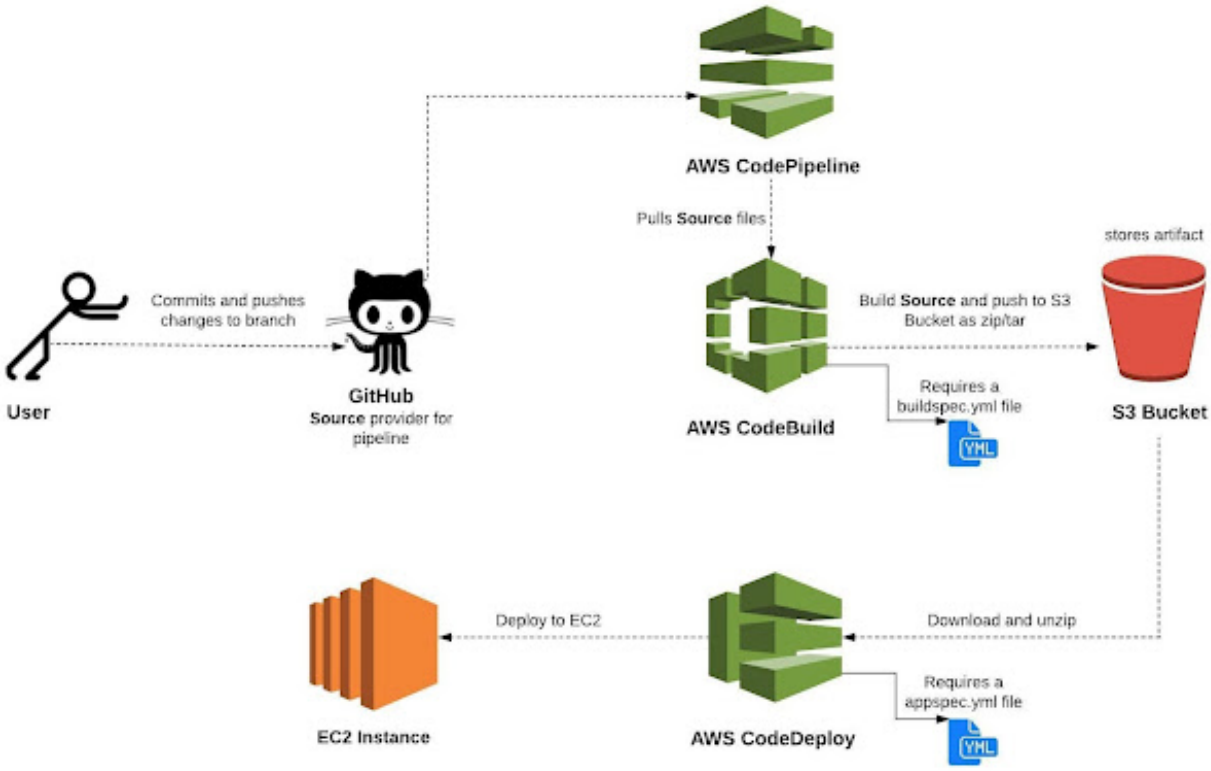
Deploy a ReactJS Application to AWS EC2 Instance using AWS CodePipeline

AWS CodePipeline Notes (Topics Covered : CodeBuild,CodeDeploy and Deployment of an App

AWS CodePipeline is a continuous integration and continuous delivery (CI/CD) AWS service that allows you to automate the release process for your application or service. Every time you commit a code change to your source(GitHub, AWS CodeCommit, etc), CodePipeline automatically builds, tests, and deploys your code based on the release process models you define while initializing your CodePipeline. This enables you to rapidly and reliably deliver features and updates.



Deploy a ReactJS Application to AWS EC2 Instance using AWS CodePipeline

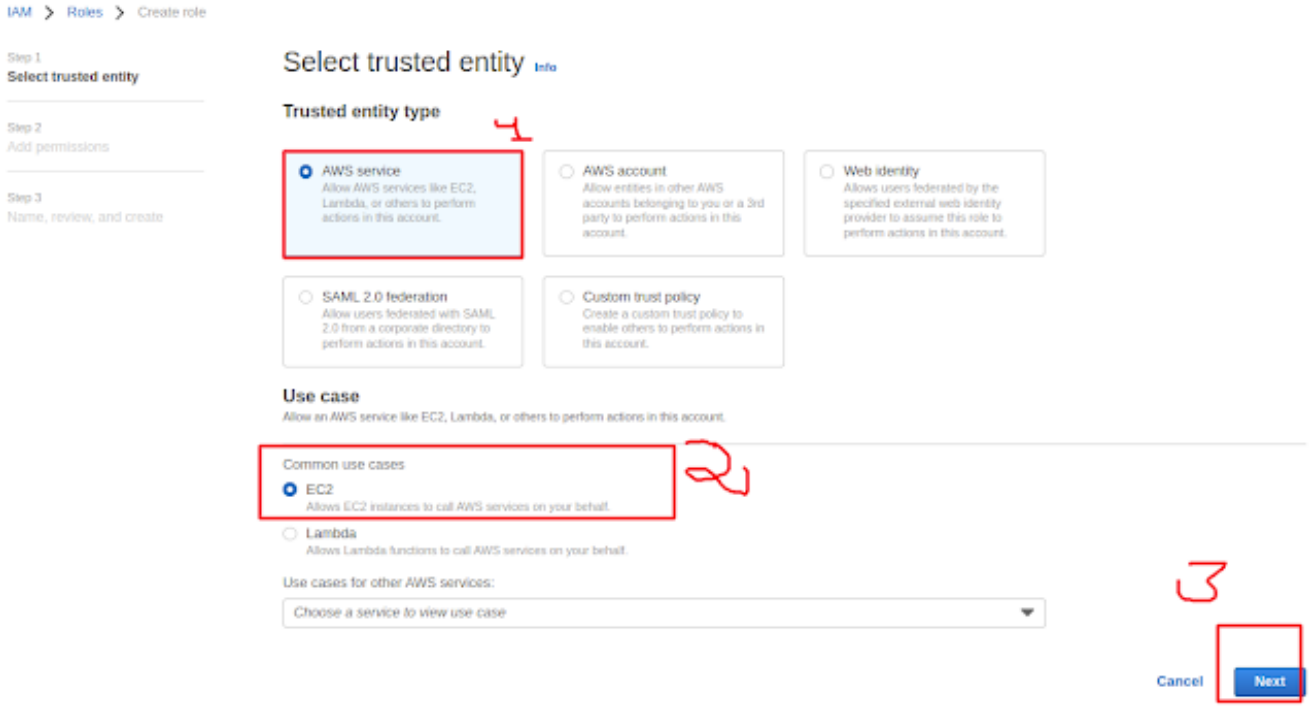
Steps that we will follow:

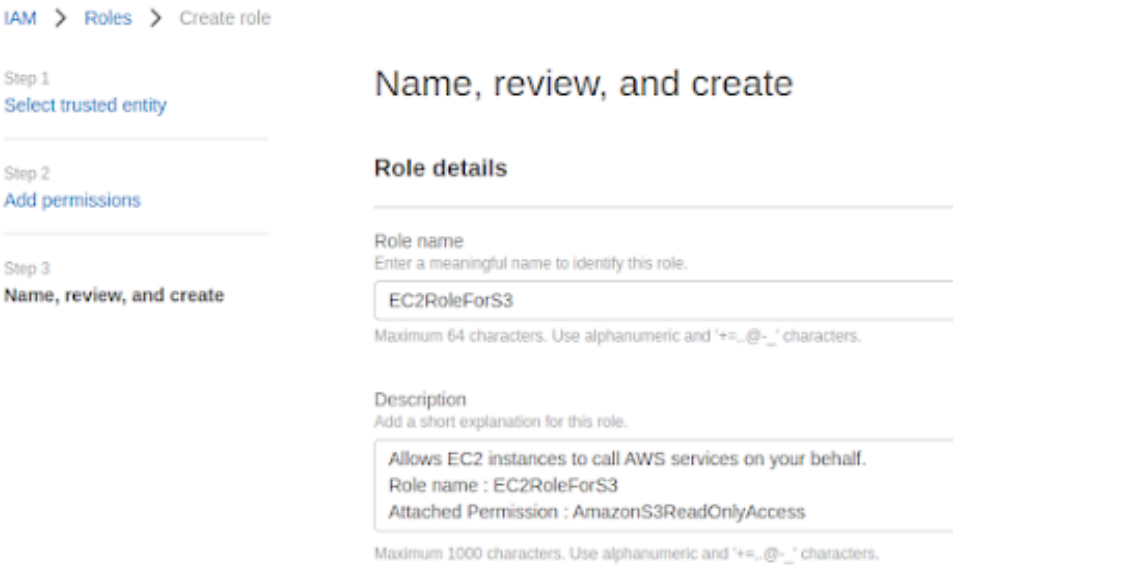
1. Create IAM Role for EC2 & AWS CodeDeploy
2. Launch an EC2 instance & then will attach that EC2 Role.
3. Create a CodePipeline using Github, CodeBuild and CodeDeploy

Abdur Razzaq
Linkedin :: @abdurrazzaq04

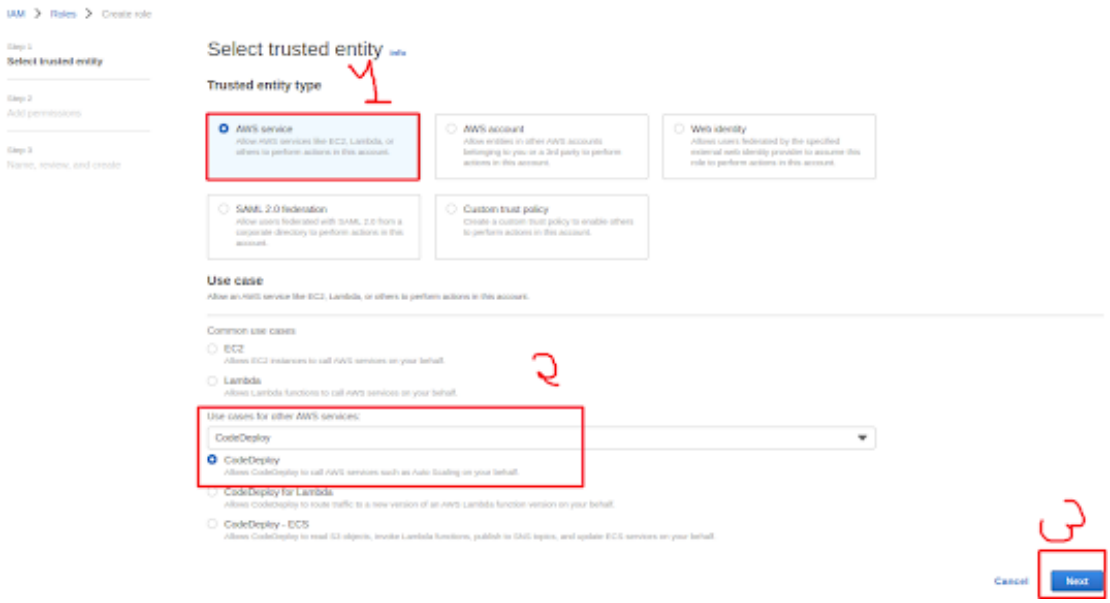
Step 1) Create IAM Role for EC2 and AWS CodeDeploy

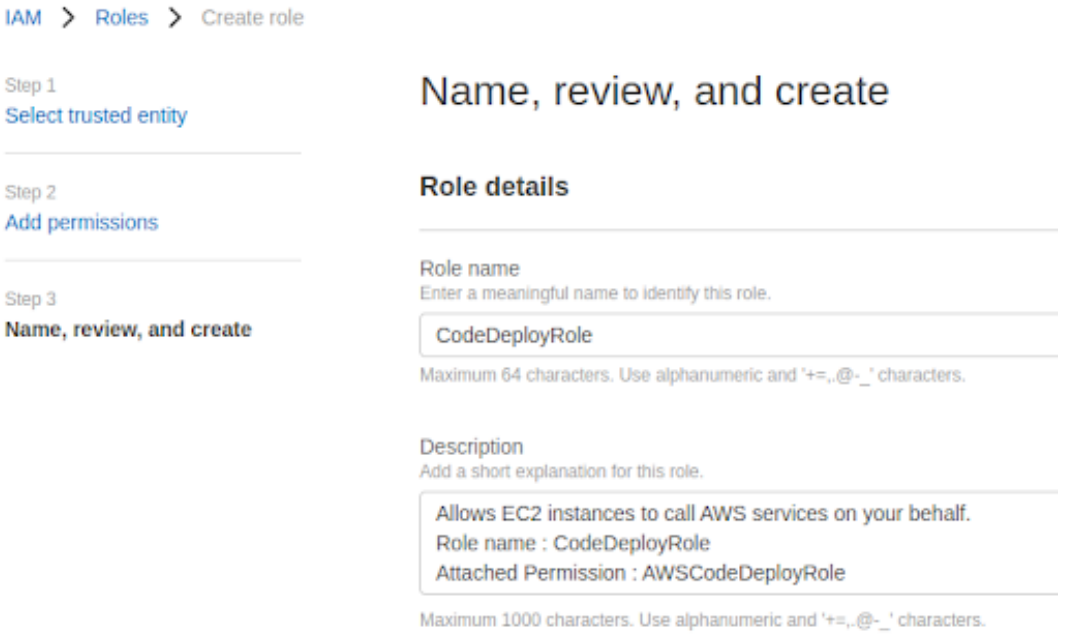
Create a new role for **EC2** and attach **AmazonS3ReadOnlyAccess** policy which will allow our EC2 instance to access stored artifacts from the Amazon S3 bucket.





Create a new service role for **CodeDeploy** and attach **AWSCodeDeployRole** policy which will provide the permissions for our service role to read tags of our EC2 instance, publish information to Amazon SNS topics and much more task.





Step 2) Launch an Linux EC2 instance

Note: Please Select t3.medium Instance Type

And then Install code depoloy agent by follwing commands on terminal.

For Centos & Amazon Linux 2

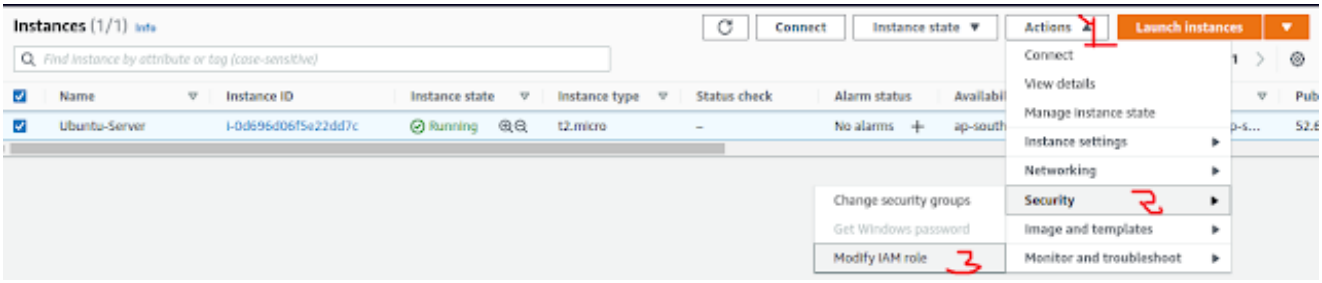
```
sudo yum update
sudo yum install ruby
sudo yum install wget
wget https://aws-codedeploy-us-east-1.s3.amazonaws.com/latest/install
chmod +x ./install
sudo ./install auto
sudo service codedeploy-agent status
```

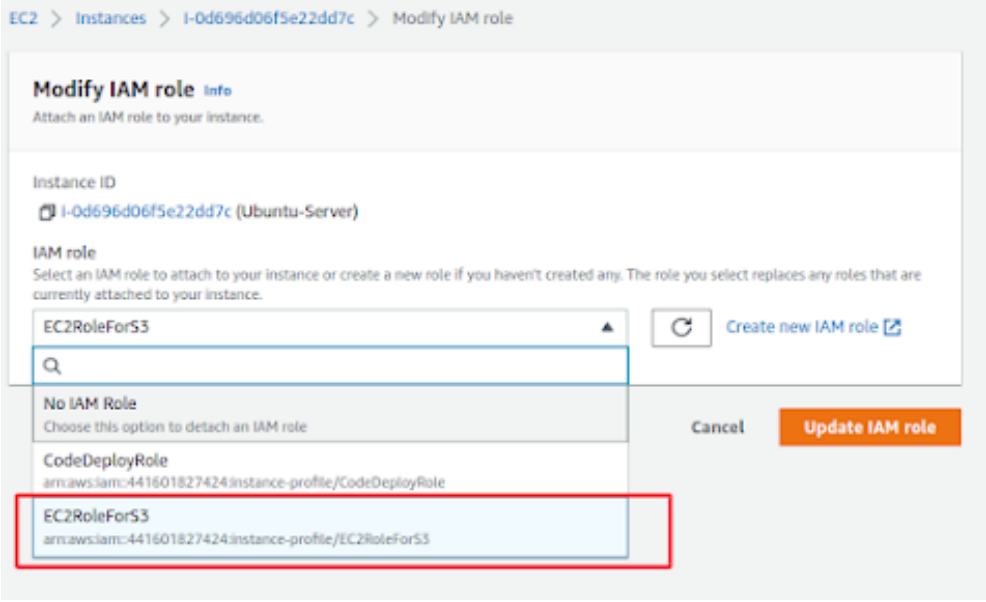
For Ubuntu & Debian Based

```
sudo apt-get update
sudo apt-get install ruby

sudo apt-get install wget
cd /home/ubuntu
wget https://aws-codedeploy-ca-central-1.s3.ca-central-1.amazonaws.com/latest/install
chmod +x ./install
sudo ./install auto
sudo service codedeploy-agent status
```

Now, Let's Attach that EC2 Role with it.





Note : Let's Make a Github Repository and download the following source code and push on it.

Download-SourceCode

SourceCode File Structure

README.md
package-lock.json
package.json
public/
src/

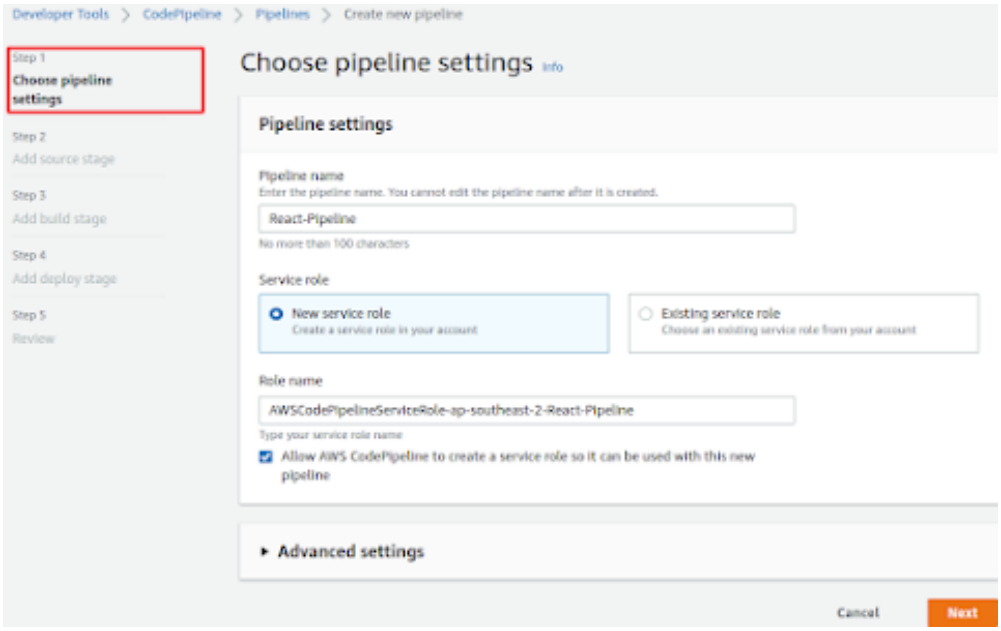
buildspec.yml ----Build File for AWS CodeBuild
appspec.yml ----Information About Server & Hooks
scripts/ appspec.yml >> file for AWS CodeDeploy
after_install.sh
app_start.sh
before_install.sh

server.js

Step 3) Create a CodePipeline using Github, CodeBuild and CodeDeploy

a) Create CodePipeline

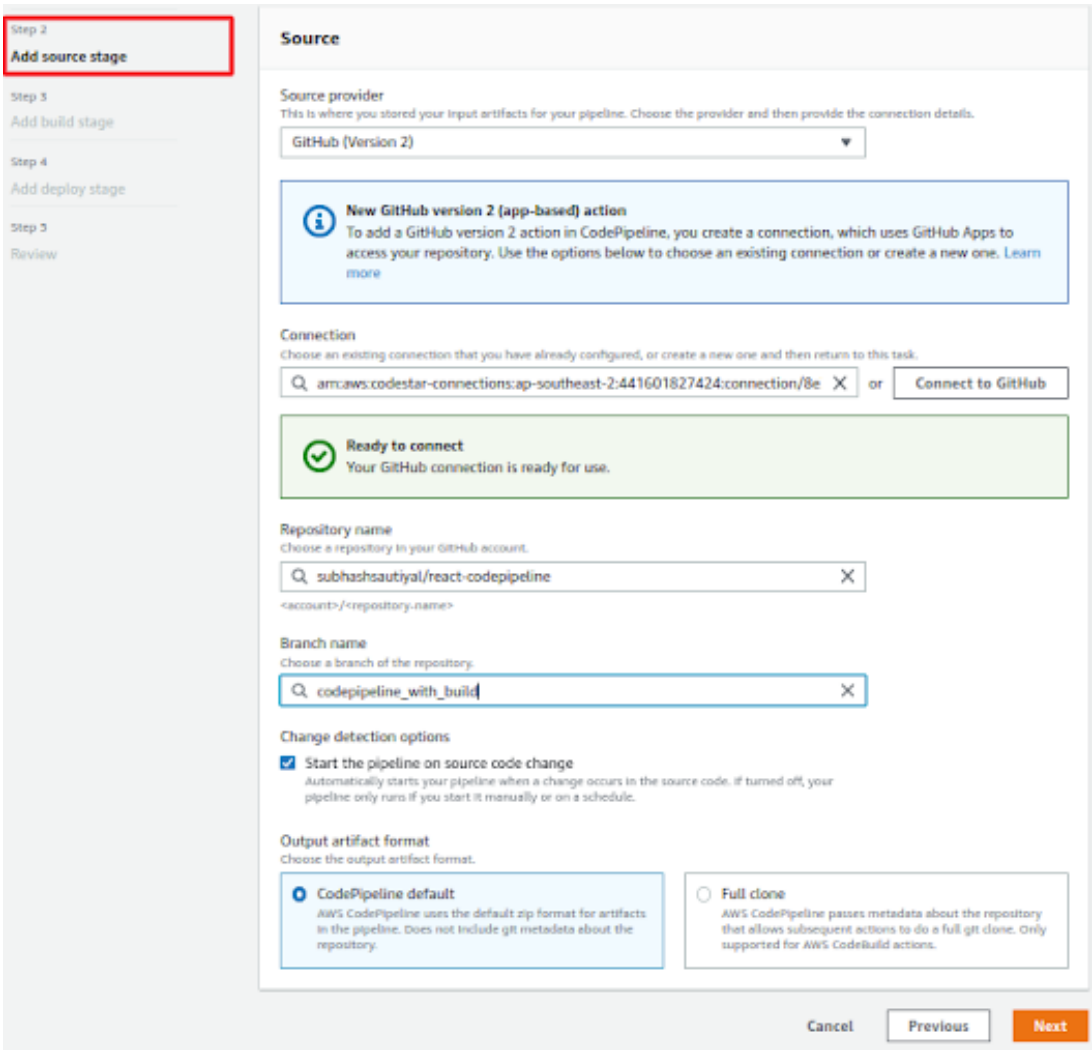
Let's navigate to CodePipeline via AWS Management Console and click on Create pipeline:



b) Choose Github in Code Source

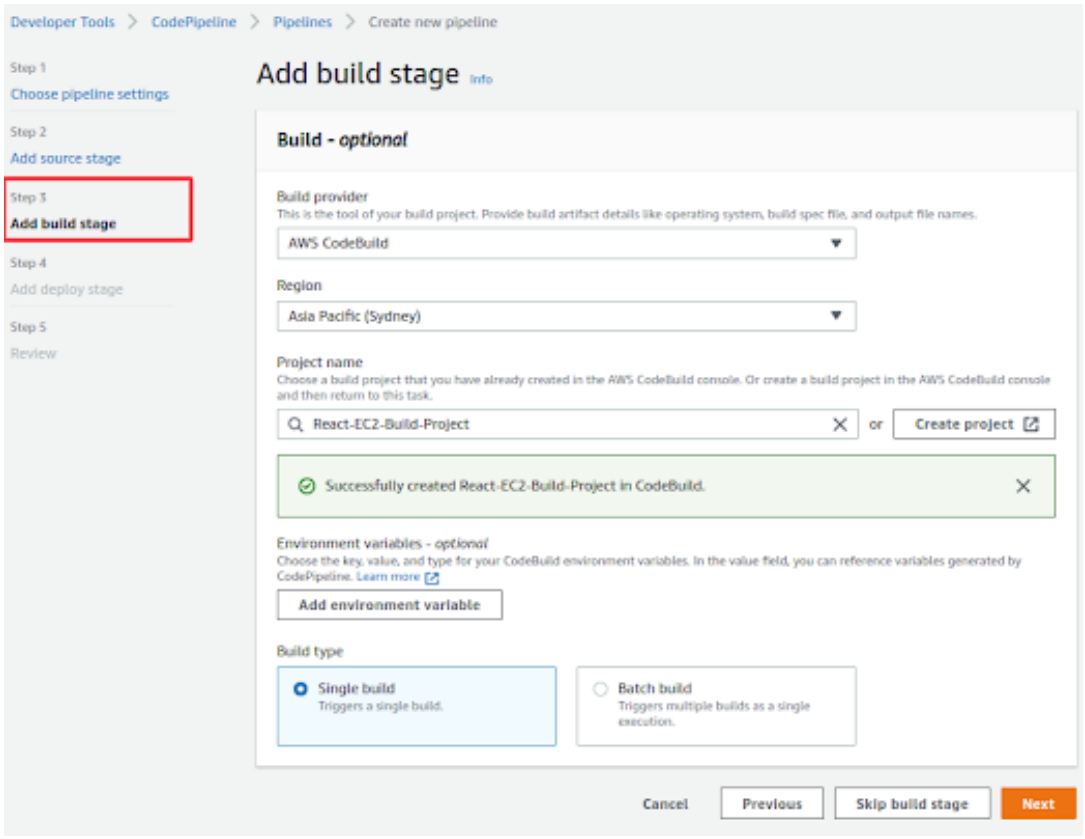
After selecting GitHub as the source provider, click on the Connect to GitHub button. You'll then be prompt to enter your GitHub login credentials

Once you grant AWS CodePipeline access to your GitHub repository, you can select a repository and branch for CodePipeline to upload commits to this repository to your pipeline



c) Configure CodeBuild (Optional)

If you haven't created a project prior to creating your pipeline,then you can create a project directly from here by clicking Create project button.

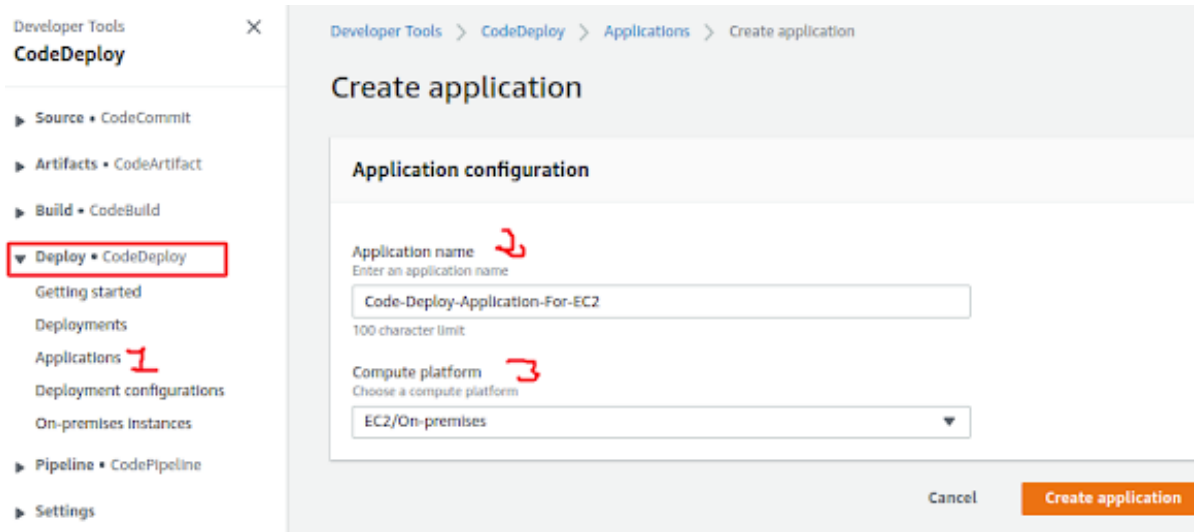


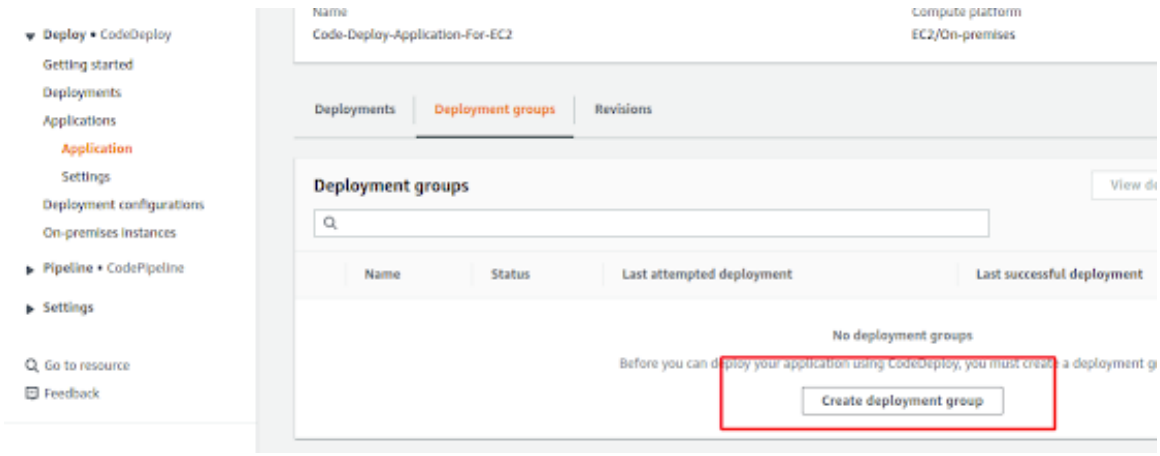
Note: Buildspec file is a collection of build commands and related settings, in YAML format, that CodeBuild uses to run a build. For my project, I created a buildspec.yaml file and added it in the root of my project directory:

d) Add Depoly Stage

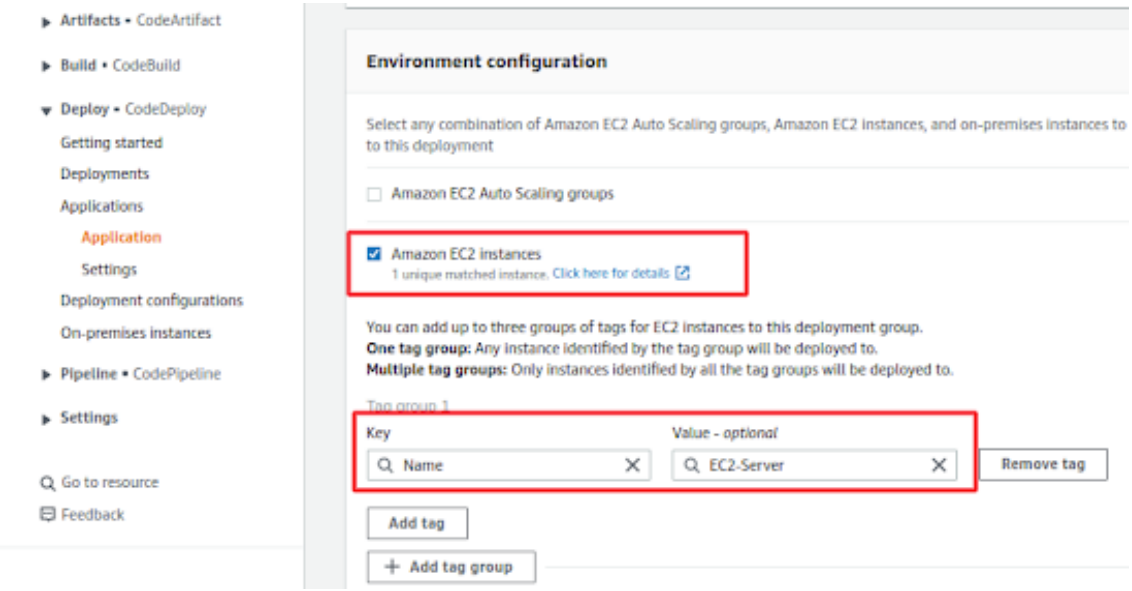
Note : Before going to configure Add Depoly Stage, Let's make duplicate tab of current tab.

and then go to code deploy in the nevigation, Select Application, then add create a deployment group.

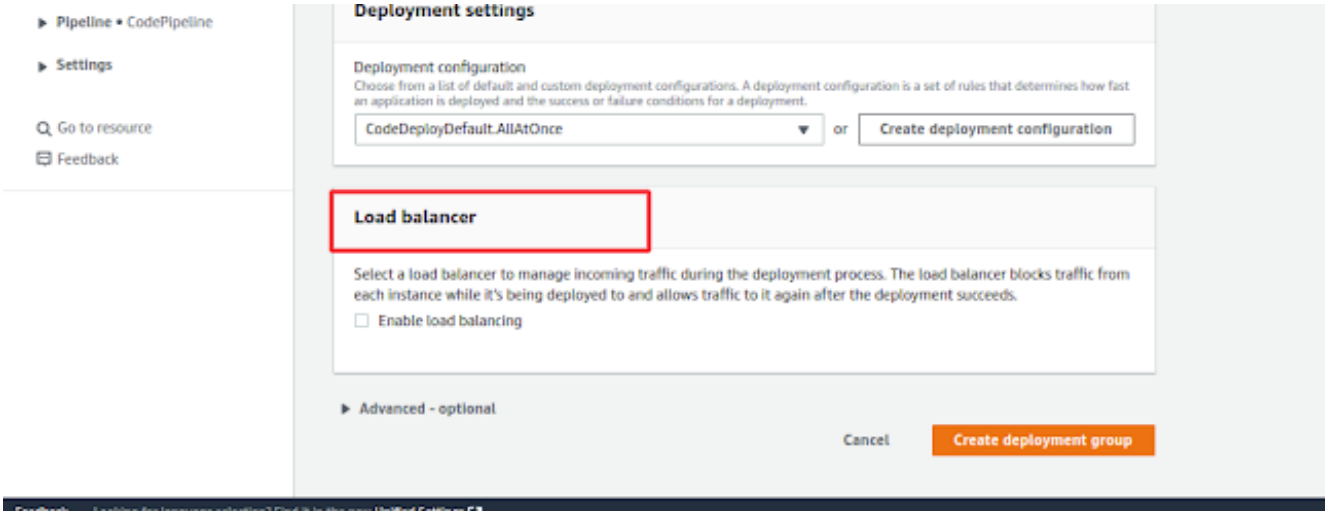




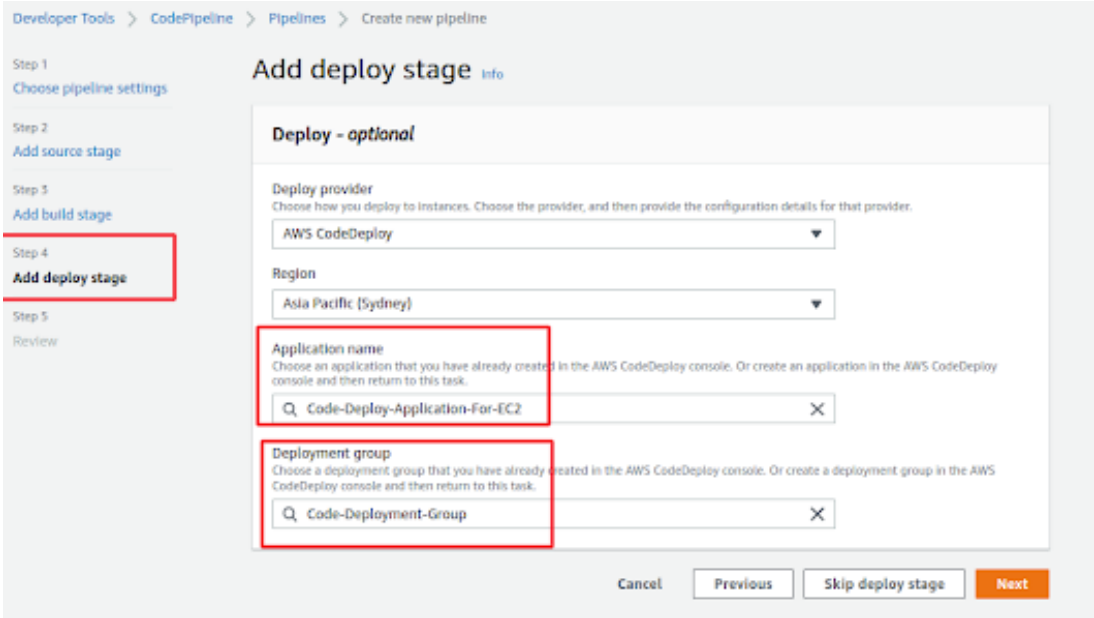
In deployment group Select EC2 instances and select Tag and Value



Untick Load Blancer Option



and Finally Come on Add Deploy Stage and select that created Application name & Deployment group



e) At the end, just review and create.

If You will Get below error in pipeline then follow further instruction to resolve it.

> **Error :** CodeDeploy agent was not able to receive the lifecycle event. Check the CodeDeploy agent logs on your host and make sure the agent is running and can connect to the CodeDeploy server.

Conclusion: Its happening Because we installed codedeployment-agent before attaching ec2role to EC2 Instance, and error will be resolve when you will first attach ec2role and then install codedeployment-agent on EC2.

Run the following commands.

```
sudo systemctl stop codedeploy-agent

sudo yum erase codedeploy-agent -y

sudo rm -rf /opt/codedeploy-agent

cd /home/ec2-user/server

sudo ./install auto

sudo systemctl status codedeploy-agent
```