**INFORME DE AVANCE DEL PROYECTO - SISTEMA DE CONTROL DE MULTAS CIUDADANAS**

Fecha del reporte: 7 de octubre de 2025

**1. ¿EN QUÉ ESTÁ EL PROYECTO?**

El proyecto está en desarrollo activo. Estamos trabajando en un sistema completo de control de multas ciudadanas que permite a los usuarios gestionar infracciones, hacer acuerdos de pago, y recibir notificaciones automáticas. La última semana (hace 5-6 días) hemos estado enfocados en arreglar la inicialización de datos, modificar rutas, y realizar ajustes en las migraciones de base de datos. También integramos cambios importantes del módulo de anexar multas con correcciones y acuerdos de pago.

**2. ¿QUÉ HEMOS LOGRADO IMPLEMENTAR?**

Sistema de Autenticación Completo

Login con usuario y contraseña tradicional

Login con Google (OAuth)

Sistema de verificación por correo electrónico con códigos de verificación

Recuperación de contraseña por email

Tokens JWT con refresh tokens para mantener la sesión activa

Control de sesiones con tiempo de inactividad y tiempo máximo de sesión

Sistema completo de roles y permisos (quién puede hacer qué en el sistema)

Gestión de Infracciones y Multas Ciudadanas

Registro de documentos de infracción (multas ciudadanas de todo tipo)

Consulta de multas por número de documento de la persona

Cálculo automático de valores de multas basado en el SMLDV (Salario Mínimo Legal Diario Vigente)

Reportes de inspectoría con generación de PDFs

Sistema para anexar nuevas multas a infracciones existentes, con notificación automática por email

Nuevo: Control de fecha y estado "completado" al anexar multas

Acuerdos de Pago

Esta es una de las funcionalidades más robustas que hemos desarrollado:

Los ciudadanos pueden hacer acuerdos de pago para diferir sus multas

El sistema calcula intereses automáticamente (2% mensual configurable)

Prorratea intereses por días cuando es necesario

Tiene un período de gracia de 30 días antes de enviar a cobro coactivo

Genera PDFs automáticos del acuerdo de pago

Envía emails automáticos cuando se crea un acuerdo o se anexan multas

Integración completa con el módulo de anexar multas

Sistema de Descuentos en Tiempo Real

WebSockets implementados para notificar descuentos en tiempo real

SignalR para comunicación bidireccional entre servidor y clientes

Los ciudadanos reciben notificaciones instantáneas cuando hay descuentos disponibles en sus multas

Notificaciones Automáticas

Sistema de envío de emails configurado con Gmail SMTP

Bot de Telegram integrado para enviar notificaciones

Programador de tareas (Scheduler) que envía correos automáticos a horas específicas

Actualmente configurado para enviar a las 4:21 PM (hora de Bogotá)

Notificaciones cuando se anexan multas nuevas

Notificaciones de acuerdos de pago

Integración con APIs Externas

Gateway API para consumir datos externos

API pública para exponer información de atracciones turísticas de Colombia

Servicio de RECAPTCHA de Google para proteger formularios contra bots

Arquitectura y Patrones de Diseño

El proyecto está muy bien organizado:

Patrón Strategy para manejar diferentes tipos de eliminación (lógica vs. física)

Patrón Repository para acceso a datos

Inyección de dependencias en todo el proyecto

Validaciones con FluentValidation para todos los modelos

Separación en capas: Web (API), Business (lógica), Entity (modelos), Data (base de datos), Helpers, Utilities

Base de Datos Multi-Motor

El sistema es muy flexible:

Soporta SQL Server, PostgreSQL y MySQL

Puede migrar automáticamente al iniciar la aplicación

Todas las entidades tienen soporte para eliminación lógica (no se borra nada realmente, solo se marca como eliminado)

Sistema de auditoría automático

Módulos de Seguridad

Gestión completa de usuarios y personas

Sistema de roles personalizables

Permisos granulares por formulario y módulo

Tipos de usuario diferenciados

Parámetros del sistema (departamentos, municipios, tipos de documento, frecuencias de pago)

Documentación

Swagger/Swashbuckle integrado para documentar todas las APIs automáticamente

Cada endpoint está documentado y se puede probar desde el navegador

**3. TECNOLOGÍAS QUE ESTAMOS USANDO**

.NET 8.0 - La última versión estable de .NET

Entity Framework Core 9.0 - Para trabajar con las bases de datos

JWT Bearer Authentication - Para seguridad

SignalR - Para comunicación en tiempo real

Telegram.Bot - Para notificaciones por Telegram

FluentValidation - Para validar datos de entrada

Swagger - Para documentar las APIs

Docker - El proyecto está preparado para contenedores

3 bases de datos - SQL Server, PostgreSQL, MySQL (multi-motor)

**4. TRABAJO RECIENTE EN ESTA RAMA (ÚLTIMAS SEMANAS)**

Hace 5-6 días (trabajo más reciente):

Arreglamos la inicialización de datos (DataInit)

Modificamos rutas del sistema

Realizamos migraciones de base de datos

Integramos cambios de la rama cor/anex/acuer (correcciones, anexo y acuerdos)

Agregamos control de fecha y estado "completado" al anexar multas

Correcciones generales

Hace 2 semanas:

Implementamos el sistema de descuentos con WebSockets

Terminamos los acuerdos de pago con anexo de multas

Agregamos envío automático de emails cuando se anexan multas

Completamos el menú principal del sistema

Integramos cambios de la rama arregloModel

Hace 3 semanas:

Finalizamos la generación de PDFs para reportes y acuerdos

Implementamos el código de verificación por email

Desarrollamos todo el módulo de agregar multas

Creamos prototipos de nuevas funcionalidades

Correcciones varias en el sistema

Hace un mes:

Finalizamos los acuerdos de pago (incluyendo detalles finos)

Reestructuramos todo el backend para mejor organización

Implementamos el sistema de envío automático de correos programados

Agregamos método de consulta por número de documento

Correcciones en la entidad Person

**5. EQUIPO Y ORGANIZACIÓN**

El proyecto tiene varios desarrolladores trabajando en paralelo:

Rama rama-ingrid (actual) - Rama principal de desarrollo de Ingrid

Rama azure-rama - Preparando despliegue en Azure

Rama rama-daniel - Desarrollo de Daniel

Rama movil/rama-api-movil - API para la aplicación móvil

Rama recaptcha-camilo - Implementación de ReCAPTCHA

Rama cor/anex/acuer - Correcciones, anexos y acuerdos (ya integrada en rama-ingrid)

Rama arregloModel - Arreglos de modelos (ya integrada en rama-ingrid)

El trabajo de varios colaboradores como Santiago también está reflejado en los commits. Esto demuestra que el equipo está bien coordinado y trabajando en diferentes features en paralelo.

**6. CONFIGURACIÓN ACTUAL**

Bases de datos soportadas:

SQL Server (puerto 1433)

PostgreSQL (puerto 5433)

MySQL (puerto 3307)

Seguridad:

Sesión expira después de 1 minuto de inactividad (configurable)

Sesión máxima de 120 minutos

ReCAPTCHA con score mínimo de 0.5

Notificaciones:

Emails configurados vía Gmail

Envío programado diario a las 4:21 PM (hora de Bogotá)

Acuerdos de pago:

Interés del 2% mensual

Período de gracia de 30 días

Prorrateo diario de intereses

**7. ESTADO ACTUAL DE LA RAMA**

Estado del repositorio:

Rama No sincronizada con origin/rama-ingrid

No hay cambios pendientes por commitear

Base de código limpia y lista para seguir trabajando

Últimos merges integrados:

Rama arregloModel (hace 12 días)

Rama cor/anex/acuer (hace 6 días)

**8. ¿QUÉ FALTA POR HACER?**

Posiblemente hacer merge de rama-ingrid hacia main cuando esté todo listo

Continuar con pruebas unitarias (ya existe el proyecto de Tests)

Sincronizar cambios con otras ramas si es necesario

Documentar procesos de negocio específicos desarrollados

**9. ESTADO GENERAL**

El proyecto está en muy buen estado:

Arquitectura sólida y escalable

Código bien organizado y modular

Funcionalidades principales implementadas

Sistema de notificaciones robusto

Seguridad implementada correctamente

Múltiples bases de datos soportadas

Preparado para Docker

APIs documentadas

Rama actualizada y sincronizada

Áreas de oportunidad:

Mejorar mensajes de commits para mejor

Aumentar cobertura de pruebas

Finalizar documentación técnica