# Trabajo Final: Análisis Inteligente de la Demanda Energética

**1. Presentación del Grupo**

Este proyecto ha sido desarrollado de manera individual, asumiendo la responsabilidad integral del ciclo de vida del software (SDLC). El rol ha abarcado desde el diseño de una arquitectura N-Tier y el modelado de datos en PostgreSQL, hasta la implementación del backend con Python/FastAPI, el frontend con React, y la orquestación de servicios de Inteligencia Artificial.

* **Rol:** Desarrollador Full Stack & Arquitecto de Soluciones.

**2. Presentación del Proyecto**

**Nombre de la Aplicación:** Análisis Inteligente de la Demanda ⚡🤖

Este proyecto consiste en una plataforma web de "Inteligencia Activa" diseñada para la gestión y análisis avanzado del consumo energético. Más que un simple visualizador, el sistema actúa como un auditor virtual que opera 24/7.

La aplicación permite transformar datos brutos en ahorro tangible. Al cargar históricos de consumo y establecer líneas base de comportamiento (año base), el sistema utiliza la Inteligencia Artificial de Google (Gemini) para comparar curvas de carga reales contra comportamientos históricos esperados. El objetivo central es detectar automáticamente anomalías, consumos fantasma y cambios de hábitos operativos, generando valor a través del ahorro directo y el mantenimiento predictivo.

**3. Problemática Principal**

A pesar de la modernización con medidores inteligentes, la gestión energética eficiente se enfrenta a barreras críticas que generan "costos ocultos":

1. **Dificultad para definir la "Normalidad":** El consumo es dinámico (varía por hora, clima y producción). Sin una línea base contextual, es imposible distinguir si un consumo de 100 kWh es rutinario o una fuga costosa.
2. **Brecha de Interpretación (Datos vs. Decisión):** Los sistemas tradicionales (SCADA) muestran *qué* pasó, pero no *por qué*. El análisis de causa raíz es manual, reactivo y lento; para cuando se detecta el problema, el dinero ya se ha perdido.
3. **Ceguera ante Cambios de Hábitos:** Las alertas estáticas por umbral (ej. "avisar si > 200kW") son ineficaces para detectar ineficiencias acumulativas, como encender la climatización una hora antes de lo necesario o fallos sutiles en maquinaria.

**4. Propuesta de Solución**

La solución implementada es una plataforma de **Auditoría Energética Asistida por IA** que automatiza el análisis cognitivo en tres niveles:

1. **Comparativa Contextual (Matemática):** El backend construye una curva de consumo "esperado" basada en patrones históricos, comparándola punto a punto con la realidad para identificar desviaciones objetivas.
2. **Diagnóstico con IA Generativa:** Se integra el modelo **Google Gemini 2.0 Flash** para interpretar las discrepancias. El sistema "lee" las curvas y genera diagnósticos en lenguaje natural, explicando la causa probable (ej. "Pico inusual asociado a climatización fuera de horario").
3. **Proactividad:** El sistema convierte la detección de anomalías en recomendaciones operativas claras, permitiendo pasar de un mantenimiento reactivo a uno anticipatorio.

**5. Alcance de la Solución**

El alcance actual del proyecto (MVP) establece una arquitectura sólida para el crecimiento:

* **Gestión de Datos Robusta:** Carga masiva de históricos (CSV) y persistencia en base de datos relacional, compatible con cualquier estructura de medidor.
* **Visualización de Alto Impacto:** Dashboard interactivo que superpone visualmente la "brecha" entre el consumo real (lo pagado) y el ideal (lo esperado).
* **Análisis Explicativo Automatizado:** Generación de reportes que clasifican el estado del día (Normal, Alerta, Crítico) utilizando el **Patrón de Diseño Observer**, lo que permite desacoplar el análisis de las notificaciones y alertas.
* **Arquitectura Escalable:** Diseño modular N-Tier (separación estricta entre Frontend, Backend y Datos) preparada para escalar a múltiples usuarios y grandes volúmenes de datos.

*Limitaciones actuales:* Operación en modo "offline" (carga de archivos) sin conexión streaming MQTT en esta fase.

**6. Presentación de las Herramientas**

Se seleccionó un stack tecnológico moderno, priorizando el rendimiento, el tipado estático y la escalabilidad:

Diagrama

El contenido generado por IA puede ser incorrecto.

**Frontend (Capa de Presentación):**

* **React + Vite:** Seleccionado por su velocidad de desarrollo (HMR) y renderizado eficiente.
* **TypeScript:** Garantiza un código robusto y mantenible mediante tipado estático.
* **Recharts:** Librería especializada para renderizar curvas de carga complejas con alto rendimiento.
* **TailwindCSS & Lucide:** Para una interfaz limpia, responsiva y consistente.

**Backend (Capa de Negocio):**

* **Python 3.10+ & FastAPI:** Framework asíncrono elegido por su alta velocidad en la gestión de APIs y validación de datos (vía Pydantic).
* **Pandas:** Motor de cálculo para la manipulación de series temporales y limpieza de datos CSV.
* **SQLAlchemy:** ORM para abstraer y gestionar las transacciones con la base de datos de forma segura.

**Base de Datos & IA:**

* **PostgreSQL:** Base de datos relacional robusta, ideal para almacenar series temporales y millones de registros de mediciones.
* **Google Gemini 2.0 Flash:** Modelo multimodal optimizado para velocidad y razonamiento lógico, integrado vía google-generativeai.

**7. Presentación de la Solución (Funcionamiento)**

El flujo de trabajo se ha diseñado para ser intuitivo y potente:

1. **Ingesta y Normalización:** El usuario carga los datos históricos. El backend procesa, limpia y almacena la información estructurada en PostgreSQL.
2. **Configuración del Análisis:** El usuario define el "Año Base" para entrenar el modelo de comportamiento y la "Fecha Objetivo" a auditar.
3. **Diagnóstico Inteligente (Pipeline):**
   * El sistema recupera los datos y calcula las diferencias.
   * Se genera una gráfica visual donde el **Área Azul** (Real) se contrasta con la **Línea Punteada** (Esperado).
   * La IA analiza el contexto y devuelve un veredicto textual en el panel "Diagnóstico Observer", sugiriendo acciones inmediatas para corregir desviaciones.

**8. Proyecciones Futuras**

Para evolucionar el MVP hacia un producto comercial completo, se plantean las siguientes fases:

* **Integración IoT en Tiempo Real:** Conexión vía protocolos MQTT/Modbus para análisis en vivo, eliminando la dependencia de archivos CSV.
* **Predicción de Facturación:** Uso de modelos predictivos para estimar el costo financiero a fin de mes antes de que llegue la factura.
* **Soporte Multi-Energía:** Extensión del modelo de datos (models.py) para soportar medidores de Agua y Gas.
* **Sistema de Notificaciones Push:** Implementación de nuevos "Observers" en el backend para enviar alertas al móvil de los técnicos de mantenimiento ante anomalías críticas.