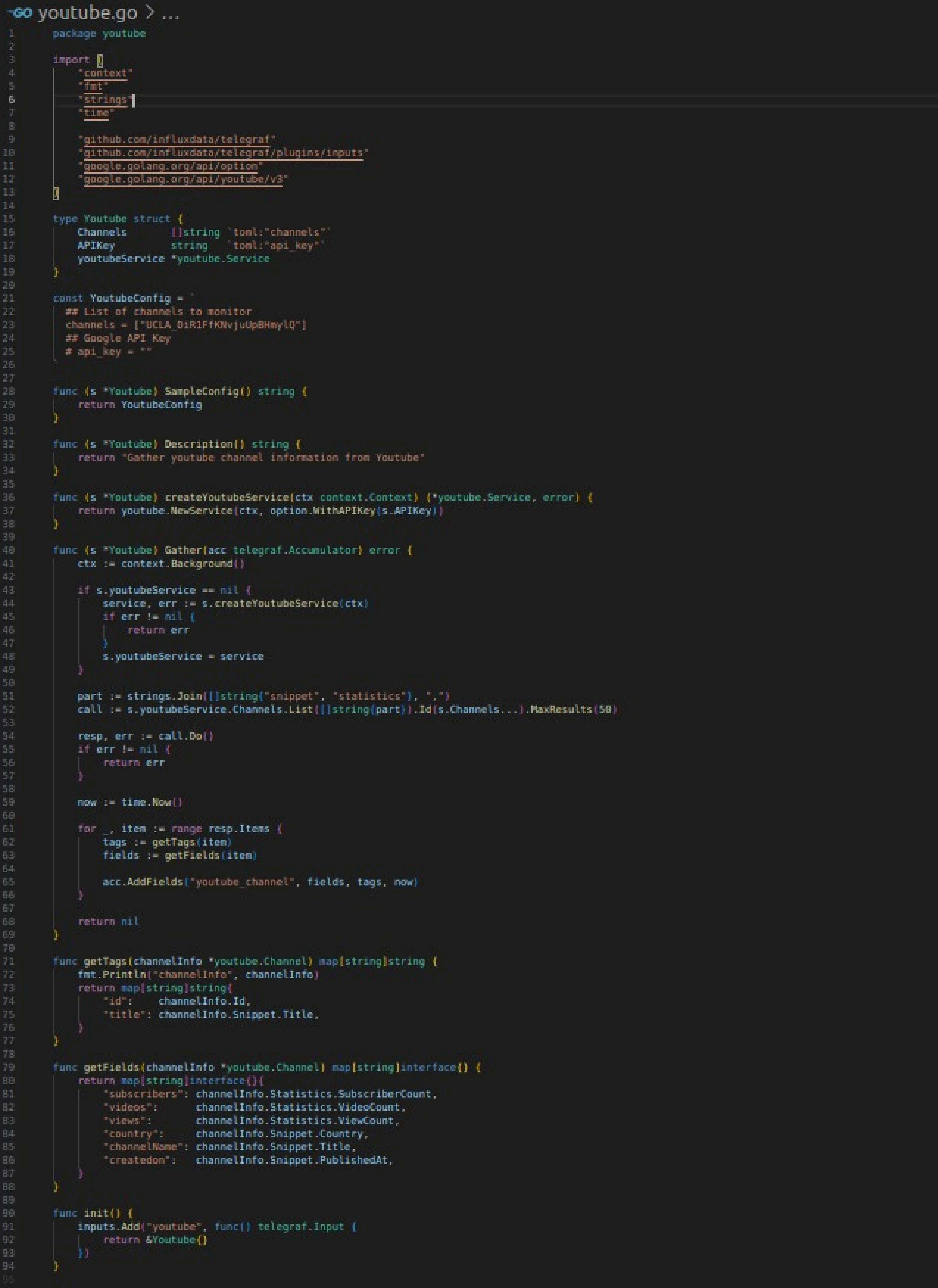
CUSTOM YOUTUBE PLUGİN

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:

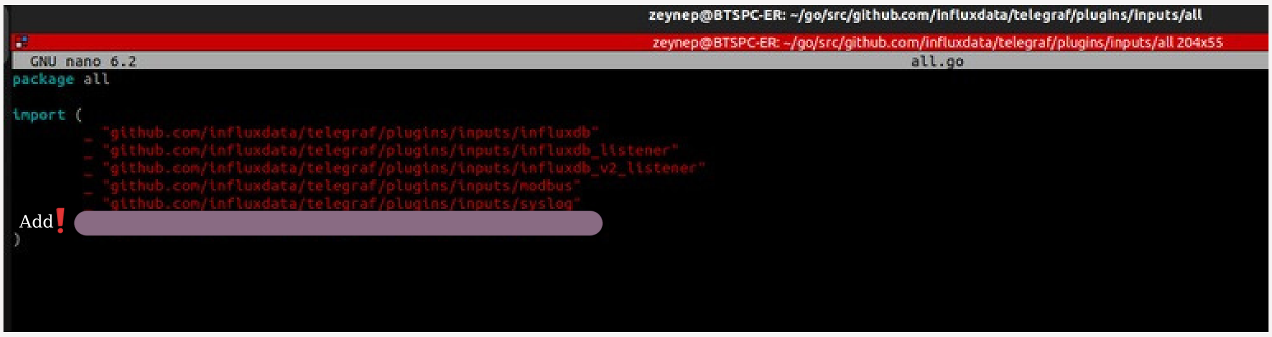
- Subscriber Count
- Video Count
- Total View Count
- Channel Country
- Channel Name
- Channel Creation Date



youtube.go > ...

```go
package youtube

import (
    "context"
    "fmt"
    "strings"
    "time"

    "github.com/influxdata/telegraf"
    "github.com/influxdata/telegraf/plugins/inputs"
    "google.golang.org/api/option"
    "google.golang.org/api/youtube/v3"
)

type Youtube struct {
    Channels       []string `toml:"channels"`
    APIKey         string   `toml:"api_key"`
    youtubeService *youtube.Service
}

const YoutubeConfig = `
  ## List of channels to monitor
  channels = ["UCLA_DiR1FfKNvjuUpBHmylQ"]
  ## Google API Key
  # api_key = ""
`

func (s *Youtube) SampleConfig() string {
    return YoutubeConfig
}

func (s *Youtube) Description() string {
    return "Gather youtube channel information from Youtube"
}

func (s *Youtube) createYoutubeService(ctx context.Context) (*youtube.Service, error) {
    return youtube.NewService(ctx, option.WithAPIKey(s.APIKey))
}

func (s *Youtube) Gather(acc telegraf.Accumulator) error {
    ctx := context.Background()

    if s.youtubeService == nil {
        service, err := s.createYoutubeService(ctx)
        if err != nil {
            return err
        }
        s.youtubeService = service
    }

    part := strings.Join([]string{"snippet", "statistics"}, ",")
    call := s.youtubeService.Channels.List([]string{part}).Id(s.Channels...).MaxResults(50)

    resp, err := call.Do()
    if err != nil {
        return err
    }

    now := time.Now()

    for _, item := range resp.Items {
        tags := getTags(item)
        fields := getFields(item)

        acc.AddFields("youtube_channel", fields, tags, now)
    }

    return nil
}

func getTags(channelInfo *youtube.Channel) map[string]string {
    fmt.Println("channelInfo", channelInfo)
    return map[string]string{
        "id":    channelInfo.Id,
        "title": channelInfo.Snippet.Title,
    }
}

func getFields(channelInfo *youtube.Channel) map[string]interface{} {
    return map[string]interface{}{
        "subscribers": channelInfo.Statistics.SubscriberCount,
        "videos":      channelInfo.Statistics.VideoCount,
        "views":       channelInfo.Statistics.ViewCount,
        "country":     channelInfo.Snippet.Country,
        "channelName": channelInfo.Snippet.Title,
        "createdon":   channelInfo.Snippet.PublishedAt,
    }
}

func init() {
    inputs.Add("youtube", func() telegraf.Input {
        return &Youtube{}
    })
}
```

2. Edit All.go File

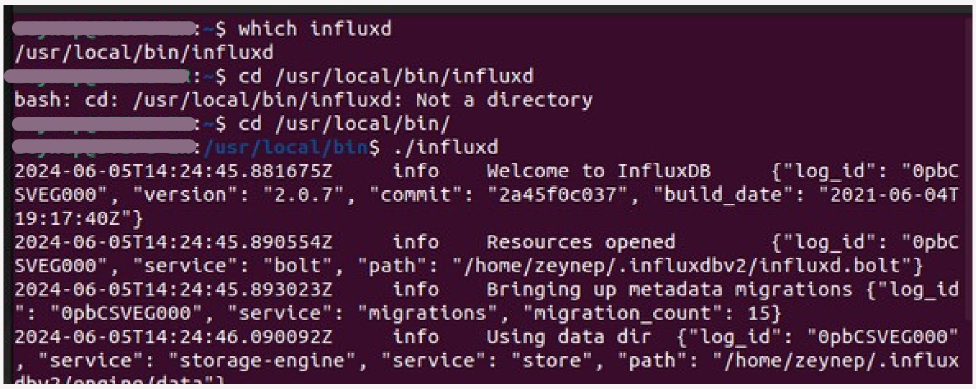


```
zeynep@BTSPC-ER: ~/go/src/github.com/influxdata/telegraf/plugins/inputs/all
zeynep@BTSPC-ER: ~/go/src/github.com/influxdata/telegraf/plugins/inputs/all 204x55
  GNU nano 6.2                                                        all.go
package all

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"
 Add❗ [redacted]
)
```

3.Create telegraf binary file

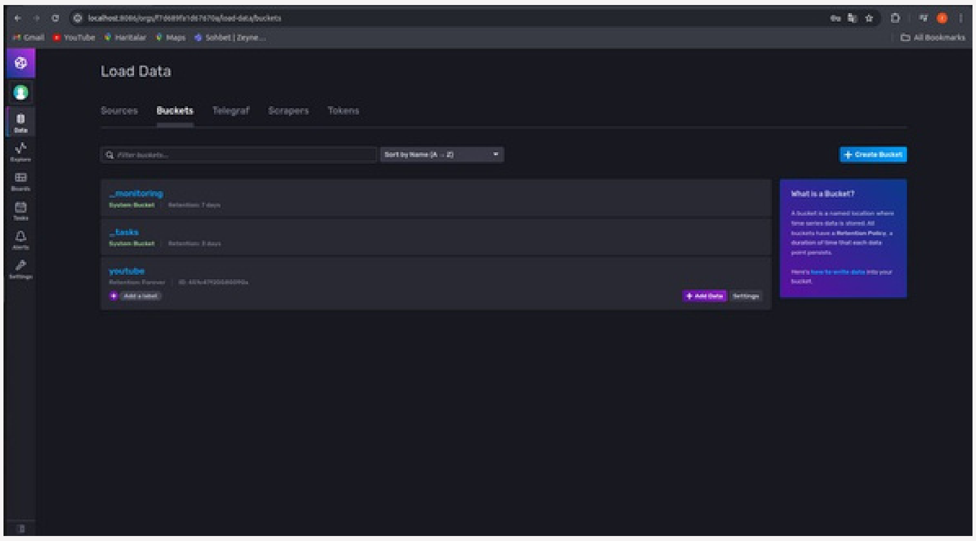
zeynep:~/go/src/telegraf$ make telegraf

4. Start Influxdb



```
[redacted]:~$ which influxd
/usr/local/bin/influxd
[redacted]:~$ cd /usr/local/bin/influxd
bash: cd: /usr/local/bin/influxd: Not a directory
[redacted]:~$ cd /usr/local/bin/
[redacted]:/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
```

5.Create Youtube Bucket



Load Data

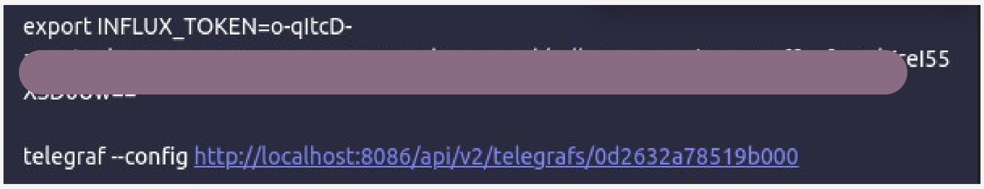
Sources | Buckets | Telegraf | Scrapers | Tokens

_monitoring
System Bucket | Retention: 7 days

_tasks
System Bucket | Retention: 3 days

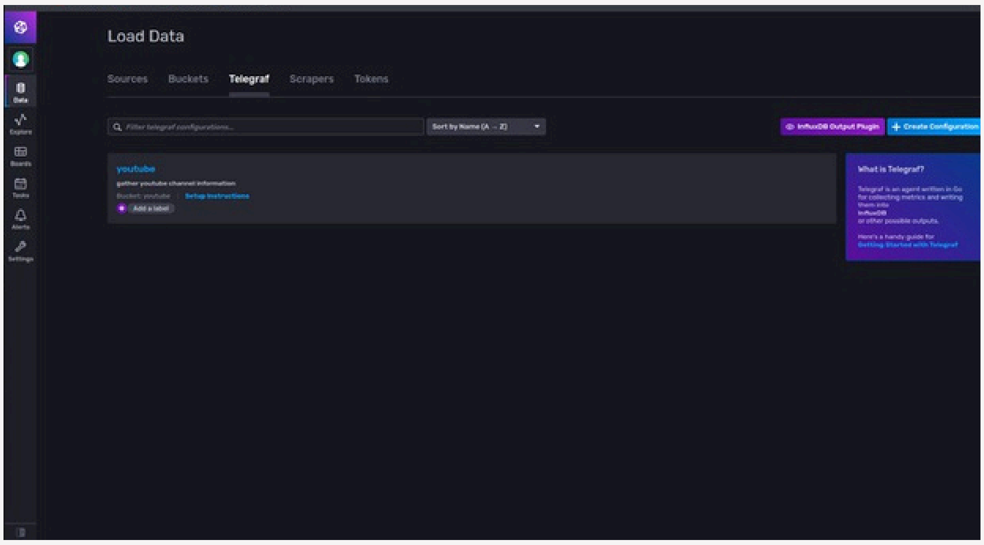
youtube
Retention: Forever

6.Run token commands



```
export INFLUX_TOKEN=o-qltcD-
[redacted]

telegraf --config http://localhost:8086/api/v2/telegrafs/0d2632a78519b000
```

7.Create Youtube conf. File

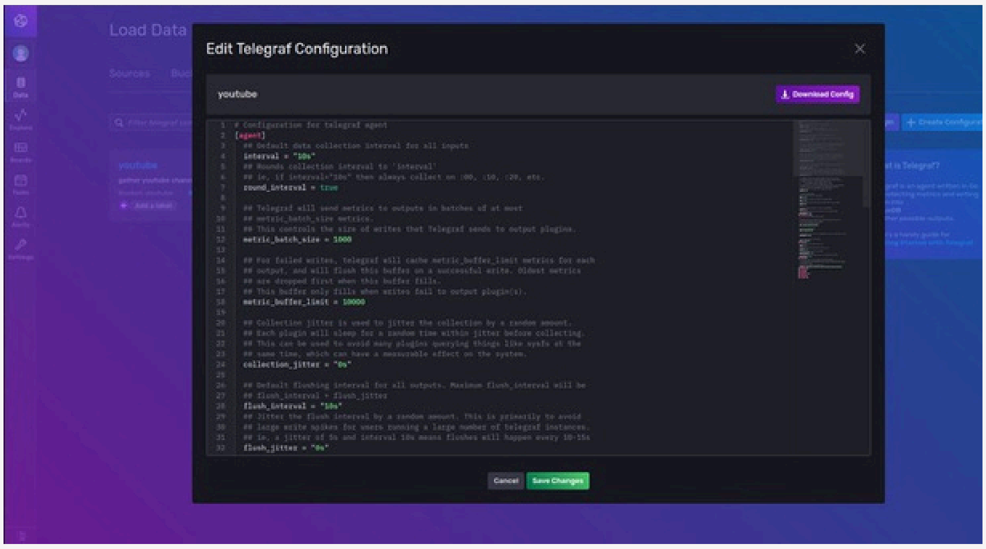


youtube
gather youtube channel information
Bucket: youtube

What is Telegraf?

Telegraf is an agent written in Go for collecting metrics and writing them into InfluxDB or other possible outputs.

Here's a handy guide for Getting Started with Telegraf

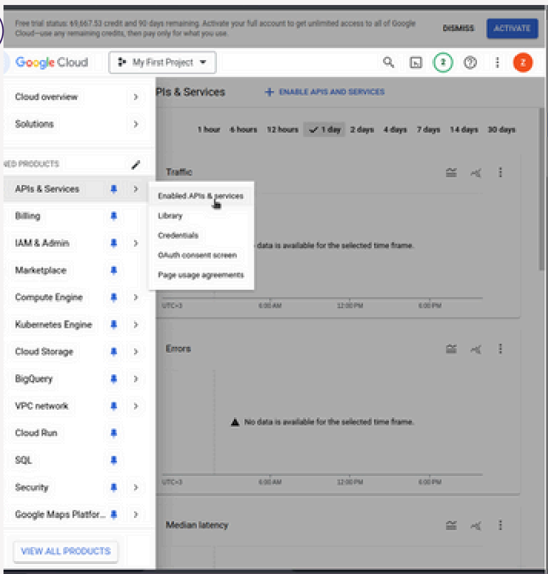


youtube

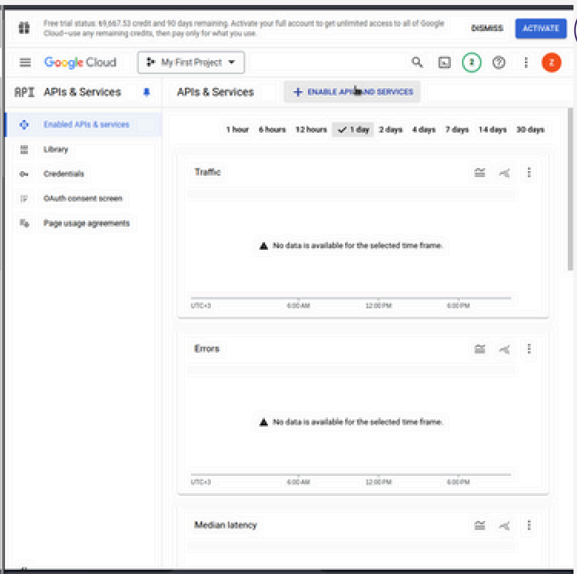
```
# Configuration for telegraf agent
[agent]
  ## Default data collection interval for all inputs
  interval = "10s"
  ## Rounds collection interval to 'interval'
  ## ie, if interval="10s" then always collect on :00, :10, :20, etc.
  round_interval = true

  ## Telegraf will send metrics to outputs in batches of at most
  ## metric_batch_size metrics.
  ## This controls the size of writes that Telegraf sends to output plugins.
  metric_batch_size = 1000

  ## For failed writes, telegraf will cache metric_buffer_limit metrics for each
  ## output, and will flush this buffer on a successful write. Oldest metrics
  ## are dropped first when this buffer fills.
  ## This buffer only fills when writes fail to output plugin(s).
  metric_buffer_limit = 10000

  ## Collection jitter is used to jitter the collection by a random amount.
  ## Each plugin will sleep for a random time within jitter before collecting.
  ## This can be used to avoid many plugins querying things like sysfs at the
  ## same time, which can have a measurable effect on the system.
  collection_jitter = "0s"

  ## Default flushing interval for all outputs. Maximum flush_interval will be
  ## flush_interval + flush_jitter
  flush_interval = "10s"
  ## Jitter the flush interval by a random amount. This is primarily to avoid
  ## large write spikes for users running a large number of telegraf instances.
  ## ie, a jitter of 5s and interval 10s means flushes will happen every 10-15s
  flush_jitter = "0s"
```

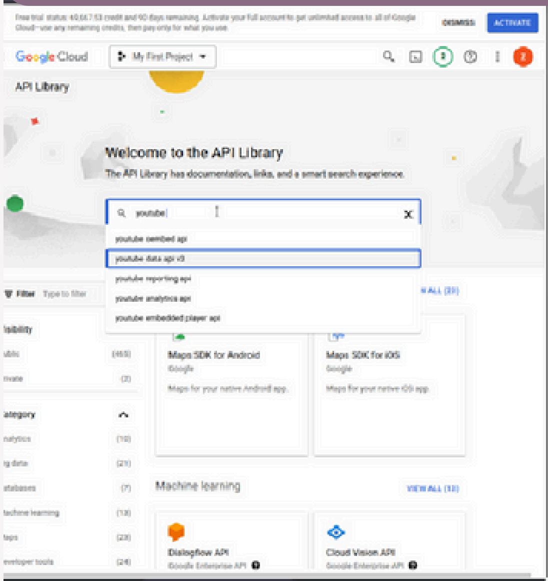
8.Google cloud/Youtube Data API v3 Key

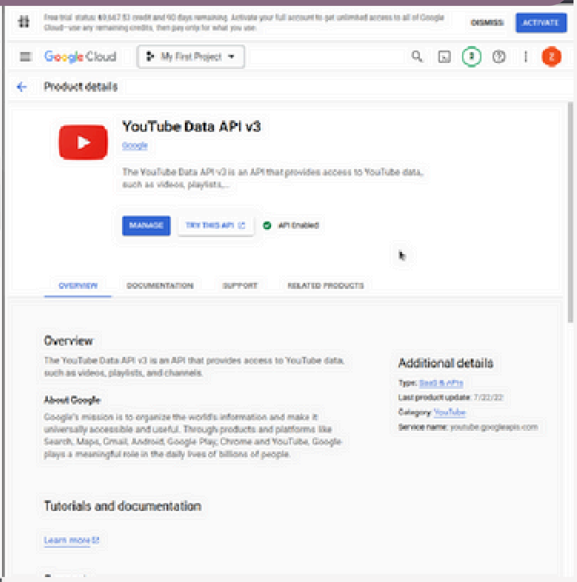
1

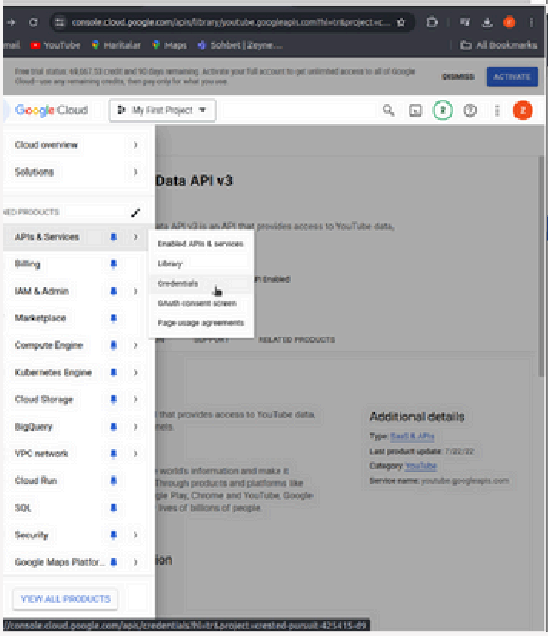


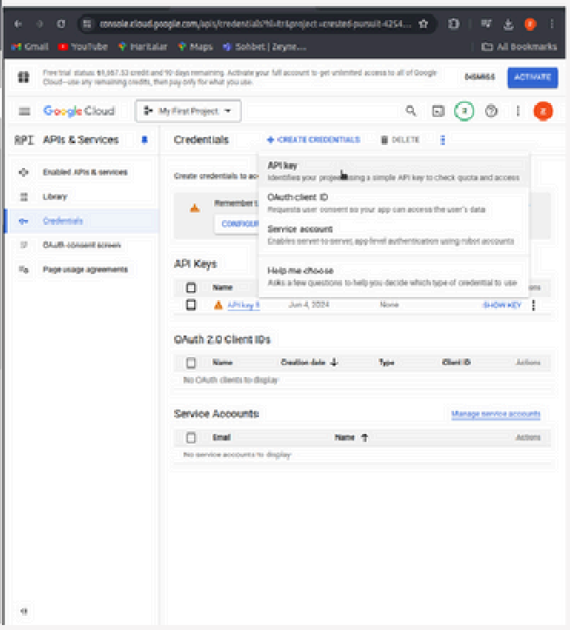
2



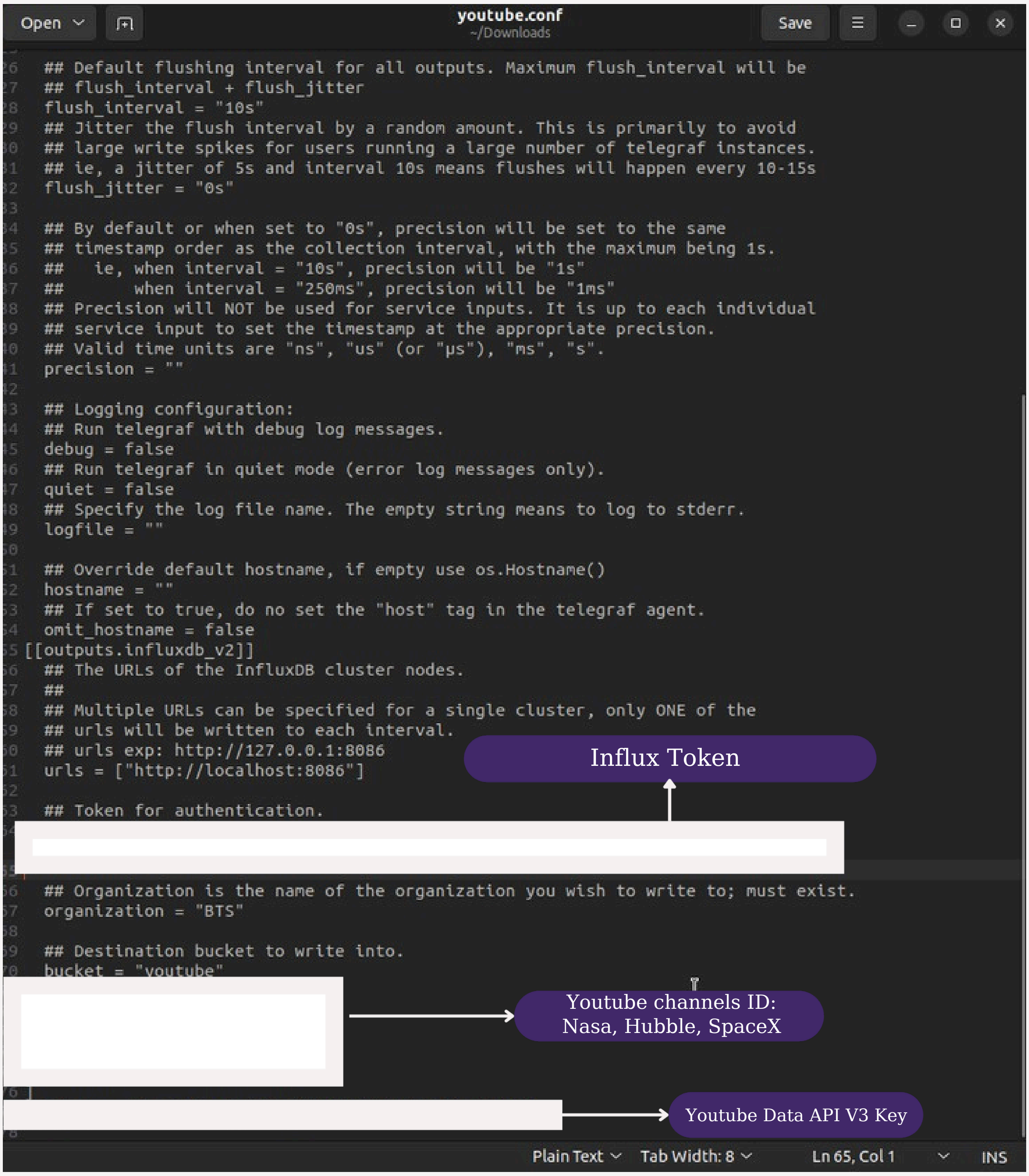

API Library
Welcome to the API Library
The API Library has documentation, links, and a smart search experience.
youtube
youtube oembed api
youtube data api v3
youtube reporting api
youtube analytics api
youtube embedded player api





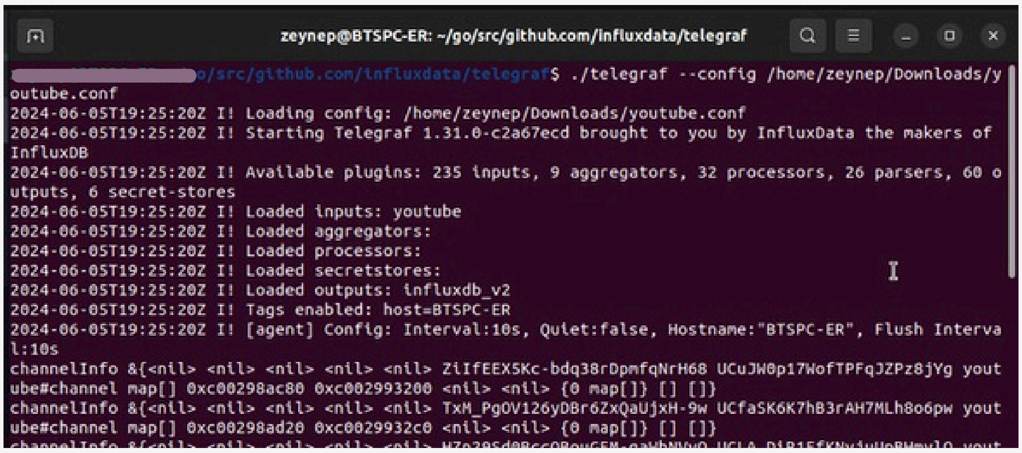


9.Edit Telegraf Conf.



```
  ## Default flushing interval for all outputs. Maximum flush_interval will be
  ## flush_interval + flush_jitter
  flush_interval = "10s"
  ## Jitter the flush interval by a random amount. This is primarily to avoid
  ## large write spikes for users running a large number of telegraf instances.
  ## ie, a jitter of 5s and interval 10s means flushes will happen every 10-15s
  flush_jitter = "0s"

  ## By default or when set to "0s", precision will be set to the same
  ## timestamp order as the collection interval, with the maximum being 1s.
  ##   ie, when interval = "10s", precision will be "1s"
  ##       when interval = "250ms", precision will be "1ms"
  ## Precision will NOT be used for service inputs. It is up to each individual
  ## service input to set the timestamp at the appropriate precision.
  ## Valid time units are "ns", "us" (or "µs"), "ms", "s".
  precision = ""

  ## Logging configuration:
  ## Run telegraf with debug log messages.
  debug = false
  ## Run telegraf in quiet mode (error log messages only).
  quiet = false
  ## Specify the log file name. The empty string means to log to stderr.
  logfile = ""

  ## Override default hostname, if empty use os.Hostname()
  hostname = ""
  ## If set to true, do no set the "host" tag in the telegraf agent.
  omit_hostname = false
[[outputs.influxdb_v2]]
  ## The URLs of the InfluxDB cluster nodes.
  ##
  ## Multiple URLs can be specified for a single cluster, only ONE of the
  ## urls will be written to each interval.
  ## urls exp: http://127.0.0.1:8086
  urls = ["http://localhost:8086"]

  ## Token for authentication.
[redacted]

  ## Organization is the name of the organization you wish to write to; must exist.
  organization = "BTS"

  ## Destination bucket to write into.
  bucket = "youtube"
[redacted]

[redacted]
```

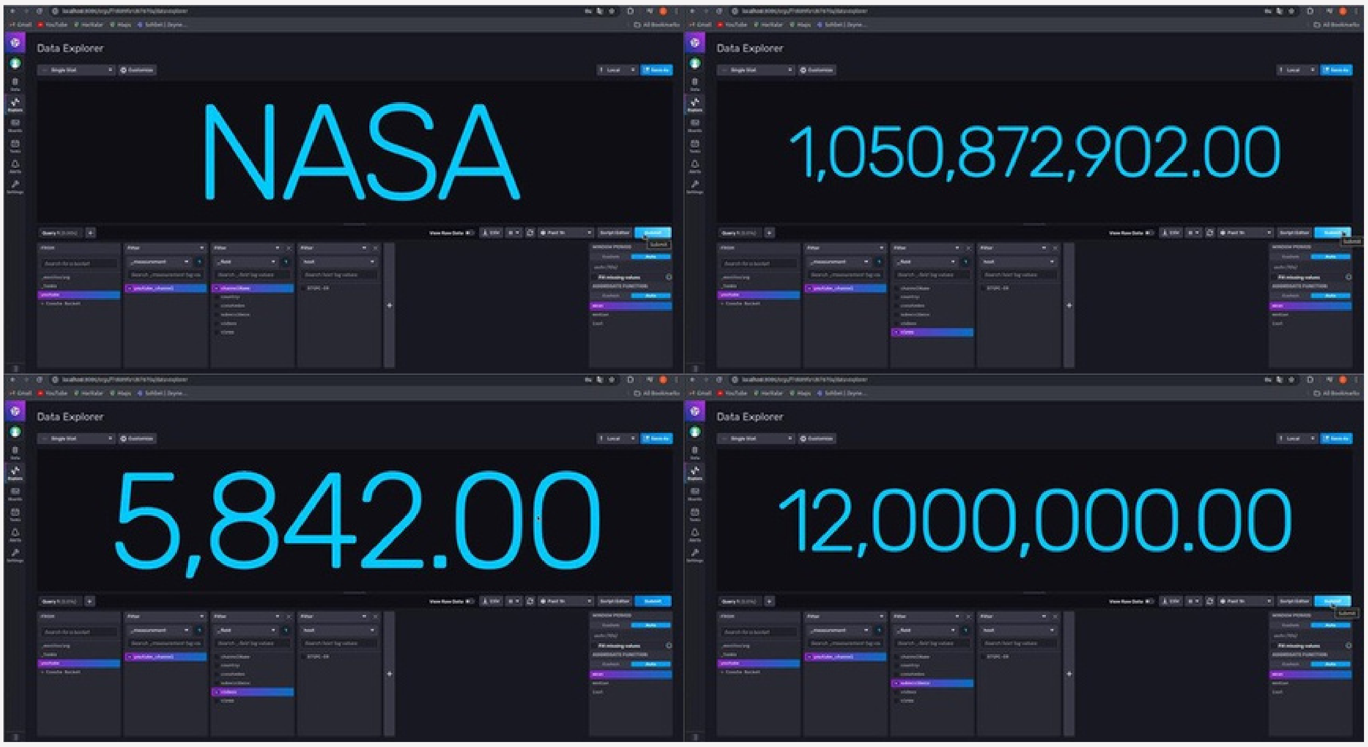
10. Start Telegraf conf.



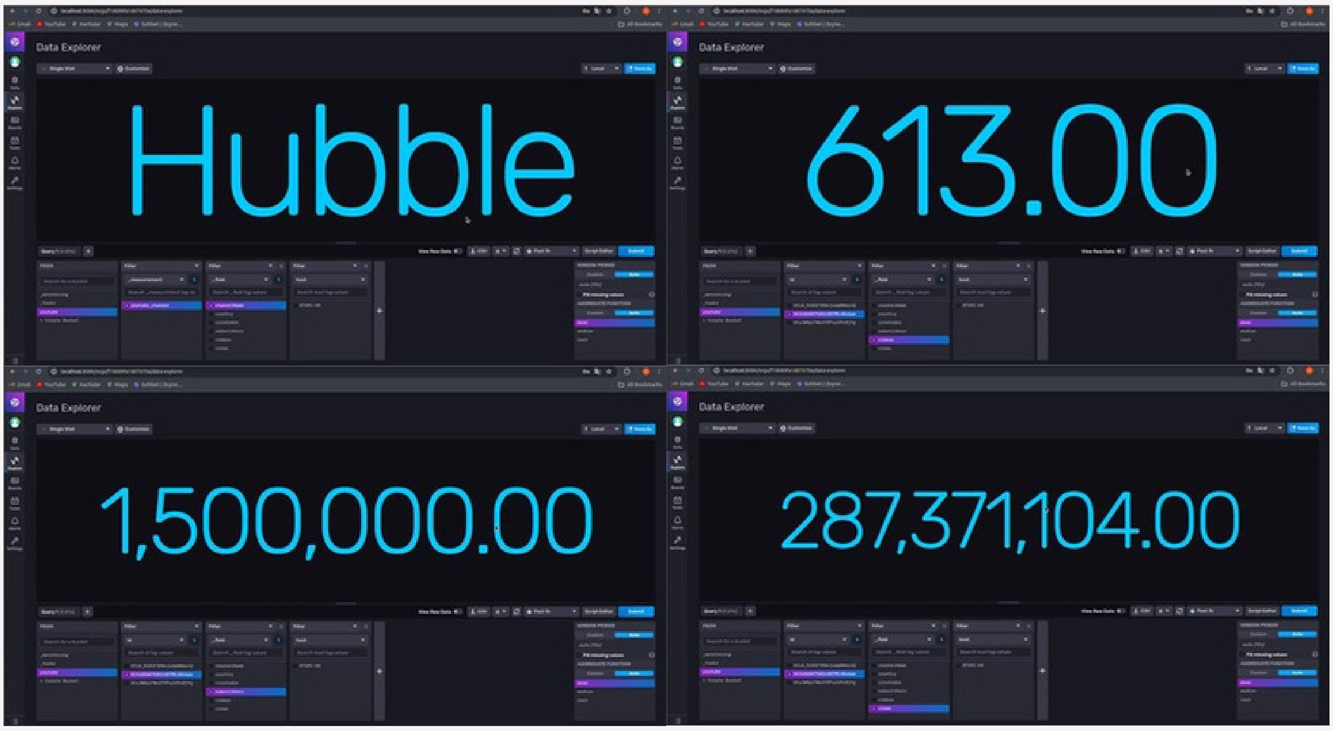
```
zeynep@BTSPC-ER: ~/go/src/github.com/influxdata/telegraf
[redacted]o/src/github.com/influxdata/telegraf$ ./telegraf --config /home/zeynep/Downloads/y
outube.conf
2024-06-05T19:25:20Z I! Loading config: /home/zeynep/Downloads/youtube.conf
2024-06-05T19:25:20Z I! Starting Telegraf 1.31.0-c2a67ecd brought 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 o
utputs, 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 Interva
l:10s
channelInfo &{<nil> <nil> <nil> <nil> <nil> ZiIfEEX5Kc-bdq38rDpmfqNrH68 UCuJW0p17WofTPFqJZPz8jYg yout
ube#channel map[] 0xc00298ac80 0xc002993200 <nil> <nil> {0 map[]} [] []}
channelInfo &{<nil> <nil> <nil> <nil> <nil> TxM_PgOV126yDBr6ZxQaUjxH-9w UCfaSK6K7hB3rAH7MLh8o6pw yout
ube#channel map[] 0xc00298ad20 0xc0029932c0 <nil> <nil> {0 map[]} [] []}
```

11.Complete Project

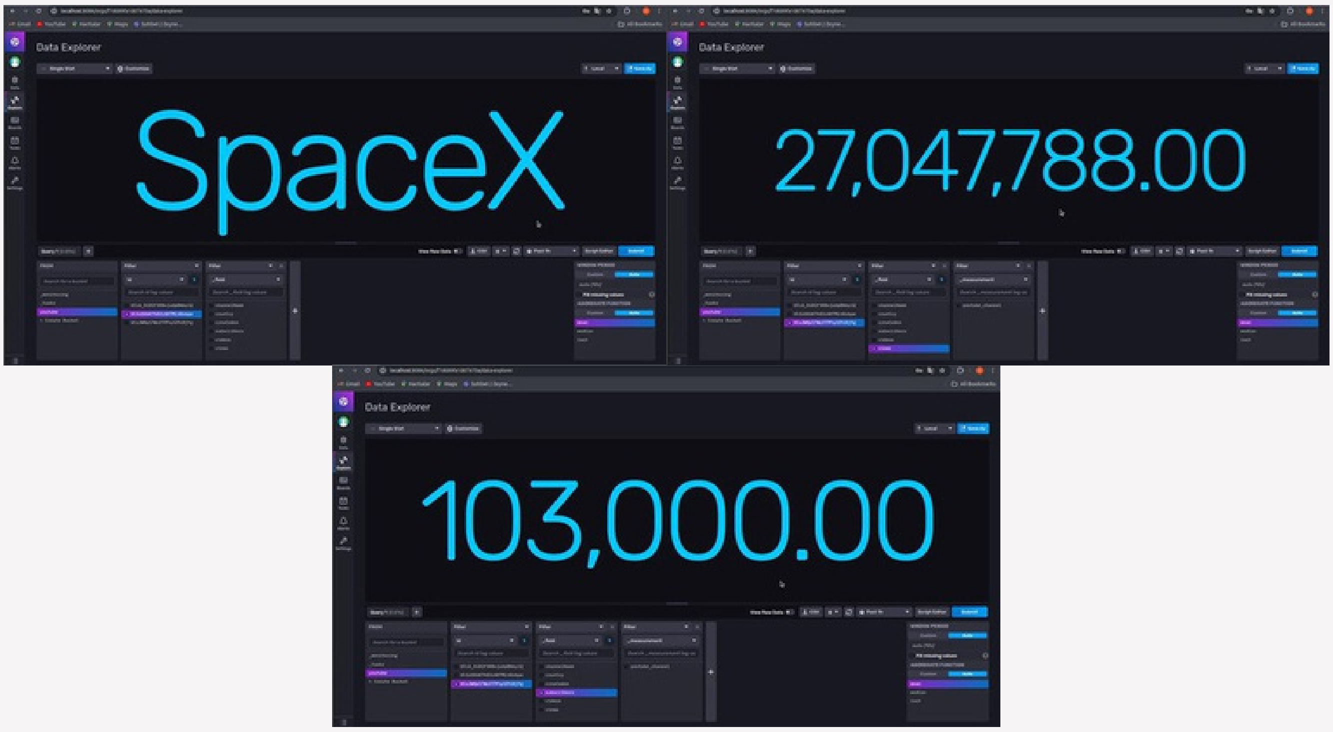
- NASA



- Hubble



- SpaceX


Data Explorer
SpaceX
27,047,788.00
103,000.00