

*GLUCOGUARD*

Aplicación de monitoreo y apoyo para personas con diabetes

Integrantes: Mariano Leal

Steffi Ryser

Índice

[**1. Introducción 3**](#_heading=h.1zycasi6cbu6)

[**2. Datos del documento 3**](#_heading=h.9oa8eg1d0d4y)

[**3. Información del Proyecto 3**](#_heading=h.osxdxk55u46r)

[3.1 Objetivo general 3](#_heading=h.9k4igvpk14he)

[3.2 Objetivos específicos 4](#_heading=h.f38ygdqcpdy1)

[**4. Visión del Proyecto Scrum 4**](#_heading=h.4sf0k8c3fb1e)

[**5. (Para la entrega del primer avance debe tener los siguientes item) 4**](#_heading=h.u6uafggmkeyd)

# 

# Introducción

La diabetes y las enfermedades relacionadas con el nivel de glucosa representan un desafío constante para quienes las padecen. La falta de herramientas prácticas para registrar, monitorear y analizar los datos hace que muchas personas tengan un control deficiente sobre su salud.

GlucoGuard nace como una solución tecnológica que permite registrar niveles de glucosa, recibir alertas críticas, visualizar el historial de datos y contar con apoyo social y educativo. El proyecto se desarrolla bajo la metodología ágil Extreme Programming (XP), priorizando entregas rápidas, calidad del código y mejora continua en cada iteración.

# Datos del documento

| Nombre del documento | GlucoGuard – Documento General del Pro |
| --- | --- |
| Versión | 1.0 |
| Fecha de creación | 10/09/2025 |
| Última actualización | 24/09/2025 |
| Autores | Mariano Leal - Frontend developer  Steffi Ryser - Backend developer |
| Estado del documento | En desarrollo |

# Información del Proyecto

GlucoGuardes una aplicación móvil diseñada para facilitar el monitoreo y control de los niveles de glucosa en personas con diabetes o condiciones relacionadas. La plataforma permite registrar y consultar mediciones, recibir notificaciones ante valores críticos y acceder a material educativo confiable. Además, incorpora un rol de acompañamiento que promueve la participación activa de familiares o amigos en el proceso de control, fomentando el apoyo y la adherencia al tratamiento.

## 3.1 Objetivo general

Desarrollar una aplicación accesible, segura y confiable para el control de la glucosa, aplicando las prácticas de **Extreme Programming (XP)** con el propósito de garantizar la calidad del código, mantener flexibilidad ante los cambios y realizar entregas iterativas que aporten valor continuo al usuario.

## 3.2 Objetivos específicos

* Desarrollar un sistema de registro de glucosa sencillo y confiable.
* Implementar alertas inmediatas ante situaciones críticas.
* Habilitar la participación de un rol de acompañamiento.
* Diseñar un sistema de visualización e historial gráfico de los registros.
* Incluir contenido educativo y guías prácticas de salud.
* Garantizar seguridad en el acceso de usuarios y gestión de datos.

# Propósito del plan de proyecto

El propósito de este plan es definir la estructura, lineamientos y etapas de desarrollo del proyecto GlucoGuard, asegurando que todas las actividades, entregables y decisiones se realicen conforme a la metodología ágil Extreme Programming (XP).

Este documento tiene como objetivo organizar el trabajo del equipo de desarrollo, establecer las prácticas técnicas y los compromisos de iteración, y garantizar que cada entrega aporte valor real e incremental al usuario final.

Además, el plan busca alinear los esfuerzos del equipo con los objetivos del proyecto, definir los roles, riesgos, costos y criterios de calidad, y servir como guía de control y seguimiento para la PMO y los tutores del Capstone.

En síntesis, este plan de proyecto permitirá:

1.- Mantener una visión clara y compartida del producto.

2.- Facilitar la planificación iterativa y la mejora continua.

3.- Asegurar la calidad del código y la funcionalidad del sistema.

4.- Reducir riesgos y optimizar los tiempos de desarrollo.

# Visión del proyecto ágil

Crear una aplicación móvil accesible y confiable que permita a las personas con diabetes o prediabetes registrar y monitorear sus niveles de glucosa, recibir alertas automáticas ante valores críticos y acceder a información educativa, fomentando un acompañamiento activo de familiares o cuidadores para mejorar la calidad de vida y la prevención en salud

| **Grupo Objetivo** | **Necesidades** | **Producto / Servicio** | **Valor** |
| --- | --- | --- | --- |
| Personas con diabetes, prediabetes y familiares de apoyo. | Controlar y registrar fácilmente los niveles de glucosa.  Recibir alertas críticas a tiempo.  Acceder a consejos y educación en salud. | Aplicación móvil GlucoGuard, desarrollada con React Native y Node.js, que integra registro de glucosa, gráficos, alertas automáticas y módulo educativo. | Mejora la gestión personal de la salud.  Promueve la prevención y autocuidado.  Entrega datos claros, alertas oportunas y acompañamiento humano. |

Visión y roles detallados:

<https://docs.google.com/document/d/1HGSes-TShXC761HE_ph4XR_aHtLEK92I/edit?usp=sharing&ouid=103471268962838002645&rtpof=true&sd=true>

# Alcance del proyecto

### **Incluye**

| Desarrollo de la aplicación móvil GlucoGuard para Android/iOS usando React Native. | Alertas automáticas cuando los valores de glucosa se encuentran fuera del rango normal. |
| --- | --- |
| Implementación del registro y autenticación de usuarios. | Rol de acompañante, que permite a familiares o amigos visualizar los valores y alertas del usuario principal. |
| Módulo de ingreso y monitoreo de niveles de glucosa (manual o mediante sensor). | Sección educativa con consejos sobre nutrición, hábitos saludables y autocuidado. |
| Integración simulada o parcial con un dispositivo de monitoreo continuo (CGM), como FreeStyle Libre o Dexcom, mediante lectura de API o datos simulados. | Almacenamiento seguro de la información en una base de datos MySQL, comunicada con una API REST Node.js. |
| Visualización de los niveles de glucosa en gráficos dinámicos y registro histórico. | Diseño responsivo, interfaz intuitiva y accesible, priorizando la usabilidad. |

### **No incluye**

| Integración completa con sensores médicos certificados (solo simulación de lectura). | Panel web para profesionales de la salud. |
| --- | --- |
| Funcionalidades de diagnóstico médico o prescripción | Funciones de mensajería o chat entre usuarios. |

# Metodología de desarrollo

El desarrollo del proyecto GlucoGuard se llevará a cabo utilizando la metodología ágil Extreme Programming (XP), seleccionada por su enfoque en la entrega continua de valor, la calidad del código y la adaptabilidad a los cambios.

Esta metodología prioriza la retroalimentación constante, la comunicación efectiva entre los integrantes del equipo y la mejora continua del producto en cada iteración.

XP promueve la simplicidad en el diseño, el desarrollo incremental y la corrección temprana de errores, reduciendo el riesgo y aumentando la satisfacción del usuario final.

El equipo de desarrollo aplicará las principales prácticas de XP de la siguiente manera:

| **Práctica XP** | **Aplicación en GlucoGuard** |
| --- | --- |
| **Iteraciones cortas** | Se trabajará en ciclos de 2 semanas, entregando incrementos funcionales verificables. |
| **Desarrollo dirigido por pruebas (TDD)** | Se crearán pruebas unitarias antes del código definitivo para asegurar calidad y prevenir regresiones. |
| **Programación en pareja (Pair Programming)** | Los desarrolladores (Mariano Leal y Steffi Ryser) revisarán y optimizarán el código de forma colaborativa. |
| **Integración continua** | Se realizarán commits frecuentes en GitHub y pruebas automáticas de integración entre frontend y backend. |
| **Refactorización constante** | Se mejorará el código de forma continua para mantenerlo limpio, legible y eficiente. |
| **Propiedad colectiva del código** | Todo el equipo podrá modificar y mejorar cualquier parte del código, fomentando responsabilidad compartida. |
| **Cliente disponible / feedback continuo** | La validación del avance se realizará con el tutor o usuario representativo al final de cada iteración. |
| **Entregas frecuentes** | Cada iteración entregará una versión funcional (build estable) del sistema GlucoGuard. |

1. **Definición de Roles y Responsabilidades**

En el proyecto GlucoGuard, el equipo de desarrollo está compuesto por dos integrantes con funciones complementarias.

Cada rol tiene responsabilidades específicas orientadas a garantizar la calidad del producto, la integración entre componentes y la correcta implementación de la metodología ágil XP.

| Rol | Nombre | Responsabilidades principales |
| --- | --- | --- |
| Frontend Developer | Mariano Leal | - Diseñar y desarrollar la interfaz de usuario de la aplicación móvil usando React Native y Expo.  - Implementar componentes visuales intuitivos, responsivos y accesibles.  - Integrar las vistas con el backend mediante API REST.  - Gestionar estados, validaciones y flujos de navegación.  - Participar en pruebas de aceptación y validación de usabilidad. |
| Backend Developer | Steffi Ryser | - Diseñar y desarrollar la lógica del servidor en Node.js y Express.  - Crear y documentar endpoints de la API REST.  - Diseñar la estructura de la base de datos MySQL y asegurar su integridad.  - Implementar autenticación, validaciones y seguridad de datos.  - Realizar pruebas unitarias y de integración.  - Gestionar despliegue y mantenimiento de servicios backend. |

# Costos

El proyecto GlucoGuard se desarrollará como parte de un entorno académico, pero incluye una estimación de costos reales asociados al desarrollo, infraestructura y prototipado con un **sensor de monitoreo continuo de glucosa (CGM)**, similar a los dispositivos **FreeStyle Libre** o **Dexcom**

| **Categoría** | **Detalle** | **Costo estimado (CLP)** | **Observación** |
| --- | --- | --- | --- |
| **Desarrollo (recursos humanos)** | Trabajo de dos programadores (Mariano Leal y Steffi Ryser) | $18.000/hr | Se estiman 9 sprints de una semana cada 1 |
| **Dispositivo de monitoreo continuo (CGM)** | Sensor tipo **FreeStyle Libre 2** o **Dexcom G6**, compatible con app móvil | **$130.000 – $180.000** | Incluye sensor + lector o parche con Bluetooth |
| **Backend / API y base de datos** | Servidor Node.js + MySQL (Render / Railway / Vercel) | $30.000 | Hosting durante la duración del proyecto |
| **Frontend móvil** | React Native / Expo (gratuito) | $0 | Framework open-source |
| **Diseño y prototipado (Figma, Canva, etc.)** | Licencias y herramientas de diseño | $10.000 | Licencia Canva Pro (mensual estimada) |
| **Pruebas y simulación del sensor** | Datos simulados o conexión API dummy | $10.000 | Costo de setup y validación con software auxiliar |
| **Material educativo y documentación** | Manuales, plan de pruebas, capacitación | $5.000 | Impresiones o formato digital |
| **Despliegue y mantenimiento** | Actualizaciones menores y control de versiones | $10.000 | GitHub + integraciones CI/CD |
| **Terra API sensor** | Api para conectar a sensor | $300.000 | Para testear con cuentas de sensor de prueba |
| **Total estimado del proyecto** |  | **≈ $195.000 – $245.000 CLP** | Dependiendo del sensor adquirido |

**Detalle técnico del dispositivo**

| **Dispositivo estimado** | **Marca / Modelo** | **Costo promedio (CLP)** | **Duración / Observación** |
| --- | --- | --- | --- |
| Sensor **FreeStyle Libre 2** | Abbott | $130.000 | Cada sensor dura 14 días, permite conexión Bluetooth con apps móviles. |
| Sensor **Dexcom G6** | Dexcom | $180.000 | Mayor precisión y duración (hasta 10 días por sensor). Compatible con Android/iOS. |

### **Conclusión de costos**

El costo principal del proyecto corresponde al trabajo de los dos programadores durante los 9 sprints, estimado en **$13.680.000 CLP**, constituyendo la mayor parte del presupuesto.

El **dispositivo de monitoreo continuo (CGM)** sigue siendo un gasto relevante, con un valor entre **$130.000 y $180.000 CLP**, mientras que los demás costos se mantienen bajos gracias al uso de **herramientas y tecnologías open source**.

En total, el proyecto alcanza un costo aproximado de **$14 millones de pesos chilenos**.

# Resumen de riesgos

El proyecto presenta riesgos técnicos y operativos relacionados con la conexión del sensor CGM, la pérdida de datos y la compatibilidad de versiones.  
 Se aplican estrategias de mitigación como uso de APIs simuladas, respaldos automáticos y planificación iterativa para asegurar la continuidad y estabilidad del sistema.  
  
Se anexa excel;

<https://docs.google.com/spreadsheets/d/1DOGWMOZDJJnx93P9Oy1BWL-xi1L25IVh/edit?usp=sharing&ouid=103471268962838002645&rtpof=true&sd=true>

# Épicas para el proyecto

| **Épica** | **Descripción** | **Prioridad** |
| --- | --- | --- |
| EP1 – Monitoreo sin sensor | Registro manual de niveles de glucosa y visualización de gráficos históricos. | Must |
| EP2 – Usuarios y autenticación | Registro, inicio de sesión, roles y recuperación de contraseña. | Must |
| EP3 – Monitoreo con sensor | Integración con sensores para obtener datos en tiempo real. | Should |
| EP4 – Contacto de apoyo | Contacto con profesionales y alertas de apoyo. | Should |
| EP5 – Soporte y educación | Contenido educativo y guía sobre hábitos saludables. | Could |
| EP6 – Gestión de riesgos y costos | Alertas de riesgo y gestión de costos de insumos. | Must |
| EP7 – Capacitación y documentación | Tutoriales, manuales y documentación técnica del sistema. | Could |

# Priorización de epicas

Para organizar el desarrollo de GlucoGuard, se utilizó la técnica **MoSCoW**, que clasifica las funcionalidades según su importancia:

* **Must (Debe tener):** Funcionalidades críticas que **son esenciales para el lanzamiento** del producto. Sin estas, la aplicación no cumple su objetivo mínimo. Por ejemplo, en GlucoGuard, el registro manual de glucosa (EP1) y la gestión de usuarios (EP2) son Must.
* **Should (Debería tener):** Funcionalidades importantes que **mejoran la experiencia del usuario**, pero que no son imprescindibles para la primera versión. Por ejemplo, la integración con sensores (EP3) o el contacto de apoyo (EP4).
* **Could (Podría tener):** Funcionalidades opcionales o complementarias que **aportan valor adicional**, pero que pueden implementarse en fases posteriores. Por ejemplo, contenido educativo (EP5) o manuales y tutoriales (EP7).

Esta priorización permite **enfocar al equipo en lo crítico primero**, dejando las mejoras para sprints posteriores.

# Definición de historias de usuario

Cada épica se descompone en **historias de usuario (HU)** que describen funcionalidades específicas:

**EP1 – Monitoreo sin sensor**

* HU01 – Ingreso manual de valores de glucosa
* HU02 – Visualización de gráfico inicial
* HU03 – Configuración de umbrales de glucosa
* HU04 – Coloreo de valores manuales según rango
* HU05 – Alertas según valores de rangos

**EP2 – Usuarios y autenticación**

* HU06 – Notificación a contacto de apoyo según valores críticos
* HU07 – Registro de usuario
* HU08 – Registro con sensor vinculado
* HU09 – Ingreso de datos personales
* HU10 – Guardado de datos y fecha en base de datos
* HU11 – Inicio de sesión

**EP3 – Monitoreo con sensor**

* HU12 – Lectura automática desde sensor
* HU13 – Visualización gráfica con colores según rango
* HU14 – Notificación a contacto de apoyo en valores críticos
* HU15 – Detección de fallas del sensor

**EP4 – Contacto de apoyo**

* HU16 – Creación visualización de contacto de apoyo
* HU17 – Recibir notificaciones de alertas

**EP5 – Soporte y educación**

* HU18 – Acceso a guía educativa
* HU19 – Aprendizaje de términos de diabetes en guía educativa

**EP6 – Gestión de riesgos y costos**

* HU20 – Registro de riesgos del proyecto
* HU21 – Flujo de caja del proyecto
* HU22 – Reporte de costos por épica

**EP7 – Capacitación y documentación**

* HU23 – Plan de capacitación de usuarios
* HU24 – Manual de usuario digital
* HU25 – Documentación técnica del sistema

El archivo contiene las **historias de usuario del proyecto GlucoGuard**, organizadas según prioridad, épica y tareas vinculadas, conforme a la metodología ágil XP.

<https://docs.google.com/spreadsheets/d/1cDMhhoVt_jGXv1M4r3XQA43fdCqGbICl/edit?usp=sharing&ouid=103471268962838002645&rtpof=true&sd=true>

# Product backlog del proyecto

El Product Backlog de GlucoGuard reúne todas las historias de usuario priorizadas, definidas según esfuerzo, iteración y valor de negocio, alineadas a la metodología ágil XP.

Este documento da inicio a la planificación del proyecto y se encuentra anexado en el siguiente enlace.

<https://docs.google.com/spreadsheets/d/1PeNTeyc3BR6u4mtm7vHgEYvqLDBlYiAb/edit?usp=sharing&ouid=103471268962838002645&rtpof=true&sd=true>

Se adjunta el archivo en formato Excel con el **Sprint Backlog del proyecto GlucoGuard**, que detalla las historias de usuario, tareas vinculadas y responsables según la metodología ágil XP.

<https://docs.google.com/spreadsheets/d/1FDAB0whQ1brLF3NuwFJvZ0F4sBc4PcDN/edit?usp=sharing&ouid=103471268962838002645&rtpof=true&sd=true>

# Estimación de puntos de historias

Para medir la **complejidad y esfuerzo** de cada historia de usuario (HU), se utilizó la **escala Fibonacci (1, 2, 3, 5, 8)**.

La escala Fibonacci es una **serie de números donde cada número es la suma de los dos anteriores**: 1, 2, 3, 5, 8, 13… Se utiliza en Scrum para estimar tareas porque ayuda a reflejar **incrementos no lineales de complejidad**, es decir, a medida que una historia es más compleja, el esfuerzo crece de forma más que proporcional.

* **Historias fáciles (1–3 puntos):** HU simples, sin dependencias ni integración compleja. Ej.: acceso a guías educativas, registro básico de datos.
* **Historias medianas (5 puntos):** HU con cierta complejidad o integración parcial. Ej.: registro de usuario, alertas básicas.
* **Historias complejas (8 puntos):** HU que implican múltiples dependencias, interacción con sensores o alertas críticas. Ej.: configuración de umbrales de glucosa, visualización avanzada de gráficos.

Esta estimación ayuda a **planificar sprints balanceados**, asignar tareas según capacidad del equipo y asegurar entregas de valor incremental.

Se adjunta el archivo en formato Excel con el Product Backlog completo del proyecto GlucoGuard.

<https://docs.google.com/spreadsheets/d/18MXHrWP3S8M8XwotOkwUUKScWQpVcQpG/edit?usp=sharing&ouid=103471268962838002645&rtpof=true&sd=true>

# Sprint planning

# Definición y estimación de tareas

# 