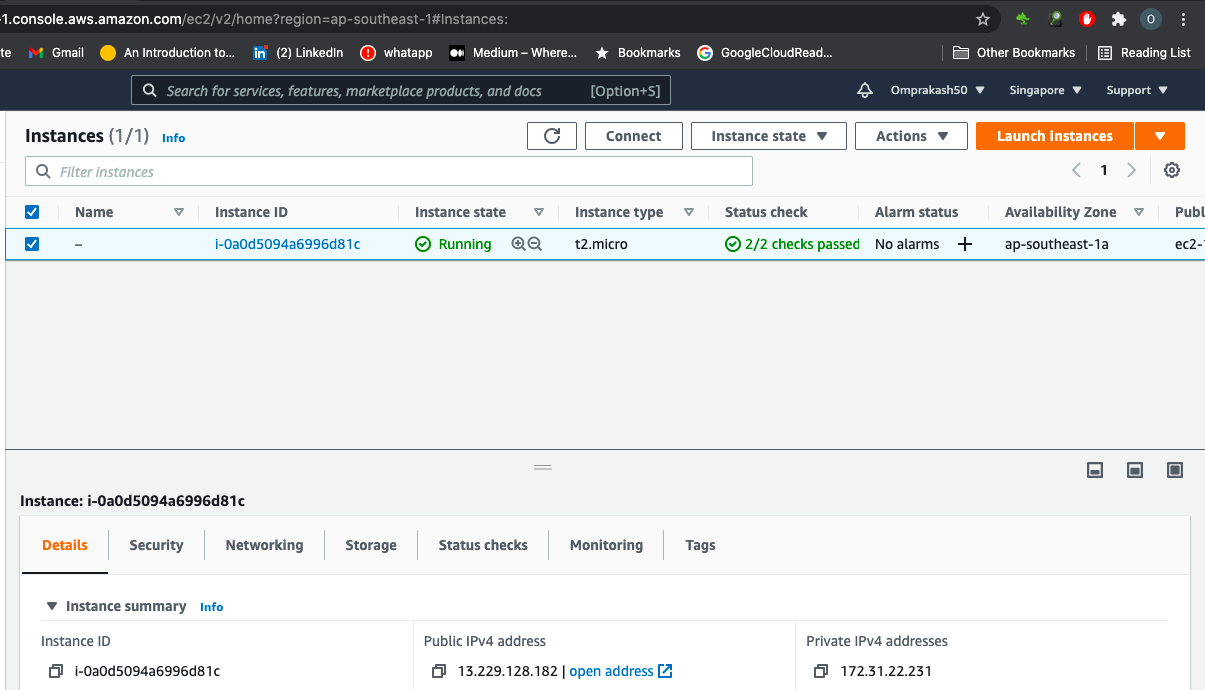
Task -

Use your own free tier AWS account for this assignment

- configure SSL with nginx web server with domain example.com and if end user request with http he must be redirected to https

- create docker build of nginx web server that if user requests it should show "Hello from SoluLab"

First launched instance



For Giving Domain name to this instance i used noip.com

Install and start nginx in dockerhost (baseos):

$yum install nginx -y

$systemctl start nginx

Now For SSL:

dnf install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm

dnf install certbot python3-certbot-nginx

certbot certonly --standalone

**Now type email and domain name, then certbot will provide public ip and private ip and will give path of those key's to us.**

Nginx conf file in dockerhost (baseos):

**In nginx.conf file to listen 443 and to enable ssl write {Listen 443 ssl;}**

**And also give public ip and private ip path in front of keywords ssl\_certificate and ssl\_certificate\_key respectively.**

**Step5: restart nginx**

[root@ip-172-31-22-231 nginx]# cat nginx.conf

http {

log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '

'$status $body\_bytes\_sent "$http\_referer" '

'"$http\_user\_agent" "$http\_x\_forwarded\_for"';

access\_log /var/log/nginx/access.log main;

sendfile on;

tcp\_nopush on;

tcp\_nodelay on;

keepalive\_timeout 65;

types\_hash\_max\_size 2048;

include /etc/nginx/conf.d/\*.conf;

server {

listen 80 default\_server;

## set up domain name here ##

server\_name omprakashexample.ddns.net;

access\_log off;

error\_log off;

return 301 https://$server\_name$request\_uri;

}

server {

listen 443 ssl http2;

ssl\_certificate /etc/letsencrypt/live/omprakashexample.ddns.net/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/omprakashexample.ddns.net/privkey.pem;

ssl\_protocols TLSv1.3;

location / {

proxy\_pass http://localhost:8080/;

}

}

}

events { }

Now this nginx which is located in docker host will do proxy at port no 8080, So idea is to expose docker container on port no 8080.

Launching Container : I am using CentOS image :

$docker run -it -p 8080:80 --name op centos

$yum install nginx -y (install nginx)

$/usr/sbin/nginx (start nginx service)

Docker container nginx configuration file:

[root@8e8c92dc43e1 nginx]# cat nginx.conf

user nginx;

worker\_processes auto;

error\_log /var/log/nginx/error.log;

pid /run/nginx.pid;

# Load dynamic modules. See /usr/share/doc/nginx/README.dynamic.

include /usr/share/nginx/modules/\*.conf;

events {

worker\_connections 1024;

}

http {

log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '

'$status $body\_bytes\_sent "$http\_referer" '

'"$http\_user\_agent" "$http\_x\_forwarded\_for"';

access\_log /var/log/nginx/access.log main;

sendfile on;

tcp\_nopush on;

tcp\_nodelay on;

keepalive\_timeout 65;

types\_hash\_max\_size 2048;

include /etc/nginx/mime.types;

default\_type application/octet-stream;

include /etc/nginx/conf.d/\*.conf;

server {

listen 80 default\_server;

listen [::]:80 default\_server;

server\_name \_;

root /usr/share/nginx/html;

# Load configuration files for the default server block.

include /etc/nginx/default.d/\*.conf;

location / {

}

error\_page 404 /404.html;

location = /40x.html {

}

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

}

}

}

Now create the webpage with content in docker container:

"Hello from SoluLab"

Final ouput of <http://omprakashexample.ddns.net/>

