

EXPERIMENT NO:2

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:

AWS Elastic Beanstalk

AWS Elastic Beanstalk is a versatile Platform as a Service (PaaS) offering from Amazon Web Services (AWS) that simplifies the deployment and management of applications in the cloud. It abstracts much of the underlying infrastructure complexity, allowing developers to focus on building and maintaining their applications while AWS takes care of provisioning resources like EC2 instances, load balancers, and databases.

Key Features of Elastic Beanstalk:

1. **Simplified Application Deployment:**
 - Elastic Beanstalk streamlines the deployment process by automatically handling the setup, configuration, and maintenance of the infrastructure required to run your application. This allows developers to concentrate on writing and refining their code rather than managing servers.
2. **Broad Language and Framework Support:**
 - Elastic Beanstalk supports multiple programming languages and frameworks, such as Java, .NET, Node.js, Python, Ruby, PHP, Go, and Docker. This flexibility makes it easy for developers to deploy applications built with their preferred technologies.
3. **Automatic Resource Scaling:**
 - Elastic Beanstalk includes automatic scaling capabilities, dynamically adjusting the number of instances running your application based on current demand. This ensures that your application remains responsive and cost-efficient during traffic fluctuations.
4. **Robust Health Monitoring:**
 - The platform continuously monitors the health of your application and provides detailed metrics and logs. It can automatically replace any unhealthy instances, ensuring high availability and reliability of your application.
5. **Multiple Environment Support:**
 - Elastic Beanstalk allows you to manage different environments, such as development, staging, and production. You can deploy updates to one environment without impacting others, providing a controlled and organized deployment process.

AWS CodeBuild

AWS CodeBuild is a fully managed continuous integration (CI) service that compiles your source code, runs tests, and produces deployable artifacts. It automates the build process, ensuring consistent compilation and testing across various environments.

Key Features of AWS CodeBuild:

1. **Managed Build Infrastructure:**

- With AWS CodeBuild, there's no need to manage your own build servers. AWS handles all infrastructure management, allowing you to focus on developing and testing your applications.
- 2. **Automatic Scaling:**
 - CodeBuild automatically scales to meet your needs, handling multiple builds simultaneously. This ensures that even during peak times, your builds are processed quickly and efficiently.
- 3. **Customizable Build Environments:**
 - You can define custom build environments using Docker images, providing the flexibility to tailor the build environment to meet the specific needs of your application.
- 4. **Seamless AWS Integration:**
 - CodeBuild integrates effortlessly with other AWS services, such as CodePipeline, CodeCommit, and S3, enabling you to create a complete CI/CD pipeline that automates your entire build and deployment process.
- 5. **Cost-Effective Pricing:**
 - CodeBuild uses a pay-as-you-go pricing model, charging you only for the build time you consume. This makes it an economical solution for automating your build processes.

Deploying on S3 Using AWS CodePipeline

AWS CodePipeline is a fully managed continuous delivery service that automates the steps required to release your software, from building and testing to deployment. By integrating with services like CodeBuild and S3, CodePipeline provides a streamlined process for deploying applications.

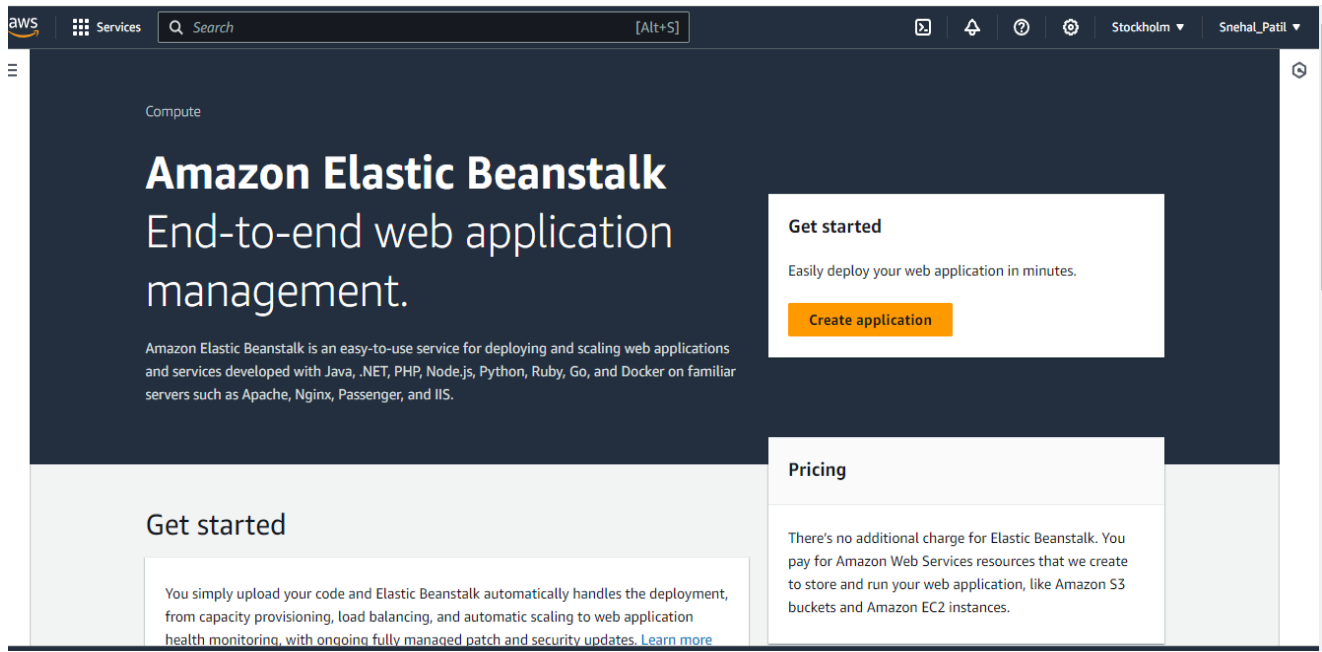
Key Steps to Deploy on S3 Using AWS CodePipeline:

1. **Source Code Retrieval:**
 - The pipeline begins with a source stage, where the source code is retrieved from a repository such as AWS CodeCommit, GitHub, or an S3 bucket. This source code is then used in subsequent stages for building and deploying the application.
2. **Build and Package with CodeBuild:**
 - In the build stage, AWS CodePipeline triggers CodeBuild to compile the source code, run tests, and package the application. The output of this process is a deployable artifact stored in an S3 bucket.
3. **Deployment to S3:**
 - In the deployment stage, CodePipeline automatically deploys the artifacts generated during the build stage to an S3 bucket. The S3 bucket can be configured to host a static website or store files accessed by your application.
4. **Automation and Notifications:**
 - CodePipeline can be configured to automatically trigger builds and deployments when changes are made to the source code repository. Additionally, it can send notifications via Amazon SNS to keep you informed about the status of your pipeline.
5. **Pipeline Monitoring and Logging:**
 - CodePipeline provides tools for monitoring and logging the progress of each stage in your pipeline. This allows you to quickly identify and address any issues that arise during the build or deployment process, ensuring a smooth release cycle.

Implementation :

Elastic Beanstalk

Step 1: create environment



The screenshot shows the AWS Management Console for Amazon Elastic Beanstalk. The top navigation bar includes the AWS logo, 'Services' link, a search bar, and user information for 'Snehal_Patil' in the 'Stockholm' region. The main content area has a dark blue header with the text 'Compute' and 'Amazon Elastic Beanstalk End-to-end web application management.' Below this, a paragraph describes the service as an easy-to-use platform for deploying and scaling web applications. To the right, a 'Get started' box contains the text 'Easily deploy your web application in minutes.' and a prominent orange 'Create application' button. Further down, a 'Pricing' section states that there are no additional charges for Elastic Beanstalk itself, only for the underlying AWS resources like Amazon S3 and Amazon EC2.

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

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

Must be from 4 to 40 characters in length. The name can contain only letters, numbers, and hyphens. It can't start or end with a hyphen. This name must be unique within a region in your account.

Domain

.eu-north-1.elasticbeanstalk.com

Check availability

Environment description

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

Platform branch

Platform version

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

Cancel

Next

[IAM](#) > Dashboard

IAM Dashboard



Security recommendations 1



- Add MFA for root user**
Add MFA for root user - Enable multi-factor authentication (MFA) for the root user to improve security for this account.
- Root user has no active access keys**
Using access keys attached to an IAM user instead of the root user improves security.

Add MFA

IAM resources

Resources in this AWS Account



User groups	Users	Roles	Policies	Identity providers

AWS Account

Account ID
 825765388229


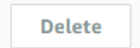
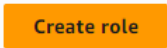
Account Alias
[Create](#)

Sign-in URL for IAM users in this account
 <https://825765388229.signin.aws.amazon.com/console>


Quick Links

[My security credentials](#)

Roles (2) Info

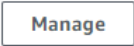
  

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

< 1 > 

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

Roles Anywhere Info



Authenticate your non AWS workloads and securely provide access to AWS services.



Access AWS from your non AWS workloads



X.509 Standard

Use your own existing PKI infrastructure or



Temporary credentials

Use temporary credentials with ease and

Step 1
[Select trusted entity](#)

Step 2
Add permissions

Step 3
Name, review, and create

Add permissions Info


Permissions policies (3/946) Info







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











Filter by Type

All types

14 matches

< 1 > 

<input type="checkbox"/>	Policy name	Type	Description
<input type="checkbox"/>	 AdministratorAccess-...	AWS managed	Grants account administrative permissions...
<input type="checkbox"/>	 AWSElasticBeanstalkC...	AWS managed	Provide the instance in your custom platf...
<input type="checkbox"/>	 AWSElasticBeanstalkE...	AWS managed	AWS Elastic Beanstalk Service policy for H...
<input type="checkbox"/>	 AWSElasticBeanstalk...	AWS managed	This policy is for the AWS Elastic Beanstal...
<input checked="" type="checkbox"/>	 AWSElasticBeanstalk...	AWS managed	Provide the instances in your multicontain...
<input type="checkbox"/>	 AWSElasticBeanstalkR...	AWS managed	Grants read-only permissions. Explicitly all...

<input checked="" type="checkbox"/>		AWS Elastic Beanstalk Role for Amazon EC2	AWS managed	Provide the instances in your multicontain...
<input type="checkbox"/>		AWS Elastic Beanstalk Role for Amazon EC2	AWS managed	Grants read-only permissions. Explicitly all...
<input type="checkbox"/>		AWS Elastic Beanstalk Role for Amazon EC2	AWS managed	AWS Elastic Beanstalk RoleCore (Elastic Bea...
<input type="checkbox"/>		AWS Elastic Beanstalk Role for Amazon EC2	AWS managed	(Elastic Beanstalk operations role) Allows ...
<input type="checkbox"/>		AWS Elastic Beanstalk Role for Amazon EC2	AWS managed	(Elastic Beanstalk operations role) Allows ...
<input type="checkbox"/>		AWS Elastic Beanstalk Role for Amazon EC2	AWS managed	(Elastic Beanstalk operations role) Allows ...
<input type="checkbox"/>		AWS Elastic Beanstalk Role for Amazon EC2	AWS managed	(Elastic Beanstalk operations role) Allows ...
<input type="checkbox"/>		AWS Elastic Beanstalk Role for Amazon EC2	AWS managed	(Elastic Beanstalk operations role) Allows ...
<input checked="" type="checkbox"/>		AWS Elastic Beanstalk Role for Amazon EC2	AWS managed	Provide the instances in your web server e...
<input checked="" type="checkbox"/>		AWS Elastic Beanstalk Role for Amazon EC2	AWS managed	Provide the instances in your worker envir...

► Set permissions boundary - optional

Cancel

Previous

Next

Identity and Access Management (IAM)

Search IAM

Dashboard

▼ Access management

User groups

Users

Roles

Policies

Identity providers

Account settings

▼ Access reports

Access Analyzer

External access

Role aws-Snehal-ec2-role created.

View role

Roles (3) Info

Refresh

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

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

Roles Anywhere Info

Authenticate your non AWS workloads and securely provide access to AWS services.

Manage

Step 2 : add your Ec2 key pair and instance profile

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

Existing service roles

Choose an existing IAM role for Elastic Beanstalk to assume as a service role. The existing IAM role must have the required IAM managed policies.

aws-Snehal-ec2-role

↕

↻

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-Snehal-ec2-role

↕

↻

Service role

☐ Create and use new service role

☒ Use an existing service role

Existing service roles

Choose an existing IAM role for Elastic Beanstalk to assume as a service role. The existing IAM role must have the required IAM managed policies.

aws-Snehal-ec2-role

↕

↻

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-Snehal-ec2-role

↕

↻

[View permission details](#)

Cancel

Skip to review

Previous

Next

Review [Info](#)

Step 1: Configure environment

[Edit](#)

Environment information

Environment tier	Application name
Web server environment	Snehal123
Environment name	Application code
Snehal123-env	Sample application
Platform	
arn:aws:elasticbeanstalk:eu-north-1::platform/Python	
3.11 running on 64bit Amazon Linux 2023/4.1.3	

Step 2: Configure service access

[Edit](#)

Service access [Info](#)

Step 2: Configure service access

[Edit](#)

Service access [Info](#)

Configure the service role and EC2 instance profile that Elastic Beanstalk uses to manage your environment. Choose an EC2 key pair to securely log in to your EC2 instances.

Service role	EC2 instance profile
arn:aws:iam::825765388229:role/aws-Snehal-ec2-role	aws-Snehal-ec2-role

Step 3: Set up networking, database, and tags

[Edit](#)

Networking, database, and tags [Info](#)

Configure VPC settings, and subnets for your environment's EC2 instances and load balancer. Set up an Amazon RDS database that's integrated with your environment.

No options configured

Tags

Step 4: Configure instance traffic and scaling

[Edit](#)

Instance traffic and scaling [Info](#)

Customize the capacity and scaling for your environment's instances. Select security groups to control instance traffic. Configure the software that runs on your environment's instances by setting platform-specific options.

Instances

IMDSv1

Deactivated

Capacity

Environment type	Fleet composition	On-demand base
Single instance	On-Demand instance	0
On-demand above base	Capacity rebalancing	Scaling cooldown
0	Deactivated	360
Processor type	Instance types	AMI ID
x86_64	t3.micro,t3.small	ami-030d0ebd08fe18778

View all settings

View all settings

View all settings

Step 5: Configure updates, monitoring, and logging

[Edit](#)

Updates, monitoring, and logging [Info](#)

Define when and how Elastic Beanstalk deploys changes to your environment. Manage your application's monitoring and logging settings, instances, and other environment resources.

Monitoring

System	Cloudwatch custom metrics - instance	Cloudwatch custom metrics - environment
enhanced	—	—
Log streaming	Retention	Lifecycle
Deactivated	7	false

Updates

Managed updates	Deployment batch size	Deployment batch size type
Activated	100	Percentage
Command timeout	Deployment policy	Health threshold

Platform software

Lifecycle	Log streaming	NumProcesses
false	Deactivated	1
NumThreads	WSGIPath	Proxy server
15	application	nginx
Logs retention	Rotate logs	Update level
7	Deactivated	minor
X-Ray enabled		
Deactivated		

Environment properties

Key	Value
PYTHONPATH	/var/app/venv/staging-LQM1lest/bin

Cancel Previous Submit

Environment successfully launched.

Elastic Beanstalk > Environments > Snehal123-env

Snehal123-env

Actions Upload and deploy

Environment overview

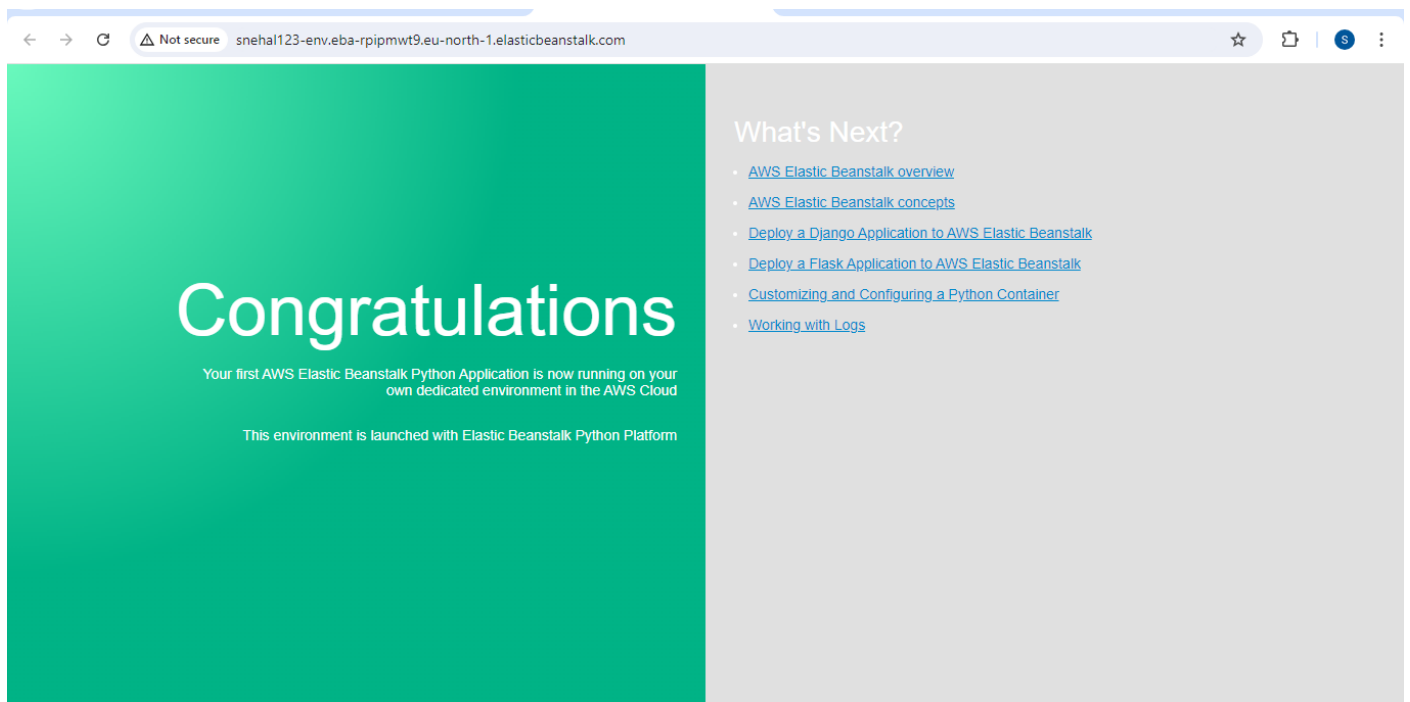
Health	Environment ID
Warning	e-nm5tuux6pa
Domain	Application name
Snehal123-env.eba-rpipmwt9.eu-north-1.elasticbeanstalk.com	Snehal123

Platform Change version

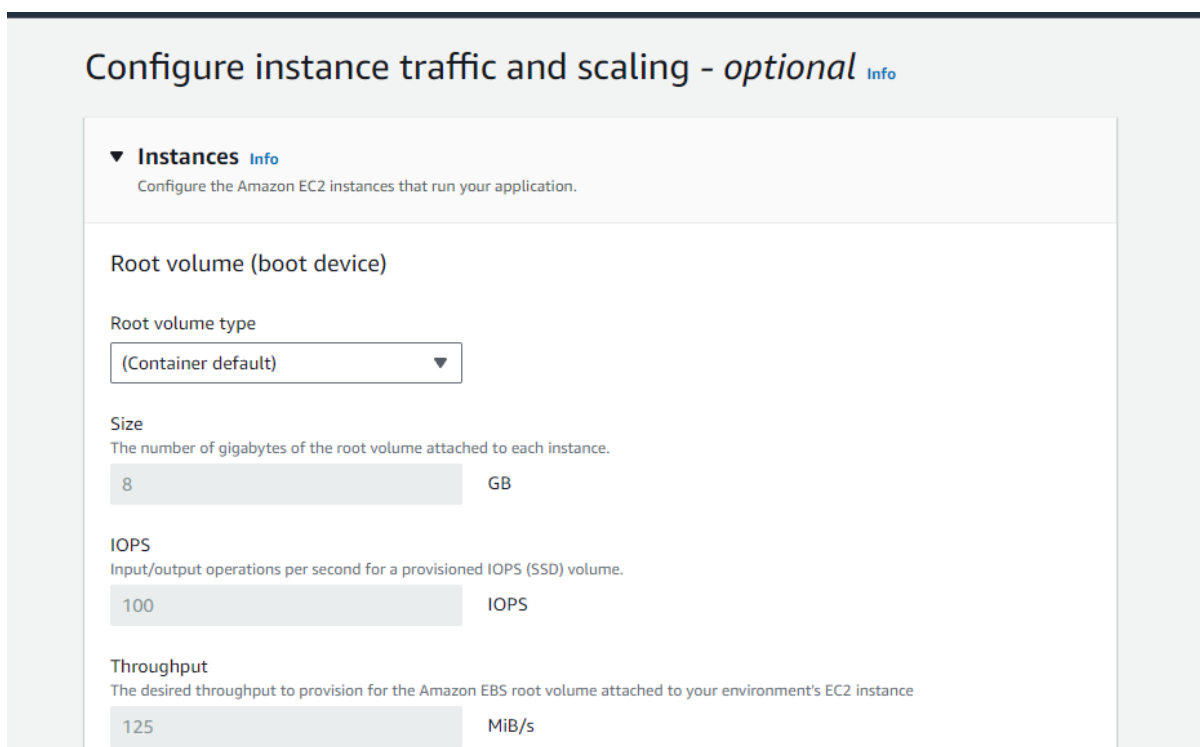
Platform
Python 3.11 running on 64bit Amazon Linux 2023/4.1.3
Running version
-
Platform state
Supported

Events Health Logs Monitoring Alarms Managed updates Tags

Step 3: Beanstalk environment is created



Step 4 : add security config and review all settings



EC2 security groups

Select security groups to control traffic.

EC2 security groups (6)



Filter security groups

<input type="checkbox"/>	Group name ▲	Group ID ▼	Name ▼
<input type="checkbox"/>	default	sg-0a1301dad692be19a	
<input type="checkbox"/>	launch-wizard-1	sg-08f567b852d69d909	
<input type="checkbox"/>	launch-wizard-2	sg-09df1f87b10436fac	
<input type="checkbox"/>	launch-wizard-3	sg-03ee263ce302b1e04	
<input type="checkbox"/>	launch-wizard-4	sg-01d4af8f20286924d	
<input checked="" type="checkbox"/>	launch-wizard-5	sg-0bed7f87bc908ca03	

▼ Capacity [Info](#)

PIPELINE CREATION:

Step 1 : click on create pipeline and give name

> [CodePipeline](#) > [Pipelines](#) > Create new pipeline

Choose pipeline settings [Info](#)

Step 1 of 5

Pipeline settings

Pipeline name

Enter the pipeline name. You cannot edit the pipeline name after it is created.

pipeline_Snehal1

No more than 100 characters

Pipeline type

You can no longer create V1 pipelines through the console. We recommend you use the V2 pipeline type with improved release safety, pipeline triggers, parameterized pipelines, and a new billing model.

Execution mode

Choose the execution mode for your pipeline. This determines how the pipeline is run.

☐ Superseded

A more recent execution can overtake an older one. This is the default.

☒ Queued (Pipeline type V2 required)

Step 2 : Add Your github account and add the file to add to pipeline deployment

[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

 **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

Q Snehal490102/Nykaa-E-commerce_css ✕

Branch

Q main ✕

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

Step 3 : Add deploy config choosing the elastic beanstalk

Step 1

Choose pipeline settings

Step 2

Add source stage

Step 3

Add build stage

Step 4


Add deploy stage

Step 5

Review

Add deploy stage [Info](#)

Step 4 of 5

**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

Europe (Stockholm) ▼

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

Region

Europe (Stockholm) ▼

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

SourceArtifact ▼

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 Snehal123

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 Snehal123-env

X

☐ Configure automatic rollback on stage failure

Cancel

Previous

Next

Step 4 : review changes and submit

> [CodePipeline](#) > [Pipelines](#) > Create new pipeline

Review Info

Settings

Step 5 of 5

Step 1: Choose pipeline settings

Pipeline settings

Pipeline name

pipeline_Snehal1

Pipeline type

V2

Execution mode

QUEUED

Artifact location

codepipeline-eu-north-1-77720346183

Service role name

AWSCodePipelineServiceRole-eu-north-1-pipeline_Snehal1

Step 2: Add source stage

Source action provider

Source action provider

GitHub (Version 1)

PollForSourceChanges

false

Repo

Nykaa-E-commerce_css

Owner

Snehal490102

Branch

main

Step 3: Add build stage

Step 2: Add source stage

Source action provider

Source action provider
GitHub (Version 1)
PollForSourceChanges
false
Repo
Nykaa-E-commerce_css
Owner
Snehal490102
Branch
main

Step 4: Add deploy stage

Deploy action provider

Deploy action provider
AWS Elastic Beanstalk
ApplicationName
Snehal123
EnvironmentName
Snehal123-env
Configure automatic rollback on stage failure
Disabled

Cancel

Previous

Create pipeline

pipeline_Snehal

Notify ▼

Edit

Stop execution

Clone pipeline

Release change

Pipeline type: V2 Execution mode: QUEUED

✔ Source Succeeded

Pipeline execution ID: [bd6f8320-0a79-412d-9825-256bb1ec64d0](#)

Source

[GitHub \(Version 1\)](#) [🔗](#)

✔ Succeeded - [Just now](#)

[b8307259](#) [🔗](#)

View details

[b8307259](#) [🔗](#) Source: Update style.css

Disable transition

⋮ Deploy ⓘ In progress

Disable transition

⋮ Deploy ⓘ In progress

Pipeline execution ID: [bd6f8320-0a79-412d-9825-256bb1ec64d0](#)

Deploy

[AWS Elastic Beanstalk](#) [🔗](#)

⋮ In progress - [Just now](#)

View details

[b8307259](#) [🔗](#) Source: Update style.css

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 Snehal123 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 Snehal123-Env X

☐ Configure automatic rollback on stage failure

Step 6 : Check the deployed website at beanstalk link

