Placement Empowerment Program

Cloud Computing and DevOps Centre

Implement DNS for Your Application: Set up a DNS record to map your web application's IP or load balancer to a domain name.

Name: Gopinath M Department: ADS

Introduction

In cloud computing, establishing a proper **Domain Name System (DNS)** configuration is essential for ensuring that applications are accessible over the internet. AWS offers **Route 53**, a highly available and scalable DNS web service, which allows users to manage domain names and point them to AWS resources like EC2 instances. This Proof of Concept (PoC) demonstrates the process of creating and configuring a DNS record in **AWS Route 53**, pointing it to a web server hosted on an **EC2 instance**, thus making the web application accessible via a custom domain name.

Overview

This PoC involves:

1. **Launching an EC2 instance** to serve a web application.
2. **Setting up a web server** (Apache or Nginx) on the EC2 instance to host the application.
3. **Creating a hosted zone** in AWS Route 53 for a custom domain (e.g., myapp.local).
4. **Configuring an A record** in Route 53 to map the domain to the EC2 instance's public IP.
5. **Modifying the hosts file** on a local machine to test the custom domain name before making it publicly available.
6. **Testing the configuration** by accessing the application using the custom domain name.

Objective

The main objectives of this PoC are:

1. **Familiarize with Route 53 DNS configuration**: Understand how to use AWS Route 53 to manage domain names and map them to cloud resources.

2. **Learn EC2 setup and configuration**: Gain hands-on experience in launching an EC2 instance and configuring a web server to serve a web application.

3. **Enable custom domain access**: Configure a custom domain name to point to the EC2 instance, ensuring that the web application is easily accessible through the domain.

4. **Test and verify the configuration**: Ensure that the domain correctly points to the EC2 instance by testing it in a browser and troubleshooting any issues.

Importance

1. **Improves Web Access**: Provides a user-friendly way to access applications via a custom domain.

2. **Scalable DNS Solution**: Route 53 offers scalable and reliable DNS management.

3. **Hands-on Cloud Skills**: Essential for cloud architects and developers to work with AWS services.

4. **Cost-Effective Testing**: Utilizes AWS Free Tier for testing without incurring costs.

Step-by-Step Overview

Step 1:

1. Go to AWS Management Console.
2. Enter your username and password to log in.

Step 2:

Launch an instance named **route instance** .

Configure Security Group:

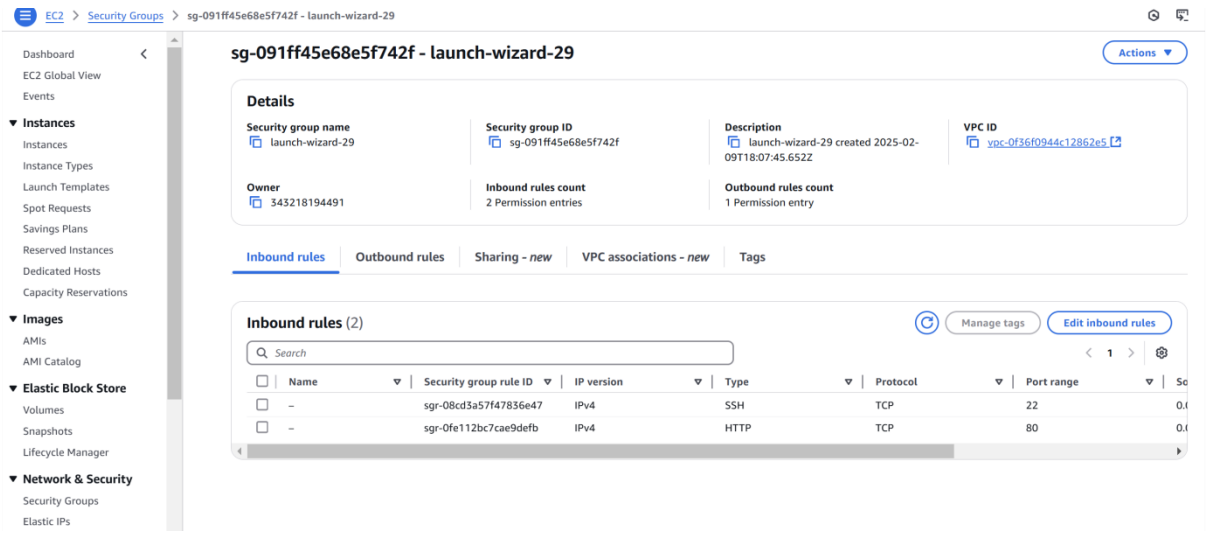
Add a **new security group** with a rule for **HTTP** (port 80) and **SSH** (port 22).

For HTTP, set the source to **Anywhere (0.0.0.0/0)** to allow access via the web.

For SSH, set the source to your **IP** (recommended for security), or use **Anywhere** for now.

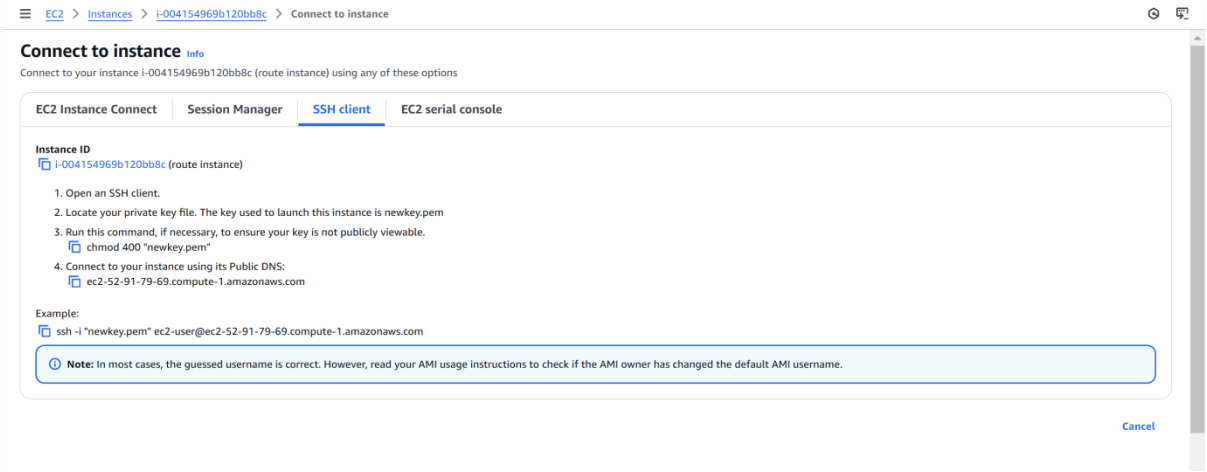
Create a Key Pair (or use an existing one) and download the key file (.pem).

Review and click **Launch.**



Step 3:

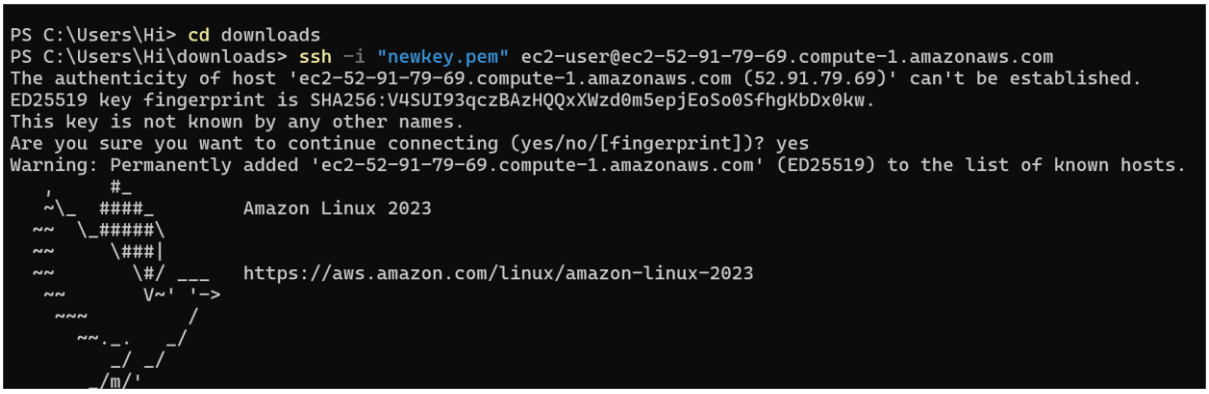
Click the 'Connect' option on your launched instance, go to the SSH client section, and copy the command provided under the 'Example' section.



Step 4:

Open PowerShell, navigate to the 'Downloads' directory where the downloaded key pair is located using the **cd Downloads** command

Paste the command copied from the EC2 Connect's SSH client section, replace the key pair name with your downloaded key (e.g., new.pem), press Enter, and type 'yes' when prompted.



Step 5:

Install Apache :

sudo yum install httpd -y

```
[ec2-user@ip-172-31-82-55 ~]$ sudo yum install httpd -y
```

Step 6:

Start Apache**:**

sudo systemctl start httpd Make

Apache start on boot**:**

sudo systemctl enable httpd

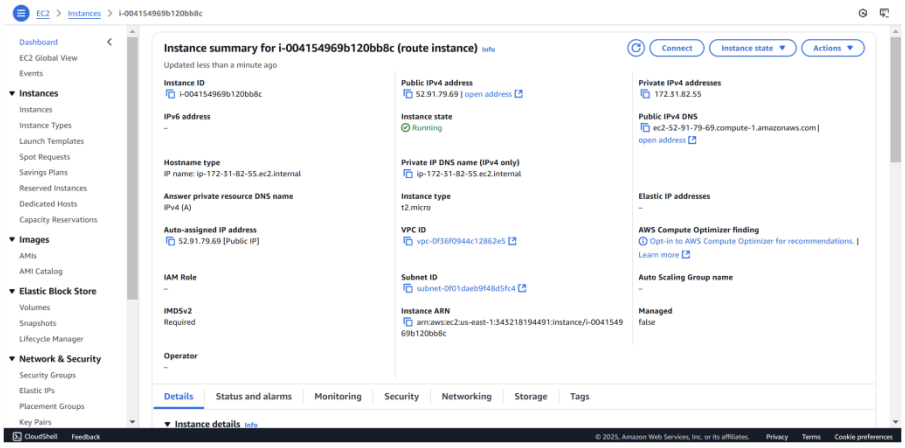
```
[ec2-user@ip-172-31-82-55 ~]$ sudo systemctl start httpd
[ec2-user@ip-172-31-82-55 ~]$ sudo systemctl enable httpd
```

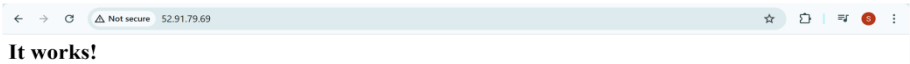
Step 7:

Verify Apache is running:

In your browser, enter the **EC2 public IP** (e.g., http://<your-ec2public-ip>).

You should see the **Apache default page**. This means your EC2 instance is set up to serve websites.





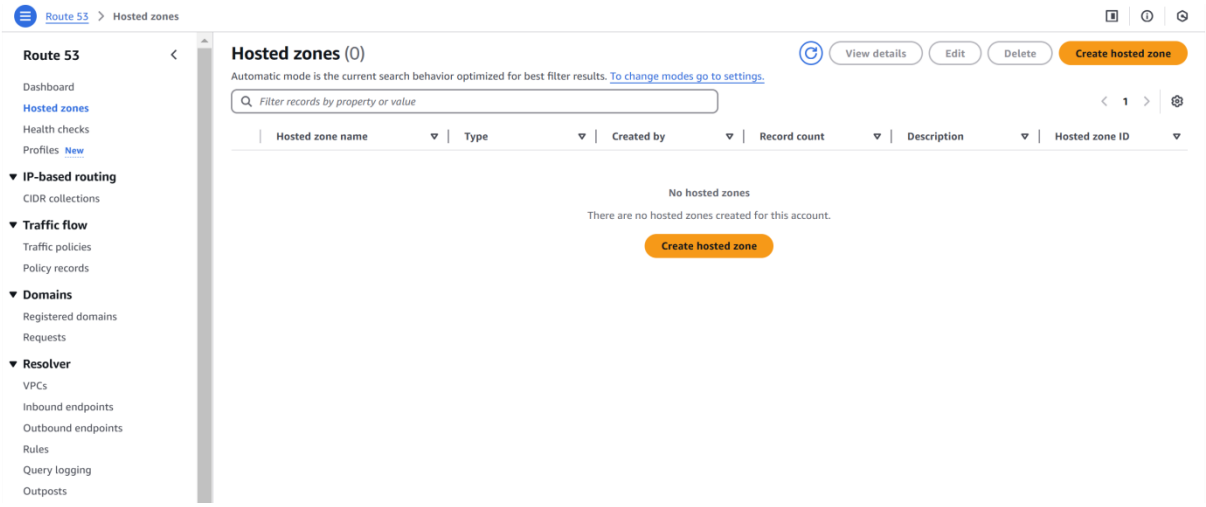
It works!

Step 8:

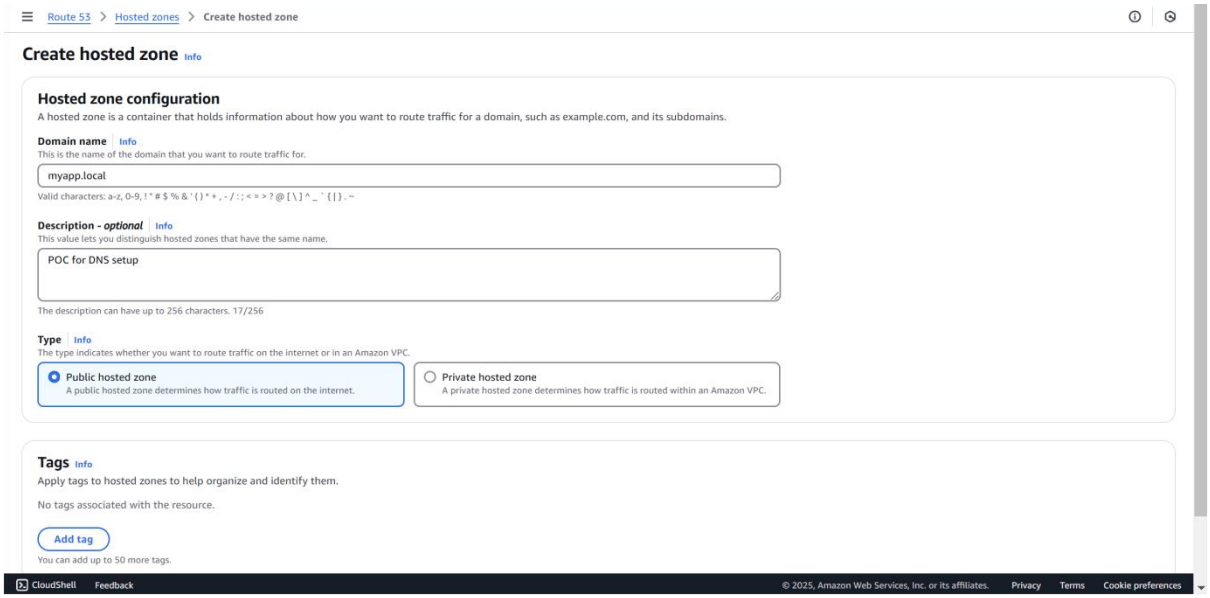
In the AWS Console, search for **Route 53** and select it.

Step 8:

1. Click on **Create hosted zone**.
2. Enter a **Domain Name** (e.g., myapp.local).
3. Set the **Type** to **Public Hosted Zone**.
4. Click **Create hosted zone**.



Step 9:



1. In your hosted zone, click **Create Record**.

2. **Record Name**: Leave it empty for the root domain (myapp.local),

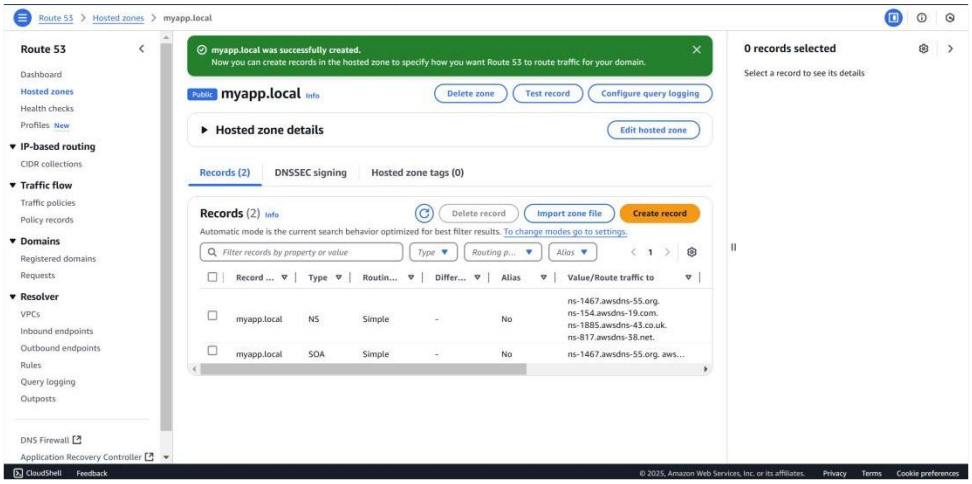
3. **Record Type**: Select **A – IPv4 address**.

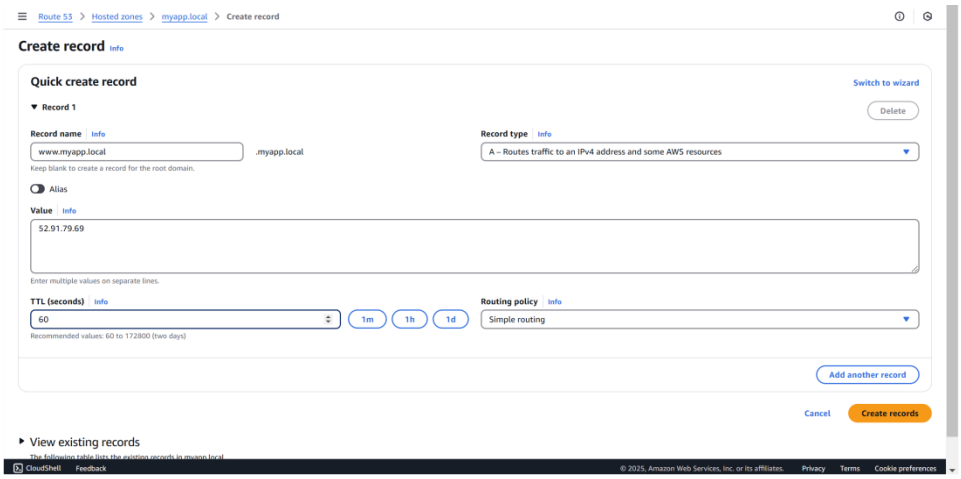
4. **Value**: Enter the **Public IP** of your EC2 instance.

5. **TTL**: Set to 60 seconds.

6. Click **Create records**.

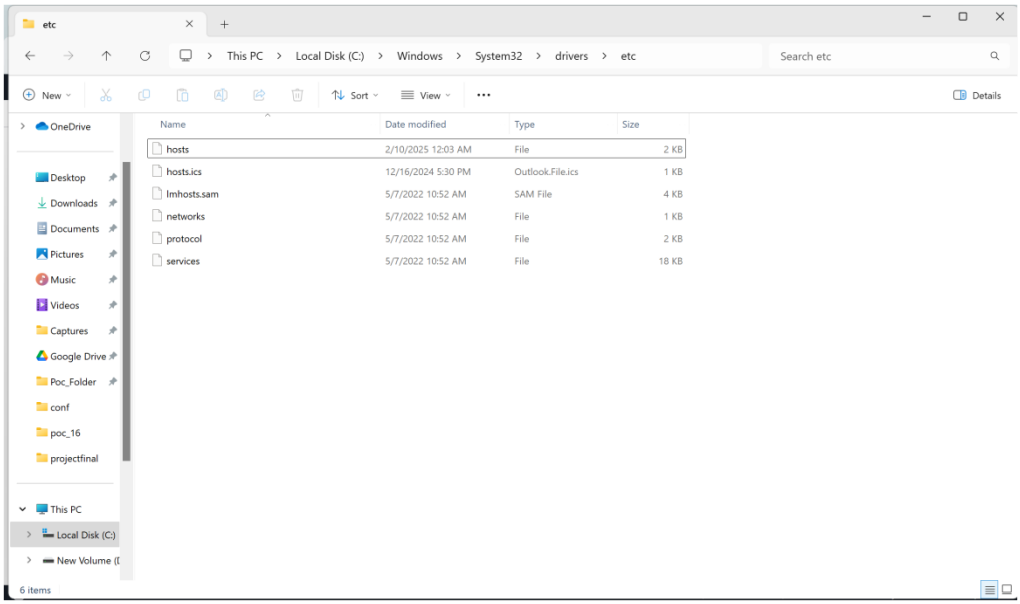
Step 10:





1. Go to **FileExplorer > Open**.
2. Navigate to: C:\Windows\System32\drivers\etc.
3. In the file name field, type hosts and press **Enter**.

Step 11:



Step 12:

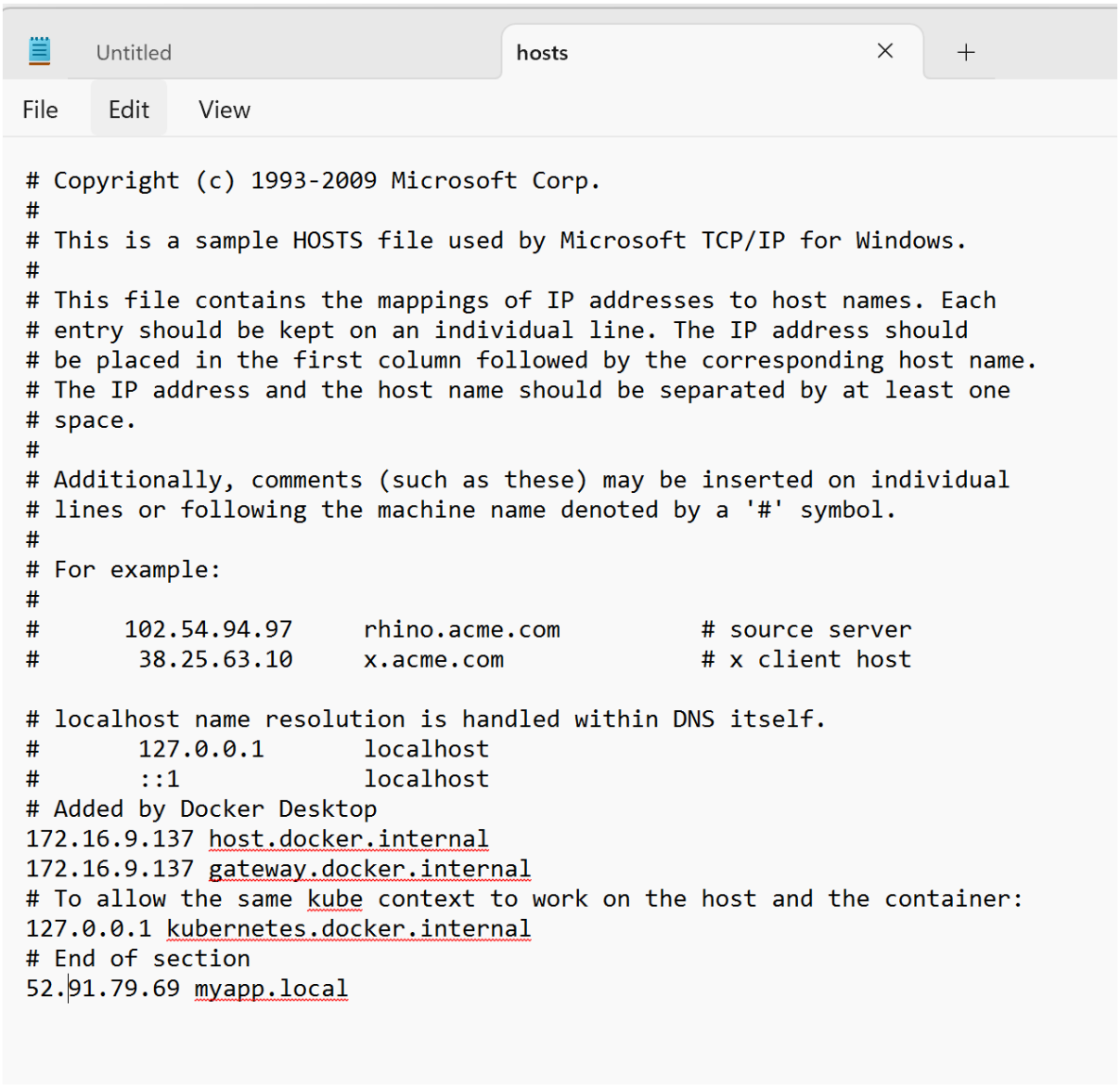
1. At the bottom of the file, add:

   <Your EC2 Public IP> myapp.local

   Replace <Your EC2 Public IP> with the public IP you copied.

   (Eg: 52.91.79.69 myapp.local)

2. Save the file and close Notepad.

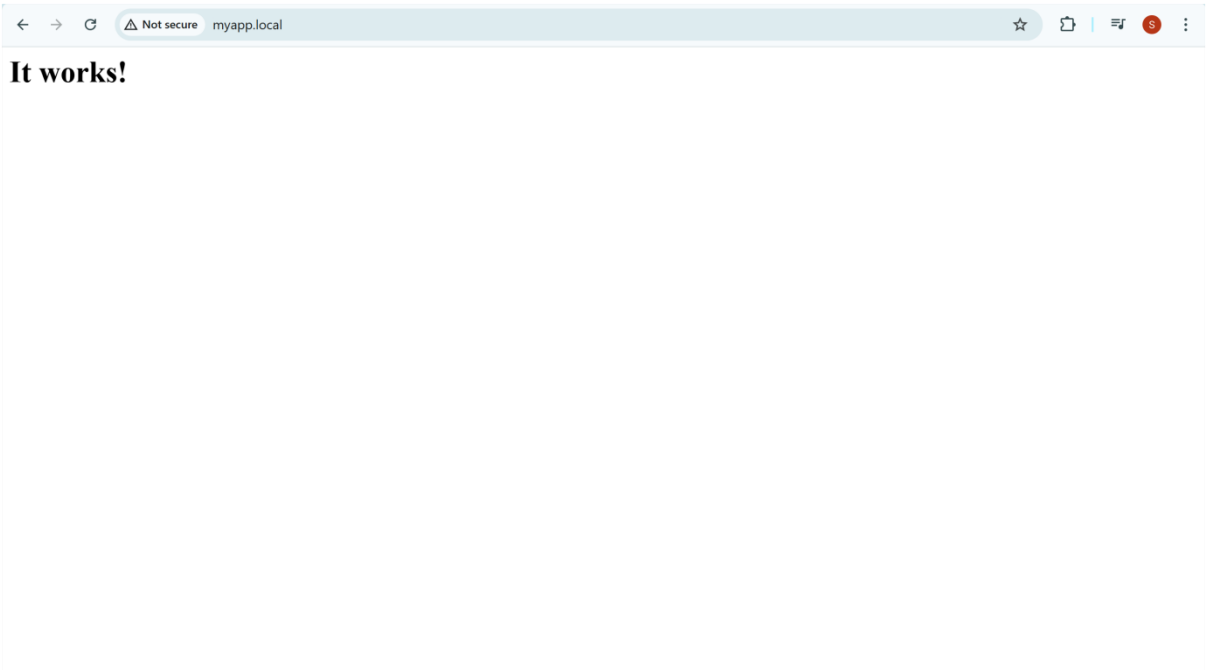


Step 12:

Open your **web browser**.

Type myapp.local in the address bar and press **Enter**.

You should see the Apache default page



Outcome

By completing this PoC of configuring DNS for your application using AWS Route 53 and EC2, you will:

1. Launch and configure an EC2 instance with a web server (e.g., Apache or Nginx).
2. Deploy a sample web application on the EC2 instance and ensure it is accessible via the instance's public IP.
3. Create a hosted zone in AWS Route 53 for DNS management.
4. Set up an A record in Route 53 to map your custom domain (e.g., myapp.local) to the EC2 instance’s public IP.

4. Modify the local hosts file on your system to resolve the custom domain name locally for testing.

5. Successfully access your web application using the custom domain name in a browser.