**Harvard University Extension School**

**"Principles of Big Data Processing"**

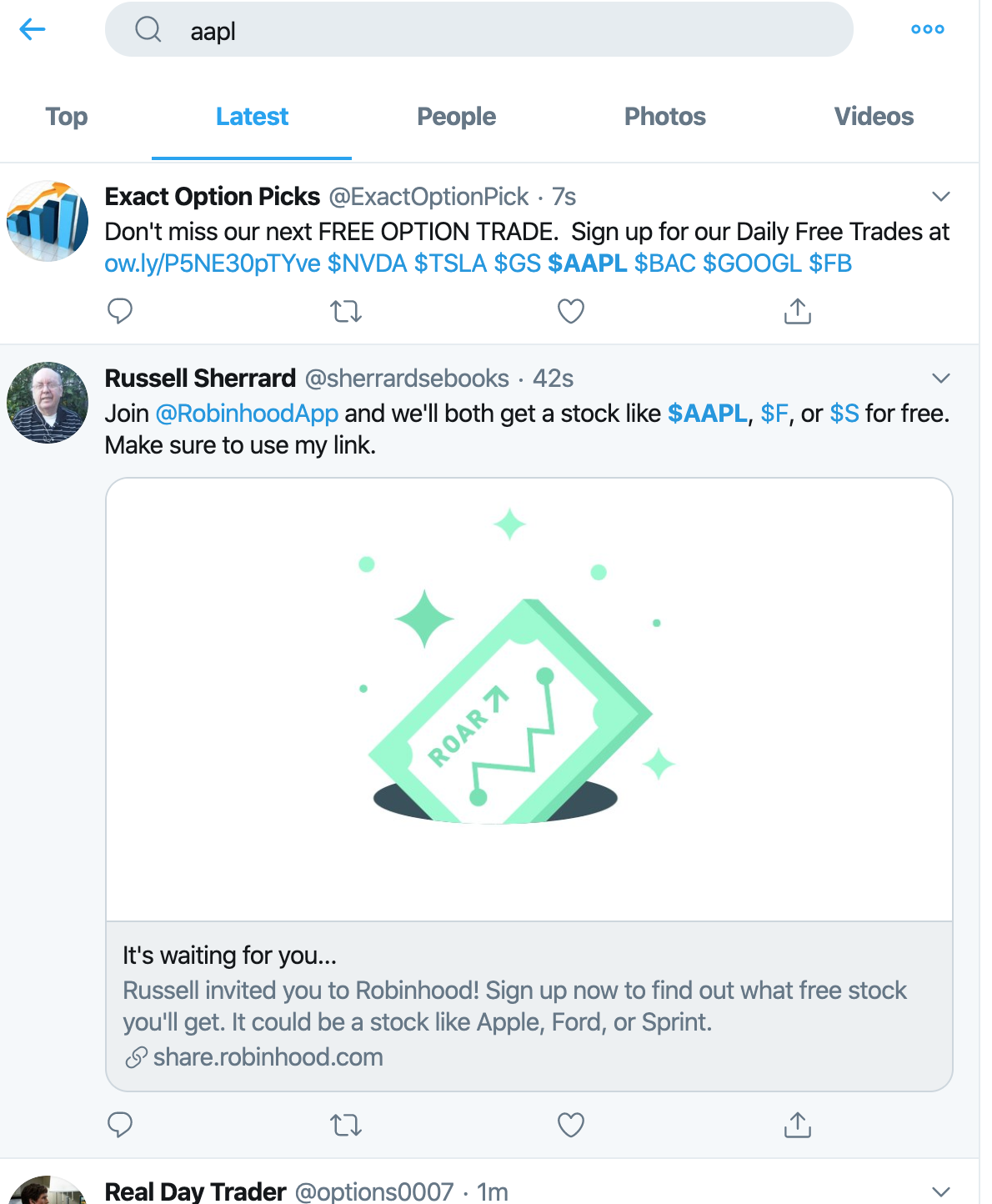
**CSCI E-88, Fall 2019**

**Final Project**

**by Loi Cheng**

## Project Goal and Problem Statement

This project’s goal is to study real-time tweets of the 30 stock symbols in the Dow Jones Industrial Average. We will demonstrate how to build a system that collects only twitter data that mentions these stock symbols and indexes them into ElasticSearch for further analytics. For example, below are some mentions for AAPL, which is Apple Inc.



The 30 Dow stocks are listed below:

|  |  |  |  |
| --- | --- | --- | --- |
| **Company** | **Exchange** | **Symbol** | **Industry** |
| [3M](https://en.wikipedia.org/wiki/3M) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: MMM | [Conglomerate](https://en.wikipedia.org/wiki/Conglomerate_(company)) |
| [American Express](https://en.wikipedia.org/wiki/American_Express) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: AXP | [Financial services](https://en.wikipedia.org/wiki/Financial_services) |
| [Apple Inc.](https://en.wikipedia.org/wiki/Apple_Inc.) | [NASDAQ](https://en.wikipedia.org/wiki/NASDAQ) | [AAPL](http://www.nasdaq.com/symbol/aapl) | [Information technology](https://en.wikipedia.org/wiki/Information_technology) |
| [Boeing](https://en.wikipedia.org/wiki/Boeing) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: BA | Aerospace manufacturer and Arms industry |
| [Caterpillar Inc.](https://en.wikipedia.org/wiki/Caterpillar_Inc.) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: CAT | Construction and Mining |
| [Chevron Corporation](https://en.wikipedia.org/wiki/Chevron_Corporation) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: CVX | [Petroleum industry](https://en.wikipedia.org/wiki/Petroleum_industry) |
| [Cisco Systems](https://en.wikipedia.org/wiki/Cisco_Systems) | [NASDAQ](https://en.wikipedia.org/wiki/NASDAQ) | [CSCO](http://www.nasdaq.com/symbol/csco) | [Information technology](https://en.wikipedia.org/wiki/Information_technology) |
| [Dow Inc.](https://en.wikipedia.org/wiki/Dow_Inc.) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: DOW | [Chemical industry](https://en.wikipedia.org/wiki/Chemical_industry) |
| [ExxonMobil](https://en.wikipedia.org/wiki/ExxonMobil) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: XOM | [Petroleum industry](https://en.wikipedia.org/wiki/Petroleum_industry) |
| [Goldman Sachs](https://en.wikipedia.org/wiki/Goldman_Sachs) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: GS | [Financial services](https://en.wikipedia.org/wiki/Financial_services) |
| [IBM](https://en.wikipedia.org/wiki/IBM) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: IBM | [Information technology](https://en.wikipedia.org/wiki/Information_technology) |
| [Intel](https://en.wikipedia.org/wiki/Intel) | [NASDAQ](https://en.wikipedia.org/wiki/NASDAQ) | [INTC](http://www.nasdaq.com/symbol/intc) | [Information technology](https://en.wikipedia.org/wiki/Information_technology) |
| [Johnson & Johnson](https://en.wikipedia.org/wiki/Johnson_%26_Johnson) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: JNJ | [Pharmaceutical industry](https://en.wikipedia.org/wiki/Pharmaceutical_industry) |
| [JPMorgan Chase](https://en.wikipedia.org/wiki/JPMorgan_Chase) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: JPM | [Financial services](https://en.wikipedia.org/wiki/Financial_services) |
| [McDonald's](https://en.wikipedia.org/wiki/McDonald%27s) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: MCD | [Food industry](https://en.wikipedia.org/wiki/Food_industry) |
| [Merck & Co.](https://en.wikipedia.org/wiki/Merck_%26_Co.) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: MRK | [Pharmaceutical industry](https://en.wikipedia.org/wiki/Pharmaceutical_industry) |
| [Microsoft](https://en.wikipedia.org/wiki/Microsoft) | [NASDAQ](https://en.wikipedia.org/wiki/NASDAQ) | [MSFT](http://www.nasdaq.com/symbol/msft) | [Information technology](https://en.wikipedia.org/wiki/Information_technology) |
| [Nike](https://en.wikipedia.org/wiki/Nike,_Inc.) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: NKE | [Apparel](https://en.wikipedia.org/wiki/Apparel) |
| [Pfizer](https://en.wikipedia.org/wiki/Pfizer) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: PFE | [Pharmaceutical industry](https://en.wikipedia.org/wiki/Pharmaceutical_industry) |
| [Procter & Gamble](https://en.wikipedia.org/wiki/Procter_%26_Gamble) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: PG | [Fast moving consumer goods](https://en.wikipedia.org/wiki/Fast_moving_consumer_goods) |
| [The Coca-Cola Company](https://en.wikipedia.org/wiki/The_Coca-Cola_Company) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: KO | [Food industry](https://en.wikipedia.org/wiki/Food_industry) |
| [The Home Depot](https://en.wikipedia.org/wiki/The_Home_Depot) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: HD | [Retailing](https://en.wikipedia.org/wiki/Retailing) |
| [The Travelers Companies](https://en.wikipedia.org/wiki/The_Travelers_Companies) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: TRV | [Financial services](https://en.wikipedia.org/wiki/Financial_services) |
| [The Walt Disney Company](https://en.wikipedia.org/wiki/The_Walt_Disney_Company) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: DIS | Broadcasting and entertainment |
| [United Technologies](https://en.wikipedia.org/wiki/United_Technologies) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: UTX | [Conglomerate](https://en.wikipedia.org/wiki/Conglomerate_(company)) |
| [UnitedHealth Group](https://en.wikipedia.org/wiki/UnitedHealth_Group) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: UNH | [Managed health care](https://en.wikipedia.org/wiki/Managed_health_care) |
| [Verizon](https://en.wikipedia.org/wiki/Verizon_Communications) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: VZ | [Telecommunication](https://en.wikipedia.org/wiki/Telecommunication) |
| [Visa Inc.](https://en.wikipedia.org/wiki/Visa_Inc.) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: V | [Financial services](https://en.wikipedia.org/wiki/Financial_services) |
| [Walgreens Boots Alliance](https://en.wikipedia.org/wiki/Walgreens_Boots_Alliance) | [NASDAQ](https://en.wikipedia.org/wiki/NASDAQ) | [WBA](http://www.nasdaq.com/symbol/wba) | [Retailing](https://en.wikipedia.org/wiki/Retailing) |
| [Walmart](https://en.wikipedia.org/wiki/Walmart) | [NYSE](https://en.wikipedia.org/wiki/New_York_Stock_Exchange) | NYSE: WMT | [Retailing](https://en.wikipedia.org/wiki/Retailing) |

## YouTube Video URL

<https://youtu.be/f0dU5-QMWaQ>

## Big Data Source

Twitter streaming data

## Expected results

As a result of this processing pipeline, I expect to be able find which stocks have higher mentions, and what effect does the time of day on the results.

## Processing Pipeline

Flume

Kafka

Logstash

ES



twitter

## Pipeline Overview and Technologies used

* Collection tier: Flume with experimental Twitter streaming source
  + Flume will be ingesting data from Twitter, filtered by stock symbol tags
  + The docker image “openjdk” is used to run flume
* Messaging Tier: Kafka
  + Flume will push events into Kafka for further processing
  + The docker images “zookeeper:3.4.9” and “wurstmeister/kafka:2.12-2.3.0” are used to run Kafka
* Stream Processing Tier: Logstash
  + Connector will filter and index data into indexes
  + The docker image “elk” (ElasticSearch, Logstash, Kibana) is used
* Visualization Tier: we will use Kibana with ElasticSearch to visualize received data and discover which stocks are popular and in which area
  + The same docker image “elk” (ElasticSearch, Logstash, Kibana) is used

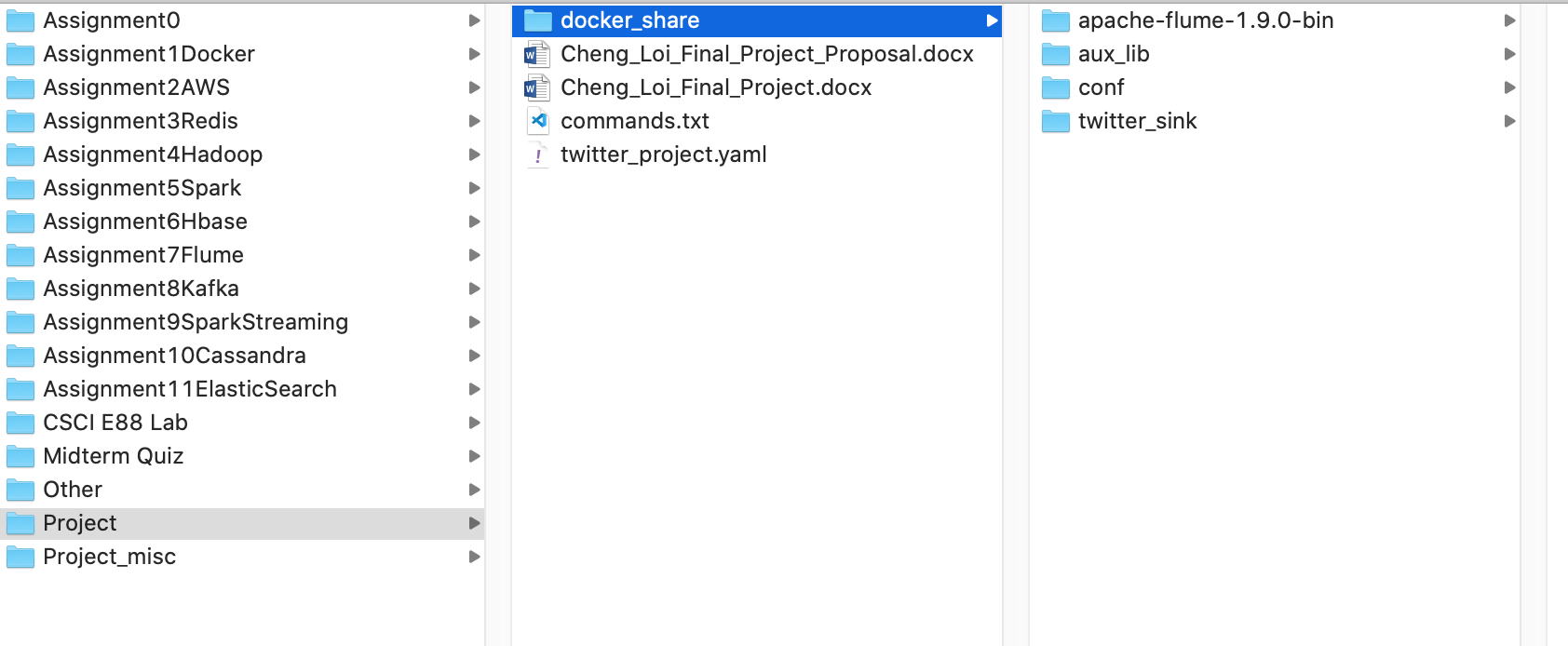
All the docker images used are available online at the docker hub



## Implementation

*This is the main section. It should contain all implementation details for each tier. Copy all relevant configuration and code. Code should be well documented.*

All the required files are placed inside the “Project” folder. Here, a “docker\_share” folder contains files and configurations that some of docker containers will access:



A set of docker containers are used to implement the system, the docker-compose yaml configuration is shown below:

version: '3.5'

networks:

twitter-demo:

name: twitter-demo-net

driver: bridge

services:

zookeeper:

image: zookeeper:3.4.9

ports:

- "2181:2181"

networks:

- twitter-demo

kafka:

image: wurstmeister/kafka:2.12-2.3.0

environment:

# KAFKA\_BROKER\_ID: 1

# (Hack for Mac)use this if you want to have docker host node to be used as broadcast ip

HOSTNAME\_COMMAND: "/sbin/ip route|awk '/src/ { print $$NF }'"

# Use below for Linux

# HOSTNAME\_COMMAND: "ip route get 1.2.3.4 | awk '{print $$7}'"

KAFKA\_ADVERTISED\_PORT: 9092

KAFKA\_ZOOKEEPER\_CONNECT: zookeeper:2181

# KAFKA\_CREATE\_TOPICS: "varnish\_raw\_logs:10:1"

depends\_on:

- zookeeper

ports:

- 9092

networks:

- twitter-demo

#jdk container is used to run flume

jdk:

image: openjdk

depends\_on:

- kafka

- zookeeper

volumes:

- "./docker\_share:/docker\_share"

command: 'tail -F anything'

networks:

- twitter-demo

#elk container is used to run logstash, elasticsearch, and kibana

elk:

image: sebp/elk

volumes:

- "./docker\_share:/docker\_share"

depends\_on:

- kafka

- zookeeper

ports:

- "5601:5601"

- "9200:9200"

- "5044:5044"

networks:

- twitter-demo

Below is the flume config file:

# Licensed to the Apache Software Foundation (ASF) under one

# or more contributor license agreements. See the NOTICE file

# distributed with this work for additional information

# regarding copyright ownership. The ASF licenses this file

# to you under the Apache License, Version 2.0 (the

# "License"); you may not use this file except in compliance

# with the License. You may obtain a copy of the License at

#

# http://www.apache.org/licenses/LICENSE-2.0

#

# Unless required by applicable law or agreed to in writing,

# software distributed under the License is distributed on an

# "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY

# KIND, either express or implied. See the License for the

# specific language governing permissions and limitations

# under the License.

# The configuration file needs to define the sources,

# the channels and the sinks.

# Sources, channels and sinks are defined per agent,

# in this case called 'TwitterAgent'

TwitterAgent.sources = Twitter

TwitterAgent.channels = MemChannel

TwitterAgent.sinks = kafkasink #filesink

TwitterAgent.sources.Twitter.type = com.cloudera.flume.source.TwitterSource

#TwitterAgent.sources.Twitter.type = org.apache.flume.source.twitter.TwitterSource

TwitterAgent.sources.Twitter.channels = MemChannel

TwitterAgent.sources.Twitter.consumerKey = sW1adEu8y6y0Oel88CJJRqQfx

TwitterAgent.sources.Twitter.consumerSecret = ahYFxOmhPR83oG3db09drJe3JpvG0GBoWZP9ocPNXdumG6zqRM

TwitterAgent.sources.Twitter.accessToken = 2492115098-Se1OHfCX0J1VXetNW0IgAkrKSj1Zzu44srhTfQ4

TwitterAgent.sources.Twitter.accessTokenSecret = iD8nL3plF80cZQbhVGqxYZbN5wnNEwOtnLcF2syCkhzUK

TwitterAgent.sources.Twitter.keywords = $MMM, $AXP, $AAPL, $BA, $CAT, $CVX, $CSCO, $KO, $DOW, $XOM, $GS, $HD, $IBM, $INTC, $JNJ, $JPM, $MCD, $MRK, $MSFT, $NKE, $PFE, $PG, $TRV, $UNH, $UTX, $VZ, $V, $WMT, $WBA, $DIS

#filesink

# TwitterAgent.sinks.filesink.type = file\_roll

# TwitterAgent.sinks.filesink.channel = MemChannel

# TwitterAgent.sinks.filesink.sink.directory = /docker\_share/twitter\_sink

# TwitterAgent.sinks.filesink.sink.pathManager.extension = out

# TwitterAgent.sinks.filesink.sink.pathManager.prefix = project

# TwitterAgent.sinks.filesink.sink.rollInterval = 3600

#kafkasink

TwitterAgent.sinks.kafkasink.type = org.apache.flume.sink.kafka.KafkaSink

TwitterAgent.sinks.kafkasink.topic = project

TwitterAgent.sinks.kafkasink.brokerList = project\_kafka\_1:9092

TwitterAgent.sinks.kafkasink.channel = MemChannel

TwitterAgent.sinks.kafkasink.batchSize = 1

# add header and text to sink

# TwitterAgent.sinks.kafkasink.sink.serializer = header\_and\_text

# TwitterAgent.sinks.kafkasink.sink.serializer.appendNewline = true

TwitterAgent.channels.MemChannel.type = memory

TwitterAgent.channels.MemChannel.capacity = 10000

TwitterAgent.channels.MemChannel.transactionCapacity = 100

Below is the Logstash config file

input {

kafka {

bootstrap\_servers => "project\_kafka\_1:9092"

topics => ["project"]

codec => json

}

}

filter{

date {

match => ["timestamp\_ms", "UNIX\_MS"]

target => "@timestamp"

}

mutate {

convert => { "id" => "string" }

}

if [extended\_tweet][full\_text]{

mutate {

copy => { "[extended\_tweet][full\_text]" => "text" }

}

}

prune {

whitelist\_names => [ "^text$", "^id$", "^@timestamp$" ]

}

}

output {

elasticsearch {

hosts => ["localhost:9200"]

index => "project"

workers => 1

}

}

Below are the commands used to start the system. The commands are written based on the “Project” folder structure as previously shown. These commands are run from the terminal

# 1. start docker containers

docker-compose -f twitter\_project.yaml up -d --scale kafka=3

# 2. create topic using kafka container

docker exec -i project\_kafka\_1 kafka-topics.sh \

--create \

--zookeeper zookeeper:2181 \

--replication-factor 2 \

--partitions 2 \

--topic project

# 3. start flume in jdk container to collect twitter data

docker exec -it project\_jdk\_1 bash

docker\_share/apache-flume-1.9.0-bin/bin/flume-ng agent \

-c /docker\_share/conf \

-f /docker\_share/conf/flume\_project.conf \

-n TwitterAgent \

-C /docker\_share/aux\_lib \

-Dflume.root.logger=DEBUG,console

# 4. (optional) start kafka consumer to view messages

docker exec -i project\_kafka\_2 kafka-console-consumer.sh \

--bootstrap-server project\_kafka\_1:9092 \

--topic project

# 5. run elk container, use logstash to consume kafka messages and send to elasticsearch

docker exec -it project\_elk\_1 bash

service logstash stop

/opt/logstash/bin/logstash -f /docker\_share/conf/logstash-kafka.conf

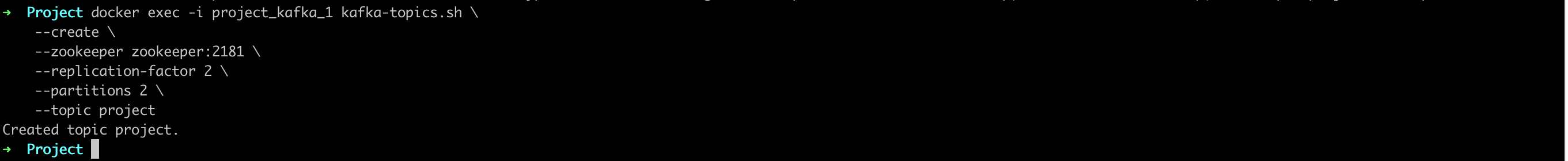
## Results

*Demonstrate your results. Provide all relevant screenshots of the log files, data stored in DB and GUI tier if applicable.*

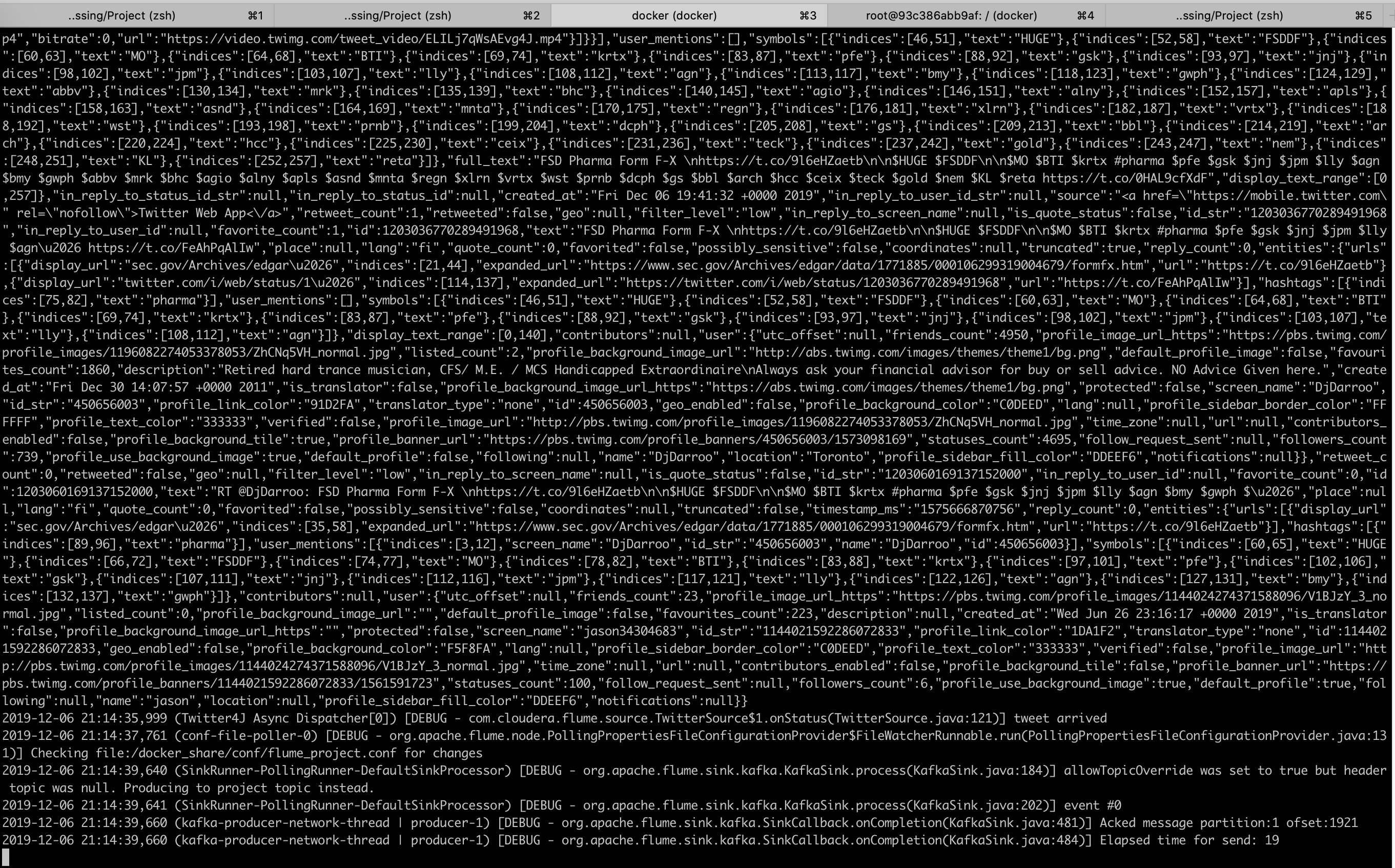
All the docker images should be up and running as shown:

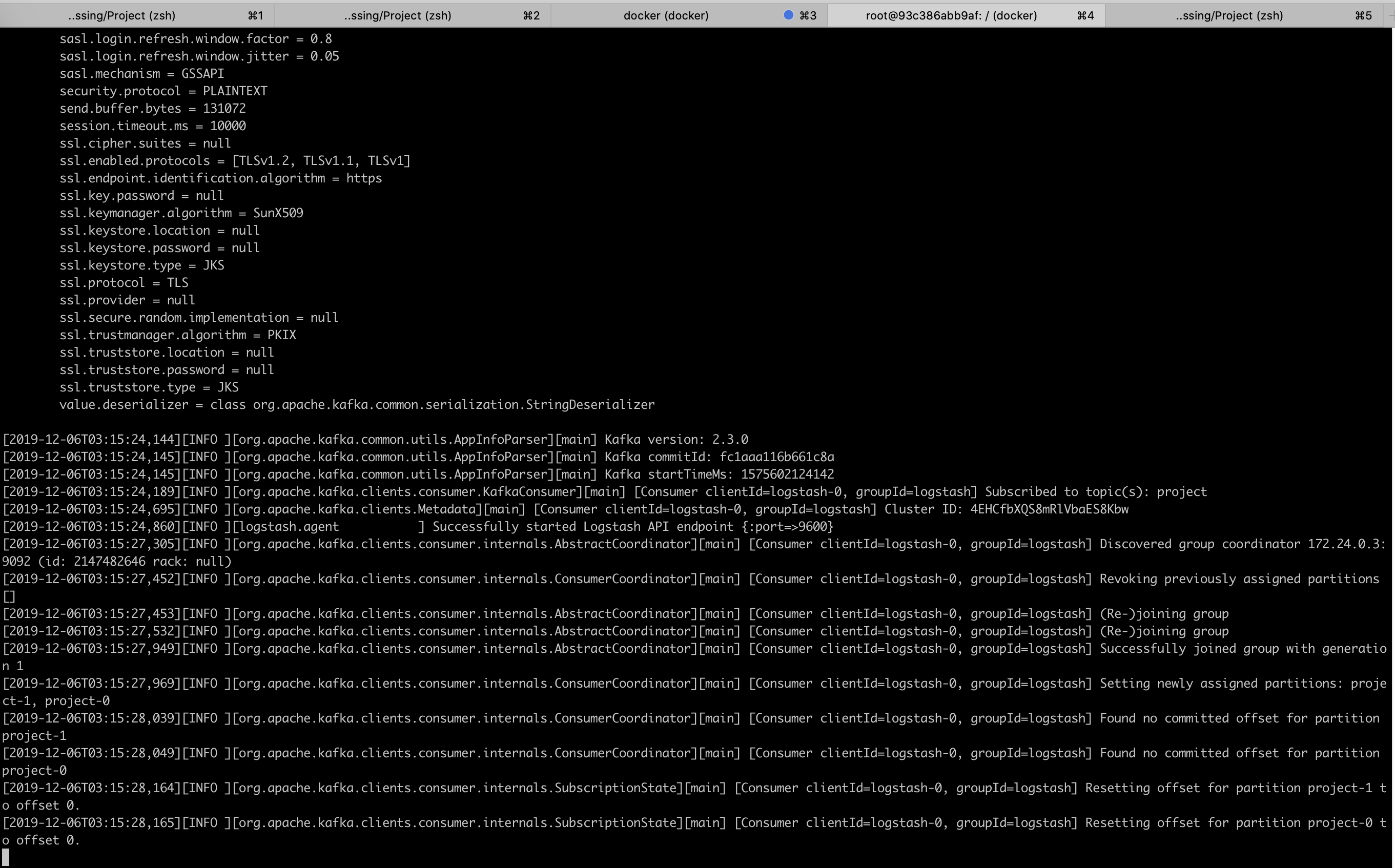


The Kafka topic is created:

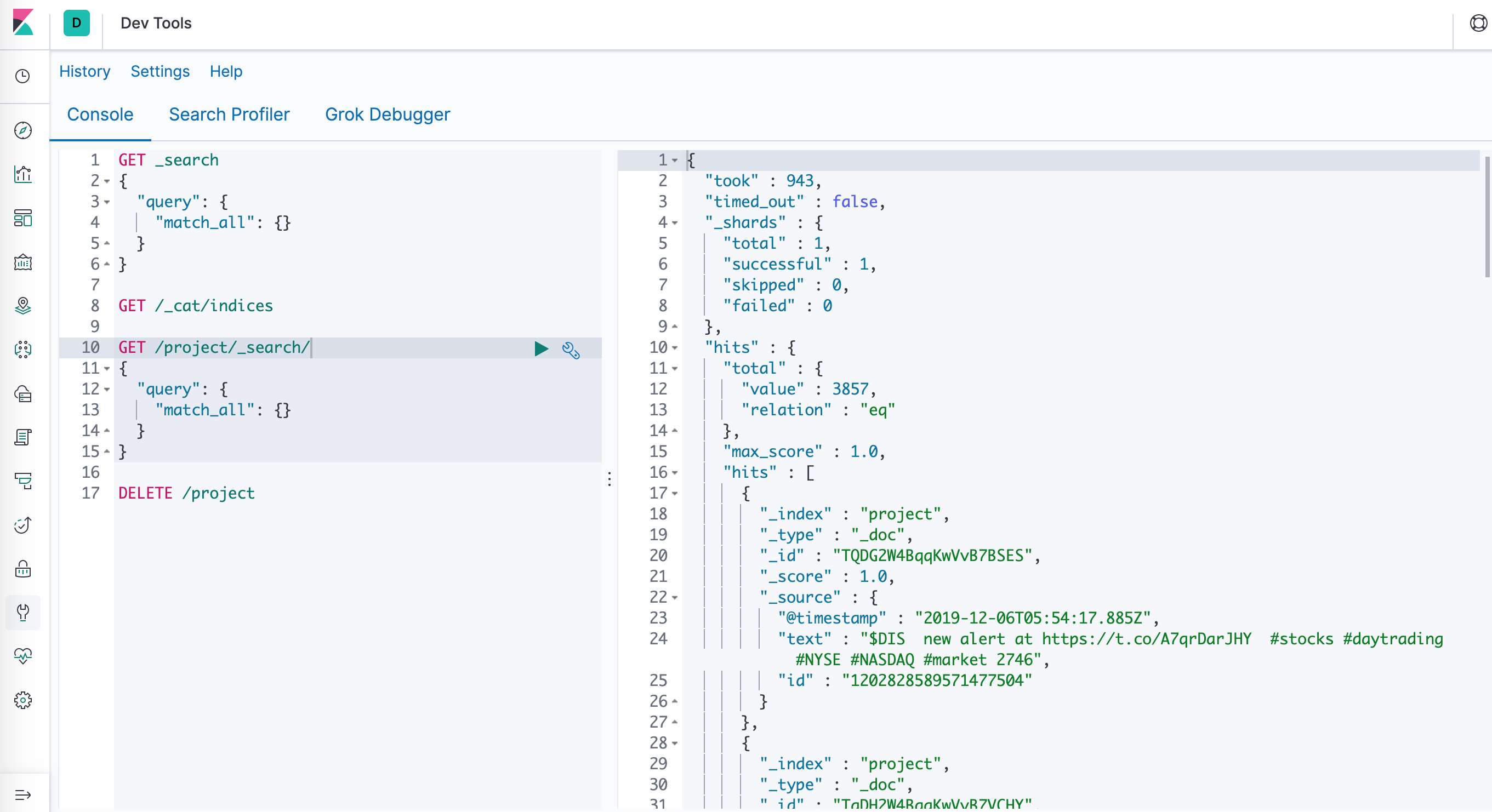


Flume is up and running, and sending messages to Kafka:

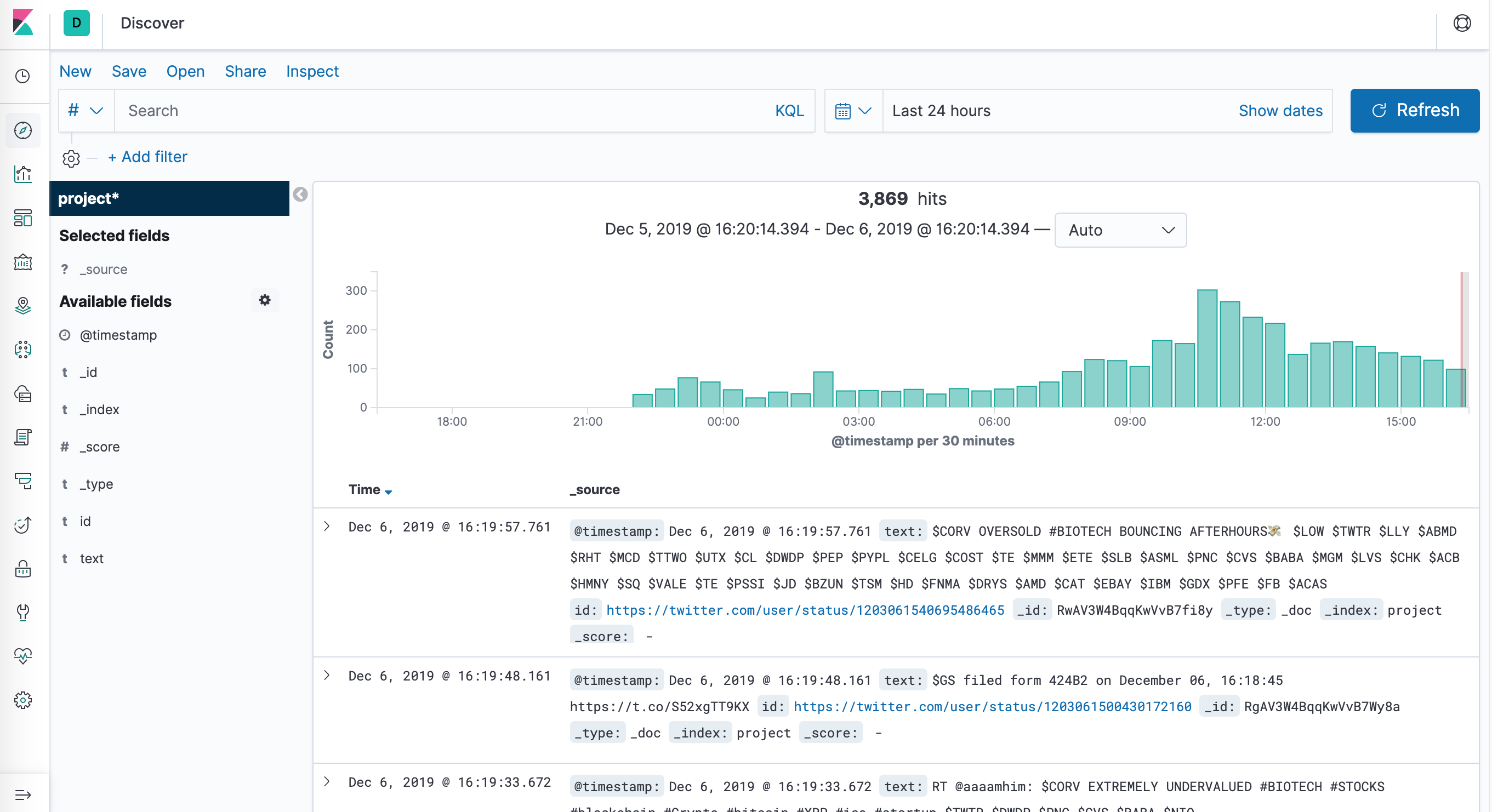


Logstash is up and running, and consuming messages from Kafka and sending data to ElasticSearch

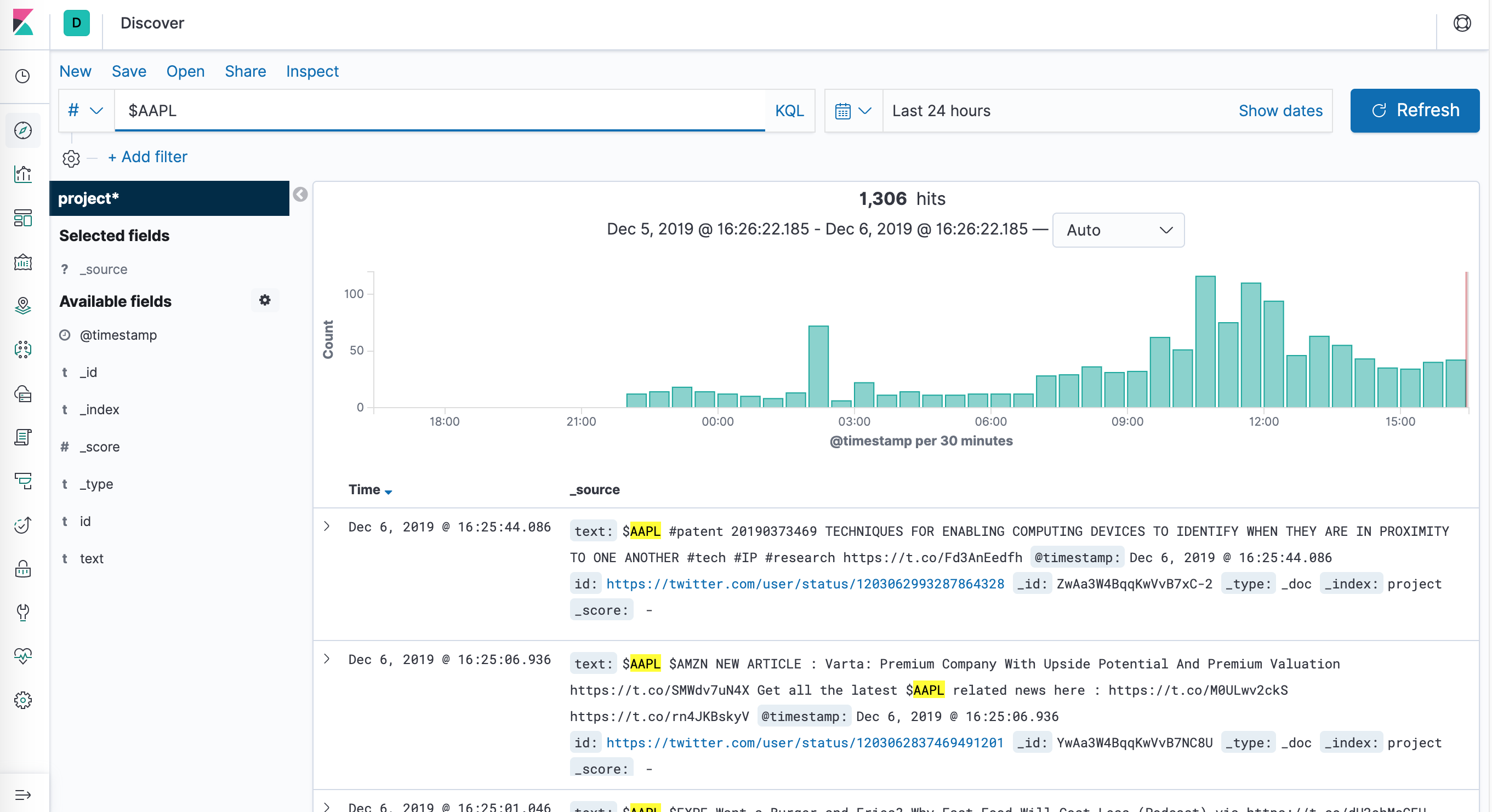
We can verify the data is getting into ElasticSearch from Kibana dev tools:



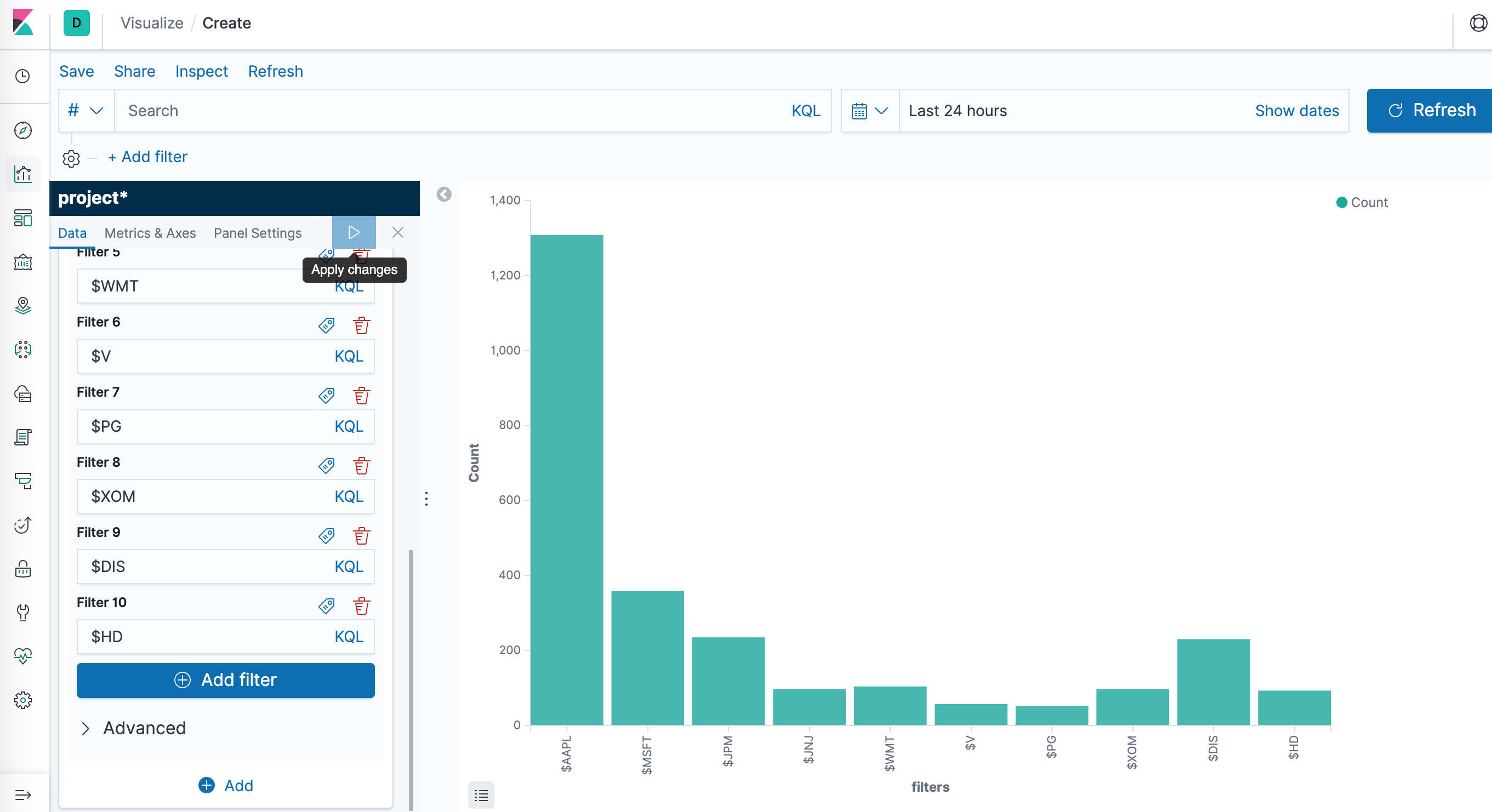
We can visualize the data in Kibana. The counts over time of the stock symbols, along with the most recent tweets, are shown below:



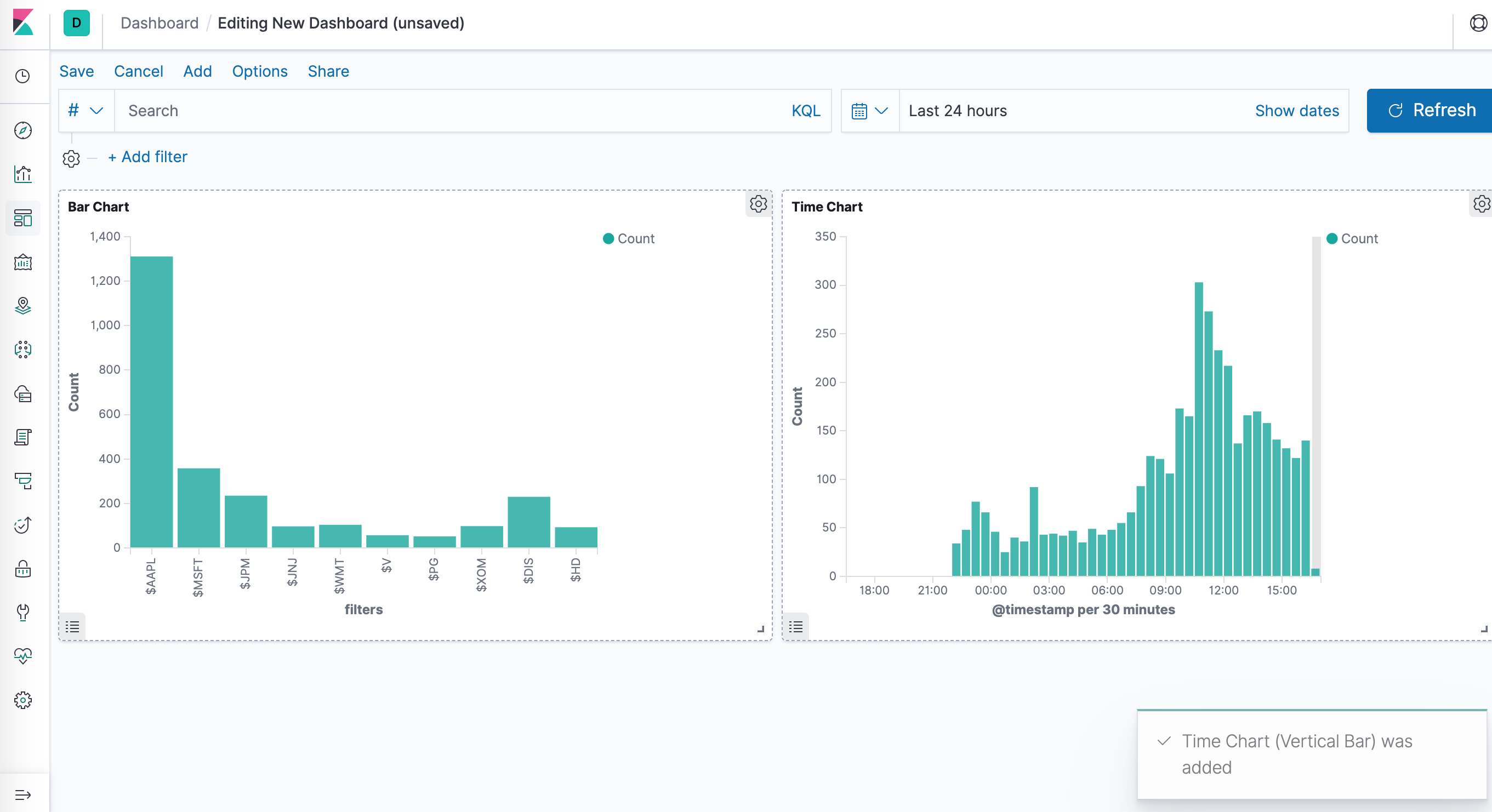
We can also filter the data for a specific company, like Apple ($AAPL), as shown below:



Below is a barchart showing a comparison of the mentions by symbol for the 10 largest companies in the Dow Index. Apple ($AAPL) has the most mentions.



We can also make a dashboard with multiple visualizations, as shown below:



## Conclusions and Lesson Learned

Describe what you have learned during this project:

*What issues did you have?*

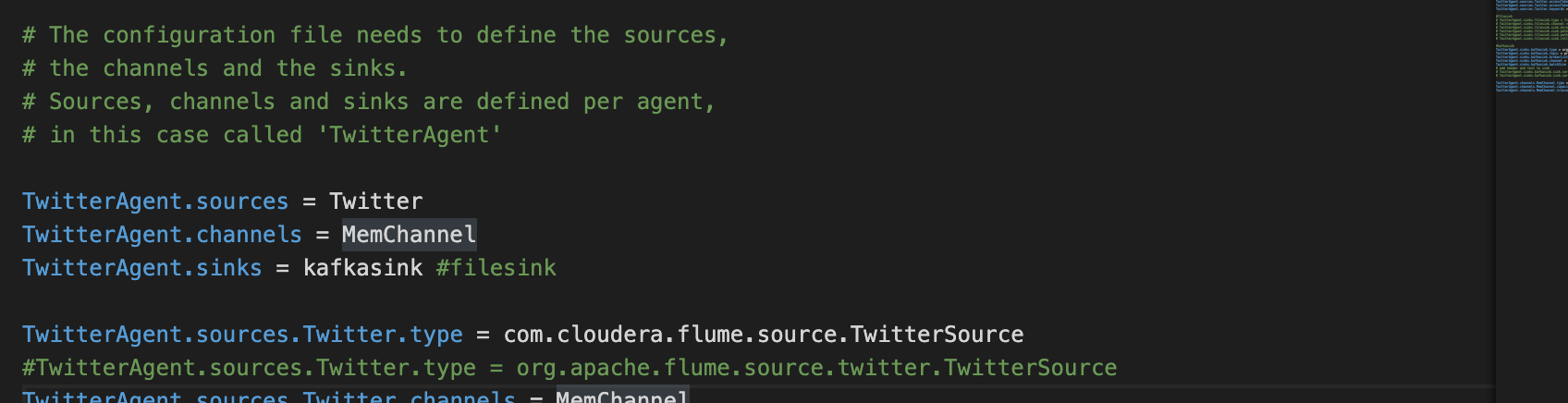
**Issue 1**

In Flume, the built-in twitter source type cannot filter tweets by keyword. To use the Cloudera twitter source type, the source code was downloaded from GitHub and compiled. In the command to launch Flume, it includes a reference to the folder where our custom files are located

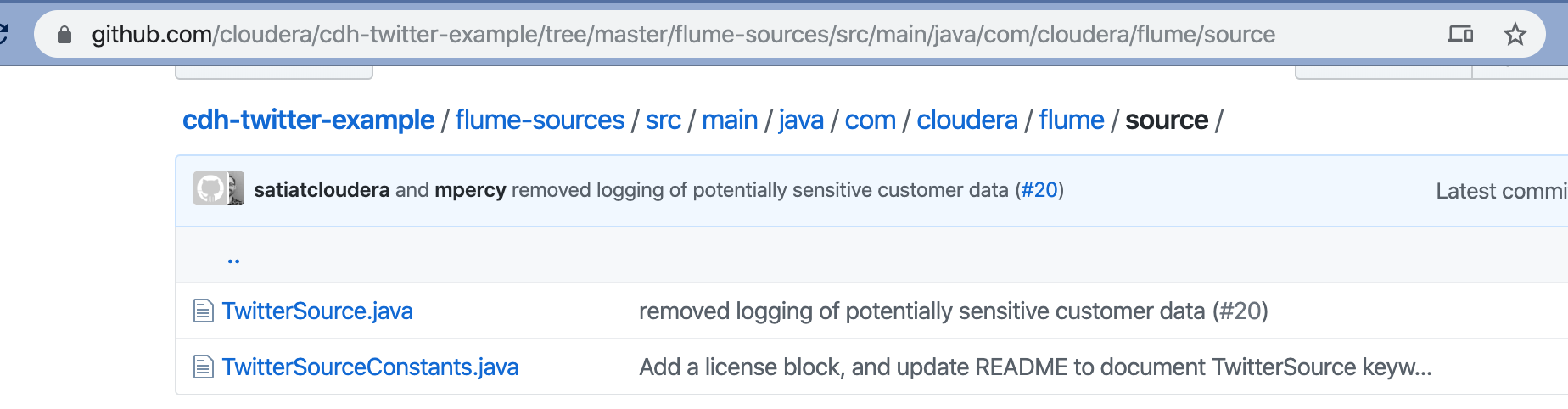
Built in: org.apache.flume.source.twitter.TwitterSource

Cloudera: com.cloudera.flume.source.TwitterSource

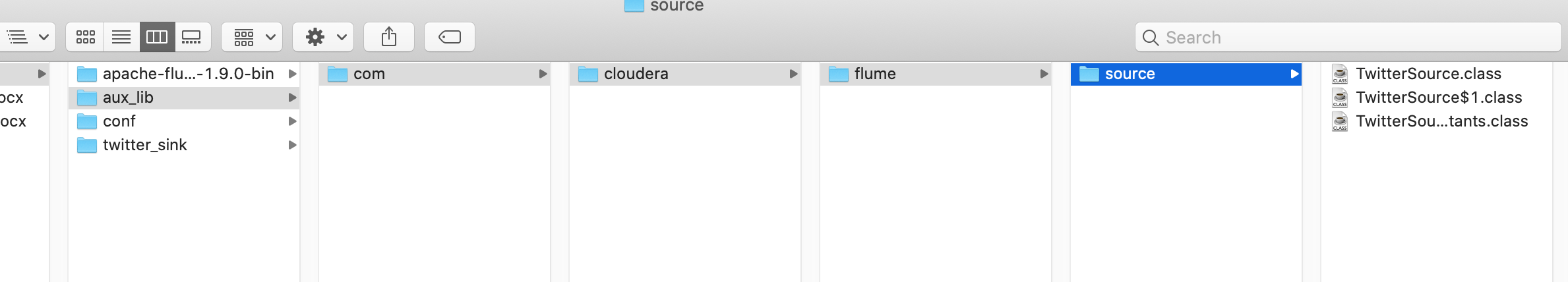
Flume config file showing the Twitter source type used



The java code for the Cloudera Twitter source on GitHub



The “aux\_lib” folder that contains the custom complied code for the twitter source



The command to run Flume, with the -C /docker\_share/aux\_lib command to specify the location of the custom Twitter source



Guide to using custom source for Flume:

<http://www.dataprocessingtips.com/2016/04/24/custom-twitter-source-for-apache-flume/>

**Issue 2**

There was no Docker image on Docker Hub that has flume version 1.9. The only available one I found was v1.8. I tried the v1.8 image, and it worked to a file sink, but did not work with a Kafka sink. So, I downloaded flume v1.9 and used an image with Java (openjdk) to run Flume.

*What limitations, if any, did you run into with the technologies used?*

I tried using Flume with both file sink and Kafka sink, but the two sinks were storing different Twitter data. So, I only used Flume with a Kafka sink.

*What would you do differently next time?*

It seems like Logstash can get data directly from twitter with the plugin below: <https://www.elastic.co/guide/en/logstash/current/plugins-inputs-twitter.html>

I may try using this plug-in instead of Flume to collect Twitter data.

*What would you like to improve if you had more time?*

I would add another Kafka consumer to consume the data and store in HDFS or another kind of long-term storage.

*What alternatives to technologies you used you might consider?*

Instead of using locally run docker images, I might consider running the images on AWS Fargate

*Where would you take your project next?*

I would build a custom webpage to visualize the data and make it publicly available.