

CI-CD INTEGRATION

AIM : Jenkins CICD with GitHub Integration. CICD -Continuous Integration and Continuous Deployment. We are going to do CICD pipeline in this assessment.

We are going to implement a pipeline where the developer code run effectively.

Tools used :

1. AWS
2. JENKINS
3. Github
4. Docker

Resources implemented :

5. EC2
- 6.

Process /Implementation:

>First we take the code from the developer from the GitHub, where it is usually stored in the repository .The DevOps Engineer takes the code from it and performs the required actions.

> The DevOps Engineer role is to check that the code is continuously Integrated in the required environment or not.

> The DevOps Engineer needs to make sure that the code integrated in every type of environment(MacOS, windows, linux).

>The DevOps Engineer uses a tool called as Docker to make a virtualised container so that the code runs in any environment.

>We use Amazon EC2 instance to deliver the containerized product.

>We use the pipeline, for making a flow for taking the code, containerising it and then delivering it . Jenkins does this work .

1. Create an Instance

The image displays three sequential screenshots of the AWS Management Console interface, illustrating the steps to create an EC2 instance.

Screenshot 1: Launch an instance

The console shows the "Launch an instance" page. The "Name and tags" section has a name field set to "jenkins-master". The "Application and OS Images (Amazon Machine Image)" section is expanded, showing a search bar and a "Quick Start" section with various AMIs. The "Summary" panel on the right shows the configuration: Number of instances: 1, Software Image (AMI): Canonical, Ubuntu, 22.04 LTS, Virtual server type (instance type): t2.micro, Firewall (security group): New security group, and Storage (volumes): 1 volume(s) - 8 GiB. The "Launch instance" button is visible.

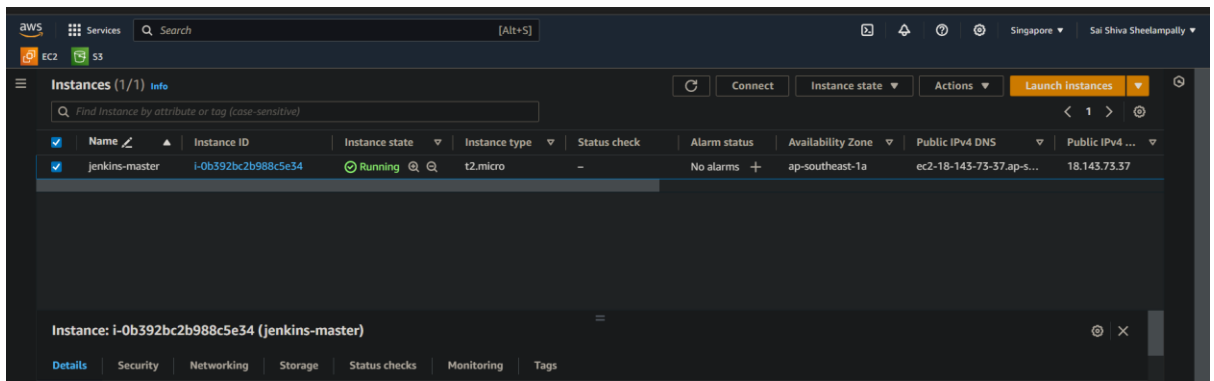
Screenshot 2: Application and OS Images (Amazon Machine Image)

The console shows the "Application and OS Images (Amazon Machine Image)" section. The "Quick Start" section is expanded, showing a search bar and a "Browse more AMIs" button. The "Amazon Machine Image (AMI)" section is expanded, showing a list of AMIs. The "Summary" panel on the right shows the configuration: Number of instances: 1, Software Image (AMI): Canonical, Ubuntu, 22.04 LTS, Virtual server type (instance type): t2.micro, Firewall (security group): New security group, and Storage (volumes): 1 volume(s) - 8 GiB. The "Launch instance" button is visible.

Screenshot 3: Network settings

The console shows the "Network settings" section. The "Network" section is expanded, showing a search bar and a "Quick Start" section. The "Subnet" section is expanded, showing a search bar and a "Browse more subnets" button. The "Auto-assign public IP" section is expanded, showing a search bar and a "Browse more public IP addresses" button. The "Firewall (security groups)" section is expanded, showing a search bar and a "Browse more security groups" button. The "Summary" panel on the right shows the configuration: Number of instances: 1, Software Image (AMI): Canonical, Ubuntu, 22.04 LTS, Virtual server type (instance type): t2.micro, Firewall (security group): New security group, and Storage (volumes): 1 volume(s) - 8 GiB. The "Launch instance" button is visible.

Changing the network settings so that anyone from internet can access this



2. Now connect the instance and install Jenkins on aws

For that run some commands on the server. They are

- `sudo apt update`
- `sudo apt install openjdk-11-jre`
- `sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \`
<https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key>
- `echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \`
`https://pkg.jenkins.io/debian-stable binary/ | sudo tee \`
`/etc/apt/sources.list.d/jenkins.list > /dev/null`
- `sudo apt-get update`
- `sudo apt-get install jenkins`

```
us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=us-east-1&connType=standard&instanceId=i-047b7...
AWS Services Search [Alt+S]
EC2 S3
Instances (1/1) Info
Find Instance by attribute or tag (case-sensitive)
Name Instance ID Instance state Instance type Status check Alarm status Availability Zone Public IPv4 DNS Public IPv4 ...
jenkins-master i-0b392bc2b988c5e34 Running t2.micro - No alarms + ap-southeast-1a ec2-18-143-73-37.ap-s... 18.143.73.37
Instance: i-0b392bc2b988c5e34 (jenkins-master)
Details Security Networking Storage Status checks Monitoring Tags

Building dependency tree... Done
Reading state information... Done
openjdk-11-jre is already the newest version (11.0.21+9-0ubuntu1-22.04).
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
ubuntu@ip-172-31-17-252:~$ sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
--2024-01-15 17:06:22-- https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
Resolving pkg.jenkins.io (pkg.jenkins.io)... 146.75.34.133, 2a04:4e42:79::645
Connecting to pkg.jenkins.io (pkg.jenkins.io)[146.75.34.133]:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3175 (3.1k) [application/pgp-keys]
Saving to: '/usr/share/keyrings/jenkins-keyring.asc'

/usr/share/keyrings/jenkins-keyring.asc 100%[=====] 3.10K --.-KB/s in 0s

2024-01-15 17:06:22 (48.9 MB/s) - '/usr/share/keyrings/jenkins-keyring.asc' saved [3175/3175]

ubuntu@ip-172-31-17-252:~$ echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
/etc/apt/sources.list.d/jenkins.list > /dev/null
ubuntu@ip-172-31-17-252:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Ign:4 https://pkg.jenkins.io/debian-stable binary/ InRelease
Get:5 https://pkg.jenkins.io/debian-stable binary/ Release [2044 B]
Get:6 https://pkg.jenkins.io/debian-stable binary/ Release.gpg [833 B]
Hit:7 http://security.ubuntu.com/ubuntu jammy-security InRelease
Get:8 https://pkg.jenkins.io/debian-stable binary/ Packages [26.0 kB]
Fetched 26.9 kB in 1s (36.3 kB/s)
Reading package lists... Done
ubuntu@ip-172-31-17-252:~$
```

Jenkins successfully installed

```
(Reading database ... 66425 files and directories currently installed.)
Preparing to unpack .../net-tools_1.60+git20181103.0eebece-1ubuntu5_amd64.deb ...
Unpacking net-tools (1.60+git20181103.0eebece-1ubuntu5) ...
Selecting previously unselected package jenkins.
Preparing to unpack .../jenkins_2.426.2_all.deb ...
Unpacking jenkins (2.426.2) ...
Setting up net-tools (1.60+git20181103.0eebece-1ubuntu5) ...
Setting up jenkins (2.426.2) ...
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /lib/systemd/system/jenkins.service.
Processing triggers for man-db (2.10.2-1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-94-111:~$
```

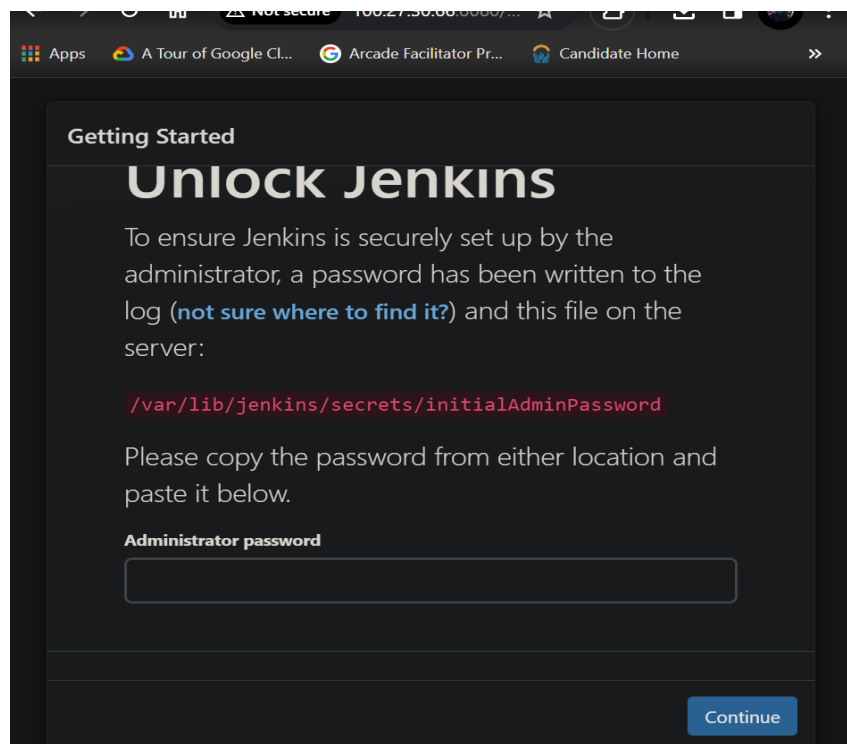
i-039353b36bc1593a4 (jenkins-master) ×

PublicIPs: 100.27.30.66 PrivateIPs: 172.31.94.111

Start Jenkins by running the following commands;

- `sudo systemctl enable jenkins`
- `sudo systemctl start jenkins`
- `sudo systemctl status Jenkins`

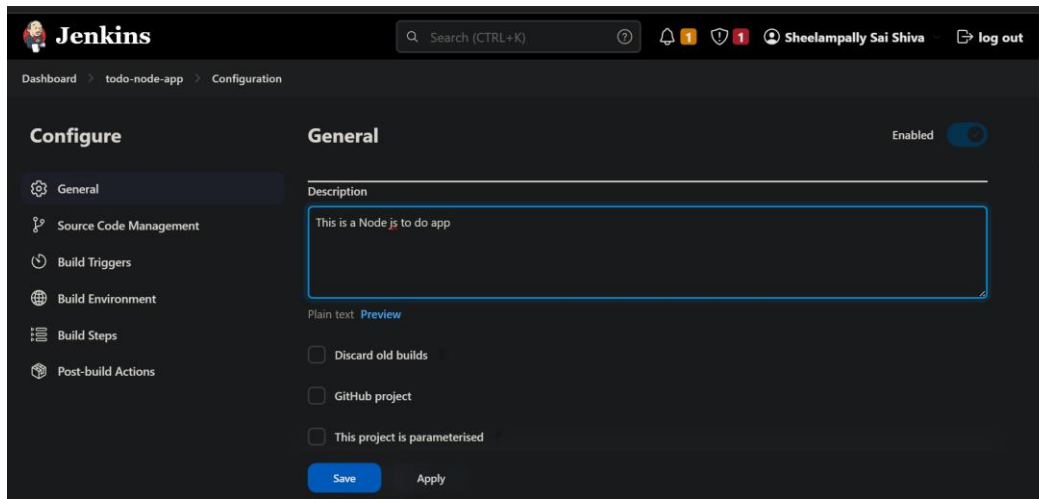
Open the Jenkins application in your web browser by entering the public IP address of your instance, followed by port 8080.



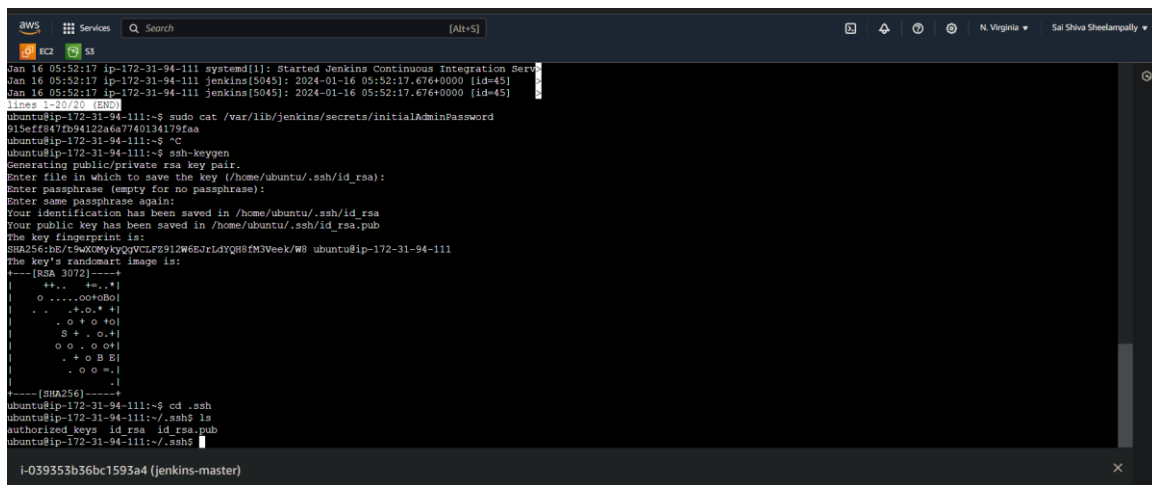
```
ubuntu@ip-172-31-94-111:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
915eff847fb94122a6a7740134179faa
ubuntu@ip-172-31-94-111:~$
```

i-039353b36bc1593a4 (jenkins-master)
PublicIPs: 100.27.30.66 PrivateIPs: 172.31.94.111

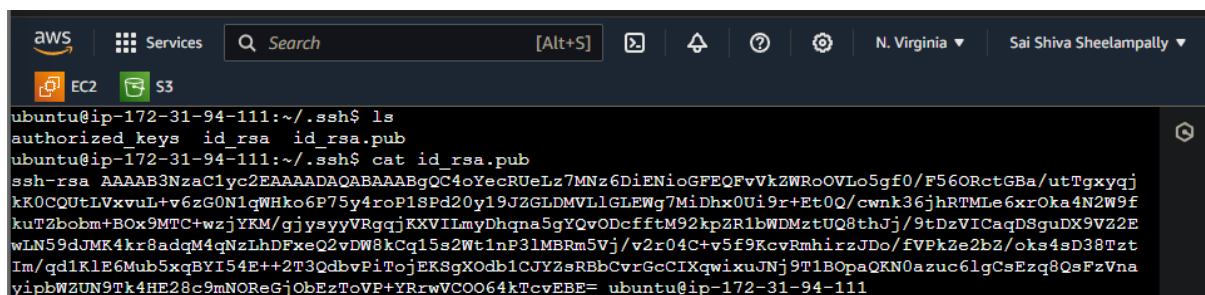
Paste the password then create a new account and then do the freestyle app of Node js
Todo app



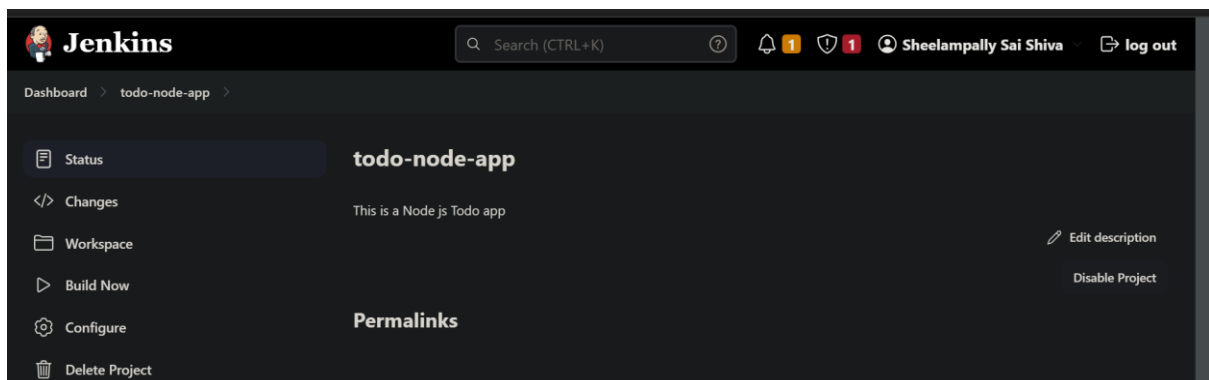
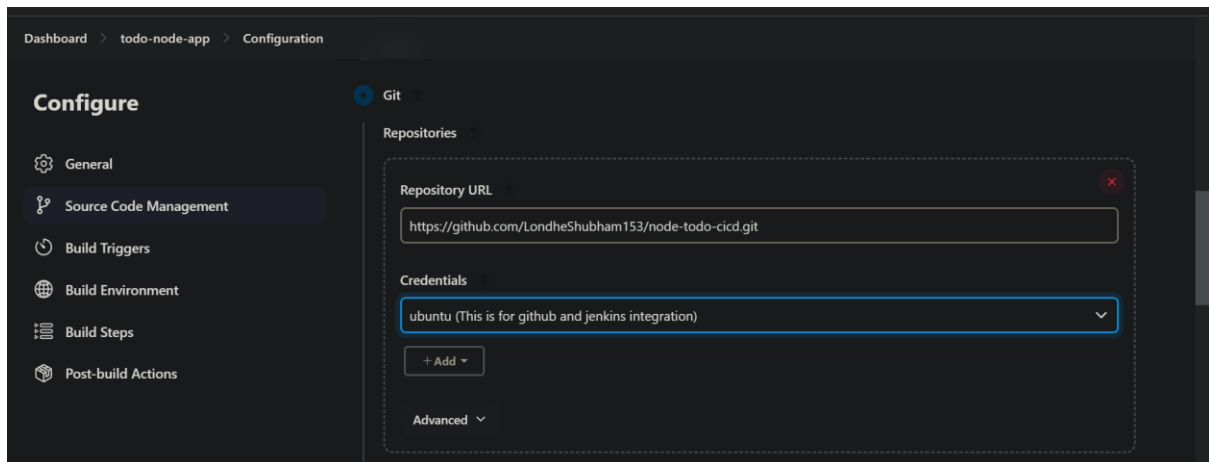
Key-generation



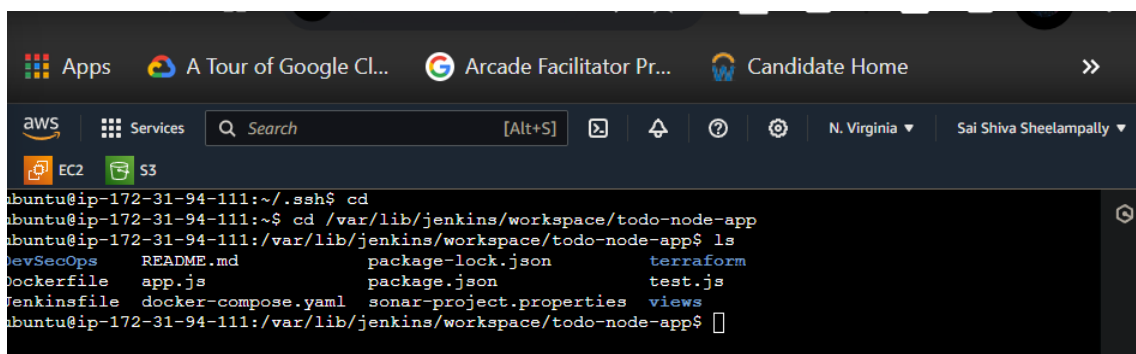
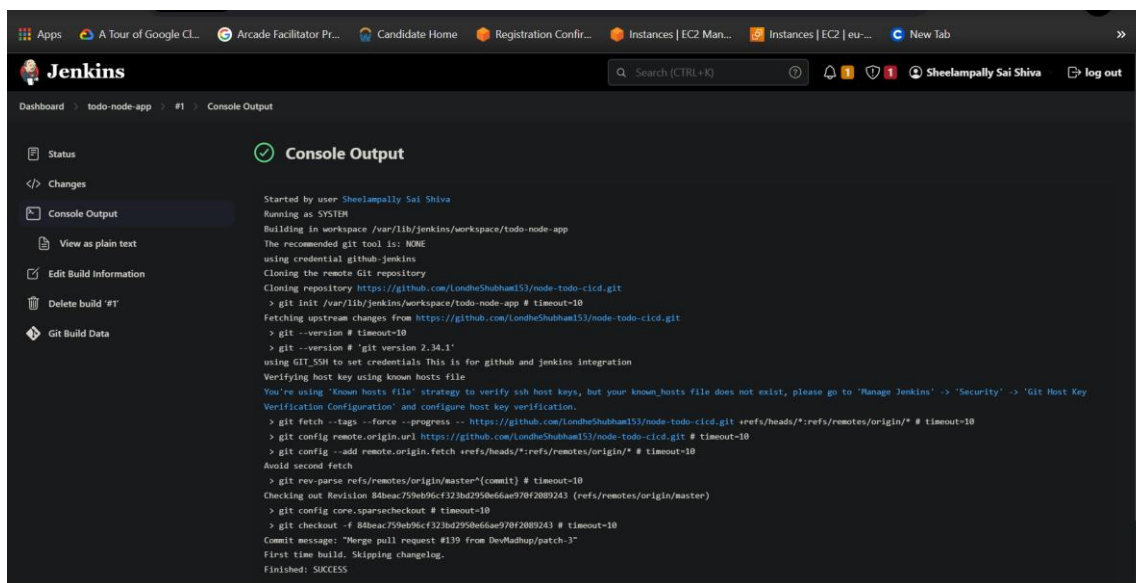
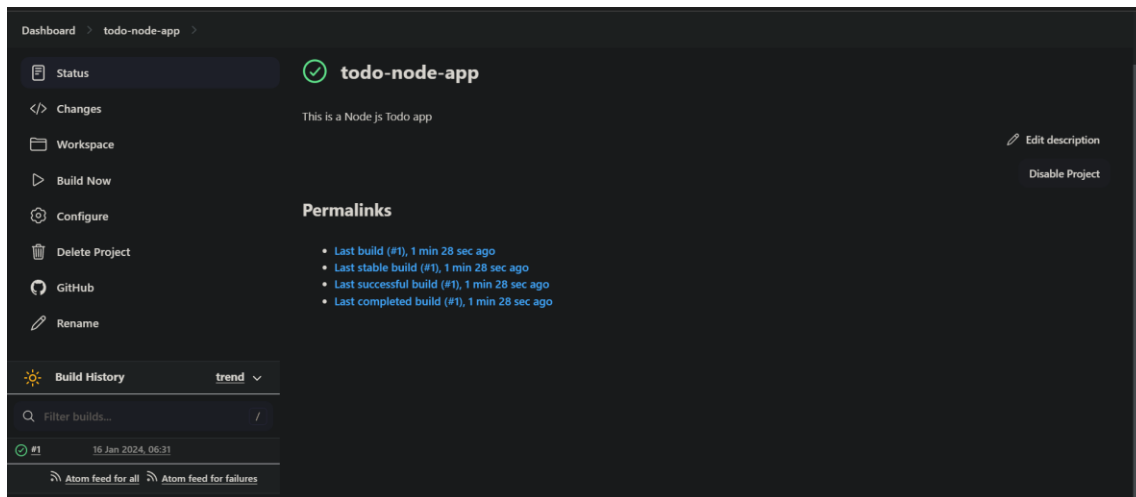
Public key



Private key



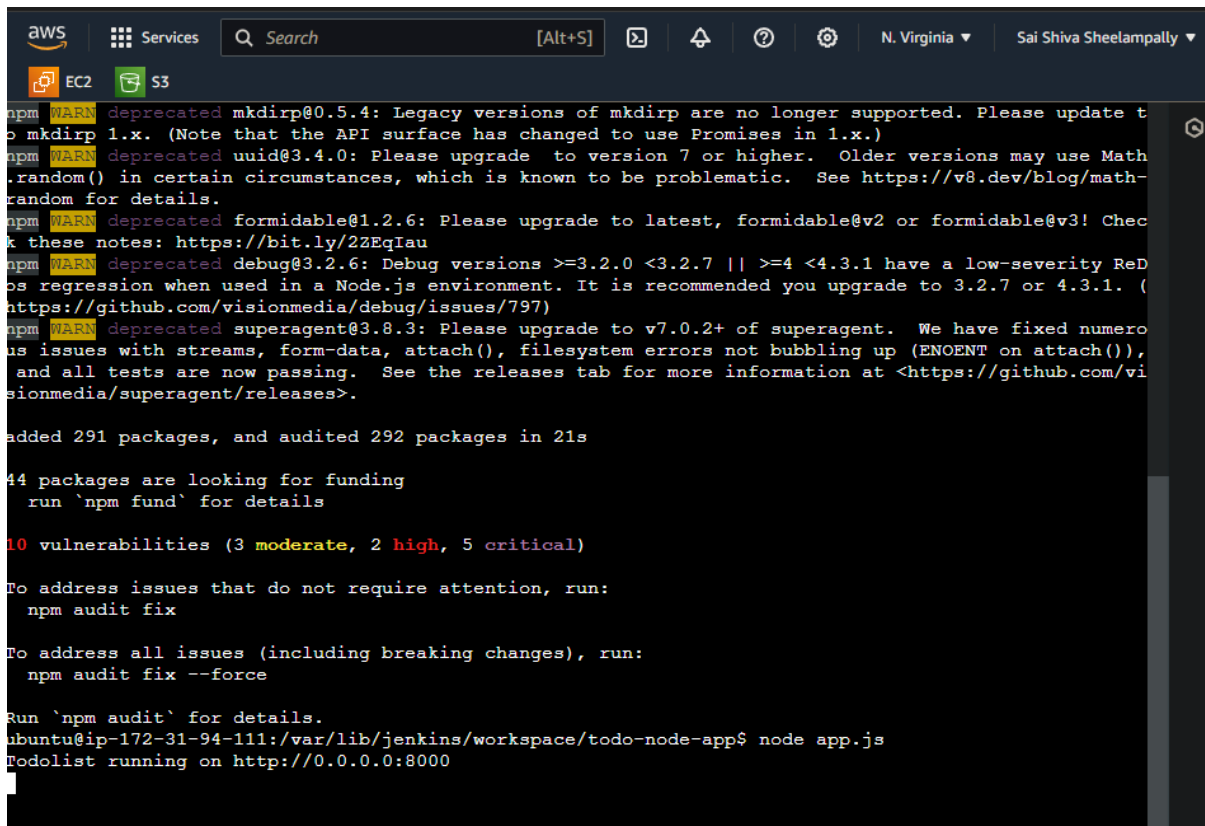
Let us now enter the build now for the piple*****



Now we can confirm that the Jenkins and github are integrated

Now for running the code run the following commands in the EC2 instance connect terminal

- `sudo apt install nodejs`
- `sudo apt install npm`
- `sudo npm install`
- To run the app run this command :: `node app.js`



```
aws
Services
Search [Alt+S]
N. Virginia
Sai Shiva Sheelampally

EC2 S3

npm WARN deprecated mkdirp@0.5.4: Legacy versions of mkdirp are no longer supported. Please update to
mkdirp 1.x. (Note that the API surface has changed to use Promises in 1.x.)
npm WARN deprecated uuid@3.4.0: Please upgrade to version 7 or higher. Older versions may use Math
.random() in certain circumstances, which is known to be problematic. See https://v8.dev/blog/math-
random for details.
npm WARN deprecated formidable@1.2.6: Please upgrade to latest, formidable@v2 or formidable@v3! Chec
k these notes: https://bit.ly/2ZEgIau
npm WARN deprecated debug@3.2.6: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity ReD
os regression when used in a Node.js environment. It is recommended you upgrade to 3.2.7 or 4.3.1. (
https://github.com/visionmedia/debug/issues/797)
npm WARN deprecated superagent@3.8.3: Please upgrade to v7.0.2+ of superagent. We have fixed numero
us issues with streams, form-data, attach(), filesystem errors not bubbling up (ENOENT on attach()),
and all tests are now passing. See the releases tab for more information at <https://github.com/vi
sionmedia/superagent/releases>.

added 291 packages, and audited 292 packages in 21s

44 packages are looking for funding
  run `npm fund` for details

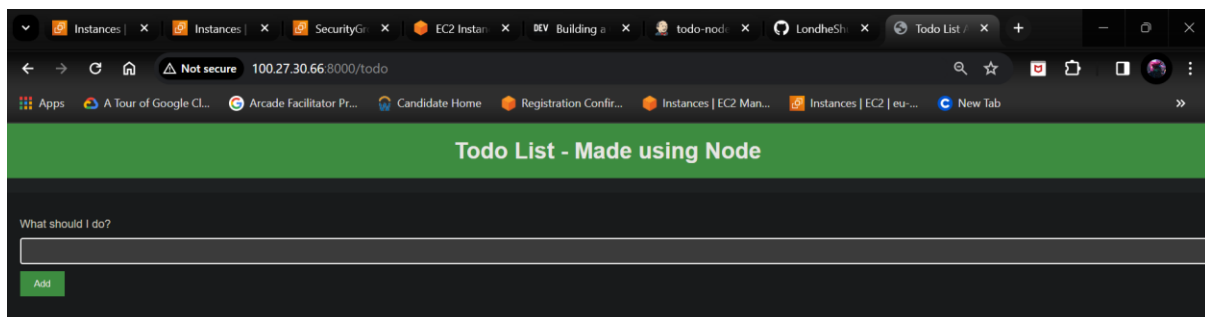
10 vulnerabilities (3 moderate, 2 high, 5 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
ubuntu@ip-172-31-94-111:/var/lib/jenkins/workspace/todo-node-app$ node app.js
Todolist running on http://0.0.0.0:8000
```

Change inbound rules and paste the public IP along with the port 8000



Now we need to make docker to make the app as the virtualised container

For that we need to install Docker. So now by using this command we can install docker

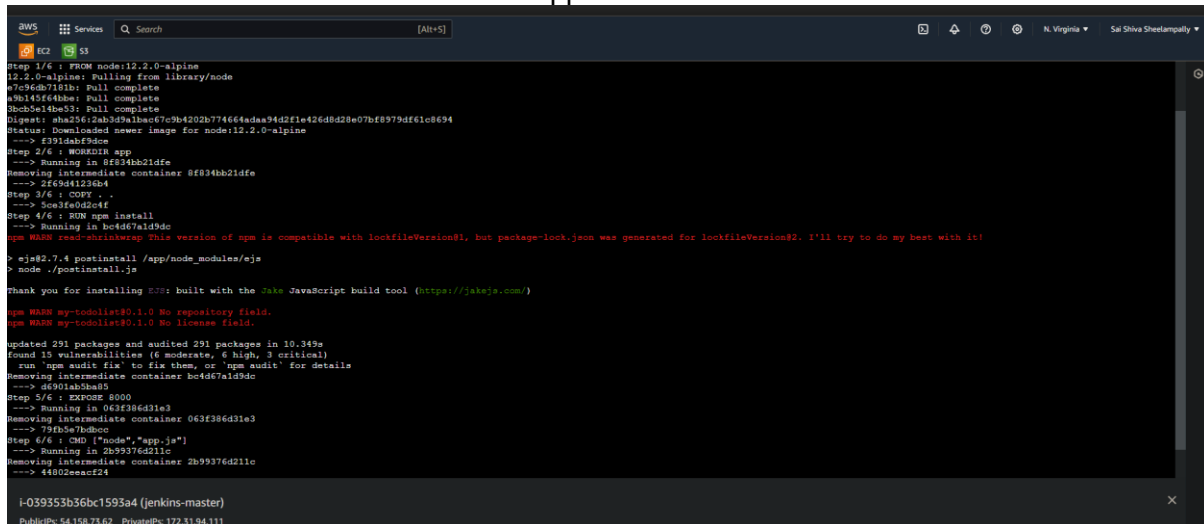
- `sudo apt install docker.io`
- for making docker file we use this command :: `vi Dockerfile`

Now Enter these commands in the file

- `FROM node:12.2.0-alpine`
- `WORKDIR app`
- `COPY . .`
- `RUN npm install`
- `EXPOSE 8000`
- `CMD ["node","app.js"]`

Now for building the docker file use this command

- `docker build . -t node-app`
- `sudo usermod -a -G docker $USER`
- `sudo reboot`
- `docker build . -t node-app`



```
Step 1/6 : FROM node:12.2.0-alpine
12.2.0-alpine: Pulling from library/node
e7c96db7181b: Pull complete
a9b145f64bbe: Pull complete
3bcb3e14be53: Pull complete
Digest: sha256:2ab3d9abac67c9b4202b774664adaa94d2f1e426d8d28e07b8979df61c08694
Status: Downloaded newer image for node:12.2.0-alpine
--> 4391dabf9dce
Step 2/6 : WORKDIR app
--> Running in 8f834bb21dfe
Removing intermediate container 8f834bb21dfe
--> 2f69d41236b4
Step 3/6 : COPY . .
--> 5ca3fe0d2c4f
Step 4/6 : RUN npm install
--> Running in bc4d67ald9dc
npm WARN read-shrinkwrap This version of npm is compatible with lockfileVersion@1, but package-lock.json was generated for lockfileVersion@2. I'll try to do my best with it!
> ejs@2.7.4 postinstall /app/node_modules/ejs
> node ./postinstall.js

Thank you for installing EJS: built with the Jake JavaScript build tool \(https://jakejs.com/\)

npm WARN my-todolist@0.1.0 No repository field.
npm WARN my-todolist@0.1.0 No license field.

updated 251 packages and audited 291 packages in 10.349s
found 15 vulnerabilities (6 moderate, 6 high, 3 critical)
  run `npm audit fix` to fix them, or `npm audit` for details
Removing intermediate container bc4d67ald9dc
--> 4d901ab3ba45
Step 5/6 : EXPOSE 8000
--> Running in 063f386d31e3
Removing intermediate container 063f386d31e3
--> 79fb5e7bd8cc
Step 6/6 : CMD ["node", "app.js"]
--> Running in 2b99376d211c
Removing intermediate container 2b99376d211c
--> 44802eeac2f4

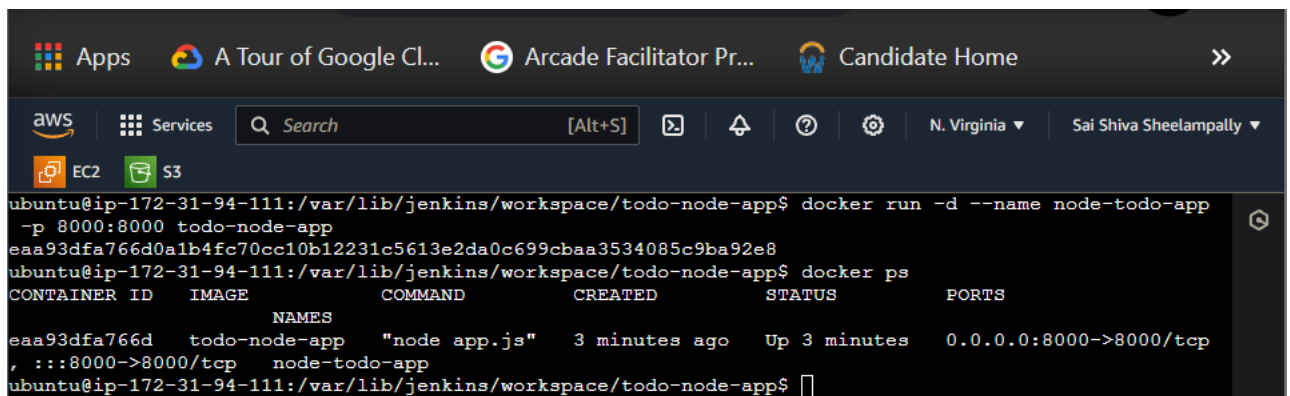
I-039353b36bc1593a4 (jenkins-master)
PublicIP: 54.158.75.62 PrivateIP: 172.31.94.111
```

We have successfully built docker container

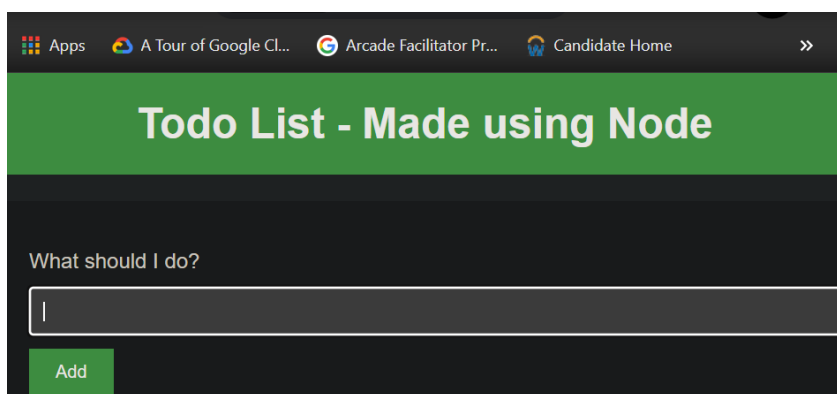
To run the container run the following command

- `docker run -d --name node-todo-app -p 8000:8000 todo-node-app`

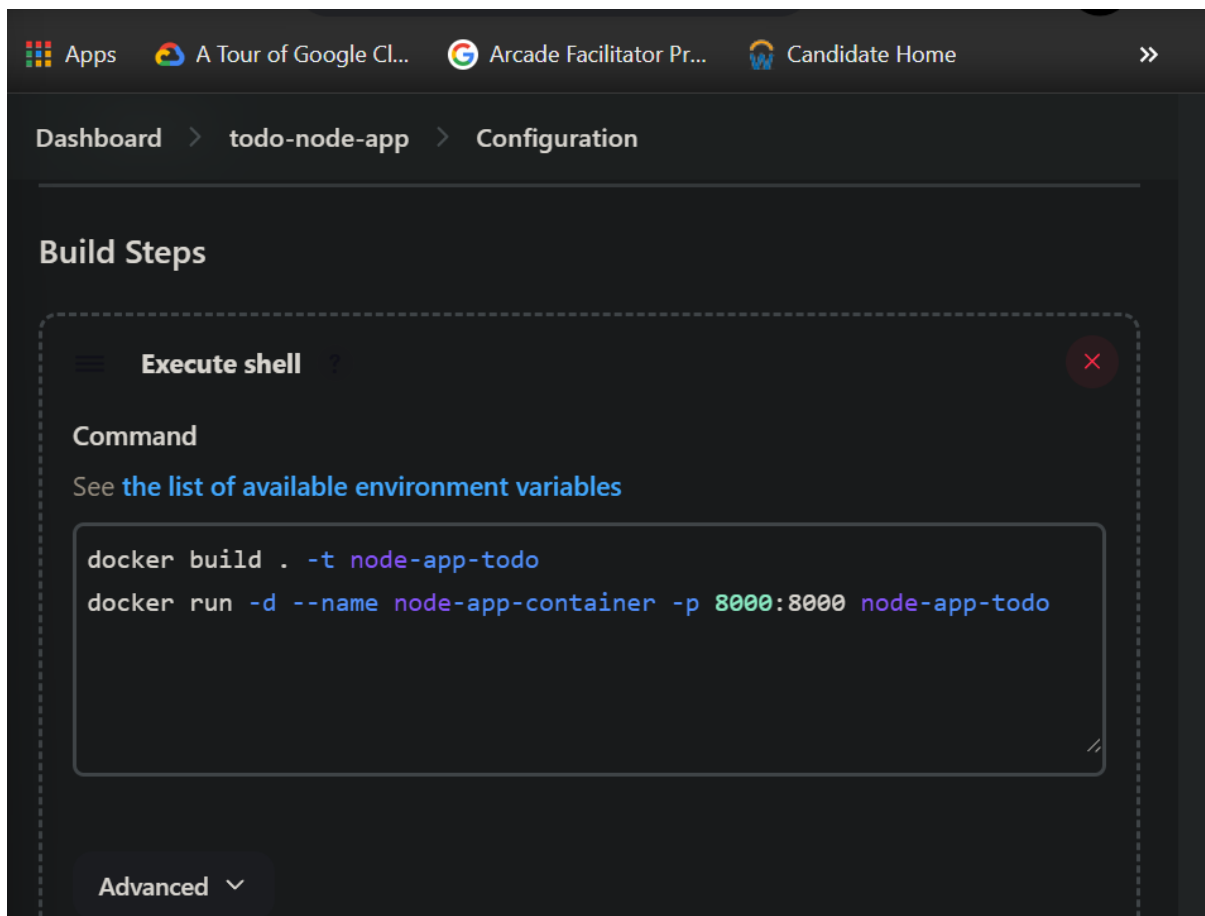
Now the app is running



```
ubuntu@ip-172-31-94-111:/var/lib/jenkins/workspace/todo-node-app$ docker run -d --name node-todo-app
-p 8000:8000 todo-node-app
eaa93dfa766d0a1b4fc70cc10b12231c5613e2da0c699cbaa3534085c9ba92e8
ubuntu@ip-172-31-94-111:/var/lib/jenkins/workspace/todo-node-app$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS
eaa93dfa766d   todo-node-app  "node app.js"           3 minutes ago  Up 3 minutes  0.0.0.0:8000->8000/tcp
, :::8000->8000/tcp  node-todo-app
ubuntu@ip-172-31-94-111:/var/lib/jenkins/workspace/todo-node-app$
```

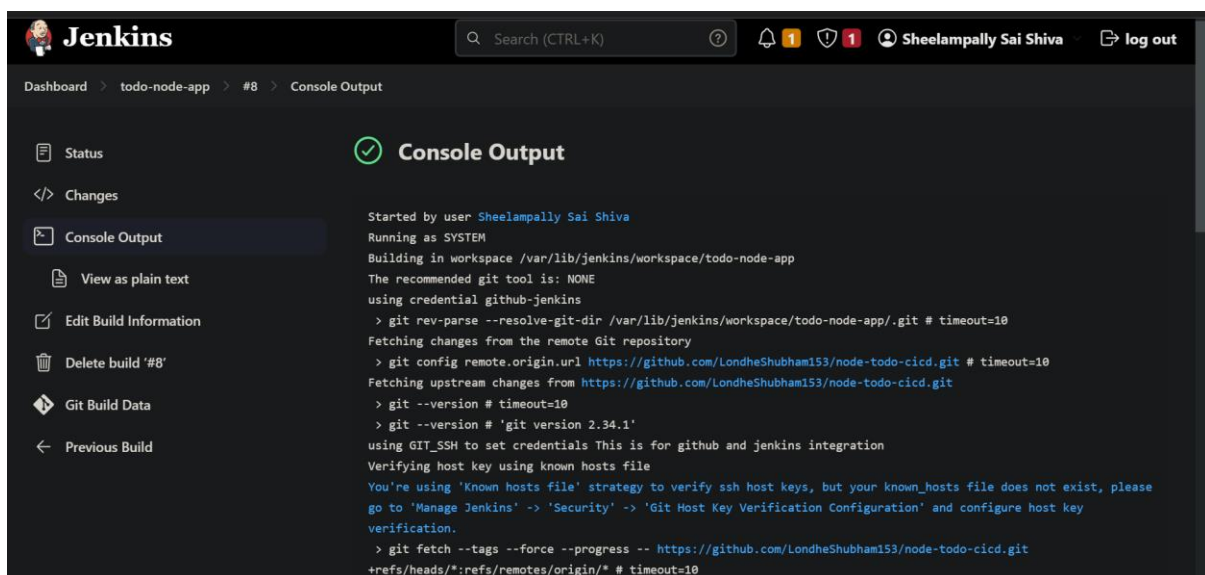


Now go to jenkins job and then to Execute shell and paste the following commands



sudo usermod -a -G docker Jenkins

sudo systemctl restart Jenkins

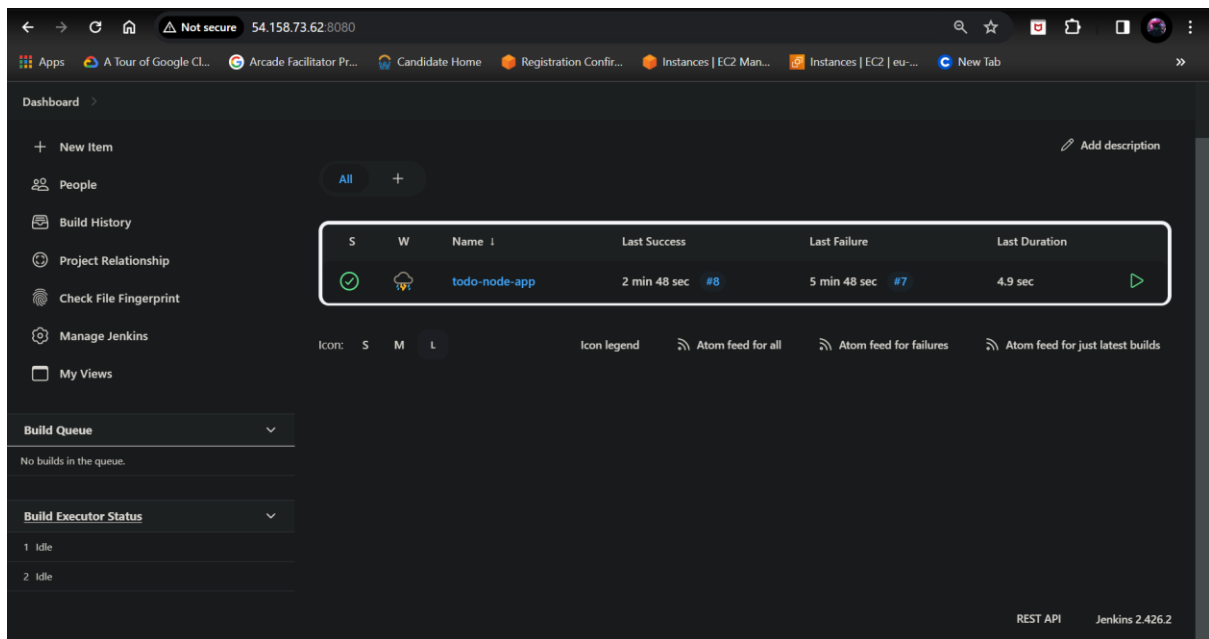


```
Dashboard > todo-node-app > #8 > Console Output

Sending build context to Docker daemon 25.3MB

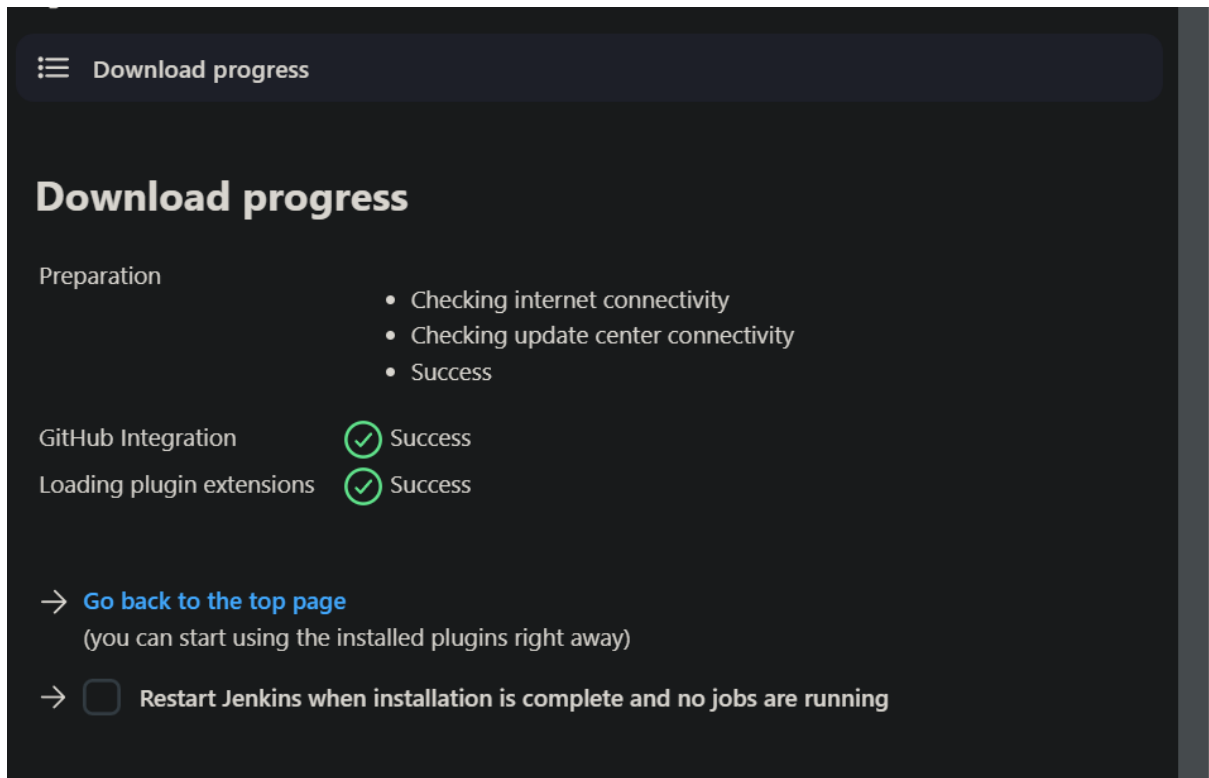
Step 1/7 : FROM node:12.2.0-alpine
----> f391dabf9dce
Step 2/7 : WORKDIR app
----> Using cache
----> 2f69d41236b4
Step 3/7 : COPY . .
----> Using cache
----> ealc09344898
Step 4/7 : RUN npm install
----> Using cache
----> 80ff3c694779
Step 5/7 : RUN npm run test
----> Using cache
----> c671d4887290
Step 6/7 : EXPOSE 8000
----> Using cache
----> 6491823c305c
Step 7/7 : CMD ["node","app.js"]
----> Using cache
----> b258eaf1984
Successfully built b258eaf1984
Successfully tagged node-app-todo:latest
+ docker run -d --name node-app-container2 -p 8000:8000 node-app-todo
c37ad7b8d6a00c658ce20ab1cf4b10878d268046bfb939b16a651d389ff76eb
Finished: SUCCESS
```

Now the app is running through Jenkins

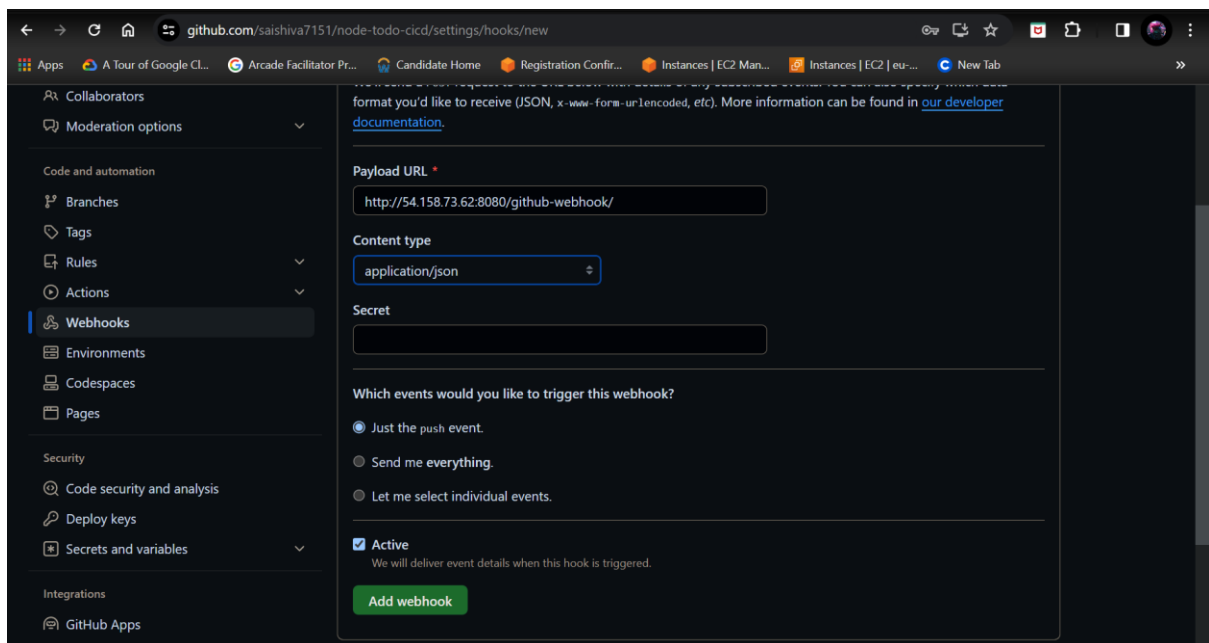


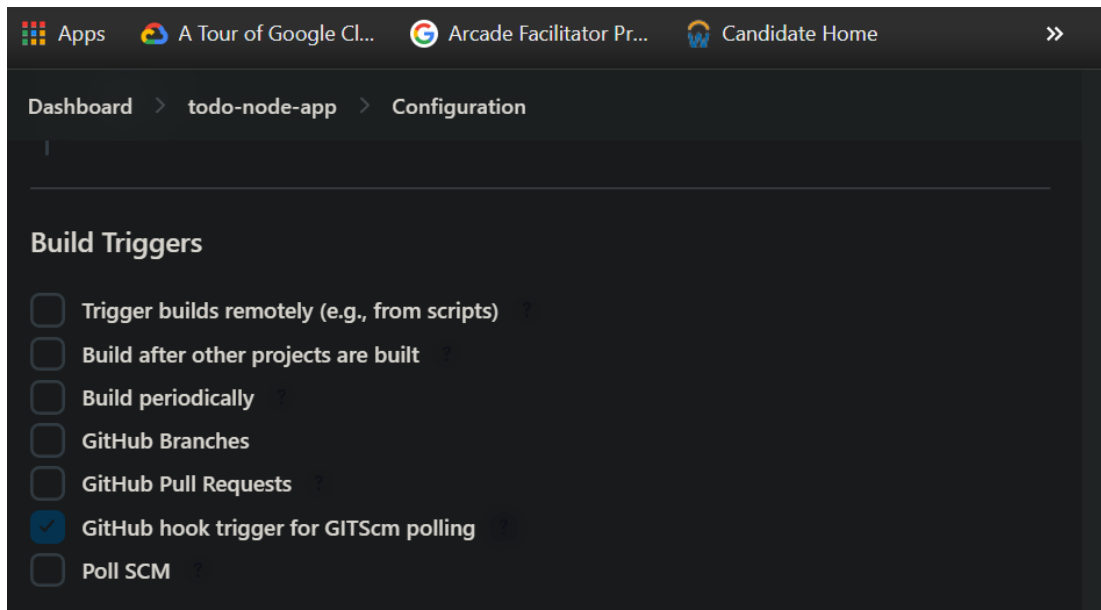
Thus we have implemented CI/CD pipeline and now let us do this by using webhooks :
Automated Jenkins

>>For this first we have install some plugins, check for github integration plugin

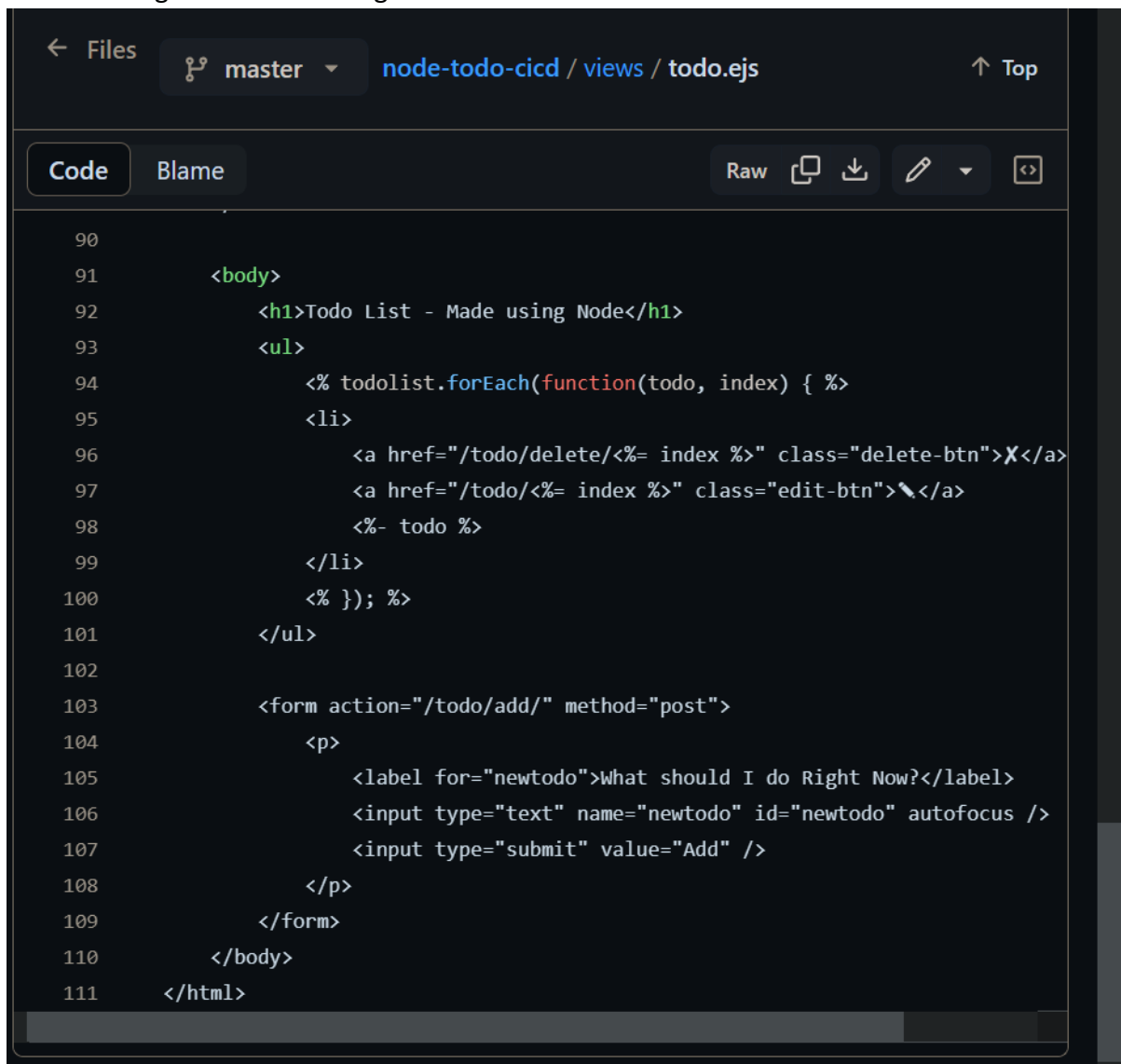


Add webhooks

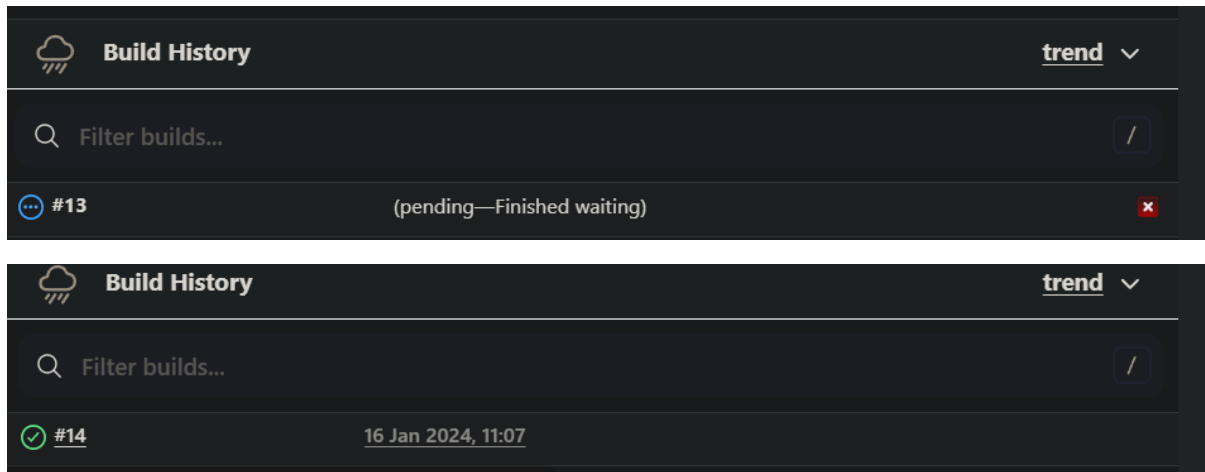




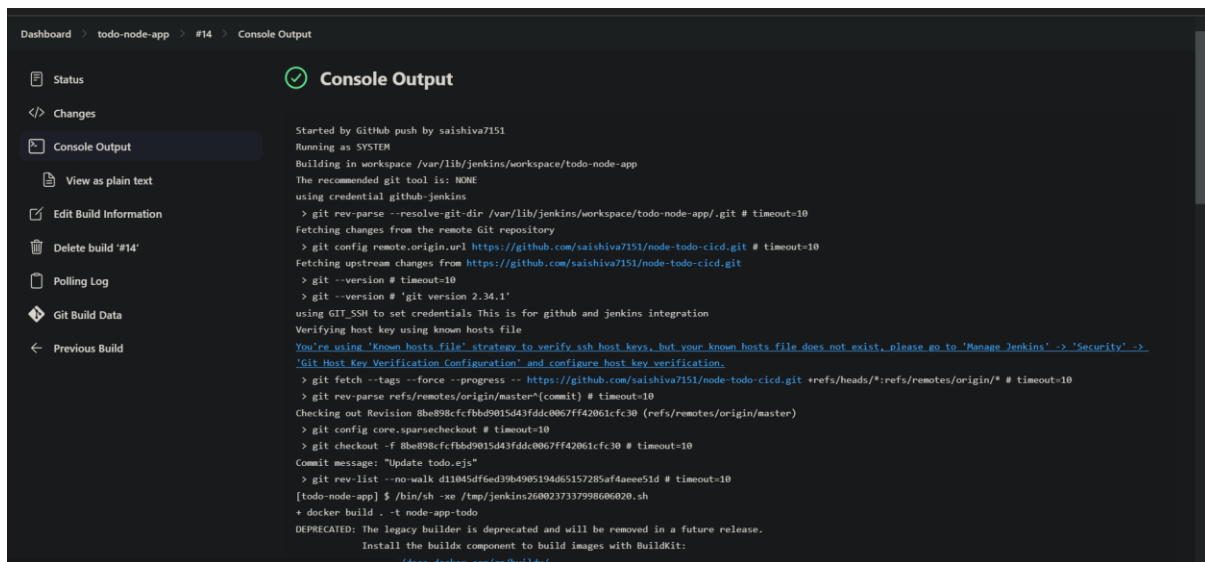
After changing the code into the github account code



New build history generated like this pattern



If success it is shown like this



Sending build context to Docker daemon 25.32MB

Step 1/7 : FROM node:12.2.0-alpine

----> f391dabf9dce

Step 2/7 : WORKDIR /app

----> Using cache

----> 2f69d41236b4

Step 3/7 : COPY . .

----> 2d4cac7d315

Step 4/7 : RUN npm install

----> Running in b5047dd31aa8

npm WARN deprecated shrinkwrap@0.0.4 This version of npm is compatible with lockfileVersion@1, but package-lock.json was generated for lockfileVersion@2. I'll try to do my best with it!

npm

> ej@2.7.4 postinstall /app/node_modules/ejs

> node ./postinstall.js

Thank you for installing B[35eF358] built with the B[32m] JavaScript build tool (B[32m]https://jakejs.com/B[32m])

B[91m] B[91m] B[91m] B[91m] my-todolist@0.1.0 No repository field.

B[91m] B[91m] B[91m] B[91m] my-todolist@0.1.0 No license field.

B[91m] B[91m]

B[91m] B[91m] 291 packages and audited 291 packages in 10.387s

found 15 vulnerabilities (6 moderate, 6 high, 3 critical)

run 'npm audit fix' to fix them, or 'npm audit' for details

Removing intermediate container b5047dd31aa8

----> d81275158fe5

Step 5/7 : RUN npm run test

----> Running in 57bfc24a99e5

> my-todolist@0.1.0 test /app

> mocha --recursive --exit

✓ Is returning 5 when adding 2 + 3

executes before every test

✓ Is returning 6 when multiplying 2 * 3

Test2

executes before every test

✓ Is returning 4 when adding 2 + 3

executes before every test

✓ Is returning 8 when multiplying 2 * 4

This part executes once after all tests

4 passing (12ms)

Removing intermediate container 57bfc24a99e5

----> 4c351919927f

Step 6/7 : EXPOSE 8000

----> Running in 642e8aa7af64

Removing intermediate container 642e8aa7af64

----> fe68f49b6722

Step 7/7 : CMD ["node","app.js"]

----> Running in 0d583d0d25be

Removing intermediate container 0d583d0d25be

----> 48330b6f2df9

Successfully built 48330b6f2df9

Successfully tagged node-app-todo:latest

+ docker run -d --name node-app-container5 -p 8000:8000 node-app-todo

d03be70de103a5314b2425fa2295217ada3443bbd8e21b165f071d2b84c9d3a

Finished: SUCCESS