



CASE STUDY PRESENTATION

OLIVE CHAUDHURI
KALUVA PREETHI
MENTORED BY: PRATEEK SRIVASTAVA



www.sigmoid.com

Sigmoid Private and Confidential- Not intended for copy or circulation outside the organization.

PROBLEM STATEMENT

- The Twitter Producer should get data from Twitter based on some keywords and put them in any distributed messaging system.
- The Consumer component should get data from your messaging system.
- Insertion of the data into a distributed datastore after doing some basic transformations
- Write an API service which can be queried using any tool like Postman

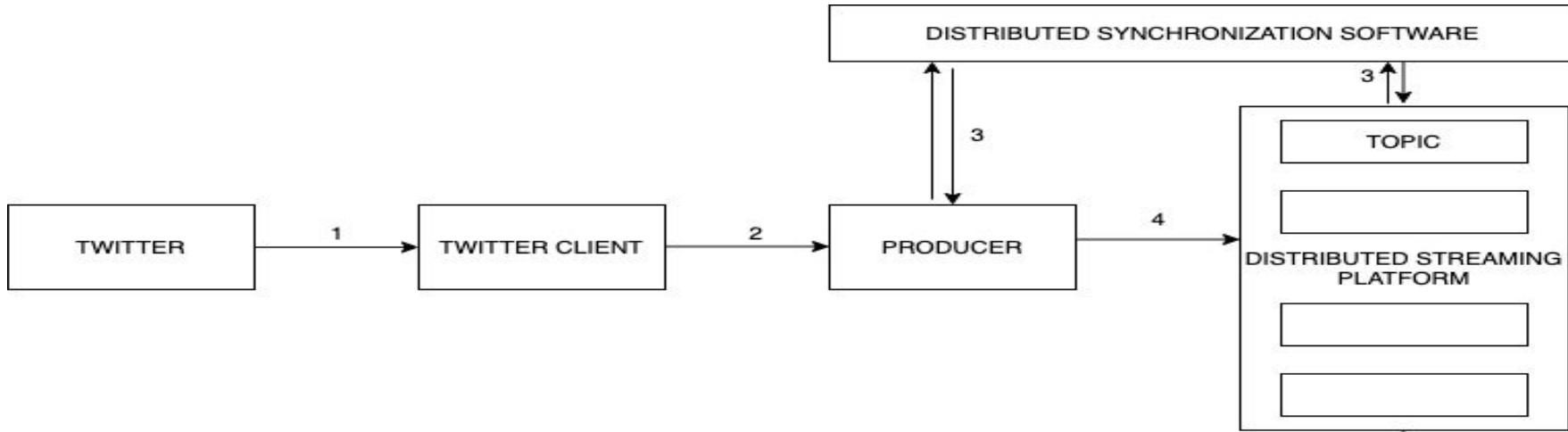
Implementation of following functionalities :

1. Need to find overall number of tweets on coronavirus (keywords can be virus/covid-19/corona etc etc) per country in the last 3 months.
2. Next, need to find overall number of tweets per country on a daily basis.
3. Also need to find the top 100 words occurring on tweets involving coronavirus. (words should be nouns/verbs and not involving common ones like the, is, are, etc etc).
4. Find the top 100 words occurring on tweets involving coronavirus on a per country basis.
5. Total no. of donations received towards COVID 19 country wise, in all the affected countries.

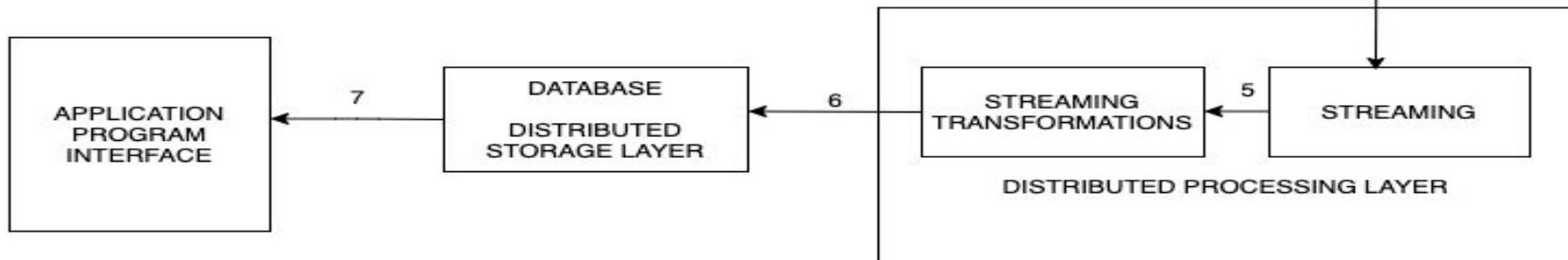
TECHNICAL REQUIREMENTS

1. Twitter developer account - For the twitter data.
2. Twitter Client - To fetch the data from twitter.
3. Streaming Producer - To store the data in the topic of distributed streaming platforms.
4. Distributed Streaming Service - To get the data, stream it, transform it in the distributed processing layer.
5. Distributed Datastore - to store all the data in a distributed storage layer.
6. REST API - to fetch the required data from the database (integration with UI)

FLOW DIAGRAM



FLOW DIAGRAM



TECHNOLOGY STACK

Twitter Streaming API



Apache Kafka



Apache Spark Stream



MongoDB

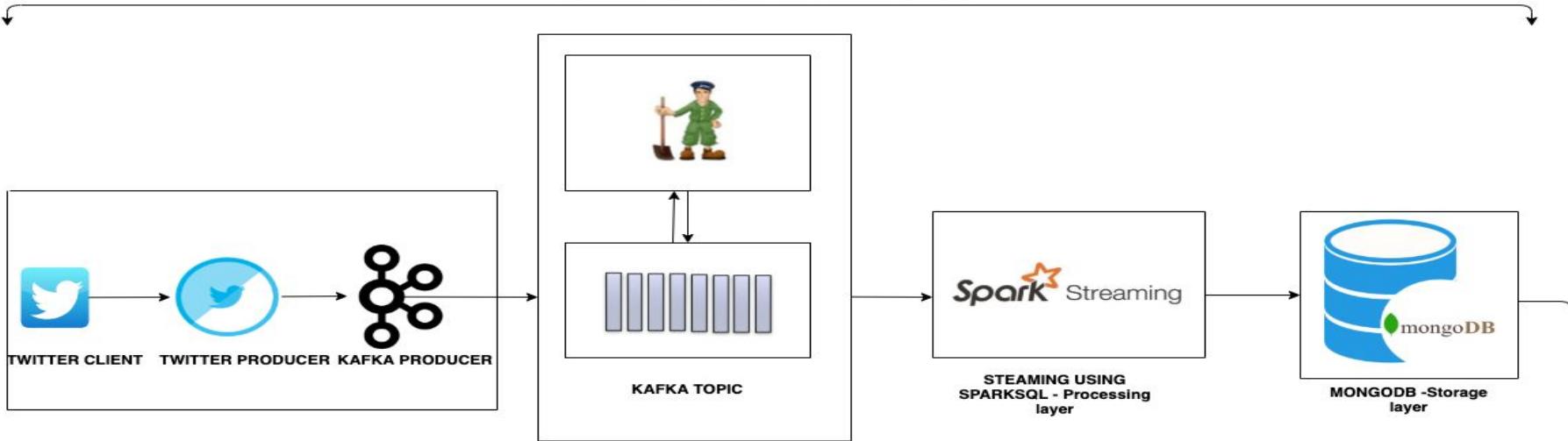


Flask

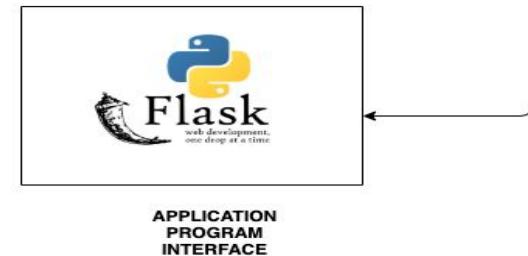


ARCHITECTURE DIAGRAM

COVID 19 ANALYTICS PIPELINE



REST API



APPROACH

Module 1 : Data Fetching

1. Creation of Twitter client
2. Creation of Kafka Producer

Module 2 : Streaming and Processing

1. Integration of spark shell and kafka topic using Spark Streaming
2. Setting format of the tweets.
3. Insertion of tweets into MongoDB collections

Module 3 : Rest API

1. Creation of a flask API in python
2. Performing queries on MongoDB

CORNER CASES HANDLED WHILE FORMATTING TWEETS

- Removal of invalid country names by performing a lookup in a set of predefined countries
- Extraction of tweeted text filtered using a set of stopwords
- Extraction of date formatted as mmm dd, yyyy
- In case of unavailability of the extracted date, timestamp(in ms) is used and converted to above date format

These three data attributes were inserted into 4 collections and an update of count is maintained. This is done to make the query faster.

SCHEMA OF MONGODB COLLECTIONS:

4 collections have been used to solve 5 queries

1st Collection:

_id : system generated unique object id for this row during creation

country : names of country

count : counter of the number of tweets updated for each country

<code>_id</code>	<code>country</code>	<code>count</code>
<code>5e99c2dac1344b2bc720050d</code>	"United States"	13049
<code>5e99c2dac1344b2bc720050e</code>	"India"	26921
<code>5e99c2dac1344b2bc7200510</code>	"United Kingdom"	9129
<code>5e99c2dac1344b2bc7200511</code>	"Paraguay"	680
<code>5e99c2dac1344b2bc7200512</code>	"Chile"	2875
<code>5e99c2dac1344b2bc72005db</code>	"Argentina"	5150
<code>5e99c2dac1344b2bc72005dd</code>	"Sweden"	540
<code>5e99c2dac1344b2bc720061c</code>	"England"	20005
<code>5e99c2dac1344b2bc720061e</code>	"France"	15561
<code>5e99c2dbc1344b2bc720076b</code>	"Ecuador"	2062
<code>5e99c2dbc1344b2bc7200787</code>	"Venezuela"	6520
<code>5e99c2dcc1344b2bc72007c9</code>	"Indonesia"	7100
<code>5e99c2dcc1344b2bc720087c</code>	"Colombia"	4195
<code>5e99c2dcc1344b2bc72008c9</code>	"Thailand"	476
<code>5e99c2dcc1344b2bc72008d6</code>	"Jamaica"	285
<code>5e99c2ddc1344b2bc72009d7</code>	"Portugal"	1199
<code>5e99c2ddc1344b2bc72009f9</code>	"Peru"	1735
<code>5e99c2dec1344b2bc7200a69</code>	"Denmark"	266
<code>5e99c2dec1344b2bc7200ac2</code>	"Nigeria"	11169
<code>5e99c2dec1344b2bc7200b1a</code>	"Malaysia"	2639
<code>5e99c2dec1344b2bc7200b67</code>	"Spain"	2017
<code>5e99c2dfc1344b2bc7200beb</code>	"Pakistan"	3902
<code>5e99c2dfc1344b2bc7200bf5</code>	"United Arab E...	898
<code>5e99c2dfc1344b2bc7200c33</code>	"Philippines"	1000
<code>5e99c2e0c1344b2bc7200cf5</code>	"Algeria"	26
<code>5e99c2e1c1344b2bc7200ecb</code>	"Georgia"	386
<code>5e99c2e2c1344b2bc7200f58</code>	"South Africa"	4855
<code>5e99c2e2c1344b2bc7200f5c</code>	"Canada"	4817
<code>5e99c2e2c1344b2bc7200fb1</code>	"Brazil"	1430
<code>5e99c2e2c1344b2bc7200fd1</code>	"Kenya"	2452

2nd Collection:

_id : system generated unique object id for this row during creation

country : names of country

date : date of tweet

count : counter of the number of tweets updated for each country and date

<u>id</u>	country	date	count
5e99c2dac1344b2bc7200559	Chile	"Apr 10, 2020	274
5e99c2dac1344b2bc720055b	India	"Apr 10, 2020	1010
5e99c2dac1344b2bc720055c	United States	"Apr 10, 2020	1070
5e99c2dac1344b2bc720055d	United Kingdom	"Apr 10, 2020	366
5e99c2dac1344b2bc720055e	Paraguay	"Apr 10, 2020	76
5e99c2dac1344b2bc72005e7	Argentina	"Apr 10, 2020	364
5e99c2dac1344b2bc72005e9	Sweden	"Apr 10, 2020	27
5e99c2dac1344b2bc7200625	England	"Apr 10, 2020	895
5e99c2dac1344b2bc720062a	France	"Apr 10, 2020	763
5e99c2dbc1344b2bc7200779	Ecuador	"Apr 10, 2020	206
5e99c2dbc1344b2bc7200793	Venezuela	"Apr 10, 2020	477
5e99c2dcc1344b2bc72007d5	Indonesia	"Apr 10, 2020	135
5e99c2dcc1344b2bc7200887	Colombia	"Apr 10, 2020	340
5e99c2dcc1344b2bc72008d1	Thailand	"Apr 10, 2020	31
5e99c2dcc1344b2bc72008e1	Jamaica	"Apr 10, 2020	17
5e99c2ddc1344b2bc72009e1	Portugal	"Apr 10, 2020	39
5e99c2ddc1344b2bc7200a00	Peru	"Apr 10, 2020	221
5e99c2dec1344b2bc7200a70	Denmark	"Apr 10, 2020	16
5e99c2dec1344b2bc7200aca	Nigeria	"Apr 10, 2020	454
5e99c2dec1344b2bc7200b27	Malaysia	"Apr 10, 2020	147
5e99c2dec1344b2bc7200b72	Spain	"Apr 10, 2020	95
5e99c2dfc1344b2bc7200bf3	Pakistan	"Apr 10, 2020	115
5e99c2dfc1344b2bc7200c00	United Arab Emirates	"Apr 10, 2020	29
5e99c2dfc1344b2bc7200c3c	Philippines	"Apr 10, 2020	15
5e99c2e0c1344b2bc7200cff	Algeria	"Apr 10, 2020	3
5e99c2e1c1344b2bc7200ed5	Georgia	"Apr 10, 2020	49
5e99c2e2c1344b2bc7200f62	South Africa	"Apr 10, 2020	250
5e99c2e2c1344b2bc7200f65	Canada	"Apr 10, 2020	418
5e99c2e2c1344b2bc7200fb7	Brazil	"Apr 10, 2020	73
5e99c2e2c1344b2bc7200fd9	Kenya	"Apr 10, 2020	163

3rd Collection:

_id : system generated unique object id for this row during creation

word : individual words extracted from filtered tweet

count : counter of the number of words updated for each record

<u>_id</u>	<u>word</u>	<u>count</u>
5e99c2dac1344b2bc7200531	"_ americans	2782
5e99c2dac1344b2bc7200532	"_ always	670
5e99c2dac1344b2bc7200533	"_ mazzoleni	19
5e99c2dac1344b2bc7200534	"_ within	454
5e99c2dac1344b2bc7200535	"_ indians	347
5e99c2dac1344b2bc7200536	"_ con	7112
5e99c2dac1344b2bc7200537	"_ better	887
5e99c2dac1344b2bc7200538	"_ dont	2155
5e99c2dac1344b2bc720055a	"_ data	1202
5e99c2dac1344b2bc720055f	"_ hombre	167
5e99c2dac1344b2bc7200560	"_ recordings	12
5e99c2dac1344b2bc7200561	"_ perfil	7
5e99c2dac1344b2bc7200562	"_ aclara	16
5e99c2dac1344b2bc7200563	"_ extra	285
5e99c2dac1344b2bc7200564	"_ trump	10683
5e99c2dac1344b2bc7200565	"_ employment	69
5e99c2dac1344b2bc7200566	"_ home	3716
5e99c2dac1344b2bc7200567	"_ pacientes	917
5e99c2dac1344b2bc7200568	"_ office	369
5e99c2dac1344b2bc7200569	"_ trust	518
5e99c2dac1344b2bc720056a	"_ frecuente	5
5e99c2dac1344b2bc720056b	"_ take	3294
5e99c2dac1344b2bc720056c	"_ precaution	22
5e99c2dac1344b2bc720056d	"_ last	2487
5e99c2dac1344b2bc720056e	"_ sobrepeso	5
5e99c2dac1344b2bc720056f	"_ lost	1820
5e99c2dac1344b2bc7200570	"_ del	12577
5e99c2dac1344b2bc7200571	"_ two	1990
5e99c2dac1344b2bc7200572	"_ help	3019
5e99c2dac1344b2bc7200573	"_ unprepared	43

4th Collection:

_id : system generated unique object id for this row during creation

country : name of country

word : individual words extracted from filtered tweet

count : counter of the number of words for each country updated for each record

_id	country	word	count
5e9985acf3d...	Uganda	relief	30
5e9985acf3d...	Uganda	mps	25
5e9985acf3d...	France	trolley	1
5e9985adf3d...	France	usage	2
5e9985adf3d...	United States	given	28
5e9985adf3d...	France	problem	7
5e9985adf3d...	France	produits	10
5e9985adf3d...	United States	fact	146
5e9985adf3d...	France	via	281
5e9985adf3d...	France	psychoactifs	1
5e9985adf3d...	United States	china	617
5e9985adf3d...	France	les	1998
5e9985adf3d...	United States	moved	11
5e9985adf3d...	France	addictions	1
5e9985adf3d...	United States	quickly	67
5e9985adf3d...	United States	shut	73
5e9985adf3d...	France	leurs	48
5e9985adf3d...	United States	travel	50
5e9985adf3d...	France	promouvoir	1
5e9985adf3d...	United States	domestically	10
5e9985adf3d...	France	des	1431
5e9985adf3d...	United States	wuhan	331
5e9985adf3d...	France	facteurs	6
5e9985adf3d...	France	pro	9
5e9985adf3d...	United States	rest	24
5e9985adf3d...	Germany	beat	2
5e9985adf3d...	Zimbabwe	corona	56
5e9985adf3d...	Germany	guyssss	1
5e9985adf3d...	Zimbabwe	joke	2
5e9985adf3d...	South Africa	national	35
5e9985adf3d...	South Africa	credit	13

CORNER CASES HANDLED IN THE API

- “Page Not found-404” error and “Database Connection Failure - 500” handled with an error handler that displays a default message.
- For the questions we have solved we have given an additional option to enter a country name as an input in the URL.
- Country name can be given in lowercase or uppercase with an underscore in between two words.
- Date can also be entered with underscores between the month, date and year in the same order.

API SCHEMA

- localhost:5000/tweets_per_country
- localhost:5000/tweets_per_country/<country_name>
- localhost:5000/tweets_per_country_date
- localhost:5000/tweets_per_country_date/<country_name>
- localhost:5000/tweets_per_country_date/date/<date>
- localhost:5000/tweets_per_country_date/date_country/<country_name>/<date>
- localhost:5000/overall_top_100words
- localhost:5000/top_100words_country
- localhost:5000/top_100words_country/<country_name>
- localhost:5000/overall_donations_country
- localhost:5000/overall_donations_country/<country_name>

localhost:5000/tweets_per_country : Finding count of overall tweets per country.

The screenshot shows the Postman application interface. In the top navigation bar, there are buttons for 'New', 'Import', 'Runner', and 'Sign In'. The 'My Workspace' dropdown is open, and the 'Invite' button is visible. On the left side, there's a sidebar with 'History' selected, showing a list of recent API calls. Below the sidebar, there's a 'Launchpad' section with a 'GET' request to 'http://127.0.0.1:5000/tweets_per_country'. The main area displays the API response in various formats: Pretty, Raw, Preview, Visualize, and JSON. The JSON response is as follows:

```
1 [  
2   {  
3     "count": 26921,  
4     "country": "India"  
5   },  
6   {  
7     "count": 20005,  
8     "country": "England"  
9   },  
10  {  
11    "count": 15561,  
12    "country": "France"  
13  },  
14  {  
15    "count": 13049,  
16    "country": "United States"  
17  },  
18  {  
19    "count": 11169,  
20    "country": "Nigeria"  
21  },  
22  {  
23    "count": 9129,  
24    "country": "United Kingdom"  
25  },  
26  {  
27    "count": 7100,  
28    "country": "Indonesia"  
29  },  
30  {  
31    "count": 6520,  
32    "country": "Venezuela"  
33  },  
34  {  
35    "count": 5150,  
36    "country": "Brazil"  
37  }]
```

At the bottom of the screen, there are icons for 'Bootcamp', 'Inbox', and 'Help'.

localhost:5000/tweets_per_country/<country_name> : Finding count of overall tweets per country for a particular searched country.

The screenshot shows the Postman application interface. On the left, there's a sidebar with tabs for 'History' (which is selected), 'Collections', and 'APIs'. Below the tabs, there are buttons for 'Save Responses' and 'Clear all'. The main workspace is titled 'Launchpad' and contains a single request card. The request method is 'GET', the URL is 'http://127.0.0.1:5000/tweets_per_country/united_states', and the status is 'Success' (indicated by a green dot). To the right of the card are 'Send' and 'Save' buttons. Below the card, there are tabs for 'Pretty', 'Raw', 'Preview', 'Visualize', and 'JSON' (which is currently selected). The JSON response is displayed as follows:

```
1 [  
2 {  
3   "count": 13049,  
4   "country": "United States"  
5 }  
6 ]
```

At the bottom of the interface, there are several icons for navigation and search, along with a 'Bootcamp' link and a help icon.

localhost:5000/tweets_per_country_date : Finding overall count of tweets per country on a daily basis.

Screenshot of a REST API testing tool interface showing a successful GET request to `http://127.0.0.1:5000/tweets_per_country_date`.

The response is displayed in JSON format:

```
1 [
2   {
3     "count": 8464,
4     "country": "India",
5     "date": "Apr 19, 2020"
6   },
7   {
8     "count": 7403,
9     "country": "India",
10    "date": "Apr 18, 2020"
11  },
12  {
13    "count": 5991,
14    "country": "India",
15    "date": "Apr 16, 2020"
16  },
17  {
18    "count": 5212,
19    "country": "India",
20    "date": "Apr 15, 2020"
21  },
22  {
23    "count": 4539,
24    "country": "India",
25    "date": "Apr 14, 2020"
26  },
27  {
28    "count": 4059,
29    "country": "India",
30    "date": "Apr 17, 2020"
31  },
32  {
33    "count": 3953,
34    "country": "India",
35    "date": "Apr 13, 2020"
36 }
```

The left sidebar shows a history of previous requests, all of which were successful (HTTP 200 OK).

localhost:5000/tweets_per_country_date/<country_name> : Finding count of overall tweets per country on a daily basis for a particular searched country.

The screenshot shows the Postman application interface. The left sidebar has tabs for 'History' (selected), 'Collections', and 'APIs'. Below 'History' are buttons for 'Save Responses' and 'Clear all'. The main area shows a 'Launchpad' with a single item: 'GET http://127.0.0.1:5000/api/query2'. The 'Pretty' tab is selected in the response view, displaying the following JSON data:

```
1 [
2   {
3     "count": 2245,
4     "country": "United Kingdom",
5     "date": "Apr 19, 2020"
6   },
7   {
8     "count": 1050,
9     "country": "United Kingdom",
10    "date": "Apr 18, 2020"
11  },
12  {
13    "count": 835,
14    "country": "United Kingdom",
15    "date": "Apr 17, 2020"
16  },
17  {
18    "count": 834,
19    "country": "United Kingdom",
20    "date": "Apr 15, 2020"
21  },
22  {
23    "count": 751,
24    "country": "United Kingdom",
25    "date": "Apr 13, 2020"
26  },
27  {
28    "count": 693,
29    "country": "United Kingdom",
30    "date": "Apr 16, 2020"
31  },
32  {
33    "count": 554,
34    "country": "United Kingdom",
35    "date": "Apr 14, 2020"
36  }
]
```

The top right of the interface shows 'My Workspace' and 'Invite' buttons. The top right corner includes icons for notifications, a heart, and 'Sign In', along with a gear icon for settings.

localhost:5000/tweets_per_country_date/date/<date> : Finding count of overall tweets per country on a daily basis for a particular searched date.

The screenshot shows the Postman interface with the following details:

- Header Bar:** Includes "New", "Import", "Runner", "Launchpad", "My Workspace", "Invite", and "Sign In".
- Left Sidebar:** "History" is selected, showing a list of recent requests. Other tabs include "Collections" and "APIs". A "Save Responses" toggle is also present.
- Request Section:** Method is "GET", URL is "http://127.0.0.1:5000/tweets_per_country_date/date/Apr_18_2020".
- Response Section:** Response type is "JSON". The response body is a JSON array of objects, each representing a country's tweet count on April 18, 2020. The data is as follows:

```
[{"count": 7403, "country": "India", "date": "Apr 18, 2020"}, {"count": 3847, "country": "Nigeria", "date": "Apr 18, 2020"}, {"count": 3192, "country": "Indonesia", "date": "Apr 18, 2020"}, {"count": 2737, "country": "United States", "date": "Apr 18, 2020"}, {"count": 2131, "country": "England", "date": "Apr 18, 2020"}, {"count": 1816, "country": "Venezuela", "date": "Apr 18, 2020"}, {"count": 1444, "country": "France", "date": "Apr 18, 2020"}]
```

localhost:5000/tweets_per_country_date/date_country/<country_name><date> : Finding count of overall tweets per country on a daily basis for a particular searched date and country.

The screenshot shows the Postman application interface. At the top, there are buttons for 'New', 'Import', 'Runner', and a dropdown for 'My Workspace'. On the right side, there are icons for 'Sign In' and other user settings. Below the header, there's a search bar labeled 'Filter' and tabs for 'History', 'Collections', and 'APIs'. A 'Save Responses' toggle and a 'Clear all' button are also present. The main workspace shows a 'Launchpad' section with a GET request to 'http://127.0.0.1:5000/api/query2'. The URL is expanded to 'http://127.0.0.1:5000/tweets_per_country_date/date_country/pakistan/Apr_18_2020'. To the right of the URL are 'Send' and 'Save' buttons. Below the URL, there are tabs for 'Pretty', 'Raw', 'Preview', 'Visualize', and 'JSON'. The 'Pretty' tab is selected, displaying the JSON response in a multi-line, indented format. The response is as follows:

```
1 {
2   "count": 961,
3   "country": "Pakistan",
4   "date": "Apr 18, 2020"
5 }
```

On the left side of the workspace, a list of previous API requests is shown, each with a 'GET' method and a full URL. The requests include various dates and countries such as Pakistan, Russia, Sri Lanka, United Kingdom, India, and United States, along with some placeholder dates like 'Apr_18+21'.

localhost:5000/overall_top_100words : Finding the top 100 words occurring on tweets.

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'History' selected, showing a list of recent API calls. In the center, a 'Launchpad' tab is active, displaying a GET request to 'http://127.0.0.1:5000/overall_top_100words'. The response is shown in a JSON editor with line numbers. The JSON array contains 100 objects, each representing a word and its count. The first few items are:

```
[{"count": 61142, "word": "virus"}, {"count": 50165, "word": "corona"}, {"count": 17263, "word": "que"}, {"count": 13242, "word": "people"}, {"count": 10715, "word": "coronavirus"}, {"count": 9917, "word": "amp"}, {"count": 9067, "word": "china"}, {"count": 7727, "word": "para"}, {"count": 7635, "word": "covid"}]
```

At the bottom of the screen, there are various navigation and status icons.

localhost:5000/top_100words_country: Finding top 100 words per country.

The screenshot shows the Postman application interface. On the left, there's a sidebar with tabs for 'History' (which is selected), 'Collections', and 'APIs'. Below the tabs are buttons for 'Save Responses' and 'Clear all'. The main area has a 'Launchpad' tab and a 'GET' request to 'http://127.0.0.1:5000/top_100words_country'. The response is displayed in a code editor with line numbers, showing a JSON object with an '_id' field and a 'top100_words_forThisCountry' array containing 35 words. The JSON is as follows:

```
1  {
2    "_id": "Afghanistan",
3    "top100_words_forThisCountry": [
4      "test",
5      "would",
6      "like",
7      "whether",
8      "destination",
9      "country",
10     "requires",
11     "certificates",
12     "yousuf",
13     "aryabi",
14     "one",
15     "died",
16     "days",
17     "ago",
18     "brother",
19     "says",
20     "family",
21     "members",
22     "also",
23     "lost",
24     "india",
25     "use",
26     "saarc",
27     "emergency",
28     "fund",
29     "send",
30     "antimalarial",
31     "drug",
32     "afghanistan",
33     "please",
34     "download",
35     "malaria"
36   ]
37 }
```

At the bottom of the screen, there are icons for 'Bootcamp', 'Help', and 'Feedback'.

localhost:5000/top_100words_country/<country_name>: Finding top 100 words per country for a particular searched country.

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'History' selected, displaying a list of previous requests. The main area shows an 'Untitled Request' with a GET method and URL `http://127.0.0.1:5000/top_100words_country/albania`. The 'Params' tab is active, showing a single parameter 'Key' with value 'Value'. The 'Body' tab is selected, showing the JSON response:

```
1 {
2     "_id": "Albania",
3     "top100_words_forThisCountry": [
4         "doctors",
5         "work",
6         "telling",
7         "never",
8         "exact",
9         "vaccine",
10        "strain",
11        "virus",
12        "life",
13        "neve",
14        "stresin",
15        "nga",
16        "bruksel",
17        "ministrat",
18        "ekonomike",
19        "shkaktuar",
20        "shba",
21        "fondacioni",
22        "gates",
23        "prodhimin",
24        "antivirus"
25    ]
}
```

The status bar at the bottom indicates `Status: 200 OK Time: 239 ms Size: 1.89 KB`.

localhost:5000/overall_donations_country: Finding total number of donations per country.

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'New', 'Import', 'Runner', and a dropdown menu. Below it are sections for 'History' (selected), 'Collections', and 'APIs'. A 'Save Responses' toggle and a 'Clear all' button are also present. The main area has tabs for 'Launchpad' and 'Untitled Request'. Under 'Untitled Request', a GET request is defined with the URL `http://127.0.0.1:5000/overall_donations_country`. The 'Params' tab is selected, showing a table with one row: 'Key' (empty) and 'Value' (empty). The 'Body' tab is selected, showing a JSON response with 25 numbered lines. The response is a list of countries with their donation counts:

```
1 [ {  
2   "_id": "Afghanistan",  
3   "count_per_country": 1  
4 },  
5 {  
6   "_id": "Algeria",  
7   "count_per_country": 1  
8 },  
9 {  
10  "_id": "Antarctica",  
11  "count_per_country": 1  
12 },  
13 {  
14  "_id": "Argentina",  
15  "count_per_country": 3  
16 },  
17 {  
18  "_id": "Australia",  
19  "count_per_country": 7  
20 },  
21 {  
22  "_id": "Austria",  
23  "count_per_country": 1  
24 }]
```

The status bar at the bottom indicates a 200 OK response with a time of 265 ms and a size of 4.09 KB. Navigation icons for back, forward, search, and help are at the bottom left, and a 'Bootcamp' link is at the bottom right.

localhost:5000/overall_donations_country/<country_name>: Finding total number of donations per country for a particular searched country.

The screenshot shows the Postman application interface. The top navigation bar includes 'New', 'Import', 'Runner', and 'Sign In' buttons. The main workspace is titled 'My Workspace' with an 'Invite' button. A 'Launchpad' tab is selected, showing a recent request: 'GET http://127.0.0.1:5000/api/query2'. Below it, an 'Untitled Request' is shown with the URL 'http://127.0.0.1:5000/overall_donations_country/india'. The 'Params' tab is active, displaying a single query parameter 'Key' with value 'Value'. The 'Body' tab is also visible. The status bar at the bottom indicates 'Status: 200 OK' and 'Time: 251 ms'. On the left, a sidebar lists various API requests under 'History' and 'Today'.

Untitled Request

GET http://127.0.0.1:5000/overall_donations_country/india

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (4) Test Results Status: 200 OK Time: 251 ms Size: 208 B Save Response

Pretty Raw Preview Visualize JSON

```
1 [  
2 {  
3   "_id": "India",  
4   "count_per_country": 594  
5 }  
6 ]
```

History Collections APIs

Save Responses Clear all

Today

- GET http://127.0.0.1:5000/overall_donations_country/india
- GET http://127.0.0.1:5000/overall_donations_country
- GET http://127.0.0.1:5000/top_100words_country/albania
- GET http://127.0.0.1:5000/top_100words_country/india
- GET http://127.0.0.1:5000/top_100words_country/india
- GET http://127.0.0.1:5000/top_100words_country/india
- GET http://127.0.0.1:5000/top_100words_country
- GET http://127.0.0.1:5000/overall_top_100words_country
- GET http://127.0.0.1:5000/overall_top_100words
- GET http://127.0.0.1:5000/tweets_per_country/date/country/pakistan/Apr_18_2020
- GET http://127.0.0.1:5000/tweets_per_country/date/country/russia/Apr_18_2020
- GET http://127.0.0.1:5000/tweets_per_country/date/country/sri_lanka/Apr_18_2020
- GET http://127.0.0.1:5000/tweets_per_country/date/Apr_18_2020
- GET http://127.0.0.1:5000/tweets_per_country/united_kingdom

Bootcamp

TEST CASES

- Twitter producer
 - Authentication of credentials.
- Spark Streaming
 - Testing the business logic involving formatting of tweets.
- Flask API
 - Verification of API routes.

FURTHER IMPROVEMENT

1. Communication Security: Network Protocols like Kerberos can be added to Kafka to provide strong authentication for client/server
2. Security of REST API: Configuration of APIs with SSL can help to keep sensitive information sent across the internet encrypted when the server is moved away from localhost.
3. Integration testing : Advanced mocking techniques like Mockito can be used to mock the client connection.

THANK YOU !

APPENDIX

SLIDE 4: FLOW DIAGRAM

1. TWITTER CLIENT GETTING THE INFORMATION (TWEETS)
2. FETCHING TWITTER DATA
3. DISTRIBUTED SYNCHRONISATION SOFTWARE RUNNING IN THE BACKGROUND (ZOOKEEPER IN THIS CASE)
4. WRITING DATA TO A DISTRIBUTED STREAMING PLATFORM (KAFKA IN THIS CASE)
5. STREAM AND TRANSFORMATIONS (SPARK SQL IN THIS CASE)
6. STORING DATA IN THE DATABASE (MONGODB IN THIS CASE)
7. TO ACCESS THE DATABASE AND GET THE DESIRED OUTPUT USING API