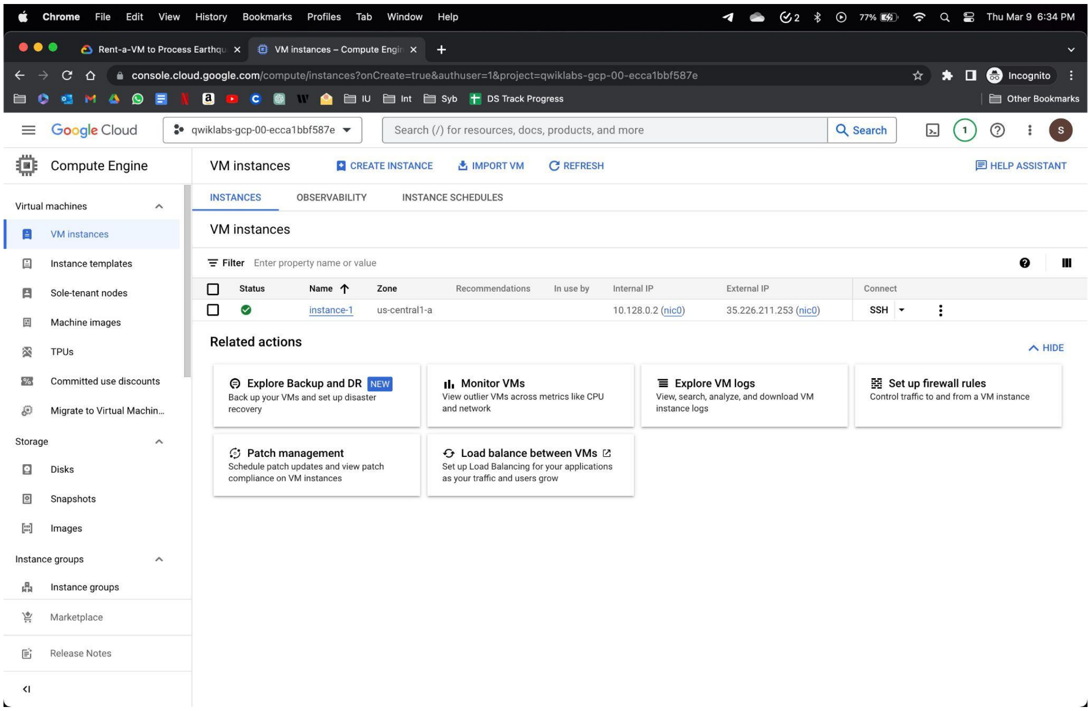
Lifecycle and pipelines

- Aditya Sanjay Mhaske

Rent-a-VM to Process Earthquake Data

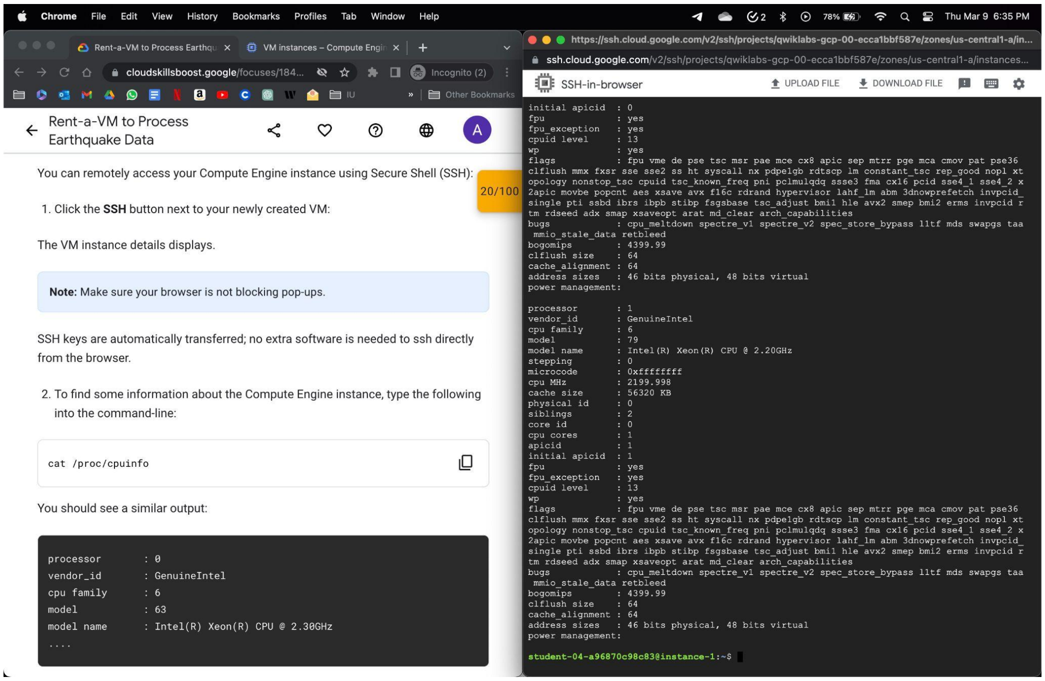
Task 1. Create Compute Engine instance with the necessary API access


Google Cloud
qwiklabs-gcp-00-ecca1bbf587e
Search (/) for resources, docs, products, and more
Search
Compute Engine
VM instances
CREATE INSTANCE
IMPORT VM
REFRESH
HELP ASSISTANT
INSTANCES
OBSERVABILITY
INSTANCE SCHEDULES
Virtual machines
VM instances
Instance templates
Sole-tenant nodes
Machine images
TPUs
Committed use discounts
Migrate to Virtual Machin...
Storage
Disks
Snapshots
Images
Instance groups
Instance groups
Marketplace
Release Notes
Filter Enter property name or value

Status	Name ↑	Zone	Recommendations	In use by	Internal IP	External IP	Connect
✓	instance-1	us-central1-a			10.128.0.2 (nic0)	35.226.211.253 (nic0)	SSH

Related actions
HIDE
Explore Backup and DR NEW
Back up your VMs and set up disaster recovery
Monitor VMs
View outlier VMs across metrics like CPU and network
Explore VM logs
View, search, analyze, and download VM instance logs
Set up firewall rules
Control traffic to and from a VM instance
Patch management
Schedule patch updates and view patch compliance on VM instances
Load balance between VMs
Set up Load Balancing for your applications as your traffic and users grow

Task 2. SSH into the instance



You can remotely access your Compute Engine instance using Secure Shell (SSH):

1. Click the **SSH** button next to your newly created VM:

The VM instance details displays.

Note: Make sure your browser is not blocking pop-ups.

SSH keys are automatically transferred; no extra software is needed to ssh directly from the browser.

2. To find some information about the Compute Engine instance, type the following into the command-line:

```
cat /proc/cpuinfo
```

You should see a similar output:

```
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 63
model name      : Intel(R) Xeon(R) CPU @ 2.30GHz
....
```

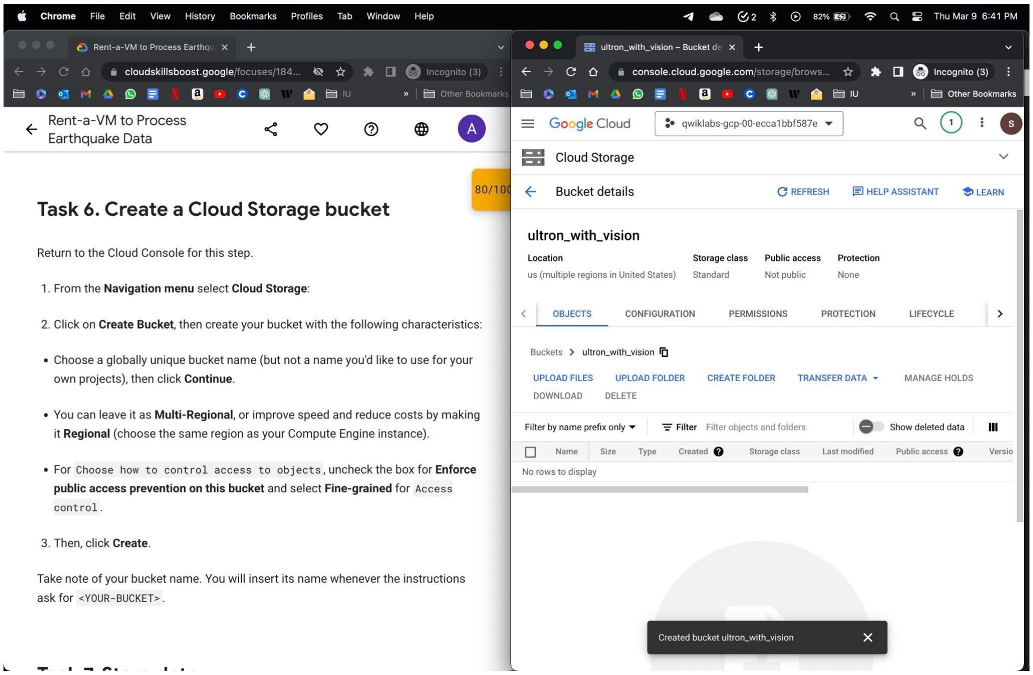
```
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 13
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36
clflush mmx fxsr sse sse2 ss ht syscall nx pdpe1gb rdtscp lm constant_tsc rep_good nopl xt
opology nonstop_tsc cpuid tsc_known_freq pni pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4_2 x
2apic movbe popcnt aes xsave avx f16c rdrand hypervisor lahf_lm abm 3dnowprefetch invpcid_
single pti ssbd ibrs ibpb stibp fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms invpcid r
tm rdseed adx smap xsaveopt arat md_clear arch_capabilities
bugs            : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds swapgs taa
 mmio_stale_data retbleed
bogomips        : 4399.99
clflush size    : 64
cache_alignment : 64
address sizes   : 46 bits physical, 48 bits virtual
power management:

processor       : 1
vendor_id       : GenuineIntel
cpu family      : 6
model           : 79
model name      : Intel(R) Xeon(R) CPU @ 2.20GHz
stepping        : 0
microcode       : 0xffffffff
cpu MHz         : 2199.998
cache size      : 56320 KB
physical id     : 0
siblings        : 2
core id         : 0
cpu cores       : 1
apicid          : 1
initial apicid  : 1
fpu             : yes
fpu_exception   : yes
cpuid level     : 13
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36
clflush mmx fxsr sse sse2 ss ht syscall nx pdpe1gb rdtscp lm constant_tsc rep_good nopl xt
opology nonstop_tsc cpuid tsc_known_freq pni pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4_2 x
2apic movbe popcnt aes xsave avx f16c rdrand hypervisor lahf_lm abm 3dnowprefetch invpcid_
single pti ssbd ibrs ibpb stibp fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms invpcid r
tm rdseed adx smap xsaveopt arat md_clear arch_capabilities
bugs            : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds swapgs taa
 mmio_stale_data retbleed
bogomips        : 4399.99
clflush size    : 64
cache_alignment : 64
address sizes   : 46 bits physical, 48 bits virtual
power management:

student-04-a96870c98c83@instance-1:~$
```

Task 3. Install software

Task 4. Ingest USGS data

Task 5. Transform the data

Task 6. Create a Cloud Storage bucket


Chrome File Edit View History Bookmarks Profiles Tab Window Help
82%
Thu Mar 9 6:41 PM
Rent-a-VM to Process Earthqu
cloudskillsboost.google/focuses/184...
Incognito (3)
Other Bookmarks
ultron_with_vision – Bucket de
console.cloud.google.com/storage/brows...
Rent-a-VM to Process Earthquake Data
80/100
Google Cloud
qwiklabs-gcp-00-ecca1bbf587e
Cloud Storage
REFRESH
HELP ASSISTANT
LEARN
OBJECTS
CONFIGURATION
PERMISSIONS
PROTECTION
LIFECYCLE
UPLOAD FILES
UPLOAD FOLDER
CREATE FOLDER
TRANSFER DATA
MANAGE HOLDS
DOWNLOAD
DELETE
Filter by name prefix only
Filter
Filter objects and folders
Show deleted data

Task 6. Create a Cloud Storage bucket

Return to the Cloud Console for this step.

1. From the **Navigation menu** select **Cloud Storage**:
2. Click on **Create Bucket**, then create your bucket with the following characteristics:

- Choose a globally unique bucket name (but not a name you'd like to use for your own projects), then click **Continue**.
- You can leave it as **Multi-Regional**, or improve speed and reduce costs by making it **Regional** (choose the same region as your Compute Engine instance).
- For `Choose how to control access to objects`, uncheck the box for **Enforce public access prevention on this bucket** and select **Fine-grained** for `Access control`.

3. Then, click **Create**.

Take note of your bucket name. You will insert its name whenever the instructions ask for `<YOUR-BUCKET>`.

Bucket details

ultron_with_vision

Location	Storage class	Public access	Protection
us (multiple regions in United States)	Standard	Not public	None

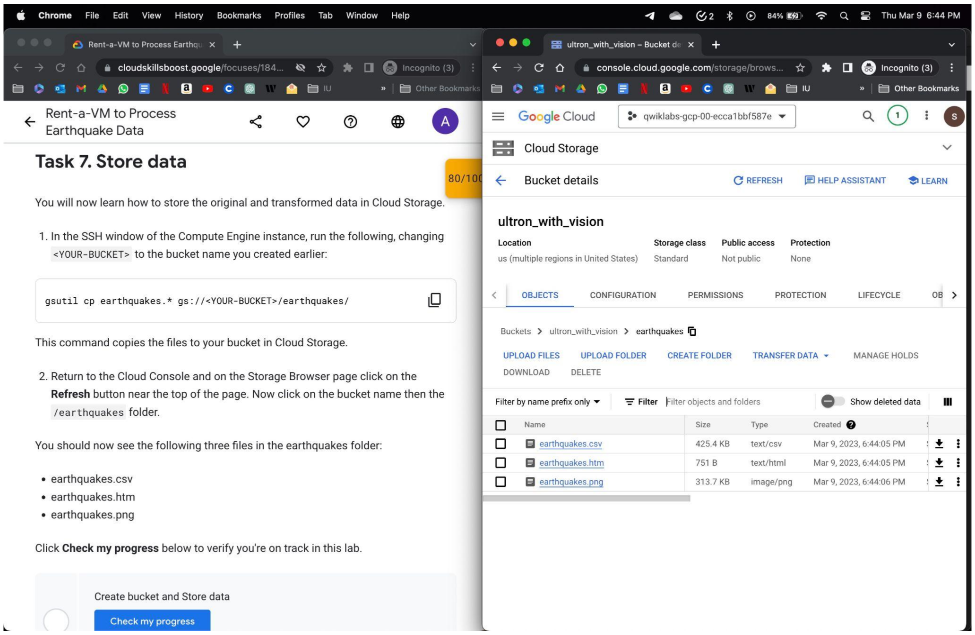
Buckets > ultron_with_vision

Name	Size	Type	Created	Storage class	Last modified	Public access	Versio

No rows to display

Created bucket ultron_with_vision

Task 7. Store data


Chrome File Edit View History Bookmarks Profiles Tab Window Help
84%
Thu Mar 9 6:44 PM
Rent-a-VM to Process Earthqu
cloudskillsboost.google/focuses/184...
Incognito (3)
Other Bookmarks
ultron_with_vision – Bucket de
console.cloud.google.com/storage/brows...
Rent-a-VM to Process Earthquake Data
80/100
Google Cloud
qwiklabs-gcp-00-ecca1bbf587e
Cloud Storage
REFRESH
HELP ASSISTANT
LEARN
OBJECTS
CONFIGURATION
PERMISSIONS
PROTECTION
LIFECYCLE
OB
UPLOAD FILES
UPLOAD FOLDER
CREATE FOLDER
TRANSFER DATA
MANAGE HOLDS
DOWNLOAD
DELETE
Filter by name prefix only
Filter
Filter objects and folders
Show deleted data

Task 7. Store data

You will now learn how to store the original and transformed data in Cloud Storage.

1. In the SSH window of the Compute Engine instance, run the following, changing `<YOUR-BUCKET>` to the bucket name you created earlier:

```
gsutil cp earthquakes.* gs://<YOUR-BUCKET>/earthquakes/
```

This command copies the files to your bucket in Cloud Storage.

2. Return to the Cloud Console and on the Storage Browser page click on the **Refresh** button near the top of the page. Now click on the bucket name then the `/earthquakes` folder.

You should now see the following three files in the earthquakes folder:

- earthquakes.csv
- earthquakes.htm
- earthquakes.png

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

Create bucket and Store data

Check my progress

Bucket details

ultron_with_vision

Location	Storage class	Public access	Protection
us (multiple regions in United States)	Standard	Not public	None

Buckets > ultron_with_vision > earthquakes

Name	Size	Type	Created
earthquakes.csv	425.4 KB	text/csv	Mar 9, 2023, 6:44:05 PM
earthquakes.htm	751 B	text/html	Mar 9, 2023, 6:44:05 PM
earthquakes.png	313.7 KB	image/png	Mar 9, 2023, 6:44:06 PM

Task 8. Publish Cloud Storage files to web

Congratulations!

Describe the high-level architecture of this pipeline by covering the following:

- Data sources and types
- Data storage technologies
- Data transformations tools and techniques (make sure to look into the Python code to understand that)
- Other data components that are included (e.g., visualization, reporting, etc.)

1. Data sources and types:

The pipeline processes earthquake data from the US Geological Survey (USGS) website, which provides real-time data on earthquakes worldwide. The data is in the form of CSV files containing information such as each earthquake's date, time, location, magnitude, and depth.

2. Data storage technologies:

The pipeline uses several Google Cloud storage technologies, including Google Cloud Storage (GCS), Google Bigtable, and Google Pub/Sub. The CSV files are stored in GCS buckets, and the pipeline uses Bigtable to store and manage earthquake metadata. Pub/Sub is used to decouple the different stages of the pipeline and facilitate communication between them.

3. Data transformations tools and techniques:

The pipeline uses a combination of Python libraries and Google Cloud services to transform and process earthquake data. The main Python libraries used are pandas, NumPy, and Matplotlib, which are used for data manipulation, processing, and visualization. The pipeline uses Google Dataflow to parallelize the data processing and apply transformations such as filtering, grouping, and aggregating. Additionally, the pipeline uses Google Cloud Functions to trigger the data processing when new data is available in the GCS buckets.

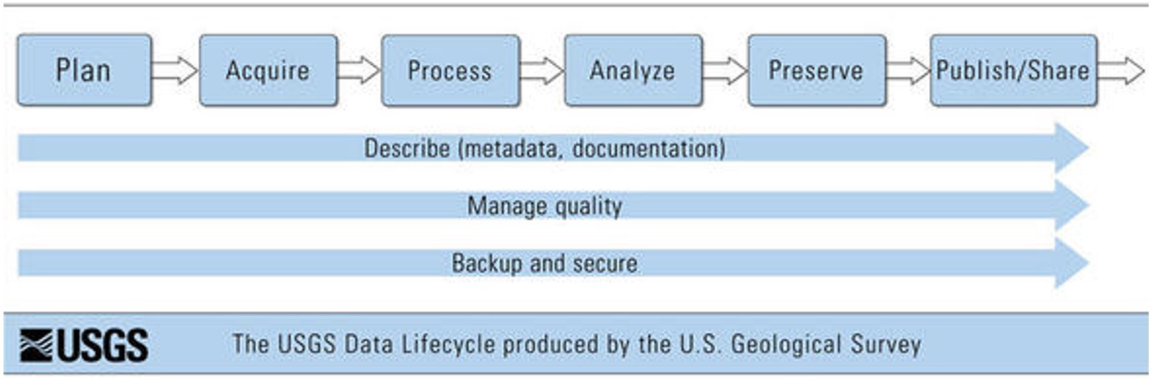
4. Other data components included:

The pipeline includes several data components, including data visualization and reporting. Matplotlib is used to create various plots and visualizations of the earthquake data, such as scatter plots and histograms. The pipeline also generates earthquake reports, including summaries of the earthquake data by magnitude and location. The reports are stored in GCS buckets and can be accessed through a web application built using Google App Engine.

This pipeline covers several aspects of the data lifecycle, including:

In this week's module, the USGS model was explained. The following steps were taken:

- Defining the data plan, which involved identifying the required process and resources.
- Collecting the data from the USGS website into the VM.
- Processing the data using the transform.py file and implementing the mentioned transformations.
- Analyzing the data by creating visualizations on a map.
- Storing the data on the bucket.
- Ensuring accessibility of the data by making it available on the bucket and generating an earthquakes.htm file to facilitate sharing/publishing.


Plan
Acquire
Process
Analyze
Preserve
Publish/Share
Describe (metadata, documentation)
Manage quality
Backup and secure
USGS
The USGS Data Lifecycle produced by the U.S. Geological Survey

Citation:

1. https://www.usgs.gov/data-management/data-lifecycle
2. https://docs.gcp.databricks.com/getting-started/data-pipeline-get-started.html

Discuss what could be added to the pipeline using this module's materials.

There are several components that could be added to this pipeline using the Google Cloud Module's materials:

1. Real-time data streaming: Currently, the pipeline processes earthquake data that is collected periodically from the USGS website. By adding real-time data streaming capabilities using Google Cloud Pub/Sub, the pipeline could process and analyze earthquake data in real time, which would be useful for disaster response and emergency management.

2. Machine learning models: The pipeline could be enhanced by adding machine learning models that could predict the likelihood and severity of earthquakes based on historical data. Google Cloud's AutoML and AI Platform could be used to train and deploy these models.

3. Anomaly detection: The pipeline could include an anomaly detection system to detect unusual patterns or trends in the earthquake data, which could be an indication of seismic activity that requires further investigation.

4. Integration with other data sources: The pipeline could be expanded to include data from other sources, such as weather data or satellite imagery, to provide more context and insights into the earthquake data.