Project: Reactjs E-commerce Application

1.Docker:

1.Docker and Docker-compose Installed:

```
root@ip-172-31-24-31:~# docker --version && docker-compose --version
Docker version 27.5.1, build 27.5.1-0ubuntu3~24.04.2
docker-compose version 1.29.2, build unknown
root@ip-172-31-24-31:~#
```

2.Source Code File:

```
root@ip-172-31-24-31:~/Project1# ls
build
root@ip-172-31-24-31:~/Project1#
```

3.Creating Dockerfile:

```
root@ip-172-31-24-31:~/Project1# nano Dockerfile
```

4.DockerFile Code:

```
  GNU nano 7.2                                                  Dockerfile
FROM nginx:alpine

RUN rm -rf /usr/share/nginx/html/*

COPY build/ /usr/share/nginx/html/

EXPOSE 80

CMD ["nginx", "-g", "daemon off;"]
```

5.Creating Docker Image:

```
root@ip-172-31-24-31:~/Project1# docker build -t static-app .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  3.419MB
Step 1/6 : FROM nginx:alpine
alpine: Pulling from library/nginx
9824c27679d3: Pull complete
6bc572a340ec: Pull complete
403e3f251637: Pull complete
9adfbae99cb7: Pull complete
7a8a46741e18: Pull complete
c9ebe2ff2d2c: Pull complete
a992fbc61ecc: Pull complete
cb1ff4086f82: Pull complete
Digest: sha256:42a516af16b852e33b7682d5ef8acbd5d13fe08fecadc7ed98605ba5e3b26ab8
Status: Downloaded newer image for nginx:alpine
 ---> 4a86014ec699
Step 2/6 : WORKDIR /usr/share/nginx/html
 ---> Running in df7b442476fa
 ---> Removed intermediate container df7b442476fa
 ---> afbdd68e0ffd
Step 3/6 : RUN rm -rf ./*
```

6.Docker Image Created:

```
root@ip-172-31-24-31:~/Project1# docker images
REPOSITORY   TAG       IMAGE ID       CREATED              SIZE
static-app   latest    96a05b569493   About a minute ago   55.9MB
nginx        alpine    4a86014ec699   9 days ago           52.5MB
root@ip-172-31-24-31:~/Project1#
```

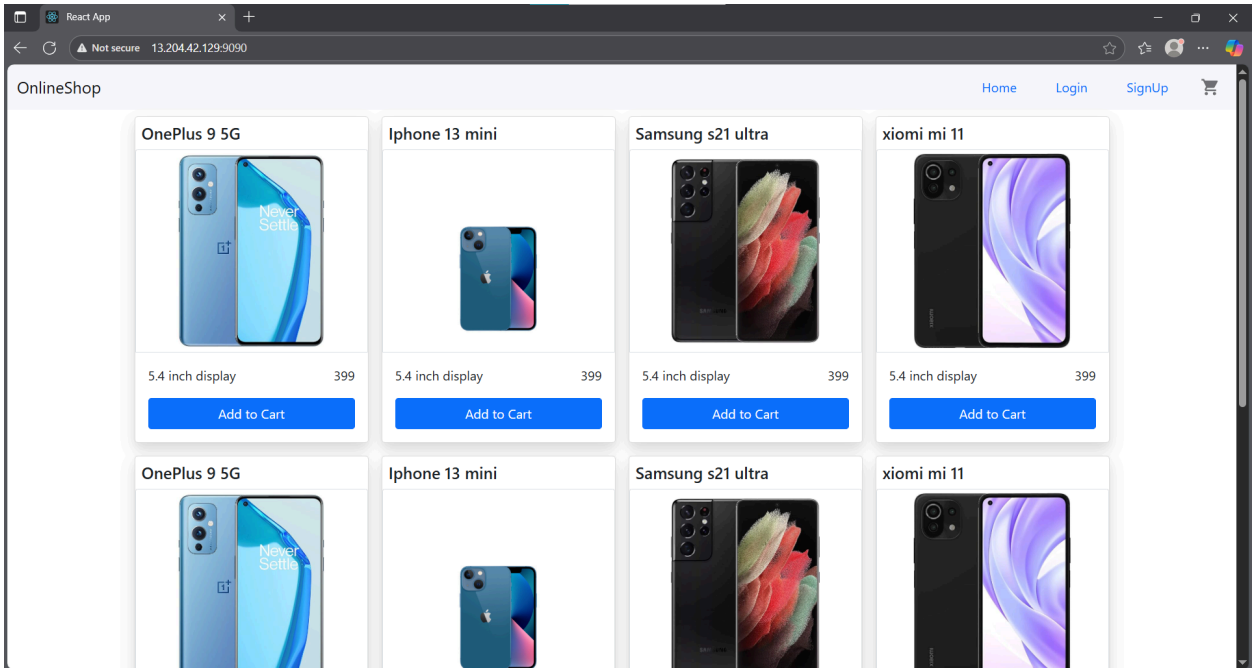
7.Docker-compose Code:

```
  GNU nano 7.2                                  docker-compose.yaml
version: '3.8'


services:
  Dev:
    image: static-app:latest
    ports:
      - "9090:80"
```

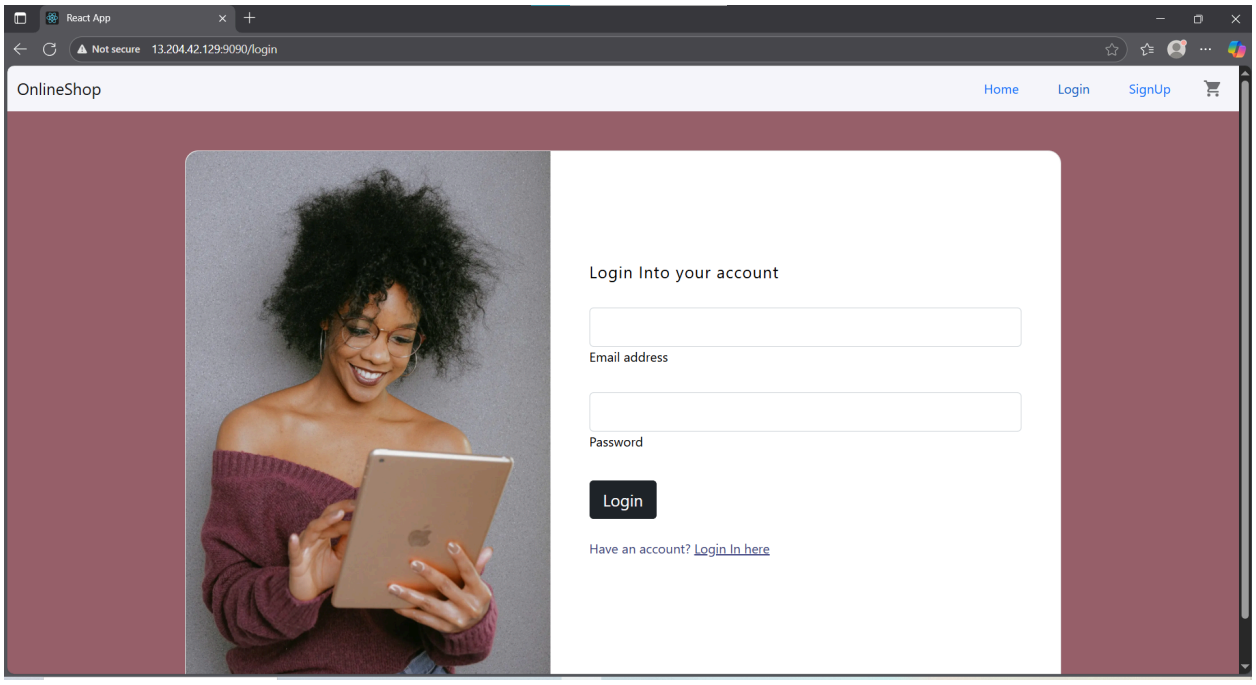
8.Running The Docker-compose:

```
root@ip-172-31-24-31:~/Project1# docker-compose up -d
Creating network "project1_default" with the default driver
Creating project1_Dev_1 ... done
root@ip-172-31-24-31:~/Project1#
```

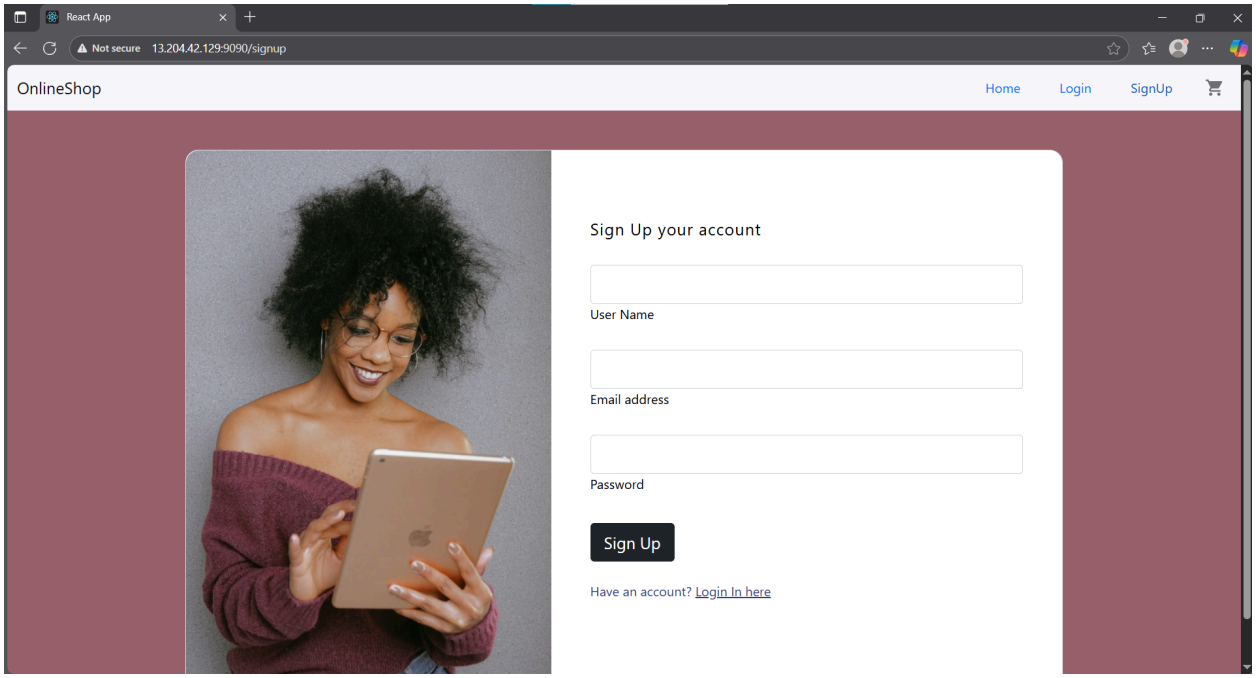
9.Docker Application Output 1:



10.Docker Application Output 2:



11.Docker Application Output 3:



2.Bash_Scripting:

1.Creating Script file to Build Docker Image:

2.Creating Script file to Run Docker Container:

```
  GNU nano 7.2                                                deploy.sh *
#! /bin/bash

IMAGE_NAME="static-web"

CONTAINER_NAME="static_application"

docker rm -f $CONTAINER_NAME 2>/dev/null || true

docker run -idt -p "9091:80" --name $CONTAINER_NAME $IMAGE_NAME
```

3.Using Docker Ignore to ignore some files in Docker Build:

```
  GNU nano 7.2                                                .dockerignore *
*.sh

*.yaml
```

4.Giving Executable permission to Build and Deploy file:

```
root@ip-172-31-24-31:~/Project1# chmod +x build.sh deploy.sh
root@ip-172-31-24-31:~/Project1#
```

5.Running Build File:

```
root@ip-172-31-24-31:~/Project1# ./build.sh
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon   3.42MB
Step 1/5 : FROM nginx:alpine
 ---> 4a86014ec699
Step 2/5 : RUN rm -rf /usr/share/nginx/html/*
 ---> Using cache
 ---> dcf90923529c
Step 3/5 : COPY build/ /usr/share/nginx/html/
 ---> Using cache
 ---> 8ae700e55a18
Step 4/5 : EXPOSE 80
 ---> Using cache
 ---> 8d976824691c
Step 5/5 : CMD ["nginx", "-g", "daemon off;"]
 ---> Using cache
 ---> 836000789d26
Successfully built 836000789d26
Successfully tagged static-web:latest
root@ip-172-31-24-31:~/Project1#
```

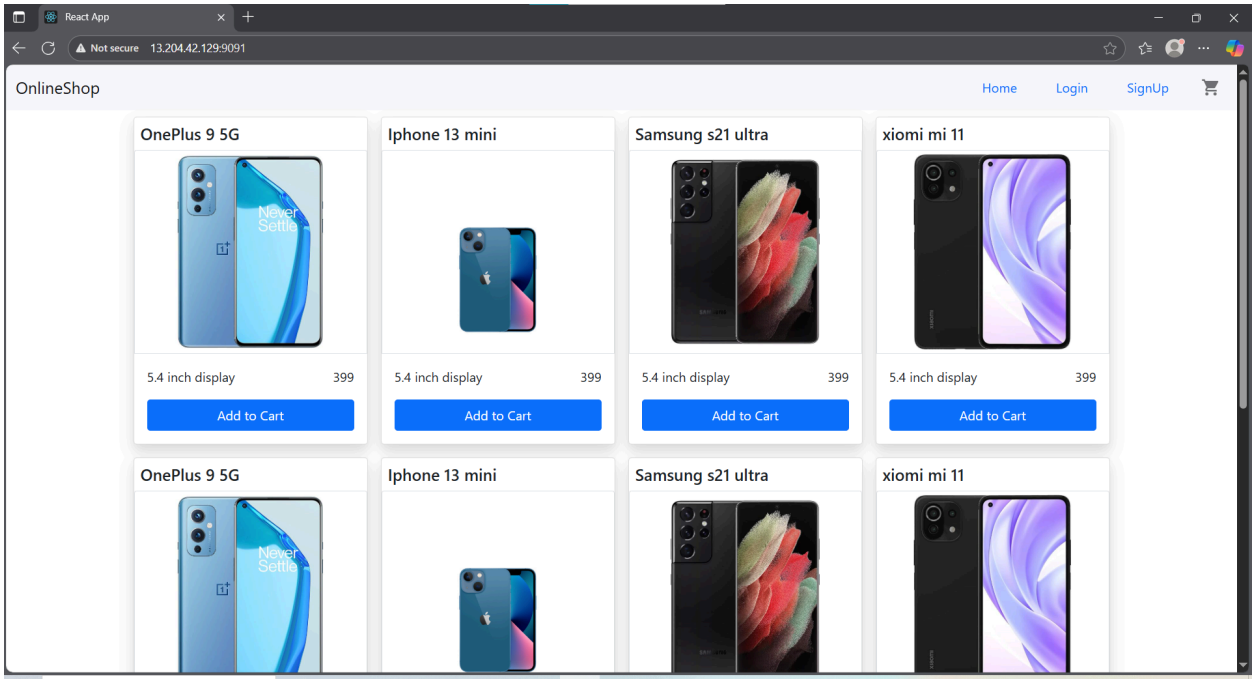
6.Docker Image created:

```
root@ip-172-31-24-31:~/Project1# docker images
REPOSITORY   TAG       IMAGE ID       CREATED          SIZE
static-web   latest    fded10026b8c   17 seconds ago   55.1MB
nginx        alpine    4a86014ec699   9 days ago       52.5MB
root@ip-172-31-24-31:~/Project1#
```

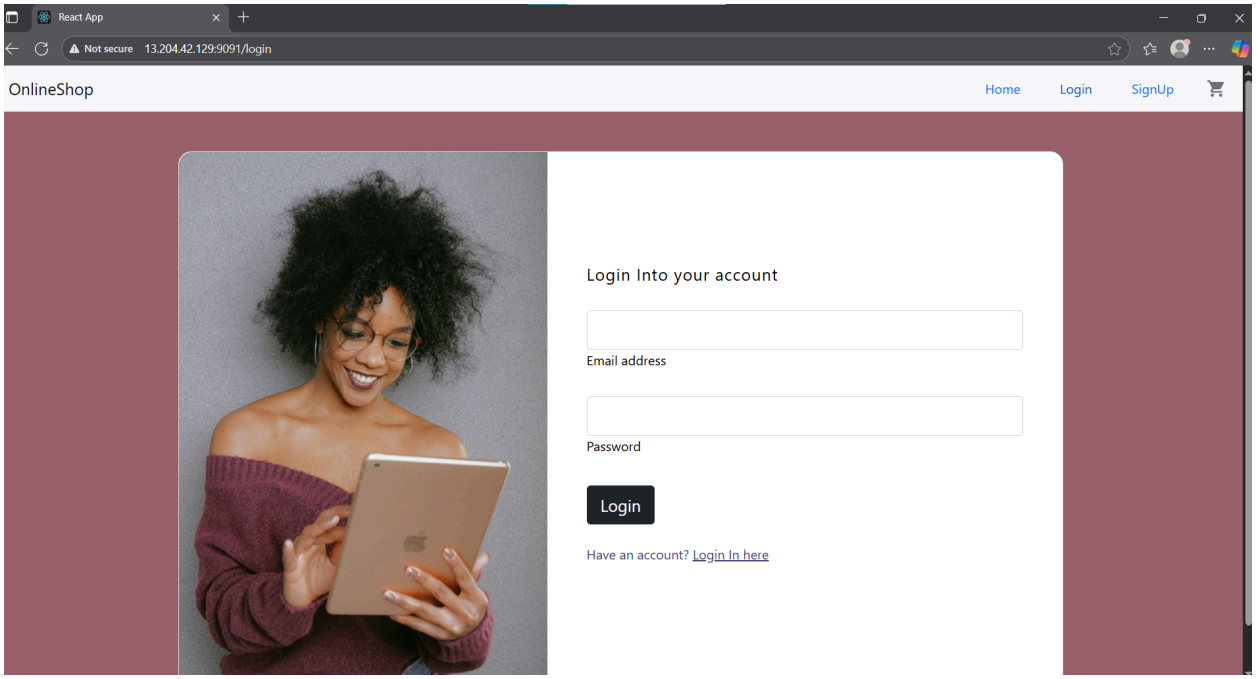
7.Docker Container Created:

```
root@ip-172-31-24-31:~/Project1# ./deploy.sh
76e6a35d87d1fead112387509a039673b2892befbbce524b03231cd2fff83f72
root@ip-172-31-24-31:~/Project1# docker ps
CONTAINER ID   IMAGE        COMMAND                  CREATED         STATUS         PORTS                                       NAMES
76e6a35d87d1   static-web   "/docker-entrypoint.…"   7 seconds ago   Up 6 seconds   0.0.0.0:9091->80/tcp, [::]:9091->80/tcp   static_application
root@ip-172-31-24-31:~/Project1#
```

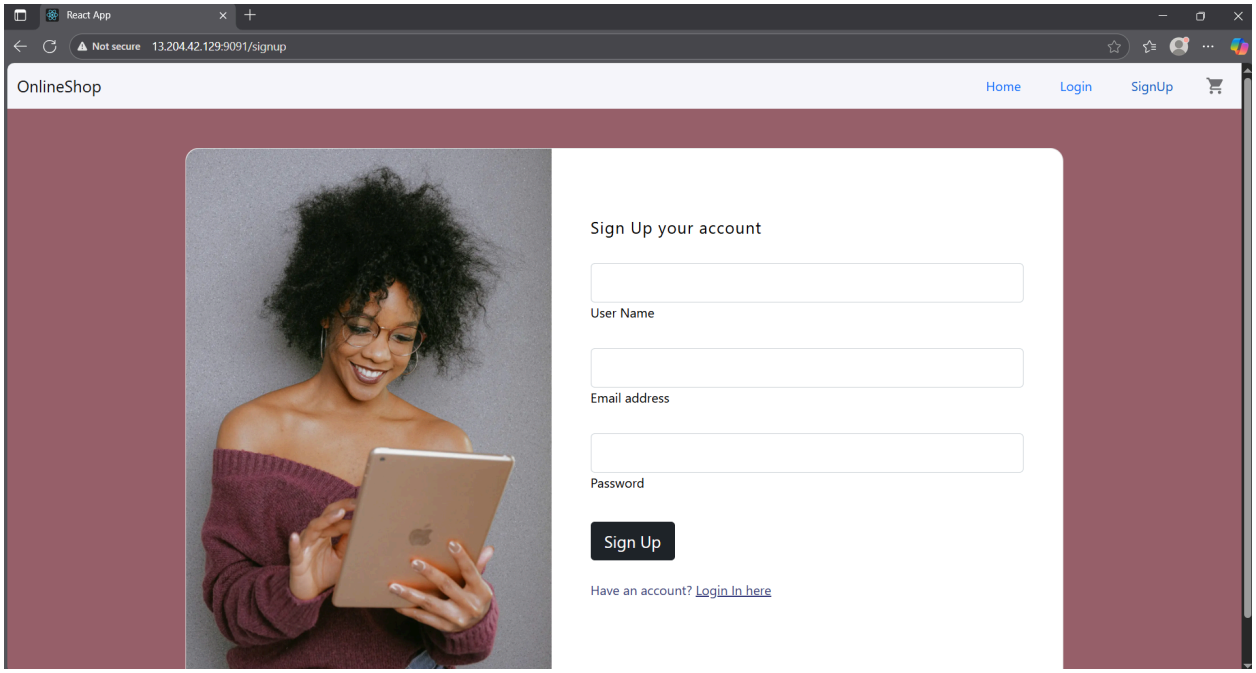
8.Output 1:



9.Output 2:

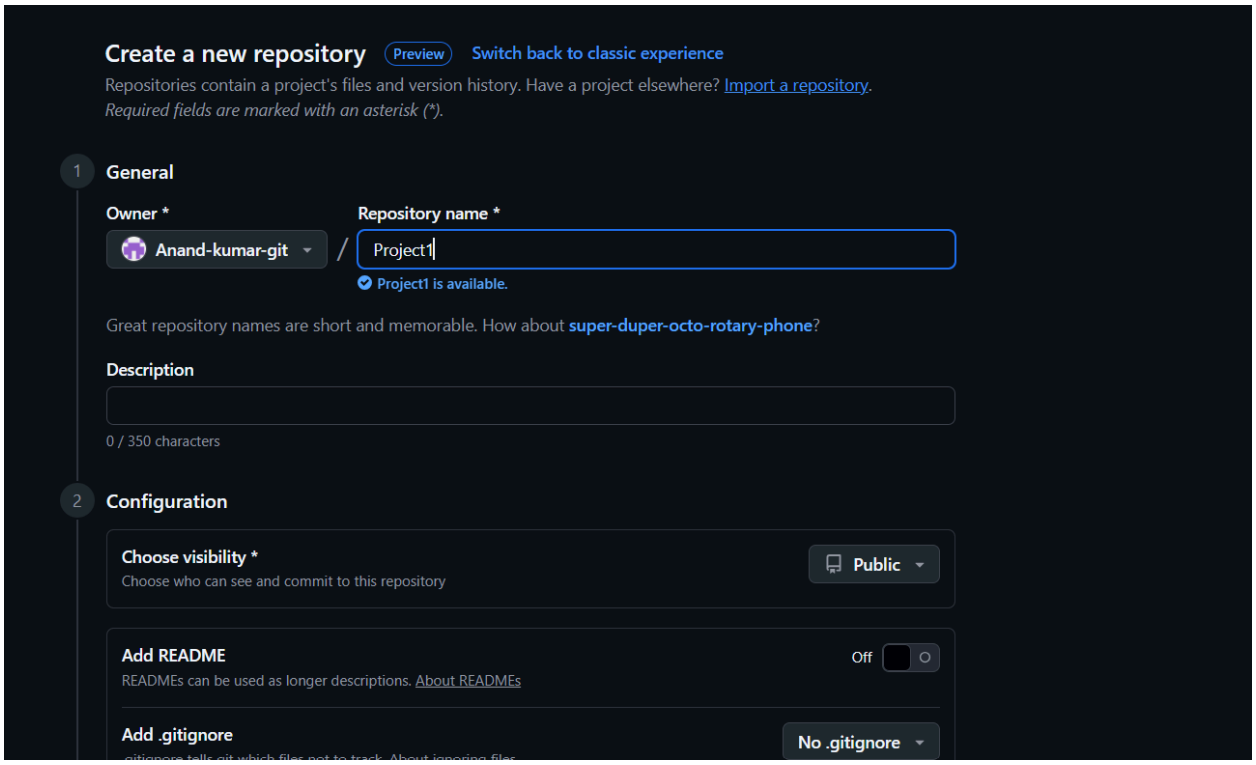


10.Output 3:

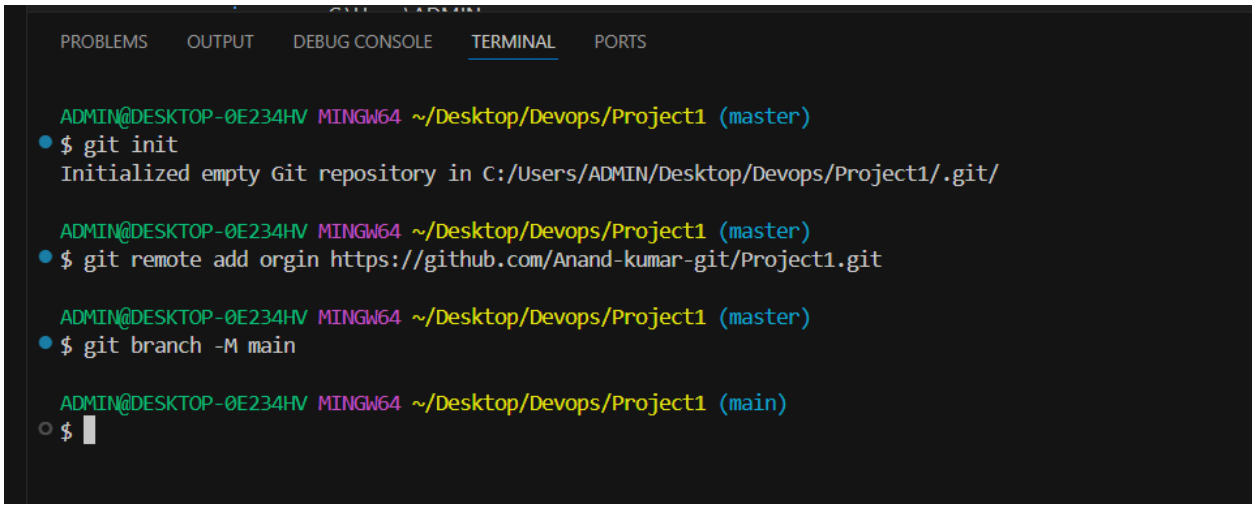

React App
Not secure 13.204.42.129:9091/signup
OnlineShop
Home
Login
SignUp
Sign Up your account
User Name
Email address
Password
Sign Up
Have an account? Login In here

3.Version_Control:

1.Creating GitHub Repository:



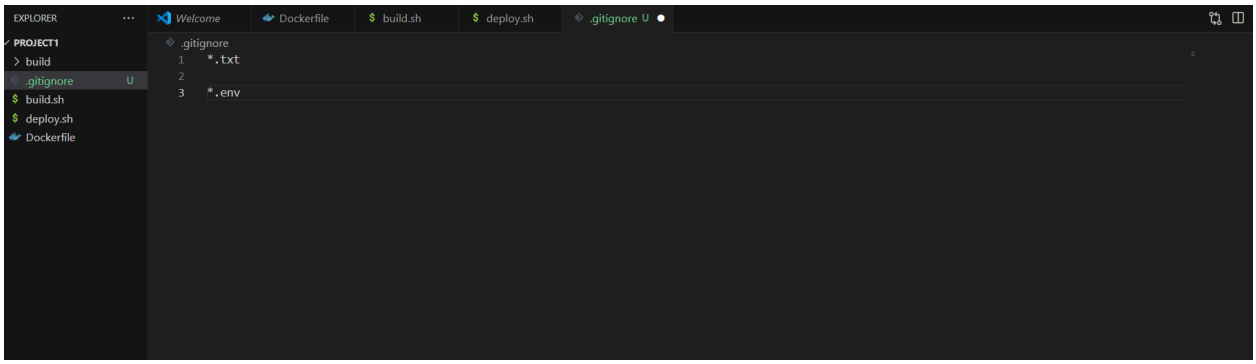
2.Cloning the git Repository:



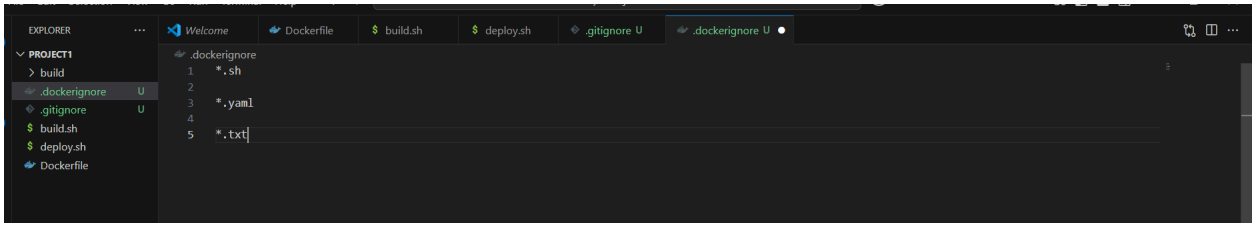
3.Creating Dev Branch:

```
ADMIN@DESKTOP-0E234HV MINGW64 ~/Desktop/Devops/Project1 (main)
$ git checkout -b dev
Switched to a new branch 'dev'

ADMIN@DESKTOP-0E234HV MINGW64 ~/Desktop/Devops/Project1 (dev)
$
```

4.Using Git ignore file:

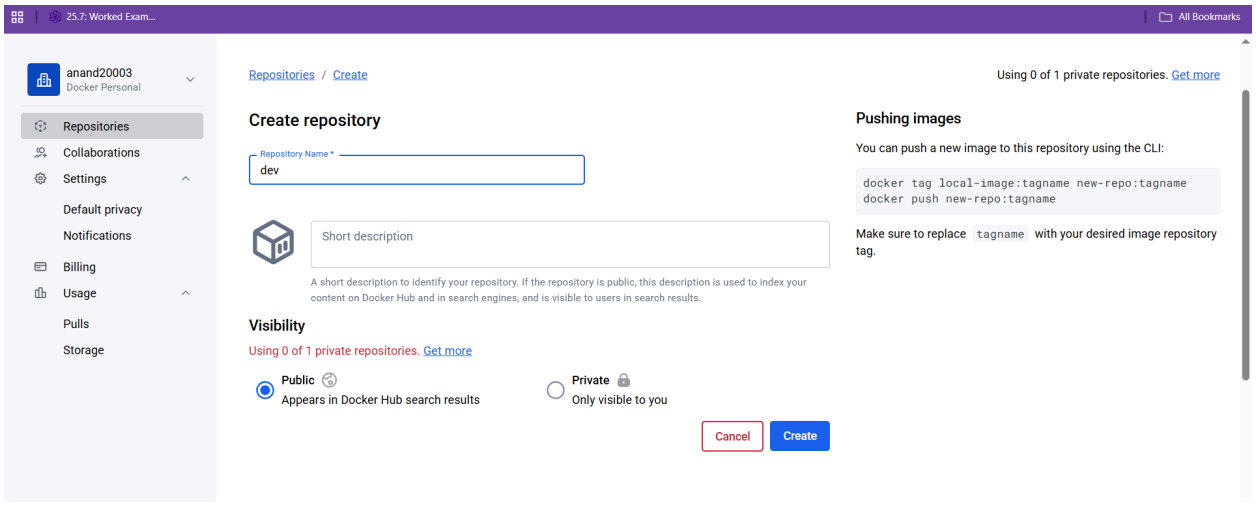


5.Using .dockerignore file:

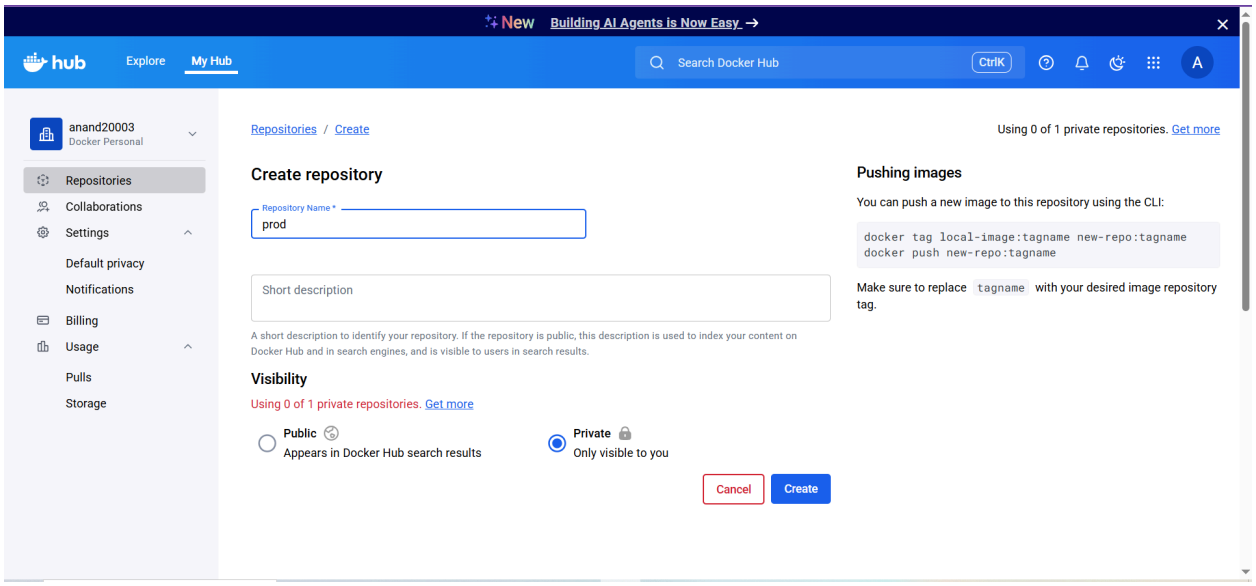


4.Docker_Hub:

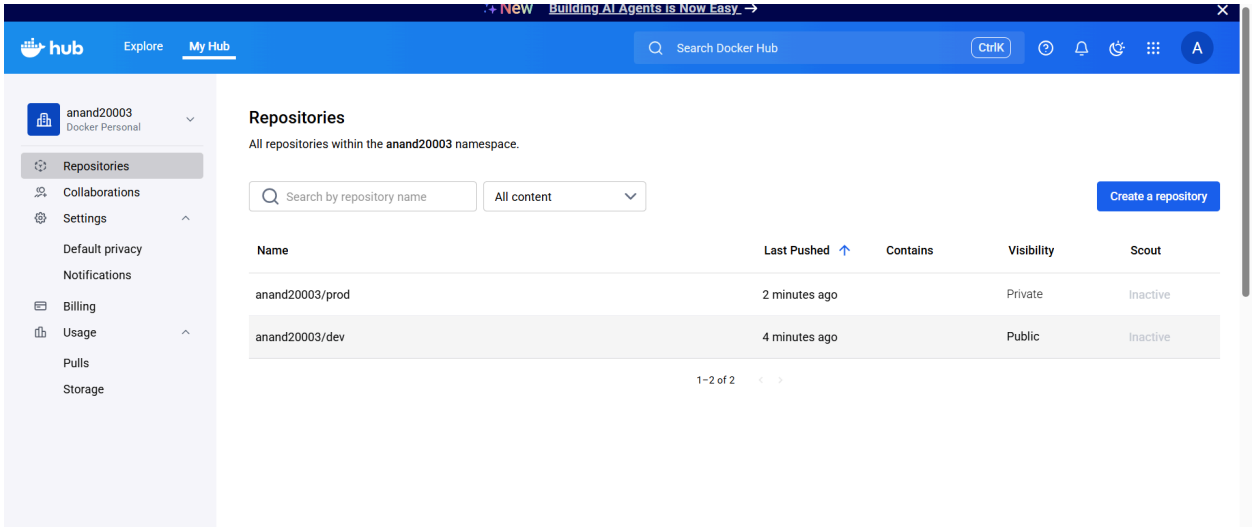
1.Creating Dev Repository in Dockerhub:



2.Creating Prod Private Repository:

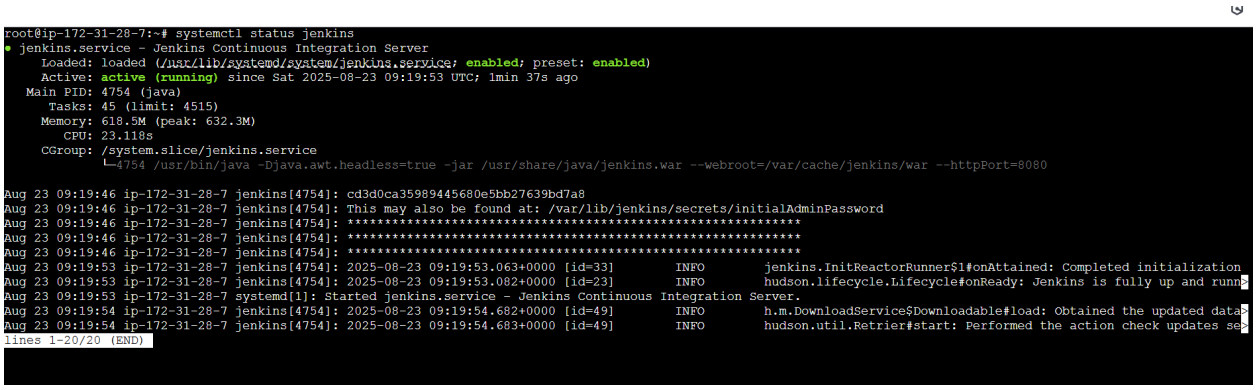


3.Dev and Prod Repository Created:

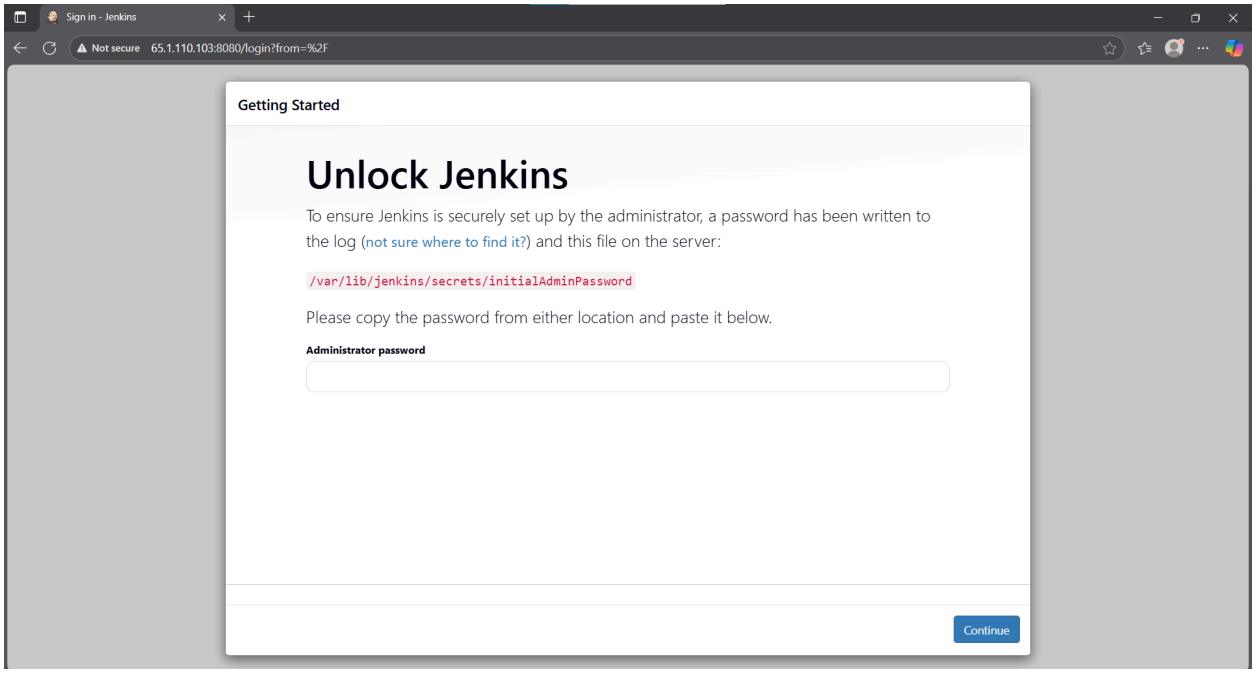


5.Jenkins:

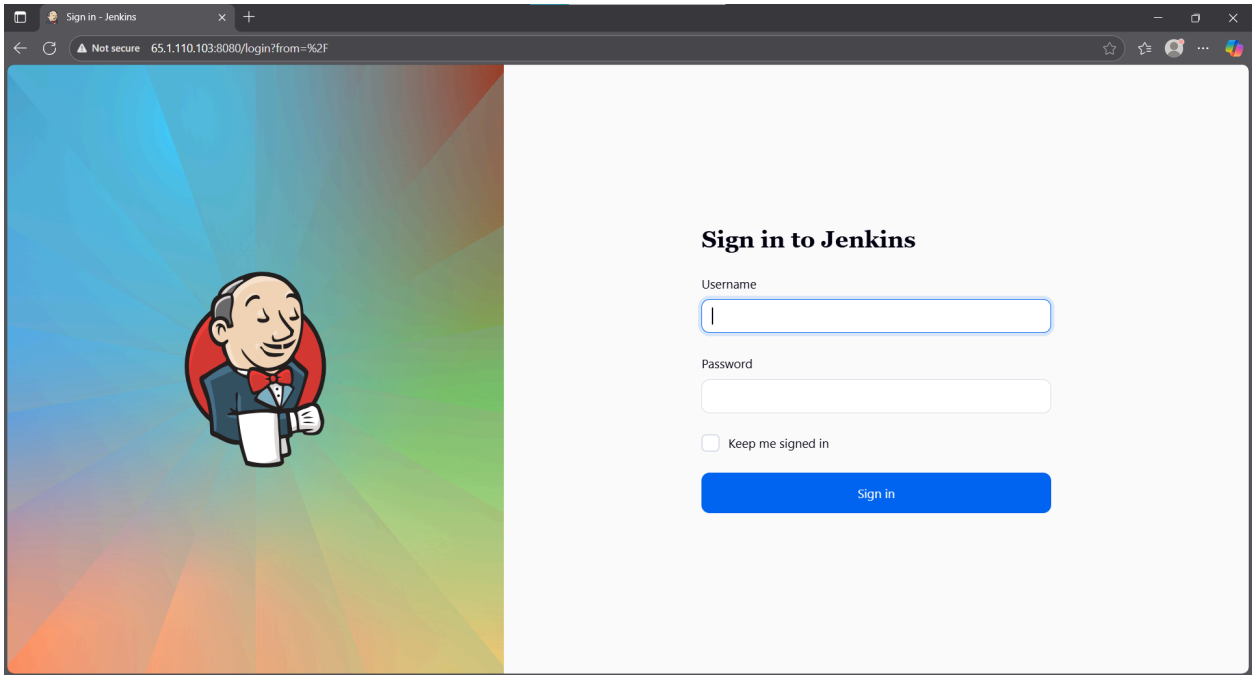
1.Jenkins Installed:



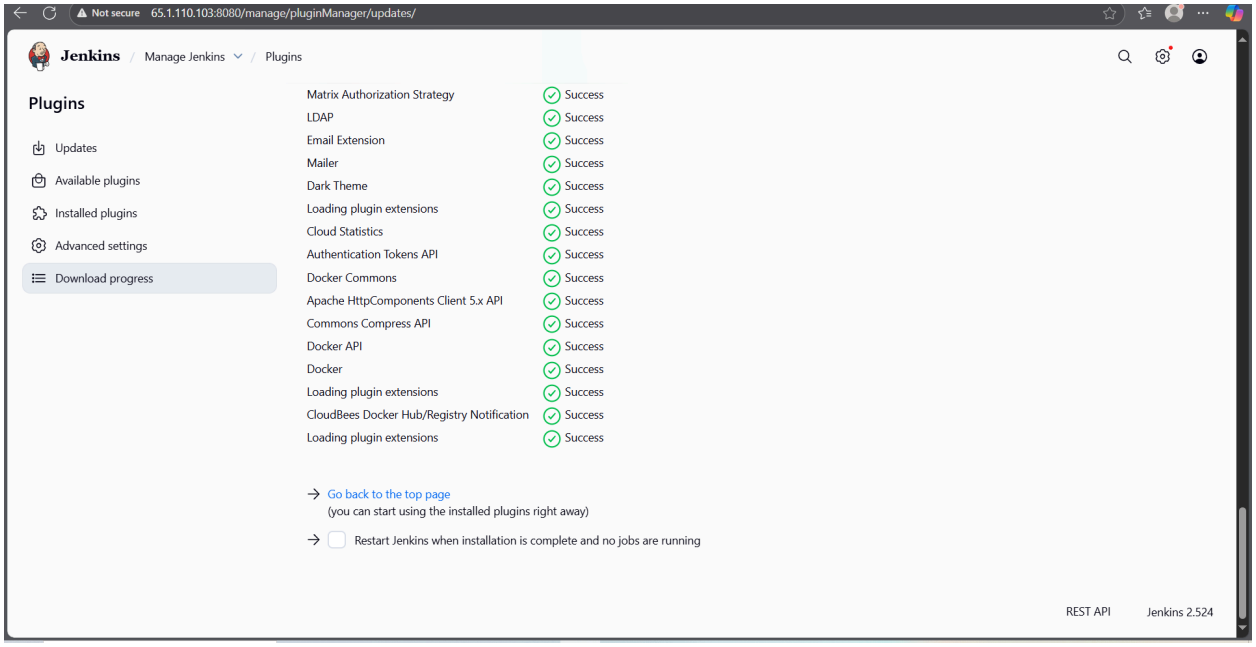
2.Acsessing Jenkins With Port 8080:



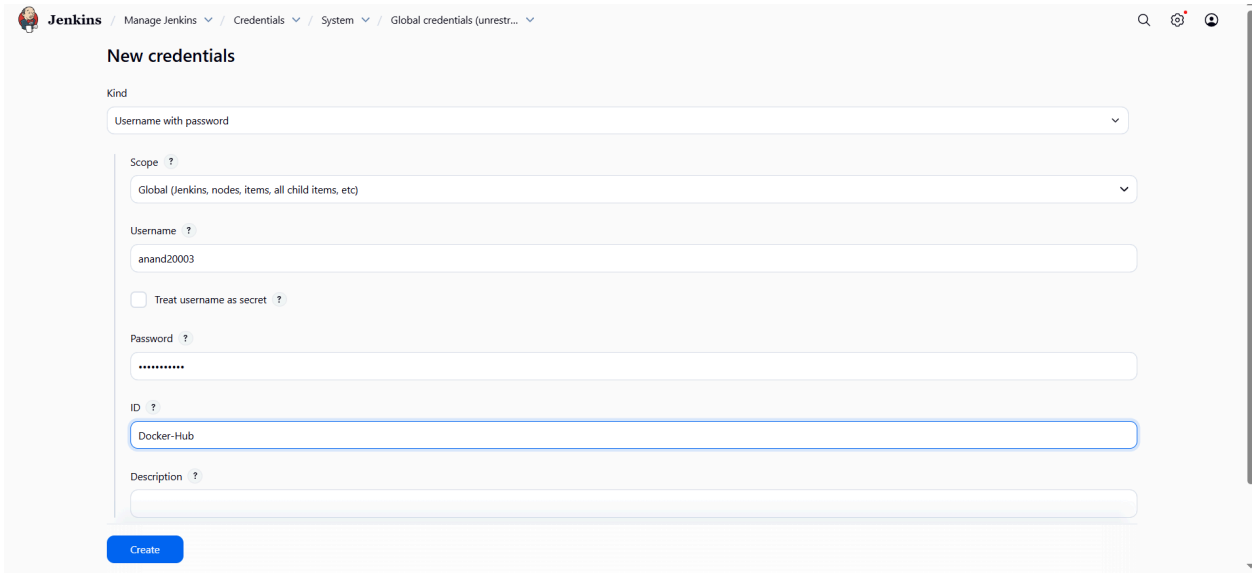
3.Jenkins Login Page:



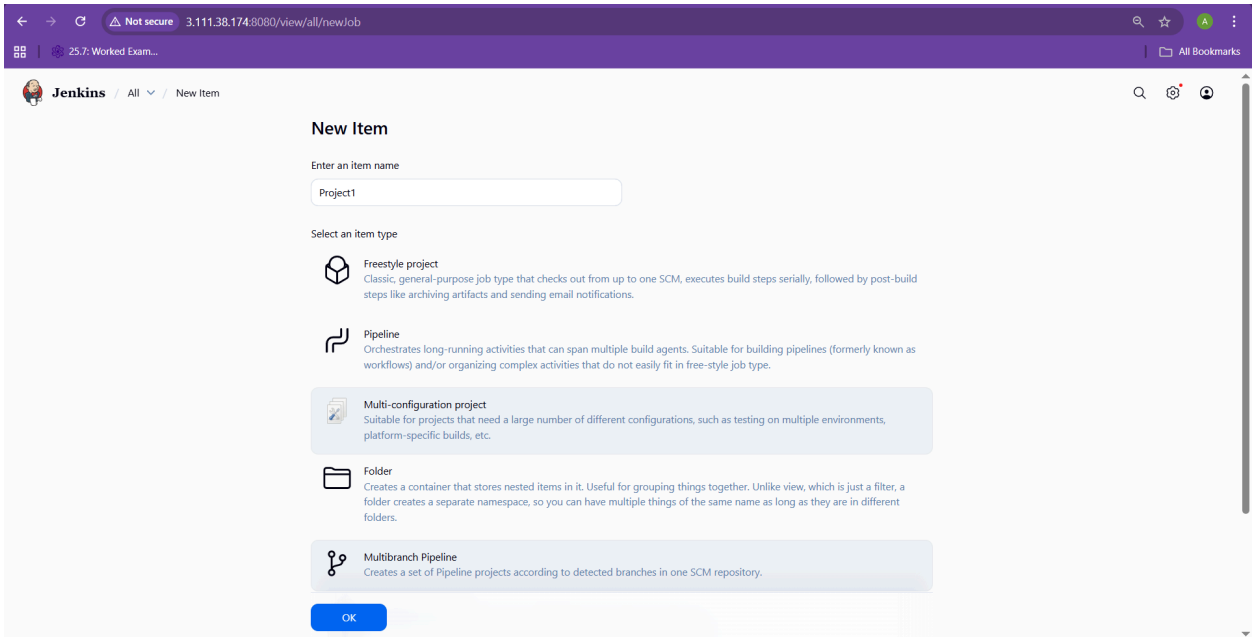
4.Installing Required plugins:



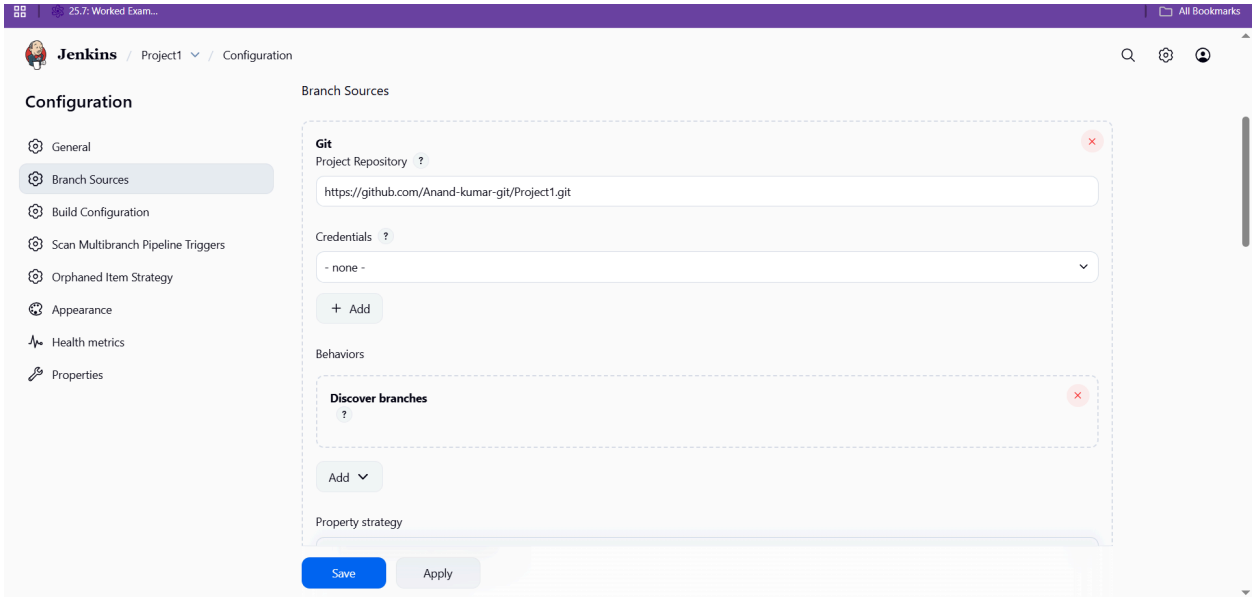
5.Adding DockerHub Credential in Jenkins:



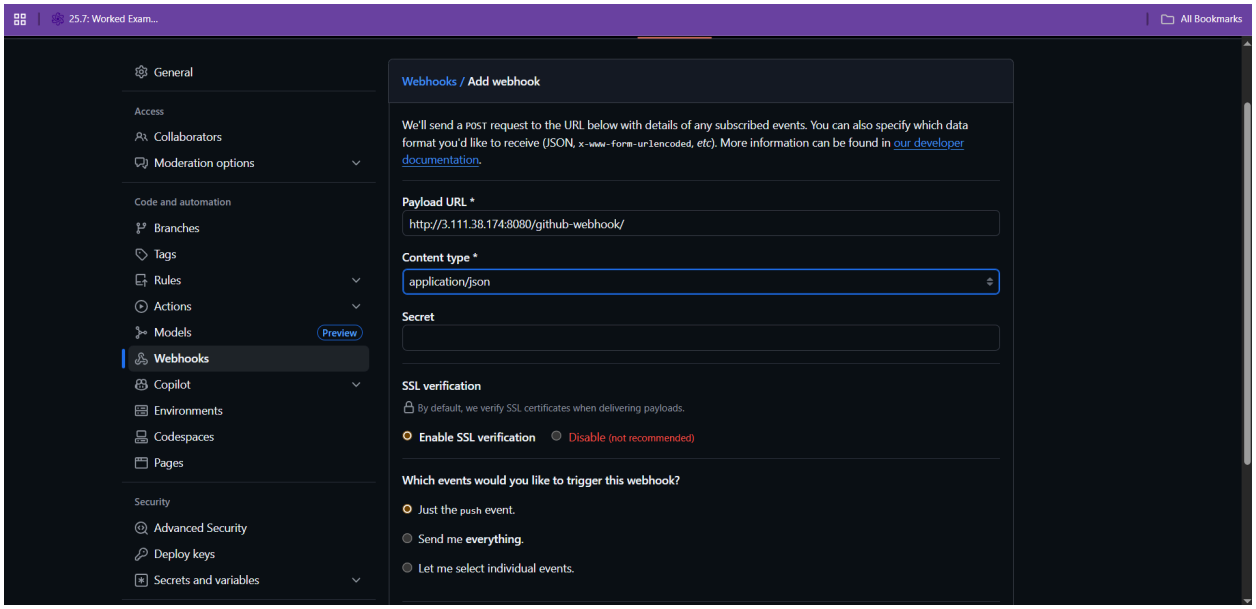
6.Creating Jenkins MultiBranch Pipeline:



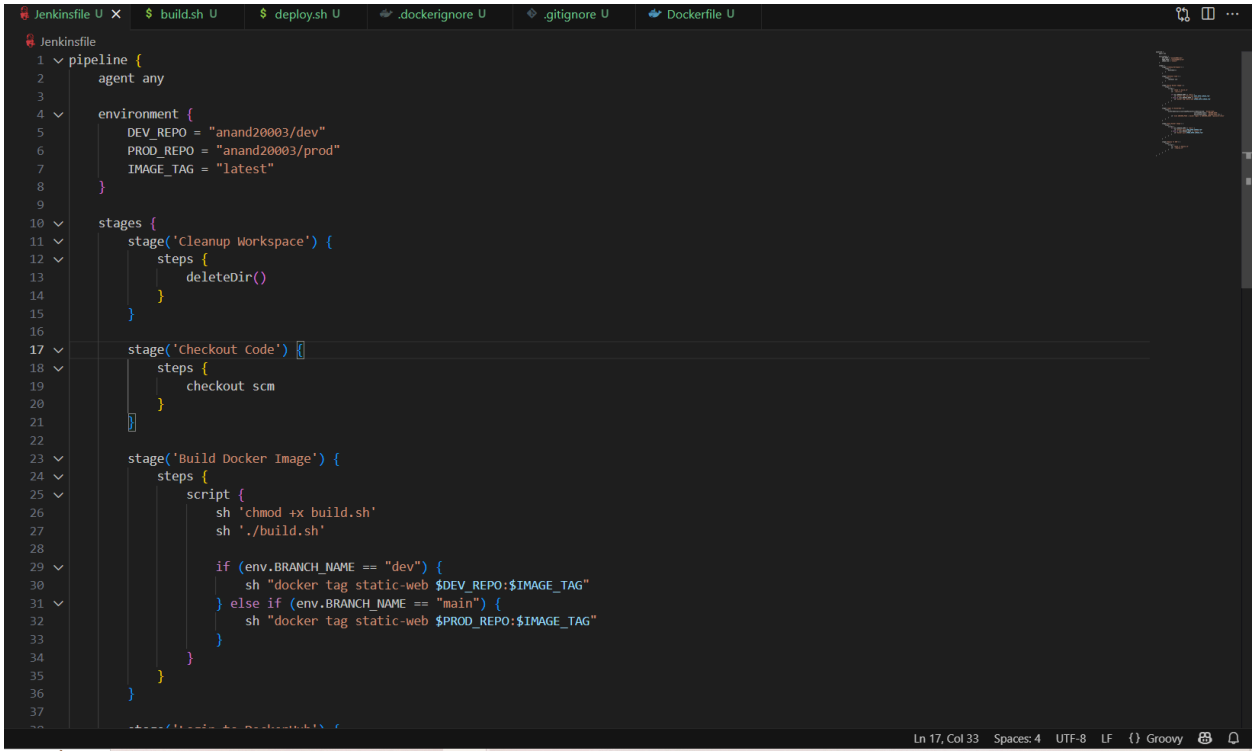
7.Multibranch Pipeline Configuration:



8.Adding Github Webhook:

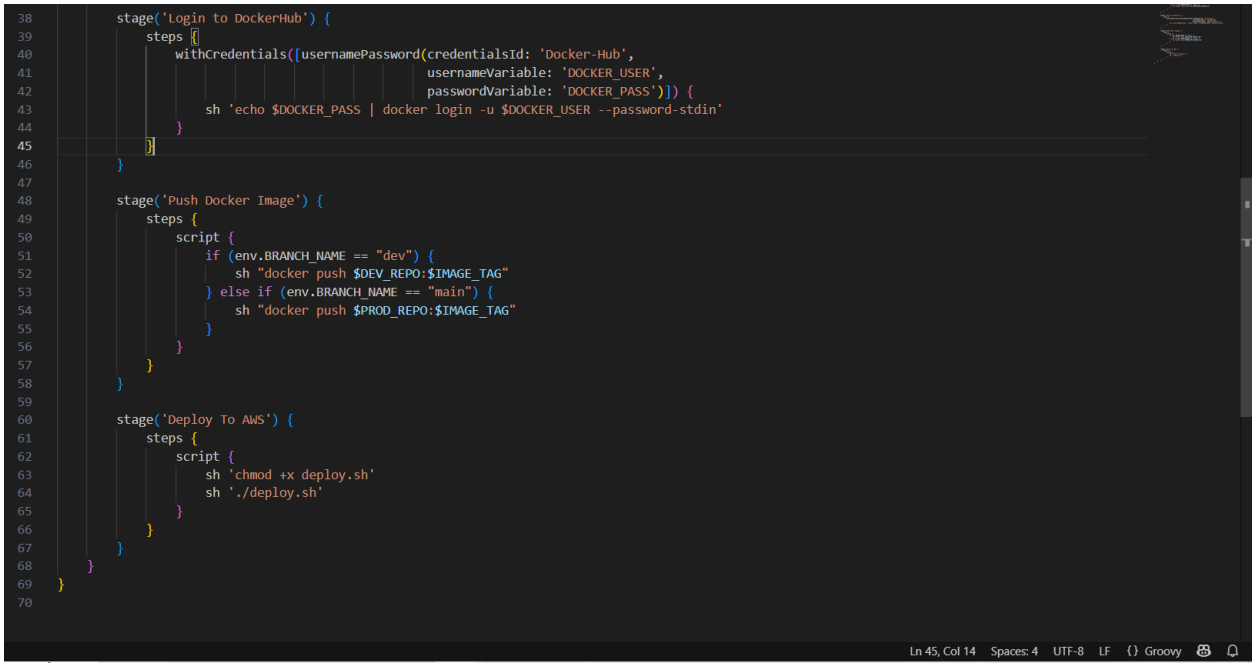


9.Jenkinsfile First part:



```
pipeline {
    agent any

    environment {
        DEV_REPO = "anand20003/dev"
        PROD_REPO = "anand20003/prod"
        IMAGE_TAG = "latest"
    }

    stages {
        stage('Cleanup Workspace') {
            steps {
                deleteDir()
            }
        }

        stage('Checkout Code') {
            steps {
                checkout scm
            }
        }

        stage('Build Docker Image') {
            steps {
                script {
                    sh 'chmod +x build.sh'
                    sh './build.sh'

                    if (env.BRANCH_NAME == "dev") {
                        sh "docker tag static-web $DEV_REPO:$IMAGE_TAG"
                    } else if (env.BRANCH_NAME == "main") {
                        sh "docker tag static-web $PROD_REPO:$IMAGE_TAG"
                    }
                }
            }
        }
```

10.Jenkinsfile Second part:



```
        stage('Login to DockerHub') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'Docker-Hub',
                                                  usernameVariable: 'DOCKER_USER',
                                                  passwordVariable: 'DOCKER_PASS')]) {
                    sh 'echo $DOCKER_PASS | docker login -u $DOCKER_USER --password-stdin'
                }
            }
        }

        stage('Push Docker Image') {
            steps {
                script {
                    if (env.BRANCH_NAME == "dev") {
                        sh "docker push $DEV_REPO:$IMAGE_TAG"
                    } else if (env.BRANCH_NAME == "main") {
                        sh "docker push $PROD_REPO:$IMAGE_TAG"
                    }
                }
            }
        }

        stage('Deploy To AWS') {
            steps {
                script {
                    sh 'chmod +x deploy.sh'
                    sh './deploy.sh'
                }
            }
        }
    }
}

```

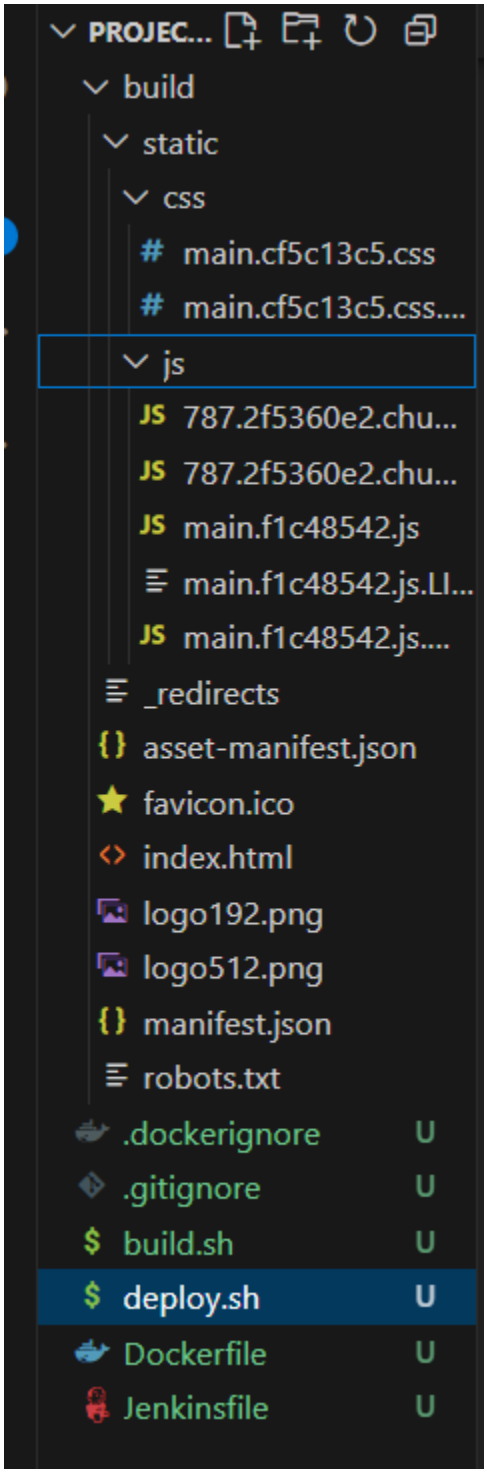
Ln 45, Col 14 Spaces: 4 UTF-8 LF {} Groovy

11.Dockerfile:

```
Dockerfile
FROM nginx:alpine

RUN rm -rf /usr/share/nginx/html/*

# Copy only prebuilt React app
COPY build/. /usr/share/nginx/html/

EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]

```

Ln 10, Col 1 Spaces: 4 UTF-8 LF {} Docker

12.File Structure:



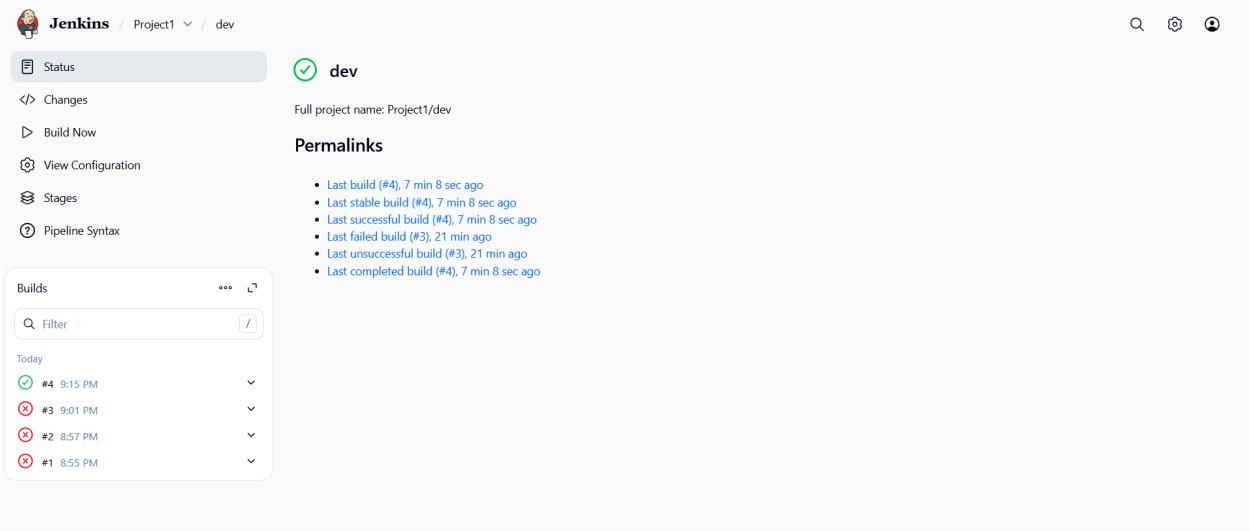
13.Adding Jenkins to Docker Group:

```
root@ip-172-31-28-7:~# usermod -aG docker jenkins
root@ip-172-31-28-7:~#
```

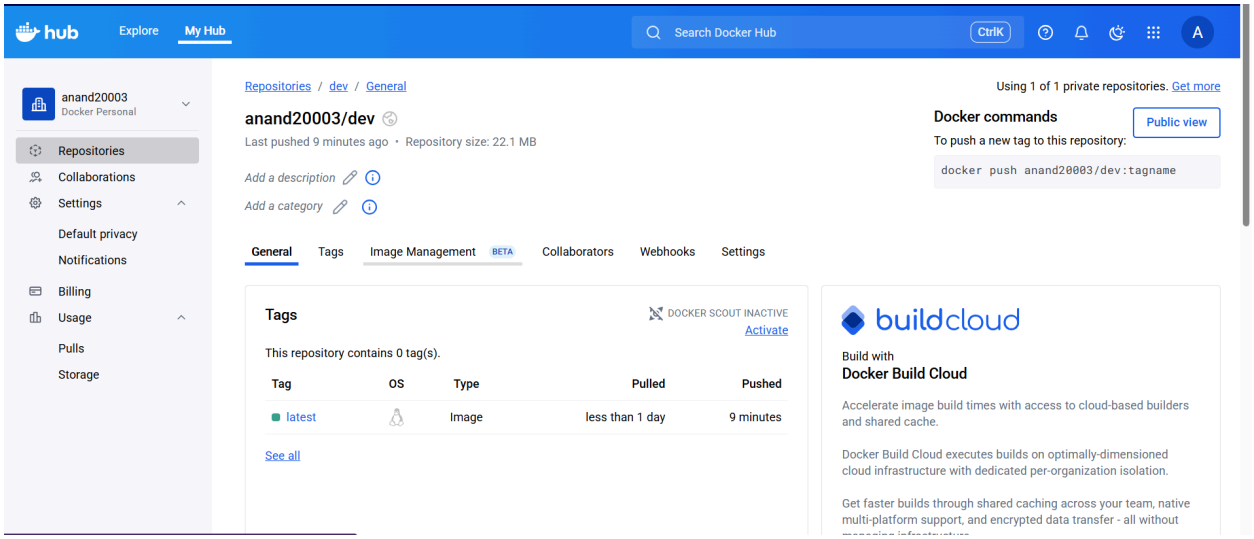
14.Pushed to dev Branch:

```
ADMIN@DESKTOP-0E234HV MINGW64 ~/Project1 (dev)
$ git commit -m "First Commit"
[dev b1a4281] First Commit
 1 file changed, 1 insertion(+), 1 deletion(-)

ADMIN@DESKTOP-0E234HV MINGW64 ~/Project1 (dev)
$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 288 bytes | 288.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/Anand-kumar-git/Project1.git
   46d7993..b1a4281  dev -> dev

ADMIN@DESKTOP-0E234HV MINGW64 ~/Project1 (dev)
$
```

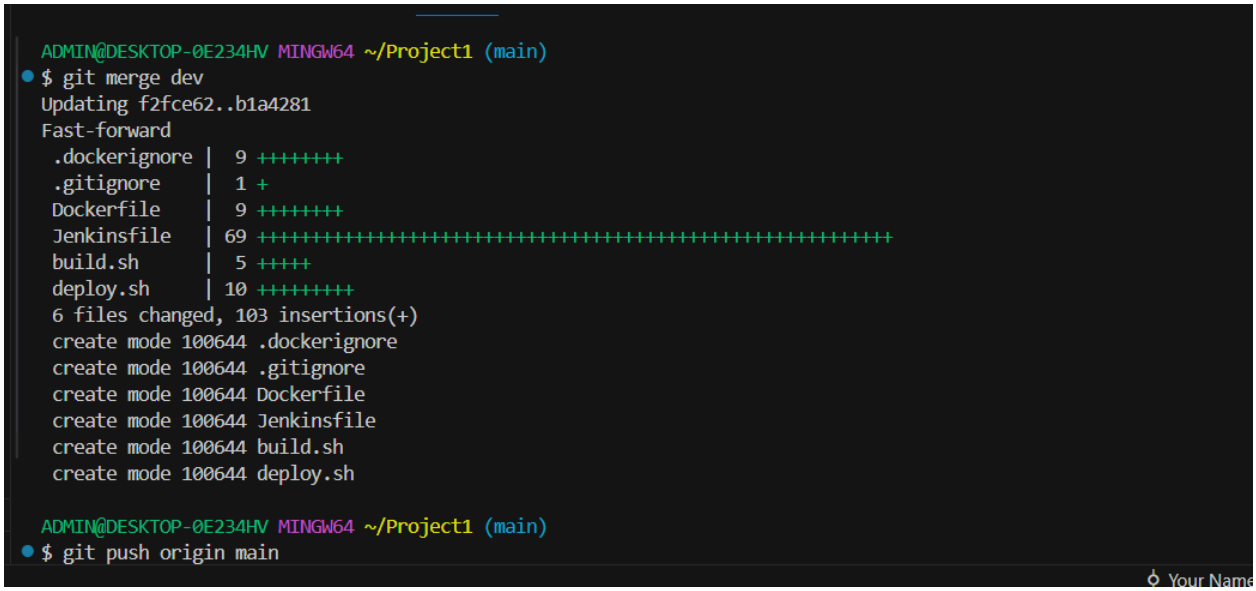
15.Pipeline Build in dev Branch:



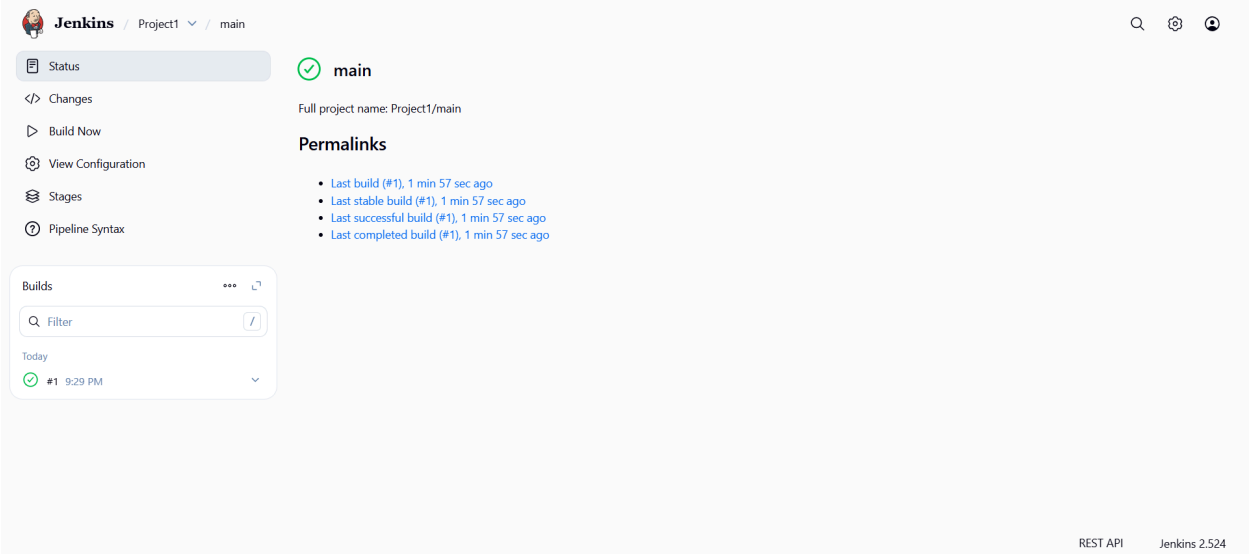
16.Docker Image Pushed to Dockerhub dev Repository:



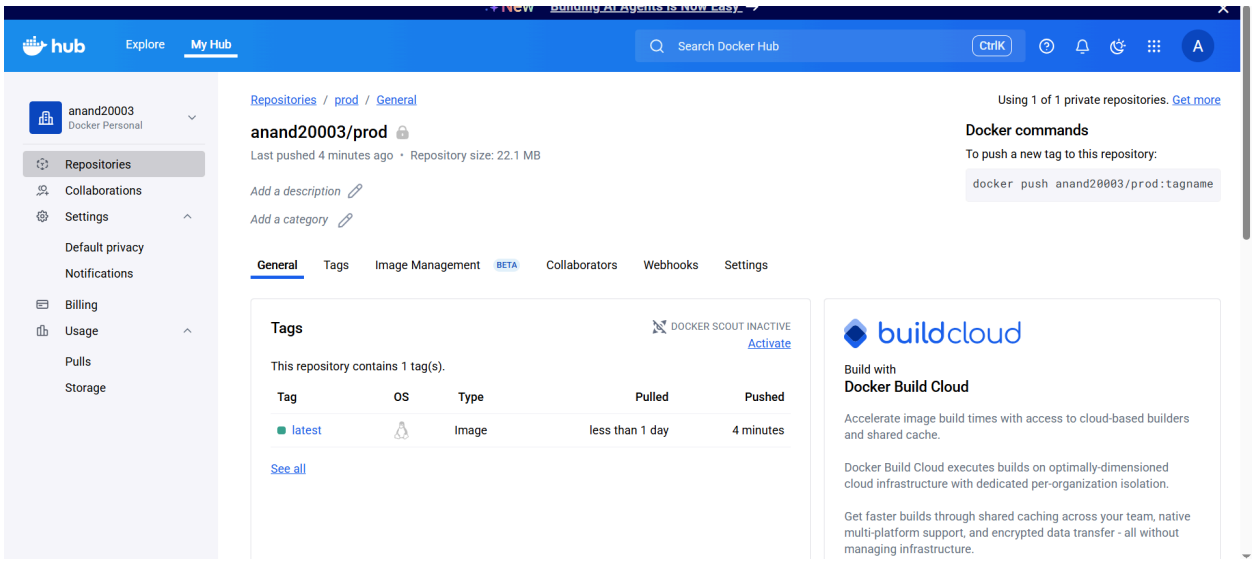
17.Merging dev to main:



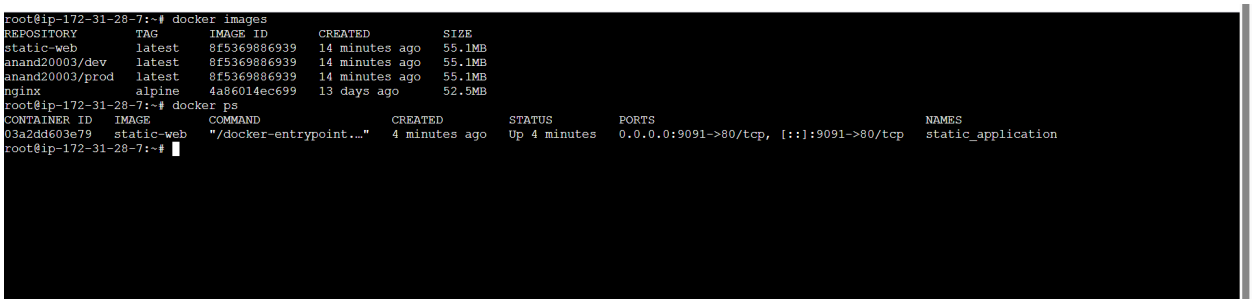
18.Pipeline Build in Main branch:



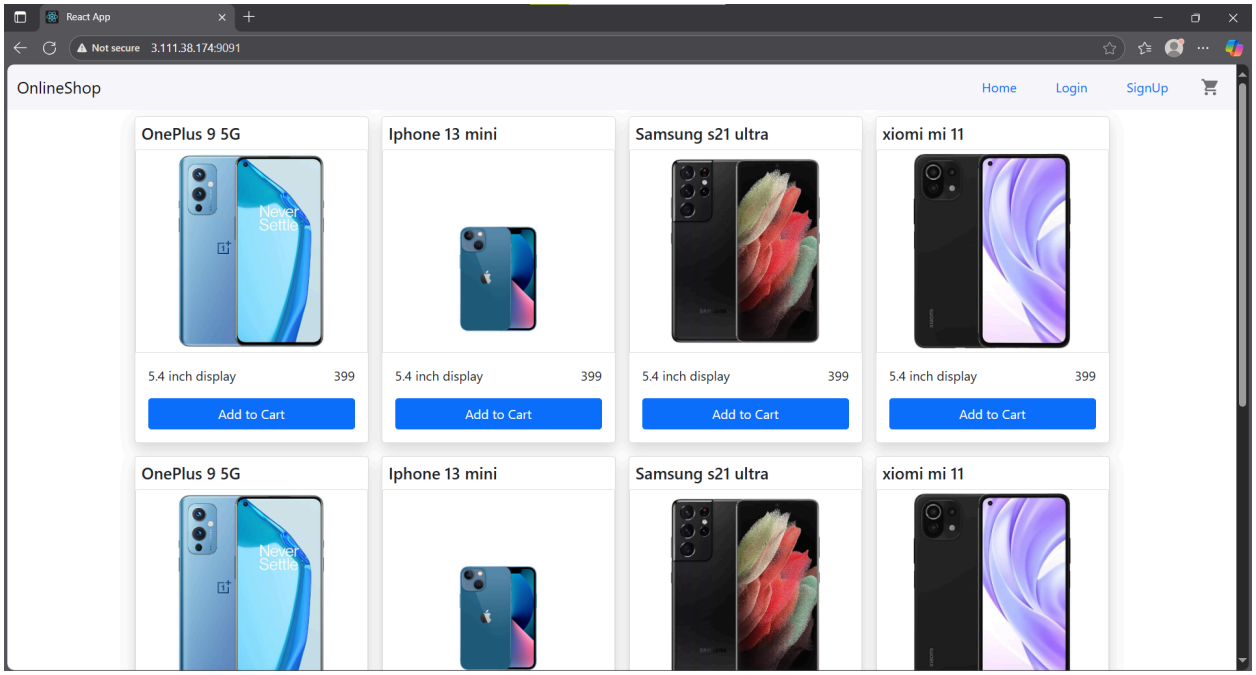
19.Docker Image Pushed to DockerHub prod Private Repository:



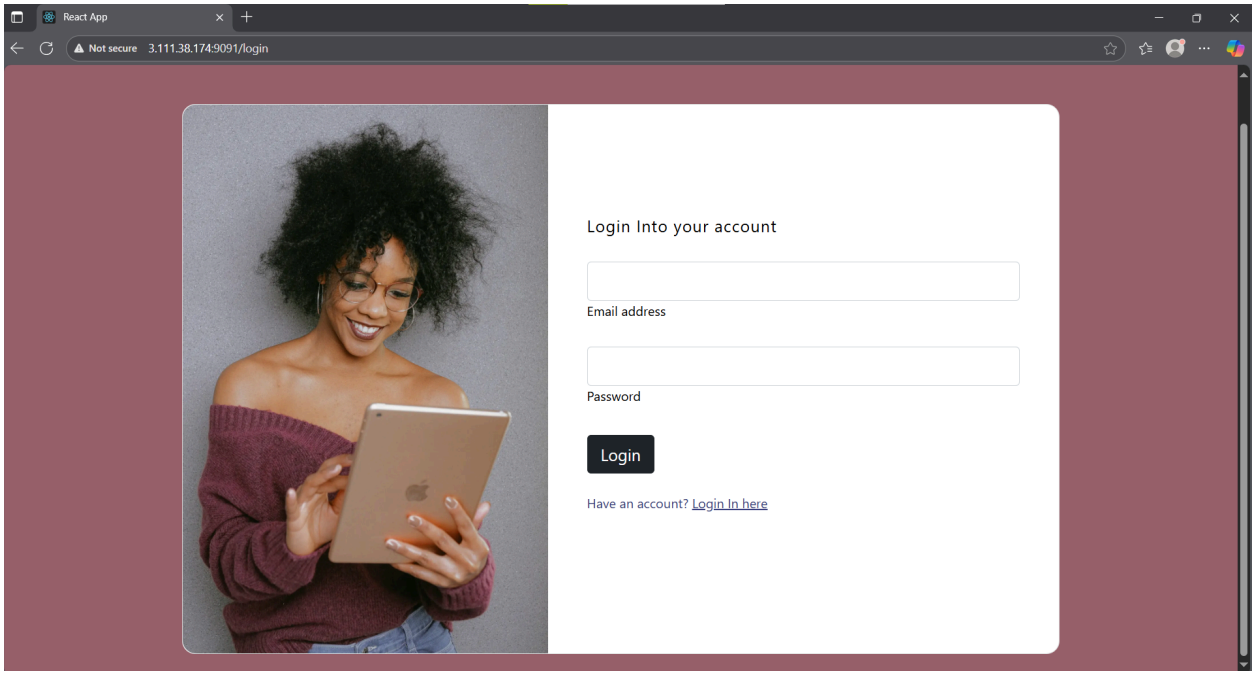
20.Docker Image and Container Build Locally:



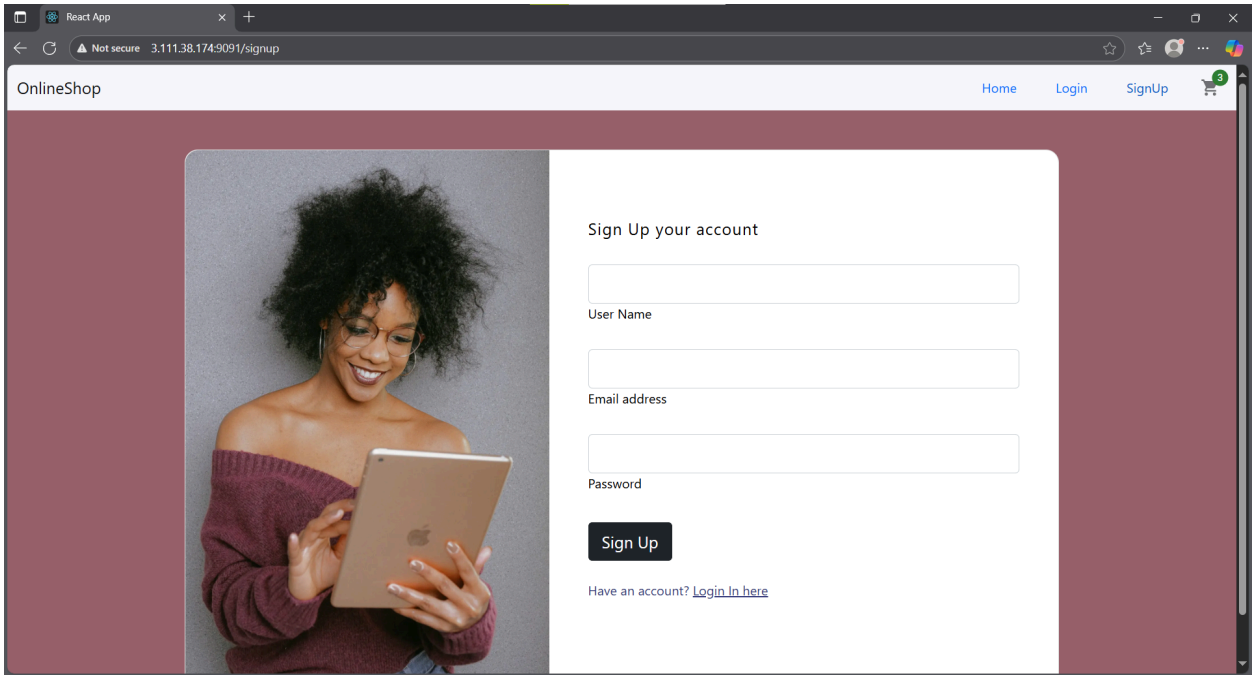
21.Output 1:



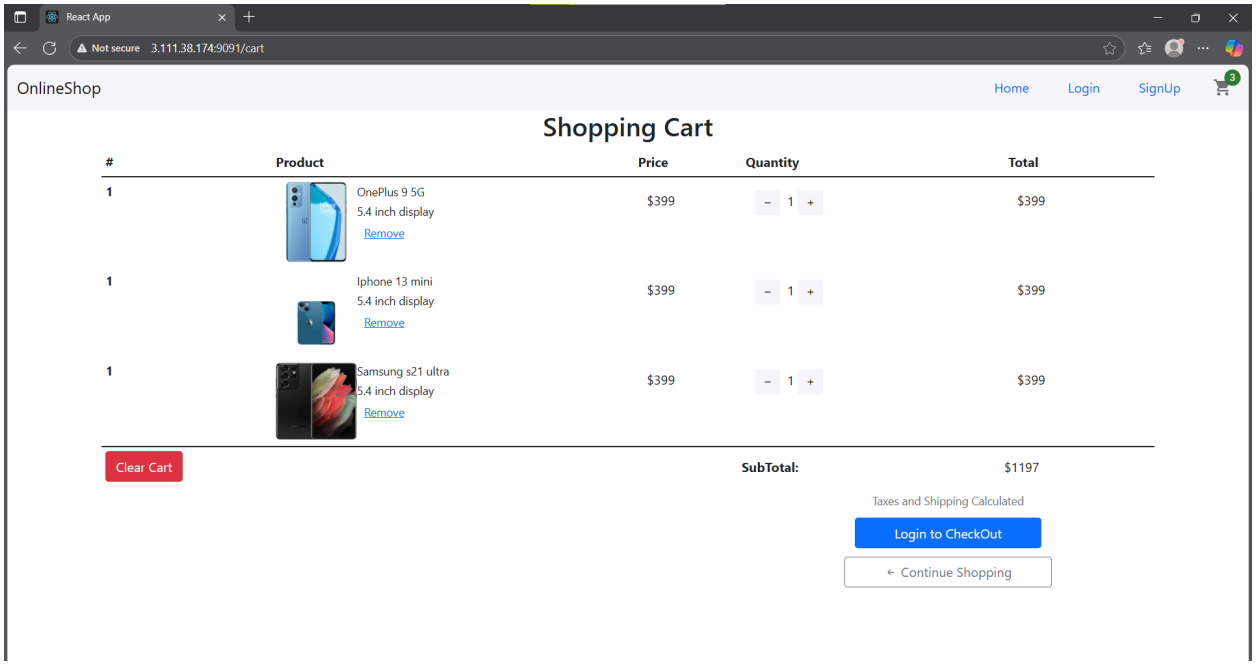
22.Output 2:



23.Output 3:

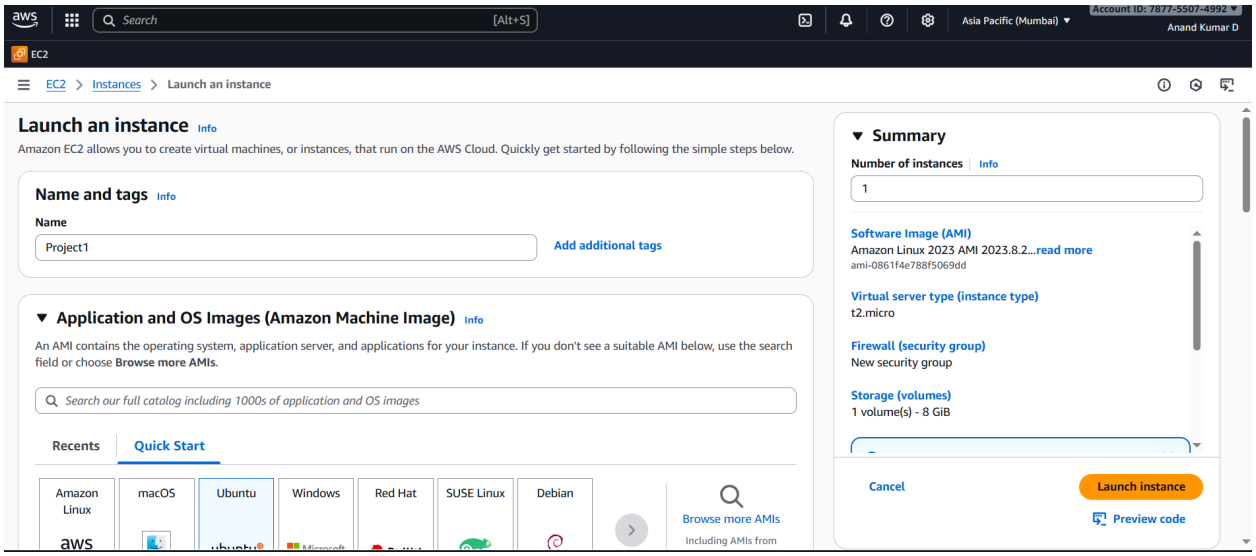


24.Output 4:

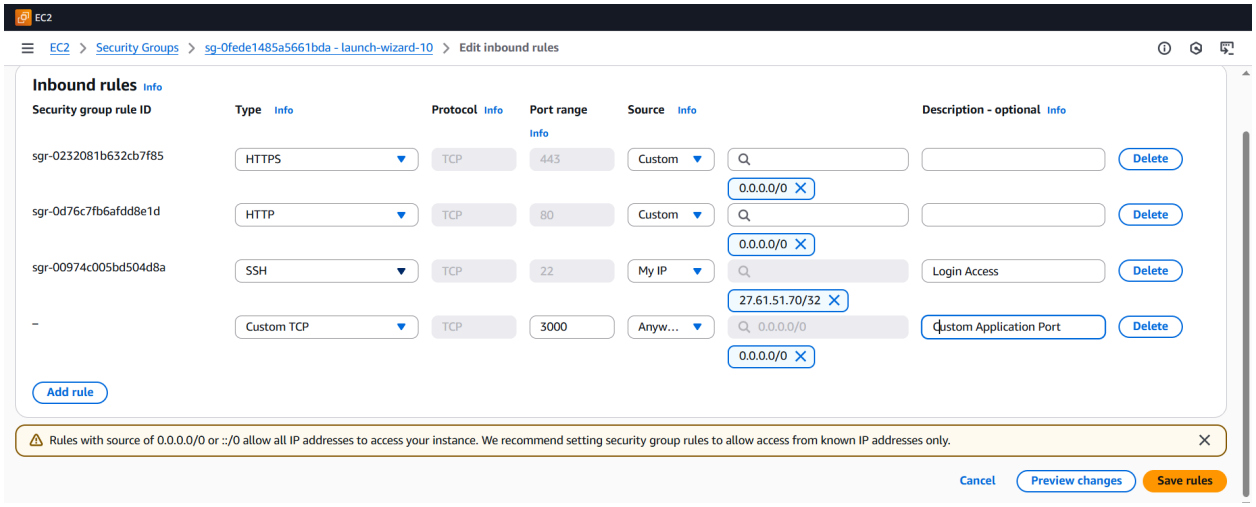


6.AWS:

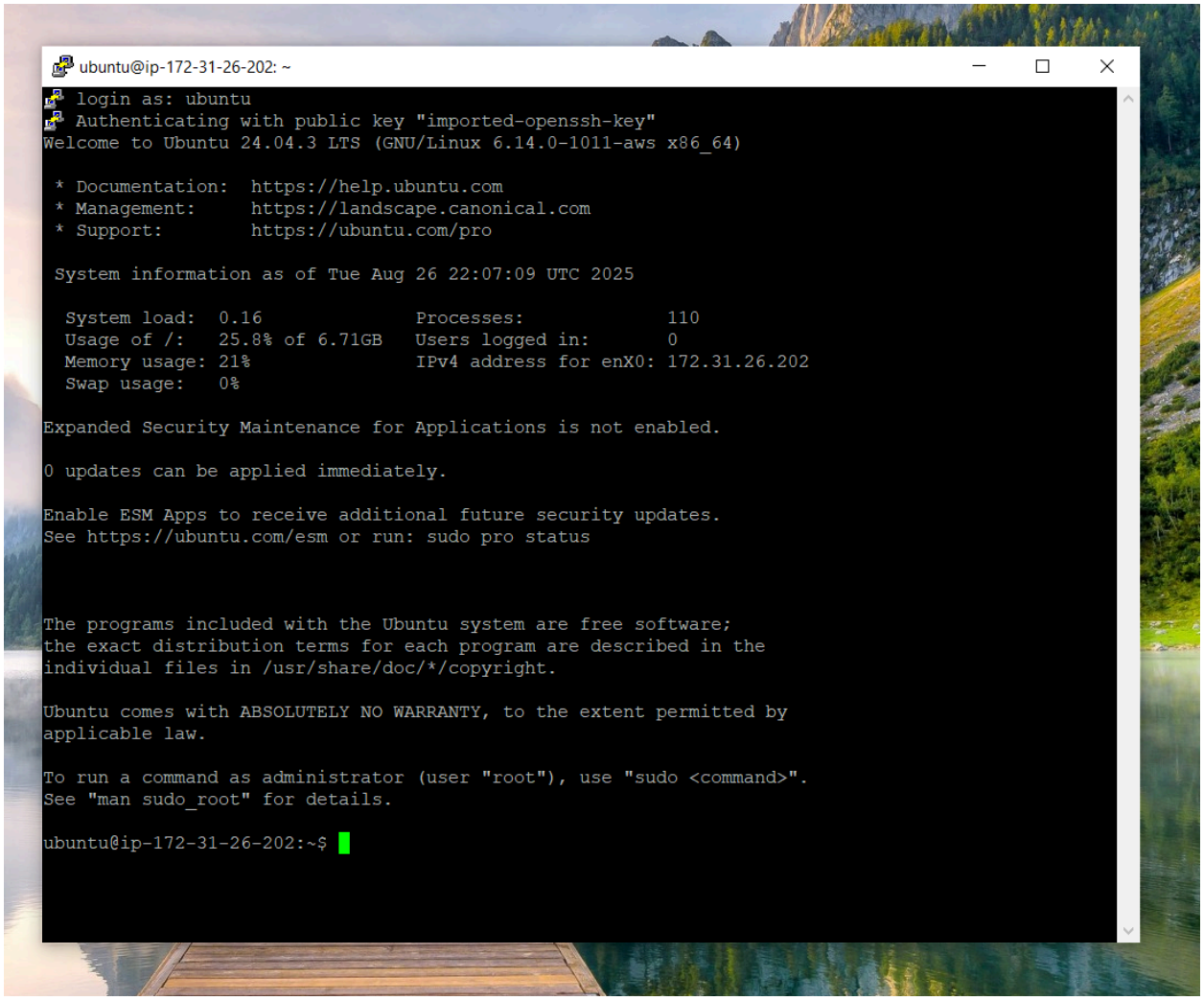
1.Launching EC2 Instance with t2.micro:



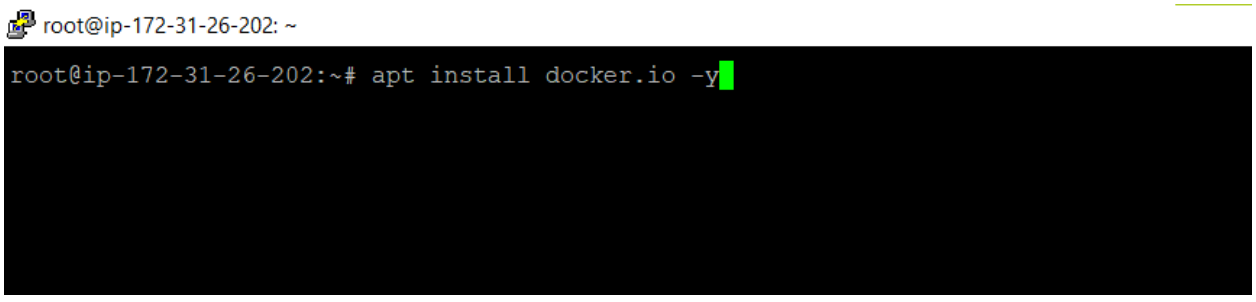
2. Security Group Configurations:



3.Login into the server with my IP:



4.Installing Docker:



5.Docker Installed:

root@ip-172-31-26-202: ~

```
root@ip-172-31-26-202:~# docker --version
Docker version 27.5.1, build 27.5.1-0ubuntu3~24.04.2
root@ip-172-31-26-202:~#
```

6.DockerHub Login Done:

```
root@ip-172-31-26-202:~# docker login

USING WEB-BASED LOGIN
To sign in with credentials on the command line, use 'docker login -u <username>'

Your one-time device confirmation code is: CGLG-JJKQ
Press ENTER to open your browser or submit your device code here: https://login.docker.com/activate

Waiting for authentication in the browser…
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credential-stores

Login Succeeded
root@ip-172-31-26-202:~#
```

7.Pulling Docker Image From Dockerhub:

root@ip-172-31-26-202: ~

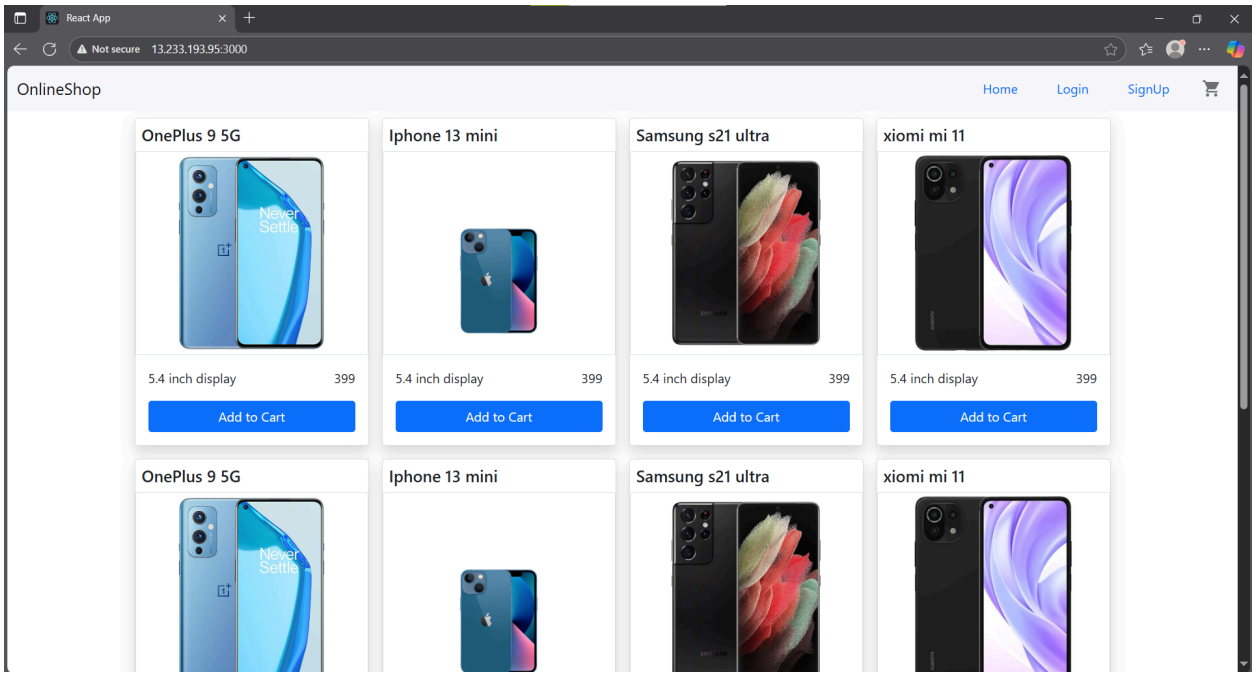
```
root@ip-172-31-26-202:~# docker pull anand20003/prod:latest
latest: Pulling from anand20003/prod
9824c27679d3: Pull complete
6bc572a340ec: Pull complete
403e3f251637: Pull complete
9adfbae99cb7: Pull complete
7a8a46741e18: Pull complete
c9ebe2ff2d2c: Pull complete
a992fbc61ecc: Pull complete
cb1ff4086f82: Pull complete
7252e8e57291: Pull complete
8e6b521f19db: Pull complete
Digest: sha256:f11b94cb4989c2d6062c45385ab49c49163fdadfe2752a3ad7ac1df6995f7c64
Status: Downloaded newer image for anand20003/prod:latest
docker.io/anand20003/prod:latest
root@ip-172-31-26-202:~#
```

8.Container Created With Pulled Image:

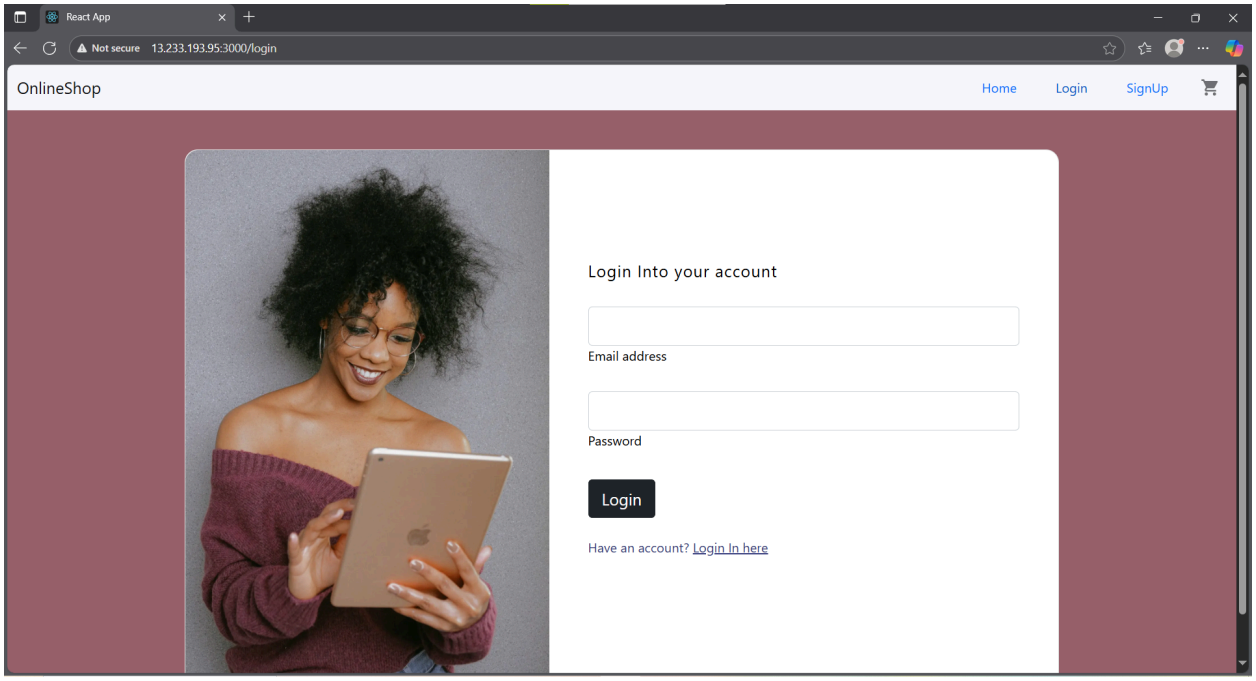
root@ip-172-31-26-202: ~

```
root@ip-172-31-26-202:~# docker images
REPOSITORY        TAG       IMAGE ID       CREATED             SIZE
anand20003/prod   latest    8f5369886939   About an hour ago   55.1MB
root@ip-172-31-26-202:~# docker run -itd -p "3000:80" --name Reactjs_Application anand20003/prod
44455847f161ad69616edcb1ff9dd1cb03783418d542415e56314fc89d027d32
root@ip-172-31-26-202:~#
```

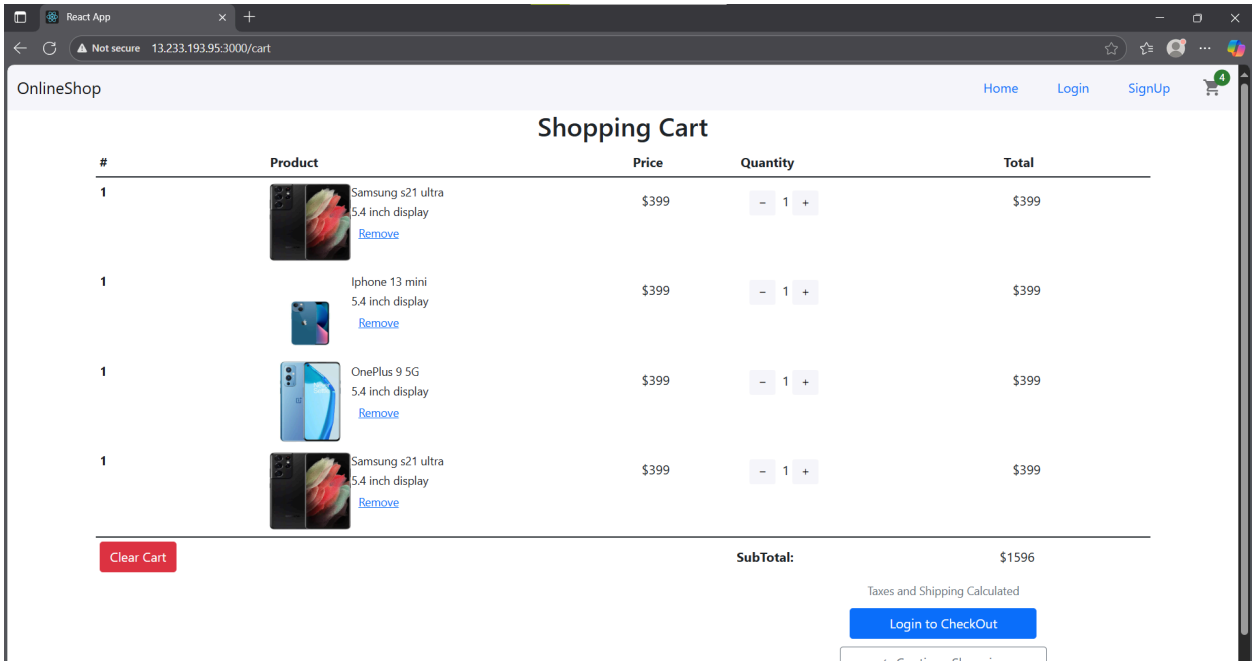
9.Output 1:



10.Output 2:

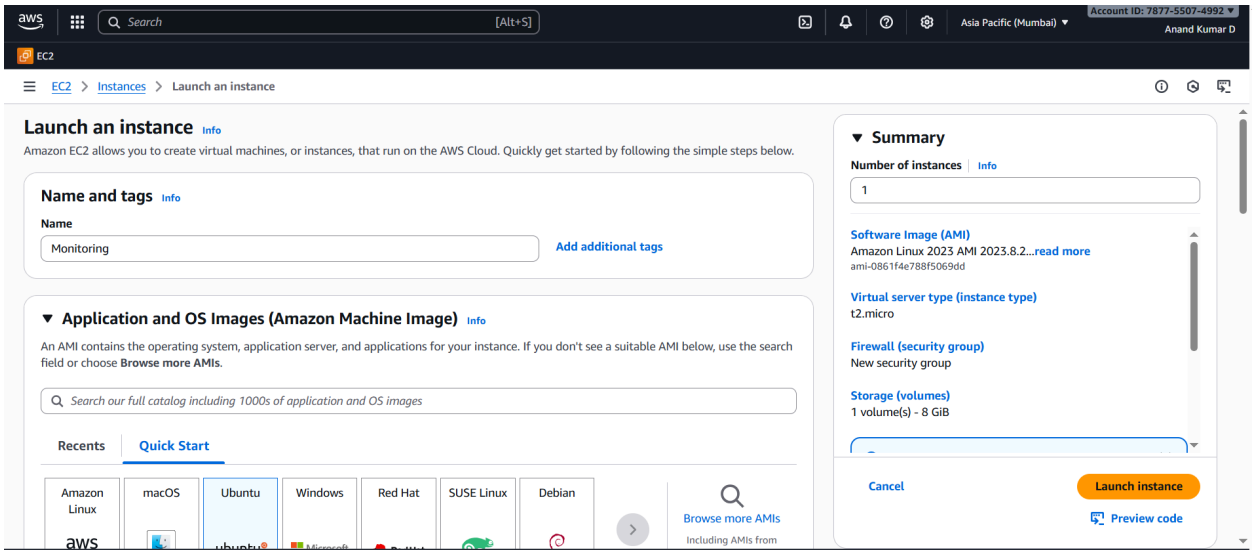


11.Output 3:



7.Monitoring:

1.Launching Instance For Monitoring:



2.Installing Docker-compose:

```
root@ip-172-31-23-1:~# apt install docker-compose
```

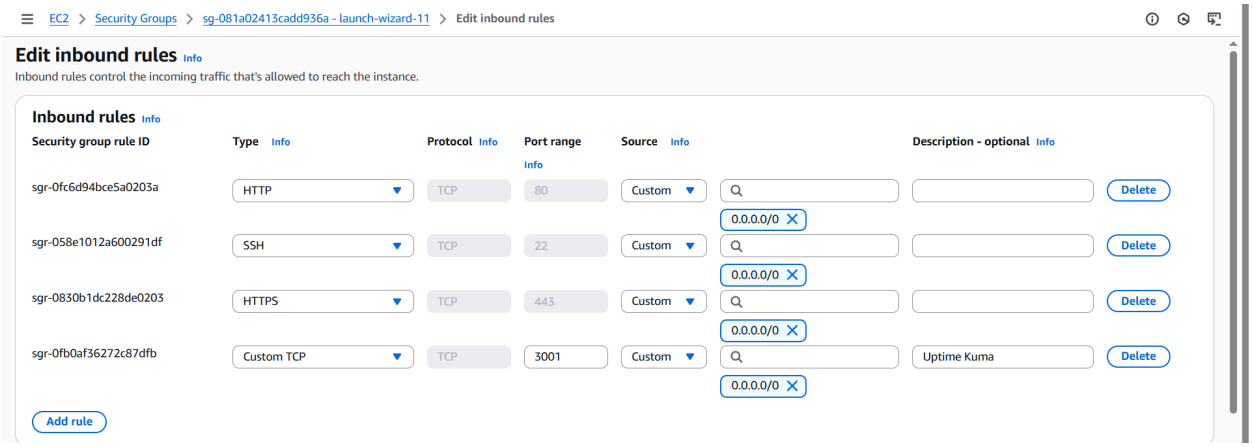
3.Docker-compose.yaml file for Uptime Kuma:

```
  GNU nano 7.2                                          docker-compose.yml
version: '3'
services:
  uptime-kuma:
    image: louislam/uptime-kuma:1
    container_name: uptime-kuma
    restart: always
    ports:
      - "3001:3001"
    volumes:
      - ./data:/app/data
```

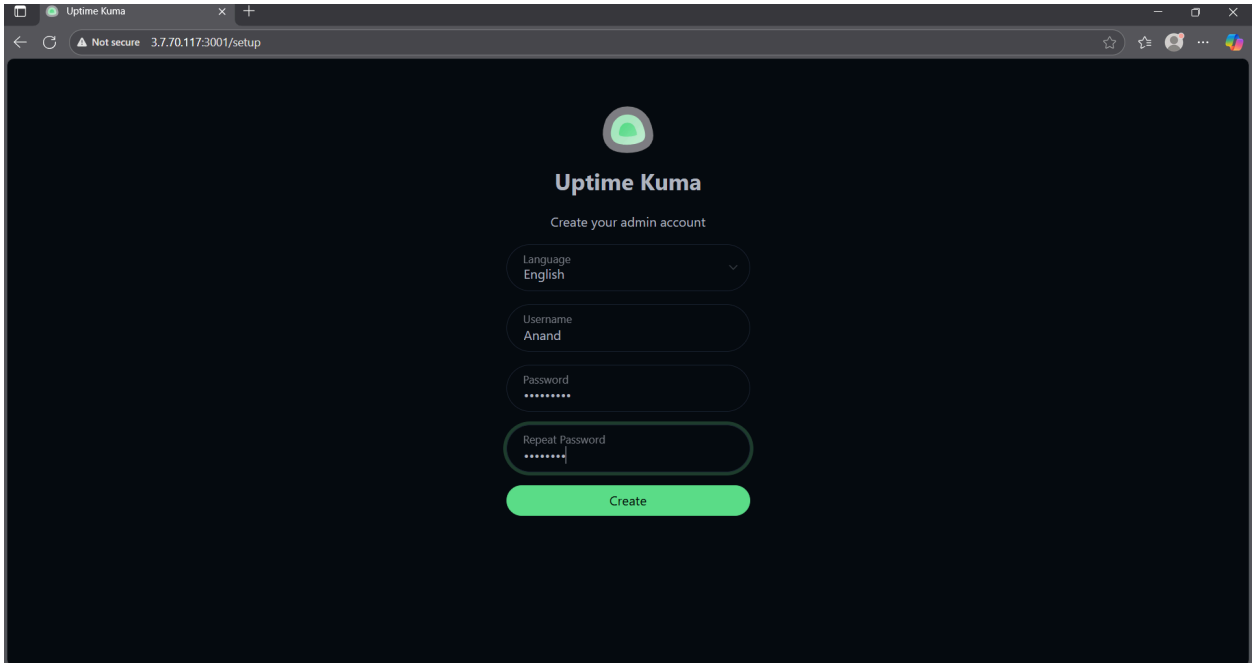
4.Running Docker-compose File:

```
root@ip-172-31-23-1:~# docker-compose up -d
Creating network "root_default" with the default driver
Pulling uptime-kuma (louislam/uptime-kuma:1)...
1: Pulling from louislam/uptime-kuma
b338562f40a7: Pull complete
874bf4d93720: Pull complete
b16337721583: Pull complete
7d955db85b85: Pull complete
2c706596bd17: Pull complete
88a5c59ed14f: Pull complete
5a1d0a896c33: Pull complete
e68c2f25b946: Pull complete
2e6c90f010d6: Pull complete
ff15b10fabb8: Pull complete
4f4fb700ef54: Pull complete
d2a400cc8adb: Pull complete
Digest: sha256:431fee3be822b04861cf0e35daf4beef6b7cb37391c5f26c3ad6e12ce280fe18
Status: Downloaded newer image for louislam/uptime-kuma:1
Creating uptime-kuma ... done
root@ip-172-31-23-1:~#
```

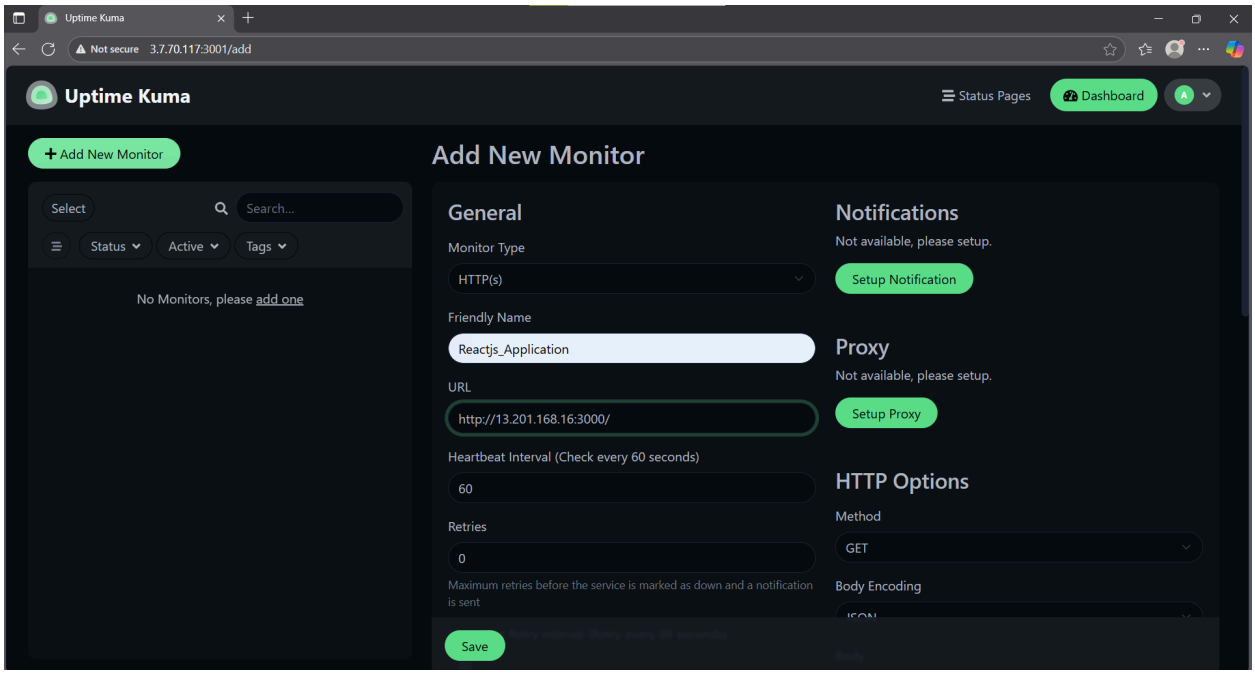
5.Enabling Port 3001 to access Uptime-Kuma for Monitoring:



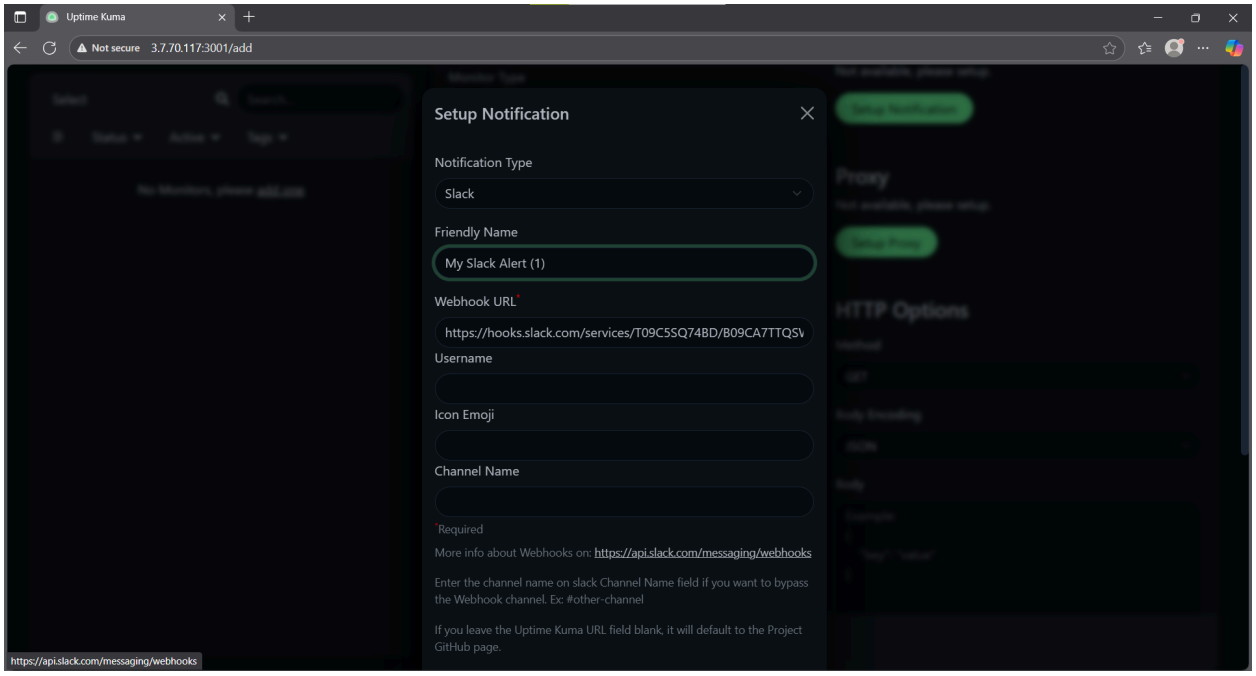
6.Creating Account in Uptime-Kuma for Monitoring:



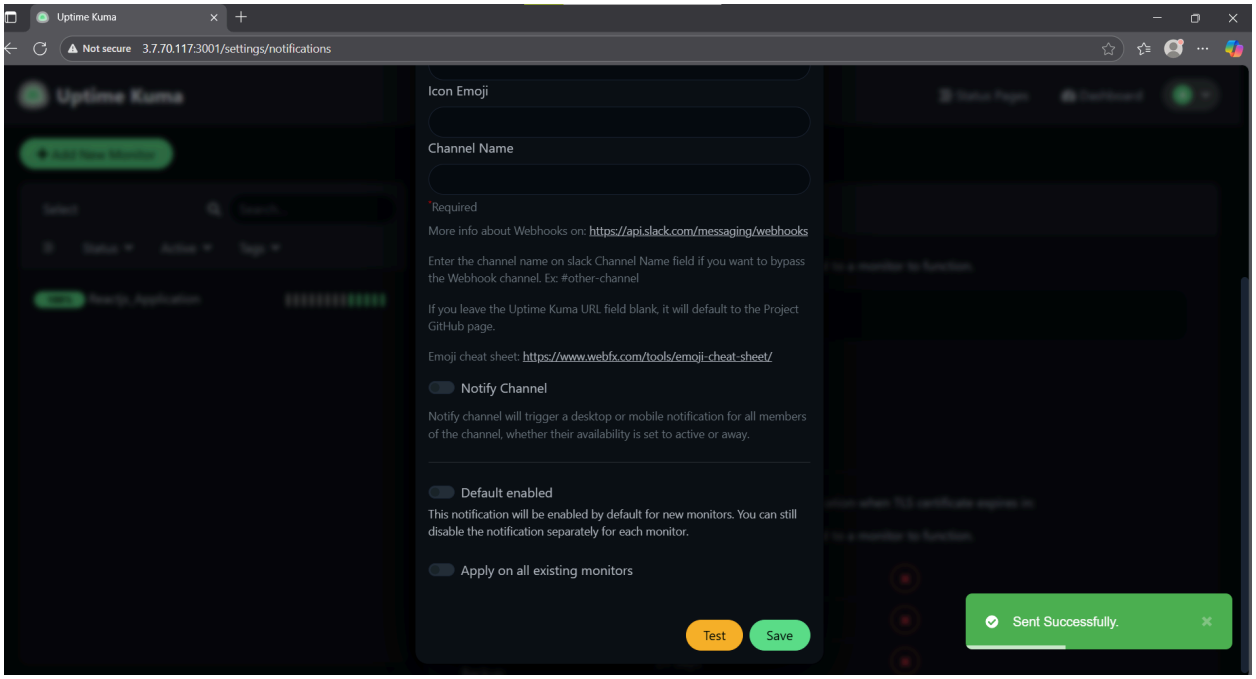
7.Adding New Monitor:



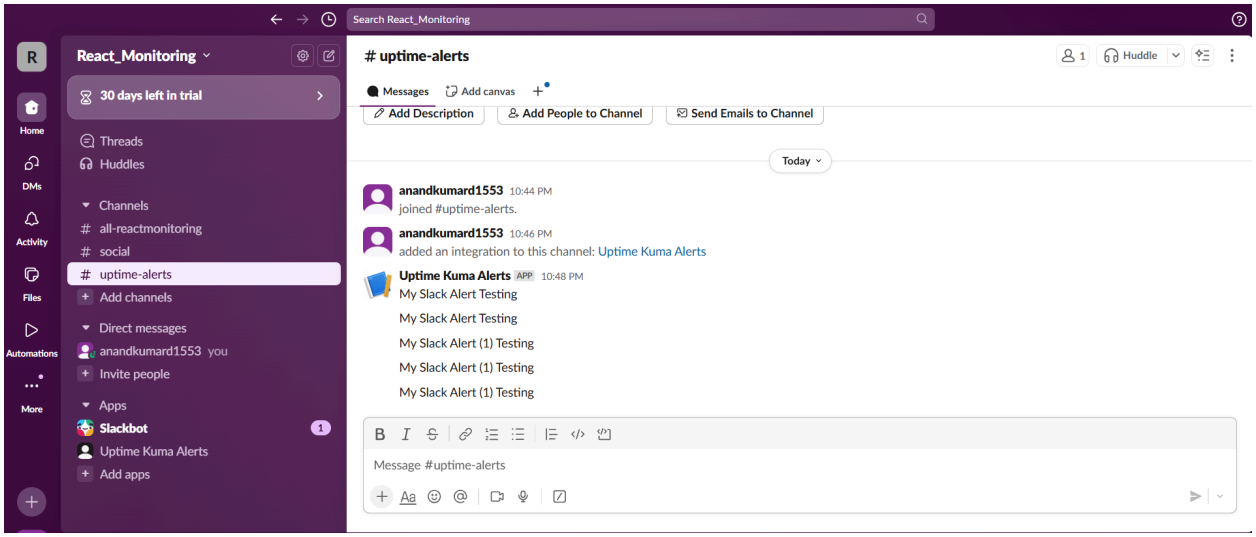
8.Setting up Slack Notification:



9.Testing the Notification:



10.Notification Received in Slack:



11.Monitor Dashboard in Uptime Kuma: