

**UNIVERSIDAD DEL VALLE DE GUATEMALA**

*1CC30901020251 - Ingeniería en Software 1*

Sección 10

Ing. Lynette García Pérez



*Excelencia que trasciende*

**DEL VALLE**  
GRUPO EDUCATIVO

## Raíces de vida

CHRISTA ISABELLA OBANDO GUZMAN #23074

ANTHONY LOU SCHWANK #23410

ROBERTO JOSE BARREDA SIEKAVIZZA #23354

ANGGELIE LIZETH VELASQUEZ ASECIO # 221181

MIA ALEJANDRA FUENTES MERIDA #23775

**GUATEMALA, 24 de marzo de 2025**

## **Resumen**

La desnutrición crónica es un problema que afecta a miles de personas, especialmente a niños que habitan en zonas rurales; en Guatemala casi la mitad de los menores de edad sufren de esta enfermedad. Por lo que este proyecto tiene como objetivo reducir la desnutrición crónica en Guatemala mejorando la comunicación entre comunidades vulnerables, ONGs y servicios de salud.

Debido al impacto mundial que ha tenido la tecnología, en esta fase se trabajó en los prototipos, funcionabilidad de los mismos y elección de la tecnología utilizar sobre una plataforma móvil que permita facilitar el monitoreo, identificación y casos críticos de desnutrición. Asimismo, cumpliendo con las necesidades vistas en los usuarios mediante Historia de usuarios, descripción de actores y entrevistas, se desarrollaron interfaces centradas en el usuario para garantizar accesibilidad y eficiencia en la recolección y transmisión de datos.

## **Introducción**

Este proyecto será de ayuda para luchar contra la desnutrición crónica en Guatemala, mejorando la comunicación entre las comunidades afectadas y las organizaciones no gubernamentales. Guatemala es uno de los países con mayor desnutrición crónica en Latinoamérica, siendo los menores de edad los más afectados. (*Guatemala: El País de América Latina Con Más Desnutrición Crónica / Acción Contra el Hambre*, s. f.)

El proyecto está orientado a desarrollar una plataforma móvil que facilite la comunicación y colaboración entre las ONGs, voluntarios y líderes comunitarios, así facilitar la detección de casos críticos de desnutrición. Por lo tanto, se busca optimizar el tiempo de respuesta, mejorar la recolección y transmisión de datos de los casos críticos de desnutrición. Dicha plataforma móvil tomara en cuenta accesibilidad, eficiencia y facilidad de uso según las necesidades de los usuarios clave (ONGs, líderes comunitarios y voluntarios).

El objetivo general es reducir la desnutrición crónica en Guatemala mejorando la comunicación entre comunidades vulnerables, ONGs y servicios de salud. Para lograrlo, se plantean los siguientes objetivos específicos, desarrollar una página web o aplicación móvil que permita el reporte y seguimiento de casos críticos; y establecer una interacción entre los usuarios clave del proyecto (Organizaciones no gubernamentales, líderes comunitarios y voluntarios).

# Design Thinking

Imagen 1. Primer prototipo diseñado.

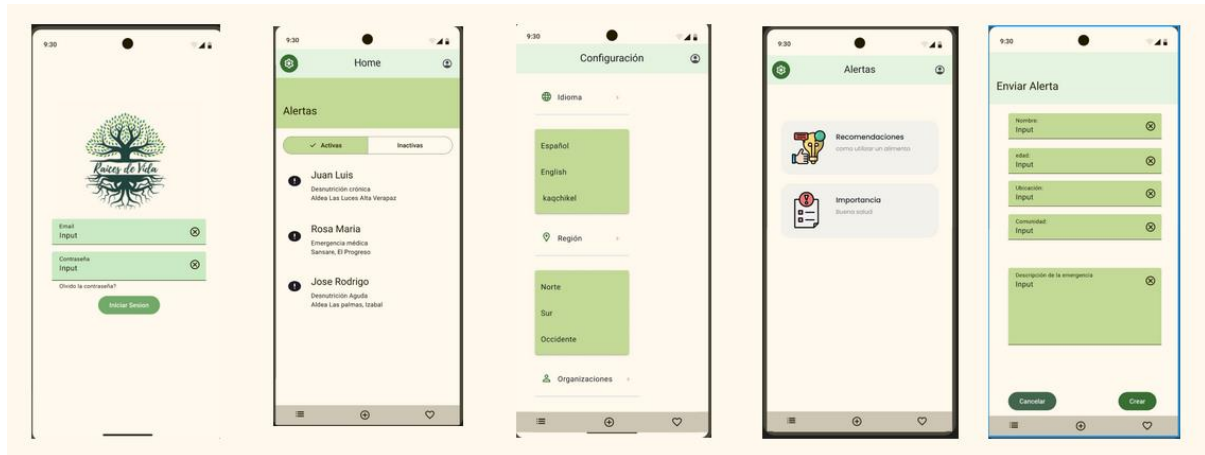
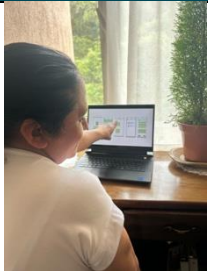








Imagen 2. Segundo prototipo diseñado.



Tabla 1. Evidencias y comentarios de futuros usuarios ante prototipos.

Nombre	Prototipo	Comentarios	Foto/Video
Lorena Pérez	v1	“No leo las letras, están pequeñas.”; “¿Qué hago?” (hace notar que es poco intuitiva); “¿Eso es un botón?” (falta de diferenciación de botones)	
Lorena Pérez	v2	“No leo” (letras muy pequeñas todavía); “Ahí no se mira nada, nene” (cuando le enseñé el mapa)	
Joaquín Ramírez	v1	“Les falta imágenes y orden”	
Joaquín Ramírez	v2	“Se ve mucho mejor”	
Candy Sandoval	v1	“Ingresar más idiomas”. “Cambiar el color porque, no da a entender que quiere aportar nuestra app”	
Candy Sandoval	v2	“Que exista una sección donde se pueda seleccionar la gravedad de los casos de una manera más efectiva”, “Que para facilitar el acceso de los usuarios de las comunidades se utilice DPI”	
Juan Pablo Aguirre	v1	“Se ve interactivo”	


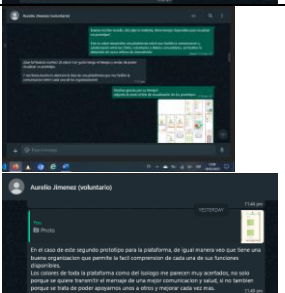

Juan Pablo Aguirre	v2	“Tiene un buen uso para nuestro día a día”	
Aurelio Jiménez	v1	“Buena organizacion que permite la facil comprension de cada una de sus funciones disponibles. Los colores de toda la plataforma como del isologo me parecen muy acertados, no solo porque se quiere transmitir el mensaje de una mejor comunicacion y salud, si no tambien porque se trata de poder apoyarnos unos a otros y mejorar cada vez mas.”	
Aurelio Jiménez	v2	“Me parece algo muy estetico y organizado!” “Se puede entender con facilidad la funcionalidad de cada ícono que esta en la barra inferior”	

Tabla 2. Comentarios y su relación con requisitos existentes

Comentario	Requisito afectado	Acción propuesta a incluir
V1 - “No leo las letras, están pequeñas.”	Usabilidad y Accesibilidad	Aumentar tamaño de fuente y usar una mejor tipografía para una mejor legibilidad
V1 - “¿Qué hago?”	Interfaz Externa	Rediseñar flujos con iconos y un tutorial interactivo.
V1 - “¿Eso es un botón?”	Interfaz Externa	Añadir sombras, colores contrastantes (ej: verde para acciones, rojo para cancelar) y efectos hover.
V2 - “No leo”	Usabilidad	Implementar opción de zoom en textos y voz en off

		para usuarios con baja alfabetización
V2 - “Ahí no se mira nada, nene”	Rendimiento	Optimizar mapa: reducir resolución de imágenes, usar capas vectoriales y añadir leyendas con iconos.
V1 - 'Les falta imágenes y orden'	Diseño y Organización	Añadir más imágenes representativas y mejorar la estructura visual del contenido.
V2 - 'Se ve mucho mejor'	Usabilidad y Estética	Mantener la nueva organización y optimizar el diseño visual para mayor claridad.
V1 - 'Ingresar más idiomas'.	Accesibilidad	Agregar soporte para múltiples idiomas en la configuración de la app.
V1 - 'Cambiar el color porque no da a entender que quiere aportar nuestra app'	Diseño de interfaz	Modificar la paleta de colores para mejorar la percepción del propósito de la app.
V2 - 'Que exista una sección donde se pueda seleccionar la gravedad de los casos de una manera más efectiva'	Usabilidad	Añadir una sección que permita clasificar los casos según su gravedad de forma clara y sencilla.
V2 - 'Que para facilitar el acceso de los usuarios...'	Usabilidad y navegación	Optimizar la estructura de navegación para que los usuarios accedan más fácilmente a las funciones clave.
V1 - “Se ve interactivo”	Diseño y Organización.	Persistir con el diseño del prototipo.
V2 - “Tiene un buen uso para nuestro día a día”	Usabilidad	Persistir con la usabilidad del diseño, mejorando el impacto de los requisitos funcionales y los no funcionales.
V1 - “Buena organizacion que permite la facil comprension de cada una de sus funciones disponibles...”	Diseño y Organización.	El prototipo debe tener una organización fácil de comprender, mantener la idea del mismo.
V2 - “Se puede entender con facilidad la funcionalidad de cada ícono que esta en la barra inferior”	Usabilidad	Barra home debe ser intuitiva y fácil de acceder de la misma.

## Impacto en Requisitos Funcionales y No Funcionales

- Interfaz intuitiva:
  - Especificar tamaño mínimo de fuente y uso de iconos. Esto para garantizar accesibilidad para usuarios con baja alfabetización.
- Mapa interactivo:
  - Incluir capas simplificadas y leyendas visuales. Con el fin de, mejorar comprensión del mapa en zonas rurales con conexión limitada.
- Rendimiento:
  - Cargar el mapa en  $\leq 2$  segundos usando caché de ubicaciones.

## Requisitos no funcionales

Tabla 3. Presentación de requisitos no funcionales.

Requisito no funcional	Categoría	Forma en que se medirá su cumplimiento.
La carga de los datos y la visualización de la información debe completarse en al menos 3 segundos (principal) en condiciones de una conexión estándar.	Rendimiento	Pruebas de tiempo de respuesta con herramientas de monitoreo del rendimiento.
Internacionalización: El contenido está en distintos idiomas (inglés, español, lengua Maya).	Soporte	La información está disponible en varios idiomas. Se ha planeado incluir español, inglés y una lengua adicional para facilitar la comunicación con los líderes comunitarios. Esta última aún no ha sido definida.
Los paquetes de información deben ser concisos y completos	Interfaz interna	Tamaño analítico de los datos
Navegación autoexplicativa	Apariencia o interfaz externa	Pruebas de usuario con 5 personas analfabetas para saber si la navegación se entiende a sí misma. Pruebas de UX



Información en grande con uso de ilustraciones, colores y cuadros de ingreso de datos accesibles y obvios	Apariencia o interfaz externa	Pruebas de usuario con 5 niños para saber si entienden el funcionamiento de la plataforma. Pruebas de UX
Solo los líderes comunitarios y ONGs pueden modificar o agregar información sobre los casos reportados.	Seguridad	Pruebas de acceso y autenticación de usuario con roles de usuario.
La interfaz debe ser intuitiva para usuarios con conocimiento básico en tecnología.	Usabilidad	Prueba de usuario con al menos 5 participantes de los usuarios. Pruebas de UX
El sistema debe estar disponible al menos el 99.90% del tiempo, sin tomar en cuentas las interrupciones prolongadas (actualizaciones).	Disponibilidad	Monitoreo del up-time con herramientas como Pingdom.
El sistema debe permitir actualizaciones y mantenimiento sin afectar la disponibilidad por más de 2 horas.	Mantenibilidad	Registro de tiempo en la inactividad durante las actualizaciones.
El sistema debe contar con respaldo automático de datos cada 24 horas.	Confiabilidad	Se validará revisando registros de respaldo y pruebas de restauración de datos para periodos de 24 horas.
Los usuarios deben recibir confirmación visual y/o sonora de sus acciones dentro de la aplicación	Usabilidad	Se validará con 5 pruebas de usuario si los mensajes de confirmación y feedback visual son claros y comprensibles. Pruebas de UX
Un sistema que no requiere más de 3 GB de ram	Hardware	Calidad en tiempo a relación de capacidad computacional de la

		máquina
Instrucciones detalladas en línea de cómo funciona la plataforma	Ayudas y documentación en línea	Pruebas con 5 usuarios para verificar si entienden el funcionamiento de la plataforma. Pruebas de UX
Funcionamiento offline (caché)	Accesibilidad	Pruebas sin conexión: Usuarios pueden guardar casos y sincronizar al recuperar internet (100% de los casos guardados)

## Cambios justificados

- Métricas cuantificables:
  - Por ejemplo, que en el desarrollo se pueda cumplir con expectativas como: "90% logra completar flujos clave" en lugar de "pruebas con 5 usuarios".
  - Alineación con criterios del tercer corte: Requisitos medibles en fase de pruebas.
- Nuevo requisito: Funcionamiento offline:
  - Base: Necesidad identificada en comunidades rurales (Bitácora 25/02/2025).
  - Tecnología asociada: React Native + AsyncStorage para caché.

Tabla 4. Tecnologías por trabajar con sus requisitos asociados.

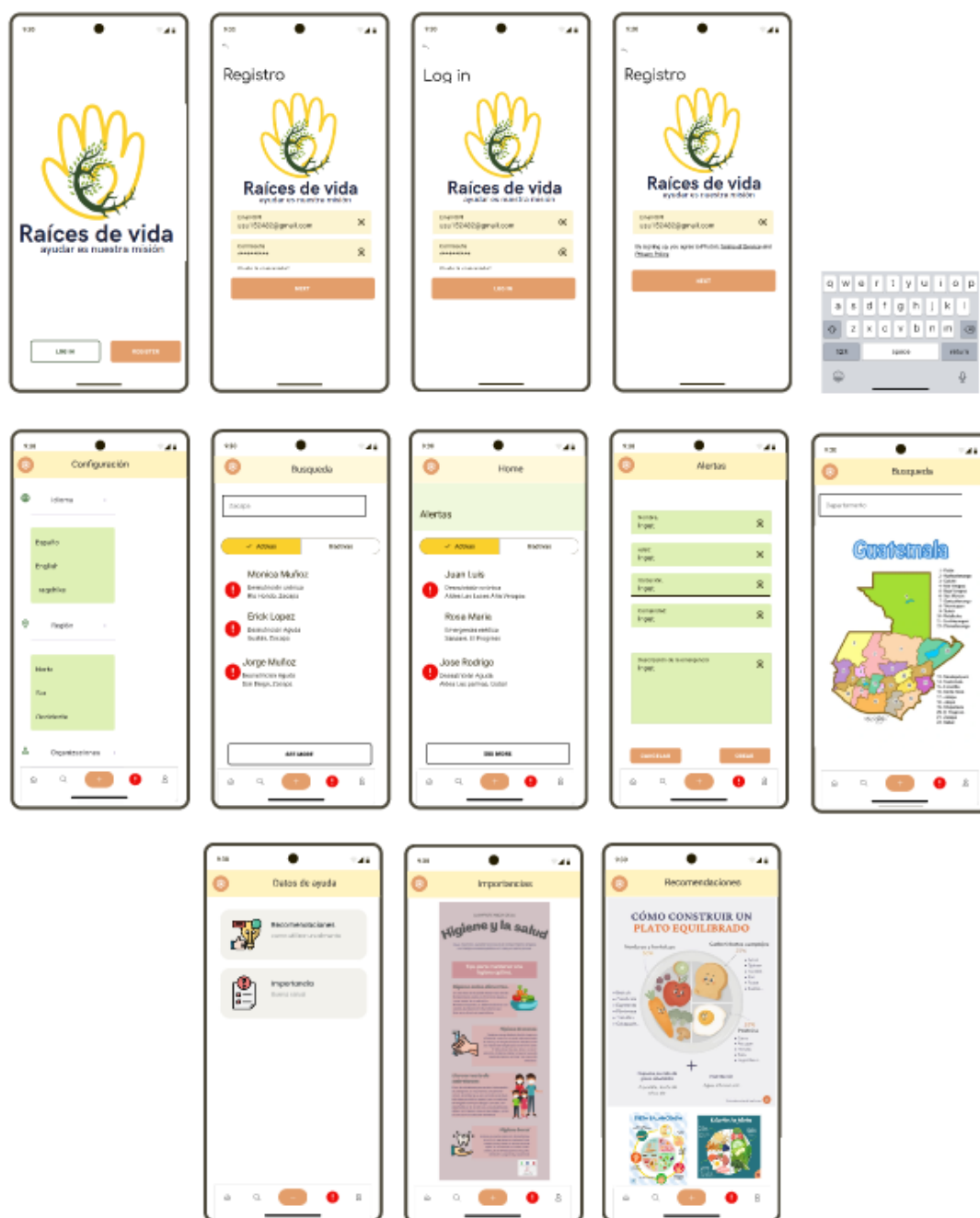
Tecnología	Requisitos Asociados	Justificación
<b>React Native</b>	Funcionamiento sin estar en línea, interfaz intuitiva	AsyncStorage para caché y componentes nativos para rendimiento
<b>PostgreSQL</b>	Disponibilidad $\geq 99.90\%$ ; Respallos automáticos	Replicación en AWS y pgBackRest para respaldos
<b>Node.js</b>	Rendimiento ( $\leq 3$ segundos), Consumo de RAM $\leq 3$ GB	Arquitectura asíncrona y optimización de consultas con Redis
<b>Traducciones</b>	Soporte multilingüe	Uso de bibliotecas como i18next para gestión de idiomas y contenido adaptable

## Prototipos mejorados

Imagen 3. Primer prototipo mejorado.



Imagen 4. Segundo prototipo mejorado.



## Análisis

1. Elabore una lista de requisitos funcionales de su sistema.
  - a. Requisitos funcionales:
    - i. Inicio de sesión:
      1. El sistema debe permitir a los líderes comunitarios registrarse proporcionando su información personal y la de su comunidad.

2. El sistema debe permitir a los voluntarios registrarse con su información de contacto.
  3. El sistema debe permitir a las ONGs registrarse proporcionando datos institucionales y de contacto.
  4. El sistema debe permitir el inicio de sesión de los usuarios con correo electrónico y contraseña.
- ii. Gestión de los casos de desnutrición:
1. Los líderes comunitarios deben poder registrar un caso de desnutrición proporcionando datos como nombre, edad, ubicación y nivel de gravedad.
  2. El sistema debe permitir a los líderes comunitarios adjuntar comentarios adicionales sobre el caso reportado.
  3. El sistema debe permitir a los líderes comunitarios/ONGs actualizar el estado de un caso (Ejemplo: "Pendiente", "En proceso", "Atendido").
  4. El sistema debe permitir a las ONGs visualizar los casos reportados, filtrándolos por ubicación y nivel de gravedad.
  5. El sistema debe permitir a los voluntarios visualizar casos en un mapa interactivo según su ubicación.
  6. El sistema debe generar una alerta para las ONGs y voluntarios cuando se registre un nuevo caso crítico.
- iii. Coordinación de ayuda:
1. Las ONGs deben poder contactar a los líderes comunitarios dentro de la plataforma.
  2. El sistema debe permitir que los líderes comunitarios y voluntarios confirmen la ayuda en los casos atendidos.
- iv. Recursos y donaciones:
1. El sistema debe permitir que los líderes comunitarios realicen solicitudes de recursos específicos.
  2. El sistema debe notificar a las ONGs cuando haya solicitudes de recursos pendientes.
- v. Alertas:
1. El sistema debe enviar notificaciones a los usuarios cuando se registre un nuevo caso crítico en su área.
  2. El sistema debe notificar a las ONGs sobre nuevas solicitudes de ayuda.
  3. El sistema debe notificar a los voluntarios sobre oportunidades de participación en campañas cercanas.
- vi. Seguridad y mantenibilidad:
1. El sistema debe restringir el acceso a la información de los casos solo a usuarios registrados.
  2. El sistema debe permitir a los usuarios modificar su perfil y datos personales.

3. Los datos de los afectados por la desnutrición deben ser anónimos para proteger su identidad.
  4. El sistema debe permitir que los líderes comunitarios eliminen casos que hayan sido resueltos.
- vii. Usabilidad y accesibilidad:
1. El sistema debe contar con una interfaz de usuario intuitiva y accesible para personas con bajo nivel de alfabetización digital.
  2. El sistema debe ofrecer la opción de cambiar el idioma entre español, inglés y una lengua maya.
  3. El sistema debe permitir el acceso a la información incluso con conexiones a Internet limitadas, mediante almacenamiento en caché.
- b. Historias de usuario:
- i. Actor: Líder comunitario
    1. Descripción: El líder comunitario ingresa a la aplicación, registra un nuevo caso de desnutrición e ingresa detalles como nombre del afectado, edad, nivel de desnutrición, y ubicación
    2. Flujo principal:
      - a. El líder comunitario inicia sesión en la app/página Web.
      - b. Debe poder cambiar el idioma si desea.
      - c. Selecciona la opción de “Reportar caso”.
      - d. Completa el formulario con la información que se solicita
      - e. Confirma y envía el formulario.
      - f. El sistema almacena la información y la pone a disposición de ONGs y voluntarios.
    3. Excepciones:
      - a. Si falta información obligatoria el sistema alerta al usuario.
      - b. Si hay problemas de conexión el sistema permite guardar el caso temporalmente para enviarlo más tarde.
  - ii. Actor: Voluntario
    1. Descripción: El voluntario ingresa a la aplicación para visualizar los casos de desnutrición reportados y decidir en qué comunidad brindar ayuda.
    2. Flujo principal:
      - a. El voluntario inicia sesión en la app/página web.
      - b. Debe poder cambiar el idioma si desea.
      - c. Selecciona la opción de “Ver casos reportados”.
      - d. Filtra los casos por ubicación o nivel de gravedad.
      - e. Revisa los detalles de un caso específico.
    3. Excepciones:
      - a. Si no hay casos disponibles en la zona seleccionada, el sistema informa al usuario.

- b. Si hay problemas de conexión, la app muestra un mensaje de error y permite reintentar más tarde.
  - iii. Actor: ONG
    1. Descripción: La ING accede a la aplicación para visualizar los casos más críticos y comunicarse con los líderes comunitarios para coordinar la entrega de ayuda.
    2. Flujo principal:
      - a. La ONG inicia sesión en la app/página web.
      - b. Debe poder cambiar el idioma si desea.
      - c. Selecciona la opción “Ver casos reportados”.
      - d. Filtra casos por ubicación y nivel de gravedad.
      - e. Contacta al líder comunitario del caso seleccionado.
      - f. Coordina la entrega de recursos o asistencia médica.
    3. Excepciones:
      - a. Si el contacto con el líder comunitario falla, el sistema permite dejar un mensaje en la aplicación.
      - b. Si no hay disponibilidad inmediata de recursos, la ONG puede registrar una campaña para recolectar apoyo.
  - c. Enlazar cada requisito funcional con una historia de usuario:

Tabla 5. Enlace requisito funcional e historia de usuario.

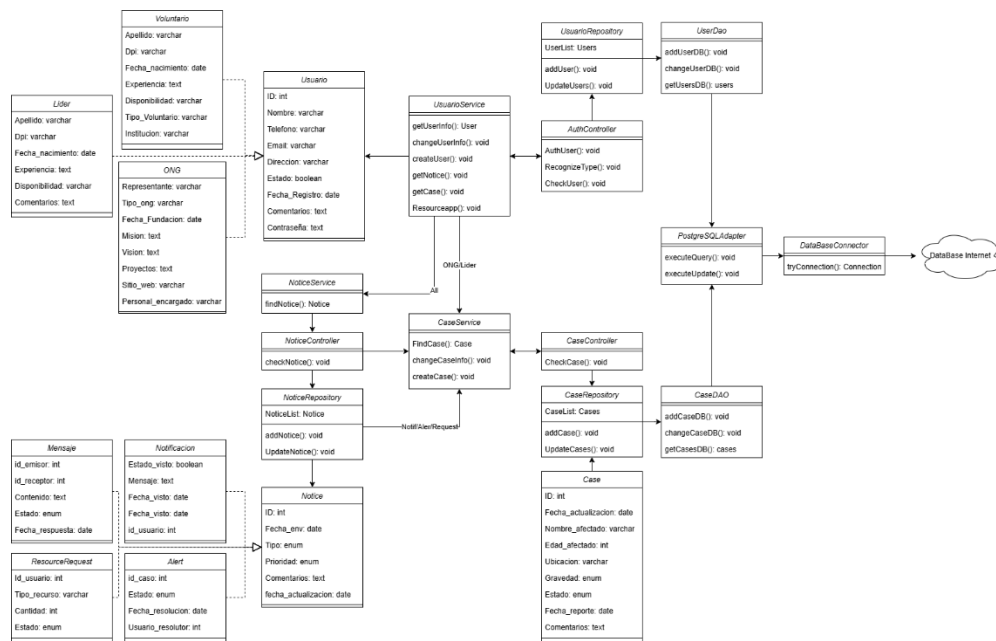
Requisito Funcional	Historia de Usuario	Paso o Relación
Inicio de sesión		
Registro de líderes comunitarios	Líder comunitario	Registro e inicio de sesión
Registro de voluntarios	Voluntario	Registro e inicio de sesión
Registro de ONGs	ONG	Registro e inicio de sesión
Inicio de sesión con correo y contraseña	Todas	Inicio de sesión
Gestión de casos de desnutrición		
Registro de un caso de desnutrición	Líder comunitario	Formulario de reporte
Adjuntar comentarios al caso	Líder comunitario	comentarios
Actualización del estado del caso	Líder comunitario / ONG	Actualizar estado del caso
Visualización y filtrado de casos por ONGs	ONG	Filtrar por ubicación y gravedad
Visualización de casos en mapa por voluntarios	Voluntario	Ver mapa de casos
Alertas para ONGs y voluntarios por casos críticos	ONG / Voluntario	Sistema notifica automáticamente
Coordinación de ayuda		
ONGs pueden contactar líderes comunitarios	ONG	Contacto con líderes
Confirmación de ayuda en casos atendidos	Líder comunitario / Voluntario	Confirmación de ayuda al atender casos

Recursos y donaciones		
Solicitud de recursos por líderes comunitarios	Líder comunitario	Opción de solicitar recursos
Notificación a ONGs de solicitudes de recursos	ONG	Notificación de solicitudes pendientes
Alertas		
Notificación de casos críticos en la zona	ONG / Voluntario	Sistema envía alertas
Notificación de solicitudes de ayuda a ONGs	ONG	Sistema envía alertas
Notificación a voluntarios sobre campañas	Voluntario	Sistema envía alertas
Seguridad y mantenibilidad		
Acceso restringido a información de casos	Todas	Solo usuarios registrados pueden acceder
Modificación de perfil y datos personales	Todas	Opción en la configuración del usuario
Anonimato de datos de los afectados	Líder comunitario	Datos anonimizados en el reporte
Eliminación de casos resueltos	Líder comunitario	Opción para eliminar casos cerrados
Usabilidad y accesibilidad		
Interfaz intuitiva y accesible	Todas	UI diseñada para facilidad de uso
Cambio de idioma	Todas	Idioma establecido por región
Acceso con conexión limitada (caché)	Líder comunitario	Almacenamiento temporal

## 2. Clases Preliminares:



Diagrama 1. Diagrama de Clases.



[Link a Draw.io](https://www.draw.io)

Tabla 6. Descripción de las clases.

Bloque	Clase	Descripción
Usuarios	Lider	Hereda de Usuario, funciona con permisos similares a ONG y tiene acceso a la gran mayoría de cosas
	ONG	Hereda de Usuario, tiene la mayor cantidad de accesos
	Voluntario	Hereda de Usuario, solo puede visualizar datos
	Usuario	Tiene componentes que define a cada tipo de Usuario, y es quien dicta el manejo de sus datos
	UsuarioService	Usa funciones de CaseService, y NoticeService para hacer su relay de informacion. Simultaneamente, usa AuthController para hacer la diferenciacion de acceso a la modificacion de esa informacion.
	AuthController	Chequea que el usuario exista y el tipo de Usuario del que se está manejando, dando sus restricciones acordemente
	UsuarioRepository	Se guardan los usuarios de forma local con este Repositorio, se guardarian en una lista de tipo Usuario para que se guarden todos los que heredan de él.
Noticias	Mensaje	Hereda de notice, su funcion es retener texto de mensaje y sus horarios, ademas de quienes mandan y reciben
	Notificación	Hereda de notice, funciona como una notificacion comun, con la informacion deseada y quien lo mando.

	Alert	Hereda de notice, similar a notificación, pero con un agregado que se recibiría particularmente a Usuarios de mayor jerarquía por la naturaleza de su información.
	ResourceRequest	Hereda de notice, este solo llega a ONG, y solo lideres y otras ONG pueden realizarlo. Funciona para pedir algún tipo de recurso en particular para cierta comunidad.
	Notice	Sus componentes definen las partes iguales de cada una de las formas de comunicación que tomarían parte en la interfaz humana.
	NoticeRepository	Se guardan localmente todas las datas tipo Notice, similarmente a UsuarioRepository.
	NoticeController	Hace un chequeo del formato de los datos tipo Notice, para verificar que su integridad de dato este al estandar del repositorio local.
	NoticeService	Encargado de mandar las noticias y sus datos directamente a CaseService y a UsuarioService, además de recibir de ellos.
Casos	Case	Tiene todos los componentes de información que nos interesaría de un caso de desnutrición.
	CaseRepository	Almacenamiento local de los Casos
	CaseController	Verifica el formato de los datos de los Casos que se reciben, además de sus modificaciones.
	CaseService	Es quien recibe las solicitudes de cambio y quien manda los datos de los Casos hacia UsuarioService, tambien está conectado a NoticeService, lo que ayuda a asociar los pedidos a un caso en particular si así fuera necesario.
Conexión	DataBaseConnector	Su único funcionamiento es hacer un try a la conexión de internet hacia la base de datos. Este devuelve la conexion para poder usarla una vez funcione.
	PostgreSQLAdapter	Recibe la luz verde de DataBaseConnector, su trabajo una vez funciona es realizar los Querys hacia la base de datos SQL.
	UserDAO	Le hace pedidos a PostgreSQL para poder recibir la información de usuarios, además de poder modificarla también. Esto se lo manda al repositorio para almacenarlo localmente.
	CaseDAO	Lo mismo que UserDAO pero para casos

a. Diagrama de paquetes:

Diagrama 2. Diagrama de paquetes.

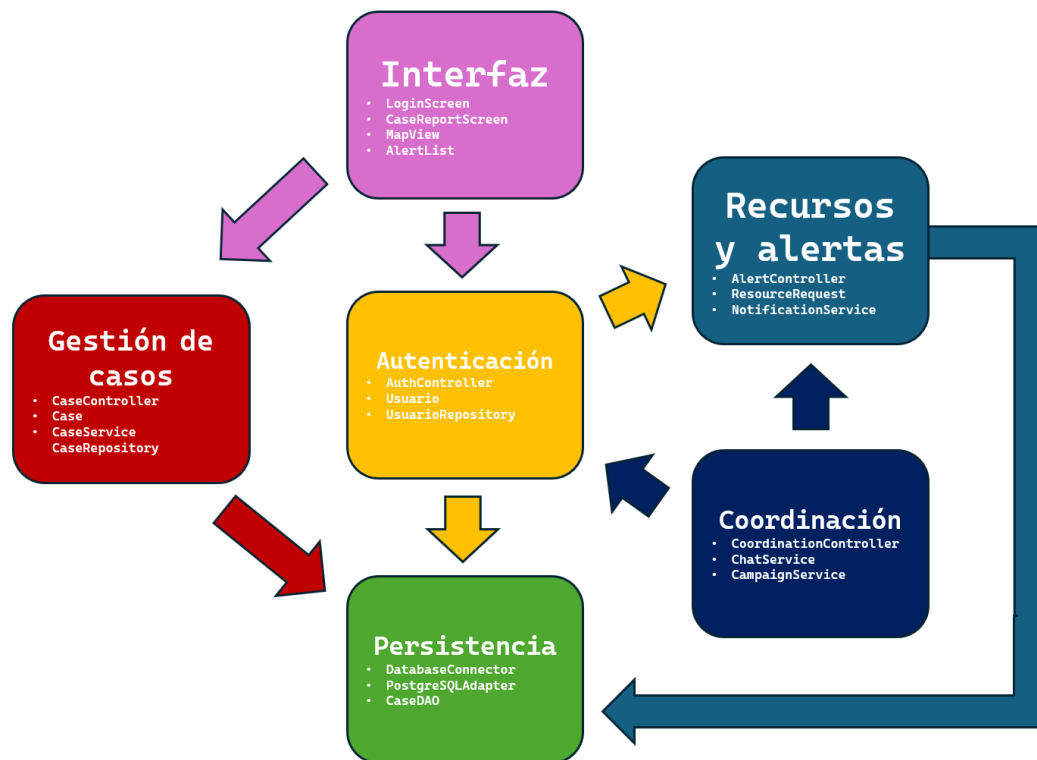


Tabla 7. Descripción de paquetes y componentes

Paquete	Componentes	Descripción
Autenticación	AuthController, Usuario, UsuarioRepository	Mira el registro, inicio de sesión y los 3 roles (líder, voluntario, ONG)
Gestión de Casos	CaseController, Case, CaseService, CaseRepository	Permite reportar, actualizar y filtrar casos
Coordinación	CoordinationController, ChatService, CampaignService	Facilita comunicación interna y organización de campañas
Recursos y Alertas	AlertController, ResourceRequest, NotificationService	Maneja solicitudes de recursos y envía alertas a usuarios
Persistencia	DatabaseConnector, PostgreSQLAdapter, CaseDAO	Conexión con PostgreSQL y operaciones de base de datos
Interfaz	LoginScreen, CaseReportScreen, MapView, AlertList	Componentes visuales para móvil con React Native

### 3. Persistencia de datos:

Diagrama 3. Clases persistentes.

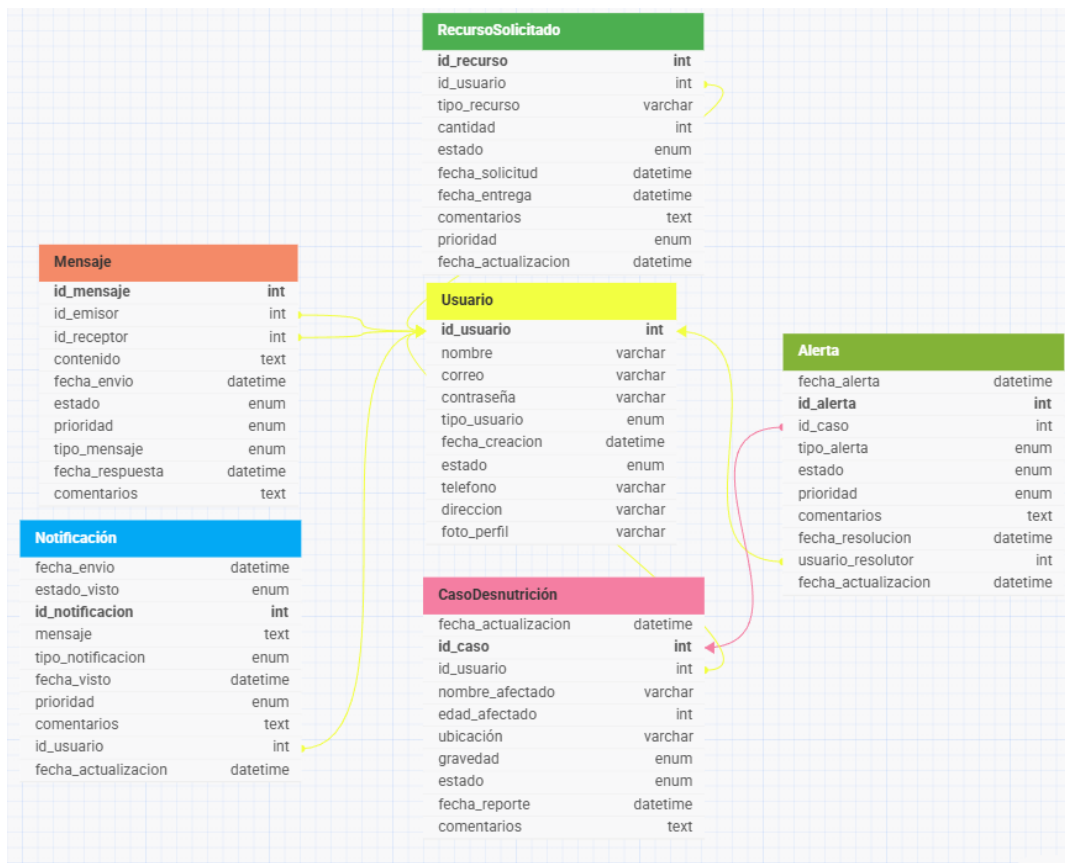


Diagrama 4. Entidad Relación, de su base de datos relacional.



## Diseño

### 1. Lenguaje de Programación: JavaScript

- a. JavaScript es un lenguaje de programación interpretado, orientado a objetos, comúnmente utilizado en el desarrollo web y móvil. Es un lenguaje flexible y existe una gran comunidad de desarrolladores que lo utilizan. JavaScript permite el desarrollo tanto del frontend como del backend. También existen varios frameworks compatibles para el diseño de interfaces.

Tabla 8. Ventajas y desventajas de JavaScript.

Ventajas	Desventajas
<ul style="list-style-type: none"><li>• Compatible con diferentes entornos; navegadores, servidores, dispositivos móviles.</li><li>• Gran cantidad de bibliotecas y frameworks disponibles.</li><li>• Permite la creación de interfaces dinámicas e interactivas.</li><li>• Tiene una amplia comunidad y actualizaciones constantes por parte de la comunidad de desarrolladores.</li></ul>	<ul style="list-style-type: none"><li>• No es tan rápido como lenguajes compilados como C++ o Rust.</li><li>• Puede ser vulnerable a ataques o inyecciones de código.</li><li>• No ofrece una tipificación estricta como TypeScript, lo que puede generar errores en proyectos grandes.</li></ul>

### b. Otros lenguajes de programación considerados:

- Kotlin:  
Es el lenguaje oficial de android por lo que es un lenguaje nativo. Este puede utilizar en el desarrollo multiplataforma con Kotlin Multiplatform Mobile o Java.

Tabla 9. Pros y contras de Kotlin.

Razones por las que se consideró	Aspectos negativos
<ul style="list-style-type: none"><li>• Nativo de Android, tiene compatibilidad con herramientas como Jetpack.</li><li>• Se puede integrar con Java.</li><li>• Código más conciso y seguro en comparación con Java.</li><li>• Soporte para programación funcional y orientada a objetos.</li></ul>	<ul style="list-style-type: none"><li>• No es tan utilizado fuera del ecosistema Android, lo que limita su adopción en otros entornos.</li><li>• Ya que usa Kotlin Multiplatform Mobile, este aún no tiene el mismo nivel de madurez que otros frameworks multiplataforma.</li></ul>

<ul style="list-style-type: none"> <li>• Compatible con Kotlin Multiplatform Mobile para compartir lógica.</li> </ul>	
---	--

- Dart:  
Dart es un lenguaje desarrollado por Google, enfocado en la creación de aplicaciones rápidas y multiplataforma. Se utiliza principalmente con **Flutter**, un framework que permite desarrollar aplicaciones para distintos sistemas operativos.

Tabla 10. Pros y contras de Dart.

Razones por las que se consideró	Aspectos negativos
<ul style="list-style-type: none"> <li>• Se integra perfectamente con Flutter, permitiendo desarrollo rápido y con una sola base de código.</li> <li>• Alto rendimiento gracias a la compilación nativa.</li> <li>• Código moderno, orientado a objetos y con tipado opcional.</li> </ul>	<ul style="list-style-type: none"> <li>• Poca adopción fuera de Flutter.</li> <li>• No es un lenguaje nativo de Android ni iOS, lo que puede generar desventajas en la integración con APIs nativas.</li> <li>• Curva alta de aprendizaje si solo se conoce Java o Kotlin.</li> </ul>

## 2. Frameworks: React Native

- a. React Native es un framework de código abierto que permite la creación de aplicaciones móviles multiplataforma utilizando JavaScript y React. Utiliza componentes nativos para ofrecer una experiencia de usuario similar a las aplicaciones desarrolladas en lenguajes nativos.

Tabla 11. Ventajas y desventajas de React Native.

Ventajas	Desventajas
<ul style="list-style-type: none"> <li>• Permite escribir código una sola vez y ejecutarlo en iOS y Android.</li> <li>• Usa componentes nativos en lugar de WebViews, lo que mejora la velocidad y eficiencia.</li> <li>• Tiene una gran comunidad de desarrolladores y librerías preexistentes.</li> <li>• Soporte para módulos nativos mediante bridging en Kotlin y Swift.</li> </ul>	<ul style="list-style-type: none"> <li>• Puede ser menos eficiente en el uso de memoria comparado con aplicaciones completamente nativas.</li> <li>• Algunas funcionalidades requieren paquetes de terceros, lo que puede generar problemas de compatibilidad.</li> <li>• Tiene una curva de aprendizaje alta.</li> </ul>

b. **Otros frameworks considerados:**

- **Kotlin Multiplatform Mobile**

Permite compartir el código de Kotlin entre Android e iOS, manteniendo interfaces nativas.

Tabla 12. Pros y contras de Kotlin Multiplatform Mobile.

Razones por las que se consideró	Aspectos negativos
<ul style="list-style-type: none"><li>• Permite aprovechar la UI nativa de cada plataforma.</li><li>• Compatible con Kotlin.</li><li>• Mayor flexibilidad al permitir compartir solo la lógica y no la interfaz.</li></ul>	<ul style="list-style-type: none"><li>• No es una solución completamente multiplataforma como Flutter.</li><li>• Requiere desarrollo separado de la UI para cada plataforma.</li><li>• Tiene más complejidad en comparación con Flutter.</li></ul>

- **Flutter**

Flutter es un framework de UI de Google que permite desarrollar aplicaciones para Android, iOS, Web y escritorio con un solo código base.

Tabla 13. Pros y contras de Flutter.

Razones por las que se consideró	Aspectos negativos
<ul style="list-style-type: none"><li>• Usa un solo código base para múltiples plataformas.</li><li>• Widgets personalizables y altamente optimizados.</li><li>• Alto rendimiento gracias a su propio motor gráfico.</li></ul>	<ul style="list-style-type: none"><li>• El tamaño de las aplicaciones es mayor en comparación con apps nativas.</li><li>• Puede ser más difícil integrar con código nativo.</li><li>• Al ser un framework en desarrollo no tiene el mismo nivel de adopción que los frameworks nativos.</li></ul>

3. **Base de datos: PostgreSQL**

a. PostgreSQL es un sistema de gestión de bases de datos relacional de código abierto, es robusto, flexible y cumple con estándares SQL. También soporta consultas avanzadas, integridad referencial y transacciones ACID.

Tabla 14. Ventajas y desventajas de PostgresSql.

Ventajas	Desventajas
<ul style="list-style-type: none"> <li>• Soporte para Transacciones ACID.</li> <li>• Permite la creación de funciones y extensiones personalizadas.</li> <li>• Tiene alta concurrencia, maneja múltiples conexiones eficientemente.</li> <li>• Funciona bien en aplicaciones con grandes volúmenes de datos.</li> </ul>	<ul style="list-style-type: none"> <li>• Puede ser más pesado que MySQL en términos de uso de memoria.</li> <li>• Su curva de aprendizaje es más compleja que otros sistemas como Firebase o MySQL.</li> </ul>

**b. Otras opciones de bases de datos considerados:**

- Firebase

Firebase es una plataforma de Google que ofrece una base de datos en la nube con sincronización en tiempo real y un backend sin servidor.

Tabla 15. Pros y contras de Firebase.

Razones por las que se consideró	Aspectos negativos
<ul style="list-style-type: none"> <li>• Fácil integración con Android y Flutter.</li> <li>• Sincronización en tiempo real.</li> <li>• Incluye autenticación, almacenamiento y hosting.</li> </ul>	<ul style="list-style-type: none"> <li>• Modelo NoSQL.</li> <li>• Puede volverse costoso en aplicaciones con mucho tráfico.</li> </ul>

- Supabase

Supabase es una alternativa de código abierto a Firebase, basada en PostgreSQL.



Tabla 16. Pros y contras de Supabase.

Razones por las que se consideró	Aspectos negativos
<ul style="list-style-type: none"> <li>• Usa PostgreSQL, lo que permite consultas SQL avanzadas.</li> <li>• Código abierto, lo que evita depender de una empresa propietaria.</li> <li>• Más control sobre la base de datos en comparación con Firebase.</li> </ul>	<ul style="list-style-type: none"> <li>• Tiene menos soporte ya que aún está en desarrollo.</li> <li>• La sincronización en tiempo real no es tan optimizada como en Firebase.</li> </ul>

## Lista con las tareas propuestas

Tabla 17. Lista de propuestas y miembro asignado.

Tarea	Miembro responsable
Recopilación de partes de los informes anteriores	Isabella Obando, Anggelie Velásquez
Lista de requisitos funcionales	Mia Fuentes
Enlazar cada requisito funcional a una historia de usuario	Mia Fuentes
Cambios de diseño del prototipo	Anggelie Velásquez
Elabore un diagrama de clases	Anthony Lou Schwank
Descripción de las clases	Anthony Lou Schwank
Diagrama de paquetes	Roberto Barreda
Descripción de cada paquete y de sus componentes	Roberto Barreda
Diagrama de clases persistentes	Anggelie Velásquez
Diagrama Entidad Relación	Anggelie Velásquez
Selección de la tecnología a utilizar	Isabella Obando
Seleccionar la tecnología para almacenamiento	Isabella Obando

Presentación	Todos los integrantes
--------------	-----------------------

## Referencias

*Guatemala: el país de América Latina con más desnutrición crónica* / Acción contra el  
*Hambre*. (s. f.). Guatemala: El País de América Latina Con Más Desnutrición Crónica  
| Acción Contra el Hambre. <https://accioncontraelhambre.org/es/guatemala-el-pais-america-latina-mas-desnutricion-cronica>

## **Anexos**

### **Presentación:**

[https://www.canva.com/design/DAGh\\_2q08rI/b5F4mdG25scdfgdf\\_C4KRw/edit?utm\\_content=DAGh\\_2q08rI&utm\\_campaign=designshare&utm\\_medium=link2&utm\\_source=sharebutton](https://www.canva.com/design/DAGh_2q08rI/b5F4mdG25scdfgdf_C4KRw/edit?utm_content=DAGh_2q08rI&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton)

### **Github:**

<https://github.com/bar23354/Ra-ces-de-Vida.git>

### **Informe de Gestión:**

[Gestion de tiempo.xlsx](#)

### **Figma:**

<https://www.figma.com/design/vv1noCcotbjbJnaSo8opH3/Raices-de-vida?node-id=0-1&t=MjJ33NzuWqmKqnjW-1>