Networking Module - Mini project/assignment.

Project Objectives:

1. **Buy your own domain via Cloudflare or AWS Route53.**
2. **Create an EC2 instance running NGINX on port 80**. Add an A record to Cloudflare/ Route53 and point this to your EC2 instance. I want to be able to access NGINX webpage via your domain for example "nginx.luqman.co.uk"

What is route 53?

Route 53 is a scalable and highly available DNS web service provided by AWS. At a high level, it:

- Translates domain names - for example www.example.com into IP addresses
- Routes user traffic to AWS Services or external resources reliably and globally
- Has many uses such as supporting domain registration, DNS management and health checks.
- Has low latency and seamless integration with AWS Services

I have opted for AWS Route 53 over other domain registration providers such as Cloudflare due to its seamless integration with other AWS Services (e.g. EC2 instances).

Navigating AWS console:

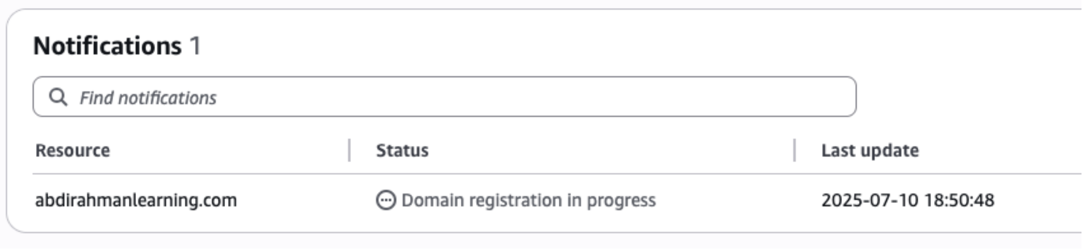
First enter Route 53 in the search bar (using your IAM user) and register a domain.

Enter the name of the domain you wish to create and buy. In this area I selected the name "Abdirahmanlearning.com".

Once you have entered your personal details, you request to create this domain.

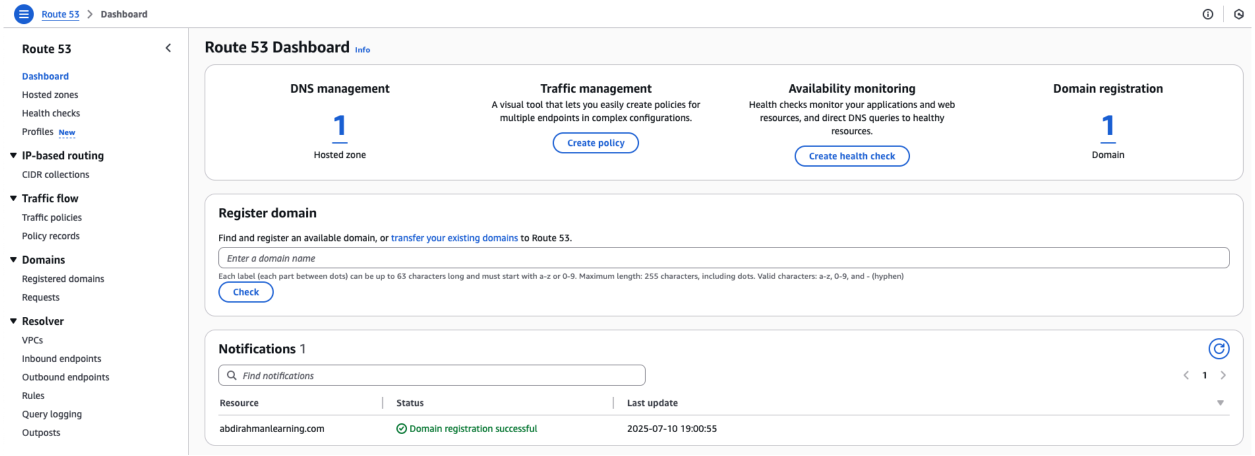
Domain registration request:

AWS will email you to verify your request and they begin to process your domain registration request. (see below image)



Resource	Status	Last update
abdirahmanlearning.com	Domain registration in progress	2025-07-10 18:50:48

It took me about 10-15 minutes for the domain to be registered (AWS will email you once domain registration is successful) - see below for this notification in the AWS Route 53 Dashboard.



Route 53 Dashboard

DNS management: 1 Hosted zone

Traffic management: A visual tool that lets you easily create policies for multiple endpoints in complex configurations.

Availability monitoring: Health checks monitor your applications and web resources, and direct DNS queries to healthy resources.

Domain registration: 1 Domain

Register domain

Find and register an available domain, or transfer your existing domains to Route 53.

Enter a domain name

Each label (each part between dots) can be up to 63 characters long and must start with a-z or 0-9. Maximum length: 255 characters, including dots. Valid characters: a-z, 0-9, and - (hyphen)

Notifications 1

Find notifications

Resource	Status	Last update
abdirahmanlearning.com	✓ Domain registration successful	2025-07-10 19:00:55

Create an EC2 instance running NGINX on port 80:

Launch an EC2 instance using Amazon Linux 2023 AMI and t2.micro as the instance type (free tier eligible). Select a key pair (create one if required, but I have one set up already).

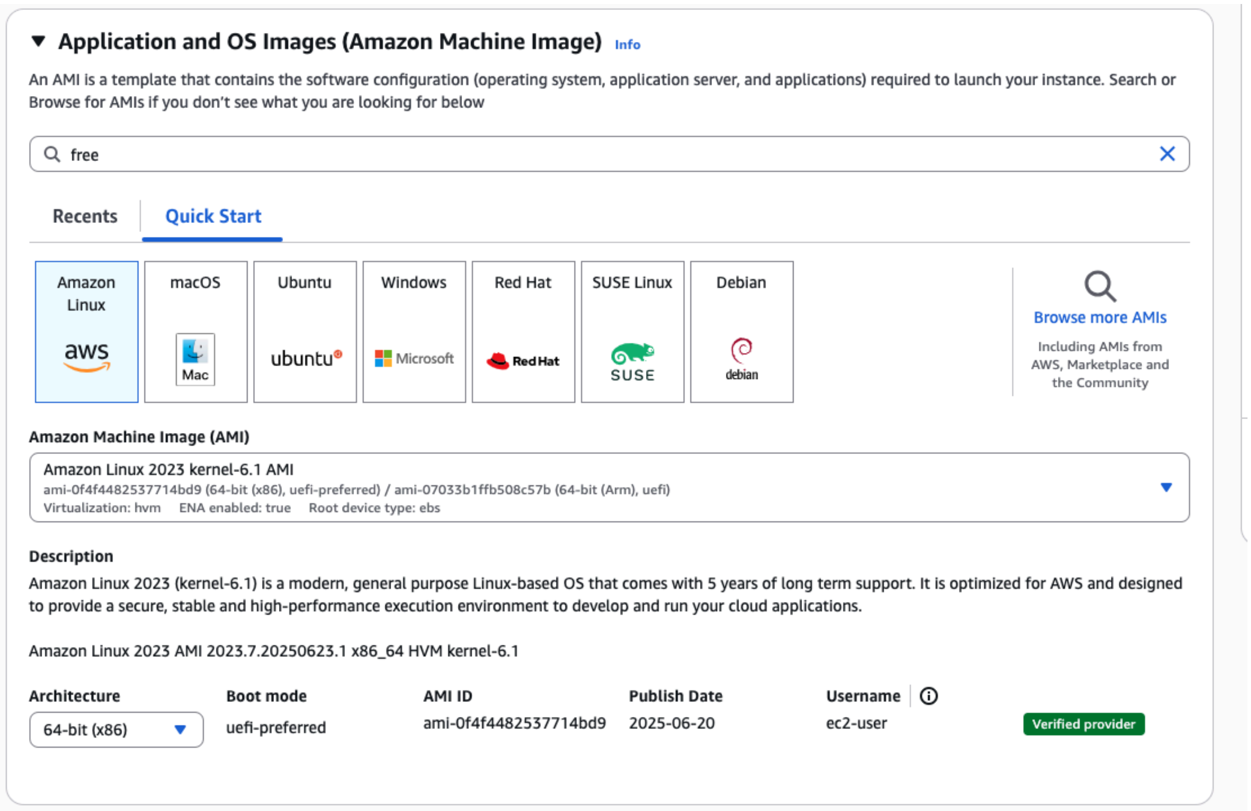
Next, set up the Network settings. I selected the default VPC and a default subnet in my region EU-West-2. Ensure auto-assign public IP is enabled - this gives the EC2 instance a public IPv4 address so it can be reached from the internet.

Without a public IP, Route 53 cannot route traffic from your domain to your EC2 instance over the public internet.

Onto security settings. Create a security group and configure inbound security group rules, they must allow:

- SSH (TCP port 22): Allows you to connect via SSH (e.g., with your key pair).
- HTTP (TCP port 80): Allows public web traffic to access your NGINX server.

The EC2 Instance is now ready to be launched. (Screenshots are shown below on the configuration of the EC2 instance)



▼ Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

free

Amazon Machine Image (AMI)

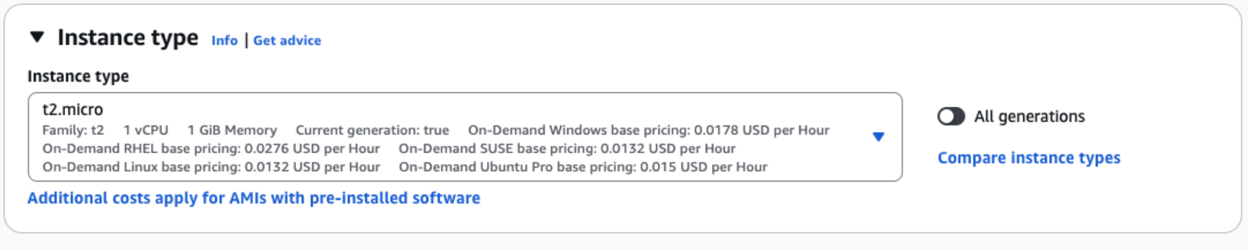
Amazon Linux 2023 kernel-6.1 AMI
ami-0f4f4482537714bd9 (64-bit (x86), uefi-preferred) / ami-07033b1ffb508c57b (64-bit (Arm), uefi)
Virtualization: hvm   ENA enabled: true   Root device type: ebs

Description

Amazon Linux 2023 (kernel-6.1) is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications.

Amazon Linux 2023 AMI 2023.7.20250623.1 x86_64 HVM kernel-6.1

Architecture	Boot mode	AMI ID	Publish Date	Username	
64-bit (x86)	uefi-preferred	ami-0f4f4482537714bd9	2025-06-20	ec2-user	Verified provider

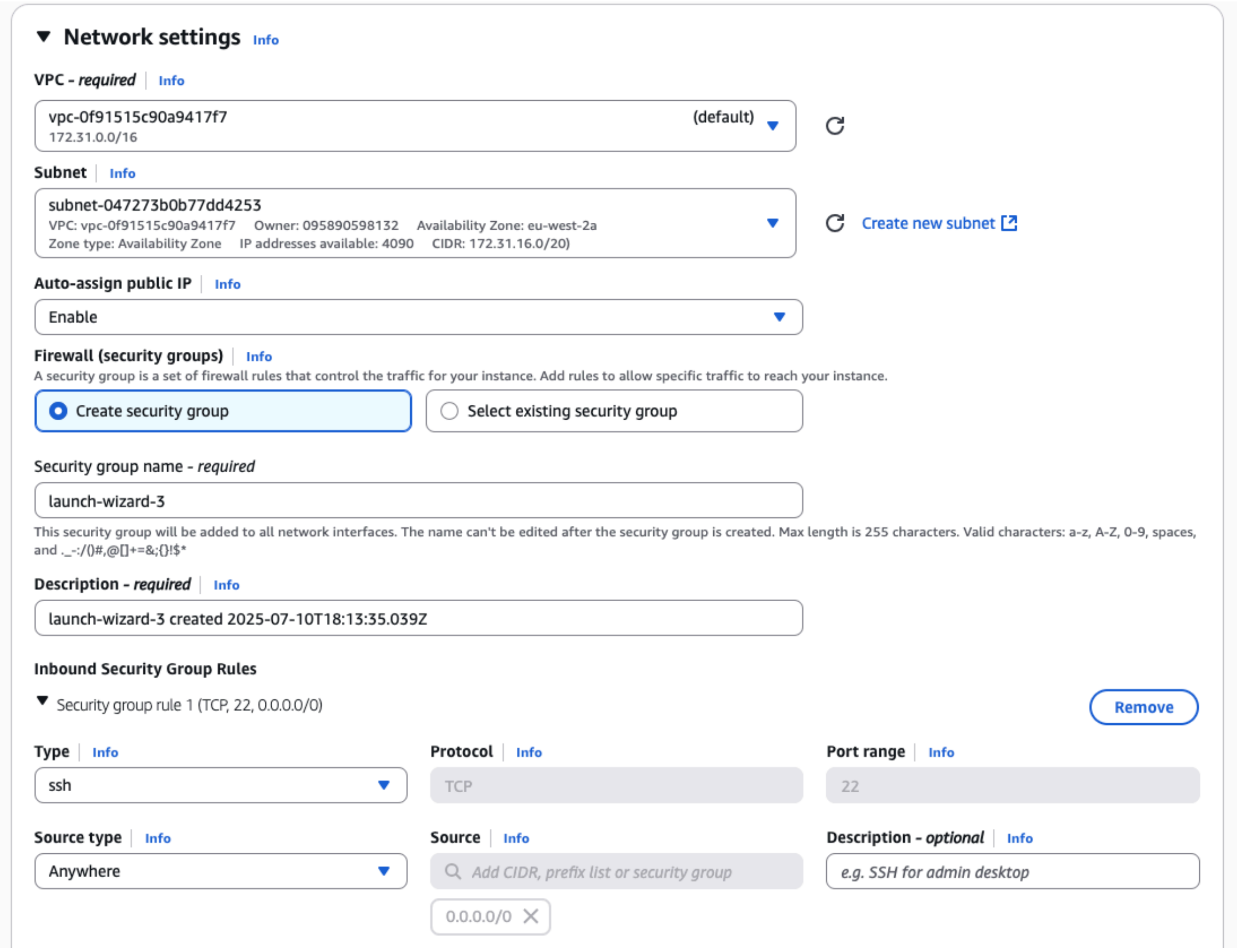


▼ Instance type Info | Get advice

Instance type

t2.micro
Family: t2   1 vCPU   1 GiB Memory   Current generation: true   On-Demand Windows base pricing: 0.0178 USD per Hour
On-Demand RHEL base pricing: 0.0276 USD per Hour   On-Demand SUSE base pricing: 0.0132 USD per Hour
On-Demand Linux base pricing: 0.0132 USD per Hour   On-Demand Ubuntu Pro base pricing: 0.015 USD per Hour

Additional costs apply for AMIs with pre-installed software



▼ Network settings Info

VPC - *required* | Info

vpc-0f91515c90a9417f7 (default)
172.31.0.0/16

Subnet | Info

subnet-047273b0b77dd4253
VPC: vpc-0f91515c90a9417f7 Owner: 095890598132 Availability Zone: eu-west-2a
Zone type: Availability Zone IP addresses available: 4090 CIDR: 172.31.16.0/20)

Auto-assign public IP | Info

Enable

Firewall (security groups) | Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

● Create security group ○ Select existing security group

Security group name - *required*

launch-wizard-3

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and ._-:/()#,@[]+=&;{}!$*

Description - *required* | Info

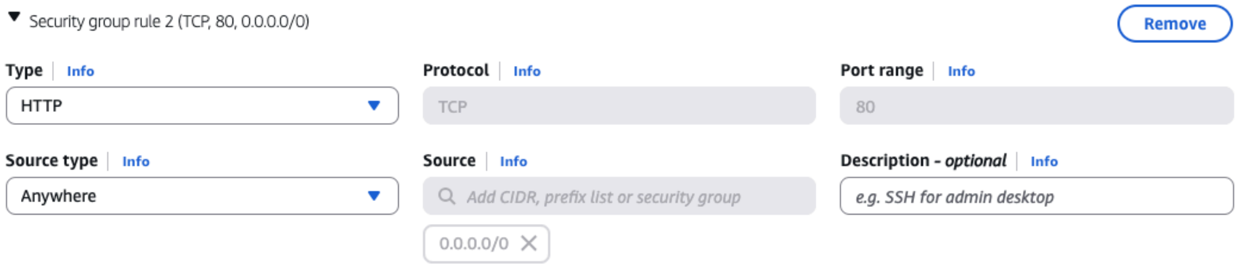
launch-wizard-3 created 2025-07-10T18:13:35.039Z

Inbound Security Group Rules

▼ Security group rule 1 (TCP, 22, 0.0.0.0/0)

Type	Protocol	Port range
ssh	TCP	22

Source type	Source	Description - *optional*
Anywhere	Add CIDR, prefix list or security group 0.0.0.0/0	e.g. SSH for admin desktop



▼ Security group rule 2 (TCP, 80, 0.0.0.0/0)

Type	Protocol	Port range
HTTP	TCP	80

Source type	Source	Description - *optional*
Anywhere	Add CIDR, prefix list or security group 0.0.0.0/0	e.g. SSH for admin desktop

The EC2 instance is now “running”.

Connect to EC2 instance via SSH:

Click onto the running EC2 instance and select "connect to instance”. The method we will connect to our instance is via the terminal using an SSH client. This is where our private key pair, that we selected during the EC2 set up, will come in.

Open your Terminal, locate where you saved your private key and enter the directory it is located in using “cd”. Then run the command:

chmod 400 “<your private key pair name>.pem”

This is assigning permissions so that our private key pair is not publicly viewable.

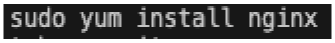
Next connect to the EC2 instance via SSH by running the command:

ssh -i "<your private key pair name.pem>"
ec2-user@ec2-18-133-181-14.eu-west-2.compute.amazonaws.com

We have now connected to the EC2 instance via SSH:

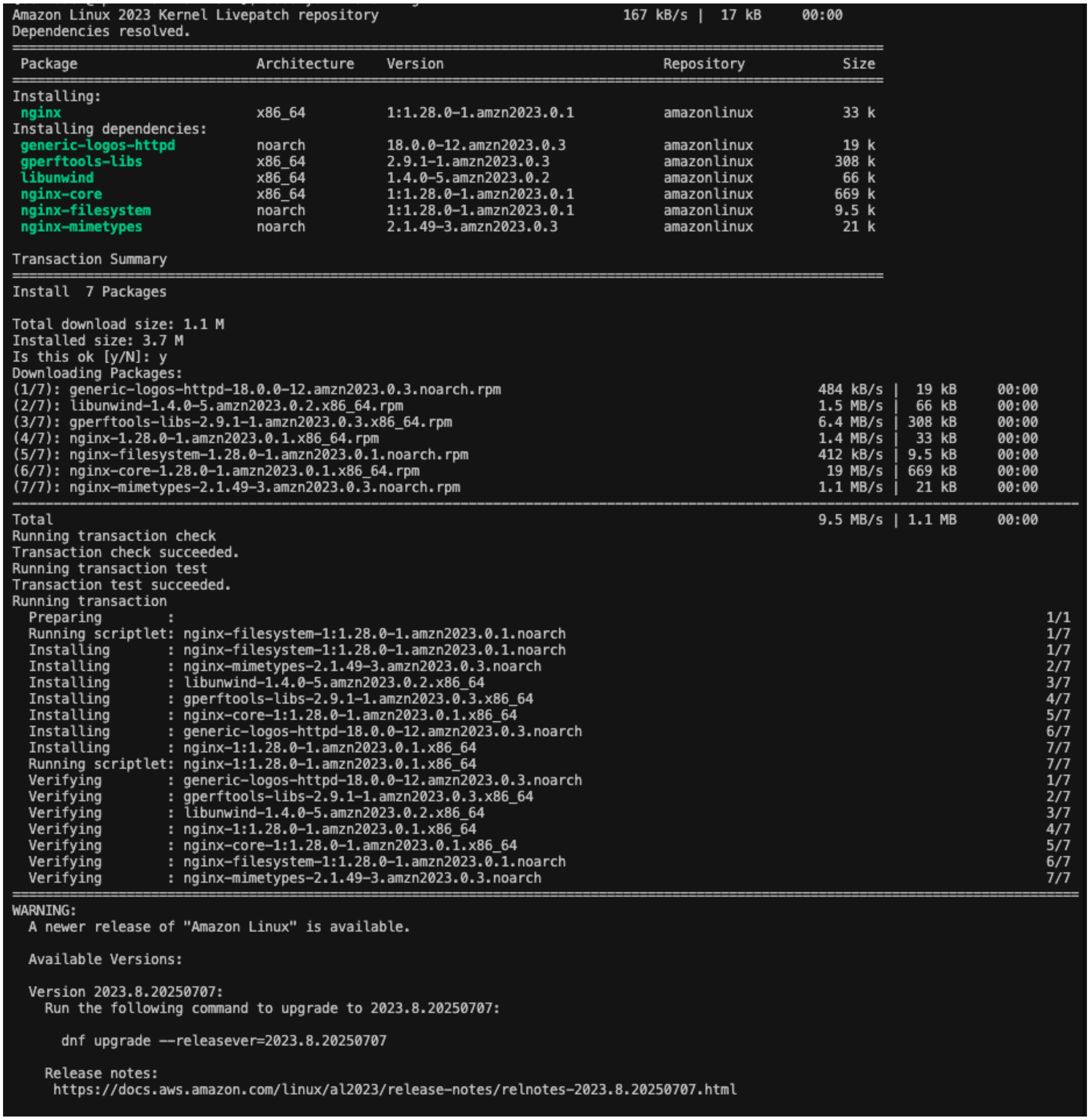
Install NGINX on your instance:

Install NGINX in the CLI using the command:

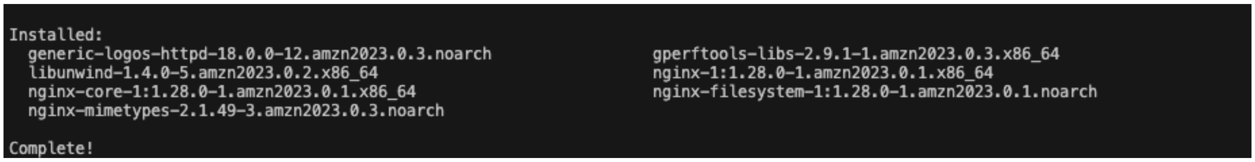


```
sudo yum install nginx
```

Below is the installation output:

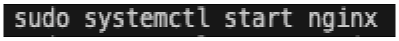


```
Amazon Linux 2023 Kernel Livepatch repository                          167 kB/s |  17 kB     00:00
Dependencies resolved.
===========================================================================================================
 Package                      Architecture    Version                           Repository            Size
===========================================================================================================
Installing:
 nginx                        x86_64          1:1.28.0-1.amzn2023.0.1           amazonlinux           33 k
Installing dependencies:
 generic-logos-httpd          noarch          18.0.0-12.amzn2023.0.3            amazonlinux           19 k
 gperftools-libs              x86_64          2.9.1-1.amzn2023.0.3              amazonlinux          308 k
 libunwind                    x86_64          1.4.0-5.amzn2023.0.2              amazonlinux           66 k
 nginx-core                   x86_64          1:1.28.0-1.amzn2023.0.1           amazonlinux          669 k
 nginx-filesystem             noarch          1:1.28.0-1.amzn2023.0.1           amazonlinux          9.5 k
 nginx-mimetypes              noarch          2.1.49-3.amzn2023.0.3             amazonlinux           21 k

Transaction Summary
===========================================================================================================
Install  7 Packages

Total download size: 1.1 M
Installed size: 3.7 M
Is this ok [y/N]: y
Downloading Packages:
(1/7): generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch.rpm                            484 kB/s |  19 kB     00:00
(2/7): libunwind-1.4.0-5.amzn2023.0.2.x86_64.rpm                                        1.5 MB/s |  66 kB     00:00
(3/7): gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64.rpm                                  6.4 MB/s | 308 kB     00:00
(4/7): nginx-1.28.0-1.amzn2023.0.1.x86_64.rpm                                           1.4 MB/s |  33 kB     00:00
(5/7): nginx-filesystem-1.28.0-1.amzn2023.0.1.noarch.rpm                                412 kB/s | 9.5 kB     00:00
(6/7): nginx-core-1.28.0-1.amzn2023.0.1.x86_64.rpm                                       19 MB/s | 669 kB     00:00
(7/7): nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch.rpm                                 1.1 MB/s |  21 kB     00:00
-----------------------------------------------------------------------------------------------------------------------------------
Total                                                                                   9.5 MB/s | 1.1 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                                            1/1
  Running scriptlet: nginx-filesystem-1:1.28.0-1.amzn2023.0.1.noarch                                                            1/7
  Installing       : nginx-filesystem-1:1.28.0-1.amzn2023.0.1.noarch                                                            1/7
  Installing       : nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch                                                               2/7
  Installing       : libunwind-1.4.0-5.amzn2023.0.2.x86_64                                                                      3/7
  Installing       : gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64                                                                4/7
  Installing       : nginx-core-1:1.28.0-1.amzn2023.0.1.x86_64                                                                  5/7
  Installing       : generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch                                                          6/7
  Installing       : nginx-1:1.28.0-1.amzn2023.0.1.x86_64                                                                       7/7
  Running scriptlet: nginx-1:1.28.0-1.amzn2023.0.1.x86_64                                                                       7/7
  Verifying        : generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch                                                          1/7
  Verifying        : gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64                                                                2/7
  Verifying        : libunwind-1.4.0-5.amzn2023.0.2.x86_64                                                                      3/7
  Verifying        : nginx-1:1.28.0-1.amzn2023.0.1.x86_64                                                                       4/7
  Verifying        : nginx-core-1:1.28.0-1.amzn2023.0.1.x86_64                                                                  5/7
  Verifying        : nginx-filesystem-1:1.28.0-1.amzn2023.0.1.noarch                                                            6/7
  Verifying        : nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch                                                               7/7
===================================================================================================================================
WARNING:
  A newer release of "Amazon Linux" is available.

  Available Versions:

  Version 2023.8.20250707:
    Run the following command to upgrade to 2023.8.20250707:

      dnf upgrade --releasever=2023.8.20250707

    Release notes:
     https://docs.aws.amazon.com/linux/al2023/release-notes/relnotes-2023.8.20250707.html
```



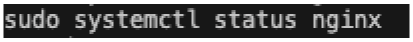
```
Installed:
  generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch                    gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64
  libunwind-1.4.0-5.amzn2023.0.2.x86_64                                nginx-1:1.28.0-1.amzn2023.0.1.x86_64
  nginx-core-1:1.28.0-1.amzn2023.0.1.x86_64                            nginx-filesystem-1:1.28.0-1.amzn2023.0.1.noarch
  nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch

Complete!
```

NGINX is now installed. Now lets start it up using the command:



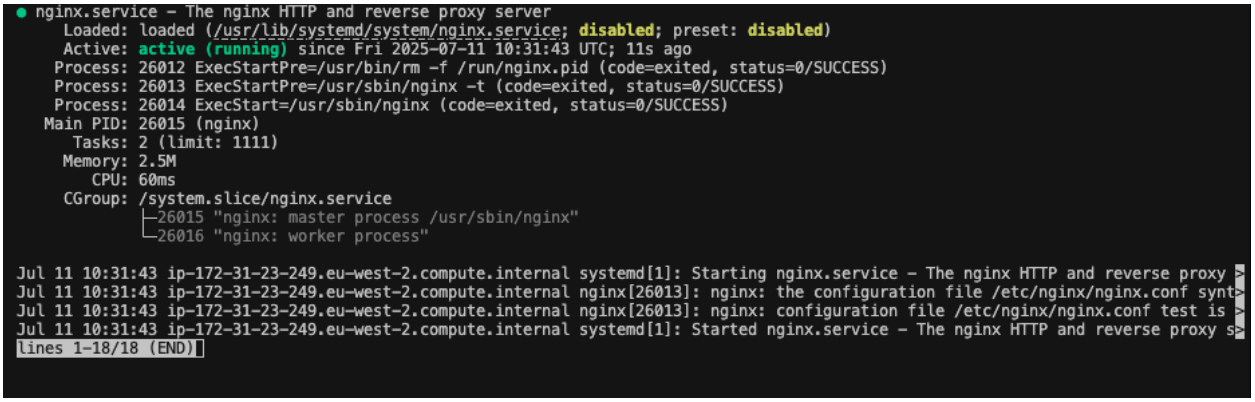
```
sudo systemctl start nginx
```

To confirm that it is running use the command:



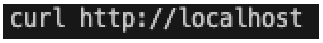
```
sudo systemctl status nginx
```

Below is the status output:

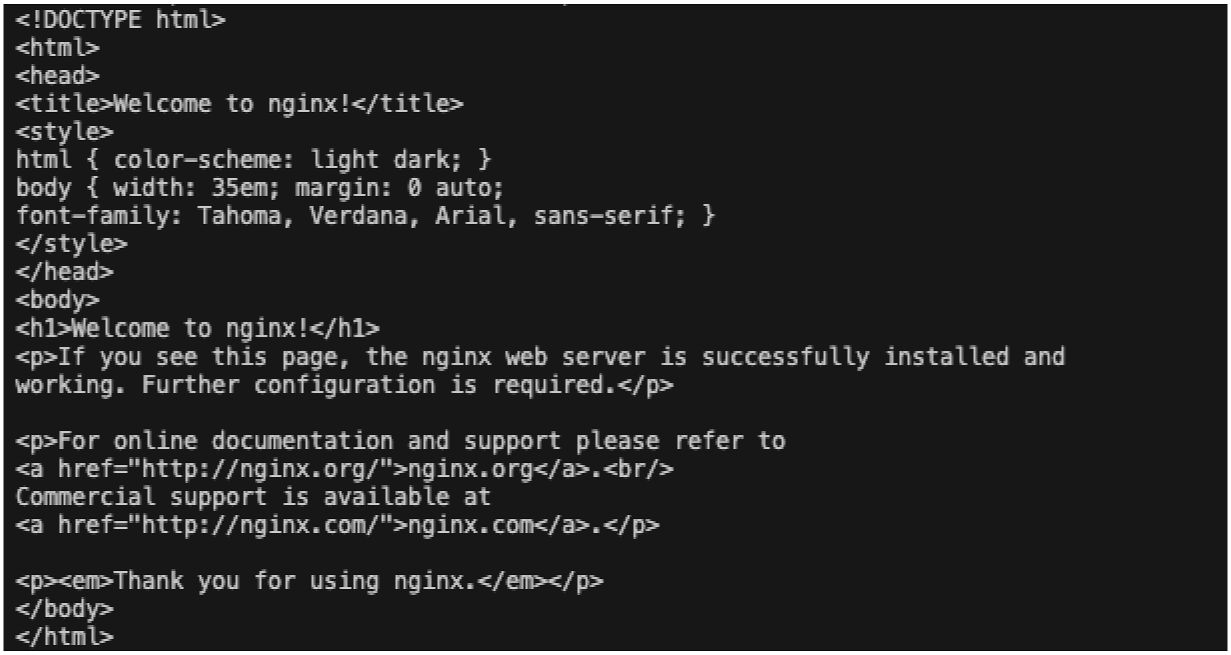


```
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; preset: disabled)
     Active: active (running) since Fri 2025-07-11 10:31:43 UTC; 11s ago
    Process: 26012 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
    Process: 26013 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
    Process: 26014 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
   Main PID: 26015 (nginx)
      Tasks: 2 (limit: 1111)
     Memory: 2.5M
        CPU: 60ms
     CGroup: /system.slice/nginx.service
             ├─26015 "nginx: master process /usr/sbin/nginx"
             └─26016 "nginx: worker process"

Jul 11 10:31:43 ip-172-31-23-249.eu-west-2.compute.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy >
Jul 11 10:31:43 ip-172-31-23-249.eu-west-2.compute.internal nginx[26013]: nginx: the configuration file /etc/nginx/nginx.conf synt>
Jul 11 10:31:43 ip-172-31-23-249.eu-west-2.compute.internal nginx[26013]: nginx: configuration file /etc/nginx/nginx.conf test is >
Jul 11 10:31:43 ip-172-31-23-249.eu-west-2.compute.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy s>
lines 1-18/18 (END)
```

We can also test NGINX locally, using the curl command:



```
curl http://localhost
```



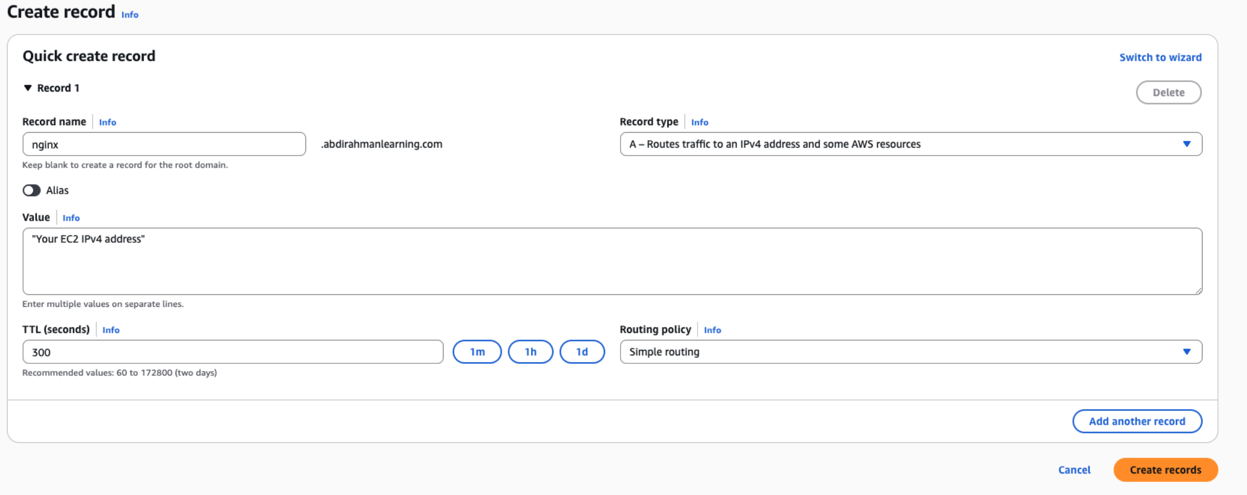
```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

We can confirm that the NGINX HTML Page is working on the instance itself.

Create A record using our domain in Route 53:

Return to Route 53 on the AWS management console and head to "Hosted Zones". Now, select the domain you have registered, in my case it is "abdirahmanlearning.com". Then select "create record".

Fill in the details for the record. Naming the A record as “nginx.abdirahmanlearning.com” In this case we are setting up an “A” record. In the “Value” section, we input the EC2 Public IPv4 address. Keep the TTL as 300s and create the record. We are also keeping the routing policy as “Simple routing” The screenshot below illustrates this:



Create record

Quick create record

▼ Record 1

Record name: nginx .abdirahmanlearning.com

Keep blank to create a record for the root domain.

Alias

Record type: A – Routes traffic to an IPv4 address and some AWS resources

Value: "Your EC2 IPv4 address"

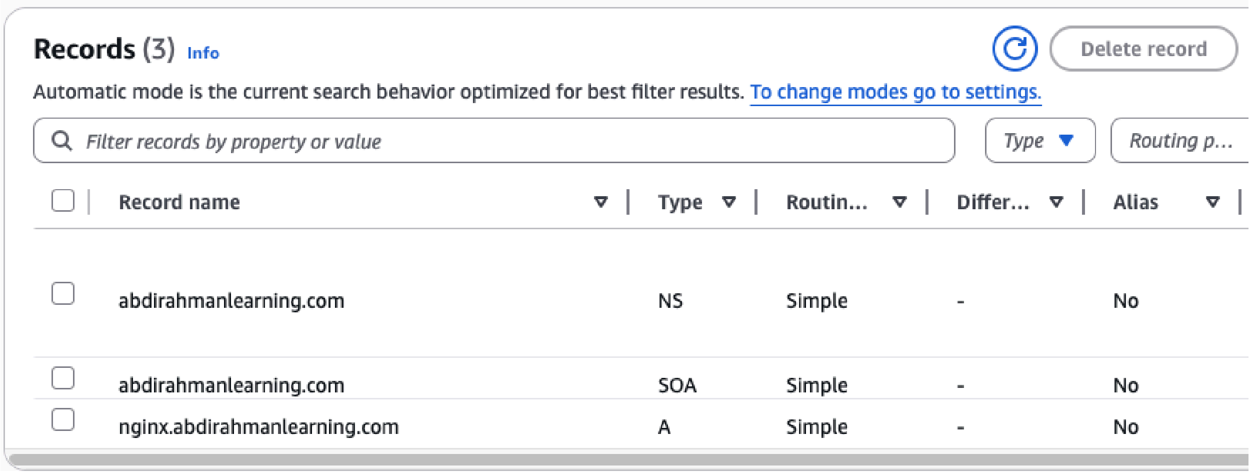
Enter multiple values on separate lines.

TTL (seconds): 300

Recommended values: 60 to 172800 (two days)

Routing policy: Simple routing

We have now created our A record:

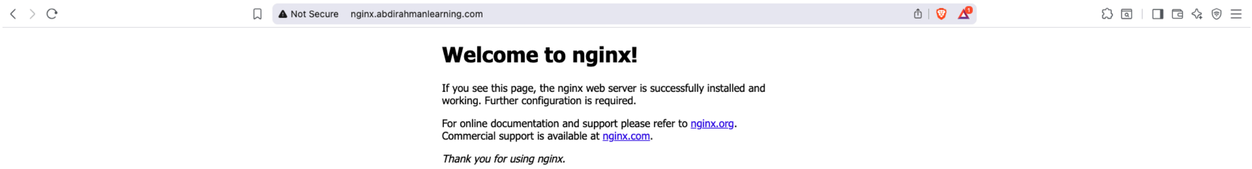


Records (3)

Automatic mode is the current search behavior optimized for best filter results. To change modes go to settings.

□	Record name	Type	Routin…	Differ…	Alias
□	abdirahmanlearning.com	NS	Simple	-	No
□	abdirahmanlearning.com	SOA	Simple	-	No
□	nginx.abdirahmanlearning.com	A	Simple	-	No

You can now copy the A record name “nginx.abdirahmanlearning.com” into a new tab and search for it on the internet. The desired result is the default NGINX HTML page (when we used “curl” in the CLI).



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

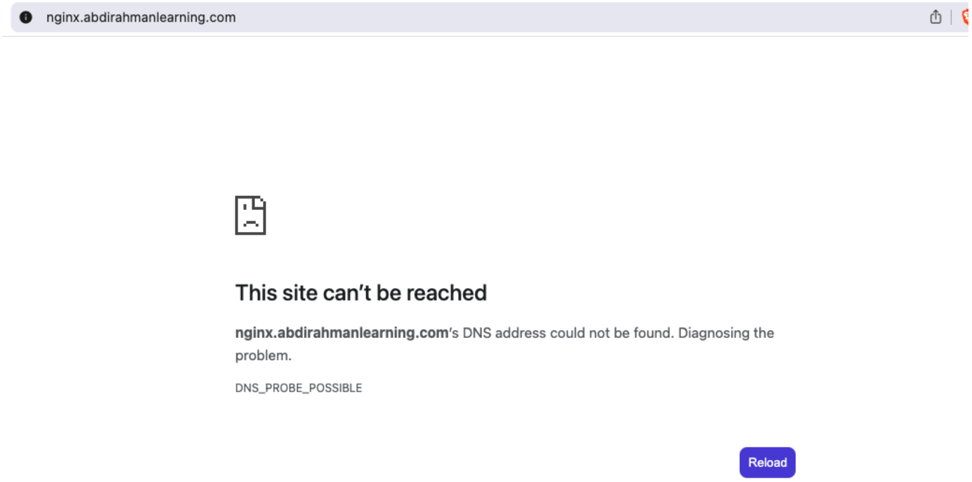
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

We have now completed this project and to summarise, we have just:

1. Registered and configured a custom domain
2. Deployed and secured an EC2 instance
3. Installed and served a web server
4. Set up DNS routing with Route 53

Before closing the management console, ensure to terminate your EC2 instance to make sure you don't incur unwanted costs! (The A record should not work once this is done - see below)



This site can’t be reached

nginx.abdirahmanlearning.com’s DNS address could not be found. Diagnosing the problem.

DNS_PROBE_POSSIBLE