Automated News Scraping and Storage Pipeline

Sakthe Balan G K

February 2, 2025

Abstract

This report describes an automated pipeline for scraping, storing, and deduplicating news articles using Python, Playwright, PostgreSQL, and cronjob. The pipeline extracts real-time news, processes articles, stores them in a structured database, and removes duplicates based on similarity constraints.

1 Introduction

With the vast amount of information available online, automating news extraction is crucial for applications like trend analysis and real-time monitoring. This project implements a modular pipeline that automates news scraping, processing, and deduplication.

1.1 Objectives

- Dynamically scrape news articles using Playwright.
- Store extracted news articles and images in PostgreSQL.
- Remove duplicate articles based on text similarity.
- Automate execution using cronjob in linux.

2 Pipeline Overview

The pipeline is divided into six modules, each responsible for a distinct processing step:

Module	Name	Functionality
1	Preprocessing	Fetches the Google News homepage HTML.
2	Transformation	Extracts the "Top Stories" section dynamically.
3	Scraping	Extracts news headlines, URLs, images, and timestamps using Playwright.
4	Database Storage	Inserts extracted articles into PostgreSQL and stores images as binary data.
5	Deduplication	Removes duplicate articles using URL comparison and text similarity.
6	Orchestration	Automates execution of all modules in sequence.

2.1 Config.json: Central Configuration File

The pipeline uses a JSON configuration file, `config.json`, to manage key parameters.

```
{
    "news_homepage": "https://news.google.com/",
    "url": "https://news.google.com/topics/CAAqKggKIi...",
    "output_file": "top_stories_news.json",
    "scroll_timeout": 2000,
    "headless": true,
    "user_agent": "Mozilla/5.0␣(Windows␣NT␣10.0;␣Win64;␣x64)"
        ,
    "selectors": {
        "story_container": "article",
        "headline": "a.gPFEn:first-of-type",
        "thumbnail": "figure␣img"
    }
}
```

Listing 1: config.json Configuration

Key Parameters:

- `news_homepage` - Base URL for news scraping.
- `output_file` - JSON file where extracted articles are stored.
- `scroll_timeout` - Time delay for page scrolling.
- `selectors` - CSS selectors for extracting headlines, images, and URLs.

3 Implementation Details

3.1 Module 3: Scraping News Data

The scraper extracts news headlines, URLs, and timestamps using Playwright.

```
with sync_playwright() as p:
    browser = p.chromium.launch(headless=True)
    page = browser.new_page()
    page.goto(news_url, wait_until="networkidle")
    headlines = page.query_selector_all("article␣a.gPFEn")
```

Listing 2: Module 3: Scraping with Playwright

3.2 Module 4: Storing Data in PostgreSQL

News articles and images are stored in a relational database.

```
CREATE TABLE articles (
    id SERIAL PRIMARY KEY,
    headline TEXT NOT NULL,
    url TEXT NOT NULL,
    scrape_time TIMESTAMP NOT NULL,
    article_date DATE NOT NULL,
    image_id INTEGER REFERENCES images(id) ON DELETE CASCADE
);
CREATE TABLE images (
    id SERIAL PRIMARY KEY,
    image_data BYTEA NOT NULL
);
```

Listing 3: PostgreSQL Table Schema

4 Challenges and Solutions

Issue	Solution
URLs extracted were incorrect	Used proper URL resolution techniques with `urljoin()` in Module 3.
Duplicate articles were not removed correctly	Implemented a similarity threshold of 85% for headlines in Module 5.
Images were stored as file paths instead of binary data	Modified Module 4 to store images in PostgreSQL using `BYTEA`.

Task Scheduler was failing silently	Added error logging to `pipeline.log` in Module 6.

5 How to Run the Pipeline

To execute the pipeline, follow these steps:

5.1 1. Install Dependencies

Run the following command to install all required Python libraries:

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

5.2 2. Run the Pipeline Manually

Before running, make sure to install postgresql, and the tables are correctly created as shown above in section 3.2 (module 4).
To test the pipeline, run the module_6.py file. This will orchestrate the whole web scraping from module 1 to 5.

When the module_6.py file is run, We will see a pipeline.log file which is the log file. Check that for the proper running of the code and the outputs. I have included the pipeline file, news_stories json file (output file of module_3) for reference.

6 Conclusion

This project successfully automates real-time news scraping, processing, and storage using a structured MLOps pipeline.