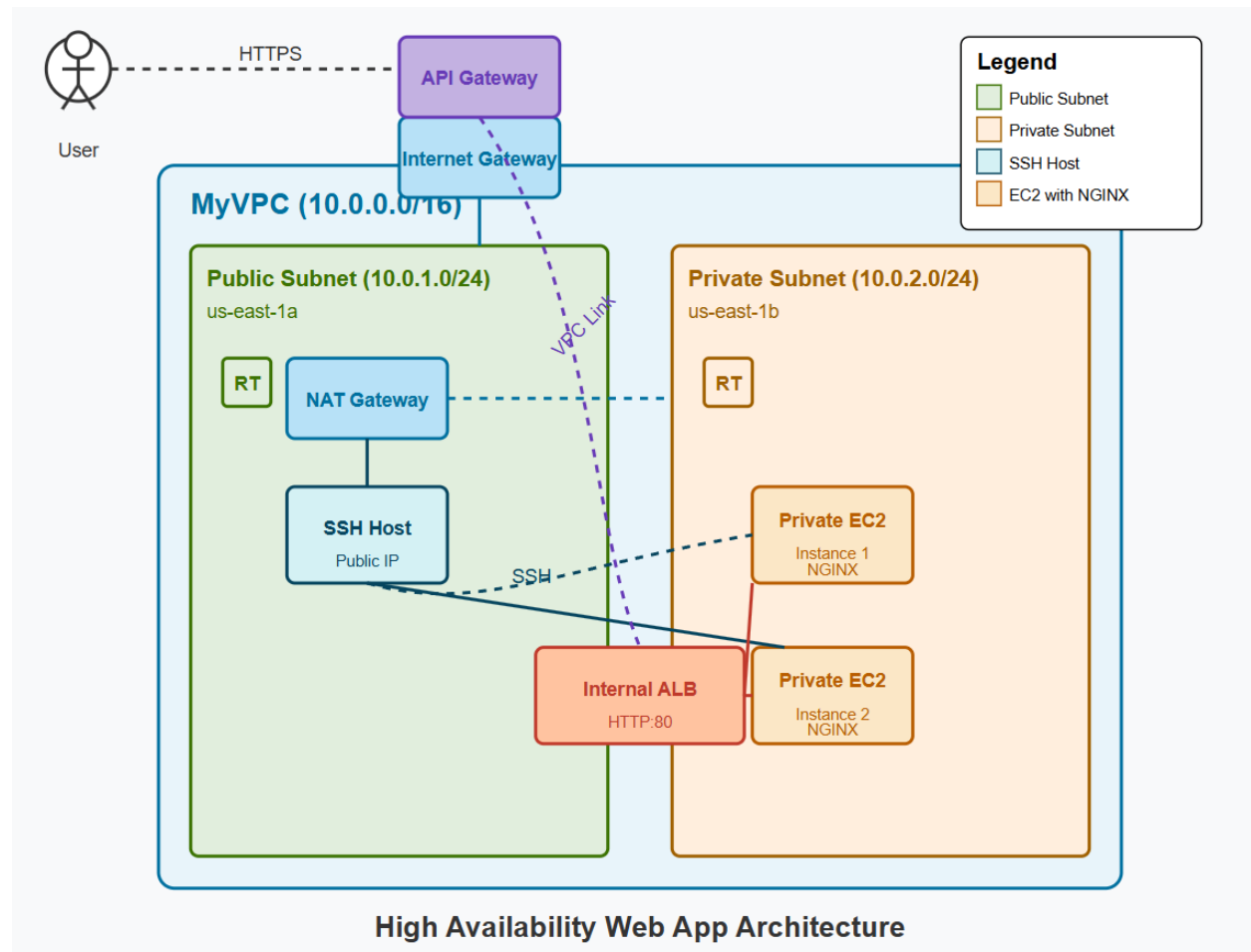


High Availability Web App with Internal ALB, API Gateway, and SSH Host



Project Objective

Deploy a highly available and secure web application on AWS that:

- Runs on private EC2 instances behind an internal Application Load Balancer (ALB)
- Is accessible publicly through API Gateway (HTTPS)

- Uses a publicly accessible SSH Host in a public subnet to securely access private EC2 instances
- Utilizes only AWS CLI (no root account or console access)

Pre-Requisites

- AWS CLI configured
- IAM permissions to use EC2, VPC, IAM, ELB, SSM, and API Gateway
- RHEL shell running inside WSL on Windows

1. Create VPC

Command:

```
aws ec2 create-vpc --cidr-block 10.0.0.0/16 --tag-specifications  
'ResourceType=vpc,Tags=[{Key=Name,Value=MyVPC}]'
```

```
[root@rhelclient home]# aws ec2 create-vpc --cidr-block 10.0.0.0/16 --tag-specifications 'ResourceType=vpc,Tags=[{Key=Name,Value=ProjectABCCorp}]'
```

```
{  
  "Vpc": {  
    "OwnerId": "551899295811",  
    "InstanceTenancy": "default",  
    "Ipv6CidrBlockAssociationSet": [],  
    "CidrBlockAssociationSet": [  
      {  
        "AssociationId": "vpc-cidr-assoc-02528cfad97bcb738",  
        "CidrBlock": "10.0.0.0/16",  
        "CidrBlockState": {  
          "State": "associated"  
        }  
      }  
    ],  
    "IsDefault": false,  
    "Tags": [  
      {  
        "Key": "Name",  
        "Value": "ProjectABCCorp"  
      }  
    ],  
    "VpcId": "vpc-02870049f6b4e3df8",  
    "State": "pending",  
    "CidrBlock": "10.0.0.0/16",  
    "DhcpOptionsId": "dopt-089681d47180034cf"  
  }  
}
```

```
[root@rhelclient home]#
```

Where: RHEL shell with AWS CLI configured

Why: Creates an isolated virtual network for all infrastructure

Impact: All resources are deployed inside this VPC

2. Create Route Tables Before Subnets

Command to create Private Route Table:

```
aws ec2 create-route-table --vpc-id <vpc-id> --tag-specifications  
'ResourceType=route-table,Tags=[{Key=Name,Value=PrivateRT}]'
```

```
[root@rhelclient home]# aws ec2 create-route-table --vpc-id vpc-02870049f6b4e3df8 --tag-specifications 'ResourceType=route-table,Tags=[{Key=Name,Value=PrivateRT}]'  
{  
  "RouteTable": {  
    "Associations": [],  
    "PropagatingVgws": [],  
    "RouteTableId": "rtb-0b4fa860bf0817f9c",  
    "Routes": [  
      {  
        "DestinationCidrBlock": "10.0.0.0/16",  
        "GatewayId": "local",  
        "Origin": "CreateRouteTable",  
        "State": "active"  
      }  
    ],  
    "Tags": [  
      {  
        "Key": "Name",  
        "Value": "PrivateRT"  
      }  
    ],  
    "VpcId": "vpc-02870049f6b4e3df8",  
    "OwnerId": "551899295811"  
  },  
  "ClientToken": "58e6641c-f122-4428-a27d-54a78092c791"  
}
```

Command to get the Main (Public) Route Table:

```
aws ec2 describe-route-tables --filters "Name=vpc-id,Values=vpc-id" --  
query 'RouteTables[?Associations[?Main==`true`]].RouteTableId' --output  
text
```

```
[root@rhelclient home]# aws ec2 describe-route-tables --filters "Name=vpc-id,Values=vpc-02870049f6b4e3df8" --query 'RouteTables[?Associations[?Main==`true`]].RouteTableId' --output text  
rtb-079de18fc5ff6209a  
[root@rhelclient home]#
```

Where: RHEL shell

Why: Routing logic must be in place before subnet creation

Impact: Enables NAT vs IGW separation for traffic

3. Create Subnets

Command for Public Subnet:

```
aws ec2 create-subnet --vpc-id <vpc-id> --cidr-block 10.0.1.0/24 --availability-zone us-east-1a --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=PublicSubnet}]'
```

```
[root@helixliant home]# aws ec2 create-subnet --vpc-id vpc-02870049f6b4e3df8 --cidr-block 10.0.1.0/24 --availability-zone us-east-1a --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=PublicSubnet}]'
```

```
{
  "Subnet": {
    "AvailabilityZoneId": "usel-azd",
    "OwnerId": "551899295811",
    "AssignIpv6AddressesOnCreation": false,
    "Ipv6CidrBlockAssociationSet": [],
    "Tags": [
      {
        "Key": "Name",
        "Value": "PublicSubnet"
      }
    ],
    "SubnetArn": "arn:aws:ec2:us-east-1:551899295811:subnet/subnet-0f31c41fe7a08609",
    "EnableDns64": false,
    "Ipv6Native": false,
    "PrivateDnsNameOptionsOnLaunch": {
      "HostnameType": "ip-name",
      "EnableResourceNameDnsRecord": false,
      "EnableResourceNameDnsAAAARecord": false
    },
    "SubnetId": "subnet-0f31c41fe7a08609",
    "State": "available",
    "VpcId": "vpc-02870049f6b4e3df8",
    "CidrBlock": "10.0.1.0/24",
    "AvailableIpAddressCount": 251,
    "AvailabilityZone": "us-east-1a",
    "DefaultForAz": false,
    "MapPublicIpOnLaunch": false
  }
}
```

```
[root@helixliant home]#
```

Command for Private Subnet:

```
aws ec2 create-subnet --vpc-id <vpc-id> --cidr-block 10.0.2.0/24 --availability-zone us-east-1b --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=PrivateSubnet}]'
```

```
[root@helixliant home]# aws ec2 create-subnet --vpc-id vpc-02870049f6b4e3df8 --cidr-block 10.0.2.0/24 --availability-zone us-east-1b --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=PrivateSubnet}]'
```

```
{
  "Subnet": {
    "AvailabilityZoneId": "usel-azb",
    "OwnerId": "551899295811",
    "AssignIpv6AddressesOnCreation": false,
    "Ipv6CidrBlockAssociationSet": [],
    "Tags": [
      {
        "Key": "Name",
        "Value": "PrivateSubnet"
      }
    ],
    "SubnetArn": "arn:aws:ec2:us-east-1:551899295811:subnet/subnet-0a87ace1177174f0e",
    "EnableDns64": false,
    "Ipv6Native": false,
    "PrivateDnsNameOptionsOnLaunch": {
      "HostnameType": "ip-name",
      "EnableResourceNameDnsRecord": false,
      "EnableResourceNameDnsAAAARecord": false
    },
    "SubnetId": "subnet-0a87ace1177174f0e",
    "State": "available",
    "VpcId": "vpc-02870049f6b4e3df8",
    "CidrBlock": "10.0.2.0/24",
    "AvailableIpAddressCount": 251,
    "AvailabilityZone": "us-east-1b",
    "DefaultForAz": false,
    "MapPublicIpOnLaunch": false
  }
}
```

```
[root@helixliant home]#
```

Command to associate Private Subnet with Private Route Table:

```
aws ec2 associate-route-table --route-table-id <private-rt-id> --subnet-id <private-subnet-id>
```

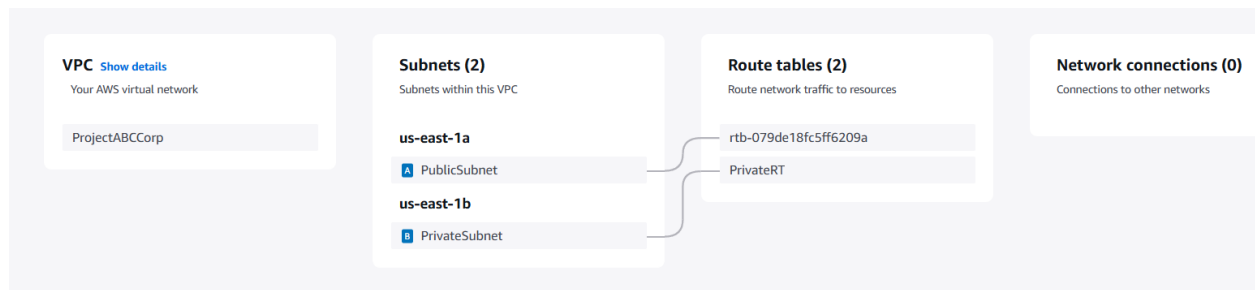
```
[root@rhelclient home]# aws ec2 associate-route-table --route-table-id rtb-0b4fa860bf0817f9c --subnet-id subnet-0a87ace1177174f0e
{
  "AssociationId": "rtbassoc-03fc364b4990afb00",
  "AssociationState": {
    "State": "associated"
  }
}
[root@rhelclient home]#
```

Where: RHEL shell

Why: Assigns public subnet to default RT and private to NAT-enabled RT

Impact: Private EC2s route through NAT only

On AWS console under Resource Map the diagram should look like this



4. Create and Attach Internet Gateway

Create IGW:

```
aws ec2 create-internet-gateway --tag-specifications 'ResourceType=internet-gateway,Tags=[{Key=Name,Value=MyIGW}]'
```

```
[root@rhelclient home]# aws ec2 create-internet-gateway --tag-specifications "ResourceType=internet-gateway,Tags=[{Key=Name,Value=MyIGW}]"
{
  "InternetGateway": {
    "Attachments": [],
    "InternetGatewayId": "igw-024eb9e7f96e6f9d7",
    "OwnerId": "551899295811",
    "Tags": [
      {
        "Key": "Name",
        "Value": "MyIGW"
      }
    ]
  }
}
[root@rhelclient home]#
```

Attach IGW:

aws ec2 attach-internet-gateway --vpc-id <vpc-id> --internet-gateway-id <igw-id>

```
[root@rhelclient home]# aws ec2 attach-internet-gateway --vpc-id vpc-02870049f6b4e3df8 --internet-gateway-id igw-024eb9e7f96e6f9d7
[root@rhelclient home]#
```

Where: CLI

Why: Needed for outbound internet access from public subnet

Impact: Required for NAT gateway and public IP traffic

5. Create NAT Gateway

Allocate Elastic IP:

aws ec2 allocate-address --domain vpc

```
[root@rhelclient home]# aws ec2 allocate-address --domain vpc
{
  "AllocationId": "eipalloc-08fda723eaea45b9e",
  "PublicIpv4Pool": "amazon",
  "NetworkBorderGroup": "us-east-1",
  "Domain": "vpc",
  "PublicIp": "3.223.8.252"
}
[root@rhelclient home]#
```

Create NAT Gateway:

```
aws ec2 create-nat-gateway --subnet-id <public-subnet-id> --allocation-id  
<eip-alloc-id> --tag-specifications  
'ResourceType=natgateway,Tags=[{Key=Name,Value=MyNAT}]'
```

```
[root@rhelclient home]# aws ec2 create-nat-gateway --subnet-id subnet-0f33c41f2e7a0809 --allocation-id eipalloc-08fda723eaea45b9e --tag-specifications 'ResourceType=natgateway,Tags=[{Key=Name,Value=MyNAT}]'  
{  
  "ClientToken": "cfda44ba-bee9-49dc-bc8c-c5ebc7f96038",  
  "NatGateway": {  
    "CreateTime": "2025-04-13T09:53:49+00:00",  
    "NatGatewayAddresses": [ {  
      "AllocationId": "eipalloc-08fda723eaea45b9e",  
      "IsPrimary": true,  
      "Status": "associating"  
    } ],  
    "NatGatewayId": "nat-00019357380fec675",  
    "State": "pending",  
    "SubnetId": "subnet-0f33c41f2e7a0809",  
    "VpcId": "vpc-02870049f6bde3df8",  
    "Tags": [ {  
      "Key": "Name",  
      "Value": "MyNAT"  
    } ],  
    "ConnectivityType": "public"  
  }  
}  
[root@rhelclient home]#
```

Where: CLI

Why: Enables private subnet to reach internet without public IP

Impact: Private EC2s can download packages, update OS, etc.

6. Update Route Tables

Route for Public RT:

```
aws ec2 create-route --route-table-id <public-rt-id> --destination-cidr-block  
0.0.0.0/0 --gateway-id <igw-id>
```

```
[root@rhelclient home]# aws ec2 create-route --route-table-id rtb-079de18fc5ff6209a --destination-cidr-block 0.0.0.0/0 --gateway-id igw-024eb9e7f96e6f9d7  
{  
  "Return": true  
}  
[root@rhelclient home]#
```


Route for Private RT:

```
aws ec2 create-route --route-table-id <private-rt-id> --destination-cidr-block 0.0.0.0/0 --nat-gateway-id <nat-id>
```

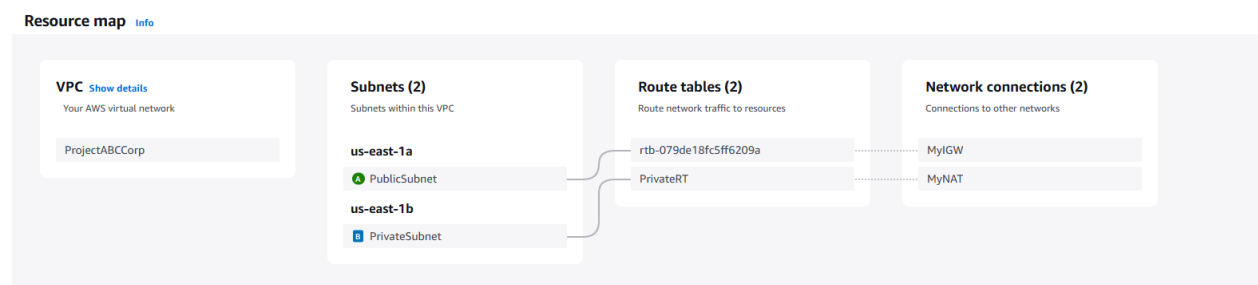
```
[root@rhelclient home]# aws ec2 create-route --route-table-id rtb-0b4fa860bf0817f9c --destination-cidr-block 0.0.0.0/0 --nat-gateway-id nat-00619357386fec675
{
  "Return": true
}
[root@rhelclient home]#
```

Where: CLI

Why: Directs traffic from subnets to the correct egress point

Impact: Critical for network flow and internet access

On AWS console the Resource Map under ProjectABCCorp should look like this



7. Launch and Configure EC2 Instances with SSH Host

=> 7.1 Create an SSH Key Pair for EC2 Access

aws ec2 create-key-pair --key-name MyNewKeyPair --query "KeyMaterial" --output text > MyNewKeyPair.pem && chmod 400 MyNewKeyPair.pem

```
[root@rhelclient home]# aws ec2 create-key-pair --key-name MyNewKeyPair --query "KeyMaterial" --output text > MyNewKeyPair.pem && chmod 400 MyNewKeyPair.pem
[root@rhelclient home]#
```

=> 7.2 Launch Private EC2 Instances (No Public IP, Private Subnet)

Private Instance 1

aws ec2 run-instances --image-id ami-084568db4383264d4 --count 1 --instance-type t2.micro --key-name MyNewKeyPair --subnet-id <private-subnet-id> --no-associate-public-ip-address --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=PrivateInstance1}]'

```
[root@rhelclient home]# aws ec2 run-instances --image-id ami-084568db4383264d4 --count 1 --instance-type t2.micro --key-name MyNewKeyPair --subnet-id subnet-0a87ace1177174f0e --no-associate-public-ip-address --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=PrivateInstance1}]'
{
  "ReservationId": "r-0122d197126b8dce1",
  "OwnerId": "551809295811",
  "Groups": [],
  "Instances": [
    {
      "Architecture": "x86_64",
      "BlockDeviceMappings": [],
      "ClientToken": "96c807e9-7cc2-498c-b55b-6d09ebf25709",
      "EbsOptimized": false,
      "EnaSupport": true,
      "Hypervisor": "xen",
      "NetworkInterfaces": [
        {
          "Attachment": {
            "AttachTime": "2025-04-11T10:28:59+00:00",
            "AttachmentId": "eni-attach-0737543419f14d8af",
            "DeleteOnTermination": true,
            "DeviceIndex": 0,
            "Status": "attaching",
            "NetworkCardIndex": 0
          },
          "Description": "",
          "Groups": [
            {
              "GroupId": "sg-029f6320d688ebd0e",
              "GroupName": "default"
            }
          ],
          "Ipv6Addresses": [],
          "MacAddress": "0e:3c:9c:19:ad:a1",
          "NetworkInterfaceId": "eni-01735d73a7732cb09",
          "OwnerId": "551809295811",
          "PrivateIpAddress": "10.0.2.156",
          "PrivateIpAddresses": [
            {
              "Primary": true,
              "PrivateIpAddress": "10.0.2.156"
            }
          ],
          "SourceDestCheck": true,
          "Status": "in-use",
          "SubnetId": "subnet-0a87ace1177174f0e",
          "VpcId": "vpc-028700d9f6bdc3df8",
          "InterfaceType": "interface",
          "Operator": {
            "Managed": false
          }
        }
      ]
    }
  ]
}
```

Private Instance 2

aws ec2 run-instances --image-id ami-084568db4383264d4 --count 1 --instance-type t2.micro --key-name MyNewKeyPair --subnet-id <private-

```
subnet-id> --no-associate-public-ip-address --tag-specifications  
'ResourceType=instance,Tags=[{Key=Name,Value=PrivateInstance2}]'
```

```
[root@rhelclient home]# aws ec2 run-instances --tags-id ami-004f660b43812644d --count 1 --instance-type t2.micro --key-name MyNewKeyPair --subnet-id subnet-0a07ace1177174f0e --no-associate-public-ip-address  
tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=PrivateInstance2}]'  
{  
  "ReservationId": "r-007aadd8bfb789d0",  
  "OwnerId": "551899295811",  
  "Groups": [],  
  "Instances": [ {  
    "Architecture": "x86_64",  
    "BlockDeviceMappings": [],  
    "ClientToken": "0151a88e-fa45-4968-bada-9cc54c7b2c5b",  
    "EbsOptimized": false,  
    "EnaSupport": true,  
    "Hypervisor": "xen",  
    "NetworkInterfaces": [ {  
      "Attachment": {  
        "AttachTime": "2025-04-13T10:32:46+00:00",  
        "AttachmentId": "eni-attach-01bae44ae62fc5044",  
        "DeleteOnTermination": true,  
        "DeviceIndex": 0,  
        "Status": "attaching",  
        "NetworkCardIndex": 0  
      },  
      "Description": "",  
      "Groups": [ {  
        "GroupId": "sg-029f6320d688ebd6e",  
        "GroupName": "default"  
      } ]  
    },  
    "IpAddresses": [],  
    "NetworkInterfaces": [ {  
      "NetworkInterfaceId": "eni-0742808cfc238d4e",  
      "OwnerId": "551899295811",  
      "PrivateIpAddress": "10.0.2.50",  
      "PrivateIpAddresses": [ {  
        "Primary": true,  
        "PrivateIpAddress": "10.0.2.50"  
      } ]  
    },  
    "SourceDestCheck": true,  
    "Status": "in-use",  
    "SubnetId": "subnet-0a07ace1177174f0e",  
    "VpcId": "vpc-02870049f6bde3d48",  
    "InterfaceType": "interface",  
    "Operation": {  
      "Managed": false  
    }  
  } ]  
}
```

=> 7.3 Fetch Security Group IDs of Private Instances

```
aws ec2 describe-instances --filters
```

```
"Name=tag:Name,Values=PrivateInstance1,PrivateInstance2" --query
```

```
"Reservations[].Instances[].SecurityGroups[].GroupId" --output text
```

```
[root@rhelclient home]# aws ec2 describe-instances --filters "Name=tag:Name,Values=PrivateInstance1,PrivateInstance2" --query "Reservations[].Instances[].SecurityGroups[].GroupId" --output text  
sg-029f6320d688ebd6e sg-029f6320d688ebd6e  
[root@rhelclient home]#
```

=> 7.4 Create Security Group for SSH Host

aws ec2 create-security-group --group-name SSHHostSG --description
"Security group for SSH Host" --vpc-id <vpc-id>

```
[root@rhelclient home]# aws ec2 create-security-group --group-name SSHHostSG --description "Security group for SSH Host" --vpc-id vpc-02870049f6b4e3df8
{
  "GroupId": "sg-01832c06af2582c79",
  "SecurityGroupArn": "arn:aws:ec2:us-east-1:551899295811:security-group/sg-01832c06af2582c79"
}
[root@rhelclient home]#
```

=> 7.5 Allow SSH to SSH Host from Your Public IP

aws ec2 authorize-security-group-ingress --group-id <sshhost-sg-id> --
protocol tcp --port 22 --cidr <your-public-ip>/32

```
[root@rhelclient home]# aws ec2 authorize-security-group-ingress --group-id sg-01832c06af2582c79 --protocol tcp --port 22 --cidr [REDACTED]/32
{
  "Return": true,
  "SecurityGroupRules": [
    {
      "SecurityGroupRuleId": "sgr-0892455843512d658",
      "GroupId": "sg-01832c06af2582c79",
      "GroupOwnerId": "551899295811",
      "IsEgress": false,
      "IpProtocol": "tcp",
      "FromPort": 22,
      "ToPort": 22,
      "CidrIpv4": "[REDACTED]/32",
      "SecurityGroupRuleArn": "arn:aws:ec2:us-east-1:551899295811:security-group-rule/sgr-0892455843512d658"
    }
  ]
}
[root@rhelclient home]#
```

=> 7.6 Allow SSH from SSH Host to Private Instances

aws ec2 authorize-security-group-ingress --group-id <private-sg-id> --
protocol tcp --port 22 --source-group <sshhost-sg-id>

```
[root@rhelclient home]# aws ec2 authorize-security-group-ingress --group-id sg-029f6320d688ebd6e --protocol tcp --port 22 --source-group sg-01832c06af2582c79
{
  "Return": true,
  "SecurityGroupRules": [
    {
      "SecurityGroupRuleId": "sgr-0aa4cbea60f96953e",
      "GroupId": "sg-029f6320d688ebd6e",
      "GroupOwnerId": "551899295811",
      "IsEgress": false,
      "IpProtocol": "tcp",
      "FromPort": 22,
      "ToPort": 22,
      "ReferencedGroupInfo": {
        "GroupId": "sg-01832c06af2582c79",
        "UserId": "551899295811"
      },
      "SecurityGroupRuleArn": "arn:aws:ec2:us-east-1:551899295811:security-group-rule/sgr-0aa4cbea60f96953e"
    }
  ]
}
[root@rhelclient home]#
```

=> 7.7 Launch SSH Host EC2 (Public Subnet with Public IP)

```
aws ec2 run-instances --image-id ami-084568db4383264d4 --count 1 --
instance-type t2.micro --key-name MyNewKeyPair --subnet-id <public-
subnet-id> --associate-public-ip-address --security-group-ids <sshhost-sg-
id> --tag-specifications
'ResourceType=instance,Tags=[{Key=Name,Value=SSHHost}]'
```

```
[root@rhelclient home]# aws ec2 run-instances --image-id ami-084568db4383264d4 --count 1 --instance-type t2.micro --key-name MyNewKeyPair --subnet-id subnet-0f331c41f2e7a8809 --associate-public-ip-address --sec
urity-group-ids sg-01832c06af2582c79 --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=SSHHost}]'
{
  "ReservationId": "r-071cf64cf2dd4534",
  "OwnerId": "551899295811",
  "Groups": [],
  "Instances": [
    {
      "Architecture": "x86_64",
      "BlockDeviceMappings": [],
      "ClientToken": "034a1440-790c-4a25-af07-5005f4aba54c",
      "EbsOptimized": false,
      "EnaSupport": true,
      "Hypervisor": "xen",
      "NetworkInterfaces": [
        {
          "Attachment": {
            "AttachTime": "2025-04-13T10:46:35+00:00",
            "AttachmentId": "eni-attach-0f7553bcd7b00546",
            "DeleteOnTermination": true,
            "DeviceIndex": 0,
            "Status": "attaching",
            "NetworkCardIndex": 0
          },
          "Description": "",
          "Groups": [
            {
              "GroupId": "sg-01832c06af2582c79",
              "GroupName": "SSHHostSG"
            }
          ],
          "Ipv6Addresses": [],
          "MacAddress": "0a:ff:e3:07:11:2f",
          "NetworkInterfaceId": "eni-095f73d8590de3bd5",
          "OwnerId": "551899295811",
          "PrivateIpAddress": "10.0.1.78",
          "PrivateIpAddresses": [
            {
              "Primary": true,
              "PrivateIpAddress": "10.0.1.78"
            }
          ],
          "SourceDestCheck": true,
          "Status": "in-use",
          "SubnetId": "subnet-0f331c41f2e7a8809",
          "Type": "t2c-020700d0f0b4e3d18",
          "InterfaceType": "interface",
          "Operator": {
            "Managed": false
          }
        }
      ]
    }
  ]
}
```

=> 7.8 Fetch Instance IDs of All Created EC2s

aws ec2 describe-instances --filters

"Name=tag:Name,Values=PrivateInstance1,PrivateInstance2,SSHHost" --
query "Reservations[].Instances[].InstanceId" --output text

```
[root@rhelclient home]# aws ec2 describe-instances --filters "Name=tag:Name,Values=PrivateInstance1,PrivateInstance2,SSHHost" --query "Reservations[].Instances[].InstanceId" --output text
i-00ad8116af9e93d14    i-096e2a09b2f8a0a2e    i-04fe370269772c283
[root@rhelclient home]#
```

=> 7.9 Fetch Public IP of SSH Host

aws ec2 describe-instances --filters "Name=tag:Name,Values=SSHHost" --
query "Reservations[].Instances[].PublicIpAddress" --output text

```
[root@rhelclient home]# aws ec2 describe-instances --filters "Name=tag:Name,Values=SSHHost" --query "Reservations[].Instances[].PublicIpAddress" --output text
98.81.229.221
[root@rhelclient home]#
```

=> 7.10 Fetch Private IPs of the Private Instances

aws ec2 describe-instances --filters

"Name=tag:Name,Values=PrivateInstance1,PrivateInstance2" --query
'Reservations[].Instances[].PrivateIpAddress' --output text

```
[root@rhelclient home]# aws ec2 describe-instances --filters "Name=tag:Name,Values=PrivateInstance1,PrivateInstance2" --query "Reservations[].Instances[].PrivateIpAddress" --output text
10.0.2.50    10.0.2.156
[root@rhelclient home]#
```

=> 7.11 Transfer SSH Key to SSH Host

```
scp -i /home/MyNewKeyPair.pem /home/MyNewKeyPair.pem  
ubuntu@<sshhost-public-ip>:~/
```

```
[root@rhelclient home]# ls  
ndain  MyNewKeyPair.pem  prashant  
[root@rhelclient home]# scp -i /home/MyNewKeyPair.pem /home/MyNewKeyPair.pem ubuntu@98.81.229.221:~/  
MyNewKeyPair.pem 100% 1679 3.5KB/s 00:00  
[root@rhelclient home]#
```

Note: Since the present working directory for this project on RHEL Shell was /home so please make sure that you create the key in /home directory

=> 7.12 SSH Into SSH Host

```
ssh -i "MyNewKeyPair.pem" ubuntu@<sshhost-public-ip>
```

```
[root@rhelclient home]# ssh -i "MyNewKeyPair.pem" ubuntu@98.81.229.221
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 Sun Apr 13 10:58:34 UTC 2025

System load:  0.01               Processes:            105
Usage of /:   25.3% of 6.71GB    Users logged in:     0
Memory usage: 20%               IPv4 address for enX0: 10.0.1.78
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-10-0-1-78:~$
```

=> 7.13 SSH Into PrivateInstance1 from SSH Host

ssh -i "MyNewKeyPair.pem" ubuntu@<private-instance-1-ip>


```
ubuntu@ip-10-0-1-78:~$ ssh -i "MyNewKeyPair.pem" ubuntu@10.0.2.50
The authenticity of host '10.0.2.50 (10.0.2.50)' can't be established.
ED25519 key fingerprint is SHA256:b4cdiK4MQkZj6WDCYehMvF1FMCaAJti0m4igjvSzdCs
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.0.2.50' (ED25519) to the list of known hosts.
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 Sun Apr 13 11:00:20 UTC 2025

System load:  0.0               Processes:            103
Usage of /:   25.0% of 6.71GB   Users logged in:     0
Memory usage: 20%              IPv4 address for enx0: 10.0.2.50
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

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

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

ubuntu@ip-10-0-2-50:~$
```

=> 7.14 Install NGINX on PrivateInstance1

sudo apt-get update && sudo apt-get upgrade -y && sudo apt install nginx -y

```
ubuntu@ip-10-0-2-50:~$ sudo apt-get update && sudo apt-get upgrade -y && sudo apt install nginx -y
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [741 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [991 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [219 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [151 kB]
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 c-n-f Metadata [13.5 kB]
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [1052 kB]
Get:19 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [265 kB]
Get:20 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [367 kB]
Get:21 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [26.0 kB]
Get:22 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [892 kB]
Get:23 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [182 kB]
Get:24 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 B]
Get:25 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 c-n-f Metadata [492 B]
Get:26 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [21.5 kB]
Get:27 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [4788 B]
Get:28 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [940 B]
Get:29 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [592 B]
Get:30 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Packages [39.1 kB]
Get:31 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main Translation-en [8676 B]
Get:32 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [7064 B]
Get:33 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 c-n-f Metadata [272 B]
Get:34 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [26.4 kB]
Get:35 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [16.3 kB]
Get:36 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [15.8 kB]
Get:37 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [1304 B]
Get:38 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:39 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [116 B]
Get:40 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:41 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 B]
Get:42 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [142 kB]
Get:43 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [8956 B]
Get:44 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [7068 B]
Get:45 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [829 kB]
Get:46 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [180 kB]
88% [8 Components-amd64 store 0 B] [46 Translation-en 1131 B/180 kB 1%]
```

=> 7.15 Edit NGINX Landing Page

```
sudo nano /var/www/html/index.html
```

Paste your HTML and save using:

CTRL+O → Enter → CTRL+X

```

<!-- main.js -->
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>ABC Corp - Cloud Computing Solutions</title>
  <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500;600;700&display=swap" rel="stylesheet">
  <style>
    *{
      margin: 0;
      padding: 0;
      box-sizing: border-box;
      font-family: 'Poppins', sans-serif;
    }

    :root {
      --primary-color: #2e3e50;
      --secondary-color: #d9d9db;
      --accent-color: #e0f4fc;
      --text-color: #333;
      --light-bg: #f8f9fa;
    }

    body {
      line-height: 1.6;
      color: var(--text-color);
    }

    .navbar {
      background: var(--primary-color);
      padding: 1rem 5%;
      position: fixed;
      width: 100%;
      z-index: 1000;
      box-shadow: 0 2px 10px rgba(0,0,0,0.1);
    }

    .navbar .logo {
      color: white;
      font-size: 1.5rem;
      font-weight: 700;
      text-decoration: none;
    }

```

=> 7.16 Validate NGINX Config

```
sudo nginx -t
```

```
ubuntu@ip-10-0-2-50:~$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
ubuntu@ip-10-0-2-50:~$
```

=> 7.17 Restart NGINX

```
sudo systemctl restart nginx
```

```
ubuntu@ip-10-0-2-50:~$ sudo systemctl restart nginx
ubuntu@ip-10-0-2-50:~$
```

=> 7.18 Verify HTTP Listening (No net-tools)

`sudo ss -tuln | grep 80 && sudo nginx -T | grep "listen 80"`

```
ubuntu@ip-10-0-2-50:~$ sudo ss -tuln | grep 80 && sudo nginx -T | grep "listen 80"
tcp    LISTEN 0      511      0.0.0.0:80      0.0.0.0:*
tcp    LISTEN 0      511      [::]:80        [::]:*
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
      listen 80 default_server;
#      listen 80;
ubuntu@ip-10-0-2-50:~$
```

7.19 Repeat 7.13 to 7.18 for PrivateInstance2

Why:

- Installing NGINX allows serving HTTP traffic.
- Replacing the default HTML lets you identify which backend served the request.
- `nginx -t` ensures the configuration is valid.
- `ss` confirms NGINX is listening (alternative to `netstat`).

Impact:

- Confirms the instance is correctly serving the site.

- Ensures NGINX is up, content is in place, and the app is ready for ALB forwarding.

8. Create Target Group and Internal ALB

Create TG:

aws elbv2 create-target-group --name MyTG --protocol HTTP --port 80 --vpc-id <vpc-id> --target-type instance

```
[root@rhelclient home]# aws elbv2 create-target-group --name MyTG --protocol HTTP --port 80 --vpc-id vpc-02870049f6b4e3df8 --target-type instance
{
  "TargetGroups": [
    {
      "TargetGroupArn": "arn:aws:elasticloadbalancing:us-east-1:551899295811:targetgroup/MyTG/7e89dca7ddb793ec",
      "TargetGroupName": "MyTG",
      "Protocol": "HTTP",
      "Port": 80,
      "VpcId": "vpc-02870049f6b4e3df8",
      "HealthCheckProtocol": "HTTP",
      "HealthCheckPort": "traffic-port",
      "HealthCheckEnabled": true,
      "HealthCheckIntervalSeconds": 30,
      "HealthCheckTimeoutSeconds": 5,
      "HealthyThresholdCount": 5,
      "UnhealthyThresholdCount": 2,
      "HealthCheckPath": "/",
      "Matcher": {
        "HttpCode": "200"
      },
      "TargetType": "instance",
      "ProtocolVersion": "HTTP1",
      "IpAddressType": "ipv4"
    }
  ]
}
```

Copy the target group arn from the output

Register EC2 Instances:

aws elbv2 register-targets --target-group-arn <tg-arn> --targets

Id=<PrivateInstnace1-id>

aws elbv2 register-targets --target-group-arn <tg-arn> --targets

Id=<PrivateInstance2-id>

```
[root@rhelclient home]# aws elbv2 register-targets --target-group-arn arn:aws:elasticloadbalancing:us-east-1:551899295811:targetgroup/MyTG/7e89dca7ddb793ec --targets Id-1-04fe370269772c283
aws elbv2 register-targets --target-group-arn arn:aws:elasticloadbalancing:us-east-1:551899295811:targetgroup/MyTG/7e89dca7ddb793ec --targets Id-1-096e2a09b2f8a0a2e
[root@rhelclient home]#
```

Verify

aws elbv2 describe-target-health --target-group-arn <target-group-arn-id>

```
[root@rhelclient home]# aws elbv2 describe-target-health --target-group-arn arn:aws:elasticloadbalancing:us-east-1:551899295811:targetgroup/MyIG/7e89dca7ddb793ec
{
  "TargetHealthDescriptions": [
    {
      "Target": {
        "Id": "i-096e2a09b2f8a0a2e",
        "Port": 80
      },
      "HealthCheckPort": "80",
      "TargetHealth": {
        "State": "unused",
        "Reason": "Target.NotInUse",
        "Description": "Target group is not configured to receive traffic from the load balancer"
      }
    },
    {
      "Target": {
        "Id": "i-04fe370269772c283",
        "Port": 80
      },
      "HealthCheckPort": "80",
      "TargetHealth": {
        "State": "unused",
        "Reason": "Target.NotInUse",
        "Description": "Target group is not configured to receive traffic from the load balancer"
      }
    }
  ]
}
[root@rhelclient home]#
```

Fetch Security Group IDs for Public and Private Subnet using their IDs

aws ec2 describe-instances --filters "Name=subnet-id,Values=subnetid" --
query "Reservations[].Instances[].SecurityGroups[].GroupId" --output text

```
[root@rhelclient home]# aws ec2 describe-instances --filters "Name=subnet-id,Values=subnet-0f331c41f2e7a8069" --query "Reservations[].Instances[].SecurityGroups[].GroupId" --output text
sg-01832c06af2582c79
[root@rhelclient home]# aws ec2 describe-instances --filters "Name=subnet-id,Values=subnet-0a87ace1177174f0e" --query "Reservations[].Instances[].SecurityGroups[].GroupId" --output text
sg-029f6320d688ebd6e sg-029f6320d688ebd6e
[root@rhelclient home]#
```

Create ALB:

aws elbv2 create-load-balancer --name my-alb --subnets <public-subnet-id>
<private-subnet-id> --security-groups <public-subnet-sg-id> <private-subnet-
sg-id> --scheme internal --type application

```
[root@helliclient home]# aws elbv2 create-load-balancer --name my-alb --subnets subnet-0f331c41f2e7a8809 subnet-0a87ace1177174f0e --security-groups sg-01832c06af2582c79 sg-029f6320d688ebd6e --scheme internal --type application
{
  "LoadBalancers": [
    {
      "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-east-1:551899295811:loadbalancer/app/my-alb/abc725ce83c36de",
      "DNSName": "internal-my-alb-5936426.us-east-1.elb.amazonaws.com",
      "CanonicalHostedZoneId": "Z355LD01MQZX7A",
      "CreatedTime": "2025-04-13T11:25:19.118000+00:00",
      "LoadBalancerName": "my-alb",
      "Scheme": "internal",
      "VpcId": "vpc-02870049f6bde3df8",
      "State": {
        "Code": "provisioning"
      },
      "Type": "application",
      "AvailabilityZones": [
        {
          "ZoneName": "us-east-1a",
          "SubnetId": "subnet-0f331c41f2e7a8809",
          "LoadBalancerAddresses": [ ]
        },
        {
          "ZoneName": "us-east-1b",
          "SubnetId": "subnet-0a87ace1177174f0e",
          "LoadBalancerAddresses": [ ]
        }
      ],
      "SecurityGroups": [
        "sg-01832c06af2582c79",
        "sg-029f6320d688ebd6e"
      ],
      "IpAddressType": "ipv4"
    }
  ]
}
[root@helliclient home]#
```

Copy the Loadbalancer arn from the output

Create Listener:

aws elbv2 create-listener --load-balancer-arn <alb-arn> --protocol HTTP --port 80 --default-actions Type=forward,TargetGroupArn=<tg-arn>

```
[root@helliclient home]# aws elbv2 create-listener --load-balancer-arn arn:aws:elasticloadbalancing:us-east-1:551899295811:loadbalancer/app/my-alb/abc725ce83c36de --protocol HTTP --port 80 --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-east-1:551899295811:targetgroup/MyTG/7e89dca7ddb793ec
{
  "Listeners": [
    {
      "ListenerArn": "arn:aws:elasticloadbalancing:us-east-1:551899295811:listener/app/my-alb/abc725ce83c36de/37e2daa3252170de",
      "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-east-1:551899295811:loadbalancer/app/my-alb/abc725ce83c36de",
      "Port": 80,
      "Protocol": "HTTP",
      "DefaultActions": [
        {
          "Type": "Forward",
          "TargetGroupArn": "arn:aws:elasticloadbalancing:us-east-1:551899295811:targetgroup/MyTG/7e89dca7ddb793ec",
          "ForwardConfig": {
            "TargetGroups": [
              {
                "TargetGroupArn": "arn:aws:elasticloadbalancing:us-east-1:551899295811:targetgroup/MyTG/7e89dca7ddb793ec",
                "Weight": 1
              }
            ],
            "TargetGroupStickinessConfig": {
              "Enabled": false
            }
          }
        }
      ]
    }
  ]
}
[root@helliclient home]#
```

Copy Listener ARN from the output

Health check for attached Instances

aws elbv2 describe-target-health --target-group-arn <target-group-arn>

```
[root@rhelclient home]# aws elbv2 describe-target-health --target-group-arn arn:aws:elasticloadbalancing:us-east-1:551899295811:targetgroup/MyTG/7e89dca7ddb793ec
{
  "TargetHealthDescriptions": [
    {
      "Target": {
        "Id": "i-096e2a09b2f8a0a2e",
        "Port": 80
      },
      "HealthCheckPort": "80",
      "TargetHealth": {
        "State": "healthy"
      },
      "AdministrativeOverride": {
        "State": "no_override",
        "Reason": "AdministrativeOverride.NoOverride",
        "Description": "No override is currently active on target"
      }
    },
    {
      "Target": {
        "Id": "i-04fe370269772c283",
        "Port": 80
      },
      "HealthCheckPort": "80",
      "TargetHealth": {
        "State": "healthy"
      },
      "AdministrativeOverride": {
        "State": "no_override",
        "Reason": "AdministrativeOverride.NoOverride",
        "Description": "No override is currently active on target"
      }
    }
  ]
}
[root@rhelclient home]#
```

Where: CLI

Why: ALB distributes HTTP traffic across EC2s

Impact: Enables high availability

9. Public Access via API Gateway

Create VPC Link:

```
aws apigatewayv2 create-vpc-link --name "PrivateALBLink" --subnet-ids
<public-subnet-id> <private-subnet-id> --security-group-ids <public-subnet-
sg-id> <private-subnet-sg-id>
```



```
[root@rhelclient home]# aws apigatewayv2 create-vpc-link --name "PrivateALBLink" --subnet-ids subnet-0f331c41f2e7a0809 subnet-0a87ace1177174f8e --security-group-ids sg-01812c06af2582c79 sg-029f632bd088ebd8e
{
  "CreateDate": "2025-04-13T12:00:33+00:00",
  "Name": "PrivateALBLink",
  "SecurityGroupId": [
    "sg-01812c06af2582c79",
    "sg-029f632bd088ebd8e"
  ],
  "SubnetIds": [
    "subnet-0f331c41f2e7a0809",
    "subnet-0a87ace1177174f8e"
  ],
  "Tags": {},
  "VpcLinkId": "azynan",
  "VpcLinkStatus": "PENDING",
  "VpcLinkStatusMessage": "VPC link is provisioning ENIs",
  "VpcLinkVersion": "v2"
}
[root@rhelclient home]#
```

Copy VPC Link ID from the output

Create API:

aws apigatewayv2 create-api --name "MyAPI" --protocol-type HTTP

```
[root@rhelclient home]# aws apigatewayv2 create-api --name "MyAPI" --protocol-type HTTP
{
  "ApiEndpoint": "https://4sfydyjqk5i.execute-api.us-east-1.amazonaws.com",
  "ApiId": "4sfydyjqk5i",
  "ApiKeySelectionExpression": "$request.header.x-api-key",
  "CreateDate": "2025-04-13T12:04:14+00:00",
  "DisableExecuteApiEndpoint": false,
  "IpAddressType": "ipv4",
  "Name": "MyAPI",
  "ProtocolType": "HTTP",
  "RouteSelectionExpression": "$request.method $request.path"
}
[root@rhelclient home]#
```

Copy API ID from the output

Fetch the DNS Name for Internal Application Load Balancer using the command mentioned below

aws elbv2 describe-load-balancers --names my-alb --query "LoadBalancers[].DNSName" --output text

```
[root@rhelclient home]# aws elbv2 describe-load-balancers --names my-alb --query "LoadBalancers[].DNSName" --output text
internal-my-alb-5936426.us-east-1.elb.amazonaws.com
[root@rhelclient home]#
```

Create Integration:

```
aws apigatewayv2 create-integration --api-id <api-id> --integration-type
HTTP_PROXY --connection-type VPC_LINK --connection-id <vpc-link-id> --
integration-method ANY --integration-uri <alb-listener-arn> --payload-format-
version 1.0
```

```
[root@rhelclient home]# aws apigatewayv2 create-integration --api-id 4sfyjdjk5i --integration-type HTTP_PROXY --connection-type VPC_LINK --connection-id 4zwnan --integration-method ANY --integration-uri arn:aws:elasticloadbalancing:us-east-1:551899295811:listener/app/my-alb/abce725ce83c36de/37e2daa3252179de --payload-format-version 1.0
{
  "ConnectionId": "4zwnan",
  "ConnectionType": "VPC_LINK",
  "IntegrationId": "70pjonn",
  "IntegrationMethod": "ANY",
  "IntegrationType": "HTTP_PROXY",
  "IntegrationUri": "arn:aws:elasticloadbalancing:us-east-1:551899295811:listener/app/my-alb/abce725ce83c36de/37e2daa3252179de",
  "PayloadFormatVersion": "1.0",
  "TimeoutInMillis": 30000
}
[root@rhelclient home]#
```

Copy Integration ID from the output

Create Route:

```
aws apigatewayv2 create-route --api-id <api-id> --route-key "ANY /" --target
integrations/<integration-id>
```

```
[root@rhelclient home]# aws apigatewayv2 create-route --api-id 4sfyjdjk5i --route-key "ANY /" --target integrations/70pjonn
{
  "ApiKeyRequired": false,
  "AuthorizationType": "NONE",
  "RouteId": "8dw7c7q",
  "RouteKey": "ANY /",
  "Target": "integrations/70pjonn"
}
[root@rhelclient home]#
```

Deploy Stage:

```
aws apigatewayv2 create-stage --api-id <api-id> --stage-name prod --auto-
deploy
```

```
[root@rhelclient home]# aws apigatewayv2 create-stage --api-id 4sfydjqk5i --stage-name prod --auto-deploy
{
  "AutoDeploy": true,
  "CreateDate": "2025-04-13T12:18:54+00:00",
  "DefaultRouteSettings": {
    "DetailedMetricsEnabled": false
  },
  "LastUpdatedDate": "2025-04-13T12:18:54+00:00",
  "RouteSettings": {},
  "StageName": "prod",
  "StageVariables": {},
  "Tags": {}
}
[root@rhelclient home]#
```

Get Public URL:

aws apigatewayv2 get-apis --query "Items[?ApiId=='<api-id>'].ApiEndpoint" --
output text

```
[root@rhelclient home]# aws apigatewayv2 get-apis --query "Items[?ApiId=='4sfydjqk5i'].ApiEndpoint" --output text
https://4sfydjqk5i.execute-api.us-east-1.amazonaws.com
[root@rhelclient home]#
```

access the link like this:

<https://4sfydjqk5i.execute-api.us-east-1.amazonaws.com/prod/>

10. Fixing 404 Error



To resolve this error we need to fix the NGINX route
SSH into both of the Instances using the SSH Host

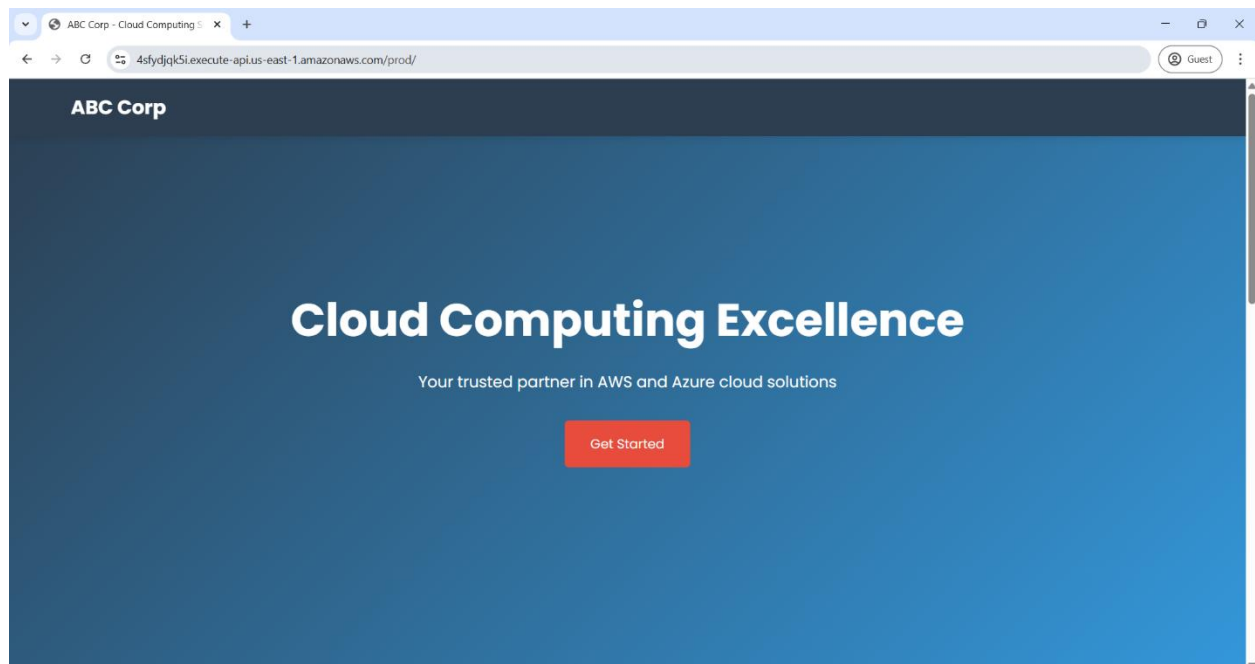
Make sure API Gateway matches NGINX route:

```
sudo mkdir -p /var/www/html/prod && sudo cp /var/www/html/index.html  
/var/www/html/prod/index.html && sudo systemctl restart nginx
```

```
ubuntu@ip-10-0-2-50:~$ sudo mkdir -p /var/www/html/prod && sudo cp /var/www/html/index.html /var/www/html/prod/index.html && sudo systemctl restart nginx  
ubuntu@ip-10-0-2-50:~$
```

```
ubuntu@ip-10-0-2-156:~$ sudo mkdir -p /var/www/html/prod && sudo cp /var/www/html/index.html /var/www/html/prod/index.html && sudo systemctl restart nginx
```

Reload or use the link fetched earlier for API Gateway into a new tab



11. To Simulate Failover

You can test HA by stopping NGINX on one instance:

```
sudo systemctl stop nginx
```

```
ubuntu@ip-10-0-2-50:~$ sudo systemctl stop nginx
ubuntu@ip-10-0-2-50:~$
```

Then hit the API Gateway URL again:

<https://4sfydjqk5i.execute-api.us-east-1.amazonaws.com/prod/>

You should still get a response — served by the second instance.

If you refresh multiple times or constantly you will get the error for 502 Bad Gateway but once you get error refresh the webpage and it should show the index.html stored on nginx

Note: When using **API Gateway → VPC Link → Internal ALB → EC2 Instances (NGINX)**, a **502 Bad Gateway** error can occur **temporarily** in the following scenario:

During Failover

- When **one EC2 instance becomes unreachable** (e.g., NGINX service stopped or instance fails), the **ALB health check** takes a few seconds to **detect** the failure.
- During that window, **API Gateway may route a request to the now-unhealthy instance.**

- Since the backend doesn't respond correctly (or at all), the ALB returns a 502 Bad Gateway, and that error is passed to API Gateway → Client.

Where: CLI

Why: Provide HTTPS access to internal ALB

Impact: Secure public endpoint for external users

Final Outcome

- EC2s are private, secured
- Internal ALB handles backend traffic
- API Gateway exposes public endpoint with HTTPS
- Routing is logically isolated
- All infrastructure built and controlled via CLI