

## Advance DevOps Lab Experiment 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.

Roll No.	22
Name	Sarthak Harade
Class	D15B
Subject	Advance DevOps Lab
LO Mapped	LO1: To understand the fundamentals of Cloud Computing and be fully proficient with Cloud based DevOps solution deployment options to meet your business requirements.
Grade:	

**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 is a key practice in modern DevOps, enabling organizations to streamline their software release process by automating the deployment of application updates. It allows for the seamless delivery of code revisions to production environments without requiring explicit approval from a developer, thereby reducing time-to-market and enhancing the overall efficiency of the development lifecycle.

AWS CodePipeline is a continuous integration and continuous delivery (CI/CD) service that facilitates the building, testing, and deployment of code whenever there is a change in the source code repository. By automating these steps, CodePipeline ensures that new features, bug fixes, and updates are reliably and consistently delivered to users. One of the critical components of a continuous deployment pipeline is the deployment environment, which is typically made up of virtual servers or containers that host the application.

Amazon Elastic Beanstalk (EBS) is a Platform as a Service (PaaS) offering that simplifies the deployment and management of applications in the cloud. It abstracts the underlying infrastructure, such as EC2 instances, load balancers, and scaling configurations, allowing developers to focus on writing code without worrying about provisioning and maintaining the infrastructure.

In a typical AWS CodePipeline workflow, the source code for an application is stored in a version control system like GitHub, an S3 bucket, or AWS CodeCommit. The pipeline monitors this source repository for changes and triggers a series of automated actions whenever a change is detected.

These actions might include building the application, running automated tests, and finally deploying the code to a live environment. The deployment target in this setup could be an Amazon EC2 instance managed by Elastic Beanstalk, which takes care of the deployment details like setting up the necessary resources, deploying the code, and ensuring that the application is running smoothly. This integration with Elastic Beanstalk offers an out-of-the-box deployment solution that is both scalable and resilient.

AWS CodePipeline's integration with Elastic Beanstalk ensures that every code change goes through a consistent deployment process, thereby minimizing human errors and ensuring that the application remains stable and reliable. This automated process not only accelerates the development cycle but also improves the quality of the software by providing immediate feedback on the code's performance in a production-like environment.

**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
- Unused access
- Analyzer settings
- Credential report
- Organization activity

**Roles (5)** 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.

Search

<input type="checkbox"/>	Role name	Trusted entities	Last activity
<input type="checkbox"/>	<a href="#">aws-elasticbeanstalk-service-role</a>	AWS Service: elasticbeanstalk	48 days ago
<input type="checkbox"/>	<a href="#">AWSCodePipelineServiceRole-us-east-1-aryan-pipeline</a>	AWS Service: codepipeline	14 days ago
<input type="checkbox"/>	<a href="#">AWSServiceRoleForAutoScaling</a>	AWS Service: autoscaling (Service-Link)	14 minutes ago
<input type="checkbox"/>	<a href="#">AWSServiceRoleForSupport</a>	AWS Service: support (Service-Link)	-
<input type="checkbox"/>	<a href="#">AWSServiceRoleForTrustedAdvisor</a>	AWS Service: trustedadvisor (Service-Link)	-

**Roles Anywhere** Info

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

Manage

**Access AWS from your non AWS workloads**

**X.509 Standard**  
Use your own existing PKI infrastructure or use AWS

**Temporary credentials**  
Use temporary credentials with ease and benefit

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Add EC2 for a service or use case.

**Create role | IAM | Global**

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.

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Give name to the role.

The screenshot shows the AWS IAM console interface for creating a role. The breadcrumb navigation is IAM > Roles > Create role. The left sidebar shows three steps: Step 1: Select trusted entity, Step 2: Add permissions, and Step 3: Name, review, and create (which is the active step). The main content area is titled 'Name, review, and create'. Under 'Role details', there is a 'Role name' field containing 'sarthak\_iam' and a 'Description' field containing 'Allows EC2 instances to call AWS services on your behalf.' Below this is a 'Trust policy' section showing a JSON snippet. At the bottom right, there is an 'Edit' button.

Step 1: Select trusted entities

Trust policy

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
```

Required policies (permissions) to be added while creating IAM user.

The screenshot shows the AWS IAM console interface for adding permissions to a role. The breadcrumb navigation is IAM > Roles > Create role. The left sidebar shows three steps: Step 1: Select trusted entity, Step 2: Add permissions (which is the active step), and Step 3: Add tags. The main content area is titled 'Step 2: Add permissions'. It includes a 'Permissions policy summary' table with three rows of AWS managed policies. Below the table is a 'Step 3: Add tags' section with an 'Add tags - optional' link and a text area for tags. At the bottom right, there are 'Cancel', 'Previous', and 'Create role' buttons.

Step 2: Add permissions

Permissions policy summary

Policy name	Type	Attached as
<a href="#">AWSElasticBeanstalkMulticontainerDocker</a>	AWS managed	Permissions policy
<a href="#">AWSElasticBeanstalkWebTier</a>	AWS managed	Permissions policy
<a href="#">AWSElasticBeanstalkWorkerTier</a>	AWS managed	Permissions policy

Step 3: Add tags

Add tags - optional [Info](#)

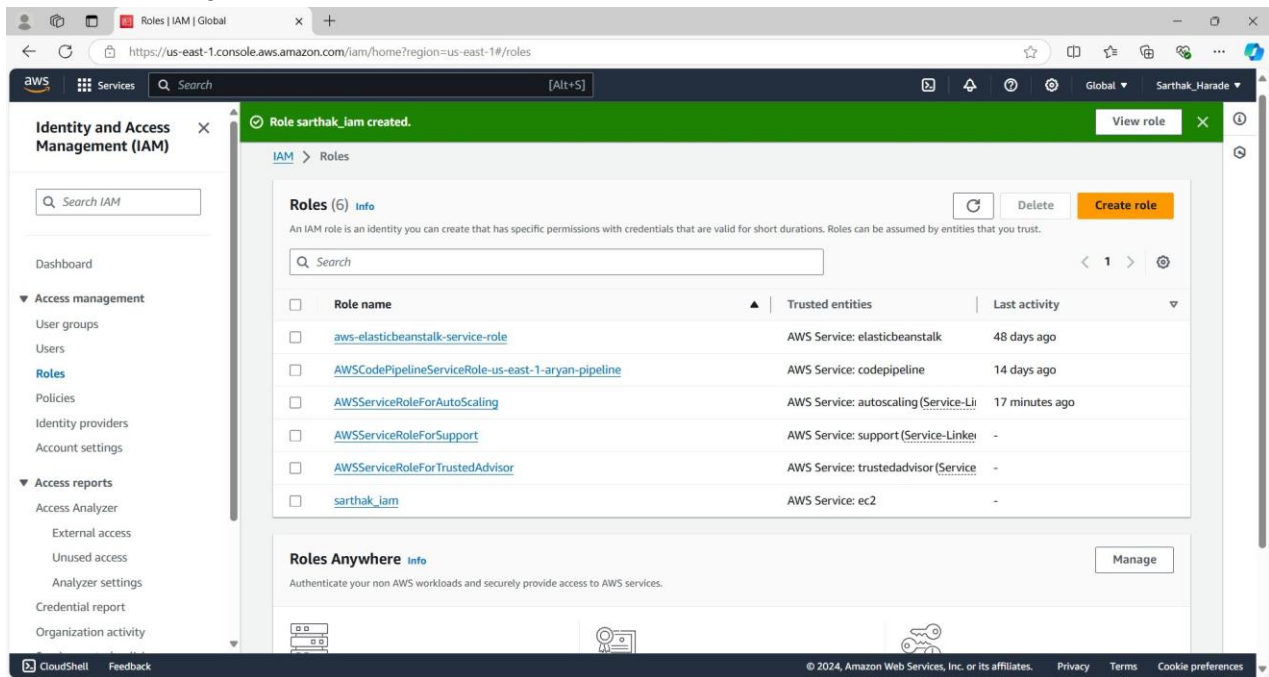
Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

No tags associated with the resource.

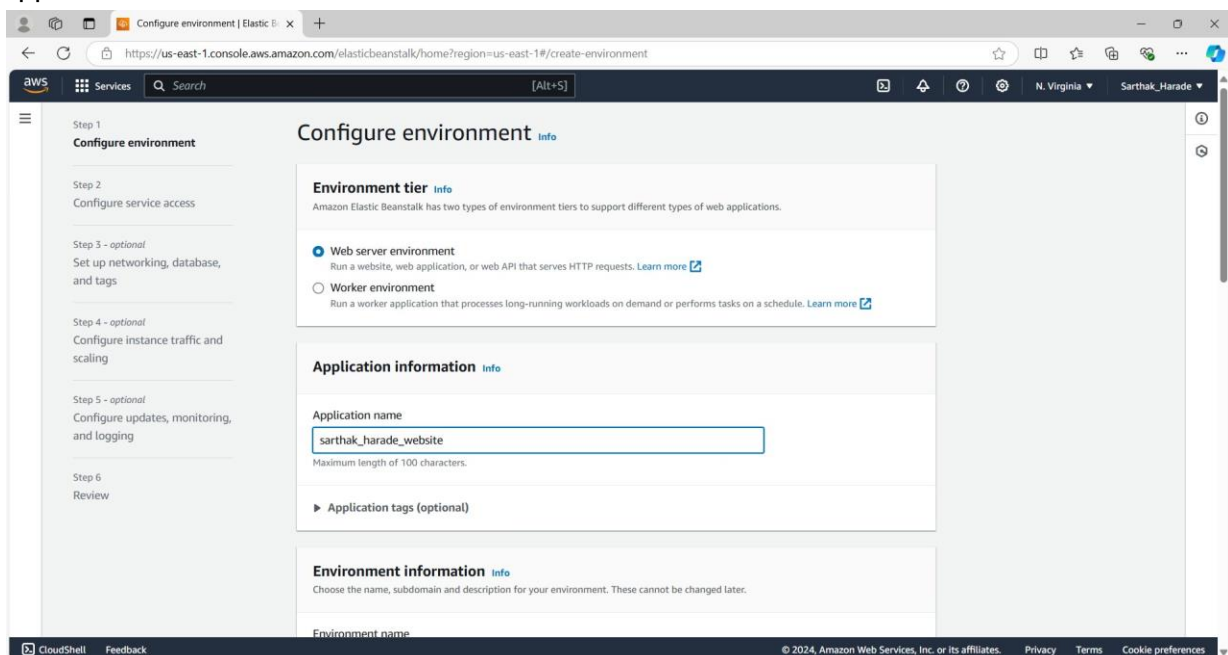
[Add new tag](#)

You can add up to 50 more tags.

## IAM role is being created



Go to the Elastic beanstalk and create an application. Give the appropriate name for the application.



Select the platform as PHP.

The screenshot shows the 'Platform info' configuration page in the AWS Elastic Beanstalk console. The 'Platform type' is set to 'Managed platform'. The 'Platform' dropdown is set to 'PHP'. The 'Platform branch' is set to 'PHP 8.3 running on 64bit Amazon Linux 2023'. The 'Platform version' is set to '4.3.3 (Recommended)'. The 'Application code' section shows 'Sample application' selected. The page includes a sidebar with navigation links and a footer with copyright information.

Platform info

Platform type

- ☒ Managed platform
- ☐ Custom platform

Platform

PHP

Platform branch

PHP 8.3 running on 64bit Amazon Linux 2023

Platform version

4.3.3 (Recommended)

Application code info

- ☒ Sample application
- ☐ Existing version
- ☐ Upload your code

In an ec2 instance profile, select the created IAM role.

The screenshot shows the 'Configure service access' configuration page in the AWS Elastic Beanstalk console. The 'Service role' is set to 'Use an existing service role'. The 'Existing service roles' dropdown is set to 'sarthak\_iam'. The 'EC2 key pair' is set to 'Choose a key pair'. The 'EC2 instance profile' is set to 'sarthak\_iam'. The page includes a sidebar with navigation links and a footer with copyright information.

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.

sarthak\_iam

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

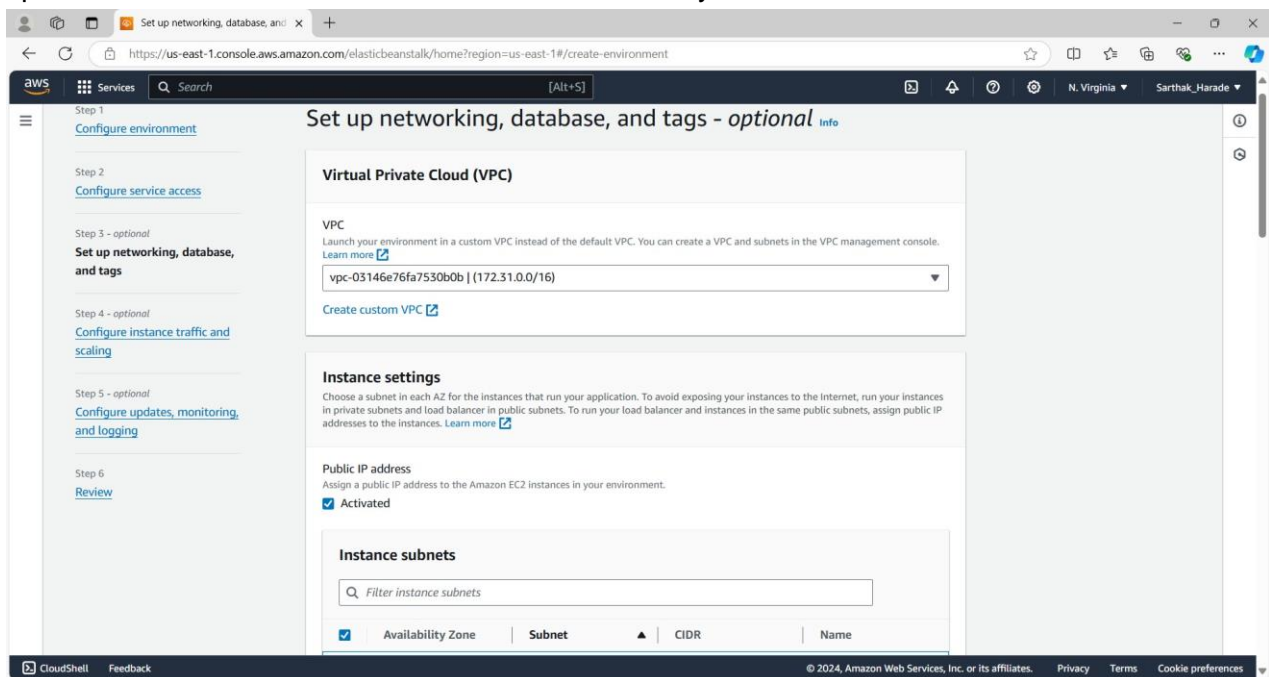
Choose an IAM instance profile with managed policies that allow your EC2 instances to perform required operations.

sarthak\_iam

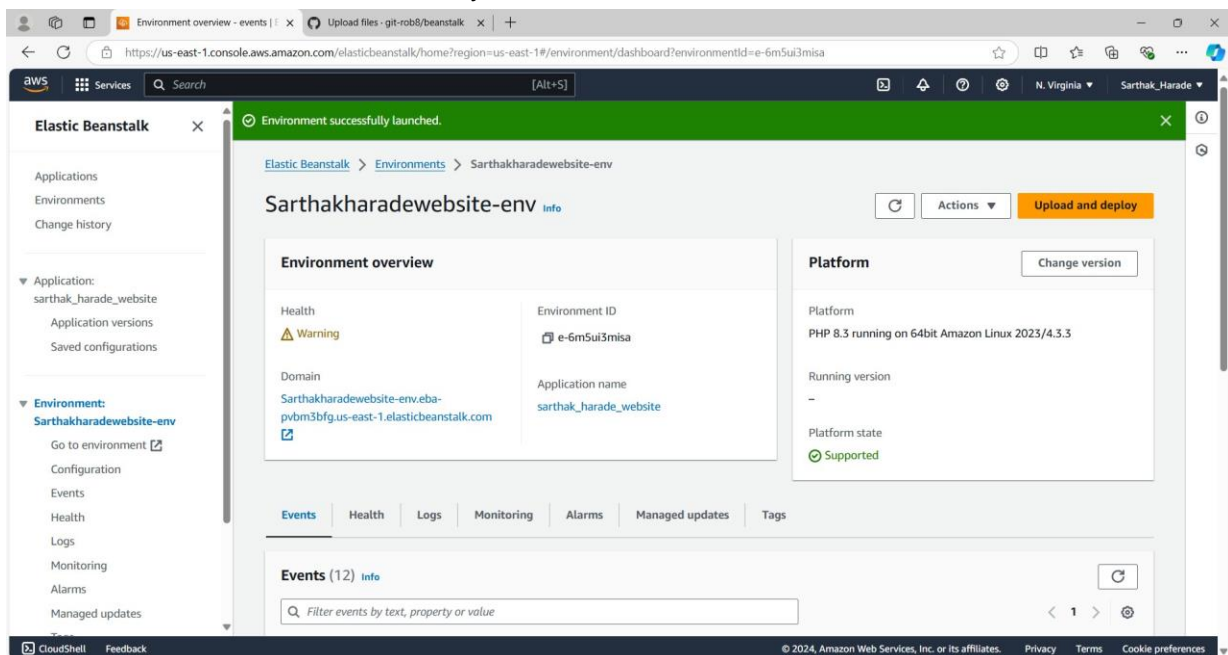
View permission details

Cancel Skip to review Previous Next

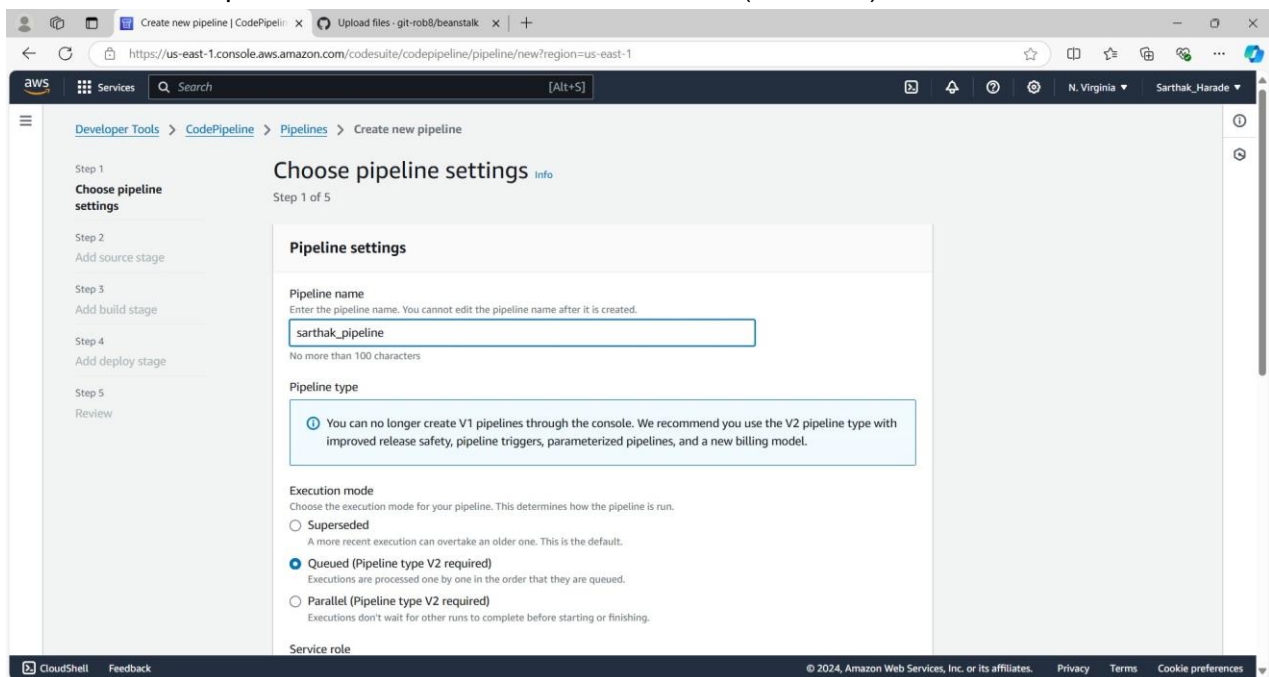
Vpc is to be selected. Public IP address and availability is to be checked.



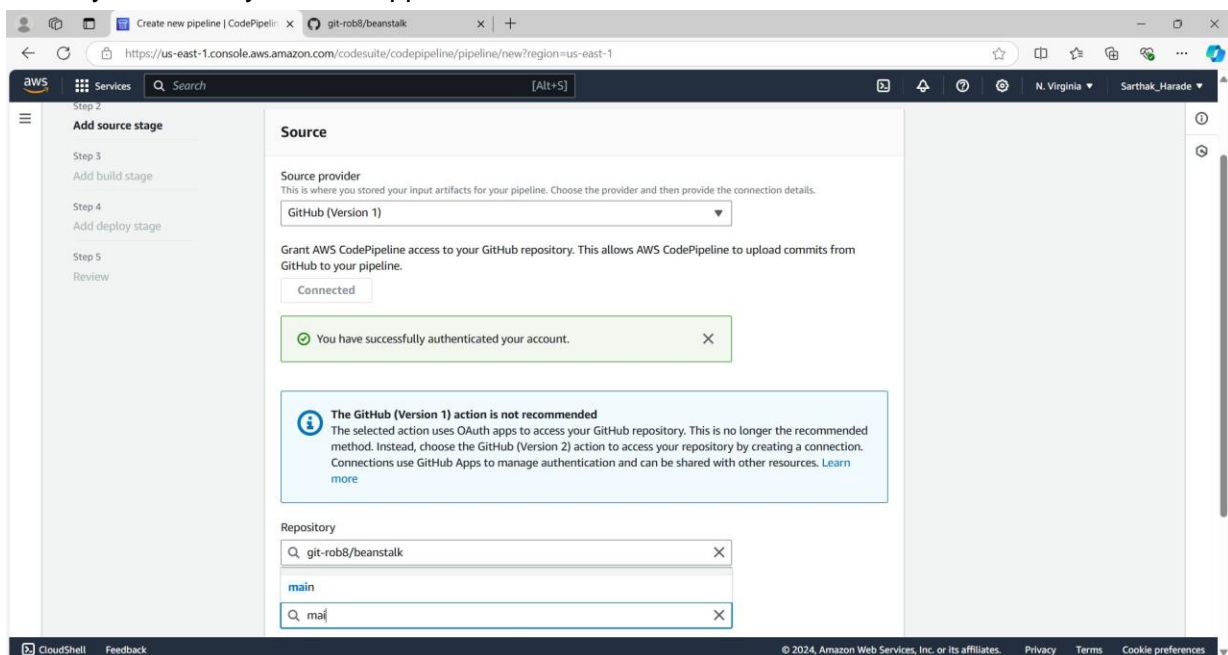
Environment is launched successfully.



Go to the CodePipeline and select the source as GitHub (version 1).



After skipping the build stage, AWS Elastic beanstalk is to be selected in the Deploy Provider. Select your recently created application name and environment name.





Pipeline is created. The Source and Deploy section is also successful.

The image consists of two screenshots from the AWS CodePipeline console, showing the creation and successful execution of a pipeline named 'sarthak\_pipeline'.

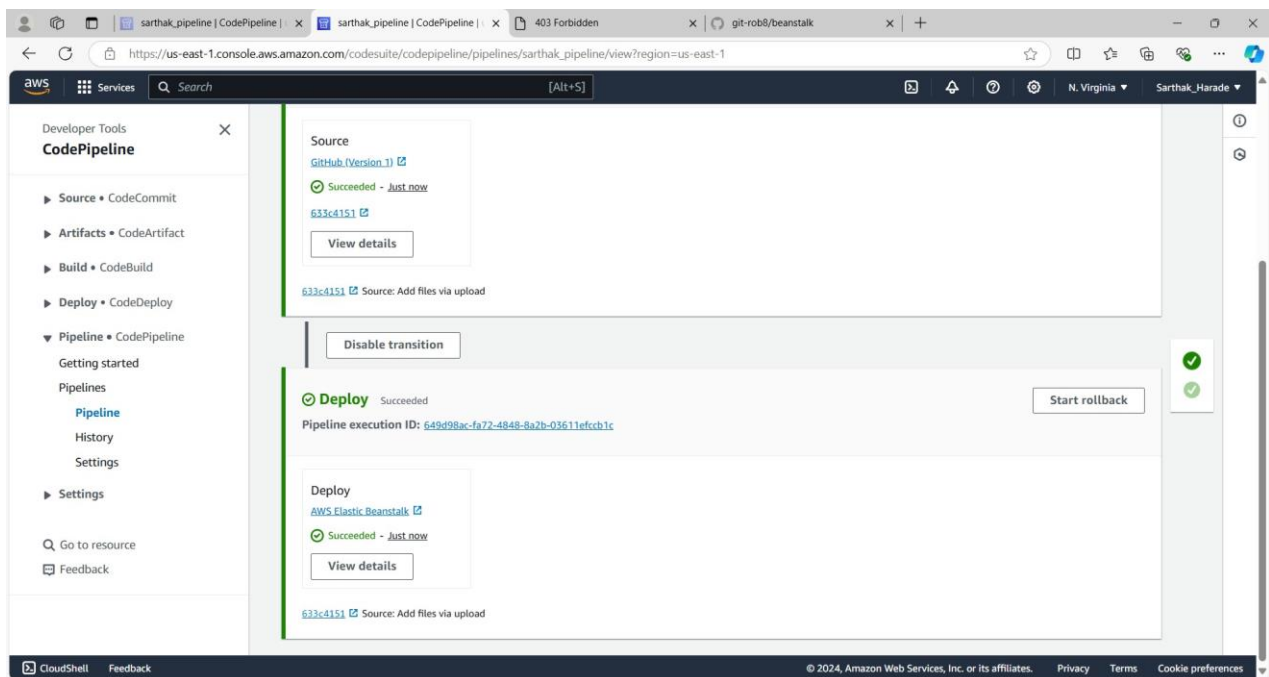
**Top Screenshot: Create new pipeline**

- URL:** <https://us-east-1.console.aws.amazon.com/codesuite/codepipeline/pipeline/new?region=us-east-1>
- Deploy provider:** AWS Elastic Beanstalk
- Region:** US East (N. Virginia)
- Input artifacts:** SourceArtifact
- Application name:** sarthak\_harade\_website
- Environment name:** Sarthakharadewebsite-env
- Buttons:** Cancel, Previous, Next

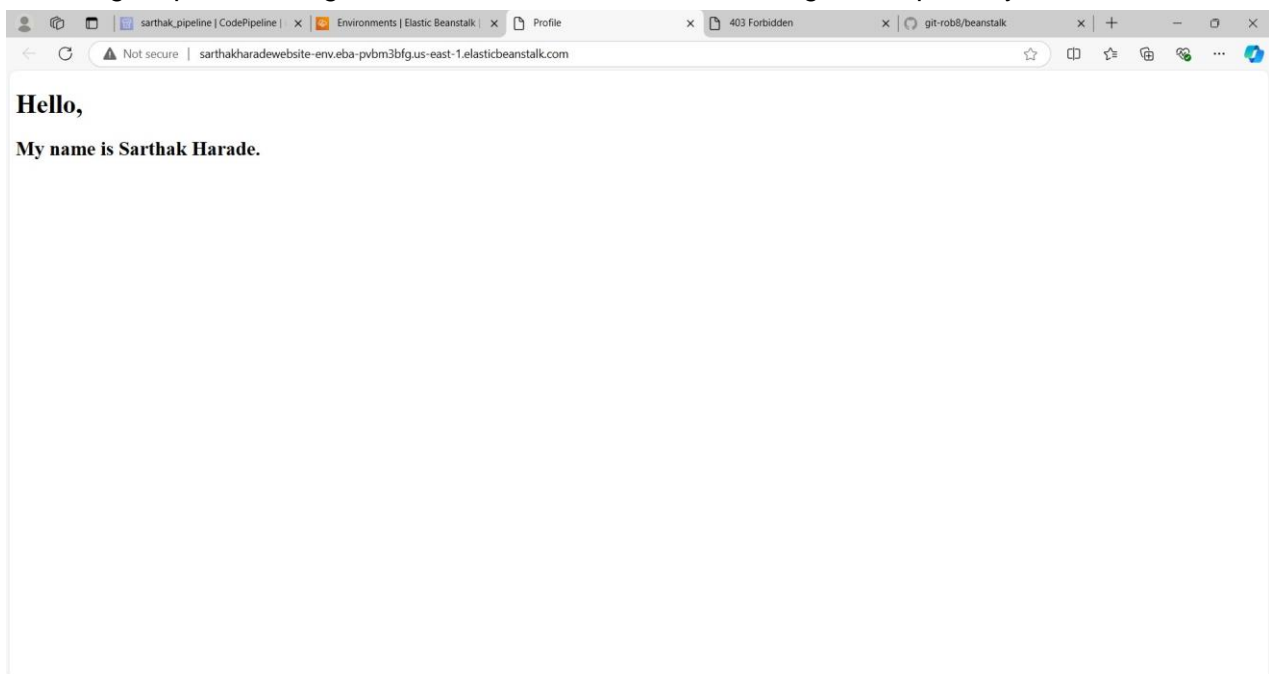
**Bottom Screenshot: Pipeline view**

- URL:** [https://us-east-1.console.aws.amazon.com/codesuite/codepipeline/pipelines/sarthak\\_pipeline/view?region=us-east-1](https://us-east-1.console.aws.amazon.com/codesuite/codepipeline/pipelines/sarthak_pipeline/view?region=us-east-1)
- Success message:** Pipeline was saved successfully.
- Pipeline name:** sarthak\_pipeline
- Pipeline type:** V2
- Execution mode:** QUEUED
- Source section:** Succeeded. Pipeline execution ID: 649d98ac-fa72-4848-8a2b-03611efcch1c. Source: GitHub (Version 1). Succeeded - Just now. 633c4151. View details.
- Buttons:** Notify, Edit, Stop execution, Clone pipeline, Release change, Disable transition.

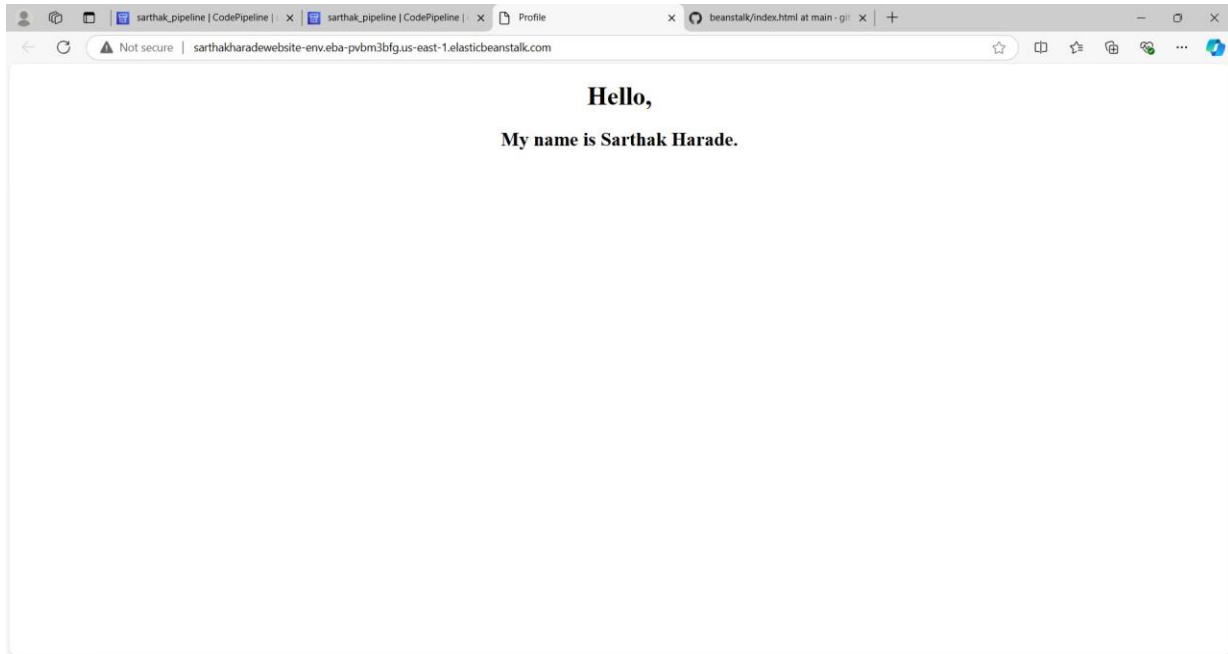
Go to the elastic beanstalk environment and click on domain.



Following output is to be generated of the code which is in the github repository.



Changes are done in the code of the Github repository and it is being directly deployed without any configurations.



**CONCLUSION :** Continuous deployment using AWS CodePipeline and Elastic Beanstalk represents a powerful approach to modern software development, where automation plays a crucial role in delivering high-quality software quickly and efficiently. This method supports the agile methodology by enabling rapid iterations and continuous improvements, leading to more responsive and innovative applications.