



**KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY
(AN AUTONOMOUS INSTITUTION)**



Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH,
Narayanguda, Hyderabad – 500029



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

LAB RECORD

CLOUD COMPUTING LAB

B.Tech. III YEAR II SEM (KR21)

ACADEMIC YEAR 2023-24



Certificate

This is to certify that following is a Bonafide Record of the workbook task done by

_____ bearing Roll No _____ of _____

Branch of _____ year B.Tech Course in the _____

Subject during the Academic year _____ & _____ under our supervision.

Number of week tasks completed: _____

Signature of Staff Member Incharge

Signature of Head of the Dept.

Signature of Internal Examiner

Signature of External Examiner



KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY (AN AUTONOMOUS INSTITUTE)



**Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH, Hyderabad
Department of Computer Science & Engineering**

Daily Laboratory Assessment Sheet

Name of the Lab:

Name of the Student:

Class:

HT.No:

Faculty Incharge

INDEX

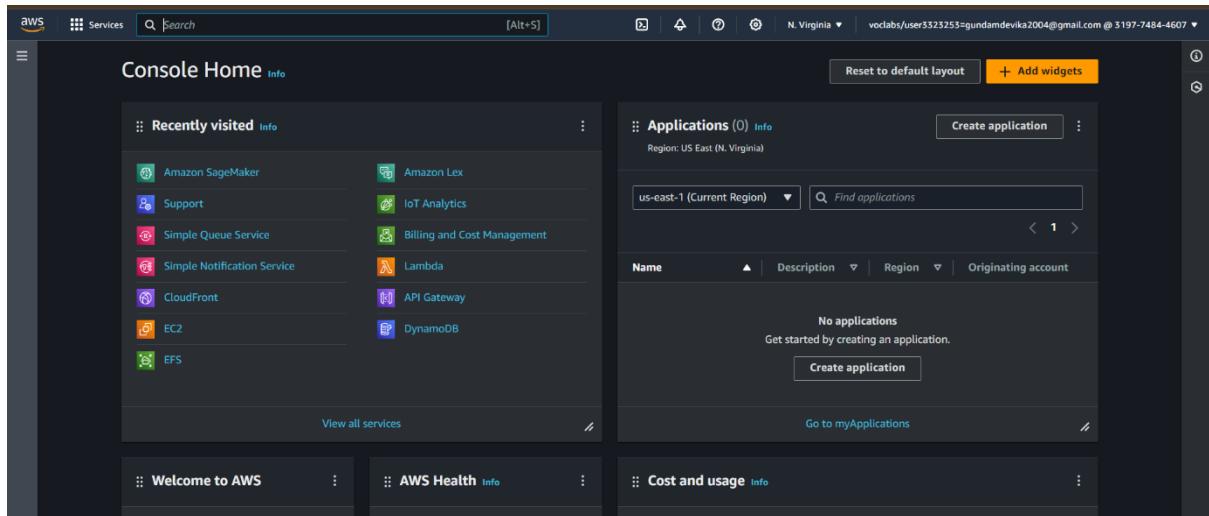
S.NO	CONTENTS	PAGE NO
Exp No	List of Experiments	
1		
2		
3		
4		
5		
6		

S.NO	CONTENTS	PAGE NO
7		
8		
9		
10		
11		

Experiment-1

Aim: Establish an AWS account. Use the AWS Management Console to launch an EC2 instance and connect to it.

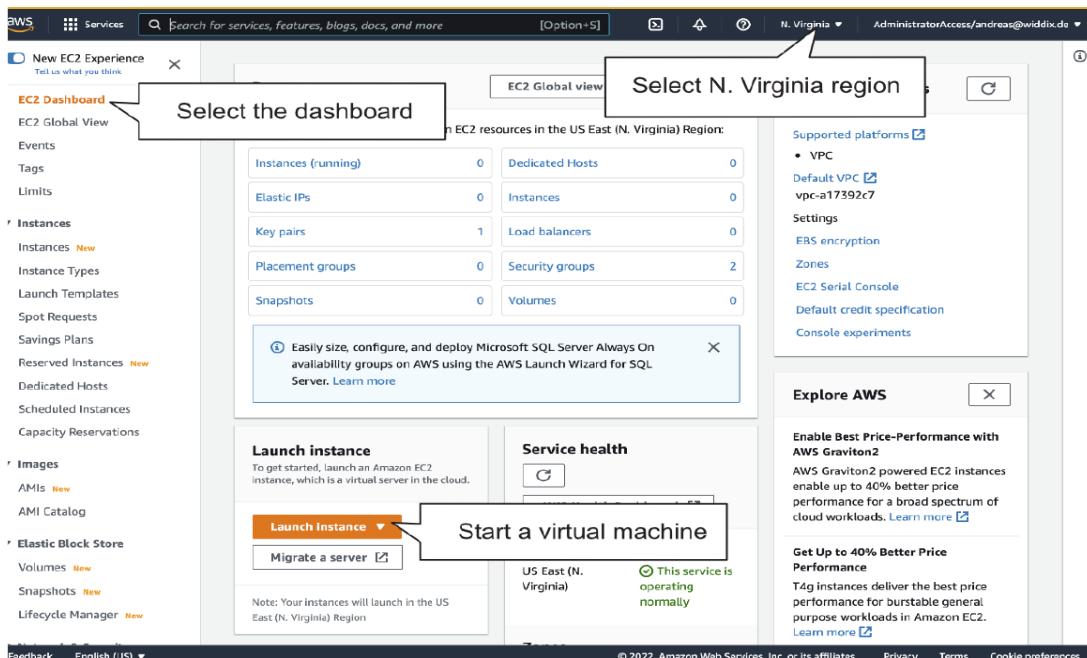
Step 1: Login to the AWS Management Console using Login Credentials



Step 2: Make sure we're in the N. Virginia (US East) region because we optimized our examples for this region.

Step 3: Click on Services and search for EC2.

Step 4: Click Launch instance to start the wizard for launching a virtual machine as shown in the figure.



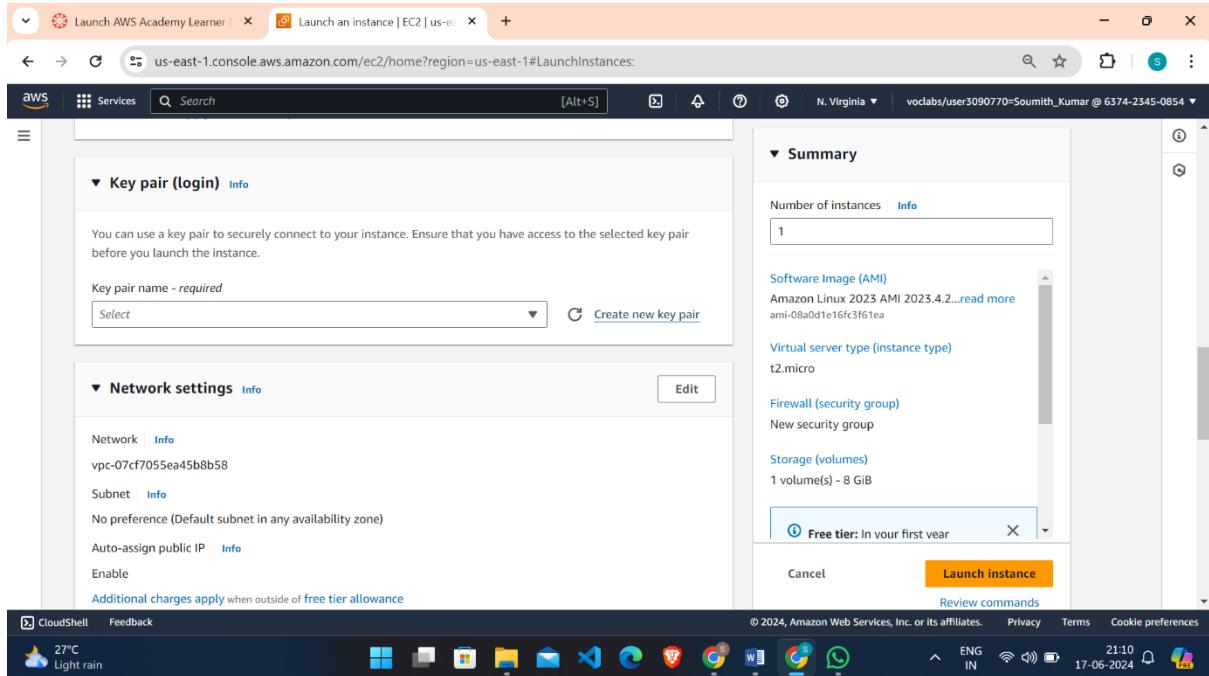
Step 5: Give a name to the instance Exp1

The screenshot shows the AWS EC2 'Launch an instance' wizard. In the 'Name and tags' section, the 'Name' field is filled with 'Exp1'. The 'Software Image (AMI)' section shows 'Amazon Linux 2023 AMI 2023.4.2...' with a note about being free tier eligible. The 'Virtual server type (instance type)' is set to 't2.micro'. The 'Launch instance' button is highlighted in orange at the bottom right.

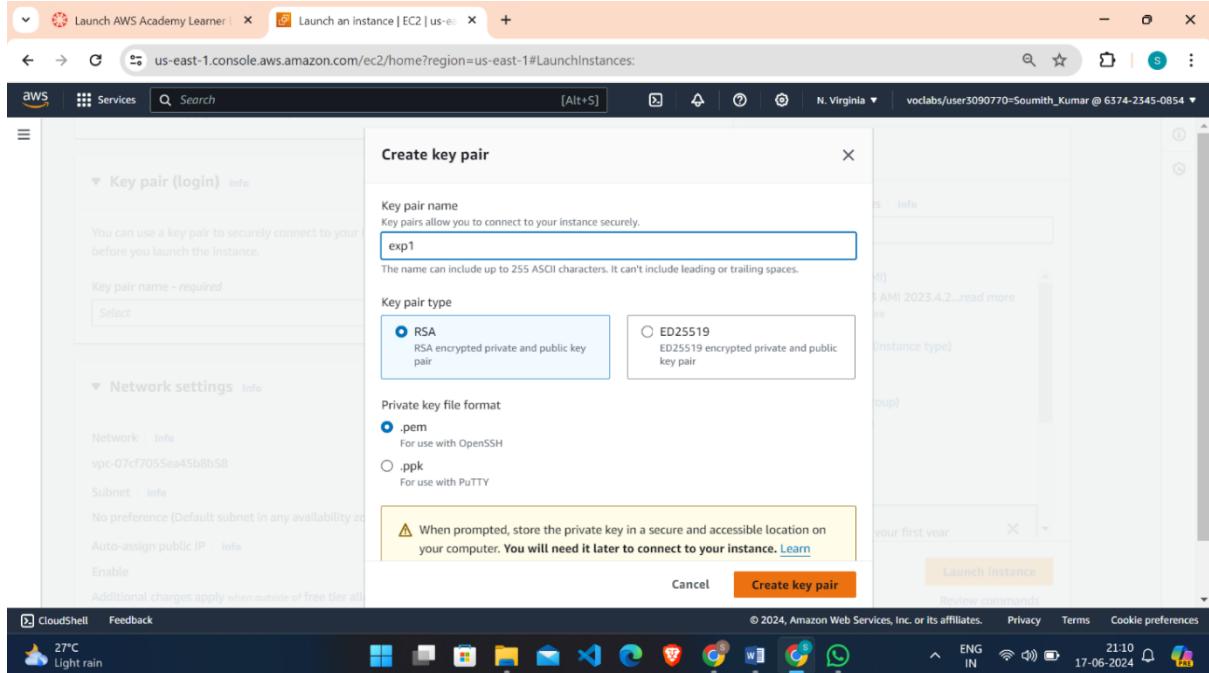
Step 6: Select the Amazon Linux as our Operating System and select instance type as t2.micro

The screenshot shows the 'Application and OS Images (Amazon Machine Image)' section. The 'Amazon Linux 2023 AMI' is selected. The 'Virtual server type (instance type)' is set to 't2.micro'. The 'Launch instance' button is highlighted in orange at the bottom right.

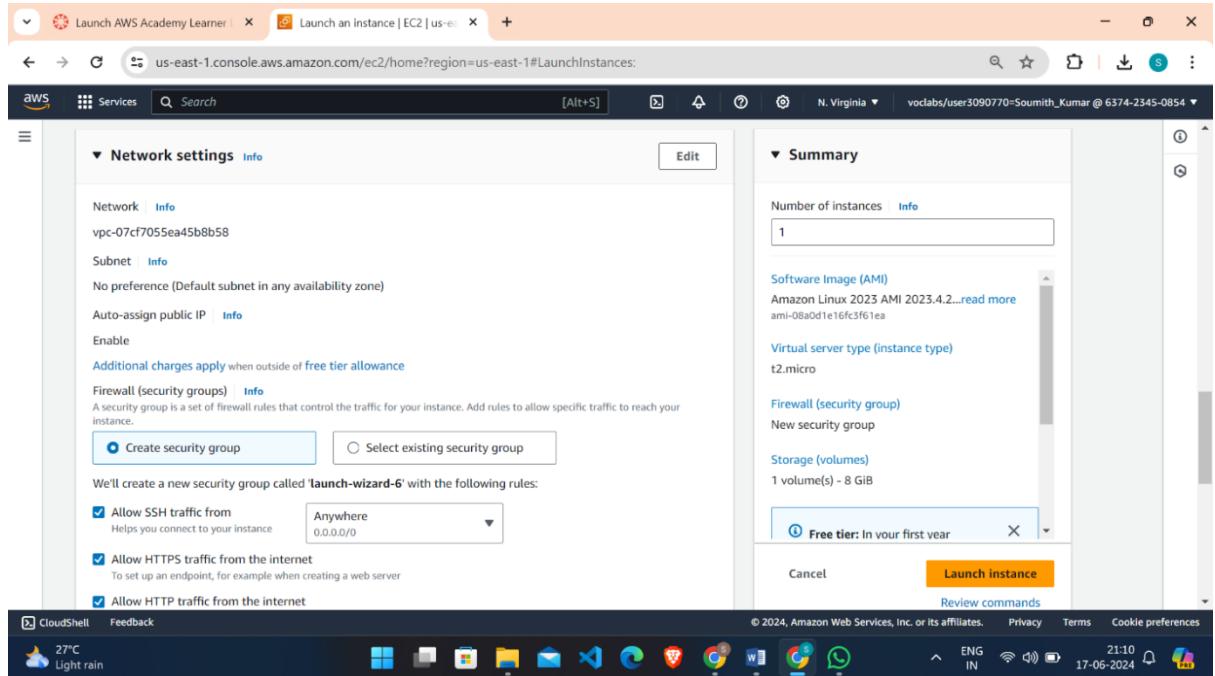
Step 7: Now create a key pair by clicking on the hyperlink



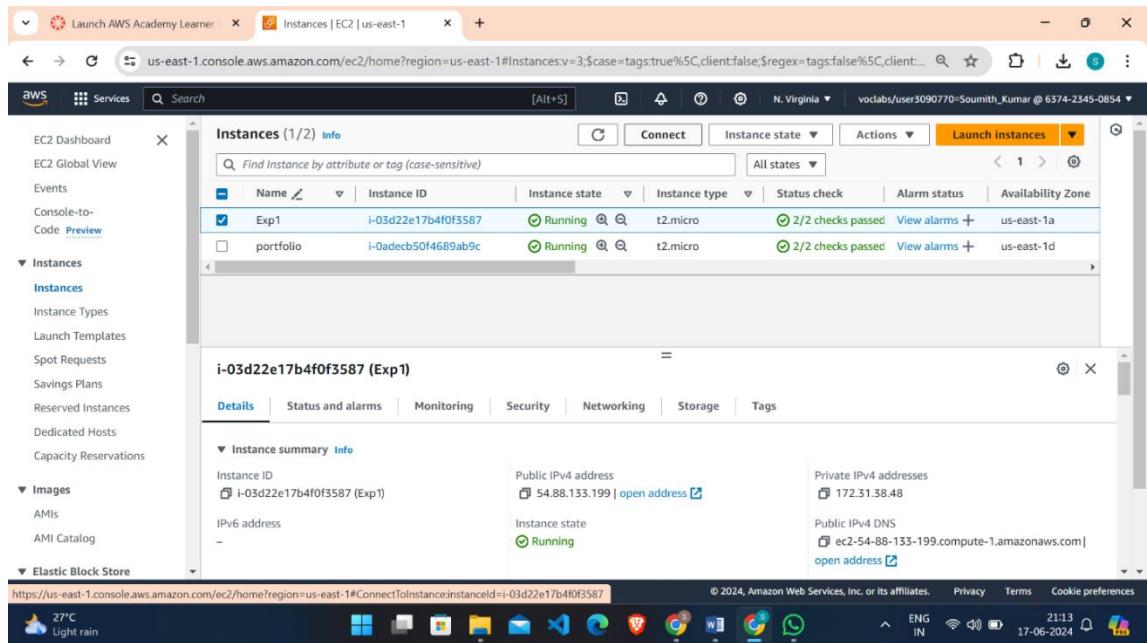
Step 8: Name the key pair exp1 and click on create key pair keeping other fields default.



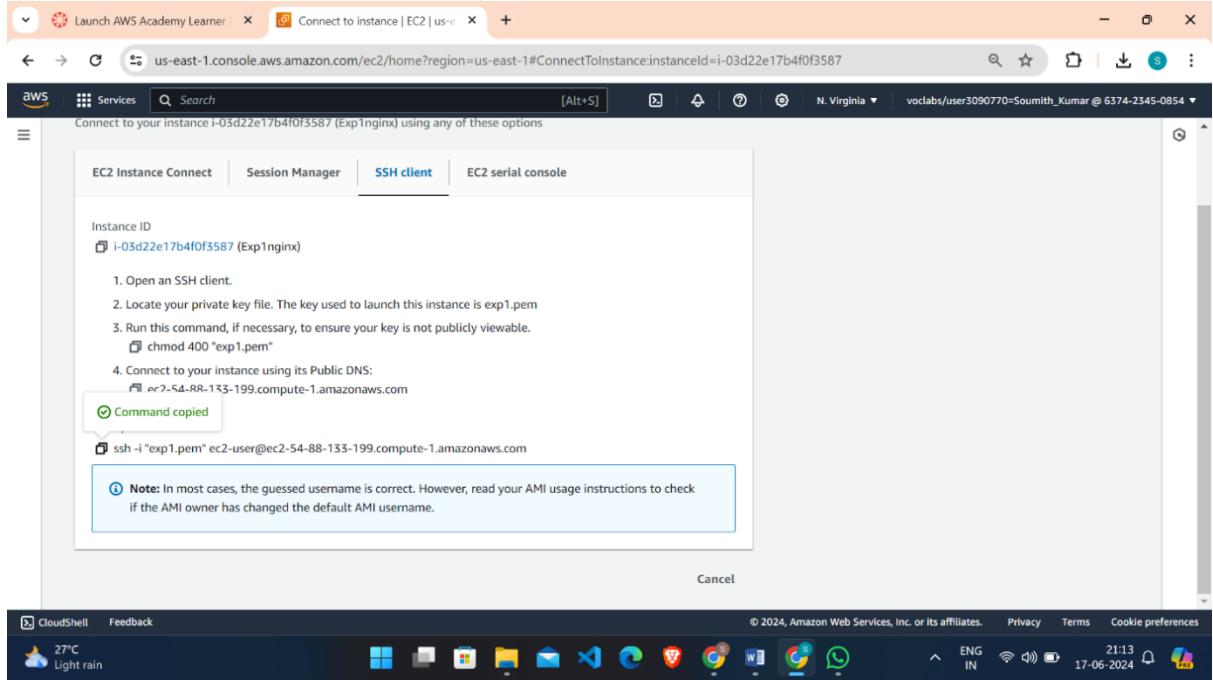
Step 9: Click on the checkboxes Allow HTTPS and HTTP traffic from internet along with SSH under Network settings. Then keeping everything else default click on Launch instance.



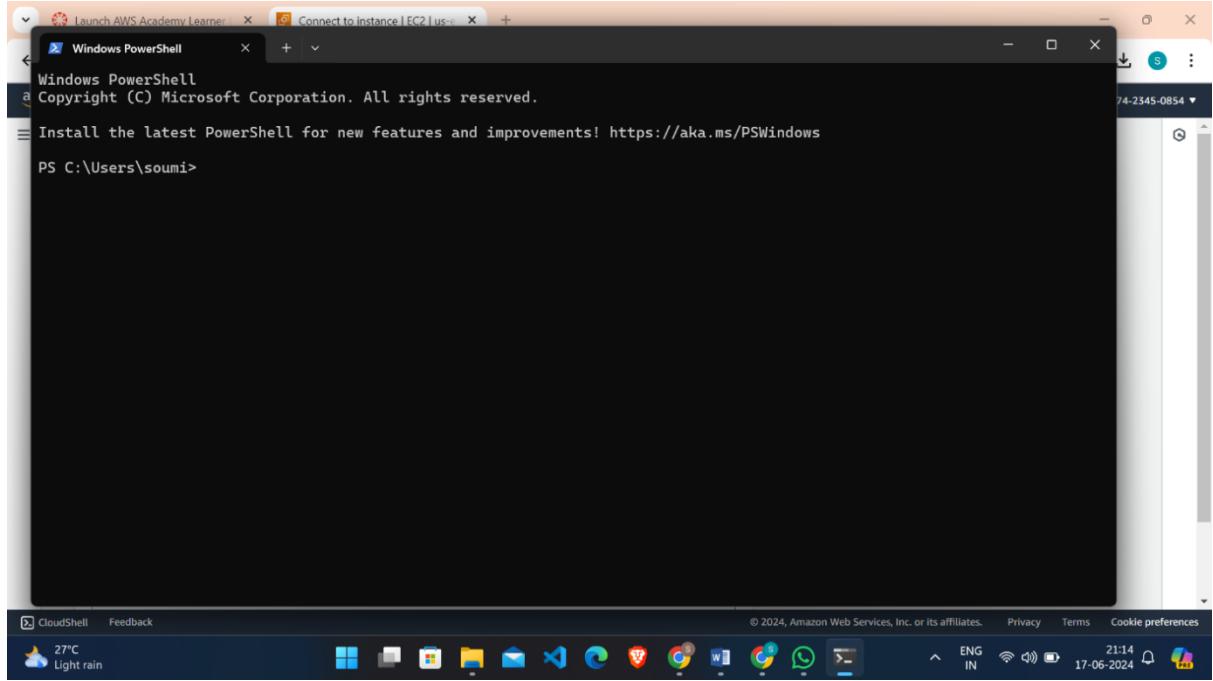
Step 10: Go to instances and select the given instance and click on connect after we see the instance state as running and 2/2 status checks



Step 11: Go to SSH client and copy the ssh key



Step 12: Now open PowerShell on desktop

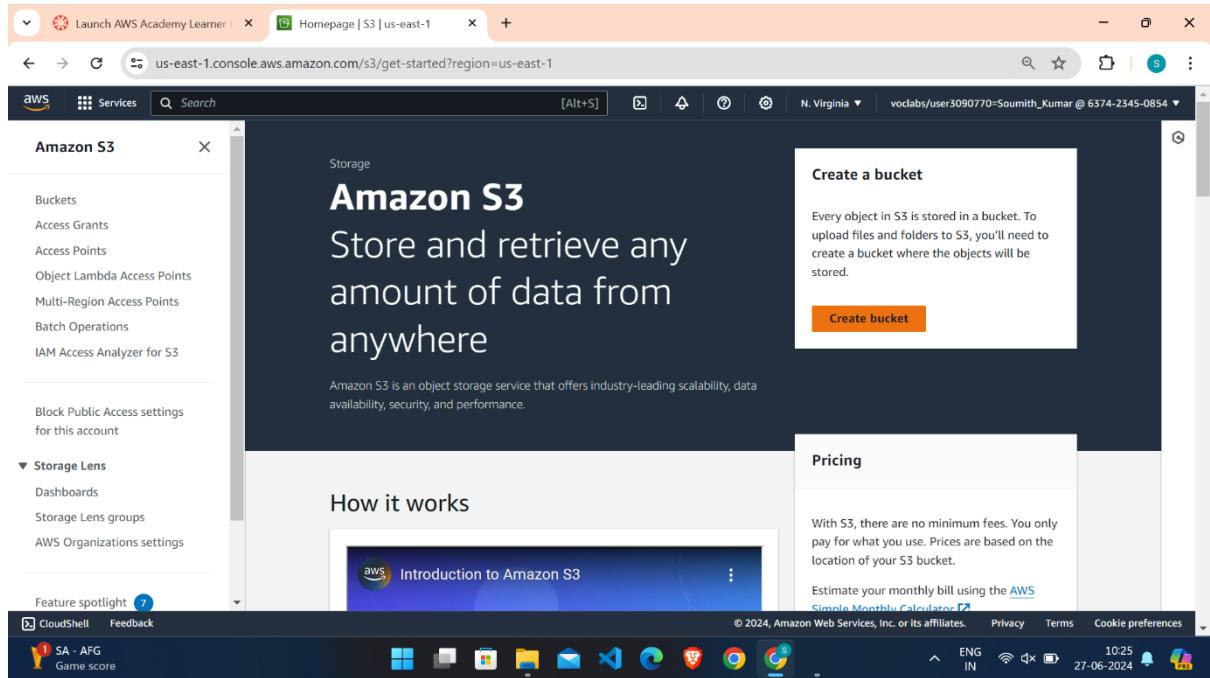


Step13: Change the directory where key.pem file is located by using cd and then paste the ssh key.

Experiment– 2

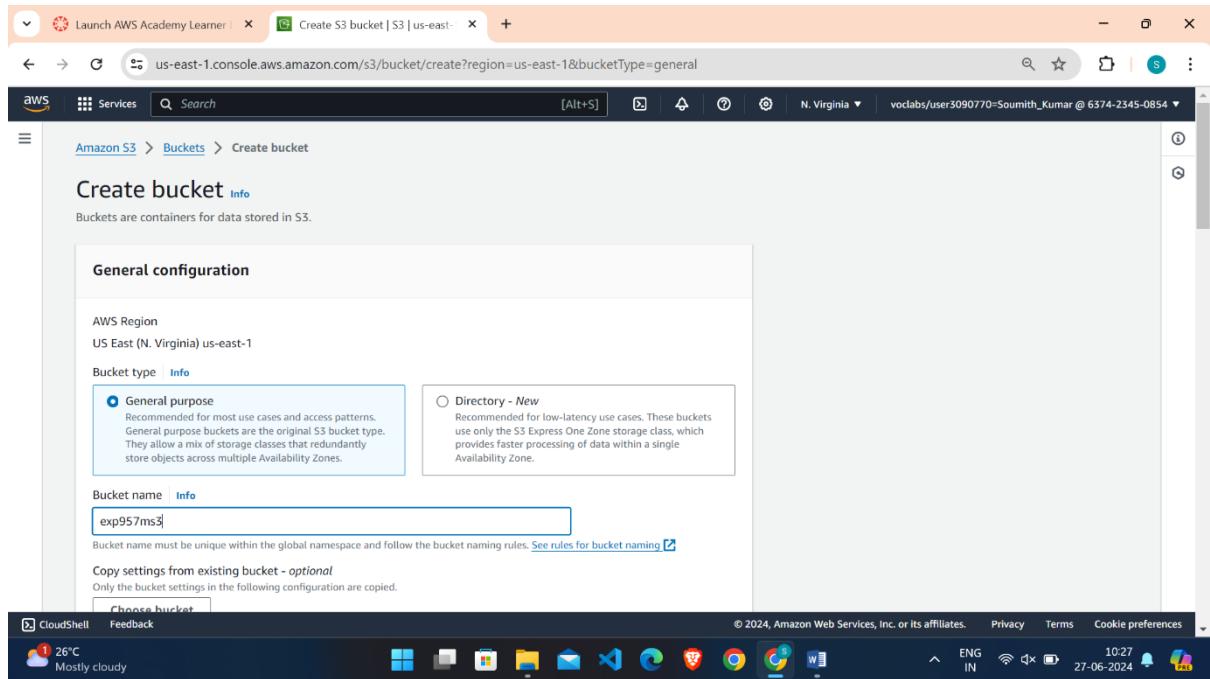
Aim : Create our First AWS S3 Bucket and Upload Content to Bucket and Manage their Access and Create Static Website using AWS S3

Step 1: Navigate to AWS console and search for S3 and click on create bucket.



Step 2: Enter the following details

- Bucket type – General purpose
- Bucket name – must be unique within global namespace.



Step 3: Select Object Ownership as ACLs enabled

The screenshot shows the 'Object Ownership' step of the 'Create S3 bucket' wizard. It includes sections for 'Object Ownership Info', 'Object Ownership', and 'Bucket owner preferred' settings. A note at the bottom recommends disabling ACLs unless needed for individual object access or sharing.

Object Ownership Info
Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

ACLs disabled (recommended)
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

ACLs enabled
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

⚠️ We recommend disabling ACLs, unless you need to control access for each object individually or to have the object writer own the data they upload. Using a bucket policy instead of ACLs to share data with users outside of your account simplifies permissions management and auditing.

Object Ownership
 Bucket owner preferred
If new objects written to this bucket specify the bucket-owner-full-control canned ACL, they are owned by the bucket owner. Otherwise, they are owned by the object writer.

Object writer
The object writer remains the object owner.

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 10:28 27-06-2024

- Allow all public access

The screenshot shows the 'Block all public access' step of the 'Create S3 bucket' wizard. It lists several options under 'Block all public access' and includes a note about turning off the setting.

Block all public access
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

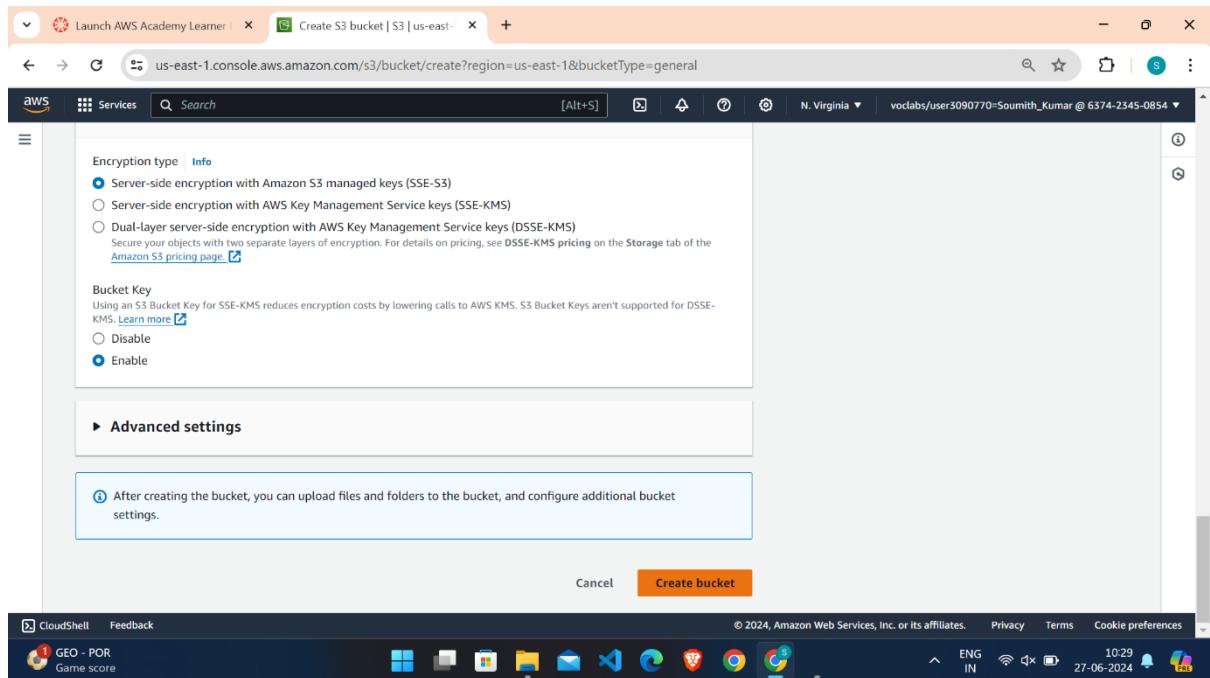
- Block public access to buckets and objects granted through new access control lists (ACLs)**
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
- Block public access to buckets and objects granted through any access control lists (ACLs)**
S3 will ignore all ACLs that grant public access to buckets and objects.
- Block public access to buckets and objects granted through new public bucket or access point policies**
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
- Block public and cross-account access to buckets and objects through any public bucket or access point policies**
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

⚠️ Turning off block all public access might result in this bucket and the objects within becoming public
AWS recommends that you turn on block all public access, unless public access is required for specific and verified use cases such as static website hosting.

I acknowledge that the current settings might result in this bucket and the objects within becoming public

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 10:28 27-06-2024

- Click on Create Bucket



Step 4: Click on bucket

Name	AWS Region	IAM Access Analyzer	Creation date
57m-s3-trigger	US East (N. Virginia) us-east-1	View analyzer for us-east-1	June 26, 2024, 12:19:19 (UTC+05:30)
awsbucketexp957m	US East (N. Virginia) us-east-1	View analyzer for us-east-1	June 24, 2024, 00:08:46 (UTC+05:30)
awsbucketexp957m2	US West (Oregon) us-west-2	View analyzer for us-west-2	June 24, 2024, 00:10:29 (UTC+05:30)
cloudfrontbucket57m	US East (N. Virginia) us-east-1	View analyzer for us-east-1	June 18, 2024, 12:32:27 (UTC+05:30)
exp957ms3	US East (N. Virginia) us-east-1	View analyzer for us-east-1	June 27, 2024, 10:29:10 (UTC+05:30)
sns---s3	US East (N. Virginia) us-east-1	View analyzer for us-east-1	June 18, 2024, 12:21:27 (UTC+05:30)
sqs-bucket21	US East (N. Virginia) us-east-1	View analyzer for us-east-1	June 22, 2024, 10:38:19 (UTC+05:30)
version-model-sou	US East (N. Virginia) us-east-1	View analyzer for us-east-1	June 23, 2024, 23:45:10 (UTC+05:30)

- Click on upload and select file to upload and click on upload.

Screenshot of the AWS S3 console showing an empty bucket named "exp957ms3".

The browser address bar shows: us-east-1.console.aws.amazon.com/s3/buckets/exp957ms3?region=us-east-1&bucketType=general&tab=objects

The AWS S3 navigation bar includes: Services, Search, [Alt+S], N. Virginia, vclabs/user3090770=Soumith_Kumar @ 6374-2345-0854.

The main content area shows the "Objects (0) info" section with a search bar and a table header: Name, Type, Last modified, Size, Storage class. A message states: "No objects" and "You don't have any objects in this bucket." An "Upload" button is present.

The taskbar at the bottom shows various application icons and the date/time: 27-06-2024, 10:30.

Screenshot of the AWS S3 console showing the upload process for the "exp957ms3" bucket.

The browser address bar shows: us-east-1.console.aws.amazon.com/s3/upload/exp957ms3?region=us-east-1&bucketType=general

The AWS S3 navigation bar includes: Services, Search, [Alt+S], N. Virginia, vclabs/user3090770=Soumith_Kumar @ 6374-2345-0854.

The main content area shows the "Upload" section with instructions: "Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API." A "Learn more" link is provided. A large dashed box allows dragging and dropping files. Below it, a table lists "Files and folders (1 Total, 34.3 KB)": hospitalfrontpagefewd2.html (text/html). Buttons for "Remove", "Add files", and "Add folder" are available. A "Destination" section is also present.

The taskbar at the bottom shows various application icons and the date/time: 27-06-2024, 10:31.

Step 5: Select the file and go to Permissions tab.

The screenshot shows the AWS S3 console with the file `hospitalfrontpagefewd2.html` selected. The **Permissions** tab is active. The Access Control List (ACL) table shows the following grants:

Grantee	Object	Object ACL
Object owner (your AWS account) Canonical ID: 25fa02414f6c5561ef77623c581b8edc6eb547470e02a8770c1716d33ba62900	Read	Read, Write
Everyone (public access) Group: http://acs.amazonaws.com/groups/global/AllUsers	-	-
Authenticated users group (anyone with an AWS account) Group: http://acs.amazonaws.com/groups/global/AuthenticatedUsers	-	-

Step 6: Click on edit and allow all the Object and Object ACL permissions and click on save changes

The screenshot shows the **Edit access control list (ACL)** page for the file `hospitalfrontpagefewd2.html`. The **Access control list (ACL)** table has been modified to grant full permissions to all entities:

Grantee	Objects	Object ACL
Object owner (your AWS account) Canonical ID: 25fa02414f6c5561ef77623c581b8edc6eb547470e02a8770c1716d33ba62900	<input checked="" type="checkbox"/> Read <input checked="" type="checkbox"/> Write	<input checked="" type="checkbox"/> Read <input checked="" type="checkbox"/> Write
Everyone (public access) Group: http://acs.amazonaws.com/groups/global/AllUsers	<input checked="" type="checkbox"/> Read <input type="checkbox"/> Write	<input checked="" type="checkbox"/> Read <input type="checkbox"/> Write
Authenticated users group (anyone with an AWS account) Group: http://acs.amazonaws.com/groups/global/AuthenticatedUsers	<input checked="" type="checkbox"/> Read <input type="checkbox"/> Write	<input checked="" type="checkbox"/> Read <input type="checkbox"/> Write

Step 7: Select the bucket and go to properties tab.

The screenshot shows the AWS S3 Bucket Properties page for 'exp957ms3'. The top navigation bar includes tabs for Objects, Properties (which is selected), Permissions, Metrics, Management, and Access Points. The main content area is titled 'Bucket overview' and displays basic information: AWS Region (US East (N. Virginia) us-east-1), Amazon Resource Name (ARN) (arn:aws:s3:::exp957ms3), and Creation date (June 27, 2024, 10:29:10 (UTC+05:30)). Below this, the 'Bucket Versioning' section is shown, indicating it is disabled. A note states: 'Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures.' An 'Edit' button is present in this section. At the bottom of the page, there are links for CloudShell, Feedback, and various system status indicators like weather and time.

Step 8: Scroll down for Static web hosting and click on edit. Enable the static web hosting and enter the index document name as “index.html”

The screenshot shows the 'Edit static website hosting' page for the 'exp957ms3' bucket. Under the 'Static website hosting' section, the 'Enable' option is selected. In the 'Hosting type' section, the 'Host a static website' option is selected. A note explains that for public readability, S3 Block Public Access settings must be edited. In the 'Index document' section, the file 'index.html' is specified as the home page. The bottom of the page features standard AWS navigation links for CloudShell, Feedback, and system status.

Step 9: We can find a link of the static website copy it.

The screenshot shows the AWS S3 console for a bucket named 'exp957ms3'. Under the 'Static website hosting' section, 'Hosting type' is set to 'Bucket hosting'. A message indicates that the bucket has been configured as a static website, with the endpoint available at [amazonaws.com](https://exp957ms3.s3-website-us-east-1.amazonaws.com). The status of 'Bucket website endpoint copied' is shown as green.

Step 10: Paste it in new tab and we can access our file.

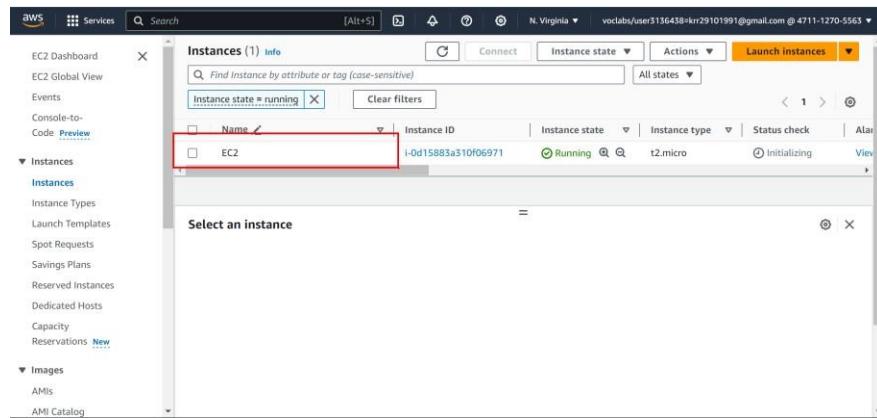
The screenshot shows a web browser window with a blue header bar. The main content area displays a newsletter sign-up form titled 'Subscribe to Our Newsletter'. It includes fields for 'Enter your email' and 'Enter your name (optional)', and a 'Send' button. The background features abstract blue and white circular patterns. The browser's address bar shows the URL exp957ms3.s3-website-us-east-1.amazonaws.com.

Experiment-3

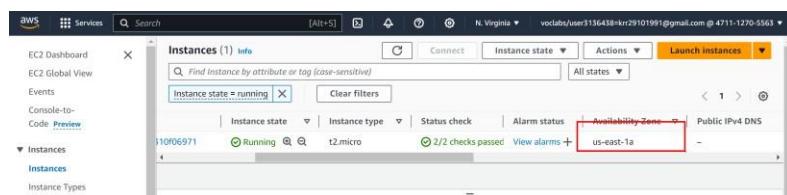
Aim: Exemplify the principle of rapid elasticity through a practical exercise involving the setup of an EC2 Amazon instance and the creation of multiple Elastic Block Store (EBS) volumes to be attached to EC2 instance.

→ Attach a new Volume to EC2

Step 1: Launch an EC2 instance

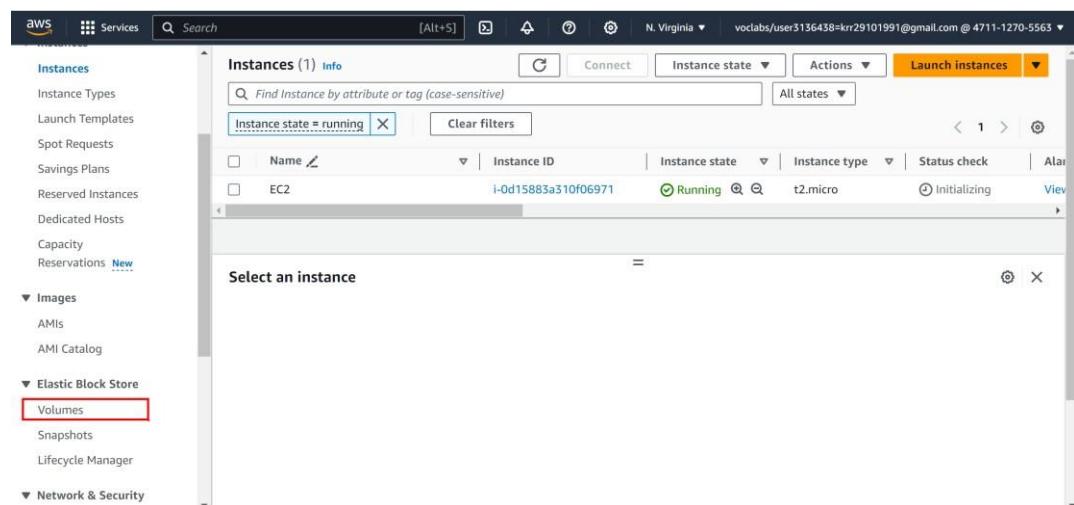


Step 2: Check the Availability Zone === In this case its us-east-1a



- Create a volume

Step 3: Click on volume From Elastic Block Store



Step 4: Click on create Volume

The screenshot shows the AWS Management Console with the EBS service selected. In the left navigation pane, 'Volumes' is under 'Elastic Block Store'. The main area displays a table of existing volumes with columns for Name, Volume ID, Type, Size, IOPS, Throughput, and Snapshot. A yellow box highlights the 'Create volume' button in the top right corner of the table header.

Step 5 :Change the size to required capacity == In this case I am using 15 GB

The screenshot shows the 'Volume settings' page. Under 'Volume type', 'General Purpose SSD (gp3)' is selected. In the 'Size (GiB)' field, the value '15' is entered and highlighted with a red box. A note below states: 'Min: 1 GiB, Max: 16384 GiB. The value must be an integer.'

Step 6: Check the Availability Zone == In this case it should be us-east-1a (as EC2 is launched in that AZ)

The screenshot shows the 'Volume settings' page. Under 'Availability Zone', 'us-east-1a' is selected and highlighted with a red box. Other fields shown include 'IOPS' (3000), 'Throughput (MiB/s)' (125), and 'Encryption' options.

Step 7: Click on Create Volume

The screenshot shows a confirmation dialog box. At the bottom right, the 'Create volume' button is highlighted with a yellow box. Other visible text includes 'Tags - optional' and 'Snapshot summary'.

Successfully created volume vol-0e5729116f510e661.

Volumes (4) Info

Name	Volume ID	Type	Size	IOPS	Throughput	Snapshot
-	vol-089d4e0f2ca721fc6	gp2	30 GiB	100	-	snap-053648e...
-	vol-0be3ef4a55f2eb2bf	gp2	30 GiB	100	-	snap-053648e...
-	vol-0891d7681ad96866c	gp3	8 GiB	3000	125	snap-05df0f2...
-	vol-0e5729116f510e661	gp3	15 GiB	3000	125	-

Name the Volume == to easily recognize == IN this case I will name as Extra Volume

Step 10: Now attach the volume to the EC2 Instance

- I. Select the Extra Volume
- II. Drop down the Actions
- III. Click on Attach Volume

Volumes (1/4) Info

Name	Volume ID	Type	Size	IOPS
-	vol-089d4e0f2ca721fc6	gp2	30 GiB	100
-	vol-0be3ef4a55f2eb2bf	gp2	30 GiB	100
-	vol-0891d7681ad96866c	gp3	8 GiB	3000
<input checked="" type="checkbox"/> Extra Volume	vol-0e5729116f510e661	gp3	15 GiB	3000

Actions

- Modify volume
- Create snapshot
- Create snapshot lifecycle policy
- Delete volume
- Attach volume**
- Detach volume
- Force detach volume
- Manage auto-enabled I/O
- Manage tags
- Fault injection

- IV. Select your ec2 instance
- V. Name the Device Name as /dev/sdf
- VI. Click on Attach Volume

Volume ID
vol-0e5729116f510e661 (Extra Volume)

Availability Zone
us-east-1a

Instance **Info**
i-0d15883a310f06971

Device name **Info**
/dev/sdf

Newer Linux kernels may rename your devices to **/dev/xvdf** through **/dev/xvdः** internally, even when the device name entered here (and shown in the details) is **/dev/sdf** through **/dev/sdp**.

Attach volume

Step 11: Now Connect to the EC2 Instance named “EC2”

- Select the instance
- Click on connect

Step 12: Now run few commands to verify the disks attached to the Ec2 Instance

- Change over to root user ===== sudo su
- Check the disks ===== fdisk -l

```
[ec2-user@ip-172-31-1-51 ~]$ sudo su
[root@ip-172-31-1-51 ec2-user]# fdisk -l

Disk /dev/xvdf: 15 GiB, 16106127360 bytes, 31457280 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

- Create a directory ===== mkdir /mnt/test
- Now format the newly attached disk ===== mkfs /dev/xvdf

```

Disk /dev/xvdf: 15 GiB, 16106127360 bytes, 31457280 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
[root@ip-172-31-1-51 ec2-user]# mkdir /mnt/test
[root@ip-172-31-1-51 ec2-user]# mkfs /dev/xvdf
mke2fs 1.46.5 (30-Dec-2021)
Creating filesystem with 3932160 4k blocks and 983040 inodes
Filesystem UUID: 6cacedf0-bc25-4702-b0ef-2255fa4ee337
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208

```

- e) Now mount the disk on the directory

```
[root@ip-172-31-1-51 ec2-user]# mount /dev/xvdf /mnt/test
```

- f) Now change the directory and create a file

```

[root@ip-172-31-1-51 ec2-user]# cd /mnt/test
[root@ip-172-31-1-51 test]# nano input.txt
[root@ip-172-31-1-51 test]# cat input.txt
hi hello
[root@ip-172-31-1-51 test]#

```

→Detach the new volume and Attach to another EC2 Instance

Step 1: Click on volumes

The screenshot shows the AWS Cloud9 Instances page. On the left sidebar, under the 'Elastic Block Store' section, the 'Volumes' option is highlighted with a red box. In the main pane, the 'Instances (1)' table lists one instance named 'EC2' with the ID 'i-0d15883a310f06971'. The instance state is 'Running'. Below the table, a modal window titled 'Select an instance' is open, showing the same 'EC2' entry.

Name	Instance ID	Instance state	Instance type	Status check
EC2	i-0d15883a310f06971	Running	t2.micro	2/2 checks passed

Step 2: Select the volume then drop down the Actions and click on detach volume then click on detach and volume state needs to change from in-use to available

Volumes (1/2) Info

Name	Volume ID	Type	Size	IOPS
-	vol-0891d7681ad96866c	gp3	8 GiB	3000
<input checked="" type="checkbox"/> Extra Volume	vol-0e5729116f510e661	gp3	15 GiB	3000

Actions ▾ Create volume

- Modify volume
- Create snapshot
- Create snapshot lifecycle policy
- Delete volume
- Attach volume
- Detach volume**
- Force detach volume
- Manage auto-enabled I/O
- Manage tags
- Fault injection

Volumes (2) Info

Throughput	Snapshot	Created	Availability Zone	Volume state	Alarm status
125	snap-05df0f2...	2024/04/18 16:10 GMT+5...	us-east-1a	In-use	No alarms
125	-	2024/04/18 16:21 GMT+5...	us-east-1a	Available	No alarms

Step 3: Now go to EC2 dashboard and Launch another new instance named as EC2-New

Instances (2) Info

Name	Instance ID	Instance state	Instance type	Status check
<input type="checkbox"/> EC2-New	i-000e13dfc48d5fff	Running	t2.micro	-
<input type="checkbox"/> EC2	i-0d15883a310f06971	Running	t2.micro	2/2 checks passed

Step 4: Now click again on Volumes

Instances (2) Info

Name	Instance ID	Instance state	Instance type	Status check
<input type="checkbox"/> EC2-New	i-000e13dfc48d5fff	Running	t2.micro	-
<input type="checkbox"/> EC2	i-0d15883a310f06971	Running	t2.micro	2/2 checks passed

Step 5: Select the Extra Volume then drop down the Actions and click on Attach

Name	Volume ID	Type	Size	IOPS
vol-0891d7681ad96866c	gp3	8 GiB	3000	
Extra Volume	vol-0e5729116f510e661	gp3	15 GiB	3000

Actions ▾

- Modify volume
- Create snapshot
- Create snapshot lifecycle policy
- Delete volume
- Attach volume**
- Detach volume
- Force detach volume
- Manage auto-enabled I/O
- Manage tags
- Fault injection

Step 6: Drop down and Select the EC2-New then search for Device name /dev/sdf and select the diskThen click on attach volume

Basic details

Volume ID: vol-0e5729116f510e661 (Extra Volume)

Availability Zone: us-east-1a

Instance Info:

Search instance ID or name tag: i-000e13dfc48d5ffff (EC2-New) (running)

Instance Info:

Device: /dev/sdf

Recommended for data volumes: /dev/sdf

Notes: Newer Linux kernels may rename your devices to /dev/xvdf through /dev/xvdp internally, even when the device name entered here (and shown in the details) is /dev/sdf through /dev/sdp.

Attach volume

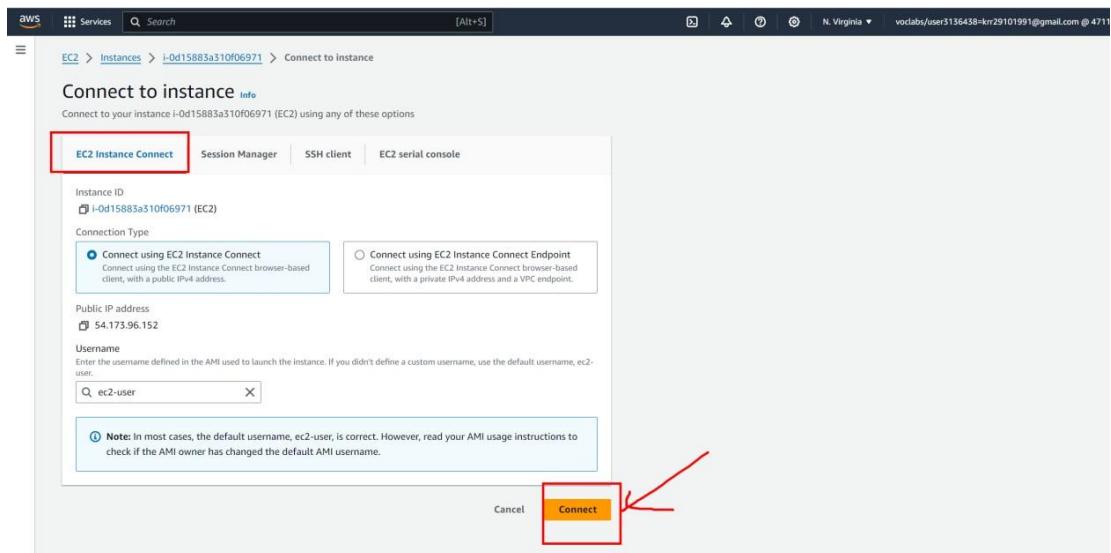
Step 7: Select the EC2-New Instance then Click on Connect

Name	Instance ID	Instance state	Instance type	Status check
EC2-New	i-000e13dfc48d5ffff	Running	t2.micro	-
EC2	i-0d15883a310f06971	Running	t2.micro	2/2 checks passed

Connect

Instance: i-000e13dfc48d5ffff (EC2-New)

Details | Status and alarm view | Monitoring | Security | Networking | Storage | Tape



Step 8: Now change user to root then create a directory and verify the disk

```

aws Services Search [Alt+S] N. Virginia v vocabs/user3136438=krr29101991@gmail.com @ 4711-1270-5563 ▾
~/m/'[ec2-user@ip-172-31-1-227 ~]$ sudo su
[root@ip-172-31-1-227 ec2-user]# mkdir /mnt/sample
[root@ip-172-31-1-227 ec2-user]# fdisk -l
Disk /dev/xvda: 8 GiB, 8589934592 bytes, 16777216 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 174813BE-7DC8-40F8-A11F-AC3DAA194142

Device      Start    End  Sectors Size Type
/dev/xvda1   24576 16777182 16752607   8G Linux filesystem
/dev/xvda127 22528   24575     2048   1M BIOS boot
/dev/xvda128 2048    22527    20480  10M EFI System

Partition table entries are not in disk order.

Disk /dev/xvdf: 15 GiB, 16106127360 bytes, 31457280 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
[root@ip-172-31-1-227 ec2-user]# [REDACTED]

aws Services Search [Alt+S] N. Virginia v vocabs/user3136438=krr29101991@gmail.com @ 4711-1270-5563 ▾
[root@ip-172-31-1-227 ec2-user]# cd /mnt/sample/
[root@ip-172-31-1-227 sample]# ls
input.txt  lost+found
[root@ip-172-31-1-227 sample]# [REDACTED]

```

→Create a snapshot and copy to another region

Step 1: Click on Volumes

The screenshot shows the AWS EC2 Instances page. On the left sidebar, under the 'Elastic Block Store' section, the 'Volumes' link is highlighted with a red box. The main content area displays a table of instances with columns for Name, Instance ID, Instance state, Instance type, and Status check.

Name	Instance ID	Instance state	Instance type	Status check
EC2-New	i-000e13dfc48d5fff	Running	t2.micro	2/2 checks passed
Public Ec2	i-0bb4dcce15c13eebb	Terminated	t2.micro	-
EC2	i-0d15883a310f06971	Running	t2.micro	2/2 checks passed
Private-EC2	i-0cd5f1c53a84323fe	Terminated	t2.micro	-

Step 2: Select the Extra Volume then Drop down the Actions and Click on Create Snapshot

The screenshot shows the AWS EBS Volumes page. A volume named 'Extra Volume' is selected and highlighted with a red box. The 'Actions' dropdown menu is open, and the 'Create snapshot' option is highlighted with a blue box. Other options in the menu include Modify volume, Create volume, Create snapshot lifecycle policy, Delete volume, Attach volume, Detach volume, Force detach volume, Manage auto-enabled I/O, Manage tags, and Fault injection.

Name	Volume ID	Type	Size	IOPS
-	vol-0891d7681ad96866c	gp3	8 GiB	3000
<input checked="" type="checkbox"/> Extra Volume	vol-0e5729116f510e661	gp3	15 GiB	3000
-	vol-02f75ef7a18b7d971	gp3	8 GiB	3000

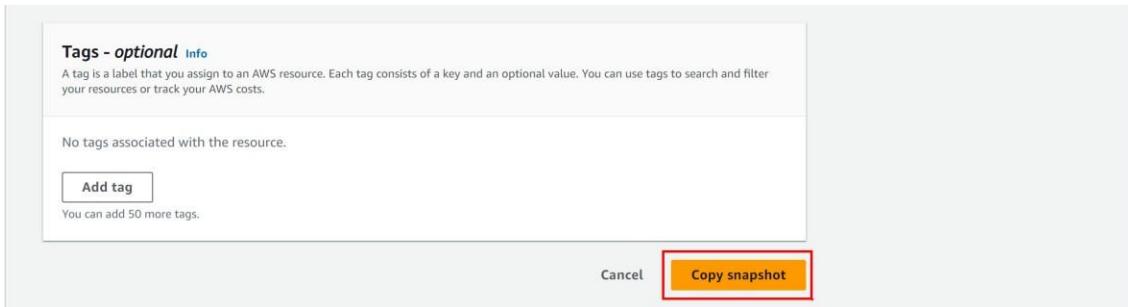
Step 3:Click on Create Snapshot

The screenshot shows the 'Create snapshot' dialog for the volume 'vol-0e5729116f510e661'. The 'Details' tab is active, showing the Volume ID 'vol-0e5729116f510e661 (Extra Volume)' and a description field containing 'SS1'. The 'Encryption info' section indicates 'Not encrypted'. The 'Tags' tab shows no tags associated with the resource. At the bottom right, the 'Create snapshot' button is highlighted with a red box.

Step 4: Click on Snapshots

Step 5: Select the Snapshot and then drop-down Actions then click on Copy Snapshot presently weare in N. Virginia Region

Step 6: Change the Destination only and only to Us-West-2 for AWS Academy users rest can whohave a free tier account can select any region then click Copy Snapshot



→Create volume from the snapshot and attach to an EC2 Instance

Step 1: Now change the Region from N. Virginia to Us-West-2 (Oregon)

Region	Snapshot ID
US East (N. Virginia)	us-east-1
US East (Ohio)	us-east-2
US West (N. California)	us-west-1
US West (Oregon)	us-west-2
Asia Pacific (Mumbai)	ap-south-1
Asia Pacific (Osaka)	ap-northeast-3
Asia Pacific (Seoul)	ap-northeast-2
Asia Pacific (Singapore)	ap-southeast-1
Asia Pacific (Sydney)	ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1
Canada (Central)	ca-central-1
Europe (Frankfurt)	eu-central-1

Name	Snapshot ID	Volume size	Description	Storage tier	Status
Snapshot1	snap-0bdb52c7dcaba1362	20 GiB	[Copied snap-08298c8cd3...]	Standard	Completed
DEMO	snap-0e84c5402d34dfcf2	2 GiB	[Copied snap-09e73b3958...]	Standard	Completed
-	snap-0eab71b049fe4f13d	15 GiB	[Copied snap-0201330dae...]	Standard	Completed

Step 2: Select the Copied Snapshot then drop down the actions and click on Create volume fromsnapshot

Step 3: Leave everything as default just click on Create Volume

Step 4: Click on Volumes

Screenshot of the AWS EBS Volumes page. The left sidebar shows 'Images' and 'Elastic Block Store' sections. The main table lists two volumes:

Name	Volume ID	Type	Size	IOPS	Throughput	Snapshot
-	vol-07125ef498042f5dc	gp3	2 GiB	3000	125	snap-0e84c54...
-	vol-0deb486c35051c2c0	gp3	15 GiB	3000	125	snap-0eab71b...

Step 5: Launch an EC2 Instance and attach the volume then connect to Instance

Screenshot of the AWS EC2 Instances page. The left sidebar shows 'Instances' section. The main table lists one instance:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status
EC2	i-0f3265b29668f1eba	Pending	t2.micro

Screenshot of the AWS EBS Volumes page. The left sidebar shows 'Volumes' section. The right panel shows the Actions menu for the selected volume 'vol-0deb486c35051c2c0':

- Actions ▲
- Create volume
- Modify volume
- Create snapshot
- Create snapshot lifecycle policy
- Delete volume
- Attach volume** (highlighted with a red box)
- Detach volume
- Force detach volume
- Manage auto-enabled I/O
- Manage tags
- Fault injection

Screenshot of the 'Attach Volume' dialog box. The 'Instance' dropdown is set to 'i-0f3265b29668f1eba'. The 'Device name' dropdown is set to '/dev/sdf'. The 'Attach volume' button is highlighted with a red box.

Volume ID: vol-0deb486c35051c2c0
Availability Zone: us-west-2a
Instance: i-0f3265b29668f1eba
Device name: /dev/sdf
Newer Linux kernels may rename your devices to /dev/xvdf through /dev/xvdp internally, even when the device name entered here (and shown in the details) is /dev/sdf through /dev/sdp.

Instances (1/1) Info

EC2 Dashboard Services Search [Alt+S] Oregon voleabs/user3136438=krr29101991@gmail.com @ 4711-1270-5563

Instances

Name: EC2

Instance ID: i-0f3265b29668f1eba

Instance state: Running

Instance type: t2.micro

Status check: -

View alarms +

EC2 Instance Connect Session Manager SSH client EC2 serial console

Instance ID: i-0f3265b29668f1eba (EC2)

Connection Type:

- Connect using EC2 Instance Connect
Connect using the EC2 Instance Connect browser-based client, with a public IPv4 address.
- Connect using EC2 Instance Connect Endpoint
Connect using the EC2 Instance Connect browser-based client, with a private IPv4 address and a VPC endpoint.

Public IP address: 34.208.234.136

Username: ec2-user

Note: In most cases, the default username, ec2-user, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

Cancel Connect

```
[ec2-user@ip-172-31-18-232 ~]$ sudo su
[root@ip-172-31-18-232 ec2-user]# mkdir /mnt/sample123
[root@ip-172-31-18-232 ec2-user]# fdisk -l
Disk /dev/xvda: 8 GiB, 8589934592 bytes, 16777216 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 174813BE-7DC8-40F8-A11F-AC3DAA194142

Device      Start    End Sectors Size Type
/dev/xvda1   24576 16777182 16752607   8G Linux filesystem
/dev/xvda12  22528   24575    2048 1M BIOS boot
/dev/xvda128  2048   22527   20480 10M EFI System

Partition table entries are not in disk order.

Disk /dev/xvdf: 15 GiB, 16106127360 bytes, 31457280 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
[root@ip-172-31-18-232 ec2-user]# mount /dev/xvdf /mnt/sample123
[root@ip-172-31-18-232 ec2-user]#
```

i-0f3265b29668f1eba (EC2)

```
[root@ip-172-31-18-232 ec2-user]# cd /mnt/sample123
[root@ip-172-31-18-232 sample123]# ls
input.txt lost+found
[root@ip-172-31-18-232 sample123]# cat input.txt
hi hello
[root@ip-172-31-18-232 sample123]#
```

i-0f3265b29668f1eba (EC2)

Public IPs: 34.208.234.136 Private IPs: 172.31.18.232

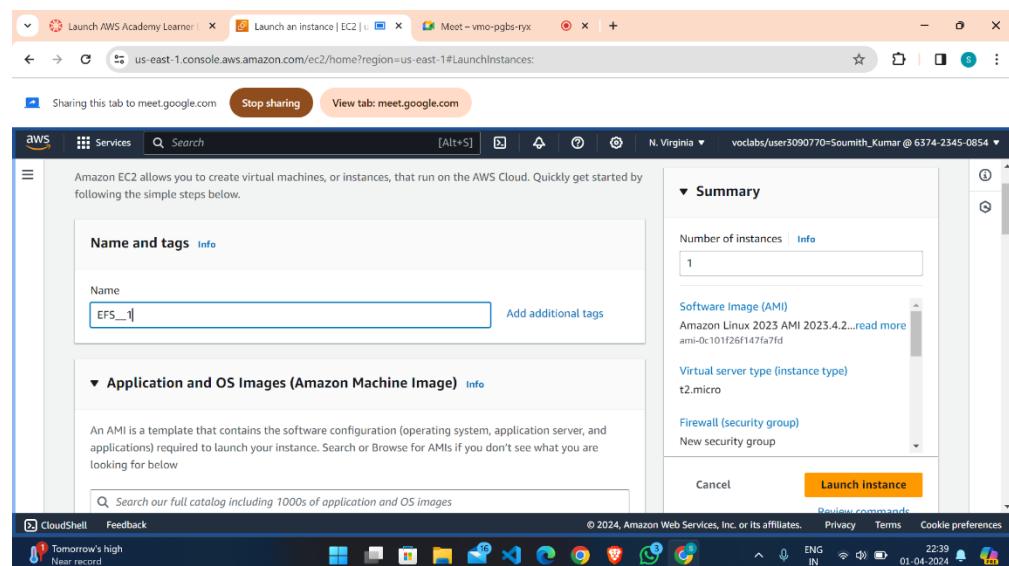
Experiment-4

Aim: Provide a step-by-step demonstration of the process for setting up and configuring file sharing mechanisms using EFS.

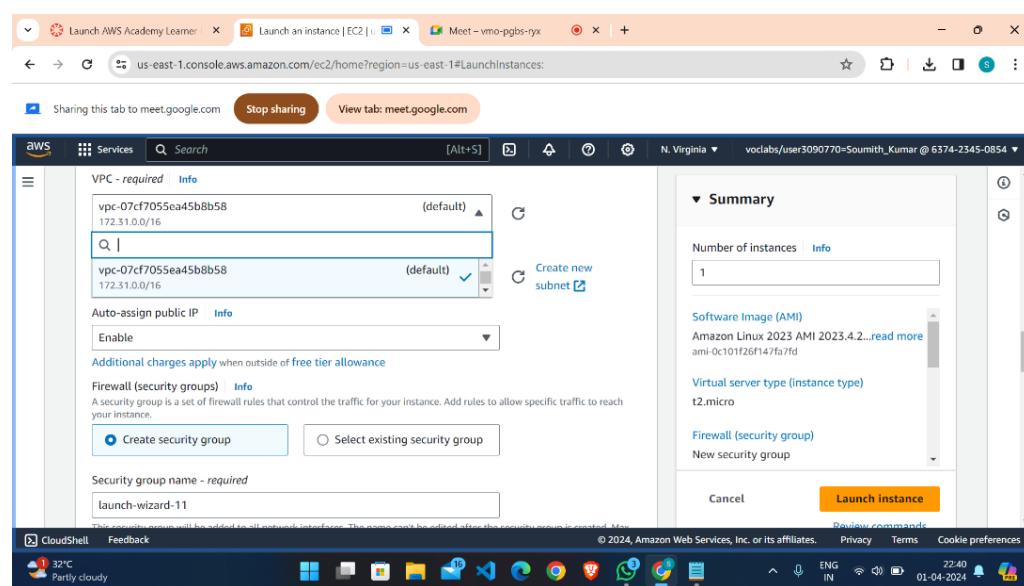
Step 1: Create an instance using following rules-

First Instance:

1. Create a Linux Instance.(EFS_1)
2. In the Network settings click on edit
3. Select the subnet as in which region we want to create instance.(us-east-1a)
4. Select the security group as Create New Security Group.
5. Give the name of the security Group.



The screenshot shows the AWS EC2 'Launch Instances' wizard. On the left, there's a sidebar with 'Services' and a search bar. The main area has tabs for 'Name and tags', 'Application and OS Images (Amazon Machine Image)', and 'Network & security'. Under 'Name and tags', the instance name is 'EFS_1'. Under 'Application and OS Images (Amazon Machine Image)', the AMI is 'Amazon Linux 2023 AMI 2023.4.2...'. Under 'Network & security', the 'Virtual server type (instance type)' is 't2.micro' and the 'Firewall (security group)' is 'New security group'. At the bottom right is a large orange 'Launch instance' button.



The screenshot shows the 'Network & security' step of the EC2 wizard. It lists two subnets: 'vpc-07cf7055ea45b8b58 (default)' and 'vpc-07cf7055ea45b8b58 (default)'. Under 'Auto-assign public IP', 'Enable' is selected. In the 'Firewall (security groups)' section, 'Create security group' is selected. The 'Security group name - required' field contains 'launch-wizard-11'. The 'Launch instance' button is highlighted at the bottom right.

6. Add the security group rule by clicking on it.

7. Set the type as ‘NFS’ and source type as ‘Anywhere’ and click on Launch instance.

The screenshot shows the AWS Cloud Console interface for creating a security group rule. On the left, a sidebar lists 'Security group rule 3 (TCP, 80, 0.0.0.0/0)'. The main panel displays a configuration for a new rule:

- Type: HTTP
- Protocol: TCP
- Port range: 80
- Source type: Anywhere
- Description: e.g. SSH for admin desktop

A warning message at the bottom states: "⚠️ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only." Below this is a button labeled "Add security group rule".

The right side of the screen shows a summary section with the following details:

- Number of instances: 1
- Software Image (AMI): Amazon Linux 2023 AMI 2023.4.2... (read more)
- Virtual server type (instance type): t2.micro
- Firewall (security group): New security group

At the bottom right, there are "Cancel" and "Launch instance" buttons.

This screenshot shows the continuation of the security group rule configuration. A new rule, "Security group rule 4 (TCP, 2049, 0.0.0.0/0)", has been added:

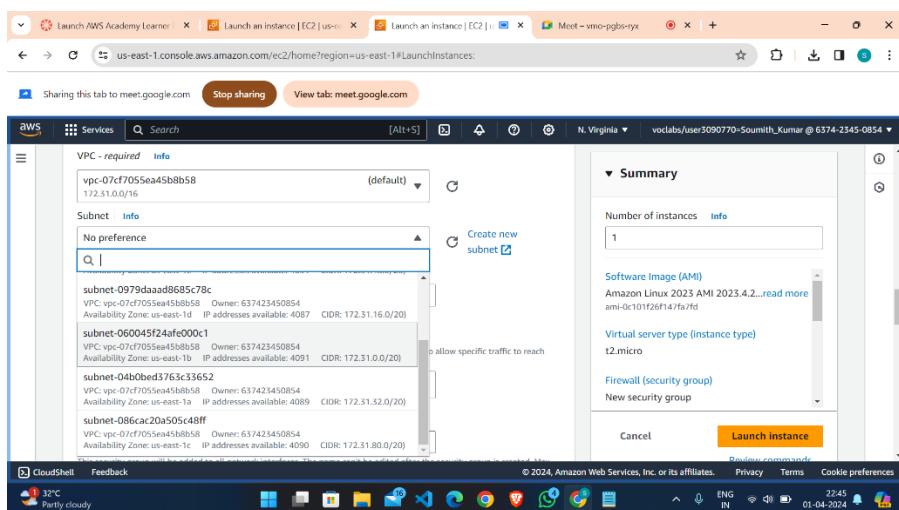
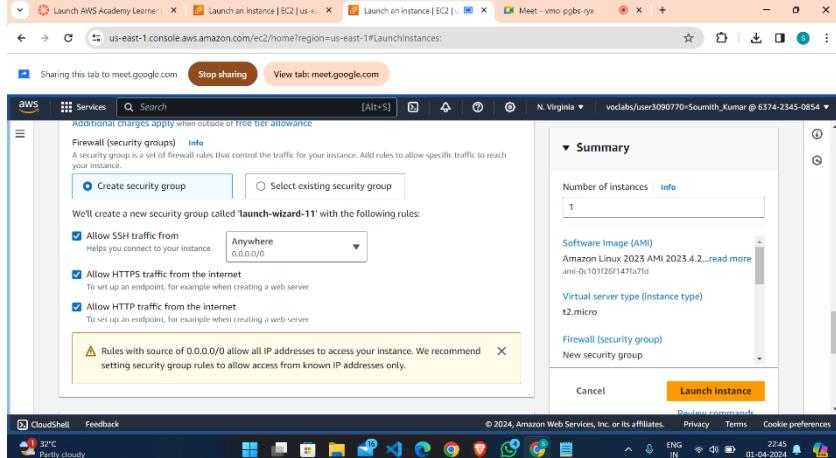
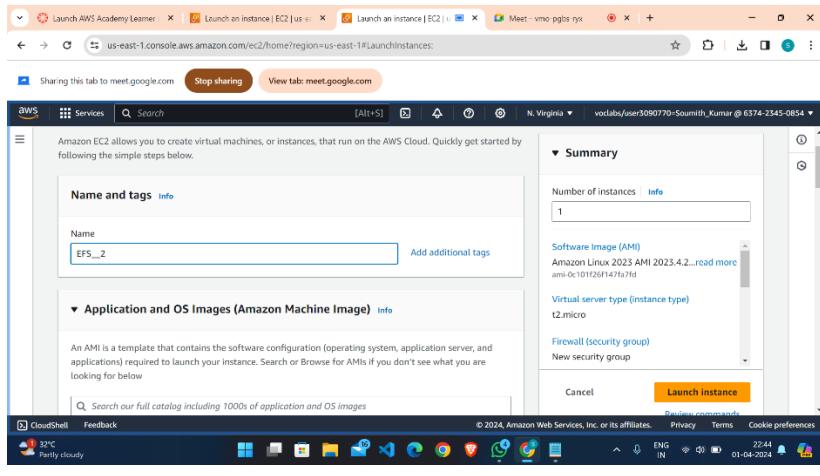
- Type: NFS
- Protocol: TCP
- Port range: 2049
- Source type: Anywhere
- Description: e.g. SSH for admin desktop

The rest of the interface remains the same, including the summary section and the "Launch instance" button.

This screenshot shows the progress of launching the instance. The status bar indicates "21%" completion. The message "Please wait while we launch your instance. Do not close your browser while this is loading." is displayed prominently.

Second Instance:

1. Create another instance with same configuration.(EFS_2)
2. In the Network settings click on edit
3. Select the subnet as in which region we want to create instance.(us-east-1b)
4. Select the security group as Select Existing Security Group.
5. Select the Security group which was created in the first instance from the dropdown.



A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group

Select existing security group

Security group name - required
efs2

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and _-/.@[]+=&,\$^*

Description - required [Info](#)
launch-wizard-11 created 2024-04-01T17:13:50.671Z

Inbound Security Group Rules

▼ Security group rule 1 (TCP, 22, 0.0.0.0/0)

Type [Info](#) Protocol [Info](#) Port range [Info](#)

ssh TCP 22

Remove

Cancel **Launch instance** Review commands

HTTP TCP 80

Source type [Info](#) Source [Info](#) Description - optional [Info](#)
Anywhere [Add CIDR, prefix list or security group](#) e.g. SSH for admin desktop
0.0.0.0/0

▼ Security group rule 4 (TCP, 2049, 0.0.0.0/0)

Type [Info](#) Protocol [Info](#) Port range [Info](#)

NFS TCP 2049

Source type [Info](#) Source [Info](#) Description - optional [Info](#)
Anywhere [Add CIDR, prefix list or security group](#) e.g. SSH for admin desktop
0.0.0.0/0

⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend [restricting access](#).

Cancel **Launch instance** Review commands

EC2 > [instances](#) > Launch an instance

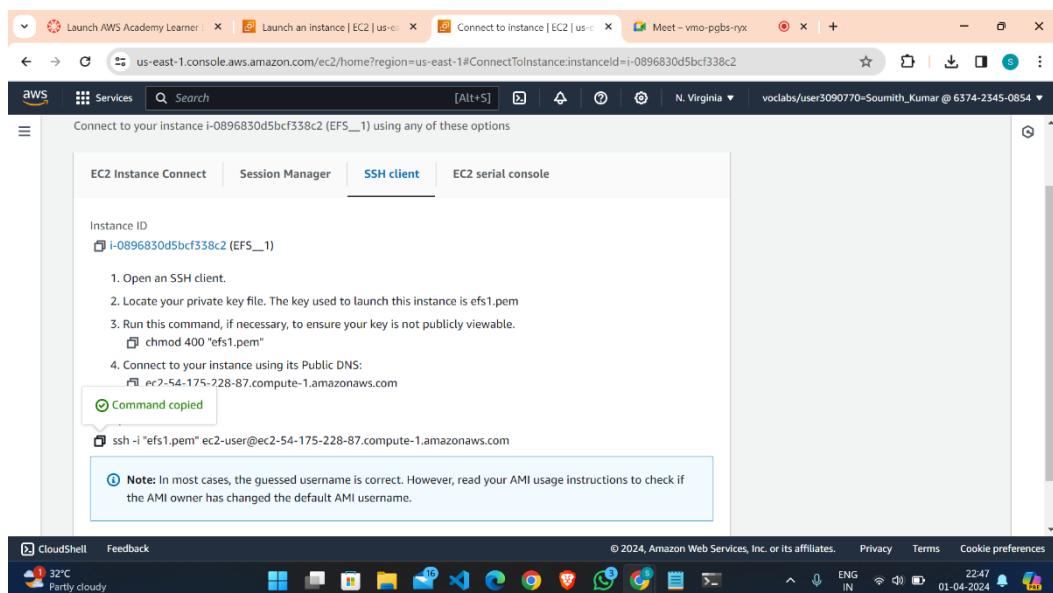
Launching instance
Creating security groups

14%

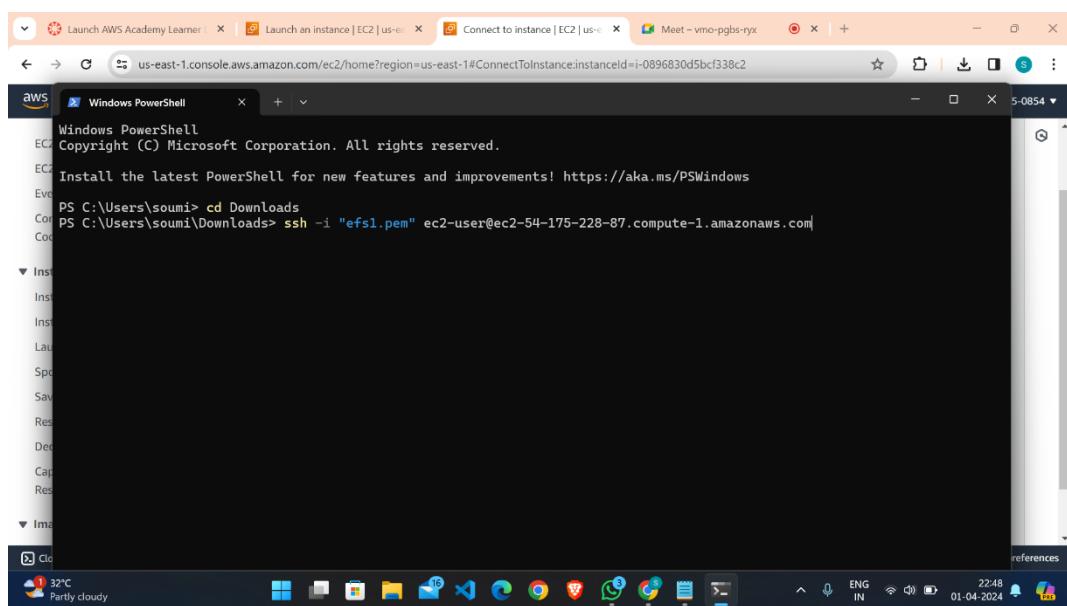
Please wait while we launch your instance.
Do not close your browser while this is loading.

CloudShell Feedback

Step 2: Click on instance 1 and click on connect and choose SSH client , copy the command



- Go to Windows PowerShell and then paste the ssh command



Step 3: Do to the same for instance 2

```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\soumi> cd Downloads
PS C:\Users\soumi\Downloads> ssh -i "efs1.pem" ec2-user@ec2-3-237-180-37.compute-1.amazonaws.com
The authenticity of host 'ec2-3-237-180-37.compute-1.amazonaws.com (3.237.180.37)' can't be established.
ED25519 key fingerprint is SHA256:a3UOTbkfFXCJvPDyZWihpVNYZHgco7+Nmfkj9AoUC8.
This key is not known by any other names
Code
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-237-180-37.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

Instance-1: Amazon Linux 2023
Instantanc
Launch
Spot R
Saving
Reserv
Dedicat
Capaci
Reserv
Image-1

[ec2-user@ip-172-31-3-33 ~]$ sudo su|

```

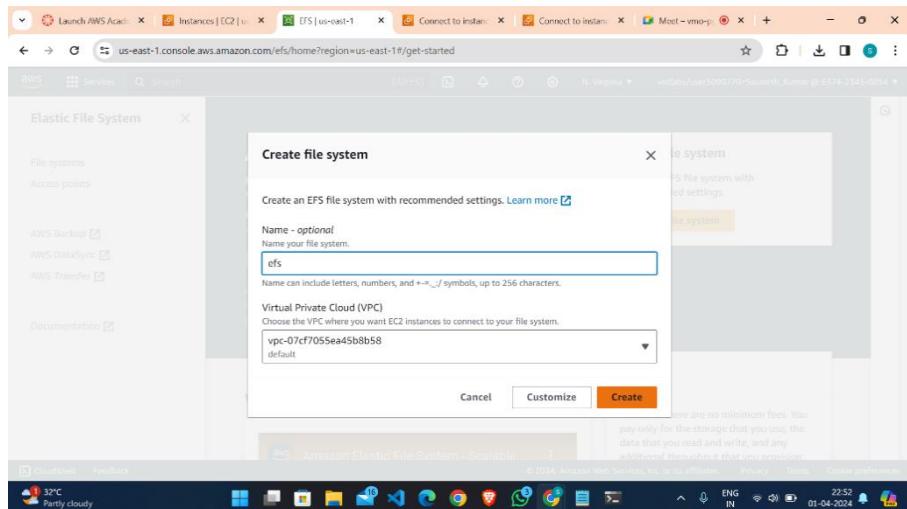
The screenshot shows a Windows PowerShell window with several tabs open. The active tab shows the user is connected to an EC2 instance via SSH, running PowerShell commands to update to the latest version and connect to another instance using a private key. The taskbar at the bottom shows various icons and system status.

- Enter sudo su commands in both terminals

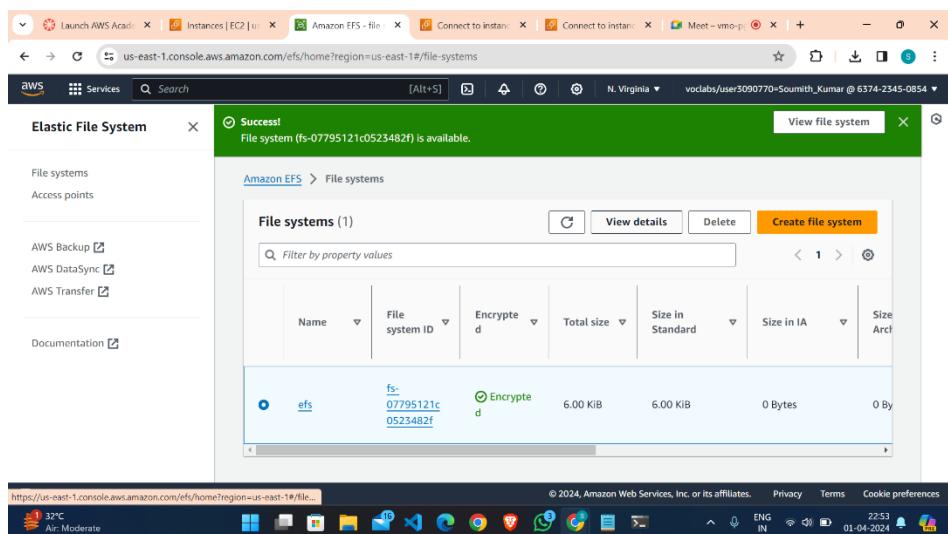
Step 4: Go to AWS and search for EFS in the Services

The screenshot shows the AWS Management Console with the URL <https://us-east-1.console.aws.amazon.com/efs/home?region=us-east-1#/get-started>. The left sidebar shows 'Elastic File System' selected under 'Services'. The main content area displays the 'Amazon Elastic File System' landing page, which includes a 'Create file system' button and information about the service's scalability and pricing.

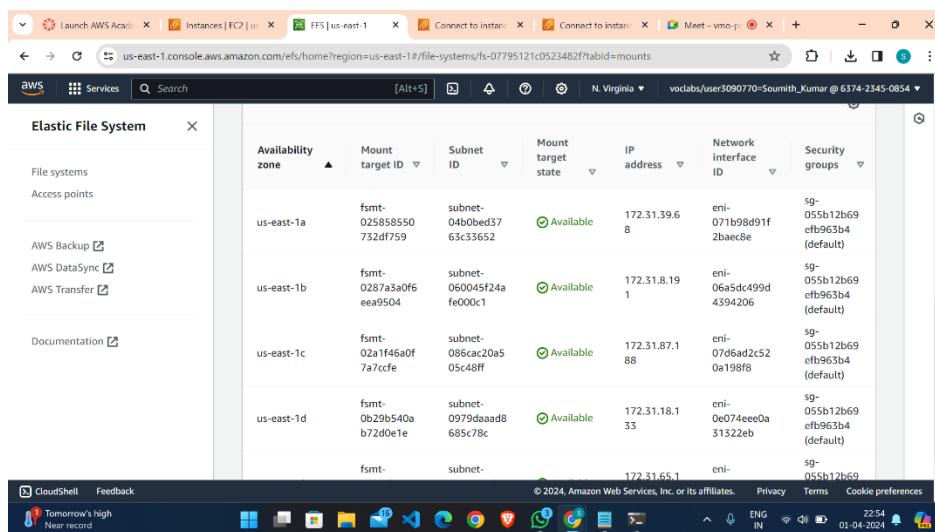
- Click on Create File System and Enter the name of the file system and VPC as default and click on create.



2. Select the efs and click on the hyperlink at the name of the instance



3. Go to Network and wait until all Mount Target State will come to ‘Available’



- I. Click on Manage and change the mount target Security groups to our security group which is created while creating the instance.
- II. Remove the default and select ‘EFS’ for “us-east-1a” and “us-east-1b”

The screenshot shows the AWS EFS console with the 'File systems' section selected. On the right, the 'Network' tab is active under the 'File system policy' section. It displays network configuration details including Availability zone (N. Virginia), IP address (172.31.39.68), and security groups (sg-055b12b69efb, sg-963b4). A table lists mount targets across three availability zones: us-east-1a, us-east-1b, and us-east-1c.

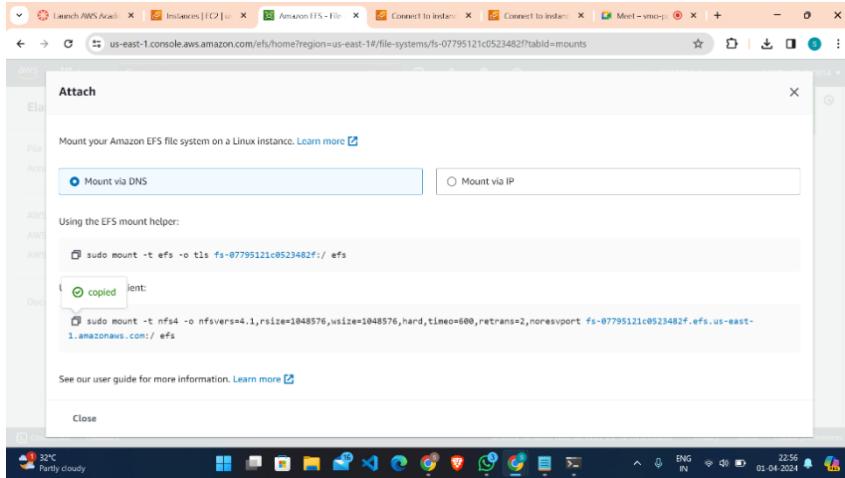
Availability zone	Mount target ID	Subnet ID	Mount target state	IP address	Network interface ID	Security groups
us-east-1a	fsmr-025858550 732df759	subnet-04b0bed37 63c33652	Available	172.31.39.68	eni-071b98d91f 2baec8e	sg-055b12b69efb 963b4 (default)
us-east-1b						
us-east-1c						

The screenshot shows the 'Mount targets' section of the AWS EFS console. It lists three mount targets across availability zones us-east-1a, us-east-1b, and us-east-1c, each associated with a specific subnet and IP address. Security groups are assigned to each target.

Availability zone	Subnet ID	IP address	Security groups
us-east-1a	subnet-04b0bed37	172.31.39.68	sg-08a99ec54354 7de18 nfs
us-east-1b	subnet-060045f24	172.31.8.191	sg-08a99ec54354 7de18 nfs
us-east-1c	subnet-086cac20a!	172.31.87.188	sg-08a99ec54354 7de18 nfs

The screenshot shows the 'Add mount target' dialog box. It allows adding new mount targets by specifying the availability zone, subnet, and IP address. A dropdown menu for security groups is open, showing options like sg-055b12b69efb, sg-963b4, and default. At the bottom, there are 'Cancel' and 'Save' buttons.

- Click on save.
- Now click on attach
- Copy the NFS client address



Step 5: Now go to PowerShell and type the commands in both the terminals.

yum install amazon-efs-utils –y

```

PS C:\Users\asound> cd Downloads
PS C:\Users\asound> wget -q "https://aws.amazon.com/linux/amazon-linux-2023"
[...]
[ec2-user@ip-172-31-3-33 ~]$ sudo su
[root@ip-172-31-3-33 ec2-user]# mkdir efs
[root@ip-172-31-3-33 ec2-user]# yum install amazon-efs-utils

```

- paste the copied nfs client address and hit enter in both the terminals

```

root@ip-172-31-44-98:~# sudo mount -t nfs4 -o nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport fs-07795121c0523482.efs.us-east-1.amazonaws.com:/ efs

```

1. Create a folder using ‘mkdir efs’ in both terminals.
2. Now go to folder using ‘cd efs’
3. Create a file in any of the one terminal using ‘sudo touch FileName’

```
Total download size: 212 k
Installed size: 557 k
Is this ok [y/N]: y
Downloading Packages:
(1/2): stunnel-5.58-1.amzn2023.0.2.x86_64.rpm           2.4 MB/s | 156 kB   00:00
(2/2): amazon-efs-utils-1.35.2-1.amzn2023.noarch.rpm     808 kB/s |  55 kB   00:00
Total                                         1.6 MB/s | 212 kB   00:00

Preparing          :
Installing        : stunnel-5.58-1.amzn2023.0.2.x86_64          1/1
Running scriptlet: stunnel-5.58-1.amzn2023.0.2.x86_64          1/2
Installing        : amazon-efs-utils-1.35.2-1.amzn2023.noarch      1/2
Running scriptlet: amazon-efs-utils-1.35.2-1.amzn2023.noarch      2/2
Verifying         : stunnel-5.58-1.amzn2023.0.2.x86_64          2/2
Verifying         : amazon-efs-utils-1.35.2-1.amzn2023.noarch      1/2
2/2

Installed:
  amazon-efs-utils-1.35.2-1.amzn2023.noarch             stunnel-5.58-1.amzn2023.0.2.x86_64

Complete!
[root@ip-172-31-44-98 ec2-user]# sudo mount -t nfs4 -o nfsvers=4.1,rsize=1048576,wszie=1048576,hard,timeo=600,retrans=2,noresvport fs-07795121c0523482f.efs.us-east-1.amazonaws.com:/ efs
[root@ip-172-31-44-98 ec2-user]# cd efs
[root@ip-172-31-44-98 efs]# touch f1
[root@ip-172-31-44-98 efs]# ls
f1
[root@ip-172-31-44-98 efs]#
```

4. Now Enter ls in both the terminal and we can see f1 file in both the terminals

```
[root@ip-172-31-44-98 ec2-user]# sudo mount -t nfs4 -o nfsvers=4.1,rsize=1048576,wszie=1048576,hard,timeo=600,retrans=2,noresvport fs-07795121c0523482f.efs.us-east-1.amazonaws.com:/ efs
[root@ip-172-31-44-98 ec2-user]# cd efs
[root@ip-172-31-44-98 efs]# touch f1
[root@ip-172-31-44-98 efs]# ls
f1
[root@ip-172-31-44-98 efs]#
```

```
[root@ip-172-31-3-33 ec2-user]# sudo mount -t nfs4 -o nfsvers=4.1,rsize=1048576,wszie=1048576,hard,timeo=600,retrans=2,noresvport fs-07795121c0523482f.efs.us-east-1.amazonaws.com:/ efs
[root@ip-172-31-3-33 ec2-user]# cd efs
[root@ip-172-31-3-33 efs]# ls
f1
```

Experiment-5

Aim: Demonstrate the execution of a simple Python program using AWS Lambda functions. Include step-by-step instructions for creating and configuring the Lambda function, list out the languages supported by AWS Lambda.

Step 1 : We need to open 3 Services parallelly in 3 different tabs

- S3 Bucket
- Dynamo DB
- Lambda Function

Step 2: Navigate to S3 via AWS Management Console.

The screenshot shows the AWS Management Console with the S3 service selected. The main content area displays the "Amazon S3" landing page, which features a large heading "Amazon S3" and the subtext "Store and retrieve any amount of data from anywhere". Below this, there's a section titled "How it works" with a video player showing a thumbnail for "Introduction to Amazon S3". To the right, there are two boxes: "Create a bucket" (containing instructions about buckets and a "Create bucket" button) and "Pricing" (containing information about no minimum fees and a link to the AWS Simple Monthly Calculator). On the left sidebar, there are several navigation links under "Amazon S3" and "Storage". At the bottom, there are links for CloudShell, Feedback, and a copyright notice for 2024.

Step 3: Click on Create Bucket , Choose General purpose radio option and proceed to provide a unique name to the bucket.

The screenshot shows the "Create bucket" wizard in the AWS Management Console. The first step, "General configuration", is displayed. It includes fields for "Bucket name" (set to "LambdaFuncLab") and "AWS Region" (set to "US East (N. Virginia) us-east-1"). Under "Bucket type", the "General purpose" radio button is selected, with a note explaining it's recommended for most use cases. There is also a "Directory - New" option. Below these, there are sections for "Copy settings from existing bucket - optional" and "Format: s3://bucket/prefix". At the bottom, there are buttons for "Choose bucket" and "Next Step". The top navigation bar shows the URL "us-east-1.console.aws.amazon.com/s3/bucket/create?region=us-east-1&bucketType=general".

Step 4: Enable ACLs and uncheck the Block public access checkbox and proceed to accept the conditions by checking the “I acknowledge” checkbox.

The screenshot shows the 'Block Public Access settings for this bucket' section. It includes a note about granting access through various methods like ACLs, bucket policies, or IAM roles. It also shows the 'Block all public access' setting being turned off.

Step 5: Proceed to create the Bucket.

The screenshot shows the 'Advanced settings' step of the creation wizard. It includes sections for 'Encryption type' (set to SSE-S3), 'Bucket Key' (set to 'Enable'), and a note about uploading files after creation. A prominent orange 'Create bucket' button is at the bottom right.

Step 6: Now Navigate to Dynamo DB via Management Console to configure it upon trigger by Lambda function , Proceed to click on create Table.

The screenshot shows the main dashboard of the Amazon DynamoDB Management Console. It features a large central area with the text 'Amazon DynamoDB' and 'A fast and flexible NoSQL database service for any scale'. To the left is a sidebar with links like 'Dashboard', 'Tables', 'Explore items', etc. On the right, there are 'Get started' and 'Pricing' sections.

Step 7: Enter the name of the table and Enter the partition key . Make sure that the table name is same as the name mentioned in the python script and the partition key is same as mentioned in the json file that is to be uploaded into the S3 bucket .

Table details Info
DynamoDB is a schemaless database that requires only a table name and a primary key when you create the table.

Table name
This will be used to identify your table.
 Between 3 and 255 characters, containing only letters, numbers, underscores (_), hyphens (-), and periods (.)

Partition key
The partition key is part of the table's primary key. It is a hash value that is used to retrieve items from your table and allocate data across hosts for scalability and availability.

Sort key - optional
You can use a sort key as the second part of a table's primary key. The sort key allows you to sort or search among all items sharing the same partition key.

Step 8: Proceed to Click on Create Table .

Provisioned write capacity	5 WCU	Yes
Auto scaling	On	Yes
Local secondary indexes	-	No
Global secondary indexes	-	Yes
Encryption key management	Owned by Amazon DynamoDB	Yes
Deletion protection	Off	Yes
Resource-based policy	Not active	Yes

Tags
Tags are pairs of keys and optional values, that you can assign to AWS resources. You can use tags to control access to your resources or track your AWS spending.

No tags are associated with the resource.

You can add 50 more tags.

Step 9: Now, Navigate to Lambda function via AWS Management Console and click on Create Function.

Lambda

- Dashboard
- Applications
- Functions
- Additional resources
 - Code signing configurations
 - Event source mappings
 - Layers
 - Replicas
- Related AWS resources
 - Step Functions state machines

Resources for US East (N. Virginia)

Lambda function(s) 6 **Code storage** 6 kB (0% of 75 GB) **Full account concurrency** 400 **Unreserved account concurrency** 400

Account-level metrics
The charts below show metrics across all your Lambda functions in this AWS Region.

Error count and success rate **Throttles** **Invocations**

Count	%	Count	Count
1	100	1	4
0.5	99.5	0.5	3
0	99	0	2

Step 10: Enter a name and click on Change default execution role , choose Use an existing role and proceed to choose LabRole .

Basic Information

Function name
Enter a name that describes the purpose of your function.
cs1f

Runtime Info
Choose the language to use when writing your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Node.js 20.x

Architecture Info
Choose the instruction set architecture you want for your function code.
x86_64

Permissions Info
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

▼ Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.

Create a new role with basic Lambda permissions

Use an existing role

Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

LabRole

View the LabRole role [on the IAM console](#).

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step 11: Click on Create function.

▼ Advanced settings

Enable Code signing [Info](#)
Use code signing configurations to ensure that the code has been signed by an approved source and has not been altered since signing.

Enable function URL [Info](#)
Use function URLs to assign HTTP(S) endpoints to your Lambda function.

Enable tags [Info](#)
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources, track your AWS costs, and enforce attribute-based access control.

Enable VPC [Info](#)
Connect your function to a VPC to access private resources during invocation.

Create function

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step 12: Once the Function is created it will give us a window to create a Source to create a Trigger , so Click on Add Trigger , Choose S3 as AWS service and choose the bucket we have created .

The screenshot shows the 'Add trigger' configuration page for an AWS Lambda function. The 'Trigger configuration' dropdown is set to 'S3'. The 'Bucket' field contains 's3/lambdafunclab'. The 'Event types' section shows 'All object create events' selected. The 'Prefix - optional' field contains 'e.g. images/'. The 'Add' button is visible at the bottom right.

Step 13: Click on Add to add Source Trigger .

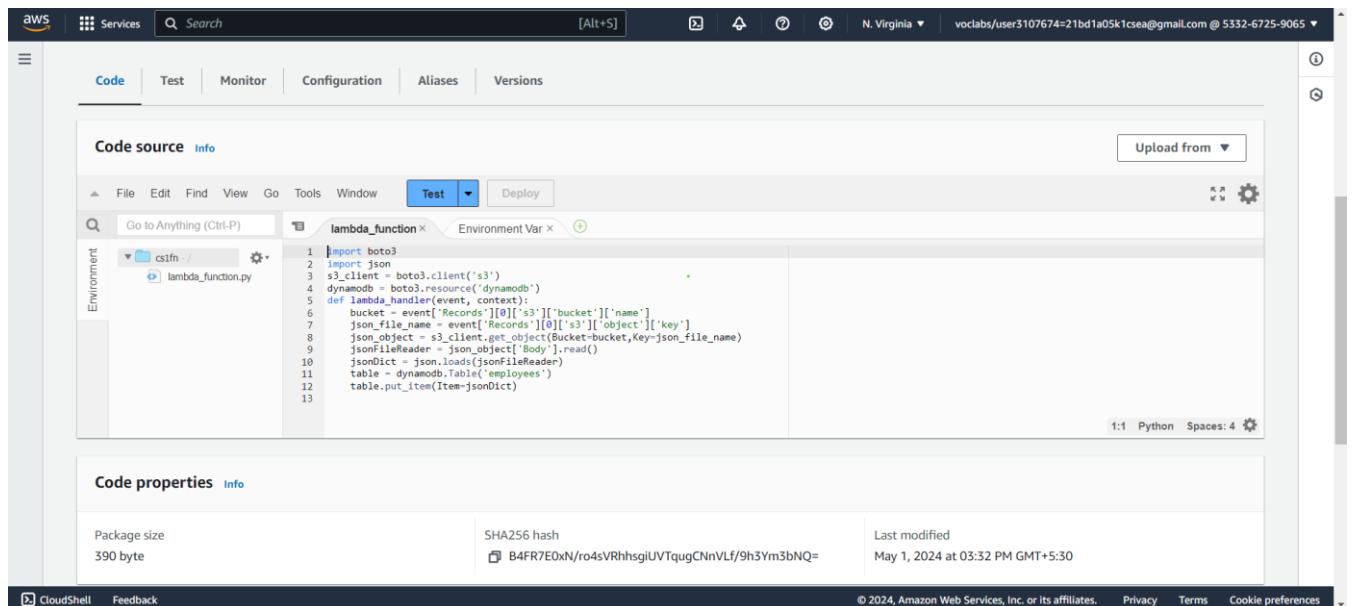
The screenshot shows the 'Add trigger' configuration page for an AWS Lambda function. The 'Trigger configuration' dropdown is set to 'S3'. The 'Bucket' field contains 's3/lambdafunclab'. The 'Event types' section shows 'All object create events' selected. The 'Prefix - optional' field contains 'e.g. images/'. The 'Suffix - optional' field contains 'e.g. jpg'. The 'Recursive invocation' section contains a note about using different S3 buckets for input and output. A checkbox is checked for acknowledging the use of the same S3 bucket for both input and output. The 'Add' button is highlighted in orange at the bottom right.

Step 14: Once Created procced to Code option in the same window and paste to following code

Code:

```
import boto3
import json
s3_client = boto3.client('s3')
dynamodb = boto3.resource('dynamodb')
def lambda_handler(event, context):
    bucket = event['Records'][0]['s3']['bucket']['name']
    json_file_name = event['Records'][0]['s3']['object']['key']
    json_object = s3_client.get_object(Bucket=bucket, Key=json_file_name)
    json.FileReader = json_object['Body'].read()
    jsonDict = json.loads(json.FileReader)
    table = dynamodb.Table('employees')
    table.put_item(Item=jsonDict)
```

→Further proceed and click on Test and give it a name , An error less code will procced to a successful compilation of the code and then proceed to click on Deploy, If not deployed the trigger will not be effective .



The screenshot shows the AWS Lambda function editor interface. The top navigation bar includes the AWS logo, Services, Search, and account information. Below the navigation is a toolbar with Code, Test, Monitor, Configuration, Aliases, and Versions tabs, with Code selected. The main workspace is titled "lambda_function" and contains a "Code source" tab. The code editor displays the Python script provided in the previous step. The code imports boto3 and json, creates S3 and DynamoDB clients, defines a lambda handler, reads a JSON file from S3, converts it to a dictionary, and inserts it into a DynamoDB table. The bottom section shows "Code properties" with details like package size (390 byte), SHA256 hash (B4FR7E0xN/ro4sVRhsqjUVTqugCNnVLF/9h3Ym3bNQ=), and last modified date (May 1, 2024 at 03:32 PM GMT+5:30). The footer includes CloudShell, Feedback, and various AWS links.

Step 15: Now Go back to S3 Bucket we have created and upload the Json file into it and Click on Upload.

The screenshot shows the AWS S3 'Upload' interface. At the top, the navigation bar includes 'Amazon S3 > Buckets > lambdafunclab > Upload'. Below the navigation is a large central area with a placeholder message: 'Drag and drop files and folders you want to upload here, or choose Add files or Add folder.' A sub-section titled 'Files and folders (0)' displays a table header with columns 'Name', 'Folder', and 'Type'. Below the header, a message states 'No files or folders' and 'You have not chosen any files or folders to upload.' At the bottom of the main area, there are 'Remove', 'Add files', and 'Add folder' buttons. Below this is a 'Destination' section with a 'Destination' dropdown set to 's3://lambdafunclab' and a 'Destination details' link. The footer includes standard AWS links like CloudShell, Feedback, and a copyright notice: '© 2024, Amazon Web Services, Inc. or its affiliates.'

Step 16: Click on Add Files to upload the hello.json file and Click on Upload.

This screenshot shows the same AWS S3 'Upload' interface after a file has been added. The 'Files and folders (1 Total, 53.0 B)' section now lists 'hello.json' with a file type of 'application/'. The rest of the interface remains largely the same, including the 'Destination' section and the 'Upload' button at the bottom.

Once we click upload the lambda function is triggered and the result is directed to the Dynamo DB where we have our employees table with emp_id as the partition key.

Step 17: Navigate to Dynamo DB and refresh the page to observe items of the table, Click on Explore items , we can see that the contents uploaded in the form of json in the S3 bucket have been updated in Dynamo DB employees table .

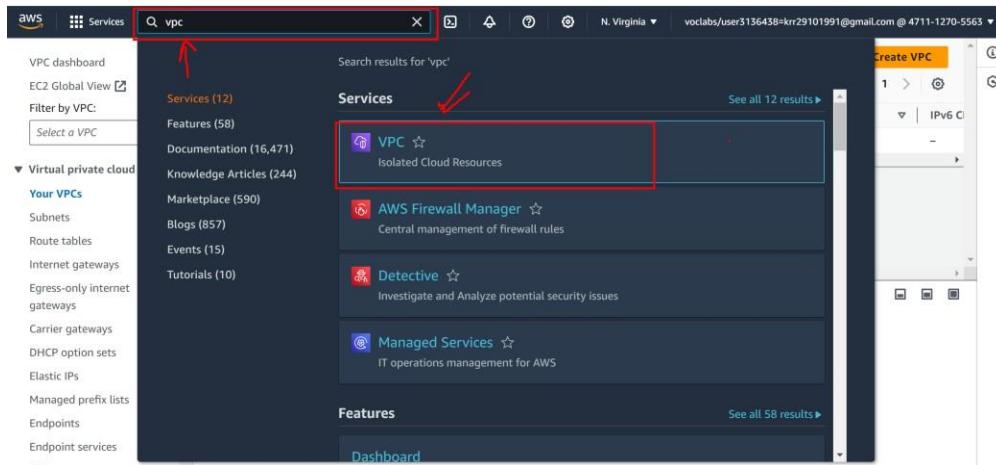
The screenshot shows the AWS DynamoDB 'Explore Items' interface. On the left, the navigation bar includes 'Dashboard', 'Tables', 'Explore items' (which is selected), 'PartiQL editor', 'Backups', 'Exports to S3', 'Imports from S3', 'Integrations', 'Reserved capacity', and 'Settings'. Under 'DAX', it lists 'Clusters', 'Subnet groups', 'Parameter groups', and 'Events'. The main area shows the 'employees' table with three items listed: 'employees', 'something', and 'Student'. A 'Scan' button is selected under 'Scan or query items'. The table details show 'Table - employees' and 'All attributes' for attribute projection. A message indicates 'Completed. Read capacity units consumed: 0.5'. The results table shows one item: emp_id (String) 123, Age 38, Name Bob.

emp_id (String)	Age	Name
123	38	Bob

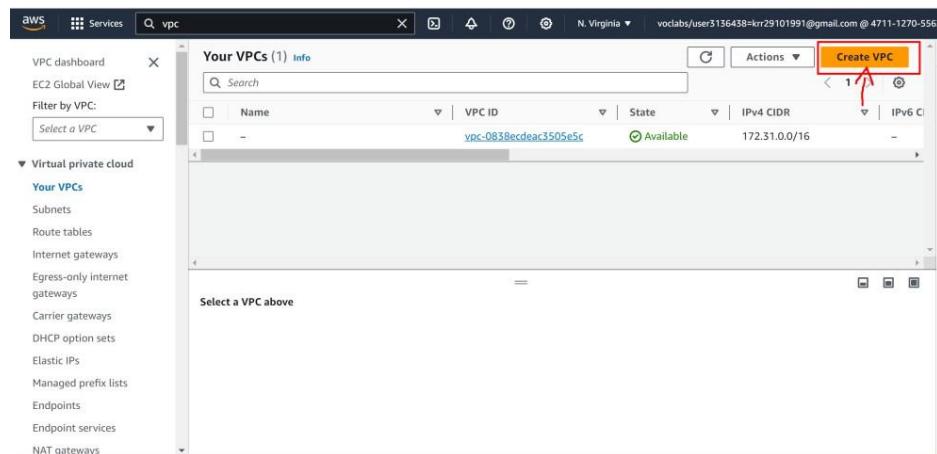
Experiment-6

Aim: Demonstrate the use of VPC using a public and private subnet. Establish the connection with public subnet using Internet Gateway and Route table. As well as launch EC2 instances in each subnet and establish the connection in between two EC2 instances.

Step 1:Firstly Search for **VPC** and click on **VPC**



Step 2 :Click on **Create VPC**



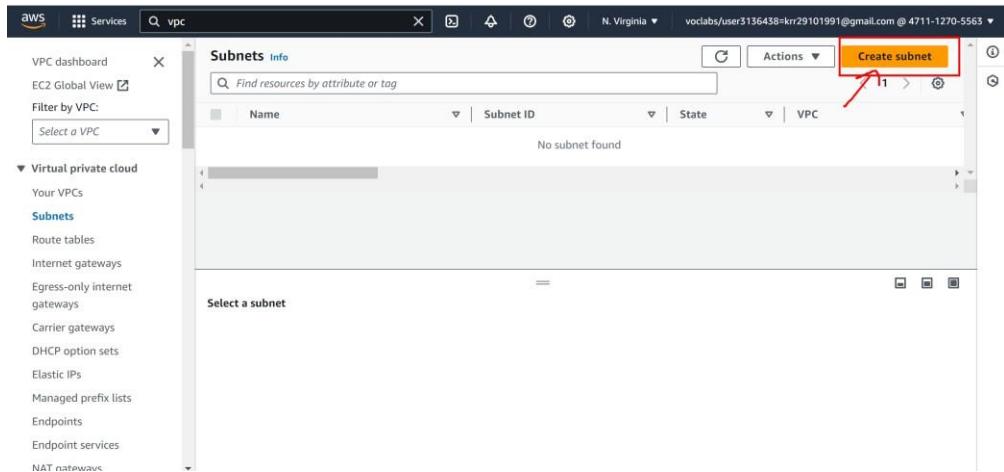
Step 3: Name our VPC and Insert CIDR and Click on Create VPC

The screenshot shows the 'VPC settings' page in the AWS Management Console. The 'Resources to create' section has 'VPC only' selected. A 'Name tag - optional' field contains 'MY-KMIT-VPC'. The 'IPv4 CIDR block' field is set to '192.168.0.0/16'. The 'Tags' section contains a single tag 'Name: MY-KMIT-VPC'. The 'Create VPC' button is highlighted with a red box.

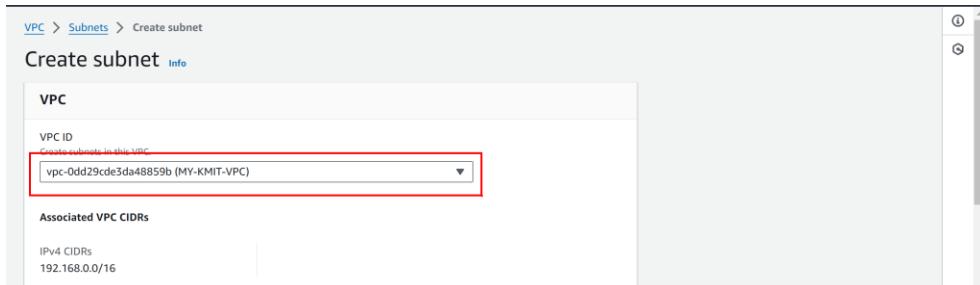
Step 4: Now click on subnets to create two subnets

The screenshot shows the 'VPC dashboard' page. In the left sidebar, under 'Your VPCs', the 'Subnets' link is highlighted with a red box. The main content area displays the details for the VPC 'vpc-0dd29cde3da48859b / MY-KMIT-VPC'. The 'Details' tab is selected, showing information like VPC ID, State (Available), and DNS resolution settings. The 'Resource map' tab is also visible at the bottom.

Step 5: Click on Create Subnet



Step 6: Select the VPC which you have created in previous step from drop down



Then

- Name our subnet
- Select the Availability Zone
- Set the IPV4 Subnet CIDR block
- Click on create subnet

A screenshot of the 'Subnet settings' configuration page. It shows fields for 'Subnet name' (with 'Public-subnet' entered), 'Availability Zone' (with 'US East (N. Virginia) / us-east-1a' selected), 'IPv4 VPC CIDR block' (with '192.168.0.0/16'), and 'IPv4 subnet CIDR block' (with '192.168.1.0/24'). Below these are 'Tags - optional' fields where a tag 'Name' is added with value 'Public-subnet'. At the bottom are 'Cancel' and 'Create subnet' buttons, with the latter highlighted by a red box.

- Again Create one more subnet

The screenshot shows the AWS VPC Subnets list page. On the left, there's a sidebar with options like 'Virtual private cloud', 'Your VPCs', and 'Subnets'. The main area shows a table with one row: 'Public-subnet' (Subnet ID: subnet-0653faaba3d2cd0ec, State: Available, VPC: vpc-0dd29cde3da48859b). At the top right of the main area, there's a prominent orange 'Create subnet' button.

Select our VPC which you have created Then

- Name our subnet
- Select the Availability Zone
- Set the IPv4 Subnet CIDR block
- Click on create subnet

This screenshot shows the 'Create subnet' wizard. Step 1 is 'VPC'. It has a dropdown labeled 'VPC ID' containing 'vpc-0dd29cde3da48859b (MY-KMIT-VPC)', which is also highlighted with a red box.

This screenshot shows the 'Create subnet' wizard. Step 2 is 'Subnet settings'. It includes fields for 'Subnet name' (highlighted with a red box), 'Availability Zone' (highlighted with a red box), 'IPv4 CIDR block' (highlighted with a red box), and 'IPv4 subnet CIDR block' (highlighted with a red box). At the bottom, there are 'Tags - optional' fields and a 'Create subnet' button.

Step 7: Now Create two EC2 instances in two different subnets. Search for EC2 and click on EC2

The screenshot shows the AWS search interface with the search bar containing 'EC2'. The results list includes 'EC2' (Virtual Servers in the Cloud), 'EC2 Image Builder' (A managed service to automate build, customize and deploy OS images), 'Recycle Bin' (Protect resources from accidental deletion), and 'Amazon Inspector' (Continual vulnerability management at scale). The 'EC2' item is highlighted with a red box.

Step 8: Click on Launch Instance

The screenshot shows the AWS EC2 Dashboard. On the left sidebar, under 'Instances', 'Launch Templates' is selected. In the main area, there is a 'Launch instance' button highlighted with a red box. To the right, there is a 'Service health' section showing 'AWS Health Dashboard' and a note that the service is operating normally. A red 'X' is drawn over the 'Launch instance' button.

Step 9: Name the Ec2 Instance

The screenshot shows the 'Launch an instance' wizard. In the 'Name and tags' section, the 'Name' field contains 'Public Ec2', which is highlighted with a red box. The 'Summary' section shows 'Number of instances' set to 1. The 'Software Image (AMI)' section lists 'Amazon Linux 2023 AMI 2023.4.2...' and the 'Virtual server type (instance type)' is set to 't2.micro'.

Step 11: Select the AMI

Application and OS Images (Amazon Machine Image)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below.

Quick Start

My AMIs

Windows (highlighted with a red box)

Red Hat

Browse more AMIs

Amazon Machine Image (AMI)

Microsoft Windows Server 2022 Base
ami-03cd80cfecbb4481 (64-bit (x86))
Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible

Summary

Number of instances: 1

Software Image (AMI): Microsoft Windows Server 2022 ...read more
ami-03cd80cfecbb4481

Virtual server type (instance type): t2.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 30 GiB

Launch instance

Step 12: Create a Keypair

Instance type

t2.micro (highlighted with a red box)
Free tier eligible

Family: t2 - 1 vCPU 1 GiB Memory Current generation: true
On-Demand Windows base pricing: 0.0162 USD per Hour
On-Demand SUSE base pricing: 0.0116 USD per Hour
On-Demand RHEL base pricing: 0.0716 USD per Hour
On-Demand Linux base pricing: 0.0116 USD per Hour

All generations

Compare instance types

Key pair (login)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - **required**: demo123 (highlighted with a red box)

Create new key pair

For Windows instances, you use a key pair to decrypt the administrator password. You then use the decrypted password to connect to your instance.

Summary

Number of instances: 1

Software Image (AMI): Microsoft Windows Server 2022 ...read more
ami-03cd80cfecbb4481

Virtual server type (instance type): t2.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 30 GiB

Launch instance

Step 13: Click on Edit in Network Settings

Network settings

Network: Info
vpc-0838ecdeac3505e5c

Subnet: Info

Auto-assign public IP: Info

Firewall (security groups): Info

We'll create a new security group called 'launch-wizard-1' with the following rules:

Allow RDP traffic from Anywhere
Helps you connect to your instance

Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server

Edit (highlighted with a red box)

Summary

Number of instances: 1

Software Image (AMI): Microsoft Windows Server 2022 ...read more
ami-03cd80cfecbb4481

Virtual server type (instance type): t2.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 30 GiB

Launch instance

Step 14: Select the VPC which you have created then Select the Public subnet and Enable the Auto assign IpAddress

Network settings

VPC - required: vpc-0dd29cde3da48859b (MY-KMIT-VPC)

Subnet: subnet-0653faaba3d2cd0ec (Public-subnet)

Auto-assign public IP: Enable

Summary

Number of instances: 1

Software Image (AMI): Microsoft Windows Server 2022

Virtual server type (instance type): t2.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 30 GiB

Launch instance

Step 15 :Click on Launch Instance

Configure storage

Root volume (Not encrypted): 30 GiB gp2

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage.

Add new volume

The selected AMI contains more instance store volumes than the instance allows. Only the first 0 instance store volumes from the AMI will be accessible from the instance.

Click refresh to view backup information

Number of instances: 1

Software Image (AMI): Microsoft Windows Server 2022

Virtual server type (instance type): t2.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 30 GiB

Launch instance

Step 16:Click on Launch Instance

Instances (5)

Launch instances

Name	Instance ID	Instance state	Instance type	Status check
Public Ec2	i-0bb4dcce15c13eebb	Running	t2.micro	Initializing
Vishnu-Ec2-Private	i-0cb7aee6602ca5f8e	Terminated	t2.micro	-
MY-A2-Private-EC2	i-0513e210d14b28463	Terminated	t2.micro	-
Vishnu-Ec2-public	i-0ff676f04b59a1r2	Terminated	t2.micro	-

Select an instance

Step 17: Name the Ec2 Instance

The screenshot shows the 'Launch an instance' page in the AWS Management Console. In the 'Name and tags' section, the 'Name' field contains 'Private-EC2'. To the right, there's a summary panel with details like the number of instances (1), software image (Amazon Linux 2023 AMI 2023.4.2...), virtual server type (t2.micro), and firewall settings.

Step 18: Select the AMI

The screenshot shows the 'Application and OS Images (Amazon Machine Image)' page. The 'Windows' AMI is selected and highlighted with a red box. On the right, a summary panel shows the selected AMI (Microsoft Windows Server 2022 Base), virtual server type (t2.micro), and storage options (1 volume(s) - 30 GiB). A 'Launch instance' button is visible at the bottom right.

Step 19: Create a Keypair

The screenshot shows the 'Instance type' and 'Key pair (login)' sections of the instance creation wizard. In the 'Key pair (login)' section, the 'Key pair name - required' field is filled with 'demo123'. A note below states: 'For Windows instances, you use a key pair to decrypt the administrator password. You then use the decrypted password to connect to your instance.' On the right, a summary panel shows the selected AMI (Microsoft Windows Server 2022 Base), virtual server type (t2.micro), and storage options (1 volume(s) - 30 GiB). A 'Launch instance' button is visible at the bottom right.

Step 20 : Click on Edit in Network Settings

Network settings

Number of instances: 1

Software Image (AMI): Microsoft Windows Server 2022 ...read more
ami-03cd80cfecbb4481

Virtual server type (instance type): t2.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 30 GiB

Edit

Create security group (radio button selected) **Select existing security group**

We'll create a new security group called 'launch-wizard-1' with the following rules:

- Allow RDP traffic from Anywhere (0.0.0.0/0)
- Allow HTTPS traffic from the internet To set up an endpoint, for example when creating a web server

Launch Instance

Step 21: Select the VPC which you have created then Select the Private subnet and Disable the Auto assign IP Address

VPC - required

vpc-0dd29cde3da48859b (MY-KMIT-VPC)
192.168.0.0/16

Subnet

subnet-0e67aac1c8a88e807 Private-subnet
VPC: vpc-0dd29cde3da48859b Owner: 471112705563 Availability Zone: us-east-1b IP addresses available: 251 CIDR: 192.168.2.0/24

Auto-assign public IP

Disable

Launch Instance

Step 22: Click on Launch Instance

Configure storage

Advanced

1x 30 GiB gp2 Root volume (Not encrypted)

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage

Add new volume

The selected AMI contains more instance store volumes than the instance allows. Only the first 0 instance store volumes from the AMI will be accessible from the instance

Click refresh to view backup information

The tags that you assign determine whether the instance will be backed up by any Data Lifecycle Manager policies.

Launch Instance

Step 23: Now Search for VPC and Select the VPC

The screenshot shows the AWS Services search interface. The search bar at the top contains the text 'VPC'. Below the search bar, there are two main sections: 'Services' and 'Features'. The 'Services' section is expanded, showing 12 results. The first result, 'VPC', is highlighted with a red box. It has a star icon and the description 'Isolated Cloud Resources'. Other services listed include AWS Firewall Manager, Detective, and Managed Services. The 'Features' section below shows 58 results, including Documentation, Marketplace, Blogs, Events, and Tutorials. The 'Dashboard' section is also visible at the bottom.

Step 24: Now click on Internet Gateway

The screenshot shows the VPC dashboard. On the left sidebar, under 'Virtual private cloud', the 'Internet gateways' option is highlighted with a red box. The main content area displays various VPC resources: 'Create VPC', 'Launch EC2 Instances', 'Service Health', 'Settings', 'Additional Information', and 'AWS Network Manager'. There are cards for 'VPCs', 'NAT Gateways', 'Subnets', 'Route Tables', 'VPC Peering Connections', 'Network ACLs', 'Internet Gateways', 'Security Groups', and 'Egress-only Internet'.

Step 25 :Click on Create Internet Gateways

The screenshot shows the 'Internet gateways' list page. The sidebar on the left is identical to the previous VPC dashboard screenshot. The main area shows a table with one item: 'igw-0b9c2f3729e6c175a'. The 'Actions' dropdown menu for this item includes an orange 'Create internet gateway' button, which is highlighted with a red box. Below the table, there is a note: 'Select an internet gateway above'.

Step 26: Name our Internet gateway and click on Create internet gateway

Create internet gateway Info

An internet gateway is a virtual router that connects a VPC to the internet. To create a new internet gateway specify the name for the gateway below.

Internet gateway settings

Name tag
Creates a tag with a key of `Name` and a value that you specify.
MY-IG

Tags - optional
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key Value - optional
Name MY-IG Remove

Add new tag You can add 49 more tags.

Cancel **Create internet gateway**

Step 27 : Click on Attach to a VPC

VPC dashboard EC2 Global View Filter by VPC: Select a VPC

Virtual private cloud Your VPCs Subnets Route tables **Internet gateways** Egress-only internet gateways Carrier gateways DHCP option sets Elastic IPs Managed prefix lists Endpoints Endpoint services

The following internet gateway was created: igw-094293fb993d1b8c6 - MY-IG. You can now attach to a VPC to enable the VPC to communicate with the internet.

igw-094293fb993d1b8c6 / MY-IG

Details Info

Internet gateway ID igw-094293fb993d1b8c6	State Detached	VPC ID -	Owner 471112705563
--	-------------------	-------------	-----------------------

Tags Manage tags

Key	Value
Name	MY-IG

Actions **Attach to a VPC**

Step 28:Select our VPC and Click On Attach Internet Gateway

The following internet gateway was created: igw-094293fb993d1b8c6 - MY-IG. You can now attach to a VPC to enable the VPC to communicate with the internet.

Attach to VPC (igw-094293fb993d1b8c6) Info

VPC
Attach an internet gateway to a VPC to enable the VPC to communicate with the internet. Specify the VPC to attach below.

Available VPCs
Attach the internet gateway to this VPC.
vpc-0dd29cde3da48859b X
Use: "vpc-0dd29cde3da48859b"

vpc-0dd29cde3da48859b - MY-KMIT-VC

Cancel **Attach internet gateway**

Step 29: Now click on Route Tables

The screenshot shows the AWS VPC dashboard. On the left, there's a sidebar with various VPC-related options like 'Your VPCs', 'Subnets', 'Route tables', etc. The 'Route tables' option is highlighted with a red box. The main content area shows the details of an Internet gateway named 'igw-094293fb993d1b8c6 / MY-IG'. It displays the Internet gateway ID, state (Attached), VPC ID, and owner information. Below this, there's a 'Tags' section with a search bar and a table showing a single tag named 'MY-IG'.

Step 30 :Click On create Route tables Name our Route table

The screenshot shows the AWS VPC dashboard with the 'Route tables' section selected. It lists two existing route tables: 'rtb-0c186a4ad99036295' and 'rtb-054d3b7cd61e50a3'. Below the list, there's a 'Select a route table' section. At the top right of the list area, there's a 'Create route table' button, which is highlighted with a red box.

Step 31: Select our VPC and Click Create Route Table

This screenshot shows the 'Create route table' dialog box. In the 'Route table settings' section, the 'Name - optional' field contains 'My-RT' and the 'VPC' dropdown is set to 'vpc-0dd29cde3da48859b (MY-KMIT-VPC)'. Both of these fields are highlighted with red boxes. In the 'Tags' section, there's a single tag 'Name: My-RT'. At the bottom right of the dialog, there's a 'Create route table' button, which is highlighted with a red box.

Step 32: Now Click on Edit Routes

The screenshot shows the AWS VPC Route Tables interface. On the left, there's a sidebar with 'Virtual private cloud' navigation. In the main area, it shows a route table named 'rtb-03d01f603145c517e / My-RT'. Under the 'Routes' tab, there is one route entry: 'Destination: 192.168.0.0/16, Target: local, Status: Active, Propagated: No'. A red box highlights the 'Edit routes' button at the top right of the table.

Step 33: Click on Add Route

The screenshot shows the 'Edit routes' dialog box. It has fields for 'Destination' (192.168.0.0/16), 'Target' (local), and 'Status' (Active). A red box highlights the 'Add route' button at the bottom left. At the bottom right, there are 'Cancel', 'Preview', and 'Save changes' buttons, with 'Save changes' being highlighted.

Step 34 :Now add from Any Where IP Address then select the Internet Gateway from the drop down and select our Internet Gateway then click on Save Changes

The screenshot shows the 'Edit routes' dialog box again. This time, the 'Destination' field is set to '0.0.0.0/0' and the 'Target' dropdown is set to 'Internet Gateway'. A red box highlights both the 'Add route' button and the 'Save changes' button at the bottom right. The 'Save changes' button is highlighted.

Step 36: Click on Subnet Association

Route table ID: rtb-03d01f603145c517e / My-RT successfully

Destination	Target	Status	Propagated
0.0.0.0/0	igw-094293fb993d1b8c6	Active	No
192.168.0.0/16	local	Active	No

Step 37: Click on Edit Subnet Associations

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
Public-subnet	subnet-0653faaba3d2cd0ec	192.168.1.0/24	-
Private-subnet	subnet-0e67aac1c8a88e807	192.168.2.0/24	-

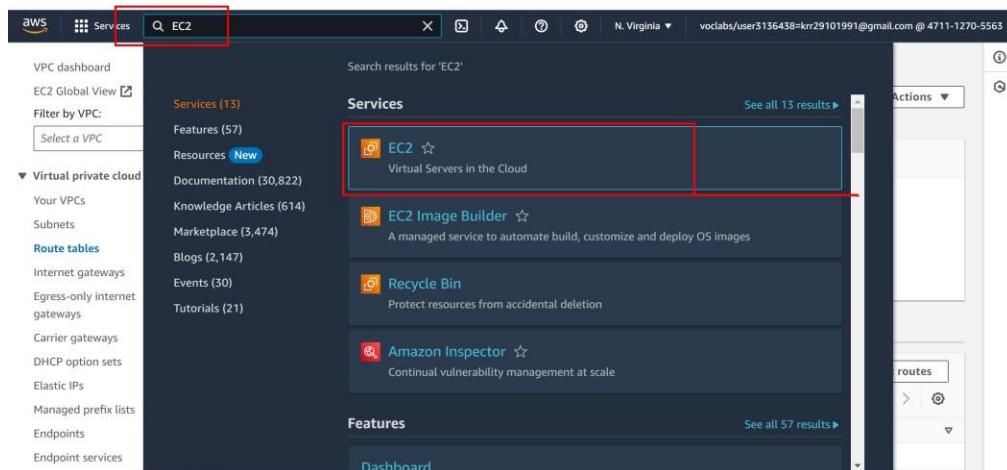
Step 38: Select the Public-subnet and click on save associations

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
<input checked="" type="checkbox"/> Public-subnet	subnet-0653faaba3d2cd0ec	192.168.1.0/24	-	Main (rtb-0c186a4ad99036295)
<input type="checkbox"/> Private-subnet	subnet-0e67aac1c8a88e807	192.168.2.0/24	-	Main (rtb-0c186a4ad99036295)

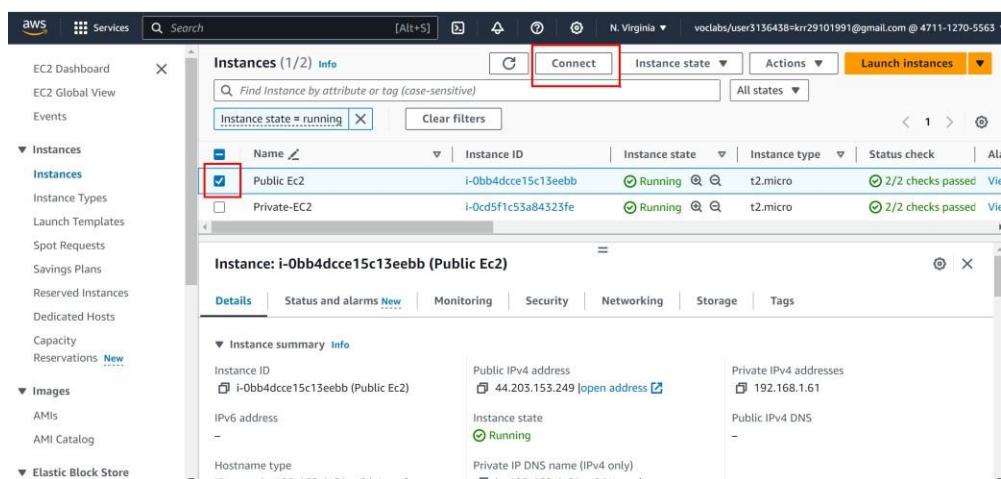
Selected subnets: subnet-0653faaba3d2cd0ec / Public-subnet

Save associations

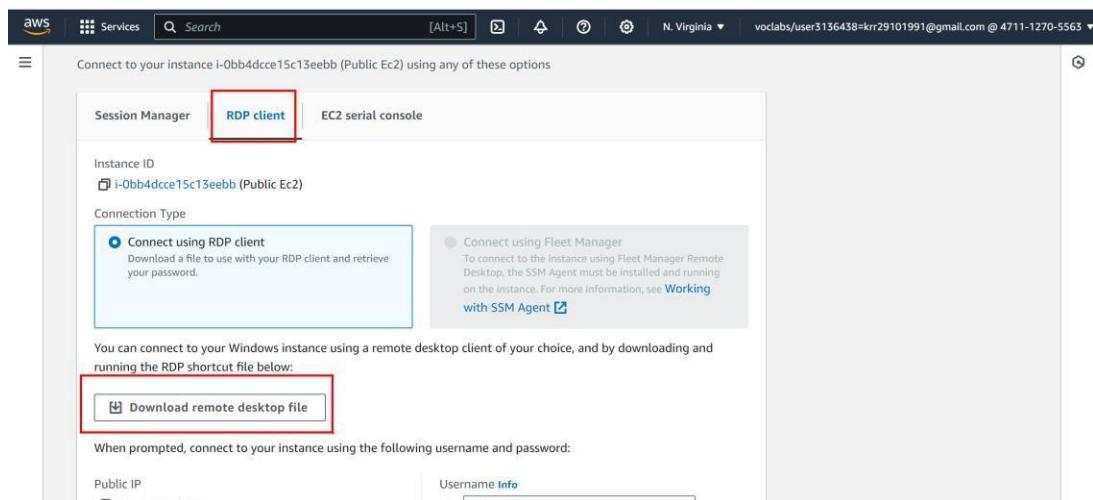
Step 39 : Now search for EC2 And click on the Ec2



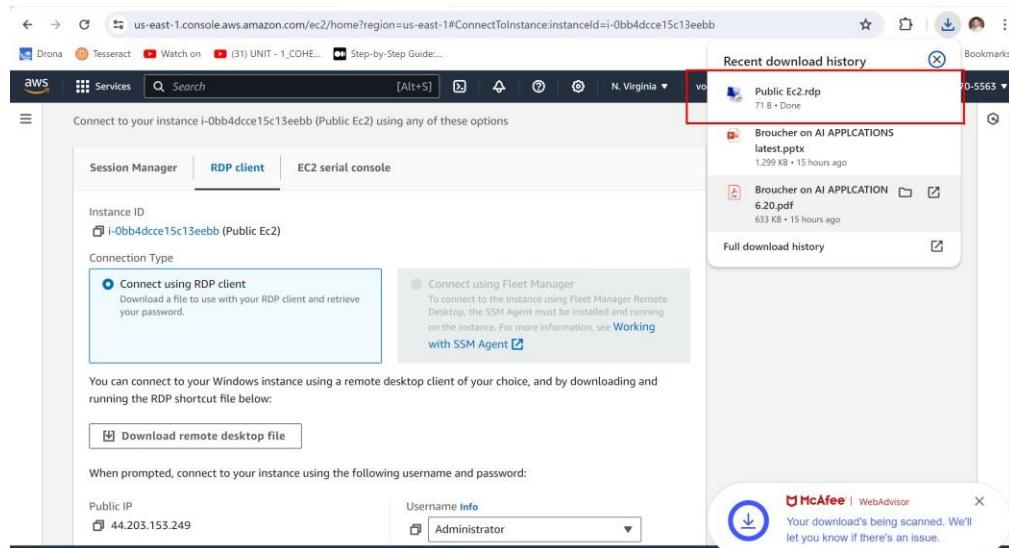
Step 40:Select the EC2 Instance and Click on Connect



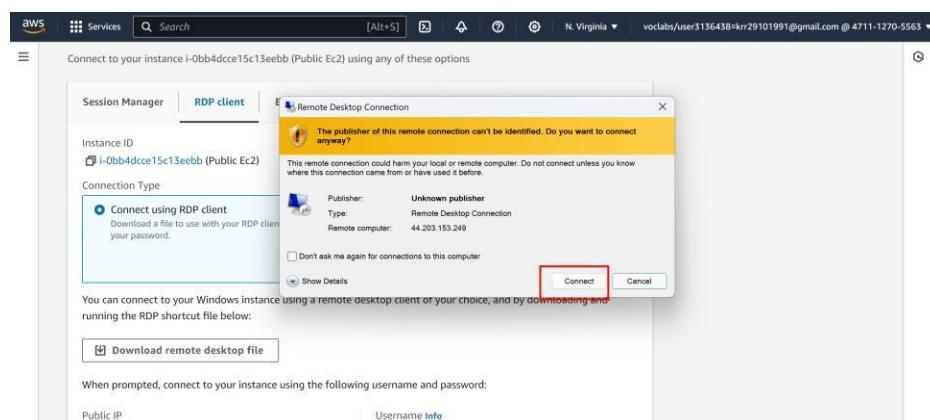
Step 41:Click on RDP Client and Click on Download remote desktop file



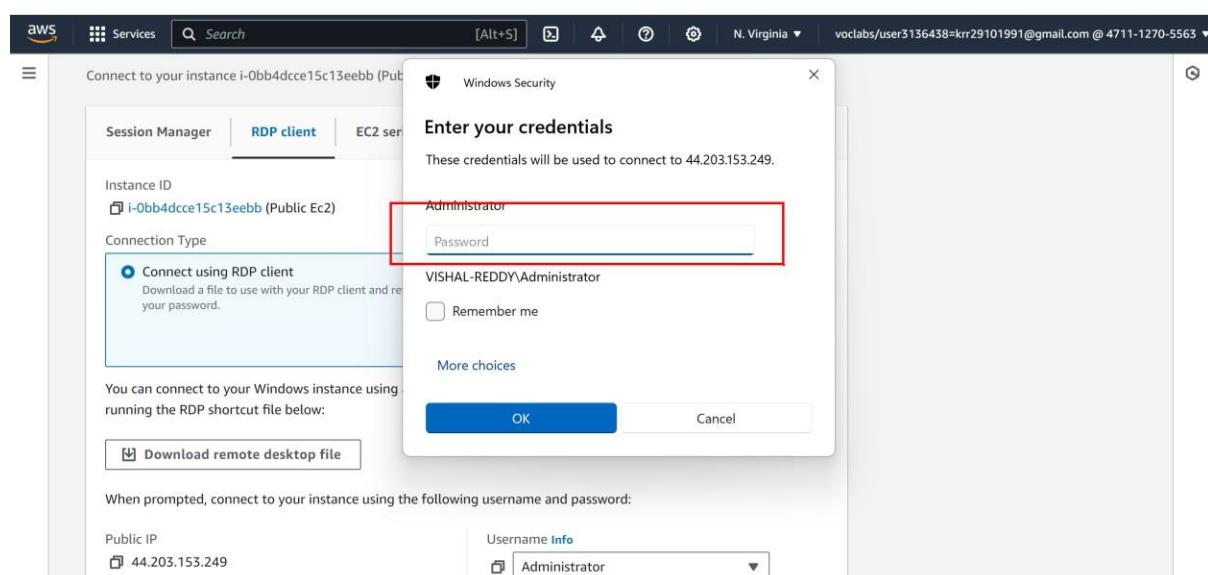
Step 42: Open the downloaded file by clicking on it



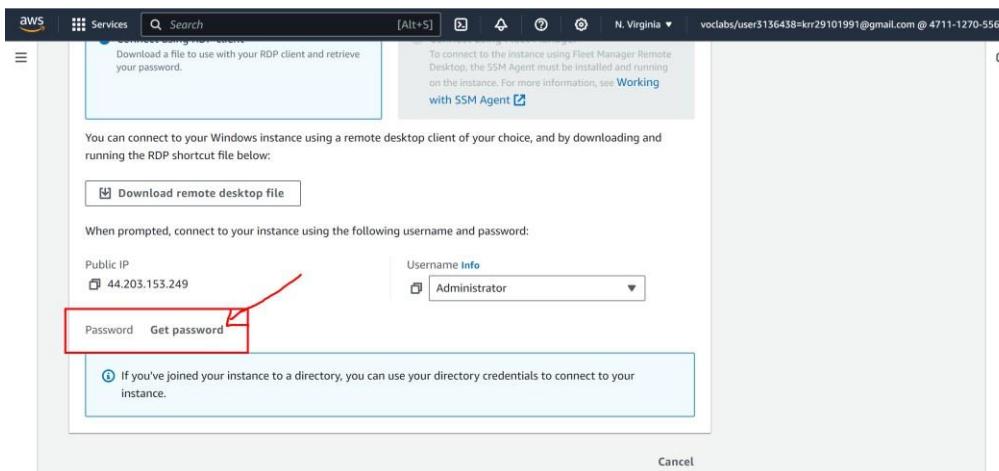
Step 43 :Click on connect



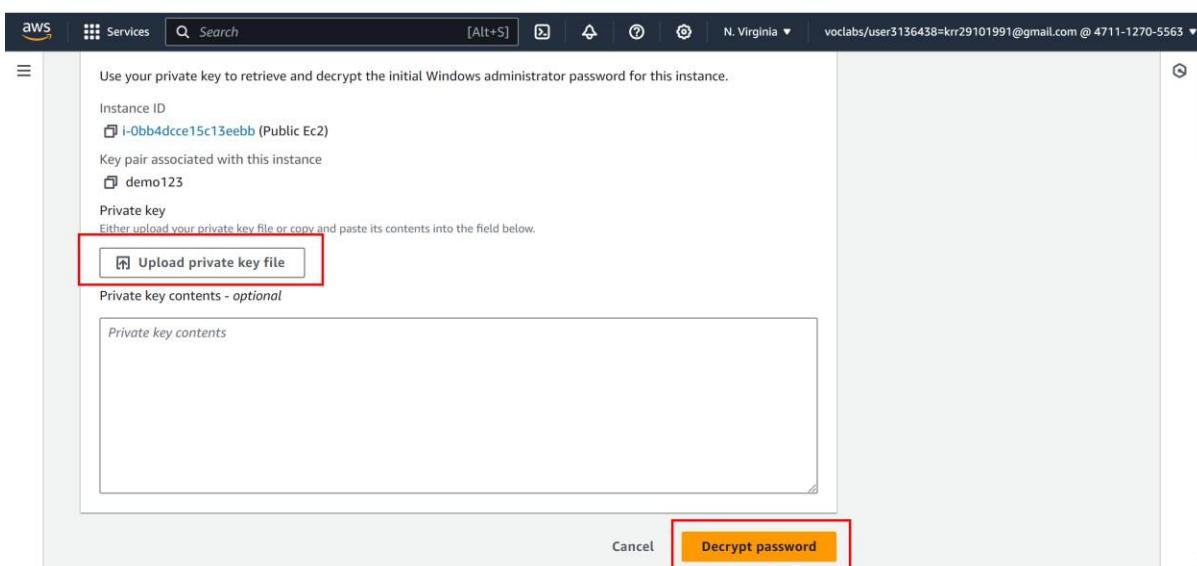
It will prompt for Password



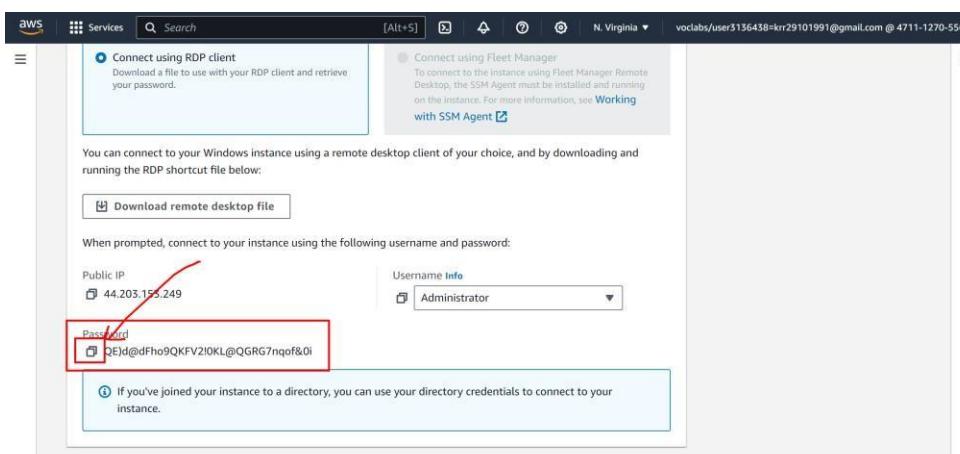
Step 44 : For Password get back to AWS EC2 Browser and click on get password

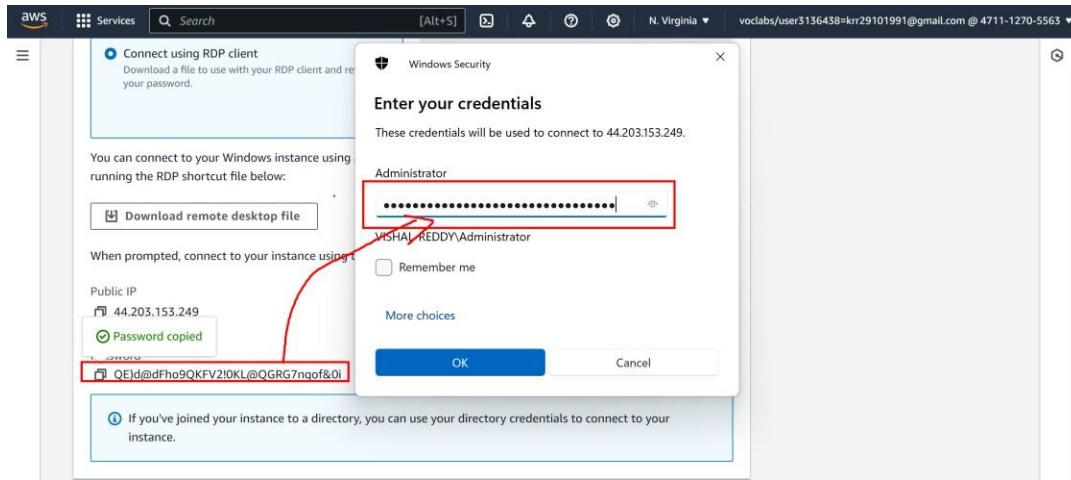


Step 45: Upload our private key and click on decrypt password

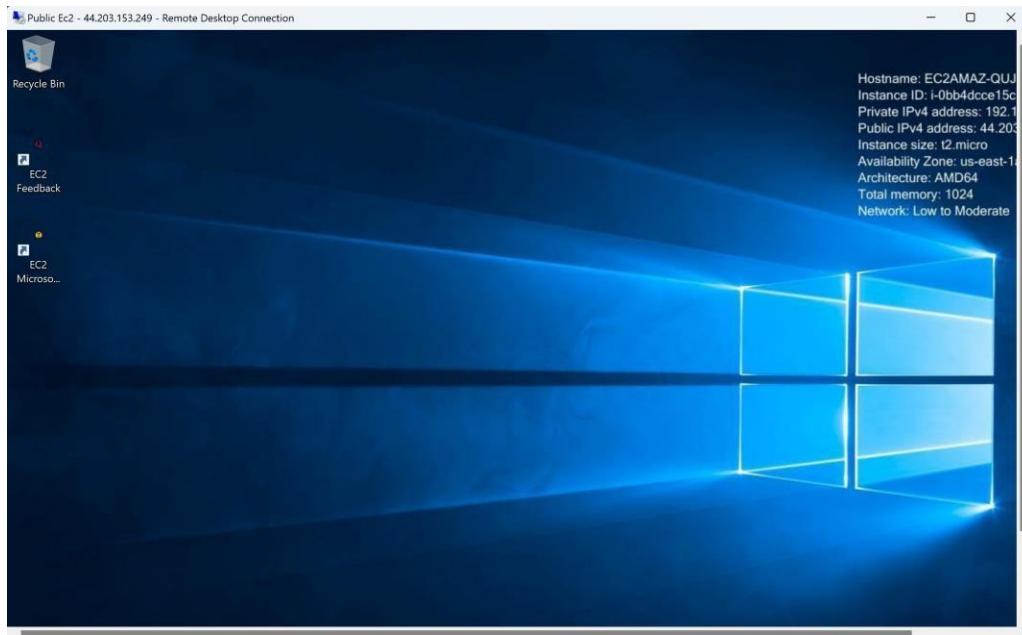


Step 46: Copy the password and paste it on the window and then click on OK and yes

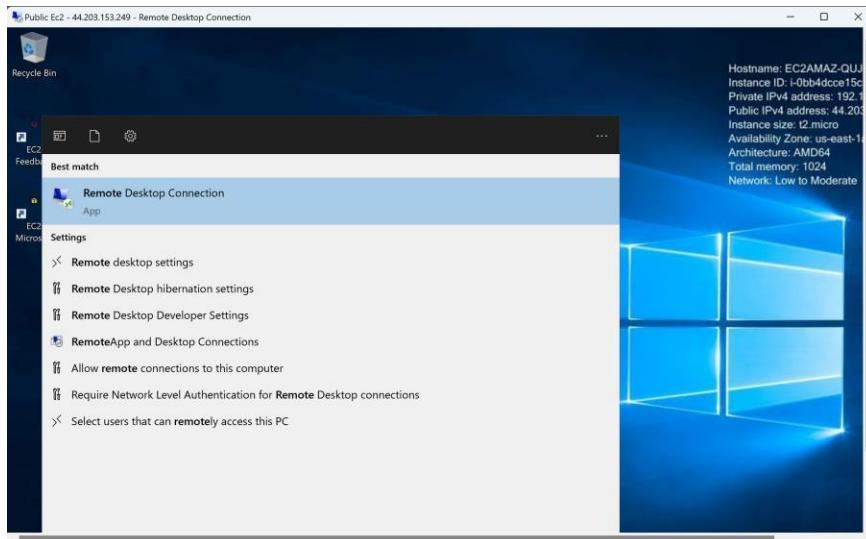


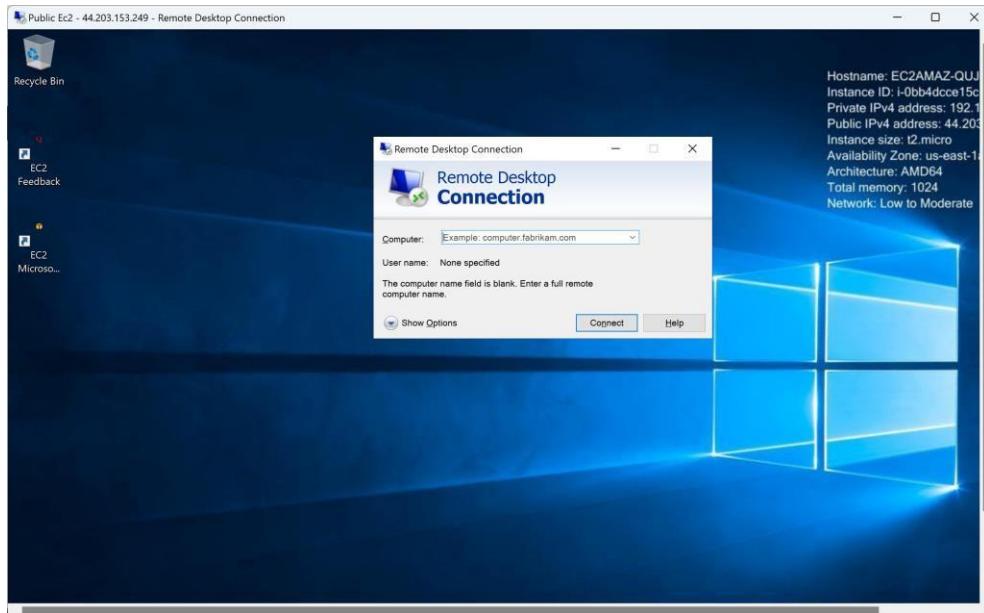


Our EC2 WILL BE OPENED



Step 47: Search for remote desktop connection within our EC2 Instance



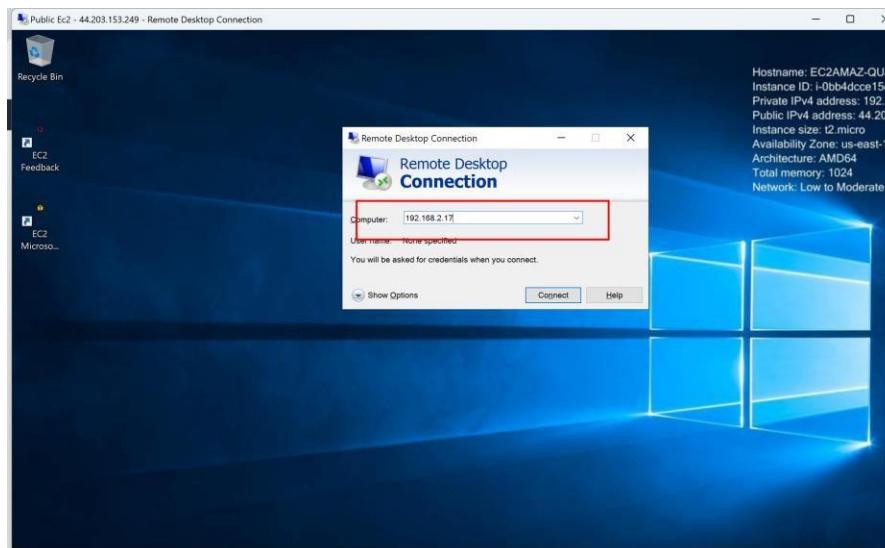


Step 48: Go back to the Browser and Open instances now select the Private EC2 instance and click on connect

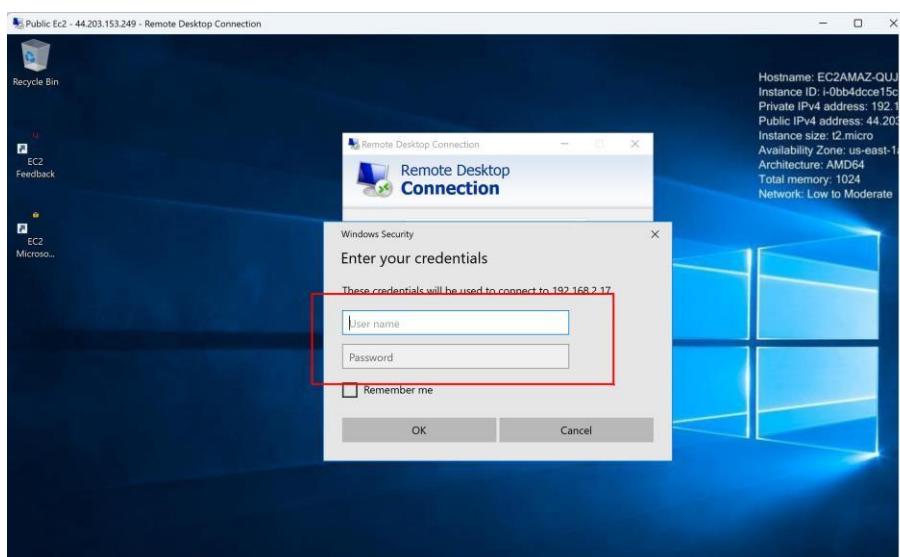
Name	Instance ID	Instance state	Instance type	Status check
Public Ec2	i-0bb4ddce15c13eebb	Running	t2.micro	2/2 checks passed
Private-EC2	i-0cd5f1c53a84323fe	Running	t2.micro	2/2 checks passed

Step 49: Click on RDP Client and Copy the Private IP

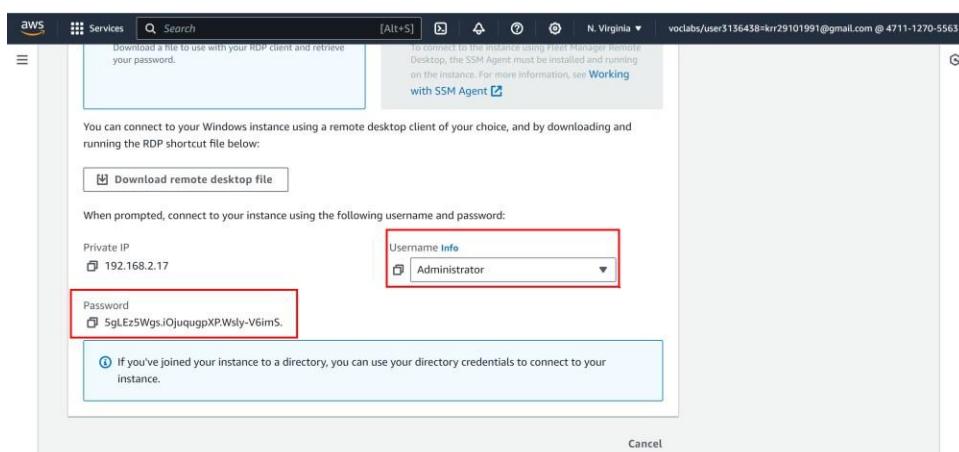
Step 50: Paste the Private Ip on the EC2 Remote Desktop Connection and then click on connect



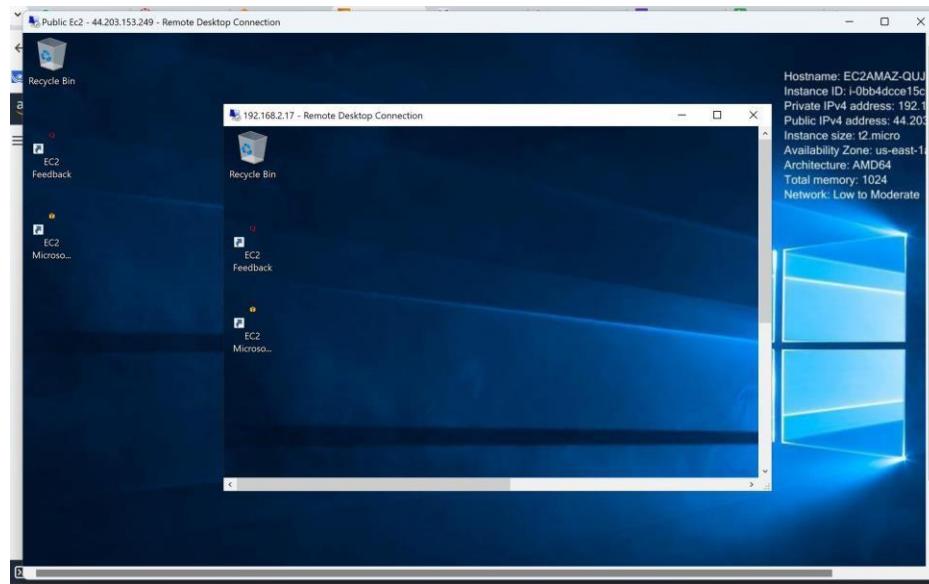
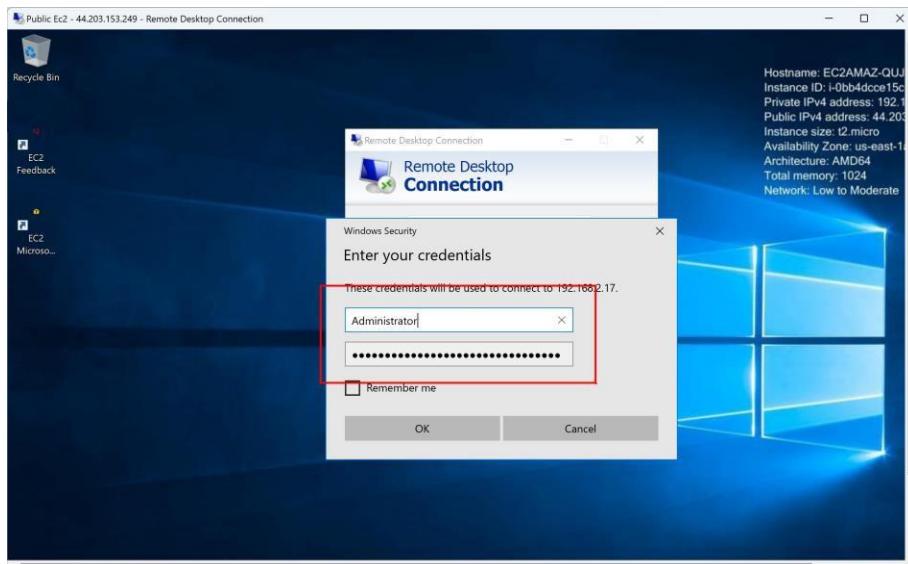
Step 51: You will be prompted for Username and Password



Step 52: Now go back to EC2 Browser and click on the getpassword and upload private key and click on decrypt password copy the password



Step 53: Paste the Username and Password on the EC2 Instance and click on ok and then yes



Experiment-7

Aim: Create a help desk assistant integrating Amazon LEX.

Step 1 : Navigate to Amazon Lex service via Amazon Console.

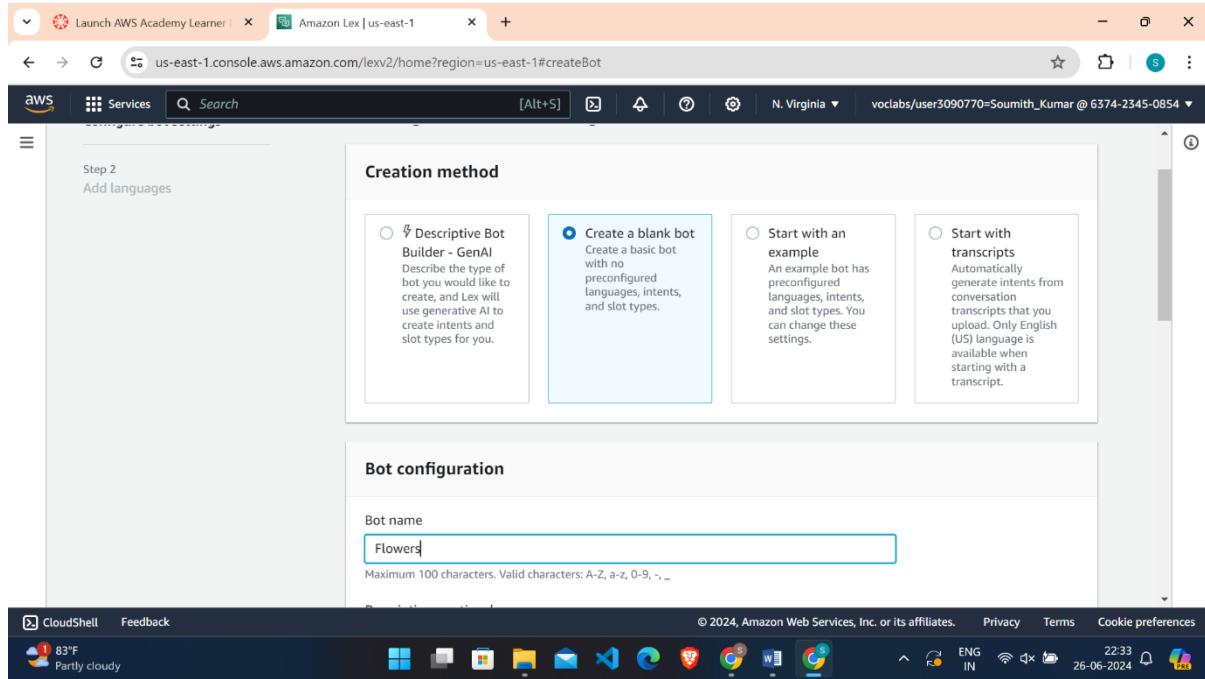
The screenshot shows the Amazon Lex service home page. At the top, there's a banner for "Machine Learning". Below it, the title "Amazon Lex" is displayed with the subtitle "Conversational AI for self-service bots". A sub-section titled "How it works" shows "Step 1: Script conversation". To the right, there are two callout boxes: "Build a bot using Generative AI" (describing Descriptive Bot Builder) and "Pricing (US)" (mentioning pay-as-you-go pricing). The bottom navigation bar includes links for CloudShell, Feedback, and various AWS services like Lambda, S3, and CloudWatch.

Step 2: Click on Create bot

The screenshot shows the "Configure bot settings" step of the creation wizard. It's titled "Configure bot settings" and includes a "Creation method" section. The "Descriptive Bot Builder - GenAI" option is selected, showing a brief description of how it generates intents and slot types. Other options include "Create a blank bot", "Start with an example", and "Start with transcripts". A note at the bottom states: "You must have Amazon Bedrock set up in order to use this feature. Please ensure you have requested access to Anthropic's V2 model." The bottom navigation bar is identical to the previous screenshot.

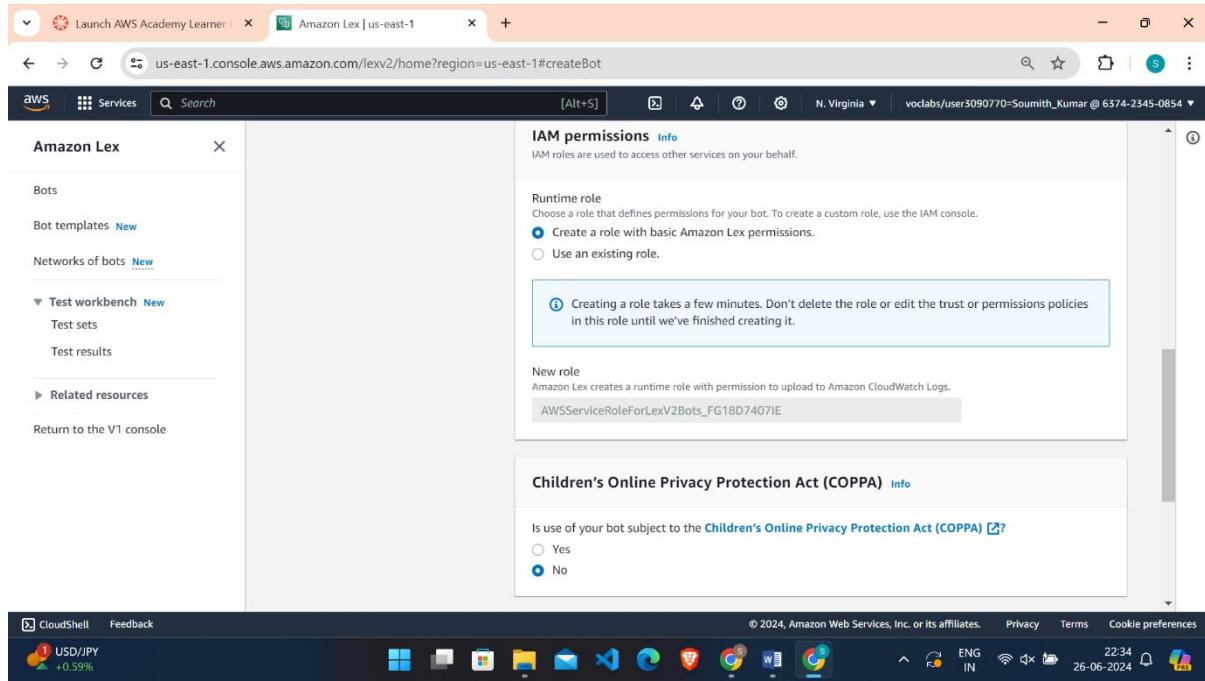
Step 3: Configure the bot settings

- Select Create a blank bot
- Give name for the bot.



Step 4: In IAM permissions Choose create a role with basic Lex Permissions.

- Check No radio option in the COPPA field.
- Click on next.



Step 5: Click on Done

The screenshot shows the 'Add language to bot' step in the Lex console. On the left, there's a sidebar with 'Step 1 Configure bot settings' and 'Step 2 Add languages'. The main area is titled 'Add language to bot' with an 'Info' link. It has a section for 'Language: English (US)' with a dropdown menu set to 'English (US)'. Below it is a 'Description - optional' field with a note about maximum 200 characters. Under 'Voice interaction', it says 'None. This is only a text based application'. At the bottom right of the main area is a 'Done' button.

Step 4: Give Intent name as ‘Flowers’

The screenshot shows the 'Intent details' configuration for the newly created bot. The title bar says 'Successfully created bot: Flowers'. The 'Intent name' field is filled with 'Flowers'. The 'Intent and utterance generation description' field is empty. At the bottom, there are tabs for 'Editor' (which is selected), 'Visual builder', and 'New'. A large orange 'Save intent' button is located at the bottom right. The status bar at the bottom indicates 'CloudShell' and 'Feedback'.

Step 5: Scroll down for Utterances

- Add some utterances like “Hi”, “Hello”, “book order” etc. When these utterances listen by the bot these intent will be invoked.

The screenshot shows the Amazon Lex console interface. The main title bar says "Successfully created bot: Flowers". Below it, there are tabs for "Draft version", "English (US)", and "Not built". A message indicates "English (US) has not built changes." with buttons for "Build" and "Test". The main area displays a list of utterances under the "NewIntent" intent:

- Hi
- I want to order flowers
- order flowers
- Hello
- start

At the bottom right is a "Save intent" button. The footer includes standard AWS navigation links and a status bar showing "CloudShell Feedback" and system metrics like S&P 500 at -0.03%.

Step 6: Initial Response – when bot listen some input these message will be sent first . Generally , Initial response message is used to greet the user.

- In the message box we can write our initial response (“Hello, How can I help we today”).

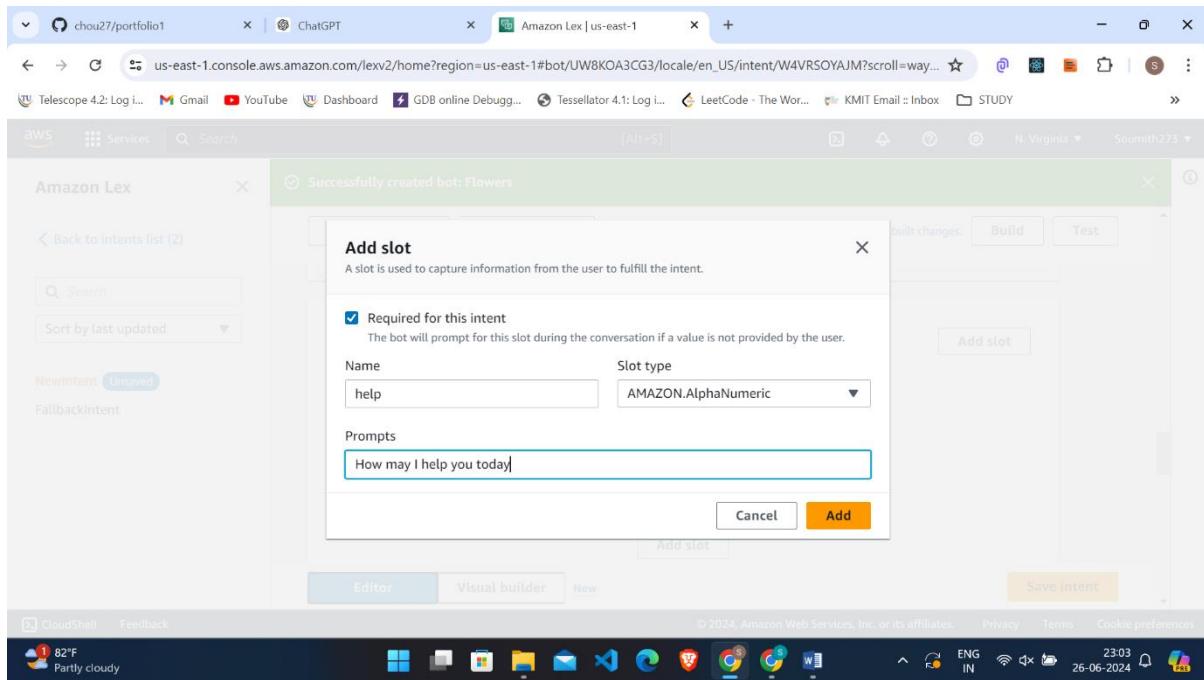
The screenshot shows the Amazon Lex console interface for the "Flowers" bot. The main title bar says "Successfully created bot: Flowers". Below it, there are tabs for "Draft version", "English (US)", and "Not built". A message indicates "English (US) has not built changes." with buttons for "Build" and "Test". The main area displays the configuration for the "Hello" initial response:

- Response to acknowledge the user's request**: Message: Hello.How may I help you today!
- Message group Info**: You can define a text message group to respond using plain text.
Message - optional: Hello.How may I help you today!
- Variations - optional**: Advanced options

At the bottom right is a "Save intent" button. The footer includes standard AWS navigation links and a status bar showing "CloudShell Feedback" and system metrics like S&P 500 at -0.03%.

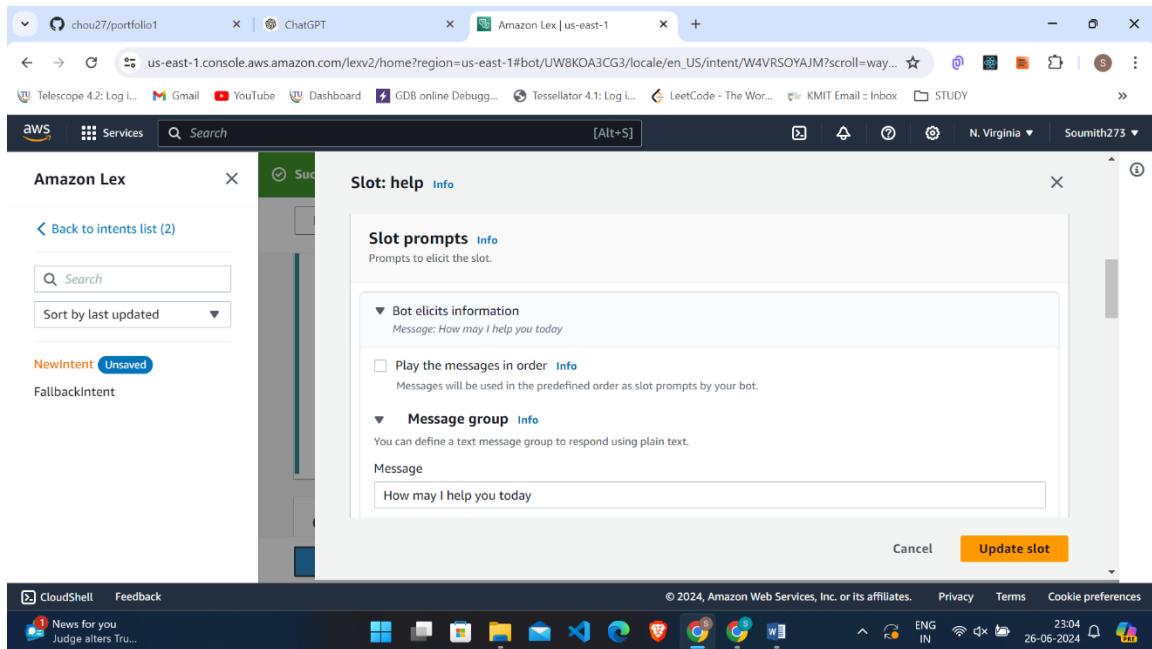
Step 7: Slots - Slots are values provided by the user to fulfill the intent.

- Click on “add slot”
- Give the name of the slot . eg: name ,flower,wishes,etc.
- Select the slot type . there are several types like alphanumeric, date, city .
- Give prompt like how bot should respond .



Step 8: To add cards .

- Select the slot
- Click on Advanced Options
- Under “Slot Prompts” , click on Bot elicits information .



- Click on More prompt options.

The screenshot shows the AWS Lambda Slot prompts editor. In the center, there's a modal window titled "Slot: help > Slot prompts editor". The first section is "Prompts settings" with the sub-instruction "Additional settings for how your bot uses the prompts." It contains three checked checkboxes:

- Users can interrupt the prompt when it is being read. A note below says: "This functionality is available only in streaming conversations."
- Play the messages in order. A note below says: "Messages will be used in the predefined order as slot prompts by your bot."
- Invoke Lambda code hook after each elicitation.

 Below these checkboxes is a text input field labeled "Invocation label - optional" with the placeholder "Maximum 100 characters. Valid characters: A-Z, a-z, 0-9, -, _". At the bottom right of the modal are "Cancel" and "Update prompts" buttons. The background shows the AWS Lambda interface with tabs like "Newintent" and "FallbackIntent".

- Scroll down for slot prompts and click on add .

This screenshot shows the same AWS Lambda Slot prompts editor interface, but the modal window now displays the "Slot prompts" section. It includes a "Preview" button and a "Message group" entry. The "Message" field contains the text "How may I help you today". At the bottom right of the modal are "Cancel" and "Update prompts" buttons. The background remains the same with the "Newintent" tab selected.

- Select “Add card group” .

The screenshot shows the AWS Lambda console interface. The main title is "Slot: help > Slot prompts editor". Below it, a message states: "Maximum number of retries cannot be greater than 5." There is a section titled "Slot prompts" with a "Preview" button. To the right, a context menu is open under "Message group" with the following options: "Add", "Add text message group", "Add card group", "Add custom payload", and "Add SSML language". At the bottom right of the editor are "Cancel" and "Update prompts" buttons.

- Give the title such as “flowers”
- Click on Buttons
- Add Button

The screenshot shows the AWS Lambda console interface. The main title is "Slot: help > Slot prompts editor". Below it, a message says: "You can use response card with necessary resources, static / third party or your own client applications." There is a section for "Image URL" with a field containing "http://www.example.com/image.png" and a note: "Must be an Amazon S3 object URL.". Below that is a "Title" field with the value "flowers" and a note: "Maximum 250 characters. Valid characters: A-Z, a-z, 0-9, @, #, \$". There is also a "Subtitle" field with a note: "Maximum 250 characters. Valid characters: A-Z, a-z, 0-9, @, #, \$". Under "Buttons - optional", there is a "Add button" button. At the bottom right of the editor are "Cancel" and "Update prompts" buttons.

- Add buttons . For each Button , give Button title(which will displayed in the card) and Button value(returned when we select that button).
- Click on update prompts and then update slot.

The screenshot shows the 'Slot: help > Slot prompts editor' window in the AWS Lambda console. It displays four buttons with their respective titles and values:

- Button 1 title:** Rose
Button 1 value: rose
- Button 2 title:** Hibiscus
Button 2 value: hibiscus
- Button 3 title:** Lotus
Button 3 value: lotus
- Button 4 title:** (empty)
Button 4 value: (empty)

Below the buttons are two buttons: 'Cancel' and 'Update prompts'.

Step 9: To add conditional branching to slot.

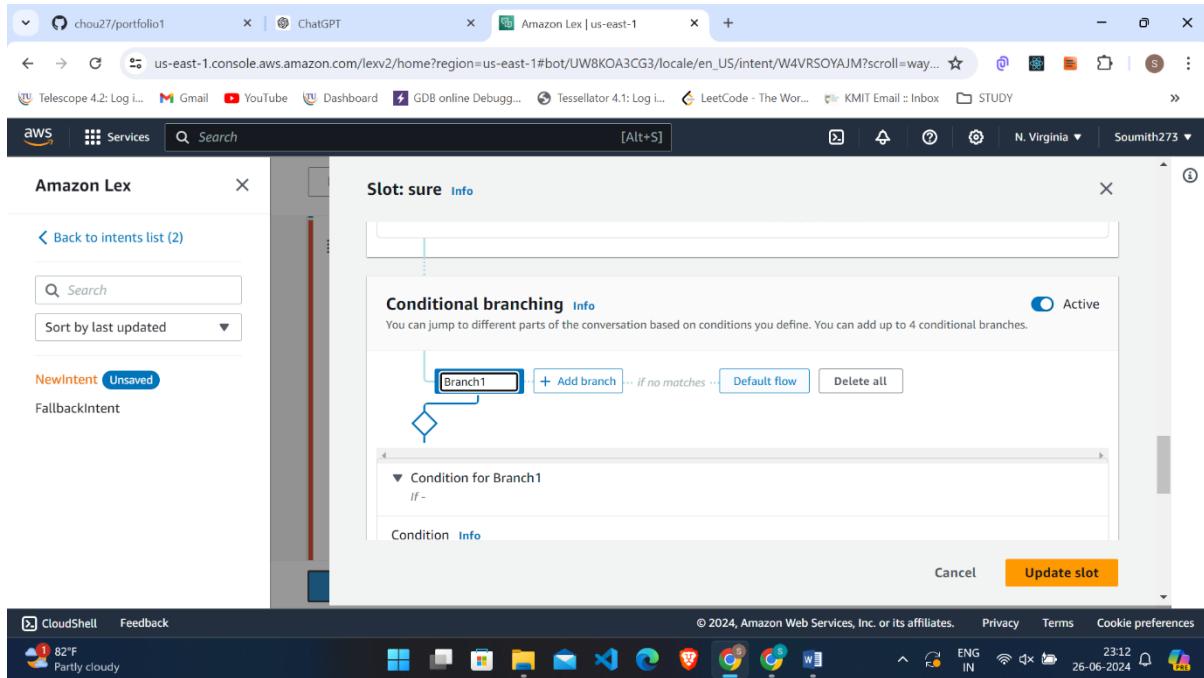
- Create a slot .

The screenshot shows the 'Add slot' dialog box in the AWS Lambda console. The dialog box contains the following fields:

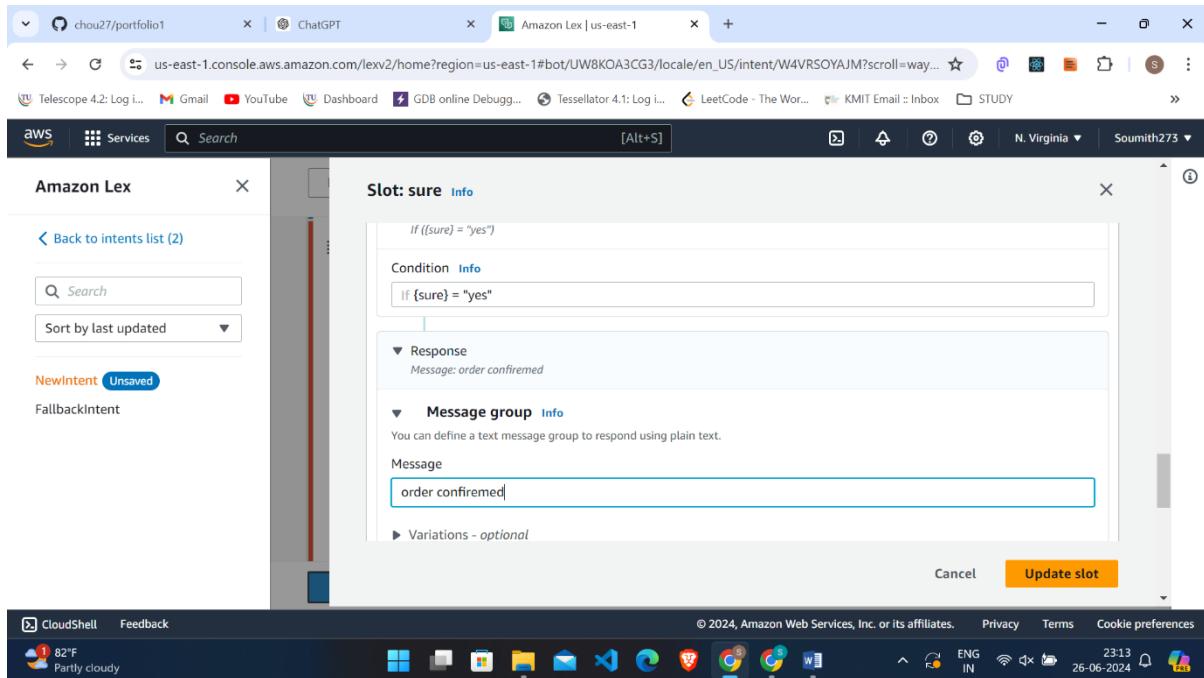
- Required for this intent**: A checked checkbox indicating the slot is required for the intent.
- Name**: The name of the slot is 'sure'.
- Slot type**: The slot type is set to 'AMAZON.AlphaNumeric'.
- Prompts**: A text input field containing the prompt 'Are you sure to confirm your order?'.

At the bottom of the dialog box are 'Cancel' and 'Add' buttons. The 'Add' button is highlighted in orange.

- Click on Advanced options
- Goto slot captures : Success response
- Click on add Conditional branching



- Under Branch1 , write condition for the branch1 and write Message we want to show if success. In my condition , if sure equals yes order confirmation message will be displayed.



- We can add branches according to usage .
- In my case I added second branch with condition if sure equals no then thanks for using bot

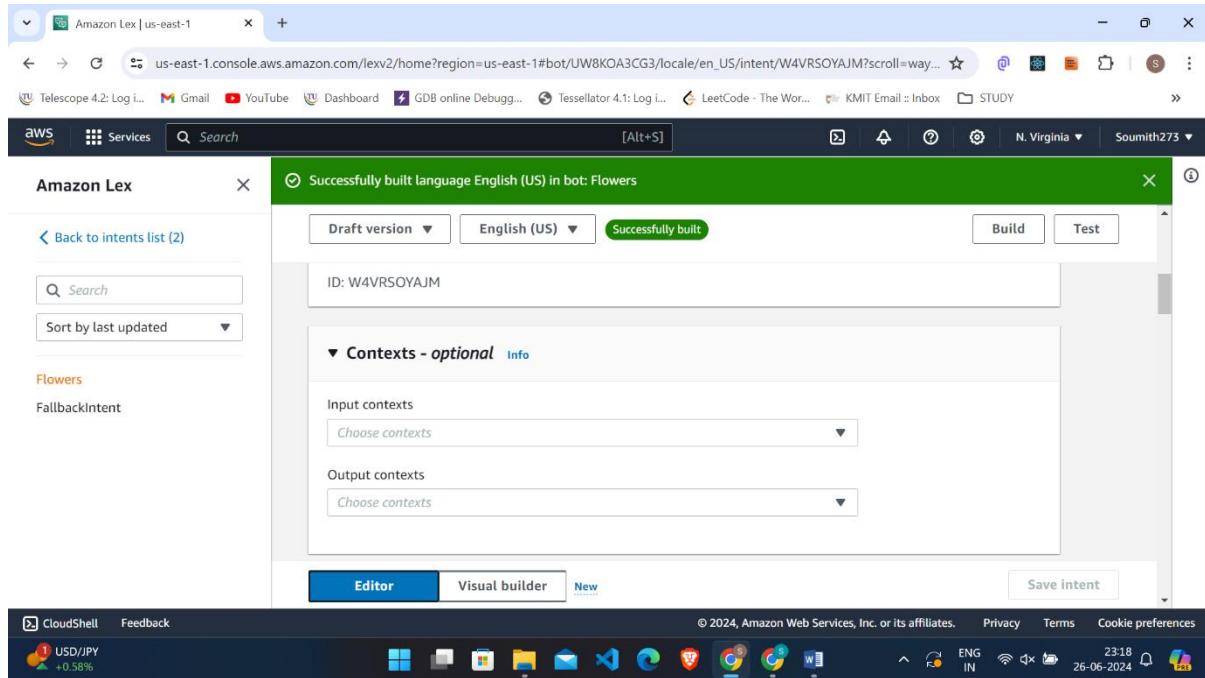
The screenshot shows the AWS Lambda console interface. A Lambda function named "FlowersBot" is selected. The function is triggered by an S3 event and uses the "aws-lambda-nodejs" runtime. The handler is "index.handler". The configuration includes environment variables: "BOT_NAME" set to "FlowersBot", "QUEUE_URL" set to "https://SQS.us-east-1.amazonaws.com/01234567891234567890/flowers-orders", and "QUEUE_NAME" set to "flowers-orders". The role "Lambda execution role" is attached.

Step 10: In same way , add more slots as many as we needed .

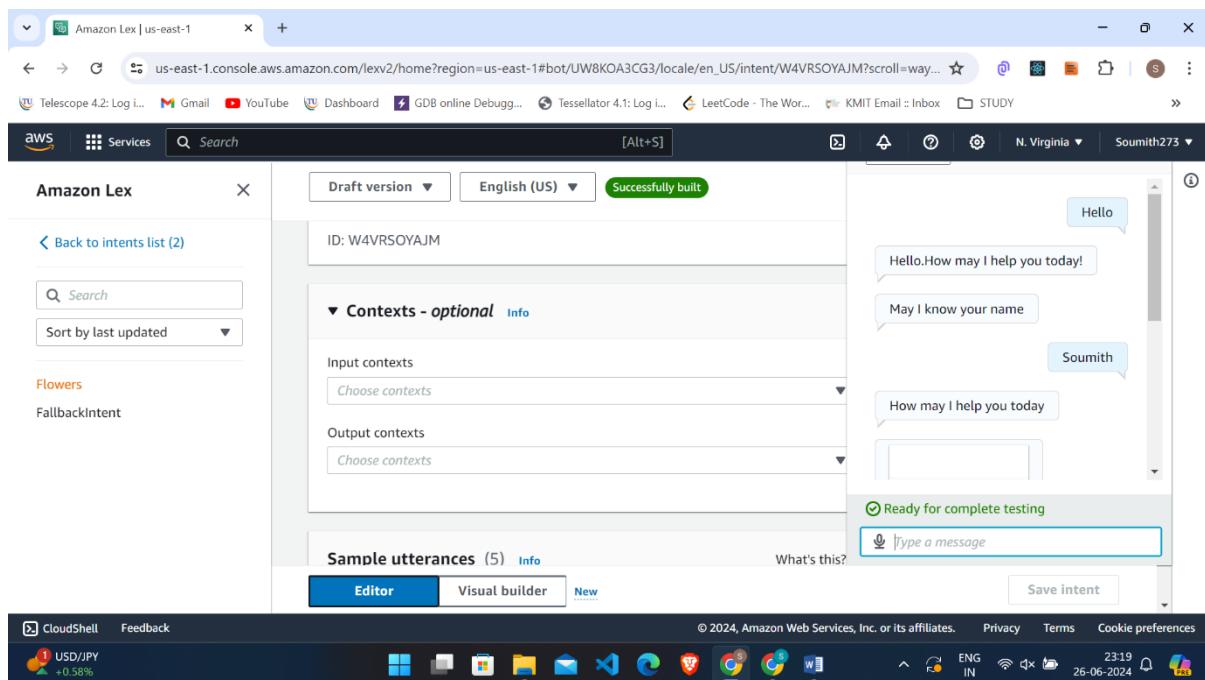
- After adding all slots , Click on Save Intent

The screenshot shows the AWS Lambda console interface. A Lambda function named "FlowersBot" is selected. The function is triggered by an S3 event and uses the "aws-lambda-nodejs" runtime. The handler is "index.handler". The configuration includes environment variables: "BOT_NAME" set to "FlowersBot", "QUEUE_URL" set to "https://SQS.us-east-1.amazonaws.com/01234567891234567890/flowers-orders", and "QUEUE_NAME" set to "flowers-orders". The role "Lambda execution role" is attached.

Step 11: Click on Build and wait until the bot is build .



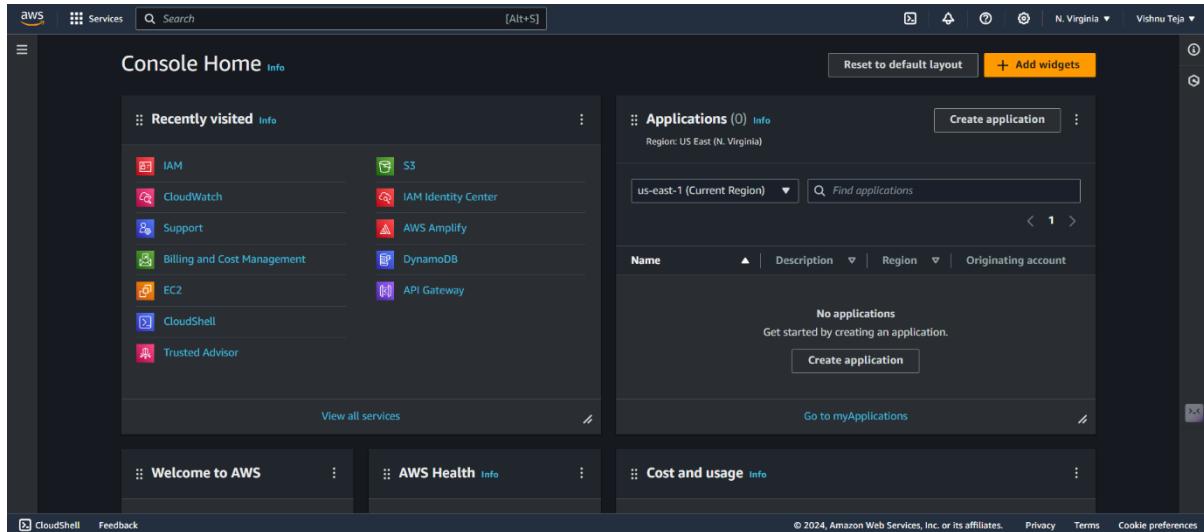
Step 12: Click on Test and check our bot.



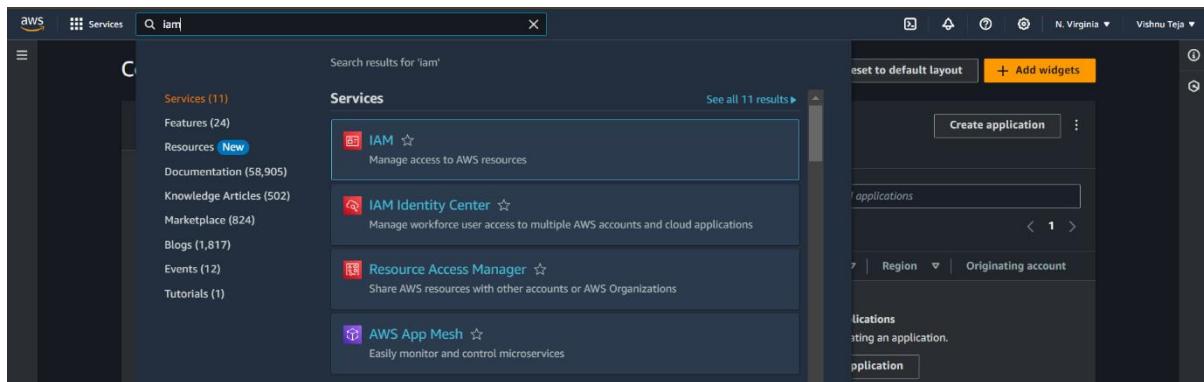
Experiment – 8

Aim: Create an AWS IAM User with attached policy and also Implement a role based access between AWS services as well as Create an group with attached policies

Step 1: Login to AWS Free Tier Account with the Login Credentials.



Step 2: Navigate to the Service IAM through the search bar.



Step 3: In the IAM Dashboard, Left pane Search for Users and Navigate to the user dashboard.

The screenshot shows the AWS IAM Dashboard. On the left, there's a sidebar with navigation links like Dashboard, Access management, Access reports, and CloudShell. The main area has sections for Security recommendations (with one notification about adding MFA to the root user) and IAM resources (listing 0 User groups, 0 Users, 3 Roles, 0 Policies, and 0 Identity providers). To the right, there's a panel for the AWS Account (Account ID: 533267052984, Account Alias: Create, Sign-in URL: https://533267052984.sigin.aws.amazon.com/console), and a Quick Links section for My security credentials.

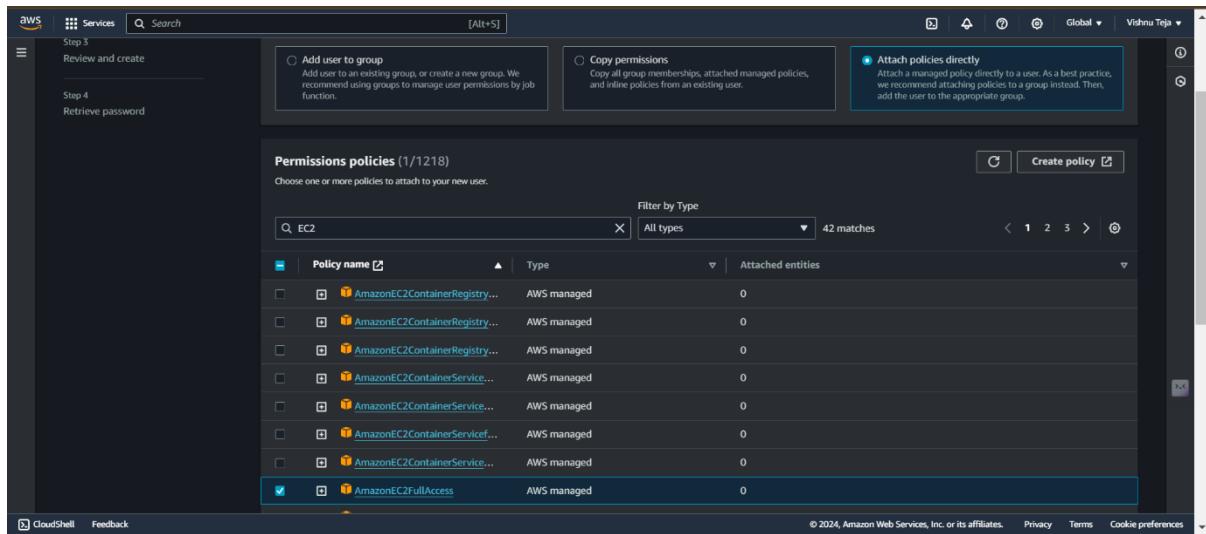
Step 4: To create a user with the IAM policy Click on Create User.

The screenshot shows the AWS IAM Users page. The sidebar includes links for Dashboard, Access management (Users selected), and CloudShell. The main content area shows a table titled "Users (0)" with a single row "No resources to display". There are buttons for Delete and Create user.

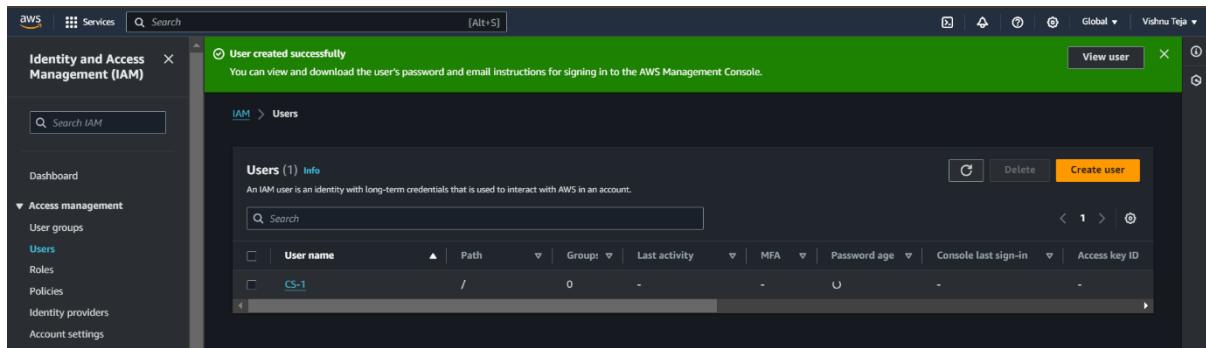
Step 5: Specify the User Name, Provide the access to Management console and create a Custom password and click on next.

The screenshot shows the "Specify user details" step of the IAM User creation wizard. It asks for a User name (entered as "Cs-1"), provides options for User type (selected "I want to create an IAM user"), and password settings (selected "Custom password" with "Km1t25S"). It also includes checkboxes for "Provide user access to the AWS Management Console - optional" and "If you are providing console access to a person, provide their AWS Identity Center ARN". A note at the bottom says "If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Kinesis, or a backing credential for emergency account access, users automatically get the AWSAccessKeyId and AWS密钥 (AWS Secret Access Key) policy to allow them to change their own password." There are "Cancel" and "Next Step" buttons at the bottom.

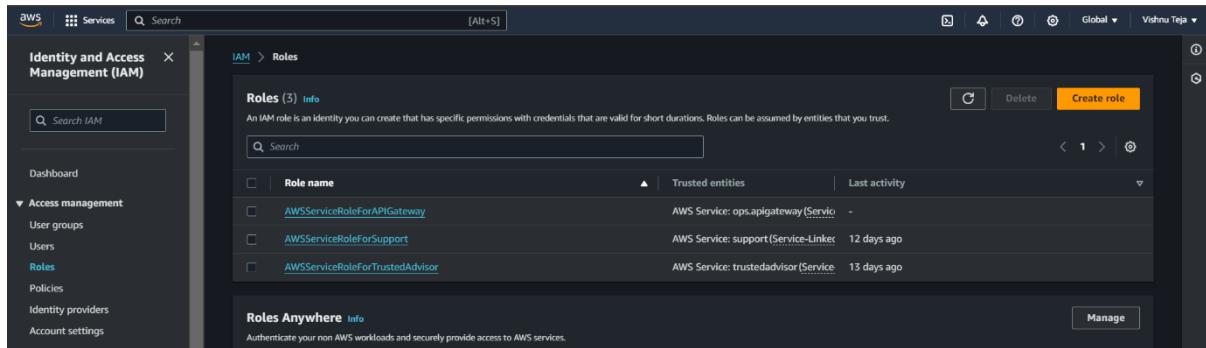
Step 6: Select Attach Policies directly, and search for EC2fullAccess policy and click on next.



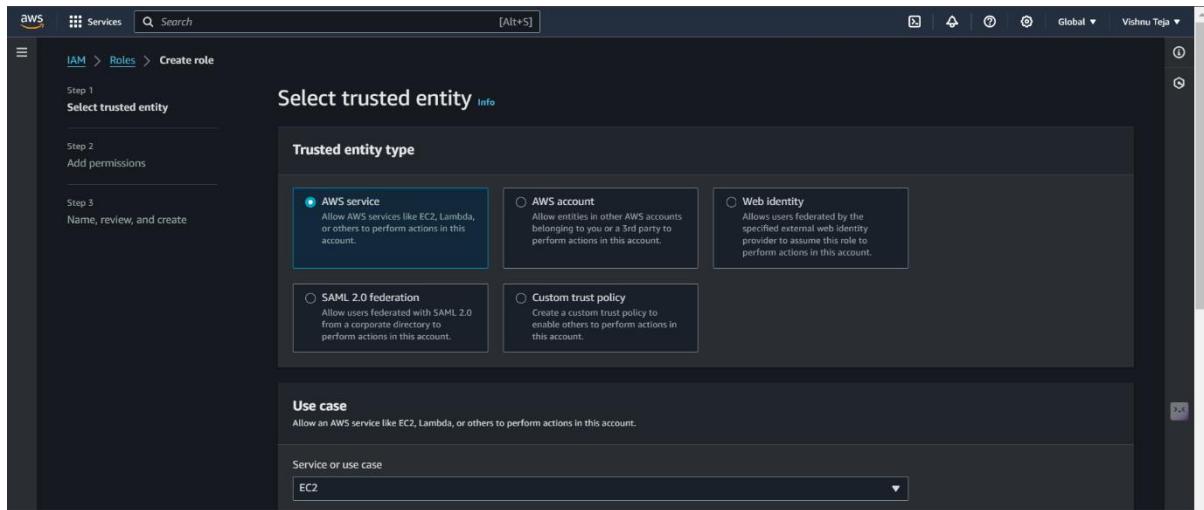
Step 7: Review the policy and create user. Now we have create a User with attach policy



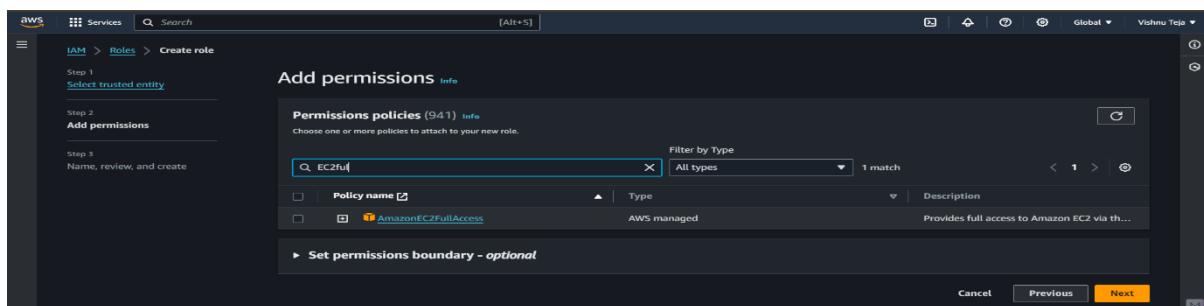
Step 8: Click on Roles from the left pane, and click create role.



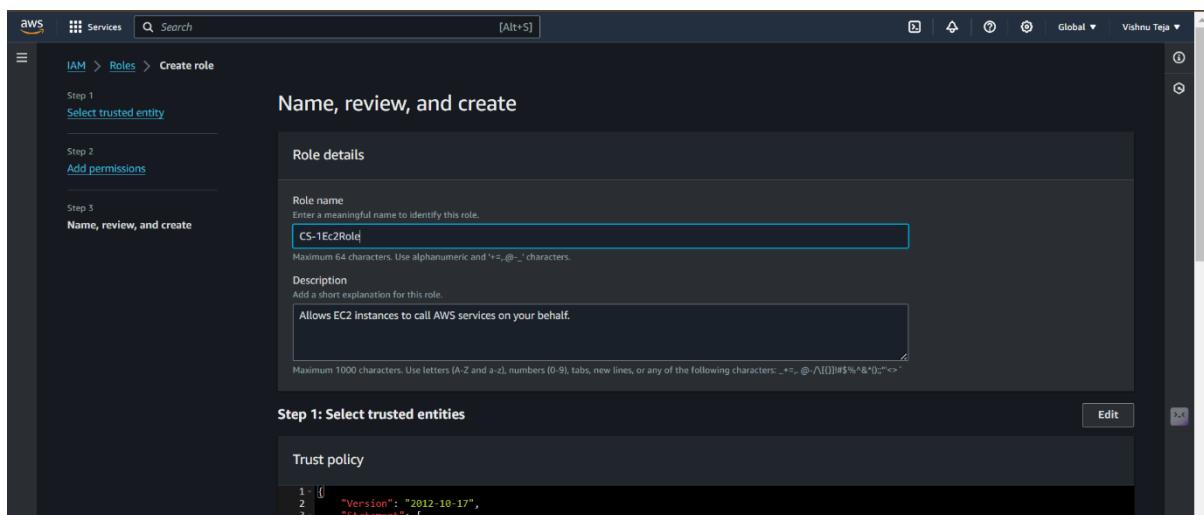
Step 9: Select AWS Service, From the list of use case, select EC2 and click on next.



Step 10: Search for the permission EC2FullAccess, Assign it and click on next.



Step 11: Give a Name to the role and review the policy and click on create role.



Step 12: Now successfully we have created a role based access between AWS service.

The screenshot shows the AWS IAM Roles page. At the top, a green banner says "Role CS-1Ec2Role created." Below it, the "Roles (4) Info" section is displayed. A table lists four roles: "AWSServiceRoleForAPIGateway" (AWS Service: ops.apigateway [Service]), "AWSServiceRoleForSupport" (AWS Service: support [Service-Linked]), "AWSServiceRoleForTrustedAdvisor" (AWS Service: trustedadvisor [Service]), and "CS-1Ec2Role" (AWS Service: ec2). The "Create role" button is visible at the top right.

Step 13: To create user group, navigate to the user group dashboard and select create group.

The screenshot shows the AWS IAM User groups page. At the top, a green banner says "User groups (0) Info". Below it, the "User groups (0) Info" section is displayed. A table shows no resources, with columns for "Group name", "Users", "Permissions", and "Creation time". The "Create group" button is visible at the top right.

Step 14: Give a name to the Group and select the users from the list to add to the group.

The screenshot shows the "Create user group" page. In the "Name the group" section, the "User group name" field contains "CS-1Group". In the "Add users to the group - Optional (1/1) Info" section, the "User name" dropdown shows "CS-1" with a checked checkbox. The "Create group" button is visible at the bottom right.

Step 15: Give necessary permissions to the group and click on create group. Now successfully we have created a user group and added a user to that group.

The screenshot shows the AWS IAM User groups page. At the top, a green banner says "CS-1Group user group created." Below it, the "User groups (1) Info" section is displayed. A table shows one group named "CS-1Group" with a status of "Defined" and a creation time of "Now". The "View group" button is visible at the top right.

Experiment-9

Aim: Launch a NoSQL database using Amazon DynamoDB.

Step 1: Navigate to Dynamo DB Service via AWS Management console and Click on Create Table.

The screenshot shows the AWS DynamoDB service page. On the left, there's a sidebar with options like Dashboard, Tables, Explore items, PartiQL editor, Backups, Exports to S3, Imports from S3, Integrations (New), Reserved capacity, and Settings. Below that is a section for DAX with Clusters, Subnet groups, Parameter groups, and Events. At the bottom of the sidebar are CloudShell and Feedback links. The main content area has a title 'Amazon DynamoDB' and subtitle 'A fast and flexible NoSQL database service for any scale'. It includes a brief description of DynamoDB as a fully managed, key-value, and document database. There are two call-to-action buttons: 'Create table' and 'Learn more about pricing'. A 'How it works' section features a video thumbnail with the text 'What is Amazon DynamoDB?' and a 'Copy link' button.

Step 2: Enter the name of the table and enter the partition key which is essentially the primary key of the table which is unique.

The screenshot shows the 'Create table' wizard. The top navigation bar includes the AWS logo, Services, a search bar, and account information. The main heading is 'Create table'. Under 'Table details', there's a note that DynamoDB is schemaless and requires a table name and primary key. The 'Table name' field is set to 'Demo'. The 'Partition key' section specifies 'id' as the primary key of type 'String'. The 'Sort key - optional' section has an empty 'Enter the sort key name' field. The bottom of the page shows the URL as 'https://us-east-1.console.aws.amazon.com/console/home?region=us-east-1' and standard copyright and footer links.

Step 3: Leave the other options as default and proceed to Click on Create Table .

The screenshot shows the 'Create Table' wizard in the AWS Management Console. The top section displays configuration details:

Capacity mode	Provisioned	Yes
Provisioned read capacity	5 RCU	Yes
Provisioned write capacity	5 WCU	Yes
Auto scaling	On	Yes
Local secondary indexes	-	No
Global secondary indexes	-	Yes
Encryption key management	Owned by Amazon DynamoDB	Yes
Deletion protection	Off	Yes
Resource-based policy	Not active	Yes

The 'Tags' section indicates no tags are associated with the resource and provides an 'Add new tag' button.

At the bottom right, there are 'Cancel' and 'Create table' buttons.

Step 4: Once the table is created and running select the created table from the dashboard and click on Explore items .

The screenshot shows the 'Explore items' interface for the 'Demo' table. On the left, a sidebar lists tables: Demo (selected), employees, something, and Student. The main area displays the 'Scan or query items' interface:

- Scan or query items**: Scan (selected) and Query buttons.
- Select a table or index**: Table - Demo.
- Select attribute projection**: All attributes.
- Filters**: Buttons for Run and Reset.
- Completed message**: Completed. Read capacity units consumed: 0.5.
- Items returned (0)**: A table showing No items and No items to display. Buttons for Actions, Create item, and a navigation bar.

We can see at present there are no items in the table.

Step 5: To create items in the table click on Create item and enter some value for the partition key.

The screenshot shows the AWS DynamoDB 'Create item' interface. At the top, there are tabs for 'Form' (selected) and 'JSON view'. Below the tabs is a section titled 'Attributes' with a table. The table has columns for 'Attribute name', 'Value', and 'Type'. A single row is present, labeled 'id - Partition key' with the value '1' and type 'String'. At the bottom right of the form are 'Cancel' and 'Create item' buttons. The background shows the AWS navigation bar and a sidebar.

We can also add other fields/Attributes in the table .

Step 6: To add Additional Attributes in the table click on Add new attribute drop-box and select the data type of the new attribute.

The screenshot shows the AWS DynamoDB 'Create item' interface. The 'Add new attribute' dropdown menu is open, displaying various data types: String, Number, Boolean, Binary, Null, String set, Number set, Binary set, List, and Map. The 'Number' option is currently selected. The rest of the interface is similar to the previous screenshot, showing the 'Create item' form with one attribute entry.

Step 7: Proceed to Enter the name of the Attribute and value of the attribute and click on Create Item.

The screenshot shows the 'Create item' interface in the AWS DynamoDB console. It displays two attributes: 'id - Partition key' with value '1' and type 'String', and 'cost' with value '1000' and type 'Number'. The 'Create item' button at the bottom right is highlighted with a yellow box.

We can see that value is added to the table .

The screenshot shows the 'Explore items' table in the AWS DynamoDB console. It lists one item with the primary key 'id' having the value '1' and the attribute 'cost' having the value '1000'.

Items returned (1)		
	id (String)	cost
1	1	1000

Step 8: To update the item select the item and go to actions and select edit item .

The screenshot shows the 'Edit item' interface in the AWS DynamoDB console. It displays the same item with 'id' as '1' and 'cost' as '1000'. A context menu is open on the item, with the 'Edit item' option highlighted with a yellow box.

Step 9: Update the value and click on save and close.

The screenshot shows the 'Edit item' interface in the AWS DynamoDB console. The 'cost' attribute has been updated from '1000' to '999'. The 'Save and close' button at the bottom right is highlighted with a yellow box.

Step 10: To display the items navigate to the scan or query items panel and select all attributes from select attribute projection and click on run.

▼ Scan or query items

Scan Query

Select a table or index
Table - Demo

Select attribute projection
All attributes

▶ Filters

Run Reset

Step 11: Upon successful query we can see all the values of the table .

Completed. Read capacity units consumed: 0.5

Items returned (1)

id (String) | cost

1 | 999

Actions ▾ **Create item**

< 1 > ⌂ ⌂

Step 12: To Delete an item , select the item ,click on Actions Drop-down box and click on Delete Items.

Completed. Read capacity units consumed: 0.5

Items returned (1/1)

id (String) | cost

1 | 999

Actions ▾ **Create item**

< 1 > ⌂ ⌂

Step 13: Click on Run from the Scan or query Items panel and click on Run to see the updates in the table.

Completed. Read capacity units consumed: 0.5

Items returned (0)

No items

No items to display.

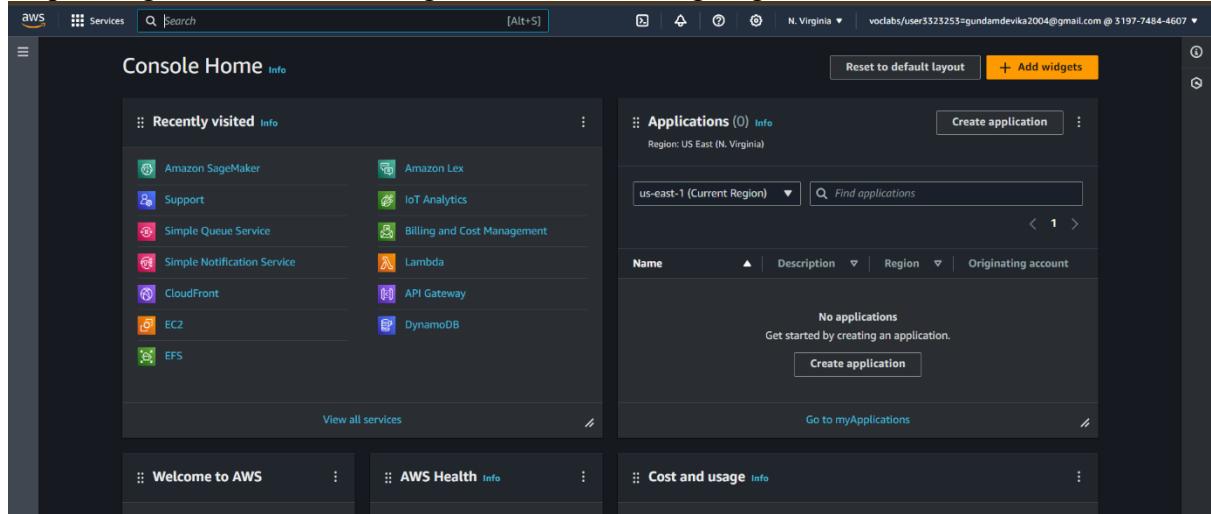
Create item

We can see that the table is empty and there are no values in the table.

Experiment-10

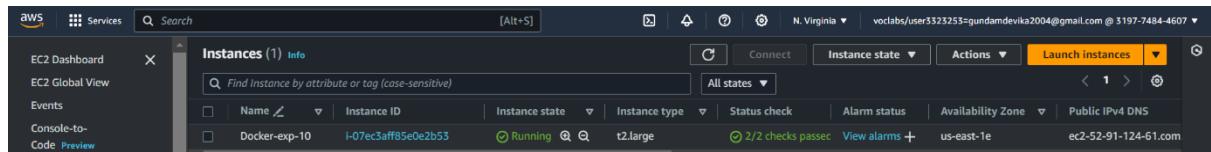
Aim: Migrate a website from local server to Cloud using Docker.

Step 1: Login to the AWS Management Console using Login Credentials



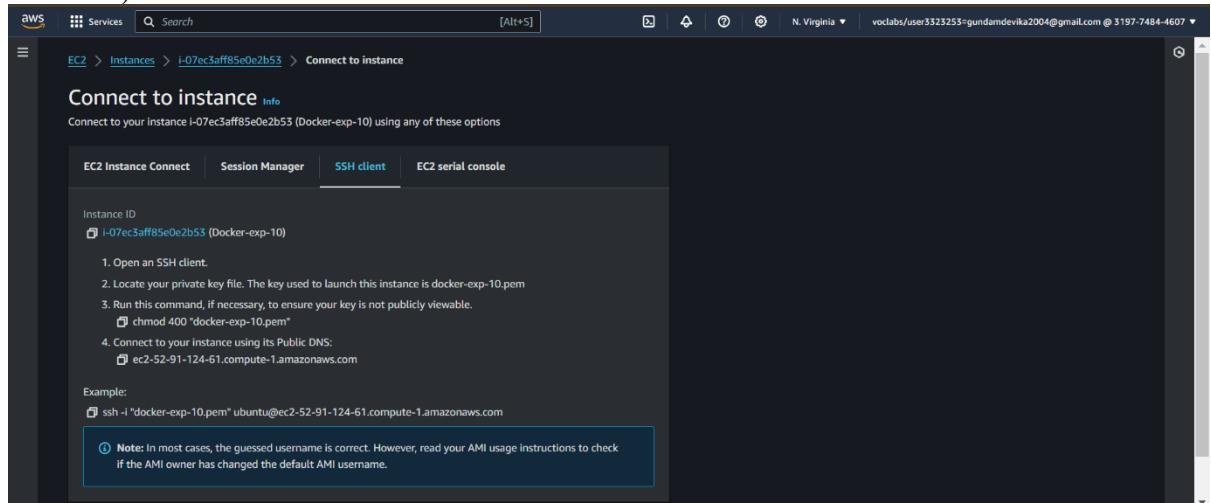
The screenshot shows the AWS Management Console home page. On the left, there's a sidebar with 'Recently visited' services like Amazon SageMaker, Support, Simple Queue Service, Simple Notification Service, CloudFront, EC2, and EFS. Below that is a 'Welcome to AWS' section. On the right, the main area is titled 'Applications (0)' with a 'Create application' button. It shows the region as 'US East (N. Virginia)'. A search bar at the top right says 'Find applications'. Below the search bar, there's a table header for 'Name', 'Description', 'Region', and 'Originating account'. A message below the table says 'No applications. Get started by creating an application.' with a 'Create application' button. At the bottom right, there's a 'Go to myApplications' link.

Step 2: Navigate to the EC2 Service by using Search bar and Launch an EC2 instance with required storage (30GB) and type (t2.large) and Launch the instance



The screenshot shows the AWS EC2 Instances page. The left sidebar has links for 'EC2 Dashboard', 'EC2 Global View', 'Events', and 'Console-to-Code Preview'. The main area shows a table for 'Instances (1)'. The first row shows an instance named 'Docker-exp-10' with Instance ID 'i-07ec3aff85e0e2b53', status 'Running', type 't2.large', and 2/2 checks passed. It's located in 'us-east-1e' with Public IPv4 DNS 'ec2-52-91-124-61.com'. There are buttons for 'Connect', 'Actions', and 'Launch instances'.

Step 3: Connect to the instance through SSH Client (or through any other way to connect to instance)



The screenshot shows the 'Connect to instance' page for the instance 'i-07ec3aff85e0e2b53'. The top navigation shows 'EC2 > Instances > i-07ec3aff85e0e2b53 > Connect to instance'. The main content area has tabs for 'EC2 Instance Connect', 'Session Manager', 'SSH client' (which is selected), and 'EC2 serial console'. Under 'SSH client', it says 'Instance ID: i-07ec3aff85e0e2b53 (Docker-exp-10)'. It provides instructions: 1. Open an SSH client, 2. Locate your private key file (key used is docker-exp-10.pem), 3. Run this command if necessary to ensure your key is not publicly viewable (chmod 400 docker-exp-10.pem), and 4. Connect to your instance using its Public DNS (ec2-52-91-124-61.compute-1.amazonaws.com). An example command is shown: ssh -i "docker-exp-10.pem" ubuntu@ec2-52-91-124-61.compute-1.amazonaws.com. A note at the bottom says: 'Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.'

Step 4: Login to the GitHub and get the URL's for the FrontEnd and BackEnd repositories and clone them into the instance connected, such as \$git clone <https://url-to-frontend.git>

```
ubuntu@ip-172-31-55-159: ~
ubuntu@ip-172-31-55-159:~$ git clone https://github.com/procareer3fwd/realgrandebackend.git
```

Step 5: Perform same step for the backend repository also. Now we can check the two repositories have cloned to our local server. \$ls

```
ubuntu@ip-172-31-55-159: ~
ubuntu@ip-172-31-55-159:~$ ls
realgrandebackend  realgrandefrontend
ubuntu@ip-172-31-55-159:~$ |
```

Step 6: Perform the update operation on Instance. \$sudo apt update, all the required packages and updates will be done to the instance.

```
ubuntu@ip-172-31-55-159: ~
ubuntu@ip-172-31-55-159:~$ ls
realgrandebackend  realgrandefrontend
ubuntu@ip-172-31-55-159:~$ sudo apt update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [185 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [51.1 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [166 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [43.6 kB]
Fetched 698 kB in 1s (822 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
35 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@ip-172-31-55-159:~$ |
```

Step 7: After updating the instance, now to create the images of our front end and back end we need to install the docker on the local machine. \$sudo apt -y install docker.io

```
ubuntu@ip-172-31-55-159:~$ sudo apt -y install docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
docker.io is already the newest version (24.0.7-0ubuntu4).
0 upgraded, 0 newly installed, 0 to remove and 35 not upgraded.
ubuntu@ip-172-31-55-159:~$ |
```

Step 8: Now Navigate to the realgrandebackend folder and search for .env file whether it exists in the folder or not, if not we need to create the .env file

```
ubuntu@ip-172-31-55-159: ~/realgrandebackend
ubuntu@ip-172-31-55-159:~/realgrandebackend$ ls
Dockerfile  models  package-lock.json  package.json  routes  server.js
ubuntu@ip-172-31-55-159:~/realgrandebackend$ |
```

Step 9: To create .env file, run the command \$nano .env and specify the database credentials in this file such as mongoburl, dbusername, dbpassword, in frontenduri specify the public ip of our instance

```
ubuntu@ip-172-31-55-159: ~/realgrandebackend
GNU nano 7.2
.env
MONGOBURL="mongodb+srv://fsd04.2hxrda.mongodb.net/realgrande?retryWrites=true&w=majority"
DBUSERNAME=procareer3
DBPASSWORD=ISobjBDohsFqEAqq
FRONTENDURI="http://34.229.184.54"
```

Step 10: Now we need to build the docker image for the backend, run the command to build the docker image. \$sudo docker build -t backend_server . and check the docker image \$sudo docker ps

```
ubuntu@ip-172-31-55-159:~/realgrandebackend$ sudo docker build -t backend_server .
DEPRECATION: The legacy builder is deprecated and will be removed in a future release.
              Install the buildx component to build images with BuildKit:
              https://docs.docker.com/go/buildx/
Sending build context to Docker daemon 179.2kB
Step 1/6 : FROM node
--> 3d4b037e6712
Step 2/6 : WORKDIR /app
--> Using cache
--> ac262bc309f2
Step 3/6 : COPY . /app
--> Using cache
--> f49f99875b21
Step 4/6 : RUN npm install
--> Using cache
--> cd5ebdecfc7a
Step 5/6 : EXPOSE 5000
--> Using cache
--> 88583bf19491
Step 6/6 : CMD ["npm", "start"]
--> Using cache
--> b057381e8a17
Successfully built b057381e8a17
Successfully tagged backend_server:latest
ubuntu@ip-172-31-55-159:~/realgrandebackend$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS          NAMES
ubuntu@ip-172-31-55-159:~/realgrandebackend$ |
```

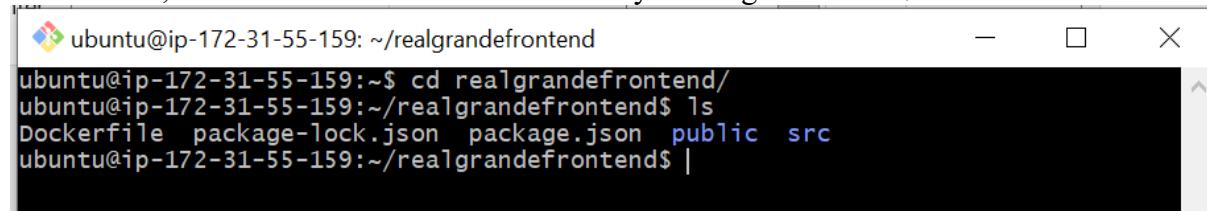
Step 11: Now run the image created for the backend \$sudo docker run -d -p 2001:5000 backend_server

```
ubuntu@ip-172-31-55-159:~/realgrandebackend$ sudo docker run -d -p 2001:5000 backend_server
d1548fd58799eeb01b512d773784179e7a5367a3d239c9c77f78de42f753a708
ubuntu@ip-172-31-55-159:~/realgrandebackend$ |
```

Step 12: Now try to access the data in the browser by <Ec2_ip_address>:2001/api, we need to get the json data

```
[  
  {  
    "name": "Adeel Solangi",  
    "language": "Sindhi",  
    "id": "V590F92YF627HFY0",  
    "bio": "Donec lobortis eleifend condimentum. Cras dictum dolor lacinia lectus vehicula rutrum. Maecenas quis nisi nunc. Nam tristique feugiat est vitae mollis. Maecenas quis nisi nunc.",  
    "version": 6.1  
  },  
  {  
    "name": "Afzal Ghaffar",  
    "language": "Sindhi",  
    "id": "ENTOCR13RSCLZ6KU",  
    "bio": "Aliquam sollicitudin ante ligula, eget malesuada nibh efficitur et. Pellentesque massa sem, scelerisque sit amet odio id, cursus tempor urna. Etiam congue dignissim volutpat. Vestibulum pharetra libero et velit gravida euismod.",  
    "version": 1.88  
},
```

Step 13: Perform the above steps for frontend, navigate to the frontend folder and search for the .env file, if not found create a new .env file by running command \$nano .env



```
ubuntu@ip-172-31-55-159: ~/realgrandefrontend  
ubuntu@ip-172-31-55-159:~$ cd realgrandefrontend/  
ubuntu@ip-172-31-55-159:~/realgrandefrontend$ ls  
Dockerfile package-lock.json package.json public src  
ubuntu@ip-172-31-55-159:~/realgrandefrontend$ |
```

Step 14: Replace our public ip address in the .env file

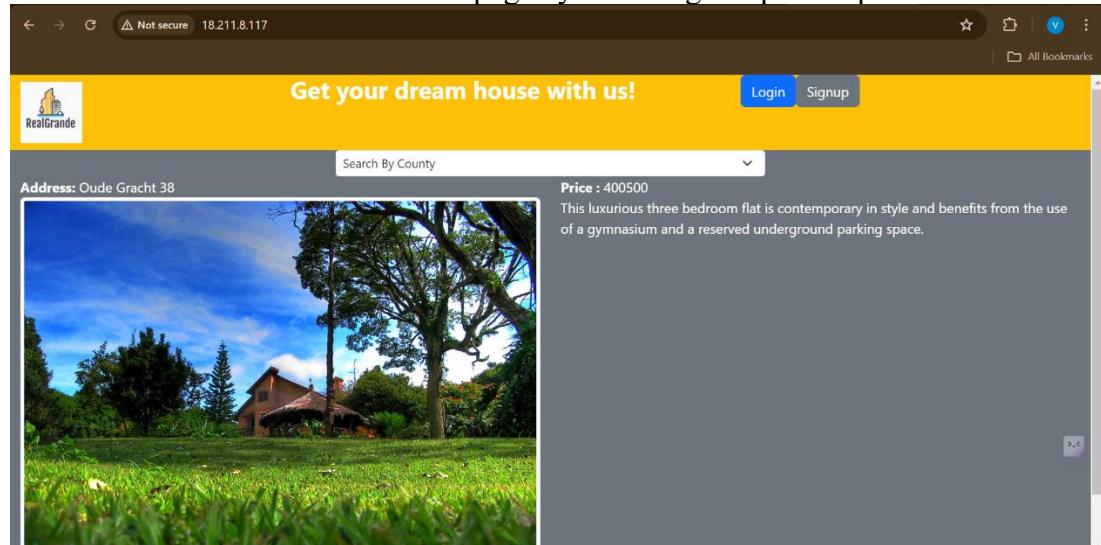


```
ubuntu@ip-172-31-55-159: ~/realgrandefrontend  
GNU nano 7.2  
REACT_APP_BACKEND_URL="http://18.211.8.117:2001/api"
```

Step 15: build the docker image for the front end and run the image

```
Successfully built 91a03b10b17c  
Successfully tagged frontend:latest  
ubuntu@ip-172-31-55-159:~/realgrandefrontend$ sudo docker run -d -p 80:3000 front  
end  
3dcec6d7b9f6cccd3a56b513078ad5b3462785ef64a96deea9f510f3994e4ebf6  
ubuntu@ip-172-31-55-159:~/realgrandefrontend$ |
```

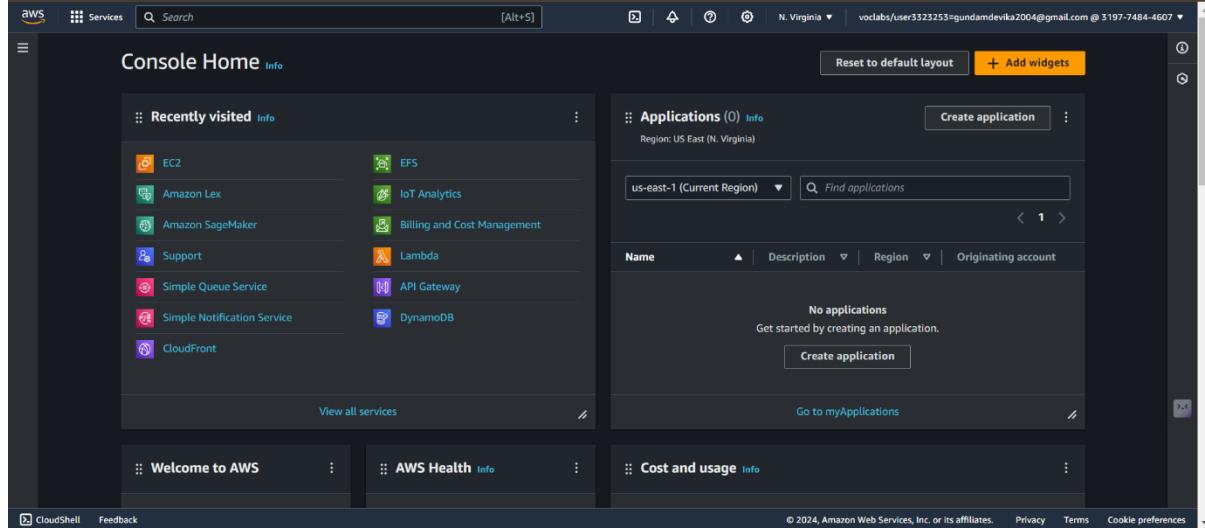
Step 16: Now successfully we created the frontend and backend images by using docker containers and Access the front end page by browsing our public ip address of our instance.



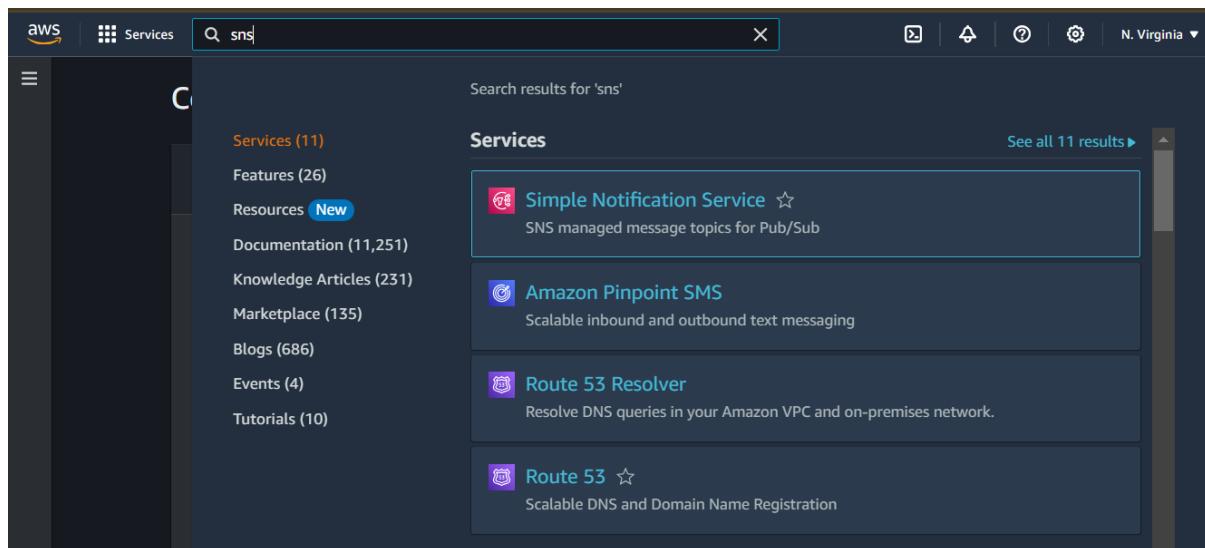
Experiment-11

Aim: Create loosely coupled services with Amazon SQS and Amazon SNS to process data received from the applications.

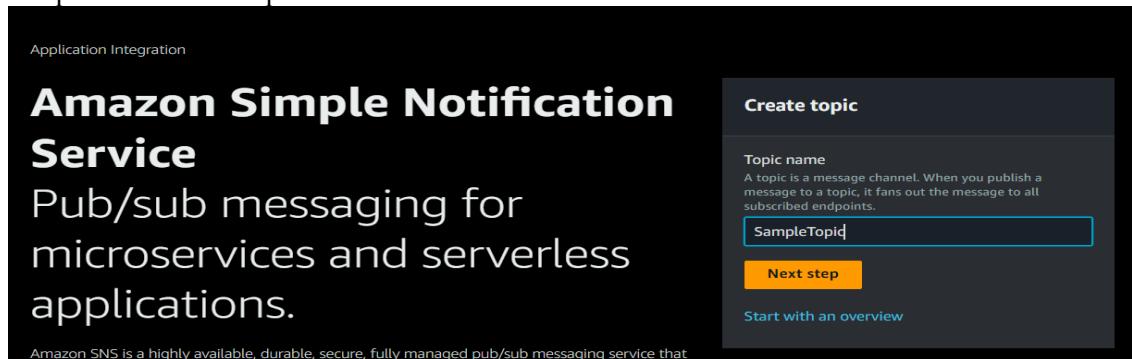
Step1: Login to the AWS Management Console with login credentials



Step 2: Navigate to the search bar and search for the service SNS(Simple Notification Service)



Step 3: After Selecting the SNS service from the search bar, navigate to the service dashboard, give a sample name to the topic and click next



Step 4: Select the topic type to Standard and in the topic name and give some description if needed.

The screenshot shows the 'Create topic' page in the AWS SNS console. In the 'Type' section, the 'Standard' option is selected, indicated by a blue outline around the radio button. The 'FIFO (first-in, first-out)' option is shown with its characteristics: Strictly-preserved message ordering, Exactly-once message delivery, High throughput, up to 300 publishes/second, and Subscription protocols: SQS, Lambda, HTTP, SMS, email, mobile application endpoints. Below the type selection, the 'Name' field contains 'SampleTopic fifo'. The 'Display name - optional' field contains 'Sample topic sending email to subscribers'. Both fields have their respective character limits displayed below them: 256 characters for the name and 100 characters for the display name.

Step 5: Then Leave all the optional settings to default and scroll down to the bottom of the page and click on create topic. After successfully creating the topic then click on create subscription.

The screenshot shows the 'SampleTopic' details page in the AWS SNS console. A green banner at the top indicates that the topic was created successfully. The 'Details' section displays the topic's configuration: Name (SampleTopic), Display name (Sample topic sending email to subscribers), ARN (arn:aws:sns:us-east-1:319774844607:SampleTopic), and Type (Standard). Below the details, there are tabs for Subscriptions, Access policy, Data protection policy, Delivery policy (HTTP/S), Delivery status logging, Encryption, Tags, and Integrations. The 'Subscriptions' tab is active, showing '(0)' and a 'Create subscription' button. At the bottom of the page, there are links for CloudShell, Feedback, and a footer with copyright information and links to Privacy, Terms, and Cookie preferences.

Step 6: Select the protocol as email and specify the Endpoint i.e. the subscriber email ID to which we want to send the email and click on create subscription.

Details

Topic ARN
arn:aws:sns:us-east-1:319774844607:SampleTopic

Protocol
The type of endpoint to subscribe
Email

Endpoint
An email address that can receive notifications from Amazon SNS.
vishnuteja.gundam@gmail.com

ⓘ After your subscription is created, you must confirm it. [Info](#)

► Subscription filter policy - *optional* [Info](#)
This policy filters the messages that a subscriber receives.

► Redrive policy (dead-letter queue) - *optional* [Info](#)
Send undeliverable messages to a dead-letter queue.

Create subscription

Step 7: After successfully creation of subscription, verify the email send to the subscriber mail id, by clicking confirm subscription

AWS Notification - Subscription Confirmation [Inbox](#) ×

Sample topic sending email to subscribers <no-reply@sns.amazonaws.com>
to me ▾

You have chosen to subscribe to the topic:
arn:aws:sns:us-east-1:319774844607:SampleTopic

To confirm this subscription, click or visit the link below (If this was in error no action is necessary):
[Confirm subscription](#)

Please do not reply directly to this email. If you wish to remove yourself from receiving all future SNS subscription confirmation requests please send an email to [sns-opt-out](#)

Step 8: After successfully Confirming the Subscription, then click on publish message.

Amazon SNS > Topics > SampleTopic

SampleTopic

Details

Name: SampleTopic
ARN: arn:aws:sns:us-east-1:319774844607:SampleTopic
Type: Standard

Subscriptions (1)

ID	Endpoint	Status	Protocol
a8ed8378-81c6-4048-bf11-5cafc2cd2ba0	vishnuteja.gundam@gmail.com	Confirmed	EMAIL

Edit Delete Request confirmation Confirm subscription Create subscription

Step 9: Enter the subject and the body of the mail to want to send to the subscriber.

Amazon SNS > Topics > SampleTopic > Publish message

Publish message to topic

Message details

Topic ARN
arn:aws:sns:us-east-1:319774844607:SampleTopic

Subject - optional
Good Morning Welcom to AWS SNS

Maximum 100 printable ASCII characters

Time to Live (TTL) - optional | Info
This setting applies only to mobile application endpoints. The number of seconds that the push notification service has to deliver the message to the endpoint.

Message body

Message structure

Identical payload for all delivery protocols.
The same payload is sent to endpoints subscribed to the topic, regardless of their delivery protocol.

Custom payload for each delivery protocol.
Different payloads are sent to endpoints subscribed to the topic, based on their delivery protocol.

Message body to send to the endpoint

```
1 Good Morning Everyone,
2 here you learnt a sample exercise how to work with the SNS service.
3 Thank You.
4 |
```

Step 10: Click on Publish Message, Now check the subscriber email and find the mail send by the SNS service.

Good Morning Welcom to AWS SNS [Inbox](#) [Reply](#) [Forward](#)

Sample topic sending email to subscribers <no-reply@sns.amazonaws.com>
to me [View in browser](#)

10:34 PM (0 minutes ago) [Star](#) [Unstar](#) [Reply](#) [Forward](#) [More](#)

[Translate to Bulgarian](#) [X](#)

Good Morning Everyone,
here you learnt a sample exercise how to work with the SNS service.
Thank You.

If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
<https://sns.us-east-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-east-1:319774844607:SampleTopic:a8ed8378-81c6-4048-b5f1-5caf52cd28a0&Endpoint=vishnuteja.gundam@gmail.com>

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at <https://aws.amazon.com/support>

Step 11: Navigate to the SQS service through the search bar. Click on create Queue.

aws Services Search [Alt+S] N. Virginia voclabs/user5325255=gundamdevika2004@gmail.com @ 3197-7484-4607 ▾

Amazon SQS > Queues

Queues (1)

Search queues by prefix

Name	Type	Created	Messages available	Messages in flight	Encryption	Content-based deduplication
MyQueue_fifo	FIFO	2024-06-21T11:56+05:30	0	0	Amazon SQS key (SSE-SQS)	Enabled

Step 12: Name the Queue with .fifo extension and enable the content-based duplication. And click on create queue.

The screenshot shows the AWS SQS Queues page. At the top, there is a search bar and navigation links for Services and N. Virginia. Below the header, a table lists two queues:

Name	Type	Created	Messages available	Messages in flight	Encryption	Content-based deduplication
MyQueue.fifo	FIFO	2024-06-21T11:56+05:30	0	0	Amazon SQS key (SSE-SQS)	Enabled
MyQueue1.fifo	FIFO	2024-06-30T21:48+05:30	0	0	Disabled	Enabled

Step 13: click on your queue and and click on send and receive messages in the right side top .

The screenshot shows the details page for the MyQueue.fifo queue. The top navigation bar includes links for Services, N. Virginia, and the specific queue path Amazon SQS > Queues > MyQueue.fifo. Below the navigation, there are buttons for Edit, Delete, Purge, Send and receive messages, and Start DLQ redrive. The main section displays the queue's configuration:

Name	Type	ARN
MyQueue.fifo	FIFO	arn:aws:sqs:us-east-1:319774844607:MyQueue.fifo

Details include:

- Name: MyQueue.fifo
- Type: FIFO
- Encryption: Amazon SQS key (SSE-SQS)
- URL: https://sqs.us-east-1.amazonaws.com/319774844607/MyQueue.fifo
- Dead-letter queue: -

Step 14: Enter the message body, group ID, and click on send message.

The screenshot shows the Send and receive messages page for the MyQueue.fifo queue. The top navigation bar includes links for Services, N. Virginia, and the specific queue path Amazon SQS > Queues > MyQueue.fifo > Send and receive messages. The main section has a button for Send message. A success message is displayed: "Your message has been sent and is ready to be received." Below this, there is a message body input field containing "Hi, this message is from Myqueue." and a message group ID input field containing "1234".

Step 15: Scroll down, to the receive message pane, their click for poll messages and select the message sent from the queue.

The screenshot shows the AWS Lambda console interface. At the top, there are fields for 'Message group ID' (1234) and 'Message deduplication ID - Optional' (Enter message deduplication id). Below this, the 'Receive messages' section is visible, showing 1 message available, a polling duration of 30, a maximum message count of 10, and a 50% completion poll progress. The message details table lists one message with ID f0115a35-da7b-4be7-9536-9aa871a5a21a, sent on 2024-06-30T21:51+05:30, which is 34 bytes in size and has a receive count of 1.

Step 16: The message from the queue sent will be displayed here.

The screenshot shows a modal dialog box titled 'Message: f0115a35-da7b-4be7-9536-9aa871a5a21a'. The 'Body' tab is selected, showing the message content: 'Hi, this message is from Myqueue.'. A yellow 'Done' button is located at the bottom left of the dialog.