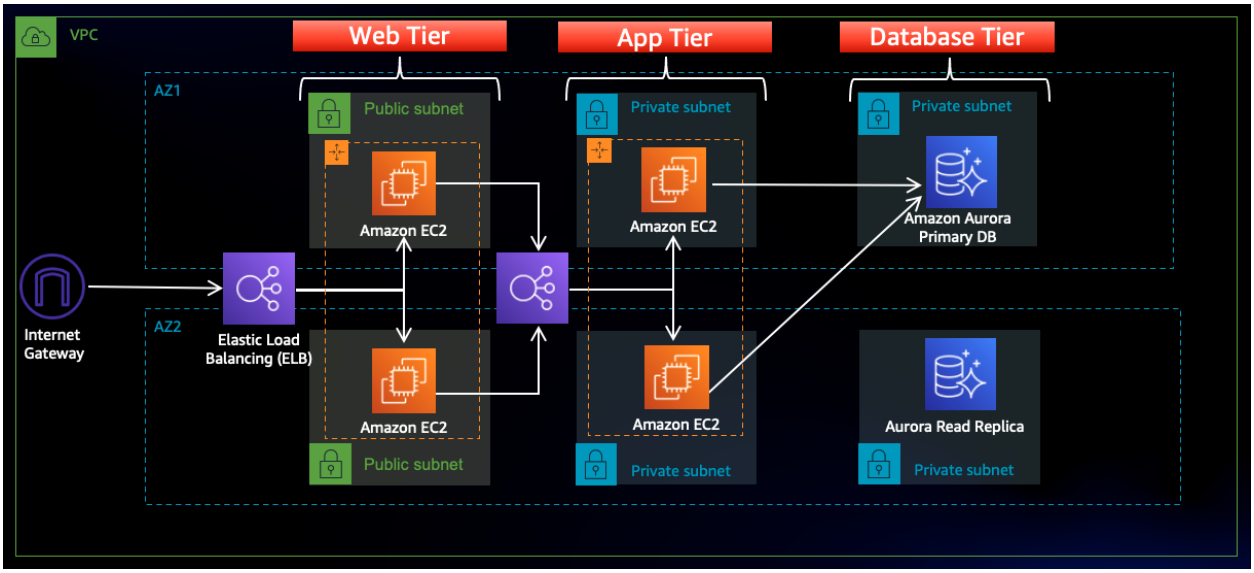
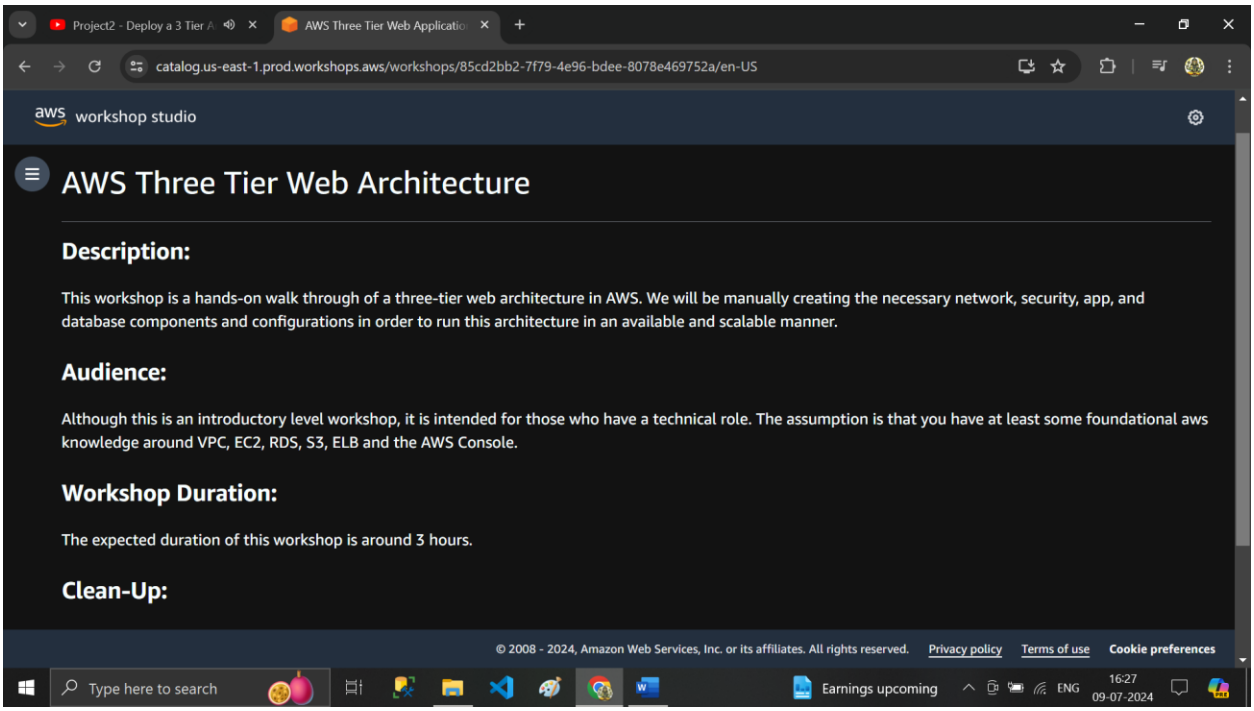
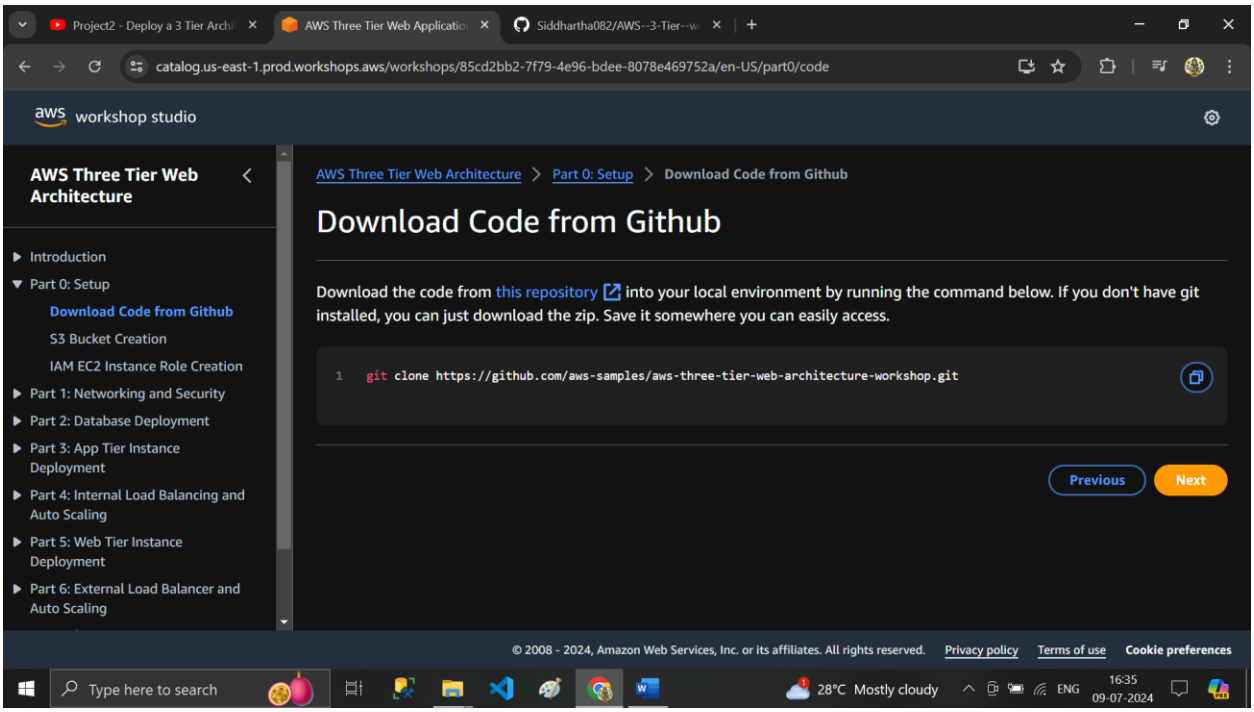
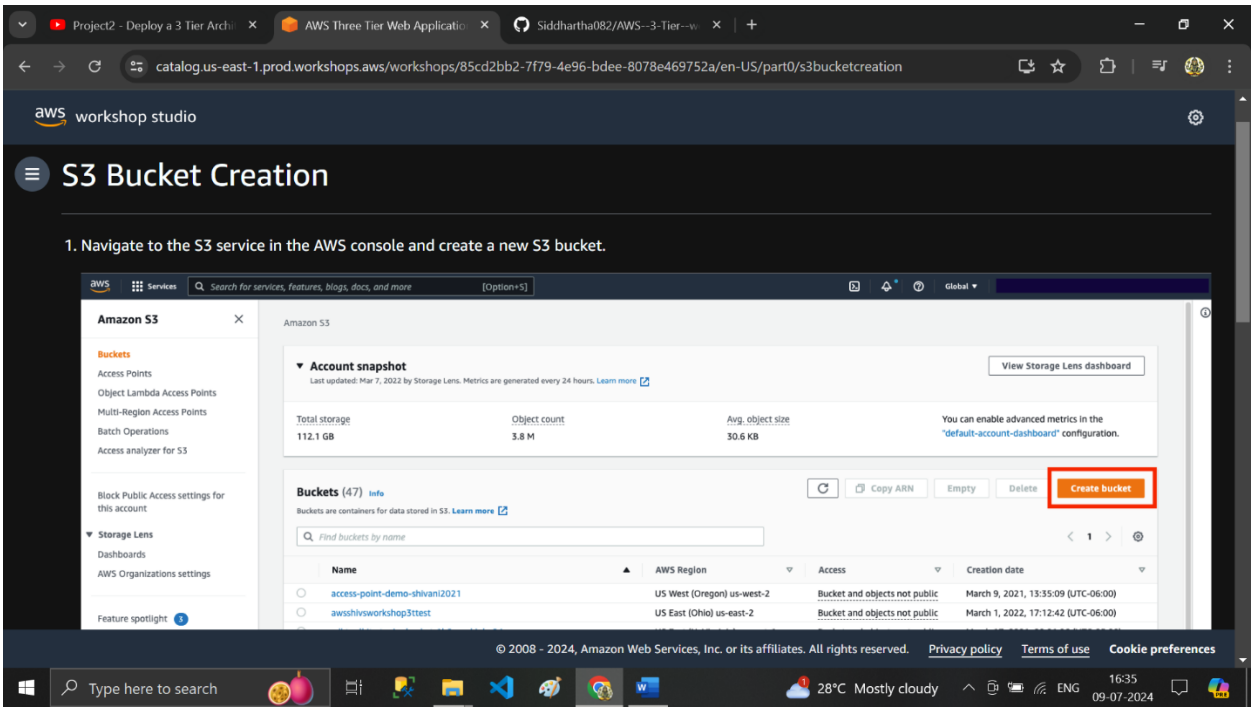
Architecture Diagram for AWS

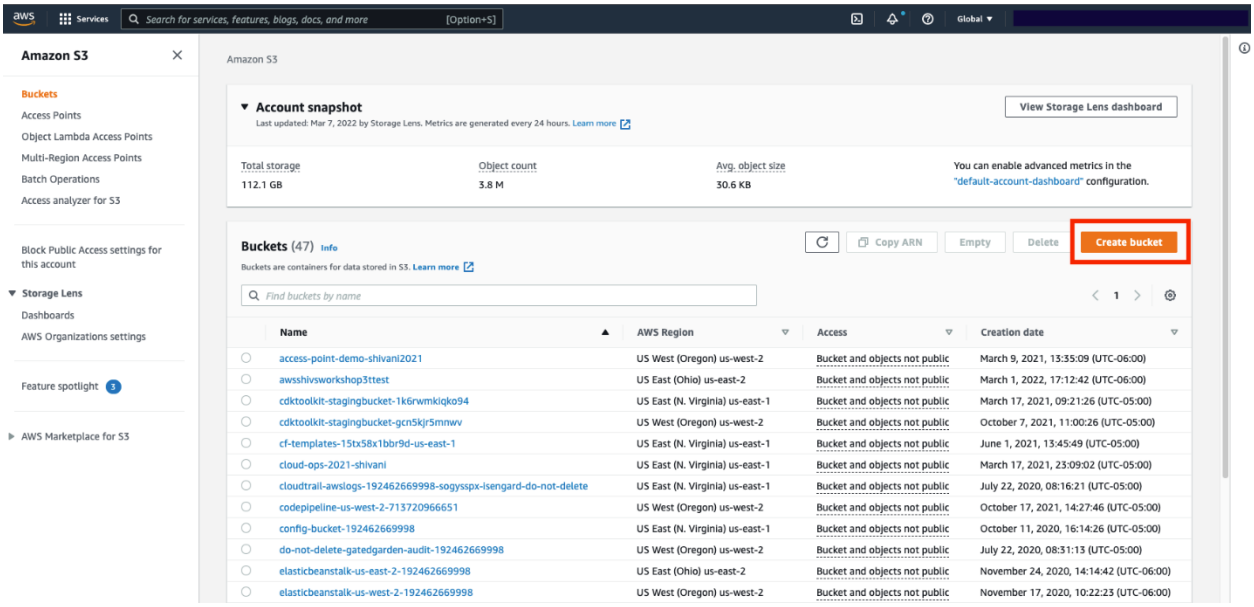


In this architecture, a public-facing Application Load Balancer forwards client traffic to our web tier EC2 instances. The web tier is running Nginx webservers that are configured to serve a React.js website and redirects our API calls to the application tier's internal facing load balancer. The internal facing load balancer then forwards that traffic to the application tier, which is written in Node.js. The application tier manipulates data in an Aurora MySQL multi-AZ database and returns it to our web tier. Load balancing, health checks and autoscaling groups are created at each layer to maintain the availability of this architecture.

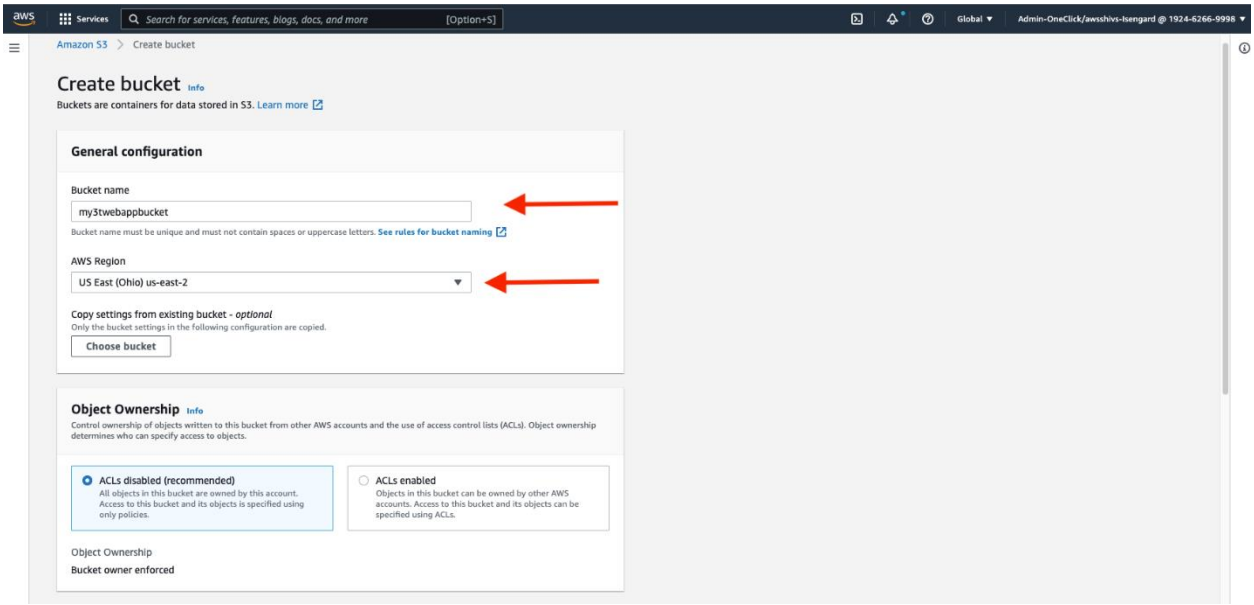

Project2 - Deploy a 3 Tier A
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US
aws workshop studio
AWS Three Tier Web Architecture
Description:
This workshop is a hands-on walk through of a three-tier web architecture in AWS. We will be manually creating the necessary network, security, app, and database components and configurations in order to run this architecture in an available and scalable manner.
Audience:
Although this is an introductory level workshop, it is intended for those who have a technical role. The assumption is that you have at least some foundational aws knowledge around VPC, EC2, RDS, S3, ELB and the AWS Console.
Workshop Duration:
The expected duration of this workshop is around 3 hours.
Clean-Up:
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
Earnings upcoming
ENG
16:27
09-07-2024


Project2 - Deploy a 3 Tier Archi
AWS Three Tier Web Applicatio
Siddhartha082/AWS--3-Tier--w
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part0/code
aws workshop studio
AWS Three Tier Web Architecture
Introduction
Part 0: Setup
Download Code from Github
S3 Bucket Creation
IAM EC2 Instance Role Creation
Part 1: Networking and Security
Part 2: Database Deployment
Part 3: App Tier Instance Deployment
Part 4: Internal Load Balancing and Auto Scaling
Part 5: Web Tier Instance Deployment
Part 6: External Load Balancer and Auto Scaling
AWS Three Tier Web Architecture > Part 0: Setup > Download Code from Github
Download Code from Github
Download the code from this repository into your local environment by running the command below. If you don't have git installed, you can just download the zip. Save it somewhere you can easily access.
1 git clone https://github.com/aws-samples/aws-three-tier-web-architecture-workshop.git
Previous
Next
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
28°C Mostly cloudy
ENG
16:35
09-07-2024



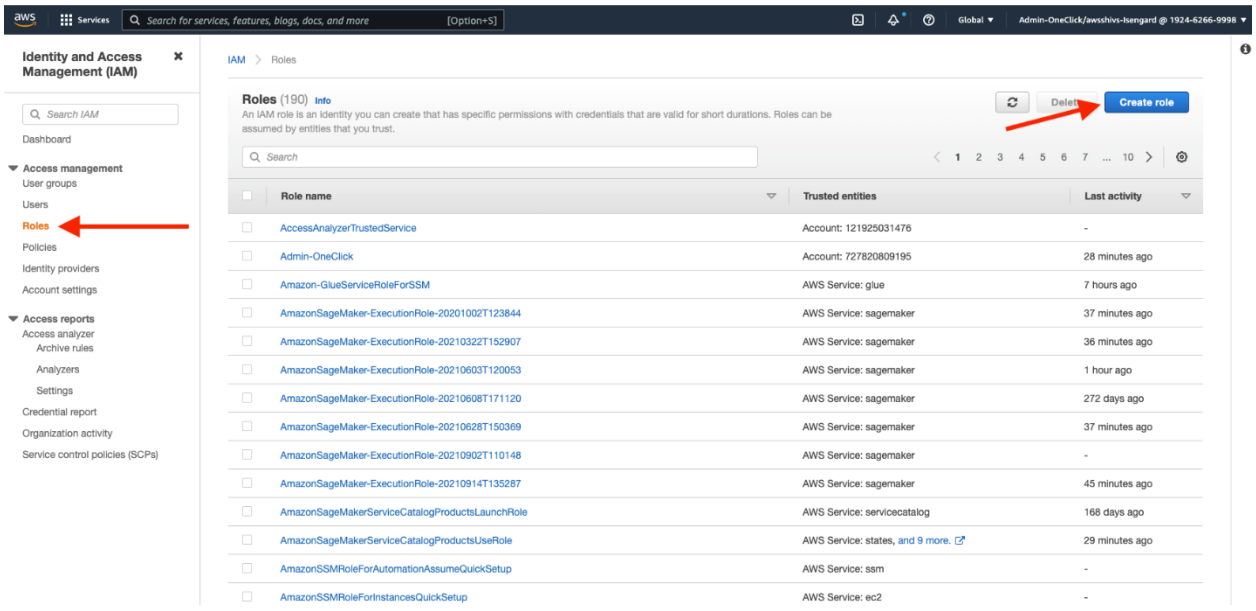


Give it a unique name, and then leave all the defaults as in. Make sure to select the region that you intend to run this whole lab in. This bucket is where we will upload our code later.

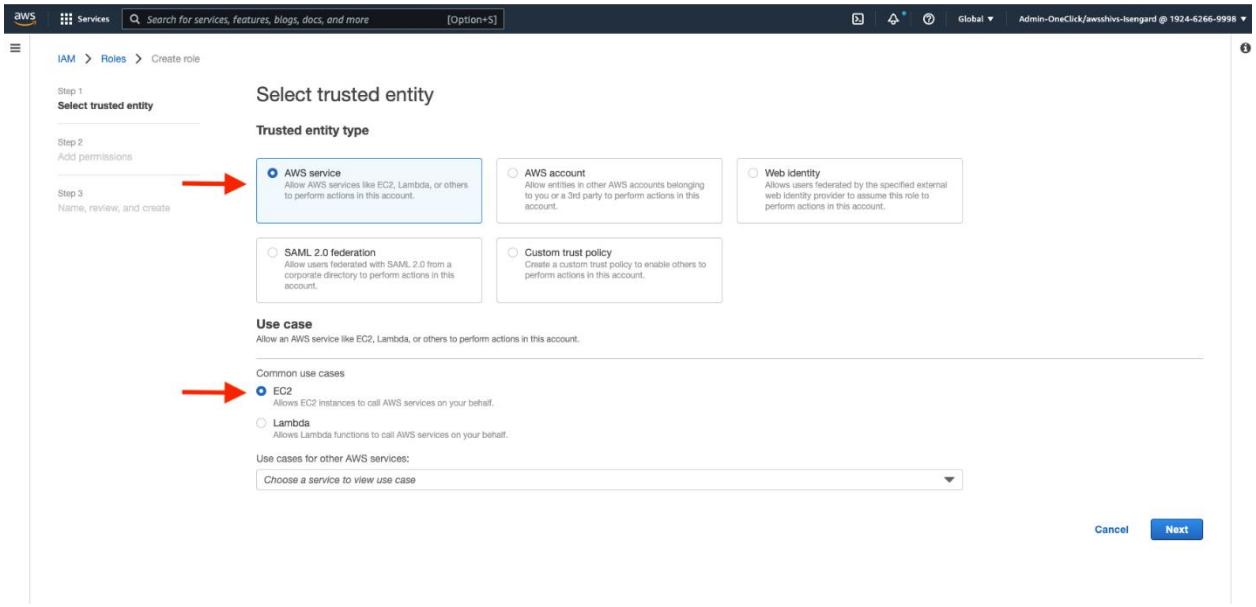


IAM EC2 Instance Role Creation

1 .Navigate to the IAM dashboard in the AWS console and create an EC2 role.

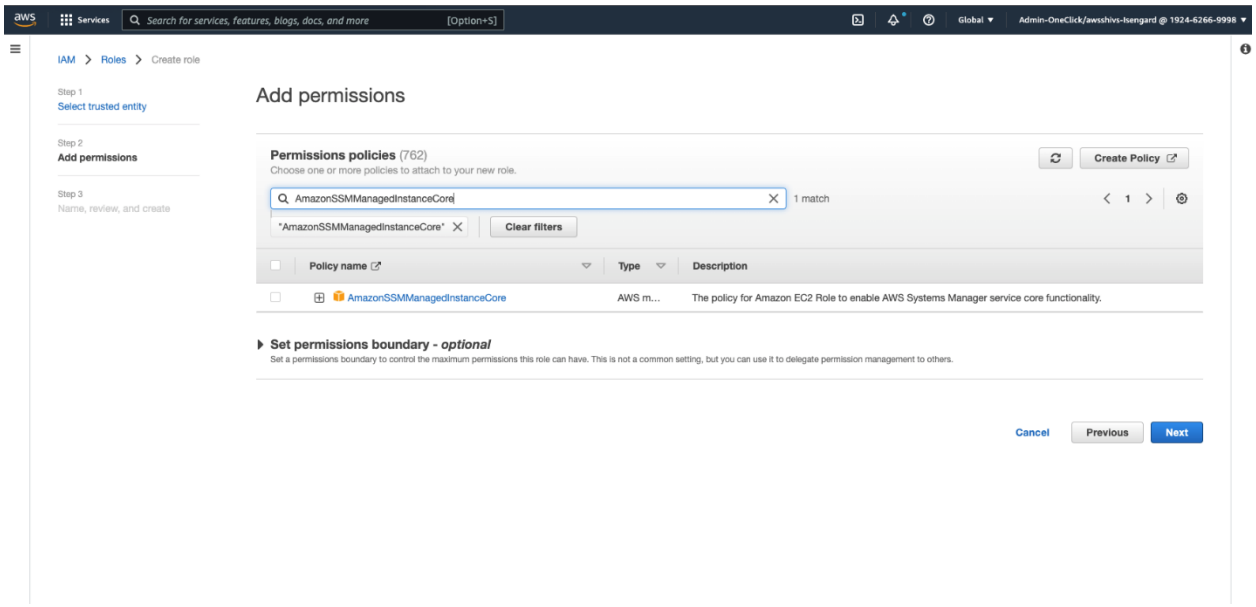


Select EC2 as the trusted entity.

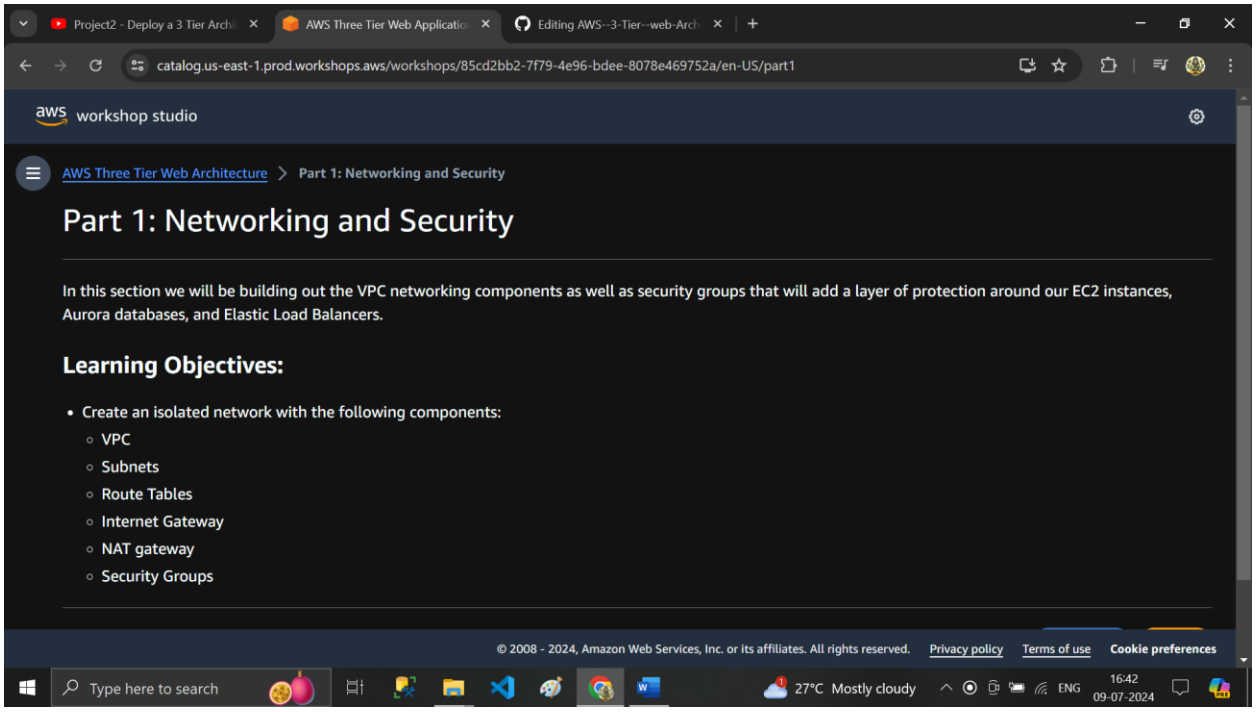


When adding permissions, include the following AWS managed policies. You can search for them and select them. These policies will allow our instances to download our code from S3 and use Systems Manager Session Manager to securely connect to our instances without SSH keys through the AWS console.

- **AmazonSSMManagedInstanceCore**
- **AmazonS3ReadOnlyAccess**

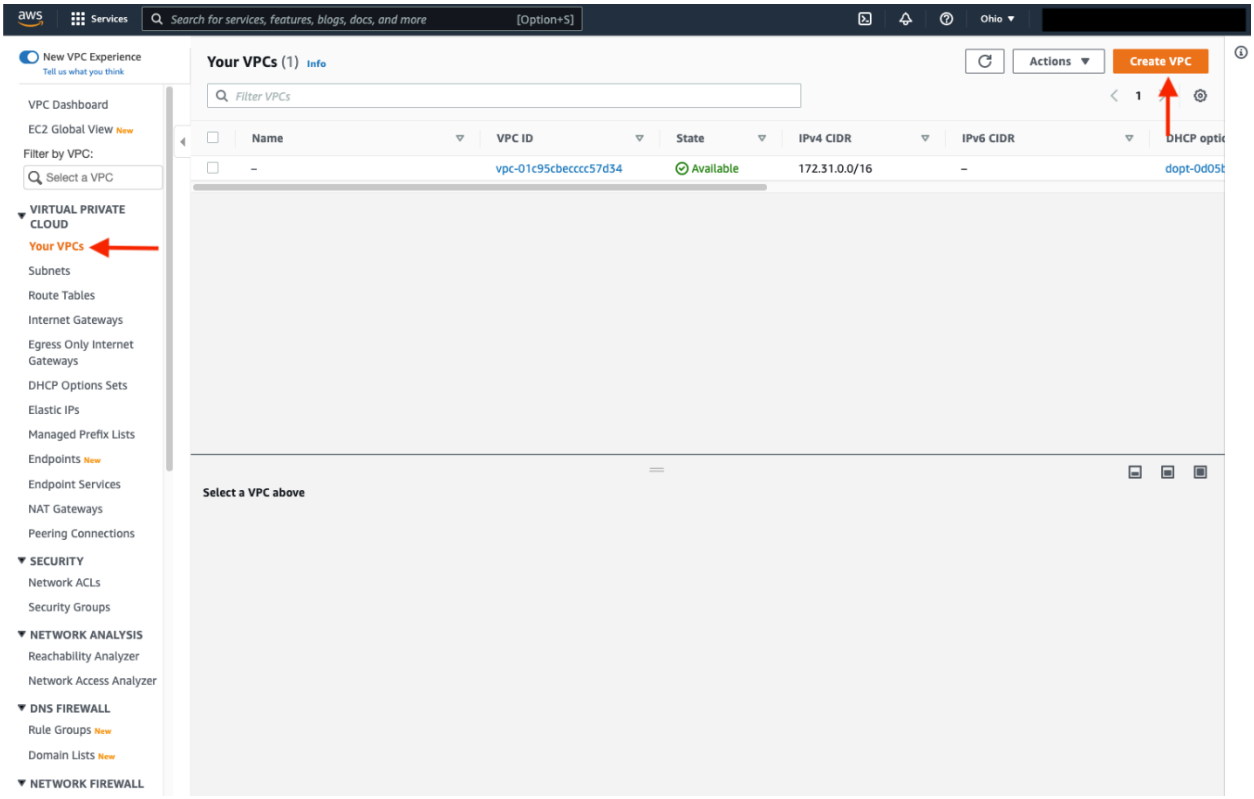


1. Give your role a name, and then click **Create Role**.



VPC and Subnets

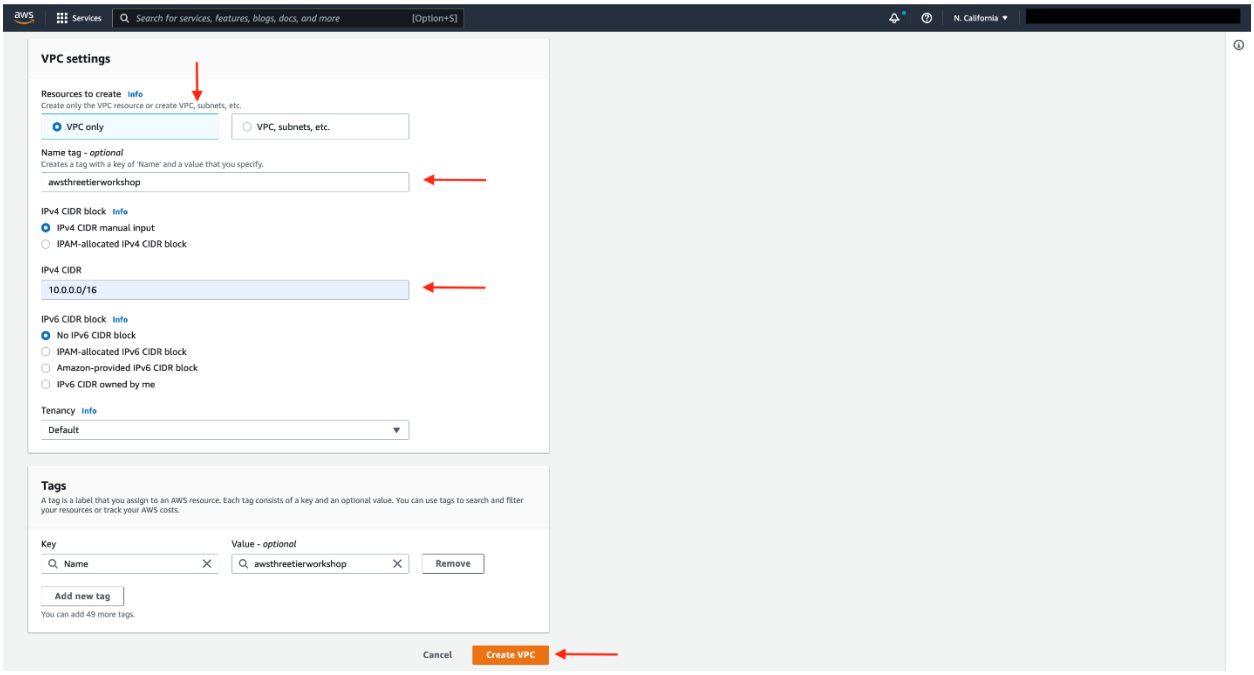
1. **VPC Creation**
    1. Navigate to the VPC dashboard in the AWS console and navigate to **Your VPCs** on the left hand side.



2. Make sure **VPC only** is selected, and fill out the VPC Settings with a **Name tag** and a **CIDR range** of your choice.

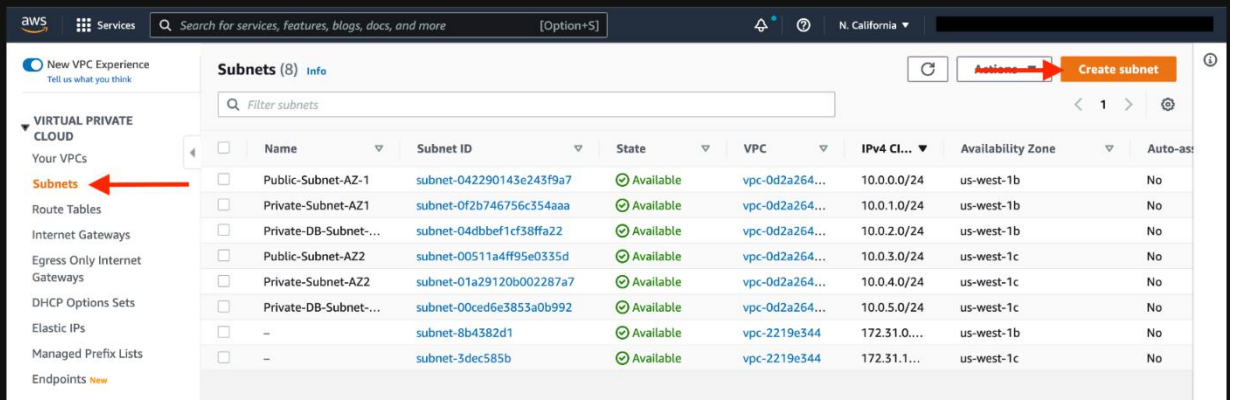
NOTE: *Make sure you pay attention to the region you're deploying all your resources in. You'll want to stay consistent for this workshop.*

NOTE: *Choose a CIDR range that will allow you to create at least 6 subnets.*



2. Subnet Creation

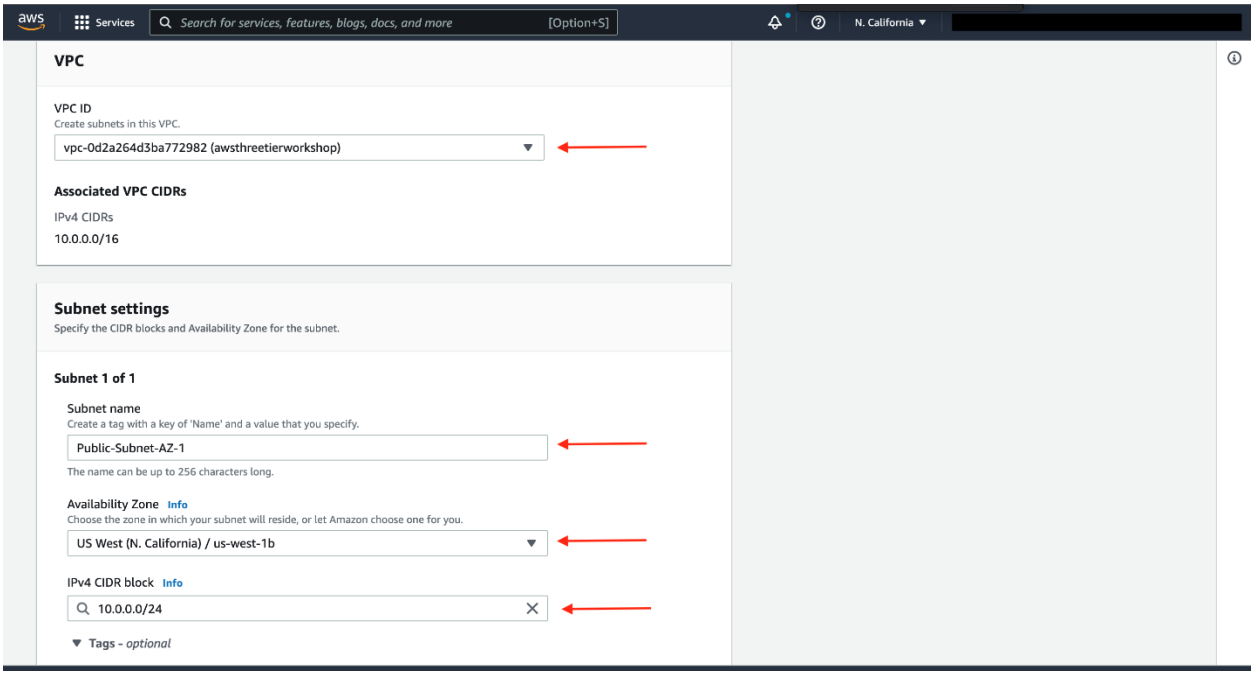
1. Next, create your subnets by navigating to **Subnets** on the left side of the dashboard and clicking **Create subnet**.

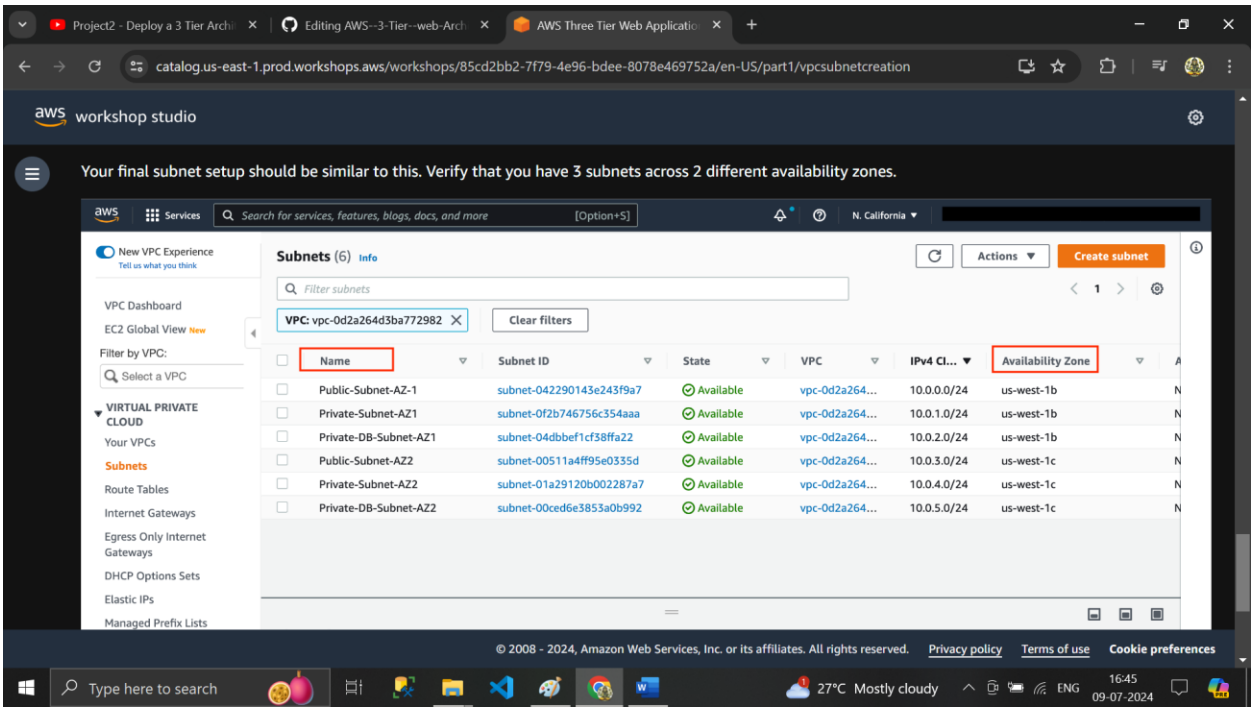


2. We will need **six** subnets across **two** availability zones. That means that **three** subnets will be in one availability zone, and three subnets will be in another zone. Each subnet in one availability zone will correspond to one layer of our three tier architecture. Create each of the 6 subnets by specifying the VPC we created in part 1 and then choose a name, availability zone, and appropriate CIDR range for each of the subnets.

   NOTE: It may be helpful to have a naming convention that will help you remember what each subnet is for. For example in one AZ you might have the following: ***Public-Web-Subnet-AZ-1, Private-App-Subnet-AZ-1, Private-DB-Subnet-AZ-1****.*

   NOTE: Remember, your CIDR range for the subnets will be subsets of your VPC CIDR range.

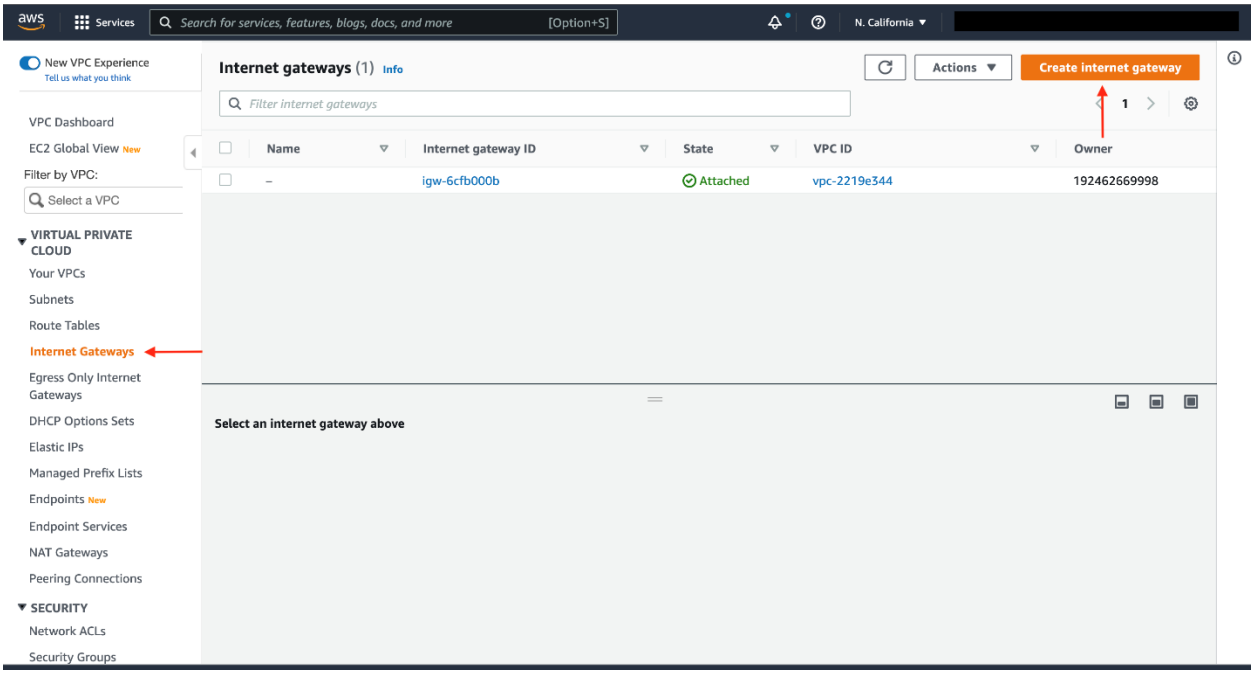


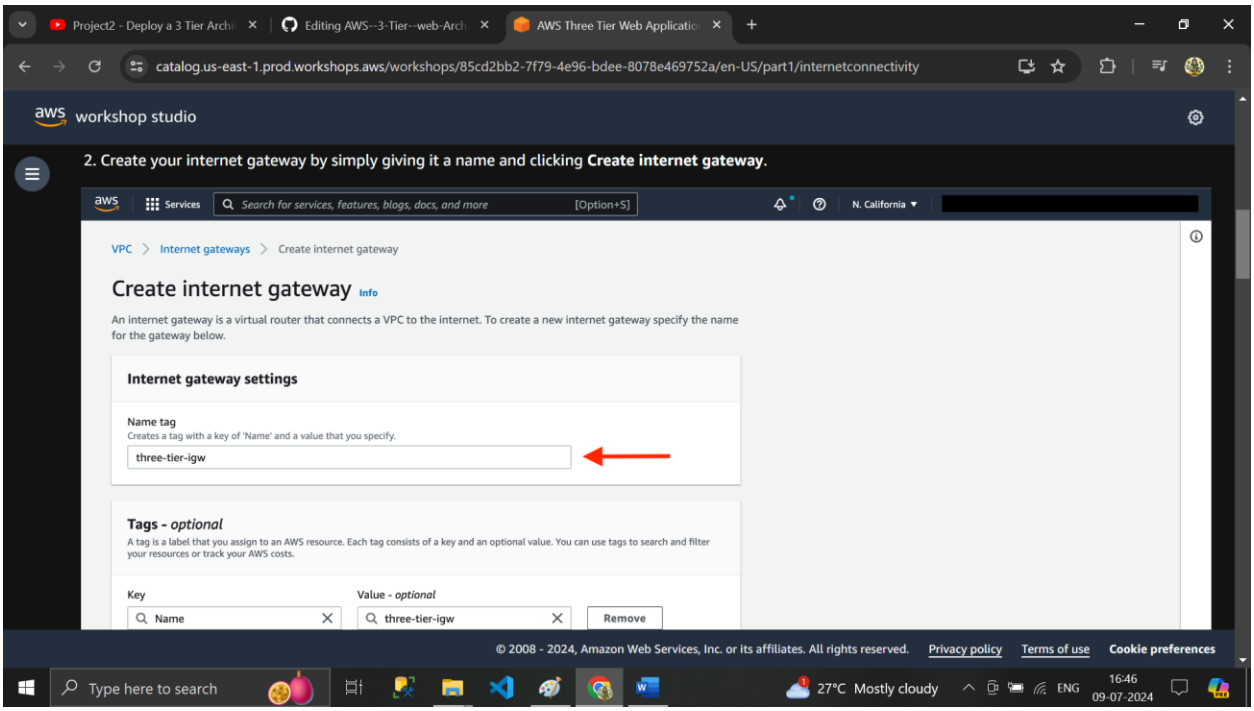


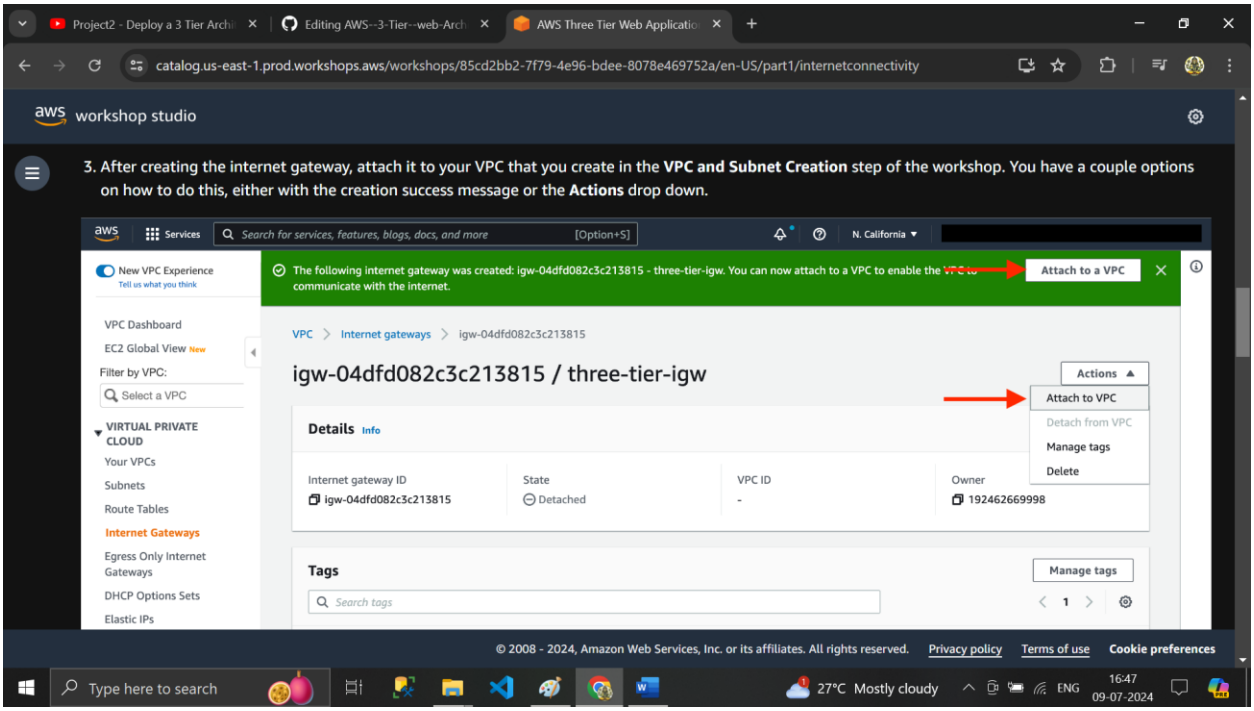
Internet Connectivity

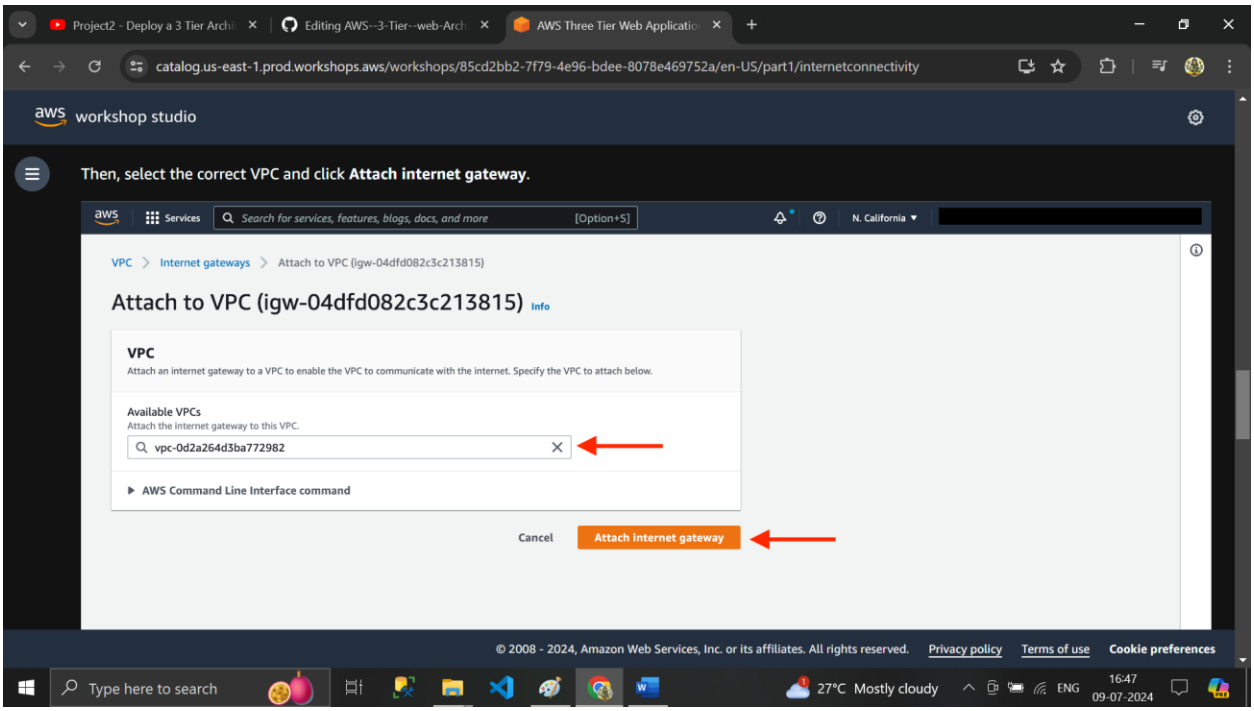
1. **Internet Gateway**

    1. In order to give the public subnets in our VPC internet access we will have to create and attach an Internet Gateway. On the left hand side of the VPC dashboard, select **Internet Gateway**.



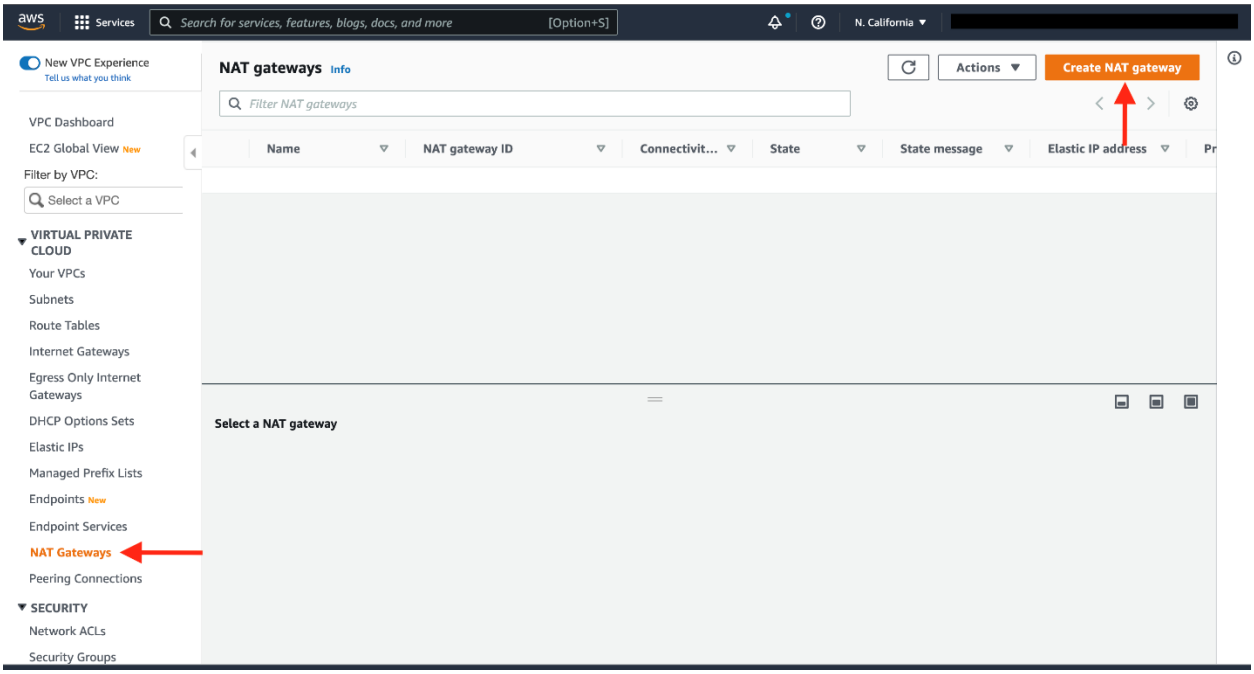

Project2 - Deploy a 3 Tier Archi
Editing AWS--3-Tier--web-Arch
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part1/internetconnectivity
aws workshop studio
2. Create your internet gateway by simply giving it a name and clicking Create internet gateway.
Services
Search for services, features, blogs, docs, and more
[Option+S]
N. California
VPC > Internet gateways > Create internet gateway
Create internet gateway Info
An internet gateway is a virtual router that connects a VPC to the internet. To create a new internet gateway specify the name for the gateway below.
Internet gateway settings
Name tag
Creates a tag with a key of 'Name' and a value that you specify.
three-tier-igw
Tags - optional
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.
Key
Value - optional
Name
three-tier-igw
Remove
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
27°C Mostly cloudy
ENG
16:46
09-07-2024

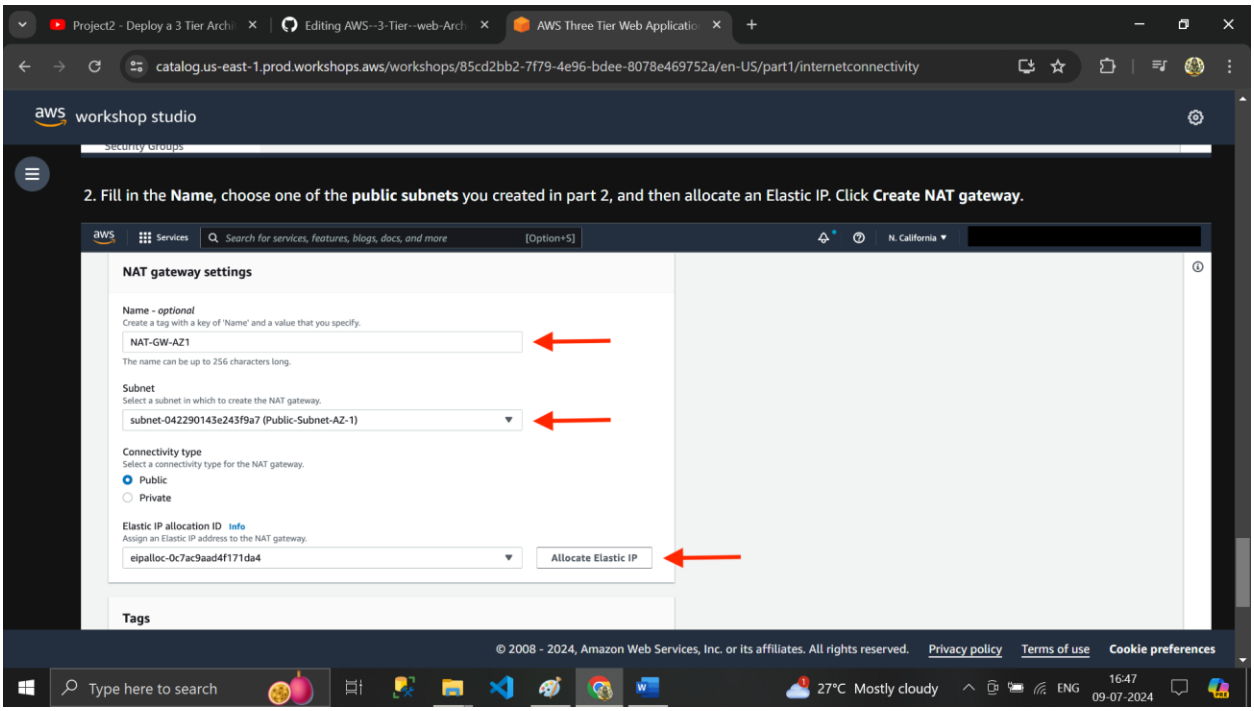

Project2 - Deploy a 3 Tier Archi
Editing AWS--3-Tier--web-Arch
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part1/internetconnectivity
aws workshop studio
3. After creating the internet gateway, attach it to your VPC that you create in the VPC and Subnet Creation step of the workshop. You have a couple options on how to do this, either with the creation success message or the Actions drop down.
Services
Search for services, features, blogs, docs, and more
[Option+S]
N. California
New VPC Experience
Tell us what you think
VPC Dashboard
EC2 Global View New
Filter by VPC:
Select a VPC
VIRTUAL PRIVATE CLOUD
Your VPCs
Subnets
Route Tables
Internet Gateways
Egress Only Internet Gateways
DHCP Options Sets
Elastic IPs
The following internet gateway was created: igw-04dfd082c3c213815 - three-tier-igw. You can now attach to a VPC to enable the VPC to communicate with the internet.
Attach to a VPC
VPC > Internet gateways > igw-04dfd082c3c213815
igw-04dfd082c3c213815 / three-tier-igw
Actions
Attach to VPC
Detach from VPC
Manage tags
Delete
Details Info
Internet gateway ID
igw-04dfd082c3c213815
State
Detached
VPC ID
-
Owner
192462669998
Tags
Manage tags
Search tags
1
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
27°C Mostly cloudy
ENG
16:47
09-07-2024

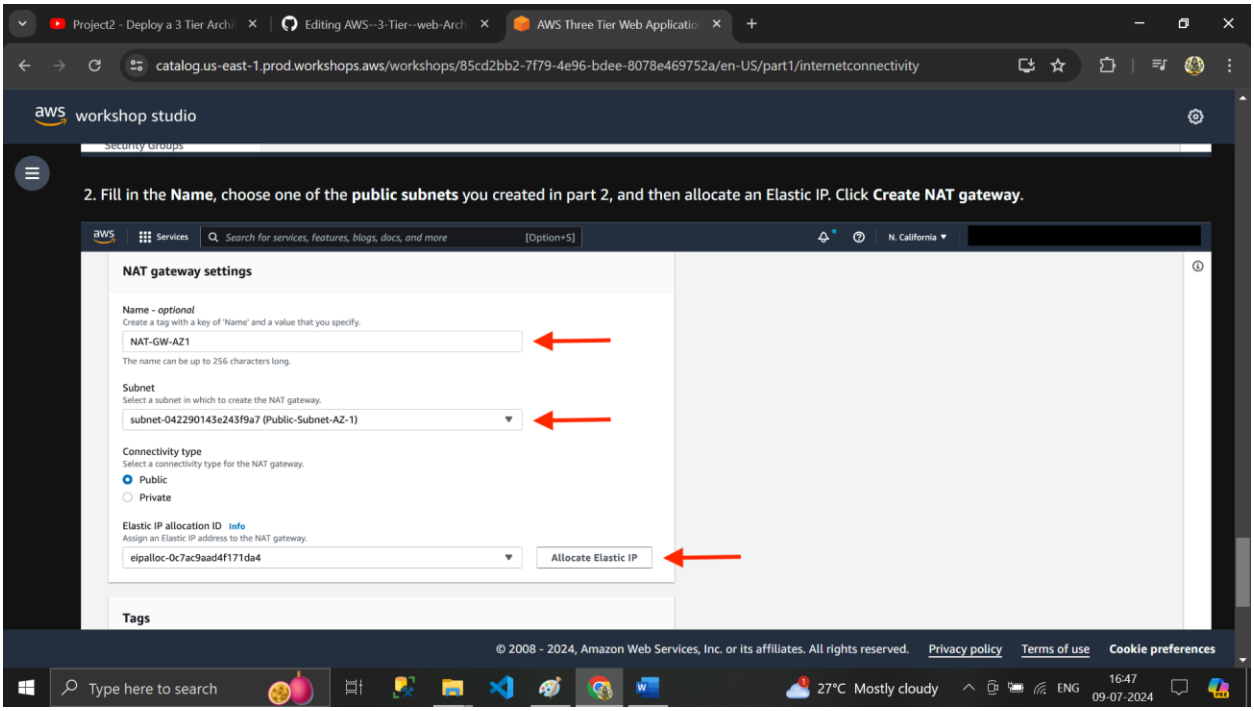


2. NAT Gateway

1. In order for our instances in the app layer private subnet to be able to access the internet they will need to go through a NAT Gateway. For high availability, you'll deploy one NAT gateway in each of your **public** subnets. Navigate to **NAT Gateways** on the left side of the current dashboard and click **Create NAT Gateway**.

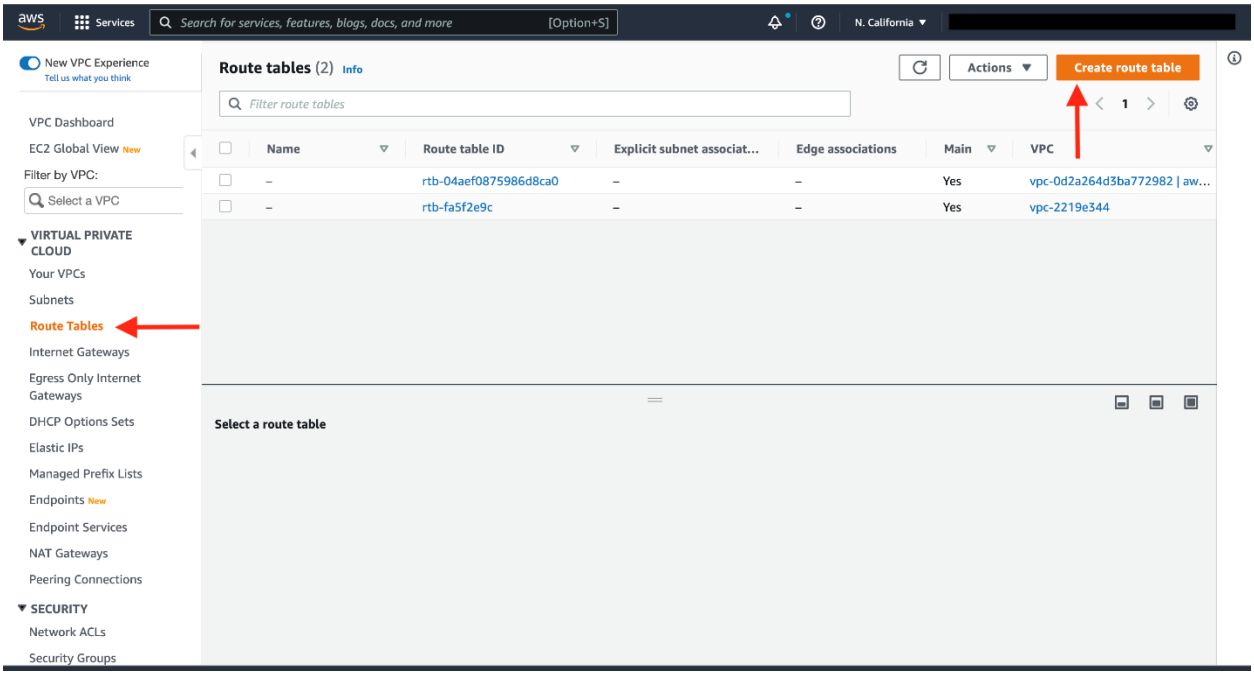


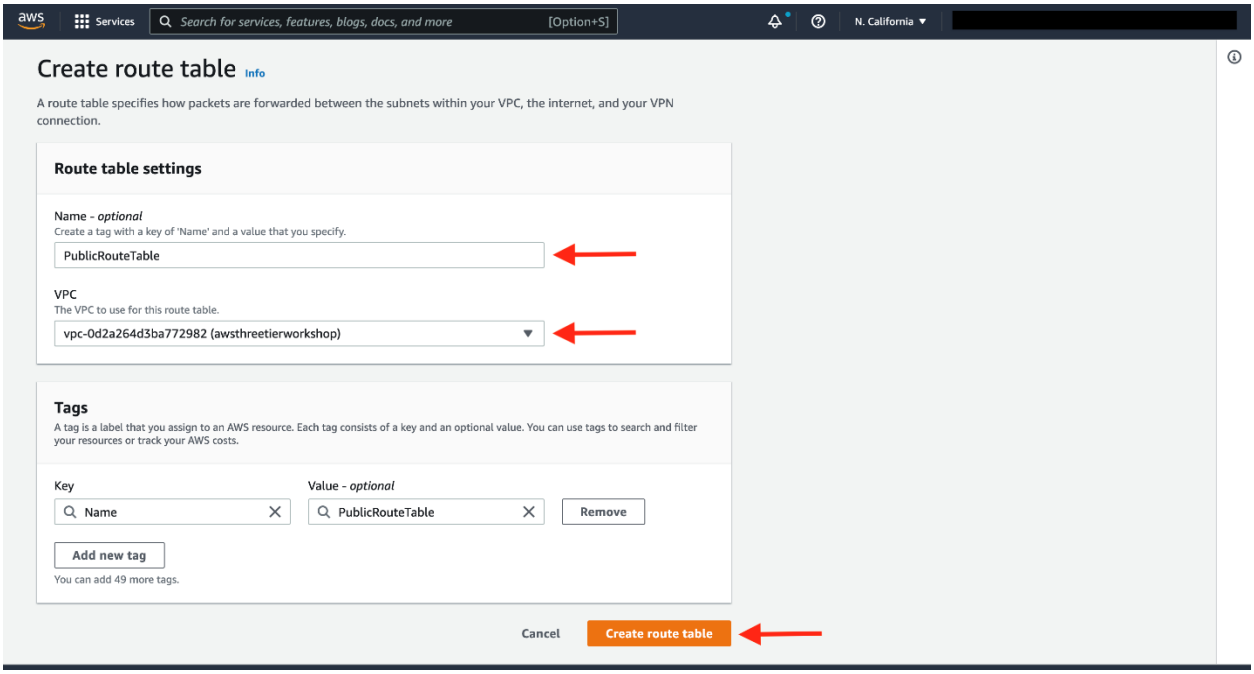


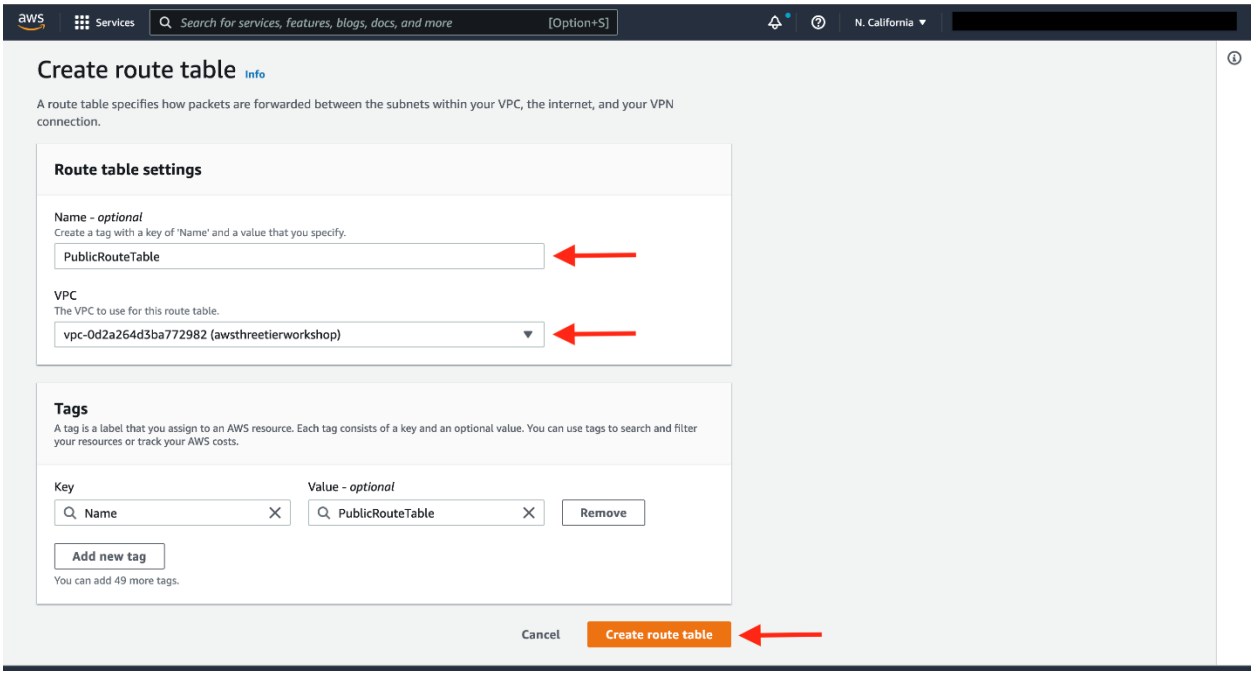


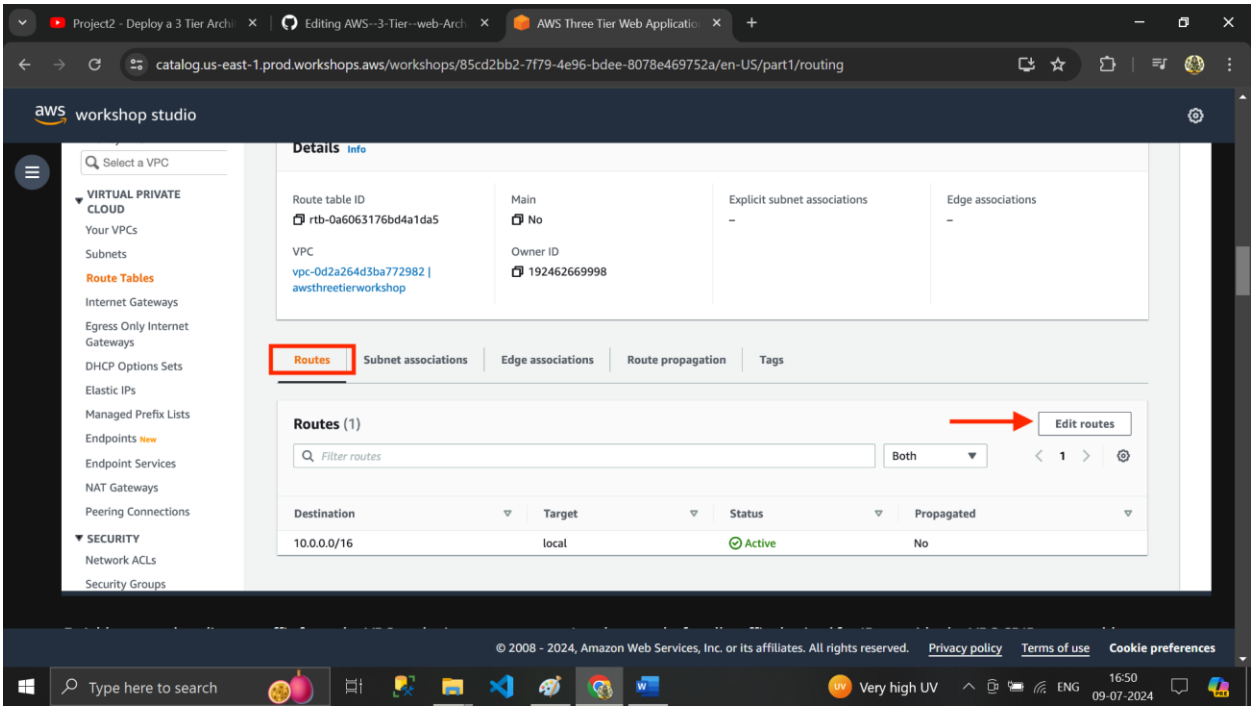
Routing Configuration

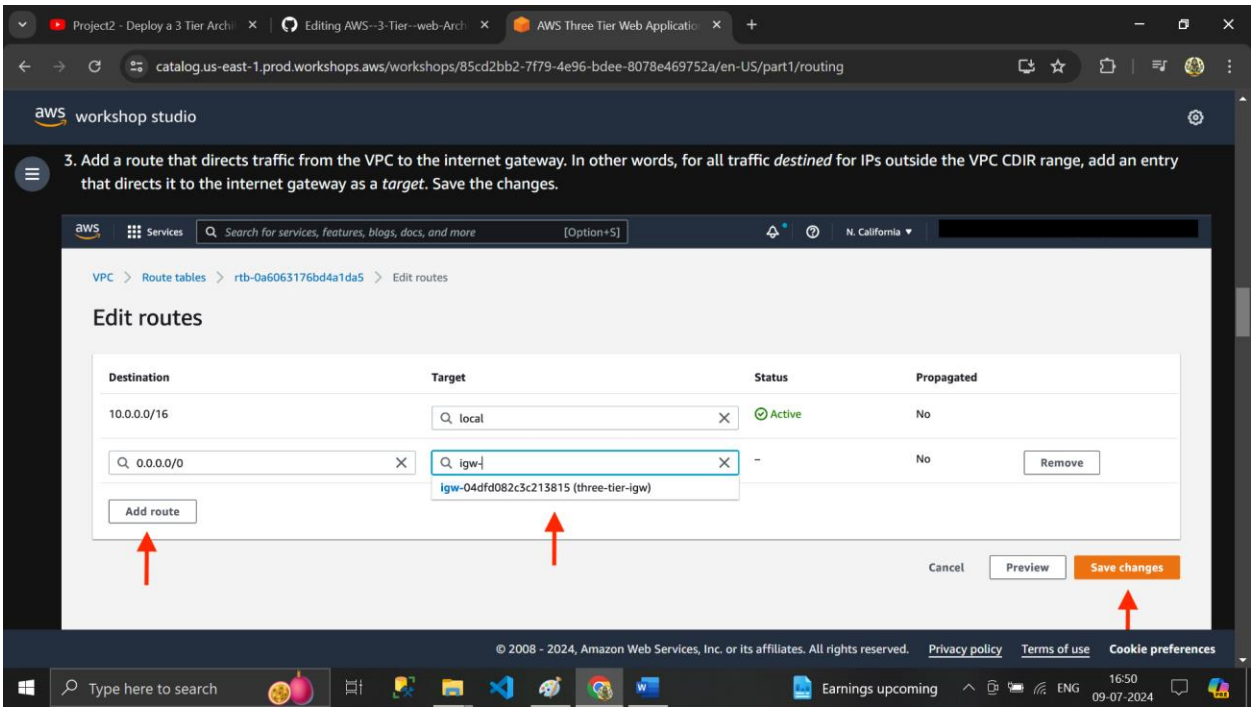
1. Navigate to **Route Tables** on the left side of the VPC dashboard and click **Create route table** First, let's create one route table for the web layer *public subnets* and name it accordingly.


Services
Search for services, features, blogs, docs, and more
[Option+S]
N. California
New VPC Experience
Tell us what you think
VPC Dashboard
EC2 Global View New
Filter by VPC:
Select a VPC
VIRTUAL PRIVATE CLOUD
Your VPCs
Subnets
Route Tables
Internet Gateways
Egress Only Internet Gateways
DHCP Options Sets
Elastic IPs
Managed Prefix Lists
Endpoints New
Endpoint Services
NAT Gateways
Peering Connections
SECURITY
Network ACLs
Security Groups
Route tables (2) Info
Actions
Create route table
Filter route tables
Name
Route table ID
Explicit subnet associat...
Edge associations
Main
VPC
rtb-04aef0875986d8ca0
rtb-fa5f2e9c
Yes
vpc-0d2a264d3ba772982 | aw...
vpc-2219e344
Select a route table

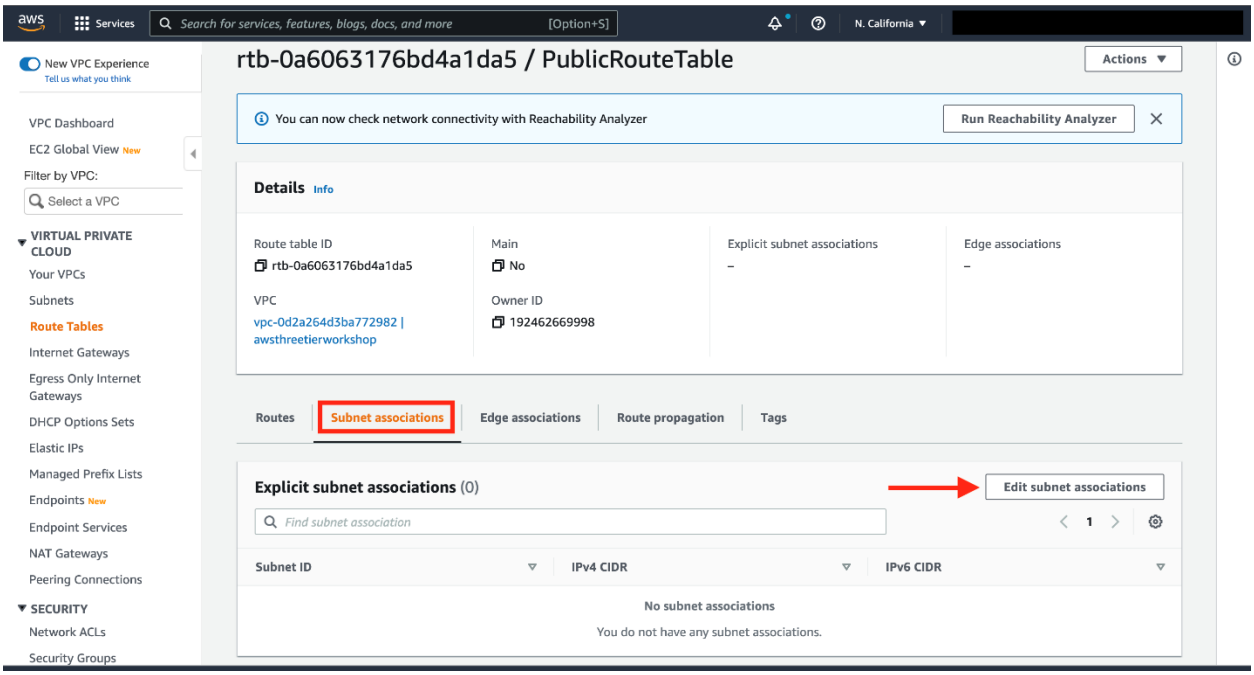

Services
Search for services, features, blogs, docs, and more
[Option+S]
N. California
Create route table Info
A route table specifies how packets are forwarded between the subnets within your VPC, the internet, and your VPN connection.
Route table settings
Name - optional
Create a tag with a key of 'Name' and a value that you specify.
PublicRouteTable
VPC
The VPC to use for this route table.
vpc-0d2a264d3ba772982 (awsthreetierworkshop)
Tags
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.
Key
Value - optional
Name
PublicRouteTable
Remove
Add new tag
You can add 49 more tags.
Cancel
Create route table


Services
Search for services, features, blogs, docs, and more
[Option+S]
N. California
Create route table Info
A route table specifies how packets are forwarded between the subnets within your VPC, the internet, and your VPN connection.
Route table settings
Name - optional
Create a tag with a key of 'Name' and a value that you specify.
PublicRouteTable
VPC
The VPC to use for this route table.
vpc-0d2a264d3ba772982 (awsthreetierworkshop)
Tags
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.
Key
Value - optional
Name
PublicRouteTable
Remove
Add new tag
You can add 49 more tags.
Cancel
Create route table

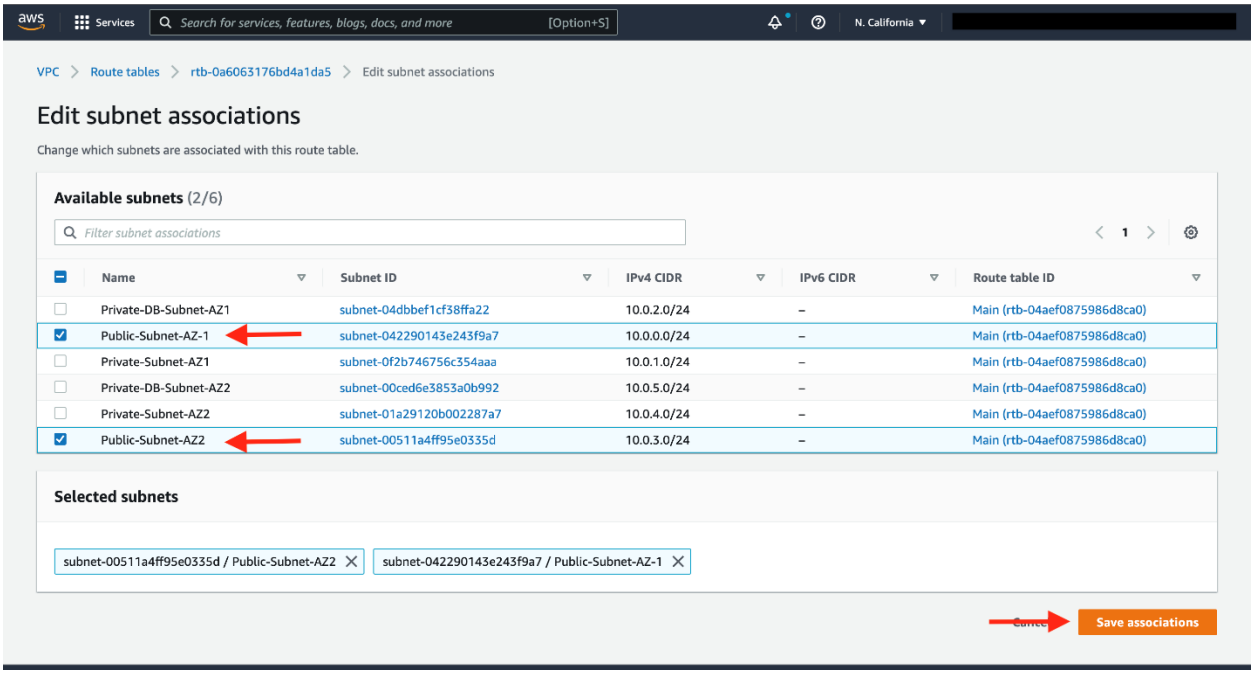

catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part1/routing
aws workshop studio
Select a VPC
VIRTUAL PRIVATE CLOUD
Your VPCs
Subnets
Route Tables
Internet Gateways
Egress Only Internet Gateways
DHCP Options Sets
Elastic IPs
Managed Prefix Lists
Endpoints New
Endpoint Services
NAT Gateways
Peering Connections
SECURITY
Network ACLs
Security Groups
Details Info
Route table ID
rtb-0a6063176bd4a1da5
Main
No
Explicit subnet associations
-
Edge associations
-
VPC
vpc-0d2a264d3ba772982 | awsthreetierworkshop
Owner ID
192462669998
Routes
Subnet associations
Edge associations
Route propagation
Tags
Routes (1)
Edit routes
Filter routes
Both
1
Destination
Target
Status
Propagated
10.0.0.0/16
local
Active
No
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
Very high UV
ENG
16:50
09-07-2024

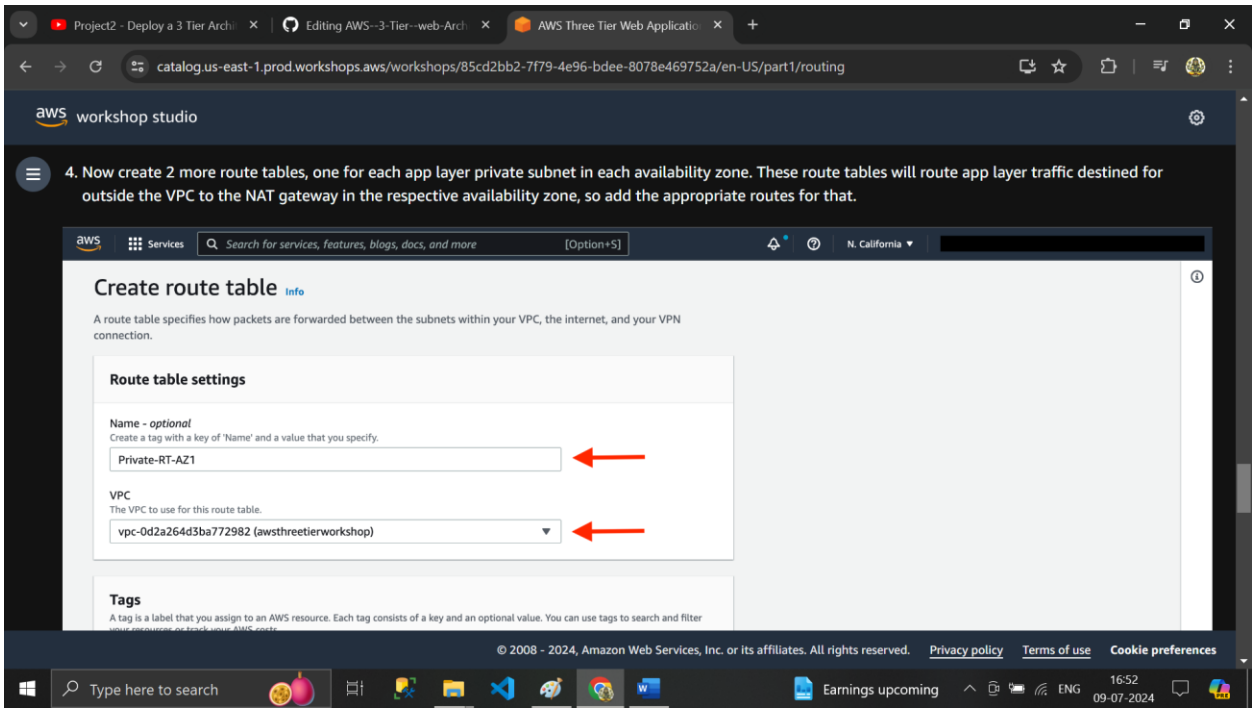


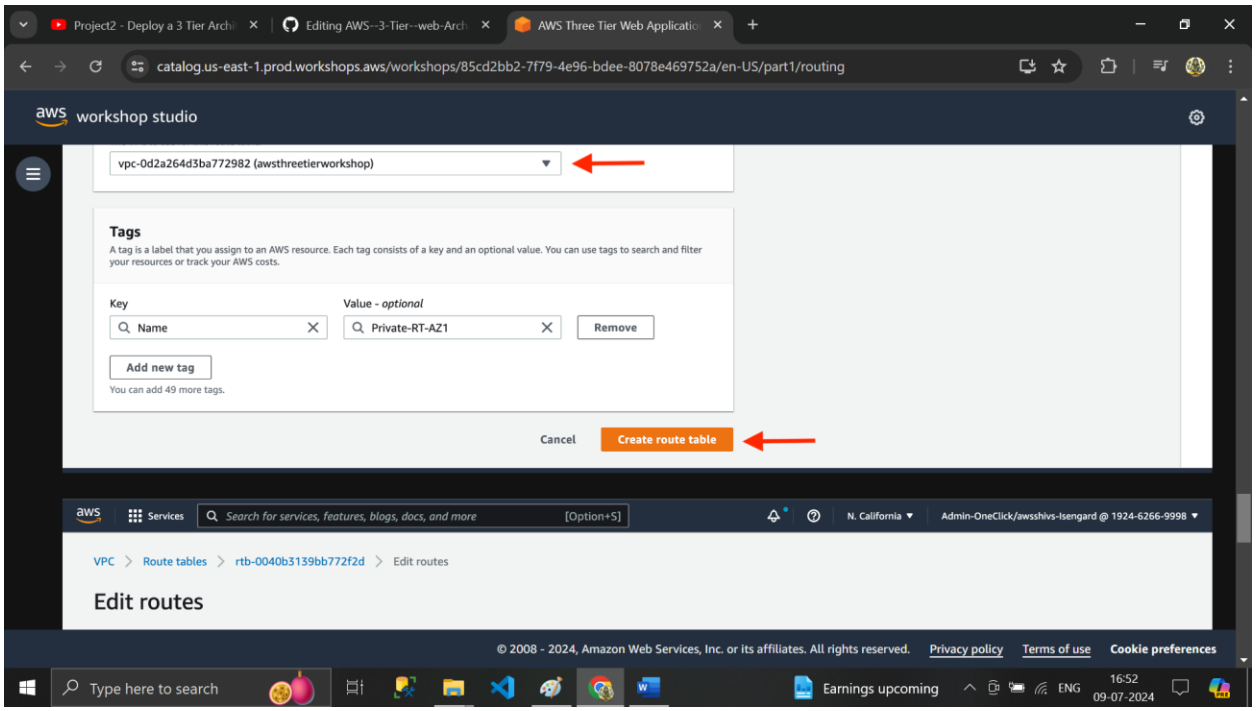
4. Edit the *Explicit Subnet Associations* of the route table by navigating to the route table details again. Select **Subnet Associations** and click **Edit subnet associations**.

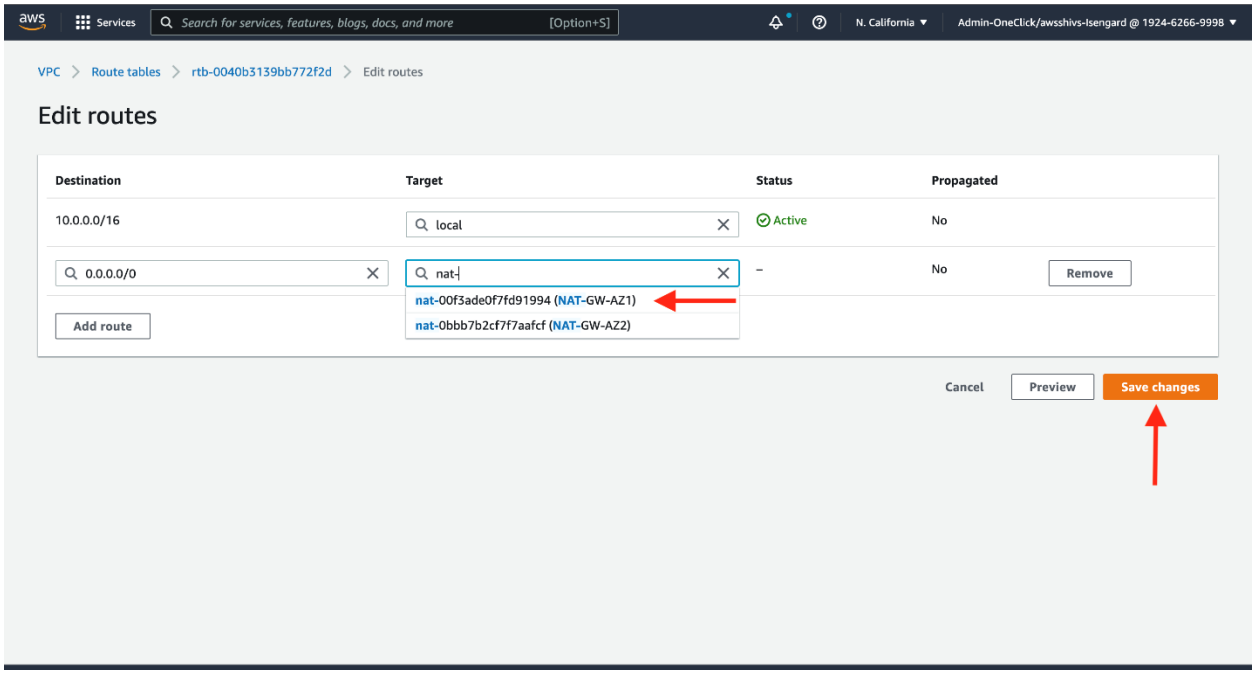


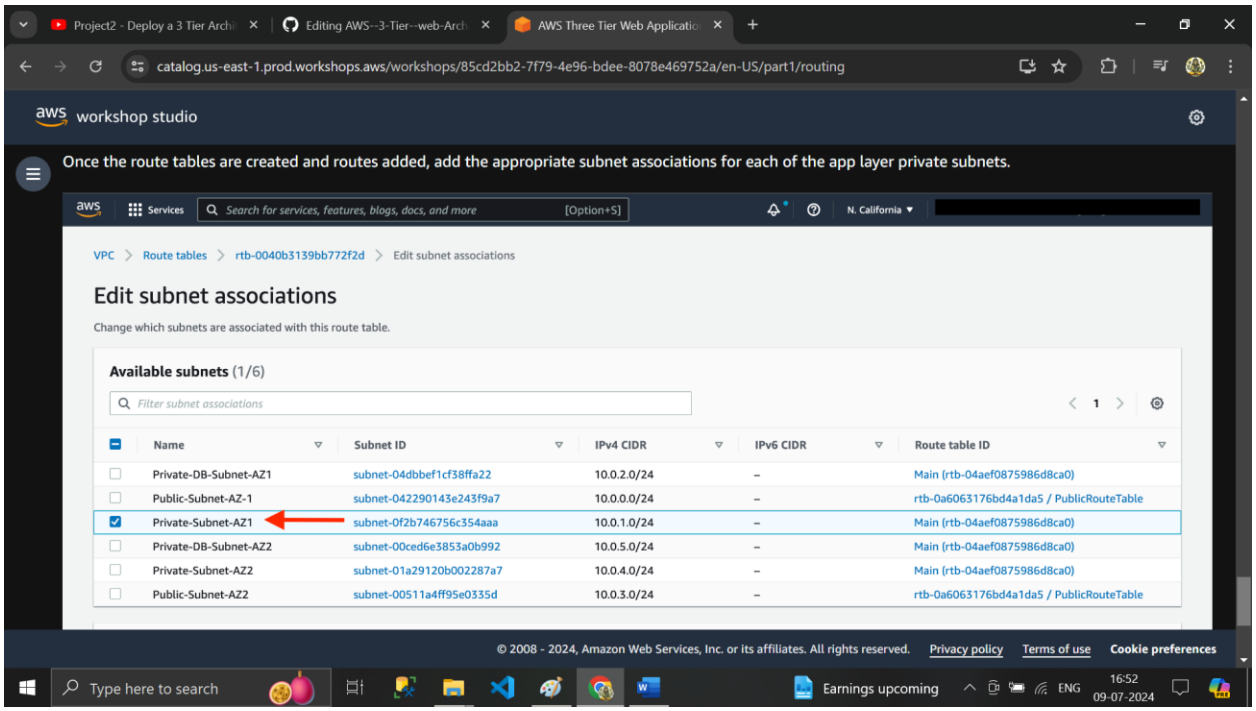
Select the two web layer public subnets you created eariler and click Save associations.

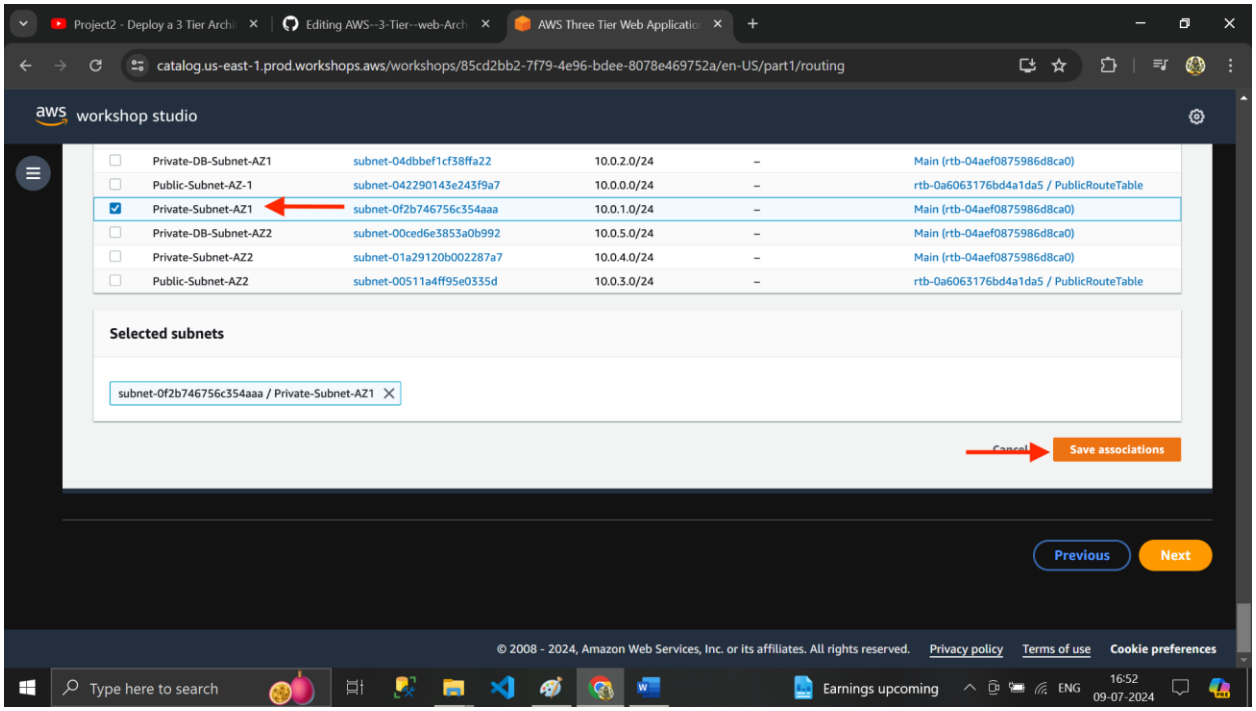





Project2 - Deploy a 3 Tier Archi
Editing AWS--3-Tier--web-Arch
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part1/routing
aws workshop studio
vpc-0d2a264d3ba772982 (awsthreetierworkshop)
Tags
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.
Key
Value - optional
Name
Private-RT-AZ1
Remove
Add new tag
You can add 49 more tags.
Cancel
Create route table
aws
Services
Search for services, features, blogs, docs, and more
[Option+S]
N. California
Admin-OneClick/awsshivs-Isengard @ 1924-6266-9998
VPC
Route tables
rtb-0040b3139bb772f2d
Edit routes
Edit routes
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
Earnings upcoming
ENG
16:52
09-07-2024

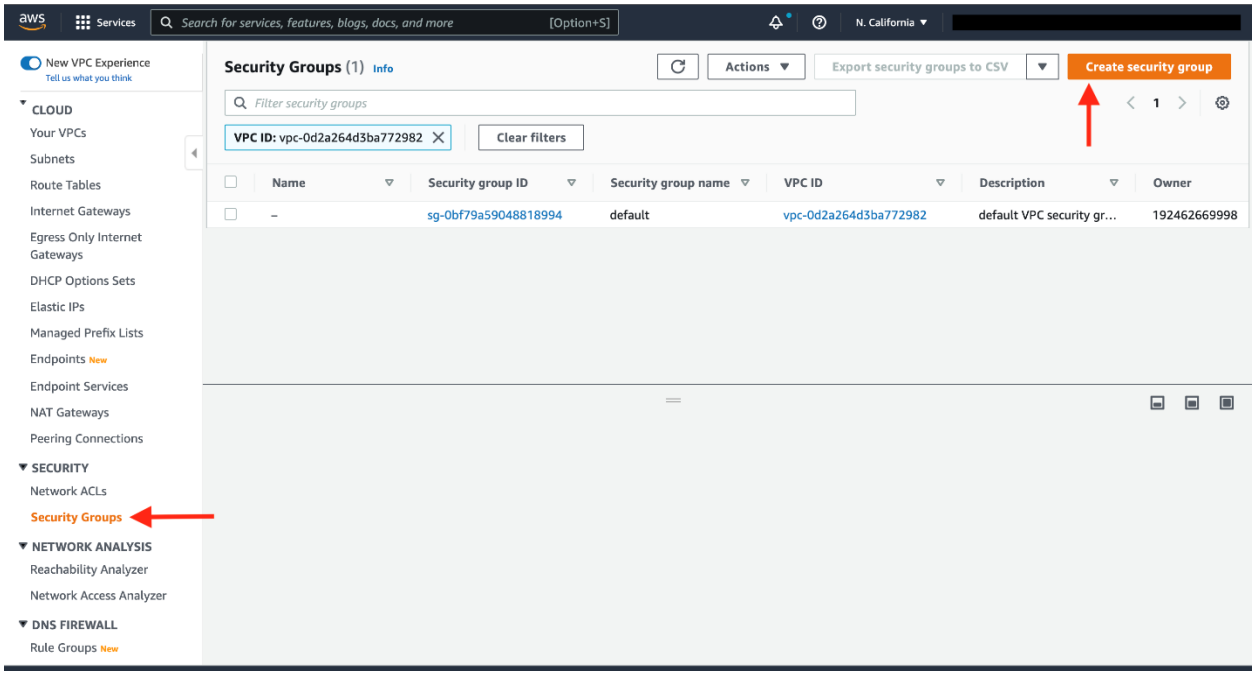

aws
Services
Search for services, features, blogs, docs, and more
[Option+S]
N. California
Admin-OneClick/awsshivs-Isengard @ 1924-6266-9998
VPC
Route tables
rtb-0040b3139bb772f2d
Edit routes
Edit routes
Destination
Target
Status
Propagated
10.0.0.0/16
local
Active
No
0.0.0.0/0
nat-
nat-00f3ade0f7fd91994 (NAT-GW-AZ1)
nat-0bbb7b2cf7f7aafcf (NAT-GW-AZ2)
–
No
Remove
Add route
Cancel
Preview
Save changes



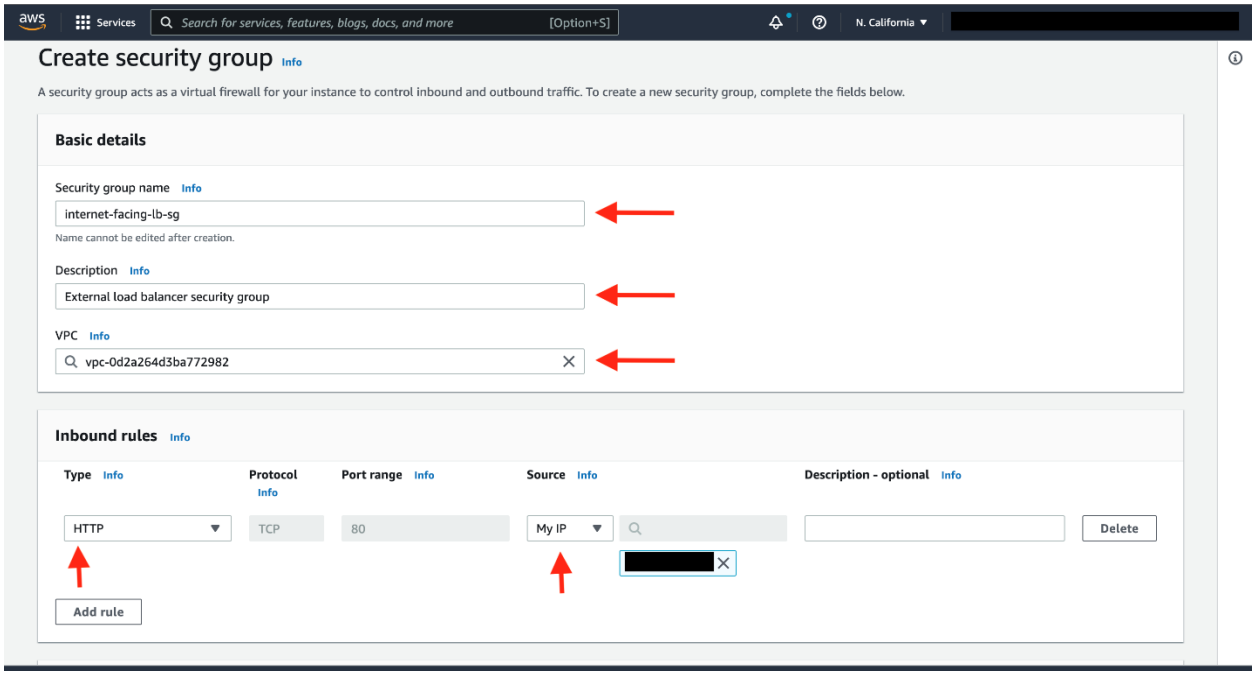


Security Groups

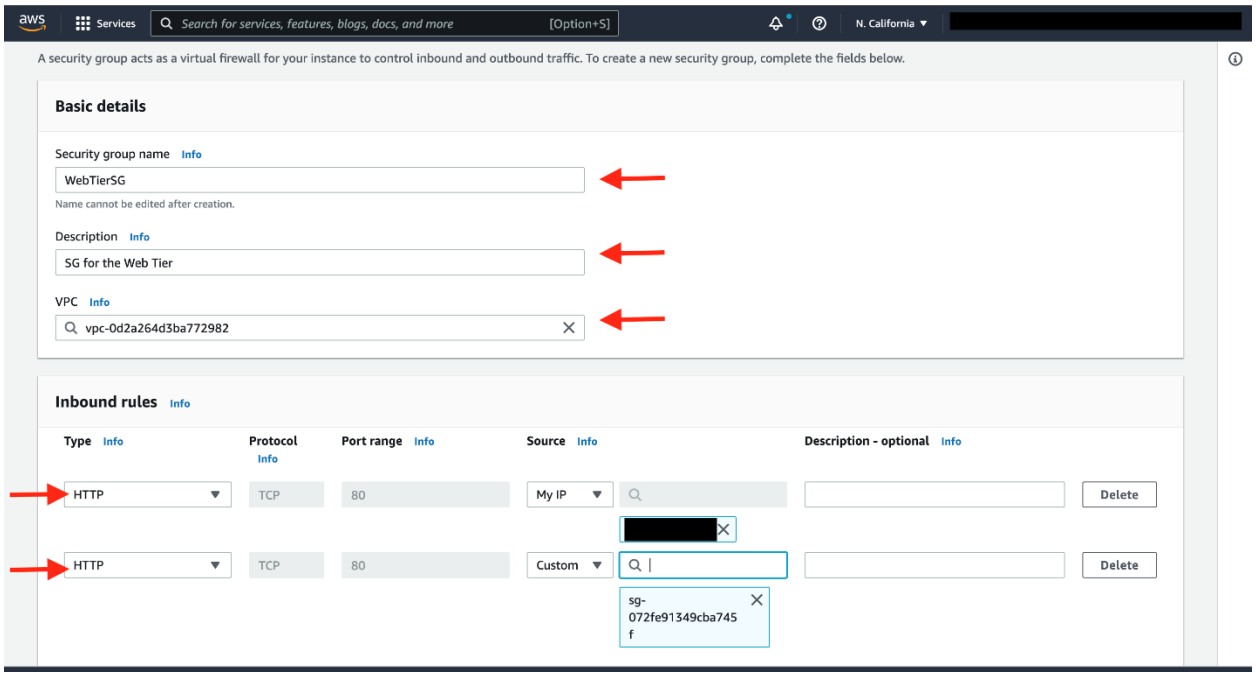
1. Security groups will tighten the rules around which traffic will be allowed to our Elastic Load Balancers and EC2 instances. Navigate to **Security Groups** on the left side of the VPC dashboard, under **Security**.



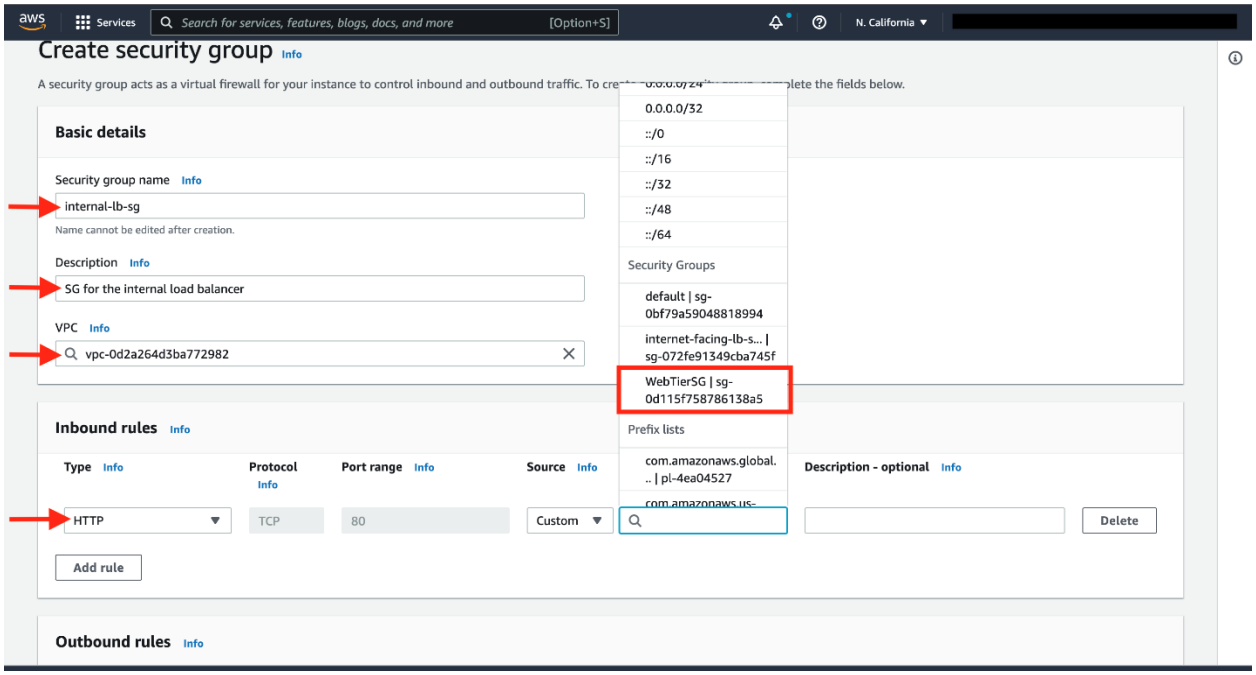
2. The first security group you'll create is for the public, **internet facing** load balancer. After typing a name and description, add an inbound rule to allow **HTTP** type traffic for your **IP**.



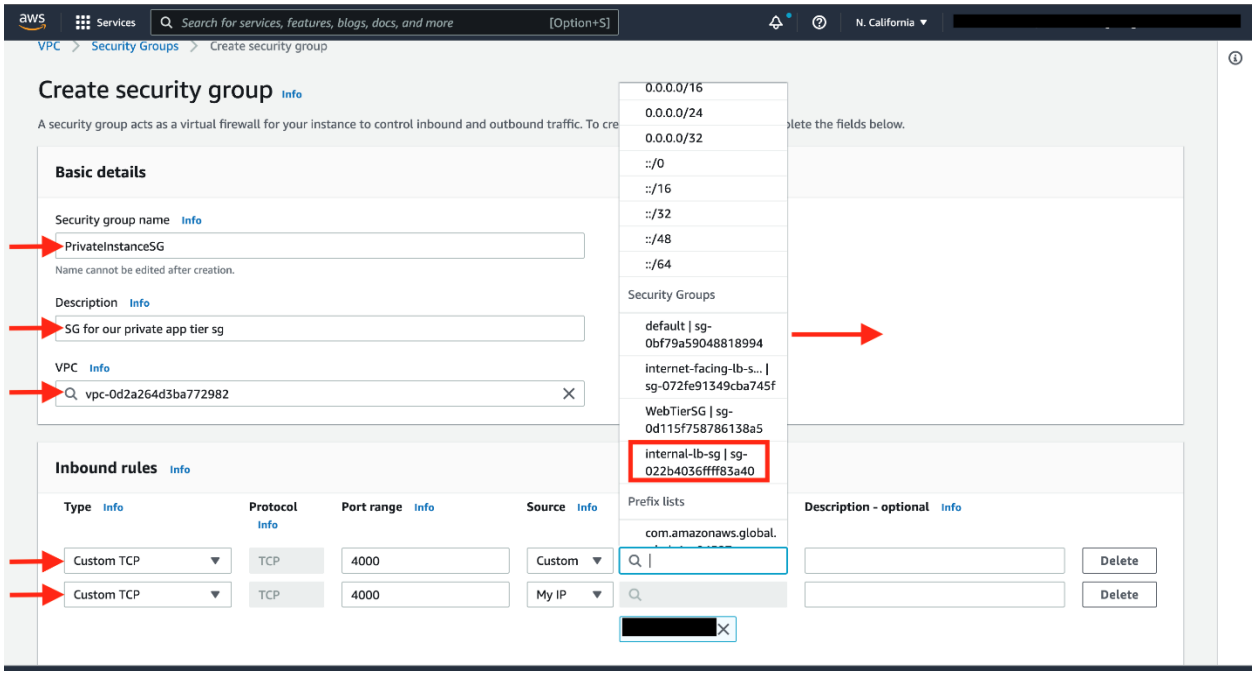
3. The second security group you'll create is for the public instances in the web tier. After typing a name and description, add an inbound rule that allows **HTTP** type traffic from your internet facing load balancer security group you created in the previous step. This will allow traffic from your public facing load balancer to hit your instances. Then, add an additional rule that will allow HTTP type traffic for your IP. This will allow you to access your instance when we test.



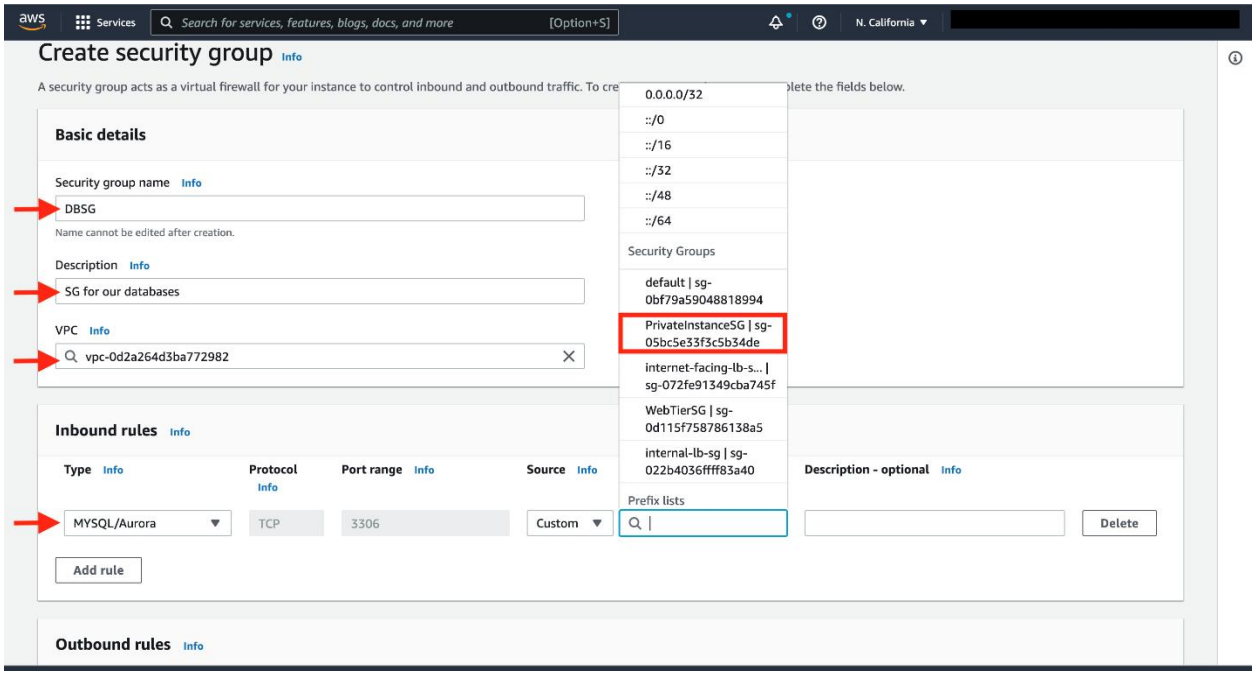
4. # The third security group will be for our internal load balancer. Create this new security group and add an inbound rule that allows **HTTP** type traffic from your public instance security group. This will allow traffic from your web tier instances to hit your internal load balancer.

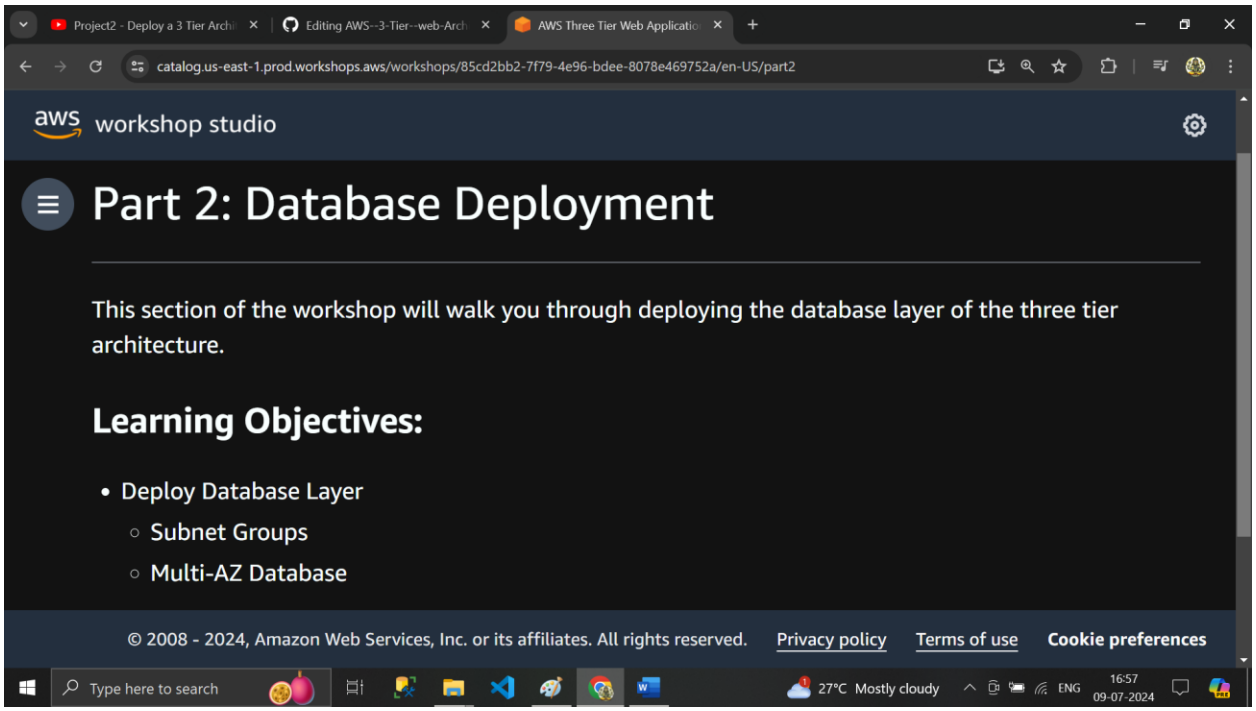


5. The fourth security group we'll configure is for our private instances. After typing a name and description, add an inbound rule that will allow **TCP** type traffic on port **4000** from the **internal load balancer security group** you created in the previous step. This is the port our app tier application is running on and allows our internal load balancer to forward traffic on this port to our private instances. You should also add another route for port **4000** that allows **your IP** for testing.



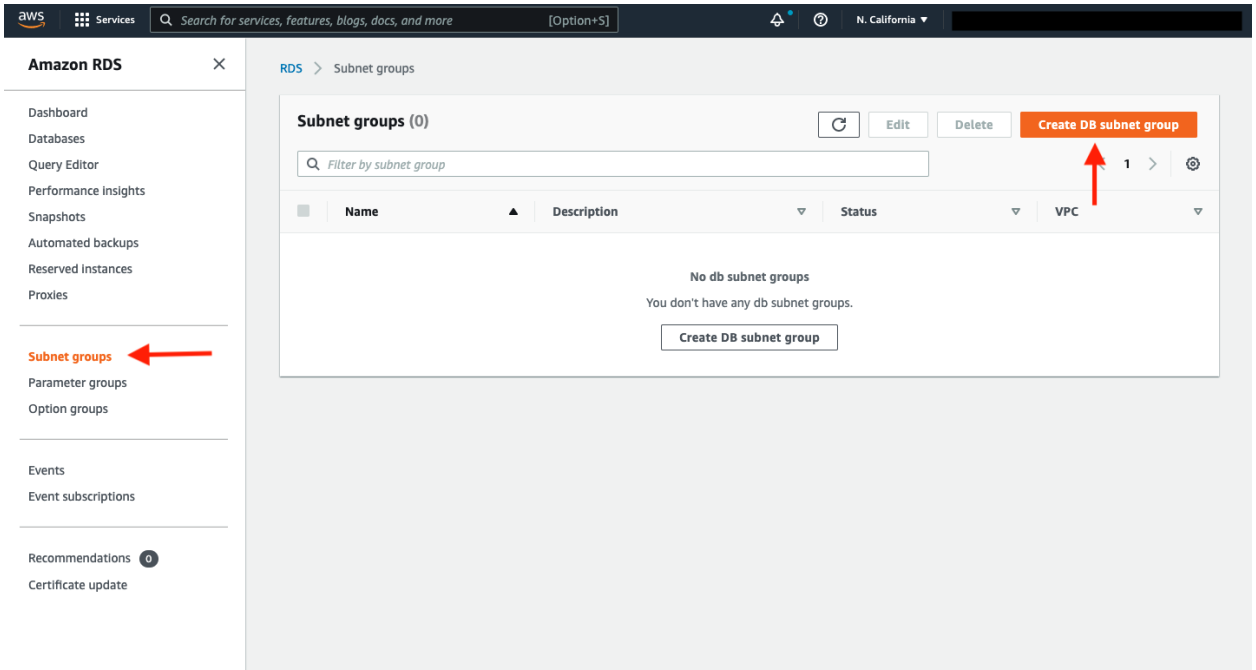
6. The fifth security group we'll configure protects our private database instances. For this security group, add an inbound rule that will allow traffic from the private instance security group to the MYSQL/Aurora port (3306).



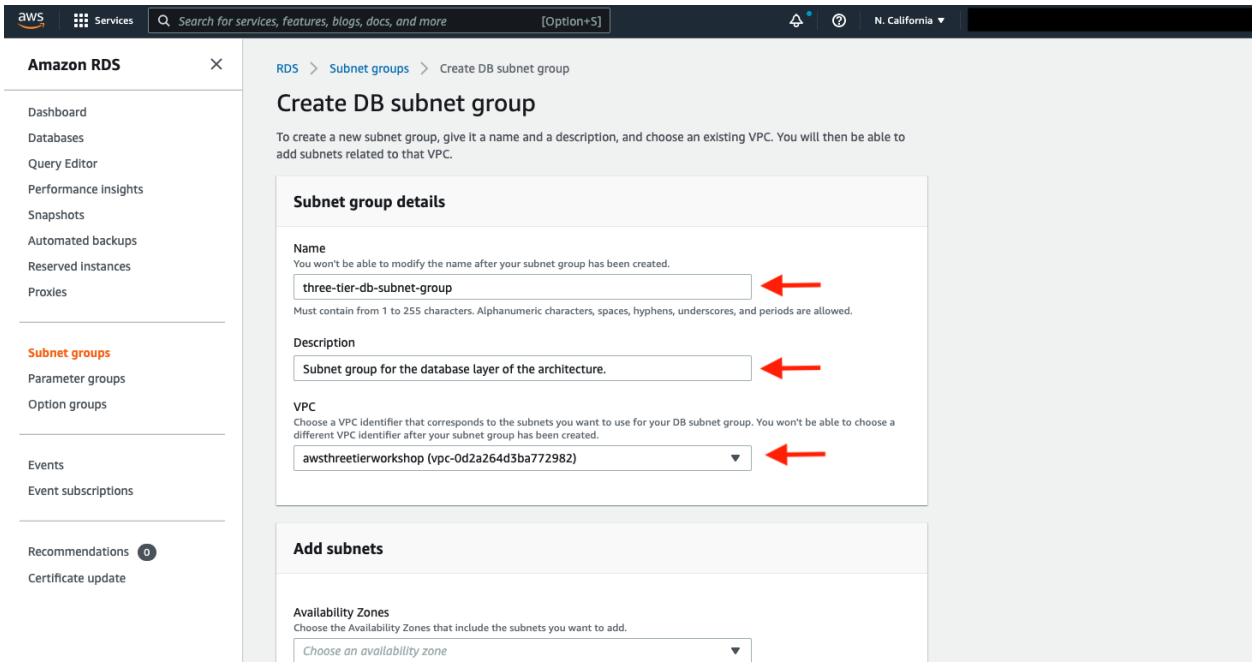


Subnet Groups

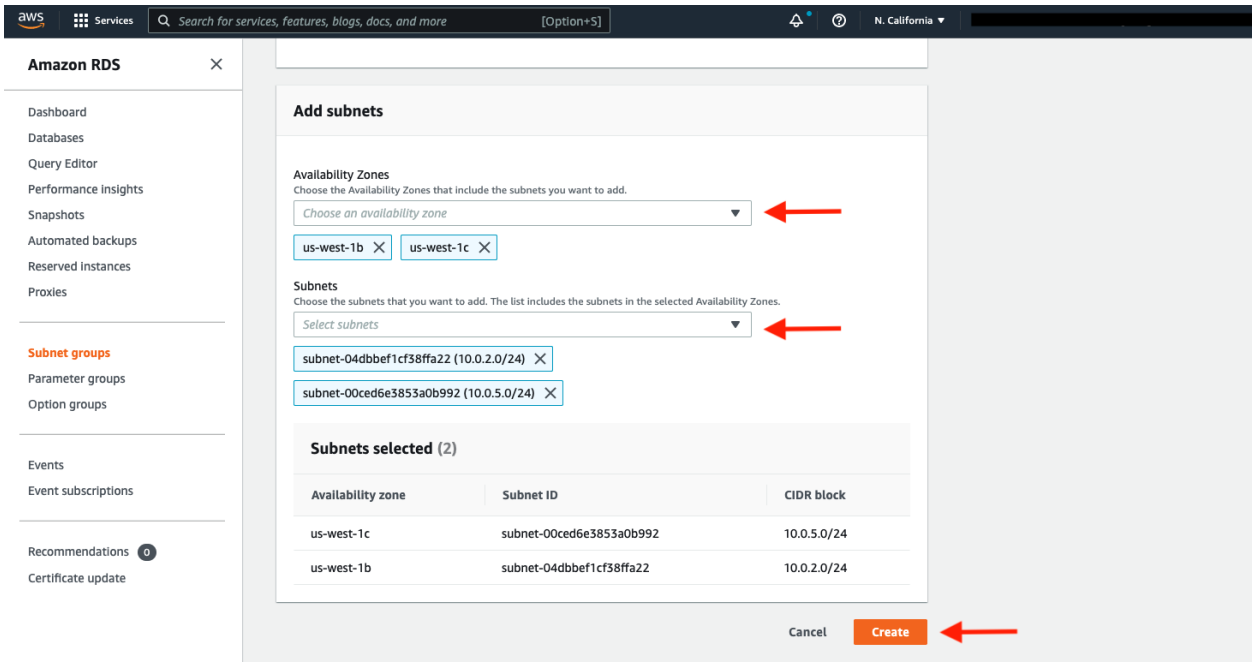
1. Navigate to the RDS dashboard in the AWS console and click on **Subnet groups** on the left hand side. Click **Create DB subnet grou**



2. Give your subnet group a name, description, and choose the VPC we created.

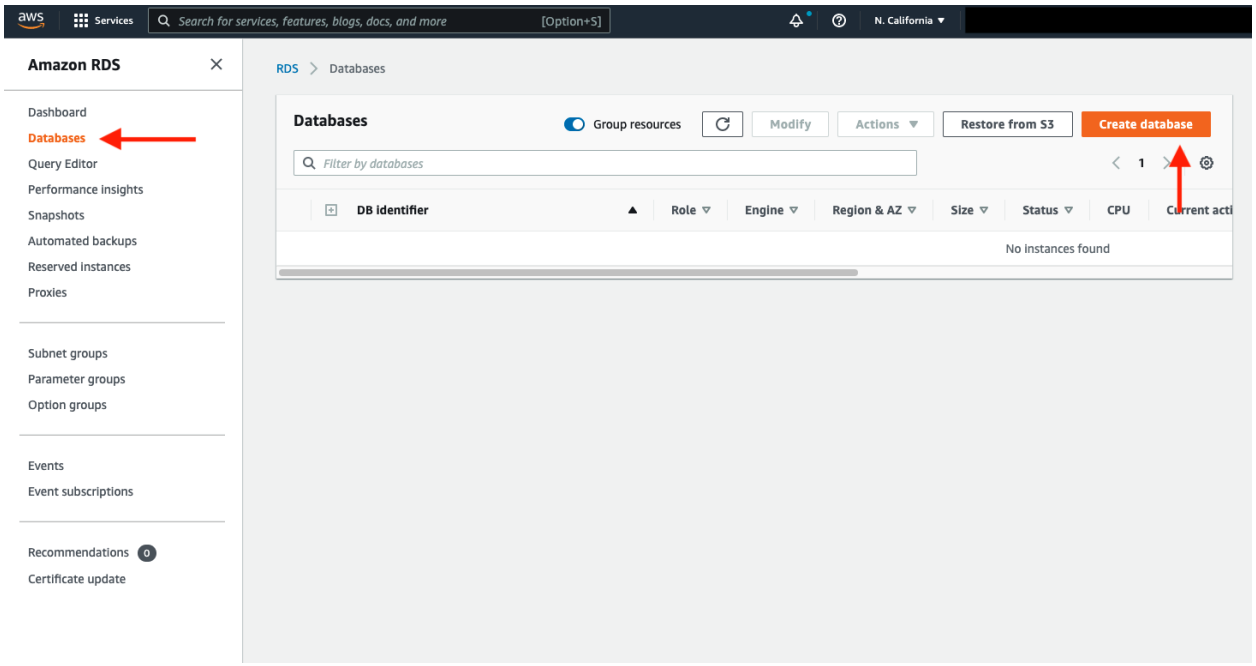


3. When adding subnets, make sure to add the subnets we created in each availability zone specificaly for our database layer. You may have to navigate back to the VPC dashboard and check to make sure you're selecting the correct subnet IDs.

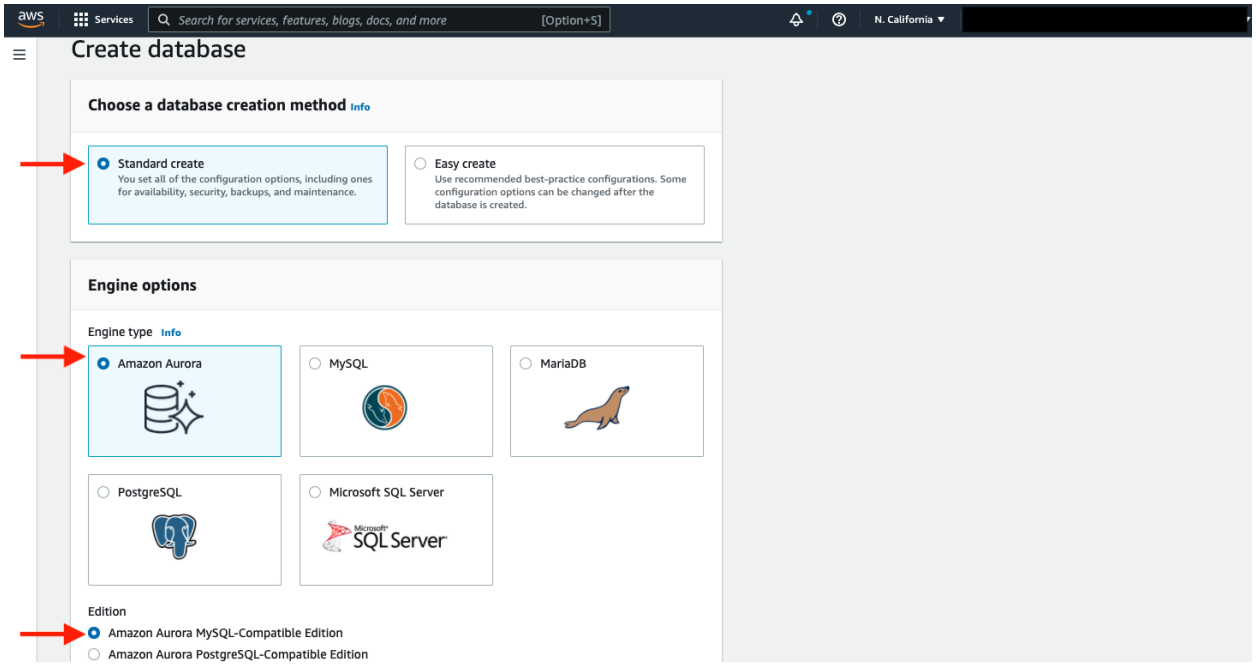


Database Deployment

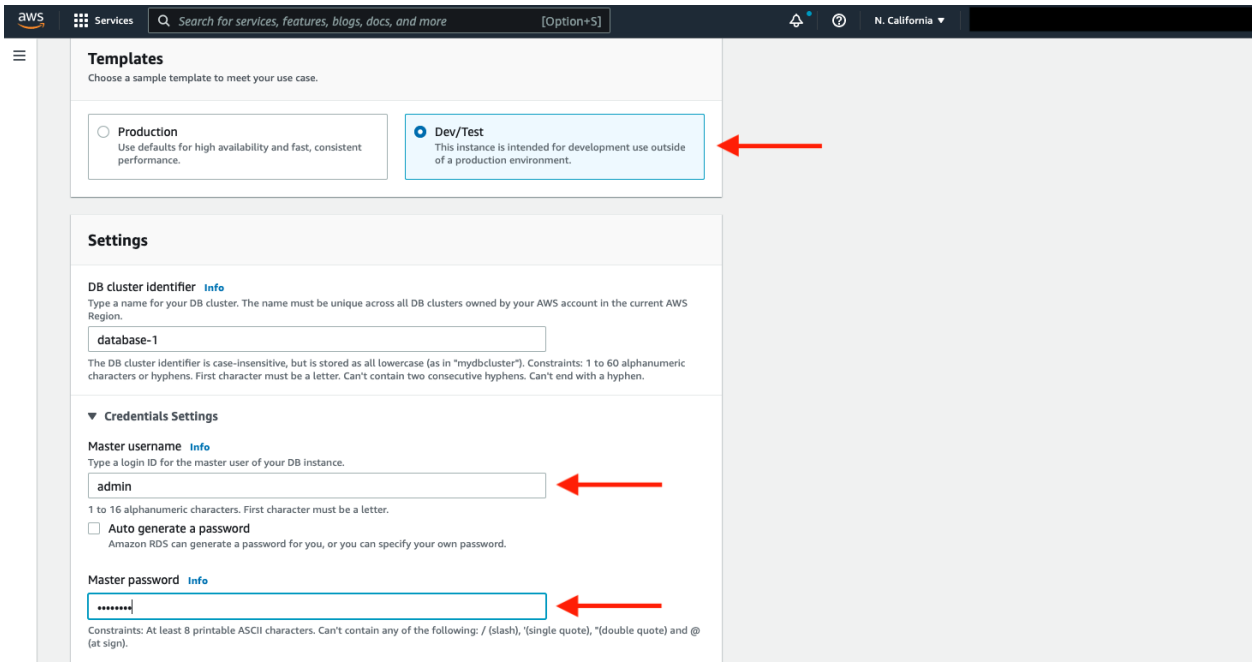
1. Navigate to **Databases** on the left hand side of the RDS dashboard and click **Create database.**



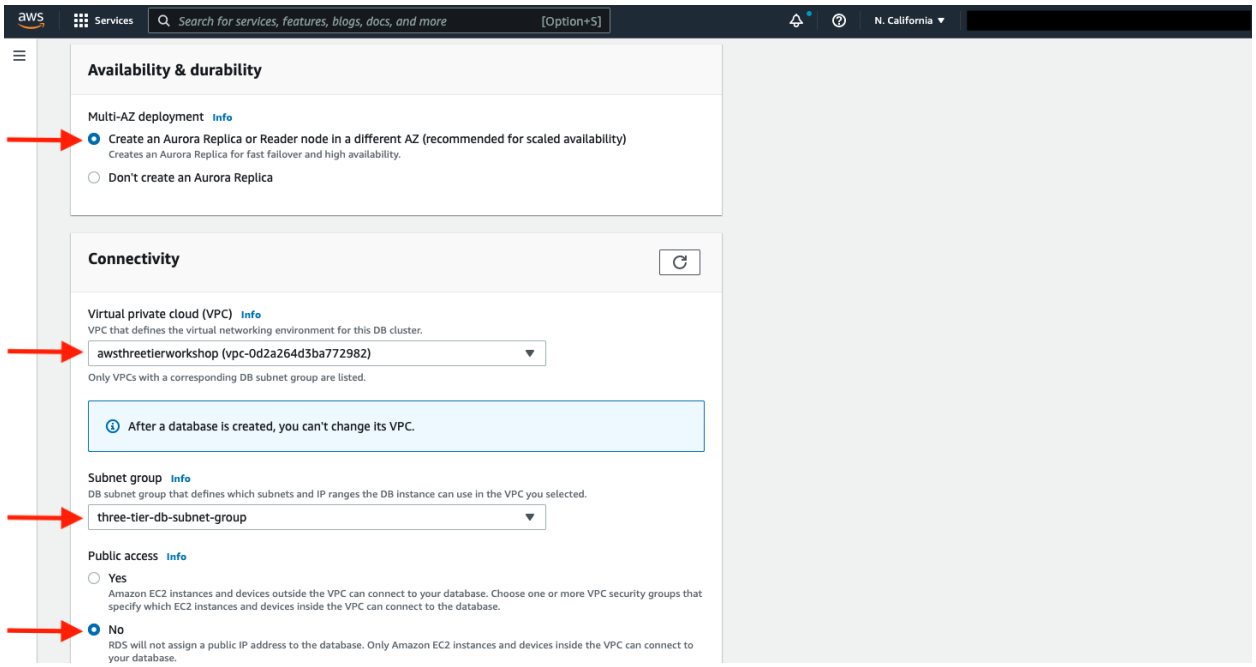
2. We'll now go through several configuration steps. Start with a **Standard create** for this **MySQL-Compatible Amazon Aurora** database. Leave the rest of the defaults in the **Engine options** as default.



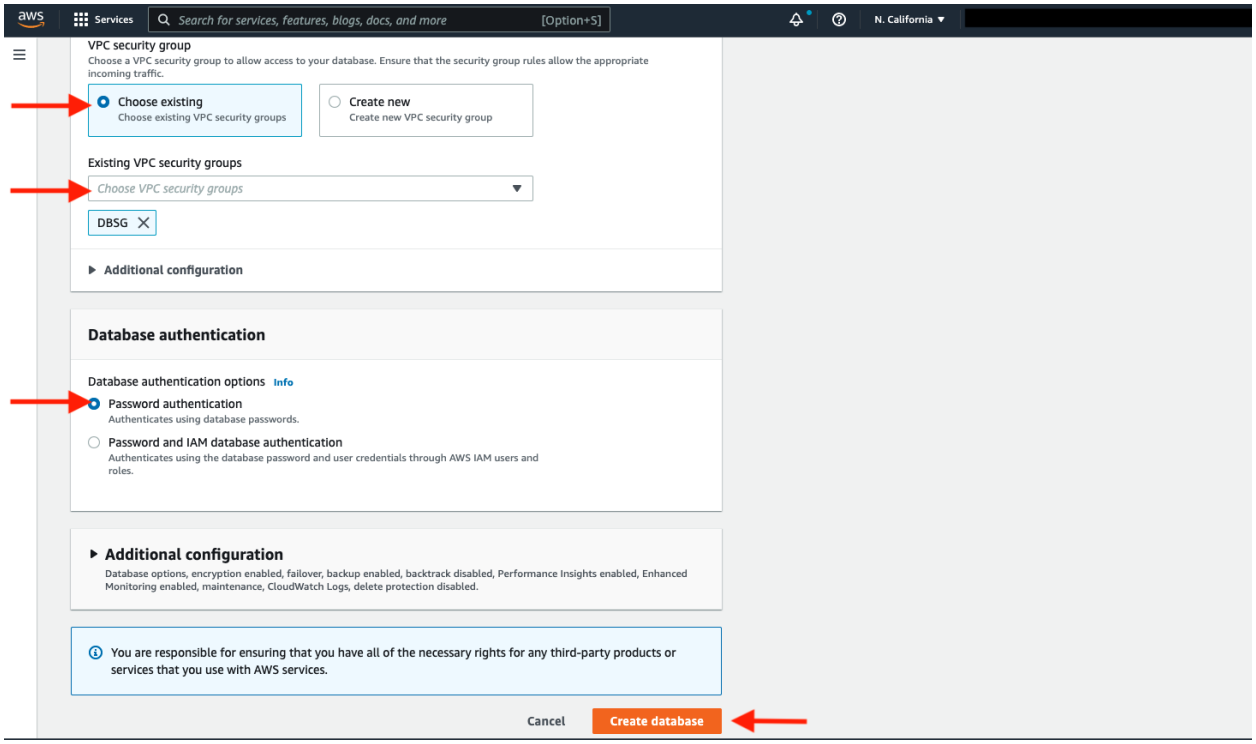
Under the **Templates** section choose **Dev/Test** since this isn't being used for production at the moment. Under **Settings** set a username and password of your choice and note them down since we'll be using password authentication to access our database.



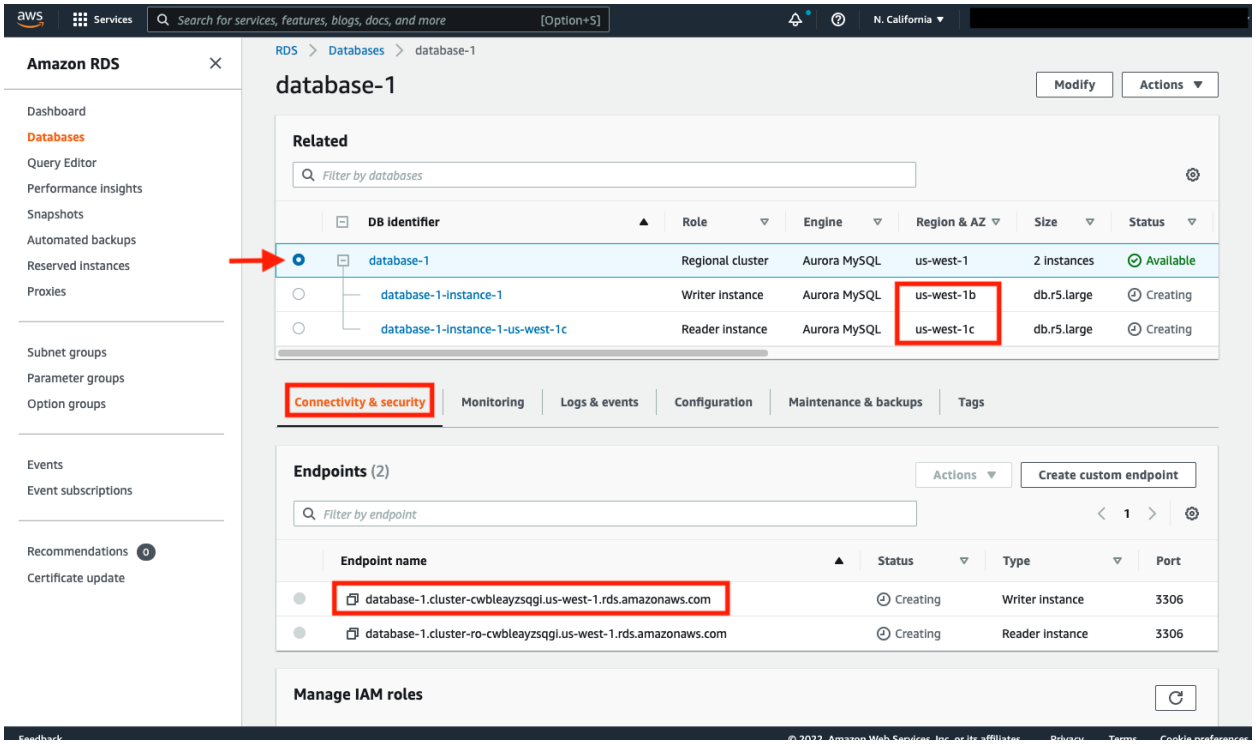
Next, under **Availability and durability** change the option to create an Aurora Replica or reader node in a different availability zone. Under **Connectivity**, set the VPC, choose the subnet group we created earlier, and select no for public access.

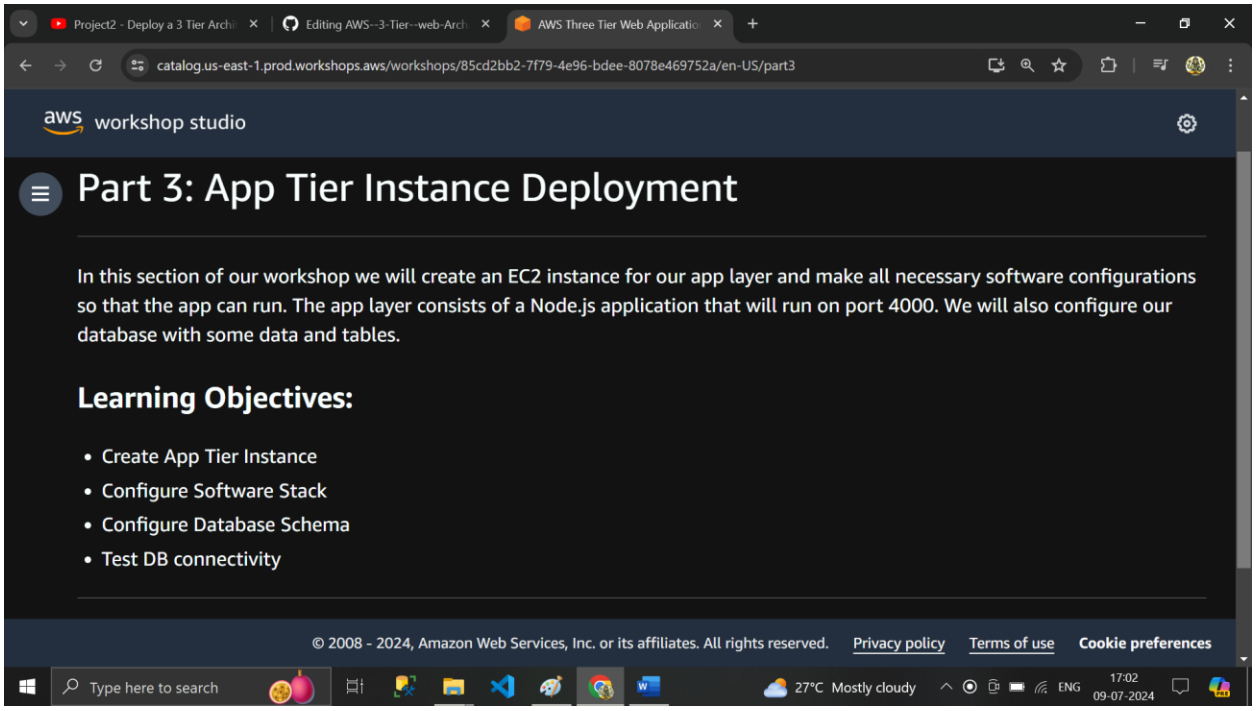


Set the security group we created for the database layer, make sure **password authentication** is selected as our authentication choice, and create the database.



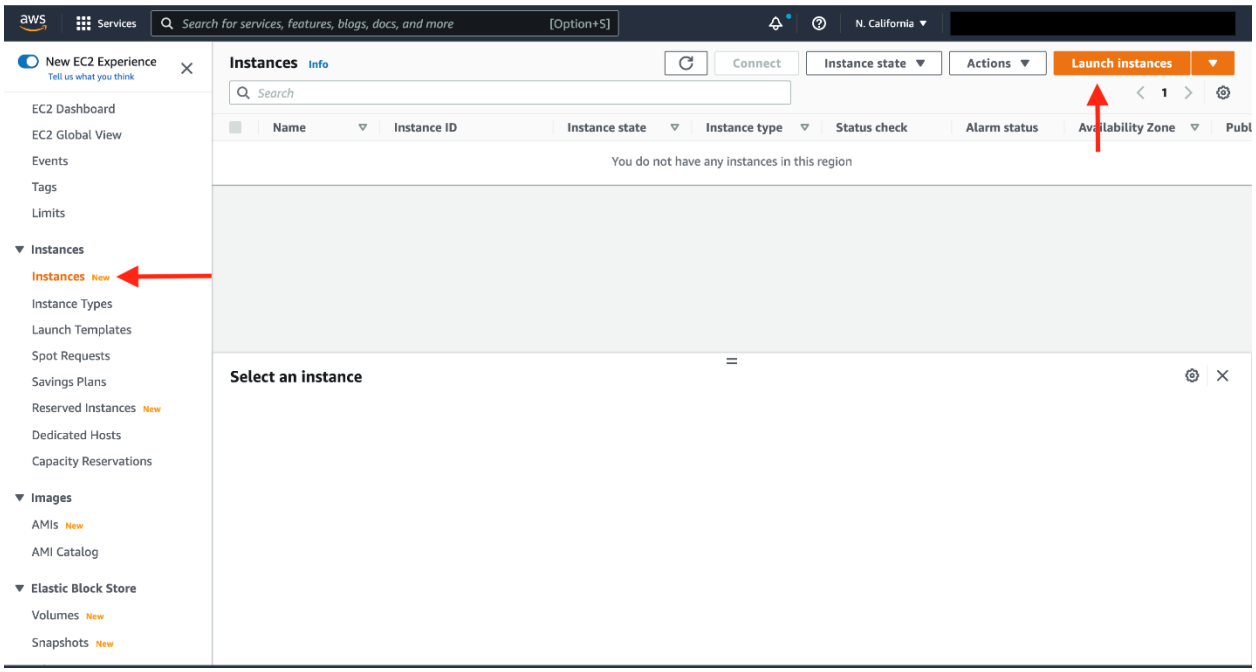
3. When your database is provisioned, you should see a reader and writer instance in the database subnets of each availability zone. Note down the writer endpoint for your database for later use.



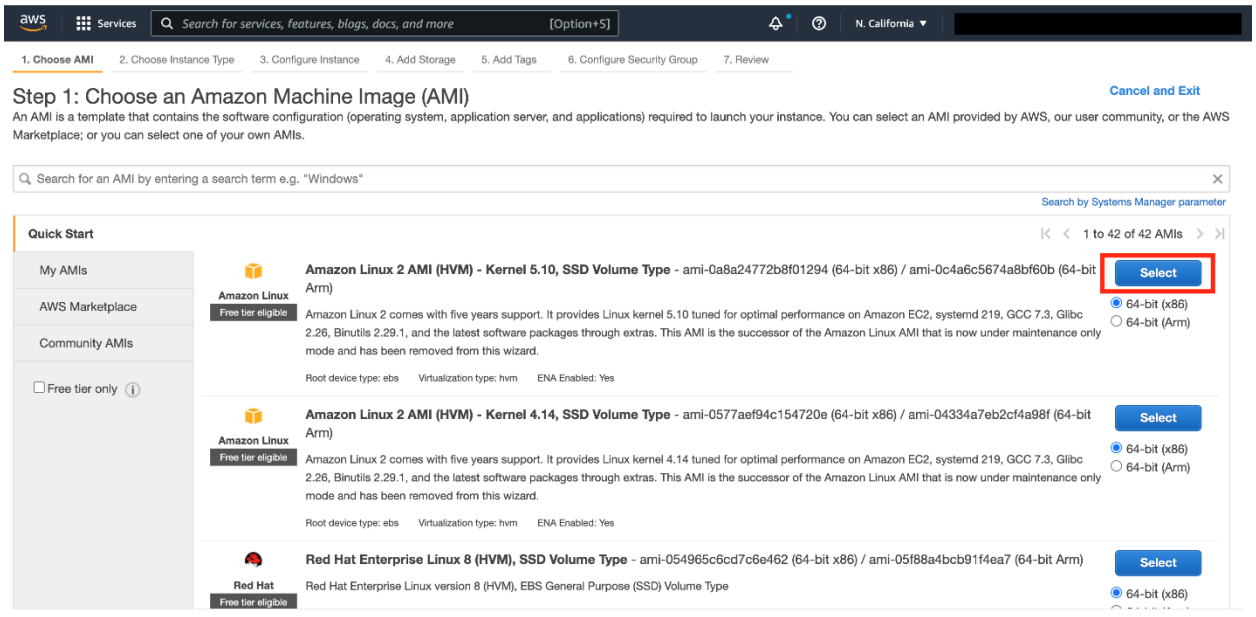


App Instance Deployment

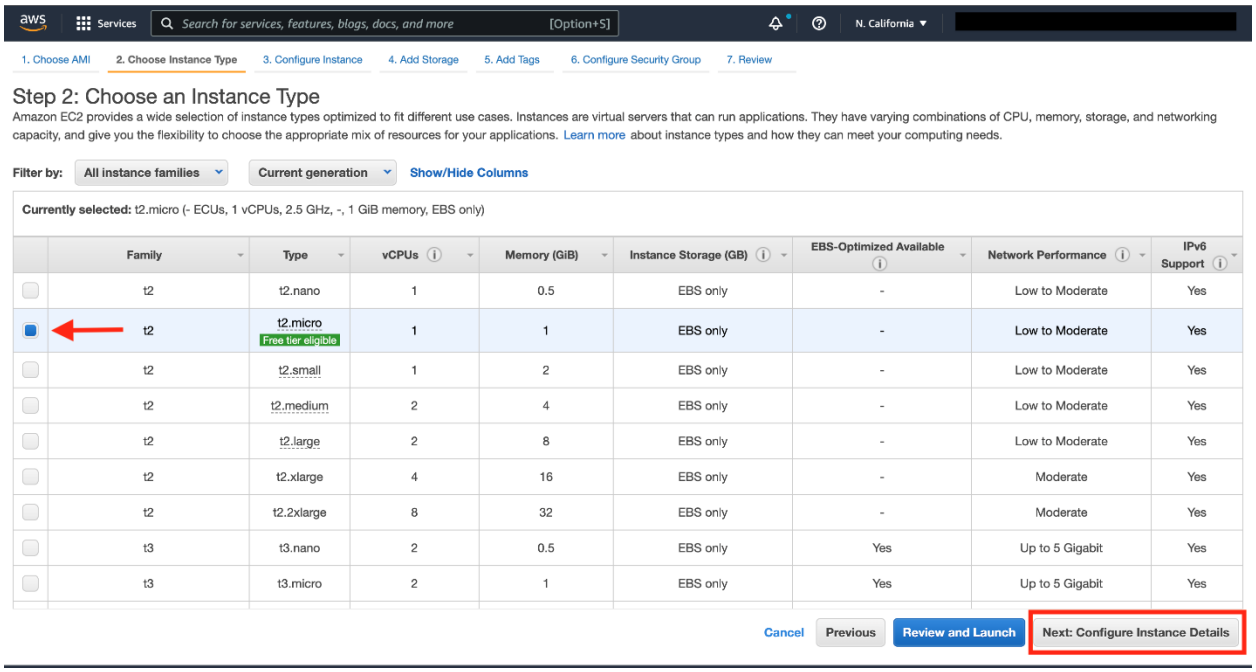
1. Navigate to the EC2 service dashboard and click on **Instances** on the left hand side. Then, click **Launch Instances**.



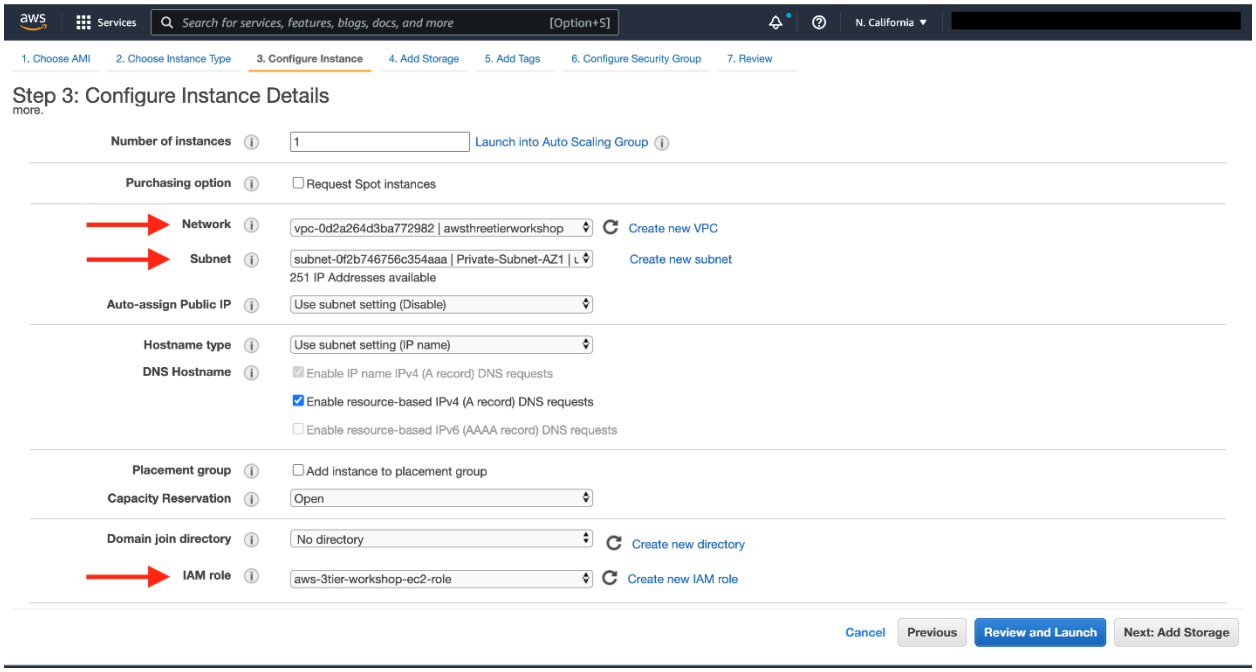
2. Select the first **Amazon Linux 2 AMI**



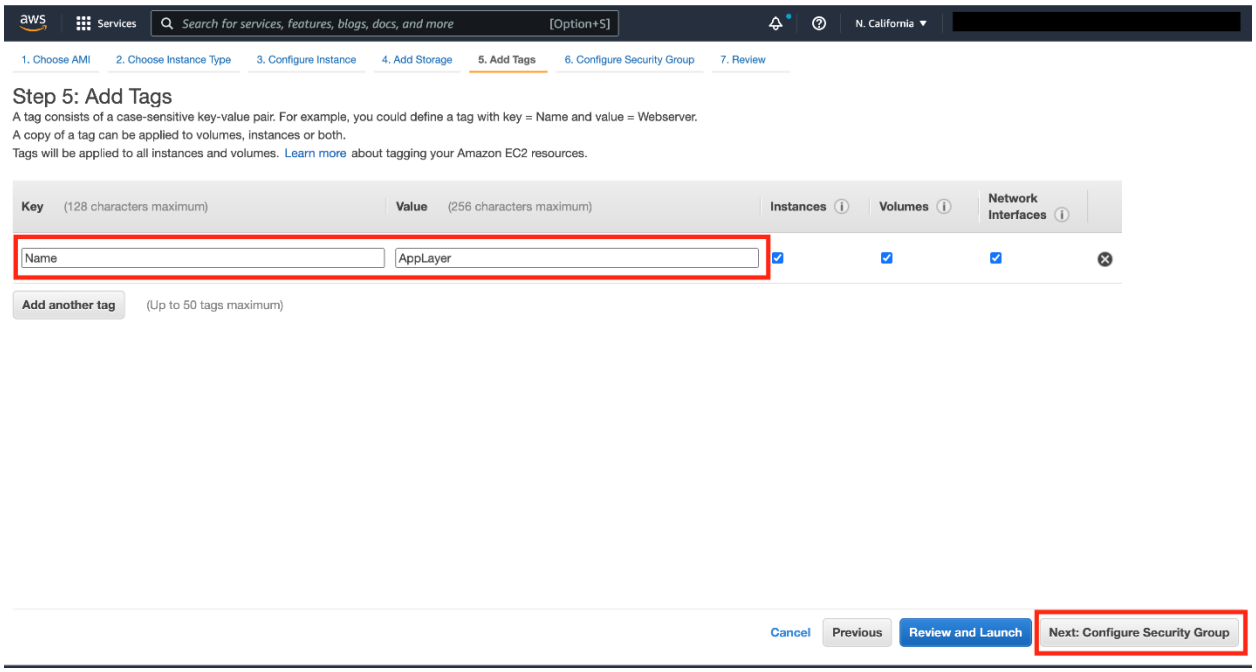
3. We'll be using the free tier eligible **T.2 micro** instance type. Select that and click **Next: Configure Instance Details**.



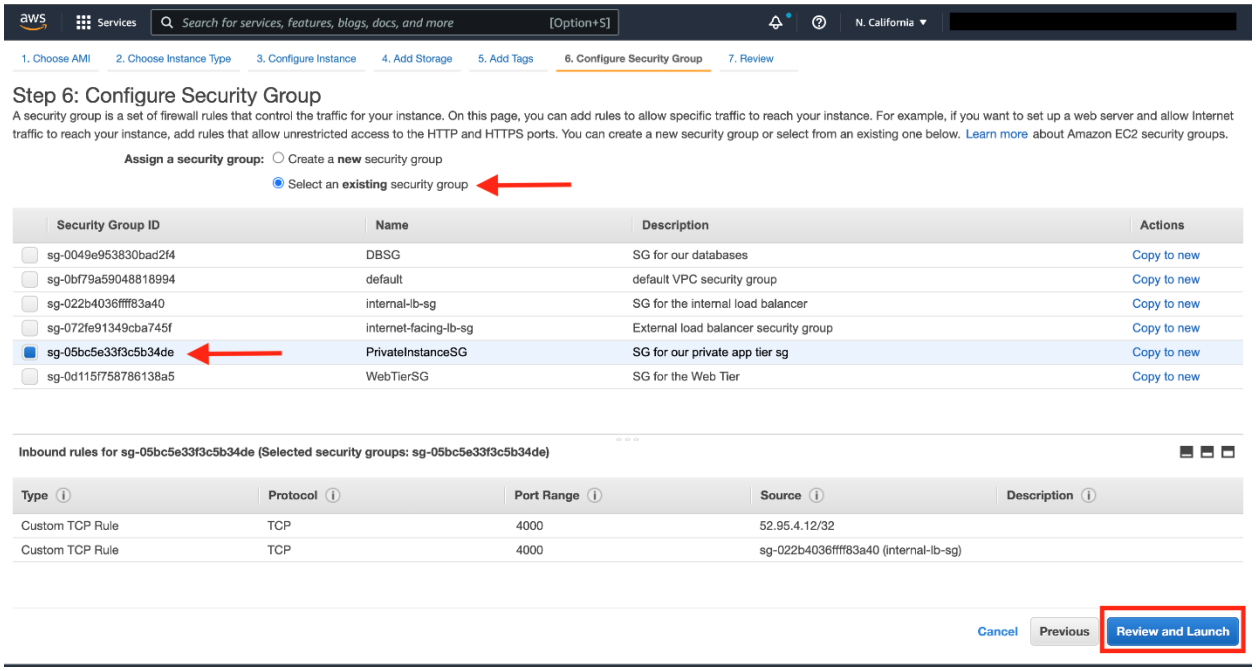
When configuring the instance details, make sure to select to correct **Network**, **subnet**, and **IAM role** we created. Note that this is the app layer, so use one of the private subnets we created for this layer



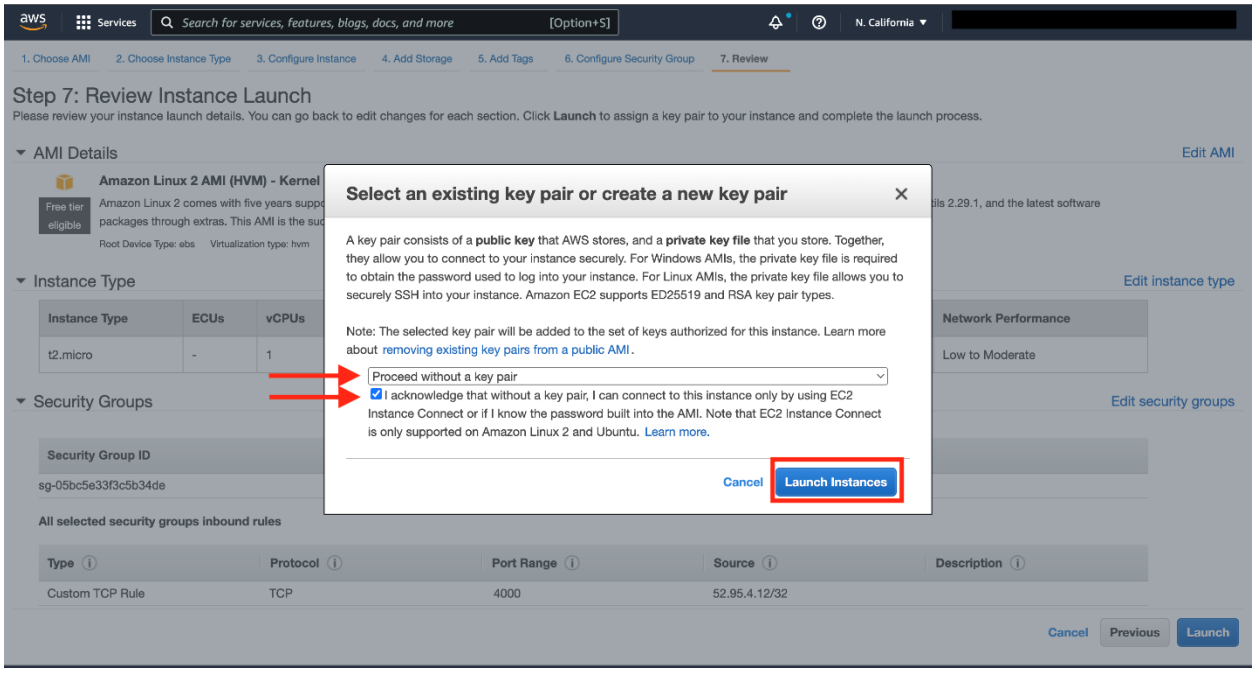
5. We'll be keeping the defaults for storage so click next twice. When you get to the tag screen input a **Name** as a key and call the instance AppLayer. It's a good idea to tag your instances so you can easily keep track of what each instance was created for. Click **Next: Configure Security Group**.



6. Earlier we created a security group for our private app layer instances, so go ahead and select that in this next section. Then click **Review and Launch**. Ignore the warning about connecting to port 22- we don't need to do that.



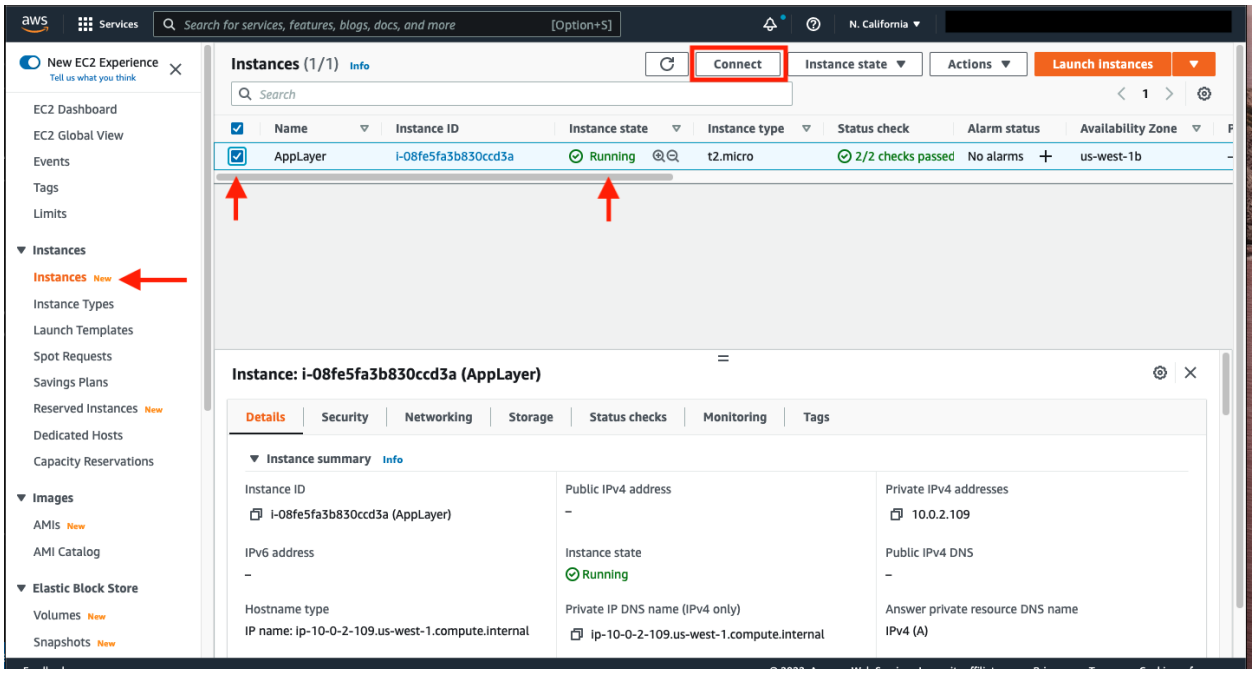
7. When you get to the **Review Instance Launch** page, review the details you configured and click **Launch**. You'll see a pop up about creating a key pair. Since we are using Systems Manager Session Manager to connect to the instance, **proceed without a keypair**. Click **Launch**.

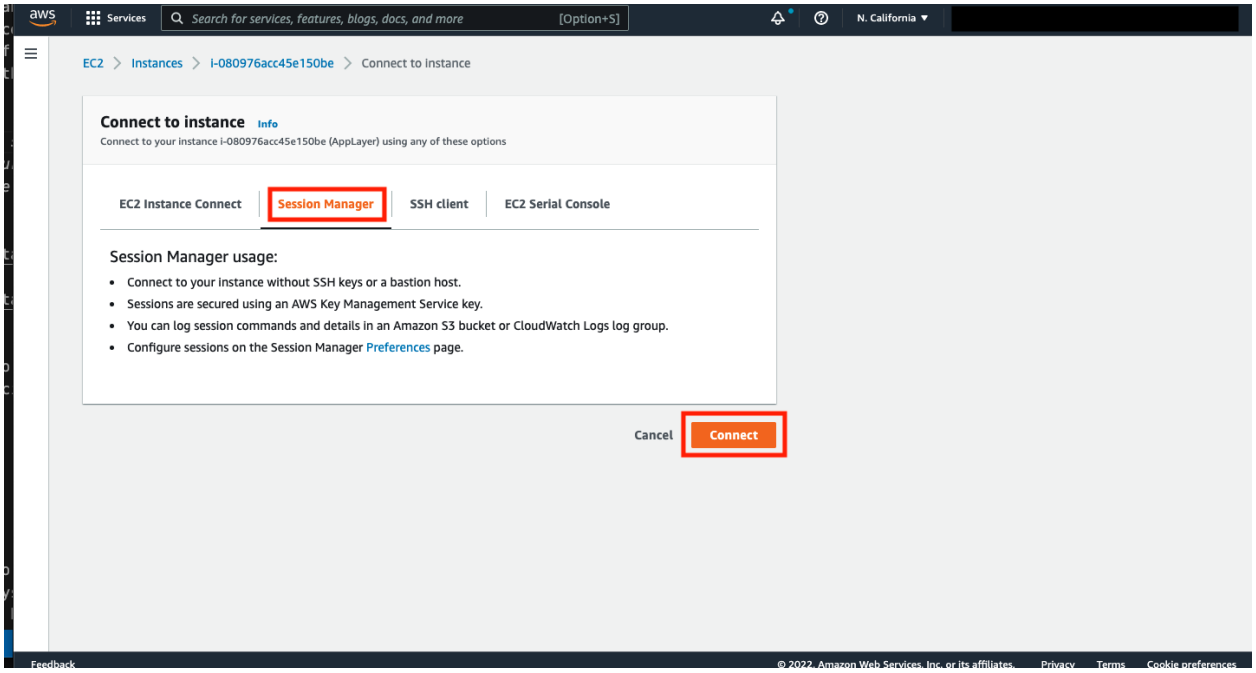


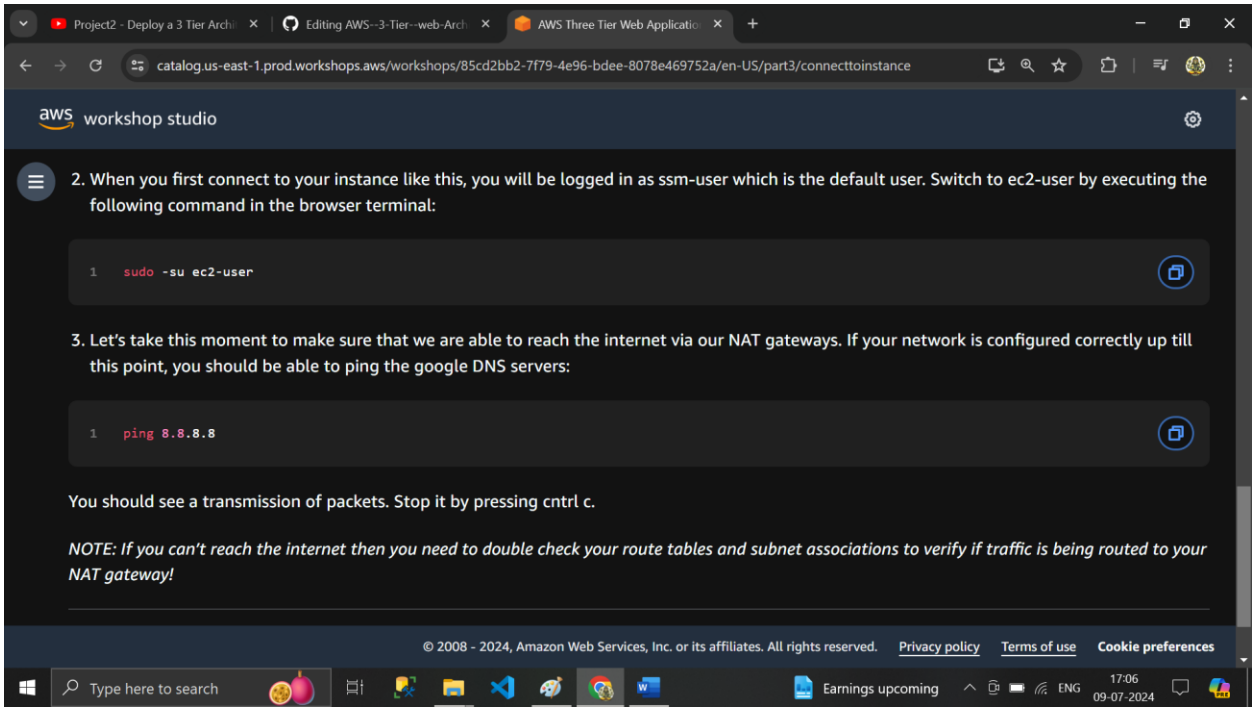
Connect to Instance

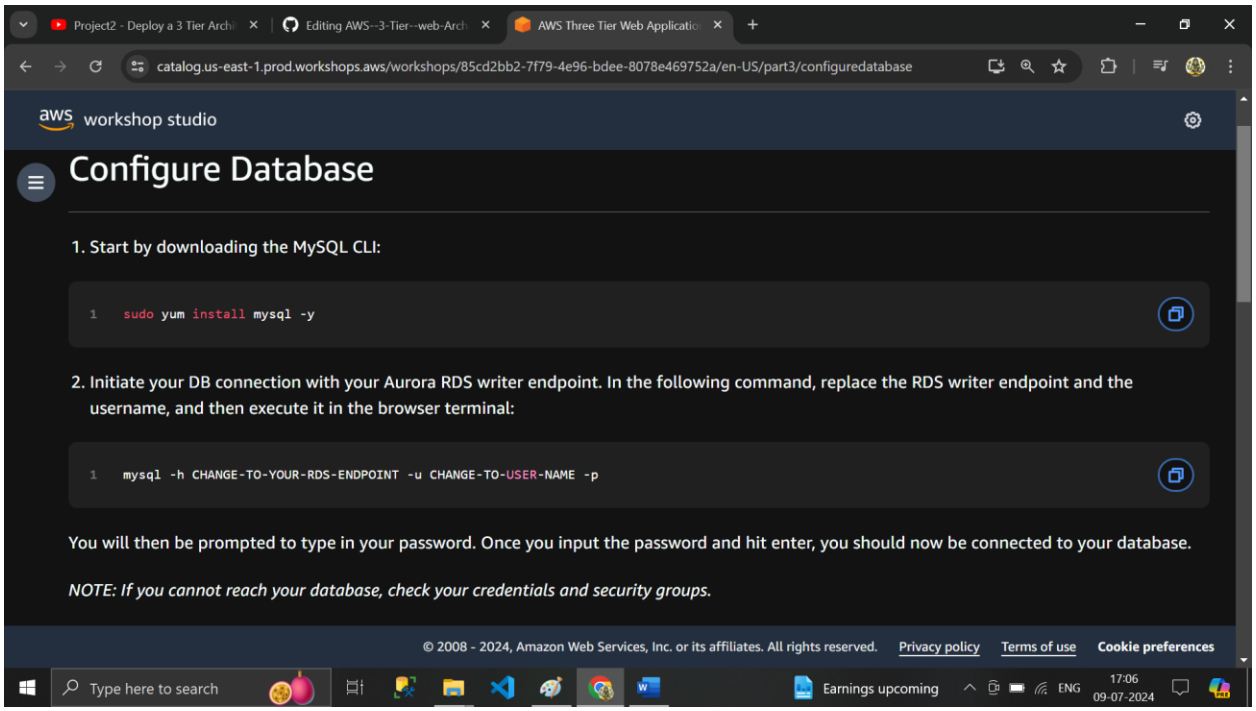
1. Navigate to your list of running EC2 Instances by clicking on **Instances** on the left hand side of the EC2 dashboard. When the instance state is running, connect to your instance by clicking the checkmark box to the left of the instance, and click the connect button on the top right corner of the dashboard.Select the Session Manager tab, and click connect. This will open a new browser tab for you.

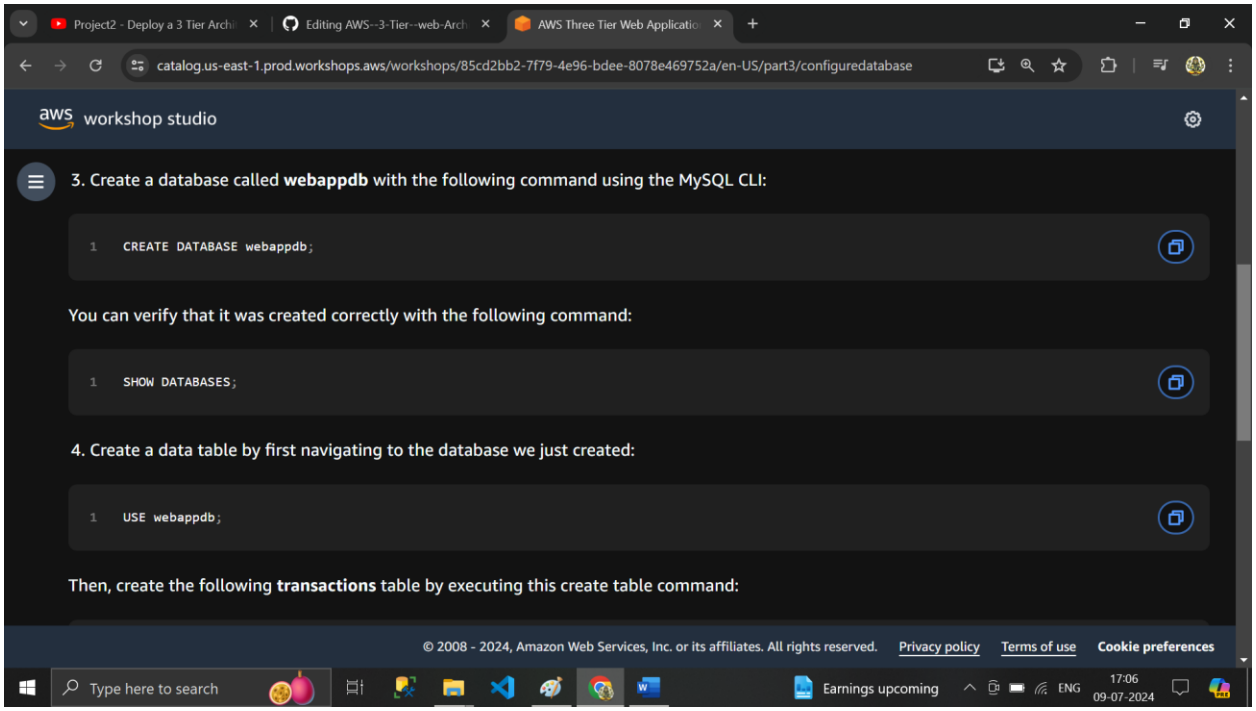
NOTE: If you get a message saying that you cannot connect via session manager, then check that your instances can route to your NAT gateways and verify that you gave the necessary permissions on the IAM role for the Ec2 instance.

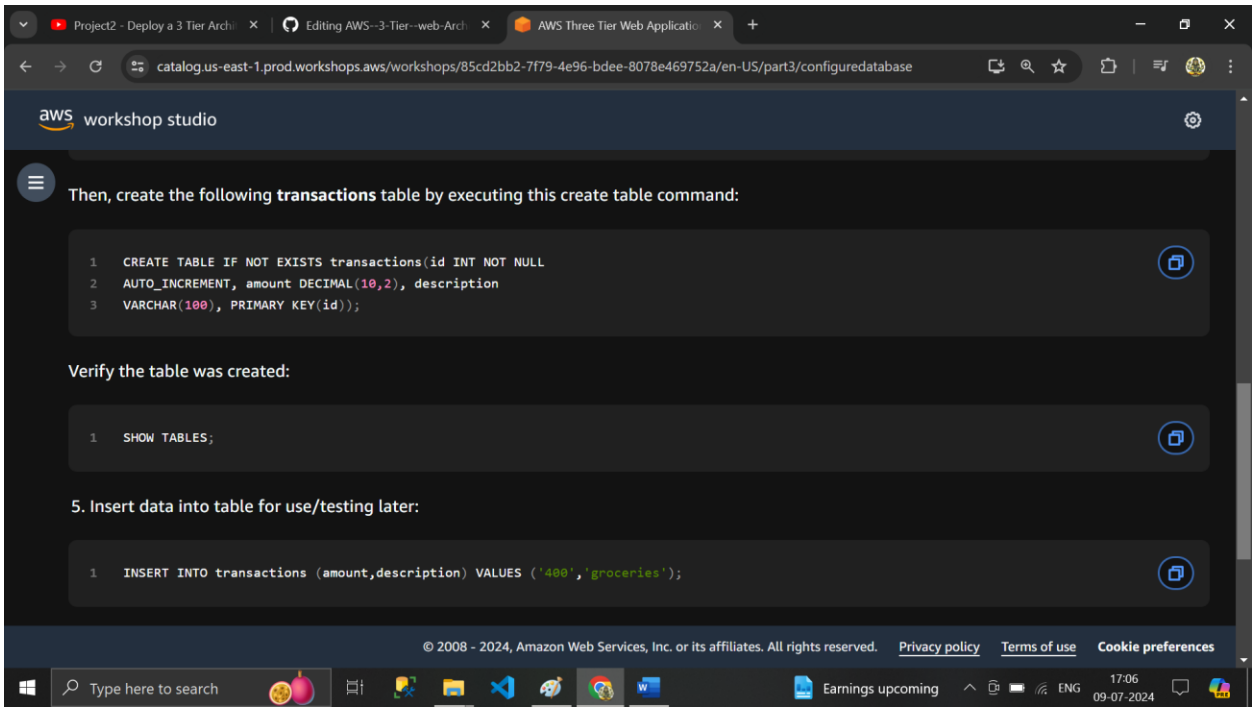


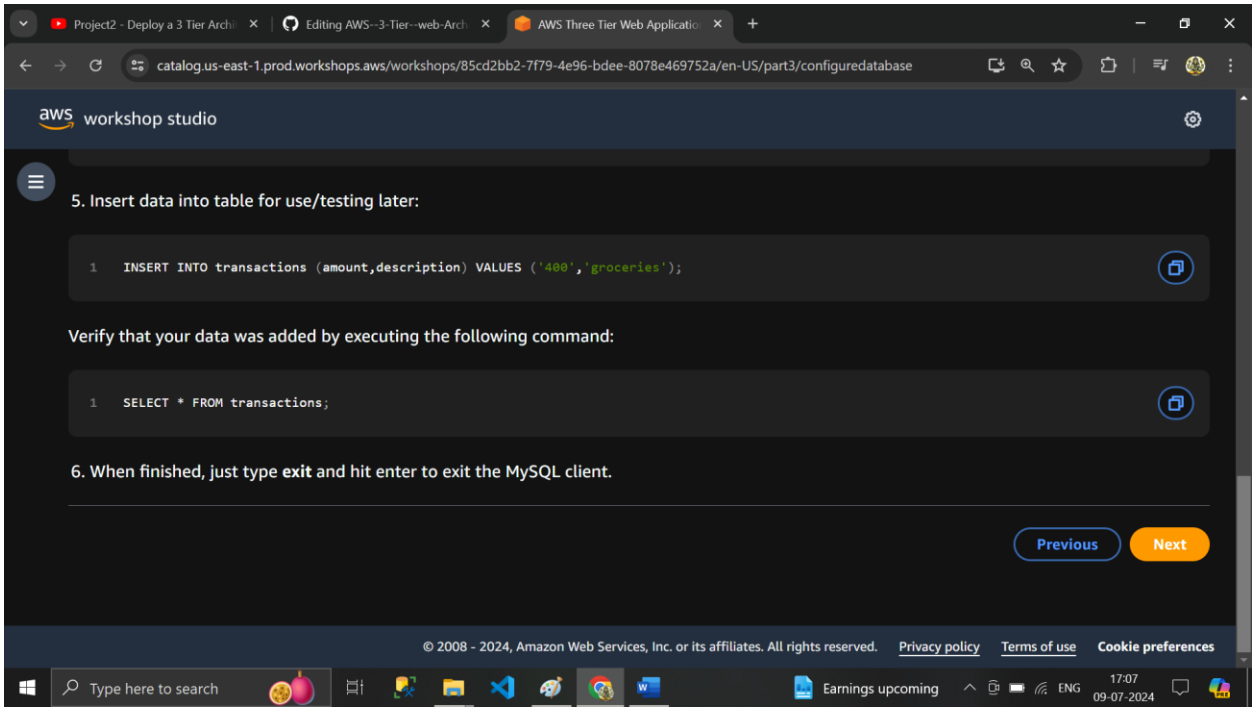

Services
Search for services, features, blogs, docs, and more
[Option+S]
N. California
EC2 > Instances > i-080976acc45e150be > Connect to instance
Connect to instance Info
Connect to your instance i-080976acc45e150be (AppLayer) using any of these options
EC2 Instance Connect
Session Manager
SSH client
EC2 Serial Console
Session Manager usage:
Connect to your instance without SSH keys or a bastion host.
Sessions are secured using an AWS Key Management Service key.
You can log session commands and details in an Amazon S3 bucket or CloudWatch Logs log group.
Configure sessions on the Session Manager Preferences page.
Cancel
Connect
Feedback
© 2022, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Project2 - Deploy a 3 Tier Archi
Editing AWS--3-Tier--web-Arch
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part3/connecttoinstance
aws workshop studio
2. When you first connect to your instance like this, you will be logged in as ssm-user which is the default user. Switch to ec2-user by executing the following command in the browser terminal:
1 sudo -su ec2-user
3. Let's take this moment to make sure that we are able to reach the internet via our NAT gateways. If your network is configured correctly up till this point, you should be able to ping the google DNS servers:
1 ping 8.8.8.8
You should see a transmission of packets. Stop it by pressing cntrl c.
NOTE: If you can't reach the internet then you need to double check your route tables and subnet associations to verify if traffic is being routed to your NAT gateway!
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
Earnings upcoming
ENG
17:06
09-07-2024


Project2 - Deploy a 3 Tier Archi
Editing AWS--3-Tier--web-Arch
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part3/configuredatabase
aws workshop studio
Configure Database
1. Start by downloading the MySQL CLI:
1 sudo yum install mysql -y
2. Initiate your DB connection with your Aurora RDS writer endpoint. In the following command, replace the RDS writer endpoint and the username, and then execute it in the browser terminal:
1 mysql -h CHANGE-TO-YOUR-RDS-ENDPOINT -u CHANGE-TO-USER-NAME -p
You will then be prompted to type in your password. Once you input the password and hit enter, you should now be connected to your database.
NOTE: If you cannot reach your database, check your credentials and security groups.
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
Earnings upcoming
ENG
17:06
09-07-2024


Project2 - Deploy a 3 Tier Archi
Editing AWS--3-Tier--web-Arch
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part3/configuredatabase
aws workshop studio
3. Create a database called webappdb with the following command using the MySQL CLI:
1 CREATE DATABASE webappdb;
You can verify that it was created correctly with the following command:
1 SHOW DATABASES;
4. Create a data table by first navigating to the database we just created:
1 USE webappdb;
Then, create the following transactions table by executing this create table command:
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
Earnings upcoming
ENG
17:06
09-07-2024

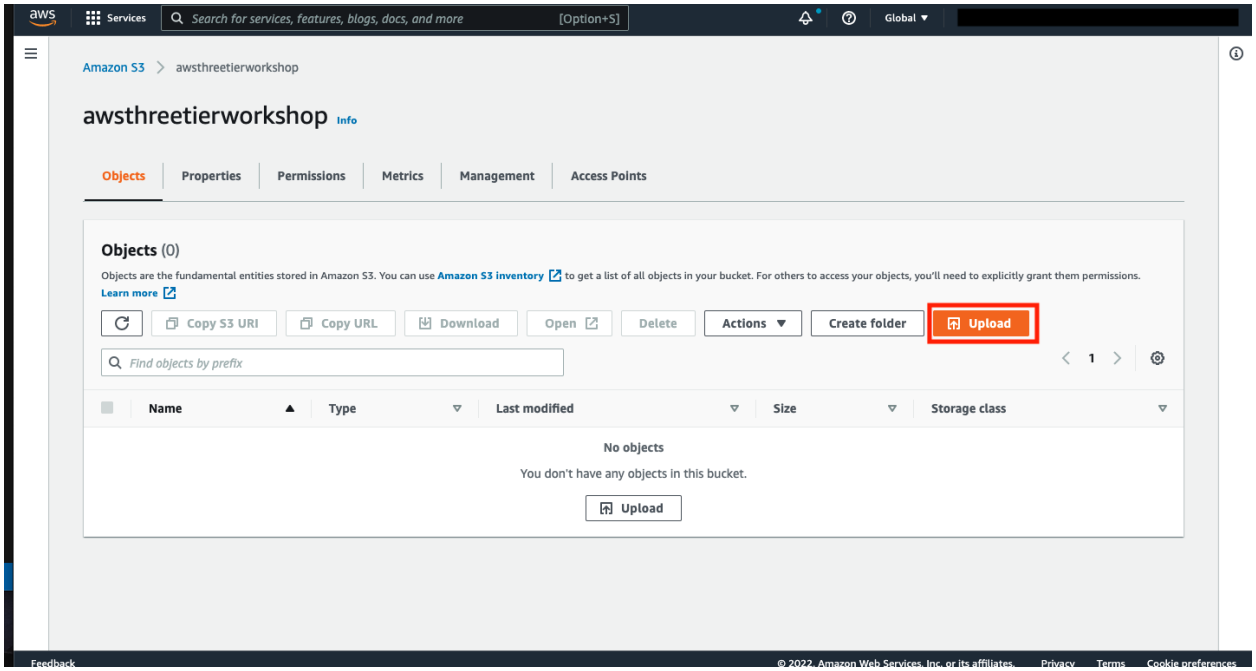

Project2 - Deploy a 3 Tier Archi
Editing AWS--3-Tier--web-Arch
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part3/configuredatabase
aws workshop studio
Then, create the following transactions table by executing this create table command:
CREATE TABLE IF NOT EXISTS transactions(id INT NOT NULL
AUTO_INCREMENT, amount DECIMAL(10,2), description
VARCHAR(100), PRIMARY KEY(id));
Verify the table was created:
SHOW TABLES;
5. Insert data into table for use/testing later:
INSERT INTO transactions (amount,description) VALUES ('400','groceries');
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
Earnings upcoming
ENG
17:06
09-07-2024

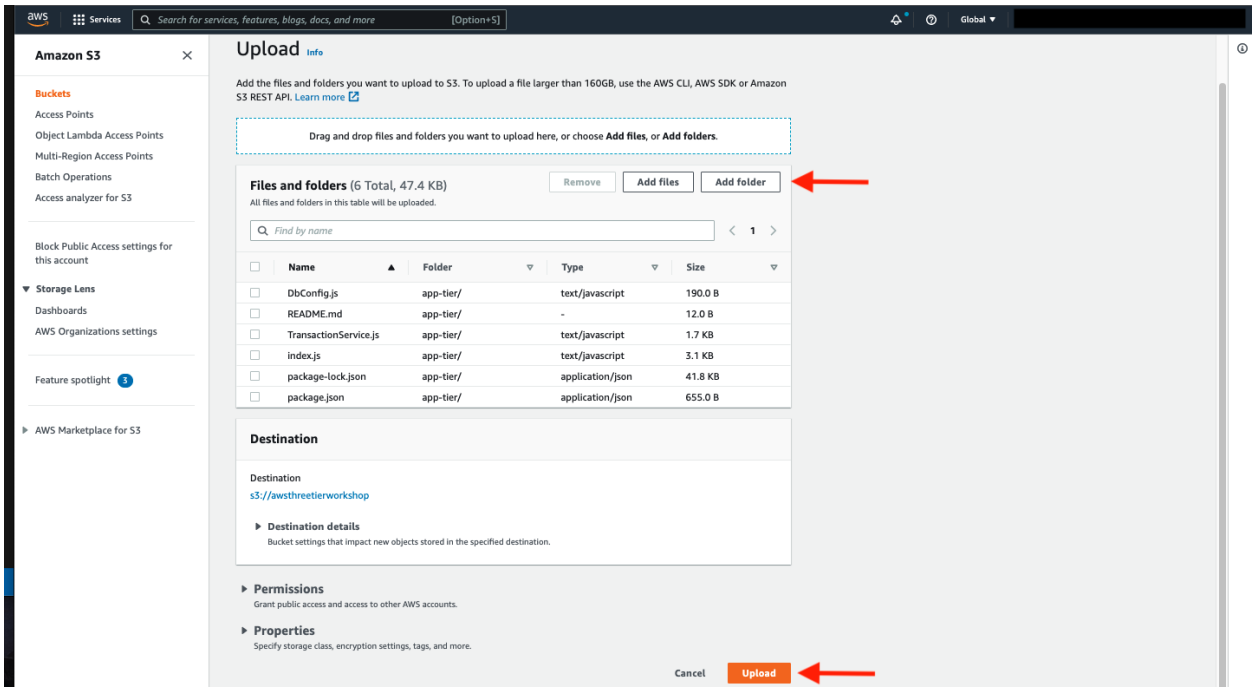

Project2 - Deploy a 3 Tier Archi
Editing AWS--3-Tier--web-Arch
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part3/configuredatabase
aws workshop studio
5. Insert data into table for use/testing later:
INSERT INTO transactions (amount,description) VALUES ('400','groceries');
Verify that your data was added by executing the following command:
SELECT * FROM transactions;
6. When finished, just type exit and hit enter to exit the MySQL client.
Previous
Next
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
Earnings upcoming
ENG
17:07
09-07-2024

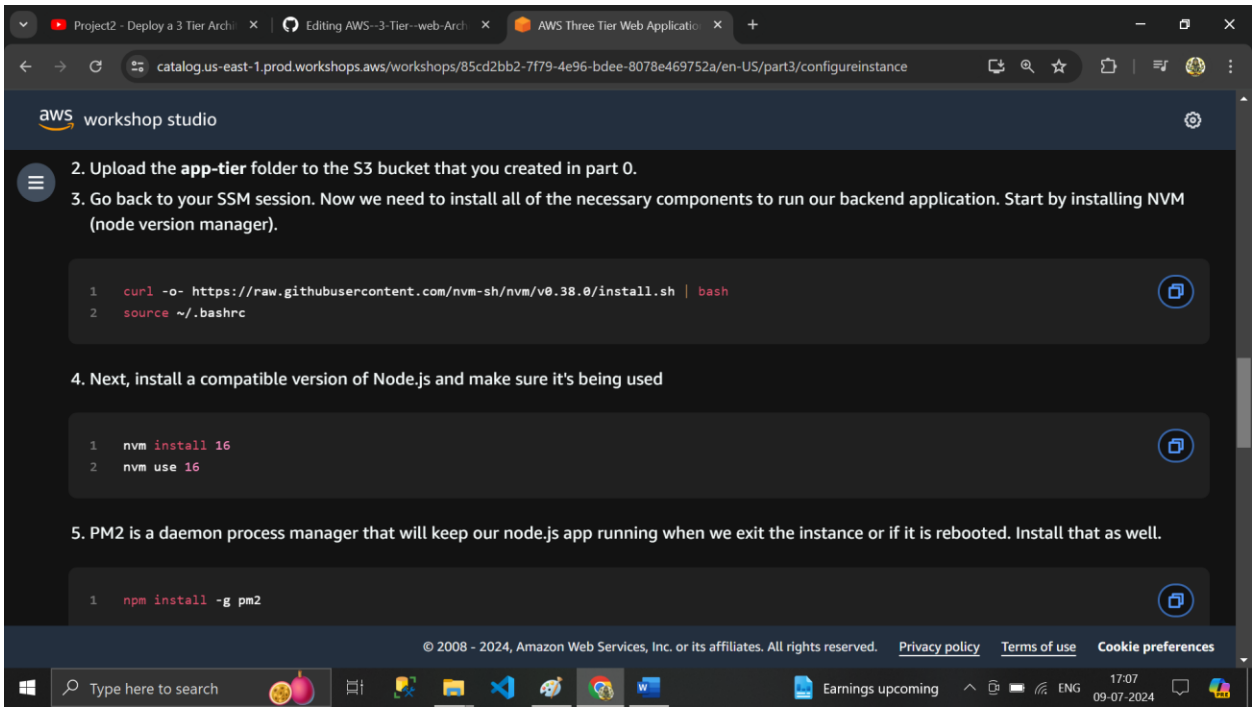
Configure App Instance

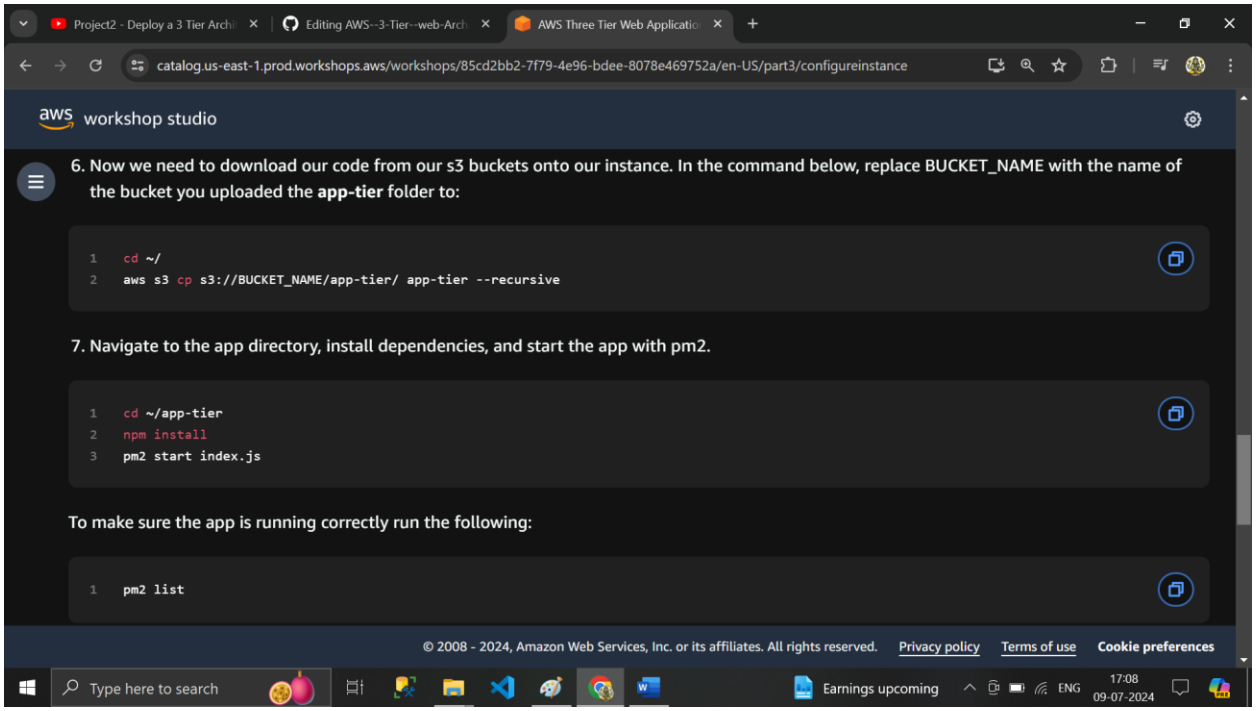
1. The first thing we will do is update our database credentials for the app tier. To do this, open the **application-code/app-tier/DbConfig.js** file from the github repo in your favorite text editor on your computer. You'll see empty strings for the hostname, user, password and database. Fill this in with the credentials you configured for your database, the **writer** endpoint of your database as the hostname, and **webappdb** for the database. Save the file.

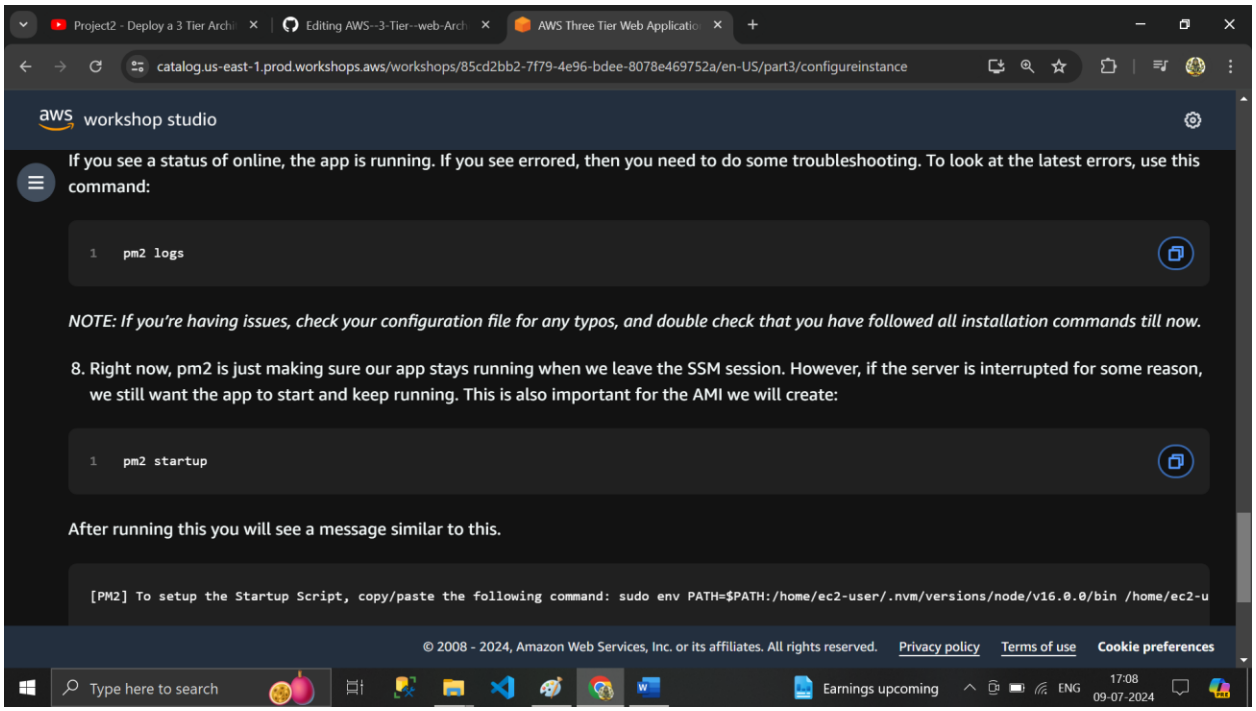
NOTE: This is NOT considered a best practice, and is done for the simplicity of the lab. Moving these credentials to a more suitable place like Secrets Manager is left as an extension for this workshop.

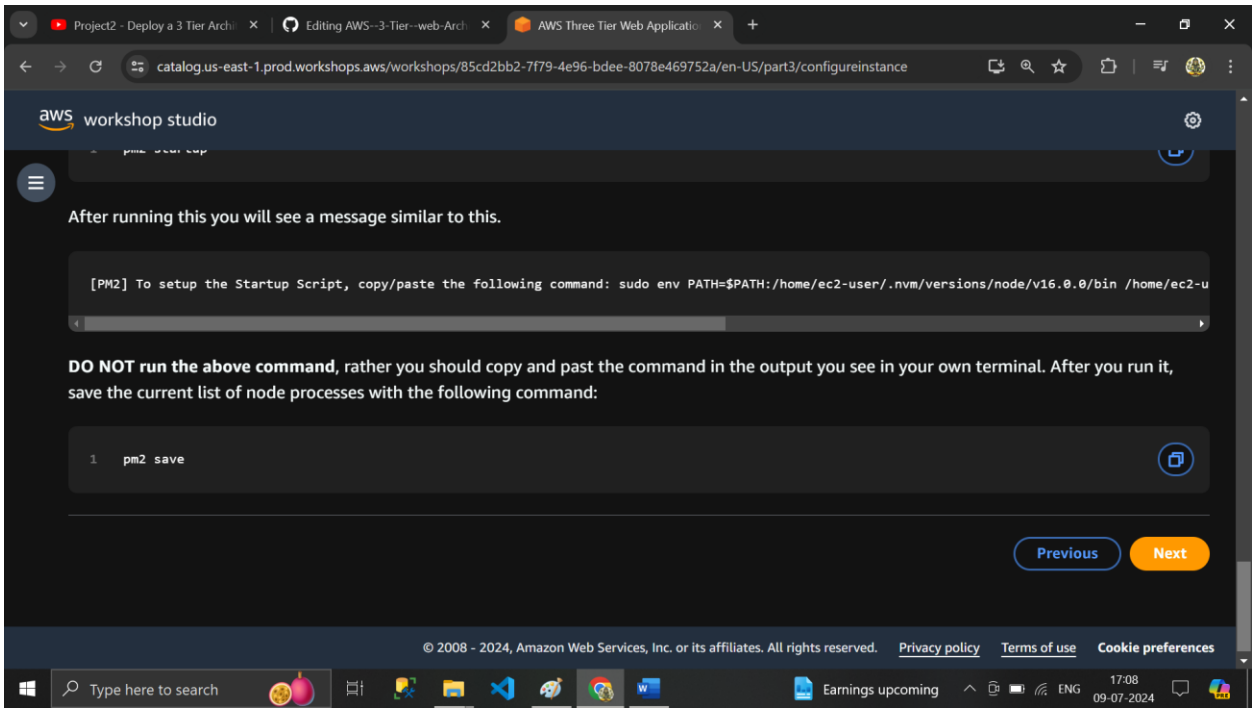


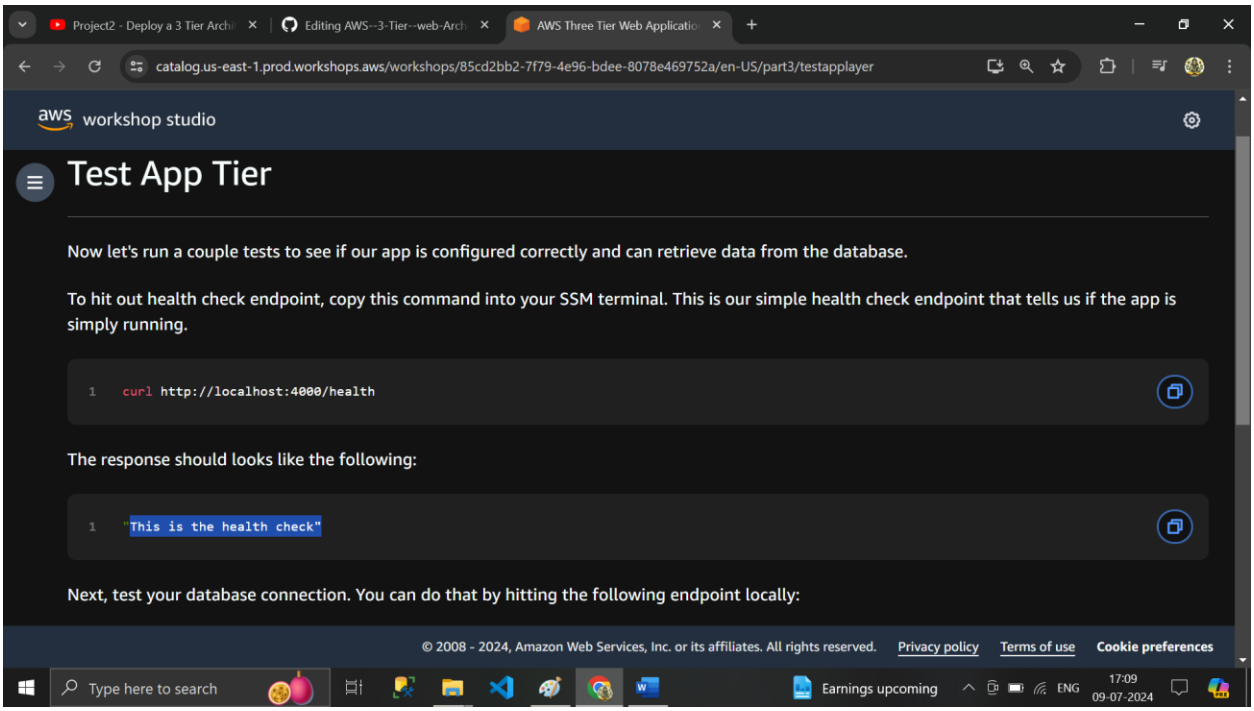


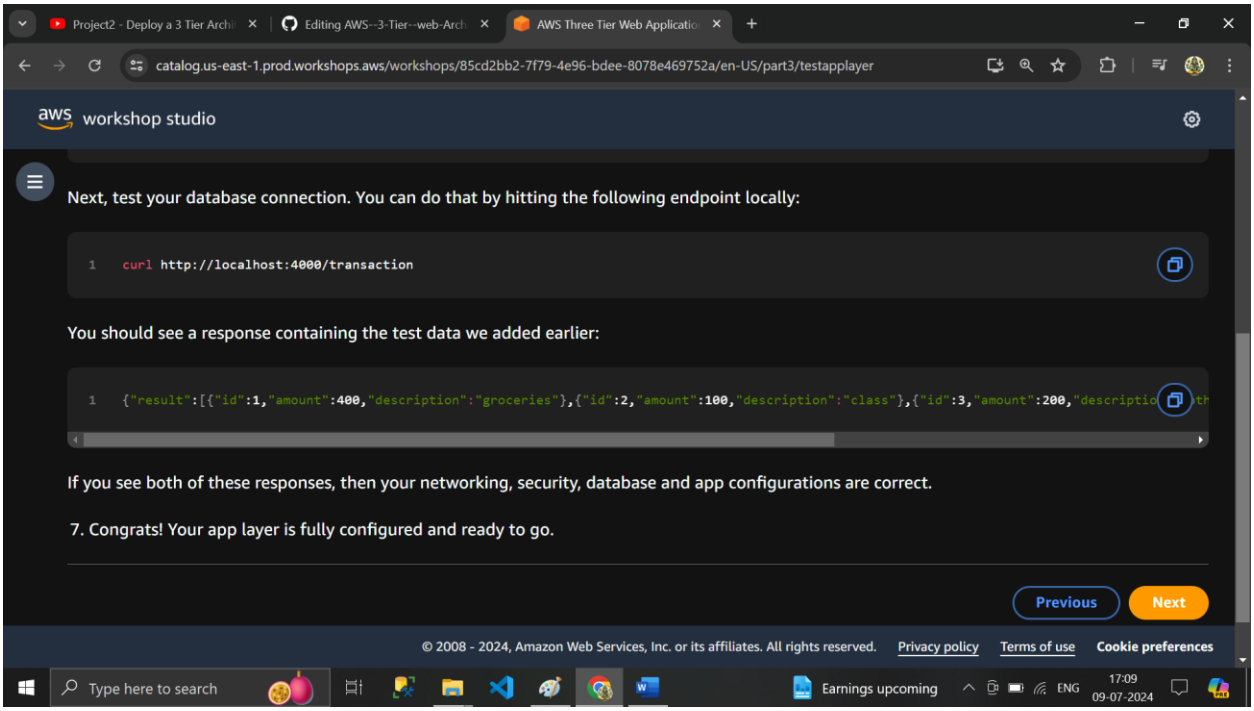

Project2 - Deploy a 3 Tier Archi
Editing AWS--3-Tier--web-Arch
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part3/configureinstance
aws workshop studio
2. Upload the app-tier folder to the S3 bucket that you created in part 0.
3. Go back to your SSM session. Now we need to install all of the necessary components to run our backend application. Start by installing NVM (node version manager).
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.38.0/install.sh | bash
source ~/.bashrc
4. Next, install a compatible version of Node.js and make sure it's being used
nvm install 16
nvm use 16
5. PM2 is a daemon process manager that will keep our node.js app running when we exit the instance or if it is rebooted. Install that as well.
npm install -g pm2
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
Earnings upcoming
ENG
17:07
09-07-2024

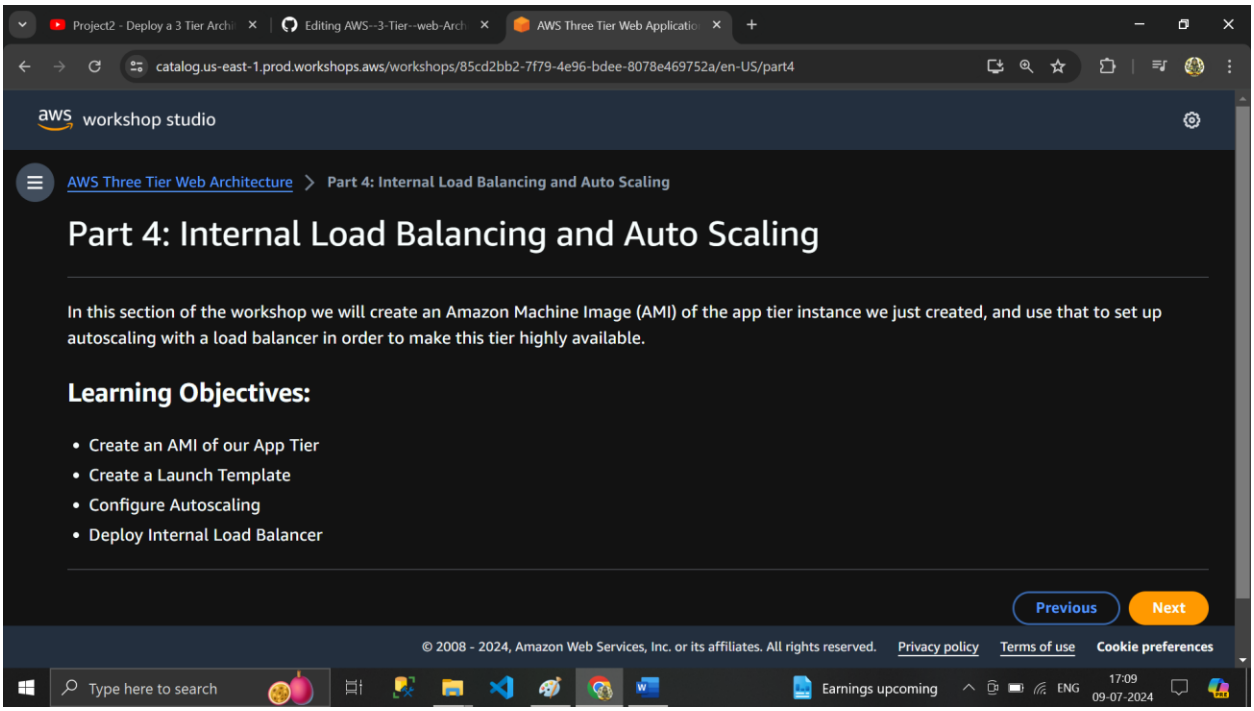

Project2 - Deploy a 3 Tier Archi
Editing AWS--3-Tier--web-Arch
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part3/configureinstance
aws workshop studio
6. Now we need to download our code from our s3 buckets onto our instance. In the command below, replace BUCKET_NAME with the name of the bucket you uploaded the app-tier folder to:
cd ~/
aws s3 cp s3://BUCKET_NAME/app-tier/ app-tier --recursive
7. Navigate to the app directory, install dependencies, and start the app with pm2.
cd ~/app-tier
npm install
pm2 start index.js
To make sure the app is running correctly run the following:
pm2 list
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
Earnings upcoming
ENG
17:08
09-07-2024


Project2 - Deploy a 3 Tier Archi
Editing AWS--3-Tier--web-Arch
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part3/configureinstance
aws workshop studio
If you see a status of online, the app is running. If you see errored, then you need to do some troubleshooting. To look at the latest errors, use this command:
1 pm2 logs
NOTE: If you're having issues, check your configuration file for any typos, and double check that you have followed all installation commands till now.
8. Right now, pm2 is just making sure our app stays running when we leave the SSM session. However, if the server is interrupted for some reason, we still want the app to start and keep running. This is also important for the AMI we will create:
1 pm2 startup
After running this you will see a message similar to this.
[PM2] To setup the Startup Script, copy/paste the following command: sudo env PATH=$PATH:/home/ec2-user/.nvm/versions/node/v16.0.0/bin /home/ec2-u
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
Earnings upcoming
ENG
17:08
09-07-2024


Project2 - Deploy a 3 Tier Archi
Editing AWS--3-Tier--web-Arch
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part3/configureinstance
aws workshop studio
After running this you will see a message similar to this.
[PM2] To setup the Startup Script, copy/paste the following command: sudo env PATH=$PATH:/home/ec2-user/.nvm/versions/node/v16.0.0/bin /home/ec2-u
DO NOT run the above command, rather you should copy and past the command in the output you see in your own terminal. After you run it, save the current list of node processes with the following command:
1 pm2 save
Previous
Next
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
Earnings upcoming
ENG
17:08
09-07-2024

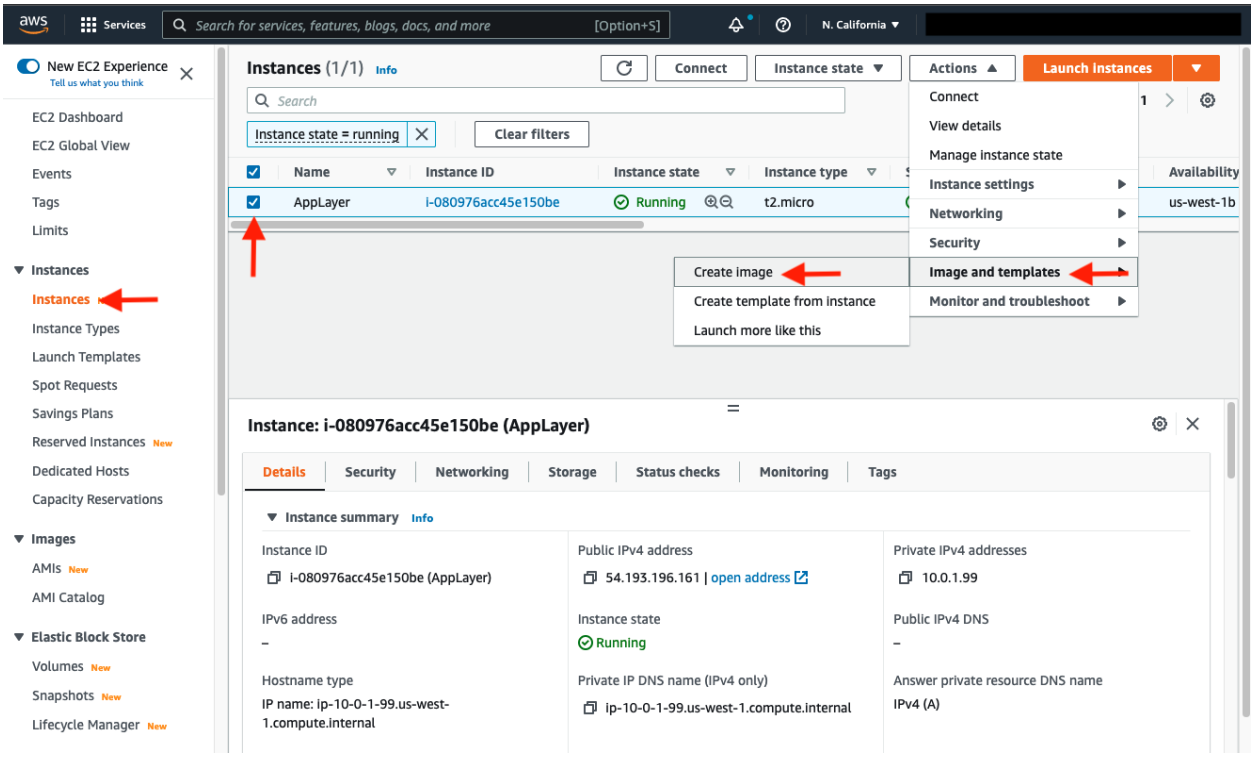

Project2 - Deploy a 3 Tier Archi
Editing AWS--3-Tier--web-Arch
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part3/testapplayer
aws workshop studio
Test App Tier
Now let's run a couple tests to see if our app is configured correctly and can retrieve data from the database.
To hit out health check endpoint, copy this command into your SSM terminal. This is our simple health check endpoint that tells us if the app is simply running.
1 curl http://localhost:4000/health
The response should looks like the following:
1 "This is the health check"
Next, test your database connection. You can do that by hitting the following endpoint locally:
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
Earnings upcoming
ENG
17:09
09-07-2024


Project2 - Deploy a 3 Tier Archi
Editing AWS--3-Tier--web-Arch
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part3/testapplayer
aws workshop studio
Next, test your database connection. You can do that by hitting the following endpoint locally:
1 curl http://localhost:4000/transaction
You should see a response containing the test data we added earlier:
1 {"result":[{"id":1,"amount":400,"description":"groceries"},{"id":2,"amount":100,"description":"class"},{"id":3,"amount":200,"descriptio
If you see both of these responses, then your networking, security, database and app configurations are correct.
7. Congrats! Your app layer is fully configured and ready to go.
Previous
Next
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
Earnings upcoming
ENG
17:09
09-07-2024

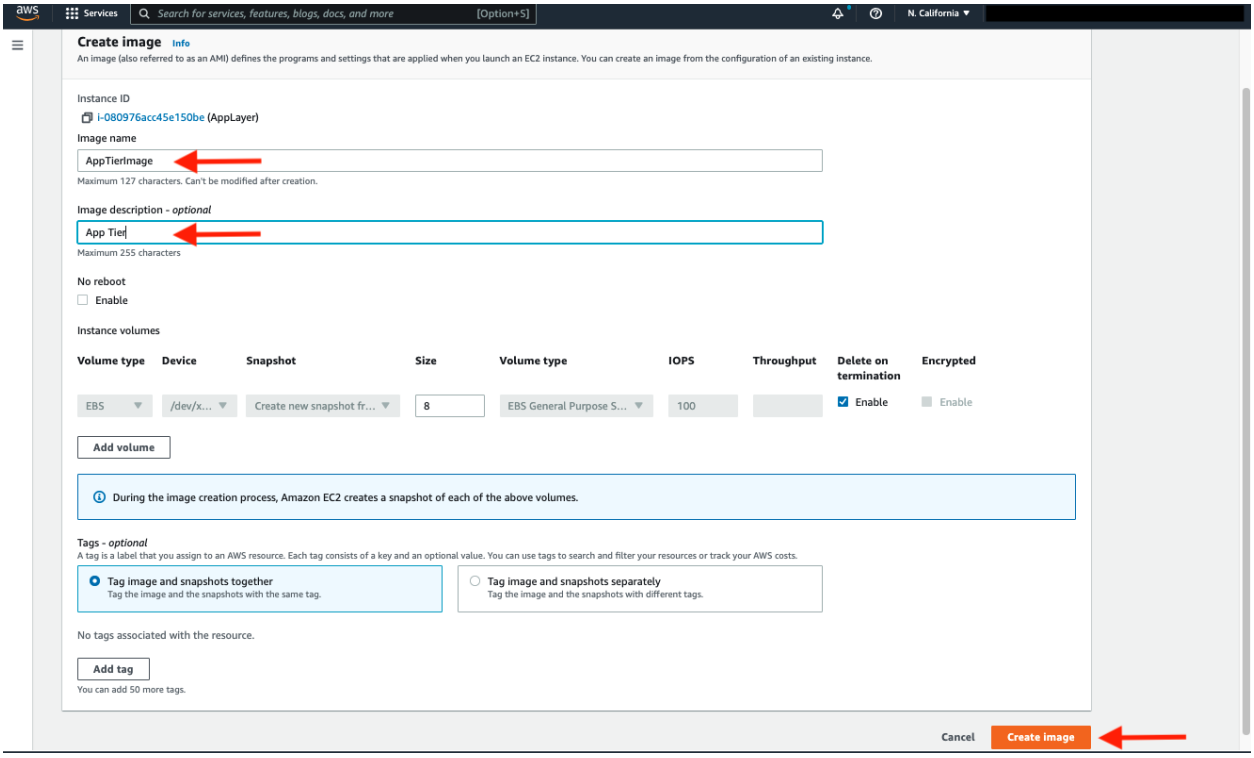


App Tier AMI

1. Navigate to **Instances** on the left hand side of the EC2 dashboard. Select the app tier instance we created and under **Actions** select **Image and templates**. Click **Create Image**.

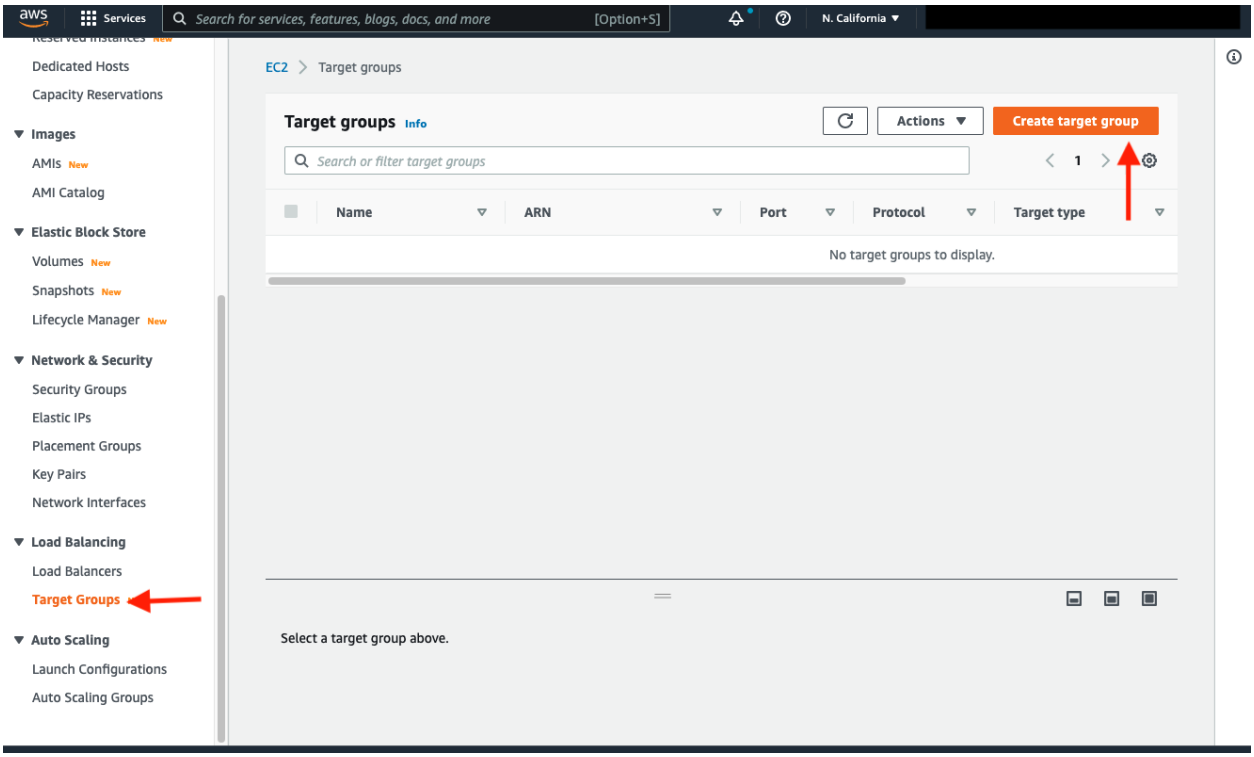


Give the image a name and description and then click **Create image**. This will take a few minutes, but if you want to monitor the status of image creation you can see it by clicking **AMIs** under **Images** on the left hand navigation panel of the EC2 dashboard

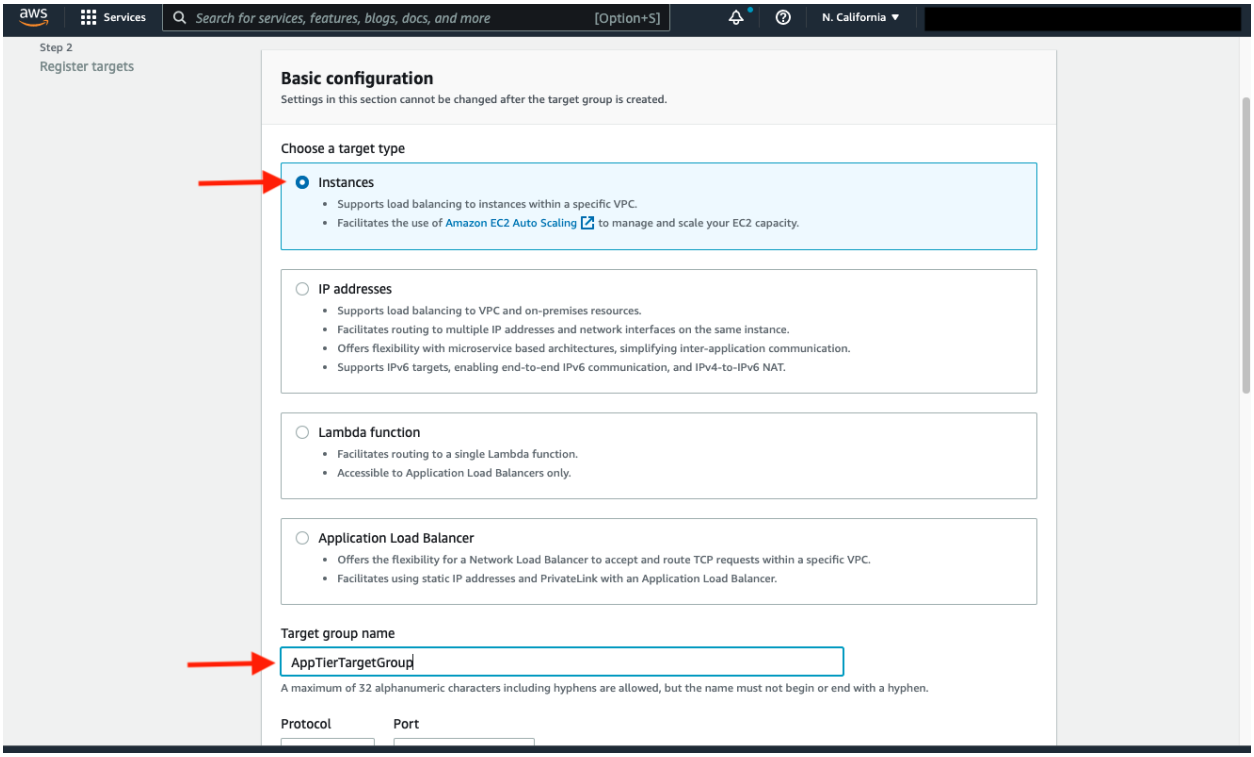


Target Group

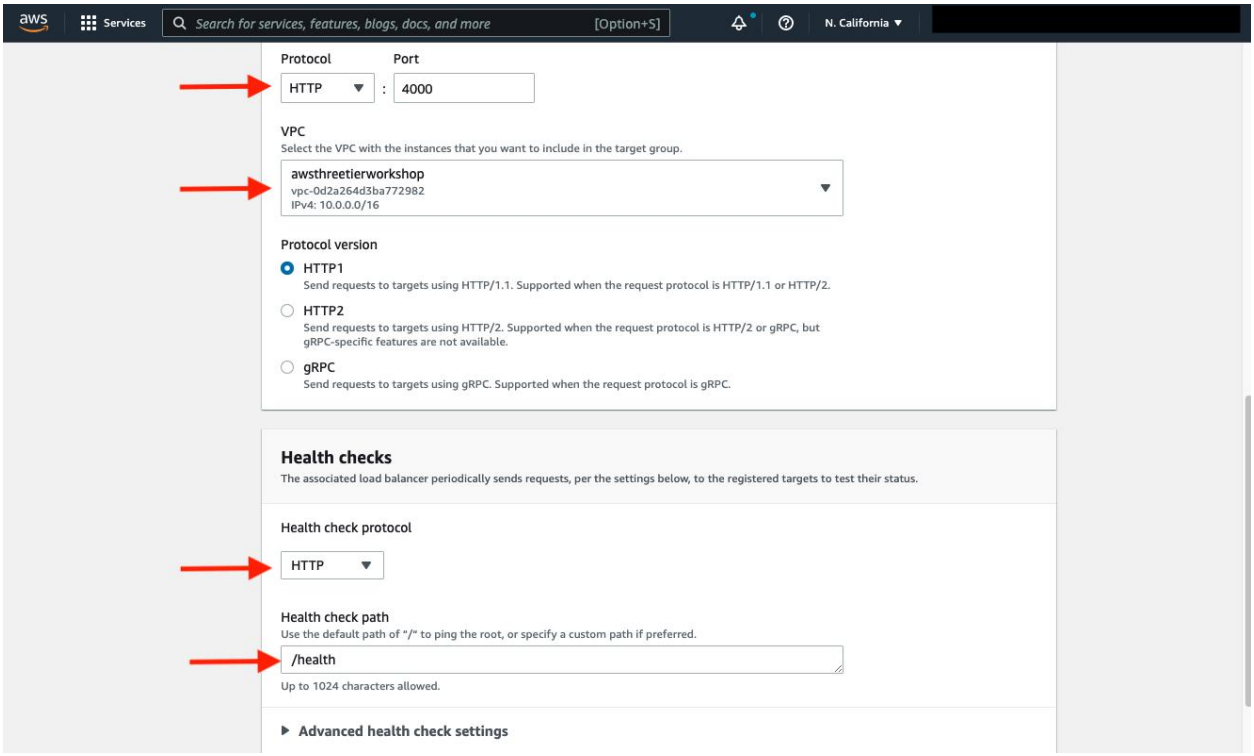
1. While the AMI is being created, we can go ahead and create our target group to use with the load balancer. On the EC2 dashboard navigate to **Target Groups** under **Load Balancing** on the left hand side. Click on **Create Target Group**



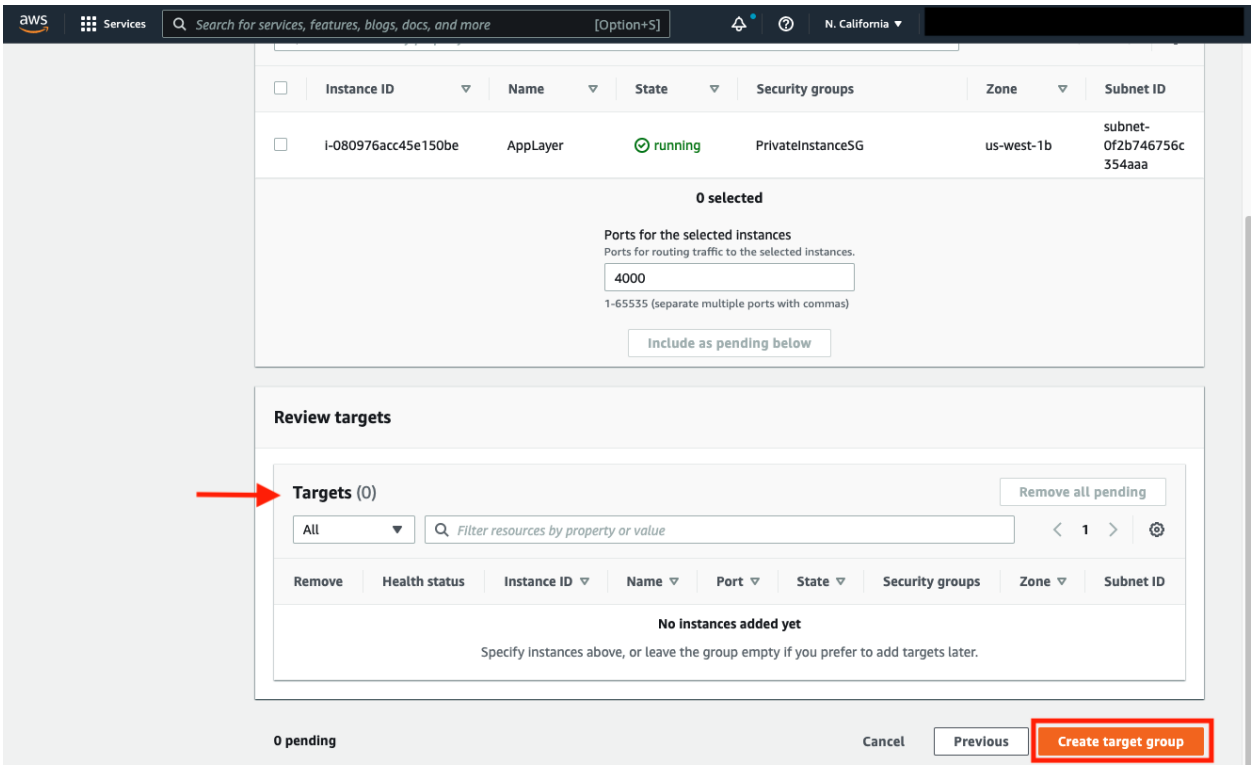
2. The purpose of forming this target group is to use with our load blancer so it may balance traffic across our private app tier instances. Select Instances as the target type and give it a name.



hen, set the protocol to **HTTP** and the port to 4000. Remember that this is the port our Node.ja app is running on. Select the VPC we've been using thus far, and then change the health check path to be **/health**. This is the health check endpoint of our app. Click **Next**.

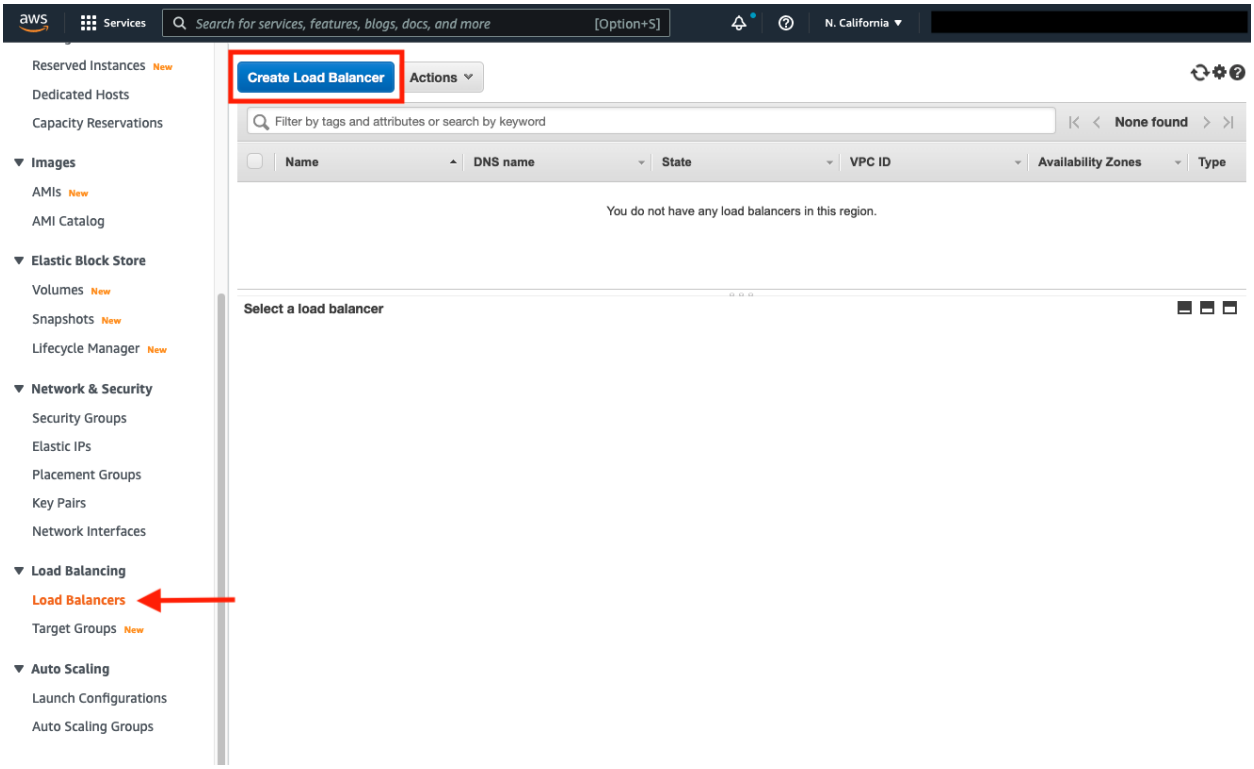


3. We are **NOT** going to register any targets for now, so just skip that step and create the target group.

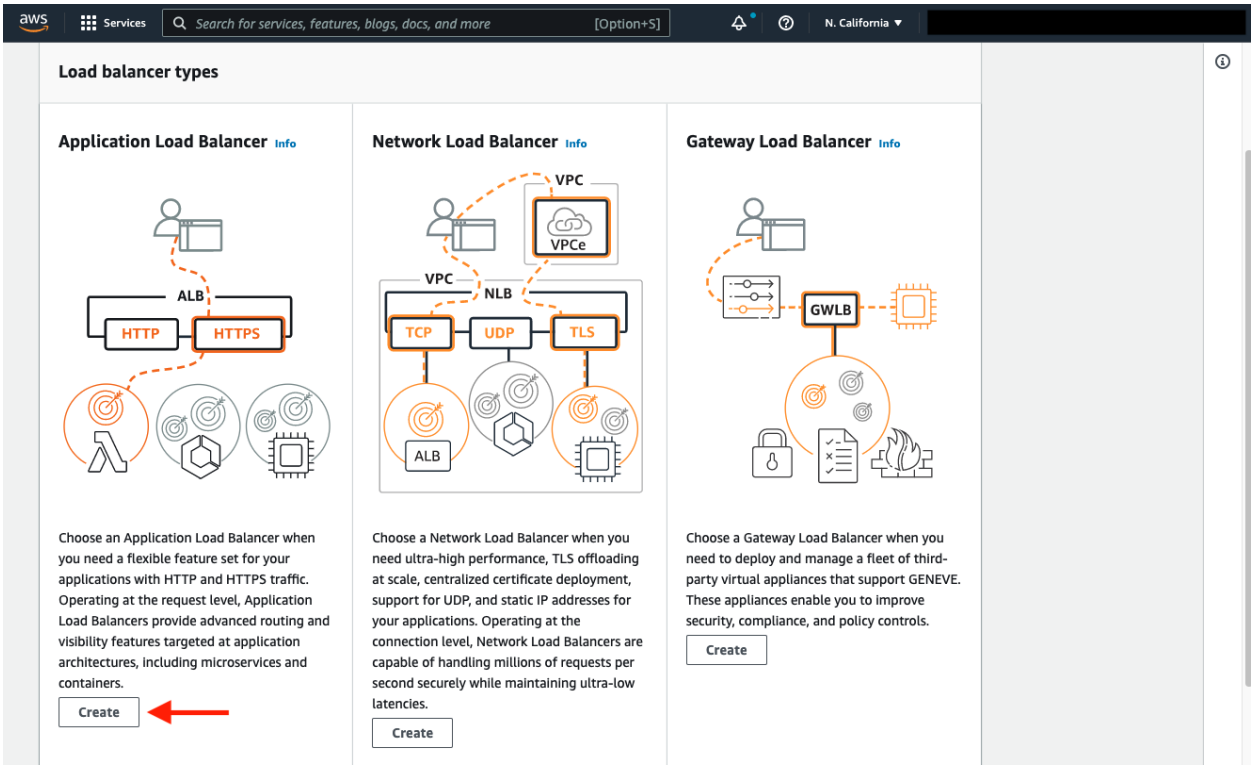


Internal Load Balancer

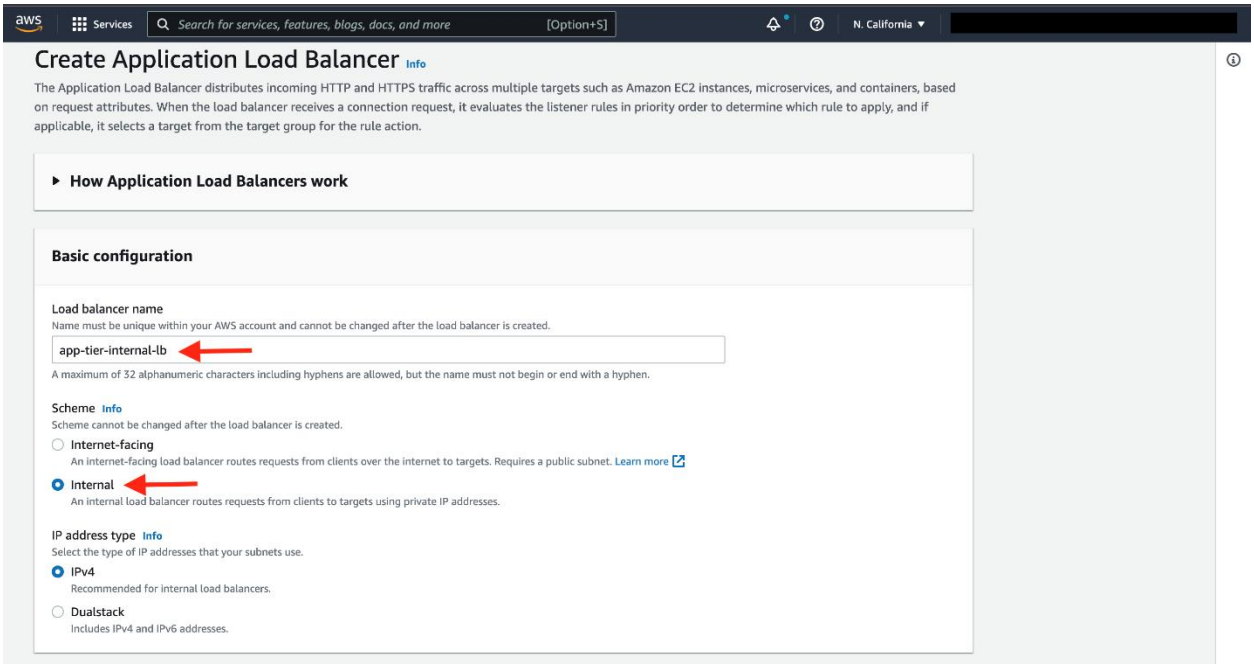
1. On the left hand side of the EC2 dashboard select **Load Balancers** under **Load Balancing** and click **Create Load Balancer**.



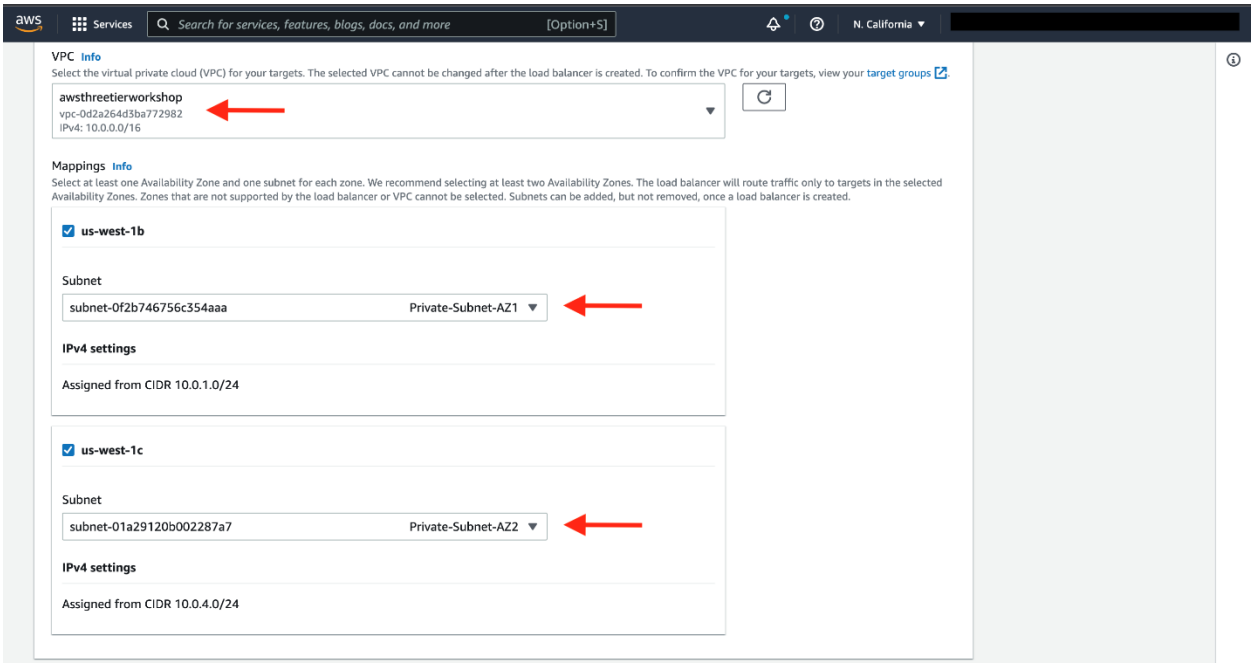
2. We'll be using an **Application Load Balancer** for our **HTTP** traffic so click the create button for that option.



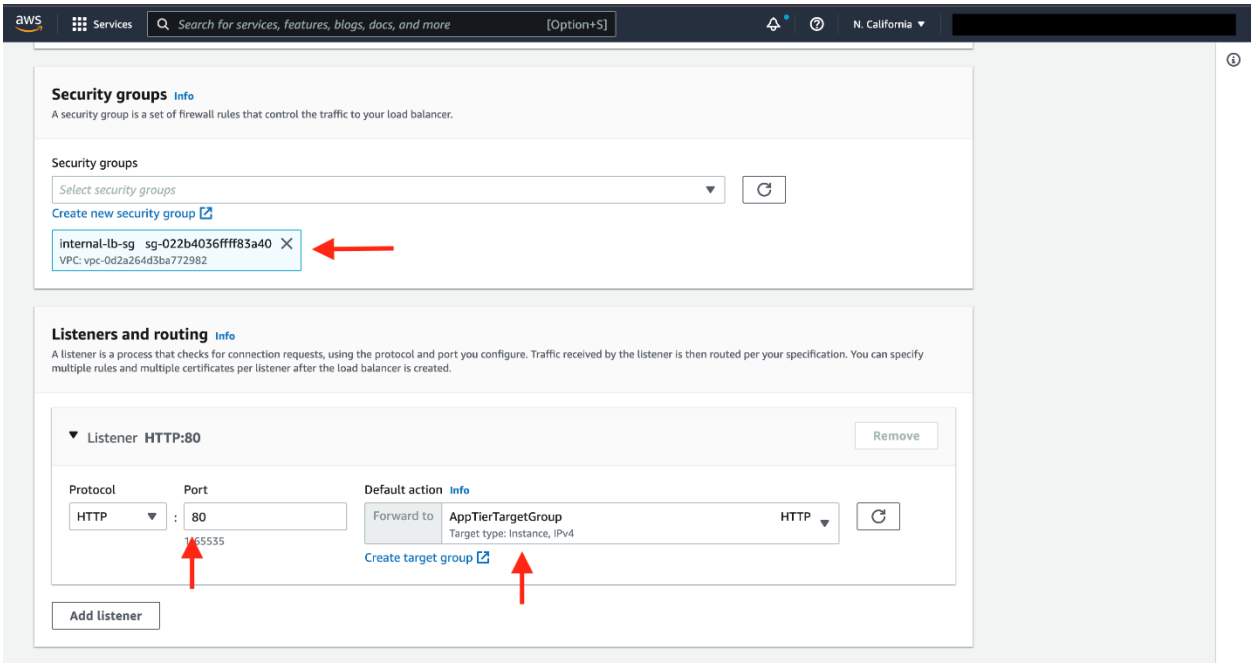
3. After giving the load balancer a name, be sure to select **internal** since this one will not be public facing, but rather it will route traffic from our web tier to the app tier.



Select the correct network configuration for VPC and private subnet

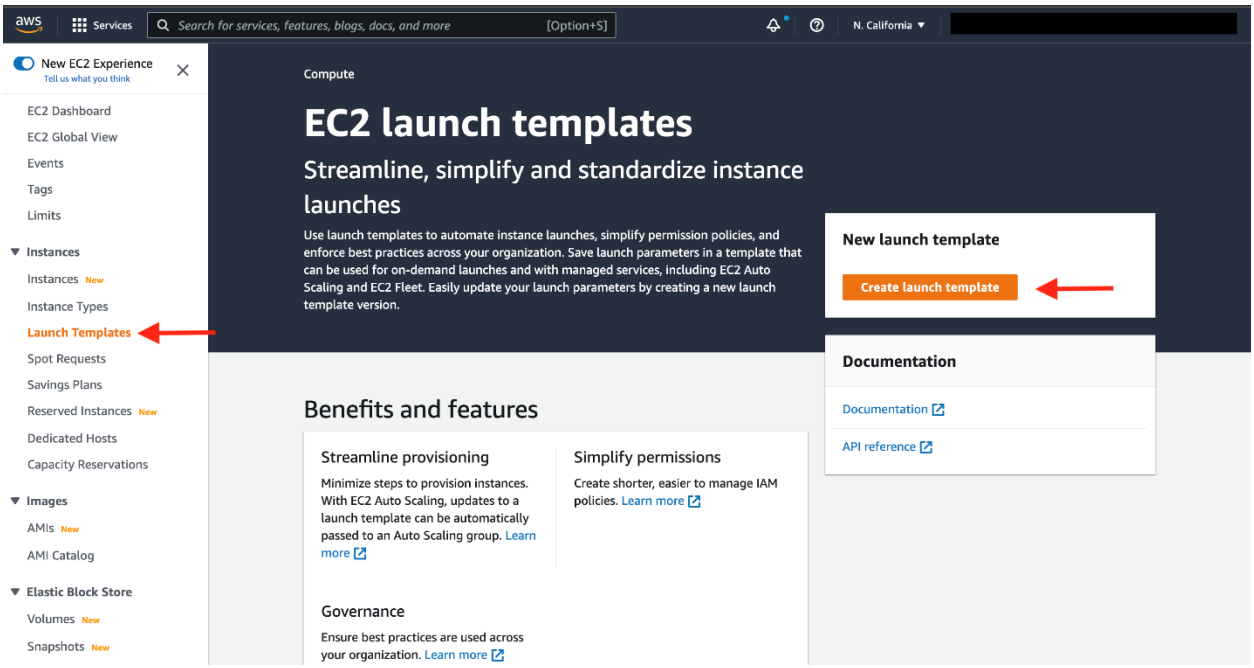


Select the security group we created for this internal ALB. Now, this ALB will be listening for HTTP traffic on port 80. It will be forwarding the traffic to our **target group** that we just created, so select it from the dropdown, and create the load balancer.

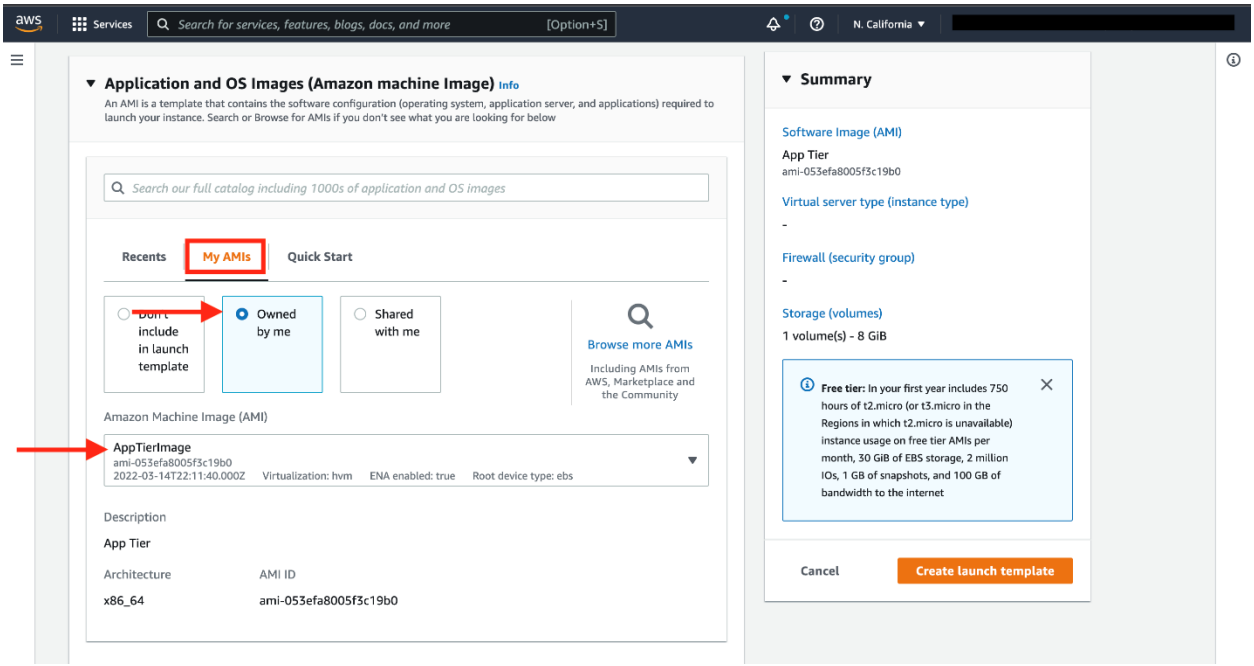


Launch Template

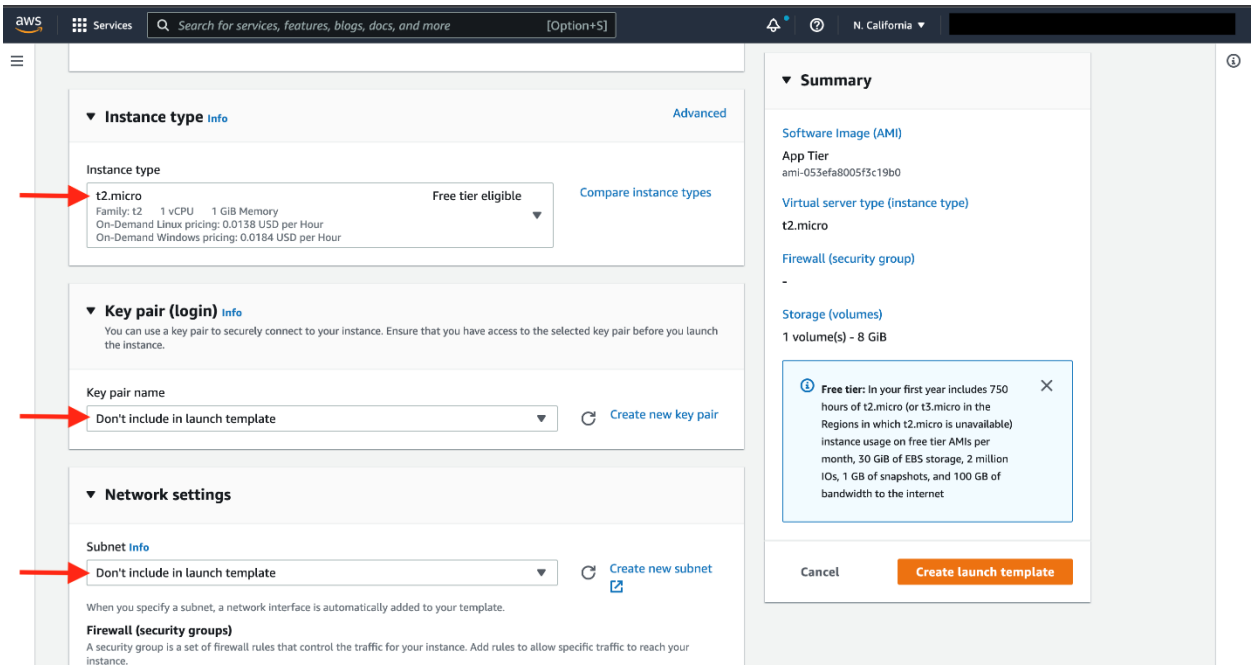
1. Before we configure Auto Scaling, we need to create a Launch template with the AMI we created earlier. On the left side of the EC2 dashboard navigate to **Launch Template** under **Instances** and click **Create Launch Template**.



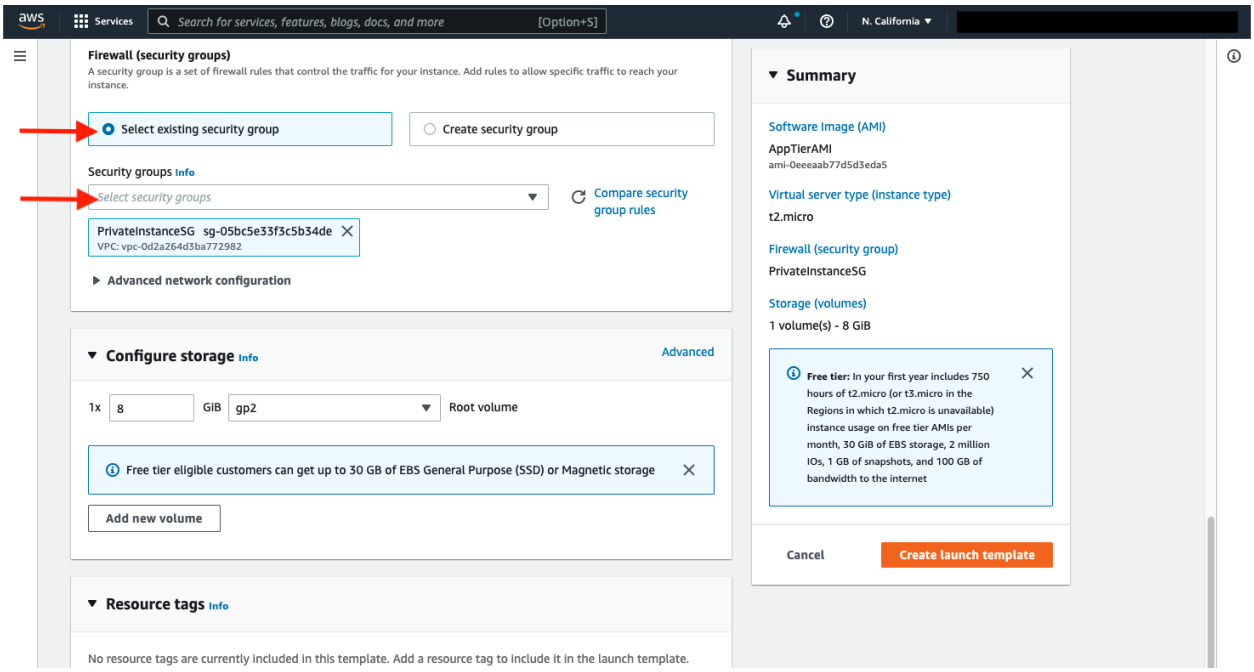
2. Name the Launch Template, and then under **Application and OS Images** include the app tier AMI you created.

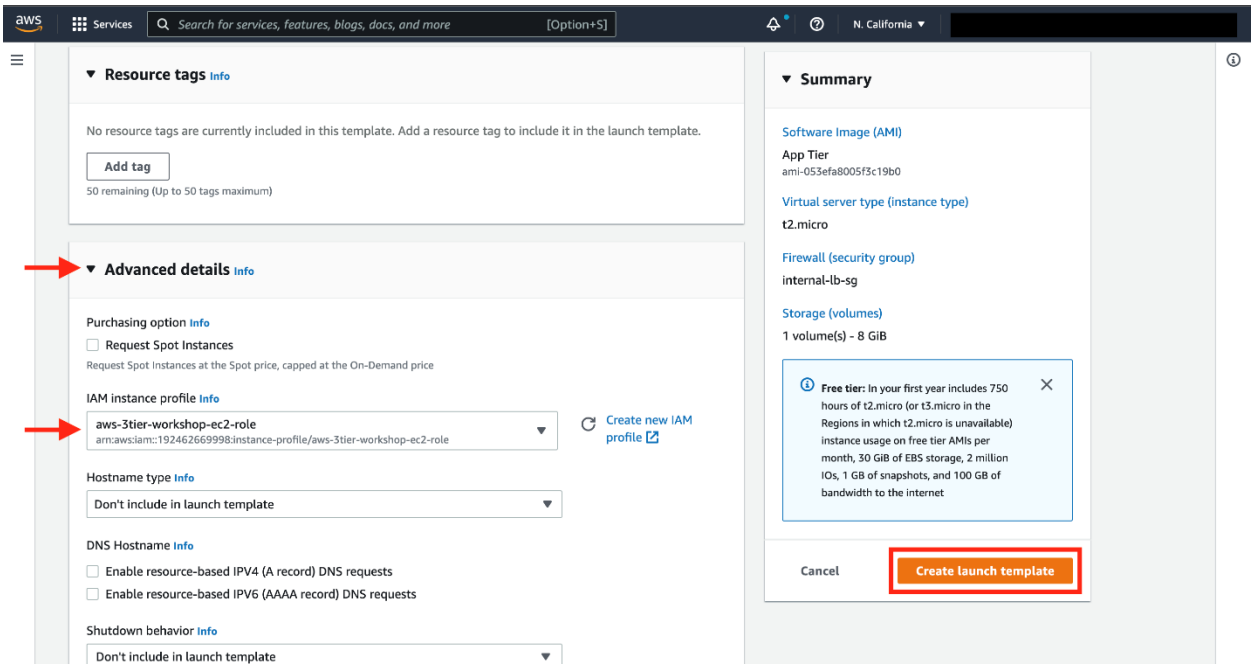


Under **Instance Type** select t2.micro. For **Key pair** and **Network Settings** don't include it in the template. We don't need a key pair to access our instances and we'll be setting the network information in the autoscaling group.



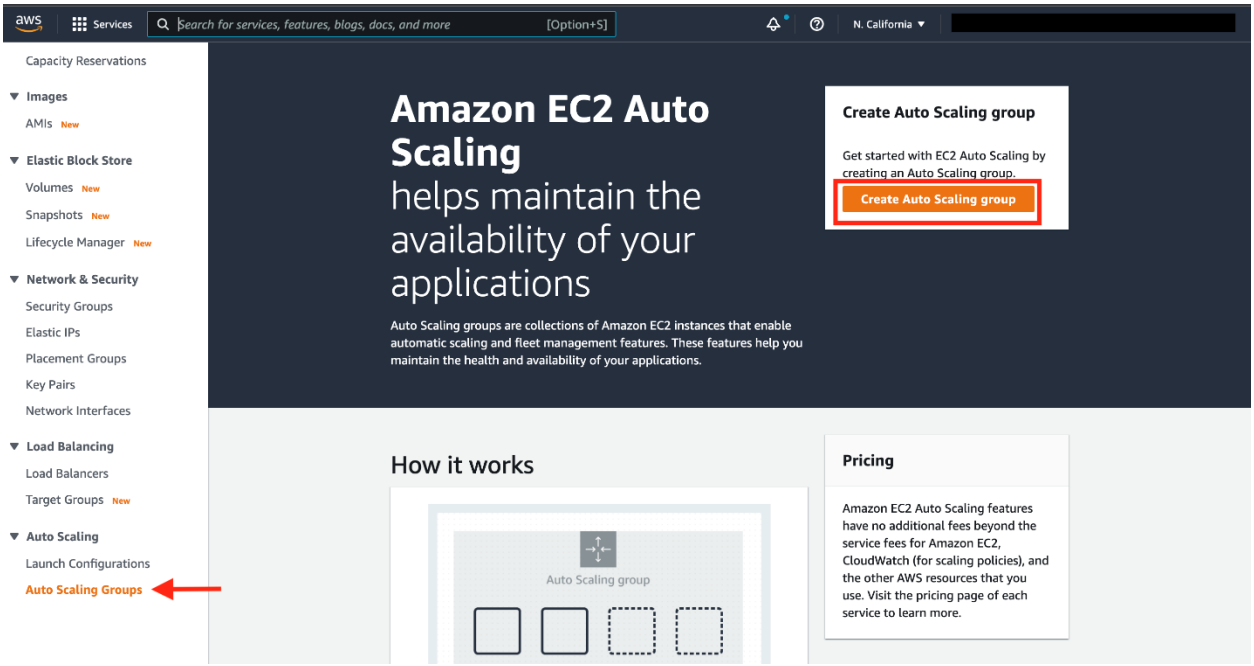
Set the correct security group for our app tier, and then under **Advanced details** use the same IAM instance profile we have been using for our EC2 instances.

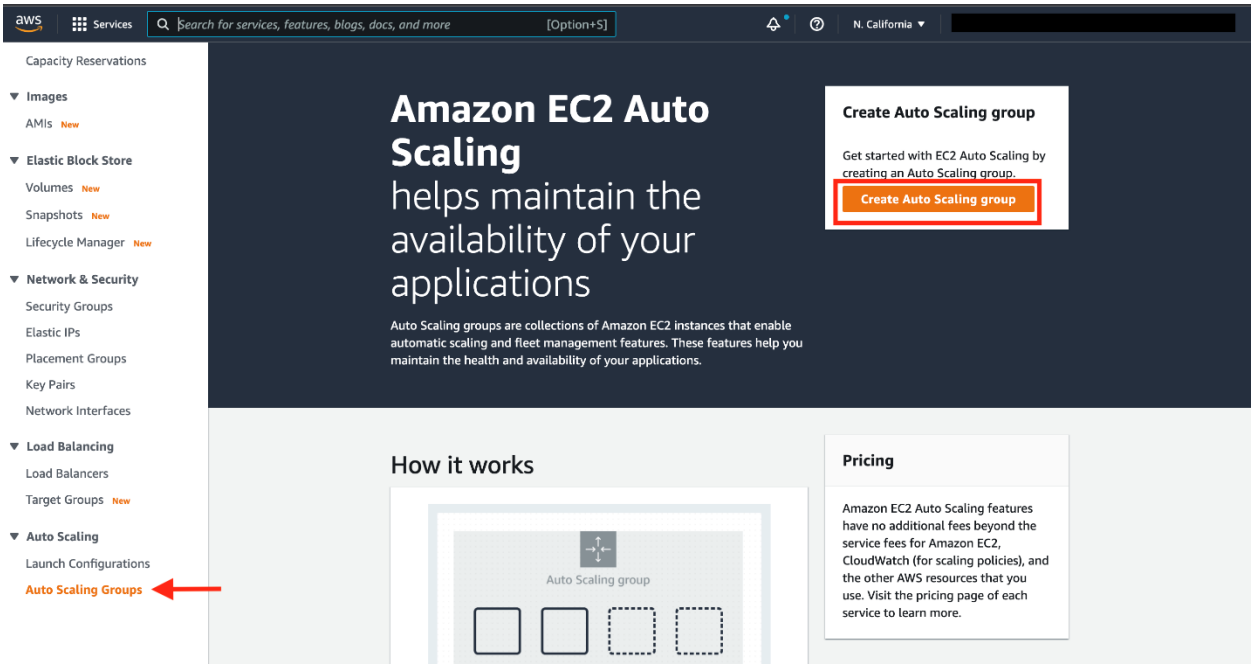




Auto Scaling

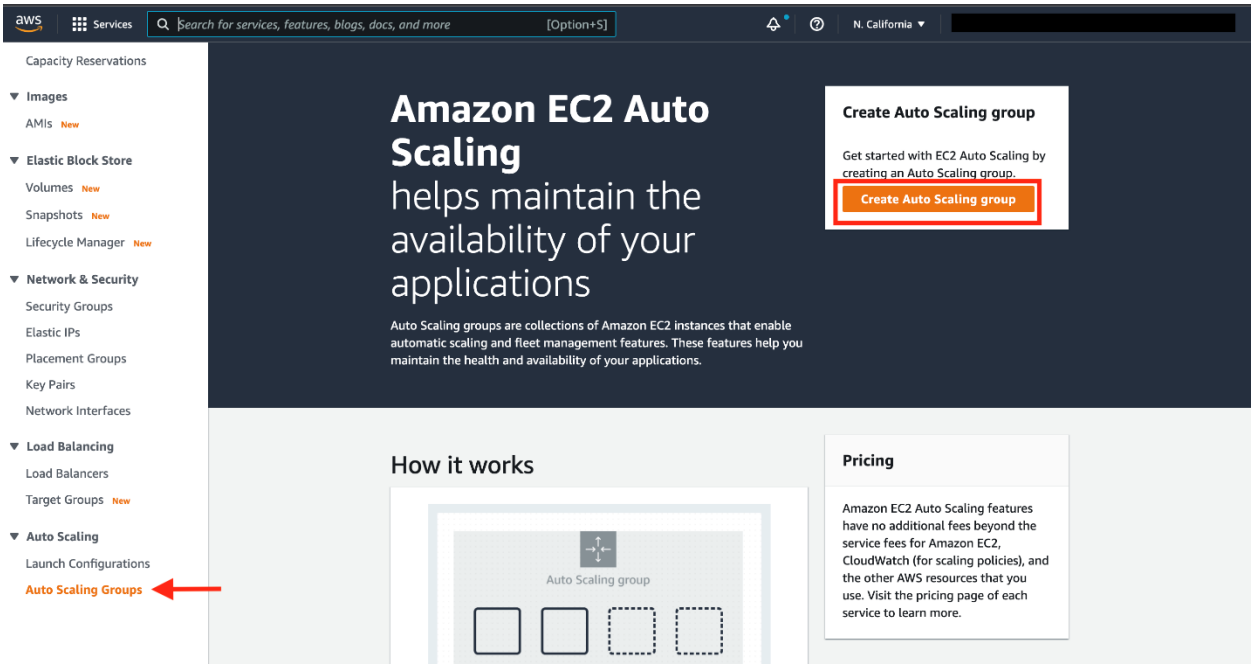
1. We will now create the Auto Scaling Group for our app instances. On the left side of the EC2 dashboard navigate to **Auto Scaling Groups** under **Auto Scaling** and click **Create Auto Scaling group**.



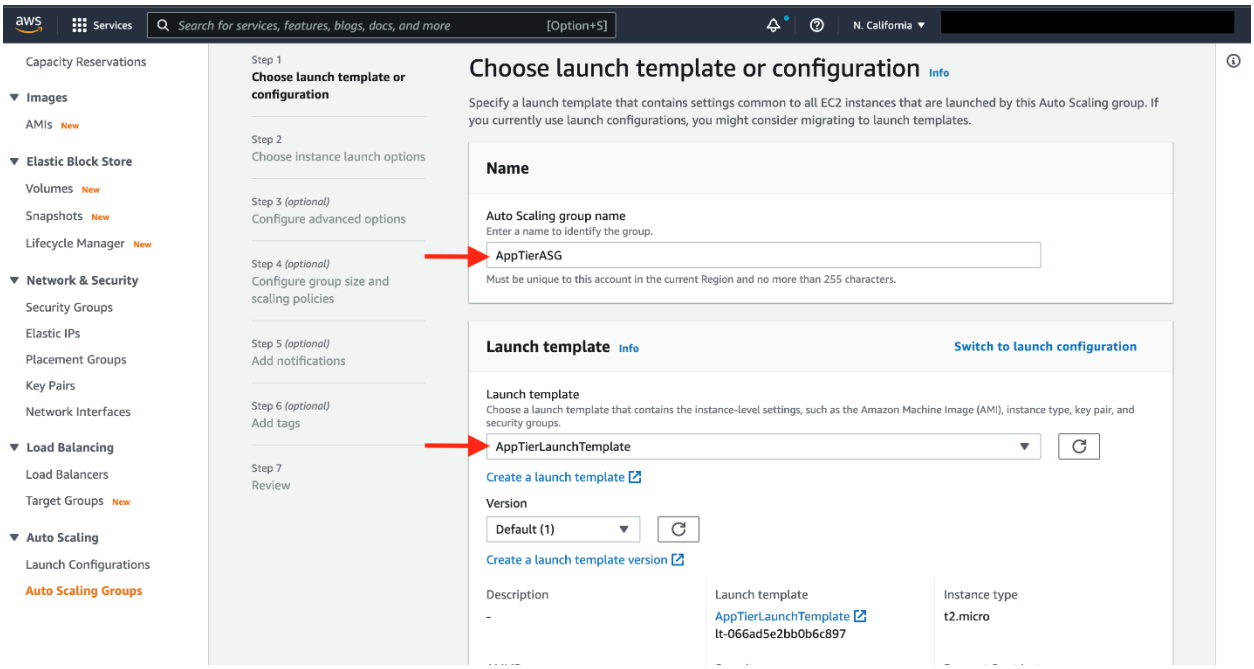


Auto Scaling

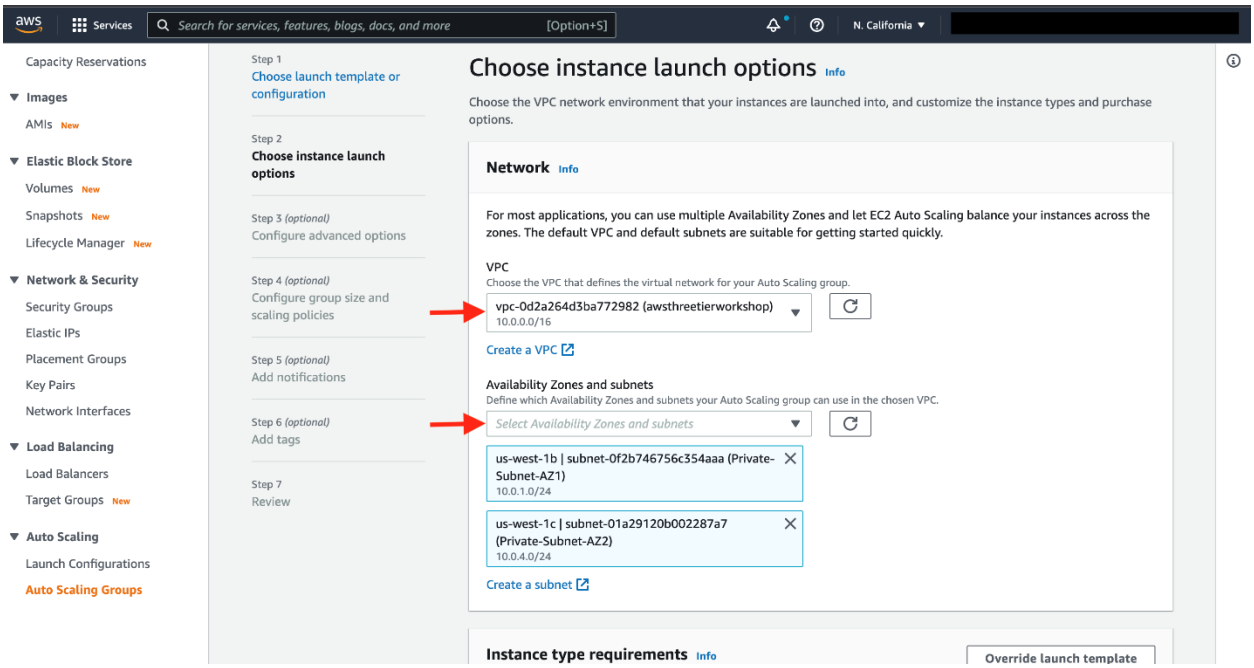
1. We will now create the Auto Scaling Group for our app instances. On the left side of the EC2 dashboard navigate to **Auto Scaling Groups** under **Auto Scaling** and click **Create Auto Scaling group**.



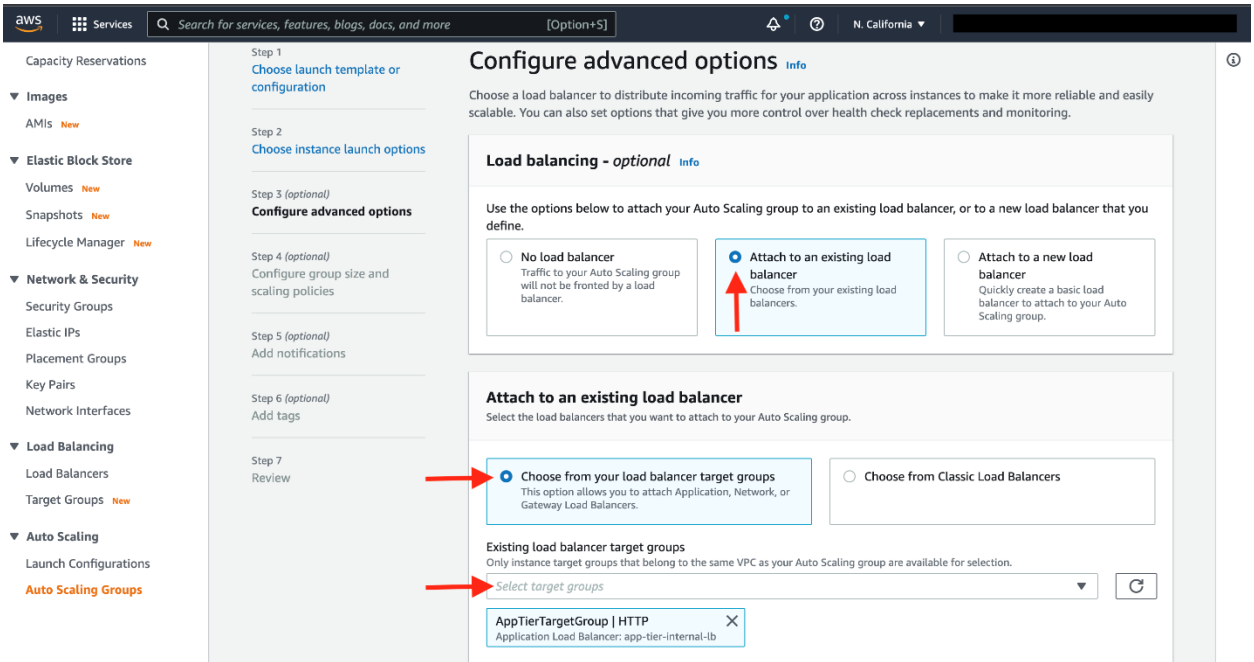
2. Give your Auto Scaling group a name, and then select the Launch Template we just created and click next.



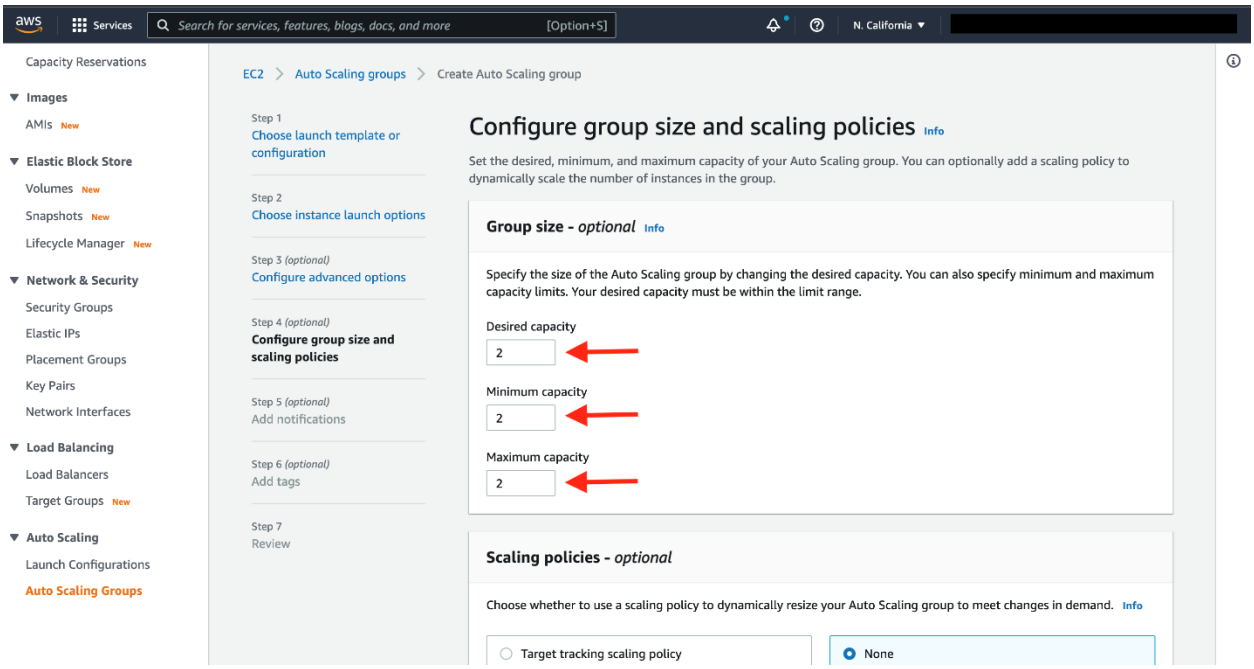
3. On the **Choose instance launch options** page set your VPC, and the private instance subnets for the app tier and continue to step 3.



For this next step, attach this Auto Scaling Group to the Load Balancer we just created by selecting the existing load balancer's target group from the dropdown. Then, click next.

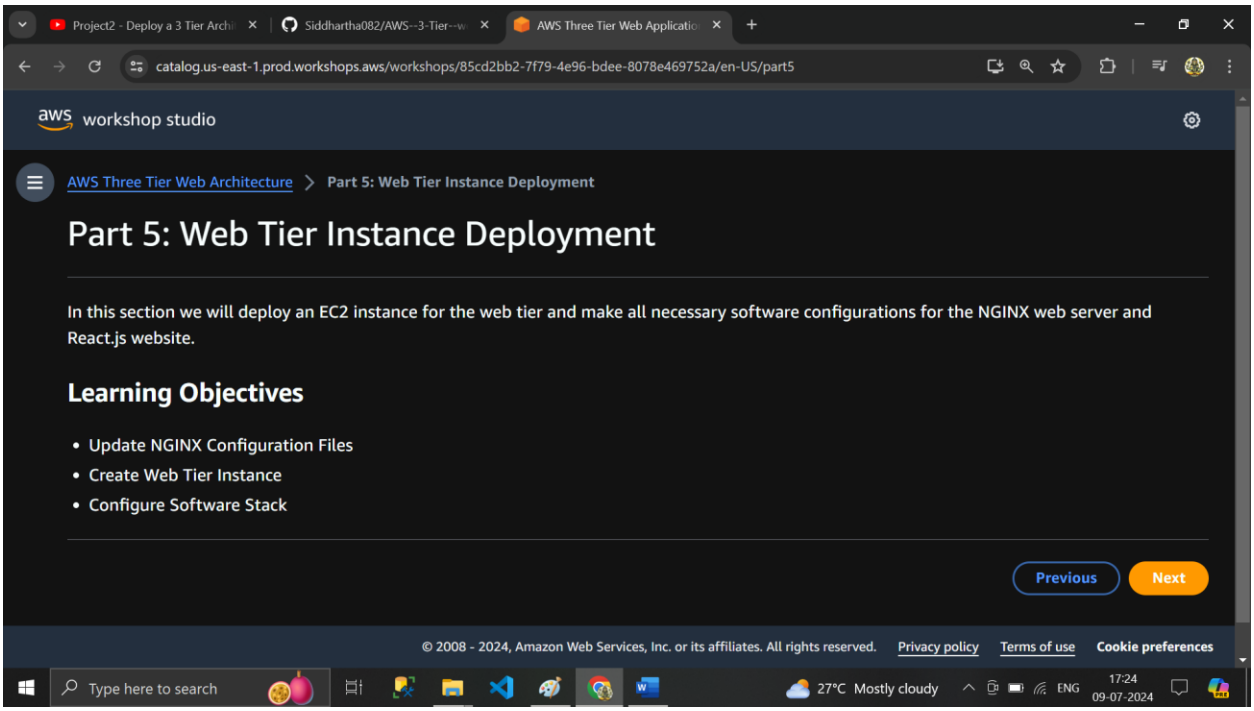


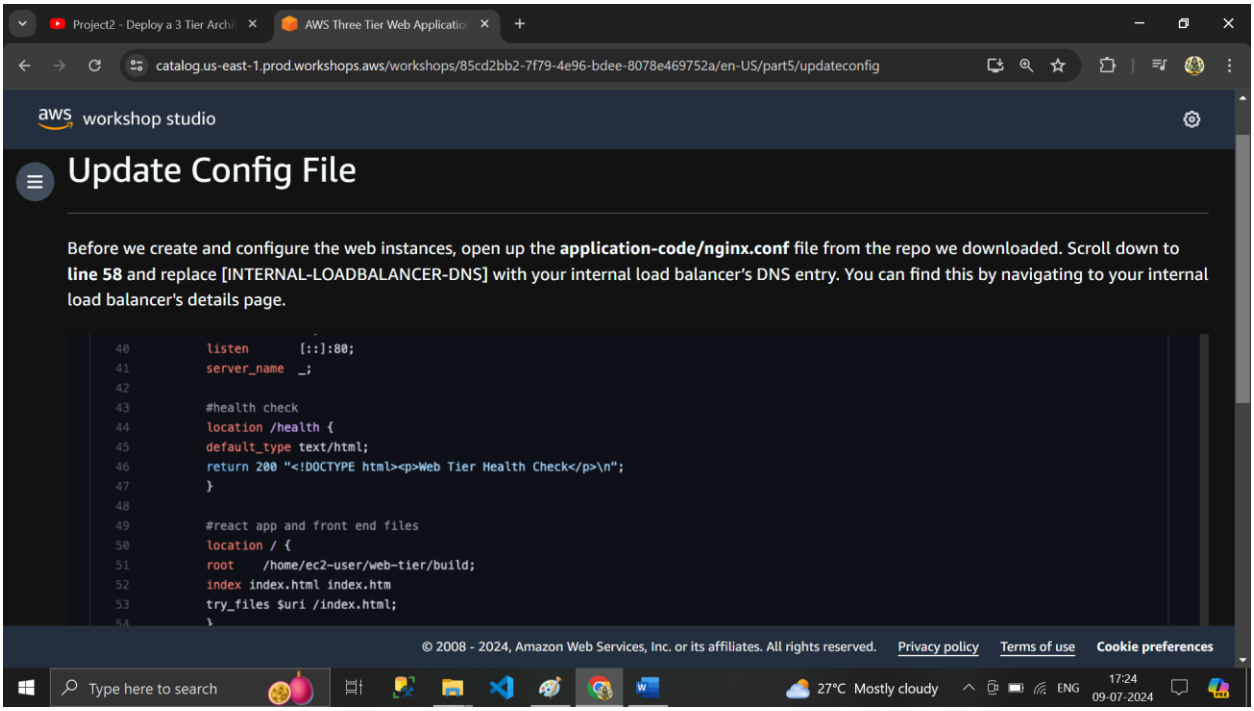
5. For **Configure group size and scaling policies**, set desired, minimum and maximum capacity to **2**. Click skip to review and then Create Auto Scaling Group.

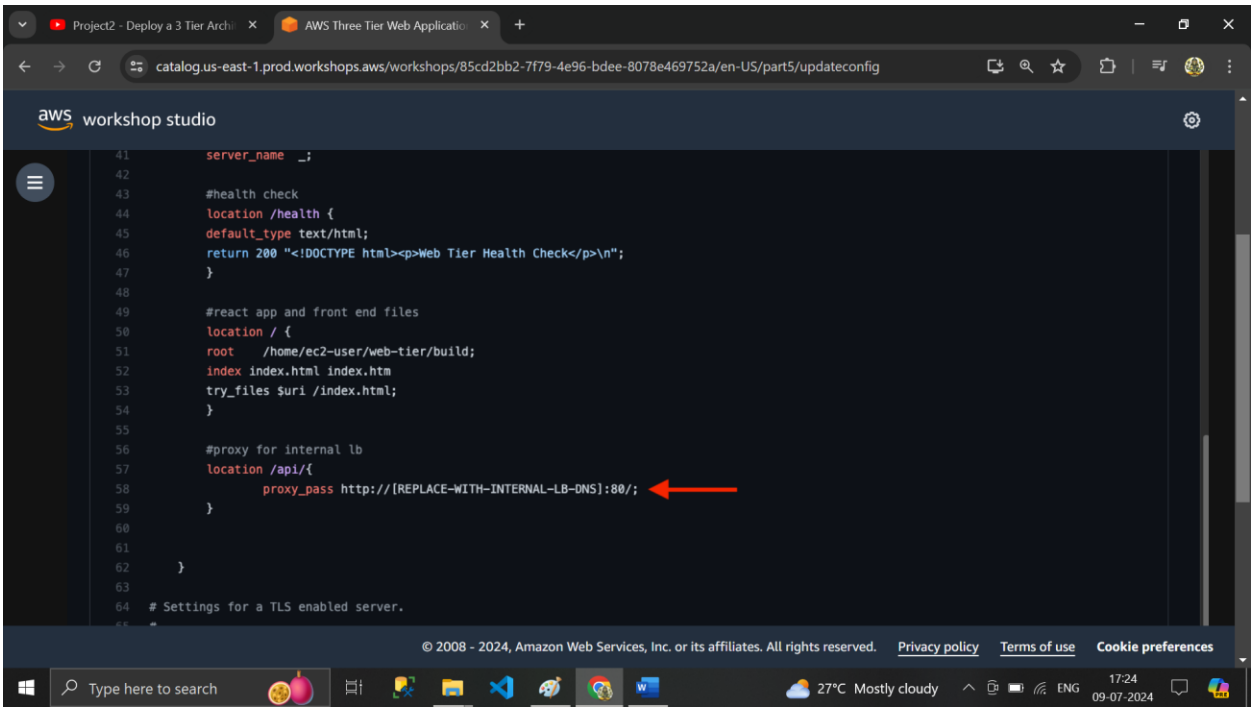


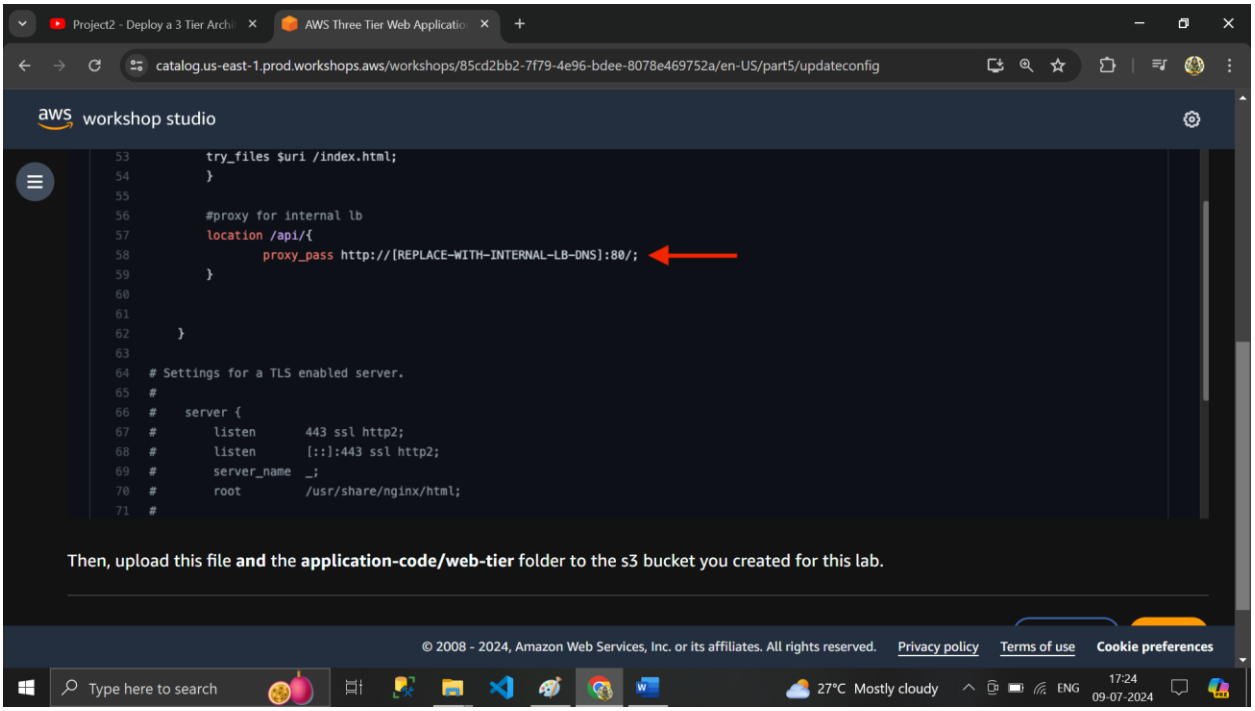
You should now have your internal load balancer and autoscaling group configured correctly. You should see the autoscaling group spinning up 2 new app tier instances. If you wanted to test if this is working correctly, you can delete one of your new instances manually and wait to see if a new instance is booted up to replace it.

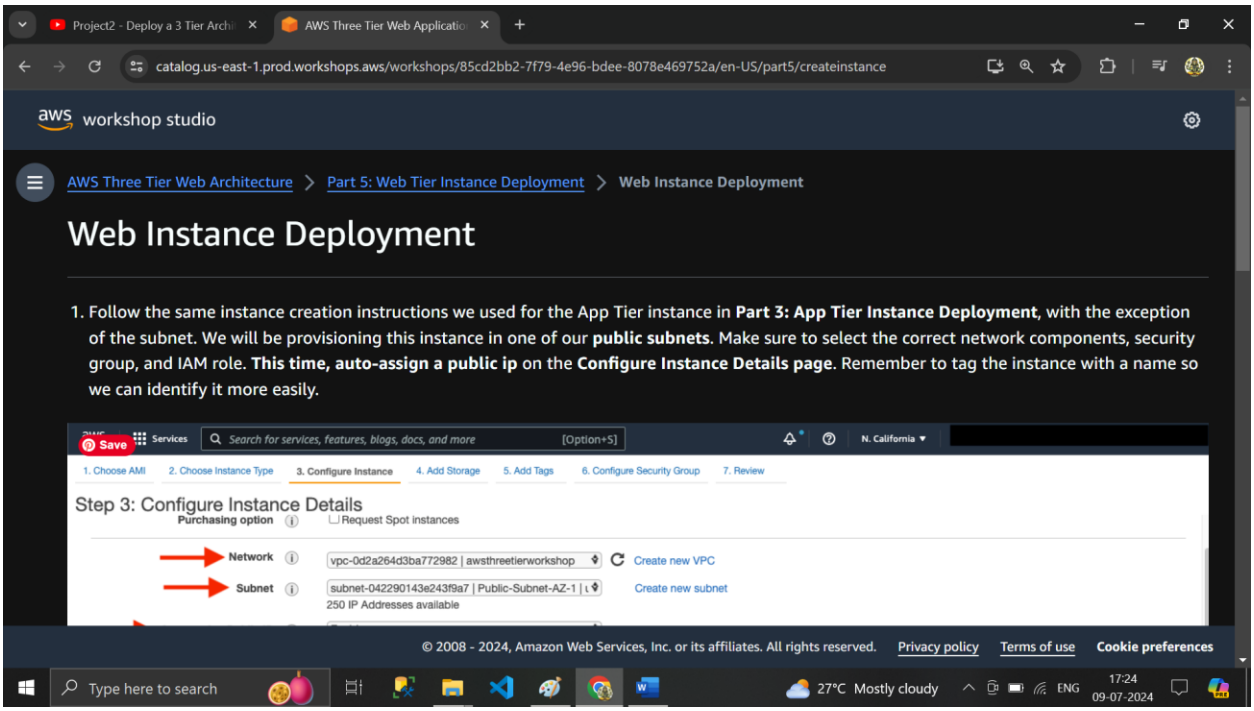
NOTE: Your original app tier instance is excluded from the ASG so you will see 3 instances in the EC2 dashboard. You can delete your original instance that you used to generate the app tier AMI but it's recommended to keep it around for troubleshooting purposes.

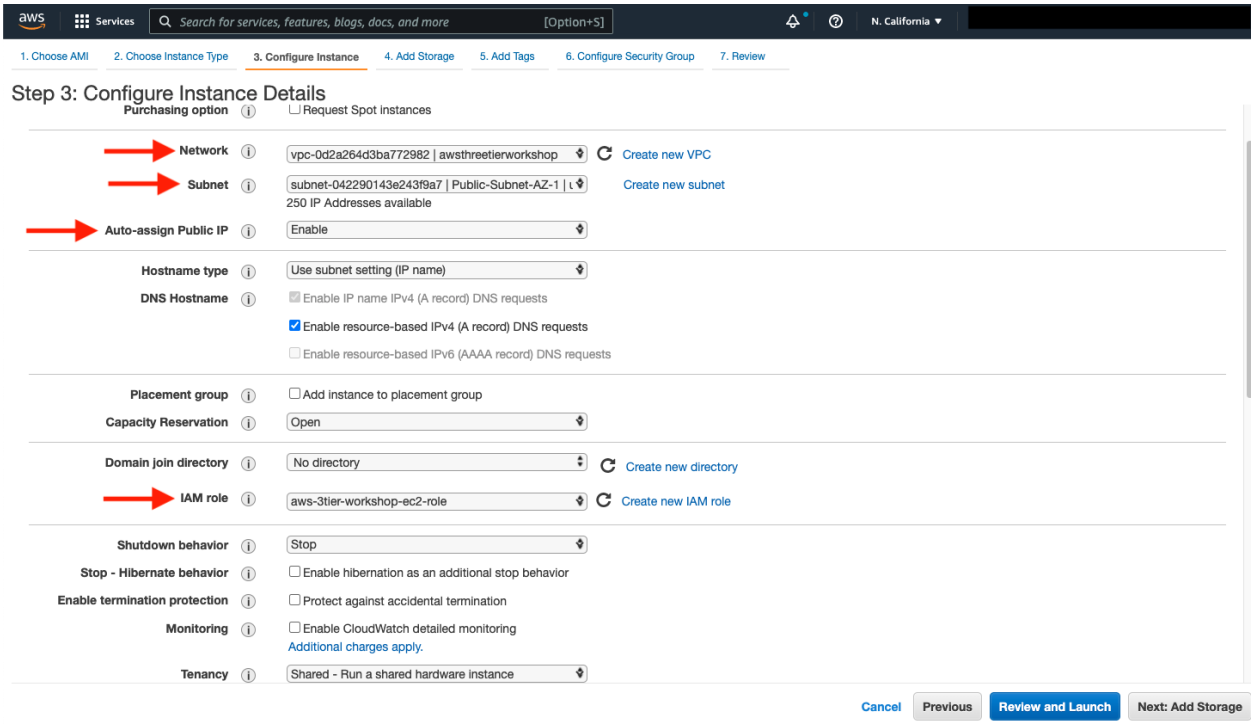

Project2 - Deploy a 3 Tier Archi
Siddhartha082/AWS--3-Tier--w
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part5
aws workshop studio
AWS Three Tier Web Architecture
Part 5: Web Tier Instance Deployment
Part 5: Web Tier Instance Deployment
In this section we will deploy an EC2 instance for the web tier and make all necessary software configurations for the NGINX web server and React.js website.
Learning Objectives
Update NGINX Configuration Files
Create Web Tier Instance
Configure Software Stack
Previous
Next
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
27°C Mostly cloudy
ENG
17:24
09-07-2024

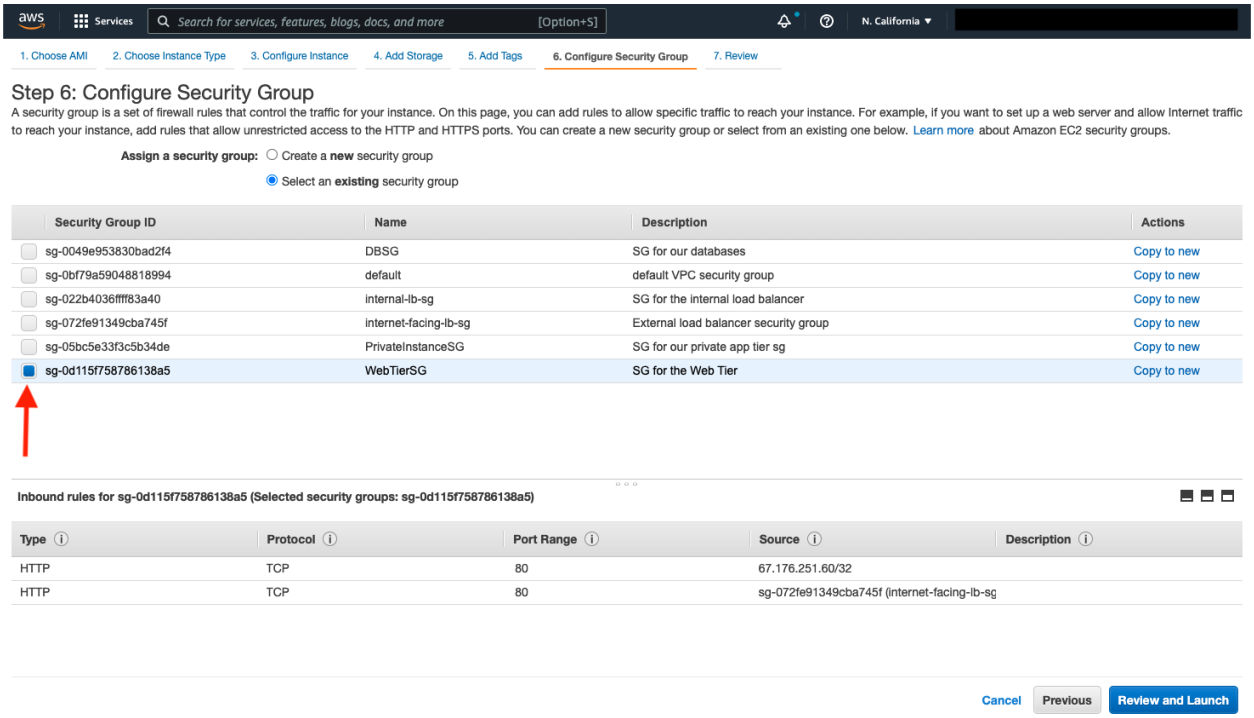

Project2 - Deploy a 3 Tier Archi
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part5/updateconfig
aws workshop studio
Update Config File
Before we create and configure the web instances, open up the application-code/nginx.conf file from the repo we downloaded. Scroll down to line 58 and replace [INTERNAL-LOADBALANCER-DNS] with your internal load balancer's DNS entry. You can find this by navigating to your internal load balancer's details page.
40 listen [::]:80;
41 server_name _;
42
43 #health check
44 location /health {
45 default_type text/html;
46 return 200 "<!DOCTYPE html> Web Tier Health Check \n";
47 }
48
49 #react app and front end files
50 location / {
51 root /home/ec2-user/web-tier/build;
52 index index.html index.htm
53 try_files $uri /index.html;
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
27°C Mostly cloudy
ENG
17:24
09-07-2024


Project2 - Deploy a 3 Tier Archi
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part5/updateconfig
aws workshop studio
server_name _;
#health check
location /health {
default_type text/html;
return 200 "<!DOCTYPE html> Web Tier Health Check \n";
}
#react app and front end files
location / {
root /home/ec2-user/web-tier/build;
index index.html index.htm
try_files $uri /index.html;
}
#proxy for internal lb
location /api/{
proxy_pass http://[REPLACE-WITH-INTERNAL-LB-DNS]:80/;
}
}
# Settings for a TLS enabled server.
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
27°C Mostly cloudy
ENG
17:24
09-07-2024

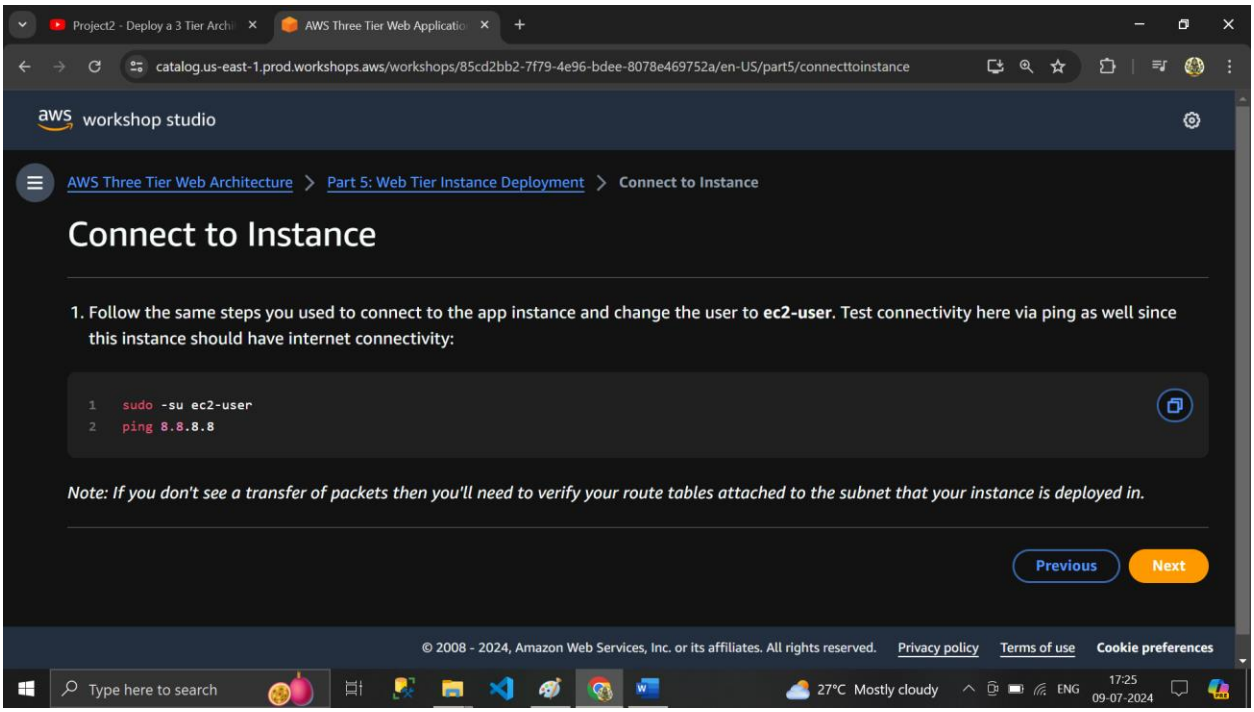

Project2 - Deploy a 3 Tier Archi
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part5/updateconfig
aws workshop studio
try_files $uri /index.html;
}
#proxy for internal lb
location /api/{
proxy_pass http://[REPLACE-WITH-INTERNAL-LB-DNS]:80/;
}
}
# Settings for a TLS enabled server.
# server {
# listen 443 ssl http2;
# listen [::]:443 ssl http2;
# server_name _;
# root /usr/share/nginx/html;
Then, upload this file and the application-code/web-tier folder to the s3 bucket you created for this lab.
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
27°C Mostly cloudy
ENG
17:24
09-07-2024

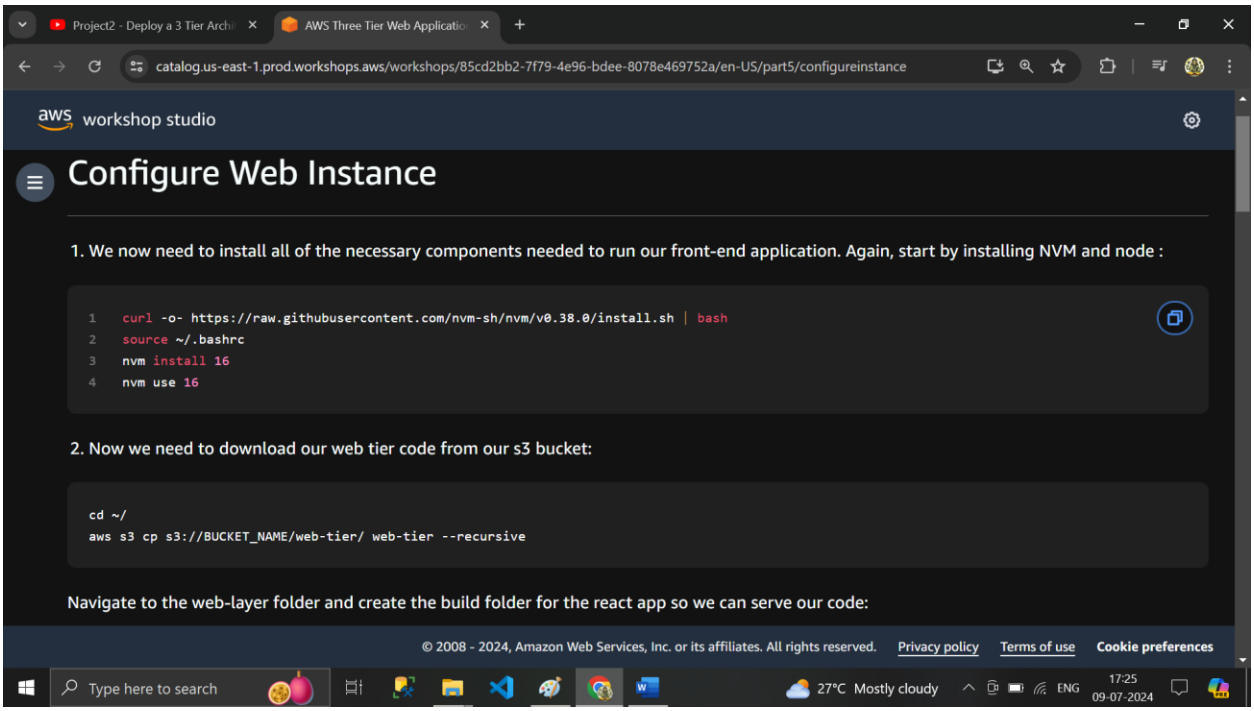

Project2 - Deploy a 3 Tier Archi
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part5/createinstance
aws workshop studio
AWS Three Tier Web Architecture > Part 5: Web Tier Instance Deployment > Web Instance Deployment
Web Instance Deployment
1. Follow the same instance creation instructions we used for the App Tier instance in Part 3: App Tier Instance Deployment, with the exception of the subnet. We will be provisioning this instance in one of our public subnets. Make sure to select the correct network components, security group, and IAM role. This time, auto-assign a public ip on the Configure Instance Details page. Remember to tag the instance with a name so we can identify it more easily.
Save
Services
Search for services, features, blogs, docs, and more
[Option+S]
N. California
1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review
Step 3: Configure Instance Details
Purchasing option
Request Spot instances
Network
vpc-0d2a264d3ba772982 | awsthreetierworkshop
Create new VPC
Subnet
subnet-042290143e243f9a7 | Public-Subnet-AZ-1
Create new subnet
250 IP Addresses available
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
27°C Mostly cloudy
ENG
17:24
09-07-2024

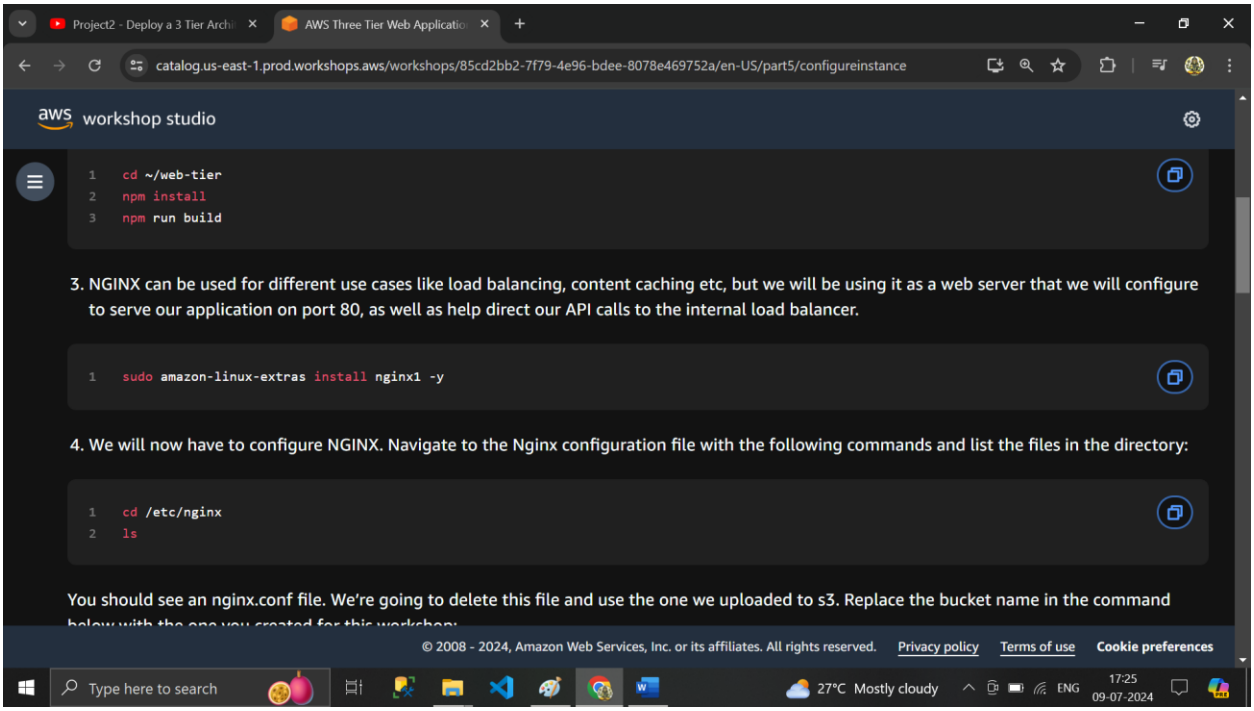

aws
Services
Search for services, features, blogs, docs, and more
[Option+S]
N. California
1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review
Step 3: Configure Instance Details
Purchasing option
Request Spot instances
Network
vpc-0d2a264d3ba772982 | awsthreetierworkshop
Create new VPC
Subnet
subnet-042290143e243f9a7 | Public-Subnet-AZ-1
Create new subnet
250 IP Addresses available
Auto-assign Public IP
Enable
Hostname type
Use subnet setting (IP name)
DNS Hostname
Enable IP name IPv4 (A record) DNS requests
Enable resource-based IPv4 (A record) DNS requests
Enable resource-based IPv6 (AAAA record) DNS requests
Placement group
Add instance to placement group
Capacity Reservation
Open
Domain join directory
No directory
Create new directory
IAM role
aws-3tier-workshop-ec2-role
Create new IAM role
Shutdown behavior
Stop
Stop - Hibernate behavior
Enable hibernation as an additional stop behavior
Enable termination protection
Protect against accidental termination
Monitoring
Enable CloudWatch detailed monitoring
Additional charges apply.
Tenancy
Shared - Run a shared hardware instance
Cancel
Previous
Review and Launch
Next: Add Storage

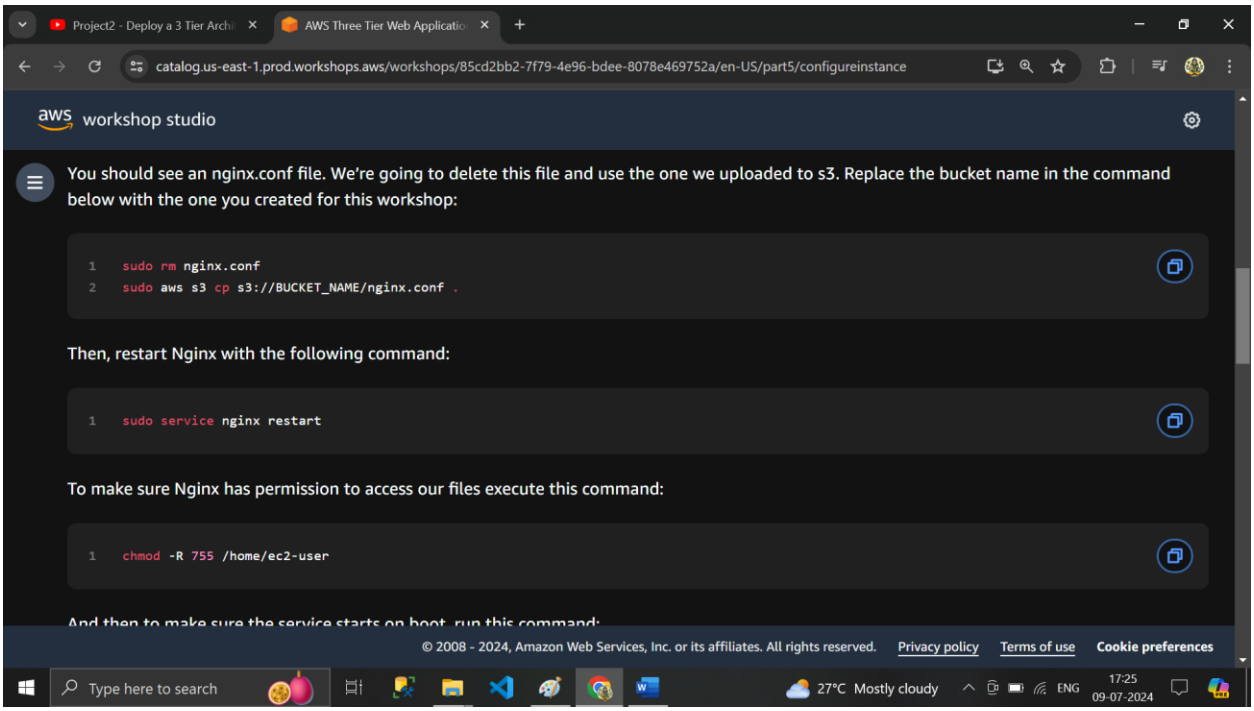


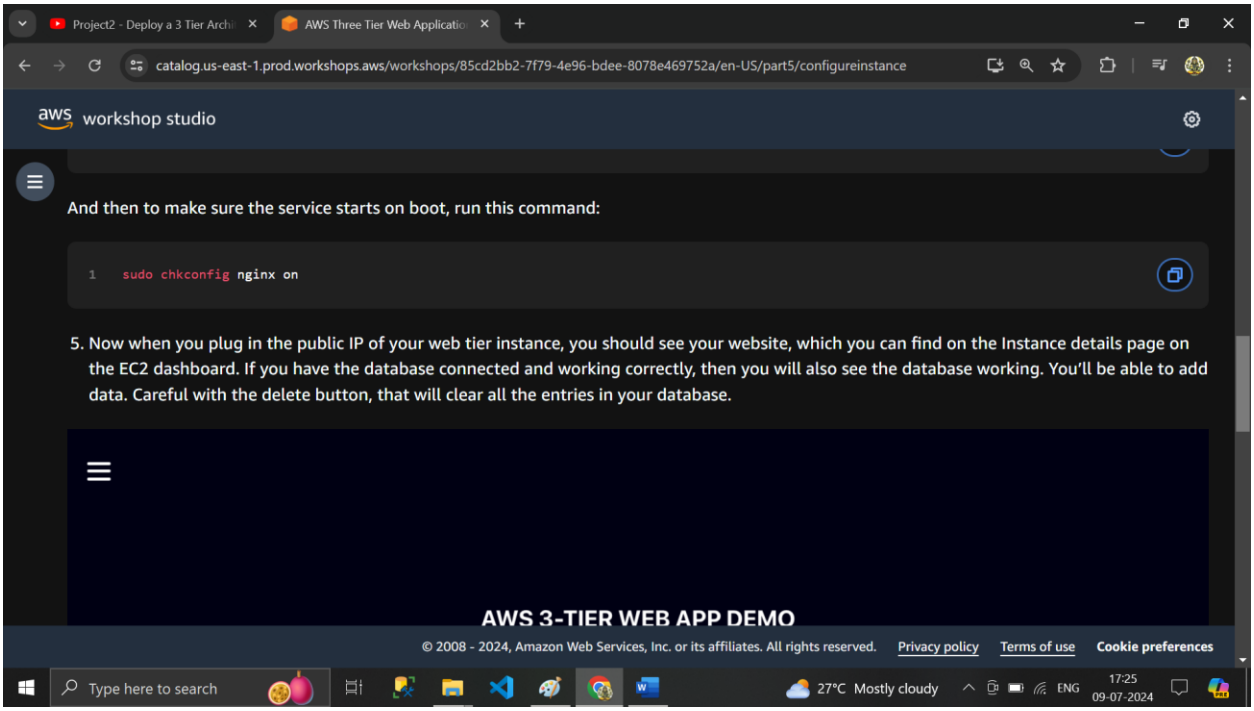
Then at the end, proceed without a key pair for this instance.

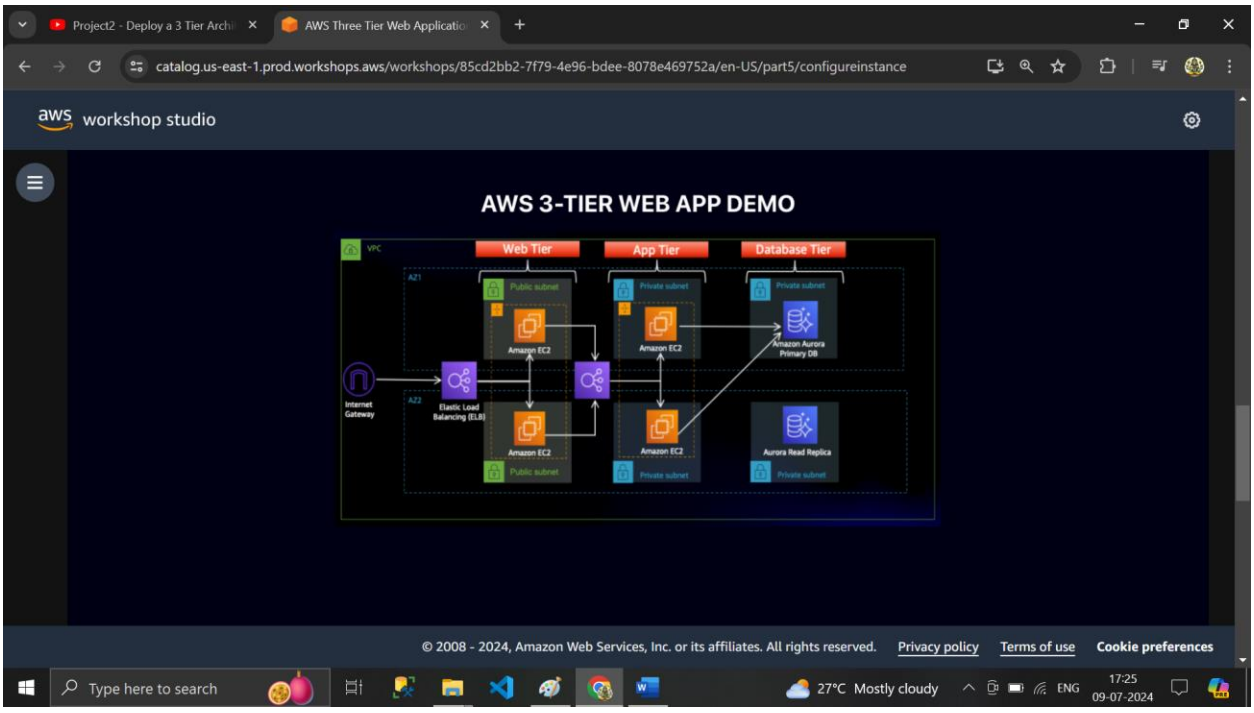


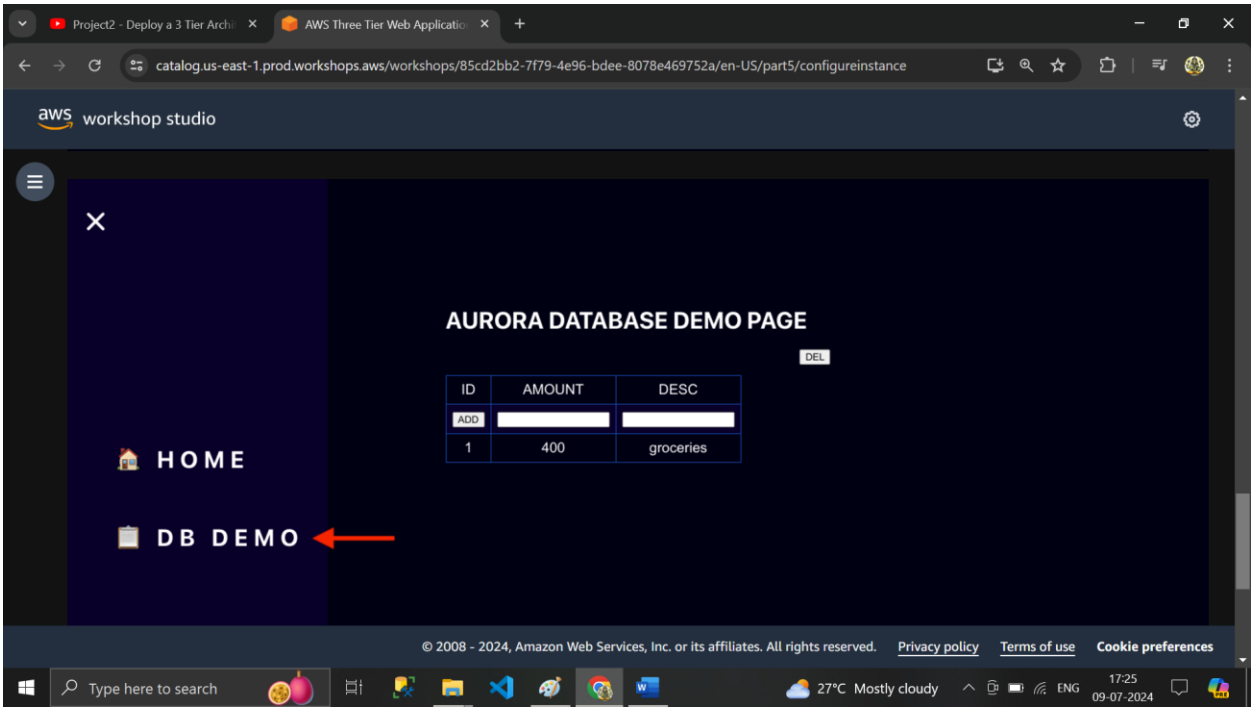

Project2 - Deploy a 3 Tier Archi
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part5/configureinstance
aws workshop studio
Configure Web Instance
1. We now need to install all of the necessary components needed to run our front-end application. Again, start by installing NVM and node :
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.38.0/install.sh | bash
source ~/.bashrc
nvm install 16
nvm use 16
2. Now we need to download our web tier code from our s3 bucket:
cd ~/
aws s3 cp s3://BUCKET_NAME/web-tier/ web-tier --recursive
Navigate to the web-layer folder and create the build folder for the react app so we can serve our code:
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
27°C Mostly cloudy
ENG
17:25
09-07-2024


Project2 - Deploy a 3 Tier Archi
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part5/configureinstance
aws workshop studio
cd ~/web-tier
npm install
npm run build
3. NGINX can be used for different use cases like load balancing, content caching etc, but we will be using it as a web server that we will configure to serve our application on port 80, as well as help direct our API calls to the internal load balancer.
sudo amazon-linux-extras install nginx1 -y
4. We will now have to configure NGINX. Navigate to the Nginx configuration file with the following commands and list the files in the directory:
cd /etc/nginx
ls
You should see an nginx.conf file. We're going to delete this file and use the one we uploaded to s3. Replace the bucket name in the command
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
27°C Mostly cloudy
ENG
17:25
09-07-2024


Project2 - Deploy a 3 Tier Archi
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part5/configureinstance
aws workshop studio
You should see an nginx.conf file. We're going to delete this file and use the one we uploaded to s3. Replace the bucket name in the command below with the one you created for this workshop:
1 sudo rm nginx.conf
2 sudo aws s3 cp s3://BUCKET_NAME/nginx.conf .
Then, restart Nginx with the following command:
1 sudo service nginx restart
To make sure Nginx has permission to access our files execute this command:
1 chmod -R 755 /home/ec2-user
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
27°C Mostly cloudy
ENG
17:25
09-07-2024


Project2 - Deploy a 3 Tier Archi
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part5/configureinstance
aws workshop studio
And then to make sure the service starts on boot, run this command:
1 sudo chkconfig nginx on
5. Now when you plug in the public IP of your web tier instance, you should see your website, which you can find on the Instance details page on the EC2 dashboard. If you have the database connected and working correctly, then you will also see the database working. You'll be able to add data. Careful with the delete button, that will clear all the entries in your database.
AWS 3-TIER WEB APP DEMO
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
27°C Mostly cloudy
ENG
17:25
09-07-2024


Project2 - Deploy a 3 Tier Archi
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part5/configureinstance
aws workshop studio
AWS 3-TIER WEB APP DEMO
Web Tier
App Tier
Database Tier
VPC
AZ1
AZ2
Public subnet
Private subnet
Amazon EC2
Amazon Aurora Primary DB
Aurora Read Replica
Internet Gateway
Elastic Load Balancing (ELB)
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
27°C Mostly cloudy
ENG
17:25
09-07-2024


Project2 - Deploy a 3 Tier Archi
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part5/configureinstance
aws workshop studio
AURORA DATABASE DEMO PAGE
DEL
ID
AMOUNT
DESC
ADD
1
400
groceries
HOME
DB DEMO
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
27°C Mostly cloudy
ENG
17:25
09-07-2024

Project2 - Deploy a 3 Tier Archi | AWS Three Tier Web Applicatio

catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part6

aws workshop studio

AWS Three Tier Web Architecture > Part 6: External Load Balancer and Auto Scaling

Part 6: External Load Balancer and Auto Scaling

In this section of the workshop we will create an Amazon Machine Image (AMI) of the web tier instance we just created, and use that to set up autoscaling with an external facing load balancer in order to make this tier highly available.

Learning Objectives:

- Create an AMI of our Web Tier
- Create a Launch Template
- Configure Auto Scaling
- Deploy External Load Balancer

Previous Next

Type here to search 27°C Mostly cloudy ENG 17:25 09-07-2024

Project2 - Deploy a 3 Tier Archi | AWS Three Tier Web Applicatio

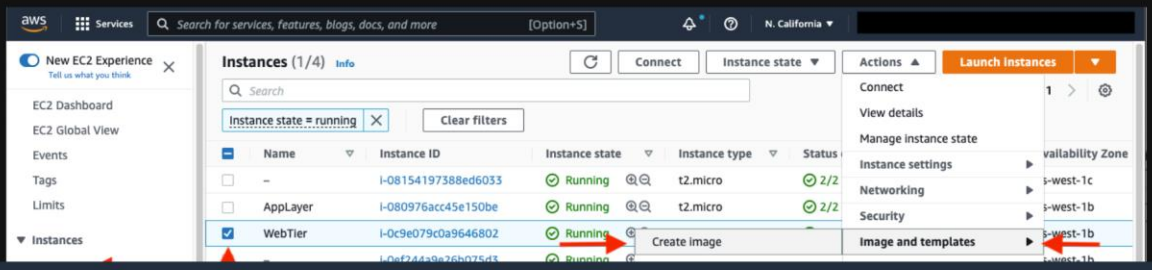
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part6/createami

aws workshop studio

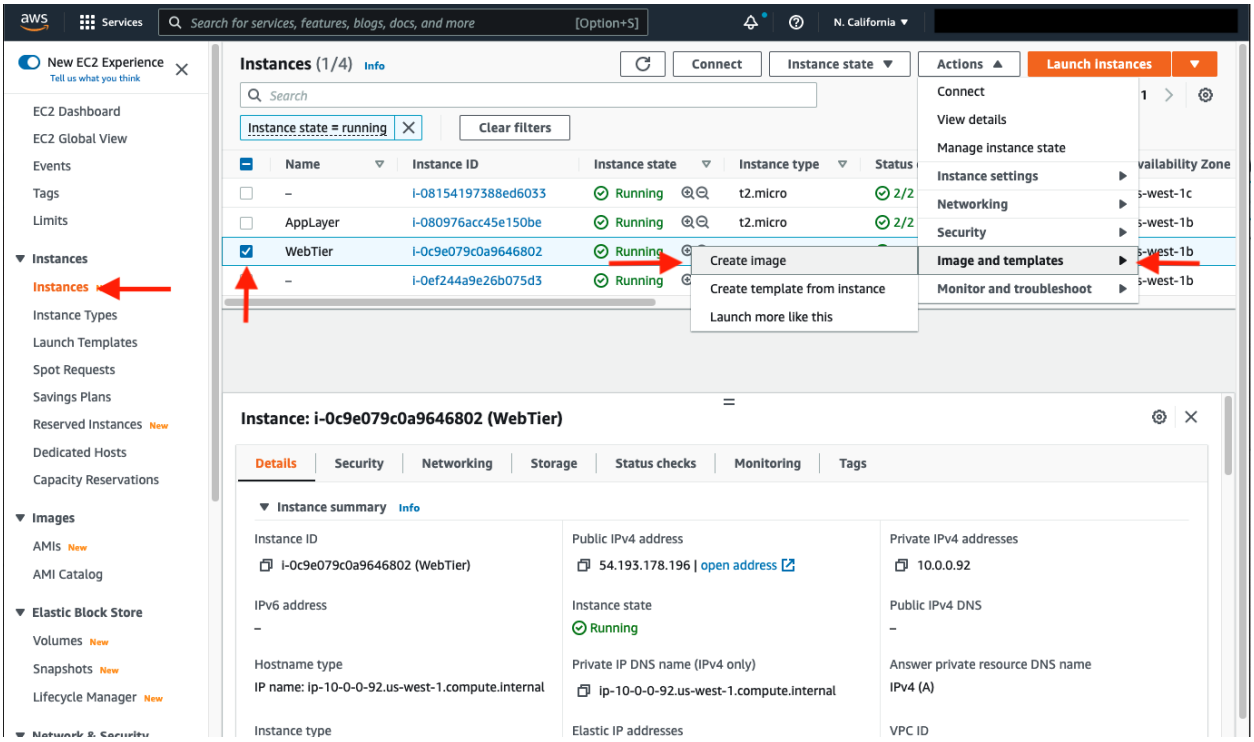
AWS Three Tier Web Architecture > Part 6: External Load Balancer and Auto Scaling > Web Tier AMI

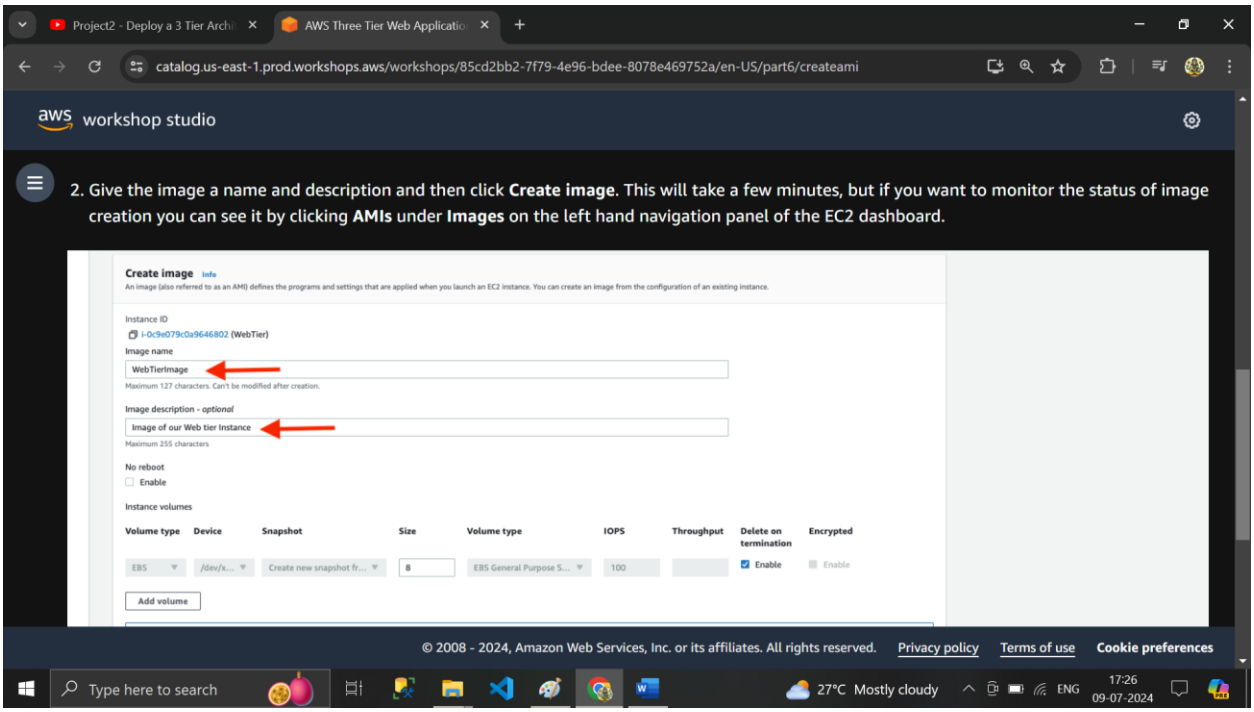
Web Tier AMI

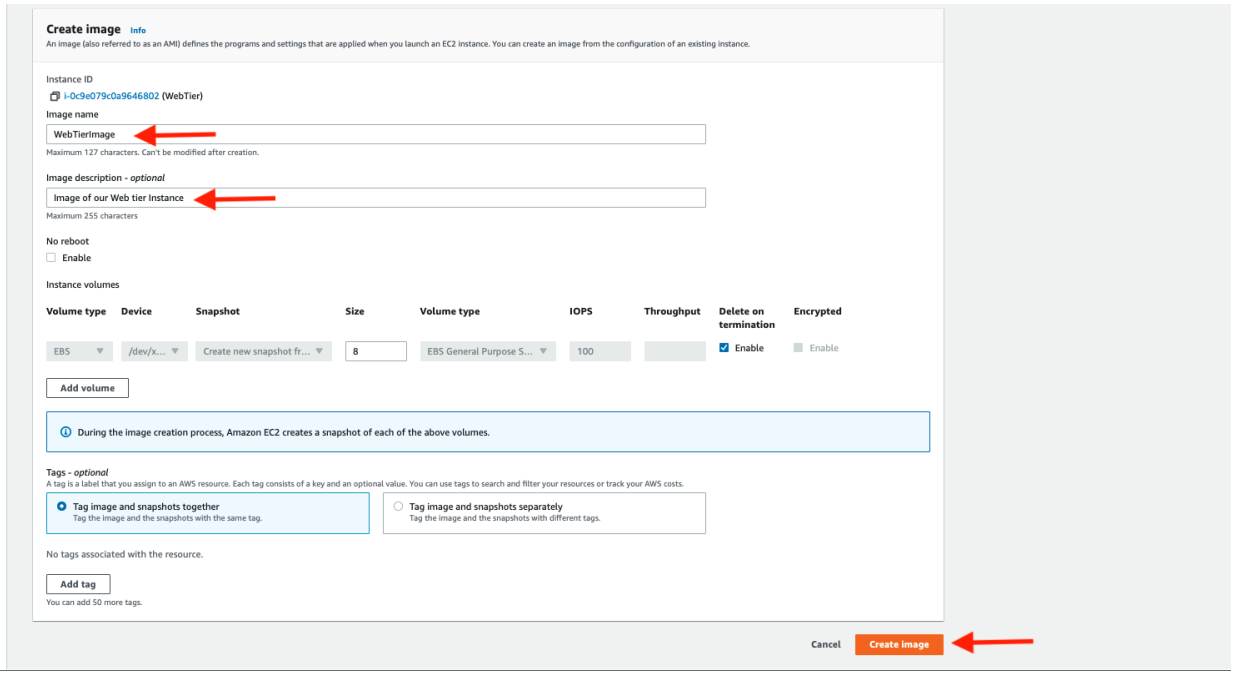
1. Navigate to **Instances** on the left hand side of the EC2 dashboard. Select the web tier instance we created and under **Actions** select **Image and templates**. Click **Create Image**.

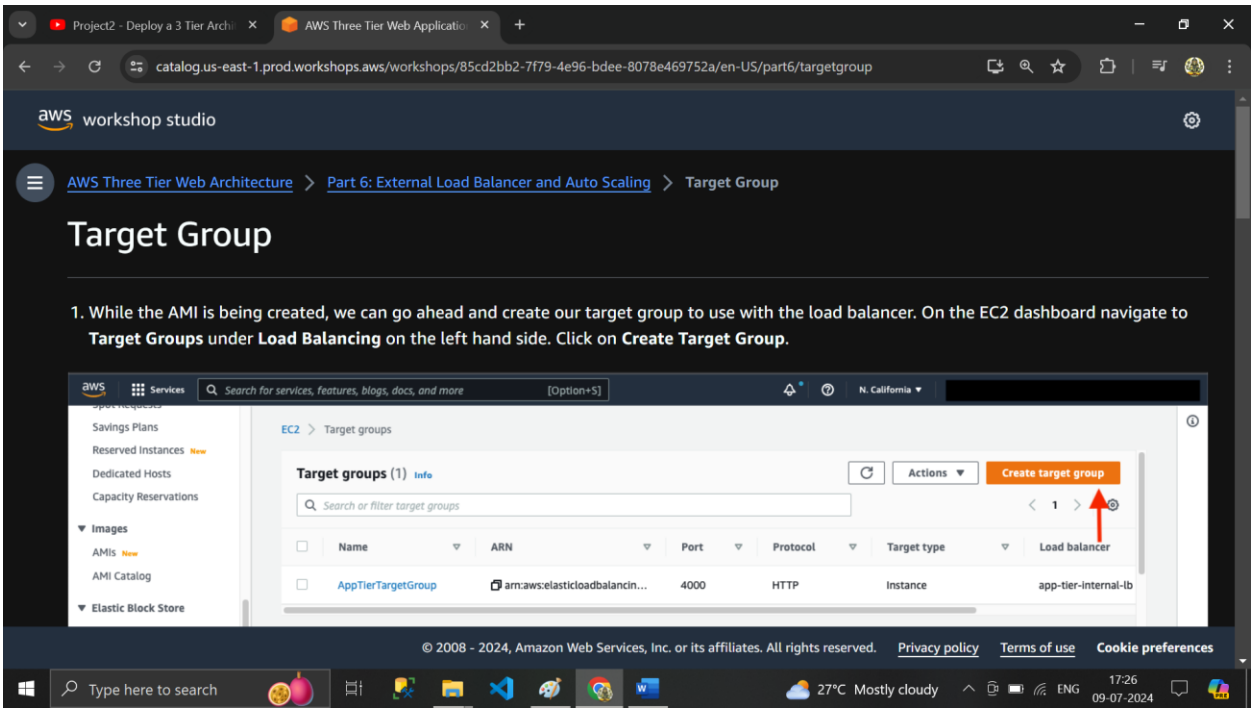


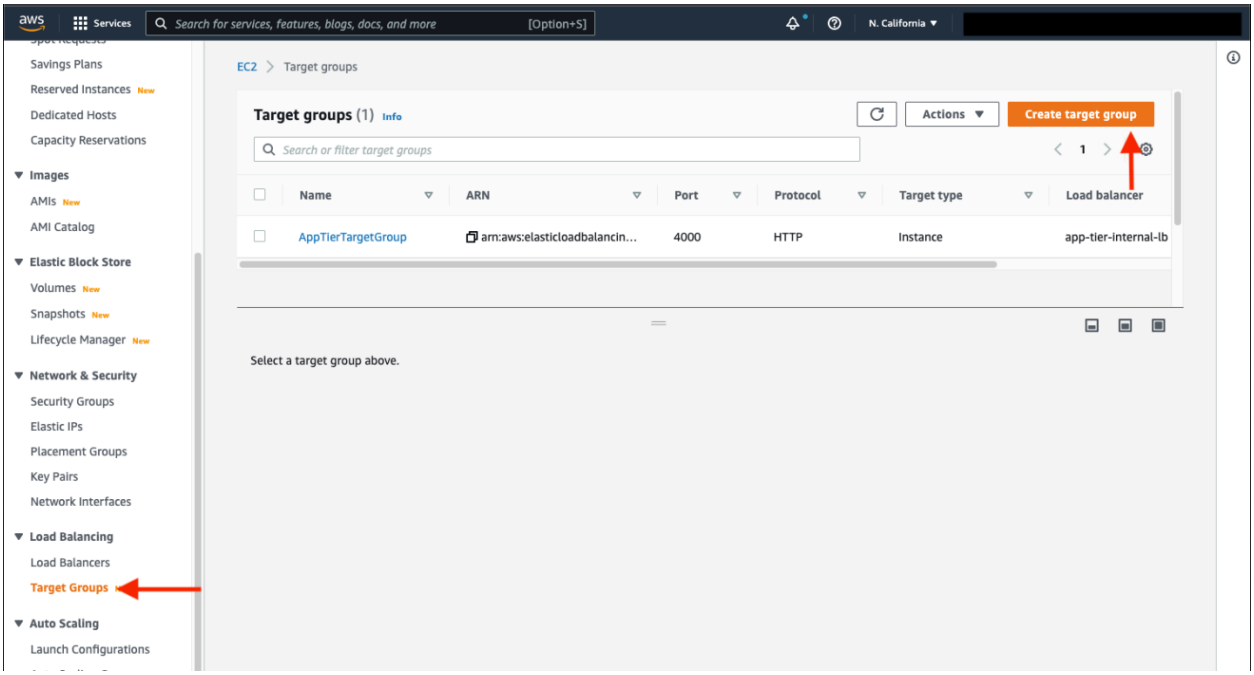
Type here to search 27°C Mostly cloudy ENG 17:26 09-07-2024


Instances (1/4) Info
Connect
Instance state
Actions
Launch Instances
Instance state = running
Clear filters
Name
Instance ID
Instance state
Instance type
Status
AppLayer
i-080976acc45e150be
WebTier
i-0c9e079c0a9646802
Create image
Create template from instance
Launch more like this
Image and templates
Instance: i-0c9e079c0a9646802 (WebTier)
Details
Security
Networking
Storage
Status checks
Monitoring
Tags
Instance summary Info
Instance ID
i-0c9e079c0a9646802 (WebTier)
Public IPv4 address
54.193.178.196 | open address
Private IPv4 addresses
10.0.0.92
IPv6 address
Instance state
Running
Public IPv4 DNS
Hostname type
IP name: ip-10-0-0-92.us-west-1.compute.internal
Private IP DNS name (IPv4 only)
ip-10-0-0-92.us-west-1.compute.internal
Answer private resource DNS name
IPv4 (A)
Instance type
Elastic IP addresses
VPC ID

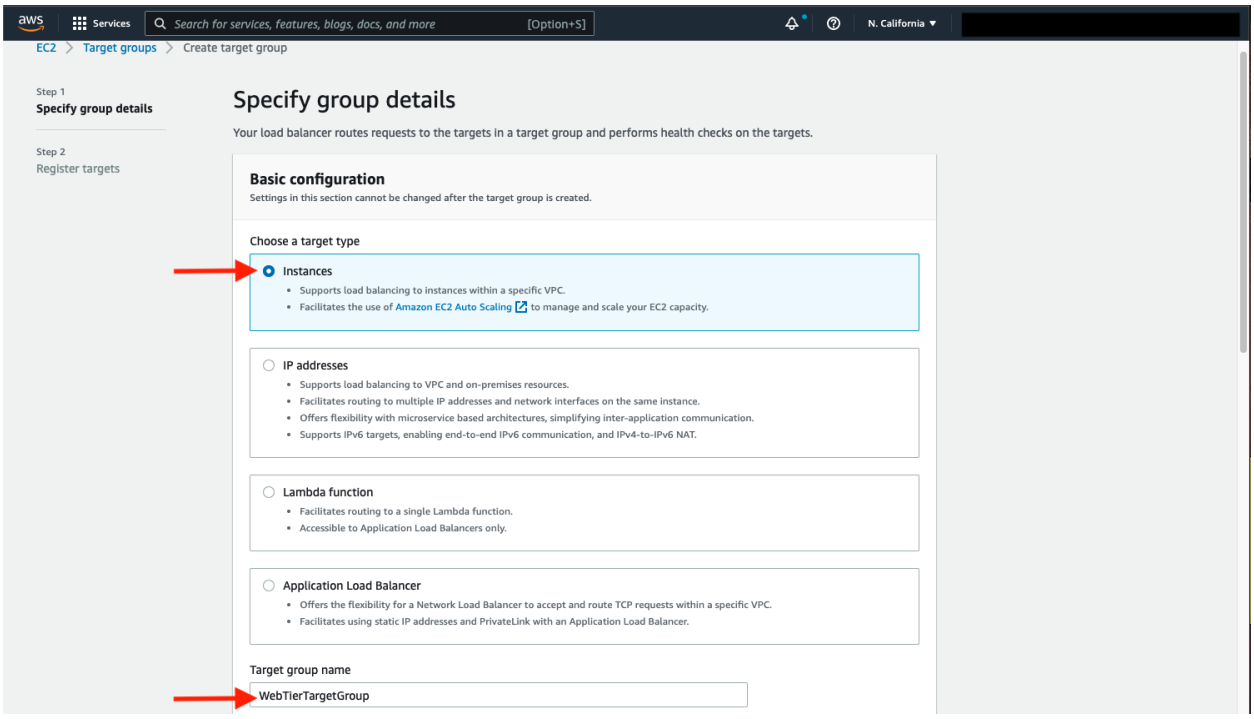

catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part6/createami
aws workshop studio
2. Give the image a name and description and then click Create image. This will take a few minutes, but if you want to monitor the status of image creation you can see it by clicking AMIs under Images on the left hand navigation panel of the EC2 dashboard.
Create image
Image name
WebTierimage
Image description - optional
Image of our Web tier Instance
No reboot
Enable
Instance volumes
Add volume
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences


Create image Info
An image (also referred to as an AMI) defines the programs and settings that are applied when you launch an EC2 instance. You can create an image from the configuration of an existing instance.
Instance ID
i-0c9e079c0a9646802 (WebTier)
Image name
WebTierImage
Maximum 127 characters. Can't be modified after creation.
Image description - optional
Image of our Web tier Instance
Maximum 255 characters
No reboot
Enable
Instance volumes
Volume type
Device
Snapshot
Size
Volume type
IOPS
Throughput
Delete on termination
Encrypted
EBS
/dev/x...
Create new snapshot fr...
8
EBS General Purpose S...
100
Enable
Enable
Add volume
During the image creation process, Amazon EC2 creates a snapshot of each of the above volumes.
Tags - optional
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.
Tag image and snapshots together
Tag the image and the snapshots with the same tag.
Tag image and snapshots separately
Tag the image and the snapshots with different tags.
No tags associated with the resource.
Add tag
You can add 50 more tags.
Cancel
Create image

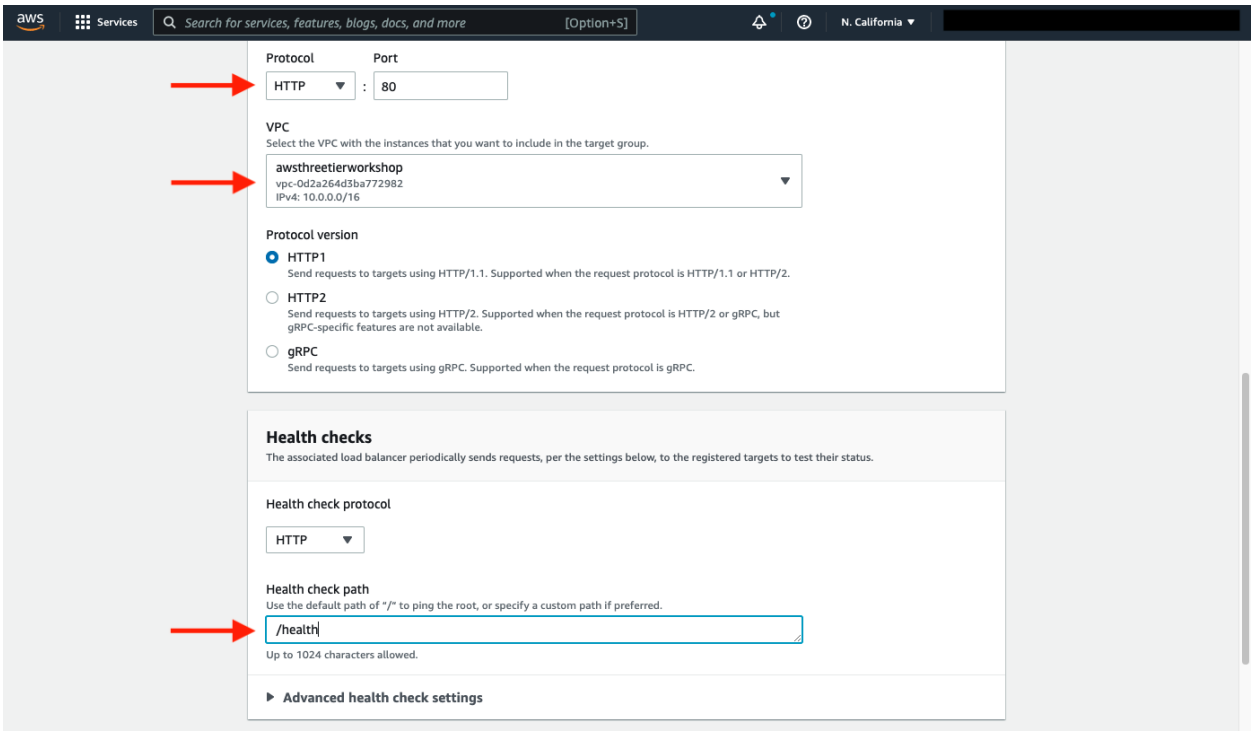

Project2 - Deploy a 3 Tier Archi
AWS Three Tier Web Applicatio
catalog.us-east-1.prod.workshops.aws/workshops/85cd2bb2-7f79-4e96-bdee-8078e469752a/en-US/part6/targetgroup
aws workshop studio
AWS Three Tier Web Architecture > Part 6: External Load Balancer and Auto Scaling > Target Group
Target Group
1. While the AMI is being created, we can go ahead and create our target group to use with the load balancer. On the EC2 dashboard navigate to Target Groups under Load Balancing on the left hand side. Click on Create Target Group.
Services
Search for services, features, blogs, docs, and more
[Option+S]
N. California
Savings Plans
Reserved Instances New
Dedicated Hosts
Capacity Reservations
Images
AMIs New
AMI Catalog
Elastic Block Store
EC2 > Target groups
Target groups (1) Info
Actions
Create target group
Search or filter target groups
Name
ARN
Port
Protocol
Target type
Load balancer
AppTierTargetGroup
arn:aws:elasticloadbalancin...
4000
HTTP
Instance
app-tier-internal-lb
© 2008 - 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy policy
Terms of use
Cookie preferences
Type here to search
27°C Mostly cloudy
ENG
17:26
09-07-2024



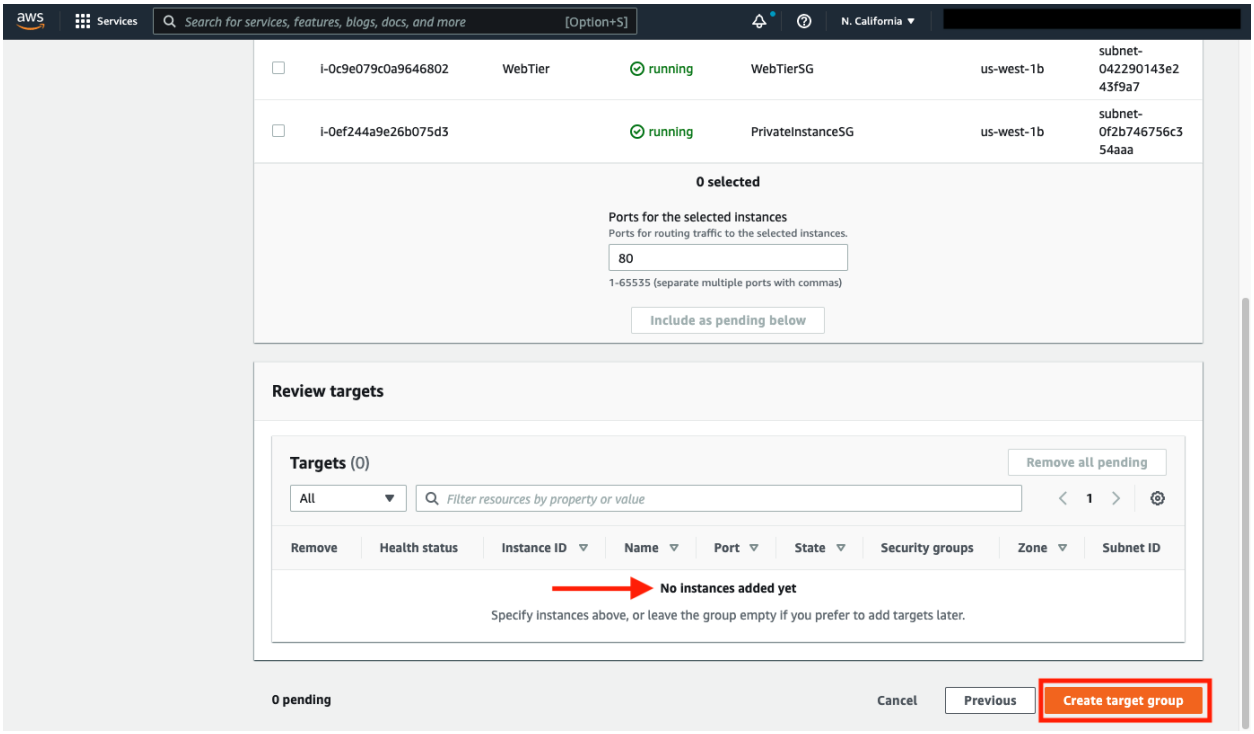
2. The purpose of forming this target group is to use with our load blancer so it may balance traffic across our public web tier instances. Select Instances as the target type and give it a name.



Then, set the protocol to **HTTP** and the port to 80. Remember this is the port NGINX is listening on. Select the VPC we've been using thus far, and then change the health check path to be **/health**. Click **Next**.

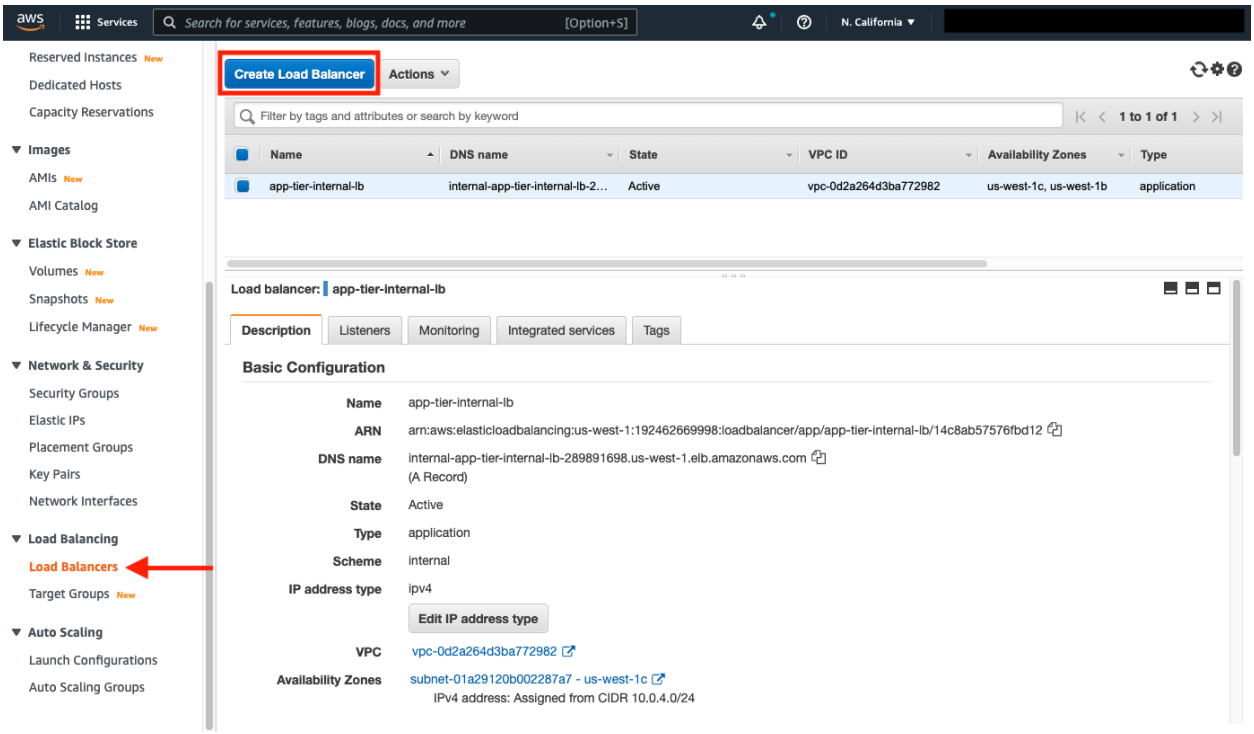


3. We are **NOT** going to register any targets for now, so just skip that step and create the target group.

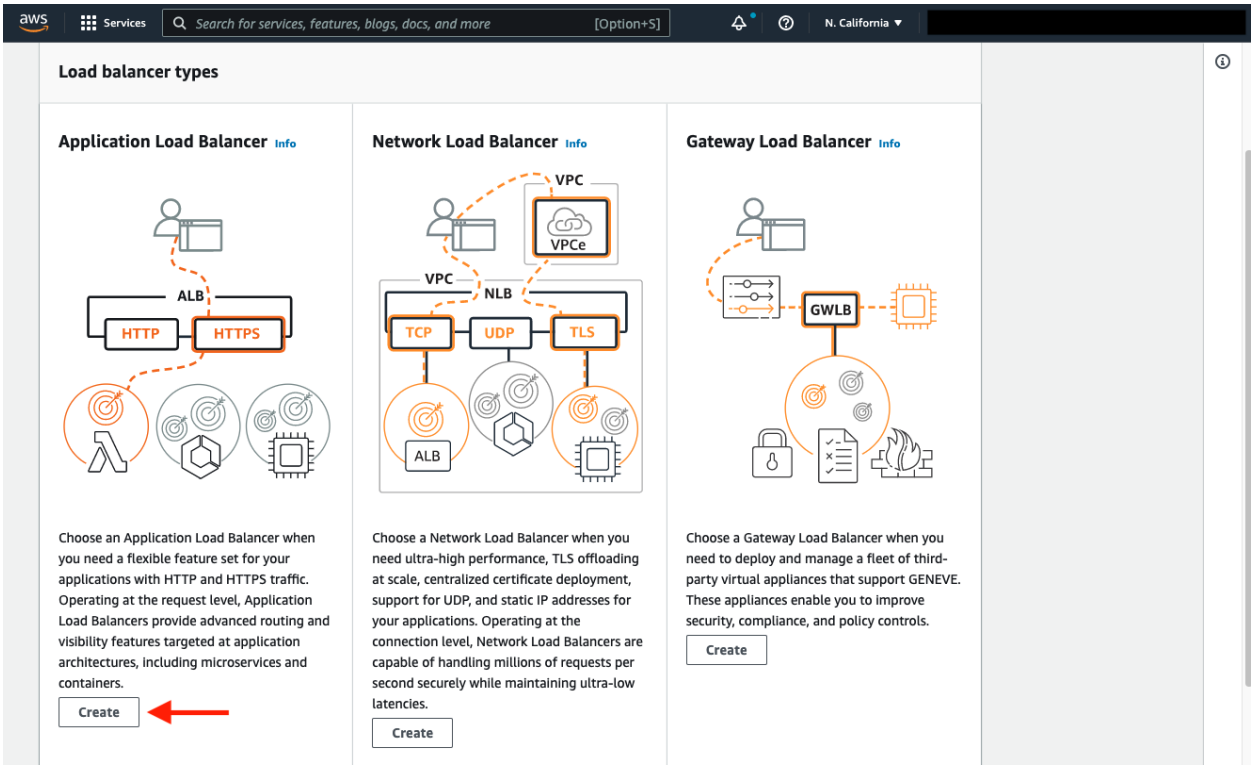


Internet Facing Load Balancer

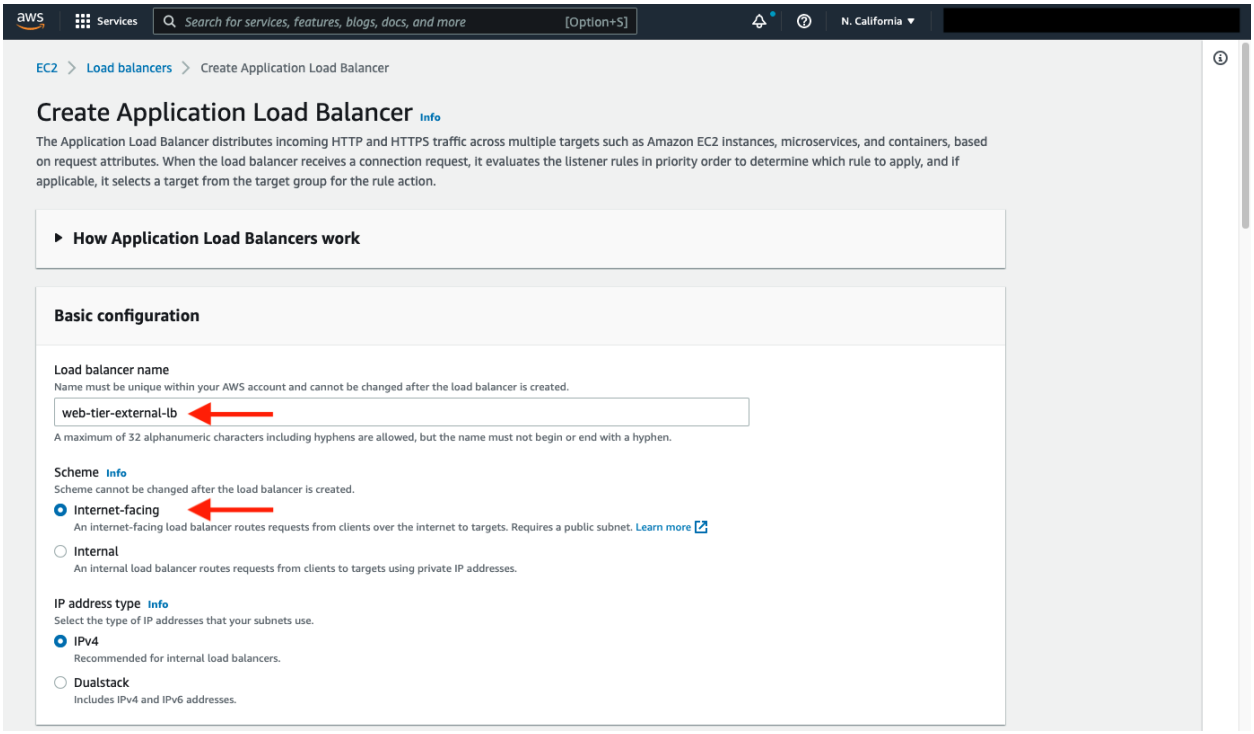
1. On the left hand side of the EC2 dashboard select **Load Balancers** under **Load Balancing** and click **Create Load Balancer**.



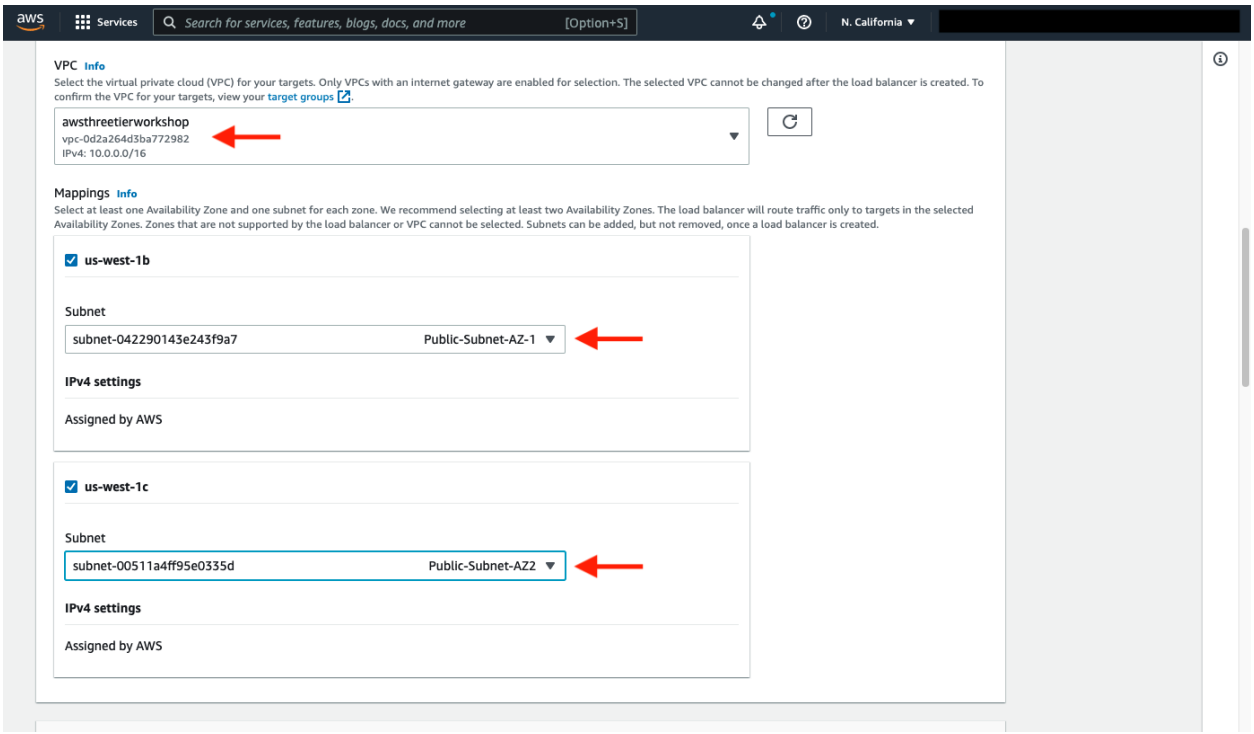
We'll be using an **Application Load Balancer** for our **HTTP** traffic so click the create button for that opt



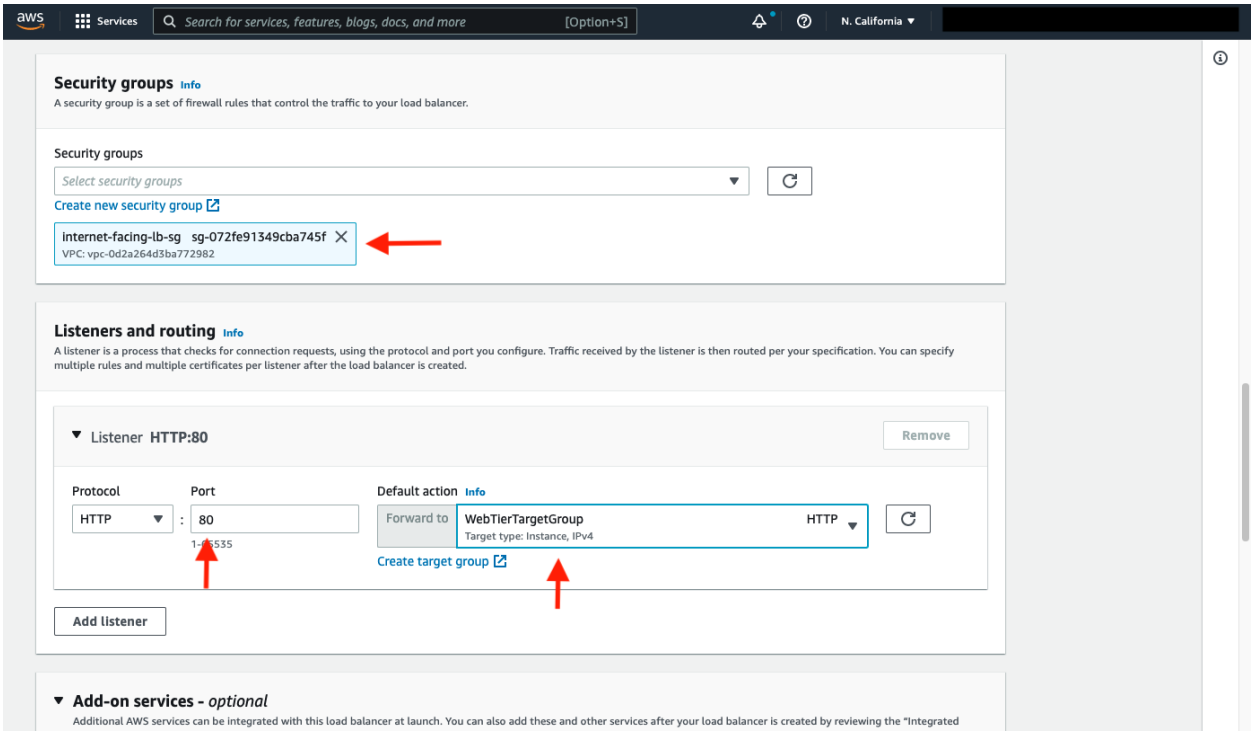
3. After giving the load balancer a name, be sure to select **internet facing** since this one will not be public facing, but rather it will route traffic from our web tier to the app tier.



Select the correct network configuration for VPC and **public** subnets

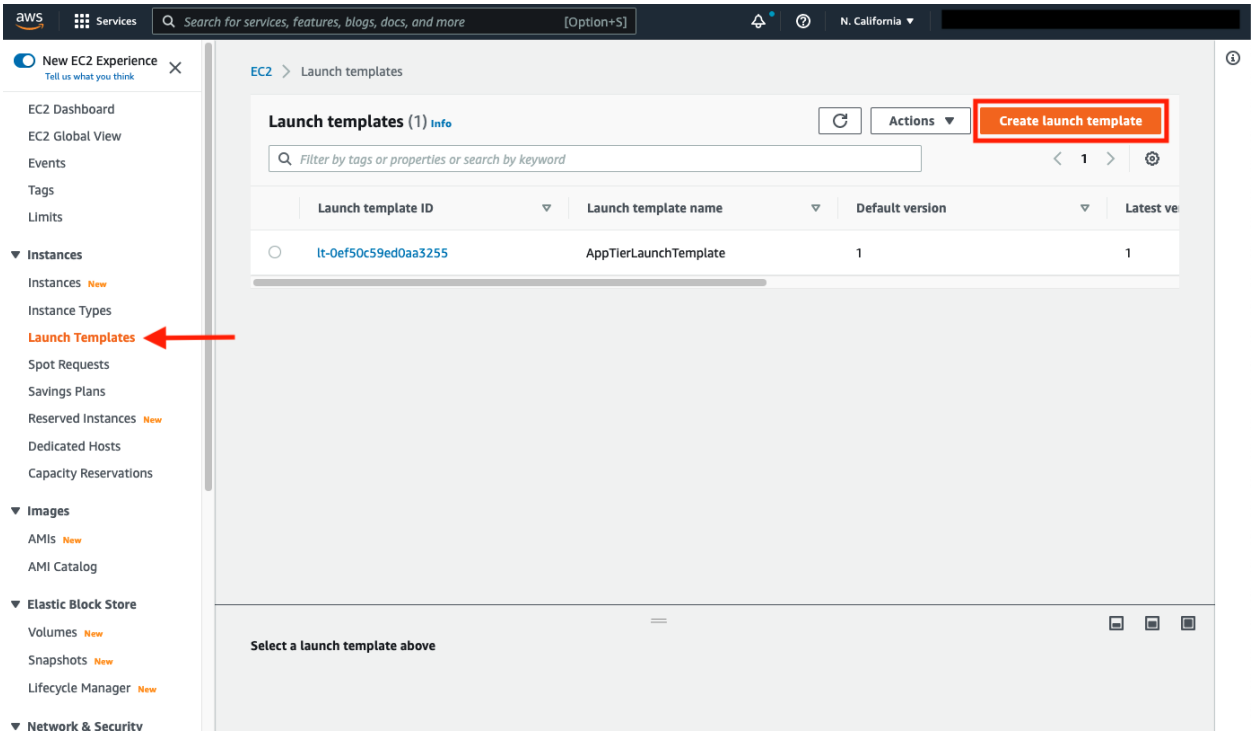


Select the security group we created for this internal ALB. Now, this ALB will be listening for HTTP traffic on port 80. It will be forwarding the traffic to our **target group** that we just created, so select it from the dropdown, and create the load balancer.

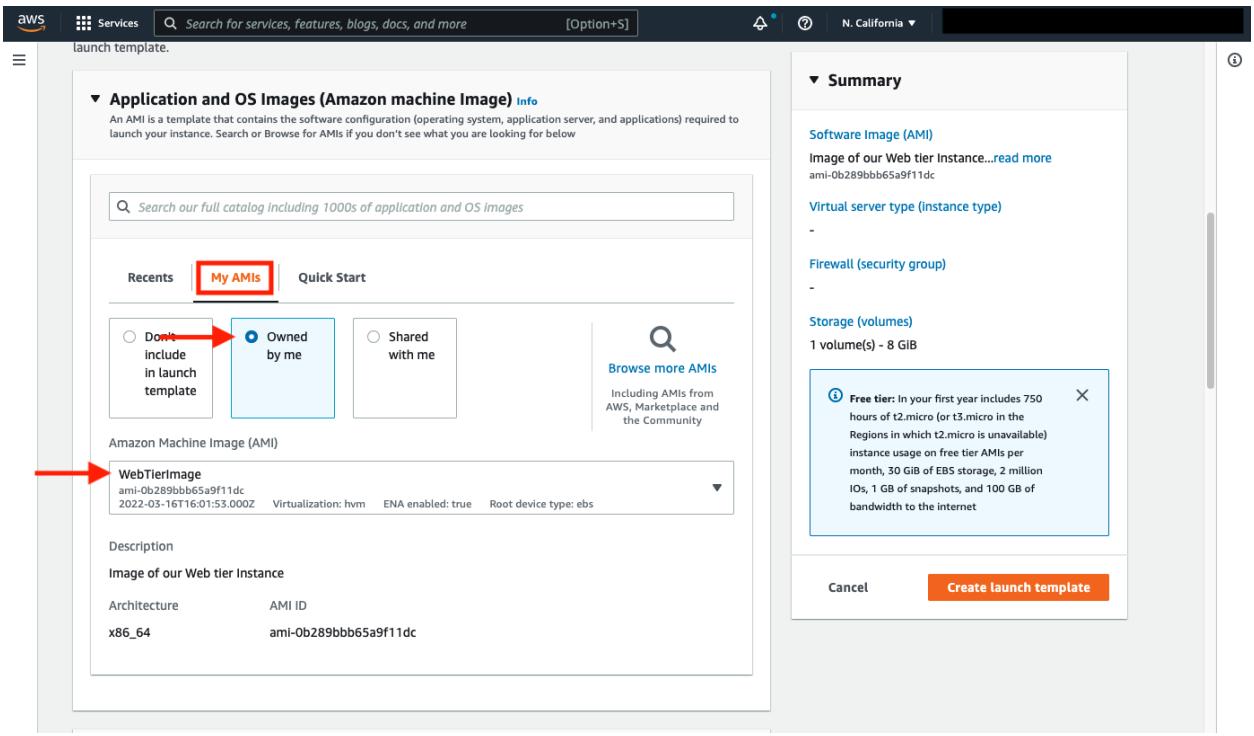


Launch Template

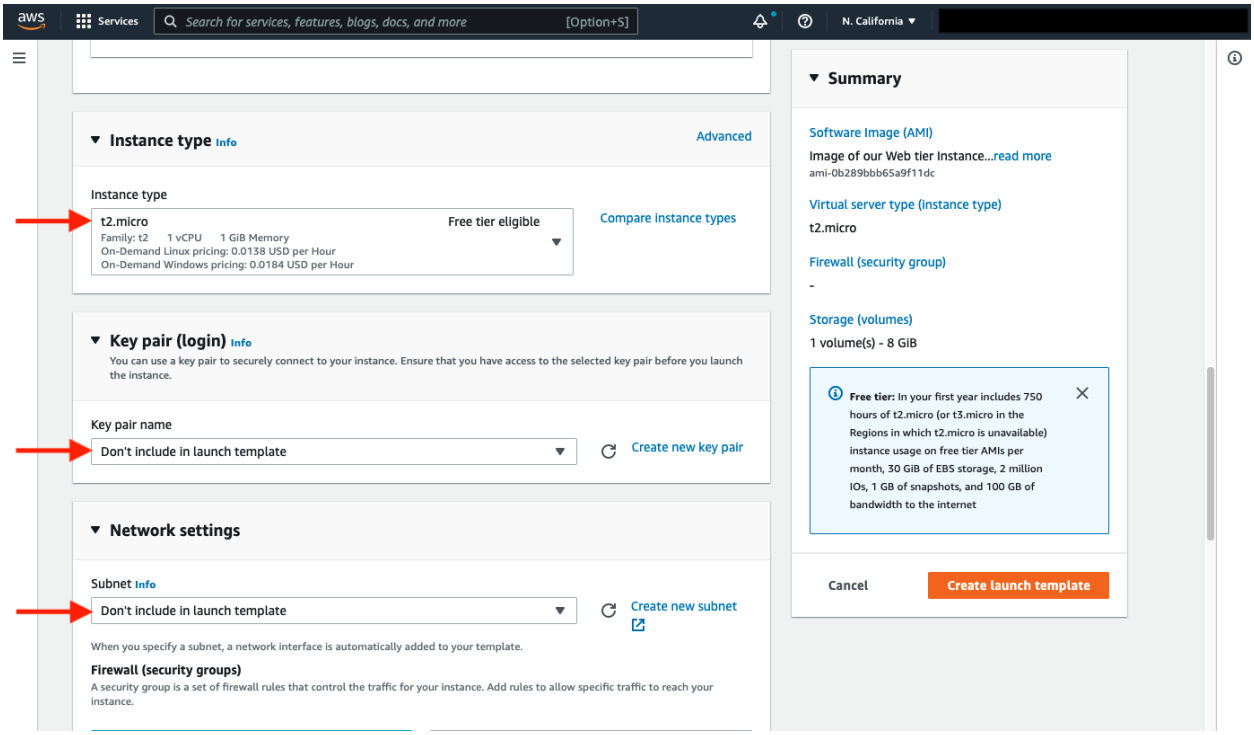
1. Before we configure Auto Scaling, we need to create a Launch template with the AMI we created earlier. On the left side of the EC2 dashboard navigate to **Launch Template** under **Instances** and click **Create Launch Template**



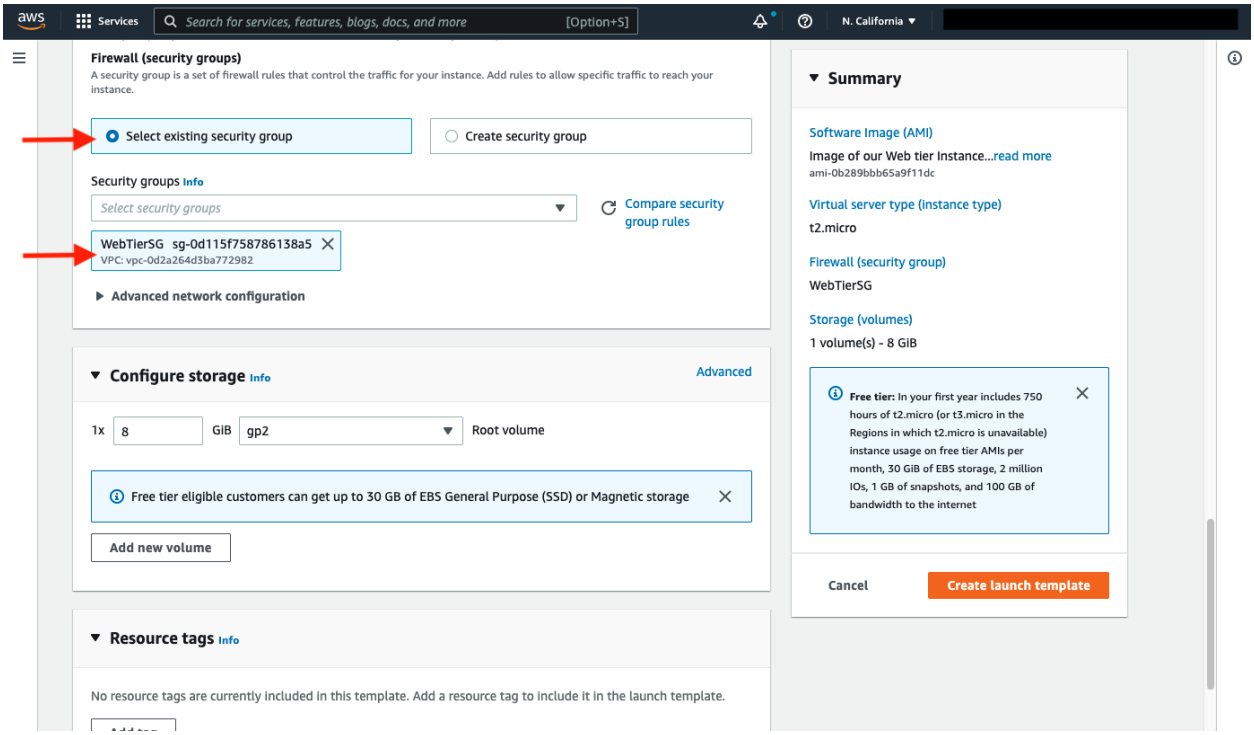
2. Name the Launch Template, and then under **Application and OS Images** include the app tier AMI you created.

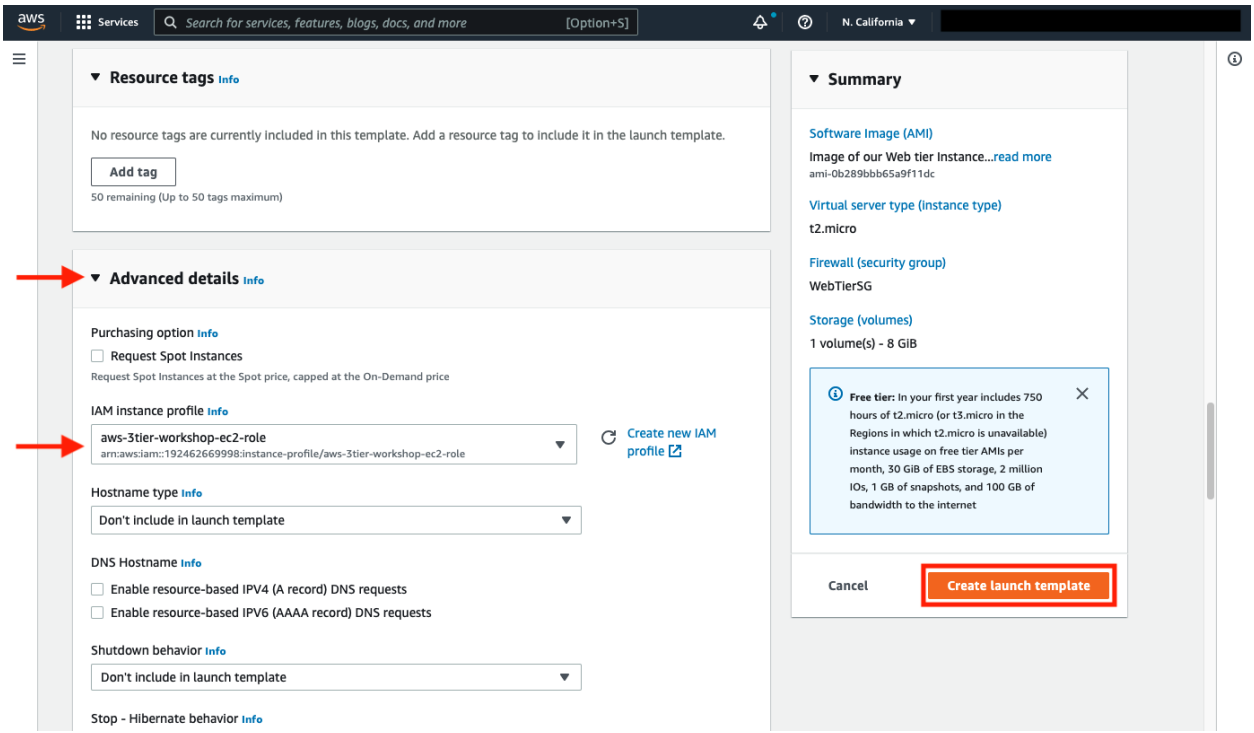


Under **Instance Type** select t2.micro. For **Key pair** and **Network Settings** don't include it in the template. We don't need a key pair to access our instances and we'll be setting the network information in the autoscaling group.



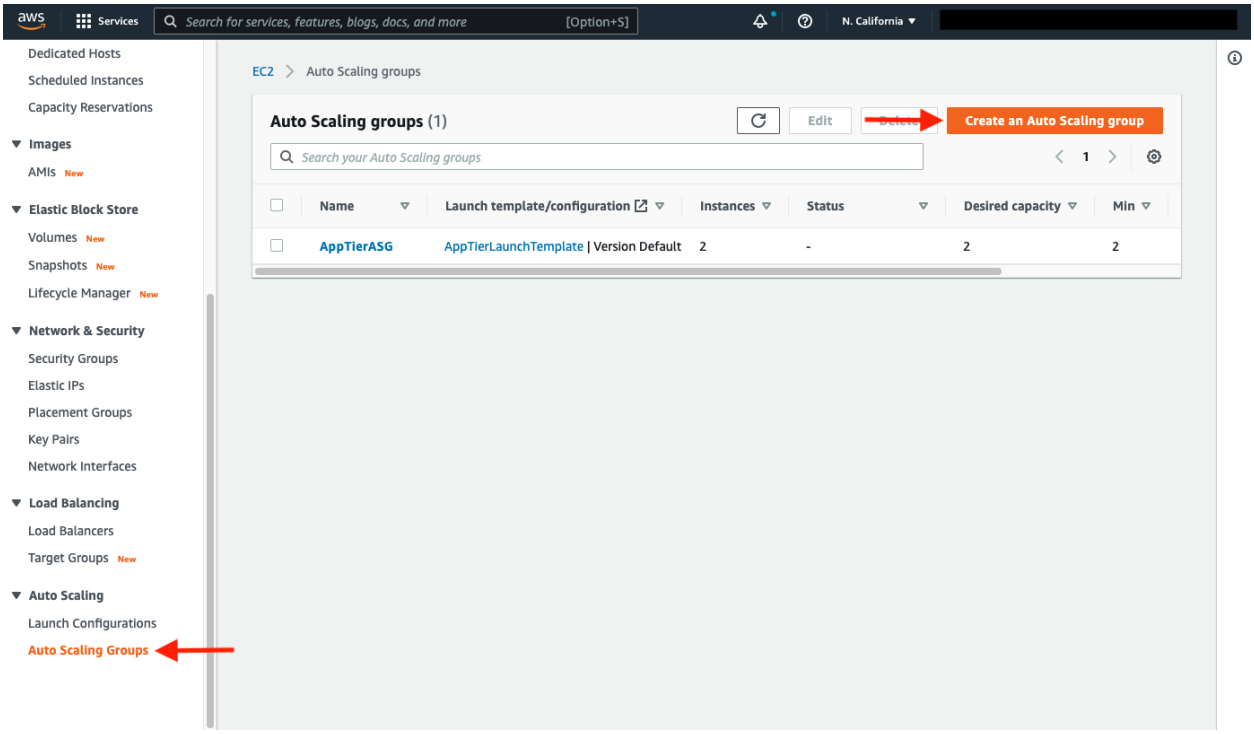
Set the correct security group for our web tier, and then under **Advanced details** use the same IAM instance profile we have been using for our EC2 instances.


aws
Services
Search for services, features, blogs, docs, and more
[Option+S]
N. California
Firewall (security groups)
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.
Select existing security group
Create security group
Security groups Info
Select security groups
Compare security group rules
WebTierSG sg-0d115f758786138a5
VPC: vpc-0d2a264d3ba772982
Advanced network configuration
Configure storage Info
Advanced
1x
8
GiB
gp2
Root volume
Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage
Add new volume
Resource tags Info
No resource tags are currently included in this template. Add a resource tag to include it in the launch template.
Summary
Software Image (AMI)
Image of our Web tier Instance...read more
ami-0b289bbb65a9f11dc
Virtual server type (instance type)
t2.micro
Firewall (security group)
WebTierSG
Storage (volumes)
1 volume(s) - 8 GiB
Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet
Cancel
Create launch template

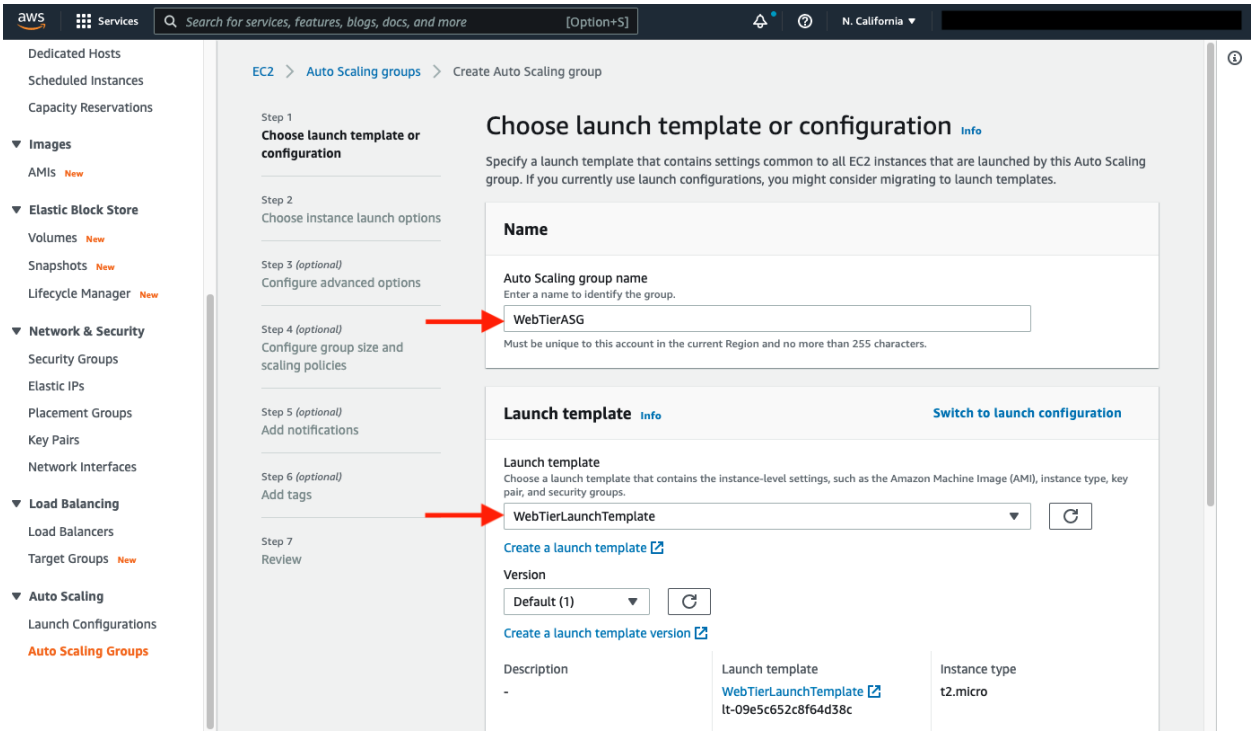

aws
Services
Search for services, features, blogs, docs, and more
[Option+S]
N. California
Resource tags Info
No resource tags are currently included in this template. Add a resource tag to include it in the launch template.
Add tag
50 remaining (Up to 50 tags maximum)
Advanced details Info
Purchasing option Info
Request Spot Instances
Request Spot Instances at the Spot price, capped at the On-Demand price
IAM instance profile Info
aws-3tier-workshop-ec2-role
arn:aws:iam::192462669998:instance-profile/aws-3tier-workshop-ec2-role
Create new IAM profile
Hostname type Info
Don't include in launch template
DNS Hostname Info
Enable resource-based IPV4 (A record) DNS requests
Enable resource-based IPV6 (AAAA record) DNS requests
Shutdown behavior Info
Don't include in launch template
Stop - Hibernate behavior Info
Summary
Software Image (AMI)
Image of our Web tier Instance...read more
ami-0b289bbb65a9f11dc
Virtual server type (instance type)
t2.micro
Firewall (security group)
WebTierSG
Storage (volumes)
1 volume(s) - 8 GiB
Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet
Cancel
Create launch template

Auto Scaling

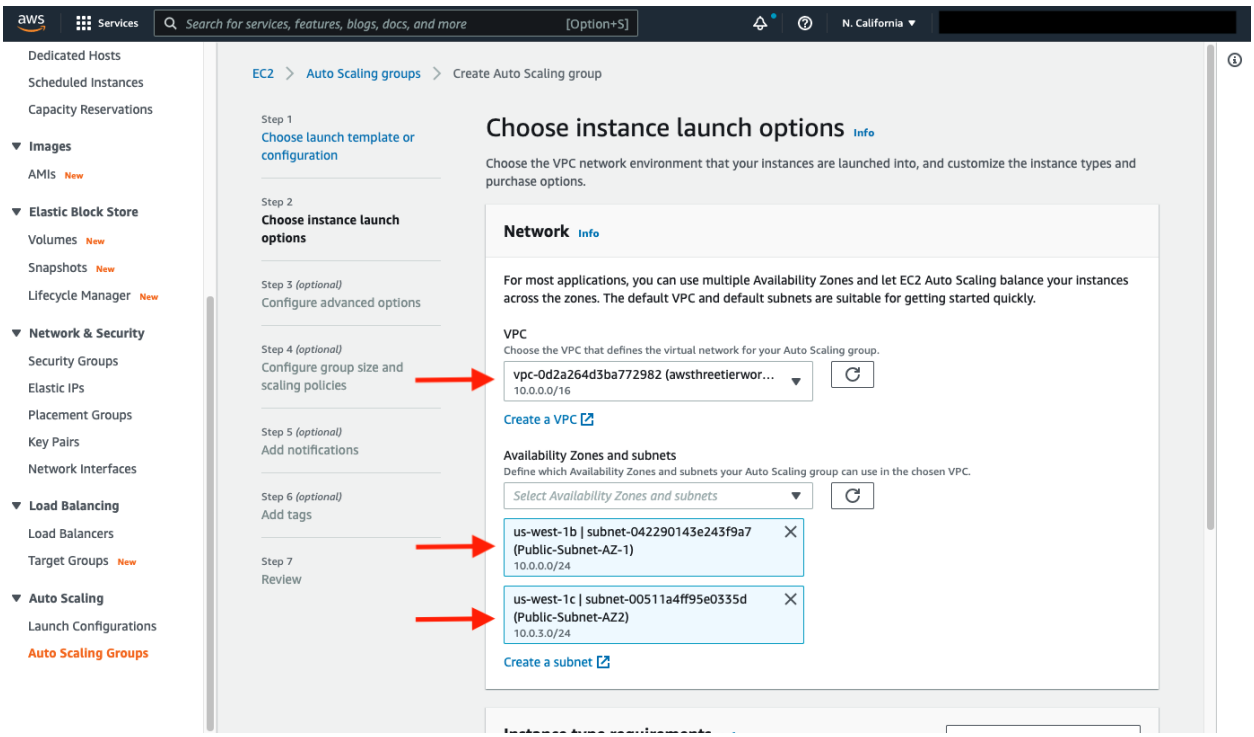
1. We will now create the Auto Scaling Group for our web instances. On the left side of the EC2 dashboard navigate to **Auto Scaling Groups** under **Auto Scaling** and click **Create Auto Scaling group**.



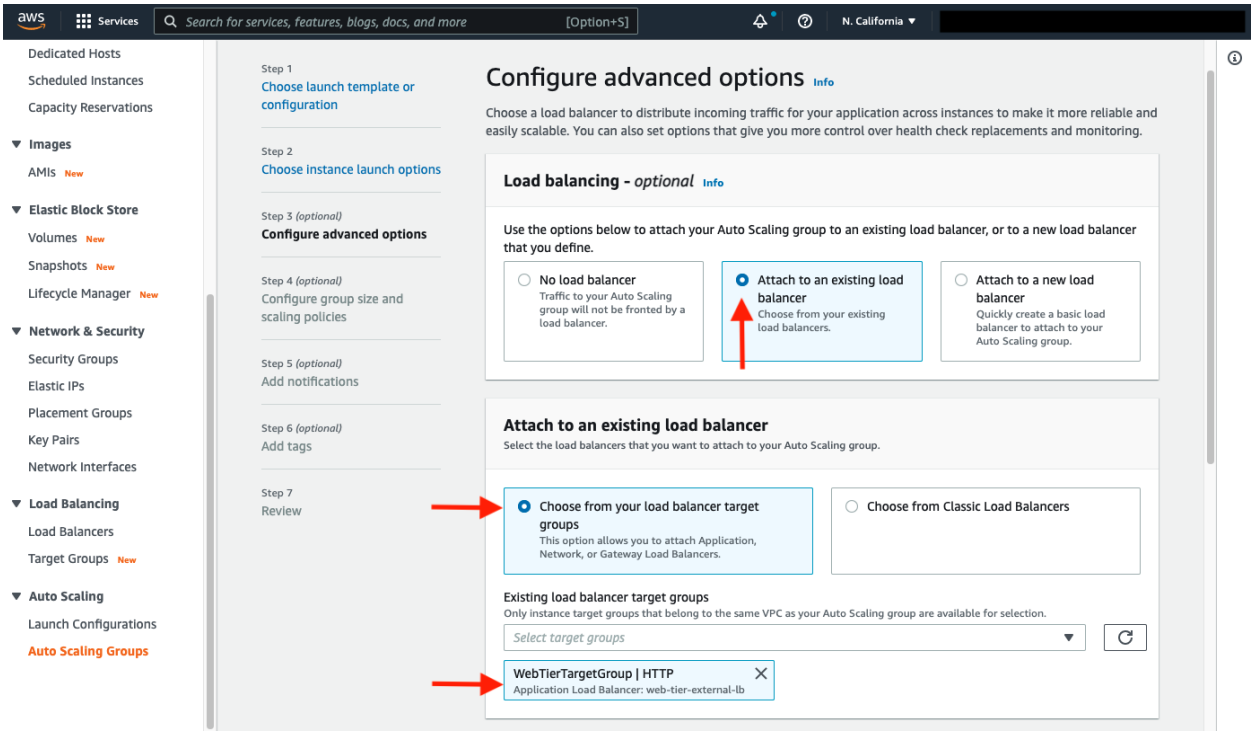
2. Give your Auto Scaling group a name, and then select the Launch Template we just created and click next.



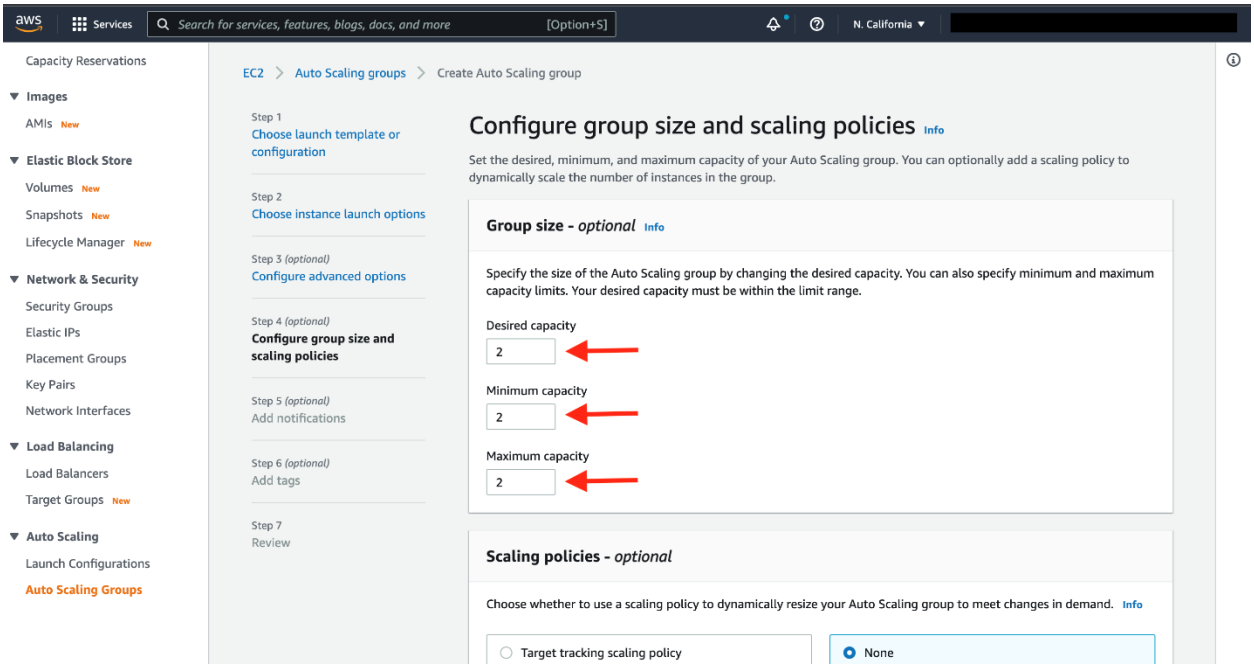
3. On the **Choose instance launch options** page set your VPC, and the public subnets for the web tier and continue to step 3.



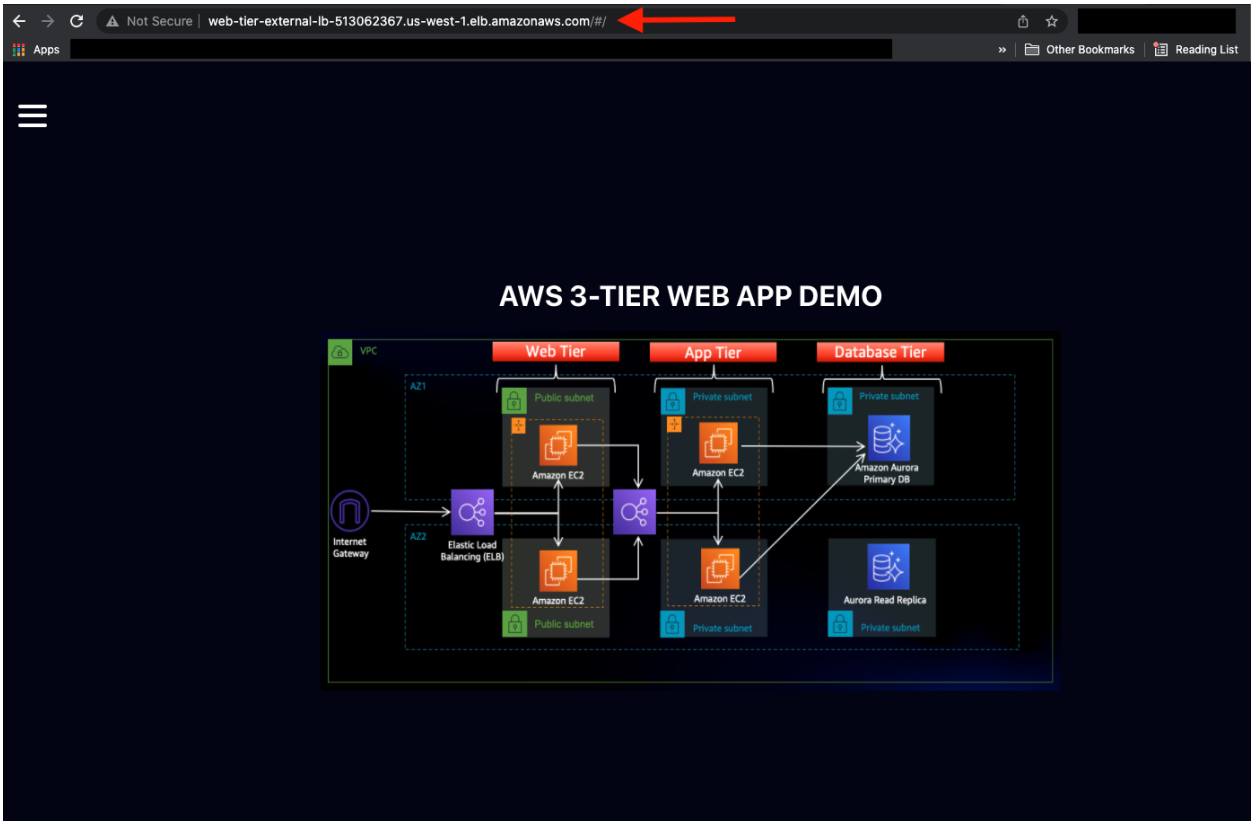
For this next step, attach this Auto Scaling Group to the Load Balancer we just created by selecting the existing web tier load balancer's target group from the dropdown. Then, click next



5. For **Configure group size and scaling policies**, set desired, minimum and maximum capacity to **2**. Click skip to review and then Create Auto Scaling Group.



You should now have your external load balancer and autoscaling group configured correctly. You should see the autoscaling group spinning up 2 new web tier instances. If you wanted to test if this is working correctly, you can delete one of your new instances manually and wait to see if a new instance is booted up to replace it. To test if your entire architecture is working, navigate to your external facing loadbalancer, and plug in the DNS name into your browse



NOTE: Again, your original web tier instance is excluded from the ASG so you will see 3 instances in the EC2 dashboard. You can delete your original instance that you used to generate the web tier AMI but it's recommended to keep it around for troubleshooting purposes.

Congrats! You've Implemented a 3 Tier Web Architecture!