Docker vs VM

Docker containers and virtual machines are both ways of deploying applications inside environments that are isolated from the underlying hardware.

Introduction

DOCKER	VM
➔ Docker is a software solution for performing virtualization.	➔ A virtual machine is a system which acts exactly like a computer.
➔ It binds application and its dependencies inside a container.	➔ A VM is a simulated computer system.
➔ Docker provides the ability to package and run an application in a loosely isolated environment called a container.	➔ Virtual machines have a **host operating system** and a **guest operating system** inside each VM. The guest OS can be any OS, such as Linux or Windows,
➔ Containers allow a developer to package up an application with all its necessary parts, such as libraries and other dependencies and deploy it as one package.	➔ VMs run as virtual environments on the same hardware.
➔ Allows you to run many containers simultaneously on a given host.	➔ It runs in a “sandboxed” environment on a computer or server.

Benefits

DOCKER	VM
➔ Docker containers share the host operating system, and that is why they are **lightweight.**	➔ Multiple OS environments can exist simultaneously on the same host machine, isolated from each other.
➔ Fast, consistent delivery of your applications	➔ **More secure** ➔ Sandboxed environments do not have direct access to their host system's operating system (OS), files, or hardware. 1. Strong isolation in the host kernel 2. Does not share operating system. 3. You don't get direct access to the resources, and hypervisor is there to restrict the usage of resources.
➔ Scaling up and duplicating a Docker container is simple and easy	
➔ Less resource and memory intensive	
➔ Process-isolated and does not require a hardware hypervisor.	

Disadvantages

DOCKER	VM
➔ Not secure 1. A single infected application can hack the entire host system. 2. A container has a lot of security risks, and vulnerabilities because they have shared host kernel.	➔ Each virtual machine has its guest operating system above the host operating system, which makes virtual machines heavy.
➔ Docker has a complex usage mechanism consisting of both	➔ VMs are isolated from their OS, and so they are not portable across multiple

third party and docker managed tools.	platforms without incurring compatibility issues.
	➔ Requires entire OS to be loaded before starting the surface, so less efficient.
	➔ High overhead
	➔ Deployment is lengthy as separate instances are responsible for execution.
	➔ Resource-intensive
	➔ Inefficient hypervisor and long boot uptime

Use cases

DOCKER	**VM**
➔ When you want to run multiple applications over a single operating system kernel.	➔ If you have applications or servers that need to run on different operating system versions.
➔ Containers are great for continuous integration and continuous delivery (CI/CD) workflows.	➔ Applications needing more privileges and security run on virtual machines.
➔ For development purposes where the applications must be developed and tested in different platforms/ if you want your application to be portable.	➔ Considered a suitable choice in a production environment since they run on their own OS without being a threat to the host computer.
➔ Docker lets you run each microservice that makes up an application in its own container.	

EC2 INSTANCES

1. What are EC2 instances?
   - ➔ EC2 stands for : Elastic Compute Cloud
   - ➔ It is a virtual machine in the cloud.
   - ➔ It is used to run applications on AWS infrastructure.
   - ➔ Provides secure, resizable compute capacity in the cloud.

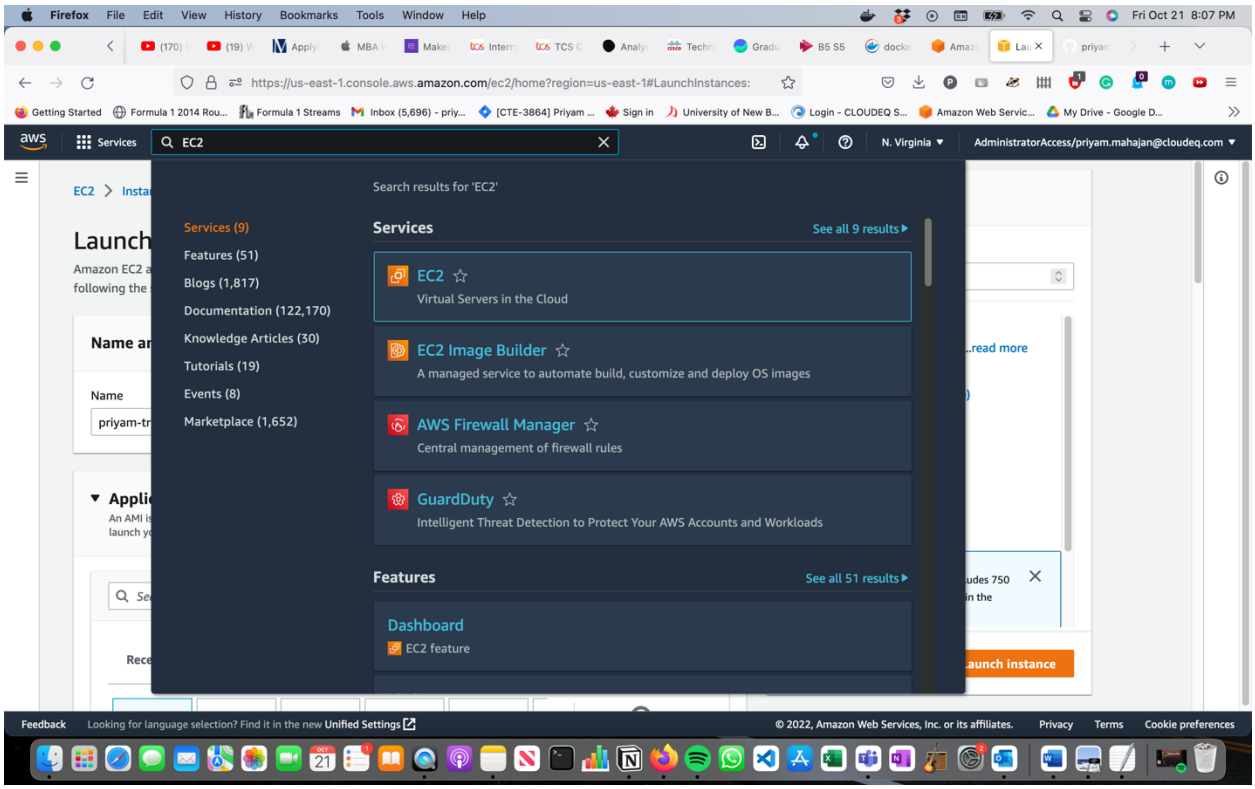
2. Why do we use EC2 instances? /Applications of EC2
   - ➔ Amazon EC2 eliminates your need to invest in hardware up front, so you can develop and deploy applications faster.
   - ➔ Amazon EC2 is used to launch virtual servers
   - ➔ Amazon EC2 is used to configure security and networking
   - ➔ Amazon EC2 is used to manage storage.

3. Pricing of EC2
   - ➔ Pricing is per instance-hour consumed for each instance, from the time an instance is launched until it is terminated or stopped.
   - ➔ Rates depend upon location time and region as well as operating system, instance type and vCPU.

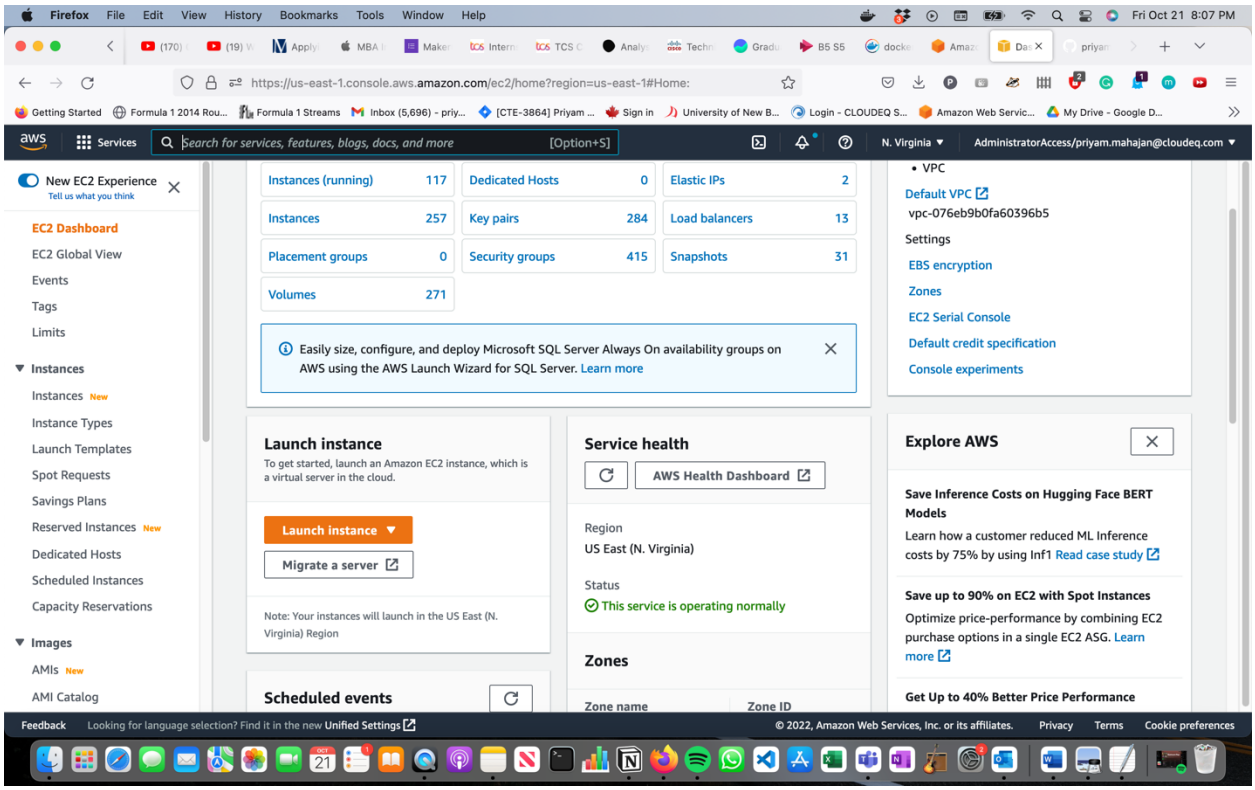
4. Features of EC2 instances
   - ➔ Instances *:* Virtual computing environments
   - ➔ *Amazon Machine Images (AMIs*) : Preconfigured templates which can be used for instances.
   - ➔ instance types: Various configurations of CPU, memory, storage, and networking capacity for your instances
   - ➔ key pairs: Store secure login information for your instances
   - ➔ *Regions* and *Availability Zones:* Multiple physical locations for your resources, such as instances and Amazon EBS volumes
   - ➔ Security groups: A firewall that enables you to specify the protocols, ports, and source IP ranges that can reach your instances

5. Practical demonstration of creating EC2 instances.
   Steps to creating an EC2 instance.

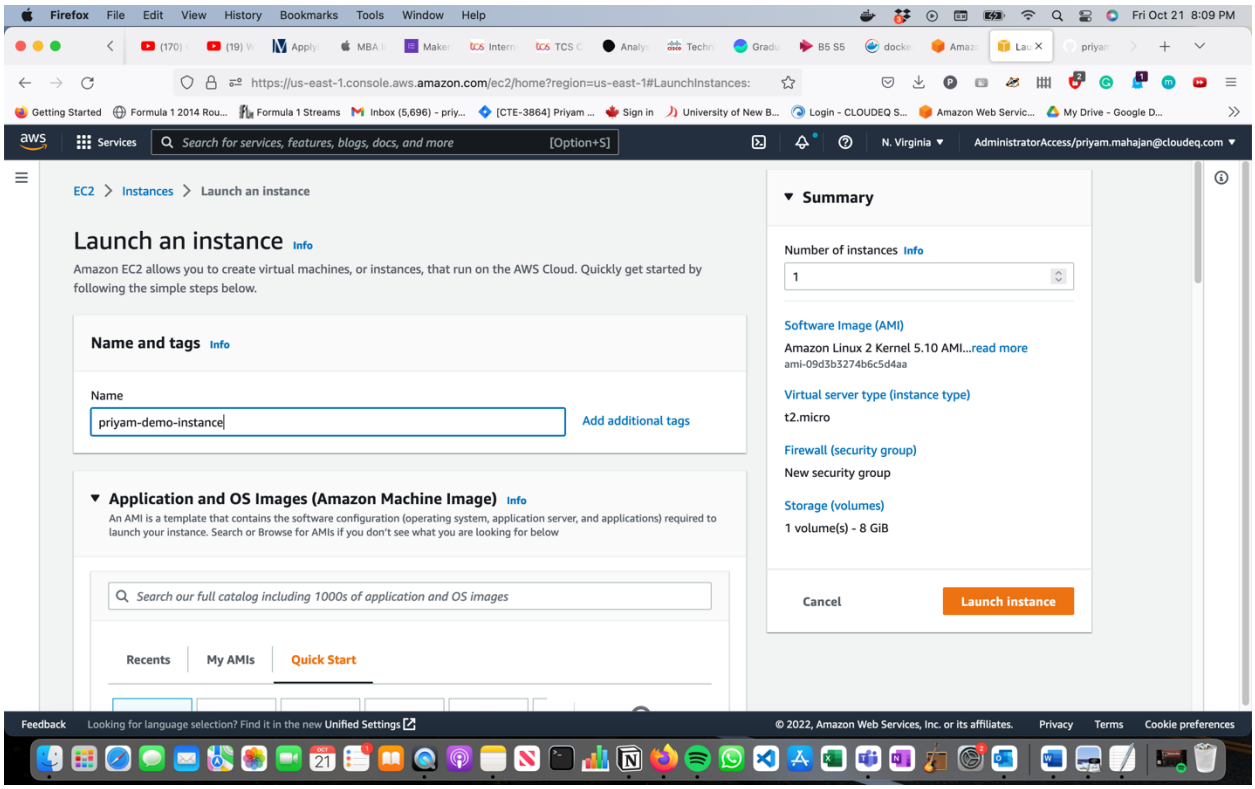
   1. Log in to AWS account.
   2. In the search bar , search for “EC2”



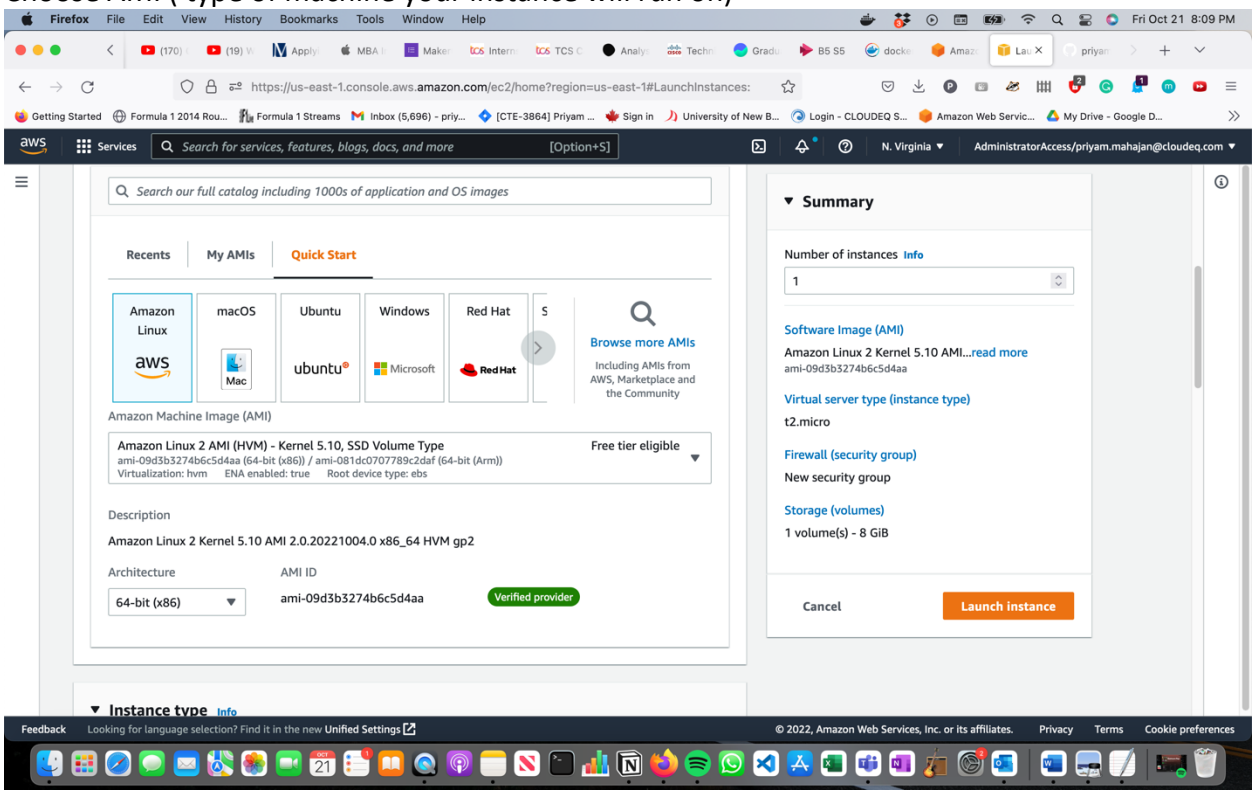
3. Click on ‘Launch instance’



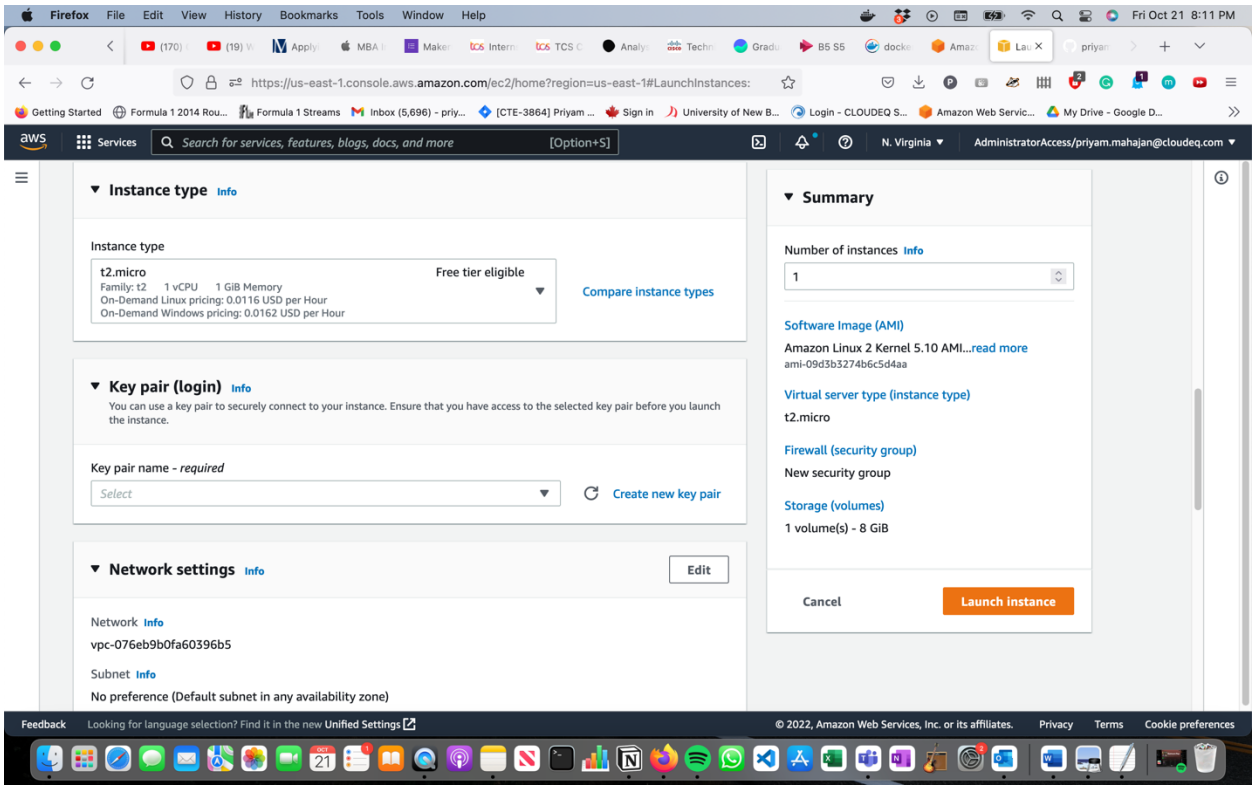
4. Add the name of your instance and tags required.



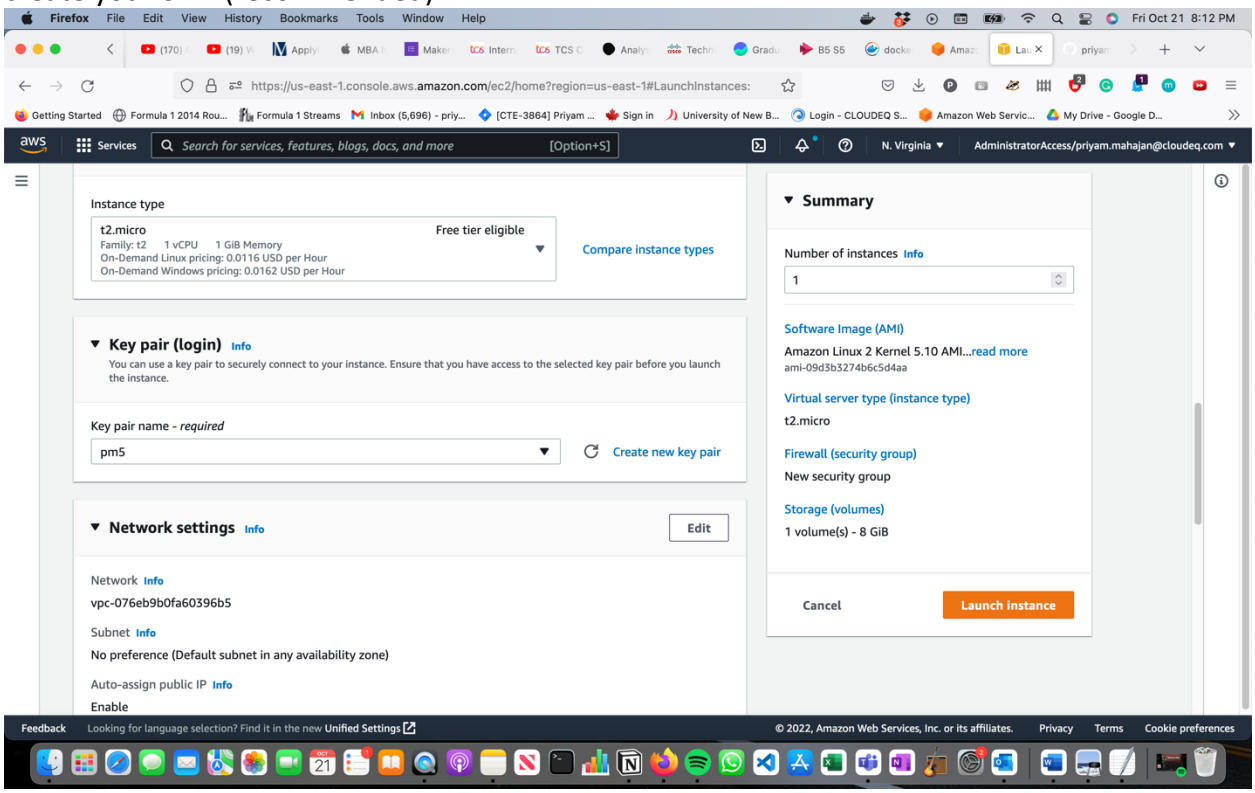
5. Choose AMI (type of machine your instance will run on)



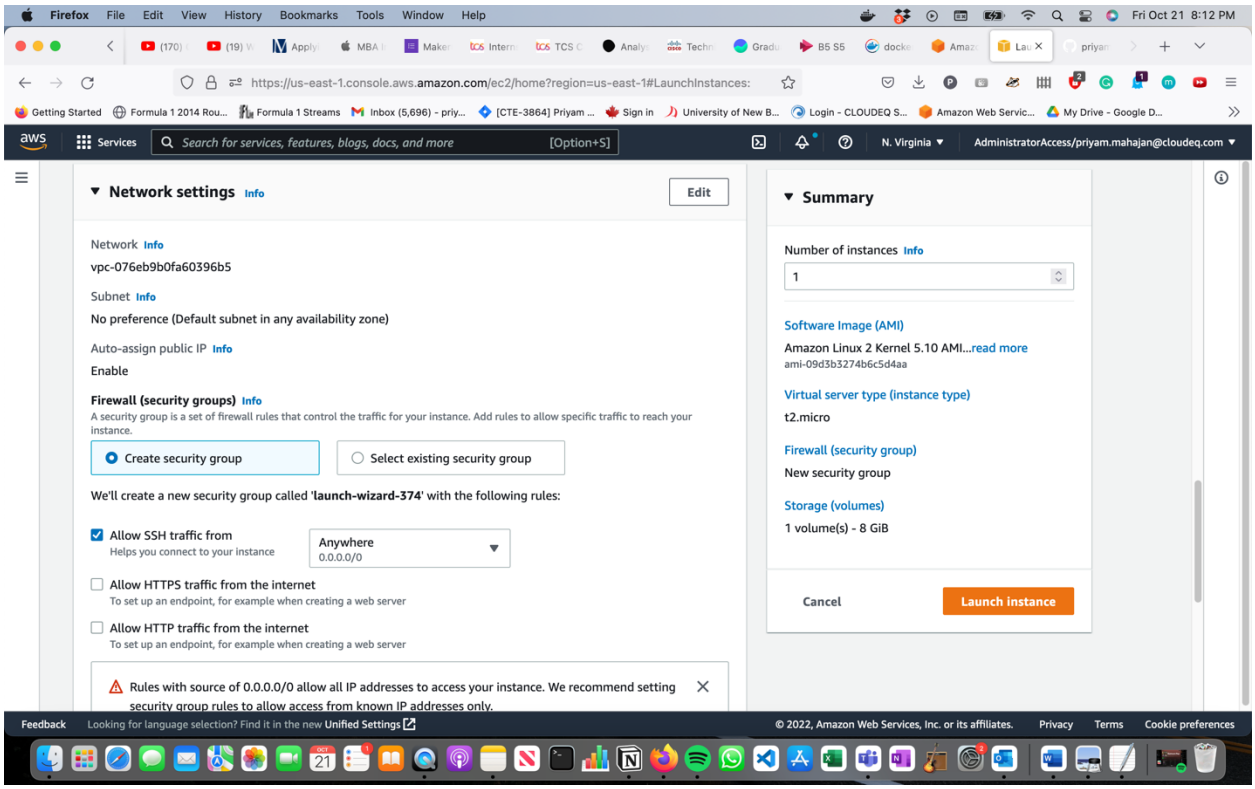
6. Choose instance type. (I have used t2.micro – free tier)



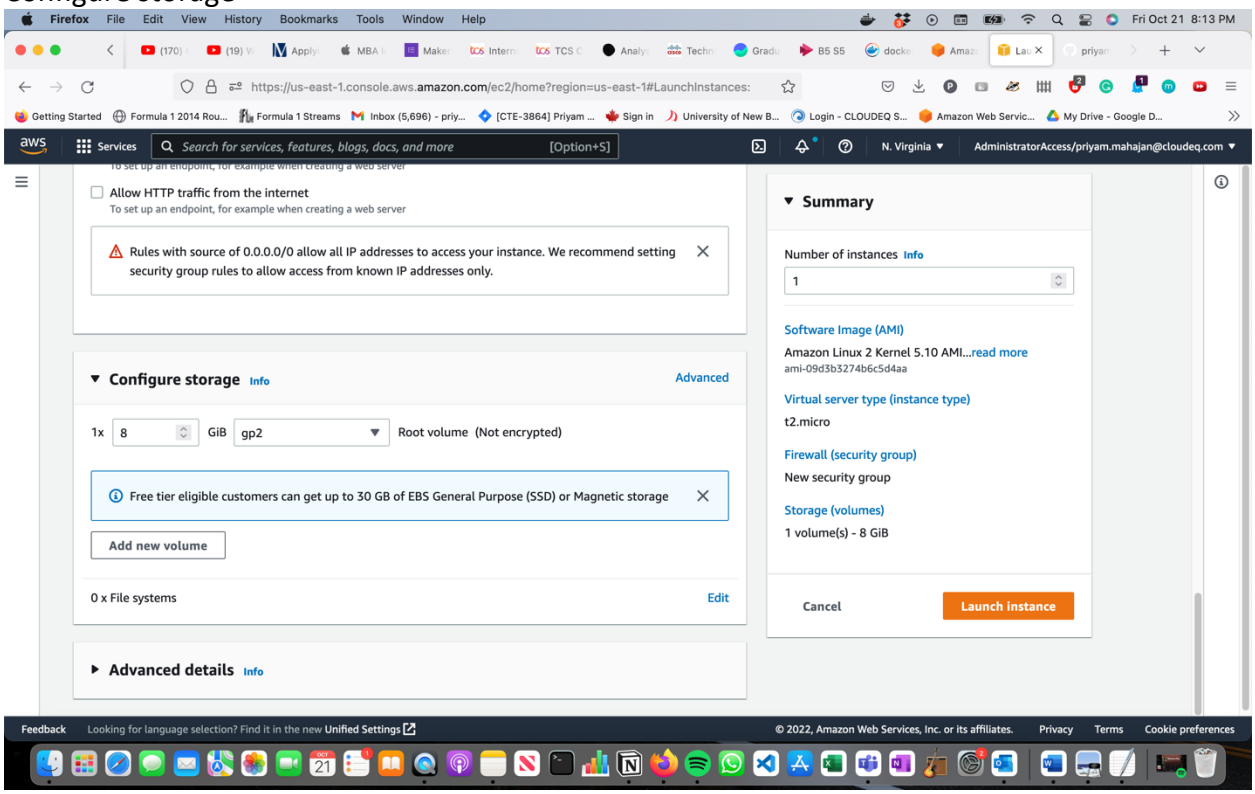
7. Create key-pair for secure login (If required) – you can choose an existing pair or create your own (recommended)



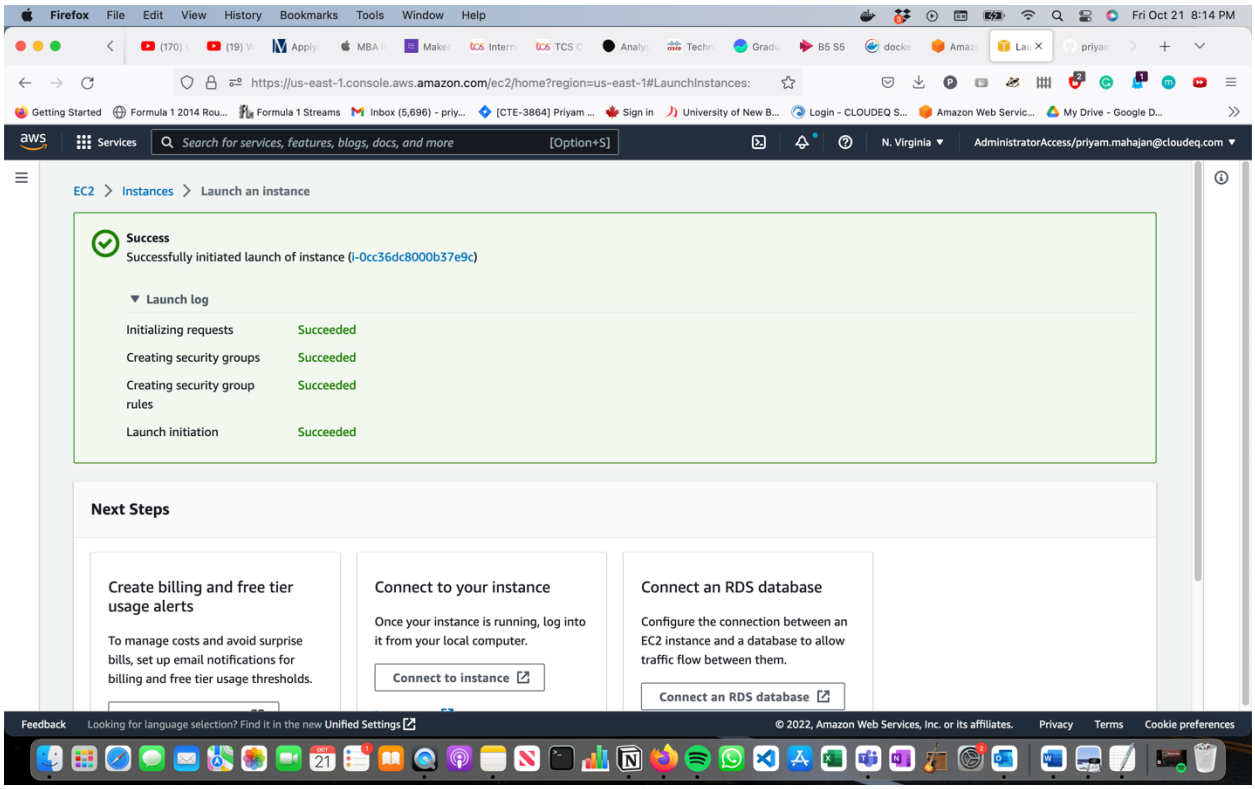
8. Create security group



9. Configure storage



10. Done!


Firefox File Edit View History Bookmarks Tools Window Help
Fri Oct 21 8:14 PM
https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LaunchInstances:
aws
Services
Search for services, features, blogs, docs, and more [Option+S]
N. Virginia
AdministratorAccess/priyam.mahajan@cloudeq.com
EC2 > Instances > Launch an instance
Success
Successfully initiated launch of instance (i-0cc36dc8000b37e9c)
Launch log
Initializing requests Succeeded
Creating security groups Succeeded
Creating security group rules Succeeded
Launch initiation Succeeded
Next Steps
Create billing and free tier usage alerts
To manage costs and avoid surprise bills, set up email notifications for billing and free tier usage thresholds.
Connect to your instance
Once your instance is running, log into it from your local computer.
Connect to instance
Connect an RDS database
Configure the connection between an EC2 instance and a database to allow traffic flow between them.
Connect an RDS database
Feedback
Looking for language selection? Find it in the new Unified Settings
© 2022, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences