Setup SSL Enabled Website

Connect to Ubuntu Docker Server and become root

```
Unset
sudo su -
```

Run an Ubuntu Container with pre-installed Apache2

```
Unset
docker run -d -p 80:80 -p 443:443 ubuntu/apache2:latest
```

Find the Container ID

```
Unset
docker ps -a
```

Connect to the Container and log inside

```
Unset
docker exec -i -t <containerID> /bin/bash
```

Within the container execute the commands to install a website
What we are doing is downloading a Website from the URL mentioned, unzipping it and transferring website files to a folder `/var/www/cloudiq.online.`

This folder will be used as our base folder for the website.

In your practice you need to change the domain name you purchased.

In my case I am using cloudiq.online. I want my website to be accessible by these 2 URLs:

https://www.cloudiq.online or https://cloudiq.online

```
Unset
cd /root
mkdir /var/www/cloudiq.online

#Download the website you want to host and unzip it
#Install wget
apt install wget

cd /tmp

wget
https://www.free-css.com/assets/files/free-css-templates/downl
oad/page289/bluene.zip

apt-get install unzip
unzip bluene.zip

#Copy the Unzipped folder contents to the base folder
cp -R bluene-html/. /var/www/cloudiq.online

#Setting a few permissions to the Website base folder
chown -R $USER:$USER /var/www/cloudiq.online
     chmod -R 755 /var/www/cloudiq.online
```

Create the Website configuration file and save it

```
Unset
#Install nano
apt install -y nano

nano /etc/apache2/sites-available/cloudiq.online.conf

<VirtualHost *:80>
    ServerAdmin webmaster@cloudiq.online
    ServerName cloudiq.online
    ServerAlias www.cloudiq.online
    DocumentRoot /var/www/cloudiq.online
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
```

```
</VirtualHost>
```

Enable the website and disable the default website which we don't need

```
Unset
a2ensite cloudiq.online.conf
service apache2 reload
a2dissite 000-default.conf
service apache2 reload

#verify accessing the non-SSL website
http://cloudiq.online
```

Enable the SSL by installing a tool called certbot

```
Unset
apt install -y certbot python3-certbot-apache

#Install the SSL to our website
certbot --apache
```

Do the following selections as the above command gives interactive output

```
Unset
#Output

Saving debug log to /var/log/letsencrypt/letsencrypt.log
Enter email address (used for urgent renewal and security
notices)
 (Enter 'c' to cancel): webmaster@cloudiq.online
```

After providing a valid email address, press ENTER to proceed to the next step. You will then be prompted to confirm if you agree to Let's Encrypt terms of service. You can confirm by pressing Y and then ENTER:

```
Unset
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - -
Please read the Terms of Service at
https://letsencrypt.org/documents/LE-SA-v1.2-November-15-2017.
pdf. You must
agree in order to register with the ACME server at
https://acme-v02.api.letsencrypt.org/directory
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - -
(Y)es/(N)o: Y
```

Next, you'll be asked if you would like to share your email with the Electronic Frontier Foundation to receive news and other information. If you do not want to subscribe to their content, write N. Otherwise, write Y then press ENTER to proceed to the next step:

```
Unset
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - -
Would you be willing to share your email address with the
Electronic Frontier
Foundation, a founding partner of the Let's Encrypt project
and the non-profit
organization that develops Certbot? We'd like to send you
email about our work
encrypting the web, EFF news, campaigns, and ways to support
digital freedom.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - -
(Y)es/(N)o: N
```

The next step will prompt you to inform Certbot of which domains you'd like to activate HTTPS for.

```
Unset
Which names would you like to activate HTTPS for?
- - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - -
1: your_domain #in my case cloudiq.online
2: www.your_domain #in my case www.cloudiq.online
- - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - -
Select the appropriate numbers separated by commas and/or
spaces, or leave input
blank to select all options shown (Enter 'c' to cancel):

1
```

SSL certificate is now installed and loaded into Apache's configuration. Try reloading your website using: **https://cloudiq.online**

Main Step

Export the container as Docker Image

```
Unset
docker commit <containerID> <newImageName>
```

```
docker commit 097b9d54aa52 website-ssl
```

Verify the Image creation

```
Unset
root@ubuntu22:~# docker images
REPOSITORY       TAG       IMAGE ID       CREATED        SIZE
website-ssl      latest    26a6015f3619   8 hours ago    291MB
```

Remove all the containers and run a new container with the created image. You should be able to access using https://cloudiq.online

```
Unset
#Delete all running containers

root@ubuntu22:~# docker rm -f $(docker ps -aq)
097b9d54aa52
efd934f877df
```

Create a new containenr based on our newly created image.

```
Unset
root@ubuntu22:~# docker run -d -p 80:80 -p 443:443 website-ssl
af14a5dce9b08fc3292721604fc7dd0e0c63411228539e1356ca53892b73f7
48
```

Verify the container running state.

```
Unset
root@ubuntu22:~# docker ps
CONTAINER ID   IMAGE         COMMAND                  CREATED
STATUS          PORTS
NAMES
af14a5dce9b0   website-ssl   "apache2-foreground"   35 seconds
ago   Up 35 seconds   0.0.0.0:80->80/tcp, :::80->80/tcp,
0.0.0.0:443->443/tcp, :::443->443/tcp   epic_franklin
```

Access the website:
https://cloudiq.online or https://www.cloudiq.online

Make Image from the Container and Push Image to docker
`docker commit <containerID> <newImageName>`

`docker commit 2efb1b8e59ed tanvisinghny/ssl-website`

`docker push tanvisinghny/ssl-website`