Documento de Especificaciones Técnicas: "Construcción de IA Prompt"

1. Visión General

Desarrollar una estación de trabajo web avanzada que transforme ideas o imágenes en prompts fotográficos ultrarrealistas y sugerencias de animación para generar un video, gestionando una base de datos de prompts favoritos.

---

2. Arquitectura de Modelos (Prompt Chaining)

La aplicación utilizará una estructura de tres capas para maximizar la calidad y evitar la censura:

1. **Motor de Visión (Modelo B):** Tal como se menciona en el documento GEM, si existe una imagen a analizar, este deberá ser enviada a un motor de visión, el cual deberá analizar imágenes subidas y las convierte en texto técnico, devolviendo descripción de la imagen.
2. **Motor Lógico Maestro (Modelo A):**
    1. Aplica el documento GEM en el caso de ser requerido la construcción de un prompt solo con una idea, donde se le enviara la idea para ser convertida en un prompt más completo como lo solicita en GEM, y podría incluir la gestión de los personajes (Andy/Cony), si fue solicitada dicha información.
    2. Aplica el documento GEM en el caso de ser requerido la construcción de un prompt con imagen, donde la información de la imagen propiamente tal es la respuesta del motor de visión (modelo B), más la gestión de los personajes (Andy/Cony/o ambas) si es requerido y esta es enviada al modelo A, el que generará el prompt final.
3. **Módulo de Animación:** El resultado del modelo A deberá ser enviado a un nuevo modelo capaz de tomar esta información y modificarla de forma tal de generar un prompt para una animación atractiva de este resultado y que deberá ser compatible con modelos de video como WAN 2.2, similares o superiores.

---

3. Flujo de Trabajo y Entradas (Inputs)

A. Entradas de Usuario:

- **Texto:** Idea libre.
- **Imagen:** Referencia visual (JPG/PNG).
- **Selector de Personaje:** [Andy] [Cony] [Ambos][Ninguno].

- **Selector de Estilo:** [Fotografía (Default)] [Ilustración] [Arte Digital][Acuarela]. (Combobox, con diferentes tipos de estilos, variados y adicionales a los ya mencionados)

B. Lógica Interna:

1. **Si hay Imagen:** Esta, se envía al **Modelo B**. La salida técnica o respuesta del modelo B, se usa como contexto para el Modelo A.
2. **Manejo de Variables de Personaje:** El sistema debe inyectar las fichas técnicas:
    - **Variable {Andy}:** 29 años, rubia, ojos azules, física atlético-femenino, sensual, alegre.
    - **Variable {Cony}:** 21 años, latina, piel canela, ojos verdes, cabello negro, física atlético-femenino, mirada seductora.

---

4. Esquema de Prompts (System Prompts)

Prompt para el Modelo B (Análisis de Imagen):

"Describe esta imagen con enfoque fotográfico: Pose exacta, prendas de vestir (textura/color), entorno detallado, iluminación, clima y composición de cámara. Sé técnico y directo." Más detalles en documento GEM.

Prompt para el Modelo A (El GEM Maestro):

El desarrollador debe integrar el **GEM completo** que definimos, asegurando que:

1. Use terminología artística/anatómica neutral (ej. "curvatura del busto").
2. Aplique configuraciones de cámara (Lente 35mm, 85mm, ISO, Apertura).
3. Descripción de personajes (Andy/Cony) en caso de ser requerido
4. Respuesta del Modelo B, de ser requerido.
5. Genere la salida en un solo párrafo compacto.

---

5. Salidas Requeridas (Outputs)

La web debe mostrar tres resultados por cada petición:

1. **Prompt Fotográfico (Español):** El párrafo detallado según el GEM.
2. **Prompt Fotográfico (Inglés):** Traducción exacta del anterior.
3. **Sugerencia de Animación (WAN/Luma):**
    - *Regla:* "Describe el movimiento de cámara (Zoom, Pan, Tilt) y la acción del sujeto (viento en el cabello, parpadeo, caminar hacia la cámara)".

---

6. Módulo de Base de Datos (Excel/CSV)

El sistema debe integrarse con el archivo Nuevos-Prompt.csv:

- **Lectura:** Mostrar en una tabla los prompts ya existentes en el archivo.
- **Escritura:** Un botón "Guardar en Favoritos" que añada el nuevo prompt generado al archivo CSV con las columnas: ID, Título, Personaje, Prompt_ES, Prompt_EN, Prompt_Video.
- **Buscador:** Filtrar por palabras clave dentro de los prompts guardados.

---

7. Requerimientos de Interfaz (UI/UX) o Pantalla requerida.

- **Dashboard:** Panel Inferior para ver los favoritos (base de datos, archivo CSV) y panel central / superior, para generación de los prompt.
- **El Dashboard:** El generador deberá contener:
    - **Textbox;** para ingresar ideas a desarrollar.
    - **Label1:** bajo este textbox (lbel pequeño que indique el nombre del modelo a utilizar)
    - **Chexbox:** para seleccionar Personajes, Andy, Cony, Ambas, Ninguna
    - **Combobox;** Para seleccionar estilo; por defecto Ultrarealismo.
    - **Botón;** para subir imágenes
    - Visor de Imagen: Si se sube una imagen, mostrar una miniatura
    - **Label2:** bajo este Visor (label pequeño que indique el nombre del modelo a utilizar por el visor)
    - botón Enviar información, el que desencadena todo el trabajo

  A continuación del Generador, deberán existir:

- 3 Textbox, que mostrarán el resultado del modelo A. El primero muestro el prompt en español, el segundo en inglés, y el tercero el prompt para video
- Controles de Copiado**:** Botones asociados a cada textbox de un solo clic para copiar el prompt del resultado en español, en inglés o del video por separado.
- Botón agregar a favoritos**;** El que producirá, que estas respuestas sean añadidas al archivo CSV

  El panel inferior, deberá incluir buscador en la tabla que busque alguna palabra coincidente en cualquier campo, que, al dar encontrar coincidente, deberá destacar

la fila. En caso de no encontrar coincidente, indicar mediante alerta, que no hay resultados para la búsqueda. Si existe más de un resultado, se deberá destacar la primera coincidencia y permitir saltar luego a las otras.

---

8. Notas para el Desarrollador (Stack Sugerido)

- **Backend:** Python con FastAPI o Streamlit (para prototipado rápido) o similar que permita el desarrollo eficiente del backend.
- **Conectividad:** API de OpenRouter (permite intercambiar Qwen2-VL, Llama 3 y Mistral fácilmente) u con otros ambientes que permitan interactuar con modelos tanto de visión como de construcción de prompt. Puede adicionalmente conectarse con un ambiente para ocupar un modelo, y con otro ambiente para ocupar otro modelo. Por ejemplo OpenRouter, para ocupar Modelo B (modelo capaz de vision) y luego Gemini, para un modelo A, o viceversa.
- **Seguridad:** Las API Keys deben manejarse mediante archivos .env en el servidor, nunca en el cliente. Este archivo maneja más datos, como API, modelos a utilizar, reintentos, interfaz, perfiles.
- **Procesamiento de CSV:** Usar la librería Pandas o similares, para la gestión de la base de datos de prompts.
- Deberá existir manejo de errores, que capturen tanto errores de la propia aplicación como respuestas de los distintos modelos utilizados.
- Interfaz amigable con el usuario, de preferencia tonos oscuros que ayuden al descanso de la vista.

9. Estrategia de Resiliencia: "Multi-Model Fallback"

Para asegurar que tu estación de trabajo no se detenga si una API falla, debes implementar un sistema de **conmutación por error (failover)**.

El flujo lógico recomendado es:

1. **Reintentos Exponenciales:** Si el Modelo A (por ejemplo, Gemini) no responde, el sistema debe intentar 3 veces con una pausa que aumente gradualmente (ej. 2s, 4s, 8s).
2. **Cambio de "Tier" (Modelo de Respaldo):** Si tras los reintentos persiste el error, el sistema debe cambiar automáticamente a un "Modelo Nuevo" de respaldo que tenga capacidades similares. Actualizando el label respectivo con el nombre del nuevo modelo.
3. **Alertas Visuales:** Si todos los modelos fallan, se debe disparar la **Alerta de Error (Rojo Lava)** con el efecto *glow* y *backdrop-blur* para informar al usuario que el

servicio está temporalmente caído. Si el archivo .env, no se encuentra debe salir un mensaje de error que dispare tu **Alerta Roja Lava** (del documento de gráfica) indicando: "Configuración no encontrada".

Arquitectura sugerida:

- **Modelo A (Lógica):** Gemini 1.5/2.0 Pro (Principal) -> Grok 3 (Respaldo).
- **Modelo B (Visión):** Qwen2-VL, Llama 3 y Mistral
- **Módulo Video:** Wan 2.2.

Estructura .env

```
# CONFIGURACIÓN DE APIS (Credenciales)
OPENROUTER_API_KEY=clave
GEMINI_API_KEY=clave
OPENAI_API_KEY=clave

# CONFIGURACIÓN DE MODELOS (Aquí es donde cambias versiones sin programar)
MODELO_A_LOGICO=google/gemini-2.0-pro-exp:free
MODELO_B_VISION=qwen/qwen-2-vl-72b-instruct
MODELO_ANIMACION=wan/wan-2.1-t2v-480p

# PARÁMETROS DE REINTENTOS
MAX_RETRIES=3
RETRY_DELAY_SECONDS=2

# INTERFAZ
MODO_ESTETICO=oscuro_lava

#PERFILES
PERFIL_ANDY="29 años, rubia, ojos azules, física atlético-femenino, sensual, alegre"
```

PERFIL_CONY="21 años, latina, piel canela, ojos verdes, cabello negro, física atlético-femenino, mirada seductora"