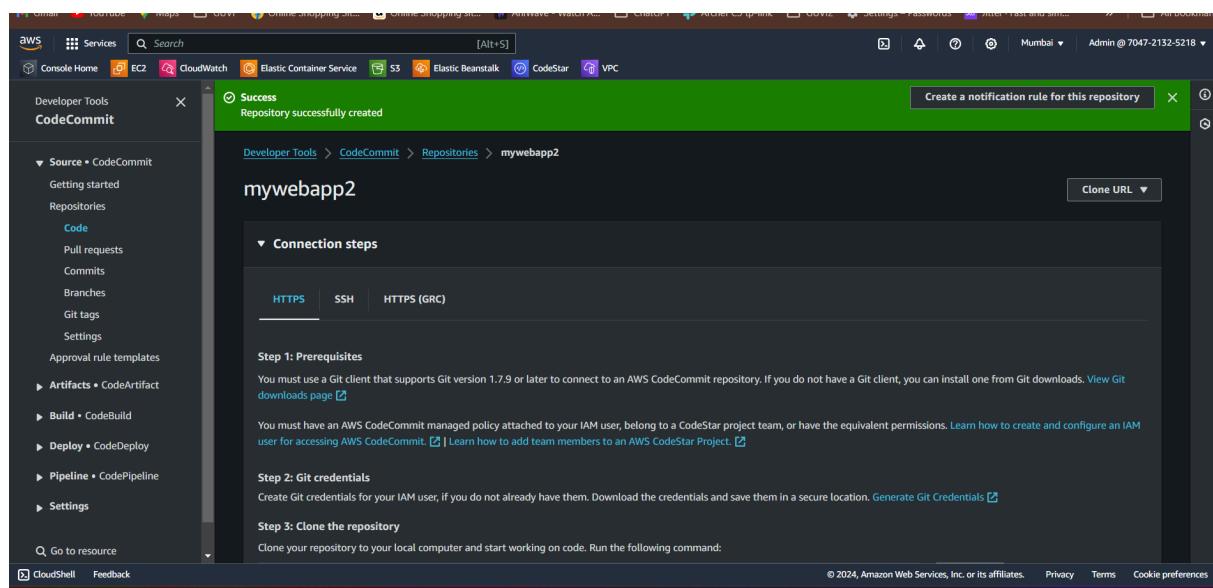
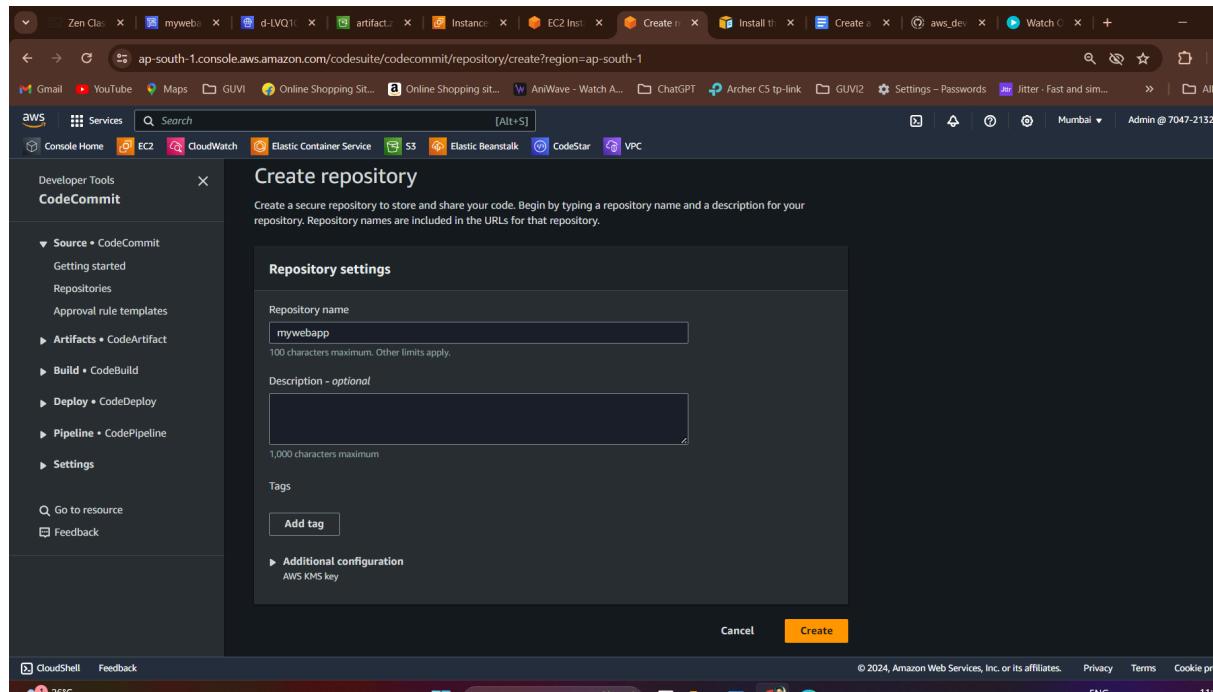


Deploy a simple Nginx application using AWS code commit and deploy & access via browser

Step 1: Create AWS CodeCommit repository - (CC).



Note: It is not good to access CodeCommit through root user, so we have to use IAM user to access CodeCommit.

Step 2: Click on Clone URL —> Clone HTTPS —> Copy the URL link - (**CC**).

Step 1: Go to IAM user —> Security credentials —> HTTPS Git credentials for AWS CodeCommit —> Click Generate credentials —> Click Download - (**IAM**).

The screenshot shows the AWS IAM console. On the left, the navigation menu includes 'Identity and Access Management (IAM)', 'Access management' (with 'Users' selected), and 'Access reports'. The main content area displays the 'Admin' user profile under the 'Users' section. The 'Summary' tab is active, showing details like ARN (arn:aws:iam::704721325218:user/Admin), Console access (Enabled with MFA), and two access keys. The 'Security credentials' tab is also visible. At the bottom, there's a 'Console sign-in' section with a 'Manage console access' button.

The screenshot shows the 'HTTPS Git credentials for AWS CodeCommit' page. It lists one credential for the 'Admin' user, generated on May 29, 2024, at 21:25 (UTC+05:30). The credential is active and was created 2 days ago. There are 'Actions' and 'Generate credentials' buttons at the top right.

Step 2: Use the above credentials and authenticate Codecommit repository - (**IAM**).

Step 3: Open Git Bash —> Enter Download folder —> Clone the HTTPS URL link from AWS CodeCommit as below (Actual folder is mywebapp and the one shown in 1 and 2 SS is example and Use above downloaded **HTTPS credentials as Password** for the below for git clone repository link authentication) - (**IAM**).

1.

```
MINGW64:/c/Users/Syed Abdul Rahim/Downloads/mywebapp2
$ git clone https://git-codecommit.ap-south-1.amazonaws.com/v1/repos/mywebapp2
Cloning into 'mywebapp2'.
warning: You appear to have cloned an empty repository.

Syed Abdul Rahim@LAPTOP-UDGP3QIT MINGW64 ~/Downloads/mywebapp2 (master)
$ ls
mywebapp2

Syed Abdul Rahim@LAPTOP-UDGP3QIT MINGW64 ~/Downloads/mywebapp2 (master)
$ ls
appspec.yml buildspec.yml index.html scripts

Syed Abdul Rahim@LAPTOP-UDGP3QIT MINGW64 ~/Downloads/mywebapp2 (master)
$ git add .
warning: in the working copy of 'Downloads/mywebapp2/appspec.yml', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Downloads/mywebapp2/buildspec.yml', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Downloads/mywebapp2/index.html', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Downloads/mywebapp2/scripts/install_nginx.sh', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Downloads/mywebapp2/scripts/start_nginx.sh', LF will be replaced by CRLF the next time Git touches it

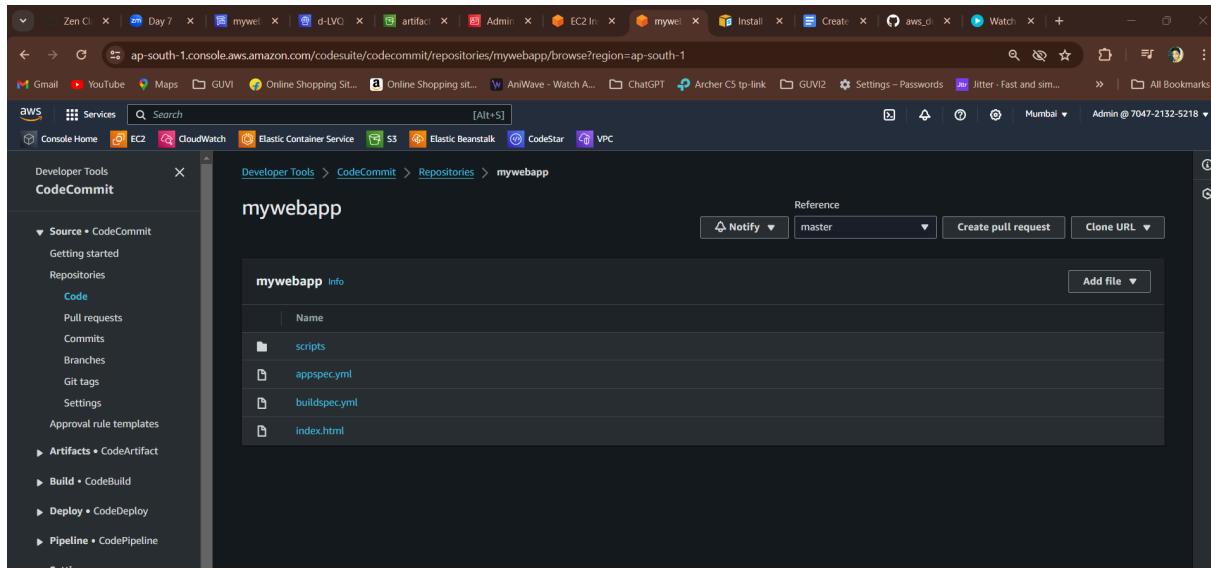
Syed Abdul Rahim@LAPTOP-UDGP3QIT MINGW64 ~/Downloads/mywebapp2 (master)
$ git commit -m "application files"
[master 66c08df] application files
 5 files changed, 68 insertions(+)
 create mode 100644 Downloads/mywebapp2/appspec.yml
 create mode 100644 Downloads/mywebapp2/buildspec.yml
 create mode 100644 Downloads/mywebapp2/index.html
 create mode 100644 Downloads/mywebapp2/scripts/install_nginx.sh
 create mode 100644 Downloads/mywebapp2/scripts/start_nginx.sh

Syed Abdul Rahim@LAPTOP-UDGP3QIT MINGW64 ~/Downloads/mywebapp2 (master)
$ git push
```

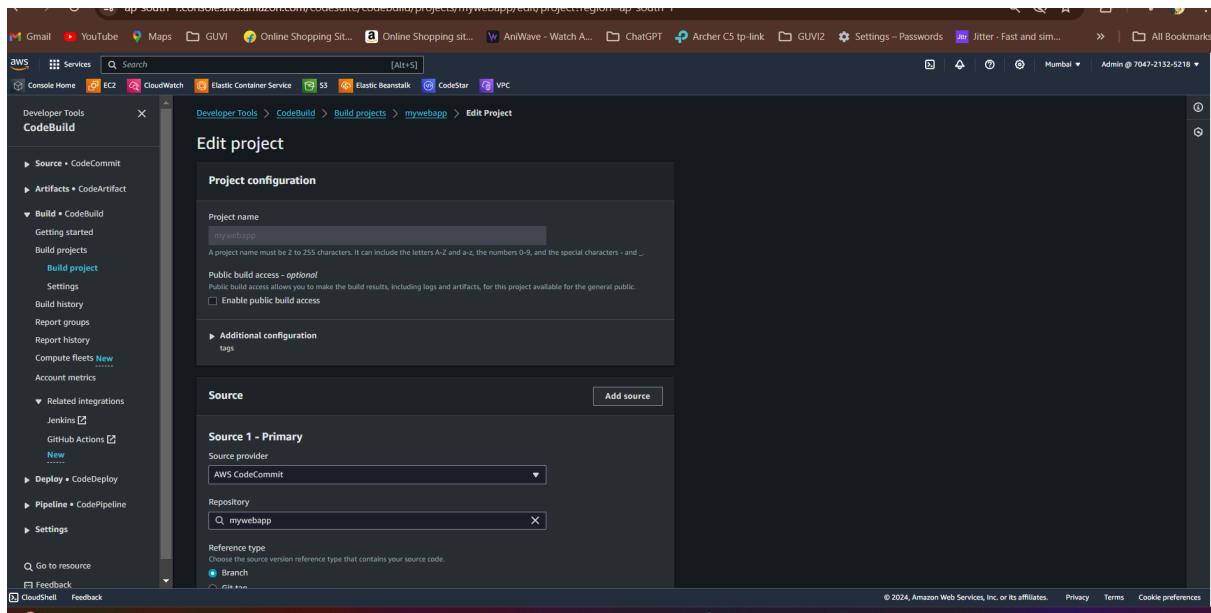
2.

```
yasmin@abdul-personal demo % cd ..
yasmin@abdul-personal ~ % cd Downloads
yasmin@abdul-personal Downloads % git clone https://git-codecommit.ap-south-1.amazonaws.com/v1/repos/mywebapp
Cloning into 'mywebapp'...
Username for 'https://git-codecommit.ap-south-1.amazonaws.com/v1/repos/mywebapp': yasmin-at-150878085644
Password for 'https://yasmin-at-150878085644@git-codecommit.ap-south-1.amazonaws.com/v1/repos/mywebapp':
warning: You appear to have cloned an empty repository.
yasmin@abdul-personal Downloads % cd mywebapp
yasmin@abdul-personal mywebapp % ls
yasmin@abdul-personal mywebapp % ls
appspec.yml    buildspec.yml   index.html      scripts
yasmin@abdul-personal mywebapp % git add .
yasmin@abdul-personal mywebapp % git commit -m "application files"
[master (root-commit) 378ebd9] application files
 5 files changed, 68 insertions(+)
 create mode 100644 appspec.yml
 create mode 100644 buildspec.yml
 create mode 100644 index.html
 create mode 100644 scripts/install_nginx.sh
 create mode 100644 scripts/start_nginx.sh
yasmin@abdul-personal mywebapp % git push
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 8 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (8/8), 1.07 KiB | 546.00 KiB/s, done.
Total 8 (delta 0), reused 0 (delta 0), pack-reused 0
remote: Validating objects: 100%
To https://git-codecommit.ap-south-1.amazonaws.com/v1/repos/mywebapp
 * [new branch]      master -> master
yasmin@abdul-personal mywebapp %
```

Step 4: Goto CodeCommit —> Check the CodeCommit repository —> It should have cloned the data from local git bash repository as per above **Step 3 - (IAM)**.



Step 1: Click on CodeBuild —> Create Project (Below is already filled data with 5 SS) - (CB).



The screenshot shows the AWS CodeBuild configuration interface. On the left, a sidebar menu lists various options under 'CodeBuild'. The 'Source' section is expanded, showing 'CodeCommit' and 'CodeArtifact'. The 'Build' section is expanded, showing 'Getting started', 'Build projects', 'Build project' (which is selected), 'Settings', 'Build history', 'Report groups', 'Report history', 'Compute fleets New', 'Account metrics', 'Related integrations' (with Jenkins and GitHub Actions listed), 'New', 'Deploy' (with CodeDeploy listed), 'Pipeline' (with CodePipeline listed), and 'Settings'. Below these are links for 'Go to resource' and 'Feedback', and buttons for 'CloudShell' and 'Feedback'.

The main configuration area is divided into two sections: 'Source' and 'Environment'.

Source:

- Branch:** master
- Commit ID - optional:** (Search bar)
- Source version info:** refs/heads/master
44f7579a application files
- Additional configuration:** Git clone depth, Git submodules

Environment:

- Provisioning model info:** On-demand (selected) - 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 (selected) - Use an image managed by AWS CodeBuild.
- Custom image:** Specify a Docker image.
- Compute:** EC2 (selected) - Optimized for flexibility during action runs.
- Lambda:** Optimized for speed and minimizes the start up time of workflow actions.

This screenshot shows the same AWS CodeBuild configuration interface as the previous one, but the 'Buildspec' tab is active in the sidebar.

The 'Buildspec' section contains the following settings:

- Current buildspec:** Using the buildspec.yml in the source code root directory
- Build specifications:** Use a buildspec file (selected) - Store build commands in a YAML-formatted buildspec file.
- Buildspec name - optional:** buildspec.yml

The screenshot shows the AWS CodeBuild console with the 'Artifacts' configuration page. The left sidebar is titled 'CodeBuild' and includes sections for Source, Artifacts, Build, Deploy, Pipeline, and Settings. The main area is titled 'Artifacts' and contains the following fields:

- Artifact 1 - Primary**
- Type**: Amazon S3
- Bucket name**: mycodebuilddemobucket
- Name**: artifact
- Enable semantic versioning**: Unchecked
- Path - optional**: artifact.zip
- Namespace type - optional**: None
- Artifacts packaging**: Zip (selected)
- Disable artifact encryption**: Unchecked
- Additional configuration**

The screenshot shows the AWS CodeBuild console with the 'CloudWatch' configuration page. The left sidebar is identical to the previous screenshot. The main area is titled 'CloudWatch' and contains the following fields:

- CloudWatch logs - optional**: Checked
- Group name - optional**: aws/codebuild/mywebapp
- Stream name prefix - optional**: (empty)
- S3**
- S3 logs - optional**: Checked
- Bucket**: mycodebuilddemobucket
- Path prefix**: (empty)
- Disable S3 log encryption**: Unchecked
- Service role permissions**
- Service role**: Choose an existing service role from your account. A search bar shows arn:aws:iam:704721325218:role/service-role/codebuild-mywebapp-service-ri
- Allow AWS CodeBuild to modify this service role so it can be used with this build project**: Checked

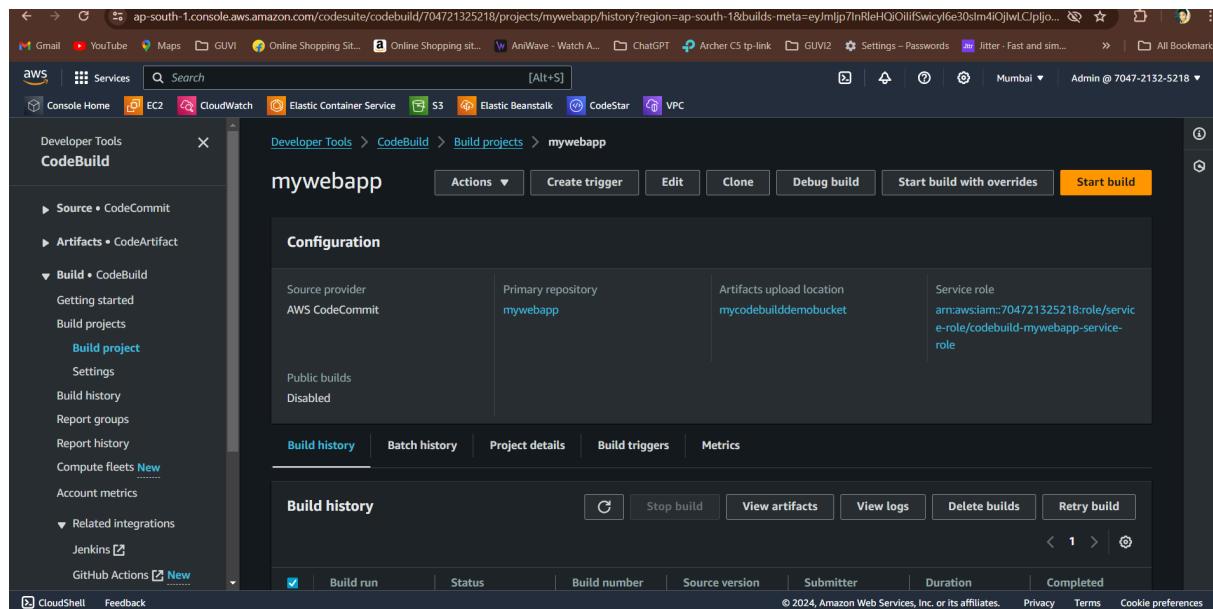
Step 2: Click on Create build project for above SS - (CB).

Step 1: Create a s3 Bucket —> Just give the Bucket name —> Create Bucket - (s3 B).

The screenshot shows the 'Create bucket' page in the AWS S3 console. The 'Bucket name' field contains 'mycodebuilddemobucket'. The 'General configuration' section includes fields for 'AWS Region' (Asia Pacific (Mumbai) ap-south-1) and 'Bucket name' (mycodebuilddemobucket). A 'Choose bucket' button is available for copying settings from an existing bucket. The 'Object Ownership' section notes that object ownership is determined by ACLs. The left sidebar shows navigation options like Buckets, Storage Lens, and Feature spotlight.

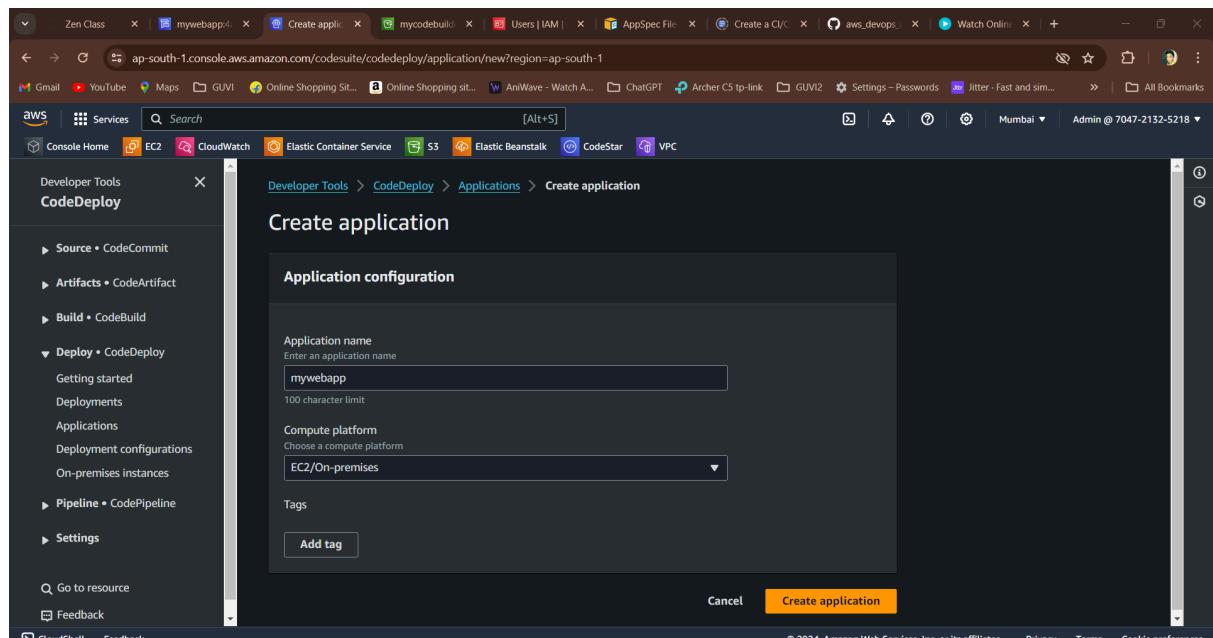
The screenshot shows the 'Create bucket' page in the AWS S3 console, focusing on advanced settings. It includes sections for 'Encryption type' (Server-side encryption with Amazon S3 managed keys (SSE-S3) is selected), 'Bucket Key' (Bucket Keys are disabled), and 'Advanced settings'. A note at the bottom states: 'After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.' The 'Create bucket' button is visible at the bottom right.

Step 2: Click on Start Build - (s3 B).



The screenshot shows the AWS CodeBuild console with the URL <https://ap-south-1.console.aws.amazon.com/codesuite/codebuild/704721325218/projects/mywebapp/history?region=ap-south-1&builds.meta=eyJmIjp7InRleHQiOjIifSwicyI6e30slm4lQjwLClpj...>. The page displays the configuration for the 'mywebapp' project, including source provider (AWS CodeCommit), primary repository ('mywebapp'), artifacts upload location ('mycodebuilddemobucket'), and service role ('arn:aws:iam::704721325218:role/service-role/codebuild-mywebapp-service-role'). The 'Build history' tab is selected, showing one build run that is currently running. The 'Actions' bar at the top includes 'Create trigger', 'Edit', 'Clone', 'Debug build', 'Start build with overrides', and the 'Start build' button, which is highlighted in orange.

Step 1: Create CodeDeploy —> Create application - (CD).



The screenshot shows the AWS CodeDeploy console with the URL <https://ap-south-1.console.aws.amazon.com/codesuite/codedeploy/application/new?region=ap-south-1>. The page is titled 'Create application' under the 'Application configuration' section. It requires an 'Application name' ('mywebapp') and a 'Compute platform' ('EC2/On-premises'). A 'Tags' section with an 'Add tag' button is also present. The 'Create application' button is highlighted in orange at the bottom right of the form.

Created application.

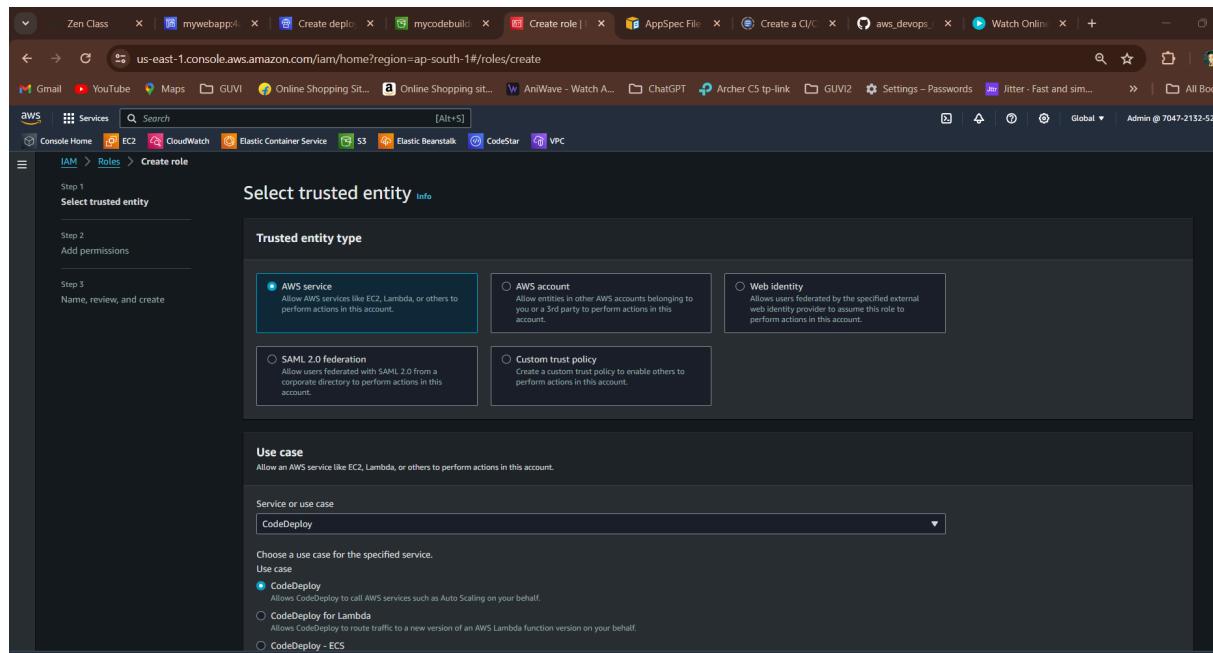
The screenshot shows the AWS CodeDeploy console. On the left, a sidebar menu for 'CodeDeploy' is open, showing options like 'Source', 'Artifacts', 'Build', 'Deploy', 'Pipeline', and 'Settings'. Under 'Deploy', 'CodeDeploy' is selected. In the main content area, a green banner at the top says 'Application created' with the sub-instruction 'In order to create a new deployment, you must first create a deployment group.' Below this, the application name 'mywebapp' is displayed. A 'Notify' button and a 'Delete application' button are visible. The 'Deployment groups' tab is selected, showing a table with columns for Name, Status, Last attempted deploy..., Last successful deploy..., and Trigger count. A 'Create deployment group' button is highlighted in orange. The bottom of the screen includes standard AWS navigation links for CloudShell, Feedback, and copyright information.

Step 2: Deploy the above application in EC2 —> Click on Create deployment group - (CD).

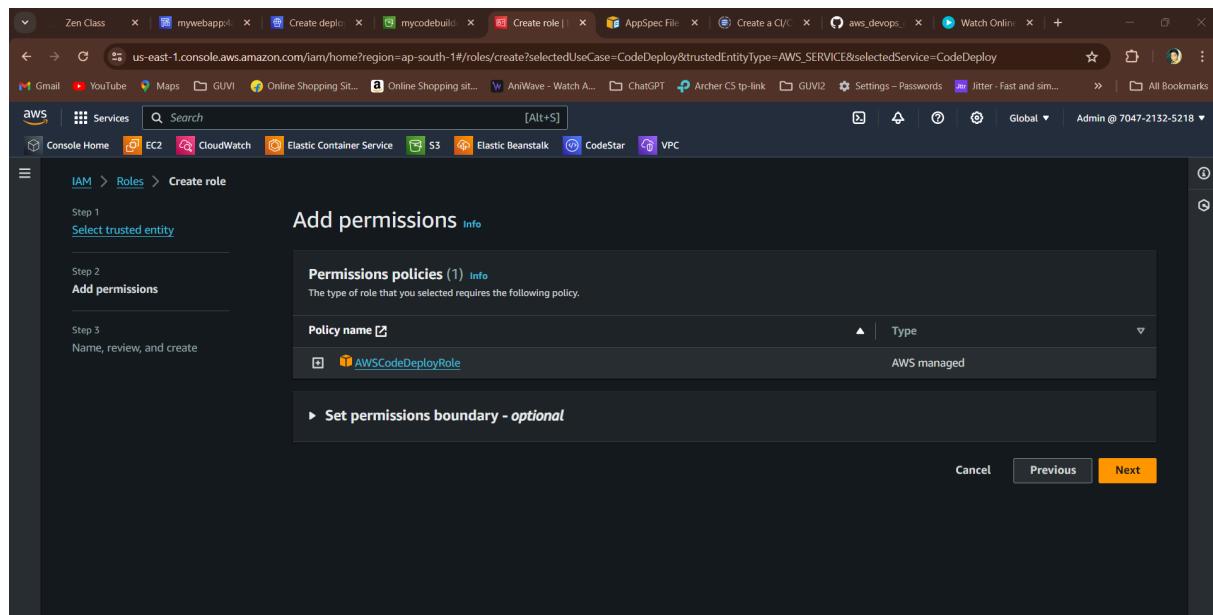
The screenshot shows the 'Create deployment group' wizard. The sidebar on the left is identical to the previous screenshot. The main area has a header 'Create deployment group'. It contains three sections: 'Application' (which shows 'mywebapp' selected), 'Deployment group name' (with a text input field containing 'mywebapp-dg'), and 'Service role' (which is currently empty). The bottom of the screen includes standard AWS navigation links for CloudShell, Feedback, and copyright information.

Step 1: In Service role (There won't be any default code deploy role options, so we create it manually) —> Goto IAM —> Click on Roles —> Create Role - (CD).

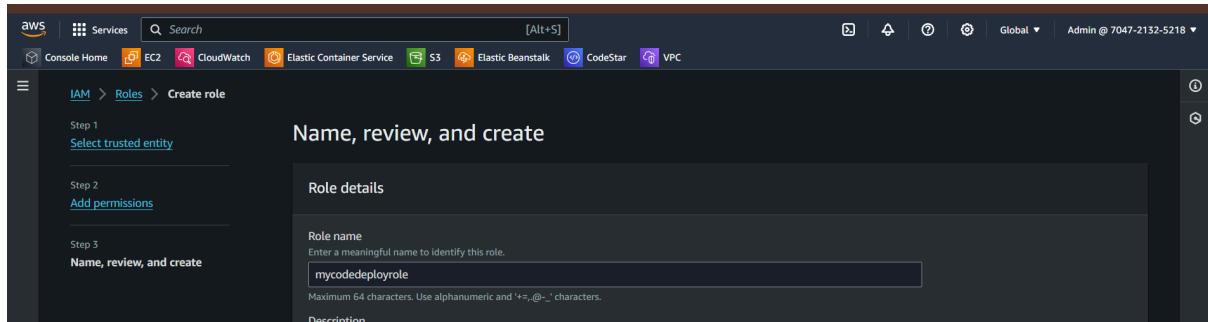
Step 2: Create IAM role (01) and fill as per requirements like below —> Click next - (CD).



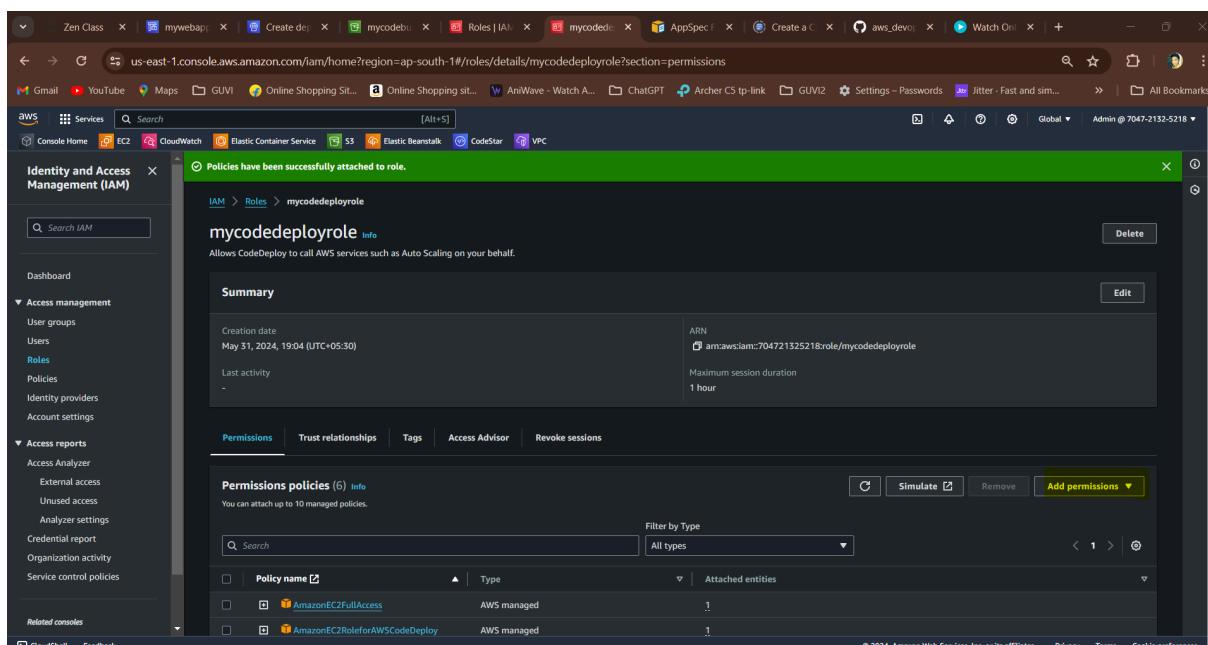
Step 3: As it is for codedeploy we make no changes - (CD).



Step 4: Just give the Role name —> Click Create Role(Make no other changes) - (CD).



Step 5: As more permissions need to be added —> Click Add permissions —> Select Attach policies —> And select the below permissions - (CD).



The screenshot shows the AWS IAM Permissions policies page. A green banner at the top states "Policies have been successfully attached to role." Below this, there is a table listing six AWS managed policies:

Policy name	Type	Attached entities
AmazonEC2FullAccess	AWS managed	1
AmazonEC2RoleforAWSCodeDeploy	AWS managed	1
AmazonEC2RoleforAWSCodeDeployLi...	AWS managed	1
AmazonS3FullAccess	AWS managed	2
AWSCodeDeployFullAccess	AWS managed	1
AWSCodeDeployRole	AWS managed	1

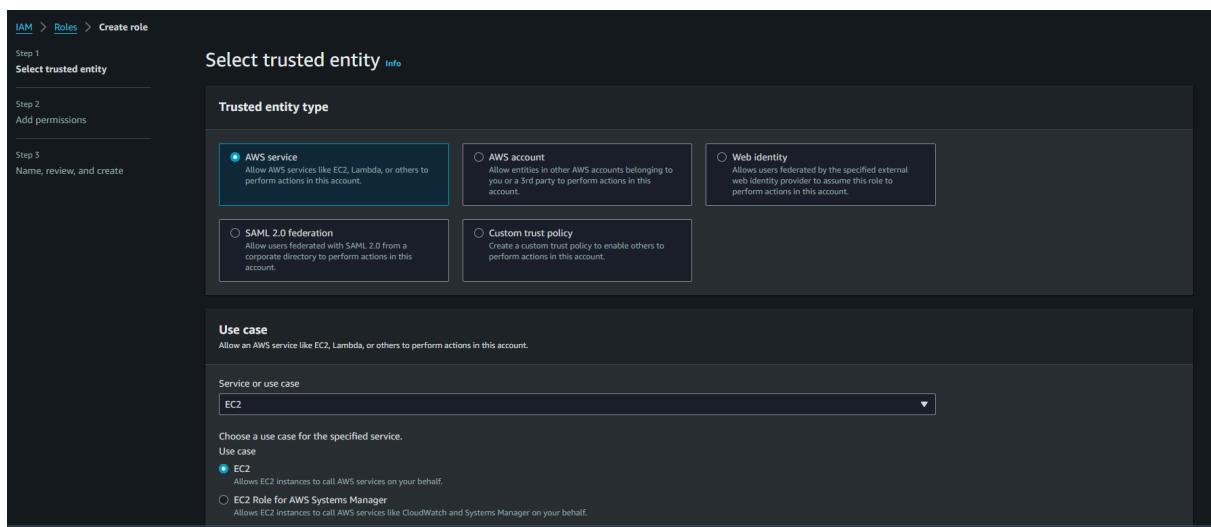
Step 3: Select the above Service role created from the option - (CD).

The screenshot shows the AWS CodeDeploy Create New Deployment Group page. The deployment group name is set to "mywebapp-dg". In the "Service role" section, the role "mycodedeployrole" is selected. Under "Deployment type", the "In-place" option is chosen. The "In-place" description indicates it updates instances with the latest application revisions.

Step 4: Select Amazon EC2 Instance —> Select an EC2 instance (If not create one manually with version Ubuntu Server 22.04 LTS) - (CD).

The screenshot shows the AWS EC2 Instances page. It lists two instances: "Docker-Day10..." (Stopped) and "myec2-cicd" (Running). The "myec2-cicd" instance is highlighted in yellow.

Step 1: Create a IAM role (02) for the above EC2 instance as per below —> Click next - (CD).



Step 2: As this is for EC2 instance, name and permissions added —> Click Create role - (CD).

Name, review, and create

Role details

Role name
Enter a meaningful name to identify this role.
ec2-codeddeploy
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

Trust policies

Step 2: Add permissions Edit

Permissions policy summary

Policy name	Type	Attached as
AmazonEC2FullAccess	AWS managed	Permissions policy
AmazonS3FullAccess	AWS managed	Permissions policy
AWSCodeDeployFullAccess	AWS managed	Permissions policy

Step 3: Add tags

Add tags - optional

Step 5: How to attach EC2 IAM role → Action → Security → Modify IAM role - (CD).

Instances (1/2) Info

Actions Launch instances

- Connect
- View details
- Manage instance state
- Instance settings
- Networking
- Security**
- Image and templates
- Monitor and troubleshoot

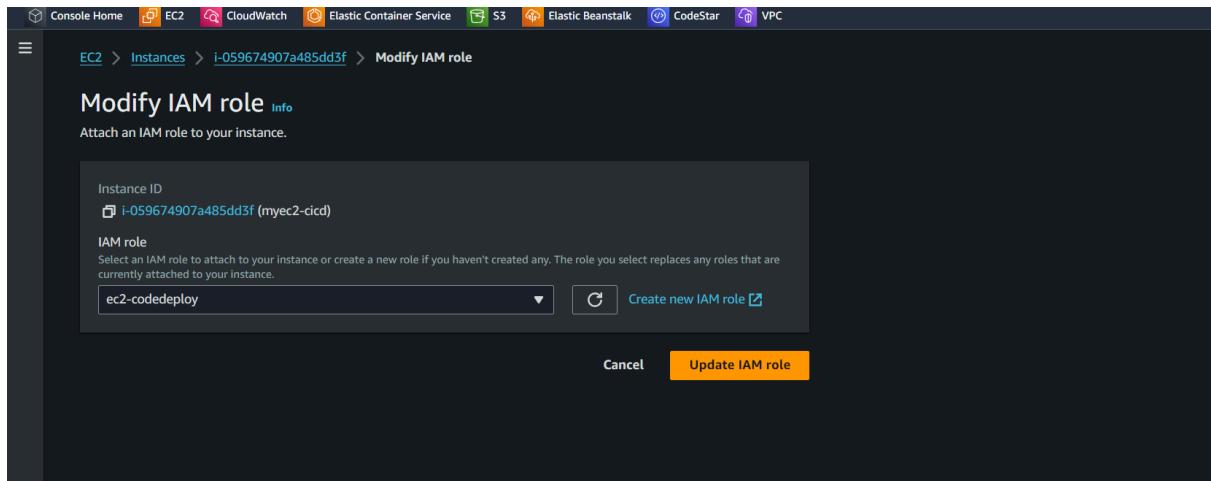
i-059674907a485dd3f (myec2-cicd)

Details Status and alarms Monitoring Security Networking Storage Tags

Instance summary

Instance ID: i-059674907a485dd3f (myec2-cicd)
Public IPv4 address: 13.126.116.251 | [open address](#)
Private IPv4 addresses: 172.31.32.204
Public IPv4 DNS: ec2-13-126-116-251.ap-south-1.compute.amazonaws.com | [open address](#)

Step 6: Select the IAM role that was created for EC2 instance —> Click Update IAM role - (CD).



Step 7: Launch the created EC2 instance —> Install a CodeDeploy agent for this EC2 instance —> through one of the below link —> After installation is complete it should show as per below **active (running)** - (CD).

<https://docs.aws.amazon.com/codedeploy/latest/userguide/codedeploy-agent-operations-install-ubuntu.html>

OR

https://docs.google.com/document/d/1PWghD1ukIVI37KiR92n9ln8WEUyO6bbbkXU_U_ZtC13E/edit?usp=sharing

```

dpkg-deb: building package 'codedeploy-agent' in 'codedeploy-agent_1.3.2-1902_all.deb'.
Selecting previously unselected package codedeploy-agent.
(Reading database ... 81702 files and directories currently installed.)
Preparing to unpack codedeploy-agent_1.3.2-1902_all.deb ...
Unpacking codedeploy-agent (1.3.2-1902) ...
Setting up codedeploy-agent (1.3.2-1902) ...
codedeploy-agent.service is not a native service, redirecting to systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable codedeploy-agent
codedeploy-agent.service                                loaded active running LSB: AWS CodeDeploy Host Agent
● codedeploy-agent.service - LSB: AWS CodeDeploy Host Agent
  Loaded: loaded (/etc/init.d/codedeploy-agent; generated)
  Active: active (running) since Fri 2024-05-31 17:21:28 UTC; 84ms ago
    Docs: man:systemd-sysv-generator(8)
  Process: 2760 ExecStart=/etc/init.d/codedeploy-agent start (code=exited, status=0/SUCCESS)
    Tasks: 2 (limit: 1121)
   Memory: 37.8M
      CPU: 553ms
     CGroup: /system.slice/codedeploy-agent.service
             ├─2766 "codedeploy-agent: master 2766" " "
             └─2768 "codedeploy-agent: booting child" " "
May 31 17:21:28 ip-172-31-41-63 systemd[1]: Starting LSB: AWS CodeDeploy Host Agent...
May 31 17:21:28 ip-172-31-41-63 codedeploy-agent[2760]: Starting codedeploy-agent:
May 31 17:21:28 ip-172-31-41-63 systemd[1]: Started LSB: AWS CodeDeploy Host Agent.
ubuntu@ip-172-31-41-63:~$ 

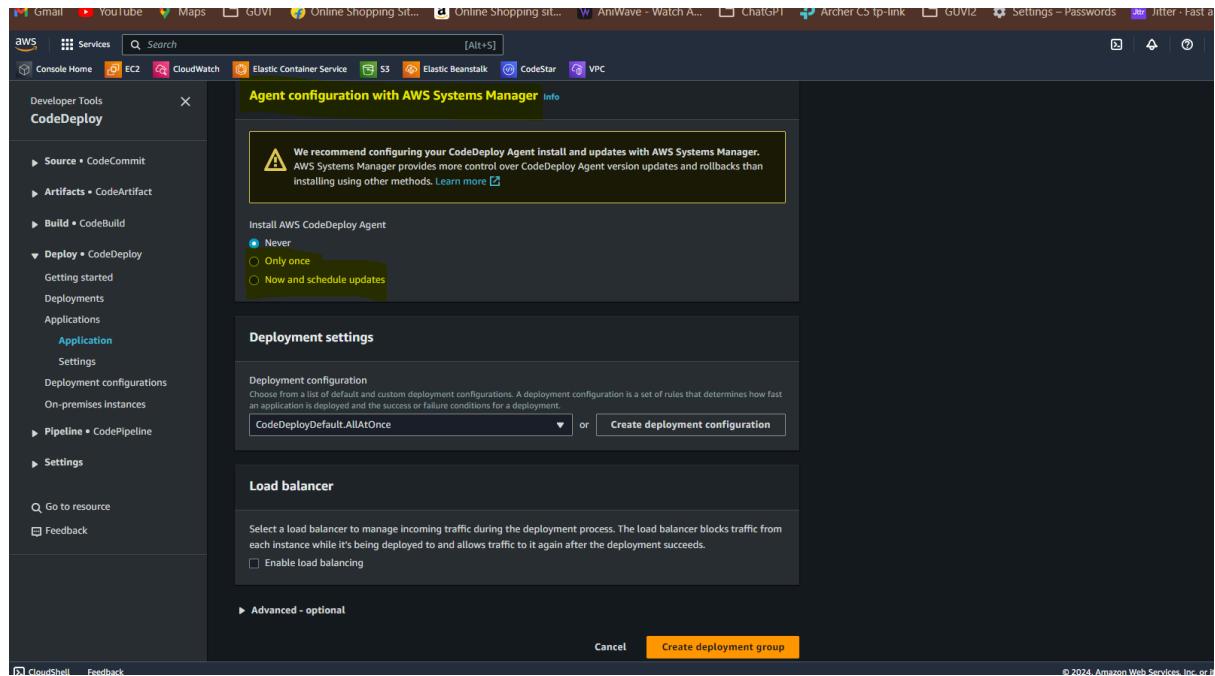
```

i-0322fa7a199ba4b16 (myec2-cicd)
Public IPs: 15.207.86.0 Private IPs: 172.31.41.63

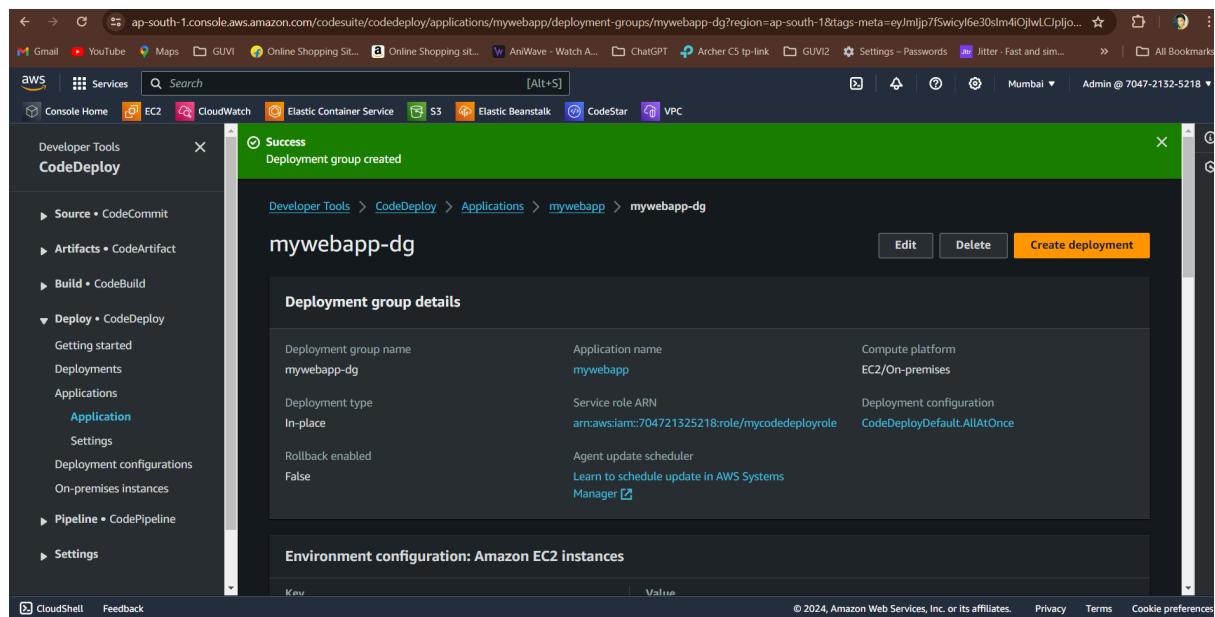
Step 8: Select the EC2 instance that was created with role - (CD).

The screenshot shows the AWS CodeDeploy environment configuration screen. On the left, there's a sidebar with navigation links like Source, Artifacts, Build, Deploy, Pipeline, and Settings. The Deploy section is expanded, showing sub-options like Auto Scaling groups, EC2 Instances, Applications, and On-premises instances. The main area is titled 'Environment configuration' and contains a message about instances being taken offline for update. Below this, it says 'Select any combination of Amazon EC2 Auto Scaling groups, Amazon EC2 instances, and on-premises instances to add to this deployment'. Under 'Amazon EC2 Instances', there's a checkbox labeled 'Amazon EC2 Instances' which is checked. A note below says '2 unique matched instances. Click here for details'. There's also a 'Tag group' section where a tag named 'Name' with value 'myec2-cicd' is listed. At the bottom, there's a 'Matching instances' section with a note '2 unique matched instances. Click here for details'.

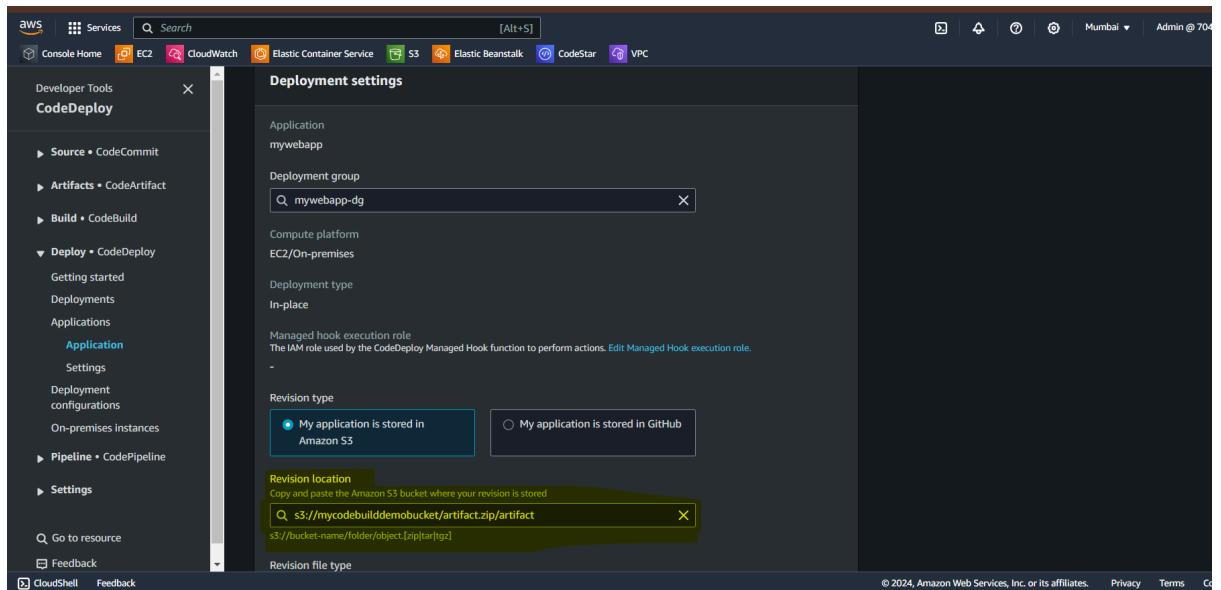
Step 9: We can also install CodeDeploy agent directly as per below, however if you want to manually install CodeDeploy agent then follow the link in **Step 7** —> Uncheck Load balancer —> Click Create deployment group - (CD).



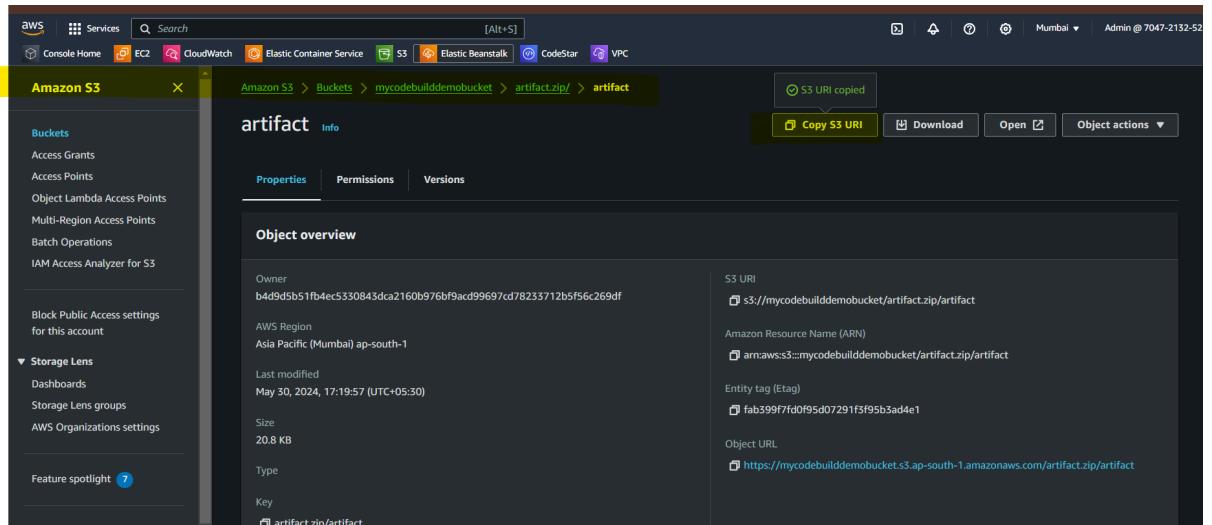
Step 10: Deployment group created —> Click Create deployment - (CD).



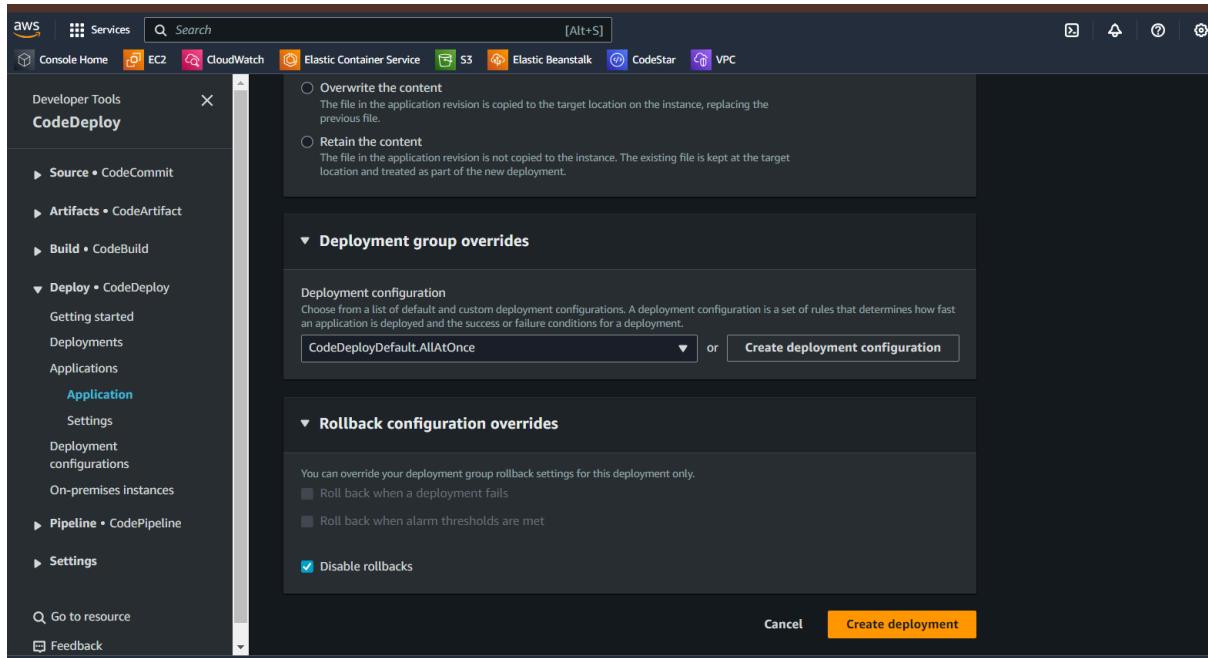
Step 11: Fill the Revision location with **s3 URL link** - (CD).



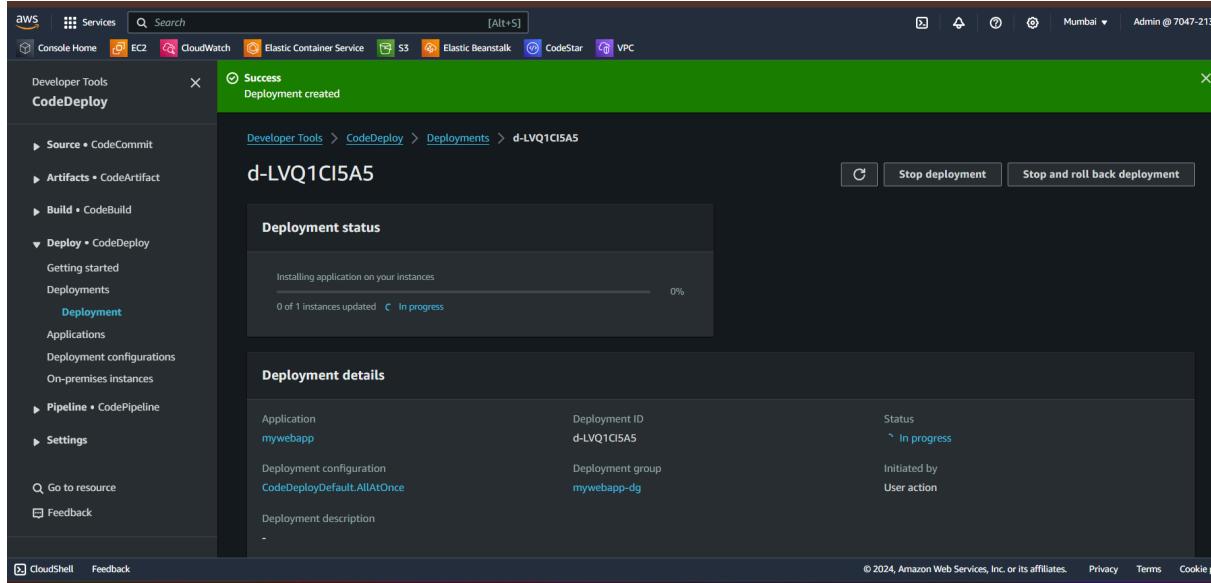
You can find the **s3 URL link** in below location.



Step 12: Leave the rest as default —> Click Create deployment - (CD).



Step 13: CodeDeploy getting deployed - (CD).



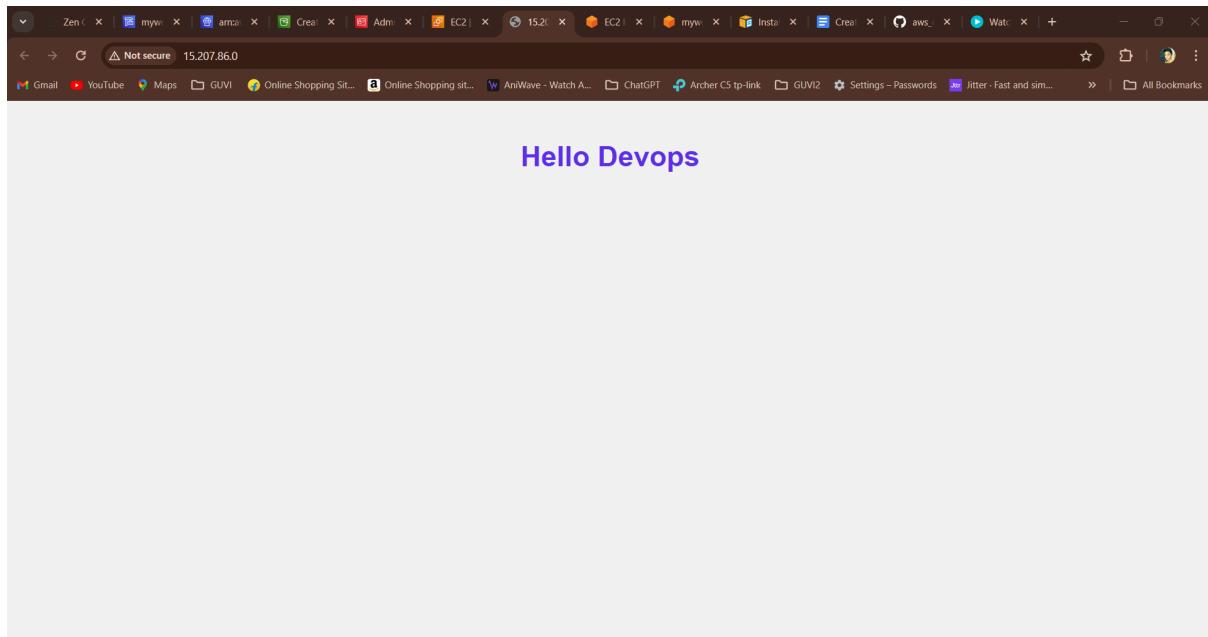
Deployment Completed Successfully.

The screenshot shows the AWS CodeDeploy console with a successful deployment status. The deployment ID is d-LVQ1CIS45, for application mywebapp, using configuration CodeDeployDefault.AllAtOnce. The deployment group is mywebapp-dg, initiated by User action. The revision location is s3://mycodebuilddemobucket/artifact.zip/artifact, created 9 minutes ago. The deployment lifecycle events show one instance (i-0322fa7a199ba4b16) successfully validating the service. The status is Succeeded.

Goto EC2 instance —> Click on the Public-Open address.

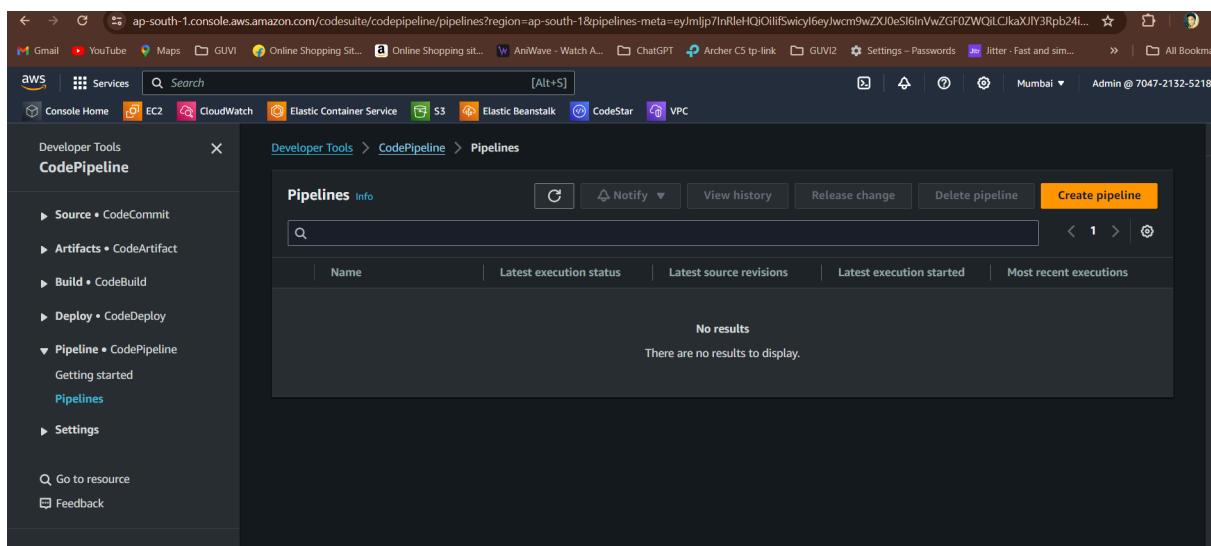
The screenshot shows the AWS EC2 Instances page with one instance listed: myec2-cicd (Instance ID i-0322fa7a199ba4b16). The instance is running in the t2.micro instance type, located in the ap-south-1a availability zone, with a public IPv4 address of 15.207.86.0. The instance summary shows it has passed 2/2 checks and is running.

Output:

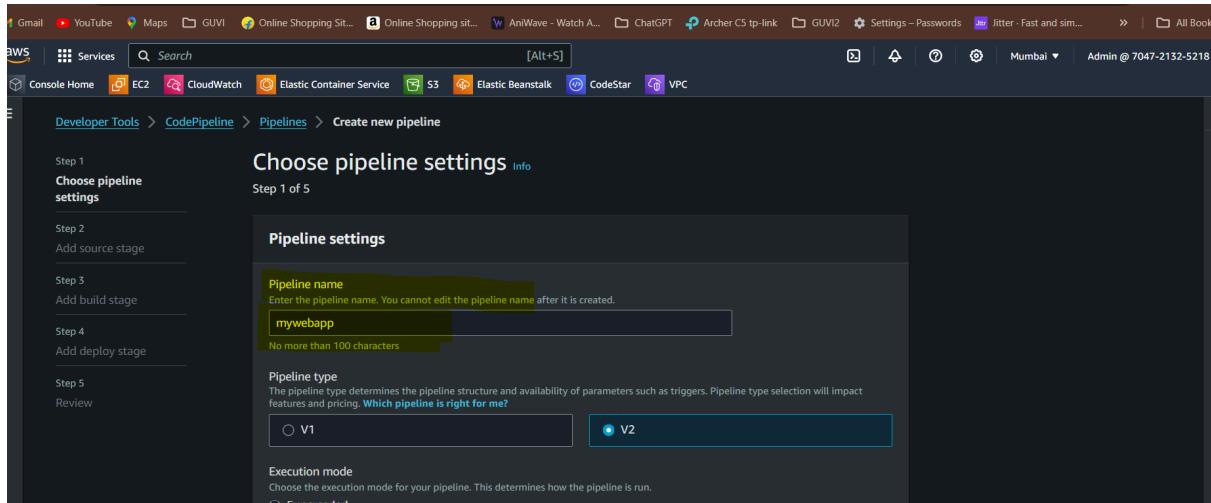


Note: Make sure the port no. is 80 in security —> Security group (If not, then add inbound rule 80 port with HTTP)

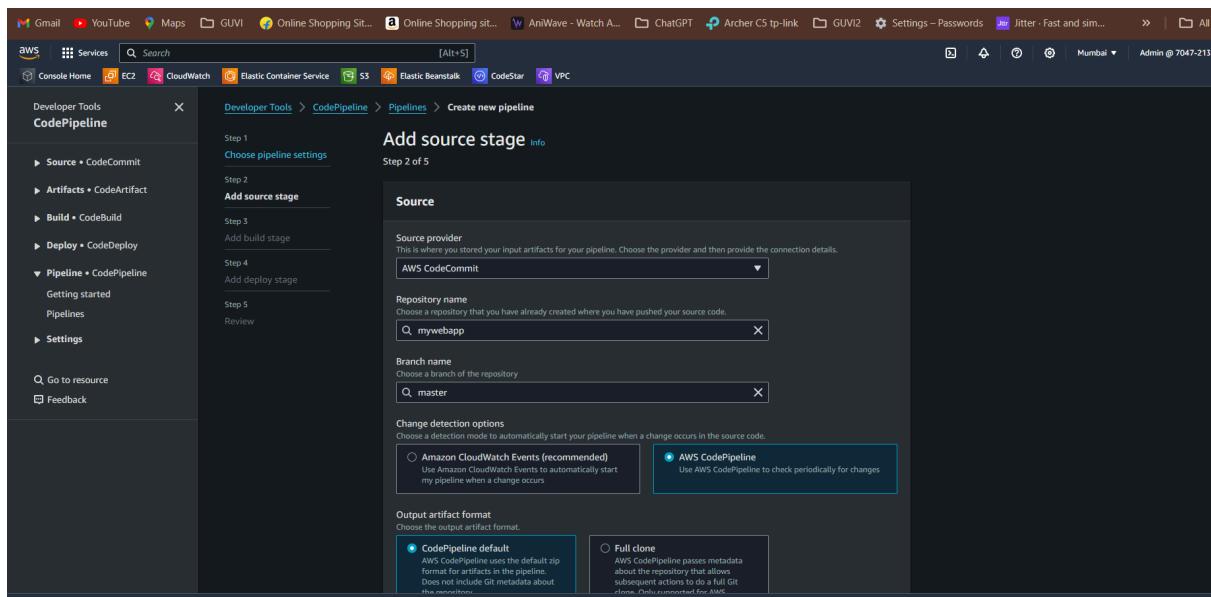
Step 1: Create a Pipeline so that any changes to your main code will automatically be saved and updated - (PI).



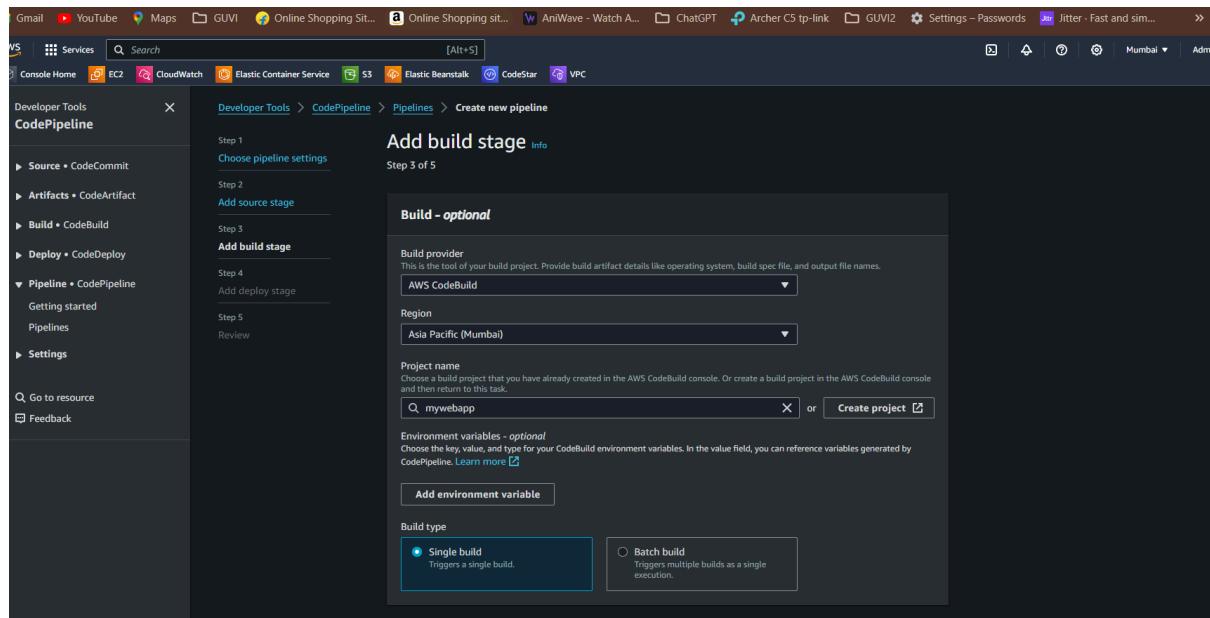
Step 2: Just give the Pipeline name —> next - (PI).



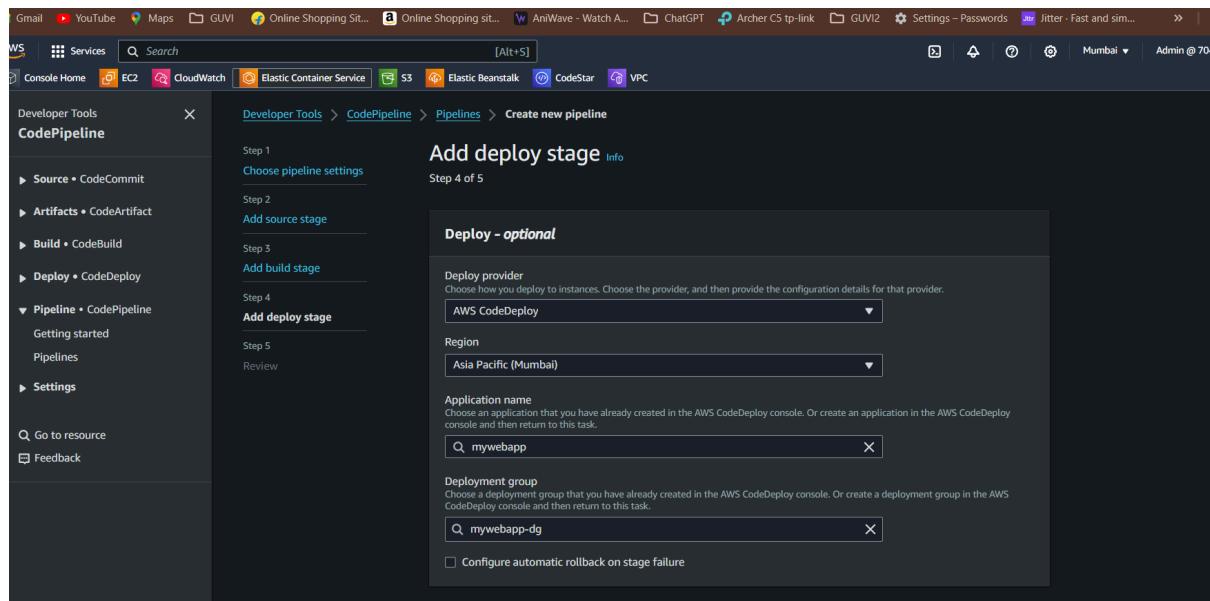
Step 3: Filled as per below, default info that is already created —> Click next.



Step 4: Fill the info as per below —> Click next.



Step 5: Fill the info as per below —> Click next —> Click Create Piplile.



Pipline created successfully.

The screenshot shows the AWS CodePipeline console with a success message: "Congratulations! The pipeline mywebapp has been created." The pipeline details are displayed, including the pipeline type (V2), execution mode (QUEUED), and a successful Source stage (AWS CodeCommit). The pipeline ID is cd178f77-dd79-4e28-a90c-e4ac61d7426a. A sidebar on the left shows navigation options like Source, Artifacts, Build, Deploy, Pipeline, Pipelines, Pipeline, History, Settings, and Go to resource. A feedback link is also present. The bottom right corner includes copyright information: © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences.

Pipeline Output:

The screenshot shows a browser window displaying the output of the pipeline. The URL is 15.207.86.0. The page content is a simple "Hello Devops and AWS DO12WD". The browser's address bar shows "△ Not secure" and the IP address. The top of the browser window has a standard toolbar with various icons and tabs.