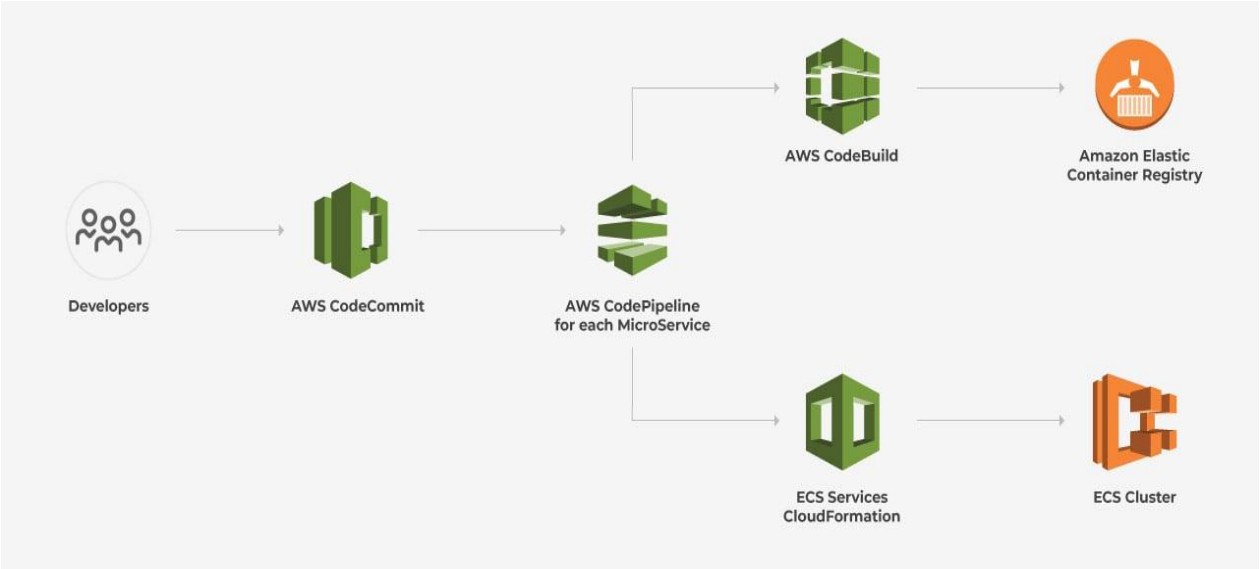
Title

CI/CD Pipeline Implementation Using AWS Services for Python Application Deployment



INDEX

1. Introduction

This document describes how to create a CI/CD (Continuous Integration and Continuous Deployment) pipeline using AWS services. CI/CD pipelines automate the process of building, testing, and deploying software, allowing developers to quickly and safely release changes.

In this task, we built an end-to-end pipeline for deploying a Python application. The pipeline automatically pulls code from GitHub, builds a Docker image, pushes it to Amazon Elastic Container Registry (ECR), and deploys it to an EC2 instance using CodeDeploy. All of this was orchestrated using AWS CodePipeline.

This guide is suitable for beginners and provides step-by-step instructions to set up this pipeline using various AWS tools.

2. Technologies and Tools Overview

- **GitHub**

  GitHub is a platform for version control that hosts code repositories. Developers use GitHub to store their project code and collaborate with others. In this case, we used GitHub to store the Python application source code.

- **AWS CodeBuild**

  AWS CodeBuild is a fully managed build service that automates the process of compiling source code, running tests, and generating software packages. It simplifies the process of building Docker images for applications. In our case, CodeBuild was used to build a Docker image of the Python application and push it to Amazon ECR.

- **AWS CodeDeploy**

  AWS CodeDeploy is a deployment service that automates the process of delivering applications to compute instances like EC2 or Lambda. Here, it is used to deploy the Docker image from ECR to an EC2 instance, ensuring the latest version of the application is running.

- **AWS CodePipeline**

  AWS CodePipeline is a CI/CD service that automates the workflow for continuous integration and delivery. It connects various AWS services (like GitHub, CodeBuild, and CodeDeploy) into a sequence that automatically triggers the next action, like building the application or deploying it, after each change in the code repository.

- **AWS EC2 and ECR**

  EC2 (Elastic Compute Cloud): EC2 is a service that provides virtual machines (instances) on which you can run applications. We used an EC2 instance to run the Python application.
  ECR (Elastic Container Registry): ECR is a managed Docker image storage service. We used ECR to store Docker images built by CodeBuild.

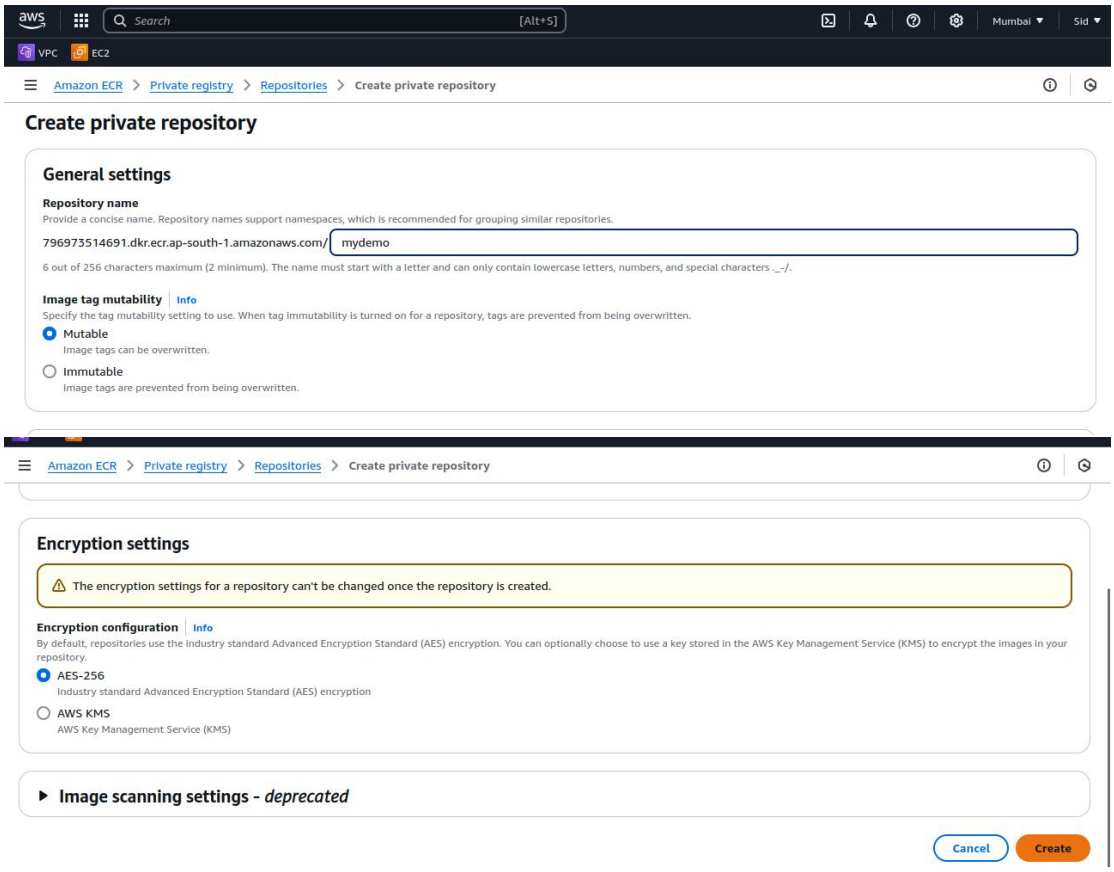
3. Step-by-Step Implementation

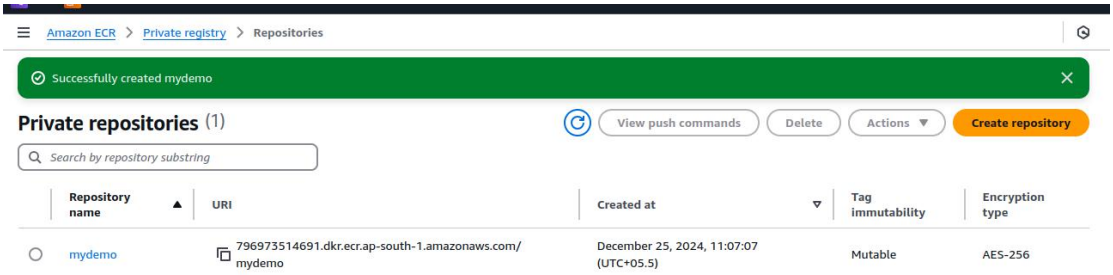
1] Create Repository in Amazon ECR

What is Amazon ECR? Amazon ECR is where you can store your Docker images. When you build an application in Docker, you can push that Docker image to ECR so it can be accessed by other services like CodeDeploy.

How to do it:

1. Go to the **Amazon ECR Console.**
2. Click **Create repository.**
3. Name the repository (e.g., mydemo).
4. Use the default settings for the repository and create it.



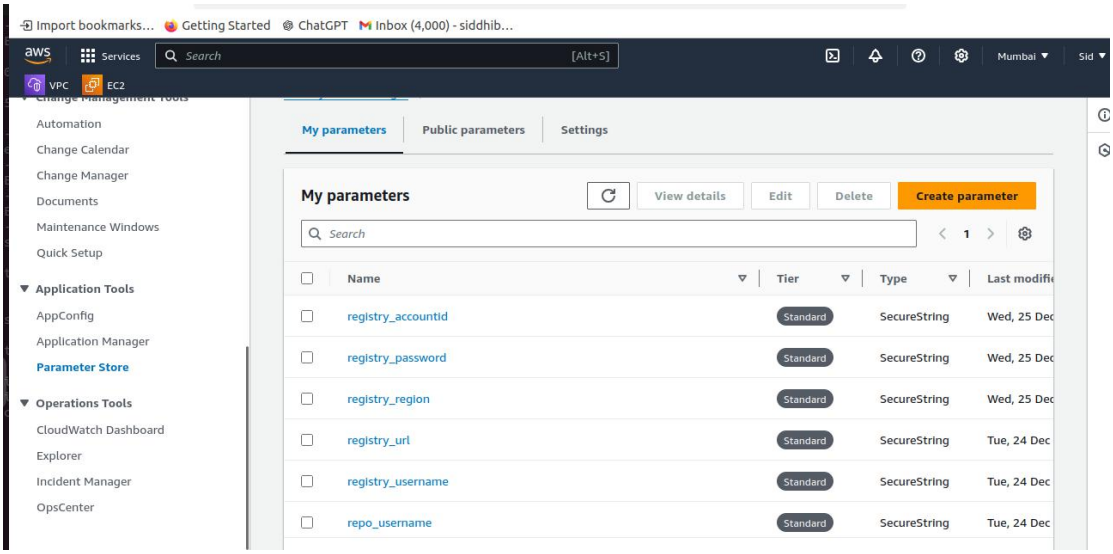


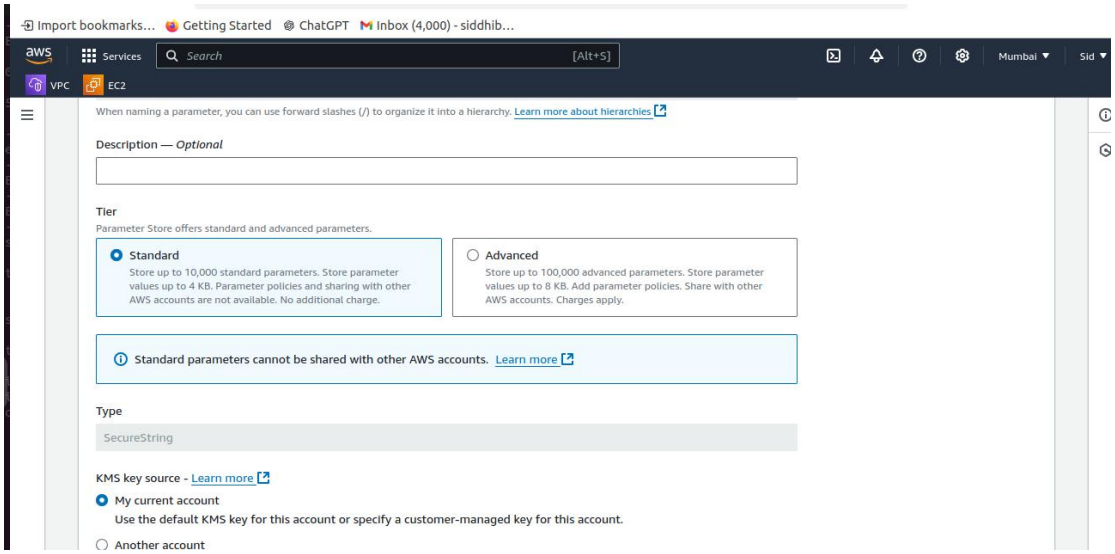
2] Store Parameters in AWS Systems Manager

What is AWS Systems Manager Parameter Store? AWS Systems Manager Parameter Store is a service to store configuration data, such as credentials, securely. Here, we store sensitive information like Docker registry credentials, so the build process can access them securely.

- **How to do it:**

1. Go to **AWS Systems Manager**.
2. Choose **Parameter Store** and click **Create parameter**.
3. Store parameters like registry_url, registry_username, and registry_password as **SecureString** to keep them encrypted.





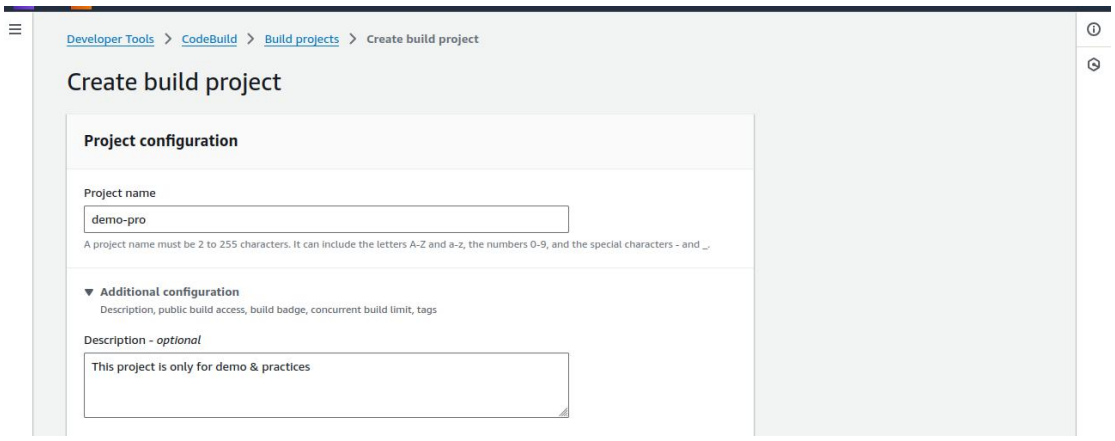
3] Set Up AWS CodeBuild Project

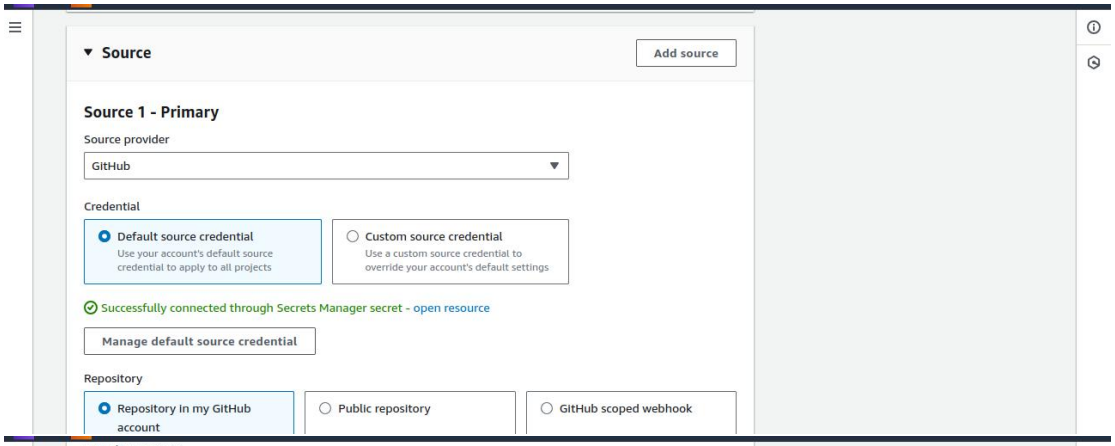
What is AWS CodeBuild? AWS CodeBuild is used to automatically build and test your application whenever there's a code change.

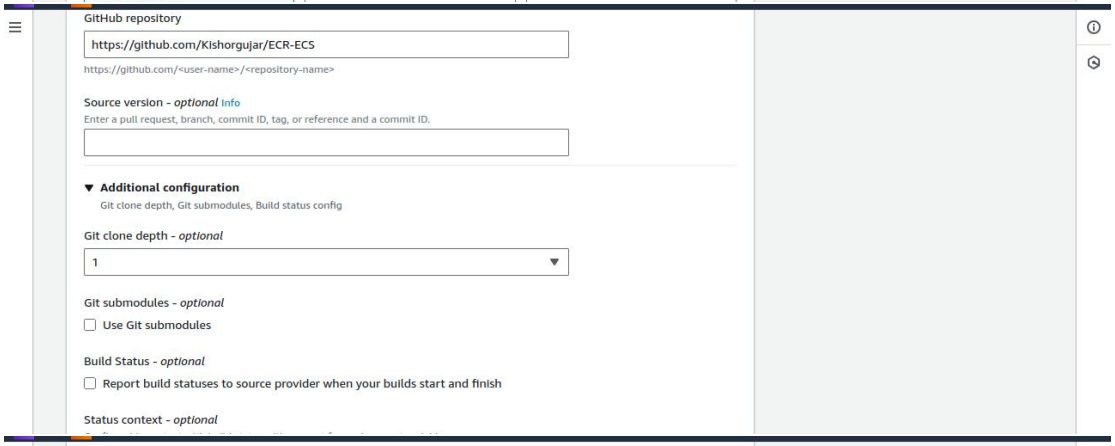
· **How to do it:**

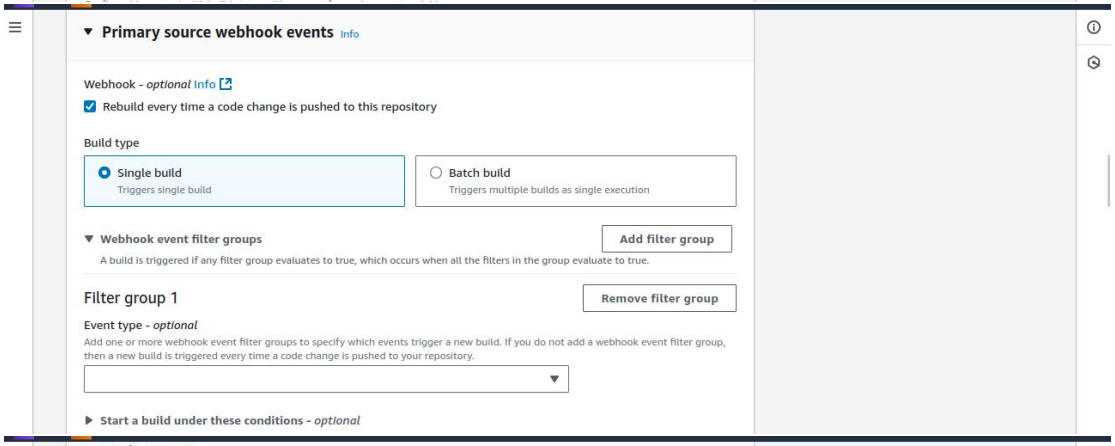
1. Go to **AWS CodeBuild Console** and create a new project.
2. Choose **GitHub** as the source provider, so CodeBuild pulls code from your GitHub repository.
3. Select **Ubuntu** as the operating system.
4. In the buildspec file (a file that tells CodeBuild how to build your application), use commands to:
    1. Install dependencies for your Python application.
    2. Log in to the Docker registry (ECR).
    3. Build and push the Docker image to ECR.

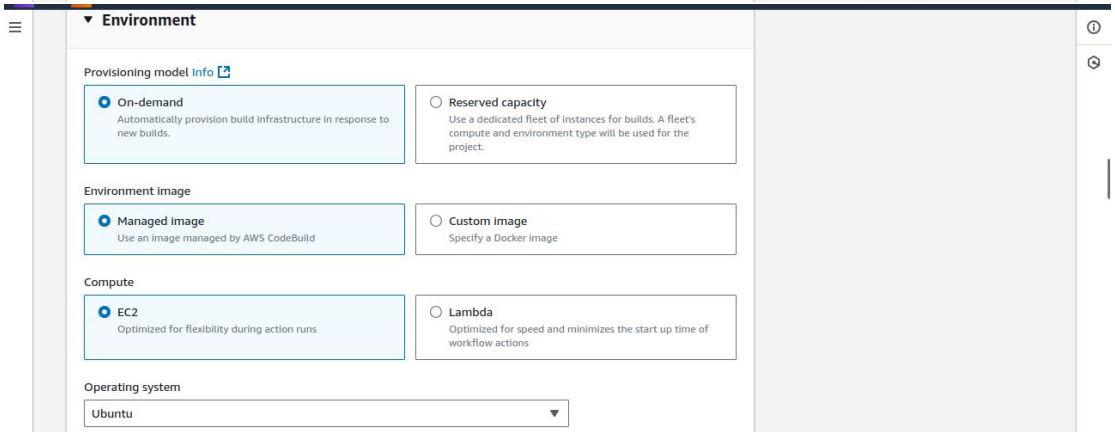
Example buildspec.yml: https://github.com/Kishorgujar/ECR-ECS/blob/master/simple-python-app/buildspec.yml

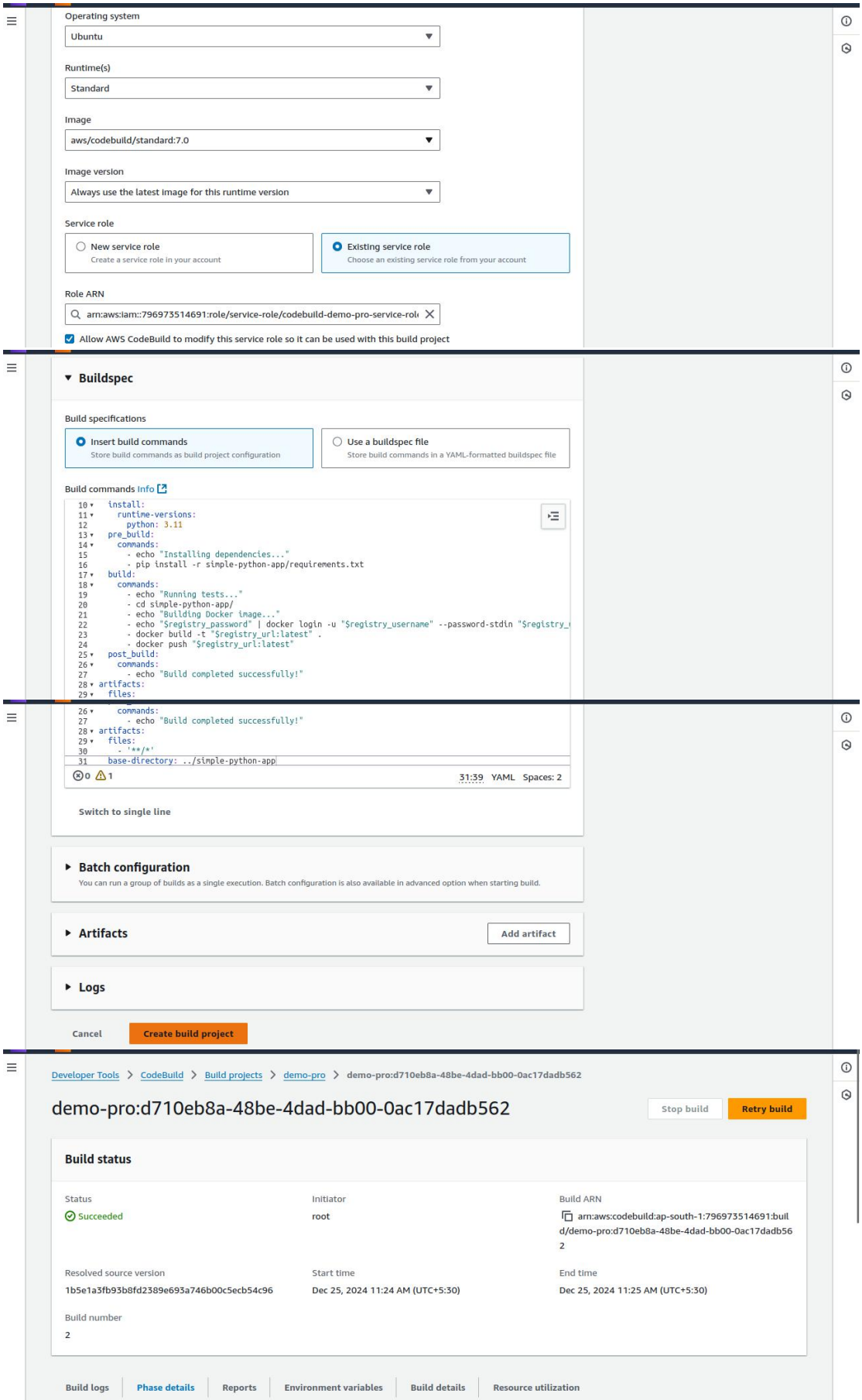



Source
Add source
Source 1 - Primary
Source provider
GitHub
Credential
Default source credential
Use your account's default source credential to apply to all projects
Custom source credential
Use a custom source credential to override your account's default settings
Successfully connected through Secrets Manager secret - open resource
Manage default source credential
Repository
Repository in my GitHub account
Public repository
GitHub scoped webhook


GitHub repository
https://github.com/Kishorgujar/ECR-ECS
https://github.com/<user-name>/<repository-name>
Source version - optional Info
Enter a pull request, branch, commit ID, tag, or reference and a commit ID.
Additional configuration
Git clone depth, Git submodules, Build status config
Git clone depth - optional
1
Git submodules - optional
Use Git submodules
Build Status - optional
Report build statuses to source provider when your builds start and finish
Status context - optional


Primary source webhook events Info
Webhook - optional Info
Rebuild every time a code change is pushed to this repository
Build type
Single build
Triggers single build
Batch build
Triggers multiple builds as single execution
Webhook event filter groups
Add filter group
A build is triggered if any filter group evaluates to true, which occurs when all the filters in the group evaluate to true.
Filter group 1
Remove filter group
Event type - optional
Add one or more webhook event filter groups to specify which events trigger a new build. If you do not add a webhook event filter group, then a new build is triggered every time a code change is pushed to your repository.
Start a build under these conditions - optional


Environment
Provisioning model Info
On-demand
Automatically provision build infrastructure in response to new builds.
Reserved capacity
Use a dedicated fleet of instances for builds. A fleet's compute and environment type will be used for the project.
Environment image
Managed image
Use an image managed by AWS CodeBuild
Custom image
Specify a Docker image
Compute
EC2
Optimized for flexibility during action runs
Lambda
Optimized for speed and minimizes the start up time of workflow actions
Operating system
Ubuntu

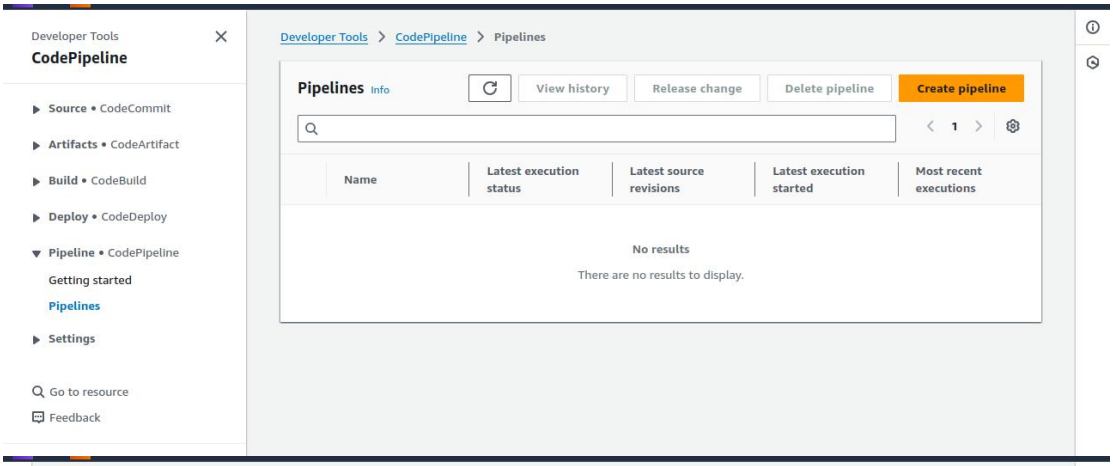

Operating system
Ubuntu
Runtime(s)
Standard
Image
aws/codebuild/standard:7.0
Image version
Always use the latest image for this runtime version
Service role
New service role
Create a service role in your account
Existing service role
Choose an existing service role from your account
Role ARN
arn:aws:iam::796973514691:role/service-role/codebuild-demo-pro-service-rol
Allow AWS CodeBuild to modify this service role so it can be used with this build project
Buildspec
Build specifications
Insert build commands
Store build commands as build project configuration
Use a buildspec file
Store build commands in a YAML-formatted buildspec file
Build commands Info
install:
runtime-versions:
python: 3.11
pre_build:
commands:
- echo "Installing dependencies..."
- pip install -r simple-python-app/requirements.txt
build:
commands:
- echo "Running tests..."
- cd simple-python-app/
- echo "Building Docker image..."
- docker build -t "$registry_url:latest" .
- docker push "$registry_url:latest"
post_build:
commands:
- echo "Build completed successfully!"
artifacts:
files:
- '**/*'
base-directory: ../simple-python-app
31:39 YAML Spaces: 2
Switch to single line
Batch configuration
You can run a group of builds as a single execution. Batch configuration is also available in advanced option when starting build.
Artifacts
Add artifact
Logs
Cancel
Create build project
Developer Tools > CodeBuild > Build projects > demo-pro > demo-pro:d710eb8a-48be-4dad-bb00-0ac17dadb562
demo-pro:d710eb8a-48be-4dad-bb00-0ac17dadb562
Stop build
Retry build
Build status
Status
Succeeded
Initiator
root
Build ARN
arn:aws:codebuild:ap-south-1:796973514691:build/demo-pro:d710eb8a-48be-4dad-bb00-0ac17dadb562
Resolved source version
1b5e1a3fb93b8fd2389e693a746b00c5ecb54c96
Start time
Dec 25, 2024 11:24 AM (UTC+5:30)
End time
Dec 25, 2024 11:25 AM (UTC+5:30)
Build number
2
Build logs
Phase details
Reports
Environment variables
Build details
Resource utilization

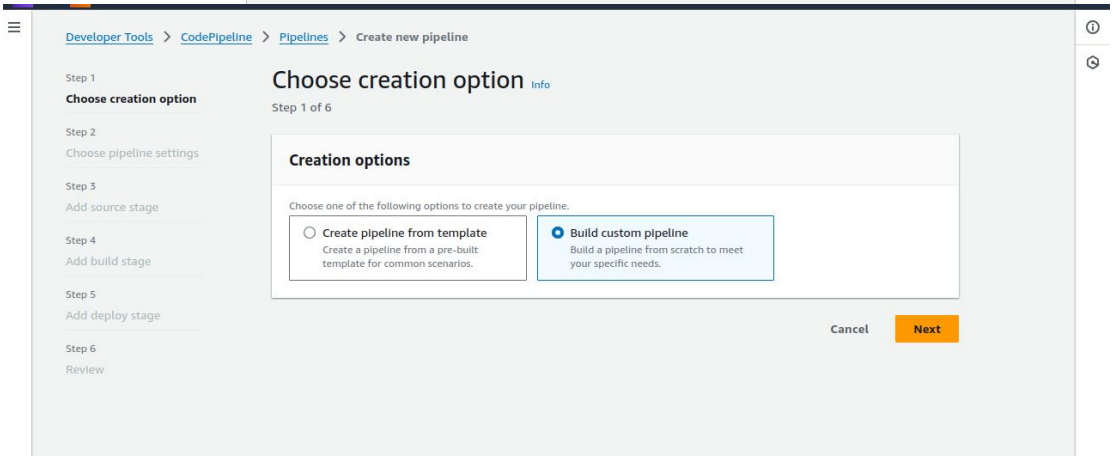
4] Set Up AWS CodePipeline

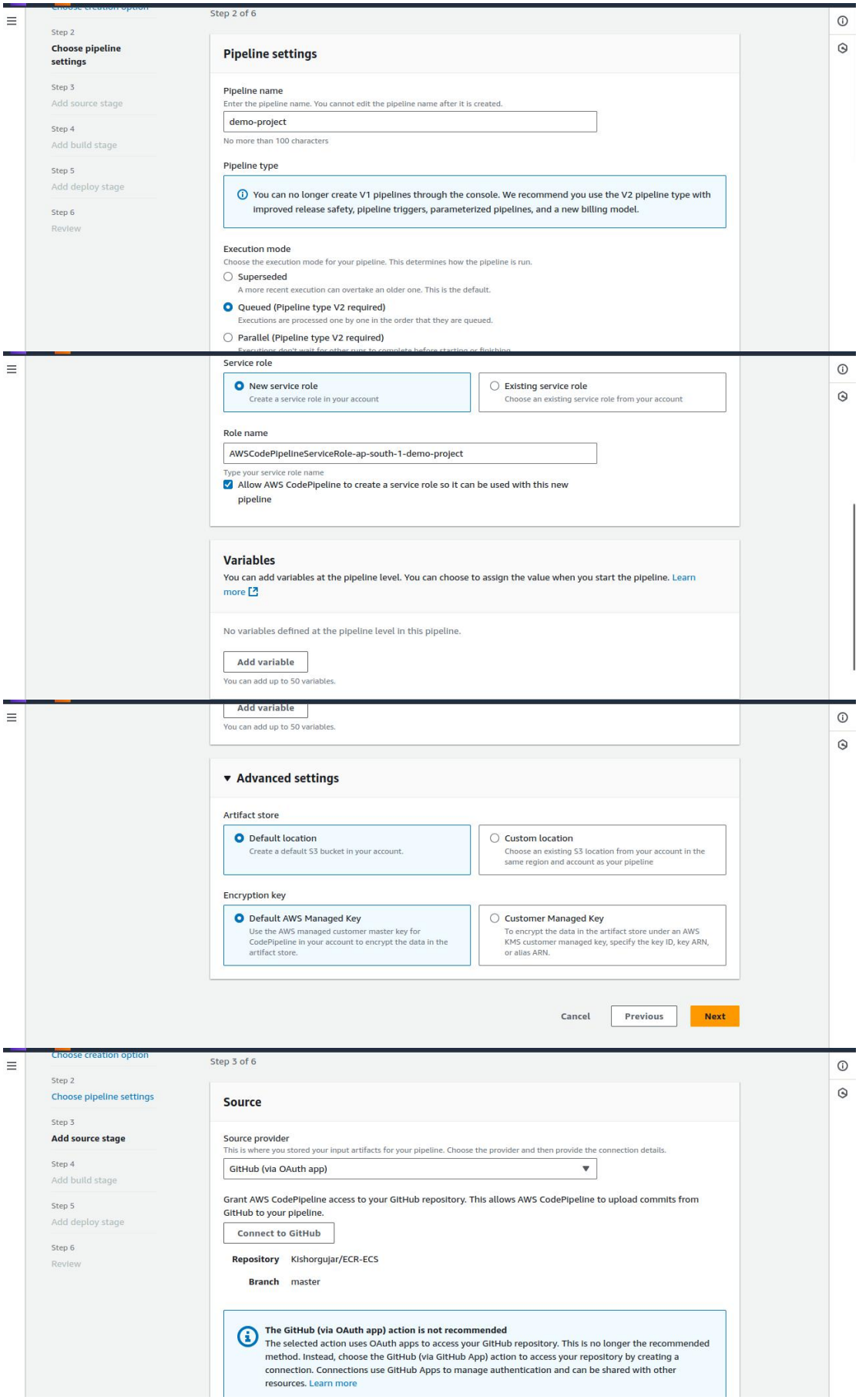
What is AWS CodePipeline? AWS CodePipeline is used to automate the CI/CD workflow. It connects the different steps of your pipeline (like CodeBuild for building and CodeDeploy for deployment).

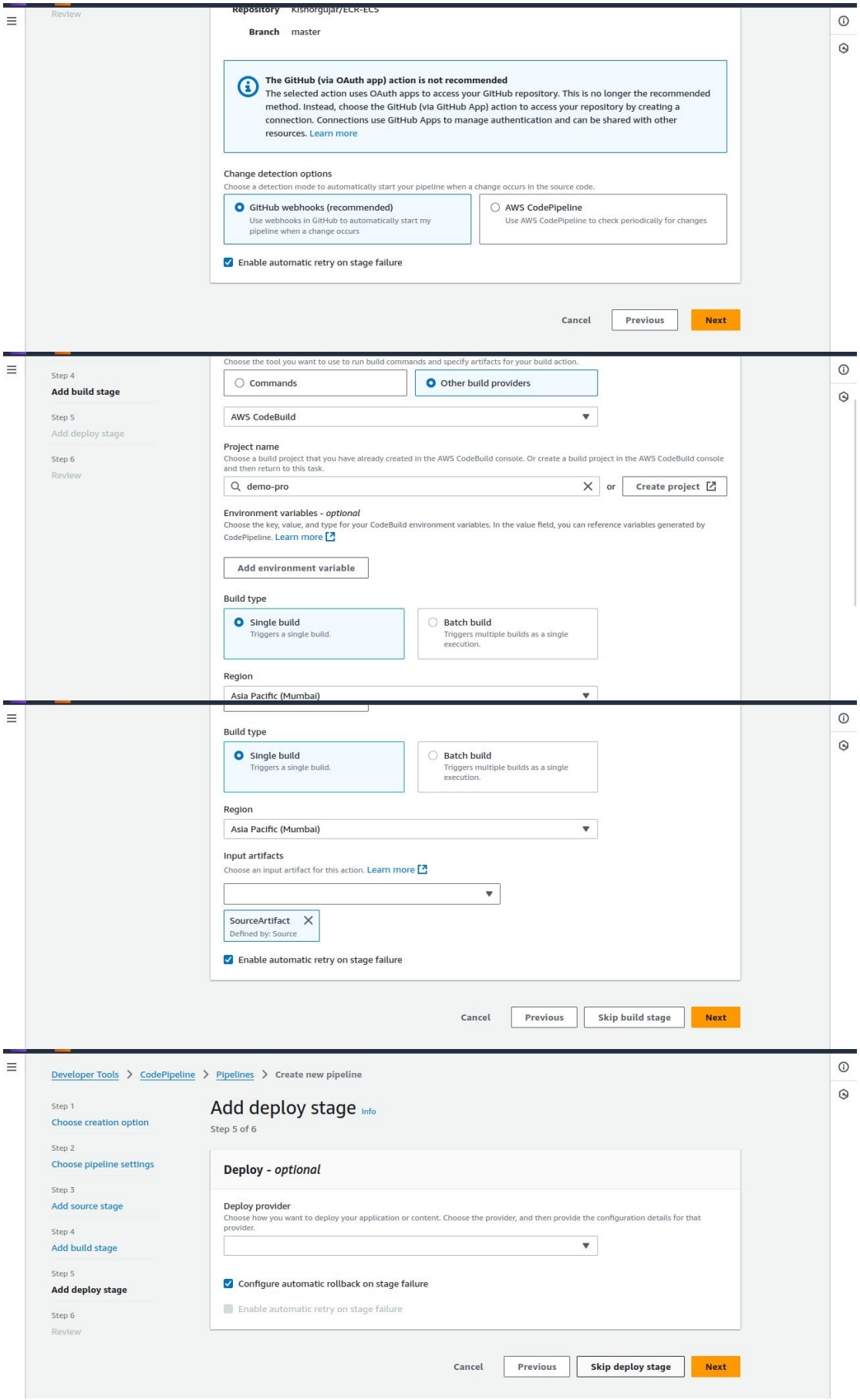
· **How to do it:**

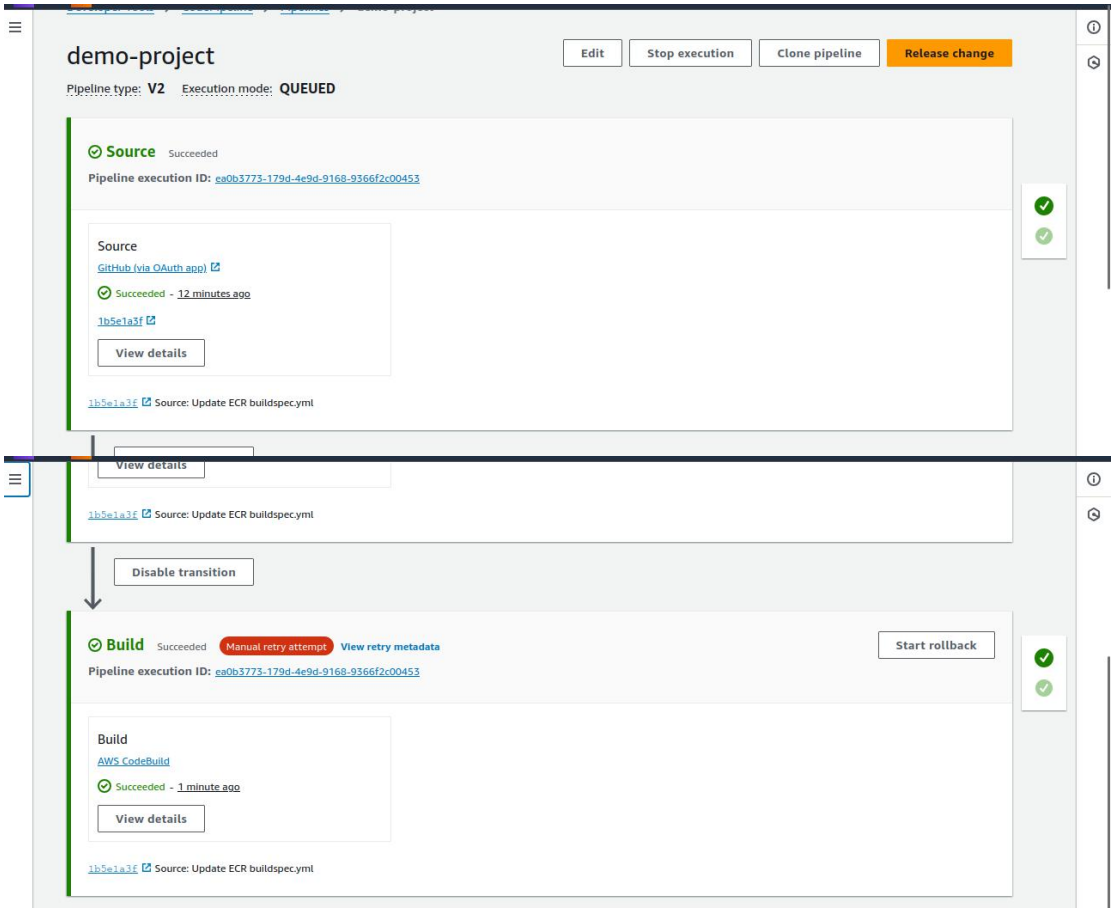
1. Go to **AWS CodePipeline Console**.
2. Create a new pipeline named demo-project.
3. Select **GitHub** as the source.
4. Add **AWS CodeBuild** as the build provider.
5. Skip the deploy stage for now (we will add this later)






Step 2 of 6
Step 2
Choose pipeline settings
Step 3
Add source stage
Step 4
Add build stage
Step 5
Add deploy stage
Step 6
Review
Pipeline settings
Pipeline name
Enter the pipeline name. You cannot edit the pipeline name after it is created.
demo-project
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)
Executions are processed one by one in the order that they are queued.
Parallel (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
AWSCodePipelineServiceRole-ap-south-1-demo-project
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. Learn more
No variables defined at the pipeline level in this pipeline.
Add variable
You can add up to 50 variables.
Add variable
You can add up to 50 variables.
Advanced settings
Artifact store
Default location
Create a default S3 bucket in your account.
Custom location
Choose an existing S3 location from your account in the same region and account as your pipeline
Encryption key
Default AWS Managed Key
Use the AWS managed customer master key for CodePipeline in your account to encrypt the data in the artifact store.
Customer Managed Key
To encrypt the data in the artifact store under an AWS KMS customer managed key, specify the key ID, key ARN, or alias ARN.
Cancel
Previous
Next
Choose creation option
Step 2
Choose pipeline settings
Step 3
Add source stage
Step 4
Add build stage
Step 5
Add deploy stage
Step 6
Review
Step 3 of 6
Source
Source provider
This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.
GitHub (via OAuth app)
Grant AWS CodePipeline access to your GitHub repository. This allows AWS CodePipeline to upload commits from GitHub to your pipeline.
Connect to GitHub
Repository Kishorgujar/ECR-ECS
Branch master
The GitHub (via OAuth app) 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 (via GitHub App) 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


Review
Repository kishorgujar/ECR-ECS
Branch master
The GitHub (via OAuth app) 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 (via GitHub App) 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
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
Enable automatic retry on stage failure
Cancel
Previous
Next
Step 4
Add build stage
Step 5
Add deploy stage
Step 6
Review
Choose the tool you want to use to run build commands and specify artifacts for your build action.
Commands
Other build providers
AWS CodeBuild
Project name
Choose a build project that you have already created in the AWS CodeBuild console. Or create a build project in the AWS CodeBuild console and then return to this task.
demo-pro
or
Create project
Environment variables - optional
Choose the key, value, and type for your CodeBuild environment variables. In the value field, you can reference variables generated by CodePipeline. Learn more
Add environment variable
Build type
Single build
Triggers a single build.
Batch build
Triggers multiple builds as a single execution.
Region
Asia Pacific (Mumbai)
Build type
Single build
Triggers a single build.
Batch build
Triggers multiple builds as a single execution.
Region
Asia Pacific (Mumbai)
Input artifacts
Choose an input artifact for this action. Learn more
SourceArtifact
Defined by: Source
Enable automatic retry on stage failure
Cancel
Previous
Skip build stage
Next
Developer Tools > CodePipeline > Pipelines > Create new pipeline
Step 1
Choose creation option
Step 2
Choose pipeline settings
Step 3
Add source stage
Step 4
Add build stage
Step 5
Add deploy stage
Step 6
Review
Add deploy stage Info
Step 5 of 6
Deploy - optional
Deploy provider
Choose how you want to deploy your application or content. Choose the provider, and then provide the configuration details for that provider.
Configure automatic rollback on stage failure
Enable automatic retry on stage failure
Cancel
Previous
Skip deploy stage
Next

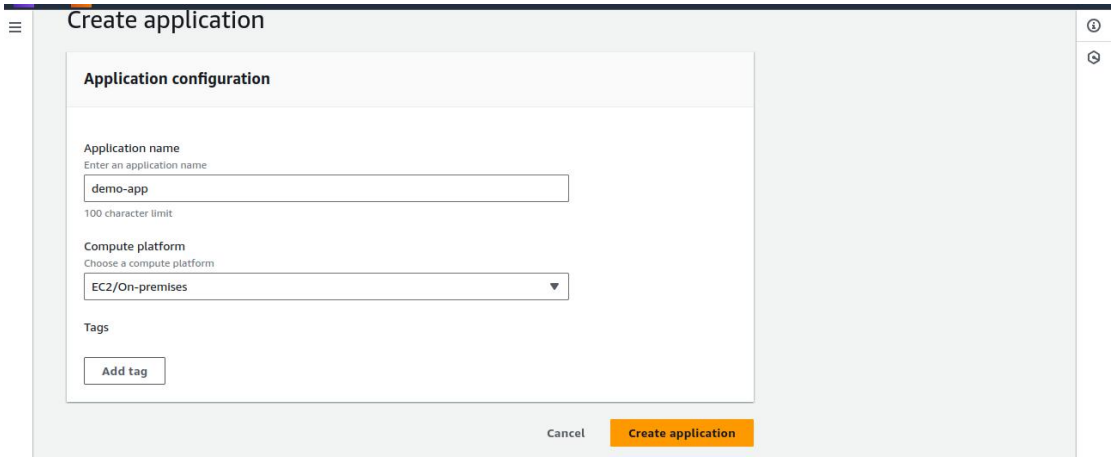


5] Create CodeDeploy Application and Deployment Group

What is CodeDeploy? CodeDeploy is used to deploy your application to EC2 or Lambda. In this case, we deploy the Docker image to an EC2 instance.

· **How to do it:**

1. Go to **AWS CodeDeploy Console**.
2. Create an application (e.g., demo-app).
3. Create a deployment group (e.g., demo-deploy-group) and choose **EC2** as the compute platform.

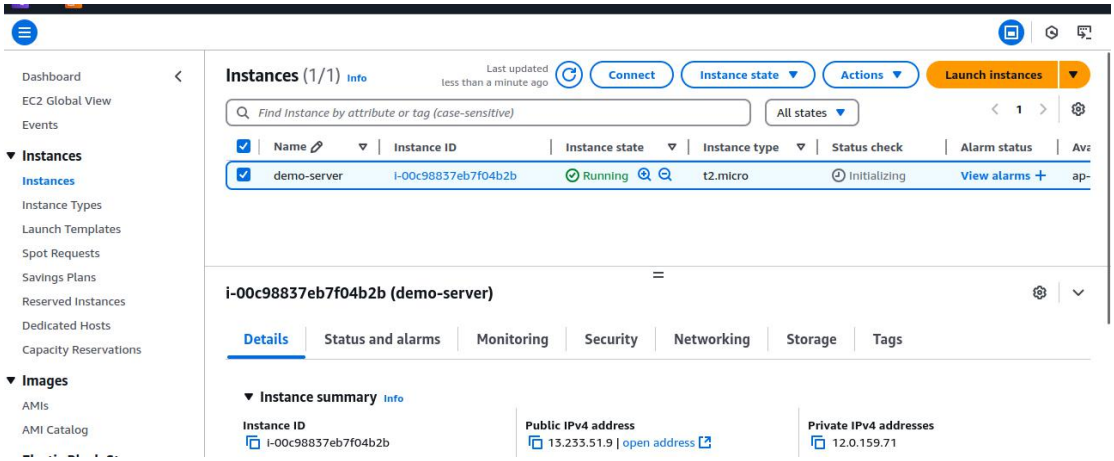


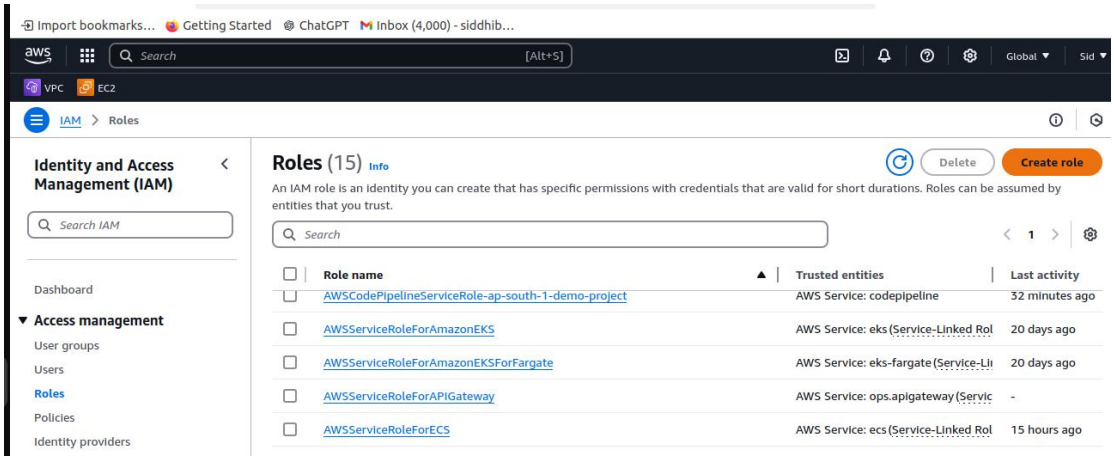
6] Set Up EC2 Instance for Deployment

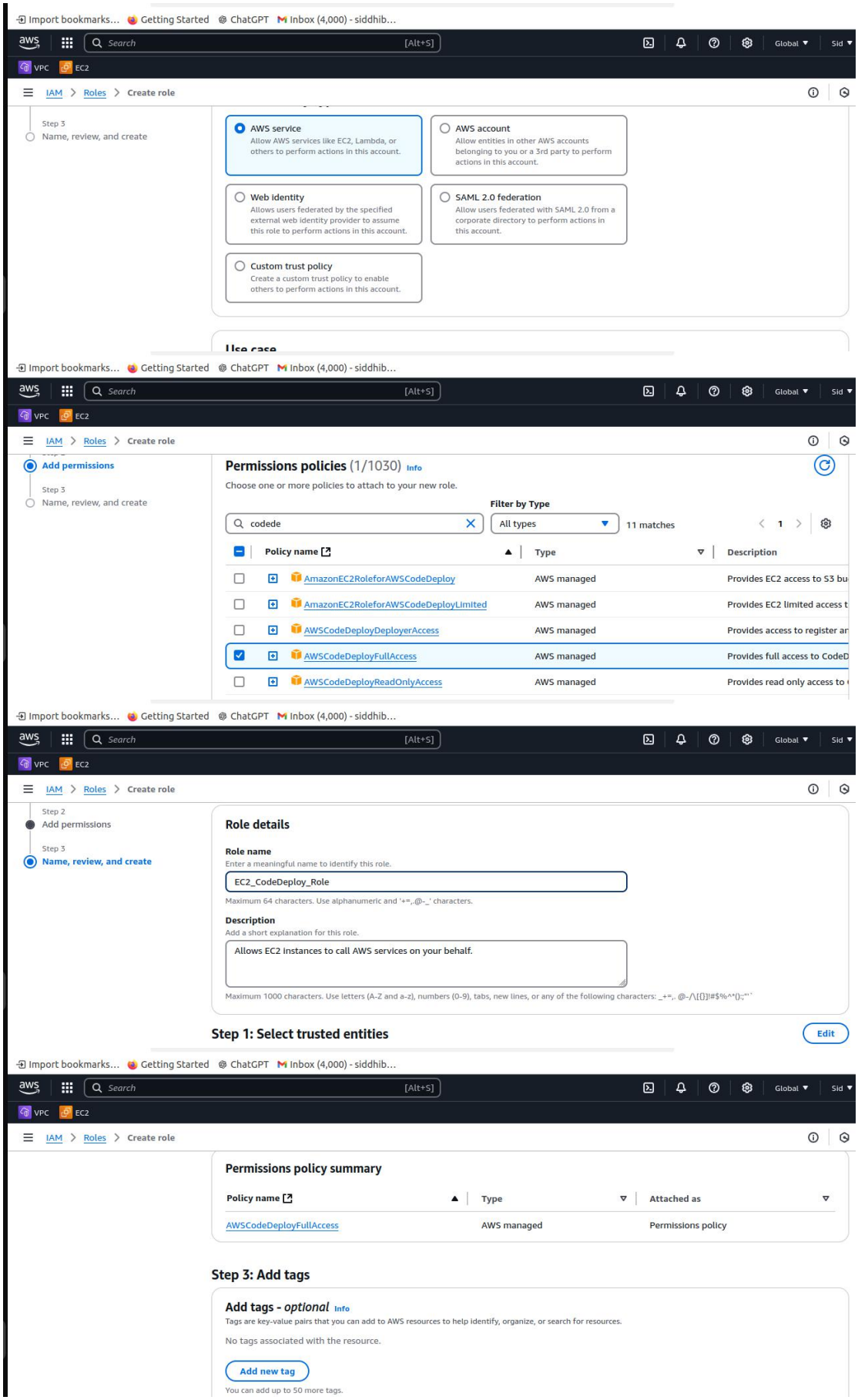
What is EC2? EC2 provides virtual machines that can run your applications.

· **How to do it:**

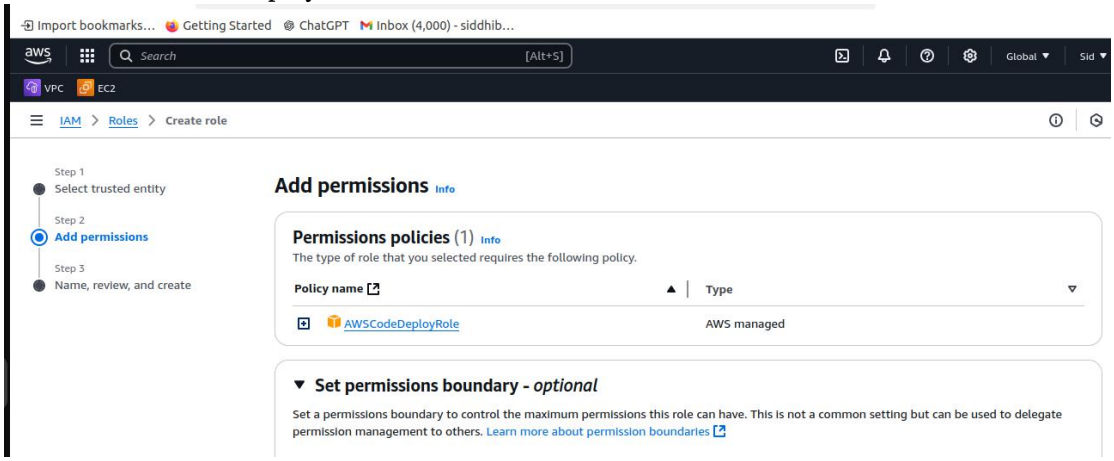
1. Launch an EC2 instance (Ubuntu) from the EC2 console.
2. Install the **CodeDeploy agent** on the EC2 instance using the following documentation: https://docs.aws.amazon.com/codedeploy/latest/userguide/codedeploy-agent-operations-install-ubuntu.html
3. Attach necessary IAM roles to the EC2 instance and ensure they have permissions to access CodeDeploy, ECR, and other resources.

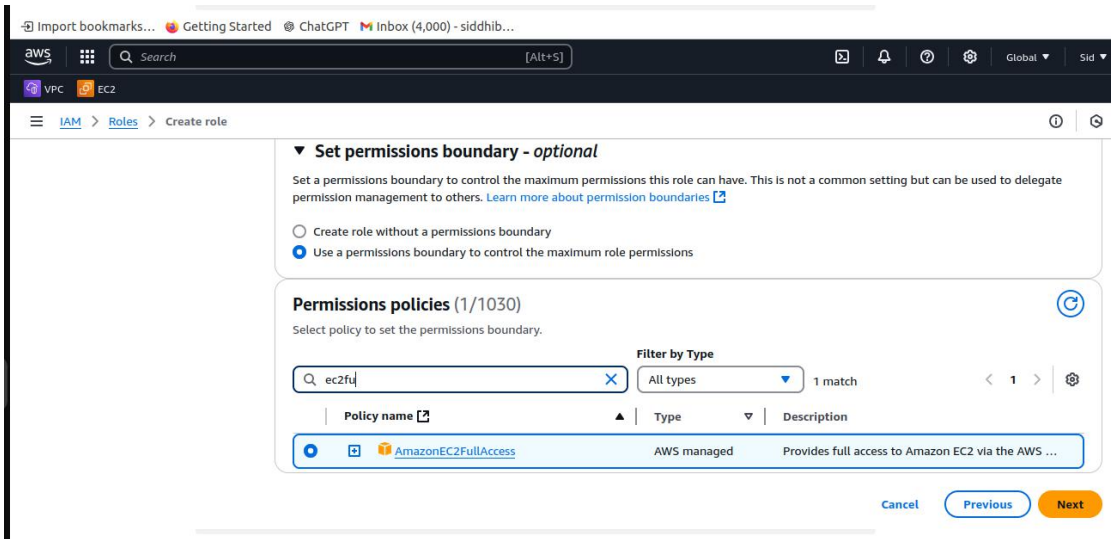


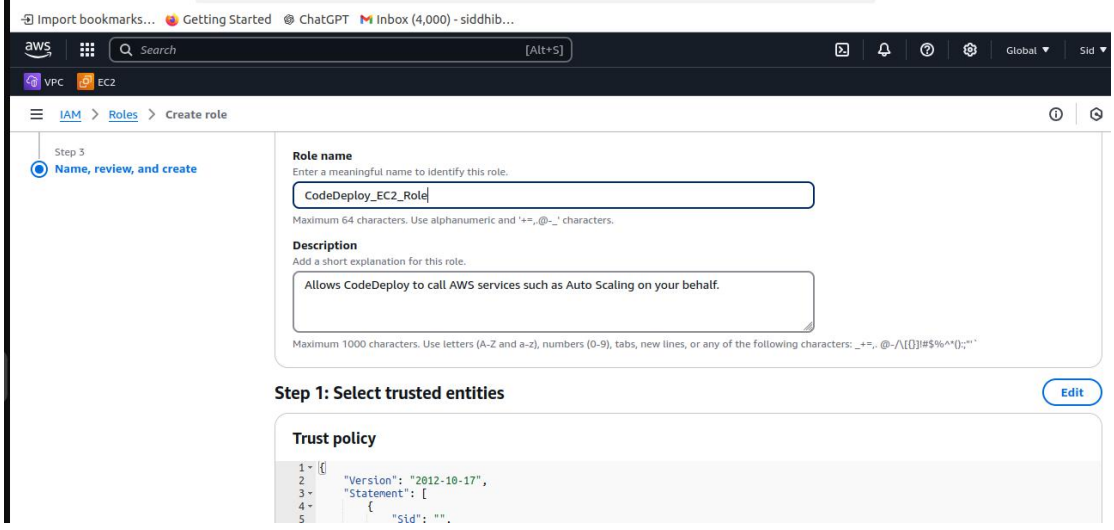


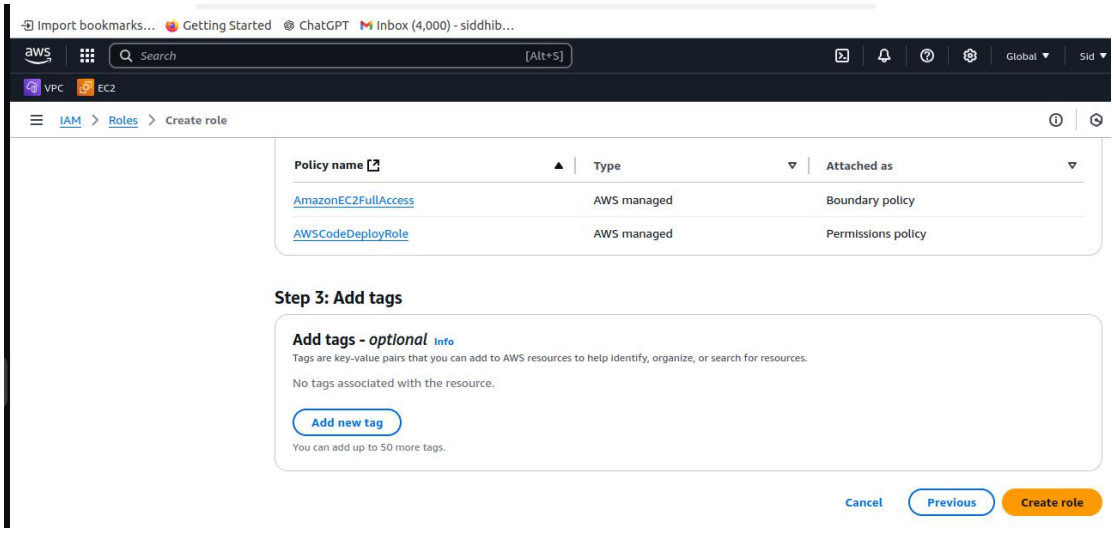

Import bookmarks... Getting Started ChatGPT Inbox (4,000) - siddhib...
Search [Alt+S]
Global Sid
VPC EC2
IAM > Roles > Create role
Step 3
Name, review, and create
AWS service
Allow AWS services like EC2, Lambda, or others to perform actions in this account.
AWS account
Allow entities in other AWS accounts 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
Add permissions
Permissions policies (1/1030) Info
Choose one or more policies to attach to your new role.
Filter by Type
codede
All types
11 matches
Policy name
Type
Description
AmazonEC2RoleforAWSCodeDeploy
AWS managed
Provides EC2 access to S3 bu
AmazonEC2RoleforAWSCodeDeployLimited
AWS managed
Provides EC2 limited access t
AWSCodeDeployDeployerAccess
AWS managed
Provides access to register ar
AWSCodeDeployFullAccess
AWS managed
Provides full access to CodeD
AWSCodeDeployReadOnlyAccess
AWS managed
Provides read only access to
Step 2
Add permissions
Step 3
Name, review, and create
Role details
Role name
Enter a meaningful name to identify this role.
EC2_CodeDeploy_Role
Maximum 64 characters. Use alphanumeric and '+=,.@-_' characters.
Description
Add a short explanation for this role.
Allows EC2 instances to call AWS services on your behalf.
Maximum 1000 characters. Use letters (A-Z and a-z), numbers (0-9), tabs, new lines, or any of the following characters: _+=,. @-/\[{}]!#$%^*();:"'
Step 1: Select trusted entities
Edit
Permissions policy summary
Policy name
Type
Attached as
AWSCodeDeployFullAccess
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.

- Role For CodeDeploy

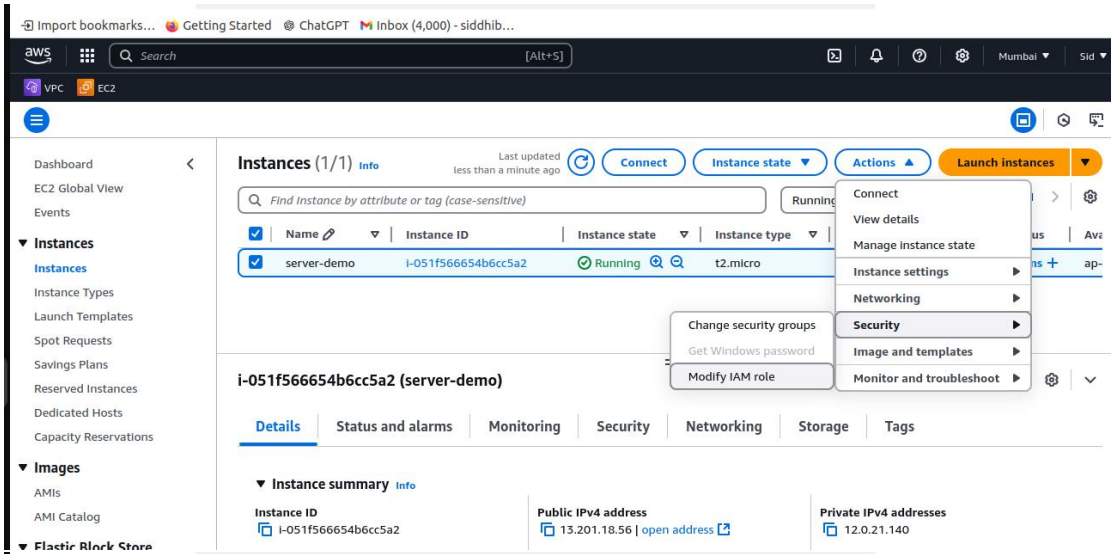


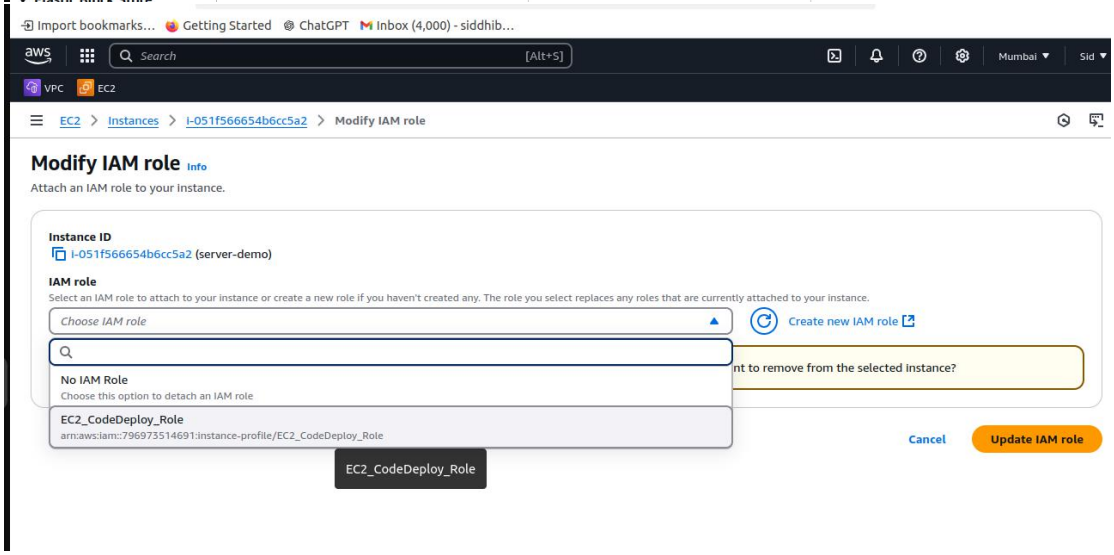






- Assign role to instance



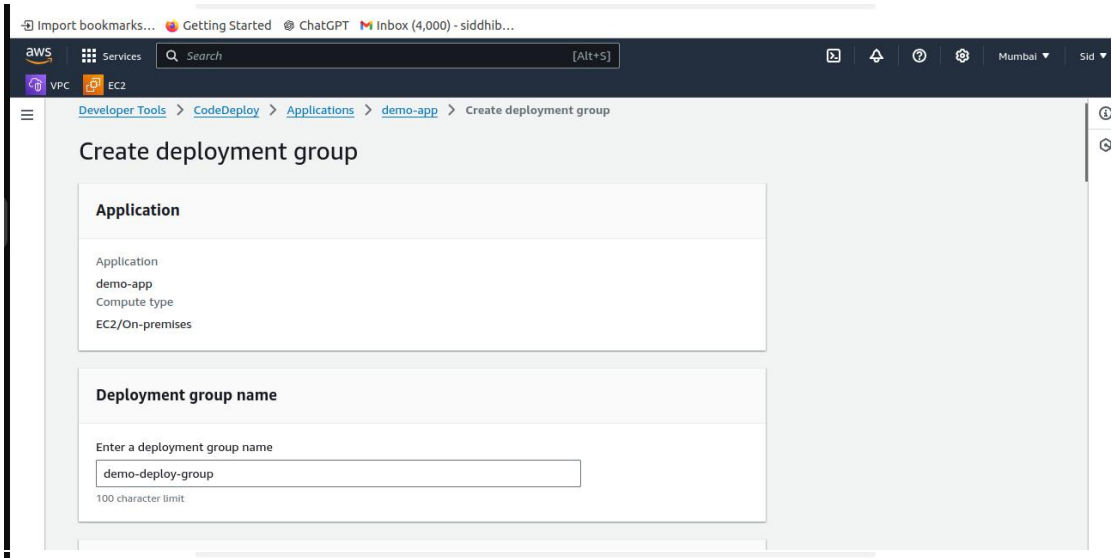


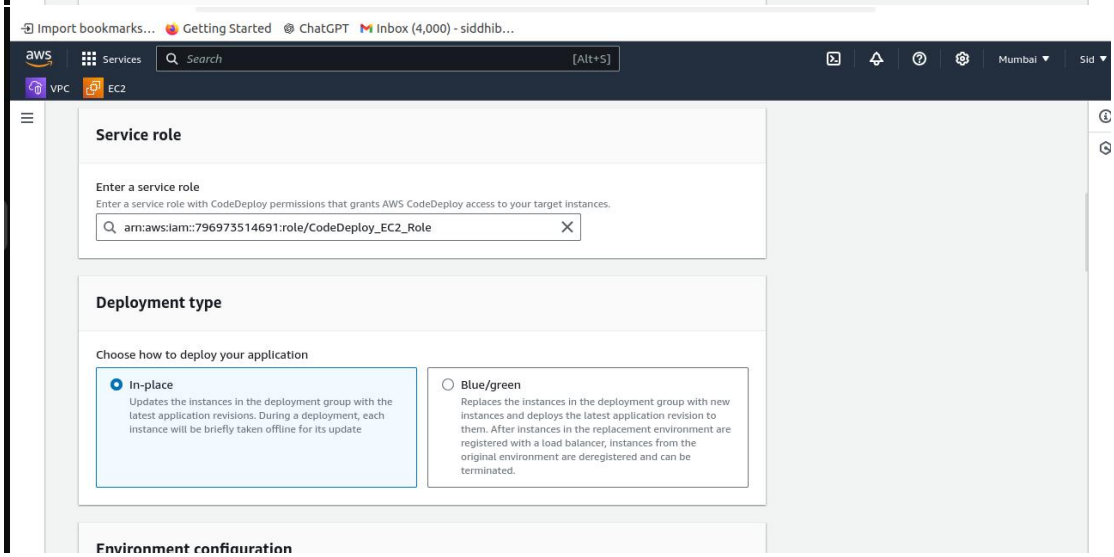
7] Configure CodeDeploy Deployment Group

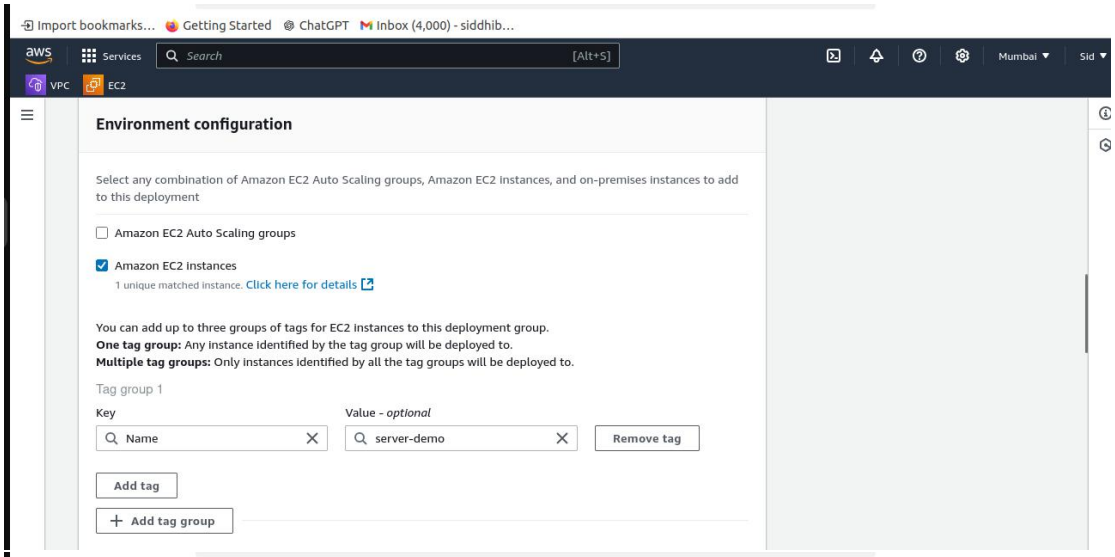
What is Deployment Group? A deployment group is a set of EC2 instances where CodeDeploy deploys the application.

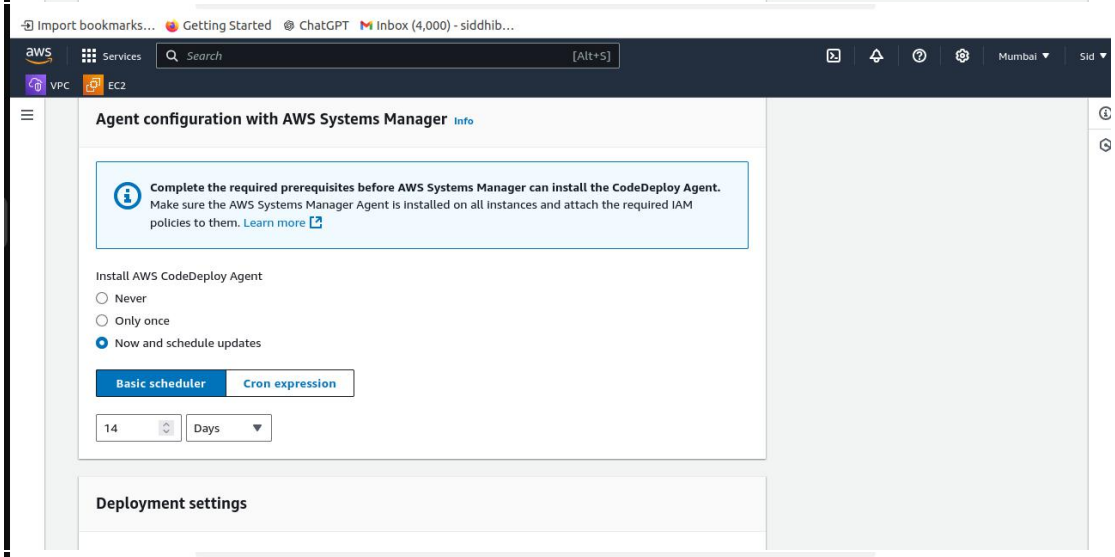
· **How to do it:**

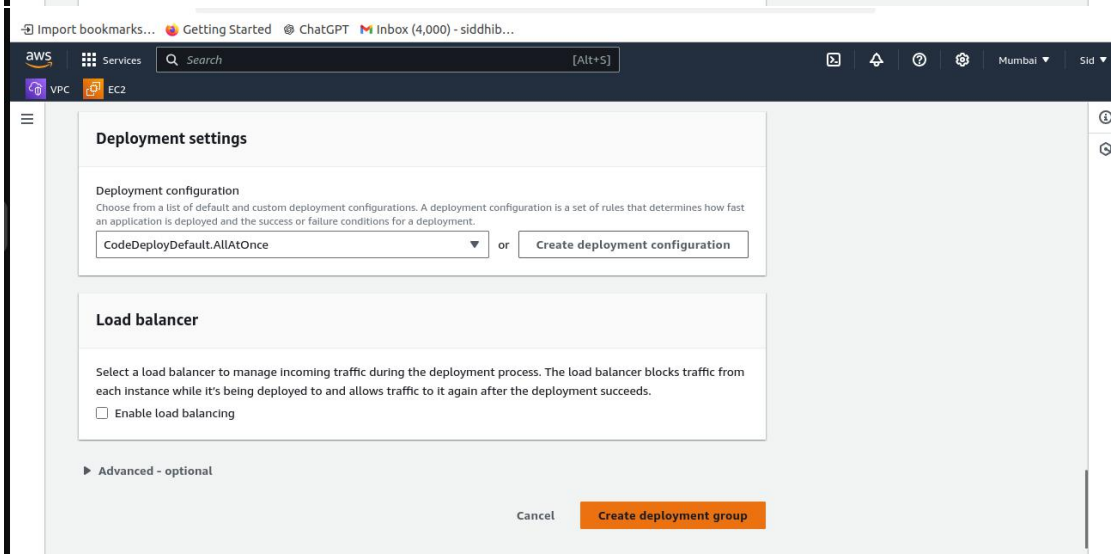
1. Define the deployment group in CodeDeploy (e.g., demo-deploy-group).
2. Choose **In-Place** deployment type, which updates the existing EC2 instance.






Import bookmarks... Getting Started ChatGPT Inbox (4,000) - siddhib...
aws Services Search [Alt+S] Mumbai Sid
VPC EC2
Environment configuration
Select any combination of Amazon EC2 Auto Scaling groups, Amazon EC2 instances, and on-premises instances to add to this deployment
Amazon EC2 Auto Scaling groups
Amazon EC2 instances
1 unique matched instance. Click here for details
You can add up to three groups of tags for EC2 instances to this deployment group.
One tag group: Any instance identified by the tag group will be deployed to.
Multiple tag groups: Only instances identified by all the tag groups will be deployed to.
Tag group 1
Key
Name
Value - optional
server-demo
Remove tag
Add tag
Add tag group


Import bookmarks... Getting Started ChatGPT Inbox (4,000) - siddhib...
aws Services Search [Alt+S] Mumbai Sid
VPC EC2
Agent configuration with AWS Systems Manager Info
Complete the required prerequisites before AWS Systems Manager can install the CodeDeploy Agent.
Make sure the AWS Systems Manager Agent is installed on all instances and attach the required IAM policies to them. Learn more
Install AWS CodeDeploy Agent
Never
Only once
Now and schedule updates
Basic scheduler
Cron expression
14
Days
Deployment settings

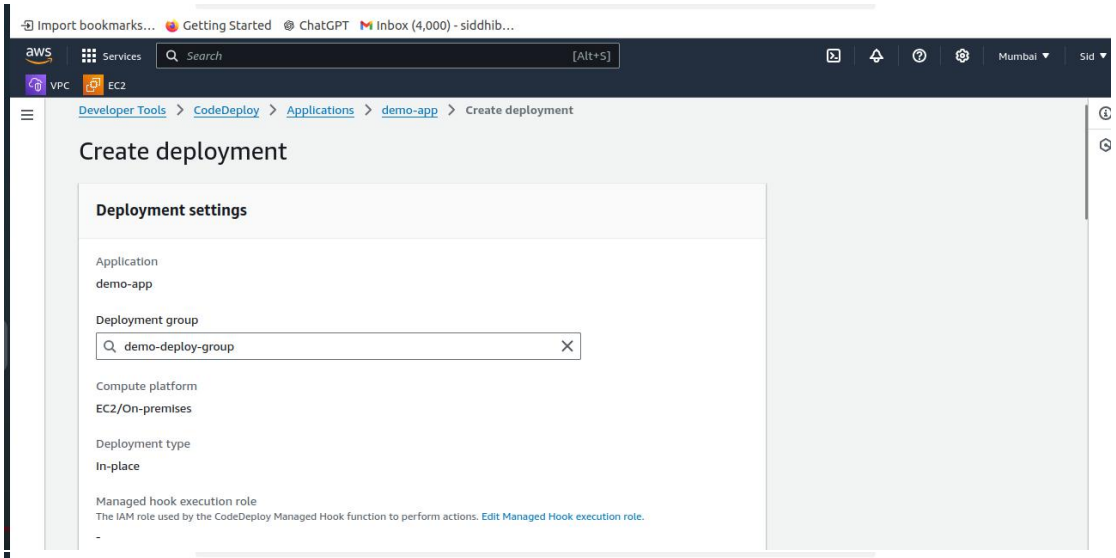

Import bookmarks... Getting Started ChatGPT Inbox (4,000) - siddhib...
aws Services Search [Alt+S] Mumbai Sid
VPC EC2
Deployment settings
Deployment configuration
Choose from a list of default and custom deployment configurations. A deployment configuration is a set of rules that determines how fast an application is deployed and the success or failure conditions for a deployment.
CodeDeployDefault.AllAtOnce
or
Create deployment configuration
Load balancer
Select a load balancer to manage incoming traffic during the deployment process. The load balancer blocks traffic from each instance while it's being deployed to and allows traffic to it again after the deployment succeeds.
Enable load balancing
Advanced - optional
Cancel
Create deployment group

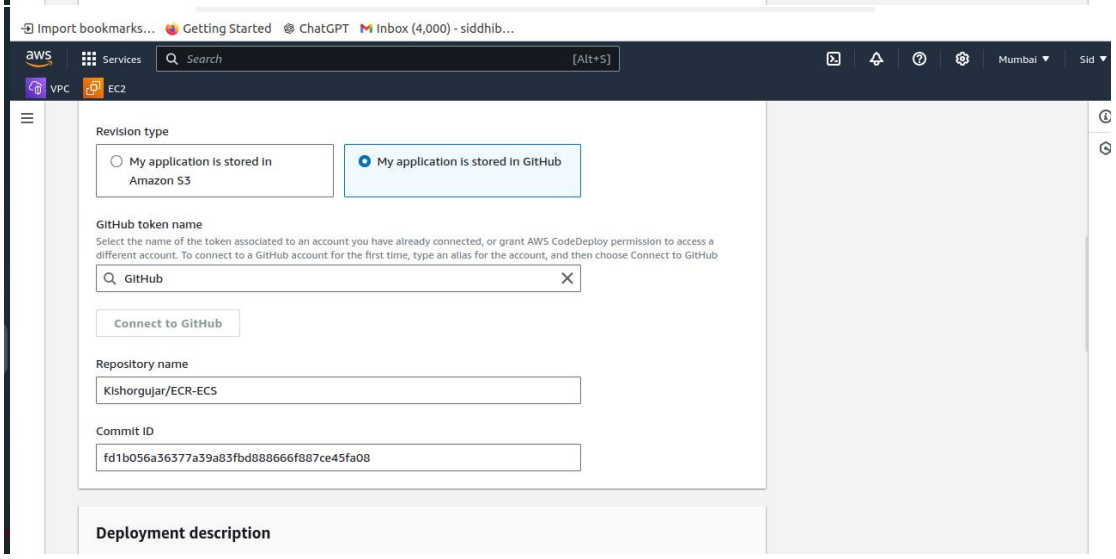
8] Create a Deployment Using CodeDeploy

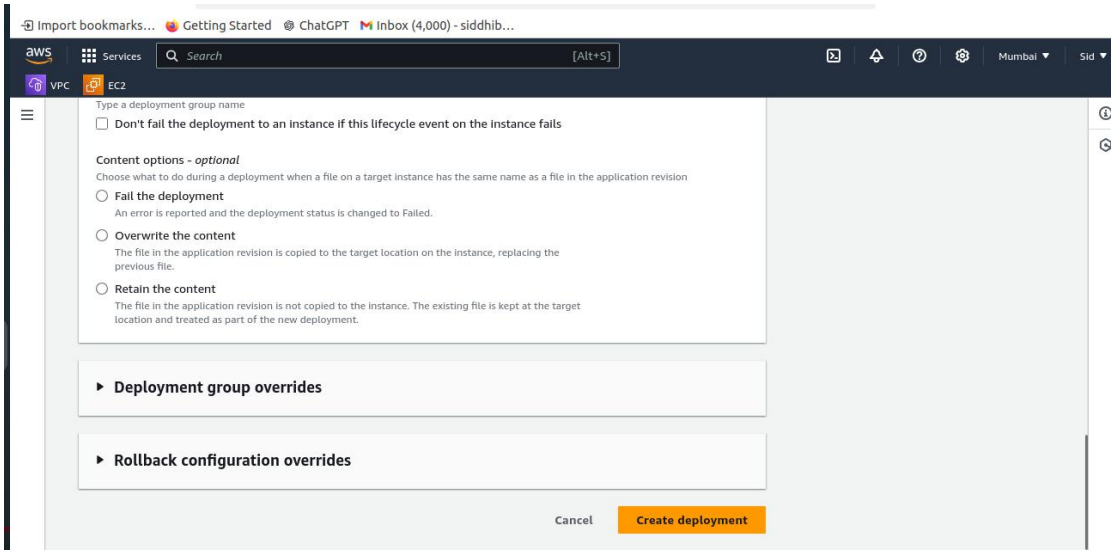
What is a Deployment? A deployment is the process of transferring the built Docker image to the EC2 instance.

· **How to do it:**

1. Create a deployment in CodeDeploy, choosing the GitHub repository as the source.
2. Select the deployment group demo-deploy-group.



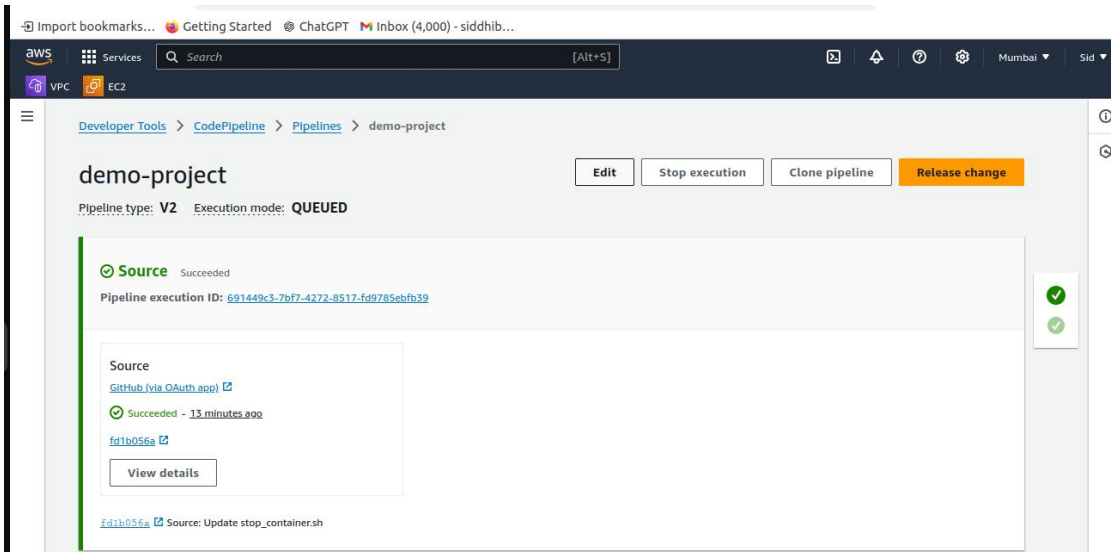


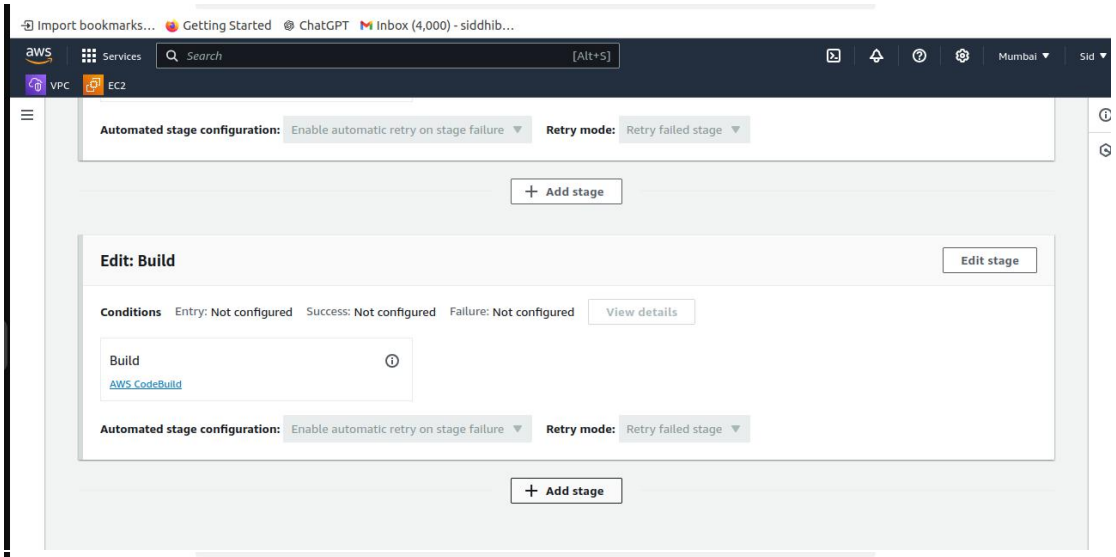


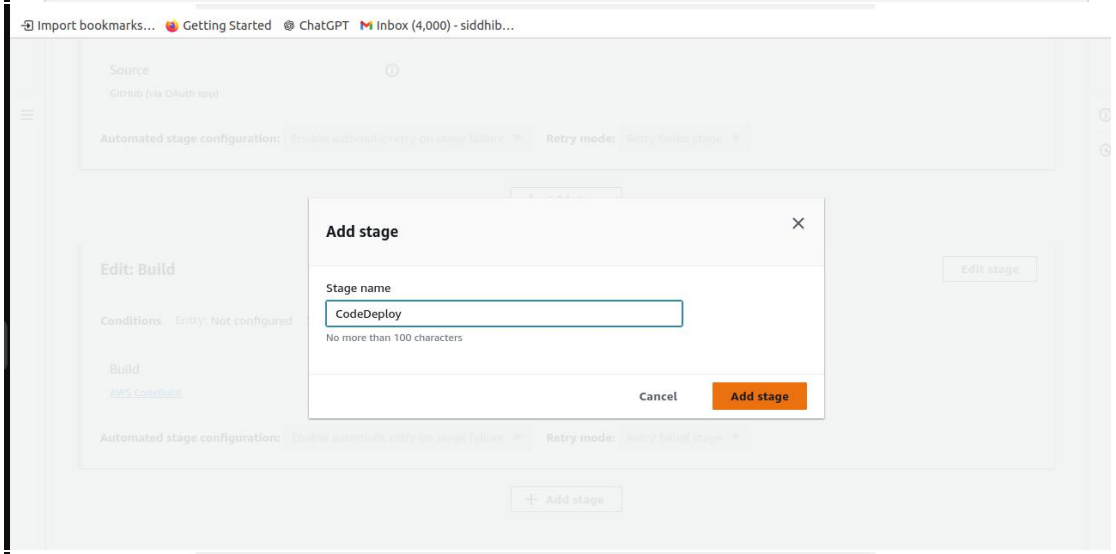
9] Add CodeDeploy Stage in CodePipeline

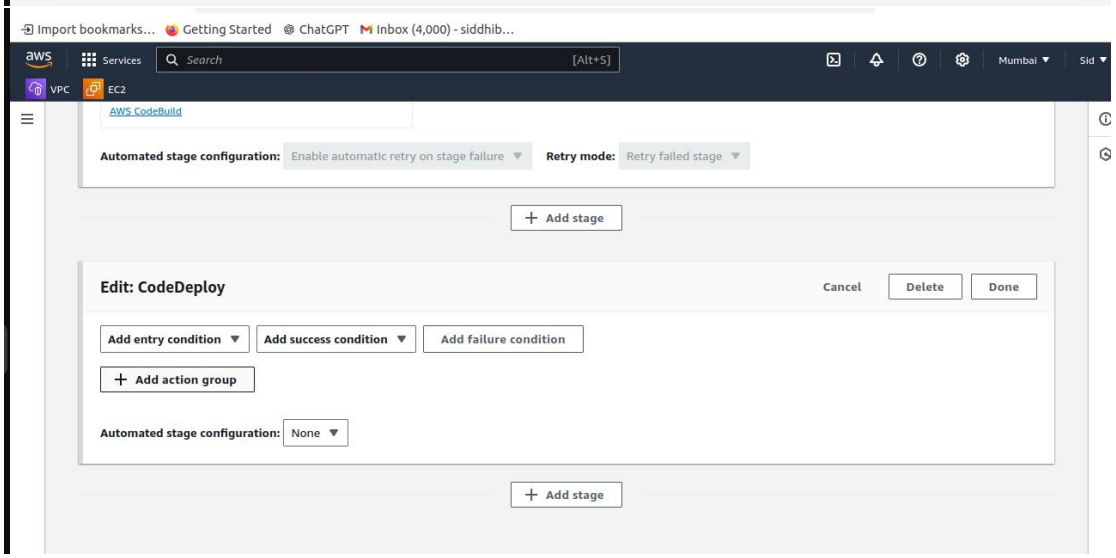
How to do it:

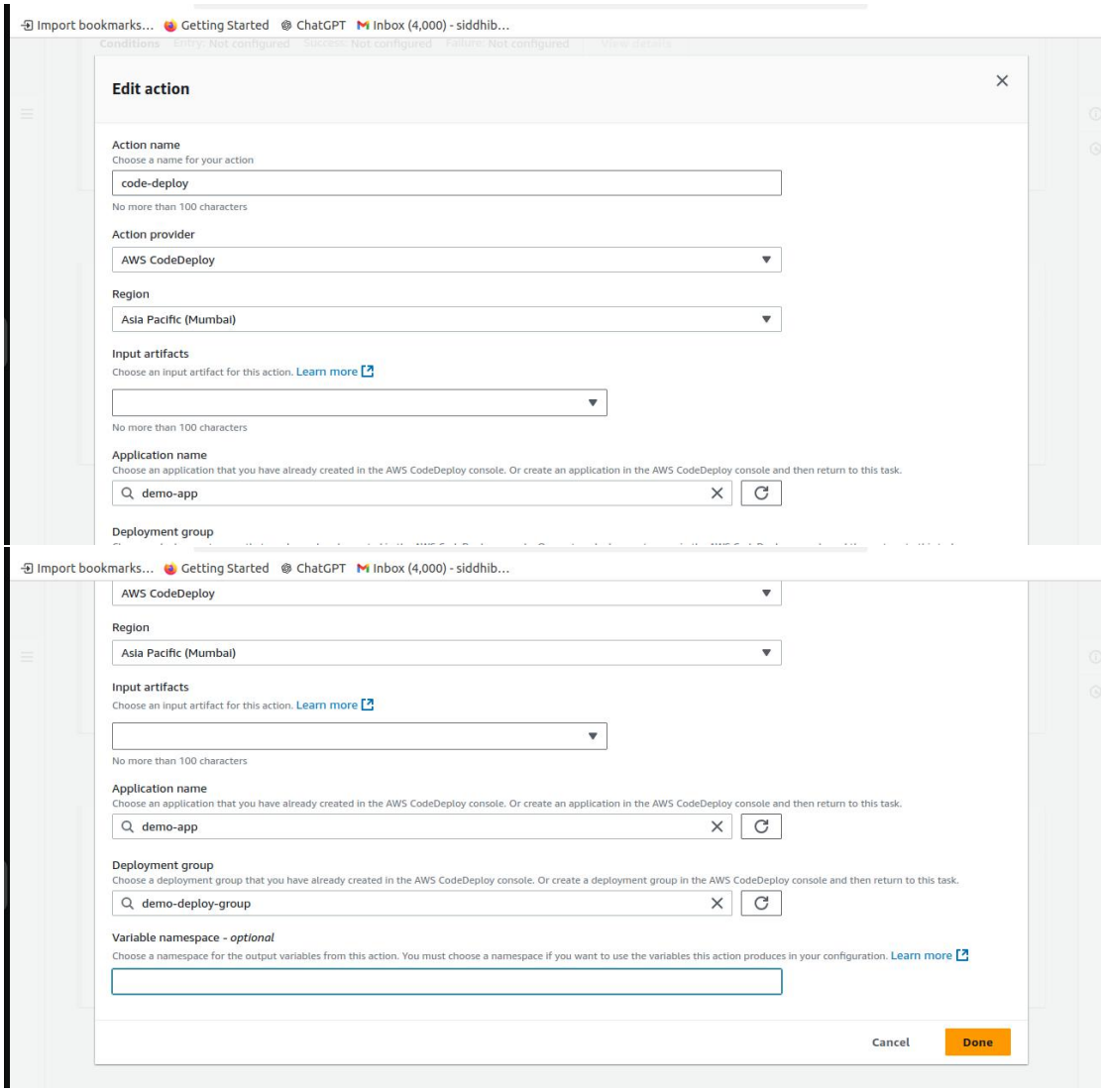
1. In **AWS CodePipeline**, add a new stage for **CodeDeploy** after the build step.
2. Provide details like the application name and deployment group to the pipeline.




Import bookmarks... Getting Started ChatGPT Inbox (4,000) - siddhib...
aws Services Search [Alt+S] Mumbai Sid
VPC EC2
Automated stage configuration: Enable automatic retry on stage failure Retry mode: Retry failed stage
+ Add stage
Edit: Build
Edit stage
Conditions Entry: Not configured Success: Not configured Failure: Not configured View details
Build
AWS CodeBuild
Automated stage configuration: Enable automatic retry on stage failure Retry mode: Retry failed stage
+ Add stage


Import bookmarks... Getting Started ChatGPT Inbox (4,000) - siddhib...
Source
GitHub (via OAuth app)
Automated stage configuration: Retry mode:
Add stage
Stage name
CodeDeploy
No more than 100 characters
Cancel
Add stage
Edit: Build
Edit stage
Conditions Entry: Not configured
Build
AWS CodeBuild
Automated stage configuration: Retry mode:
+ Add stage

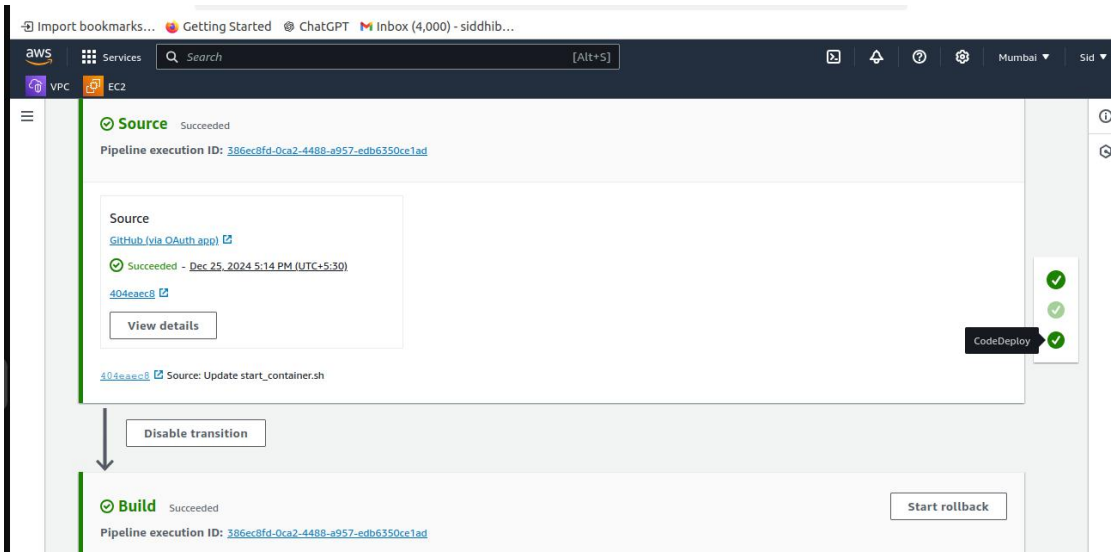

Import bookmarks... Getting Started ChatGPT Inbox (4,000) - siddhib...
aws Services Search [Alt+S] Mumbai Sid
VPC EC2
AWS CodeBuild
Automated stage configuration: Enable automatic retry on stage failure Retry mode: Retry failed stage
+ Add stage
Edit: CodeDeploy
Cancel Delete Done
Add entry condition Add success condition Add failure condition
+ Add action group
Automated stage configuration: None
+ Add stage


Import bookmarks... Getting Started ChatGPT Inbox (4,000) - siddhib...
Edit action
Action name
Choose a name for your action
code-deploy
No more than 100 characters
Action provider
AWS CodeDeploy
Region
Asia Pacific (Mumbai)
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 CodeDeploy console. Or create an application in the AWS CodeDeploy console and then return to this task.
demo-app
Deployment group
Import bookmarks... Getting Started ChatGPT Inbox (4,000) - siddhib...
AWS CodeDeploy
Region
Asia Pacific (Mumbai)
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 CodeDeploy console. Or create an application in the AWS CodeDeploy console and then return to this task.
demo-app
Deployment group
Choose a deployment group that you have already created in the AWS CodeDeploy console. Or create a deployment group in the AWS CodeDeploy console and then return to this task.
demo-deploy-group
Variable namespace - optional
Choose a namespace for the output variables from this action. You must choose a namespace if you want to use the variables this action produces in your configuration. Learn more
Cancel
Done

10] Test the Pipeline

How to do it:

1. Make changes to your GitHub repository to trigger the pipeline.
2. Check the CodePipeline to ensure the code is built and deployed automatically without any manual intervention.



4. Conclusion

By following the steps outlined in this guide, you have successfully created a CI/CD pipeline using AWS tools. This pipeline automates the process of building, testing, and deploying a Python application, allowing you to quickly release new updates in a controlled and automated manner. AWS services like CodeBuild, CodeDeploy, and CodePipeline work together seamlessly to ensure fast and reliable software delivery.

This approach significantly reduces manual tasks and helps in delivering production-ready applications quickly and with fewer errors.