# CI_CD Project

Ayman Hesham

Email: ayman19982016@outlook.com

Github:
https://github.com/Ayman1231/Booster_CI_CD_Project

Project requirements:

Create CI/CD pipeline using jenkinsfile to deploy simple django web app as a microservice running on docker container locally.

# Steps

1- Fork this repo to your account

2- write dockerfile inside the forked repo to create new image from base image ubuntu and install python3.6 and pip3 and copy the source code files of the app to this image and configure it to start the server when creating container (check the below section for steps to start the django server)

3- configure ubuntu slave to use it for the pipeline

4- create slck workspace and integrate it with jenkins

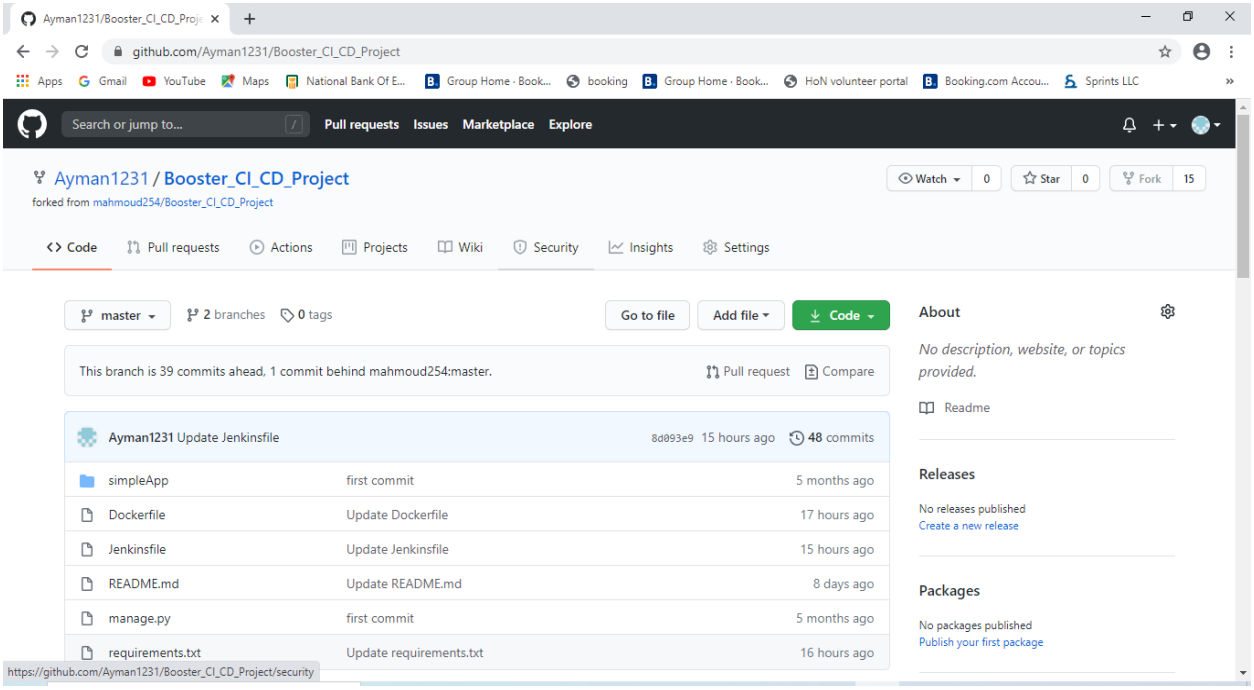
5- install any plugin from your choice to create statistics about builds

6- write jenkinsfile with the following four stages for both dev and master branch

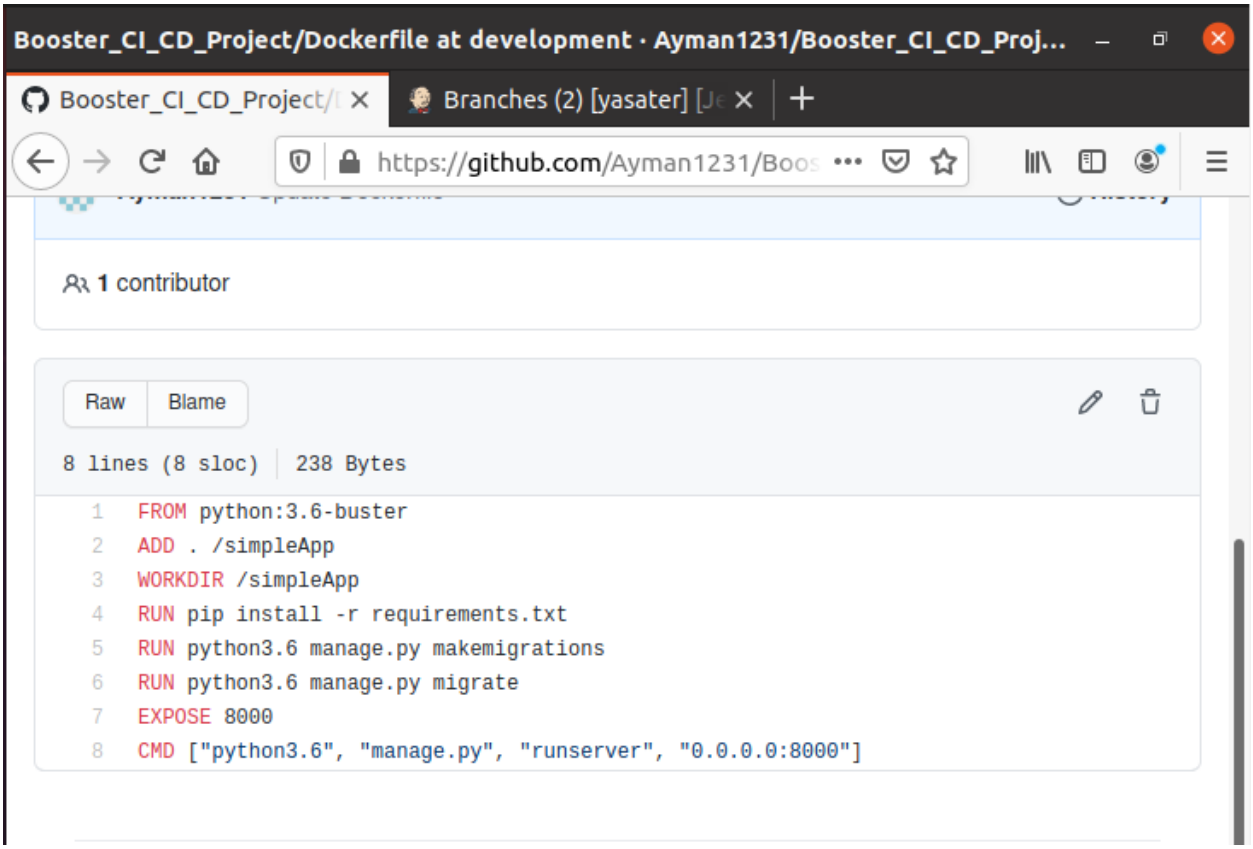
- preparation: checkout the code
- build image: build image using the dockerfile
- push image: push the built image to docker registry(docker hub)
- deploy: deploy a container from the pushed image
- notification: send slack message with the build status

7- configure job in jenkins using multibranch pipeline type with the forked git repo url

# Step:1) Fork this repo to your account



Step:2) write dockerfile inside the forked repo to create new image from base image ubuntu and install python3.6 and pip3 and copy the source code files of the app to this image and configure it to start the server when creating container (check the below section for steps to start the django server)



Step:3) configure ubuntu slave to use it for the pipeline

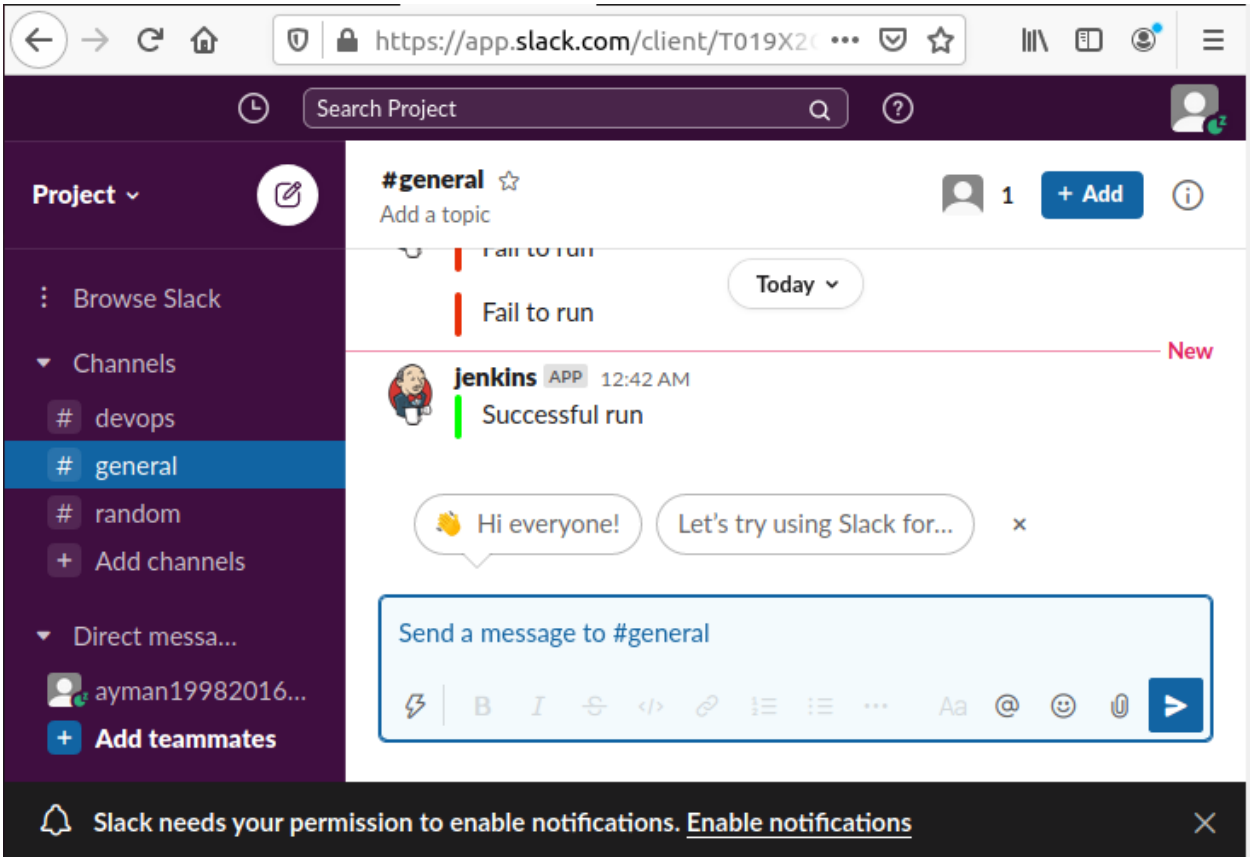
ssh-keygen (to create the private and public key)

cat id_rsa_pub (to put the value of the public key into the slave)

docker inspect bridge (to get the ip address so we can connect the master with the slave)

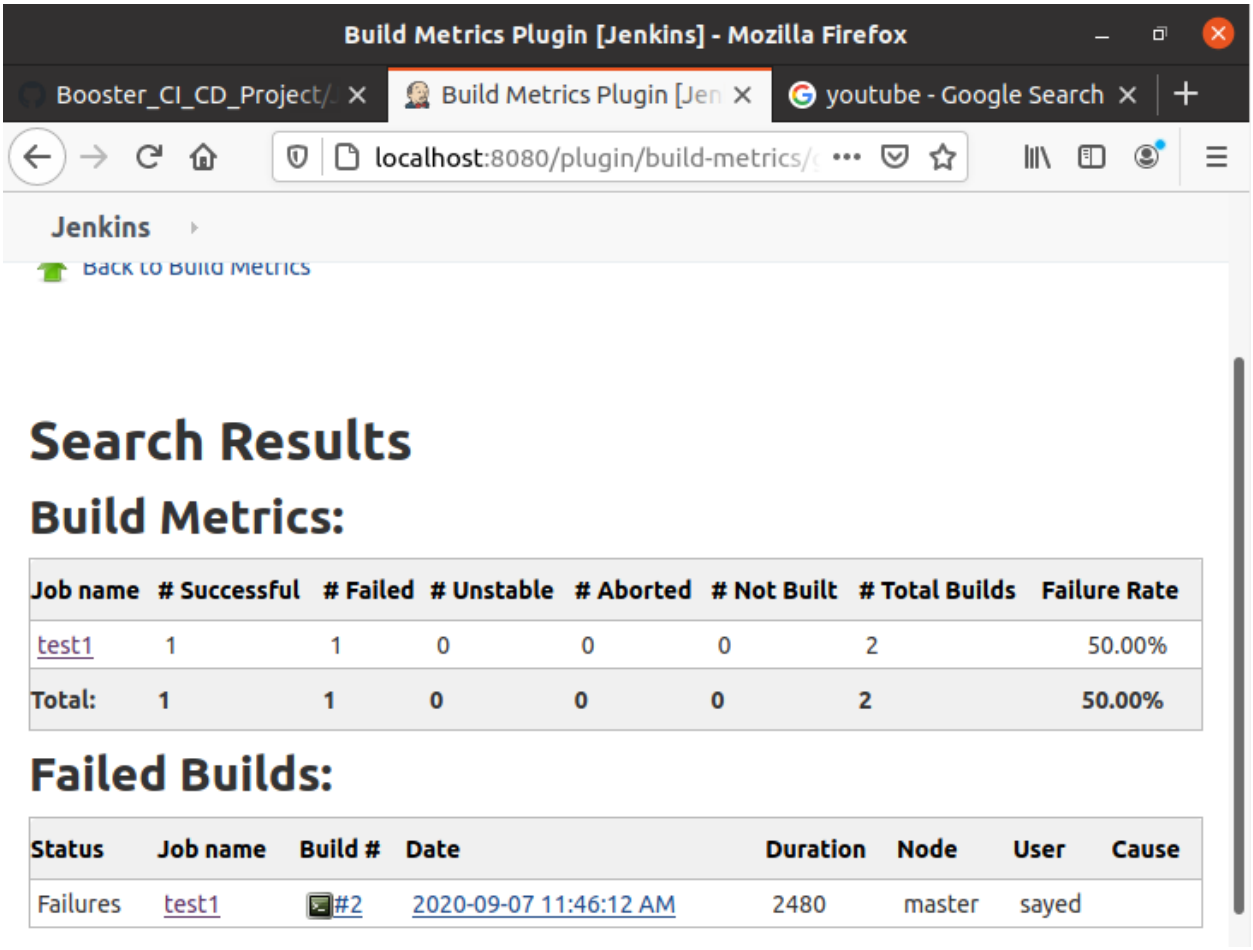
service ssh start

Step:4) create slck workspace and integrate it with jenkins



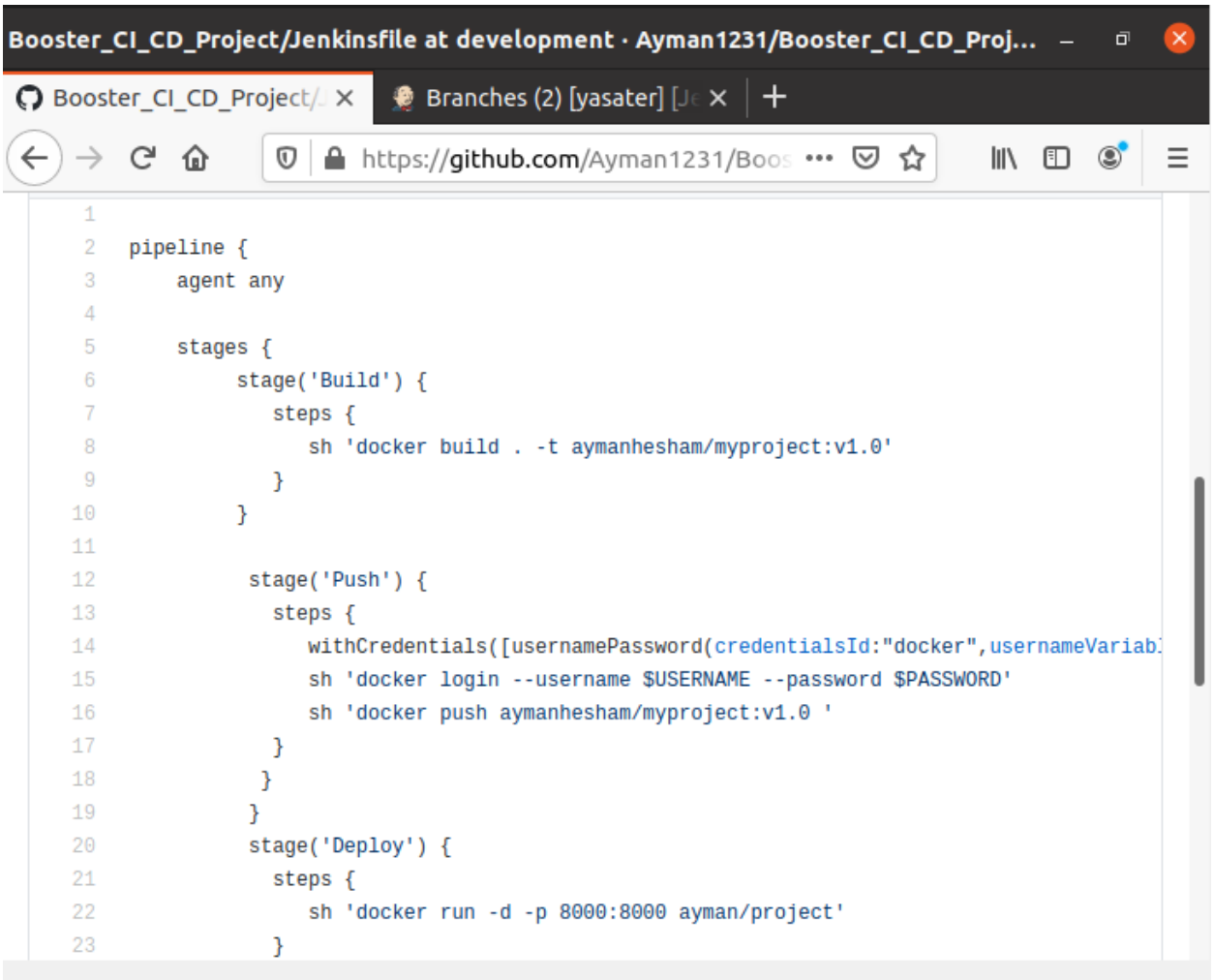
Step:5) install any plugin from your choice to create statistics about builds

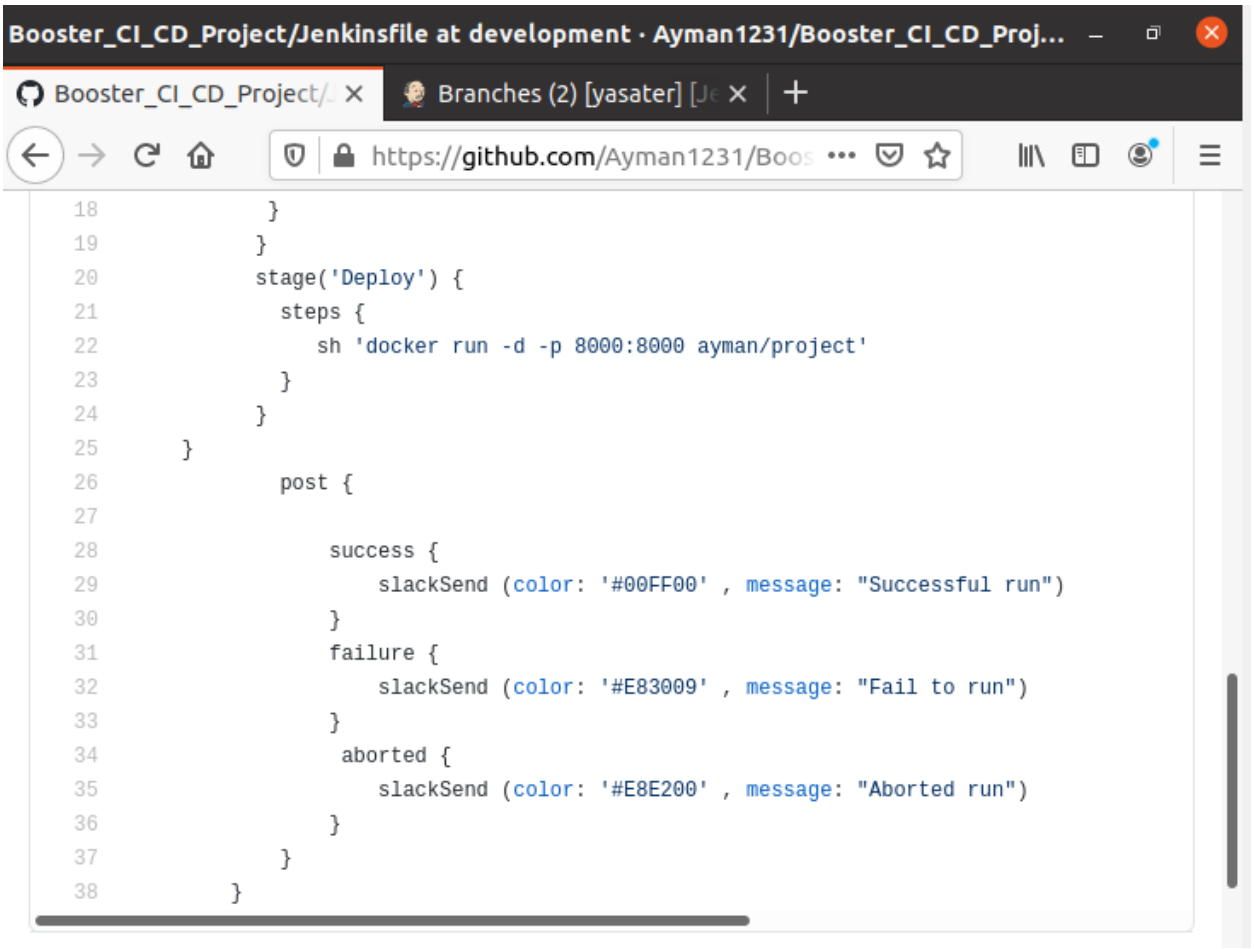
I used the Build Metrics plugin that get the stats of the project (how many builds that are successful and how many of them are failed)



Step:6) write jenkinsfile with the following four stages for both dev and master branch

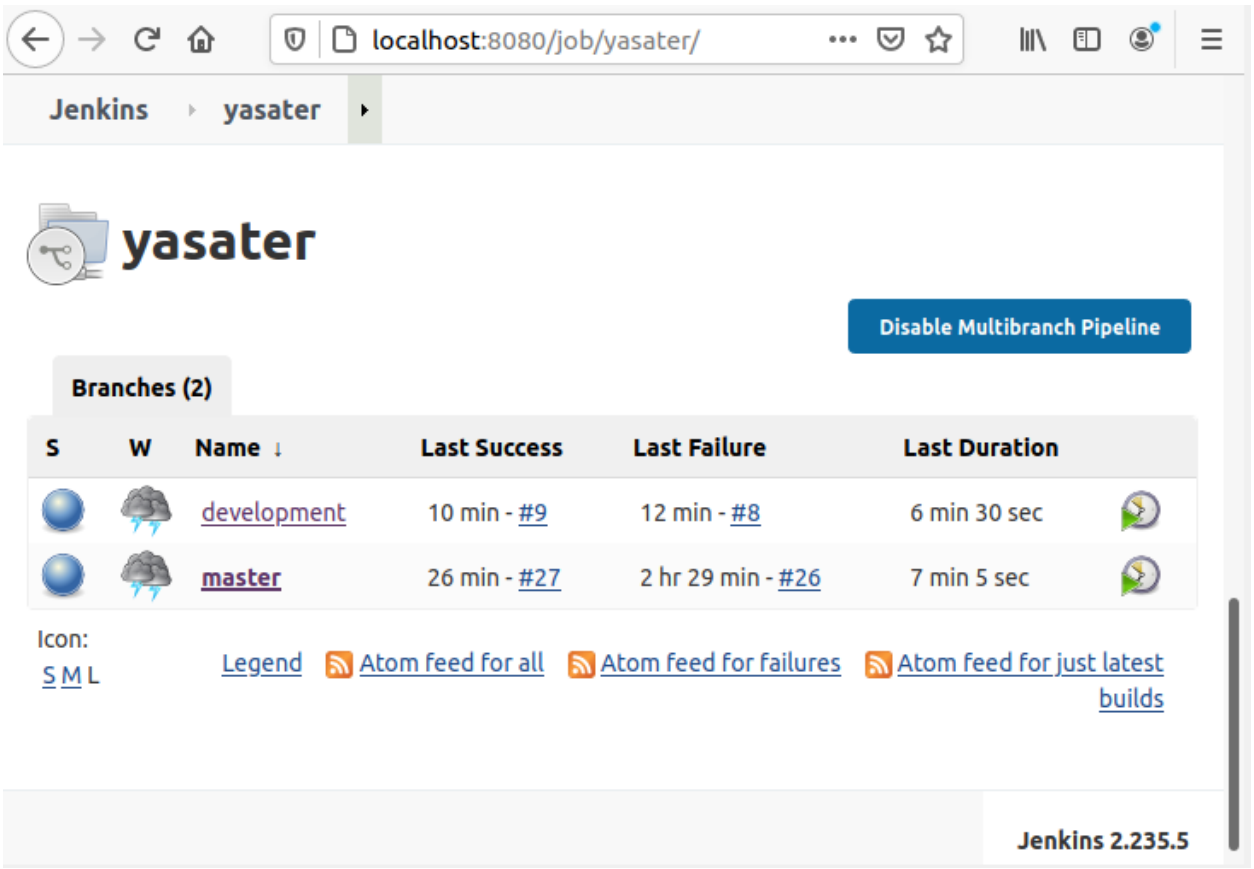
- preparation: checkout the code
- build image: build image using the dockerfile
- push image: push the built image to docker registry(docker hub)
- deploy: deploy a container from the pushed image
- notification: send slack message with the build status

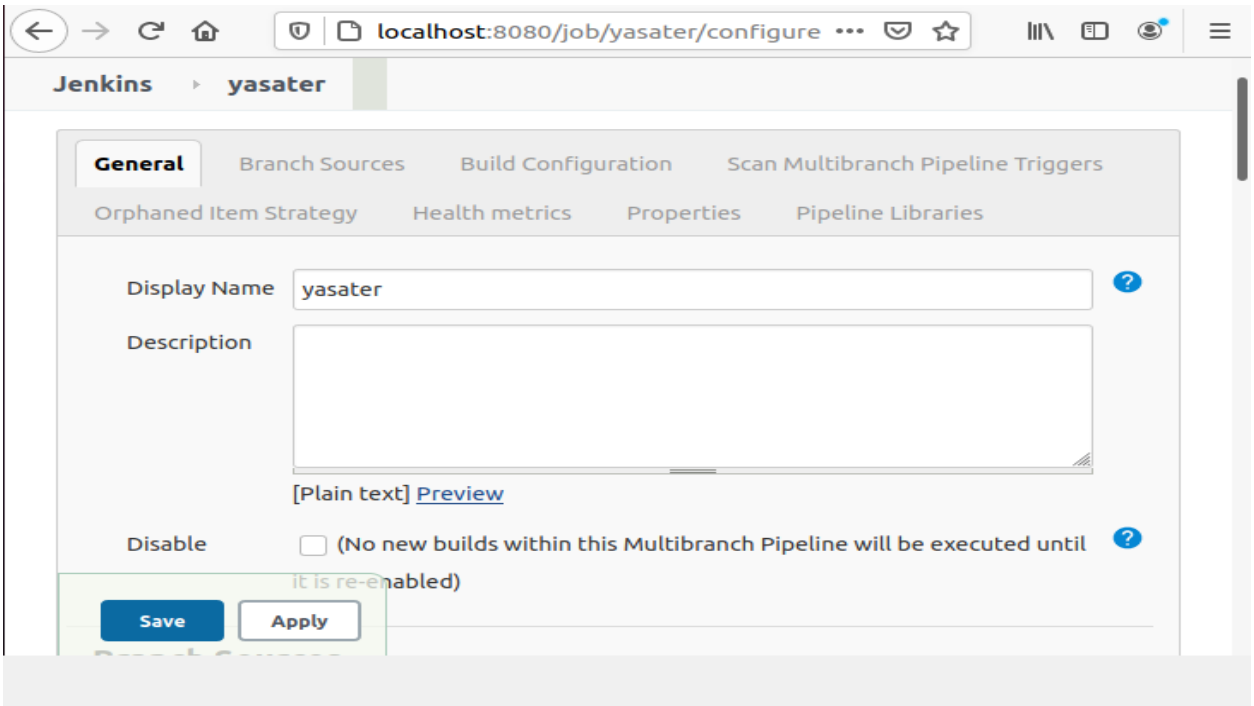


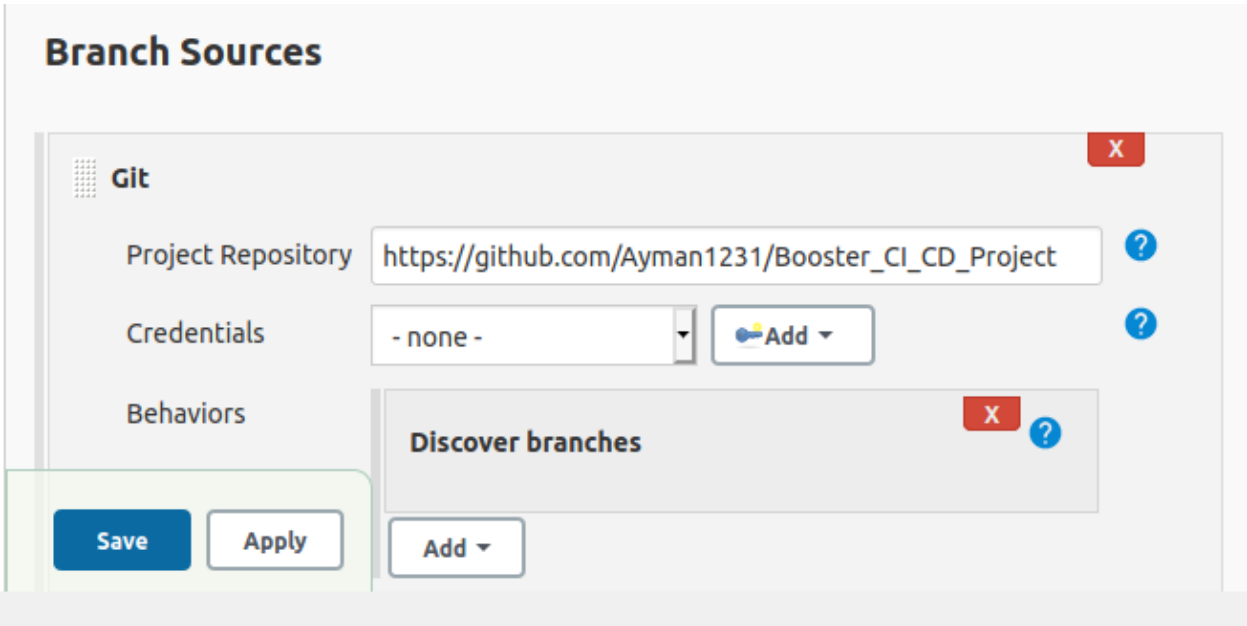



```
18            }
19          }
20          stage('Deploy') {
21            steps {
22               sh 'docker run -d -p 8000:8000 ayman/project'
23            }
24          }
25      }
26            post {
27
28                success {
29                    slackSend (color: '#00FF00' , message: "Successful run")
30                }
31                failure {
32                    slackSend (color: '#E83009' , message: "Fail to run")
33                }
34                 aborted {
35                    slackSend (color: '#E8E200' , message: "Aborted run")
36                }
37            }
38        }
```

Step:7) configure job in jenkins using multibranch pipeline type with the forked git repo url




Branch Sources
Git
Project Repository
https://github.com/Ayman1231/Booster_CI_CD_Project
Credentials
- none -
Add
Behaviors
Discover branches
Save
Apply
Add


Build Configuration
Mode
by Jenkinsfile
Script Path
Jenkinsfile