PDFTranslate

Versión 1.0

Pablo Caño

05 de septiembre de 2025

Contenidos:

CAPÍTULO 1

Documentación del Backend

El backend es una aplicación FastAPI que gestiona las solicitudes de traducción, el estado de las tareas y la descarga de archivos.

1.1 API Principal

Este módulo contiene la aplicación FastAPI, define todos los endpoints, los modelos de datos Pydantic y la configuración inicial de Celery. Módulo principal de la API para la traducción de documentos PDF.

Este archivo define la aplicación FastAPI, sus middlewares, modelos de datos Pydantic, y todos los endpoints necesarios para subir, traducir, monitorear y descargar documentos PDF. Utiliza Celery para el procesamiento asíncrono y S3/MinIO para el almacenamiento de archivos.

class backend.main.**PageTranslation**(**args: Any*, ***kwargs: Any*)

Bases: `BaseModel`

Contiene todas las traducciones de una única página.

page_number: int

translations: List[TranslationText]

class backend.main.**ProcessingStep**(**values*)

Bases: `str`, `Enum`

Enumera los pasos intermedios durante el procesamiento de una traducción.

COMBINING_PDF = 'Finalizando documento'

CONVERTING_PDF = 'Analizando páginas'

PROCESSING_PAGES = 'Traduciendo contenido'

QUEUED = 'Iniciando traducción'

UNKNOWN = 'Procesando'

UPLOADING_PDF = 'Preparando documento'

class backend.main.**TaskStatus**(**values*)

Bases: `str`, `Enum`

Enumera los posibles estados finales de una tarea de traducción.

COMPLETED = 'COMPLETED'

FAILED = 'FAILED'

PENDING = 'PENDING'

PROCESSING = 'PROCESSING'

class backend.main.**TranslationData**(**args: Any*, ***kwargs: Any*)

Bases: `BaseModel`

Estructura completa de los datos de traducción de un documento.

pages: List[PageTranslation]

class backend.main.**TranslationProgress**(**args: Any*, ***kwargs: Any*)

Bases: `BaseModel`

Modela el progreso de una tarea, indicando el paso actual y detalles.

details: Any | None = None

step: ProcessingStep = 'Iniciando traducción'

class backend.main.**TranslationTask**(**args: Any*, ***kwargs: Any*)

Bases: `BaseModel`

Modelo de respuesta completo para el estado de una tarea.

error: str | None = None

id: str

originalFile: str

progress: TranslationProgress | None = None

status: TaskStatus

translatedFile: str | None = None

class backend.main.**TranslationText**(**args: Any*, ***kwargs: Any*)

Bases: `BaseModel`

Representa un único fragmento de texto original y su traducción.

id: int

original_text: str

translated_text: str

class backend.main.**UploadResponse**(**args: Any*, ***kwargs: Any*)

Bases: BaseModel

Modelo de respuesta tras subir un archivo para traducir.

taskId: str

backend.main.**get_final_task_result**(*task_id: str*) → celery.result.AsyncResult | None

Sigue la cadena de tareas de Celery para encontrar el resultado final.

Algunas tareas (orquestadoras) inician otras. Esta función navega desde la tarea inicial hasta la tarea final que contiene el resultado real.

Parámetros
task_id (str) – El ID de la tarea de Celery inicial (orquestadora).

Devuelve
El objeto AsyncResult de la tarea final, o la tarea orquestadora si aún no ha terminado.

Tipo del valor devuelto
Optional[AsyncResult]

async backend.main.**get_original_pdf**(*task_id: str*)

Genera y redirige a una URL prefirmada para descargar el PDF original.

Parámetros
task_id (str) – El ID de la tarea asociada al PDF original.

Devuelve
Una redirección HTTP 307 a la URL de descarga del archivo.

Tipo del valor devuelto
RedirectResponse

Muestra
HTTPException – 404 si el archivo no se encuentra, 500 para otros errores.

async backend.main.**get_position_data**(*task_id: str*)

Obtiene los datos de posición de los textos en el PDF.

Devuelve un JSON con las coordenadas y dimensiones de cada fragmento de texto, necesario para la regeneración del PDF.

Parámetros
task_id (str) – El ID de la tarea.

Devuelve
Los datos de posición en formato JSON.

Tipo del valor devuelto
dict

Muestra
HTTPException – 404 si los datos no se encuentran.

async backend.main.**get_task_status**(*task_id: str*)

Consulta el estado y progreso de una tarea de traducción.

Este endpoint mapea el estado interno de Celery a un formato comprensible para el frontend, incluyendo el paso actual del proceso.

Parámetros
task_id (str) – El ID de la tarea a consultar.

Devuelve
Un objeto *TranslationTask* con el estado detallado de la tarea.

Tipo del valor devuelto
dict

Muestra
HTTPException – 500 si ocurre un error al consultar el estado.

async backend.main.**get_translated_pdf**(*task_id: str*)

Genera y redirige a una URL prefirmada para descargar el PDF traducido.

Verifica que la tarea se haya completado con éxito y que el archivo exista en S3. Luego, genera una URL de descarga temporal y segura.

Parámetros
task_id (str) – El ID de la tarea completada.

Devuelve
Una redirección HTTP 307 a la URL de descarga del archivo.

Tipo del valor devuelto
RedirectResponse

Muestra
HTTPException – 404 si la tarea o el archivo no se encuentran, 500 para otros errores.

async backend.main.**get_translation_data**(*task_id: str*)

Obtiene los datos de traducción (texto original y traducido) en formato JSON.

Este endpoint permite al frontend recuperar los datos de la traducción para mostrarlos en una interfaz de edición.

Parámetros
task_id (str) – El ID de la tarea.

Devuelve
Los datos de traducción en formato JSON.

Tipo del valor devuelto
dict

Muestra
HTTPException – 404 si los datos no se encuentran.

async backend.main.**health_check**()

Endpoint de verificación de salud del servicio.

Comprueba la conectividad con dependencias críticas, como el bucket S3.

Devuelve
Un JSON indicando el estado de salud del servicio.

Tipo del valor devuelto
dict

async backend.main.**root**()

Endpoint raíz que proporciona información básica de la API.

Devuelve
Un JSON con un mensaje de bienvenida y detalles de la configuración.

Tipo del valor devuelto
dict

async backend.main.**translate_pdf_endpoint**(*file: fastapi.UploadFile = fastapi.File*, *srcLang: str = fastapi.Form*, *tgtLang: str = fastapi.Form*, *languageModel: str = fastapi.Form*, *confidence: float = fastapi.Form*)

Recibe un archivo PDF y comienza una tarea de traducción asíncrona.

Valida que el archivo sea un PDF, lo lee en memoria y lo envía a una tarea de Celery para su procesamiento. Devuelve inmediatamente el ID de la tarea para su posterior consulta.

Parámetros

- **file** (UploadFile) – El archivo PDF a traducir.
- **srcLang** (str) – Idioma de origen (o "auto" para detección automática).
- **tgtLang** (str) – Idioma de destino.
- **languageModel** (str) – Modelo de lenguaje a utilizar para la traducción.
- **confidence** (float) – Nivel de confianza para el modelo.

Devuelve
Un objeto JSON con el *taskId* de la tarea iniciada.

Tipo del valor devuelto
UploadResponse

Muestra
HTTPException – 400 si el archivo no es PDF, 500 para errores internos.

async backend.main.**update_translation_data**(*task_id: str*, *translation_data:* TranslationData = *fastapi.Body*)

Actualiza los datos de traducción y lanza una tarea para regenerar el PDF.

Recibe los datos de traducción modificados, los guarda en S3 y luego inicia una tarea de Celery para crear un nuevo PDF con los textos actualizados.

Parámetros

- **task_id** (str) – El ID de la tarea a actualizar.
- **translation_data** (TranslationData) – El objeto con los datos de traducción actualizados.

Devuelve
Un mensaje de confirmación.

Tipo del valor devuelto
dict

Muestra
HTTPException – 404 si los datos de posición no se encuentran, 500 para otros errores.

1.2 Infraestructura de Almacenamiento

Este módulo abstrae toda la lógica de comunicación con el servicio de almacenamiento compatible con S3 (como MinIO), incluyendo la subida, descarga y generación de URLs seguras. Módulo de infraestructura para la interacción con el almacenamiento S3 (o compatible como MinIO).

Proporciona funciones de alto nivel para subir, descargar, verificar, eliminar y generar URLs prefirmadas para objetos en un bucket S3. La configuración del cliente se realiza de forma global al importar el módulo.

backend.src.infrastructure.storage.s3.**delete_prefix**(*prefix: str*)

Elimina todos los objetos en S3 que comiencen con un prefijo dado.

Es útil para limpiar todos los archivos asociados a una tarea. Maneja la paginación y la eliminación en lotes de 1000 objetos.

Parámetros
prefix (str) – El prefijo de las claves a eliminar (ej. "task-id/").

backend.src.infrastructure.storage.s3.**download_bytes**(*key: str*) → bytes

Descarga un objeto de S3 como bytes.

Parámetros
key (str) – La clave del objeto a descargar.

Devuelve
El contenido del objeto.

Tipo del valor devuelto
bytes

Muestra
Exception – Propaga cualquier excepción si el objeto no existe o hay un error.

backend.src.infrastructure.storage.s3.**ensure_bucket_exists**()

Verifica si el bucket configurado existe y lo crea si es necesario.

Usa una llamada *head_bucket* para comprobar la existencia, que es más eficiente que listar todos los buckets.

Muestra
ClientError – Si ocurre un error al contactar con el servicio S3 que no sea un "404 Not Found".

backend.src.infrastructure.storage.s3.**key_exists**(*key: str*) → bool

Comprueba de forma eficiente si una clave existe en el bucket.

Parámetros
key (str) – La clave a comprobar.

Devuelve
True si el objeto existe, False en caso contrario.

Tipo del valor devuelto
bool

Muestra
ClientError – Si ocurre un error de Boto3 que no sea un 404.

backend.src.infrastructure.storage.s3.**list_keys**(*prefix: str = ''*) → list[str]

Lista todas las claves en el bucket que coinciden con un prefijo.

Parámetros
prefix (str) – El prefijo a buscar. Si está vacío, lista todas las claves.

Devuelve
Una lista de strings, donde cada string es una clave de objeto.

Tipo del valor devuelto
list[str]

backend.src.infrastructure.storage.s3.**presigned_get_url**(*key: str*, *expires: int = 3600*, *inline_filename: str | None = None*, *content_type: str = 'application/pdf'*) → str

Genera una URL S3 prefirmada y la adapta a la URL pública si está configurada.

El proceso consiste en generar primero una URL interna (ej. http://minio:9000/...) y luego, si se ha definido una URL pública (AWS_S3_PUBLIC_ENDPOINT_URL), reemplaza el host y el puerto manteniendo la ruta y los parámetros de firma, resultando en una URL accesible desde el exterior.

Parámetros

- **key** (str) – La clave del objeto para el que se generará la URL.
- **expires** (int) – Duración de la validez de la URL en segundos.
- **inline_filename** (Optional[str]) – Si se proporciona, la URL incluirá cabeceras para que el navegador muestre el archivo en lugar de descargarlo, con el nombre de archivo especificado.
- **content_type** (str) – El tipo MIME del contenido, usado con *inline_filename*.

Devuelve
Una URL temporal y segura para acceder al objeto.

Tipo del valor devuelto
str

backend.src.infrastructure.storage.s3.**upload_bytes**(*key: str*, *data: bytes*, *content_type: str | None = None*)

Sube un objeto de bytes a una clave específica en S3.

Parámetros

- **key** (str) – La ruta completa (clave) donde se almacenará el objeto en el bucket.
- **data** (bytes) – El contenido del objeto en bytes.
- **content_type** (Optional[str]) – El tipo MIME del contenido (ej. “application/pdf”).

Muestra
Exception – Propaga cualquier excepción ocurrida durante la subida.

CAPÍTULO 2

Documentación del Worker

... Módulo principal de tareas Celery para el procesamiento de documentos PDF.

Este archivo define el flujo de trabajo asíncrono para la traducción de PDFs. Contiene la tarea orquestadora principal que divide el trabajo, las tareas que procesan lotes de páginas en paralelo, y la tarea finalizadora que ensambla el documento traducido.

Flujo de trabajo principal:

1. **process_pdf_document**: Orquesta todo el proceso. Convierte PDF a imágenes, crea lotes de páginas y lanza un *chord* de Celery.
2. **extract_and_translate_batch**: Tarea ejecutada en paralelo para cada lote. Realiza la detección de layout, OCR y traducción.
3. **assemble_final_pdf**: Tarea finalizadora del *chord*. Recopila los resultados de todos los lotes, construye el PDF final y guarda los metadatos.
4. **regenerate_pdf_from_storage**: Tarea independiente para regenerar un PDF a partir de datos de traducción previamente guardados y modificados.

worker.tasks.**assemble_final_pdf**(*self*, *results_from_batches: List[List[Dict[str, Any]]]*, *task_id: str*, *original_key: str*, *src_lang: str*, *tgt_lang: str*)

Tarea finalizadora que ensambla el PDF completo a partir de los resultados de los lotes.

Esta tarea se ejecuta una vez que todas las tareas *extract_and_translate_batch* de un *chord* han finalizado. Consolida los resultados, construye el PDF, genera y guarda los metadatos de traducción y posición.

Parámetros

- **self** – Instancia de la tarea de Celery.
- **results_from_batches** (`List[List[Dict[str, Any]]]`) – Una lista de listas, donde cada sublista es el resultado de una tarea de procesamiento de lote.
- **task_id** (`str`) – Identificador único de la tarea global.
- **original_key** (`str`) – Clave de S3 del PDF original.

- **src_lang** (str) – Código del idioma de origen.
- **tgt_lang** (str) – Código del idioma de destino.

Devuelve
Un diccionario con el estado final y las claves de S3 de los artefactos generados.

Tipo del valor devuelto
dict

worker.tasks.**build_translated_pdf**(*results_list: List[Dict[str, Any]]*, *task_id: str*, *target_language: str*) → bytes

Construye un documento PDF a partir de una lista de datos de página procesados.

Itera sobre los datos de cada página, dibuja las regiones de imagen recortadas y renderiza los párrafos de texto traducido en sus posiciones correspondientes.

Parámetros

- **results_list** (List[Dict[str, Any]]) – Lista de diccionarios, cada uno representando una página con sus regiones de texto, imagen y dimensiones.
- **task_id** (str) – El ID de la tarea, usado para descargar imágenes de página desde S3.
- **target_language** (str) – El código del idioma de destino para seleccionar la fuente adecuada.

Devuelve
El documento PDF completo como un objeto de bytes.

Tipo del valor devuelto
bytes

worker.tasks.**extract_and_translate_batch**(*self*, *task_id: str*, *page_batch_info: List[Tuple[int, str]]*, *src_lang: str*, *tgt_lang: str*, *language_model: str*, *confidence: float*)

Tarea de worker que procesa un lote de páginas.

Realiza la detección de layout, OCR y traducción para un conjunto de páginas. Está diseñada para ser ejecutada en paralelo por múltiples workers de Celery.

Parámetros

- **self** – Instancia de la tarea de Celery.
- **task_id** (str) – Identificador único de la tarea global.
- **page_batch_info** (List[Tuple[int, str]]) – Lista de tuplas *(page_number, storage_key)* para el lote.
- **src_lang** (str) – Código del idioma de origen.
- **tgt_lang** (str) – Código del idioma de destino.
- **language_model** (str) – Modelo de IA a utilizar.
- **confidence** (float) – Umbral de confianza para la detección de layout.

Devuelve
Una lista de diccionarios, cada uno conteniendo los datos procesados de una página (regiones de texto, imágenes, etc.).

Tipo del valor devuelto
List[Dict[str, Any]]

worker.tasks.**process_pdf_document**(*self*, *file_content: bytes*, *src_lang: str*, *tgt_lang: str*, *language_model: str = 'openai/gpt-4o-mini'*, *confidence: float = 0.45*)

Tarea orquestadora principal que inicia el flujo de traducción de un PDF.

Sube el PDF original, lo convierte en imágenes por página, agrupa las páginas en lotes y lanza un *chord* de Celery para procesar los lotes en paralelo. La tarea finalizadora del *chord* (*assemble_final_pdf*) se encargará de unir los resultados.

Parámetros

- **self** – La instancia de la tarea de Celery (inyectada por *bind=True*).
- **file_content** (bytes) – El contenido del archivo PDF en bytes.
- **src_lang** (str) – Código del idioma de origen.
- **tgt_lang** (str) – Código del idioma de destino.
- **language_model** (str) – Identificador del modelo de IA a usar para la traducción.
- **confidence** (float) – Umbral de confianza para la detección de layout.

Devuelve
Un diccionario con el estado del proceso y el ID de la tarea finalizadora.

Tipo del valor devuelto
dict

worker.tasks.**regenerate_pdf_from_storage**(*task_id: str*, *translation_data: dict*, *position_data: dict*)

Regenera un PDF a partir de datos de traducción y posición almacenados.

Esta tarea se utiliza cuando un usuario edita las traducciones a través de la interfaz. Recibe los textos actualizados y la información de layout original para reconstruir el PDF sin necesidad de un reprocesamiento completo (OCR, etc.).

Parámetros

- **task_id** (str) – Identificador único de la tarea.
- **translation_data** (dict) – Diccionario con los datos de texto (original y traducido).
- **position_data** (dict) – Diccionario con la información de layout (posiciones, dimensiones).

Devuelve
Un diccionario indicando el éxito y la clave de S3 del nuevo PDF.

Tipo del valor devuelto
dict

async worker.tasks.**translate_extracted_text**(*extracted_data: List[Dict[str, Any]]*, *tgt_lang: str*, *language_model: str*) → List[Dict[str, Any]]

Gestiona llamadas concurrentes a la API de traducción para un lote de páginas.

Utiliza *asyncio.gather* para realizar todas las solicitudes de traducción de un lote de forma paralela, mejorando significativamente el rendimiento.

Parámetros

- **extracted_data** (List[Dict[str, Any]]) – Lista de datos de página, cada una con regiones de texto extraídas.
- **tgt_lang** (str) – Código del idioma de destino.
- **language_model** (str) – Identificador del modelo de IA.

Devuelve
La misma estructura de datos de entrada, pero con el campo *translated_text* añadido a cada región de texto.

Tipo del valor devuelto
List[Dict[str, Any]]

... Módulo de procesamiento de páginas para extracción de datos.

Este módulo contiene la lógica central para analizar imágenes de páginas de PDF. Su función principal, *extract_page_data_in_batch*, está optimizada para procesar múltiples páginas de manera eficiente, realizando la detección de layout en lote y luego el OCR para cada región de texto identificada.

worker.src.domain.translator.processor.**extract_page_data_in_batch**(*page_images: List[PIL.Image.Image]*, *confidence: float*) → List[Dict[str, Any]]

Procesa un lote de imágenes de página para extraer texto y layout.

Esta es una función clave para el rendimiento del worker. Realiza la detección de layout para todas las imágenes en una sola pasada y luego itera sobre cada página para realizar el OCR en las regiones de texto detectadas.

Parámetros

- **page_images** (List[Image.Image]) – Una lista de objetos *Image* de PIL, cada una representando una página.
- **confidence** (float) – El umbral de confianza para el modelo de detección de layout.

Devuelve
Una lista de diccionarios. Cada diccionario contiene los datos extraídos de una página, incluyendo regiones de texto, regiones de imagen y dimensiones de la página.

Tipo del valor devuelto
List[Dict[str, Any]]

worker.src.domain.translator.processor.**regenerate_pdf**(*output_pdf_path: str*, *translation_data: dict*, *position_data: dict*, *target_language: str*) → dict

Regenera un PDF usando datos de traducción y posición previamente guardados.

Esta función es una utilidad que permite reconstruir un documento PDF rápidamente después de que las traducciones hayan sido editadas manualmente, sin necesidad de volver a ejecutar el costoso proceso de OCR y layout.

Parámetros

- **output_pdf_path** (str) – Ruta del archivo donde se guardará el PDF regenerado.
- **translation_data** (dict) – Diccionario que contiene los textos traducidos por página y región.
- **position_data** (dict) – Diccionario que contiene las dimensiones de página y las posiciones de cada región de texto e imagen.
- **target_language** (str) – Código del idioma de destino para la selección de fuentes.

Devuelve
Un diccionario indicando el resultado de la operación.

Tipo del valor devuelto
dict

... Módulo de análisis de layout de documentos.

Este archivo contiene la lógica para detectar la estructura de una página (párrafos, títulos, imágenes, tablas, etc.) utilizando un modelo YOLOv10. Proporciona una clase singleton para cargar el modelo eficientemente y una función optimizada para procesar imágenes en lote.

class worker.src.domain.translator.layout.**LayoutElement**(*box:* Rectangle, *type: str*, *score: float*)

Bases: NamedTuple

Representa un elemento detectado en el layout del documento.

Parámetros

- **box** (Rectangle) – El cuadro delimitador del elemento.
- **type** (str) – El tipo de elemento (ej. "TextRegion", "ImageRegion").
- **score** (float) – La puntuación de confianza de la detección.

box: Rectangle

Alias for field number 0

score: float

Alias for field number 2

type: str

Alias for field number 1

class worker.src.domain.translator.layout.**LayoutModel**(*model_type='yolov10_doc'*)

Bases: object

Clase singleton para gestionar la carga y acceso al modelo YOLOv10.

Asegura que el modelo se cargue una sola vez en memoria, optimizando el uso de recursos.

get_model()

Devuelve la instancia del modelo YOLOv10 cargado.

class worker.src.domain.translator.layout.**Rectangle**(*x_1: float*, *y_1: float*, *x_2: float*, *y_2: float*)

Bases: NamedTuple

Representa un cuadro delimitador con coordenadas (x1, y1, x2, y2).

x_1: float

Alias for field number 0

x_2: float

Alias for field number 2

y_1: float

Alias for field number 1

y_2: float

Alias for field number 3

worker.src.domain.translator.layout.**get_layouts_in_batch**(*images: List[PIL.Image.Image]*, *model_type='yolov10_doc'*, *confidence: float = 0.45*, *batch_size: int = 16*) → List[List[*LayoutElement*]]

Obtiene el layout para una lista de imágenes usando inferencia por lotes.

Esta función es una optimización clave, ya que procesa múltiples imágenes en una sola llamada al modelo, reduciendo la sobrecarga y acelerando el proceso de detección.

Parámetros

- **images** (List[Image.Image]) – Lista de objetos *PIL.Image* a procesar.
- **model_type** (str) – El tipo de modelo a cargar (definido en *YOLO_MODEL_CONFIG*).
- **confidence** (float) – Umbral de confianza para filtrar las detecciones.
- **batch_size** (int) – Número de imágenes a procesar en cada lote de inferencia.

Devuelve
Una lista de layouts. Cada layout es, a su vez, una lista de *LayoutElement*.

Tipo del valor devuelto
List[List[*LayoutElement*]]

worker.src.domain.translator.layout.**merge_overlapping_text_regions**(*layout: List[*LayoutElement*]*) → Tuple[List[Tuple[*Rectangle*, str]], List[Tuple[*LayoutElement*, str]]]

Separa los elementos de un layout en regiones de texto y de imagen.

Actualmente, no realiza una fusión de regiones, simplemente clasifica los elementos detectados en dos categorías para su procesamiento posterior.

Parámetros
layout (List[LayoutElement]) – Una lista de *LayoutElement* detectados en una página.

Devuelve
Una tupla con dos listas: una para regiones de texto y otra para regiones de imagen.

Tipo del valor devuelto
Tuple[List[Tuple[*Rectangle*, str]], List[Tuple[*LayoutElement*, str]]]

... Módulo de Reconocimiento Óptico de Caracteres (OCR).

Proporciona una función simple para extraer texto de una imagen utilizando la biblioteca Tesseract a través de su wrapper *pytesseract*.

worker.src.domain.translator.ocr.**extract_text_from_image**(*image: PIL.Image.Image*) → str

Extrae texto de un objeto de imagen utilizando Tesseract OCR.

Parámetros
image (Image.Image) – El objeto *PIL.Image* del cual se extraerá el texto.

Devuelve
El texto extraído como una cadena. Devuelve una cadena vacía si ocurre un error.

Tipo del valor devuelto
str

... Módulo de Traducción de Texto.

Este archivo se encarga de la comunicación con la API de traducción (OpenAI a través de OpenRouter). Proporciona una función asíncrona para traducir lotes de texto, utilizando el formato de respuesta estructurada para garantizar la fiabilidad de la salida.

class worker.src.domain.translator.translator.**TranslationResponse**(**args: Any*, ***kwargs: Any*)

Bases: BaseModel

Define la estructura de respuesta esperada de la API de traducción.

translations: List[str]

async worker.src.domain.translator.translator.**translate_text_async**(*texts: List[str]*, *target_language: str*, *language_model: str = 'openai/gpt-4o-mini'*) → List[str]

Traduce una lista de textos de forma asíncrona al idioma especificado.

Utiliza la funcionalidad de *parse* (Structured Outputs) del cliente de OpenAI para forzar al modelo a devolver un JSON que se ajuste al esquema *TranslationResponse*. Esto aumenta la robustez y evita errores de formato en la respuesta.

Parámetros

- **texts** (List[str]) – Una lista de cadenas de texto para traducir.
- **target_language** (str) – El código ISO del idioma de destino (ej. "es", "fr").
- **language_model** (str) – El identificador del modelo a utilizar (ej. "openai/gpt-4o-mini").

Devuelve
Una lista de los textos traducidos. En caso de error, devuelve la lista de textos originales como fallback.

Tipo del valor devuelto
List[str]

Muestra
APIConnectionError, RateLimitError, APIStatusError – Si hay problemas de conexión con la API.

. . . Utilidades para el manejo de archivos PDF y sus representaciones como imágenes.

Este módulo contiene funciones auxiliares para tareas relacionadas con PDFs, como calcular las dimensiones de una página en puntos a partir de su imagen.

worker.src.domain.translator.pdf_utils.**get_page_dimensions_from_image**(*image_path: str*, *dpi: int = 300*) → Tuple[float, float]

Calcula las dimensiones de una página en puntos (points) a partir de una imagen.

Las dimensiones en PDF se miden en puntos (1 pulgada = 72 puntos). Esta función convierte las dimensiones en píxeles de una imagen a puntos, basándose en los DPI (puntos por pulgada) con los que se renderizó la imagen.

Parámetros

- **image_path** (str) – La ruta al archivo de imagen de la página.
- **dpi** (int) – Los DPI utilizados para crear la imagen desde el PDF.

Devuelve
Una tupla *(ancho, alto)* en puntos.

Tipo del valor devuelto
Tuple[float, float]

Muestra
Exception – Si no se puede abrir o procesar la imagen.

... Módulo de utilidades generales para el procesamiento de documentos.

Contiene funciones auxiliares para tareas comunes como la limpieza de texto, la gestión de fuentes para la generación de PDFs y el ajuste dinámico del tamaño de fuente para que el texto encaje en un cuadro delimitador.

worker.src.domain.translator.utils.**adjust_paragraph_font_size**(*paragraph: reportlab.platypus.Paragraph*, *available_width: float*, *available_height: float*, *style: reportlab.lib.styles.ParagraphStyle*, *min_font_size: int = 6*, *max_font_size: int = 72*) → reportlab.platypus.Paragraph

Ajusta dinámicamente el tamaño de fuente de un párrafo para que quepa en un área.

Intenta encontrar el mayor tamaño de fuente posible sin que el párrafo exceda el alto disponible. Primero reduce la fuente si es necesario y luego la aumenta si hay espacio sobrante, dentro de los límites definidos.

Parámetros

- **paragraph** (Paragraph) – El objeto *Paragraph* de ReportLab a ajustar.
- **available_width** (float) – El ancho máximo disponible para el párrafo.
- **available_height** (float) – El alto máximo disponible para el párrafo.
- **style** (ParagraphStyle) – El estilo del párrafo que se modificará.
- **min_font_size** (int) – El tamaño de fuente mínimo permitido.
- **max_font_size** (int) – El tamaño de fuente máximo permitido.

Devuelve
El objeto *Paragraph* con el tamaño de fuente ajustado.

Tipo del valor devuelto
Paragraph

worker.src.domain.translator.utils.**clean_text**(*text: str*) → str

Limpia una cadena de texto eliminando saltos de página y de línea.

Parámetros
text (str) – El texto a limpiar.

Devuelve
El texto limpio y sin espacios extra al principio o al final.

Tipo del valor devuelto
str

worker.src.domain.translator.utils.**get_font_for_language**(*target_language: str*) → str

Selecciona y devuelve el nombre de la fuente apropiada para un idioma.

Utiliza fuentes CID especiales para idiomas CJK (Chino, Japonés, Coreano) para garantizar la correcta renderización de los caracteres. Para el resto de idiomas, utiliza "OpenSans".

Parámetros
target_language (str) – El código ISO del idioma de destino.

Devuelve
El nombre de la fuente registrada en ReportLab.

Tipo del valor devuelto
str

CAPÍTULO 3

Indices and tables

- genindex
- modindex
- search

Índice de Módulos Python

b

w

Índice

L

M

O

P

Q

R

S

T

U

W

X

Y