**PUBLICAR UNA APP (NODEJS Y ANGULAR) CON NGINX EN CENTOS 8 (HOST DIGITAL OCEAN)**

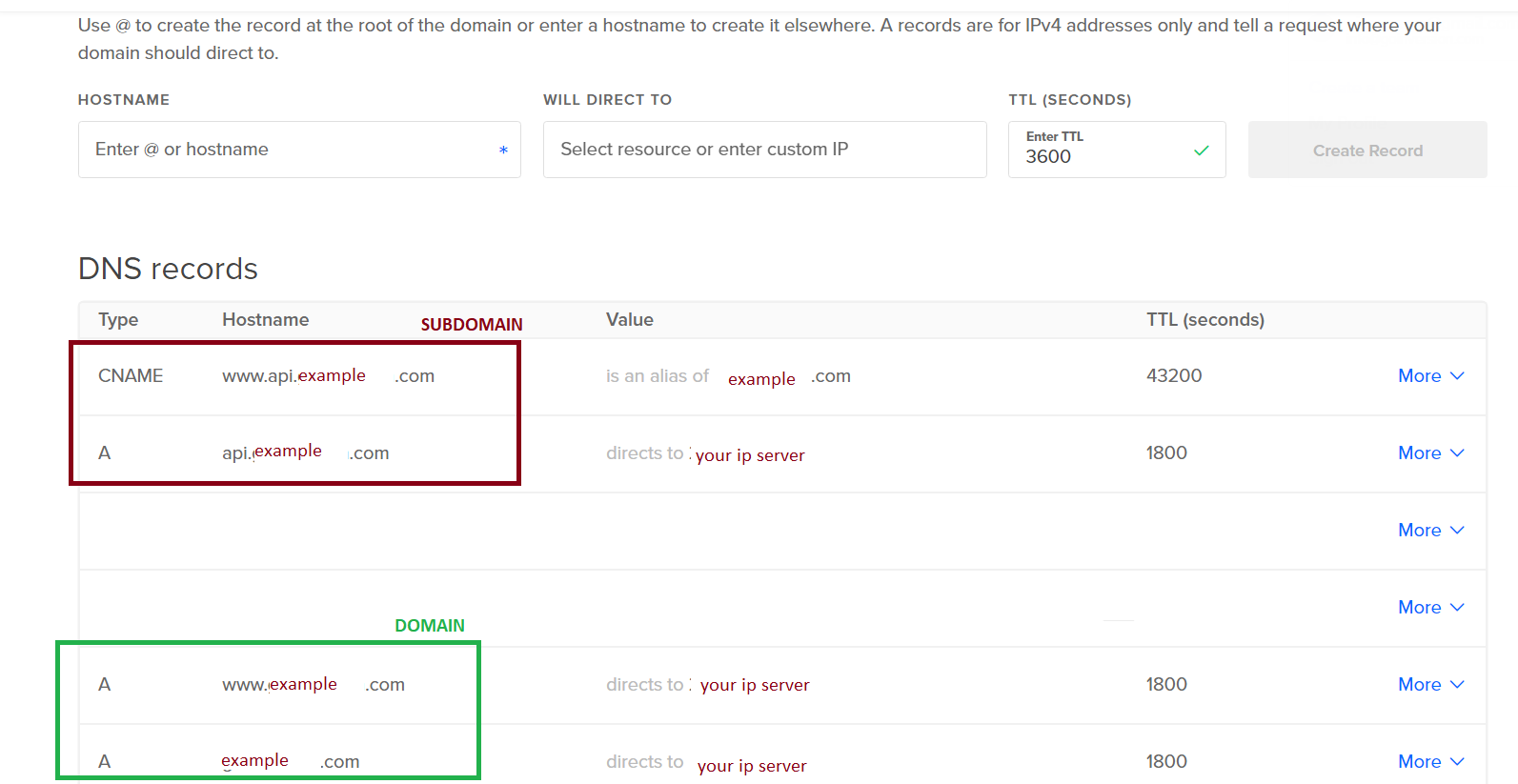
**LA APP HA PUBLICAR ESTA DESARROLLADA EN:**

**BACKEND:** FRAMEWORK NESTJS (NODEJS)

**FRONTEND:** ANGULAR 9

1. **Requisitos:**

* Tener ejecutándose el backend y haber hecho el ng build –prod para obtener la carpeta dist
* Tener dominio y subdominios creados previamente ejemplo:



1. **Creación de carpeta para Frontend**

Creamos la carpeta donde tendremos que colocar los archivos de la carpeta dist del frontend cambiar example.com por tu dominio:

$ sudo mkdir -p /var/www/example.com/html

Copiamos todos los archivos de la carpeta dist (solo archivos) a nuestra dirección creada previamente y la renombramos como html:

$ cp -R dist     /var/www/var/www/example.com/html

Otorgamos permisos (en $USER colocar el nombre del usuario actual ejemplo root:root):

$ sudo chown -R $USER:$USER /var/www/your\_domain/html

El siguiente comando permitirá presentar el root de su documento personalizado como contenido HTTP:

$ chcon -vR system\_u:object\_r:httpd\_sys\_content\_t:s0 /var/www/your\_domain/

1. **Instalación de NGINX en Centos:**

Ejecutar los siguientes comandos necesarios para el funcionamiento e instalación correcta de nginx:

$ sudo yum install epel-release

$ sudo yum install nginx

Habilitamos permanentemente las conexiones http y https:

$ sudo firewall-cmd --permanent --add-service=http

$ sudo firewall-cmd --permanent --add-service=https

Reiniciamos el Firewall

$ sudo firewall-cmd –-reload

Usar el editor de su preferencia para edición de archivos en el Shell en este caso se usará vim:

Nos dirigimos a la carpeta conf.d en la siguiente ruta

$ cd /etc/nginx/conf.d

Crear un archivo con extencion .conf ejemplo:

$ vim example.com.conf

Dentro del archivo configuramos lo siguiente cambiar los datos de example por el dominio y subdominio de su aplicación.

#Frontend

server{

listen 80;

       server\_name example.com  www.example.com;

       location / {

                root /var/www/example.com/html;

                index index.html index.htm index.nginx-debian.html;

                try\_files $uri $uri/ /app/index.html;

       }

}

#Backend

server {

listen: 80;

       server\_name api.example.com  www.api.example.com;

       location / {

                proxy\_pass http://localhost:3001;

                proxy\_http\_version 1.1;

                proxy\_set\_header Upgrade $http\_upgrade;

                proxy\_set\_header Connection 'upgrade';

                proxy\_set\_header Host $host;

                proxy\_cache\_bypass $http\_upgrade;

       }

}

Guardamos los cambios y ejecutamos los siguientes comandos:

$ sudo systemctl start nginx

$ sudo systemctl enable nginx

Comprobamos el estado de Nginx

$ sudo systemctl status nginx

Activamos los siguientes valores booleanos:

$ sudo setsebool -P httpd\_can\_network\_relay on

$ sudo setsebool -P httpd\_can\_network\_connect on

Reiniciamos Nginx

$ sudo systemctl restart nginx

1. **Certificados SSL con LetsEncrypt**

Para tener los certificados ssl en nuestros dominios y subdominios ejecutamos los siguientes comandos:

$ sudo dnf install certbot

$ sudo dnf install certbot python3-certbot-nginx

Añadimos los certificados a los dominios y subdominios remplazar example.com por su dominio:

$ sudo certbot --nginx -d example.com -d www.example.com -d api.example.com -d www.api.example.com

Para renovar automáticamente el certificado copiar y pegar el siguiente comando:

$ echo "0 0,12 \* \* \* root python3 -c 'import random; import time; time.sleep(random.random() \* 3600)' && certbot renew -q" | sudo tee -a /etc/crontab > /dev/null

No solicitara un correo lo ingresamos y Enter.

Aceptamos los términos con A y Enter.

**No** agregamos nuestro correo a la fundación Electronic Frontier colocamos N y Enter.

Esperamos la validación y ya estaría configurados los SSL en tus dominios y subdominio.