CS570: Week 2 Homework 1 : MapReduce on Ubuntu

Md Shahadat Hossain Khan (19609)

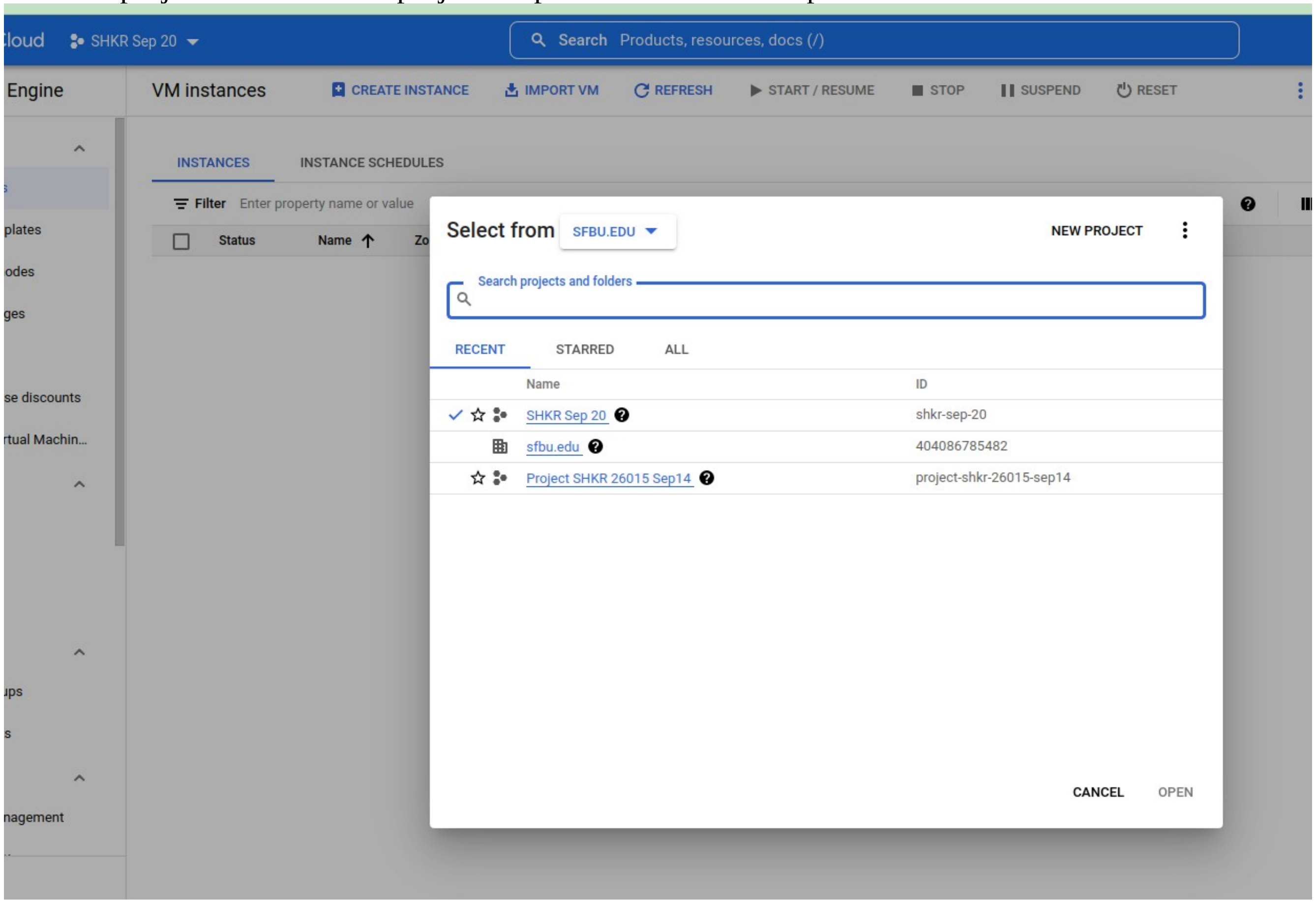
MapReduce on Ubuntu - IDE

Step 1: Create a Ubuntu virtual Machine on GCP
Step 2: Hadoop: Setting up a Single Node Cluster.
Step 3: MapReduce Tutorial: Word Count

>>>

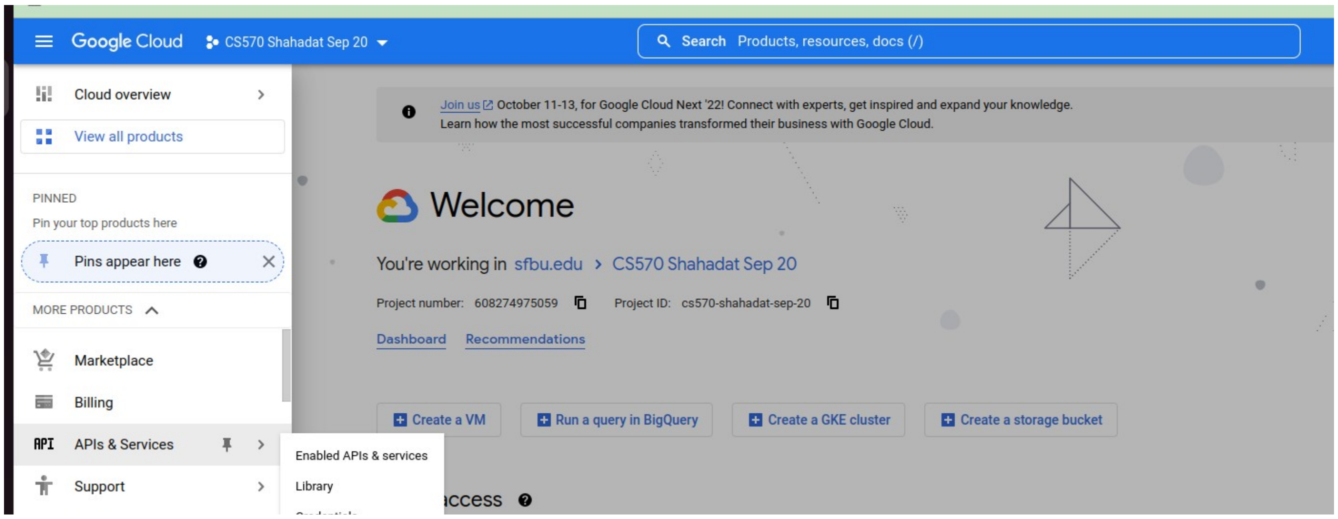
Create a Ubuntu virtual Machine on GCP

> Create project: Click current project drop down menu from top menu



Name	ID
SHKR Sep 20	shkr-sep-20
sfbu.edu	404086785482
Project SHKR 26015 Sep14	project-shkr-26015-sep14

> Click “NEW PROJECT” from top right button of opened pop-up
> Provide a project name and click “Create”
> Now we need to select our new project, to do that we need to click project drop down menu from top menu to open project list pop-up
> Click on newly create project
> Now we need to enable “Compute Engine API”, to do that click “API & Services” from left menu.

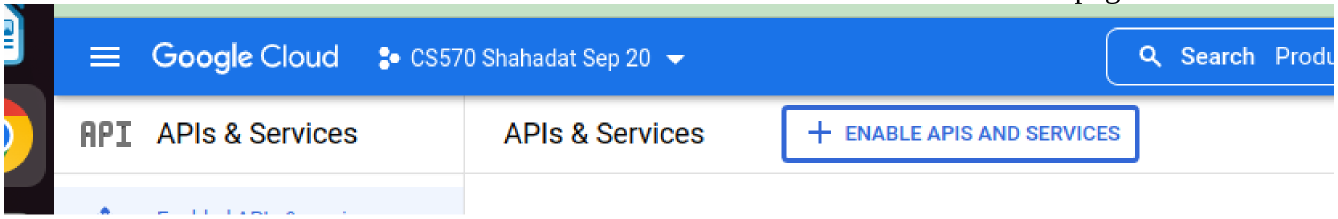


Join us October 11-13, for Google Cloud Next '22! Connect with experts, get inspired and expand your knowledge. Learn how the most successful companies transformed their business with Google Cloud.

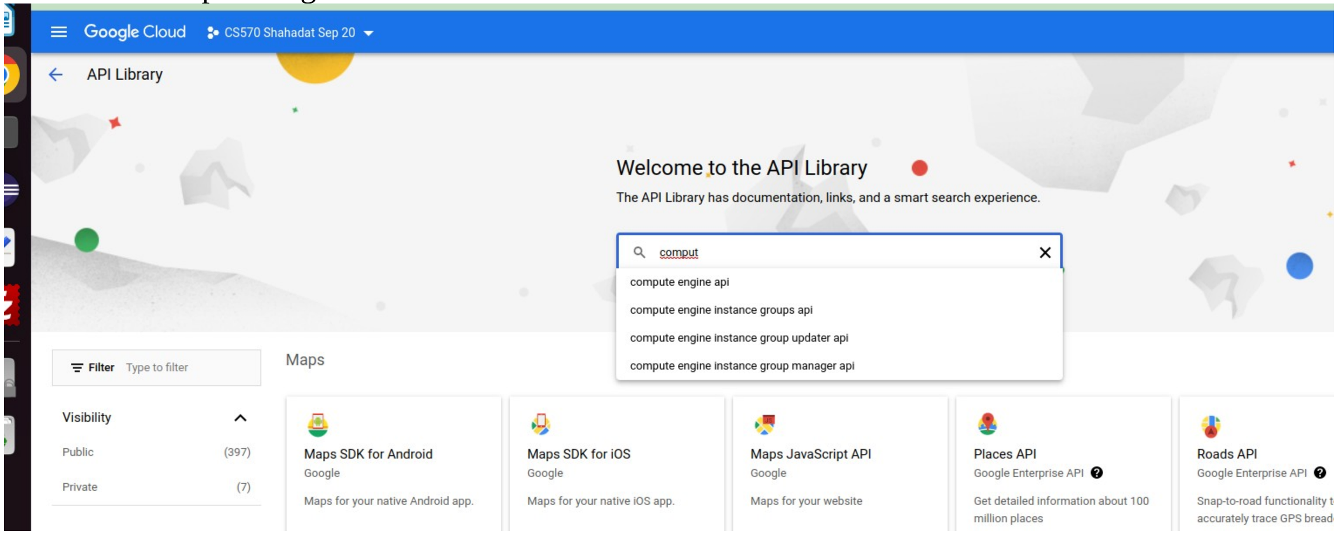
You're working in sfbu.edu > CS570 Shahadat Sep 20

Project number: 608274975059 Project ID: cs570-shahadat-sep-20

> Now need to click "+ ENABLE APIS AND SERVICES" from header area of the page



> Search "Compute Engine API" at search box



Welcome to the API Library

The API Library has documentation, links, and a smart search experience.

Maps for your native Android app.

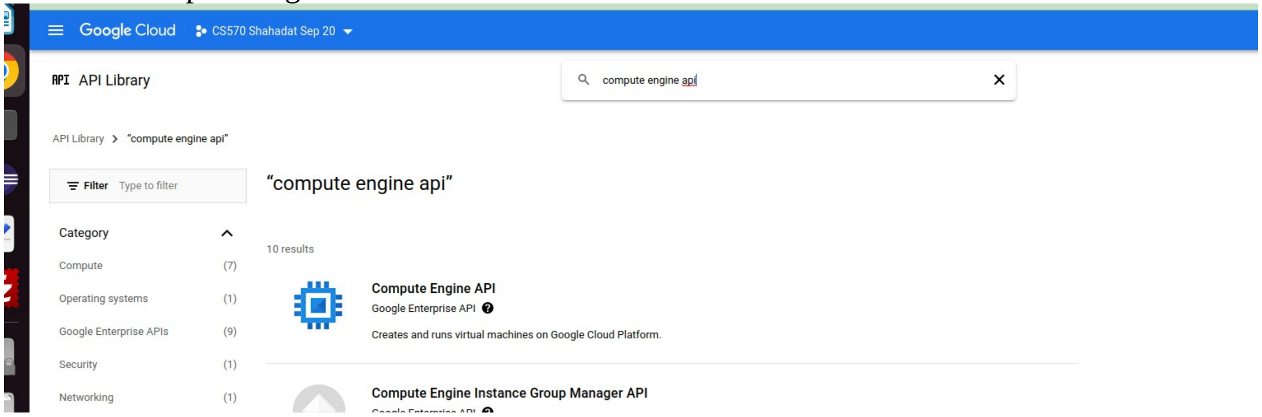
Maps for your native iOS app.

Maps for your website

Get detailed information about 100 million places

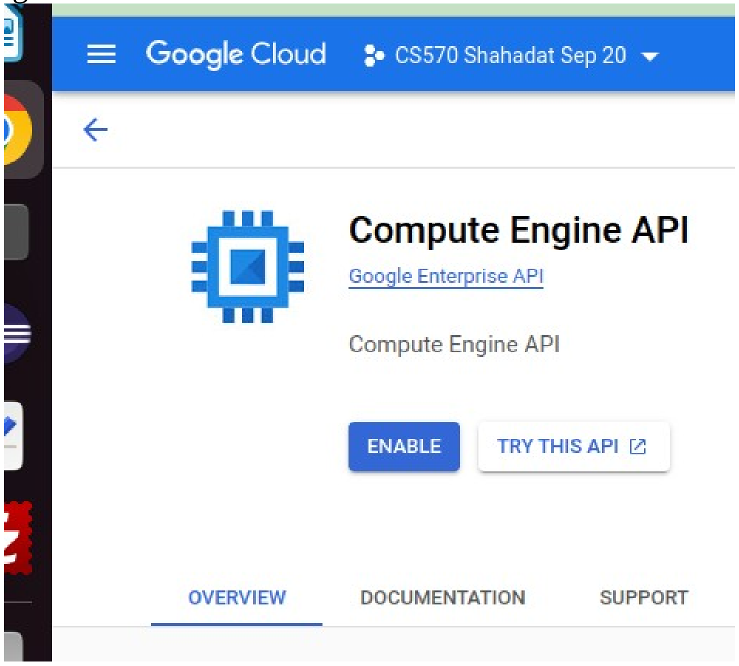
Snap-to-road functionality t accurately trace GPS bread

> Click on "Compute Engine API" from search result

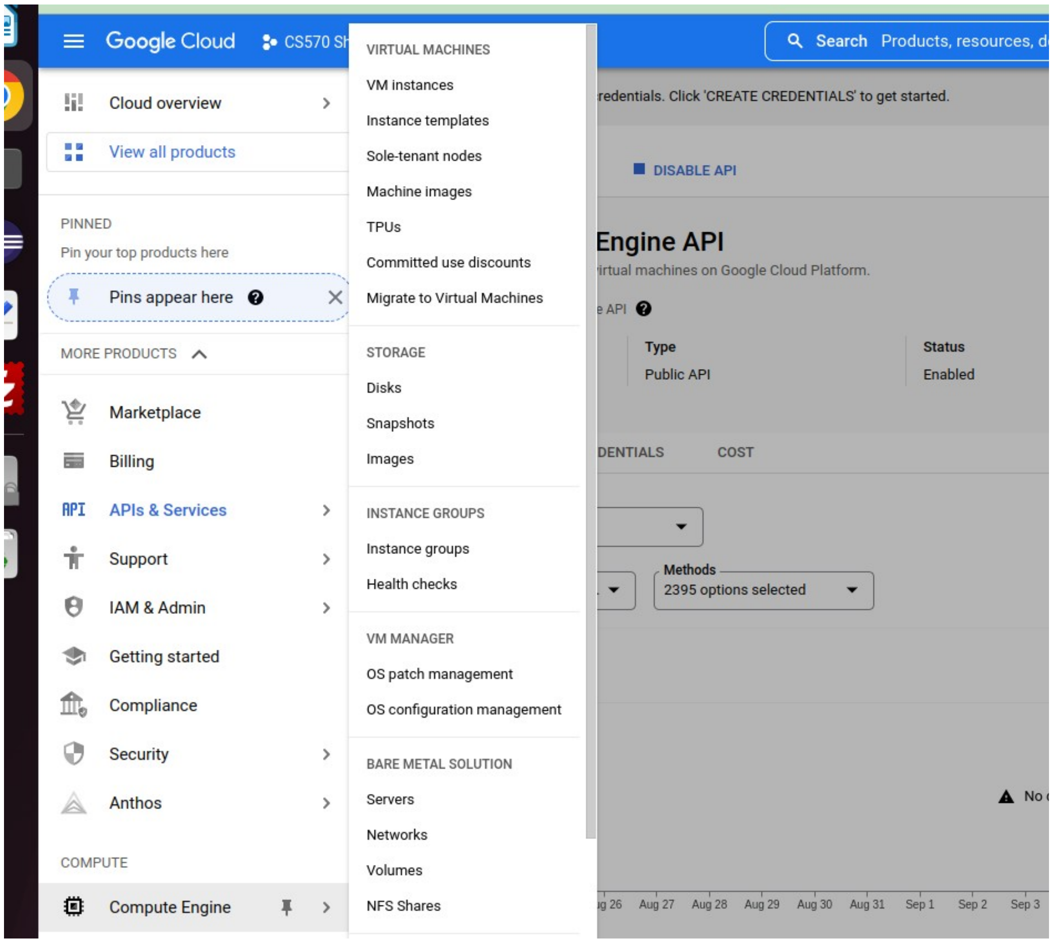


Creates and runs virtual machines on Google Cloud Platform.

> Enable API by clicking “Enable” button



> After enabling API open VM instances page by clicking “Compute Engine” from left menu



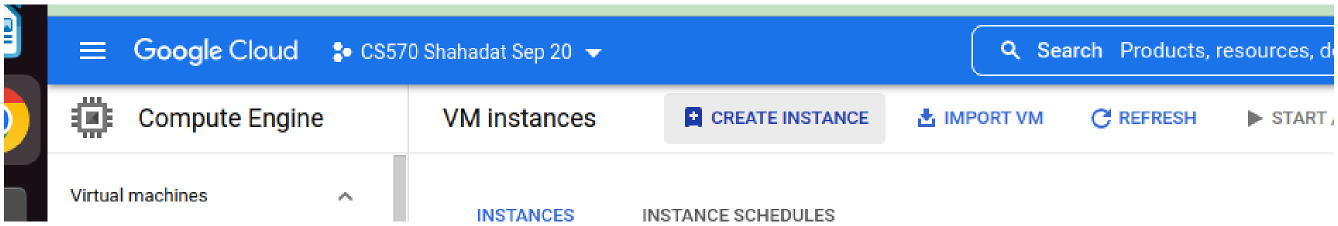
redentials. Click 'CREATE CREDENTIALS' to get started.

irtual machines on Google Cloud Platform.

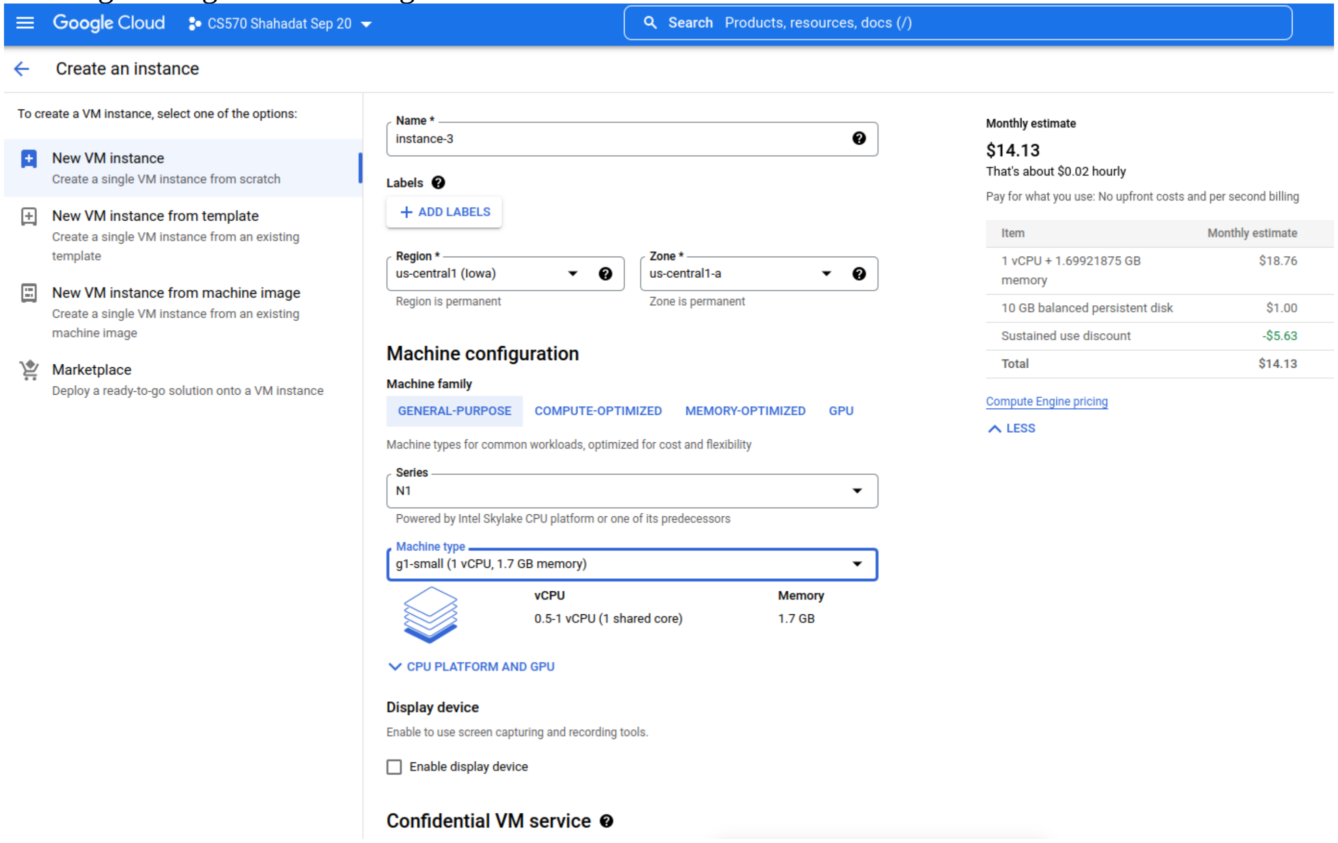
Type	Status
Public API	Enabled

Methods: 2395 options selected

> Click “Create Instance” from header menu



> Change configuration as image below



To create a VM instance, select one of the options:

New VM instance — Create a single VM instance from scratch

New VM instance from template — Create a single VM instance from an existing template

New VM instance from machine image — Create a single VM instance from an existing machine image

Marketplace — Deploy a ready-to-go solution onto a VM instance

Name *: instance-3

Labels

Region *: us-central1 (Iowa) — Region is permanent

Zone *: us-central1-a — Zone is permanent

Machine configuration

Machine family: GENERAL-PURPOSE

Machine types for common workloads, optimized for cost and flexibility

Series: N1 — Powered by Intel Skylake CPU platform or one of its predecessors

Machine type: g1-small (1 vCPU, 1.7 GB memory)

vCPU	Memory
0.5-1 vCPU (1 shared core)	1.7 GB

Display device

Enable to use screen capturing and recording tools.

☐ Enable display device

Confidential VM service

Monthly estimate

$14.13

That's about $0.02 hourly

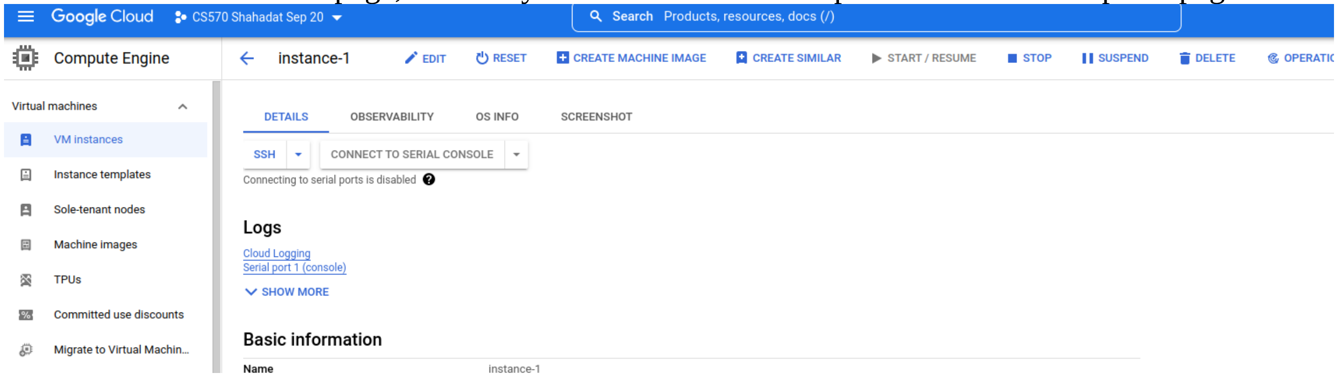
Pay for what you use: No upfront costs and per second billing

Item	Monthly estimate
1 vCPU + 1.69921875 GB memory	$18.76
10 GB balanced persistent disk	$1.00
Sustained use discount	-$5.63
Total	$14.13

> Click “Create” button and wait until its ready to use

We are going to add our SSH key into new VM

> At list of VM instances page, click on your new VM name to open that VM control panel page



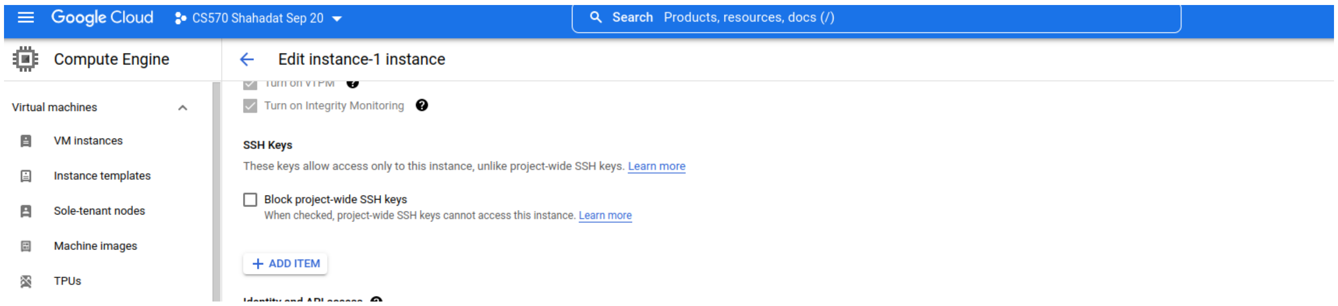
Connecting to serial ports is disabled

Logs

Basic information

Name: instance-1

> Now click “Edit” from that control panel
> From Edit page find “SSH” section click “+ ADD ITEM”



Turn on vTPM

Turn on Integrity Monitoring

SSH Keys

These keys allow access only to this instance, unlike project-wide SSH keys. Learn more

Block project-wide SSH keys
When checked, project-wide SSH keys cannot access this instance. Learn more

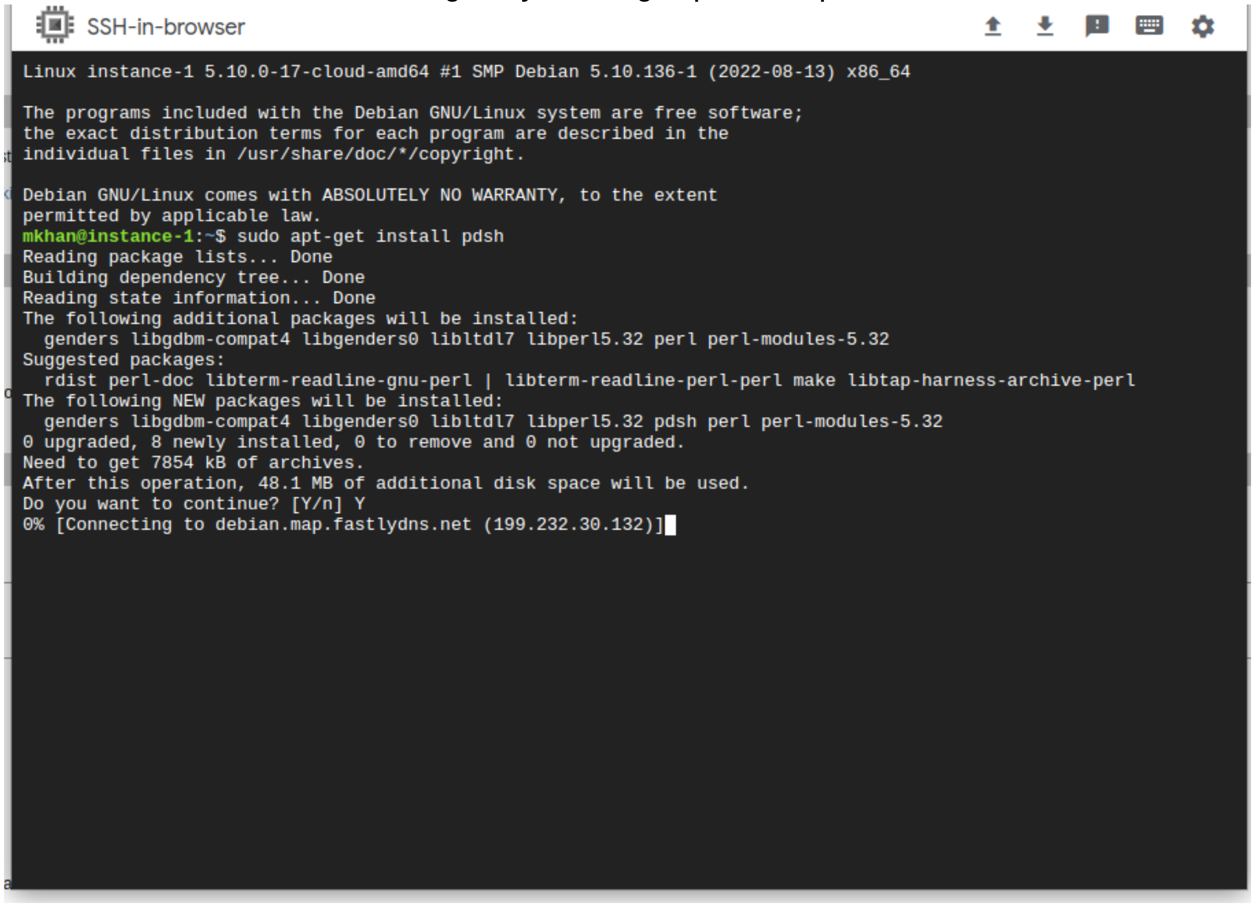
+ ADD ITEM

> Paste your local SSH public key into text box showed up and click "Save" button at the very bottom part of this page

Note: Please refer my assignment from week 1 and homework 2 to find proper format of SSH key before paste.

Hadoop: Setting up a Single Node Cluster

> sudo apt-get update
> sudo apt-get install wget
> sudo apt-get install pdsh
> echo "ssh" | sudo tee /etc/pdsh/rcmd_default
Add `export PDSH_RCMD_TYPE=ssh into ~/.bashrc`
`You can check rcmd settings by using "pdsh -q -w localhost"`



```
Linux instance-1 5.10.0-17-cloud-amd64 #1 SMP Debian 5.10.136-1 (2022-08-13) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
mkhan@instance-1:~$ sudo apt-get install pdsh
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  genders libgdbm-compat4 libgenders0 libltdl7 libperl5.32 perl perl-modules-5.32
Suggested packages:
  rdist perl-doc libterm-readline-gnu-perl | libterm-readline-perl-perl make libtap-harness-archive-perl
The following NEW packages will be installed:
  genders libgdbm-compat4 libgenders0 libltdl7 libperl5.32 pdsh perl perl-modules-5.32
0 upgraded, 8 newly installed, 0 to remove and 0 not upgraded.
Need to get 7854 kB of archives.
After this operation, 48.1 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
0% [Connecting to debian.map.fastlydns.net (199.232.30.132)]
```

> sudo apt-get install software-properties-common

At this stage we need to install java but hadoop need java 8.

We can install Java 8 automatically or manually. I did it manually, so I download java 8 from oracle archive site. Here are some link https://www.oracle.com/java/technologies/javase/javase8u211-later-archive-downloads.html and untared it and set some configuration as follows -

```
> cd /usr/lib/jvm/
> sudo tar -xvzf jdk-8u333-linux-x64.tar.gz
> sudo update-alternatives --install "/usr/bin/java" "java" "/usr/lib/jvm/jdk1.8.0_333/bin/java" 0
> sudo update-alternatives --install "/usr/bin/javac" "javac" "/usr/lib/jvm/jdk1.8.0_333/bin/javac" 0
> sudo update-alternatives --set java /usr/lib/jvm/jdk1.8.0_333/bin/java
> sudo update-alternatives --set javac /usr/lib/jvm/jdk1.8.0_333/bin/javac
```

That's all for my Java installation, now we need to set java path (if it is not set already) by modifying /etc/environment file, where we need append following line at the end.

JAVA_HOME="/usr/lib/jvm/jdk1.8.0_333"

Also we need to change PATH variable into this file. We need to prepend ${JAVA_HOME}

```
	e.g.
	PATH="/usr/lib/jvm/jdk1.8.0_333:/usr/local/sbin:/usr/local/bin:/usr/sbin:/
	usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin"
	JAVA_HOME="/usr/lib/jvm/jdk1.8.0_333"
```

Download Hadoop

Now we can access our VM through local ssh.

Visit https://dlcdn.apache.org/hadoop/common/stable/ or https://www.apache.org/dyn/closer.cgi/hadoop/common/ or https://dlcdn.apache.org/hadoop/common/

And using wget like -

```
	wget https://dlcdn.apache.org/hadoop/common/stable/hadoop-3.3.4.tar.gz
```

After untar the hadoop, we can run hadoop locally. But we need to modify some settings to to run hadoop as single node cluster. Here is the commands that need to apply -

```
> cd <hadoop untared location>

> vi etc/hadoop/hadoop-env.sh
```

Add following line at the end of file or find somewhere you found "HADOOP_CLASSPATH" and add there.

```
	export HADOOP_CLASSPATH="${JAVA_HOME}/lib/tools.jar"
```

Now we are ready to test hadoop!

```
> cd ~

> mkdir hadoop-namenode

> mkdir hadoop-datanode
```

> cd <hadoop untared location>

> vi etc/hadoop/core-site.xml

```
<configuration>
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://localhost:9000</value>
    </property>
</configuration>
```

> vi etc/hadoop/hdfs-site.xml

```
<configuration>
    <property>
        <name>dfs.replication</name>
        <value>1</value>
    </property>
<property>
        <name>dfs.permission</name>
        <value>false</value>
    </property>
<property>
        <name>dfs.namenode.name.dir</name>
        <value>file:///home/mkhan/hadoop-namenode</value>
    </property>
    <property>
        <name>dfs.datanode.data.dir</name>
        <value>file:///home/mkhan/hadoop-datanode</value>
    </property>
</configuration>
```

Now we need to able ssh to the localhost without a passphrase. To achieve this apply following commands -

> ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa

> cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys

> chmod 0600 ~/.ssh/authorized_keys

We all set, now we need to reboot our instance, like stop and start or we can restart.

Now we ready to run hadoop as single node cluster. Here is example execution -

> bin/hdfs namenode -format

> sbin/start-dfs.sh

> wget http://localhost:9870/

> bin/hdfs dfs -mkdir /user

> bin/hdfs dfs -mkdir /user/<username>

```
Note: Please use your login user name of your system/instance. In my case "mkhan",
but I don't research about this username properly
```

> bin/hdfs dfs -mkdir input

> bin/hdfs dfs -put etc/hadoop/*.xml input

> bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-3.3.4.jar grep input output 'dfs[a-z.]+'

```
Note: Many warning may come at screen, we can ignore by this time. I didn't
research on these warning.
```

> bin/hdfs dfs -cat output/*

> sbin/stop-dfs.sh

So, we ran hadoop successfully! Now I'm going to execute a real example like compiling code by Java. Following https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html

> bin/hdfs namenode -format

> sbin/start-dfs.sh

> bin/hdfs dfs -mkdir /user

> bin/hdfs dfs -mkdir /user/mkhan

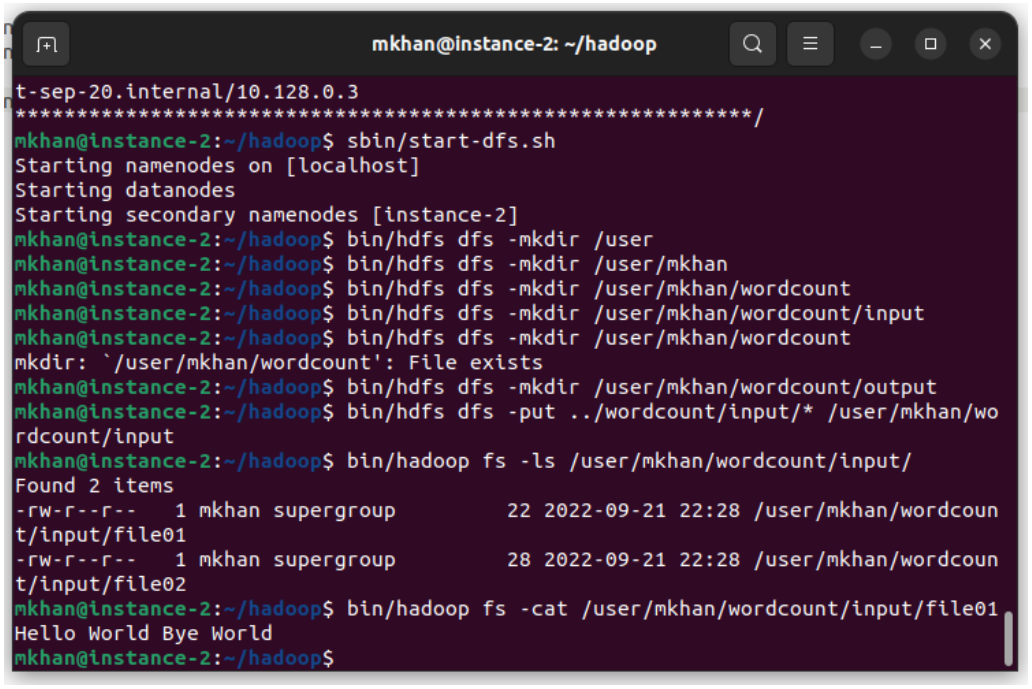
> bin/hdfs dfs -mkdir /user/mkhan/wordcount

> bin/hdfs dfs -mkdir /user/mkhan/wordcount/input

> bin/hdfs dfs -put ../wordcount/input/* /user/mkhan/wordcount/input

> bin/hadoop fs -ls /user/mkhan/wordcount/input/

> bin/hadoop fs -cat /user/mkhan/wordcount/input/file01



mkhan@instance-2: ~/hadoop

```
t-sep-20.internal/10.128.0.3
************************************************************/
mkhan@instance-2:~/hadoop$ sbin/start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [instance-2]
mkhan@instance-2:~/hadoop$ bin/hdfs dfs -mkdir /user
mkhan@instance-2:~/hadoop$ bin/hdfs dfs -mkdir /user/mkhan
mkhan@instance-2:~/hadoop$ bin/hdfs dfs -mkdir /user/mkhan/wordcount
mkhan@instance-2:~/hadoop$ bin/hdfs dfs -mkdir /user/mkhan/wordcount/input
mkhan@instance-2:~/hadoop$ bin/hdfs dfs -mkdir /user/mkhan/wordcount
mkdir: `/user/mkhan/wordcount': File exists
mkhan@instance-2:~/hadoop$ bin/hdfs dfs -mkdir /user/mkhan/wordcount/output
mkhan@instance-2:~/hadoop$ bin/hdfs dfs -put ../wordcount/input/* /user/mkhan/wo
rdcount/input
mkhan@instance-2:~/hadoop$ bin/hadoop fs -ls /user/mkhan/wordcount/input/
Found 2 items
-rw-r--r--   1 mkhan supergroup         22 2022-09-21 22:28 /user/mkhan/wordcoun
t/input/file01
-rw-r--r--   1 mkhan supergroup         28 2022-09-21 22:28 /user/mkhan/wordcoun
t/input/file02
mkhan@instance-2:~/hadoop$ bin/hadoop fs -cat /user/mkhan/wordcount/input/file01
Hello World Bye World
mkhan@instance-2:~/hadoop$
```

I already create our source file into {user root}/wordcount/src directory. Now I'm going to compiling my code with hadoop and java

> cd ../wordcount/src

> ../../hadoop/bin/hadoop com.sun.tools.javac.Main WordCount.java

Now creating "jar" package to run our program into hadoop.

> jar cf wc.jar WordCount*.class

Here we executing our code by hadoop!

> ../../hadoop/bin/hadoop jar wc.jar WordCount /user/mkhan/wordcount/input /user/mkhan/wordcount/output

We all done! Now displaying output and stop our hadoop instance.

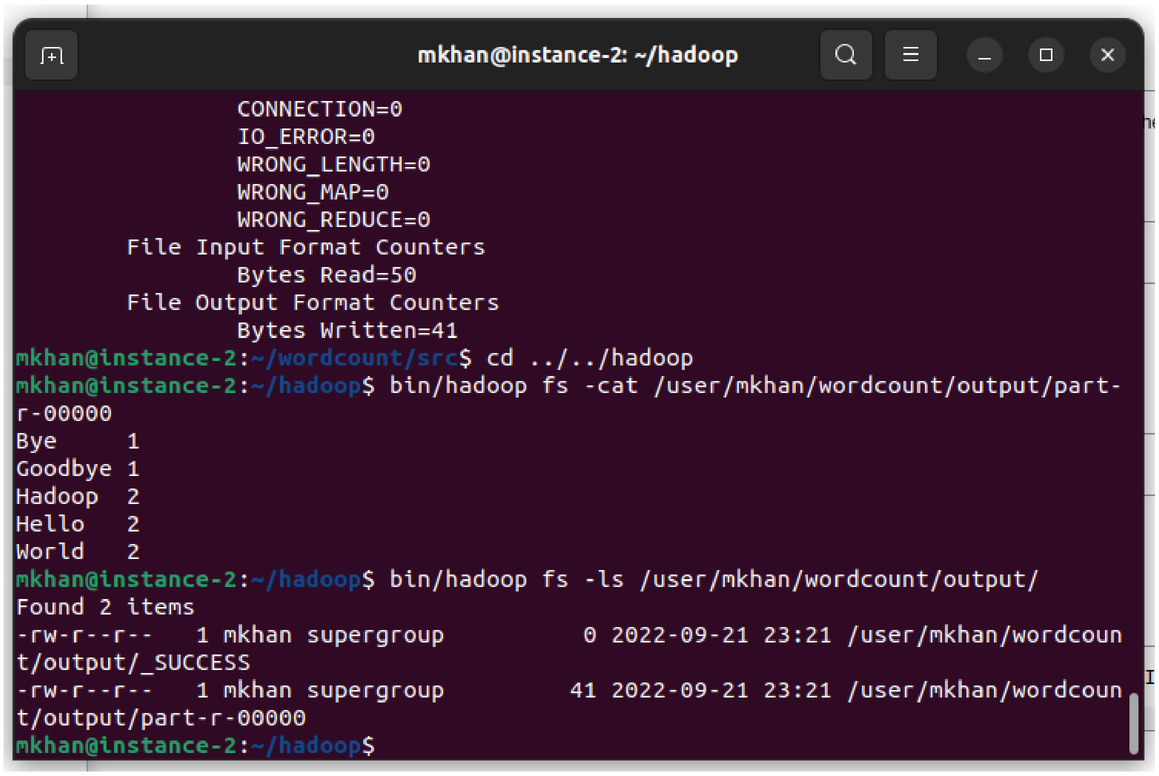
> cd ../../hadoop

> bin/hadoop fs -cat /user/mkhan/wordcount/output/part-r-00000

> bin/hadoop fs -ls /user/mkhan/wordcount/output/

> sbin/stop-dfs.sh

And here is the final result that created by hadoop process...



```
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=50
        File Output Format Counters
                Bytes Written=41
mkhan@instance-2:~/wordcount/src$ cd ../../hadoop
mkhan@instance-2:~/hadoop$ bin/hadoop fs -cat /user/mkhan/wordcount/output/part-
r-00000
Bye     1
Goodbye 1
Hadoop  2
Hello   2
World   2
mkhan@instance-2:~/hadoop$ bin/hadoop fs -ls /user/mkhan/wordcount/output/
Found 2 items
-rw-r--r--   1 mkhan supergroup          0 2022-09-21 23:21 /user/mkhan/wordcoun
t/output/_SUCCESS
-rw-r--r--   1 mkhan supergroup         41 2022-09-21 23:21 /user/mkhan/wordcoun
t/output/part-r-00000
mkhan@instance-2:~/hadoop$
```

Thank You