Dockerizing a PHP Application and Deploying on AWS

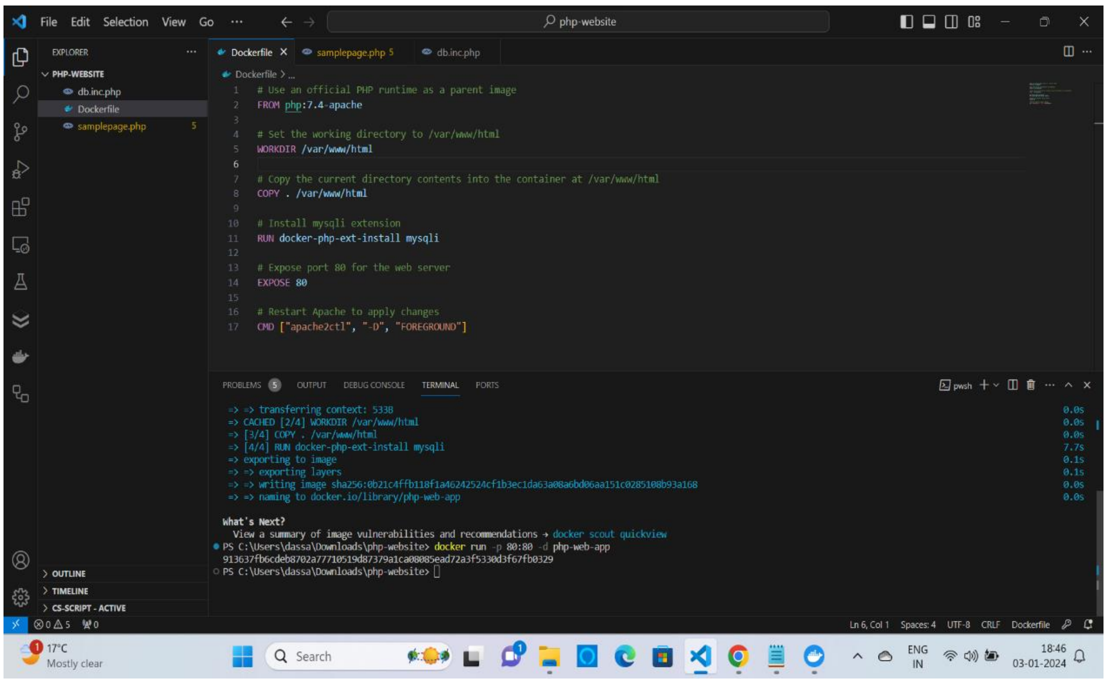
1. Setting up the PHP Application and Dockerizing it

1.1 **Choose a PHP Application:**

- Go to GitHub and find a simple PHP application.

1.2 **Create Dockerfile:**

- Create a `Dockerfile` in the project root with the content mentioned in the initial instructions.
- Adjust the Dockerfile based on application's specific requirements.

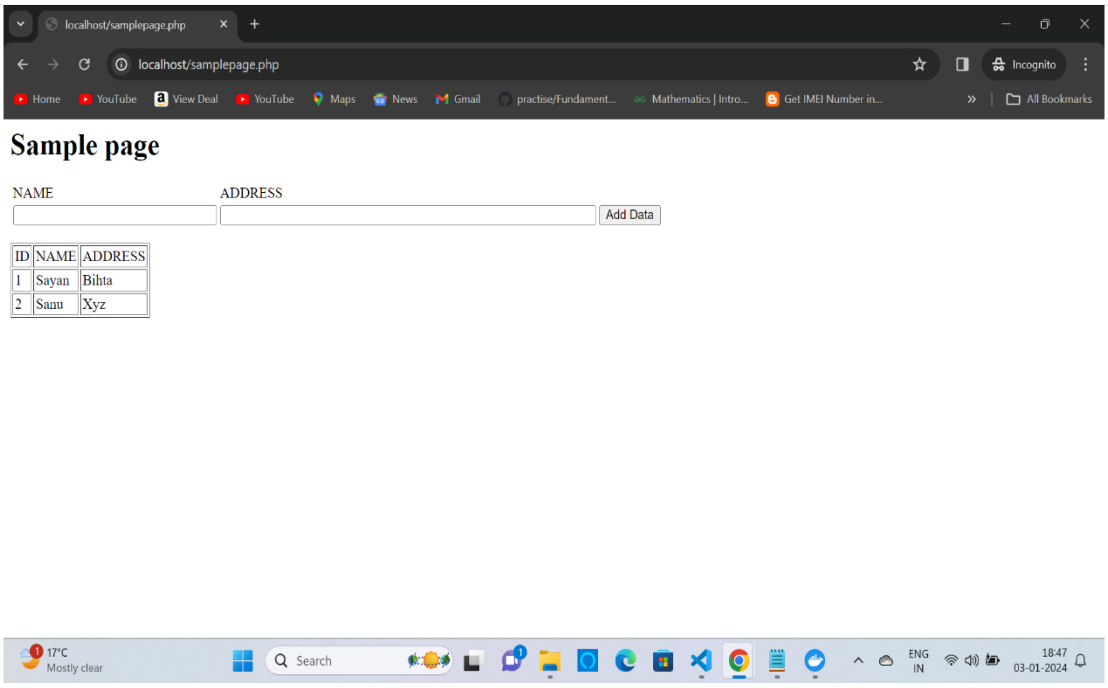


```dockerfile
# Use an official PHP runtime as a parent image
FROM php:7.4-apache

# Set the working directory to /var/www/html
WORKDIR /var/www/html

# Copy the current directory contents into the container at /var/www/html
COPY . /var/www/html

# Install mysqli extension
RUN docker-php-ext-install mysqli

# Expose port 80 for the web server
EXPOSE 80

# Restart Apache to apply changes
CMD ["apache2ctl", "-D", "FOREGROUND"]
```

```
 => => transferring context: 533B                                                              0.0s
 => CACHED [2/4] WORKDIR /var/www/html                                                         0.0s
 => [3/4] COPY . /var/www/html                                                                 0.0s
 => [4/4] RUN docker-php-ext-install mysqli                                                    7.7s
 => exporting to image                                                                         0.1s
 => => exporting layers                                                                        0.1s
 => => writing image sha256:0b21c4ffb118f1a46242524cf1b3ec1da63a08a6bd06aa151c0285108b93a168  0.0s
 => => naming to docker.io/library/php-web-app                                                 0.0s

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\Users\dassa\Downloads\php-website> docker run -p 80:80 -d php-web-app
913637fb6cdeb8702a77710519d87379a1ca08085ead72a3f5330d3f67fb0329
PS C:\Users\dassa\Downloads\php-website>
```

1.3 **Build and Test Docker Image Locally:**

- Access `http://localhost` to ensure the PHP application is running.



localhost/samplepage.php

Sample page

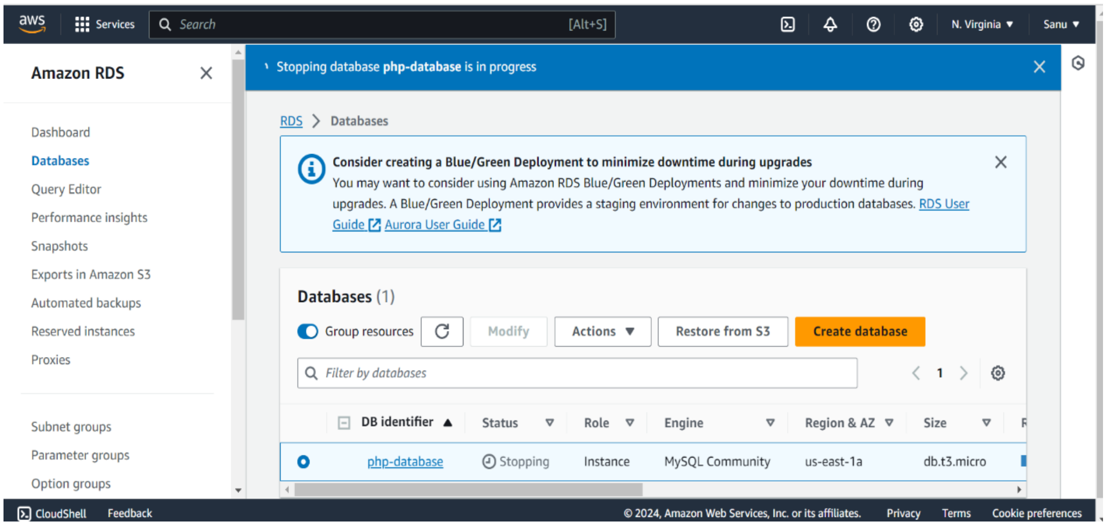
NAME ADDRESS Add Data

ID	NAME	ADDRESS
1	Sayan	Bihta
2	Sanu	Xyz

2. Configuring AWS RDS and Updating the PHP Application

2.1 **Create MySQL Database on AWS RDS:**

- Follow AWS documentation to create an RDS instance and configure a MySQL database.



Stopping database php-database is in progress

RDS > Databases

Consider creating a Blue/Green Deployment to minimize downtime during upgrades
You may want to consider using Amazon RDS Blue/Green Deployments and minimize your downtime during upgrades. A Blue/Green Deployment provides a staging environment for changes to production databases. RDS User Guide Aurora User Guide

Databases (1)

DB identifier	Status	Role	Engine	Region & AZ	Size
php-database	Stopping	Instance	MySQL Community	us-east-1a	db.t3.micro

2.2 **Update PHP Application Configuration:**

- Modify the PHP application configuration to connect to the newly created AWS RDS database.

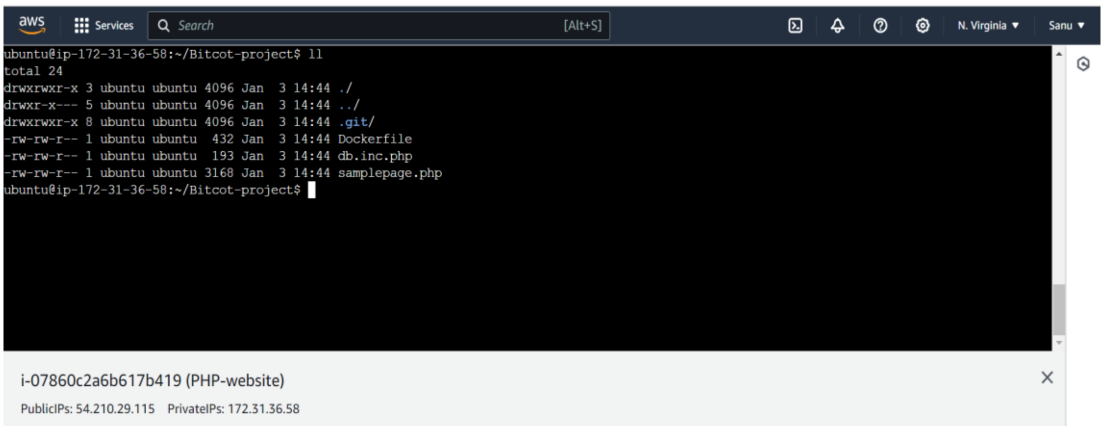
3. Manually Deploying to an EC2 Instance

3.1 Create an EC2 Instance and Install Docker:

- Launch an EC2 instance and install Docker.

3.2 Transfer Files and Build Docker Image:

- Transfer PHP application files and Dockerfile to the EC2 instance.
- SSH into the EC2 instance and build the Docker image.



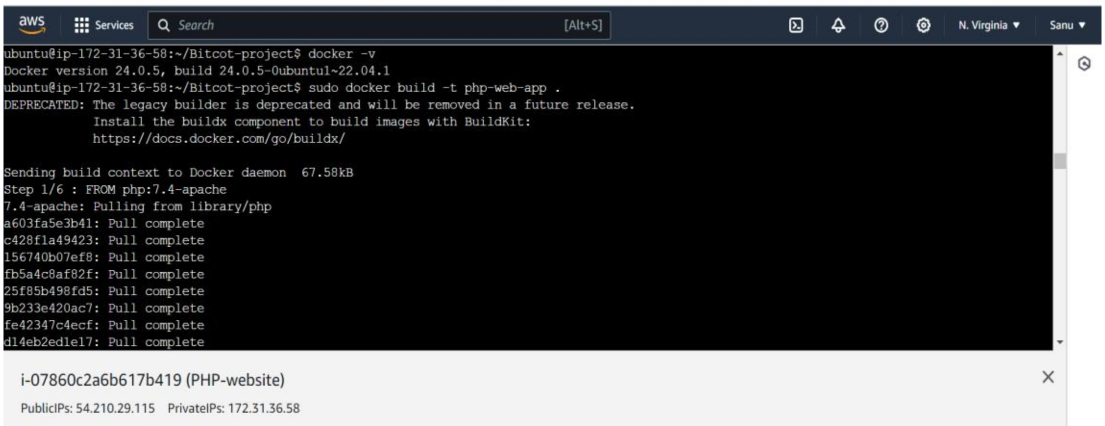
```
ubuntu@ip-172-31-36-58:~/Bitcot-project$ ll
total 24
drwxrwxr-x 3 ubuntu ubuntu 4096 Jan  3 14:44 ./
drwxr-x--- 5 ubuntu ubuntu 4096 Jan  3 14:44 ../
drwxrwxr-x 8 ubuntu ubuntu 4096 Jan  3 14:44 .git/
-rw-rw-r-- 1 ubuntu ubuntu  432 Jan  3 14:44 Dockerfile
-rw-rw-r-- 1 ubuntu ubuntu  193 Jan  3 14:44 db.inc.php
-rw-rw-r-- 1 ubuntu ubuntu 3168 Jan  3 14:44 samplepage.php
ubuntu@ip-172-31-36-58:~/Bitcot-project$ 
```

i-07860c2a6b617b419 (PHP-website)

PublicIPs: 54.210.29.115 PrivateIPs: 172.31.36.58

3.3 Run Docker Container:

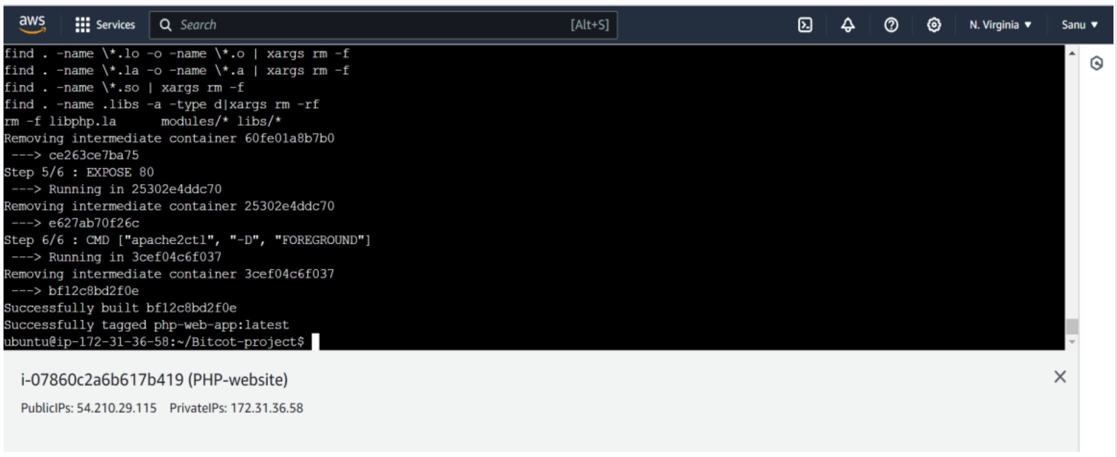
- Start the Docker container.
- Access the PHP application using the EC2 instance's public IP.



```
ubuntu@ip-172-31-36-58:~/Bitcot-project$ docker -v
Docker version 24.0.5, build 24.0.5-0ubuntu1~22.04.1
ubuntu@ip-172-31-36-58:~/Bitcot-project$ sudo docker build -t php-web-app .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  67.58kB
Step 1/6 : FROM php:7.4-apache
7.4-apache: Pulling from library/php
a603fa5e3b41: Pull complete
c428f1a49423: Pull complete
156740b07ef8: Pull complete
fb5a4c8af82f: Pull complete
25f85b498fd5: Pull complete
9b233e420ac7: Pull complete
fe42347c4ecf: Pull complete
d14eb2ed1e17: Pull complete
```

i-07860c2a6b617b419 (PHP-website)

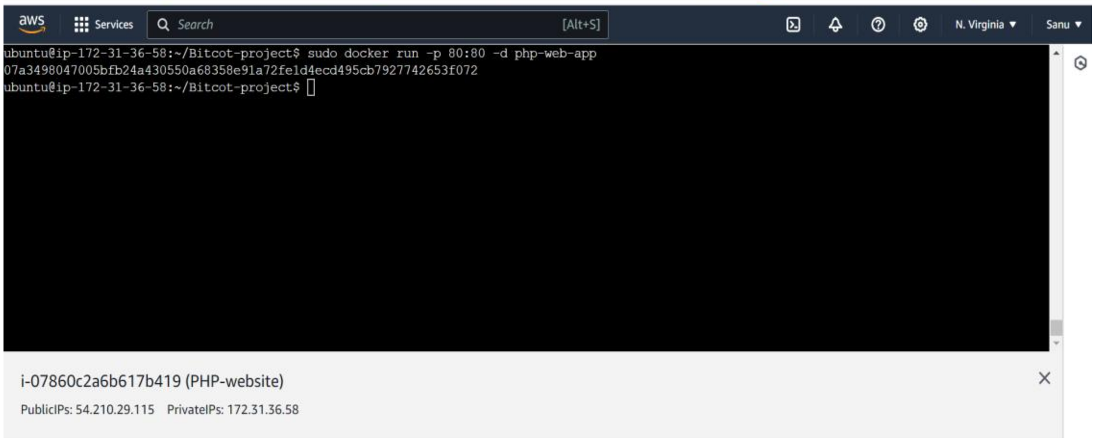
PublicIPs: 54.210.29.115 PrivateIPs: 172.31.36.58



```
find . -name \*.lo -o -name \*.o | xargs rm -f
find . -name \*.la -o -name \*.a | xargs rm -f
find . -name \*.so | xargs rm -f
find . -name .libs -a -type d|xargs rm -rf
rm -f libphp.la      modules/* libs/*
Removing intermediate container 60fe01a8b7b0
 ---> ce263ce7ba75
Step 5/6 : EXPOSE 80
 ---> Running in 25302e4ddc70
Removing intermediate container 25302e4ddc70
 ---> e627ab70f26c
Step 6/6 : CMD ["apache2ctl", "-D", "FOREGROUND"]
 ---> Running in 3cef04c6f037
Removing intermediate container 3cef04c6f037
 ---> bf12c8bd2f0e
Successfully built bf12c8bd2f0e
Successfully tagged php-web-app:latest
ubuntu@ip-172-31-36-58:~/Bitcot-project$
```

i-07860c2a6b617b419 (PHP-website)

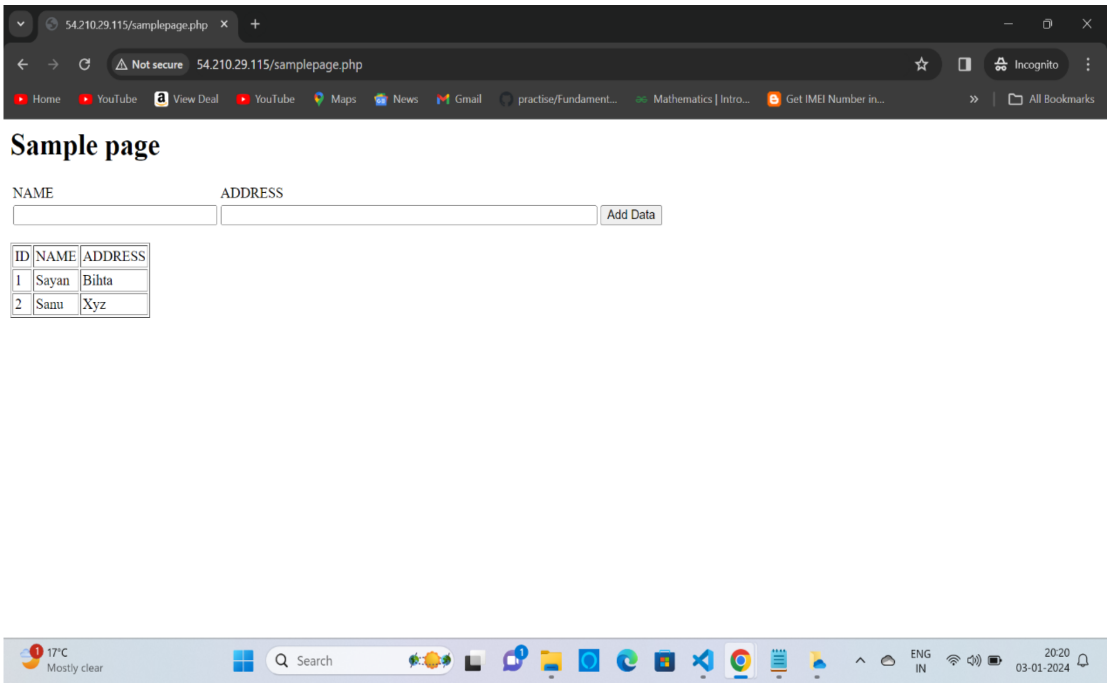
PublicIPs: 54.210.29.115 PrivateIPs: 172.31.36.58



```
ubuntu@ip-172-31-36-58:~/Bitcot-project$ sudo docker run -p 80:80 -d php-web-app
07a3498047005bfb24a430550a68358e91a72fe1d4ecd495cb7927742653f072
ubuntu@ip-172-31-36-58:~/Bitcot-project$
```

i-07860c2a6b617b419 (PHP-website)

PublicIPs: 54.210.29.115 PrivateIPs: 172.31.36.58



54.210.29.115/samplepage.php

Not secure 54.210.29.115/samplepage.php

Sample page

NAME ADDRESS

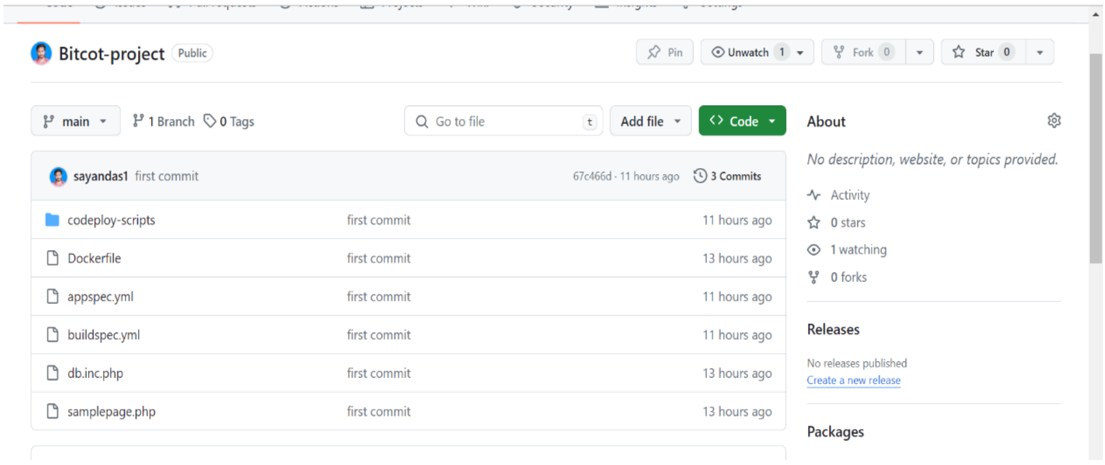
Add Data

ID	NAME	ADDRESS
1	Sayan	Bihta
2	Sanu	Xyz

4. Setting up CI/CD with AWS CodePipeline, CodeBuild, and CodeDeploy

4.1 **GitHub Repository:**

- Create a GitHub repository for the PHP application.



Bitcot-project Public

Pin · Unwatch 1 · Fork 0 · Star 0

main · 1 Branch · 0 Tags · Go to file · Add file · Code

sayandas1 first commit · 67c466d · 11 hours ago · 3 Commits

codeploy-scripts	first commit	11 hours ago
Dockerfile	first commit	13 hours ago
appspec.yml	first commit	11 hours ago
buildspec.yml	first commit	11 hours ago
db.inc.php	first commit	13 hours ago
samplepage.php	first commit	13 hours ago

About

No description, website, or topics provided.

Activity · 0 stars · 1 watching · 0 forks

Releases

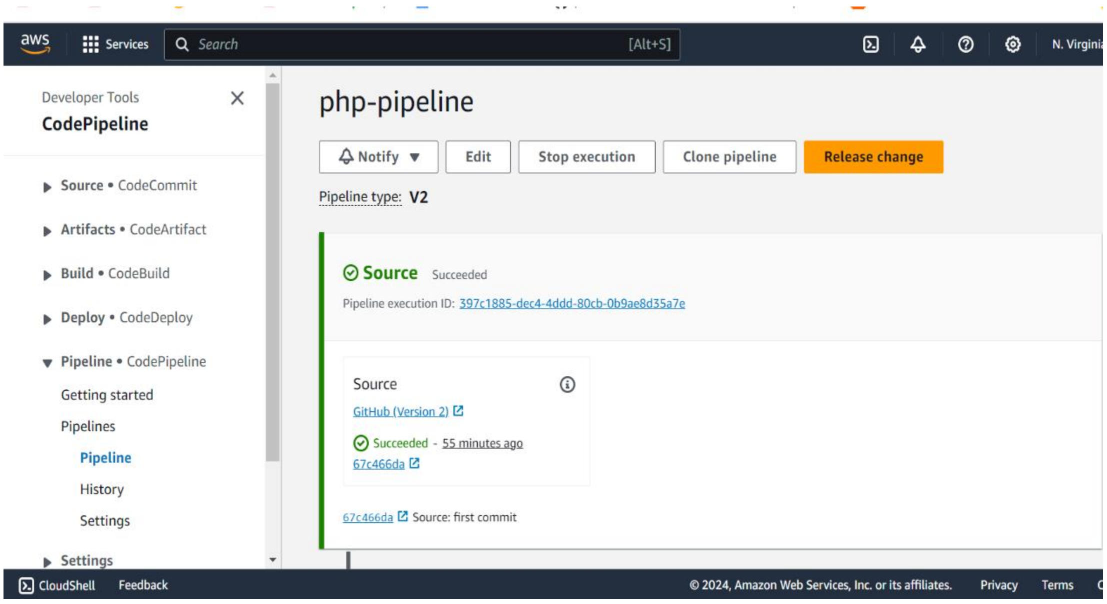
No releases published

Create a new release

Packages

4.2 **Configure AWS CodePipeline:**

- Set up a new CodePipeline with GitHub as the source, CodeBuild for the build stage, and CodeDeploy for deployment.



php-pipeline

Pipeline type: V2

Source Succeeded

Pipeline execution ID: 397c1885-dec4-4ddd-80cb-0b9ae8d35a7e

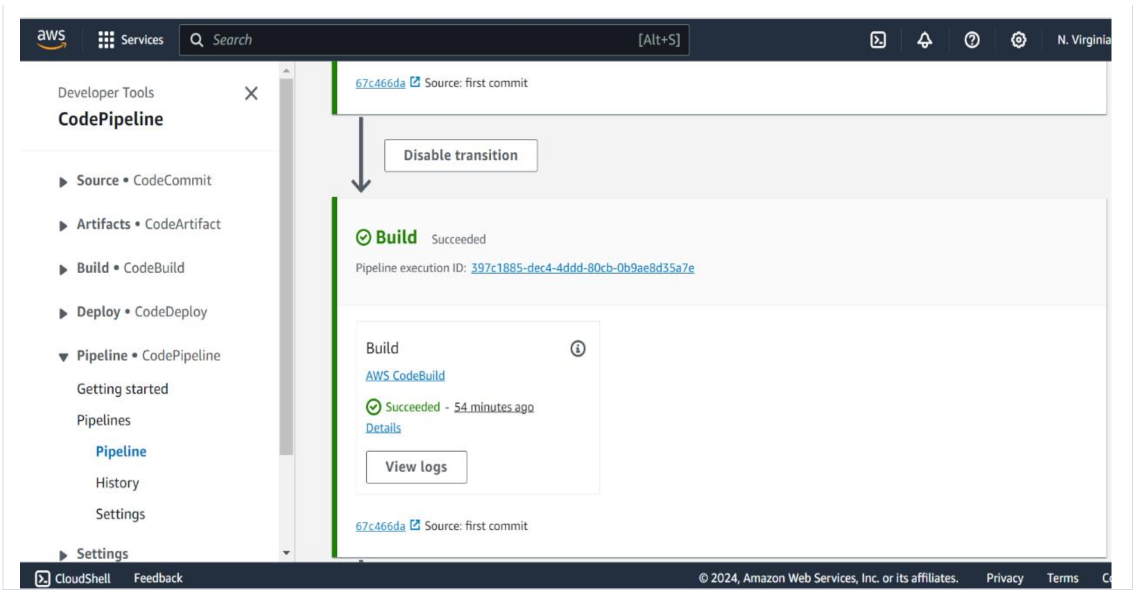
Source

GitHub (Version 2)

Succeeded - 55 minutes ago

67c466d

67c466d Source: first commit



67c466da Source: first commit

Build Succeeded

Pipeline execution ID: 397c1885-dec4-4ddd-80cb-0b9ae8d35a7e

Build

AWS CodeBuild

Succeeded - 54 minutes ago

Details

67c466da Source: first commit

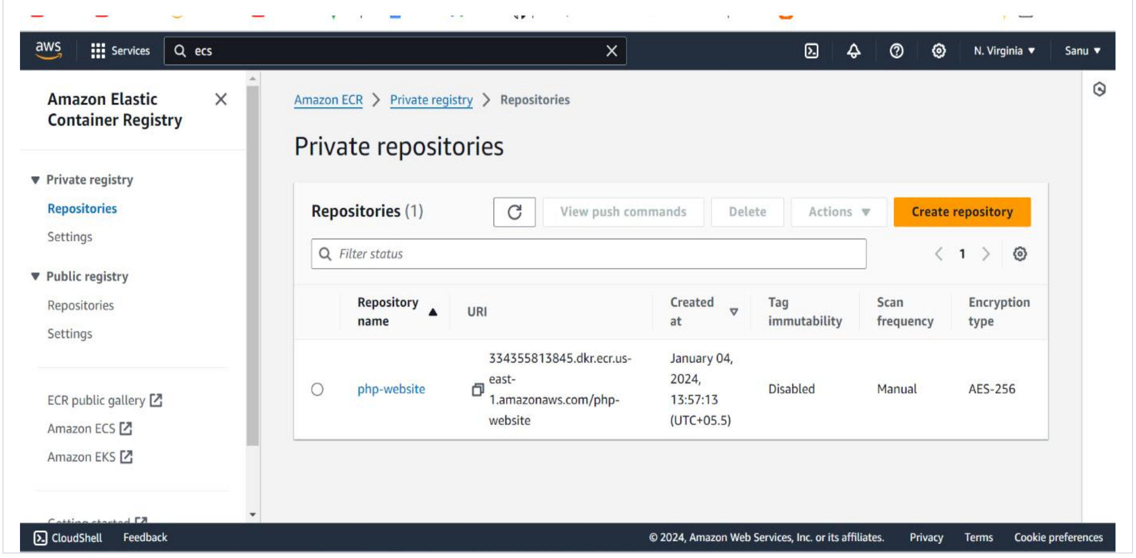
© 2024, Amazon Web Services, Inc. or its affiliates.

4.3 **Build and Deployment Configuration Files:**

- Create `buildspec.yml` and `appspec.yml` files as provided in the initial instructions.
- Adjust these files based on the project structure and requirements.

4.4 **Push the Docker image to ECR:**

- Create an ECR repository.
- Push the Docker image to ECR during build process.
- Write the credentials and commands to push the image to ECR on `buildspec.yml` file.



Amazon ECR > Private registry > Repositories

Private repositories

Repositories (1)

Repository name	URI	Created at	Tag immutability	Scan frequency	Encryption type
php-website	334355813845.dkr.ecr.us-east-1.amazonaws.com/php-website	January 04, 2024, 13:57:13 (UTC+05.5)	Disabled	Manual	AES-256

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