News Headlines API

Overview

The API matches news headlines to employees on a specific publication date.

Using the API

API has been developed in python3. **Python3 must be installed on your machine to be able to run it**.

Files included in project

- invoker.py - This python file makes a call to the API with publication date
- news_match.py - API that creates matching and outputs results.txt with all matches
- gen_employee.py - Python script to generate a text file with 10,000 employees
- gen_headline.py - Python script to generate a text file with 3,000 news headlines
- employee.txt - Text file that contains 10,000 random employees and their information
- headlines.txt - Text file that contains 3,000 random news headlines
- result.txt - Text file that is created by news_match.py with all headlines matches

Getting Started

Currently, the API is not on a cloud and needs to be executed on a local machine.

1. Download the API from GitHub
2. Make sure all files are in the same folder
3. Open terminal/cmd and navigate inside folder that contains all files
4. To invoke the API run invoker.py in terminal/cmd by typing "***python3 invoker.py***"
5. This will prompt you enter a publication date in the following format "***yyyy-mm-dd***"
6. This will take a couple of seconds to finish. result.txt will have been created with all news headlines published on that date, with a list of employees that should see the headline

API Calls

Default API Requirement

Default requirement of the API is to return all headlines on a specific date and match it with list of employees. This is done by calling function:
match_headline2employee(search_d).

search_d is the argument which is the publication date that is provided by the user in invoker.py. This function takes search_d and loops through all headlines and checks to see if search_d matches the publication date of the headline. If the dates match, the function now has to determine what employees match the headline, which is done by a few different criteria. The function then goes through all employees and checks if the headlines language matched the spoken language of the employee, and checks if the employees job roles or department matches the keyword in the headline. If there is a match this employee is stored under the headline and then next employee is checked.

Additional API Calls (Taking API to the Next Level)

The following functions have also been implemented in the API to allow headlines to be organized in additional ways.

Sorting All Headlines by Department:

Function match_dept2headline(search_d):

This function goes through all departments and lists all headlines if headline keyword matches the department. Argument search_d is optional, if user would like to see all headlines regardless of date they would call the api in the following way:

match_dept2headline(None)

Function will create a new text file called ***dept_headlines.txt***

Sorting All Headlines by Job Role:

Function match_job2headline(search_d):

This function goes through all job roles and lists all headlines if headline keyword matches the job role. Argument search_d is optional, if user would like to see all headlines regardless of date they would call the api in the following way:

match_job2headline(None)

Function will create a new text file called ***job_headlines.txt***

Sorting All Headlines by Language:

Function match_lang2headline(search_d):

This function goes through all languages and lists all headlines if headline language matches the current language. Argument search_d is optional, if user would like to see all headlines regardless of date they would call the api in the following way:

match_lang2headline(search_d):

Function will create a new text file called ***language_headlines.txt***

Additional Considerations

Handling 100 million employees and 300 million headlines across the globe.

Cloud

This API should live on Cloud solution such as IBM Cloud. Which would be distributed across multiple servers with a geographical location.

Load Balancer

Having a load balancer such as NGINX to distribute requests across servers based on users location. Having a load balancer would allow us to increase concurrent users and the reliability of the API.

NoSQL Database Management System

Employees and news headlines would be stored in a NoSQL Database such as Cassandra, rather than .txt files. NoSQL databases are designed to handle large amounts of data across many servers. This provides high availability without a single point of failure. Each employee would have their list of headlines stored 3 times. So that when an employee makes a request to see their matched headlines. The first cluster list to retrieve will be shown.