**📚 Documentación Completa - Sistema de Sanciones INSEVIG**

**📋 Tabla de Contenidos**

1. [Visión General](https://claude.ai/chat/63e01d8e-62bc-4281-9cd5-6e39f2262c80#visi%C3%B3n-general)
2. [Arquitectura del Sistema](https://claude.ai/chat/63e01d8e-62bc-4281-9cd5-6e39f2262c80#arquitectura-del-sistema)
3. [Estructura de Datos](https://claude.ai/chat/63e01d8e-62bc-4281-9cd5-6e39f2262c80#estructura-de-datos)
4. [Sistema de Autenticación](https://claude.ai/chat/63e01d8e-62bc-4281-9cd5-6e39f2262c80#sistema-de-autenticaci%C3%B3n)
5. [Funcionalidad Offline](https://claude.ai/chat/63e01d8e-62bc-4281-9cd5-6e39f2262c80#funcionalidad-offline)
6. [Flujos de Trabajo](https://claude.ai/chat/63e01d8e-62bc-4281-9cd5-6e39f2262c80#flujos-de-trabajo)
7. [Integración con Supabase](https://claude.ai/chat/63e01d8e-62bc-4281-9cd5-6e39f2262c80#integraci%C3%B3n-con-supabase)
8. [Guía de Componentes](https://claude.ai/chat/63e01d8e-62bc-4281-9cd5-6e39f2262c80#gu%C3%ADa-de-componentes)
9. [API y Servicios](https://claude.ai/chat/63e01d8e-62bc-4281-9cd5-6e39f2262c80#api-y-servicios)
10. [Troubleshooting](https://claude.ai/chat/63e01d8e-62bc-4281-9cd5-6e39f2262c80#troubleshooting)

**🎯 Visión General**

**Descripción**

Sistema de registro y gestión de sanciones laborales para INSEVIG, desarrollado en Flutter con arquitectura dual (Web + Móvil) que permite:

* Registro de sanciones con evidencia fotográfica y firma digital
* Flujo de aprobación jerárquico (Supervisor → Gerencia → RRHH)
* Funcionamiento offline completo en móvil
* Sincronización automática cuando hay conexión
* Gestión de empleados desde proyecto Supabase separado

**Características Principales**

* **🌐 Multiplataforma**: Web y Móvil (Android/iOS)
* **📱 Modo Offline**: Funcionalidad completa sin conexión (solo móvil)
* **🔄 Sincronización**: Automática al recuperar conexión
* **📸 Compresión de Imágenes**: Automática antes de subir
* **✍️ Firma Digital**: Captura de firma del sancionado
* **👥 Roles Jerárquicos**: Supervisor, Gerencia, RRHH, Admin
* **🔐 Autenticación**: Basada en Supabase Auth

**🏗️ Arquitectura del Sistema**

**Diagrama de Arquitectura**

graph TB

subgraph "Frontend Flutter"

UI[UI Layer<br/>Screens & Widgets]

PROV[Providers<br/>State Management]

REPO[Repositories<br/>Data Access]

SVC[Services<br/>Business Logic]

OFF[Offline Manager<br/>Sync & Cache]

end

subgraph "Backend Supabase"

AUTH[Supabase Auth<br/>Authentication]

DB1[DB Sanciones<br/>Main Database]

DB2[DB Empleados<br/>Employee Database]

STOR[Storage<br/>Photos & Signatures]

end

subgraph "Local Storage"

HIVE[Hive DB<br/>Offline Cache]

end

UI --> PROV

PROV --> REPO

REPO --> SVC

REPO --> OFF

OFF --> HIVE

SVC --> AUTH

SVC --> DB1

SVC --> DB2

SVC --> STOR

OFF -.->|Sync| SVC

**Capas del Sistema**

**1. Capa de Presentación (UI)**

* **Screens**: Pantallas principales de la aplicación
* **Widgets**: Componentes reutilizables
* **Estado**: Manejado con Provider

**2. Capa de Lógica de Negocio**

* **Repositories**: Abstracción de acceso a datos con soporte offline
* **Services**: Lógica de negocio y comunicación con Supabase
* **Providers**: Gestión de estado global

**3. Capa de Datos**

* **Modelos**: Estructuras de datos (Sancion, Empleado, User)
* **Offline Manager**: Gestión de cache y sincronización
* **Hive**: Base de datos local para modo offline

**4. Infraestructura**

* **Supabase Config**: Configuración dual de proyectos
* **Connectivity Service**: Detección de conexión
* **Image Compression**: Optimización de imágenes

**📊 Estructura de Datos**

**Modelo de Sanciones (SancionModel)**

class SancionModel {

String id; // UUID único

String supervisorId; // ID del supervisor que crea

int empleadoCod; // Código del empleado

String empleadoNombre; // Nombre completo

String puesto; // Departamento del empleado

String agente; // Nombre del empleado (quien comete la falta)

DateTime fecha; // Fecha de la sanción

String hora; // Hora del incidente

String tipoSancion; // Tipo de sanción (13 tipos)

String? observaciones; // Descripción detallada

String? observacionesAdicionales; // Info adicional

bool pendiente; // Estado pendiente/resuelto

String? fotoUrl; // URL de foto evidencia

String? firmaPath; // URL de firma digital

int? horasExtras; // Solo para HORAS EXTRAS

String status; // borrador|enviado|aprobado|rechazado

String? comentariosGerencia; // Incluye código descuento

String? comentariosRrhh; // Comentarios de RRHH

DateTime? fechaRevision; // Fecha de última revisión

String? reviewedBy; // Usuario que revisó

DateTime createdAt; // Fecha creación

DateTime updatedAt; // Última actualización

}

**Modelo de Empleados (EmpleadoModel)**

class EmpleadoModel {

int id; // ID interno

int cod; // Código único empleado

String? nombres; // Nombres

String? apellidos; // Apellidos

String? cedula; // Cédula identidad

String? nombresCompletos; // Nombre completo

String? nomcargo; // Nombre del cargo

String? nomdep; // Departamento

String? seccion; // Sección específica

bool esActivo; // Estado activo

bool esLiquidado; // Estado liquidado

bool esSuspendido; // Estado suspendido

// ... más campos disponibles

}

**Modelo de Usuarios (UserModel)**

class UserModel {

String id; // UUID Supabase

String email; // Email login

String fullName; // Nombre completo

String role; // supervisor|gerencia|rrhh|admin

String? department; // Departamento asignado

bool isActive; // Estado activo

// Permisos calculados:

bool canCreateSanciones; // Puede crear sanciones

bool canApprove; // Puede aprobar

bool canViewAllSanciones; // Ver todas las sanciones

}

**Tipos de Sanciones Disponibles**

static const List<String> tiposSancion = [

'FALTA', // ❌ Ausencia injustificada

'ATRASO', // ⏰ Llegada tarde

'PERMISO', // 📋 Permiso no autorizado

'DORMIDO', // 😴 Dormido en servicio

'MALA URBANIDAD', // 🗣️ Comportamiento inadecuado

'FALTA DE RESPETO', // 😠 Irrespeto a superiores

'MAL UNIFORMADO', // 👔 Uniforme incorrecto

'ABANDONO DE PUESTO', // 🏃 Dejar puesto sin permiso

'MAL SERVICIO DE GUARDIA', // 🛡️ Servicio deficiente

'INCUMPLIMIENTO DE POLITICAS', // 📋 No seguir políticas

'MAL USO DEL EQUIPO DE DOTACIÓN', // ⚠️ Mal uso equipos

'HORAS EXTRAS', // ⏱️ Registro horas extras

'FRANCO TRABAJADO' // 📅 Trabajo en franco

];

**🔐 Sistema de Autenticación**

**Flujo de Autenticación**

sequenceDiagram

participant U as Usuario

participant L as LoginScreen

participant A as AuthProvider

participant S as Supabase Auth

participant P as Profiles Table

U->>L: Ingresa credenciales

L->>A: signIn(email, password)

A->>S: signInWithPassword()

S-->>A: Auth Response

A->>P: Obtener perfil usuario

P-->>A: UserModel data

A-->>L: Success/Error

L->>U: Navigate to Home/Show Error

**Roles y Permisos**

| **Rol** | **Crear Sanciones** | **Ver Todas** | **Aprobar** | **Revisar RRHH** |
| --- | --- | --- | --- | --- |
| **Supervisor** | ✅ | ❌ | ❌ | ❌ |
| **Gerencia** | ✅ | ✅ | ✅ | ❌ |
| **RRHH** | ❌ | ✅ | ✅ | ✅ |
| **Admin** | ✅ | ✅ | ✅ | ✅ |

**Implementación**

// AuthProvider gestiona el estado de autenticación

class AuthProvider with ChangeNotifier {

UserModel? \_currentUser;

// Login

Future<bool> signIn(String email, String password);

// Verificar permisos

bool get isSupervisor;

bool get isGerencia;

bool get isRrhh;

bool get isAdmin;

}

**📱 Funcionalidad Offline**

**Arquitectura Offline**

graph LR

subgraph "Modo Online"

A[App] --> B[Repository]

B --> C[Service]

C --> D[Supabase]

end

subgraph "Modo Offline"

E[App] --> F[Repository]

F --> G[OfflineManager]

G --> H[Hive Cache]

G -.->|Queue| I[Sync Queue]

end

subgraph "Sincronización"

I -->|When Online| C

end

**Características Offline**

1. **Detección Automática**: ConnectivityService monitorea el estado
2. **Cache Local**: Hive almacena empleados y sanciones
3. **Cola de Sincronización**: Operaciones pendientes se guardan
4. **Sync Automático**: Al recuperar conexión se sincronizan datos

**Operaciones Soportadas Offline**

* ✅ Buscar empleados (desde cache)
* ✅ Crear sanciones (se sincronizan después)
* ✅ Ver historial de sanciones
* ✅ Actualizar sanciones existentes
* ⚠️ Fotos/Firmas se suben al sincronizar

**Implementación**

// OfflineManager coordina todo

class OfflineManager {

// Detectar modo

bool get isOfflineMode;

// Operaciones

Future<String?> createSancion(...);

Future<List<EmpleadoModel>> searchEmpleados(String query);

// Sincronización

Future<bool> syncNow();

Map<String, dynamic> getOfflineStats();

}

**🔄 Flujos de Trabajo**

**1. Flujo de Creación de Sanción**

stateDiagram-v2

[\*] --> Crear: Supervisor inicia

Crear --> Buscar: Busca empleado

Buscar --> Completar: Autocompleta campos

Completar --> Evidencia: Agrega foto/firma

Evidencia --> Guardar: Guarda sanción

Guardar --> Borrador: Como borrador

Guardar --> Enviado: Enviar a gerencia

Borrador --> Enviado: Enviar después

Enviado --> [\*]: Fin

**2. Flujo de Aprobación Jerárquica**

stateDiagram-v2

[\*] --> Enviado: Sanción enviada

Enviado --> Gerencia: Revisión gerencia

Gerencia --> Aprobado: Aprobar con código

Gerencia --> Rechazado: Rechazar

Aprobado --> RRHH: Revisión RRHH

RRHH --> Confirmado: Confirmar

RRHH --> Modificado: Modificar decisión

RRHH --> Anulado: Anular sanción

Confirmado --> [\*]: Procesado

Modificado --> [\*]: Procesado

Anulado --> [\*]: Anulado

Rechazado --> [\*]: Fin

**3. Flujo de Sincronización Offline**

sequenceDiagram

participant App

participant OfflineManager

participant Queue as Sync Queue

participant Service

participant Supabase

App->>OfflineManager: Crear sanción offline

OfflineManager->>Queue: Agregar a cola

OfflineManager-->>App: ID temporal

Note over App,Supabase: Dispositivo sin conexión...

App->>OfflineManager: Detecta conexión

OfflineManager->>Queue: Obtener pendientes

loop Por cada operación

OfflineManager->>Service: Ejecutar operación

Service->>Supabase: Sincronizar

Supabase-->>Service: Respuesta

Service-->>OfflineManager: Success/Error

end

OfflineManager->>Queue: Limpiar cola

OfflineManager-->>App: Sync completado

**🔗 Integración con Supabase**

**Arquitectura Dual de Proyectos**

Proyecto 1 - Sistema Sanciones:

URL: https://syxzopyevfuwymmltbwn.supabase.co

Tablas:

- profiles (usuarios del sistema)

- sanciones (registros de sanciones)

Storage:

- sancion-photos (fotos de evidencia)

- sancion-signatures (firmas digitales)

Auth: Maneja autenticación de usuarios

Proyecto 2 - Empleados INSEVIG:

URL: https://buzcapcwmksasrtjofae.supabase.co

Tablas:

- empleados (base de datos de empleados)

Propósito: Base centralizada de empleados

**Configuración**

class SupabaseConfig {

// Inicialización dual

static Future<void> initialize() {

// 1. Proyecto principal (sanciones + auth)

await Supabase.initialize(url: sancionesUrl, ...);

// 2. Cliente secundario (empleados)

\_empleadosClient = SupabaseClient(empleadosUrl, ...);

}

// Acceso a clientes

static SupabaseClient get sancionesClient;

static SupabaseClient get empleadosClient;

}

**Políticas RLS Necesarias**

-- Tabla sanciones

- Supervisores: INSERT, SELECT propias, UPDATE borradores

- Gerencia: SELECT todas, UPDATE para aprobar

- RRHH: SELECT todas, UPDATE para revisar

- Admin: Todos los permisos

-- Tabla empleados (proyecto separado)

- Todos los usuarios autenticados: SELECT

- Sin permisos de modificación

-- Storage

- Usuarios autenticados: INSERT, SELECT

- Sin DELETE (auditoría)

**🧩 Guía de Componentes**

**Pantallas Principales**

**1. LoginScreen (login\_screen.dart)**

* Autenticación de usuarios
* Sin credenciales hardcodeadas
* Manejo de errores amigable

**2. HomeScreen (home\_screen.dart)**

* Dashboard con estadísticas
* Acciones rápidas por rol
* Indicador de modo offline
* Herramientas de debug (solo desarrollo)

**3. CreateSancionScreen (create\_sancion\_screen.dart)**

* Búsqueda de empleados con autocompletado
* Captura de foto con compresión automática
* Pad de firma digital
* Validación de campos
* Guardado como borrador o envío directo

**4. HistorialSancionesScreen (historial\_sanciones\_screen.dart)**

* Lista de sanciones con filtros
* Tabs por estado (pendientes/aprobadas/etc)
* Búsqueda y ordenamiento
* Acciones según rol del usuario

**Widgets Reutilizables**

**EmpleadoSearchField (empleado\_search\_field.dart)**

// Campo de búsqueda con autocompletado

EmpleadoSearchField(

onEmpleadoSelected: (empleado) {

// Autocompletar campos

},

)

**SancionCard (sancion\_card.dart)**

// Tarjeta de sanción con acciones

SancionCard(

sancion: sancion,

onTap: () => \_verDetalle(),

showActions: true,

)

**Diálogos de Aprobación**

* AprobacionGerenciaDialog: Códigos de descuento
* RevisionRrhhDialog: Revisión final
* CodigoDescuentoDialog: Selector de códigos

**🔧 API y Servicios**

**Servicios Principales**

**1. SancionService**

class SancionService {

// CRUD Básico

Future<String> createSancion({sancion, foto, firma});

Future<bool> updateSancion(sancion);

Future<bool> deleteSancion(id);

// Consultas

Future<List<SancionModel>> getAllSanciones();

Future<List<SancionModel>> getMySanciones(supervisorId);

Future<SancionModel?> getSancionById(id);

// Flujo jerárquico

Future<bool> aprobarConCodigoGerencia(...);

Future<bool> revisionRrhh(...);

// Estadísticas

Future<Map<String, dynamic>> getEstadisticas();

}

**2. EmpleadoService**

class EmpleadoService {

// Búsqueda con paginación

Future<List<EmpleadoModel>> searchEmpleados(query);

Future<List<EmpleadoModel>> getAllEmpleadosActivos();

// Consultas específicas

Future<EmpleadoModel?> getEmpleadoByCod(cod);

Future<List<String>> getDepartamentos();

Future<bool> puedeSerSancionado(cod);

}

**3. ImageCompressionService**

class ImageCompressionService {

// Compresión automática

static Future<File> compressImage(File image);

// Configuración

static const int maxWidth = 1024;

static const int maxHeight = 1024;

static const int quality = 85;

}

**Repositories (Capa de Abstracción)**

**SancionRepository**

* Wrapper sobre SancionService
* Gestiona modo online/offline automáticamente
* Mantiene consistencia de datos

**EmpleadoRepository**

* Wrapper sobre EmpleadoService
* Cache local de empleados
* Búsqueda offline desde cache

**🛠️ Troubleshooting**

**Problemas Comunes y Soluciones**

**1. Error de Conexión con Supabase**

Error: Failed host lookup

**Solución**:

* Verificar conexión a internet
* Verificar URLs y API keys en supabase\_config.dart
* En móvil: La app cambiará a modo offline automáticamente

**2. Empleados no aparecen en búsqueda**

La búsqueda no retorna resultados

**Solución**:

* Verificar que el empleado esté activo (es\_activo = true)
* En modo offline: Sincronizar para actualizar cache
* Verificar permisos RLS en proyecto de empleados

**3. Fotos no se suben**

Error al subir foto de evidencia

**Solución**:

* La compresión es automática
* En web: Verificar formato de imagen (JPG/PNG)
* En offline: Las fotos se subirán al sincronizar

**4. Sanción no cambia de estado**

El botón de aprobar no funciona

**Solución**:

* Verificar rol del usuario (debe ser gerencia/admin)
* La sanción debe estar en estado 'enviado'
* En offline: El cambio se aplicará al sincronizar

**5. Error de sincronización**

Las sanciones offline no se sincronizan

**Solución**:

// Forzar sincronización manual

final offlineManager = OfflineManager.instance;

await offlineManager.syncNow();

// Verificar operaciones pendientes

final stats = offlineManager.getOfflineStats();

print('Pendientes: ${stats['pending\_sync']}');

**Comandos de Debug**

// En HomeScreen, activar modo debug:

// Configuración > Herramientas de Desarrollo

// 1. Test de persistencia Hive

\_testPersistence();

// 2. Verificar cache de empleados

\_verificarCacheDetallado();

// 3. Test modo offline

\_runOfflineTest();

// 4. Forzar sincronización

OfflineManager.instance.syncNow();

// 5. Limpiar cache (cuidado!)

OfflineManager.instance.clearOfflineData();

**Logs Importantes**

// main.dart - Inicialización

🚀 Iniciando Sistema de Sanciones INSEVIG...

✅ Hive inicializado correctamente

📊 Estado inicial: X empleados en cache

// Modo offline

🔌 OFFLINE: Usando cache local

📱 Guardado en cola de sincronización

// Sincronización

🔄 INICIANDO SINCRONIZACIÓN...

✅ X operaciones sincronizadas exitosamente

// Errores comunes

❌ Error de conexión: SocketException

❌ Error Supabase: Invalid JWT

❌ Error sincronización: Timeout

**📈 Métricas y Monitoreo**

**Estadísticas Disponibles**

// Desde HomeScreen

{

'total': 150, // Total sanciones

'borradores': 5, // En borrador

'enviadas': 10, // Esperando gerencia

'aprobadas': 120, // Aprobadas

'rechazadas': 15, // Rechazadas

'pendientes': 30, // Pendientes resolver

'ultimoMes': 45, // Último mes

'porTipo': { // Por tipo de sanción

'FALTA': 20,

'ATRASO': 50,

...

}

}

**Estado Offline**

// OfflineManager.getOfflineStats()

{

'mode': 'offline', // online/offline

'empleados\_cached': 1250, // Empleados en cache

'sanciones\_cached': 45, // Sanciones locales

'pending\_sync': 3, // Operaciones pendientes

'last\_sync': '2024-01-15', // Última sincronización

}

**🚀 Siguientes Pasos**

**Para trabajar con archivos individuales:**

1. **Identificar la capa**: ¿Es UI, Service, Repository o Model?
2. **Verificar dependencias**: ¿Qué otros archivos importa?
3. **Considerar offline**: ¿Necesita soporte offline?
4. **Revisar el flujo**: ¿En qué parte del flujo participa?

**Archivos más modificados:**

1. **Nuevas pantallas**: Heredar estructura de existing screens
2. **Nuevos widgets**: Seguir patrón de widgets/
3. **Nueva funcionalidad**: Agregar a service → repository
4. **Offline**: Siempre a través de repositories

**Convenciones del proyecto:**

* **Idioma**: Comentarios y logs en español
* **Null Safety**: Proyecto usa null safety estricto
* **State Management**: Provider para estado global
* **Estilos**: Material Design con colores INSEVIG
* **Iconos**: Emojis para mejor UX

**📞 Contacto y Soporte**

* **Desarrollo**: Pereira Systems
* **Cliente**: INSEVIG
* **Año**: 2025
* **Basado en**: Aplicación Kivy original

Esta documentación te proporciona todo el contexto necesario para trabajar con cualquier archivo individual del sistema. Cada sección está diseñada para ser una referencia rápida cuando necesites entender cómo funciona una parte específica del sistema.