



Universidad Tecmilenio

Amir Sebastián Flores Cardona

Ingeniería en desarrollo de software.

Proyecto Integrador de la metodología DevOps

Propuesta de Proyecto

Alera: Plataforma Móvil para Registro, Análisis y Consulta Inteligente de Datos Personales

1. Problema o Necesidad que Atiende

Actualmente, los usuarios generan datos personales de manera constante relacionados con sus hábitos y actividades diarias (salud, productividad, bienestar). Sin embargo, estas métricas suelen permanecer dispersas o sin un análisis estructurado que permita:

- Registrar información de forma organizada y consistente
- Analizar tendencias básicas a partir de datos históricos
- Visualizar métricas clave de manera clara e intuitiva

2. Arquitectura del Sistema

La arquitectura de Alera se compone de seis capas principales:

1. **Aplicación Móvil (Frontend)** — React Native + Expo
2. **Backend/API** — Node.js con API REST y cron jobs
3. **Base de Datos** — PostgreSQL administrado mediante Supabase
4. **Autenticación** — Supabase Auth con JWT
5. **Inteligencia Artificial** — Integración vía OpenAI API
6. **DevOps** — CI/CD con GitHub Actions y Docker

3. Aplicación Móvil (Cliente)

Tipo de Aplicación

Aplicación móvil multiplataforma desarrollada con:

- React Native + Expo
- Un solo código base para Android e iOS

Funcionalidades Principales

1. Autenticación de usuarios (login y registro)
2. Registro de hábitos diarios (limitado a 2 tipos de hábito)
3. Visualización de métricas diarias básicas
4. Asistente conversacional con IA (contexto limitado)
5. Gestión básica de objetivos diarios

Tipo de Información Gestionada

- Datos de perfil del usuario
- Registros diarios de hábitos
- Métricas diarias agregadas
- Historial limitado de conversaciones con IA
- Objetivos diarios por hábito

4. Servidor (Backend API)

Funciones del Servidor

El backend en Node.js actúa como el núcleo del sistema y es responsable de:

- Exponer una API REST para la aplicación móvil
- Validar tokens JWT emitidos por Supabase Auth
- Orquestrar la lógica de negocio
- Ejecutar cron jobs para análisis de datos
- Preparar contexto y comunicarse con la IA

Procesos Implementados (MVP)

- CRUD de registros de hábitos
- Validación de datos de entrada
- Cálculo de métricas **diarias**
- Ejecución de **un cron job diario**
- Integración con IA con contexto limitado
- Rate limiting básico por usuario
- Métricas semanales y mensuales
- Comparaciones entre períodos
- Análisis predictivo

5. Base de Datos

Tipo de Base de Datos

Tabla: habits

- id (UUID)
- user_id (UUID)
- name (VARCHAR)
- type (VARCHAR)
- unit (JSONB)
- created_at (TIMESTAMP)
- updated_at (TIMESTAMP)

Tabla: habits_log

- id (UUID)
- user_id (UUID)
- habit_id (UUID)
- value (JSONB)
- metadata (JSONB)
- created_at (TIMESTAMP)

Tabla: metrics

- id (UUID)
- user_id (UUID)
- habit_id (VARCHAR)
- metric_type (VARCHAR)
- value (NUMERIC)
- date (DATE)
- calculated_at (TIMESTAMP)

Tabla: ai_conversations

- id (UUID)
- user_id (UUID)
- message (TEXT)
- role (VARCHAR)
- created_at (TIMESTAMP)

Tabla: user_goals

- id (UUID)
- user_id (UUID)
- habit_type (VARCHAR)
- target_value (NUMERIC)
- created_at (TIMESTAMP)

Seguridad

- Supabase Auth gestiona identidad y JWT
- Row Level Security (RLS) garantiza aislamiento por usuario

6. Inteligencia Artificial**Uso de IA en el MVP**

La IA se utiliza como un asistente conversacional que:

- Responde preguntas sobre hábitos recientes
- Utiliza contexto limitado preparado por el backend

Contexto Incluido

- Métricas del día actual
- Métricas de los **últimos 7 días**
- Métricas de los últimos 3 meses
- Contexto de los últimos 7 días relacionado a conversaciones con IA

7. Flujo DevOps

Pipeline CI/CD

1. Trigger por push o pull request
2. Análisis de código (ESLint)
3. Build del backend
4. Construcción de imagen Docker

Architecture Diagram



