VOD Scraper — Interview Task

Project Overview

Build a small data pipeline that crawls Video-on-Demand (VoD) platforms, stores normalized metadata in a persistent database, and exposes that data through a RESTful API.

> **Requirement (mandatory):** The interviewee **must crawl at least two distinct VoD platforms** and **map identical content across platforms to a single unified item** in the database. For example, the movie *Inception* should be stored as one item even if it appears on both Filimo and Namava; each platform's `source_id` should be stored for that item.

Requirements

1) Web Scraper

- Crawl **at least two** VoD platforms (examples: Filimo, Namava, and similar regional platforms).
- **Data collection (minimum):**
  - Title
  - Release year
  - Genres
  - Source identifier on the platform (e.g., `source_id`)
- **Metadata (bonus / highly encouraged):**
  - Producers
  - Directors
  - Actors/cast
  - Runtime, language, synopsis if available
- **Content mapping rule (mandatory):**
  - Implement logic to identify when two platform items refer to the same real-world content (movie or series) and map them to one unified database item.
  - For each unified item store a list of sources with `platform` and `source_id` pairs.
  - Implement a reasonable matching strategy (e.g., normalized title matching + release year + optional fuzzy matching on alternate titles or external identifiers such as IMDB ID if available).

2) RESTful API (Django + DRF recommended)

Implement endpoints that provide:

- `GET /series/` — list of series, **sorted by release year** (descending).
- `GET /movies/` — list of movies, **sorted by release year**.
- `GET /items/{id}/` — item details, including:
  - Canonical metadata (title, year, genres, synopsis, runtime, etc.)
  - A list of `sources` (each with `platform`, `source_id`, and `url`)
  - Any available credits (directors, producers, actors), if scraped

---

What to submit

1- Scraper code (Scrapy or other framework) with instructions to run.
2- Django project (or equivalent) with migrations and a minimal set of fixtures or instructions to seed the DB.

Technologies: Scrapy (optional), Django + Django REST Framework, PostgreSQL, Redis (optional for caching).

Good luck — this is a practical test of scraping, data modeling, and API design. Please include clear instructions so the reviewer can reproduce your results.