2021

CAB432 Assignment 2

CAB432

Assignment 2

Benjamin Semple N10533885 &

Joshua Paterson n10193197

11/1/2021

Contents

Introduction

Purpose & description

The application this report detail is Scenario 1 provided in the Assessment 2 brief. The purpose of the app is to filter live twitter messages by user provided tags/inputs and use those messages to perform live and in real time sentiment analysis to determine the overall sentiment of twitter messages with the provided tags at any given moment. This app provides value as it provides the ability to quickly get a general idea of the public's sentiment of any topic.

Services used

Twitter Standard Search API (v.1.1)

Returns a collection of relevant Tweets matching a specified query – may also be filtered based on popularity or geocoding however for this project filtering was based on tags.

Endpoint: `https://api.twitter.com/2/tweets/search/recent`

Docs: https://developer.twitter.com/en/docs/twitter-api/v1/tweets/search/api-reference/get-search-tweets

Python

To perform sentiment analysis on the tweets provided by the Twitter Standard Search API a python script is used. Note that the application is primarily a web application written in JavaScript. To perform sentiment analysis on the tweets within the python script two libraries are imported being textblob and vaderSentiment.

vaderSentiment

VADER (Valence Aware Dictionary and sEntiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media which works well on texts from other domains.

textblob

Textblob is an open-source python library for processing textual data. Used to convert messages into usable objects for vadarSentiment.

Use cases

US 1

As a	academic data scientist
I want	to get data from the site including a pre-processed opinion of the tweet
So that	allow for an easier analysis of twitter

US 2

As a	regular twitter user
I want	to see a general statement of opinions of a topic
So that	I can get that without having to read every single tweet

Both the above US cases have implemented these services by creating a query to the twitter api using user provided tags then using that data in a python program that performs sentiment analysis on individual messages then the application stores the results. These results are used to take an average of the polarity of sentiment on all messages provided to determine and then display results to users on screen.

Technical breakdown

Architecture

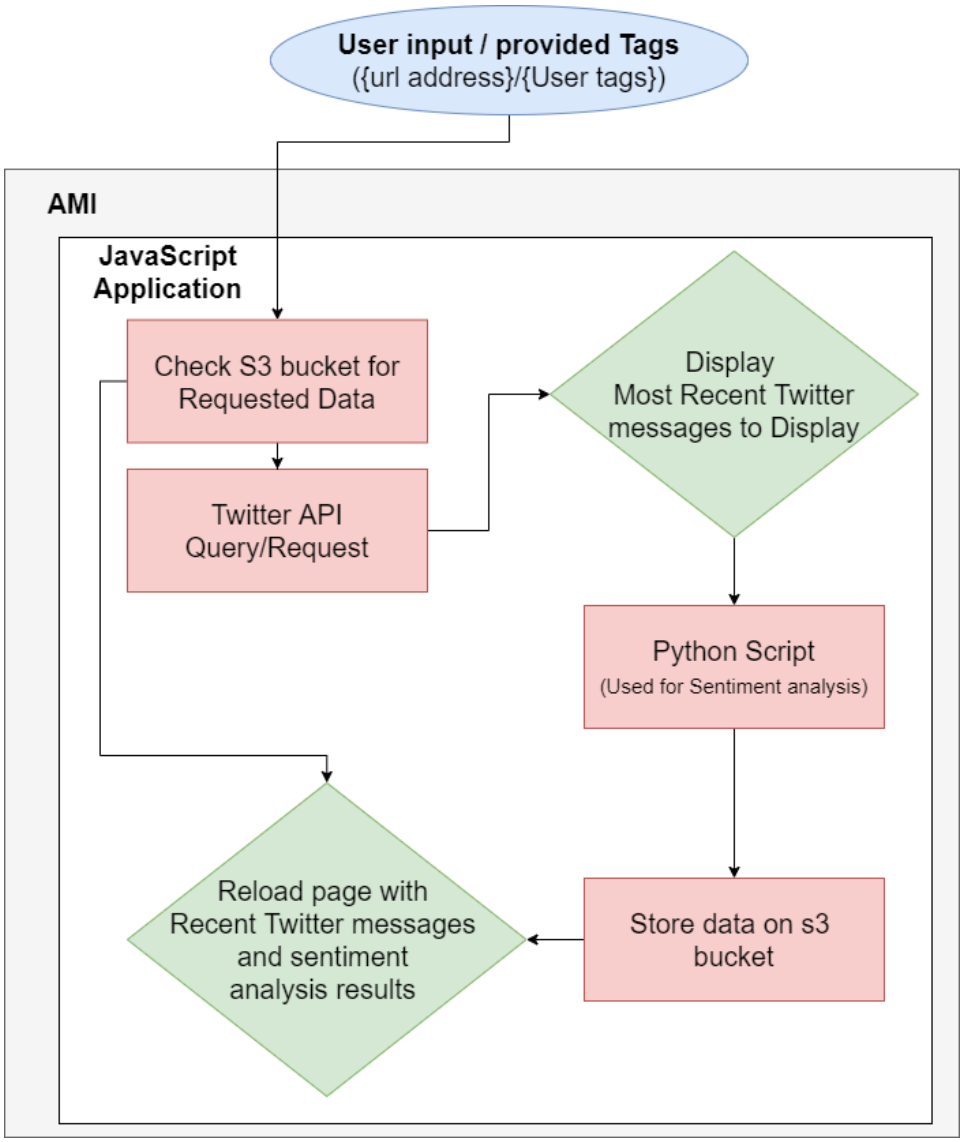


Figure 1: Application Data Flow Diagram

The architecture of the application is as seen above in figure 1. This application can receive user input at the URL level an example of this is below in figures 2 and 3. The user will insert the tags they wish to filter twitter messages by going to the applications URL address then inserting their search tags in after as seen in figure 2 then the application will access this information using the req variable.

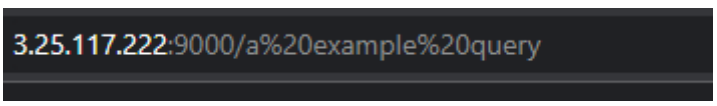


Figure 2: URL user input example

```
router.get("/:query", (req, res) => {
```

Figure 3: in code user input example

Once the application has been given an input from a user it will then check an AWS s3 bucket for both twitter messages and the sentiment of those messages. If the requested data is available, then that data will be imported and used to display the results to user's screens.

If the data is unavailable from the s3 bucket it will then send a request to the twitter API for the required twitter messages for processing. Then to allow the web application to remain responsive if sentiment analysis requires a high amount of time to process the live (or most recent) twitter messages are displayed on screen.

Once messages have been displayed the JavaScript web application will call a python script which has the function of taking in a single message then perform a sentiment analysis of the individual message then returns that messages polarity (positive or negative on a scale from -1 to 1) and its subjectivity (confidence in polarity from 0 to 100%). The JavaScript application will loop through all messages provided calling the python script for all and storing the results in memory. The application will then take messages with a high level of subjectivity then average their polarity to find an answer for twitters publics sentiment for messages with the users inputted tags.

With the results from the sentiment analysis the application will then store the results on the previously mentioned s3 bucket. This data will be stored on the bucket for 60 seconds. This low amount of time is necessary to allow users to retrieve current data while also providing performance benefits for high interest subjects.

After the data is stored on the s3 bucket it will then be used to display the same information on screen for users.

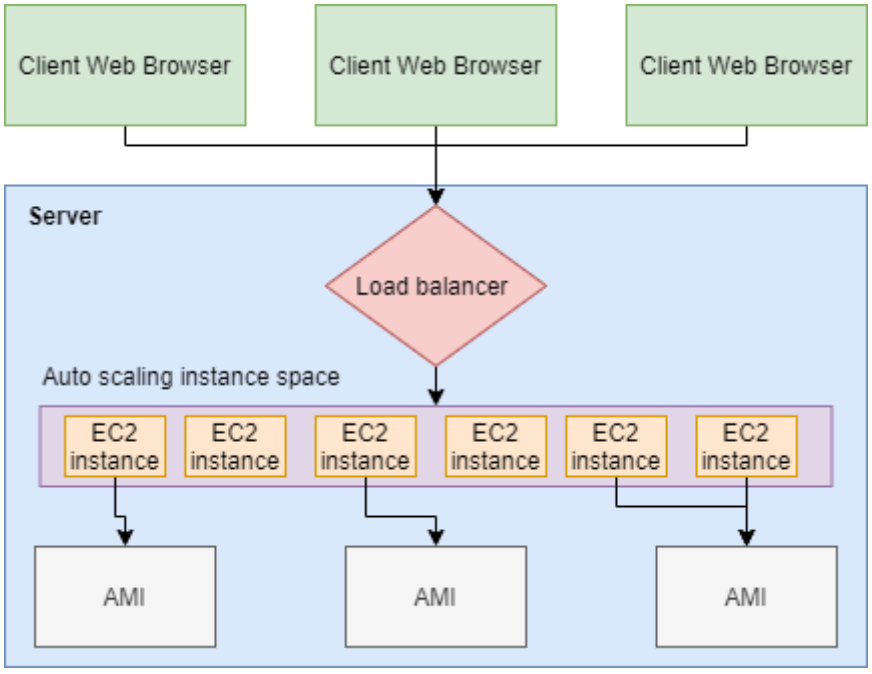


Figure 4: AWS Scaling Diagram

The above image or figure 4 details how the server will increase the network capacity when specific client requires higher levels then initially provided. As seen in the diagram there will be some number of client web browsers making request to the server these requests will be filtered through a load balancer which will provided a EC2 instance to each individual client the AMI which can be seen in figure 1 will then start processing the request. When an AMI will require more network usage the Auto scalier will then assign more EC2 instances to an AMI when required.

It should be noted that network capacity is the metric that was decided on as due to the limited number of tweets able to be taken from the twitter api without payment the amount of processing capacity is never able to exceed a instance limit however if a client sends many request to the server it can require higher amounts of network usage from instances and therefore will require scaling.

Client / server demarcation of responsibilities

The server in this application is performing most of the work on an AWS virtual machine that is created each time a user accesses the URL. This virtual machine will then perform the work required then send the display information or html and style to the user's system for their browser to handle and then display to them.

Response filtering / data object correlation

The data pass through the sentiment analysis python function is given a subjectivity rating which can be redefined as a certainty rating. Some messages where are not counted towards the overall sentiment of a topic due to a low certainty rating.

Scaling and Performance

The scaling in this application is based around the AWS auto scaling group. The more users who connect to the application, the more CPU power is required to run all the required calculations (from 25.7% to 9.9% of CPU usage) and therefore was the metric decided on to scale the application.

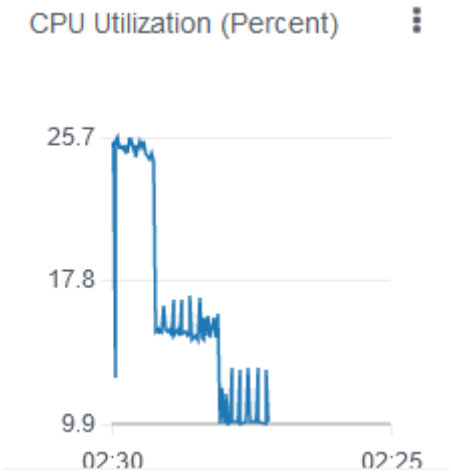


Figure 2: Network usage of server

As seen in the image below or figure 3 the auto scaling group for this application had a pool of 3 instances and as seen in the desired capacity and in services instances the application was able to respond to the increased demand for CPU usage by increasing the number of active instances to the desired capacity when require as seen because both graphs mirror each other. This will lead to improving the overall responsiveness of the application when faced with high demand. Note that the increase demand for this service was tested using Postman to simulate increased network usage and thus cause calculations and CPU use.

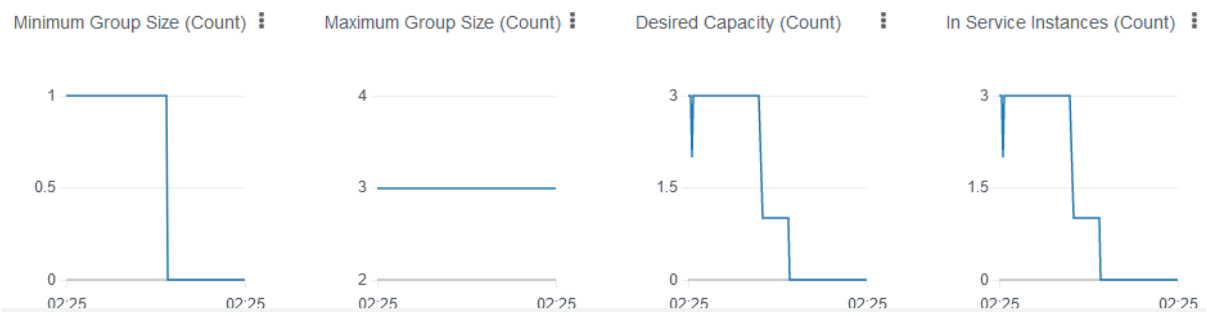


Figure 3: Scaling instances graphs

The image below shows the successful launching for EC2 instances in response to increase network demand.

Successful	Launching a new EC2 instance: i-062c9f25ca77ced31	At 2021-10-29T02:52:53Z a monitor alarm TargetTracking-semple-paterson2-AlarmHigh-715d7c67-2d44-4dc0-a966-a891f24ad99a in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 1 to 2. At 2021-10-29T02:52:59Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 1 to 2.	2021 October 29, 12:53:00 PM +10:00	2021 October 29, 12:58:32 PM +10:00
Successful	Terminating EC2 instance: i-062c9f25ca77ced31	At 2021-10-29T12:22:54Z a monitor alarm TargetTracking-semple-paterson2-AlarmLow-11f6a137-78a9-4f30-ab83-2bfe66e58e96 in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 2 to 1. At 2021-10-29T12:23:16Z an instance was taken out of service in response to a difference between desired and actual capacity, shrinking the capacity from 2 to 1. At 2021-10-29T12:23:16Z instance i-062c9f25ca77ced31 was selected for termination.	2021 October 29, 10:23:16 PM +10:00	2021 October 29, 10:30:40 PM +10:00
Successful	Terminating EC2 instance: i-03efbde9401abf840	At 2021-10-29T12:06:54Z a monitor alarm TargetTracking-semple-paterson2-AlarmLow-11f6a137-78a9-4f30-ab83-2bfe66e58e96 in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 3 to 2. At 2021-10-29T12:07:04Z an instance was taken out of service in response to a difference between desired and actual capacity, shrinking the capacity from 3 to 2. At 2021-10-29T12:07:04Z instance i-03efbde9401abf840 was selected for termination.	2021 October 29, 10:07:04 PM +10:00	2021 October 29, 10:14:32 PM +10:00
Successful	Launching a new EC2 instance: i-0fa191c8c6f9cd9fb	At 2021-10-29T03:01:53Z a monitor alarm TargetTracking-semple-paterson2-AlarmHigh-715d7c67-2d44-4dc0-a966-a891f24ad99a in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 2 to 3. At 2021-10-29T03:02:14Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 2 to 3.	2021 October 29, 01:02:16 PM +10:00	2021 October 29, 01:08:17 PM +10:00

Figure 4:EC2 launch log

Test plan

Task	Expected outcome	Result	Screenshot/s Appendix 1
Search an uncached query	The result should return however it may take some time due to the computation required	PASS	Image 1
Search a cached Query	The response should be faster as the computation is not required	PASS	Image 2
Search a cached query which has expired	Result should take longer as the results need to be updated	PASS	Image 3
Search an invalid search	An error page should show up	PASS	Image 4
Search for a query, manually clear the cache then search again	The page should refresh with different results	PASS	Image 5 &6
Get a query with a negative sentiment	A set of results with negative sentiment should show up	PASS	Image 8
Get a query with a neutral sentiment	A set of results with neutral sentiment should show up	PASS	Image 5 & 6
Get a query with a positive sentiment	A set of results with positive sentiment should show up	PASS	Image 7

Unresolved & persistent errors

An unresolved issue with the current version of the app is that the server is only capability of handling a certain amount of request at any given moment when flooded with request the server can send the wrong data to the wrong location. It is believed that the issue come form the fact that the server can only do one thing at a time if it becomes flooded by request it returns response as they become available which may not always be in the correct order. A second persistent error within this version of the application is that credentials for aws expire roughly each day causing the AMI used to need to be updated each day.

User guide

This application is simple to use. For demonstrational purposes let say you want to find out if people are generally against or for vaccines.

To accomplish this you will need the URL of the web application which is as follows.

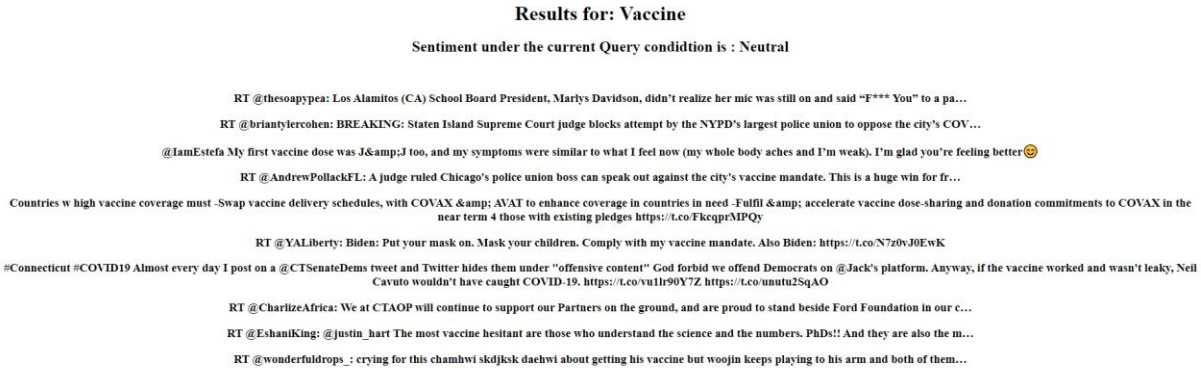
http://13.54.199.230:3000/

Insert the web address into a web browser of choices in this case chrome. then follow the URL with

/Vaccine

The full web address can be seen below. Note you may replace /vaccine with any tag or get information on that specific topic.

The URL will send you to the following webpage



Appendix 1 screenshots

Image 1

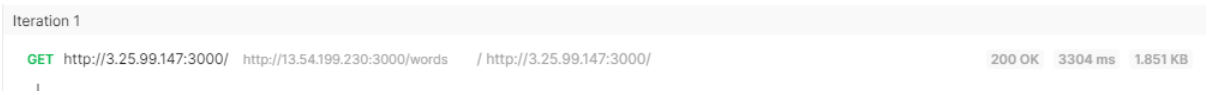


Image 2

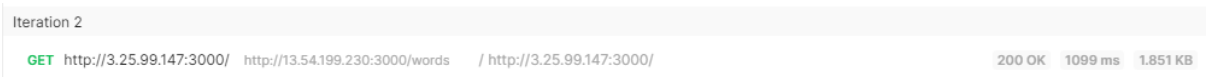


Image 3

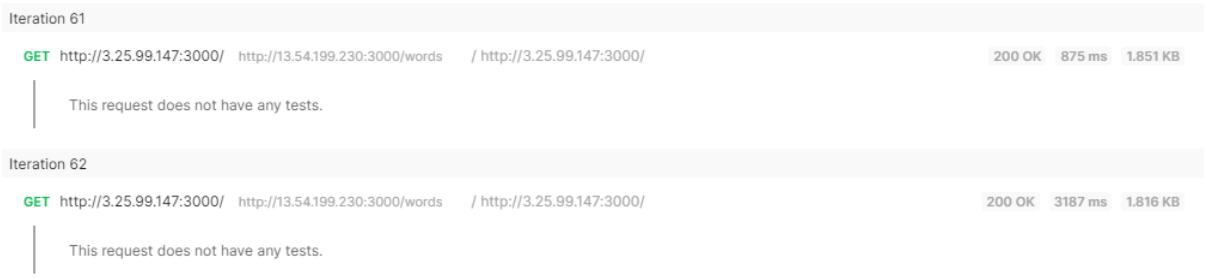


Image 4

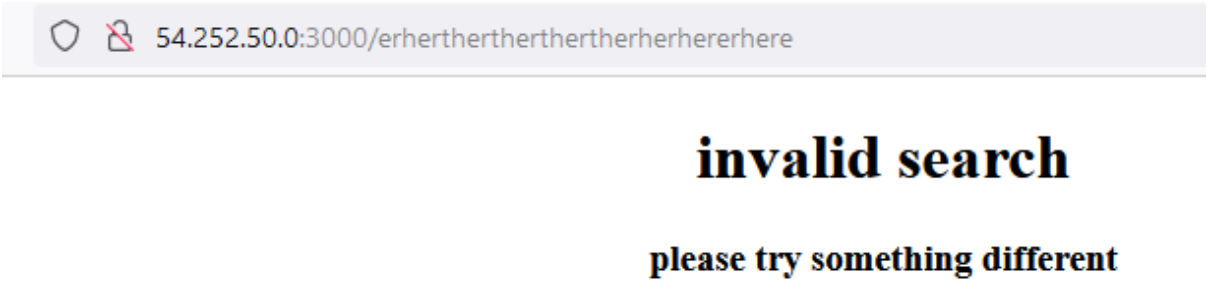


Image 5

Image 6

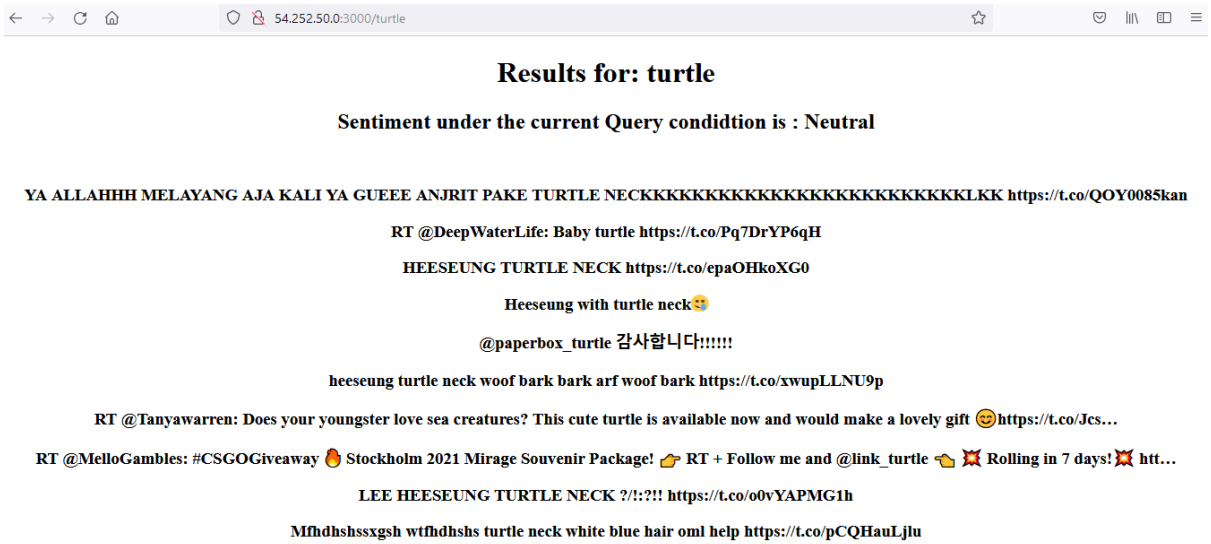
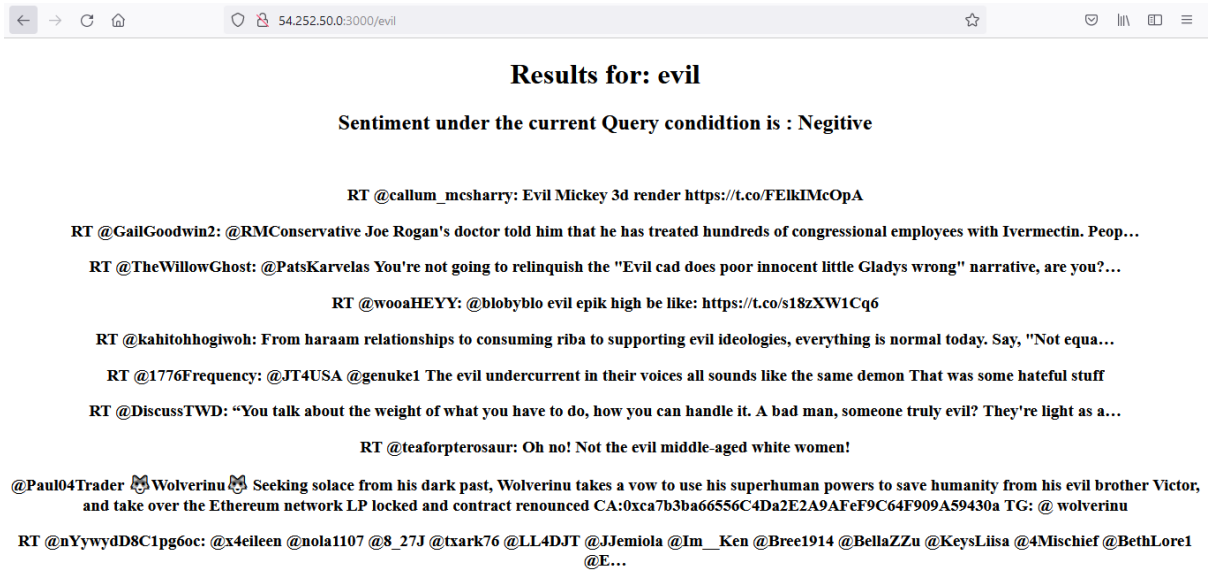


Image 7


54.252.50.0:3000/turtle
Results for: turtle
Sentiment under the current Query condidtion is : Positive
Kyeom turtle neck leather pnts exposed forehead. I'm loving his outfits every show https://t.co/1fnBySR3XF
RT @solosanctuarybo: 'hello elle' https://t.co/tVJIfEHs4S https://t.co/MPcRFqDTp5
heeseung in a turtle neck
@Anarqxista Many indigenous peoples believe the same thing. In many North American/Turtle Island native spiritual traditions 'animals' are nations unto themselves and even rocks, trees, water, etc. contain spirit/souls. I agree with you that it's a very powerful way to view our world.
Wonderful project & this project is very good & this projector has a lot of attractions. so hopefully the project will be better in the future and the cryptocurrency @trandang1687 @Turtle_Small_ #solana #BTC #Ethereum #Airdrop #SHIB #dogecoin https://t.co/xNe5uZ4Dc0
I caught a snapping turtle! How can it snap without fingers?
heeseung turtle neck https://t.co/V0606HTbR3
@turtle_rbx TYSMM
manifesting get hee with turtle neck selca
ETTSS ETTSS TURTLE NECK ETTTSS https://t.co/d7yH3cqji1

Image 8


54.252.50.0:3000/evil
Results for: evil
Sentiment under the current Query condidtion is : Negitive
RT @callum_mcsharry: Evil Mickey 3d render https://t.co/FElkIMcOpA
RT @GailGoodwin2: @RMConservative Joe Rogan's doctor told him that he has treated hundreds of congressional employees with Ivermectin. Peop…
RT @TheWillowGhost: @PatsKarvelas You're not going to relinquish the "Evil cad does poor innocent little Gladys wrong" narrative, are you?…
RT @wooaHEYY: @blobyblo evil epik high be like: https://t.co/s18zXW1Cq6
RT @kahitohhogiwoh: From haraam relationships to consuming riba to supporting evil ideologies, everything is normal today. Say, "Not equa…
RT @1776Frequency: @JT4USA @genuke1 The evil undercurrent in their voices all sounds like the same demon That was some hateful stuff
RT @DiscussTWD: "You talk about the weight of what you have to do, how you can handle it. A bad man, someone truly evil? They're light as a…
RT @teaforpterosaur: Oh no! Not the evil middle-aged white women!
@Paul04Trader Wolverinu Seeking solace from his dark past, Wolverinu takes a vow to use his superhuman powers to save humanity from his evil brother Victor, and take over the Ethereum network LP locked and contract renounced CA:0xca7b3ba66556C4Da2E2A9AFeF9C64F909A59430a TG: @ wolverinu
RT @nYywydD8C1pg6oc: @x4eileen @nola1107 @8_27J @txark76 @LL4DJT @JJemiola @Im__Ken @Bree1914 @BellaZZu @KeysLiisa @4Mischief @BethLore1 @E…