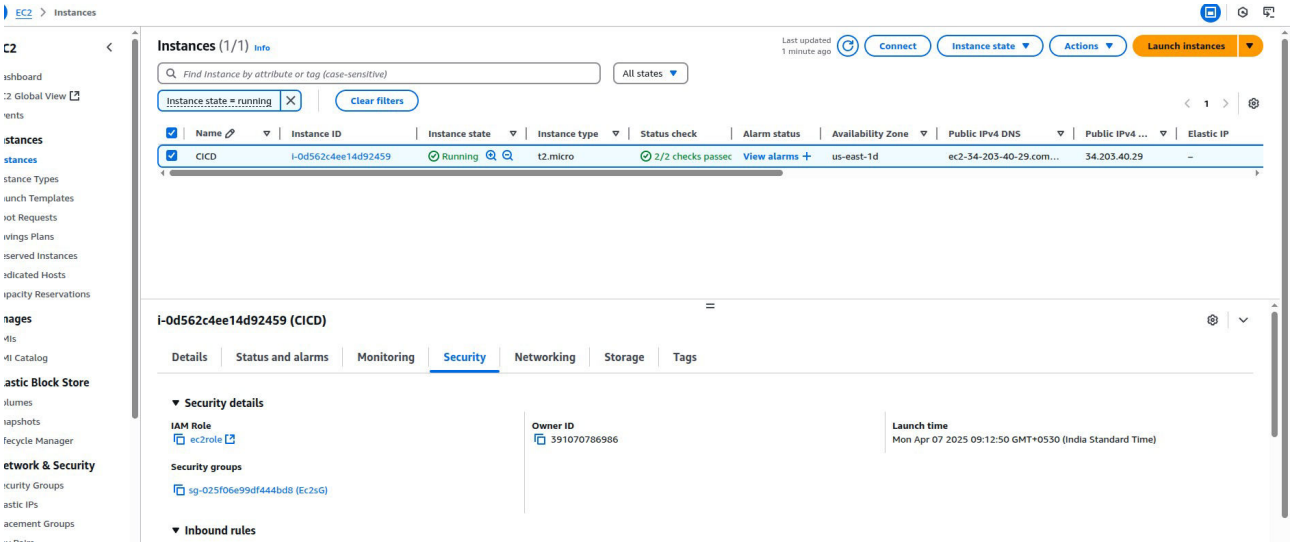
Task Description:

Deploy a simple web application using AWS code commit, code build and deploy & access via browser and automate via codepipeline.

Solution:

🧱 STEP 1: Prepare EC2 Instance (CodeDeploy Target)

1. **Launch EC2 Instance**
    - Amazon Linux 2 or Ubuntu
    - Assign a key pair
    - Open ports: HTTP (80), SSH (22)
    - Attach IAM Role with this policy: `AmazonEC2RoleforAWSCodeDeploy` *(or create a custom one with CodeDeploy + S3 access)*



2. **Install CodeDeploy Agent**

We'll update your canvas with the proper script using the **correct region**.

Update your script in canvas to:

```
sudo yum update -y
sudo yum install ruby wget -y
cd /home/ec2-user
wget https://aws-codedeploy-us-east-1.s3.us-east-1.amazonaws.com/latest/install
chmod +x ./install
```

```
sudo ./install auto
sudo systemctl start codedeploy-agent
sudo systemctl enable codedeploy-agent
```

Replace `us-east-1` with your actual region.

📂 STEP 2: Prepare GitHub Repo

Your GitHub repo should have:

```
my-web-app/
├── index.html (or app files)
├── buildspec.yml
├── appspec.yml
└── scripts/
    └── restart_server.sh
```

Example `buildspec.yml`:

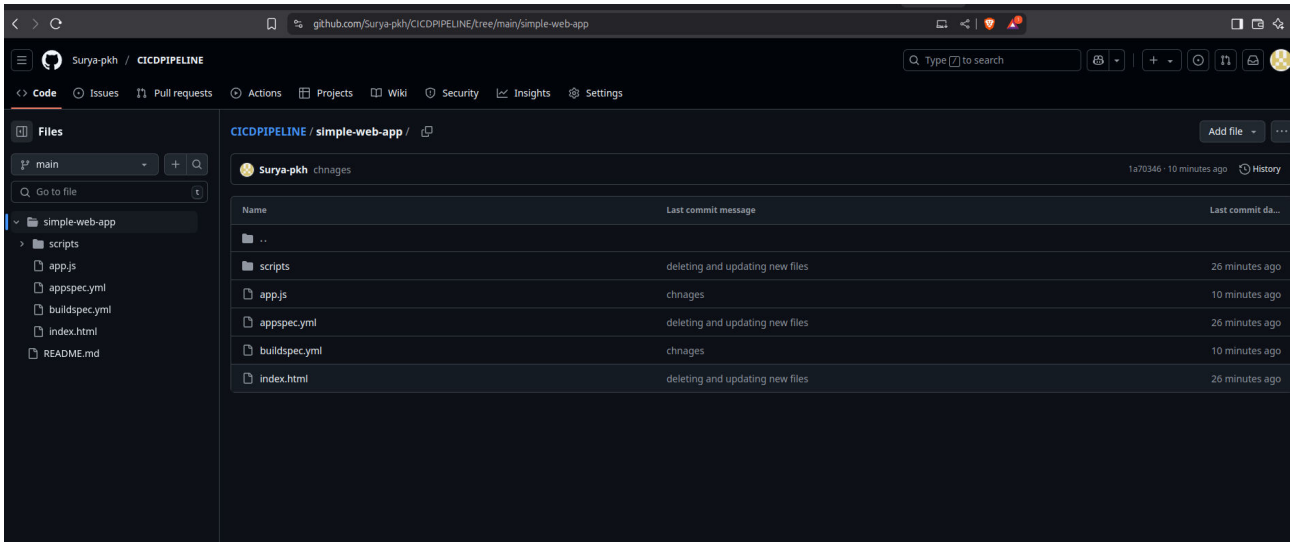
```
version: 0.2
phases:
  build:
    commands:
      - echo Build started on `date`
artifacts:
  files:
    - '**/*'
```

Example `appspec.yml`:

```
version: 0.0
os: linux
files:
  - source: /
    destination: /var/www/html
hooks:
  AfterInstall:
    - location: scripts/restart_server.sh
      timeout: 300
      runas: root
```

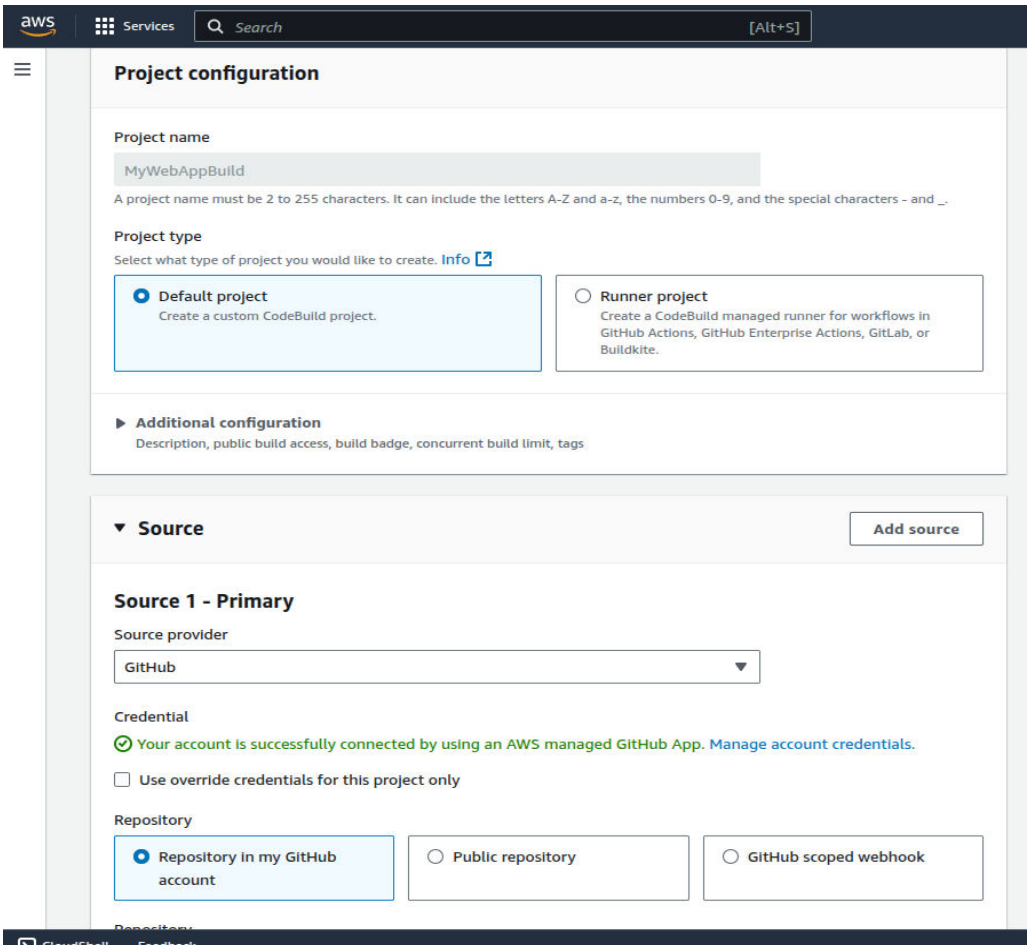
Example `restart_server.sh`:

```
#!/bin/bash
sudo systemctl restart httpd || sudo systemctl restart nginx
```

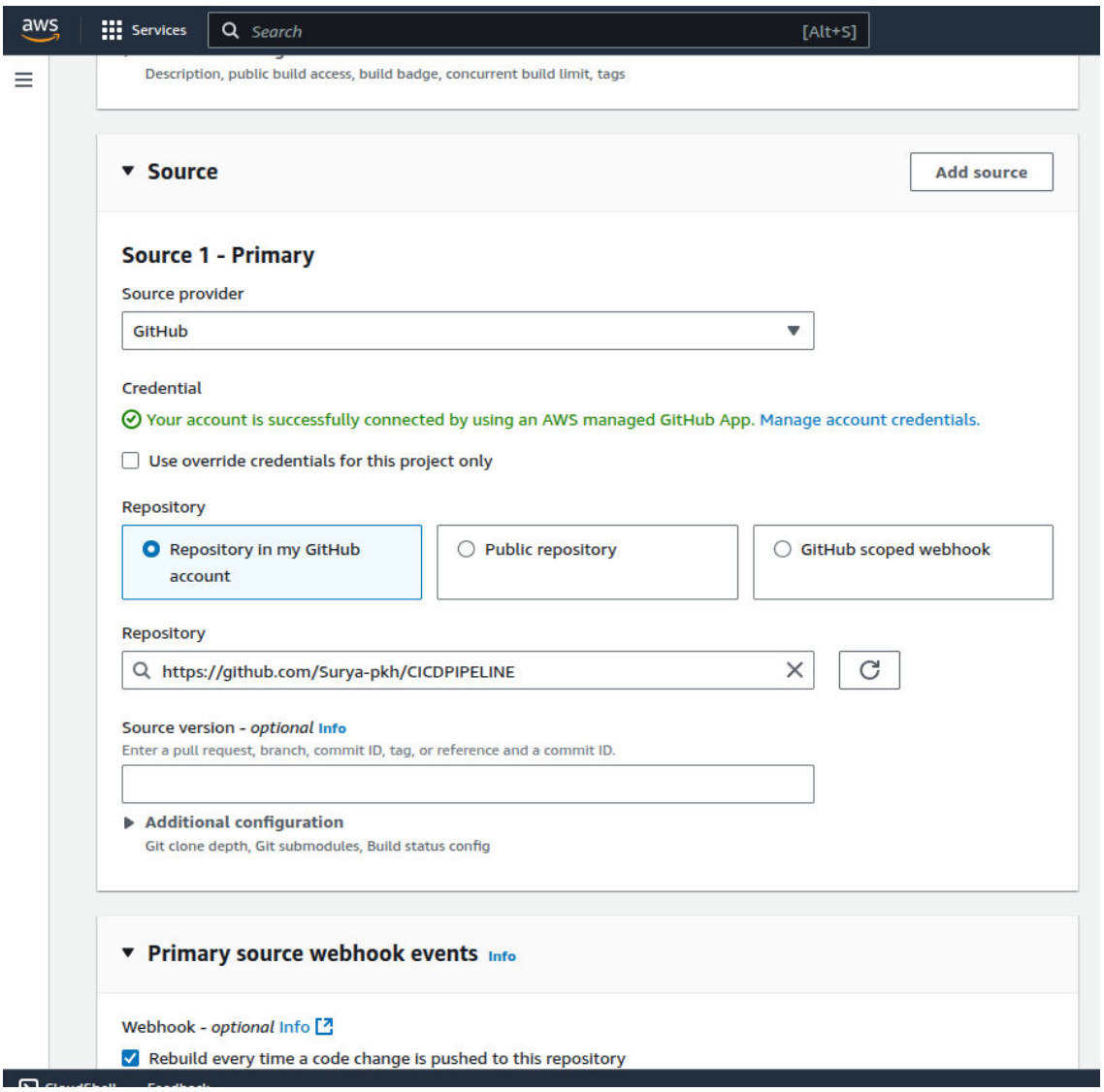


🔧 STEP 3: Set Up CodeBuild

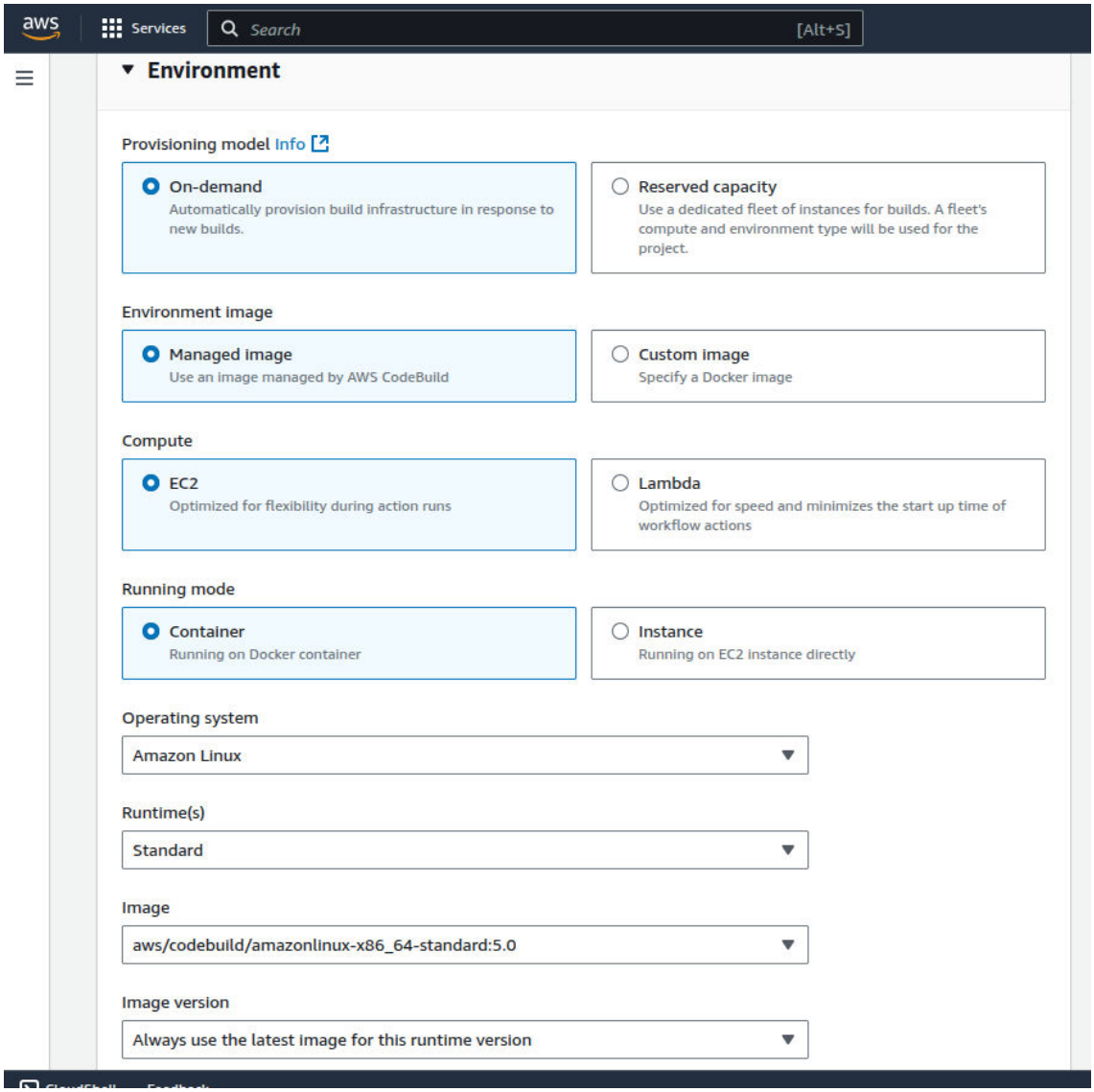
1. Go to **CodeBuild** > **Create Build Project**
2. Set Name: `MyWebAppBuild`



3. Source Provider: **GitHub**

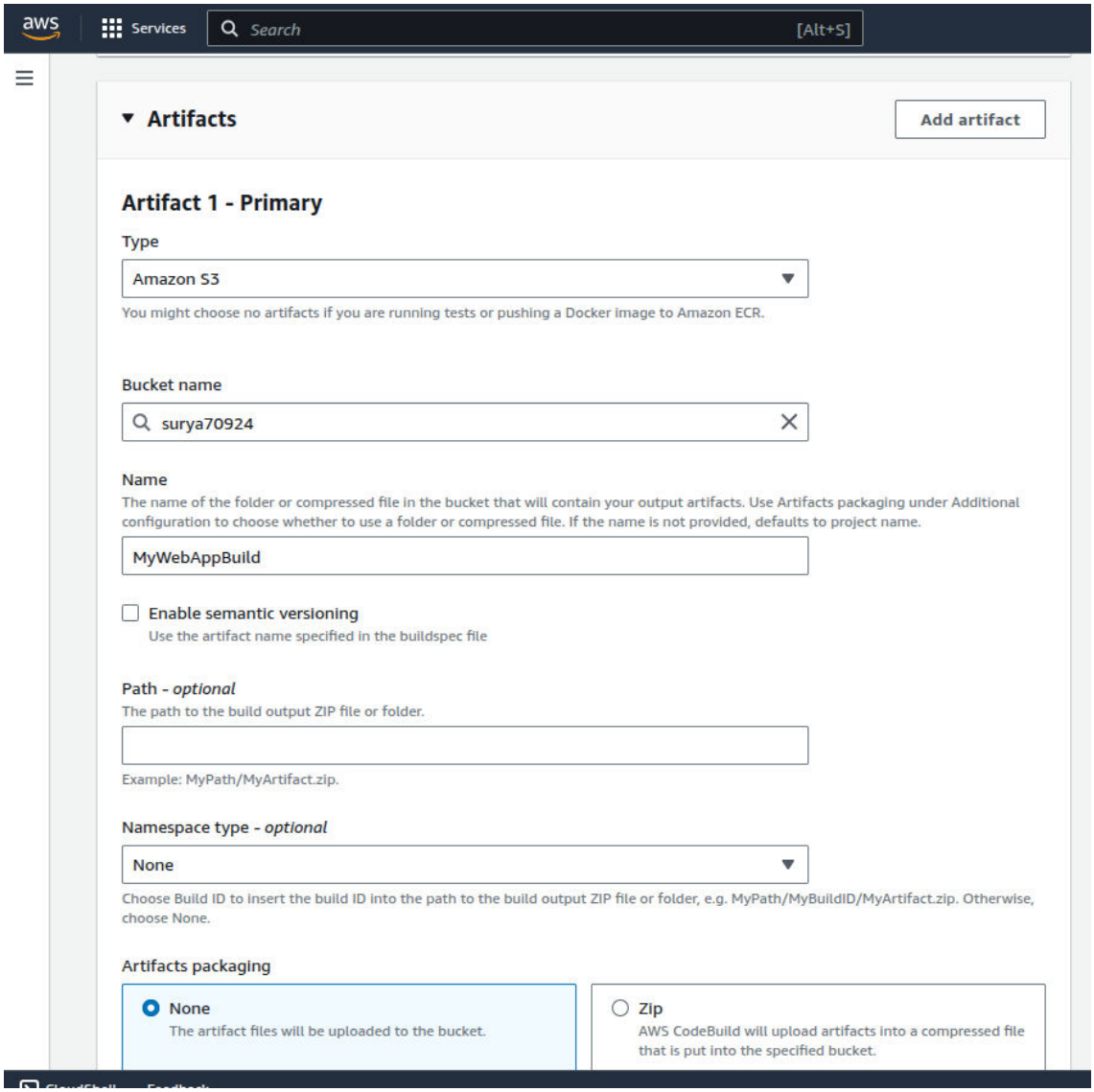


4. Environment:
    - Managed image: Amazon Linux
    - Runtime: Node.js or standard
    - Allow AWS to create a new role

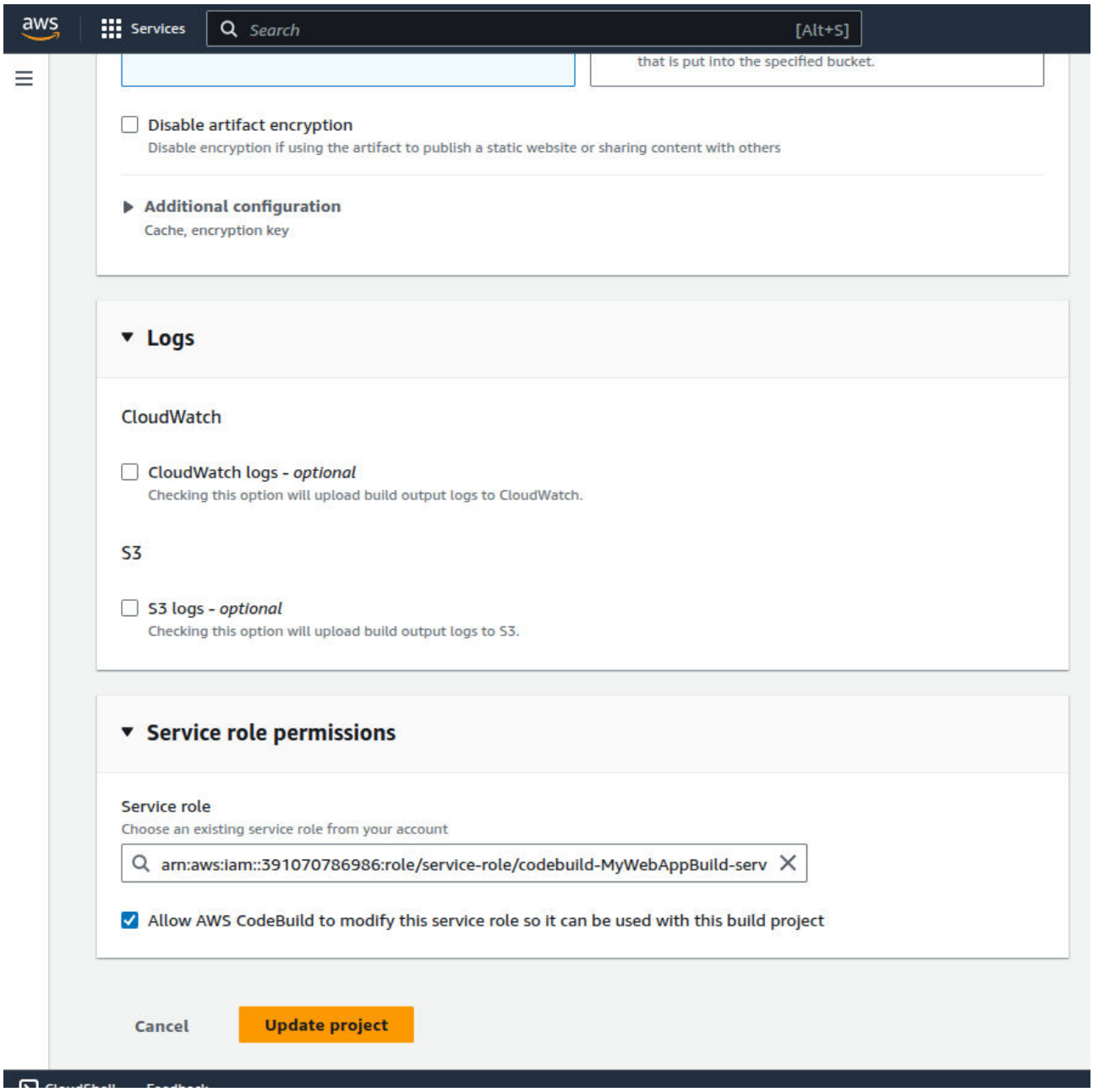


5. Artifacts:

    - Type: Amazon S3 (create a new bucket)

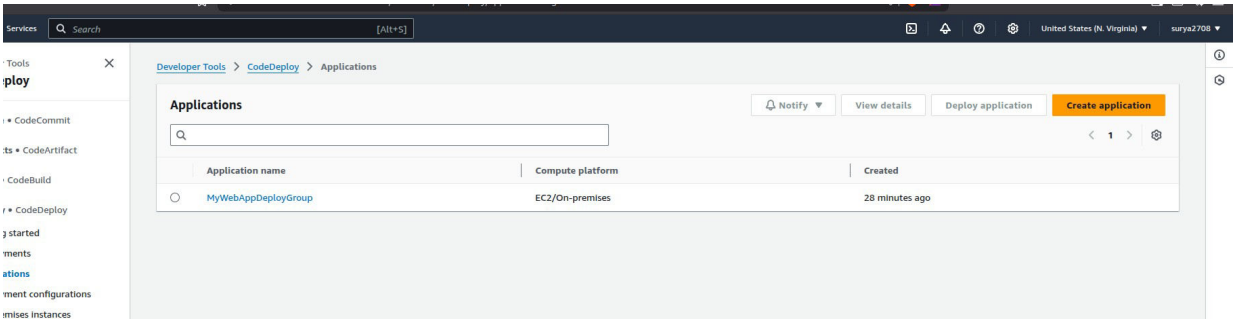

aws
Services
Search
[Alt+S]
Artifacts
Add artifact
Artifact 1 - Primary
Type
Amazon S3
You might choose no artifacts if you are running tests or pushing a Docker image to Amazon ECR.
Bucket name
surya70924
Name
The name of the folder or compressed file in the bucket that will contain your output artifacts. 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.
MyWebAppBuild
Enable semantic versioning
Use the artifact name specified in the buildspec file
Path - optional
The path to the build output ZIP file or folder.
Example: MyPath/MyArtifact.zip.
Namespace type - optional
None
Choose Build ID to insert the build ID into the path to the build output ZIP file or folder, e.g. MyPath/MyBuildID/MyArtifact.zip. Otherwise, choose None.
Artifacts packaging
None
The artifact files will be uploaded to the bucket.
Zip
AWS CodeBuild will upload artifacts into a compressed file that is put into the specified bucket.

6. Buildspec: From source (`buildspec.yml`)

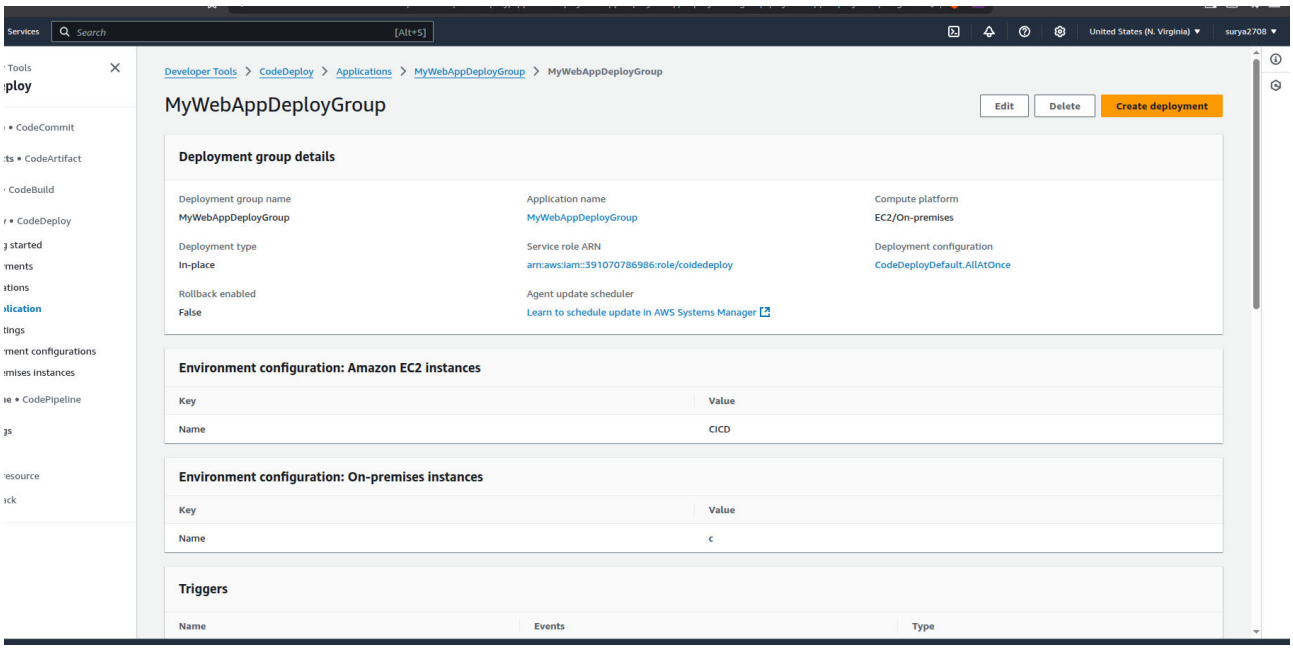


🚀 STEP 4: Set Up CodeDeploy

1. Go to **CodeDeploy** > **Applications** > **Create application**



2. Compute Platform: EC2/On-premise
3. Create Deployment Group:
    - Name: `MyWebAppDeployGroup`
    - Service role: Create a role with `AWSCodeDeployRole`
    - Target: EC2 instance (tag-based or manually attach)

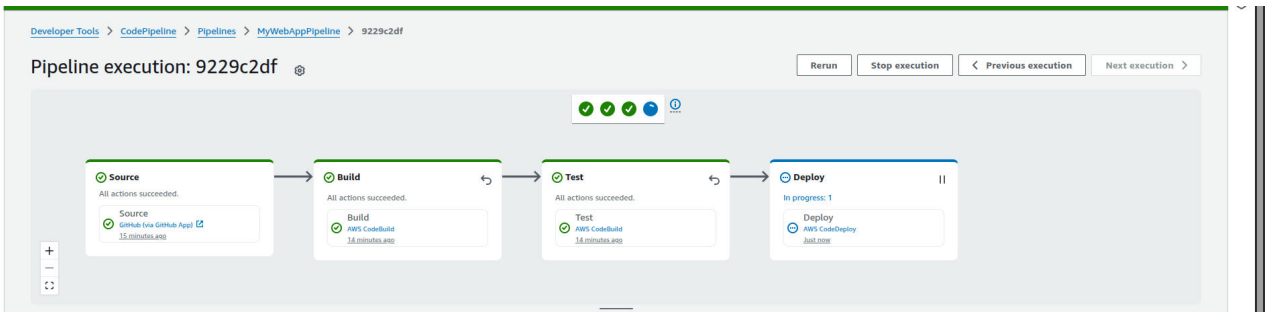


⚙️ STEP 5: Create CodePipeline

1. Go to **CodePipeline** > **Create pipeline**
2. Pipeline Name: `MyWebAppPipeline`
3. Source:

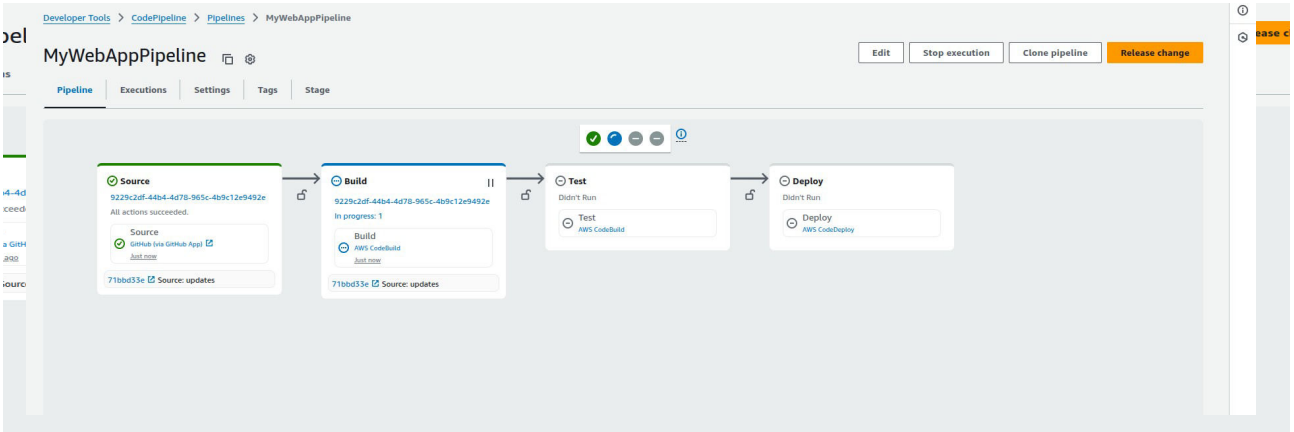
- Provider: **GitHub**
- Connect and select your repo/branch

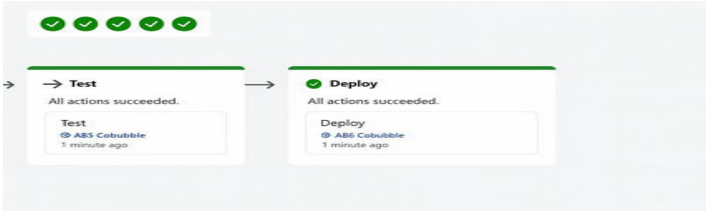
4. Build:
   - Provider: **CodeBuild**
   - Choose the project `MyWebAppBuild`
5. Deploy:
   - Provider: **CodeDeploy**
   - Choose the app and deployment group



✅ STEP 6: Trigger Deployment

- Push any code change to GitHub.
- CodePipeline will automatically build and deploy to EC2.





- Visit: `http://<your-ec2-public-ip>` to see your web app.