

# **CUSTOM YOUTUBE PLUGIN**

## **Project Objective**

The goal of this project is to automatically collect and analyze key statistics from YouTube channels. Using the YouTube Data API v3, our Telegraf plugin gathers the following data for specified channels:

- Subscriber Count
- Video Count
- Total View Count
- Channel Country
- Channel Name
- Channel Creation Date

60 > youtube.go > ...

```
1 package youtube
2
3 import (
4     "context"
5     "fmt"
6     "strings"
7     "time"
8
9     "github.com/influxdata/telegraf"
10    "github.com/influxdata/telegraf/plugins/inputs"
11    "google.golang.org/api/option"
12    "google.golang.org/api/youtube/v3"
13 )
14
15 type Youtube struct {
16     Channels []string `toml:"channels"`
17     APIKey   string  `toml:"api_key"`
18     youtubeService *youtube.Service
19 }
20
21 const YoutubeConfig = `
22     ## List of channels to monitor
23     channels = ["UCLA_DiR1fFKNvjuUpBHmylQ"]
24     ## Google API Key
25     # api_key = ""
26 `
27
28 func (s *Youtube) SampleConfig() string {
29     return YoutubeConfig
30 }
31
32 func (s *Youtube) Description() string {
33     return "Gather youtube channel information from Youtube"
34 }
35
36 func (s *Youtube) createYoutubeService(ctx context.Context) (*youtube.Service, error) {
37     return youtube.NewService(ctx, option.WithAPIKey(s.APIKey))
38 }
39
40 func (s *Youtube) Gather(acc telegraf.Accumulator) error {
41     ctx := context.Background()
42
43     if s.youtubeService == nil {
44         service, err := s.createYoutubeService(ctx)
45         if err != nil {
46             return err
47         }
48         s.youtubeService = service
49     }
50
51     part := strings.Join([]string{"snippet", "statistics"}, ",")
52     call := s.youtubeService.Channels.List([]string(part)).Id(s.Channels...).MaxResults(50)
53
54     resp, err := call.Do()
55     if err != nil {
56         return err
57     }
58
59     now := time.Now()
60
61     for _, item := range resp.Items {
62         tags := getTags(item)
63         fields := getFields(item)
64
65         acc.AddFields("youtube_channel", fields, tags, now)
66     }
67
68     return nil
69 }
70
71 func getTags(channelInfo *youtube.Channel) map[string]string {
72     fmt.Println("channelInfo", channelInfo)
73     return map[string]string{
74         "id": channelInfo.Id,
75         "title": channelInfo.Snippet.Title,
76     }
77 }
78
79 func getFields(channelInfo *youtube.Channel) map[string]interface{} {
80     return map[string]interface{}{
81         "subscribers": channelInfo.Statistics.SubscriberCount,
82         "videos": channelInfo.Statistics.VideoCount,
83         "views": channelInfo.Statistics.ViewCount,
84         "country": channelInfo.Snippet.Country,
85         "channelName": channelInfo.Snippet.Title,
86         "createdon": channelInfo.Snippet.PublishedAt,
87     }
88 }
89
90 func init() {
91     inputs.Add("youtube", func() telegraf.Input {
92         return &Youtube{}
93     })
94 }
95
```

## 2. Edit All.go File

```
zeynep@BTSPC-ER: ~/go/src/github.com/influxdata/telegraf/plugins/inputs/all
GNU nano 6.2 all.go
package all

import (
    "github.com/influxdata/telegraf/plugins/inputs/influxdb"
    "github.com/influxdata/telegraf/plugins/inputs/influxdb_listener"
    "github.com/influxdata/telegraf/plugins/inputs/influxdb_v2_listener"
    "github.com/influxdata/telegraf/plugins/inputs/modbus"
    "github.com/influxdata/telegraf/plugins/inputs/syslog"
    "github.com/influxdata/telegraf/plugins/inputs/youtube"
)

Add!
```

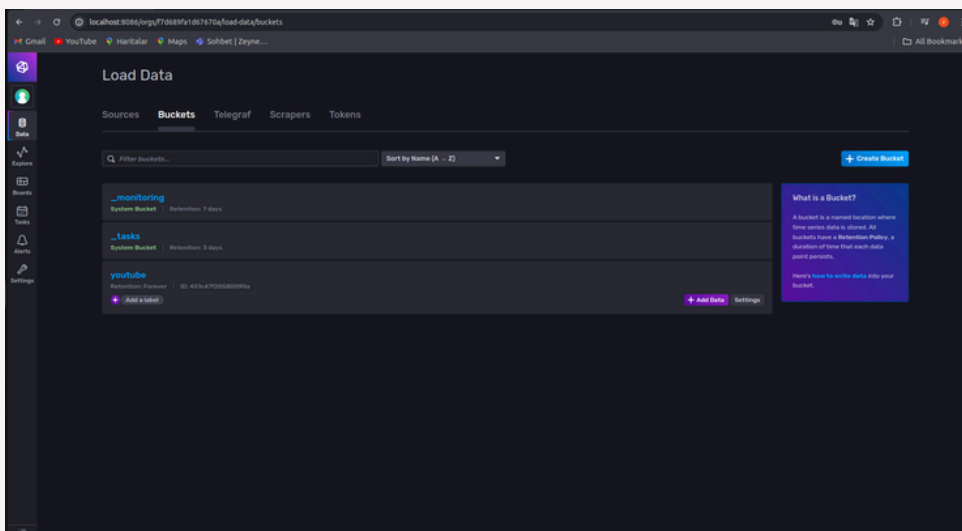
## 3. Create telegraf binary file

zeynep@BTSPC-ER: ~/go/src/telegraf\$ make telegraf

## 4. Start Influxdb

```
zeynep@BTSPC-ER:~$ which influxd
/usr/local/bin/influxd
zeynep@BTSPC-ER:~$ cd /usr/local/bin/influxd
bash: cd: /usr/local/bin/influxd: Not a directory
zeynep@BTSPC-ER:~$ cd /usr/local/bin/
zeynep@BTSPC-ER:~/usr/local/bin$ ./influxd
2024-06-05T14:24:45.881675Z info Welcome to InfluxDB {"log_id": "0pbC
SVEG000", "version": "2.0.7", "commit": "2a45f0c037", "build_date": "2021-06-04T
19:17:40Z"}
2024-06-05T14:24:45.890554Z info Resources opened {"log_id": "0pbC
SVEG000", "service": "bolt", "path": "/home/zeynep/.influxdbv2/influxd.bolt"}
2024-06-05T14:24:45.893023Z info Bringing up metadata migrations {"log_id
": "0pbCSVEG000", "service": "migrations", "migration_count": 15}
2024-06-05T14:24:46.090092Z info Using data dir {"log_id": "0pbCSVEG000"
, "service": "storage-engine", "service": "store", "path": "/home/zeynep/.influx
dbv2/engine/data"}
```

## 5. Create Youtube Bucket



## 6.Run token commands

```
export INFLUX_TOKEN=o-qltcD-  
a8YntwdHZF3Xz3oWX1LWnTzTVIDN5ObV0EqTErbkClhR8ZsUPJDj2XoFYVffa3fDMdXrel55  
XSD0Uw==  
  
telegraf --config http://localhost:8086/api/v2/telegrafs/0d2632a78519b000
```

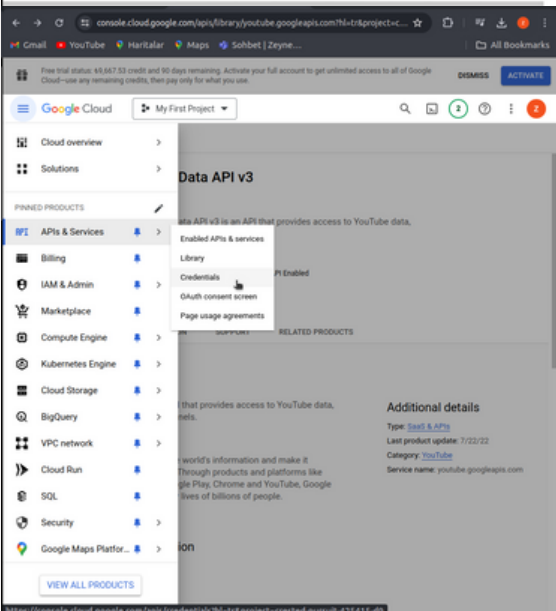
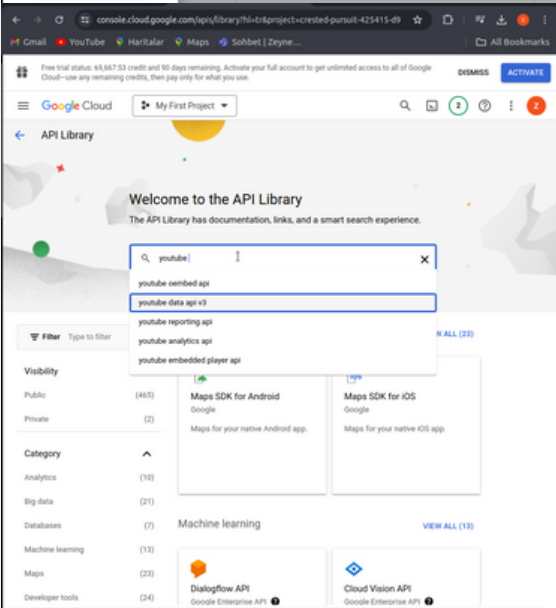
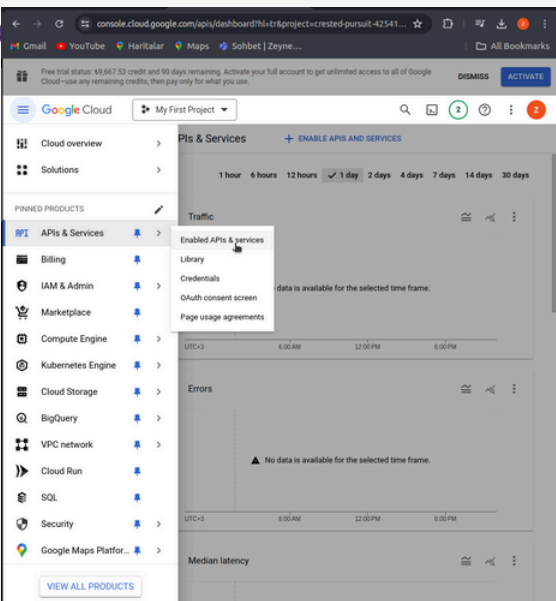
## 7.Create Youtube conf. File

The screenshot shows the InfluxDB Telegraf configuration interface. The 'Load Data' page is visible, showing a list of configurations. The 'youtube' configuration is selected, and the 'Edit Telegraf Configuration' dialog is open. The dialog shows the configuration for the 'youtube' source, which is a plugin for gathering YouTube channel information. The configuration includes fields for 'interval', 'round\_interval', 'metric\_batch\_size', 'metric\_buffer\_limit', 'collection\_jitter', 'flush\_interval', and 'flush\_jitter'. The configuration is as follows:

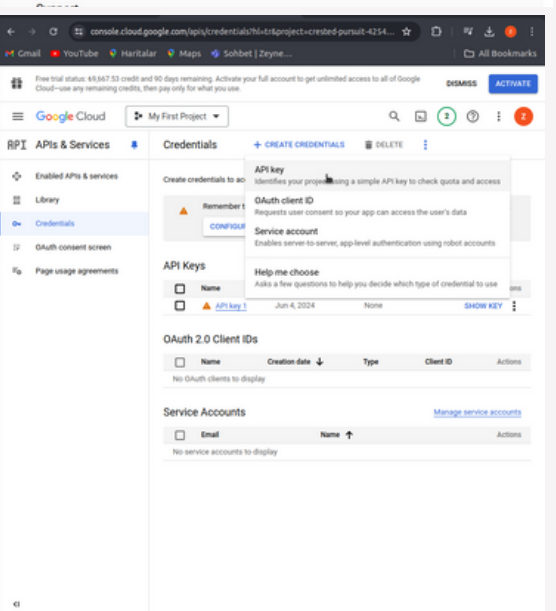
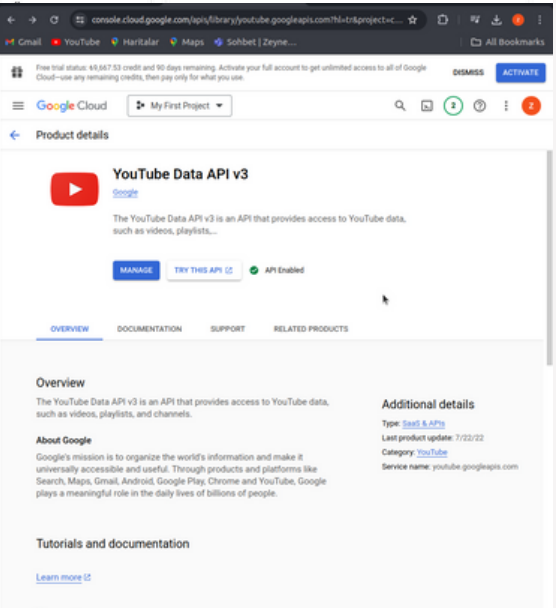
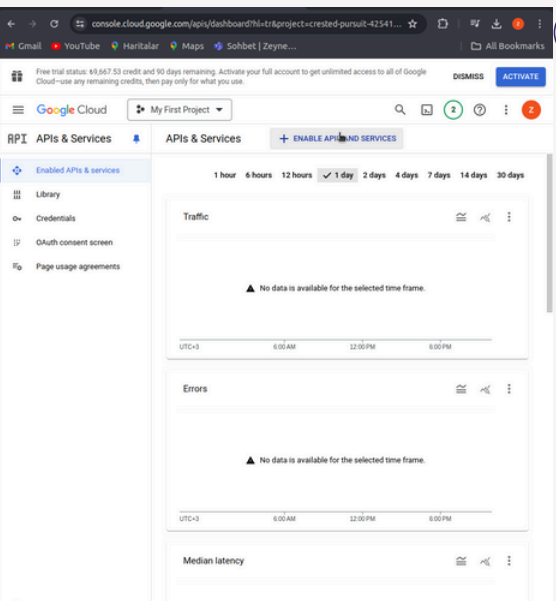
```
1 # Configuration for telegraf agent  
2 [agent]  
3 # Default data collection interval for all inputs  
4 interval = "10s"  
5 # Rounds collection interval to 'interval'  
6 # ie., if interval="10s" then always collect on :00, :10, :20, etc..  
7 round_interval = true  
8  
9 # Telegraf will send metrics to outputs in batches of at most  
10 # metric_batch_size metrics.  
11 # This controls the size of writes that telegraf sends to output plugins.  
12 metric_batch_size = 1000  
13  
14 # For failed writes, telegraf will cache metric_buffer_limit metrics for each  
15 # output, and will flush this buffer on a successful write. Oldest metrics  
16 # are dropped first when this buffer fills.  
17 # This buffer only fills when writes fail to output plugin(s).  
18 metric_buffer_limit = 10000  
19  
20 # Collection jitter is used to jitter the collection by a random amount.  
21 # Each plugin will sleep for a random time within jitter before collecting.  
22 # This can be used to avoid many plugins querying things like sysfs at the  
23 # same time, which can have a measurable effect on the system.  
24 collection_jitter = "0s"  
25  
26 # Default flushing interval for all outputs. Maximum flush_interval will be  
27 # flush_interval * flush_jitter  
28 flush_interval = "10s"  
29  
30 # Jitter the flush interval by a random amount. This is primarily to avoid  
31 # large write bursts for users running a large number of telegraf instances.  
32 # ie., a jitter of 5s and interval 10s means flushes will happen every 10-15s  
33 flush_jitter = "0s"
```

# 8. Google cloud/Youtube Data API v3 Key

1



2



## 9. Edit Telegraf Conf.

```
26 ## Default flushing interval for all outputs. Maximum flush_interval will be
27 ## flush_interval + flush_jitter
28 flush_interval = "10s"
29 ## Jitter the flush interval by a random amount. This is primarily to avoid
30 ## large write spikes for users running a large number of telegraf instances.
31 ## ie, a jitter of 5s and interval 10s means flushes will happen every 10-15s
32 flush_jitter = "0s"
33
34 ## By default or when set to "0s", precision will be set to the same
35 ## timestamp order as the collection interval, with the maximum being 1s.
36 ## ie, when interval = "10s", precision will be "1s"
37 ## when interval = "250ms", precision will be "1ms"
38 ## Precision will NOT be used for service inputs. It is up to each individual
39 ## service input to set the timestamp at the appropriate precision.
40 ## Valid time units are "ns", "us" (or "µs"), "ms", "s".
41 precision = ""
42
43 ## Logging configuration:
44 ## Run telegraf with debug log messages.
45 debug = false
46 ## Run telegraf in quiet mode (error log messages only).
47 quiet = false
48 ## Specify the log file name. The empty string means to log to stderr.
49 logfile = ""
50
51 ## Override default hostname, if empty use os.Hostname()
52 hostname = ""
53 ## If set to true, do not set the "host" tag in the telegraf agent.
54 omit_hostname = false
55 [[outputs.influxdb_v2]]
56 ## The URLs of the InfluxDB cluster nodes.
57 ##
58 ## Multiple URLs can be specified for a single cluster, only ONE of the
59 ## urls will be written to each interval.
60 ## urls exp: http://127.0.0.1:8086
61 urls = ["http://localhost:8086"]
62
63 ## Token for authentication.
64 token = "o-qItcD-
a8YntwdHZF3Xz3oWX1LWnTzTVIDN50bV0EqTErbkClhR8ZsUPJDj2XoFYVffa3fDMdXreI55XSD0Uw=="
65
66 ## Organization is the name of the organization you wish to write to; must exist.
67 organization = "BTS"
68
69 ## Destination bucket to write into.
70 bucket = "youtube"
71 [[inputs.youtube]]
72 channels = [
73 "UCLA_DiR1FfKNvjuUpBHmylQ",
74 "UCuJW0p17WofTPFqJZPz8jYg",
75 "UCfaSK6K7hB3rAH7MLh8o6pw"
76 ]
77 api_key = "AIzaSyAirfp6PC-qMkYC0S6iXU0tK0d8mr5GW24"
78
```

The screenshot shows a text editor window titled 'youtube.conf' with a dark theme. The file contains configuration for the Telegraf agent's InfluxDB output and YouTube input. Three specific lines are highlighted with grey boxes and annotated with purple callouts:

- Line 64: `token = "o-qItcD-a8YntwdHZF3Xz3oWX1LWnTzTVIDN50bV0EqTErbkClhR8ZsUPJDj2XoFYVffa3fDMdXreI55XSD0Uw=="` is annotated with a purple box labeled 'Influx Token'.
- Lines 73-75: The `channels` array containing three YouTube channel IDs is annotated with a purple box labeled 'Youtube channels ID: Nasa, Hubble, SpaceX'.
- Line 77: `api_key = "AIzaSyAirfp6PC-qMkYC0S6iXU0tK0d8mr5GW24"` is annotated with a purple box labeled 'Youtube Data API V3 Key'.

The status bar at the bottom indicates 'Plain Text', 'Tab Width: 8', 'Ln 65, Col 1', and 'INS'.

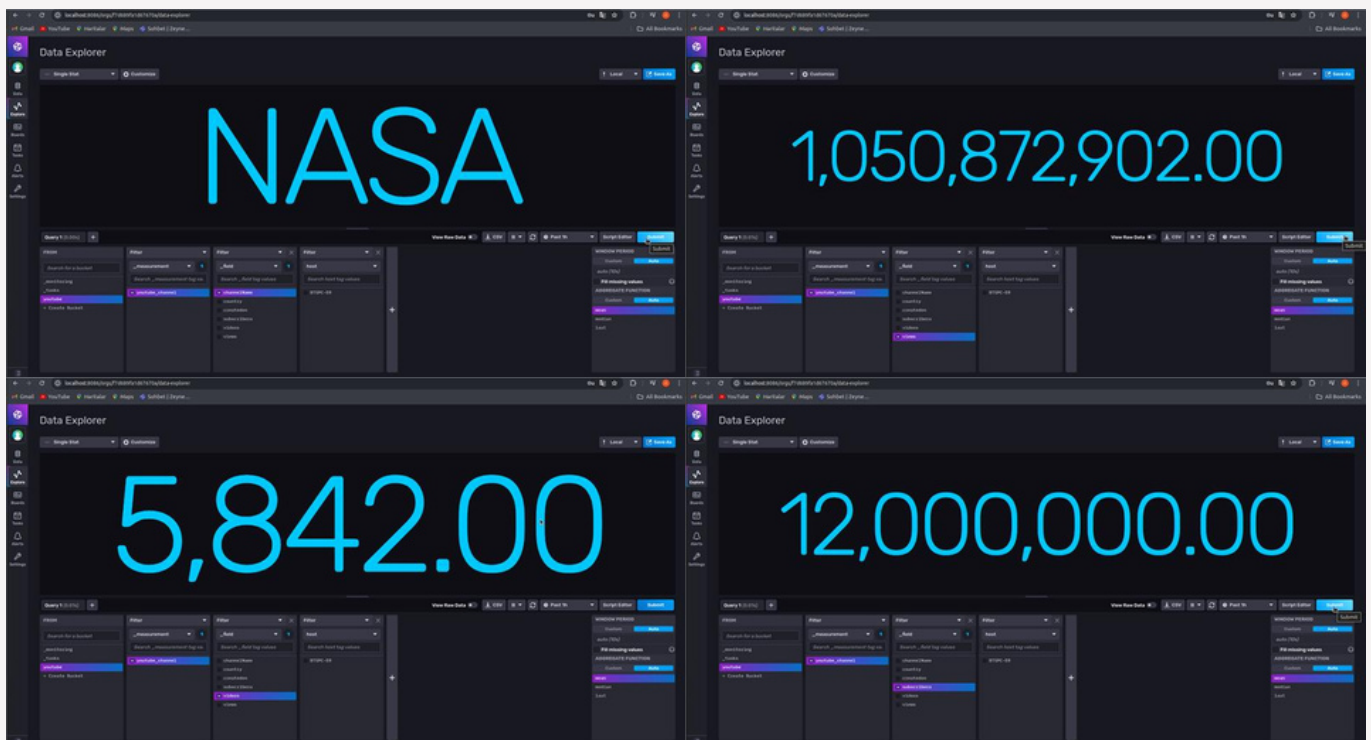


## 10. Start Telegraf conf.

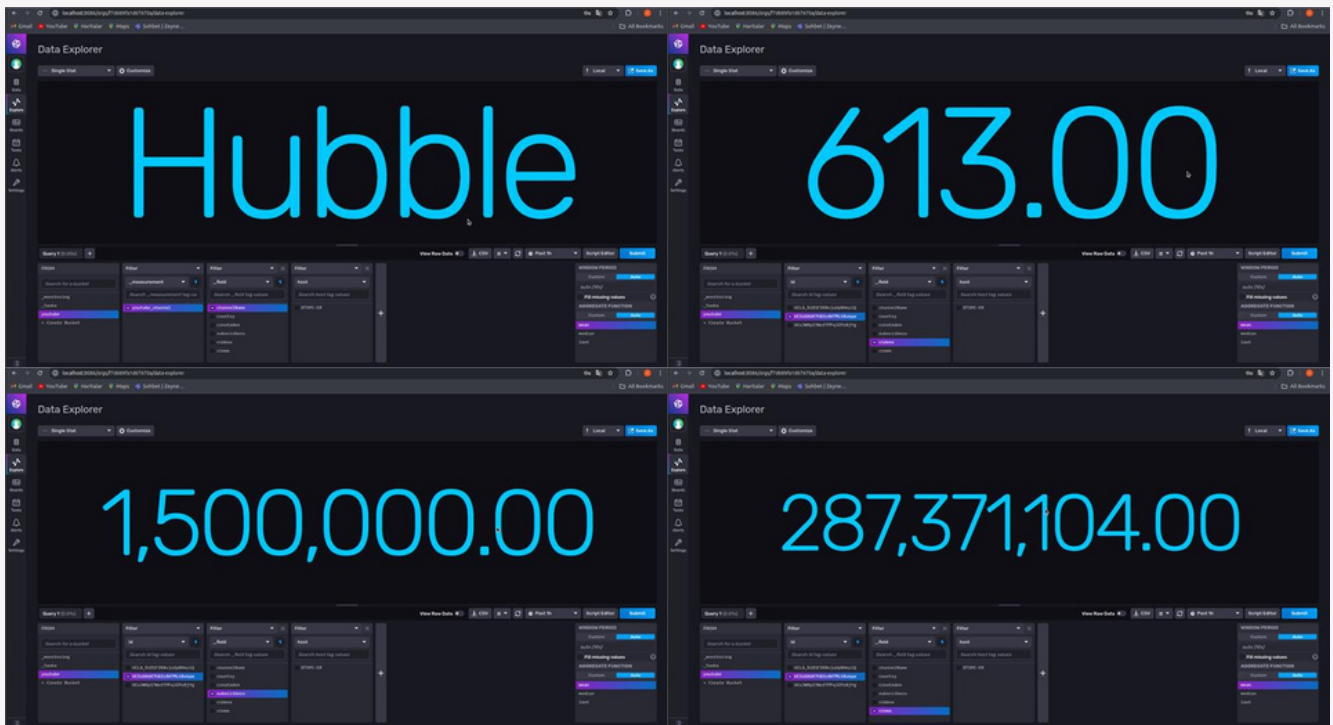
```
zeynep@BTSPC-ER: ~/go/src/github.com/influxdata/telegraf
zeynep@BTSPC-ER:~/go/src/github.com/influxdata/telegraf$ ./telegraf --config /home/zeynep/Downloads/youtube.conf
2024-06-05T19:25:20Z I! Loading config: /home/zeynep/Downloads/youtube.conf
2024-06-05T19:25:20Z I! Starting Telegraf 1.31.0-c2a67ecd brought to you by InfluxData the makers of InfluxDB
2024-06-05T19:25:20Z I! Available plugins: 235 inputs, 9 aggregators, 32 processors, 26 parsers, 60 outputs, 6 secret-stores
2024-06-05T19:25:20Z I! Loaded inputs: youtube
2024-06-05T19:25:20Z I! Loaded aggregators:
2024-06-05T19:25:20Z I! Loaded processors:
2024-06-05T19:25:20Z I! Loaded secretstores:
2024-06-05T19:25:20Z I! Loaded outputs: influxdb_v2
2024-06-05T19:25:20Z I! Tags enabled: host=BTSPC-ER
2024-06-05T19:25:20Z I! [agent] Config: Interval:10s, Quiet:false, Hostname:"BTSPC-ER", Flush Interval:10s
channelInfo &{<nil> <nil> <nil> <nil> <nil> ZiIfEEX5Kc-bdq38rDpmfqNrH68 UCuJW0p17WofTPFqJZPz8jYg youtube#channel map[] 0xc00298ac80 0xc002993200 <nil> <nil> {0 map[]} [] []}
channelInfo &{<nil> <nil> <nil> <nil> <nil> TxM_PgOV126yDBr6ZxQaUjxH-9w UCfaSK6K7hB3rAH7MLh8o6pw youtube#channel map[] 0xc00298ad20 0xc0029932c0 <nil> <nil> {0 map[]} [] []}
channelInfo &{<nil> <nil> <nil> <nil> <nil> HZa295d0B6c0BouCFM-gakbHwV0 UCLA-D3R15fKNu4u10RHMu10 youtube#channel map[] 0xc00298ad20 0xc0029932c0 <nil> <nil> {0 map[]} [] []}
```

## 11. Complete Project

- NASA



- Hubble



- SpaceX

