Backend - PDF a Audio

> **Advertencia:** Actualmente este backend solo funciona en Windows. El soporte para otras plataformas podría añadirse en el futuro.

Este backend está construido con Flask y expone un endpoint para convertir archivos PDF a texto y luego a audio usando OCR y edge-tts.

- Python 3.13.2 (se recomienda usar el entorno virtual incluido en esta carpeta: `env`)
- Tesseract-OCR instalado en el sistema (y accesible desde la variable de entorno `TESSERACT_CMD`)

Instalación y ejecución local

1. Navega a la carpeta `backend`:

```
cd backend
```

2. Crea un entorno virtual:

```
python -m venv env
env\Scripts\activate  # En Windows
# o
source env/bin/activate  # En Linux/Mac
```

3. Instala las dependencias:

```
pip install -r requirements.txt
```

4. Crea un archivo `.env` con la ruta a tu ejecutable de Tesseract **y la ruta a ffmpeg**:

```
```

```
TESSERACT_CMD="C:\\Ruta\\A\\Tesseract-OCR\\tesseract.exe"
TESSDATA_PREFIX="C:\\Ruta\\A\\Tesseract-OCR\\tessdata"
FFMPEG_PATH="C:\\Ruta\\A\\ffmpeg\\bin\\ffmpeg.exe"
```

Manejo robusto de fragmentos y errores

- El backend divide el texto en fragmentos de hasta 3000 caracteres, cortando siempre en el punto final más cercano.
- Si un fragmento da error, el sistema intenta limpiarlo (eliminando caracteres no imprimibles y espacios redundantes) y reintenta.
- Si sigue fallando, subdivide el fragmento en partes más pequeñas y vuelve a intentar.
- Si después de todos los intentos (limpieza y subdivisión) algún fragmento o subfragmento no puede ser procesado, el proceso se detiene y muestra el error, sin continuar con los siguientes fragmentos.
- Esto garantiza que el audio generado sea lo más completo posible y que cualquier problema real se reporte de inmediato para su corrección.

5. Ejecuta el servidor:

```
python app.py
```

El backend estará disponible en `http://localhost:5000`.

Endpoint principal

- **POST** `/procesar`
  - **Body:** `form-data`
    - `pdf` (File, requerido): El archivo PDF a procesar
    - `lang` (Text, opcional): Idioma OCR (por defecto: spa)
    - `voice` (Text, opcional): Voz para el audio (por defecto: es-ES-ElviraNeural)
    - `out` (Text, opcional): Nombre del archivo de texto de salida
    - `audio` (Text, opcional): Nombre del archivo de audio de salida
  - **Respuesta:** JSON con los nombres de los archivos generados

---

✦ ✦

Despliegue en Render

- **Servicio:** Web Service
- **Root Directory:** `backend`
- **Build Command:** `bash build.sh`
- **Start Command:** `gunicorn app:app`

Notas

- Los archivos subidos se guardan en la carpeta `input/`.
- Los resultados se guardan en la carpeta `output/`.
- El backend acepta conexiones desde cualquier IP (`host=0.0.0.0`).

✦ ✦