

Task 2: Jenkins CI/CD Pipeline with Docker

Objective:

To create a CI/CD pipeline using Jenkins to build, test, and deploy a simple web application containerized using Docker, and run it on an AWS EC2 instance.

Tools Used

- AWS EC2 Ubuntu Instance
- Jenkins
- Docker
- GitHub
- Node.js (for the app)
- Jenkins Plugins: Git, Docker, NodeJS, Pipeline, Pipeline Stage View

Definitions

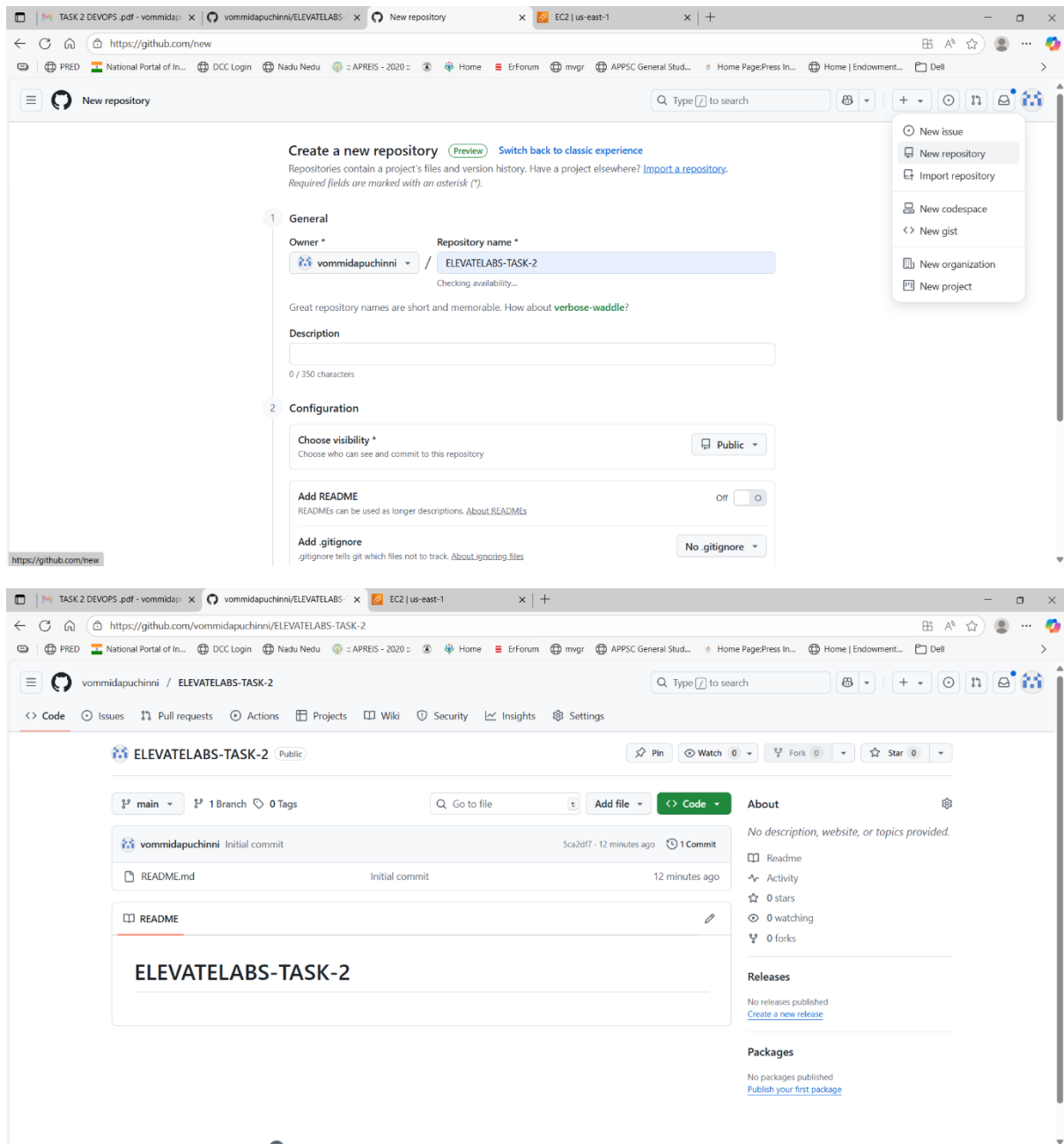
- **CI/CD:** Continuous Integration and Continuous Deployment – a process where code changes are automatically built, tested, and deployed.
- **Jenkins:** An open-source automation server used to implement CI/CD pipelines.
- **Docker:** A tool that allows applications to run in isolated containers.
- **EC2:** AWS virtual server where Jenkins and the application are hosted.
- **GitHub:** A version control platform where the code is stored.
- **Webhook** (optional): It automatically notifies Jenkins of code changes in GitHub to trigger the pipeline.

What I Did

1. Created a GitHub Repository

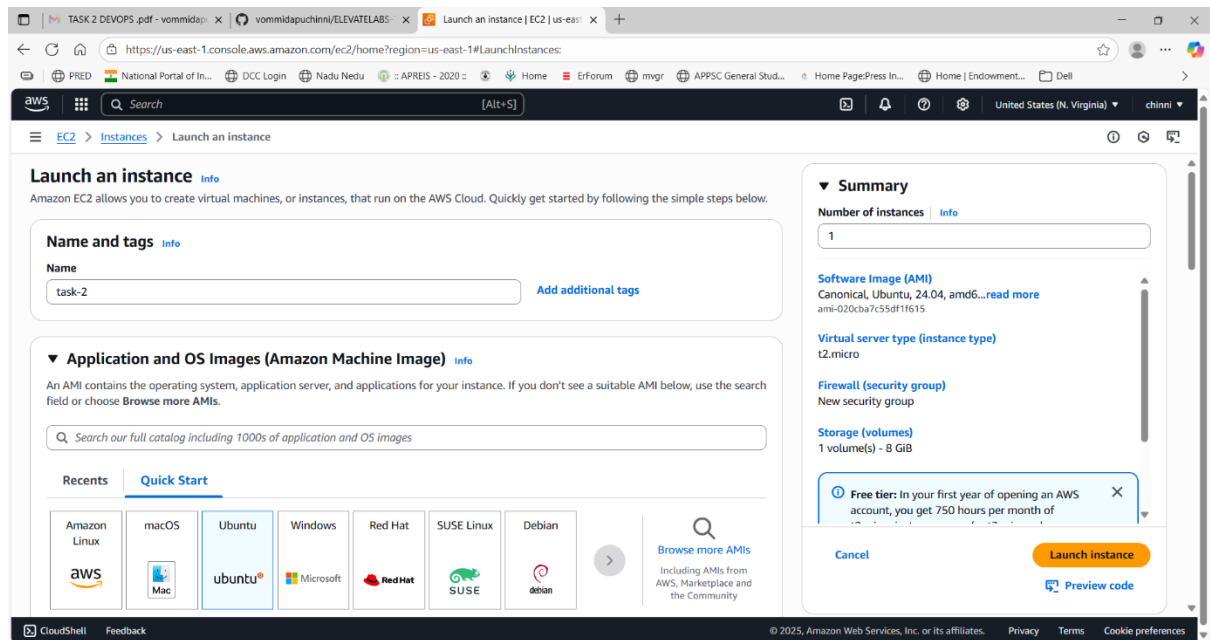
Click on + new repository.

Made a public repository and added a README file to it.

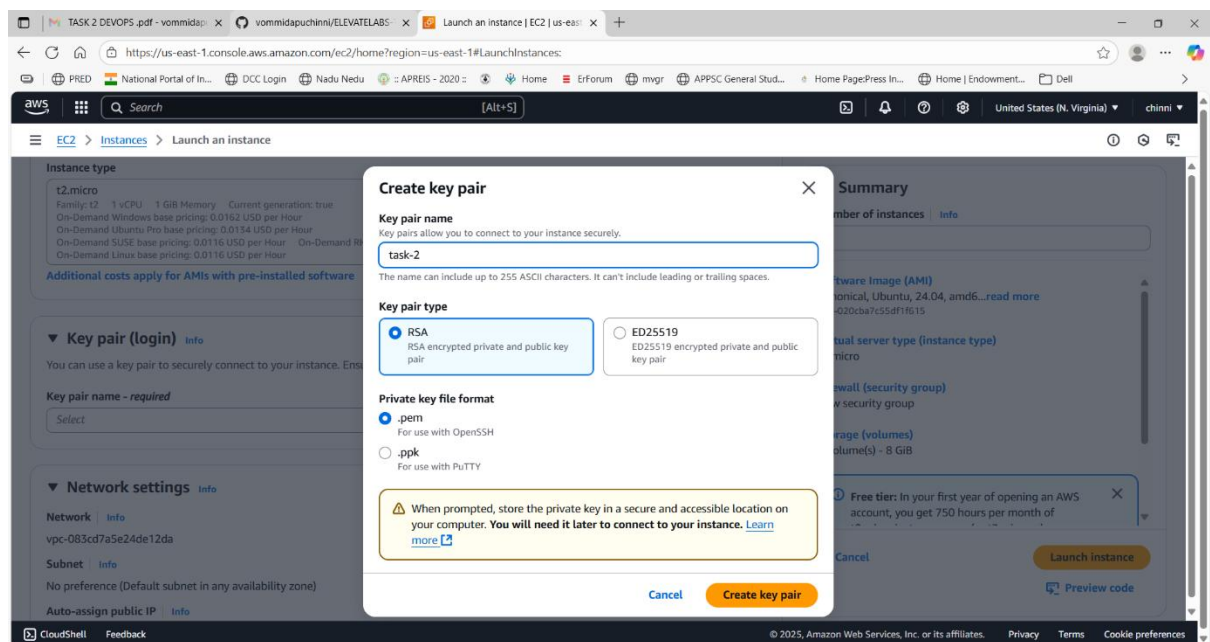


2. Launched EC2 Instance

- Login to AWS console management
- Click on launch instance
- Choose AMI as ubuntu (Linux distribution)
- Click on create new key pair
- Give name .pem is normal key .py is putty key pair.

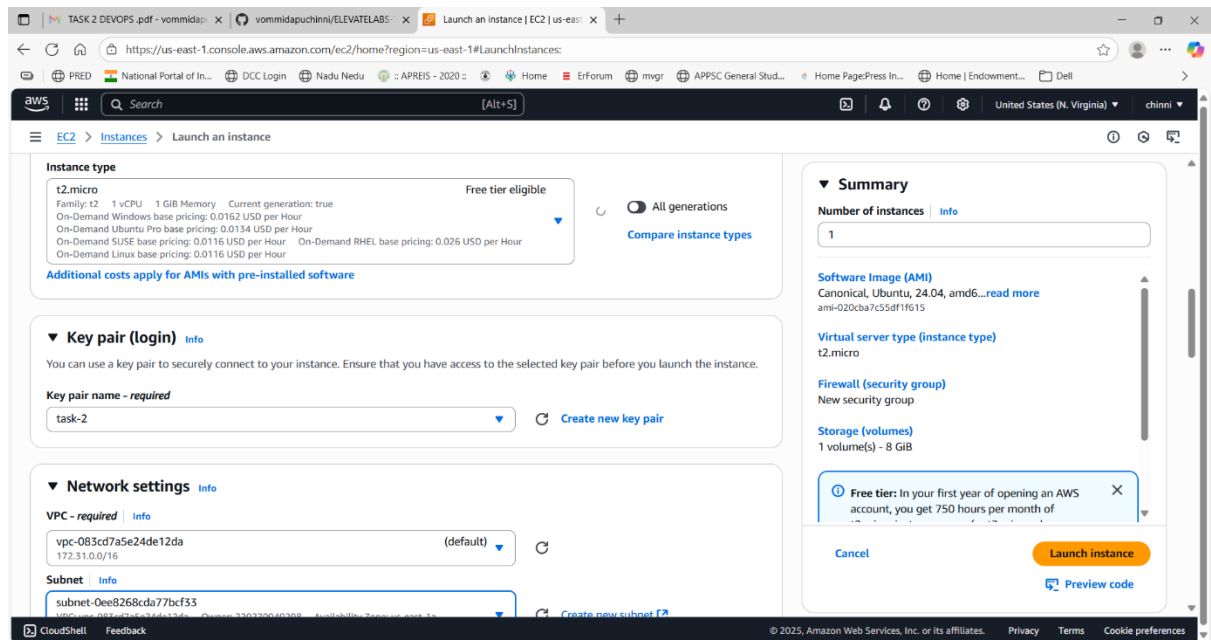


- Ubuntu 22.04 t2.micro instance.

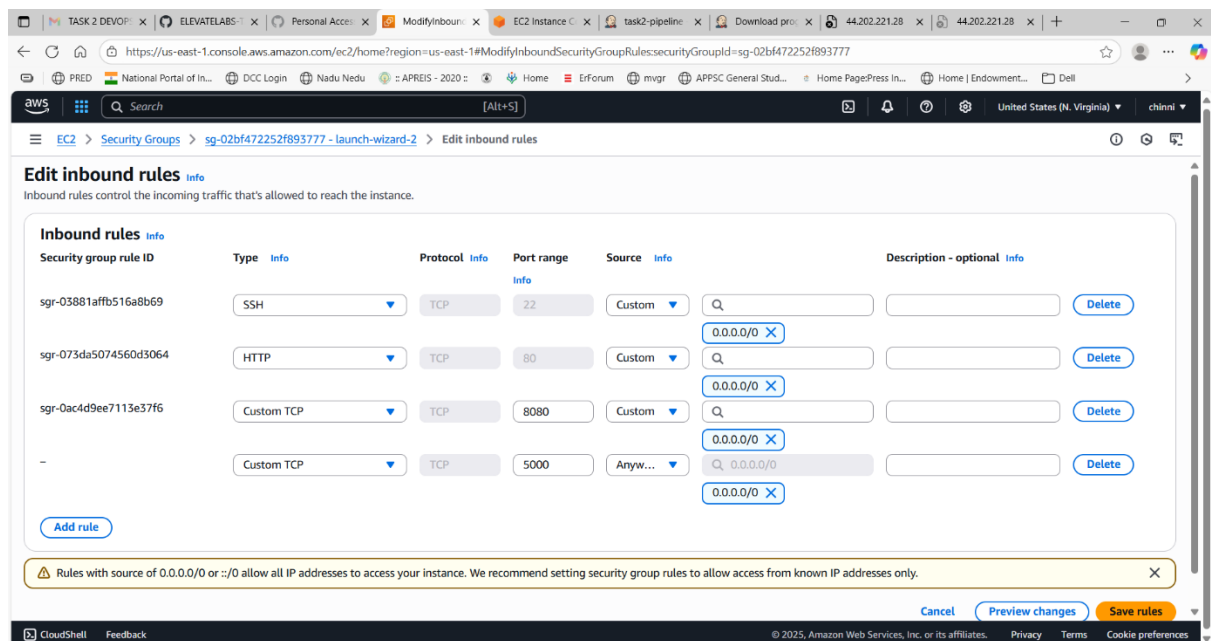


Choose VPC as default one.

Choose any subnet out of subnets.



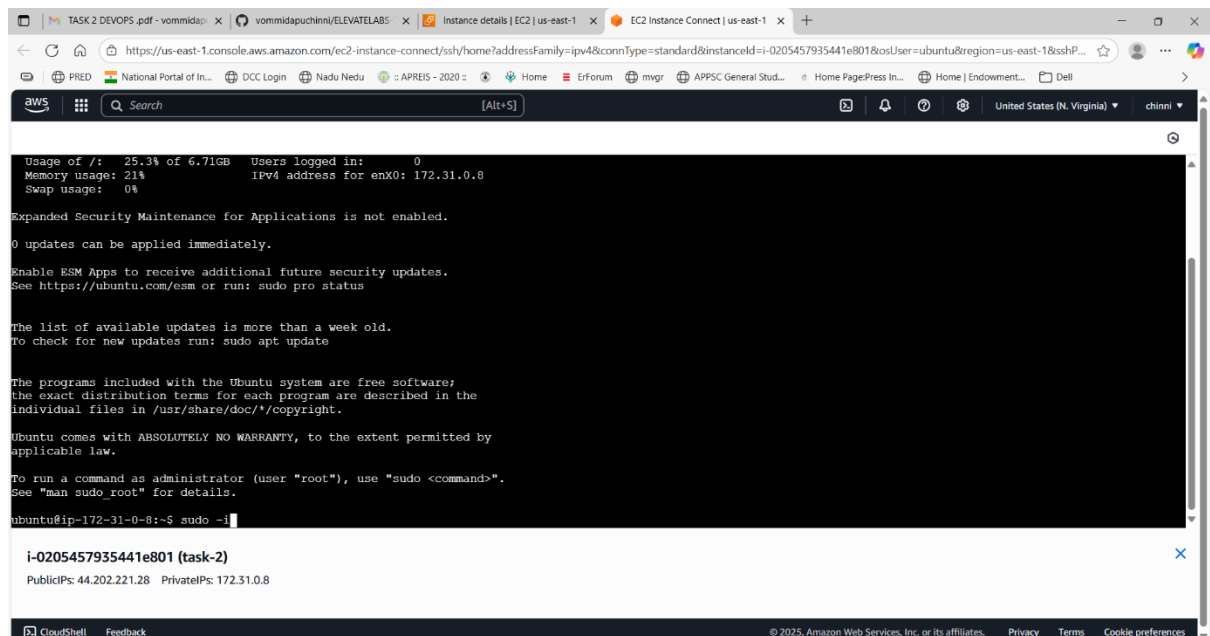
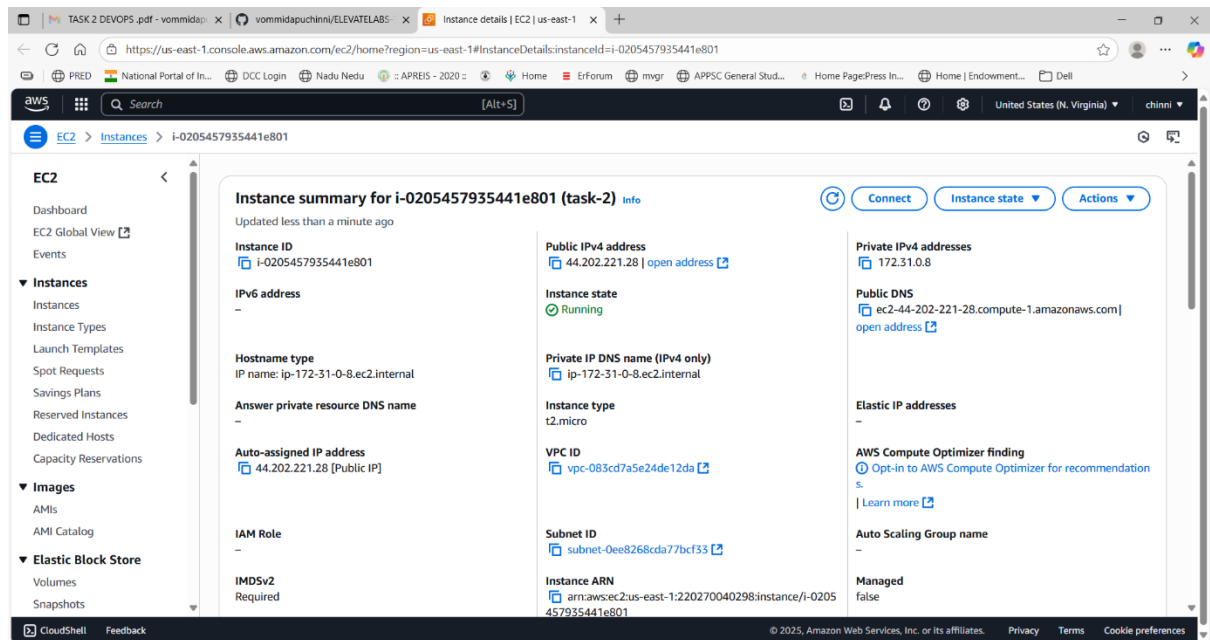
- Allowed port **5000, 8080, 80** in the security group.



After launching an instance, we can see like this.

Click on connect to connect to an instance.

After again click on connect.



sudo -i is used to change normal user to root user.

apt update -y is used to update

```
root@ip-172-31-0-8:~# apt update -y
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [1054 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [1314 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [264 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [164 kB]
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [1120 kB]
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [287 kB]
Get:19 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [377 kB]
Get:20 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [26.0 kB]
Get:21 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [1645 kB]
Get:22 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [359 kB]
Get:23 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 B]
Get:24 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [33.2 kB]
Get:25 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [6772 B]
Get:26 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [1940 B]
```

- Installed required tools:

- Jenkins
- Docker
- Git
- Node.js

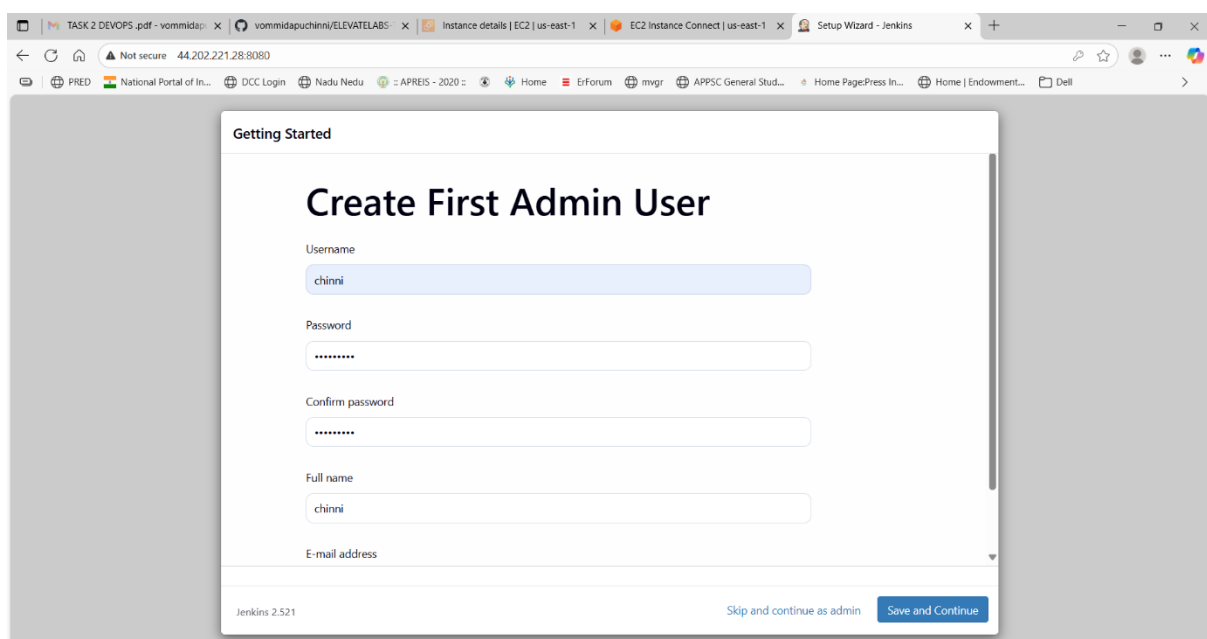
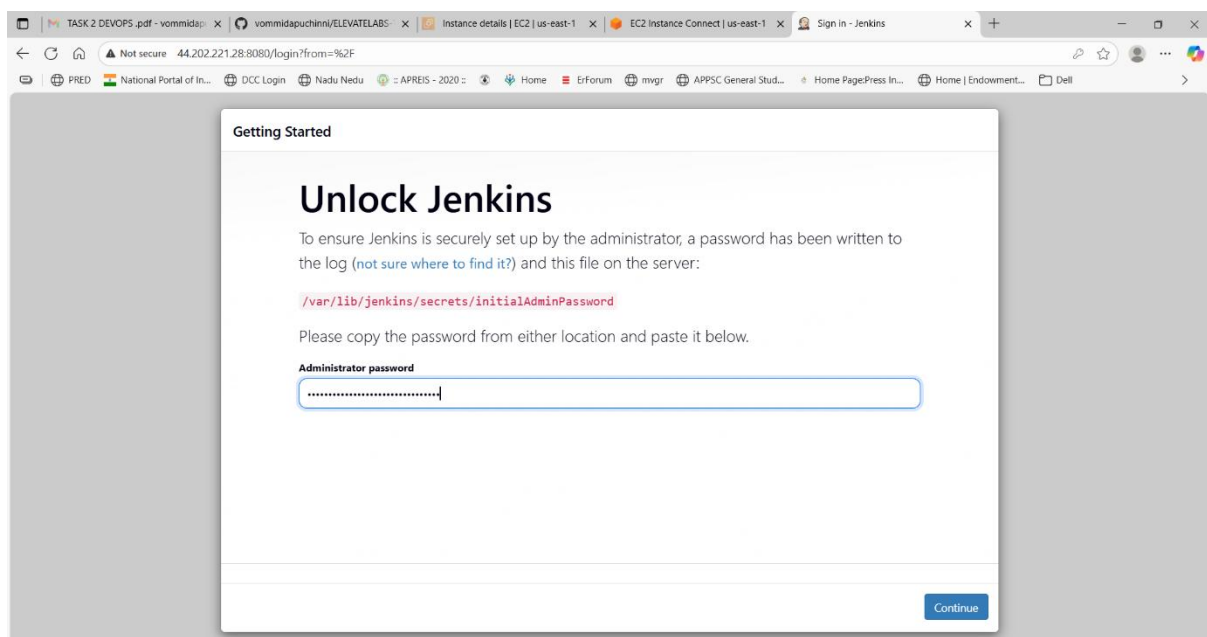
```
root@ip-172-31-0-8:~# apt update -y
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [1054 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [1314 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [264 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [164 kB]
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [1120 kB]
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [287 kB]
Get:19 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [377 kB]
Get:20 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [26.0 kB]
Get:21 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [1645 kB]
Get:22 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [359 kB]
Get:23 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 B]
Get:24 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [33.2 kB]
Get:25 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [6772 B]
Get:26 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [1940 B]

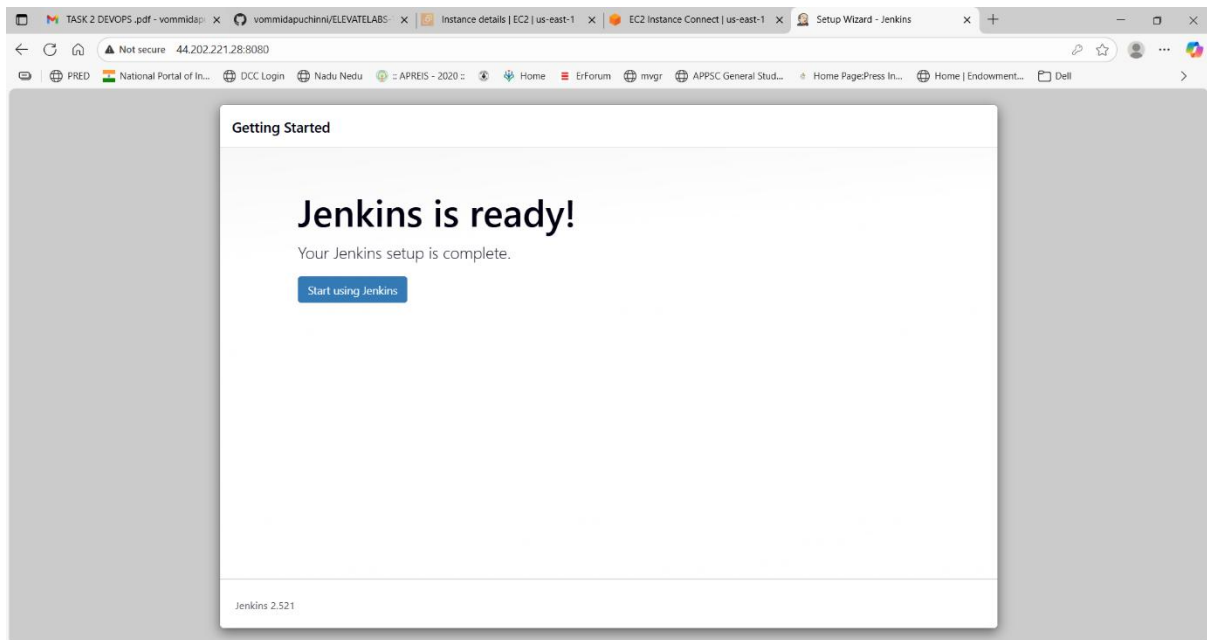
root@ip-172-31-0-8:~# sudo systemctl status jenkins
jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
   Active: active (running) since Tue 2025-08-05 07:37:18 UTC; 26s ago
     Main PID: 9560 (java)
       Tasks: 44 (limit: 1124)
      Memory: 350.7M (peak: 359.6M)
         CPU: 17.978s
    CGroup: /system.slice/jenkins.service
            └─9560 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

Aug 05 07:37:10 ip-172-31-0-8 jenkins[9560]: 795c284eac8742d59bb6alb40674d13b
Aug 05 07:37:10 ip-172-31-0-8 jenkins[9560]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Aug 05 07:37:10 ip-172-31-0-8 jenkins[9560]: *****
Aug 05 07:37:10 ip-172-31-0-8 jenkins[9560]: *****
Aug 05 07:37:10 ip-172-31-0-8 jenkins[9560]: *****
Aug 05 07:37:18 ip-172-31-0-8 jenkins[9560]: 2025-08-05 07:37:18.395+0000 [id=31] INFO jenkins.InitReactorRunner$1onAttained: Completed initialization
Aug 05 07:37:18 ip-172-31-0-8 jenkins[9560]: 2025-08-05 07:37:18.426+0000 [id=23] INFO hudson.lifecycle.Lifecycle#onReady: Jenkins is fully up and running
Aug 05 07:37:18 ip-172-31-0-8 jenkins[9560]: 2025-08-05 07:37:18.686+0000 [id=47] INFO h.m.DownloadService$Downloadable#load: Obtained the updated data
Aug 05 07:37:18 ip-172-31-0-8 jenkins[9560]: 2025-08-05 07:37:18.688+0000 [id=47] INFO hudson.util.Retrier#start: Performed the action check updates se
lines 1-20/20 (END)
root@ip-172-31-0-8:~# sudo cat /var/lib/jenkins/secrets/initialAdminPassword
795c284eac8742d59bb6alb40674d13b
root@ip-172-31-0-8:~#
```

- Updated the system and installed Jenkins using commands

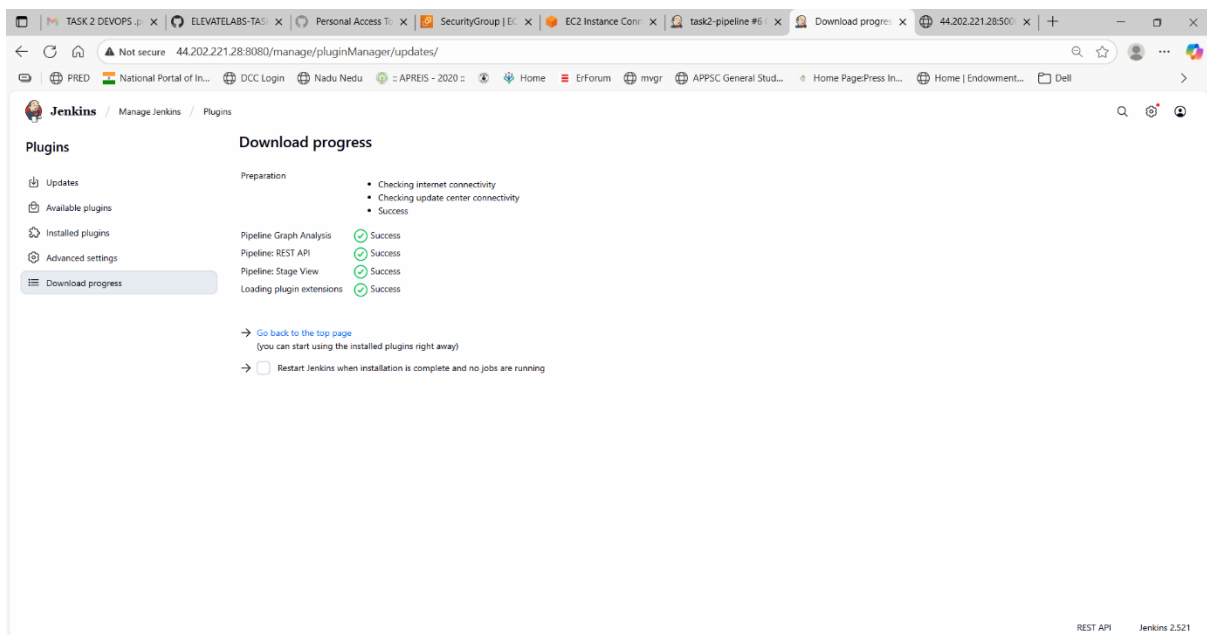
- Installed Java (required for Jenkins)
- Started Jenkins and accessed it from browser using `http://<EC2_PUBLIC_IP>:8080`
- **Note:** Used `sudo cat /var/lib/jenkins/secrets/initialAdminPassword` to get initial password
- `systemctl status Jenkins` → used to see the status of Jenkins.
- Added jenkins user to the docker group to allow Jenkins to run Docker commands





3. Install Jenkins Plugins

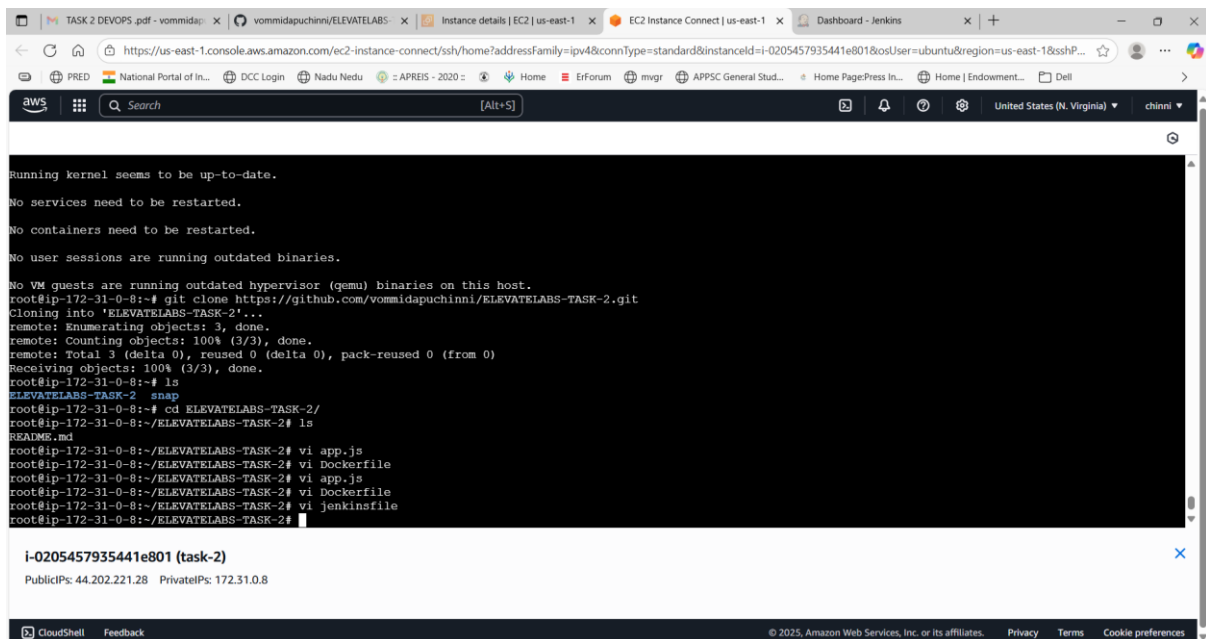
- Installed the following plugins:
 - Git
 - Docker Pipeline
 - Pipeline
 - GitHub Integration



Jenkins plugins add new features and integrations to Jenkins for different tools like Git, Docker, etc.

4. app.js

- A simple app with one endpoint:
- It contains an Express server with a single route
- Listens on port 5000
- **Purpose:** To test and deploy a working Node.js app via Jenkins CI/CD.
- GET / returns: Node.js App Deployed via Jenkins!



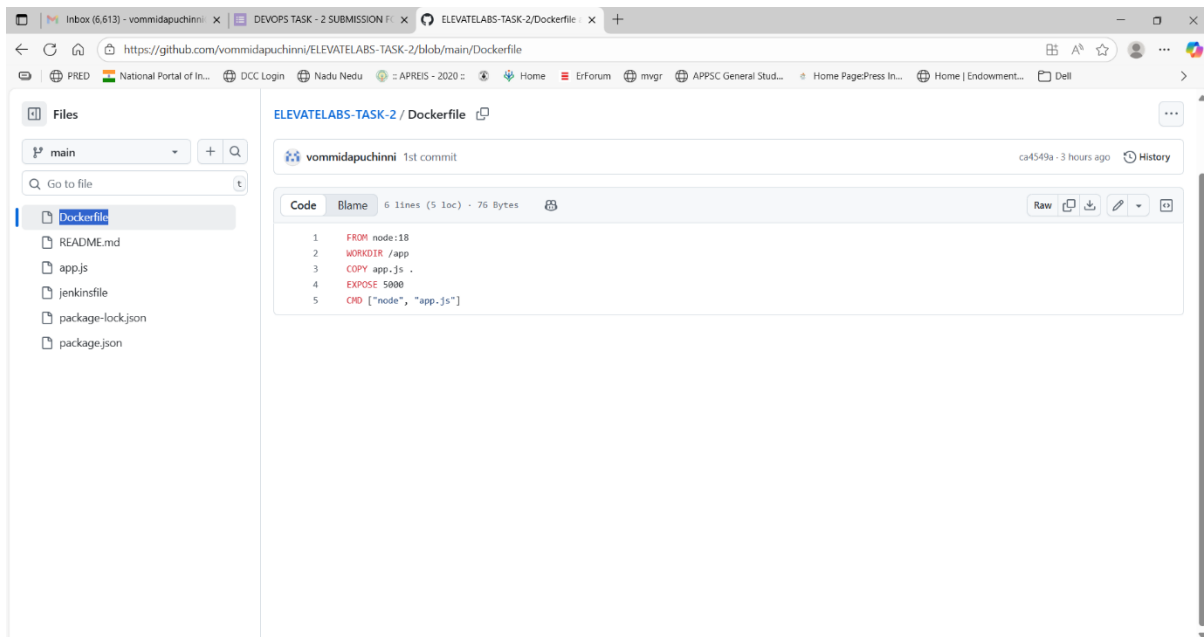
The screenshot shows an AWS CloudShell terminal window with the following content:

```
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.
root@ip-172-31-0-8:~# git clone https://github.com/vommidapuchinni/ELEVATELABS-TASK-2.git
Cloning into 'ELEVATELABS-TASK-2'...
remote: enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
root@ip-172-31-0-8:~# ls
ELEVATELABS-TASK-2  snap
root@ip-172-31-0-8:~# cd ELEVATELABS-TASK-2/
root@ip-172-31-0-8:~/ELEVATELABS-TASK-2# ls
README.md
root@ip-172-31-0-8:~/ELEVATELABS-TASK-2# vi app.js
root@ip-172-31-0-8:~/ELEVATELABS-TASK-2# vi Dockerfile
root@ip-172-31-0-8:~/ELEVATELABS-TASK-2# vi app.js
root@ip-172-31-0-8:~/ELEVATELABS-TASK-2# vi Dockerfile
root@ip-172-31-0-8:~/ELEVATELABS-TASK-2# vi jenkinsfile
root@ip-172-31-0-8:~/ELEVATELABS-TASK-2#
```

Below the terminal output, the instance ID is shown: i-0205457935441e801 (task-2). At the bottom, the public and private IP addresses are listed: PublicIPs: 44.202.221.28 PrivateIPs: 172.31.0.8.

5. Dockerfile: This file is used to build a Docker image for your Node.js app

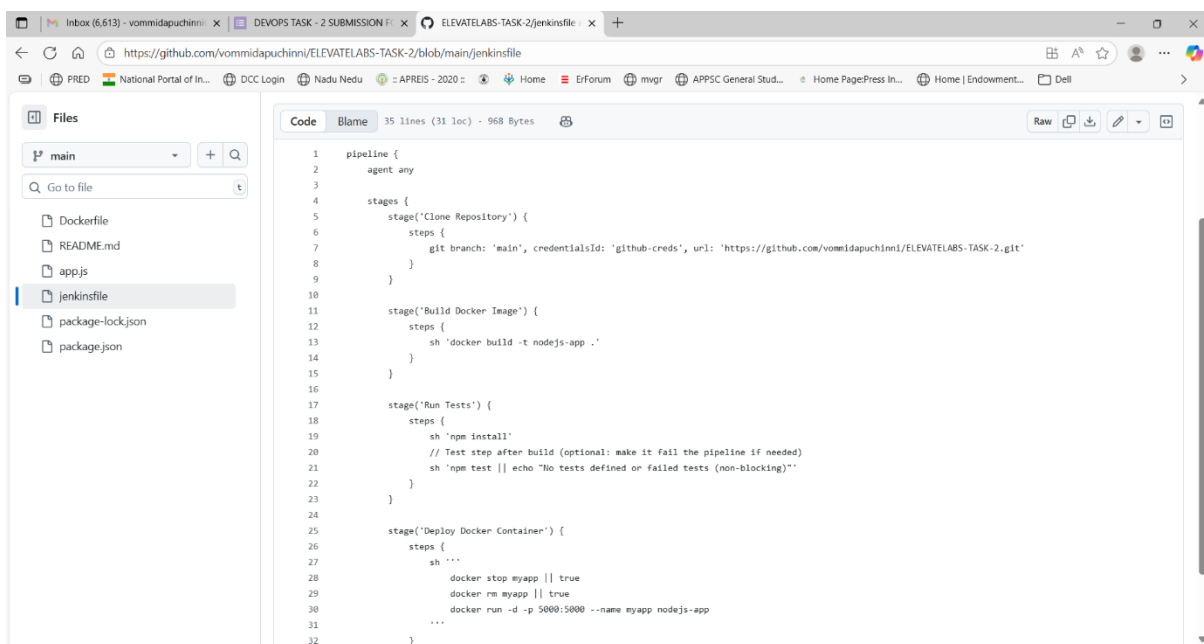
- Uses Node 18 base image
- Copies code
- Exposes port 5000
- Starts the app



6. Jenkinsfile (CI/CD Pipeline): Defines your CI/CD pipeline using stages:

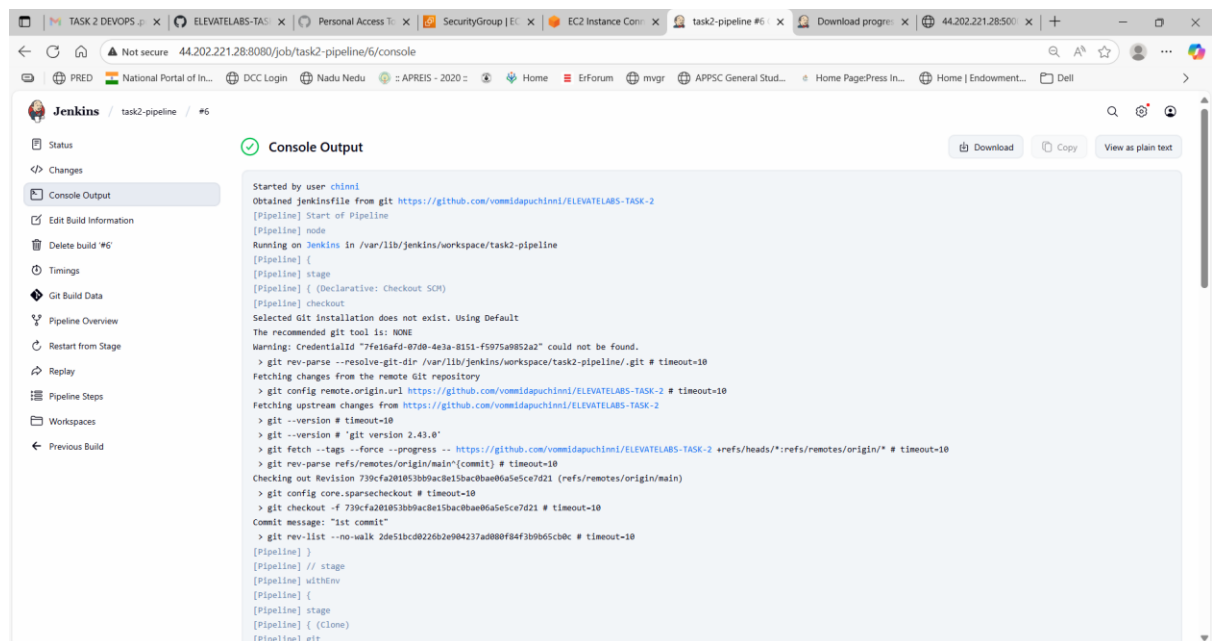
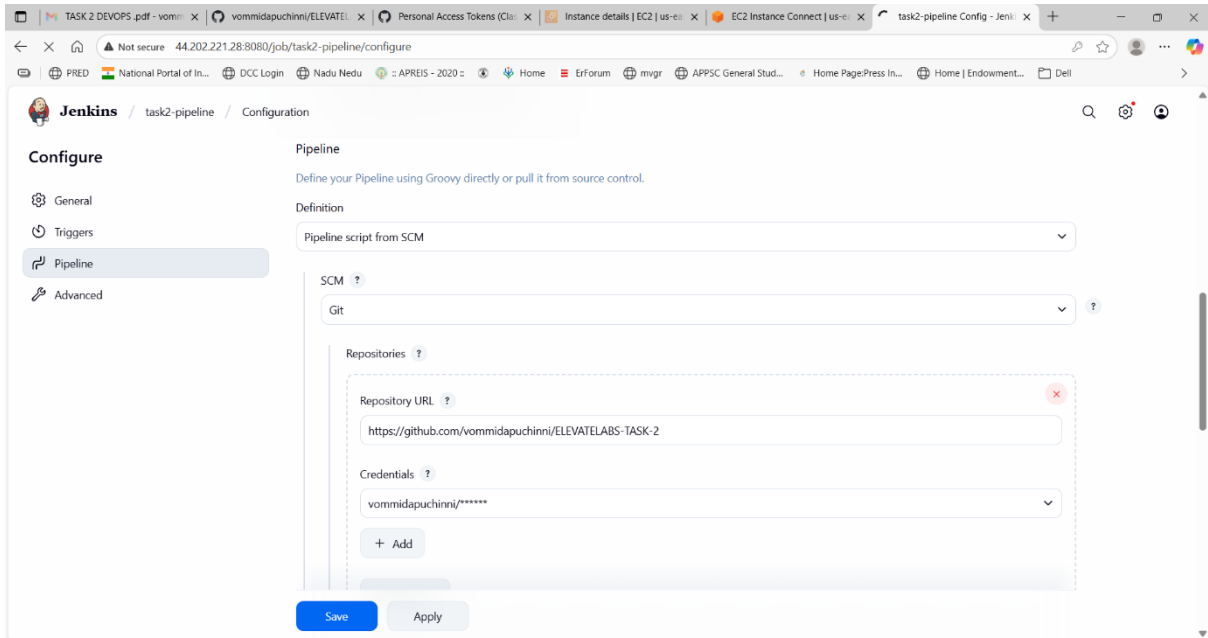
Stages:

- Clone: Pull code from GitHub
- Build: Build Docker image
- Install & Test: Install deps and test
- Deploy: Run container on port 5000



7. Configure Jenkins Pipeline Job

- Created a new Jenkins pipeline project
- Selected "Pipeline script from SCM"
- Set SCM to Git and provided GitHub repo URL
- Chose branch: main



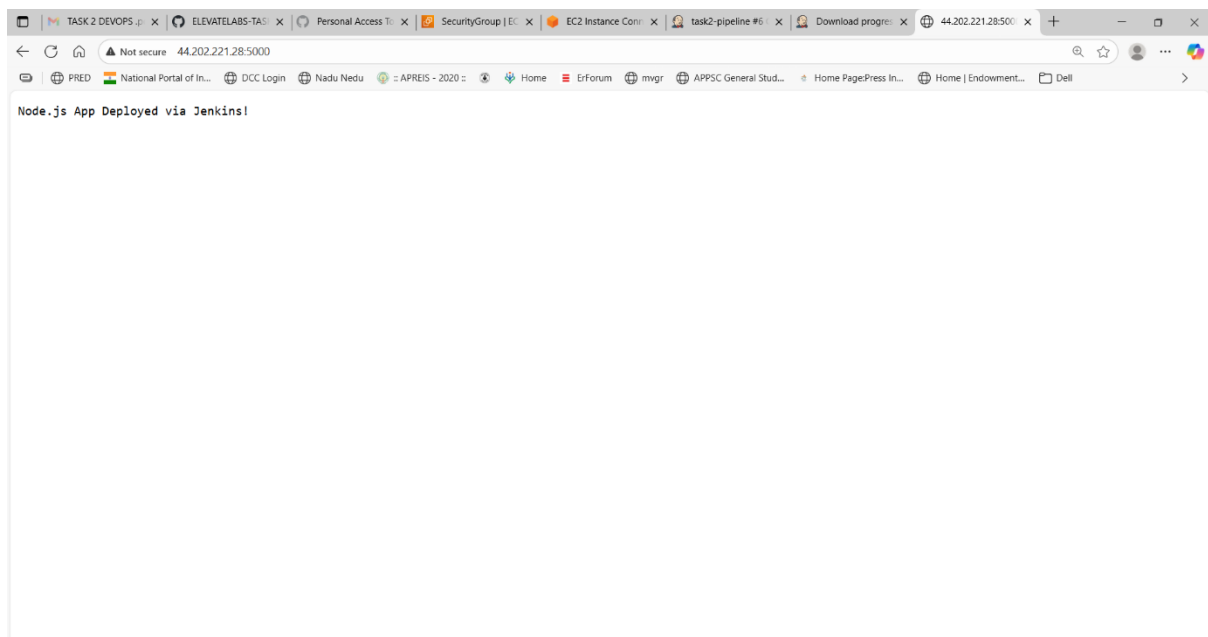
8. Access the App

The screenshot shows a terminal window in AWS CloudShell. The terminal output includes the following commands and results:

```
root@ip-172-31-0-8:~/ELEVATELABS-TASK-2# docker exec -it myapp curl http://localhost:6000
Node.js App Deployed via Jenkins!root@ip-172-31-0-8:~/ELEVATELABS-TASK-2# sudo ss -tln | grep 6000
tcp        LISTEN     0          4096      0.0.0.0:6000      0.0.0.0:*
tcp        LISTEN     0          4096      ::::6000         ::::6000
root@ip-172-31-0-8:~/ELEVATELABS-TASK-2# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
6ac2c0588ebc  nodejs-app    "docker-entrypoint.s..." 3 minutes ago  Up 3 minutes  0.0.0.0:6000->6000/tcp, :::6000->6000/tcp  myapp
root@ip-172-31-0-8:~/ELEVATELABS-TASK-2# curl http://localhost:6000
Node.js App Deployed via Jenkins!root@ip-172-31-0-8:~/ELEVATELABS-TASK-2#
root@ip-172-31-0-8:~/ELEVATELABS-TASK-2# curl http://http://44.202.221.28:6000
curl: (6) Could not resolve host: http
root@ip-172-31-0-8:~/ELEVATELABS-TASK-2# curl http://44.202.221.28:6000
Node.js App Deployed via Jenkins!root@ip-172-31-0-8:~/ELEVATELABS-TASK-2# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
72abbfde76e3  nodejs-app    "docker-entrypoint.s..." 15 seconds ago Up 14 seconds  0.0.0.0:5000->5000/tcp, :::5000->5000/tcp  myapp
root@ip-172-31-0-8:~/ELEVATELABS-TASK-2# curl http://44.202.221.28:5000
^Z
[3]+  Stopped                  curl http://44.202.221.28:5000
root@ip-172-31-0-8:~/ELEVATELABS-TASK-2# curl http://44.202.221.28:5000
^Z
[4]+  Stopped                  curl http://44.202.221.28:5000
root@ip-172-31-0-8:~/ELEVATELABS-TASK-2# curl http://localhost:5000
Node.js App Deployed via Jenkins!root@ip-172-31-0-8:~/ELEVATELABS-TASK-2#
root@ip-172-31-0-8:~/ELEVATELABS-TASK-2# curl http://44.202.221.28:5000
Node.js App Deployed via Jenkins!root@ip-172-31-0-8:~/ELEVATELABS-TASK-2#
```

Below the terminal output, the instance ID is shown: i-0205457935441e801 (task-2). Public IPs: 44.202.221.28 Private IPs: 172.31.0.8.

- Browser: `http://<ec2-ip>:5000`
- Confirmed the app is up and running inside Docker

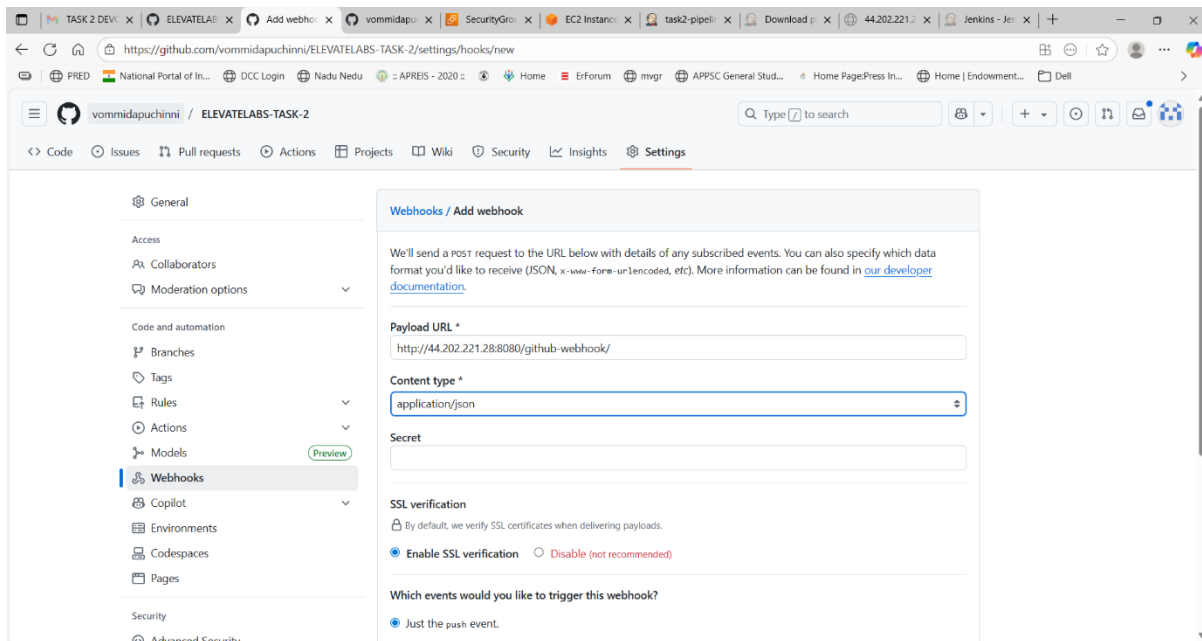


I am going to add a webhook for automatic triggering when there is an change in git repo

9. GitHub Webhook Setup

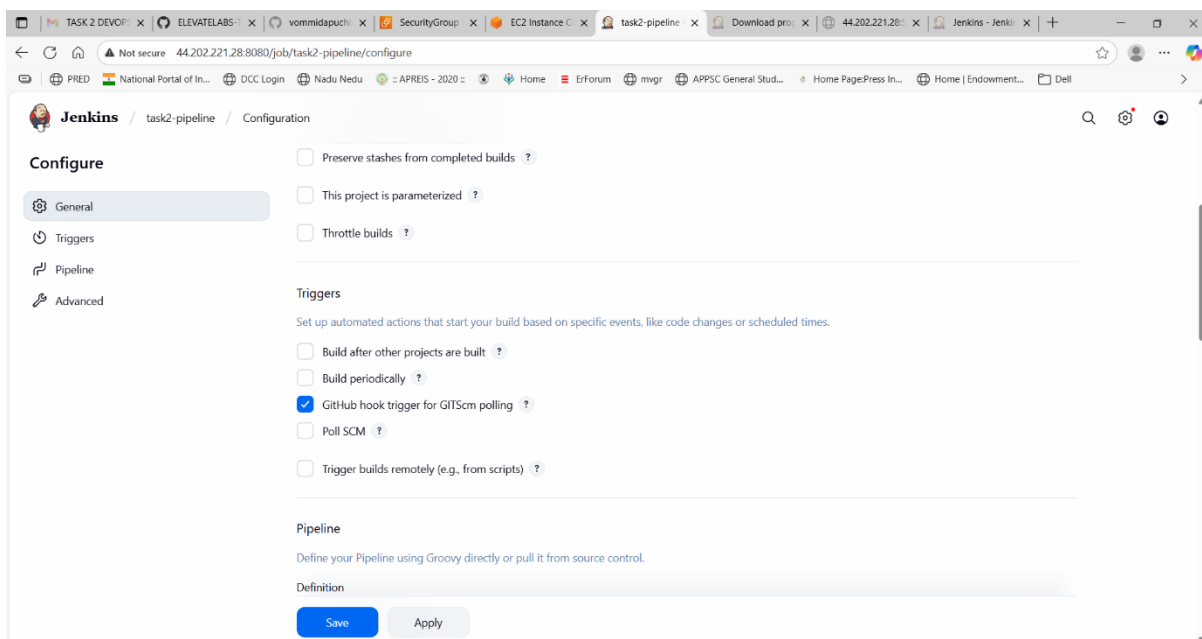
- GitHub → Repo → Settings → Webhooks → Add Webhook
- Payload URL: `http://<your-ec2-ip>:8080/github-webhook/`
- Trigger: Just push events

- Click add webhook



10. Enable Webhook Trigger in Jenkins Job

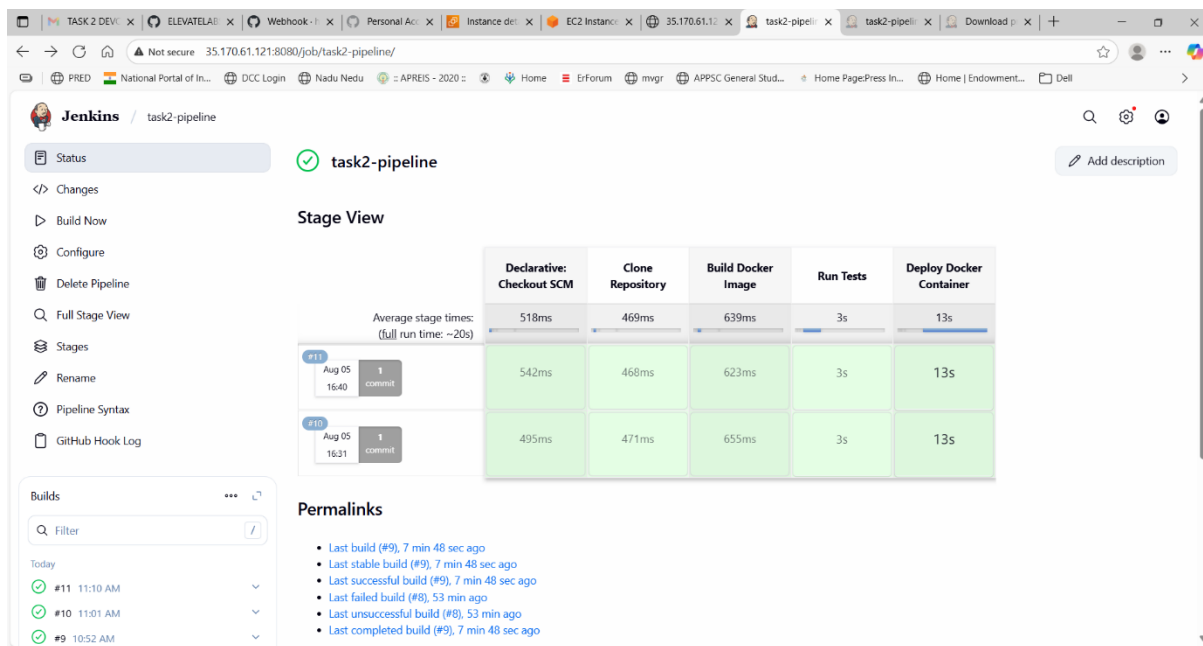
- Go to your Jenkins job → Configure → under Build Triggers:
- Tick: GitHub hook trigger for GITScm polling
- Save it come back



11. Pipeline Triggered Automatically

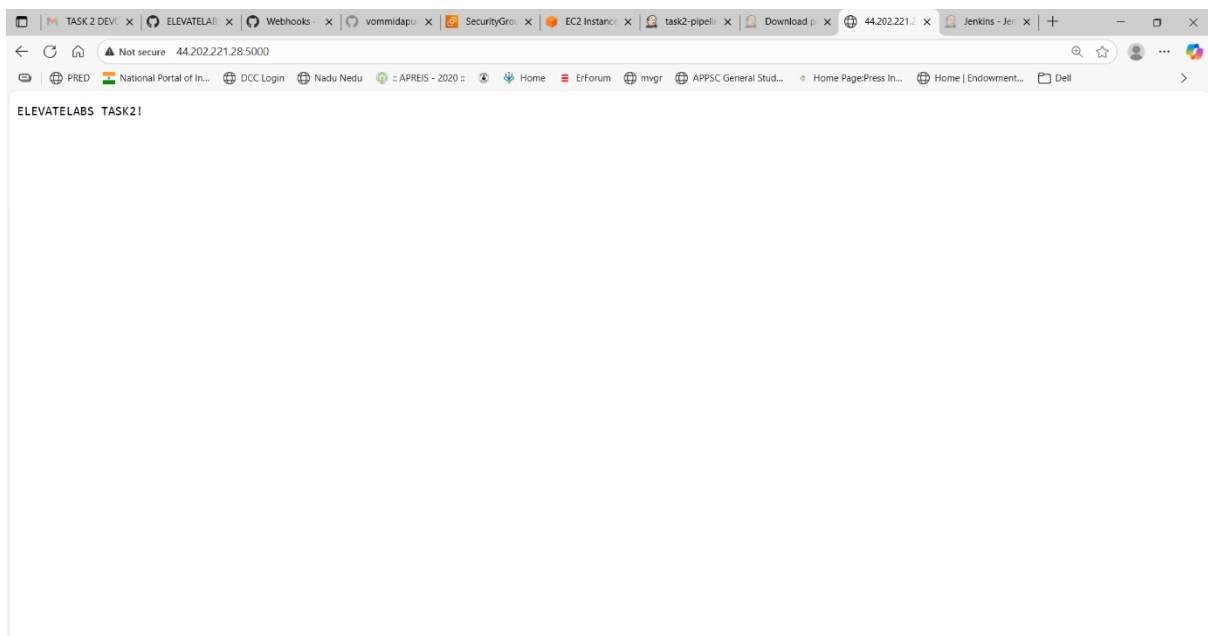
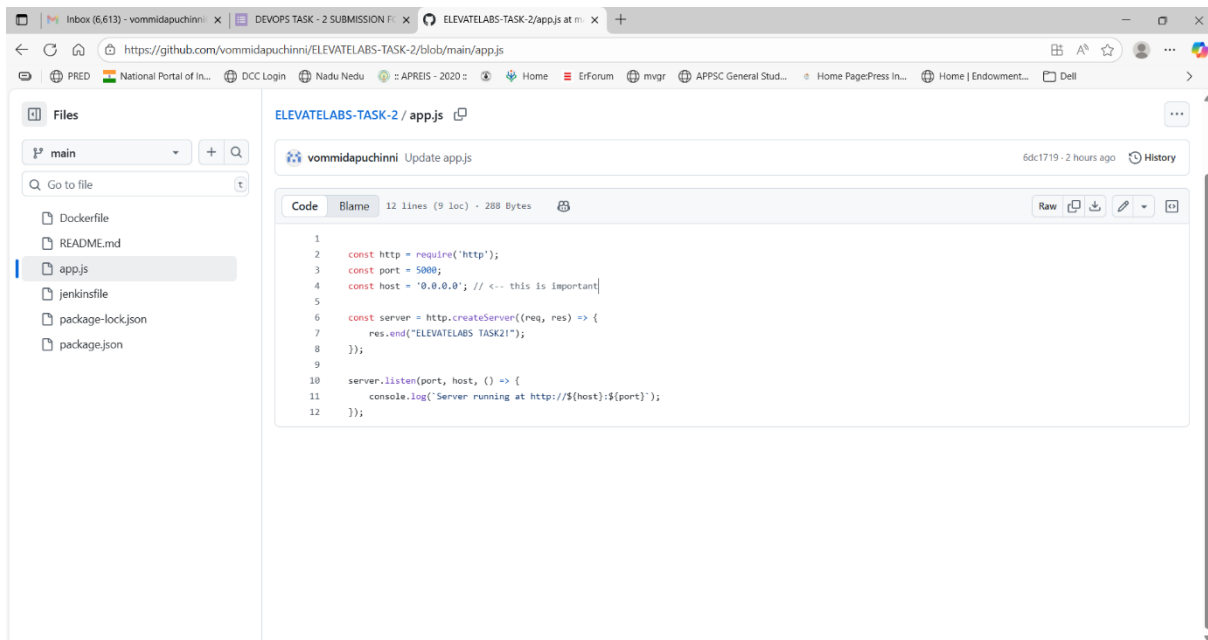
- Every time I push code to GitHub, Jenkins automatically:
 - Clones the repo
 - Builds the Docker image
 - Runs the container

12. Here as mentioned in instructions I have build tested and deployed.



Checking that automatic triggering is happening or not I have changed the code in app.js (node.js app deployed via Jenkins to elevatelabs task2!)

Its automatically triggered and accessed the application via public ip



13. Troubleshooting

Issue: Jenkins can't run Docker commands

Fix: Add Jenkins user to Docker group

sudo usermod -aG docker jenkins

sudo systemctl restart docker

Issue: App not accessible on browser

Fix:

- Check EC2 security group has port 5000 open
- Confirm container is running with `docker ps`
- Check logs using `docker logs myapp`

Issue: Webhook not triggering build

Fix:

- Ensure webhook URL is correct
- Jenkins GitHub plugin installed
- GitHub and Jenkins connected properly

14. Conclusion

This project helped me learn how to:

- Create and deploy a Node.js app
- Use Jenkins to automate CI/CD pipeline
- Containerize an app using Docker
- Host and manage the pipeline on AWS EC2
- Integrate GitHub with Jenkins using webhooks

I also solved errors related to Docker permissions, webhook misconfiguration, and deployment issues during the task.