Web Server Notes

NOTE: To access the servers, you will need a public and private key. Follow the steps under the section "Creating a Key Pair Using Amazon EC2" in this guide created by Amazon: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html#having-ec2-create-your-key-pair

- For Windows users who use PuTTY, you will need to create a private key in order to connect to the server. To create the private key using PuTTYgen, follow the steps in this guide: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/putty.html
- To access the servers on other operating systems or through other means, you can view the guides Amazon provides here: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AccessingInstances.html

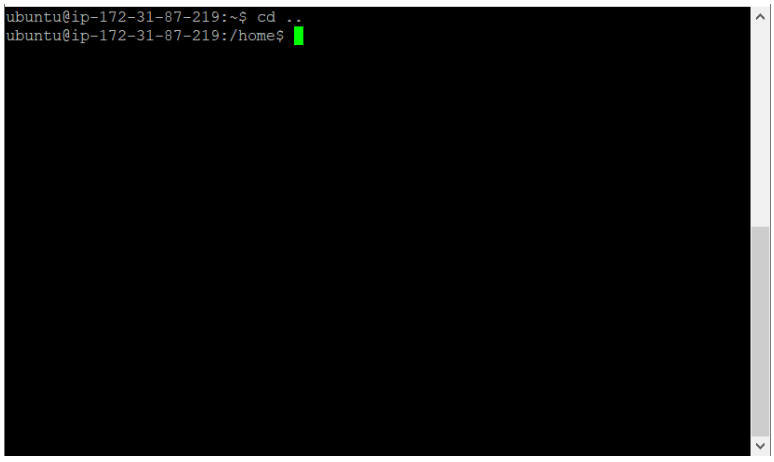
Currently installed:

- Java OpenJDK v11.0.6
- Node v10.19.0
- NPM v6.13.4
- Express v4.16.1
- Python2 v2.7.17
- Python3 v3.6.9
- NGINX v1.14.0

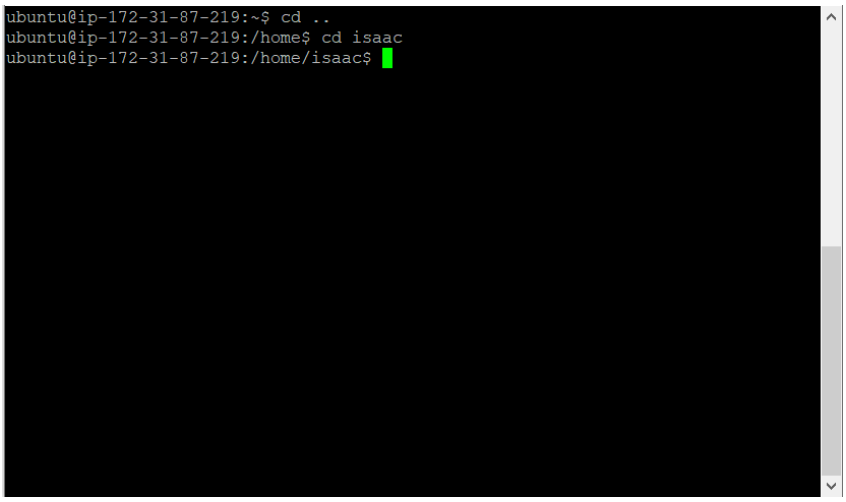
1) Launching the web application

- Log in to the web server with the credentials you have been given or created. You should be in the directory of the user that you logged in as if you were given a login name. If you were not given a login name then you should use “ubuntu” as the login name and then you should be in the ubuntu user directory.

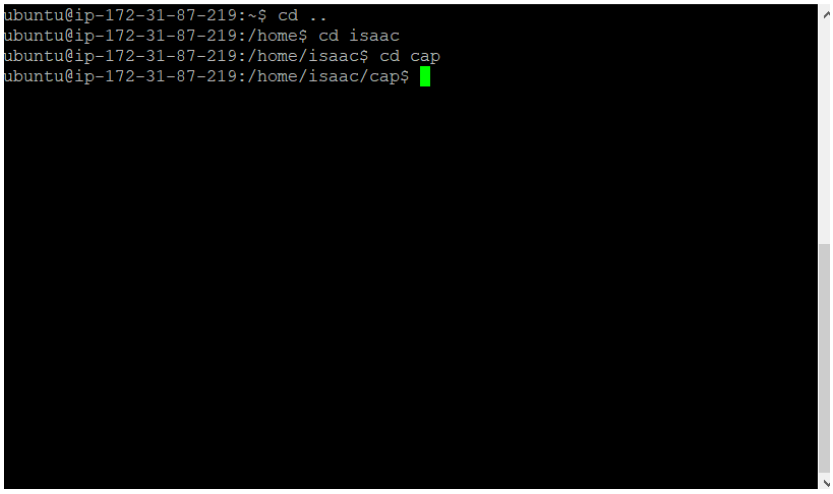
- Enter to the command “cd ..” to go back a directory. You should now be in the home directory.



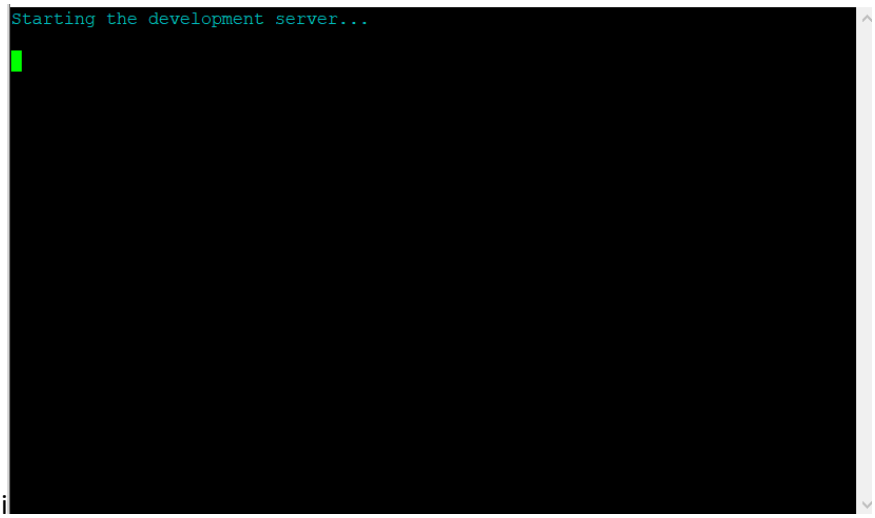
- The structure of the server as of writing this document is that each original group member has their own folder where they put their files in. To access the files for the website, we need to enter the “Isaac” directory. Enter the command “cd isaac”.



- Enter the command “cd cap”. You will now be in the directory that contains the files for the website.



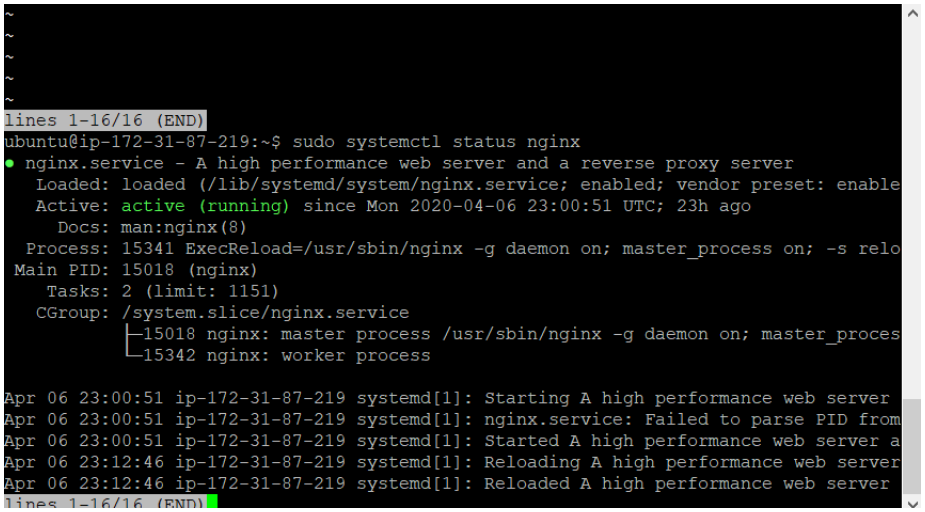
- Enter the command “npm start” to run the development server for the website. You should see this appear in your terminal.



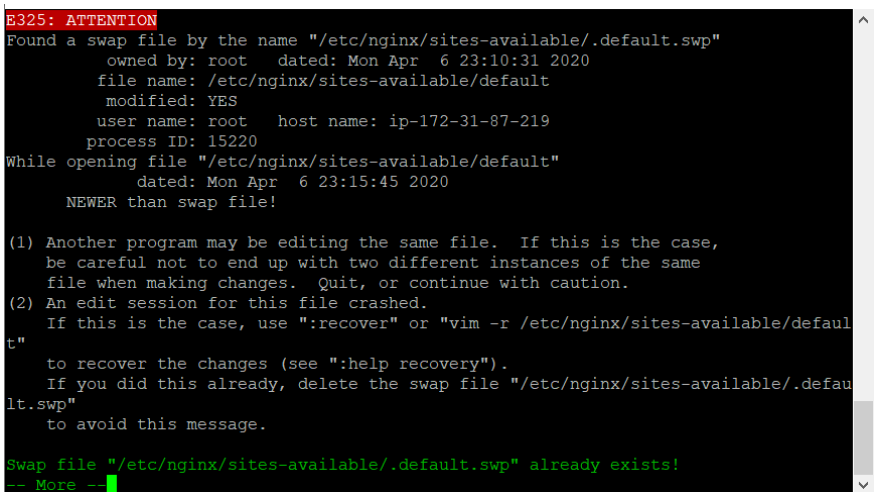
- If you enter in the public DNS/url for the web server (ec2-34-205-156-237.compute-1.amazonaws.com in this example) you should now be able to access and use the website.

2) NGINX commands

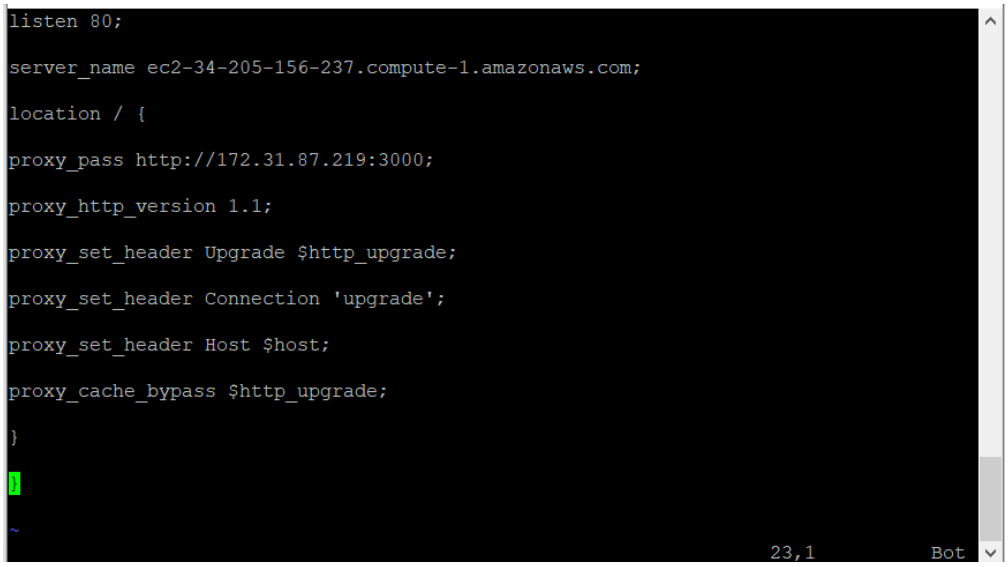
- After logging in to the server from any account, you may want to check the status of NGINX, our reverse proxy server, or modify a config file.
- Enter the command “sudo systemctl status nginx” to check the status of the NGINX server. If it is running properly, your terminal should look like this after running it.



- To tell what port NGINX to listen and redirect to, we need to edit the “sites-available/default” file in NGINX. To access this file, enter the command “sudo vi /etc/nginx/sites-available/default”. If your terminal looks like this after running the command, simply recover the file by pressing “R” then hit enter.



- You should now be able to see the file in the vi editor. The only time this will need to be modified is if the web server instance is stopped in the AWS management console. The three parts of this file that may need to be edited at any point is the “server_name” line and the “proxy_pass” line.
    - For “server_name”, you must use the current public DNS AWS provides for your server.
    - For “proxy_pass”, you need to provide the private IP and the port number that you were using to access the site. In the screenshot below the private IP begins with 172 and the port number used is 3000.
    - To view the private IP, you can go to the EC2 Management Console and in the Description Tab you will be able to see the private IP.

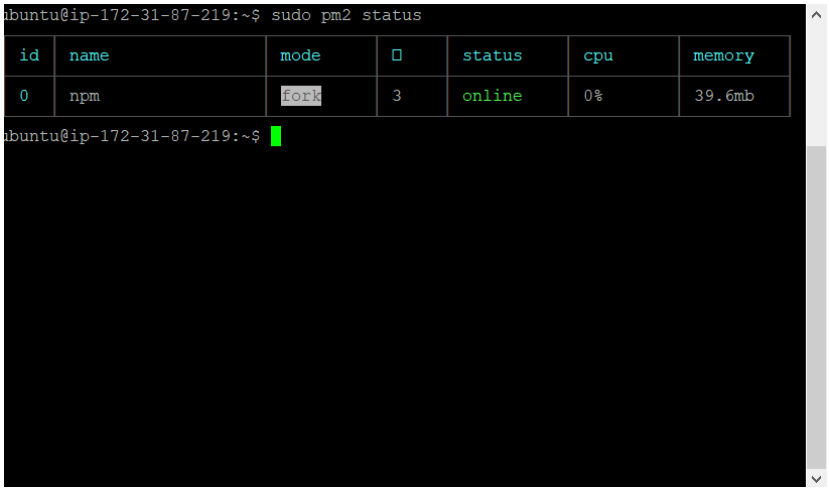


- If you need to edit the file press ‘a’ to enter insert/edit mode. To exit edit mode, press ‘esc’. To exit the file and save press ‘shift + :’ then press ‘wq’ and enter. To exit the file without saving, press ‘shift + :’ then press ’q!’ and enter to leave the file unchanged.

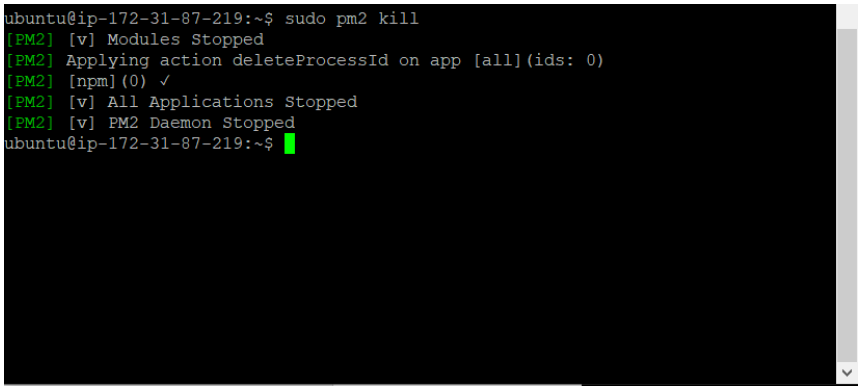
- Currently we have the NGINX server to listen on port 80 as this is the port for HTTP. You can also have it listen on port 443 for HTTPS but only if your server has SSL certificates. Since we do not have a domain name, which is necessary to gain SSL certificates, we will only be listening on port 80. To modify NGINX so it is configured properly to listen on port 443, you can follow this guide. https://nginx.org/en/docs/http/configuring_https_servers.html#single_http_https_server
- If there any issues working with NGINX, you can follow the guide that I followed in setting it up. https://medium.com/tensult/configure-nginx-with-nodejs-application-on-aws-ubuntu-18-04-server-for-different-http-ports-a2f722902032

3) Production Manager 2 (pm2)

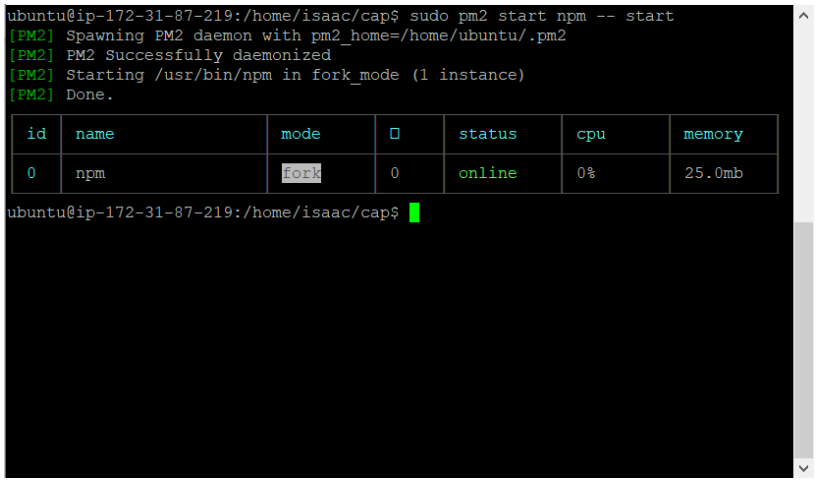
- pm2 is a process manager that allows applications to run 24/7 without having to stay connected to a server or manually launching an application.
- Currently, pm2 is set to have the website always running unless the process is killed. Below are some commands that will be useful for working with pm2.
- Since pm2 has already been set to run automatically, you will only have to start the application after killing the process. To check the status of pm2, and to see what processes are running, enter the command “sudo pm2 status”. You will be able to see all currently running processes.



- To kill a process, enter the command “sudo pm2 kill”. Any running process will now be shutdown.



- Now that the process is no longer running, we will have to manually start it up. Once we start it up, it will continue to run unless we manually kill the process again. We must go to the ‘isaac/cap’ directory to start the application. Unless you are already there, enter the command “cd ..”. You should now be in the home directory.
- Enter the command “cd issac” next.
- Enter the command “cd cap” next.
- Now you should be in the correct directory. Enter the command “sudo pm2 start npm -- start”. Your terminal should now look like this.



- You should now be able to access the website through your browser.