

03-03-2025

TASK ON R53, CDN (CLOUD FRONT), S3

1) Configure VPC peering in cross regions:

The screenshot displays the AWS Management Console interface for configuring VPC peering across regions. The top section shows the 'VPC dashboard' with a sidebar menu including 'Virtual private cloud', 'Security', and 'PrivateLink and'. The main content area is titled 'pcx-080bd0b99051ff471 / peering-mumbai-verginia'. A green notification banner at the top states: 'A VPC peering connection pcx-080bd0b99051ff471 / peering-mumbai-verginia has been requested. Remember to change your region to us-east-1 to accept the peering connection.' Below this, the 'Details' tab is active, showing the following information:

- Requester owner ID:** 971422718404
- Peering connection ID:** pcx-080bd0b99051ff471
- Status:** Initiating Request to 971422718404
- Expiration time:** Monday, March 3, 2025 at 19:17:18 GMT+5:30
- Accepter owner ID:** 971422718404
- Requester VPC:** vpc-08197ebb402e4e1f4 / my-vpc
- Requester CIDRs:** 192.168.0.0/16
- Requester Region:** Mumbai (ap-south-1)
- VPC Peering connection ARN:** arn:aws:ec2:ap-south-1:971422718404:vpc-peering-connection/pcx-080bd0b99051ff471
- Accepter VPC:** vpc-021be08e8e392964d
- Accepter CIDRs:** -
- Accepter Region:** N. Virginia (us-east-1)

The 'DNS settings' tab is also visible, showing 'Requester VPC (vpc-08197ebb402e4e1f4 / my-vpc)' and a checkbox for 'Allow accepter VPC to resolve DNS of hosts in requester VPC to private IP addresses'. Below the screenshot, the 'Peering connections (1)' table is shown, listing the established connection:

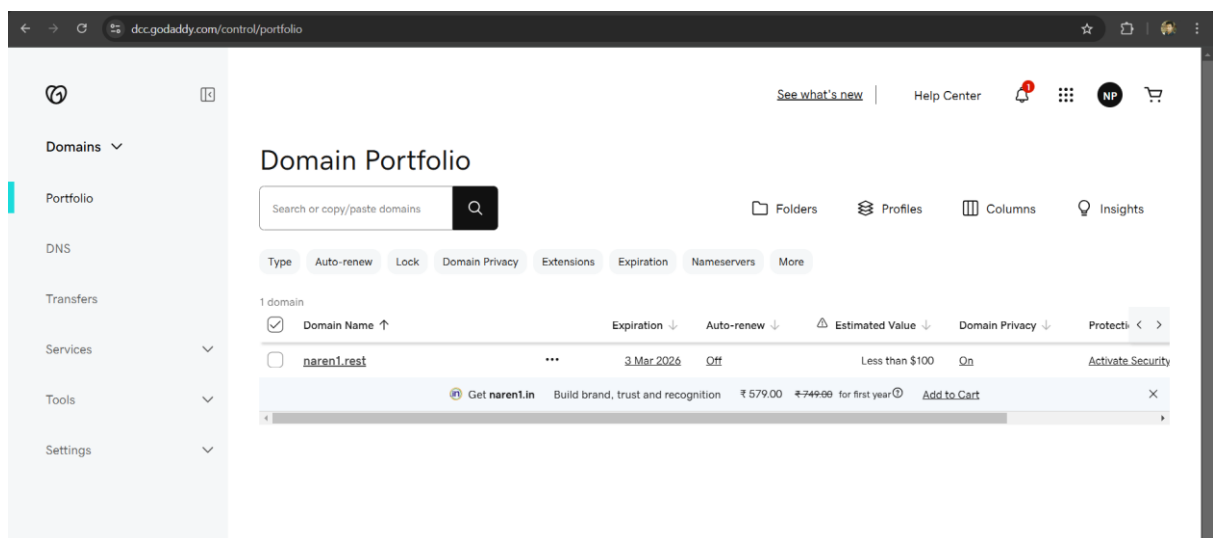
Name	Peering connection ID	Status	Requester VPC	Accepter VPC
-	pcx-080bd0b99051ff471	Active	vpc-08197ebb402e4e1f4	vpc-021be08e8e392964d

```
root@ip-192-168-0-13:~#
naren@narendar MINGW64 /d/downloads
$ ssh -i "ec2-vpc.pem" ec2-user@3.109.183.71
Last login: Mon Feb 24 10:30:38 2025 from 103.143.169.218
#
#####
      Amazon Linux 2
#####
      AL2 End of Life is 2026-06-30.
#####
      A newer version of Amazon Linux is available!
      Amazon Linux 2023, GA and supported until 2028-03-15.
      https://aws.amazon.com/linux/amazon-linux-2023/

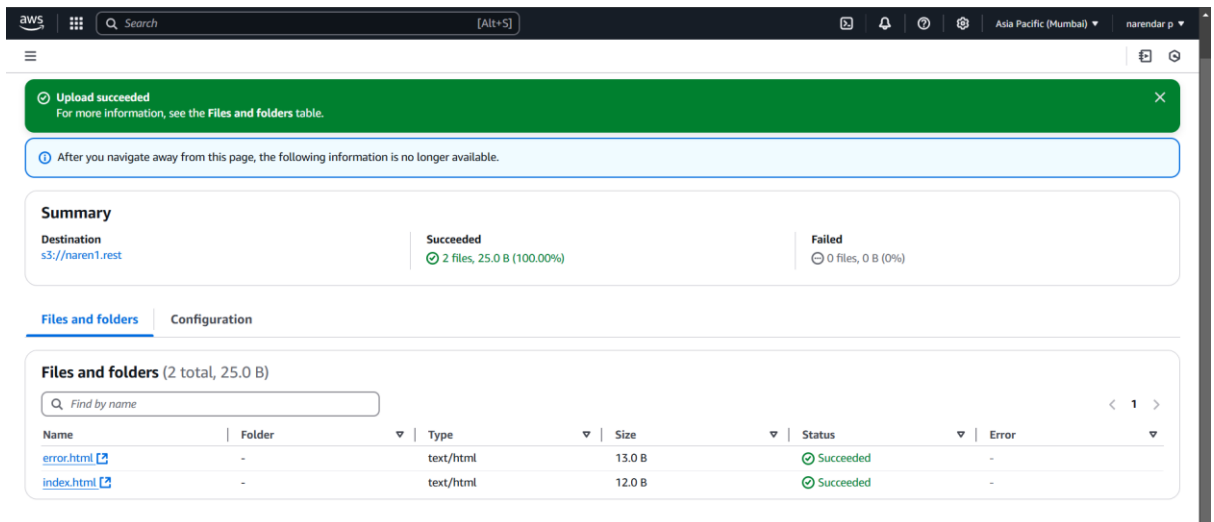
[ec2-user@ip-192-168-0-13 ~]$ sudo -i
[root@ip-192-168-0-13 ~]# ping 172.31.83.134
PING 172.31.83.134 (172.31.83.134) 56(84) bytes of data.
64 bytes from 172.31.83.134: icmp_seq=1 ttl=255 time=185 ms
64 bytes from 172.31.83.134: icmp_seq=2 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=3 ttl=255 time=185 ms
64 bytes from 172.31.83.134: icmp_seq=4 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=5 ttl=255 time=185 ms
64 bytes from 172.31.83.134: icmp_seq=6 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=7 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=8 ttl=255 time=185 ms
64 bytes from 172.31.83.134: icmp_seq=9 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=10 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=11 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=12 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=13 ttl=255 time=186 ms
64 bytes from 172.31.83.134: icmp_seq=14 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=15 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=16 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=17 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=18 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=19 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=20 ttl=255 time=185 ms
64 bytes from 172.31.83.134: icmp_seq=21 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=22 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=23 ttl=255 time=185 ms
64 bytes from 172.31.83.134: icmp_seq=24 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=25 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=26 ttl=255 time=184 ms
64 bytes from 172.31.83.134: icmp_seq=27 ttl=255 time=185 ms
64 bytes from 172.31.83.134: icmp_seq=28 ttl=255 time=184 ms
```

2) Purchase one domain from GoDaddy:

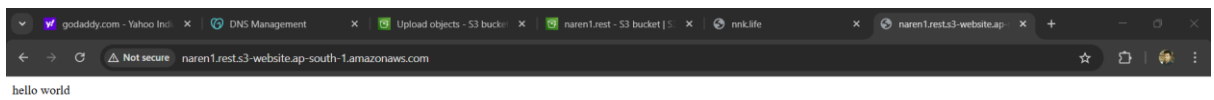
Domain:



3) Deploy static website in s3:

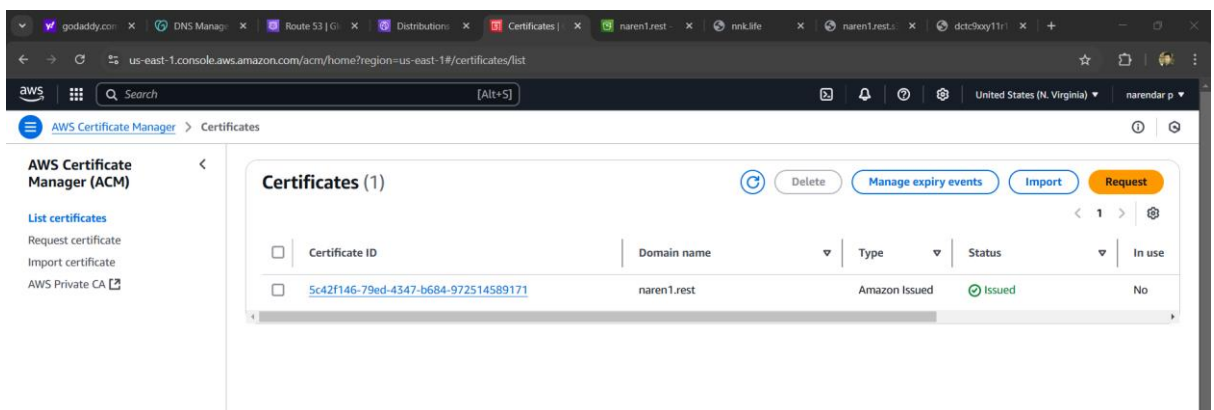


→ Access static website through URL

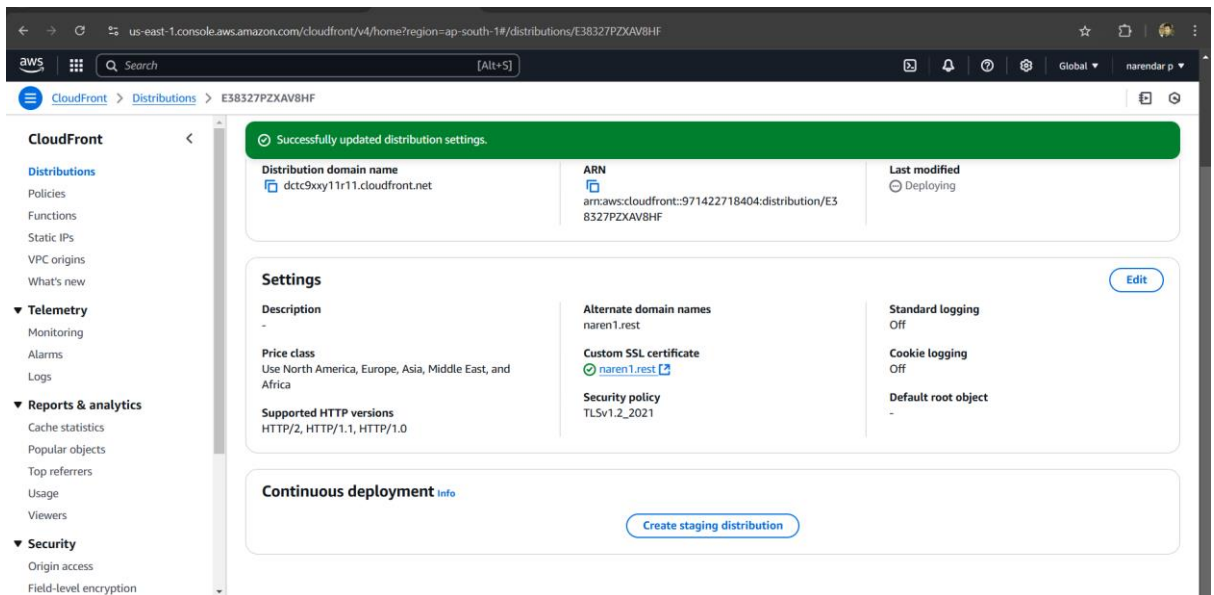


4) Create CDN and attach one SSL certificate:

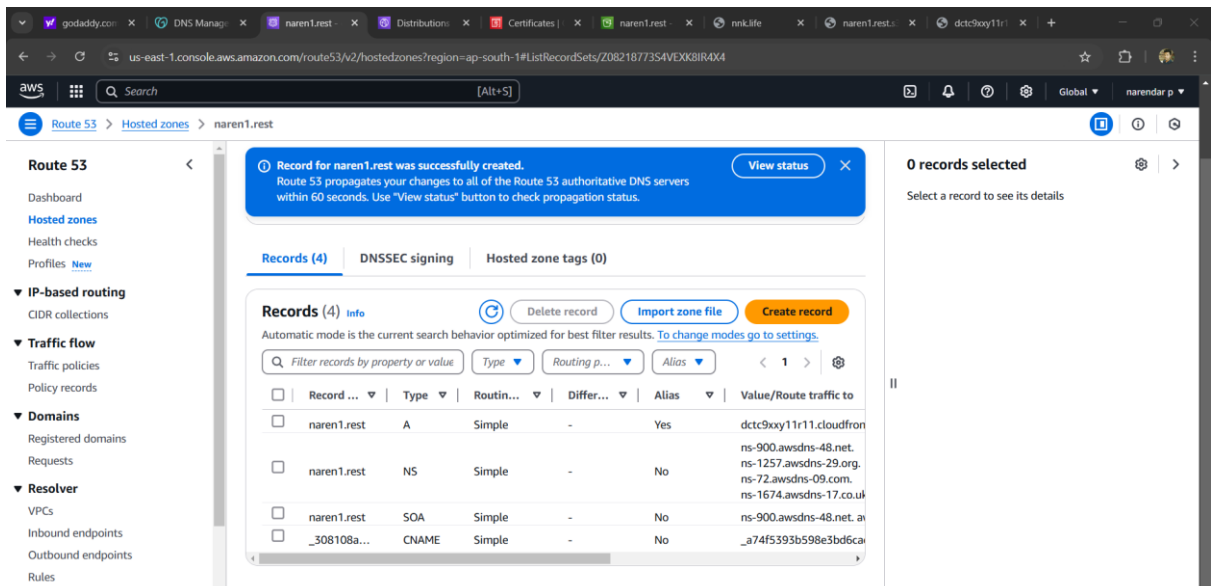
→ Requested SSL certificate and got one:



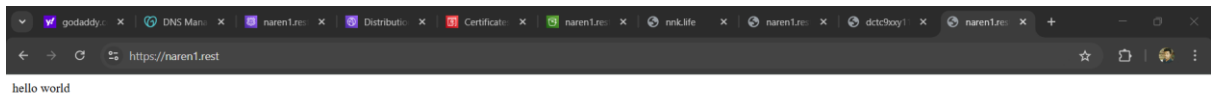
→ Attached SSL certificate to the Cloud front distribution:



5) Create Route53 hosted zone and MAP the domain with CDN:
 → Created Hosted zone and mapped domain name with CDN:

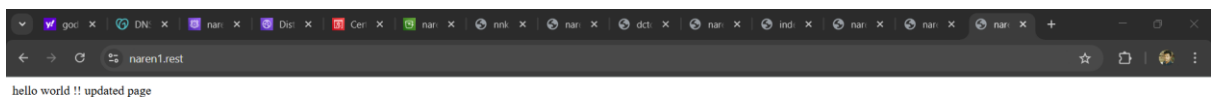


→ Accessing Static website securely:

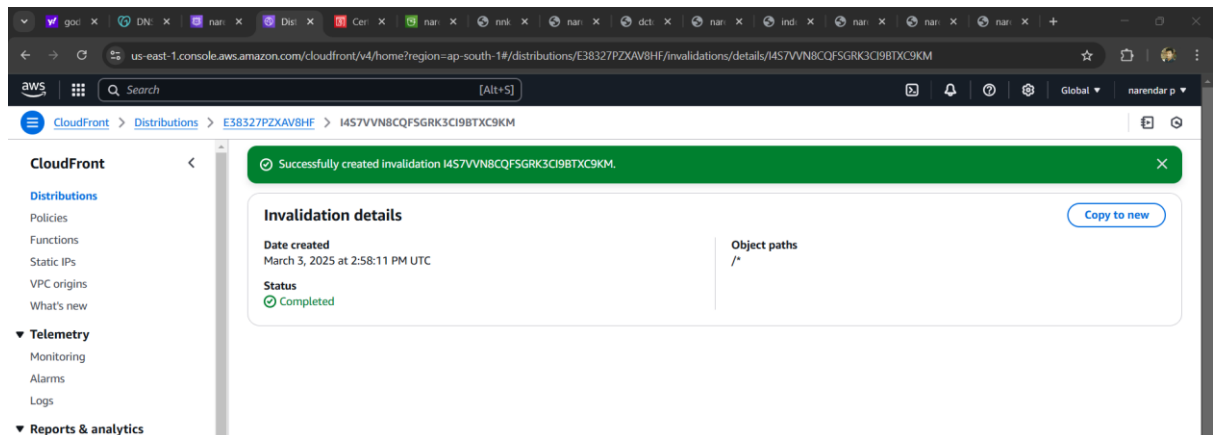


6) Update the index.html in s3 bucket and the updated file should be accessible by using domain name:

→ Updated content in static website and able to see the updated webpage by accessing Domain name:



→ after given this above one loaded:



7) Share the Domain name in slack to test the connectivity:

➔ **naren1.rest** (domain name)