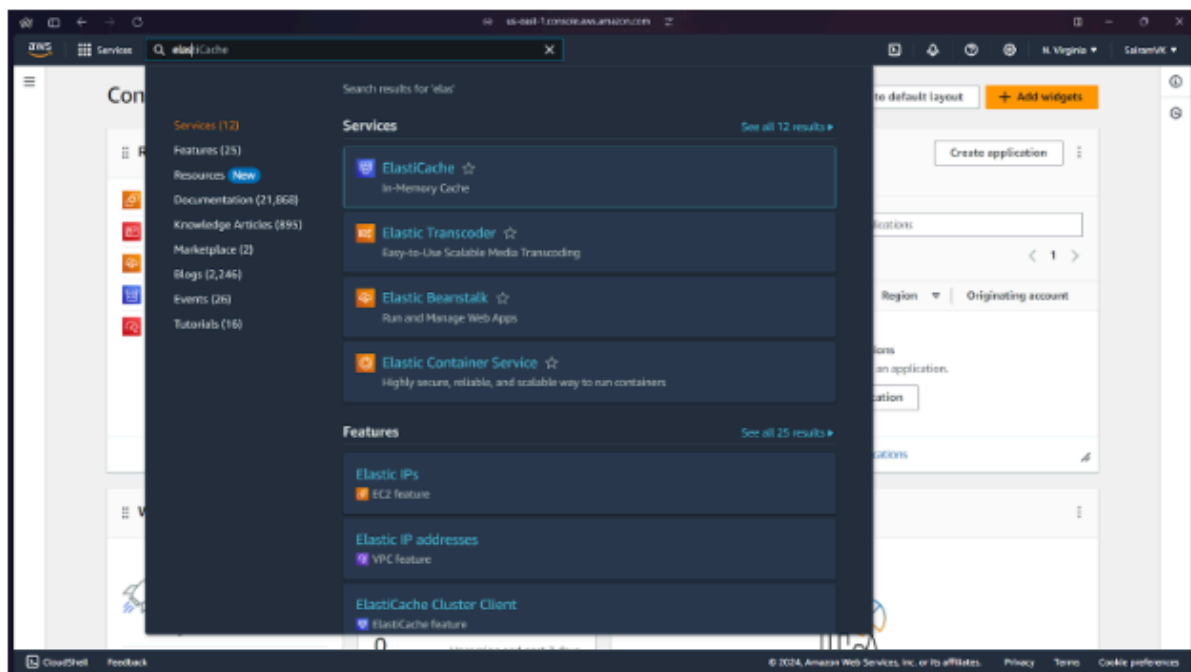
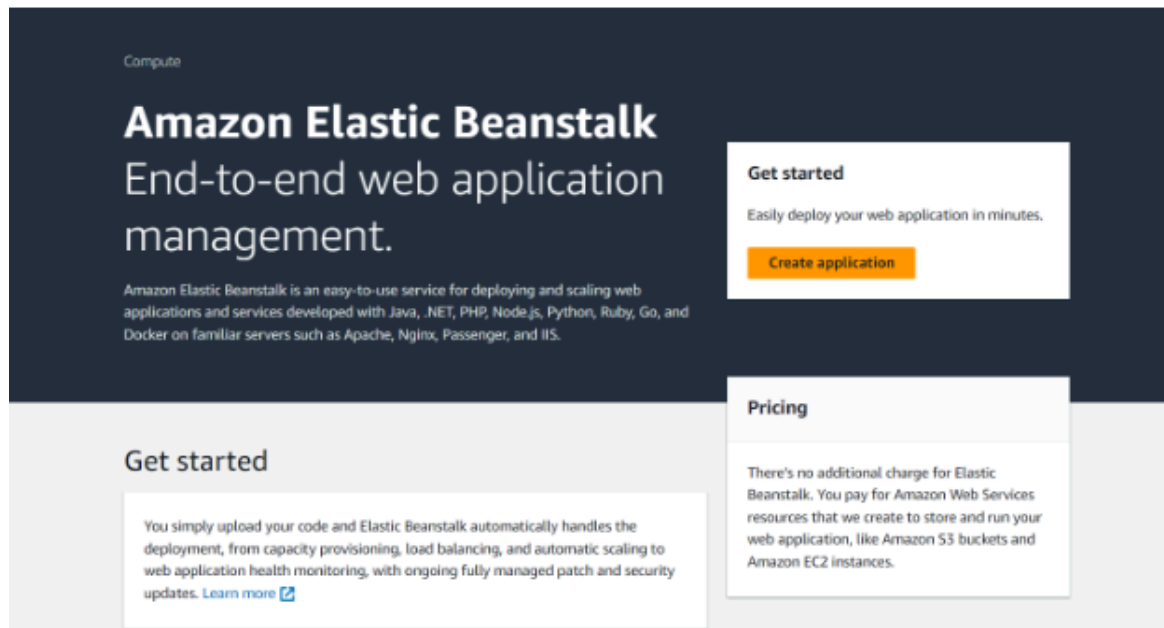


EXPERIMENT 2: To build your application using AWS codebuild and deploy on S3 / SEBS using AWS codepipeline ,deploy sample application on EC2 instance using AWS codedeploy.

STEP 1: Login to your AWS console. Search for Elastic Beanstalk in the searchbar near services.



STEP 2: Click on create application in beanstalk.



The image shows the Amazon Elastic Beanstalk landing page. At the top, it says 'Compute' and 'Amazon Elastic Beanstalk End-to-end web application management.' Below this, there's a paragraph describing the service. To the right, there's a 'Get started' section with a 'Create application' button. Below the main heading, there's a 'Get started' section with a paragraph about uploading code and a 'Learn more' link. To the right of this, there's a 'Pricing' section with a paragraph about charges.

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 application](#)

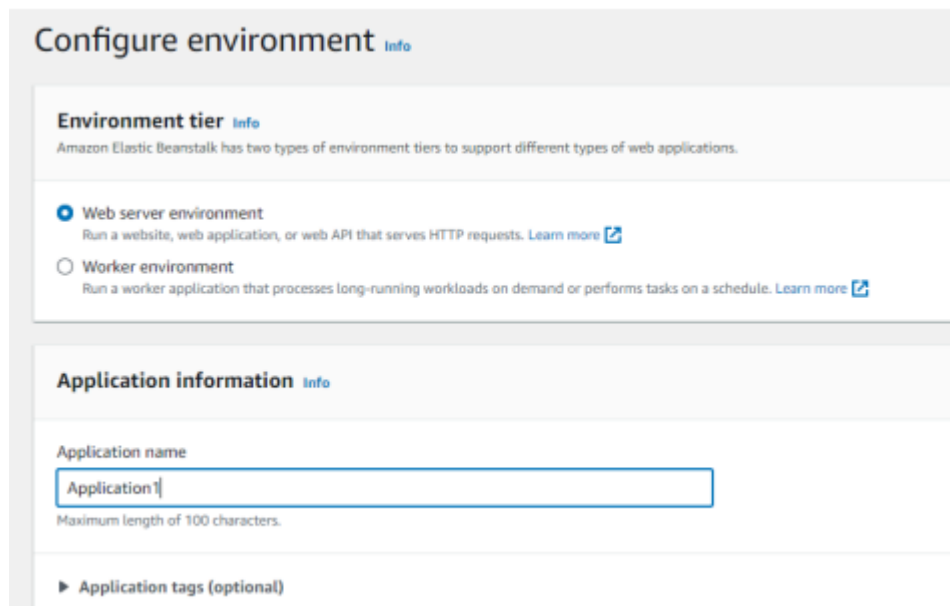
Get started

You simply upload your code and Elastic Beanstalk automatically handles the deployment, from capacity provisioning, load balancing, and automatic scaling to web application health monitoring, with ongoing fully managed patch and security updates. [Learn more](#)

Pricing

There's no additional charge for Elastic Beanstalk. You pay for Amazon Web Services resources that we create to store and run your web application, like Amazon S3 buckets and Amazon EC2 instances.

STEP 3: For creating environment click on web service environment and give application name.



The image shows the 'Configure environment' form in the AWS console. It has two main sections: 'Environment tier' and 'Application information'. The 'Environment tier' section has two radio buttons: 'Web server environment' (selected) and 'Worker environment'. The 'Application information' section has a text input field for 'Application name' with the value 'Application1' and a note about the maximum length of 100 characters. There is also a section for 'Application tags (optional)'.

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)

STEP 4: Click on service access and click on use an existing service role .

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.

AWSCloud9SSMAccessRole

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.

AWSCloud9SSMInstanceProfile

View permission details

STEP 5: From VPC there is a drop up list and from it click on vpc-0da5a3d9b481e2d7b(172.31.0.0/16)

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-0da5a3d9b481e2d7b | (172.31.0.0/16)

[Create custom VPC](#)

STEP 6: Click on review to review the content.

Review [Info](#)

Step 1: Configure environment

Edit

Environment information

Environment tier	Application name
Web server environment	Application1
Environment name	Application code
Application1-env	Sample application
Platform	
arn:aws:elasticbeanstalk:us-east-1::platform/PHP 8.3 running on 64bit Amazon Linux 2023/4.3.1	

STEP 7: Click on configure service access.

Step 2: Configure service access

Edit

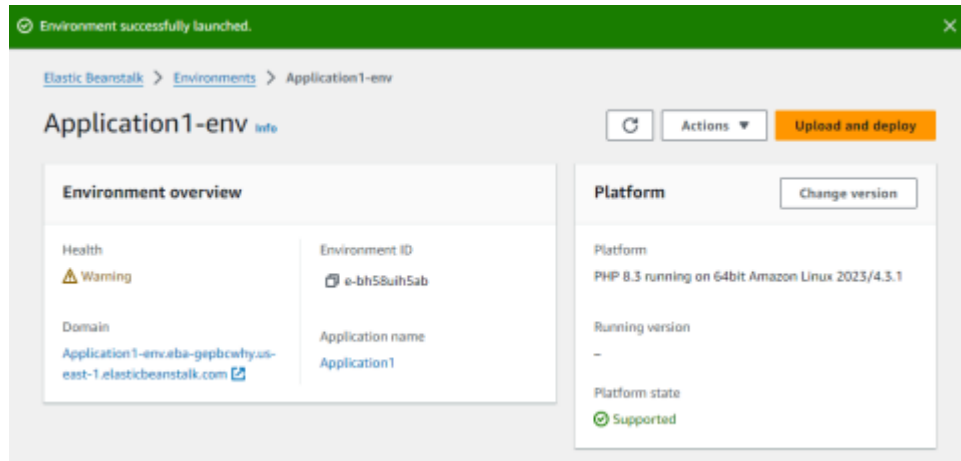
Service access [Info](#)

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::011528263337:role/service-role/AWSCloud9SSMAccessRole	AWSCloud9SSMInstanceProfile

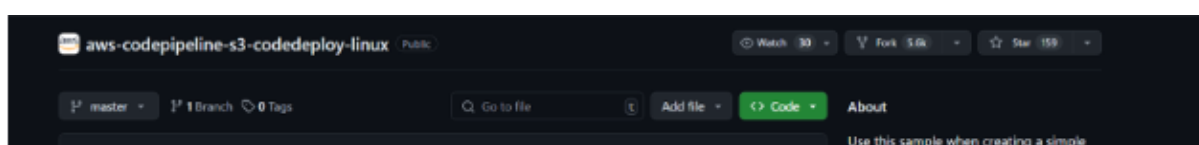
Ignore health check	Instance replacement	
false	false	
Platform software		
Lifecycle	Log streaming	Allow URL fopen
false	Deactivated	On
Display errors	Document root	Max execution time
Off	-	60
Memory limit	Zlib output compression	Proxy server
256M	Off	nginx
Logs retention	Rotate logs	Update level
7	Deactivated	minor
X-Ray enabled		
Deactivated		
Environment properties		

STEP 8:Successfully the environment is created.

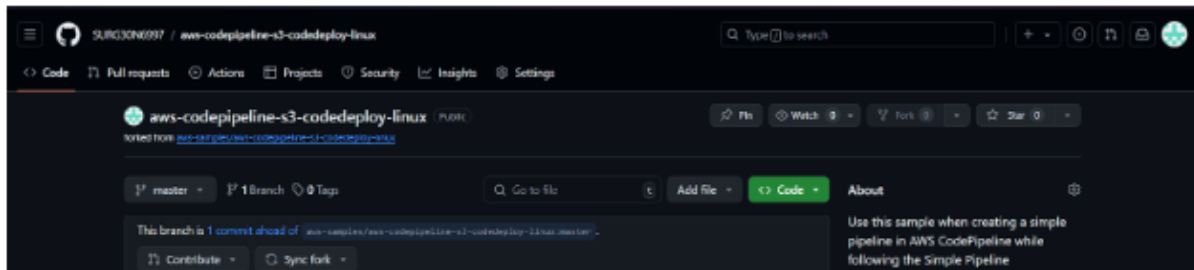


STEP 9:Go on github account and search by the link

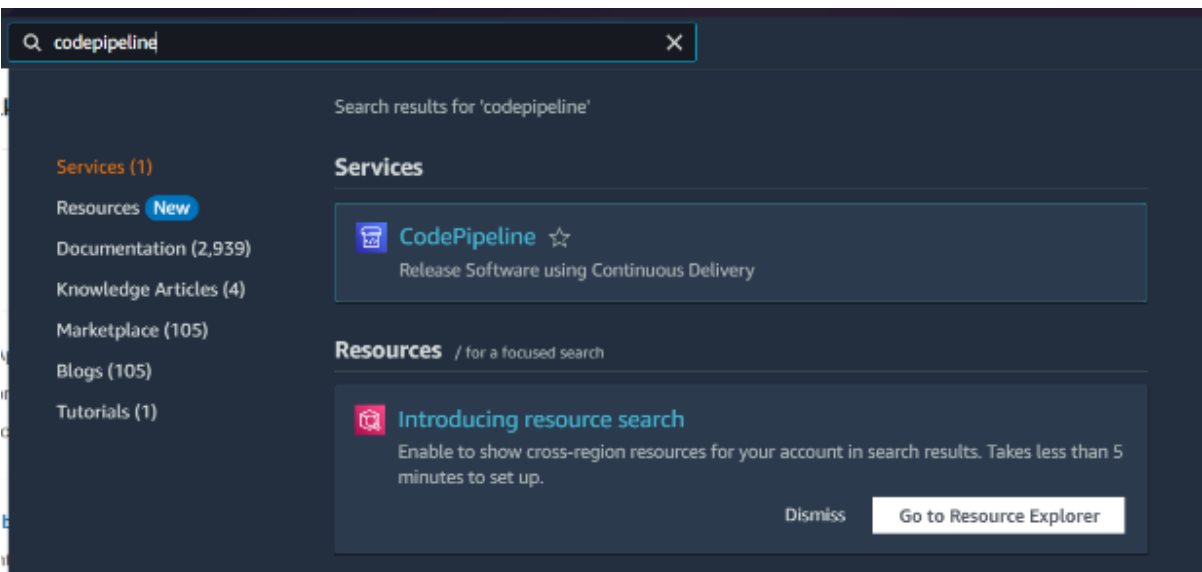
<https://github.com/aws-samples/aws-codepipeline-s3-codedeploy-linux>



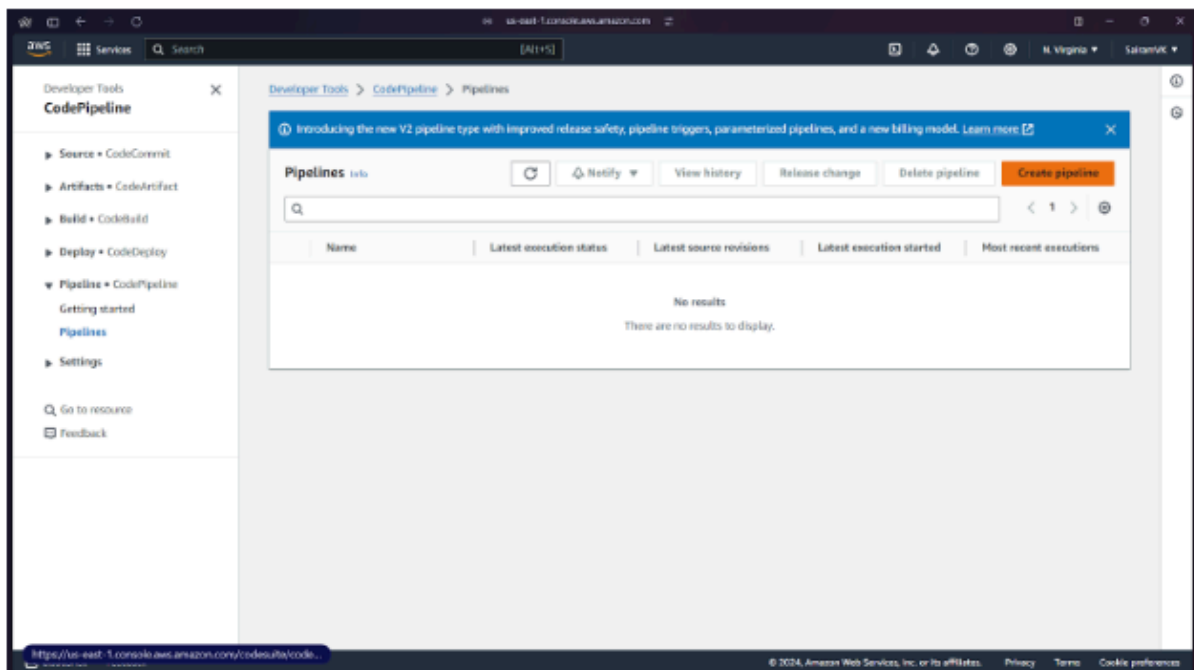
STEP 10: Go on code and beside that there is button called Fork click on it which will create a copy of the code.



STEP 11: Go on AWS academy and search codepipeline on it and click on it.



STEP 12: After clicking on codepipeline an interface will display on that interface there is a button called create pipeline click on that button.



STEP 13: Enter pipeline name in execution mode select queued and in service role select new service role for rolename there will already be an default name .

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.

No more than 100 characters

Pipeline type

i 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)

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

Type your service role name

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

STEP 14: Click on variables and if you want to add variable click on add variable.

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.

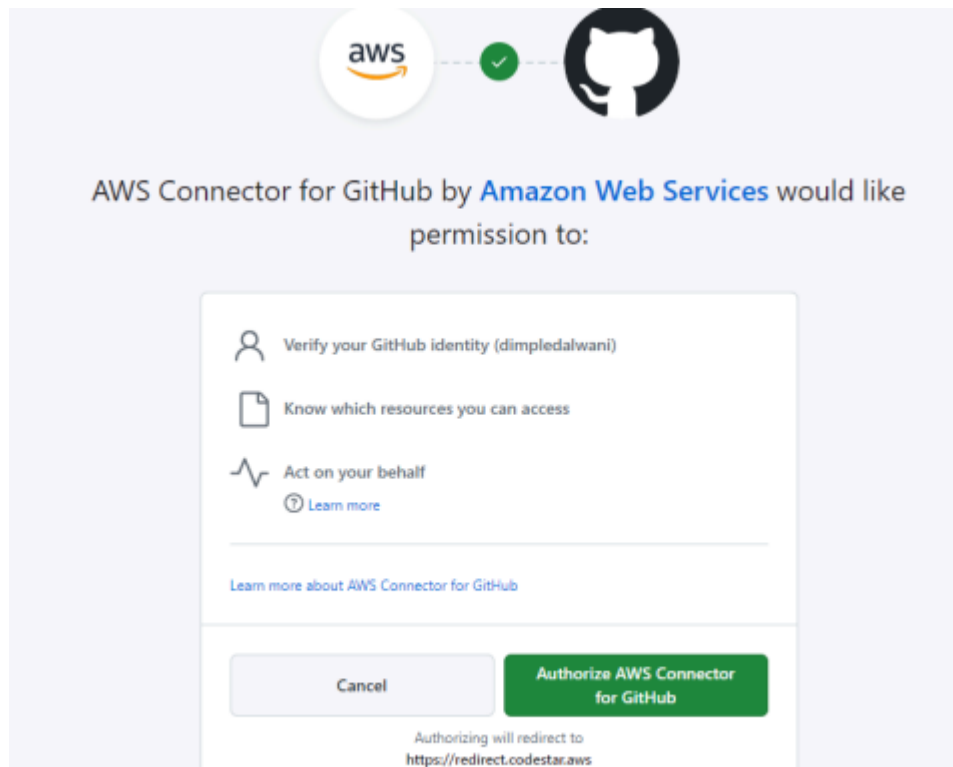
STEP 15: Go on add storage stage and in connection there is a button next to it which is Connect to Github .

The screenshot shows the 'Add source stage' page in the AWS CodePipeline console. The breadcrumb trail is 'Developer Tools > CodePipeline > Pipelines > Create new pipeline'. The left sidebar shows the steps: Step 1 (Choose pipeline settings), Step 2 (Add source stage), Step 3 (Add build stage), Step 4 (Add deploy stage), and Step 5 (Review). The main content area is titled 'Add source stage' and 'Step 2 of 5'. It features a 'Source' section with a 'Source provider' dropdown set to 'GitHub (Version 2)'. Below this is a blue box titled 'New GitHub version 2 (app-based) action' with instructions to create a connection using GitHub Apps. The 'Connection' section has a search box and a 'Connect to GitHub' button. The 'Repository name' section has a search box and a note about the format 'group/subgroup/project'. The 'Default branch' section has a search box and a note about its usage.

STEP 16: In create a connection enter connection name and click on connect to github button.

The screenshot shows the 'Create a connection' page in the AWS CodePipeline console. The breadcrumb trail is 'Developer Tools > Connections > Create connection'. The main content area is titled 'Create a connection'. It features a 'Create GitHub App connection' section with a 'Connection name' input field containing 'MyGithub'. Below this is a section for 'Tags - optional'. At the bottom right, there is an orange 'Connect to GitHub' button.

STEP 17: It will direct you to this page and click on Authorise AWS Connector for Github button.



STEP 18: Select all repositories option and click on install button.

Install AWS Connector for GitHub

Install on your personal account dimpledalwani

for these repositories:

☒ **All repositories**
This applies to all current *and* future repositories owned by the resource owner.
Also includes public repositories (read-only).

☐ **Only select repositories**
Select at least one repository.
Also includes public repositories (read-only).

with these permissions:

✓ **Read** access to issues and metadata

✓ **Read and write** access to administration, code, commit statuses, pull requests, and repository hooks


Install

Cancel

Next: you'll be directed to the GitHub App's site to complete setup.

STEP 19: Enter your repository name and set default branch as master and select codepipeline default.

am:aws:codeconnections:us-east-1:011528263357:connection/be7ab482-55 X or [Connect to GitHub](#)

 **Ready to connect**
Your GitHub connection is ready for use.

Repository name
Choose a repository in your GitHub account.

Q dimpledalwani/aws-codepipeline-s3-codedeploy-linux X

You can type or paste the group path to any project that the provided credentials can access. Use the format 'group/subgroup/project'.

Default branch
Default branch will be used only when pipeline execution starts from a different source or manually started.

Q master X

master

Choose the output artifact format.

☒ **CodePipeline default**
AWS CodePipeline uses the default zip format for artifacts in the pipeline. Does not include Git metadata about the repository.

☐ **Full clone**
AWS CodePipeline passes metadata about the repository that allows subsequent actions to do a full Git clone. Only supported for AWS CodeBuild actions.

STEP 20: Since building stage is option you can directly skip it.

Must select a project

Skip build stage X

Your pipeline will not include a build stage. Are you sure you want to skip this stage?

Cancel **Skip**

STEP 21: In Deploy enter your application name and environment name .

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](#)

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.

Application1

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.

Application1-env

STEP 22: After Which the pipeline successfully get created .

Success

Create a notification rule for this pipeline

Congratulations! The pipeline pipeline1 has been created.

Developer Tools > CodePipeline > Pipelines > pipeline1

pipeline1

Notify Edit Stop execution Clone pipeline Release change

Pipeline type: V2 Execution mode: QUEUED

Source Succeeded

Pipeline execution ID: [4b5e5556-41cc-486c-82b2-75d4f5ba2e15](#)

Source

[Github Version 21](#)

Succeeded - [Just now](#)

[8be52cha](#)

View details

STEP 23:REVIEW THE DEPLOY .

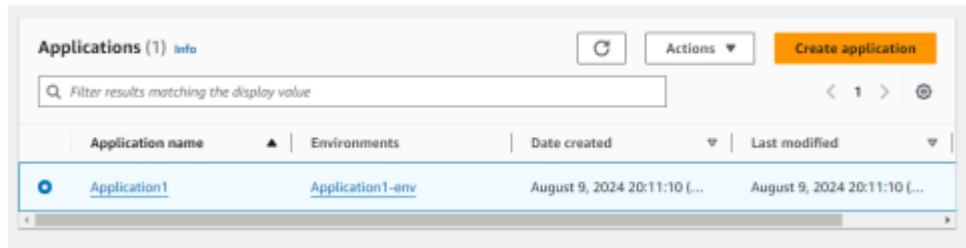
The screenshot shows the AWS CodePipeline console for a pipeline named 'pipeline1'. The 'Deploy' stage is highlighted with a blue bar on the left. The status is 'In progress' with a yellow warning icon. The pipeline execution ID is 'db5e3336-41cc-486c-82b2-23dbf5ba2a13'. Below this, a box shows the 'Deploy' stage using 'AWS Elastic Beanstalk' as the provider, with a status of 'In progress - Just now' and a 'View details' button. At the bottom, the 'Source' stage is shown with the provider '8be52cba' and the action 'Adding template'.

STEP 24:After few seconds the pipeline gets deployed successfully.

This screenshot shows the 'Source' stage of the pipeline. The status is 'Succeeded' with a green checkmark icon. The pipeline execution ID remains 'db5e3336-41cc-486c-82b2-23dbf5ba2a13'. The 'Source' stage box shows it used 'GitHub (Version 2)' as the provider, 'Succeeded - 1 minute ago', and the provider '8be52cba'. A 'View details' button is present. The bottom of the console shows the 'Source' stage with the provider '8be52cba' and the action 'Adding template'.

This screenshot shows the 'Deploy' stage of the pipeline. The status is 'Succeeded' with a green checkmark icon. The pipeline execution ID is 'db5e3336-41cc-486c-82b2-23dbf5ba2a13'. The 'Deploy' stage box shows it used 'AWS Elastic Beanstalk' as the provider, 'Succeeded - Just now', and a 'View details' button. A 'Start rollback' button is located to the right of the stage box. The bottom of the console shows the 'Source' stage with the provider '8be52cba' and the action 'Adding template'.

STEP 25: Go on Applications and select the application you have created and click on the link which is there in the environments.



STEP 26: Pipeline gets created successfully.

