

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: Sharon jenifer S

Department:
CSE

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.

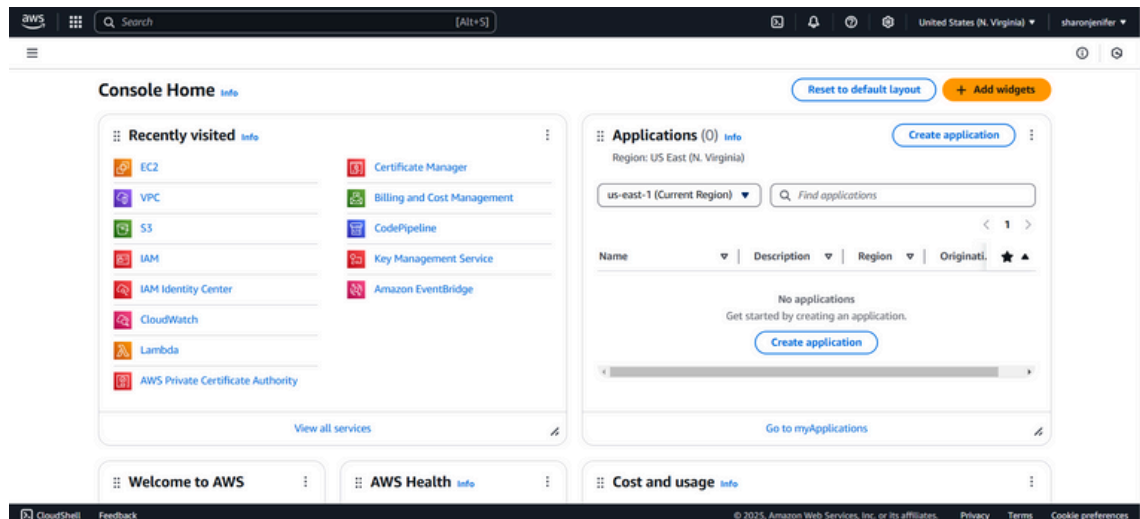
Importance

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.

Successfully initiated termination (deletion) of i-08ba3c1860e5460bd,i-0d31b57c3abc4f6e7

Instances (1/1) Info

Find Instance by attribute or tag (case-sensitive)

Running

<input checked="" type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
<input checked="" type="checkbox"/>	route instance	i-0050441094279146e	Running	t2.micro	Initializing	View alarms +	us-east-1c	ec2-3-83-2

i-0050441094279146e (route instance)

Details Status and alarms Monitoring Security Networking Storage Tags

Instance summary Info

Instance details Info

AMI ID
ami-05b10c08d247fb927

AMI name
al2023-ami-2023.6.20250218.2-kernel-6.1-x86_64

Stop protection
Disabled

Monitoring
disabled

Allowed image
-

Launch time
Wed Mar 05 2025 12:01:27 GMT+05:30 (India Standard)

Platform details
Linux/UNIX

Termination protection
Disabled

AMI location
amazon/al2023-ami-2023.6.20250218.2-kernel-6.1-x8

sg-05ab4c96f4a904ef1 - launch-wizard-18

Details

Security group name
launch-wizard-18

Security group ID
sg-05ab4c96f4a904ef1

Description
launch-wizard-18 created 2025-03-05T06:30:52.076Z

VPC ID
vpc-068af34fe69434e5f

Owner
867344462005

Inbound rules count
2 Permission entries

Outbound rules count
1 Permission entry

Inbound rules Outbound rules Sharing - new VPC associations - new Tags

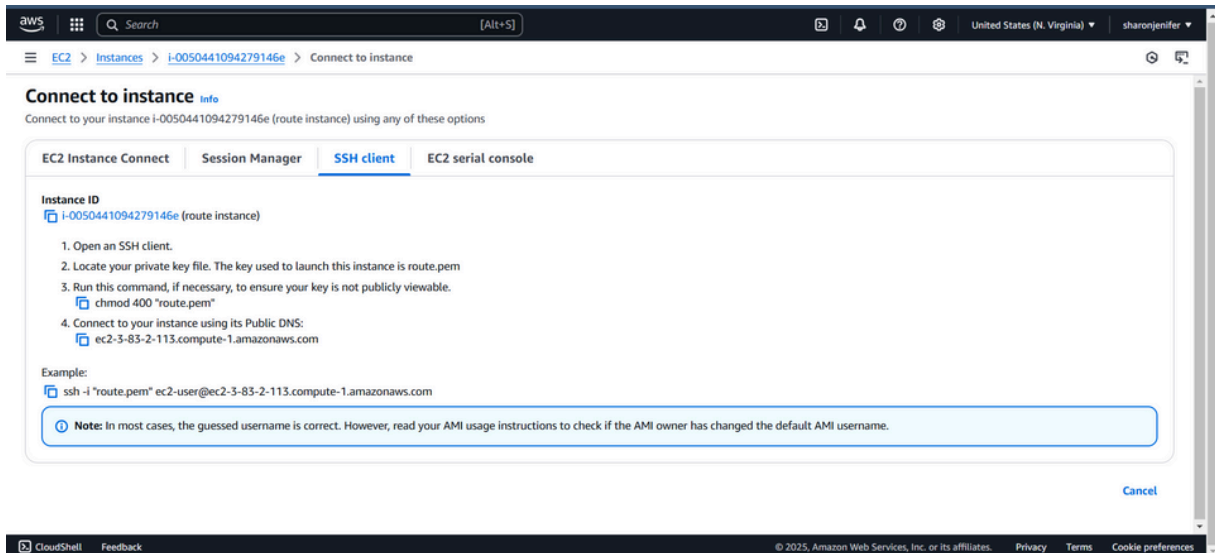
Inbound rules (2)

Search

<input type="checkbox"/>	Name	Security group rule ID	IP version	Type	Protocol	Port range
<input type="checkbox"/>	-	sgr-09aff08d9da296568	IPv4	HTTP	TCP	80
<input type="checkbox"/>	-	sgr-02f291e862a4dfa30	IPv4	SSH	TCP	22

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.

```
PS C:\Users\sharo> cd downloads
PS C:\Users\sharo\downloads> ssh -i "route.pem" ec2-user@ec2-3-83-2-113.compute-1.amazonaws.com
The authenticity of host 'ec2-3-83-2-113.compute-1.amazonaws.com (3.83.2.113)' can't be established.
ED25519 key fingerprint is SHA256://W0De90qh5GyN8GdGAGwa4pvZqnMGKwDuQJnsSJ16g.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-83-2-113.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

#
~\_ #####_ Amazon Linux 2023
pre pre \_#####\
pre pre \###|
pre pre \#/ --> https://aws.amazon.com/linux/amazon-linux-2023
pre pre V'-'
pre pre -
pre pre -/_/ -/_/ -/_/
pre pre -/_/ -/_/ -/_/
[ec2-user@in-172-31-95-253 ~]$
```

Step 5:

Install Apache :

```
sudo yum install httpd -y
```

```
[ec2-user@ip-172-31-95-253 ~]$ sudo yum install httpd -y
```

Step 6:

Start Apache: `sudo`

`systemctl start httpd` Make

Apache start on boot: `sudo`

`systemctl enable httpd`

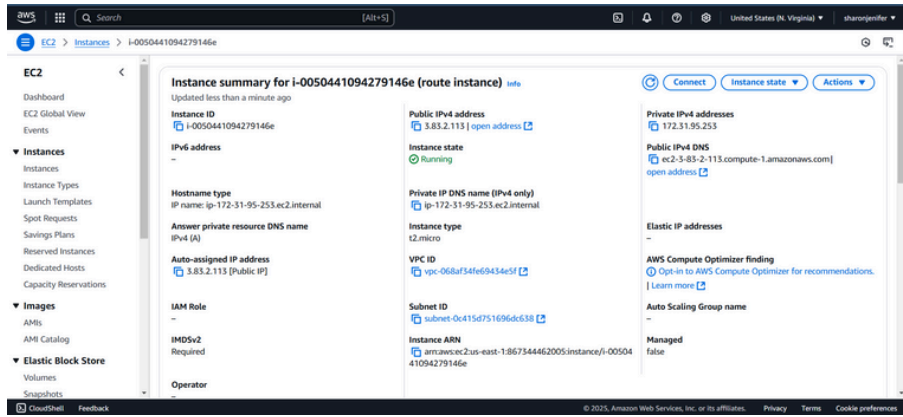
```
[ec2-user@ip-172-31-95-253 ~]$ sudo systemctl start httpd  
[ec2-user@ip-172-31-95-253 ~]$ sudo systemctl enable httpd
```

Step 7:

Verify Apache is running:

In your browser, enter the EC2 public IP (e.g., `http://<your-ec2-public-ip>`).

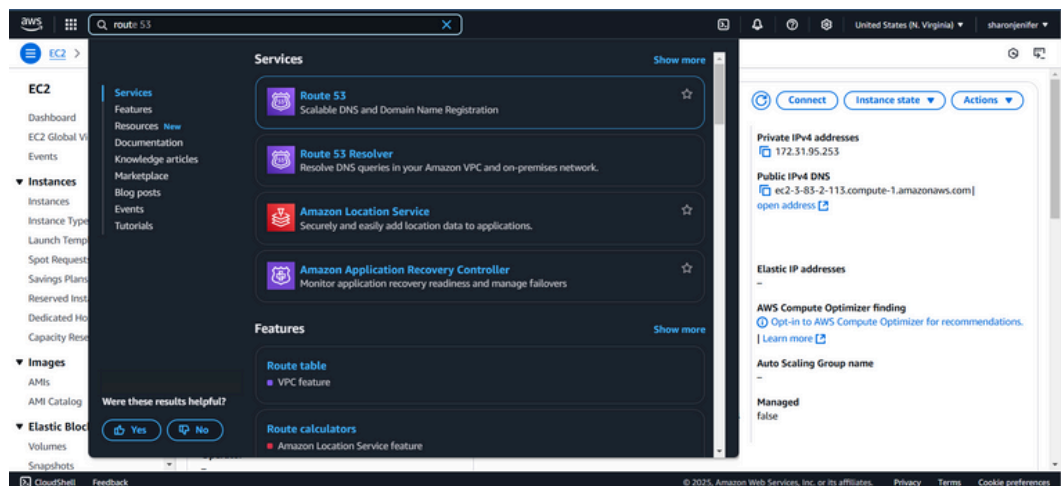
You should see the Apache default page. This means your EC2 instance is set up to serve websites.



It works!

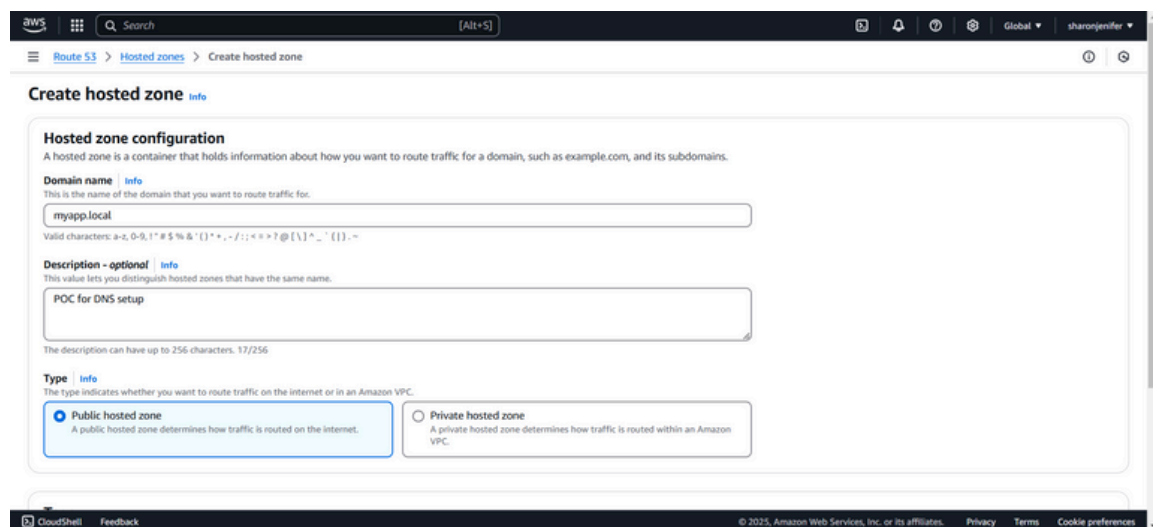
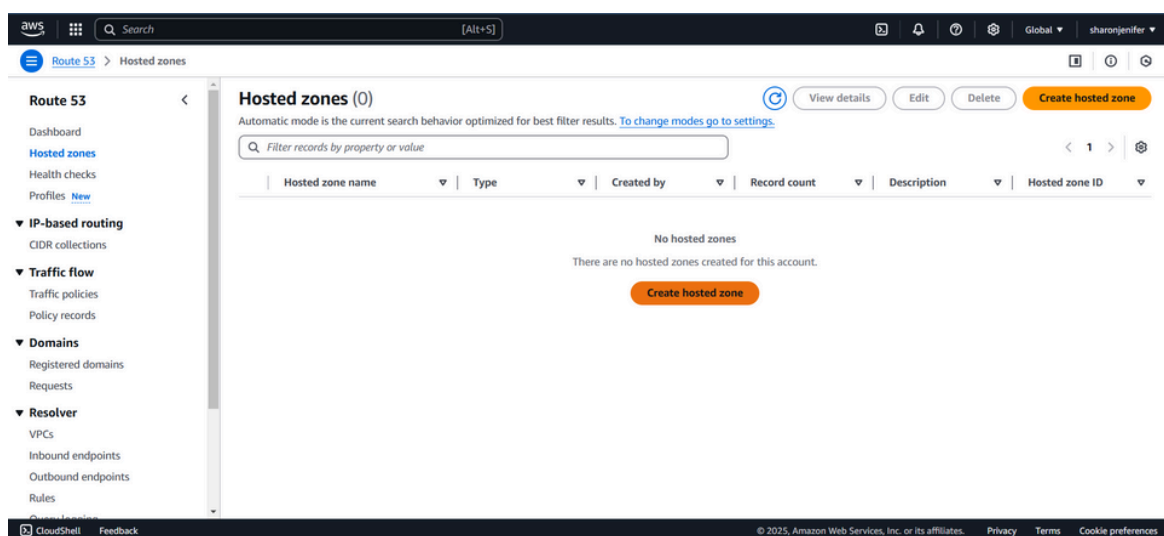
Step 8:

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



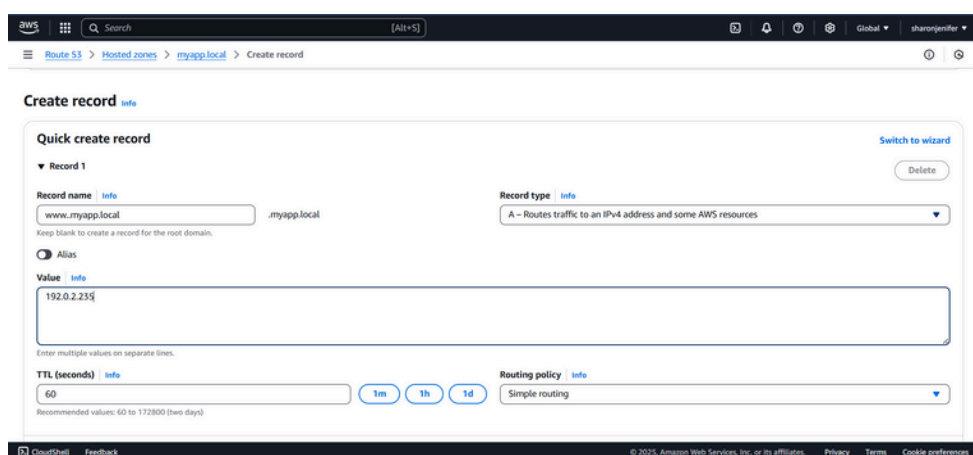
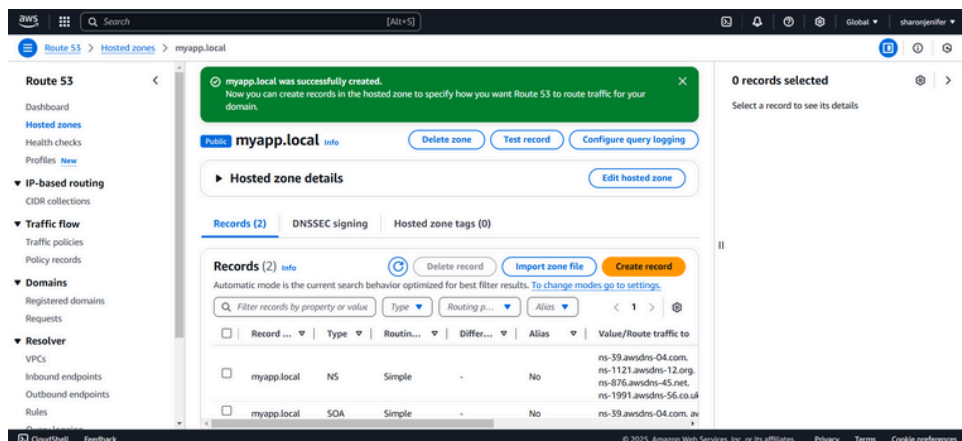
Step 9:

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.



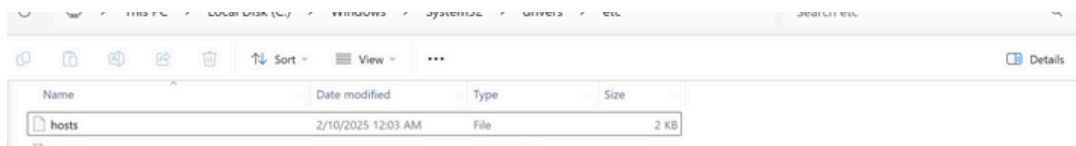
Step 10:

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 11:

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

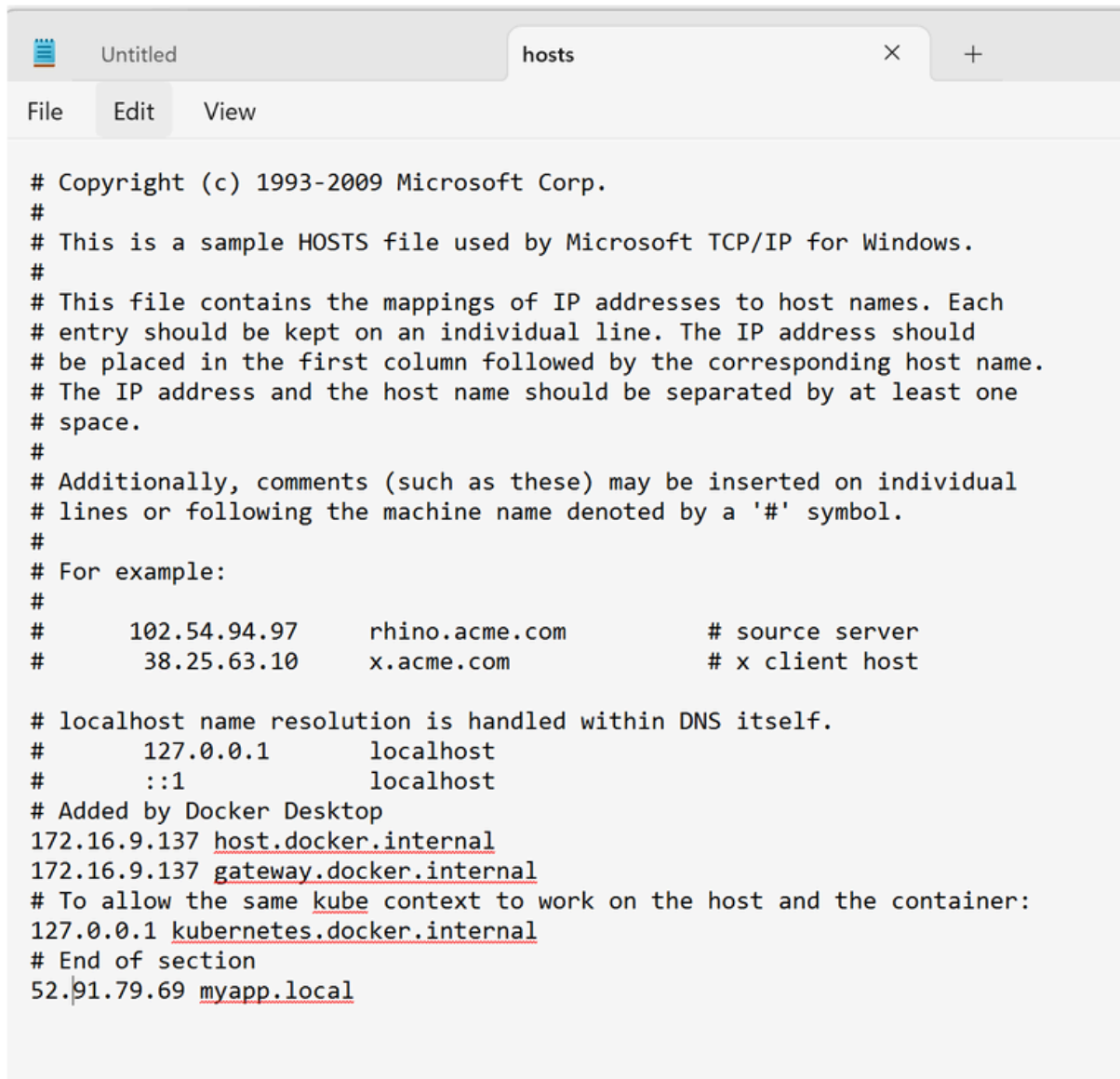


Step 12:

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.



```
# Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#       102.54.94.97       rhino.acme.com           # source server
#       38.25.63.10       x.acme.com               # x client host

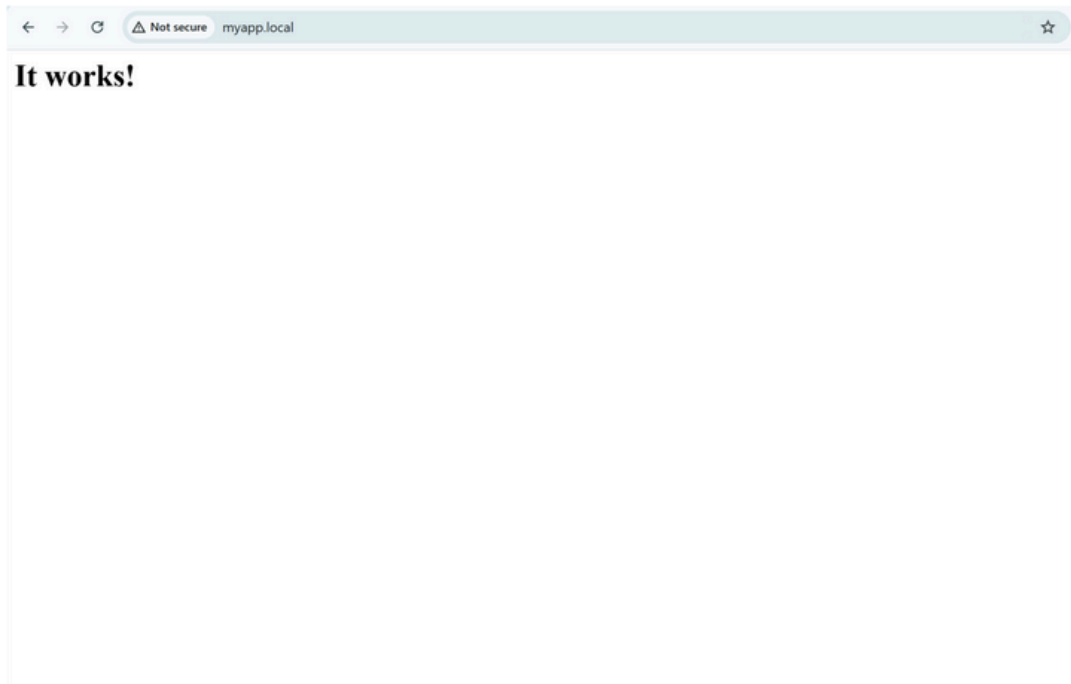
# localhost name resolution is handled within DNS itself.
#       127.0.0.1         localhost
#       ::1               localhost
# Added by Docker Desktop
172.16.9.137 host.docker.internal
172.16.9.137 gateway.docker.internal
# To allow the same kube context to work on the host and the container:
127.0.0.1 kubernetes.docker.internal
# End of section
52.91.79.69 myapp.local
```

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.