Ex:1 **Introduction to various cloud platforms**
Date:

Aim:

To demonstrate the Introduction about various cloud platforms.

Procedure/Types of Cloud Platforms:

In today's digital era, businesses are increasingly turning to cloud computing platforms to enhance scalability, flexibility, and efficiency. The leading providers are Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP). Each offers a vast array of services tailored to different business needs.

1. Amazon Web Services (AWS)

- AWS is the most popular cloud computing platform by Amazon, with solutions such as storage, analytics, and machine learning.
- It fits businesses of all sizes.
- For example, during the Black Friday sale, an e-commerce company can scale the servers on AWS and run with it.

As the pioneer and market leader in cloud computing, AWS boasts a comprehensive suite of services catering to virtually every aspect of IT infrastructure. From compute and storage to machine learning and analytics, AWS offers over 200 fully featured services.

Key Features:

Elastic Compute Cloud (EC2): Provides resizable compute capacity in the cloud, allowing users to quickly scale up or down based on demand.

Simple Storage Service (S3): Offers scalable object storage for data backup, archiving, and analytics.

Lambda: A serverless computing service that allows developers to run code without provisioning or managing servers.

Advantages:

Extensive service offerings catering to diverse business needs.

Global presence with data centers located in multiple regions worldwide.

Strong community support and extensive documentation.

Use Cases:

Startups and enterprises seeking scalable infrastructure for web hosting and application development.

Data-intensive workloads such as big data processing, analytics, and machine learning.

2. Microsoft Azure

- Azure is a cloud computing platform by Microsoft that was originally launched in February 2010 under the name Windows Azure, only to be rebranded as Microsoft Azure in 2014.
- With its rebirth, it has developed into this varied cloud platform providing suite services, including computing, analytics, storage, and networking.
- It integrates well with Microsoft tools, such as Office 365 and Dynamics.
- It provides enterprise-grade security and advanced AI capabilities.

For example, an e-commerce business harnesses the power of Microsft Azure cloud software to scale its servers up or down during peak traffic times like the annual Black Friday sale. In fact, Zoom uses cloud computing by Azure for seamless video conferencing experiences.

Azure, Microsoft's cloud computing platform, is a close competitor to AWS, offering a wide range of services for building, deploying, and managing applications through Microsoft-managed data centers.

Key Features:

Virtual Machines (VMs): Enables users to deploy and manage virtual machines running various operating systems. Azure Blob Storage: Scalable, secure object storage for unstructured data.

Azure Functions: Serverless compute service for event-driven applications.

Advantages:
Seamless integration with Microsoft products and services, including Windows Server and Office 365.
Hybrid cloud capabilities, allowing businesses to integrate on-premises infrastructure with cloud services.
Strong emphasis on security and compliance, including certifications like ISO and SOC.
Use Cases:
Enterprises leveraging Microsoft's ecosystem for application development, collaboration, and productivity tools.
Organizations with a hybrid cloud strategy seeking seamless integration between on-premises and cloud environments.

3. Google Cloud

- Google Cloud stands out with its data analytics and artificial intelligence capabilities.
- It is the best for companies whose focus is on big data and machine learning projects.
- For example, a healthcare company can utilize the Google Cloud to handle extensive patient datasets and run predictive analytics.

Google Cloud Platform, powered by Google's infrastructure, offers a wide range of cloud services, including computing, storage, databases, machine learning, and more.
Key Features:
Compute Engine: Virtual machines on Google's infrastructure, providing flexibility and scalability.
Cloud Storage: Object storage with high availability and global edge-caching capabilities.
BigQuery: Fully managed, serverless data warehouse for analytics at scale.
Advantages:
Cutting-edge technologies in areas such as machine learning, artificial intelligence, and data analytics.
Global network infrastructure with high-performance, low-latency connections.
Emphasis on sustainability with a commitment to carbon neutrality.
Use Cases:
Organizations leveraging Google's expertise in data analytics and machine learning for predictive analysis and business insights.
Startups and developers attracted to GCP's innovative services and pay-as-you-go pricing model.
In conclusion, AWS, Azure, and Google Cloud Platform each offer unique strengths and advantages, catering to a wide range of business requirements and preferences. When selecting a cloud computing platform, it's essential to consider factors such as service offerings, pricing, performance, and integration capabilities. By understanding the features and use cases of these platforms, businesses can make informed decisions to drive innovation, agility, and growth in the digital age.

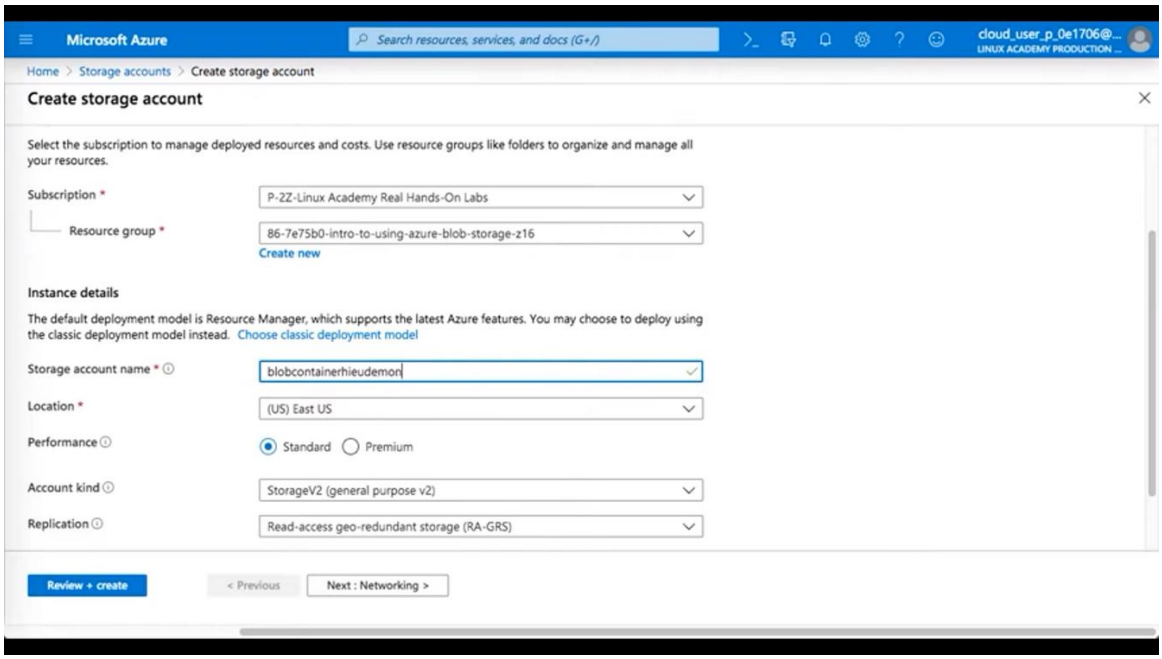
RESULT:

Ex:2 **Create a storage account and a hosted service component**
Date:

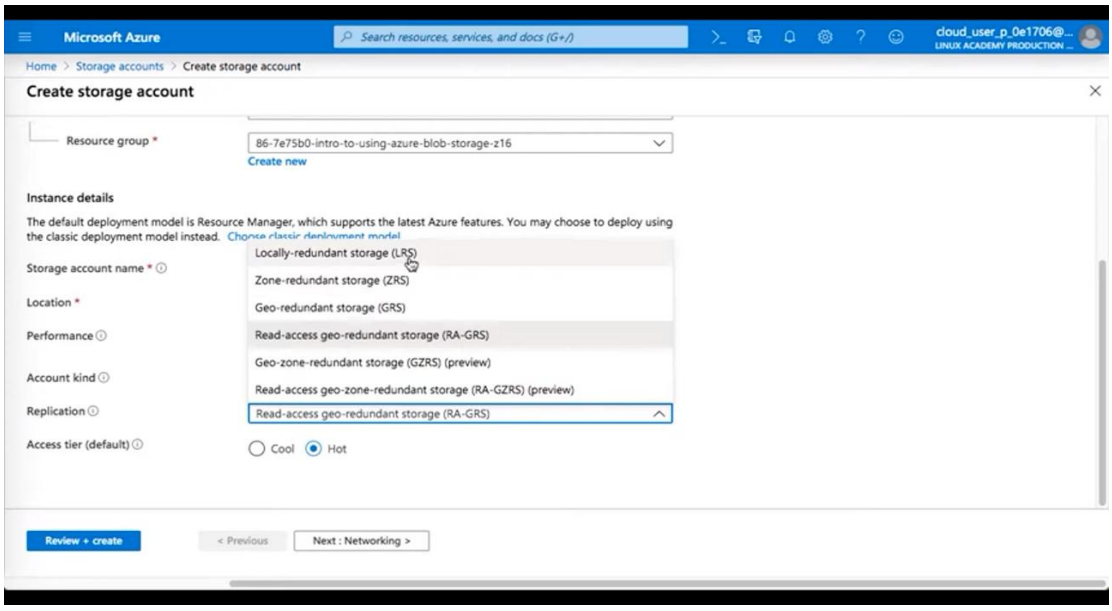
Aim:

Step by Step Procedure to create a storage account:
Step 1: Create a Storage Account

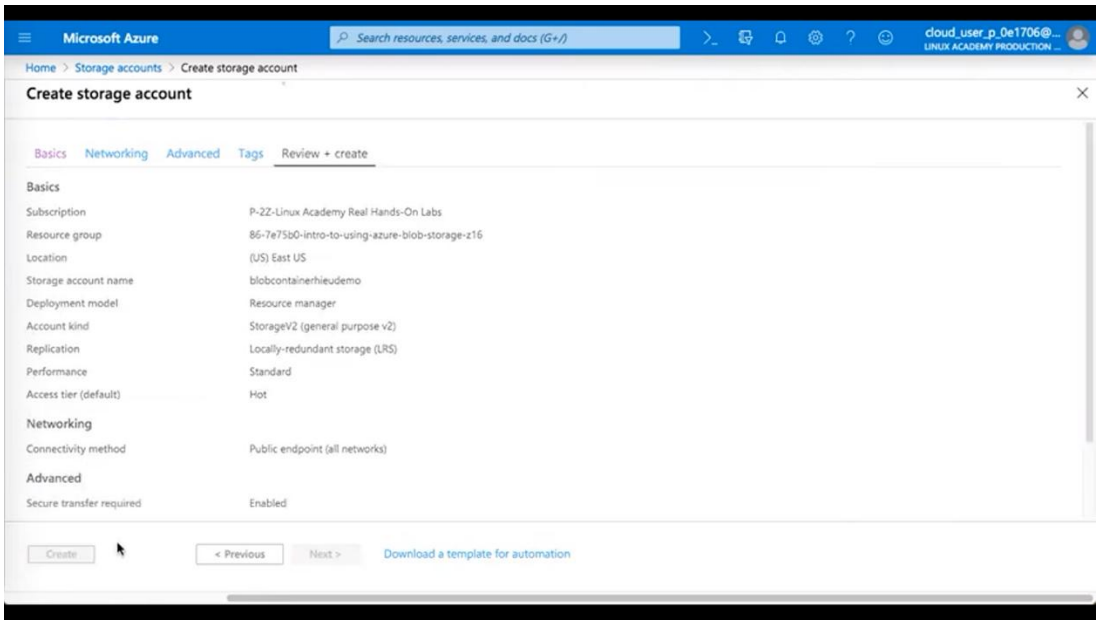
1. Sign in to Azure Portal: https://portal.azure.com
2. In the left menu, select "Storage accounts".



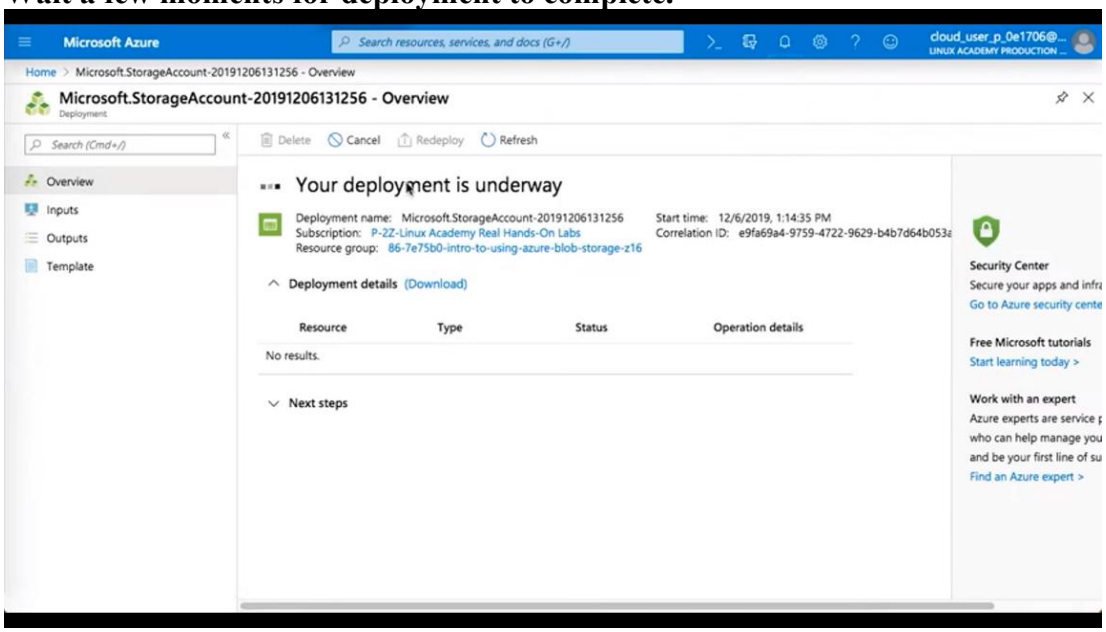
3. Click "+ Create".
4. Fill in the following:
    - Subscription: Choose your subscription.
    - Resource group: **Select an existing one or create a new one**.
    - **Storage account name: Unique name (lowercase letters and numbers only)**.
    - Region: Choose your region.
    - Performance: **Standard** or Premium.
    - Redundancy: **LRS**, GRS, etc.

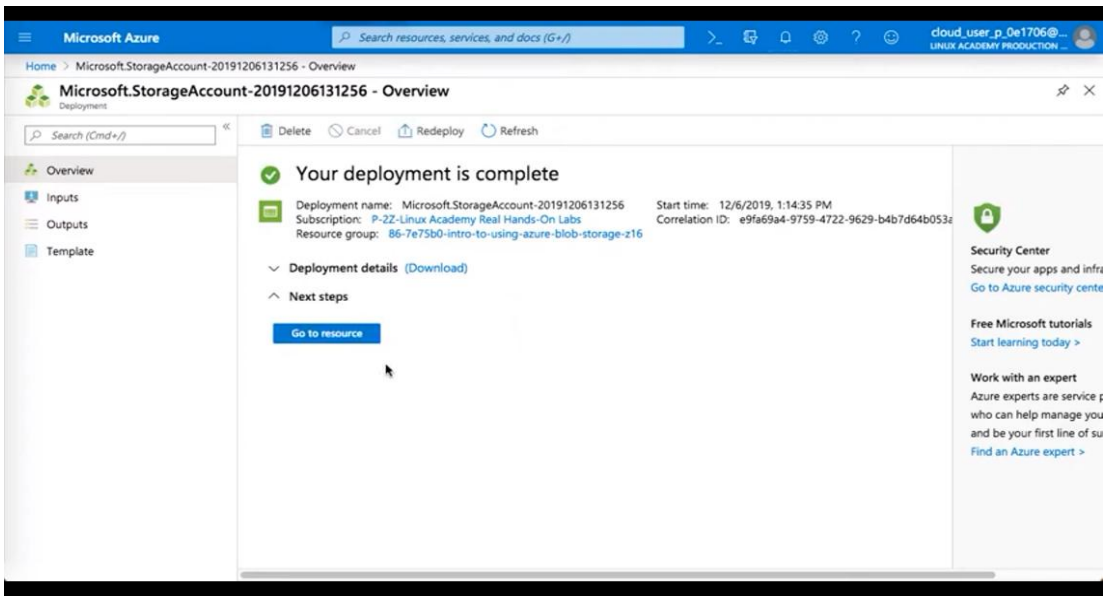


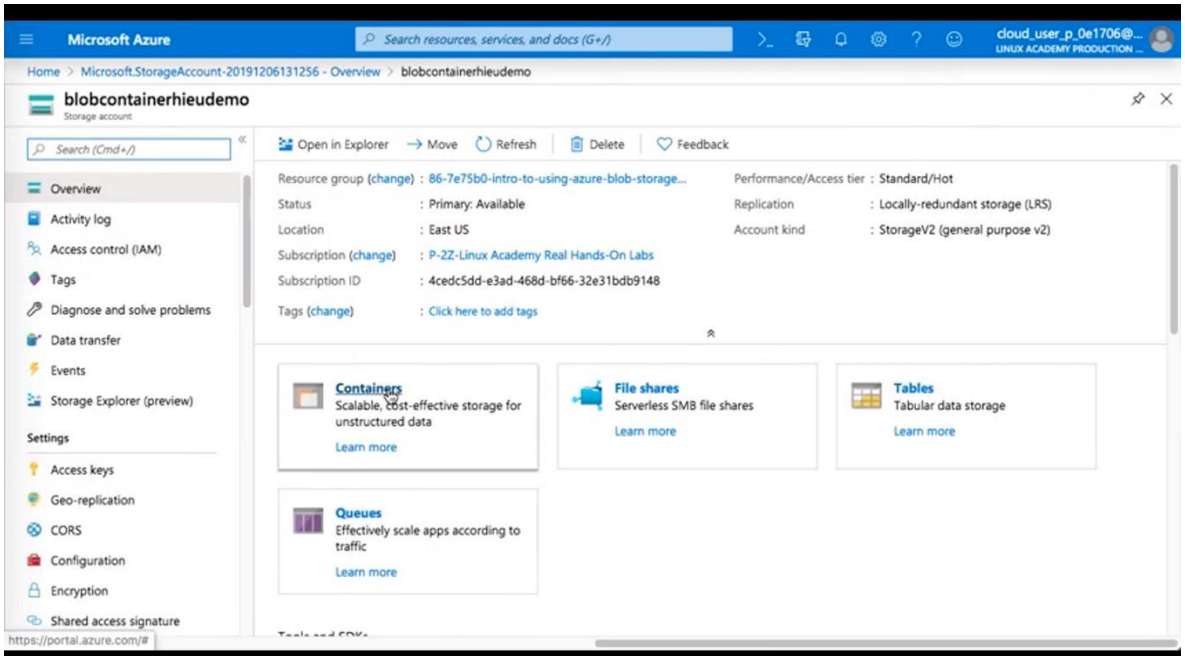
5. Click "Review + Create", then "Create".

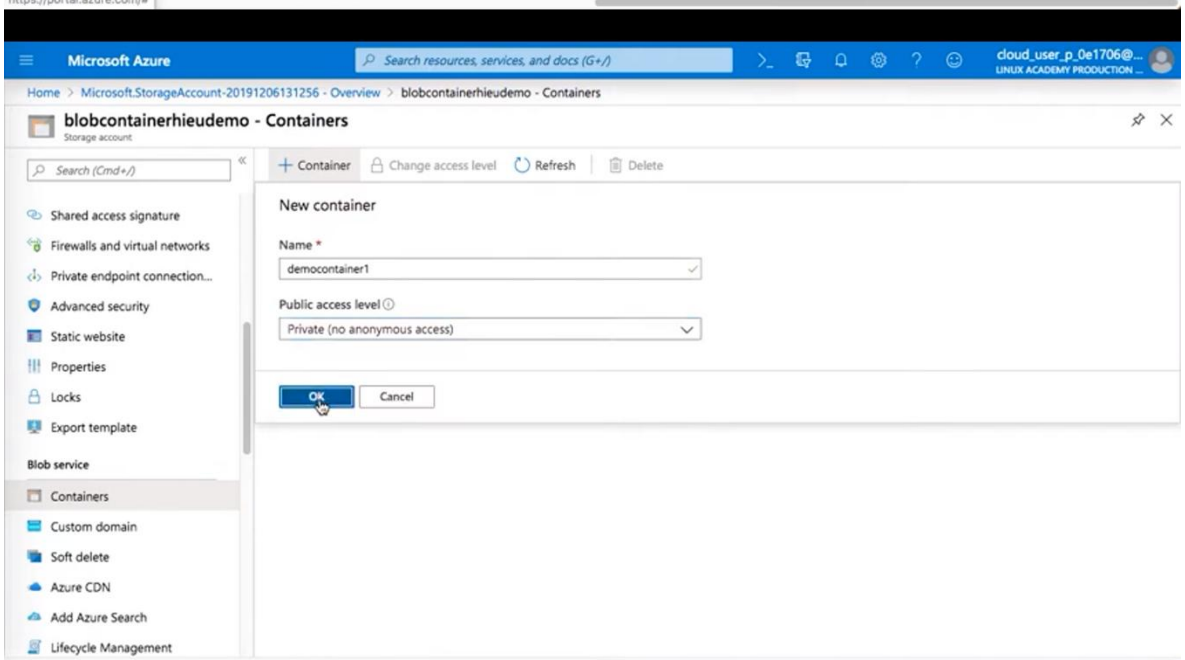


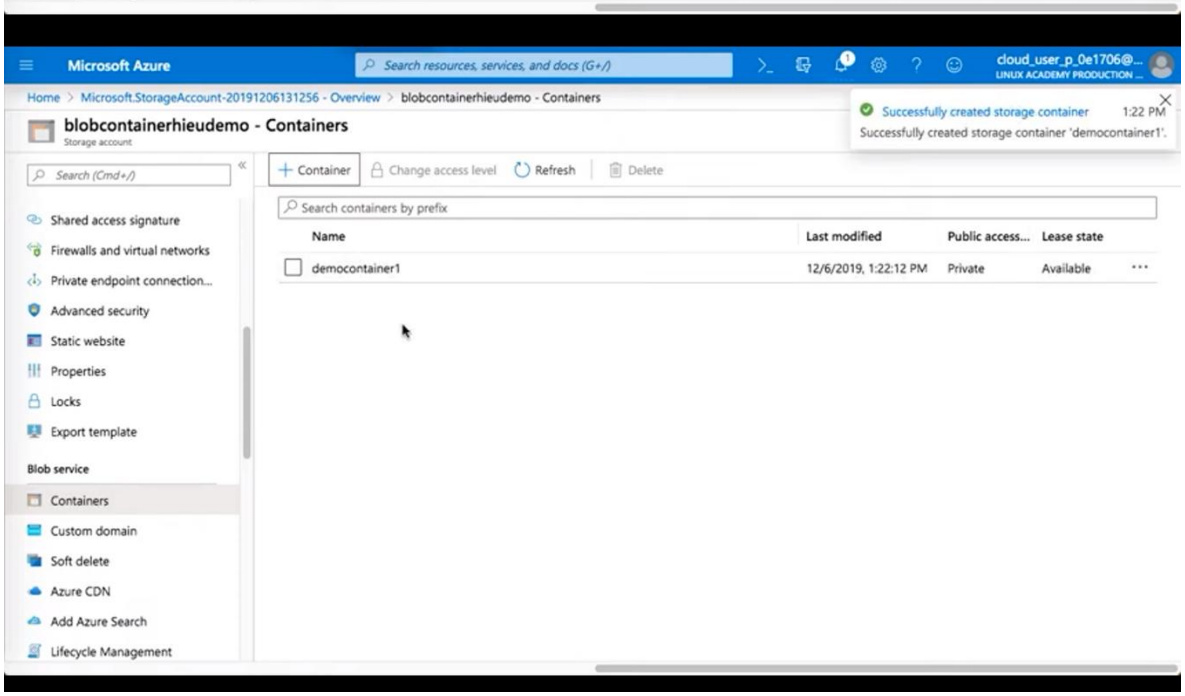
Wait a few moments for deployment to complete.

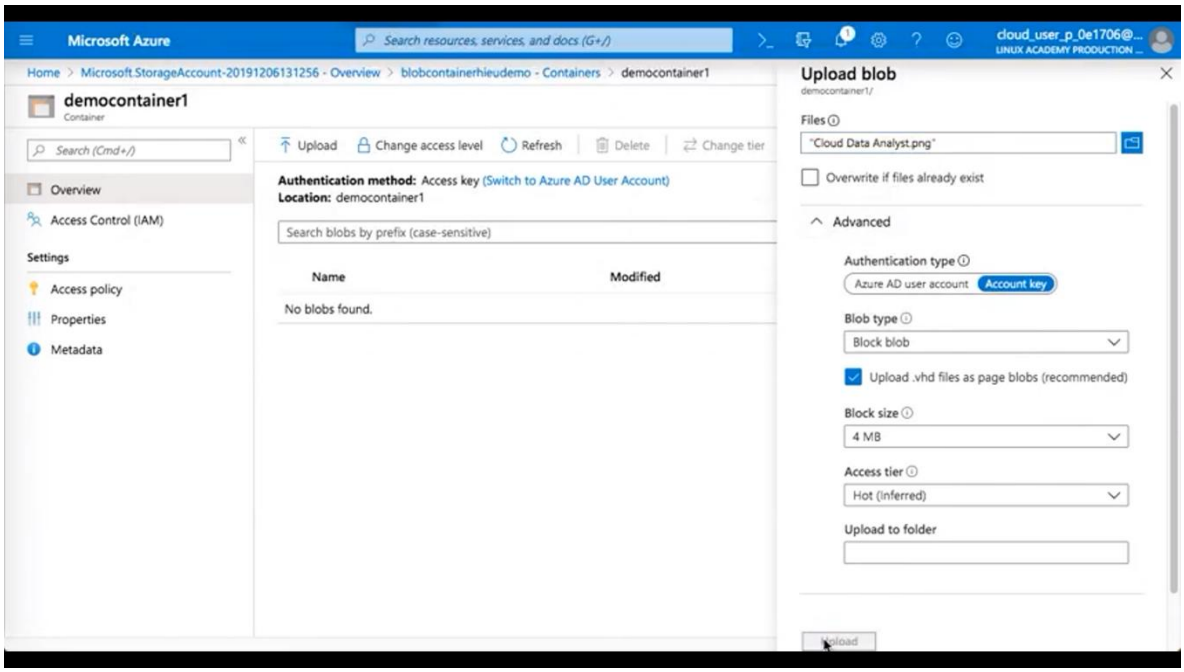




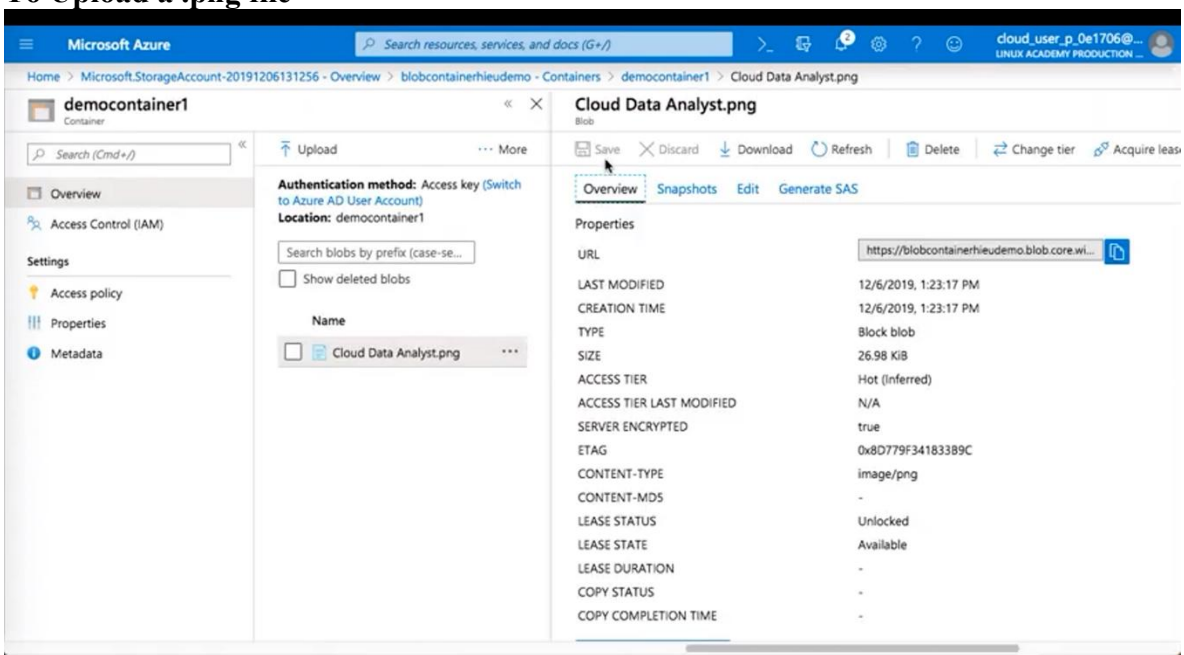

Microsoft Azure
Search resources, services, and docs (G+/)
cloud_user_p_0e1706@...
LINUX ACADEMY PRODUCTION ...
Home > Microsoft.StorageAccount-20191206131256 - Overview > blobcontainerhieudemo
blobcontainerhieudemo
Storage account
Search (Cmd+/)
Open in Explorer
Move
Refresh
Delete
Feedback
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Data transfer
Events
Storage Explorer (preview)
Settings
Access keys
Geo-replication
CORS
Configuration
Encryption
Shared access signature
Resource group (change) : 86-7e75b0-intro-to-using-azure-blob-storage...
Status : Primary: Available
Location : East US
Subscription (change) : P-2Z-Linux Academy Real Hands-On Labs
Subscription ID : 4cedc5dd-e3ad-468d-bf66-32e31bdb9148
Tags (change) : Click here to add tags
Performance/Access tier : Standard/Hot
Replication : Locally-redundant storage (LRS)
Account kind : StorageV2 (general purpose v2)
Containers
Scalable, cost-effective storage for unstructured data
Learn more
File shares
Serverless SMB file shares
Learn more
Tables
Tabular data storage
Learn more
Queues
Effectively scale apps according to traffic
Learn more
https://portal.azure.com/#

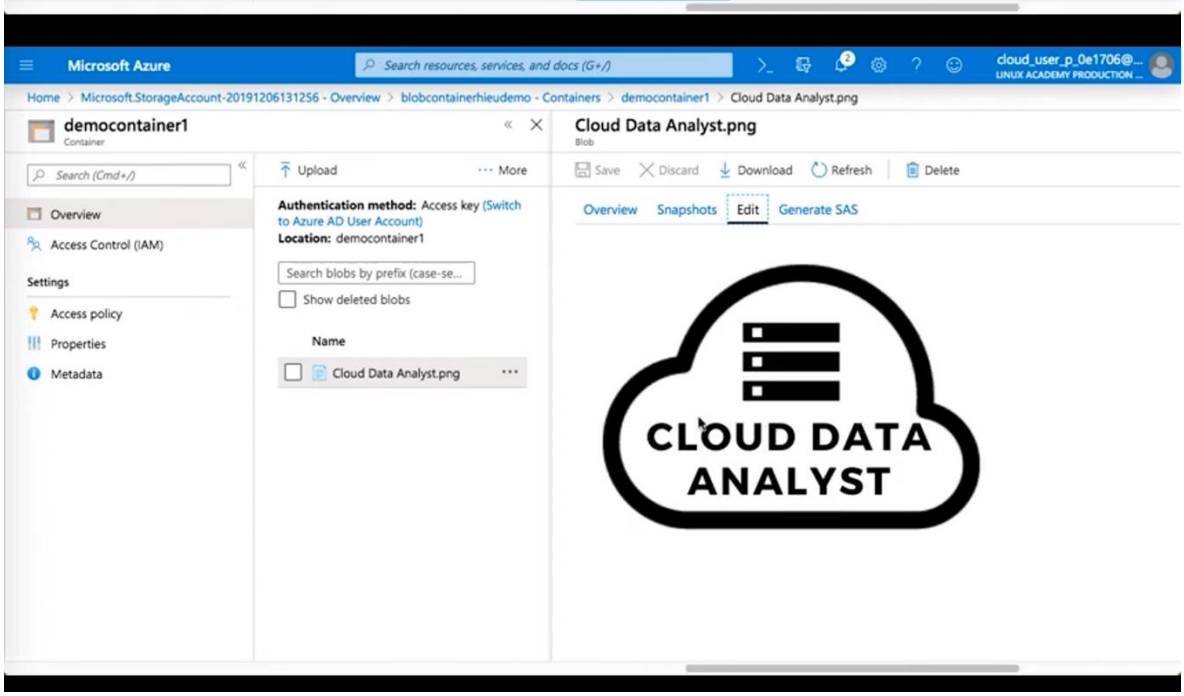

Microsoft Azure
Search resources, services, and docs (G+/)
cloud_user_p_0e1706@...
LINUX ACADEMY PRODUCTION ...
Home > Microsoft.StorageAccount-20191206131256 - Overview > blobcontainerhieudemo - Containers
blobcontainerhieudemo - Containers
Storage account
Search (Cmd+/)
Container
Change access level
Refresh
Delete
Shared access signature
Firewalls and virtual networks
Private endpoint connection...
Advanced security
Static website
Properties
Locks
Export template
Blob service
Containers
Custom domain
Soft delete
Azure CDN
Add Azure Search
Lifecycle Management
New container
Name *
democontainer1
Public access level
Private (no anonymous access)
OK
Cancel


Microsoft Azure
Search resources, services, and docs (G+/)
cloud_user_p_0e1706@...
LINUX ACADEMY PRODUCTION ...
Home > Microsoft.StorageAccount-20191206131256 - Overview > blobcontainerhieudemo - Containers
blobcontainerhieudemo - Containers
Storage account
Successfully created storage container
1:22 PM
Successfully created storage container 'democontainer1'.
Search (Cmd+/)
Container
Change access level
Refresh
Delete
Shared access signature
Firewalls and virtual networks
Private endpoint connection...
Advanced security
Static website
Properties
Locks
Export template
Blob service
Containers
Custom domain
Soft delete
Azure CDN
Add Azure Search
Lifecycle Management
Search containers by prefix
Name
Last modified
Public access...
Lease state
democontainer1
12/6/2019, 1:22:12 PM
Private
Available



To Upload a .png file





Step 2: Create a Hosted Service Component (Azure App Service Example)

Note: "Hosted Service" is a legacy term from Azure Cloud Services. Modern equivalent is App Service or Virtual Machine.

Create App Service (Web App)

1. In the left menu, click "App Services".
2. Click "+ Create".
3. On the Basics tab, enter:
    - Subscription
    - Resource Group (same as storage, if desired)
    - Name: Web app name
    - Publish: Code or Docker Container
    - Runtime stack: e.g., .NET, Node.js, Python
    - Region: Same as storage account, if needed
4. Click Next through other tabs or customize.
5. Click "Review + Create", then "Create".

RESULT:

Ex.No.3 Deployment of an application using platform management portal
Date:

AIM :

To Deploying a Flask Application to AWS Elastic Beanstalk Using Platform Management Portal

Requirements:

- AWS account
- IAM permissions to use Elastic Beanstalk
- Flask app with:
    - application.py
    - requirements.txt
    - zipped into a .zip file (no parent folder inside the zip)

Procedure:

Step 1: Prepare Your Flask App

1. Create application.py with the following code:

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def home():
    return 'Hello from Elastic Beanstalk!'

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000)
```

2. Create requirements.txt:

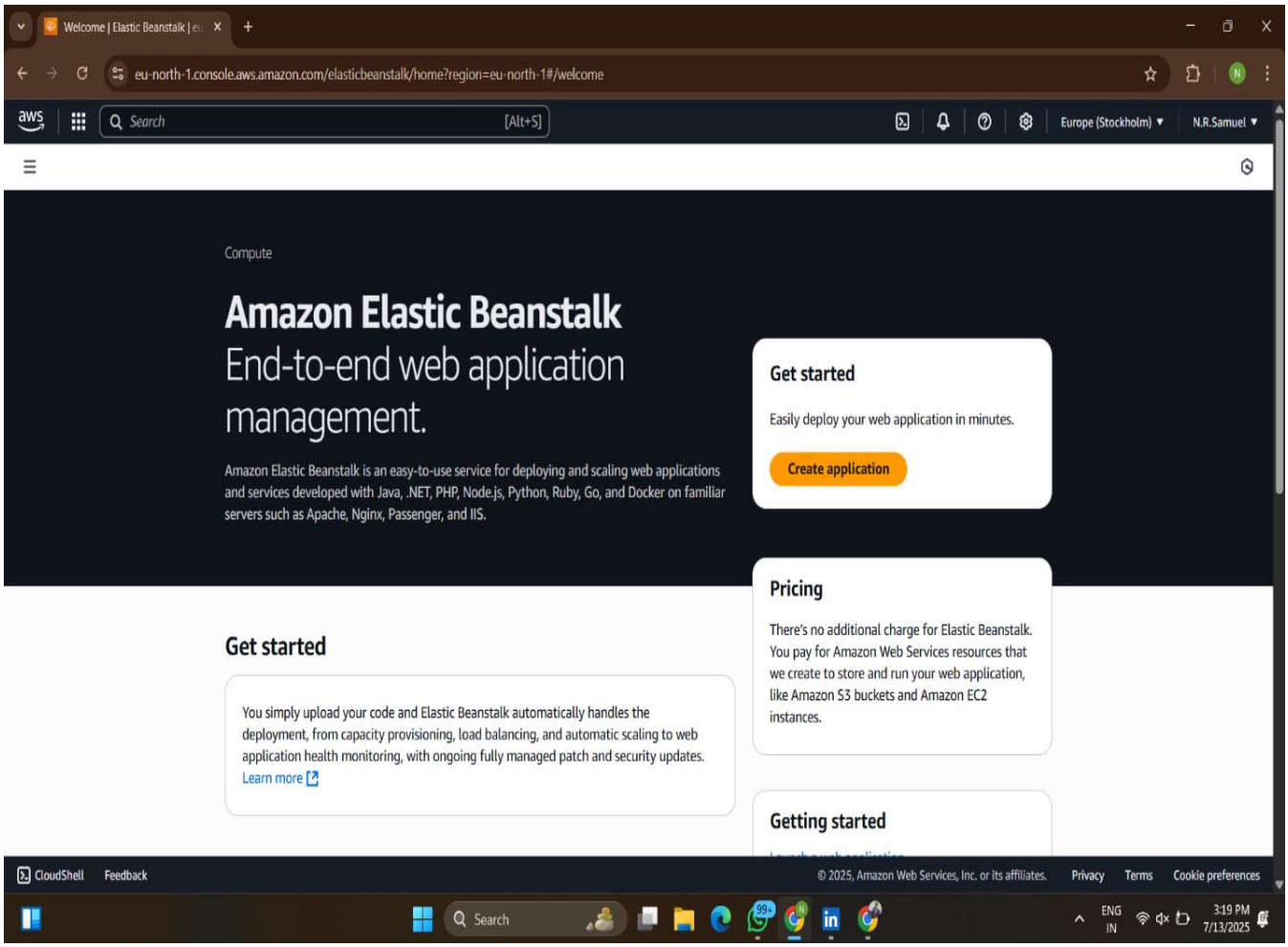
```
Flask==2.3.2
```

3. Zip the files:

application.py
requirements.txt
Make sure these files are **not inside a subfolder** in the zip.

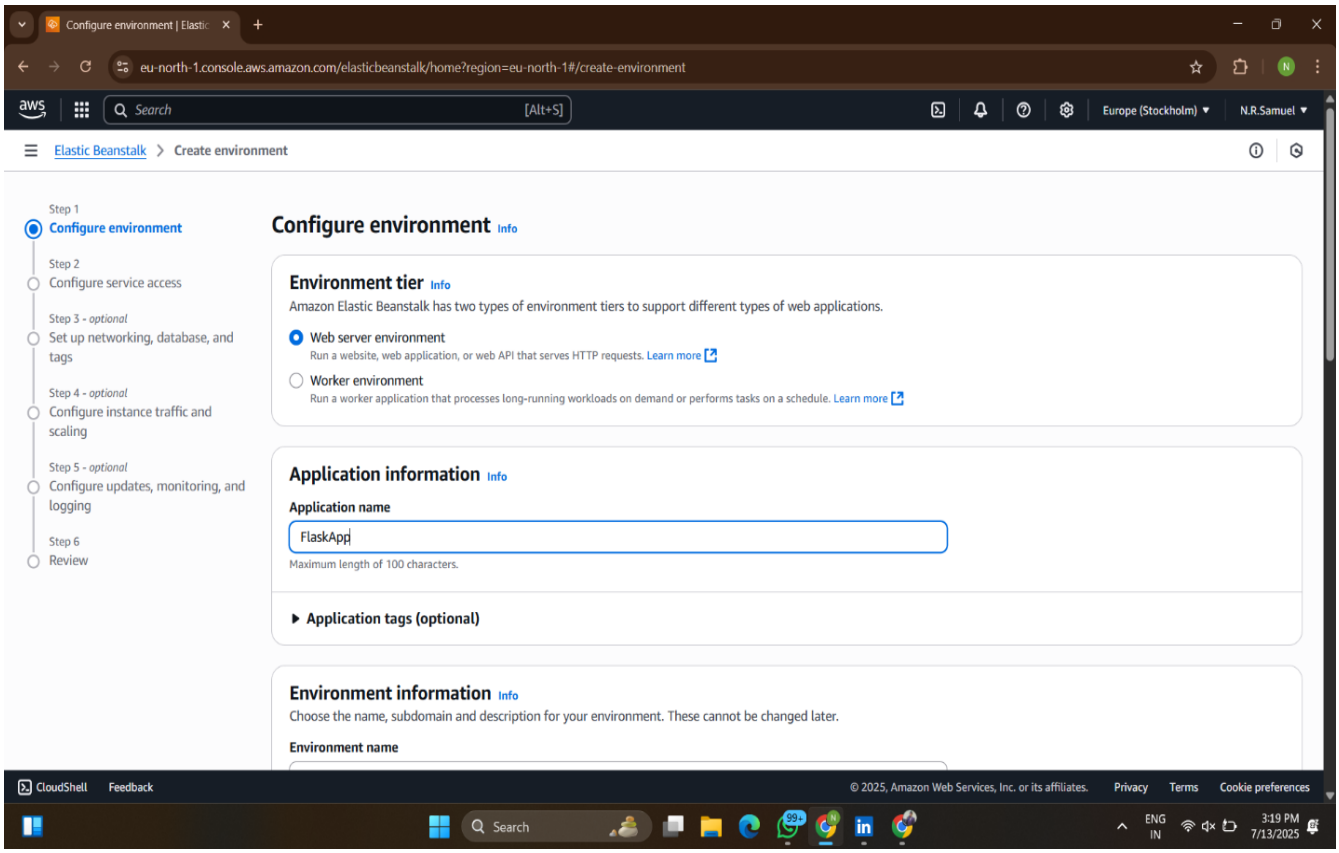
Step 2: Open AWS Elastic Beanstalk

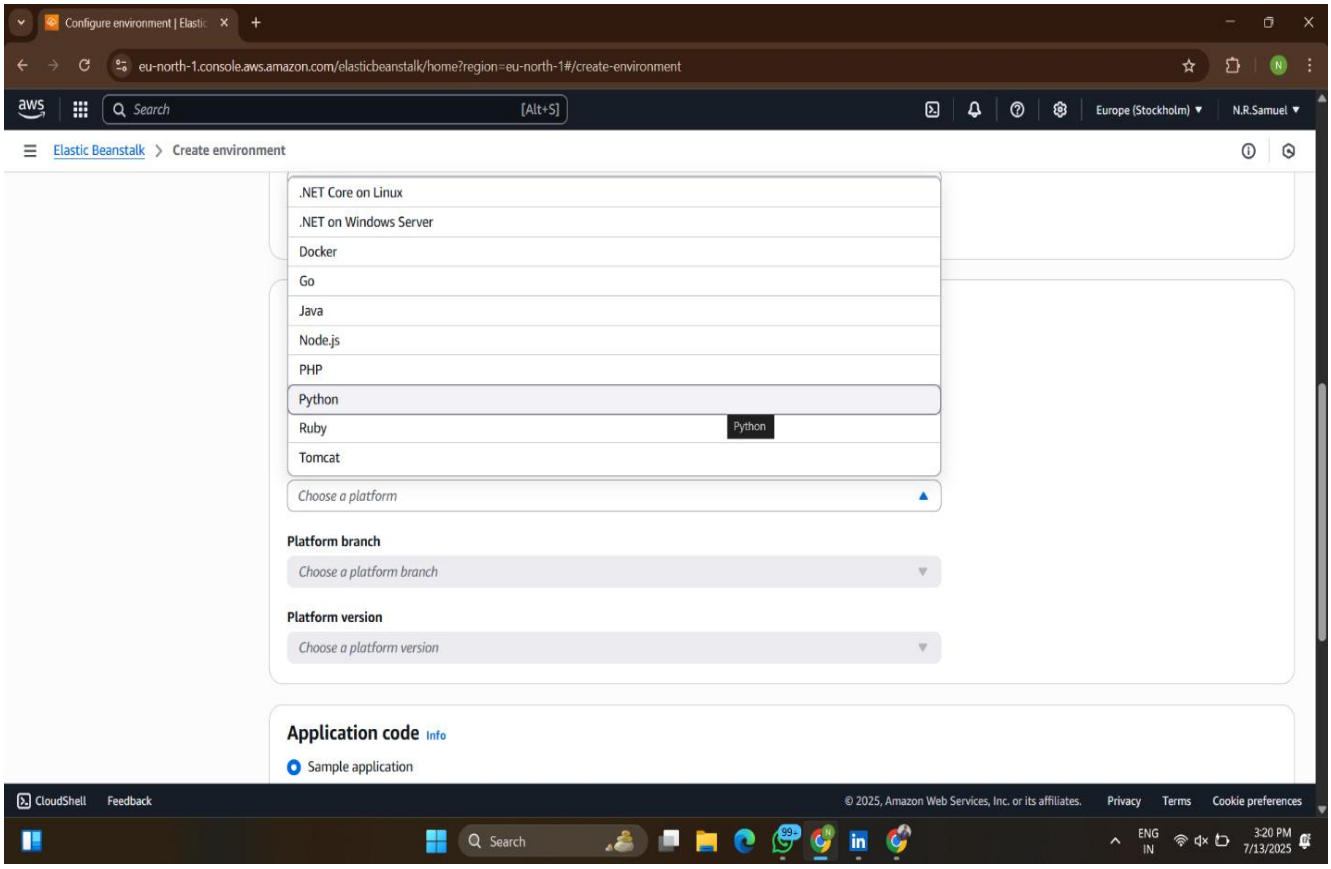
1. Sign in to AWS Console
2. Search for **Elastic Beanstalk** in the search bar and open it

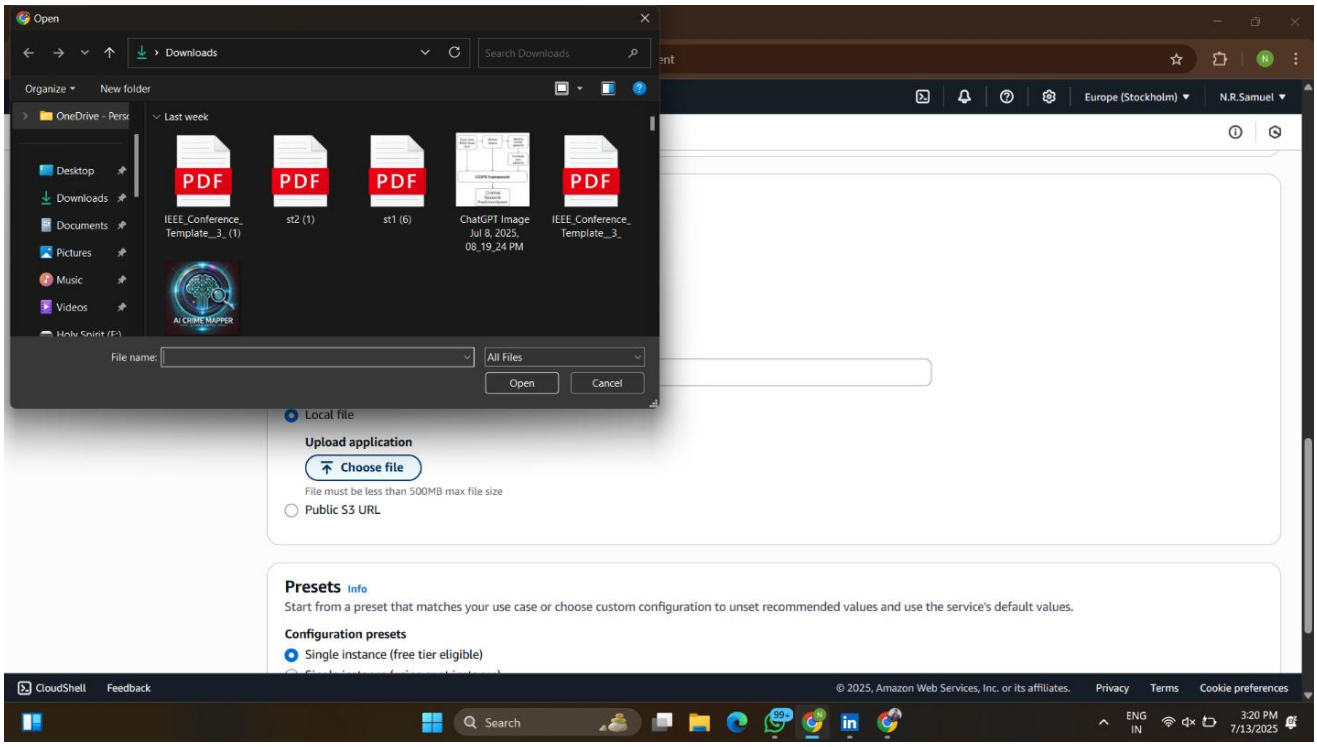


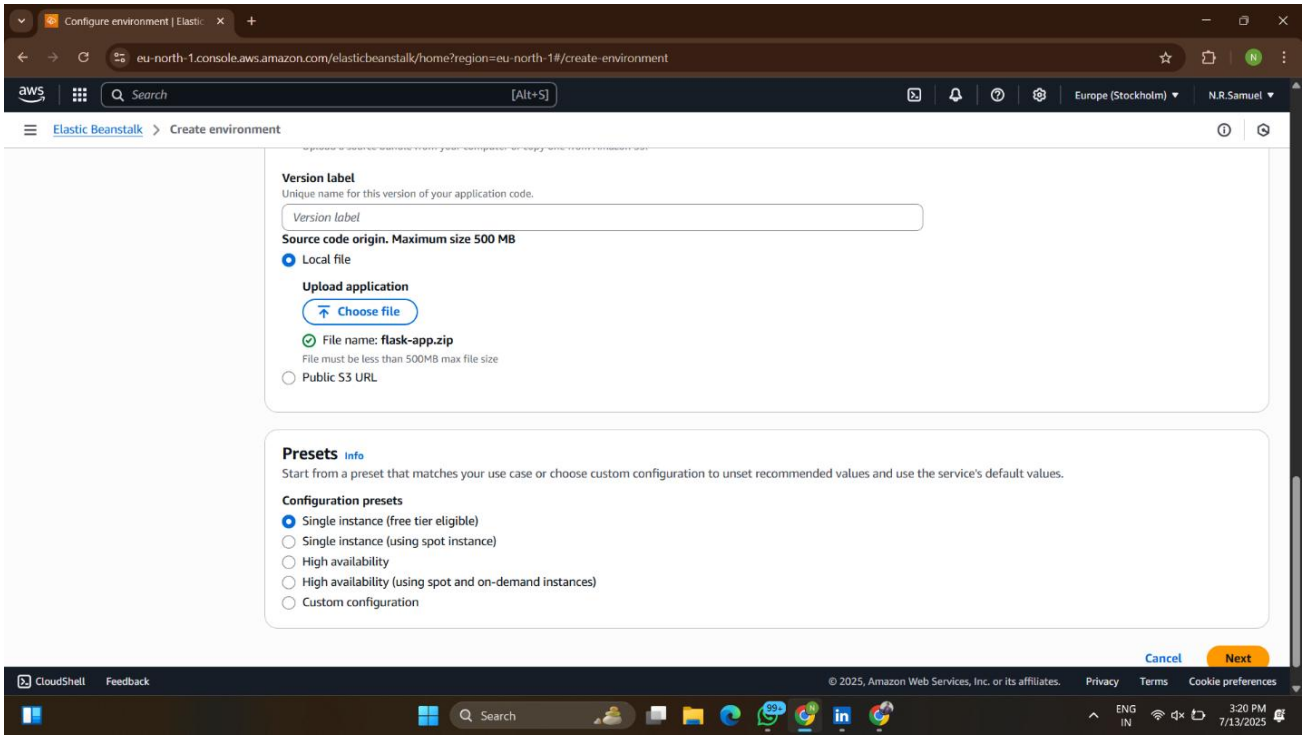
Step 3: Create a New Application

1. Click **Create application**
2. Fill in:
    - **Application name**: e.g., FlaskApp
    - **Platform**: Select **Python**
    - **Application code**: Choose **Upload your code**, then upload your .zip file
3. Click **Create environment** (it will use Web server environment)


Configure environment | Elastic
eu-north-1.console.aws.amazon.com/elasticbeanstalk/home?region=eu-north-1#/create-environment
Search
[Alt+S]
Europe (Stockholm)
N.R.Samuel
Elastic Beanstalk
Create environment
Step 1
Configure environment
Step 2
Configure service access
Step 3 - optional
Set up networking, database, and tags
Step 4 - optional
Configure instance traffic and scaling
Step 5 - optional
Configure updates, monitoring, and logging
Step 6
Review
Configure environment Info
Environment tier Info
Amazon Elastic Beanstalk has two types of environment tiers to support different types of web applications.
Web server environment
Run a website, web application, or web API that serves HTTP requests. Learn more
Worker environment
Run a worker application that processes long-running workloads on demand or performs tasks on a schedule. Learn more
Application information Info
Application name
FlaskApp
Maximum length of 100 characters.
Application tags (optional)
Environment information Info
Choose the name, subdomain and description for your environment. These cannot be changed later.
Environment name
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN
3:19 PM
7/13/2025

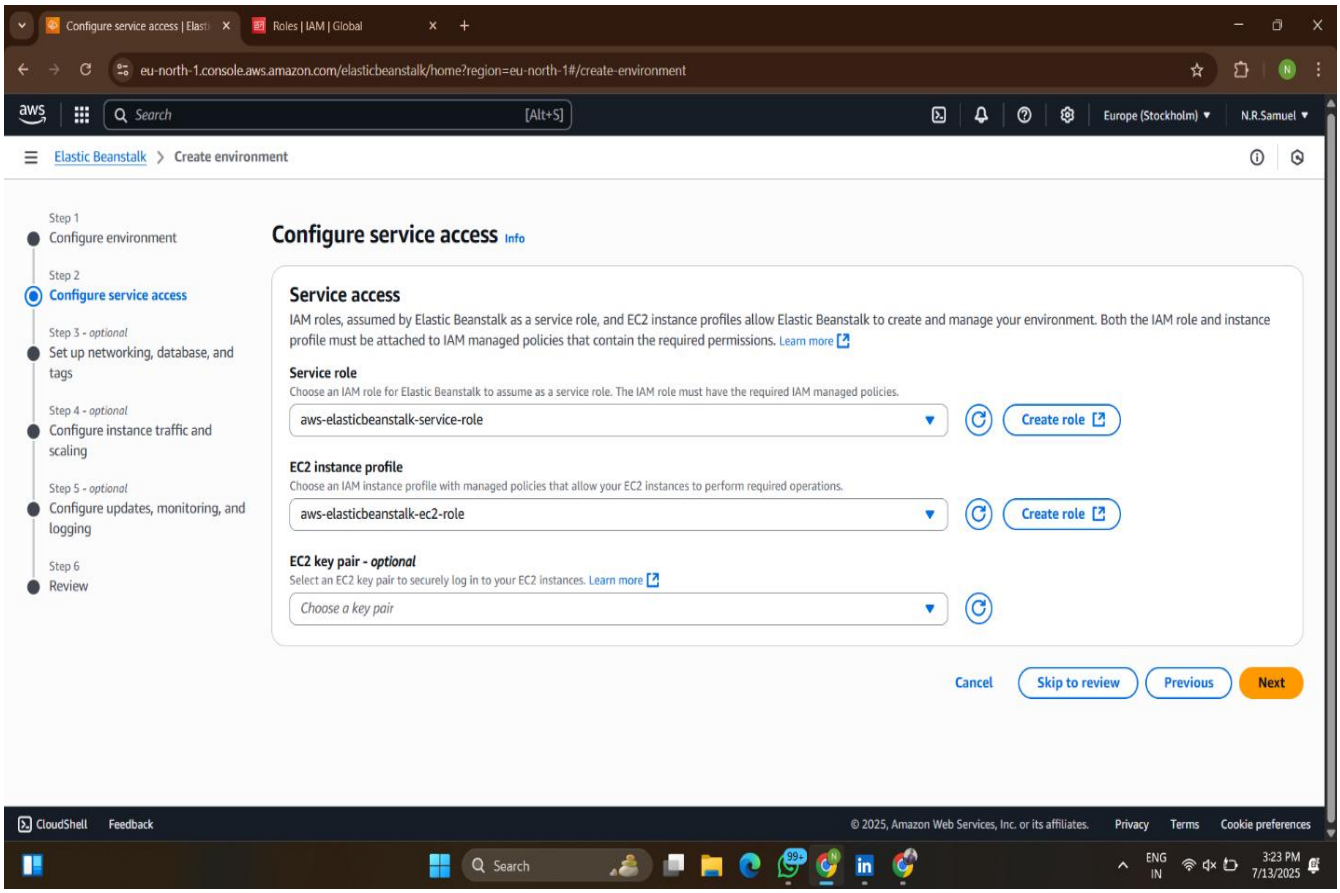

Configure environment | Elastic
eu-north-1.console.aws.amazon.com/elasticbeanstalk/home?region=eu-north-1#/create-environment
Search
[Alt+S]
Europe (Stockholm)
N.R.Samuel
Elastic Beanstalk
Create environment
.NET Core on Linux
.NET on Windows Server
Docker
Go
Java
Node.js
PHP
Python
Ruby
Tomcat
Python
Choose a platform
Platform branch
Choose a platform branch
Platform version
Choose a platform version
Application code Info
Sample application
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN
3:20 PM
7/13/2025

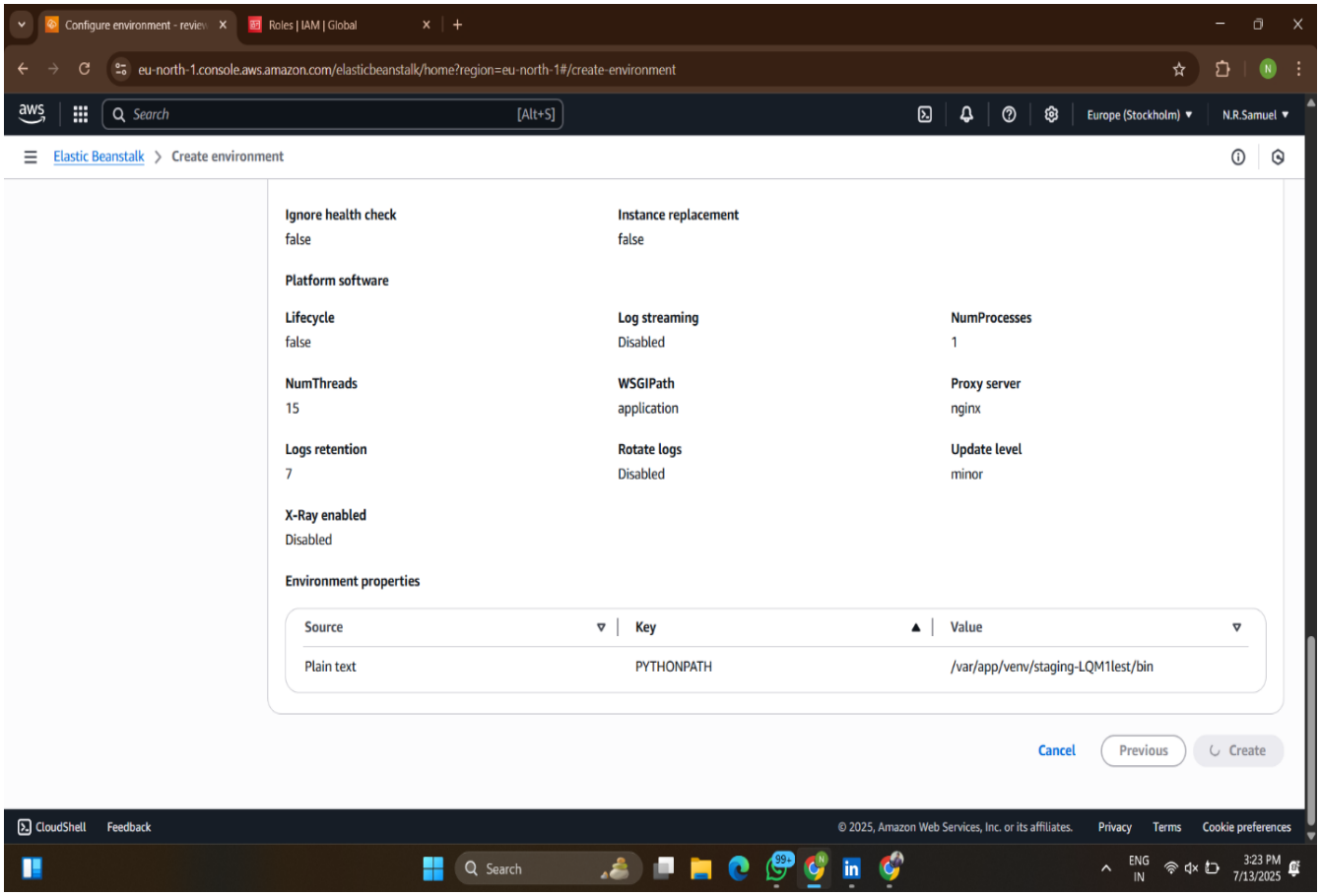

Open
Downloads
Search Downloads
Organize
New folder
OneDrive - Perso
Last week
Desktop
Downloads
Documents
Pictures
Music
Videos
IEEE_Conference_Template_3_ (1)
st2 (1)
st1 (6)
ChatGPT Image Jul 8, 2025, 08_19_24 PM
IEEE_Conference_Template_3_
AI CRIME MAPPER
File name:
All Files
Open
Cancel
Europe (Stockholm)
N.R.Samuel
Local file
Upload application
Choose file
File must be less than 500MB max file size
Public S3 URL
Presets Info
Start from a preset that matches your use case or choose custom configuration to unset recommended values and use the service's default values.
Configuration presets
Single instance (free tier eligible)
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG IN
3:20 PM 7/13/2025


Configure environment | Elastic
eu-north-1.console.aws.amazon.com/elasticbeanstalk/home?region=eu-north-1#/create-environment
aws
Search
[Alt+S]
Europe (Stockholm)
N.R.Samuel
Elastic Beanstalk > Create environment
Version label
Unique name for this version of your application code.
Version label
Source code origin. Maximum size 500 MB
Local file
Upload application
Choose file
File name: flask-app.zip
File must be less than 500MB max file size
Public S3 URL
Presets Info
Start from a preset that matches your use case or choose custom configuration to unset recommended values and use the service's default values.
Configuration presets
Single instance (free tier eligible)
Single instance (using spot instance)
High availability
High availability (using spot and on-demand instances)
Custom configuration
Cancel
Next
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG IN
3:20 PM 7/13/2025

Step 4: Configure IAM Roles

- Let AWS create default roles automatically

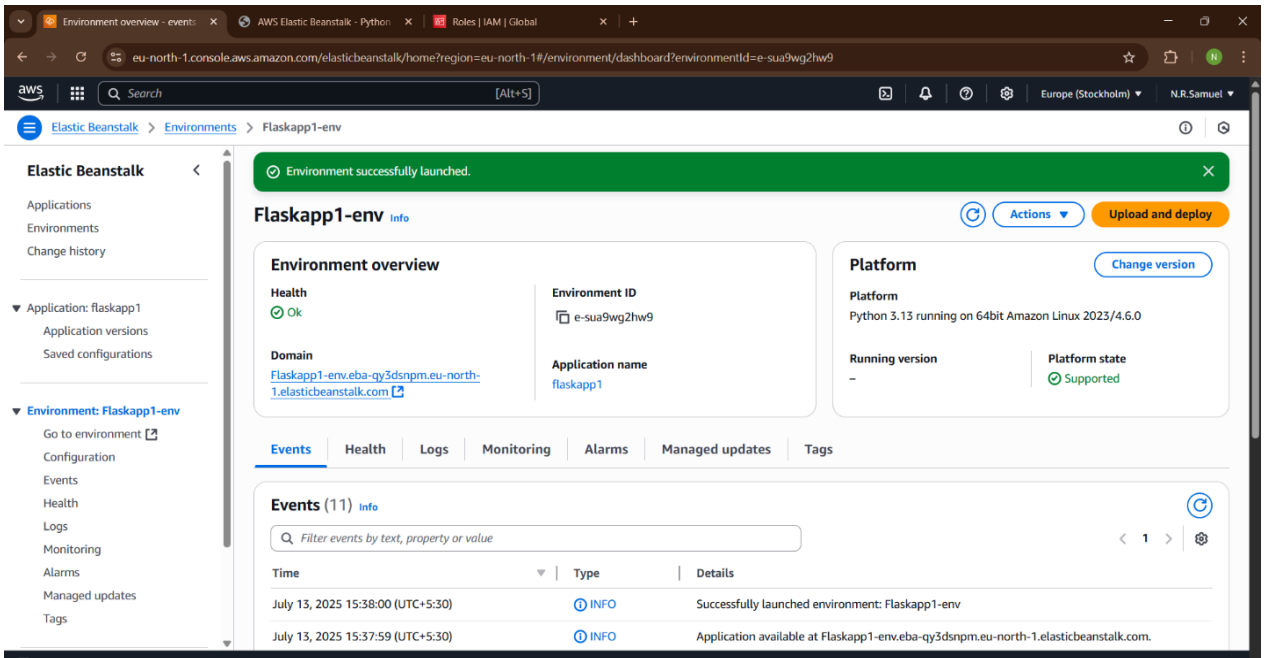


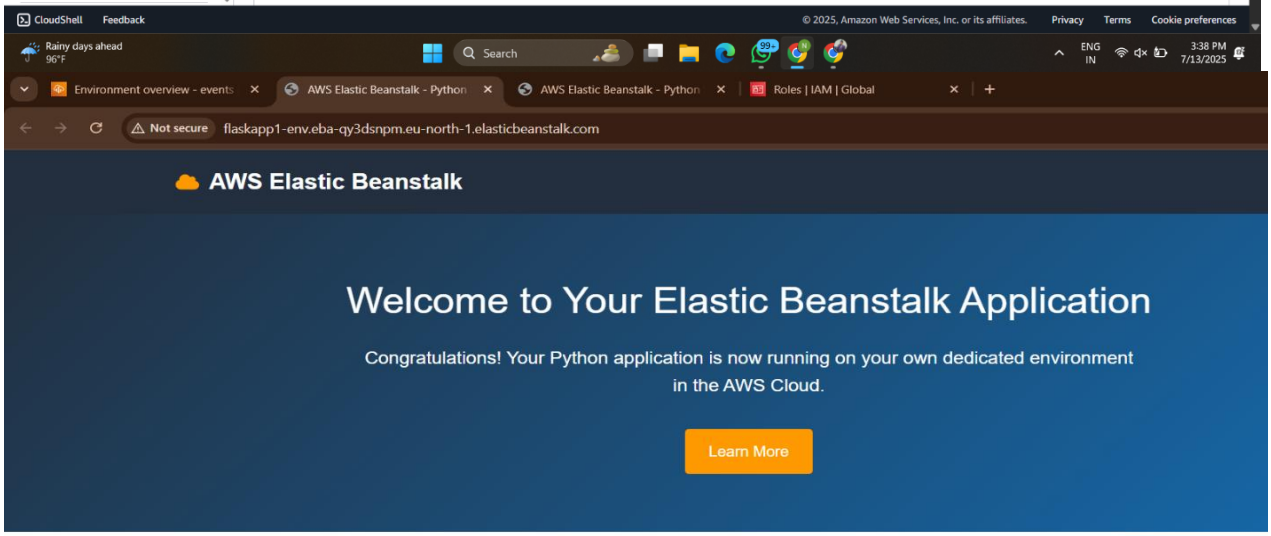


Step 5: Wait for Deployment

- Takes 2–5 minutes
- Once completed, environment status = **Green**
- At the top, you will see your app URL like:

http://myflaskapp-env.eba-xyz.elasticbeanstalk.com





Step 6: Delete Environment

1. Go to Elastic Beanstalk Console
2. Select your environment
3. Click **Actions > Terminate Environment**
4. Confirm termination

RESULT:

Ex:4 **Create a word document of your class time table and store on the**
Date: **cloud with docx and pdf format.**

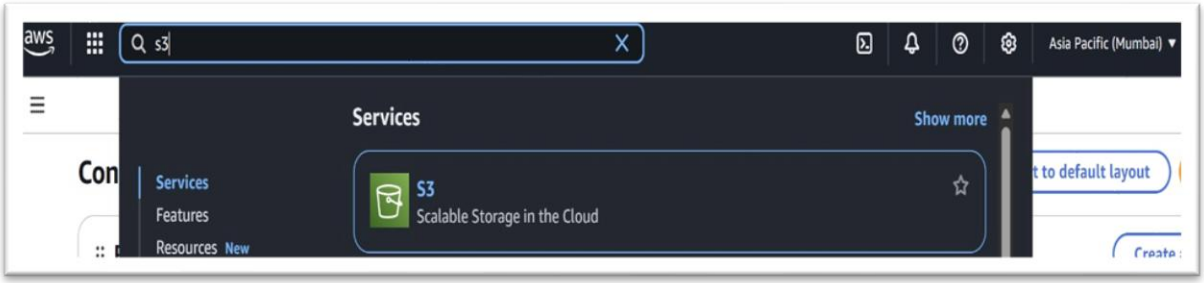
Aim:

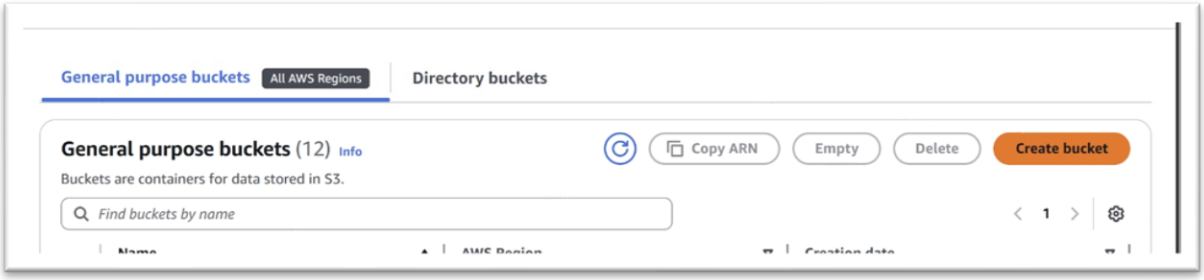
To create a class timetable in Microsoft Word format, convert it to PDF, and upload both to an AWS S3 bucket.

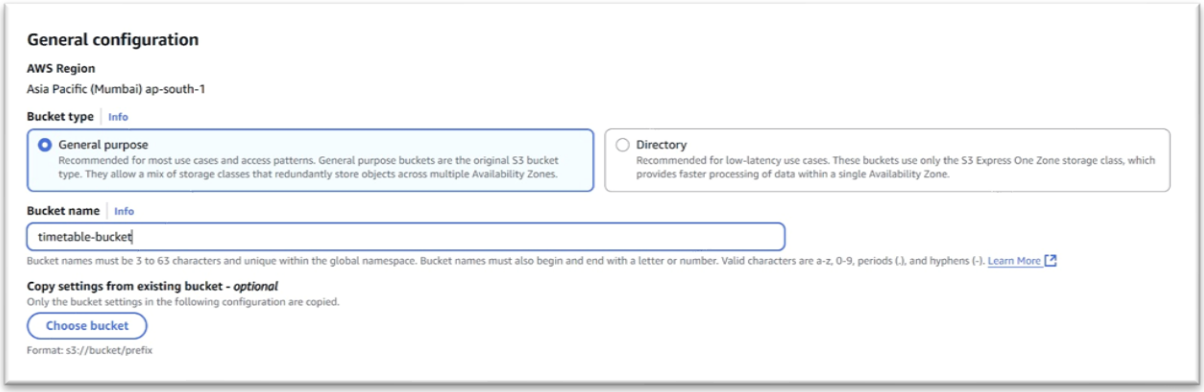
Procedure:

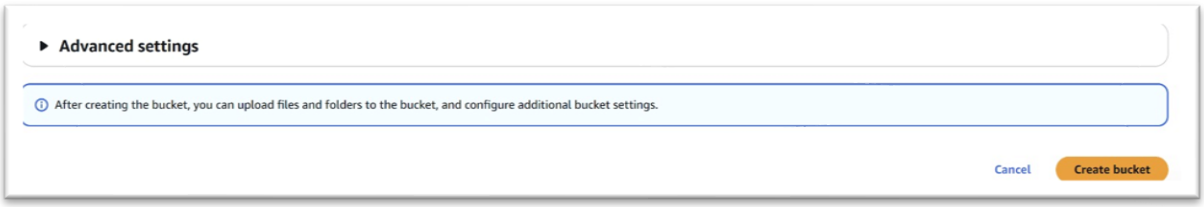
- Open Microsoft Word and create your class timetable using a table.
- Enter your subjects, timings, and days in the table.
- Save the document as 'Timetable.docx'.
- Go to File > Save As and choose PDF format to save as 'Timetable.pdf'.
- Log in to your AWS Console (https://console.aws.amazon.com).
- Search for 'S3' and open the S3 service.
- Click 'Create bucket'. Give it a unique name (e.g., timetable-bucket-<yourname>).
- Leave default options and click 'Create bucket'.
- Click your bucket name > Upload > Add files > select both DOCX and PDF files > Upload.
- To share files: click file > Permissions > Make public (optional).

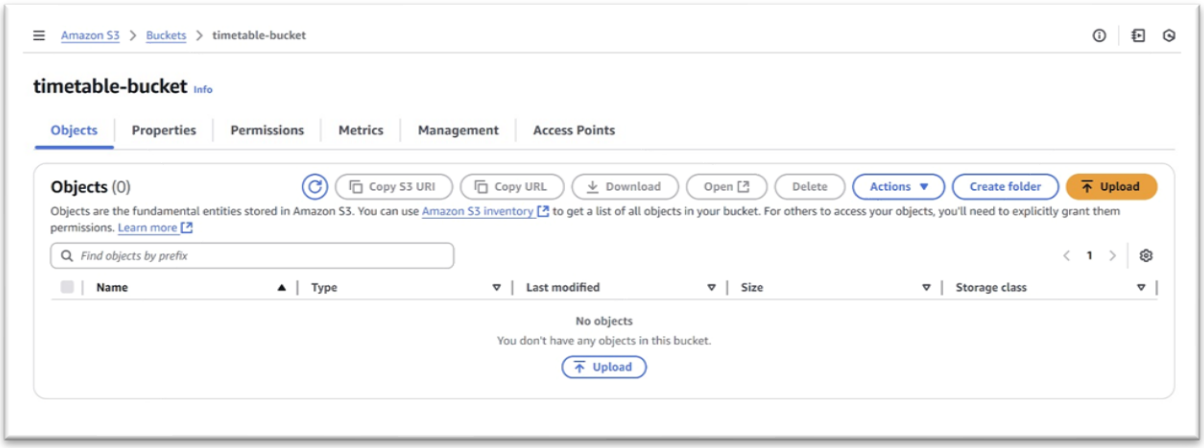
Output:

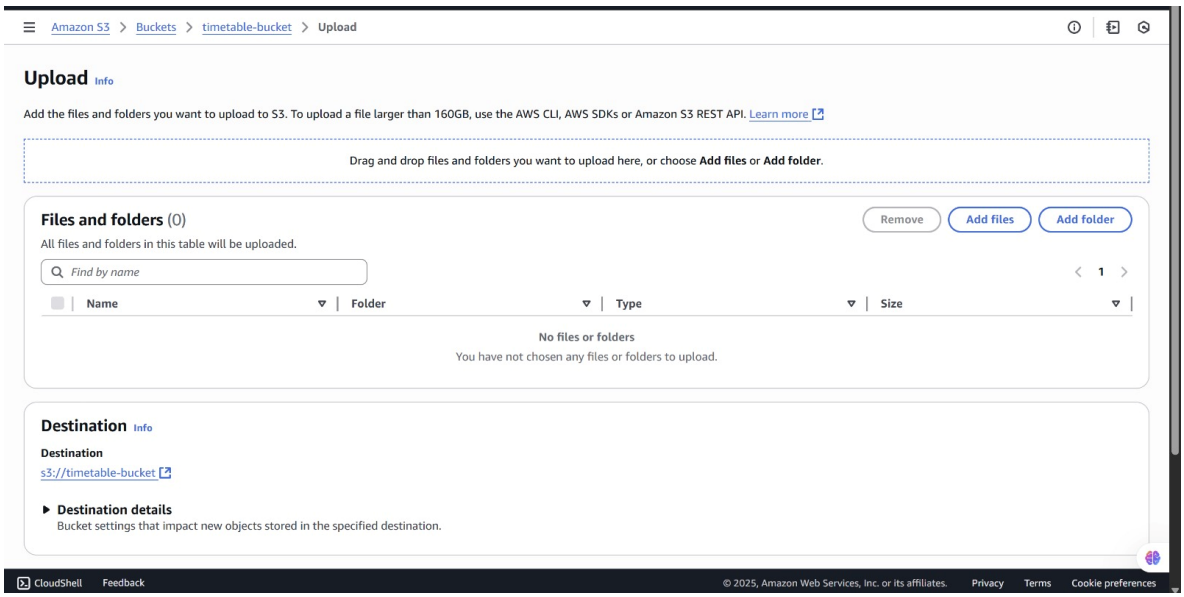


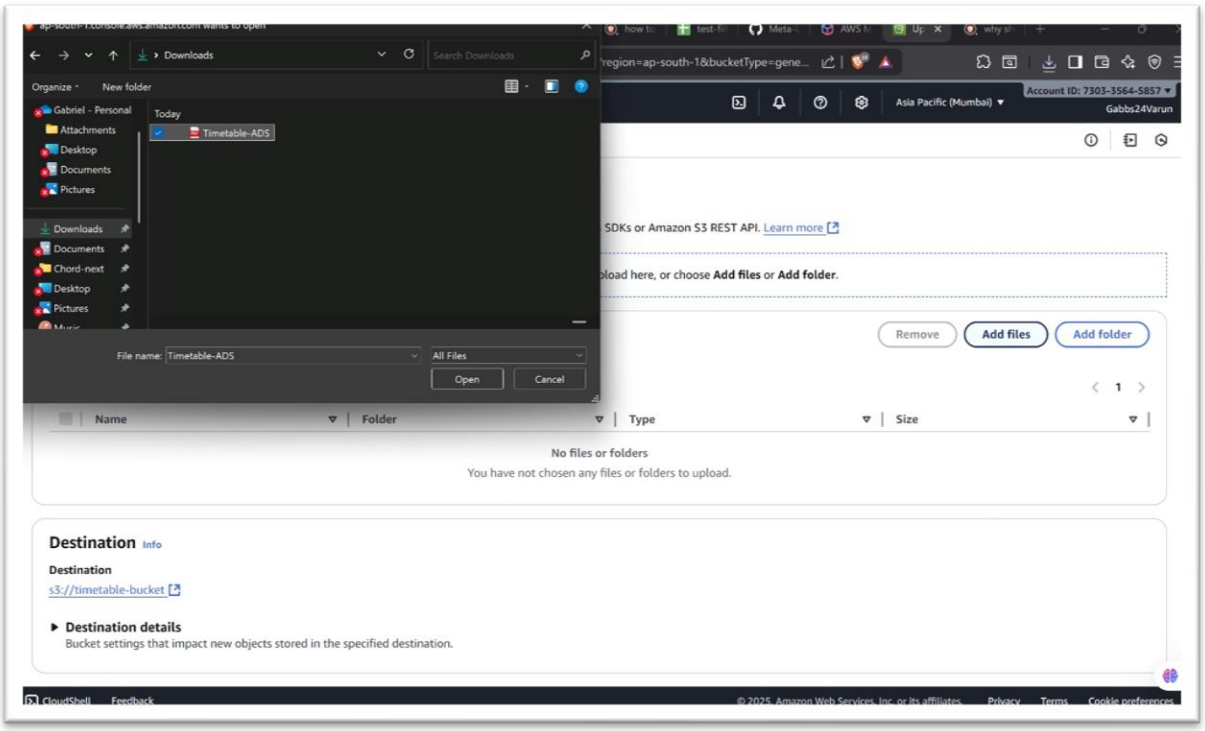


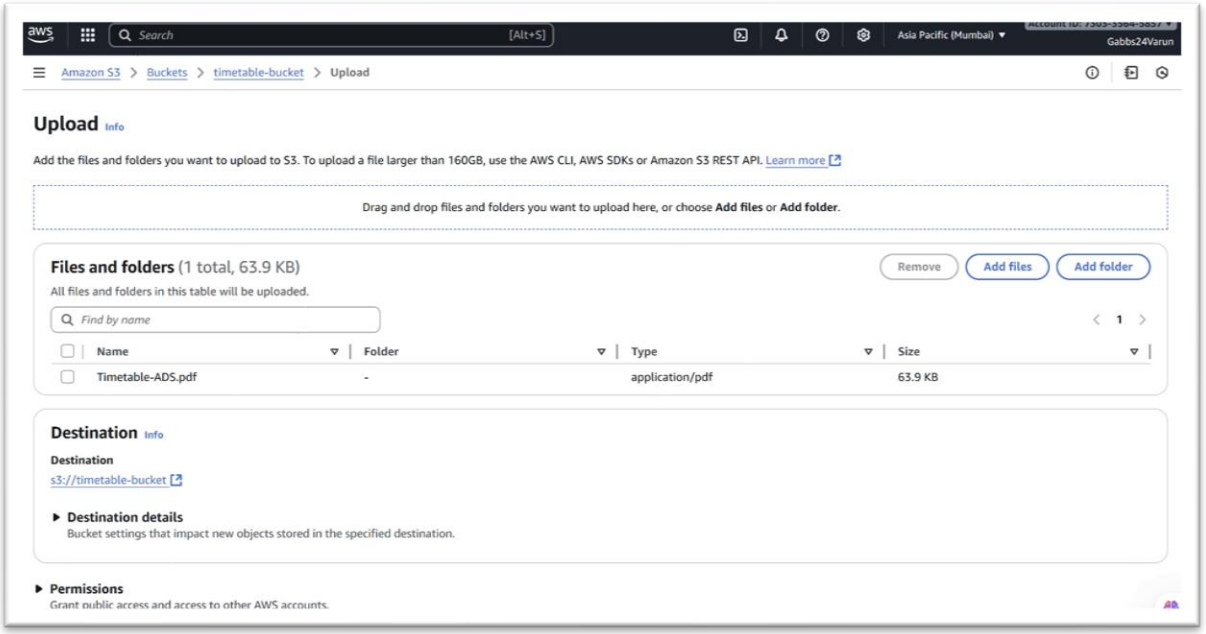

General configuration
AWS Region
Asia Pacific (Mumbai) ap-south-1
Bucket type Info
General purpose
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.
Directory
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.
Bucket name Info
timetable-bucket
Bucket names must be 3 to 63 characters and unique within the global namespace. Bucket names must also begin and end with a letter or number. Valid characters are a-z, 0-9, periods (.), and hyphens (-). Learn More
Copy settings from existing bucket - optional
Only the bucket settings in the following configuration are copied.
Choose bucket
Format: s3://bucket/prefix

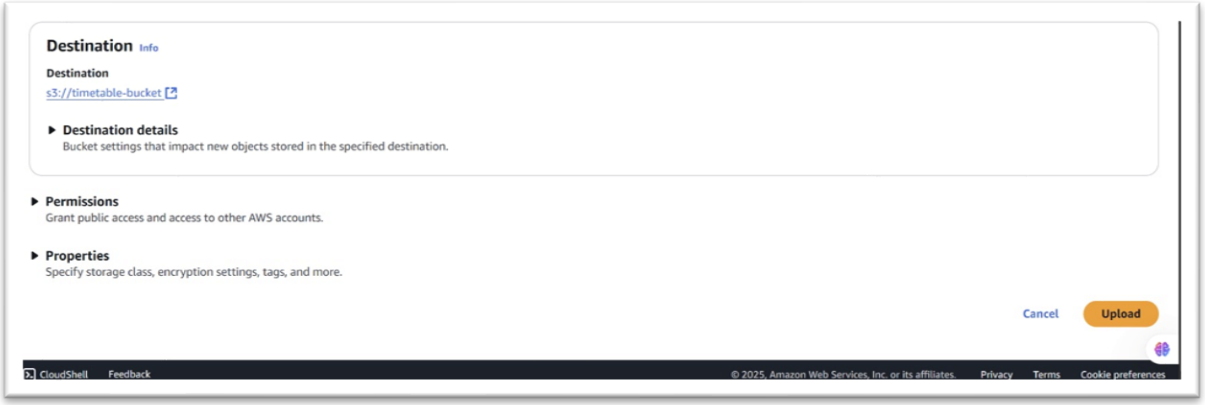

Advanced settings
After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.
Cancel
Create bucket

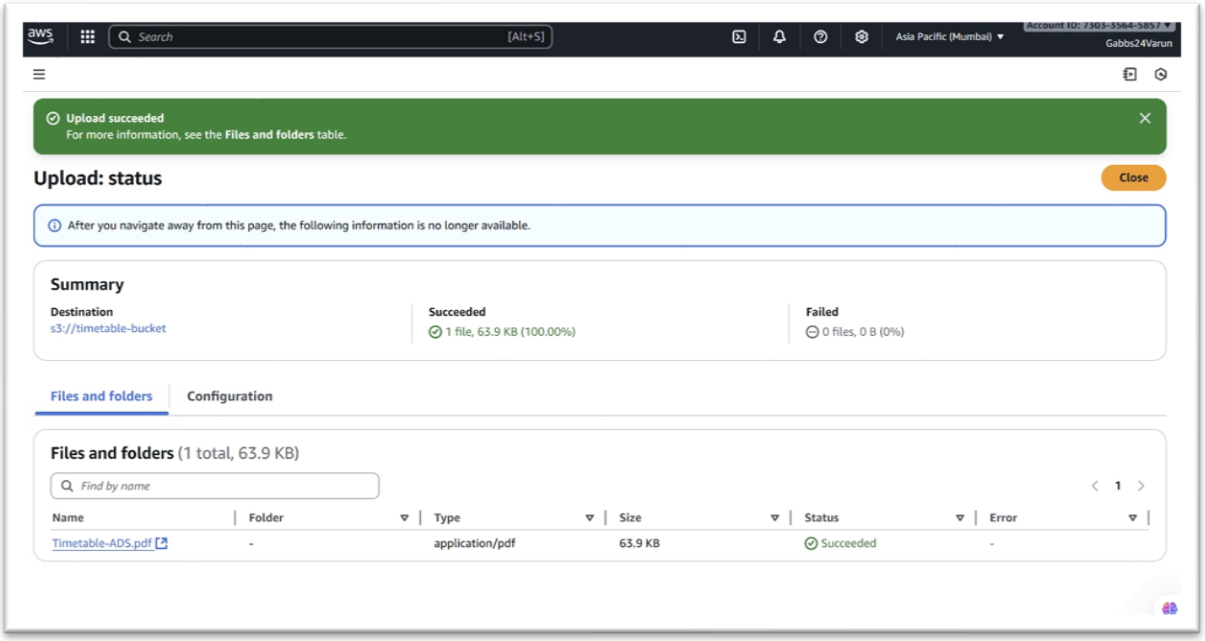

Amazon S3 > Buckets > timetable-bucket
timetable-bucket Info
Objects
Properties
Permissions
Metrics
Management
Access Points
Objects (0)
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Find objects by prefix
Name
Type
Last modified
Size
Storage class
No objects
You don't have any objects in this bucket.
Upload

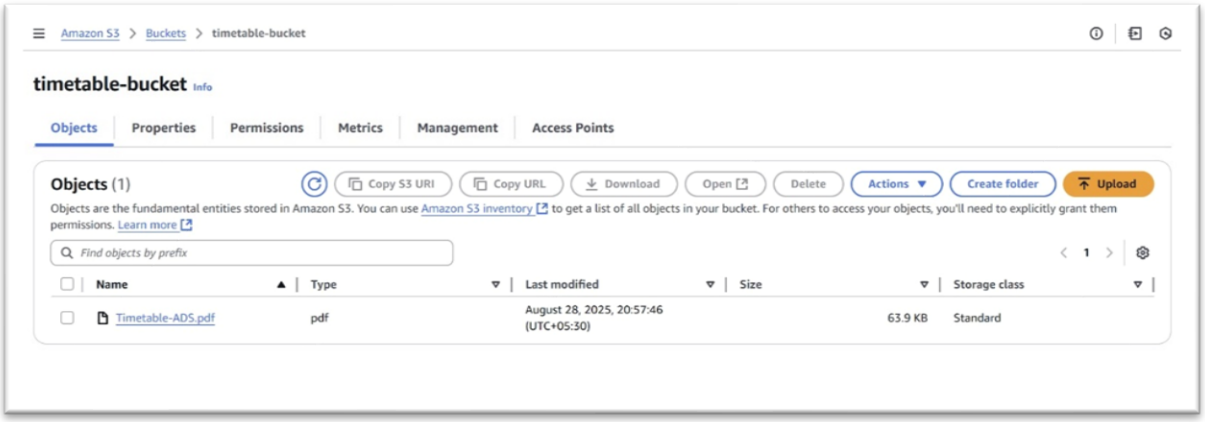

Amazon S3 > Buckets > timetable-bucket > Upload
Upload Info
Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDKs or Amazon S3 REST API. Learn more
Drag and drop files and folders you want to upload here, or choose Add files or Add folder.
Files and folders (0)
Remove
Add files
Add folder
All files and folders in this table will be uploaded.
Find by name
Name
Folder
Type
Size
No files or folders
You have not chosen any files or folders to upload.
Destination Info
Destination
s3://timetable-bucket
Destination details
Bucket settings that impact new objects stored in the specified destination.
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Downloads
Search Downloads
Organize
New folder
Gabriel - Personal
Attachments
Desktop
Documents
Pictures
Today
Timetable-ADS
Downloads
Documents
Chord-next
Desktop
Pictures
File name: Timetable-ADS
All Files
Open
Cancel
Asia Pacific (Mumbai)
Gabbs24Varun
SDKs or Amazon S3 REST API. Learn more
load here, or choose Add files or Add folder.
Remove
Add files
Add folder
Name
Folder
Type
Size
No files or folders
You have not chosen any files or folders to upload.
Destination Info
Destination
s3://timetable-bucket
Destination details
Bucket settings that impact new objects stored in the specified destination.
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Search
[Alt+S]
Asia Pacific (Mumbai)
Gabbs24Varun
Amazon S3 > Buckets > timetable-bucket > Upload
Upload Info
Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDKs or Amazon S3 REST API. Learn more
Drag and drop files and folders you want to upload here, or choose Add files or Add folder.
Files and folders (1 total, 63.9 KB)
All files and folders in this table will be uploaded.
Remove
Add files
Add folder
Find by name
Name
Folder
Type
Size
Timetable-ADS.pdf
-
application/pdf
63.9 KB
Destination Info
Destination
s3://timetable-bucket
Destination details
Bucket settings that impact new objects stored in the specified destination.
Permissions
Grant public access and access to other AWS accounts.


Destination Info
Destination
s3://timetable-bucket
Destination details
Bucket settings that impact new objects stored in the specified destination.
Permissions
Grant public access and access to other AWS accounts.
Properties
Specify storage class, encryption settings, tags, and more.
Cancel
Upload
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Search
[Alt+S]
Asia Pacific (Mumbai)
Gabbs24Varun
Upload succeeded
For more information, see the Files and folders table.
Upload: status
Close
After you navigate away from this page, the following information is no longer available.
Summary
Destination
s3://timetable-bucket
Succeeded
1 file, 63.9 KB (100.00%)
Failed
0 files, 0 B (0%)
Files and folders
Configuration
Files and folders (1 total, 63.9 KB)
Find by name
Name
Folder
Type
Size
Status
Error
Timetable-ADS.pdf
-
application/pdf
63.9 KB
Succeeded
-


Amazon S3
Buckets
timetable-bucket
timetable-bucket
Info
Objects
Properties
Permissions
Metrics
Management
Access Points
Objects (1)
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Find objects by prefix
Name
Type
Last modified
Size
Storage class
Timetable-ADS.pdf
pdf
August 28, 2025, 20:57:46 (UTC+05:30)
63.9 KB
Standard

RESULT:

Exp 5(i): **Generate ‘n’ Even Numbers and Deploy in Cloud**
Date:

Aim:

To write a Python program that generates ‘n’ even numbers and deploy it using AWS Lambda and EC2.

Python Code:

```
n = int(input("Enter a number: "))
print([i for i in range(2, 2*n+1, 2)])
```

A. Deployment Using AWS Lambda

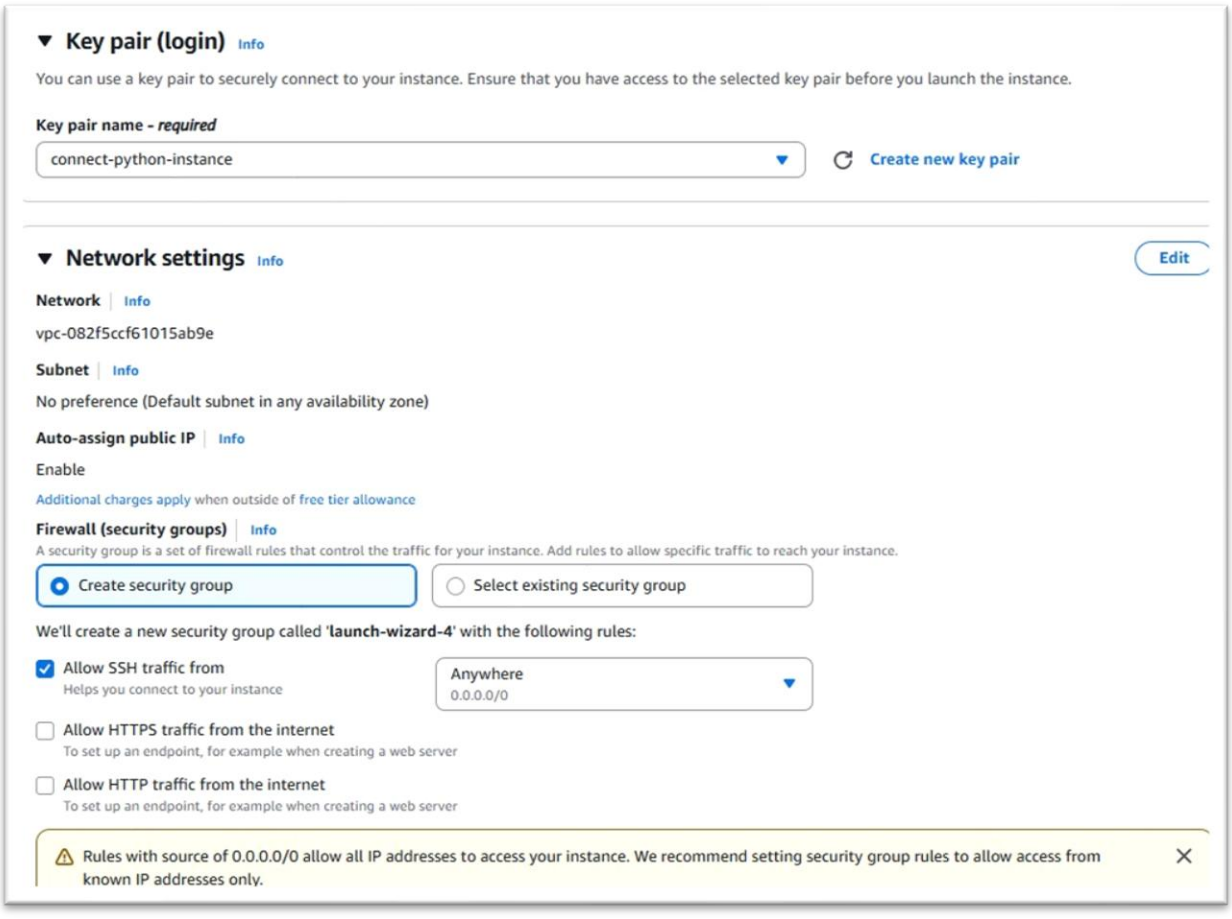
- Log in to AWS Console and go to Lambda service.
- Click 'Create function'.
- Choose 'Author from scratch', enter function name (e.g., GenerateEvenNumbers), and choose Python 3.x.
- Click 'Create function'.
- Scroll to 'Function code' > paste your Python code (use input replacement with event).
- Edit code to use event input like: n = event['n']
- Click 'Deploy'.
- Click 'Test', create test event with JSON: {"n": 5} and click 'Test'.
- Check the output in the execution results.

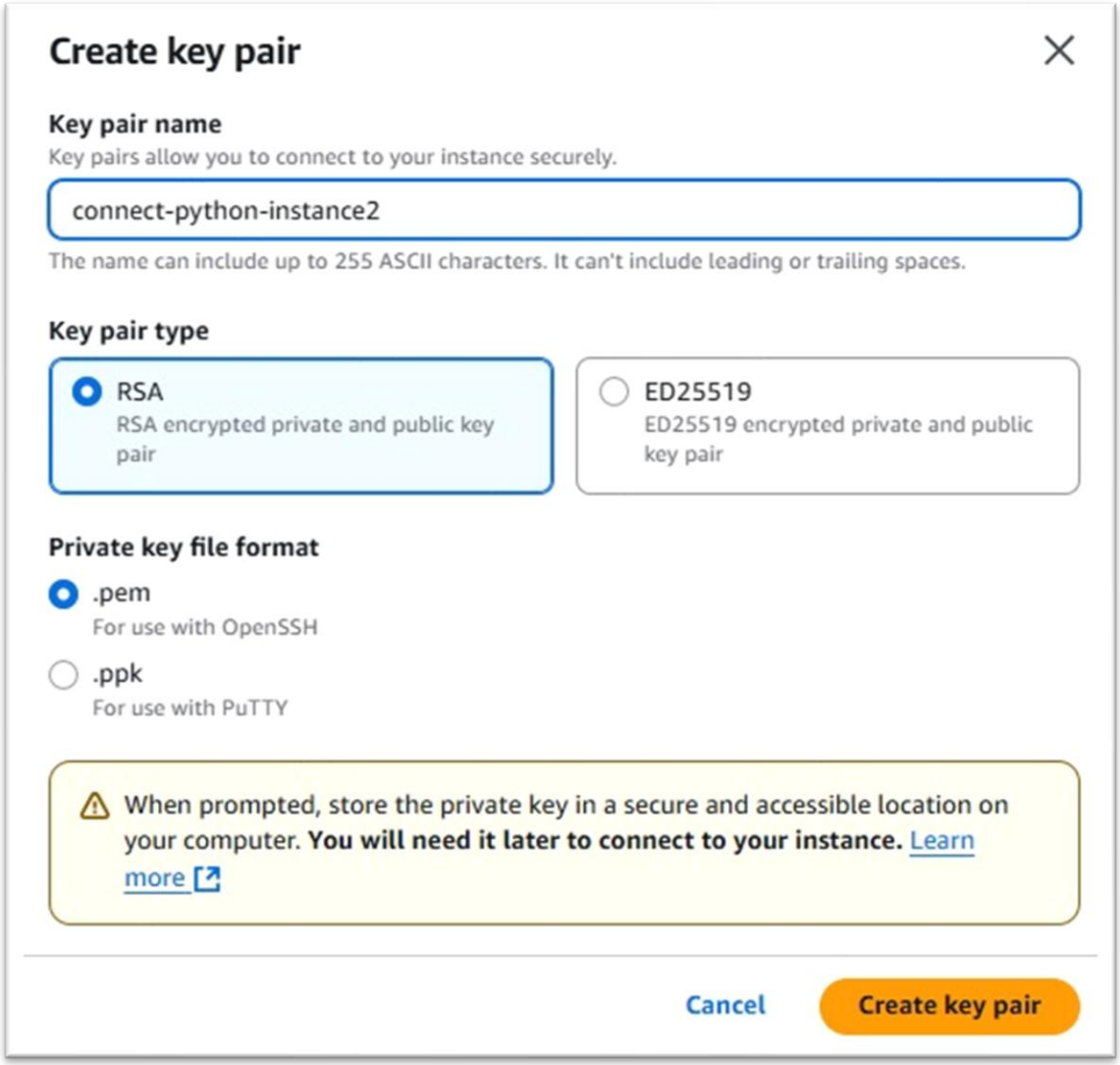
B. Deployment Using AWS EC2

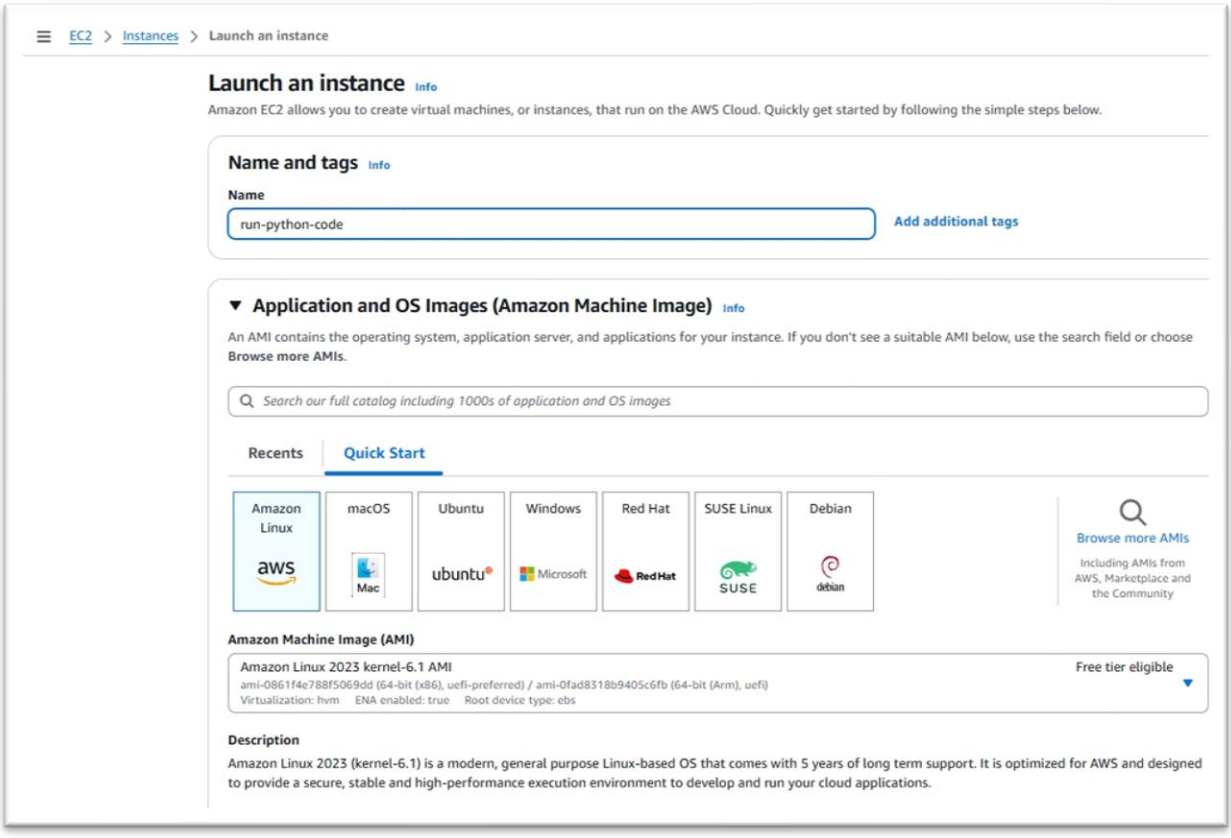
- Log in to AWS Console and go to EC2 service.
- Click 'Launch Instance'.
- Choose Amazon Linux or Ubuntu (Free tier).
- Choose t2.micro instance and click 'Next' until 'Launch'.
- Create a new key pair and download the PEM file.
- Open terminal or PowerShell and SSH into EC2:
  ssh -i yourkey.pem ec2-user@<public-ip>
- Install Python if not available (e.g., sudo yum install python3).
- Create a new file: nano even_numbers.py and paste the code.
- Run the program using: python3 even_numbers.py

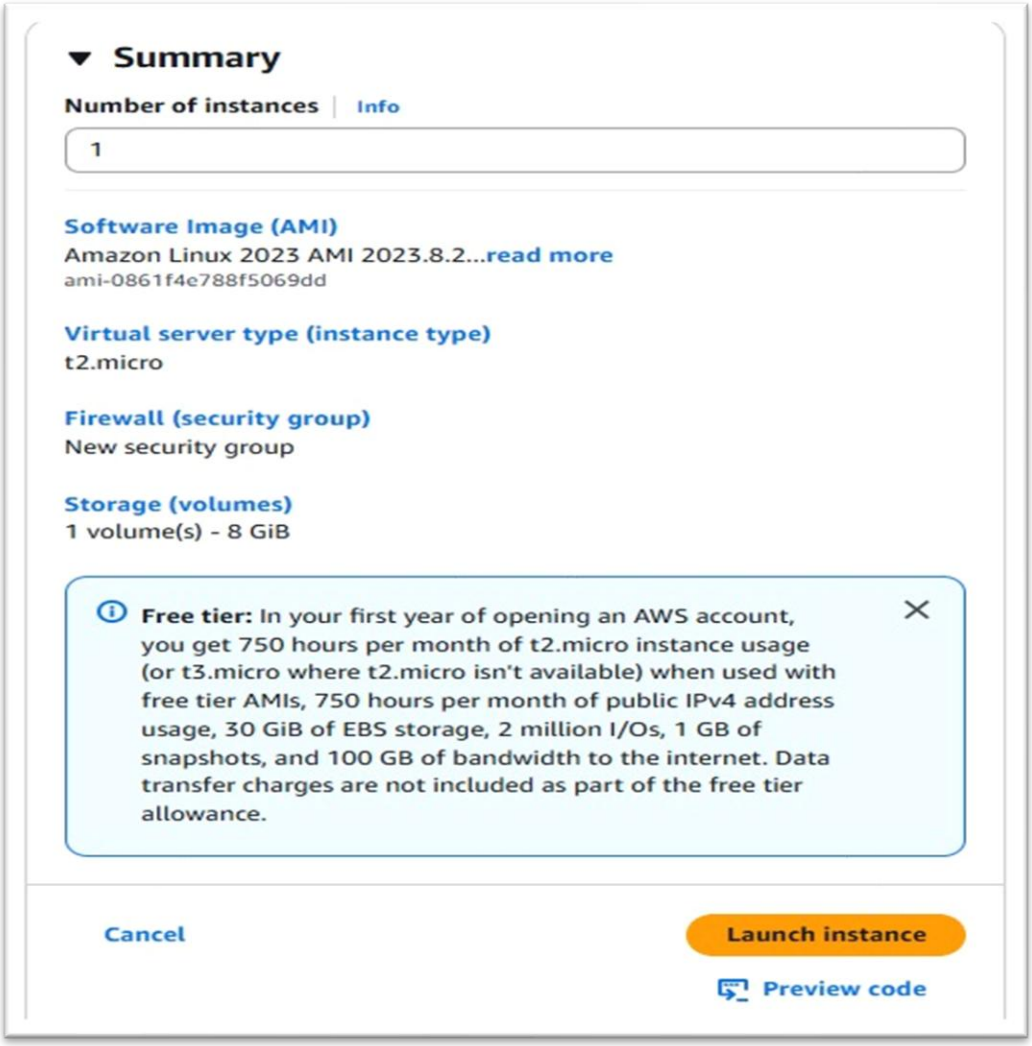
Output:

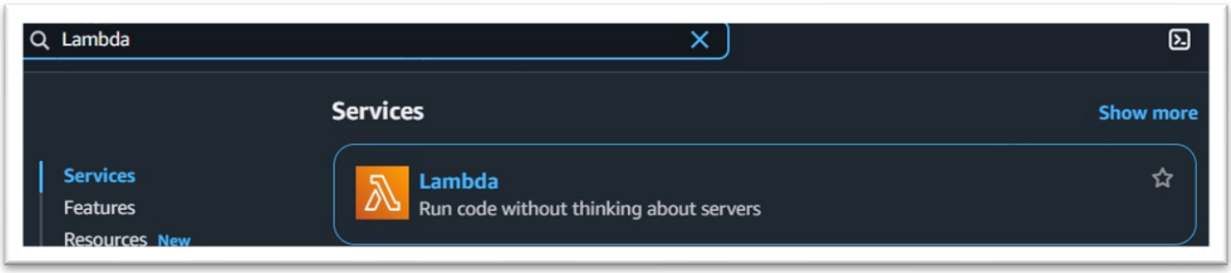

User data - optional | Info
Upload a file with your user data or enter it in the field.
Choose file
#!/bin/bash
# Update all packages
sudo yum update -y
# Install Python 3
sudo yum install -y python3
User data has already been base64 encoded

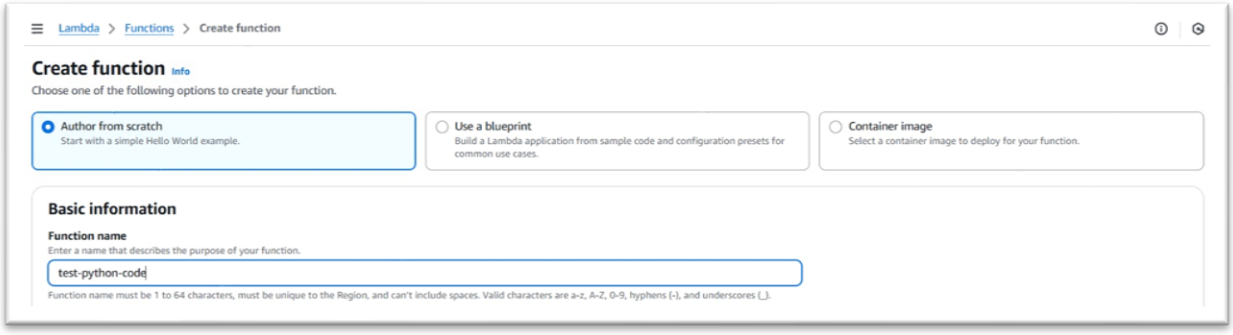

Key pair (login) Info
You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.
Key pair name - required
connect-python-instance
Create new key pair
Network settings Info
Edit
Network | Info
vpc-082f5ccf61015ab9e
Subnet | Info
No preference (Default subnet in any availability zone)
Auto-assign public IP | Info
Enable
Additional charges apply when outside of free tier allowance
Firewall (security groups) | Info
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
We'll create a new security group called 'launch-wizard-4' with the following rules:
Allow SSH traffic from
Helps you connect to your instance
Anywhere
0.0.0.0/0
Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server
Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server
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.

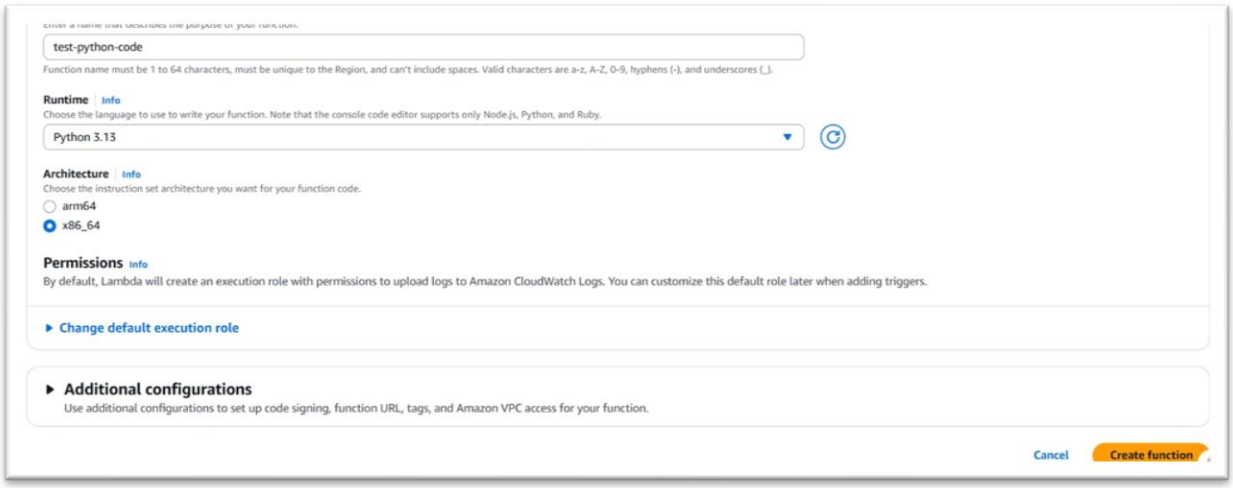

Create key pair
Key pair name
Key pairs allow you to connect to your instance securely.
connect-python-instance2
The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.
Key pair type
RSA
RSA encrypted private and public key pair
ED25519
ED25519 encrypted private and public key pair
Private key file format
.pem
For use with OpenSSH
.ppk
For use with PuTTY
When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. Learn more
Cancel
Create key pair

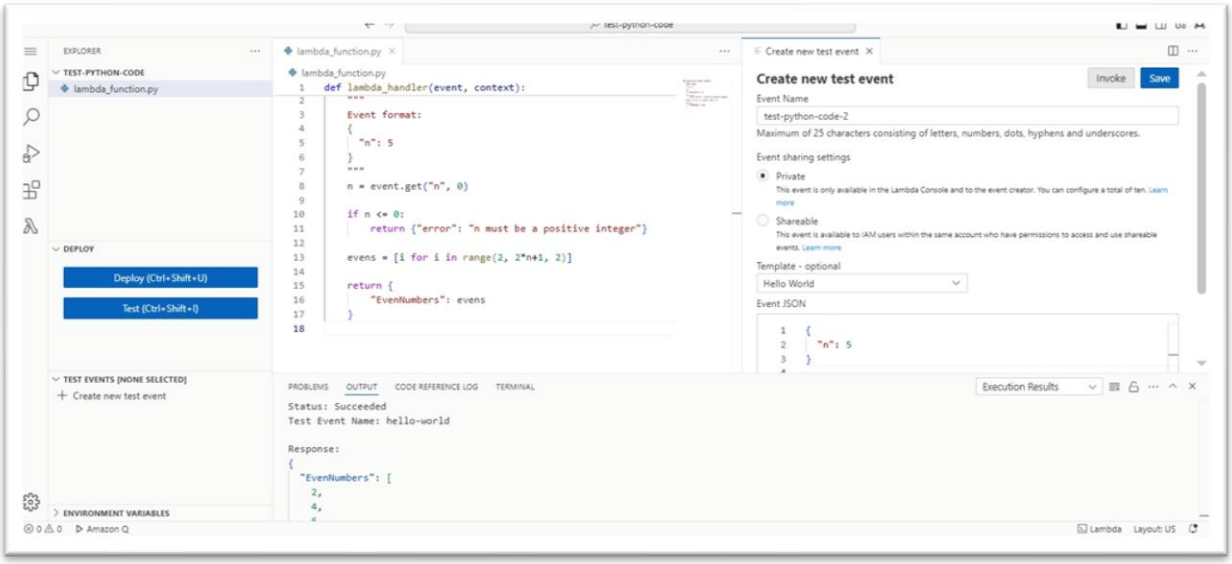

EC2 > Instances > Launch an instance
Launch an instance Info
Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.
Name and tags Info
Name
run-python-code
Add additional tags
Application and OS Images (Amazon Machine Image) Info
An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose Browse more AMIs.
Search our full catalog including 1000s of application and OS images
Recents
Quick Start
Amazon Linux
macOS
Ubuntu
Windows
Red Hat
SUSE Linux
Debian
Browse more AMIs
Including AMIs from AWS, Marketplace and the Community
Amazon Machine Image (AMI)
Amazon Linux 2023 kernel-6.1 AMI
Free tier eligible
ami-0861f4e788f5069dd (64-bit (x86), uefi-preferred) / ami-0fad8318b9405c6fb (64-bit (Arm), uefi)
Virtualization: hvm ENA enabled: true Root device type: ebs
Description
Amazon Linux 2023 (kernel-6.1) is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications.


Summary
Number of instances | Info
1
Software Image (AMI)
Amazon Linux 2023 AMI 2023.8.2...read more
ami-0861f4e788f5069dd
Virtual server type (instance type)
t2.micro
Firewall (security group)
New security group
Storage (volumes)
1 volume(s) - 8 GiB
Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included as part of the free tier allowance.
Cancel
Launch instance
Preview code


Lambda
Services
Show more
Services
Features
Resources New
Lambda
Run code without thinking about servers


Lambda > Functions > Create function
Create function Info
Choose one of the following options to create your function.
Author from scratch
Start with a simple Hello World example.
Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.
Container image
Select a container image to deploy for your function.
Basic information
Function name
Enter a name that describes the purpose of your function.
test-python-code
Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (_).





Deployment Using AWS EC2 :


```
[ec2-user@ip-172-31-15-57 ~]$ mkdir python-scripts
[ec2-user@ip-172-31-15-57 ~]$ ls
python-scripts
[ec2-user@ip-172-31-15-57 ~]$ touch generate-even-numbers.py
[ec2-user@ip-172-31-15-57 ~]$ vi generate-even-numbers.py
[ec2-user@ip-172-31-15-57 ~]$ vi generate-even-numbers.py
[ec2-user@ip-172-31-15-57 ~]$  75B written
[ec2-user@ip-172-31-15-57 ~]$ python3 generate-even-numbers.py
Enter a number: 5
[2, 4, 6, 8, 10]
[ec2-user@ip-172-31-15-57 ~]$
```

RESULT:

Exp 5(ii): **Display Nth Largest Number from List and Deploy**
Date:
Aim:

To write a Python program to find the nth largest number in a list and deploy using AWS Lambda or EC2.

Python Code:

```
def nth_largest(lst, n):
    return sorted(set(lst), reverse=True)[n-1]

lst = list(map(int, input("Enter numbers separated by space: ").split()))
n = int(input("Enter n: "))
print("Nth largest number:", nth_largest(lst, n))
```

A. Deployment Using AWS Lambda

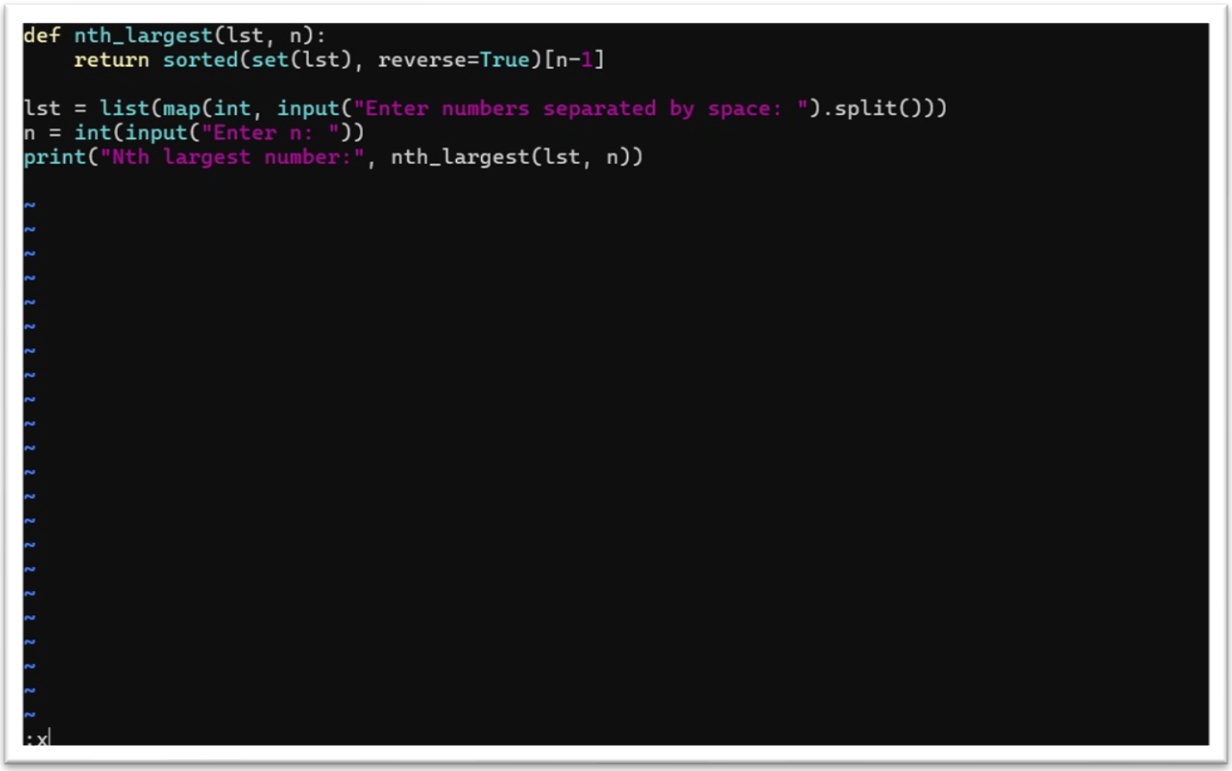
- Log in to AWS Console and go to Lambda service.
- Click 'Create function'.
- Choose 'Author from scratch', enter function name (e.g., GenerateEvenNumbers), and choose Python 3.x.
- Click 'Create function'.
- Scroll to 'Function code' > paste your Python code (use input replacement with event).
- Edit code to use event input like: n = event['n']
- Click 'Deploy'.
- Click 'Test', create test event with JSON: {"n": 5} and click 'Test'.
- Check the output in the execution results.

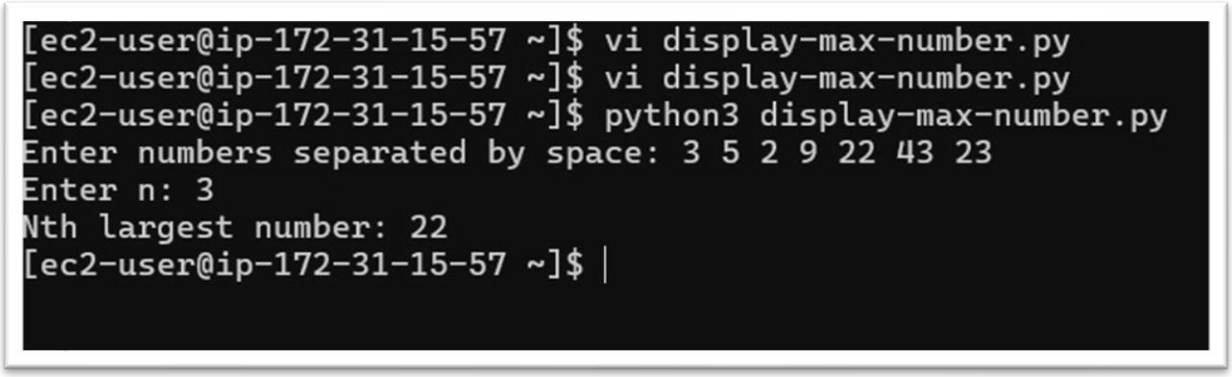
B. Deployment Using AWS EC2

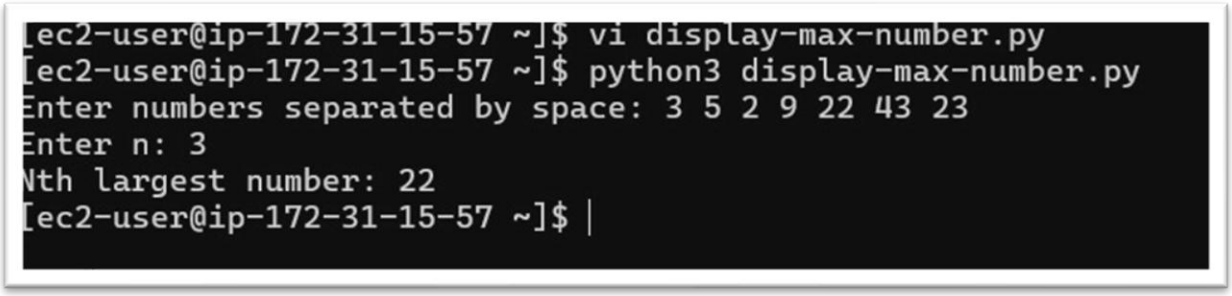
- Log in to AWS Console and go to EC2 service.
- Click 'Launch Instance'.
- Choose Amazon Linux or Ubuntu (Free tier).
- Choose t2.micro instance and click 'Next' until 'Launch'.
- Create a new key pair and download the PEM file.
- Open terminal or PowerShell and SSH into EC2:
  ssh -i yourkey.pem ec2-user@<public-ip>
- Install Python if not available (e.g., sudo yum install python3).
- Create a new file: nano even_numbers.py and paste the code.
- Run the program using: python3 even_numbers.py

Output:


def nth_largest(lst, n):
return sorted(set(lst), reverse=True)[n-1]
lst = list(map(int, input("Enter numbers separated by space: ").split()))
n = int(input("Enter n: "))
print("Nth largest number:", nth_largest(lst, n))
7,0-1
All


def nth_largest(lst, n):
return sorted(set(lst), reverse=True)[n-1]
lst = list(map(int, input("Enter numbers separated by space: ").split()))
n = int(input("Enter n: "))
print("Nth largest number:", nth_largest(lst, n))
:x


[ec2-user@ip-172-31-15-57 ~]$ vi display-max-number.py
[ec2-user@ip-172-31-15-57 ~]$ vi display-max-number.py
[ec2-user@ip-172-31-15-57 ~]$ python3 display-max-number.py
Enter numbers separated by space: 3 5 2 9 22 43 23
Enter n: 3
Nth largest number: 22
[ec2-user@ip-172-31-15-57 ~]$



Commands:

For connecting to server via SSH

1. Go to powershell

2. Note the EC2 Public IP address

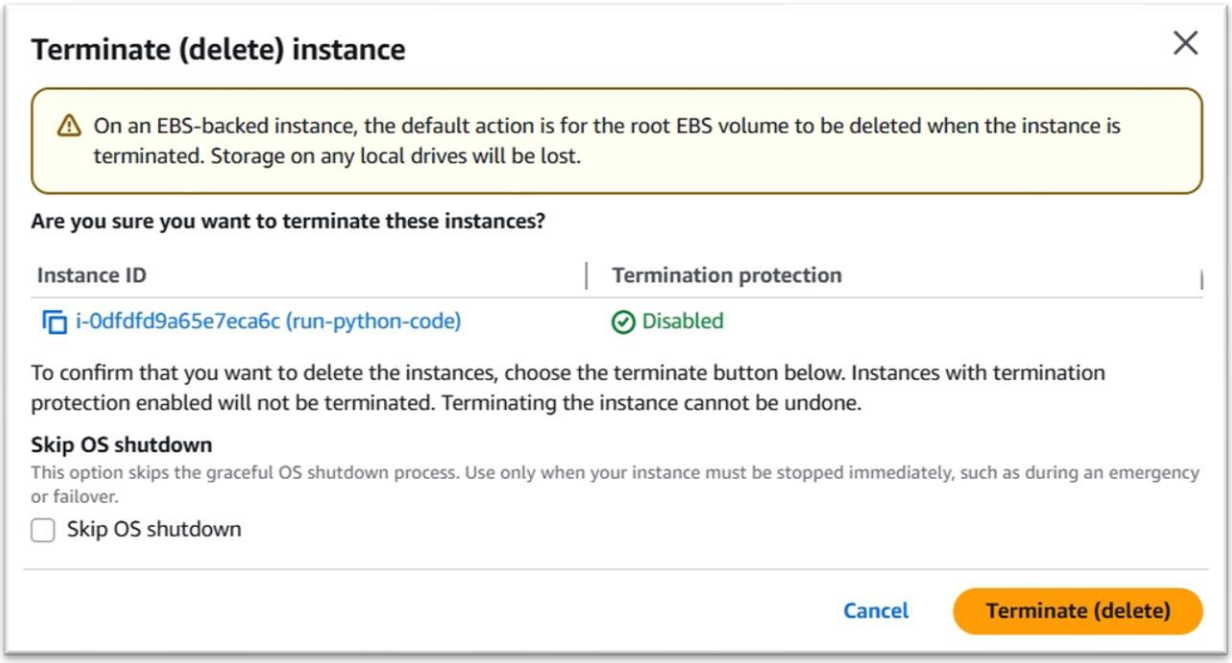
3. ssh -i "C:\path\to\your-key.pem" ec2-user@<PUBLIC_DNS_OR_IP>

- Check python version:

python3 --version

- Logout of EC2 machine:

exit



RESULT:

Exp 5(iii): User Validation and Database Login Storage

Date:

Aim:

To write a Python program to validate a user and store login credentials (username, password) in AWS DynamoDB using AWS Lambda and EC2.

Procedure:

- Log in to AWS Console.
- Go to DynamoDB > Create table.
- Table name: LoginTable | Partition key: username | Data type: String.
- Click 'Create'.
- Install Boto3 in your environment (EC2/Lambda) if needed.
- Python Code Example:

```
import boto3

def lambda_handler(event, context):
    dynamodb = boto3.resource('dynamodb')
    table = dynamodb.Table('LoginTable')

    username = event['username']
    password = event['password']

    table.put_item(Item={'username': username, 'password': password})
    return {"message": "User added successfully"}
```

Deployment Using AWS Lambda:

1. Log in to the AWS Console and navigate to the Lambda service.
2. Click 'Create function'. Choose 'Author from scratch'.
3. Set Function name (e.g., UserLoginStorage), choose Python 3.x runtime.
4. Click 'Create function'.
5. In the Function Code section, paste the following Python code:

```
import boto3

def lambda_handler(event, context):
    dynamodb = boto3.resource('dynamodb')
    table = dynamodb.Table('LoginTable')
```

```
    username = event['username']
    password = event['password']

    table.put_item(Item={'username': username, 'password': password})
    return {"message": "User added successfully"}
```

Click 'Deploy'.

Go to 'Test', create a test event like:

```
{
  "username": "john123",
  "password": "securepass"
}
```

Click 'Test' and check output under Execution Results.

Deployment Using AWS EC2 and CLI Login Setup:

1. Log in to the AWS Console and go to the EC2 service.
2. Click 'Launch Instance' and choose Amazon Linux 2 (Free tier eligible).
3. Choose t2.micro and configure defaults. Create and download a new key pair (e.g., mykey.pem).
4. Once the instance is running, copy its public IPv4 address.
5. Open a terminal (Linux/macOS) or PowerShell (Windows) and navigate to the PEM file location.
6. SSH into EC2 using the command:

```
ssh -i mykey.pem ec2-user@<your-ec2-public-ip>
```

Once logged in to EC2:

- Install Python: sudo yum install python3 -y

- Install pip: sudo yum install python3-pip -y

- Install boto3: pip3 install boto3

Create a Python file: nano user_validation.py

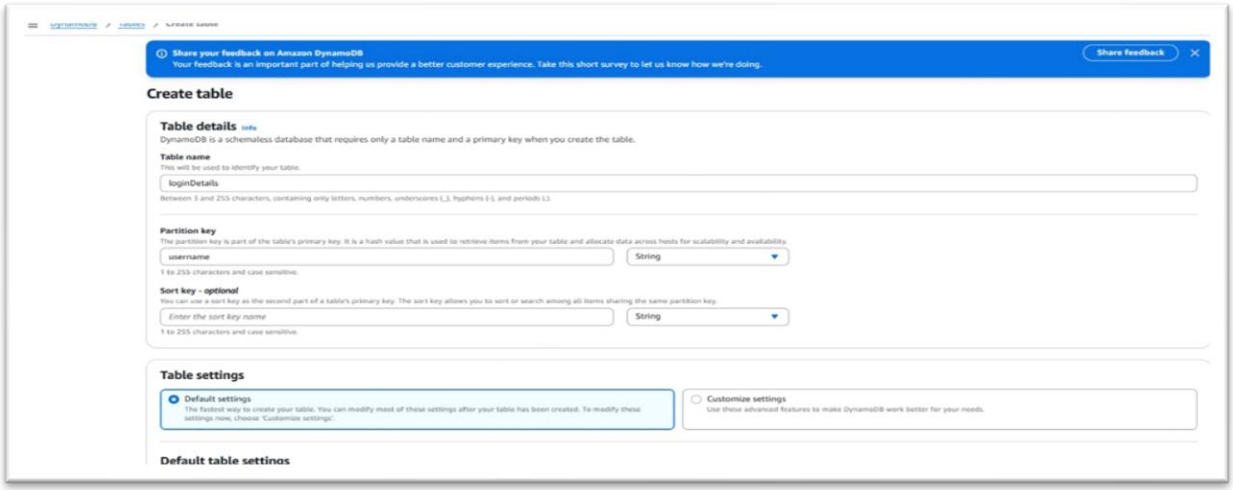
Paste the following code:

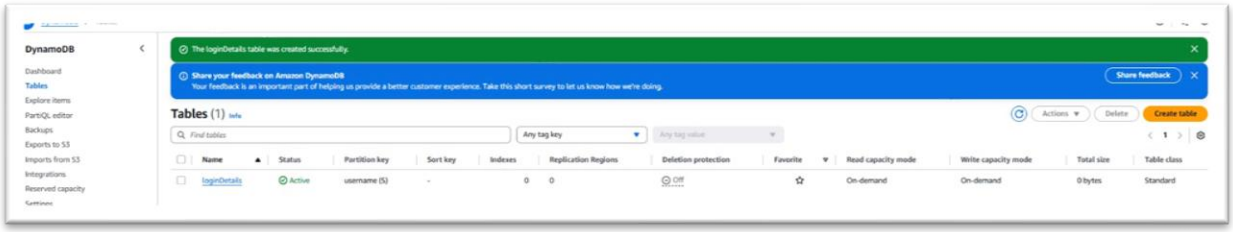
```
import boto3
dynamodb = boto3.resource('dynamodb')
table = dynamodb.Table('LoginTable')
username = input("Enter username: ")
password = input("Enter password: ")
   table.put_item(Item={'username': username, 'password': password})
print("User added successfully.")
```

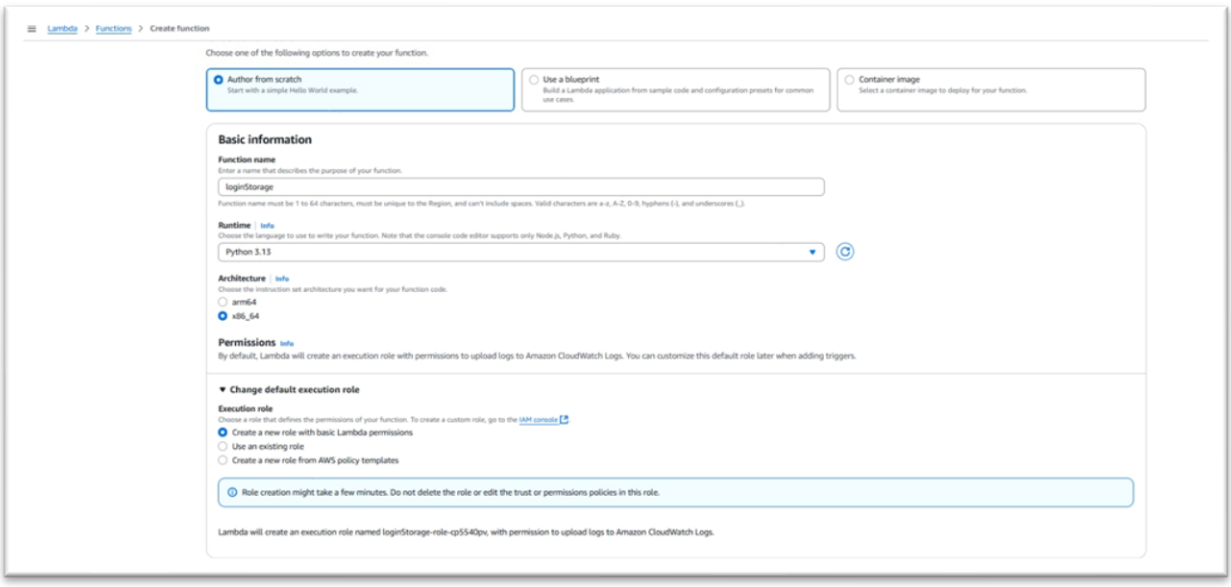
Run the code using: python3 user_validation.py

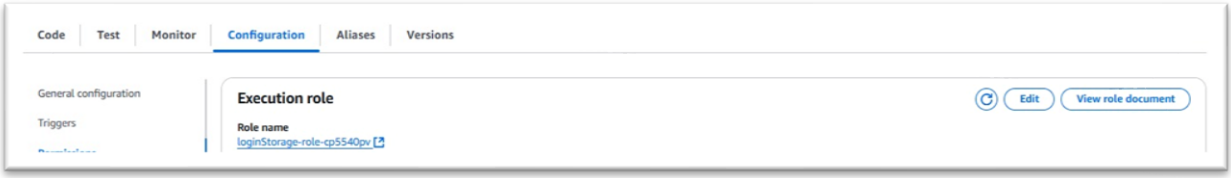
Ensure the IAM role or instance profile has DynamoDB access permissions.

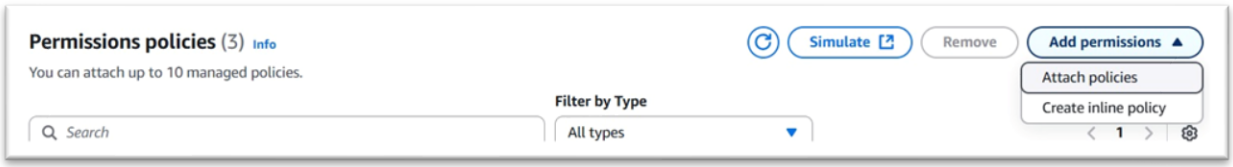
Output:

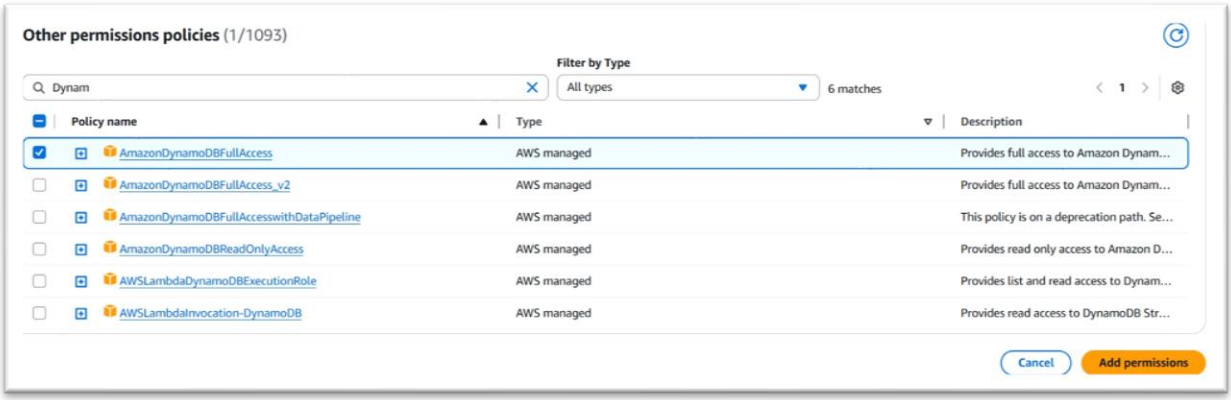


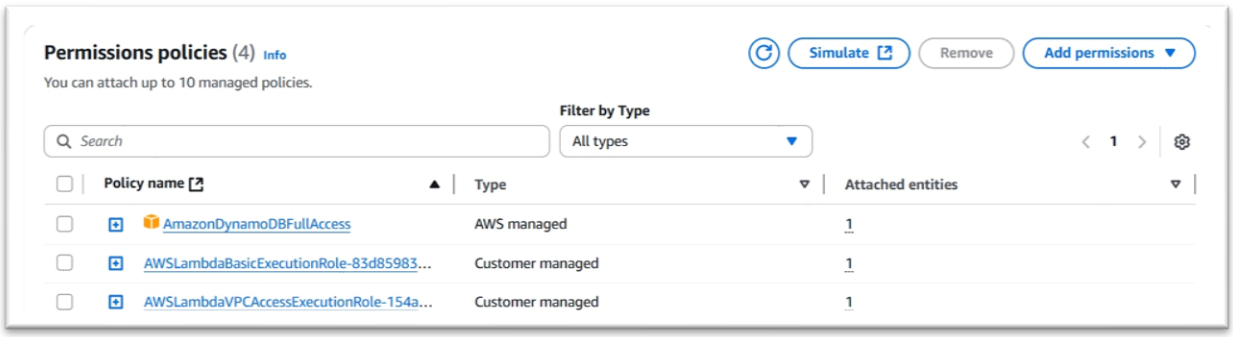


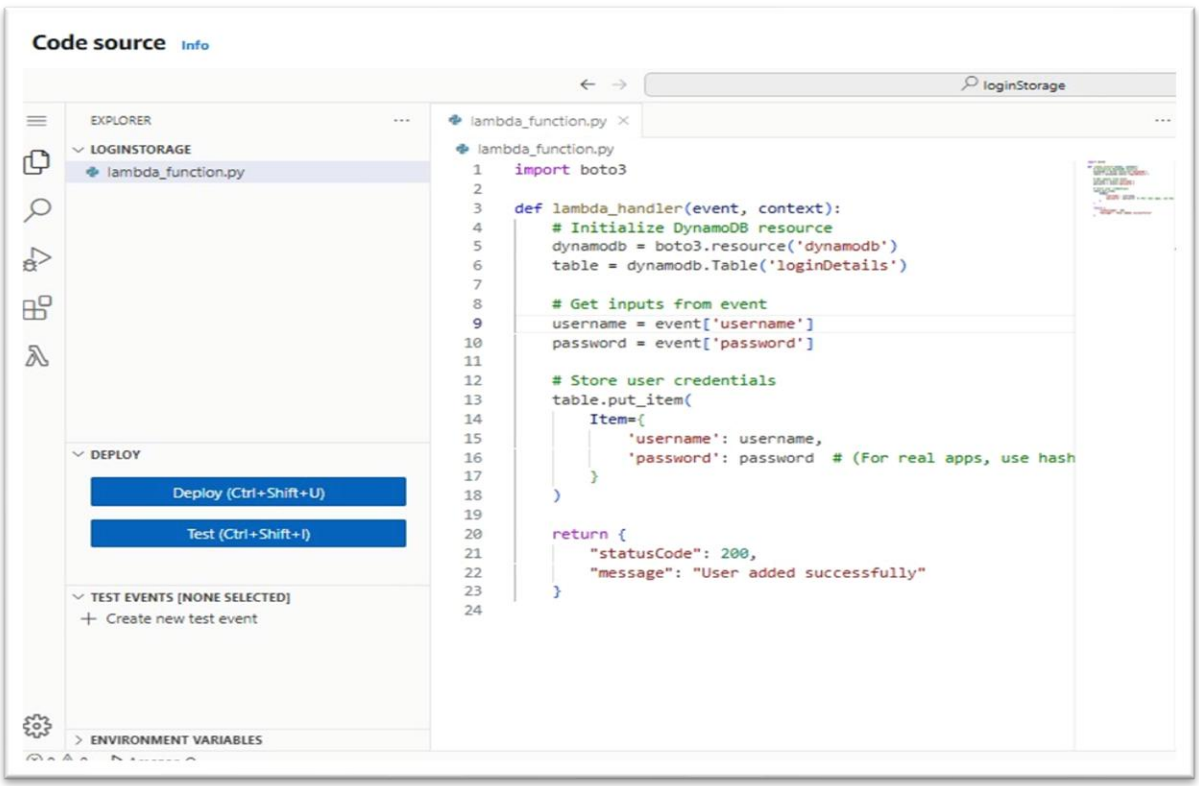


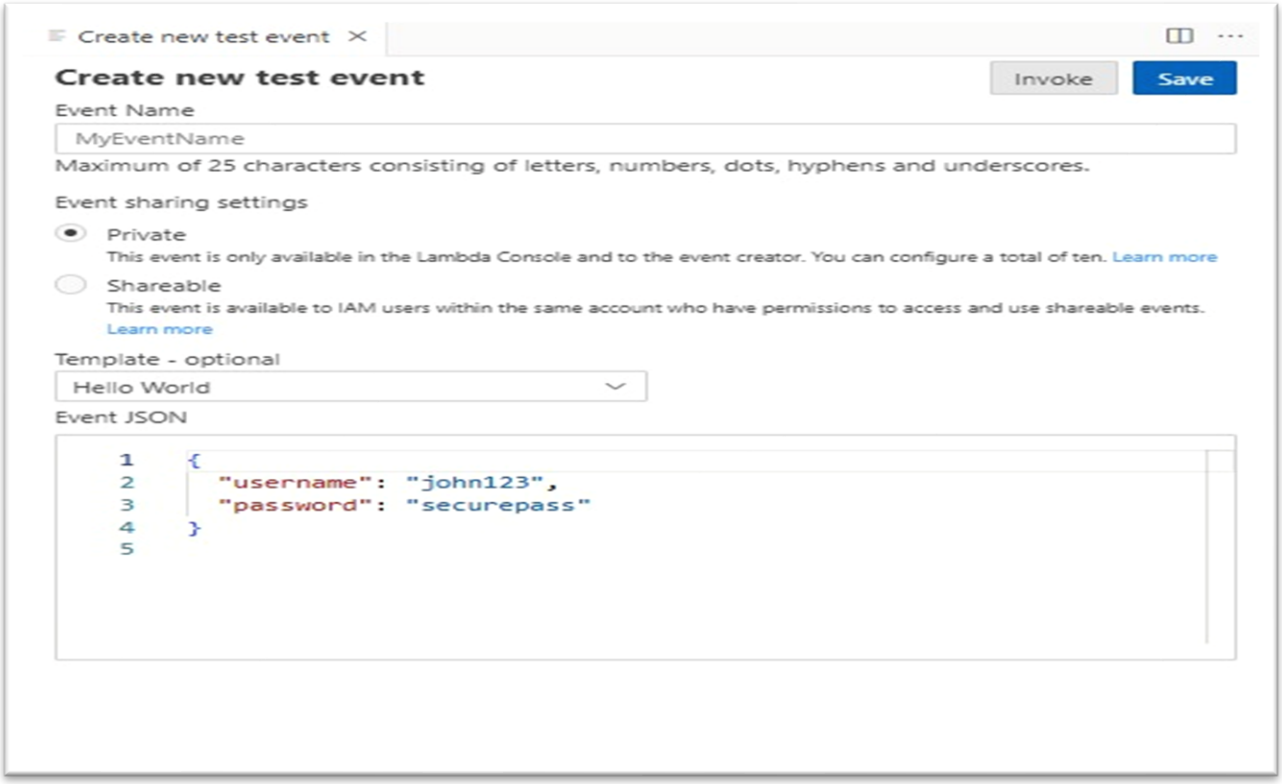



Permissions policies (3) Info
You can attach up to 10 managed policies.
Simulate
Remove
Add permissions
Attach policies
Create inline policy
Filter by Type
Search
All types
1


Other permissions policies (1/1093)
Filter by Type
Dynam
All types
6 matches
1
Policy name
Type
Description
AmazonDynamoDBFullAccess
AWS managed
Provides full access to Amazon Dynam...
AmazonDynamoDBFullAccess_v2
AWS managed
Provides full access to Amazon Dynam...
AmazonDynamoDBFullAccesswithDataPipeline
AWS managed
This policy is on a deprecation path. Se...
AmazonDynamoDBReadOnlyAccess
AWS managed
Provides read only access to Amazon D...
AWSLambdaDynamoDBExecutionRole
AWS managed
Provides list and read access to Dynam...
AWSLambdaInvocation-DynamoDB
AWS managed
Provides read access to DynamoDB Str...
Cancel
Add permissions


Permissions policies (4) Info
You can attach up to 10 managed policies.
Simulate
Remove
Add permissions
Filter by Type
Search
All types
1
Policy name
Type
Attached entities
AmazonDynamoDBFullAccess
AWS managed
1
AWSLambdaBasicExecutionRole-83d85983...
Customer managed
1
AWSLambdaVPCAccessExecutionRole-154a...
Customer managed
1


Code source Info
loginStorage
EXPLORER
LOGINSTORAGE
lambda_function.py
DEPLOY
Deploy (Ctrl+Shift+U)
Test (Ctrl+Shift+I)
TEST EVENTS [NONE SELECTED]
Create new test event
ENVIRONMENT VARIABLES
lambda_function.py
import boto3
def lambda_handler(event, context):
# Initialize DynamoDB resource
dynamodb = boto3.resource('dynamodb')
table = dynamodb.Table('loginDetails')
# Get inputs from event
username = event['username']
password = event['password']
# Store user credentials
table.put_item(
Item={
'username': username,
'password': password # (For real apps, use hash
}
)
return {
"statusCode": 200,
"message": "User added successfully"
}


Create new test event
Create new test event
Invoke
Save
Event Name
MyEventName
Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.
Event sharing settings
Private
This event is only available in the Lambda Console and to the event creator. You can configure a total of ten. Learn more
Shareable
This event is available to IAM users within the same account who have permissions to access and use shareable events. Learn more
Template - optional
Hello World
Event JSON
{
"username": "john123",
"password": "securepass"
}

RESULT:

Exp 6: **Running Virtual Machines of Different Configurations**
Date:

Aim:

To launch and monitor virtual machines (EC2 instances) of different configurations and count the number of instances that can run concurrently within AWS free tier limits or account quota.

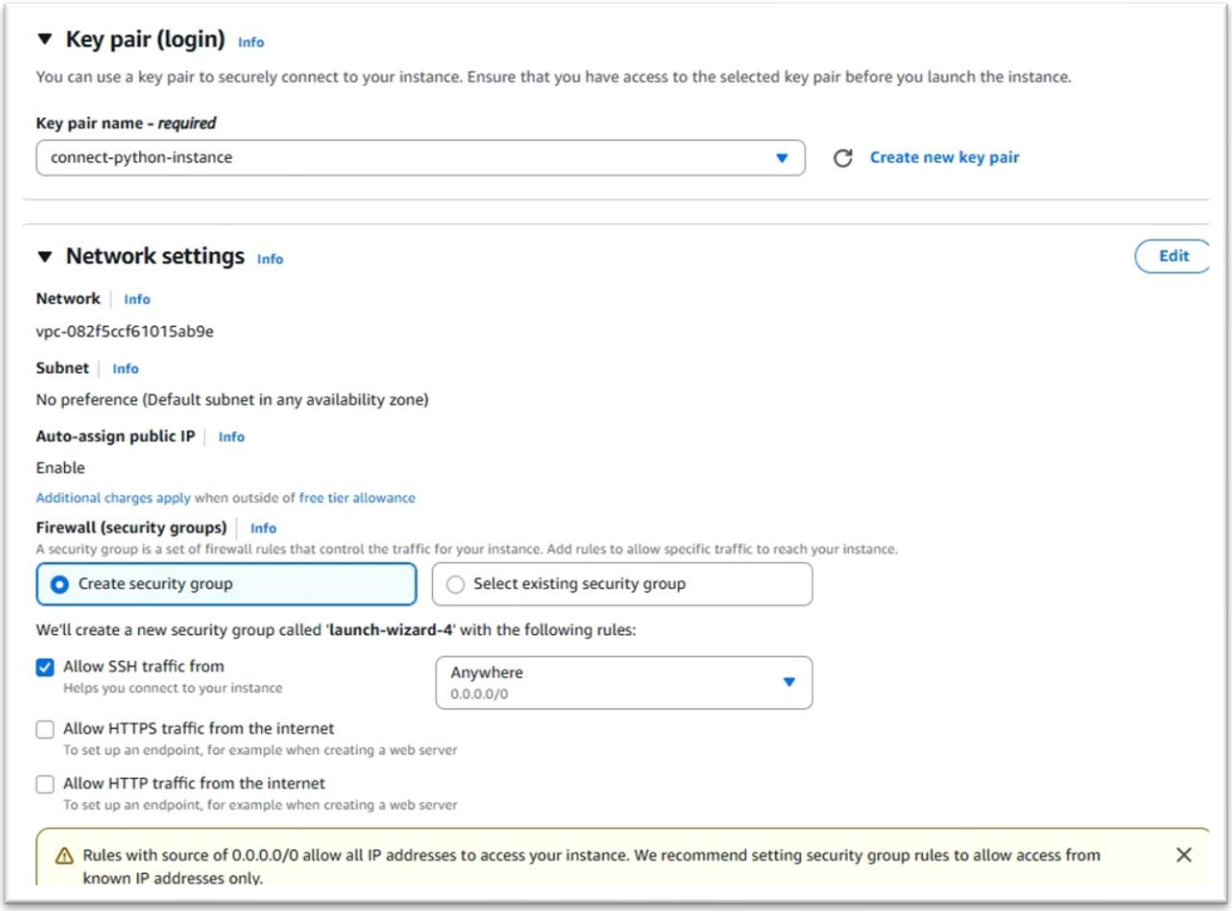
Procedure:

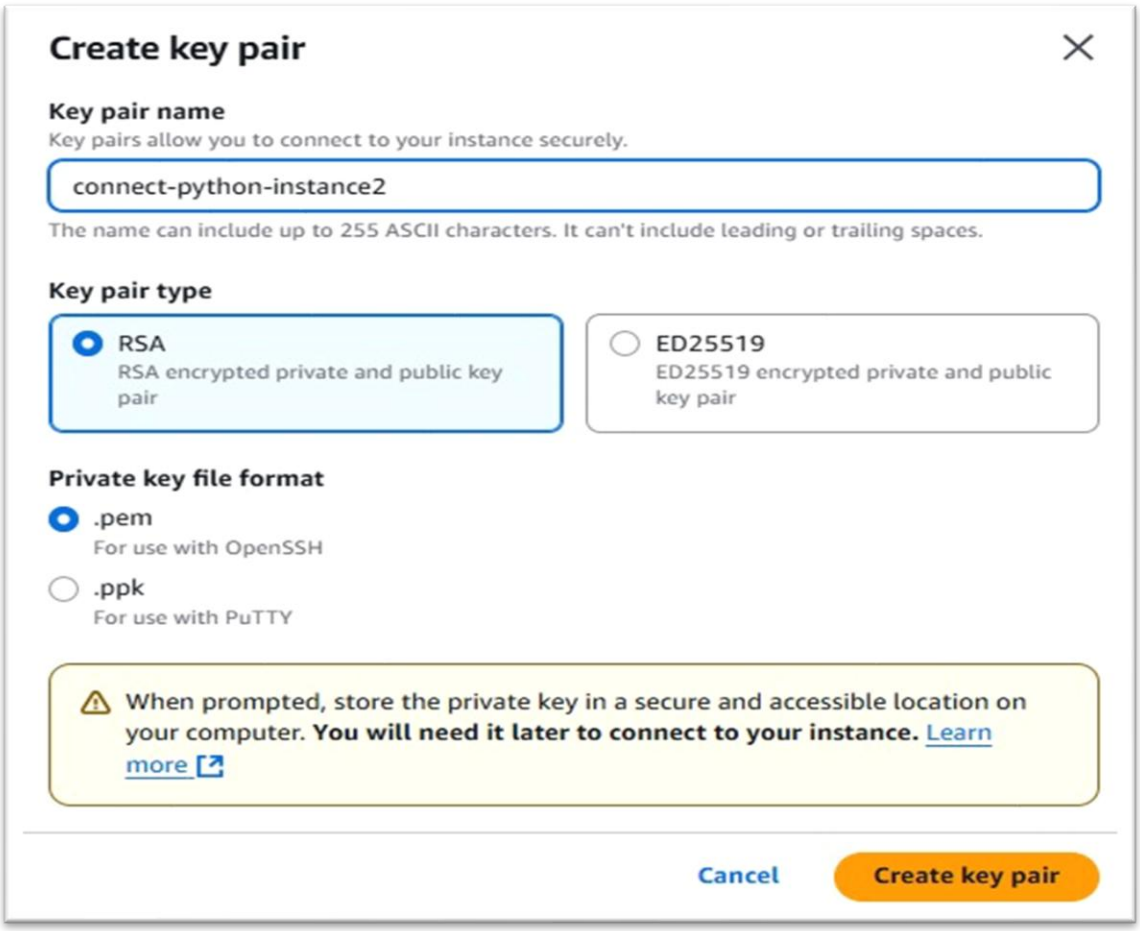
- - Log in to AWS Console.
- - Navigate to EC2 and click 'Launch Instance'.
- - Choose different AMIs and instance types (e.g., t2.micro, t3.micro).
- - Allocate names and security groups for each instance.
- - Monitor resource usage using CloudWatch.
- - Repeat until instance limit is reached.
- - Note any warnings or errors due to limit.

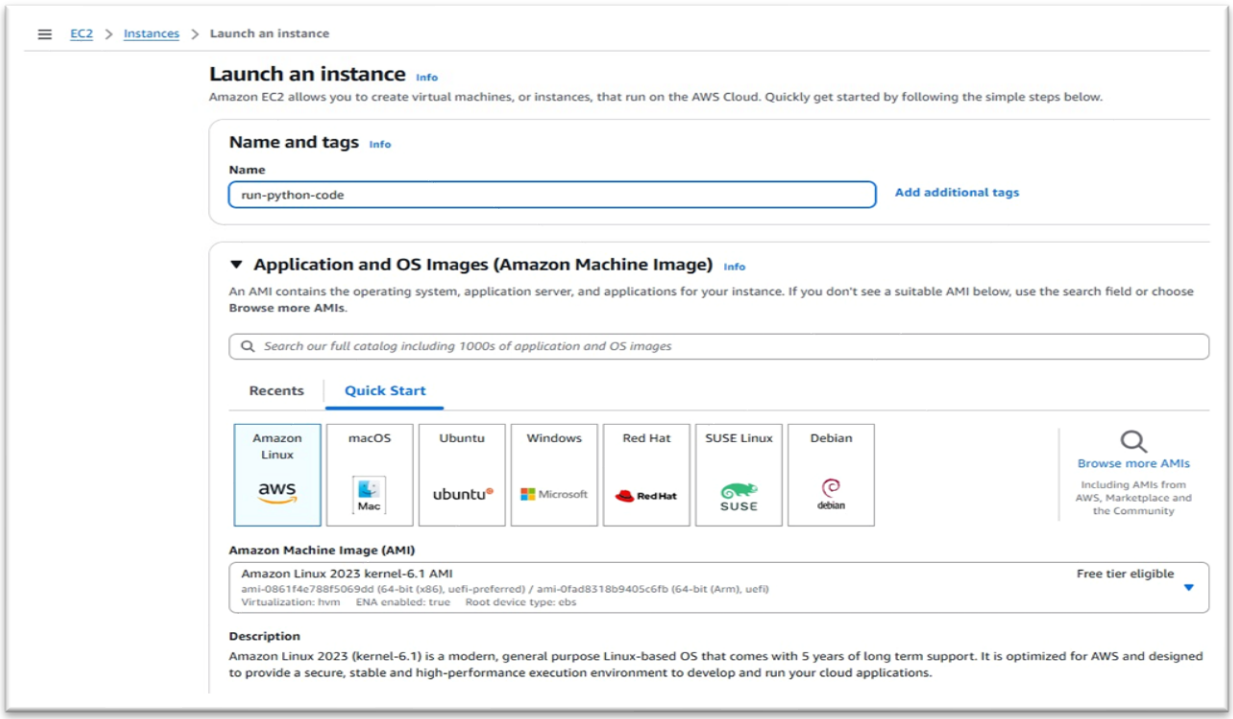
Output:

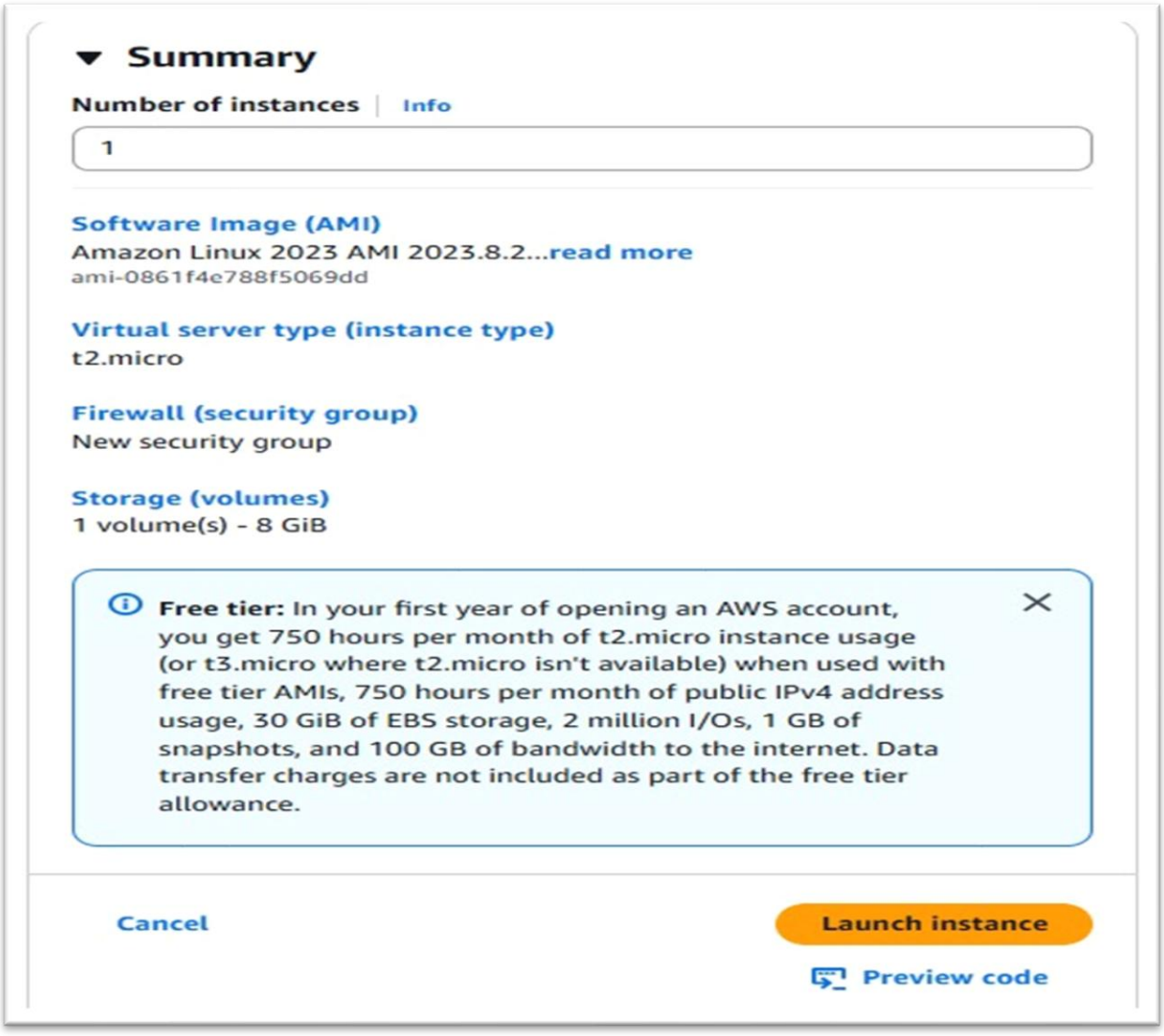
Instance 1 - Amazon Linux OS type and AMI

Launch instructions


Key pair (login) Info
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
connect-python-instance
Create new key pair
Network settings Info
Edit
Network Info
vpc-082f5ccf61015ab9e
Subnet Info
No preference (Default subnet in any availability zone)
Auto-assign public IP Info
Enable
Additional charges apply when outside of free tier allowance
Firewall (security groups) Info
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
We'll create a new security group called 'launch-wizard-4' with the following rules:
Allow SSH traffic from
Helps you connect to your instance
Anywhere
0.0.0.0/0
Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server
Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server
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.

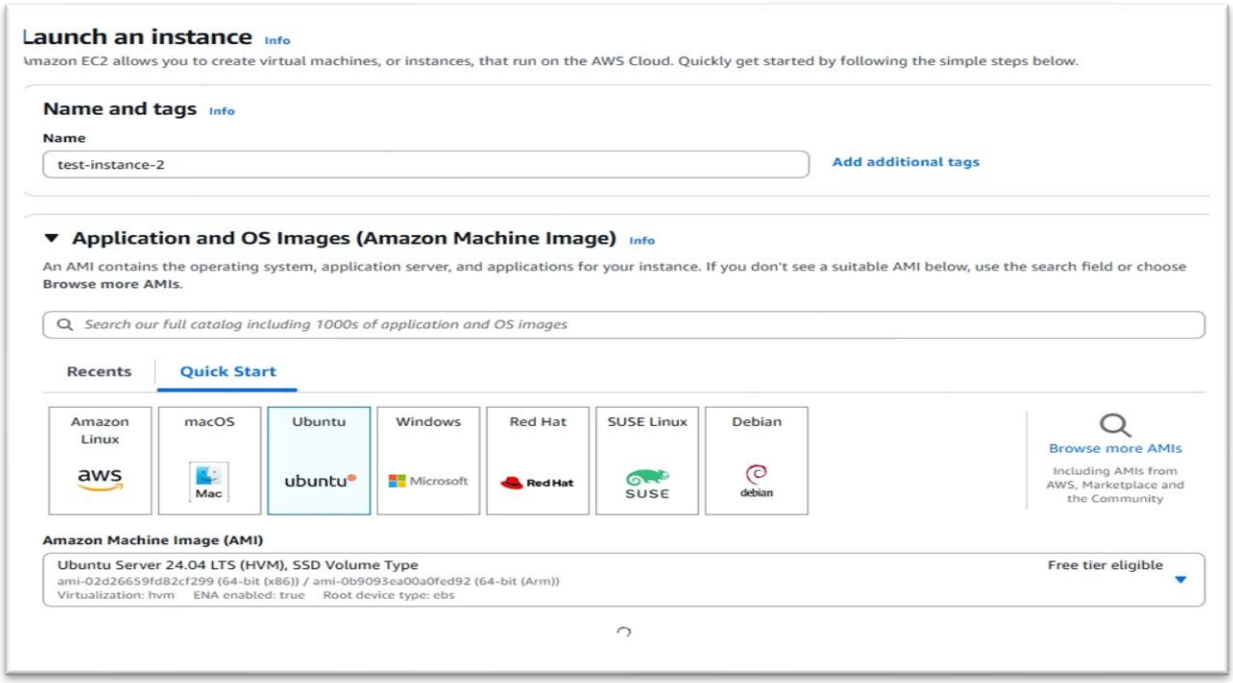

Create key pair
Key pair name
Key pairs allow you to connect to your instance securely.
connect-python-instance2
The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.
Key pair type
RSA
RSA encrypted private and public key pair
ED25519
ED25519 encrypted private and public key pair
Private key file format
.pem
For use with OpenSSH
.ppk
For use with PuTTY
When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. Learn more
Cancel
Create key pair

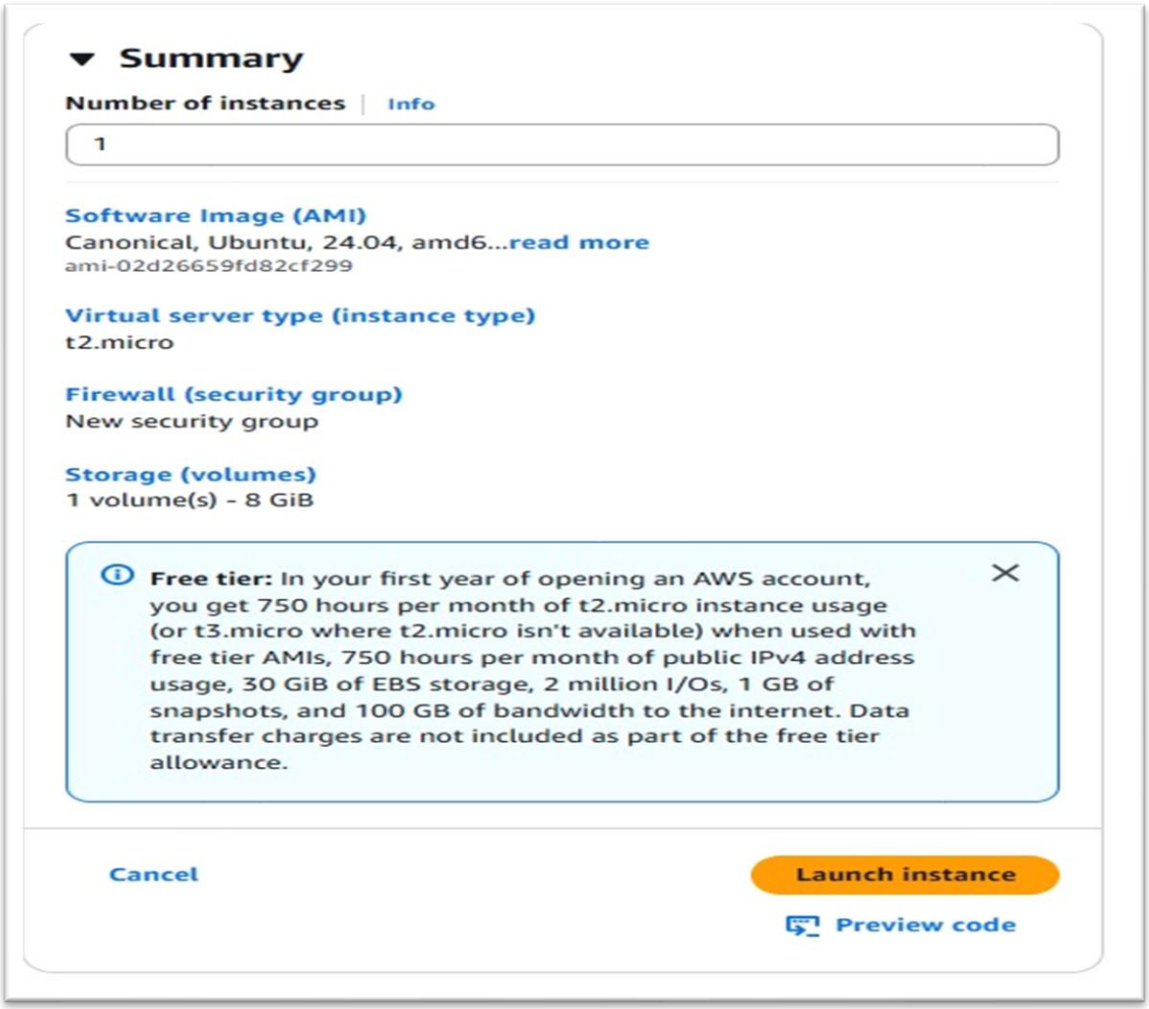

EC2 > Instances > Launch an instance
Launch an instance Info
Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.
Name and tags Info
Name
run-python-code
Add additional tags
Application and OS Images (Amazon Machine Image) Info
An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose Browse more AMIs.
Search our full catalog including 1000s of application and OS images
Recents
Quick Start
Amazon Linux
macOS
Ubuntu
Windows
Red Hat
SUSE Linux
Debian
Browse more AMIs
Including AMIs from AWS, Marketplace and the Community
Amazon Machine Image (AMI)
Amazon Linux 2023 kernel-6.1 AMI
ami-0861f4e788f5069dd (64-bit (x86), uefi-preferred) / ami-0fad8318b9405c6fb (64-bit (Arm), uefi)
Virtualization: hvm ENA enabled: true Root device type: ebs
Free tier eligible
Description
Amazon Linux 2023 (kernel-6.1) is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications.

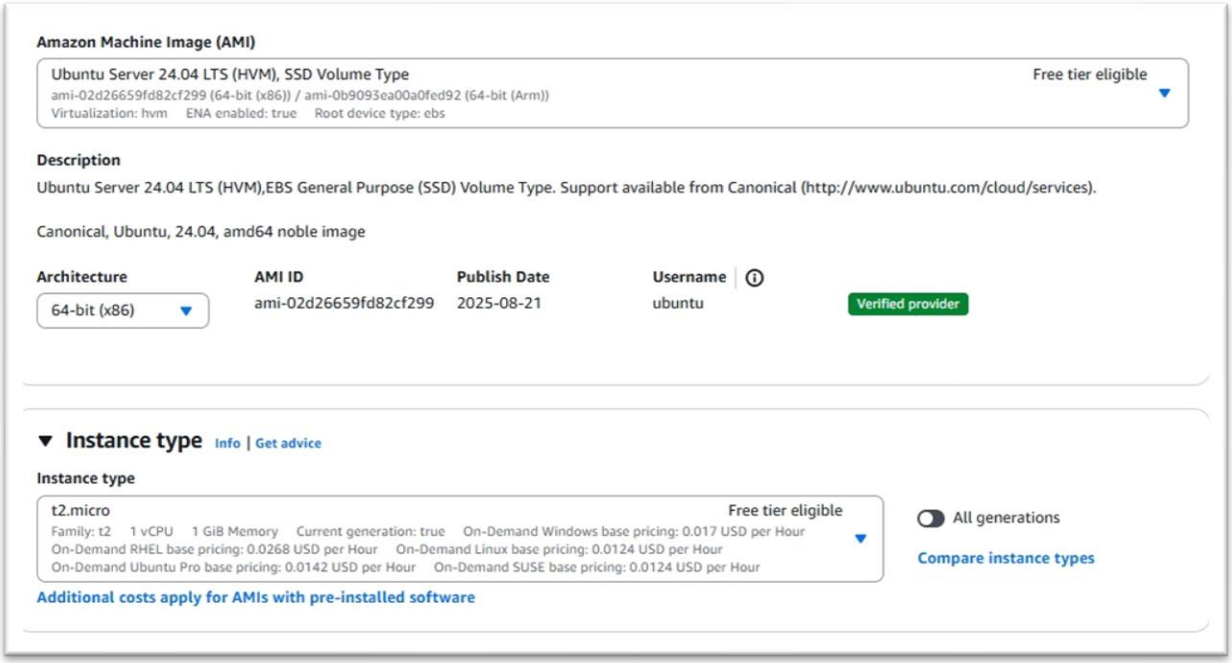

Summary
Number of instances Info
1
Software Image (AMI)
Amazon Linux 2023 AMI 2023.8.2...read more
ami-0861f4e788f5069dd
Virtual server type (instance type)
t2.micro
Firewall (security group)
New security group
Storage (volumes)
1 volume(s) - 8 GiB
Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included as part of the free tier allowance.
Cancel
Launch instance
Preview code

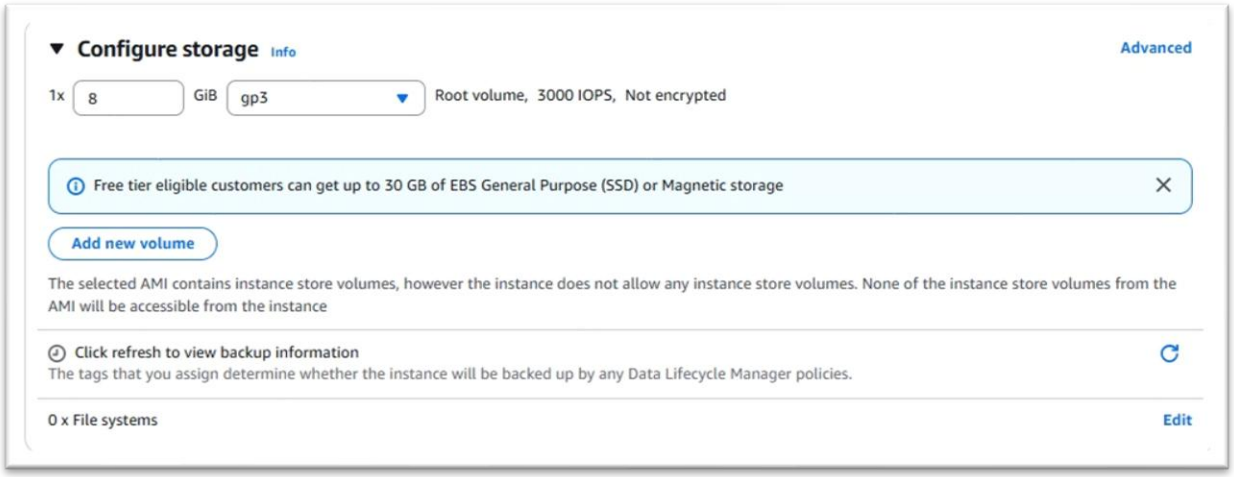
Instance 2

Ubuntu OS type and AMI






Amazon Machine Image (AMI)
Ubuntu Server 24.04 LTS (HVM), SSD Volume Type
Free tier eligible
ami-02d26659fd82cf299 (64-bit (x86)) / ami-0b9093ea00a0fed92 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs
Description
Ubuntu Server 24.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).
Canonical, Ubuntu, 24.04, amd64 noble image
Architecture
AMI ID
Publish Date
Username
64-bit (x86)
ami-02d26659fd82cf299
2025-08-21
ubuntu
Verified provider
Instance type
Info | Get advice
Instance type
t2.micro
Free tier eligible
Family: t2 1 vCPU 1 GiB Memory Current generation: true On-Demand Windows base pricing: 0.017 USD per Hour
On-Demand RHEL base pricing: 0.0268 USD per Hour On-Demand Linux base pricing: 0.0124 USD per Hour
On-Demand Ubuntu Pro base pricing: 0.0142 USD per Hour On-Demand SUSE base pricing: 0.0124 USD per Hour
All generations
Compare instance types
Additional costs apply for AMIs with pre-installed software


Configure storage
Info
Advanced
1x
8
GiB
gp3
Root volume, 3000 IOPS, 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 instance store volumes, however the instance does not allow any instance store volumes. None of the 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.
0 x File systems
Edit

RESULT:

Ex: 8 **Create your own Virtual Private Cloud (VPC)**
Date:

AIM:

PROCEDURE:
A step-by-step guide to creating a VPC in **AWS**

Step 1: Log in to the AWS Management Console

1. Navigate to AWS console.
2. Log in using your AWS credentials.

Step 2: Access the VPC Dashboard

1. Once logged in, type **VPC** in the search bar at the top of the AWS Management Console.
2. Click on **VPC** under the **Networking & Content Delivery** section.
3. This will take you to the **VPC Dashboard**, where you can manage your VPCs.

Step 3: Create a New VPC

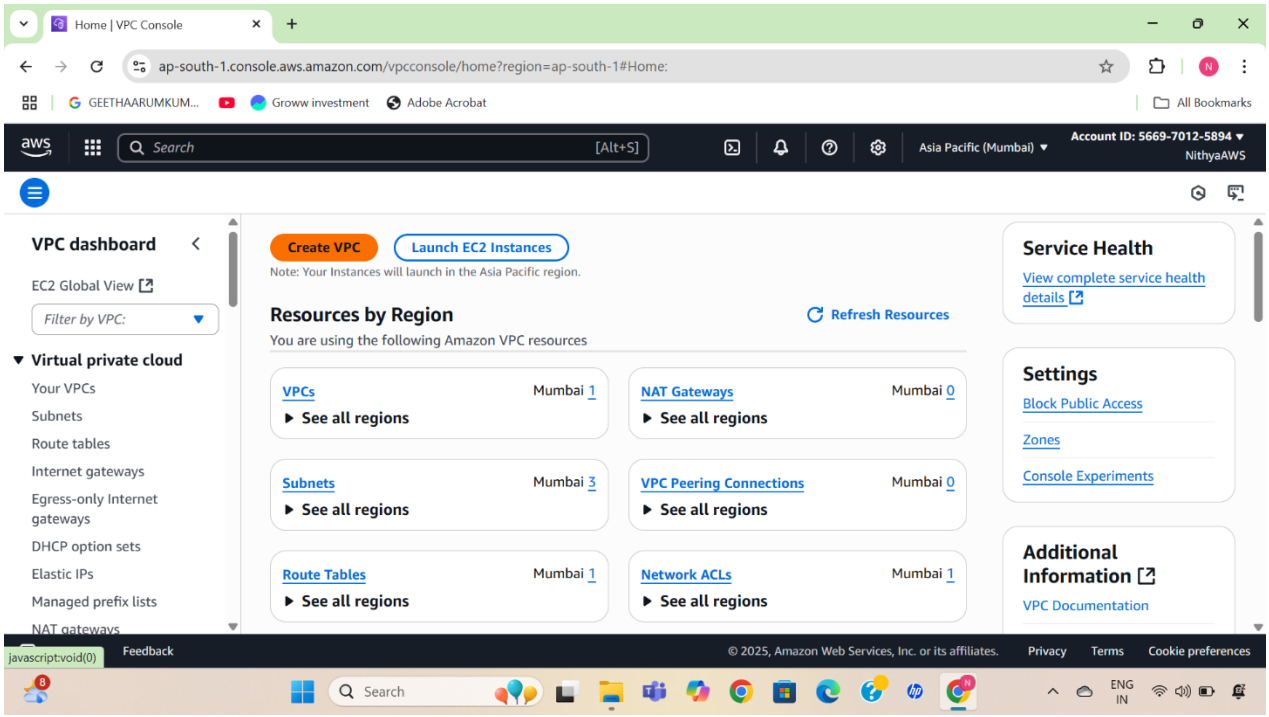
1. In the **VPC Dashboard**, click on the **Create VPC** button.

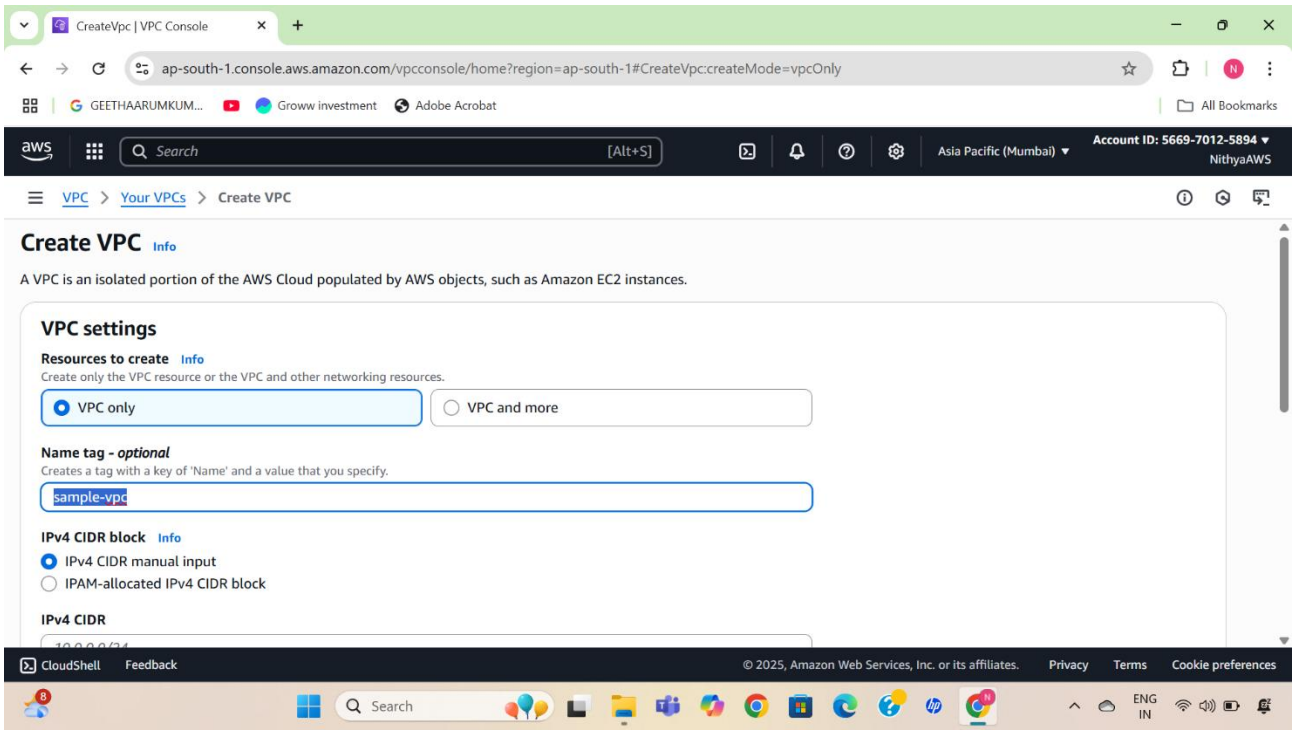
Step 4: Configure VPC Settings

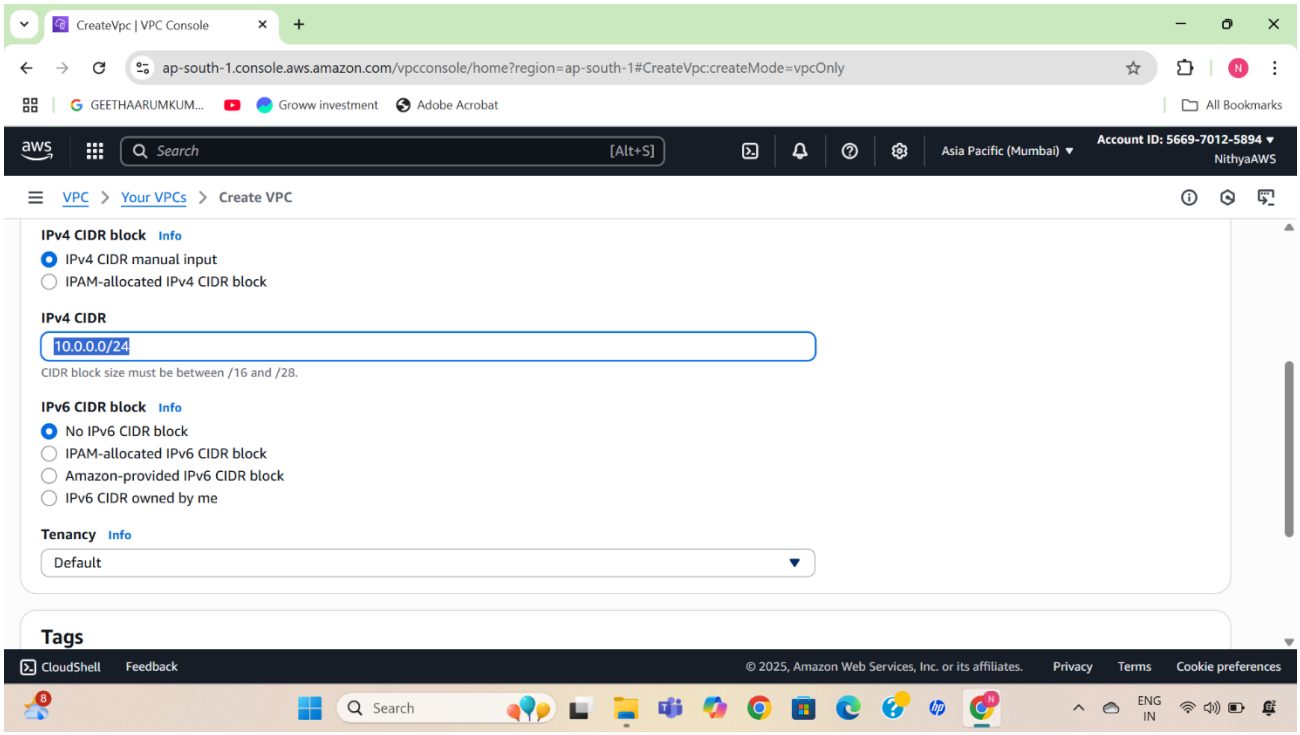
1. **Name your VPC**:
   - Choose a name that describes the purpose of the VPC (e.g., MyFirstVPC).
2. **IPv4 CIDR block**:
   - Choose a private IP range that will be used for the VPC. Example: 10.0.0.0/16. This means your VPC can have IPs from 10.0.0.0 to 10.0.255.255.
3. **IPv6 CIDR block**:
   - Leave it as **No IPv6 CIDR Block** unless you want to enable IPv6 for your VPC.
4. **Tenancy**:
   - **Default** is usually sufficient, unless you need a **dedicated** instance type, which is more expensive.

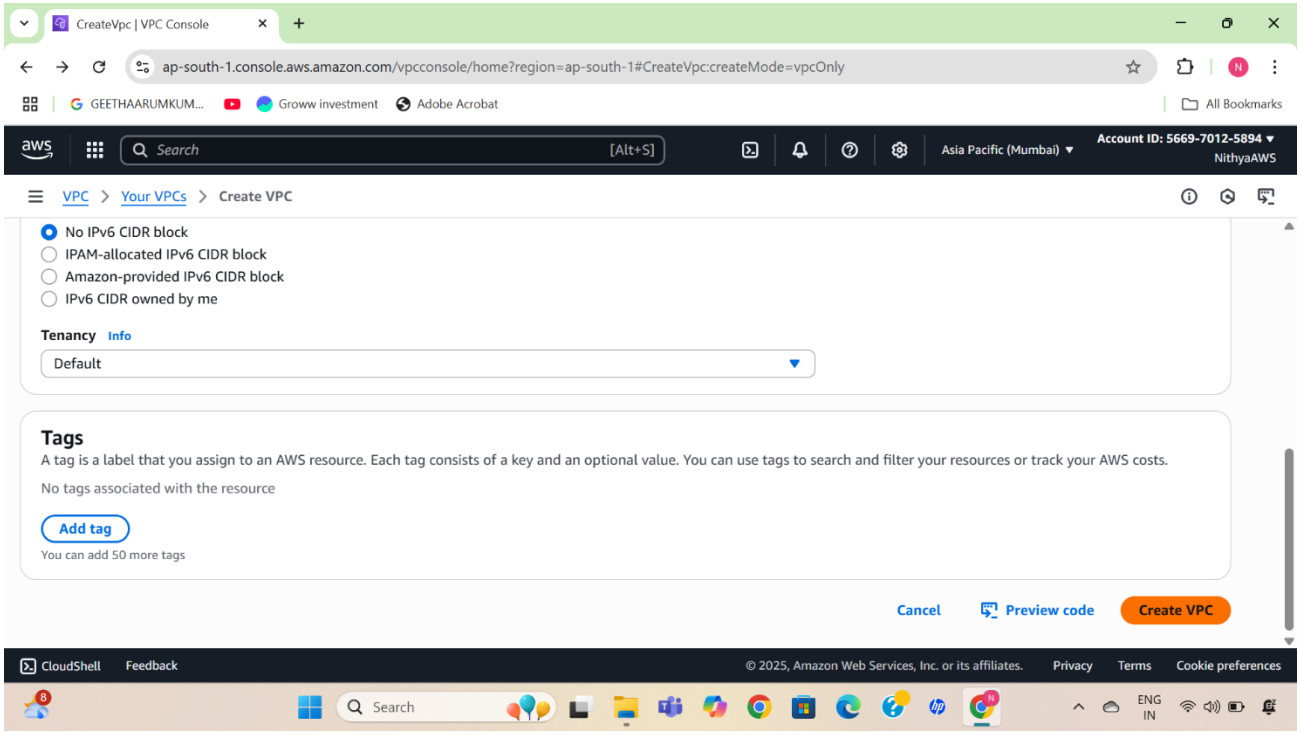
Click "Create VPC".

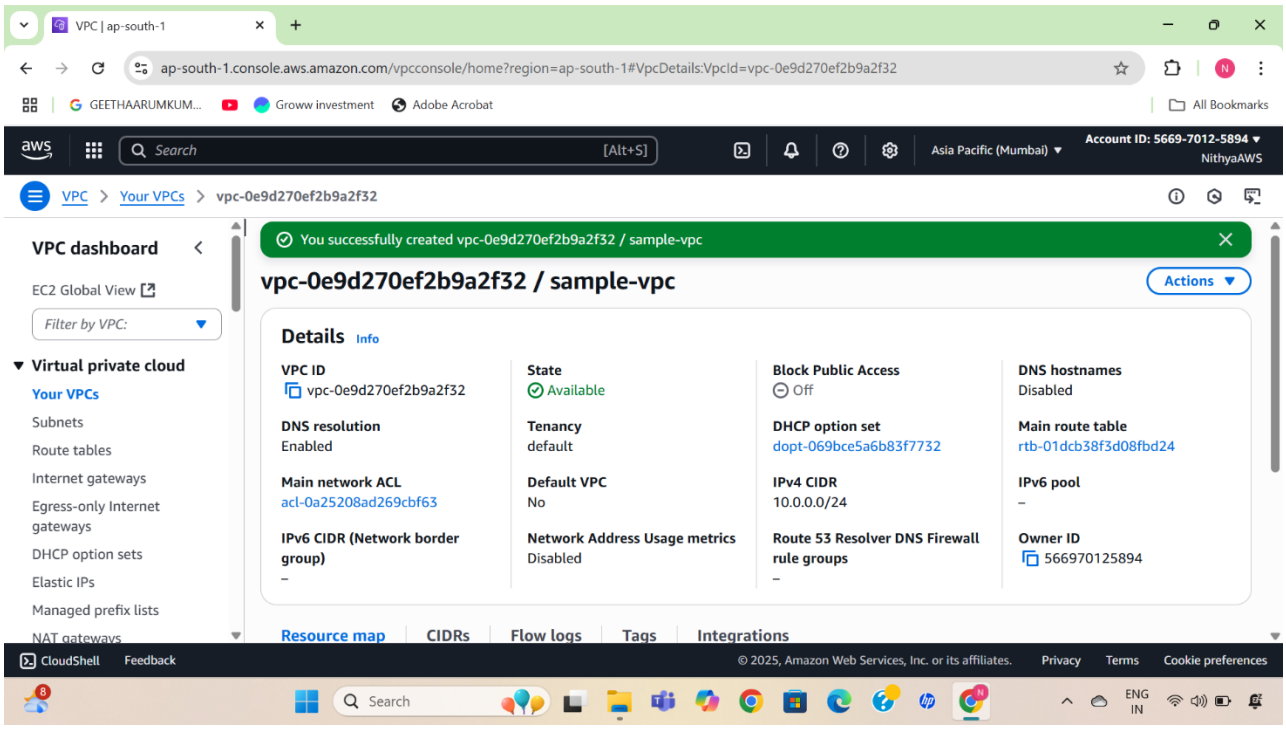
OUTPUT:

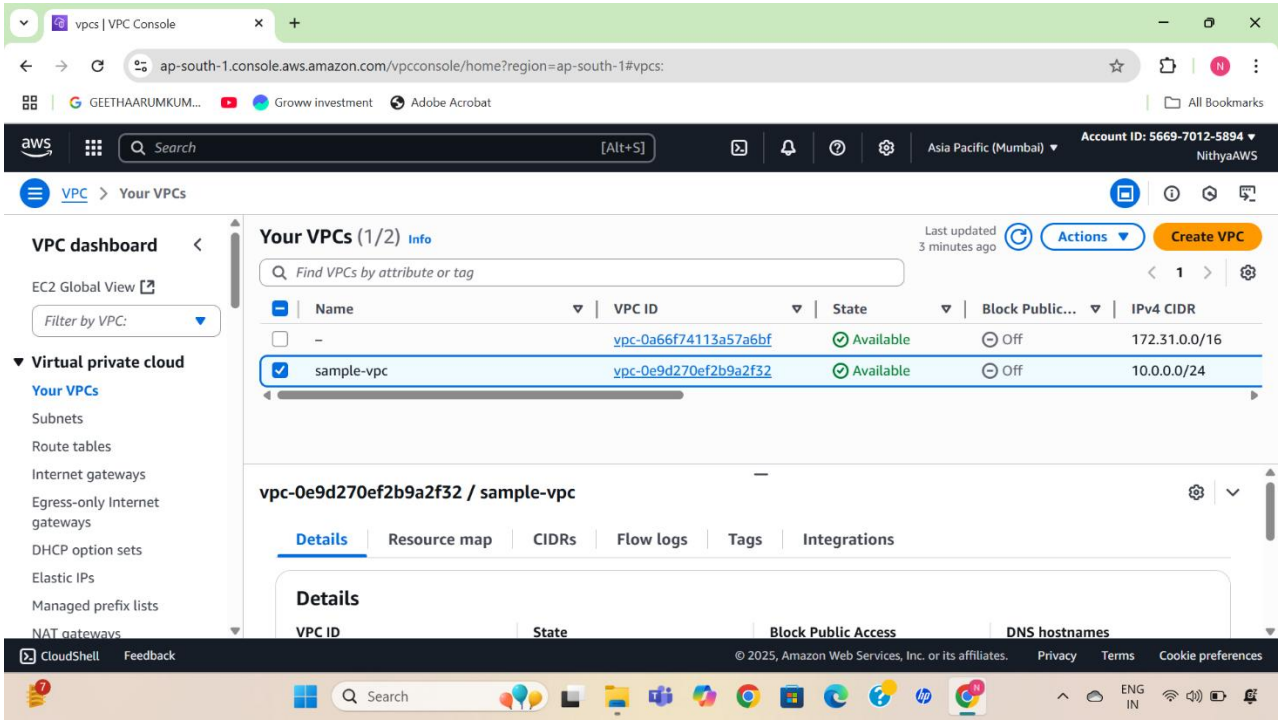

Home | VPC Console
ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#Home:
Search [Alt+S]
Asia Pacific (Mumbai)
Account ID: 5669-7012-5894
NithyaAWS
VPC dashboard
EC2 Global View
Filter by VPC:
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only Internet gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways
Create VPC
Launch EC2 Instances
Note: Your Instances will launch in the Asia Pacific region.
Resources by Region
Refresh Resources
You are using the following Amazon VPC resources
VPCs Mumbai 1
See all regions
NAT Gateways Mumbai 0
See all regions
Subnets Mumbai 3
See all regions
VPC Peering Connections Mumbai 0
See all regions
Route Tables Mumbai 1
See all regions
Network ACLs Mumbai 1
See all regions
Service Health
View complete service health details
Settings
Block Public Access
Zones
Console Experiments
Additional Information
VPC Documentation
javascript:void(0)
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


CreateVpc | VPC Console
ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#CreateVpc:createMode=vpcOnly
Search [Alt+S]
Asia Pacific (Mumbai)
Account ID: 5669-7012-5894
NithyaAWS
VPC > Your VPCs > Create VPC
Create VPC Info
A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances.
VPC settings
Resources to create Info
Create only the VPC resource or the VPC and other networking resources.
VPC only
VPC and more
Name tag - optional
Creates a tag with a key of 'Name' and a value that you specify.
sample-vpc
IPv4 CIDR block Info
IPv4 CIDR manual input
IPAM-allocated IPv4 CIDR block
IPv4 CIDR
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


CreateVpc | VPC Console
ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#CreateVpc:createMode=vpcOnly
GEETHAARUMKUM...
Groww investment
Adobe Acrobat
All Bookmarks
aws
Search
[Alt+S]
Asia Pacific (Mumbai)
Account ID: 5669-7012-5894
NithyaAWS
VPC > Your VPCs > Create VPC
IPv4 CIDR block Info
IPv4 CIDR manual input
IPAM-allocated IPv4 CIDR block
IPv4 CIDR
10.0.0.0/24
CIDR block size must be between /16 and /28.
IPv6 CIDR block Info
No IPv6 CIDR block
IPAM-allocated IPv6 CIDR block
Amazon-provided IPv6 CIDR block
IPv6 CIDR owned by me
Tenancy Info
Default
Tags
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN


CreateVpc | VPC Console
ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#CreateVpc:createMode=vpcOnly
GEETHAARUMKUM...
Groww investment
Adobe Acrobat
All Bookmarks
aws
Search
[Alt+S]
Asia Pacific (Mumbai)
Account ID: 5669-7012-5894
NithyaAWS
VPC > Your VPCs > Create VPC
No IPv6 CIDR block
IPAM-allocated IPv6 CIDR block
Amazon-provided IPv6 CIDR block
IPv6 CIDR owned by me
Tenancy Info
Default
Tags
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.
No tags associated with the resource
Add tag
You can add 50 more tags
Cancel
Preview code
Create VPC
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN


VPC | ap-south-1
ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#VpcDetails:VpcId=vpc-0e9d270ef2b9a2f32
Search
[Alt+S]
Asia Pacific (Mumbai)
Account ID: 5669-7012-5894
NithyaAWS
VPC > Your VPCs > vpc-0e9d270ef2b9a2f32
VPC dashboard
EC2 Global View
Filter by VPC:
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only Internet gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways
You successfully created vpc-0e9d270ef2b9a2f32 / sample-vpc
vpc-0e9d270ef2b9a2f32 / sample-vpc
Actions
Details Info
VPC ID
vpc-0e9d270ef2b9a2f32
State
Available
Block Public Access
Off
DNS hostnames
Disabled
DNS resolution
Enabled
Tenancy
default
DHCP option set
dopt-069bce5a6b83f7732
Main route table
rtb-01dcb38f3d08fbd24
Main network ACL
acl-0a25208ad269cbf63
Default VPC
No
IPv4 CIDR
10.0.0.0/24
IPv6 pool
-
IPv6 CIDR (Network border group)
-
Network Address Usage metrics
Disabled
Route 53 Resolver DNS Firewall rule groups
-
Owner ID
566970125894
Resource map
CIDRs
Flow logs
Tags
Integrations
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


vpcs | VPC Console
ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#vpcs:
Search
[Alt+S]
Asia Pacific (Mumbai)
Account ID: 5669-7012-5894
NithyaAWS
VPC > Your VPCs
VPC dashboard
EC2 Global View
Filter by VPC:
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only Internet gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways
Your VPCs (1/2) Info
Last updated 3 minutes ago
Actions
Create VPC
Find VPCs by attribute or tag
Name
VPC ID
State
Block Public...
IPv4 CIDR
-
vpc-0a66f74113a57a6bf
Available
Off
172.31.0.0/16
sample-vpc
vpc-0e9d270ef2b9a2f32
Available
Off
10.0.0.0/24
vpc-0e9d270ef2b9a2f32 / sample-vpc
Details
Resource map
CIDRs
Flow logs
Tags
Integrations
Details
VPC ID
State
Block Public Access
DNS hostnames
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

RESULT:

EX. No: 9 **Create public and private subnet**
Date:

AIM:

PROCEDURE:

Step 1: Create Subnets for the VPC

Once your VPC is created, you'll need to create at least two subnets:

- **Public Subnet**: For resources that require internet access (e.g., web servers).
- **Private Subnet**: For resources that should not have direct access to the internet (e.g., databases, app servers).

1. **Go to Subnets**:
    - In the **VPC Dashboard**, click on **Subnets** and then **Create Subnet**.
2. **Create Public Subnet**:
    - **Name tag**: Public Subnet.
    - **VPC**: Select the VPC you just created.
    - **Availability Zone**: Select an Availability Zone (e.g., us-east-1a).
    - **IPv4 CIDR block**: Choose a subnet CIDR block within your VPC range (e.g., 10.0.1.0/24).
    - **Auto-assign public IPv4 address**: **Yes** (this allows instances in the public subnet to receive public IP addresses).

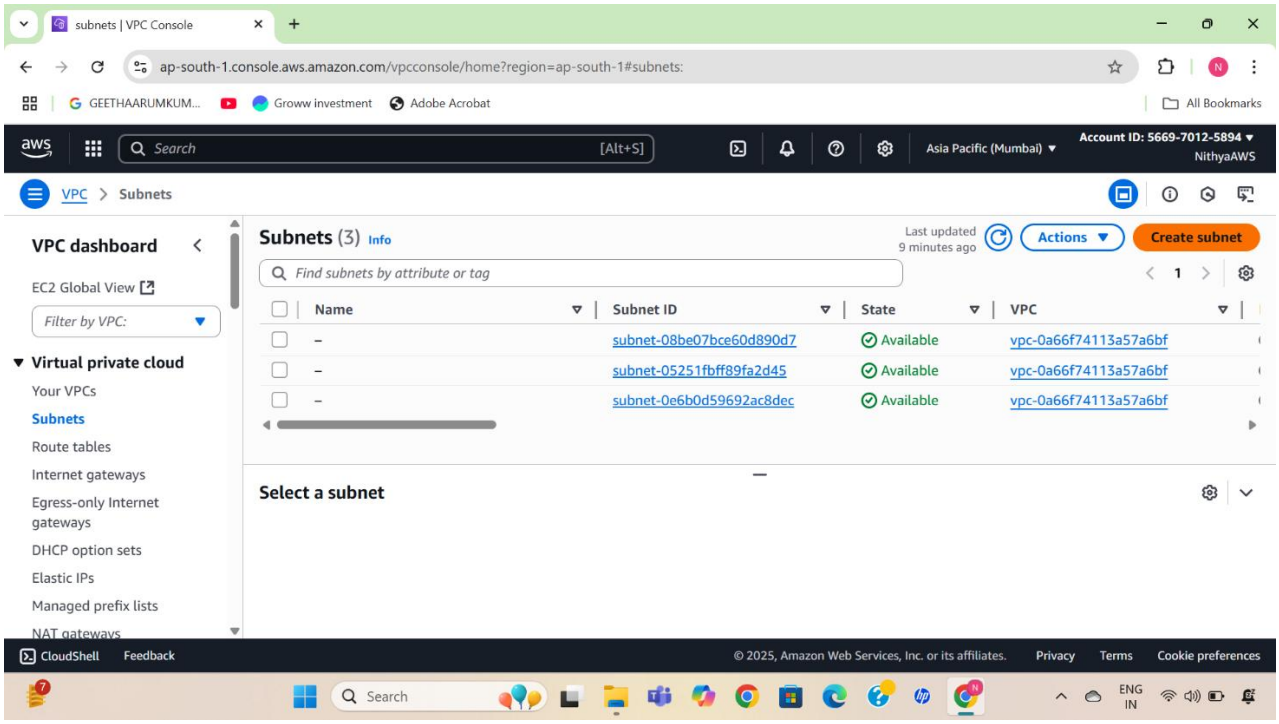
Click "Create" to create the public subnet.

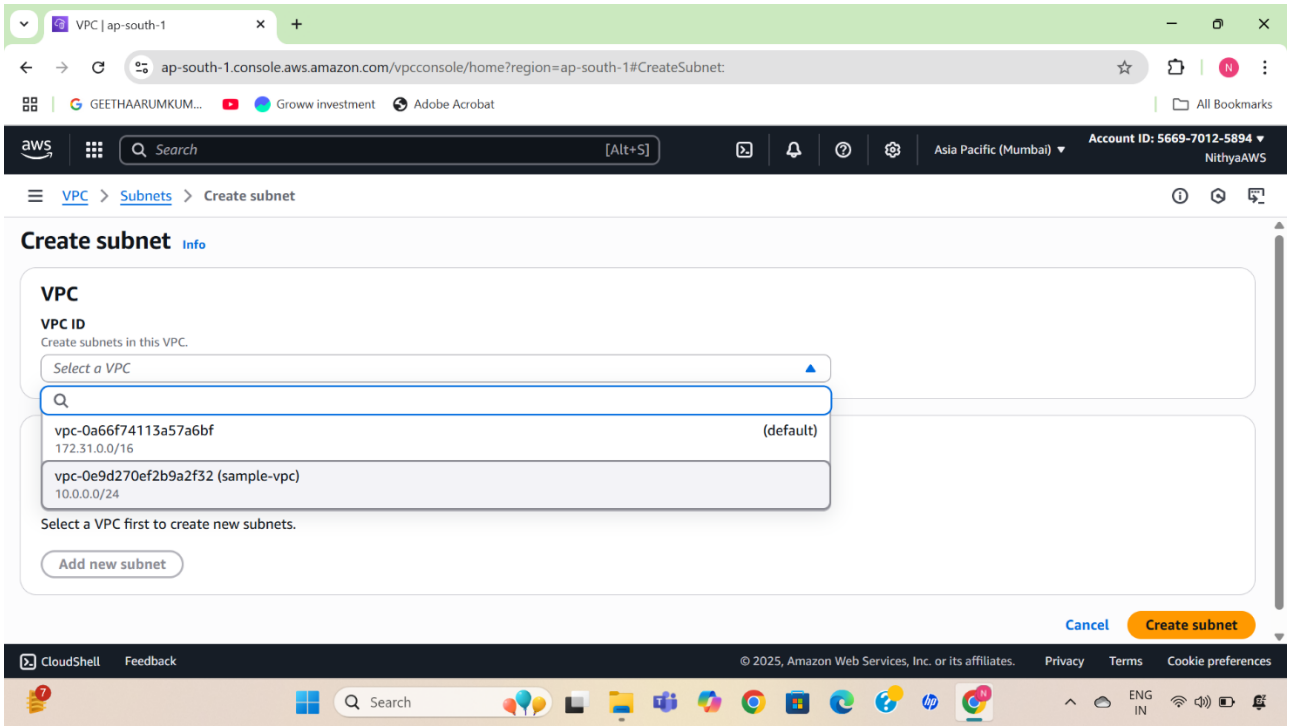
3. **Create Private Subnet**:
    - **Name tag**: PrivateSubnet.
    - **VPC**: Select the VPC you just created.
    - **Availability Zone**: Select another Availability Zone (e.g., us-east-1b).
    - **IPv4 CIDR block**: Choose a subnet CIDR block (e.g., 10.0.2.0/24).
    - Leave the **Auto-assign public IPv4 address** as **No** (since this subnet will not have public IPs).

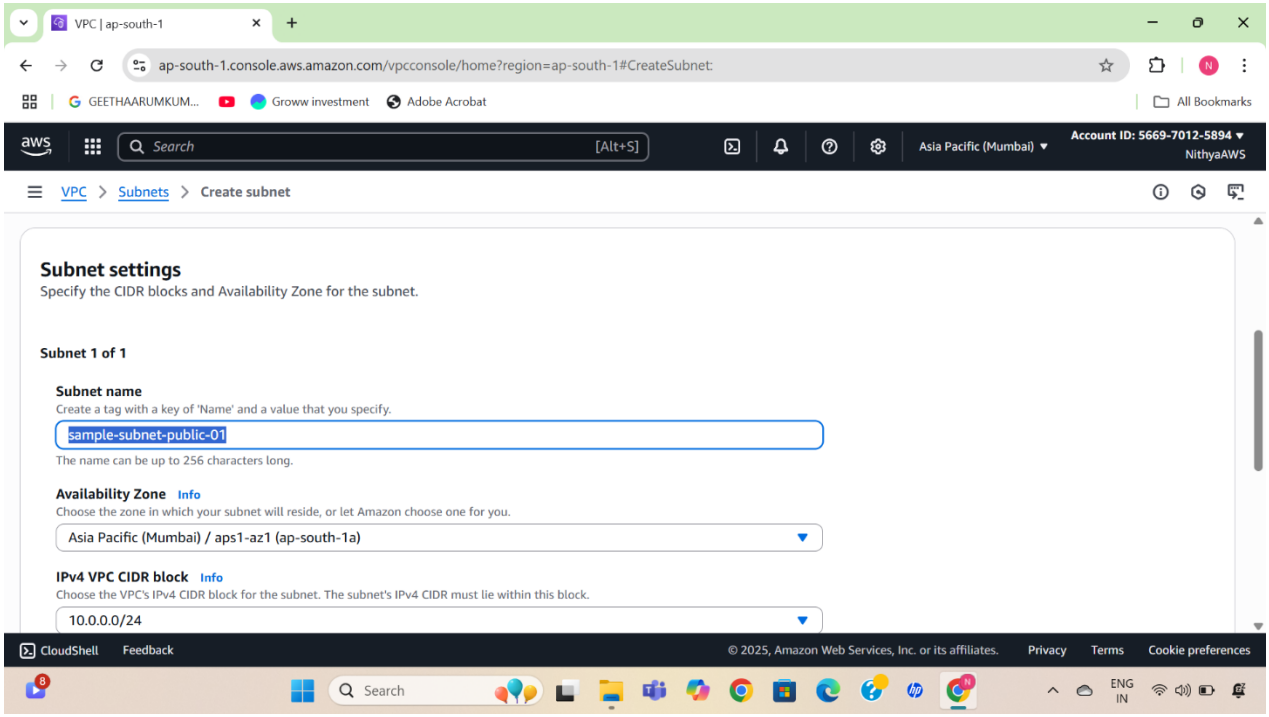
Click "Create" to create the private subnet.

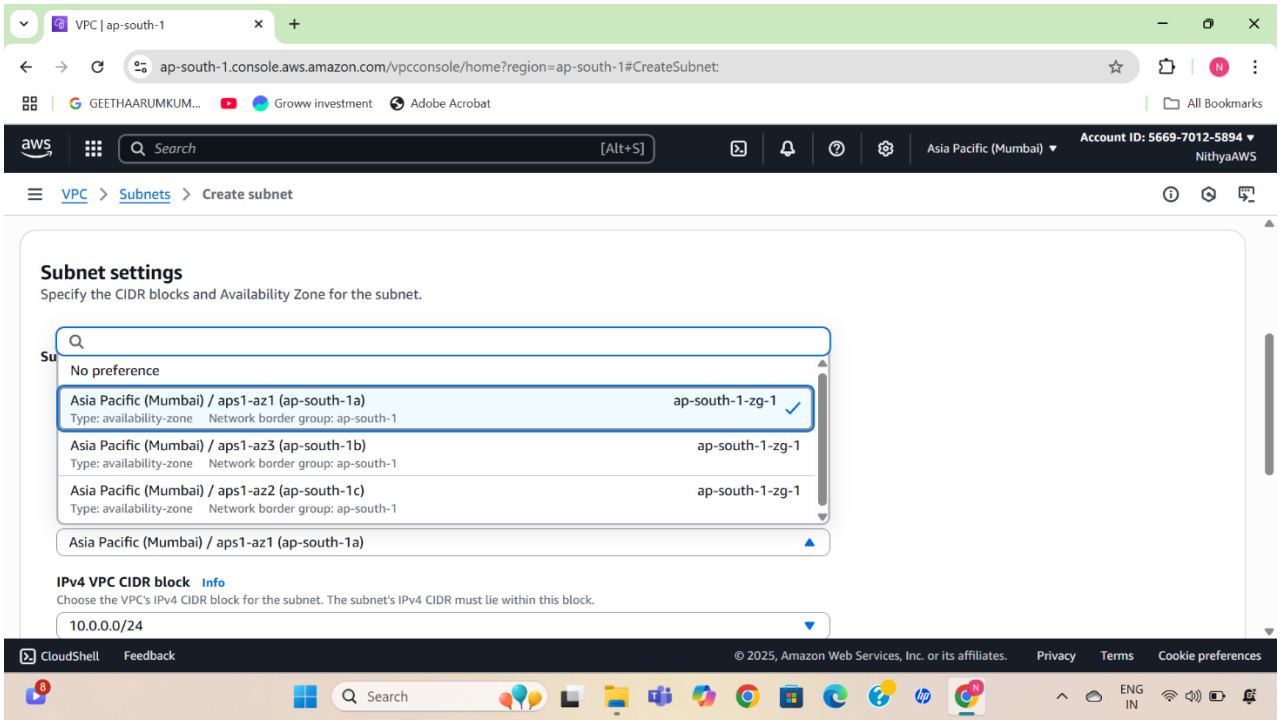
OUTPUT:

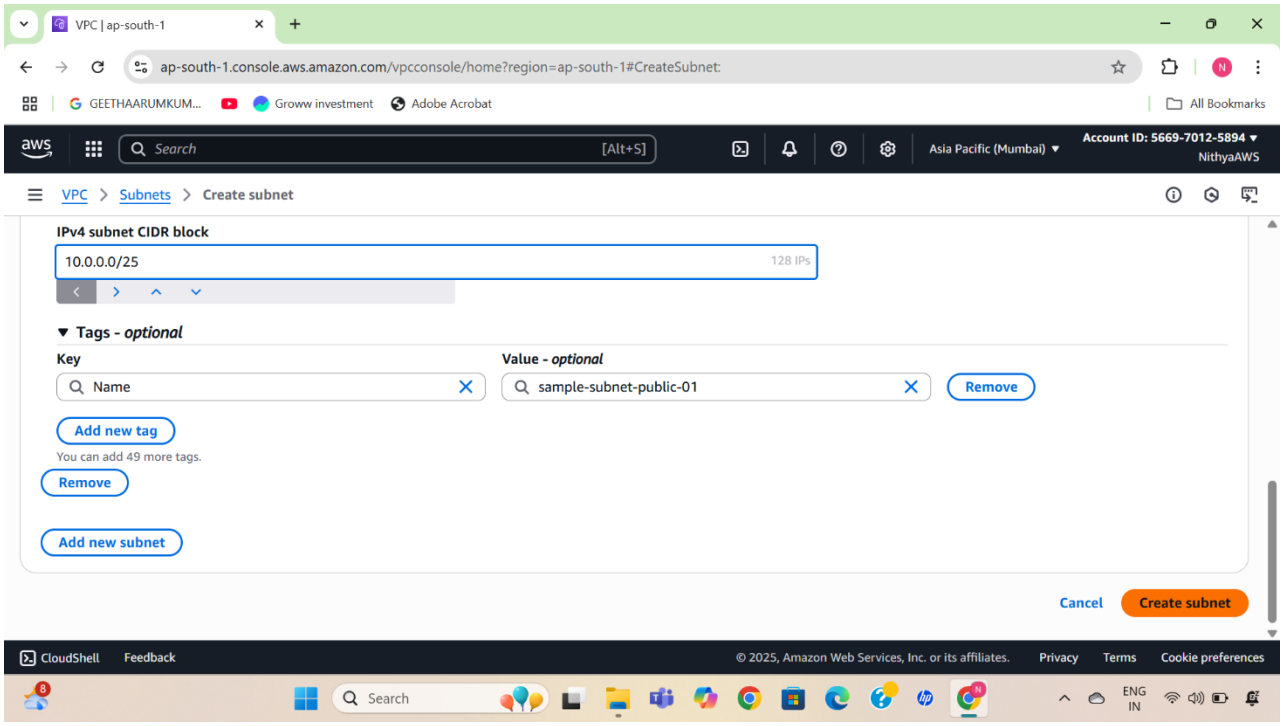
PUBLIC SUBNET:

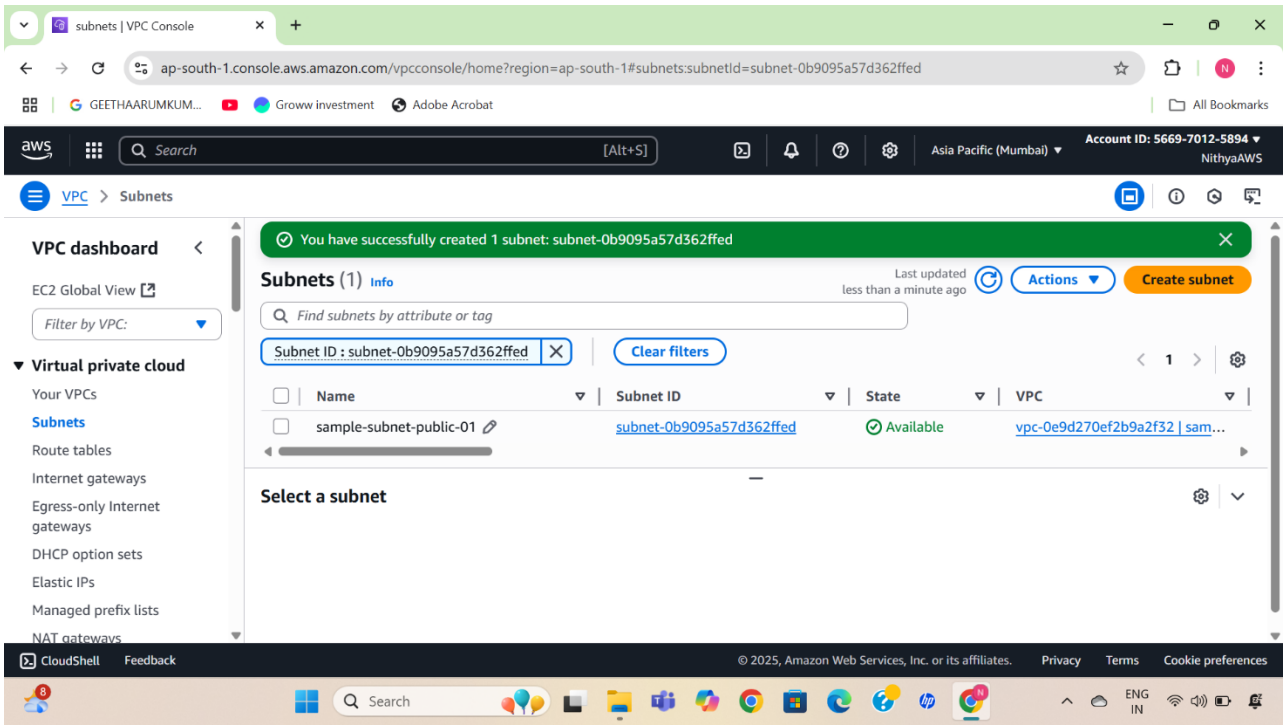




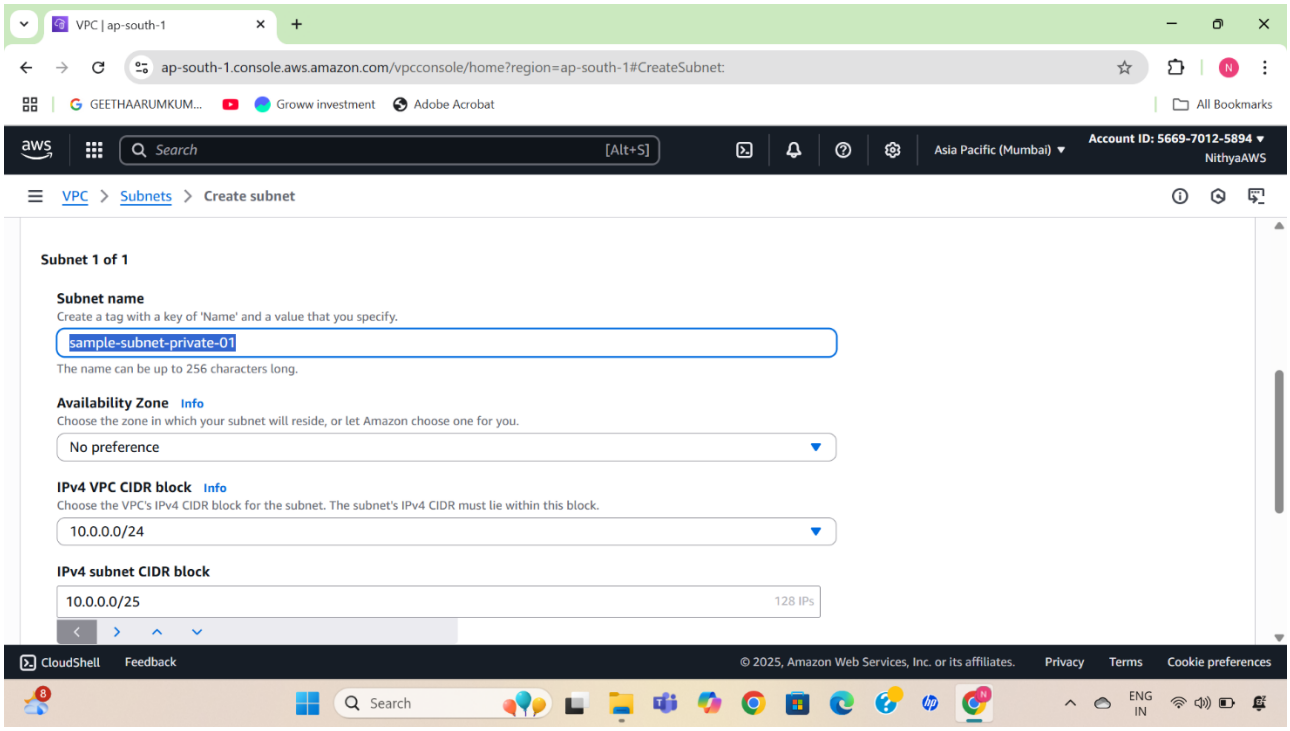

VPC | ap-south-1
ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#CreateSubnet:
GEETHAARUMKUM...
Groww investment
Adobe Acrobat
All Bookmarks
Search
[Alt+S]
Asia Pacific (Mumbai)
Account ID: 5669-7012-5894
NithyaAWS
VPC > Subnets > Create subnet
Subnet settings
Specify the CIDR blocks and Availability Zone for the subnet.
Subnet 1 of 1
Subnet name
Create a tag with a key of 'Name' and a value that you specify.
sample-subnet-public-01
The name can be up to 256 characters long.
Availability Zone Info
Choose the zone in which your subnet will reside, or let Amazon choose one for you.
Asia Pacific (Mumbai) / aps1-az1 (ap-south-1a)
IPv4 VPC CIDR block Info
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.
10.0.0.0/24
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN

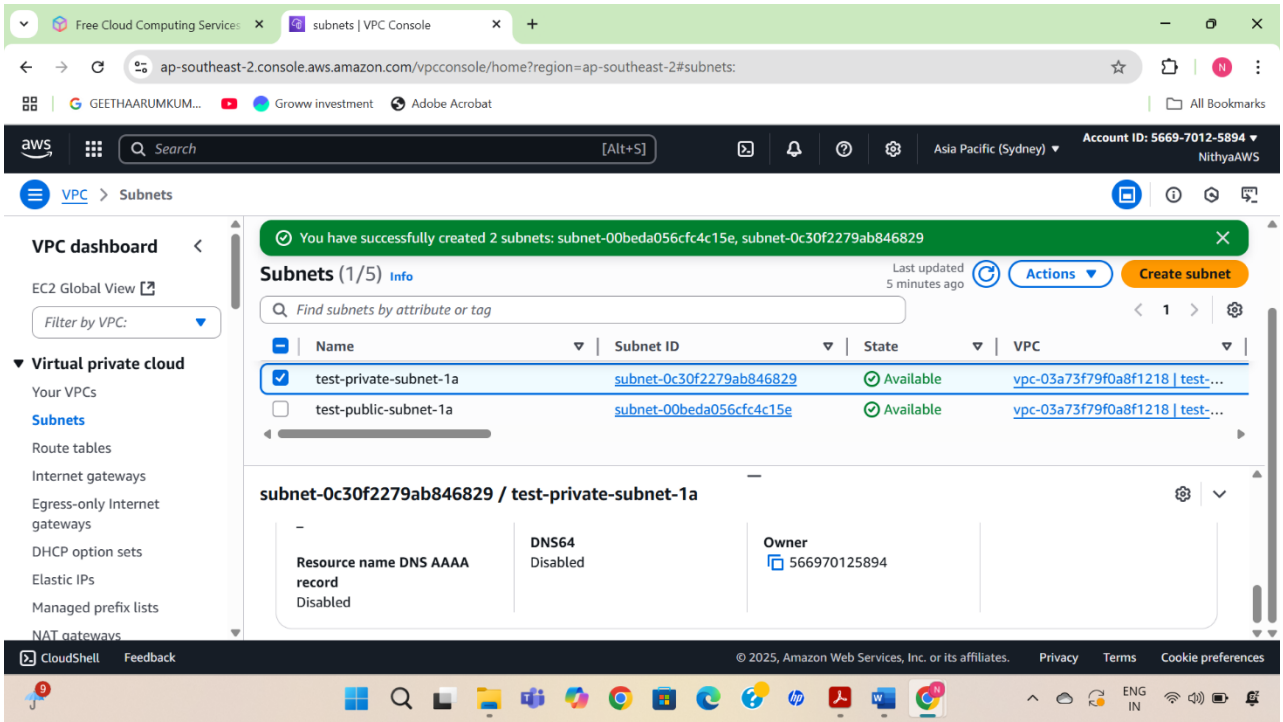

VPC | ap-south-1
ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#CreateSubnet:
GEETHAARUMKUM...
Groww investment
Adobe Acrobat
All Bookmarks
Search
[Alt+S]
Asia Pacific (Mumbai)
Account ID: 5669-7012-5894
NithyaAWS
VPC > Subnets > Create subnet
Subnet settings
Specify the CIDR blocks and Availability Zone for the subnet.
No preference
Asia Pacific (Mumbai) / aps1-az1 (ap-south-1a)
ap-south-1-zg-1
Type: availability-zone Network border group: ap-south-1
Asia Pacific (Mumbai) / aps1-az3 (ap-south-1b)
ap-south-1-zg-1
Type: availability-zone Network border group: ap-south-1
Asia Pacific (Mumbai) / aps1-az2 (ap-south-1c)
ap-south-1-zg-1
Type: availability-zone Network border group: ap-south-1
Asia Pacific (Mumbai) / aps1-az1 (ap-south-1a)
IPv4 VPC CIDR block Info
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.
10.0.0.0/24
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN

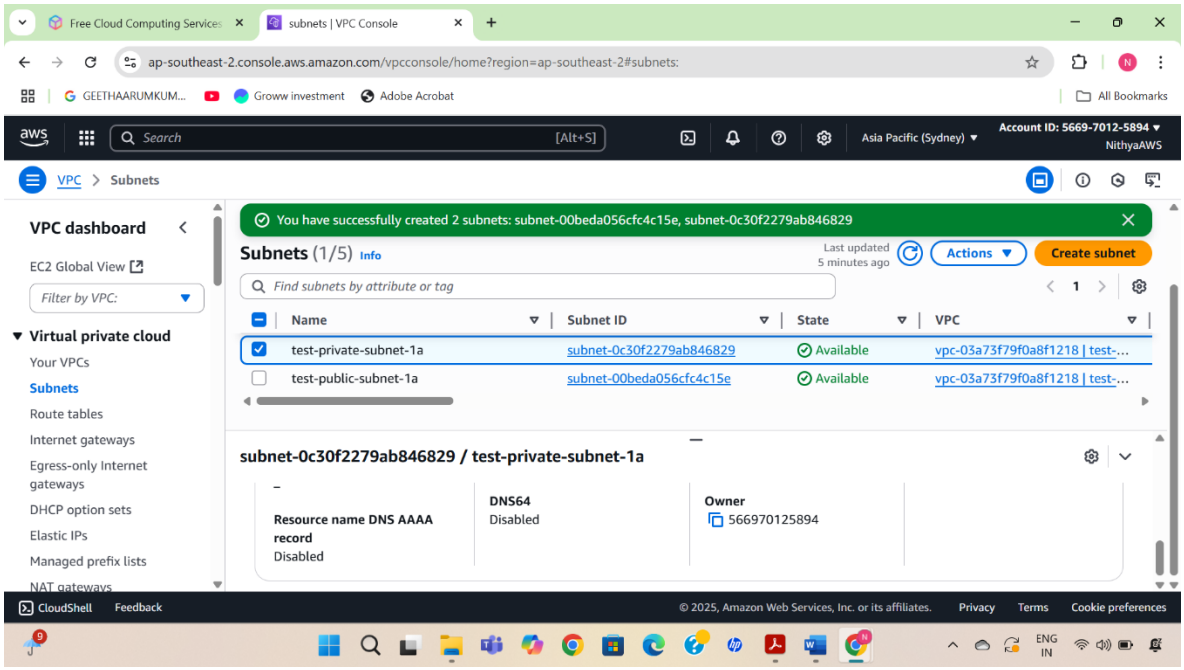

VPC | ap-south-1
ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#CreateSubnet:
Search [Alt+S]
Asia Pacific (Mumbai)
Account ID: 5669-7012-5894
NithyaAWS
VPC > Subnets > Create subnet
IPv4 subnet CIDR block
10.0.0.0/25
128 IPs
Tags - optional
Key
Value - optional
Name
sample-subnet-public-01
Remove
Add new tag
You can add 49 more tags.
Remove
Add new subnet
Cancel
Create subnet
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


subnets | VPC Console
ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#subnets:subnetId=subnet-0b9095a57d362ffed
Search [Alt+S]
Asia Pacific (Mumbai)
Account ID: 5669-7012-5894
NithyaAWS
VPC > Subnets
VPC dashboard
EC2 Global View
Filter by VPC:
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only Internet gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways
You have successfully created 1 subnet: subnet-0b9095a57d362ffed
Subnets (1) Info
Last updated less than a minute ago
Actions
Create subnet
Find subnets by attribute or tag
Subnet ID : subnet-0b9095a57d362ffed
Clear filters
Name
Subnet ID
State
VPC
sample-subnet-public-01
subnet-0b9095a57d362ffed
Available
vpc-0e9d270ef2b9a2f32 | sam...
Select a subnet
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

PRIVATE SUBNET: (Same Procedure like public subnet creation)






Free Cloud Computing Services
subnets | VPC Console
ap-southeast-2.console.aws.amazon.com/vpcconsole/home?region=ap-southeast-2#subnets:
GEETHAARUMKUM...
Groww investment
Adobe Acrobat
All Bookmarks
aws
Search
[Alt+S]
Asia Pacific (Sydney)
Account ID: 5669-7012-5894
NithyaAWS
VPC > Subnets
VPC dashboard
EC2 Global View
Filter by VPC:
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only Internet gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways
You have successfully created 2 subnets: subnet-00beda056cfc4c15e, subnet-0c30f2279ab846829
Subnets (1/5) Info
Last updated 5 minutes ago
Actions
Create subnet
Find subnets by attribute or tag
Name
Subnet ID
State
VPC
test-private-subnet-1a
subnet-0c30f2279ab846829
Available
vpc-03a73f79f0a8f1218 | test-...
test-public-subnet-1a
subnet-00beda056cfc4c15e
Available
vpc-03a73f79f0a8f1218 | test-...
subnet-0c30f2279ab846829 / test-private-subnet-1a
Resource name DNS AAAA record
Disabled
DNS64
Disabled
Owner
566970125894
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN

RESULT:

EX. No: 10 **INSTALL C COMPILER IN VIRTUAL MACHINE AND**
Date : **EXECUTE PROGRAMS**

AIM:

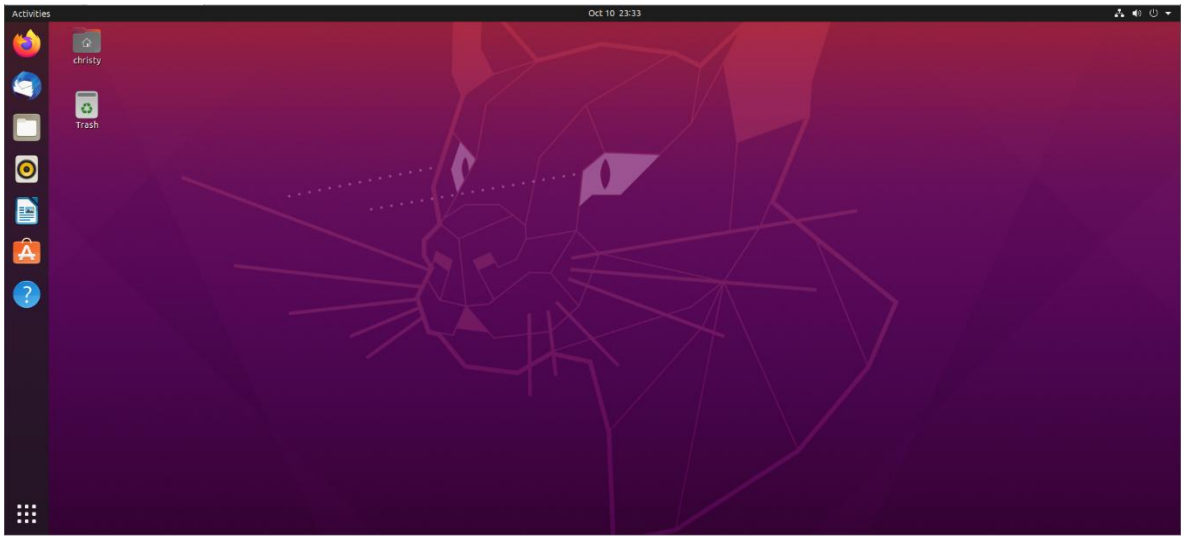
PROGRAM:

```
#include <stdio.h>
int check anagram(char [], char []);
int main()
{
        char a[1000], b[1000];
        printf("Enter two strings\n");
        scanf("%s",a);
        scanf("%s",b);
          if (check_anagram(a, b))
          printf("The strings are anagrams.\n");
          else
          printf("The strings aren't anagrams.\n");
 return 0;
}

int check_anagram(char a[], char b[])
{
 int first[26] = {0}, second[26] = {0}, c=0;
   while (a[c] != '\0') {
   first[a[c]-'a']++;
   c++;
 }
 c = 0;
 while (b[c] != '\0') {
   second[b[c]-'a']++;
   c++;
 }
 for (c = 0; c < 26; c++)
   if (first[c] != second[c])
     return 0;

 return 1;
}
```

PROCEDURE:

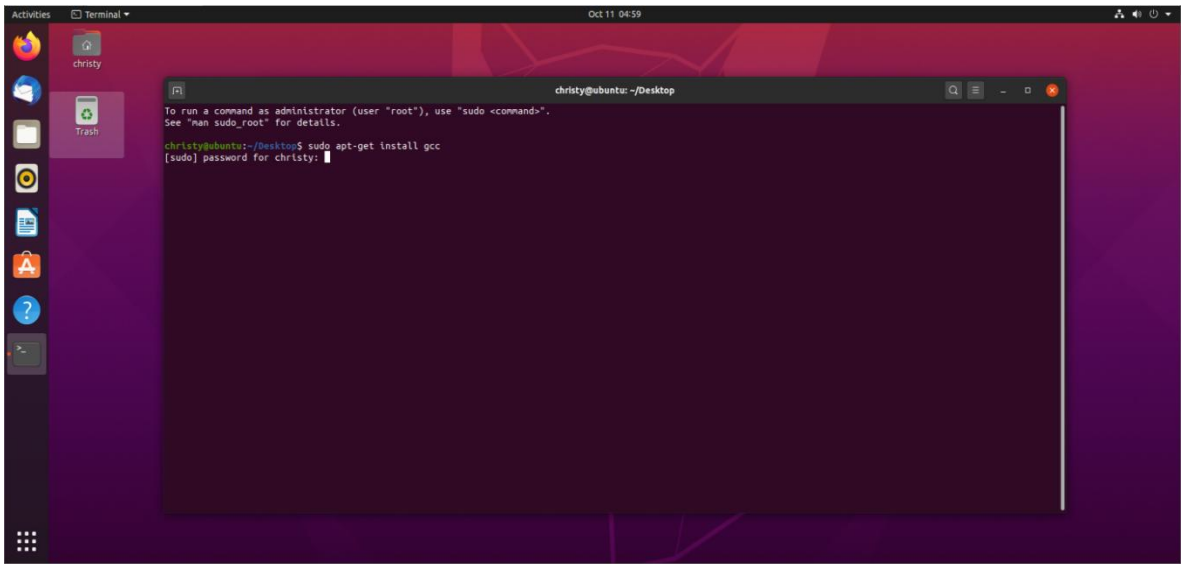
STEP 1: OPEN VIRTUAL MACHINE

Open the Ubuntu virtual machine.



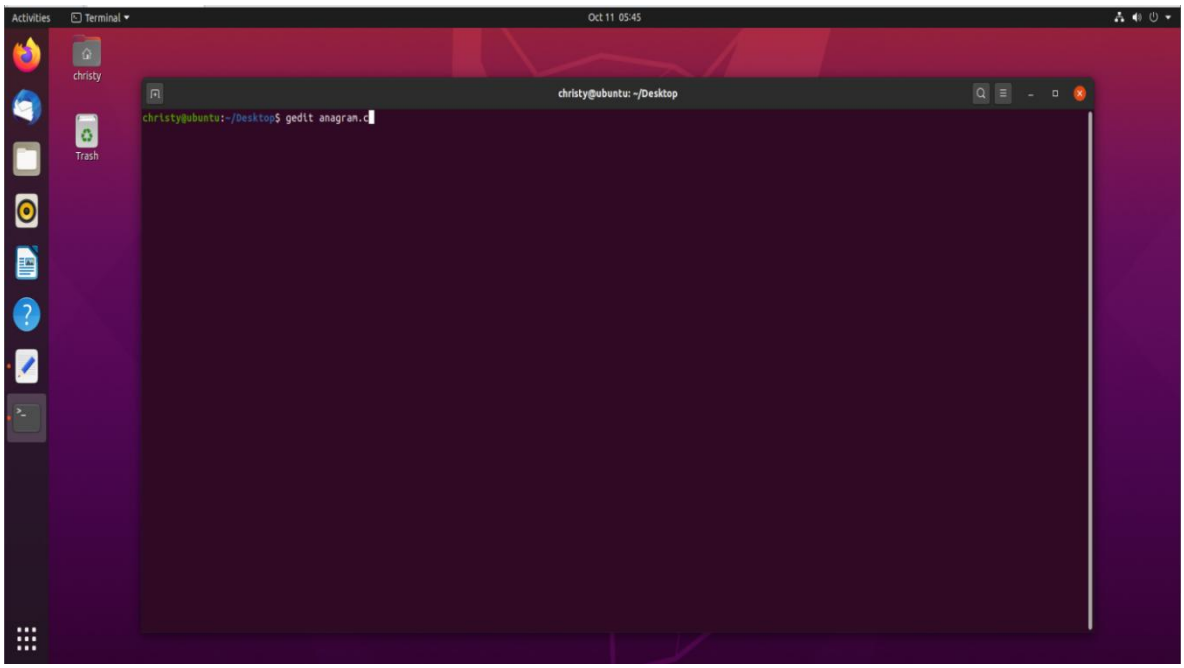
STEP 2 : INSTALL C COMPILER

- Right click and select **Open Terminal .**The terminal windows opens.
- Type the following command to install the C Compiler: **sudo apt-get install gcc**
- Type the password to allow the installation.

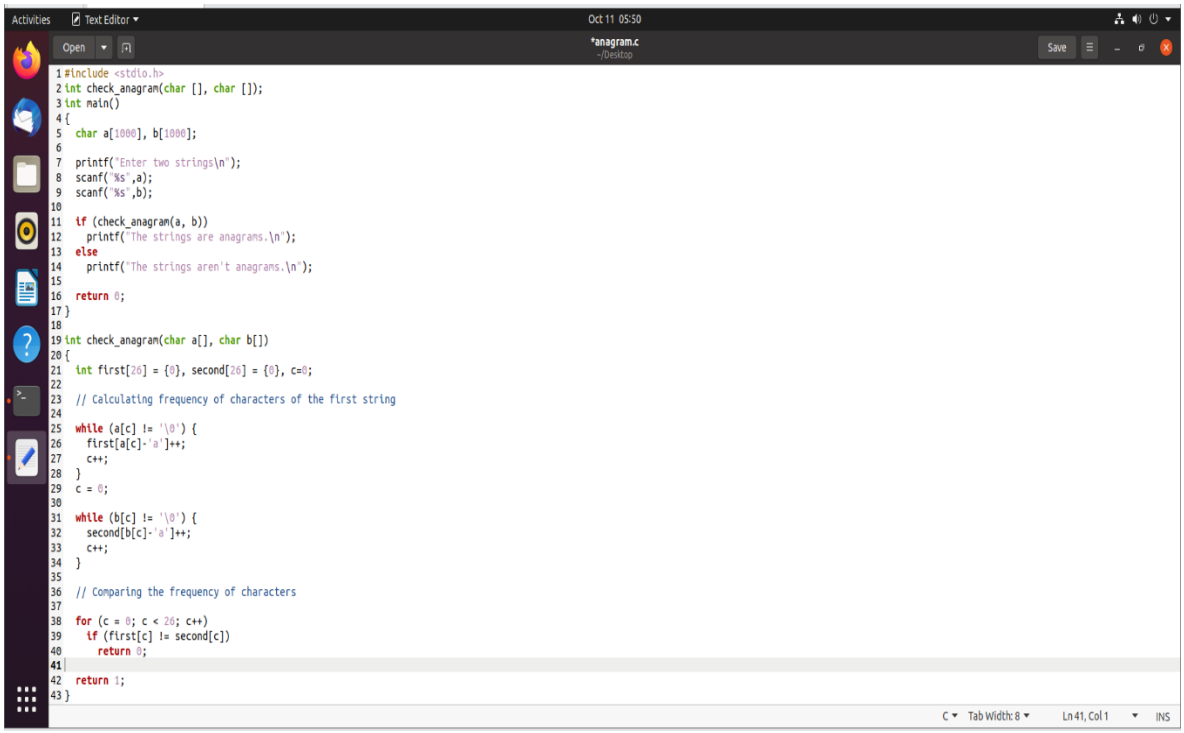


STEP 3: EXECUTE PROGRAM

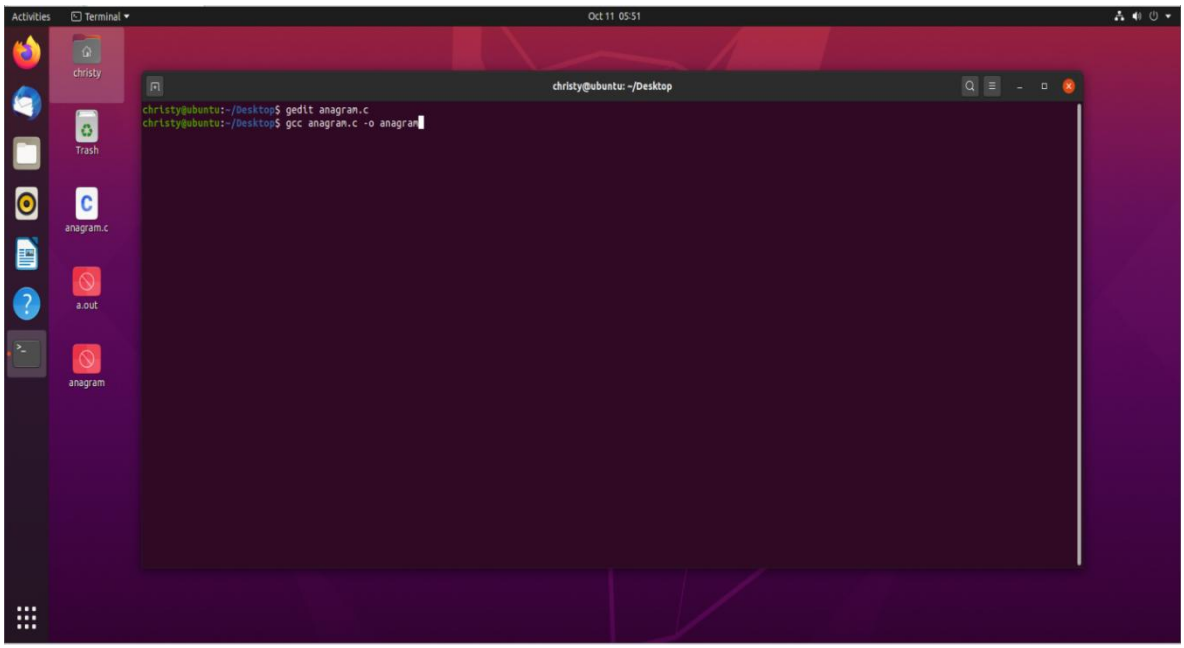
- Open the terminal and open the gedit editor using the command : **gedit filename.c**



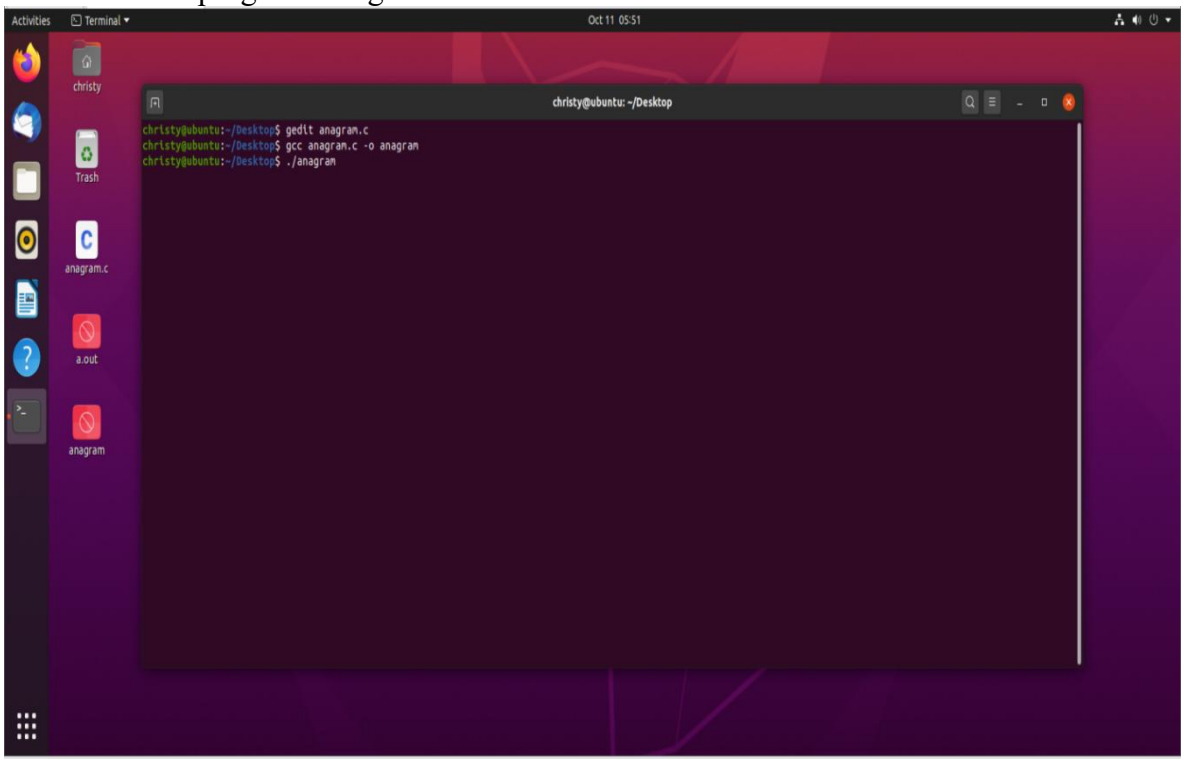
- Type the C program and save the file.



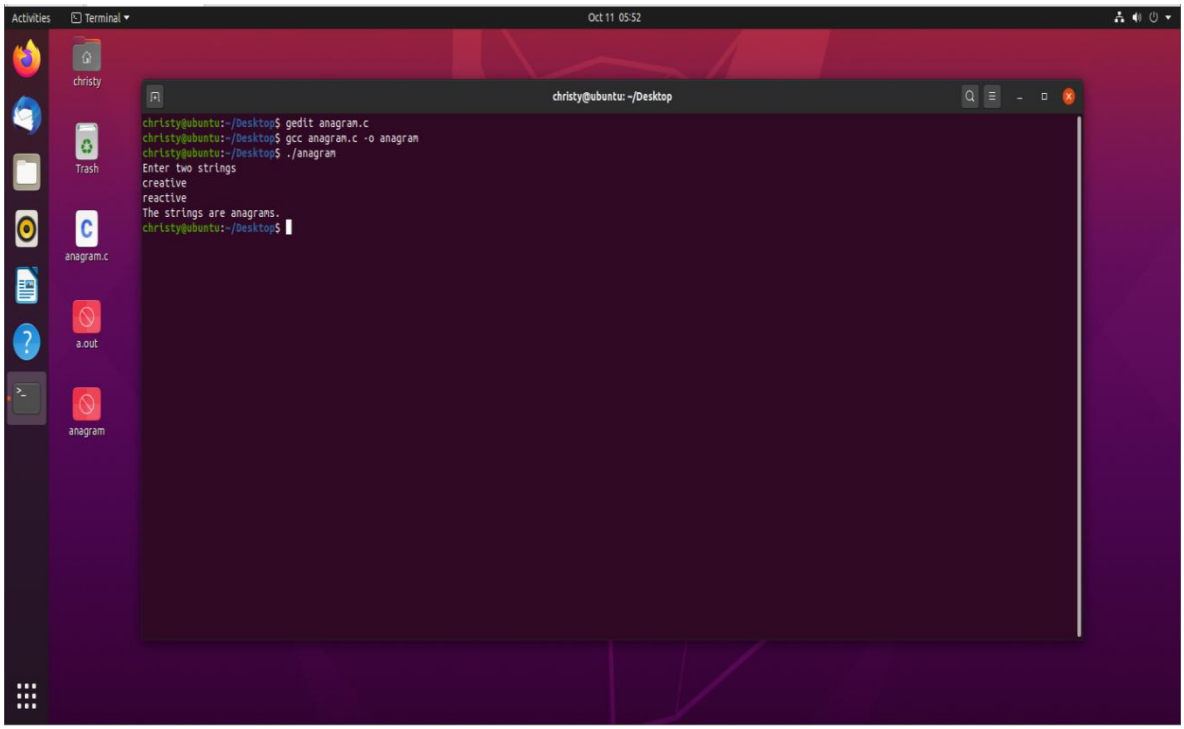
- Compile the C program using the following command : **gcc filename.c -o filename**



- Run the C program using the command : **./filename**



OUTPUT:



RESULT:

Exp: 11 Develop an IoT application to connect and configure IoT devices to the cloud
Date:

Aim:
To develop an IoT application that allows registration of devices, monitoring of telemetry, and sending remote commands through Azure IoT Hub.

Algorithm:
1. Create IoT Hub in Azure using CLI.
2. Register devices in IoT Hub using CLI or Python SDK.
3. Simulate IoT Device with Python to send telemetry and receive commands.
4. Monitor Device Messages using Event Hub-compatible endpoint.
5. Send Remote Commands from Admin application using C2D messaging.
6. Verify results by checking console outputs.
7. Create IoT Hub and device identity using Azure CLI.
8. Get device and IoT Hub connection strings.
9. Run the device simulation using: python device_sim.py
10. Run the admin application using: python admin.py

Code
(a) Device Simulation (device_sim.py)

```
import time
import random
from azure.iot.device import IoTHubDeviceClient, Message

CONNECTION_STRING = "<DEVICE_CONNECTION_STRING>"

def main():
    client = IoTHubDeviceClient.create_from_connection_string(CONNECTION_STRING)

    print("IoT device sending messages...")
    while True:
        temperature = random.randint(20, 35)
        humidity = random.randint(40, 70)
        msg = Message(f'{{"temperature": {temperature}, "humidity": {humidity}}}')
        client.send_message(msg)
        print(f"Sent: {msg}")
        time.sleep(5)

        # Listen for cloud-to-device commands
        try:
            message = client.receive_message(timeout=1)
            print("Received command:", message.data.decode())
        except:
            pass

if __name__ == '__main__':
    main()
```

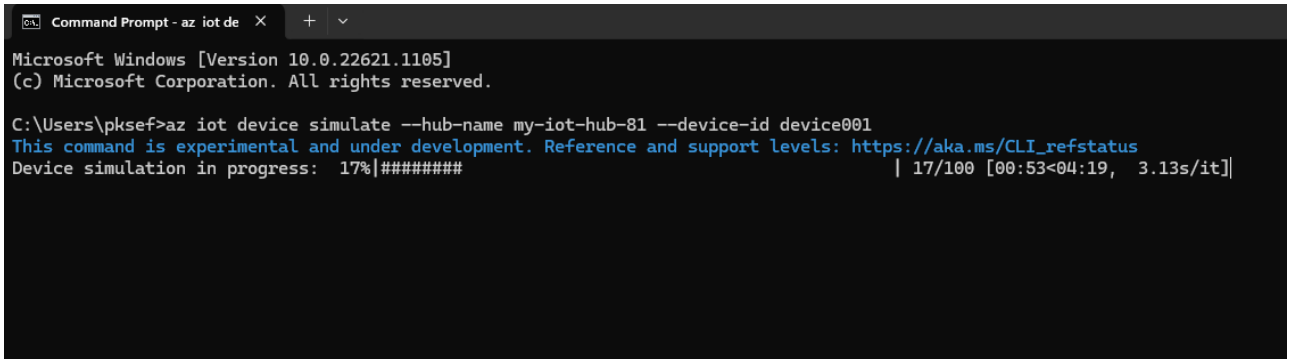
(b) Admin Application (admin.py)

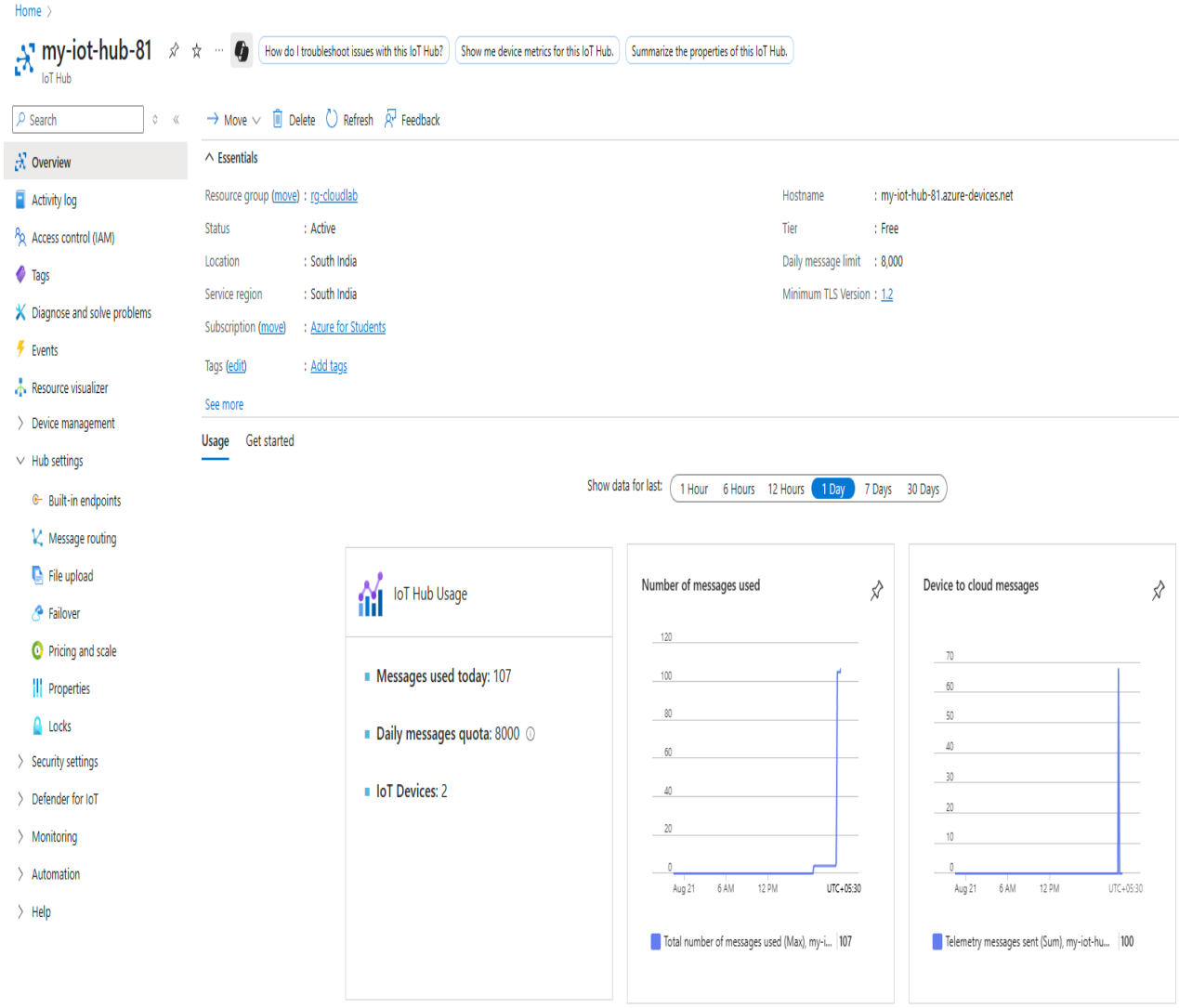
```
from azure.iot.hub import IoTHubRegistryManager

CONNECTION_STRING = "<IOTHUB_CONNECTION_STRING>"
```

```
DEVICE_ID = "device001"

registry_manager = IoTHubRegistryManager(CONNECTION_STRING)

def send_command(device_id, command):
    message = f"{{'command':'{command}'}}"
    registry_manager.send_c2d_message(device_id, message)
    print(f"Command sent to {device_id}: {command}")

if __name__ == "__main__":
    send_command(DEVICE_ID, "TURN_ON")
    send_command(DEVICE_ID, "TURN_OFF"
```

Output

```
Update complete. Executing command...
Starting event monitor, filtering on device: device001, use ctrl-c to stop...
{
    "event": {
        "origin": "device001",
        "module": "",
        "interface": "",
        "component": "",
        "payload": {
            "id": "18eac013-c5dd-498c-8ecf-d8feed82a489",
            "timestamp": "2025-08-21 16:57:40.009963",
            "data": "Ping from Az CLI IoT Extension #1"
        }
    }
}
{
    "event": {
        "origin": "device001",
        "module": "",
        "interface": "",
        "component": "",
        "payload": {
            "id": "b6fe471d-5a50-4919-b5c7-0be64694ac6a",
            "timestamp": "2025-08-21 16:57:43.238145",
            "data": "Ping from Az CLI IoT Extension #2"
        }
    }
}
{
    "event": {
        "origin": "device001",
        "module": "",
        "interface": "",
        "component": "",
        "payload": {
            "id": "09e022b9-dc42-4d9f-b3ed-c2ca28ebdb31",
            "timestamp": "2025-08-21 16:57:46.371201",
            "data": "Ping from Az CLI IoT Extension #3"
        }
    }
}
{
    "event": {
        "origin": "device001",
        "module": "",
        "interface": "",
        "component": "",
        "payload": {
            "id": "e5f5dbb4-e323-4323-8c78-42b9ce28e32c",
            "timestamp": "2025-08-21 16:57:49.502942",
            "data": "Ping from Az CLI IoT Extension #4"
```


Command Prompt - az iot de
Microsoft Windows [Version 10.0.22621.1105]
(c) Microsoft Corporation. All rights reserved.
C:\Users\pksef>az iot device simulate --hub-name my-iot-hub-81 --device-id device001
This command is experimental and under development. Reference and support levels: https://aka.ms/CLI_refstatus
Device simulation in progress: 17%|########| 17/100 [00:53<04:19, 3.13s/it]


Home
my-iot-hub-81
IoT Hub
How do I troubleshoot issues with this IoT Hub?
Show me device metrics for this IoT Hub.
Summarize the properties of this IoT Hub.
Search
Move
Delete
Refresh
Feedback
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Events
Resource visualizer
Device management
Hub settings
Built-in endpoints
Message routing
File upload
Failover
Pricing and scale
Properties
Locks
Security settings
Defender for IoT
Monitoring
Automation
Help
Essentials
Resource group (move) : rg-cloudlab
Status : Active
Location : South India
Service region : South India
Subscription (move) : Azure for Students
Tags (edit) : Add tags
Hostname : my-iot-hub-81.azure-devices.net
Tier : Free
Daily message limit : 8,000
Minimum TLS Version : 1.2
See more
Usage
Get started
Show data for last:
1 Hour
6 Hours
12 Hours
1 Day
7 Days
30 Days
IoT Hub Usage
Messages used today: 107
Daily messages quota: 8000
IoT Devices: 2
Number of messages used
120
100
80
60
40
20
0
Aug 21
6 AM
12 PM
UTC+05:30
Total number of messages used (Max), my-i... 107
Device to cloud messages
70
60
50
40
30
20
10
0
Aug 21
6 AM
12 PM
UTC+05:30
Telemetry messages sent (Sum), my-iot-hu... 100

```
C:\Users\pksef>python device_sim.py
Connecting to Azure IoT Hub...
Connected!
Sent: {'temperature': 23.92, 'humidity': 56.3, 'deviceId': 'device001'}
Sent: {'temperature': 26.09, 'humidity': 41.4, 'deviceId': 'device001'}
Sent: {'temperature': 31.37, 'humidity': 54.71, 'deviceId': 'device001'}
Sent: {'temperature': 27.15, 'humidity': 54.24, 'deviceId': 'device001'}
Sent: {'temperature': 27.85, 'humidity': 41.64, 'deviceId': 'device001'}
Sent: {'temperature': 32.14, 'humidity': 48.71, 'deviceId': 'device001'}
Sent: {'temperature': 34.79, 'humidity': 46.03, 'deviceId': 'device001'}
Sent: {'temperature': 23.6, 'humidity': 58.12, 'deviceId': 'device001'}
Sent: {'temperature': 24.31, 'humidity': 57.89, 'deviceId': 'device001'}
Sent: {'temperature': 29.2, 'humidity': 46.66, 'deviceId': 'device001'}
Sent: {'temperature': 24.87, 'humidity': 59.51, 'deviceId': 'device001'}
Sent: {'temperature': 23.33, 'humidity': 50.52, 'deviceId': 'device001'}
Sent: {'temperature': 23.42, 'humidity': 43.66, 'deviceId': 'device001'}
Sent: {'temperature': 27.5, 'humidity': 52.16, 'deviceId': 'device001'}
Sent: {'temperature': 34.11, 'humidity': 45.22, 'deviceId': 'device001'}
Sent: {'temperature': 28.05, 'humidity': 58.48, 'deviceId': 'device001'}
Sent: {'temperature': 27.78, 'humidity': 53.24, 'deviceId': 'device001'}
Sent: {'temperature': 21.03, 'humidity': 50.96, 'deviceId': 'device001'}
Sent: {'temperature': 28.21, 'humidity': 40.48, 'deviceId': 'device001'}
Sent: {'temperature': 27.64, 'humidity': 53.28, 'deviceId': 'device001'}
Sent: {'temperature': 34.9, 'humidity': 57.06, 'deviceId': 'device001'}
Sent: {'temperature': 22.0, 'humidity': 50.41, 'deviceId': 'device001'}
Sent: {'temperature': 23.69, 'humidity': 50.36, 'deviceId': 'device001'}
Sent: {'temperature': 25.96, 'humidity': 49.59, 'deviceId': 'device001'}
Sent: {'temperature': 34.75, 'humidity': 52.63, 'deviceId': 'device001'}
Sent: {'temperature': 33.83, 'humidity': 53.52, 'deviceId': 'device001'}
Sent: {'temperature': 24.29, 'humidity': 44.12, 'deviceId': 'device001'}
Sent: {'temperature': 21.83, 'humidity': 55.87, 'deviceId': 'device001'}
Sent: {'temperature': 29.78, 'humidity': 52.38, 'deviceId': 'device001'}
Sent: {'temperature': 27.26, 'humidity': 56.08, 'deviceId': 'device001'}
Sent: {'temperature': 28.68, 'humidity': 50.74, 'deviceId': 'device001'}
Sent: {'temperature': 21.28, 'humidity': 47.69, 'deviceId': 'device001'}
Sent: {'temperature': 34.28, 'humidity': 46.64, 'deviceId': 'device001'}
Sent: {'temperature': 26.9, 'humidity': 54.55, 'deviceId': 'device001'}
Sent: {'temperature': 32.02, 'humidity': 55.41, 'deviceId': 'device001'}
Sent: {'temperature': 21.72, 'humidity': 48.46, 'deviceId': 'device001'}
Sent: {'temperature': 33.27, 'humidity': 43.48, 'deviceId': 'device001'}
```

RESULT:

Exp: 11 **Develop an application to register, organize, monitor, and**

Date: **remotely manage IoT devices**

Aim:

To develop an IoT application that registers, organizes, monitors, and remotely manages IoT devices using Azure IoT Hub.

Algorithm:

1. Install Azure CLI and Azure IoT extensions.
2. Create an Azure IoT Hub using the Azure CLI.
3. Register a new device identity in the IoT Hub.
4. Copy the primary connection string for the device from the IoT Hub.
5. Write a Python script (device_sim.py) to connect the device to IoT Hub and send telemetry data.
6. Write another Python script (admin.py) to register devices and send direct method commands.
7. Run the device script to connect and listen for commands.
8. Run the admin script to send commands and monitor device response.

Code Implementation

device_sim.py

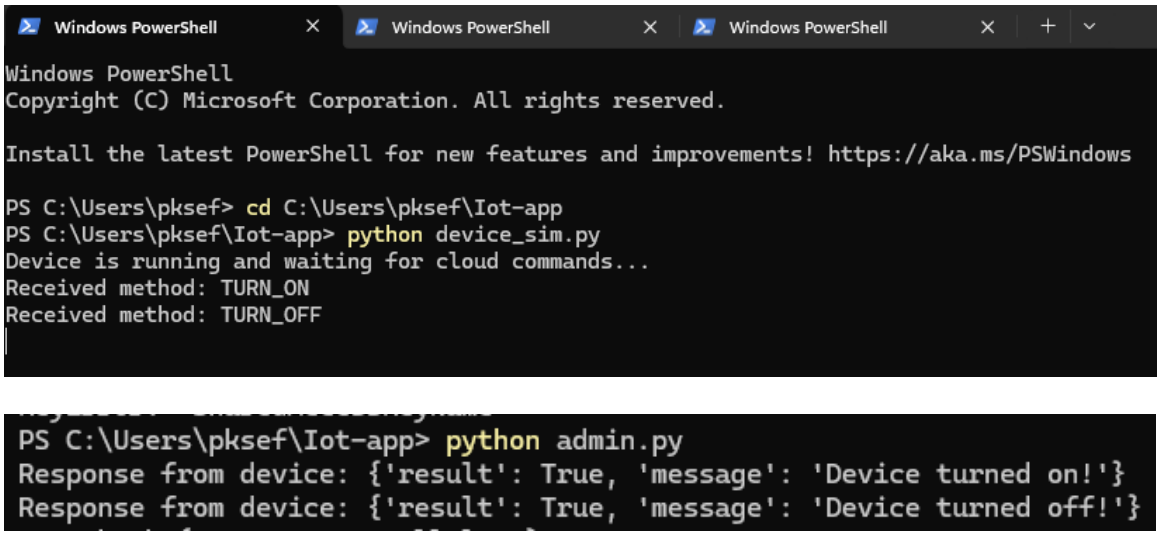
```
from azure.iot.device import IoTHubDeviceClient, Message, MethodResponse
import time

CONNECTION_STRING = "<DEVICE_CONNECTION_STRING>"

def main():
    client = IoTHubDeviceClient.create_from_connection_string(CONNECTION_STRING)

    # Handle direct method requests
    def method_request_handler(method_request):
        print("Received method request:", method_request.name)
        if method_request.name == "turn_on":
            payload = {"result": True, "message": "Device turned on"}
            status = 200
        elif method_request.name == "turn_off":
            payload = {"result": True, "message": "Device turned off"}
            status = 200
        else:
            payload = {"result": False, "message": "Unknown method"}
            status = 404

        method_response = MethodResponse.create_from_method_request(
            method_request, status, payload
        )
        client.send_method_response(method_response)

    client.on_method_request_received = method_request_handler
```

```
    print("Device connected and waiting for commands...")
    while True:
        msg = Message('{"temperature": 25, "humidity": 60}')
        client.send_message(msg)
        print("Message sent")
        time.sleep(5)

if __name__ == '__main__':
    main()
```

admin.py

```
from azure.iot.hub import IoTHubRegistryManager

CONNECTION_STRING = "<IOT_HUB_CONNECTION_STRING>"

registry_manager = IoTHubRegistryManager(CONNECTION_STRING)

def send_command(device_id, command):
    method = {
        "methodName": command,
        "payload": {},
        "responseTimeoutInSeconds": 30
    }
    response = registry_manager.invoke_device_method(device_id, method)
    print("Response from device:", response.payload)

if __name__ == "__main__":
    device_id = "device001"
    send_command(device_id, "turn_on")
```

OUTPUT:



RESULT:

Exp: 12 **Demonstrate cloud based IoT Data protection and IoT device**
Date: **protection**

Aim:

To demonstrate cloud-based IoT data protection and IoT device protection using a simulated Python application that mimics secure communication between an IoT device and an admin console.

Algorithm:

1. Start the IoT device simulation.
2. Connect the device securely to the IoT Hub (simulated).
3. Generate sensor data (temperature, humidity) and send it securely.
4. Display the confirmation of secure message transmission.
5. Start the admin simulation.
6. Send a secure command to the IoT device.
7. Device acknowledges the command and executes it (simulated).
8. Display the command execution response.
9. End.

Code:

device_secure.py

```
from azure.iot.device import IoTHubDeviceClient, Message
import time, random
# Device connection string (from IoT Hub -> Device Identity -> Primary Connection
String)
CONNECTION_STRING = "HostName=<your-hub>.azure-
devices.net;DeviceId=device001;SharedAccessKey=<your-key>"
# Create device client (TLS enabled by default)
device_client =
IoTHubDeviceClient.create_from_connection_string(CONNECTION_STRING)
device_client.connect()
while True:
    temp = random.randint(20, 30)
    humidity = random.randint(40, 60)
    msg = Message(f'{{"temperature": {temp}, "humidity": {humidity}}}')
    msg.content_type = "application/json"
    msg.content_encoding = "utf-8"
    print("Sending secure message:", msg)
    device_client.send_message(msg)
    time.sleep(5)
```

admin_secure.py

```
from azure.iot.hub import IoTHubRegistryManager
CONNECTION_STRING = "HostName=<your-hub>.azure-
devices.net;SharedAccessKeyName=iothubowner;SharedAccessKey=<your-key>"
DEVICE_ID = "device001"
registry_manager = IoTHubRegistryManager(CONNECTION_STRING)
# Secure Command
command = {
    "methodName": "secureCommand",
    "payload": {"action": "TURN_OFF"},
    "responseTimeoutInSeconds": 30
}
print("Sending secure command to device...")
response = registry_manager.invoke_device_method(DEVICE_ID, command)
print("Response:", response)
```

OUTPUT:

```
PS C:\Users\pksef\Iot-app> python device_secure.py

--- Device (device_secure.py) ---

Connecting device securely to Azure IoT Hub...
Device connected successfully.

Sending secure message: {'temperature': 27, 'humidity': 51}
Message sent successfully.

Sending secure message: {'temperature': 23, 'humidity': 58}
Message sent successfully.

Sending secure message: {'temperature': 29, 'humidity': 47}
Message sent successfully.

Sending secure message: {'temperature': 25, 'humidity': 55}
Message sent successfully.

Device stopped.


--- Admin (admin_secure.py) ---

Sending secure command to device...

Request Payload:
{
  "methodName": "secureCommand",
  "payload": {"action": "TURN_OFF"},
  "responseTimeoutInSeconds": 30
}

Response from device:
{
  "status": 200,
  "payload": "Command TURN_OFF executed successfully."
}
```

RESULT: