# Project Title: Web Application Update & Deployment- Mini Capstone 1

**Project Tasks:**

## 1. Version Control & Code Management:

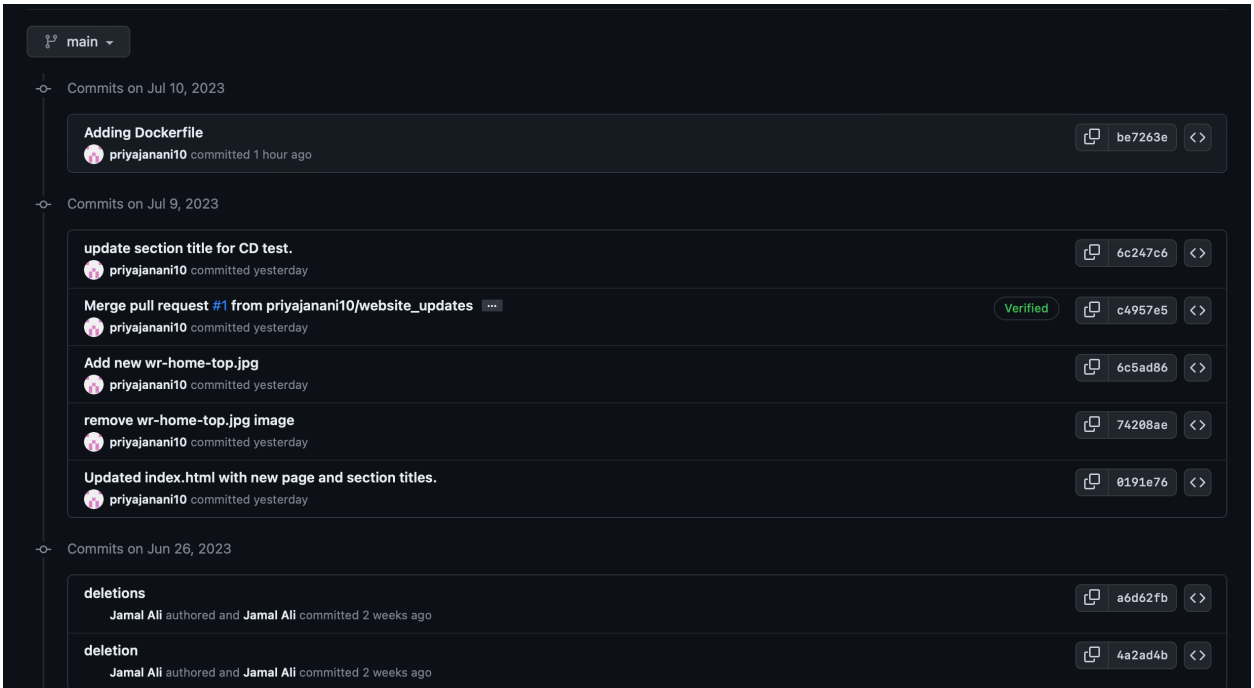
- I created my GitHub account and created a new repository.
- I used git clone https://github.com/babaJay23/capstone-aws.git command to clone the project codebase on my laptop.
- I removed the origin, set my repository as origin and pushed the code to the main branch on my repository. Commands are listed below.

```
git clone https://github.com/babaJay23/capstone-aws.git
git remote remove origin
git remote -v
git remote add origin https://github.com/priyajanani10/capstone-aws.git
git remote -v
git push
git push --set-upstream origin main
git config --global user.email "priyajanani10@gmail.com"
git config --global user.name "Priya Janani Sreenivasan"
git status
git checkout -b website_updates
```

## 2. Web Application Update:

Using three separate commits on branch website_updates I,

- Updated the page and section titles.
- Deleted the image wr-home-top.jpg
- Added a new unicorn image wr-home-top.jpg
- Pushed the branch website updates to my repository and created a pull request to merge the commits into main. I merged the commits using a merge commit.



### 3. Containerization:

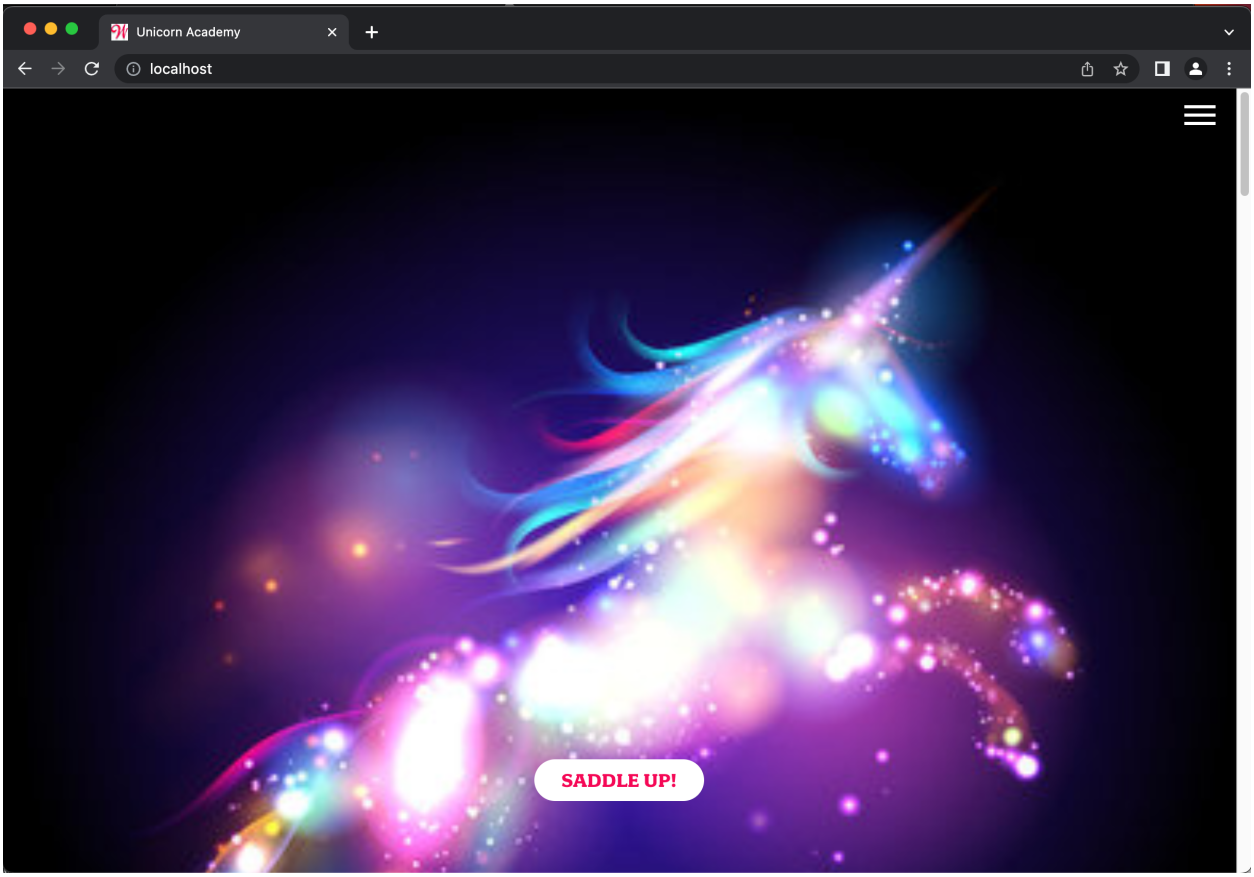
- I created a Dockerfile.

```
1    FROM nginx
2    COPY . /usr/share/nginx/html
3    EXPOSE 80
4    CMD ["nginx", "-g", "daemon off;"]
```

- I committed the Docker file to my git repository.
- I built and tested my docker image locally.

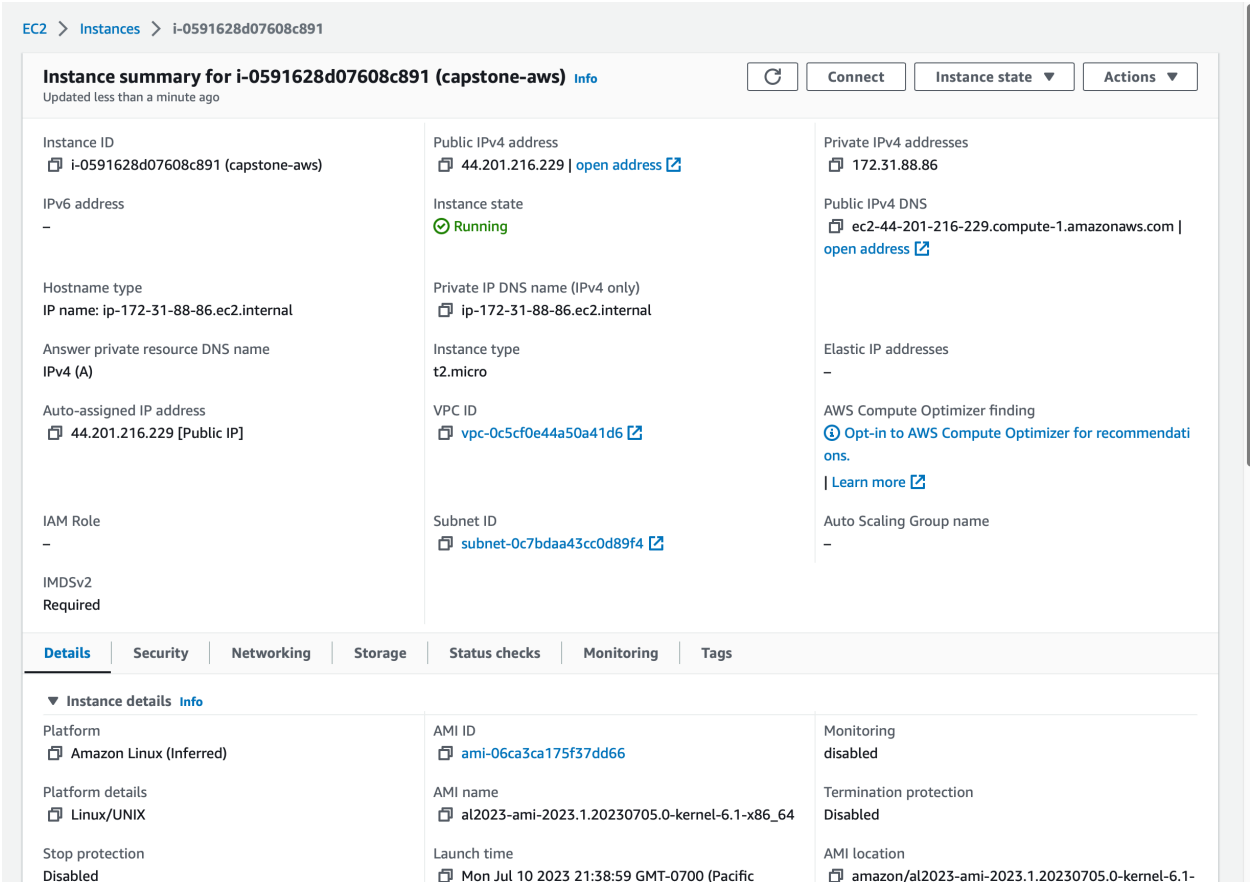
```
janani@Priyas-Air capstone-aws % docker build -t capstone-aws .
[+] Building 1.9s (8/8) FINISHED                                                                                    docker:desktop-linux
 => [internal] load build definition from Dockerfile                                                                                0.0s
 => => transferring dockerfile: 159B                                                                                                0.0s
 => [internal] load .dockerignore                                                                                                   0.0s
 => => transferring context: 2B                                                                                                     0.0s
 => [internal] load metadata for docker.io/library/nginx:latest                                                                     1.6s
 => [auth] library/nginx:pull token for registry-1.docker.io                                                                        0.0s
 => [internal] load build context                                                                                                   0.0s
 => => transferring context: 25.30kB                                                                                                0.0s
 => CACHED [1/2] FROM docker.io/library/nginx@sha256:08bc36ad52474e528cc1ea3426b5e3f4bad8a130318e3140d6cfe29c8892c7ef               0.0s
 => [2/2] COPY . /usr/share/nginx/html                                                                                              0.2s
 => exporting to image                                                                                                              0.1s
 => => exporting layers                                                                                                             0.1s
 => => writing image sha256:3085b7e624e7e84898f1767d23bb4f7a675060c9052dcb9ee969fff08a862b8f                                        0.0s
 => => naming to docker.io/library/capstone-aws                                                                                     0.0s

What's Next?
  View summary of image vulnerabilities and recommendations → docker scout quickview
janani@Priyas-Air capstone-aws % docker run -dp 80:80 capstone-aws
b618266226b8c91ca3090ac14d43905b249f7eb7aa78ec248aa51bdc8b22b736
```

- I checked to see if my container is running using localhost:80 and my updated website showed up.




**4. Cloud Deployment:**

- I launched a new EC2 instance, created a new key pair and selected the check box to allow anyone from the internet to access port 80.




- I connected to the EC2 instance using ssh, I changed the user permissions using the following commands

    chmod 400 vockey2-5.pem
    ssh -i "vockey2-5.pem" euser@ec2-44-201-216-229.compute-1.amazonaws.com

```
janani@Priyas-Air Desktop % ssh -i "vockey2-5.pem" ec2-user@ec2-44-201-216-229.compute-1.amazonaws.com
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Tue Jul 11 04:42:33 2023 from 98.33.83.15
```

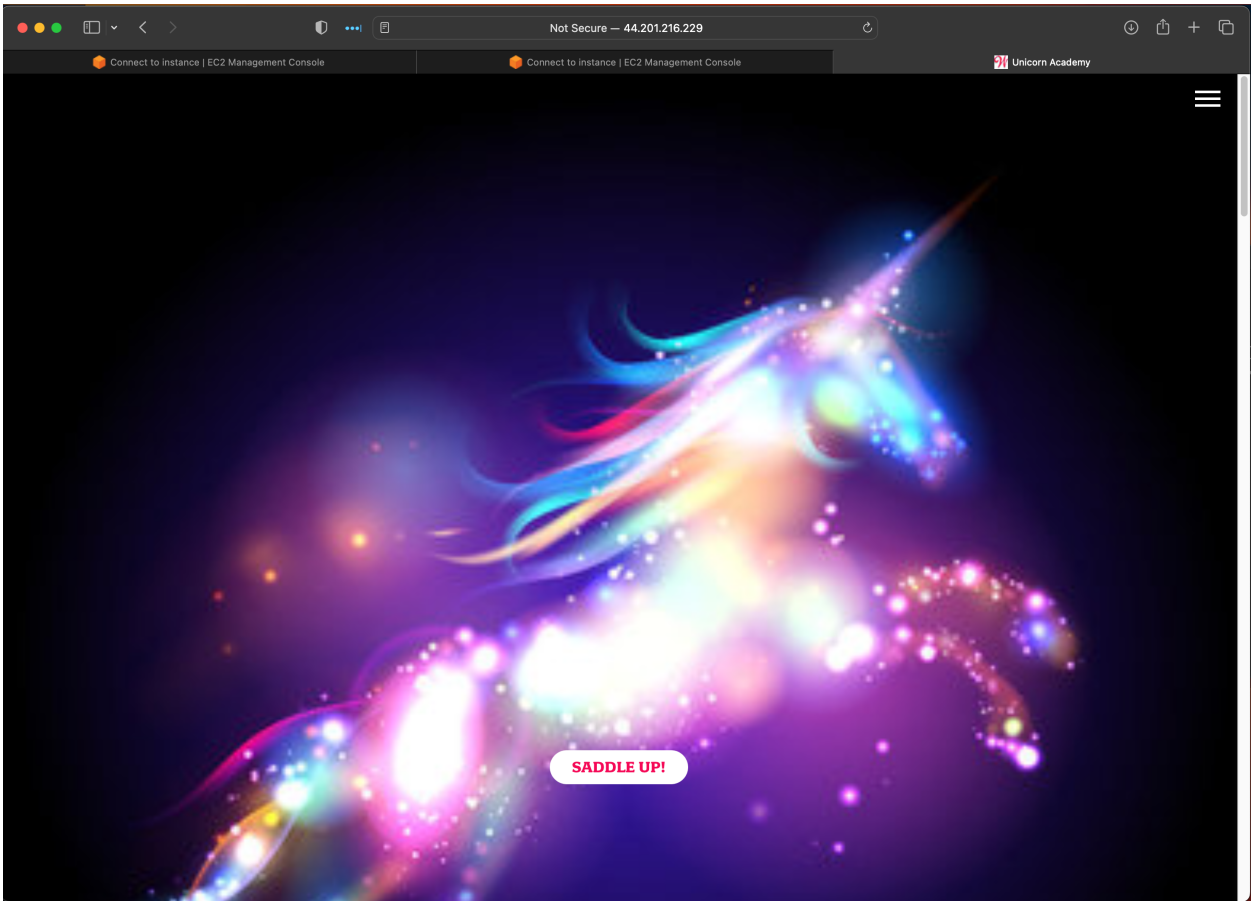
- I installed git, docker and started the docker service. The commands I used are,

    sudo yum install git -y
    sudo yum install docker -y
    sudo service  docker start
    sudo usermod -a -G docker ec2-user

- I built and ran my docker image.

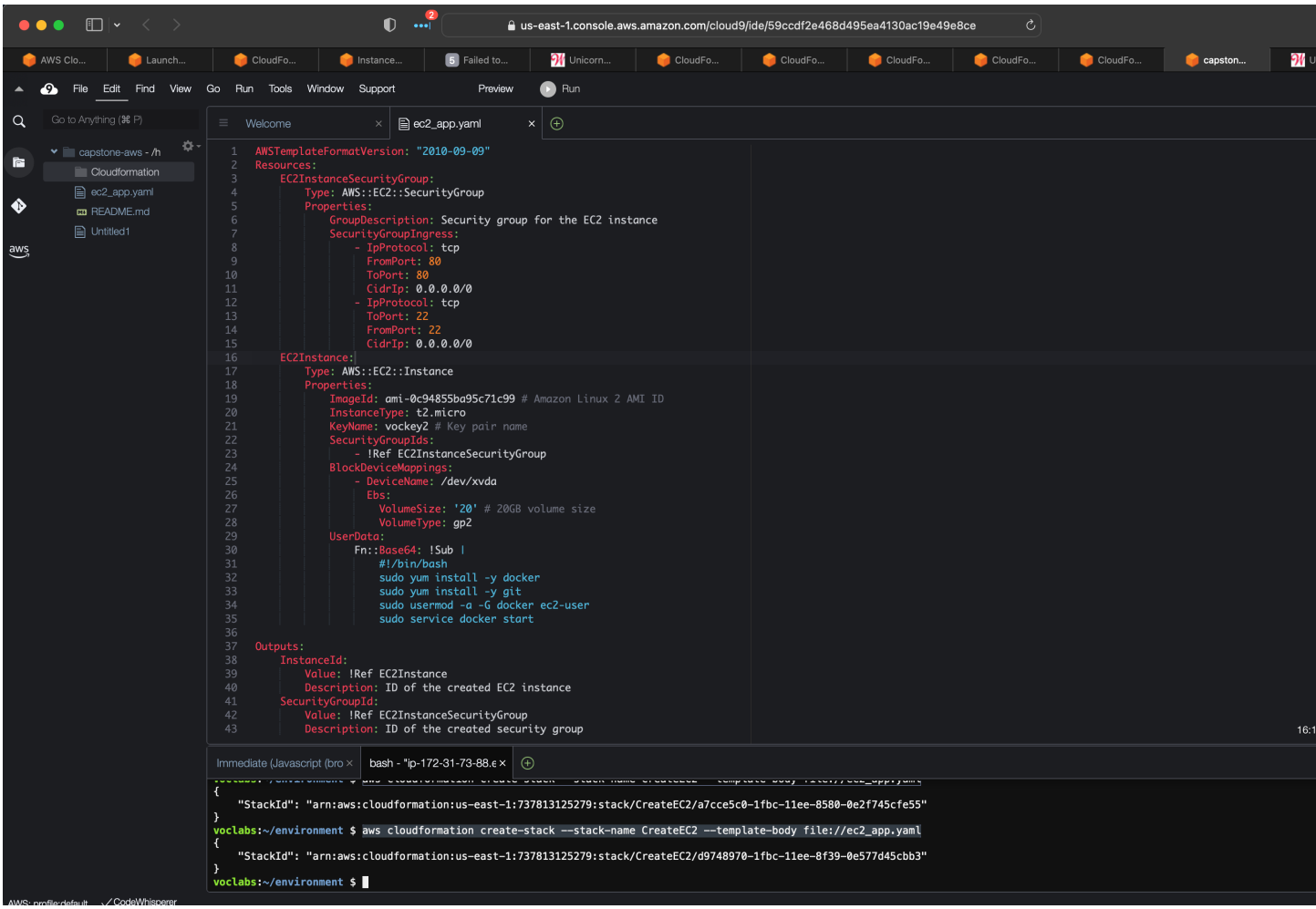
```
[ec2-user@ip-172-31-88-86 ~]$ cd capstone-aws/
[ec2-user@ip-172-31-88-86 capstone-aws]$ ls
Dockerfile  apply.html  faq.html     fonts   index.html      js            ride.html   signin.html    verify.html
README.md   css         favicon.ico  images  investors.html  register.html  robots.txt  unicorns.html
[ec2-user@ip-172-31-88-86 capstone-aws]$ docker build -t capstone-aws .
Sending build context to Docker daemon  22.62MB
Step 1/4 : FROM nginx
latest: Pulling from library/nginx
faef57eae888: Pull complete
76579e9ed380: Pull complete
cf707e233955: Pull complete
91bb7937700d: Pull complete
4b962717ba55: Pull complete
f46d7b05649a: Pull complete
103501419a0a: Pull complete
Digest: sha256:08bc36ad52474e528cc1ea3426b5e3f4bad8a130318e3140d6cfe29c8892c7ef
Status: Downloaded newer image for nginx:latest
 ---> 021283c8eb95
Step 2/4 : COPY . /usr/share/nginx/html
 ---> 67c4a9261316
Step 3/4 : EXPOSE 80
 ---> Running in a809c1a7587a
Removing intermediate container a809c1a7587a
 ---> 08eefcfd0ac0
Step 4/4 : CMD ["nginx", "-g", "daemon off;"]
 ---> Running in 5cae21723548
Removing intermediate container 5cae21723548
 ---> 4f80a31aa7e7
Successfully built 4f80a31aa7e7
Successfully tagged capstone-aws:latest
[ec2-user@ip-172-31-88-86 capstone-aws]$ docker run -dp 80:80 capstone-aws
339cabbe2a4cfa769d813eb787d265b97495bd21c4498689b90efaf9067fbbd6
[ec2-user@ip-172-31-88-86 capstone-aws]$
```

- I copied the public IPV4 address of the EC2 instance and pasted it into my browser and my application came up.



## 5. Infrastructure as code using AWS Cloud Formation:

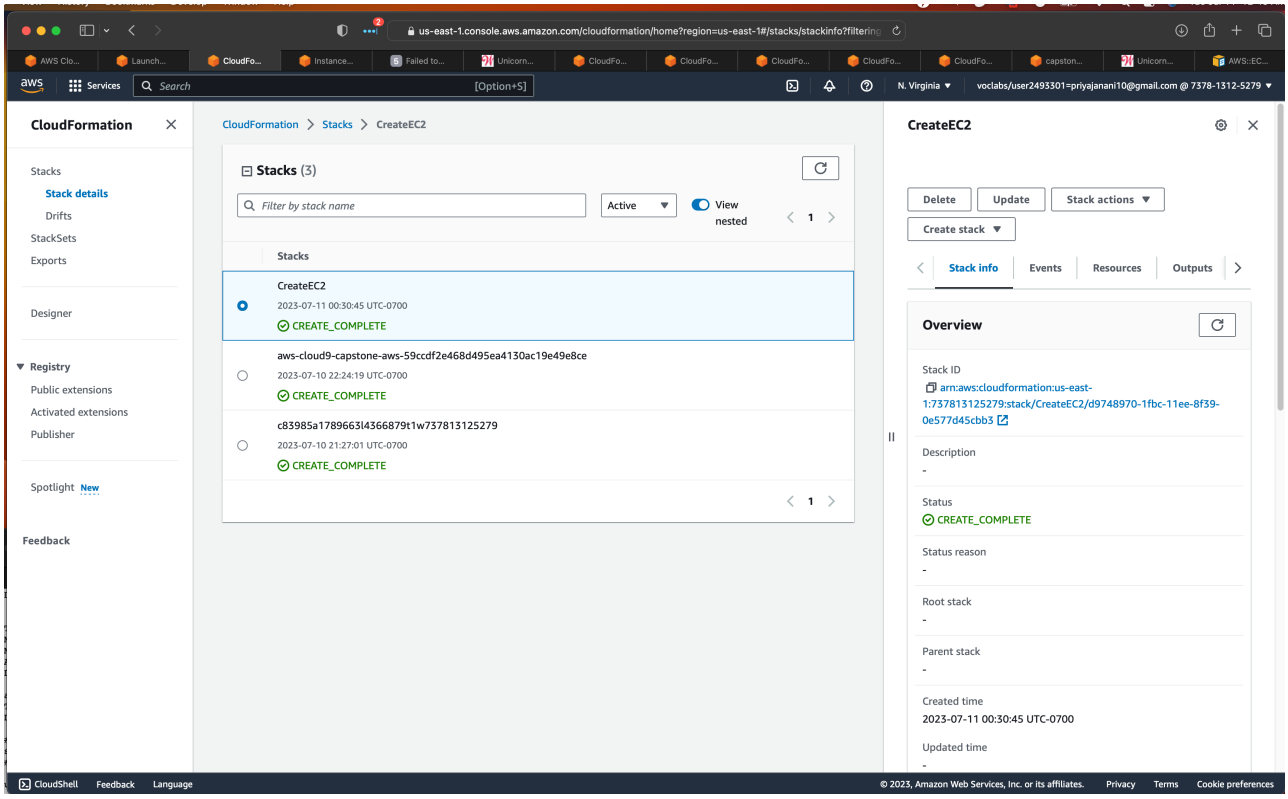
- I created a cloud9 environment and created a new cloud formation folder and a new yaml template file called ec2_app.yaml



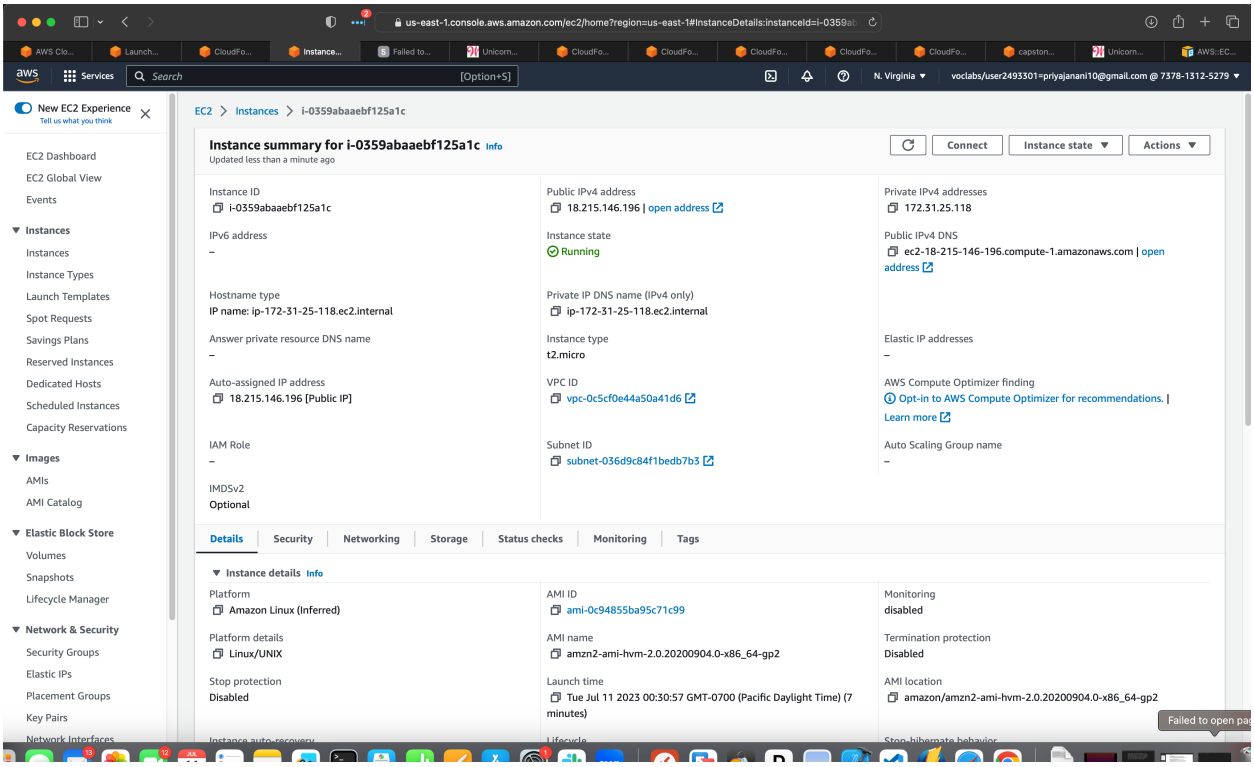

- I created the stack using the following command,

aws cloudformation create-stack --stack-name CreateEC2 --template-body file://ec2_app.yaml

- My created stack is called CreateEC2
- I went to the EC2 instances page and saw that the new EC2 instance had been created.



- This is my EC2 instance that was created.




- I connected to the instance using ssh.
- I built and ran my docker image.

```
janani@Priyas-Air Desktop % ssh -i "vockey2-5.pem" ec2-user@ec2-18-215-146-196.compute-1.amazonaws.com
Last login: Tue Jul 11 07:32:45 2023 from c-98-33-83-15.hsd1.ca.comcast.net

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
54 package(s) needed for security, out of 113 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-25-118 ~]$ cd capstone-aws/
[ec2-user@ip-172-31-25-118 capstone-aws]$ docker build -t test_app .
Sending build context to Docker daemon  22.62MB
Step 1/4 : FROM nginx
latest: Pulling from library/nginx
faef57eae888: Pull complete
76579e9ed380: Pull complete
cf707e233955: Pull complete
91bb7937700d: Pull complete
4b962717ba55: Pull complete
f46d7b05649a: Pull complete
103501419a0a: Pull complete
Digest: sha256:08bc36ad52474e528cc1ea3426b5e3f4bad8a130318e3140d6cfe29c8892c7ef
Status: Downloaded newer image for nginx:latest
 ---> 021283c8eb95
Step 2/4 : COPY . /usr/share/nginx/html
 ---> 804adb4259f2
Step 3/4 : EXPOSE 80
 ---> Running in 8f54446ec24e
Removing intermediate container 8f54446ec24e
 ---> 6e691987a497
Step 4/4 : CMD ["nginx", "-g", "daemon off;"]
 ---> Running in bc0ef7125239
Removing intermediate container bc0ef7125239
 ---> 92fc5f88f5a7
Successfully built 92fc5f88f5a7
Successfully tagged test_app:latest
[ec2-user@ip-172-31-25-118 capstone-aws]$ docker run -dp 80:80 test_app
f7ba88103793667d9c0a9ad6b05faa95bcd1ea934d8aa71e819b875ef2f12507
[ec2-user@ip-172-31-25-118 capstone-aws]$
```

- I launched my application using the public IPv4 address of the EC2 instance and it worked.
- I committed and pushed my ec2_app.yaml file to my repository.