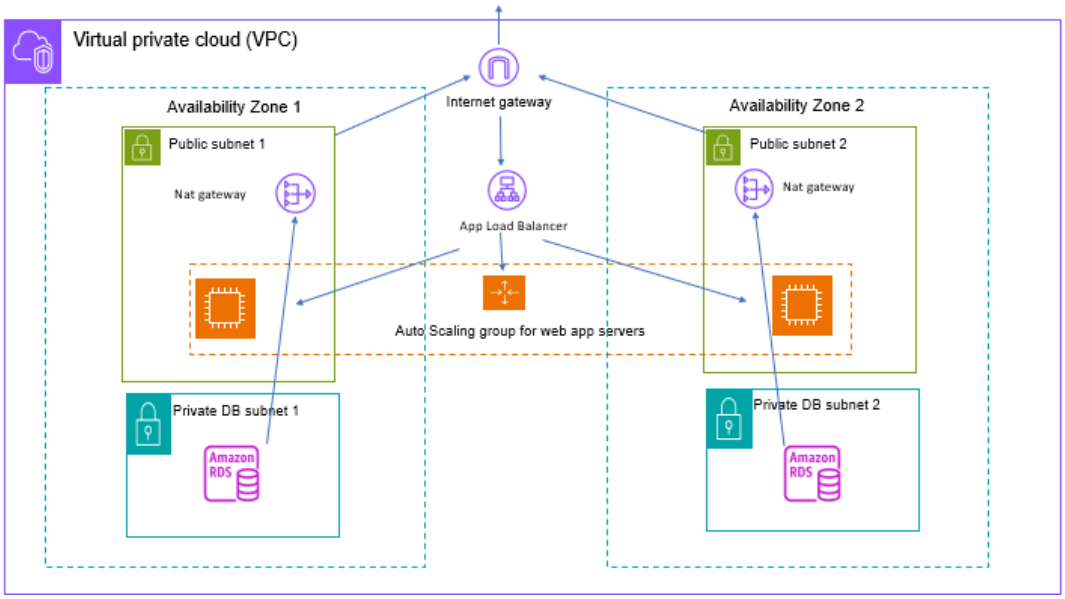
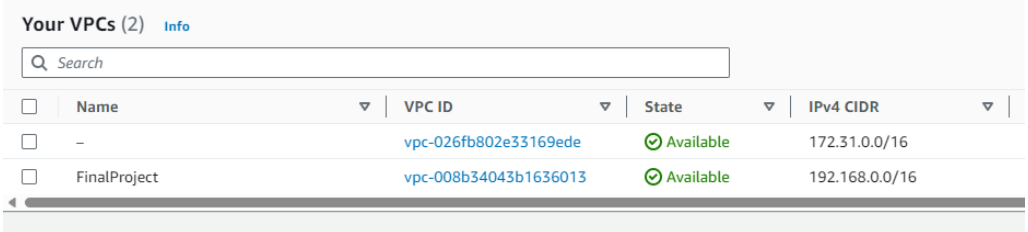
Network Diagram



Challenge: There is no second db in these instructions because our free tier doesn't allow multiAZ replication
Challenge: Writing HTML and PHP files to connect to the database and properly display data
Challenge: Finding out proper PHP and MYSQL dependencies to download for operation of web app
Challenge: S3 bucket implementation due to Learner Lab restrictions of IAM Roles, GitHub is our substitute

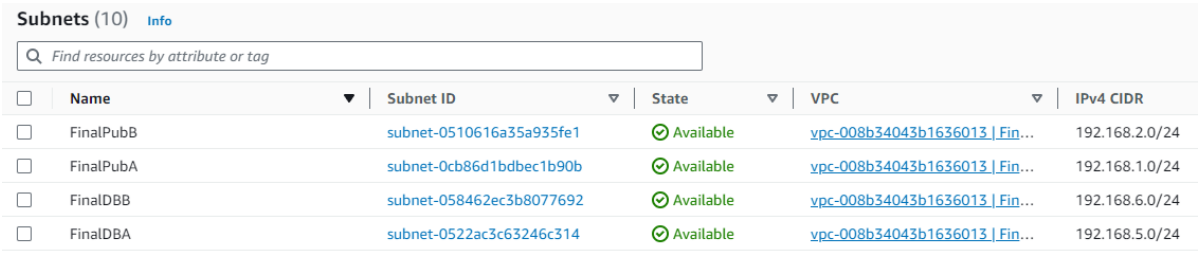
Steps To Setup

1. Go to the **VPC Dashboard** and click **Create VPC**
    a. VPC Only
    b. Name: FinalProject
    c. CIDR block: 192.168.0.0/16
    d. Ipv6 CIDR block: none
    e. Tenancy: Default



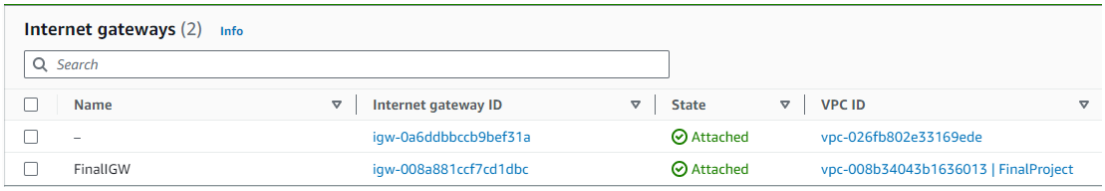
2. Go to **Subnets** and click **Create Subnet**
    a. Choose the **FinalProject** VPC
    b. Create the below subnets
        i. Public
            1. FinalPubA: 192.168.1.0/24 in us-east-1a
            2. FinalPubB: 192.168.2.0/24 in us-east-1b
        ii. Database

1. FinalDBA: 192.168.5.0/24 in us-east-1a
2. FinalDBB: 192.168.6.0/24 in us-east-1b

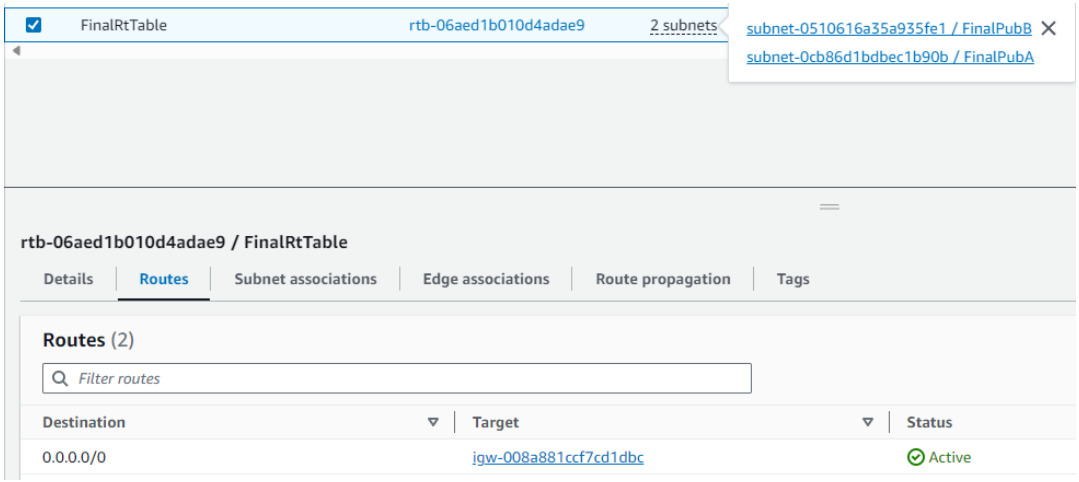


   c. Enable auto-assign public IPv4 for FinalPubA and FinalPubB by selecting the subnets and clicking **Actions > Edit Subnet Settings** and selecting **enable auto-assign ipv4 address**

3. Go to **Internet_Gateways** and click **create the internet gateway**.
   a. Name it **FinalIGW** and create it
   b. Attach it to our FinalProject VPC by clicking **Actions > Attach to VPC** and selecting the FinalProject VPC



4. Go to **Route Tables > Create Route Table**
   a. Name: FinalRtTable
   b. VPC: FinalProject VPC
   c. Once Created, select the created route table and click **Actions > Edit Routes**
      i. Add the below route:
         1. Destination 0.0.0.0/0
         2. Target: Internet Gateway, FinalIGW
   d. Select the route table and go to **Actions > Edit Subnet Associations**
      i. Associate FinalPubA and FinalPubB with FinalRtTable

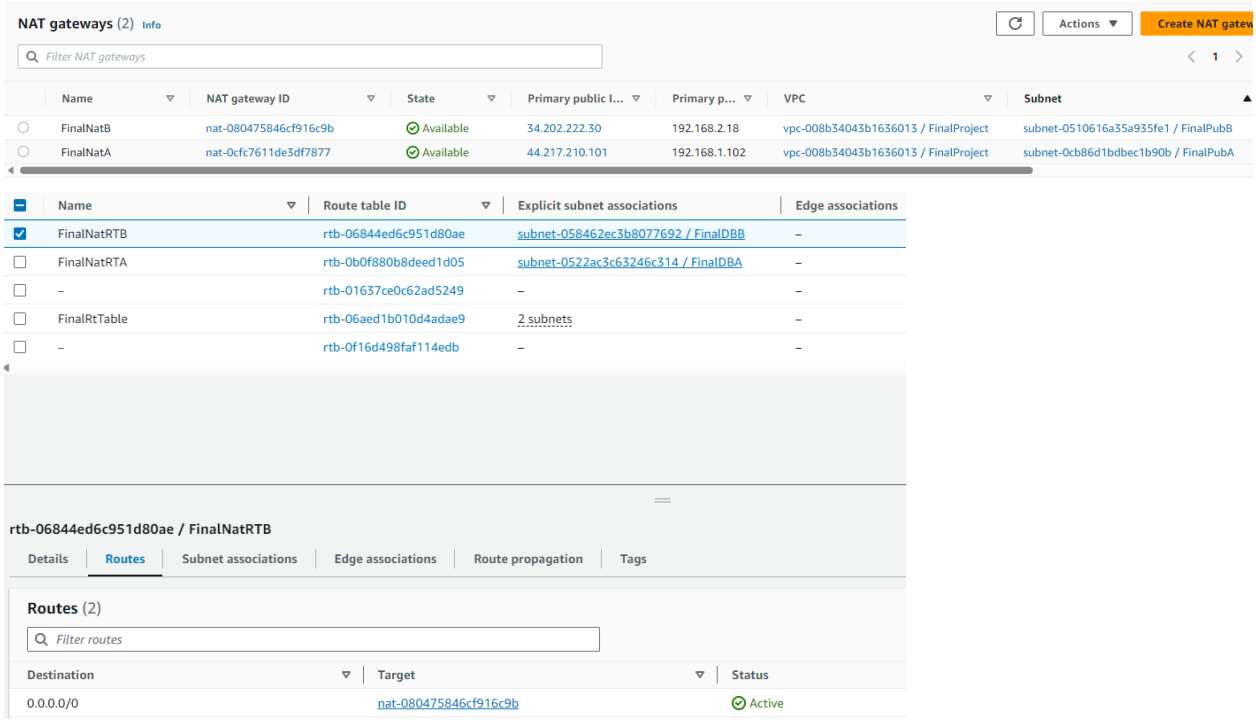


5. Create Nat Gateways and Routes to them
   a. Go to **Nat Gateways > Create NAT Gateway**

i. Name: FinalNATA
ii. Subnet: FinalPubA
iii. Allocate an Elastic IP for the gateway

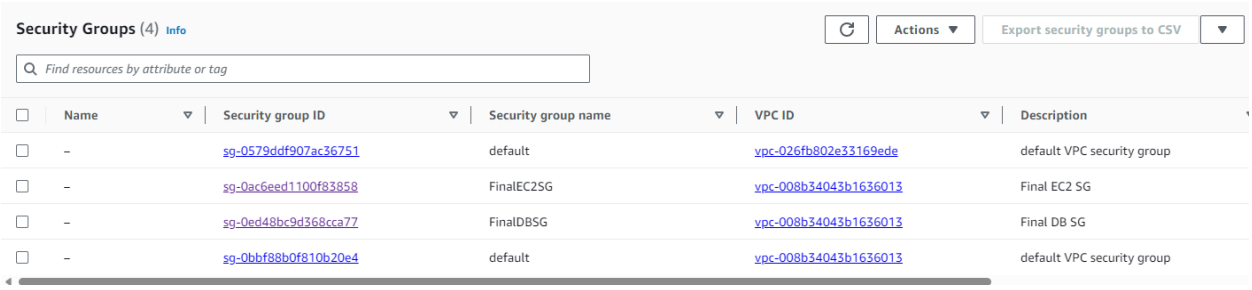
b. Go to **Route Tables > Create Route Table**
i. Name: FinalNATRTA
ii. VPC: FInalProject
iii. Once created, click **Actions > Subnet Associations**
1. Associate it with FinalDBA
iv. Click **Actions > Edit Routes**
1. Destination: 0.0.0.0/0
2. Target: NAT Gateway, FinalNATA

c. Repeat these steps for another NAT Gateway called FinalNATB with the subnet as FinalPubB, and a route table called FinalNATRTB associated with FinalDBB

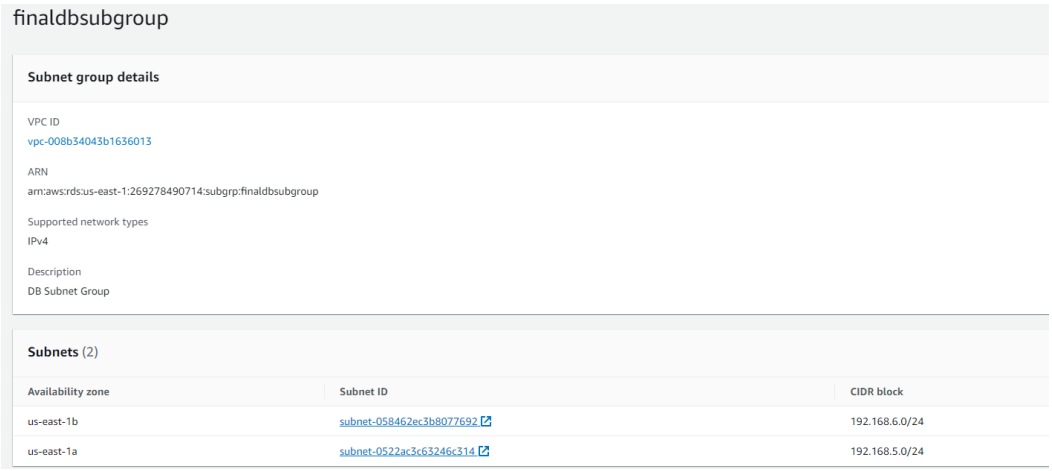


6. Database Setup
a. Go to the **VPC Dashboard > Security Groups**
i. Create security group
1. Name: FinalEC2SG
2. Description : Final EC2 SG
3. VPC: FinalProject
4. Inbound rules
a. Allow ssh from anywhere ipv4
i. Usually, you would only allow this from a bastion host, for this example we will allow from anywhere
b. Allow http from anywhere ipv4
ii. Create security group

1. Name: FinalDBSG
2. Description : Final DB SG
3. VPC: FinalProject
4. Inbound rules
    a. Allow ssh from anywhere ipv4
        i. Usually, you would only allow this from a bastion host, for this example we will allow from anywhere
    b. Allow MYSQL/Aurora from FinalEC2SG security group

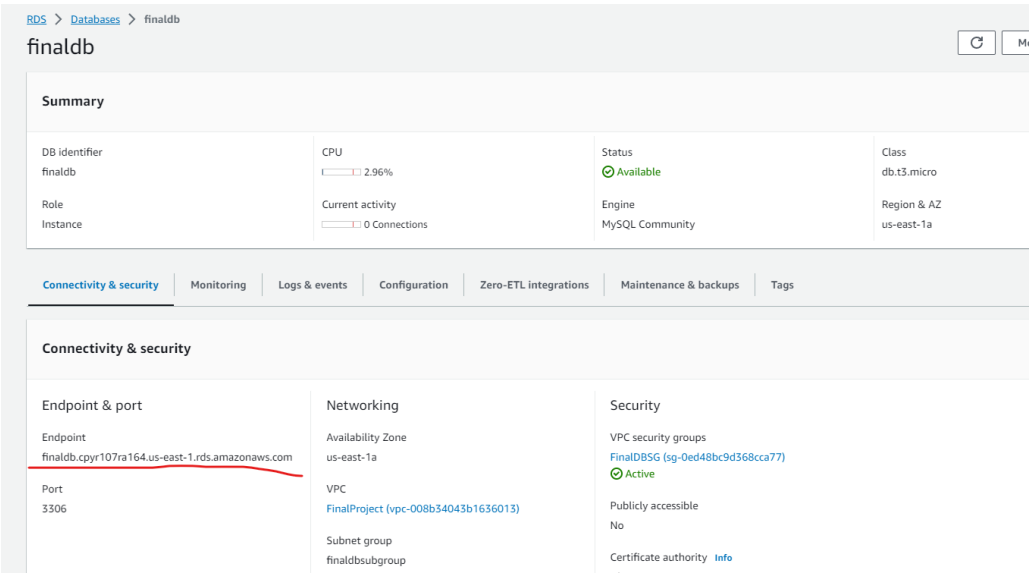


b. Go to **RDS Dashboard > Subnet Groups**
    i. Create DB subnet group
        1. Name: FinalDBSubGroup
        2. Description: DB Subnet Group
        3. VPC: FinalProject
        4. Availability Zones: us-east-1a,us-east-1b
        5. Subnets:
            a. 192.168.5.0/24
            b. 192.168.6.0/24

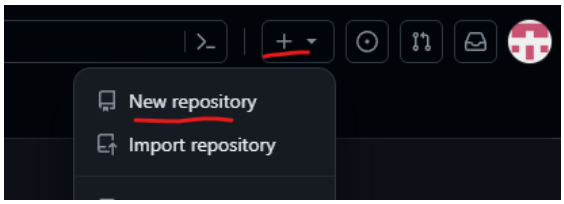


c. Go to **RDS > Databases**
    i. Click Create Database
        1. Creation Method: Standard Create
        2. Engine Type: MySQL
        3. Templates: Free tier
        4. Settings
            a. DB Instance Identifier: FinalDB

b. Credentials
    i. Username: admin
    ii. Password: goldowl77
5. Connectivity
    a. VPC: FinalProject
    b. Subnet Group: FinalDBSubGroup
    c. Existing Security Group: FinalDBSG
    d. Availability zone: us-east-1a
6. Additional Configuration:
    a. Initial Database Name: FinalDBName
7. Create Database and wait until it is created. Then, select the database and copy its Endpoint into a text file to be used later



7. Go to github.com and make a **public** github repository



    a. Add a file to the created repository with a filename of index.html and save it with the following data

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Wonderful Cloud Website</title>
</head>
<body>
```

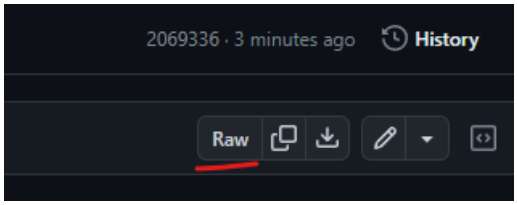
```
<h1>Look at these very awesome movies :D</h1>
<div id="output"></div>
<script>
// Use JavaScript to make an AJAX request to the PHP file
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        // Display the response in the 'output' div
        document.getElementById("output").innerHTML = this.responseText;
    }
};
xhttp.open("GET", "request.php", true);
xhttp.send();
</script>
</body>
</html>
```

b. Add a file to the repository called request.php and save to it the following data. **BE SURE TO REPLACE SERVERNAME WITH THE ENDPOINT OF YOUR RDS DATABASE INSTANCE**

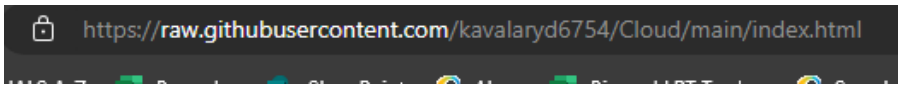
```
<?php
$hostname = gethostname();
$servername = "REPLACE ME";
$username = "admin";
$password = "goldowl77";
$dbname = "FinalDBName";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}
//RDS Movie Table Data
$sql = "SELECT title, genre, director, release_year FROM movies;";
$result = $conn->query($sql);
if($result->num_rows > 0) {
 while ($row = $result->fetch_assoc()){
  $title=$row['title'];
  $genre=$row['genre'];
  $release=$row['release_year'];
  $director=$row['director'];
  echo "Title: $title, Genre: $genre, Released in: $release, Directed by: $director <br>";
 }
}
```

```
echo "<p>Hostname: $hostname</p>";
?>
```

c. Select each file and get the raw file content link by clicking the **raw** button



d. Copy each files raw link in the address bar



8. Go to **EC2 Dashboard > Instances > Launch Instances**
    a. EC2 Details
        i. Name: FinalEC2
        ii. AMI: Amazon linux 2023 AMI
        iii. Key pair: vockey
        iv. Network Settings
            1. VPC: FinalProject
            2. Subnet: FinalPubA
            3. Existing Security Group: FinalEC2SG
        v. Advanced details
            1. User Data: **Replace the two git links with the raw github links**

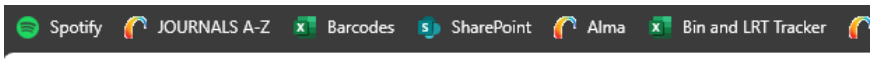
```
#!/bin/bash
sudo su
yum update -y
yum install -y httpd
yum install -y php
yum install -y mysql
yum install -y php8.2-mysqlnd.x86_64
dnf -y localinstall https://dev.mysql.com/get/mysql80-community-release-el9-4.noarch.rpm
yum install mysql mysql-community-client -y
wget https://raw.githubusercontent.com/ReplacementPath/index.html -P /var/www/html/
wget https://raw.githubusercontent.com/ReplacementPath/request.php -P /var/www/html/
systemctl start httpd
systemctl enable httpd
```

    b. Launch the instance, and when it's available and passed its checks, enter its public address into the address bar in a browser to see the web server work. It will list no movies because we have not populated the database with data yet
    c. Connect to the EC2 instance by selecting the instance and clicking **Connect > Connect**
        i. Type the below command to connect to the database: **REPLACE THE BOLD LETTERS WITH THE DATABASE ENDPOINT**
            1. sudo mysql -h **DatabaseEndpoint** -u admin -pgoldowl77

ii. Type the below to begin database creation stuff:
    1. use FinalDBName;
iii. Copy paste the below commands all at once to create a table and fill it with data, then examine the creation of data

```
CREATE TABLE movies(title VARCHAR(50) NOT NULL,genre VARCHAR(30) NOT NULL,director VARCHAR(60) NOT NULL,release_year INT NOT NULL,PRIMARY KEY(title));
DESCRIBE movies;
INSERT INTO movies VALUE ("Joker", "psychological thriller", "Todd Phillips", 2019);
INSERT INTO movies VALUE ("Alec The Movie", "Horror", "Elon Musk", 1887);
select * FROM movies;
```

d. Refresh the EC2 Public IP and you will see the page update. Adding any data to the database will show up once you request the page again

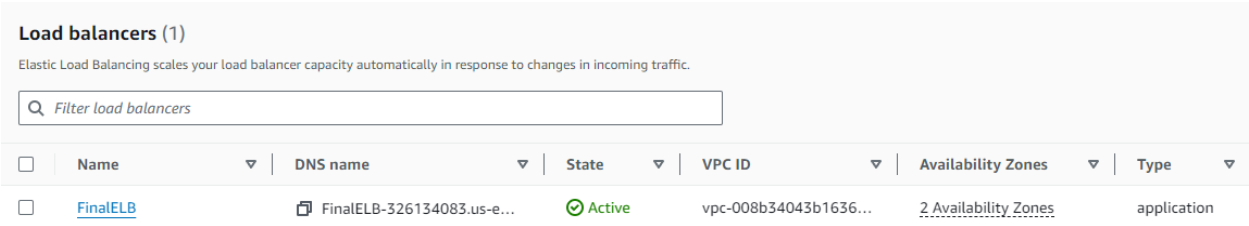


Look at these very awesome movies :D

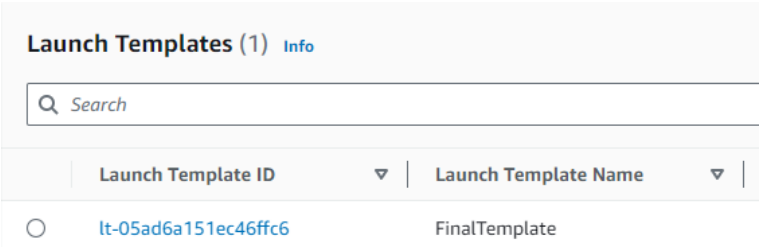
Title: Alec The Movie, Genre: Horror, Released in: 1887, Directed by: Elon Musk
Title: Joker, Genre: psychological thriller, Released in: 2019, Directed by: Todd Phillips

Hostname: ip-192-168-2-87.ec2.internal

9. Go to **EC2 Dashboard >Load Balancers > Create Load Balancer**
    a. Choose Application Load Balancer
        i. Name: FinalELB
        ii. Scheme: Internet facing
        iii. Network Mapping
            1. VPC: FinalProject VPC
            2. Select both us-east-1a and us-east-1b
                a. Choose the FinalPubA and FinalPubB subnets for the availability zones
        iv. Use the FinalEC2SG security group and de-select the default security group
        v. Under Listeners and routing click **Create target group**
            1. Target type: Instances
            2. Name: FinalTargetGroup
            3. VPC: FinalProject
            4. Next
            5. Select the EC2 instance and click “Include as pending below” to add it to the target group
            6. Click create target group
            7. Go back to the application load balancer tab
        vi. Click the refresh button next to target group and select the new FinalTargetGroup target group
        vii. Click create load balancer



10. Go back to the EC2 instances page and select the instance
    a. Click **Actions > Image and Templates > Create Image**
        i. Name: FinalAMI
        ii. Create Image
    b. Click **Instances > launch templates** in the left hand nav bar in the EC2 dashboard
        i. Click Create Launch Template
        ii. Name: FinalTemplate
        iii. Description: Template For EC2 Instances
        iv. Select the Auto Scaling guidance checkbox
        v. Application and OS Images:
            1. Select **My AMIs > Owned by me** and choose the FinalAMI AMI for the Image
        vi. Instance type:t2.Micro
        vii. KeyPair: Don't include in launch template
        viii. Network settings:
            1. Subnet: Don't include in launch template
            2. Security group: FinalEC2SG
        ix. Create launch template

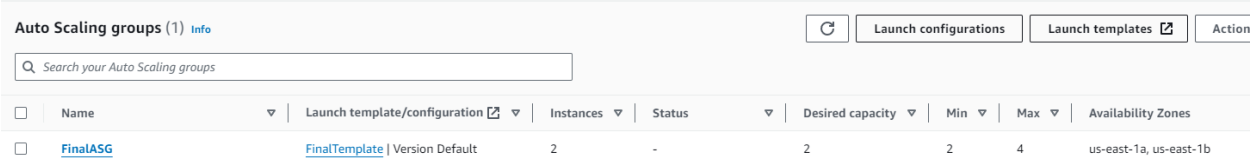


11. Go to **EC2 Dashboard > Auto Scaling Groups > Create Auto Scaling Group**
    a. Choose Launch Template
        i. Name: FinalASG
        ii. Launch templates: FinalTemplate
        iii. Click next
    b. Choose instance launch options
        i. VPC: Final Project
        ii. Availability Zones and Subnets
            1. us-east-1a, FinalPubA
            2. us-east-1b, FinalPubB
            3. Click next
    c. Configure Advanced Options

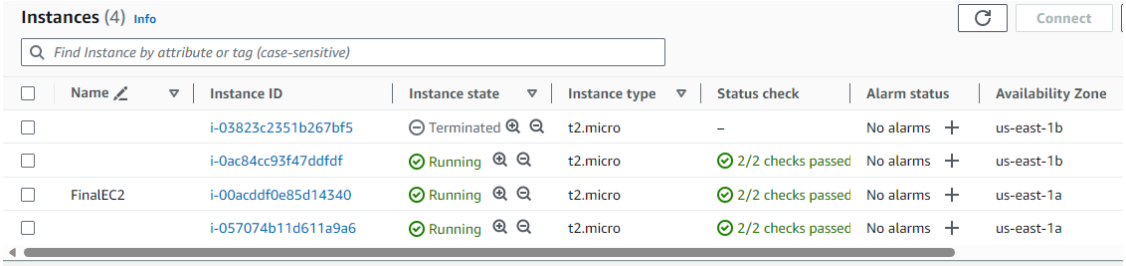
i. Attach an existing load balancer
    1. Choose from your load balancer target groups
        a. Existing LBTG: FinalTargetGroup
ii. Click next

d. Configure group size and scaling
    i. Group size
        1. Desired capacity: 3
    ii. Scaling:
        1. Minimum capacity 2
        2. Maximum capacity 4
        3. Select Target Tracking Scaling Policy
            a. Policy name: FinalTTSP
            b. Metric type: Average CPU Utilization
            c. Target Value: 50%
    iii. Next

e. Continue with defaults until creation



12. After the group is created, go back to the EC2 instance dashboard and view the created instances. Wait until these have finished being created and passed their checks before preceding. Once finished, **terminate** the FinalEC2 EC2 instance



13. Copy the DNS name of the FinalELB load balancer into several web browsers and see the effects of the load balancer