¡Entendido, Giancarlo! 🙌  
Vamos a **priorizar la división y el uso de Playground/API** para **recuperar velocidad ya mismo**.

**🔹 1. Objetivo**

* Dividir el proyecto en **bloques claros**
* Usar **Playground** y/o **API de OpenAI** para evitar el lag por conversaciones largas
* Mantener **continuidad y documentación** con **README diarios**

**🔹 2. División en Bloques**

Cada bloque será **una conversación independiente** con su propio contexto.  
Así, la interfaz no carga 20 000 líneas y responde **rápido como el fin de semana**.

**Bloque 1 – Base de Datos y Seeds**

* init\_dev\_db
* seed\_inicial.py con roles y cláusulas
* Tests en scripts/test\_relaciones\_asociacion.py

**Bloque 2 – Dashboard y Blueprints**

* base.html con Bootstrap local
* Navbar + dashboard simple con conteos de usuarios y tareas

**Bloque 3 – Seguridad y Roles**

* Flask-Login
* Control de acceso por rol
* Redirección a dashboard correspondiente

**Bloque 4 – Documentos y Librería**

* CRUD de escrituras y actas
* Selección de cláusulas y puntos
* Flujo de avisos y entregas

✅ Cada bloque tendrá su **README diario**, para mantener la continuidad.

**🔹 3. Playground y API**

Para garantizar **fluidez**, trabajaremos así:

**Opción 1 – Playground**

1. Ingresar a [OpenAI Playground](https://platform.openai.com/playground)
2. Seleccionar:
   * **Model**: gpt-4o o gpt-4.1
   * **Mode**: Chat
3. Copiar **solo el contexto del bloque actual**
4. Usar el Playground para prompts largos → Responde **sin lag**

**Opción 2 – API con Python**

1. Instalar SDK:

pip install openai

1. Crear chat\_dev.py:

from openai import OpenAI

client = OpenAI(api\_key="TU\_API\_KEY")

prompt = """

# Contexto

Estamos trabajando en el Bloque 2: Dashboard y Blueprints...

"""

response = client.chat.completions.create(

model="gpt-4o",

messages=[{"role": "user", "content": prompt}]

)

print(response.choices[0].message.content)

1. Resultado → se puede **guardar en .md automáticamente** para tu documentación diaria.

**🔹 4. Pasos para hoy**

1. **Crear una nueva sesión** para **Bloque 1**
   * Solo contexto de DB y seeds
   * Trabajamos rápido sin arrastrar toda la historia
2. **Configurar Playground/API**
   * Para que los bloques siguientes sean ligeros
3. **Guardar un README** al final de cada bloque
   * Así nunca perdemos información

Si quieres, **ahora mismo** puedo guiarte para:

1. Abrir **nueva sesión Bloque 1**
2. Configurar Playground/API paso a paso
3. Probar la fluidez sin lag

¿Quieres que primero configuremos **Playground/API** o que iniciemos ya **Bloque 1** en sesión nueva?

Te entiendo, Giancarlo 🙏.  
Si no queremos arriesgar pagos, la estrategia más segura es:

**🔹 1. Seguir en ChatGPT Plus (como ahora)**

* Sin habilitar Playground/API
* **No hay cargos extra**
* Solo necesitamos **segmentar las sesiones** para evitar el lag

**🔹 2. Estrategia de Segmentación**

Vamos a dividir el proyecto en **4 bloques**, cada uno en **una conversación separada**:

1. **Bloque 1 – Base de Datos y Seeds**
   * init\_dev\_db y seed\_inicial.py
   * Tests de relaciones y roles
2. **Bloque 2 – Dashboard y Blueprints**
   * base.html limpio con Bootstrap local
   * Navbar y dashboard inicial
3. **Bloque 3 – Seguridad y Roles**
   * Flask-Login
   * Control de permisos por rol
   * Redirecciones a dashboards por rol
4. **Bloque 4 – Documentos y Librería**
   * CRUD de escrituras y actas
   * Selección de cláusulas y puntos
   * Flujo de avisos y entregas

✅ Cada bloque tendrá su **README diario**  
✅ Mantendremos la continuidad sin arrastrar 20k líneas

**🔹 3. Flujo Diario**

1. Abrimos **nueva conversación** para el bloque actual
2. Copiamos el contexto mínimo:
   * Resumen de lo ya logrado
   * Archivos relevantes para ese bloque
3. Trabajamos el bloque hasta completarlo
4. Generamos **README diario** con:
   * Avances
   * Código generado
   * Errores y soluciones
   * Pendientes para el siguiente bloque

**🔹 4. Próximo Paso**

1. Cerramos esta sesión
2. Abrimos **Bloque 1 – Base de Datos y Seeds**
3. Trabajamos allí sin arrastrar este historial pesado → **fluidez recuperada**

¿Quieres que **ahora mismo preparemos el template de README** y **el resumen inicial para Bloque 1**, para arrancar en la nueva conversación ya optimizada?