Practical-1 Infrastructure as a service using AWS

Writeup:-

- **Cloud Computing architecture**

Cloud architecture consists of a front end and back end. The front end is the client-side interface. The back end consists of the cloud service provider's data centers, servers, storage and applications.

A central server administers the system, monitoring traffic and client demands to ensure quality of service. The underlying hardware infrastructure is distributed across various servers and locations.

- **IAAS**

Infrastructure as a Service (IaaS) provides access to fundamental computing resources such as servers, storage, networks and operating systems over the internet. The cloud provider owns and maintains the physical infrastructure and delivers these resources to customers on-demand.

Why IAAS??

- Flexibility - IaaS provides highly scalable and flexible computing resources that can be provisioned and decommissioned on-demand based on workload needs. This is useful for spiky or unpredictable workloads.
- Lower costs - With IaaS, organizations pay only for the infrastructure resources they use without having to purchase and maintain their own hardware. This eliminates capital expenditures and reduces costs.

- **AWS**

Amazon Web Services offers a broad set of global cloud-based products including compute, storage, databases, analytics, networking, mobile, developer tools, management tools, IoT, security, and enterprise applications: on-demand, available in seconds, with pay-as-you-go pricing. From data warehousing to deployment tools, directories to content delivery, over 200 AWS services are available

- EC2

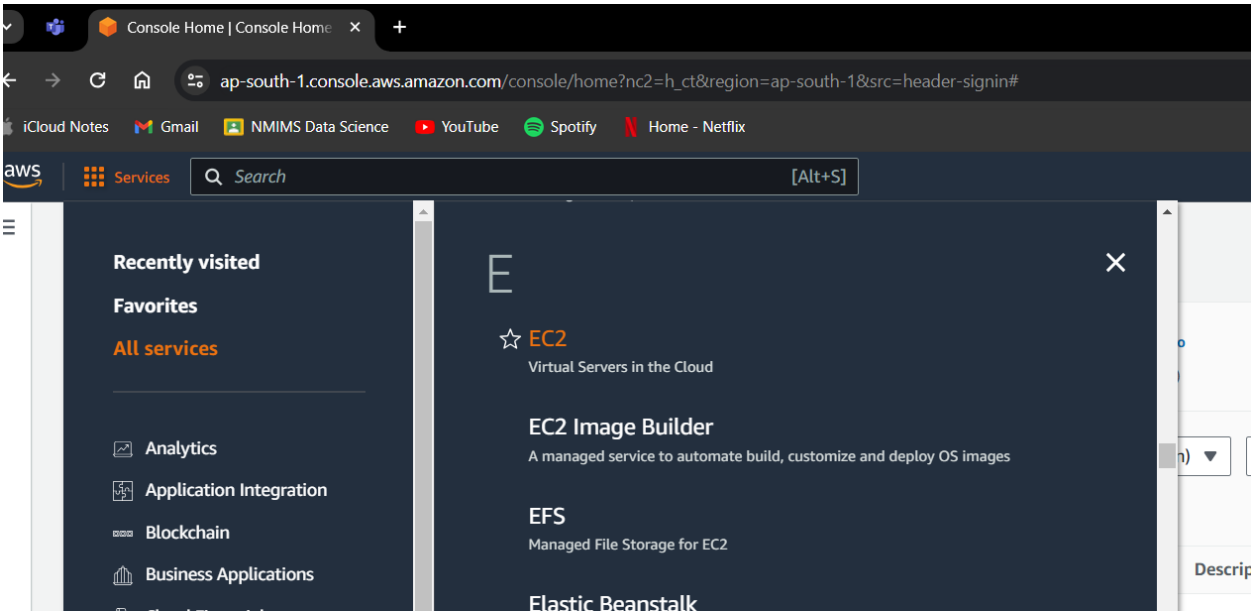
Amazon Elastic Compute Cloud (EC2) provides scalable virtual servers that can be launched and terminated on-demand. Key features include:

1. Multiple instance types for varying compute, memory and storage needs

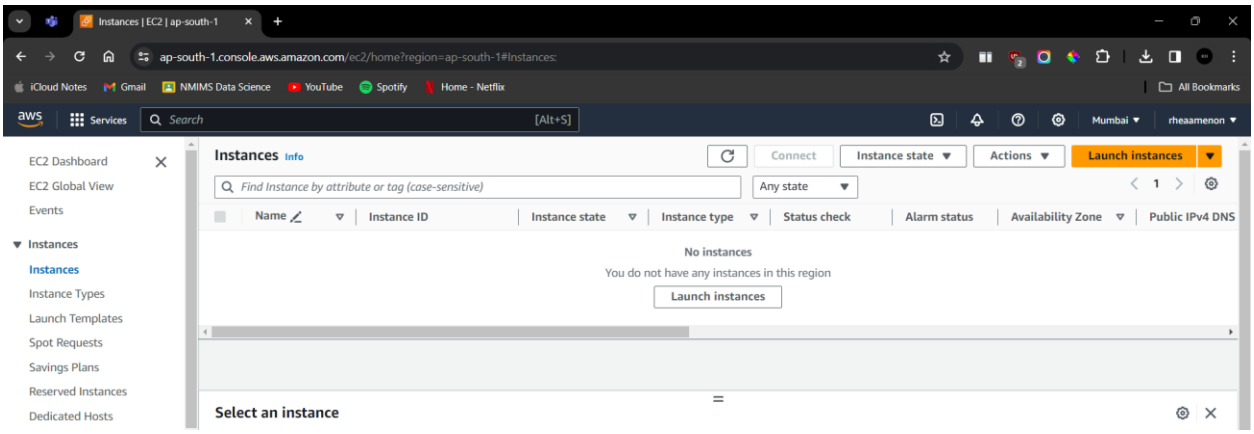
2. Auto scaling and load balancing
3. High availability within and across data centers
4. Secure network connectivity options and access controls
5. Integrated with other AWS services
6. Pay as you go pricing based on instance hours used

Implementing the windows machine using AWS EC2

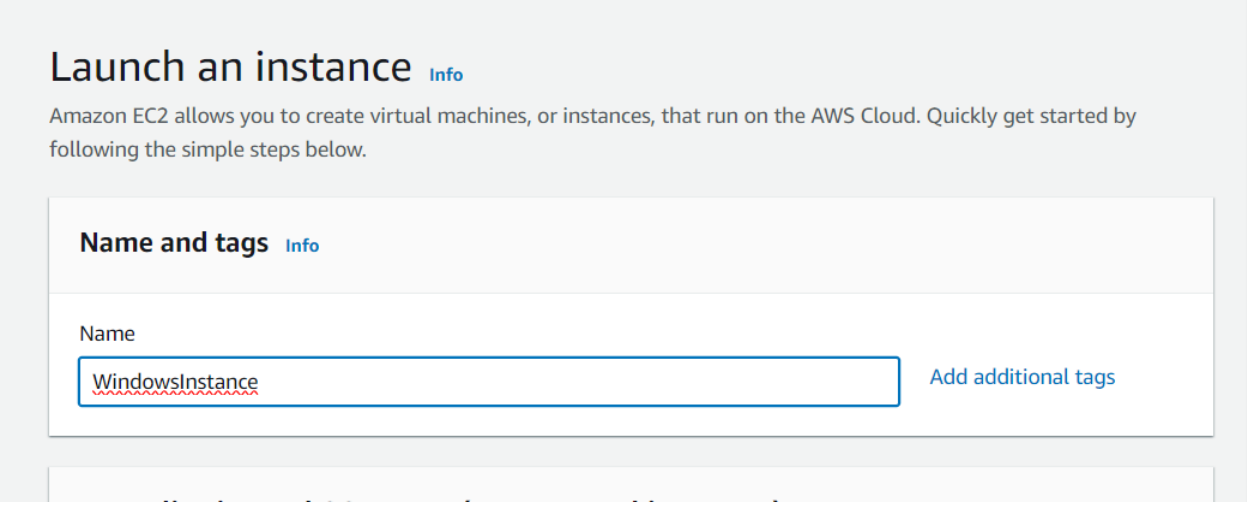
Go to the AWS console home, select All Services and select EC2.

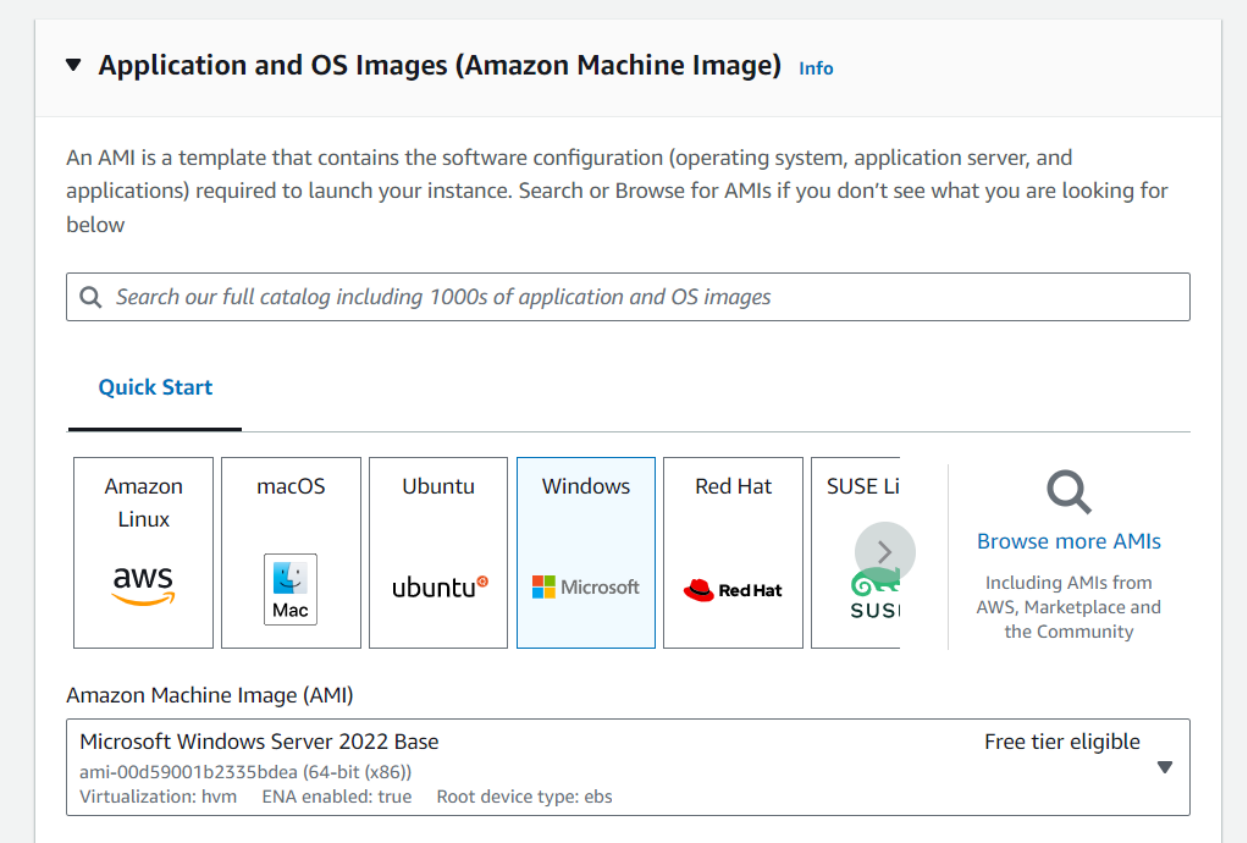


Go to Instances and launch an instance.

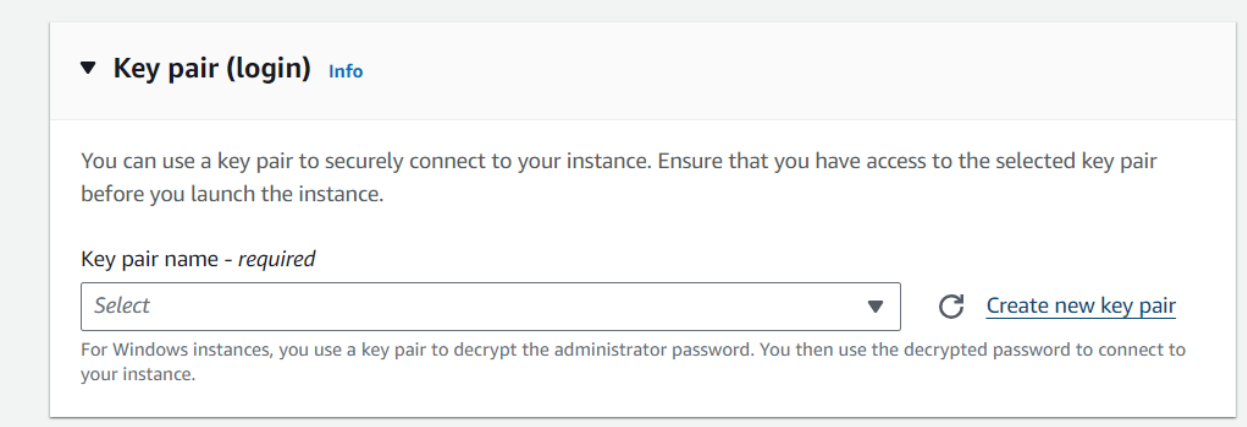


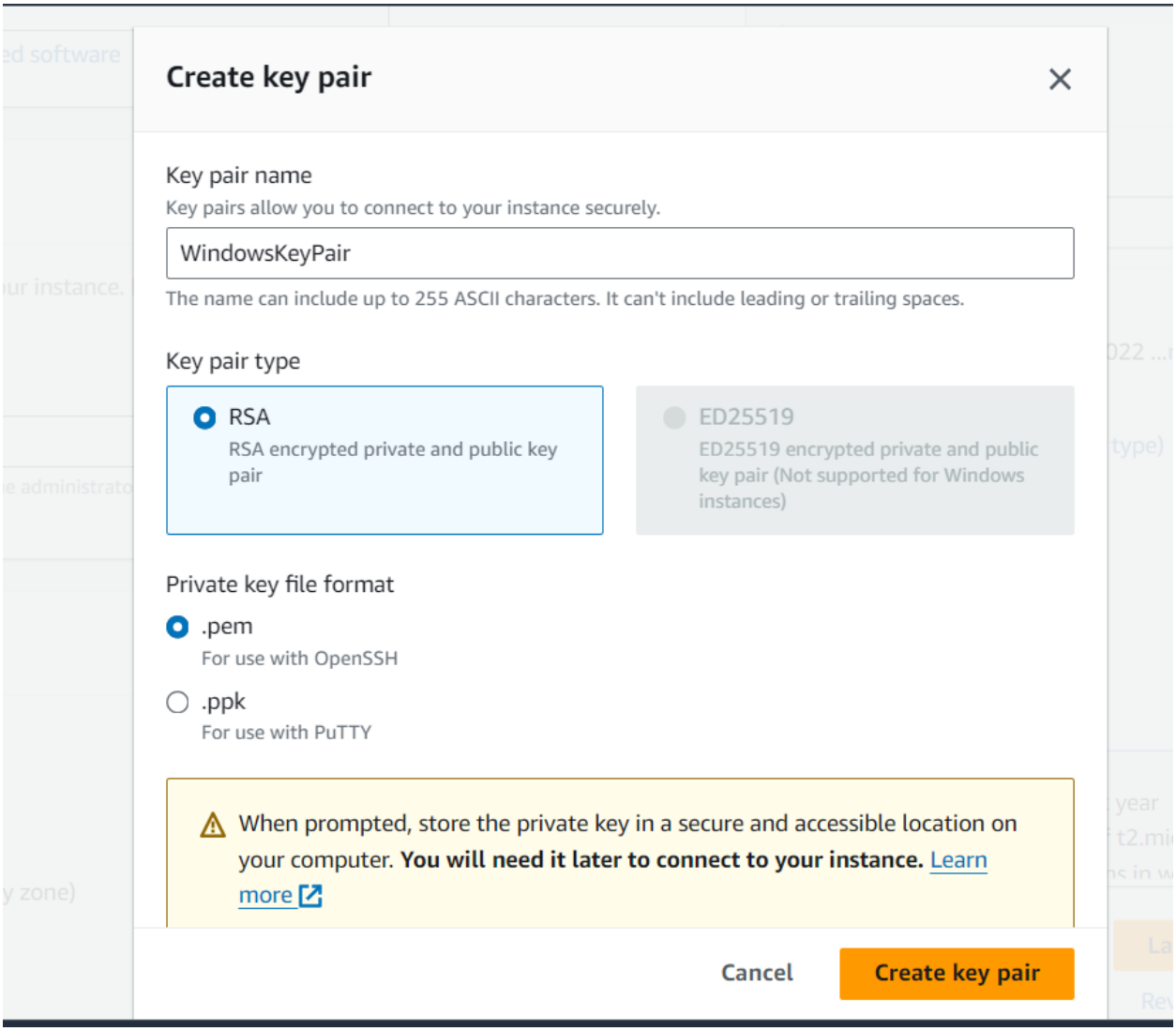
Give a name to the instance and choose Windows as the OS Image.



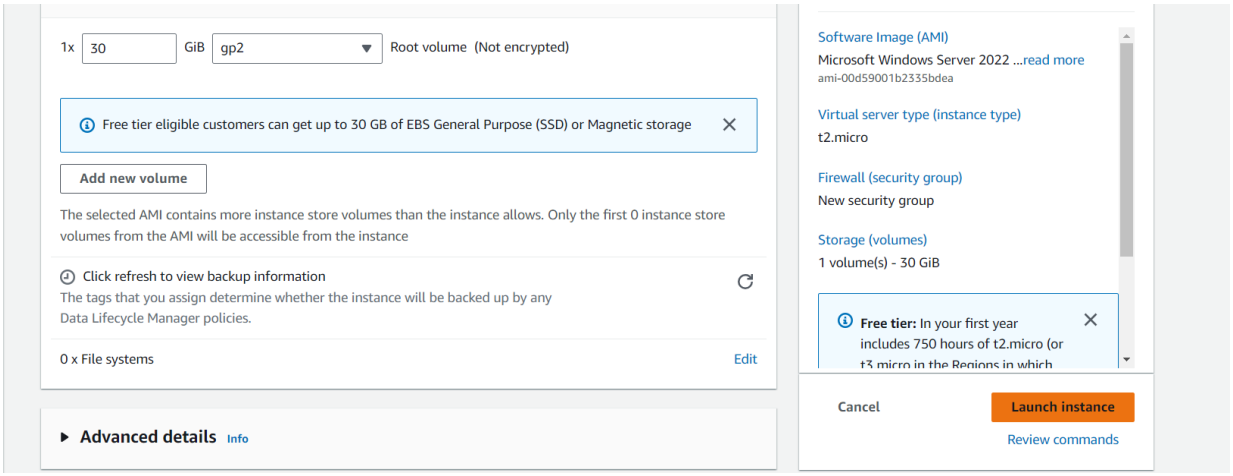


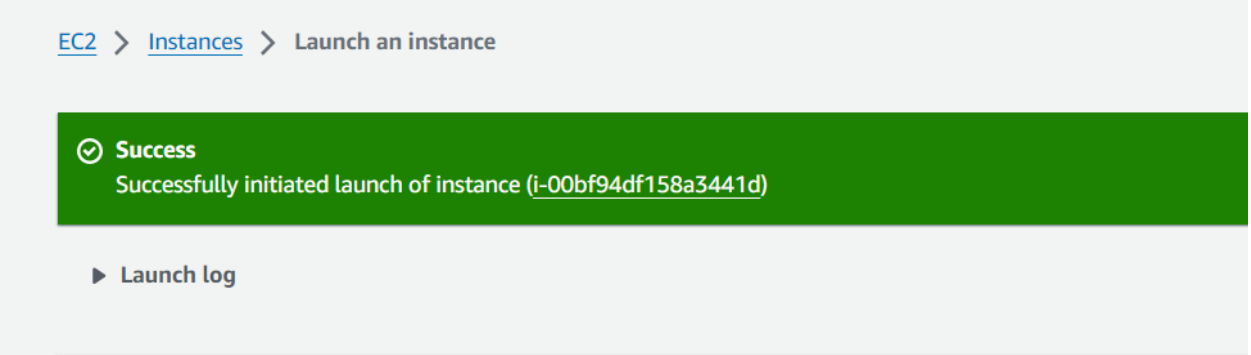
Create a new key pair and save the .pem file in your local device.



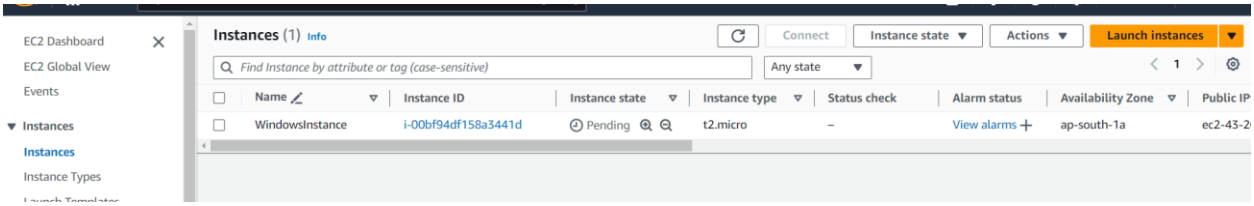


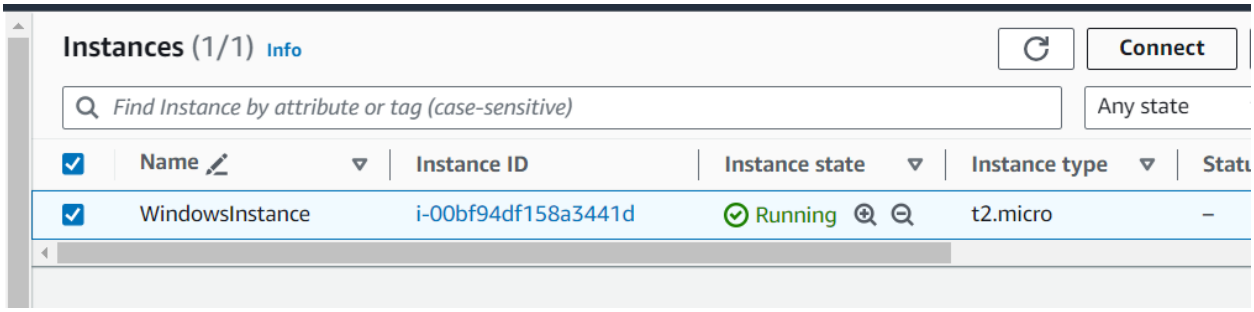
Leave all other default settings and launch the instance.



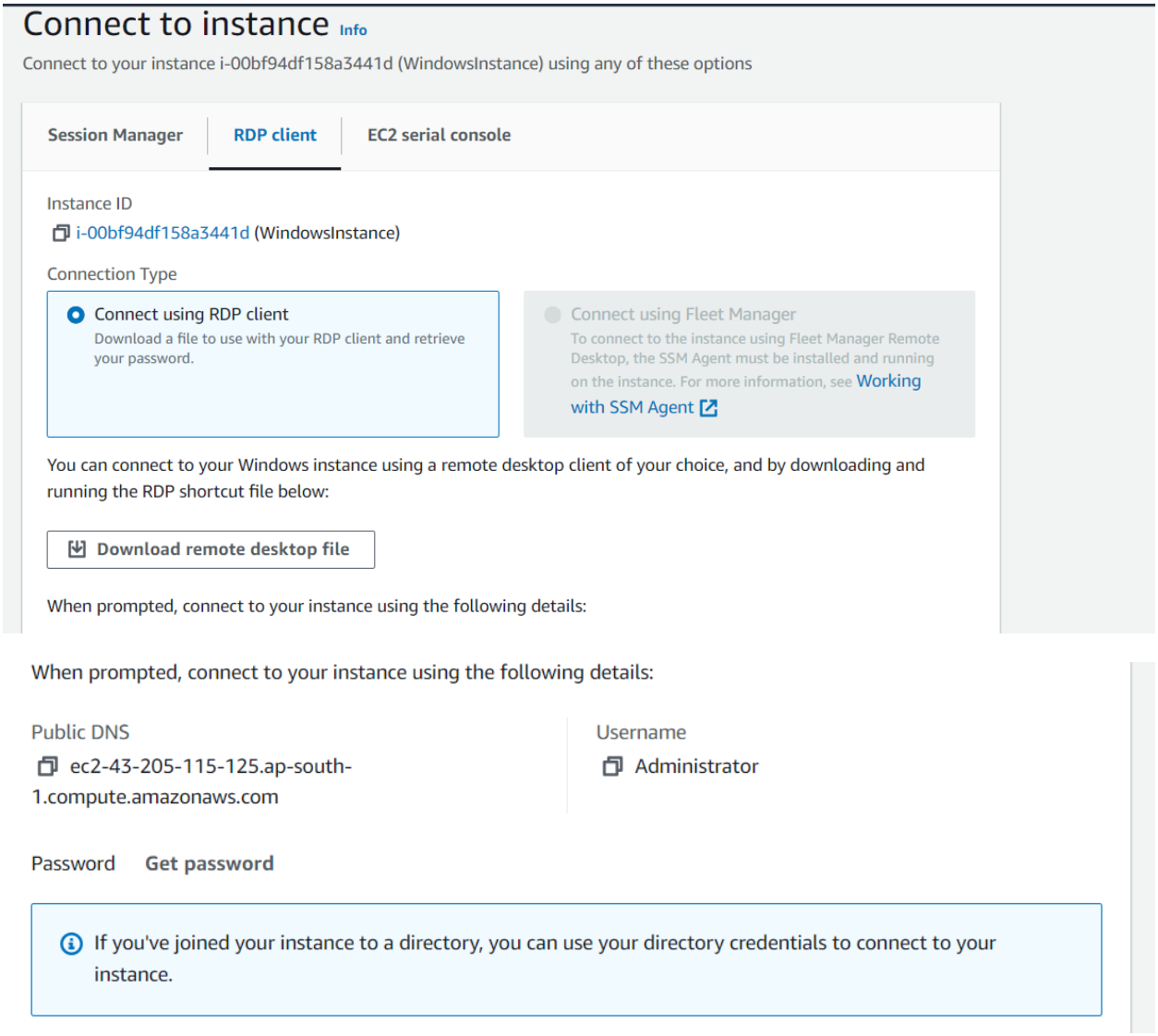


Go to the Instances menu, tick the created instance and click the Connect button.

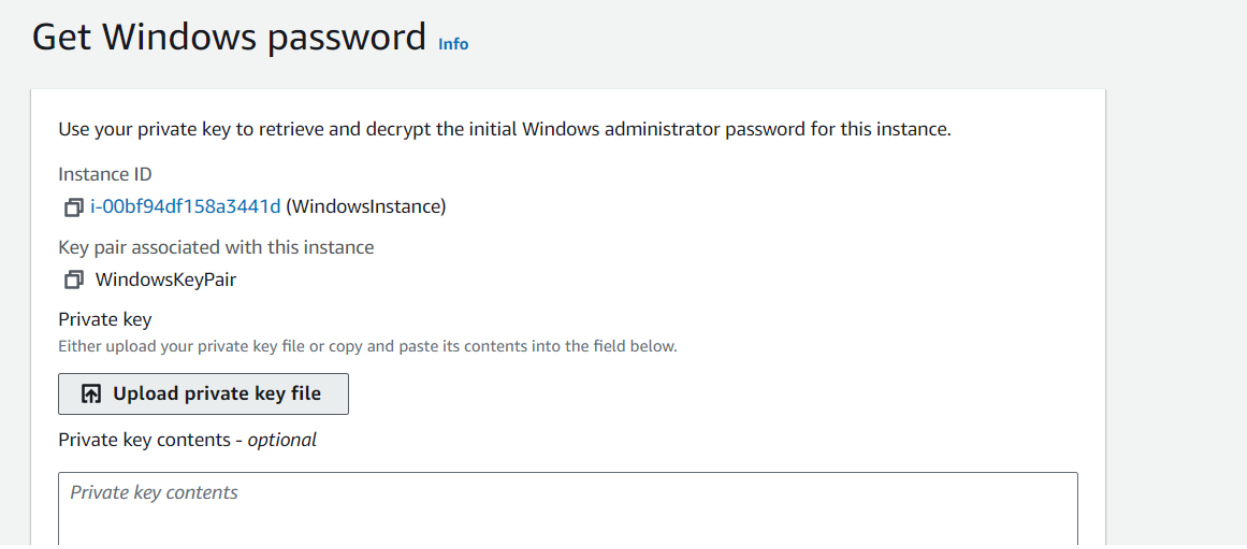


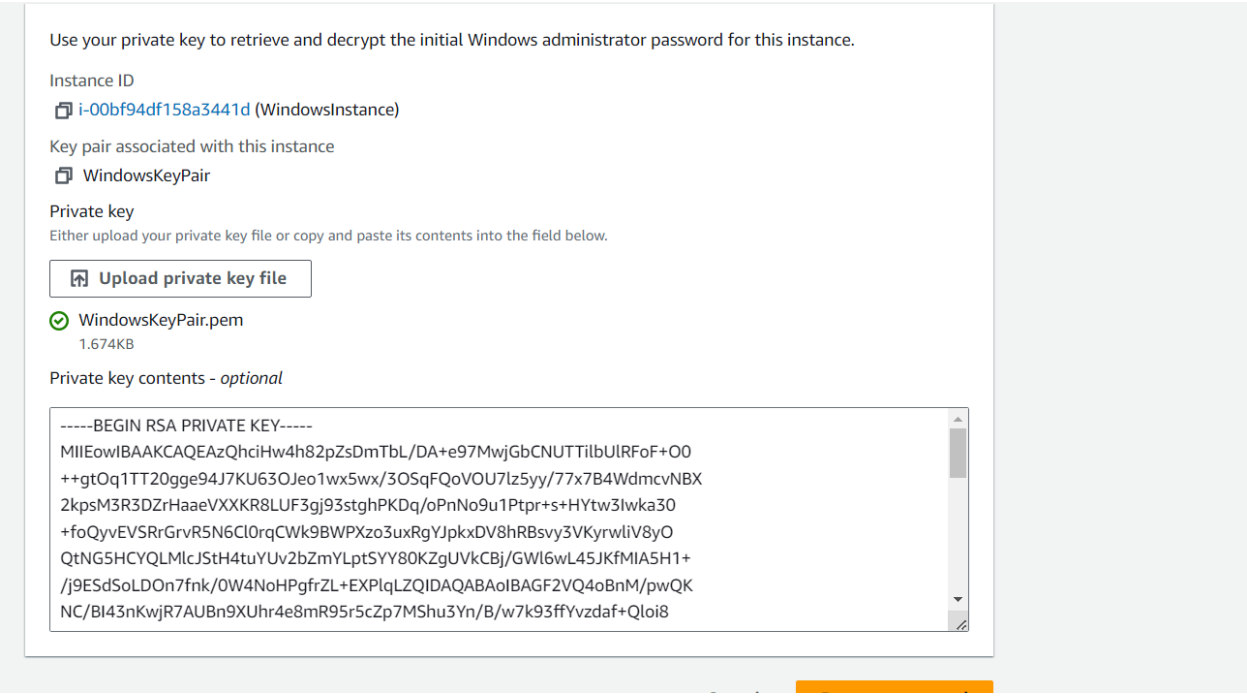


Click on the RDP Client tab and select ‘get password’.

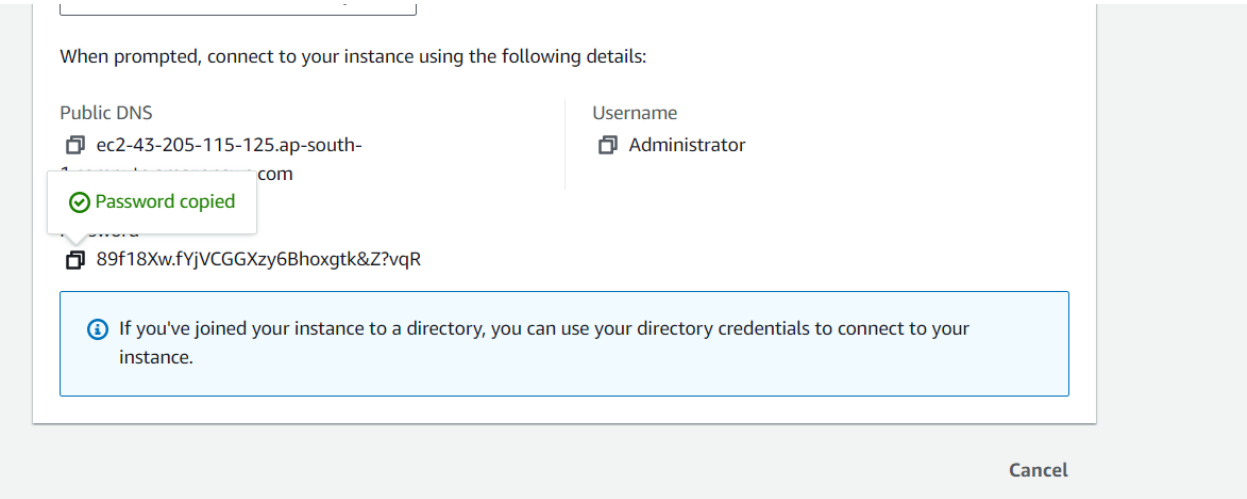


Upload the previously saved .pem file and decrypt the password.



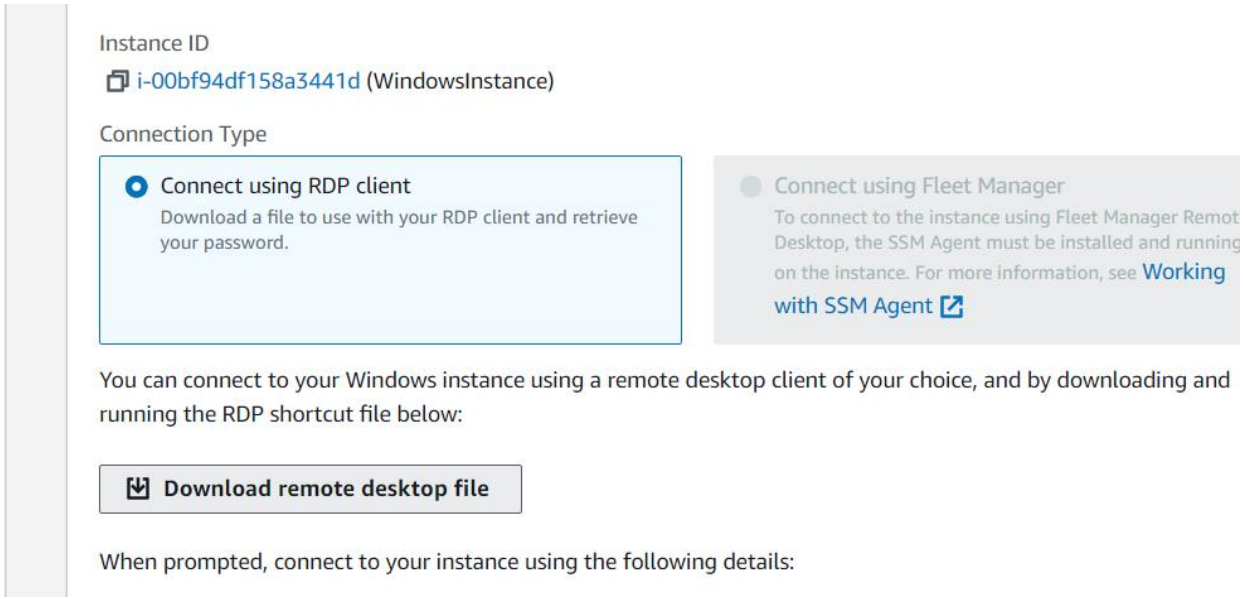


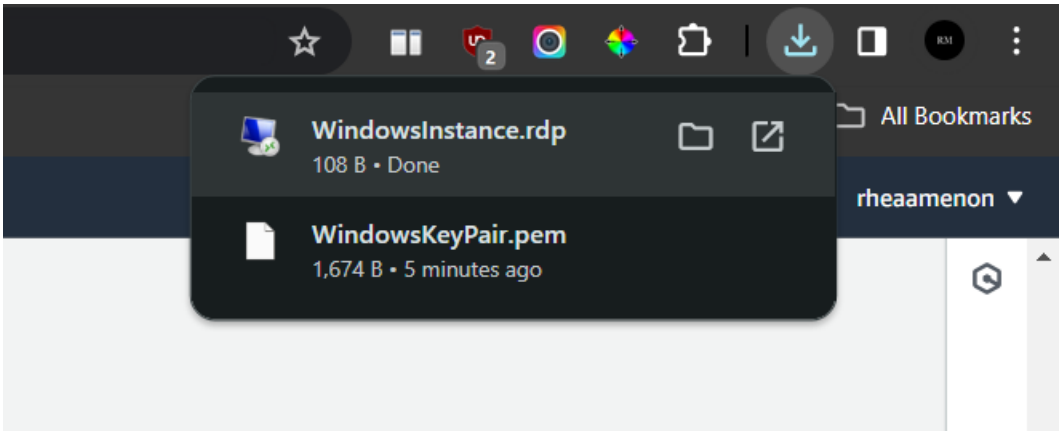
Copy the decrypted password.

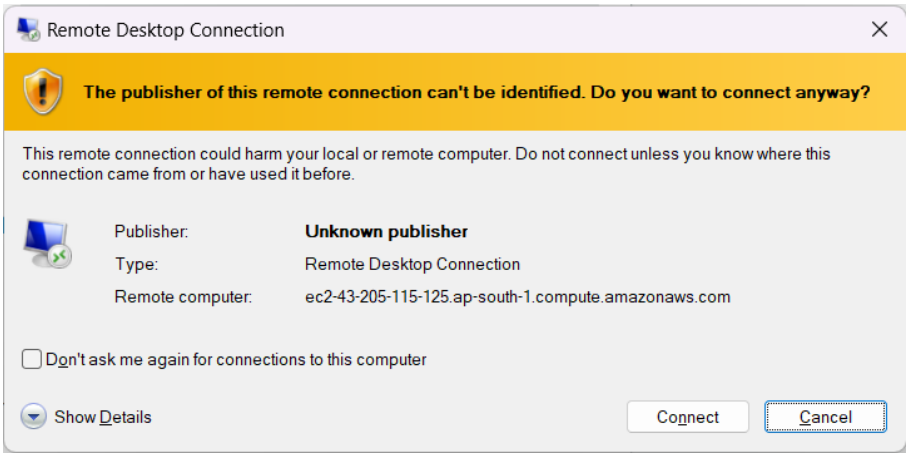


Password: 89f18Xw.fYjVCGGXzy6Bhoxgtk&Z?vqR

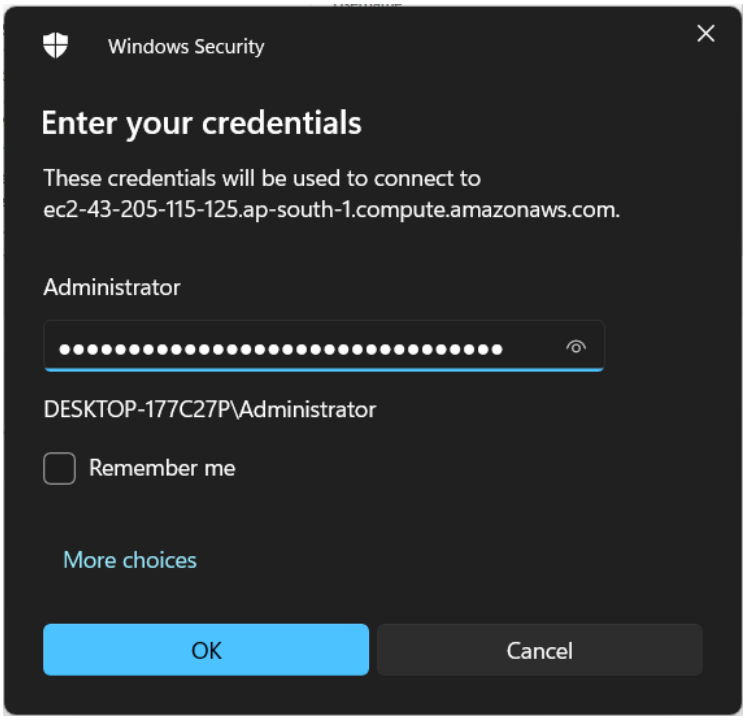
Now download the RDP File and open it.



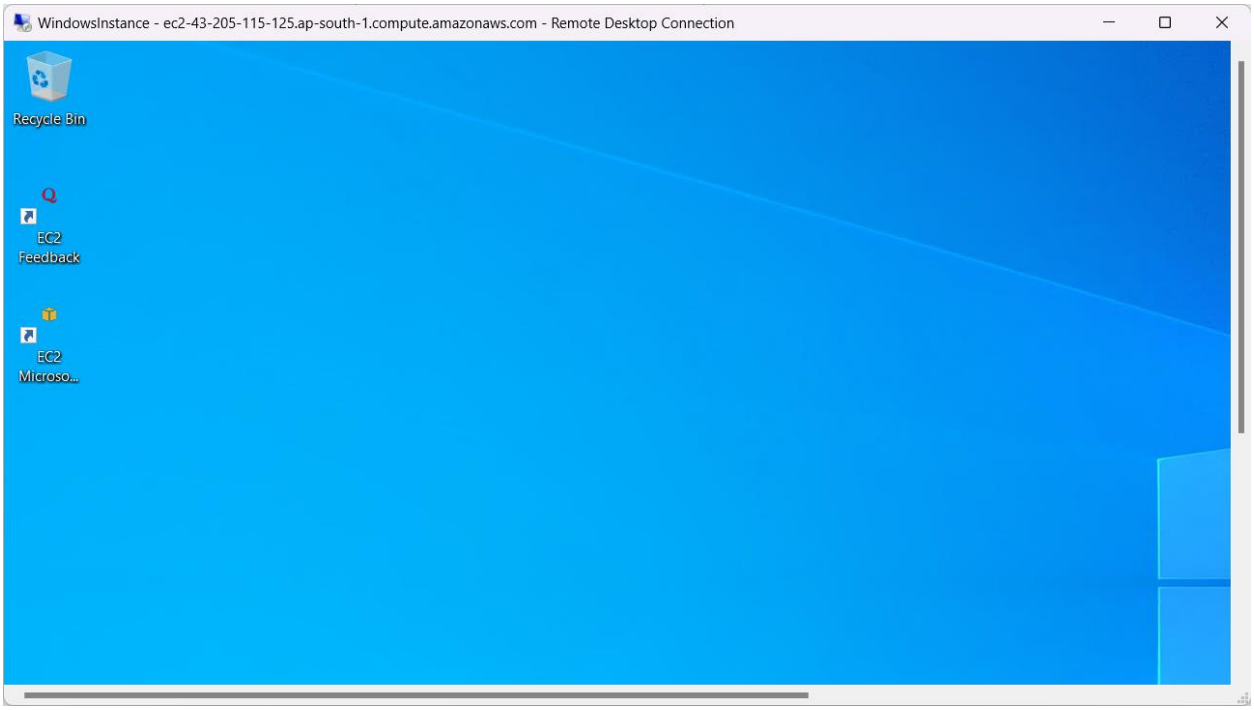




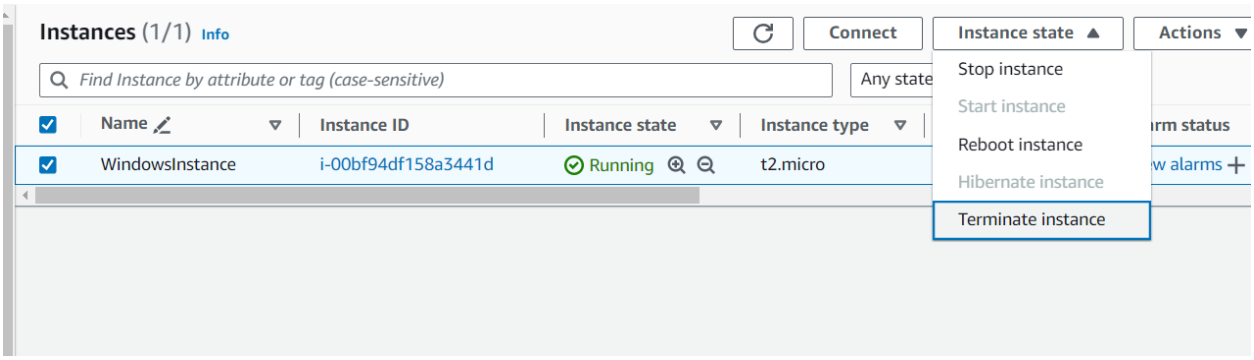
A dialog box will appear to add the password, paste the previously copied password.



The RDP Connection tab will now open with a virtual instance of the Windows OS.

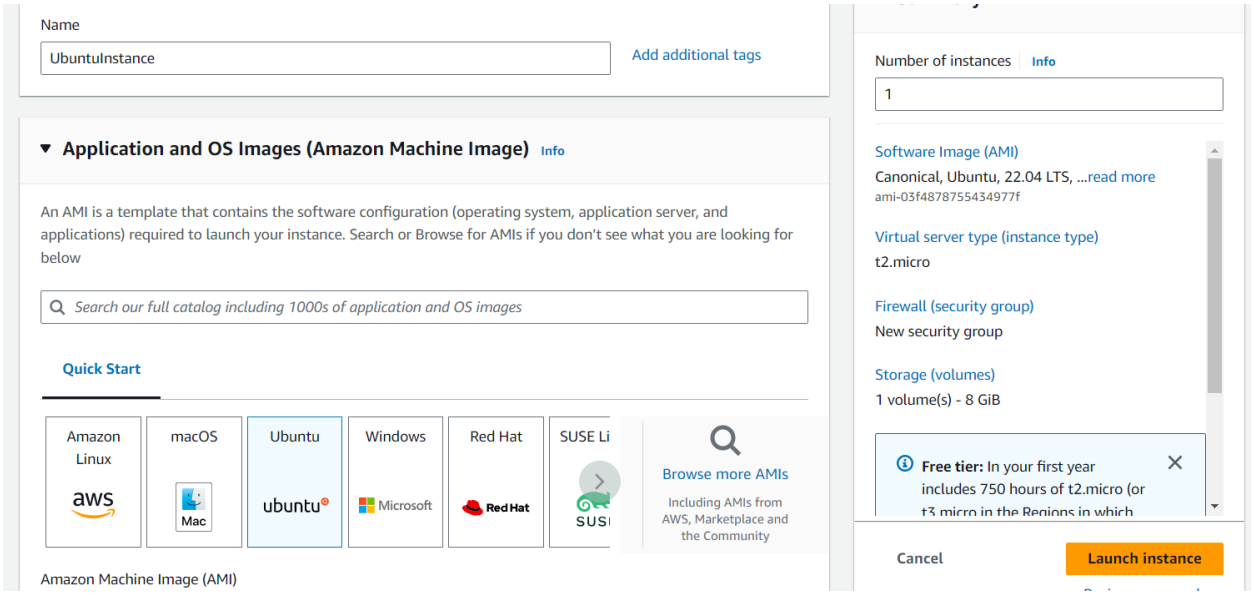


Now terminate the instance by clicking on it > Instance State > Terminate Instance.

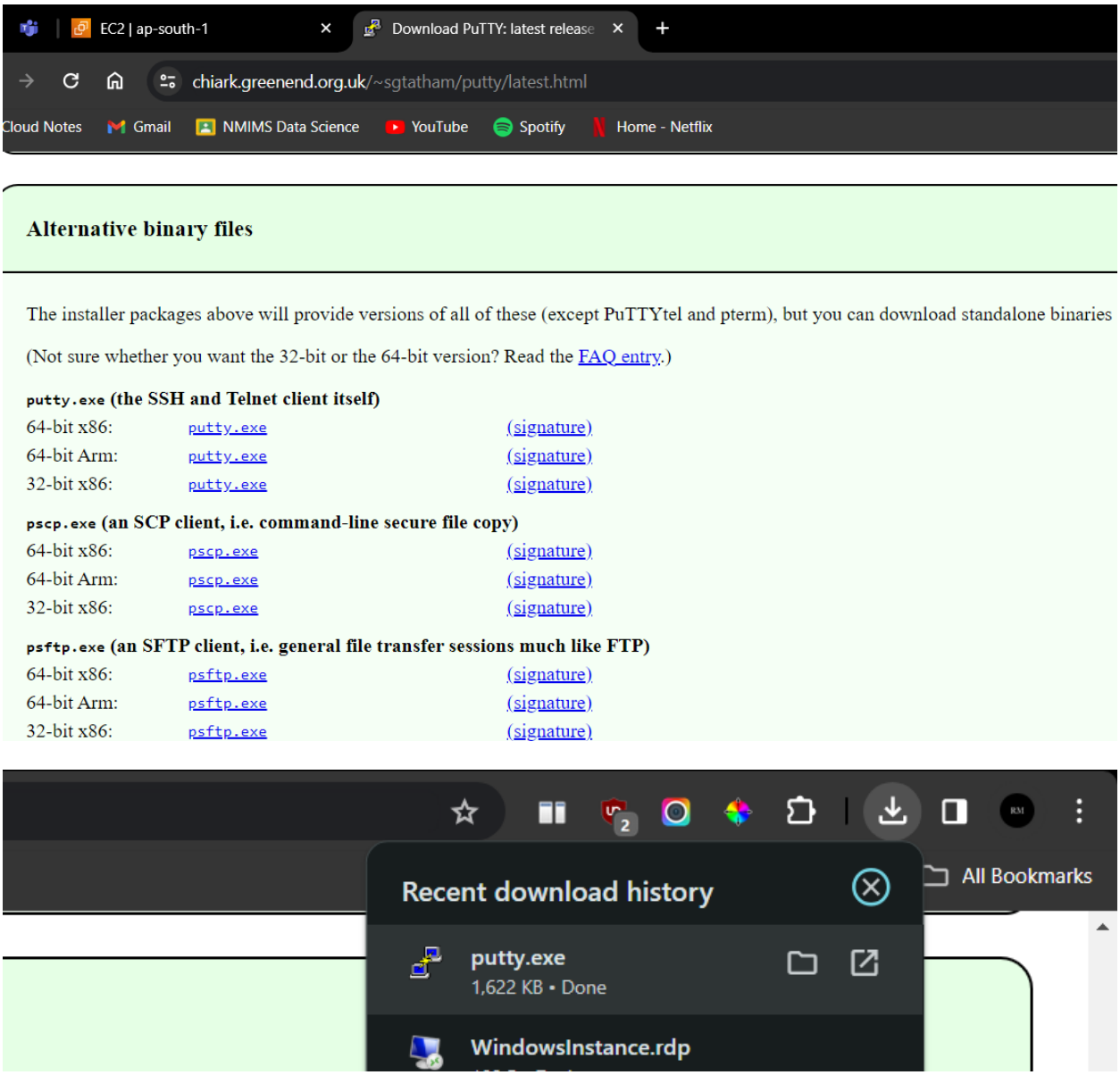


Linux OS

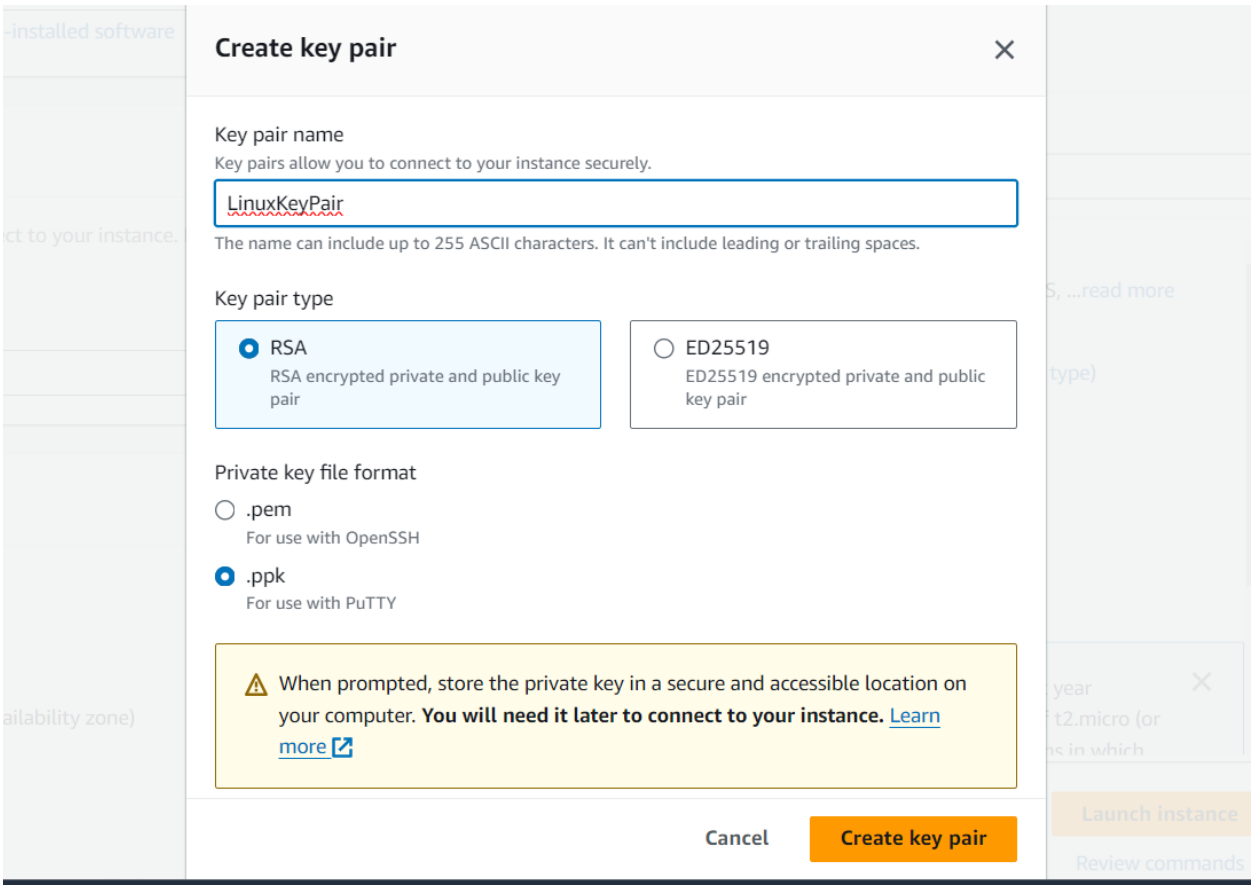
Repeat the same steps as for Windows but this time, choose Ubuntu as the OS Image.



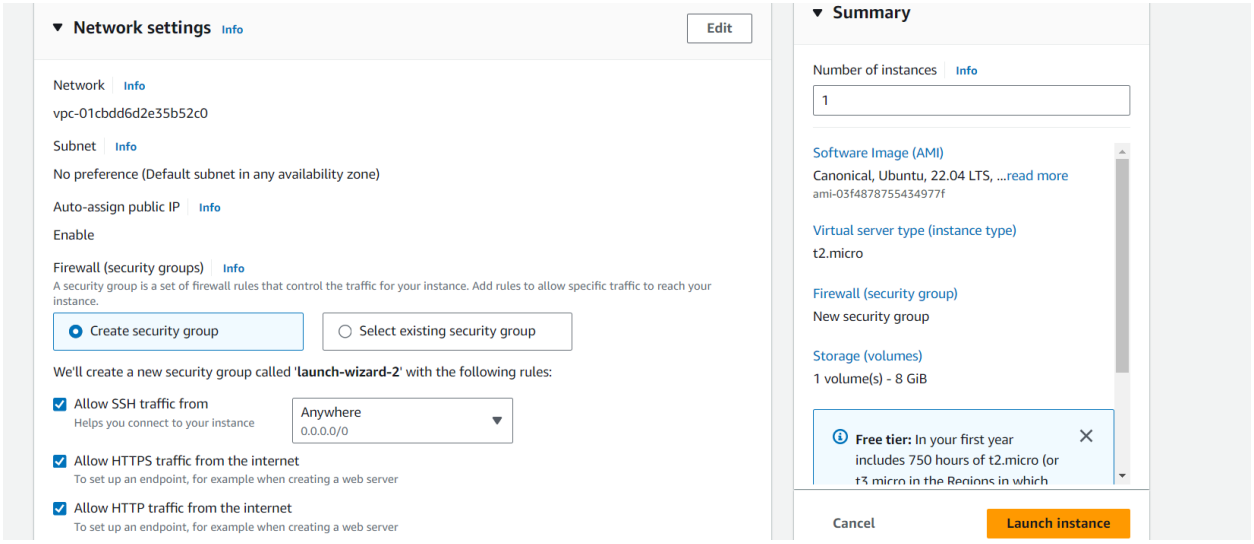
Download putty.exe from the link below.



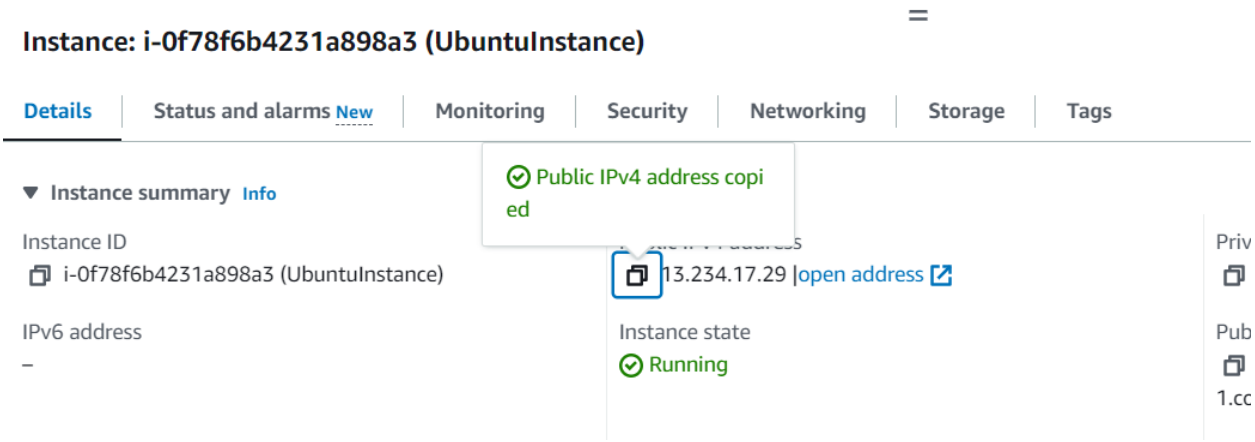
Create a new key pair for this instance and download the .ppk file.



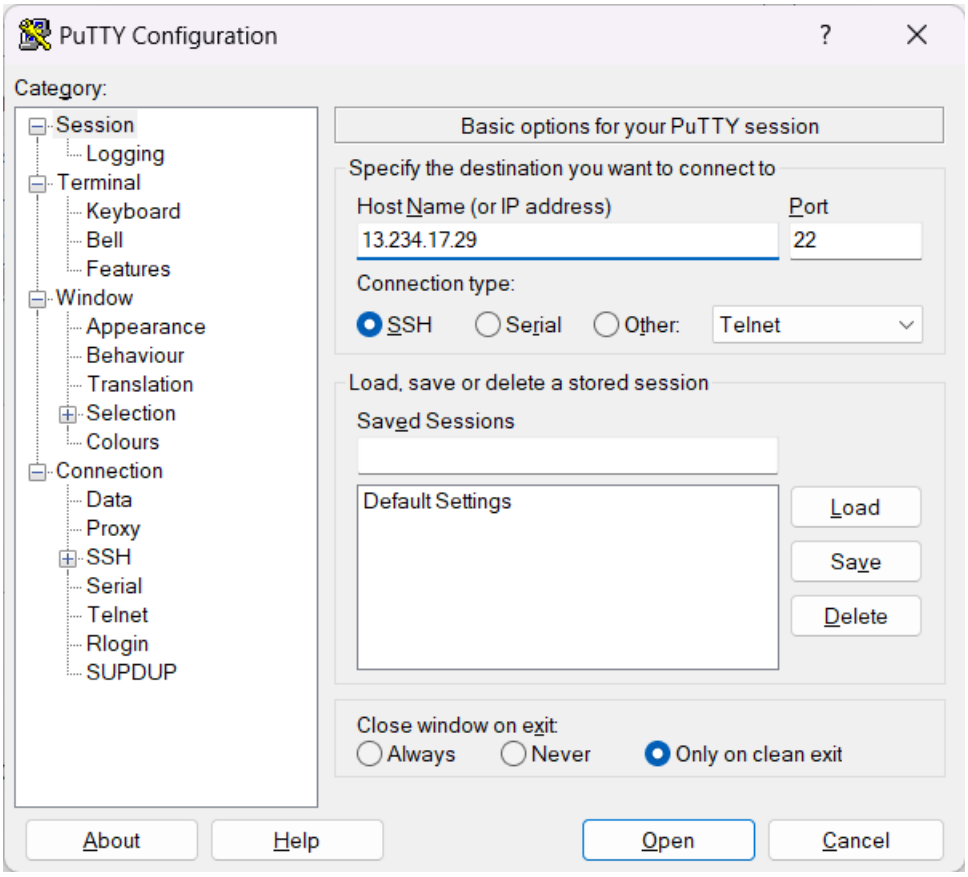
Tick all the boxes for the traffic permissions in network settings.



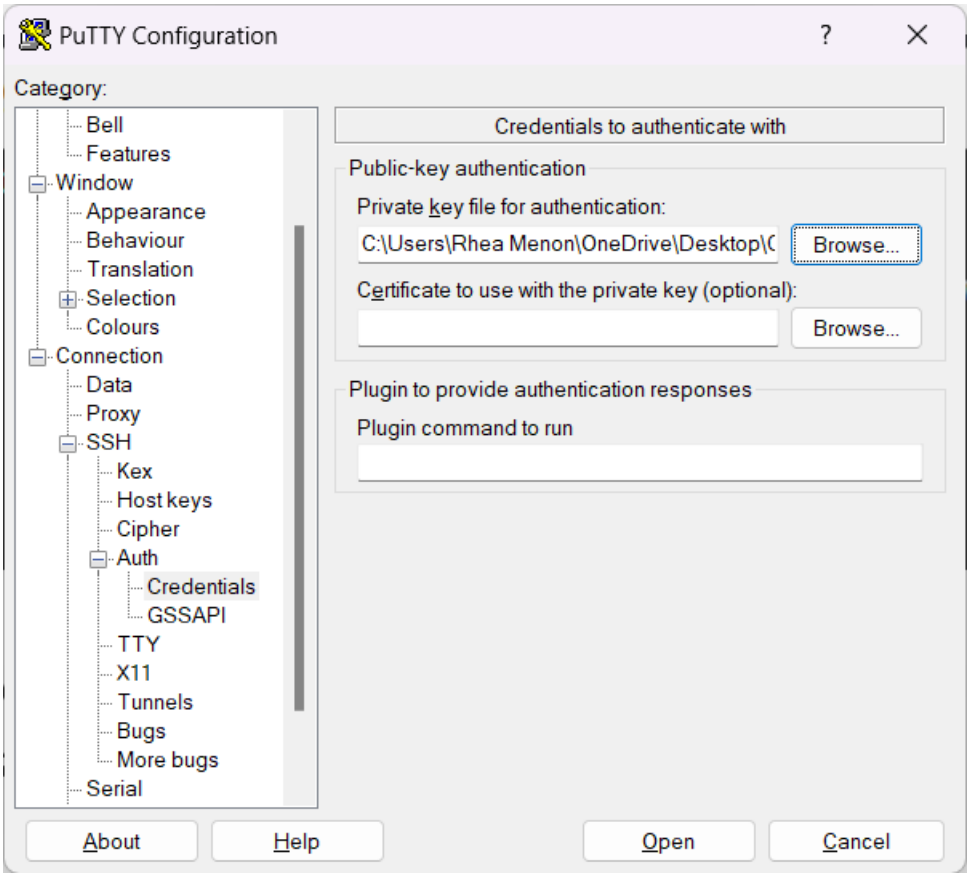
Launch the instance and in the instance summary, copy the public IPv4 address.



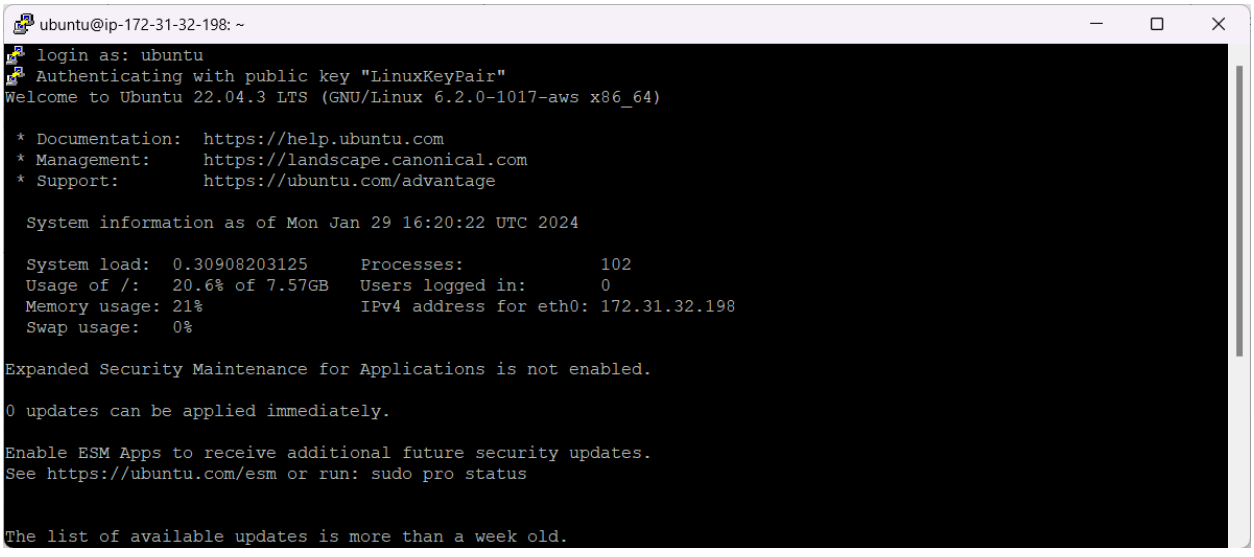
Open PuTTY and add the copied IP address to the Host IP address text box.



Go to Connection > SSH > Auth > Credentials and upload the .ppk key pair file for the private key.



Click open and ubuntu console will be launched. Login as *ubuntu*.

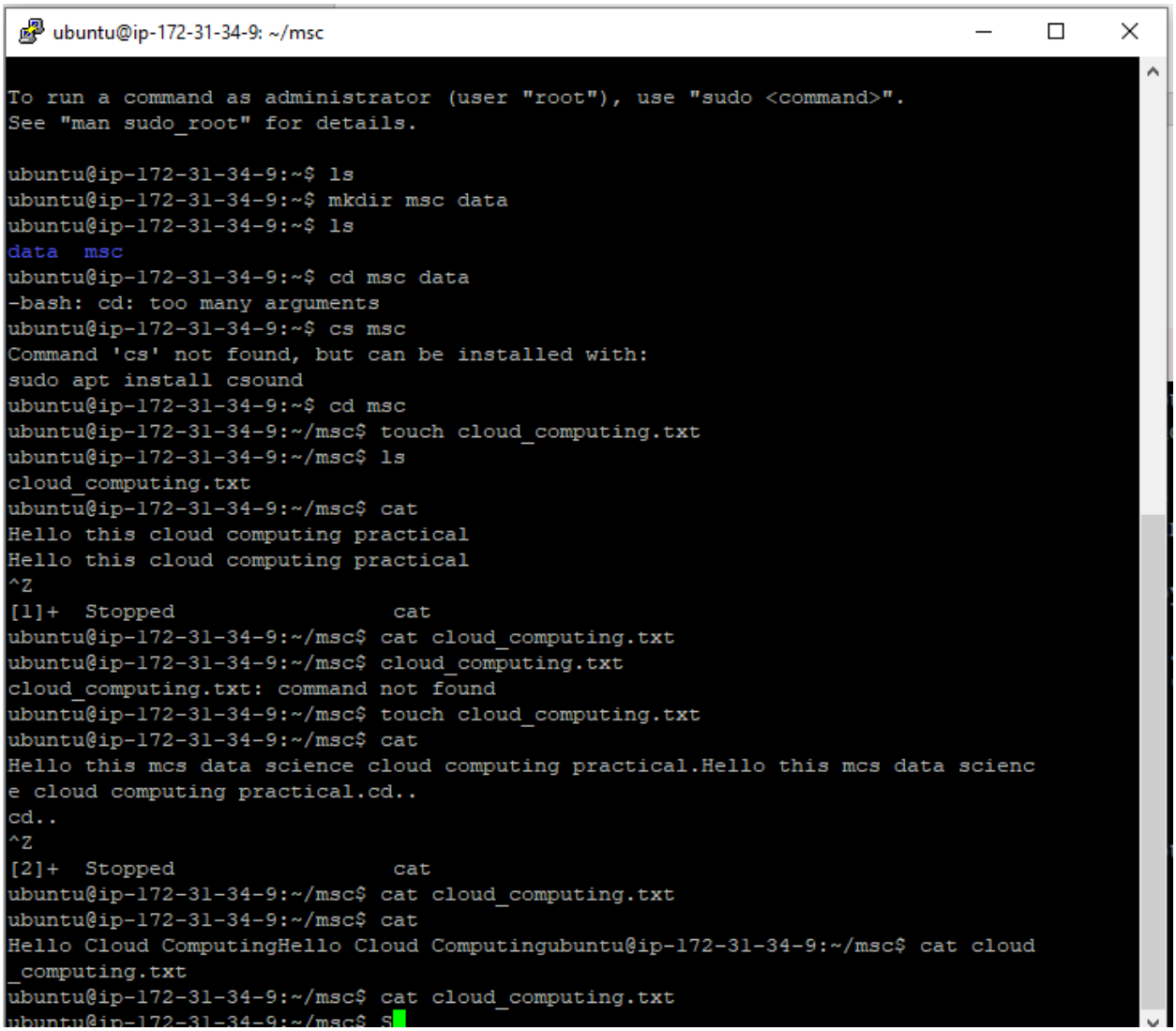


Commands:

mkdir [directory name]: makes a directory of the same name.

ls: lists all the files and directories in the current working directory.

```
ubuntu@ip-172-31-34-9:~$ ls
ubuntu@ip-172-31-34-9:~$ mkdir msc data
ubuntu@ip-172-31-34-9:~$ ls
data  msc
ubuntu@ip-172-31-34-9:~$ cd msc data
```



cat > [file name]: append text to a file.

cat [file name]: view the contents of a text file.

sudo apt install [program name]: installs the named program.

```
ubuntu@ip-172-31-32-198:~$ sudo apt install python3
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
python3 is already the newest version (3.10.6-1~22.04).
python3 set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
ubuntu@ip-172-31-32-198:~$
```

```
ubuntu@ip-172-31-32-198:~$ sudo apt install firefox
Waiting for cache lock: Could not get lock /var/lib/dpkg/lock-frontend. It is held by process 1315 (apt)
Waiting for cache lock: Could not get lock /var/lib/dpkg/lock-frontend. It is held by process 1315 (apt)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
firefox is already the newest version (1:1snap1-0ubuntu2).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
ubuntu@ip-172-31-32-198:~$
```

exit: closes the console.

Terminate this instance as well using the same steps as before.