

# Mini-pipeline de libros

Proyecto\_UT1\_RA1\_SBA

Rafael García López

Noviembre 2025

## Índice

<b>Books Pipeline (Goodreads → Google Books → Parquet)</b>	<b>2</b>
<b>Ejecución del pipeline completo</b>	<b>2</b>
BLOQUE 1 - Scraping (Goodreads → JSON)	2
1.1 URL utilizada	2
1.2 Selectores CSS (Cascading Style Sheets) empleados	2
1.3 Campos extraídos	2
1.4 Salida generada	2
1.5 Descarga de portadas	3
1.6 Pausas y buenas prácticas de scraping	3
1.7 Configuración y backend	3
BLOQUE 2 — Enriquecimiento (Google Books → CSV)	3
2.1 Endpoint utilizado	3
2.2 Prioridad de búsqueda	4
2.3 Campos extraídos de Google Books	4
2.4 Formato para campos multi-valor	4
2.5 Hipótesis de mapeo	5
2.6 Manejo de errores	5
2.7 Configuración	5
2.8 Salida	5
BLOQUE 3 - Integración y normalización (JSON + CSV → Parquet)	6
3.1 Integración de fuentes y construcción de la tabla staging	6
3.2 Normalización semántica	6
3.3 Identificador canónico ( <code>book_id</code> )	7
3.4 Deduplicación	7
3.5 Salidas	8
Conclusión	10
<b>Resumen Books Pipeline (Goodreads → Google Books → Parquet)</b>	<b>11</b>
1. Objetivo	11
2. Tecnologías utilizadas	11
3. Estructura del repositorio	11
4. Flujo del pipeline	12
5. Instalación y configuración	12
5.1. Instalar dependencias	12
5.2. Crear archivo <code>.env</code>	12
5.3. Ejecución	13
6. Publicación web (GitHub)	13

# Books Pipeline (Goodreads → Google Books → Parquet)

(README.md)

## Ejecución del pipeline completo

---

### BLOQUE 1 - Scraping (Goodreads → JSON)

Archivo: `src/scrape_goodreads.py`

Este script extrae libros desde la búsqueda pública de Goodreads sin autenticación y genera un fichero JSON con los resultados.

Opcionalmente, descarga portadas locales y extrae ISBN/ASIN accediendo a la ficha de cada libro.

#### 1.1 URL utilizada

`https://www.goodreads.com/search?q={query}&page={page}`

donde `{query}` es la cadena de búsqueda con espacios reemplazados por `+`.

#### 1.2 Selectores CSS (Cascading Style Sheets) empleados

Campo	Selector CSS
Filas	<code>table.tableList tr</code>
Título	<code>a.bookTitle</code>
Autor	<code>a.authorName</code>
Rating / votos	<code>span.minirating</code>
Portada (URL img)	<code>img.bookCover</code>

#### 1.3 Campos extraídos

Cada libro incluye:

- `title` — Título.
- `author` — Autor/a.
- `rating` — Valoración media (`float`).
- `ratings_count` — Número de valoraciones (`int`).
- `book_url` — URL absoluta a la ficha del libro.
- `cover_url` — URL de la imagen de portada (si existe).
- `cover_local_path` — Ruta relativa al fichero de portada descargado (ej. `covers/<titulo>.jpg`).
- `isbn10`, `isbn13` y `asin` — Extraídos opcionalmente desde la ficha del libro.

Por defecto, `isbn10`, `isbn13` y `asin` se dejan en `null`.

La extracción de ISBN/ASIN se activa con el parámetro de entorno `GOODREADS_FETCH_ISBN = true`

#### 1.4 Salida generada

- Archivo: `landing/goodreads_books.json`
- Codificación: UTF-8
- Contenido: Lista JSON con hasta el número solicitado de libros (`GOODREADS_MAX_BOOKS`, por defecto 15).

La ruta base del proyecto se calcula a partir de la ubicación del script y se asegura la existencia del directorio `landing/` antes de escribir el fichero.

## 1.5 Descarga de portadas

Cuando el libro tiene portada (`img.bookCover`):

1. Se obtiene la URL (`cover_url`).
2. Se genera un nombre de fichero a partir del título (`<titulo>.jpg`), saneando caracteres no válidos.
3. Se descarga la imagen y se guarda en el directorio `covers/<nombre_saneado>.jpg`
4. La ruta relativa se guarda en el campo `cover_local_path` del JSON.
5. Si la descarga falla, se registra y se muestra un mensaje de error y `cover_local_path` queda en `null`.

## 1.6 Pausas y buenas prácticas de scraping

Para evitar cargas excesivas sobre Goodreads:

- Se incluye una pausa corta (0.5s) entre peticiones a páginas y fichas.
- Se utiliza un User-Agent identificable y configurable por `.env`.

## 1.7 Configuración y backend

Los parámetros del scraping se leen de variables de entorno (con valores por defecto si no existen):

Variable	Descripción	Valor por defecto
<code>GOODREADS_SEARCH_QUERY</code>	Término de búsqueda	“Big Data”
<code>GOODREADS_MAX_BOOKS</code>	Nº máximo de libros a extraer	15
<code>GOODREADS_USER_AGENT</code>	User-Agent HTTP	“Mozilla/5.0 (Windows NT 10.0; Win64; x64)”
<code>GOODREADS_BACKEND</code>	Backend de scraping <code>requests</code> o <code>playwright</code>	“requests”
<code>GOODREADS_FETCH_ISBN</code>	Activar extracción de ISBN/ASIN desde ficha	“true”

Ejemplo de flujo de ejecución (`main()`):

1. Lee configuración desde `.env`
2. Determina el backend (`requests` o `playwright`)
3. Ejecuta `scrape_goodreads_search()`
4. Recorre páginas hasta obtener el límite solicitado
5. Entra en cada ficha y extrae ISBN/ASIN (Opcional)
6. Descarga portadas
7. Escribe el JSON resultante en `landing/goodreads_books.json`.

---

## BLOQUE 2 — Enriquecimiento (Google Books → CSV)

Archivo: `src/enrich_googlebooks.py`

Este script consulta la API pública de Google Books para enriquecer la información procedente de Goodreads (`landing/goodreads_books.json`).

Genera un archivo CSV con metadatos adicionales como autores, editorial, categorías, idiomas, identificadores y precios.

### 2.1 Endpoint utilizado

`https://www.googleapis.com/books/v1/volumes`

Parámetros enviados:

Parámetro	Descripción
<code>q</code>	Query de búsqueda (ISBN o título+autor).
<code>maxResults</code>	Siempre 1 → se toma el primer resultado devuelto.
<code>key</code>	GOOGLE_BOOKS_API_KEY (opcional).

## 2.2 Prioridad de búsqueda

Para cada libro del JSON de Goodreads se construye una query aplicando la siguiente prioridad:

1. `isbn13`
2. `isbn10`
3. `asin` (se usa también como `isbn:<asin>`)
4. Título + autor (usando campos entrecomillados):  
`intitle:"<título>" inauthor:"<autor>"`

Esto garantiza que se priorice la búsqueda por identificadores únicos y solo se use `title+author` como fallback

## 2.3 Campos extraídos de Google Books

De la respuesta de Google Books se extrae:

- Identificación y trazabilidad
  - `gb_id` — ID interno de Google Books.
  - `original_title` — Título original del JSON de Goodreads.
  - `original_author` — Autor original del JSON de Goodreads.
- Campos de Volumen:
  - `title` — Título normalizado por Google Books.
  - `subtitle` — Subtítulo (si existe).
  - `authors` — Lista de autores serializada con `|`
  - `publisher` — Editorial.
  - `pub_date` — Fecha de publicación (`publishedDate`).
  - `language` — Idioma (código ISO-639).
  - `categories` — Categorías serializadas con `|`
- Identificadores:
  - `isbn10` / `isbn13` — Extraídos de `industryIdentifiers`.
  - `asin` — Si aparece como identificador con tipo `ASIN`.
- Precio (`saleInfo`)
  - `price_amount` — Importe del precio (si existe `listPrice` o `retailPrice`).
  - `price_currency` — Moneda del precio (código ISO-4217).

Nota: Los campos ISBN/ASIN pueden sobrescribir o completar la información original de Goodreads, ya que se toman de la respuesta de Google Books.

## 2.4 Formato para campos multi-valor

Los campos con múltiples valores (varios autores o categorías) se serializan como cadenas separadas por `|`:

```
authors = "Autor1|Autor2"
categories = "Cat1|Cat2"
```

Si el campo no contiene valores, se deja vacío `""` en el CSV, lo que se interpreta como `null` al leer desde pandas.

## 2.5 Hipótesis de mapeo

- Se selecciona únicamente el primer elemento de items devuelto por Google Books.
- El primer ISBN disponible en `industryIdentifiers` con tipo `ISBN_13` o `ISBN_10` se usa como valor final.
- `ASIN` se usa solo si aparece explícitamente en la respuesta.
- `listPrice` se considera la fuente de precio preferente; `retailPrice` es fallback.
- Autores y categorías se serializan en formato de lista separada por `|`.
- Si Google Books no devuelve resultados para un libro, no se genera fila para él en el CSV.

## 2.6 Manejo de errores

El acceso a Google Books se realiza mediante la función `call_google_books_api`, que aplica:

- Reintentos automáticos cuando Google Books devuelve errores temporales:
  - Códigos 503 (Service Unavailable) o 429 (Too Many Requests).
  - Se realiza hasta `max_retries` intentos (por defecto 3), con backoff incremental (espera que aumenta en cada intento).
- Manejo explícito de fallos de red:
  - Cualquier `RequestException` se captura y se informa con un mensaje `[ERROR RED]`.
- Otros errores HTTP:
  - Para códigos distintos de 200/503/429, se muestra `[ERROR HTTP]` con el código y un fragmento de la respuesta.
- Control de excepciones generales por libro:
  - En el bucle principal se envuelve la llamada en un `try/except` de tipo “catch-all” para que, si ocurre algo inesperado con un libro concreto, el script continúe con el resto.
- Logging por libro. Para cada libro se imprime:
  - El título y autor originales.
  - La query enviada a Google Books.
  - Mensajes de enriquecido correcto, sin resultados o errores.

Se introduce una pequeña pausa entre llamadas (`time.sleep(0.3)`) para no saturar la API de Google Books.

Si `call_google_books_api` no devuelve ningún resultado (`item is None`), no se genera fila en el CSV de salida.

Si no se encuentra la variable de entorno `GOOGLE_BOOKS_API_KEY`, se muestra un aviso, pero el script intenta llamar igualmente a la API sin clave.

Si no existe el fichero de entrada `landing/goodreads_books.json`, se lanza un `FileNotFoundError` y el script termina.

## 2.7 Configuración

Variable de entorno:

`GOOGLE_BOOKS_API_KEY` — Clave de API para Google Books (opcional pero recomendada).

Ruta de entrada:

`landing/goodreads_books.json` — JSON con los libros procedentes de Goodreads.

Ruta de salida:

`landing/googlebooks_books.csv` — CSV enriquecido con datos de Google Books.

Las rutas se calculan a partir de la raíz del proyecto, utilizando `BASE_DIR` y `LANDING_DIR`.

## 2.8 Salida

Archivo generado: `landing/googlebooks_books.csv`

- Separador: `;`

- Codificación: `utf-8`
  - Cabecera incluida
  - Columnas (en este orden): `gb_id, original_title, original_author, title, subtitle, authors, publisher, pub_date, language, categories, isbn13, isbn10, asin, price_amount, price_currency`
- 

## BLOQUE 3 - Integración y normalización (JSON + CSV → Parquet)

Archivo: `src/integrate_pipeline.py`

Este módulo integra los datos procedentes de Goodreads (`landing/goodreads_books.json`) y Google Books (`landing/googlebooks_books.csv`) sin modificar los ficheros originales de la carpeta `landing/`.

La integración se basa en una tabla de `staging/` (intermedia), donde se normalizan campos clave, se aplican controles de calidad, se generan identificadores canónicos y se deduplican los registros para producir las tablas finales en `standard/`, junto con métricas y documentación.

### 3.1 Integración de fuentes y construcción de la tabla staging

La función `load_sources()` carga ambas fuentes en memoria en formato pandas.

Después, `build_staging()` realiza:

- Alineación de columnas entre Goodreads y Google Books:
  - Goodreads aporta: `title, author, rating, ratings_count, book_url, isbn10, isbn13, asin`.
  - Google Books aporta: `title, authors, publisher, pub_date, language, categories, isbn10, isbn13, asin, price_amount, price_currency`.
- Se renombran columnas para unificarlas (`titulo, autor_principal, editorial, precio, moneda`, etc.)
- Se añaden metadatos de trazabilidad:
  - `source_name` (`goodreads / googlebooks`)
  - `source_file` (`goodreads_books.json / googlebooks_books.csv`)
  - `row_number` (fila original dentro de cada fuente)
- Se escriben los registros combinados en la tabla `staging/books_staging.parquet` (fuera de `landing/`, cumpliendo el requisito de no modificar los ficheros originales).

Esta tabla staging es la base para normalización, calidad y deduplicación.

### 3.2 Normalización semántica

A partir de los datos combinados en `staging` se aplican las siguientes normalizaciones y derivaciones:

#### Fechas (ISO-8601)

Función: `normalize_date`:

- Soporta formatos típicos `YYYY`, `YYYY-MM`, `YYYY-MM-DD`.
- Si no se puede parsear, la fecha normalizada queda en `null`.
- Se mantiene también el campo original como `fecha_publicacion_raw`
- La fecha normalizada se convierte a formato `YYYY-MM-DD`, rellenando componentes faltantes con `01`

```
2018      → 2018-01-01
2017-05   → 2017-05-01
```

#### Idioma (BCP-47 simplificado)

- Se transforma a minúsculas.
- Se eliminan espacios.
- Si viene vacío o `NaN`, devuelve `null`.
- Se valida posteriormente con una aproximación a BCP-47 con `idioma_valido`

```
"EN" → "en"
"pt-BR" → "pt-br"
```

### Moneda (ISO-4217)

Función `normalize_currency`:

- Convierte a mayúsculas.
- Elimina espacios.
- Si viene vacío o NaN, devuelve null.
- Se valida posteriormente con `moneda_valida` (3 letras mayúsculas).

```
"usd" → "USD"
" eur " → "EUR"
```

### Autores / Categorías

- Campo original en texto: `autores`, `categorias`.
- Cadenas tipo "Autor1|Autor2" se convierten en listas ["Autor1", "Autor2"].
- Se eliminan vacíos y espacios.
- Se guardan como listas `autores_list` y `categorias_list`.

### Otros derivados

Además se generan:

- `titulo_normalizado` — título en minúsculas.
- `autor_principal` — si falta, se toma el primer autor de `autores_list`.
- `anio_publicacion` — año extraído de `fecha_publicacion`.
- `longitud_titulo` — longitud del título (usada como criterio de desempate).

### 3.3 Identificador canónico (`book_id`)

Función: `generate_book_id_from_row`

1. Si existe `isbn13`, se utiliza directamente como `book_id`.
2. Si no, se genera un hash (SHA-1) estable a partir de:
  - `titulo_normalizado | autor_normalizado | editorial_normalizada | anio_publicacion`

Los campos normalizados `autor_normalizado` y `editorial_normalizada` se generan en `deduplicate()` a partir de:

- `autor_principal / autor / author`
- `editorial / publisher`

### 3.4 Deduplicación

La deduplicación se realiza en dos fases:

1. Anotación de calidad
2. Elección de fila ganadora.

La función `deduplicate(staging)` aplica las reglas sobre la tabla de staging:

1. Flags y prioridad:
  - `has_isbn13` — True si `isbn13` no es nulo.
  - `has_precio` — True si `precio` no es nulo.
  - `prioridad_fuente` — `googlebooks` = 3, `goodreads` = 2, otras = 1.
2. Validación y errores (soft fail)
  - Se anotan errores con `annotate_errors`
  - Se añaden `error_codes` (lista) y `has_error` (booleano) según reglas:
    - `R1_MISSING_KEY_TITULO_AUTOR`

- R2\_INVALID\_DATE
  - R3\_INVALID\_LANGUAGE
  - R4\_INVALID\_CURRENCY
  - R5\_INVALID\_RATING
  - Los registros con errores (`has_error == True`) no participan en la deduplicación final de `dim_book` (pero sí aparecen en `book_source_detail`).
3. Selección del ganador por `book_id`
- Para cada conjunto con el mismo `book_id`, se elige la primera fila ganadora ordenando por:
    - `has_isbn13` (primero los que tienen `isbn13`)
    - `has_precio` (después los que tienen precio)
    - `prioridad_fuente` (`googlebooks > goodreads > otras`)
    - `longitud_titulo` (título más completo)
4. Unión de listas
- Autores y categorías se unen sin duplicados sobre las filas válidas del mismo `book_id`:
    - `autores_unificados` (unicón sin duplicados)
    - `categorias_unificadas` (unicón sin duplicados)
5. Construcción de `dim_book`.
- La tabla resultante contiene **una fila única por libro**, con:
    - información ganadora,
    - autores y categorías unificados,
    - normalizaciones semánticas,
    - trazabilidad (`fuelle_ganadora`, `ts_ultima_act`).

### 3.5 Salidas

Además de un Parquet de staging (`staging/books_staging.parquet`), el módulo genera las siguientes salidas:

#### `standard/dim_book.parquet`

Tabla canónica con una fila por `book_id` (solo registros válidos). Incluye:

Campo	Tipo	Nulo	Descripción
<code>book_id</code>	string	no	Identificador canónico del libro ( <code>isbn13</code> o hash estable).
<code>titulo</code>	string	sí	Título del libro tal y como aparece en la fuente ganadora.
<code>titulo_normalizado</code>	string	sí	Título en minúsculas, sin espacios extra (para comparación/deduplicación).
<code>autor_principal</code>	string	sí	Autor principal del libro.
<code>autores</code>	array	sí	Lista de autores (fusionados de todas las fuentes).
<code>editorial</code>	string	sí	Editorial del libro.
<code>anio_publicacion</code>	string	sí	Año de publicación (YYYY) derivado de la fecha de publicación.
<code>fecha_publicacion</code>	date	sí	Fecha de publicación en formato ISO-8601 (YYYY-MM-DD).
<code>idioma</code>	string	sí	Idioma en BCP-47 simplificado (ej. <code>en</code> , <code>es</code> , <code>pt-br</code> ).
<code>isbn10</code>	string	sí	Identificador ISBN-10 (si existe).
<code>isbn13</code>	string	sí	Identificador ISBN-13 (si existe).
<code>asin</code>	string	sí	Identificador ASIN (si existe).
<code>paginas</code>	int	sí	Número de páginas (no disponible en este pipeline).
<code>formato</code>	string	sí	Formato físico/digital (no disponible en este pipeline).
<code>categoria</code>	array	sí	Lista de categorías temáticas.
<code>precio</code>	float	sí	Precio del libro (si está disponible en Google Books).
<code>moneda</code>	string	sí	Moneda en ISO-4217 (ej. <code>EUR</code> , <code>USD</code> ).
<code>fuelle_ganadora</code>	string	no	Fuente del registro ganador ( <code>googlebooks</code> o <code>goodreads</code> ).
<code>ts_ultima_act</code>	timestamp	no	Marca temporal de generación del registro canónico (ISO-8601).

#### `standard/book_source_detail.parquet`



Detalle completo de todas las filas de **staging** (válidas y con error), por fuente y fila original, ya ordenadas por prioridad, incluyendo campos normalizados, flags de calidad y trazabilidad. Incluye:

- Campos originales unificados de ambas fuentes (título, autor, rating, etc.).
- Campos normalizados (titulo\_normalizado, idioma, fecha\_publicacion, etc.).
- Listas derivadas:
  - **autores\_list** — Lista de autores ya tokenizados.
  - **categorias\_list** — Lista de categorías normalizadas.
- Flags y prioridad:
  - **has\_isbn13** — Flag: indica si la fila tiene **isbn13** no nulo.
  - **has\_precio** — Flag: indica si la fila tiene **precio** no nulo.
  - **prioridad\_fuente** — Prioridad de la fuente en la regla de supervivencia (**googlebooks** > **goodreads**).
  - **has\_error** — Indica si esa fila tuvo algún problema durante el procesamiento (**true/false**)
  - **error\_codes** — Lista o cadena con los códigos de error que afectaron a la fila.
- Trazabilidad y claves:
  - **source\_name** — Nombre de la fuente (**goodreads**, **googlebooks**).
  - **source\_file** — Nombre del fichero de origen en **landing/** (**goodreads\_books.json** / **googlebooks\_books.csv**)
  - **source\_id** — ID secuencial en **book\_source\_detail**
  - **row\_number** — Número de fila original dentro de cada fuente.
  - **book\_id** — Identificador canónico asignado a la fila.
  - **book\_id\_candidato** — Usada para trazabilidad; coincide con **book\_id**
  - **ts\_ingesta** — Timestamp de carga en **staging**

La trazabilidad de procedencia se mantiene mediante **source\_name**, **source\_file** y **book\_id\_candidato**, que permiten identificar el origen de cada registro.

Además, en **dim\_book** el campo **fuentes\_ganadora** indica de qué origen proviene el valor final seleccionado tras la deduplicación.

#### Comparativa: **dim\_book** vs **book\_source\_detail**

Aspecto	<b>dim_book</b>	<b>book_source_detail</b>
Propósito	Modelo canónico final, listo para consumo	Detalle por fuente y registro original
Nivel	Capa estándar / Gold	Capa de detalle / Silver-raw
Cardinalidad	1 fila por <b>book_id</b>	Varias filas por <b>book_id</b> (una por fuente y fila original)
Estado del dato	Normalizado, deduplicado y sin errores ( <b>has_error</b> = 0)	Incluye filas válidas y con error
Origen de los valores	Fila ganadora según reglas de deduplicación	Datos tal y como llegan de cada fuente + normalizaciones básicas
Uso principal	Analítica, reporting, consumo downstream	Auditoría, trazabilidad, debugging del pipeline
Trazabilidad	<b>book_id</b> , <b>fuentes_ganadora</b> , <b>timestamps</b>	<b>source_name</b> , <b>source_file</b> , <b>row_number</b> , <b>book_id_candidato</b>

**docs/quality\_metrics.json** Fichero JSON con métricas de calidad y trazabilidad. Incluye:

- Sobre **dim\_book**:
  - Número de filas y columnas.
  - % de nulos por campo.
- Sobre **book\_source\_detail**:
  - Número de filas y columnas.
  - % de nulos por campo.

- Conteo de duplicados por isbn13, (titulo\_normalizado, autor\_principal, editorial)
- Filas por fuente (goodreads, googlebooks).
- Validaciones globales:
  - porcentaje\_idiomas\_validos
  - porcentaje\_monedas\_validas
  - porcentaje\_fechas\_validas
  - porcentaje\_clave\_titulo\_autor\_presente
  - porcentaje\_filas\_validas
  - porcentaje\_ratings\_validos (si existe rating)
  - % de nulos en campos clave: titulo, isbn13, precio
  - porcentaje\_registros\_invalidos (filas con has\_error = True)
- Logs de reglas de calidad:
  - logs.por\_archivo — número de errores por source\_file y código de regla.
  - logs.por\_regla — número total de errores por cada código de regla.
- Metadatos de entrada (metrics[“entradas”]):
  - Para goodreads y googlebooks: ruta, n\_filas, n\_columnas y tamano\_bytes

docs/schema.md

Se genera un fichero de documentación con el esquema de las tablas para dim\_book y book\_source\_detail:

- Nombre de columna
- Tipo
- Nullability (incluyendo % de nulos)
- Un valor de ejemplo

---

## Conclusión

El proyecto implementa un pipeline ETL completo:

- Extrae libros desde Goodreads.
- Los enriquece con Google Books.
- No modifica landing/ (solo lectura).
- Normaliza semánticamente fechas, idiomas y monedas.
- Integra ambas fuentes en una tabla de staging con normalización y reglas de calidad.
- Deduplica con un identificador canónico (book\_id) y criterios de supervivencia claros.
- Produce Parquets limpios (dim\_book, book\_source\_detail).
- Genera métricas de calidad (quality\_metrics.json)
- Documenta completamente el esquema (schema.md).

# Resumen Books Pipeline (Goodreads → Google Books → Parquet)

(README\_resumen.md)

---

## 1. Objetivo

Este proyecto implementa un pipeline completo de **Extracción** → **Enriquecimiento** → **Integración** para consolidar datos de libros a partir de:

- Scraping de Goodreads
- Google Books API
- Integración con normalización semántica y deduplicación

El resultado final produce:

- Un **modelo canónico** limpio en Parquet
  - Un archivo de **detalle por fuente**
  - Un conjunto de **métricas de calidad**.
- 

## 2. Tecnologías utilizadas

- Python
  - Requests
  - BeautifulSoup4
  - lxml
  - python-dotenv
  - Playwright (opcional)
  - pandas
- 

## 3. Estructura del repositorio

books-pipeline/	Proyecto completo del pipeline de libros
landing/	Datos "crudos" recién extraídos de las fuentes
goodreads_books.json	Datos sin procesar obtenidos de Goodreads
googlebooks_books.csv	Datos sin procesar obtenidos de Google Books
staging/	Datos transformados parcialmente (limpios pero no finales)
book_staging.parquet	Dataset unificado y normalizado previo al estándar
standard/	Datos finales listos para análisis
dim_book.parquet	Dimensión de libros estandarizada
book_source_detail.parquet	Detalles de origen por libro (trazabilidad)
covers/	Carpeta para almacenar portadas descargadas
docs/	Documentación del pipeline
schema.md	Descripción de los esquemas de datos
quality_metrics.json	Métricas de calidad del pipeline/dataset
src/	Código fuente del pipeline
scrape_goodreads.py	Script para extraer datos de Goodreads

<code>enrich_googlebooks.py</code>	Script para enriquecer datos usando Google Books
<code>integrate_pipeline.py</code>	Script principal que orquesta todo el pipeline
<code>README.md</code>	Descripción general del proyecto, cómo usarlo
<code>requirements.txt</code>	Lista de dependencias de Python necesarias
<code>.env.example</code>	Fichero de ejemplo de variables de entorno (API keys, ...)
<code>.env</code>	Variables de entorno (API keys, ...)

---

## 4. Flujo del pipeline

### 1. Scraping (Goodreads → JSON)

Script: `src/scrape_goodreads.py`

- Lanza una búsqueda en Goodreads (`GOODREADS_SEARCH_QUERY`).
- Uso de User-Agent personalizado y pausas opcionales para un scraping responsable.
- Extrae: título, autor, rating, número de valoraciones y URL del libro.
- Genera: `landing/goodreads_books.json`.

### 2. Enriquecimiento (Google Books → CSV)

Script: `src/enrich_googlebooks.py`

- Para cada libro de Goodreads, llama a Google Books API.
- Prioriza `isbn13 > isbn10 > combinación título + autor`.
- Extrae: título, autores, editorial, fecha de publicación, idioma, categorías, ISBNs, precio/moneda.
- Genera: `landing/googlebooks_books.csv` (; y UTF-8).
- Extrae título, autores, editorial, fecha, idioma, categorías, precios, ISBNs.

### 3. Integración y estandarización (JSON + CSV → Parquet)

Script: `src/integrate_pipeline.py`

- Lee los ficheros de `landing/`.
  - *Los ficheros de `landing/` no se modifican; `staging/` y `standard/` contienen las salidas del pipeline.*
  - Normaliza fechas (ISO-8601), idioma (BCP-47) y moneda (ISO-4217).
  - Genera un `book_id` canónico (ISBN13 o hash).
  - Deduplica registros aplicando reglas de supervivencia (`ISBN13 > precio > fuente > longitud`).
  - Salidas:
    - `standard/dim_book.parquet` → tabla canónica de libros.
    - `standard/book_source_detail.parquet` → detalle por fuente y registro.
    - `docs/quality_metrics.json` → métricas de calidad (nullos, duplicados, formatos, etc.).
    - `docs/schema.md` → descripción de los esquemas de datos
- 

## 5. Instalación y configuración

### 5.1. Instalar dependencias

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

### 5.2. Crear archivo `.env`

Puedes copiarlo desde `.env.ejemplo`. Ejemplo:

```
GOOGLE_BOOKS_API_KEY=TU_API_KEY_AQUI
GOODREADS_BACKEND=requests/playwright
GOODREADS_FETCH_ISBN=true/false
GOODREADS_SEARCH_QUERY=big data
GOODREADS_MAX_BOOKS=25
GOODREADS_USER_AGENT=Mozilla/5.0 (Windows NT 10.0; Win64; x64)
```

---

### 5.3. Ejecución

1. Scraping Goodreads → JSON

```
python src/scrape_goodreads.py
```

Salidas generadas:

- landing/goodreads\_books.json

2. Enriquecimiento Google Books API → CSV

```
python src/enrich_googlebooks.py
```

Salidas generadas:

- landing/googlebooks\_books.csv

3. Integración y estandarización → Parquet

```
python src/integrate_pipeline.py
```

Salidas generadas:

- standard/dim\_book.parquet
- standard/book\_source\_detail.parquet
- staging/books\_staging.parquet
- docs/quality\_metrics.json
- docs/schema.md

---

## 6. Publicación web (GitHub)

- <https://github.com/IA-CSG/books-pipeline>