Step-by-Step Deployment: Hosting NGINX on AWS EC2 with Cloudflare Domain

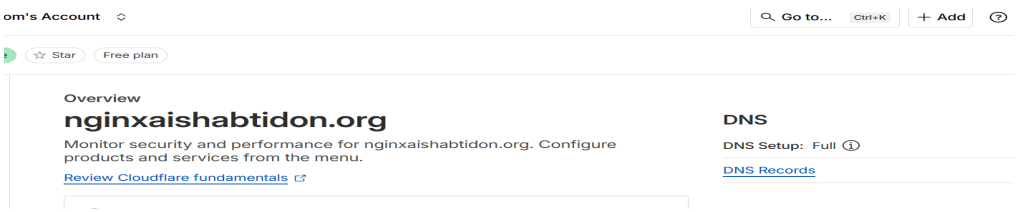
This document summarizes the steps I followed to deploy an NGINX web server on an AWS EC2 instance, connect it to a Cloudflare-managed domain (`nginxaishabtidon.org`), and troubleshoot common DNS and connectivity issues. Each step includes screenshots and comments on what was done, issues faced, and how they were fixed.

---

1. Cloudflare Domain Setup

In this step, I purchased and configured my domain `nginxaishabtidon.org` through Cloudflare.
The domain was successfully added to my Cloudflare account, and Cloudflare confirmed that it's now managing DNS and providing protection.

Screenshot:



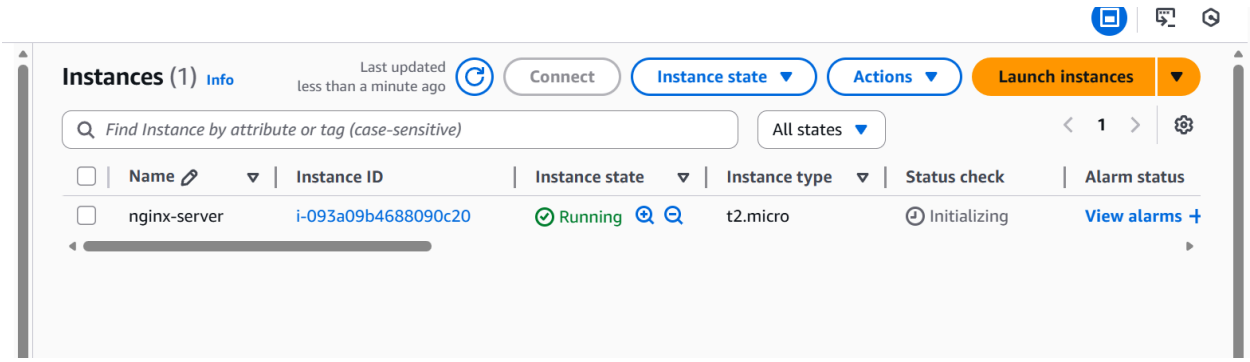
Comment:
At this point, Cloudflare automatically created default DNS records, but I later had to edit them manually to point to my EC2 instance IP.

---

2. Launching EC2 Instance on AWS

Here, I launched a new AWS EC2 instance named `nginx-server` using the **Ubuntu 22.04 LTS** AMI and `t2.micro` instance type (free tier eligible).

Screenshot:



Key actions:

- Security Group was configured to allow inbound:
    - SSH (port 22)
    - HTTP (port 80)
- Instance state shows as **Running**

Comment:
The instance successfully initialized and was assigned the public IPv4 address `98.81.181.142`. This is the IP address used later in the DNS configuration.

---

3. Connecting to EC2 via SSH

I connected to the EC2 instance using my private key file via SSH in PowerShell.

Command used:

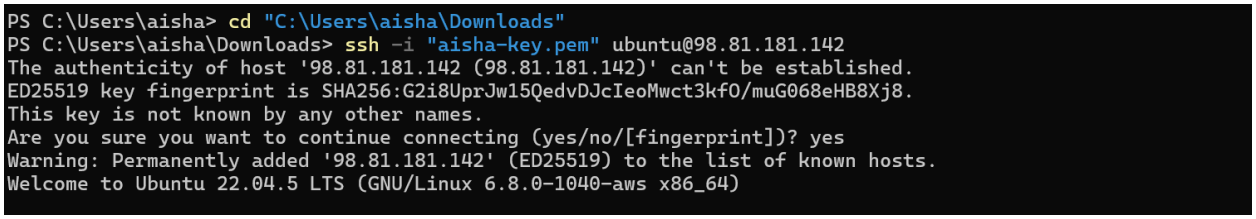
```
ssh -i "aisha-key.pem" ubuntu@98.81.181.142
```

Output summary:

- Verified host authenticity
- Connection successful

- Logged in as Ubuntu user

Screenshot:

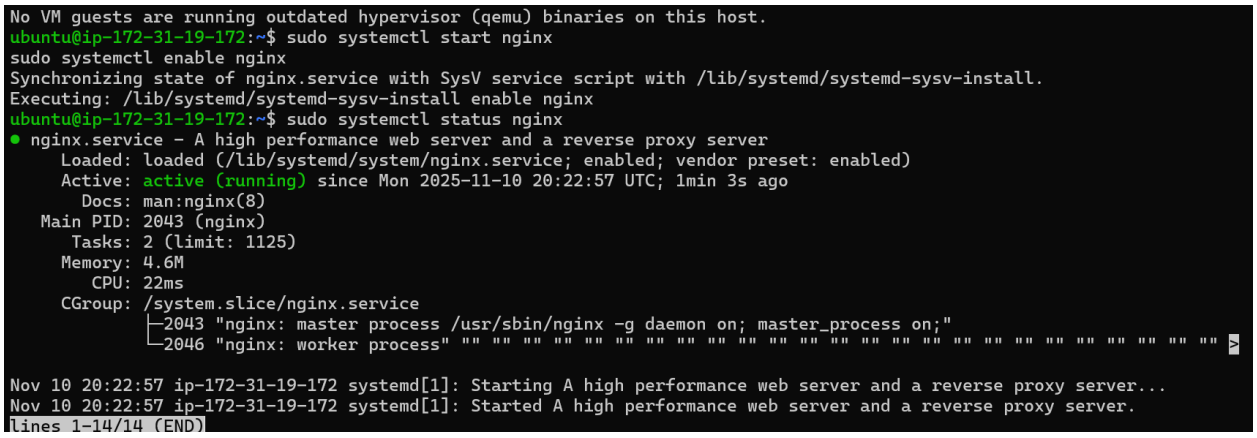


Comment:
This confirms secure SSH connectivity between my local machine and the EC2 instance.

---

4. Installing and Starting NGINX

Screenshot: Screenshot 2025-11-10 202916.png



After connecting to the instance, I installed and started the NGINX web server.

Commands used:

```
sudo apt update -y
sudo apt install nginx -y
sudo systemctl start nginx
sudo systemctl enable nginx
sudo systemctl status nginx
```

Comment:
NGINX was successfully installed and running on port 80. The systemctl output confirmed the service was active.

---

5. Verifying NGINX on Public IP

Screenshot: `Screenshot 2025-11-10 204148.png`

When I accessed the EC2 public IP `98.81.181.142` in my browser, the default NGINX welcome page loaded successfully.

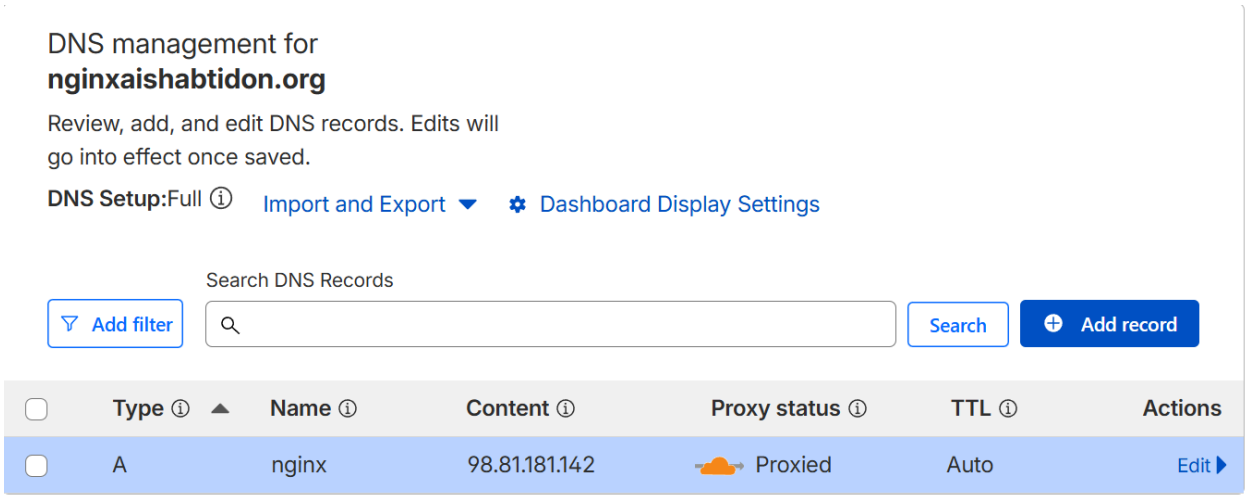
Result:
"Welcome to nginx!" page confirmed that the web server is up and reachable directly via the IP.

Comment:
At this stage, the EC2 and firewall configuration were correct.

---

6. Initial Cloudflare DNS Record Configuration

Screenshot: `Screenshot 2025-11-10 204257.png`



Next, I went back to Cloudflare and created an **A Record** to link the domain to the EC2 instance.

DNS record created:

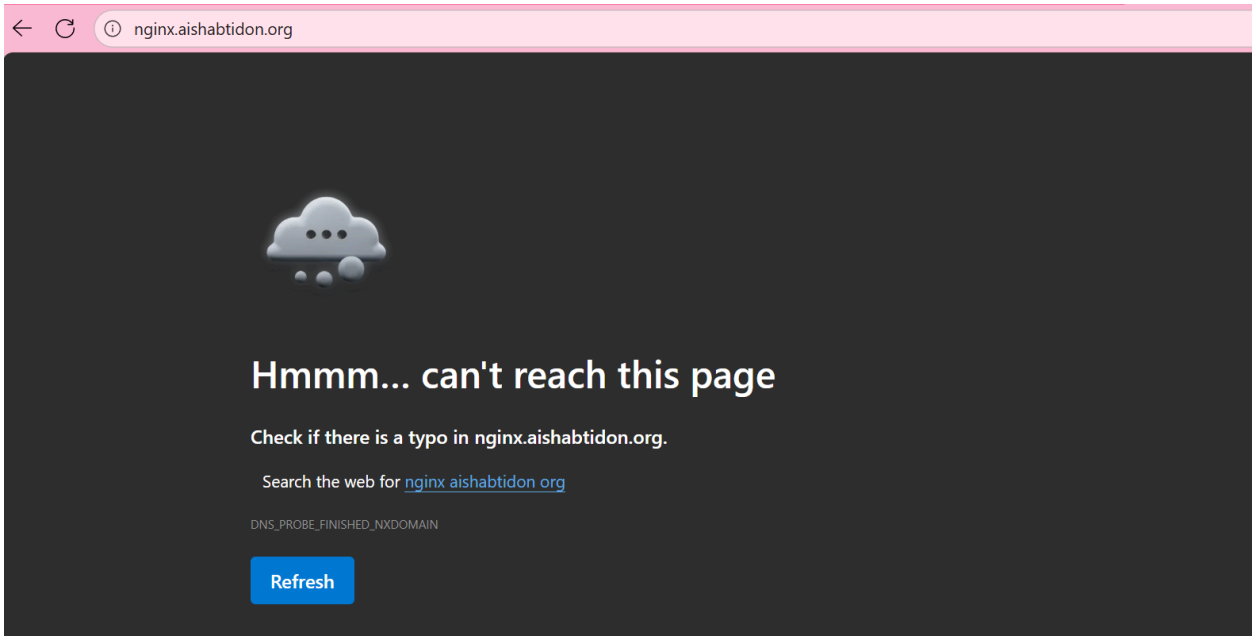
Type	Name	Content	Proxy Status	TTL
A	nginx	98.81.181.142	**Proxied (orange cloud)**	Auto

Comment:
I mistakenly left the proxy status enabled (orange cloud), which caused the domain not to resolve correctly later.

---

7. Browser Error: Site Not Reachable

Screenshot: `Screenshot 2025-11-10 205618.png`



After saving the DNS record, I tried to access the domain `nginx.aishabtidon.org`, but it failed to load.

Error message:

```
Hmmm... can't reach this page
DNS_PROBE_FINISHED_NXDOMAIN
```

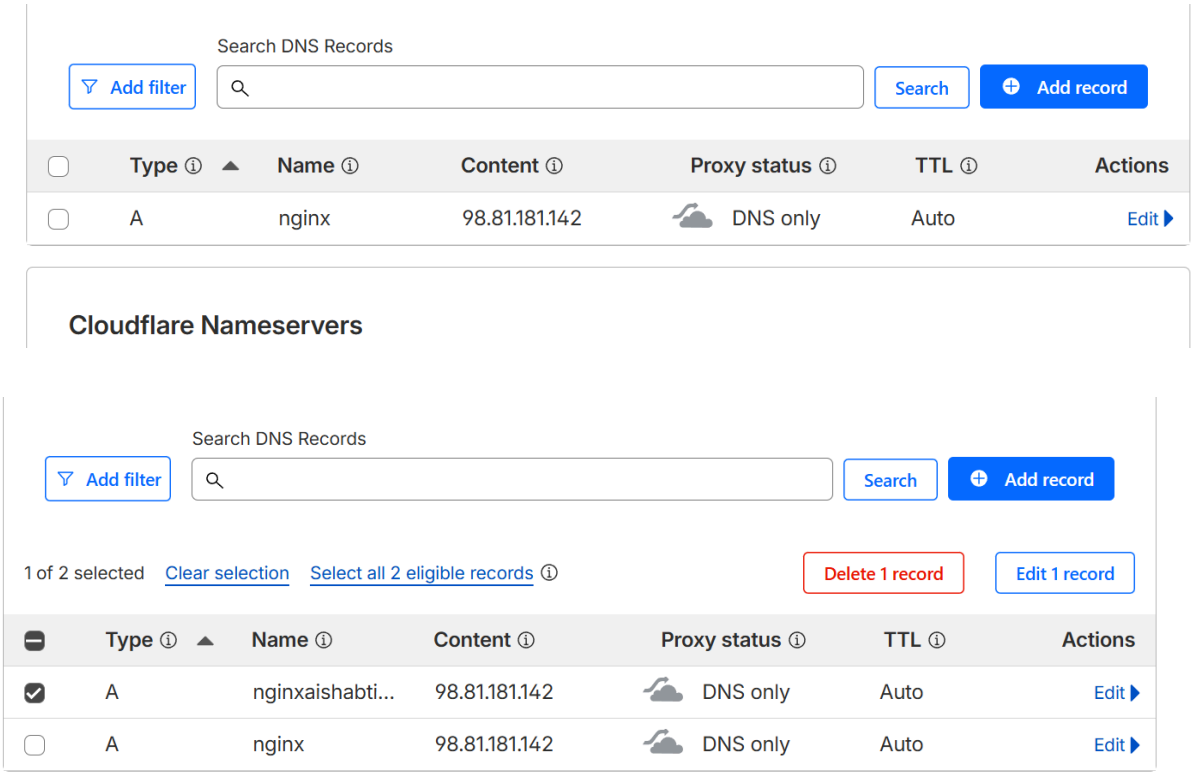
Cause:
Cloudflare's proxy mode interfered because there was no SSL or proper routing configured yet.

Fix:
I changed the proxy status from **Proxied** (orange cloud) to **DNS only** (gray cloud).

---

8. Updated DNS Record: DNS Only Mode

Screenshot: `Screenshot 2025-11-10 210119.png`



After disabling proxy, the DNS record now showed a gray cloud icon, indicating **DNS only**.

New DNS record:

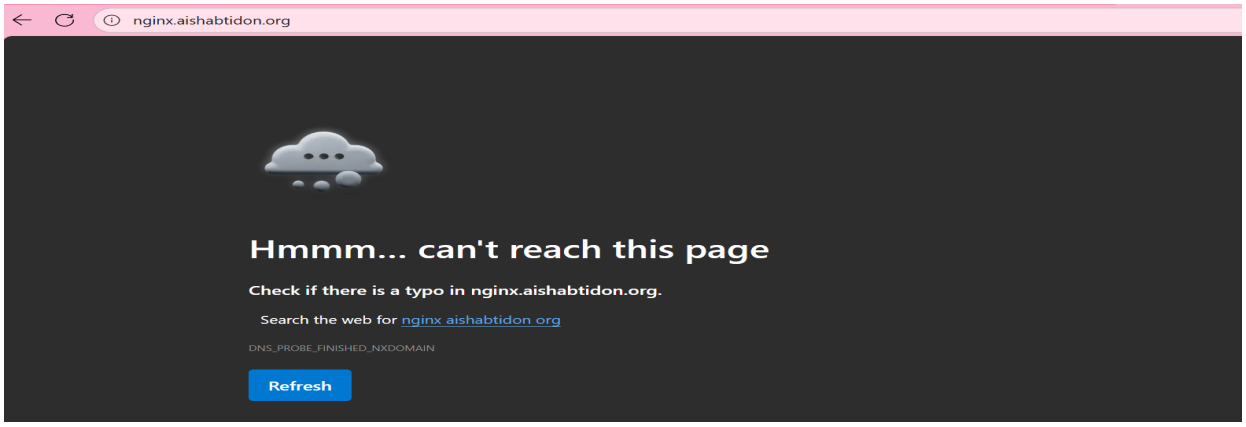
Type	Name	Content	Proxy Status	TTL
A	nginx	98.81.181.142	DNS only	Auto

Comment:
This setting allows traffic to go directly to my EC2 instance instead of routing through Cloudflare's network.

---

9. DNS Still Not Resolving (Cached Issue)

Screenshot: `Screenshot 2025-11-10 210142.png`

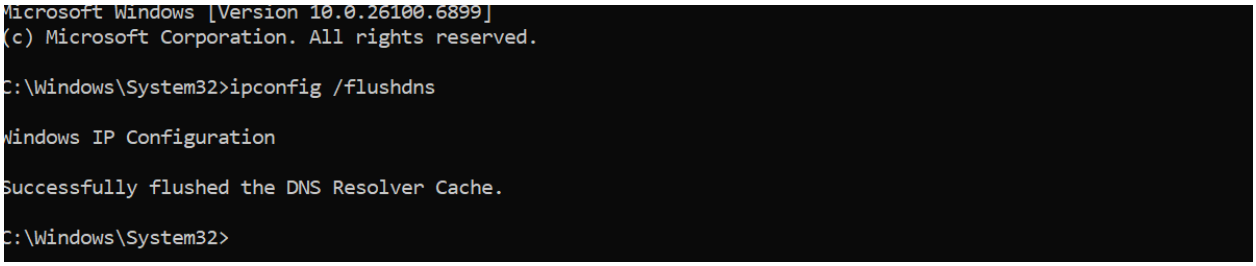


Even after fixing the DNS settings, the browser still showed the same NXDOMAIN error.
This indicated a **local DNS cache issue**, where the old Cloudflare record was still being used by my computer.

---

10. Flushing DNS Cache on Windows

Screenshot: `Screenshot 2025-11-10 211002.png`



To force my computer to fetch the new DNS record, I flushed the local DNS cache using:

```
ipconfig /flushdns
```

Output:

```
Successfully flushed the DNS Resolver Cache.
```

Comment:
After this, the domain started resolving correctly.
Both `nginxaishabtidon.org` and `nginx.aishabtidon.org` successfully displayed the NGINX welcome page.

Final Notes

Lessons Learned

1. Always check **proxy mode** in Cloudflare — “DNS only” is required when you don’t use SSL yet.

2. DNS propagation can take several minutes globally; use tools like dnschecker.org.

3. Browser or OS-level DNS caching can cause delays even after fixing records — flushing cache helps.

4. Security groups and firewall rules must always allow **port 80 (HTTP)** inbound.