

Jigmeboxinglist – Cloud Server Project Documentation

ICT171

Kuenzang Jigme Dorji

Student ID: 34836458

Project: JigmeBoxinglist – Top 10 Greatest Boxers of All Time

IP Address: 13.210.41.42

Domain: <https://www.jigmeboxinglist.click/>

<https://www.jigmeboxinglist.click/blog/>

GitHub repo: <https://github.com/Jigmedorji12/infrastructure-project>

Video Explainer:

Project Overview:

Infrastructure Project Overview — ICT171

Project Title

"jigmeboxinglist.click" — A Tribute to the Greatest Boxers of All Time

Project Summary

This infrastructure project demonstrates the design, configuration, deployment, and management of a secure, cloud-hosted web server. The centrepiece is a website titled **jigmeboxinglist.click**, dedicated to showcasing the **Top 10 Greatest Boxers of All Time**, presented using HTML, CSS, and JavaScript to highlight educational content, historic data, and motivational quotes.

In addition to static web development, this project integrates WordPress as dynamic content management systems under specific routes (e.g., /blog), and deploys a VPN server to demonstrate broader server capabilities. Security was implemented through SSL/TLS encryption via HTTPS, and domain name configuration was managed through AWS Route 53.

All code and configuration are managed via **GitHub version control**, and custom scripting was developed to automate and maintain key parts of the infrastructure.

Learning Outcomes Demonstrated

- Proficiency with Linux CLI (Ubuntu server environment)
- Manual configuration of Apache web server, VPN, and CMS platforms
- GitHub usage for collaborative version control and project tracking
- Practical understanding of server-side scripting and automation
- Implementation of IaaS with domain binding and TLS encryption
- System documentation and structured project planning

Project Timeline

Week

Tasks Completed

Week 1 EC2 setup and Route 53 domain configuration

Week 2 Installed WordPress (/blog)

Week 3 Configured OpenVPN server

Week 4 Enabled SSL/TLS (HTTPS), committed to GitHub, finalized documentation

Server Setup step (Amazon EC2, Ubuntu)

1. Launch a New EC2 Instance

Go to the AWS EC2 Dashboard and click "Launch Instance". Choose Ubuntu 20.04 LTS and make sure it's Free Tier Eligible.

2. Choose Instance Type

Select the default t2.micro instance type (free tier eligible), then click Next.

3. Configure Instance & Storage

Leave the instance settings as default.

On the Add Storage step, accept the default 8GB — that's enough for basic use.

4. Skip Tags and Set Up Security Group

Skip the "Add Tags" section.

In Configure Security Group:

- Name it: ssh-and-web
- Keep the default SSH rule.
- Click "Add Rule", choose HTTP, and leave source as Anywhere (0.0.0.0/0) to allow web traffic.

5. Review and Launch

Review your settings. AWS may warn that the server is open to the world — this is fine for a public website.

Click Launch.

6. Create Key Pair and Final Launch

When prompted, create a new key pair (e.g., webserver-key).

Download the .pem file and keep it safe — you'll need it to access your server.

Click Launch Instance, then View Instance to monitor its status.

Launch an instance

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags

Name: TEST

Application and OS Images (Amazon Machine Image)

Search our full catalog including 1000s of application and OS images

Recent: Quick Start

Amazon Linux, macOS, Ubuntu, Windows, Red Hat, SUSE Linux, Debian

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type
ami-06c19207c1ab181f0 (64-bit x86) / ami-077b8e41b5194f27 (64-bit ARM)
Virtualization: hvm, ENA enabled: true, Root device type: ebs

Free tier eligible

Description

Ubuntu Server 24.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).
Canonical, Ubuntu, 24.04, amd64 noble image

Architecture: 64-bit (x86) AMI ID: ami-06c19207c1ab181f0 Publish Date: 2025-05-30 Username: ubuntu

Instance type

t2.micro
Family: t2, 1 vCPU, 1 GB Memory, Current generation: true, On-Demand SUSE base pricing: 0.0146 USD per Hour, On-Demand Linux base pricing: 0.0146 USD per Hour, On-Demand Windows base pricing: 0.0192 USD per Hour, On-Demand RHEL base pricing: 0.0228 USD per Hour, On-Demand Ubuntu Pro base pricing: 0.0164 USD per Hour

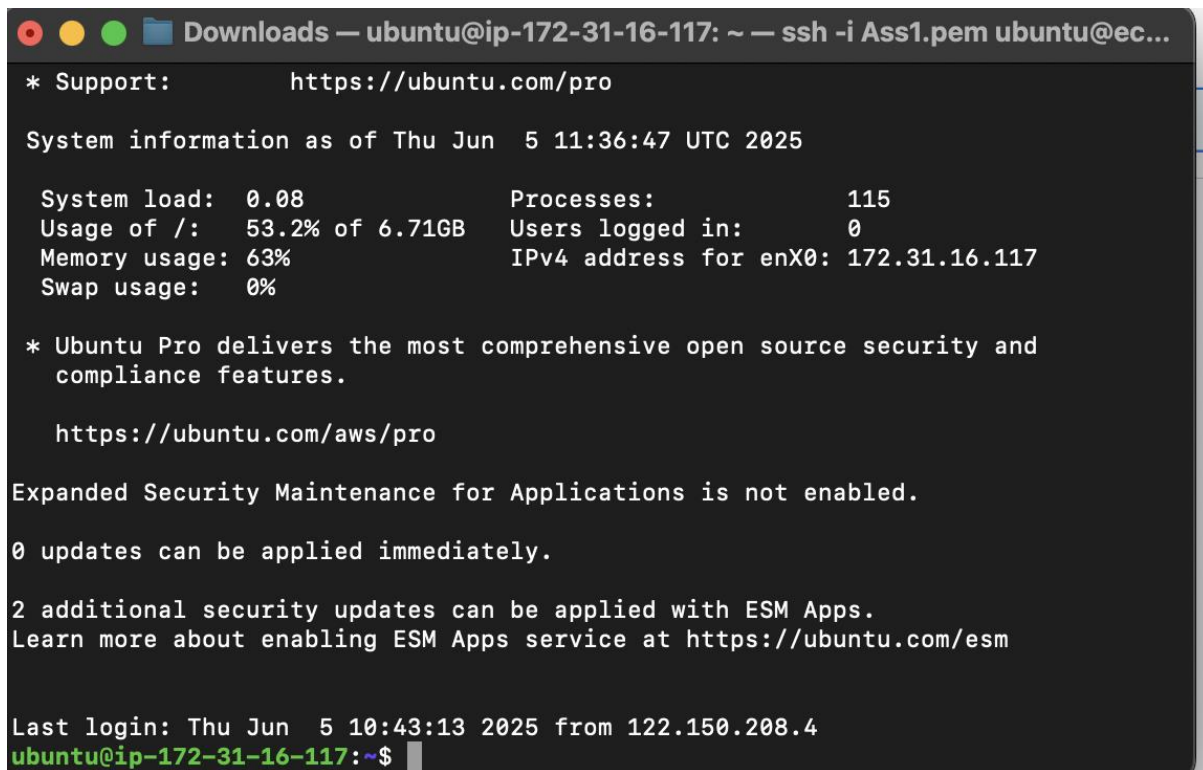
Free tier eligible

All generations

Key pair (login)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

7. Go onto your command line like terminal and enter: `ssh -i "yourkeyname.pem" ubuntu@ec2-12-123-1-35.ap-southwest-5.compute.amazonaws.com`



```
Downloads — ubuntu@ip-172-31-16-117: ~ — ssh -i Ass1.pem ubuntu@ec...
* Support:                https://ubuntu.com/pro

System information as of Thu Jun  5 11:36:47 UTC 2025

System load:  0.08          Processes:            115
Usage of /:   53.2% of 6.71GB Users logged in:          0
Memory usage: 63%          IPv4 address for enX0: 172.31.16.117
Swap usage:   0%

* Ubuntu Pro delivers the most comprehensive open source security and
  compliance features.

  https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

2 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

Last login: Thu Jun  5 10:43:13 2025 from 122.150.208.4
ubuntu@ip-172-31-16-117:~$
```

8. You then Install Apache by entering this command:

`Sudo Apt update`

`Sudo apt install apache2`

9. Test this by visiting your webserver by entering your IP into the web browser

Domain Name and DNS (Route 53)

1. On the AWS console search up Route 53
2. On the left side go on domains section and click on registered domains
3. Once you are in, click on register domains
4. Enter a domain name that suits your topic (use .click for the cheapest domain) and select the domain

Register domains [Info](#)

Pricing for domain names varies by top-level domain (TLD). For more information, view [price with different TLDs](#).

Search for domain

Check availability for a domain

Q boxing1234.click X Search

► **Standard pricing**

Pricing for domain names varies by top-level domain (TLD), such as .com or .org.

Search result

Domain	Price/year	Actions
boxing1234.click Exact match	3.00 USD Renews at 3.00 USD	Select

5. Proceed to check out and fill in your information and once it is done wait 10-20 minutes to show up on the register domains section
6. On the left section, select hosted zones, click on your new domains and click on view details, once your in click on create records.
7. Under record name use “www”, under value type in your website public ip address, make sure it a A type record, you can keep the rest on default and click. Create.

Quick create record [Switch to wizard](#)

▼ Record 1 Delete

Record name [Info](#) **Record type** [Info](#)

www .jigmeboxinglist.click A – Routes traffic to an IPv4 address and some... ▼

Keep blank to create a record for the root domain.

☐ Alias

Value [Info](#)

192.168.1.2

Enter multiple values on separate lines.

TTL (seconds) [Info](#) **Routing policy** [Info](#)

300 1m 1h 1d Simple routing ▼

Recommended values: 60 to 172800 (two days)

Add another record

SSL/TLS Certbot

1. Install Certbot, on the command link enter:

```
ssh -i your-key.pem ubuntu@13.210.41.42
```

```
sudo apt update
```

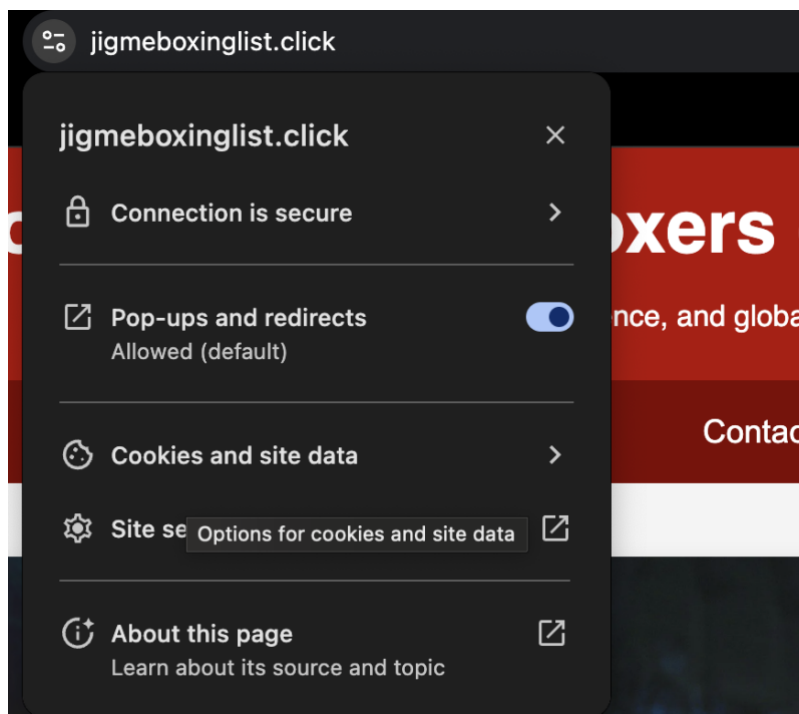
```
sudo apt install certbot python3-certbot-nginx -y
```

```
sudo certbot --nginx -d yourdomain.com -d www.yourdomain.com
```

```
Downloads — ubuntu@ip-172-31-16-117: ~ — ssh -i Ass1.pem ubuntu@ec...
Building dependency tree... Done
Reading state information... Done
certbot is already the newest version (2.9.0-1).
python3-certbot-apache is already the newest version (2.9.0-1).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
ubuntu@ip-172-31-16-117:~$ sudo certbot -apache
usage:
  certbot [SUBCOMMAND] [options] [-d DOMAIN] [-d DOMAIN] ...

Certbot can obtain and install HTTPS/TLS/SSL certificates.  By default,
it will attempt to use a webserver both for obtaining and installing the
certificate.
certbot: error: unrecognized arguments: -apache
ubuntu@ip-172-31-16-117:~$
ubuntu@ip-172-31-16-117:~$ sudo apt install certbot python3-certbot-nginx -y
sudo certbot --nginx -d yourdomain.com -d www.yourdomain.com
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
certbot is already the newest version (2.9.0-1).
python3-certbot-nginx is already the newest version (2.9.0-1).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Requesting a certificate for yourdomain.com and www.yourdomain.com
```

9. Check your website it should now be secure and your domain working



ELASTIC IP

1. ON the EC2 section under network and security click on elastic Ips
2. select Allocate Elastic IP address, you can leave the rest defaults then click allocate
3. once it been made select the elastic IP you made
4. click on action and select associated elastic IP address
5. choose the your web server on the instance and select the private IP address

EC2 > Elastic IP addresses > Associate Elastic IP address

Associate Elastic IP address [info](#)

Choose the instance or network interface to associate to this Elastic IP address (54.253.186.102)

Elastic IP address: 54.253.186.102

Resource type
Choose the type of resource with which to associate the Elastic IP address.

☒ Instance
☐ Network interface

Warning If you associate an Elastic IP address with an instance that already has an Elastic IP address associated, the previously associated Elastic IP address will be disassociated, but the add [more](#)

If no private IP address is specified, the Elastic IP address will be associated with the primary private IP address.

Instance
I-Oc82b78c0adfb8f36

Private IP address
The private IP address with which to associate the Elastic IP address.
172.31.21.99

Reassociation
Specify whether the Elastic IP address can be reassociated with a different resource if it already associated with a resource.
☐ Allow this Elastic IP address to be reassociated

SSH

1. Make sure your Key pair your created is in your downloads
2. Then on your command link use:
Cd Downloads (once you are in downloads then enter this:
ssh -i "yourkeyname.pem" [ubuntu@ec2-12-123-1-35.ap-southwest-5.compute.amazonaws.com](#)
3. You will see you are now connect through SSH

```
Last login: Thu Jun  5 11:36:48 2025 from 122.150.208.4
ubuntu@ip-172-31-16-117:~$
```

Website Implementation

1. on the command link go:

ssh -i your-key.pem ubuntu@13.210.41.42

2. once you are in your web server

3. to modify your website enter this command line:

Sudo nano /var/www/html/index.html and copy your HTML/CSS script into it

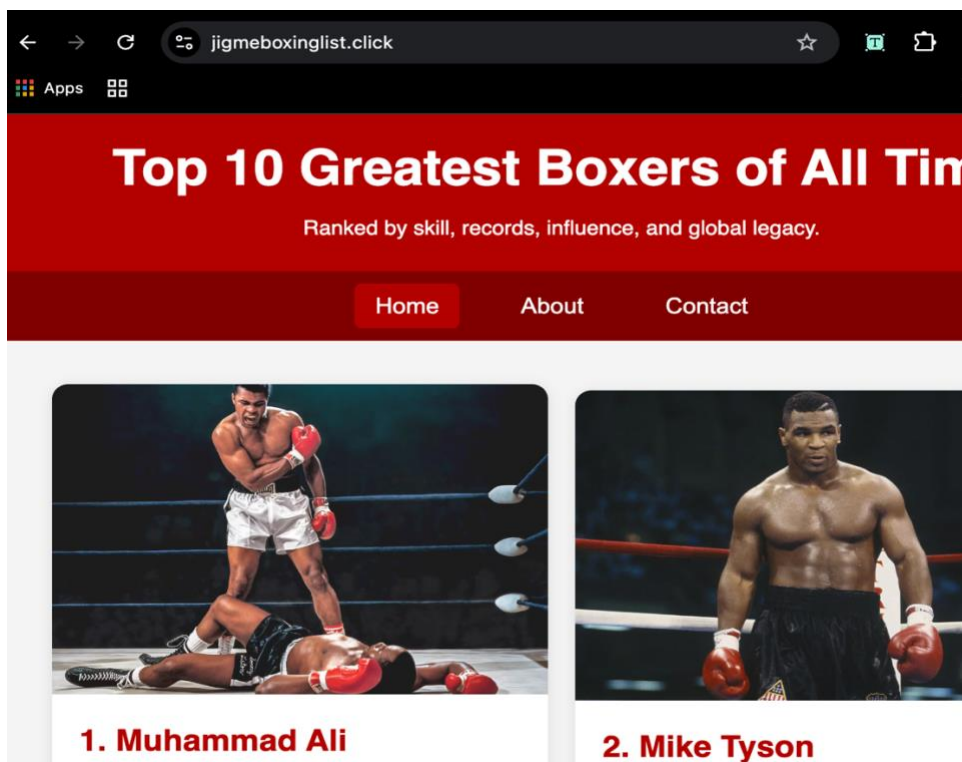
```
GNU nano 7.2 /var/www/html/index.html
</head>
<body>

<header>
  <h1>Top 10 Greatest Boxers of All Time</h1>
  <div id="clock"></div>
  <p>Ranked by skill, records, influence, and global legacy.</p>
</header>

<nav>
  <button id="nav-home" class="active" onclick="showPage('home')">Home</button>
  <button id="nav-about" onclick="showPage('about')">About</button>
  <button id="nav-contact" onclick="showPage('contact')">Contact</button>
</nav>

<!-- Home page (default visible) -->
<div id="home" class="page">
  <div class="container">

    <!-- Muhammad Ali -->
```



WordPress

1. Install WordPress on EC2 (Short Steps)

Connect to EC2:

```
ssh -i "Ass1.pem" ubuntu@your-ec2-address
```

2. Install Software:

```
sudo apt update
```

```
sudo apt install apache2 php libapache2-mod-php php-mysql mysql-server unzip wget -y
```

3. Download WordPress

```
cd /tmp
```

```
wget https://wordpress.org/latest.zip
```

```
unzip latest.zip
```

```
ubuntu@ip-172-31-16-117:~$ cd /tmp
wget https://wordpress.org/latest.zip
unzip latest.zip
sudo mv wordpress /var/www/html/blog
--2025-06-06 04:43:25-- https://wordpress.org/latest.zip
Resolving wordpress.org (wordpress.org)... 198.143.164.252
Connecting to wordpress.org (wordpress.org)|198.143.164.252|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 28551696 (27M) [application/zip]
Saving to: 'latest.zip'
```

```
latest.zip          10%[=>                ] 2.84M  152KB/s  eta 2m 46s
```

```
sudo mv wordpress /var/www/html/blog
```

4. Set Permissions

```
sudo chown -R www-data:www-data /var/www/html/blog
```

```
sudo chmod -R 755 /var/www/html/blog
```

5. Create MySQL Database

```
CREATE DATABASE wordpress_blog;
```

```
CREATE USER 'wpuser'@'localhost' IDENTIFIED BY 'StrongPass123!';
```

```
GRANT ALL PRIVILEGES ON wordpress_blog.* TO 'wpuser'@'localhost';
```

```
FLUSH PRIVILEGES;
```

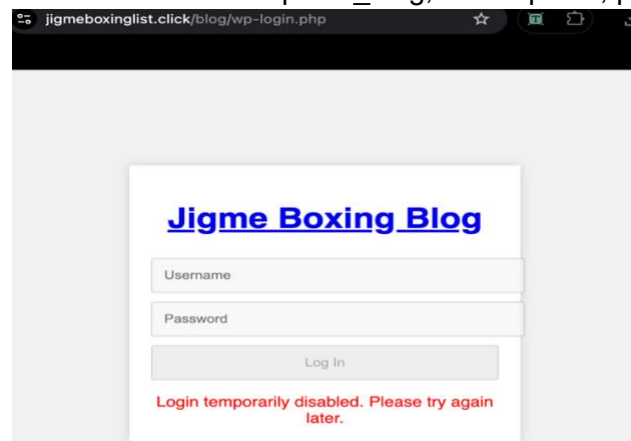
6. Restart Apache:

```
sudo systemctl restart apache2
```

7. Run WordPress Setup in Browser

🔗 <https://www.jigmeboxinglist.click/blog/>

Fill in DB name `wordpress_blog`, user `wpuser`, password `StrongPass123!`

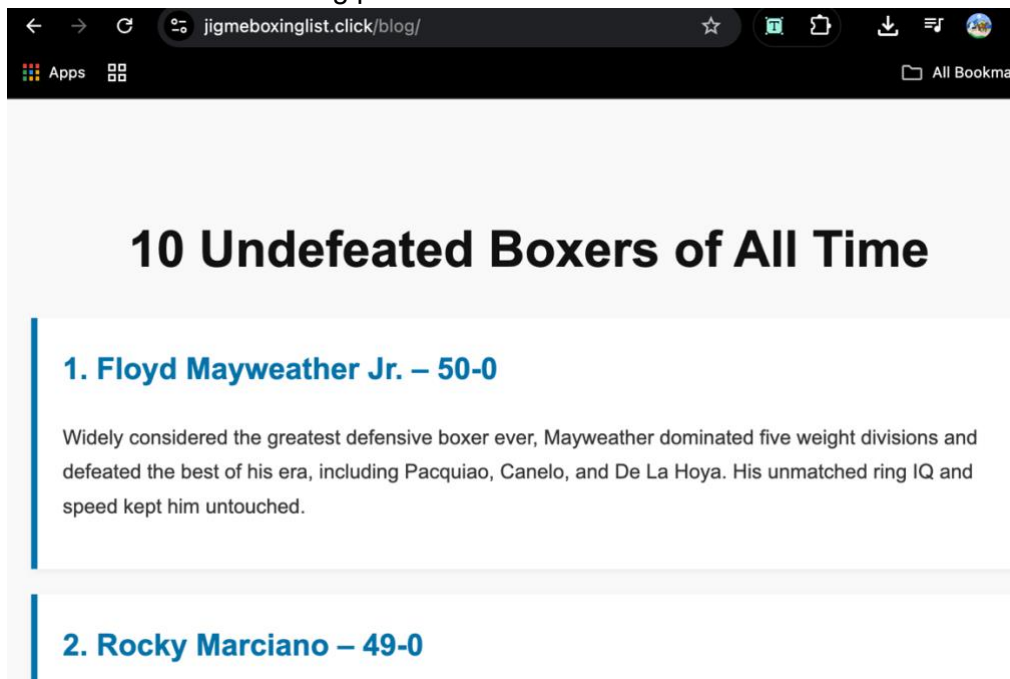


8. After Setup – Customize Your Blog

Login to Admin Panel

<https://www.jigmeboxinglist.click/blog/wp-admin>

9. You can create the Blog post



VPN

1. Update & Install Packages:

```
sudo apt update && sudo apt upgrade -y  
sudo apt install openvpn easy-rsa -y
```

2. Set Up Easy-RSA PKI

```
make-cadir ~/openvpn-ca  
cd ~/openvpn-ca  
nano vars # Edit values: country, city, org
```

3. Build CA & Server Certificates

```
./easyrsa init-pki  
./easyrsa build-ca  
./easyrsa gen-req server nopass  
./easyrsa sign-req server server  
./easyrsa gen-dh  
openvpn --genkey --secret ta.key
```

4. Create Client Certificate

```
./easyrsa gen-req jigme-client nopass  
./easyrsa sign-req client jigme-client
```

5. Configure Server

```
sudo gunzip -c /usr/share/doc/openvpn/examples/sample-config-files/server.conf.gz | sudo tee  
/etc/openvpn/server.conf
```

```
sudo nano /etc/openvpn/server.conf
```

Make sure these lines are enabled:

tls-auth, cipher, auth, user nobody, group nogroup

6. Copy Certs & Keys to OpenVPN

```
sudo cp ~/openvpn-ca/pki/ca.crt /etc/openvpn/  
sudo cp ~/openvpn-ca/pki/issued/server.crt /etc/openvpn/  
sudo cp ~/openvpn-ca/pki/private/server.key /etc/openvpn/  
sudo cp ~/openvpn-ca/pki/dh.pem /etc/openvpn/  
sudo cp ~/openvpn-ca/ta.key /etc/openvpn/
```

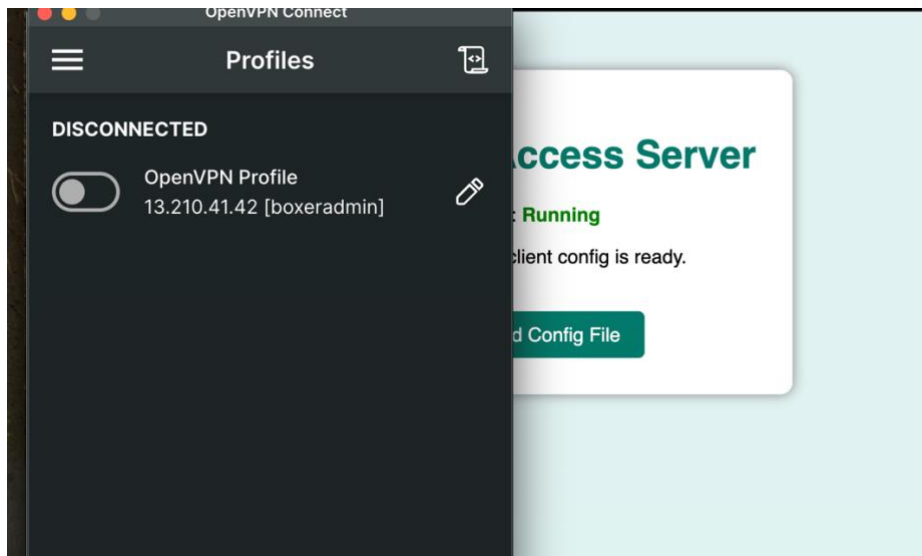
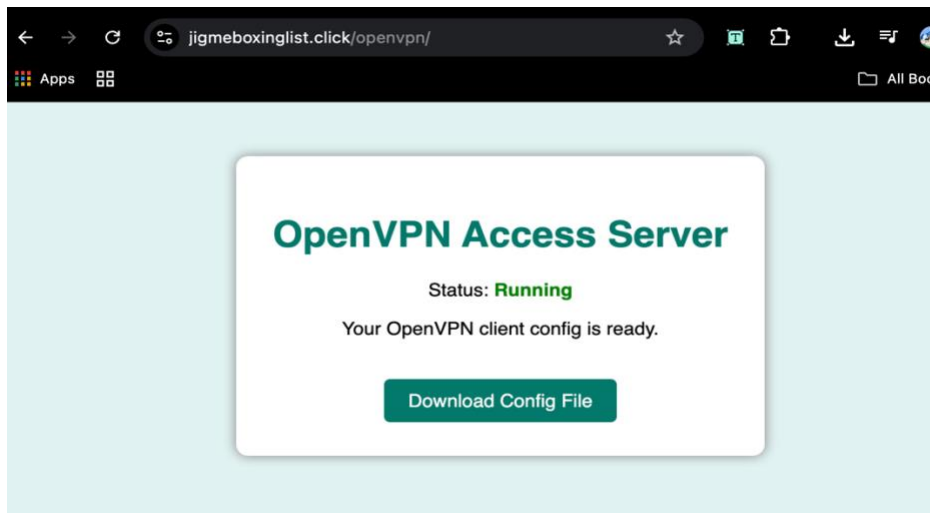
7. Enable IP Forwarding

```
sudo nano /etc/sysctl.conf  
# Un-comment: net.ipv4.ip_forward=1
```

```
sudo sysctl -p
```

9. Start OpenVPN

```
sudo systemctl start openvpn@server  
sudo systemctl enable openvpn@server  
sudo systemctl status openvpn@server
```



Script

Purpose: Checks server disk usage and logs warning if usage exceeds 80%

On the command line go:

Cd ~

nano disk_check.sh

enter:

```
#!/bin/bash
```

```
# Log the current date
```

```
date >> /var/log/disk_check.log
```

```
# Check disk usage
```

```
USAGE=$(df -h / | awk 'NR==2 {print $5}' | sed 's/%//')
```

```
# Log status
```

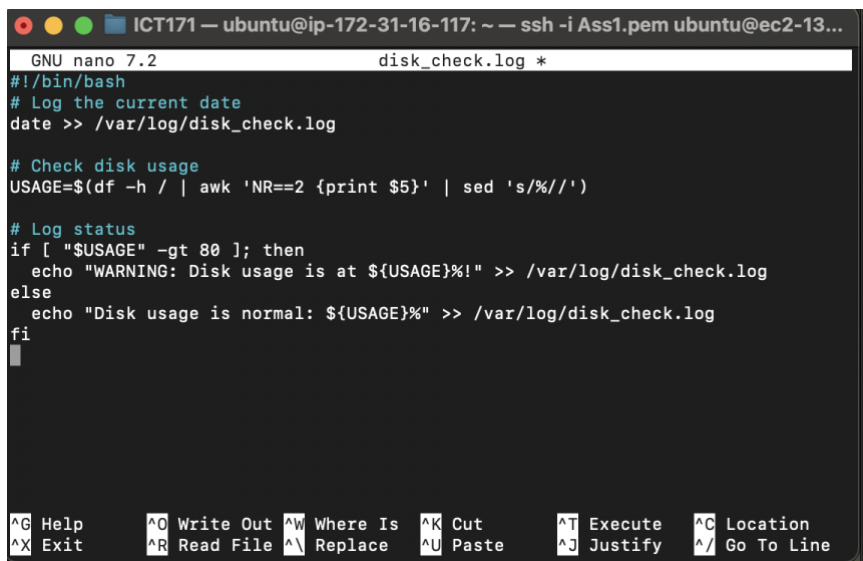
```
if [ "$USAGE" -gt 80 ]; then
```

```
    echo "WARNING: Disk usage is at ${USAGE}%" >> /var/log/disk_check.log
```

```
else
```

```
    echo "Disk usage is normal: ${USAGE}%" >> /var/log/disk_check.log
```

```
fi
```



```
GNU nano 7.2 disk_check.log *
#!/bin/bash
# Log the current date
date >> /var/log/disk_check.log

# Check disk usage
USAGE=$(df -h / | awk 'NR==2 {print $5}' | sed 's/%//')

# Log status
if [ "$USAGE" -gt 80 ]; then
    echo "WARNING: Disk usage is at ${USAGE}%" >> /var/log/disk_check.log
else
    echo "Disk usage is normal: ${USAGE}%" >> /var/log/disk_check.log
fi
```

This Monitors your server's root disk space. Logs a warning message if usage exceeds 80%.

Run this command line:

```
crontab -e
```

Add this line to check every hour:

```
0 * * * * /bin/bash /path/to/disk-check.sh
```

This Runs every hour and logs disk space info. Helps catch low storage issues before they cause problems.

Sudo cat /var/log/disk_check.log

```
ubuntu@ip-172-31-16-117:~$ sudo cat /var/log/disk_check.log
Fri Jun  6 10:21:49 UTC 2025
Disk usage is normal: 58%
ubuntu@ip-172-31-16-117:~$
```

ubuntu@ip-172-31-16-117: ~ — ssh -i Ass1.p
ubuntu@ec2-13-210-41-42.ap-