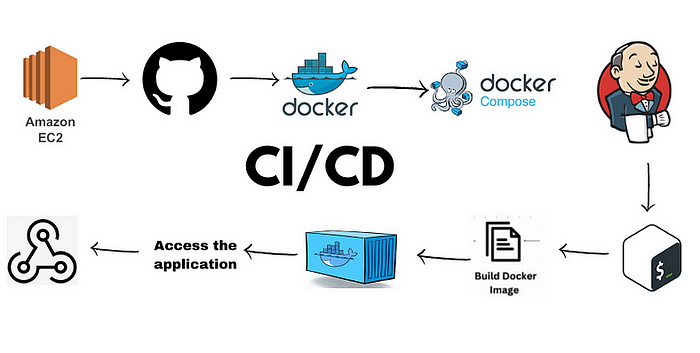
DevOps Project 1: Complete CICD Project with Docker deployed on Jenkins.



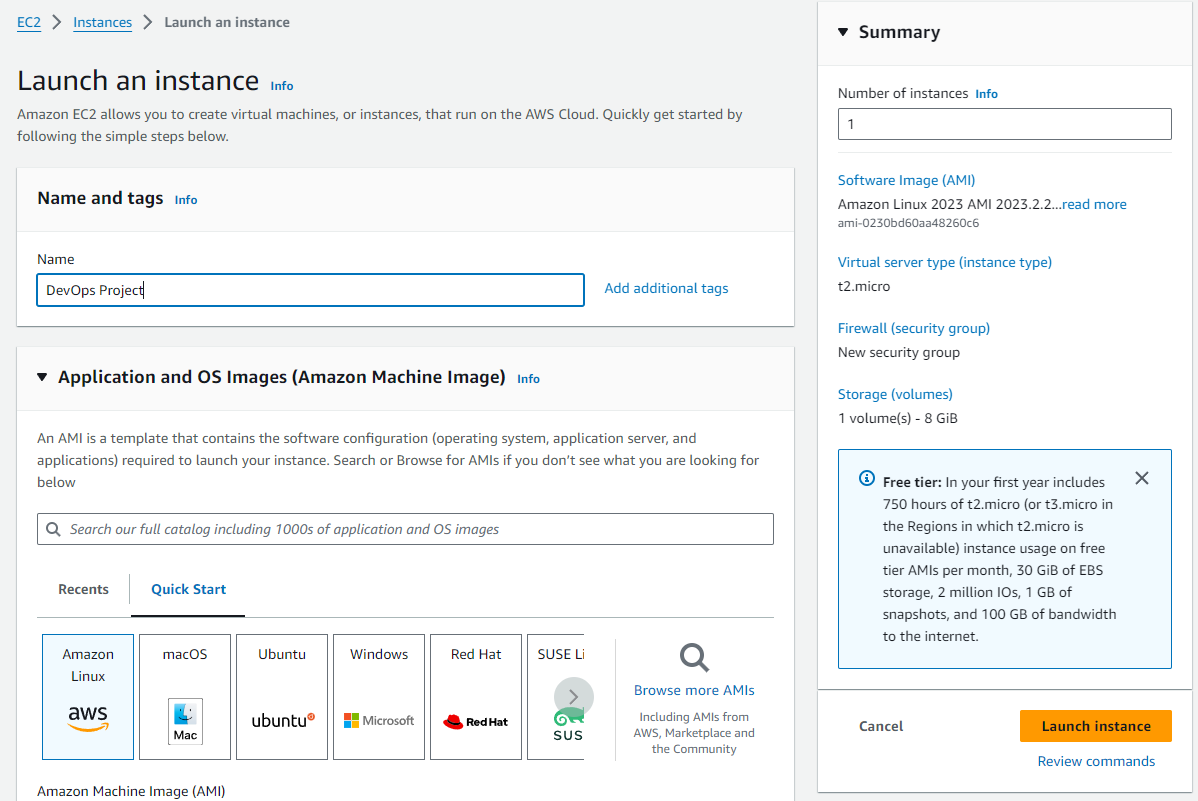
Source code: <https://github.com/aakashbshendage/DevOpsCICDProject.git>

In this project we will be deploying a node.js application on EC2 instance and we will create a CICD pipeline using Jenkins.

Tools we will be using in the project:

1. AWS-EC2
2. GitHub
3. Docker
4. Jenkins

Step 1: Create EC2 instance.



Step 2: Install Jenkins on the EC2 Server.

sudo apt-get update

sudo apt install openjdk-11-jre

sudo systemctl enable jenkins  
sudo systemctl enable jenkins

sudo systemctl status Jenkins

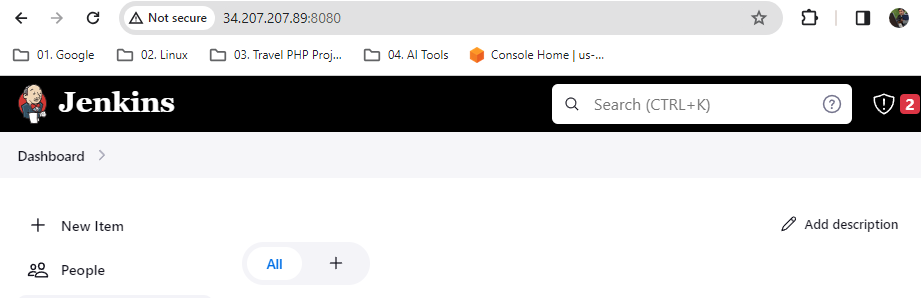
Step 3: Now add custom TCP 8080 in our security group for access Jenkins.

Now copy public IP of EC2 instance.

And paste it on browser with :8080 listener.

Now copy the administrator password from /var/lib/Jenkins/secrets/initialAdminPassword and paste it.

Now Jenkins is accessible.



Note:

Before starting with the Jenkins configurations we need to add the public key so that we can create a bridge between jenkins and Github for accessing the source code from the Github repo.

Go to your EC2 instance and type command

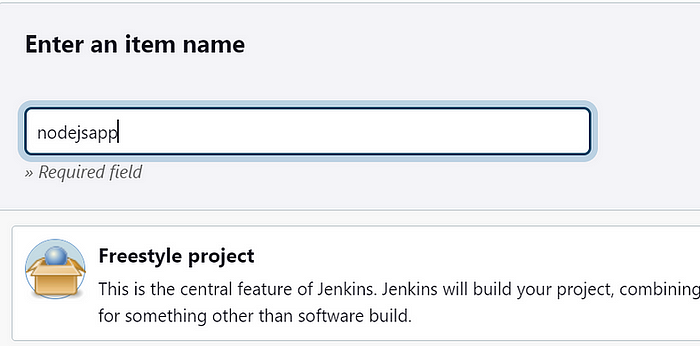
#ssh-keygen

It will create 2 keys. CD to .ssh and check the keys.

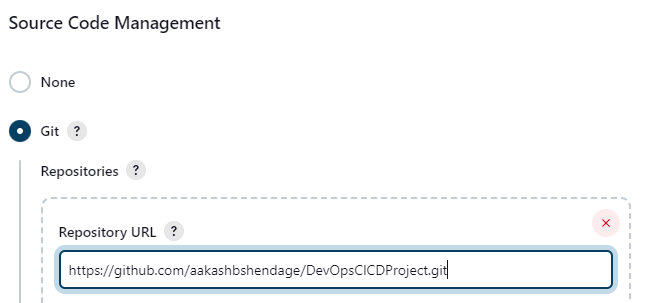
Step 4: We need to add the Public key to Github so that we can create the bridge.

Go to Github -> setting -> SSH and GPG key ->Add new key -> do cat id\_rsa.pub in the EC2instance and copy the key -> give a name to the key -> paste the copied key.

Step 5: Create a item in jenkins . Enter an item name -> Select Freestyle project -> Add the URL of the repo.





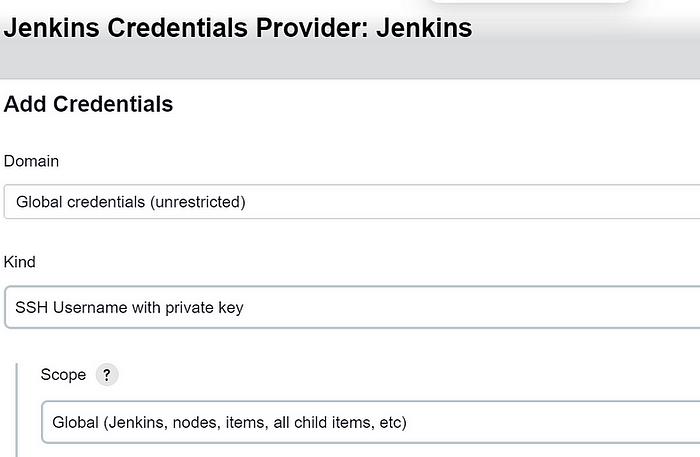




Now we will add the credentials so that jenkins can access the code from the Github.

In Source Core Management

Go to Add credentials



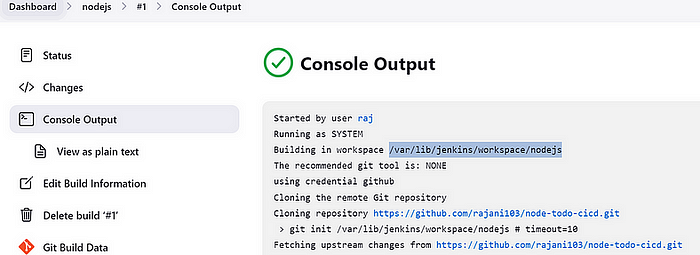




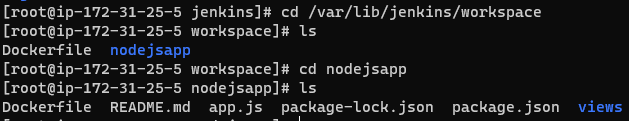
Click on Add -> save it.

Step 6: Now we are all ready to Build the job now. Click on Build now.

It will get build.



Go to instance and check if the repo is cloned there.



Step 7: Now install the necessary dependencies using following commands:

sudo yum install nodejs && sudo yum install npm

sudo apt install nodejs && sudo apt install npm

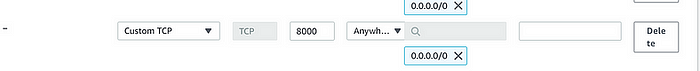
npm install

node app.js

After executing all these commands. check if you can access the URL. It is not accessible as we have not given access to the port 8000

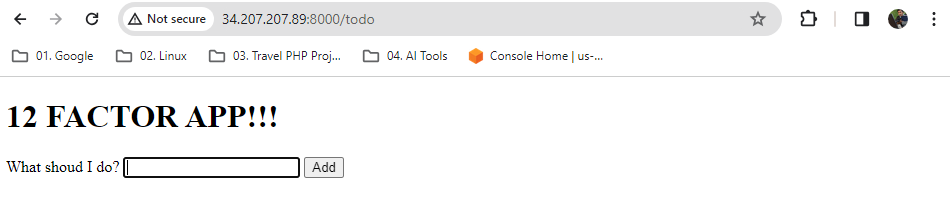


Go to instance -> Security -> Edit inbound rules -> Add rule



Save it.

Now take public ip of the instance and port 8000 and you can access the application.



Step 8: Now we will dockerize the application so that it can be accessed anywhere by anyone.

Go to instance -> install docker.

#sudo install docker.io (#apt-get install docker)(apt install docker.io)

Step 9: Create your Dockerfile.

FROM **aakash3902/nodejsapp**  
WORKDIR app  
COPY . .  
RUN npm install  
EXPOSE 8000  
CMD ["node","app.js"]

Step 10: Now build the image using this Dockerfile.

#docker build -t todoapp .

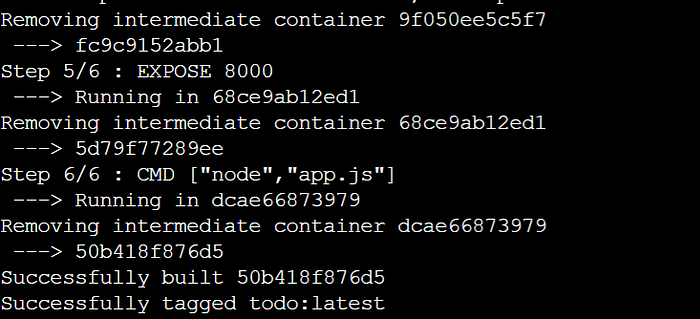
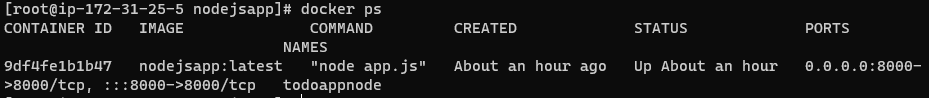


Image (**aakash3902/nodejsapp**) has been created successfully.

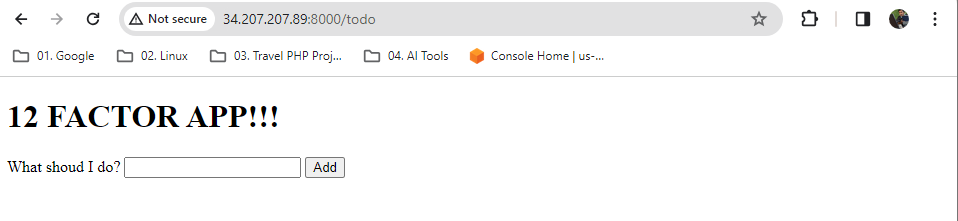
Step 11: Now we will create container from this image.

#docker run -d --name todoappnode -p 8000:8000 todoapp:latest

Container will be created after this.



Try accessing the same using the public IP and port 8000.

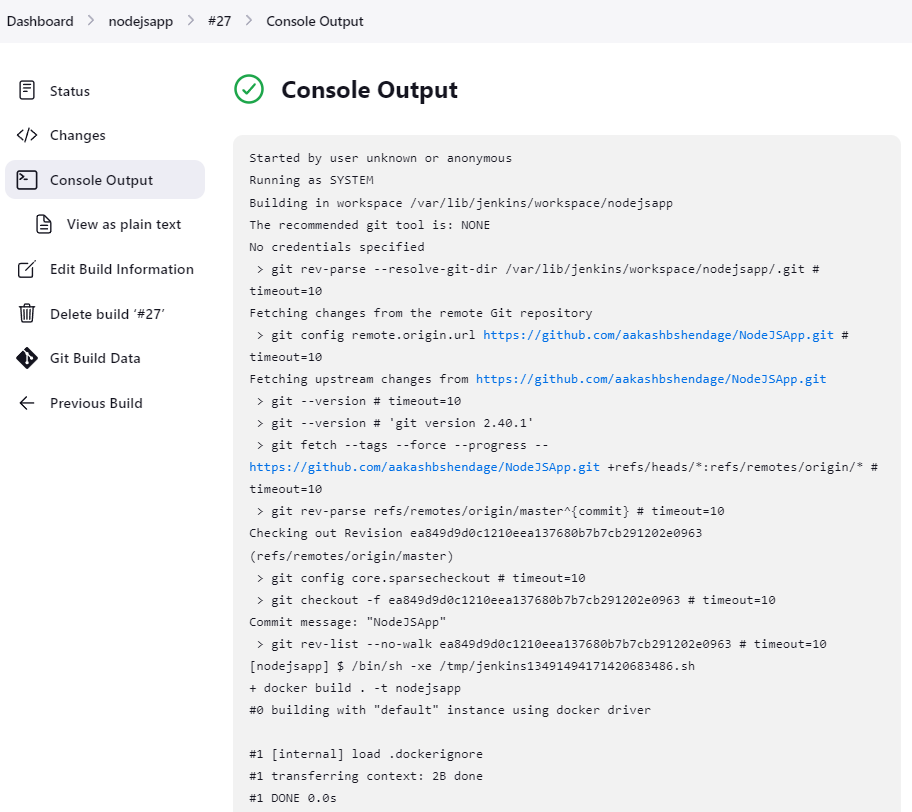


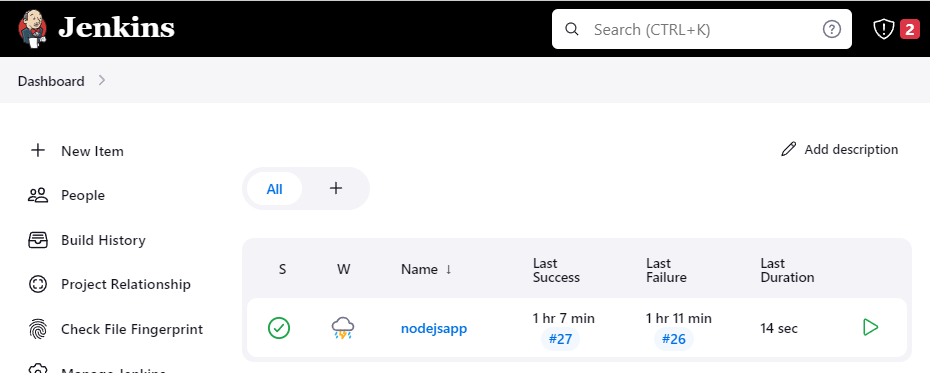
It is accessible.

Step 12: Now we will automate this process by adding the commands in the shell.



Build is successsful. We can access the same on browser.





Now all the processes that we were doing manually will be performed automatically as we did the automation.

Step 13: Now we will configure web-hook so that every time there is any updation, deletion on the repository the job should be automatically triggered and it should perform the upcoming processes.

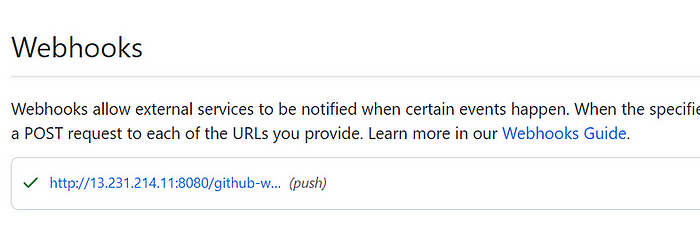
Kill the existing container first.

Jenkins -> Manage Jenkins -> Manage Plugins -> Install github integration plugin

Go to repository settings -> webhook -> Add webhook -> Payload URL add jenkins URL here -> (<http://PublicIPofEC2Instance:8080/github-webhook/>)

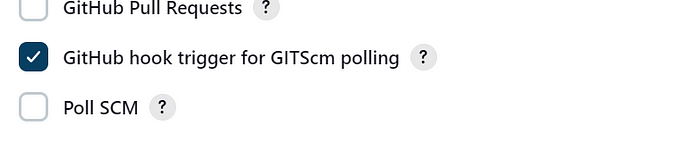
Content type -> application.json

Add webhook.

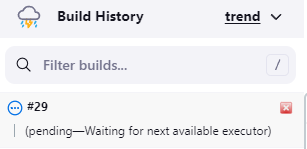


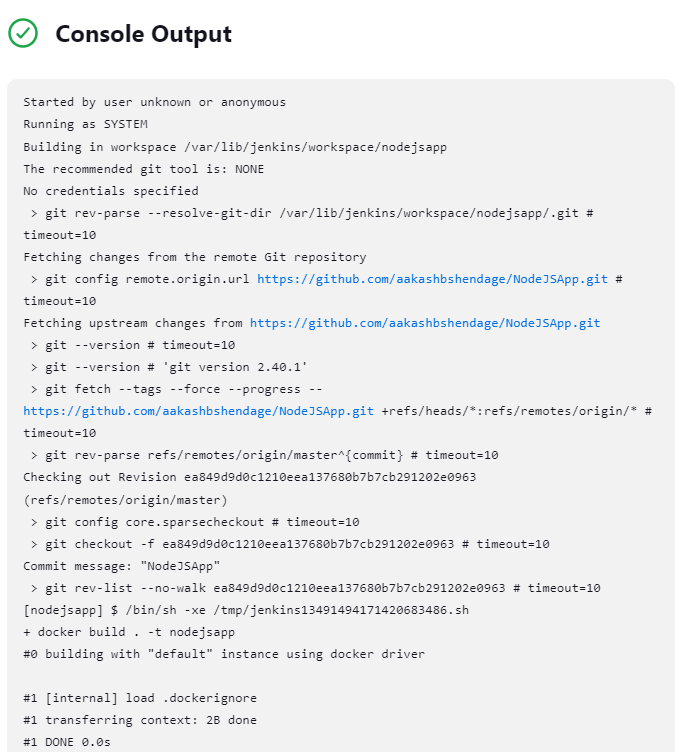
It has been added successfully.

Go to jenkins -> configure -> tick this



Now just update something in the repo and your job will be triggered automatically.





So the build is successful.

Now we are successfully added automation in our devops project.

Thank You.