CUSTOM YOUTUBE PLUGİN

Project Objective

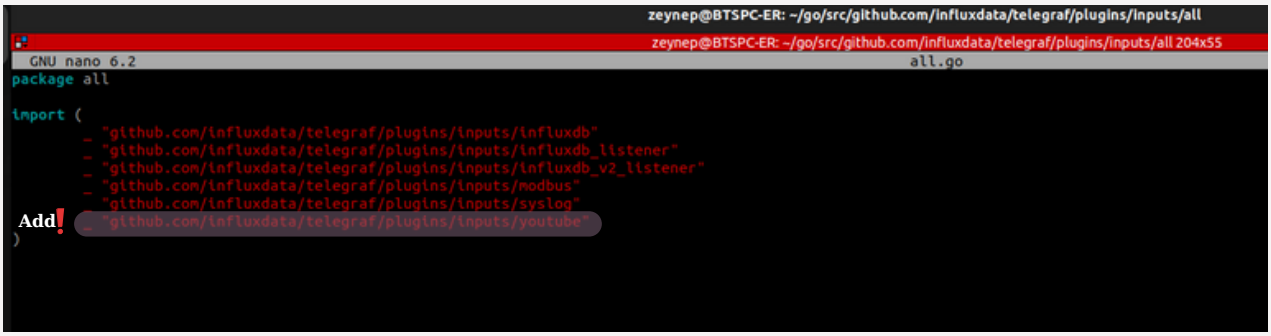
The goal of this project is to automatically collect and analyze key statistics from YouTube channels. Using the YouTube Data API v3, our Telegraf plugin gathers the following data for specified channels:

- 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{}
    })
}
```

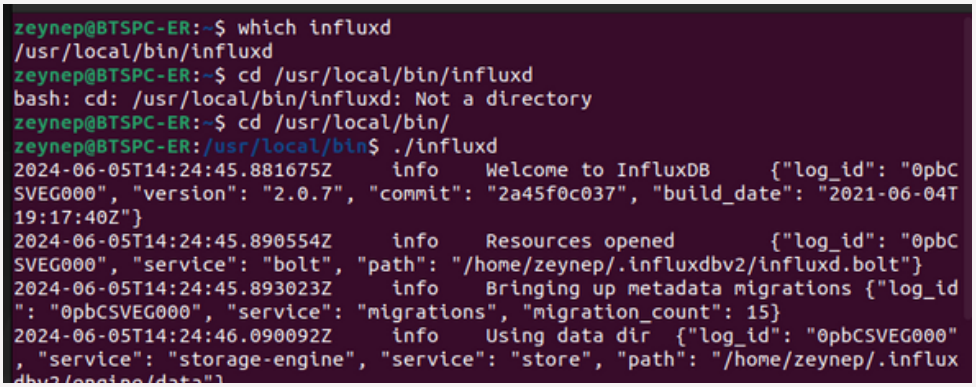
2. Edit All.go File



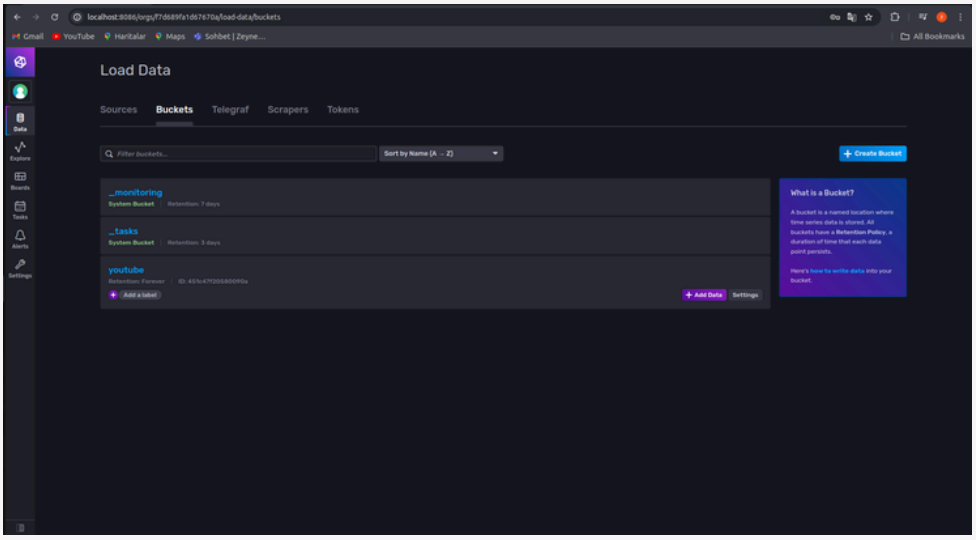
3.Create telegraf binary file

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

4. Start Influxdb



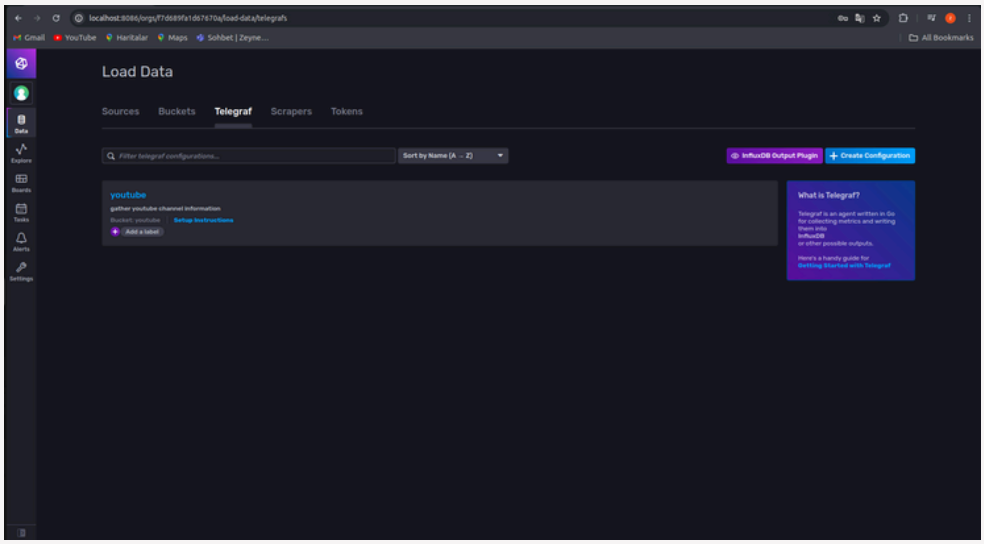
5.Create Youtube Bucket

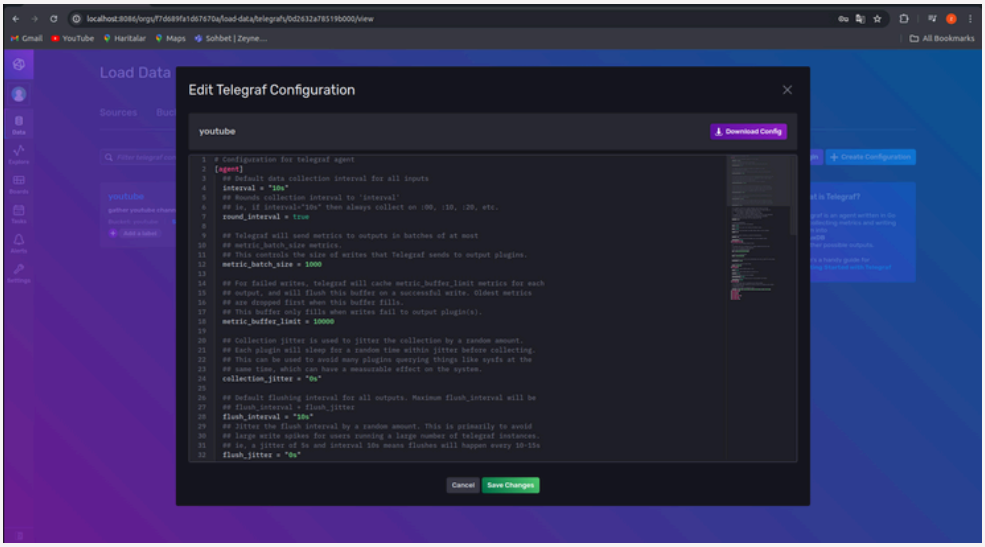


6.Run token commands

```
export INFLUX_TOKEN=o-qItcD-
a8YntwdHZF3Xz3oWX1LWnTzTVIDN5ObV0EqTErbkClhR8ZsUPJDj2XoFYVffa3fDMdXreI55
XSD0Uw==

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

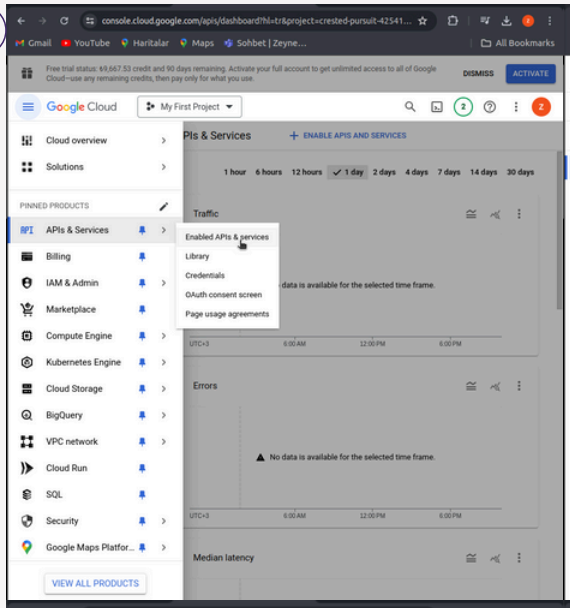
7.Create Youtube conf. File



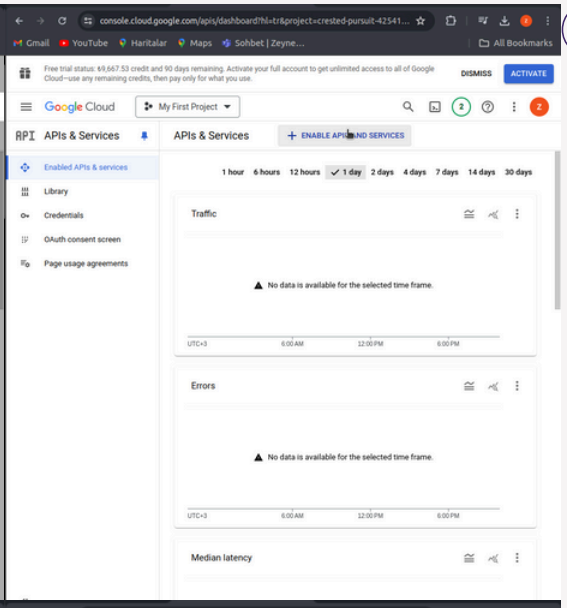


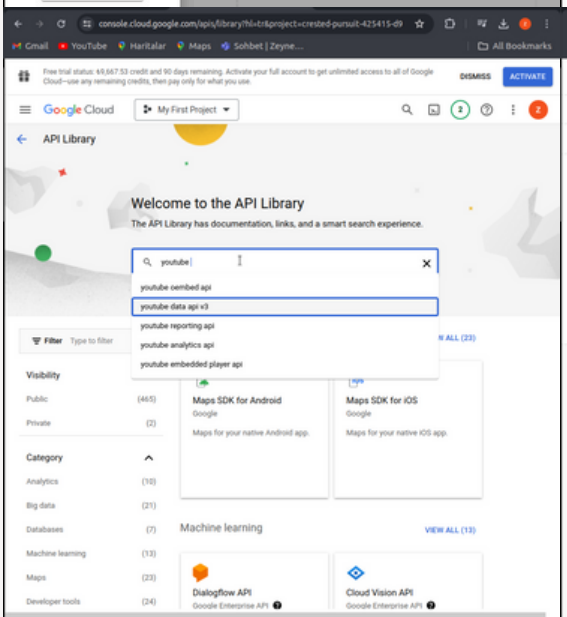
8.Google cloud/Youtube Data API v3 Key

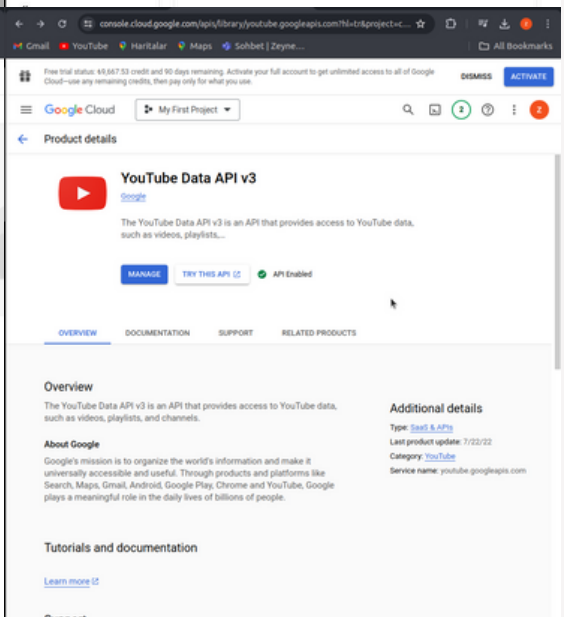
1

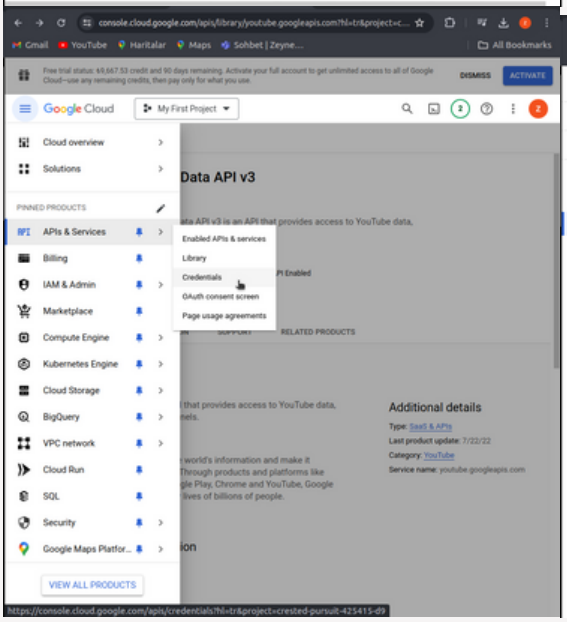


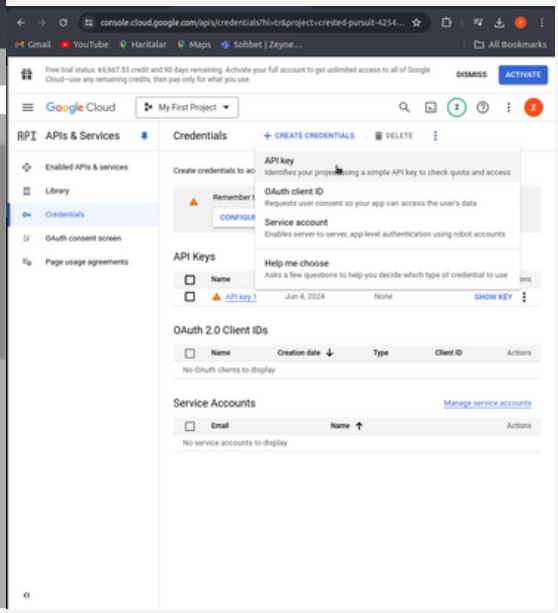
2



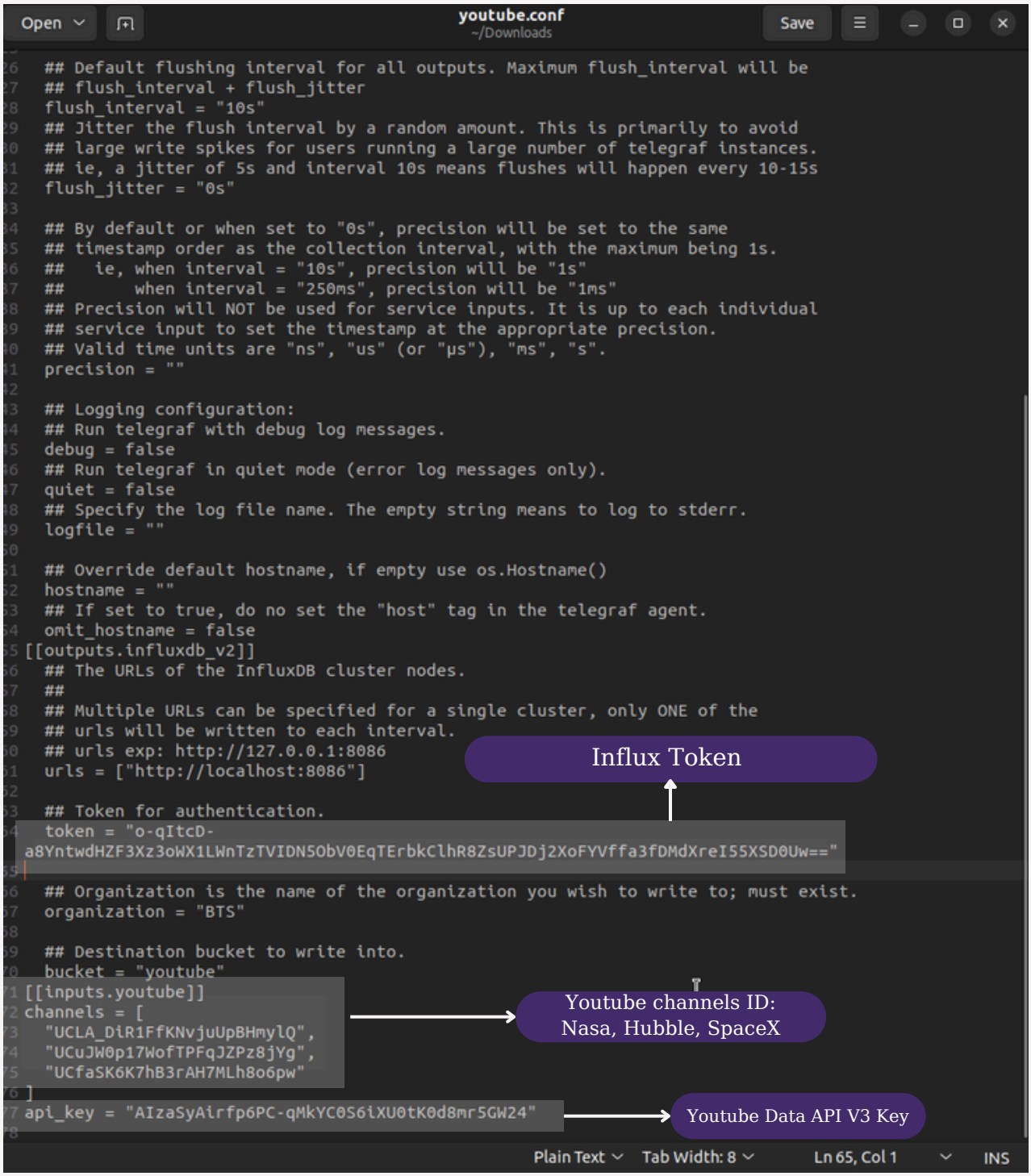




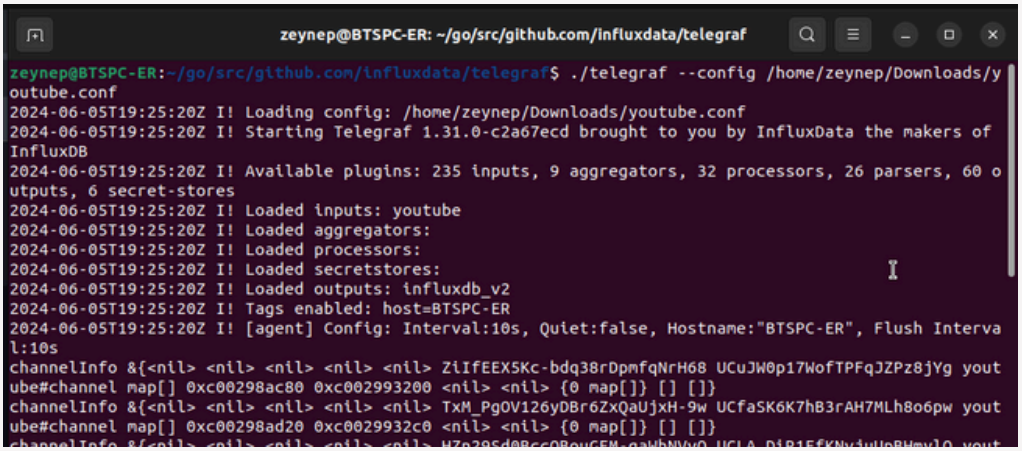




9.Edit Telegraf Conf.

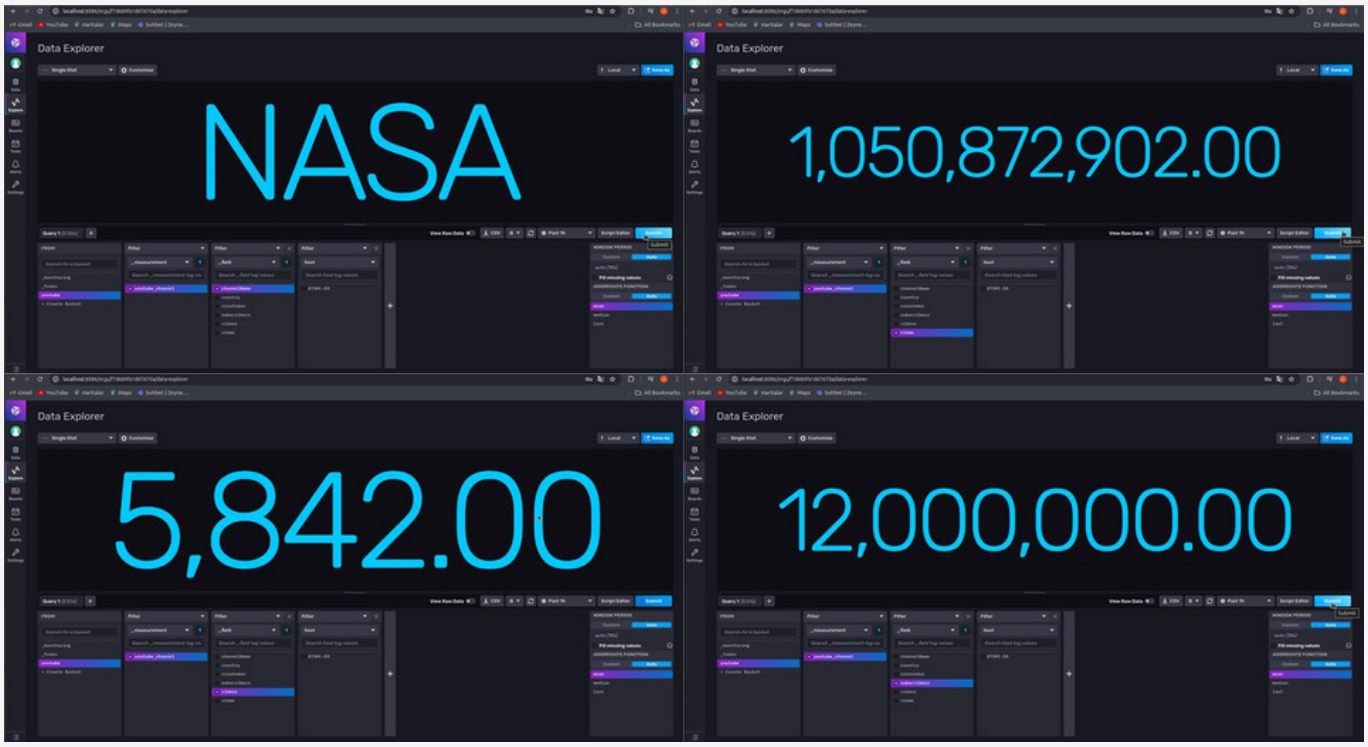


10. Start Telegraf conf.

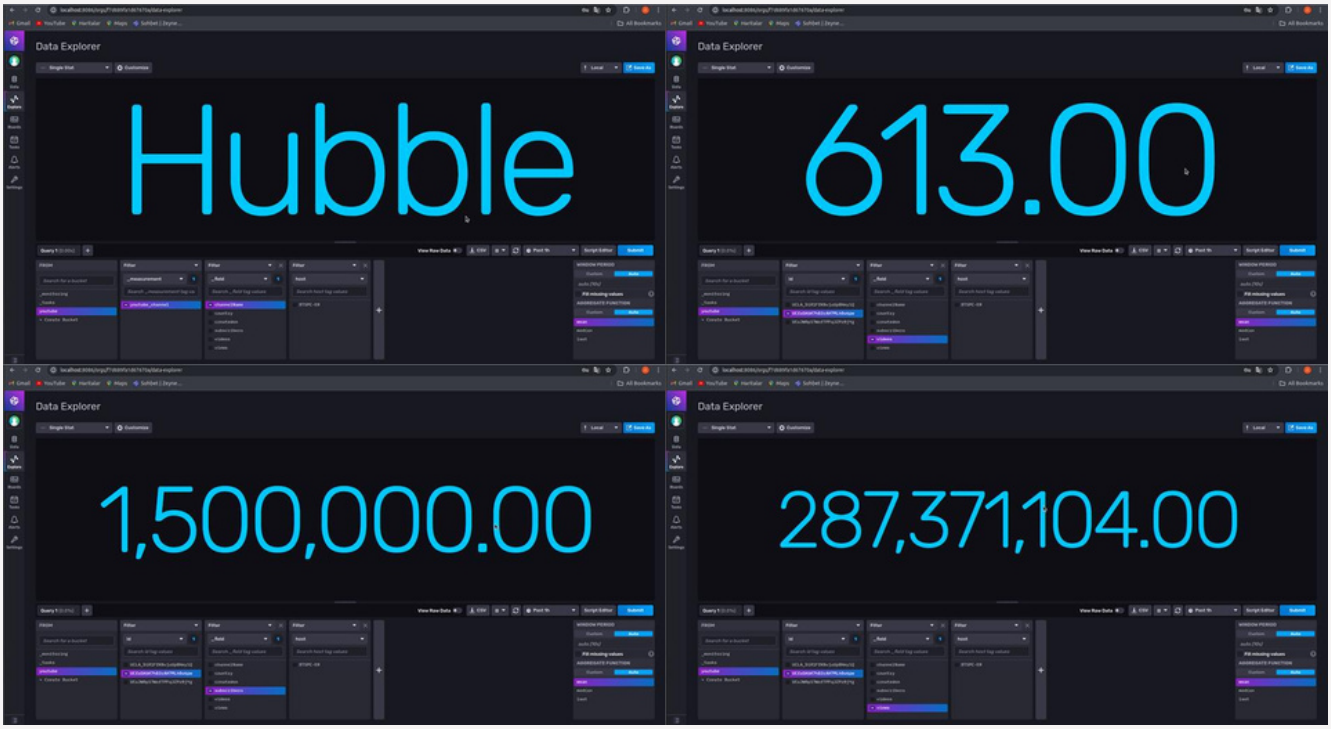


11.Complete Project

- **NASA**



- **Hubble**



- **SpaceX**