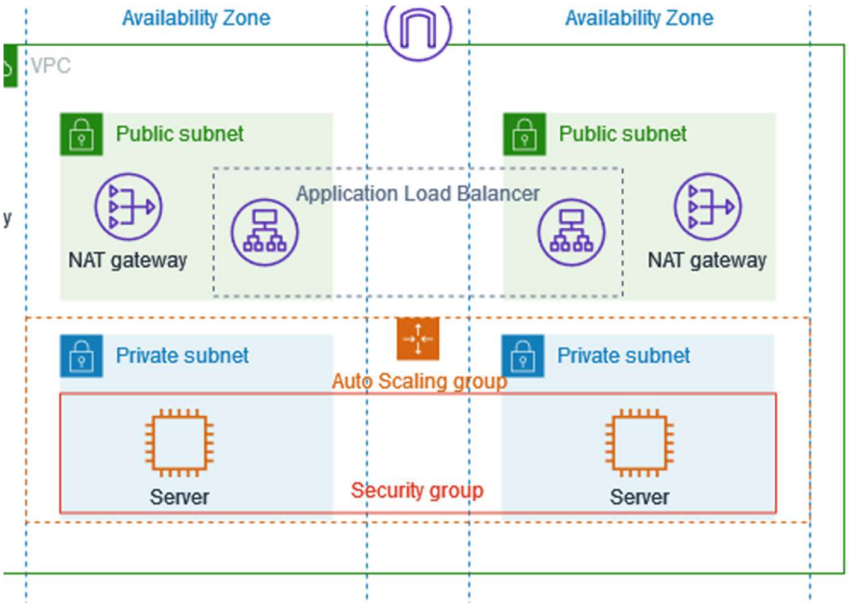
AWS PROJECT- *VPC with Public-Private Subnet.*



Main Goal:-

To securely deploy a web application where the application servers are hosted in a private subnet and can only be accessed through a public-facing Application Load Balancer (ALB) in a public subnet, ensuring the backend remains isolated from direct internet access while still being reachable via the ALB.

For example, think of the VPC as a gated community—while the ALB acts as the front gate accessible to the public, the instances (your application or website) reside securely inside. When an external user accesses the application through the ALB's public DNS, the ALB forwards the request to the instances in the private subnet and returns the response.

Thus, users do not directly access the instances; they interact with the ALB, which securely handles communication with the backend instances within the same VPC.

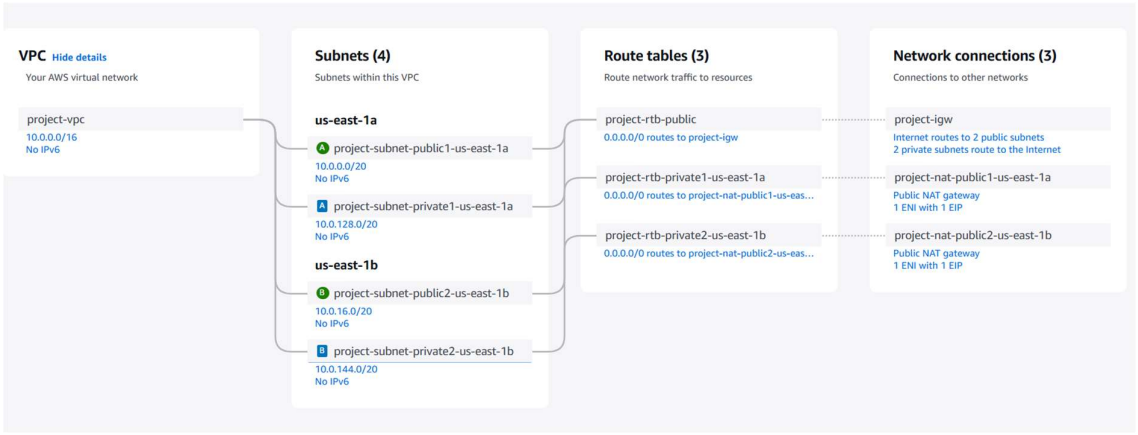
WORKING STEPS-

STEP1-

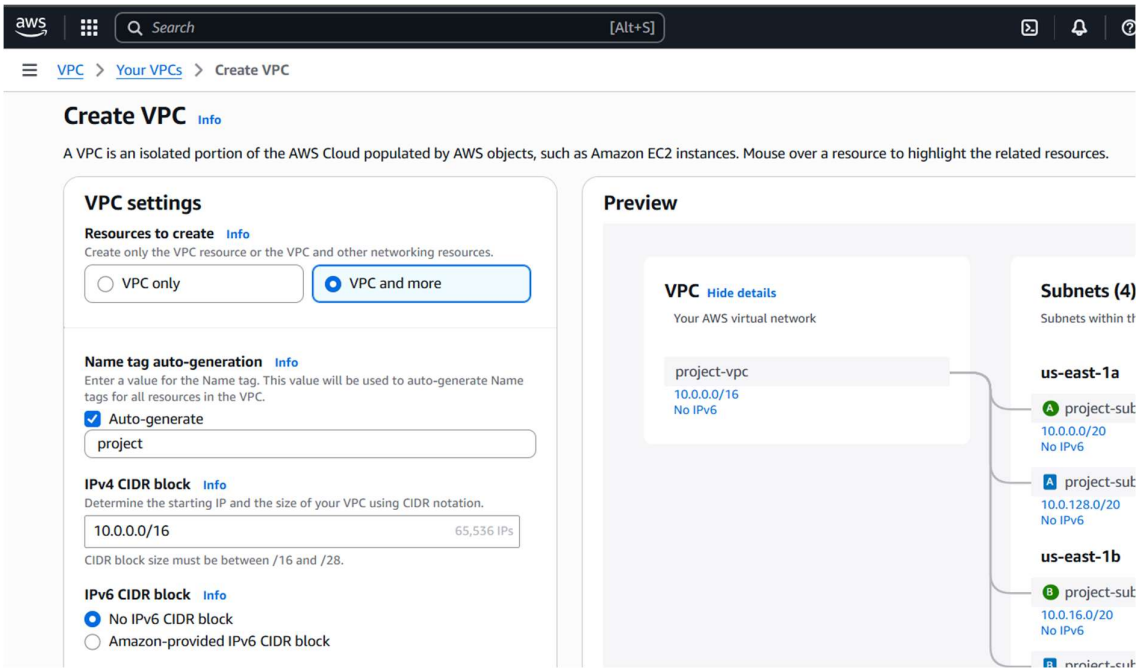
VPC and Subnet Setup

- **VPC Name**: project-vpc
- **CIDR Block**: 10.0.0.0/16
- Created **4 subnets** (2 public and 2 private) across **2 Availability Zones** (us-east-1a & us-east-1b).
    - **Public Subnets**:
        - project-subnet-public1-us-east-1a
        - project-subnet-public2-us-east-1b
    - **Private Subnets**:
        - project-subnet-private1-us-east-1a
        - project-subnet-private2-us-east-1b
- **Internet Gateway (IGW)** attached for public access.
- **NAT Gateways** created in each public subnet to allow outbound internet access from private subnets.
- **Route Tables** properly configured:
    - Public route table routes to the IGW.
    - Private route tables route to respective NAT gateways.

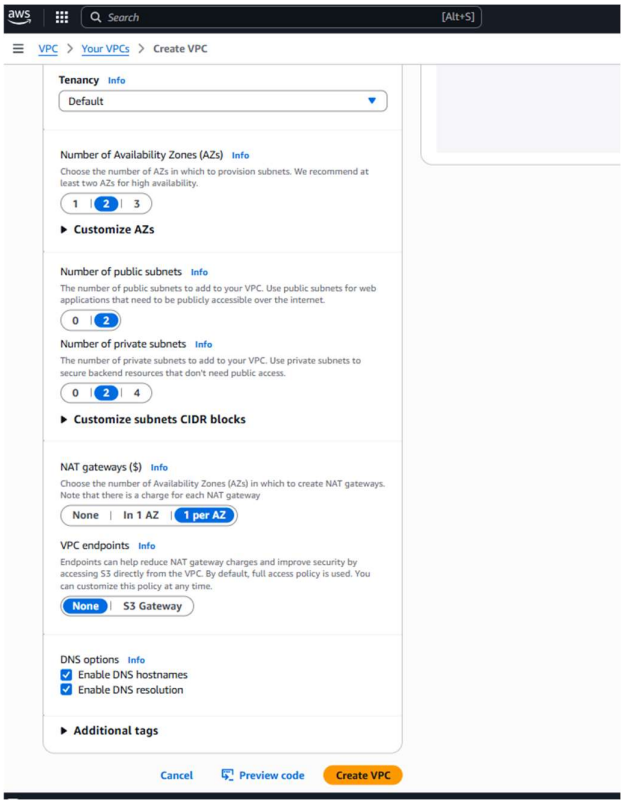
A)


VPC Hide details
Your AWS virtual network
project-vpc
10.0.0.0/16
No IPv6
Subnets (4)
Subnets within this VPC
us-east-1a
project-subnet-public1-us-east-1a
10.0.0.0/20
No IPv6
project-subnet-private1-us-east-1a
10.0.128.0/20
No IPv6
us-east-1b
project-subnet-public2-us-east-1b
10.0.16.0/20
No IPv6
project-subnet-private2-us-east-1b
10.0.144.0/20
No IPv6
Route tables (3)
Route network traffic to resources
project-rtb-public
0.0.0.0/0 routes to project-igw
project-rtb-private1-us-east-1a
0.0.0.0/0 routes to project-nat-public1-us-eas...
project-rtb-private2-us-east-1b
0.0.0.0/0 routes to project-nat-public2-us-eas...
Network connections (3)
Connections to other networks
project-igw
Internet routes to 2 public subnets
2 private subnets route to the Internet
project-nat-public1-us-east-1a
Public NAT gateway
1 ENI with 1 EIP
project-nat-public2-us-east-1b
Public NAT gateway
1 ENI with 1 EIP

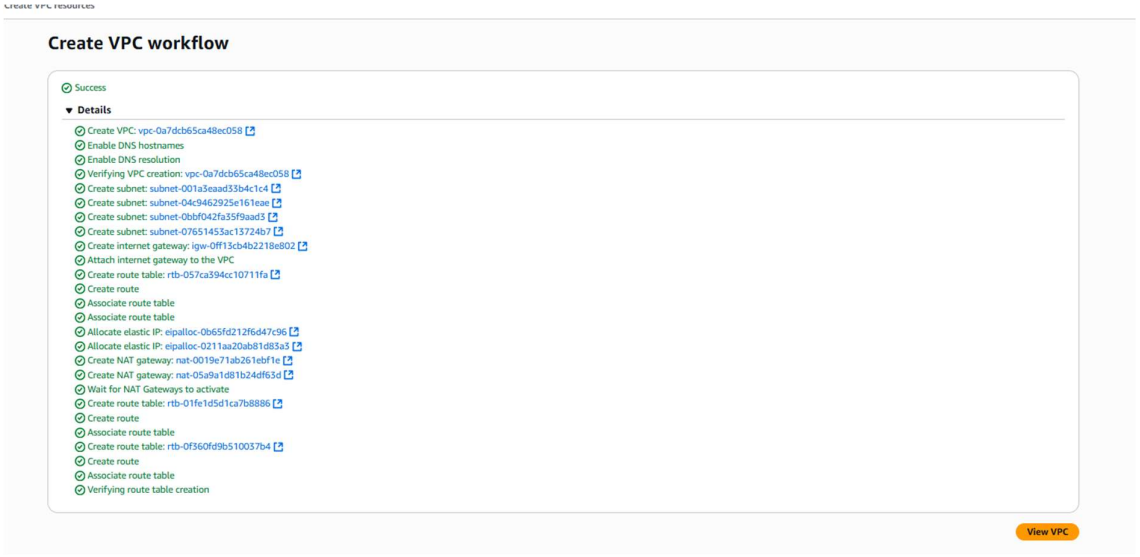
B)


aws
Search
[Alt+S]
VPC > Your VPCs > Create VPC
Create VPC Info
A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances. Mouse over a resource to highlight the related resources.
VPC settings
Resources to create Info
Create only the VPC resource or the VPC and other networking resources.
VPC only
VPC and more
Name tag auto-generation Info
Enter a value for the Name tag. This value will be used to auto-generate Name tags for all resources in the VPC.
Auto-generate
project
IPv4 CIDR block Info
Determine the starting IP and the size of your VPC using CIDR notation.
10.0.0.0/16
65,536 IPs
CIDR block size must be between /16 and /28.
IPv6 CIDR block Info
No IPv6 CIDR block
Amazon-provided IPv6 CIDR block
Preview
VPC Hide details
Your AWS virtual network
project-vpc
10.0.0.0/16
No IPv6
Subnets (4)
us-east-1a
10.0.0.0/20
No IPv6
10.0.128.0/20
No IPv6
us-east-1b
10.0.16.0/20
No IPv6

C)


aws
Search
[Alt+S]
VPC > Your VPCs > Create VPC
Tenancy Info
Default
Number of Availability Zones (AZs) Info
Choose the number of AZs in which to provision subnets. We recommend at least two AZs for high availability.
1 | 2 | 3
Customize AZs
Number of public subnets Info
The number of public subnets to add to your VPC. Use public subnets for web applications that need to be publicly accessible over the internet.
0 | 2
Number of private subnets Info
The number of private subnets to add to your VPC. Use private subnets to secure backend resources that don't need public access.
0 | 2 | 4
Customize subnets CIDR blocks
NAT gateways ($) Info
Choose the number of Availability Zones (AZs) in which to create NAT gateways. Note that there is a charge for each NAT gateway
None | In 1 AZ | 1 per AZ
VPC endpoints Info
Endpoints can help reduce NAT gateway charges and improve security by accessing S3 directly from the VPC. By default, full access policy is used. You can customize this policy at any time.
None | S3 Gateway
DNS options Info
Enable DNS hostnames
Enable DNS resolution
Additional tags
Cancel
Preview code
Create VPC

Create VPC


Create VPC workflow
Success
Details
Create VPC: vpc-0a7dcb65ca48ec058
Enable DNS hostnames
Enable DNS resolution
Verifying VPC creation: vpc-0a7dcb65ca48ec058
Create subnet: subnet-001a3eaad33b4c1c4
Create subnet: subnet-04c9462925e161eae
Create subnet: subnet-0bbf042fa35f9aad3
Create subnet: subnet-07651453ac13724b7
Create internet gateway: igw-0ff13cb4b2218e802
Attach internet gateway to the VPC
Create route table: rtb-057ca394cc10711fa
Create route
Associate route table
Associate route table
Allocate elastic IP: eipalloc-0b65fd212f6d47c96
Allocate elastic IP: eipalloc-0211aa20ab81d83a3
Create NAT gateway: nat-0019e71ab261ebf1e
Create NAT gateway: nat-05a9a1d81b24df63d
Wait for NAT Gateways to activate
Create route table: rtb-01fe1d5d1ca7b8886
Create route
Associate route table
Create route table: rtb-0f360fd9b5100037b4
Create route
Associate route table
Verifying route table creation
View VPC

STEPS AFTER CREATION OF VPC

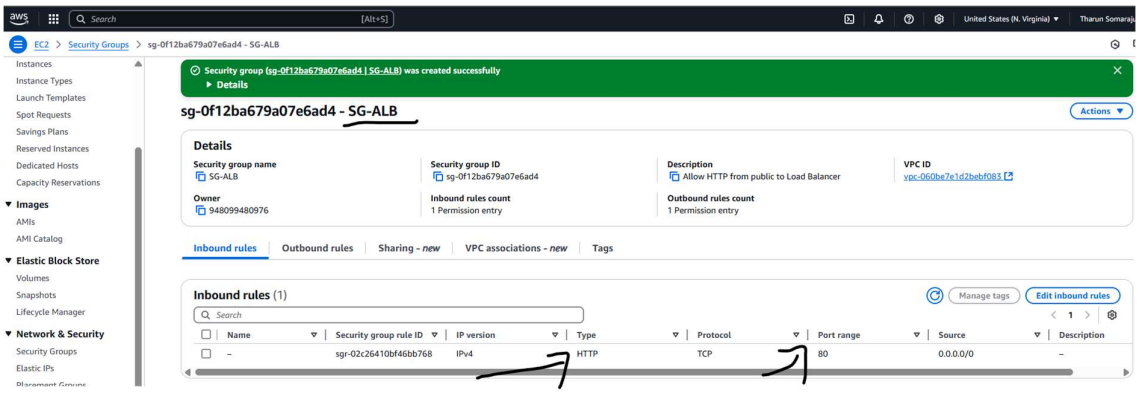
A)Create Security Groups

Set up Security Groups **before** creating the Bastion Host, Load Balancer, and Private EC2s.

🔧 **Step-by-Step Instructions**

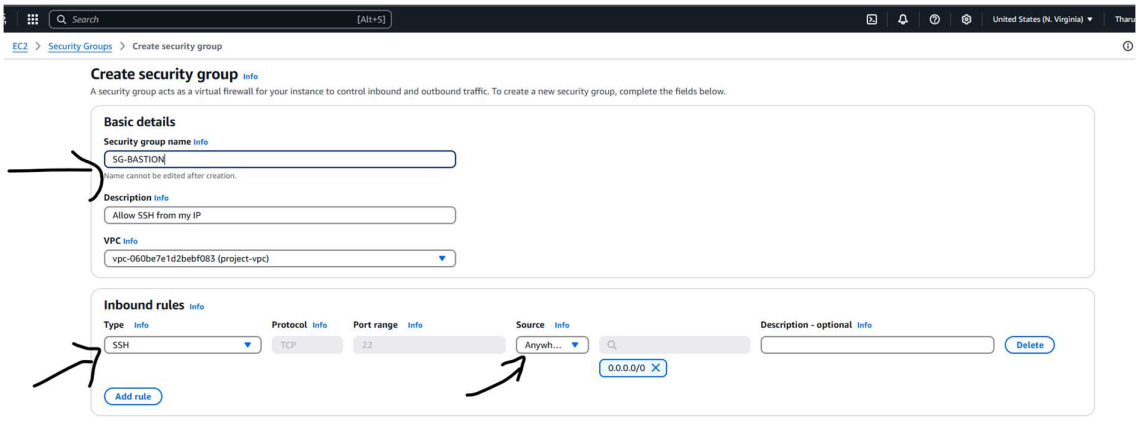
🧱 **Step 1: Create Security Group for Load Balancer (SG-ALB)**

1. Go to **EC2 Dashboard** → **Security Groups** → **Create security group**
2. Name: SG-ALB
3. Description: Allow HTTP from public to Load Balancer
4. VPC: Select your VPC (e.g., project-vpc)
5. Inbound Rule:
    - Type: HTTP
    - Port: 80
    - Source: Anywhere (0.0.0.0/0)
6. Outbound Rule: leave default (allow all)
7. Click **Create security group**



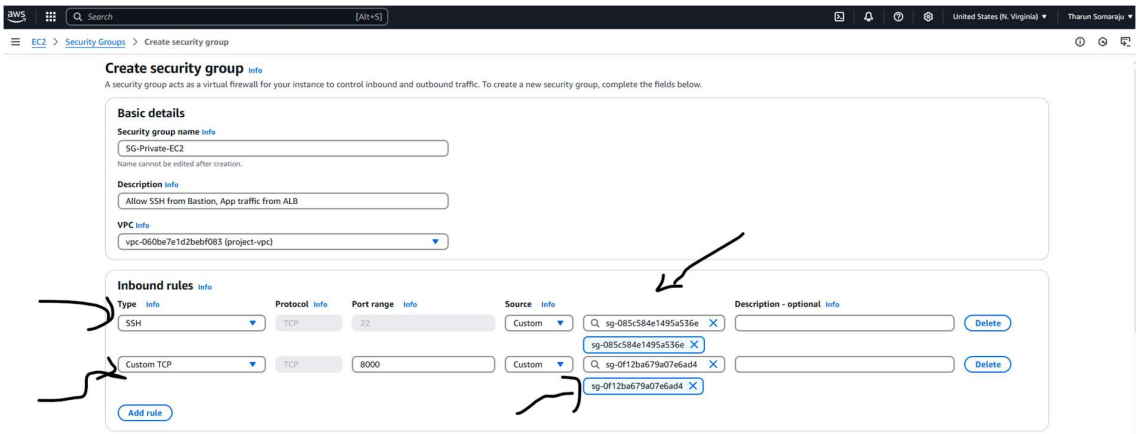
Step 2: Create Security Group for Bastion Host (SG-Bastion)

1. Go to **Security Groups → Create security group**
2. Name: SG-Bastion
3. Description: Allow SSH from my IP
4. VPC: Select your VPC
5. Inbound Rule:
    - Type: SSH
    - Port: 22
    - Source: My IP (or Anywhere if we are doing ssh from different Devices)
6. Outbound Rule: leave default
7. Click **Create security group**



🛠️ **Step 3: Create Security Group for Private EC2s (SG-Private-EC2)**

1. Go to **Security Groups** → **Create security group**
2. Name: SG-Private-EC2
3. Description: Allow SSH from Bastion, App traffic from ALB
4. VPC: Same VPC as above
5. Inbound Rules:
    - Rule 1:
        - Type: SSH
        - Port: 22
        - Source: **Custom** → choose **SG-Bastion**
    - Rule 2:
        - Type: Custom TCP
        - Port: 8000
        - Source: **Custom** → choose **SG-ALB**
6. Outbound Rule: leave default
7. Click **Create security group**



THE MAIN IMPORTANT POINTS TO BE NOTED

-So only the Bastion Host instance (that has SG-Bastion) can access your Private EC2, not the whole world.

-So only the Load Balancer (with SG-ALB) can send traffic to the private EC2 on port 8000, even though the Load Balancer is open to the public.

✅ **Done! You Now Have:**

SG Name	Use	Inbound Rules
SG-ALB	Load Balancer	HTTP (80) from Anywhere
SG-Bastion	Bastion Host	SSH (22) from your IP or Anywhere
SG-Private-EC2	Private EC2/ASG	SSH (22) from SG-Bastion TCP 8000 from SG-ALB

🎯 **Final Step:**

When you launch:

- **Load Balancer** → Attach SG-ALB
- **Bastion Host** → Attach SG-Bastion
- **Private EC2s or Auto Scaling Group** → Attach SG-Private-EC2

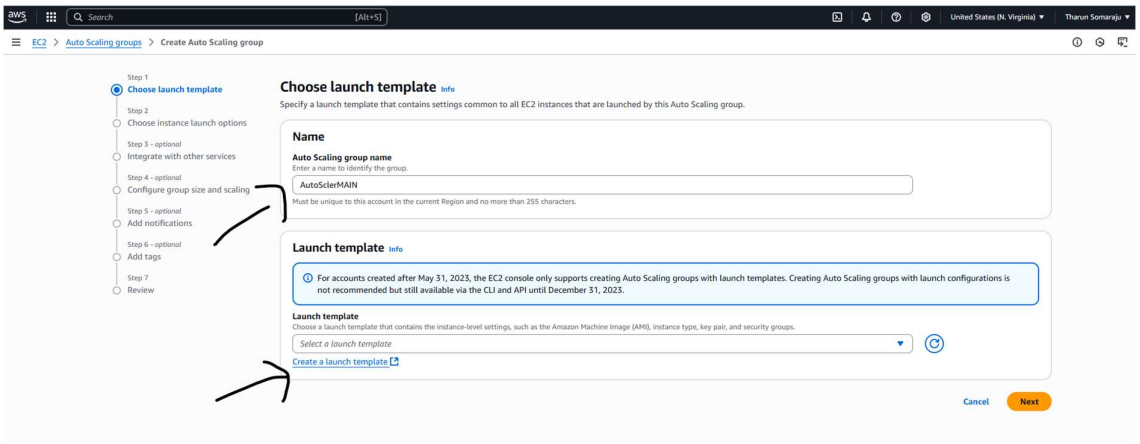
Now your architecture is secure and properly set up without using **"Anywhere" for private EC2s**. ✅

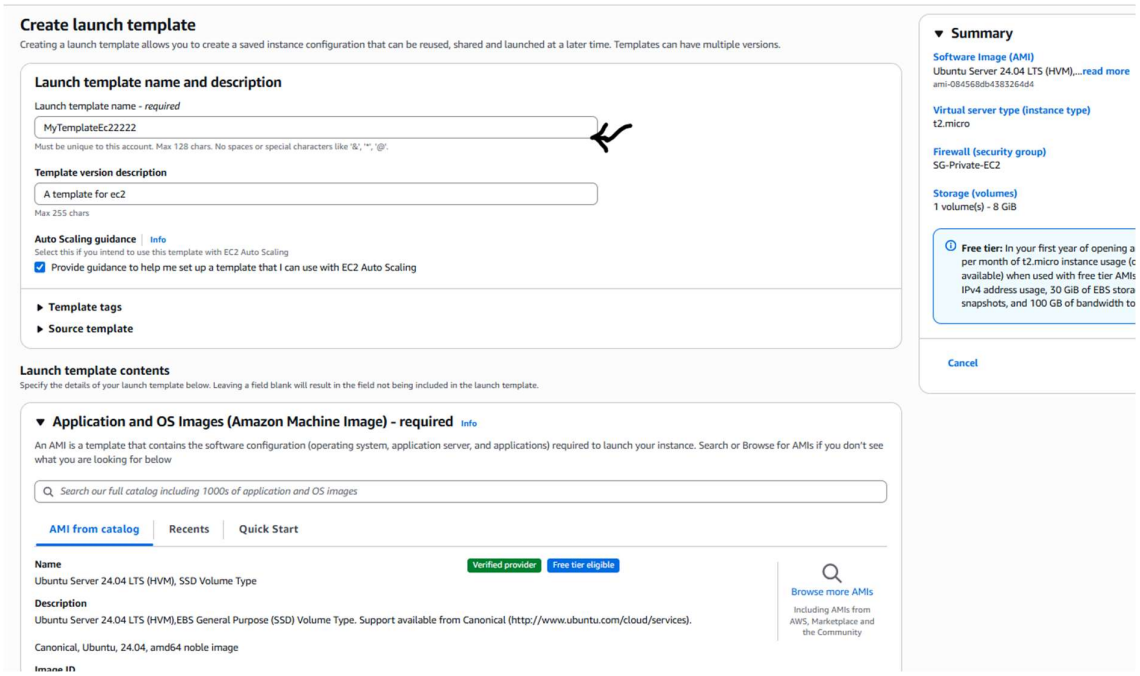
B)Add Auto Scaling Ec2 in Private Subnet

Go to **Ec2 Auto Scaling** , to create instances in the Private Subnet.

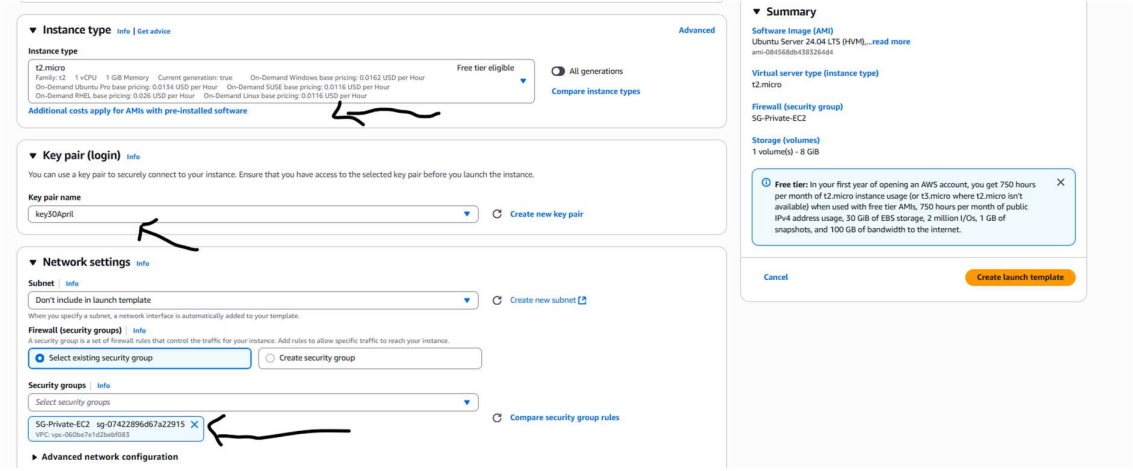
-**Create a Launch Template** for Ec2's , so we don't need to create this template again and again.

So when we want Auto Scale, again, we simply use this Template.



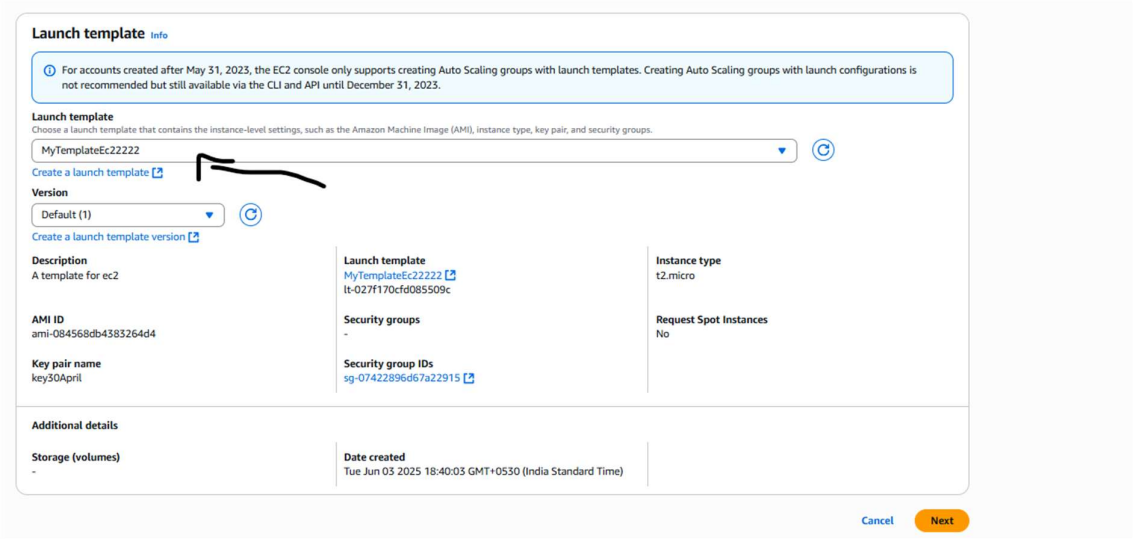


Now SELECT THE **Key-Pair**, **Security Group as (SG-Private-Ec2)**

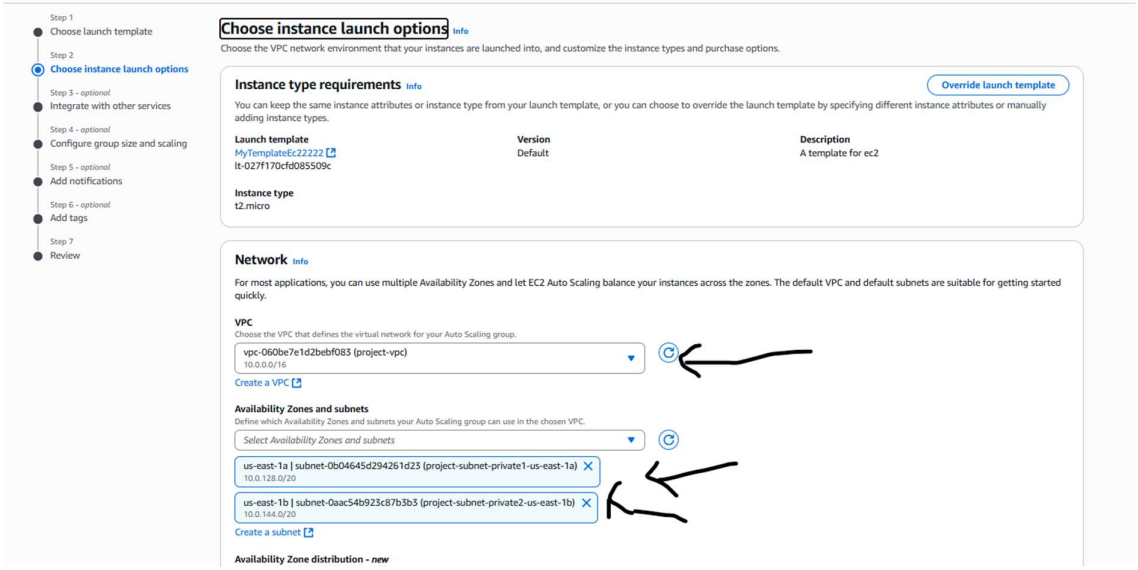


CREATE LAUNCH TEMPLATE

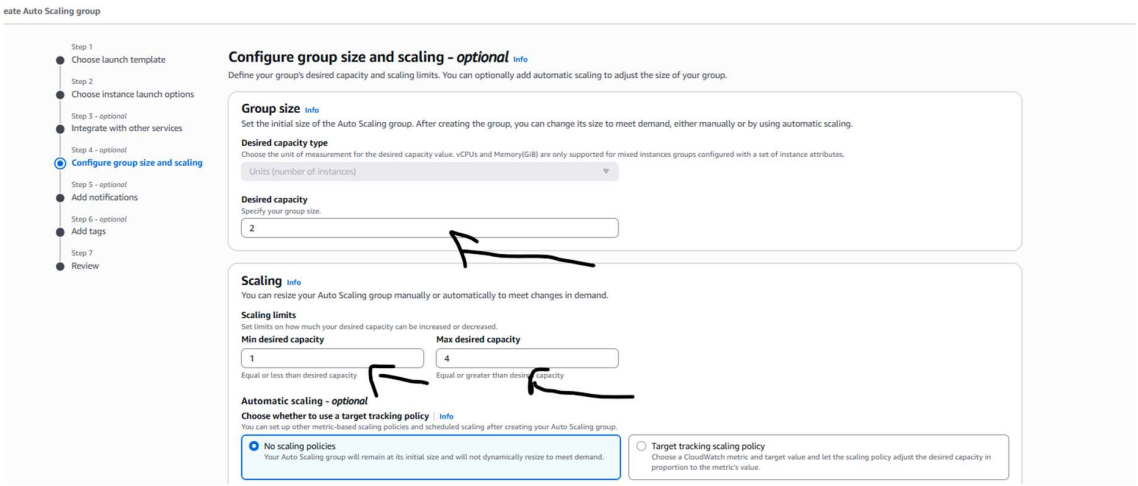
ADD THE LAUNCH TEMPLATE TO AUTOSCALING



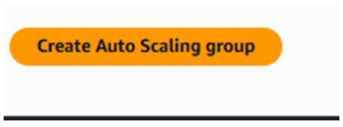
ADD THE AutoScaler to VPC, i.e in the **Private Subnets ONLY**



Specify Auto Scaling Scale Size



NOW PRESS **CREATE AUTO SCALING GROUP**

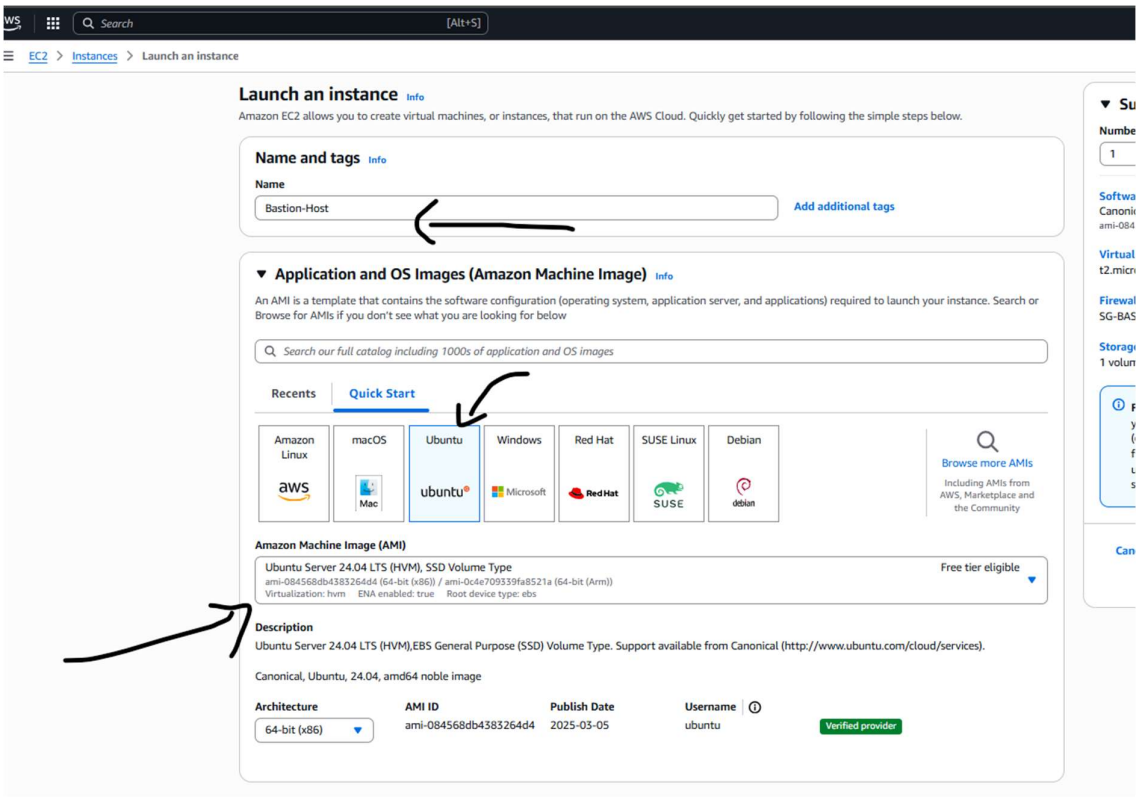


C)

Create a Bastion Host in Public Subnet(for contacting private ec2 via SSH)

-

Launch a Bastion-Host Ec2 Instance



NOW

Specify the Required terms such as

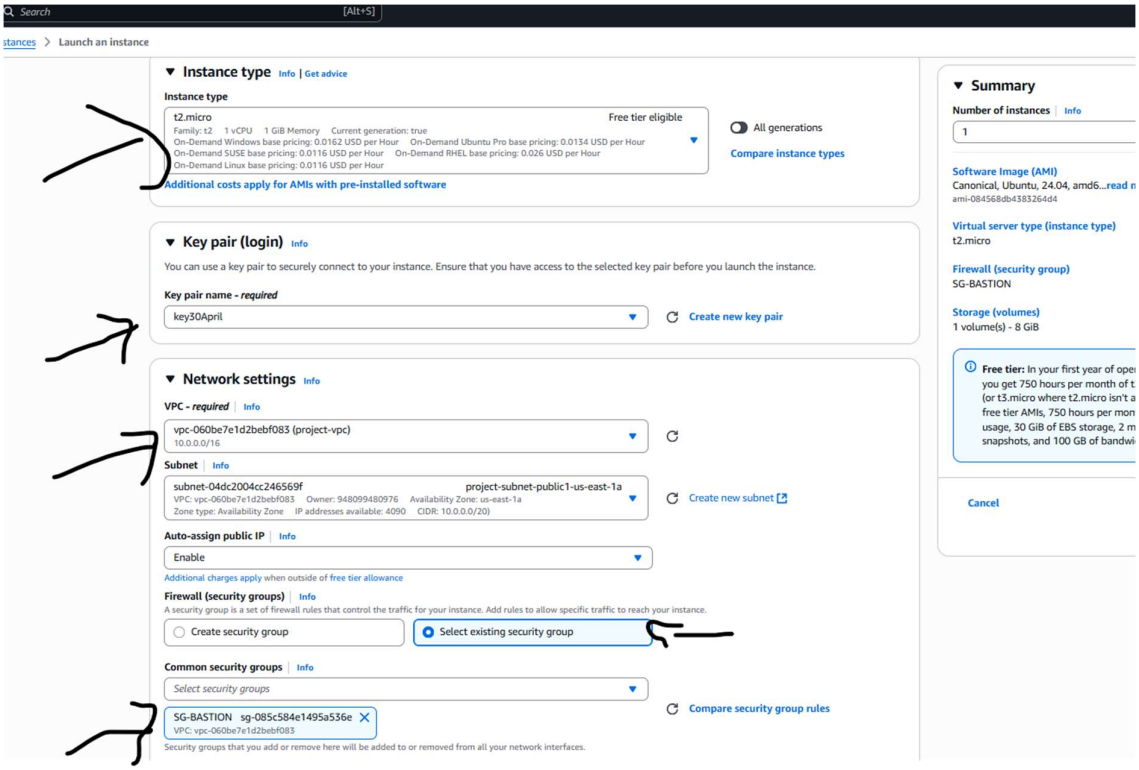
Key-pair,

VPC,

and the Subnet i.e **PUBLIC SUBNET of VPC**

AND Auto Assign IP- **Enable**

Add the Security Group that we created for Bastion (SG-Bastion)



PRESS LAUCH INSTANCE

D)HOST THE WEBSITE IN A PRIVATE INSTANCE

- We first make contact with Bastion, and from Bastion we contact Private Instance to Deploy a HTML Code and Python server for Hosting 8000

- FOR copying key **key30April.pem** from the Local Pc to Bastion (as we want to login to private subnet from bastion)

scp -i "C:\Users\Somaraju Tharun\Downloads\key30April.pem" "C:\Users\Somaraju Tharun\Downloads\key30April.pem" ubuntu@54.87.183.33:/home/ubuntu/

```
Microsoft Windows [Version 10.0.26100.4202]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Somaraju Tharun>scp -i "C:\Users\Somaraju Tharun\Downloads\key30April.pem" "C:\Users\Somaraju Tharun\Downloads\
key30April.pem" ubuntu@54.87.183.33:/home/ubuntu/
The authenticity of host '54.87.183.33 (54.87.183.33)' can't be established.
ED25519 key fingerprint is SHA256:It2A6/paGhOAwdJ2lDAtpw3L/njBoOkn/zM0Yis8j3I.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
Warning: Permanently added '54.87.183.33' (ED25519) to the list of known hosts.
key30April.pem                                                100% 1678     5.7KB/s   00:00

C:\Users\Somaraju Tharun>
```

-

To verify if the key got copied or not

-ssh -i "C:\Users\Somaraju Tharun\Downloads\key30April.pem" ubuntu@54.87.183.33

-$ ls

You will see the key

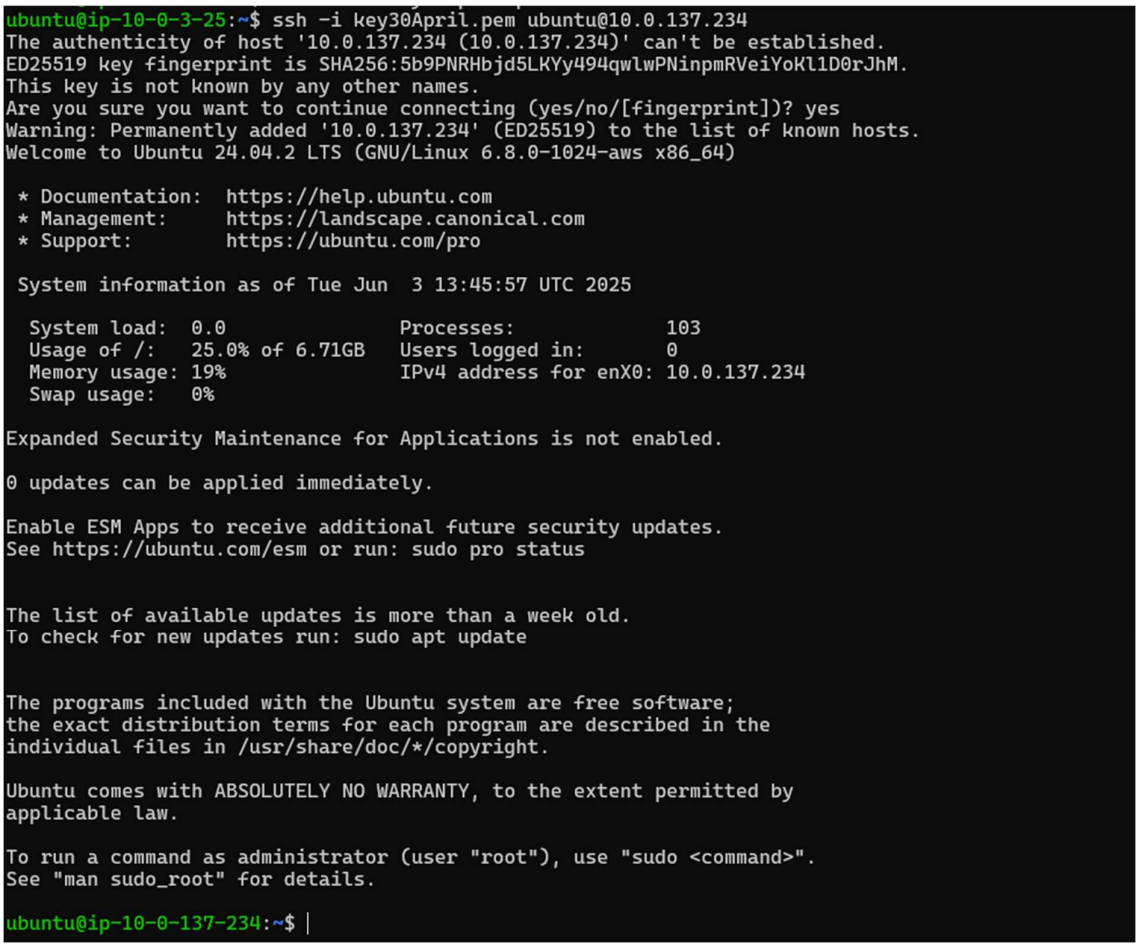
```
C:\Users\Somaraju Tharun>ssh -i "C:\Users\Somaraju Tharun\Downloads\key30April.pem" ubuntu@54.87.183.33
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-1024-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Tue Jun  3 13:42:28 UTC 2025

  System load:  0.0               Processes:             104
  Usage of /:   25.3% of 6.71GB   Users logged in:       0
  Memory usage: 21%               IPv4 address for enX0: 10.0.3.25
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-10-0-3-25:~$ ls
key30April.pem
ubuntu@ip-10-0-3-25:~$
```

-chmod 400 is used to secure the .pem file by allowing only the owner to read it, which SSH requires for safe access.

$chmod 400 key30April.pem

To SSH from the **Bastion Host** into the **private EC2 instance (10.0.137.234)**, use the following command **on the Bastion terminal**:

$ssh -i key30April.pem ubuntu@10.0.137.234



-

1) CREATE A HTML PAGE IN THE EC2 TERMINAL

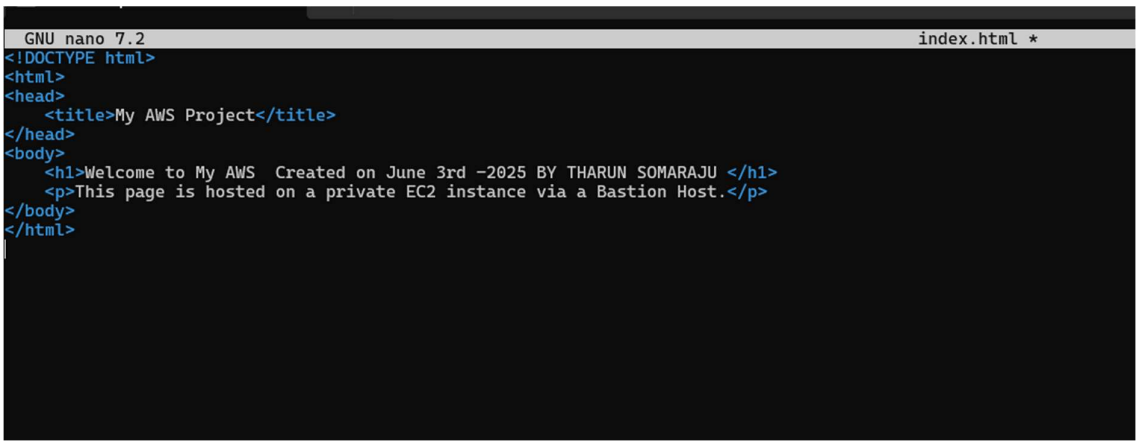
Create the file
Run this on your EC2 instance:

$nano index.html

```
ubuntu@ip-10-0-137-234:~$ nano index.html
```

Paste the following code:

```
<!DOCTYPE html>
<html>
<head>
   <title>My AWS Project</title>
</head>
<body>
   <h1>Welcome to My AWS  Created on June 3rd -2025 BY THARUN SOMARAJU </h1>
   <p>This page is hosted on a private EC2 instance via a Bastion Host.</p>
</body>
</html>
```



Save and exit:

- Press **Ctrl + O** → **Enter** to save
- Then **Ctrl + X** to exit Nano

✅ Your HTML page is ready as index.html.

Create a Python server

$ python3 -m http.server 8000

🌐 **What happens when user visits Load Balancer?**

1. The request goes to the Load Balancer.
2. Load Balancer sends the request to your private EC2.
3. The Python server running on that EC2 responds with your HTML page.
4. User sees your website!

IMPORTANT POINT TO REMEMBER ABOUT-

How does an EC2 instance recognize the key-pair?

When we
SSH FROM Local PC -→ Bastion-Host(Public Subnet)

Bastion-Host(Public Subnet) --→ Ec2 (Private Subnet)

Here we are copying KeyPair.pem from local pc to bastion , as we want this KeyPair.pem to be verified with private ec2 when we are doing SSH from bastion to Private ec2.

BELOW IS THE MAIN AND IMPORTANT DESCRIPTION about the ABOVE:-

When you **launch an EC2 instance**, you **attach a key-pair** (like key30April.pem) to it. Here's what happens:

✅ **Behind the Scenes:**

1. When the EC2 is created with a key-pair:
    - AWS puts the **public part** of the key into a file called:

/home/ubuntu/.ssh/authorized_keys

2. When you try to connect using:

ssh -i key30April.pem ubuntu@<ec2-ip>

    - Your **private key** (key30April.pem) is used by your SSH client.
    - The EC2 instance checks:
        - "Does this private key match the public key I have in authorized_keys?"
3. ✅ If it matches:
    - You are allowed to login.
4. ❌ If it doesn't match:
    - Access is denied.

🔑 **Simple Analogy:**

Think of it like a **lock and key**:

- EC2 has the **lock** (public key).
- You have the **key** (private key .pem).
- Only the **right key** can unlock it.

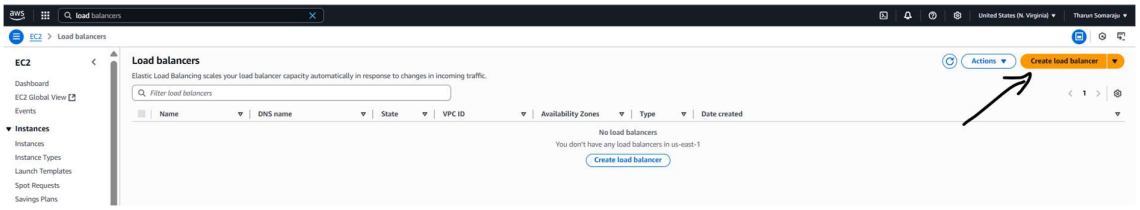
📌 **So to answer your question:**

How does EC2 recognize the key-pair?

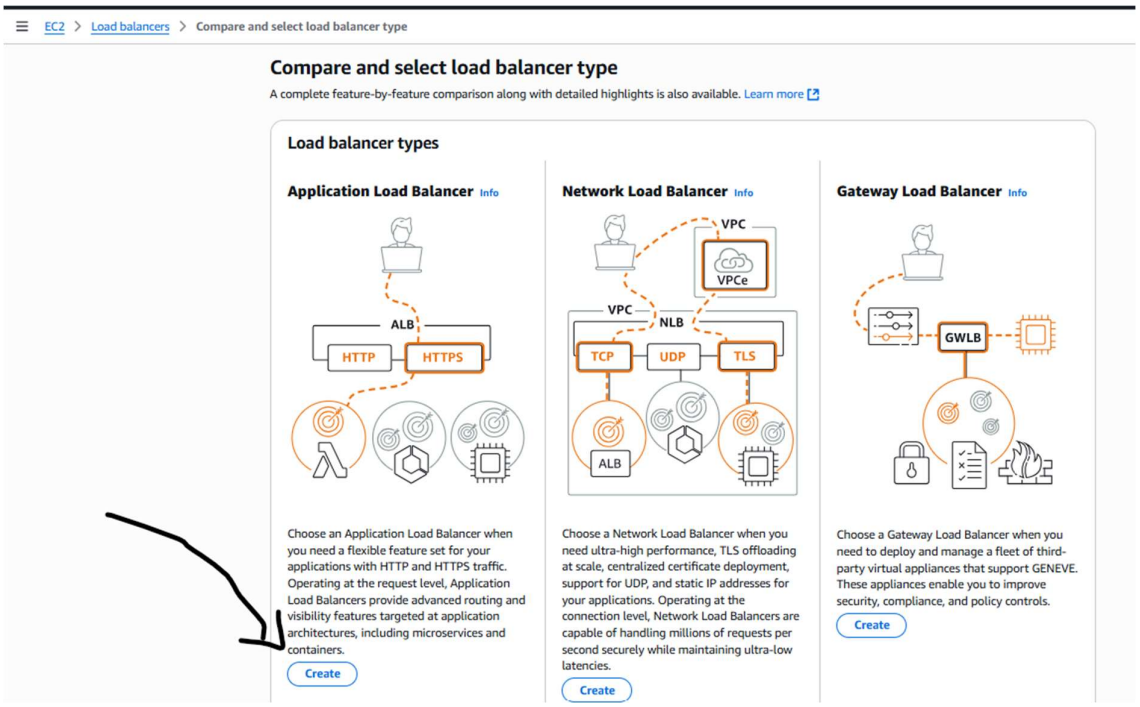
Because **when the instance is launched**, AWS **automatically saves the public part of the key-pair** into the EC2's login system. When you try to connect using the private key, it matches it with the stored public key.

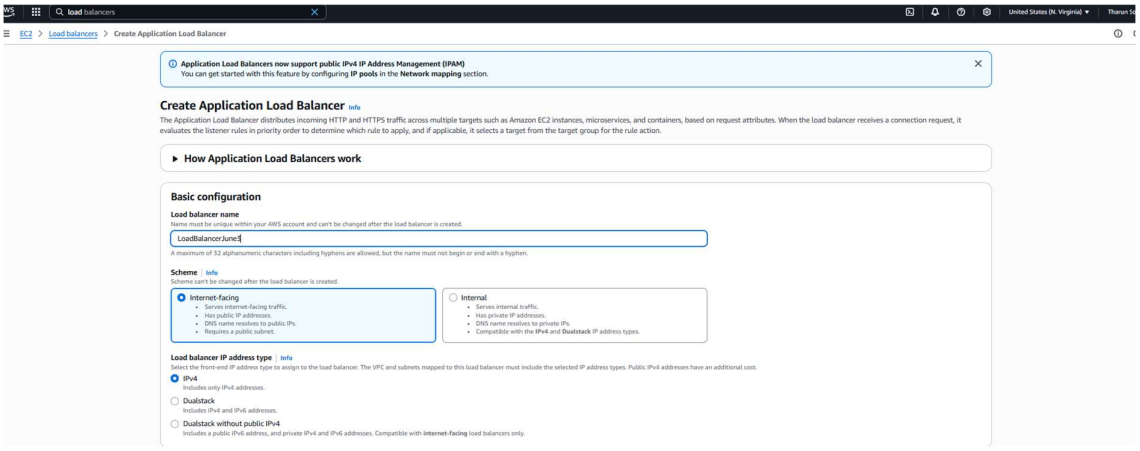
E)CREATE THE LOAD BALANCER in the Public Subnet And its Targets (Private instances)

- Create the Load balancer in the Public Subnet because , A user Can get Only LB URL to access the website present in the Private Ec2 Instance..

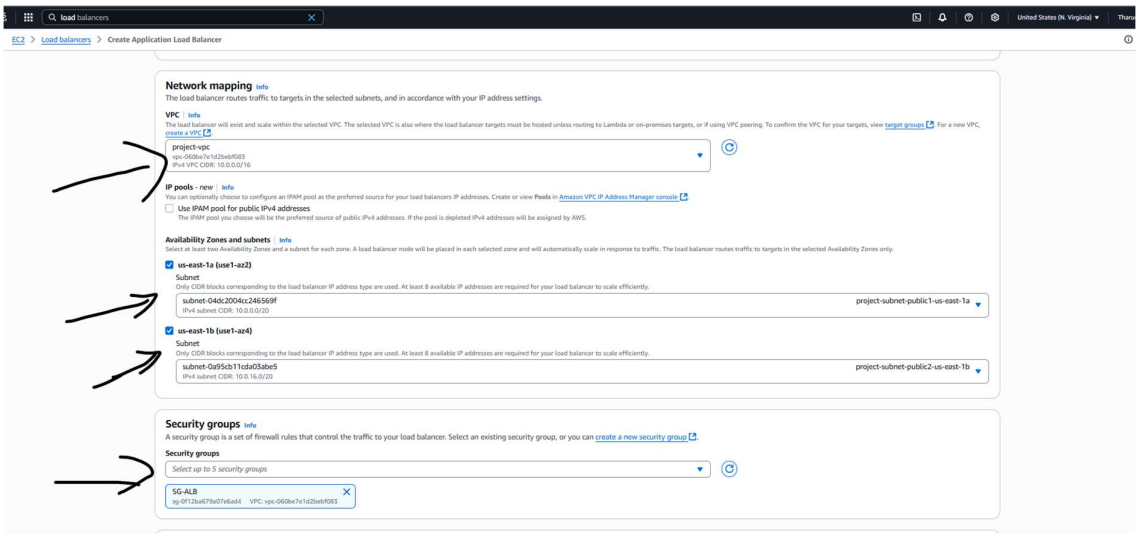


Select Application Load balancer

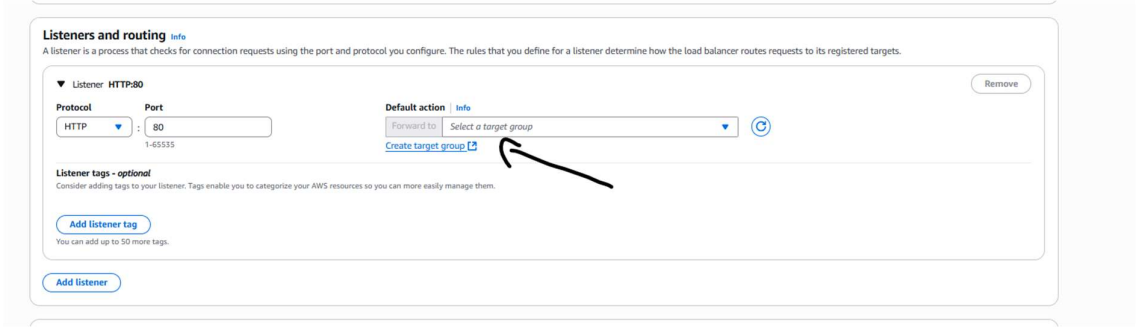


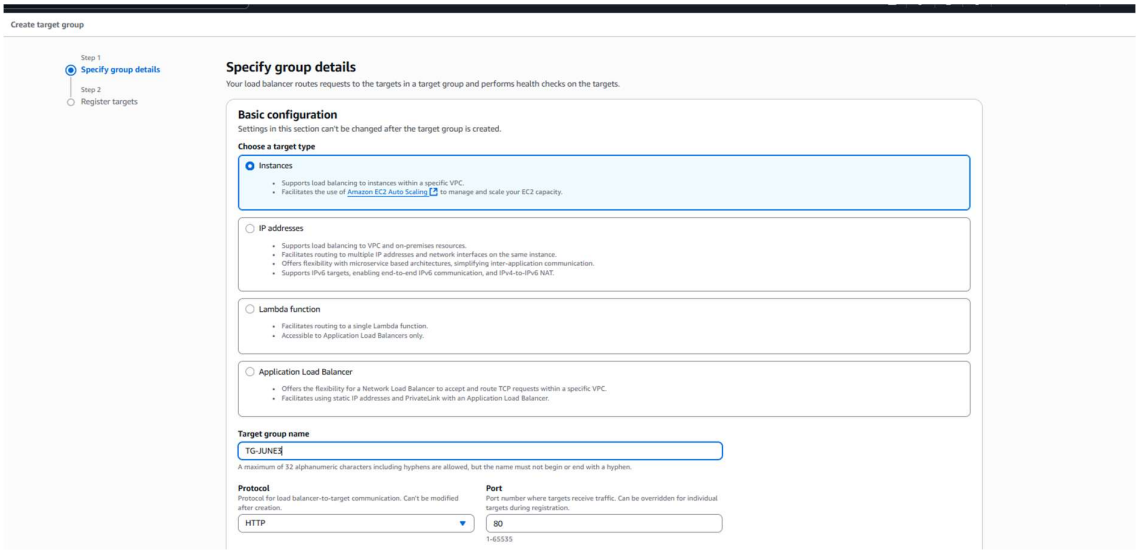


Select the VPC we created , Select the AZ's i.e Public Subnets, and Add the **Security Group(SG-ALB)** the we created At initial Working

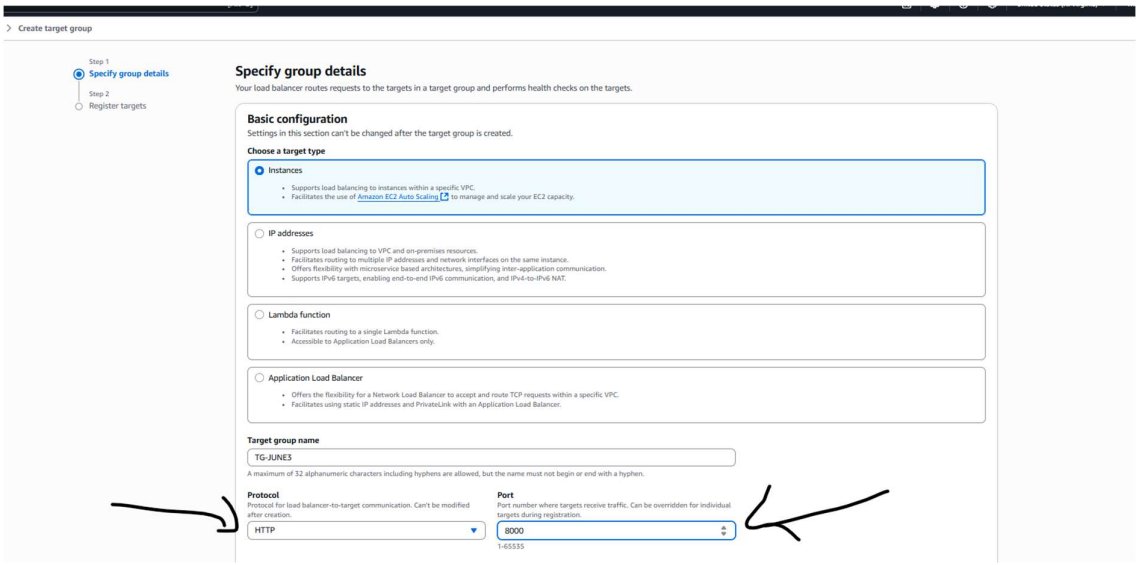


Create a Target Group that Load balancer has to pass the Traffic to..

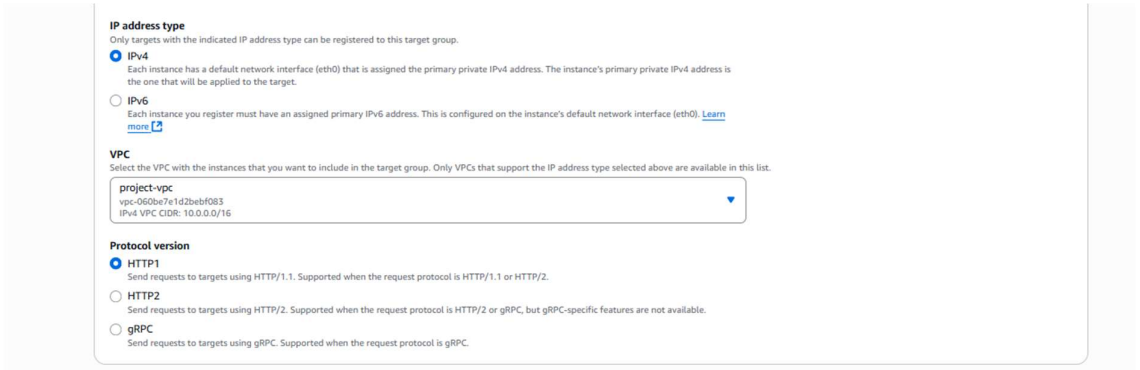




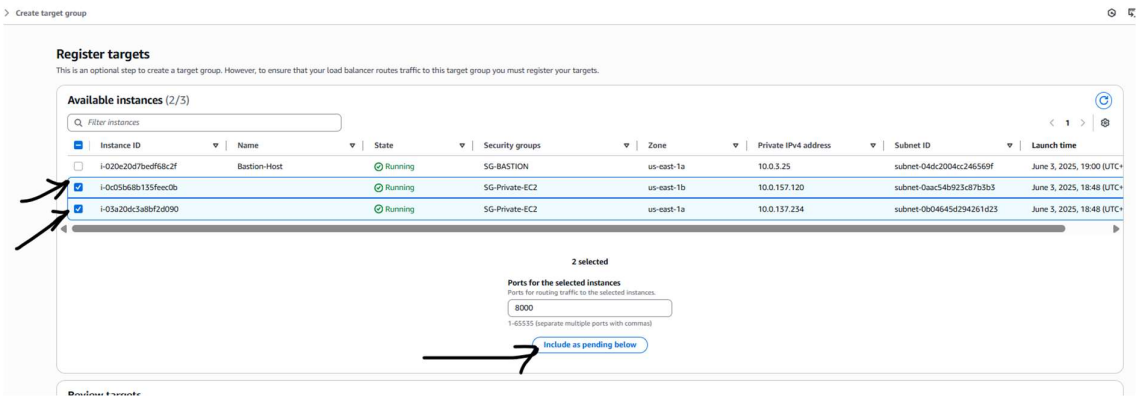
Keep the Port Number as 8000, as the Instance allowed the Inbound as 8000(SG-ALB)

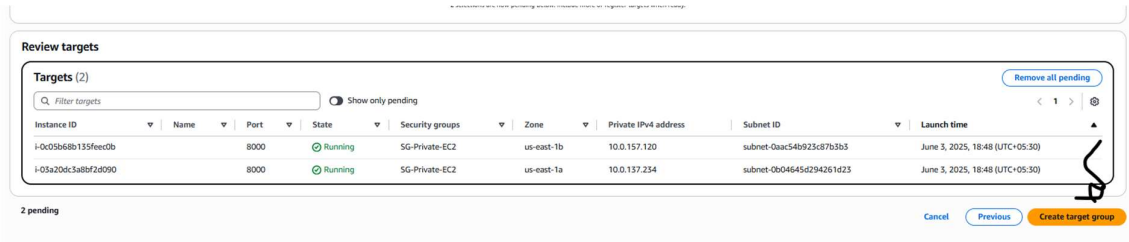


Specify the VPC

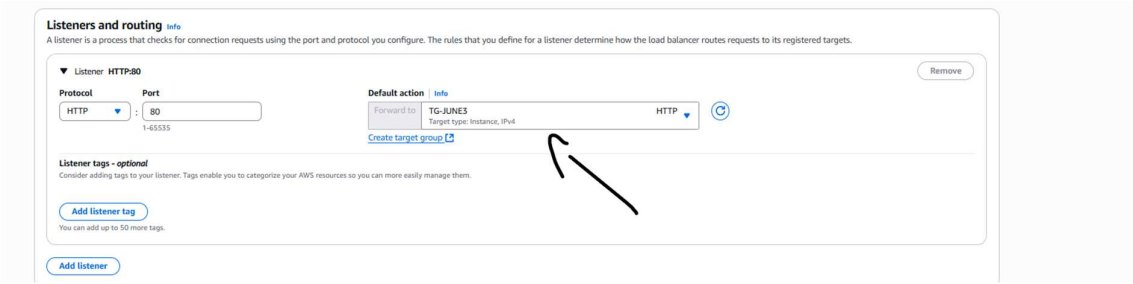


Add the Targets



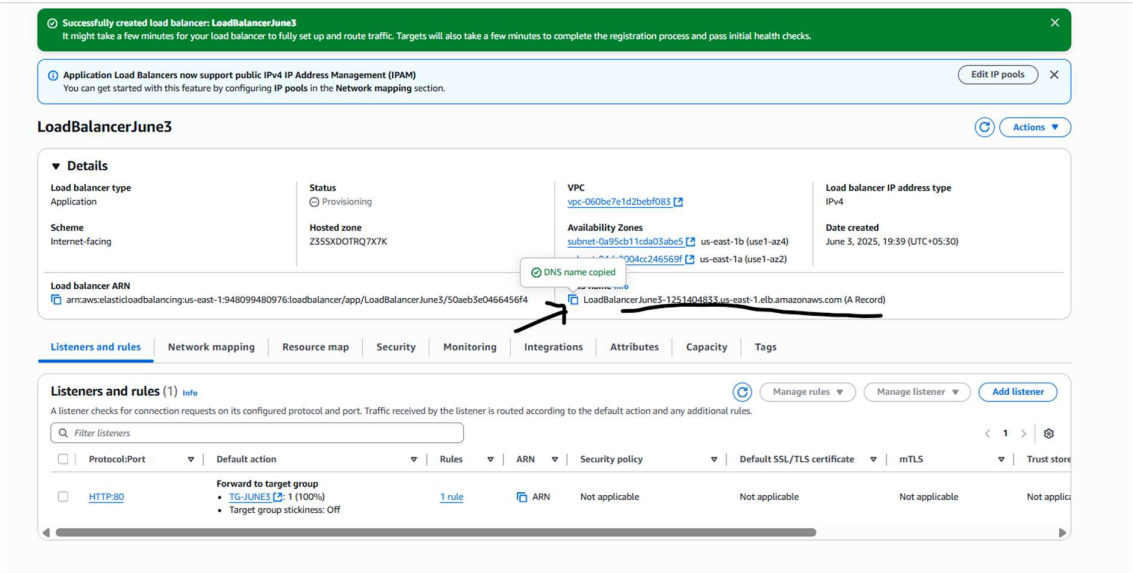


Add the Target Group to the Load Balancer



PRESS CREATE LOAD-BALANCER BUTTON

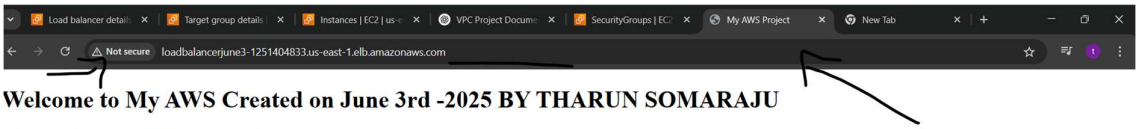
Copy the URL of Load Balancer AND HAPPILY ACCESS THE WEBSITE THAT IS PRESENT IN THE PRIVATE SUBNET Ec2 Instance..



Now We can see the Website that we Hosted on Private Subnet..

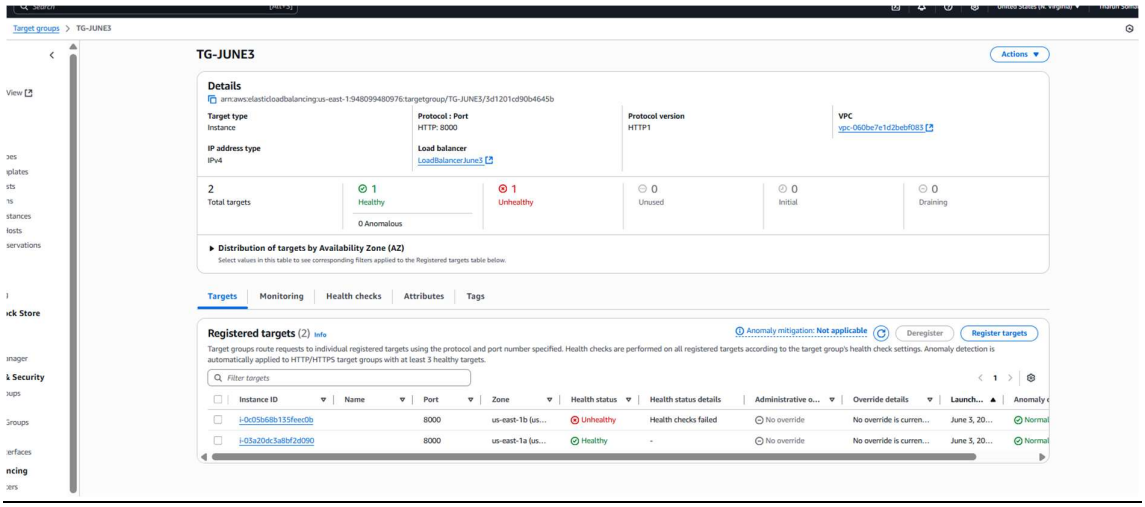
Here The user is Not Directly accessing the Private Ec2 Instances , instead Via Load Balancer User can Access the Website that is created in the Private Ec2 ..

Therefore, Now Our Main Goal is Achieved



Welcome to My AWS Created on June 3rd -2025 BY THARUN SOMARAJU

This page is hosted on a private EC2 instance via a Bastion Host.



Therefore , One instance is Healthy and another one not healthy because , We Hosted the Website in only One Ec2 Instance, So the Request Goes for only One instance..

STEPS TO DISMANTLE THE RESOURCES

Here's a simple checklist to **safely dismantle each AWS tool** from your project and avoid charges:

✅ **1. Terminate EC2 Instances**

- Go to **EC2 > Instances**
- Select all running instances (including Bastion and private ones)
- Click **"Instance State" > Terminate**

✅ **2. Delete Auto Scaling Groups**

- Go to **EC2 > Auto Scaling Groups**
- Delete your Auto Scaling Group (ASG)

✅ **3. Delete Load Balancer**

- Go to **EC2 > Load Balancers**
- Select and **delete** your Application Load Balancer (ALB)

✅ **4. Delete Target Groups**

- Go to **EC2 > Target Groups**
- Select and **delete** them (if not auto-removed)

✅ **5. Delete NAT Gateway**

- Go to **VPC > NAT Gateways**
- Select and **delete**

⚠️ NAT Gateway can **incur charges even when idle** — make sure to remove it!

✅ **6. Release Elastic IP (if any)**

- Go to **VPC > Elastic IPs**
- Disassociate and then **release** unused IPs

✅ **7. Delete Subnets and Route Tables**

- Go to **VPC > Subnets** and **Route Tables**
- Delete custom ones **after** EC2s and NAT are gone

✅ **8. Delete VPC**

- Go to **VPC > Your VPCs**
- Select and **delete your custom VPC**

----------THE END----------