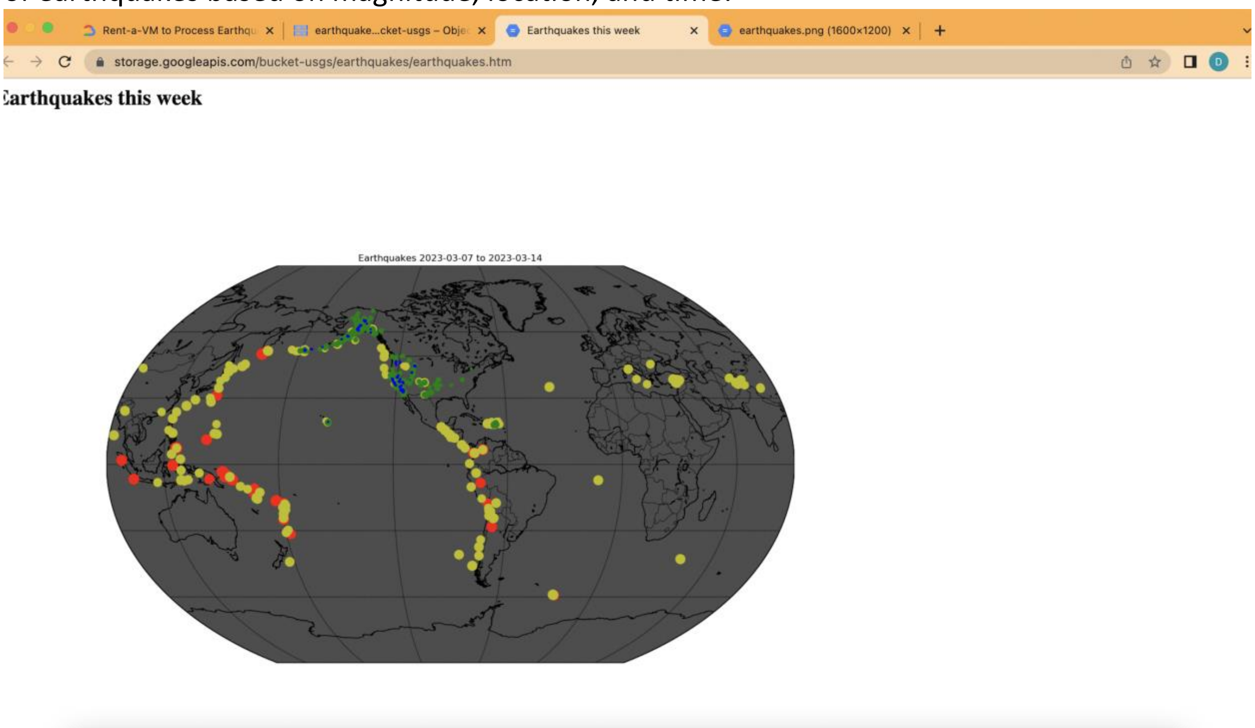
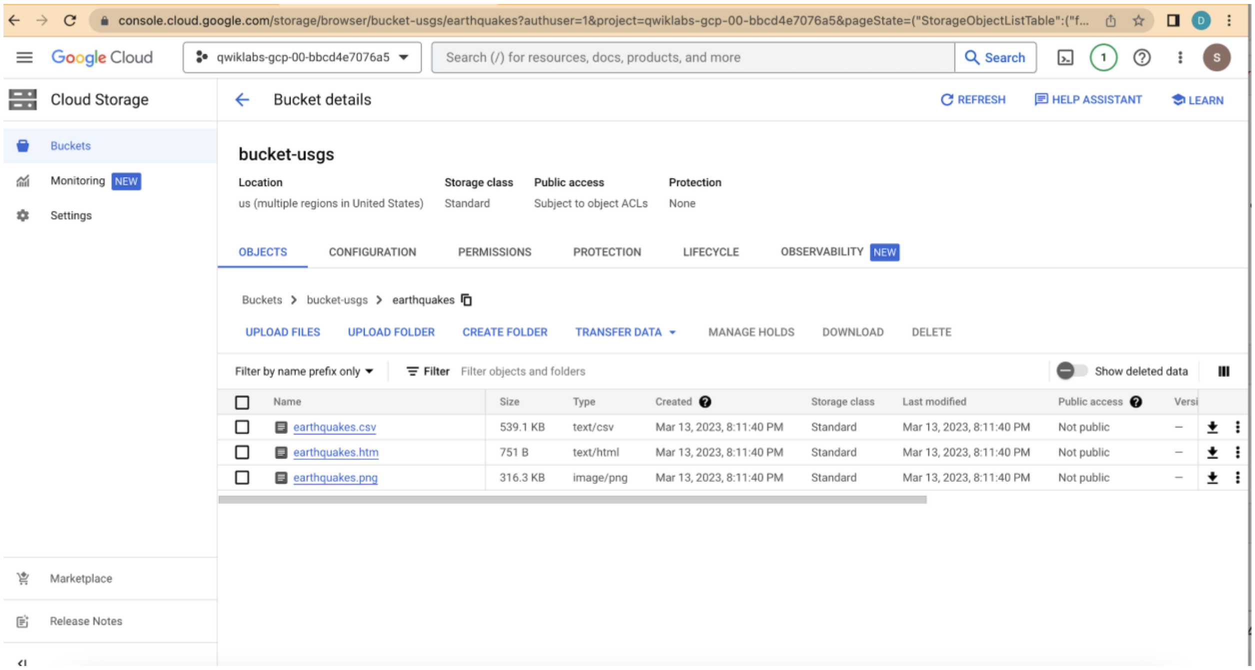
Data Lifecycles – Real time Earthquake visualization project

In this lab, we tried to implement a data pipeline that enabled the visualization of distribution of earthquakes based on magnitude, location, and time.



The different aspects followed in this pipeline are as follows(according to the USGS model in the Data Lifecycles and Pipelines module):

- Acquire (Data ingestion)- In this step, raw earthquake data is collected from the USGS earthquake database using a Python script (quakes = get_earthquake_data('http://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/all_week.csv') from the earthquake.ipynb). This script runs on a Compute Engine virtual machine (VM) instance (created on GCP)and pulls the data from the USGS database and saves it in a Google Cloud Storage (GCS) bucket (bucket-usgs).



- Process(Data Transformation) – Data is transformed to make it ready for analysis

```
import matplotlib.pyplot as plt
from mpl_toolkits.basemap import Basemap
```

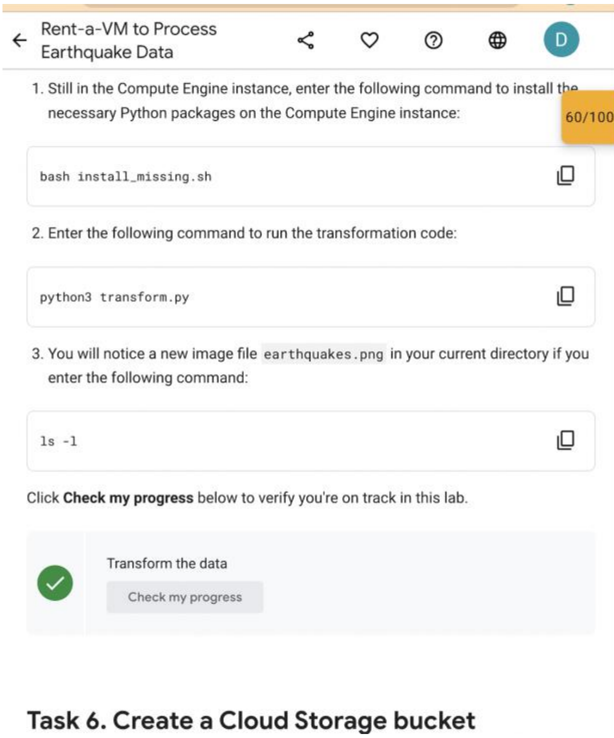
Declare classes to hold data

We'll store the latitude, longitude, magnitude and timestamp of each earthquake and get the data itself from a one-week file that USGS keeps up-to-date. So, each time you run this, you will get a similarly formatted file but the most up-to-date list of earthquakes.

In [17]:

```
class EarthQuake:
  def __init__(self, row):
    # Parse earthquake data from USGS
    self.timestamp = row[0]
    self.lat = float(row[1])
    self.lon = float(row[2])
    try:
      self.magnitude = float(row[4])
    except ValueError:
      self.magnitude = 0

def get_earthquake_data(url):
  # Read CSV earthquake data from USGS
  response = urllib2.urlopen(url)
  csvio = cStringIO.StringIO(response.read())
  reader = csv.reader(csvio)
  header = next(reader)
  quakes = [EarthQuake(row) for row in reader]
  quakes = [q for q in quakes if q.magnitude > 0]
  return quakes

quakes = get_earthquake_data('http://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/all_week.csv')
print quakes[0].__dict__
```

```
{'lat': 33.7578333, 'timestamp': '2016-03-19T19:02:26.270Z', 'magnitude': 0.84, 'lon': -117.5111667}
```



Rent-a-VM to Process Earthquake Data

60/100

1. Still in the Compute Engine instance, enter the following command to install the necessary Python packages on the Compute Engine instance:

```
bash install_missing.sh
```

2. Enter the following command to run the transformation code:

```
python3 transform.py
```

3. You will notice a new image file `earthquakes.png` in your current directory if you enter the following command:

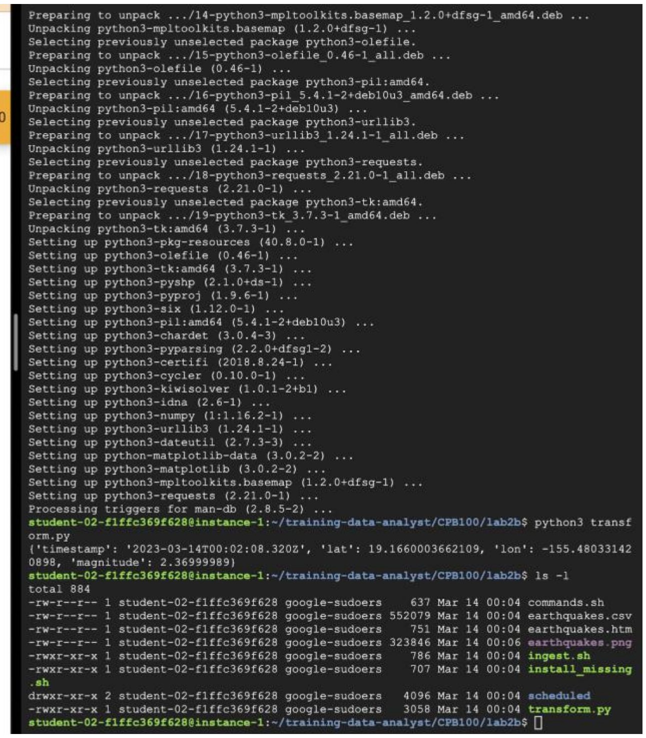
```
ls -l
```

Click **Check my progress** below to verify you're on track in this lab.

Transform the data

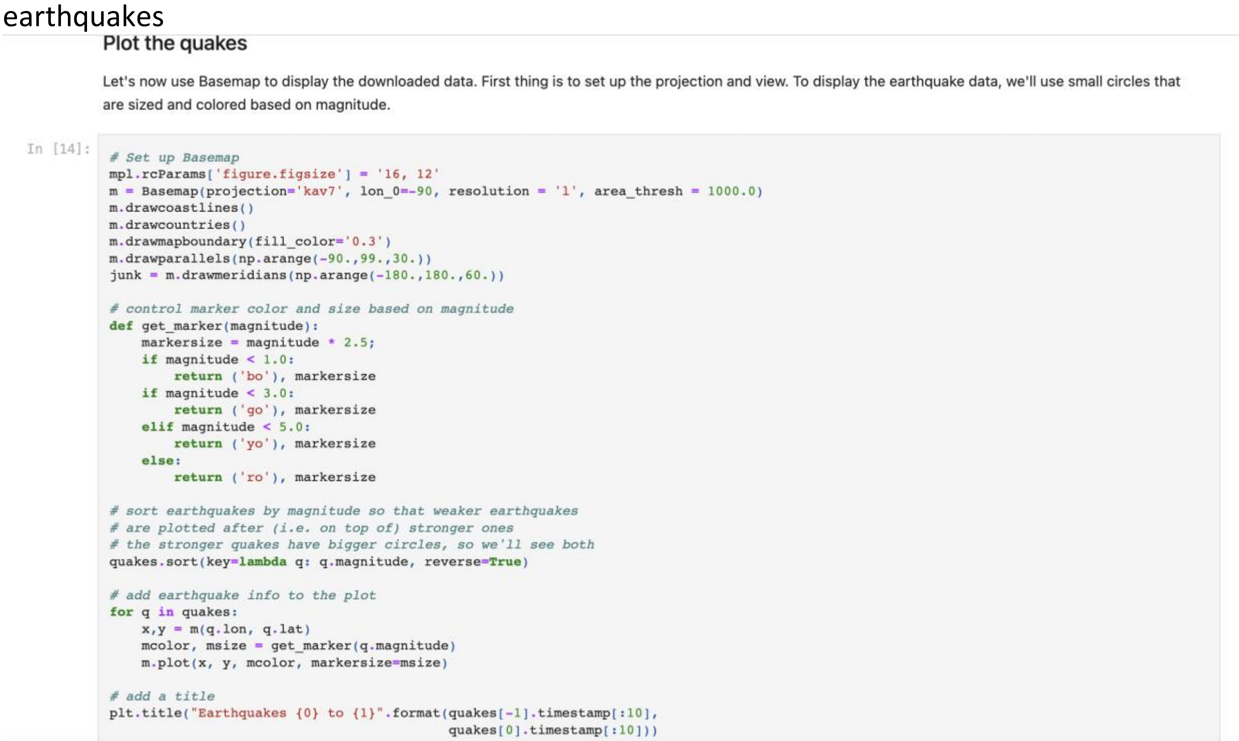
Check my progress

Task 6. Create a Cloud Storage bucket



```
Preparing to unpack .../14-python3-mpltoolkits.basemap_1.2.0+dfsg-1_amd64.deb ...
Unpacking python3-mpltoolkits.basemap (1.2.0+dfsg-1) ...
Selecting previously unselected package python3-olefile.
Preparing to unpack .../15-python3-olefile_0.46-1_all.deb ...
Unpacking python3-olefile (0.46-1) ...
Selecting previously unselected package python3-pil:amd64.
Preparing to unpack .../16-python3-pil_5.4.1-2+deb10u3_amd64.deb ...
Unpacking python3-pil:amd64 (5.4.1-2+deb10u3) ...
Selecting previously unselected package python3-urllib3.
Preparing to unpack .../17-python3-urllib3_1.24.1-1_all.deb ...
Unpacking python3-urllib3 (1.24.1-1) ...
Selecting previously unselected package python3-requests.
Preparing to unpack .../18-python3-requests_2.21.0-1_all.deb ...
Unpacking python3-requests (2.21.0-1) ...
Selecting previously unselected package python3-tk:amd64.
Preparing to unpack .../19-python3-tk_3.7.3-1_amd64.deb ...
Unpacking python3-tk:amd64 (3.7.3-1) ...
Setting up python3-pkg-resources (40.8.0-1) ...
Setting up python3-olefile (0.46-1) ...
Setting up python3-tk:amd64 (3.7.3-1) ...
Setting up python3-pyshp (2.1.0+ds-1) ...
Setting up python3-pyproj (1.9.6-1) ...
Setting up python3-six (1.12.0-1) ...
Setting up python3-pil:amd64 (5.4.1-2+deb10u3) ...
Setting up python3-chardet (3.0.4-3) ...
Setting up python3-pyparsing (2.2.0+dfsg1-2) ...
Setting up python3-certifi (2018.8.24-1) ...
Setting up python3-cycler (0.10.0-1) ...
Setting up python3-kiwisolver (1.0.1-2+b1) ...
Setting up python3-idna (2.6-1) ...
Setting up python3-numpy (1:1.16.2-1) ...
Setting up python3-urllib3 (1.24.1-1) ...
Setting up python3-dateutil (2.7.3-3) ...
Setting up python-matplotlib-data (3.0.2-2) ...
Setting up python3-matplotlib (3.0.2-2) ...
Setting up python3-mpltoolkits.basemap (1.2.0+dfsg-1) ...
Setting up python3-requests (2.21.0-1) ...
Processing triggers for man-db (2.8.5-2) ...
student-02-f1ffc369f628@instance-1:~/training-data-analyst/CPB100/lab2b$ python3 transf
orm.py
{'timestamp': '2023-03-14T00:02:08.320Z', 'lat': 19.1660003662109, 'lon': -155.48033142
0898, 'magnitude': 2.36999989}
student-02-f1ffc369f628@instance-1:~/training-data-analyst/CPB100/lab2b$ ls -l
total 884
-rw-r--r-- 1 student-02-f1ffc369f628 google-sudoers    637 Mar 14 00:04 commands.sh
-rw-r--r-- 1 student-02-f1ffc369f628 google-sudoers 552079 Mar 14 00:04 earthquakes.csv
-rw-r--r-- 1 student-02-f1ffc369f628 google-sudoers    751 Mar 14 00:04 earthquakes.htm
-rw-r--r-- 1 student-02-f1ffc369f628 google-sudoers 323846 Mar 14 00:06 earthquakes.png
-rwxr-xr-x 1 student-02-f1ffc369f628 google-sudoers    786 Mar 14 00:04 ingest.sh
-rwxr-xr-x 1 student-02-f1ffc369f628 google-sudoers    707 Mar 14 00:04 install_missing
.sh
drwxr-xr-x 2 student-02-f1ffc369f628 google-sudoers   4096 Mar 14 00:04 scheduled
-rwxr-xr-x 1 student-02-f1ffc369f628 google-sudoers   3058 Mar 14 00:04 transform.py
student-02-f1ffc369f628@instance-1:~/training-data-analyst/CPB100/lab2b$
```

- Data Analysis - In this step, the data is filtered and grouped by different parameters, such as location, magnitude,and time.
- Data visualization – Map is visualized to analyze the magnitude and location of earthquakes



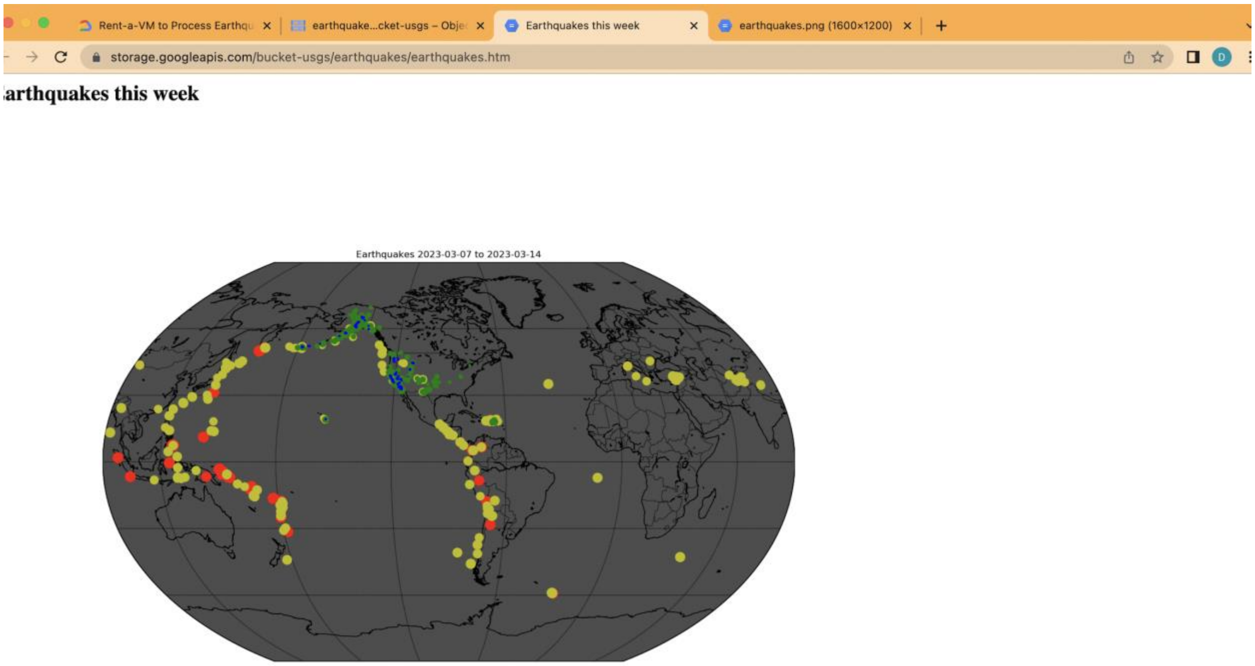
Plot the quakes

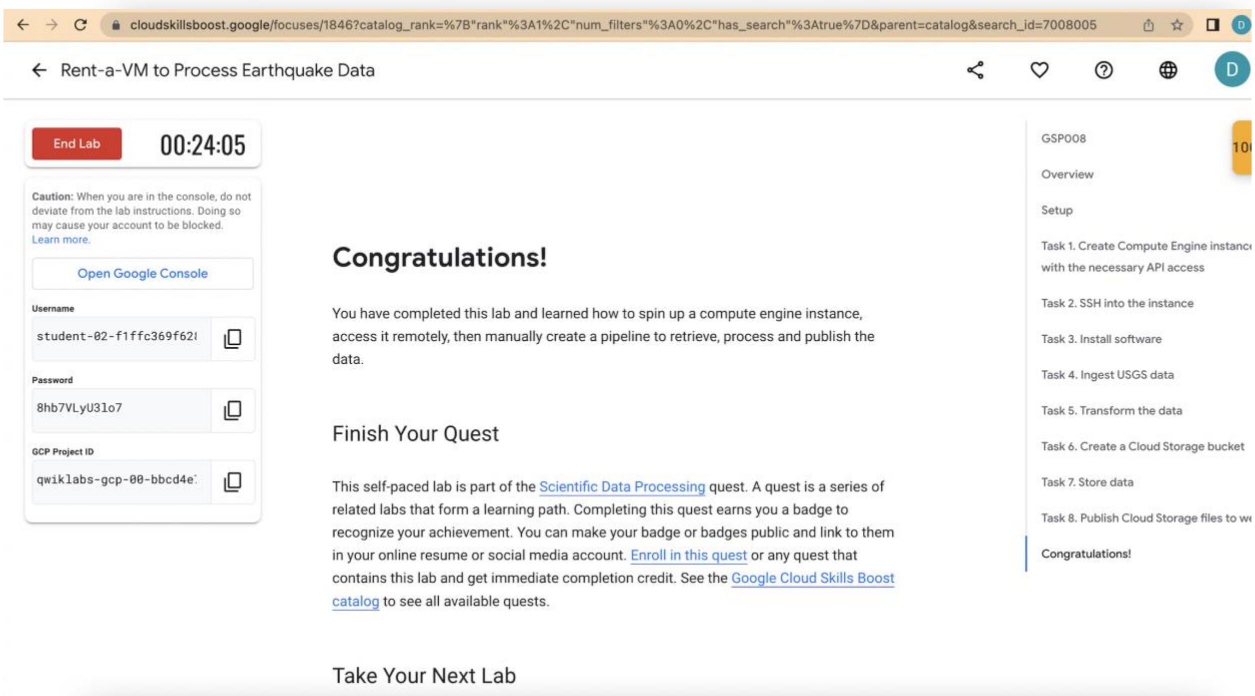
Let's now use Basemap to display the downloaded data. First thing is to set up the projection and view. To display the earthquake data, we'll use small circles that are sized and colored based on magnitude.

In [14]:

```python
# Set up Basemap
mpl.rcParams['figure.figsize'] = '16, 12'
m = Basemap(projection='kav7', lon_0=-90, resolution = 'l', area_thresh = 1000.0)
m.drawcoastlines()
m.drawcountries()
m.drawmapboundary(fill_color='0.3')
m.drawparallels(np.arange(-90.,99.,30.))
junk = m.drawmeridians(np.arange(-180.,180.,60.))

# control marker color and size based on magnitude
def get_marker(magnitude):
    markersize = magnitude * 2.5;
    if magnitude < 1.0:
        return ('bo'), markersize
    if magnitude < 3.0:
        return ('go'), markersize
    elif magnitude < 5.0:
        return ('yo'), markersize
    else:
        return ('ro'), markersize

# sort earthquakes by magnitude so that weaker earthquakes
# are plotted after (i.e. on top of) stronger ones
# the stronger quakes have bigger circles, so we'll see both
quakes.sort(key=lambda q: q.magnitude, reverse=True)

# add earthquake info to the plot
for q in quakes:
    x,y = m(q.lon, q.lat)
    mcolor, msize = get_marker(q.magnitude)
    m.plot(x, y, mcolor, markersize=msize)

# add a title
plt.title("Earthquakes {0} to {1}".format(quakes[-1].timestamp[:10],
                                          quakes[0].timestamp[:10]))
```

- Publish/Share - Publish Cloud Storage data to the web. A publically accessible URL is created for the map created in the storage bucket.





I believe the below additional steps could have been added to further refine this data pipeline:

- Data validation(data cleaning) – This could check whether the data has the required fields and whether it is of the required format.

- Data enhancement – If we could add other features like weather data or earthquake or volcano history in that particular spot, I believe the visualization could have been a little more refined.
- Machine Learning- This could enable the model to predict the occurance of earthquake with the available data.

The pipeline that we implemented with the help of this lab can be represented in the below diagram: