UDEM

Universidad de Monterrey
Integración de Aplicaciones Computacionales

Heartguard: Sistema Predictivo para Emergencias Cardiovasculares

Reporte Formal

Primera Entrega – Proyecto de Integración de Sistemas Computacionales

5 de septiembre de 2025

PhD. Raúl Morales Salcedo

Equipo 1

Aldo Elio Peña Salas - 635861

Pablo Celedón Cabriales - 597730

Damos nuestra palabra que hemos realizado esta actividad con integridad académica

ÍNDICE

Introducción

Problemática de las Enfermedades Cardiovasculares

Las enfermedades cardiovasculares (ECV) constituyen la principal causa de mortalidad a nivel mundial, representando un desafío crítico para los sistemas de salud contemporáneos. Según datos recientes de los Centros para el Control y Prevención de Enfermedades (CDC), una persona muere cada 34 segundos por enfermedad cardiovascular, y en 2023, 919,032 personas murieron por esta causa, equivalente a 1 de cada 3 muertes en Estados Unidos (CDC, 2025). Esta alarmante estadística se replica globalmente, donde las ECV persisten como la principal causa de mortalidad mundial (Emerging rapid detection methods, 2025).

Los eventos cardiovasculares agudos, como infartos del miocardio, arritmias peligrosas y crisis hipertensivas, frecuentemente se desarrollan de manera silenciosa durante sus etapas iniciales. La detección tardía de estos eventos críticos puede resultar en consecuencias devastadoras. Los signos precursores, manifestados a través de variaciones en la frecuencia cardíaca, fluctuaciones en la presión arterial sistólica y diastólica, y patrones anómalos de actividad física, a menudo pasan desapercibidos hasta que el evento cardiovascular se encuentra en una fase avanzada.

La presión arterial sistólica elevada (superior a 140 mmHg) y la diastólica alta (superior a 90 mmHg) constituyen factores de riesgo primordiales para el desarrollo de hipertensión crónica, la cual puede derivar en complicaciones severas como accidentes cerebrovasculares, insuficiencia cardíaca congestiva y daño renal irreversible. Simultáneamente, las alteraciones en la frecuencia cardíaca—tanto taquicardia como bradicardia patológicas—pueden indicar arritmias subyacentes que requieren intervención inmediata.

Necesidad de Monitoreo Continuo y Detección Temprana

La medicina cardiovascular moderna reconoce que la tecnología wearable inteligente permitirá a los médicos monitorear tendencias en métricas de salud cardiovascular y desarrollar insights accionables informados por modelos de machine learning a nivel individual y poblacional (Medicine 2032, PMC). Esta transformación hacia el monitoreo continuo representa un paradigma fundamental en la prevención y tratamiento de emergencias cardiovasculares.

Los métodos tradicionales de monitoreo cardiovascular, basados en evaluaciones periódicas en consultorios médicos o hospitales, presentan limitaciones significativas. Estos enfoques proporcionan únicamente "fotografías instantáneas" del estado cardiovascular del paciente, perdiendo información valiosa sobre variaciones circadianas, respuestas al estrés, y patrones de deterioro gradual que podrían preceder a eventos críticos.

La integración de dispositivos wearables basados en IoT tiene el potencial de proporcionar monitoreo continuo de usuarios, permitiendo la detección temprana de problemas de salud potenciales (Scientific Reports, 2023). Esta capacidad de vigilancia permanente resulta especialmente crucial para pacientes con factores de riesgo elevados o antecedentes de eventos cardiovasculares previos.

Contexto del Proyecto

Justificación Tecnológica

El proyecto Heartguard surge como respuesta directa a la necesidad crítica de desarrollar sistemas de monitoreo cardiovascular predictivo que aprovechen las capacidades de la inteligencia artificial y el Internet de las Cosas (IoT). Aunque el sistema está diseñado conceptualmente para integrar dispositivos IoT reales de monitoreo cardiovascular, para efectos de este proyecto se implementará una simulación de la recopilación de datos de sensores, permitiendo demostrar las capacidades analíticas y predictivas del sistema sin depender de hardware especializado. La convergencia de estas tecnologías ofrece oportunidades sin precedentes para revolucionar la detección temprana de emergencias cardiovasculares.

Los avances recientes en enfoques impulsados por IA, incluyendo deep learning y algoritmos de clasificación avanzados, han mostrado promesas en mejorar la precisión de la clasificación de ECV, evaluación de riesgos y monitoreo de pacientes (JMIR Medical Informatics, 2025). Estos desarrollos tecnológicos crean el contexto ideal para implementar sistemas de monitoreo predictivo que puedan procesar grandes volúmenes de datos biométricos en tiempo real.

Arquitectura de Datos del Sistema

El sistema Heartguard está diseñado para procesar datos continuos provenientes de dispositivos de monitoreo personal que recopilan información crítica de cada usuario. La estructura fundamental de datos incluye: patient_id | heart_rate | blood_pressure_sys | blood_pressure_dia | timestamp | activity_level | gps_coordinates.

Esta arquitectura permite la captura sistemática de:

- **Frecuencia cardíaca (heart_rate)**: Mediciones por minuto que detectan arritmias, taquicardia o bradicardia patológicas
- **Presión arterial sistólica (blood_pressure_sys)**: Indicador de la fuerza ejercida durante la contracción ventricular
- **Presión arterial diastólica (blood_pressure_dia)**: Medición de la presión durante la relajación cardíaca

- **Identificador del paciente (patient_id)**: Vinculación segura con historiales médicos y perfiles de riesgo
- **Coordenadas GPS**: Localización para respuesta de emergencia optimizada
- **Patrones de actividad**: Correlación entre actividad física y eventos cardiovasculares

Problemática Actual en Sistemas de Salud

Los sistemas de salud enfrentan desafíos significativos en la gestión de usuarios con riesgo cardiovascular elevado. El reporte de 2024 revela una aceleración significativa en 2020-2021, cuando 425,147 muertes fueron vinculadas a insuficiencia cardíaca, representando el 45% de las muertes cardiovasculares (HFSA, 2024). Esta escalada evidencia la urgente necesidad de sistemas de detección temprana más efectivos.

La saturación de servicios de emergencia y la limitada capacidad de monitoreo continuo en entornos hospitalarios tradicionales exacerban esta problemática. Los usuarios de alto riesgo frecuentemente experimentan eventos cardiovasculares en sus hogares, donde la detección tardía resulta en intervenciones menos efectivas y mayores tasas de mortalidad.

Innovación e Impacto Esperado

Heartguard se posiciona como una solución integral que combina monitoreo continuo simulado, análisis predictivo mediante inteligencia artificial, y sistemas de alerta temprana automatizados. A diferencia de sistemas reactivos tradicionales, nuestra plataforma implementa un enfoque proactivo que identifica patrones precursores de eventos cardiovasculares críticos mediante la simulación realista de datos de dispositivos IoT cardiovasculares.

Aunque el proyecto utiliza datos simulados para demostrar las capacidades del sistema, el diseño arquitectónico está preparado para la integración futura con dispositivos reales de monitoreo cardiovascular. Esta aproximación permite validar algoritmos de detección temprana y desarrollar interfaces de usuario optimizadas sin las limitaciones técnicas y de costo asociadas con hardware médico especializado.

La implementación exitosa de Heartguard tiene el potencial de transformar significativamente los resultados clínicos para usuarios con riesgo cardiovascular. El sistema de detección temprana puede contribuir a la reducción de tasas de

mortalidad asociadas a eventos cardíacos agudos, optimizar la utilización de recursos de emergencia, y mejorar la calidad de vida de usuarios con condiciones cardiovasculares crónicas. La arquitectura escalable del proyecto permite su adaptación tanto para entornos hospitalarios como para programas de monitoreo domiciliario, estableciendo las bases para futuras implementaciones con dispositivos IoT reales.

Arquitectura del Sistema

Visión General de la Arquitectura

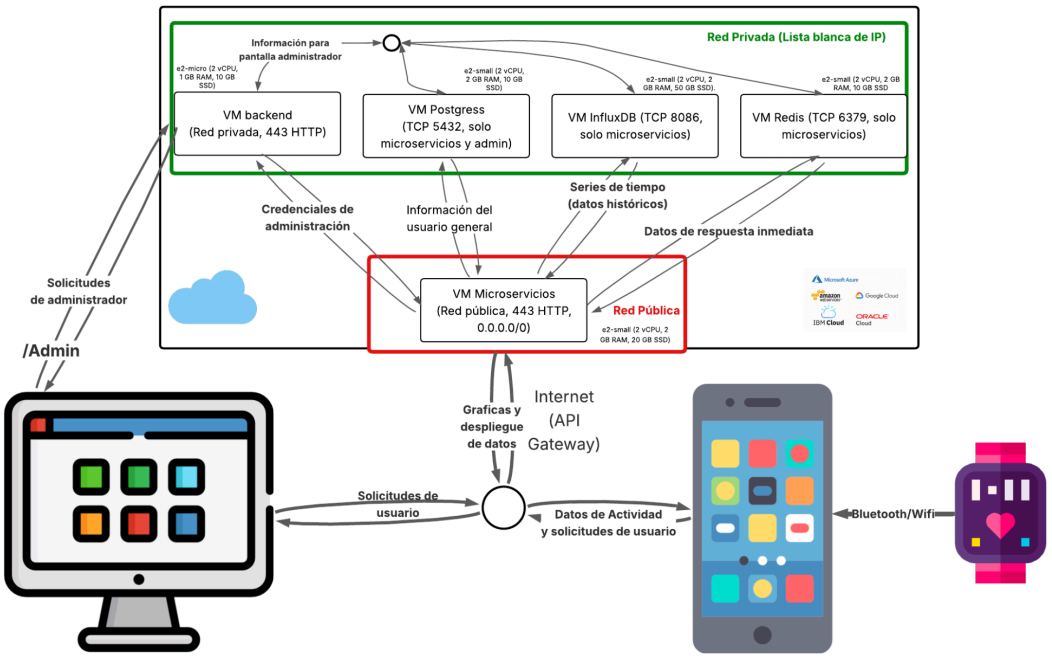
La arquitectura de Heartguard está fundamentada en un diseño de microservicios distribuidos que prioriza la escalabilidad, disponibilidad y mantenibilidad del sistema. Esta arquitectura sigue los principios de separación de responsabilidades y comunicación asíncrona, elementos críticos para sistemas de monitoreo médico en tiempo real que requieren alta disponibilidad y capacidad de respuesta inmediata.

El sistema adopta una aproximación de tres capas principales: Capa de Presentación (Web Frontend, Aplicación móvil, Backend), Capa de Microservicios (servicios especializados de dominio), y Capa de Persistencia (bases de datos especializadas). Esta separación arquitectónica permite que cada componente evolucione independientemente mientras mantiene interfaces bien definidas para la comunicación inter-servicios.

Componentes de Procesamiento Avanzado: La arquitectura está diseñada para integrar tecnologías especializadas como Apache Hadoop para el procesamiento distribuido de grandes volúmenes de datos cardiovasculares y identificación de patrones críticos, así como LeapMotion para interfaces de visualización avanzadas del panel de riesgo y evolución de usuarios. Estos componentes representan las capacidades de análisis de big data y visualización interactiva que distinguen al sistema Heartguard. Cabe mencionar que estos componentes los usaremos más adelante en el proyecto.

La elección de una arquitecura de microservicios para sistemas de salud se justifica porque esta arquitectura permite una entrega de software más rápida, pruebas automatizadas y una adaptación más fácil a las nuevas necesidades de atención médica, como el análisis complejo de datos. Además, la arquitectura de microservicios mejora la resistencia al aislar las fallas a servicios específicos, evitando que afecten a todo el sistema, un aspecto crucial para aplicaciones médicas críticas.

Arquitectura de Hardware



Arquitectura de Red Híbrida (Red Privada + Red Pública)

Implementación de una arquitectura de red híbrida con segregación entre red privada (lista blanca de IPs) para microservicios críticos y red pública para interfaces de usuario.

La implementación de una arquitectura de red híbrida se fundamenta en la necesidad de seguridad por capas, donde los microservicios que manejan datos sensibles de usuarios (InfluxDB, Redis, PostgreSQL) operan en una red privada con acceso restringido, cumpliendo con regulaciones como HIPAA y GDPR. Esta configuración permite una performance optimizada, ya que la comunicación entre microservicios internos utiliza la red privada de baja latencia, mientras que las interfaces públicas mantienen accesibilidad externa. Adicionalmente, proporciona aislamiento de fallos efectivo, garantizando que si la red pública experimenta problemas de conectividad, los servicios críticos continúen operando en la red interna sin interrupciones.

Especificaciones de Máquinas Virtuales Diferenciadas

Asignación de recursos computacionales específicos según la carga de trabajo de cada servicio:

- VM Backend (e2-micro, 2 vCPU, 1 GB RAM, 10 GB SSD): Para el backend principal en Go
- VM PostgreSQL (e2-small, 2 vCPU, 2 GB RAM, 10 GB SSD): Para gestión de datos relacionales
- VM InfluxDB (e2-small, 2 vCPU, 2 GB RAM, 50 GB SSD): Para almacenamiento de series temporales
- VM Redis (e2-small, 2 vCPU, 2 GB RAM, 10 GB SSD): Para caché y datos de respuesta inmediata

La diferenciación en las especificaciones de máquinas virtuales responde a una estrategia de optimización de costos donde cada VM está dimensionada según sus requerimientos reales de procesamiento y almacenamiento. Es particularmente notable que InfluxDB requiere mayor capacidad de almacenamiento debido a que captura datos de series temporales de alta resolución con la precisión y contexto que los modelos de IA necesitan para inferir causa y efecto, requiriendo mayor capacidad de almacenamiento para datos históricos cardiovasculares. Esta arquitectura también facilita la escalabilidad independiente, permitiendo que cada servicio pueda escalar verticalmente u horizontalmente según su demanda específica sin afectar el rendimiento de otros componentes del sistema.

Integración de Proveedores Cloud Múltiples

Soporte potencial para múltiples proveedores cloud (Microsoft Azure, Amazon AWS, Google Cloud, Oracle Cloud, IBM Cloud) como opción arquitectónica, aunque su implementación final dependerá de las necesidades específicas del proyecto y limitaciones presupuestarias.

- Evitar Vendor Lock-in: Flexibilidad para migrar entre proveedores según costos y características específicas
- Redundancia Geográfica: Capacidad de desplegar en múltiples regiones para recuperación ante desastres
- Compliance Regulatorio: Algunos proveedores tienen certificaciones específicas para datos médicos en diferentes jurisdicciones

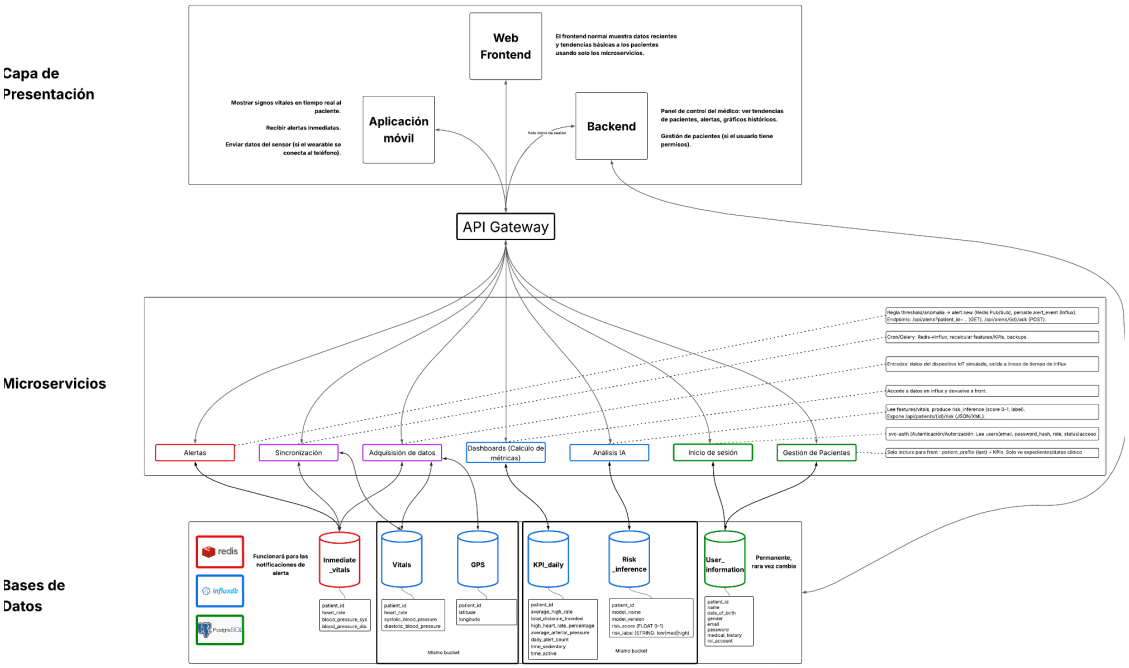
- Consideración Práctica: Esta funcionalidad representa una opción de escalabilidad futura más que un requerimiento inmediato del proyecto

Arquitectura de Dispositivos Cliente Diversificada

Soporte simultáneo para interfaces web (admin/desktop) y móvil nativa.

- Acceso Ubicuo: Usuarios y administradores requieren diferentes niveles de acceso desde diferentes dispositivos
- Experiencia Optimizada: Interfaces nativas proporcionan mejor rendimiento y experiencia de usuario que soluciones híbridas
- Preparación IoT: Arquitectura preparada para integración futura con dispositivos médicos reales mediante APIs estandarizadas

Arquitectura de Software



API Gateway como Punto de Entrada Único

Implementación de un API Gateway centralizado que gestiona todas las comunicaciones entre la capa de presentación y los microservicios.

La implementación de un API Gateway centralizado proporciona múltiples ventajas arquitectónicas críticas para un sistema médico. Permite la gestión de seguridad centralizada, consolidando autenticación, autorización y rate limiting en un único punto de control, lo que simplifica el mantenimiento y reduce vectores de ataque. La capacidad de agregación de servicios facilita la composición de respuestas de múltiples microservicios en una sola llamada, optimizando el rendimiento y reduciendo la complejidad del cliente. Adicionalmente, el gateway facilita el versionado de APIs durante actualizaciones del sistema, asegurando compatibilidad con versiones anteriores, y proporciona un punto centralizado para logging, métricas y análisis de performance, elementos esenciales para el monitoreo de sistemas críticos de salud.

Microservicios Especializados por Dominio

Separación de funcionalidades en microservicios específicos:

- Alertas: Gestión de notificaciones críticas y no críticas
- Sincronización: Coordinación de datos entre servicios distribuidos
- Adquisición de Datos: Simulación y normalización de datos de sensores IoT
- Dashboard/Métricas: Cálculo y presentación de indicadores cardiovasculares
- Análisis IA: Procesamiento de machine learning para detección predictiva
- Inicio de Sesión: Gestión de autenticación y autorización
- Gestión de Usuarios: CRUD

La separación de funcionalidades en microservicios especializados sigue el principio de responsabilidad única, donde cada microservicio tiene una responsabilidad específica y bien definida, facilitando el mantenimiento y la evolución independiente de cada componente. Esta arquitectura permite escalabilidad independiente, ya que las aplicaciones de atención médica necesitan ser resistentes y escalables para satisfacer las demandas fluctuantes, abordando esto al permitir el escalado dinámico y la distribución de recursos según las necesidades específicas de cada servicio. El enfoque también habilita el desarrollo paralelo, permitiendo que equipos diferentes trabajen en microservicios específicos sin conflictos, acelerando el tiempo de desarrollo. Fundamentalmente, proporciona tolerancia a fallos superior, ya que el fallo de un servicio no compromete la funcionalidad completa del sistema, un aspecto crítico para aplicaciones médicas donde la disponibilidad puede ser vital.

Estrategia de Bases de Datos Poliglota

Utilización de diferentes tecnologías de bases de datos según el patrón de uso de datos:

- Redis: Cache distribuido y datos de alerta inmediata
- InfluxDB: Series temporales para datos cardiovasculares históricos
- PostgreSQL: Datos relacionales (información de usuarios)

Tecnologías de Implementación Específicas

- Backend Principal: Go (Golang)

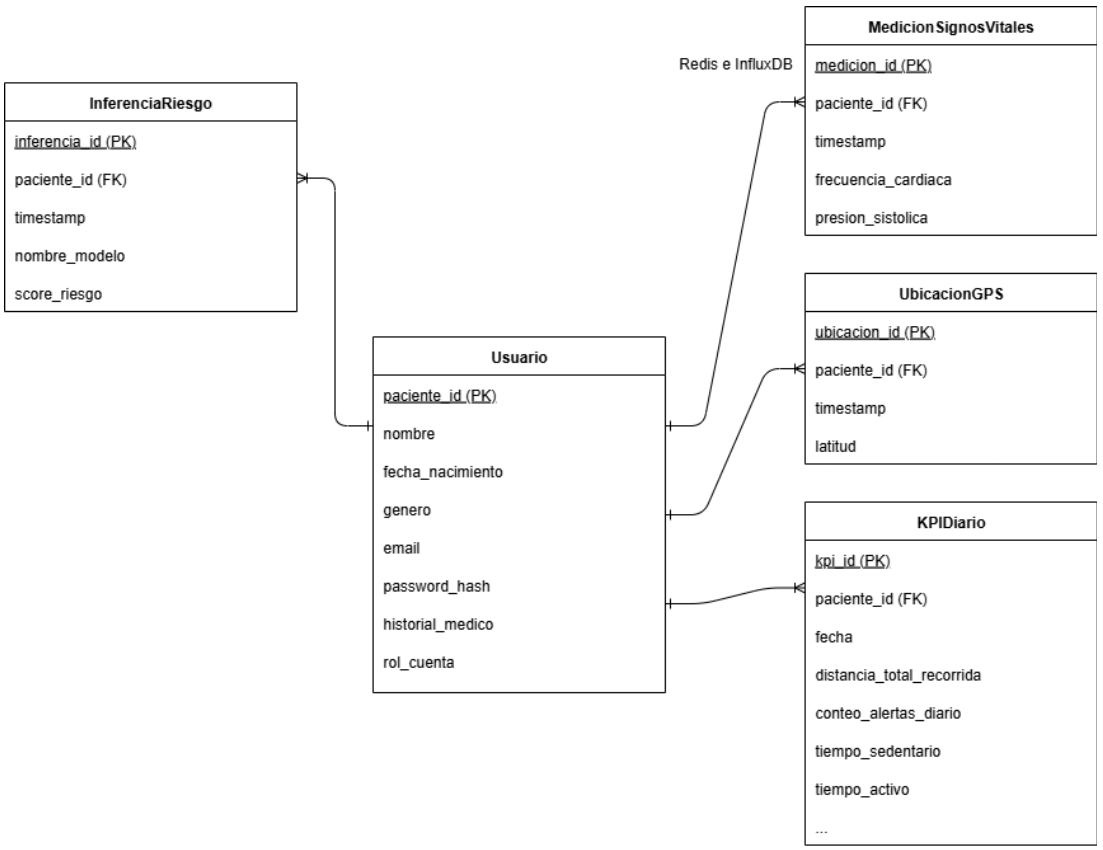
- Microservicios: Flask (Python)
- Aplicación Móvil: Java Android Nativo
- Frontend Web: Tecnologías web estándar

La selección de tecnologías específicas para cada componente optimiza el rendimiento y las capacidades según el dominio de aplicación. Go se utiliza para el backend principal debido a su performance excepcional en concurrencia, siendo ideal para el manejo de múltiples conexiones simultáneas de dispositivos médicos simulados y reales, además de su eficiencia en el uso de memoria y velocidad de ejecución. Flask se implementa para microservicios debido a su flexibilidad y rapidez de desarrollo para servicios específicos, aprovechando el excelente ecosistema Python para análisis de datos y machine learning, elementos fundamentales para el procesamiento de datos cardiovasculares. Java Android nativo se selecciona para la aplicación móvil por su performance superior y acceso completo a APIs del sistema operativo móvil, permitiendo notificaciones críticas eficientes y acceso a sensores del dispositivo. Esta separación de tecnologías representa un enfoque de separación de concerns, donde cada tecnología está optimizada para su dominio específico de aplicación, maximizando la eficiencia global del sistema.

Integración de Big Data y Control Gestual para la Visualización Cardiovascular Avanzada

Se propone incorporar Apache Hadoop más adelante en el proyecto para el procesamiento distribuido de grandes volúmenes de datos cardiovasculares y LeapMotion para desarrollar interfaces médicas sin contacto. Hadoop permitirá analizar patrones críticos en enfermedades crónicas mediante técnicas de big data, optimizando la clasificación y el monitoreo en tiempo real.

Diseño del Diagrama ER y Modelo de Datos



El diagrama entidad-relación presenta un modelo de datos centrado en la gestión integral de usuarios y monitoreo de signos vitales. La entidad principal Usuario almacena información demográfica y médica básica, conectándose con cuatro entidades especializadas: MedicionSignosVitales que registra frecuencia cardíaca y presión sistólica con timestamp; UbicacionGPS que captura la geolocalización del usuario en tiempo real; KPIDiario que consolida métricas diarias como distancia recorrida, conteo de alertas y tiempos de actividad; e InferenciaRiesgo que almacena evaluaciones de riesgo generadas por algoritmos de machine learning con scores y nombres de modelos específicos. Este diseño permite el seguimiento continuo del estado de salud del usuario mediante la correlación de datos biométricos, ubicación y patrones de actividad.

Este modelo de datos está diseñado para capturar y gestionar información integral de pacientes, incluyendo datos demográficos, signos vitales en tiempo real, ubicación GPS, métricas de actividad diaria y evaluaciones de riesgo generadas por algoritmos de machine learning, facilitando un monitoreo continuo y personalizado de la salud

User_information		
Campo	**Tipo de Dato**	**Descripción**
patient_id	VARCHAR(50) PRIMARY KEY	Identificador único del paciente
name	VARCHAR(100) NOT NULL	Nombre completo del paciente
date_of_birth	DATE	Fecha de nacimiento del paciente
gender	ENUM	Género del paciente (M/F/Otro/Prefiero no decir)
email	VARCHAR(150)	Correo electrónico del paciente
password	VARCHAR(255)	Contraseña encriptada para acceso
medical_history	TEXT	Historial médico del paciente
rol_account	ENUM	Rol de la cuenta (usuario/admin)

Inmediate_vitals		
Campo	**Tipo de Dato**	**Descripción**
patient_id	VARCHAR(50) FOREIGN KEY	Referencia al paciente
heart_rate	INTEGER	Frecuencia cardíaca en latidos por minuto
blood_pressure_sys	INTEGER	Presión arterial sistólica en mmHg
blood_pressure_dia	INTEGER	Presión arterial diastólica en mmHg
timestamp	TIMESTAMP	Fecha y hora de la medición

Vitals		
Campo	**Tipo de Dato**	**Descripción**
patient_id	VARCHAR(50) FOREIGN KEY	Referencia al paciente
heart_rate	INTEGER	Frecuencia cardíaca en latidos por minuto
systolic_blood_pressure	INTEGER	Presión arterial sistólica en mmHg
diastolic_blood_pressure	INTEGER	Presión arterial diastólica en mmHg
timestamp	TIMESTAMP	Fecha y hora de la medición

GPS

Campo	Tipo de Dato	Descripción
patient_id	VARCHAR(50) FOREIGN KEY	Referencia al paciente
latitude	DECIMAL(10,8)	Latitud de la ubicación GPS
longitude	DECIMAL(11,8)	Longitud de la ubicación GPS
timestamp	TIMESTAMP	Fecha y hora de la ubicación

KPI_daily

Campo	Tipo de Dato	Descripción
patient_id	VARCHAR(50) FOREIGN KEY	Referencia al paciente
average_high_rate	FLOAT	Promedio de frecuencia cardíaca alta del día
total_distance_traveled	FLOAT	Distancia total recorrida en metros
high_heart_rate_percentage	FLOAT	Porcentaje de tiempo con frecuencia cardíaca alta
average_arterial_pressure	FLOAT	Presión arterial promedio del día
daily_alert_count	INTEGER	Número de alertas generadas en el día
time_sedentary	TIME	Tiempo sedentario en formato HH:MM:SS
time_active	TIME	Tiempo activo en formato HH:MM:SS
date	DATE	Fecha del registro diario

Risk_Inference

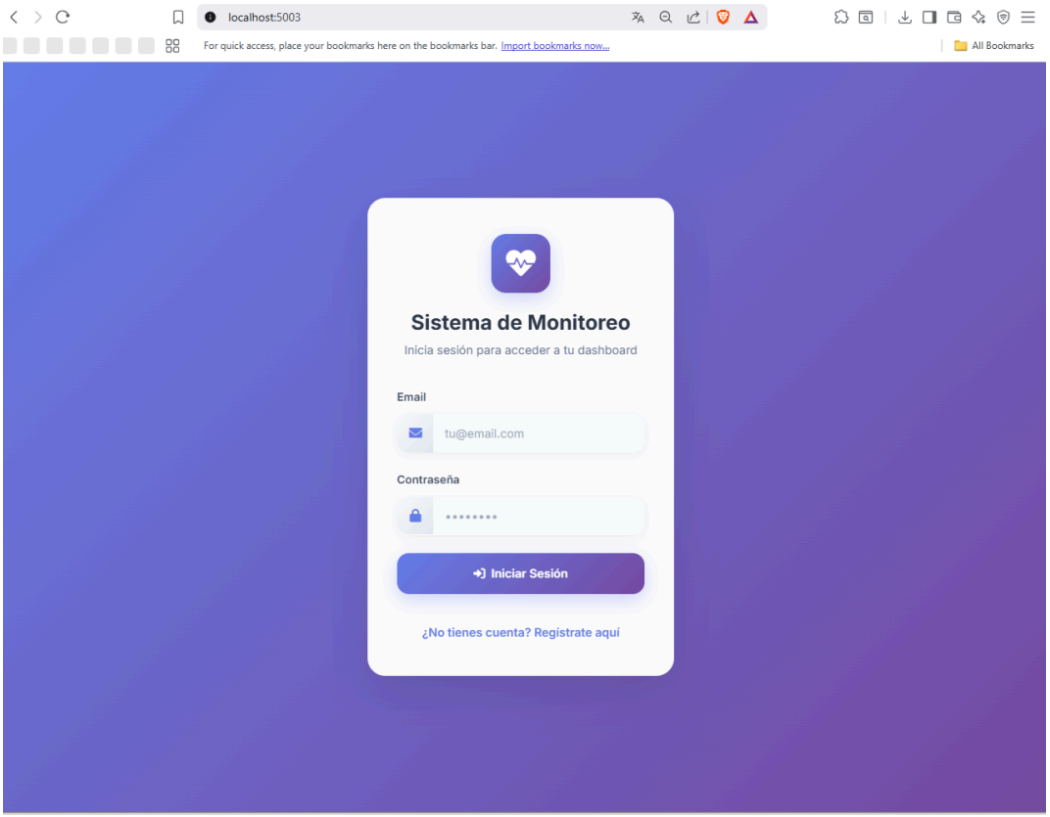
Campo	Tipo de Dato	Descripción
patient_id	VARCHAR(50) FOREIGN KEY	Referencia al paciente
model_name	VARCHAR(100)	Nombre del modelo de ML utilizado
model_version	VARCHAR(20)	Versión del modelo de inferencia
risk_score	FLOAT	Puntuación de riesgo (0.0 - 1.0)
risk_label	VARCHAR(10)	Etiqueta de riesgo (low/med/high)
timestamp	TIMESTAMP	Fecha y hora de la inferencia

Justificación del Diseño de la Interfaz

El diseño de la interfaz se basa en el objetivo central del proyecto: proveer un sistema de monitoreo en tiempo real que permita detectar signos tempranos de emergencias cardiovasculares y alertar oportunamente. Por lo tanto, la interfaz debía cumplir tres requisitos clave:

1. Claridad inmediata de la información crítica (frecuencia cardiaca, presión arterial, riesgo).
2. Separación de roles entre paciente/usuario y administrador para garantizar seguridad y control.
3. Simplicidad visual y baja carga cognitiva para que médicos, pacientes y paramédicos comprendan los datos sin entrenamiento adicional.

1. Pantalla de inicio de sesión

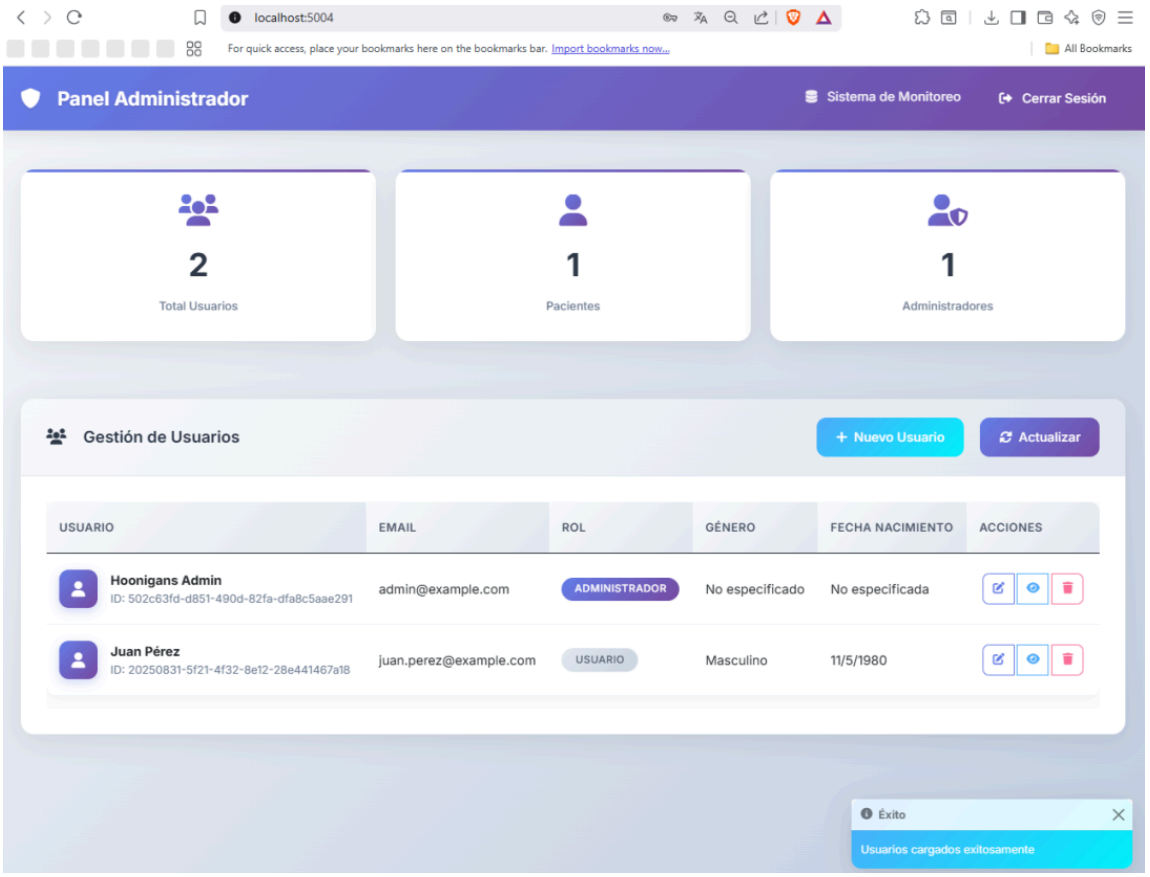


- Se diseñó un login minimalista con fondo de gradiente morado-azulado, transmitiendo confianza y profesionalismo (colores comúnmente asociados a la salud digital).

- Los campos de email y contraseña están centrados en la pantalla, reforzando la idea de acceso seguro.
- El botón de acción principal “Iniciar Sesión” está destacado en azul-morado, lo que genera un *call to action* claro.
- Se agregó la opción “¿No tienes cuenta? Regístrate aquí” para facilitar la incorporación de nuevos usuarios (pacientes o médicos).

La simplicidad evita distracciones en un momento crítico (inicio de sesión), cumple con la usabilidad de accesos seguros y refleja la existencia del microservicio svc-auth y la gestión de roles en la base de datos.

2. Panel de Administrador

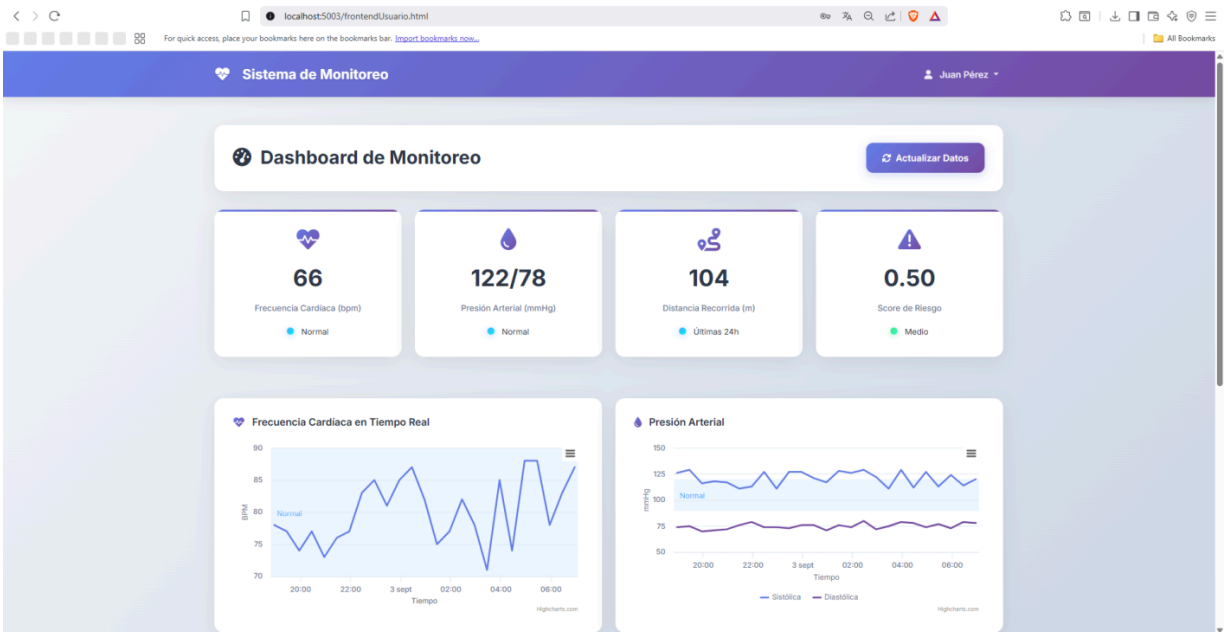


- Sección inicial con tarjetas-resumen: total de usuarios, pacientes y administradores. Esto da una vista rápida de la gestión del sistema.
- Módulo central de Gestión de Usuarios con tabla:
    - Datos clave visibles (usuario, email, rol, género, fecha nacimiento). Acciones CRUD representadas con íconos intuitivos (lápiz = editar, bote = eliminar, ojo = ver).

- Botones de “Nuevo Usuario” y “Actualizar” resaltados con colores contrastantes para tareas frecuentes.
- Se usan snackbars (mensajes emergentes no intrusivos) para confirmar acciones (“Usuarios cargados exitosamente”).

Este panel permite al administrador manejar el sistema sin sobrecarga visual, aplicando jerarquía de información. Los elementos CRUD se alinean con las operaciones de los microservicios que gestionan user_information.

3. Dashboard de Usuario/Paciente



- Diseño de tarjetas de KPIs en la parte superior con métricas críticas:
  - Frecuencia cardiaca (bpm).
  - Presión arterial (mmHg).
    Distancia recorrida (actividad física).
  - Score de riesgo (valor numérico con indicador de nivel).
- Cada tarjeta incluye ícono, valor en grande, unidad y estado textual (“Normal”, “Medio”).
- Debajo, gráficas en tiempo real desarrolladas con Highcharts:
  - Línea de frecuencia cardiaca.
  - Evolución de presión arterial (sistólica y diastólica).
- Botón “Actualizar Datos” destacado en la parte superior, asegurando que el usuario pueda refrescar la información de manera inmediata.

Se aplica un patrón de jerarquía visual: primero “estado actual” (tarjetas grandes con valores), después “tendencia histórica” (gráficas). Esto permite al médico/paciente tomar decisiones rápidas. Además, refleja directamente los datos provenientes de InfluxDB (series de tiempo) y Redis (alertas en tiempo real).

4. Principios de diseño aplicados

1. Consistencia visual: Uso de la misma paleta morado-azulada en login, panel y dashboard, generando coherencia.
2. Accesibilidad: Tipografía clara, contraste suficiente y tamaños grandes en valores numéricos críticos.
3. Feedback inmediato: Uso de colores (verde/amarillo/rojo) y mensajes emergentes para comunicar estados y resultados de acciones.
4. Reducción de carga cognitiva: Íconos y etiquetas explicativas ayudan a entender datos médicos complejos sin necesidad de capacitación técnica.
5. Diseño responsive: Basado en Bootstrap, garantizando que se adapte a monitores hospitalarios, tablets en ambulancias y dispositivos móviles.

Plan de Actividades

Semana 1 – Semana 2 (Primera Entrega – Septiembre 4)

Objetivo: BD lista (InfluxDB + Redis) + Front básico con Bootstrap + Demo parcial corriendo.

Aldo y Pablo (Arquitectura del Sistema)

- Diseñar juntos la arquitectura completa del sistema:
    - Componentes: Backend Go, microservicios Flask, InfluxDB, Redis, Front End, App Android, Hadoop.
    - Diagrama general de cómo fluyen los datos desde sensores hasta dashboards y alertas.
    - Roles de cada módulo y cómo se comunican (JSON/XML).

Aldo (BD / Backend)

- Diseñar Diagrama ER y modelo de datos lógico:
    - Entidades: Pacientes, Sensores, Signos vitales (HR, presión), Ubicación GPS, Alertas.
    - Relaciones y cardinalidades claras.
- Implementación InfluxDB + Redis:
    - Estructura para datos en tiempo real (InfluxDB para series temporales).
    - Redis para caché o alertas inmediatas.
- Script init con datos simulados (CSV).
- 5 consultas complejas:
    - Tendencias de HR y presión.
    - Pacientes con picos anormales.
    - Detección por patrones horarios.

Pablo (Front / Mockups)

- Mockups del panel de riesgo y evolución (Bootstrap).
- Prototipo Front:
    - Navbar, dashboard y cards.
    - Conexión simulada a JSON/XML (endpoints de prueba).

- Integración inicial de gráficas (Chart.js o Highcharts).
- Panel de administrador básico.

Trabajo conjunto:

- Repositorio GitHub + despliegue mínimo en Ubiquitous.
- Reporte formal:
    - Aldo: BD y ER.
    - Pablo: Mockups y diseño UI.

- Presentación técnica (10 min): práctica con demo mínima.

Semana 3 – Semana 6 (Segunda Entrega – Mediados de Octubre)

Objetivo: Backends reales (Go + Flask microservicios) + Front End consumiendo datos reales.

Aldo (Backend / BD)

- Desarrollar Backend en Go:
  - API REST principal para pacientes, alertas, signos vitales.
- Crear microservicio Flask para analítica rápida (detección de picos).
- Exponer endpoints JSON y XML.
- Dockerización de InfluxDB + Redis + Go Backend.
- Documentar APIs (Swagger o Postman).

Pablo (Front / UI)

- Conectar Front a Backend real (Go/Flask):
  - Fetch de datos reales desde InfluxDB/Redis.
- Dashboards dinámicos:
  - KPIs en vivo (HR promedio, riesgo).
  - Gráficas en tiempo real (Chart.js/Highcharts).
- Bootstrap responsive + diseño limpio.
- Dockerización del Front.

Trabajo conjunto:

- Pruebas integradas (Front + Backend + BD).
- Despliegue en Ubiquitous estable.
- Demo avanzada:
  - Alertas en tiempo real.
  - Panel de evolución.

Semana 7 – Semana 10 (Entrega Final – Mediados de Noviembre)

Objetivo: Hadoop para analítica avanzada + App Android + Despliegue en Google Cloud.

Aldo (Analítica / Infraestructura)

- Configurar clúster Hadoop (detección de patrones críticos).
- Integrar procesamiento de datos con alertas predictivas.
- Optimizar InfluxDB y Redis (rendimiento y latencia).
- Docker final de todos los módulos + despliegue en Google Cloud.

Pablo (Android / UI final)

- Desarrollar App Android (Java):
    - Ver signos vitales y alertas en tiempo real.
    - Conexión al backend.
- Pulido final del Front (Bootstrap + LeapMotion).
- Dashboard con insights Hadoop (riesgo predictivo).

Trabajo conjunto:

- Pruebas completas (Web + App + Backend + BD + Hadoop).
- Reporte final y presentación técnica completa.
- Validación del despliegue en Google Cloud (balanceo básico).

Repositorio Github

https://github.com/interminableDgo/IntegracionP-Final

Referencias

American Chemical Society. (2024). Five advances that could change heart health monitoring. ACS Central Science. Disponible en: https://pubs.acs.org/journal/acscii

Bekey.io. Cómo puede la arquitectura de microservicios beneficiar a la atención médica. Disponible en: https://bekey.io/blog/microservices-architecture-benefit-healthcare

Centers for Disease Control and Prevention. (2025). Heart Disease Facts. Recuperado de https://www.cdc.gov/heart-disease/data-research/facts-stats/index.html

Emerging rapid detection methods for the monitoring of cardiovascular diseases: Current trends and future perspectives. (2025). ScienceDirect. Disponible en: https://www.sciencedirect.com/topics/medicine-and-dentistry/cardiovascular-disease

Emids. (2023, March 16). 4 Benefits of Using a Microservices Architecture in Digital and Connected Health Solutions. Disponible en: https://www.emids.com/insights/benefits-of-a-microservices-architecture/

HealthTech Magazine. (2024, September 23). What Is Microservice Architecture & How Is Healthcare Adopting It? Disponible en: https://healthtechmagazine.net/article/2024/09/what-is-microservice-architecture-perfcon

Heart Failure Society of America. (2024). Cardiology Experts Warn of Rising Heart Failure Rates and Worsening Disparities in New 2024 Report. Disponible en: https://www.hfsa.org/news/hfsa-news/

IEEE Conference. A Method of 3D Hand Movement Recognition by a Leap Motion Sensor for Controlling Medical Image in an Operating Room. Disponible en: https://ieeexplore.ieee.org/document/8645985

InfluxData. (2022, January 15). InfluxDB Time Series Data Platform. Disponible en: https://www.influxdata.com/

InfluxData. (2022, February 3). Redis Monitoring Template. Disponible en: https://www.influxdata.com/influxdb-templates/redis/

Khatri, S., et al. (2020). A Hadoop-Based Platform for Patient Classification and Disease Diagnosis in Healthcare Applications. Sensors, 20(8), 2253. Disponible en: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7180448/

Khanna, N. N., Maindarkar, M. A., Viswanathan, V., Fernandes, J. F. E., Paul, S., Bhagawati, M., ... & Suri, J. S. (2025). The Role of AI in Cardiovascular Event Monitoring and Early Detection: Scoping Literature Review. JMIR Medical Informatics, 13, e64349. https://medinform.jmir.org/2025/1/e64349

Kumar, A., Sharma, K., Singh, H., Naugriya, S. G., Gill, S. S., & Buyya, R. (2023). Heart failure patients monitoring using IoT-based remote monitoring system. Scientific Reports, 13(1), 2471. https://www.nature.com/articles/s41598-023-29710-0

LeapMotion. (2017, September 14). 5 Medical and Assistive Technologies Being Transformed with Leap Motion. Medium. Disponible en: https://medium.com/@LeapMotion/5-medical-and-assistive-technologies-being-transformed-with-leap-motion-7087acfd309a

Li, J., et al. (2018). Application of a Mobile Chronic Disease Health-Care System for Hypertension Based on Big Data Platforms. Journal of Sensors, 2018. Disponible en: https://onlinelibrary.wiley.com/doi/10.1155/2018/3265281

Mehta, N., & Pandit, A. (2018). Concurrence of big data analytics and healthcare: A systematic review. International Journal of Medical Informatics, 114, 57-65. Disponible en: https://pmc.ncbi.nlm.nih.gov/articles/PMC4720168/

OptiSol Business. (2024, September 25). Transforming Healthcare: 5 Advantages of Implementing Microservices Architecture. Disponible en: https://www.optisolbusiness.com/insight/transforming-healthcare-5-advantages-of-implementing-microservices-architecture

Red Hat. How microservices support IT integration in healthcare. Disponible en: https://www.redhat.com/en/topics/microservices/microservices-in-healthcare

ResearchGate. (2017, March). Touchless control module for diagnostic images at the surgery room using the Leap Motion system and 3D Slicer Software. Disponible en: https://www.researchgate.net/publication/316673884_Touchless_control_module_for_diagnostic_images_at-the-surgery-room-using-the-Leap-Motion-system-and-3D-Slicer-Software

ResearchGate. (2024, November 30). Improving healthcare application scalability through microservices architecture in the cloud. Disponible en: https://www.researchgate.net/publication/386273829_Improving_healthcare_application_scalability_through_microservices-architecture-in-the-cloud

Sparrow, R. T., Torkamani, A., & Steinhubl, S. R. (2023). Medicine 2032: The future of cardiovascular disease prevention with machine learning and digital health technology. American Journal of Preventive Medicine, 64(4), 519-527. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10023328/