TASK - 3

Step 1: Create a Custom VPC

1. Navigate to VPC Dashboard

- Go to **AWS VPC Console**
- Select **Your VPCs** → Click **Create VPC**

2. Configure VPC Settings

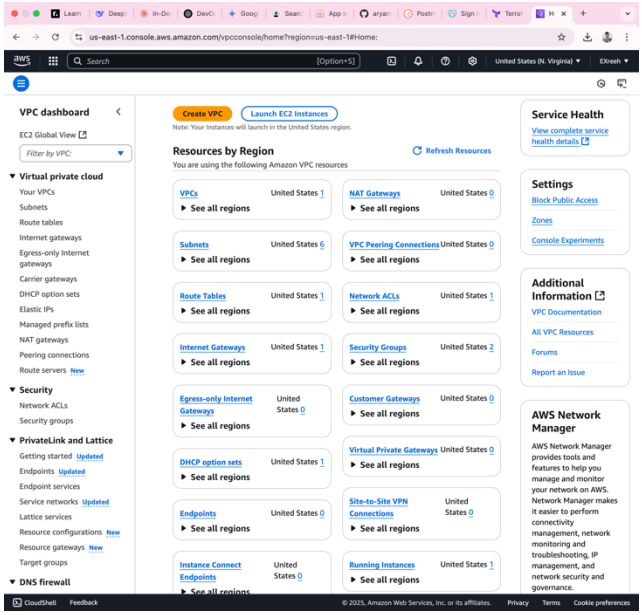
- **Name tag**: `Project-VPC` (or your preferred name)
- **IPv4 CIDR block**: `10.0.0.0/16` (provides 65,536 private IPs)
- **IPv6 CIDR block**: *Leave as "No IPv6 CIDR block"* (unless required)
- **Tenancy**: `Default` (shared hardware)

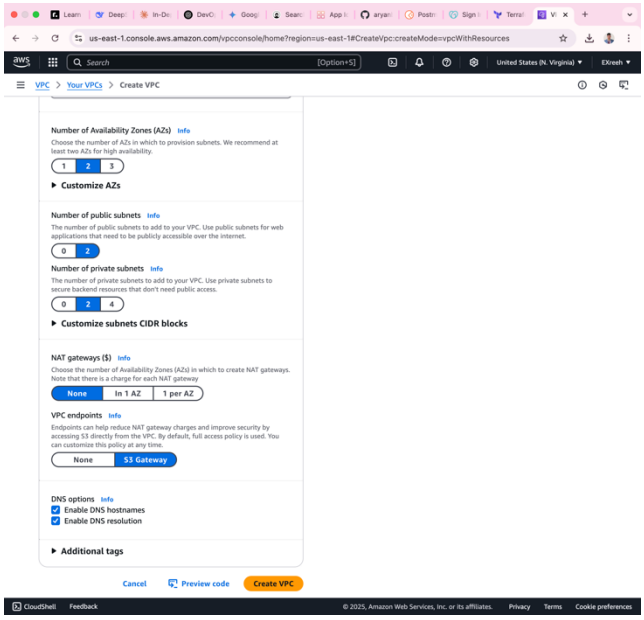
3. Enable DNS Support (Critical for EC2 Communication)

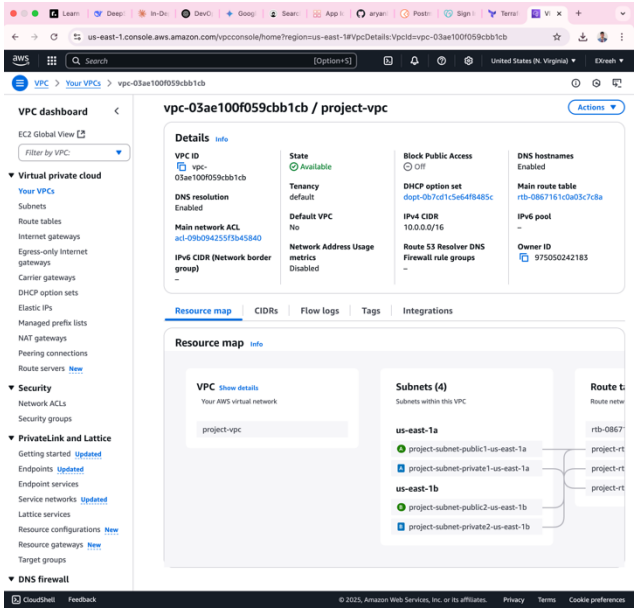
- ☑ **Enable DNS hostnames**: *Yes*
- ☑ **Enable DNS resolution**: *Yes*

4. Create the VPC

- Click **Create VPC**


us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#Home:
Search
[Option+S]
United States (N. Virginia)
EXreeh
VPC dashboard
EC2 Global View
Filter by VPC:
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only Internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways
Peering connections
Route servers New
Security
Network ACLs
Security groups
PrivateLink and Lattice
Getting started Updated
Endpoints Updated
Endpoint services
Service networks Updated
Lattice services
Resource configurations New
Resource gateways New
Target groups
DNS firewall
Create VPC
Launch EC2 Instances
Note: Your instances will launch in the United States region.
Resources by Region
Refresh Resources
You are using the following Amazon VPC resources
VPCs United States 1
NAT Gateways United States 0
Subnets United States 6
VPC Peering Connections United States 0
Route Tables United States 1
Network ACLs United States 1
Internet Gateways United States 1
Security Groups United States 2
Egress-only Internet Gateways United States 0
Customer Gateways United States 0
DHCP option sets United States 1
Virtual Private Gateways United States 0
Endpoints United States 0
Site-to-Site VPN Connections United States 0
Instance Connect Endpoints United States 0
Running Instances United States 1
See all regions
Service Health
View complete service health details
Settings
Block Public Access
Zones
Console Experiments
Additional Information
VPC Documentation
All VPC Resources
Forums
Report an Issue
AWS Network Manager
AWS Network Manager provides tools and features to help you manage and monitor your network on AWS. Network Manager makes it easier to perform connectivity management, network monitoring and troubleshooting, IP management, and network security and governance.
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#CreateVpc:createMode=vpcWithResources
Search
[Option+S]
United States (N. Virginia)
EXreeh
VPC > Your VPCs > Create VPC
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
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences



Step 2: Create Subnets in Your VPC

1. Navigate to Subnet Configuration

- Go to **AWS VPC Console**
- Select **Subnets** → Click **Create Subnet**

2. Configure Public Subnet

- **VPC ID**: Select `Project-VPC` (created earlier)
- **Subnet name**: `Public-Subnet`
- **Availability Zone**: `eu-north-1a`
- **IPv4 CIDR block**: `10.0.1.0/24`

  (Provides 251 usable IPs - 5 AWS reserved)

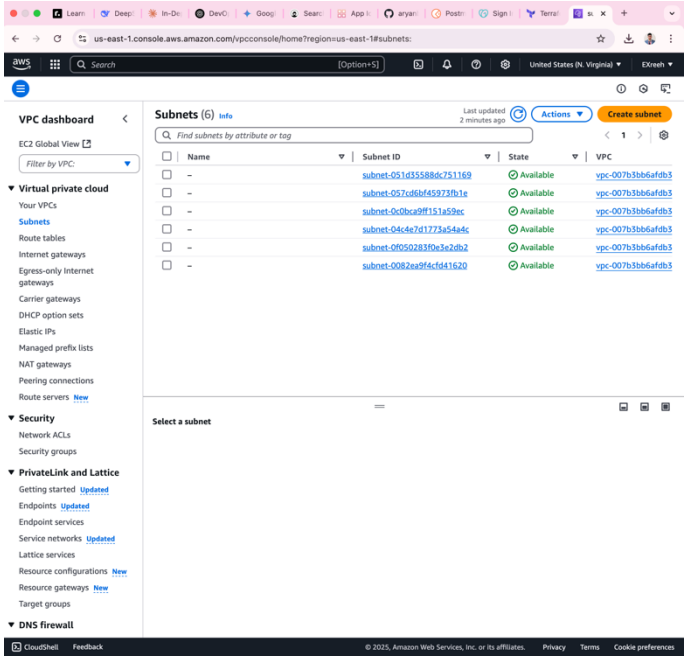
3. Configure Private Subnet

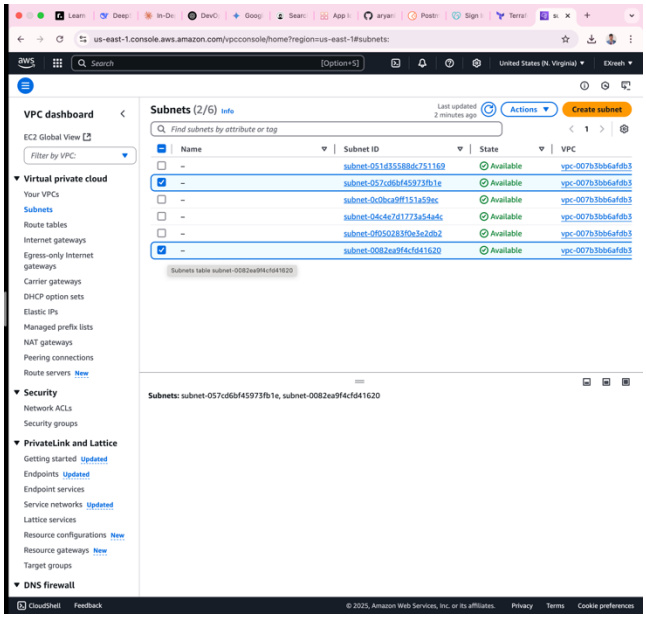
- **VPC ID**: Same `Project-VPC`
- **Subnet name**: `Private-Subnet`

- **Availability Zone**: `eu-north-1a` *(Same AZ for simplicity, but consider cross-AZ for HA)*
- **IPv4 CIDR block**: `10.0.2.0/24`

4. Create Subnets

- Click **Create Subnet** (repeat for both subnets)





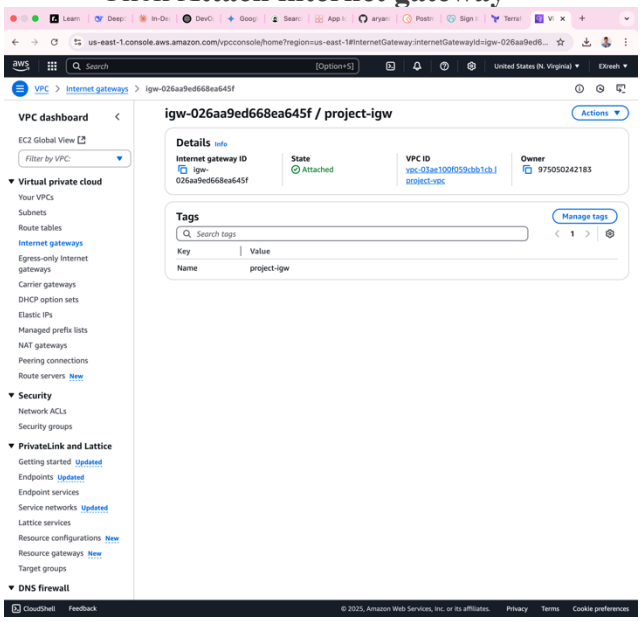
Step 3: Create and Attach an Internet Gateway (IGW)

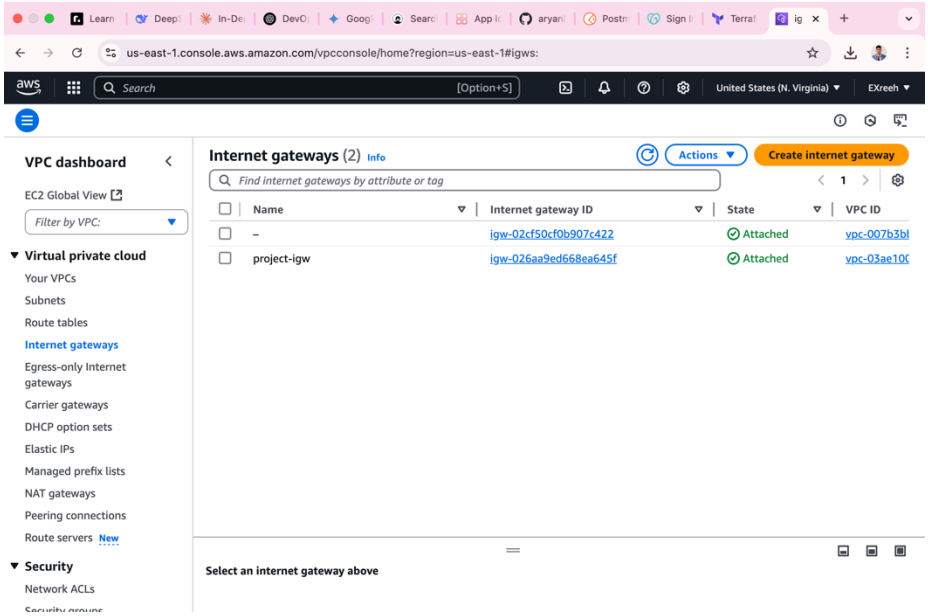
1. Create the Internet Gateway

- Go to **AWS VPC Console**
- In the left menu, select **Internet Gateways**
- Click **Create internet gateway**
- **Name tag**: `My-IGW` (or your preferred name)
- Click **Create**

2. Attach to Your VPC

- Select the newly created IGW (`My-IGW`)
- Click **Actions → Attach to VPC**
- **Available VPCs**: Select your `Project-VPC`
- Click **Attach internet gateway**





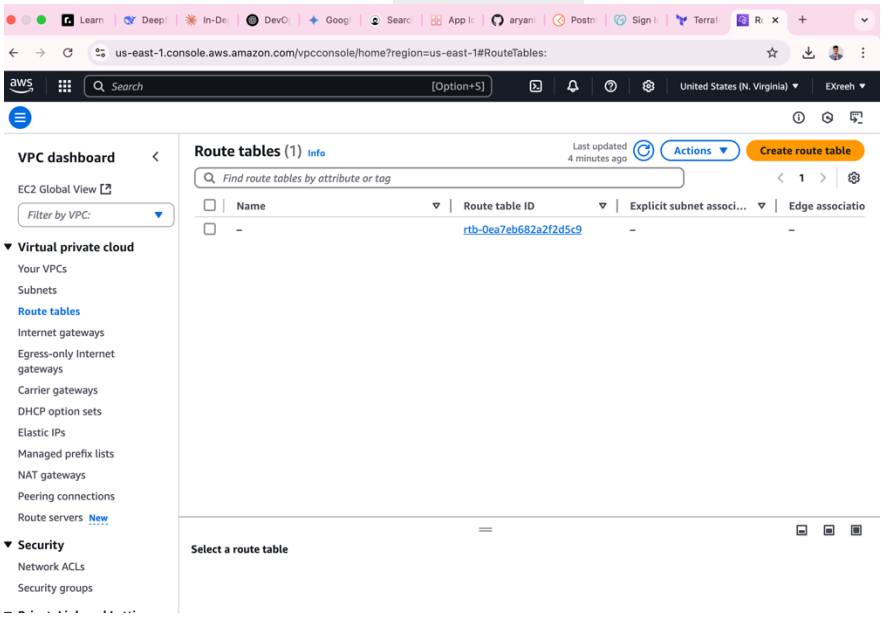
Step 4: Configure Route Tables for Public & Private Subnets

1. Create Public Route Table

- **Navigation**: VPC Console → **Route Tables** → **Create route table**
- **Settings**:
  - **Name**: `Public-RT`
  - **VPC**: Select `Project-VPC`
- **Add Internet Route**:
  - Select the new route table → **Routes** → **Edit routes** → **Add route**
    - **Destination**: `0.0.0.0/0`
    - **Target**: Select your `My-IGW` (Internet Gateway)
- **Subnet Associations**:
  - **Subnet associations** → **Edit subnet associations**
  - Select `Public-Subnet` → **Save**

2. Create Private Route Table

- **Create route table**:
  - **Name**: Private-RT
  - **VPC**: Same Project-VPC
- **Add NAT Gateway Route** *(Prerequisite: Create NAT Gateway in a public subnet)*:
  - Edit routes → Add route:
    - **Destination**: 0.0.0.0/0
    - **Target**: Select your NAT Gateway (e.g., nat-xxxx)
- **Subnet Associations**:
  - Associate with Private-Subnet



Step 5: Create Security Groups for Secure Access

1. Bastion Host Security Group (Bastion-SG)

- **Navigation**: EC2 Console → **Security Groups** → **Create security group**
- **Basic Details**:
  - **Security group name**: Bastion-SG
  - **Description**: "Allow SSH access to bastion host"
  - **VPC**: Select Project-VPC

- **Inbound Rules**:

Type	Port	Source	Description
SSH	22	Your.IP.Address/32	Restrict SSH to your IP

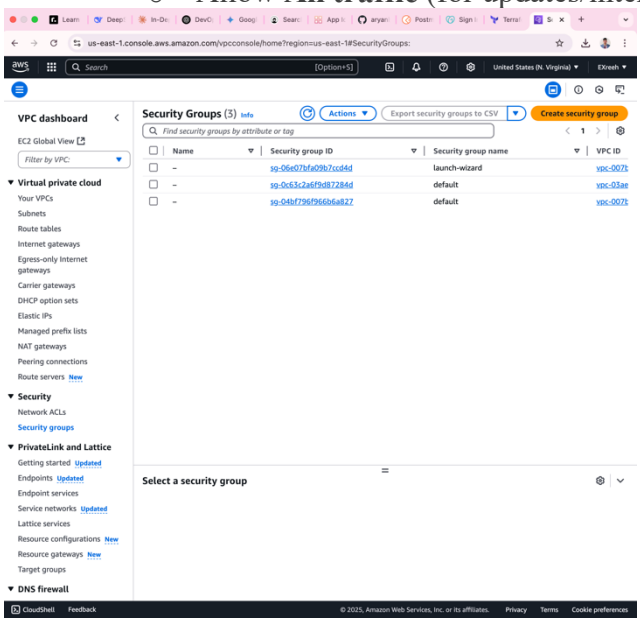
- **Outbound Rules**:
  - Allow **All traffic** (default)

2. Backend Security Group (Backend-SG)

- **Create security group**:
  - **Name**: Backend-SG
  - **Description**: "Allow SSH from bastion and internal traffic"
  - **VPC**: Project-VPC
- **Inbound Rules**:

Type	Port	Source	Description
SSH	22	10.0.1.0/24	Only from public subnet
Custom TCP	e.g., 8080	10.0.1.0/24	For Jenkins/backend apps

- **Outbound Rules**:
  - Allow **All traffic** (for updates/internal communication)



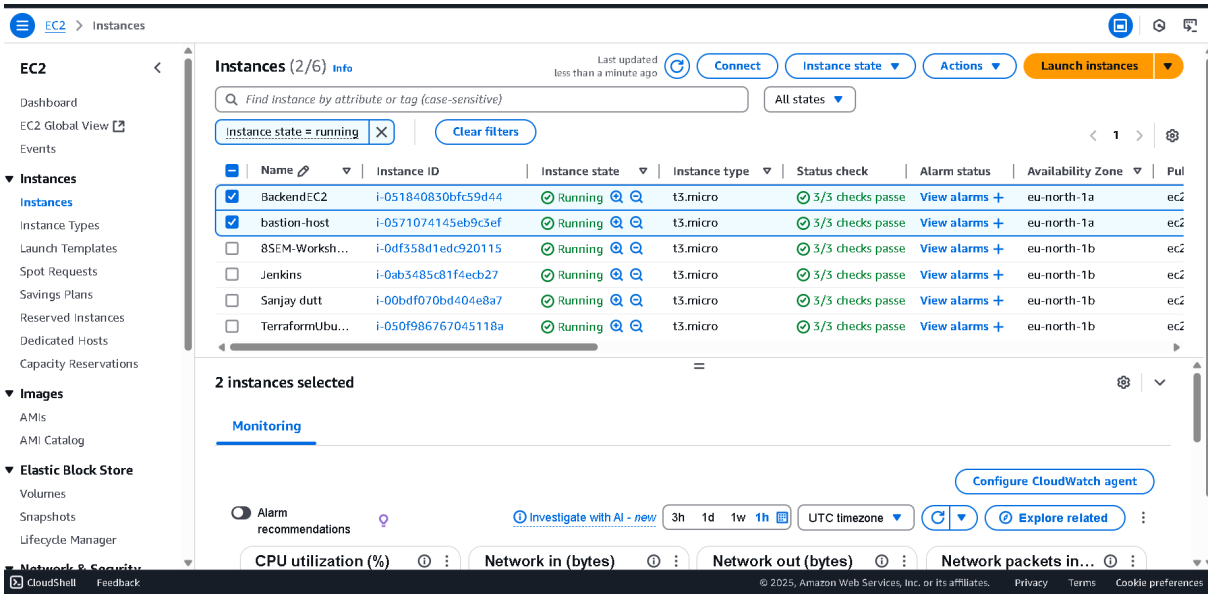
Step 6: Launch EC2 Instances in Public & Private Subnets

1. Bastion Host (Public Jump Server)

- **Navigation**: EC2 Console → **Launch Instance**
- **Configuration**:
  - **Name**: `Bastion-Host`
  - **AMI**: Ubuntu Server 22.04 LTS (Free Tier eligible)
  - **Instance Type**: `t3.micro`
  - **Key Pair**: Select existing or create new `.pem` key
  - **Network Settings**:
    - **VPC**: `Project-VPC`
    - **Subnet**: `Public-Subnet` (eu-north-1a)
    - ☑ **Auto-assign Public IP**: Enable
    - **Security Group**: `Bastion-SG` (SSH only from your IP)
  - **Storage**: Keep default 8GB GP2 volume

2. Backend Server (Private Instance)

- **Launch Instance**:
  - **Name**: `Backend-Server`
  - **AMI**: Same Ubuntu AMI
  - **Instance Type**: `t3.micro`
  - **Same Key Pair**: Reuse bastion's `.pem` file
  - **Network Settings**:
    - **Subnet**: `Private-Subnet` (eu-north-1a)
    - **Auto-assign Public IP**: Disable (default)
    - **Security Group**: `Backend-SG` (SSH only from `10.0.1.0/24`)
  - **Storage**: 8GB GP2 (or increase as needed)



Step 7: Verify Bastion Host Access & Terraform Setup

1. Test SSH Access to Bastion Host

```
ssh -i "C:\Users\disha\Downloads\bostin-host.pem" ubuntu@13.51.60.153
```

Key Checks:

- Replace `13.51.60.153` with your Bastion's **Elastic IP**
- Ensure:
  - Key permissions are secure:
    ```
    chmod 400 bostin-host.pem
    icacls .\bostin-host.pem /reset /inheritance:r /grant:r "%USERNAME%":"(R)"
    ```
  - Security Group allows your current IP (check here)

2. Terraform Project Structure

```
terraform/
├── main.tf
├── variables.tf
├── outputs.tf
└── terraform.tfvars
```

3. Terraform Workflow

1. **Initialize** (download providers/modules):
   terraform init
2. **Preview Changes** (safety check):
   terraform plan
3. **Apply Configuration**:
   terraform apply