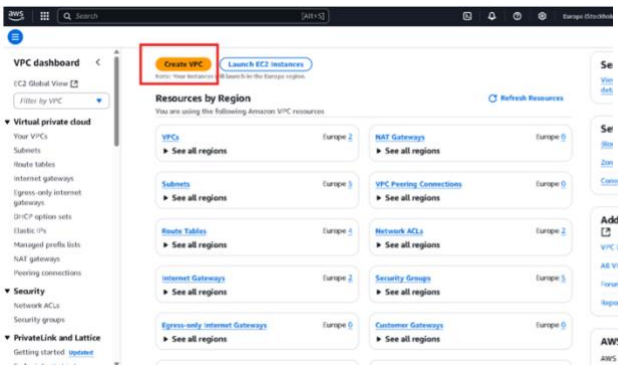
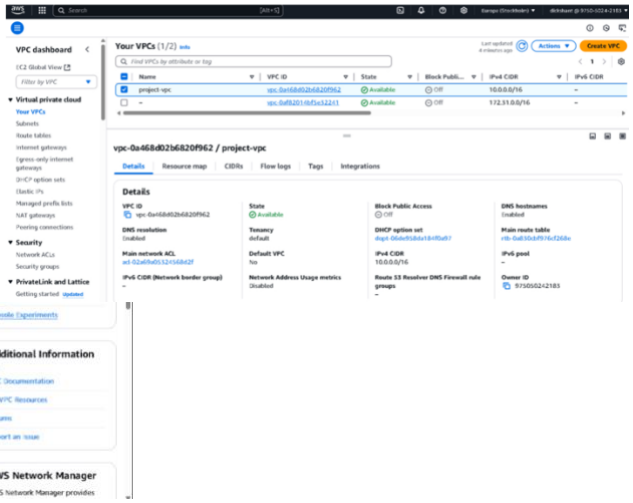
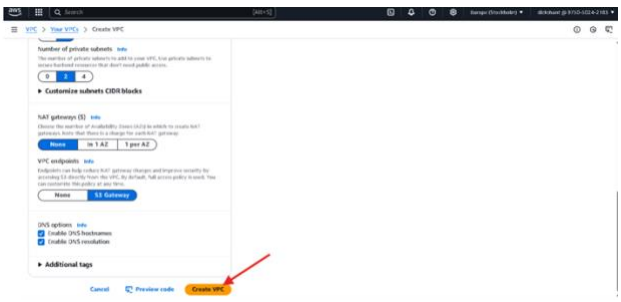
TASK 3

Step 1: Set Up the VPC

1. Head over to the VPC Console, click on Your VPCs, then choose Create VPC.
2. Set the Name tag to Project-VPC.
3. Specify the IPv4 CIDR block as
4. 10.0.0.0/16.
5. Make sure DNS hostnames are turned on.
6. Hit the Create VPC button.

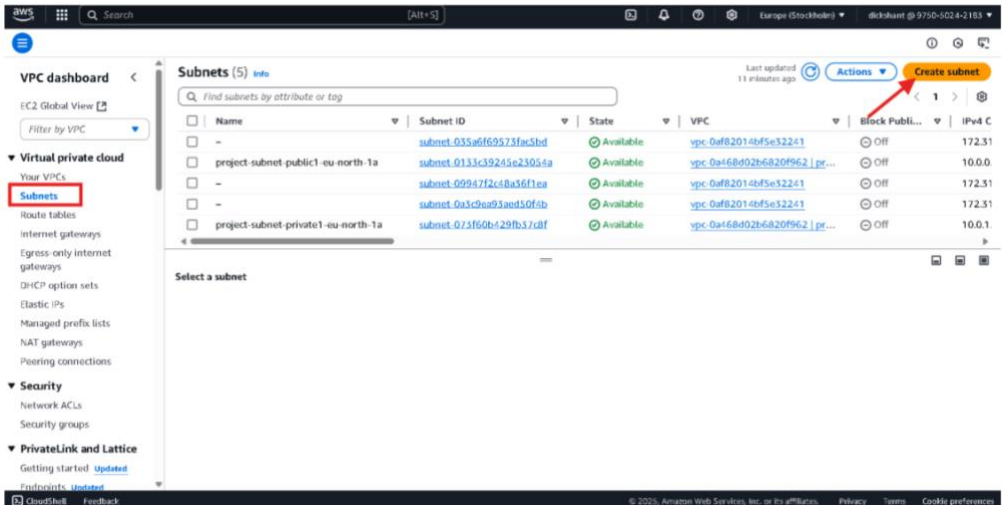


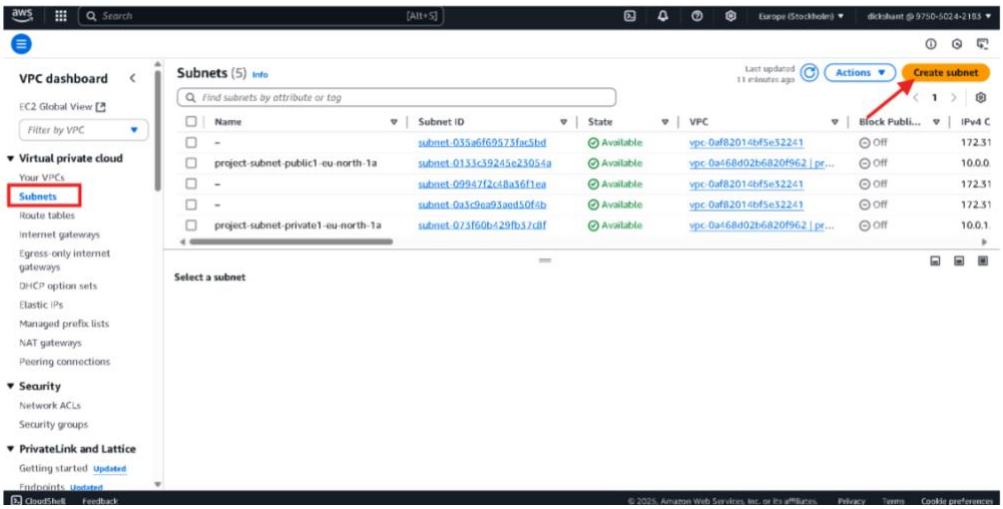


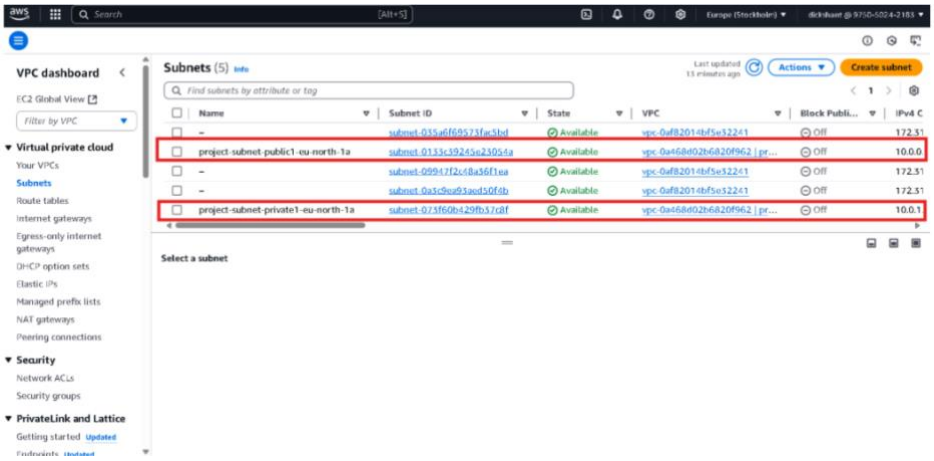


Step 2: Define Subnets

1. Navigate to the Subnets section and click Create Subnet.
2. Choose your previously created VPC.
3. For the Public Subnet:
    - Name it Public-Subnet.
    - Pick the availability zone eu-north-1a.
    - Set the CIDR block to 10.0.1.0/24.
4. For the Private Subnet:
    - Name it Private-Subnet.
    - Use the same AZ eu-north-1a.
    - Use 10.0.2.0/24 as the CIDR block

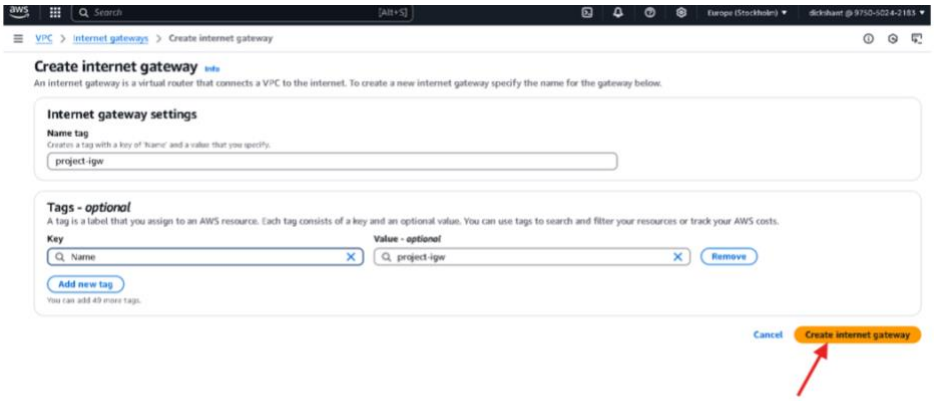


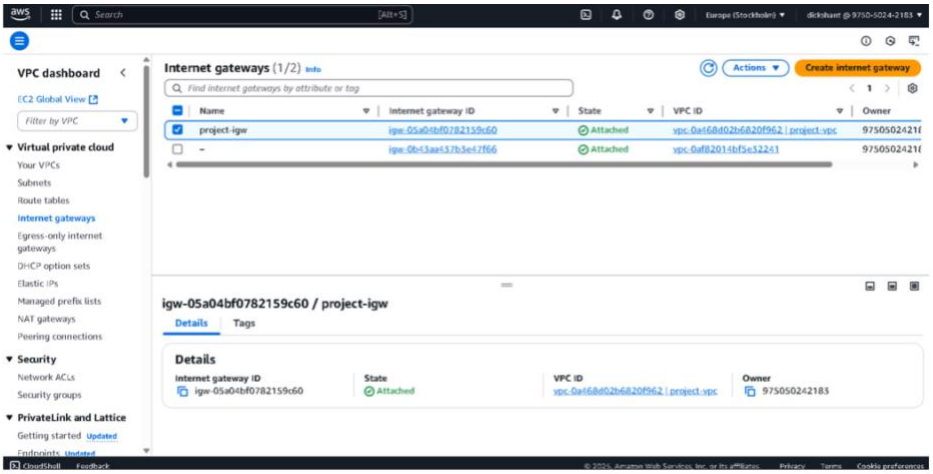




Step 3: Provision an Internet Gateway

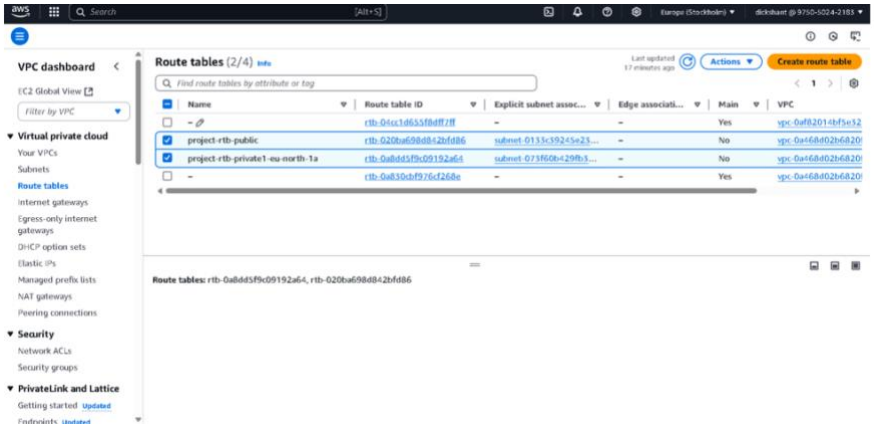
1. Go to the Internet Gateways tab and click Create Internet Gateway.
2. Give it the name My-IGW.
3. After creation, go to Actions, select Attach to VPC, and link it to your Project-VPC.





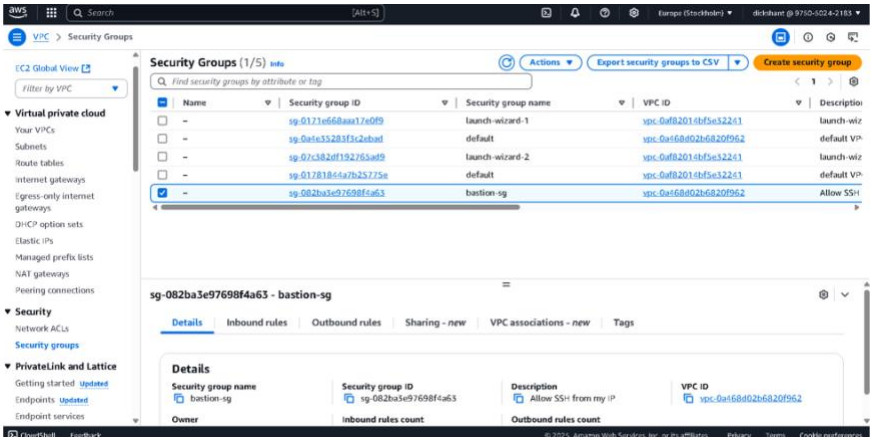
Step 4: Set Up Route Tables

1. In the Route Tables area, create a new table:
    - Name it Public-RT.
    - Link it to your Project-VPC.
2. Add a new route:
    - Destination: 0.0.0.0/0
    - Target: Your Internet Gateway (My-IGW)
3. Link this route table to your Public-Subnet.
4. Create another route table named Private-RT.
5. Add a route in Private-RT:
    - Destination: 0.0.0.0/0
    - Target: A NAT Gateway (to be created separately)
6. Associate Private-RT with Private-Subnet.



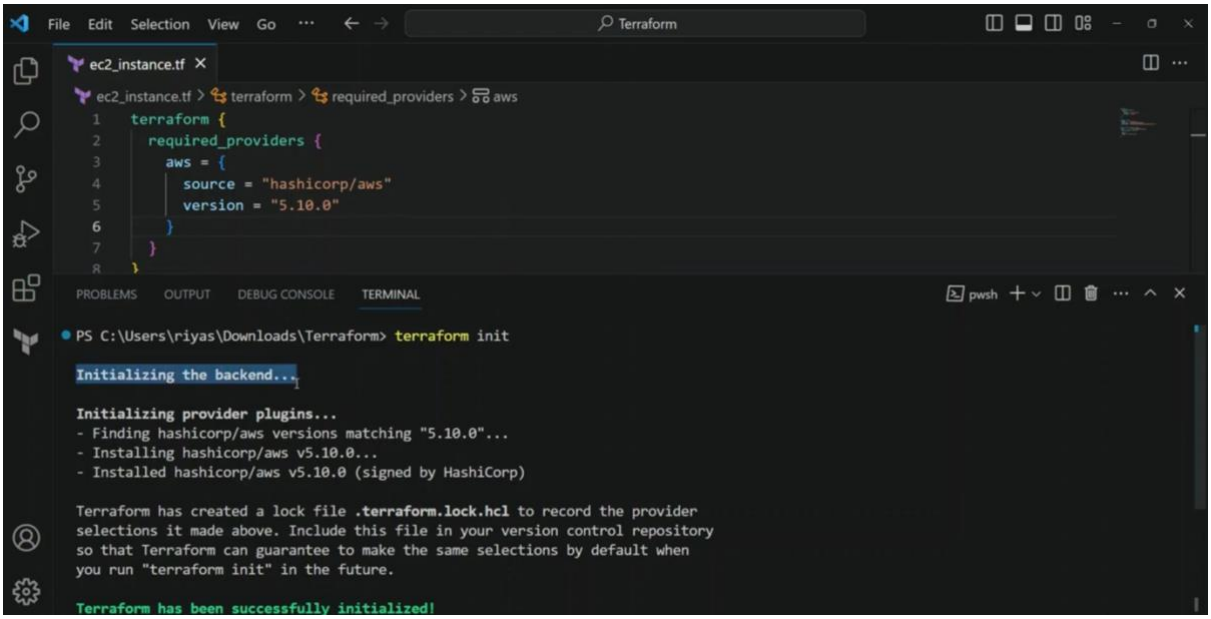
Step 5: Configure Security Groups

1. Create a security group for the Bastion host:
    - Name: Bastion-SG
    - Inbound Rules: Allow SSH (port 22) only from your IP.
    - Outbound: Allow all traffic.
2. Set up a second group for backend instances:
    - Name: Backend-SG
    - Inbound Rules: Permit SSH (port 22) only from 10.0.1.0/24 (your public subnet).
    - Outbound: Allow all traffic.



Step 6: Test SSH access to Bastion Hostfrom your local machine using the Elastic IP:

1. ssh -i C:\Users\disha\Downloads\bostin-host.pem ubuntu@13.51.60.153
2. terraform/
   ├── main.tf
   ├── variables.tf
   ├── outputs.tf
   └── terraform.tfvars
3. terraform init
4. terraform plan
5. terraform apply



```
- Installing hashicorp/aws v5.10.0...
PS C:\Users\riyas\Downloads\Terraform> terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following
symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.web will be created
  + resource "aws_instance" "web" {
      + ami                                  = "ami-053b0d53c279acc90"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
```

```
      + user_data                            = (known after apply)
      + user_data_base64                     = (known after apply)
      + user_data_replace_on_change          = false
      + vpc_security_group_ids               = (known after apply)
    }

Plan: 1 to add, 0 to change, 0 to destroy.

---

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run
"terraform apply" now.
PS C:\Users\riyas\Downloads\Terraform> terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following
symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.web will be created
  + resource "aws_instance" "web" {
      + ami                                  = "ami-053b0d53c279acc90"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
```