September Milestone Report

Group 7

- Jaideep Guntupalli(2020378)
- Cyrus Monteiro(2020368)
- Ayush Singhal(2020365)
- Pranav Sharma(2020395)

Tech Stack

OS

Linux(Ubuntu) is being used as an operating system in the virtual machine.

Web Server

Nginx will be used as web server to host the website in the alloted virtual machine.

Language

Javascript / Typescript will be used as a primary language through out the project.

Frontend Frameworks

ReactJS will be used as a primary frontend framework, combined with **TailwindCSS** for styling the frontend.

Backend Technologies

We are planning to use **NestJS**, an opinionated backend framework based on NodeJS with **Prisma** as an ORM to easily communicate with the database.

Database

We are planning to use **PostgreSQL** as primary database.

Installation Steps

1. Install nginx web server

```
sudo apt install nginx
```

2. Created a basic html site

```
mkdir site

cd site

echo "<h1>Hello World, This is Jaideep, Pranav, Ayush and Cyrus!!</h1>" >> index.html
```

This created to host the site.

3. Created a temp openssl configuration to add IP address as subject alternate name

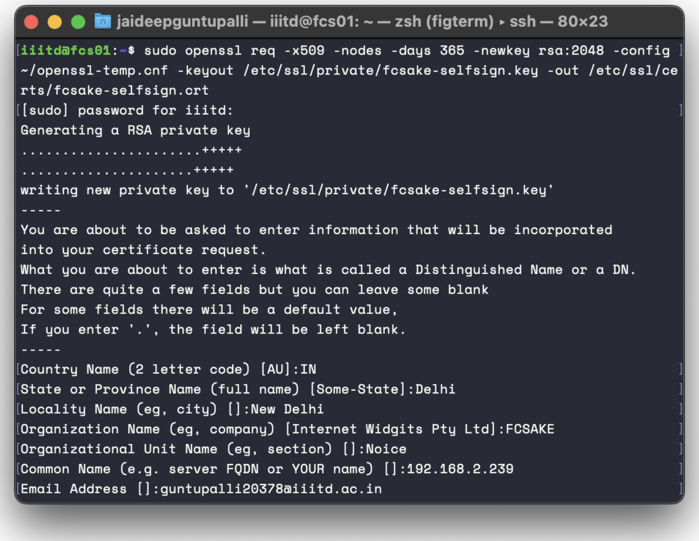
```
cp /etc/ssl/openssl.cnf ~/openssl-temp.cnf

nano openssl-temp.cnf

# Changes made
[ v3_ca ]

subjectAltName = IP:192.168.2.239
```

This configuration is created to generate ssl certificate in next step.

4. Created a self-signed certificate and key pair with OpenSSL and temp configuration

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048
-config ~/openssl-temp.cnf -keyout /etc/ssl/private/fcsake-selfsign.key
-out /etc/ssl/certs/fcsake-selfsign.crt
```

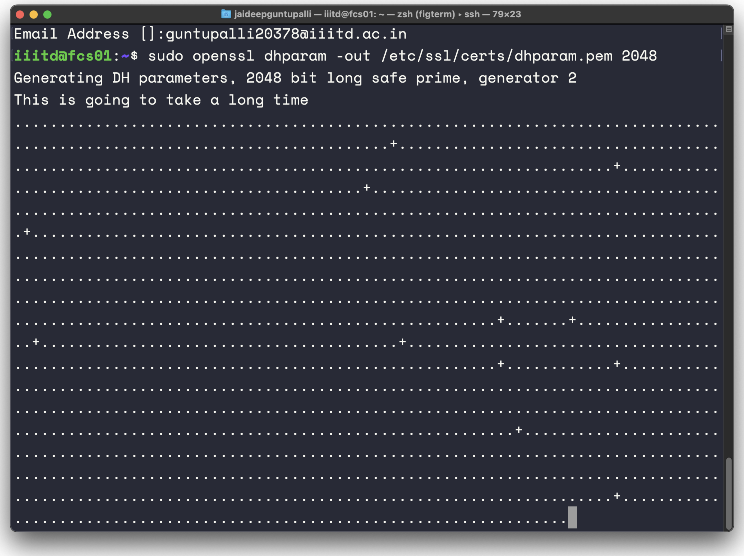
This adds the subject alternate name which is verified by browsers with the actual domain/ip address used to visit the site.


jaideepguntupalli — iiitd@fcs01: ~ — zsh (figterm) ▸ ssh — 80×23

```
iiitd@fcs01:~$ sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -config
~/openssl-temp.cnf -keyout /etc/ssl/private/fcsake-selfsign.key -out /etc/ssl/ce
rts/fcsake-selfsign.crt
[sudo] password for iiitd:
Generating a RSA private key
......................+++++
.....................+++++
writing new private key to '/etc/ssl/private/fcsake-selfsign.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:IN
State or Province Name (full name) [Some-State]:Delhi
Locality Name (eg, city) []:New Delhi
Organization Name (eg, company) [Internet Widgits Pty Ltd]:FCSAKE
Organizational Unit Name (eg, section) []:Noice
Common Name (e.g. server FQDN or YOUR name) []:192.168.2.239
Email Address []:guntupalli20378@iiitd.ac.in
```

5. Created a strong Diffie-Hellman key pair, which will be used to ensure no key will compromise even with longer sessions with clients

```
sudo openssl dhparam -out /etc/ssl/certs/dhparam.pem 2048
```

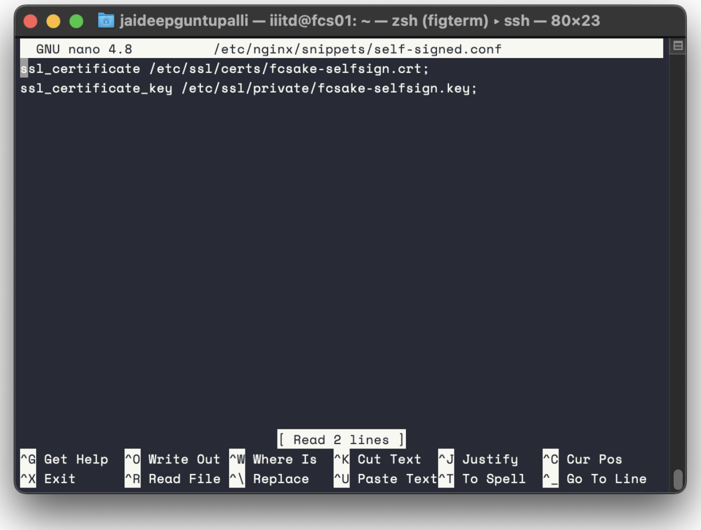


jaideepguntupalli — iiitd@fcs01: ~ — zsh (figterm) ▸ ssh — 79×23

```
Email Address []:guntupalli20378@iiitd.ac.in
iiitd@fcs01:~$ sudo openssl dhparam -out /etc/ssl/certs/dhparam.pem 2048
Generating DH parameters, 2048 bit long safe prime, generator 2
This is going to take a long time
...............................................................................
..........................................+....................................
...................................................................+...........
.......................................+.......................................
...............................................................................
.+.............................................................................
...............................................................................
...............................................................................
...............................................................................
......................................................+.......+................
..+........................................+...................................
......................................................+............+...........
...............................................................................
...............................................................................
........................................................+......................
...............................................................................
...............................................................................
...................................................................+...........
..............................................................
```

6. Created a new Nginx configuration snippet in the `/etc/nginx/snippets` directory pointing to the SSL Key and Certificate

```
sudo nano /etc/nginx/snippets/self-signed.conf
```



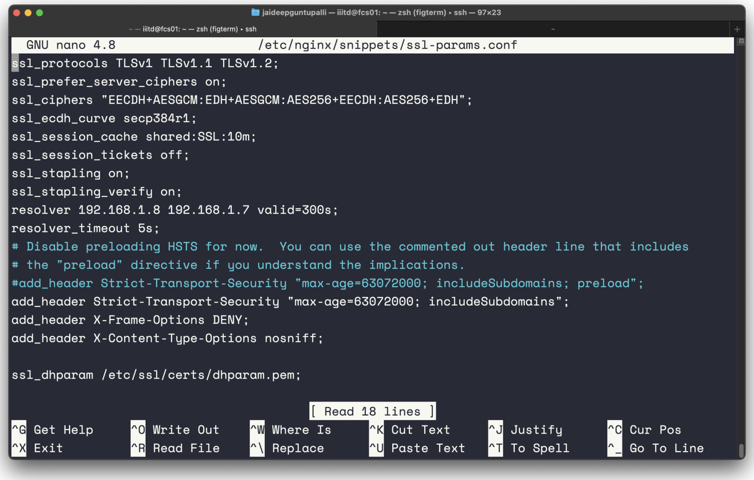
jaideepguntupalli — iiitd@fcs01: ~ — zsh (figterm) ▸ ssh — 80×23

```
  GNU nano 4.8           /etc/nginx/snippets/self-signed.conf
ssl_certificate /etc/ssl/certs/fcsake-selfsign.crt;
ssl_certificate_key /etc/ssl/private/fcsake-selfsign.key;
```

[Read 2 lines]
^G Get Help   ^O Write Out ^W Where Is   ^K Cut Text   ^J Justify   ^C Cur Pos
^X Exit   ^R Read File ^\ Replace   ^U Paste Text ^T To Spell   ^_ Go To Line

7. Created a new Nginx configuration snippet in the `/etc/nginx/snippets` directory with Strong Encryption Settings

```
sudo nano /etc/nginx/snippets/ssl-params.conf
```



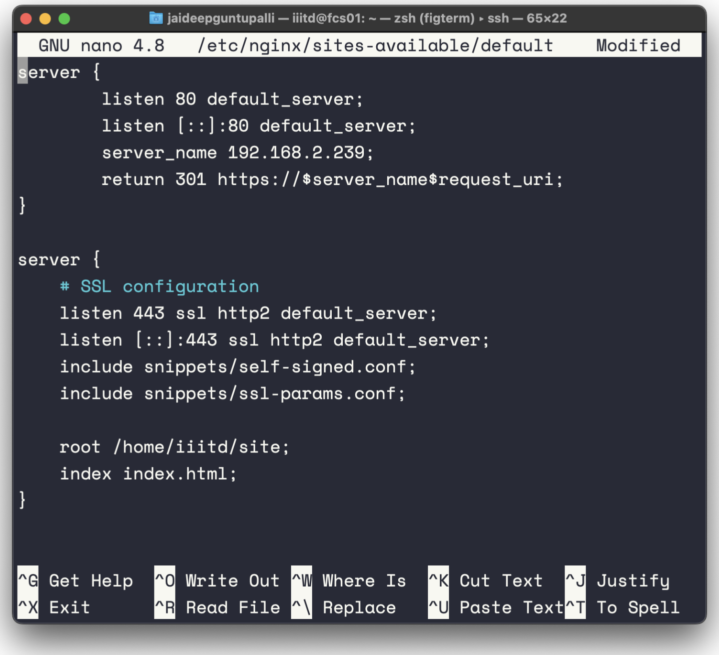
```
jaideepguntupalli — iiitd@fcs01: ~ — zsh (figterm) ▸ ssh — 97×23
~ — iiitd@fcs01: ~ — zsh (figterm) ▸ ssh
  GNU nano 4.8                    /etc/nginx/snippets/ssl-params.conf
ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
ssl_prefer_server_ciphers on;
ssl_ciphers "EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH";
ssl_ecdh_curve secp384r1;
ssl_session_cache shared:SSL:10m;
ssl_session_tickets off;
ssl_stapling on;
ssl_stapling_verify on;
resolver 192.168.1.8 192.168.1.7 valid=300s;
resolver_timeout 5s;
# Disable preloading HSTS for now.  You can use the commented out header line that includes
# the "preload" directive if you understand the implications.
#add_header Strict-Transport-Security "max-age=63072000; includeSubdomains; preload";
add_header Strict-Transport-Security "max-age=63072000; includeSubdomains";
add_header X-Frame-Options DENY;
add_header X-Content-Type-Options nosniff;

ssl_dhparam /etc/ssl/certs/dhparam.pem;

                                          [ Read 18 lines ]
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text  ^T To Spell    ^_ Go To Line
```

8. Configuring the server block to host our site with proper ssl configuration

```
sudo nano /etc/nginx/sites-available/default
```

The http server block at the top accepts requests and permanently redirects to https requests so we only get https requests.

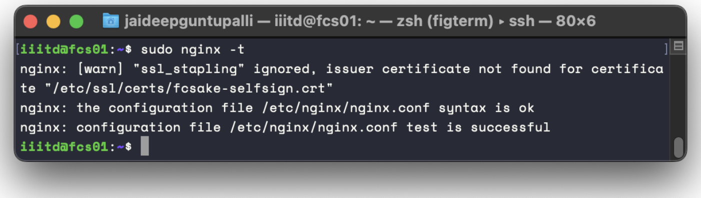
The second server block handles https requests where we mentioned to include these snippets for including the ssl certificate and strong encryption settings which we configured earlier. At last we mentioned the folder where our index.html is present. This results in nginx returning the index.html when any request is made to the server.



```
jaideepguntupalli — iiitd@fcs01: ~ — zsh (figterm) ▸ ssh — 65×22
  GNU nano 4.8    /etc/nginx/sites-available/default    Modified
server {
        listen 80 default_server;
        listen [::]:80 default_server;
        server_name 192.168.2.239;
        return 301 https://$server_name$request_uri;
}

server {
    # SSL configuration
    listen 443 ssl http2 default_server;
    listen [::]:443 ssl http2 default_server;
    include snippets/self-signed.conf;
    include snippets/ssl-params.conf;

    root /home/iiitd/site;
    index index.html;
}


^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify
^X Exit       ^R Read File  ^\ Replace    ^U Paste Text ^T To Spell
```

9. Checking whether all syntax related to nginx is ok.

```
sudo nginx -t
```


jaideepguntupalli — iiitd@fcs01: ~ — zsh (figterm) ▸ ssh — 80×6

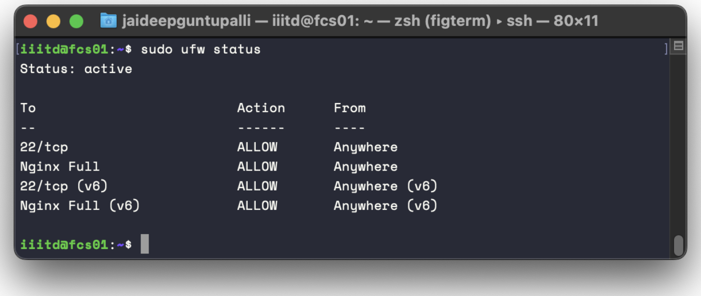
```
iiitd@fcs01:~$ sudo nginx -t
nginx: [warn] "ssl_stapling" ignored, issuer certificate not found for certifica
te "/etc/ssl/certs/fcsake-selfsign.crt"
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
iiitd@fcs01:~$
```

10. Since all is ok, we can restart the nginx service bring changes into effect

```
sudo systemctl restart nginx
```

11. Adjusted the firewall to accept NGINX Full profile to let in HTTPS traffic and ssh requests

```
sudo ufw allow 'Nginx Full'
sudo ufw allow 'ssh'
```


jaideepguntupalli — iiitd@fcs01: ~ — zsh (figterm) ▸ ssh — 80×11

```
iiitd@fcs01:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
Nginx Full                 ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
Nginx Full (v6)            ALLOW       Anywhere (v6)

iiitd@fcs01:~$
```

And the sample site is hosted. We can download the CA from here, to install it to root directory so browsers can trust the certificate and encrypt the data.

Sample site hosted

- 192.168.2.239
- Install the private CA to root by downloading from here