

Advance Devops Case Study Report

Cloud Deployment with Automation

Concepts Used:

- AWS CodePipeline
- EC2
- S3 Problem

Statement:

Build a simple web application using AWS CodeBuild and deploy it to an S3 bucket. Then, automate the deployment process using AWS CodePipeline, ensuring the application is deployed on an EC2 instance. A sample index.html page will be used for demonstration.

Tasks:

1. Set up AWS CodeBuild for the web app.
2. Create a pipeline that deploys the web app to an S3 bucket.
3. Use AWS CodeDeploy to push updates to an EC2 instance.

1. Introduction Case Study Overview:

This case study focuses on building a simple web application and automating its deployment using a combination of AWS services— AWS CodeBuild, S3, CodePipeline, and CodeDeploy. The task is to create a basic application, package it, and deploy it to an S3 bucket as the initial step. Afterward, AWS CodePipeline is used to automate the deployment, ensuring updates are automatically pushed to an EC2 instance using AWS CodeDeploy. The goal is to demonstrate a seamless, automated deployment pipeline for a web application, which simplifies continuous integration and delivery (CI/CD) processes.

Key Feature and Application:

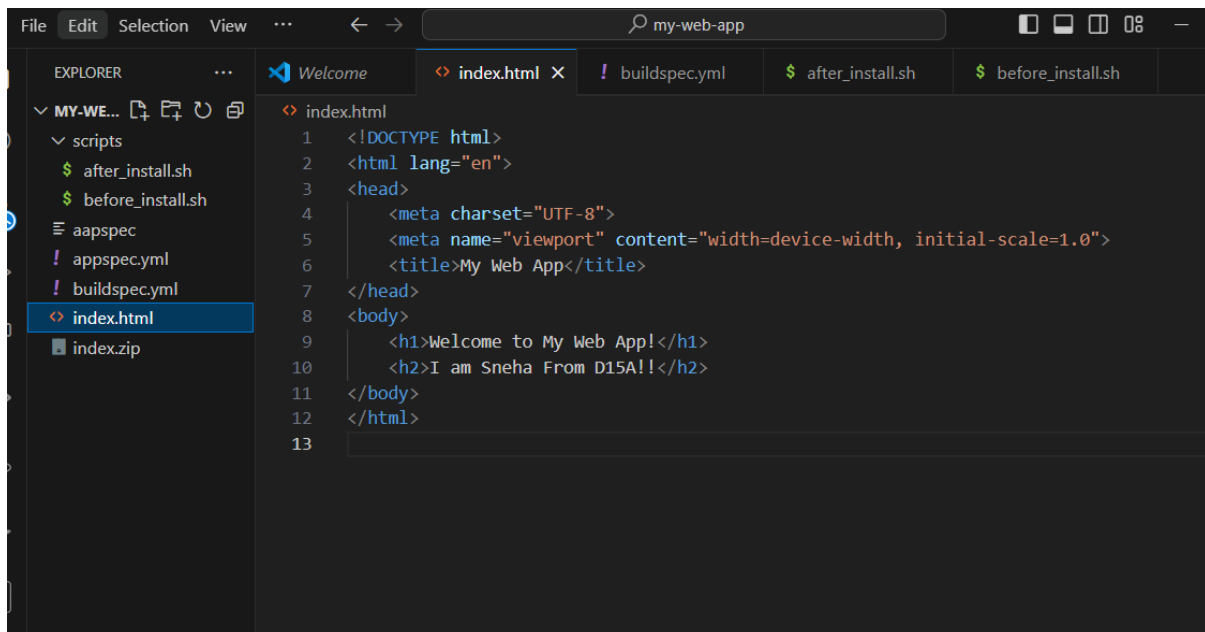
The unique feature of this case study is the integration of multiple AWS services to create a fully automated deployment pipeline. By using AWS

CodeBuild for compiling and packaging the web app, AWS S3 as the storage for the deployed app, and AWS CodePipeline for automating the process, the deployment becomes efficient and scalable. Additionally, CodeDeploy ensures that the web application can be easily pushed and updated on an EC2 instance, facilitating fast iteration and real-time updates to the deployed application. This automation greatly reduces the manual workload involved in deployment and allows for continuous delivery, making it highly practical for modern web applications that need frequent updates.

PROCEDURE & SCREENSHOTS:

Create a Simple Web App

1. First, create a simple web app with an index.html file:

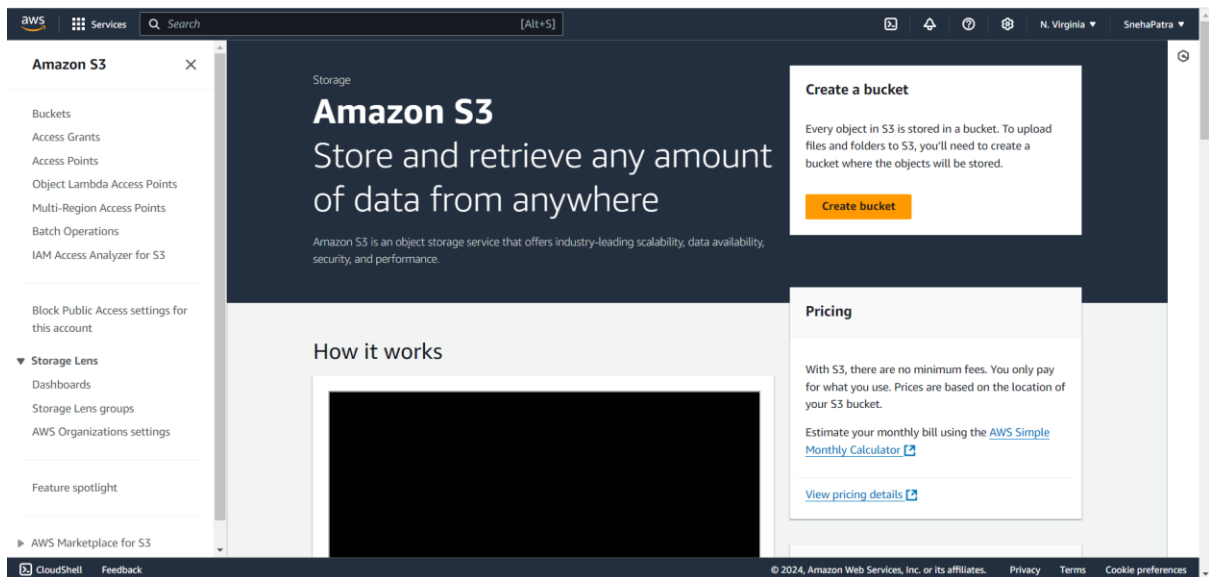
A screenshot of a code editor interface. The Explorer panel on the left shows a project named 'MY-WE...' with a 'scripts' folder containing 'after_install.sh' and 'before_install.sh', and a file named 'index.html' which is selected. The main editor area shows the content of 'index.html' with the following code:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>My Web App</title>
7 </head>
8 <body>
9   <h1>Welcome to My Web App!</h1>
10  <h2>I am Sneha From D15A!!</h2>
11 </body>
12 </html>
13
```

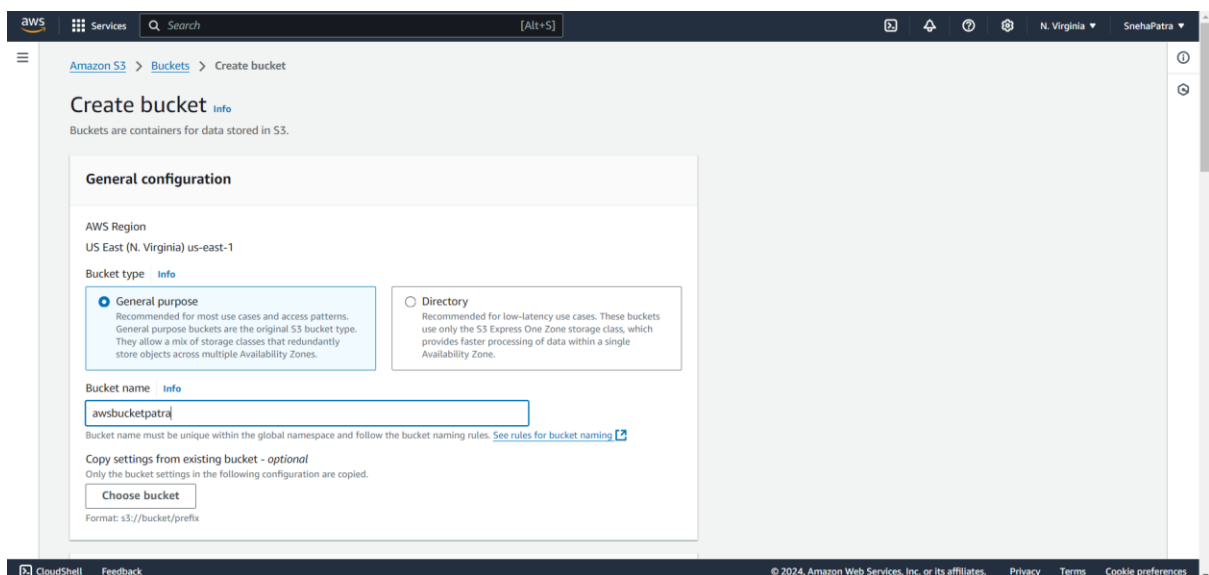
This file will serve as the web page deployed to your S3 bucket and later to the EC2 instance.

2. Set Up S3 Bucket for Web App Hosting

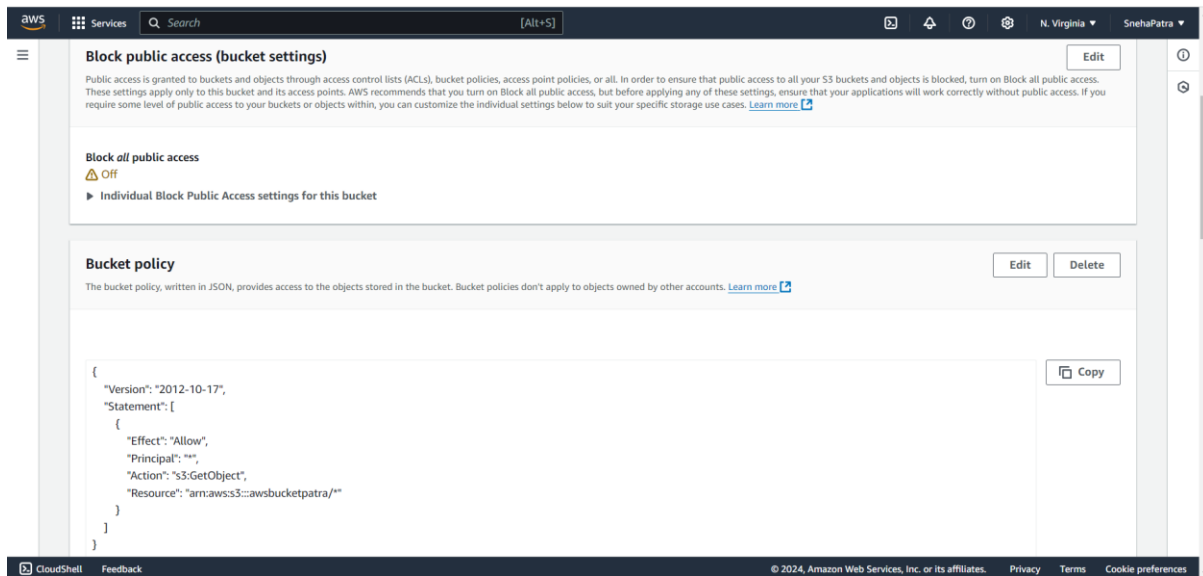
1. **Go to the AWS S3 Console:**
 - Open the [S3 console](#).



- Create a new S3 bucket, giving it a unique name (e.g., my-s3-web-bucket).

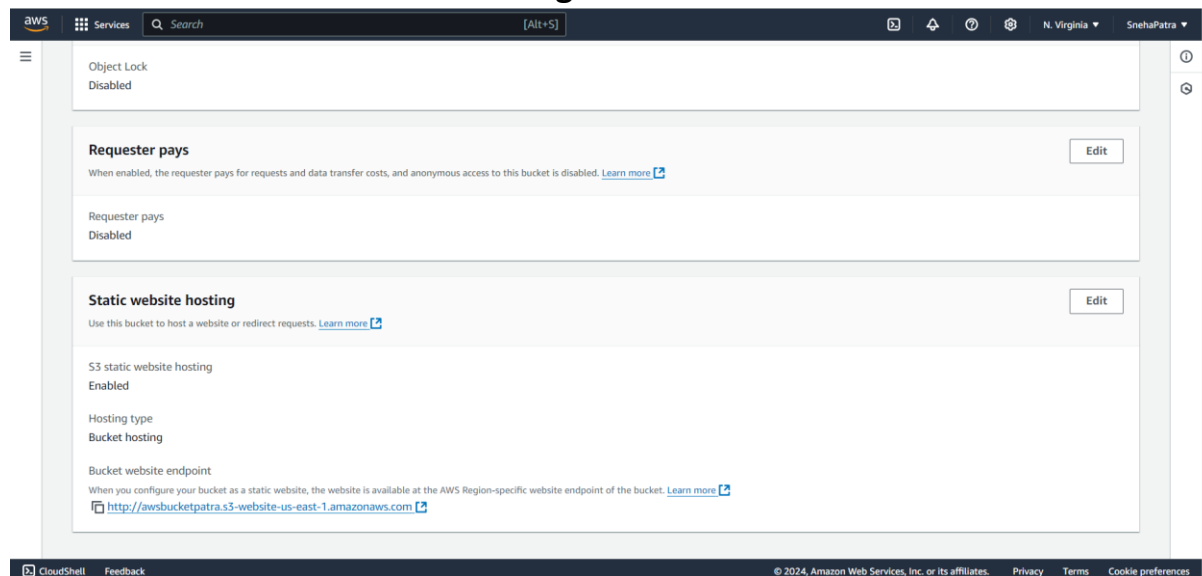


- Under **Permissions**, uncheck the "Block all public access" option, allowing public access for web hosting.



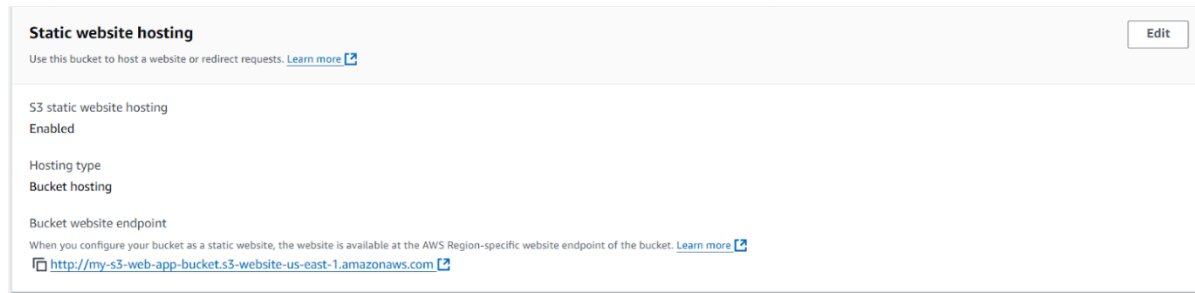
2. Configure the Bucket for Website Hosting:

- Go to the **Properties** tab of your S3 bucket.
- Scroll down to **Static website hosting**.



- Enable it, and set the **Index document** as index.html.

- Copy the bucket website URL for testing the web app later.

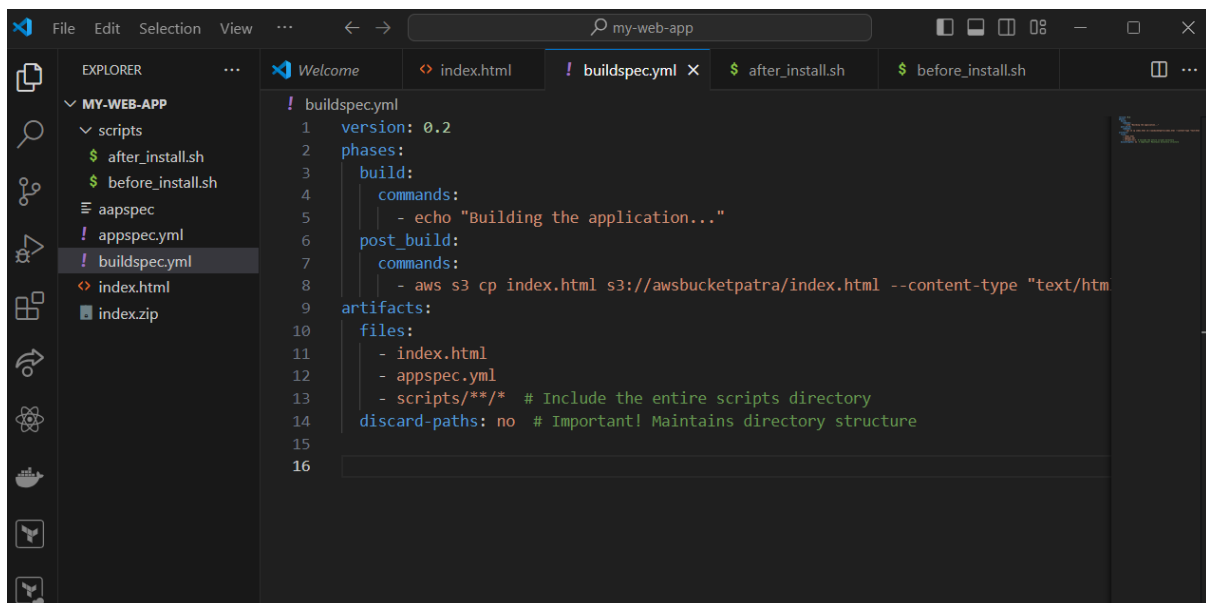


in my case it is: <http://awsbucketpatra.s3-website-us-east-1.amazonaws.com>

- if s3 website shows 403 forbidden, its a IAM permission issue.

3. Set Up CodeBuild for Your Web App

Create a Buildspec File: In your project directory (where index.html resides), create a buildspec.yml file. This file tells AWS CodeBuild what to do during the build.



Go to AWS CodeBuild:

- Open the [AWS CodeBuild console](#).
- Create a new build project.
- For **Source**, choose your source repository (e.g., GitHub, Bitbucket, or S3).

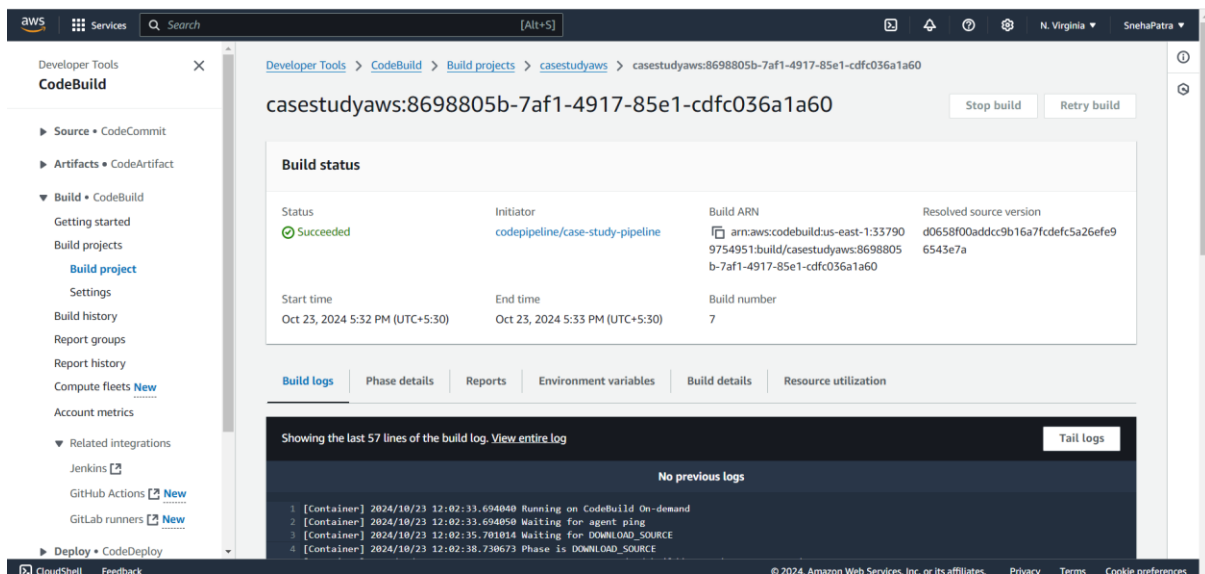
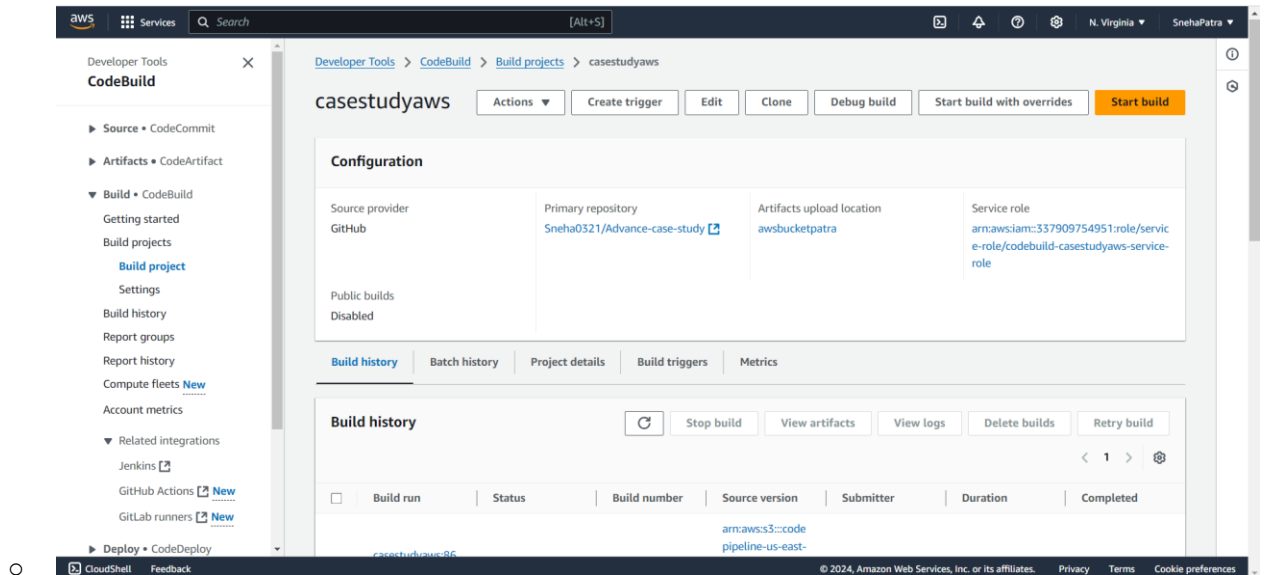
-
- specify the buildspec.yml file you created. before this add it to your git repo
- Set **Artifacts** to "S3", and choose the bucket you created earlier.

The screenshot shows the AWS Artifacts console interface. At the top, there's a navigation bar with the AWS logo, 'Services' link, search icon, and user profile 'SnehaPatra'. The main content area is titled 'Artifacts' with an 'Add artifact' button. Below this, 'Artifact 1 - Primary' is selected. The configuration fields are as follows:

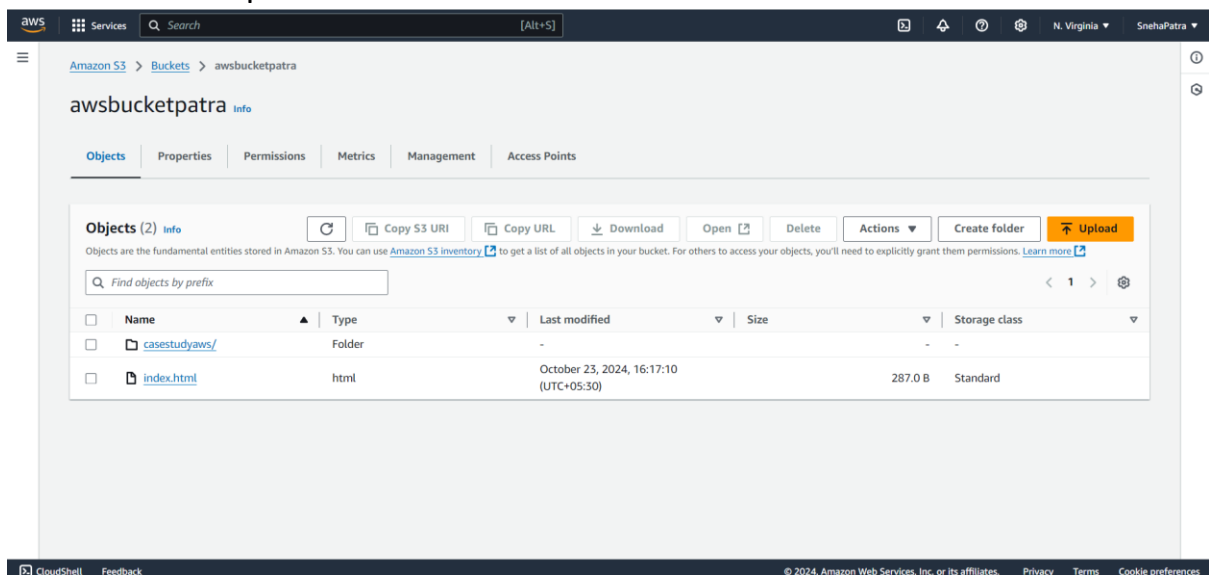
- Type:** A dropdown menu set to 'Amazon S3'. A note below states: 'You might choose no artifacts if you are running tests or pushing a Docker image to Amazon ECR.'
- Bucket name:** A search bar containing 'awsbucketpatra' with a magnifying glass icon and a close 'X' button.
- Name:** A text input field. Below it, a note says: 'The name of the folder or compressed file in the bucket that will contain your output artifacts. Use Artifacts packaging under Additional configuration to choose whether to use a folder or compressed file. If the name is not provided, defaults to project name.'
- Enable semantic versioning:** An unchecked checkbox with the label 'Enable semantic versioning' and a sub-note: 'Use the artifact name specified in the buildspec file'.
- Path - optional:** A text input field. A note below says: 'The path to the build output ZIP file or folder.' An example is provided: 'Example: MyPath/MyArtifact.zip.'
- Namespace type - optional:** This label is partially visible at the bottom of the form.

The footer of the console includes 'CloudShell', 'Feedback', 'Privacy', 'Terms', and 'Cookie preferences' links, along with a copyright notice: '© 2024 Amazon Web Services, Inc. or its affiliates'.

- Create the build project and start the build to ensure it uploads index.html to the S3 bucket.



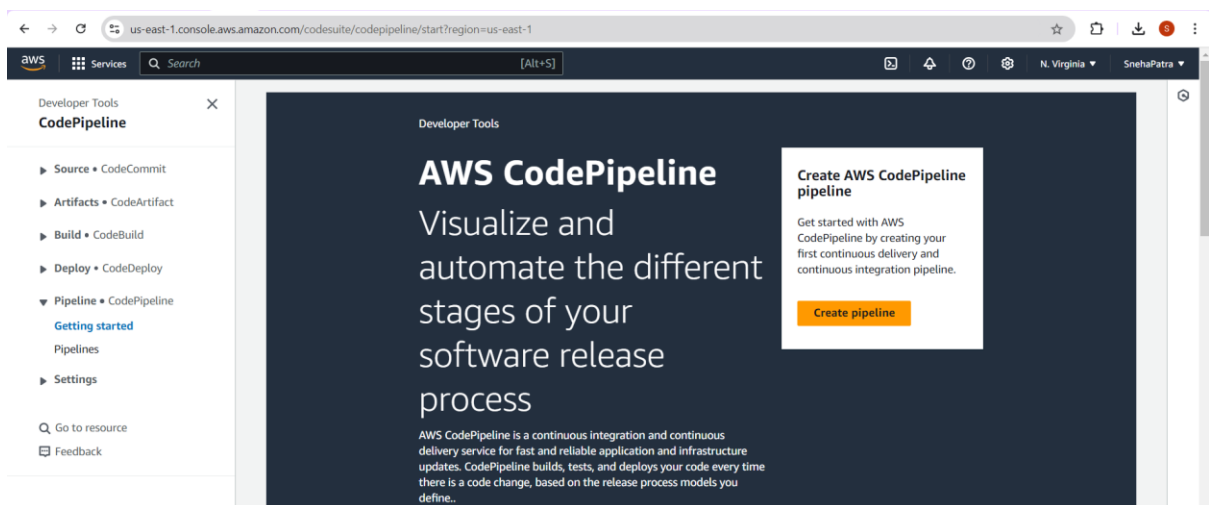
s3 bucket was updated:



4. Set Up AWS CodePipeline

1. Go to AWS CodePipeline:

- Open the [CodePipeline console](#).



- Create a new pipeline.

- For **Source**:

us-east-1.console.aws.amazon.com/codesuite/codepipeline/pipelin...

aws Services 🔍 ⓘ ⚙️ ? N. Virginia ▼ SnehaPatra ▼

Choose source [Info](#)

Step 3 of 4

Source

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

GitHub (Version 2) ▼

New GitHub version 2 (app-based) action

To add a GitHub version 2 action in CodePipeline, you create a connection, which uses GitHub Apps to access your repository. Use the options below to choose an existing connection or create a new one. [Learn more](#)

Connection
Choose an existing connection that you have already configured, or create a new one and then return to this task.

arn:aws:codeconnections:eu-north-1:33790! ✕ or [Connect to GitHub](#)

Repository name
Choose a repository in your GitHub account.

Sneha0321/Advance-case-study ✕

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

CloudShell Feedback Privacy Terms Cookie preferences

© 2024, Amazon Web Services, Inc. or its affiliates.

- Select your repository (e.g., GitHub, Bitbucket).
- Connect and choose the appropriate branch where the index.html and buildspec.yml files are.

2. Add Build Stage:

- In the **Build stage**, choose **AWS CodeBuild** as the build provider.

Build - optional

Build provider
Choose the tool you want to use to run build commands and specify artifacts for your build action.

☐ Commands ☒ Other build providers

AWS CodeBuild ▼

- Select the CodeBuild project you created earlier.

aws Services 🔍 📄 🔔 ⓘ ⚙️ N. Virginia ▼ SnehaPatra ▼

☰ **Add build stage** Info

Step 4 of 6

Build - optional

Build provider
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.

🔍 casestudyaws ✕ 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.

📄 CloudShell Feedback Privacy Terms Cookie preferences

© 2024, Amazon Web Services, Inc. or its affiliates.

3. Deploy to S3:

- In the next stage, choose **Deploy**. Select **Amazon S3**. Choose your S3 bucket (my-s3-bucket) where the index.html file will be

deployed.

The screenshot shows the AWS CloudShell interface with the 'Add deploy stage' configuration page. The page is titled 'Add deploy stage' and is part of a 6-step process. The left sidebar shows the steps: 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), and Step 6 (Review). The main content area is for Step 5, 'Add deploy stage'. It has a sub-header 'Deploy - optional' and a description: 'Choose how you deploy to instances. Choose the provider, and then provide the configuration details for that provider.' The configuration fields are: 'Deploy provider' (set to 'Amazon S3'), 'Region' (set to 'US East (N. Virginia)'), 'Input artifacts' (set to 'BuildArtifact'), 'Bucket' (set to 'awsbucketpatra'), and 'S3 object key' (set to 'casestudyaws/index.html'). The 'BuildArtifact' is defined by 'Build' and has a limit of 'No more than 100 characters'. The 'S3 object key' has a note: 'Enter the object key. You can include a file path without the delimiter character (/) at the beginning. Include the file extension. Example: SampleObject.zip'.

4. Test the Pipeline:

- Once the pipeline is set up, click **Release Change** to start the pipeline. This should fetch the latest code, build it, and upload index.html to the S3 bucket.

Developer Tools

CodePipeline

Source • CodeCommit

Artifacts • CodeArtifact

Build • CodeBuild

Deploy • CodeDeploy

Pipeline • CodePipeline

Getting started

Pipelines

Pipeline

History

Settings

Settings

Go to resource

Feedback

Success

Congratulations! The pipeline case-study-pipeline has been created.

Create a notification rule for this pipeline

Developer Tools

CodePipeline

Pipelines

case-study-pipeline

case-study-pipeline

Notify

Edit

Stop execution

Clone pipeline

Release change

Pipeline type: V2

Execution mode: QUEUED

Source

Succeeded

Pipeline execution ID: 063ade0b-9de6-4a9f-a6fd-360546dad263

Source

GitHub (Version 2)

Succeeded - 1 minute ago

0f16b85a

View details

0f16b85a Source: Add files via upload

Disable transition

✓

✓

✓

CloudShell

Feedback

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

Developer Tools

CodePipeline

Source • CodeCommit

Artifacts • CodeArtifact

Build • CodeBuild

Deploy • CodeDeploy

Pipeline • CodePipeline

Getting started

Pipelines

Pipeline

History

Settings

Settings

Go to resource

Feedback

Build

Succeeded

Start rollback

Pipeline execution ID: 063ade0b-9de6-4a9f-a6fd-360546dad263

Build

AWS CodeBuild

Succeeded - Just now

View details

0f16b85a Source: Add files via upload

Disable transition

Deploy

Succeeded

Start rollback

Pipeline execution ID: 063ade0b-9de6-4a9f-a6fd-360546dad263

Deploy

Amazon S3

Succeeded - Just now

View details

✓

✓

✓

CloudShell

Feedback

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

- Visit the S3 bucket's website URL to verify that the index.html page is live.

← → ↻ ⚠ Not secure awsbucketpatra.s3-website-us-east-1.amazonaws.com

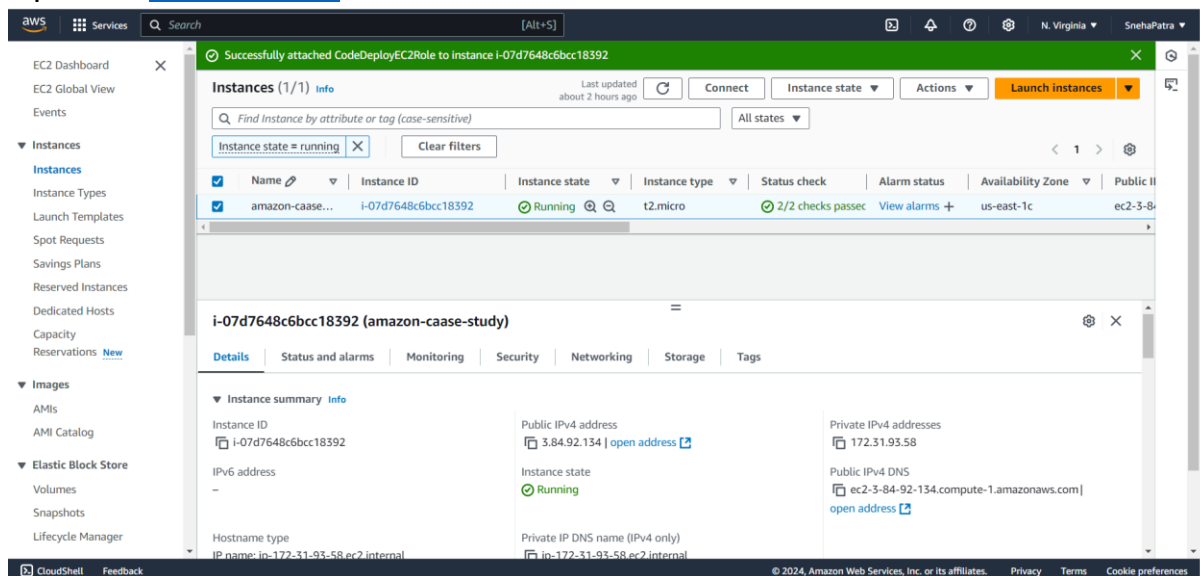
Welcome to My Web App!

I am Sneha From D15A!!

5. Set Up EC2 Instance for Web Hosting

1. Launch an EC2 Instance:

- Open the [EC2 console](#).



- Launch a new instance, selecting an Amazon Linux 2 AMI.

- Choose the default t2.micro instance type.
- Configure instance settings and storage (use defaults for now).
- In **Configure Security Group**, allow HTTP traffic by adding a rule to open port 80.

2. Connect to the EC2 Instance:

- Once the instance is running, connect via SSH.

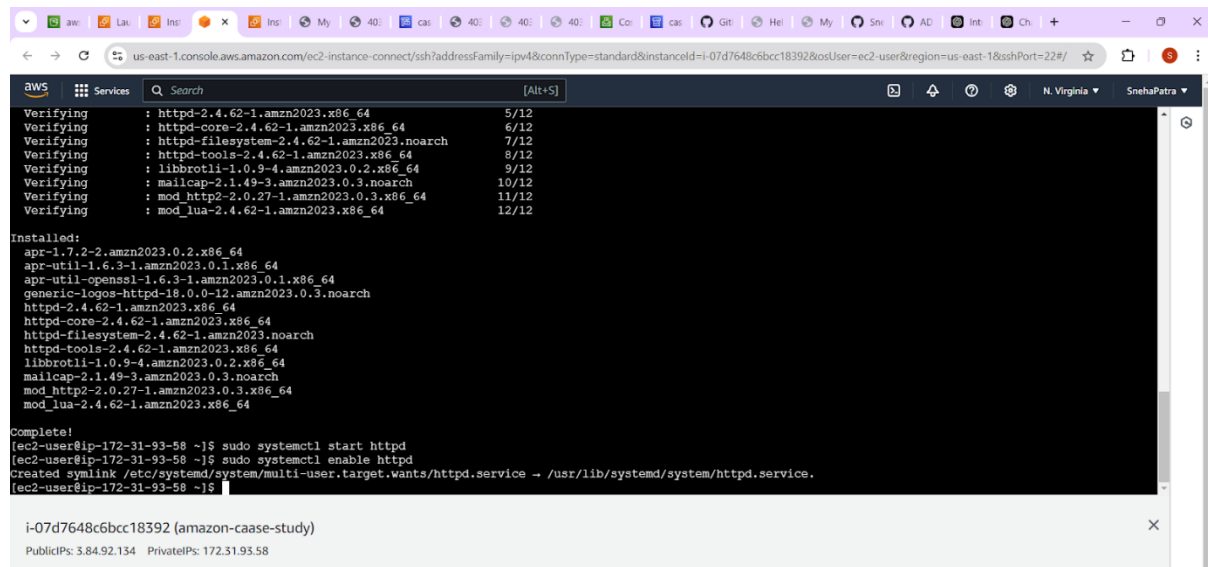
Install the required web server (Apache) on your instance:

```
sudo yum update -y
```

```
sudo yum install httpd -y
```

```
sudo systemctl start httpd
```

```
sudo systemctl enable httpd
```



```

aws
Services
Search [Alt+S]
N. Virginia SnehPatra

Verifying : httpd-2.4.62-1.amzn2023.x86_64 5/12
Verifying : httpd-core-2.4.62-1.amzn2023.x86_64 6/12
Verifying : httpd-filesystem-2.4.62-1.amzn2023.noarch 7/12
Verifying : httpd-tools-2.4.62-1.amzn2023.x86_64 8/12
Verifying : libbrotli-1.0.9-4.amzn2023.0.2.x86_64 9/12
Verifying : mailcap-2.1.49-3.amzn2023.0.3.noarch 10/12
Verifying : mod_http2-2.0.27-1.amzn2023.0.3.x86_64 11/12
Verifying : mod_lua-2.4.62-1.amzn2023.x86_64 12/12

Installed:
apr-1.7.2-2.amzn2023.0.2.x86_64
apr-util-1.6.3-1.amzn2023.0.1.x86_64
apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64
generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch
httpd-2.4.62-1.amzn2023.x86_64
httpd-core-2.4.62-1.amzn2023.x86_64
httpd-filesystem-2.4.62-1.amzn2023.noarch
httpd-tools-2.4.62-1.amzn2023.x86_64
libbrotli-1.0.9-4.amzn2023.0.2.x86_64
mailcap-2.1.49-3.amzn2023.0.3.noarch
mod_http2-2.0.27-1.amzn2023.0.3.x86_64
mod_lua-2.4.62-1.amzn2023.x86_64

Complete!
[ec2-user@ip-172-31-93-58 ~]$ sudo systemctl start httpd
[ec2-user@ip-172-31-93-58 ~]$ sudo systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
[ec2-user@ip-172-31-93-58 ~]$

i-07d7648c6bcc18392 (amazon-caase-study)
PublicIPs: 3.84.92.134 PrivateIPs: 172.31.93.58

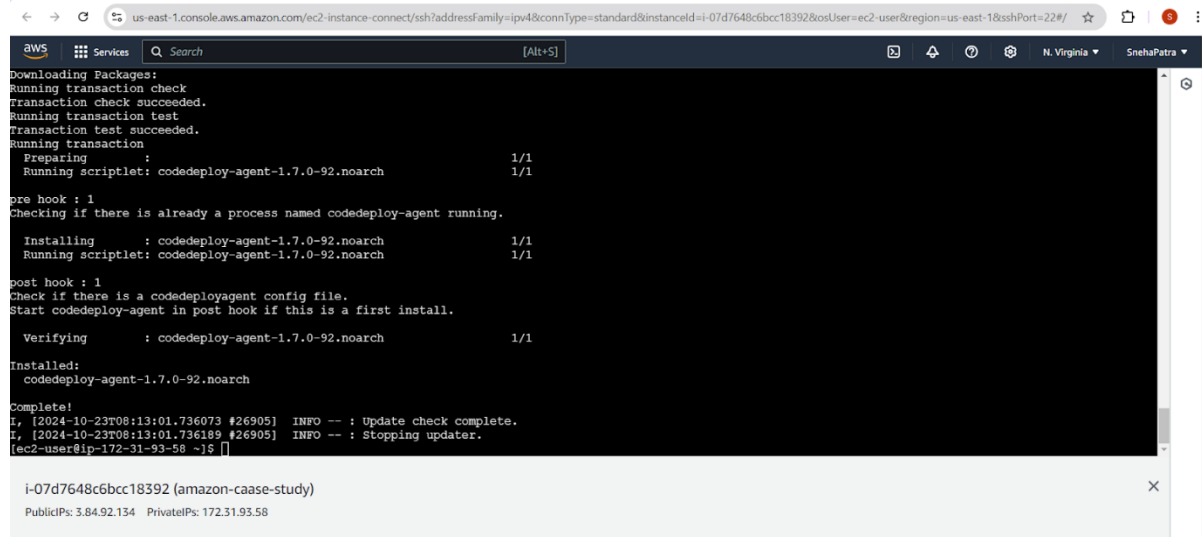
```

6. Set Up AWS CodeDeploy to Push Updates to EC2

1. Install CodeDeploy Agent on EC2:

- Connect to the EC2 instance and Install the CodeDeploy agent:

```
sudo yum update -y
sudo yum install -y ruby wget
wget https://aws-codedeploy-us-east-1.s3.amazonaws.com/latest/install
chmod +x ./install
sudo ./install auto
```



The screenshot shows a terminal window within the AWS console, connected to an EC2 instance. The terminal output displays the steps for installing the CodeDeploy agent: downloading packages, running transaction checks, preparing the scriptlet, and installing the agent. The process completes successfully, and the terminal shows the agent is installed and ready for use. The AWS console interface at the top shows the 'Services' tab and the instance details for 'i-07d7648c6bcc18392'.

```
Download Packages:
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing      : codedeploy-agent-1.7.0-92.noarch                1/1
  Running scriptlet: codedeploy-agent-1.7.0-92.noarch                1/1
pre hook : 1
Checking if there is already a process named codedeploy-agent running.

  Installing      : codedeploy-agent-1.7.0-92.noarch                1/1
  Running scriptlet: codedeploy-agent-1.7.0-92.noarch                1/1
post hook : 1
Check if there is a codedeployagent config file.
Start codedeploy-agent in post hook if this is a first install.

  Verifying       : codedeploy-agent-1.7.0-92.noarch                1/1
Installed:
codedeploy-agent-1.7.0-92.noarch

Complete!
1, [2024-10-23T08:13:01.736073 #26905] INFO -- : Update check complete.
1, [2024-10-23T08:13:01.736189 #26905] INFO -- : Stopping updater.
[ec2-user@ip-172-31-93-50 ~]$
```

2. Set Up CodeDeploy Application:

3.1 Create Application

1. Using AWS CLI from Local Terminal on pc (not ec2, use your own pc's command line):

First ensure AWS CLI is installed

```
aws --version
```

if not installed, install from <https://awscli.amazonaws.com/AWSCLIV2.msi>

```
aws configure
```

2. Create application:

```
aws deploy create-application --application-name my-webapp
```

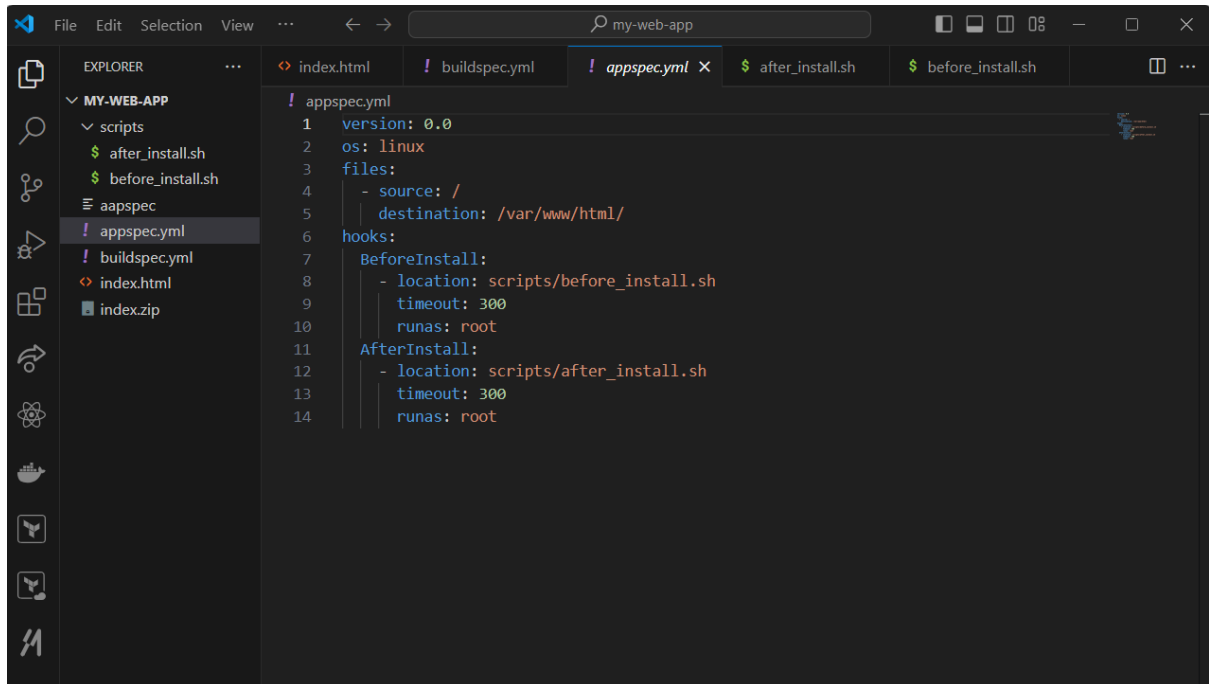
3.2 Create Deployment Group

- o Create deployment group:

```
aws deploy create-deployment-group --application-name my-webapp --  
deployment-group-name my-webapp-group --service-role-arn  
arn:aws:iam::ACCOUNT_ID:role/CodeDeployServiceRole
```

- enter your ACCOUNT_ID above!!

Create appspec.yml for CodeDeploy: In your project folder, create an appspec.yml file to specify how CodeDeploy should handle the deployment:

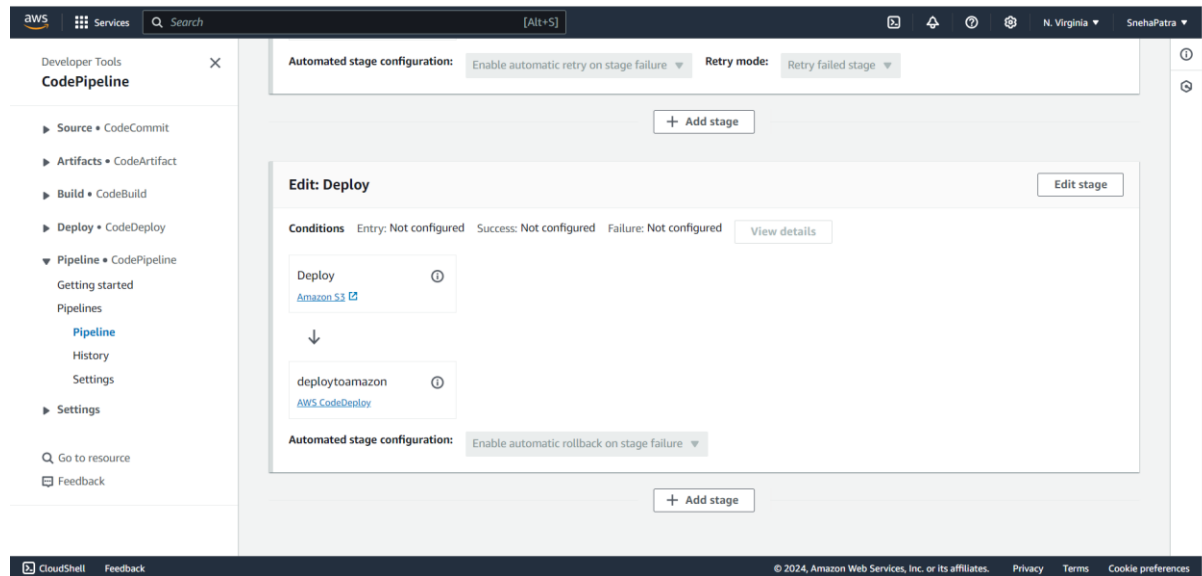


Now create a scripts folder inside your repo which will contain 2 files:
before_install.sh and after_install.sh

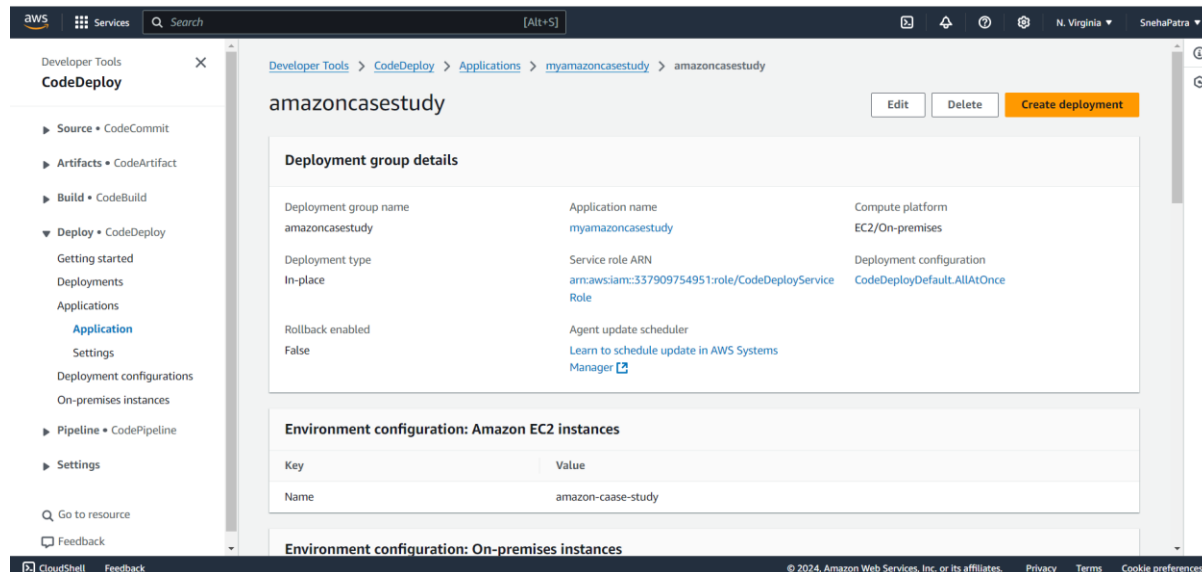
3. Add Deployment Stage to CodePipeline:

- Go back to your CodePipeline.

- Add a new stage for deployment.

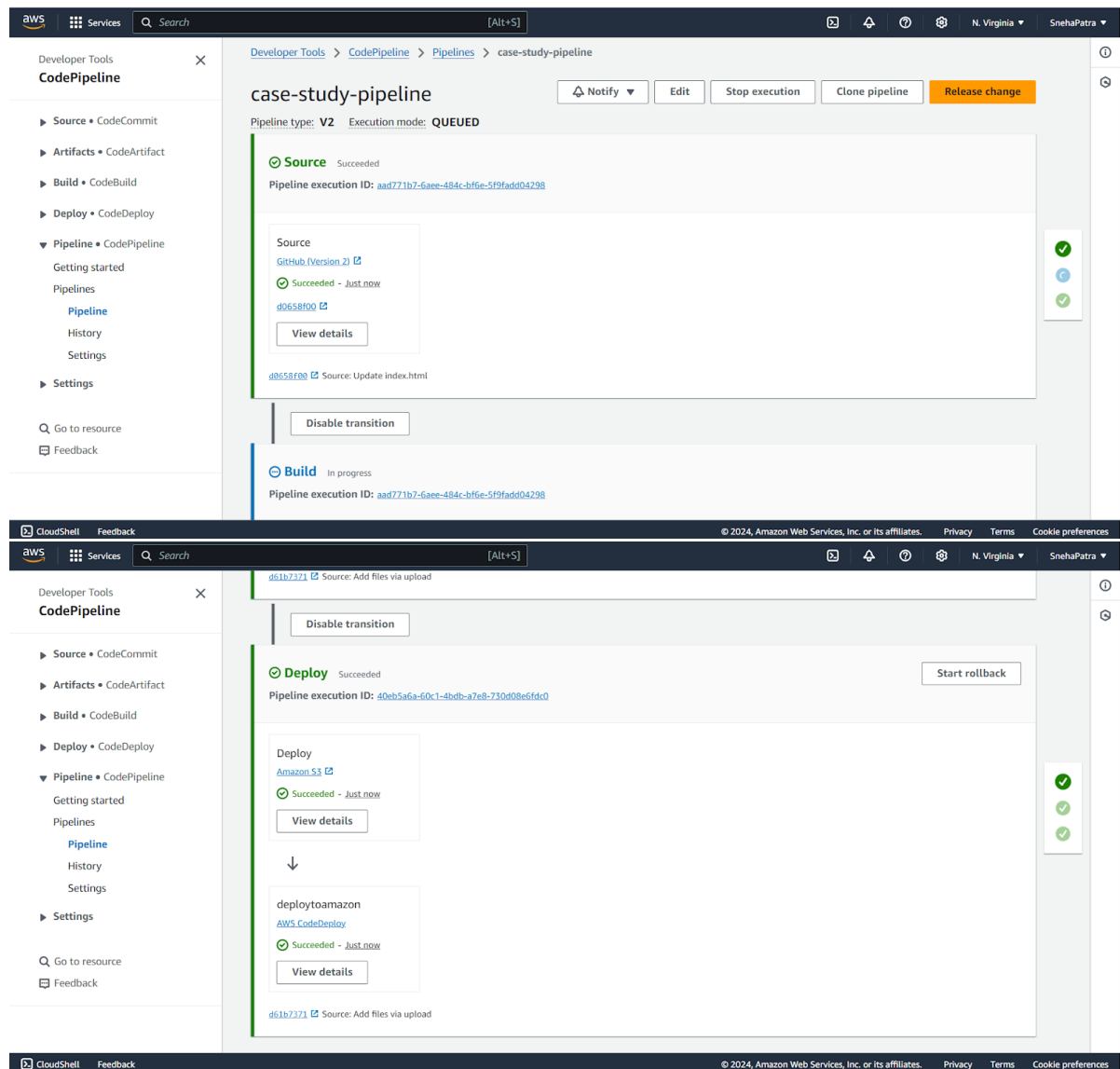


- Select **AWS CodeDeploy** and choose the application and deployment group you created earlier.



4. Test Deployment:

- Make a change to index.html in your source repository and push it.
- This should trigger the pipeline, rebuild the app, push it to S3, and deploy it to the EC2 instance.



- You can access your EC2 instance via its public IP to view the updated web app.

7. Verify Automation

- Now that your pipeline is set up, any changes to your repository (e.g., modifying the index.html file) should automatically trigger the build, deploy it to S3, and push updates to your EC2 instance.



CHALLENGES FACED:

1. **Permission Issues:** One of the main hurdles was setting the correct permissions in IAM roles and the S3 bucket policy. Errors like 403 Forbidden when accessing S3 were resolved by adjusting the S3 bucket policy.
2. **Missing AppSpec File:** The CodeDeploy process failed initially because the appspec.yml file was not placed correctly in the root directory of the build artifacts.
3. **CodeDeploy Agent Issues:** The EC2 instance had issues with the CodeDeploy agent, such as failing to start or showing Permission Denied errors. These were resolved by restarting the agent and ensuring the instance had appropriate permissions to access S3.

Despite these challenges, the overall deployment was successful, demonstrating the effectiveness of AWS's automation tools in managing continuous deployment workflows.

CONCLUSION:

This experiment successfully demonstrated the process of building and deploying a simple web application using AWS CodePipeline, CodeBuild, CodeDeploy, and EC2. The pipeline automates the workflow from the moment the code is pushed to a repository until the application is deployed on an EC2 instance.