Technical Design Document

Overview

This document outlines the primary requirements for developing a news tracking solution, while also addressing key challenges and proposing strategies to overcome them.

The project offers a solution for UK students who are studying in a PR direction. The students need to track the most relevant news and make a presentation based on the most important news. To automate the process of “trend tracking” the university wants to implement a tool that collects the top daily news in the United Kingdom.

The students will monitor how the biggest UK television BBC is holding the TOP of searches in the web.

Background

BBC API

While BBC does not have a free public API available for developers, BBC does offer an RSS feed. **RSS (Really Simple Syndication)** is a web feed format used to publish frequently updated information such as news headlines, blog posts, or podcasts. The BBC provides RSS feeds for many of its news sections, allowing users, developers, or applications to access up to date news headlines and summaries directly. The RSS feeds are free to use and publicly accessible, though the copyright rules must be respected regardless of the RSS feeds being public, the content is still owned by BBC.

These are some of the feeds offered by BBC:

- Top Stories
- World
- UK
- Business
- Politics
- Health
- Education & Family
- Science & Environment
- Technology
- Entertainment & Arts

These RSS feeds simplify how the data is accessed and parsed by the backend.

According to the task's requirements, all BBC sections must be polled, which is possible using RSS feeds since BBC provides a feed for each of the sections featured in their web page.

What's featured in the RSS feeds?

Each of the feeds, when accessed via a web browser (alternatively can be accessed with a script using http methods) returns an xml file with the content corresponding to the news related to the feed's section.

The XML file includes:

- Title
- Description
- Link to news article
- Published date
- Thumbnail

While the RSS feeds include all the news featured in the corresponding section, some of the most recent news articles may not appear right away.

Alternatives

The BBC's website is publicly accessible and can be scraped with programming tools but it may not be as simple as using RSS feeds.

Google Trends API

Just like BBC, Google does not provide a public API for Google Trends. Google does provide a non-official alpha version of the API, but the user must apply for approvement to use the API. This version of the API is still in testing stage and provides a more extensive set of functionalities compared to the public Google Trends website.

Since access to the Google Trends API alpha may not be guaranteed, it is worth looking into other alternatives.

Pytrends

Pytrends is an unofficial API that provides a simple interface to automate the retrieval of reports from Google Trends. The library streamlines the process of programmatically accessing trends data such as search interest over time, related queries, and regional interest.

However, it is important to note that Pytrends is not an official Google product, and its last stable release was on April 13, 2023. Due to the unofficial nature of the API and the infrequent updates, there is a risk that changes on Google's end could affect its functionality. This potential instability should be carefully considered when integrating Pytrends into any production application.

SerpApi Google Trends API

The SerpApi API for Google Trends provides a reliable alternative to both Pytrends and the official Google Trends API. It enables users to scrape data directly from the Google Trends search page and offers endpoints to access current trending topics, autocomplete suggestions, and related news.

However, the free tier is limited to **100 requests per month**, which can be a notable limitation for applications that require a higher volume of queries. This constraint should be taken into account when evaluating SerpApi as a solution.

SearchApi

SearchApi offers an alternative to SerpApi, supporting similar query capabilities. A key distinction is that the **Trending Now** endpoint in SearchApi returns a ranking score for each trending topic. This feature simplifies implementation by offloading the ranking computation to the third-party API, thereby reducing the amount of processing required in the backend.

However, it is important to note that SearchApi operates behind a paywall, with pricing starting at $40 USD per month for a quota of 10,000 searches. This cost consideration should be weighed when evaluating SearchApi as a solution for trend data retrieval.

Response Payload

Each response returned by the API provides the same results as the Google Trends website would, it can provide several kinds of information regarding the search term provided, such as, interest over time, compared breakdown by region, interest by region, related topics and related queries.

Using the SerpApi Google Trends API as a reference, the following can be noted:

Interest Over Time

The **Interest Over Time** endpoint provides data reflecting the popularity of a given search term over a specified period. The response includes the date on which the interest was recorded, the query itself, and a numerical value representing the level of interest for that term at that time.

The API supports several optional parameters such as category, property (type of search), date range, and geographic region, among others. These options allow users to refine their queries and focus on more specific subsets of related topics. Each query has data of the interest for several dates.

In cases where Google Trends does not find any matches for the search term provided, the API will return an error code indicating that no relevant information was available for the query.

Related Topics

The related topics endpoint can provide similar queries to the search term provided. Among the information provided by this endpoint is the type of the topic which can be used to observe how the result relates to the query, the title and a value which indicates the popularity of the related topic.

Trending Now

The **Trending Now** endpoint returns a list of news topics experiencing a significant rise in popularity within a specified time period. The data provided by this endpoint includes the search query, search volume, percentage increase in interest, associated categories, and a detailed trend breakdown.

The **trend breakdown** offers additional context by presenting related topics that contribute to the overall search volume for the principal trend. This information is valuable for understanding the composition and drivers behind emerging trends in real time.

How does Google relate a keyword to a topic?

Queries

For Google to relate a keyword to a topic, first, the query needs to interpret the query. To interpret the query, Google's algorithm cleans the query and checks for **entities,** and removes unnecessary words. Also, the query is **expanded** to match similar ones. Once the query is interpreted by Google's system it is sent to the index to retrieve the results.

Ranking

After the search engine has processed the query, the results are ranked based on different parameters such as the following:

- Quality of the page

- Uniqueness of content
- Relative importance on the internet.
- Location

Related to Google Trends Api

The Google Trends API, as well as third-party services such as SerpApi, offer an autocompletion feature designed to assist users in discovering related topics and keywords. This endpoint accepts a user-provided keyword and queries Google's system to identify relevant matches. It returns a list of suggested keywords and topics that are semantically or contextually related to the original input.

These suggestions may include different types of keywords and thematic groupings, helping users refine their queries or uncover associated trends to broaden their research or analysis effectively.

Natural Language Processing

Natural Language Processing (NLP) is a specialized branch of computer science and artificial intelligence focused on enabling computers to comprehend, interpret, and generate human language through the application of machine learning techniques.

Within the scope of this document, we will concentrate specifically on **Named Entity Recognition (NER)**, a fundamental NLP task that facilitates alternative methods for processing queries and establishing relationships between topics.

NER involves identifying and classifying predefined categories of entities—such as names of people, organizations, locations, dates, and more—within unstructured text. Several methodologies exist for entity extraction, including:

- **Rule-based approaches**, which rely on handcrafted rules and patterns;

- **Machine learning approaches**, which utilize statistical models trained on annotated datasets;
- **Hybrid approaches**, combining both rule-based and machine learning techniques to improve accuracy and flexibility.

Employing these approaches enables the matching of news articles to trending topics through various mechanisms, such as:

- **Lexical Matching:** Direct comparison of textual content based on exact word matches.
- **Semantic Matching:** Evaluation of meaning-based similarities between entities or topics.
- **Categorical Matching:** Correlation based on news categories combined with identified entities.
- **Variation Matching:** Recognition and handling of different linguistic forms representing the same underlying concept.

Together, these techniques support more nuanced and effective integration of news data with trend analysis.

Requirements

Business Requirements

- Students must receive only verified information from BBC News and Google Trends.
- The solution must get all news from all BBC sections at least 3 times a day.
- Each new story's popularity must be estimated with Google Trends.
- Google Trends' current trends must be stored separately for each day.
- Headlines of BBC's main page must be stored separately as their "trend".
- Every day, the solution must send an email with 3 lists for the previous day (between 00:00 and 01:00 am): 1. Google trends; 2. BBC trends; 3. BBC all news ordered by trend score (max 50).
- The email should contain the percentage of matches between Google and BBC trends.
- A Web UI is required only for Administration purposes: changing the target email, number of news items, and the email sending period.

Non-Functional Requirements

- The technology stack must be kept consistent with the university's existing software products.
- Use **Selenium** for Web-pages parsing.
- Use **FastAPI** as the main framework.
- Integrate Google Trends via an existing open-source library or API.
- Use **AWS SES** (best option) or SMTP for email sending.
- UI is optional in this phase of the project.

Solution Proposal

The proposed system will aggregate news articles from the BBC and estimate their popularity using data from Google Trends.

Users will receive a consolidated list featuring:

- News headlines from the BBC main page,
- Current trending topics from Google Trends,
- The top 50 BBC news articles ranked according to their estimated popularity as determined by Google Trends data.

This solution aims to provide timely and relevant insights by combining authoritative news sources with dynamic trend analysis.

How will this be achieved?

Before determining the most suitable architecture for this project, it is essential to evaluate the key challenges that may arise during development.

A primary requirement is to retrieve all BBC news articles at least three times per day. While this is technically straightforward, it is crucial to rely on a dependable news source, preferably one provided officially by the BBC.

Another important consideration is minimizing the number of API calls made daily to both the BBC and Google Trends. Reducing these calls is vital to maintaining the overall performance and responsiveness of the application, as well as ensuring compliance with any rate limits imposed by the respective API providers.

Jobs

We believe that implementing jobs, is the most viable solution for this requirement, this allows us to schedule the scraping of both current BBC ´s news articles and Google Trends' current most popular trends.

To implement each of the processes, two alternatives arise, both provided by AWS, Lambda and EC2. EC2 is a service that allows using virtual machine instances to host solutions, EC2 is scalable and easy to maintain, EC2 falls under the category of infrastructure as a service.

Furthermore, lambda is a platform that allows to execute a piece of code written in one of the compatible languages (Python in this case) when a trigger is fired. Lambda does not need to be managed, since the control of the instance is offloaded to AWS.

The core details about AWS functionality won't be discussed further for the scope of this document, please refer to AWS documentation for further inquiries.

Scraper

The workflow will proceed as follows:

A scraper will be scheduled to execute at least three times daily. During each run, it will perform a GET request to the RSS feeds provided by the BBC, retrieving the most recent news items. The raw data obtained from these feeds will then be parsed and transformed into a format suitable for persistence in the database (the database schema will be detailed in a subsequent section).

Once the relevant information is extracted and stored, each news headline will be enqueued for further processing in downstream tasks.

Simultaneously, the scraper will also retrieve the current trending topics from the Google Trends "Current Trends" section. This operation will involve approximately six API calls per day at a minimum, which is considered manageable within system constraints.

Queue

To ensure a high-performance user experience, the system will adopt an asynchronous architecture so that end users do not have to wait for scraping and data processing to complete.

All incoming news headlines will be enqueued in an Amazon SQS (Simple Queue Service) queue. A dedicated worker process will then consume messages from the queue, extracting up to three keywords from each headline. These keywords will be combined into a composed query which is then submitted to the Google Trends API to retrieve historical data on the relative popularity of that specific news item.

Upon receiving the trend data, the system will store the estimated trend values and other relevant metrics in the database for subsequent retrieval and analysis.

By outsourcing the query handling and trend data retrieval to the Google Trends API, the backend system's complexity is reduced, improving maintainability and performance. This approach also allows for more precise data aggregation, resulting in enhanced accuracy in estimating news popularity.

Worker

After enqueueing the corresponding data into SQS through the lambda scraper jobs, a worker will poll for messages and extract each message's headline, the worker will extract at most 3 keywords from each headline.

Each keyword's popularity will be estimated through the SerpApi's google trends endpoint, which provides sufficient information to collect data regarding the headline's popularity.

Though SerpApi has one caveat, the free tier is limited to 250 calls per month, which means, there would need to be a tier upgrade to increase the API calls cap. This is because, approximately, there are 300 headlines per day, each containing 3 keywords, in the worst case, every headline has different keywords, meaning results won't be cached and thus a new api call will be needed. The solution won't be able to reach the end of month without

consuming all the free tier's allowed API calls. Therefore, this solution will need extra payment.

Email Service

Each user will receive a daily email containing a curated list of the most relevant news articles from the previous day. To facilitate this, a dedicated scheduled job—hereafter referred to as the **Reporter**—will be implemented to manage the generation and delivery of these messages.

The Reporter will perform the following tasks:

1. Retrieve the relevant news data from the database, aggregating and formatting the information into a user-friendly and readable email layout.
2. Verify the configured recipient email addresses for each user.
3. Dispatch the email at the pre-established time of day, ensuring consistent delivery schedules.
4. Log each sent email in the database, recording metadata including timestamps and key details for auditing and tracking purposes.

This design guarantees that users receive timely, tailored news summaries each day and provides the flexibility to reschedule delivery times as needed.

Admin API

The admin API provides endpoints to access the information extracted by the solution, such as, headlines, trends, keywords and news reports. This a single gateway is provided to access the important data and it can be served to any of the several end recipients. Also, it allows to change settings within the solution's behavior.

Admin UI

The admin UI offers and interactive interface to configure the project's preferences, for example, registering new target emails, summary send time, number of headlines sent, etc.

This way, a simple alternative is provided to manage the solution's behavior, and eliminates the need to use programming languages to configure the project.

Pricing

In the long run, lambda will increase the app's maintenance cost if the solution scale increases significatively, for reference, lambda costs $0.20 per 1 million requests and $0.0000166667 for every GB-second of compute time, while EC2's most affordable option has a cost of approximately $2.82 per month. When scaling for a large user base this challenge must be addressed.

Lambda becomes cost-prohibitive when:

- Consistent, predictable traffic patterns
- Long-running processes (>15 minutes)
- High-memory, compute-intensive workloads running continuously
- Monthly compute costs exceed equivalent EC2 Reserved Instance pricing

EC2 becomes inefficient when:

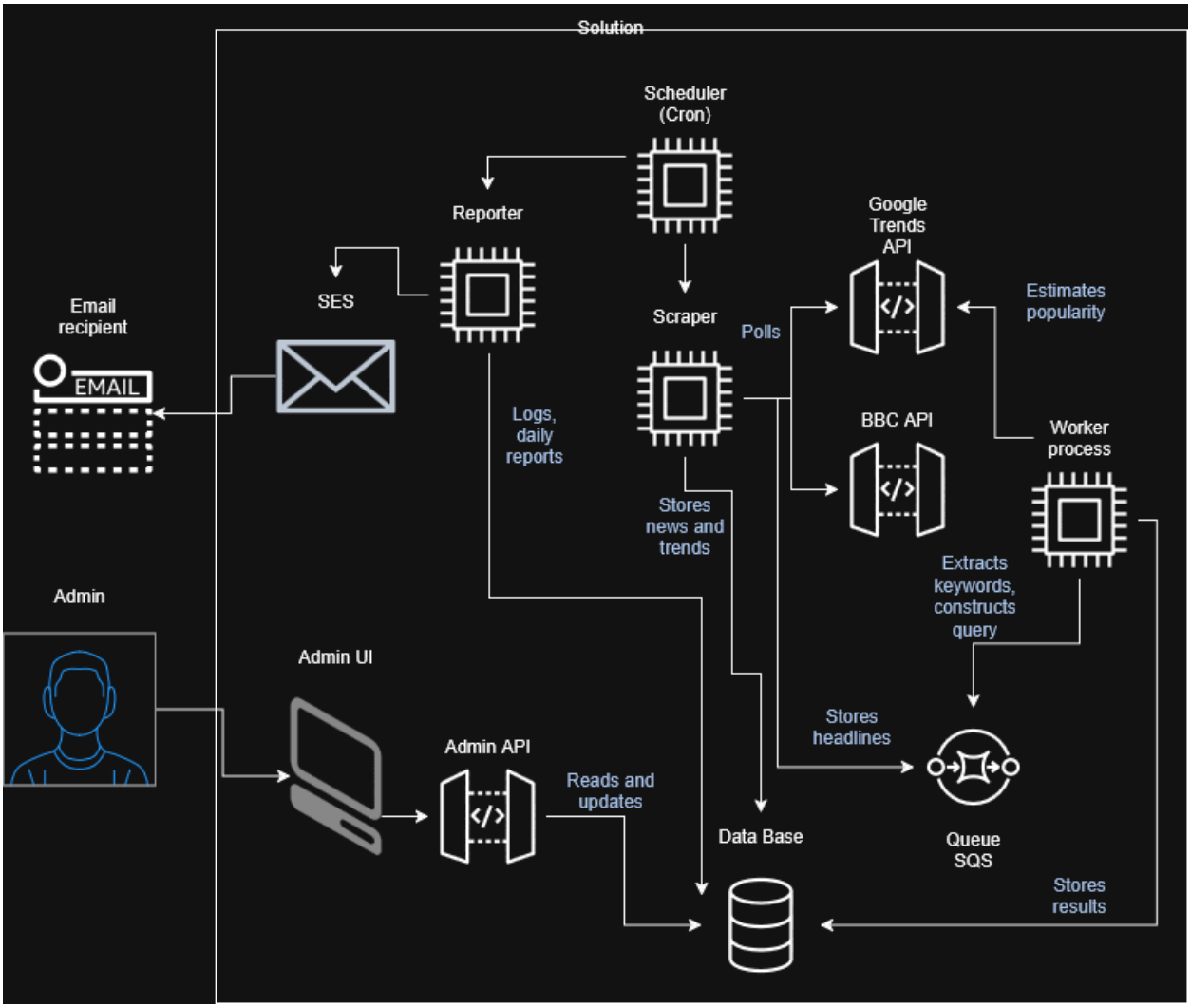
- Highly variable or unpredictable traffic
- Low utilization rates (<20% average)
- Frequent scaling requirements
- Small team with limited operational expertise

SerpApi's pricing:

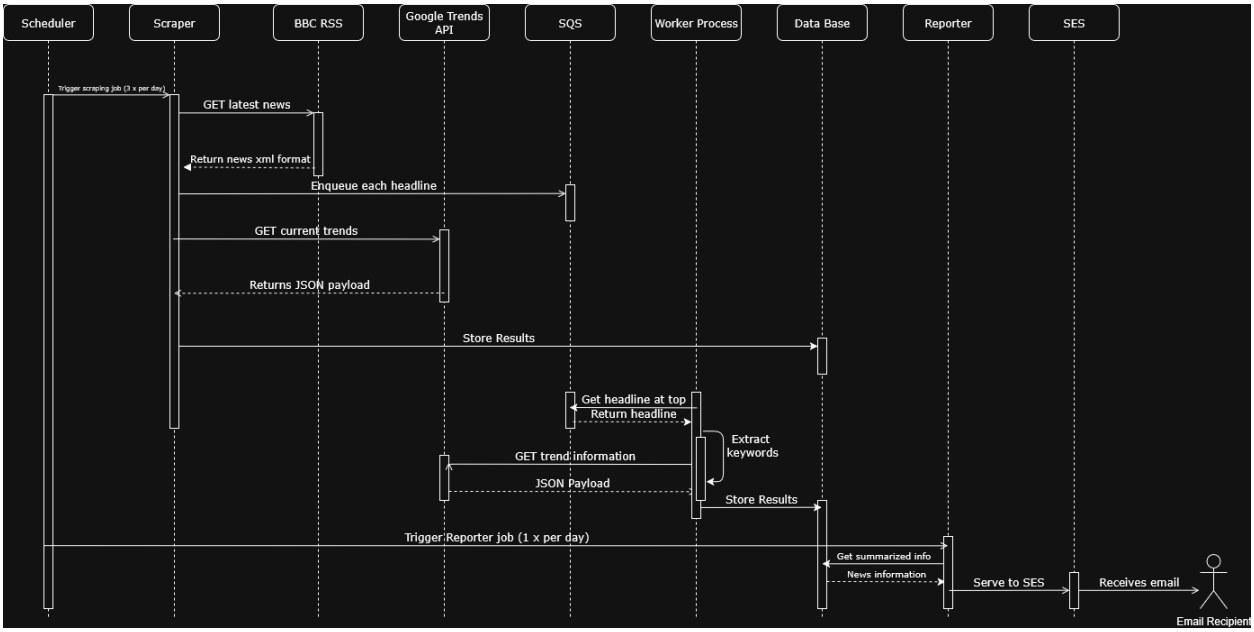
The developer tier offers sufficient API calls per month to suffice for this solution's requirements, this tier costs 75 USD per month offering 5,000 searches per month. This rate fits our requirements.

Architecture Diagram

In the following diagram you can observe a simplified version of what the system would look like.


Solution
Scheduler (Cron)
Reporter
SES
Email recipient
EMAIL
Logs, daily reports
Scraper
Polls
Google Trends API
Estimates popularity
BBC API
Worker process
Stores news and trends
Extracts keywords, constructs query
Admin
Admin UI
Admin API
Reads and updates
Stores headlines
Queue SQS
Data Base
Stores results

Sequence Diagram



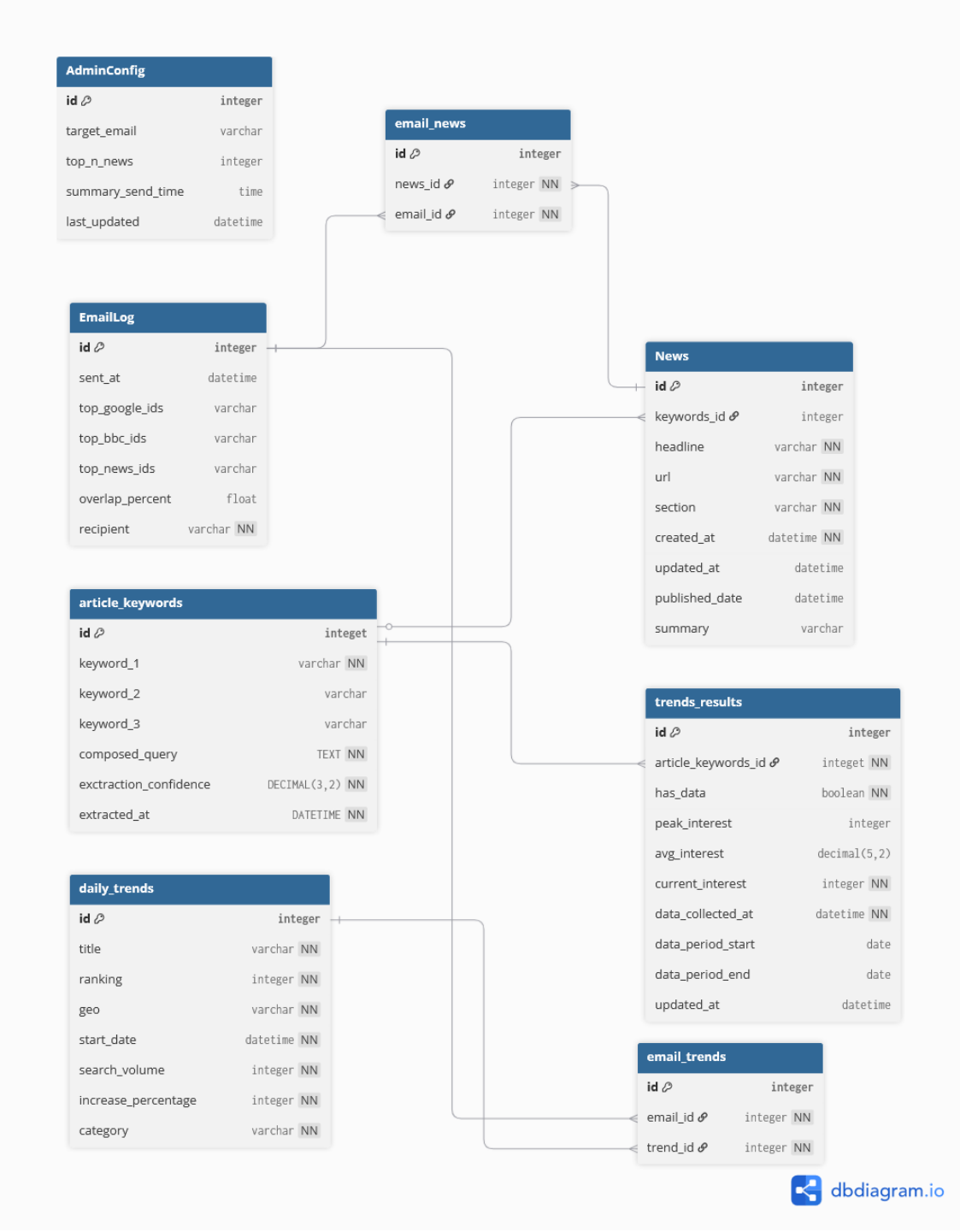
Summary

The solution will consist of the following components:

- Scheduler (CRON Jobs): The scheduler will assure that the information is up to date.
- Scraper: The scraper itself will get the initial information from Google Trends and BBC news, this includes headings and current trends.
- SQS Queue: Here the headlines will wait to be processed, this way we can estimate each headline's popularity asynchronously.
- Worker Process: This process will extract each headline from the queue, extract entities or keywords, and conform a composed query of at most 3 keywords to estimate the popularity of each headline through the Google Trends API.
- Data Base: All the information will be stored in a relational data base, including headlines and trends.
- Reporter: The reporter will extract data from the data base, summarize it, provide it a structure and serve it into the SES service.
- SES: The SES service will notify each user of the previous day's news and trends.
- Admin API: Through this API, users with admin permissions will be able to change the settings of the solution, for example, email recipients, cadency of the jobs, etc.
- Admin UI: Through the UI the admin user can access the Admin API, making the administration simpler.

Data Base

Entity Relation Diagram



References

Amazon Web Services. (n.d.). *EC2 Documentation*. Retrieved from AWS: https://docs.aws.amazon.com/ec2/

Amazon Web Services. (n.d.). *EC2 pricing*. Retrieved from AWS: https://aws.amazon.com/ec2/pricing/on-demand/

Amazon Web Services. (n.d.). *Lambda Documentation*. Retrieved from AWS: https://docs.aws.amazon.com/lambda/

Amazon Web Services. (n.d.). *Lambda pricing*. Retrieved from AWS: https://aws.amazon.com/lambda/pricing/