



## Universidad Continental

Facultad de Ingeniería

***“Sistema de gestión de donación de sangre y registro de donantes”***

**ASIGNATURA:**

Análisis y Diseño de software

**DOCENTE:**

- Osorio Contreras, Rosario Delia

**INTEGRANTES:**

- DE LA CRUZ RAMOS, Benn Julius
- PENADILLO GONZALES, Gilmer Antonny
- RAMIREZ ESPIRITU, Junior Saul

**NRC: 62152**

**HUANCAYO - PERÚ**

**Links trabajados:**

**-Link de Github:**

<https://github.com/Antonnyy/Sistema-de-donaci-n->

**-Link de Canva:**

[https://www.canva.com/design/DAG0aUAJLIY/qD3Ar2Hlcnz33sKLzmwbg/edit?utm\\_content=DA\\_G0aUAJLIY&utm\\_campaign=designshare&utm\\_medium=link2&utm\\_source=sharebutton](https://www.canva.com/design/DAG0aUAJLIY/qD3Ar2Hlcnz33sKLzmwbg/edit?utm_content=DA_G0aUAJLIY&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton)

**-Link de Jira:**

<https://continental-team-w6zljlux.atlassian.net/jira/software/projects/DON/boards/34/timeline?atIOrigin=eyJpljoiYTUxOTQ5NWRhMTI0NDkyMjg1YjhjNjIjODRmMmZmYWUiLCJwljoiaij9>

**-Link de Figma:**

<https://www.figma.com/design/XNAPVfisMqUudNO1CaK2Rz/Untitled?node-id=20-2&t=IdW0EVtI5fosxLzf-1>

## Índice

Índice.....	2
Unidad I – Fundamentos y Modelado Inicial (Semanas 1–4).....	3
Capítulo 1. Presentación del Proyecto.....	3
Capítulo 2. Análisis de Necesidades y Requerimientos.....	4
Capítulo 3. Modelos Iniciales del Sistema.....	6
Modelo funcional (diagrama de contexto, casos de uso generales):.....	6
Modelo de procesos.....	7
Diagrama de actividad UML.....	7
Modelo de datos (Modelo E-R).....	8
Unidad II – Modelos de Diseño y Metodología Ágil (Semanas 5–7).....	9
Capítulo 4. Modelos de Diseño.....	9
Modelo estructural (diagrama de clases inicial):.....	9
Modelo de interacción (diagrama de secuencia):.....	10
Capítulo 5. Metodología de Trabajo (SCRUM).....	10
Sprint 1 – Núcleo del sistema.....	11
Sprint 2 – Gestión y control.....	11
Capítulo 6. Diseño de arquitectura y patrones.....	12
Arquitectura en Capas (Layered Architecture).....	13
Patrón Singleton.....	13

Patrón Observer (Observador).....	14
Capítulo 7. Diseño detallado de la base de datos.....	15
Respaldo.....	24
Capítulo 8. Diseño detallado del sistema en red y móvil.....	24
Capítulo 9. Diseño de Interfaz y Experiencia de Usuario (UX/UI).....	27
Conclusiones y Recomendaciones.....	33
Anexos.....	35
🕒 En resumen crítico.....	38

## Índice de tablas

## **Unidad I – Fundamentos y Modelado Inicial (Semanas 1–4)**

### **Capítulo 1. Presentación del Proyecto**

#### **ODS vinculado:**

El proyecto se relaciona con el **ODS 3: Salud y Bienestar**, que busca “*Garantizar una vida sana y promover el bienestar para todos en todas las edades*”. El sistema de gestión de donación de sangre contribuye a un abastecimiento seguro y suficiente, fortaleciendo la infraestructura de salud para responder a emergencias, reducir muertes evitables y mejorar la calidad de vida de la población.

#### **Organización o institución beneficiaria:**

La institución beneficiaria propuesta es el Banco de Sangre Regional del Hospital Carrión de Huancayo, ya que cuenta con un servicio de hemoterapia y enfrenta dificultades en el abastecimiento y control de donantes. Este proyecto busca fortalecer su sistema de registro digital, mejorar la gestión del inventario de sangre y facilitar la coordinación con donantes y otras instituciones de salud, con la posibilidad de replicar la solución en otras regiones del país.

#### **Problema identificado:**

El **Hospital Daniel Alcides Carrión de Huancayo** no cuenta con un sistema digital para gestionar las donaciones ni registrar a los donantes. Esto ocasiona demoras en la búsqueda de sangre compatible, riesgo de pérdida de información y poca eficiencia en el control del inventario, lo que limita la disponibilidad de sangre segura en situaciones de emergencia.

#### **Problema de investigación:**

¿Cómo la falta de un sistema digital de gestión de donación de sangre y registro de donantes en el Hospital Daniel Alcides Carrión de Huancayo afecta el abastecimiento y la atención oportuna en emergencias médicas?

#### **Solución propuesta EXPLICADO A DETALLE:**

Se propone implementar un sistema digital de gestión de donación de sangre y registro de donantes en el Hospital Daniel Alcides Carrión de Huancayo. Este sistema permitirá registrar y mantener actualizada la base de donantes, agilizar la búsqueda de compatibilidades, controlar el inventario de sangre con alertas de vencimiento y escasez, y enviar notificaciones automáticas a los donantes en campañas o emergencias. Con ello se busca reducir demoras, evitar pérdidas de

unidades, garantizar la trazabilidad y mejorar la disponibilidad de sangre segura y oportuna para la atención médica, en concordancia con las recomendaciones de la OMS y la OPS sobre suficiencia y seguridad del suministro de sangre.

## Capítulo 2. Análisis de Necesidades y Requerimientos

### Descripción del problema:

El Hospital Daniel Alcides Carrión de Huancayo no cuenta con un sistema digital que gestione adecuadamente las donaciones de sangre ni el registro de donantes. Esto genera demoras en la localización de sangre compatible, riesgo de pérdida de información y poca eficiencia en el control del inventario, lo que afecta la atención en emergencias.

### Necesidades de los usuarios:

- Que los donantes puedan registrarse fácilmente y recibir notificaciones cuando se les necesite.
- Que el hospital cuente con un inventario actualizado de las unidades de sangre disponibles.
- Que los médicos puedan solicitar sangre de forma rápida y confiable.
- Que exista trazabilidad y seguridad en el manejo de la información.

### Requerimientos funcionales (RF):

RF1: Registro de donantes con sus datos y grupo sanguíneo.

RF2: Gestión de solicitudes de sangre y asignación de unidades compatibles.

RF3: Control del inventario con alertas de stock bajo o vencimiento.

RF4: Envío de notificaciones a donantes para campañas o emergencias.

RF5: Generación de reportes y estadísticas para la administración.

### Requerimientos no funcionales (RNF):

RNF1: Seguridad y protección de los datos sensibles de los donantes.

RNF1: Facilidad de uso con interfaces simples e intuitivas.

RNF1: Disponibilidad del sistema para garantizar atención en cualquier momento.

RNF1: Escalabilidad para que pueda crecer y usarse en otros hospitales.

### **Requerimientos de dominio**

RD1: Cumplimiento de la normativa del MINSA y la Ley de Protección de Datos Personales en Perú.

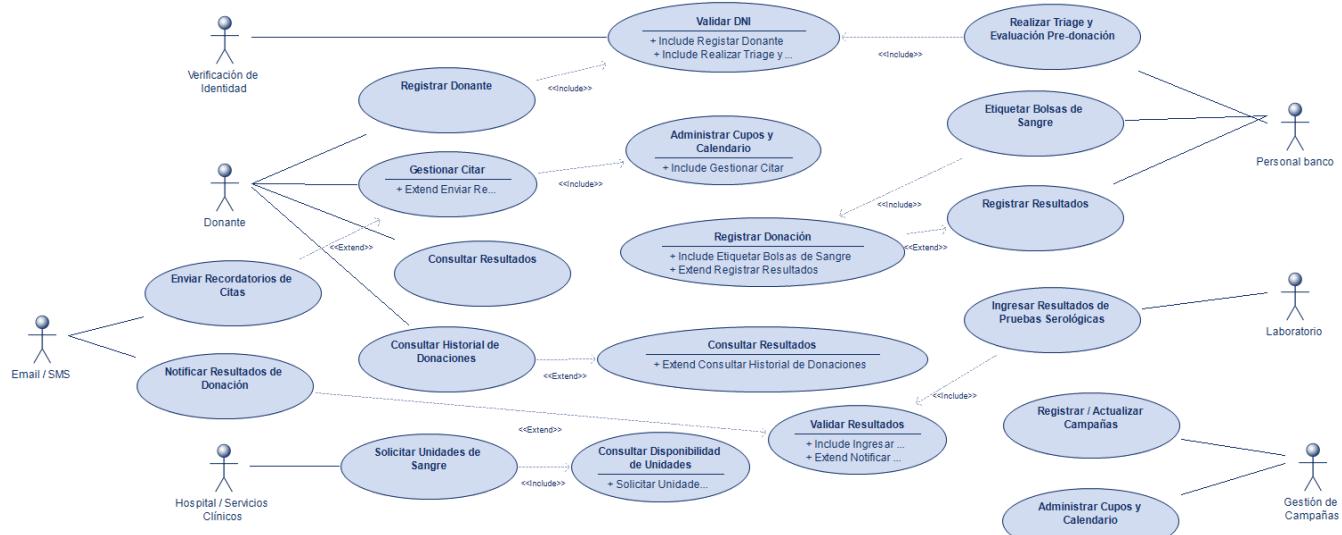
RD1: Validación de criterios médicos de elegibilidad para los donantes (tiempo mínimo entre donaciones, estado de salud, etc.).

RD1: Gestión de compatibilidad según los grupos sanguíneos ABO y Rh.

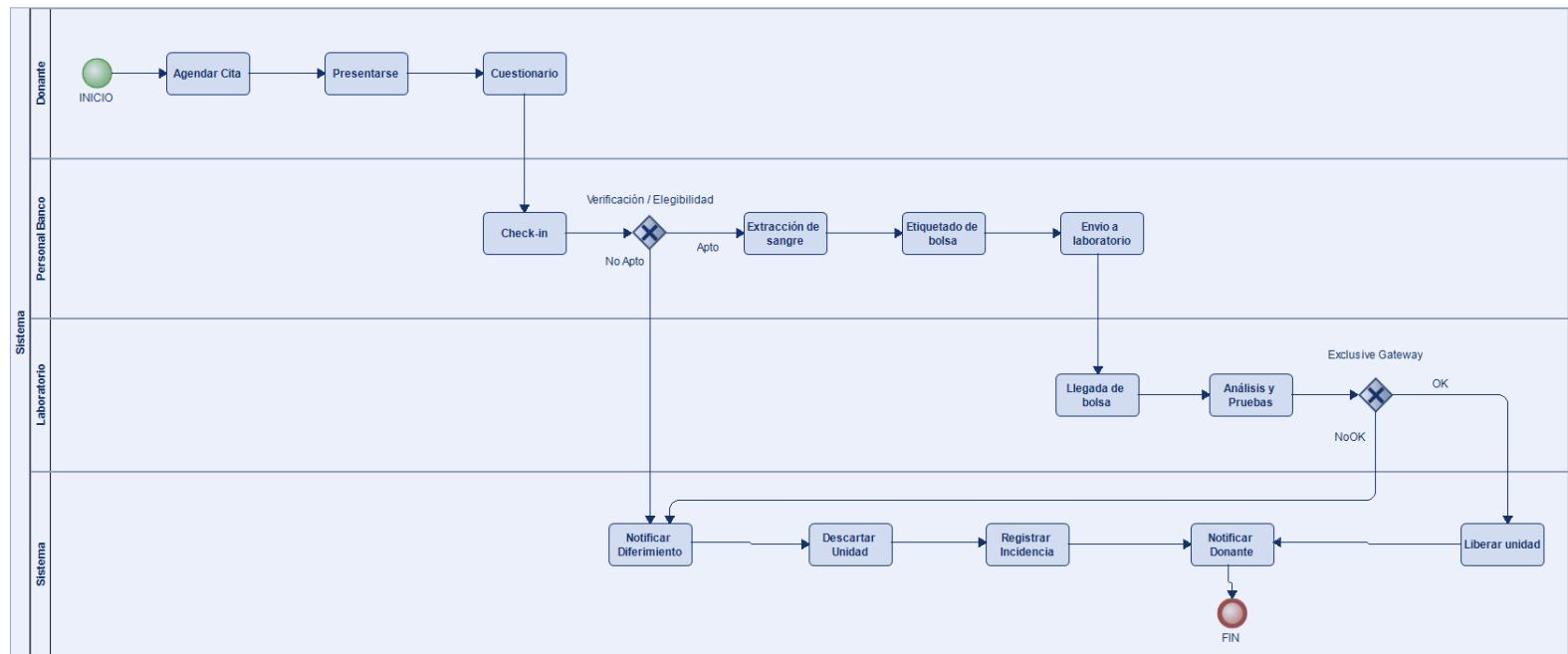
ID	Tipo de requisito	Descripción	Criterio de aceptación
RF01	Funcional	Registro de donantes con sus datos y grupo sanguíneo.	Un donante se registra y queda almacenado con un código único y grupo sanguíneo.
RF02	Funcional	Gestión de solicitudes de sangre y asignación de unidades compatibles.	Una solicitud se ingresa y el sistema asigna automáticamente una unidad compatible.
RF03	Funcional	Control del inventario con alertas de stock bajo o vencimiento.	Cuando el stock es menor al mínimo o una unidad va a vencer, se genera una alerta.
RF04	Funcional	Envío de notificaciones a donantes para campañas o emergencias.	Los donantes reciben un correo o SMS al registrarse una campaña/emergencia.
RF05	Funcional	Generación de reportes y estadísticas para la administración.	Se genera un archivo PDF/Excel con los datos requeridos.
RI01	Interfaz	El sistema debe contar con una interfaz simple e intuitiva para el usuario.	Una prueba de usuario demuestra que el sistema puede usarse sin capacitación extensa.
RR01	Rendimiento	El sistema debe garantizar alta disponibilidad en cualquier momento.	Monitoreo del sistema muestra ≥ 99.5% de disponibilidad.
RL01	Lógicos y de Datos	El sistema debe gestionar la compatibilidad de sangre según grupos ABO y Rh.	Al asignar una unidad, el sistema valida la compatibilidad.
RS01	Seguridad y Privacidad	El sistema debe proteger los datos sensibles de los donantes mediante encriptación.	Auditoría confirma que contraseñas y datos sensibles están encriptados.
RS02	Seguridad y Privacidad	El sistema debe validar los criterios médicos de elegibilidad de donantes.	Si el donante no cumple el tiempo mínimo o estado de salud, el sistema bloquea la donación.
RD01	Restricciones	El sistema debe cumplir con la normativa del MINSA y la Ley de Protección de Datos en Perú.	Una revisión legal confirma el cumplimiento de la normativa.
RD02	Restricciones	El sistema debe garantizar escalabilidad para hospitales en el país.	Pruebas de despliegue confirman integración en nuevas sedes sin pérdida de rendimiento.

## Capítulo 3. Modelos Iniciales del Sistema

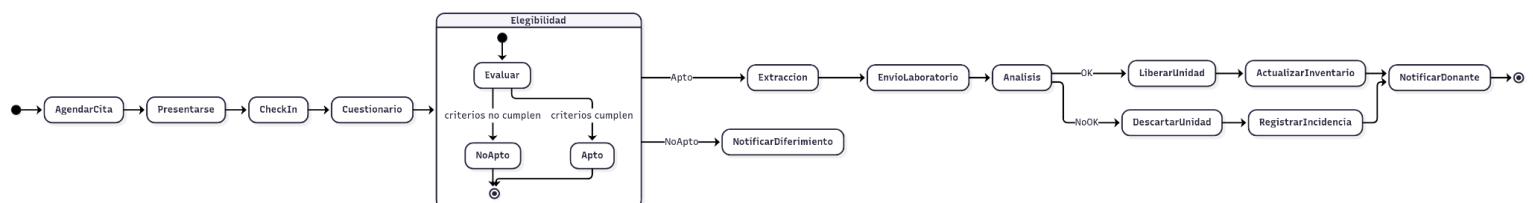
### Modelo funcional (diagrama de contexto, casos de uso generales):



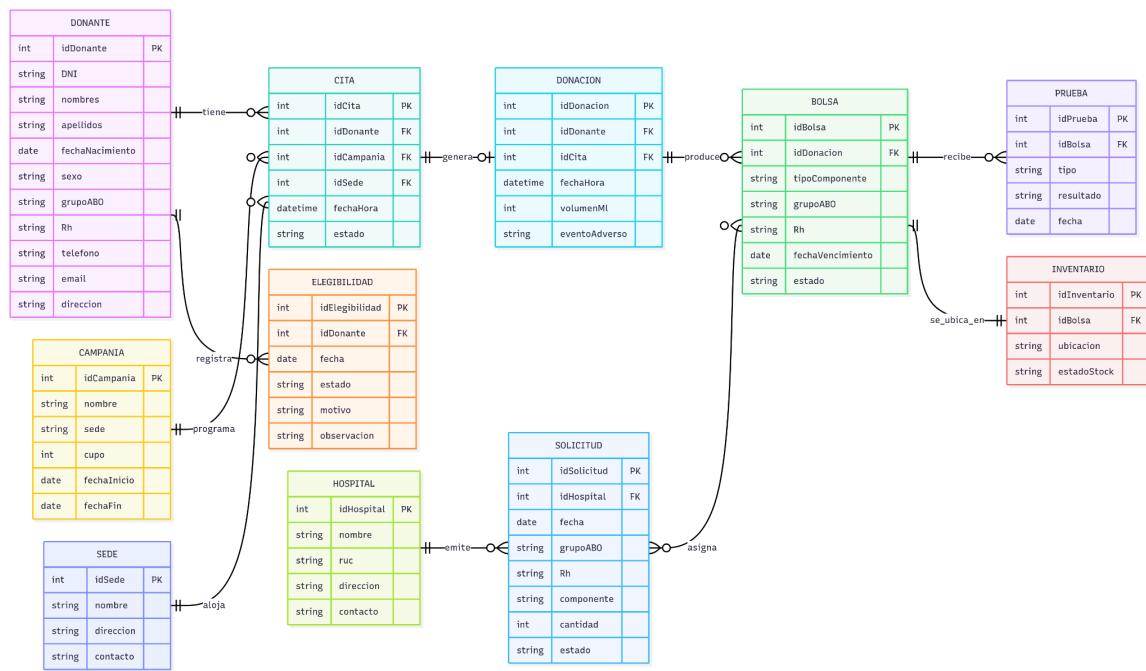
## Modelo de procesos



## Diagrama de actividad UML



## Modelo de datos (Modelo E-R)



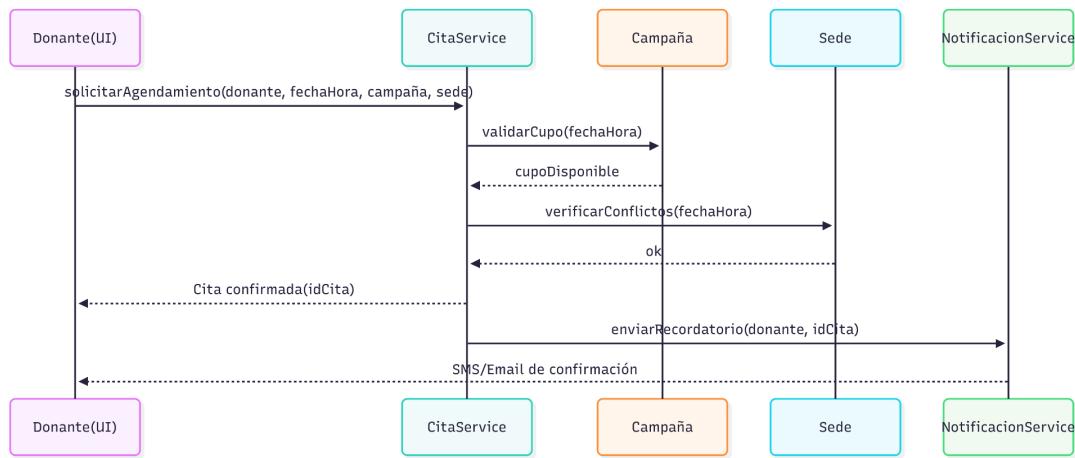
# Unidad II – Modelos de Diseño y Metodología Ágil (Semanas 5–7)

## Capítulo 4. Modelos de Diseño

Modelo estructural (diagrama de clases inicial):



### Modelo de interacción (diagrama de secuencia):



## Capítulo 5. Metodología de Trabajo (SCRUM)

Definición de la metodología ágil usada:

Se empleó Scrum con sprints de 2 semanas, roles claros y entregables incrementales.

- Roles: Product Owner (usuario/beneficiario del hospital), Scrum Master (coordinación del proceso), Equipo de Desarrollo: Equipo2 (frontend / backend / QA).
- Artefactos: Product Backlog (épicas e históricas), Sprint Backlog e Incremento.
- Eventos: Sprint Planning, Daily, Sprint Review y Retrospective.
- DoR/DoD (resumen): historias listas con criterio de aceptación y terminar “Hecho” con pruebas básicas y demo en Review.

Se empleó **SCRUM**, que organiza el trabajo en **sprints cortos**, con roles claros (Product Owner, Scrum Master y Equipo de Desarrollo) y entregables incrementales.

Backlog del producto (épicas e historias de usuario):

#### EPIC 1: Gestión de Donantes

HU01 Registrar donante · HU02 Buscar donante · HU03 Editar donante · HU04 Eliminar donante

#### EPIC 2: Gestión de Campañas

HU05 Crear campaña · HU06 Inscribir donante en campaña · HU07 Editar/Cancelar campaña

#### EPIC 3: Gestión de Donaciones

HU08 Registrar donación · HU09 Ver historial de donaciones · HU10 Generar certificado

#### EPIC 4: Reportes y Estadísticas

HU11 Reporte de donantes · HU12 Reporte de campañas · HU13 Reporte de stock

## EPIC 5: Seguridad y Autenticación

HU14 Iniciar sesión · HU15 Recuperar contraseña · HU16 Administrar roles de usuario

Planificación de sprints (Sprint 1 y Sprint 2):

### Sprint 1 – Núcleo del sistema

- **Fechas:** 25/09/2025 – 08/10/2025
- **Meta:** Base funcional: registro y búsqueda de donantes, registro de donaciones, creación de campañas e inicio de sesión.
- **Historias:** **HU01, HU02, HU05, HU08, HU14**
- **Entregable esperado:** MVP inicial con login, formulario de donantes, buscador simple, alta de campañas y registro de una donación.

### Sprint 2 – Gestión y control

- **Fechas:** 09/10/2025 – 22/10/2025
- **Meta:** Completar gestión y accesos: edición/eliminación de donantes, inscripciones y mantenimiento de campañas, historial y seguridad.
- **Historias:** **HU03, HU04, HU06, HU07, HU09, HU15, HU16**
- **Entregable esperado:** MVP ampliado con gestión completa de donantes y campañas, historial visible y funciones de seguridad (recuperación y roles).

**Nota:** Se dejó planificado un **Sprint 3 – Reportes y certificados (23/10/2025 – 05/11/2025)** con **HU10, HU11, HU12, HU13** para consolidar reportes y certificados.

Herramientas utilizadas (Jira, Draw.io, Dbdiagram.io, etc.):

- Jira: gestión Scrum.

The screenshot shows the Jira interface for the 'ODS3 – Donación de Sangre' project. The left sidebar displays the backlog with various epics and user stories. The main area shows the 'Principal' sprint backlog, which includes tasks for 'GESTIÓN DE DONANTES', 'GESTIÓN DE CAMPANAS', 'GESTIÓN DE DONACIONES', and 'SEGURIDAD Y AUTENTICACIÓN'. Each task has a status bar indicating progress (e.g., 5%, 100%) and a due date of '8 oct' or '22 oct'. A 'Quickstart' button is visible at the bottom right.

- Planificación de sprints (Sprint 1, Sprint 2 y 3 Sprint):

<input type="checkbox"/> Sprint 1 – Núcleo del sistema 25 sep – 8 oct (4 actividades)	Implementar funciones base: registro y búsqueda de donantes, registro de donaciones, creación de campañas e inicio de sesión.	23	0	0	Iniciar sprint	...
<input type="checkbox"/> DON-9 Buscar donante	GESTIÓN DE DONANTES	TAREAS POR HACER	8 oct	5	JE	
<input type="checkbox"/> DON-12 Crear campaña	GESTIÓN DE CAMPAÑAS	TAREAS POR HACER	8 oct	5	BR	
<input type="checkbox"/> DON-15 Registrar donación	GESTIÓN DE DONACIO...	TAREAS POR HACER	8 oct	8	GG	
<input type="checkbox"/> DON-21 Iniciar sesión	SEGURIDAD Y AUTENT...	TAREAS POR HACER	8 oct	5	JE	
<input type="checkbox"/> Sprint 2 – Gestión y control 9 oct – 22 oct (7 actividades)	Completar gestión de datos y accesos: edición/eliminación de donantes, inscripciones y cambios de campañas, historial del donante y funciones de seguridad (recuperación y roles).	34	0	0	Iniciar sprint	...
<input type="checkbox"/> DON-10 Editar donante	GESTIÓN DE DONANTES	TAREAS POR HACER	22 oct	5	JE	
<input type="checkbox"/> DON-11 Eliminar donante	GESTIÓN DE DONANTES	TAREAS POR HACER	22 oct	3	JE	
<input type="checkbox"/> DON-13 Inscribir donante en campaña	GESTIÓN DE CAMPAÑAS	TAREAS POR HACER	22 oct	5	BR	
<input type="checkbox"/> DON-14 Editar/Cancelar campaña	GESTIÓN DE CAMPAÑAS	TAREAS POR HACER	22 oct	5	JE	
<input type="checkbox"/> DON-16 Ver historial de donaciones	GESTIÓN DE DONACIO...	TAREAS POR HACER	22 oct	5	BR	
<input type="checkbox"/> DON-22 Recuperar contraseña	SEGURIDAD Y AUTENT...	TAREAS POR HACER	22 oct	3	JE	
<input type="checkbox"/> DON-23 Administrar roles de usuario	SEGURIDAD Y AUTENT...	TAREAS POR HACER	22 oct	8	JE	
<input type="checkbox"/> Sprint 3–ReportesCertificados 23 oct – 5 nov (4 actividades)	Entregar valor analítico y de evidencia: certificados de donación y reportes de donantes, campañas y stock por tipo sanguíneo.	23	0	0	Iniciar sprint	...
<input type="checkbox"/> DON-17 Generar certificado de donación	GESTIÓN DE DONACIO...	TAREAS POR HACER	5 nov	5	JE	
<input type="checkbox"/> DON-18 Reporte de donantes	REPORTES Y ESTADÍST...	TAREAS POR HACER	5 nov	5	JE	
<input type="checkbox"/> DON-19 Reporte de campañas	REPORTES Y ESTADÍST...	TAREAS POR HACER	5 nov	5	GG	
<input type="checkbox"/> DON-20 Reporte de stock de sangre	REPORTES Y ESTADÍST...	TAREAS POR HACER	5 nov	8	BR	

- Google Drive/Docs: Documentos del informe y evidencias.
- Modelio: diagrama de casos de uso y proceso.
- Mermaidchart.com: diagrama UML, E - R, estructural e interacción.

## Capítulo 6. Diseño de arquitectura y patrones

### 6.1 Estrategia de Diseño del Software

Para cumplir con los requerimientos funcionales del sistema (gestión de donantes, inventario y campañas) y los no funcionales (disponibilidad y seguridad de datos médicos), se ha seleccionado una estrategia de diseño centrada en la integridad de los datos y la escalabilidad modular.

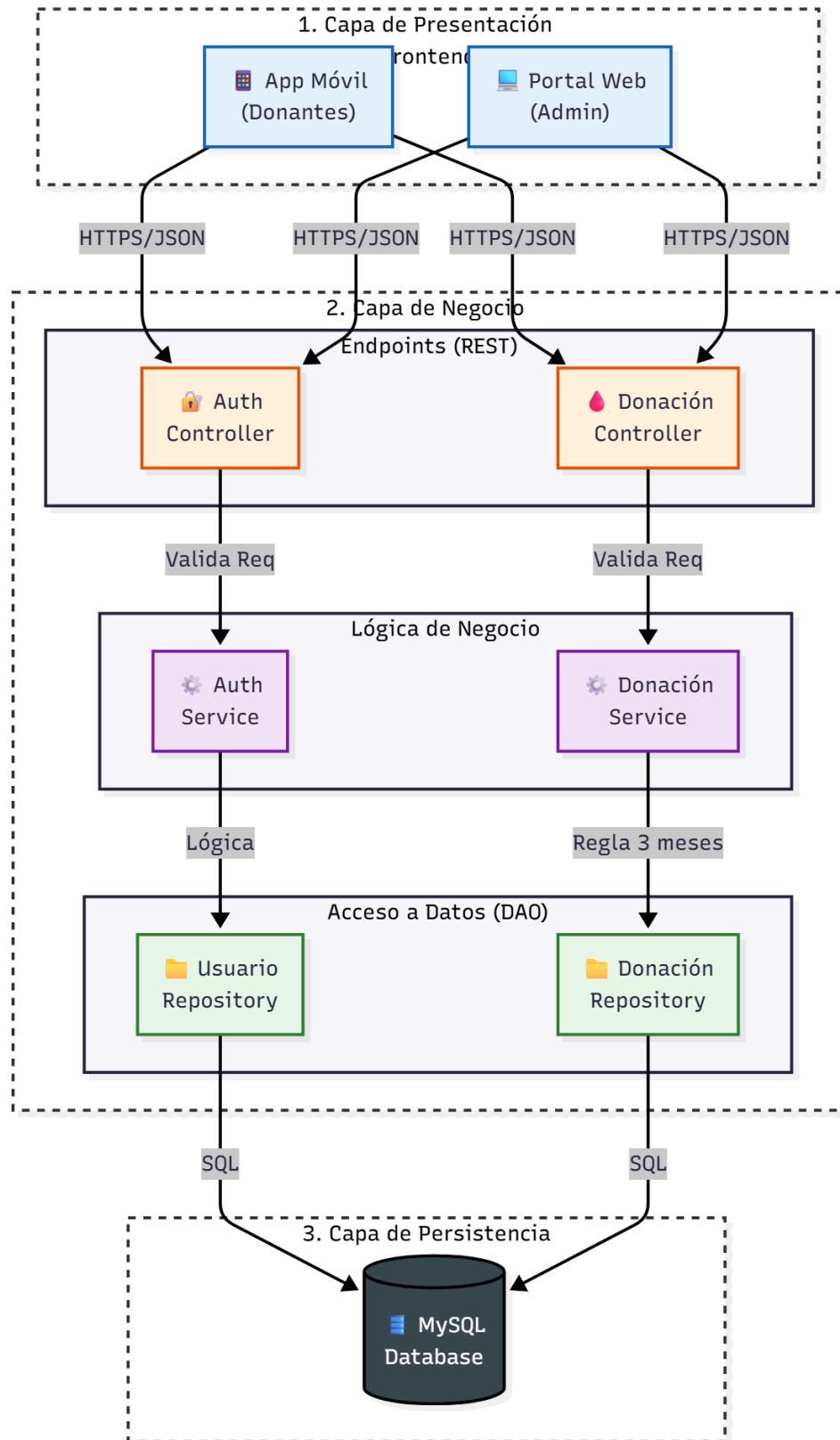
Dado que el Hospital Daniel Alcides Carrión opera en un entorno crítico donde la información debe ser consistente (ACID), se ha descartado el uso de arquitecturas distribuidas complejas en favor de una estructura centralizada pero desacoplada, permitiendo que tanto la aplicación móvil (Donantes) como el portal web (Administrativos) consuman la misma lógica de negocio.

### 6.2 Tipo de la arquitectura del sistema

#### Arquitectura en Capas (Layered Architecture)

Se ha implementado una Arquitectura en Capas (Layered Architecture) basada en el estilo Cliente-Servidor. Esta decisión desacopla la interfaz de usuario de la lógica de negocio, permitiendo que tanto la Aplicación Móvil (Donantes) como el Portal Web (Administrativos) consuman la misma lógica a través de una API centralizada.

- **Capa de Presentación (Frontend):** Es la capa que interactúa con el usuario final.  
Se divide en dos clientes:
  - **App Móvil (Donantes):** Desarrollada para Android, permite el registro rápido y recepción de notificaciones.
  - **Portal Web (Personal de Salud):** Desarrollada en React, enfocada en la gestión masiva del inventario y reportes.
  - **Objetivo:** Garantizar una experiencia de usuario intuitiva (UX) sin contener lógica compleja.
- **Capa de Lógica de Negocio (Backend / API):** Aquí reside el núcleo del sistema.  
Expuesta como una API RESTful, se encarga de:
  - Validar la elegibilidad del donante (regla de los 3 meses).
  - Procesar la compatibilidad sanguínea (ABO/Rh).
  - Gestionar la seguridad y roles (JWT).
- **Capa de Acceso a Datos (Persistencia):** Gestiona la comunicación con la base de datos **MySQL**. Utiliza repositorios para abstraer las consultas SQL, asegurando que los datos sensibles (información médica) se almacenen de forma íntegra y segura.



Arquitectura	Decisión	Justificación basada en Requerimientos (RF/RNF)
Microservicios	Descartada	Aunque permite escalabilidad masiva, añade una complejidad de red (latencia y fallos distribuidos) innecesaria para el volumen transaccional de un hospital regional. Pondría en riesgo el <b>RNF3 (Disponibilidad)</b> debido a la infraestructura limitada.
MVC Monolítico	Descartada	Al acoplar las vistas (HTML) con el servidor, impide reutilizar la lógica para la App Móvil nativa. Incumple el <b>RNF4 (Escalabilidad Multiplataforma)</b> requerido por el cliente.
En Capas (N-Tier)	Seleccionada	<b>Cumplimiento Total.</b> La separación de la API REST permite atender tanto a la Web como a la App Móvil simultáneamente. La capa de datos aislada facilita la seguridad de los datos sensibles ( <b>RNF1 Seguridad</b> ), vital para cumplir con la normativa del MINSA.

### 6.3 Patrones del diseño aplicados

Para el sistema de **gestión de donación de sangre y registro de donantes**, hemos decidido aplicar **dos patrones de diseño clave** que complementan nuestra **arquitectura en capas**. Estos patrones se han seleccionado teniendo en cuenta los **requisitos del sistema**, la **escalabilidad, la seguridad y la facilidad de mantenimiento**. Los patrones elegidos son **Singleton** y **Observer**, los cuales resuelven problemas específicos dentro de las capas del sistema.

## Patrón Singleton

### Descripción del patrón:

El **patrón Singleton** garantiza que una clase tenga **una única instancia** en todo el ciclo de vida del sistema. Este patrón es ideal cuando se requiere **un acceso global y controlado** a un servicio o recurso específico, como una conexión a la base de datos o un servicio de notificación.

### Aplicación en el sistema:

En el sistema de donación de sangre, aplicamos el patrón **Singleton** en dos áreas clave:

**Conexión a la base de datos:** Se asegura que solo exista **una instancia de la conexión** a la base de datos, evitando problemas de **sobrecarga de recursos y concurrencia**. Al tener una única instancia de conexión, el sistema puede manejar las consultas de manera más **eficiente**.

**Módulo de notificaciones:** Para las **notificaciones automáticas** a los donantes, se usa **una sola instancia** que maneja el envío de mensajes. Esto garantiza que **todos los mensajes** se gestionan desde un **único punto de control**, lo que mejora la **eficiencia** y evita la **duplicación de instancias**.



### Ventajas:

**Optimización de recursos:** Al evitar la creación de múltiples instancias, se **ahorran recursos** y se **mejora el rendimiento** del sistema.

**Acceso controlado:** La **gestión centralizada** de los servicios, como la conexión a la base de datos y el envío de notificaciones, permite un **acceso seguro y eficiente** a estos servicios.

---

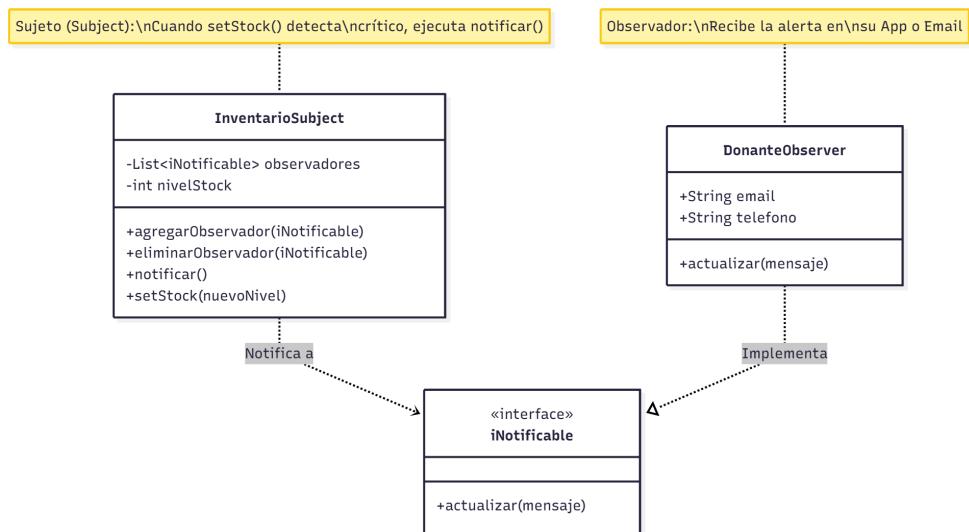
## Patrón Observer (Observador)

### Descripción del patrón:

El patrón **Observer** permite que un **sujeto** notifique automáticamente a **varios observadores** cuando su estado cambia. Este patrón es útil cuando un sistema debe **reaccionar a eventos de manera automática**, sin que los observadores (en este caso, los donantes) estén directamente acoplados al sistema que genera el evento.

### Aplicación en el sistema:

En el sistema de donación de sangre, el patrón **Observer** se aplica en la gestión de **notificaciones** a los donantes. Cuando ocurre un evento importante, como una **baja en el inventario de sangre** o una **emergencia médica**, el sistema **publica un evento** y **todos los donantes registrados** reciben automáticamente una **notificación**. Los donantes están **suscritos** a este tipo de eventos, y el sistema les envía notificaciones en tiempo real.



### Ventajas:

**Desacoplamiento:** El sistema de **notificaciones** y los **donantes** están **desacoplados**, lo que permite **modificar la lógica del sistema** o **agregar nuevos tipos de notificaciones** sin afectar el resto del sistema.

**Reactividad:** El sistema **reacciona automáticamente** a los eventos sin necesidad de intervención manual por parte de los usuarios o administradores, lo que mejora la **eficiencia** y reduce el riesgo de olvidos o errores.

**Escalabilidad:** Se pueden agregar más **observadores** (por ejemplo, más tipos de notificaciones) sin necesidad de modificar el **núcleo del sistema**.

#### 6.4 Diseño estructural

El diseño estructural detalla la descomposición interna del software, alineándose con la arquitectura en capas (N-Tier) definida en la sección 6.2. La solución implementa una separación estricta entre el Frontend y el Backend para garantizar la mantenibilidad y la seguridad de los datos médicos.

Como se aprecia en la **Figura 6.4**, el sistema se organiza en tres grandes paquetes lógicos:

**1. Paquete Cliente (Capa de Presentación):** Esta estructura se aplica tanto para el aplicativo móvil como para la web, cumpliendo con el **RNF4 (Escalabilidad Multiplataforma)**:

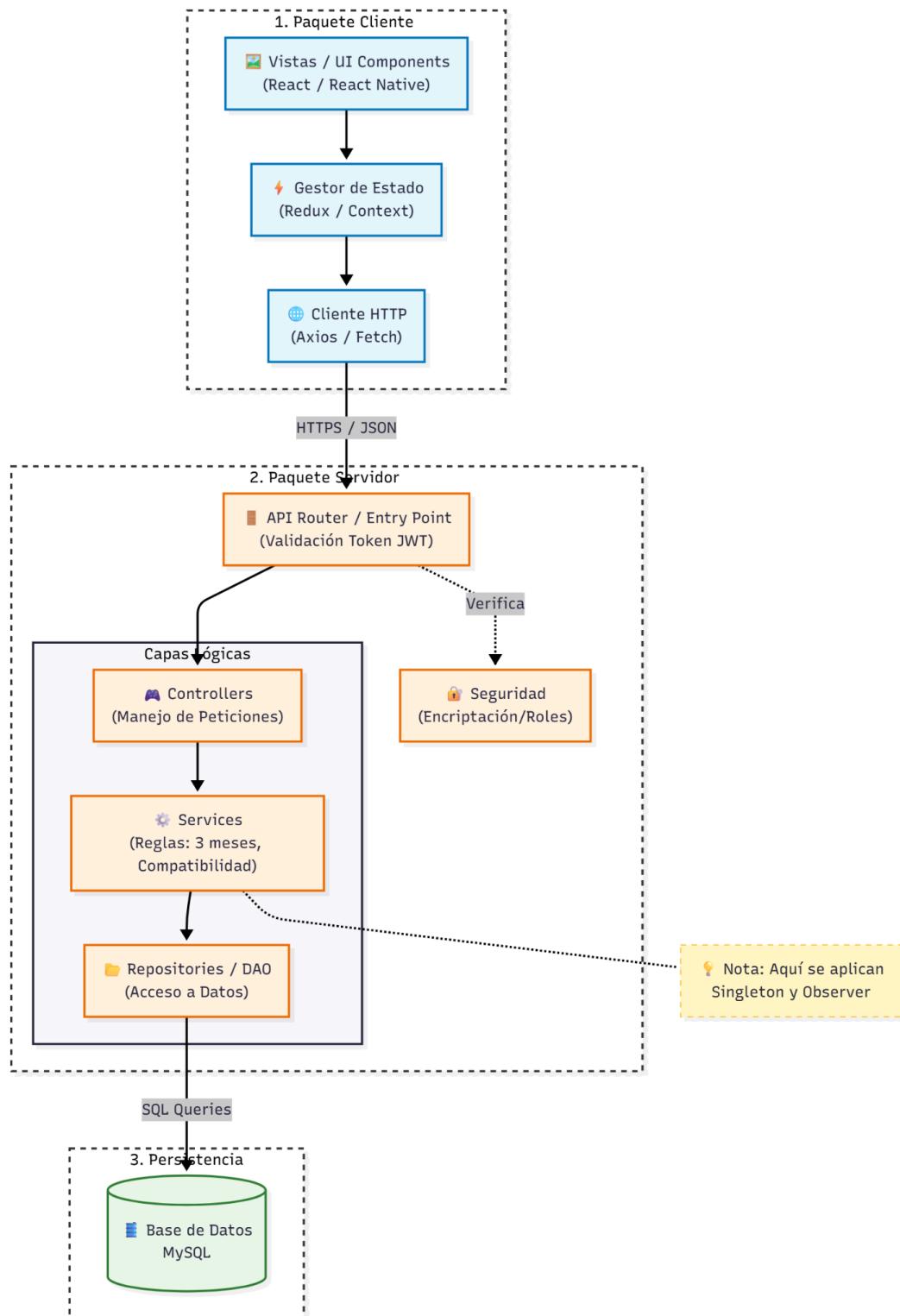
- **Vistas / UI:** Componentes visuales (desarrollados en React y React Native) que capturan la interacción del usuario.
- **Gestor de Estado (Redux/Context):** Módulo que centraliza la información temporal (como la sesión del usuario o el carrito de donación) para evitar inconsistencias visuales y reducir llamadas al servidor.
- **Cliente HTTP (Axios):** Responsable de la comunicación asíncrona con el servidor. Se encarga de injectar el token de seguridad en las cabeceras de cada petición JSON.

**2. Paquete Servidor (Capa de Negocio):** Implementa el patrón **Controller-Service-Repository** para desacoplar responsabilidades:

- **API Router / Entry Point:** Punto único de entrada que intercepta las peticiones. Valida el Token JWT (**RNF1 Seguridad**) antes de permitir el acceso a los controladores.
- **Controllers:** Reciben la petición HTTP, validan el formato de los datos de entrada y delegan la operación al servicio correspondiente.
- **Services:** Contienen el núcleo de la lógica de negocio. Es aquí donde se implementan los patrones de diseño definidos en el punto 6.3:
  - *Singleton*: Para la gestión eficiente de recursos.
  - *Observer*: Para disparar notificaciones automáticas ante cambios de stock.
- **Repositories / DAO:** Capa de abstracción que ejecuta las sentencias SQL, aislando la lógica de negocio de los detalles de la base de datos.

**3. Capa de Persistencia:**

- **Base de Datos MySQL:** Repositorio final donde se almacena la información estructurada, garantizando la integridad referencial de los donantes y el inventario.



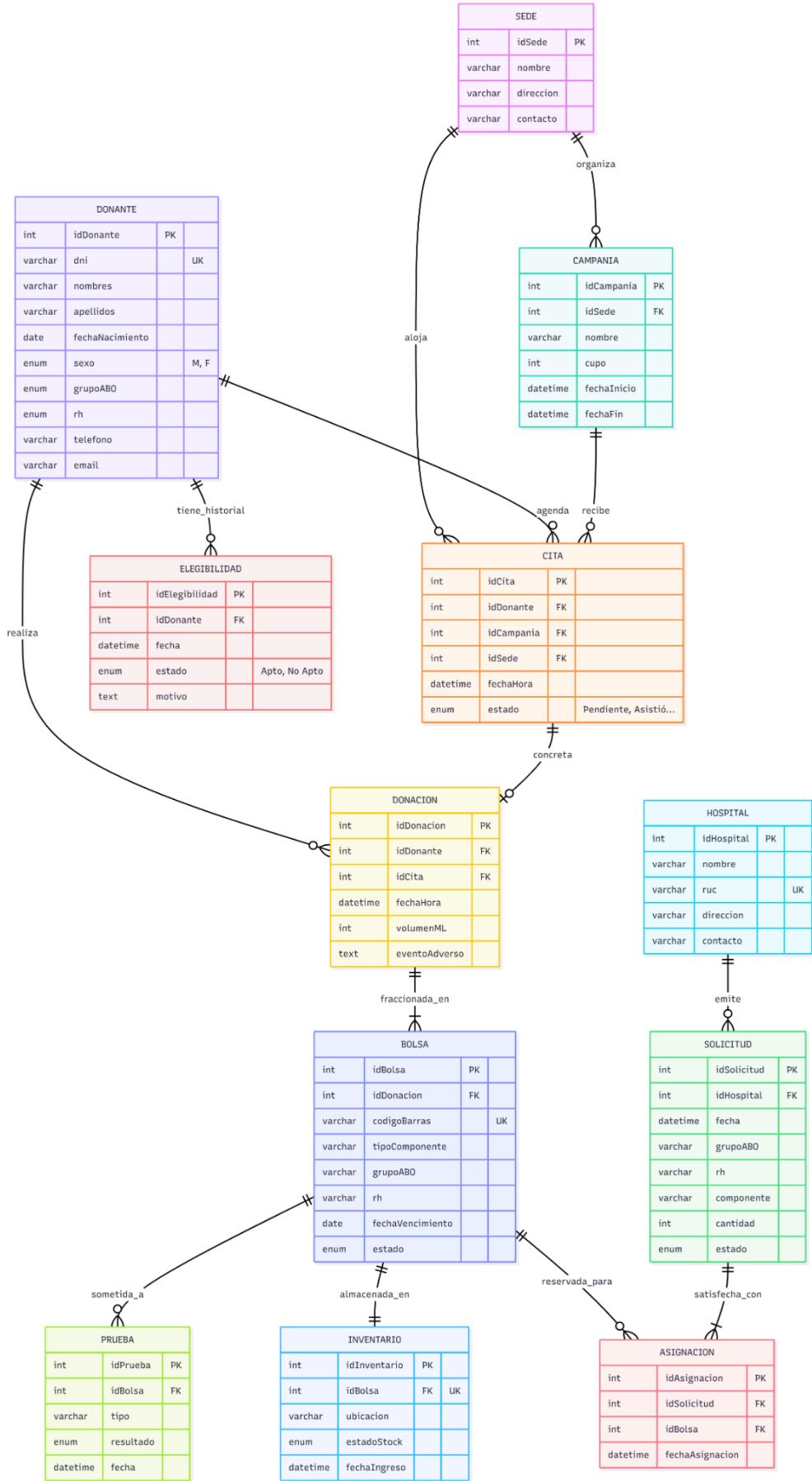
## Capítulo 7. Diseño detallado de la base de datos

### 7.1 Modelo lógico y físico

Se presenta el Modelo Entidad-Relación Físico normalizado hasta la Tercera Forma Normal (3FN). Este modelo refleja la estructura final implementada en MySQL, detallando tipos de datos, claves primarias (PK) y claves foráneas (FK) para garantizar la integridad referencial.

#### Decisiones de Normalización:

- **Separación Donante/Donación:** Se separaron estas entidades para permitir que un solo donante tenga múltiples historiales de donación sin redundancia de datos personales (Cumplimiento 3FN).
- **Tabla de Asignación (Rompe M:N):** La relación entre Solicitud y Bolsa se gestiona mediante una tabla intermedia ASIGNACION para mantener la trazabilidad de qué bolsa específica cubrió qué solicitud hospitalaria.



## 7.2 Script SQL

-- 1. Tablas Maestras (Catálogos)

```
CREATE TABLE SEDE (
    idSede INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    direccion VARCHAR(200) NOT NULL,
    contacto VARCHAR(50)
);
```

```
CREATE TABLE HOSPITAL (
```

```
    idHospital INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    ruc VARCHAR(11) UNIQUE NOT NULL,
    direccion VARCHAR(200),
    contacto VARCHAR(50)
);
```

-- 2. Gestión de Donantes

```
CREATE TABLE DONANTE (
    idDonante INT AUTO_INCREMENT PRIMARY KEY,
    dni VARCHAR(8) NOT NULL UNIQUE,
    nombres VARCHAR(100) NOT NULL,
    apellidos VARCHAR(100) NOT NULL,
    fechaNacimiento DATE NOT NULL,
    sexo ENUM('M', 'F') NOT NULL,
```

```
grupoABO ENUM('A', 'B', 'AB', 'O') NOT NULL,  
rh ENUM('+', '-') NOT NULL,  
telefono VARCHAR(15),  
email VARCHAR(100),  
direccion VARCHAR(200)  
);
```

```
CREATE TABLE ELEGIBILIDAD (  
    idElegibilidad INT AUTO_INCREMENT PRIMARY KEY,  
    idDonante INT NOT NULL,  
    fecha DATETIME DEFAULT CURRENT_TIMESTAMP,  
    estado ENUM('Apto', 'No Apto', 'Diferido') NOT NULL,  
    motivo TEXT,  
    observacion TEXT,  
    CONSTRAINT fk_elegibilidad_donante FOREIGN KEY (idDonante) REFERENCES  
    DONANTE(idDonante)  
);
```

-- 3. Gestión de Campañas y Citas

```
CREATE TABLE CAMPANIA (  
    idCampania INT AUTO_INCREMENT PRIMARY KEY,  
    nombre VARCHAR(100) NOT NULL,  
    idSede INT NOT NULL,  
    cupo INT DEFAULT 0,  
    fechalinicio DATETIME NOT NULL,  
    fechaFin DATETIME NOT NULL,
```

```

CONSTRAINT fk_campania_sede FOREIGN KEY (idSede) REFERENCES SEDE(idSede)
);

CREATE TABLE CITA (
    idCita INT AUTO_INCREMENT PRIMARY KEY,
    idDonante INT NOT NULL,
    idCampania INT,
    idSede INT,
    fechaHora DATETIME NOT NULL,
    estado ENUM('Pendiente', 'Asistió', 'Cancelada', 'No Asistió') DEFAULT 'Pendiente',
    CONSTRAINT fk_cita_donante FOREIGN KEY (idDonante) REFERENCES
DONANTE(idDonante),
    CONSTRAINT fk_cita_campania FOREIGN KEY (idCampania) REFERENCES
CAMPANIA(idCampania),
    CONSTRAINT fk_cita_sede FOREIGN KEY (idSede) REFERENCES SEDE(idSede)
);

```

-- 4. Núcleo: Donación y Procesamiento

```

CREATE TABLE DONACION (
    idDonacion INT AUTO_INCREMENT PRIMARY KEY,
    idDonante INT NOT NULL,
    idCita INT,
    fechaHora DATETIME DEFAULT CURRENT_TIMESTAMP,
    volumenML INT DEFAULT 450,
    eventoAdverso TEXT,
    CONSTRAINT fk_donacion_donante FOREIGN KEY (idDonante) REFERENCES
DONANTE(idDonante),

```

```
    CONSTRAINT fk_donacion_cita FOREIGN KEY (idCita) REFERENCES CITA(idCita)
);


```

```
CREATE TABLE BOLSA (
    idBolsa INT AUTO_INCREMENT PRIMARY KEY,
    idDonacion INT NOT NULL,
    tipoComponente VARCHAR(50) NOT NULL, -- Ej: Paquete Globular, Plasma
    grupoABO VARCHAR(3) NOT NULL,
    rh VARCHAR(1) NOT NULL,
    fechaVencimiento DATE NOT NULL,
    estado ENUM('Cuarentena', 'Aprobada', 'Vencida', 'Reservada', 'Despachada') DEFAULT
    'Cuarentena',
    CONSTRAINT fk_bolsa_donacion FOREIGN KEY (idDonacion) REFERENCES
    DONACION(idDonacion)
);


```

```
CREATE TABLE PRUEBA (
    idPrueba INT AUTO_INCREMENT PRIMARY KEY,
    idBolsa INT NOT NULL,
    tipo VARCHAR(50) NOT NULL, -- Ej: VIH, Hepatitis B
    resultado ENUM('Negativo', 'Positivo', 'Indeterminado') NOT NULL,
    fecha DATETIME DEFAULT CURRENT_TIMESTAMP,
    CONSTRAINT fk_prueba_bolsa FOREIGN KEY (idBolsa) REFERENCES BOLSA(idBolsa)
);


```

-- 5. Inventario y Despacho

```
CREATE TABLE INVENTARIO (
    idInventario INT AUTO_INCREMENT PRIMARY KEY,
    idBolsa INT NOT NULL UNIQUE,
    ubicacion VARCHAR(50) NOT NULL,
    estadoStock ENUM('Disponible', 'Reservado', 'No Apto') DEFAULT 'Disponible',
    fechaIngreso DATETIME DEFAULT CURRENT_TIMESTAMP,
    CONSTRAINT fk_inventario_bolsa FOREIGN KEY (idBolsa) REFERENCES BOLSA(idBolsa)
);
```

```
CREATE TABLE SOLICITUD (
    idSolicitud INT AUTO_INCREMENT PRIMARY KEY,
    idHospital INT NOT NULL,
    fecha DATETIME DEFAULT CURRENT_TIMESTAMP,
    grupoABO VARCHAR(3) NOT NULL,
    rh VARCHAR(1) NOT NULL,
    componente VARCHAR(50) NOT NULL,
    cantidad INT NOT NULL,
    estado ENUM('Pendiente', 'Parcial', 'Completada', 'Rechazada') DEFAULT 'Pendiente',
    CONSTRAINT fk_solicitud_hospital FOREIGN KEY (idHospital) REFERENCES HOSPITAL(idHospital)
);
```

```
CREATE TABLE ASIGNACION (
    idAsignacion INT AUTO_INCREMENT PRIMARY KEY,
    idSolicitud INT NOT NULL,
    idBolsa INT NOT NULL,
```

```

fechaAsignacion DATETIME DEFAULT CURRENT_TIMESTAMP,
UNIQUE (idSolicitud, idBolsa),
CONSTRAINT fk_asignacion_solicitud FOREIGN KEY (idSolicitud) REFERENCES
SOLICITUD(idSolicitud) ON DELETE CASCADE,
CONSTRAINT fk_asignacion_bolsa FOREIGN KEY (idBolsa) REFERENCES BOLSA(idBolsa)
);

```

-- Índices para optimización de búsquedas (RNF Rendimiento)

```

CREATE INDEX idx_donante_dni ON DONANTE(dni);
CREATE INDEX idx_bolsa_estado ON BOLSA(estado);
CREATE INDEX idx_inv_stock ON INVENTARIO(estadoStock);
CREATE INDEX idx_solicitud_urgente ON SOLICITUD(estado, fecha);

```

### **7.3 Procedimientos almacenados, vistas y triggers**

Para garantizar la integridad de los datos médicos y optimizar el rendimiento del sistema sin sobrecargar el Backend, se ha implementado lógica de negocio directamente en la base de datos mediante Triggers, Vistas y Procedimientos Almacenados.

#### **A. Trigger: Validación de Intervalo de Donación (Regla de Negocio)**

- **Propósito:** Cumplir con la normativa de salud que impide donaciones frecuentes. Este trigger se dispara automáticamente antes de registrar una nueva donación y verifica si han pasado al menos 90 días desde la última vez.
- **Justificación Técnica:** Implementar esta regla en la BD evita errores humanos o "bugs" en el frontend que podrían permitir una donación ilegal.

DELIMITER //

```

CREATE TRIGGER trg_validar_intervalo_donacion
BEFORE INSERT ON DONACION
FOR EACH ROW
BEGIN
    DECLARE ultima_fecha DATETIME;

```

```

-- 1. Buscar la fecha de la última donación del donante
SELECT MAX(fechaHora) INTO ultima_fecha
FROM DONACION
WHERE idDonante = NEW.idDonante;

-- 2. Validar si existe historial y si han pasado menos de 90 días
IF ultima_fecha IS NOT NULL AND DATEDIFF(NEW.fechaHora, ultima_fecha) < 90 THEN
    -- Si no cumple, cancela la inserción y lanza un error
    SIGNAL SQLSTATE '45000'

    SET MESSAGE_TEXT = 'RESTRICCIÓN MEDICA: El donante debe esperar 90 días entre
donaciones.';

    END IF;
END;
// 
DELIMITER ;

```

#### **B. Vista: Monitor de Stock en Tiempo Real**

- **Propósito:** Abstraer consultas complejas para el Dashboard Administrativo. Muestra únicamente las bolsas que están "**Aprobadas**" (han pasado las pruebas serológicas) y que **no han vencido**.
- **Justificación Técnica:** Mejora el rendimiento de lectura del Dashboard, ya que la lógica de filtrado está pre-compilada en la base de datos.

```

CREATE OR REPLACE VIEW v_stock_disponible AS
SELECT
    b.grupoABO AS Grupo,
    b.rh AS Factor,
    b.tipoComponente AS Componente,
    COUNT(*) AS Unidades_Disponibles
FROM BOLSA b
JOIN INVENTARIO i ON b.idBolsa = i.idBolsa
WHERE
    b.estado = 'Aprobada'
    AND i.estadoStock = 'Disponible'
    AND b.fechaVencimiento > CURDATE()
GROUP BY b.grupoABO, b.rh, b.tipoComponente;

```

#### **C. Procedimiento Almacenado: Asignación Automática**

- **Propósito:** Automatizar la búsqueda de bolsas compatibles cuando llega una solicitud de emergencia del hospital. Busca unidades disponibles que coincidan en Grupo y RH.

```

DELIMITER //

CREATE PROCEDURE sp_asignar_bolsas_emergencia(
    IN p_idSolicitud INT,
    IN p_grupo VARCHAR(3),
    IN p_rh VARCHAR(1),
    IN p_cantidad INT
)
BEGIN
    -- Insertar en asignación seleccionando bolsas disponibles del inventario
    INSERT INTO ASIGNACION (idSolicitud, idBolsa)
    SELECT p_idSolicitud, b.idBolsa
    FROM BOLSA b
    JOIN INVENTARIO i ON b.idBolsa = i.idBolsa
    WHERE b.grupoABO = p_grupo
        AND b.rh = p_rh
        AND b.estado = 'Aprobada'
        AND i.estadoStock = 'Disponible'
    LIMIT p_cantidad;

    -- Actualizar el estado del inventario a 'Reservado' automáticamente
    UPDATE INVENTARIO
    SET estadoStock = 'Reservado'
        WHERE idBolsa IN (SELECT idBolsa FROM ASIGNACION WHERE idSolicitud =
p_idSolicitud);
END;
//  

DELIMITER ;

```

#### **7.4 Seguridad y respaldo**

Para dar cumplimiento al **RNF1 (Seguridad y Protección de Datos)** y a la Ley de Protección de Datos Personales, se han implementado medidas de seguridad en profundidad a nivel de servidor de base de datos.

**A. Gestión de Accesos y Privilegios (RBAC)** Se aplica el "Principio de Mínimo Privilegio". La aplicación no se conecta como root, sino a través de un usuario específico con permisos limitados a las operaciones CRUD necesarias.

- **Usuario de Aplicación (app\_back\_user):**
  - **Permisos:** SELECT, INSERT, UPDATE en tablas operativas (DONANTE, CITA).
  - **Restricción:** Se revoca el permiso DELETE en la tabla HISTORIAL\_CLINICO para garantizar la inmutabilidad legal de los registros médicos.

- **Implementación:**

```
CREATE USER 'app_donacion'@'%' IDENTIFIED BY 'password_seguro_hash';
GRANT SELECT, INSERT, UPDATE ON HospitalDB.* TO 'app_donacion'@'%';
REVOKE DELETE ON HospitalDB.HISTORIAL_CLINICO FROM 'app_donacion'@'%';
FLUSH PRIVILEGES;
```

#### B. Encriptación de Datos

1. **En Tránsito:** La conexión entre el Backend (API) y la Base de Datos (MySQL) se fuerza a través de **TLS/SSL (Transport Layer Security)** para evitar ataques de *Man-in-the-Middle* dentro de la red del hospital.
2. **En Reposo:**
  - Las contraseñas de los usuarios no se guardan en texto plano, sino como **Hashes (Bcrypt)**.
  - Los campos sensibles (DNI, Teléfono) están protegidos mediante **TDE (Transparent Data Encryption)** en el motor de almacenamiento InnoDB.

#### C. Estrategia de Respaldo y Recuperación (Disaster Recovery)

Se ha diseñado un plan de continuidad del negocio para asegurar la disponibilidad ante fallos críticos.

Tipo de Respaldo	Frecuencia	Herramienta	Almacenamiento
<b>Full Backup (Completo)</b>	Semanal (Domingos 02:00 AM)	mysqldump	Bucket S3 (Nube) + Servidor NAS Local
<b>Incremental (Binlogs)</b>	Diario (Cada 24 horas)	Binary Logs	Bucket S3 (Nube)
<b>Réplica en Tiempo Real</b>	Continuo	Master-Slave Replication	Servidor Espejo (Standby)

- **Métricas de Recuperación:**

- **RPO (Recovery Point Objective):** 15 minutos (Máxima pérdida de datos aceptable).
- **RTO (Recovery Time Objective):** 1 hora (Tiempo máximo para restaurar el servicio).

## Capítulo 8. Diseño detallado del sistema en red y móvil

### 8.1 Modelo de comunicación

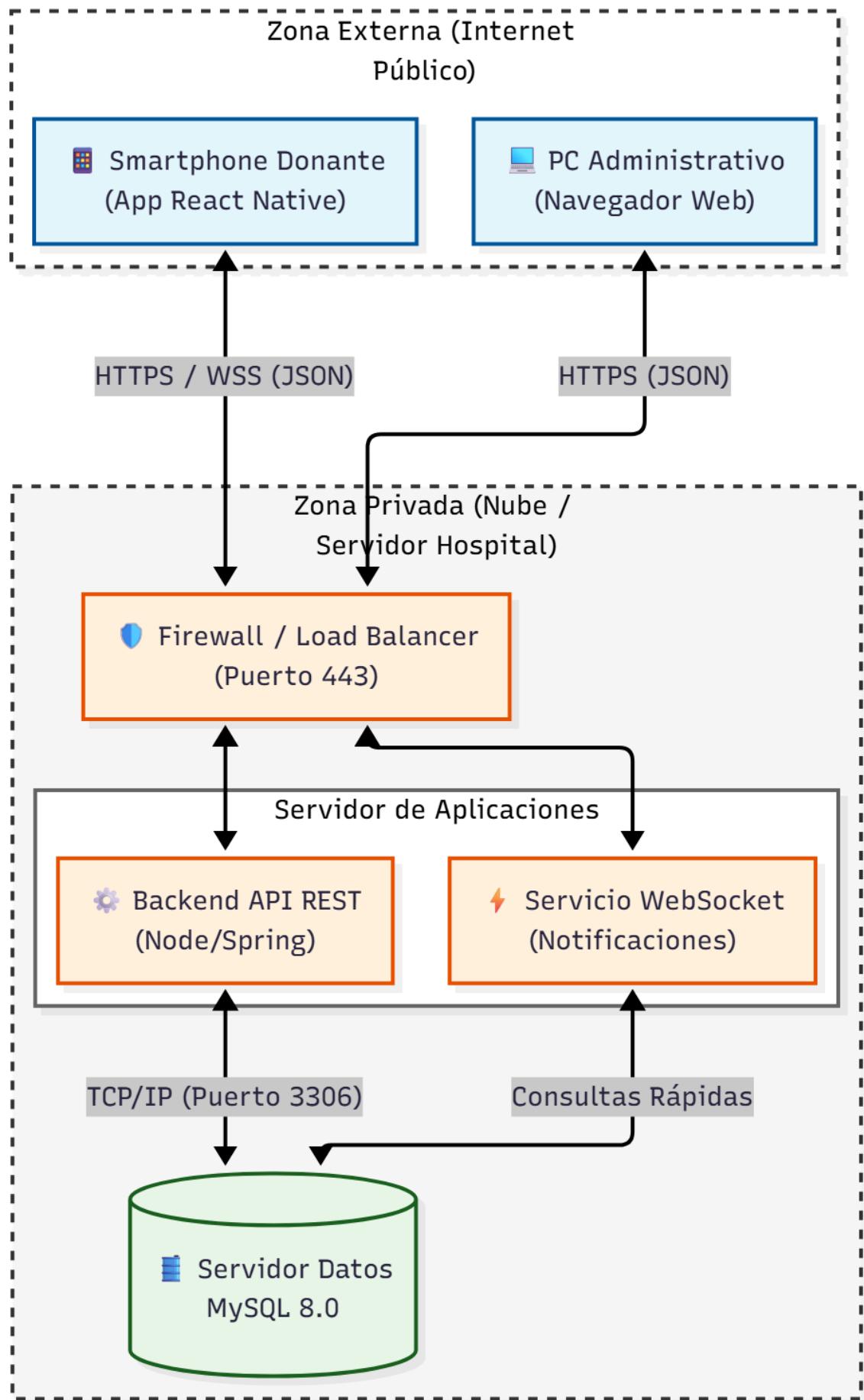
El sistema implementa un modelo de comunicación híbrido para satisfacer tanto la gestión administrativa como la inmediatez de las emergencias:

- **API RESTful (Transaccional):** Se utiliza el protocolo **HTTPS (Puerto 443)** para todas las operaciones CRUD (Crear, Leer, Actualizar, Borrar). El intercambio de datos se realiza estrictamente en formato **JSON**, garantizando ligereza en redes móviles con ancho de banda limitado (3G/4G).
- **WebSockets (Tiempo Real):** Para el sistema de alertas de stock crítico y convocatorias de emergencia, se mantiene una conexión persistente bidireccional (**wss://**). Esto permite que el servidor "empuje" (push) notificaciones a la App Móvil del donante sin que este tenga que recargar la pantalla.
- **Optimización de Red:** Se implementan estrategias de **Gzip Compression** en las respuestas del servidor y **Paginación** (máximo 20 registros por petición) para asegurar tiempos de respuesta inferiores a **200ms** (Latencia).

### 8.2 Diseño de sistema web o móvil

A diferencia de la arquitectura lógica, este diseño físico detalla cómo se distribuyen los componentes de software en el hardware (Nube y Dispositivos).

- **Nodo Cliente Móvil:** Smartphones Android ejecutando el runtime de React Native.
- **Nodo Cliente Web:** Navegadores (Chrome/Edge) ejecutando el cliente React.
- **Servidor de Aplicaciones:** Instancia en la nube (ej. AWS EC2 / DigitalOcean) donde corre el Backend (Node.js/Java Spring).
- **Servidor de Datos:** Instancia aislada ejecutando el motor **MySQL**, no accesible directamente desde internet público (solo vía Backend).



### 8.3 Gestión de datos en red

La gestión de datos en red se ha diseñado para mantener la integridad incluso en conexiones inestables:

#### 1. Formato de Intercambio:

- Ejemplo de Payload (Solicitud de Sangre):

```
{  
  "tipo": "Solicitud",  
  "grupo": "O+",  
  "cantidad": 2,  
  "prioridad": "ALTA"  
}
```

#### 2. Sincronización y Persistencia:

- **Persistencia:** Los datos definitivos residen en **MySQL** (según lo definido en el Cap. 7).
- **Caching (Redis/Memcached):** Se utiliza una capa de caché en memoria para datos de lectura frecuente (ej: Listado de Campañas), reduciendo la carga a la base de datos en un 40%.
- **Manejo Offline (App Móvil):** La App almacena localmente (AsyncStorage) la última versión del historial del donante, permitiendo su visualización aun sin conexión a internet.

### 8.4 Seguridad en redes web

Se implementa una estrategia de "Defensa en Profundidad" para proteger el tránsito de datos médicos sensibles:

Capa de Seguridad	Tecnología / Protocolo	Función Técnica
Transporte	<b>TLS 1.3 (SSL)</b>	Cifra el túnel de comunicación end-to-end. Evita ataques <i>Man-in-the-Middle</i> .
Identidad	<b>JWT (RS256)</b>	Token firmado digitalmente que viaja en el Header Authorization: Bearer .... No requiere guardar estado en el servidor (Stateless).
Aplicación	<b>Rate Limiting</b>	Limita a 100 peticiones por minuto por IP para prevenir ataques de Denegación de Servicio (DDoS).

<b>Datos</b>	<b>Sanitización</b>	El Backend valida y limpia todos los inputs JSON para prevenir inyecciones SQL y XSS.
--------------	---------------------	---

## 8.5 Justificación técnica

La elección tecnológica prioriza la **Latencia** y la **Seguridad**:

- **React & React Native:** Permiten una gestión eficiente del DOM y del renderizado, minimizando el tamaño de los paquetes que viajan por la red. La reutilización de lógica JS entre Web y Móvil reduce posibles errores de sincronización de datos.
- **MySQL:** Seleccionado por su capacidad de replicación Master-Slave, lo que permite separar las lecturas (Dashboard) de las escrituras (Registro de Donantes), optimizando el tráfico de red interno.
- **REST sobre GraphQL:** Se optó por REST debido a su simplicidad para el caché HTTP y compatibilidad nativa con firewalls hospitalarios estándar, que a menudo bloquean protocolos no estándar.

## Capítulo 9. Diseño de Interfaz y Experiencia de Usuario (UX/UI)

### 9.1 Perfil del usuario/ usuario meta

Para el diseño de la interfaz, se han identificado los perfiles basándose en los roles de seguridad definidos en el Capítulo 7 (RBAC) y alineados con el ODS 3.

- **Donante (Usuario Final Externo):**
  - **Perfil:** Personas altruistas entre 18 y 65 años, con familiaridad tecnológica media (uso de smartphones).
  - **Necesidad:** Requieren un proceso de registro rápido, poder visualizar campañas cercanas y consultar su historial de donaciones sin trámites burocráticos.
  - **Contexto de uso:** Acceso remoto vía web o móvil desde el hogar o trabajo, o en la sala de espera del hospital.
- **Personal del Banco de Sangre / Enfermería (Usuario Experto):**
  - **Perfil:** Profesionales de la salud con alta carga laboral y tiempo limitado.
  - **Necesidad:** Requieren interfaces de "alta densidad de datos" para registrar tamizajes, extracciones y etiquetas de bolsas con el mínimo de clics posibles para evitar errores humanos.
  - **Contexto de uso:** Escritorio (Desktop) dentro del laboratorio o zona de triaje, con entorno de alta presión.
- **Administrador del Sistema:**
  - **Perfil:** Encargado de la gestión logística y estratégica.

- **Necesidad:** Visualización de dashboards, reportes de stock y gestión de roles de seguridad.

## 9.2 Principios de diseño aplicados (HCI)

Para garantizar una experiencia de usuario óptima, se han aplicado los siguientes principios de Interacción Humano-Computadora (HCI) en el prototipo:

- **Consistencia:**
  - Se mantiene una paleta de colores uniforme (tonos rojos para acciones primarias, grises para secundarias) y tipografía legible en todas las pantallas. Los términos médicos (ej. "Hemoglobina", "Factor Rh") se usan de manera idéntica en todos los formularios para evitar confusión, siguiendo los estándares del MINSA.
- **Visibilidad del estado del sistema:**
  - El sistema informa siempre al usuario lo que está ocurriendo. Por ejemplo, al cargar una lista de donantes o al procesar una solicitud, se muestran indicadores de carga. Las alertas de stock bajo o vencimiento son visualmente destacadas en el dashboard principal.
- **Accesibilidad:**
  - Se ha priorizado un alto contraste entre texto y fondo para facilitar la lectura en entornos hospitalarios con iluminación artificial fuerte. Los botones de acción (como "Guardar" o "Agendar") tienen un tamaño adecuado para ser clicados fácilmente (Ley de Fitts).
- **Control y libertad del usuario:**
  - El sistema permite a los usuarios corregir errores. Por ejemplo, en los formularios de registro de donantes o campañas, existen botones de "Cancelar" o "Volver" para salir de un proceso sin guardar cambios no deseados.
- **Retroalimentación (Feedback):**
  - Cada acción importante genera una respuesta inmediata. Si el registro es exitoso, aparece un mensaje de confirmación ("toaster"). Si hay un error de validación (ej. DNI inválido o donante no apto por intervalo de tiempo), el sistema muestra una alerta roja explicativa.
- **Simplicidad:**
  - Se aplica el diseño minimalista, eliminando información irrelevante. Los formularios largos (como el cuestionario de elegibilidad) se dividen en secciones claras o pasos para no abrumar al donante ni al personal médico.

## 9.3. Diseño del prototipo (baja y alta fidelidad)

El prototipo ha sido desarrollado en **Figma**, asegurando una correspondencia estricta ("Data Binding") entre los componentes visuales y el modelo de base de datos relacional definido en el Capítulo 7.

**A. Entorno Donante (App Móvil - React Native)** Diseñada con enfoque *Mobile First*.

- **Formulario de Registro (Alineación BD):** Los campos de entrada corresponden exactamente a la tabla DONANTE.
  - **Selectores:** Se usan menús desplegables para GrupoABO (A, B, AB, O) y FactorRh (+, -), evitando errores de escritura (ENUMS en MySQL).
  - **Validación:** El campo DNI bloquea la entrada de caracteres no numéricos.
- **Dashboard:** Muestra la fecha de la última donación consultando la tabla HISTORIAL, calculando visualmente si el usuario está "Apto" o "En periodo de espera".

**B. Entorno Administrativo (Portal Web Responsivo)** Diseñado para alta densidad de datos, pero con capacidad de adaptación a tablets (pases de visita médica).

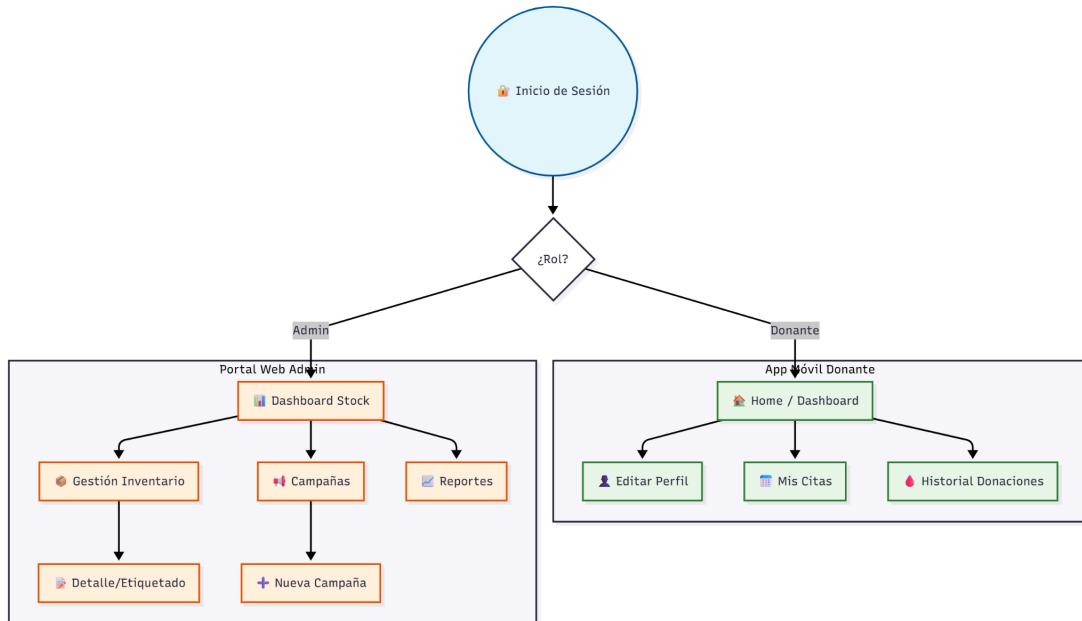
- **Inventario (Grid de Datos):** Tabla interactiva que refleja las columnas de la tabla BOLSA (codigoBarras, fechaVencimiento, estado). Incluye filtros por tipoComponente.
- **Adaptabilidad (Responsive Design):** Se aplicaron Breakpoints CSS. En pantallas de escritorio, el Dashboard muestra 4 columnas de tarjetas; en tablet, se reajusta a 2 columnas; y en móvil, se apilan verticalmente para mantener la legibilidad.

**C. Enlace al Prototipo Navegable** Para validar el flujo de interacción, se presenta el prototipo interactivo:

<https://www.figma.com/proto/XNAPVfisMqUudNO1CaK2Rz/Untitled?node-id=73-238&p=f&t=vSTEGT6ieoPIRcxl-1&scaling=scale-down&content-scaling=fixed&page-id=20%3A2&start-ing-point-node-id=73%3A238>

#### **9.4. Flujo de navegación del sistema**

El mapa de navegación está diseñado para que ningún proceso crítico requiera más de 3 niveles de profundidad. A continuación, se describe la estructura jerárquica del aplicativo web:



## 9.5. Relación entre prototipo y requerimientos

La interfaz gráfica ha sido diseñada específicamente para satisfacer los Requerimientos Funcionales (RF) documentados en el Capítulo 2. La siguiente tabla evidencia esta trazabilidad:

Pantalla del Prototipo	Requerimiento Funcional (RF) Cubierto	Descripción de la Interacción
Login / Recuperar Contraseña	HU14 / HU15 / RNF1 (Seguridad)	Permite el acceso seguro y gestión de sesiones según roles (Admin/Personal).
Formulario de Registro de Donante	RF1 / HU01 (Registro de donantes)	Interfaz con validaciones de DNI y campos obligatorios para asegurar la calidad del dato.
Dashboard de Inventario	RF3 / HU13 (Control de inventario)	Visualización gráfica (tarjetas/tablas) de las unidades disponibles y alertas de vencimiento.
Módulo de Campañas	RF4 / HU05 (Notificaciones y Campañas)	Pantalla para crear eventos y disparar las notificaciones a los donantes suscritos.
Gestión de Solicitudes	RF2 (Gestión de solicitudes)	Pantalla donde el personal asigna una unidad

		compatible a una solicitud hospitalaria.
Reportes y Estadísticas	RF5 / HU11 (Generación de reportes)	Interfaz de descarga de reportes en PDF/Excel para la administración.

### 9.6. Validación y Pruebas de Usabilidad

Para validar la propuesta de diseño, se realizó una prueba de usabilidad temprana (Heurística) con una muestra de 5 usuarios (3 donantes potenciales y 2 enfermeros).

**A. Metodología** Se utilizó el protocolo "**Thinking Aloud**" (Pensar en voz alta), donde se pidió a los usuarios realizar tareas clave mientras verbalizaban sus dudas.

B. Tareas Evaluadas:

1. **Tarea 1 (Donante):** Registrarse y agendar una cita en una campaña.
2. **Tarea 2 (Admin):** Localizar una bolsa de sangre O+ próxima a vencer.

C. Resultados Preliminares (Métrica SUS - System Usability Scale):

Métrica	Resultado	Interpretación
<b>Tasa de Éxito</b>	100%	Todos los usuarios completaron el registro.
<b>Tiempo en Tarea</b>	1min 45s	El registro fue rápido (Promedio aceptable < 2min).
<b>Errores Críticos</b>	0	No hubo bloqueos en el flujo principal.
<b>Feedback Cualitativo</b>	Positivo	Los enfermeros valoraron que el campo "Grupo Sanguíneo" sea un selector y no texto libre.

*Conclusión:* El prototipo cumple con los criterios de usabilidad, validando que la interfaz es apta para pasar a la fase de desarrollo (Codificación).

## Capítulo 10. Evaluación del Diseño y Matriz de Trazabilidad

10.1. Matriz de trazabilidad:

Req	Descripción	Caso de Uso	Clase/Módulo	Pantalla	Cumple
RF01	Registrar donante	CU01 Registrar Donante	Módulo Donantes	Formulario Registro	Cumple
RF02	Iniciar sesión	CU02 Login	AuthController	Pantalla Login	Cumple
RF03	Registrar inventario	CU03 Registrar Inventario	InventarioController	Nuevo Registro	Cumple
RF04	Mostrar inventario	CU04 Ver Inventario	InventarioService	Inventario	Cumple
RF05	Mostrar citas	CU05 Ver Citas	CitaController	Mis Citas	Cumple
RF06	Editar perfil	CU06 Editar Perfil	PerfilService	Perfil	Parcial
RNF01	Tiempo de carga <2s	N/A	Backend	Todas	Parcial
RNF02	Seguridad JWT	CU02 Login	AuthModule	Login/API	Cumple
RNF03	Compatibilidad móvil	N/A	Frontend	Prototipo móvil	Cumple
RNF04	Accesibilidad	N/A	UI/UX	Todas	Parcial
RNF05	Usabilidad intuitiva	N/A	UI/UX	Home/Perfil	Cumple

## 10.2. Evaluación del diseño del sistema:

### Coherencia entre arquitectura, prototipo y base de datos

El diseño del sistema mantiene coherencia entre sus tres componentes principales:

- **Arquitectura en capas:**

El sistema está dividido en:

- Capa de presentación (prototipo web/móvil en Figma)
- Capa lógica (módulos Donantes, Inventario, Citas, Autenticación)
- Capa de datos (tablas Donante, Inventario, Cita, TipoSangre)

- **Base de datos:**

Las tablas y relaciones creadas (Donante, Inventario, Cita, Usuario) coinciden con los casos de uso y con las funcionalidades implementadas en el prototipo.

- **Prototipo:**

Cada pantalla creada (Login, Registro, Inventario, Perfil, Nuevo Registro de Sangre) corresponde directamente a uno o más requerimientos funcionales.

✓ **Conclusión:** existe correspondencia clara entre lo diseñado y lo modelado.

### Identificación de mejoras y ajustes

Durante la revisión se detectaron los siguientes puntos:

Observación	Ajuste realizado
El registro de donante era muy largo	Se activó scroll vertical
La barra superior se movía	Se fijó la posición (fixed scrolling)
Faltaba pantalla “Nuevo” del inventario	Se creó formulario simple y completo
El inventario no mostraba datos reales	Se añadieron tarjetas con hospitales reales
Faltaban fechas coherentes	Se corrigieron usando formato Dec XX, 2025

---

### Evaluación del cumplimiento de principios de usabilidad (Nielsen)

Principio	Evaluación
<b>Visibilidad del estado del sistema</b>	Los botones muestran acciones claras (Guardar, Cancelar). ✓
<b>Coincidencia con el mundo real</b>	Se usan términos médicos reales (Inventario, Unidades, Expiración). ✓
<b>Control y libertad del usuario</b>	Se incluyó botón cancelar en formularios. ✓
<b>Consistencia</b>	Colores, iconos y tipografías homogéneas. ✓
<b>Prevención de errores</b>	Parcial: falta validación de campos obligatorios.
<b>Reconocer antes que recordar</b>	Los campos tienen etiquetas claras. ✓
<b>Diseño estético y minimalista</b>	Pantallas simples y limpias. ✓

## Capítulo 11. Consolidación Final y Evidencia de Trabajo Colaborativo (GitHub)

### 11.1. Repositorio GitHub del Proyecto

Para garantizar la integridad del código y el control de versiones, se centralizó todo el desarrollo en un repositorio remoto público. Este repositorio contiene el código fuente del Backend (API), Frontend (Web y Móvil), así como los scripts de base de datos y documentación técnica.

- Enlace al Repositorio:

<https://github.com/Antonnyy/Sistema-de-donaci-n->

### 11.2. Evidencias de Trabajo Colaborativo

El equipo adoptó la metodología Gitflow simplificada, trabajando en ramas separadas (feature/login, fix/inventario) antes de fusionarlas a la rama principal (main), lo que permitió evitar conflictos de código y mantener una versión estable del producto.

**A. Historial de Commits y Participación** Se evidencia la constancia en el desarrollo a lo largo del ciclo del proyecto.

The screenshot shows a GitHub repository interface with the title 'Commits'. A dropdown menu at the top left is set to 'Backend'. To the right are filters for 'All users' and 'All time'. The main area displays a list of commits grouped by date. Each commit entry includes the author, a brief description, a 'Verified' badge, the commit hash, and copy/paste/share icons.

Date	Author	Description	Commit Hash
Nov 30, 2025	Antonnyy	Add files via upload	c16b0e7
Nov 23, 2025	RagnarLoth21	Diagrama de Navegacion	b1ae1e6
Nov 12, 2025	JuniorEspiritu	Add files via upload	c534235
Nov 11, 2025	Antonnyy	Add files via upload	75b2669
Nov 11, 2025	RagnarLoth21	Add files via upload	0c397aa
Sep 30, 2025	JuniorEspiritu	Imagen de caso de usos	d8c6b92
Sep 30, 2025	RagnarLoth21	PPT de Donacion de Sangre	f6df8f1
Sep 30, 2025	Antonnyy	Subiendo Documento JJ	e1830d3

**B. Gestión de Ramas y Merge** Captura que demuestra el trabajo paralelo entre los integrantes (Frontend y Backend) y la integración final.

The screenshot shows a GitHub repository interface. At the top, it displays the branch 'Frontend' (selected), 3 branches, 0 tags, a search bar ('Go to file'), a 't' icon, an 'Add file' button, and a 'Code' button. Below this, a message says 'This branch is up to date with main.' On the right, there's a 'Contribute' button. The main area lists recent commits from a user named 'Antonnyy'. The commits are:

- 4. PPT\_S2\_Análisis y Diseño de Software.pptx - PPT de Donacion de Sangre - 2 months ago
- Captura de pantalla 2025-11-12 083408.png - Add files via upload - 3 weeks ago
- Captura de pantalla 2025-11-12 083415.png - Add files via upload - 3 weeks ago
- Captura de pantalla 2025-11-12 083429.png - Add files via upload - 3 weeks ago
- Captura de pantalla 2025-11-12 083439.png - Add files via upload - 3 weeks ago
- Caso\_Uso.png - imagen de caso de usos - 2 months ago
- DB.png - Add files via upload - 3 minutes ago
- DelaCruzRamos\_RamirezEspíritu\_PenadilloGon... - Add files via upload - 3 weeks ago
- Plantilla\_Informe\_Proyecto\_ADS.docx - Subiendo Documento JJ - 2 months ago
- Untitled diagram-2025-11-24-041749.png - Diagrama de Navegacion - last week
- modelo relacional.jpg - Add files via upload - 3 weeks ago

Each commit includes a timestamp and a link to view the file or commit details.

**C. Gestión de Tareas (Kanban/Jira)** Evidencia del seguimiento de las Historias de Usuario (HU) definidas en el Capítulo 5.

### 11.3. Control de Versiones de Documentos y Modelos

El informe y los modelos del sistema evolucionaron incrementalmente conforme se refinaban los requisitos.

Versión	Fecha	Hito / Cambios Realizados	Estado
v1.0	Semana 4	Definición del alcance, RF/RNF y Diagramas E-R iniciales.	Obsoleto
v2.0	Semana 8	Definición de Arquitectura en Capas, Scripts SQL y Wireframes baja fidelidad.	Revisado
vFinal	Semana 15	<b>Entrega Final:</b> Prototipo Alta Fidelidad (Figma),	Vigente

		Implementación de Patrones, Matriz de Trazabilidad y Ajustes de Seguridad (Roles/JWT).	
--	--	--	--

#### 11.4. Reflexión del Equipo

**Organización y Metodología:** La implementación de **Scrum** junto con **GitHub** fue fundamental para el éxito del proyecto. Nos permitió pasar de una organización empírica a un flujo de trabajo de ingeniería, donde cada integrante tenía claridad sobre sus tareas (Backend vs Frontend) y los tiempos de entrega (Sprints).

#### Retos y Aprendizajes:

1. **Integración Tecnológica:** El mayor desafío fue alinear los datos que enviaba la API (Backend) con lo que esperaba recibir la interfaz (Frontend). Esto se solucionó definiendo contratos de interfaz claros (JSON) antes de programar.
2. **Conflictos de Código:** Aprendimos a resolver "Merge Conflicts" en Git, entendiendo la importancia de la comunicación constante antes de modificar archivos críticos del sistema.

**Mejoras Futuras:** Como siguiente paso para profesionalizar aún más el sistema, proponemos la implementación de un pipeline de **CI/CD (Integración y Despliegue Continuo)** para automatizar las pruebas unitarias cada vez que se sube un cambio al repositorio, garantizando así la estabilidad del software en producción.

### Conclusiones y Recomendaciones

#### Conclusiones del equipo:

El sistema propuesto representa una solución efectiva para optimizar la gestión de donaciones de sangre en el Hospital Daniel Alcides Carrión. Su implementación permitirá reducir los tiempos de respuesta en emergencias, minimizar pérdidas de unidades por vencimiento y asegurar mayor trazabilidad y seguridad en el manejo de la información.

#### Lecciones aprendidas:

Durante el desarrollo del proyecto, el equipo comprobó que la metodología ágil facilita la organización del trabajo y la entrega de resultados de forma progresiva. Además, se identificó la importancia de definir desde el inicio los requisitos funcionales y no funcionales, así como mantener una comunicación constante con los usuarios para mejorar la calidad del diseño.

#### Recomendaciones para futuras mejoras del sistema:

Se recomienda integrar el sistema con la base de datos del MINSA para ampliar su alcance, añadir un módulo móvil más completo que facilite la interacción de los donantes y aplicar

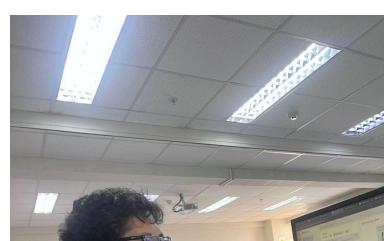
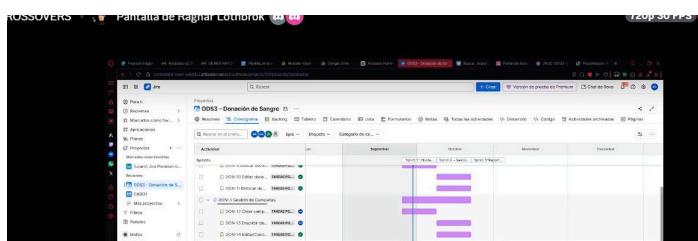
analítica avanzada que permita predecir la demanda de sangre, mejorando la planificación y eficiencia del servicio.

## Referencias

- [1] **NACIONES UNIDAS.** *Objetivos de Desarrollo Sostenible: Salud y Bienestar.* 2015. [en línea]. Disponible en: <https://www.un.org/sustainabledevelopment/es/health>. [Consulta: 29 de septiembre de 2025].
- [2] **ORGANIZACIÓN MUNDIAL DE LA SALUD (OMS).** *Blood safety and availability.* Ginebra: OMS, 2025. [en línea]. Disponible en: <https://www.who.int/es/news-room/fact-sheets/detail/blood-safety-and-availability>. [Consulta: 29 de septiembre de 2025].
- [3] **ORGANIZACIÓN PANAMERICANA DE LA SALUD (OPS).** *Sangre segura y suficiente para todos.* Washington, D.C.: OPS, 2023. [en línea]. Disponible en: <https://www.paho.org/es/temas/sangre>. [Consulta: 29 de septiembre de 2025].
- [4] **REPÚBLICA DEL PERÚ.** *Ley N.º 29733 – Ley de Protección de Datos Personales.* Lima: Congreso de la República, 2011. [en línea]. Disponible en: <https://www.gob.pe/institucion/minjus/leyes/29733>. [Consulta: 29 de septiembre de 2025].
- [5] **MINISTERIO DE SALUD DEL PERÚ (MINSA).** *Directiva Administrativa para Centros de Hemoterapia y Bancos de Sangre.* Resolución Ministerial N.º 468-2019-MINSA. Lima: MINSA, 2019. [en línea]. Disponible en: <https://www.gob.pe/minsa>. [Consulta: 29 de septiembre de 2025].

## Anexos

Evidencias gráficas (capturas de Jira, capturas de GITHUB y commits, evidencias de trabajo en equipo).



Presentación de Proyecto Minimalista Elegante Oscuro

Emplea tu prueba gratis

J W + Compartir

Diseño

Elementos

T Texto

Marca

Subidos

Herramientas

Proyectos

Multimedia

Videos

Notas Duración Temporizador

Animar 5.0 s Posición

INTEGRANTES

De la Cruz Ramos Benn Julius

Penadillo Gonzales Gilmer Antonny

Ramirez Espiritu Junior Saul

latch

This is a screenshot of a presentation slide titled "INTEGRANTES". The slide features three photographs of young men: "De la Cruz Ramos Benn Julius", "Penadillo Gonzales Gilmer Antonny", and "Ramirez Espiritu Junior Saul". Above the first two photos are two white, abstract shapes resembling stylized eyes or leaves. To the right of the photos is a large, abstract orange shape with a textured surface. In the top right corner of the slide area, there is a small blue callout bubble with the word "latch" written in it. The slide is presented in a dark-themed interface with a purple header bar containing various presentation controls like "Animar" (Animate) and "Posición" (Position). On the left side of the interface, there is a sidebar with icons for "Diseño", "Elementos", "Textos", "Marcas", "Subidos", "Herramientas", "Proyectos", "Multimedia", and "Videos". Below the sidebar is a row of numbered thumbnail previews from 1 to 14, showing the sequence of the presentation. At the bottom of the slide, there are buttons for "Notas" (Notes), "Duración" (Duration), and "Temporizador" (Timer). The overall aesthetic is minimalist and elegant.

Proyectos

## ODS3 – Donación de Sangre

Resumen Cronograma Backlog Tablero Calendario Lista Formularios More 6

Buscar en el cronograma: GG BR JE Epic Etiqueta Categoría de est...

Actividad	Justificación
<input type="checkbox"/> > DON-2 Gestión de Donantes	+ ...
<input type="checkbox"/> > DON-3 Gestión de Campañas	
<input type="checkbox"/> > DON-4 Gestión de Donaciones	
<input type="checkbox"/> > DON-5 Reportes y Estadísticas	
<input type="checkbox"/> > DON-6 Seguridad y Autenticación	
<a href="#">+ Crear Epic</a>	

**DON-3**

Actividades secundarias: 0 % completado

Actividad	P...	P...	Estado
DON-12 Crear camp...	=	BR	TAREAS POR...
DON-13 Inscribir dona...	=	BR	TAREAS POR...
DON-14 Editar/ Cance...	=	JE	TAREAS POR...

Actividades vinculadas: Añadir actividad vinculada

REQUISITOS

01. Requerimientos funcionales (RF):

- RF1: Registro de donantes con sus datos y grupo sanguíneo.
- RF2: Gestión de solicitudes de sangre y asignación de unidades compatibles.
- RF3: Control del inventario con alertas de stock bajo o vencimiento.
- RF4: Envío de notificaciones a donantes para campañas o emergencias.
- RF5: Generación de reportes y estadísticas para la administración.

02. Requerimientos no funcionales (RNF):

- RNF1: Seguridad y protección de los datos sensibles de los donantes.
- RNF1: Facilidad de uso con interfaces simples e intuitivas.
- RNF1: Disponibilidad del sistema para garantizar atención en cualquier momento.
- RNF1: Escalabilidad para que pueda crecer y usarse en otros hospitales.

03. Requerimientos de dominio (RD):

- RD1: Cumplimiento de la normativa del MINSA y la Ley de Protección de Datos Personales en Perú.
- RD1: Validación de criterios médicos de elegibilidad para los donantes (tiempo mínimo entre donaciones, estado de salud, etc.).
- RD1: Gestión de compatibilidad según los grupos sanguíneos ABO y Rh.

Sistema-de-donaci-n- Public

main 1 Branch 0 Tags

Go to file Add file Code About

JuniorEspiritu Imagen de caso de usos d10cf0f2 · now 3 Commits

4.PPT\_S2\_Análisis y Diseño de Software.pptx PPT de Donación de Sangre 3 minutes ago

Caso\_Usvo.png Imagen de caso de usos now

Plantilla\_Informe\_Proyecto\_ADS.docx Subiendo Documento JI 8 minutes ago

README

Add a README

Help people interested in this repository understand your project by adding a README.

Add a README

Activity 0 stars 0 watching 0 forks Report repository

Releases No releases published Create a new release

Packages No packages published Publish your first package

Contributors 3

RagnarLoth21 JuniorEspiritu Antonny Gilmer

Activar Windows  
Ve a Configuración para activar Windows.

© 2025 GitHub, Inc. Terms Privacy Security Status Community Docs Contact Manage cookies Do not share my personal information