos agrupado las funciones que actuaban sobre uno de los dos componentes en una sola, de tal forma que devuelvan una tupla que indique los valores relativos a cada componente hardware (anillo, tira). Un ejemplo de ello puede ser **Obtener_dispositivos_emparejados_estado()**, que antes estaba dividido en dos funciones.

10.2 Segunda iteración

10.2.1 Análisis de la segunda iteración

Descripción de casos de uso con formato ampliado

CASO DE USO: UC-08 - Visualizar historial de métricas

- **Actores:** Usuario, Aplicación BeLine, Servidor BeLine
- **Propósito:** Permitir que el usuario consulte su historial de métricas de salud y postura (incluyendo ritmo cardíaco, actividad, alineación postural y puntuaciones diarias) para analizar su evolución en distintos períodos de tiempo.
- **Descripción:** La aplicación solicita al Servidor BeLine los registros históricos del usuario según el periodo seleccionado (día, semana o mes). El servidor valida la identidad del usuario, recupera los datos y los envía a la aplicación. La aplicación procesa la información recibida, verifica su consistencia y genera visualizaciones como gráficos de líneas y tablas comparativas para facilitar la comprensión del progreso postural y de salud.
- **Tipo:** Secondary, Actual
- **Referencias:** UG-012, UG-013, UG-023
- **Curso típico de eventos:**
 1. El usuario accede a la sección de datos históricos dentro de la aplicación y selecciona el periodo que desea consultar (día, semana o mes).
 2. La Aplicación BeLine envía al Servidor BeLine una solicitud para obtener las métricas históricas según los filtros elegidos; el servidor valida al usuario y busca los registros en la base de datos.
 3. El Servidor BeLine envía los datos ordenados cronológicamente a la aplicación. La Aplicación BeLine verifica su integridad, normaliza valores y organiza la información por tipo de métrica.
 4. La Aplicación BeLine genera las visualizaciones correspondientes: gráficos de líneas, resúmenes por periodo, tablas de actividad y comparaciones entre métricas.
 5. La Aplicación BeLine muestra al usuario el historial seleccionado de forma clara y navegable; el usuario puede cambiar filtros, ampliar gráficos o regresar a la vista general.
- **Curso alternativos:**

- Paso 2. Sin datos disponibles: Si el Servidor BeLine no encuentra registros en el rango seleccionado, la aplicación muestra “No se han encontrado datos para este periodo” y permite elegir otro filtro.
- Paso 3. Datos corruptos: Si la Aplicación BeLine detecta valores incompletos o inconsistentes, solicita al servidor reenviar los registros afectados y completa el análisis al recibirlos.
- Paso 2. Error de conexión: Si ocurre un fallo de red, la aplicación informa al usuario y ofrece reintentar la carga sin perder los filtros seleccionados.

Contratos de operación

CONTRATO CO-14:

- **Nombre:** Guardar_Datos_Recuperados(datos_pendientes)
- **Responsabilidad:** Transmitir la lista recuperada de registros de métricas desde el contenedor Metricas hacia la Aplicación BeLine.
- **Referencias:**
 - **Requerimientos:** UG-012, UG-013, UG-023
 - **Casos de Uso:** UC-08
 - **Diagramas de secuencia:** SD UC-08
- **Notas:**
 - **Excepciones:**
 - N/A.
 - **Salida:** Una lista de registros históricos de métricas (*listaMetricas*).
 - **Pre-condiciones:**
 - El Servidor BeLine ha consultado correctamente los registros necesarios (*periodo*, *tipoMetrica*) y validado el *usuarioID*.
 - **Post-condiciones:**
 - La *listaMetricas* fue recibida correctamente por la Aplicación BeLine, habilitando el paso de procesamiento de datos.

CONTRATO CO-15:

- **Nombre:** Devolver_Sin_Datos(datos_pendientes)
- **Responsabilidad:** Transmitir un indicador booleano desde el contenedor Metricas hacia la Aplicación BeLine que señale que no se encontraron registros.
- **Referencias:**
 - **Requerimientos:** UG-012, UG-013, UG-023
 - **Casos de Uso:** UC-08
 - **Diagramas de secuencia:** SD UC-08
- **Notas:**
 - **Excepciones:**
 - N/A.
 - **Salida:** Indicador booleano que muestra que no se encontraron registros.
 - **Pre-condiciones:**
 - El Servidor BeLine consultó el conjunto de datos para el *usuarioID*, *periodo* y *tipoMetrica* dados, y determinó que no hay registros disponibles.

- **Post-condiciones:**
 - El indicador fue recibido por la Aplicación BeLine.

10.2.2 Diseño de la segunda iteración

Diagramas de secuencia o Wireframe

UC-08 - Visualizar historial de métricas

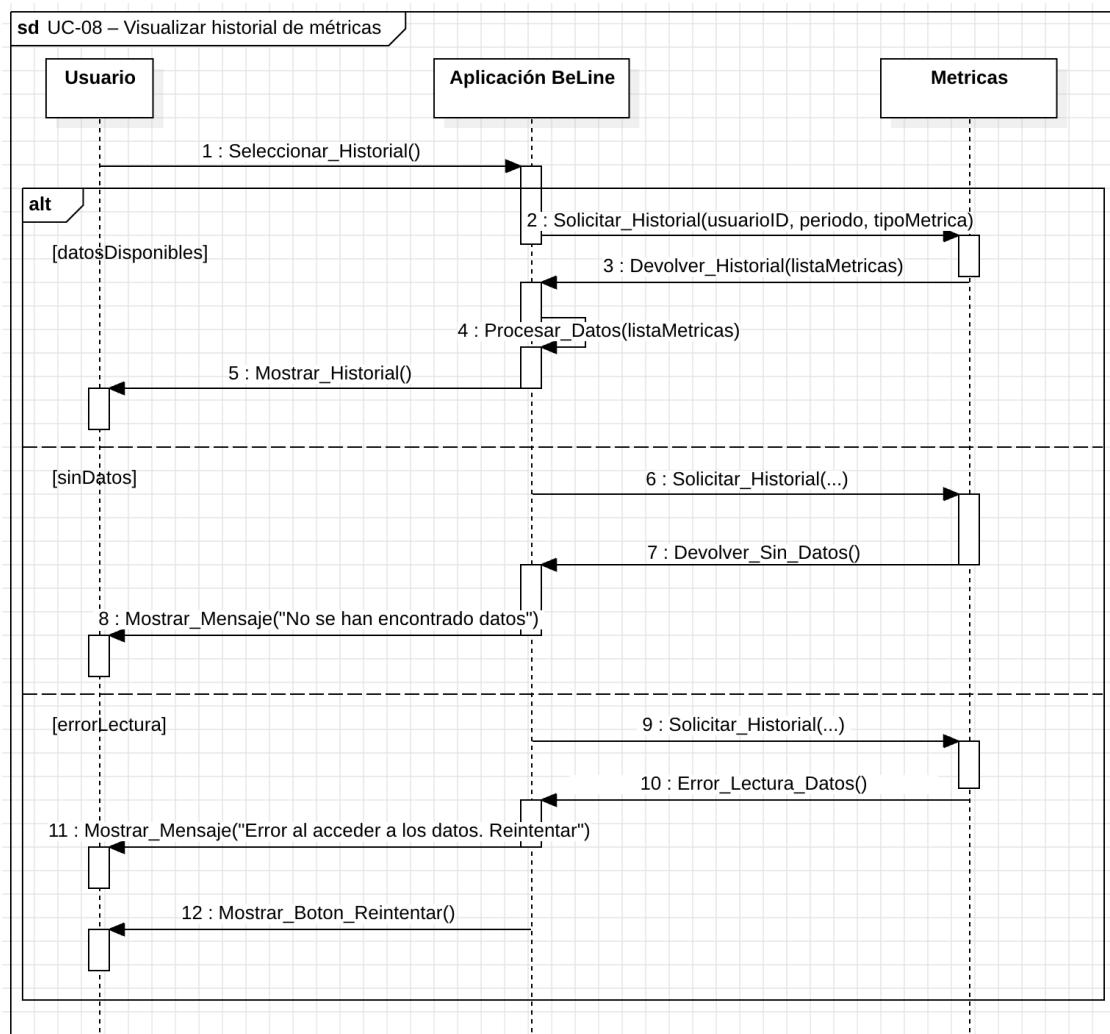


Diagrama de clases o Modelo de datos

El diagrama es el mismo que en la primera iteración.

11. Ejecución del plan de calidad

Durante la ejecución del proyecto BeLine, se ha llevado a cabo la revisión y verificación del cumplimiento de los requisitos definidos en el Plan de Calidad y del Estudio de Viabilidad.

El objetivo de este proceso es verificar que las funcionalidades, el diseño y la documentación del sistema cumplieran con los estándares establecidos en las fases de análisis y planificación.

Revisión del Estudio de Viabilidad del Sistema:

Actividad revisada	Resultado	Observaciones
Identificación de absolutamente todos los requisitos del usuario.	✓	Requisitos UG-001 a UG-016 correctamente definidos.
Coherencia entre el contenido del documento y su objetivo.	✓	La documentación está alineada con el propósito de BeLine.
Cada requisito describe la funcionalidad que le corresponde.	✓	No se ha realizado una matriz de trazabilidad, por lo que no se puede comprobar.
Correspondencia entre los requisitos del documento y los requisitos obtenidos del usuario, por lo que la especificación de requisitos es completa.	✗	No se ha realizado una matriz de trazabilidad, por lo que no se puede comprobar.
Descripción de los requisitos en un lenguaje claro, sin ambigüedades y, por tanto, preciso.	✓	No hay ambigüedades detectadas.
El estudio de viabilidad es autodescriptivo, ya que se describe su estructura y contenido.	✓	Se cumplen los criterios del plan.
Se llevará a cabo una matriz de trazabilidad de requisitos para comprobar que todos los requisitos de usuario tienen asociado al menos un requisito de software y, por tanto, están presentes en el diseño del sistema.	✗	No se ha realizado una matriz de trazabilidad.

Revisión del Diagrama de Casos de Uso:

Actividad revisada	Resultado	Observaciones
El diagrama de casos de uso describe el comportamiento del sistema, es decir, la funcionalidad completa del proyecto de software que se va a desarrollar.	✓	Se representan actores, dispositivos y app.
El diagrama de casos de uso incluye todos los casos de uso identificados que	✓	

representan todas las funcionalidades del sistema		
El diagrama de casos de uso incluye a todos los actores identificados e implicados en el sistema.	✓	Usuario, anillo, tira y app.
El diagrama de casos de uso incluye todas las dependencias y relaciones entre actores y casos de uso.	✓	Incluye relaciones de inclusión y extensión.
El diagrama de casos de uso se ajusta a la notación gráfica definida en el lenguaje de modelado UML.	✓	Siguiendo la notación UML establecida correcta, el diagrama ha sido realizado en Draw.io.
El modelo de casos de uso incluye un glosario de términos que describe la terminología utilizada.	✗	No se incluye glosario de términos.

Revisión de Casos de Uso de Alto Nivel:

Actividad revisada	Resultado	Observaciones
Los casos de uso de alto nivel contienen el nombre, los actores, la descripción y el tipo de caso de uso.	✓	Los casos de uso descritos tienen nombre, incluyen los actores involucrados, una breve descripción, el tipo y referencias
Cada caso de uso describe cómo alcanzar un único objetivo, es decir, describe una característica del sistema.	✓	Cada caso de uso describe una actividad específica
Cada caso de uso contiene una descripción textual de la funcionalidad asociada con el nivel de detalle adecuado, incluidas las formas en que los actores previstos podrían trabajar con el sistema. La descripción utilizará el lenguaje del usuario final.	✓	Las descripciones cumplen con los requisitos.
Los casos de uso no describen la funcionalidad interna del sistema ni explican cómo se implementará. No incluyen jerga técnica.	✓	No se incluyen requisitos no funcionales dentro de los casos de uso del sistema.

Cada caso de uso muestra los pasos que sigue el actor para realizar una operación.	✓	Se incluye una descripción de los pasos a seguir
Los casos de uso se ajustan a la notación gráfica definida en el lenguaje de modelado UML	✗	No se ajustan los nombres

Durante la ejecución del proyecto BeLine, se ha llevado a cabo la ejecución del Plan de Gestión de la Configuración (CMP) para asegurar la trazabilidad, integridad y coherencia de todos los componentes de hardware, software y documentación.

Revisión del Plan de Gestión de la Configuración:

Actividad revisada	Resultado	Observaciones
El proyecto incluye un Plan de Gestión de la Configuración para el control y gestión de los cambios en el que se establecen las actividades a realizar que permiten el control y gestión de los cambios en el proyecto	✓	El proyecto cuenta con un CMP documentado y aprobado.
El Plan de Gestión de la Configuración cumple con IEEE Std. 828 - 2005 y ANSI/IEEE Std. 1042 - 1987.	✓	Verificado por Responsable de Calidad
La gestión de la configuración definida en el SCM se lleva a cabo durante todas las fases de desarrollo del proyecto de software, incluidos el mantenimiento y el control de cambios	✓	
El SCM describe un mecanismo de control de cambios y versiones que garantiza la producción de software de calidad.	✓	Git, Hojas de Cálculo y Google Docs utilizados
El SCM incluye el procedimiento para generar la documentación necesaria para el registro y seguimiento de los cambios que se produzcan durante el desarrollo del proyecto.	✓	Solicitudes de cambio y Pull Requests documentadas

Revisión de la Estimación

Actividad revisada	Resultado	Observaciones
El método utilizado para estimar el esfuerzo de desarrollo del proyecto de software utiliza métricas orientadas al tamaño basadas en puntos de casos de uso.	✓	
Antes de cada iteración, verifique que la estimación se ha realizado teniendo en cuenta los casos de uso incluidos en la estimación.	✓	Verificado por Responsable de Calidad
Los puntos de uso para cada una de las iteraciones se han calculado siguiendo el procedimiento establecido para este método de estimación, que incluye los siguientes pasos:		
	Clasificar cada iteración entre actor y caos de uso en función de su complejidad y asignarle un peso en función de la misma.	✓ Se calculó en una hoja de cálculo.
	Calcule la complejidad de cada caso de uso en función del número de transacciones o pasos del caso.	✓ Se tuvo en cuenta la complejidad de los requisitos.
	Calcular los puntos de caso de uso no ajustados de la iteración.	✓ Se calcularon en una hoja de cálculo.
	Calcular los factores de complejidad técnica y medioambiental.	✓ Todos los factores se tuvieron en cuenta, siendo asignados un valor en el rango 0-5.
	Calcular los puntos de casos de uso ajustados	✓ Los cálculos fueron realizados en una hoja de cálculo.
Una vez obtenidos los puntos de casos de uso para una iteración, compruebe que se	✓ Se llevó a cabo el cálculo del esfuerzo	

ha calculado a partir de ellos el esfuerzo correspondiente necesario para llevarlos a cabo en esa iteración.		correspondiente en cada iteración siguiendo el método RUP.
--	--	--

Revisión de la planificación

Actividad revisada	Resultado	Observaciones
Se ha realizado una priorización de los casos de uso a desarrollar y se han definido las iteraciones que conformarán el desarrollo completo del software y los casos de uso incluidos en cada una de ellas.	✓	Se realizó una priorización de los casos y en qué iteración se desarrollan.
Se ha realizado una estimación de cada iteración basada en casos de uso. A partir de esta estimación, se llevará a cabo la planificación.	✓	Se realizó un estudio de estimación en puntos previos a la planificación.
Antes de iniciar una iteración, se realizará una planificación de la misma basada en la estimación del esfuerzo necesario en función de los puntos de casos de uso.	✓	Se realizó la planificación de la iteración siguiendo los pasos requeridos.
La planificación prevista para el desarrollo del proyecto de software se adaptará y actualizará a medida que avance el proyecto.		
La planificación incluye cuántas personas deben participar en el equipo del proyecto, qué conocimientos técnicos se necesitan, cuándo aumentar el número de personas y quién participará.	✓	La planificación lo incluye.
La planificación realizada define cómo se organizará el equipo que trabajará en el proyecto de desarrollo de software.	✓	
La planificación sigue la metodología aplicada al proyecto de desarrollo de software que es, en este caso, incremental iterativa basada en casos de uso.	✓	Sigue la metodología presentada en el documento.

Se incluye un diagrama de Gantt que representa todas las actividades que deben realizarse a lo largo del periodo de desarrollo del proyecto. El diagrama conecta las distintas actividades en función de sus relaciones de precedencia y define los recursos y tiempos estimados para cada una de ellas.	✓	El diagrama está incluido
El diagrama de Gantt refleja las tareas y las fechas clave, los hitos y la dependencia entre tareas.	✓	El diagrama refleja estos requisitos.

Revisión del plan de pruebas → NO SOLICITADA

Actividad revisada	Resultado	Observaciones
Debe comprobarse que existen normas de realización de las pruebas que permitan verificar que éstas se han llevado a cabo, así como indicar cómo actuar en caso de diferencias entre el resultado esperado y el obtenido.		
Debe realizarse una matriz de trazabilidad para garantizar que existen pruebas para verificar todos los requisitos del software.		

Revisión de Casos de Uso en Formato Expandido

Actividad revisada	Resultado	Observaciones
A partir de cada caso de uso de alto nivel, se ha construido un caso de uso ampliado, en cada iteración.	✗	Solo se han llevado a desarrollo 5 casos de uso ampliado en la primera interacción y 1 en la segunda.
Cada caso de uso ampliado se compone de dos secciones, la cabecera que incluye el nombre, los actores, la descripción y el tipo de caso de uso, y el cuerpo que describe los sucesos típicos y las alternativas a los sucesos típicos.	✓	Los casos de uso ampliado están compuestos por ambas secciones con todo los requisitos mencionados.

Los casos de uso ampliados definen el iniciador del caso de uso.	✓	El iniciador se especifica claramente en la sección "Actores" y, cuando no es el único actor, se etiqueta como "(iniciador)" junto al actor responsable de comenzar el proceso.
El cuerpo del caso de uso consta de dos columnas que describen las acciones del actor y las respuestas del sistema a las mismas.	✗	No se encuentra en dos columnas, sigue un formato de lista numerada secuencial.

Revisión de Modelo Conceptual del Análisis

Actividad revisada	Resultado	Observaciones
El modelo de análisis representa los aspectos del problema de forma cercana a los conceptos del dominio del problema y describe las principales características del sistema. Se validará el modelo de análisis realizado en cada una de las iteraciones que componen el proyecto.	✓	El modelo incluye conceptos clave del dominio de salud y postura: Anillo, Tira, App BeLine, Usuario, Métricas, Análisis Postural, Calibración inicial, y Pantalla.
El modelo conceptual no incluye las decisiones de aplicación. También se comprobará que es independiente de la aplicación.	✓	Las clases representan las principales funcionalidades (Módulo Análisis Postural, Notificaciones) y la estructura de datos (Datos biométricos brutos, Métricas). Las relaciones muestran cómo interactúan (la App BeLine gestiona las Notificaciones y las Métricas).
El modelo conceptual se ajusta a la notación gráfica del lenguaje de modelado UML. También debe comprobar que la notación tiene el nivel de detalle necesario para representar el problema, sin estar sobrecargada.	✓	El modelo se centra en qué entidades existen (clases) y cómo se relacionan, no se incluyen detalles de carácter técnico.

<p>El modelo conceptual se ha realizado mediante un modelo de objetos o diagrama de clases (sin métodos) que define las propiedades del sistema. Las entidades y las relaciones entre ellas se han identificado para cada iteración.</p>	<input checked="" type="checkbox"/>	<p>Las clases como <i>Usuario</i>, <i>Métricas</i>, <i>Anillo</i> y <i>Tira</i> son conceptos de dominio que existirían independientemente de la tecnología específica utilizada para construir la App BeLine.</p>
<p>Las métricas de calidad que deben aplicarse al modelo conceptual resultante del análisis en cada iteración son las siguientes:</p>		
<p>Calidad semántica: correspondencia entre el modelo y el dominio, es decir, el modelo refleja el dominio. Se comprobará la validez del modelo, es decir, que todos los hechos incluidos en el modelo son correctos y pertinentes para el dominio.</p>	<input checked="" type="checkbox"/>	<p>El modelo refleja correctamente la realidad del dominio BeLine.</p>
<p>Compleitud: el modelo se comprobará para garantizar que todos los hechos son correctos y pertinentes para el dominio.</p>	<input checked="" type="checkbox"/>	<p>El modelo es completo y representa de tal manera el dominio.</p>
<p>Calidad del lenguaje: el lenguaje de modelado utilizado para capturar el dominio es un lenguaje fácil de entender por todos los participantes. La formalización del lenguaje permite la ejecución del sistema.</p>	<input checked="" type="checkbox"/>	<p>El diagrama de clases es una de las representaciones más sencillas de UML, lo que facilita su comprensión tanto para desarrolladores como para analistas y usuarios de negocio. Se utiliza un lenguaje natural.</p>
<p>Calidad sintáctica: existe una correspondencia entre la externalización del modelo y la extensión del lenguaje en el que está escrito el modelo.</p>	<input checked="" type="checkbox"/>	<p>La notación utilizada (clases, asociaciones, herencia) se ajusta perfectamente a las reglas sintácticas del lenguaje UML.</p>

Revisión de Contratos de Operación

Actividad revisada	Resultado	Observaciones
Para cada caso de uso, debe existir un contrato de operación para cada acción del actor.	X	Se han elaborado contratos para aquellas operaciones consideradas más prioritarias.
Cada contrato de explotación constará de los siguientes campos: nombre, responsabilidades, referencias cruzadas, notas, excepciones, salida, condiciones previas y condiciones posteriores.	✓	Cada contrato tiene los campos requeridos.
Las referencias cruzadas en el contrato corresponderán a referencias a los requisitos definidos en el proyecto que se resuelven con el caso de uso al que pertenece el contrato de explotación.	✓	Las referencias son a requerimientos, casos de uso y diagrama de secuencia.

Revisión del Diagrama de Clases

Actividad revisada	Resultado	Observaciones
Se realizarán diagramas de clases para cada iteración con UML y el diseño será totalmente independiente de la implementación.	X	Se ha realizado un único diagrama de clases, uniendo ambas interacciones.
Se medirá la comprensibilidad del modelo o facilidad con la que se puede entender el diagrama de clases, la analizabilidad del modelo o facilidad que ofrece el diagrama de clases para descubrir sus deficiencias o errores, y la modificabilidad del diagrama o facilidad que ofrece el diagrama para realizar una modificación especificada, ya sea por error, por un concepto no tenido en cuenta o por un cambio en los requisitos. Se utilizarán las siguientes métricas para medir la complejidad estructural de los diagramas de clases:		
Número de clases: número total de clases	22	El tamaño es manejable
Número de atributos: número total de atributos.	18	El bajo número de características por clase (en promedio).

Número de métodos: número total de métodos.	44	El número es normal en el contexto de la aplicación
Número de asociaciones: número total de asociaciones.	12	
Número de agregaciones: número total de relaciones de agregación.	6	
Número de dependencias: número total de relaciones de dependencia.	2	
Número de generalizaciones: número total de relaciones de generalización.	8	
Número de jerarquías de generalización: número total de jerarquías de generalización	1	Pantalla y sus subclases
WMC: métodos ponderados por clase, según su complejidad.	44	Complejidad = 1
DIT máximo: es el valor máximo de DIT obtenido para cada clase en un diagrama de clases. Para una clase dentro de una jerarquía de generalización, es la longitud del camino más largo desde la clase hasta la raíz de la jerarquía.	1	
HAgg máximo: es el valor HAgg máximo obtenido para cada clase en el diagrama de clases. Para una clase dentro de una jerarquía de agregación es la longitud del camino más largo desde la clase hasta las hojas.	1	

El modelo presenta una complejidad estructural baja a moderada según las métricas calculadas (principalmente por el bajo WMC, DIT Máximo de 1, y bajo NAsso).

Revisión de Diagramas de Secuencia

Actividad revisada	Resultado	Observaciones
Para cada caso de uso, se han diseñado diagramas de secuencia que definen tanto el	X	Solo se han realizado diagramas de flujo para

curso típico como los cursos atípicos de los eventos definidos en ellos.		aquellos casos de uso con formato ampliado
Los diagramas de secuencia muestran la interacción representada por la secuencia de mensajes entre las instancias de clase y los actores. Los diagramas muestran instancias y eventos que describen la interacción entre las clases.	✓	Los diagramas de secuencia son una representación de la secuencia de mensajes cubriendo en detalle el flujo de eventos proporcionados.
El tiempo fluye por los diagramas y muestra el flujo de control de un participante a otro.	✓	El tiempo fluye y los eventos dependen del mismo.
En la definición de los diagramas se sigue la notación UML. Los elementos incluidos en el diagrama de secuencia son:		
Nombre del diagrama de secuencia.	✓	Cada diagrama de flujo tiene el nombre.
Líneas de vida para actores e instancias de clase.	✓	Cada actor que aparece tiene una línea de vida, también las instancias de clase.
Mensajes entre instancias que definen el método al que llama el mensaje en la línea de vida receptora. Además, la línea receptora está vinculada a una interfaz o clase.	✓	Cada línea tiene un método al que se llame.
Los bucles indican el número de veces que se ejecuta el bucle, si se conoce.	✗	

Revisión de Diagramas de Estado → NO SOLICITADA

Actividad revisada	Resultado	Observaciones
Los diagramas de estado definidos describen el comportamiento del sistema, y cada diagrama muestra el comportamiento de un único objeto durante todo su ciclo de vida.		
Los diagramas de estado contienen estados y transiciones, y las transiciones entre ellos		

incluyen los eventos o acciones correspondientes.		
El diagrama de estados muestra todos los posibles estados por los que pasa el objeto durante su vida en la aplicación como resultado de los eventos que le llegan.		
Hay un estado inicial y un estado final y todos los estados representados en el diagrama son accesibles.		

12. Ejecución del plan de gestión de la configuración

Para la gestión de código se utiliza GitHub, pero debido a que no se desarrolla código en esta práctica no se incluirán registros.

Para la identificación y asignación de tareas se ha utilizado Hojas de cálculo y en las imágenes podemos observar su funcionamiento:

Tr	REVISIONES	ESTADO	ENCARGADO
5.4	Necesita cambios	Anastasiia Sadova	Nicola Cindea Iciar G
5.5	No necesita cambios		
5.6	No necesita cambios		
5.7	No necesita cambios		
6.2	Necesita cambios	Nicola Cindea	
9.1.1	Necesita cambios	Aesling Florencio	Nicola Cindea
9.1.2	Necesita cambios	Nicola Cindea	
9.1.3	No necesita cambios		
9.1.4	No necesita cambios		
11	No necesita cambios		
12	No necesita cambios		

Tr	REVISIONES	ESTADO	ENCARGADO
5.4	Cambios hechos	Anastasiia Sadova	Nicola Cindea Iciar G
5.5	No necesita cambios		
5.6	No necesita cambios		
5.7	No necesita cambios		
6.2	Cambios hechos	Nicola Cindea	
9.1.1	Cambios hechos	Aesling Florencio	Nicola Cindea
9.1.2	Cambios hechos	Nicola Cindea	
9.1.3	No necesita cambios		
9.1.4	No necesita cambios		
11	No necesita cambios		
12	No necesita cambios		

APARTADO A HACER	Columna 1	ESTADO	ASIGNADO
9.2	Hecho	Aesling Florencio Nicola Cindea	
9.3	Hecho	Anastasiia Sadova	
9.4	Hecho	Anastasiia Sadova	
11 (relacionado con el punto 9)	Hecho	Iciar Garcia	
12 (relacionado con el punto 9)	Sin hacer	Iciar Garcia	
7	Hecho	Nicola Cindea	
11 (relacionado con el punto 7)	Hecho	Iciar Garcia	
12 (relacionado con el punto 7)	Sin hacer	Iciar Garcia	
8	En proceso	Salvador Ayala	
11 (relacionado con el punto 8)	En proceso	Iciar Garcia	
12 (relacionado con el punto 8)	Sin hacer	Iciar Garcia	

APARTADO A HACER	Columna 1	ESTADO	ASIGNADO
9.2	Hecho	Aesling Florencio Nicola Cindea	
9.3	Hecho	Anastasiia Sadova	
9.4	Hecho	Anastasiia Sadova	
11 (relacionado con el punto 9)	Hecho	Iciar Garcia	
12 (relacionado con el punto 9)	Hecho	Iciar Garcia	
7	Hecho	Nicola Cindea	
11 (relacionado con el punto 7)	Hecho	Iciar Garcia	
12 (relacionado con el punto 7)	Hecho	Iciar Garcia	
8	Hecho	Salvador Ayala	
11 (relacionado con el punto 8)	Hecho	Iciar Garcia	
12 (relacionado con el punto 8)	Hecho	Iciar Garcia	

Para la siguiente entrega se decidió no realizar ninguna revisión de los contenidos anteriores, por lo que no hay una organización de estos.

APARTADO A HACER	Columna 1	ESTADO	ASIGNADO
Modelo Conceptual		Hecho	Iciar García Nicola Cindea Salva
Diagrama de clases		Hecho	Iciar García Nicola Cindea Salva
Itera 1 CU formato expandic		Hecho	Aesling Florencio Anastasiia Sadov
Itera 2 CU formato expandic		Hecho	Aesling Florencio Anastasiia Sadov
Contratos		Hecho	Anastasiia Sadova
Diagrama de flujo		Hecho	Aesling Florencio
Diagrama de clases final		Hecho	Salvador Ayala
11		Hecho	Iciar García
12		Hecho	Iciar García

Como podemos observar, se identifican las tareas a realizar (en este caso apartados del propio documento dossier) y en cada tarea asignada elaborar cada subtarea como completar una hoja de cálculo o crear un diagrama. De esta manera es sencillo acceder y realizar cambios para todos los participantes.

Para la gestión de cambios de los documentos se utilizan los historiales de los documentos. Se puede ver en las siguientes imágenes:

The screenshot shows a document history interface with the following entries:

- 14 de noviembre, 15:28**: ANASTASIIA SADOVA (purple dot)
- 14 de noviembre, 11:14**: ANASTASIIA SADOVA (purple dot)
- > 14 de noviembre, 2:23**: NICOLA STEFANIA CINDEA (green dot)
- Lunes** (date header)
- 10 de noviembre, 18:40**: AISLING FLORENCIO PIMENTEL (red dot)
- > 10 de noviembre, 16:36**: SALVADOR AYALA IGLESIAS (blue dot)
- > 10 de noviembre, 16:11**: ICIAR GARCIA IZQUIERDO (orange dot), SALVADOR AYALA IGLESIAS (blue dot)
- > 13 de noviembre, 16:39**: ANASTASIIA SADOVA (purple dot), NICOLA STEFANIA CINDEA (green dot), ICIAR GARCIA IZQUIERDO (orange dot), SALVADOR AYALA IGLESIAS (blue dot)
- > 13 de noviembre, 15:43**: SALVADOR AYALA IGLESIAS (blue dot), ICIAR GARCIA IZQUIERDO (orange dot), ANASTASIIA SADOVA (purple dot)
- 13 de noviembre, 14:40**: AISLING FLORENCIO PIMENTEL (red dot)
- 13 de noviembre, 11:07**: AISLING FLORENCIO PIMENTEL (red dot)

- | | |
|--|---|
| <ul style="list-style-type: none"> > 5 de diciembre, 20:11
● SALVADOR AYALA IGLESIAS > 5 de diciembre, 20:01
● SALVADOR AYALA IGLESIAS > 5 de diciembre, 19:17
● ICIAR GARCIA IZQUIERDO > 5 de diciembre, 18:50
● ICIAR GARCIA IZQUIERDO <p>4 de diciembre, 17:13
● ICIAR GARCIA IZQUIERDO</p> <ul style="list-style-type: none"> > 4 de diciembre, 15:41
● ANASTASIIA SADOVA
● ICIAR GARCIA IZQUIERDO
● NICOLA STEFANIA CINDEA | <ul style="list-style-type: none"> > 27 de noviembre, 9:50
● ANASTASIIA SADOVA > 26 de noviembre, 20:18
● ANASTASIIA SADOVA > 26 de noviembre, 13:29
● ANASTASIIA SADOVA <p>26 de noviembre, 12:27 :
● ANASTASIIA SADOVA</p> <ul style="list-style-type: none"> > 26 de noviembre, 10:14
● AISLING FLORENCIO PIMENTEL > 26 de noviembre, 9:35
● AISLING FLORENCIO PIMENTEL |
|--|---|

13. Conclusiones generales

El desarrollo de este proyecto nos ha ayudado a entender la importancia de aplicar técnicas de dirección de proyectos software a la hora de afrontar proyectos de este tamaño. El aplicar una metodología clara y ágil, comprometiéndose a ella y siguiéndola de principio a fin, nos ha permitido trabajar de manera más eficiente y, sobre todo, más organizada.

Además, creemos que la existencia de un dossier que detalle el proyecto es crucial para mantener un correcto control y seguimiento de este. A pesar de que puede ser un paso más a tener en cuenta a la hora de desarrollar un proyecto, creemos que contribuye positivamente a la organización y entendimiento del sistema por parte de los integrantes del equipo.

Por otro lado, creemos que una de las mayores dificultades que se debe afrontar a la hora de realizar proyectos de una complejidad considerable es la coordinación y el liderazgo. A pesar de ser sólo cinco integrantes, hemos notado cómo de difícil puede llegar a resultar la coordinación entre miembros del equipo. La existencia de unos roles bien definidos, sobre todo el de Project Manager y Responsable de Calidad, han ayudado enormemente a coordinar los esfuerzos del equipo y a lograr llegar a las fechas establecidas por el cronograma.

Consideramos que este proyecto nos ha aportado una formación y una visión esencial para nuestro futuro laboral como profesionales del entorno del desarrollo de software.

14. Conclusiones individuales de cada miembro

Conclusiones: Salvador Ayala

Este proyecto me ha permitido entender la importancia de la metodología y los documentos técnicos, cómo puede ser este dossier, para el desarrollo de un proyecto software. Considero que me ha permitido consolidar conocimientos que no había llegado a aplicar en ningún caso práctico.

Además, creo que he podido explorar una faceta de mi personalidad al adoptar el rol de Project Manager. El forzarme a liderar el equipo y dictar un curso de acción a seguir, asignando tareas a cada miembro, me ha permitido aprender a ser mejor líder.

Creo que este proyecto ha sido enormemente beneficioso para mí y para mis compañeros. Y, personalmente, agradezco el entendimiento y la paciencia de los profesores. Parte del aprendizaje que he adquirido se debe a su labor al responder dudas repetidas o demasiado concretas que podrían resultar cansadas. Gracias.

Conclusiones: Iciar García

Antes de comenzar este proyecto no le daba demasiada importancia a la organización previa que requiere el desarrollo de un proyecto. Sin embargo, tras realizar las diferentes secciones, como el presupuesto, la estimación y la planificación he entendido que para que un proyecto sea exitoso es de vital importancia la organización del cuerpo de trabajo.

De la misma manera, he conseguido apreciar aspectos de este ámbito que sin realizar este trabajo no habría conocido, como el Plan de Calidad, del cual me encargué yo como responsable de calidad. En otros proyectos siempre ha sido importante realizar testeos sobre el código, para comprobar que realmente cumple con los requisitos, pero nunca había llevado a cabo un test, por así decirlo, de elementos redactados como lo que se ha llevado a cabo en este dossier.

Me gustaría concluir diciendo que este proyecto ha sido muy enriquecedor y será de gran ayuda no tan solo para otros trabajos en la universidad, sino también en proyectos personales y laborales. La ayuda ofrecida por el profesorado también ha contribuido para que este proceso de aprendizaje haya sido tan recalcable, por lo que quería resaltar que se agradece enormemente ver a un profesor tan implicado y dispuesto a resolver todas nuestras dudas. Gracias.

15. Declaración de uso de IA generativa

A continuación, se detalla el uso de IA generativa a lo largo de los distintos entregables.

15.1. Entregable 1

Para el primer entregable, hemos usado Gemini 2.5. para generar el logotipo de la empresa *StraightUp*.