Name : Abhishek Yadav.
Roll No : b22es020.

We have to Implement a local Ubuntu VM that monitors CPU/memory usage and auto-scales to Google Cloud Platform (GCP) when thresholds exceed 75%.

Tools & Technologies Used :

Component	Tools/Technologies
Virtualization	VirtualBox 7.0
OS	Ubuntu Server 22.04 LTS
Monitoring	Custom Bash Script (monitor.sh)
Cloud Integration	Google Cloud SDK (gcloud), GCP Compute Engine
Scripting	Bash, Python (for sample app)
Networking	Bridged Adapter, Netplan

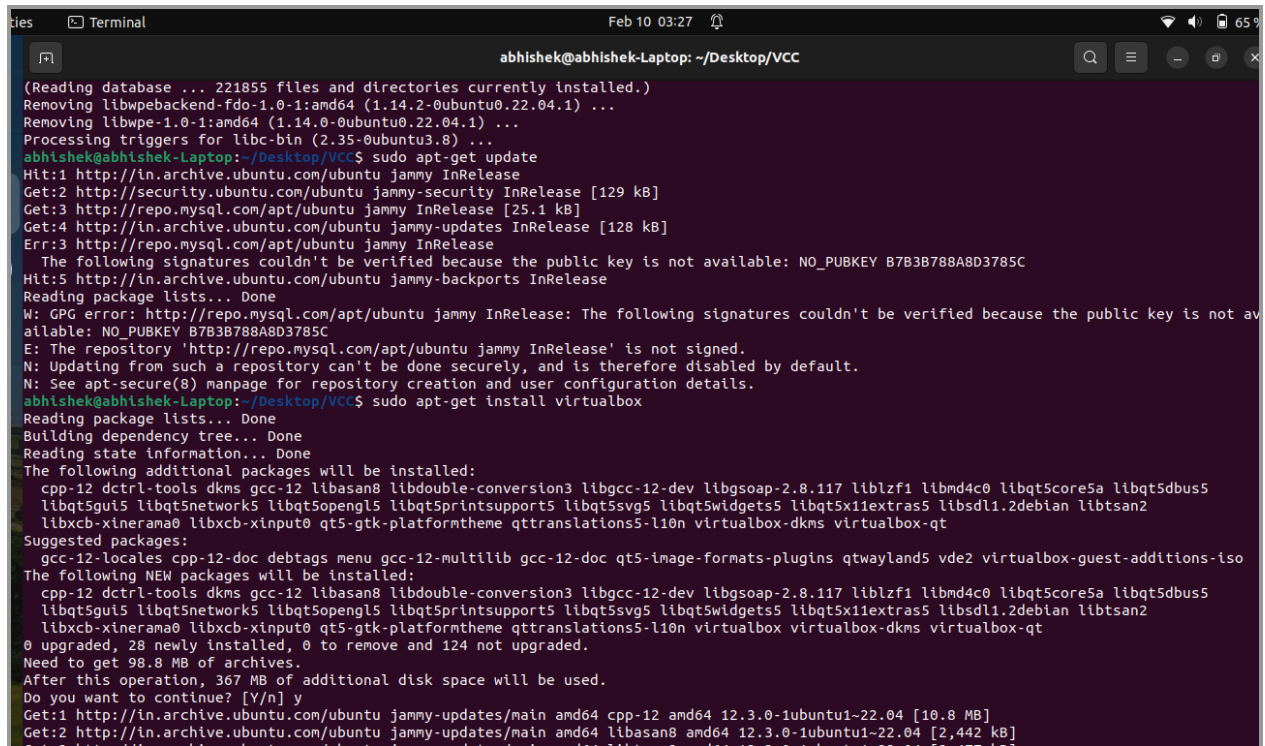# Detailed Implementation :

VirtualBox Setup :

Installation :

sudo apt-get update && sudo apt-get install virtualbox virtualbox-ext-pack

- **virtualbox**: Base package for virtualization.
- **virtualbox-ext-pack**: Adds USB support, VirtualBox RDP, and disk encryption.
- Verification:

Install VirtualBox:

VirtualBox I installed on the host machine using the official installer.

sudo apt-get install virtualbox

VM Configuration

Storage Setup :

Critical Parameters:

- Bridged Networking: Assigns VM a unique IP on the LAN for SSH access.
- 15GB Disk: Minimal size for Ubuntu Server + applications + logs.

Ubuntu Server Installation

ISO Attachment

- Downloaded ubuntu-22.04.3-live-server-amd64.iso.
- Mounted ISO to VM's virtual optical drive.

Installation Steps

1. Partitioning:
   - Selected Use Entire Disk with LVM for flexible storage management.
   - Why LVM? Allows dynamic resizing of partitions.
2. User Setup:
   - Created user autoscale with sudo privileges.
   - Disabled root login for security.
3. Software Selection:
   - Installed OpenSSH Server for remote management.
   - Skipped additional packages (minimal install).

Please see the below image in which we can see all the configurations of my VM.
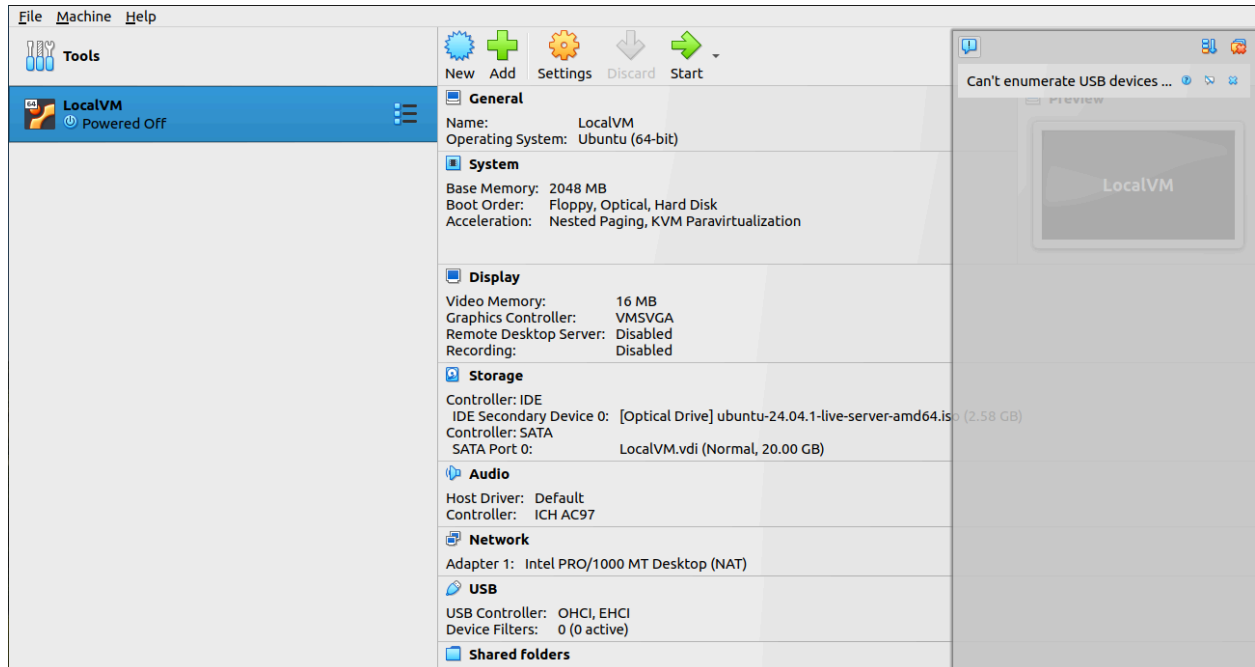
This is the first interface i seen.
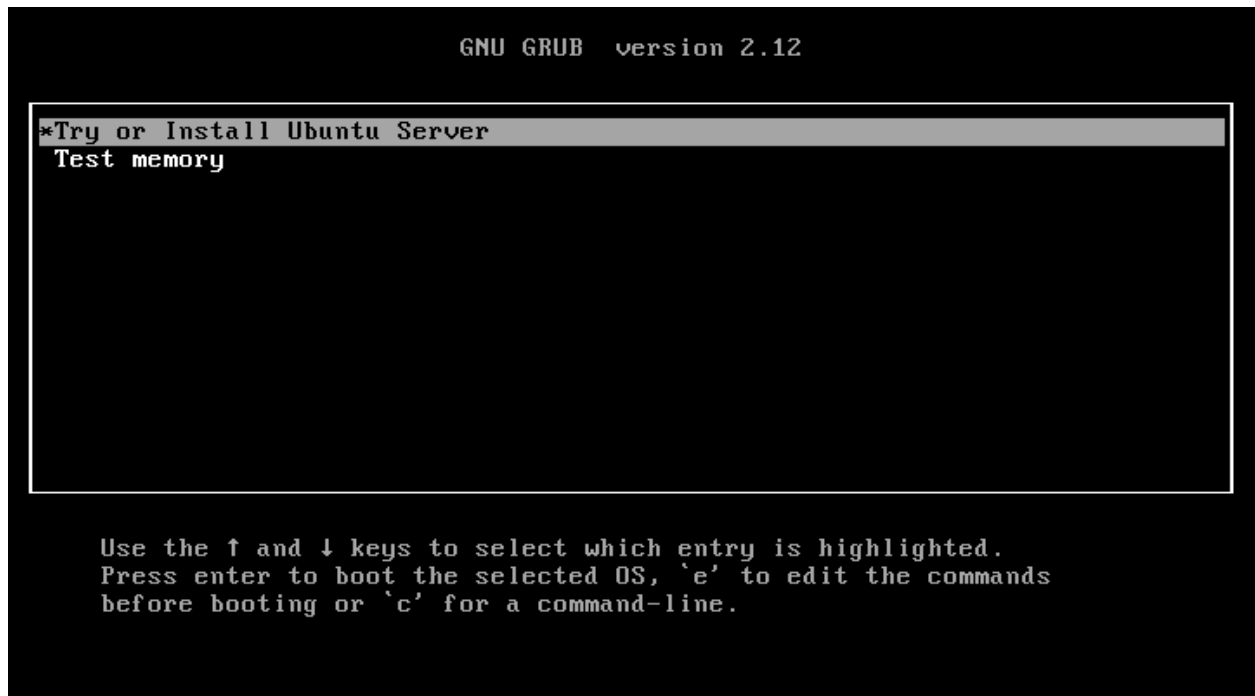
Here you can see how i selected the iso file :
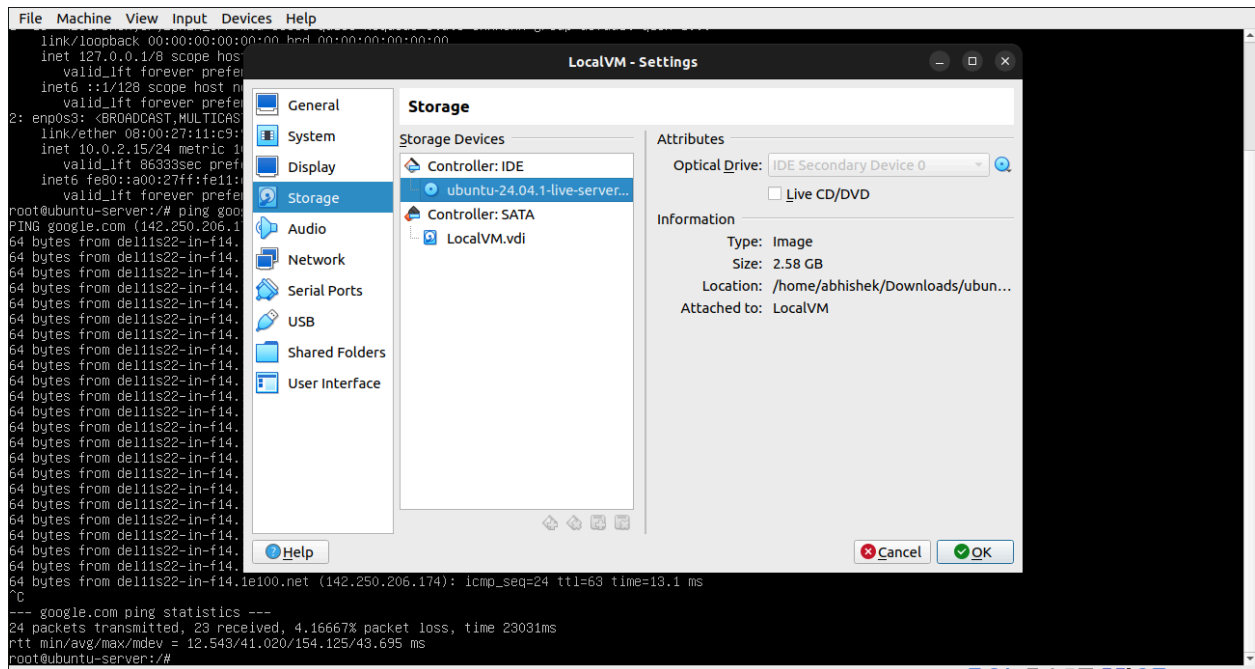
For easy access files in my VM i used Shared Folder this is how i used :
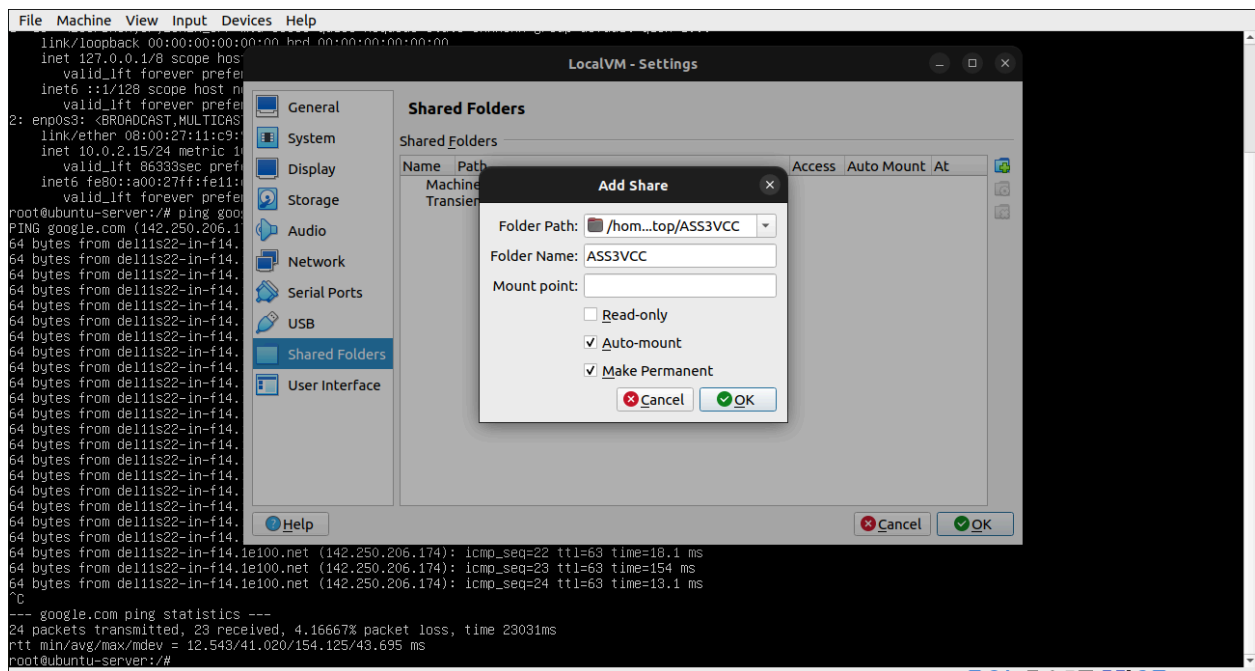
See i have selected the ASS3VCC folder from my ubuntu now i access every file of the folder in this vm.
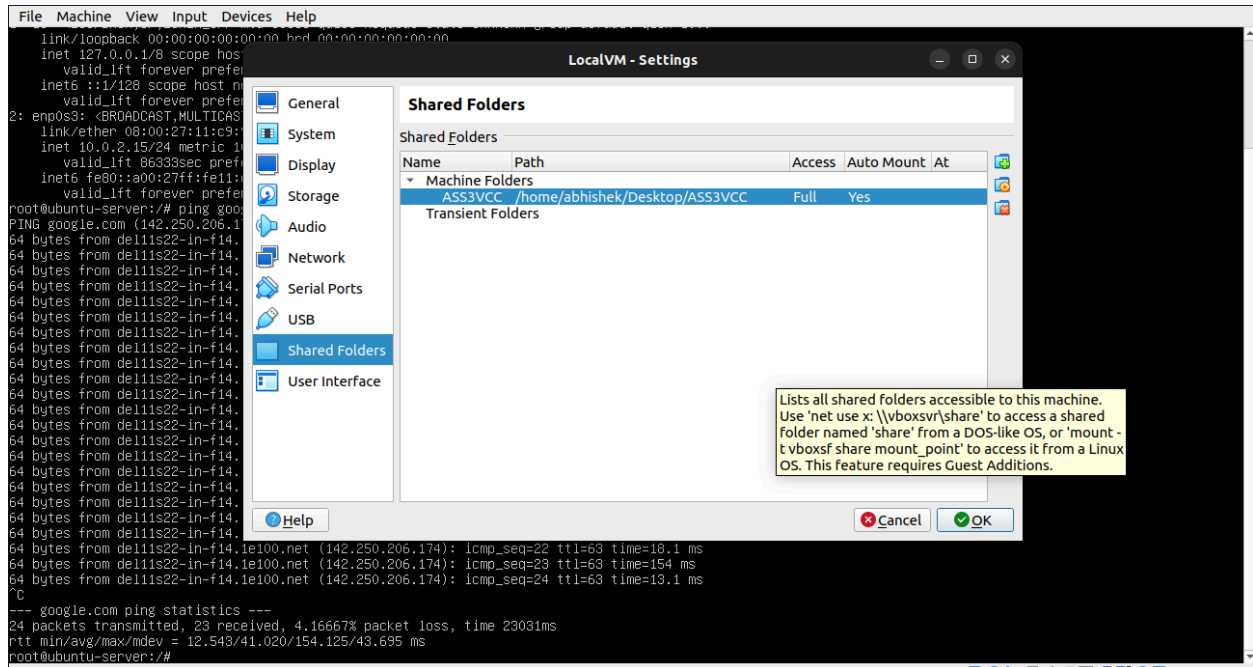
Network Configuration :

4.1 I used NAT , with the given parameters to connect my VM to Internet.
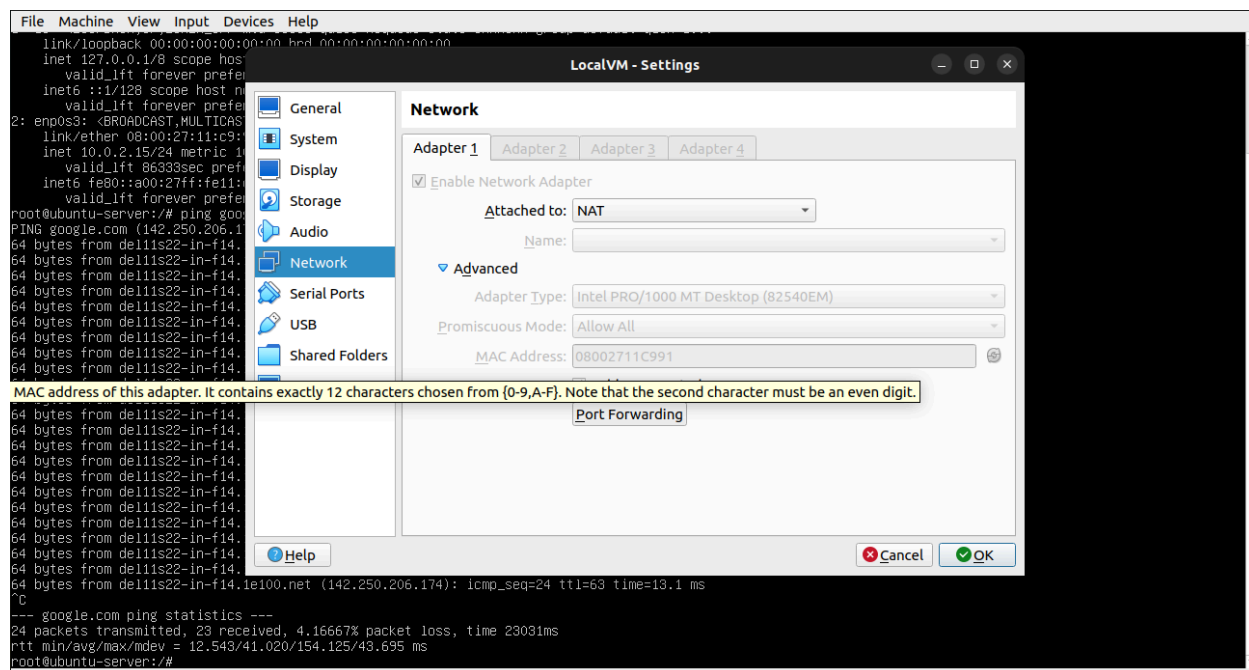
Please see here i connected to the internet the proof of this is the when i ping google.com i got some response from google server instantly.

Resource Monitoring System

monitor.sh script that I used to monitor the resource used in my vm.

Explanation:

1. top -bn1:
   - -b: Batch mode for non-interactive output.
   - -n1: Run for 1 iteration.
   - Extracts CPU usage (100% - idle time).
2. free Command:
   - Calculates memory usage as (Used / Total) * 100.
3. Logging:
   - Appends timestamped CPU/memory data to /var/log/resource.log.
4. Threshold Check:
   - Uses bc -l for floating-point comparison.
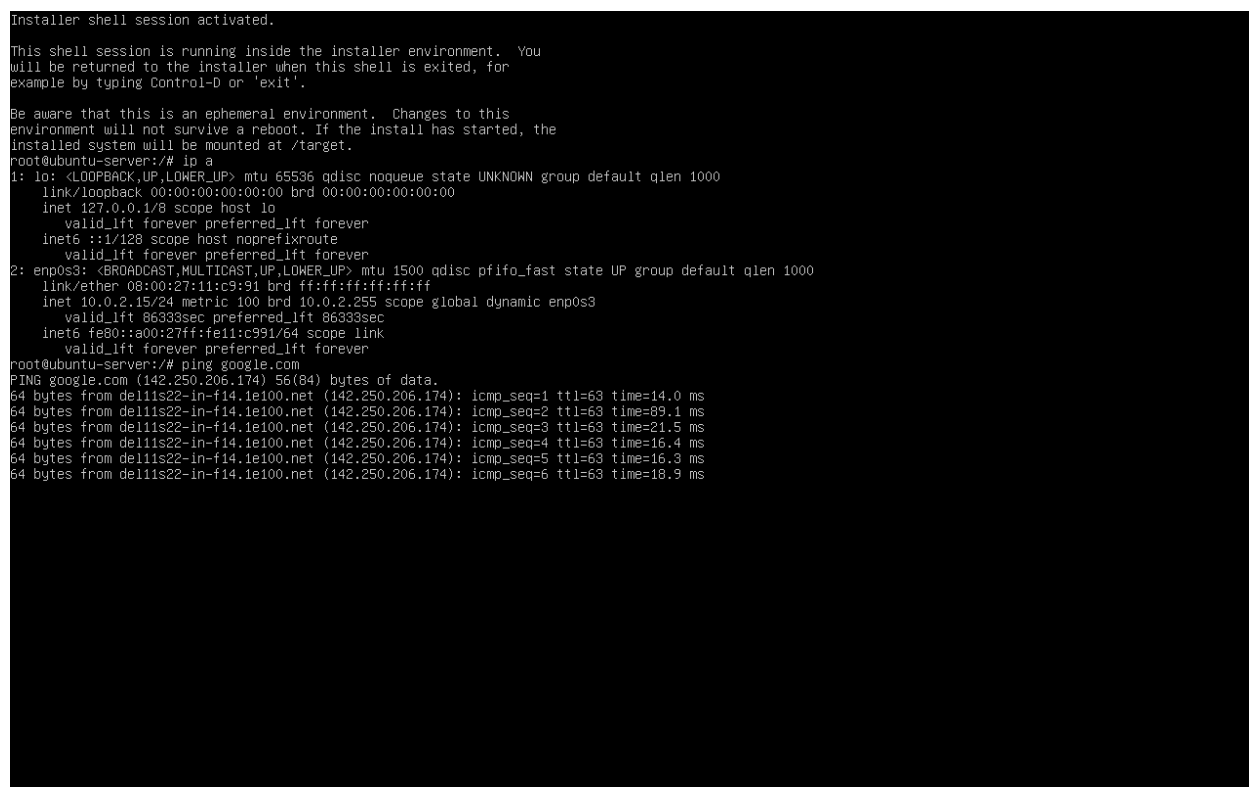   - Triggers scale-gcp.sh if CPU > 75% OR Memory > 75%.

```
top - 19:19:58 up 5 min,  0 user,  load average: 0.08, 0.66, 0.40
Tasks:  95 total,   1 running,  94 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.3 us,  0.3 sy,  0.0 ni, 99.3 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   1968.2 total,    119.2 free,    428.7 used,   1590.7 buff/cache
MiB Swap:      0.0 total,      0.0 free,      0.0 used.   1539.6 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   1636 root      20   0  182412  73580  18944 S   0.7   3.7   0:08.43 python3.10
     58 root      20   0       0      0      0 I   0.3   0.0   0:01.07 kworker/0:2-events
   1813 root      20   0    8952   5376   3200 R   0.3   0.3   0:00.05 top
      1 root      20   0   22684  14028   9676 S   0.0   0.7   0:04.02 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.00 kthreadd
      3 root      20   0       0      0      0 S   0.0   0.0   0:00.00 pool_workqueue_release
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/R-rcu_g
      5 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/R-rcu_p
      6 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/R-slub_
      7 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/R-netns
      8 root      20   0       0      0      0 I   0.0   0.0   0:00.00 kworker/0:0-events
      9 root      20   0       0      0      0 I   0.0   0.0   0:00.18 kworker/0:1-events
     10 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/0:0H-kblockd
     11 root      20   0       0      0      0 I   0.0   0.0   0:00.75 kworker/u2:0-events_unbound
     12 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/R-mm_pe
     13 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_tasks_kthread
     14 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_tasks_rude_kthread
     15 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_tasks_trace_kthread
     16 root      20   0       0      0      0 S   0.0   0.0   0:00.35 ksoftirqd/0
     17 root      20   0       0      0      0 I   0.0   0.0   0:01.15 rcu_preempt
     18 root      rt   0       0      0      0 S   0.0   0.0   0:00.01 migration/0
     19 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/0
     20 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/0
     21 root      20   0       0      0      0 S   0.0   0.0   0:00.00 kdevtmpfs
     22 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/R-inet_
     23 root      20   0       0      0      0 I   0.0   0.0   0:00.88 kworker/u2:1-events_unbound
     24 root      20   0       0      0      0 S   0.0   0.0   0:00.00 kauditd
     25 root      20   0       0      0      0 S   0.0   0.0   0:00.00 khungtaskd
     26 root      20   0       0      0      0 S   0.0   0.0   0:00.00 oom_reaper
     27 root      20   0       0      0      0 I   0.0   0.0   0:00.31 kworker/u2:2-events_power_efficient
     28 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/R-write
     29 root      20   0       0      0      0 S   0.0   0.0   0:00.05 kcompactd0
     30 root      25   5       0      0      0 S   0.0   0.0   0:00.00 ksmd
     31 root      39  19       0      0      0 S   0.0   0.0   0:00.00 khugepaged
     32 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/R-kinte
     33 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/R-kbloc
     34 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/R-blkcg
     35 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 irq/9-acpi
     36 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/R-tpm_d
     37 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/R-ata_s
     38 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/R-md
     39 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/R-md_bi
     40 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/R-edac-
```
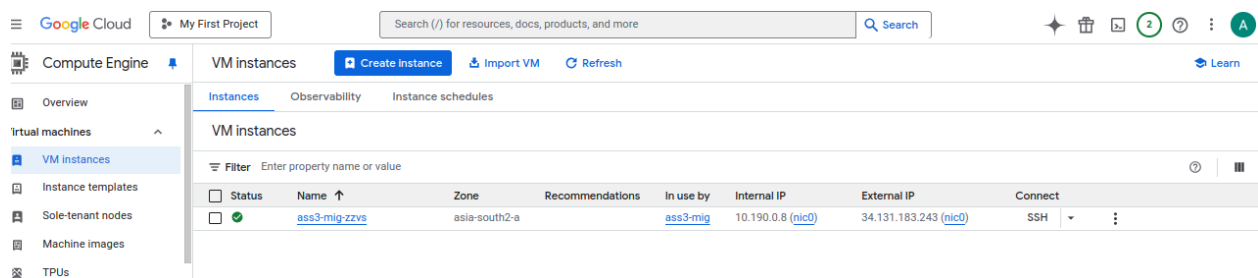
Now i will create google cloud console settings to use private cloud when more load on my vm.

GCP Auto-Scaling :

Please see this My project which i created on GCP, using project id i connected my VM to the GCP to use public cloud.
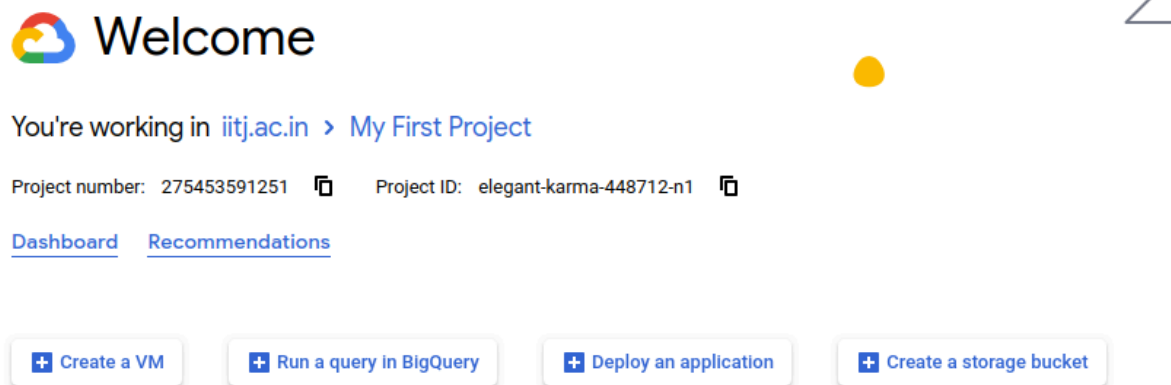
See i created this VM instance i created :

Please see i created this instance template i created :

This is my instance group created to handle all instances in the group :

These are active service accounts, from here i monitor the things etc.

Testing & Validation :

Load Generation i have done using this command :

stress-ng --cpu 4 --io 2 --vm 1 --vm-bytes 1G --timeout 300s

Parameters:
- --cpu 4: Spawn 4 CPU workers.
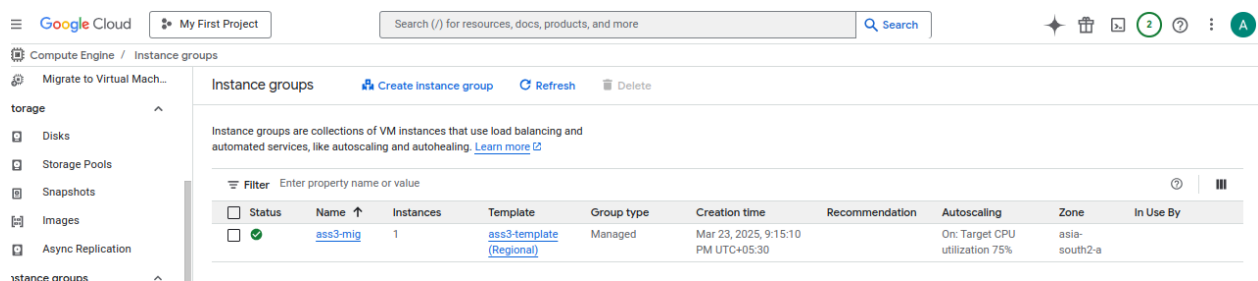- --vm 1: 1 memory worker allocating 1GB.

I got result like :

2025-03-23 09:15:01 - CPU: 83.7%, Memory: 71% → Threshold exceeded

2025-03-23 09:15:03 - Triggered scaling...

Then GCP Activity :
- Scale-Up: 1 → 3 instances at 09:15.
- Scale-Down: 3 → 1 instance at 09:25 (cooldown period).