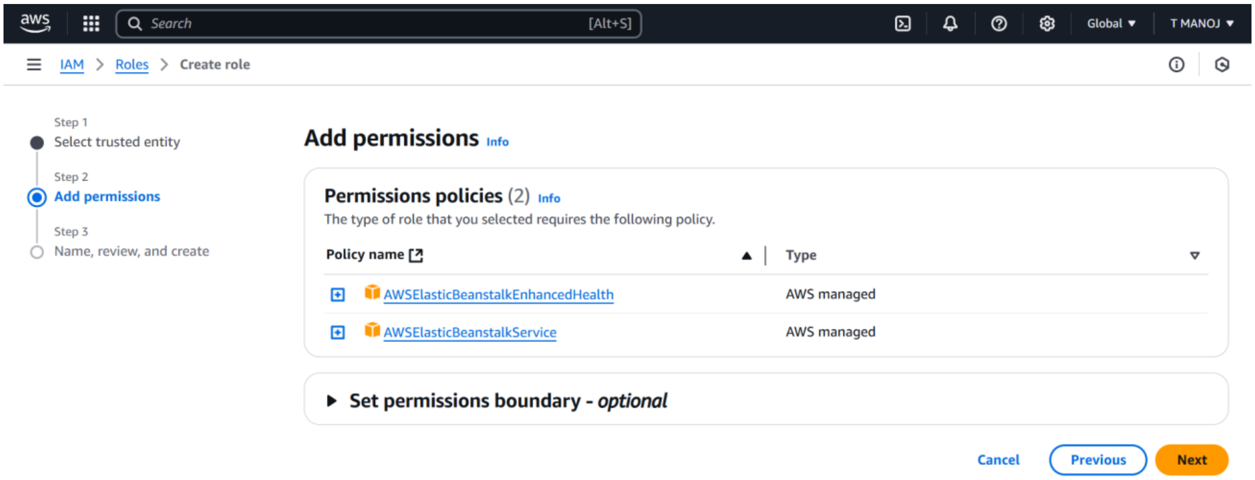
Deploying a Sample Python App on AWS Elastic Beanstalk

Objective:

This document provides a step-by-step guide to deploying a sample Python application using AWS Elastic Beanstalk, creating a new IAM user role, setting up an environment, and deploying the application.

Create a New IAM User Role

1. Sign in to AWS Console and navigate to IAM (Identity and Access Management).
2. Click on Roles > Create role.
3. Select AWS service and choose Elastic Beanstalk.
4. Click Next and attach the policy AWSElasticBeanstalkFullAccess.
5. Provide a Role name (e.g., `ElasticBeanstalkRole`).
6. Click Create role.



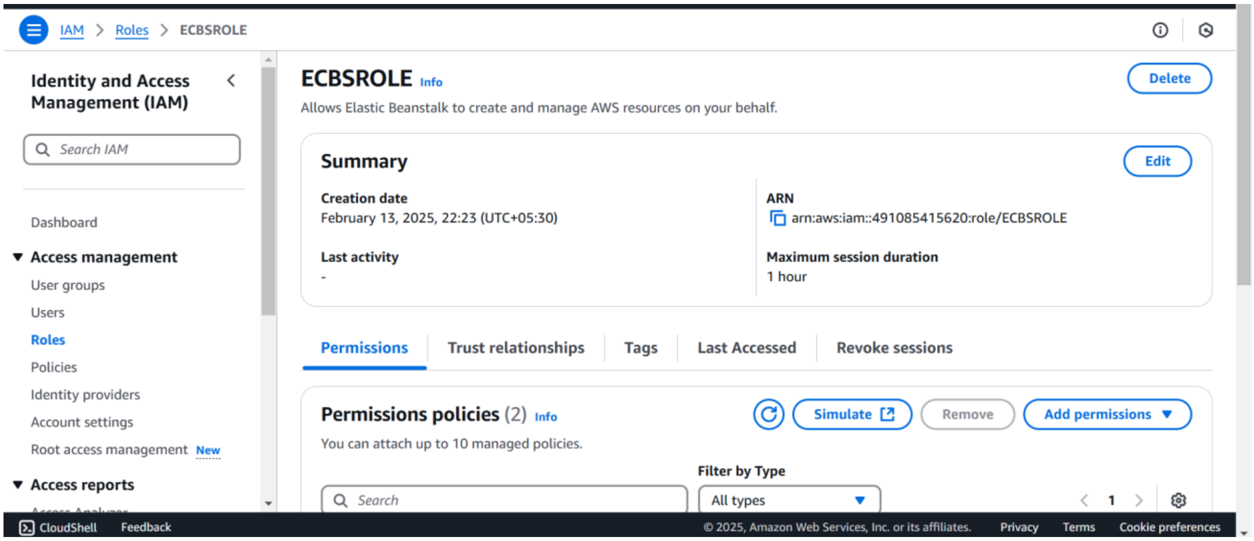
Add permissions

Permissions policies (2)

The type of role that you selected requires the following policy.

Policy name	Type
AWSElasticBeanstalkEnhancedHealth	AWS managed
AWSElasticBeanstalkService	AWS managed

Set permissions boundary - *optional*



ECBSROLE

Allows Elastic Beanstalk to create and manage AWS resources on your behalf.

Summary

Creation date: February 13, 2025, 22:23 (UTC+05:30)

ARN: arn:aws:iam::491085415620:role/ECBSROLE

Last activity: -

Maximum session duration: 1 hour

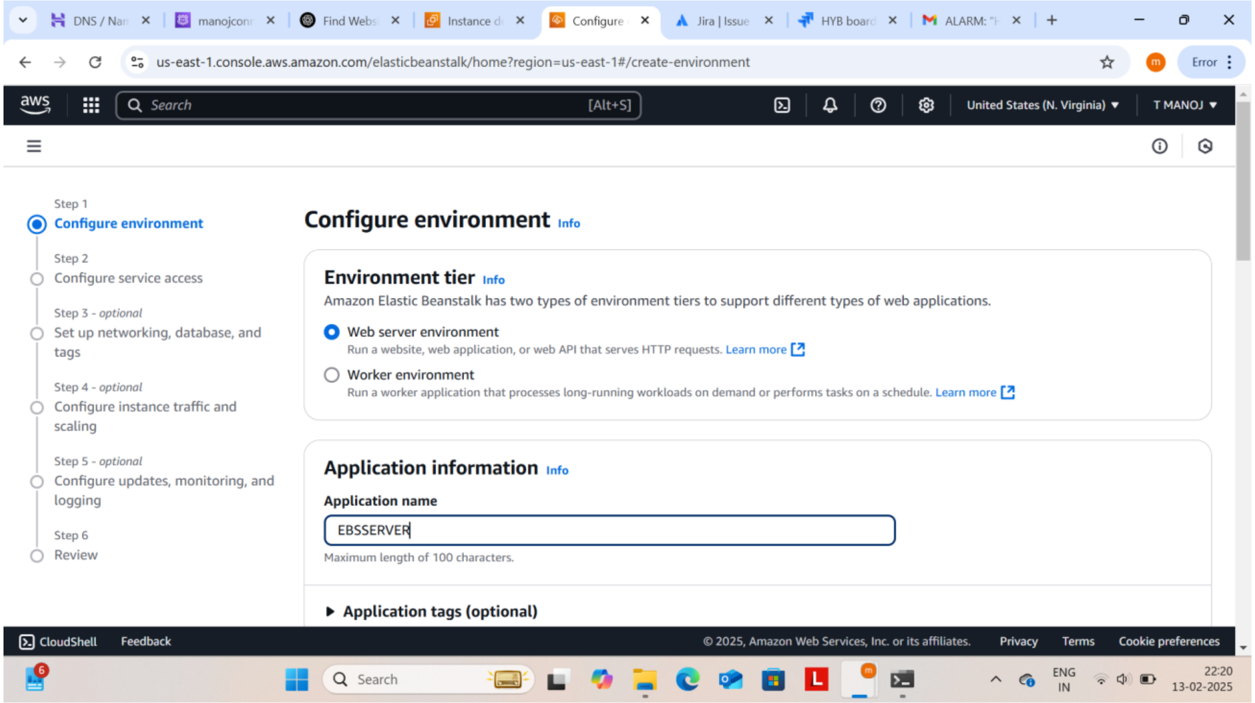
Permissions policies (2)

You can attach up to 10 managed policies.

Filter by Type: All types

Create an Elastic Beanstalk Environment

1. Go to AWS Elastic Beanstalk in the AWS Console.
2. Click Create a new environment.
3. Choose Web server environment.
4. Set the Application name (e.g., `SamplePythonApp`).
5. Select Platform as `Python` and choose the latest version.
6. Click Create environment and wait for it to launch.



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

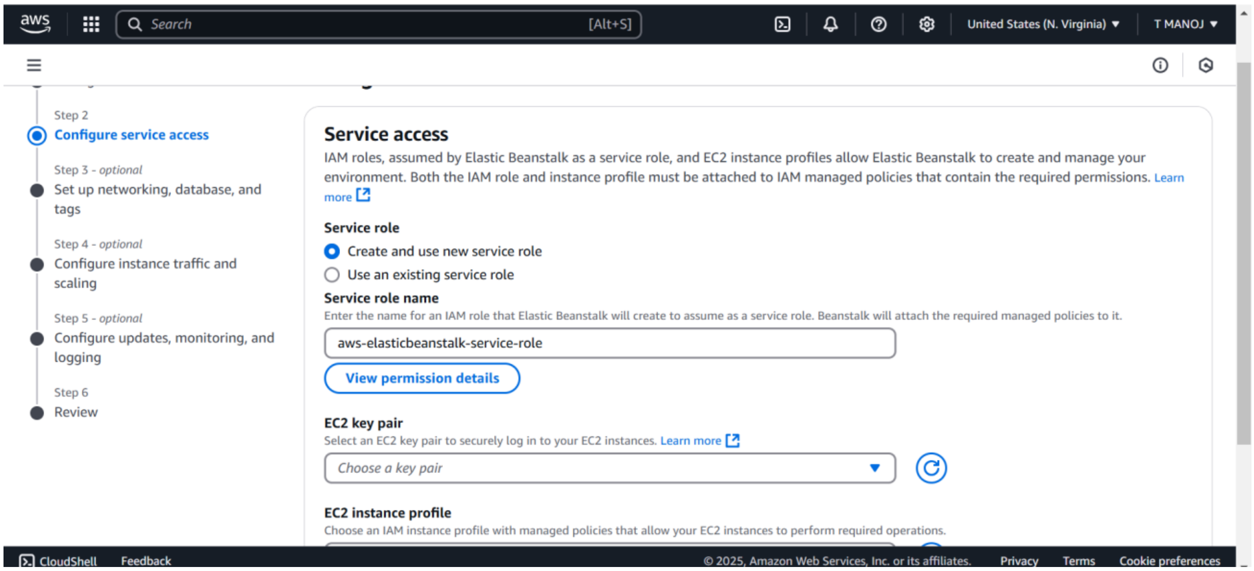
Application information Info

Application name

EBSSERVER

Maximum length of 100 characters.

▶ **Application tags (optional)**


aws
Search [Alt+S]
United States (N. Virginia) ▼
T MANOJ ▼
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

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

Service access

IAM roles, assumed by Elastic Beanstalk as a service role, and EC2 instance profiles allow Elastic Beanstalk to create and manage your environment. Both the IAM role and instance profile must be attached to IAM managed policies that contain the required permissions. Learn more

Service role

● Create and use new service role
○ Use an existing service role

Service role name
Enter the name for an IAM role that Elastic Beanstalk will create to assume as a service role. Beanstalk will attach the required managed policies to it.

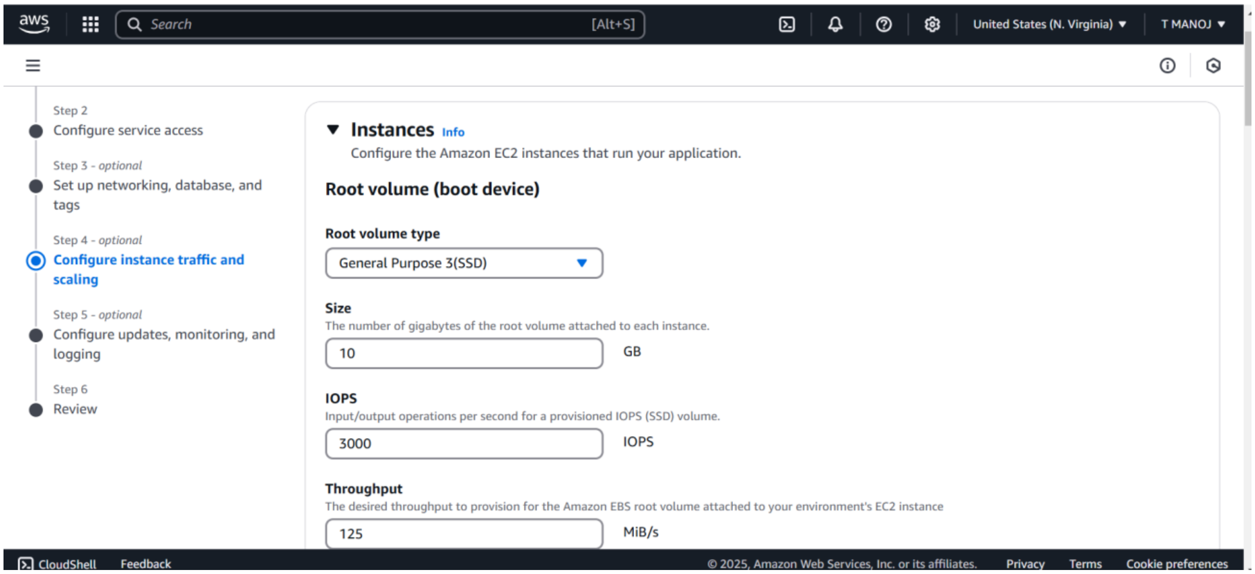
aws-elasticbeanstalk-service-role

View permission details

EC2 key pair
Select an EC2 key pair to securely log in to your EC2 instances. Learn more

Choose a key pair ▼

EC2 instance profile
Choose an IAM instance profile with managed policies that allow your EC2 instances to perform required operations.


aws
Search
[Alt+S]
United States (N. Virginia) ▾
T MANOJ ▾
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

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

Instances Info

Configure the Amazon EC2 instances that run your application.

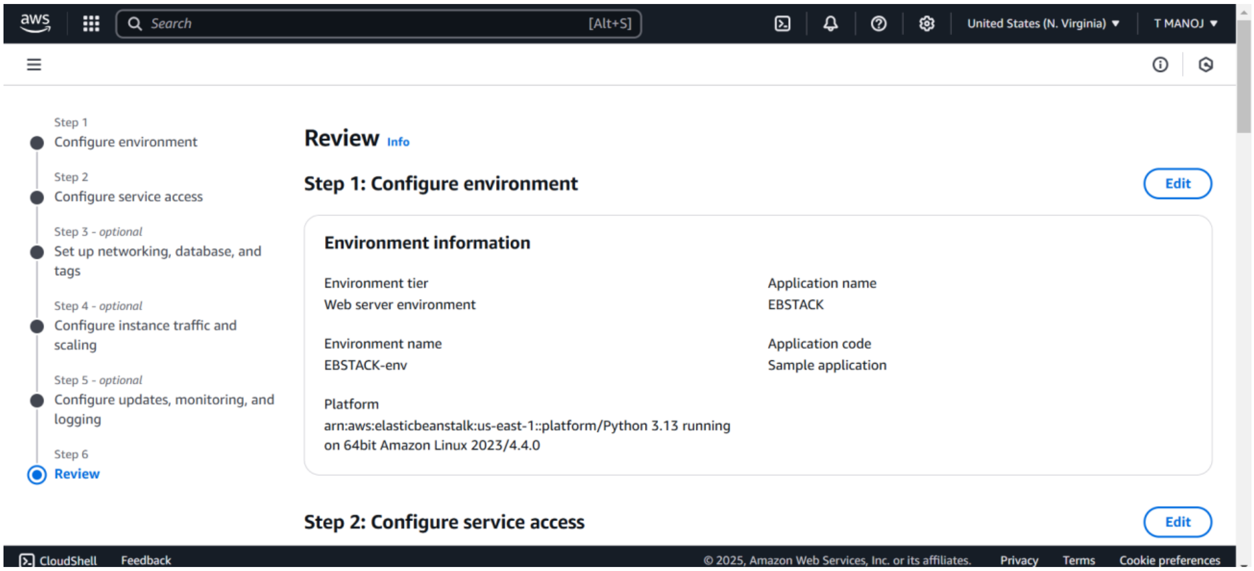
Root volume (boot device)

Root volume type
General Purpose 3(SSD) ▾

Size
The number of gigabytes of the root volume attached to each instance.
10 GB

IOPS
Input/output operations per second for a provisioned IOPS (SSD) volume.
3000 IOPS

Throughput
The desired throughput to provision for the Amazon EBS root volume attached to your environment's EC2 instance
125 MiB/s


aws
Search
[Alt+S]
United States (N. Virginia) ▾
T MANOJ ▾
Edit
Edit
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

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

Review Info

Step 1: Configure environment

Environment information

Environment tier
Web server environment

Application name
EBSTACK

Environment name
EBSTACK-env

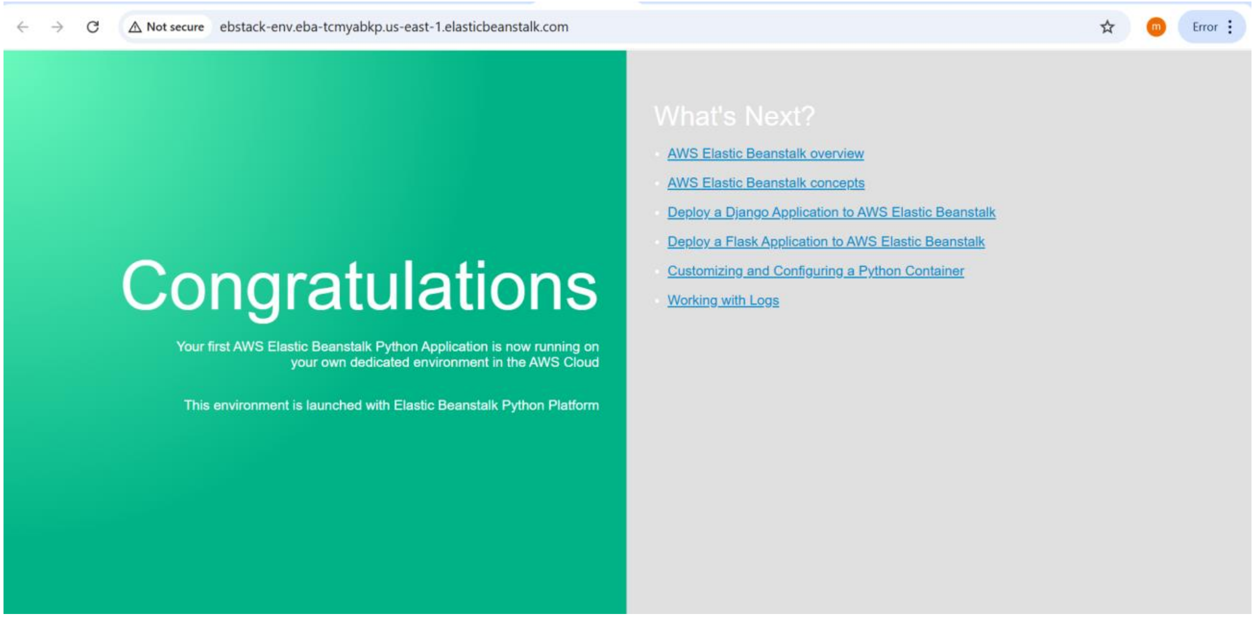
Application code
Sample application

Platform
arn:aws:elasticbeanstalk:us-east-1::platform/Python 3.13 running on 64bit Amazon Linux 2023/4.4.0

Step 2: Configure service access

Verify the Default Application

1. Once the environment is running, click on the URL generated by Elastic Beanstalk.
2. Verify that the default Elastic Beanstalk page is displayed.



Congratulations

Your first AWS Elastic Beanstalk Python Application is now running on your own dedicated environment in the AWS Cloud

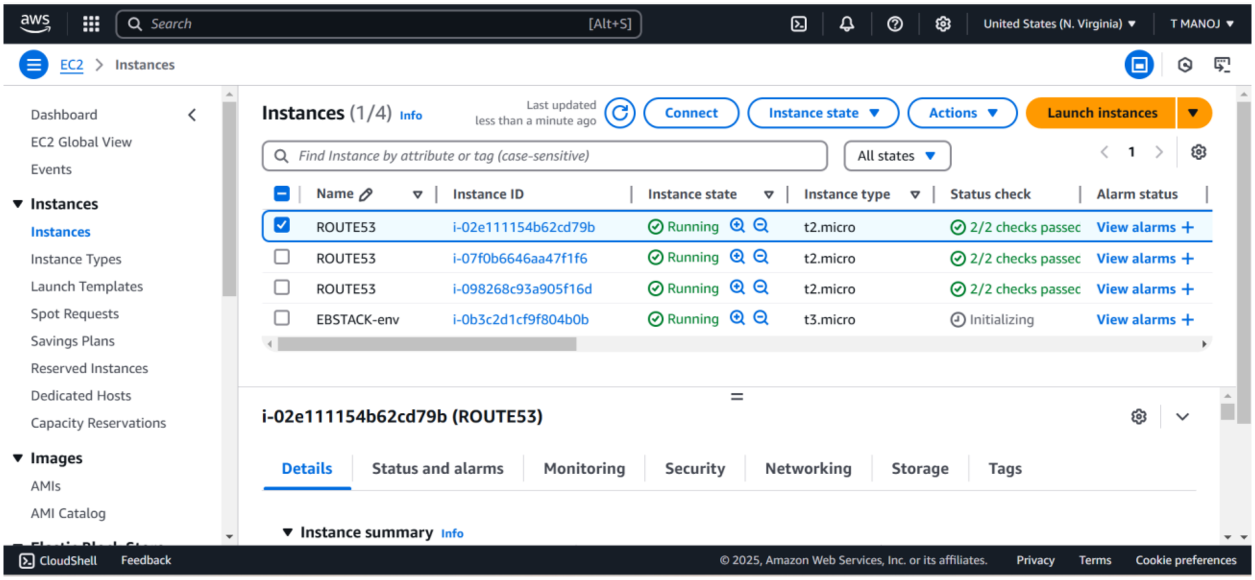
This environment is launched with Elastic Beanstalk Python Platform

What's Next?

- AWS Elastic Beanstalk overview
- AWS Elastic Beanstalk concepts
- Deploy a Django Application to AWS Elastic Beanstalk
- Deploy a Flask Application to AWS Elastic Beanstalk
- Customizing and Configuring a Python Container
- Working with Logs

Upload and Deploy a Python Application

1. Prepare a sample Python application (e.g., `app.py`).
2. Navigate to the Elastic Beanstalk Dashboard.
3. Click Upload and Deploy.
4. Choose your Python application ZIP file.
5. Click Deploy and wait for the deployment to complete.
6. Once deployed, click the Application URL to verify that your Python app is running.



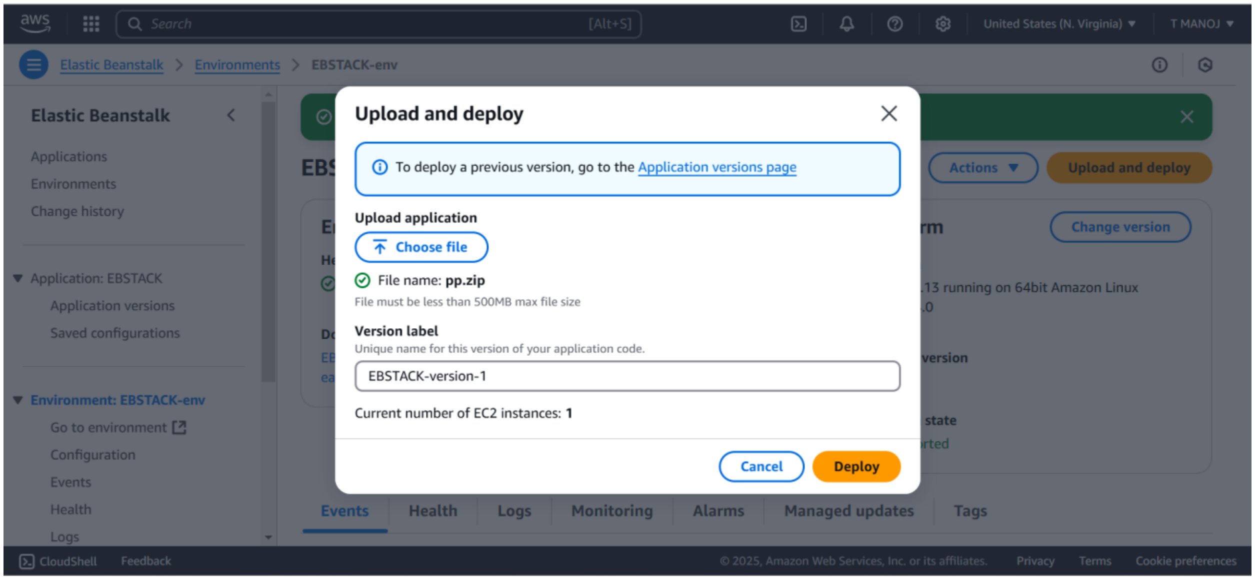
Instances (1/4) Info

☐	Name	Instance ID	Instance state	Instance type	Status check	Alarm status
☑	ROUTE53	i-02e111154b62cd79b	✓ Running	t2.micro	✓ 2/2 checks passec	View alarms +
☐	ROUTE53	i-07f0b6646aa47f1f6	✓ Running	t2.micro	✓ 2/2 checks passec	View alarms +
☐	ROUTE53	i-098268c93a905f16d	✓ Running	t2.micro	✓ 2/2 checks passec	View alarms +
☐	EBSTACK-env	i-0b3c2d1cf9f804b0b	✓ Running	t3.micro	Initializing	View alarms +

i-02e111154b62cd79b (ROUTE53)

Details | Status and alarms | Monitoring | Security | Networking | Storage | Tags

▼ Instance summary Info



Upload and deploy

ⓘ To deploy a previous version, go to the Application versions page

Upload application

Choose file

✓ File name: **pp.zip**

File must be less than 500MB max file size

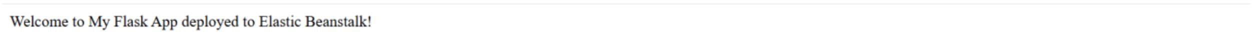
Version label

Unique name for this version of your application code.

EBSTACK-version-1

Current number of EC2 instances: **1**

Cancel | Deploy



Welcome to My Flask App deployed to Elastic Beanstalk!

Conclusion

Deploying a sample Python app on AWS Elastic Beanstalk . You have created an IAM role, launched an Elastic Beanstalk environment, and deployed a Python application successfully.