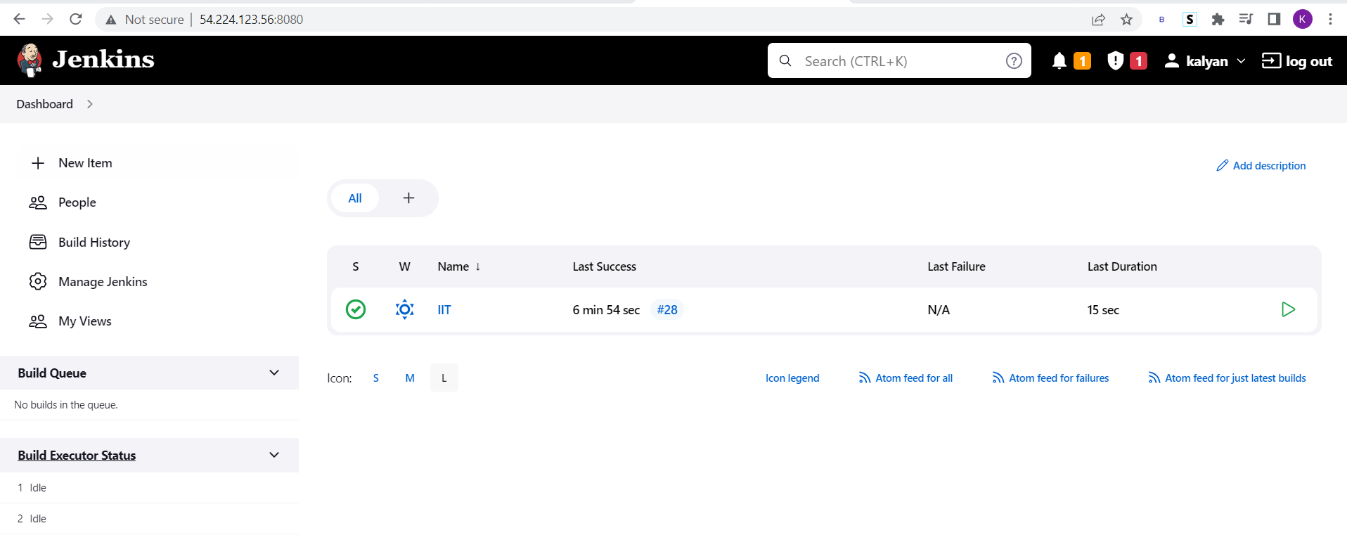
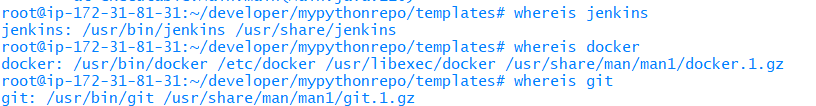
DevOps Engineer Assessment

(CI/CD Pipeline)

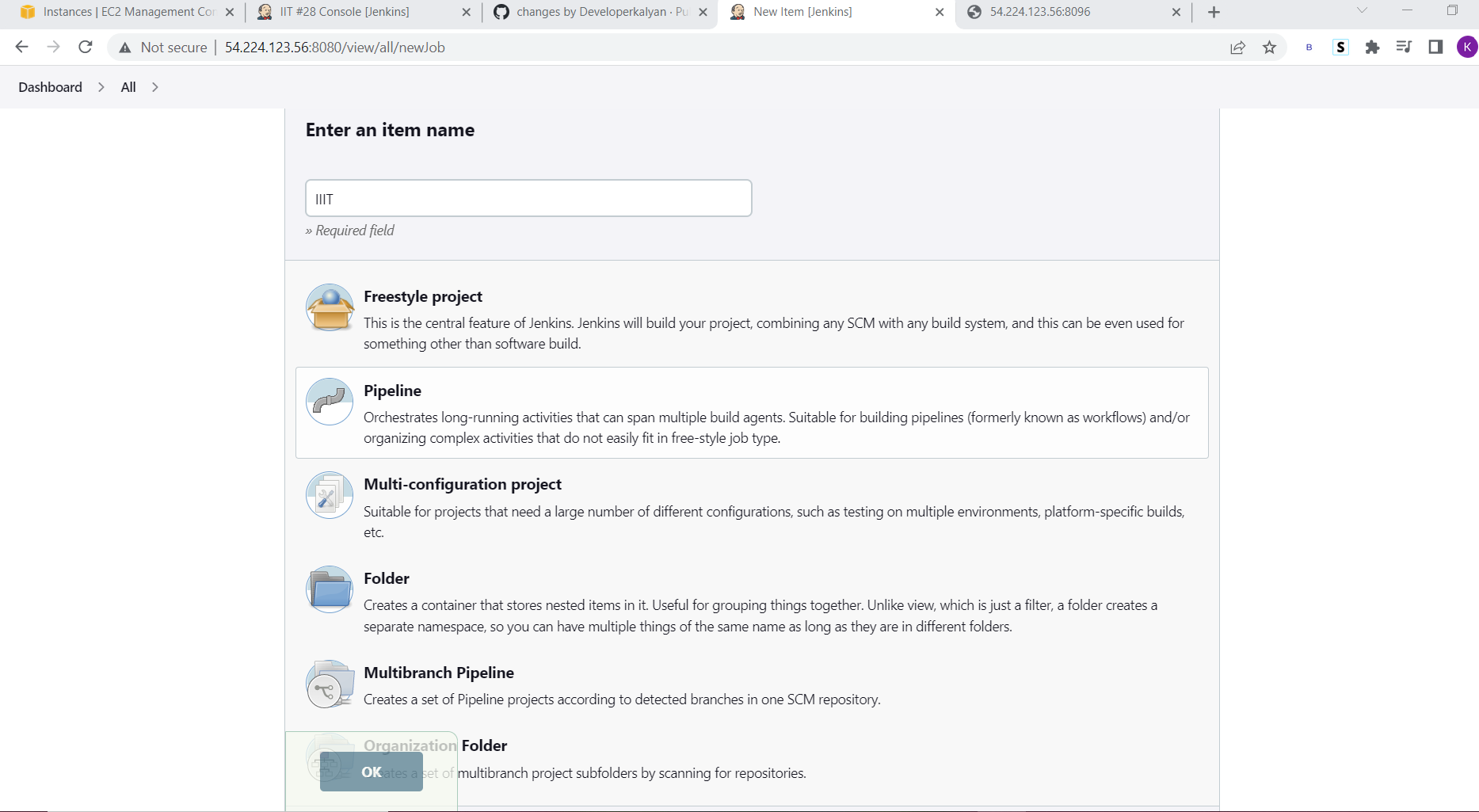
Step 1: Create Ec2 Instance in AWS cloud where I have Install jenkin, Docker and Git





Step 2: Create Pipeline Project

New item > Enter Project name>select pipeline



Configure the Project:

Write a pipeline script for git checkout The pipeline should automatically build and create a Docker image using the Dockerfile already available in the project and Run the docker Image

## Write a Pipeline script for automate the whole process

## **Here we can find whole pipeline script:**

pipeline {

agent any

environment {

registry = "kalyandocker304/mypythonapp"

}

stages {

stage('Cloning Git') {

steps {

checkout([$class: 'GitSCM', branches: [[name: '\*/master']], extensions: [], userRemoteConfigs: [[url: 'https://github.com/kalyanramudu/mypythonrepo.git']]])

}

}

// Building Docker images

stage('Building image') {

steps{

script {

dockerImage = docker.build registry

}

}

}

stage('docker stop container') {

steps {

sh 'docker ps -f name=mypythonappContainer -q | xargs --no-run-if-empty docker container stop'

sh 'docker container ls -a -fname=mypythonappContainer -q | xargs -r docker container rm'

}

}

// Running Docker container, make sure port 8096 is opened in

stage('Docker Run') {

steps{

script {

dockerImage.run("-p 8096:5000 --rm --name mypythonappContainer")

}

}

}

}

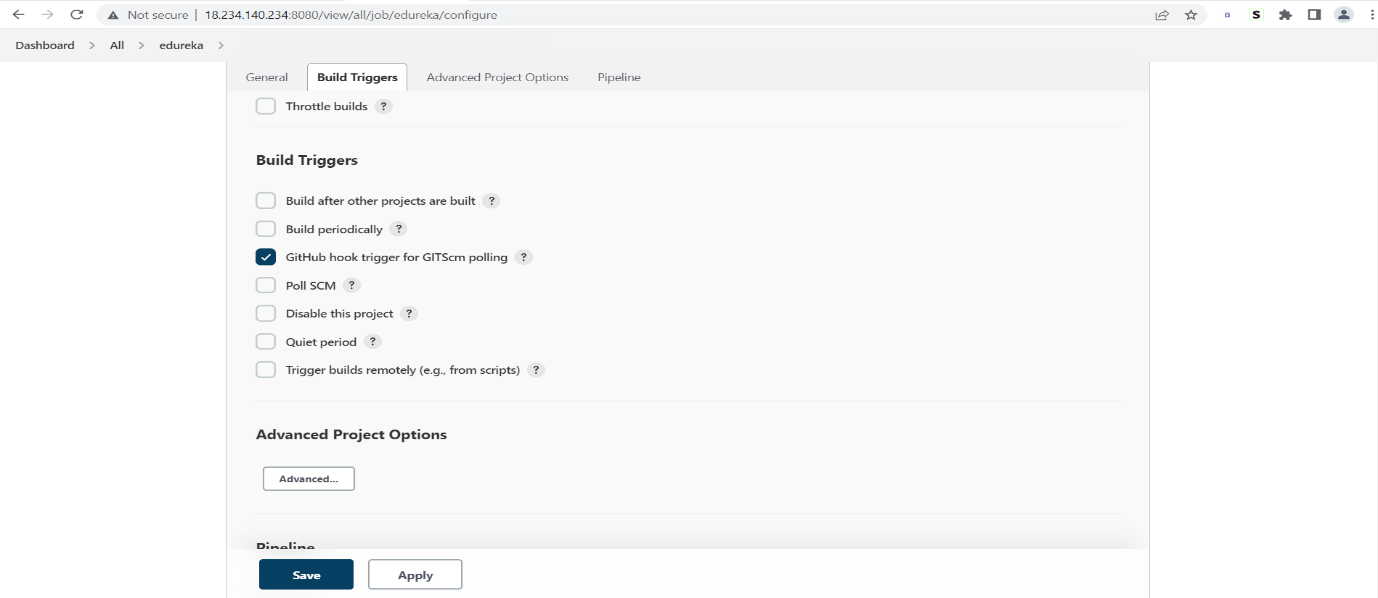
}

## **Automatically Build the trigger when made any changes in git for this:**

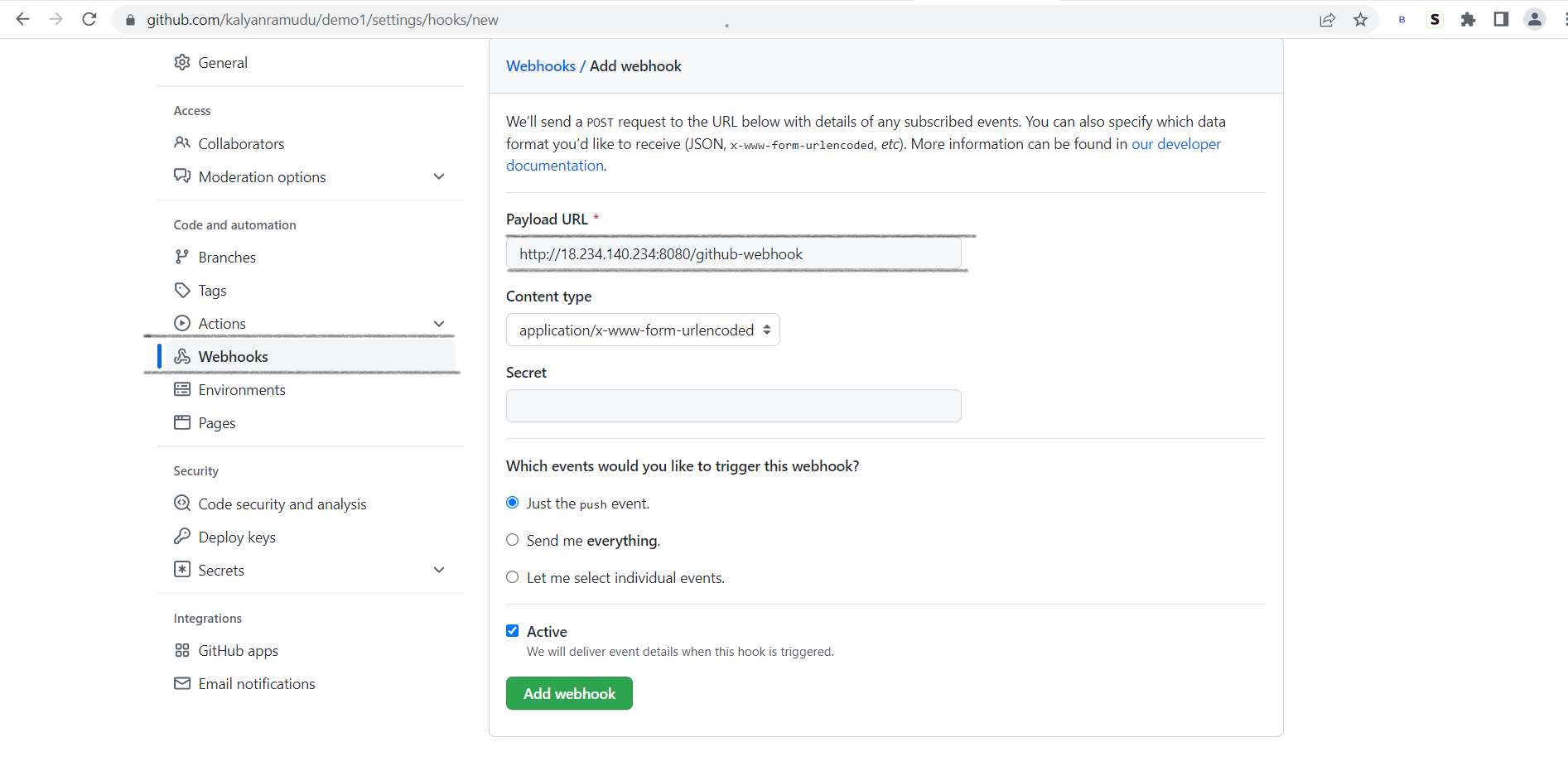
## Step1:Go to Jenkins >Select Project >Build Trigger>selectGitHub hook trigger for GITScm polling in Jenkins

## Step1:Git hub>repository>setting>webhook>add webhook

## Specify the url that is : <jenkin url>/github-webhook/



## Fig: selectGitHub hook trigger for GITScm polling in Jenkins

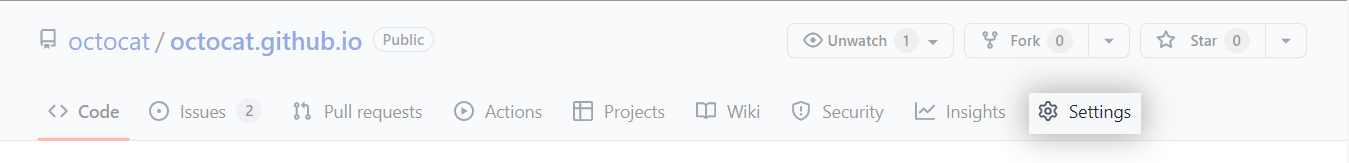
Fig: Add webhook.

# Managing a branch protection rule

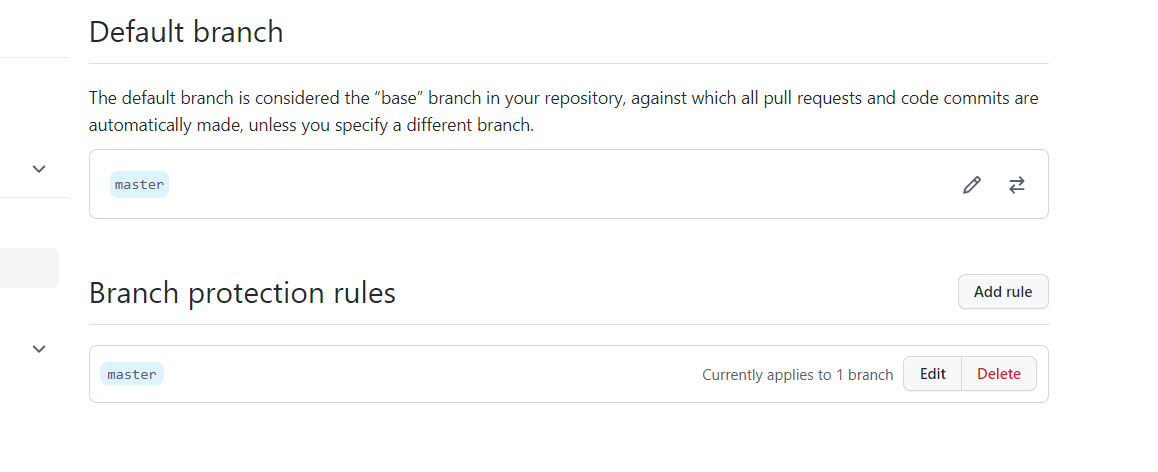
You can create a branch protection rule to enforce certain workflows for one or more branches, such as requiring an approving review or passing status checks for all pull requests merged into the protected branch.

## Creating a branch protection rule:

1. On GitHub.com, navigate to the main page of the repository.
2. Under your repository name, click **Settings.**



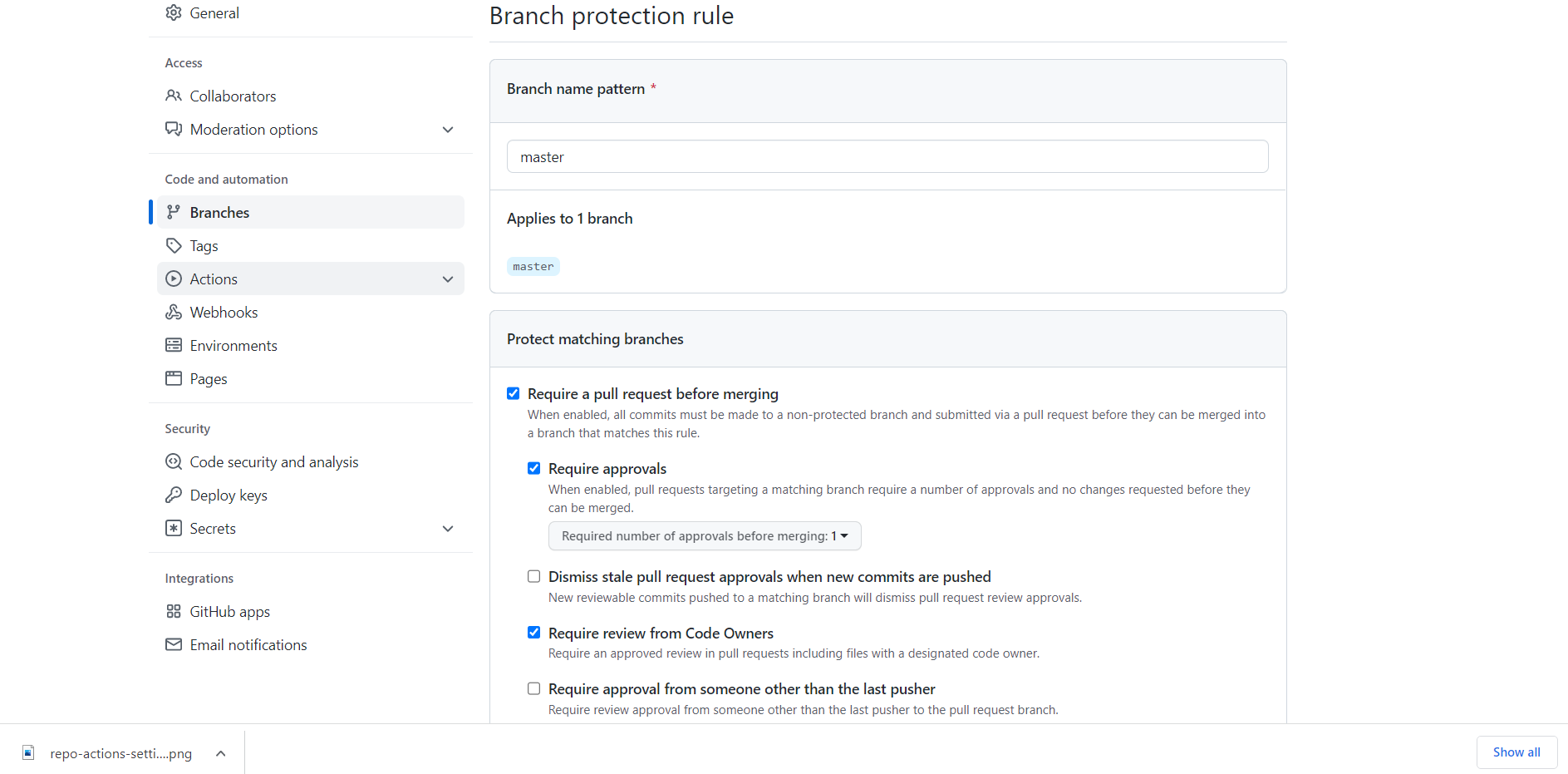
1. In the "Code and automation" section of the sidebar, click Branches.
2. Next to "Branch protection rules", click **Add rule.**



**5**, Under "Branch name pattern", type the branch name or pattern you want to protect.

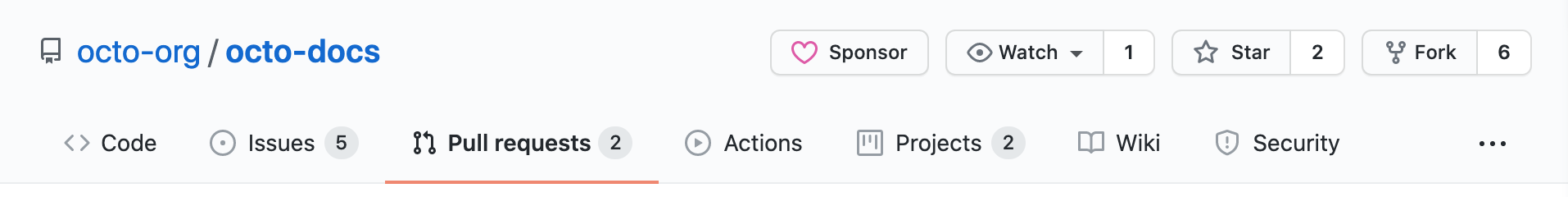
## **Branch protection rule**

.



# Approving a pull request with required reviews:

1.Under your repository name, click Pull **requests**.

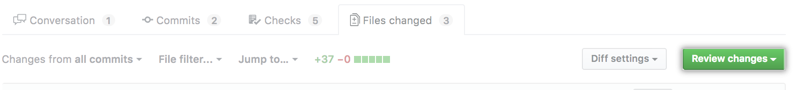


2.Under your repository name, click Pull **requests**.

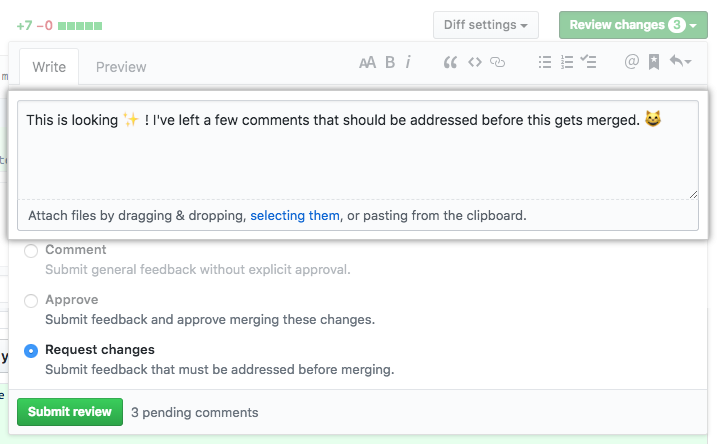
3.on the pull request, click Files **changed**.

4.Review the changes in the pull request

5.Above the changed code, click **Review changes**.



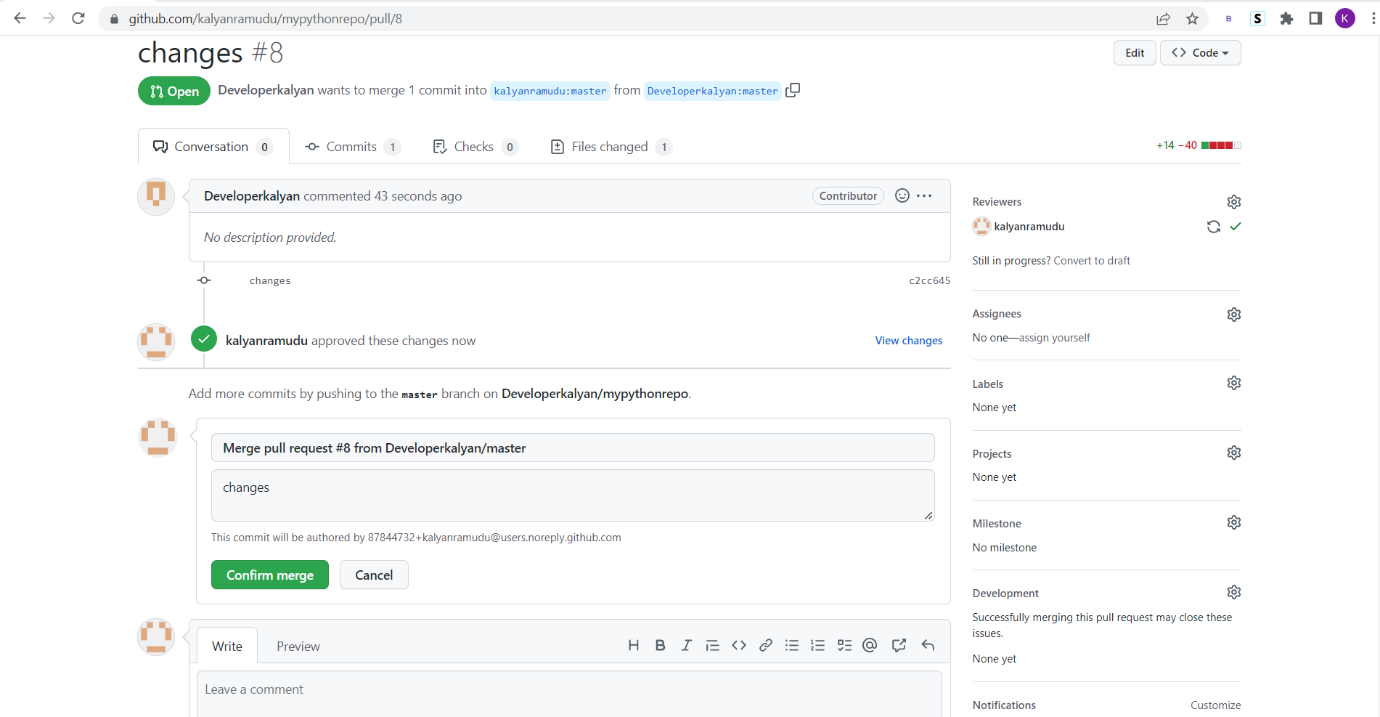
6. Type a comment summarizing your feedback on the proposed changes.



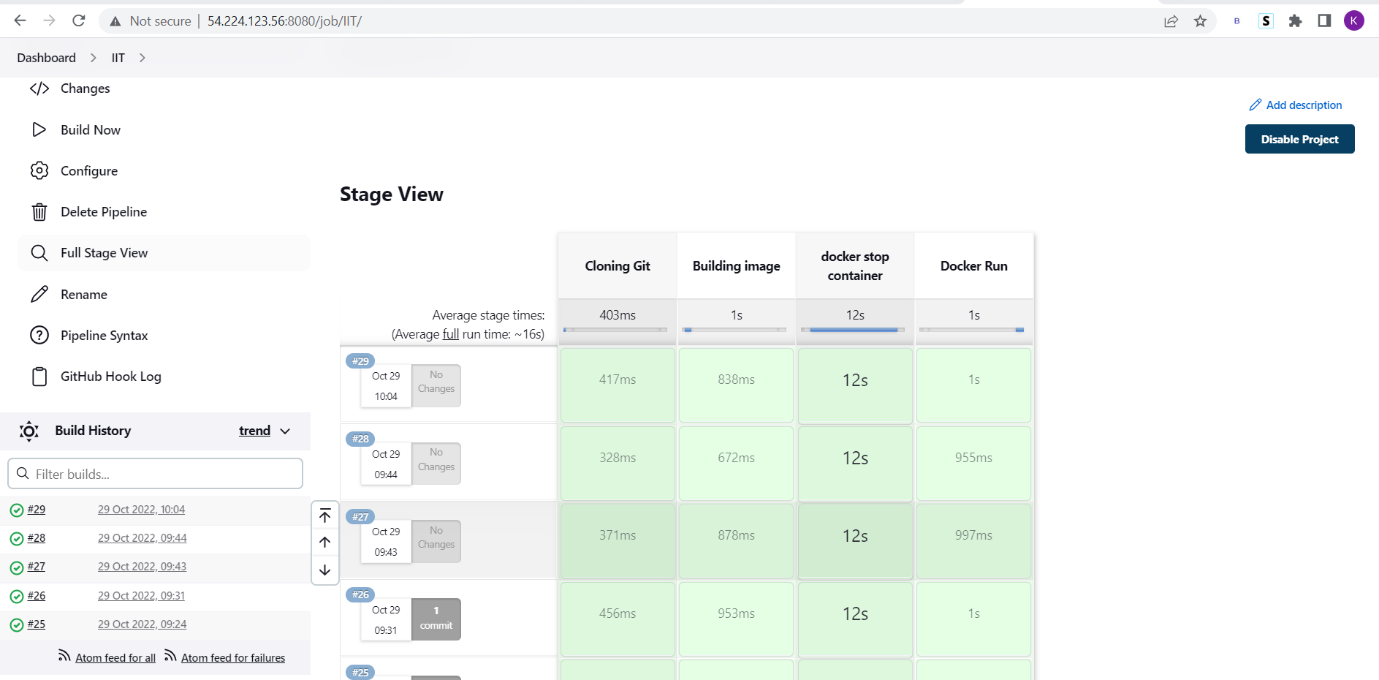
7. Select **Approve** to approve merging the changes proposed in the pull request.

8.Click **Submit review**.

9. Click **Confirm merge**,

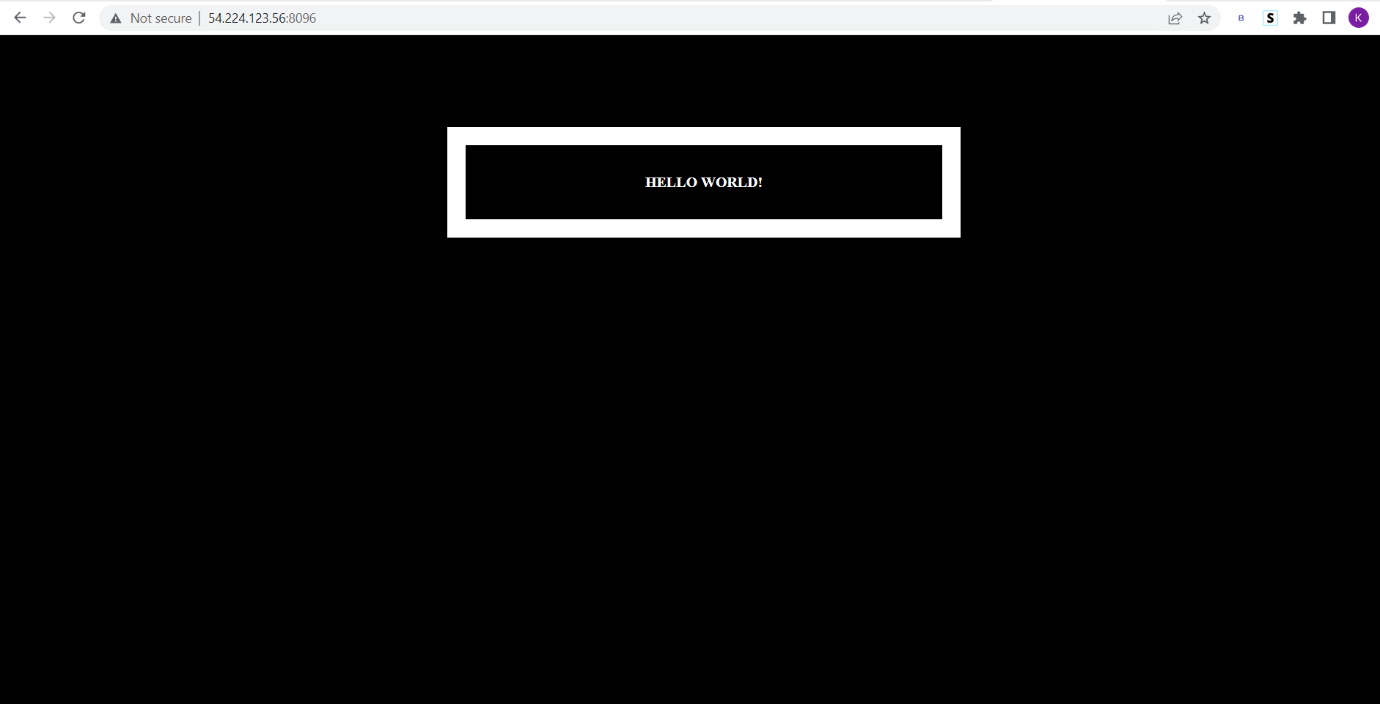


## Trigger Builds Automatically on GitHub Pull Request



## Search with Ip address of Instance with port number :8096

<http://54.224.123.56:8096/>



## Thank You