Adv. Devops Experiment no. 2

Name: Rohan Lalchandani
Class: D15A Roll no.: 25

Aim: To Build Your Application using AWS CodeBuild and Deploy on S3 / SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS CodeDeploy.

Theory:

Amazon Elastic Beanstalk is a Platform-as-a-Service (PaaS) offered by AWS that simplifies the deployment, management, and scaling of applications. It abstracts the underlying infrastructure, allowing developers to focus on writing code rather than managing servers and infrastructure.

Key Features

1. Easy Deployment:

- **Deployment Options:** Supports multiple deployment methods, including the AWS Management Console, CLI, and SDKs. You can deploy applications using ZIP files, Docker images, or from a source control repository.
- **Managed Platform Updates:** Automatically handles platform updates and patches for the underlying infrastructure.

2. Application Management:

- **Environment Management:** Provides pre-configured environments for popular application platforms like Node.js, Python, Java, .NET, PHP, Ruby, and Docker.
- **Monitoring and Logging:** Integrated with Amazon CloudWatch and AWS X-Ray for monitoring performance and logging. Offers health monitoring and application logs through the Elastic Beanstalk console.

3. Auto Scaling and Load Balancing:

- **Auto Scaling:** Automatically adjusts the number of instances based on traffic and resource utilization.

- **Load Balancing:** Distributes incoming application traffic across multiple instances to ensure high availability.

4. Customization and Flexibility:

- **Configuration Options:** Allows customization through configuration files (e.g., `.ebextensions`) and environment variables. You can configure instance types, scaling policies, and load balancer settings.
- **Integration with AWS Services:** Easily integrates with other AWS services like Amazon RDS (Relational Database Service), S3 (Simple Storage Service), and IAM (Identity and Access Management).

5. Environment Types:

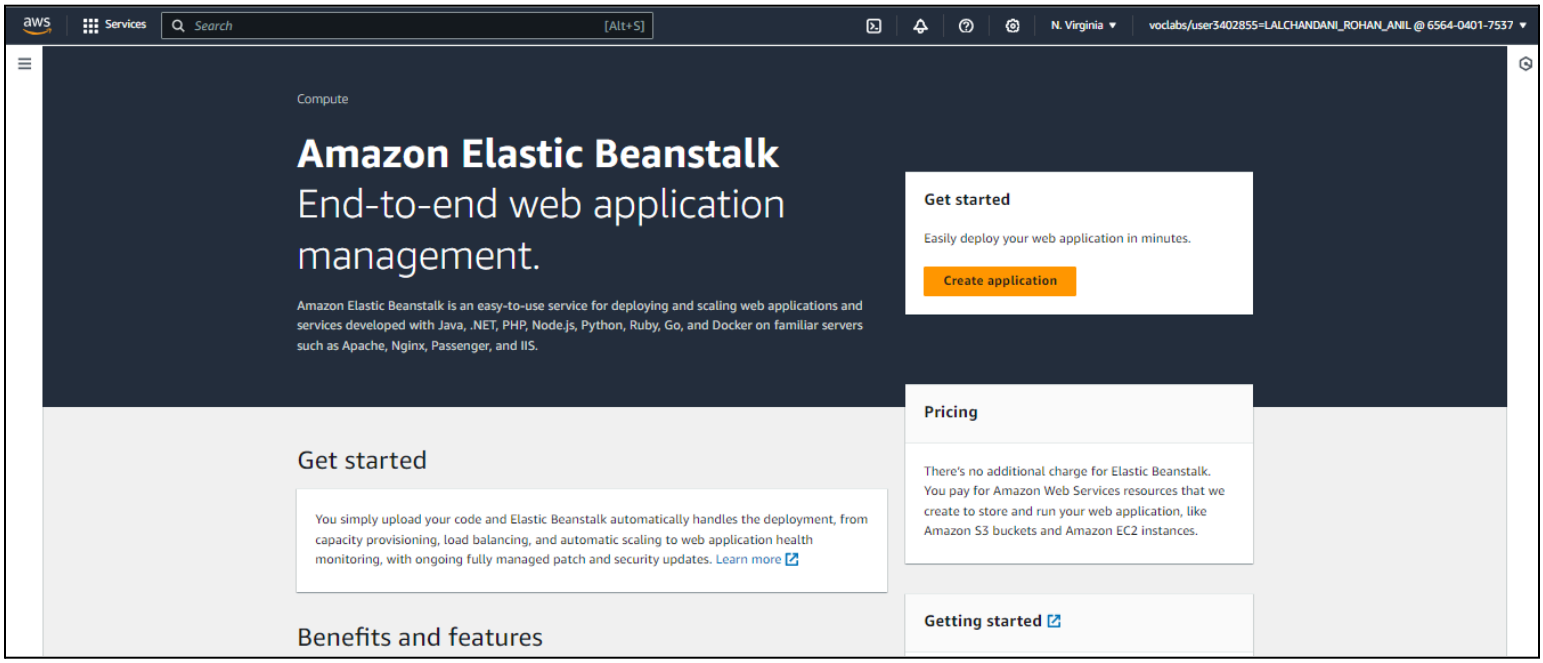
- **Single Instance Environment:** Suitable for development and testing where high availability and fault tolerance are not required.
- **Load Balanced Environment:** Designed for production with auto-scaling, load balancing, and high availability.

Common Use Cases

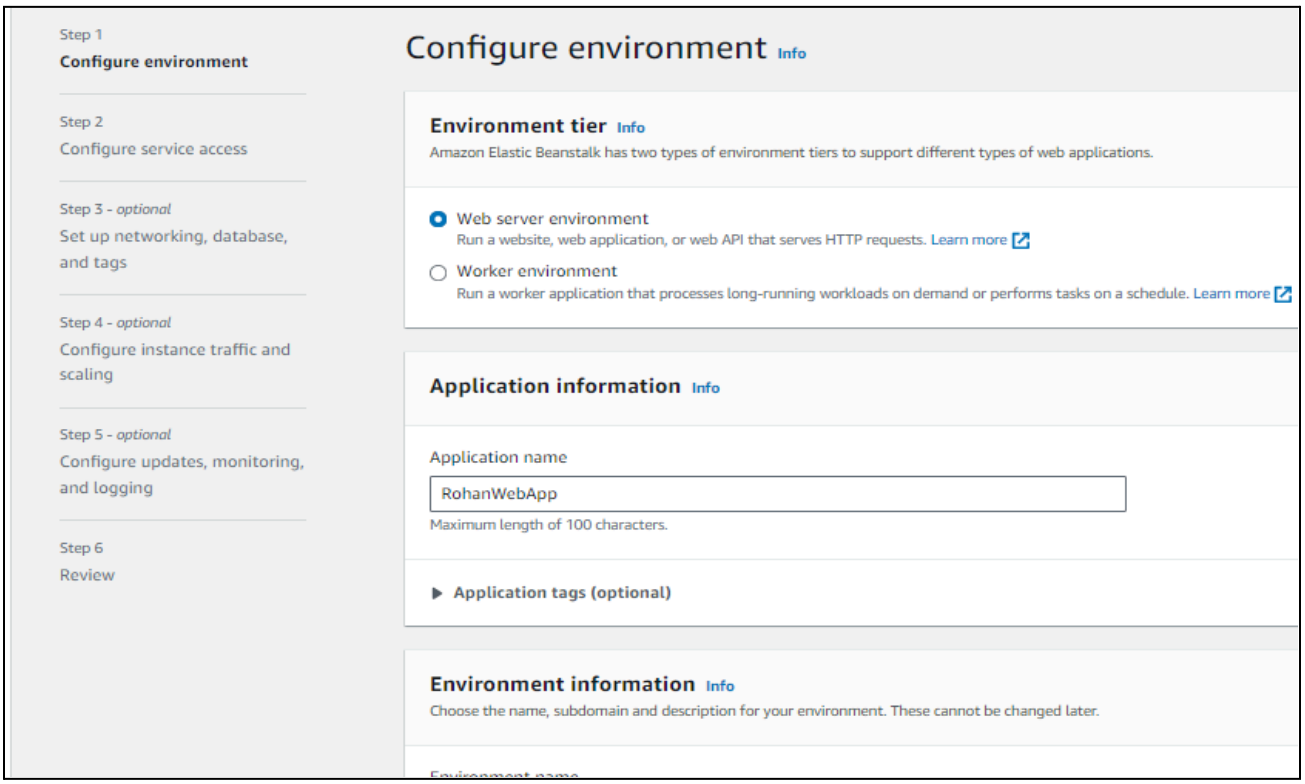
- **Web Applications:** Deploy and manage web applications with a scalable backend and frontend.
- **APIs:** Host RESTful APIs or microservices with auto-scaling and high availability.
- **Development and Testing:** Quickly spin up environments for development and testing purposes.

Implementation:

1) Search for Elastic Beanstalk in the search box and click on create application.



2) Configuring the environment.



Platform Info

Platform type

- Managed platform
  Platforms published and maintained by Amazon Elastic Beanstalk. Learn more
- Custom platform
  Platforms created and owned by you. This option is unavailable if you have no platforms.

Platform

Python

Platform branch

Python 3.11 running on 64bit Amazon Linux 2023

Platform version

4.1.3 (Recommended)

Application code Info

- Sample application
- Existing version
  Application versions that you have uploaded.
- Upload your code
  Upload a source bundle from your computer or copy one from Amazon S3.

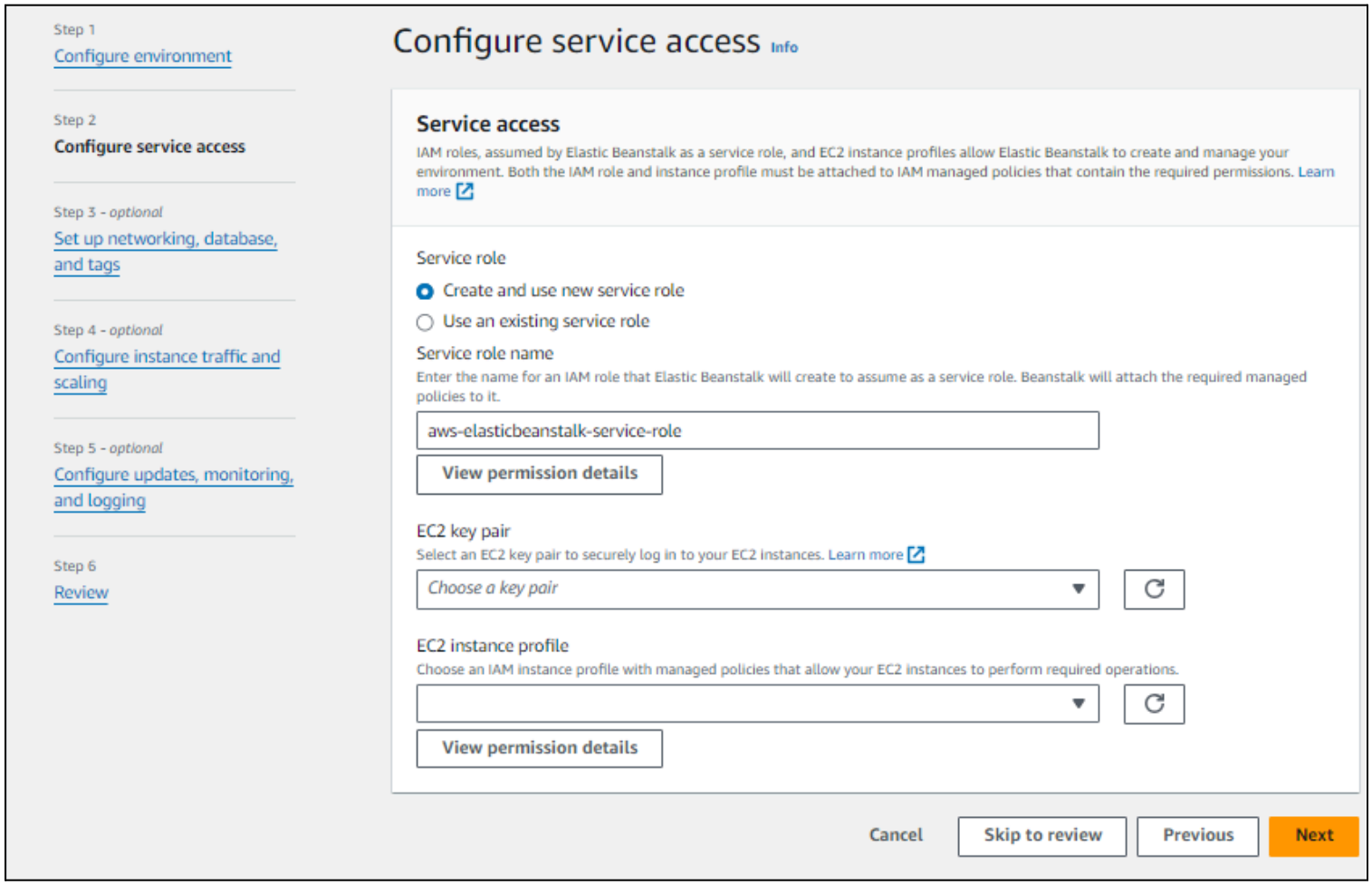
Presets Info

Start from a preset that matches your use case or choose custom configuration to unset recommended values and use the service's default values.

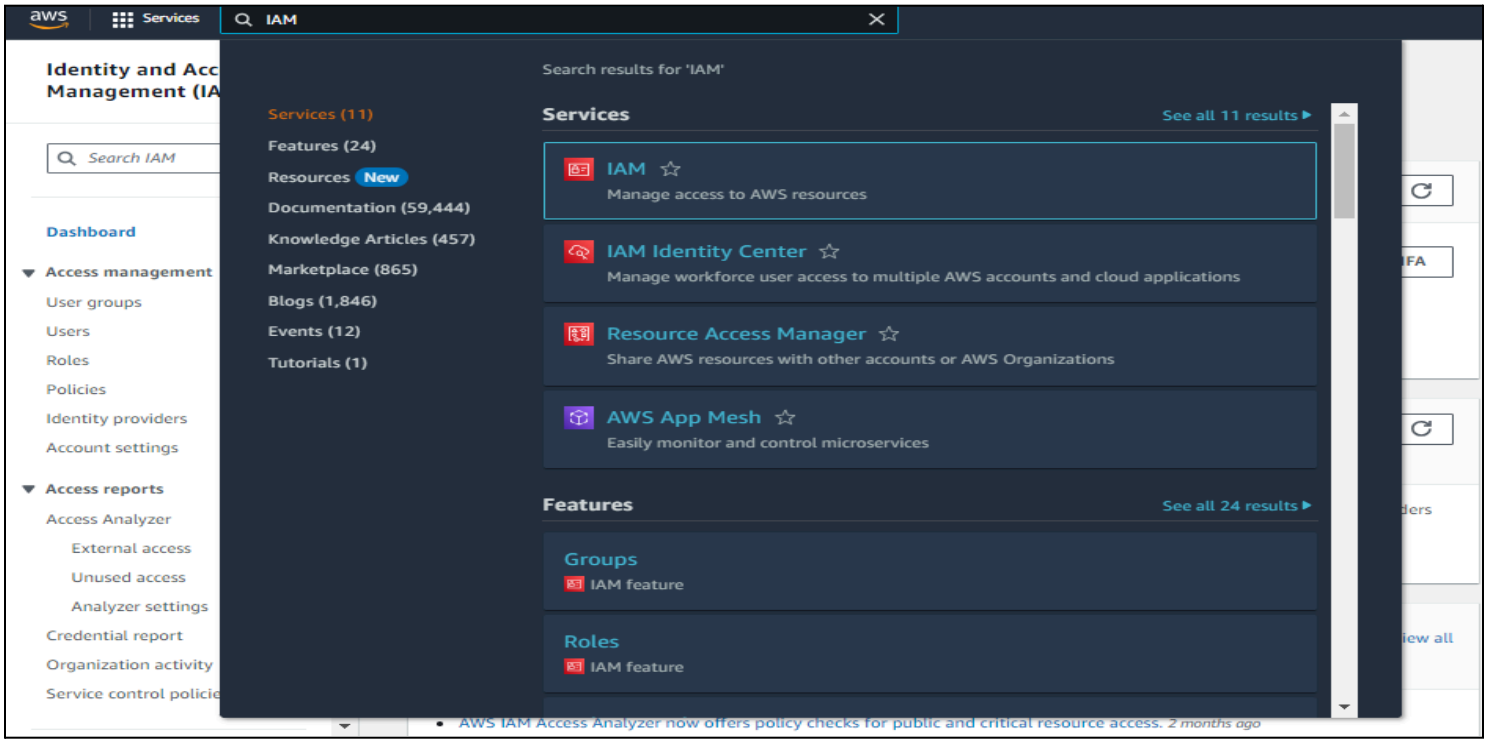
Configuration presets

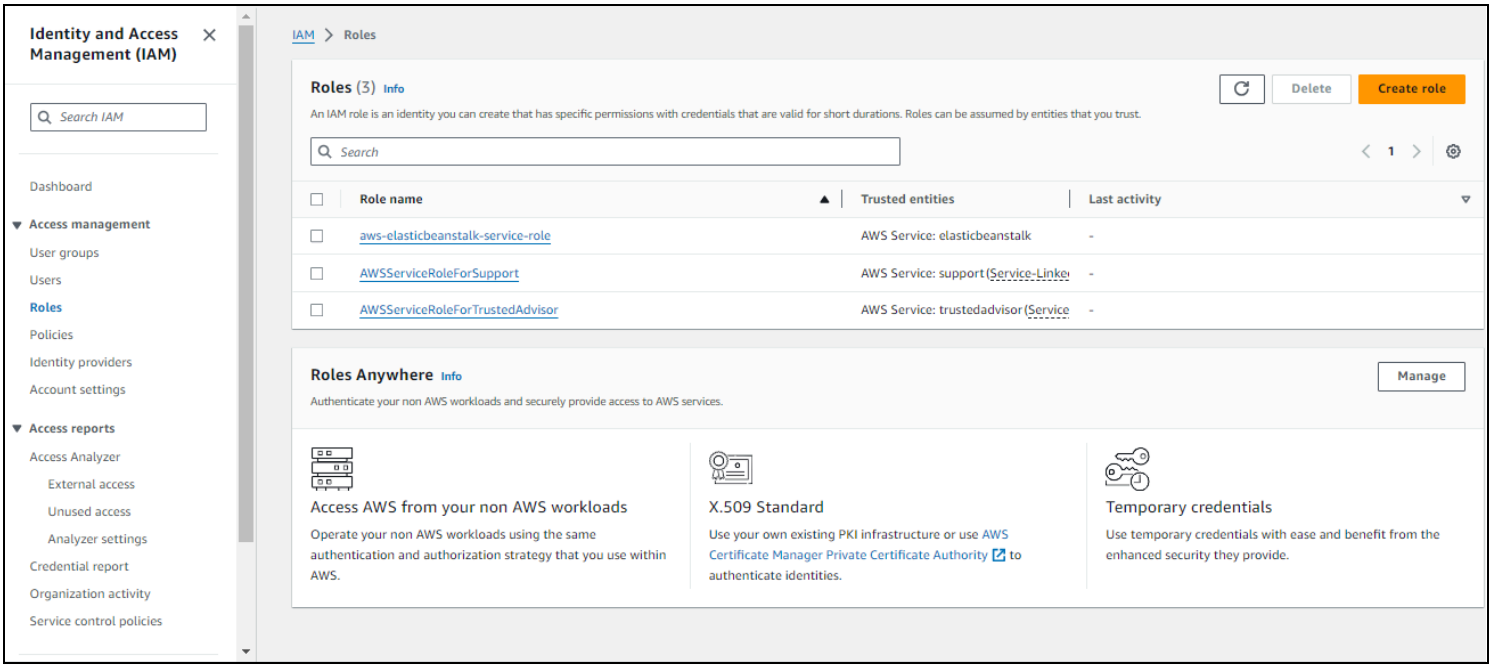
- Single instance (free tier eligible)
- Single instance (using spot instance)
- High availability
- High availability (using spot and on-demand instances)
- Custom configuration

3) Click on create service role and for selecting EC2 instance profile we need to create an IAM policy for it.

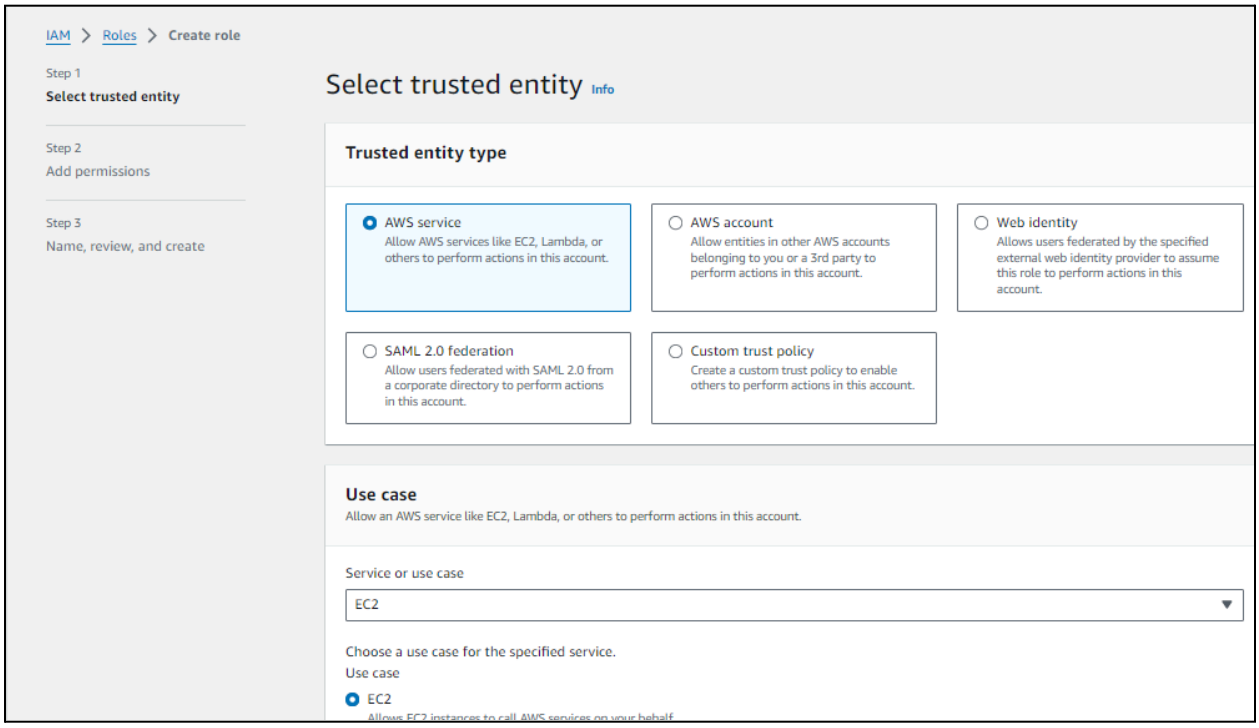


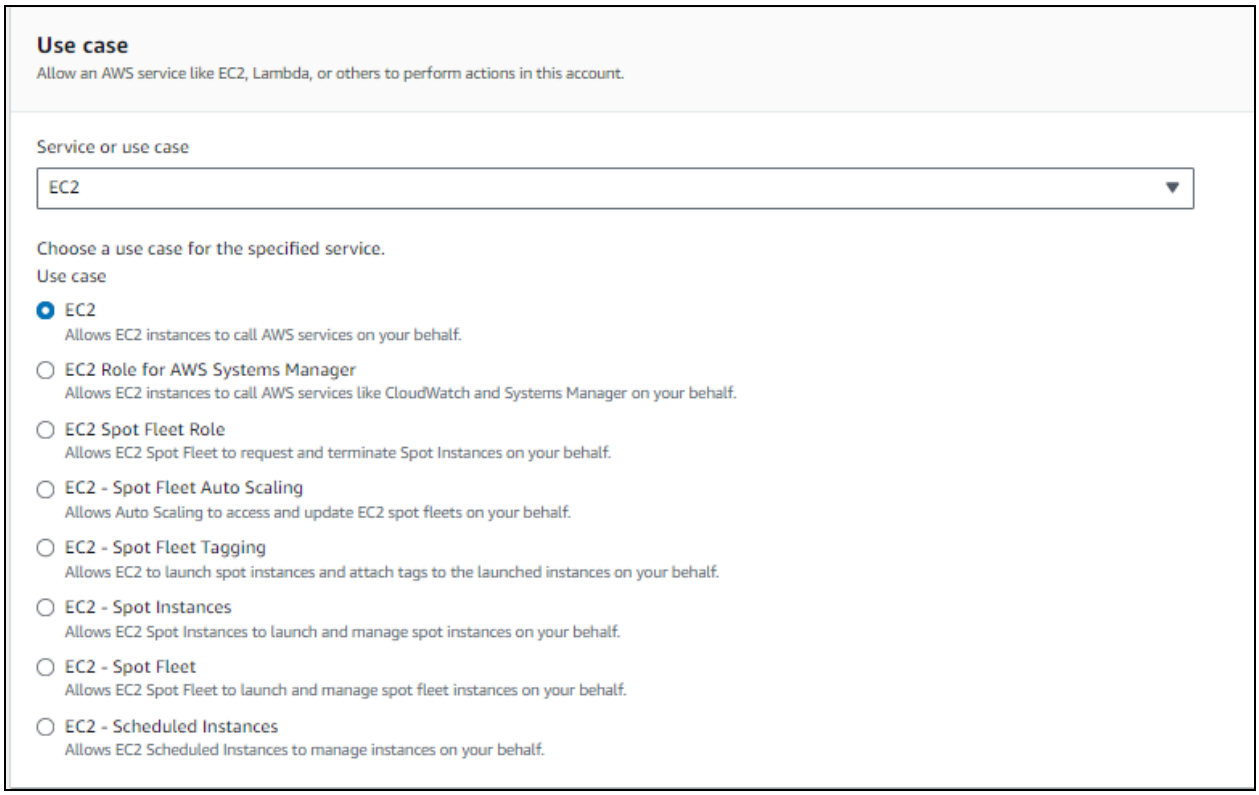
4) Search for IAM in the search box and click on “roles” on the left hand side bar and click on create role.





5) Configuring the IAM role.





6) We need to select three permission policies as shown below.

Step 2
Add permissions

Step 3
Name, review, and create

Permissions policies (3/947) Info

Choose one or more policies to attach to your new role.

Filter by Type

elasticbean | All types | 14 matches | 1

Policy name	Type	Description
AdministratorAccess-AWSElasticBeanstalk	AWS managed	Grants account administrative permissio...
AWSElasticBeanstalkCustomPlatformforEC2Role	AWS managed	Provide the instance in your custom plat...
AWSElasticBeanstalkEnhancedHealth	AWS managed	AWS Elastic Beanstalk Service policy for ...
AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy	AWS managed	This policy is for the AWS Elastic Beanst...
AWSElasticBeanstalkMulticontainerDocker	AWS managed	Provide the instances in your multiconta...
AWSElasticBeanstalkReadOnly	AWS managed	Grants read-only permissions. Explicitly ...
AWSElasticBeanstalkRoleCore	AWS managed	AWSElasticBeanstalkRoleCore (Elastic Be...
AWSElasticBeanstalkRoleCWL	AWS managed	(Elastic Beanstalk operations role) Allow...
AWSElasticBeanstalkRoleECS	AWS managed	(Elastic Beanstalk operations role) Allow...
AWSElasticBeanstalkRoleRDS	AWS managed	(Elastic Beanstalk operations role) Allow...
AWSElasticBeanstalkRoleSNS	AWS managed	(Elastic Beanstalk operations role) Allow...
AWSElasticBeanstalkRoleWorkerTier	AWS managed	(Elastic Beanstalk operations role) Allow...
AWSElasticBeanstalkWebTier	AWS managed	Provide the instances in your web server...
AWSElasticBeanstalkWorkerTier	AWS managed	Provide the instances in your worker env...

IAM > Roles > Create role

Step 1
Select trusted entity

Step 2
Add permissions

Step 3
Name, review, and create

Name, review, and create

Role details

Role name
Enter a meaningful name to identify this role.

aws-elasticbeanstalk-ec2-role

Maximum 64 characters. Use alphanumeric and '+=,.@-_' characters.

Description
Add a short explanation for this role.

Allows EC2 instances to call AWS services on your behalf.

Maximum 1000 characters. Use letters (A-Z and a-z), numbers (0-9), tabs, new lines, or any of the following characters: _+=,.@-/\[{}]:#$%^*();:"'`

Step 1: Select trusted entities Edit

Trust policy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRole"
            ],
            "Principal": {
                "Service": [
                    "ec2.amazonaws.com"
```

Step 1: Select trusted entities Edit

Trust policy

```
1  {
2      "Version": "2012-10-17",
3      "Statement": [
4          {
5              "Effect": "Allow",
6              "Action": [
7                  "sts:AssumeRole"
8              ],
9              "Principal": {
10                 "Service": [
11                     "ec2.amazonaws.com"
12                 ]
13             }
14         }
15     ]
16 }
```

Step 2: Add permissions Edit

Permissions policy summary

Policy name	Type	Attached as
AWSElasticBeanstalkMulticontainerDocker	AWS managed	Permissions policy
AWSElasticBeanstalkWebTier	AWS managed	Permissions policy
AWSElasticBeanstalkWorkerTier	AWS managed	Permissions policy

7) IAM Role for EC2 instance is successfully created, go back to “Configuring Service access” for elastic beanstalk and select the EC2 instance profile.

Role aws-elasticbeanstalk-ec2-role created. View role

IAM > Roles

Roles (4) Info

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

Delete Create role

Search

1

Role name	Trusted entities	Last activity
aws-elasticbeanstalk-ec2-role	AWS Service: ec2	-
aws-elasticbeanstalk-service-role	AWS Service: elasticbeanstalk	-
AWSServiceRoleForSupport	AWS Service: support (Service-Linke	-
AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service	-

8) Select default settings as shown below in the next section.

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

Set up networking, database, and tags - *optional* Info

Virtual Private Cloud (VPC)

VPC
Launch your environment in a custom VPC instead of the default VPC. You can create a VPC and subnets in the VPC management console. Learn more

vpc-0f5e8abf0225b8a45 | (172.31.0.0/16)

Create custom VPC

Instance settings

Choose a subnet in each AZ for the instances that run your application. To avoid exposing your instances to the Internet, run your instances in private subnets and load balancer in public subnets. To run your load balancer and instances in the same public subnets, assign public IP addresses to the instances. Learn more

Public IP address
Assign a public IP address to the Amazon EC2 instances in your environment.
☐ Activated

Instance subnets

Filter instance subnets

	Availability Zone	Subnet	CIDR	Name
☑	us-east-1a	subnet-01c3ce9cb...	172.31.32.0/20	

scaling

Step 5 - *optional*
Configure updates, monitoring, and logging

Step 6
Review

Instance settings

Choose a subnet in each AZ for the instances that run your application. To avoid exposing your instances to the Internet, run your instances in private subnets and load balancer in public subnets. To run your load balancer and instances in the same public subnets, assign public IP addresses to the instances. Learn more

Public IP address
Assign a public IP address to the Amazon EC2 instances in your environment.
☐ Activated

Instance subnets

Filter instance subnets

	Availability Zone	Subnet	CIDR	Name
☑	us-east-1a	subnet-01c3ce9cb...	172.31.32.0/20	
☑	us-east-1f	subnet-055b0c11b...	172.31.64.0/20	
☐	us-east-1e	subnet-05c5390d8...	172.31.48.0/20	
☐	us-east-1c	subnet-0733979c0...	172.31.80.0/20	
☐	us-east-1b	subnet-0a7cc4118...	172.31.0.0/20	
☐	us-east-1d	subnet-0fd9b1dd1...	172.31.16.0/20	

Database Info

Integrate an RDS SQL database with your environment. Learn more

Database Info

Integrate an RDS SQL database with your environment. Learn more

Database subnets

If your Elastic Beanstalk environment is attached to an Amazon RDS, choose subnets for your database instances. Learn more

Choose database subnets (6)

Filter database subnets

Availability Zone	Subnet	CIDR	Name
us-east-1a	subnet-01c3ce9cb...	172.31.32.0/20	
us-east-1f	subnet-055b0c11b...	172.31.64.0/20	
us-east-1e	subnet-05c5390d8...	172.31.48.0/20	
us-east-1c	subnet-0733979c0...	172.31.80.0/20	
us-east-1b	subnet-0a7cc4118...	172.31.0.0/20	
us-east-1d	subnet-0fd9b1dd1...	172.31.16.0/20	

Enable database

Restore a snapshot - *optional*

Restore an existing snapshot from a previously used database.

Snapshot

None

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 instance traffic and scaling - *optional* Info

Instances Info

Configure the Amazon EC2 instances that run your application.

Root volume (boot device)

Root volume type

(Container default)

Size

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

8 GB

IOPS

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

100 IOPS

Throughput

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

125 MiB/s

Amazon CloudWatch monitoring

The time interval between when metrics are reported from the EC2 instances

Monitoring interval

5 minute

Amazon CloudWatch monitoring

The time interval between when metrics are reported from the EC2 instances

Monitoring interval

5 minute

Instance metadata service (IMDS)

Your environment's platform supports both IMDSv1 and IMDSv2. To enforce IMDSv2, deactivate IMDSv1. Learn more

IMDSv1

With the current setting, the environment enables only IMDSv2.

☑ Deactivated

EC2 security groups

Select security groups to control traffic.

EC2 security groups (4)

Filter security groups

	Group name	Group ID	Name
☑	default	sg-009f9ea9079c28fd9	
☐	launch-wizard-1	sg-094cfd5e04d89a25f	
☐	launch-wizard-2	sg-0c0fbdd01d426cf3f	
☐	launch-wizard-3	sg-04536e9fd071d79ea	

Capacity Info

Configure the compute capacity of your environment and auto scaling settings to optimize the number of instances used.

Auto scaling group

Environment type

Select a single-instance or load-balanced environment. You can develop and test an application in a single-instance environment to save costs and then upgrade to a load-balanced environment when the application is ready for production. Learn more

Single instance

Instances

1 Min

1 Max

Fleet composition

Spot instances are launched at the lowest available price. Learn more

◉ On-Demand instance

○ Spot instance

Maximum spot price

The maximum price per instance-hour, in USD, that you're willing to pay for a Spot Instance. Setting a custom price limits your chances to fulfill your target capacity using Spot instances.

○ Default

○ Set your maximum price

On-Demand base

The minimum number of On-Demand Instances that your Auto Scaling group provisions before considering Spot Instances as your environment scales out.

0

environment scales out.

0

On-Demand above base

The percentage of On-Demand Instances as part of any additional capacity that your Auto Scaling group provisions beyond the On-Demand base instances.

0 %

Capacity rebalancing

Specifies whether to enable the capacity rebalancing feature for Spot Instances in your Auto Scaling Group. This option is only relevant when EnableSpot is true in the aws:ec2:instances namespace, and there is at least one Spot Instance in your Auto Scaling group.

Turn on capacity rebalancing

Architecture

The processor architecture determines the instance types that are made available. You can't change this selection after you create the environment. Learn more

- x86_64
  This architecture uses x86 processors and is compatible with most third-party tools and libraries.
- arm64 - *new*
  This architecture uses AWS Graviton2 processors. You might have to recompile some third-party tools and libraries.

Instance types

Add instance types for your fleet. Change the order that the instances are in to set the preferred launch order. This only affects On-Demand instances. We recommend you include at least two instance types. Learn more

Choose x86 instance types

t3.micro ✕ t3.small ✕

AMI ID

Elastic Beanstalk selects a default Amazon Machine Image (AMI) for your environment based on the Region, platform version, and processor architecture that you choose. Learn more

ami-05218d60cd8f94600

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 updates, monitoring, and logging - *optional* Info

Monitoring Info

Health reporting

Enhanced health reporting provides free real-time application and operating system monitoring of the instances and other resources in your environment. The **EnvironmentHealth** custom metric is provided free with enhanced health reporting. Additional charges apply for each custom metric. For more information, see Amazon CloudWatch Pricing

System

- Basic
- Enhanced

Health event streaming to CloudWatch Logs

Configure Elastic Beanstalk to stream environment health events to CloudWatch Logs. You can set the retention up to a maximum of ten years and configure Elastic Beanstalk to delete the logs when you terminate your environment.

Log streaming

Activated (standard CloudWatch charges apply.)

Retention

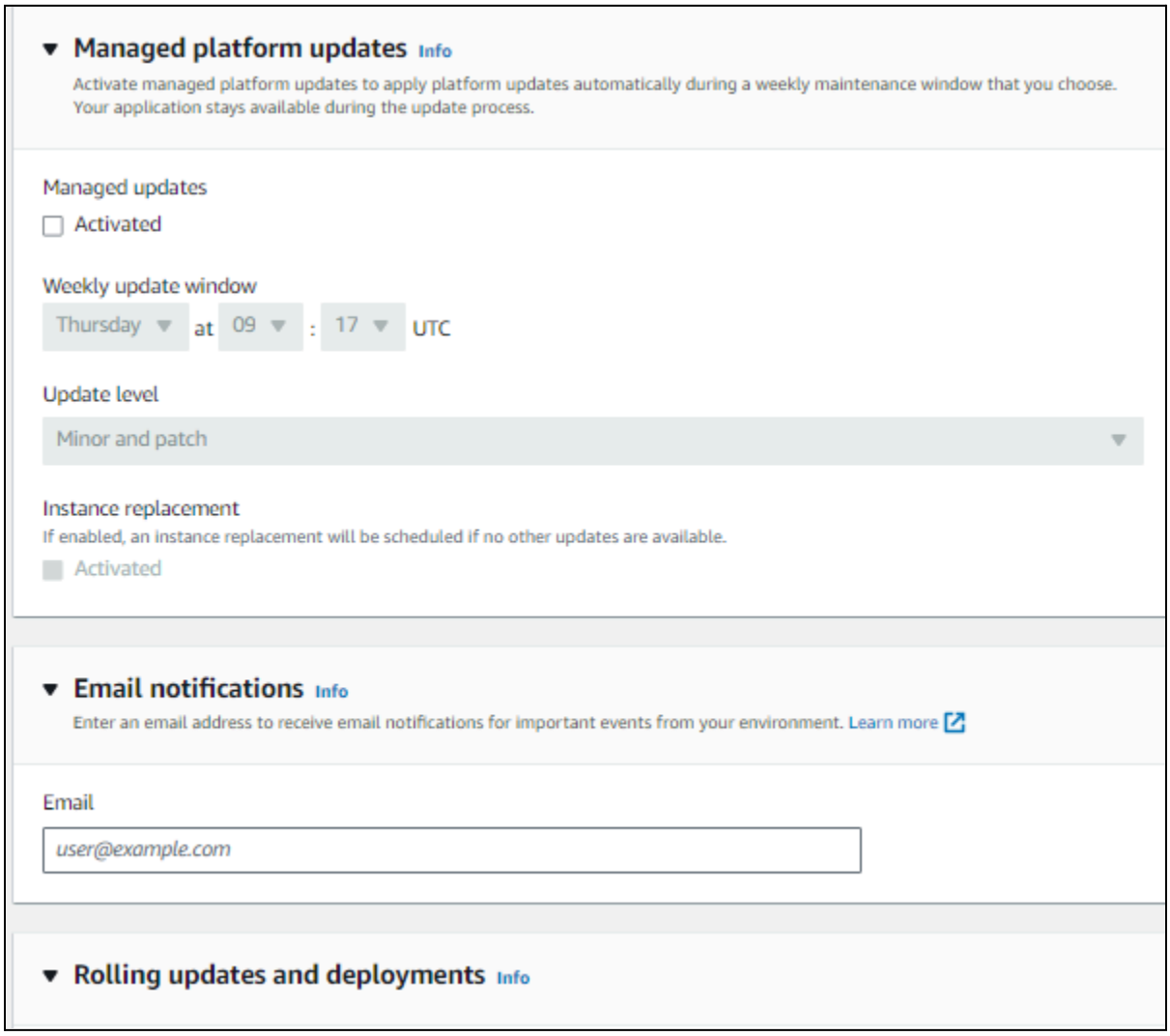
7

Lifecycle

Keep logs after terminating environment

Managed platform updates Info

9) Uncheck Managed Updates, Activated.



Rolling updates and deployments Info

Application deployments

Choose how Amazon Elastic Beanstalk propogates source code changes and software configuration updates. Learn more

Deployment policy

All at once

Batch size type

- (•) Percentage
- ( ) Fixed

Deployment batch size

100

% instances at a time

Configuration updates

Changes to virtual machine settings and VPC configuration trigger rolling updates to replace the instances in your environment without downtime. Learn more

Rolling update type

Deactivated

Deployment preferences

Customize health check requirements and deployment timeouts.

Ignore health check

Don't fail deployments due to health check failures.

Platform software Info

Configure the options available to your specific platform. These include the proxy server and OS environment properties. Learn more

Container options

Proxy server

Nginx

Amazon X-Ray

Amazon X-Ray is a service that collects data about the requests and responses that your application serves and receives. You can use the tools that X-Ray offers to view and filter the data that it provides to identify potential issues and optimization opportunities.

X-Ray daemon

(service charges may apply.)

☐ Activated

S3 log storage

Configure the instances in your environment to upload rotated logs to Amazon S3. Learn more

Rotate logs

(standard S3 charges apply.)

☐ Activated

Instance log streaming to CloudWatch logs

Configure the instances in your environment to stream logs to CloudWatch logs. You can set the retention to up to 10 years and configure Elastic Beanstalk to delete the logs when you terminate your environment. Learn more

Log streaming

(standard CloudWatch charges apply.)

Configure the instances in your environment to upload rotated logs to Amazon S3. Learn more

Rotate logs
(standard S3 charges apply.)
☐ Activated

Instance log streaming to CloudWatch logs

Configure the instances in your environment to stream logs to CloudWatch logs. You can set the retention to up to 10 years and configure Elastic Beanstalk to delete the logs when you terminate your environment. Learn more

Log streaming
(standard CloudWatch charges apply.)
☐ Activated

Retention
7

Lifecycle
Keep logs after terminating envir...

Environment properties

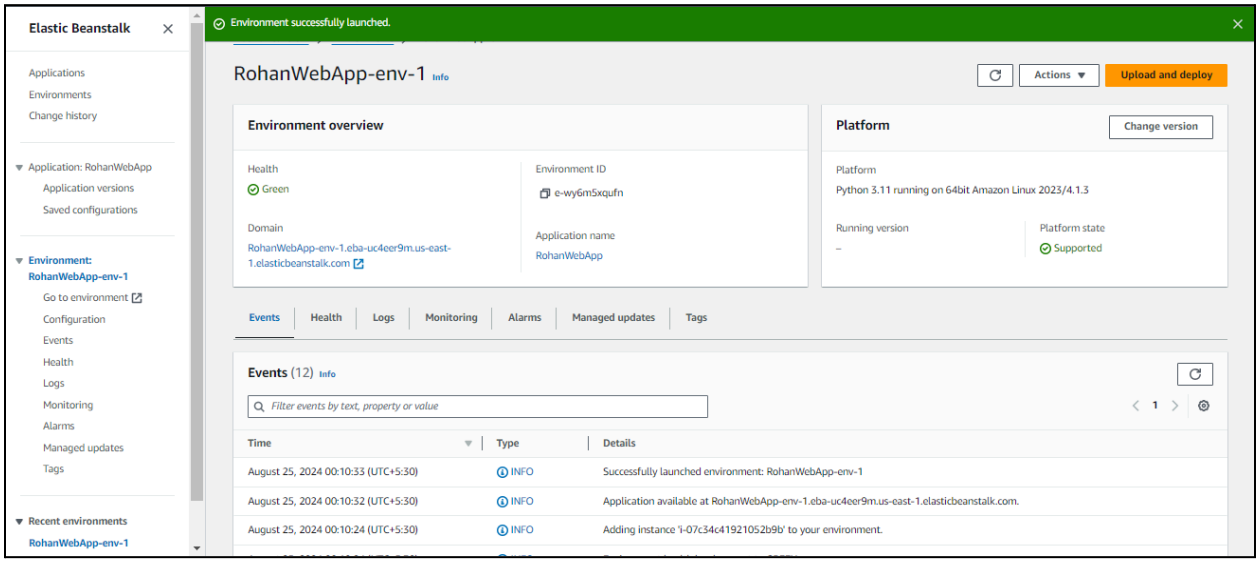
The following properties are passed in the application as environment properties. Learn more

Name	Value	
PYTHONPATH	/var/app/venv/staging-LQM1lest/bin	Remove

Add environment property

Cancel | Previous | Next

10) Environment is successfully launched.



Not secure rohanwebapp-env-1.eba-uc4eer9m.us-east-1.elasticbeanstalk.com

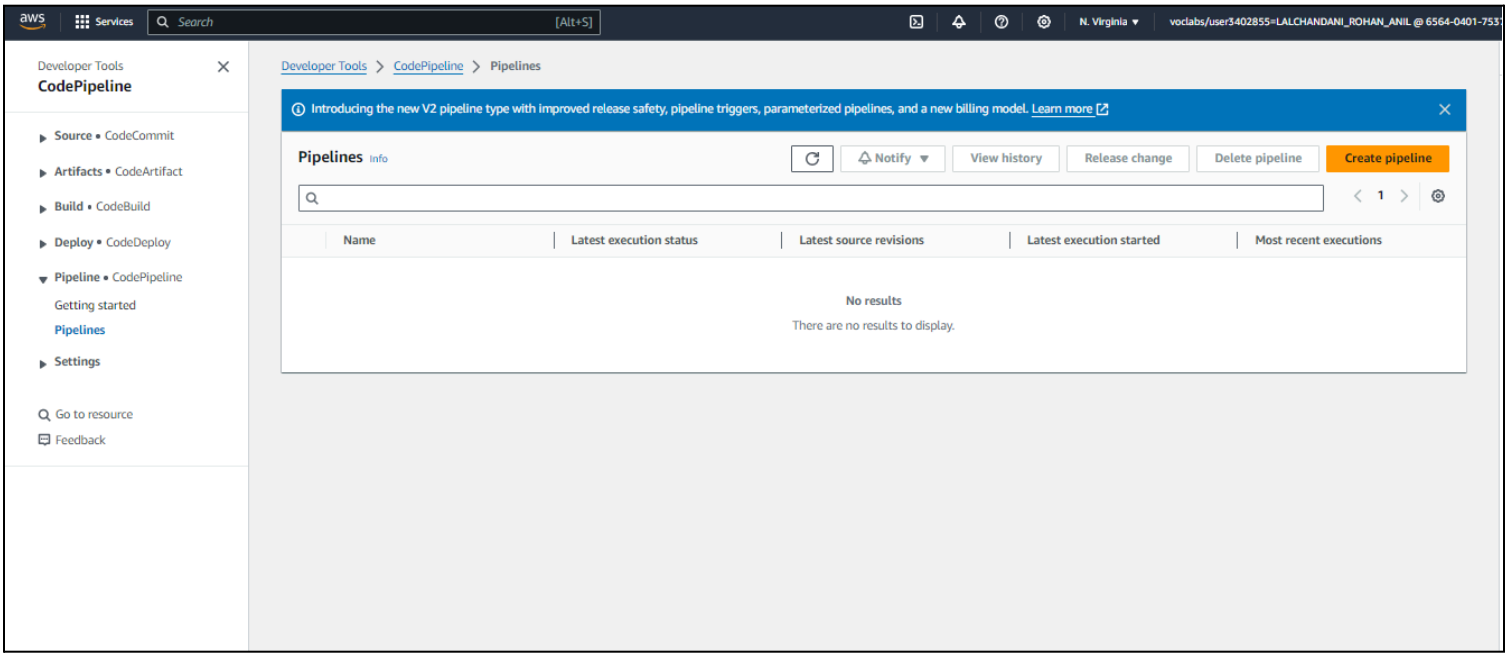
Congratulations

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

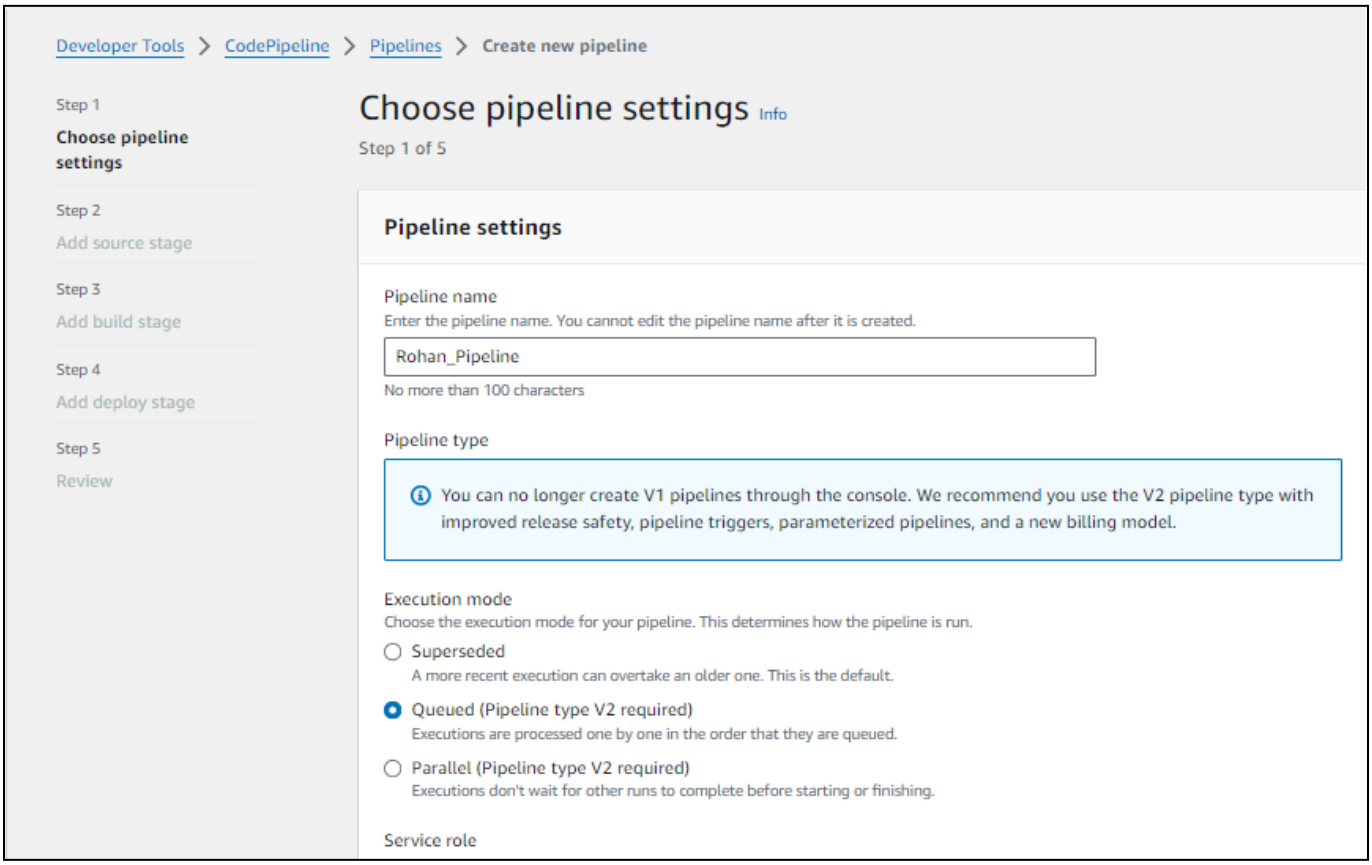
This environment is launched with Elastic Beanstalk Python Platform

Code Deploy and Code Pipeline:

1) Search for Codepipeline in the search box and click on create pipeline.



2) Configuring the pipeline.



aws Services Search [Alt+S]

Service role

New service role
Create a service role in your account

Existing service role
Choose an existing service role from your account

Role name

AWSCodePipelineServiceRole-us-east-1-Rohan_Pipeline

Type your service role name

Allow AWS CodePipeline to create a service role so it can be used with this new pipeline

Variables

You can add variables at the pipeline level. You can choose to assign the value when you start the pipeline. Choosing this option requires pipeline type V2. Learn more

No variables defined at the pipeline level in this pipeline.

Add variable

You can add up to 50 variables.

The first pipeline execution will fail if variables have no default values.

Advanced settings

Developer Tools > CodePipeline > Pipelines > Create new pipeline

Step 1
Choose pipeline settings

Step 2
Add source stage

Step 3
Add build stage

Step 4
Add deploy stage

Step 5
Review

Add source stage Info

Step 2 of 5

Source

Source provider

This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.

GitHub (Version 1)

Grant AWS CodePipeline access to your GitHub repository. This allows AWS CodePipeline to upload commits from GitHub to your pipeline.

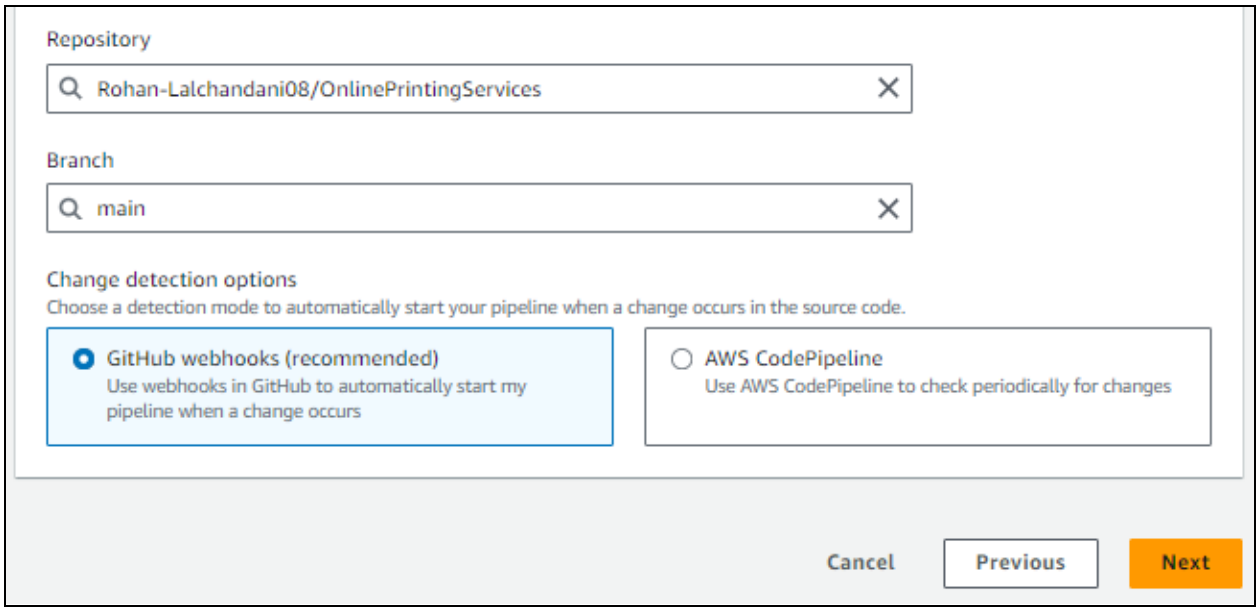
Connected

You have successfully configured the action with the provider.

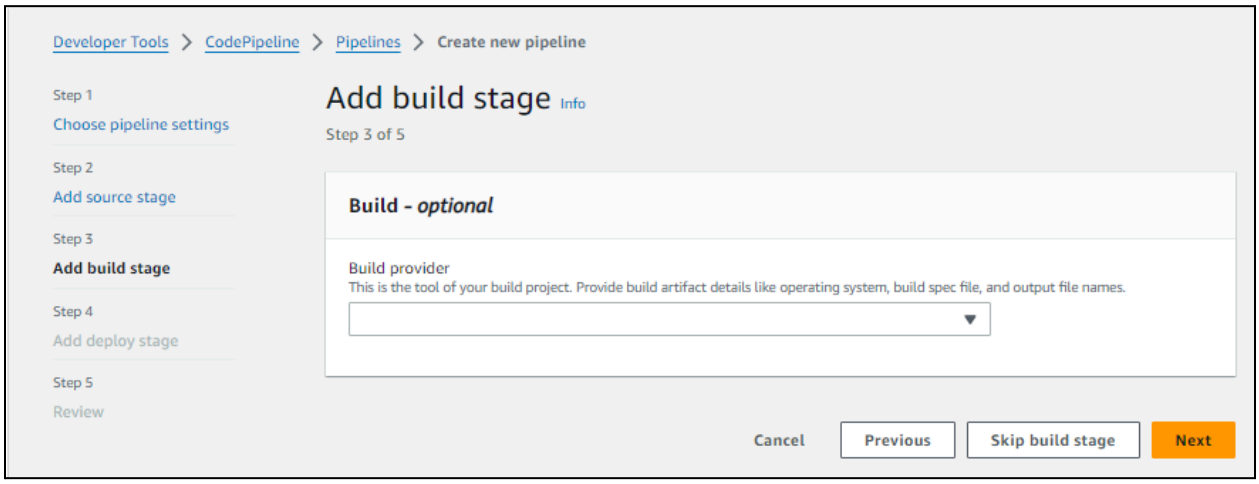
The GitHub (Version 1) action is not recommended

The selected action uses OAuth apps to access your GitHub repository. This is no longer the recommended method. Instead, choose the GitHub (Version 2) action to access your repository by creating a connection. Connections use GitHub Apps to manage authentication and can be shared with other resources. Learn more

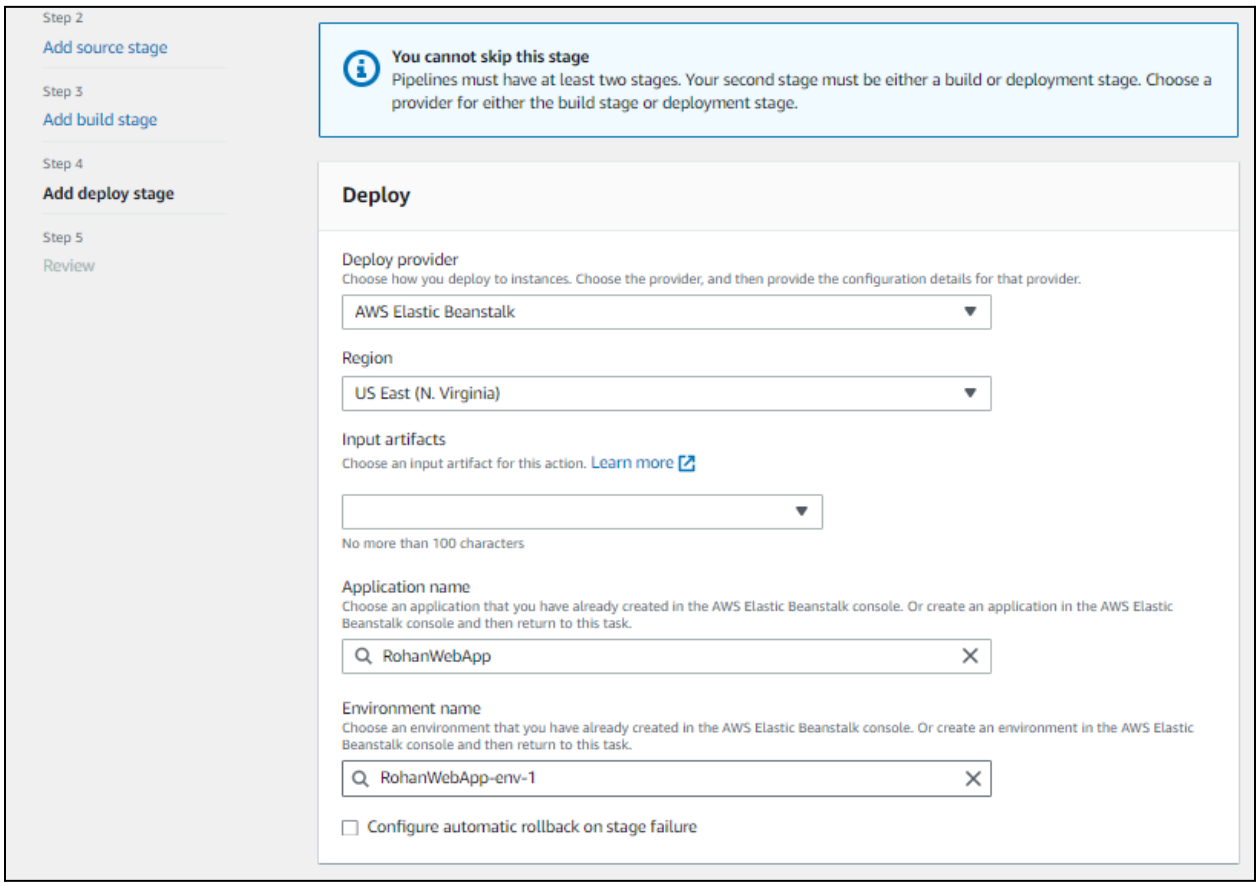
Repository



3) Skip the build stage.



4) Choose Elastic Beanstalk as the Deploy Provider.



Developer Tools > CodePipeline > Pipelines > Create new pipeline

Step 1
Choose pipeline settings

Step 2
Add source stage

Step 3
Add build stage

Step 4
Add deploy stage

Step 5
Review

Review Info

Step 5 of 5

Step 1: Choose pipeline settings

Pipeline settings

Pipeline name
Rohan_Pipeline

Pipeline type
V2

Execution mode
QUEUED

Artifact location
A new Amazon S3 bucket will be created as the default artifact store for your pipeline

Service role name
AWSCodePipelineServiceRole-us-east-1-Rohan_Pipeline

Variables

Name	Default value	Description
No variables		

Step 2: Add source stage

Source action provider

Source action provider
GitHub (Version 1)

PollForSourceChanges
false

Repo
OnlinePrintingServices

Owner
Rohan-Lalchandani08

Branch
main

Step 3: Add build stage

Build action provider

Build stage
No build

Step 4: Add deploy stage

Deploy action provider

Deploy action provider
AWS Elastic Beanstalk

ApplicationName
RohanWebApp

EnvironmentName
RohanWebApp-env-1

Configure automatic rollback on stage failure
Disabled

Cancel | Previous | Create pipeline

5) Pipeline has been created.