

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:

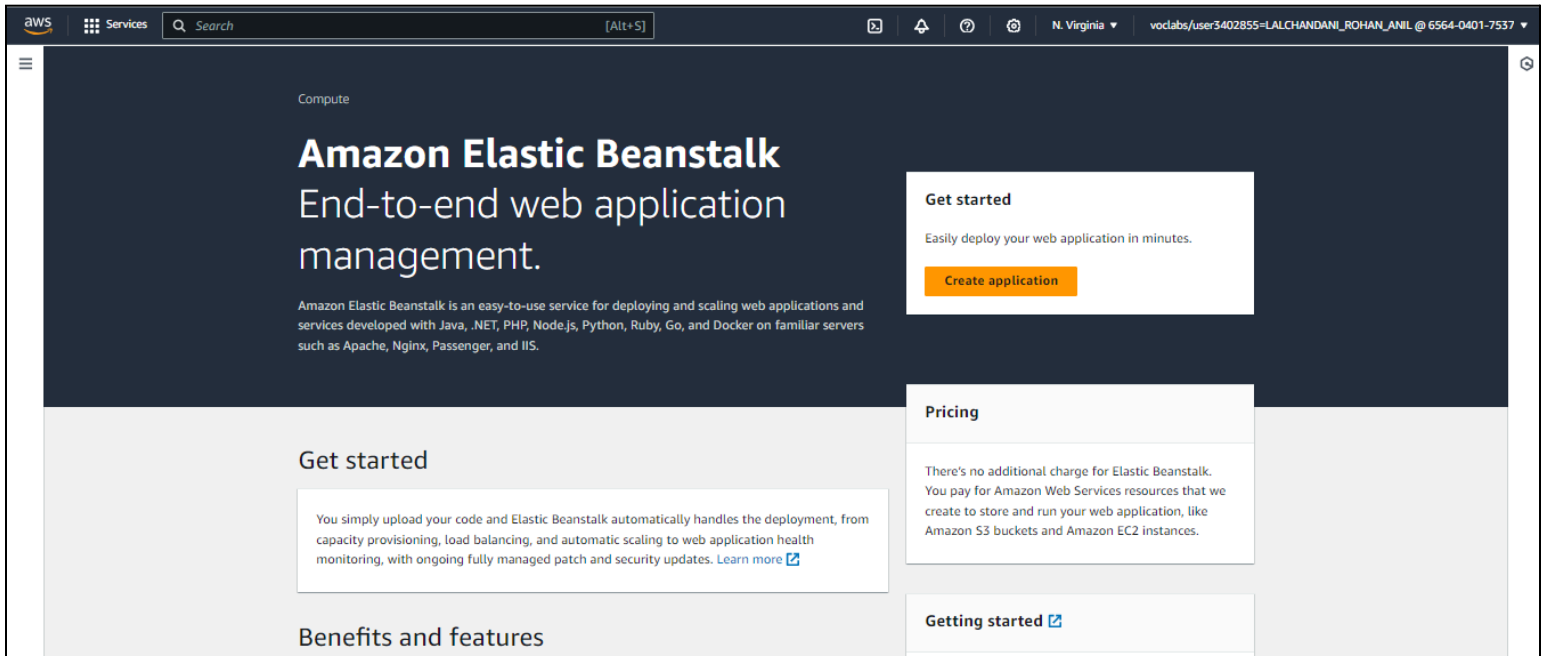
- **Single Instance Environment:** Suitable for development and testing where high availability and fault tolerance are not required.
- **Load Balanced Environment:** Designed for production with auto-scaling, load balancing, and high availability.

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.

The screenshot shows the 'Configure environment' wizard in the Amazon Elastic Beanstalk console. The wizard is divided into six steps: 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, and Step 6: Review. The current step is Step 1, 'Configure environment'. The main content area is titled 'Configure environment' and contains two sections: 'Environment tier' and 'Application information'. The 'Environment tier' section has two options: 'Web server environment' (selected) and 'Worker environment'. The 'Application information' section has a text input field for 'Application name' with the value 'RohanWebApp' and a note that the maximum length is 100 characters. Below this is a section for 'Application tags (optional)'. At the bottom, there's a section for 'Environment information' with a note that the name, subdomain, and description cannot be changed later.

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.

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 service access Info

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.

View permission details

EC2 key pair

Select an EC2 key pair to securely log in to your EC2 instances. [Learn more](#)

EC2 instance profile

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

View permission details

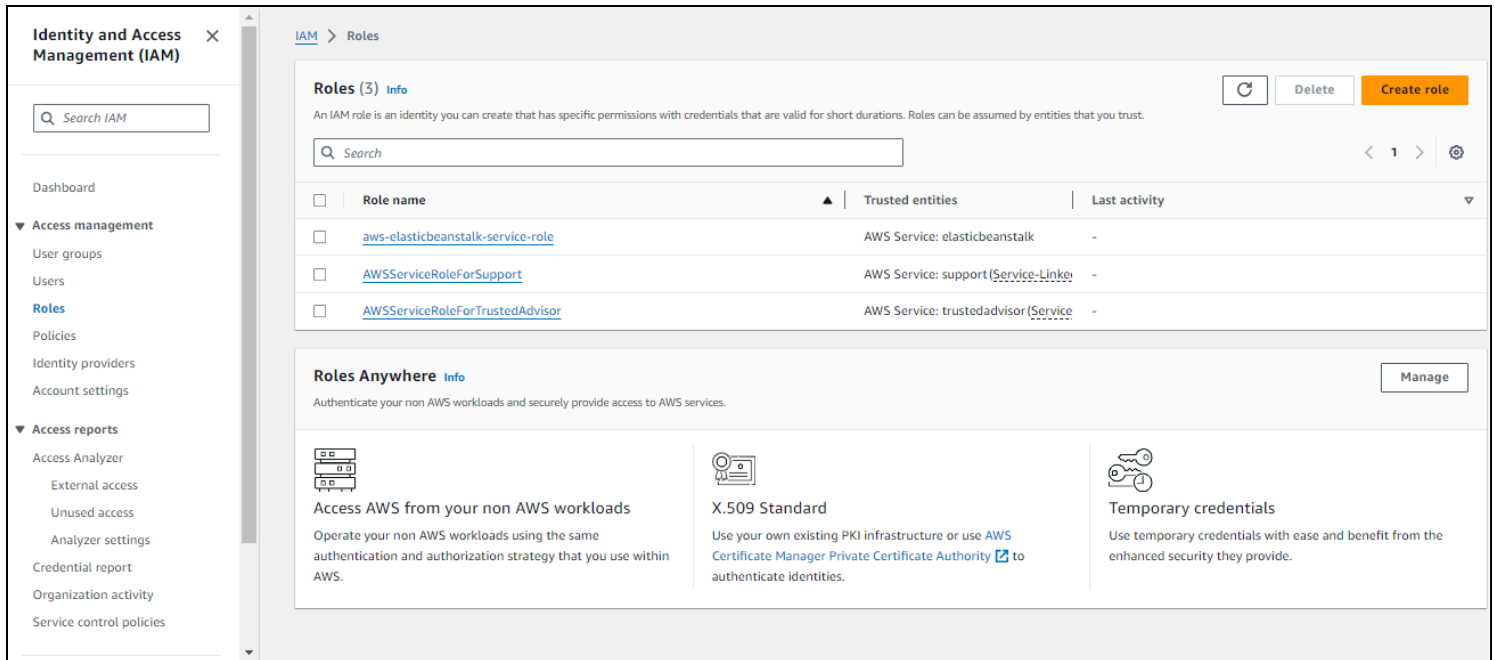
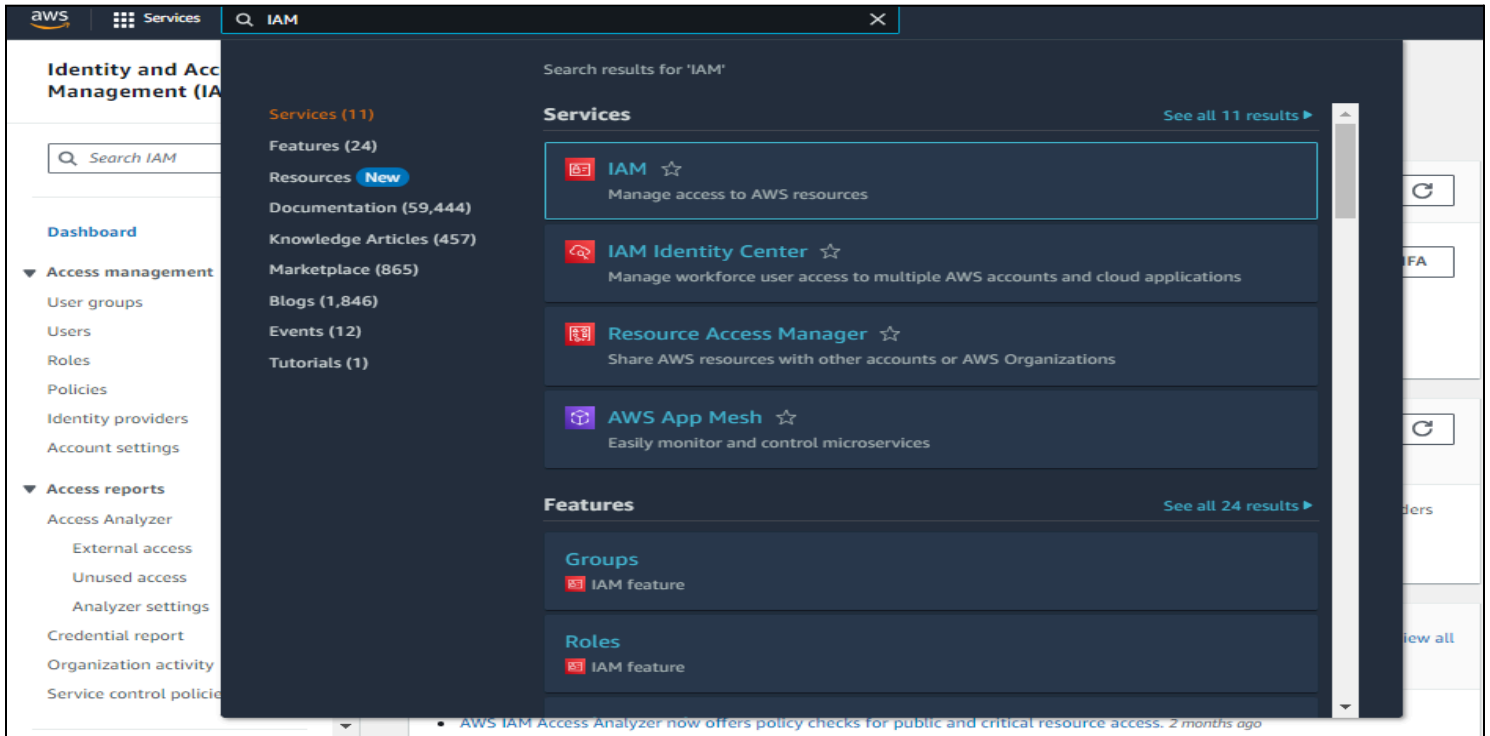
Cancel

Skip to review

Previous

Next

- 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.

[IAM](#) > [Roles](#) > Create role

Step 1
Select trusted entity

Step 2
Add permissions

Step 3
Name, review, and create

Select trusted entity [Info](#)

Trusted entity type

☒ **AWS service**
Allow AWS services like EC2, Lambda, or others to perform actions in this account.

☐ **AWS account**
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.

☐ **Web identity**
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.

☐ **SAML 2.0 federation**
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.

☐ **Custom trust policy**
Create a custom trust policy to enable others to perform actions in this account.

Use case
Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Service or use case

EC2 ▼

Choose a use case for the specified service.
Use case

☒ **EC2**
Allows EC2 instances to call AWS services on your behalf.

Use case
Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Service or use case

EC2 ▼

Choose a use case for the specified service.
Use case

☒ **EC2**
Allows EC2 instances to call AWS services on your behalf.

☐ **EC2 Role for AWS Systems Manager**
Allows EC2 instances to call AWS services like CloudWatch and Systems Manager on your behalf.

☐ **EC2 Spot Fleet Role**
Allows EC2 Spot Fleet to request and terminate Spot Instances on your behalf.

☐ **EC2 - Spot Fleet Auto Scaling**
Allows Auto Scaling to access and update EC2 spot fleets on your behalf.

☐ **EC2 - Spot Fleet Tagging**
Allows EC2 to launch spot instances and attach tags to the launched instances on your behalf.

☐ **EC2 - Spot Instances**
Allows EC2 Spot Instances to launch and manage spot instances on your behalf.

☐ **EC2 - Spot Fleet**
Allows EC2 Spot Fleet to launch and manage spot fleet instances on your behalf.

☐ **EC2 - Scheduled Instances**
Allows EC2 Scheduled Instances to manage instances on your behalf.

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

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](#)

↻

Delete

Create role

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.

🔍 Search

< 1 > ⚙

<input type="checkbox"/>	Role name	Trusted entities	Last activity
<input type="checkbox"/>	aws-elasticbeanstalk-ec2-role	AWS Service: ec2	-
<input type="checkbox"/>	aws-elasticbeanstalk-service-role	AWS Service: elasticbeanstalk	-
<input type="checkbox"/>	AWSServiceRoleForSupport	AWS Service: support (Service-Linker)	-
<input type="checkbox"/>	AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service-Linker)	-

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
<input checked="" type="checkbox"/>	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
<input checked="" type="checkbox"/>	us-east-1a	subnet-01c3ce9cb...	172.31.32.0/20	
<input checked="" type="checkbox"/>	us-east-1f	subnet-055b0c11b...	172.31.64.0/20	
<input type="checkbox"/>	us-east-1e	subnet-05c5390d8...	172.31.48.0/20	
<input type="checkbox"/>	us-east-1c	subnet-0733979c0...	172.31.80.0/20	
<input type="checkbox"/>	us-east-1b	subnet-0a7cc4118...	172.31.0.0/20	
<input type="checkbox"/>	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)

<input type="checkbox"/>	Availability Zone	Subnet ▲	CIDR	Name
<input type="checkbox"/>	us-east-1a	subnet-01c3ce9cb...	172.31.32.0/20	
<input type="checkbox"/>	us-east-1f	subnet-055b0c11b...	172.31.64.0/20	
<input type="checkbox"/>	us-east-1e	subnet-05c5390d8...	172.31.48.0/20	
<input type="checkbox"/>	us-east-1c	subnet-0733979c0...	172.31.80.0/20	
<input type="checkbox"/>	us-east-1b	subnet-0a7cc4118...	172.31.0.0/20	
<input type="checkbox"/>	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

<input checked="" type="checkbox"/>	Group name	Group ID	Name
<input checked="" type="checkbox"/>	default	sg-009f9ea9079c28fd9	
<input type="checkbox"/>	launch-wizard-1	sg-094cfd5e04d89a25f	
<input type="checkbox"/>	launch-wizard-2	sg-0c0fbdd01d426cf3f	
<input type="checkbox"/>	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

The minimum number of On-Demand instances that your Auto Scaling group provisions before considering Spot instances as your 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.

▼ Managed platform updates [Info](#)

Activate managed platform updates to apply platform updates automatically during a weekly maintenance window that you choose. Your application stays available during the update process.

Managed updates

☐ Activated

Weekly update window

Thursday ▼ at 09 ▼ : 17 ▼ UTC

Update level

Minor and patch ▼

Instance replacement

If enabled, an instance replacement will be scheduled if no other updates are available.

☐ Activated

▼ Email notifications [Info](#)


Enter an email address to receive email notifications for important events from your environment. [Learn more](#) [🔗](#)

Email

▼ Rolling updates and deployments [Info](#)

▼ Rolling updates and deployments [Info](#)

Application deployments

Choose how Amazon Elastic Beanstalk propagates 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-LQM1test/bin	Remove

Add environment property

Cancel

Previous

Next

10) Environment is successfully launched.

Elastic Beanstalk

Applications

Environments

Change history

Application: RohanWebApp

Application versions

Saved configurations

Environment: RohanWebApp-env-1

Go to environment

Configuration

Events

Health

Logs

Monitoring

Alarms

Managed updates

Tags

Recent environments

RohanWebApp-env-1

Environment successfully launched.

RohanWebApp-env-1

Environment overview

Health Green

Environment ID e-wy6n5xqufn

Domain RohanWebApp-env-1.eba-uc4eer9m.us-east-1.elasticbeanstalk.com

Application name RohanWebApp

Platform Python 3.11 running on 64bit Amazon Linux 2023/4.1.3

Running version -

Platform state Supported

Events (12)

Time	Type	Details
August 25, 2024 00:10:33 (UTC+5:30)	INFO	Successfully launched environment: RohanWebApp-env-1
August 25, 2024 00:10:32 (UTC+5:30)	INFO	Application available at RohanWebApp-env-1.eba-uc4eer9m.us-east-1.elasticbeanstalk.com.
August 25, 2024 00:10:24 (UTC+5:30)	INFO	Adding instance 'i-07c34c41921052b9b' to your environment.



Not secure

rohanwebapp-env-1.eba-uc4eer9m.us-east-1.elasticbeanstalk.com

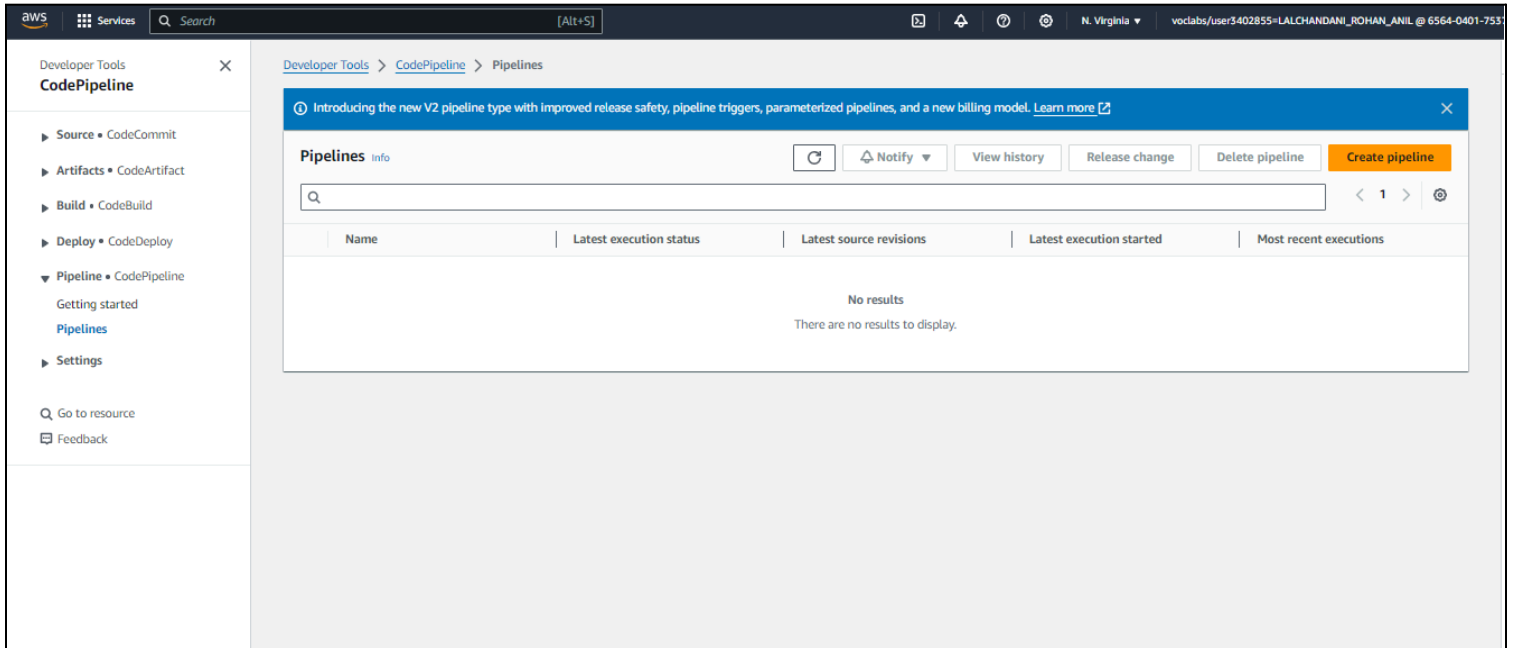
Congratulations

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

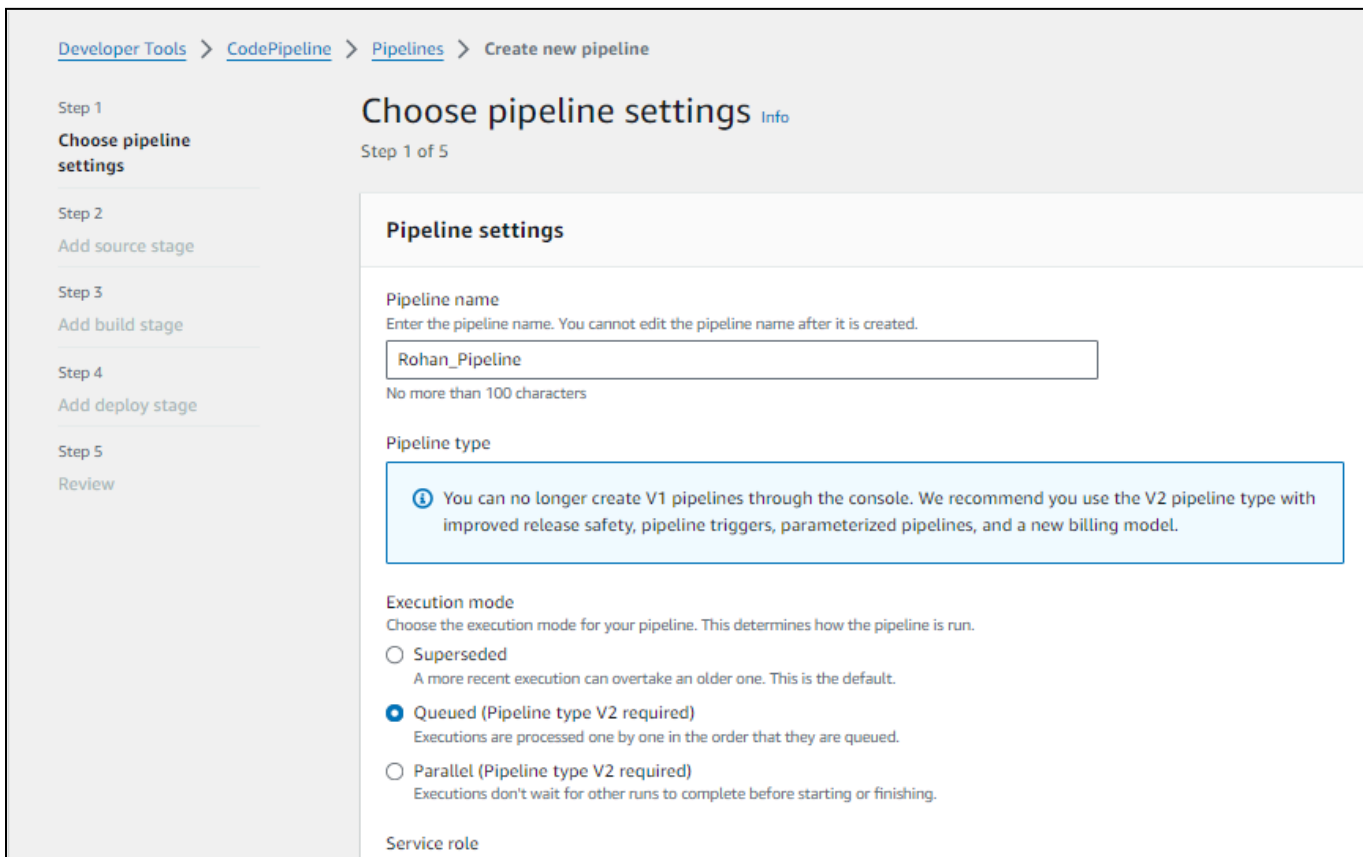
This environment is launched with Elastic Beanstalk Python Platform

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

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

Repository

Q Rohan-Lalchandani08/OnlinePrintingServices

X

Branch

Q main

X

Change detection options

Choose a detection mode to automatically start your pipeline when a change occurs in the source code.

☒ GitHub webhooks (recommended)

Use webhooks in GitHub to automatically start my pipeline when a change occurs

☐ AWS CodePipeline

Use AWS CodePipeline to check periodically for changes

Cancel

Previous

Next

3) Skip the build stage.

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 build stage

Info

Step 3 of 5

Build - optional

Build provider

This is the tool of your build project. Provide build artifact details like operating system, build spec file, and output file names.

Cancel

Previous

Skip build stage

Next

4) Choose Elastic Beanstalk as the Deploy Provider.

Step 2

Add source stage

Step 3

Add build stage

Step 4

Add deploy stage

Step 5

Review

You cannot skip this stage

Pipelines must have at least two stages. Your second stage must be either a build or deployment stage. Choose a provider for either the build stage or deployment stage.

Deploy

Deploy provider

Choose how you deploy to instances. Choose the provider, and then provide the configuration details for that provider.

AWS Elastic Beanstalk

Region

US East (N. Virginia)

Input artifacts

Choose an input artifact for this action. [Learn more](#)

No more than 100 characters

Application name

Choose an application that you have already created in the AWS Elastic Beanstalk console. Or create an application in the AWS Elastic Beanstalk console and then return to this task.

Q RohanWebApp

X

Environment name

Choose an environment that you have already created in the AWS Elastic Beanstalk console. Or create an environment in the AWS Elastic Beanstalk console and then return to this task.

Q RohanWebApp-env-1

X

☐

Configure automatic rollback on stage failure

- 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.

The screenshot shows the AWS CodePipeline console. A green banner at the top indicates 'Success: Congratulations! The pipeline Rohan_Pipeline has been created.' The left sidebar shows the 'CodePipeline' service selected. The main area displays the 'Rohan_Pipeline' details, including its type (V2) and execution mode (QUEUED). The pipeline stages are visible: 'Source' (Succeeded) and 'Deploy' (Succeeded). The 'Source' stage details show it used 'GitHub (Version 1)' and succeeded 3 minutes ago. The 'Deploy' stage also succeeded. A 'Release change' button is highlighted in orange. The bottom of the console shows the 'Disable transition' button and the 'Start rollback' button.

The screenshot shows a web application for 'PRINTING COMPANY'. The header features the company logo and the tagline 'Print It - Online Printing Services for everyone'. Below the header, there are links for 'About Us' and 'Contact Us'. A list of links includes 'Introduction Audio', 'Promotional Video', 'Company Details', and 'Services We Offer'. The main content area features a video player for 'Introduction Audio' and a 'Promotional Video' section. The video player shows a timestamp of 0:00 / 0:01. The promotional video section shows a video thumbnail with the text 'Successful business meeting of people discussing stock...' and buttons for 'Watch later' and 'Share'.