Universidad Mariano Gálvez

Ingeniería En Sistemas



Diego Andres Escobar Aguilar

Carnet: 09021 24 21379

**Proyecto Final 2: Documentación del Proyecto - Jutiapa Community App**

La aplicación *Jutiapa Community App* es una herramienta innovadora de análisis comunitario que integra tecnologías de mapeo con inteligencia artificial para generar recomendaciones de desarrollo sostenible. Tuve esta idea porque me gustaba la idea de usar un mapa y un llm. Esta solución permite:

* Selección precisa de áreas geográficas del municipio de Jutiapa, Guatemala mediante un sistema de mapas interactivo.
* Definición de características demográficas como tipo de zona (urbana/rural) y tamaño poblacional.
* Generación automatizada de análisis que identifica problemas clave, sugiere inversiones y propone proyectos sostenibles.

**Objetivos Principales**

* Proporcionar análisis comunitario accesible y de bajo costo.
* Integrar tecnologías de mapeo con modelos de lenguaje avanzados.
* Generar recomendaciones prácticas para el desarrollo local.
* Crear una plataforma escalable para análisis urbanos y rurales.

**Tecnologías Clave**

* **Plataforma Principal:** .NET 8.0
* **Interfaz de Usuario:** Windows Forms
* **Componente de Mapas:** WebView2 (Microsoft Edge Chromium)
* **Bibliotecas de Mapas:**
  + Leaflet.js (v1.9.4)
  + Leaflet.draw (v1.0.4)
* **Inteligencia Artificial:**
  + Groq API
  + Modelo Llama 3 (70B)
* **Manejo de Datos:**
  + GeoJSON
  + System.Text.Json

**Patrones de Diseño**

* **Modelo-Vista-Controlador (implícito):** Separación entre la interfaz (*MainForm*), la lógica (*GroqService*) y los datos (*GeoJSON*).
* **Programación asíncrona:**

private async void InitializeAsync()

{

await InitializeWebView2Async();

InitializeMap();

\_groqService = new GroqService();

}

* **Inyección de dependencias:** Configuración mediante appsettings.json y creación explícita de servicios.

**Desafíos Técnicos y Soluciones**

1. **Inicialización de WebView2**
   * *Problema:* El mapa no se cargaba correctamente durante la inicialización.
   * *Solución:*

await InitializeWebView2Async();

InitializeMap();

* *Lección:* Los componentes externos requieren manejo asíncrono adecuado.

1. **Respuestas JSON Inválidas**
   * *Problema:* El modelo Llama 3 8B generaba JSON mal formado.
   * *Soluciones:*
     + Implementación de logging avanzado:

SaveDebugFile("groq\_response.json", jsonResponse);

- Cambio a modelo más potente (Llama 3 70B):

private readonly string \_model = "llama3-70b-8192";

- Corrección automática de errores:

private string FixJsonErrors(string json)

{

json = Regex.Replace(json, @",\s\*([}\]])", "$1");

// Otras correcciones

}

* *Lección:* Modelos más grandes gestionan mejor tareas complejas. El logging detallado es esencial.

1. **Seguridad de Credenciales**
   * *Solución:* Uso de archivo de configuración:

{

"GroqConfig": {

"ApiKey": "tu\_clave\_privada"

}

}

**Lecciones Aprendidas**

* La secuencia de inicialización es crítica para componentes como WebView2.
* Modelos LLM grandes (70B) ofrecen mejores resultados, aunque requieren más recursos.
* Validar siempre las respuestas de APIs externas.
* Implementar mecanismos de corrección y fallback.
* El registro de depuración (debug logs) facilita el diagnóstico.

**Futuras Mejoras**

* **Integración con Google Maps API:** Obtener datos de negocios y reseñas para enriquecer el análisis.
* **Sistema de caché:** Mejorar rendimiento evitando solicitudes repetidas innecesarias.

Al final lo que más me consumió tiempo fue el segundo error, pasar varias horas batallando, pero al final era tan simple como cambiar de modelo a uno más potente.  
  
 También este proyecto fue económico porque no me gasté dinero en todo lo que hice, me puse como objetivo eso, al final lo logre.