Academic Year: 2022-23 **Semester: V**
Class / Branch: TE IT
Subject: Advanced Devops Lab (ADL)
Subject Lab Incharge: Prof. Manasi Choche

EXPERIMENT NO. 02

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:

Continuous deployment allows you to deploy revisions to a production environment automatically without explicit approval from a developer, making the entire software release process automated.

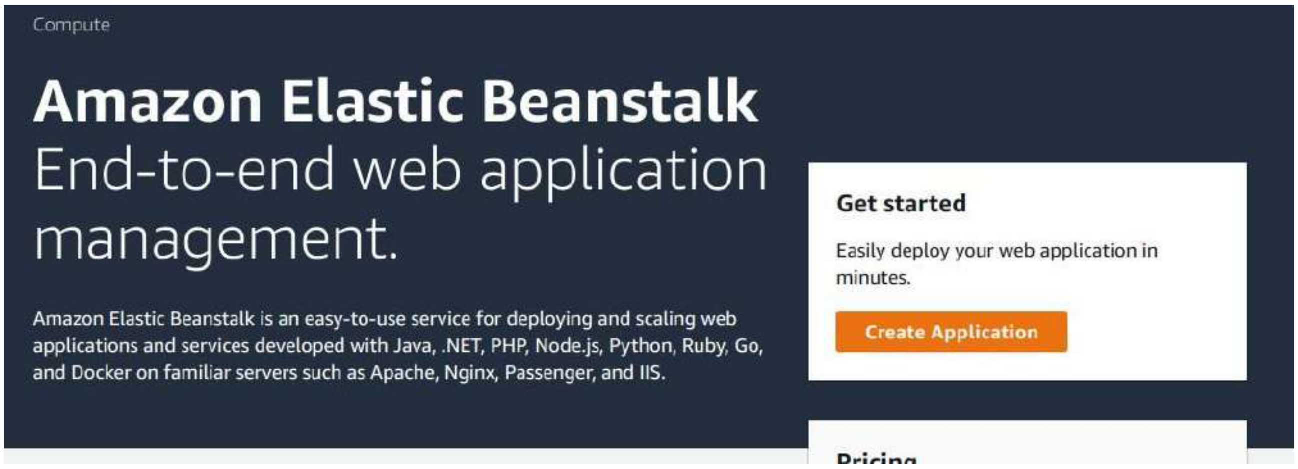
You will create the pipeline using AWS CodePipeline, a service that builds, tests, and deploys your code every time there is a code change. You will use your GitHub account, an Amazon Simple Storage Service (S3) bucket, or an AWS CodeCommit repository as the source location for the sample app's code. You will also use AWS Elastic Beanstalk as the deployment target for the sample app. Your completed pipeline will be able to detect changes made to the source repository containing the sample app and then automatically update your live sample app.

Step1: Create a deployment environment

Your continuous deployment pipeline will need a target environment containing virtual servers, or Amazon EC2 instances, where it will deploy sample code. You will prepare this environment before creating the pipeline.

1) To simplify the process of setting up and configuring EC2 instances for this tutorial, you will spin up a sample environment using AWS Elastic Beanstalk. Elastic Beanstalk lets you easily host web applications without needing to launch, configure, or operate virtual servers on your own. It automatically provisions and operates the infrastructure (e.g. virtual servers, load balancers, etc.) and provides the application stack (e.g. OS, language and framework, web and application server, etc.) for you.

2) Name your web app and choose PHP from the drop-down menu(or any other language you are interested in) and then click Create Application.



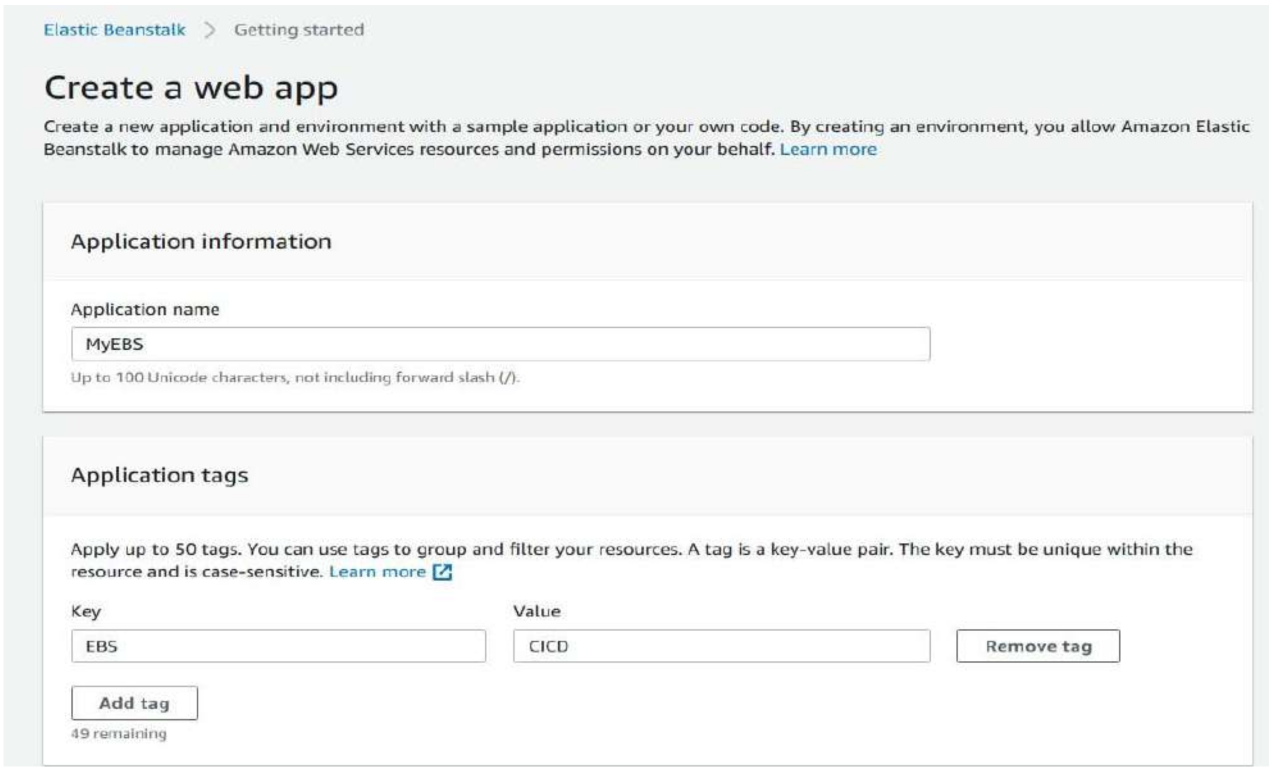
Compute

Amazon Elastic Beanstalk
End-to-end web application management.

Amazon Elastic Beanstalk is an easy-to-use service for deploying and scaling web applications and services developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker on familiar servers such as Apache, Nginx, Passenger, and IIS.

Get started

Easily deploy your web application in minutes.



Create a web app

Create a new application and environment with a sample application or your own code. By creating an environment, you allow Amazon Elastic Beanstalk to manage Amazon Web Services resources and permissions on your behalf. Learn more

Application information

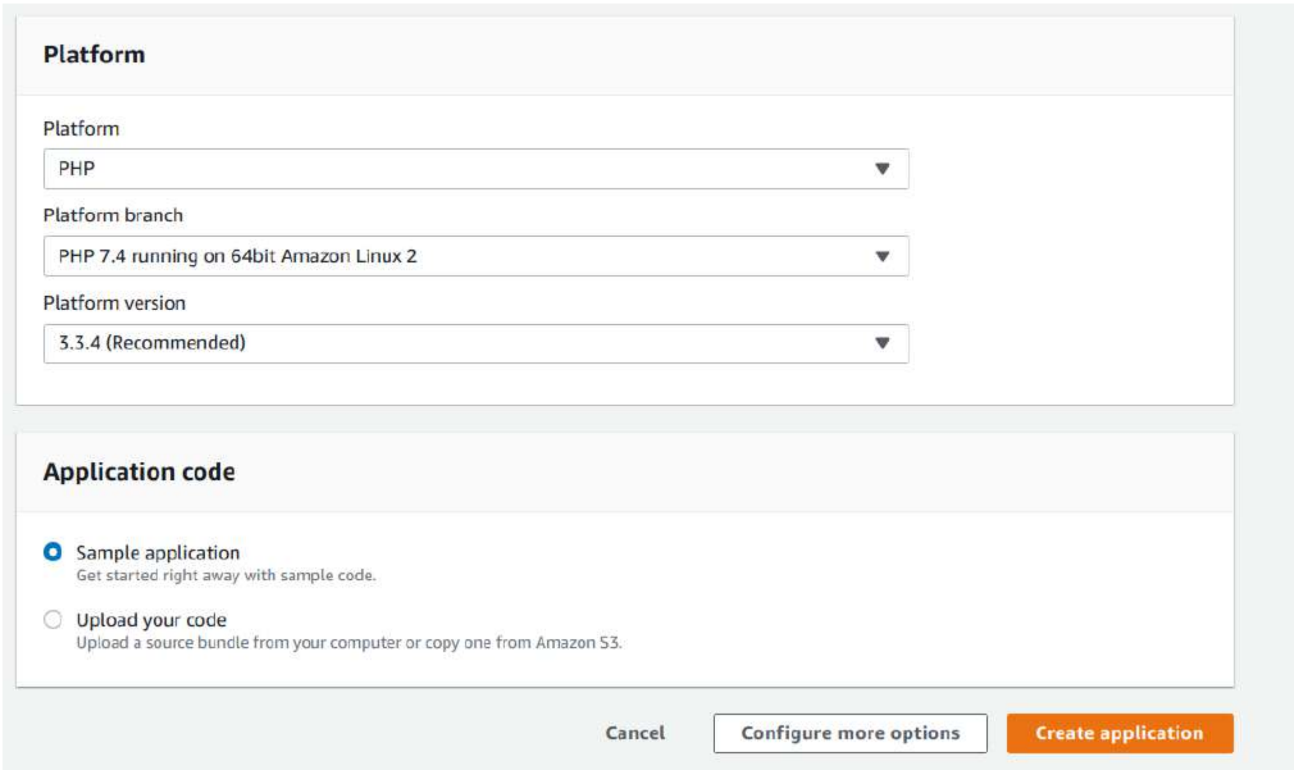
Application name: MyEBS

Up to 100 Unicode characters, not including forward slash (/).

Application tags

Apply up to 50 tags. You can use tags to group and filter your resources. A tag is a key-value pair. The key must be unique within the resource and is case-sensitive. Learn more

Key	Value
EBS	CICD


Platform

Platform: PHP

Platform branch: PHP 7.4 running on 64bit Amazon Linux 2

Platform version: 3.3.4 (Recommended)

Application code

[✓ Sample application] Get started right away with sample code.

Upload your code — Upload a source bundle from your computer or copy one from Amazon S3.

3) Elastic Beanstalk will begin creating a sample environment for you to deploy your application to. It will create an Amazon EC2 instance, a security group, an Auto Scaling group, an Amazon S3 bucket, Amazon CloudWatch alarms, and a domain name for your application.

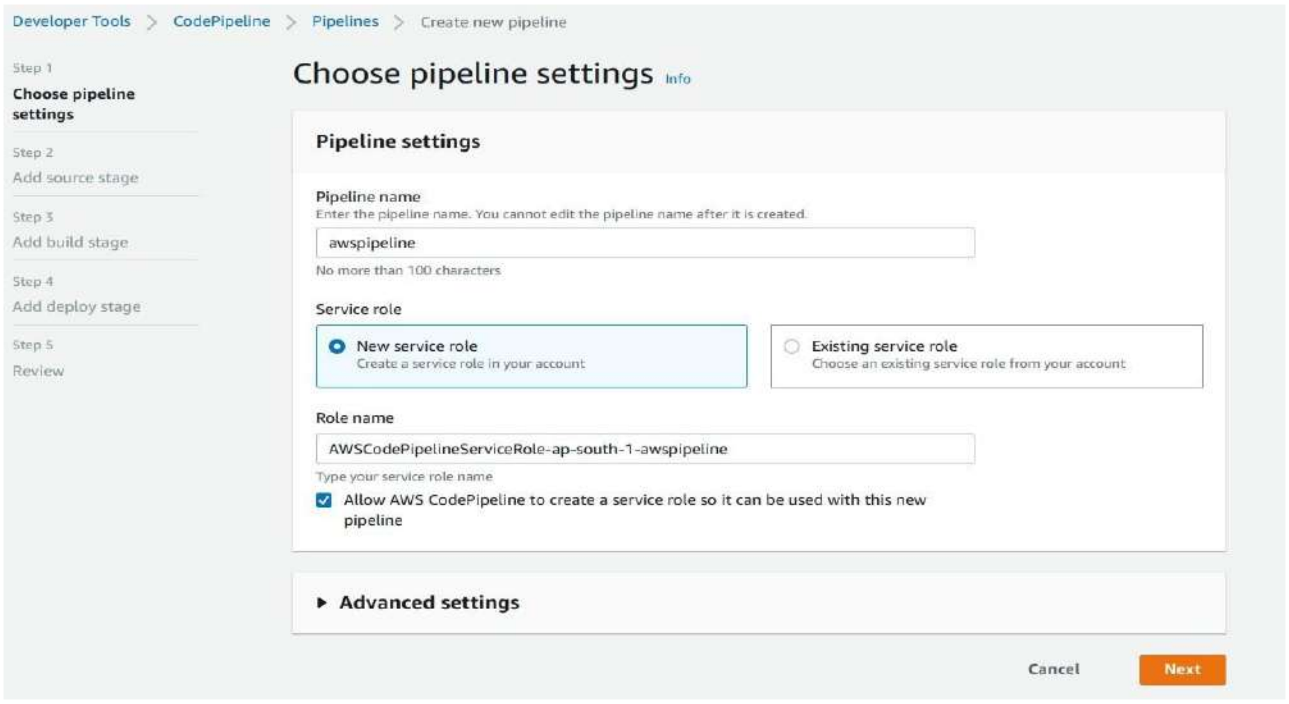
Note: This will take several minutes to complete.

Step2: Get a copy of the sample code

In this step, you will retrieve a copy of the sample app's code and choose a source to host the code.

The pipeline takes code from the source and then performs actions on it.

You can use one of three options as your source: a GitHub repository, an Amazon S3 bucket, or an AWS CodeCommit repository. Select your preference and follow the steps below:


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

Choose pipeline settings Info

Pipeline settings

Pipeline name
Enter the pipeline name. You cannot edit the pipeline name after it is created.
awspipeline
No more than 100 characters

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-ap-south-1-awspipeline
Type your service role name
☑ Allow AWS CodePipeline to create a service role so it can be used with this new pipeline

▸ Advanced settings

Cancel Next

a. If you plan to use Amazon S3 as your source, you will retrieve the sample code from the AWS GitHub repository, save it to your computer, and upload it to an Amazon S3 bucket.

- Visit our GitHub repository containing the sample code at

  https://github.com/imoisharma/aws-codepipeline-s3-codedeploy-linux-2.0

- Click the dist folder.

b. Save the source files to your computer:

- Click the file named aws-codepipeline-s3-aws-codedeploy_linux.zip
- Click View Raw.
- Save the sample file to your local computer.

c. open the Amazon S3 console and create your Amazon S3 bucket:

- Click Create Bucket
- Bucket Name: type a unique name for your bucket, such as awscodepipeline-demobucket-variables. All bucket names in Amazon S3 must be unique, so use one of your own, not one with the name shown in the example.
- Region: In the drop-down, select the region where you will create your pipeline, such as ap-South-1
- Click Create.

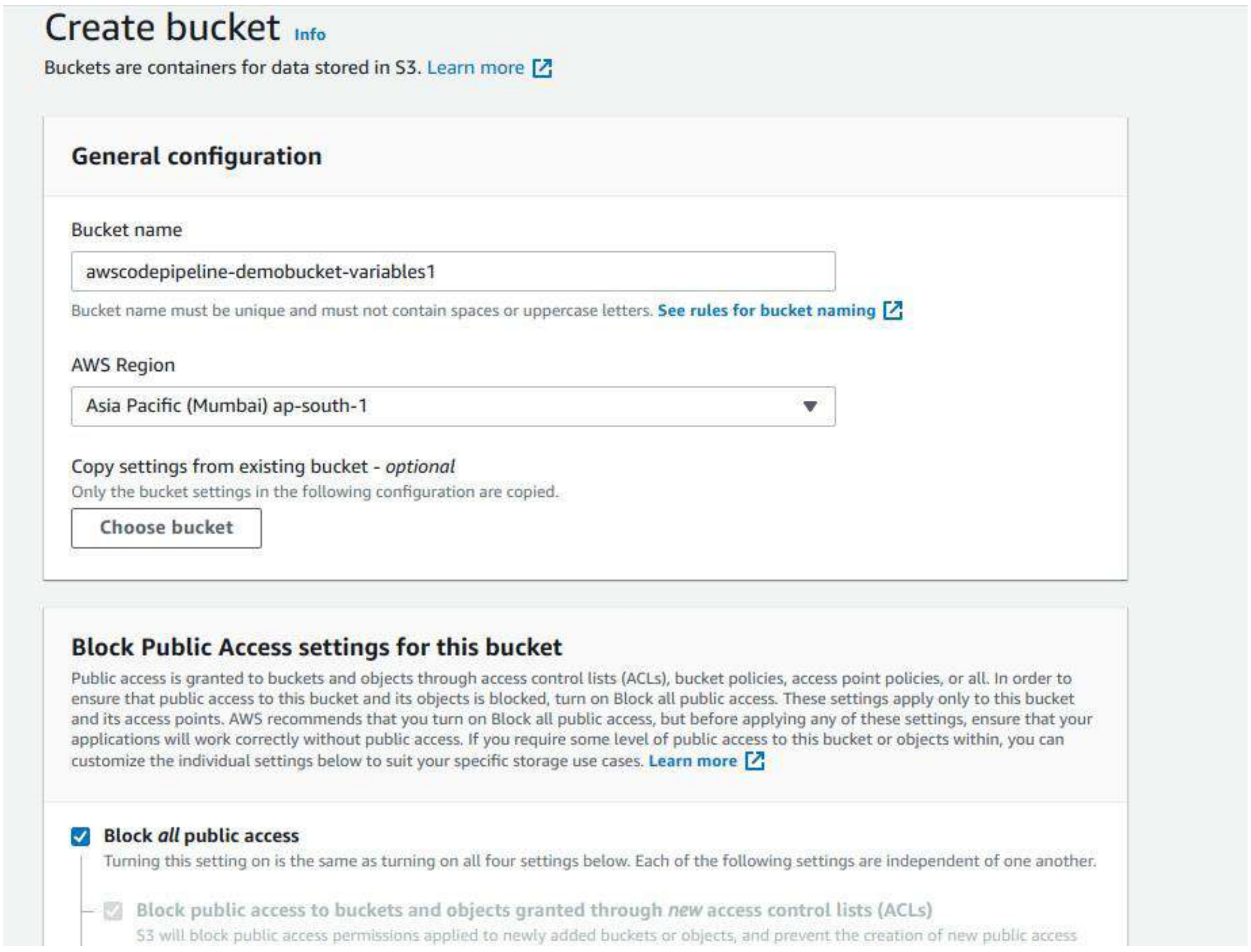
d.The console displays the newly created bucket, which is empty.

- Click Properties.
- Expand Versioning and select Enable Versioning. When versioning is enabled, Amazon S3 saves every version of every object in the bucket.

---

e. You will now upload the sample code to the Amazon S3 bucket:

- Click Upload.
- Follow the on-screen directions to upload the .zip file containing the sample code you downloaded from GitHub.



Create bucket Info

Buckets are containers for data stored in S3. Learn more

General configuration

Bucket name

awscodepipeline-demobucket-variables1

Bucket name must be unique and must not contain spaces or uppercase letters. See rules for bucket naming

AWS Region

Asia Pacific (Mumbai) ap-south-1

Copy settings from existing bucket - *optional*

Only the bucket settings in the following configuration are copied.

Block Public Access settings for this bucket

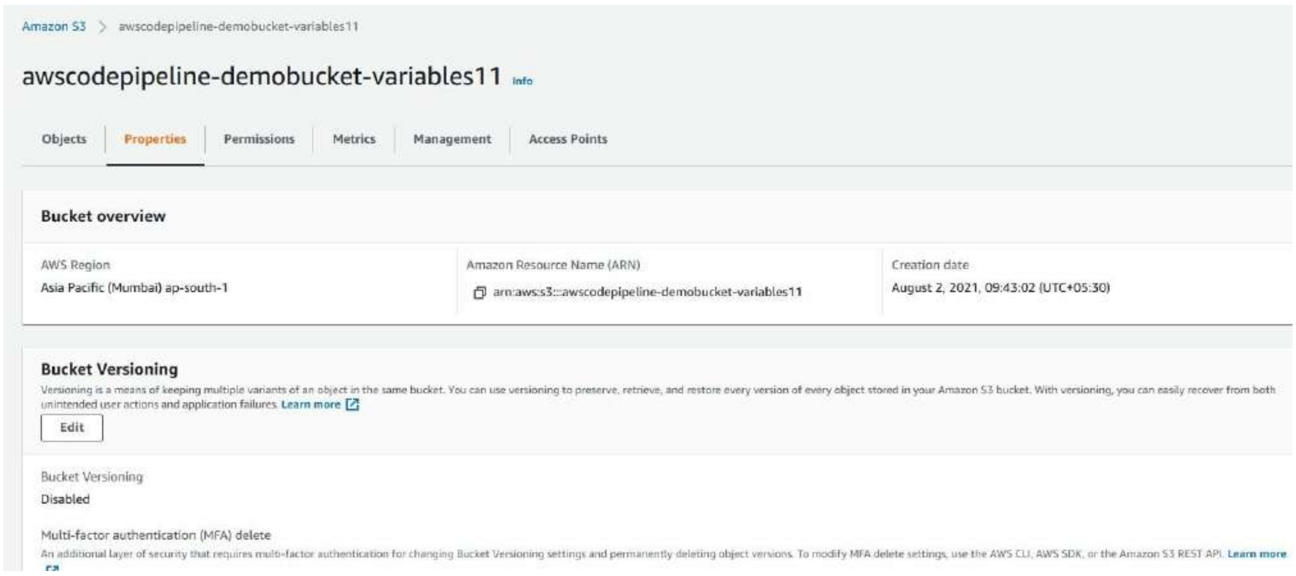
Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more

✓ Block *all* public access

Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

✓ Block public access to buckets and objects granted through *new* access control lists (ACLs)

S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access


Amazon S3 > awscodepipeline-demobucket-variables11

awscodepipeline-demobucket-variables11 Info

Bucket overview

AWS Region
Asia Pacific (Mumbai) ap-south-1

Amazon Resource Name (ARN)
arn:aws:s3:::awscodepipeline-demobucket-variables11

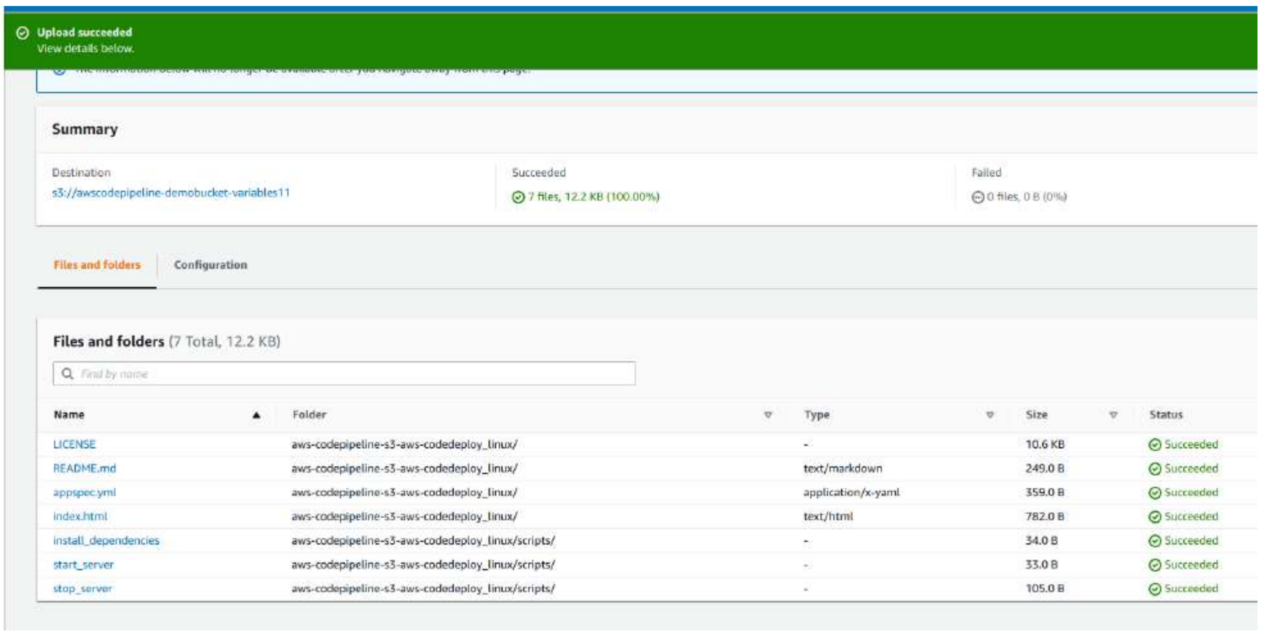
Creation date
August 2, 2021, 09:43:02 (UTC+05:30)

Bucket Versioning
Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. Learn more

Bucket Versioning
Disabled

Multi-factor authentication (MFA) delete
An additional layer of security that requires multi-factor authentication for changing Bucket Versioning settings and permanently deleting object versions. To modify MFA delete settings, use the AWS CLI, AWS SDK, or the Amazon S3 REST API. Learn more

you can upload directly zip file here from https://github.com/imoisharma/aws-codepipeline-s3-codedeploy-linux-2.0


Upload succeeded
View details below.

Summary

Destination
s3://awscodepipeline-demobucket-variables11

Succeeded
7 files, 12.2 KB (100.00%)

Failed
0 files, 0 B (0%)

Files and folders (7 Total, 12.2 KB)

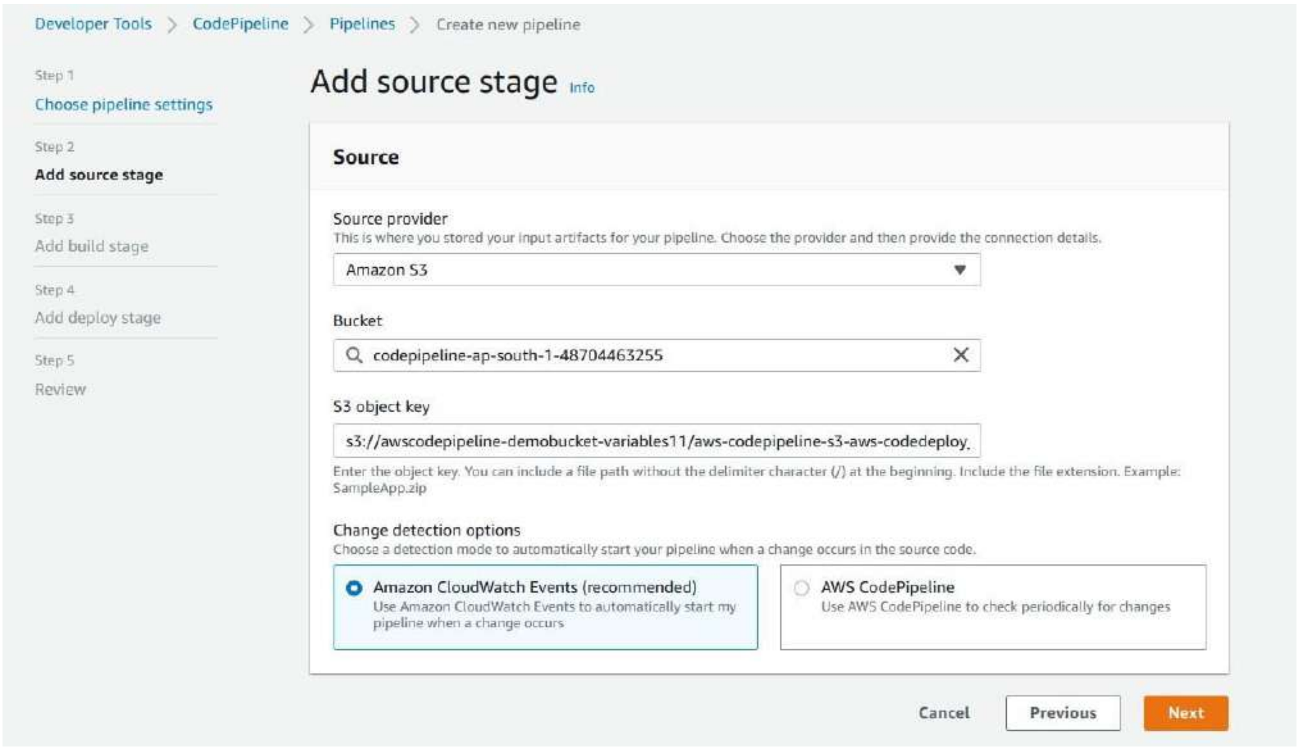
Name	Folder	Type	Size	Status
LICENSE	aws-codepipeline-s3-aws-codedeploy_linux/	-	10.6 KB	Succeeded
README.md	aws-codepipeline-s3-aws-codedeploy_linux/	text/markdown	249.0 B	Succeeded
appspec.yml	aws-codepipeline-s3-aws-codedeploy_linux/	application/x-yaml	359.0 B	Succeeded
index.html	aws-codepipeline-s3-aws-codedeploy_linux/	text/html	782.0 B	Succeeded
install_dependencies	aws-codepipeline-s3-aws-codedeploy_linux/scripts/	-	34.0 B	Succeeded
start_server	aws-codepipeline-s3-aws-codedeploy_linux/scripts/	-	33.0 B	Succeeded
stop_server	aws-codepipeline-s3-aws-codedeploy_linux/scripts/	-	105.0 B	Succeeded

Step3: Create your Pipeline

In this step, you will create and configure a simple pipeline with two actions: source and deploy. You will provide CodePipeline with the locations of your source repository and deployment environment.

A true continuous deployment pipeline requires a build stage, where code is compiled and unit tested. CodePipeline lets you plug your preferred build provider into your pipeline. However, in this we will skip the build stage.

Goto Pipeline again and create it



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

Source

Source provider
This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.
Amazon S3

Bucket
codepipeline-ap-south-1-48704463255

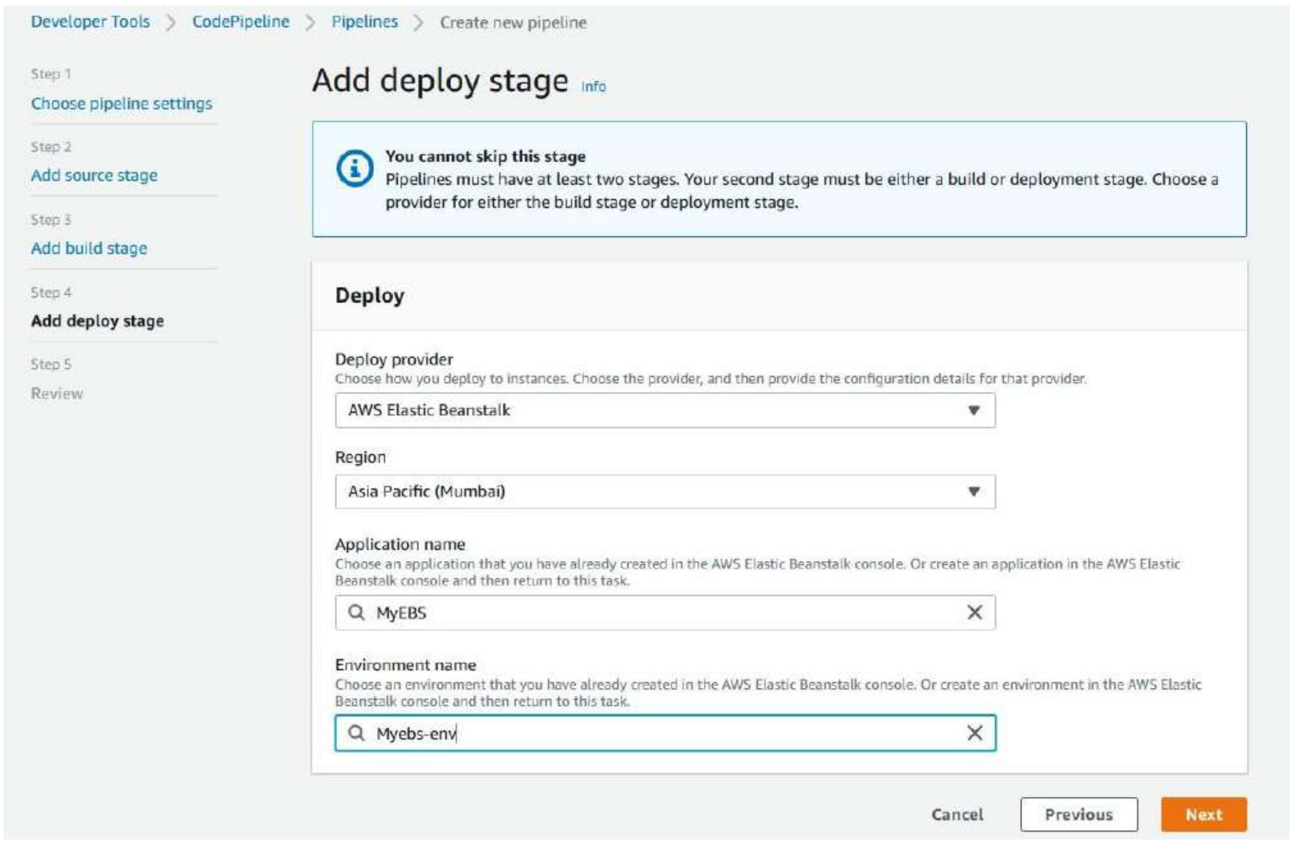
S3 object key
s3://awscodepipeline-demobucket-variables11/aws-codepipeline-s3-aws-codedeploy
Enter the object key. You can include a file path without the delimiter character (/) at the beginning. Include the file extension. Example: SampleApp.zip

Change detection options
Choose a detection mode to automatically start your pipeline when a change occurs in the source code.
[✓ Amazon CloudWatch Events (recommended)] Use Amazon CloudWatch Events to automatically start my pipeline when a change occurs
AWS CodePipeline — Use AWS CodePipeline to check periodically for changes

In above you can give zip file name in S3 object Key and choose bucket name which you created

In Step 4: Deploy Stage:

- Deployment provider: Click AWS Elastic Beanstalk.
- Application name: MYEBS.
- Environment name: Click Myebs-env.
- Click Next step.


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 deploy stage Info

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.

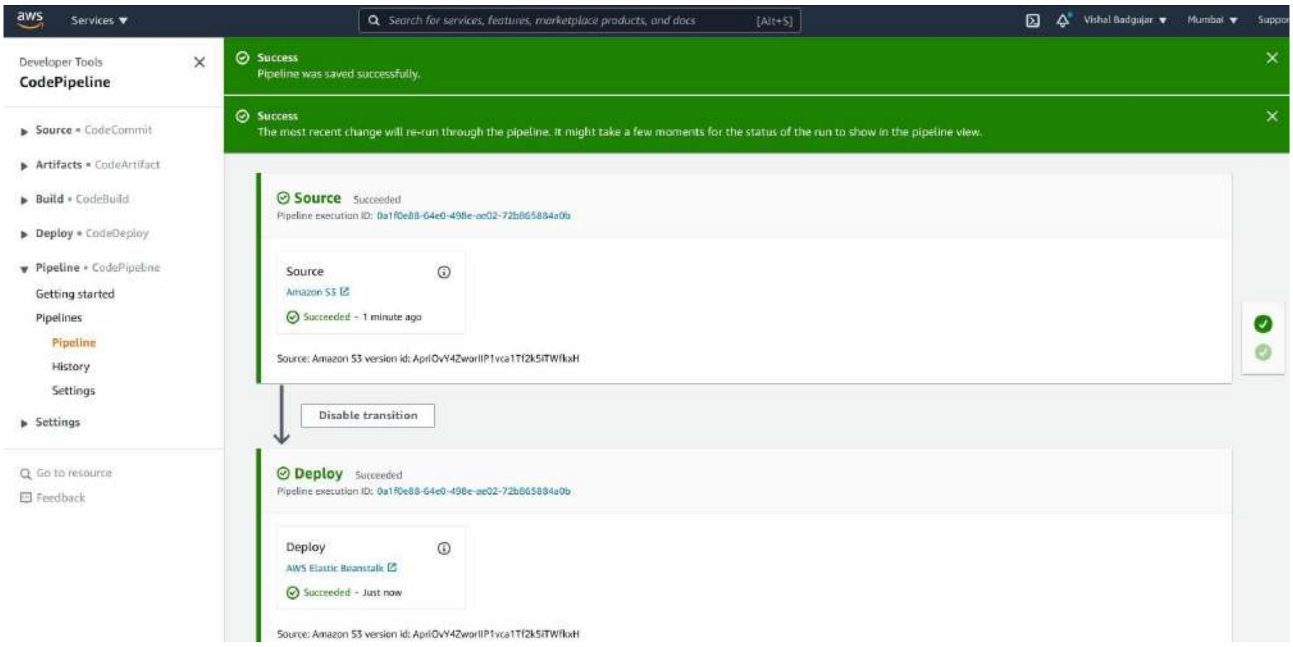
Deploy

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

Region
Asia Pacific (Mumbai)

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.
MyEBS

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.
Myebs-env



Success
Pipeline was saved successfully.

Success
The most recent change will re-run through the pipeline. It might take a few moments for the status of the run to show in the pipeline view.

Source Succeeded
Pipeline execution ID: 0a1f0e88-64e0-498e-ae02-72b865884a0b

Source
Amazon S3
Succeeded - 1 minute ago

Source: Amazon S3 version id: AprlOvY4ZworlIP1vca1Tf2k5iTWfkxH

Deploy Succeeded
Pipeline execution ID: 0a1f0e88-64e0-498e-ae02-72b865884a0b

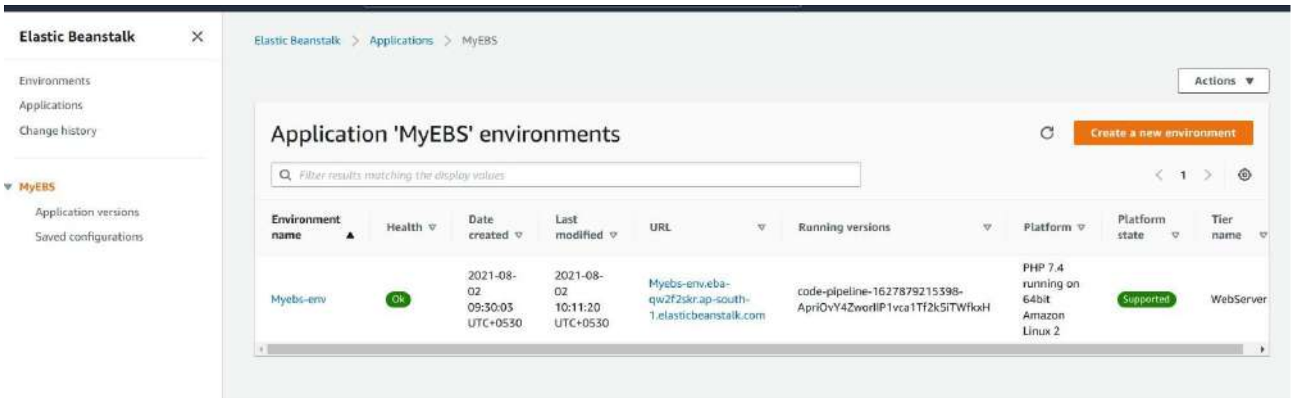
Deploy
AWS Elastic Beanstalk
Succeeded - Just now

Source: Amazon S3 version id: AprlOvY4ZworlIP1vca1Tf2k5iTWfkxH

After your pipeline is created, the pipeline status page appears and the pipeline automatically starts to run. You can view progress as well as success and failure messages as the pipeline perform each action.

To verify your pipeline ran successfully, monitor the progress of the pipeline as it moves through each stage. The status of each stage will change from No executions yet to In Progress, and then to either Succeeded or Failed. The pipeline should complete the first run within a few minutes.

Now go to your EBS environment and click on the URL to view the sample website you deployed.

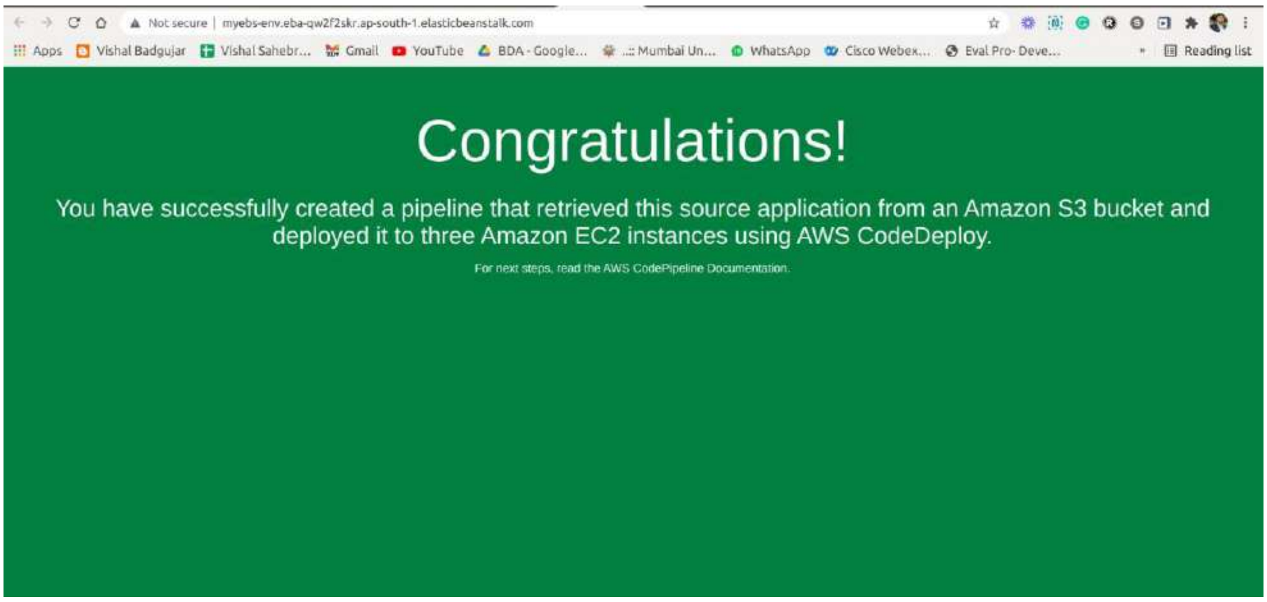


Application 'MyEBS' environments

Environment name	Health	Date created	Last modified	URL	Running versions	Platform	Platform state	Tier name
Myebs-env	Ok	2021-08-02 09:30:03 UTC+0530	2021-08-02 10:11:20 UTC+0530	Myebs-env.eba-qw2f2skr.ap-south-1.elasticbeanstalk.com	code-pipeline-1627879215398-ApriOvY4ZworlIP1vca1Tf2k5iTWfkxH	PHP 7.4 running on 64bit Amazon Linux 2	Supported	WebServer

You have successfully created an automated software release pipeline using AWS CodePipeline!

Using CodePipeline, you created a pipeline that uses GitHub, Amazon S3, or AWS CodeCommit as the source location for application code and then deploys the code to an Amazon EC2 instance managed by AWS Elastic Beanstalk.



Congratulations!

You have successfully created a pipeline that retrieved this source application from an Amazon S3 bucket and deployed it to three Amazon EC2 instances using AWS CodeDeploy.

For next steps, read the AWS CodePipeline Documentation.

Step 5: Commit a change and then update your app

In this step, you will revise the sample code and commit the change to your repository. CodePipeline will detect your updated sample code and then automatically initiate deploying it to your EC2 instance via Elastic Beanstalk.

Note that the sample web page you deployed refers to AWS CodeDeploy, a service that automates code deployments. In CodePipeline, CodeDeploy is an alternative to using Elastic Beanstalk for deployment actions. Let's update the sample code so that it correctly states that you deployed the sample using Elastic Beanstalk.

a. Visit your own copy of the repository that you forked in GitHub.

- Open index.html
- Select the Edit icon

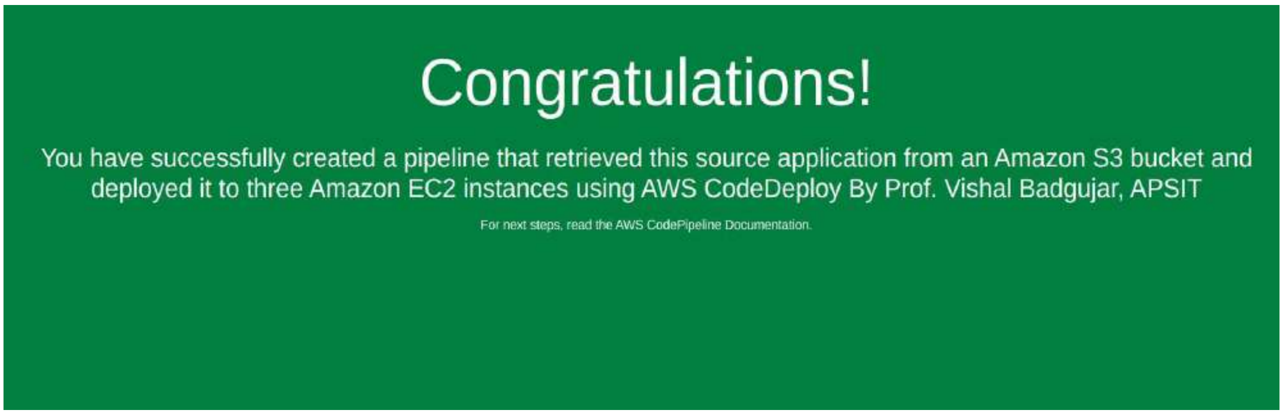
b. Update the webpage by copying and pasting the following text on line 30:

c. Commit the change to your repository.

d. Return to your pipeline in the CodePipeline console. In a few minutes, you should see the Source change to blue, indicating that the pipeline has detected the changes you made to your source repository. Once this occurs, it will automatically move the updated code to Elastic Beanstalk.

- After the pipeline status displays Succeeded, in the status area for the Beta stage, click AWS Elastic Beanstalk.

e. The AWS Elastic Beanstalk console opens with the details of the deployment. Select the environment you created earlier. And click the URL again from EBS environment again.


Congratulations!

You have successfully created a pipeline that retrieved this source application from an Amazon S3 bucket and deployed it to three Amazon EC2 instances using AWS CodeDeploy By Prof. Vishal Badgujar, APSIT

For next steps, read the AWS CodePipeline Documentation.

Step 6: Clean up your resources

To avoid future charges, you will delete all the resources you launched throughout this tutorial, which includes the pipeline, the Elastic Beanstalk application, and the source you set up to host the code.

a. First, you will delete your pipeline:

- In the pipeline view, click Edit.
- Click Delete.
- Type in the name of your pipeline and click Delete.

b. Second, delete your Elastic Beanstalk application:

- Visit the Elastic Beanstalk console.
- Click Actions.
- Then click Terminate Environment.

You have successfully created an automated software release pipeline using AWS CodePipeline! Using CodePipeline, you created a pipeline that uses GitHub, Amazon S3, or AWS CodeCommit as the source location for application code and then deploys the code to an Amazon EC2 instance managed by AWS Elastic Beanstalk. Your pipeline will automatically deploy your code every time there is a code change.

Conclusion: Write your own findings.