Assignment – ICT171

35577001

Kyle Mckenzie

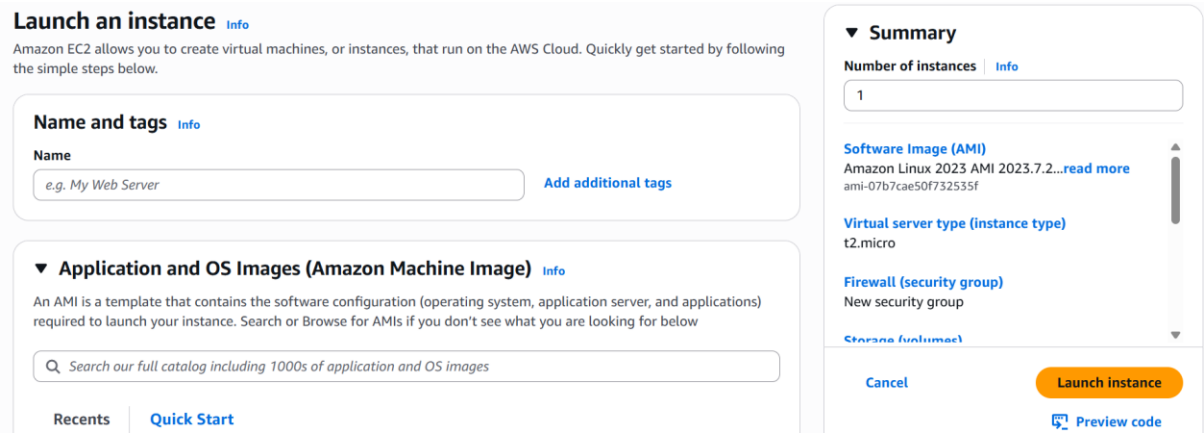
DNS name: http://www.kylemurdochproject.com/

IP address(13.210.1.190): http://13.210.1.190/

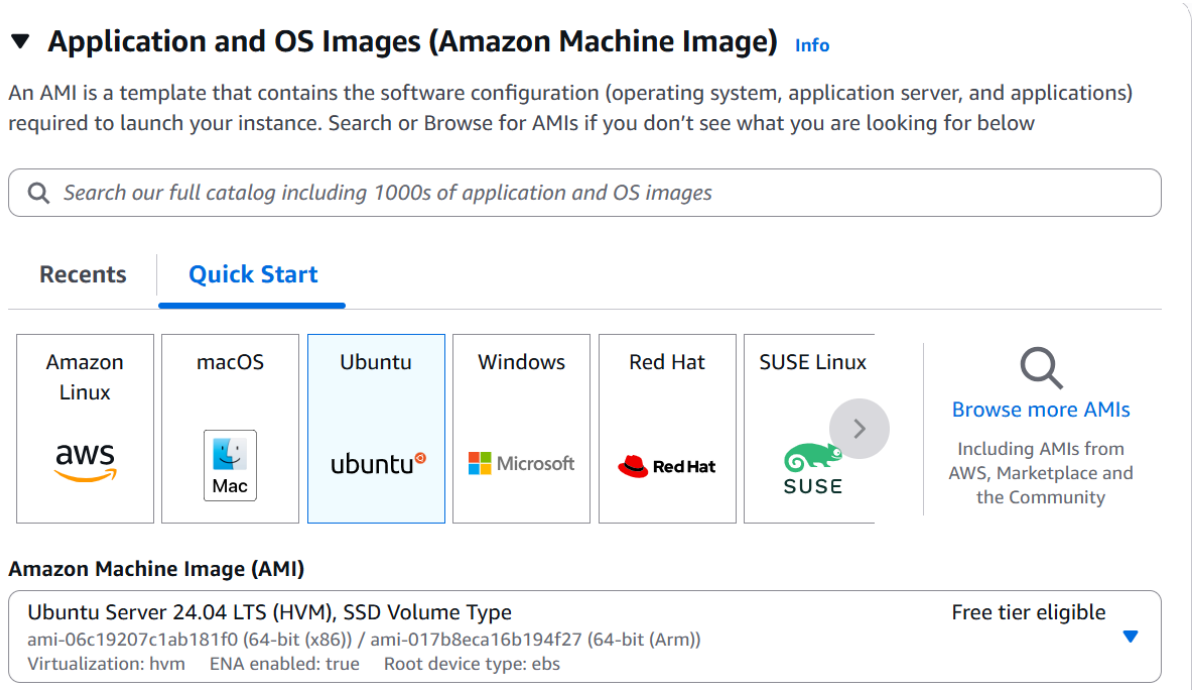
This entire process was created on the AWS EC2 free tier. Once signed in and on the main page of instances, complete the following steps.

## Server set up:

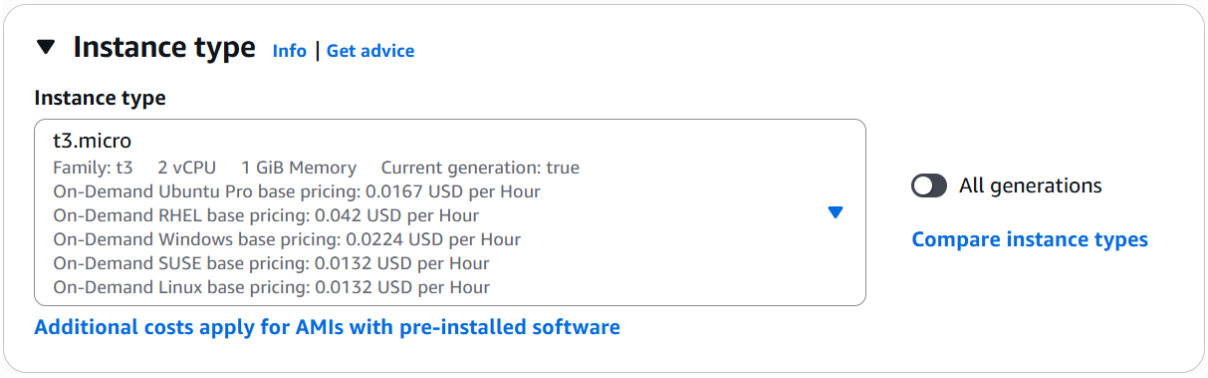
*Create and launch a new instance, this is what the entire assignment will run on:*



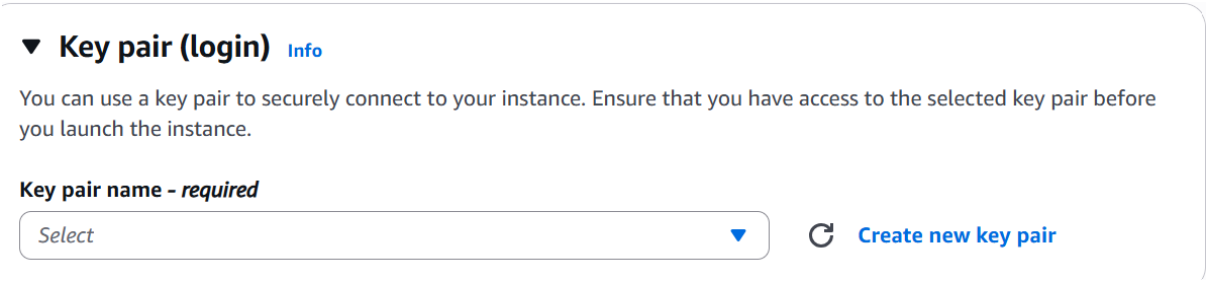

*Now, set the OS to Ubuntu, as it will be more beneficial when setting up the VPN later*

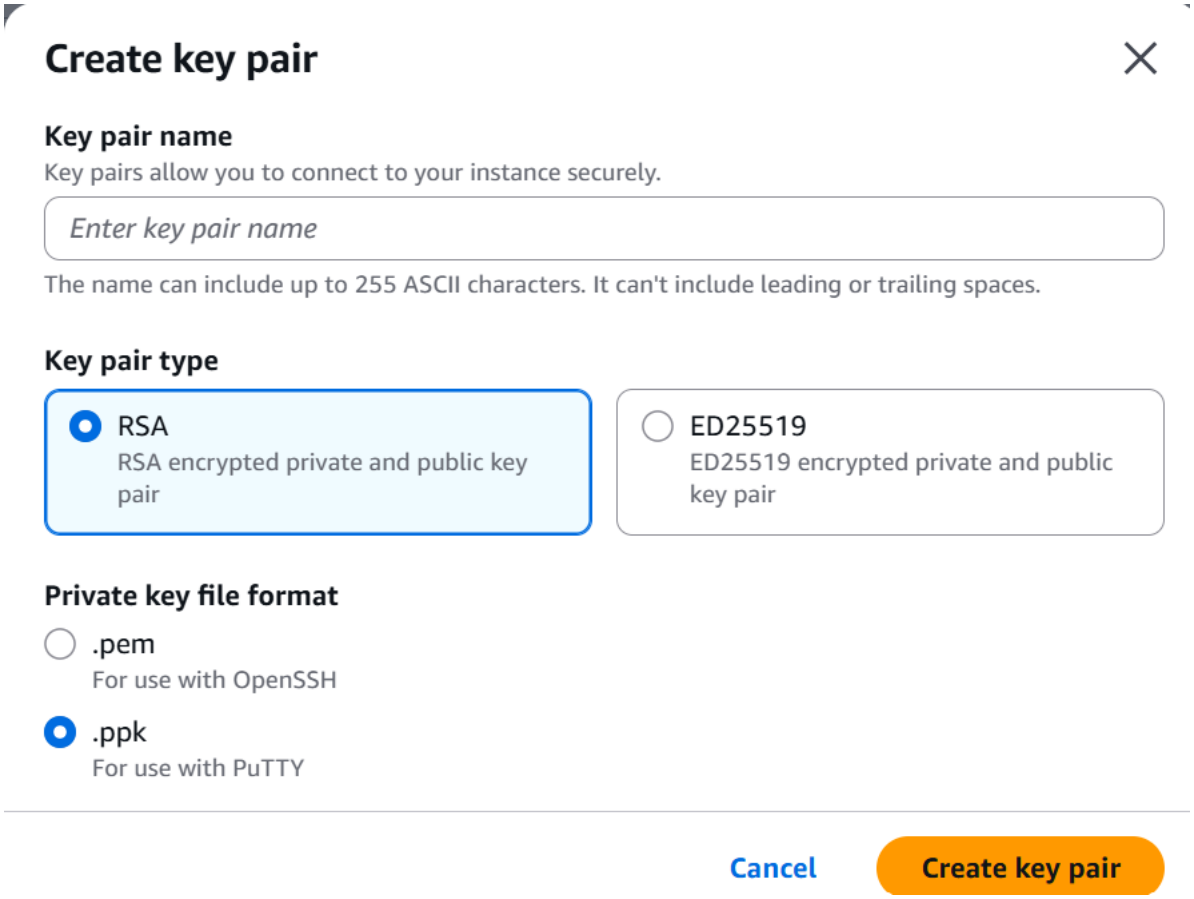


*Set instance type to t3.micro, this is still within the free tier*

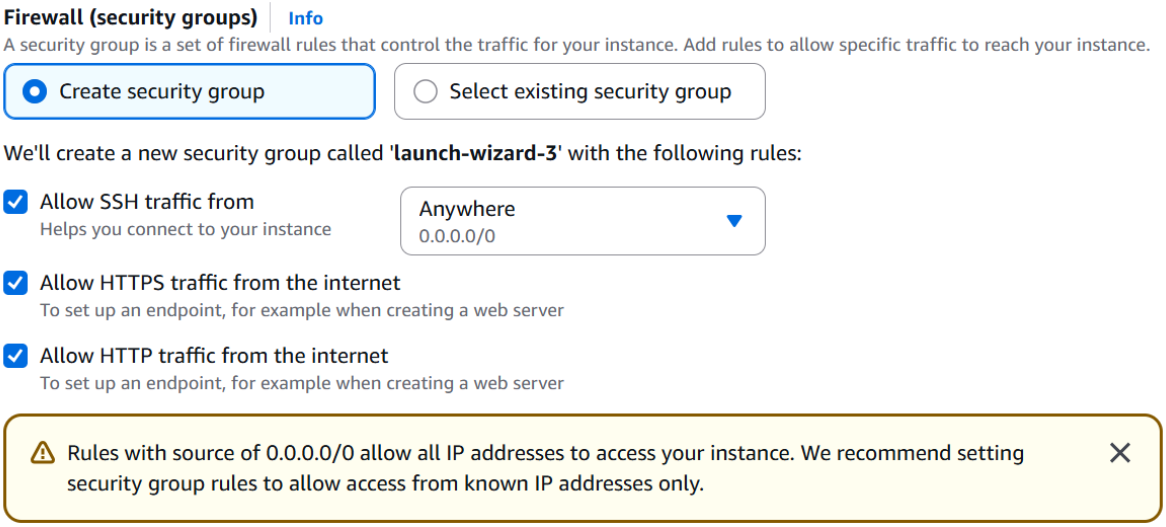



*Create a new key pair, this is how we will access the back end later*

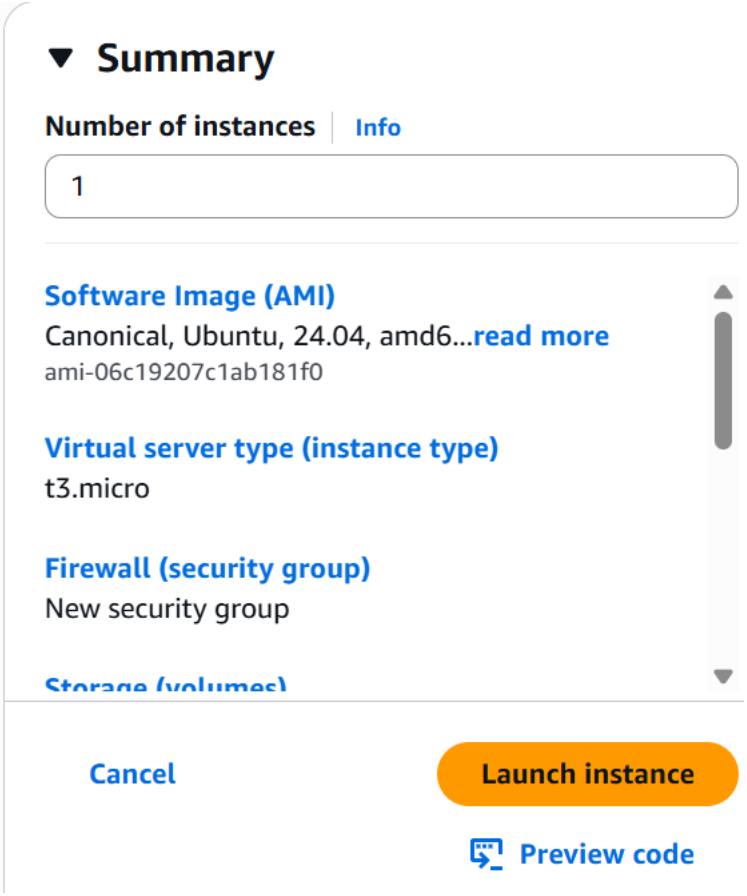


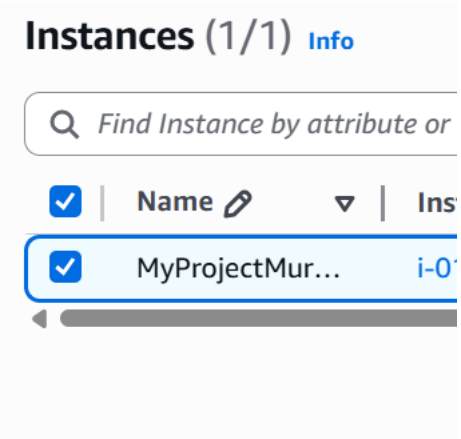
*Create a security group and change the settings to allow http/https*




*Launch the instance*

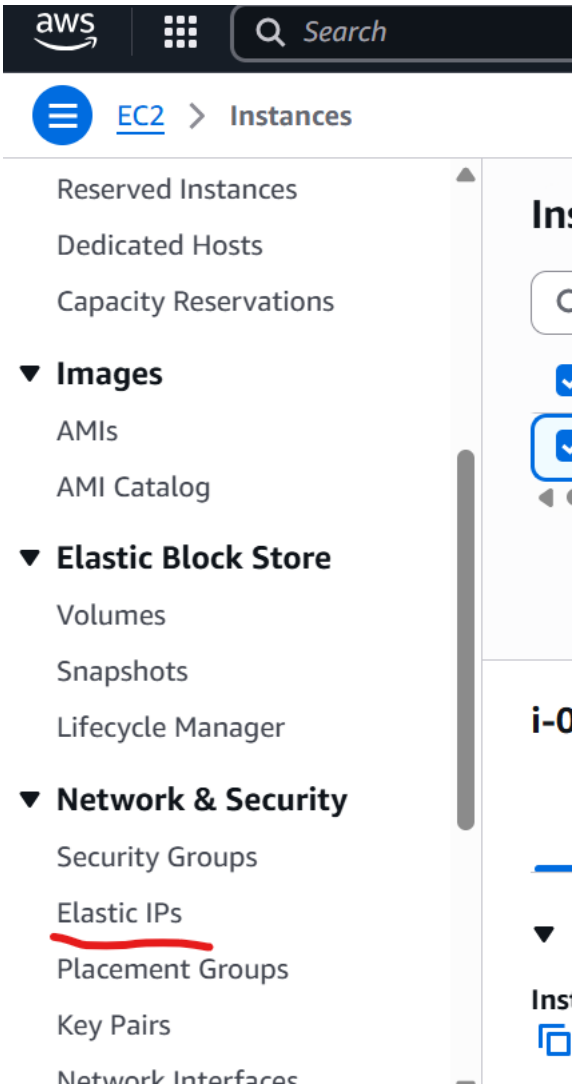


*Name the instance (optional)*

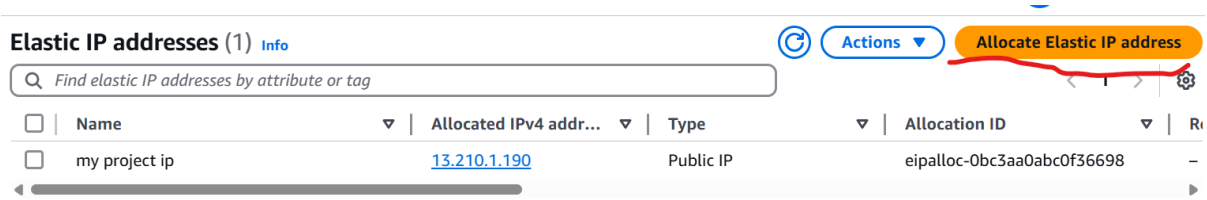


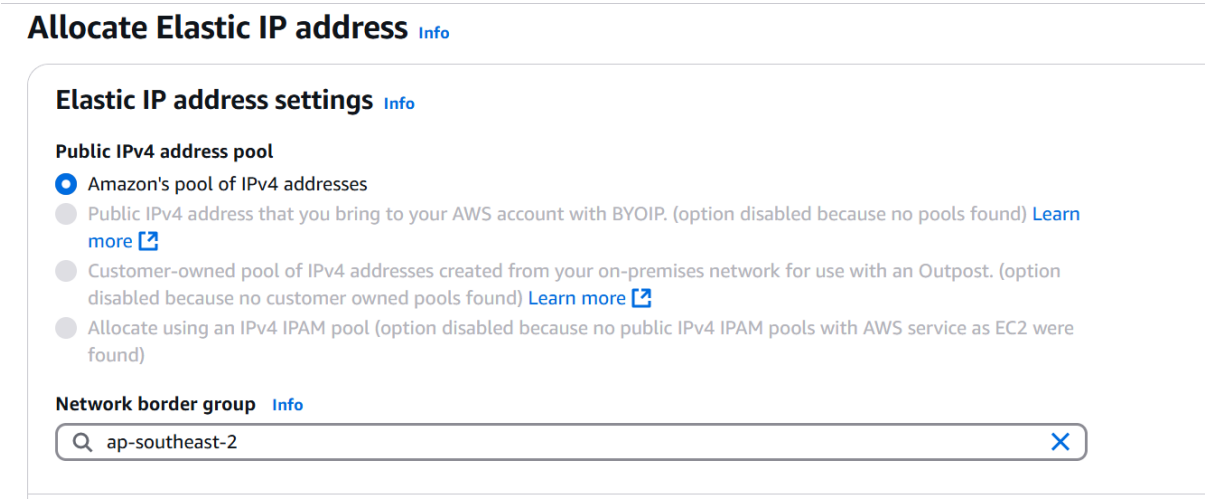

**Assign it an Elastic IP:**

*Scroll down the menu bar and open "Elastic IP"; this is where we can allocate an Elastic IP to the instance we just launched.*

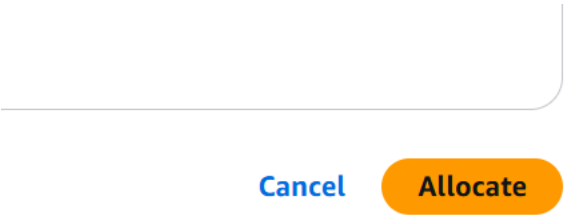


*Click on “Allocate Elastic IP address”, the orange button*

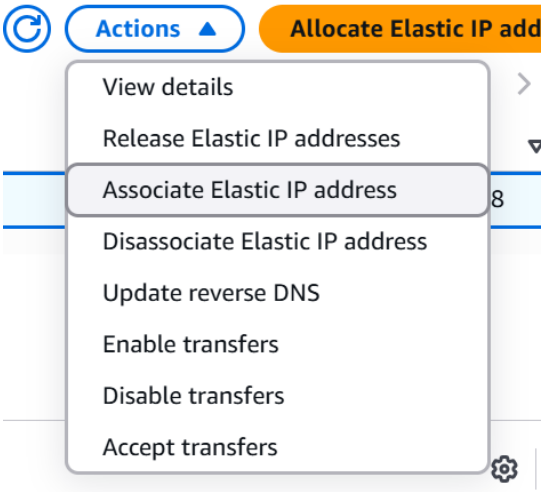



*Scroll down and click “Allocate”*

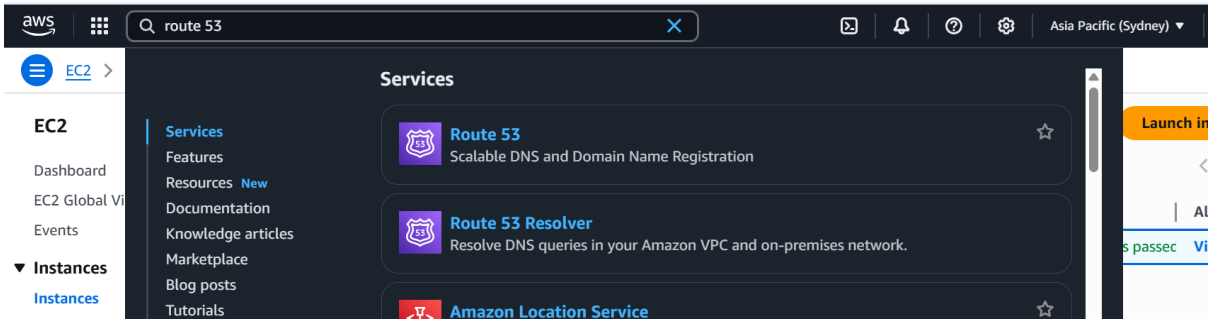


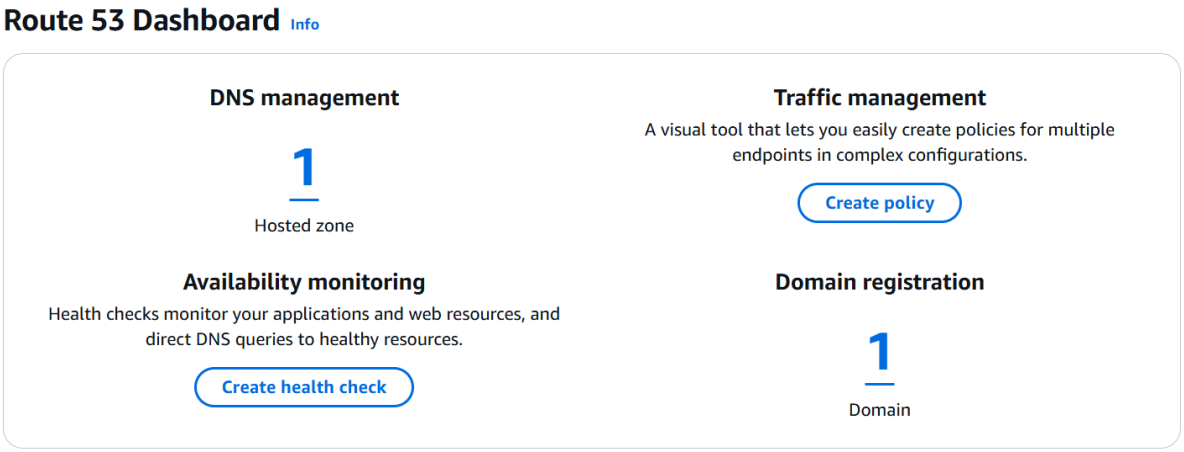

*If there is more than 1 instance, click on the “associate IP” to associate it with the wanted instance, and then click the desired instance to associate it to the correct one.*



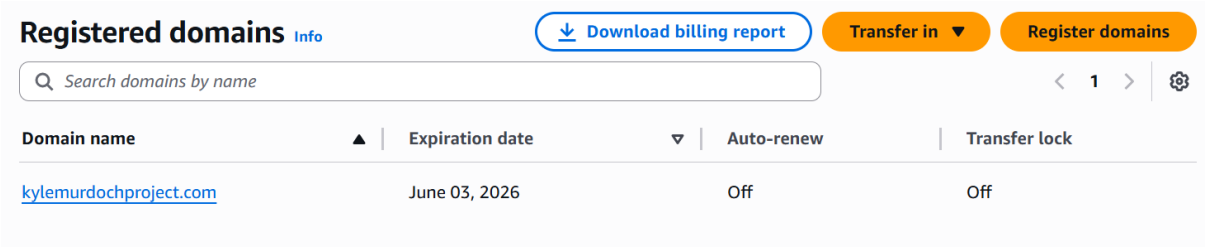
**Open Route 53:**

*Type in the top search bar "Route 53", then right-click it to open the link in another tab, this is where the DNS Name can be sorted.*

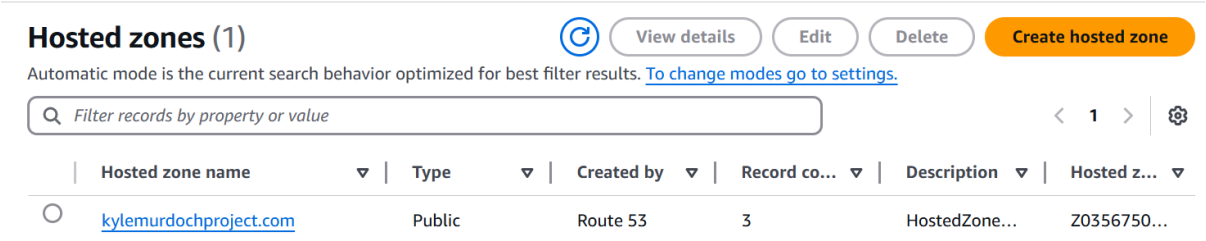


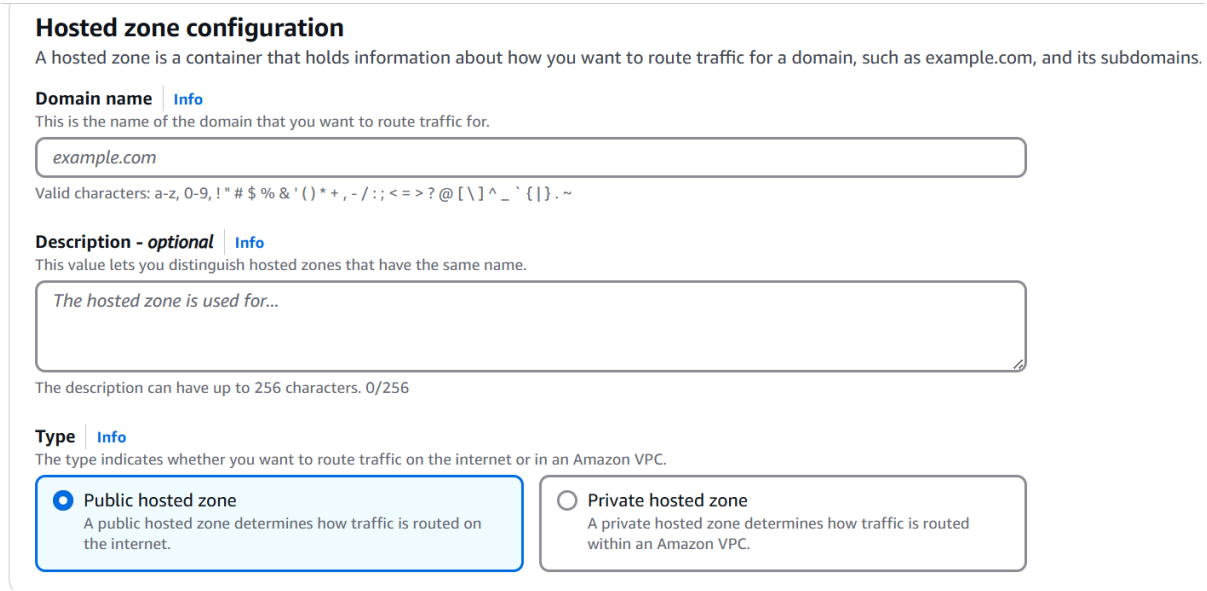

*Register and purchase a DNS name (kylemurdochproject.com)*



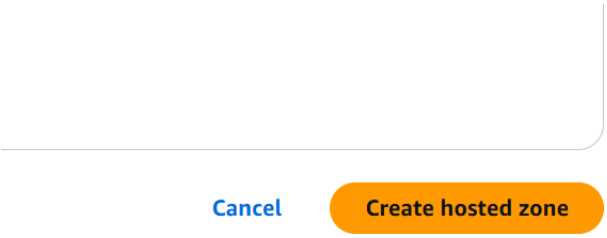

*Then go to DNS management – hosted zones*



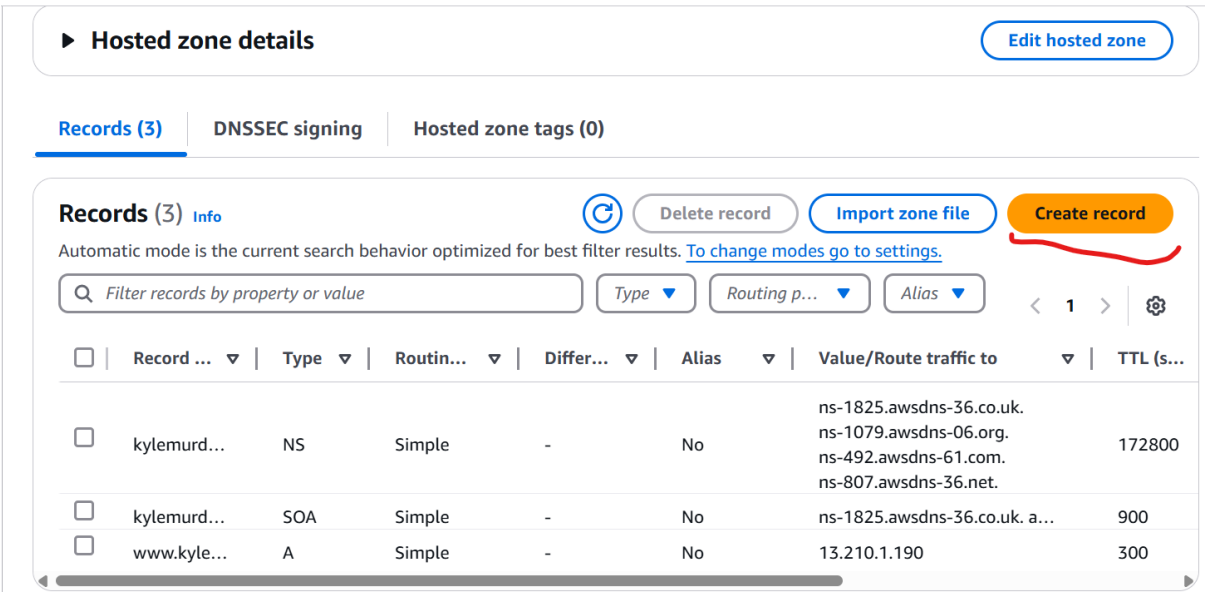
*Create a hosted zone, then configure by entering the purchased and registered DNS name in the domain name prompt, leave all the other setting as is and confirm it*



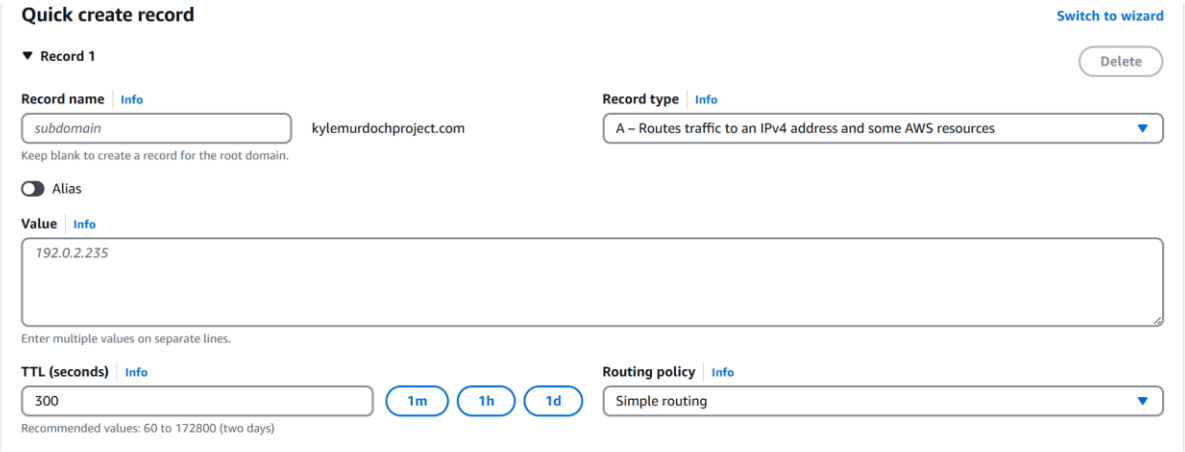

*Create it*




*Add a record, which tells the internet where to go when someone enters your domain name, linking it to the IP address. To do this, press on the name of our instance in the hosted zones, it will then show the prompt to add records*

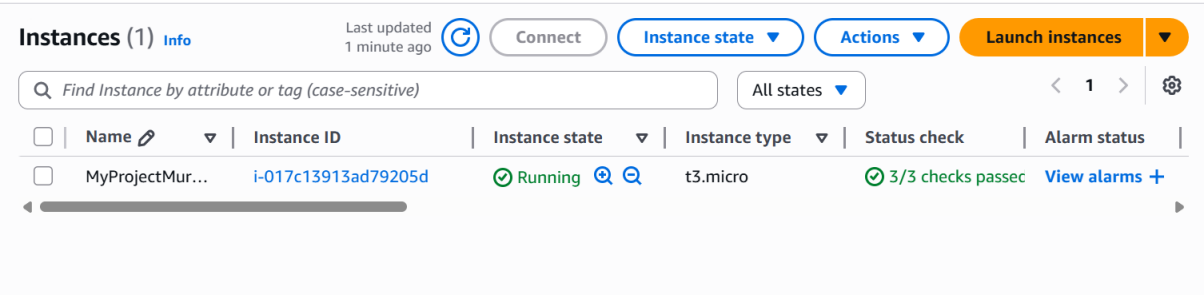


*Add the elastic IP into the value section of the record description*

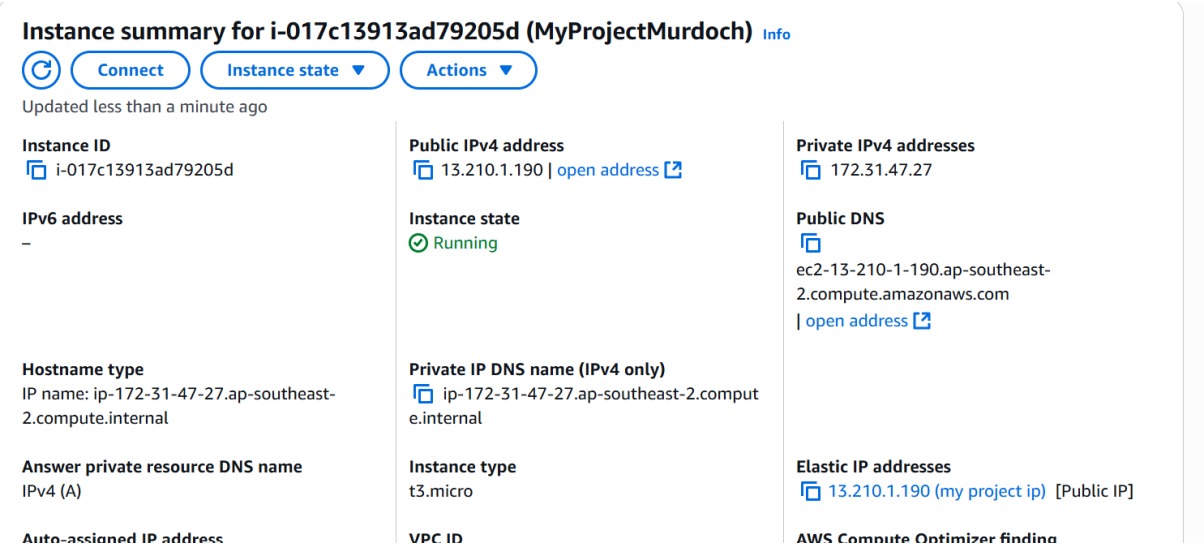



## Security, firewall settings and open ports:

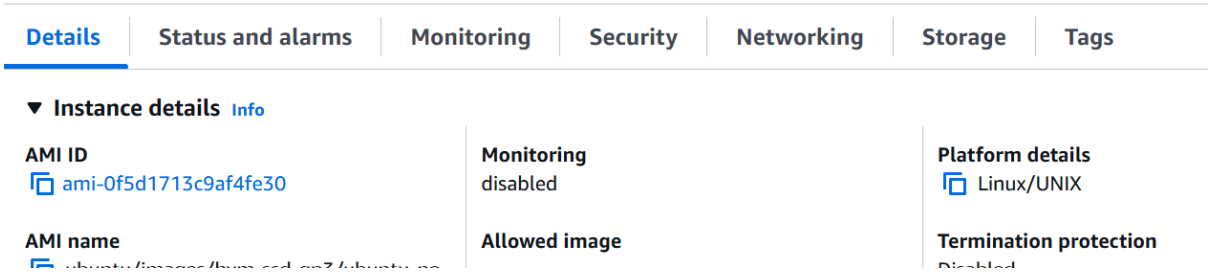
*Go back to ec2 instances dashboard*



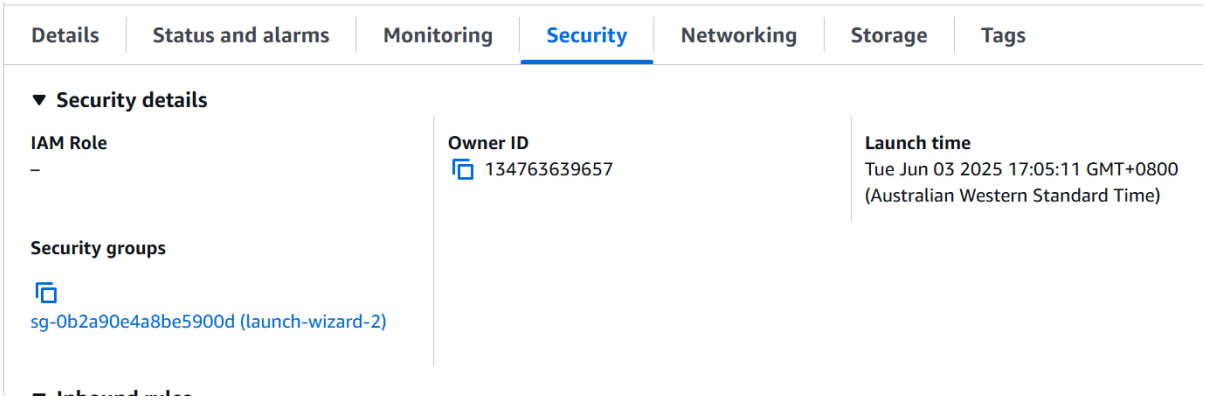

*Click on the instance ID, which will take you to the instance summary, - scroll down*



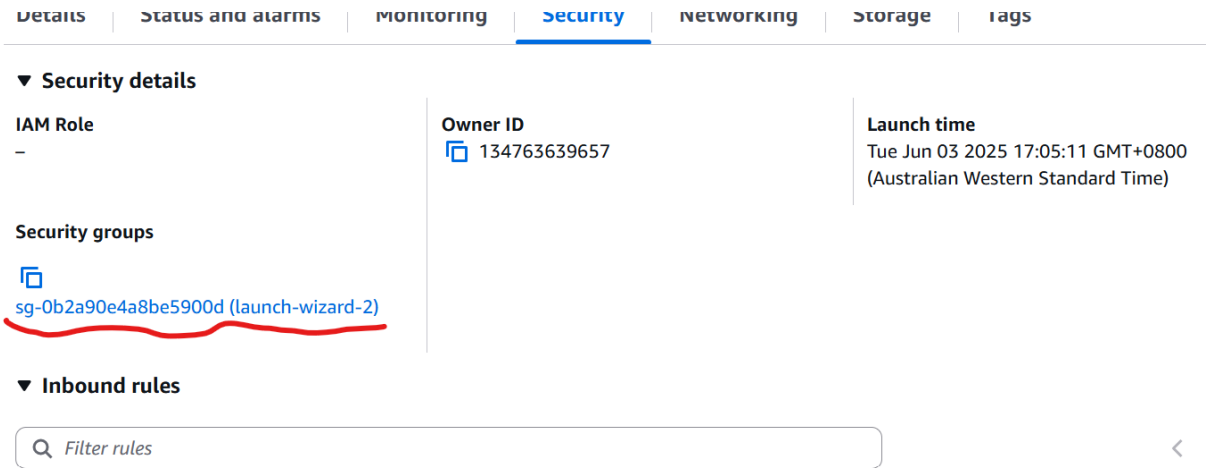
*Once at this point in the instance summary...*



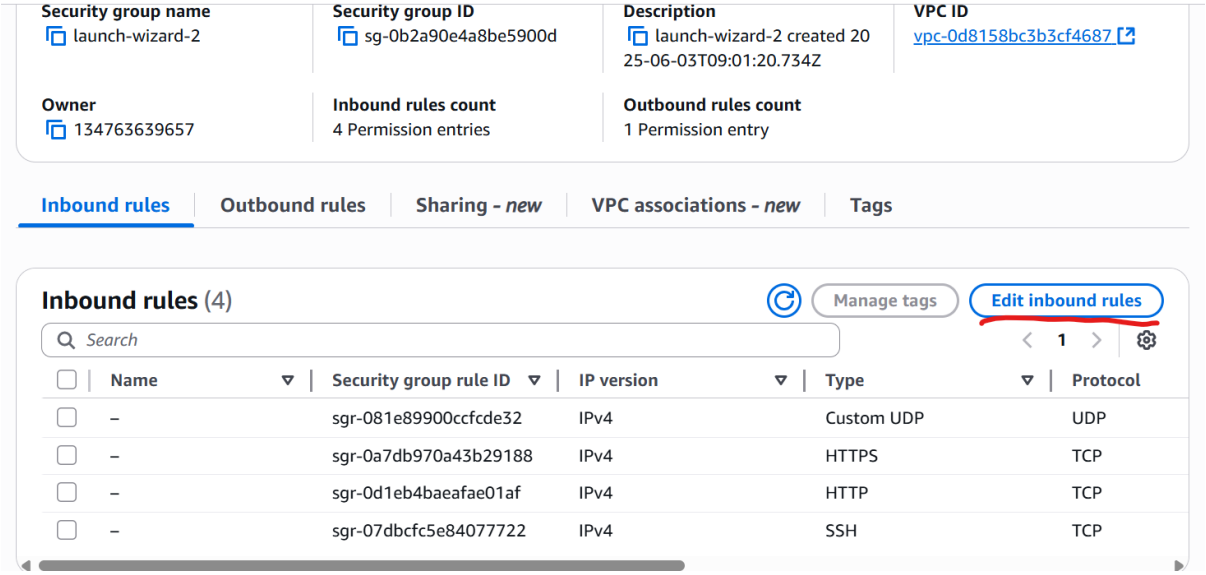

*Click on "security"*



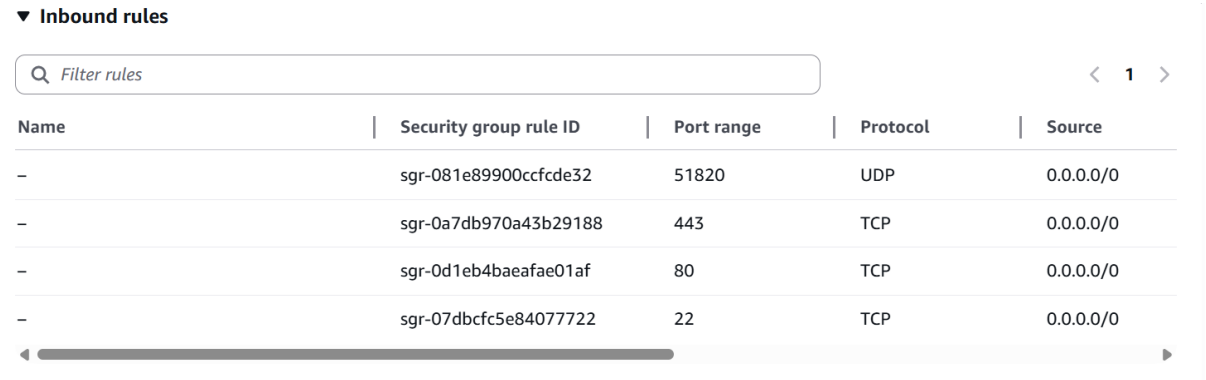

*Scroll down to "inbound rules" and then select the security group link*



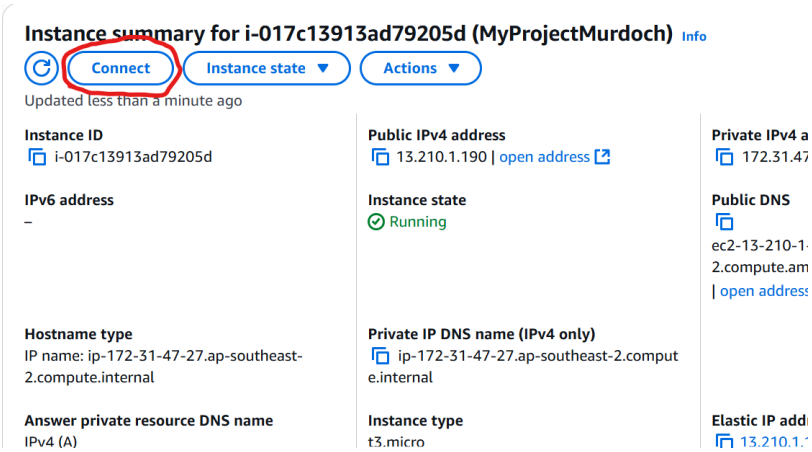
*This is where you need to create a new rule for port 51820 as UDP*




*As a result, the rule should be added to inbound rules like this*




## Connect to Instance:



*When in the terminal, it should look like this, then enter the following codes*

```
  System load:  0.08              Temperature:           -273.1 C
  Usage of /:   44.3% of 6.71GB   Processes:             116
  Memory usage: 35%               Users logged in:       0
  Swap usage:   0%                IPv4 address for ens5: 172.31.47.27

 * 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.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


*** System restart required ***
Last login: Sun Jun  8 01:12:35 2025 from 13.239.158.5
ubuntu@ip-172-31-47-27:~$
```

**i-017c13913ad79205d (MyProjectMurdoch)**

PublicIPs: 13.210.1.190 PrivateIPs: 172.31.47.27

*What to enter:*

***Sudo apt update***

***Sudo apt install wireguard***

***sudo mkdir /etc/wireguard/***

This is to make a directory for all the files

***wg genkey | sudo tee /etc/wireguard/privatekey | wg pubkey | sudo tee /etc/wireguard/publickey***

This is to generate the public and private keys for the VPN

***sudo nano /etc/wireguard/wg0.conf***

This is to create an edit the file

Enter this into the file:

**[Interface]**

**Address = 10.0.0.1/24**

**SaveConfig = true**

**ListenPort = 51820**

**PrivateKey = private_key**

**PostUp = iptables -A FORWARD -i %i -j ACCEPT; iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE**

**PostDown = iptables -D FORWARD -i %i -j ACCEPT; iptables -t nat -D POSTROUTING -o eth0 -j MASQUERADE**

Swap the private key with the actual private key created

***sudo systemctl enable wg-quick@wg0***

***sudo systemctl start wg-quick@wg0***

This starts the WireGuard service

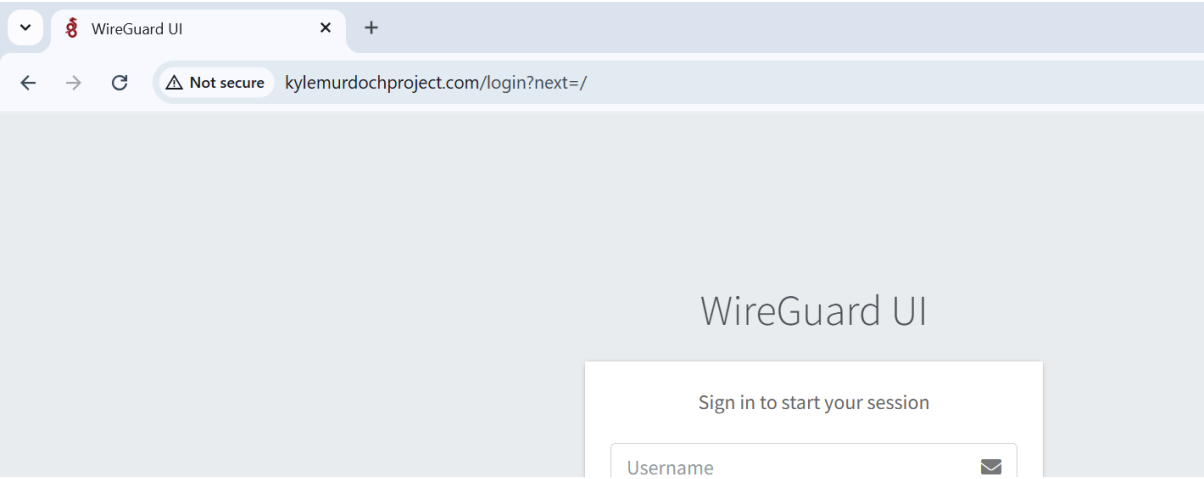
**sudo wg-quick up wg0**

This opens up the interface

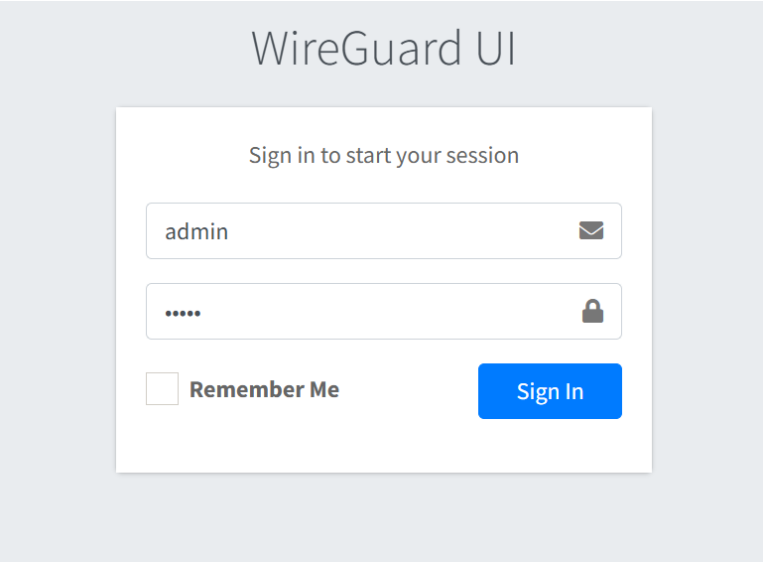
**this is all you need to input**

**Accessing the Wireguard VPN:**

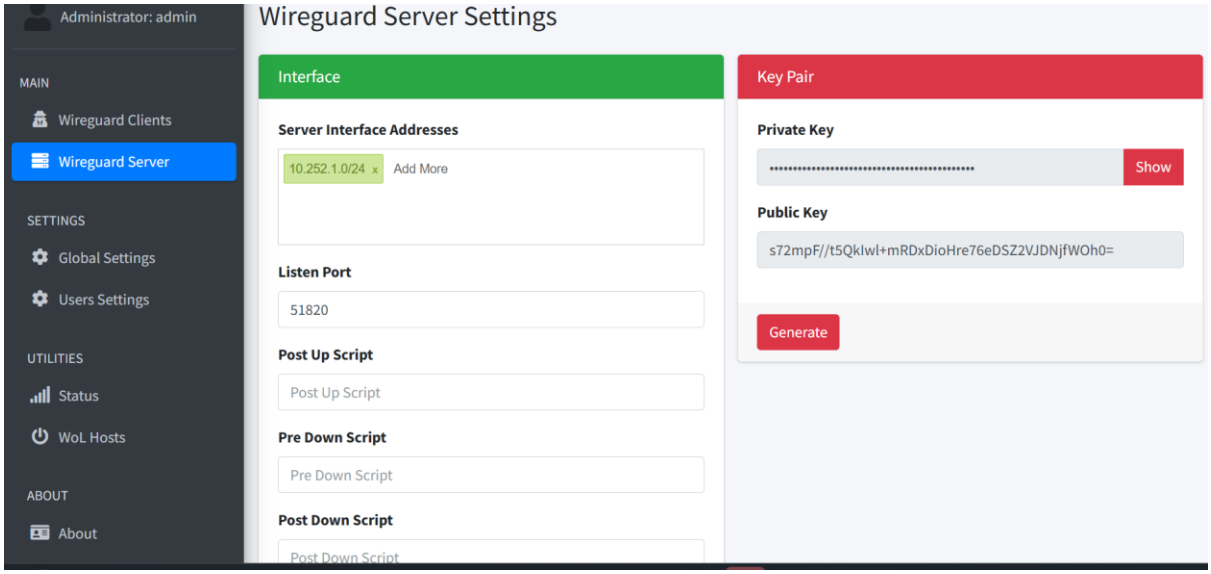
*In a new tab search up the instance's address, either the IP address or the DNS*

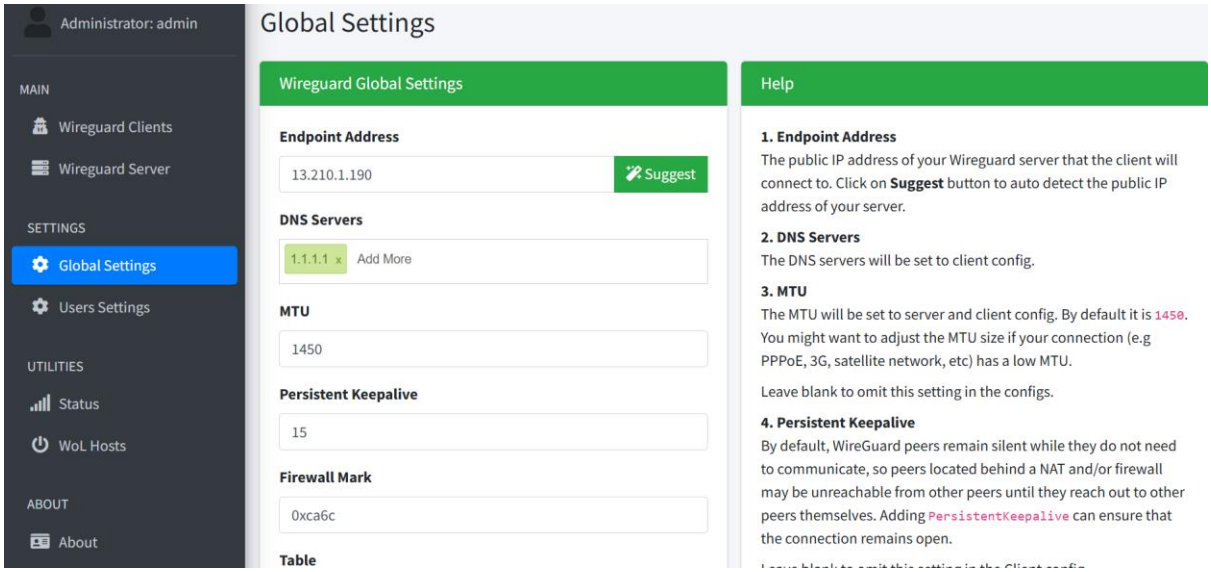


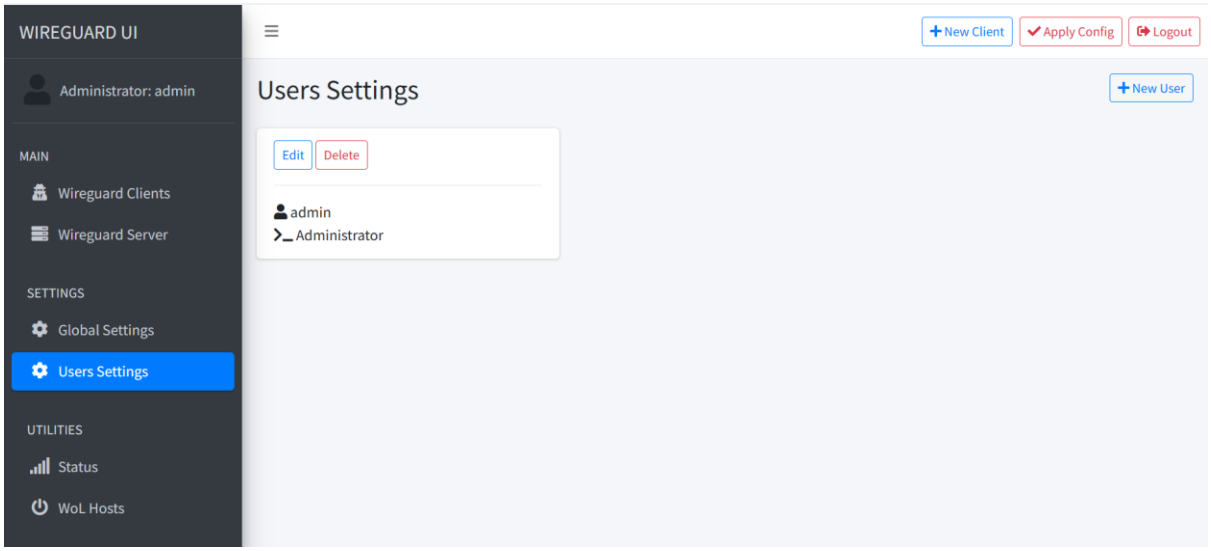
*Enter username -“admin”, password- “admin” to access the host privilege*



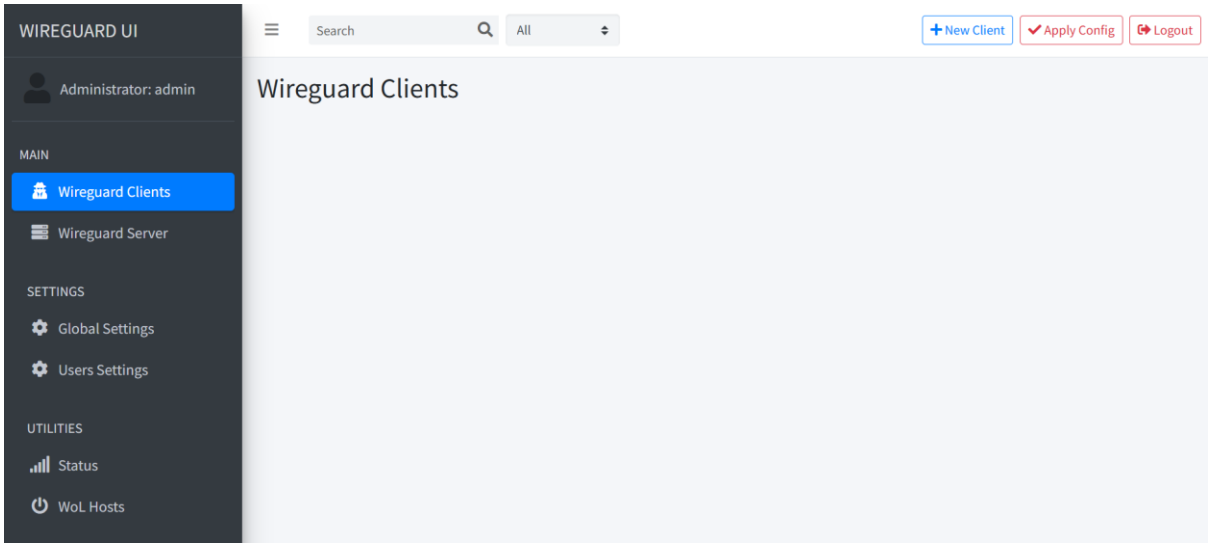

*(WireGuard interface)*

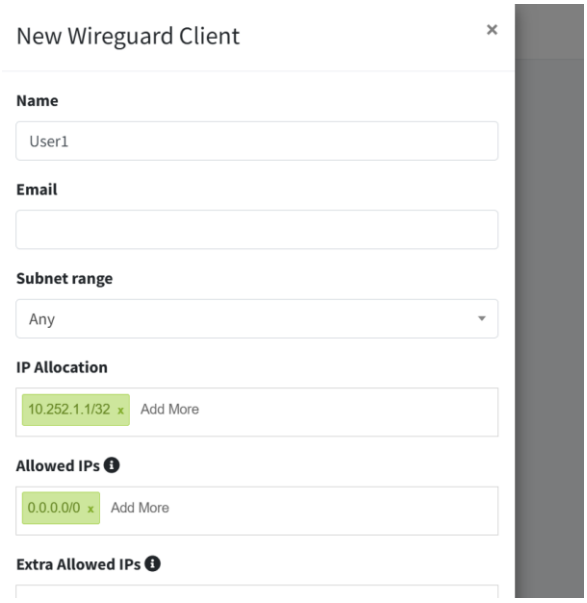


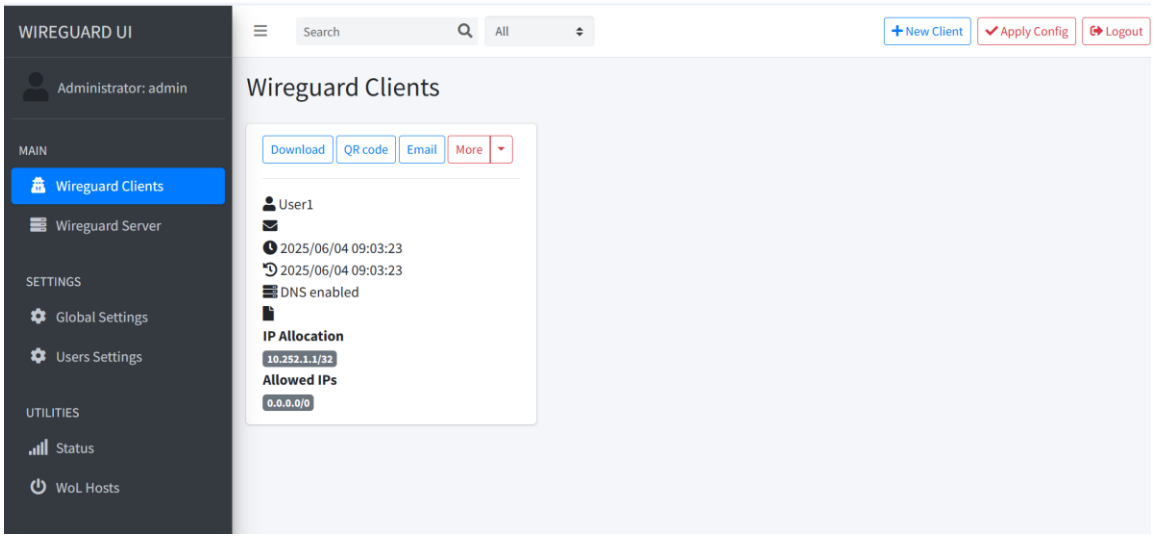


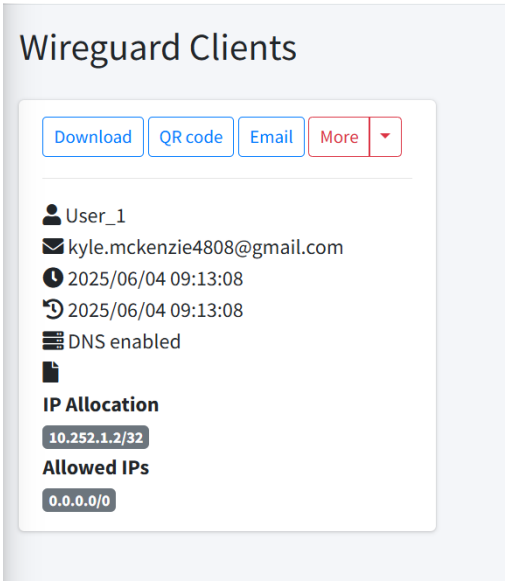




*Add users and confirm changes*



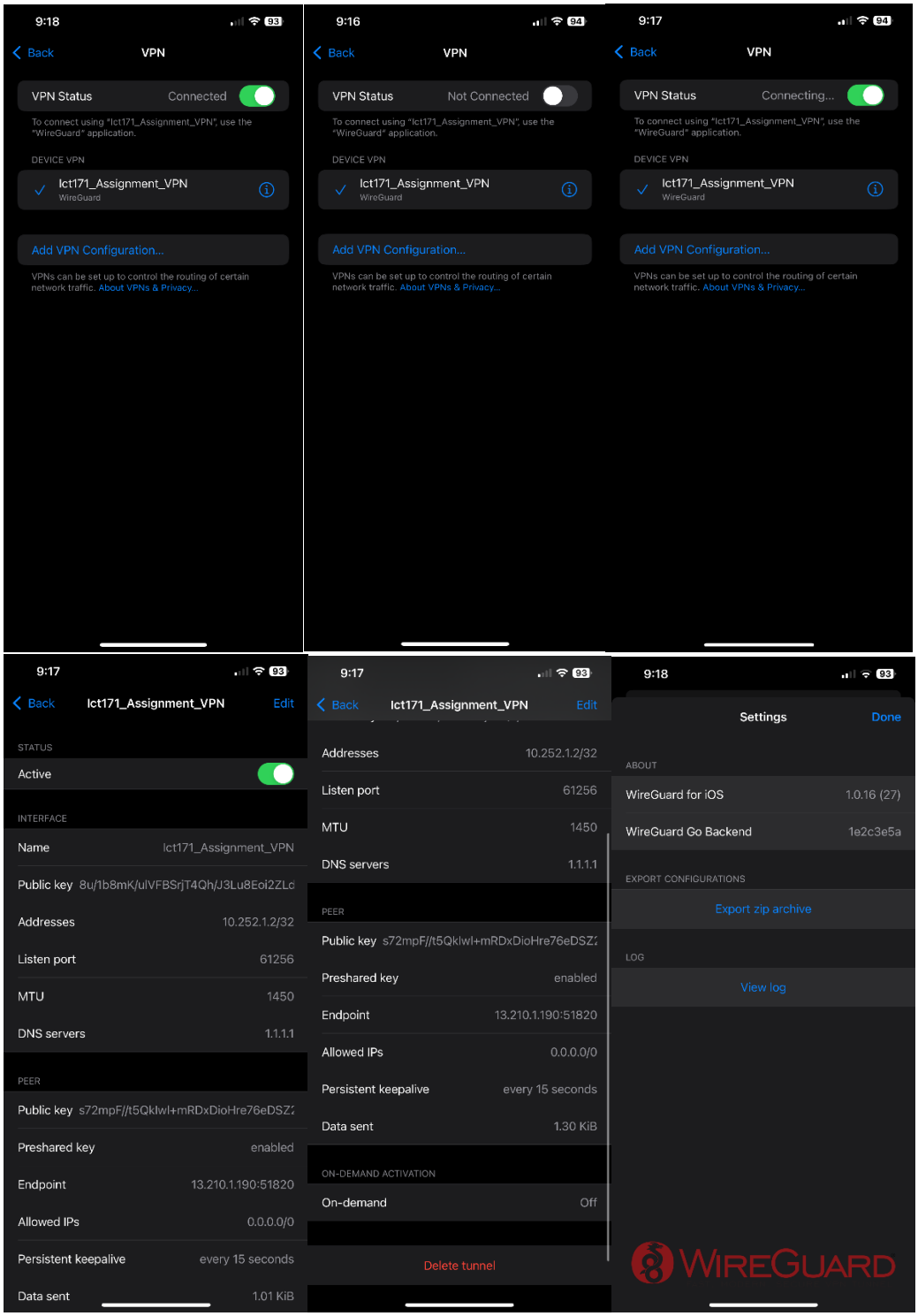

New Wireguard Client
Name
User1
Email
Subnet range
Any
IP Allocation
10.252.1.1/32
Add More
Allowed IPs
0.0.0.0/0
Add More
Extra Allowed IPs


WIREGUARD UI
Administrator: admin
MAIN
Wireguard Clients
Wireguard Server
SETTINGS
Global Settings
Users Settings
UTILITIES
Status
WoL Hosts
Search
All
New Client
Apply Config
Logout
Wireguard Clients
Download
QR code
Email
More
User1
2025/06/04 09:03:23
2025/06/04 09:03:23
DNS enabled
IP Allocation
10.252.1.1/32
Allowed IPs
0.0.0.0/0

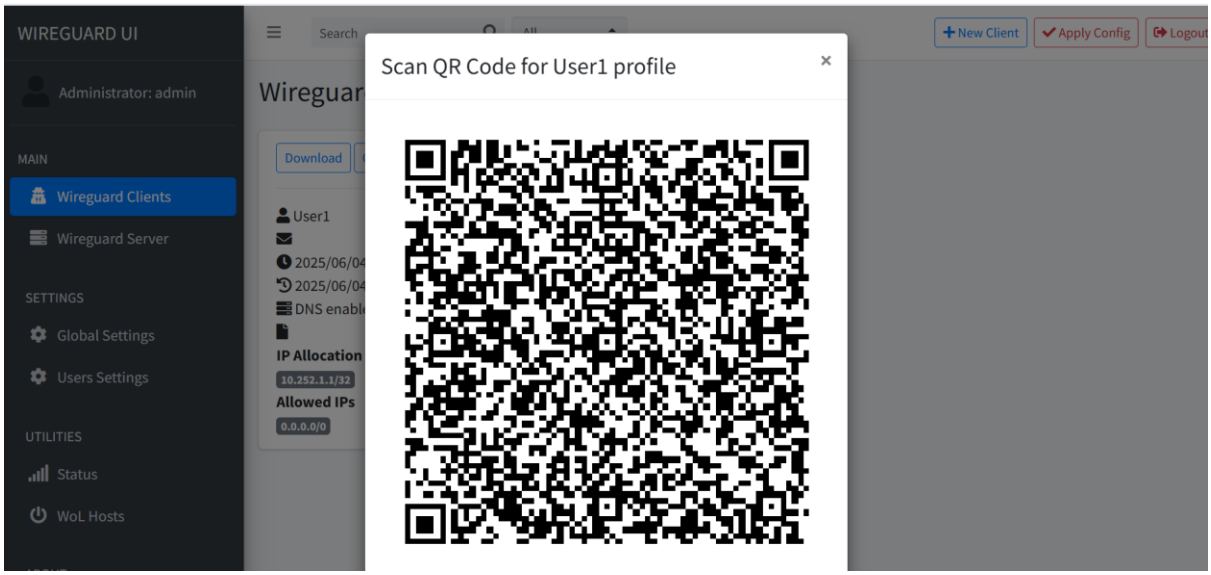

Wireguard Clients
Download
QR code
Email
More
User_1
kyle.mckenzie4808@gmail.com
2025/06/04 09:13:08
2025/06/04 09:13:08
DNS enabled
IP Allocation
10.252.1.2/32
Allowed IPs
0.0.0.0/0

*Get the user who is using it to download the application “WireGuard”*

*(Wireguard app/ iphone settings)*



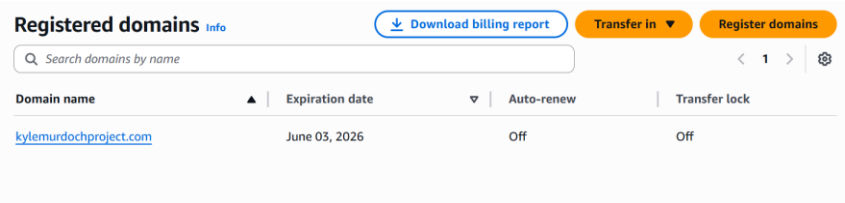
*Scan the QR code and the VPN will be set up*




## DNS linking process:

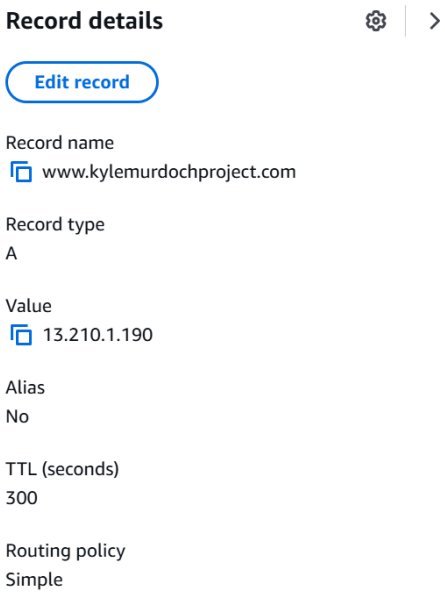
1. Purchase a domain name

   This domain was purchased from Amazon AWS Route 53. The domain name is kylemurdochproject.com, $14.50 USD/per year




2. Created an A record

   The type of the record – "A",
   the name of the record is www.kylemurdochproject.com,
   The record points to 13.210.1.190,



3. Waited roughly 10 minutes for DNS to propagate

### Configuration files and setting changes:

*Input of these configurations;*

/etc/wireguard/wg0.conf

Enter : **[Interface]**

**Address = 10.0.0.1/24**

**SaveConfig = true**

**ListenPort = 51820**

**PrivateKey = private_key**

**PostUp = iptables -A FORWARD -i %i -j ACCEPT; iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE**

**PostDown = iptables -D FORWARD -i %i -j ACCEPT; iptables -t nat -D POSTROUTING -o eth0 -j MASQUERADE**

/etc/sysctl.conf

net.ipv4.ip_forward=1

sudo sysctl -p

sudo systemctl enable wg-quick@wg0

sudo systemctl start wg-quick@wg0

## Script:

*Final script below:*

```
#!/bin/bash

# My WireGuard Status Report Script
# This script checks the status of all the  active users on my WireGuard service
# It also prints the system info like the  hostname, public IP and the uptime

# Check if wg command is available
if ! command -v wg &> /dev/null; then
```

```
  echo "Error: WireGuard is not installed or 'wg' command is not in PATH."
  exit 1
fi

# Get system information
HOSTNAME=$(hostname)
UPTIME=$(uptime -p)
PUBLIC_IP=$(curl -s ifconfig.me || echo "N/A")

# Print system info

echo "WireGuard VPN Status Report"

echo "Hostname     : $HOSTNAME"
echo "Public IP    : $PUBLIC_IP"
echo "System Uptime : $UPTIME"


# Get all active WireGuard interfaces
WG_INTERFACES=$(wg show interfaces)

if [ -z "$WG_INTERFACES" ]; then
  echo "No active WireGuard interfaces found."
  exit 0
fi

# Loop through each interface
for iface in $WG_INTERFACES; do
  echo "Interface    : $iface"


  # Interface info
  LISTEN_PORT=$(wg show "$iface" | grep "listen port" | awk '{print $3}')
```

```
  echo "Listen Port   : $LISTEN_PORT"

  # Show peers
  PEERS=$(wg show "$iface" peers)
  if [ -z "$PEERS" ]; then
    echo "No peers connected."
  else
    echo "Connected Peers:"

    for peer in $PEERS; do
      ENDPOINT=$(wg show "$iface" endpoint | grep "$peer" | awk '{print $2}')
      HANDSHAKE=$(wg show "$iface" latest-handshakes | grep "$peer" | awk '{print $2}')

      # Convert UNIX timestamp to readable format
      if [ "$HANDSHAKE" -eq 0 ]; then
        LAST_HANDSHAKE="Never"
      else
        LAST_HANDSHAKE=$(date -d @"$HANDSHAKE")
      fi

      RX=$(wg show "$iface" transfer | grep "$peer" | awk '{print $2}')
      TX=$(wg show "$iface" transfer | grep "$peer" | awk '{print $4}')

      echo "Peer: $peer"
      echo "  Endpoint       : $ENDPOINT"
      echo "  Last Handshake   : $LAST_HANDSHAKE"
      echo "  Transfer (RX/TX) : $RX / $TX"
      echo ""
    done
  fi

done
```

*Use of script*

This Bash script creates a live status report for my Wireguard VPN server. It shows the system hostname, public IP, and uptime, and checks all active WireGuard interfaces. For each interface, it lists peers, their connection endpoint, the timestamp of the last successful handshake, and data transferred in both directions. This gives a good summary of everything going on so that if anything isn't working it can be shown through a simple script.

*Output*

```
ubuntu@ip-172-31-47-27:~$ ./wg-status.sh
WireGuard VPN Status Report
Hostname      : ip-172-31-47-27
Public IP     : 13.210.1.190
System Uptime : up 5 days, 18 hours, 13 minutes
Interface     : wg0
Unable to access interface: Operation not permitted
Listen Port   :
Unable to access interface: Operation not permitted
No peers connected.
ubuntu@ip-172-31-47-27:~$
```

```
ubuntu@ip-172-31-47-27:~$ sudo ./wg-status.sh
WireGuard VPN Status Report
Hostname      : ip-172-31-47-27
Public IP     : 13.210.1.190
System Uptime : up 5 days, 18 hours, 16 minutes
Interface     : wg0
Listen Port   :
No peers connected.
ubuntu@ip-172-31-47-27:~$
```

**References:**


*Wireguard:*

Donenfeld, J. A. (n.d.). *WireGuard: Fast, modern, secure VPN tunnel*. WireGuard. https://www.wireguard.com/

*Wireguard set up:*

Tetzner, G. (2021, September 5). *Setting up a VPN with WireGuard server on AWS EC2*. DEV Community. https://dev.to/gabrieltetzner/setting-up-a-vpn-with-wireguard-server-on-aws-ec2-4a49

*Set up assistance and understanding:*

Numberphile. (2015, May 14). Why basic research is important [Video]. YouTube. https://www.youtube.com/watch?v=6gnsQjPCC