AWS VPC Project Documentation

Overview

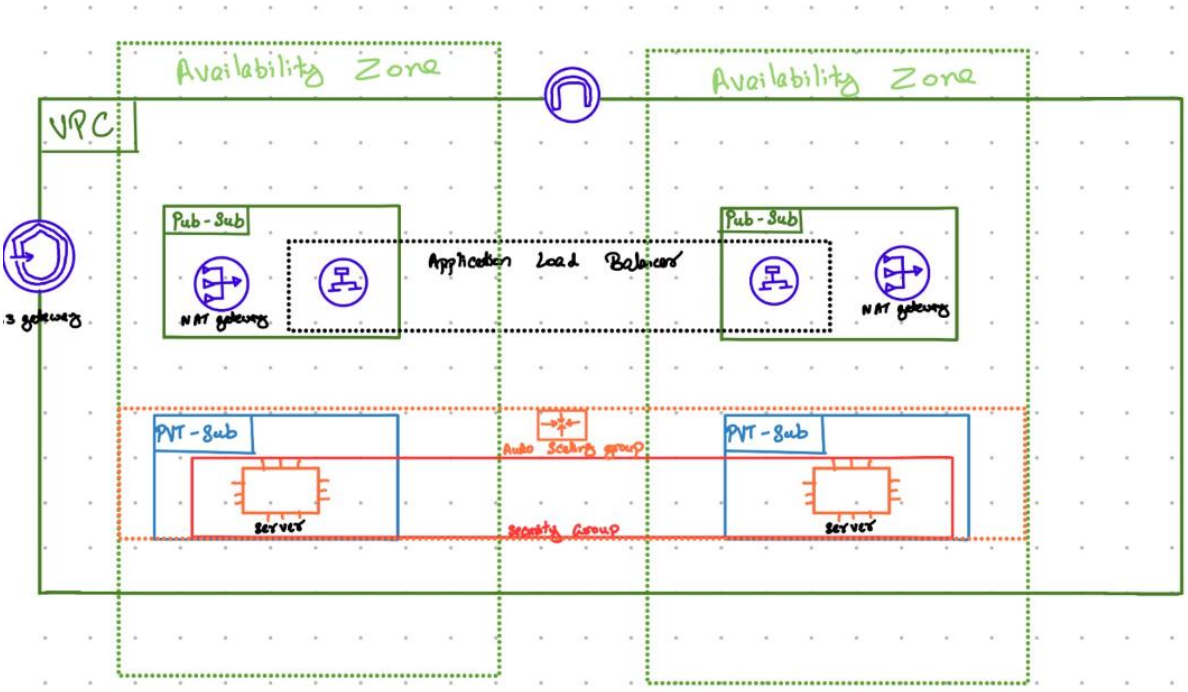
This example demonstrates how to create a VPC suitable for servers in a production environment. To improve resiliency, servers are deployed in two Availability Zones using an Auto Scaling group and an Application Load Balancer. For additional security, the servers are deployed in private subnets, receiving requests through the load balancer and connecting to the internet via a NAT gateway. The NAT gateway is deployed in both Availability Zones for redundancy.

Architecture Summary

- **VPC:** Contains public and private subnets in two Availability Zones.
- **Public Subnets:** Each contains a NAT gateway and a load balancer node.
- **Private Subnets:** Servers run here, managed by an Auto Scaling group, and receive traffic from the load balancer.
- **Internet Access:** Servers connect to the internet using the NAT gateway.

Prerequisites

- Auto Scaling Group
- Load Balancer
- Target Group
- Bastion Host or Jump Server

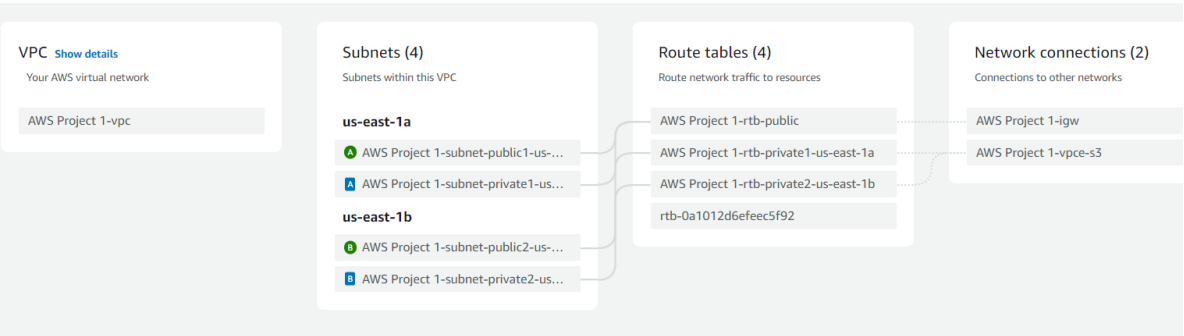


Steps to Create the VPC

1. **Create VPC**
    - VPC ID: `vpc-08c51ec2cb1d149fa`
    - Enable DNS hostnames
    - Enable DNS resolution
    - Verify VPC creation
2. **Create Subnets**
    - Subnet IDs: `subnet-0adc04bde570786be`, `subnet-0ca9e39f1b938a02f`, `subnet-0bf4c8b709c863cb8`, `subnet-0f4f6f9bf0fe50457`
3. **Create and Attach Internet Gateway**
    - Internet Gateway ID: `igw-0c3e6e451fb66d6e8`
4. **Create and Associate Route Tables**
    - Route Table IDs: `rtb-05efbba945d0d6871`, `rtb-0dc303469d7e89804`, `rtb-0802896e0c749542c`
    - Associate S3 endpoint with private subnet route tables: `vpce-0ed17b649839b54f3`

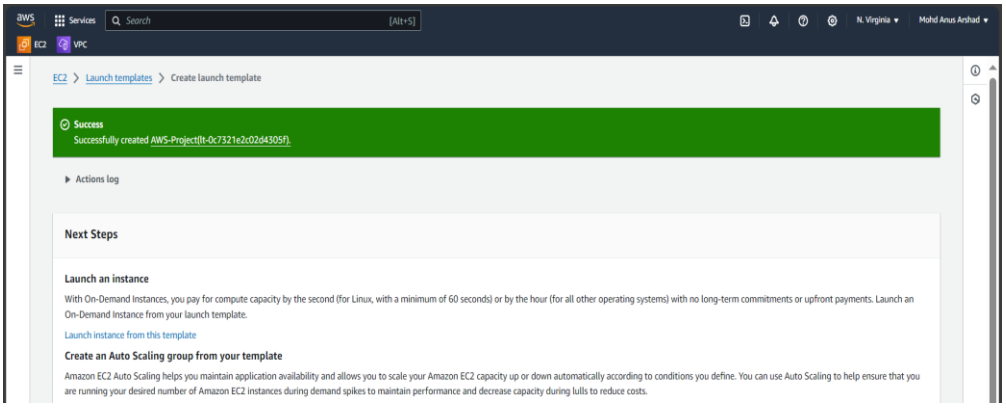
This things will be done during creation of VPC:

- Create VPC: vpc-08c51ec2cb1d149fa
- Enable DNS hostnames
- Enable DNS resolution
- Verifying VPC creation: vpc-08c51ec2cb1d149fa
- Create S3 endpoint: vpce-0ed17b649839b54f3
- Create subnet: subnet-0adc04bde570786be
- Create subnet: subnet-0ca9e39f1b938a02f
- Create subnet: subnet-0bf4c8b709c863cb8
- Create subnet: subnet-0f4f6f9bf0fe50457
- Create internet gateway: igw-0c3e6e451fb66d6e8
- Attach internet gateway to the VPC
- Create route table: rtb-05efbba945d0d6871
- Create route
- Associate route table
- Associate route table
- Create route table: rtb-0dc303469d7e89804
- Associate route table
- Create route table: rtb-0802896e0c749542c
- Associate route table
- Verifying route table creation
- Associate S3 endpoint with private subnet route tables: vpce-0ed17b649839b54f3

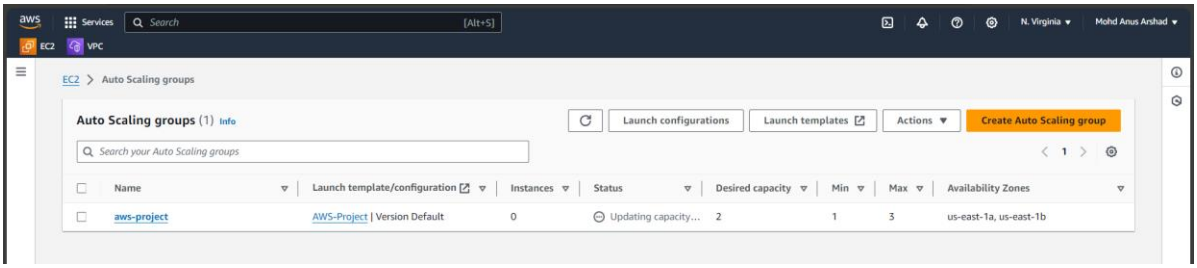


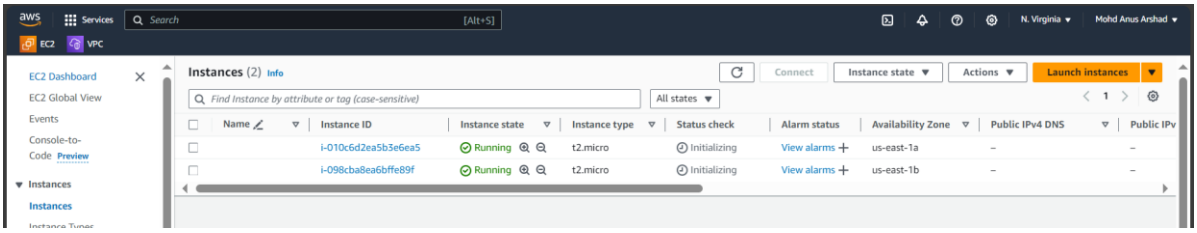
Configuring Auto Scaling Group (ASG)

1. **Create Launch Template**



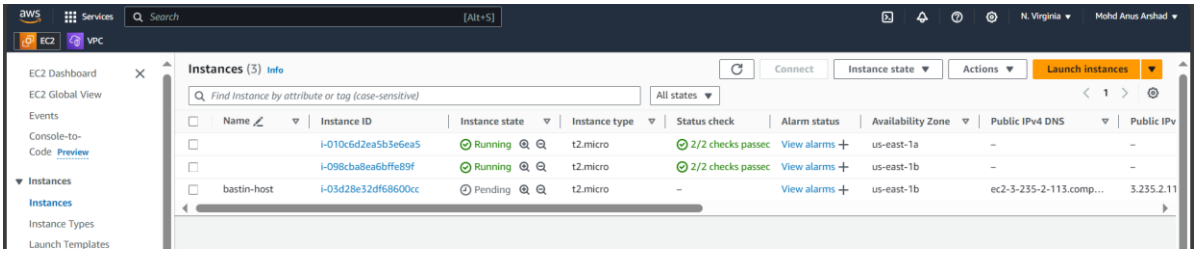
2. **Create Auto Scaling Group**
    - Verify instances running in private subnets with the help of ASG



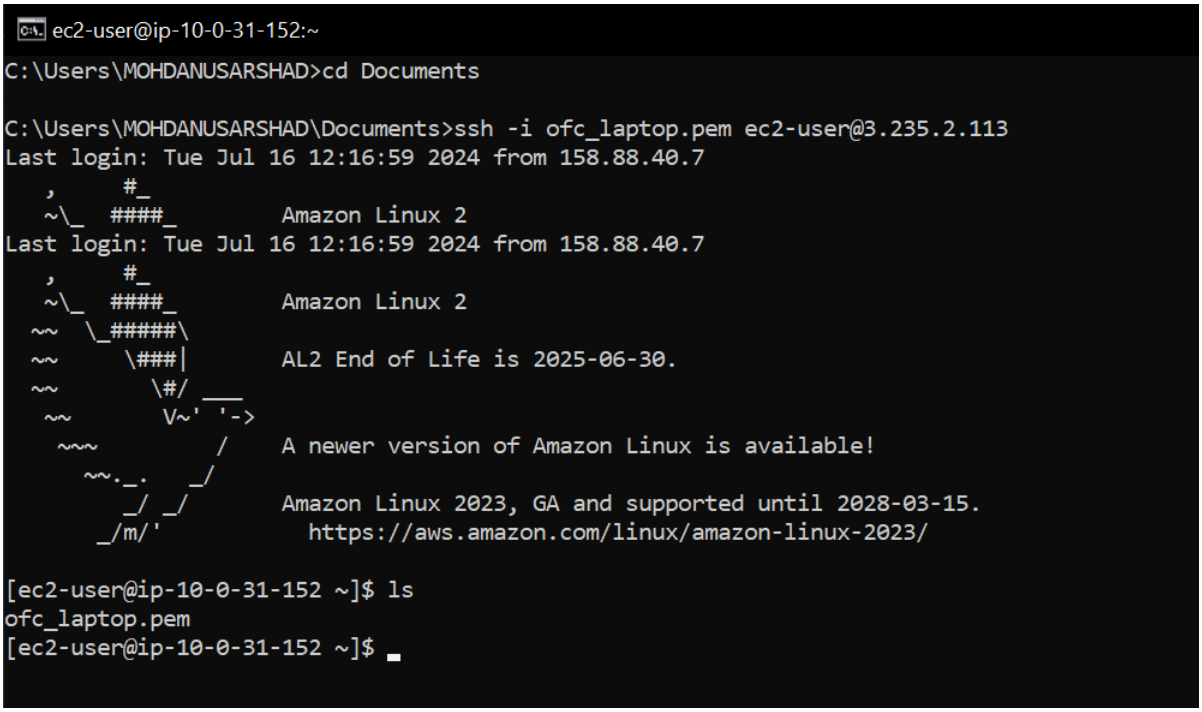


Setting Up Bastion Host

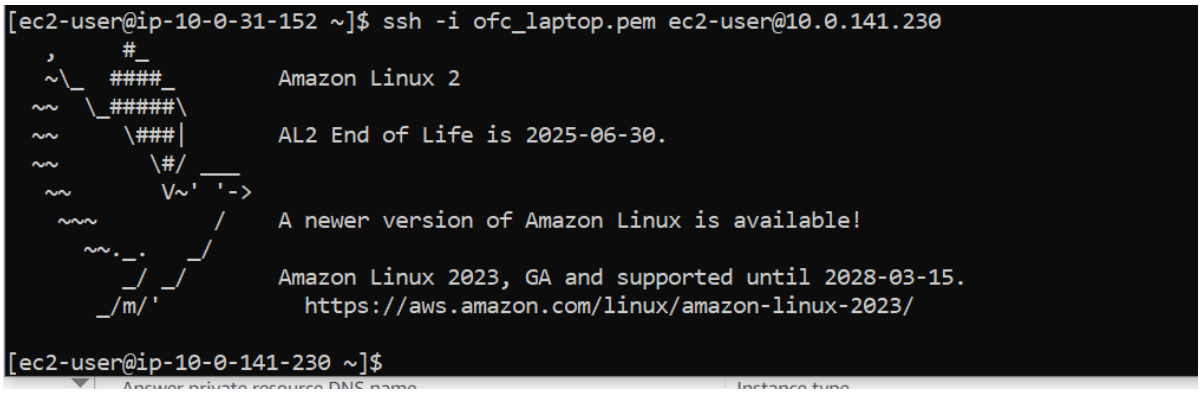
1. **Launch Bastion Host Instance**



- Place key pair of private instances in the Bastion Host server using Puttygen to convert `.pem` to `.ppk`, then transfer the file using WinSCP.

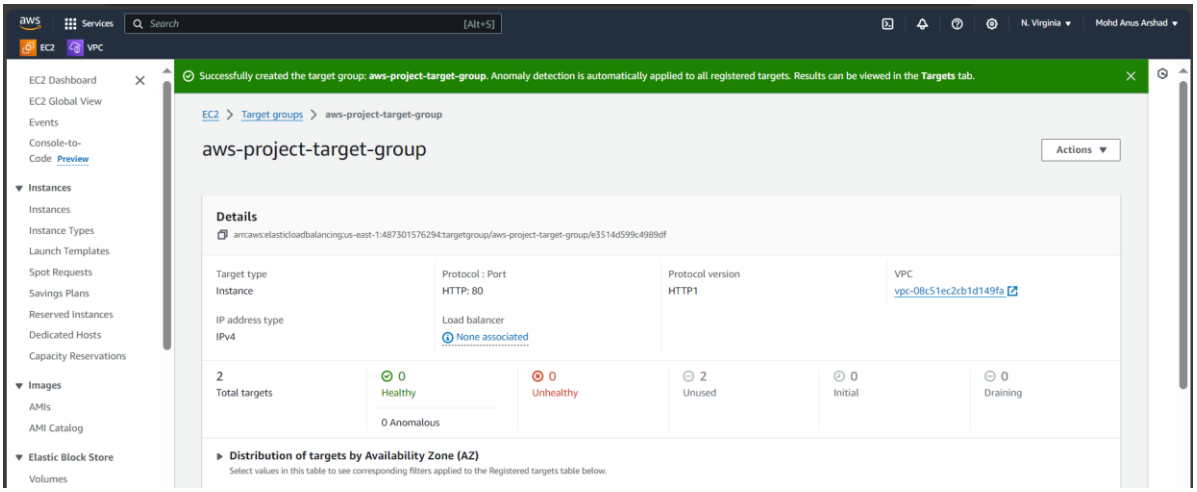


2. **Connect to Private Instances Using Bastion Host**

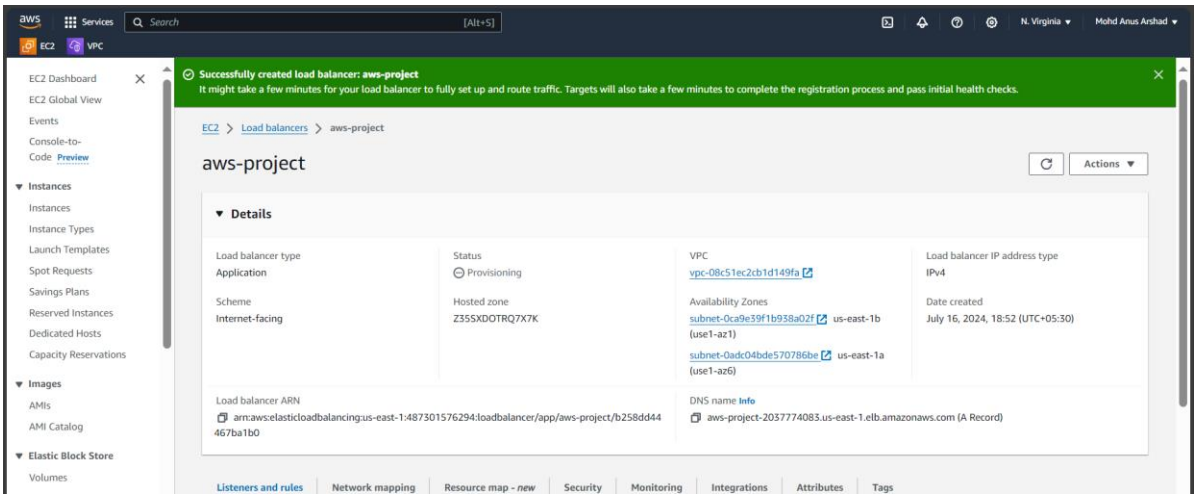


Setting Up Apache Server and ELB

1. **Setup Apache Server**
   - Install Apache on one of the private instances.
   - Create a simple HTML file (e.g., "Hello World").
2. **Create Elastic Load Balancer (ELB)**
   - Create a target group for the ELB.

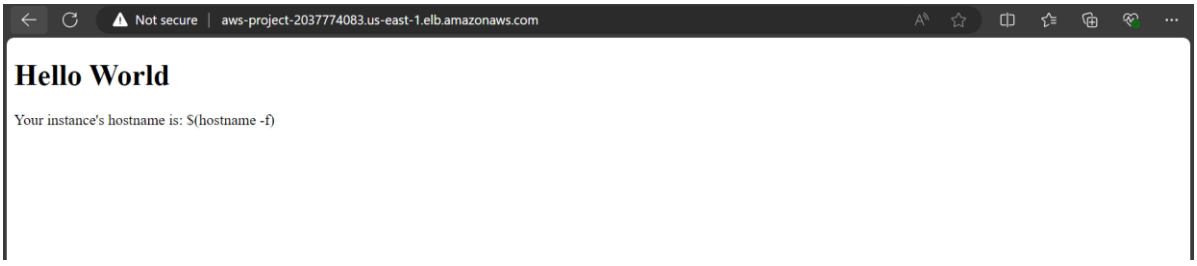


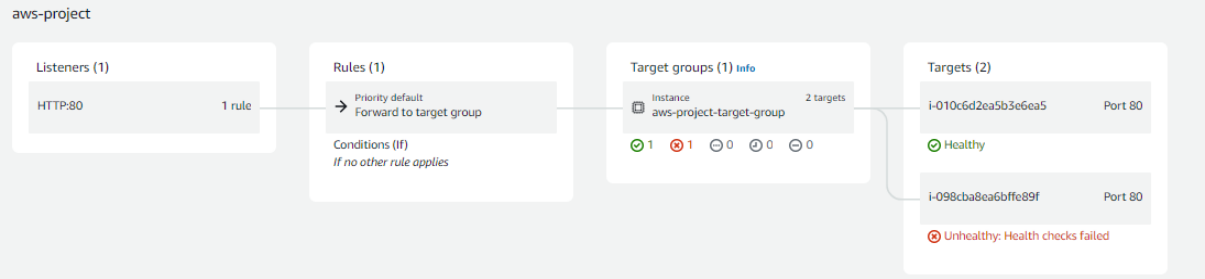
- Ensure the ELB is successfully created in the public subnet.



3. **Check Server Health**

- One server might be unhealthy if the application is not set up correctly.





Conclusion

The project demonstrates the creation and configuration of a VPC for a production environment, including setting up Auto Scaling, a Load Balancer, and secure access through a Bastion Host. The setup includes resilient and scalable infrastructure using AWS best practices.

Project completed!!