NAME: MULLA AFRAH AKKAS ALI

ROLL NO.: 612038

BRANCH: T.E. – I.T.

SEMESTER: ODD SEMESTER 5

COURSE: Advance DevOPs (ITL504)

DATE: 24-08-2022

EXPERIMENT 7

1. What is Containerization / Docker? Explain Docker Architecture with the help of diagram.

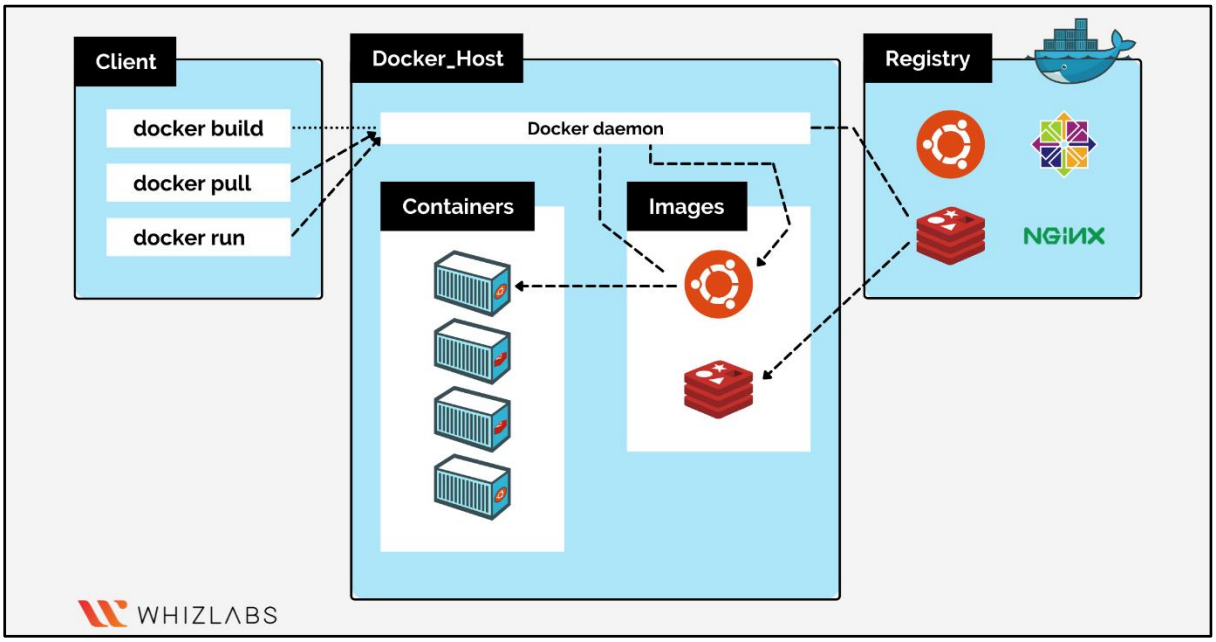
Containerization is the packaging together of software code with all it's necessary components like libraries, frameworks, and other dependencies so that they are isolated in their own "container." The container acts as a kind of bubble or a computing environment surrounding the application and keeping it independent of its surroundings.

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly.

Docker Architecture

To allow for an application to be self-contained the Docker approach moves up the abstraction of resources from the hardware level to the Operating System level.

To further understand Docker, let us look at its architecture. It uses a client-server model and comprises of the following components:



- Docker daemon: The daemon is responsible for all container related actions and receives commands via the CLI or the REST API.

- Docker Client: A Docker client is how users interact with Docker. The Docker client can reside on the same host as the daemon or a remote host.

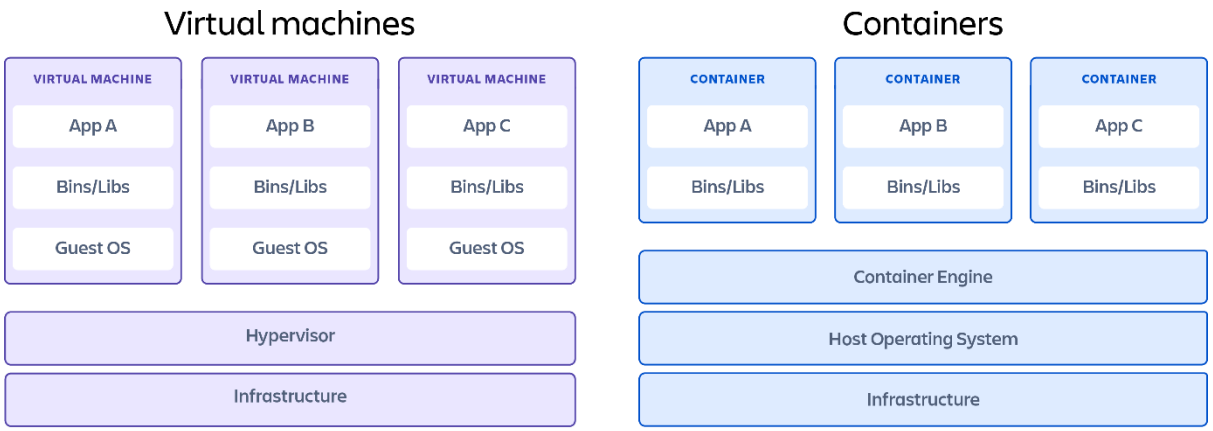
- Docker Objects: Objects are used to assemble an application. Apart from networks, volumes, services, and other objects the two main requisite objects are:

    - Images: The read-only template used to build containers. Images are used to store and ship applications.

    - Containers: Containers are encapsulated environments in which applications are run. A container is defined by the image and configuration options. At a lower level, you have containers, which is a core container runtime that initiates, and supervises container performance.

- Docker Registries: Registries are locations from where we store and download (or "pull") images.

2. Compare Containers vs VMs



- A virtual machine (VM) is a virtual environment that functions as a virtual computer system with its own CPU, memory, network interface, and storage, created on a physical hardware system (located off- or on-premises).

- Containerization and virtualization are similar in that they both allow for full isolation of applications so that they can be operational in multiple environments. Where the main differences lie are in size and portability.

- The key differentiator between containers and virtual machines is that virtual machines virtualize an entire machine down to the hardware layers and containers only virtualize software layers above the operating system level.

- VMs are the larger of the two, typically measured by the gigabyte and containing their own OS, which allows them to perform multiple resource-intensive functions at once. The increased resources available to VMs allows them to abstract, split, duplicate, and emulate entire servers, operating systems, desktops, databases, and networks.

- Containers are much smaller, typically measured by the megabyte and not packaging anything bigger than an app and its running environment.

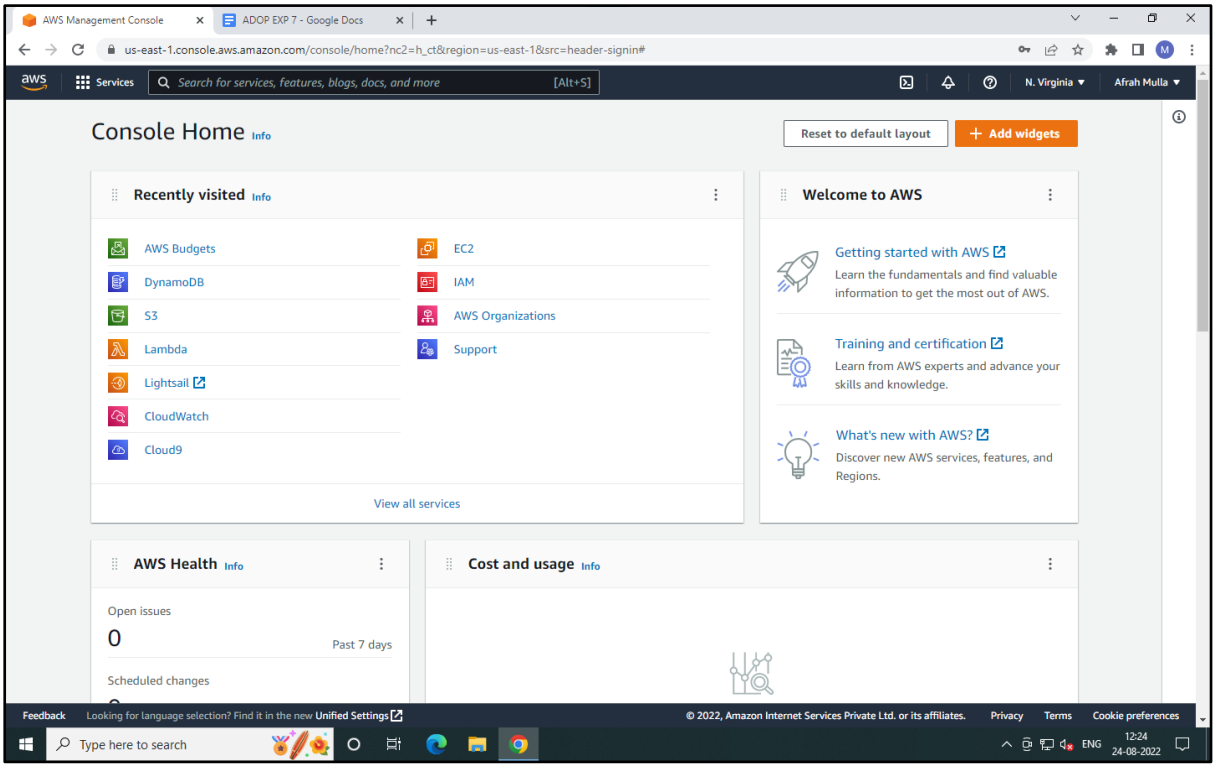
- Where VMs work well with traditional, monolithic IT architecture, containers were made to be compatible with newer and emerging technology like clouds, CI/CD, and DevOps.

3. Why are Containers lightweight?

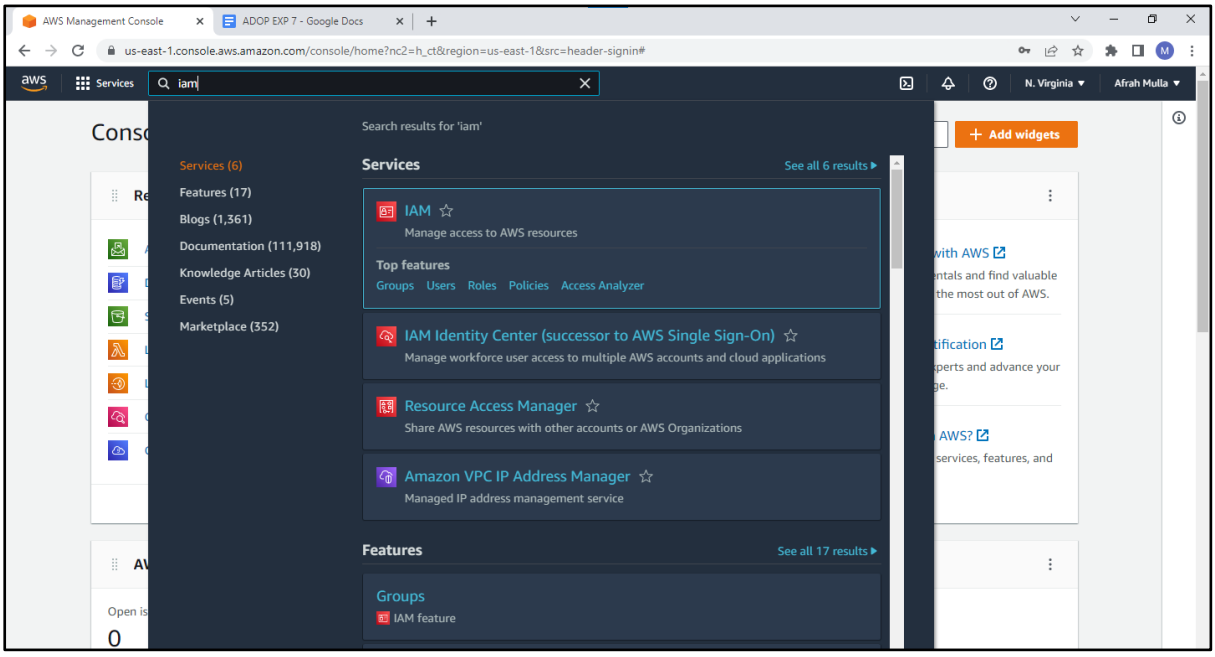
Inside a container are all the necessary executables, binary code, libraries, and configuration files. Compared to server or machine virtualization approaches, however, containers share the machine's OS system kernel and therefore do not require an OS per application, driving higher server efficiencies and reducing server and licensing costs. This makes them more lightweight and portable, with significantly less overhead.

4. Deploy a containerized web Application on AWS EC2 Linux. [Install Docker, pull nginx image and run it]. Pull python images and run the command to list all the locally stored docker images.

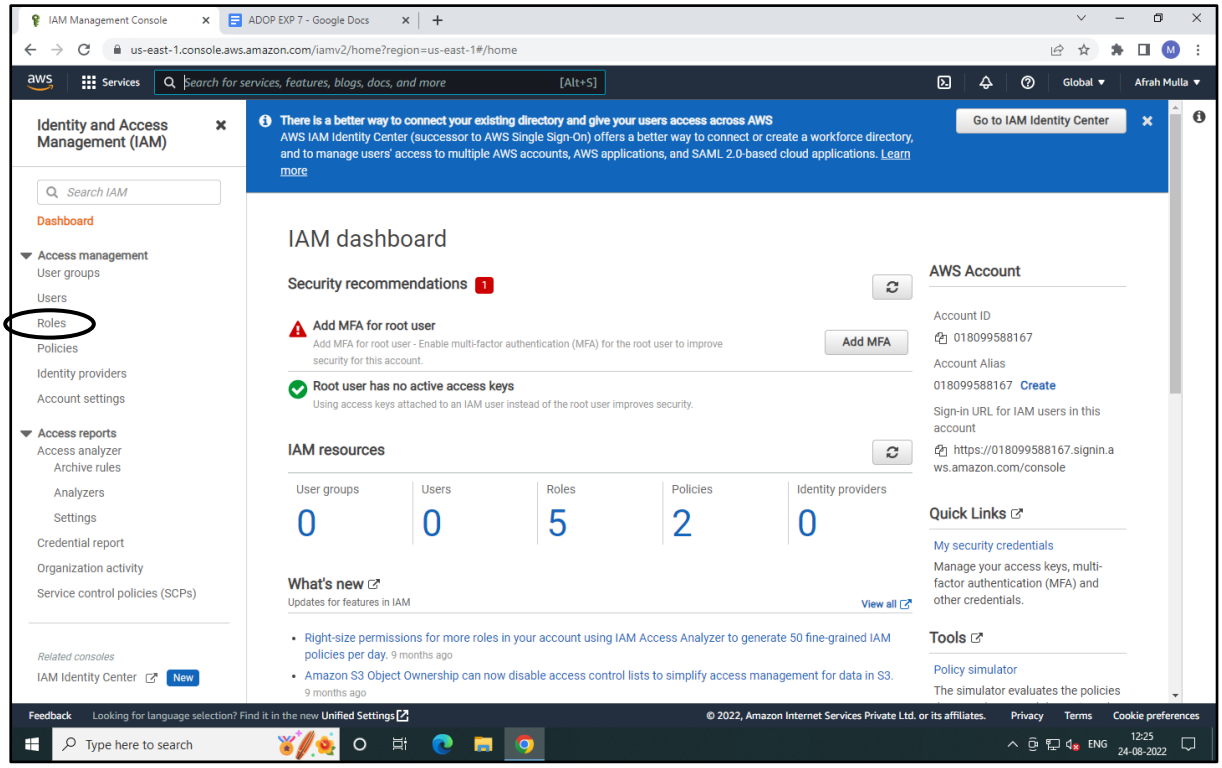
Step 1: AWS Management Console Dashboard



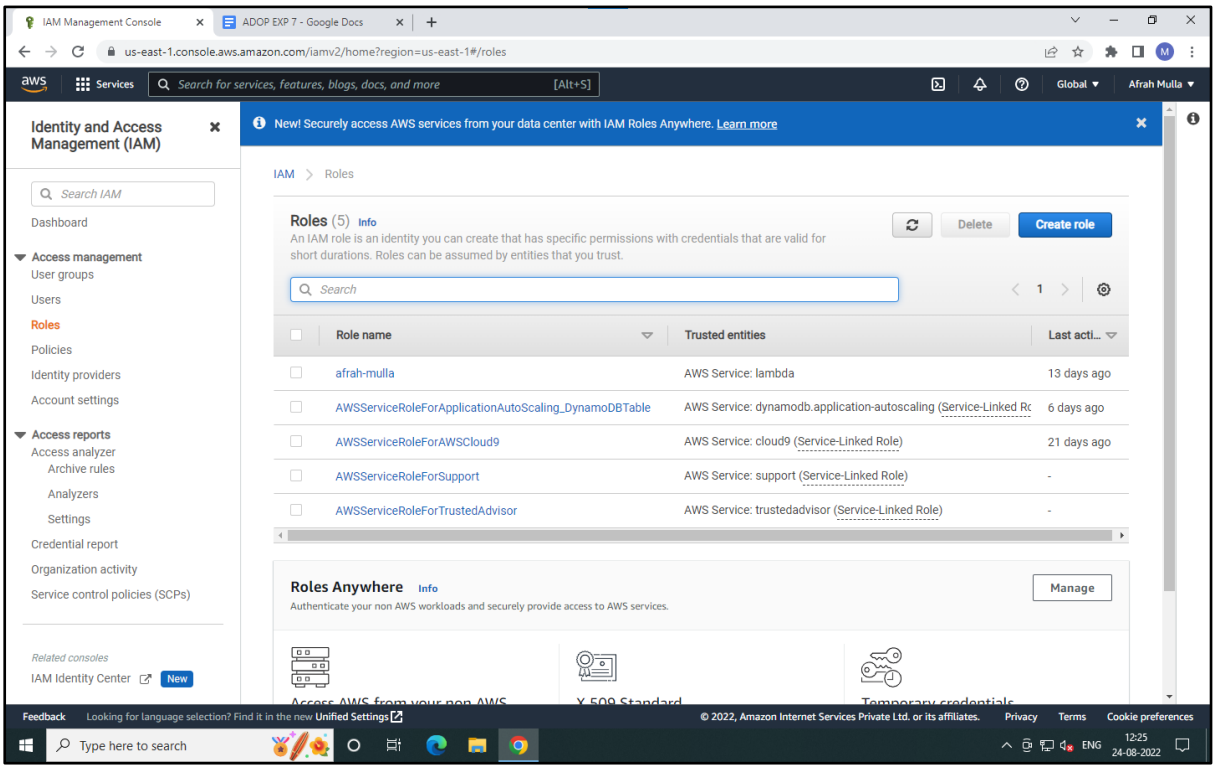
Step 2: Search for IAM and click on it



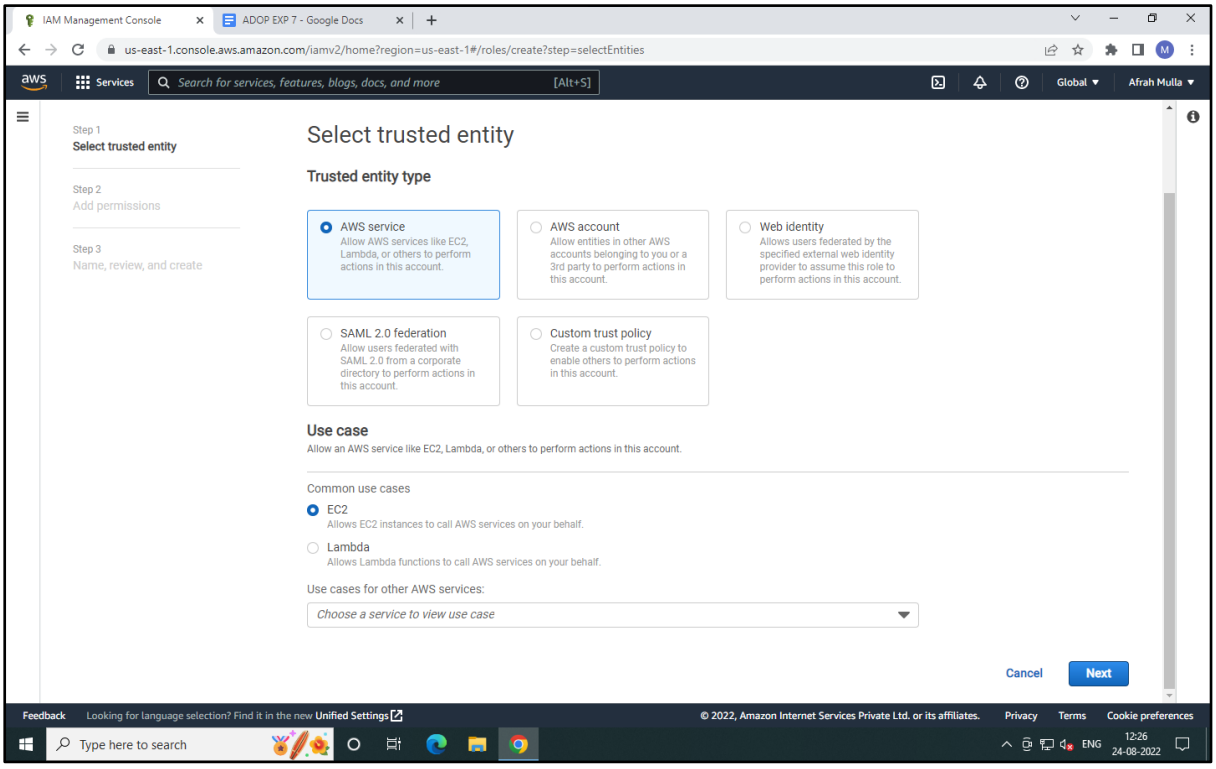
Step 3: Select ‘Roles’



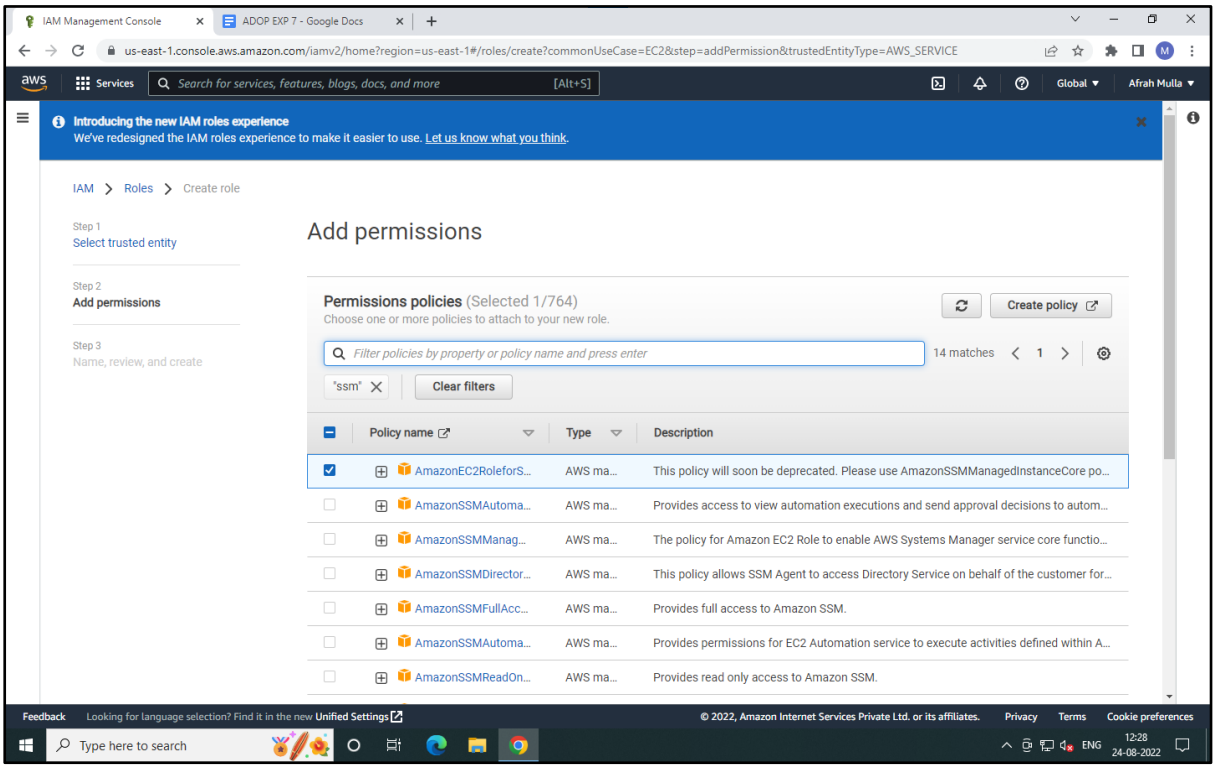
Step 4: Click on ‘Create role’



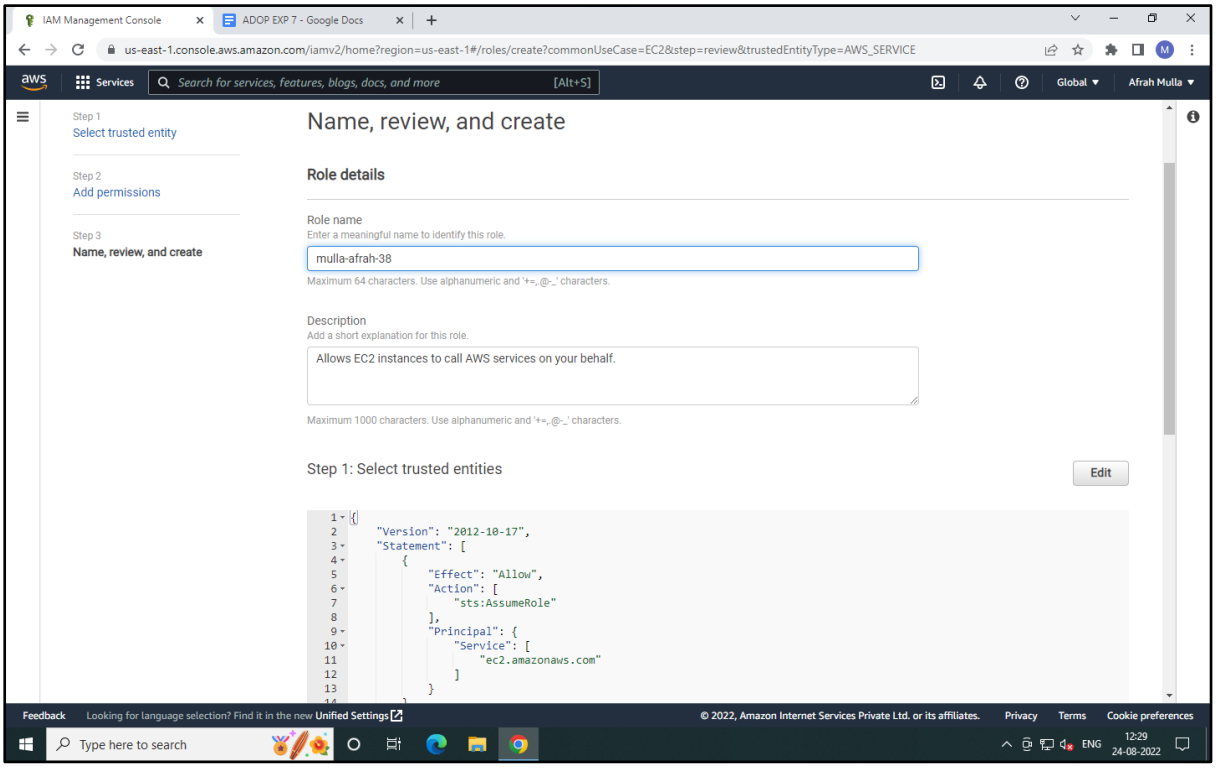
Step 5: Set Trusted entity type to AWS service, use case to EC2 -> Next

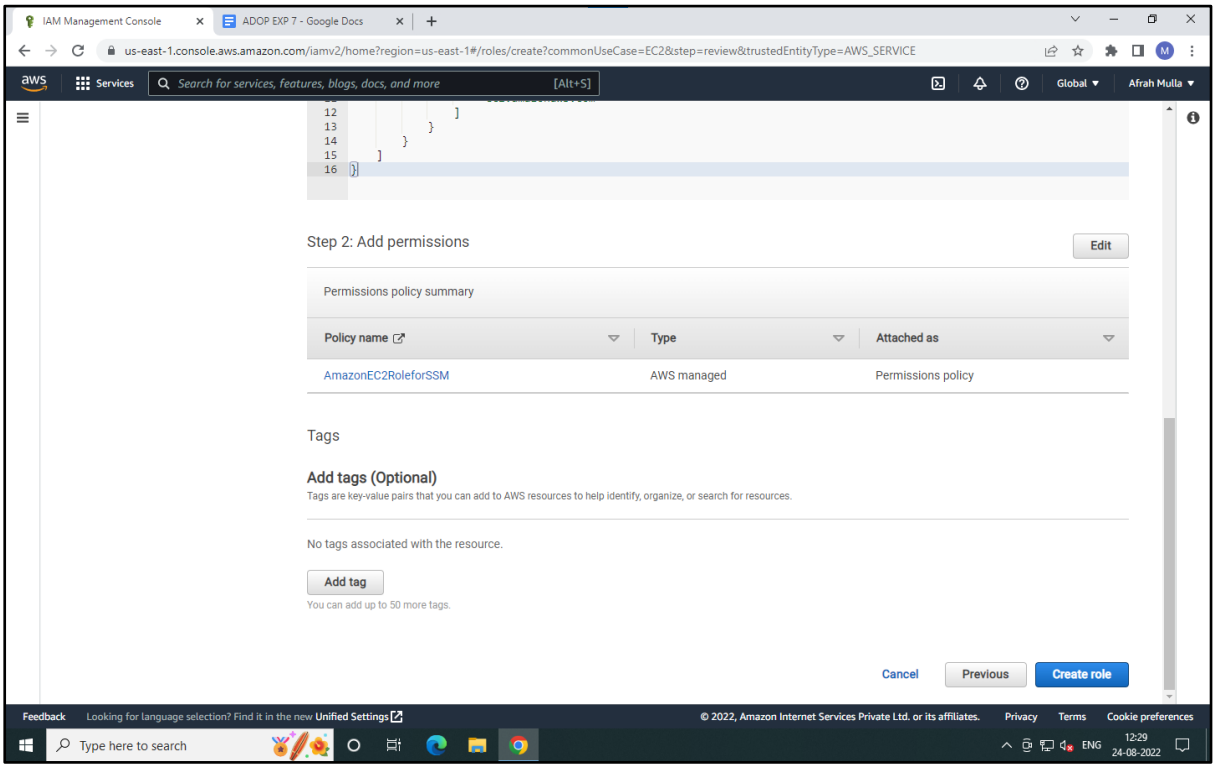


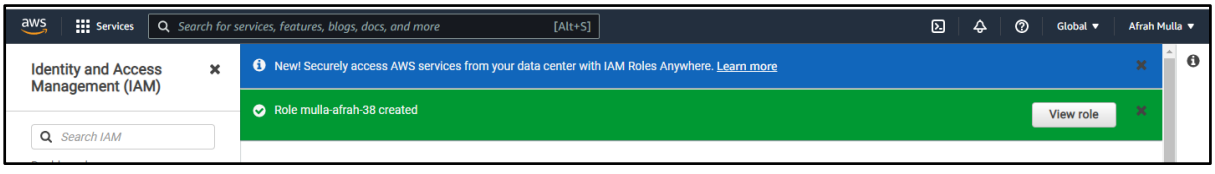
Step 6: Search for SSH in Permission policies -> Select AmazonEC2RoleforSSH -> Next



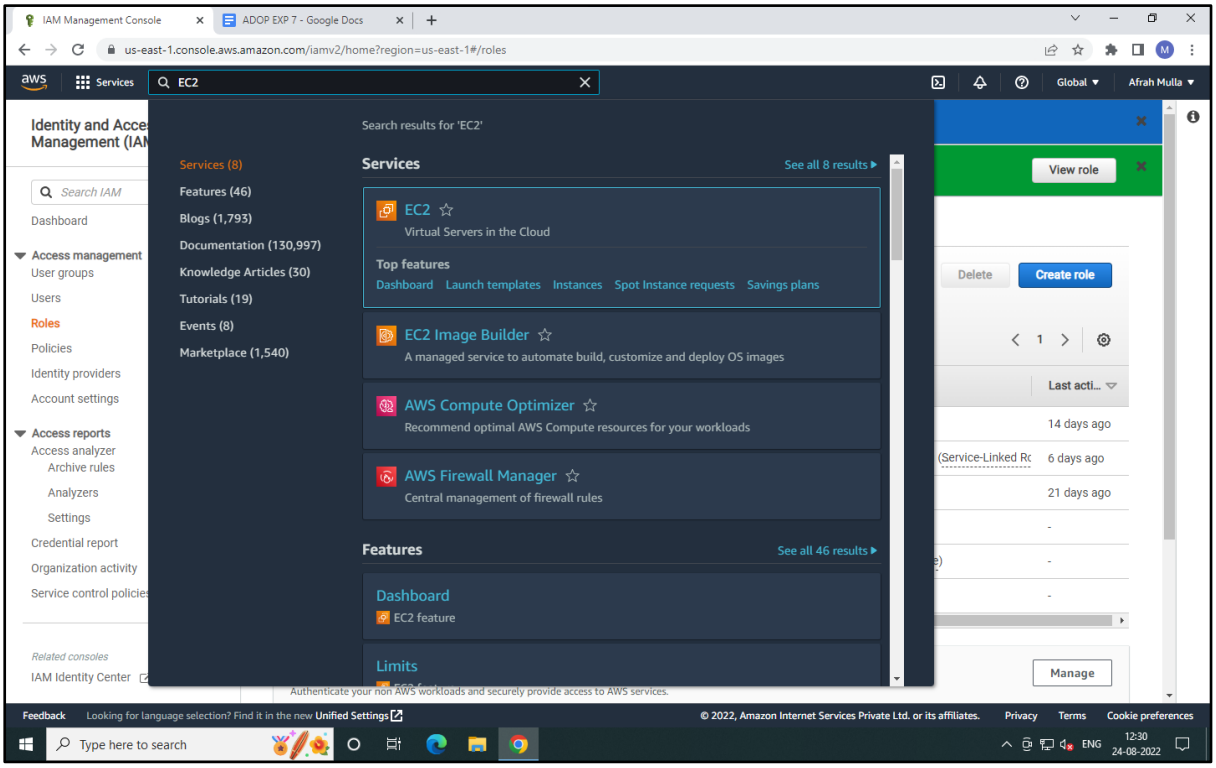
Step 7: Assign a name to your role -> Create role



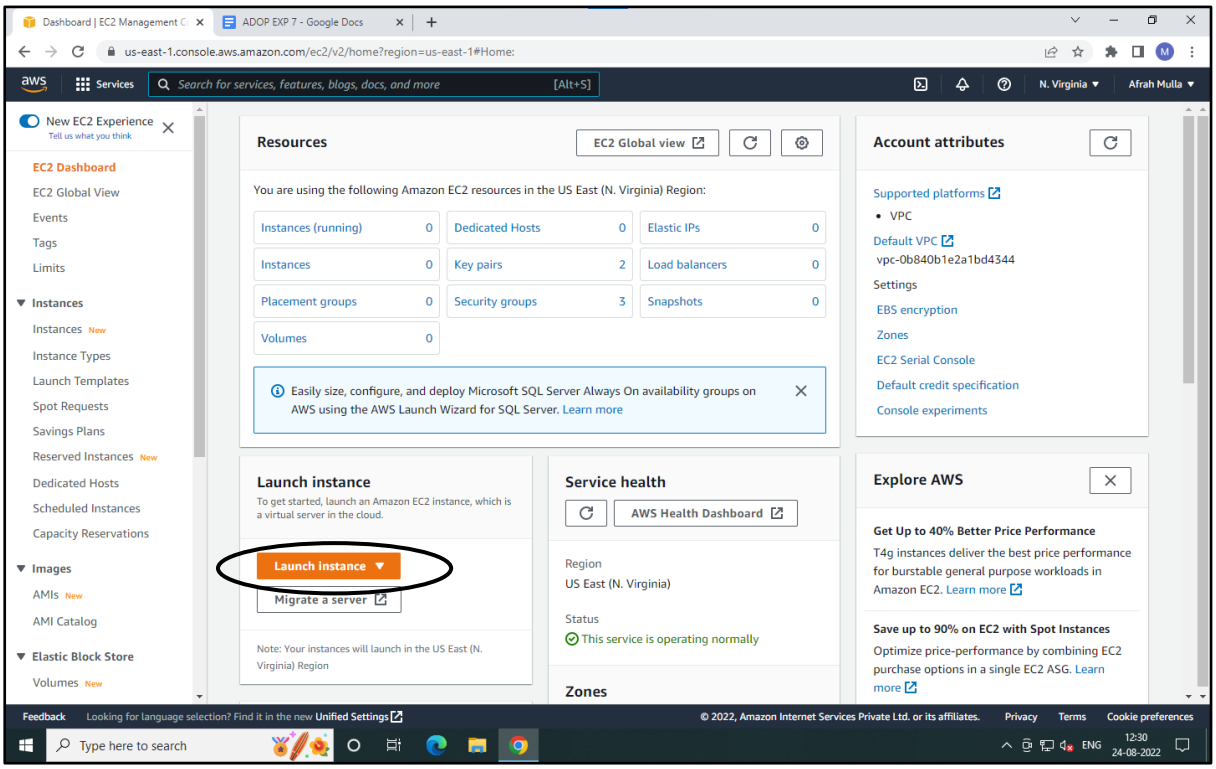




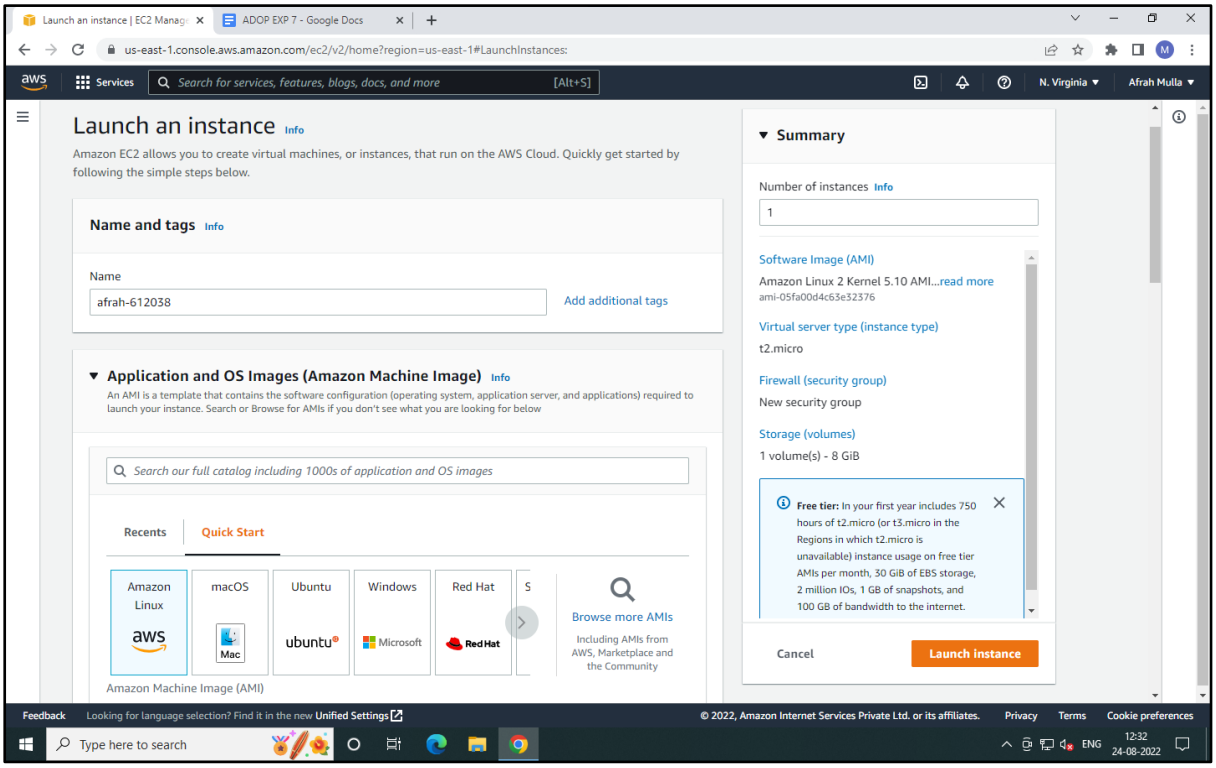
Step 8: Search for ‘EC2’ -> Select it



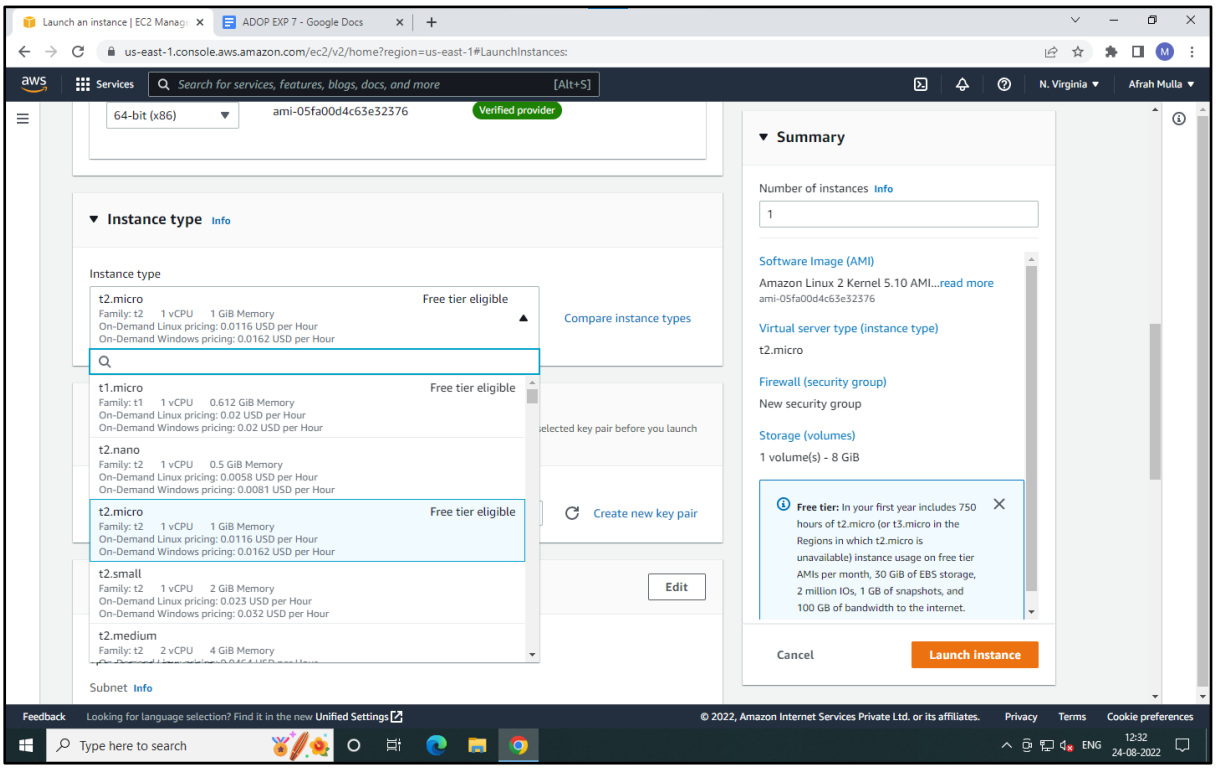
Step 9: Launch instance



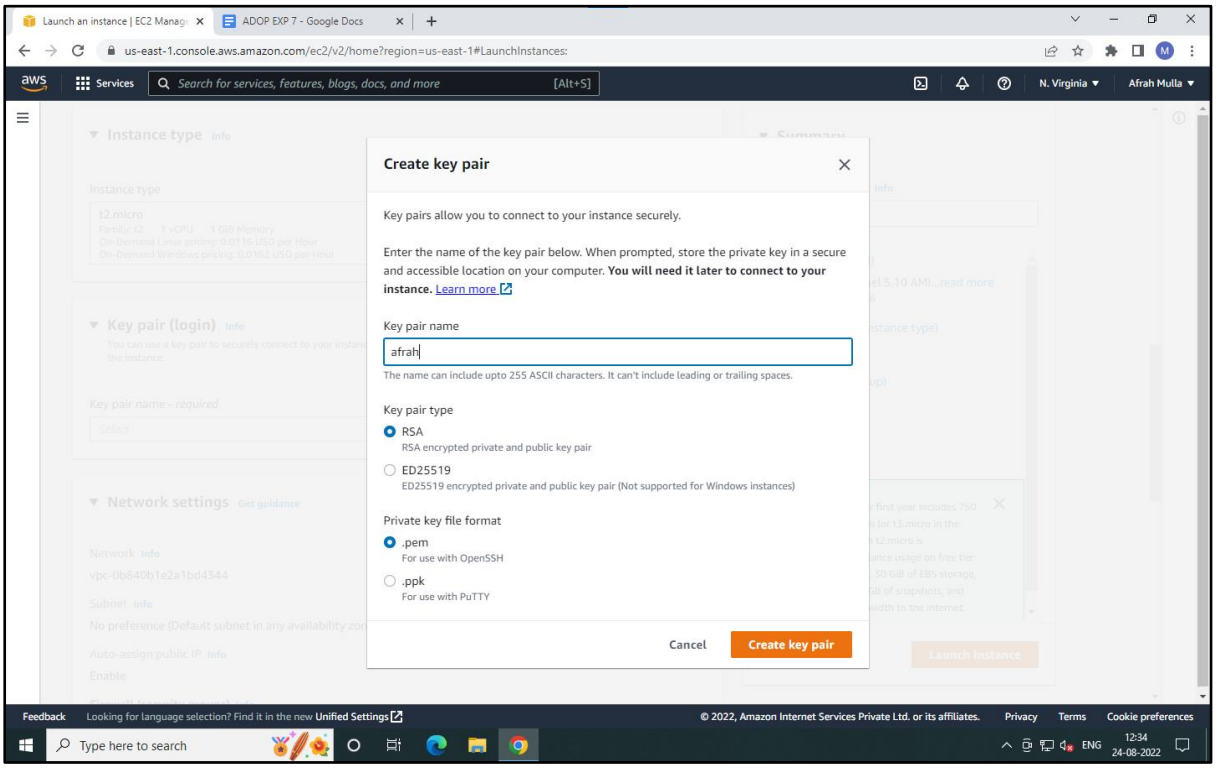
Step 10: Give a name to your instance and select Amazon Linux

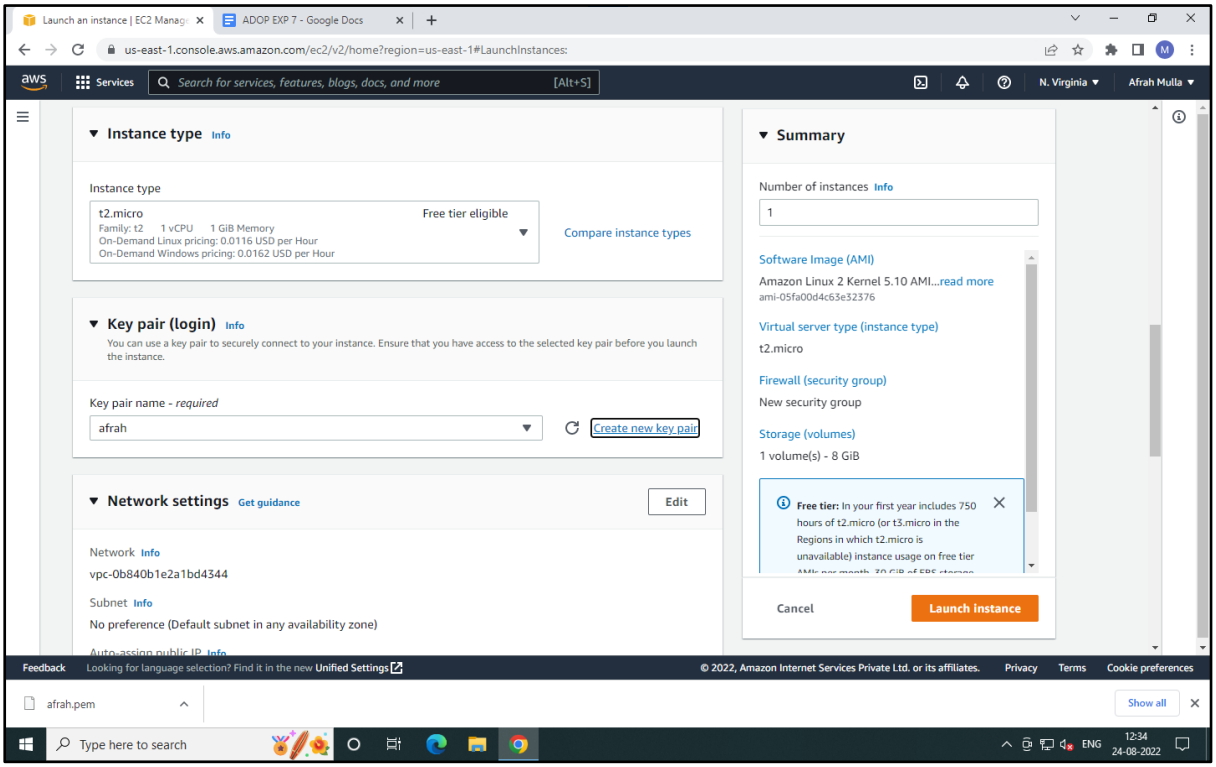


Step 11: Select instance type with free tier eligibility (t2.micro)

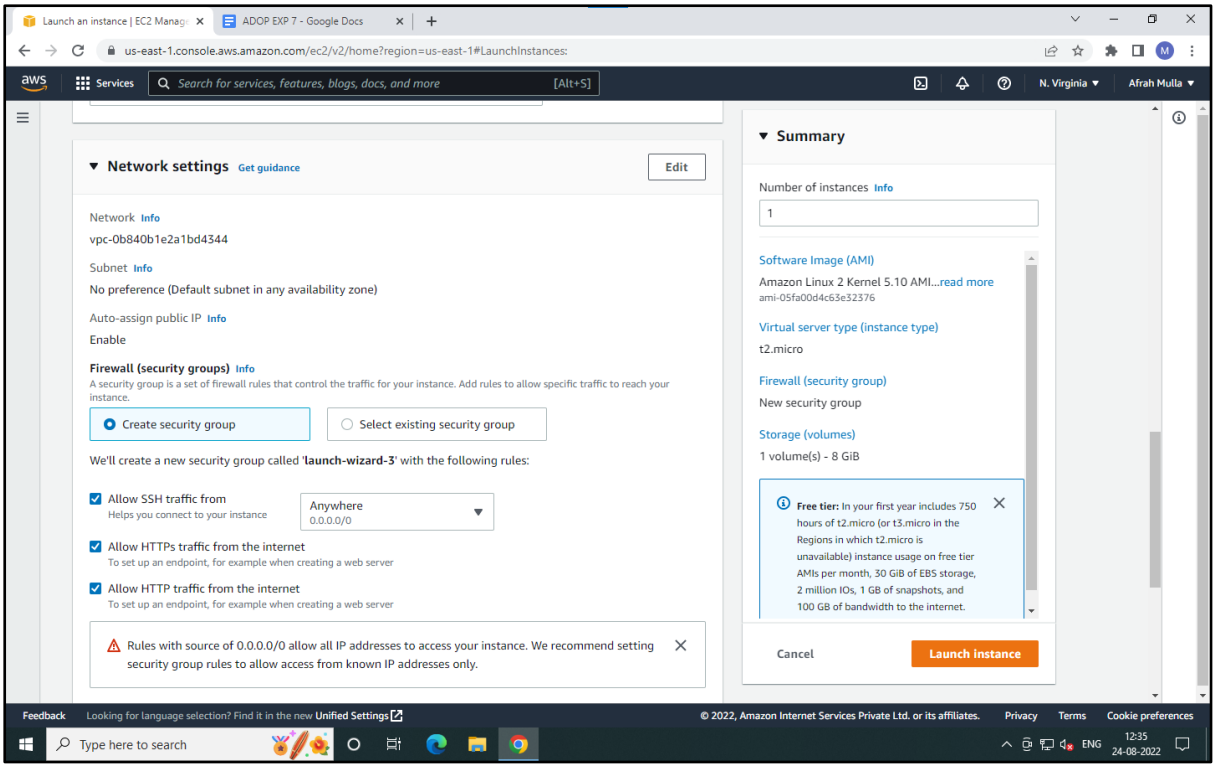


Step 12: Create key pair. A .pem file will be downloaded which will be later used to connect to the instance

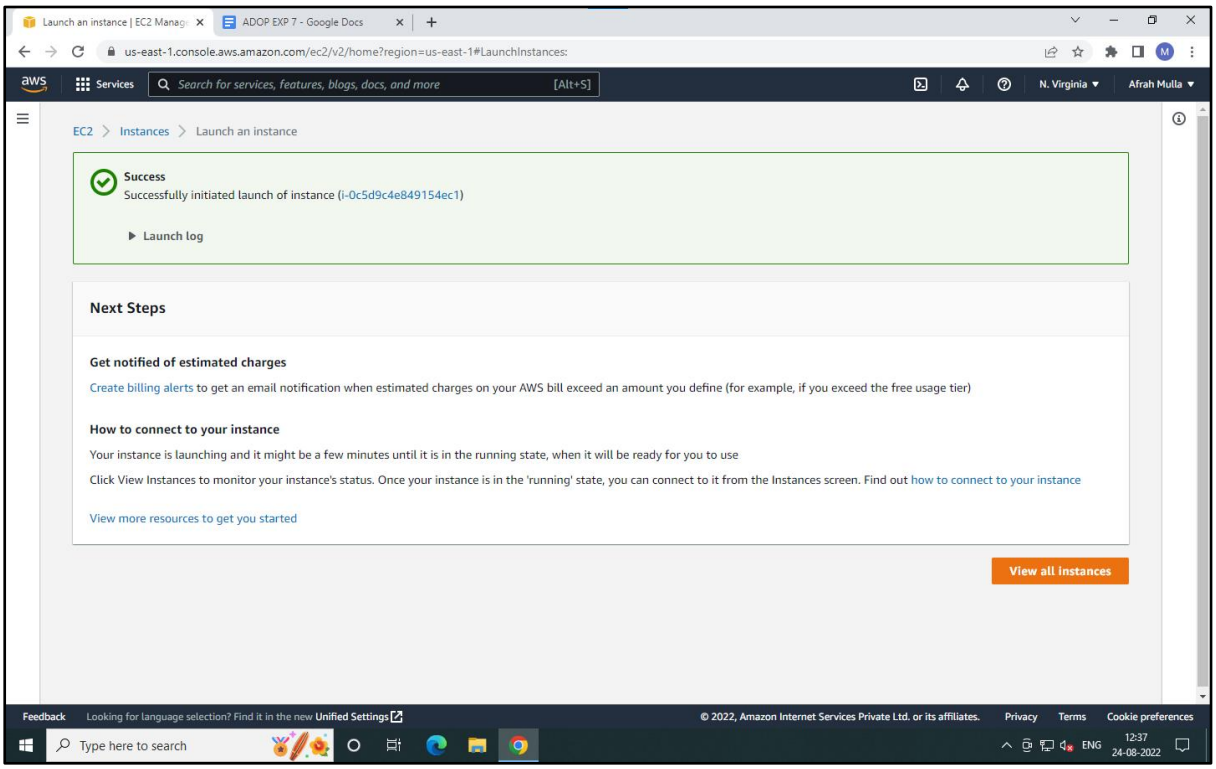




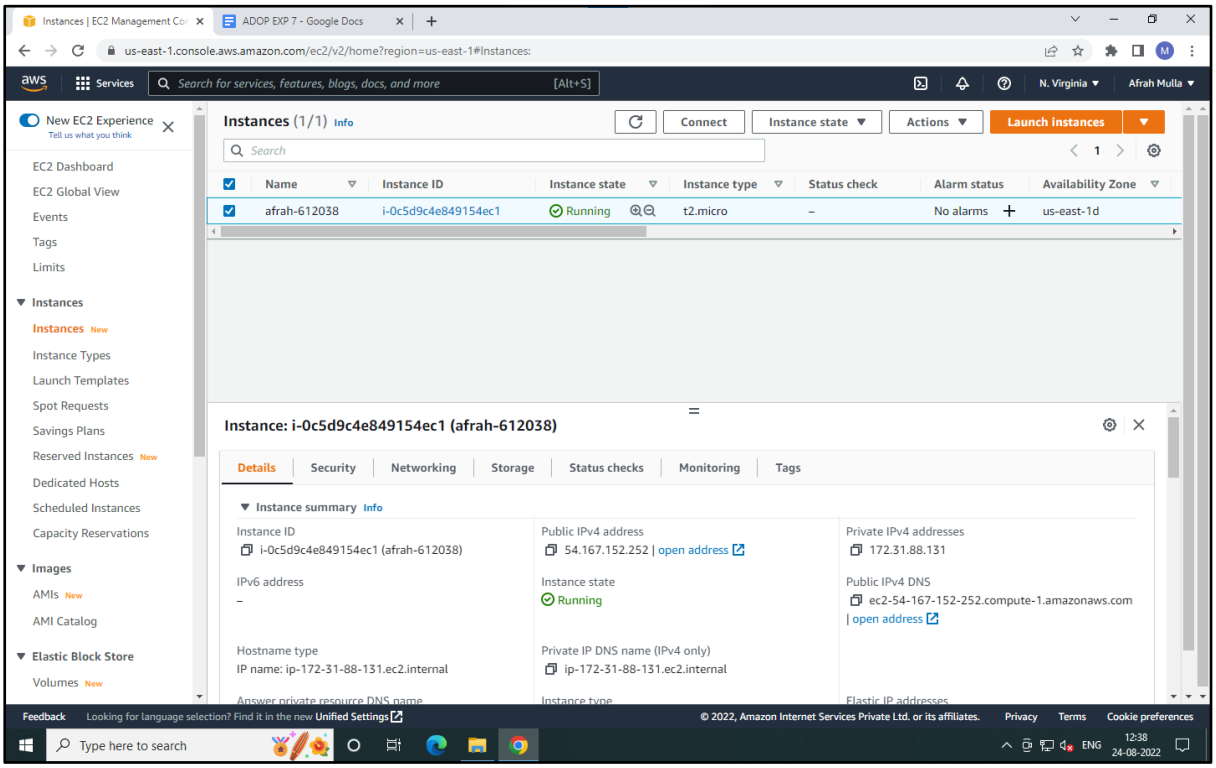
Step 13: Network Settings: Select ‘Allow SSH Traffic from’ Anywhere, Allow HTTPs traffic from the internet and Allow HTTP traffic from the internet. Then Launch Instance



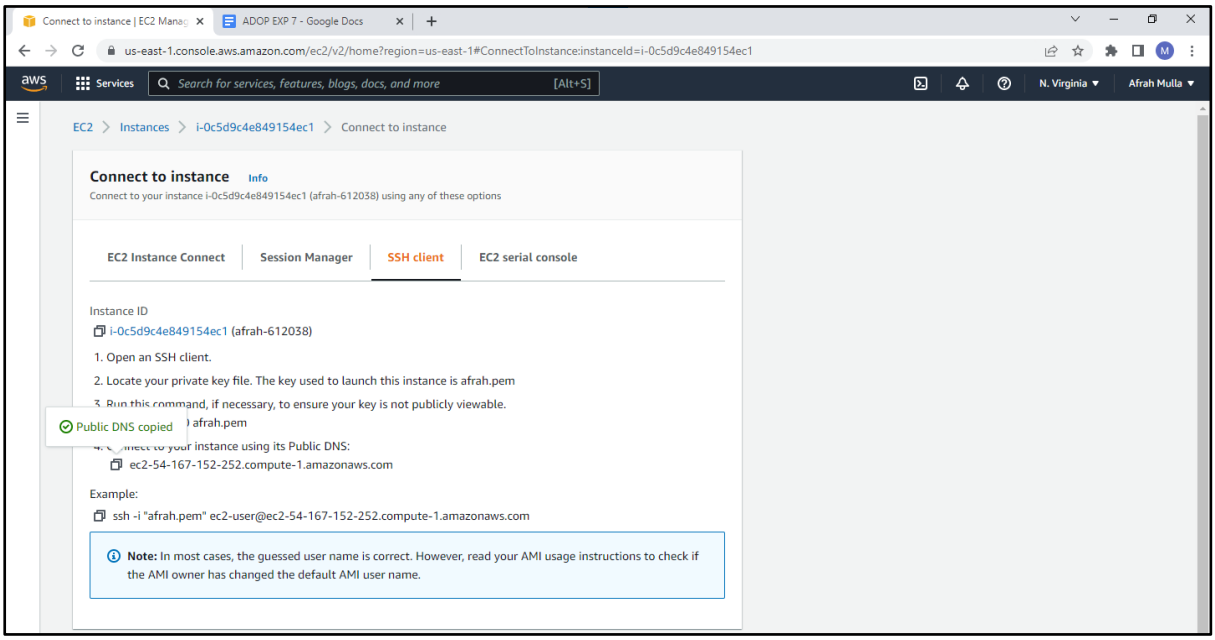
Success message will be shown after successful creation of instance



Step 14: Launch Linux instance to get the remote host and username for SSH in MobaXterm

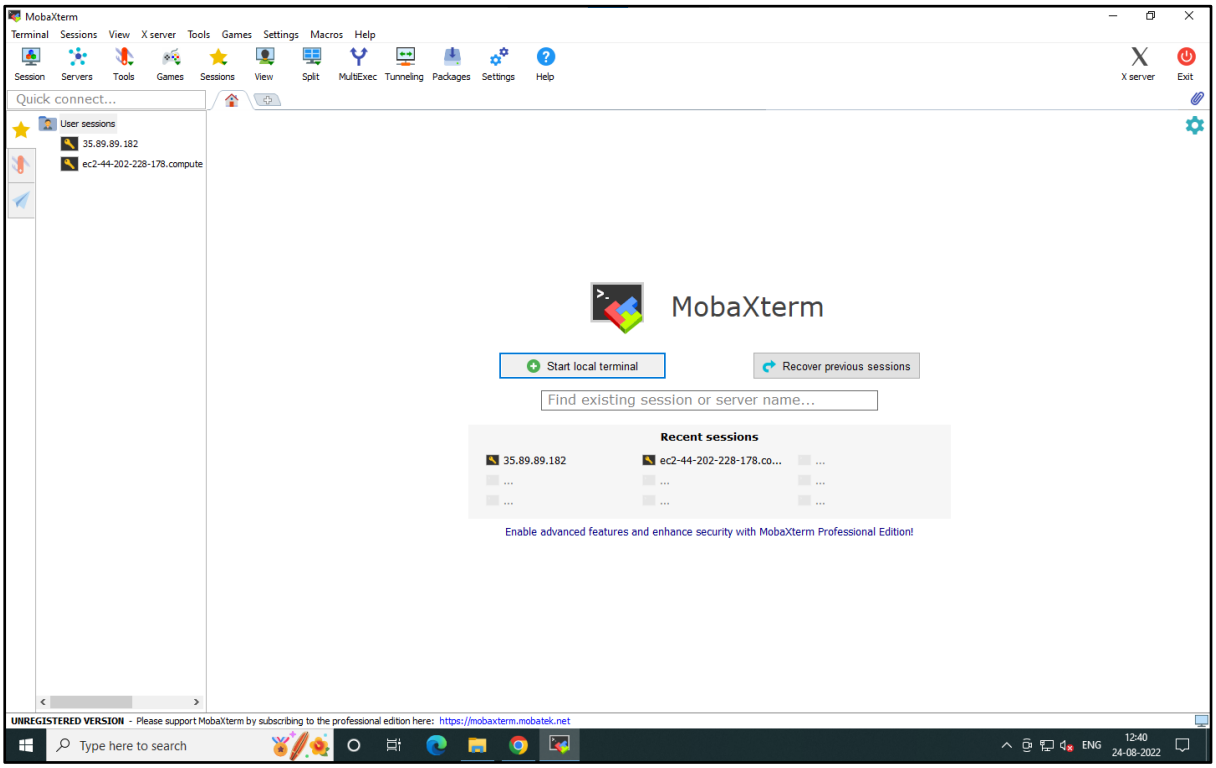


Go to SSH Client to get the remote host and username

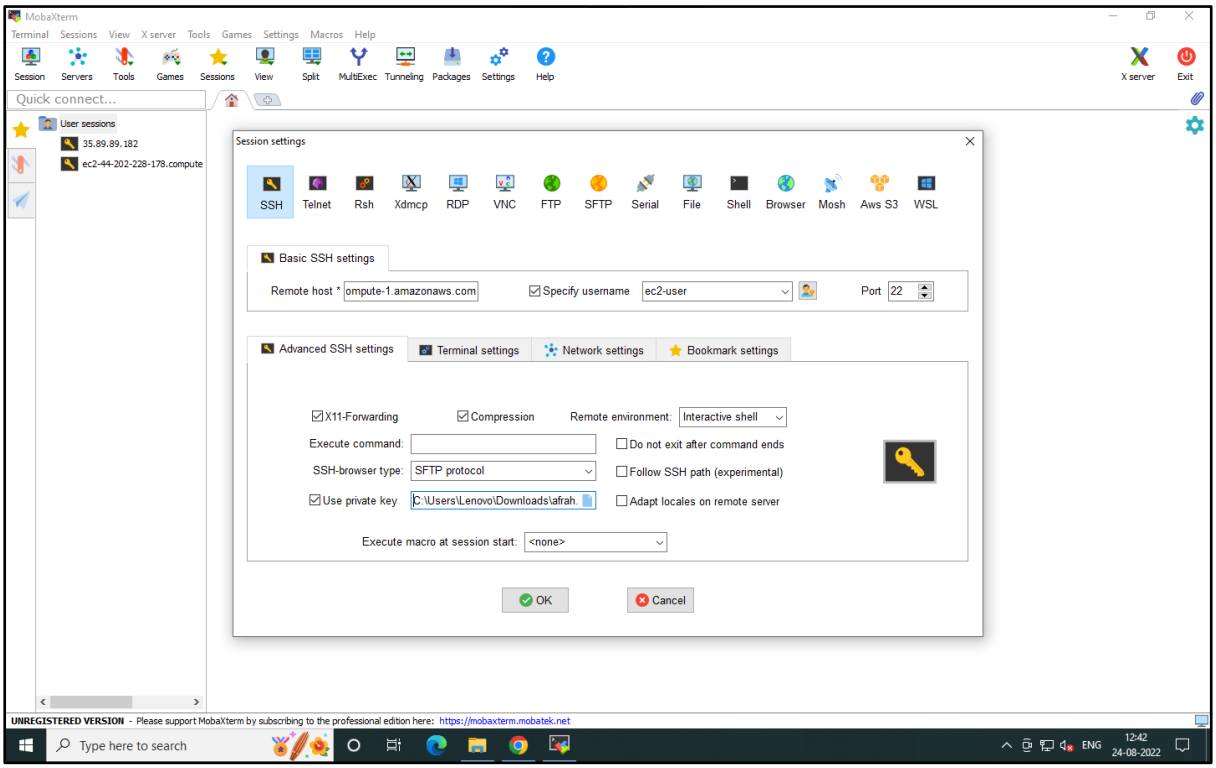


Copy the Public DNS which is your remote host and the username is the word before '@'

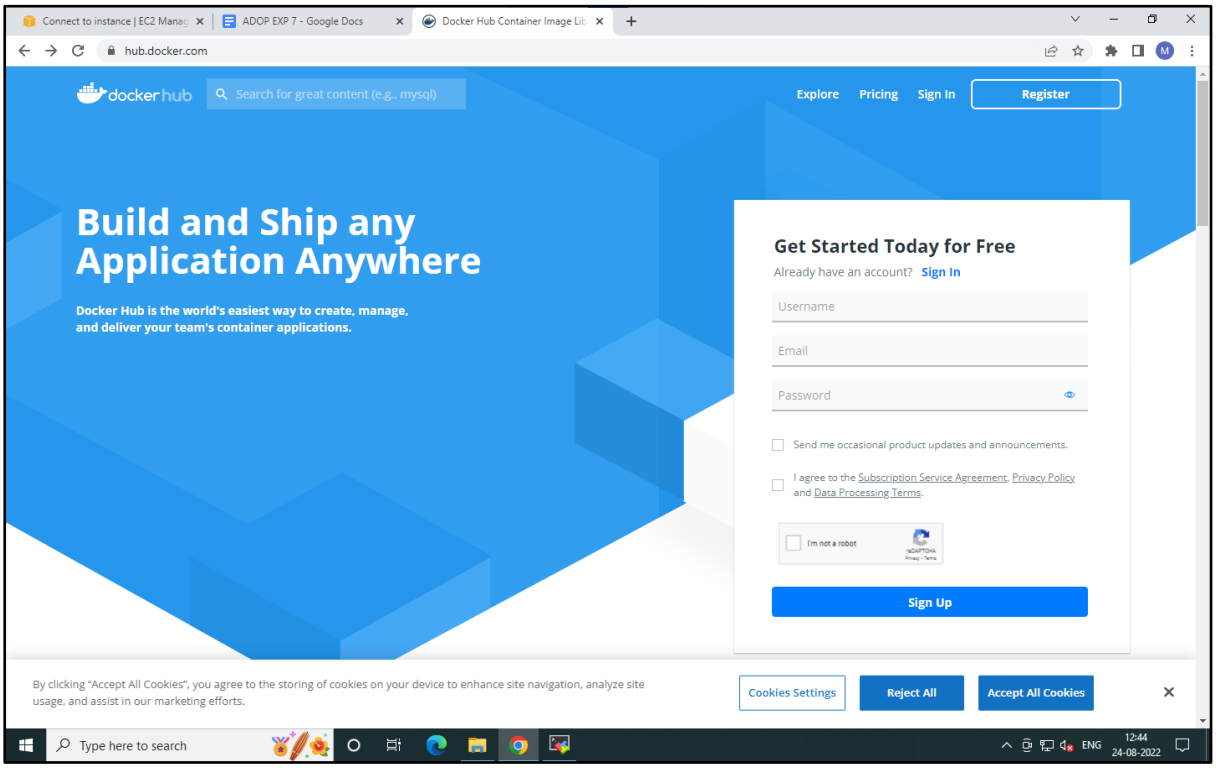
Step 15: Launch MobaXterm to connect the instance. Go to Sessions -> New Session

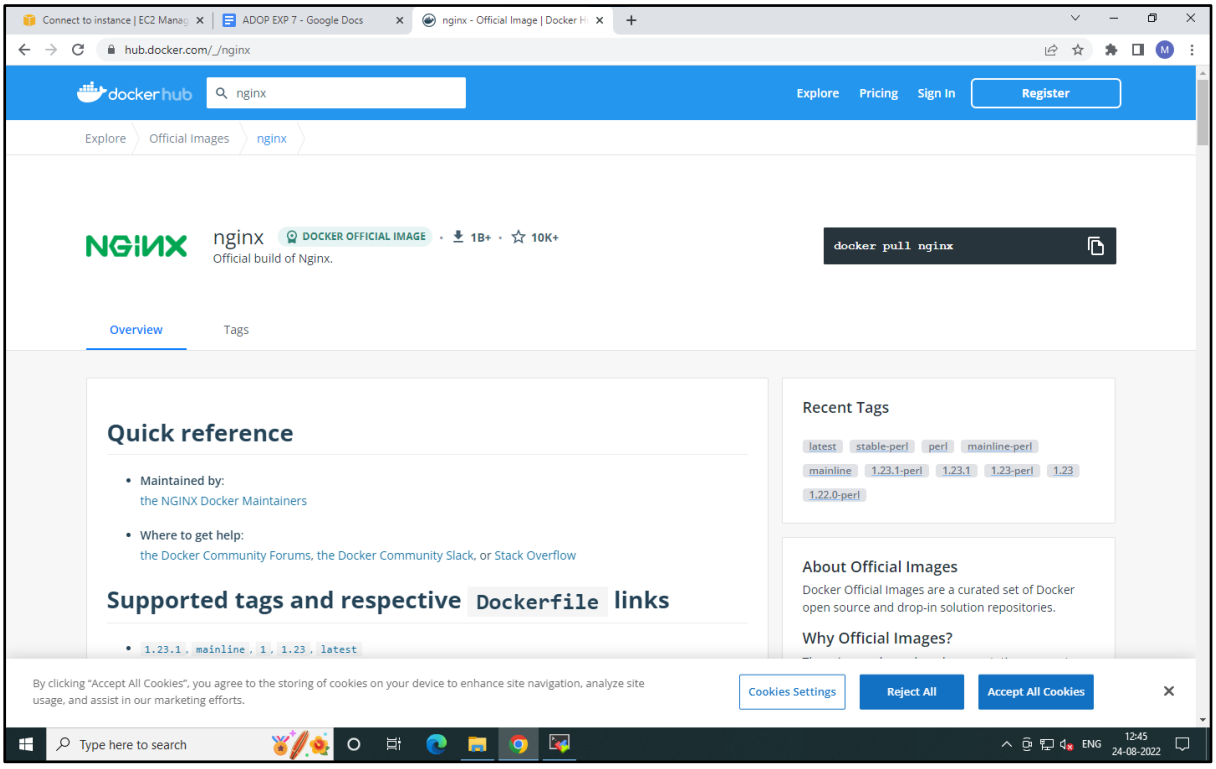


Then select SSH. Fill the basic SSH settings and attach the .pem file downloaded earlier in advanced SSH settings 'Use private key' section. Then OK. Your Linux Instance will be running



Step 16: Go to https://hub.docker.com/, search ‘nginx’ to get the commands to install Docker, pull nginx image and run it.

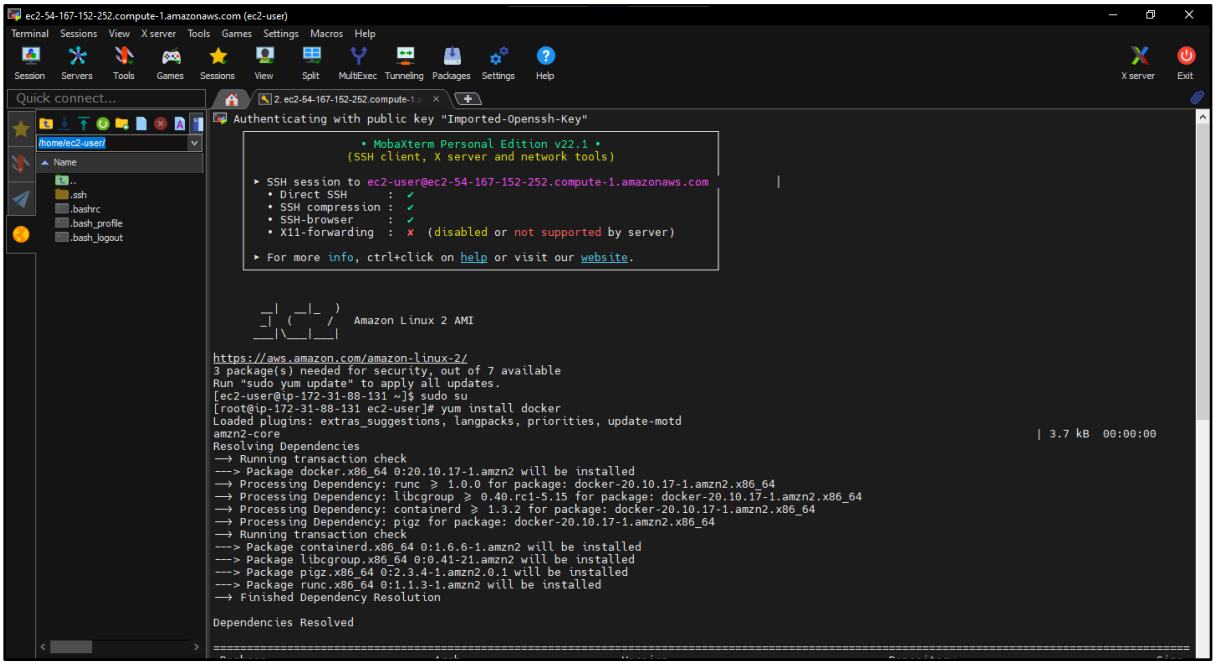


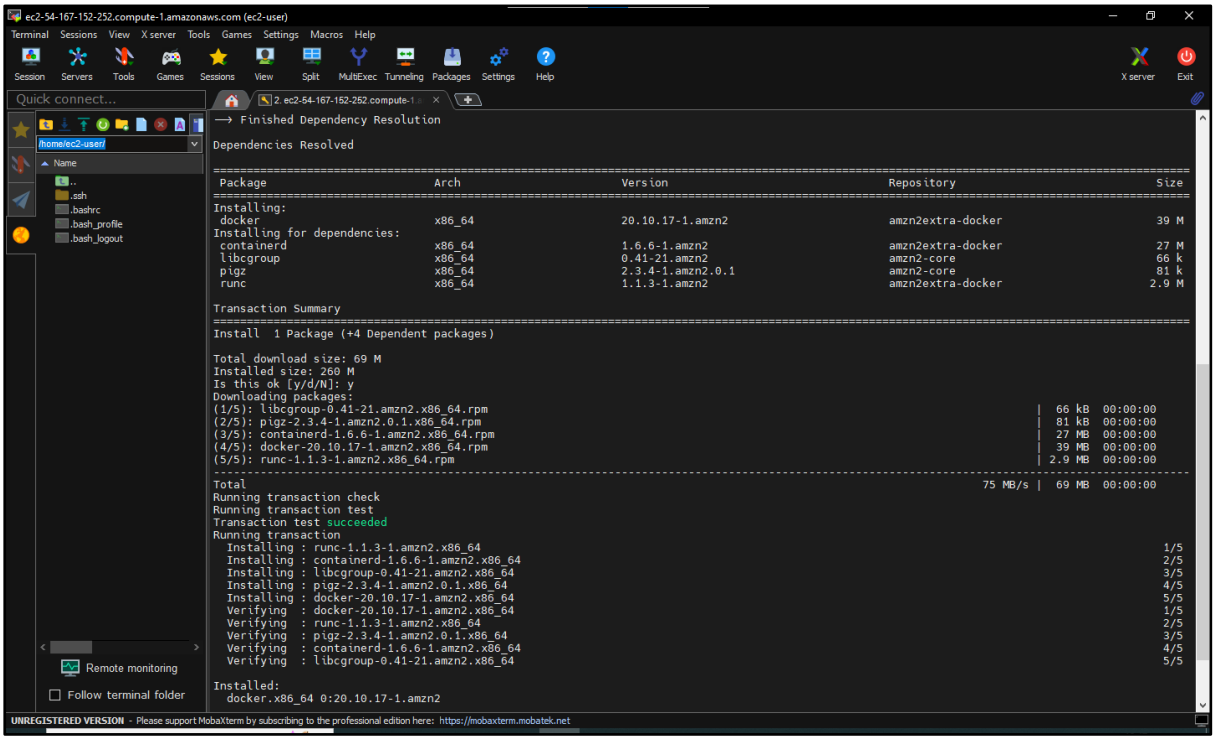


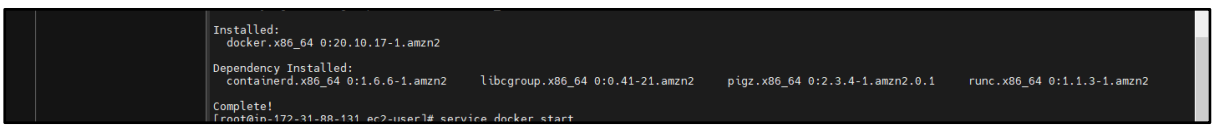
Execute the following commands –

sudo su

yum install docker

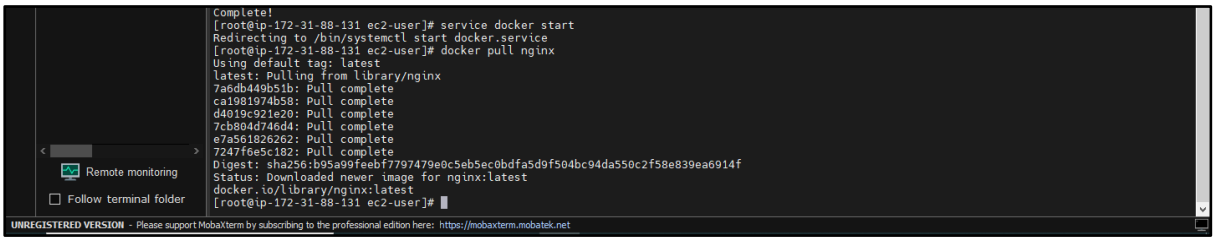




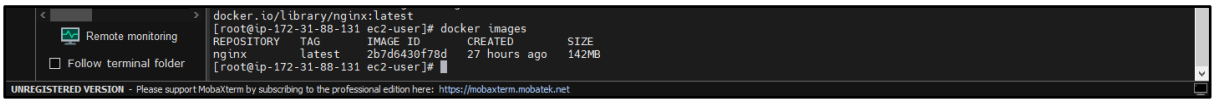


service docker start

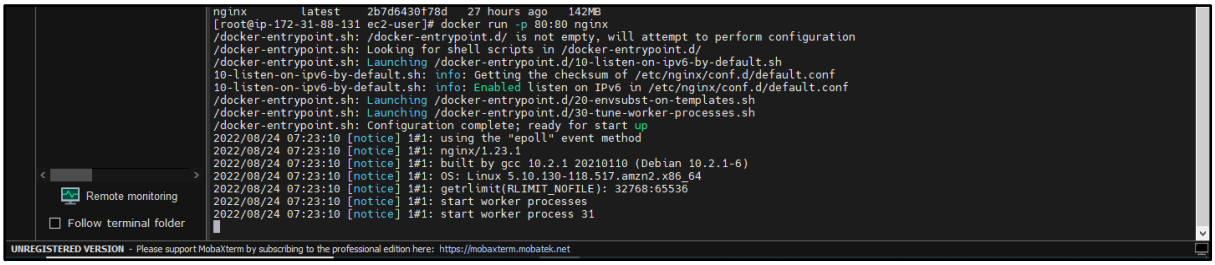
docker pull nginx



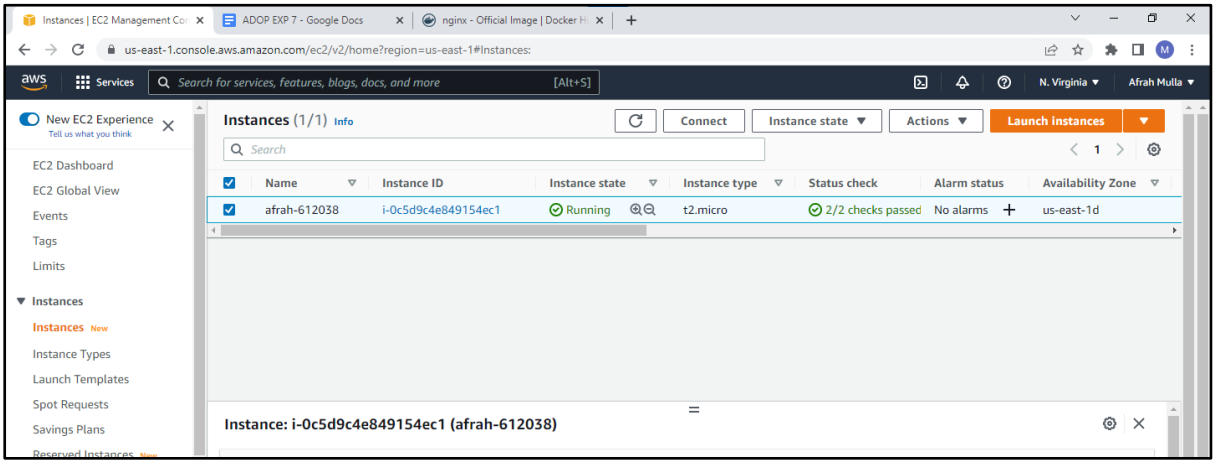
docker images

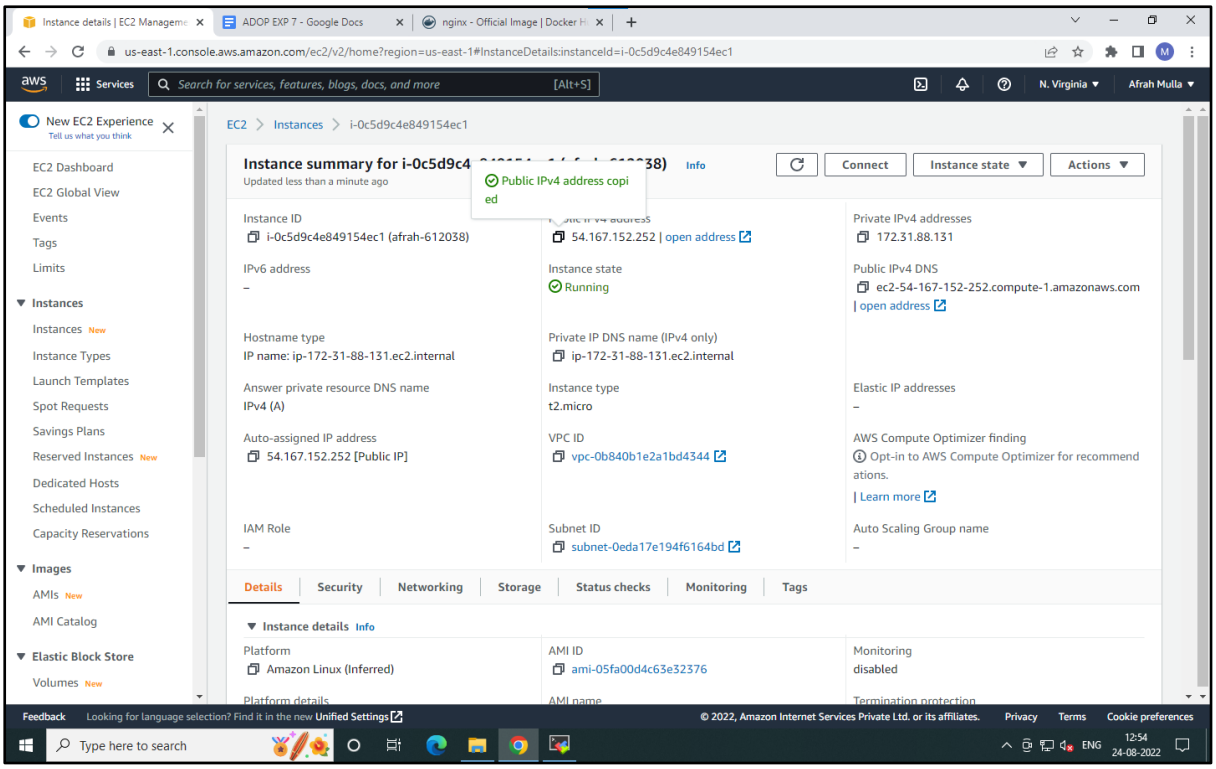


docker run -p 80:80 nginx

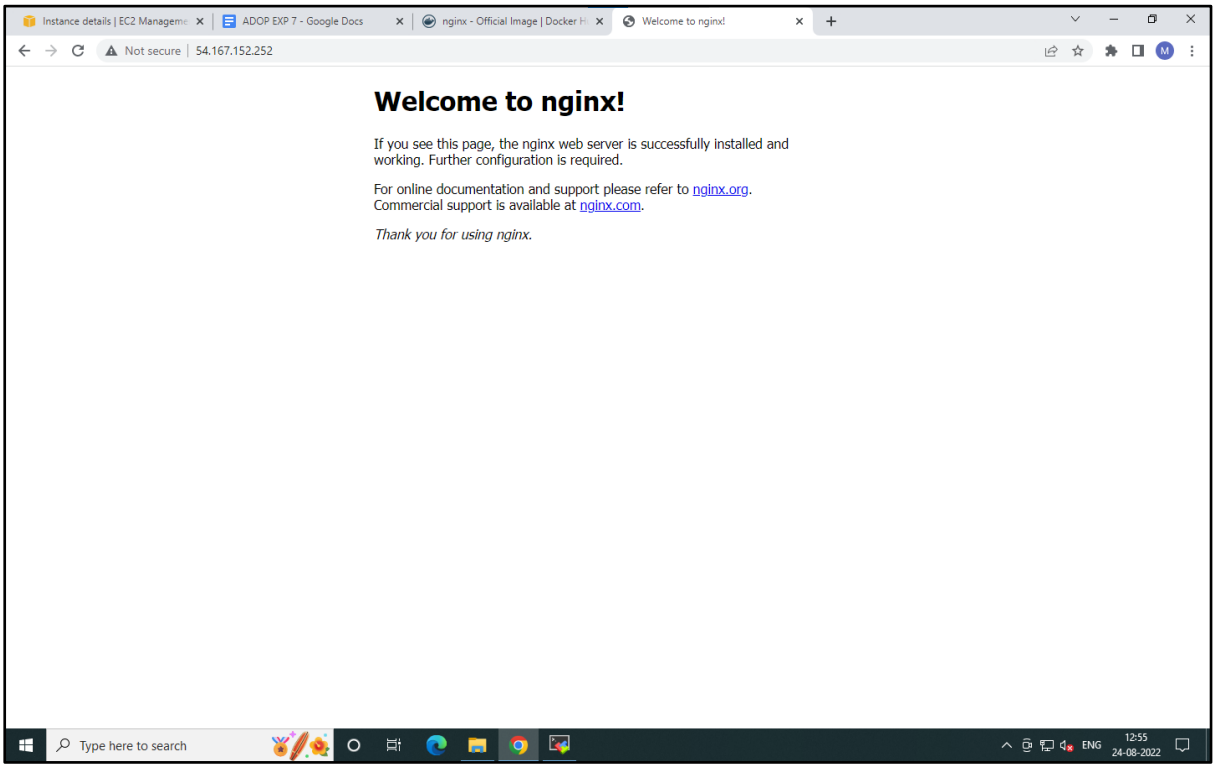


Step 17: To see if Nginx Web Server is successfully installed and working, go to Linux Instance and copy the 'Public IPv4 address'



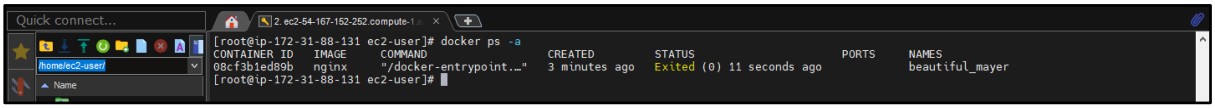


Step 18: Paste the Public IPv4 address in a web browser and the following web page will be displayed which means you have successfully installed Nginx Web Server

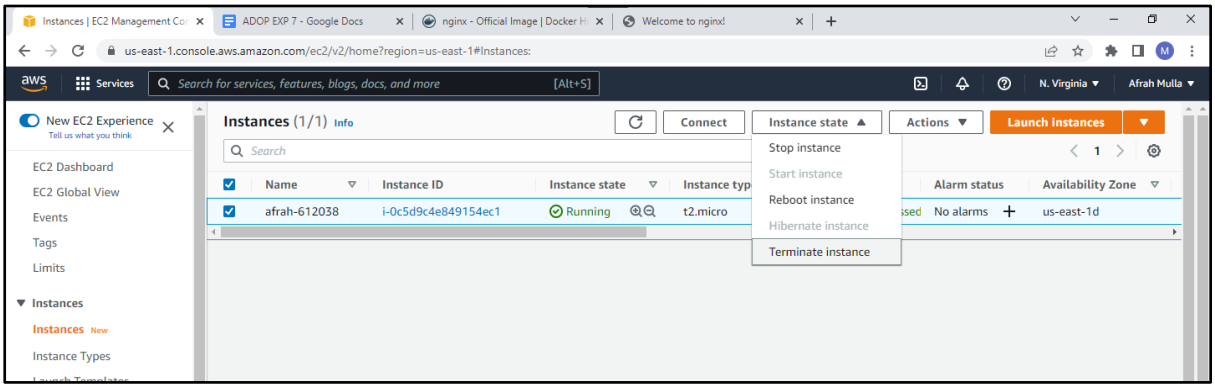


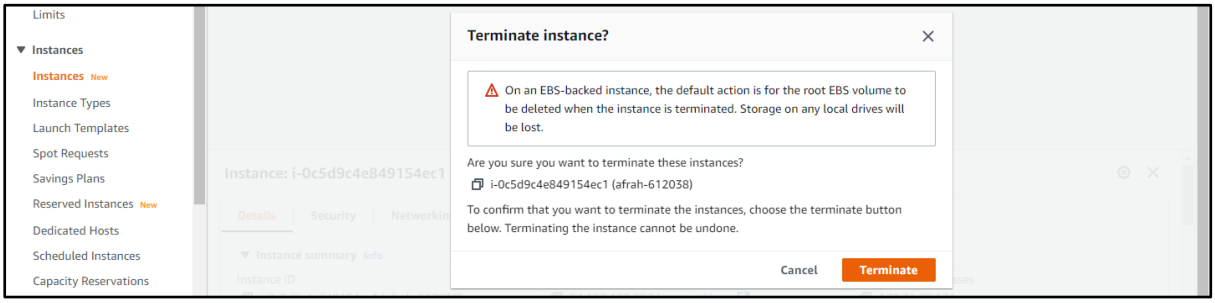
Step 19: To see the Container ID run the command –

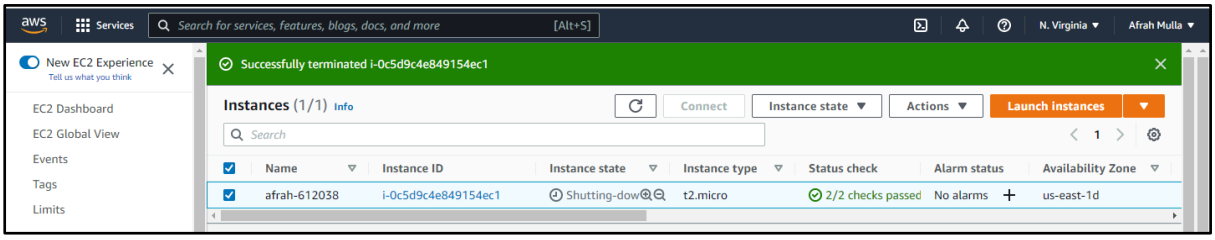
docker ps -a

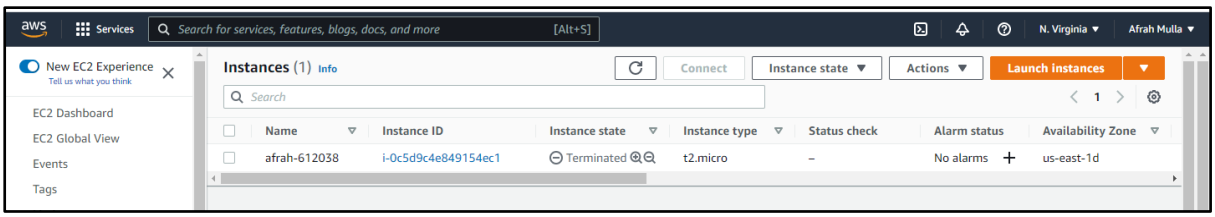


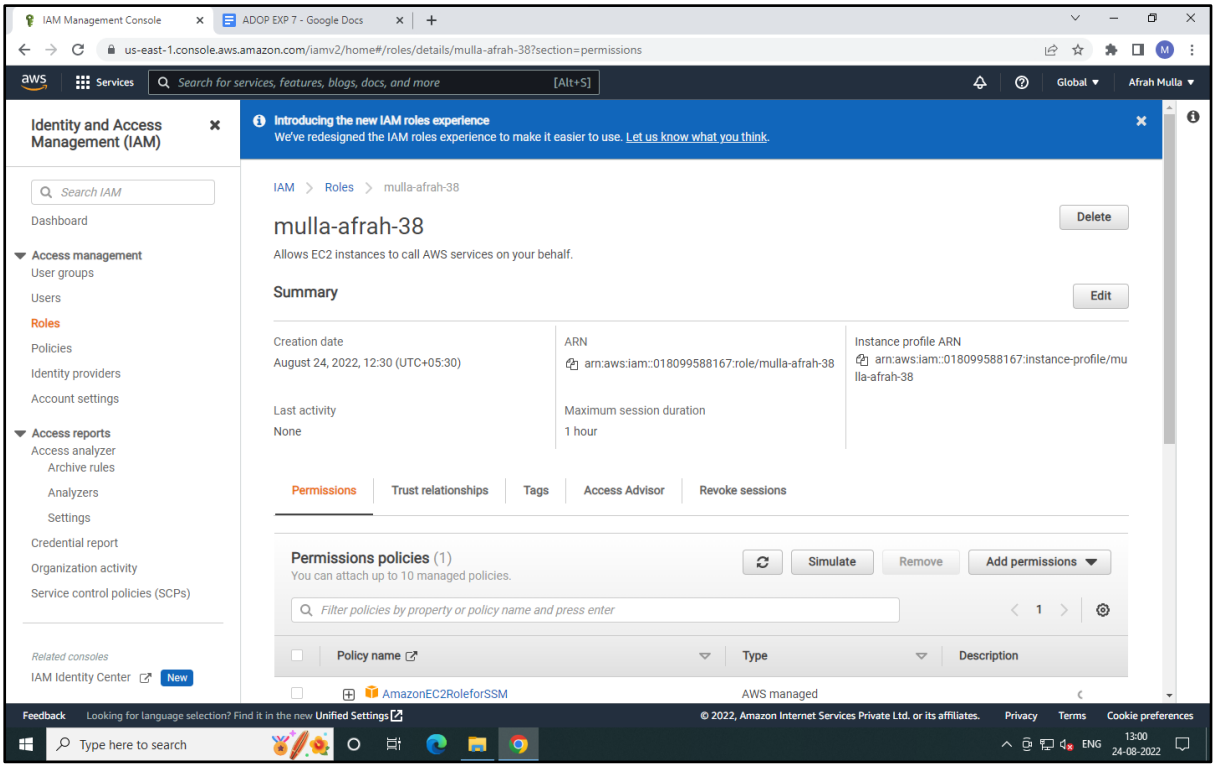
Step 20: Lastly, terminate the EC2 instance and delete the IAM role that you have created

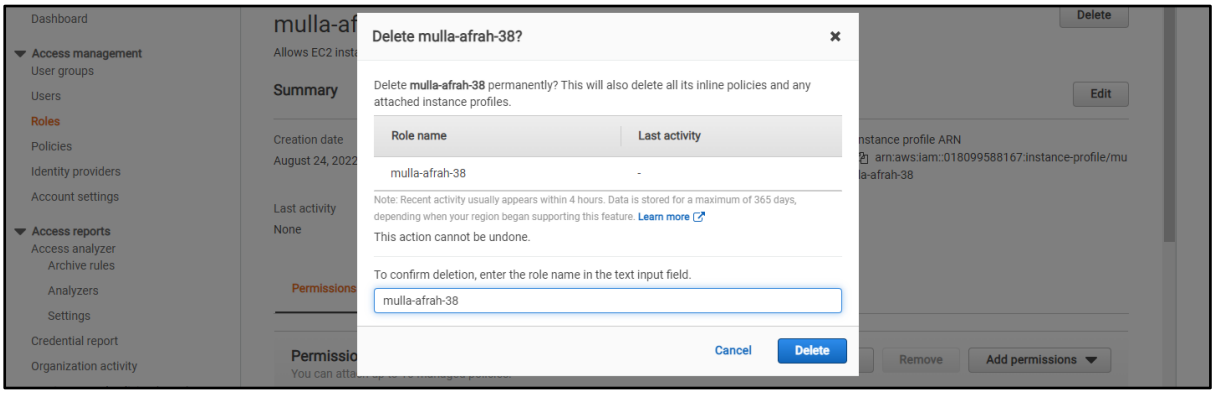


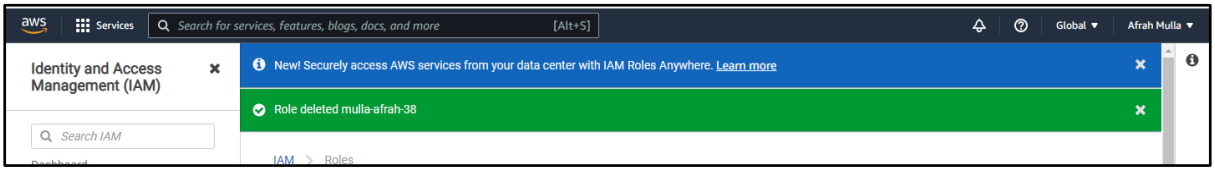







IAM Management Console
ADOP EXP 7 - Google Docs
us-east-1.console.aws.amazon.com/iamv2/home#/roles/details/mulla-afrah-38?section=permissions
Services
Search for services, features, blogs, docs, and more [Alt+S]
Global
Afrah Mulla
Identity and Access Management (IAM)
Introducing the new IAM roles experience
We've redesigned the IAM roles experience to make it easier to use. Let us know what you think.
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Access reports
Access analyzer
Archive rules
Analyzers
Settings
Credential report
Organization activity
Service control policies (SCPs)
Related consoles
IAM Identity Center
New
IAM > Roles > mulla-afrah-38
mulla-afrah-38
Delete
Allows EC2 instances to call AWS services on your behalf.
Summary
Edit
Creation date
August 24, 2022, 12:30 (UTC+05:30)
ARN
arn:aws:iam::018099588167:role/mulla-afrah-38
Instance profile ARN
arn:aws:iam::018099588167:instance-profile/mulla-afrah-38
Last activity
None
Maximum session duration
1 hour
Permissions
Trust relationships
Tags
Access Advisor
Revoke sessions
Permissions policies (1)
You can attach up to 10 managed policies.
Simulate
Remove
Add permissions
Filter policies by property or policy name and press enter
Policy name
Type
Description
AmazonEC2RoleforSSM
AWS managed
Feedback
Looking for language selection? Find it in the new Unified Settings
© 2022, Amazon Internet Services Private Ltd. or its affiliates.
Privacy
Terms
Cookie preferences
Type here to search
ENG
13:00
24-08-2022


Delete mulla-afrah-38?
Delete mulla-afrah-38 permanently? This will also delete all its inline policies and any attached instance profiles.
Role name
Last activity
mulla-afrah-38
-
Note: Recent activity usually appears within 4 hours. Data is stored for a maximum of 365 days, depending when your region began supporting this feature. Learn more
This action cannot be undone.
To confirm deletion, enter the role name in the text input field.
mulla-afrah-38
Cancel
Delete


New! Securely access AWS services from your data center with IAM Roles Anywhere. Learn more
Role deleted mulla-afrah-38
IAM > Roles