Load Balancer Setup Instructions

Installation of Pre-requisites

1. Update Apt Get

**$sudo apt-get update**

1. Install NPM

**$sudo apt-get install npm**

1. Install nginx to route http:80 to the custom port

**$sudo apt-get install nginx**

1. Install Build Tools

**$sudo apt-get install -y build-essential**

Configure Nginx to Route Traffic from :80 (http) to :8080 (node js)

Configure Nginx to route all incoming trafiff on port 80 (HTTP) to the Node JS app that is listening on port 8080.

Production:

1. **$sudo rm /etc/nginx/sites-enabled/default**
2. Create a new file: **/etc/nginx/sites-available/captionify.com** and add these lines to it:

**server {**

**listen 80;**

**server\_name www.captionify.com;**

**location / {**

**proxy\_set\_header X-Forwarded-For $remote\_addr;**

**proxy\_set\_header Host $http\_host;**

**proxy\_pass "http://127.0.0.1:8080";**

**}**

**}**

1. **$sudo ln -s /etc/nginx/sites-available/captionify.com /etc/nginx/sites-enabled/captionify.com**
2. **$sudo service nginx restart** #restart service

Staging:

1. **$sudo rm /etc/nginx/sites-enabled/default**
2. Create a new file: **/etc/nginx/sites-available/captionify.com** and add these lines to it:

**server {**

**listen 80;**

**server\_name stage.captionify.com;**

**location / {**

**proxy\_set\_header X-Forwarded-For $remote\_addr;**

**proxy\_set\_header Host $http\_host;**

**proxy\_pass "http://127.0.0.1:8080";**

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

        proxy\_set\_header X-Forwarded-Proto $scheme; #these are for passing client ip to node server

client\_max\_body\_size 200M;

**}**

**}**

1. **$sudo ln -s /etc/nginx/sites-available/captionify.com /etc/nginx/sites-enabled/captionify.com**
2. **$sudo service nginx restart** #restart service

Configure DNS Entry on Route 53

Before we can configure Nginx to route HTTPS traffic, we need to ensure two things:

* First, the DNS entry for the domain name should point to the correct public IPv4 address so the test is able to reach the right server
* Second, the server’s inbound security group settings should be opened up (temporarily, even) for both HTTP and HTTPS traffic

For setting up the DNS entry, follow the steps below:

1. Go to AWS Route 53 resource
2. Find the domain name entry with Type A record that matches production or stage url. Production URL is ‘www.captionify.com’ and stage URL is ‘stage.captionify.com’
3. If this is the first time in Route 53, you need to:
   1. First, create a Hosted Zone. You can name it captionify.com
   2. Then, in the hosted zone, created a record set, and add the IP address as guided here.
   3. Note: the domain name servers listed in ‘NS’ should match the name servers in the source domain name registrar (such as GoDaddy.com)
4. Edit the IP address and enter the EC2 server’s public IPv4 address and save it

For Editing Inbound Security Groups:

1. Go to AWS EC2 Security Group corresponding to the server (production or security)
2. Edit the Inbound security group settings
3. Add HTTP and HTTPS (separate rows) with “Anywhere” and Save

Configure Nginx to Route Traffic from :443 (https) to :8080 (node js)

Setup Certbot / Lets Encrypt for Automatic SSL Certificates (Excellent Tutorial: <https://www.digitalocean.com/community/tutorials/how-to-secure-nginx-with-let-s-encrypt-on-ubuntu-16-04)>

1. First, ensure that the domain that needs to be configured is publicly reachable (i.e., not blocked by DNS or firewall)
2. **~~$sudo add-apt-repository ppa:certbot/certbot (deprecated)~~**
3. **$sudo apt-get update**
4. sudo apt-get install certbot
5. **$sudo apt-get install** python3-certbot-nginx
6. **$sudo certbot --nginx -d www.captionify.com** #for production

OR

**$sudo certbot --nginx -d stage.captionify.com** #for stage

1. When prompted about redirection for HTTPS, select "**No Redirect**" option as we're going to set up redirection later ourselves.
2. **$sudo openssl dhparam -out /etc/ssl/certs/dhparam.pem 2048**
3. Add this line anywhere in the **'server'** block of **/etc/nginx/sites-available/captionify.com** -> **ssl\_dhparam /etc/ssl/certs/dhparam.pem;**

Note: Remove any existing setting for ssl\_dhparam, for instance something like this:

ssl\_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

1. **$sudo systemctl reload nginx** #reload Nginx
2. Setup cron job to auto-renew the certificate, if needed
   1. **$sudo crontab -e** #edit crontab file in editor
   2. Paste the following at the end of the crontab file => **15 3 \* \* \* /usr/bin/certbot renew –quiet**
3. **Important: Make sure http traffic is ENABLED for the security group on AWS, otherwise when users type in ‘captionify.com’ or ‘stage.captionify.com’ they will not be redirected to the https URL**
4. **Refer the “end status” of the file (next step). The redirection of http to https is being taken care by nginx in the first server section. Make sure to keep it intact.**
5. **In the end the /etc/nginx/sites-available/captionify.com should look like this:**

**For Staging:**

server {

        listen 80;

        server\_name stage.captionify.com;

        return 301 https://stage.captionify.com$request\_uri;

}

server {

    #listen 80;

    server\_name stage.captionify.com;

    location / {

        proxy\_set\_header   X-Forwarded-For $remote\_addr;

        proxy\_set\_header   Host $http\_host;

        proxy\_pass         "http://127.0.0.1:8080";

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

        proxy\_set\_header X-Forwarded-Proto $scheme;

client\_max\_body\_size 200M;

    }

    listen 443 ssl; # managed by Certbot

    ssl\_certificate /etc/letsencrypt/live/stage.captionify.com/fullchain.pem; # managed by Certbot

    ssl\_certificate\_key /etc/letsencrypt/live/stage.captionify.com/privkey.pem; # managed by Certbot

    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot

        ssl\_dhparam /etc/ssl/certs/dhparam.pem;

}

**for Production:**

server {

        listen 80;

        server\_name captionify.com www.captionify.com;

        return 301 https://www.captionify.com$request\_uri;

}

server {

    #listen 80;

    server\_name captionify.com www.captionify.com;

    location / {

        proxy\_set\_header   X-Forwarded-For $remote\_addr;

        proxy\_set\_header   Host $http\_host;

        proxy\_pass         "http://127.0.0.1:8080";

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

        proxy\_set\_header X-Forwarded-Proto $scheme;

client\_max\_body\_size 200M;

    }

    listen 443 ssl; # managed by Certbot

    ssl\_certificate /etc/letsencrypt/live/www.captionify.com/fullchain.pem; # managed by Certbot

    ssl\_certificate\_key /etc/letsencrypt/live/www.captionify.com/privkey.pem; # managed by Certbot

    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot

        ssl\_dhparam /etc/ssl/certs/dhparam.pem;

}

Configure Log Location for Nginx

Configure the location where Nginx would write the access and error logs by following the below steps:

1. Edit the Nginx configuration file: /etc/nginx/nginx.conf and look for lines that have access\_log and error\_log being set. Example:

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

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

1. Modify these values to the values below:

**access\_log /var/www/loadbalancer/data/log/access.log;**

**error\_log /var/www/loadbalancer/data/log/error.log;**

1. Create the directories and set permissions to make sure Nginx is able to write the logs.

**$ls -l /var/log/nginx/access.log** #find the user who owns this file – typically, ‘www-data

**$sudo mkdir -p /var/www/loadbalancer/data/log**

**$sudo chown -R <userid> /var/www/loadbalancer/data/log**

1. **$sudo service nginx restart** #restart service