Deploying web app using Jenkins CI/CD declarative pipeline.

For the code you can refer this repository (https://github.com/rohit808077/node-todo-cicd)

Steps Overview:

1. Set Up AWS EC2 Instance

2. Install Docker on EC2 Instance

3. Prepare Docker Compose File

4. Set Up Jenkins

5. Configure Jenkins Pipeline

6. GitHub Webhook Integration

7. Test the Pipeline

1. Set Up AWS EC2 Instance:

- **Create an AWS Account:**

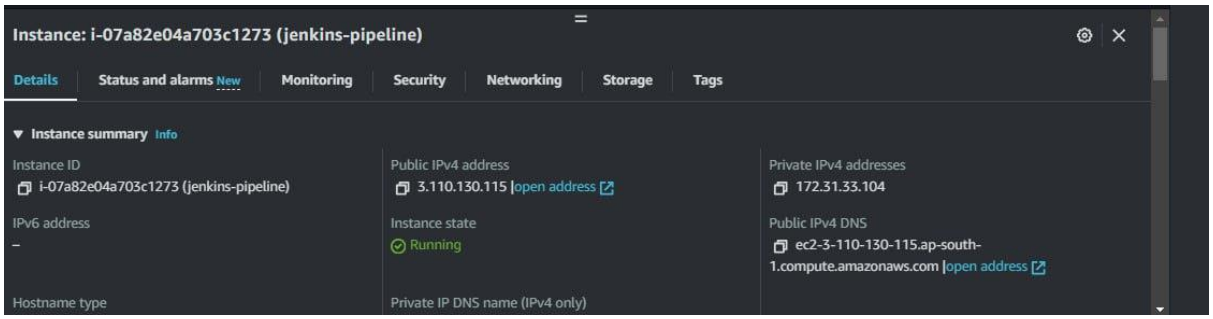
  - Go to the AWS website and sign in or create an account.

- **Launch EC2 Instance:**

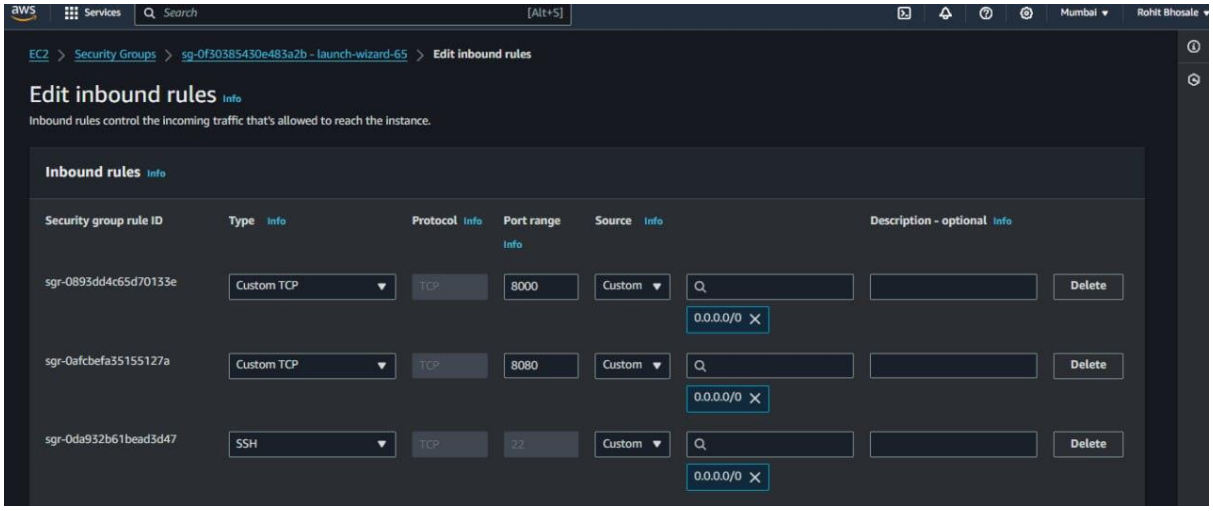
  - Go to the EC2 dashboard in the AWS Management Console.

  - Click "Launch Instance" and choose an Amazon Machine Image (AMI) (e.g., Amazon Linux, Ubuntu).

  - Select an instance type, configure instance details (like network settings, security groups, SSH key pairs), and launch the instance.



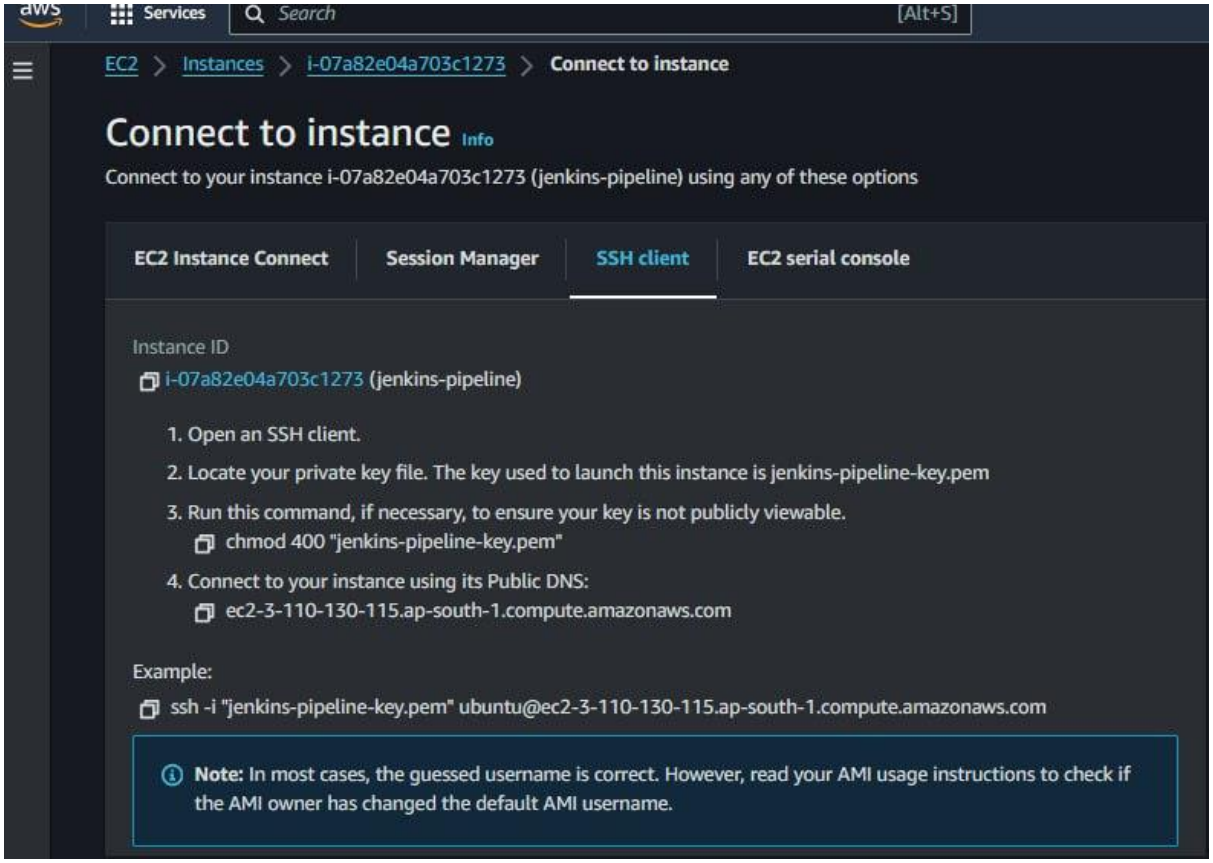
Security Group Configuration



2. Install Docker on EC2 Instance:

- **Connect to EC2 Instance:**

  - Use SSH to connect to your EC2 instance. For example: `ssh -i your-key.pem ec2-user@your-instance-ip`



- **Install Docker:**

  - Update the system: `sudo yum update` (for Amazon Linux) or `sudo apt-get update` (for Ubuntu).

- Install Docker: foll

ow Below commands

docker installtion command

#sudo apt-get install docker.io -y

#sudo apt-get install docker-compose -y

give access to user for docker

#sudo usermod -aG docker ubuntu

reboot your system

#sudo rebbot

To verify that you have docker permission

#docker ps

```
Last login: Mon Jan  8 07:36:53 2024 from 103.163.91.4
ubuntu@ip-172-31-33-104:~$  docker --version
Docker version 24.0.5, build 24.0.5-0ubuntu1~22.04.1
ubuntu@ip-172-31-33-104:~$
```

3. Prepare Docker Compose File:

- **Create `docker-compose.yml`:**

- Define services for your Todo list app, specifying the Node.js application, databases (if any), environment variables, ports, volumes, etc. Refer to the [Docker Compose file reference](https://docs.docker.com/compose/compose-file/) for details.

```
ubuntu@ip-172-31-33-104:~$ docker-compose --version
docker-compose version 1.29.2, build unknown
ubuntu@ip-172-31-33-104:~$
```

4. Set Up Jenkins:

- **Install Jenkins:**

- Follow the official Jenkins installation guide for your operating system: [Jenkins Installation Guide](https://www.jenkins.io/doc/book/installing/linux/).

```
ubuntu@ip-172-31-33-104:~$ java --version
openjdk 17.0.9 2023-10-17
OpenJDK Runtime Environment (build 17.0.9+9-Ubuntu-122.04)
OpenJDK 64-Bit Server VM (build 17.0.9+9-Ubuntu-122.04, mixed mode, sharing)
ubuntu@ip-172-31-33-104:~$
```

```
ubuntu@ip-172-31-33-104:~$ jenkins --version
2.439
ubuntu@ip-172-31-33-104:~$
```

Perform the following command after installing Jenkins

sudo usermod -ag docker jenkins

reboot your system

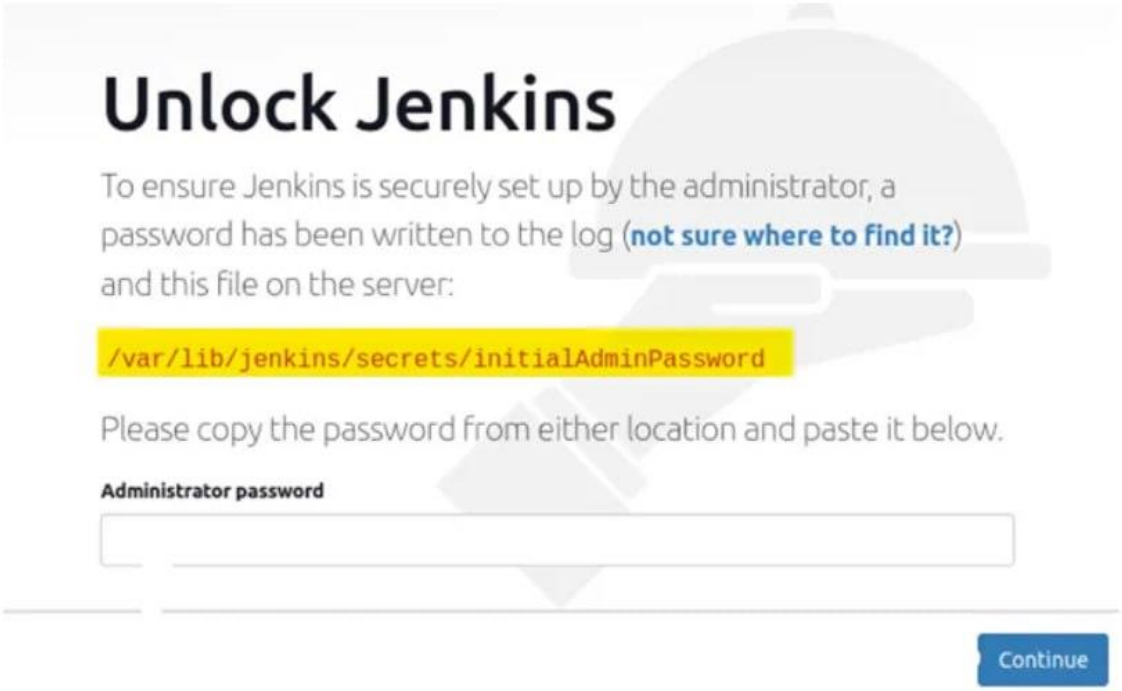
#sudo rebbot

- **Configure Jenkins:**

- Access Jenkins in a web browser (`http://your-instance-ip:8080`) and follow the setup wizard to unlock Jenkins and install recommended plugins.

To access the password to unlock Jenkins use following command

sudo cat /var/lib/Jenkins/secrets/initialAdminPassword



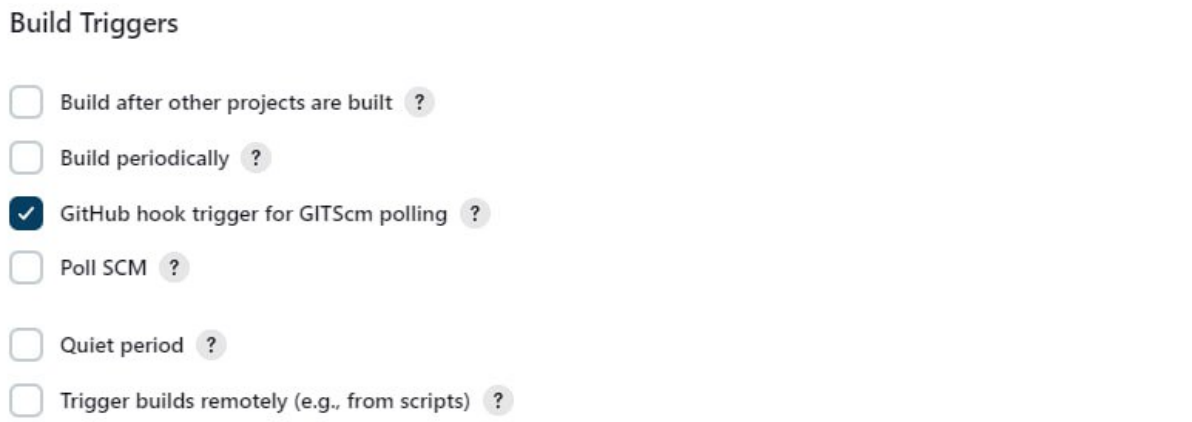
- **Configure Jenkins settings, including user accounts, security, and global tool configurations if necessary.**

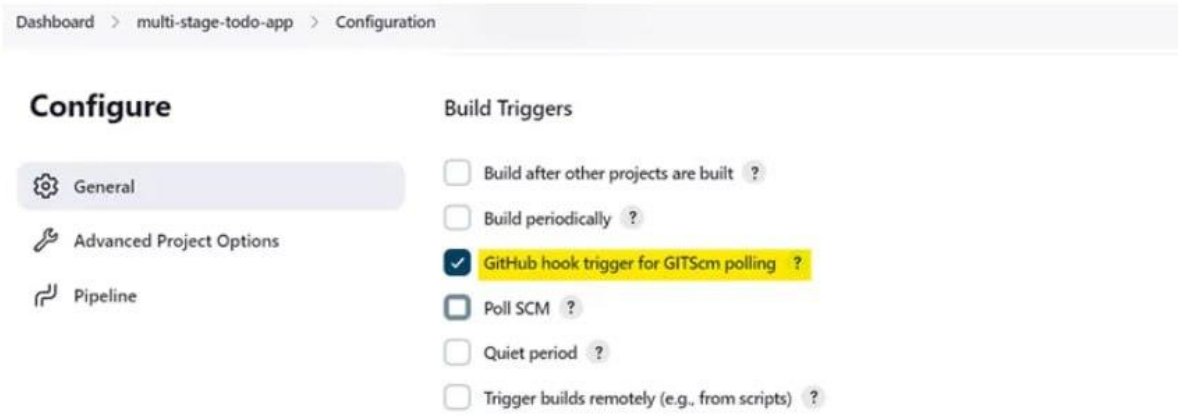
5. Configure Jenkins Pipeline:

- **Create a New Pipeline Project:**

  - Go to Jenkins dashboard and click "New Item" to create a new pipeline project.

  - Select "Pipeline" and enter a name for your project. Please get the pipeline code for reference






```
#### pipeline {
  agent any

  stages {

    stage("code"){
      steps{
        git url: "https://github.com/rohit808077/node-todo-cicd", branch: "master"
```

```
            echo 'bhaiyya code clone ho gaya'
        }
    }

    stage("build & test"){
        steps{
            sh "docker build -t node-todo-app ."
            echo 'code build & test ho gaya'
        }
    }

    stage("push on dockerhub") {
  steps {
    withCredentials([usernamePassword(credentialsId: "dockerhub", passwordVariable:
"dockerhubPass", usernameVariable: "dockerhubUser")]) {
        script {
            sh "docker login -u ${env.dockerhubUser} -p ${env.dockerhubPass}"
            sh "docker tag node-todo-app:latest ${env.dockerhubUser}/node-todo-app:latest"
            sh "docker push ${env.dockerhubUser}/node-todo-app:latest"
            echo "Image push completed"
        }
    }
  }
}

    stage("deploy on docker container "){
        steps{
            sh "docker-compose down && docker-compose up -d"
            echo 'code build & test ho gaya'
        }
    }
```

} ###

- **Write Jenkinsfile:**

  - Create a `Jenkinsfile` in the root of your project repository.

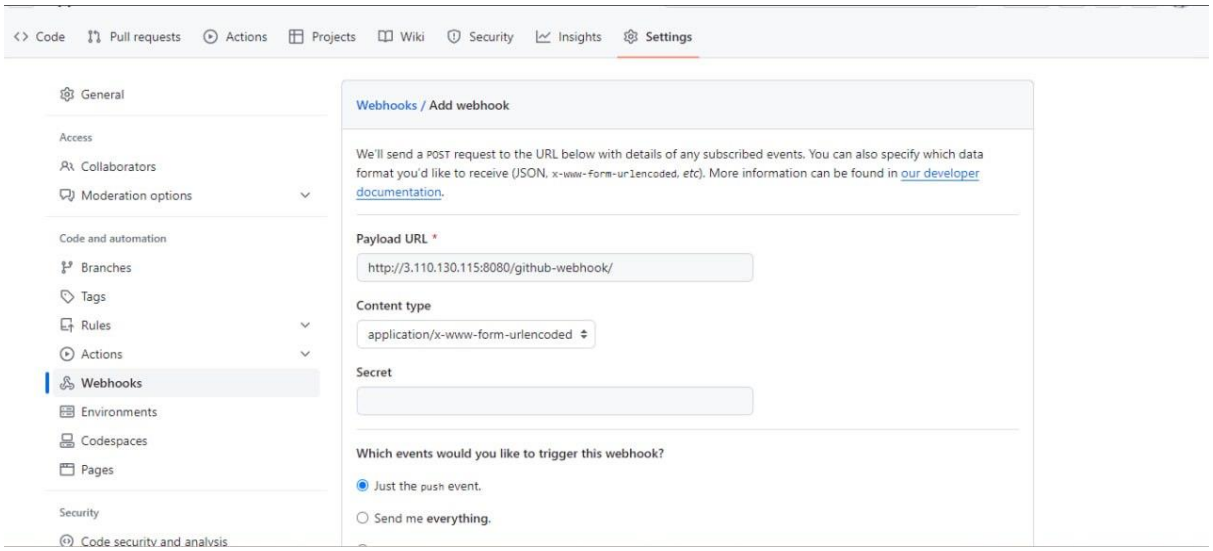
  - Write stages for checkout, build, test, Docker image creation, and deployment using Docker Compose. Use `node` blocks for Node.js-related commands.

  - Utilize Jenkins pipeline syntax and Docker commands (`docker build`, `docker-compose`, etc.) within the stages.

6. GitHub Webhook Integration:

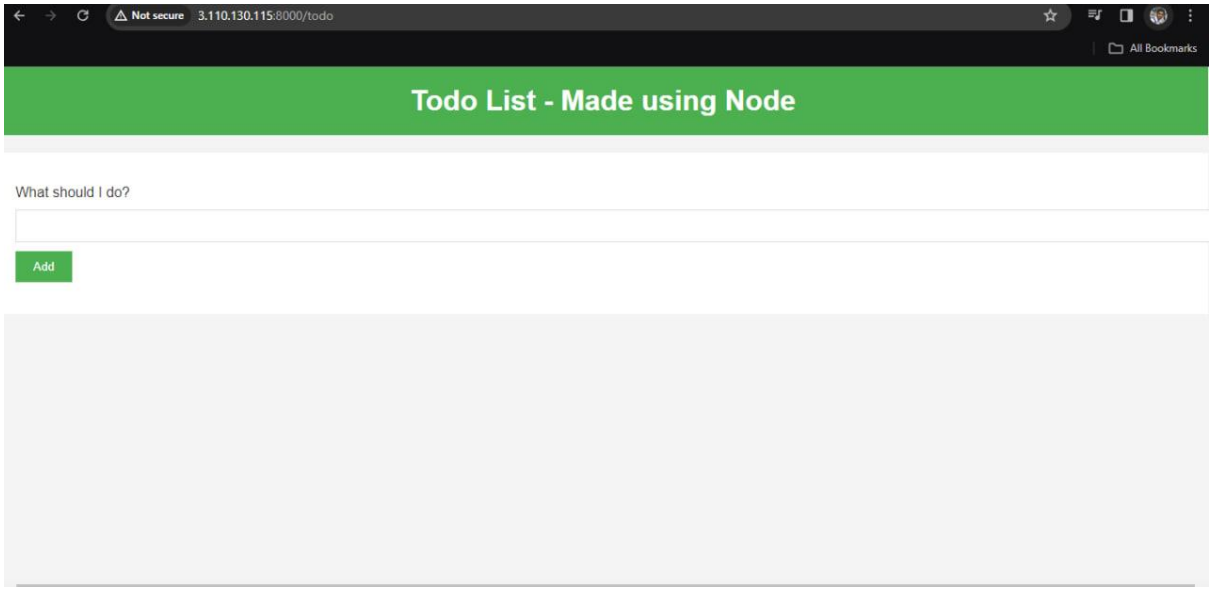
- **Configure Webhook in GitHub:**

  - In your GitHub repository, go to Settings > Webhooks.



  - Add a new webhook, provide the Jenkins endpoint URL (http://your-jenkins-server/github-webhook), and configure the webhook payload and events to trigger the Jenkins pipeline.

7. Test the Pipeline:



- **Push Changes to GitHub:**

  - Make changes to your codebase and push them to your GitHub repository.

- **Monitor Jenkins Execution:**

  - Monitor the Jenkins dashboard to observe the triggered pipeline job.

  - Check console logs and build stages to ensure successful execution, including code checkout, build, tests, Docker image creation, and deployment.