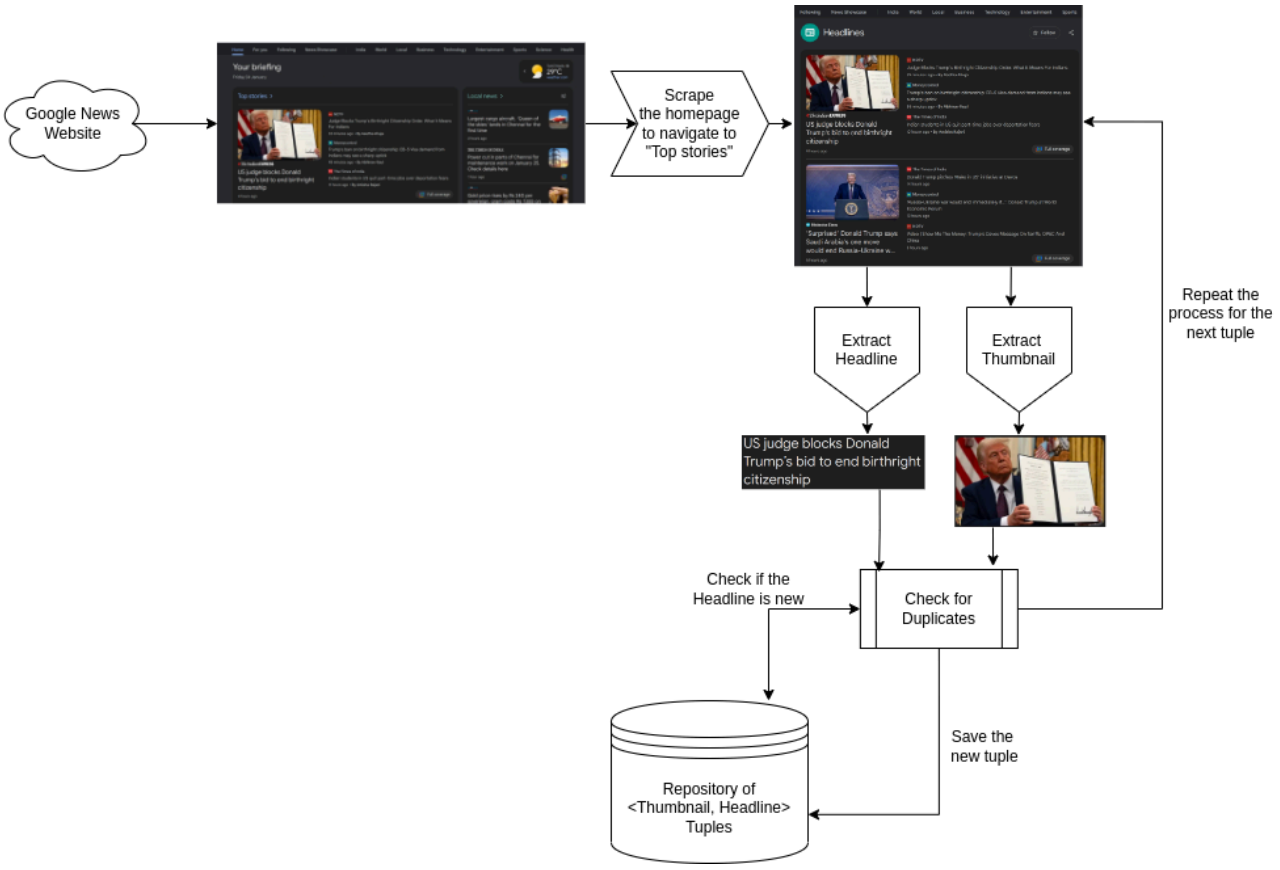
DA5402: Assignment #1

Flickr30k is a popular open-source image captioning dataset in use today, which you may download from HuggingFace and other archives.

Let's build our own data pipeline to create an "image captioning" dataset. To get there, we are going to use Google News as the Portkey. Your task is to setup the following pipeline using Python scripts and demonstrate that your pipeline, when set up for automated execution, shall produce a continuous feed of <image, caption> tuples.



Module 1 [5 points]

Create a Python script using any web scrapping libraries to scrape the home page of Google News. Don't hard code the URLs as the home page URLs may change over time. Ensure that such parameters are configurable via a command line or a configuration file.

Module 2 [5 points]

Create a Python script to scrape the "Top Stories" link from the home page. Don't hard code the "Top stories" string on the home page as the heading may change over time. Ensure that such parameters are configurable via a command line or a configuration file.

Module 3 [10 points]

Create a Python script to extract the thumbnail and the headline of every story from that page. Remember that the page is set up for lazy loading. Your script should do the needful to factor lazy loading. The layout of the “Top stories” page may change over time, so ensure that you create your module in a way for easy updates later.

Module 4 [10 points]

Create a Python script to store the extracted tuple in a database (pgsql/mariadb/mongodb) are the allowed choice. Your database table should have one table for storing the image data and the other table for storing the headlines and other meta information such as URLs, scrape timestamps, article date, etc.

Module 5 [10 points]

Create a Python script to check if a tuple is already present in the DB based on some de-duplication constraint. One naive constraint is to check the headline, but that’s not the best, obviously. Get creative here.

Module 6 [10 points]

Write a Python script to orchestrate all the above modules to execute in a cascaded style. The orchestration script should log the time stamps of invocation, error statuses, and other relevant details for debugging later. Moreover, the entire pipeline should be friendly for setting it up as a CronJob [Archwiki].