



SEKOLAH TINGGI TEKNOLOGI TERPADU NURUL FIKRI

**VISUALISASI DATA GEMPA BUMI DI DUNIA DALAM IMPLEMENTASI BIG
DATA DAN CLOUD COMPUTING**

**TUGAS PROYEK PERANGKAT LUNAK
KELOMPOK 2**

Disusun Oleh :

Ihsanul Fikri Abiyyu	0110217034
Muhammad Azhar Rasyad	0110217029
Muhammad Adil Nashrul Haq	0110217018
Triyas Tono	0110217022

**PROGRAM STUDI TEKNIK INFORMATIKA
DEPOK
JULI 2020**

a. Pendahuluan

Gempa bumi merupakan bencana alam yang dapat mengakibatkan banyak kerusakan, mulai dari bangunan yang roboh, tanah longsor, pohon tumbang, jalanan retak, hingga tanah terbelah. Dampak tersebut tentu merupakan hal yang negatif dan jika sering terjadi bencana tersebut maka akan semakin banyak kerusakan. Hal tersebut membuat kami berpikir bagaimana cara meminimalisir dampak yang terjadi akibat gempa bumi.

Salah satu cara yang kami lakukan yaitu menganalisis data-data gempa bumi yang sebelumnya pernah terjadi, sehingga data tersebut dapat kami gunakan dalam memberikan informasi apa saja yang dapat meredam kerusakan akibat bencana tersebut. Oleh karena itu penelitian ini yang kami beri judul “VISUALISASI DATA GEMPA BUMI DI DUNIA DALAM IMPLEMENTASI BIG DATA DAN CLOUD COMPUTING”, semoga dapat memberikan manfaat dalam menanggulangi dampak gempa bumi.

b. Tujuan dan Manfaat

Tujuan serta manfaat yang *InshaAllah* kami dapat berikan yaitu:

1. Memvisualisasikan data gempa bumi menjadi sebuah grafik agar mudah dipahami
2. Memberikan informasi dari data gempa bumi
3. Memberikan kesimpulan dari informasi yang telah didapat agar diimplementasikan

c. Alat dan Bahan

Alat yang digunakan pada penelitian ini yaitu:

1. Laptop
2. Internet
3. Google Drive
4. Google Document
5. Google Spreadsheet
6. Google Presentation
7. Google Colab
8. Trello
9. Sistem Operasi Linux

10. Elasticsearch
11. Kibana
12. Hadoop
13. Hive
14. Python

Adapun bahan yang digunakan dalam rentang waktu 22 Mei 2020 hingga 21 Juni 2020 yaitu:

1. https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/all_month.csv
2. https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/all_month.geojson

d. Rancangan pekerjaan/sistem

Berdasarkan penjelasan pada bagian di atas, rancangan pekerjaan yang akan dilakukan terdiri dari beberapa bagian yaitu :

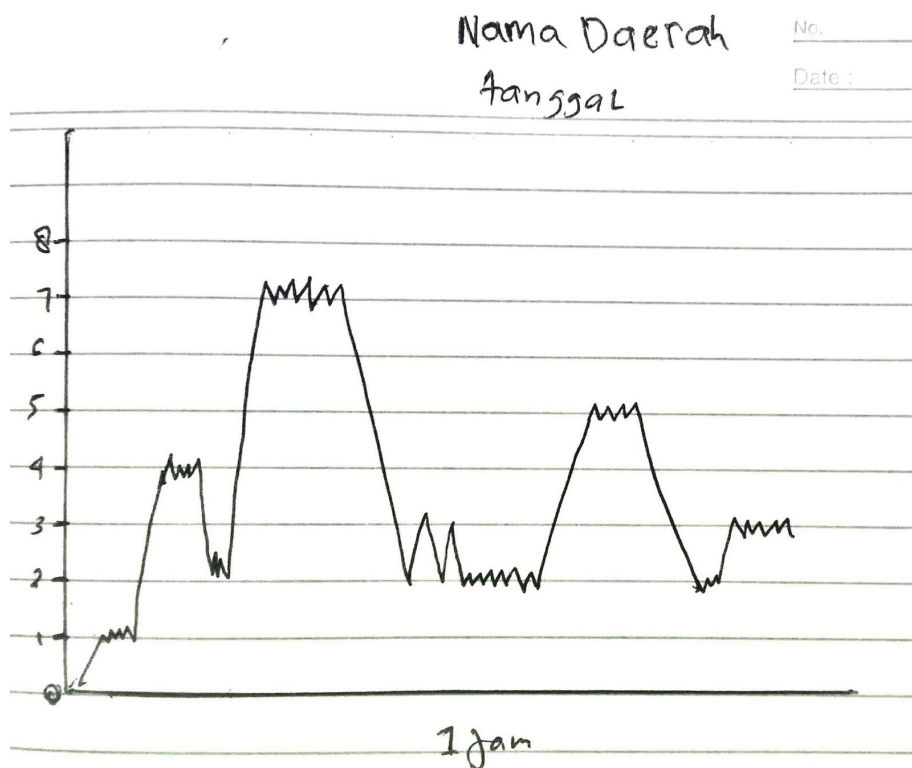
1. Membuat proposal tugas proyek perangkat lunak
2. Mencari data-data gempa bumi
3. Menginstalasi hadoop
4. Menginstalasi kibana
5. Mencari insight dari data-data gempa bumi
6. Membuat rancangan visualisasi data-data gempa bumi
7. Mengelola data-data gempa bumi dengan hadoop
8. Menampilkan data-data gempa bumi dengan kibana
9. Membuat kesimpulan dari insight data-data gempa bumi
10. Konfigurasi elasticsearch dan kibana
11. Mencari ide bisnis dari data gempa bumi
12. Mengintegrasikan hadoop dan kibana
13. Membuat dokumen dari insight gempa bumi
14. Menjelaskan tampilan data gempa bumi yang ada di kibana
15. Revisi tampilan data-data gempa bumi dengan kibana
16. Mengimplementasi design pattern pada data gempa bumi dengan bahasa pemrograman python
17. Membuat laporan hasil kerja

18. Membuat presentasi hasil kerja

e. Jadwal Penelitian

Pengerjaan	Februari				Maret				April				Mei				Juni				Juli			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Persiapan																								
Perencanaan																								
Implementasi																								
Review																								

f. Perancangan Tampilan



Gambar 1. Perancangan Tampilan 1



Gambar 2. Perancangan Tampilan 2

g. *Exploratory Data Analysis*

Berdasarkan kedua bahan sebelumnya, bahan kedua digunakan khusus memvisualisasikan data gempa bumi dalam bentuk peta sedangkan bahan pertama didapatkan beberapa kolom berikut untuk dimanfaatkan sebagai data gempa bumi diantaranya:

No	Kolom	Fungsi
1	time	Waktu peristiwa terjadi. Waktu dilaporkan dalam milidetik sejak saat (1970-01-01T00: 00: 00.000Z), dan tidak termasuk detik kabisat. Dalam format output tertentu, tanggal diformat agar dapat dibaca.
2	latitude	Derajat lintang desimal. Nilai negatif untuk lintang selatan.
3	longitude	Derajat bujur desimal. Nilai negatif untuk bujur barat.
4	depth	Tingkat kedalaman dari suatu peristiwa dalam kilometers.
5	mag	Besarnya magnitudo dari suatu peristiwa.
6	magType	Metode atau algoritma yang digunakan untuk menghitung besarnya preferensi magnitudo dari suatu peristiwa tersebut.
7	nst	Jumlah stasiun seismik yang digunakan untuk menentukan lokasi gempa.
8	gap	Kesenjangan azimuth terbesar antara stasiun yang berdekatan dengan azimuth (dalam derajat). Pada umumnya, semakin kecil nilainya, semakin akurat pula untuk perhitungan posisi horizontal

		dari gempa yang dihitung. Lokasi gempa yang dimana kesenjangan azimuthnya melebihi 180 derajat biasanya memiliki ketidakpastian besar dan kedalaman lokasinya.
9	dmin	Jarak horizontal dari pusat gempa ke stasiun terdekat (dalam derajat). 1 derajat sama dengan 111.2 kilometer. Pada umumnya, semakin kecil nilainya, semakin akurat pula perhitungan kedalaman gempanya.
10	rms	Root-mean-square (RMS) sisa waktu perjalanan, dalam detik, menggunakan semua bobot. Parameter ini menyediakan kesesuaian ukuran antara waktu kedatangan yang telah diamati dengan waktu kedatangan yang telah diprediksi untuk lokasi ini. Semakin kecil angkanya maka semakin lebih baik pula kesesuaian datanya. Nilai ini tergantung pada keakuratan model kecepatan yang digunakan untuk menghitung lokasi gempa, bobot kualitas yang ditetapkan untuk data waktu kedatangan, dan prosedur yang digunakan untuk menemukan lokasi gempa.
11	net	ID kontributor data. Mengidentifikasi jaringan yang dianggap sebagai sumber informasi pilihan untuk suatu peristiwa.
12	id	Pengidentifikasi unik untuk suatu peristiwa. Ini adalah id yang dipilih saat ini untuk peristiwa tersebut, dan dapat berubah seiring waktu.
13	updated	Waktu ketika peristiwa terakhir diperbarui. Waktu dilaporkan dalam milidetik sejak saat (1970-01-01T00: 00: 00.000Z). Dalam format output tertentu, tanggal diformat untuk dibaca.
14	place	Deskripsi tekstual dari suatu wilayah geografis bernama di dekat peristiwa. Ini mungkin nama kota, atau nama wilayah.
15	type	Daftar jenis produk yang dipisahkan koma yang terkait dengan peristiwa ini.
16	horizontalError	Ketidakpastian laporan lokasi dari sebuah kejadian, dalam kilometer.
17	depthError	Ketidakpastian laporan tingkat kedalaman dari sebuah kejadian, dalam kilometer.
18	magError	Ketidakpastian laporan besarnya magnitudo dari sebuah kejadian, dalam kilometer.
19	magNst	Jumlah dari stasiun seismic yang digunakan untuk menghitung besarnya magnitudo untuk suatu gempa.
20	status	untuk menunjukkan apakah suatu peristiwa telah ditinjau oleh manusia.

21	locationSource	Laporan jaringan lokasi yang semula diprediksi dari suatu peristiwa.
22	magSource	Laporan jaringan besarnya magnitude yang semula diprediksi dari suatu peristiwa.

Tabel 1. *Exploratory Data Analysis*

h. Implementasi

Ada 2 implementasi dalam penelitian ini yaitu:

1. Implementasi dengan Hadoop, Hive, Elasticsearch, dan Kibana
 - a. Menyalakan terlebih dahulu server hadoop agar dapat menggunakan hive

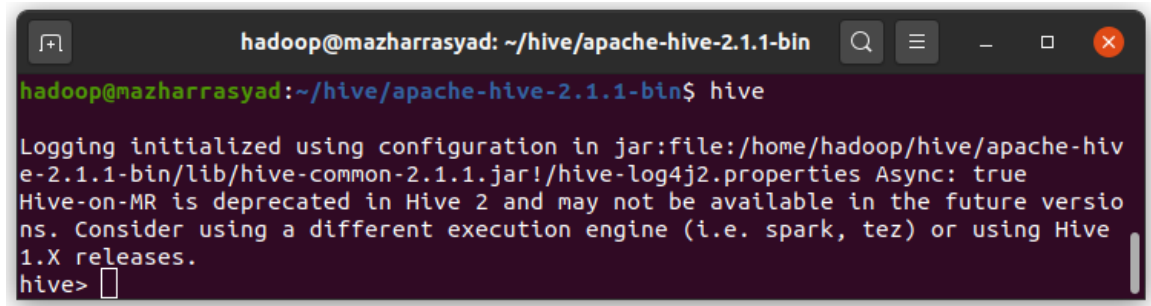
```

hadoop@mazharrasyad: ~
hadoop@mazharrasyad:~$ ./hadoop/sbin/start-dfs.sh
20/06/19 19:42:47 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
Starting namenodes on [localhost]
localhost: starting namenode, logging to /home/hadoop/hadoop/logs/hadoop-hadoop-
namenode-mazharrasyad.out
localhost: starting datanode, logging to /home/hadoop/hadoop/logs/hadoop-hadoop-
datanode-mazharrasyad.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /home/hadoop/hadoop/logs/hadoop-
hadoop-secondarynamenode-mazharrasyad.out
20/06/19 19:43:24 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
hadoop@mazharrasyad:~$ ./hadoop/sbin/start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /home/hadoop/hadoop/logs/yarn-hadoop-resour
cemanager-mazharrasyad.out
localhost: starting nodemanager, logging to /home/hadoop/hadoop/logs/yarn-hadoop-
nodemanager-mazharrasyad.out
hadoop@mazharrasyad:~$ jps
4544 NodeManager
4586 Jps
3531 NameNode
3901 SecondaryNameNode
4398 ResourceManager
3678 DataNode
hadoop@mazharrasyad:~$

```

Gambar 3. *Configuration Server Hadoop*

- b. Selanjutnya masuk ke dalam hive untuk menggunakan bahannya



```
hadoop@mazharrasyad: ~/hive/apache-hive-2.1.1-bin
hadoop@mazharrasyad:~/hive/apache-hive-2.1.1-bin$ hive

Logging initialized using configuration in jar:file:/home/hadoop/hive/apache-hiv
e-2.1.1-bin/lib/hive-common-2.1.1.jar!/hive-log4j2.properties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versio
ns. Consider using a different execution engine (i.e. spark, tez) or using Hive
1.X releases.
hive> 
```

Gambar 4. *Start Hive*

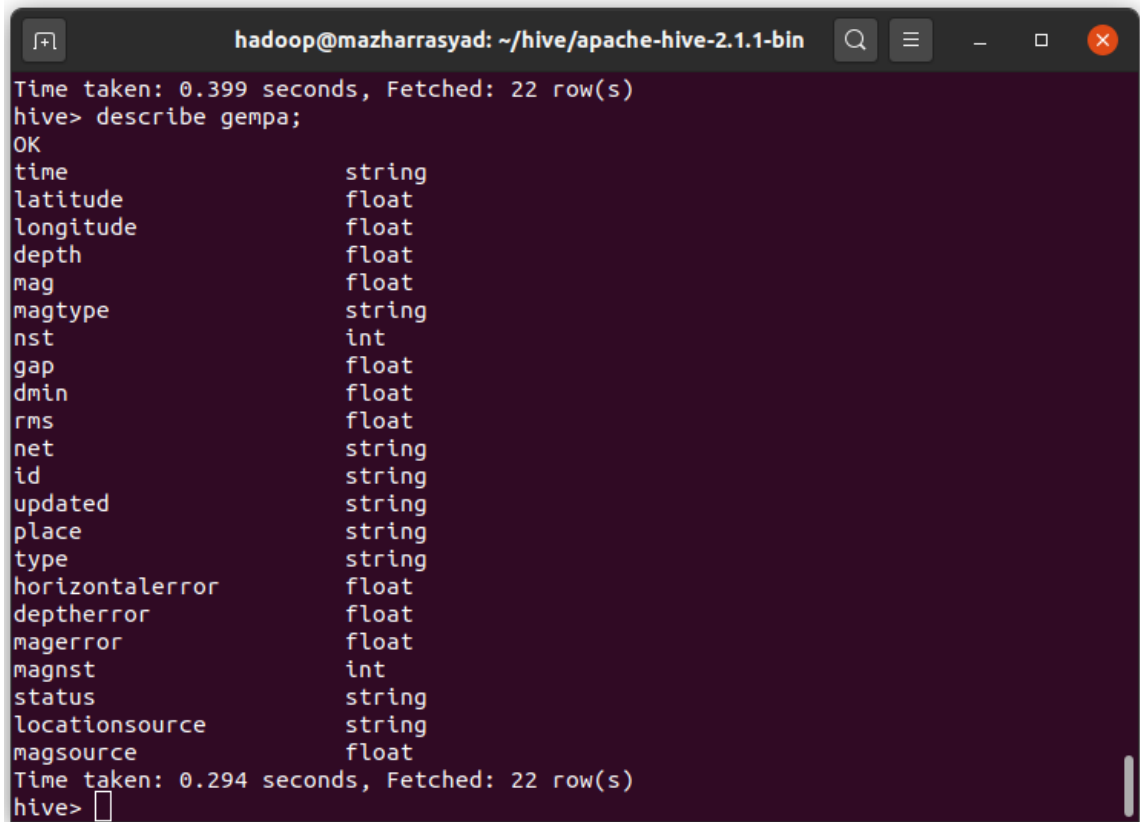
- c. Hive digunakan untuk menyimpan file data gempa bumi dan terlihat dari gambar di bawah bahwa database yang digunakan yaitu “muhasabah” dan terdapat 2 tabel yaitu gempa dan gempa_es, seperti berikut:



```
hadoop@mazharrasyad: ~/hive/apache-hive-2.1.1-bin
hive> show databases;
OK
default
muhasabah
Time taken: 4.622 seconds, Fetched: 2 row(s)
hive> use muhasabah;
OK
Time taken: 0.082 seconds
hive> show tables;
OK
gempa
gempa_es
Time taken: 0.285 seconds, Fetched: 2 row(s)
hive> 
```

Gambar 5. *Check Data Gempa Bumi*

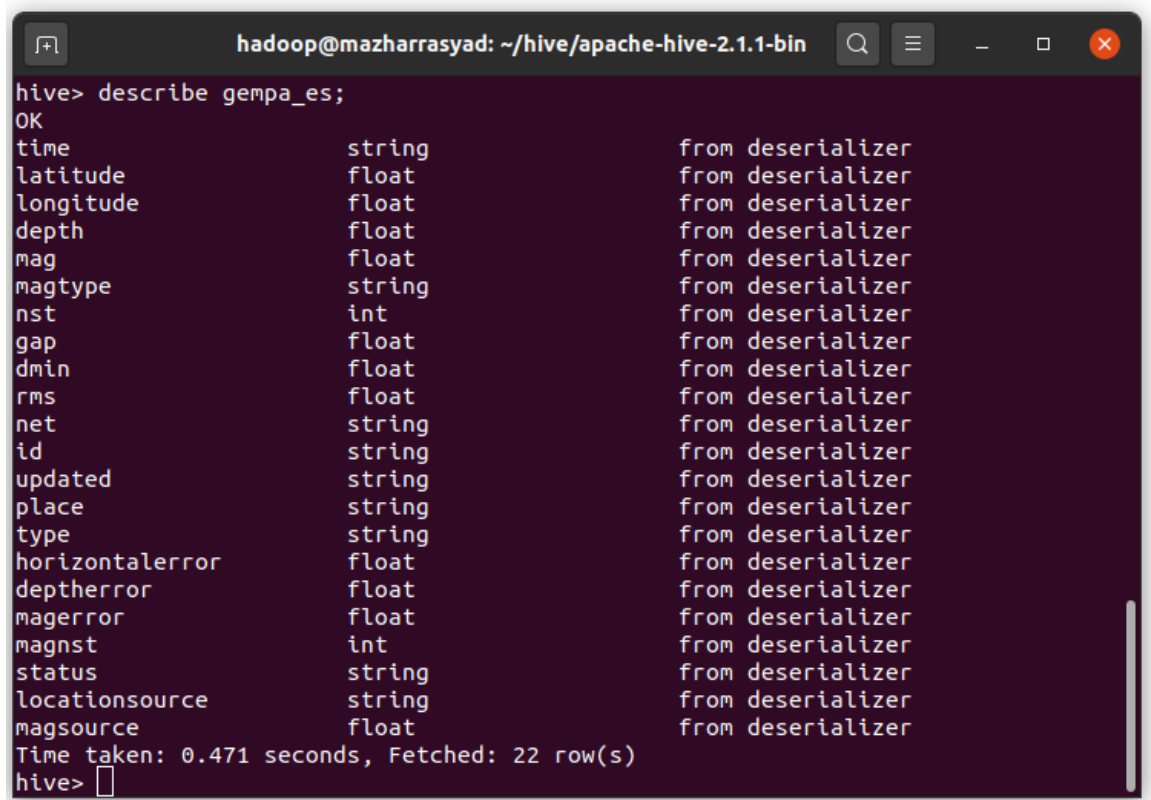
- d. Tabel gempa merupakan data asli yang pertama diimport dapat dilihat pada gambar berikut:



```
hadoop@mazharrasyad: ~/hive/apache-hive-2.1.1-bin
Time taken: 0.399 seconds, Fetched: 22 row(s)
hive> describe gempa;
OK
time                string
latitude            float
longitude            float
depth               float
mag                 float
magtype             string
nst                 int
gap                 float
dmin                float
rms                 float
net                 string
id                  string
updated             string
place               string
type                string
horizontalerror      float
deptherror           float
magerror             float
magnst              int
status              string
locationsource       string
magsource            float
Time taken: 0.294 seconds, Fetched: 22 row(s)
hive>
```

Gambar 6. *Describe Table* gempa

- e. Sedangkan tabel gempa_es merupakan hasil tabel gempa yang diindeks ke elasticsearch seperti berikut:



```
hadoop@mazharrasyad: ~/hive/apache-hive-2.1.1-bin
hive> describe gempa_es;
OK
time                string                from deserializer
latitude            float                from deserializer
longitude            float                from deserializer
depth               float                from deserializer
mag                 float                from deserializer
magtype             string              from deserializer
nst                 int                 from deserializer
gap                 float                from deserializer
dmin                float                from deserializer
rms                 float                from deserializer
net                 string              from deserializer
id                  string              from deserializer
updated             string              from deserializer
place               string              from deserializer
type                string              from deserializer
horizontalerror      float                from deserializer
deptherror           float                from deserializer
magerror             float                from deserializer
magnst              int                 from deserializer
status               string              from deserializer
locationsource        string              from deserializer
magsource            float                from deserializer
Time taken: 0.471 seconds, Fetched: 22 row(s)
hive>
```

Gambar 7. Describe Table gempa_es

Isi dari kedua tabel di atas tidak ditampilkan karena banyaknya kolom serta baris yang ada dan berikut perintah yang digunakan pada langkah b sampai e:

```
create database muhasabah;
```

```
use muhasabah;
```

```
create table gempa(  
  `time` string,  
  latitude float,  
  longitude float,  
  depth float,  
  mag float,  
  magType string,  
  nst int,  
  gap float,  
  dmin float,
```

```
rms float,  
net string,  
id string,  
updated string,  
place string,  
type string,  
horizontalError float,  
depthError float,  
magError float,  
magNst int,  
status string,  
locationSource string,  
`magSource` string)  
row format delimited  
fields terminated by ','  
tblproperties ("skip.header.line.count"="1");
```

```
load data local inpath '/home/hadoop/earthquake21jun.csv' overwrite into table  
gempa;
```

```
add jar  
/home/hadoop/elasticsearch-hadoop-7.7.0/dist/elasticsearch-hadoop-7.7.0.jar;
```

```
create external table gempa_es(  
`time` string,  
latitude float,  
longitude float,  
depth float,  
mag float,  
magType string,  
nst int,  
gap float,  
dmin float,  
rms float,  
net string,  
id string,  
updated string,  
place string,
```

```

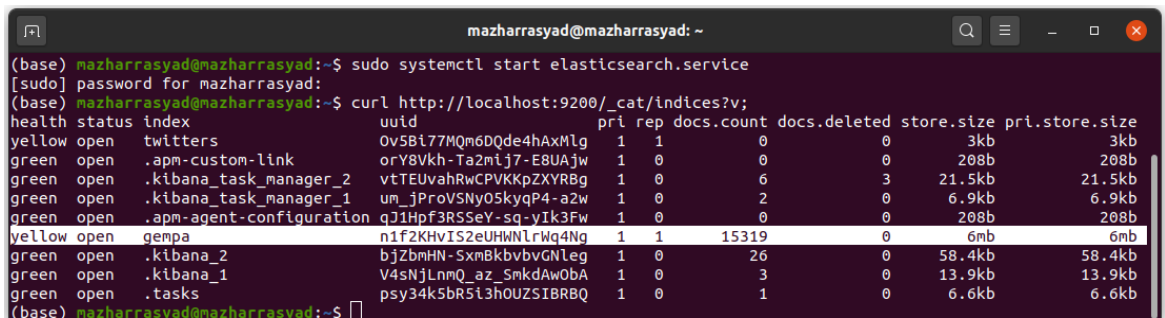
type string,
horizontalError float,
depthError float,
magError float,
magNst int,
status string,
locationSource string,
`magSource` string)
STORED BY 'org.elasticsearch.hadoop.hive.EsStorageHandler'
TBLPROPERTIES('es.resource'='gempa/gempa_es');

INSERT OVERWRITE TABLE gempa_es SELECT * FROM gempa;

curl http://localhost:9200/_cat/indices?v;

```

- f. Kemudian, menyalakan server elasticsearch apakah sudah ada indeks tersebut atau belum:



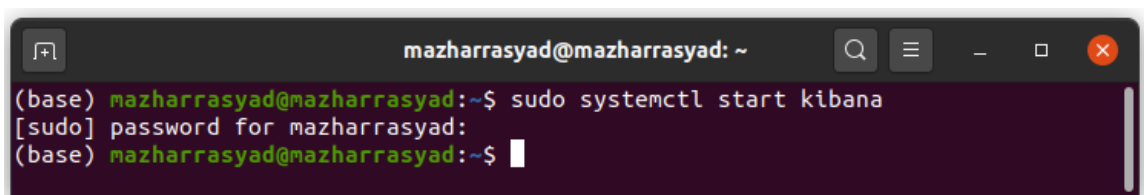
```

mazharrasyad@mazharrasyad: ~
(base) mazharrasyad@mazharrasyad:~$ sudo systemctl start elasticsearch.service
[sudo] password for mazharrasyad:
(base) mazharrasyad@mazharrasyad:~$ curl http://localhost:9200/_cat/indices?v;
health status index      uuid                                pri rep docs.count docs.deleted store.size pri.store.size
yellow open   twitters      0v5Bi77MQm6DQde4hAxMlg            1   1         0             0          3kb             3kb
green  open   .apm-custom-link orY8Vkh-Ta2mij7-E8UAJw            1   0         0             0         208b            208b
green  open   .kibana_task_manager_2 vtTEUvahRwCPVKkpZXyRBg            1   0         6             3        21.5kb           21.5kb
green  open   .kibana_task_manager_1 um_jProVSny05kyqP4-a2w            1   0         2             0         6.9kb            6.9kb
green  open   .apm-agent-configuration qJ1Hpf3RSsEY-sq-yIk3Fw            1   0         0             0         208b            208b
yellow open   gempa        n1f2KHvIS2eUHWNLrWq4Ng            1   1       15319           0          6mb             6mb
green  open   .kibana_2     bJZbmHN-SxmBkbvbnvGNleg            1   0         26             0         58.4kb           58.4kb
green  open   .kibana_1     V4sNjLnmQ_az_SmkdAw0bA            1   0         3             0         13.9kb           13.9kb
green  open   .tasks       psy34k5bR5i3hOUZSIBRBQ            1   0         1             0          6.6kb            6.6kb
(base) mazharrasyad@mazharrasyad:~$

```

Gambar 8. Configuration Server Elasticsearch

- g. Setelah diketahui bahwa indeks pada elasticsearch telah terbuat maka selanjutnya menyalakan server kibana untuk memvisualisasikan data tersebut



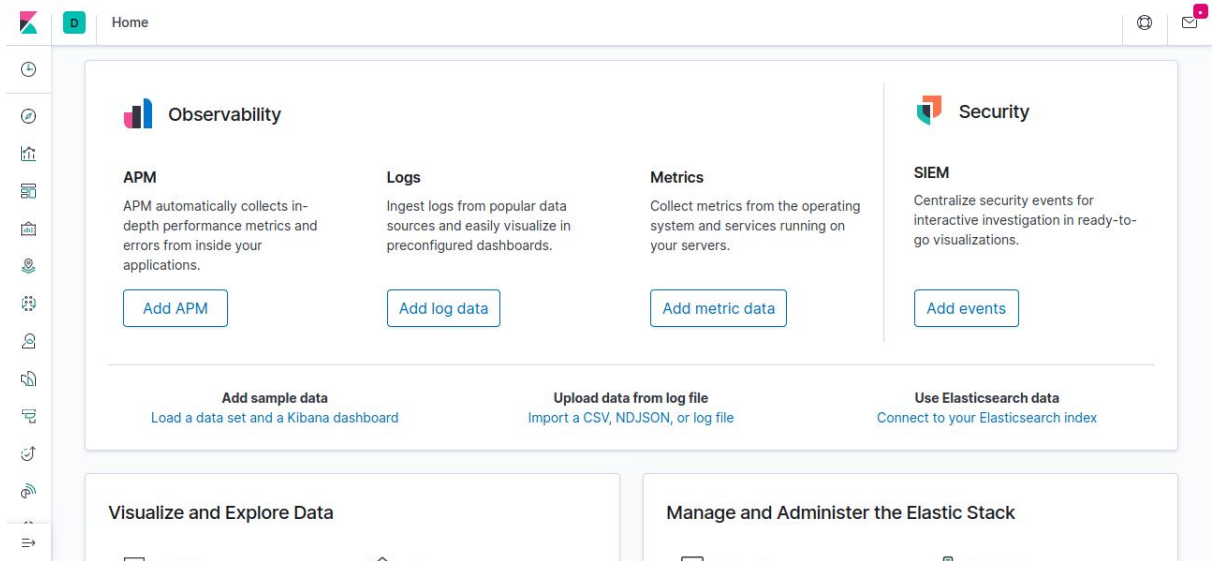
```

mazharrasyad@mazharrasyad: ~
(base) mazharrasyad@mazharrasyad:~$ sudo systemctl start kibana
[sudo] password for mazharrasyad:
(base) mazharrasyad@mazharrasyad:~$

```

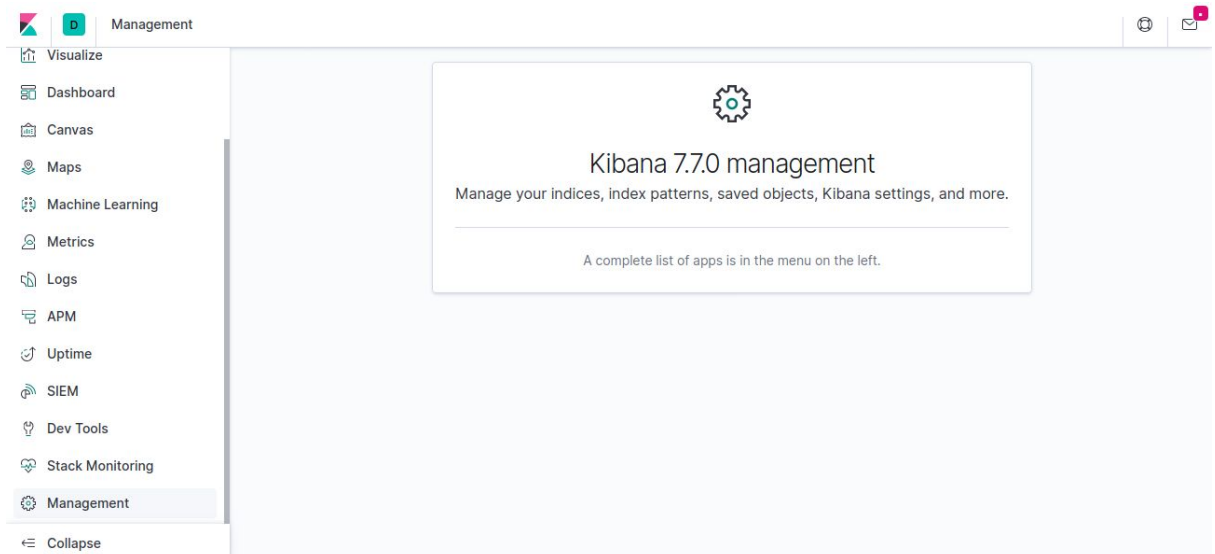
Gambar 9. Configuration Server Kibana

- h. Setelah server kibana berhasil dinyalakan maka dijalankan menggunakan browser dengan url `http://localhost:5601` seperti berikut:



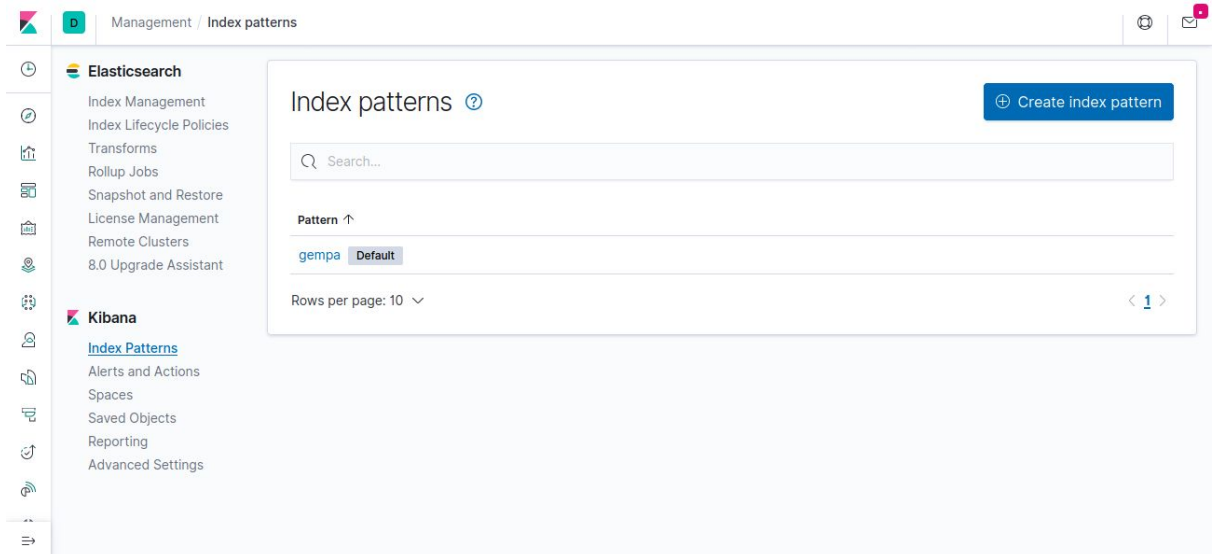
Gambar 10. *Dashboard* Kibana

- i. Pada *dashboard* kibana terdapat *sidebar* di sebelah kiri untuk konfigurasi dengan elasticsearch, karena sebelumnya hive telah dimasukkan ke dalam indeks elasticsearch namun belum dimasukkan ke dalam kibana. Oleh karena itu perlu dilakukan konfigurasi melalui menu *Management* di *sidebar* kibana, seperti berikut:



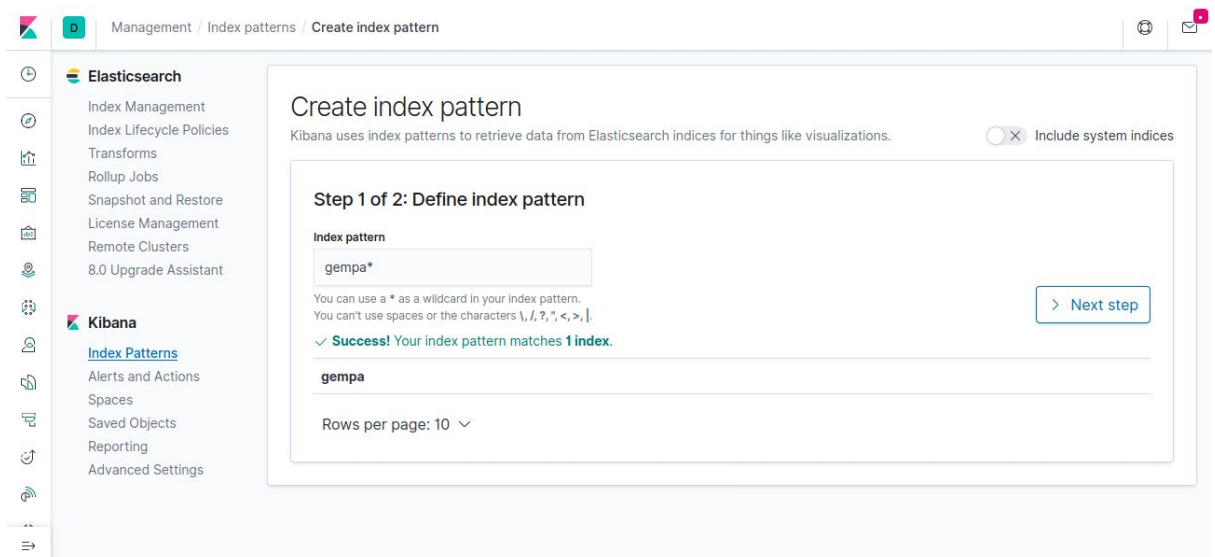
Gambar 11. *Management* Kibana

- j. Pada menu *Management* terdapat banyak opsi, untuk kali ini kami menggunakan *Index Patterns* supaya dikaitkan indeks sebelumnya dan sudah terlihat Pattern gempa karena sebelumnya kami sudah membuatnya:



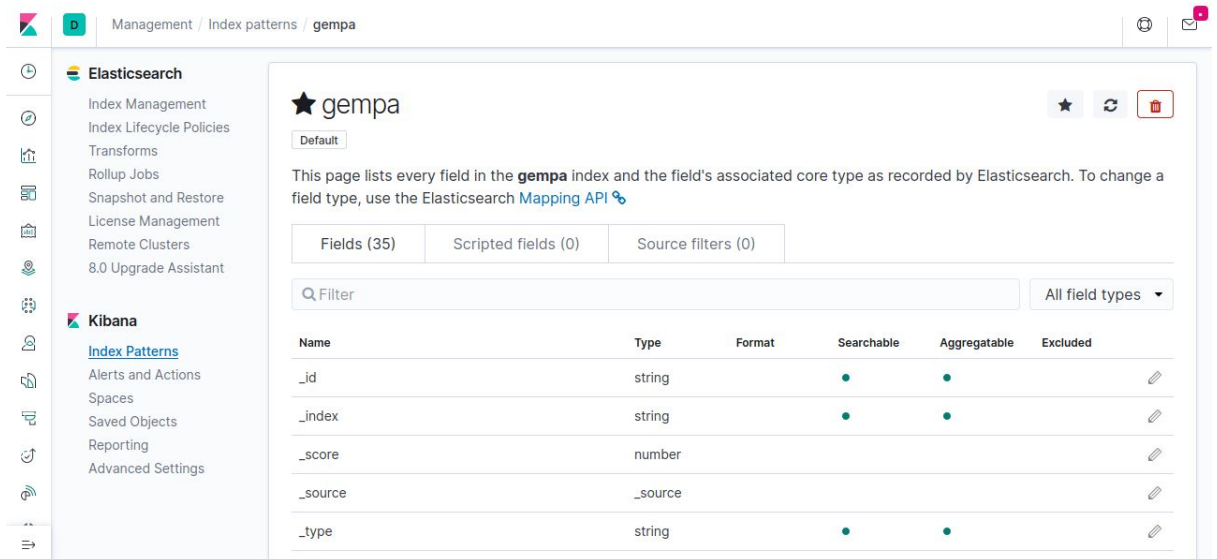
Gambar 12. *Index Patterns* Kibana

- k. Jika kami ingin membuat indeks lagi cukup dengan menggunakan button *Create index pattern* sehingga muncul tampilan berikut:



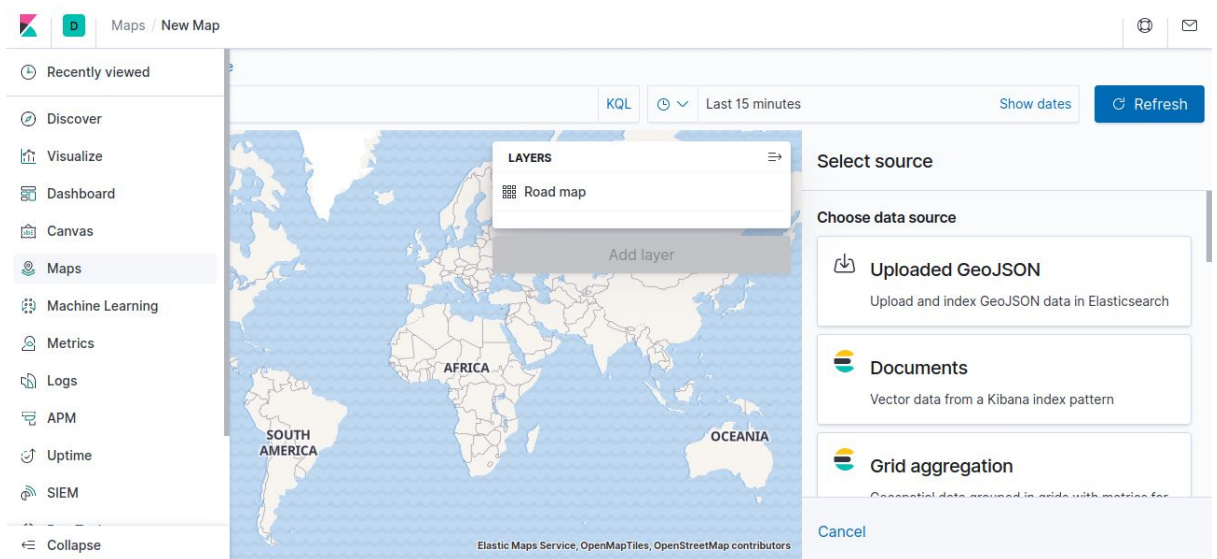
Gambar 13. *Create Index Pattern*

- l. Setelah terbuat indeks pada kibana, dapat dilihat isi indeks gempa tersebut seperti berikut:



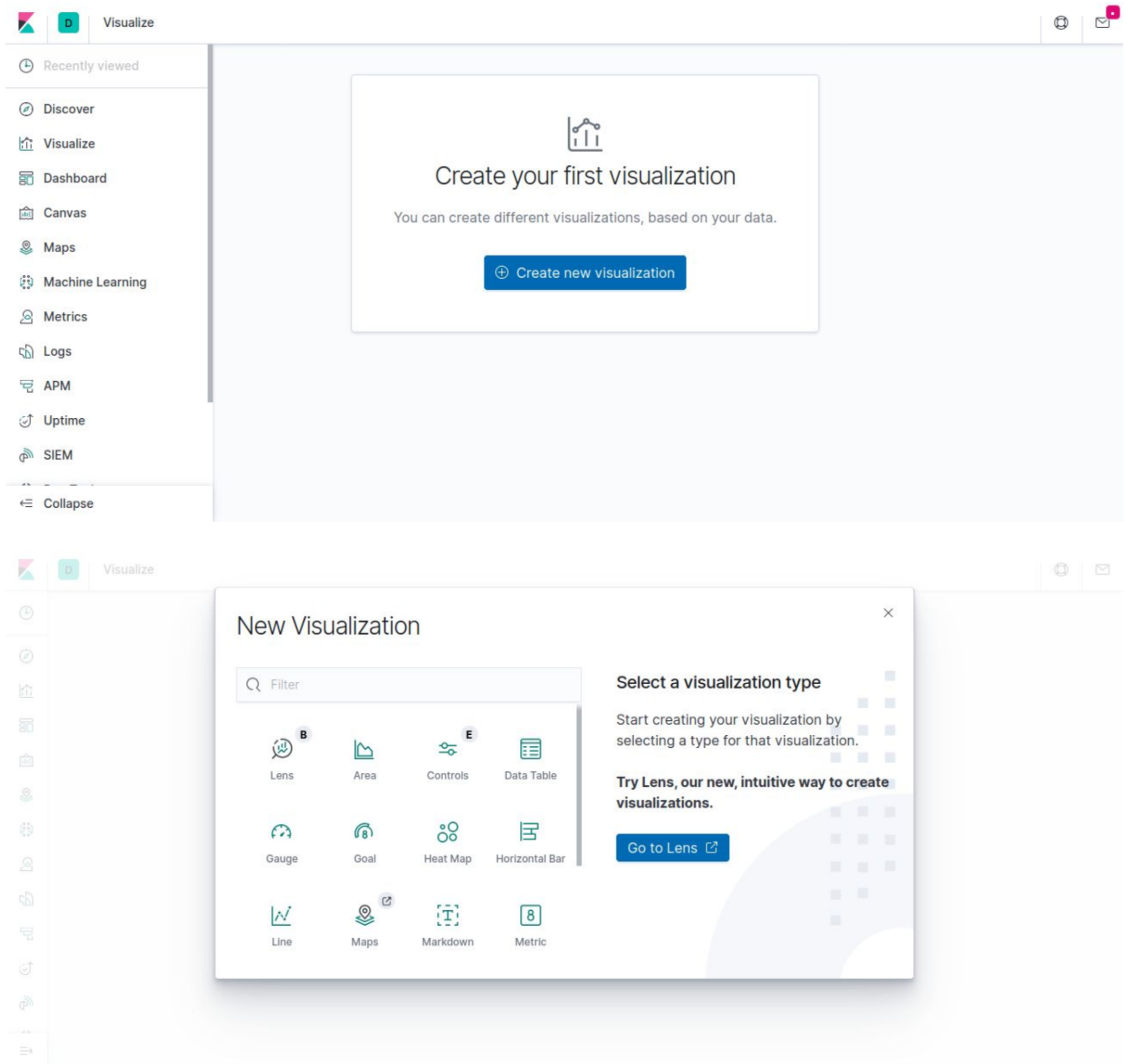
Gambar 14. Indeks gempa

- m. Setelah berhasil bahan pertama dimasukkan ke dalam indeks maka dilanjutkan dengan bahan kedua yaitu .geojson untuk memvisualisasikan dalam bentuk peta. Caranya dengan menggunakan menu Maps kemudian *upload* file bahan kedua sebelumnya, seperti berikut:



Gambar 15. Upload File geojson

- n. Tahap akhir yaitu membuat visualisasi datanya dengan menu *Visualize* dengan fitur berikut dan hasilnya dapat dilihat pada bagian Visualisasi Data selanjutnya



Gambar 16. *Visualize Kibana*

2. Implementasi dengan Google Colab dan Python

```
%matplotlib inline
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Abstract Factory method dan inisiasi data yang digunakan

```
class abstractData:
    def show_data(self): pass
```



```

def delete_data(self) :pass

class earthquake_data(abstractData):
    __shared_state = dict()
    # constructor method
    def __init__(self, tableData):
        self.__dict__ = self.__shared_state
        self.tableData = tableData

    def show_data(self):
        return self.tableData

    def delete_data(self):
        self.tableData = {}
        print("Data is gone")

read_data =
pd.read_csv("https://earthquake.usgs.gov/earthquakes/feed/v1.0/s
ummary/all_month.csv")
earthquake = earthquake_data(read_data)

# earthquake2.delete_data()
earthquake_show = earthquake.show_data()
earthquake_show

earthquake_show.plot(x="longitude", y="latitude", c="mag",
kind="scatter", figsize=(10,6));

```

Builder pattern untuk membuat plotting data

```

class plotting:
    def make_plot(self): pass

    def add_plot_data(self): pass

    def set_plot(self): pass

    def show_plot(self): pass

class scatter_plot(plotting):
    def __init__(self, data):
        self.data = data

```

```

        self.make_plot()
        self.add_plot_data()
        self.set_data()

    def make_plot(self):
        self.fig, self.ax = plt.subplots(figsize=(20,10))

    def add_plot_data(self):
        self.scatter = self.ax.scatter(x=self.data["longitude"],
                                       y=self.data["latitude"],
                                       c=self.data["mag"],
                                       cmap="Reds")

    def set_data(self):
        self.ax.set(title="Longitude, latitude, and mag",
                    xlabel="longitude",
                    ylabel="latitude");

        self.ax.legend(*self.scatter.legend_elements(),
title="magnitude");

    def show_plot(self):
        return self.fig

scatter = scatter_plot(earthquake_show)

earthquake_show.head()

class histogram_plot(plotting):
    def __init__(self, data):
        self.data = data
        self.make_plot()
        self.add_plot_data()
        self.set_data()
        self.show_plot()

    def make_plot(self):
        self.fig, self.ax = plt.subplots(figsize=(10,8))

    def add_plot_data(self):

```

```

        self.hist = self.ax.hist(self.data["mag"])

    def set_data(self):
        self.ax.set(title="Magnitude gempa",
                    xlabel="magnitude",
                    ylabel="frekuensi kemunculan");

    def show_plot(self):
        return self.fig

histogram = histogram_plot(earthquake_show)

earthquake_100 = earthquake_show.head(100)
earthquake_100

class plot(plotting):
    def __init__(self, data):
        self.data = data
        self.make_plot()
        self.add_plot_data()
        self.set_data()

    def make_plot(self):
        self.fig, self.ax = plt.subplots(figsize=(30,6))

    def add_plot_data(self):
        self.plot = self.ax.plot(self.data["time"],
self.data["mag"],)

    def set_data(self):
        self.ax.set(title="Earthquake Time and Magnitude",
                    xlabel="Time",
                    ylabel="Magnitude");

        plt.draw()
        self.ax.set_xticklabels(self.ax.get_xticklabels(),
rotation=45);
        self.ax.axhline(earthquake_100["mag"].mean());

    def show_plot(self):

```

```

        return self.fig

plot100 = plot(earthquake_100)

class scatter_plot2(plotting):
    def __init__(self, data):
        self.data = data
        self.make_plot()
        self.add_plot_data()
        self.set_data()

    def make_plot(self):
        self.fig, self.ax = plt.subplots(figsize=(30,6))

    def add_plot_data(self):
        self.scatter = self.ax.scatter(x=self.data["place"],
                                         y=self.data["depth"],
                                         c=self.data["mag"],
                                         cmap="winter")

    def set_data(self):
        self.ax.set(title="Earthquake and Depth",
                    xlabel="Place",
                    ylabel="Depth")

        plt.draw()
        self.ax.set_xticklabels(self.ax.get_xticklabels(),
rotation=45)

        self.ax.set_ylim([-5,150])
        self.ax.legend(*self.scatter.legend_elements(),
title="magnitude");

        self.ax.axhline(earthquake_100["depth"].mean(),
                        linestyle="--",
                        color='r');

    def show_plot(self):
        return self.fig

```

```
scatter2 = scatter_plot2(earthquake_100)
```

Chain of responsibility, digunakan untuk mencari grafik yang sesuai dengan permintaan user

```
class AbstractHandler(object):
    """Parent class of all concrete handlers"""
    def __init__(self, nxt):
        """change or increase the local variable using nxt"""
        self._nxt = nxt
    def handle(self, request):
        """It calls the processRequest through given request"""
        handled = self.processRequest(request)
        """case when it is not handled"""
        if not handled:
            self._nxt.handle(request)
    def processRequest(self, request):
        """throws a NotImplementedError"""
        raise NotImplementedError('First implement it !')

class FirstConcreteHandler(AbstractHandler):
    """Concrete Handler # 1: Child class of AbstractHandler"""
    def processRequest(self, request):
        if request == 'scatter':
            return scatter_plot(earthquake_show)

class SecondConcreteHandler(AbstractHandler):
    """Concrete Handler # 2: Child class of AbstractHandler"""
    def processRequest(self, request):
        if request == "histogram":
            return histogram_plot(earthquake_show)

class ThirdConcreteHandler(AbstractHandler):
    """Concrete Handler # 3: Child class of AbstractHandler"""
    def processRequest(self, request):
        if request == 'plot':
            return plot(earthquake_100)
```

```

class FourthConcreteHandler(AbstractHandler):
    """Concrete Handler # 3: Child class of AbstractHandler"""
    def processRequest(self, request):
        if request == 'scatter2':
            return scatter_plot2(earthquake_100)

class DefaultHandler(AbstractHandler):
    """Default Handler: child class from AbstractHandler"""
    def processRequest(self, request):
        """Gives the message that th request is not handled and
returns true"""
        print("This is {} telling you that request '{}' has no
handler right now.".format(self.__class__.__name__,
request))
        return True

class User:
    """User Class"""
    def __init__(self):
        """Provides the sequence of handles for the users"""
        initial = None
        self.handler =
FirstConcreteHandler(SecondConcreteHandler(ThirdConcreteHandler(
FourthConcreteHandler(DefaultHandler(initial))))))
    def agent(self, user_request):
        """Iterates over each request and sends them to specific
handles"""
        #         for request in user_request:
            self.handler.handle(user_request)
    """main method"""
    if __name__ == "__main__":
        """Create a client object"""
        user = User()
        """Create requests to be processed"""
        string = "plot"
        #         requests = list(string)
        """Send the requests one by one, to handlers as per the

```

```
sequence of handlers defined in the Client class"""
```

```
user.agent(string)
```

Proxy agar user tidak mengakses langsung class yang akan digunakan

```
class earthquake_proxy:
```

```
    '''Relatively less resource-intensive proxy acting as  
middleman.
```

```
    Instantiates a College object only if there is no fee  
due.'''
```

```
    def __init__(self):
```

```
        self.data = None
```

```
    def add_data(self, data):
```

```
        self.earthquake = earthquake_data(data)
```

```
    def show_data(self):
```

```
        return self.earthquake.show_data()
```

```
    def delete_data(self):
```

```
        self.earthquake.delete_data()
```

```
    def show_plot(self, plot):
```

```
        self.user = User()
```

```
        self.user.agent(plot)
```

```
# Instantiate the Proxy
```

```
proxy = earthquake_proxy()
```

```
#Add Data
```

```
data =
```

```
pd.read_csv("https://earthquake.usgs.gov/earthquakes/feed/v1.0/s  
ummary/all_month.csv")
```

```
proxy.add_data(data)
```

```
#Show Data
```

```
proxy.show_data().head()
```

```
#show plot
```

```
proxy.show_plot("scatter")
```

```
#delete data
```

```
proxy.delete_data()
proxy.show_data()

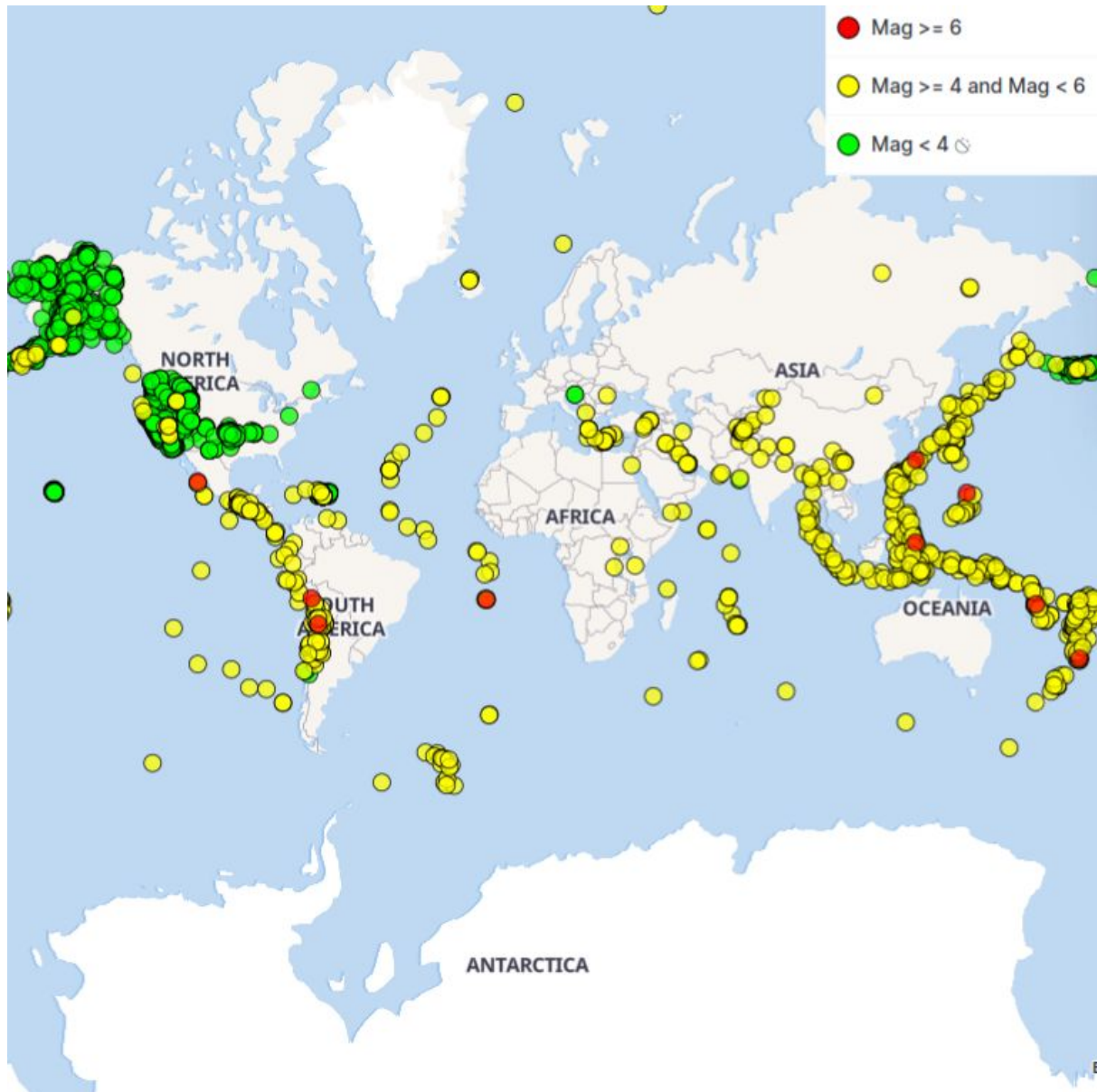
proxy.add_data =
pd.read_csv("https://earthquake.usgs.gov/earthquakes/feed/v1.0/s
ummary/all_month.csv")

proxy.show_plot("histogram")
```


i. Visualisasi Data

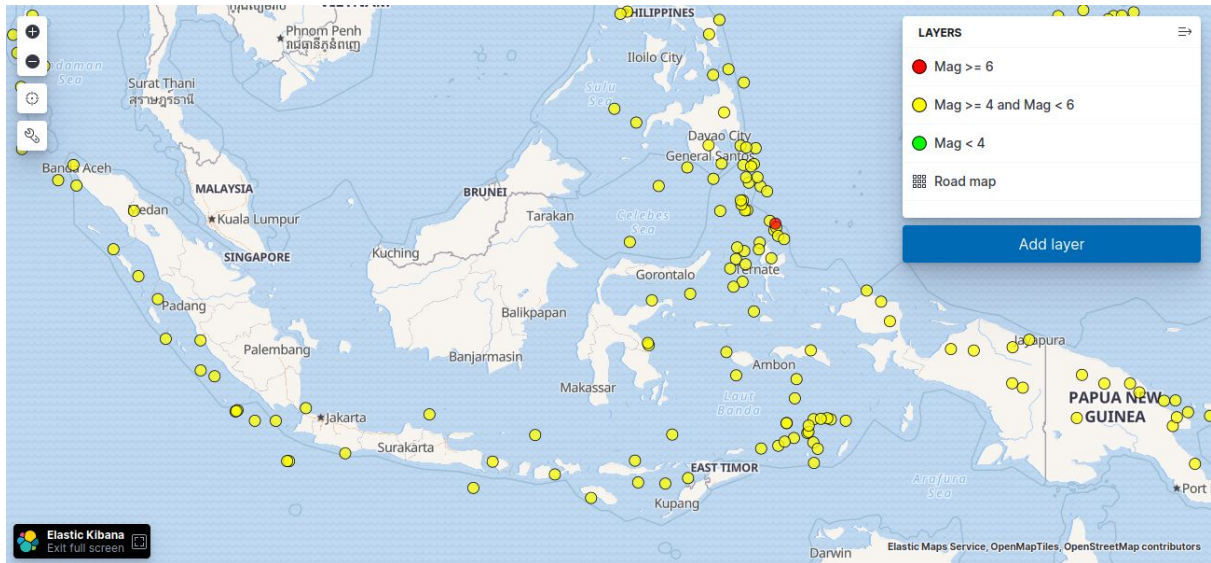
Berikut kedua hasil implementasi di atas yang dijadikan bentuk visual dengan rentang waktu 22 Mei 2020 hingga 21 Juni 2020 yaitu:

1. Tampilan Peta Dunia yang Terkena Gempa Bumi, memberikan informasi berdasarkan peta dunia dengan ketentuan magnitudo dibawah 4 ditandai dengan warna hijau, magnitudo 4 hingga dibawah 6 ditandai dengan warna kuning dan magnitudo 6 hingga lebih besar ditandai dengan warna merah.



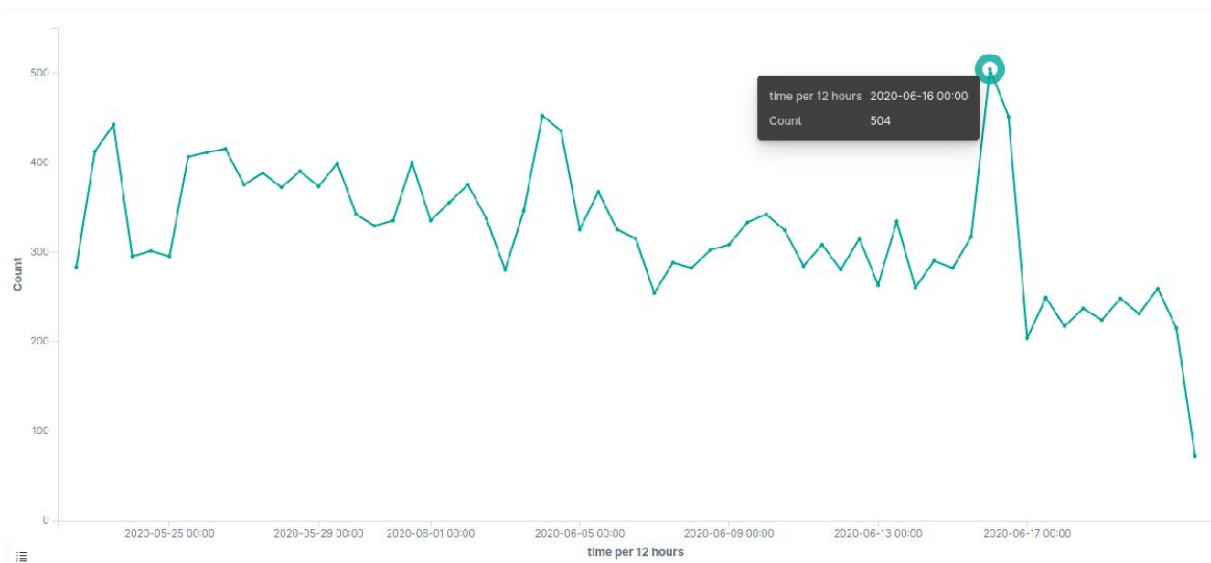
Gambar 17. Tampilan Peta Dunia yang Terkena Gempa Bumi

2. Tampilan Negara Indonesia yang Terkena Gempa Bumi, memberikan informasi dengan tampilan sebelumnya namun lebih spesifik ke negara Indonesia.



Gambar 18. Tampilan Negara Indonesia yang Terkena Gempa Bumi

3. Tampilan Jumlah Terjadinya Gempa Bumi, memberikan informasi mengenai berapa kali gempa bumi terjadi setiap 12 jam sekali.



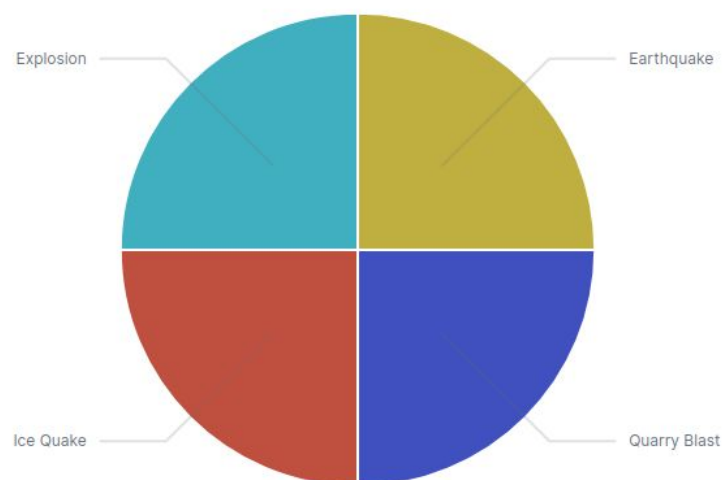
Gambar 19. Tampilan Jumlah Terjadinya Gempa Bumi

4. Tampilan Jumlah Reviewed Gempa Bumi by Human, memberikan informasi bahwa ada data gempa bumi yang secara langsung direview oleh manusia dan ada juga yang secara otomatis.



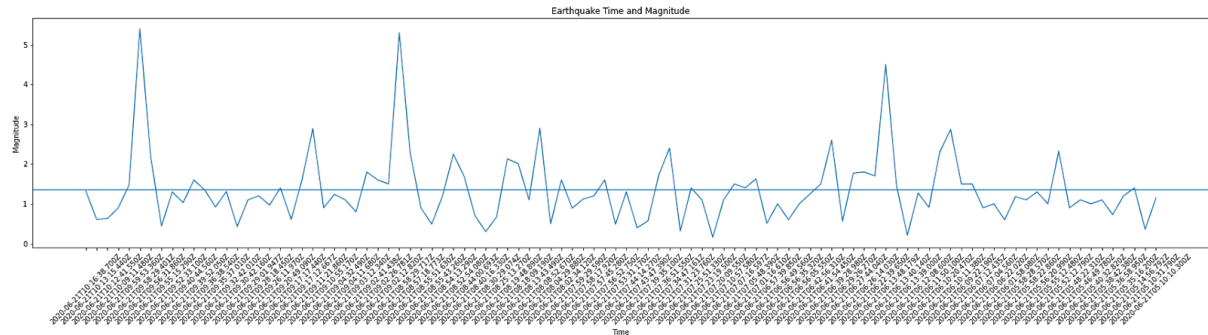
Gambar 20. Tampilan Jumlah Reviewed Gempa Bumi by Human

5. Tampilan Tipe Gempa Bumi, memberikan informasi bahwa ada empat tipe gempa bumi yaitu gempa bumi yang secara alami, *ice quake*, *explosion*, dan *quarry blast*.



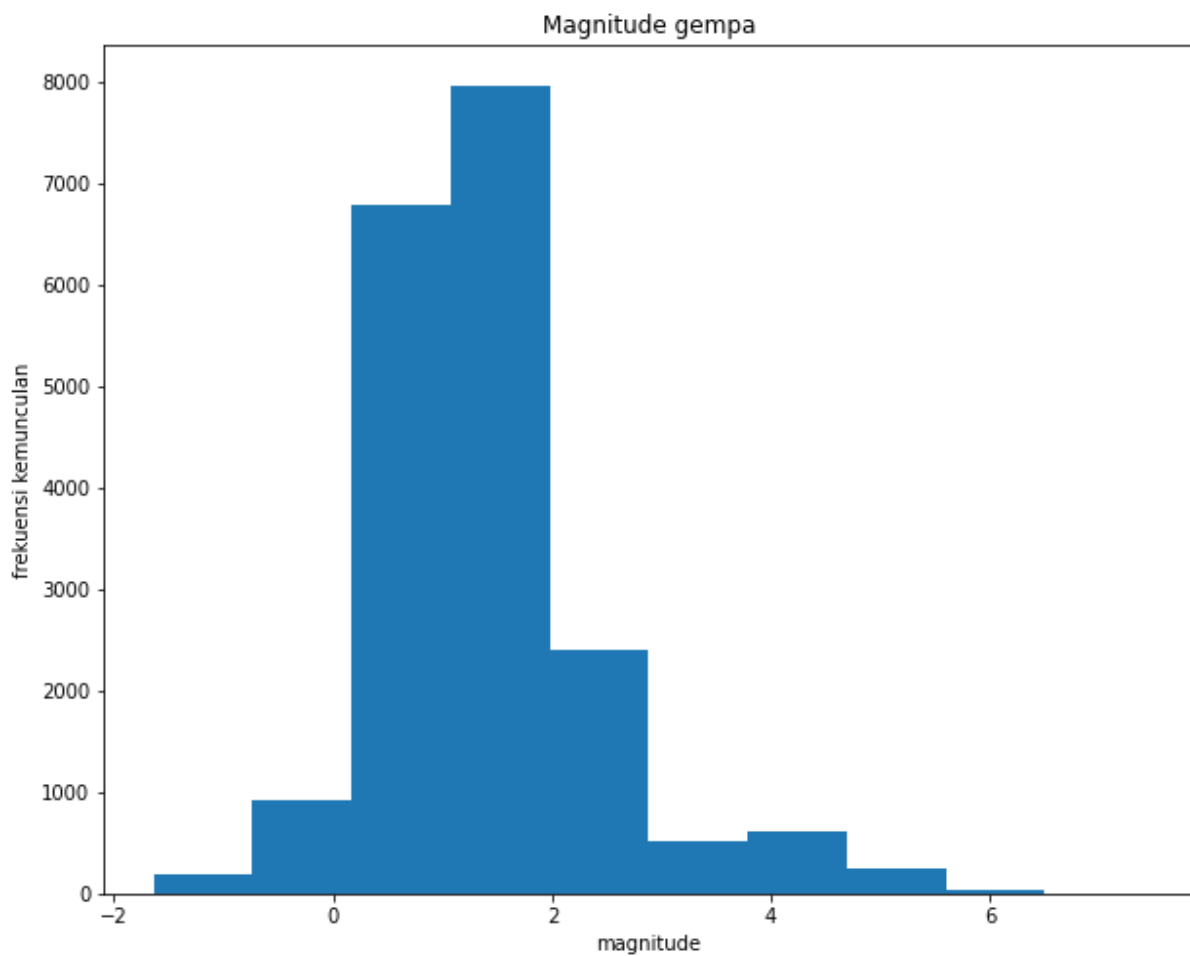
Gambar 21. Tampilan Tipe Gempa Bumi

6. Tampilan Besar Magnitudo Gempa Bumi, memberikan informasi besarnya magnitudo pada gempa bumi yang terjadi disetiap waktu tertentu.



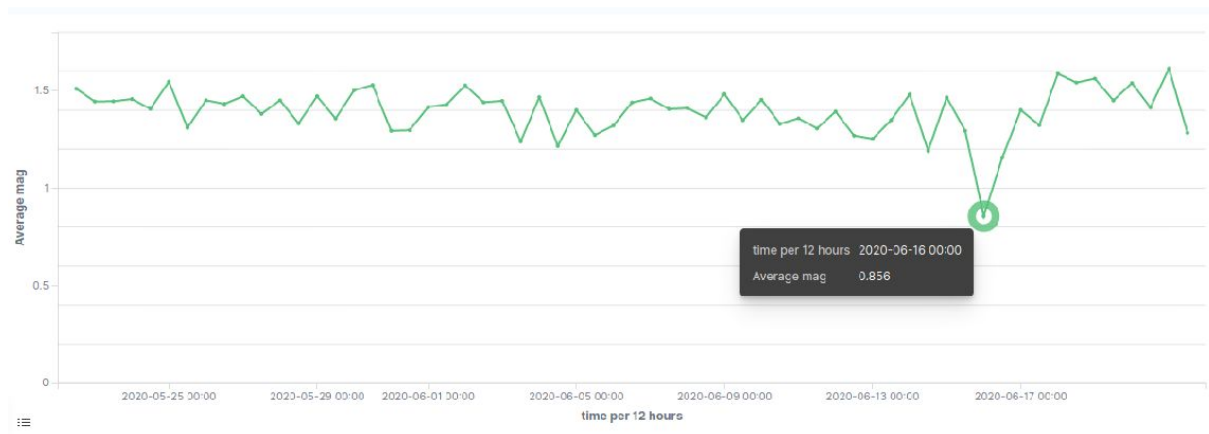
Gambar 22. Tampilan Besar Magnitudo Gempa Bumi

7. Tampilan Frekuensi Kemunculan Magnitudo, memberikan informasi berapa kali magnitudo gempa bumi dalam besaran tertentu muncul.



Gambar 23. Tampilan Frekuensi Kemunculan Magnitudo

8. Tampilan Rata-Rata Magnitudo Gempa Bumi, memberikan informasi berapa rata-rata magnitudo gempa bumi yang terjadi sepanjang waktu tertentu.



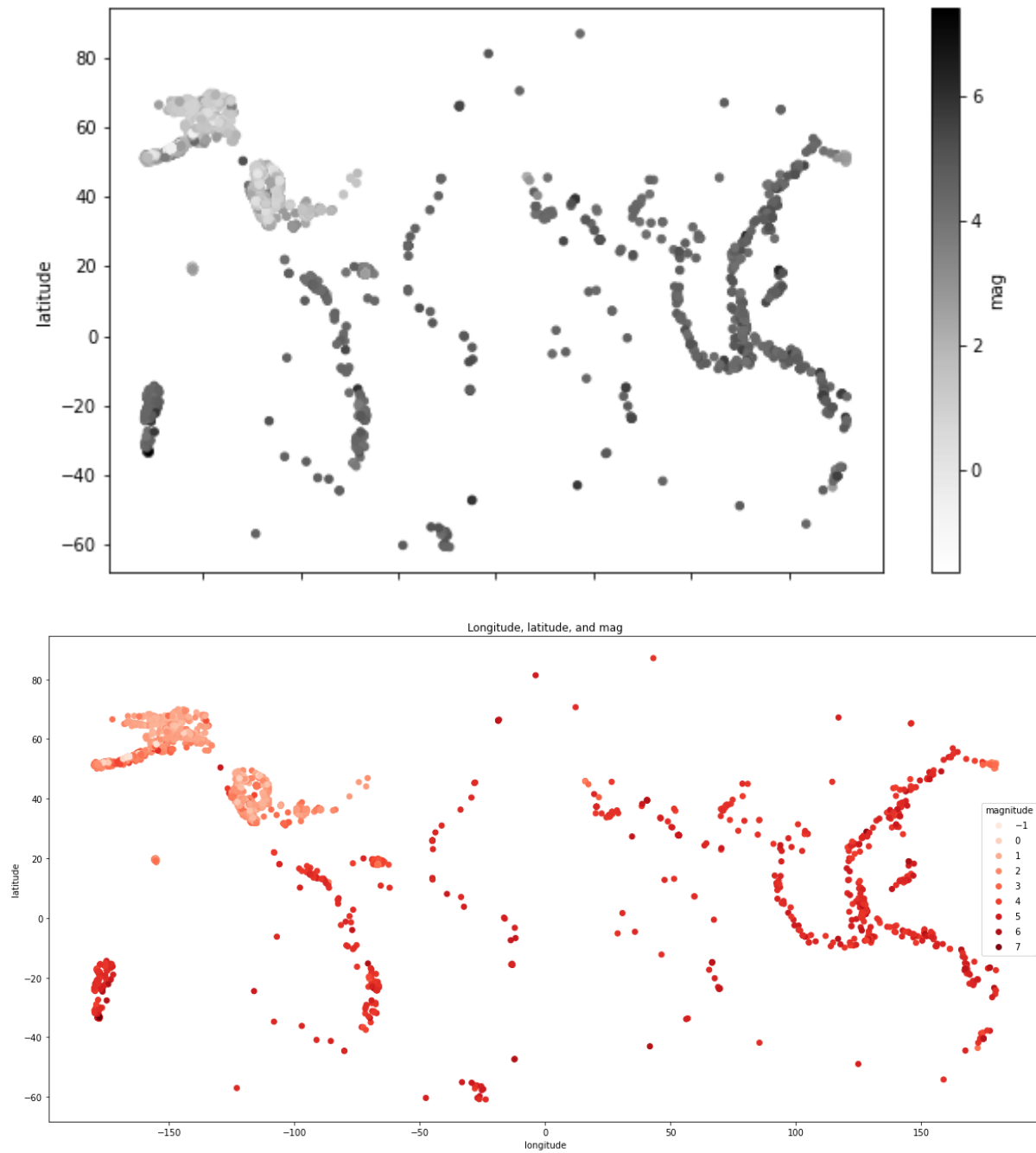
Gambar 24. Tampilan Rata-Rata Magnitudo Gempa Bumi

9. Tampilan Prediksi Rata-Rata Magnitudo Error, memberikan informasi bahwa besarnya magnitudo sebelum-sebelumnya memiliki nilai error sehingga tidak 100% benar besaran magnitudo tersebut.



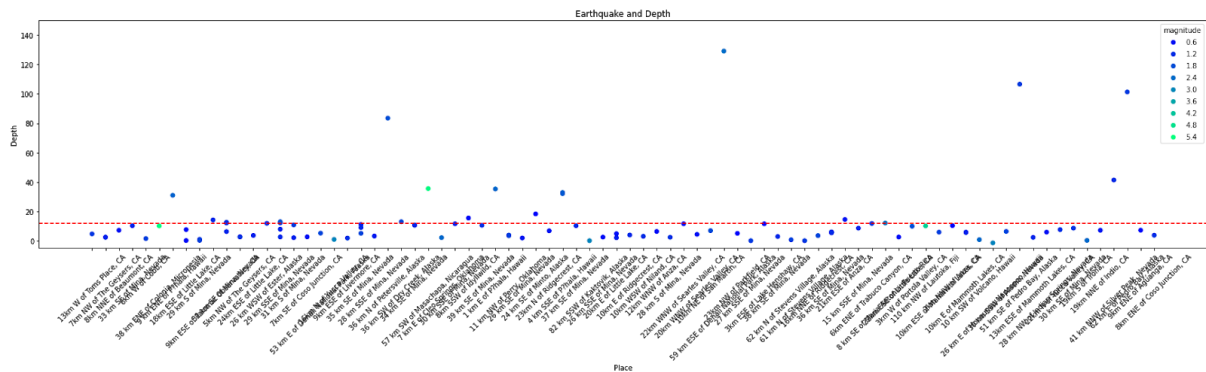
Gambar 25. Tampilan Prediksi Rata-Rata Magnitudo Error

10. Tampilan Sebaran Gempa Bumi Berdasarkan Latitude dan Longitude, memberikan informasi gempa bumi terjadi pada wilayah tertentu berdasarkan koordinat latitude dan longitude dengan besaran magnitudo tertentu yang ditandai dengan tingkat warna.



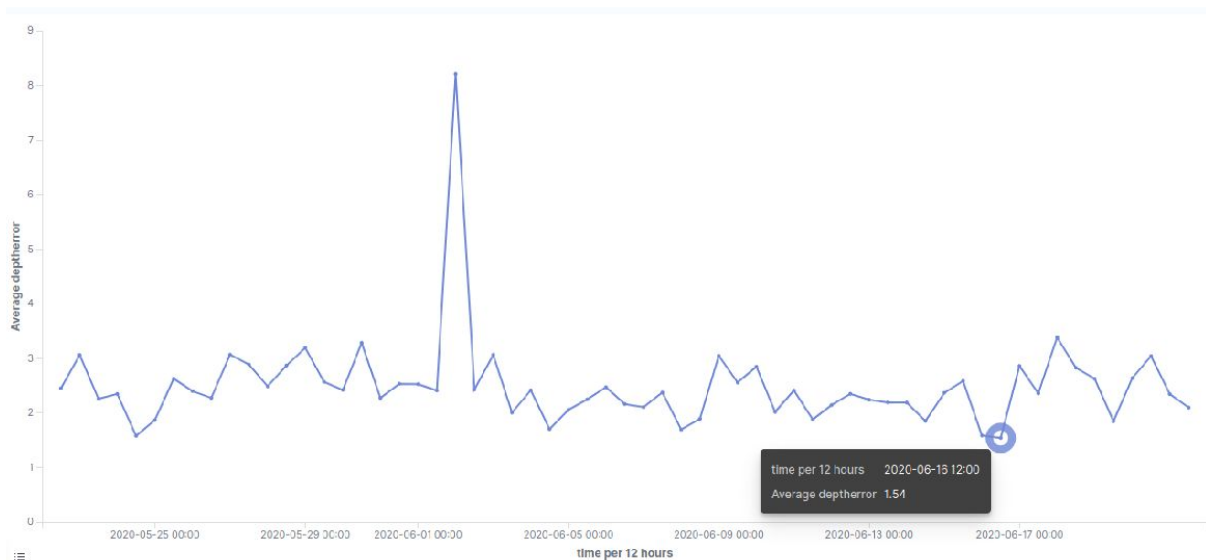
Gambar 26. Tampilan Sebaran Gempa Bumi Berdasarkan Latitude dan Longitude

11. Tampilan Magnitudo dan Depth Berdasarkan Place, memberikan informasi tingkat kedalaman suatu gempa bumi dengan besar magnitudo disetiap tempat yang terjadi.



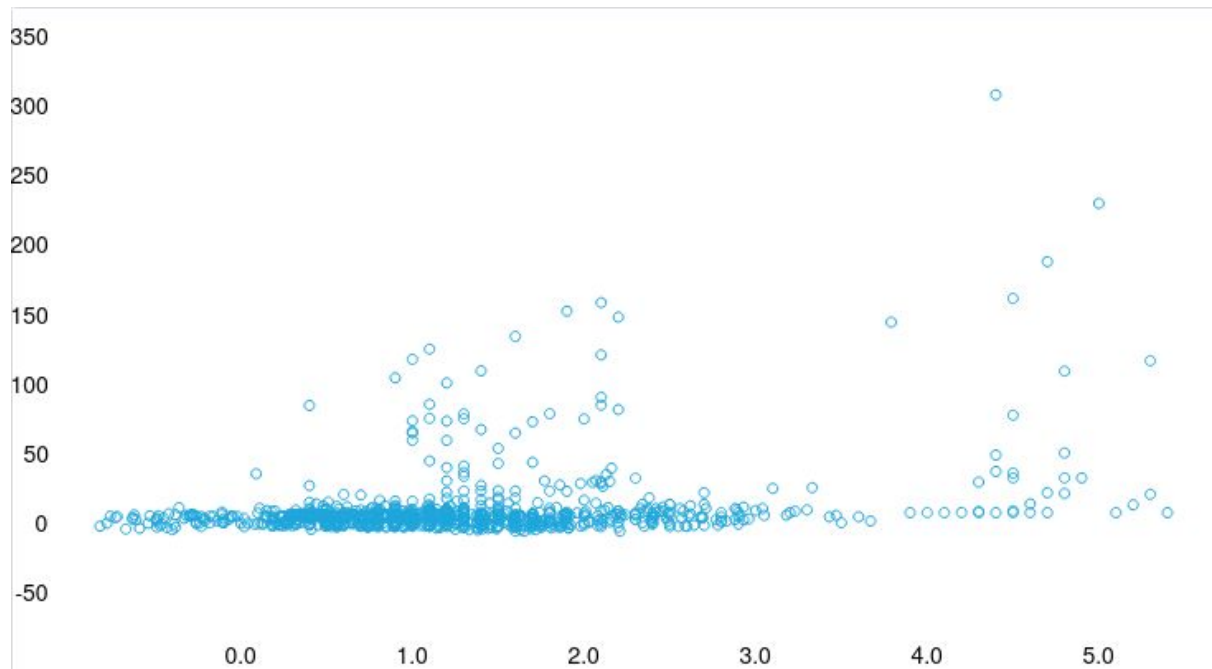
Gambar 27. Tampilan Magnitudo dan Depth Berdasarkan Place

12. Tampilan Prediksi Rata-Rata Depth Error, memberikan informasi tingkat kesalahan dalam memprediksi kedalaman suatu gempa bumi.



Gambar 28. Tampilan Prediksi Rata-Rata Depth Error

13. Tampilan Hubungan Antara Magnitudo dan Depth, memberikan informasi terkait besarnya magnitudo mempengaruhi tingkat kedalaman gempa bumi tersebut terjadi.



Gambar 29. Tampilan Hubungan Antara Magnitudo dan Depth

j. Kesimpulan

Terdapat lima *insight* utama yang dapat diambil dari visualisasi data sebelumnya diantaranya:

1. Kita bisa lebih mengetahui dimana lokasi yang sering terjadi gempa.
2. Kita bisa tahu berapa besarnya magnitude gempa yang sering terjadi di suatu lokasi.
3. Kita bisa tahu berapa kedalaman gempa yang sering terjadi di suatu lokasi.
4. Dapat membantu peneliti gempa dalam menghitung kesenjangan-kesenjangan antara data perkiraan dengan data aslinya.
5. Kita dapat mempelajari seberapa pengaruh besarnya magnitudo atas tingkat kedalaman.

Data yang ditampilkan pada peta Indonesia sebelumnya adalah titik-titik gempa yang sering di pesisir pantai atau di bagian wilayah laut indonesia dan titik-titik di gempa tersebut merupakan jalur dari *ring of fire* pasifik dan di titik-titik tersebut bisa menjadi tempat yang rawan dari gempa bumi dan yang paling parah ialah tsunami.

Adapun beberapa cara yang harus dilakukan pemerintah untuk meminimalkan dampak dari bencana tersebut di antara lain adalah menyediakan sistem peringatan dini seperti sirine, detektor, alat komunikasi, dan yang paling penting adalah memberikan wawasan terhadap masyarakat tentang pentingnya mitigasi bencana yang bertujuan untuk membiasakan masyarakat untuk bisa menghadapi dan mengambil langkah-langkah penyelamatan diri dari bencana alam. dalam bidang konstruksi harus memenuhi standarisasi bangunan tahan gempa contoh gedung perkantoran dan pendidikan harus mempunyai jalur evakuasi.

k. Referensi

- <https://earthquake.usgs.gov>
- <https://earthquake.usgs.gov/data/comcat/data-eventterms.php>
- <https://refactoring.guru/design-patterns/abstract-factory>
- <https://www.geeksforgeeks.org/abstract-factory-method-python-design-patterns/>
- <https://refactoring.guru/design-patterns/builder>
- <https://www.geeksforgeeks.org/builder-method-python-design-patterns/>
- <https://www.geeksforgeeks.org/chain-of-responsibility-python-design-patterns/>
- <https://refactoring.guru/design-patterns/chain-of-responsibility>
- <https://refactoring.guru/design-patterns/proxy>
- <https://www.geeksforgeeks.org/proxy-method-python-design-patterns/>