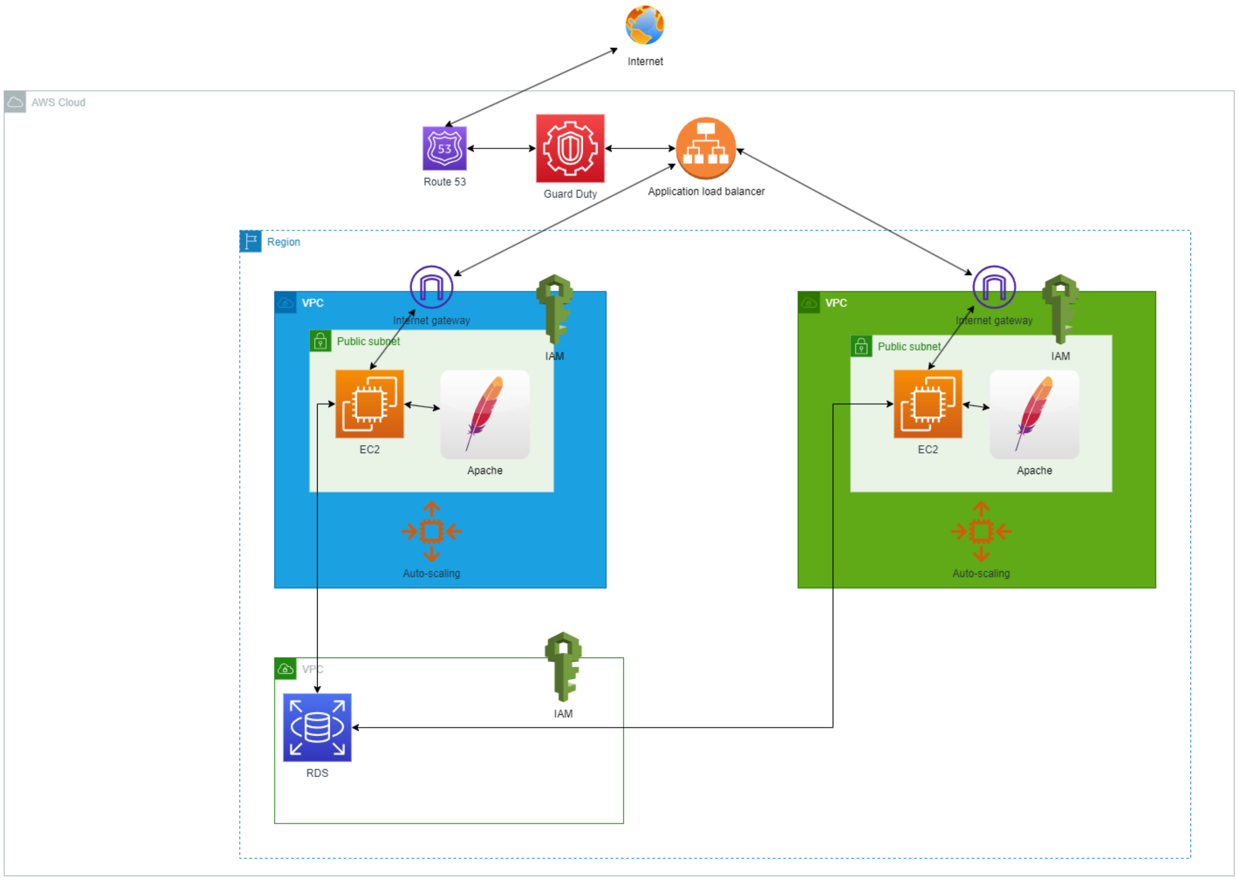
Theoretical part

Architectural diagram of the project on AWS Cloud with frontend and backend:



Picture 1

My project consists of two separate EC2 instances with public Apache servers(frontend) and private databases(RDS it can be MySql or something else; it`s backend). I`ve made internet gateways so the users outside the VPC have access. There are two VPC`s. Green and Blue. The green one gets updated first. It gets tested and if everything ok the application load balancer switches traffic over to green one. After that blue one gets the same update as a green one and then the application load balancer switches traffic over again to blue. If something wrong it rolls back to previous version and programmers change the code of update. Then i(as devops) deploy and test everything again. If some servers shut down, the auto-scaling group rewrites/remakes the EC2 instance with services. Also EC2 has auto-scaling.

To have secure access to VPC`s and everything inside them i created IAM role with MFA. And also GuardDuty works as protection from internet threats.

Practical Part

Solution for deploying a Docker application consisting of 2 components: NGINX + PHP.

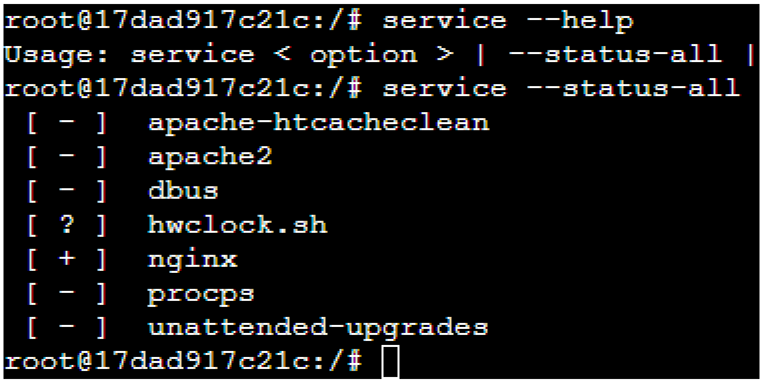
Created a EC2 instance in AWS. A docker image was created from Dockerfile.

Dockerfile code:

```
FROM ubuntu:22.04

ENV DEBIAN_FRONTEND noninteractive

RUN apt update && apt install -y tcl
```

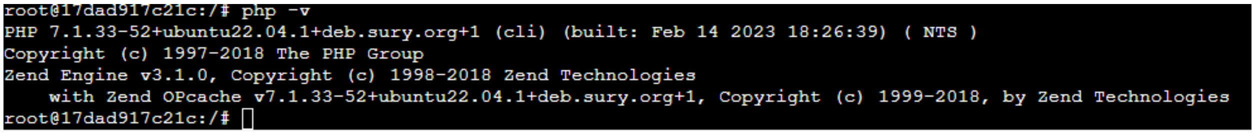
```
LABEL maintainer="TarSyr"

RUN apt-get update && \
   apt-get install -y -q curl gnupg2
RUN curl http://nginx.org/keys/nginx_signing.key | apt-key add -
RUN apt-get purge nginx nginx-common nginx-full
RUN apt-get update && \
   apt-get install -y -q nginx
RUN apt update
RUN apt install apache2 -y
RUN apt install apache2-utils -y
RUN apt clean
RUN apt-get install software-properties-common -y
RUN add-apt-repository ppa:ondrej/php
RUN apt-get update
RUN apt purge libapache2-mod-php7.1
RUN apt install libapache2-mod-php7.1 -y
RUN a2enmod php7.1
RUN service apache2 restart
EXPOSE 443 80

CMD [“apache2ctl”, “-D”, “FOREGROUND”]
CMD ["nginx", "-g", "daemon off;"]
```

Then i run it. Start the container. Checking services which has to be installed(NGINX+PHP):



```
root@17dad917c21c:/# service --help
Usage: service < option > | --status-all |
root@17dad917c21c:/# service --status-all
 [ - ]  apache-htcacheclean
 [ - ]  apache2
 [ - ]  dbus
 [ ? ]  hwclock.sh
 [ + ]  nginx
 [ - ]  procps
 [ - ]  unattended-upgrades
root@17dad917c21c:/#
```

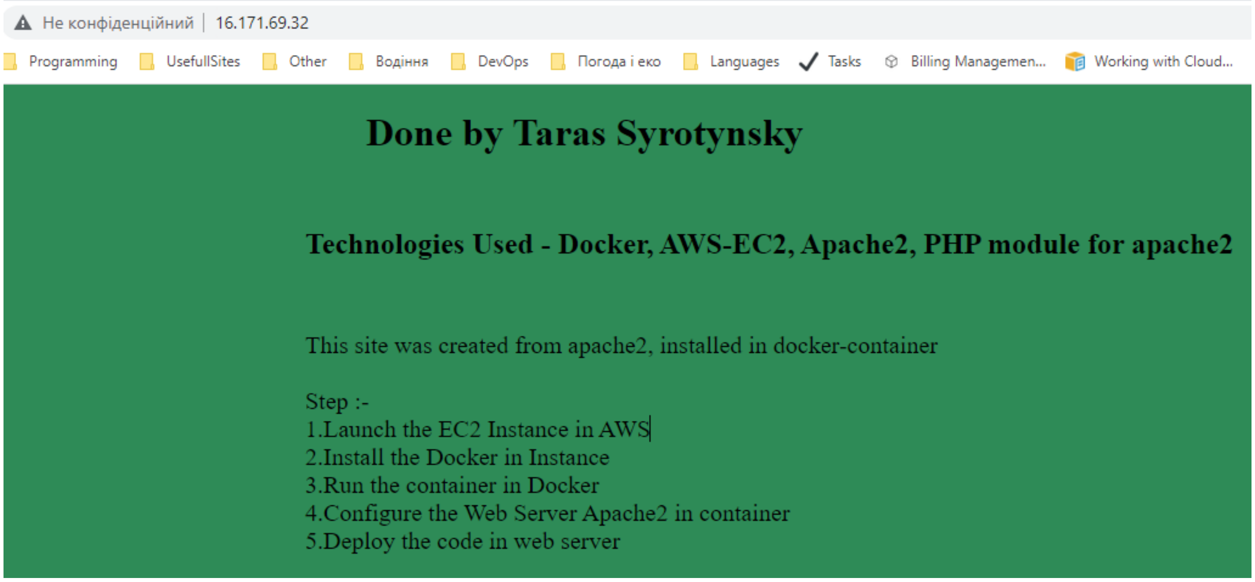
Picture 2 There is NGINX



```
root@17dad917c21c:/# php -v
PHP 7.1.33-52+ubuntu22.04.1+deb.sury.org+1 (cli) (built: Feb 14 2023 18:26:39) ( NTS )
Copyright (c) 1997-2018 The PHP Group
Zend Engine v3.1.0, Copyright (c) 1998-2018 Zend Technologies
    with Zend OPcache v7.1.33-52+ubuntu22.04.1+deb.sury.org+1, Copyright (c) 1999-2018, by Zend Technologies
root@17dad917c21c:/#
```

Picture 3 PHP version 7.1.33-52

Site is accessible for everyone in the internet. But i didn`t buy and register a domain name yet.
Public IP: http://16.171.69.32/
Results:



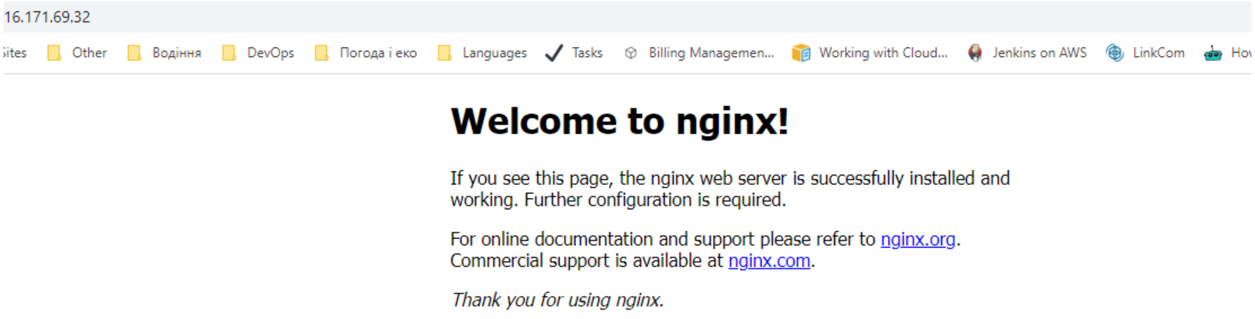
Done by Taras Syrotynsky

Technologies Used - Docker, AWS-EC2, Apache2, PHP module for apache2

This site was created from apache2, installed in docker-container

Step :-
1.Launch the EC2 Instance in AWS
2.Install the Docker in Instance
3.Run the container in Docker
4.Configure the Web Server Apache2 in container
5.Deploy the code in web server

Picture 4 Apache



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Picture 5 NGINX