

# How to setup jupyter Notebook on Google Cloud Platform



Sinan Artun · [Follow](#)

5 min read · Sep 13, 2020

Listen

Share



## Step 1: Add a VM instance

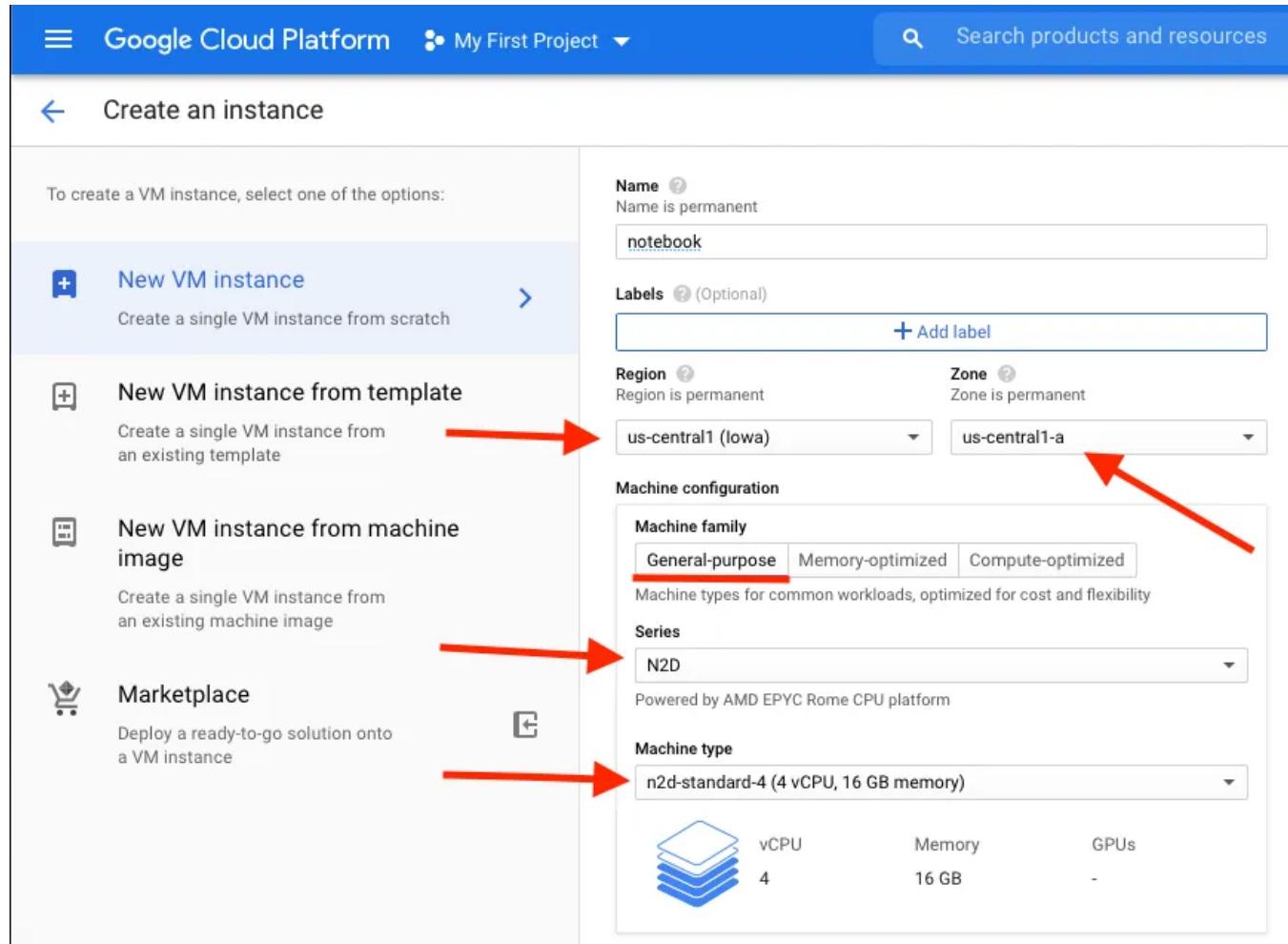
On Google Cloud Platform [Home page](#) Click  
Compute Engine → VM instances

The screenshot shows the Google Cloud Platform Home page. A red arrow points from the top-left corner to the 'Compute Engine' link in the sidebar. Another red arrow points from the 'Compute Engine' link down to the 'VM instances' option in the dropdown menu.

Click Create button

The screenshot shows the 'VM instances' page under the 'Compute Engine' section. A red arrow points to the 'Create' button at the bottom right of the main content area.

firstly give your instance a name I write “notebook” for this instance. If you are an AMD fan select us-central1 (lawa) to be able to choose the N2D cpu series, for this project I really need some compute power so I choose **n2d-standard-4** which has 4 cores and 16 GB memory



on Boot disk part click the change button

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** ?  
Name is permanent

**Labels** (Optional) [+ Add label](#)

**Region** ?  
Region is permanent

**Zone** ?  
Zone is permanent

**Machine configuration**

**Machine family** [General-purpose](#) [Memory-optimized](#) [Compute-optimized](#)  
Machine types for common workloads, optimized for cost and flexibility

**Series**   
Powered by AMD EPYC Rome CPU platform

**Machine type**

	vCPU 4	Memory 16 GB	GPUs -
---	-----------	-----------------	-----------

**CPU platform and GPU**

**Confidential VM service** ?  
 Enable the Confidential Computing service on this VM instance.

**Container** ?  
 Deploy a container image to this VM instance. [Learn more](#)

**Boot disk** ?

 New 15 GB SSD persistent disk  
Image  
Debian GNU/Linux 10 (buster) [Change](#)

**Identity and API access** ?

**Service account** ?

**Access scopes** ?  
 Allow default access  
 Allow full access to all Cloud APIs  
 Set access for each API

for this project, I want to select Ubuntu 20.04 LTS and SSD persistent disk with 20GB volume.

The screenshot shows the 'Create an instance' wizard in the Google Cloud Platform. On the left sidebar, there are options like 'New VM instance', 'New VM instance from template', 'New VM instance from machine image', and 'Marketplace'. The main area is titled 'Boot disk' and contains the following fields:

- PUBLIC IMAGES** tab is selected.
- Operating system**: Ubuntu
- Version \***: Ubuntu 20.04 LTS
- Boot disk type \***: SSD persistent disk
- Size (GB) \***: 20

A red arrow points to the 'SELECT' button at the bottom left of the configuration panel.

on Firewall section select Allow HTTP traffic and Allow HTTPS traffic

You are ready to go, click the create button

## Boot disk

Name	instance-1
Type	New SSD persistent disk
Size	20 GB
Image	 Ubuntu 20.04 LTS

[CHANGE](#)

## Identity and API access

### Service accounts

Service account

Compute Engine default service account



### Access scopes

- Allow default access
- Allow full access to all Cloud APIs
- Set access for each API

## Firewall

Add tags and firewall rules to allow specific network traffic from the Internet

- Allow HTTP traffic
- Allow HTTPS traffic



### NETWORKING, DISKS, SECURITY, MANAGEMENT, SOLE-TENANCY

Your free trial credit will be used for this VM instance. [GCP Free Tier](#)

[CREATE](#)

CANCEL

EQUIVALENT COMMAND LINE



After your VM instance setup finished you will be redirected to the VM instances page which looks like this

Name	Zone	Recommendation	In use by	Internal IP	External IP	Connect
notebook	us-central1-a			10.128.0.2 (nic0)	34.122.224.36	SSH

## Step 2: Open port on firewall

I want to use port 60000 for this notebook so I will add a firewall rule

Click your instance

## click network

The screenshot shows the Google Cloud Platform VM instance details page for a notebook instance. The left sidebar lists various Compute Engine options. The main pane displays the instance details for 'notebook'. It includes sections for Remote access (SSH), Logs (Cloud Logging, Serial port 1 (console)), Instance ID (47210759439430320), Machine type (n2d-standard-4 (4 vCPUs, 16 GB memory)), Reservation (Automatically choose), CPU platform (AMD Rome), Display device (Turn on display device), Zone (us-central1-a), Labels (None), Creation time (Sep 11, 2020, 7:34:25 AM), and Network interfaces.

Name	Network	Subnetwork	Primary internal IP	Alias IP ranges	External IP	Network Tier	IP forwarding	Network details
nic0	default	default	10.128.0.2	—	34.122.224.36 (ephemeral)	Premium	Off	<a href="#">View details</a>

Below the table, there are sections for Public DNS PTR Record (None), Firewalls (Allow HTTP traffic, Allow HTTPS traffic checked), Network tags (http-server, https-server), and Deletion protection (Enable deletion protection).

Click Firewall

VPC network details page showing the 'default' network configuration. The 'Firewall' section is highlighted with a red arrow. The 'PERMISSIONS' section shows a message: 'Please select at least one resource.'

## Click Create Firewall Rule

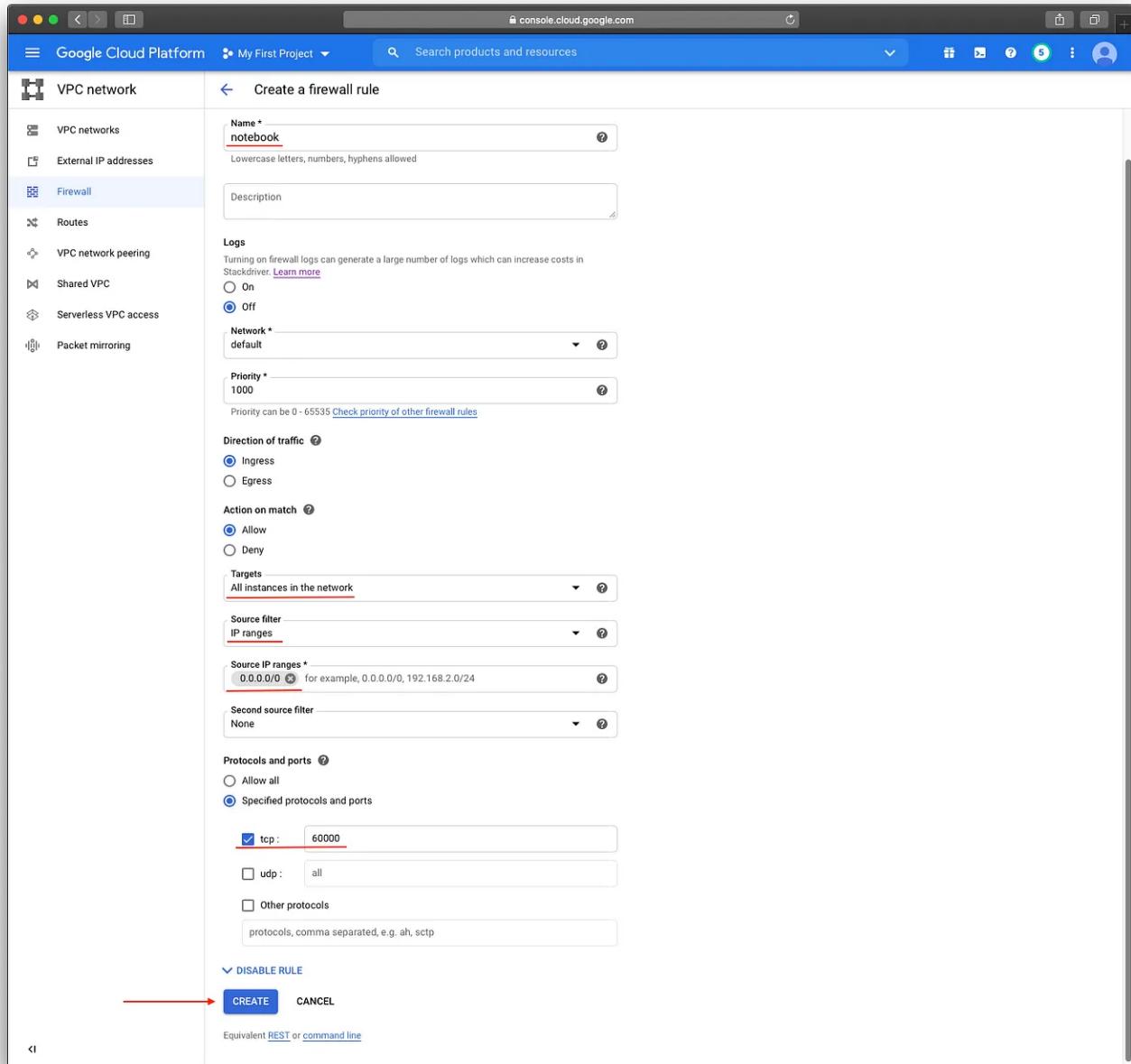
Firewall rules control incoming or outgoing traffic to an instance. By default, incoming traffic from outside your network is blocked. [Learn more](#)

Note: App Engine firewalls are managed [here](#).

Name	Type	Targets	Filters	Protocols / ports	Action	Priority	Network	Logs
default-allow-http	Ingress	http-server	IP ranges: 0.0.0.0/0	tcp:80	Allow	1000	default	Off
default-allow-https	Ingress	https-server	IP ranges: 0.0.0.0/0	tcp:443	Allow	1000	default	Off
default-allow-icmp	Ingress	Apply to all	IP ranges: 0.0.0.0/0	icmp	Allow	65534	default	Off
default-allow-internal	Ingress	Apply to all	IP ranges: 10.0.0.0/16	tcp:0-65535 udp:0-65535 icmp	Allow	65534	default	Off
default-allow-rdp	Ingress	Apply to all	IP ranges: 0.0.0.0/0	tcp:3389	Allow	65534	default	Off
default	Ingress	Apply to all	IP ranges: 0.0.0.0/0	tcp:22	Allow	65534	default	Off

- give firewall rule a name. I write “notebook”
- targets: All instances in the network
- Source filter: Ip ranges
- Source IP ranges : 0.0.0.0/0

- second source filter : **none**
- Specified protocols and ports: **tcp:60000**



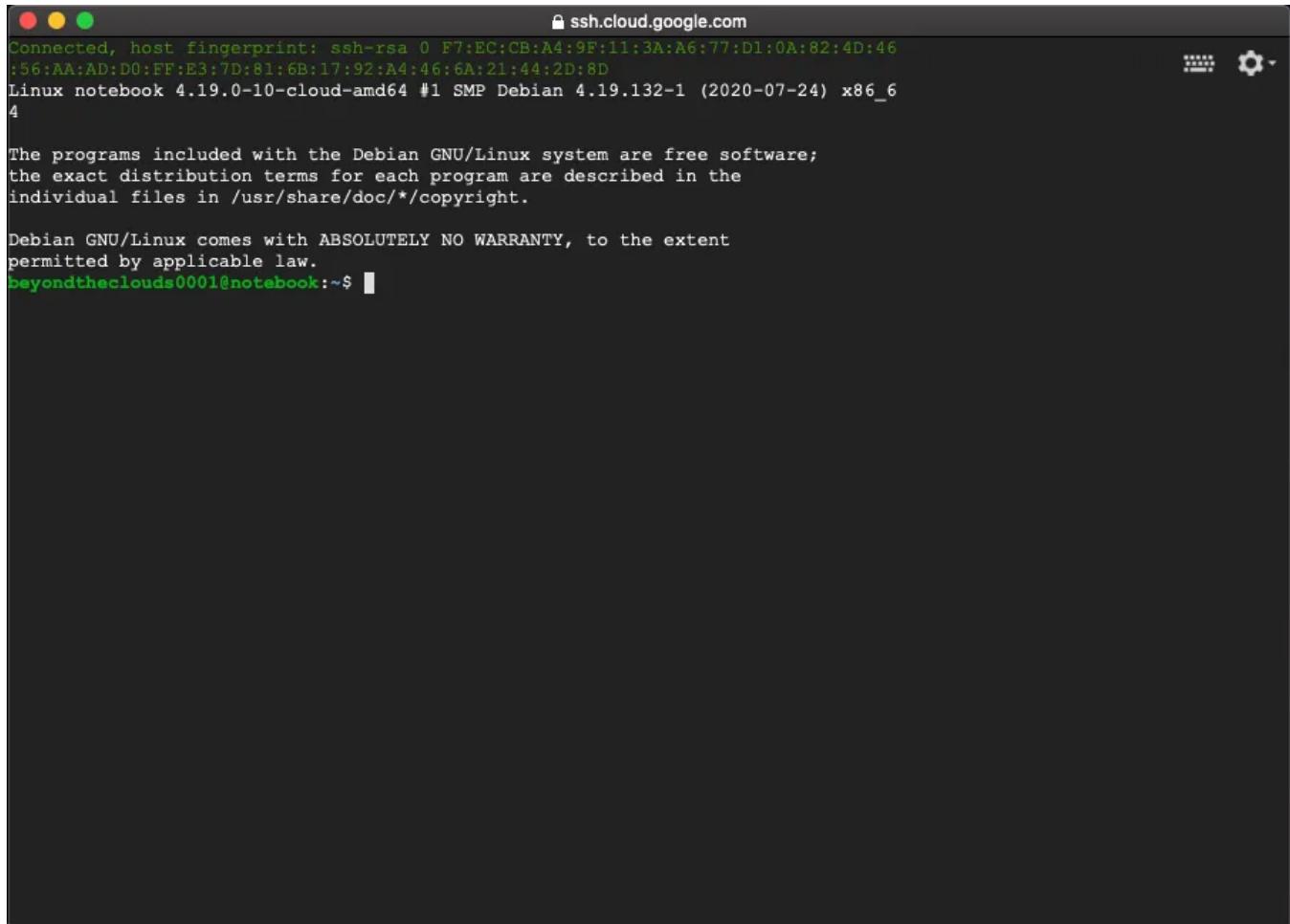
and click the **create** button. Now you can connect your notebook from **tcp:60000**.

### Step 3: Install Jupyter Notebook

On the VM instances dashboard click the **SSH** button to connect your VM instance

The screenshot shows the Google Cloud Platform Compute Engine interface. On the left sidebar, under 'Compute Engine', there are several options: VM instances, Instance groups, Instance templates, Sole-tenant nodes, Machine images, Disks, Snapshots, Images, TPUs, Migrate for Compute Engine, and Committed use discounts. The 'VM instances' option is selected. In the main area, a table lists VM instances. One row is selected, showing the instance name 'notebook', zone 'us-central1-a', internal IP '10.128.0.2 (nic0)', external IP '34.122.224.36', and an 'SSH' button with a dropdown menu. Below the table, there's a section titled 'Related Actions' with links to View Billing Report, Monitor VMs, Explore VM Logs, Setup Firewall Rules, and Patch Management.

now your browser will open another window which is your console



you need to enter all commands in this console let's start with an upgrade

```
sudo apt update && sudo apt upgrade -y
```

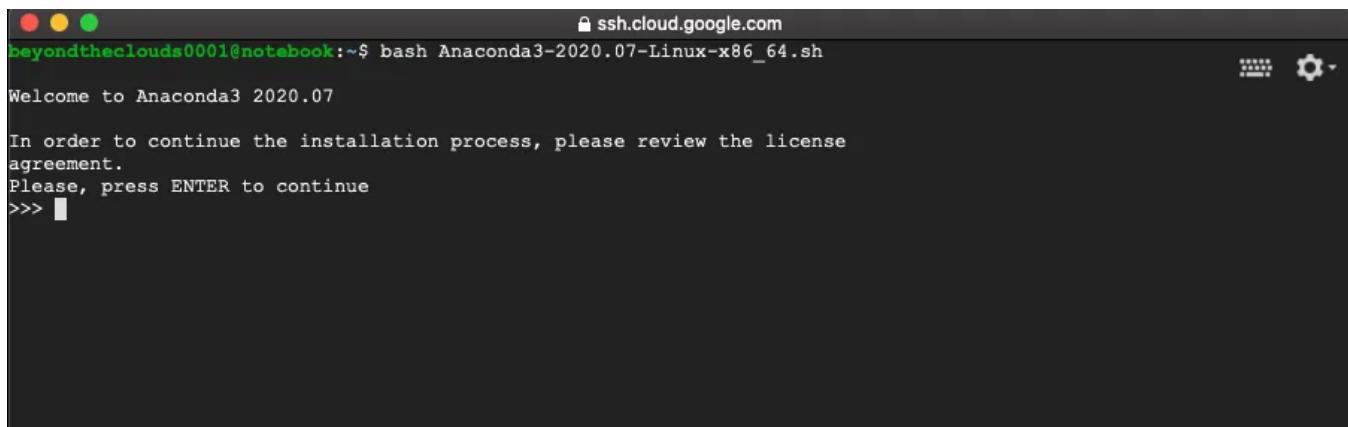
now visit this [URL](#) and find the latest release of Anaconda for Linux-x86\_64 and right-click the link and copy it.

for now, [Anaconda3-2021.11-Linux-x86\\_64.sh](https://repo.anaconda.com/archive/Anaconda3-2021.11-Linux-x86_64.sh) is the latest release. We use curl -O command to download the file

```
cd ~ && curl -O https://repo.anaconda.com/archive/Anaconda3-2021.11-Linux-x86\_64.sh
```

this is a .sh file so we use the bash command to run it

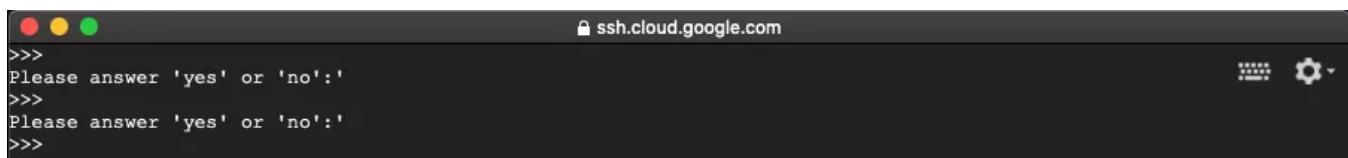
```
bash Anaconda3-2021.11-Linux-x86\_64.sh
```



```
ssh.cloud.google.com
beyondtheclouds0001@notebook:~$ bash Anaconda3-2020.07-Linux-x86_64.sh
Welcome to Anaconda3 2020.07

In order to continue the installation process, please review the license
agreement.
Please, press ENTER to continue
>>> █
```

hold ENTER till you see yes or no choice and write yes



```
ssh.cloud.google.com
>>>
Please answer 'yes' or 'no':'
>>>
Please answer 'yes' or 'no':'
>>>
```

hit the ENTER for the last time to confirm installation location and your installation will start

```
>>>
Please answer 'yes' or 'no':'
>>> yes

Anaconda3 will now be installed into this location:
/home/beyondtheclouds0001/anaconda3

- Press ENTER to confirm the location
- Press CTRL-C to abort the installation
- Or specify a different location below

[/home/beyondtheclouds0001/anaconda3] >>>
```

After installation finished it will ask to run initialize Anaconda3 say yes

```
widgetsnbextension pkgs/main/linux-64::widgetsnbextension-3.5.1-py38_0
wrapt          pkgs/main/linux-64::wrapt-1.11.2-py38h7b6447c_0
wurlitzer      pkgs/main/linux-64::wurlitzer-2.0.1-py38_0
xlrd           pkgs/main/noarch::xlrd-1.2.0-py_0
xlsxwriter     pkgs/main/noarch::xlsxwriter-1.2.9-py_0
xlwt            pkgs/main/linux-64::xlwt-1.3.0-py38_0
xmltodict      pkgs/main/noarch::xmltodict-0.12.0-py_0
xz              pkgs/main/linux-64::xz-5.2.5-h7b6447c_0
yaml            pkgs/main/linux-64::yaml-0.2.5-h7b6447c_0
yapf            pkgs/main/noarch::yapf-0.30.0-py_0
zeromq          pkgs/main/linux-64::zeromq-4.3.2-he6710b0_2
zict            pkgs/main/noarch::zict-2.0.0-py_0
zipp            pkgs/main/noarch::zipp-3.1.0-py_0
zlib            pkgs/main/linux-64::zlib-1.2.11-h7b6447c_3
zope            pkgs/main/linux-64::zope-1.0-py38_1
zope.event      pkgs/main/linux-64::zope.event-4.4-py38_0
zope.interface pkgs/main/linux-64::zope.interface-4.7.1-py38h7b6447c_0
zstd             pkgs/main/linux-64::zstd-1.4.5-h0b5b093_0

Preparing transaction: done
Executing transaction: done
installation finished.
Do you wish the installer to initialize Anaconda3
by running conda init? [yes|no]
[no] >>> yes
```

**! important : close current shell and reconnect with ssh. Its because jupyter command not recognized with current ssh session.**

now we need to generate a jupyter notebook configuration file with this command

```
jupyter notebook --generate-config
```

After generating the config file command you will see the path of the jupyter\_notebook\_config.py file copy this path and edit it with nano file editor

```
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.
Last login: Fri Sep 11 21:15:17 2020 from 35.235.240.1
(base) beyondtheclouds0001@notebook:~$ jupyter notebook --generate-config
Writing default config to: /home/beyondtheclouds0001/.jupyter/jupyter_notebook_config.py
(base) beyondtheclouds0001@notebook:~$
```

`nano /home/beyondtheclouds0001/.jupyter/jupyter_notebook_config.py`

```
GNU nano 3.2 /home/beyondtheclouds0001/.jupyter/jupyter_notebook_config.py
# Configuration file for jupyter-notebook.

# Application(SingletonConfigurable) configuration
#-----

## This is an application.

## The date format used by logging formatters for %(asctime)s
#c.Application.log_datefmt = '%Y-%m-%d %H:%M:%S'

## The Logging format template
#c.Application.log_format = '[%(name)s]%(highlevel)s %(message)s'

## Set the log level by value or name.
#c.Application.log_level = 30

#-----
# JupyterApp(Application) configuration
#-----


## Base class for Jupyter applications

## Answer yes to any prompts.
#c.JupyterApp.answer_yes = False
```

[ Read 879 lines ]

^G Get Help    ^O Write Out    ^W Where Is    ^K Cut Text    ^J Justify    ^C Cur Pos    M-U Undo  
 ^X Exit    ^R Read File    ^\ Replace    ^U Uncut Text    ^T To Spell    ^G Go To Line    M-E Redo    M-A Mark Text  
 M-6 Copy Text

you can simply copy and paste these lines at the beginning of the configuration file

```
c.NotebookApp.token = ''
c.NotebookApp.ip = '*'
c.NotebookApp.open_browser = False
c.NotebookApp.port = 60000
```

```

GNU nano 3.2 /home/beyondtheclouds0001/.jupyter/jupyter_notebook_config.py

c.NotebookApp.token = ''
c.NotebookApp.ip = '*'
c.NotebookApp.open_browser = False
c.NotebookApp.port = 60000
# Configuration file for jupyter-notebook.

# Application(SingletonConfigurable) configuration

## This is an application.

## The date format used by logging formatters for %(asctime)s
#c.Application.log_datefmt = '%Y-%m-%d %H:%M:%S'

## The Logging format template
#c.Application.log_format = '[%(name)s %(highlevel)s %(message)s'

^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify    ^C Cur Pos
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell  ^_ Go To Line

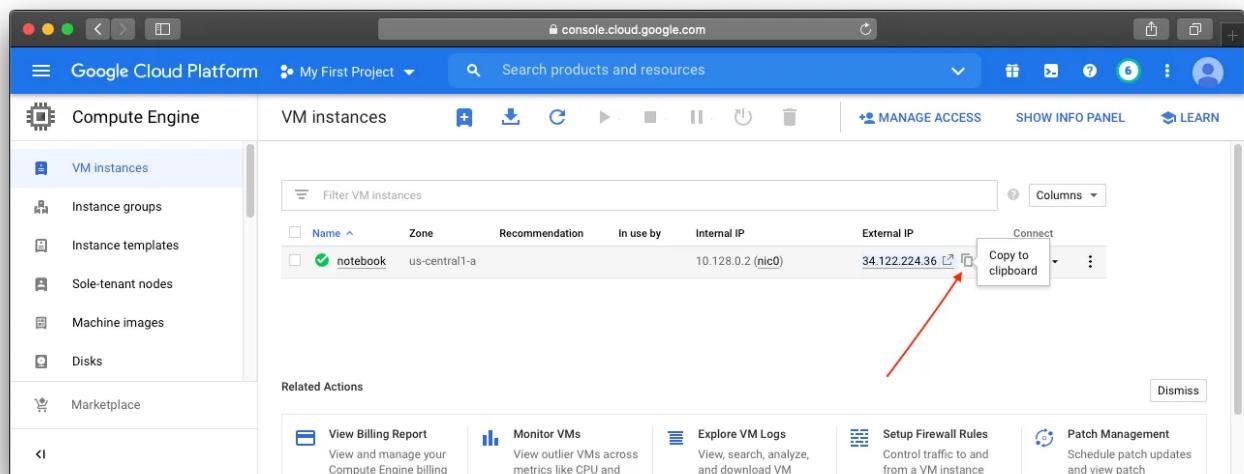
```

control + x to exit and press Y to save changes then press enter.

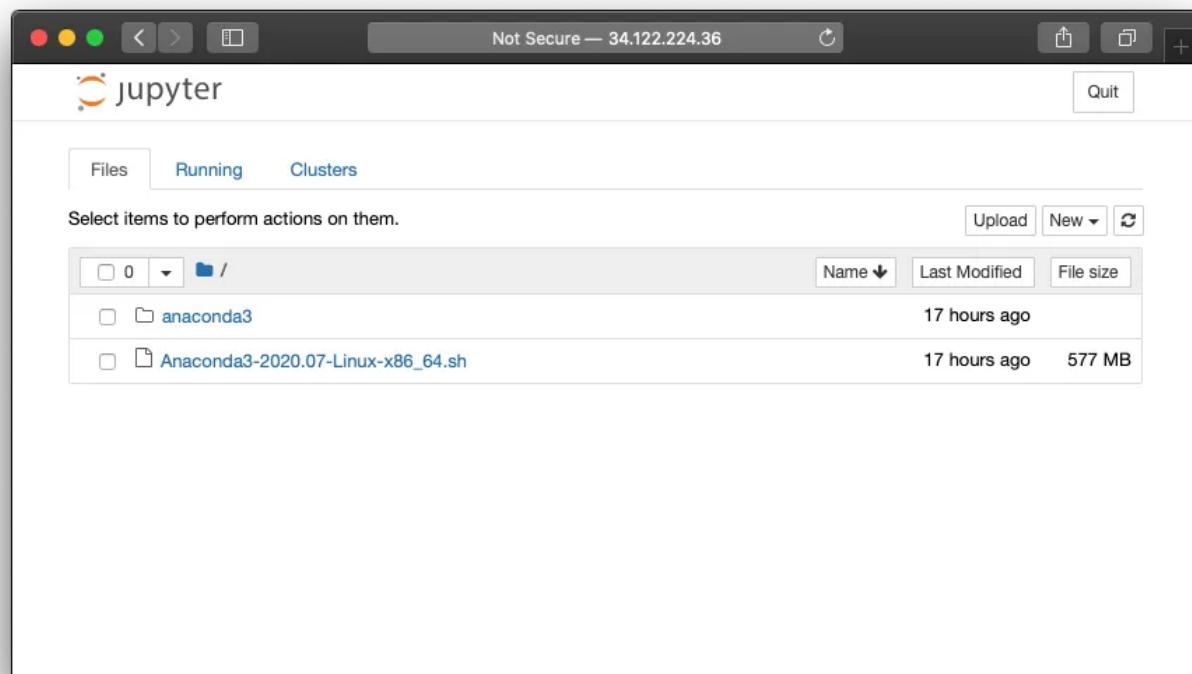
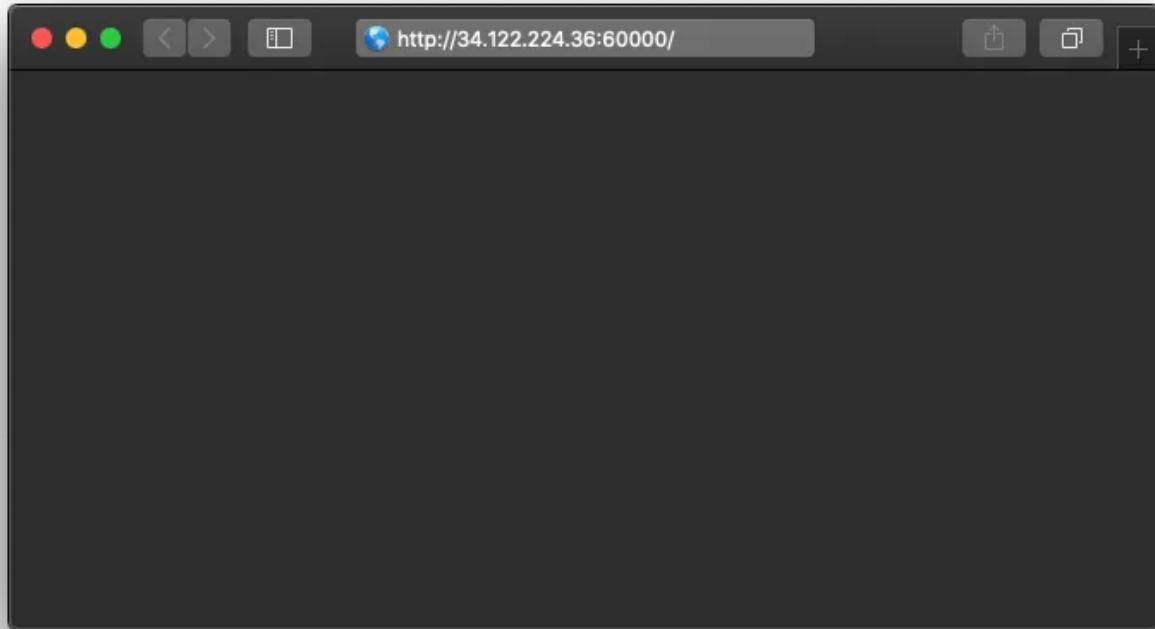
you are ready to go. Just start jupyter notebook with this command

jupyter-notebook

Go to VM instances dashboard and copy your instance external IP address



open a new browser and enter your IP address and port number as it is

[Open in app ↗](#)[Sign up](#)[Sign In](#)

process constantly run on this virtual machine you need to convert this process to a Linux service.



Add a new file under the “/etc/systemd/system/” directory  
this file name will be “jupyter.service”.

```
nano /etc/systemd/system/jupyter.service
```

change “**beyondtheclouds0001**” parts then copy-paste codes below to your notebook service file and save the file.

Note: in this example, my username on Linux is “beyondtheclouds0001”

### **where is my jupyter-notebook bin folder?**

it is under the anaconda3 folder where you download [Anaconda3-2021.11-Linux-x86\\_64.sh](#) and run it. It auto-makes a folder in the same directory. In this example, I downloaded Anaconda to my home directory which is /home/beyondtheclouds0001/

```
[Unit]
Description=Jupyter Notebook

[Service]
Type=simple
PIDFile=/run/jupyter.pid
ExecStart=/home/beyondtheclouds0001/anaconda3/bin/jupyter-notebook -
--
config=/home/beyondtheclouds0001/.jupyter/jupyter_notebook_config.py
User=beyondtheclouds0001
Group=google-sudoers
WorkingDirectory=/home/beyondtheclouds0001/
Restart=always
RestartSec=10

[Install]
WantedBy=multi-user.target
```

so in the end, there will be a “/etc/systemd/system/jupyter.service” file

we need to reload daemon so linux system will see our new jupyter.service file

```
sudo systemctl daemon-reload
```

we need to make this service started when Linux makes a restart so run this command

```
sudo systemctl enable jupyter
```

start jupyter service

```
sudo systemctl start jupyter
```

check if jupyter service is running

```
sudo systemctl status jupyter
```

```
jupyter.service - Jupyter Notebook
  Loaded: loaded (/etc/systemd/system/jupyter.service;
             Active: active (running) since Sat 2022-02-12 21:30:0
    Main PID: 4873 (jupyter-noteboo)
      Tasks: 15 (limit: 4698)
     Memory: 104.3M
        CPU: 0.000 CPU(s) since start
       CGroup: /system.slice/jupyter.service
```

these commands make your notebook service as an initial service and when you start your virtual machine notebook service will start automatically.

Do not hesitate to ask questions. keep coding 😊