Project Documentation

This document is mainly about the different implementations made in our project. Before securing the cloud infrastructure, we will first be required to create our own website and database. The website created is a simple website named Qaziwa Books with an organised user interface for browsing and purchasing books. The database created will contain all the books available in our store and customer's (registered users) credentials used to register for an account.

Creation of Website:

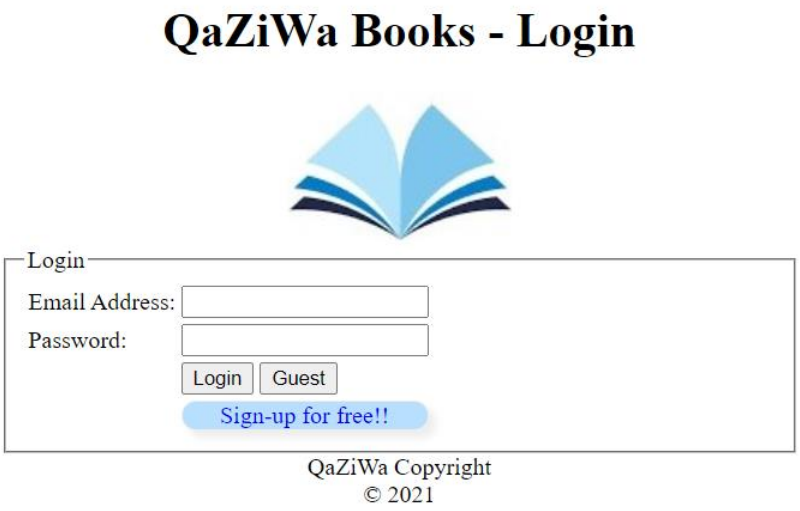
We have designed a simple website (After the security implementation) to ensure that the functionality works.

Our website consists of:

- login
- register
- logoff
- listing (guest, registered user, administrator)
- shopping cart
- payment
- successful payment
- advertisement (only at cart page)
- verification

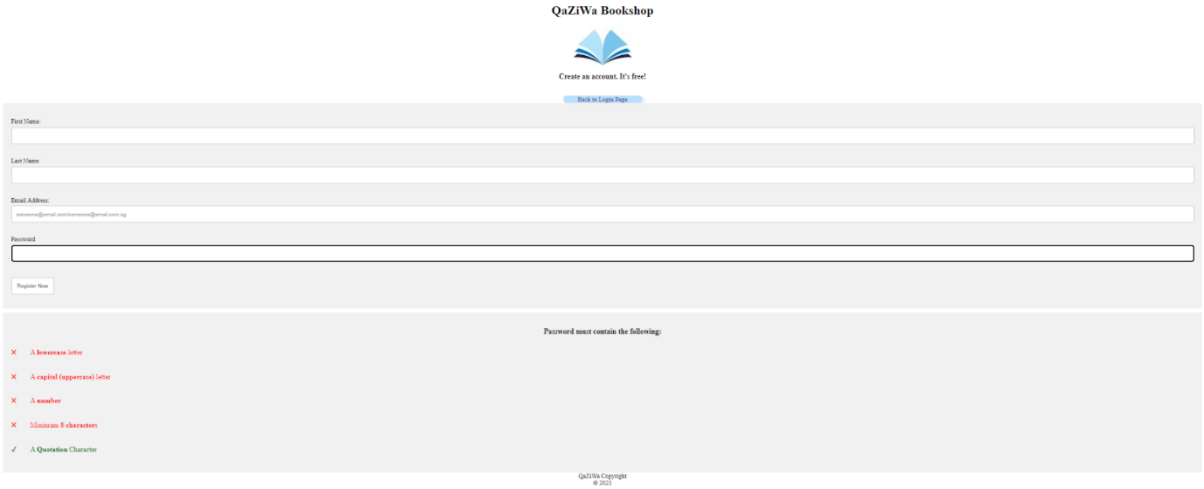
Login page:

We started by designing the base of the website which is the login page and registration. The login page code was written where it asks for the user's input which is the email address and password. After the required field is entered and the user clicks on the submit button which is the login button will then take the 2 fields and send them to doLogin.php. It then checks if the field written is similar to any users in the database. This is done by writing a SELECT statement that retrieves the user's details that is the same as the file written. If the field is correct, the page is then redirected to the listing page (with href). Since we wanted to allow the admin rights to edit the listing, we wrote a code that if the user logs in with the 'admin' account it would redirect to a separate listing page that has the ability to delete, add and edit books into the library and from there with a SQL statement, it would update the database accordingly. For users who would log in as a guest, we coded a submit type that redirects the user to the listing page for guests, allowing them to view the books.



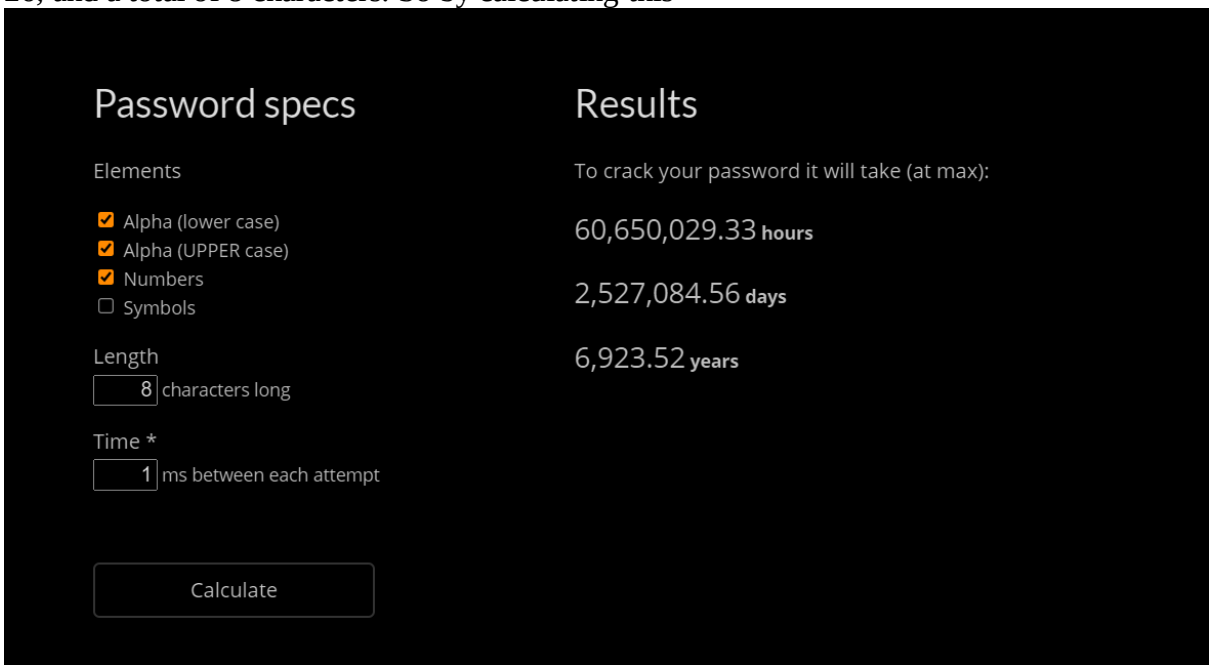
Register page:

For new users who wish to register an account, the registration button would redirect them to the registration page (register.php) from the elseif line of code stated earlier. For the registration page, we would implement the code to create fields for them to create an account. The respective inputs would then be transferred to doRegister.php with form action. In doRegister.php the first name, last name, email address, and password that has been filled by the user would be inserted into the database. We would use $_POST to retrieve the fields and write an SQL statement to insert the fields into the individual values. Each field that is posted would then be inputted into the function stripslashes() and RemoveSpecialChar() to prevent any SQL injection statements. Afterward, the code would run an if-else statement where if the insertion is successful, it would href back to the login page (login.php). However, if the insertion was not successful, it would echo “Your account creation was unsuccessful. Please try again.”.



As shown above, we want to make the password strong, therefore we implement validation to the created password field, this is the step of adding the security in which to prevent an attacker from brute force the password. As shown above, we can see that the user needs to

put a lowercase letter, uppercase letter, a number, and a minimum of 8 digit characters. Therefore, in calculation numbers have 10, alphabet (uppercase) have 26 and lower case has 26, and a total of 8 characters. So by calculating this



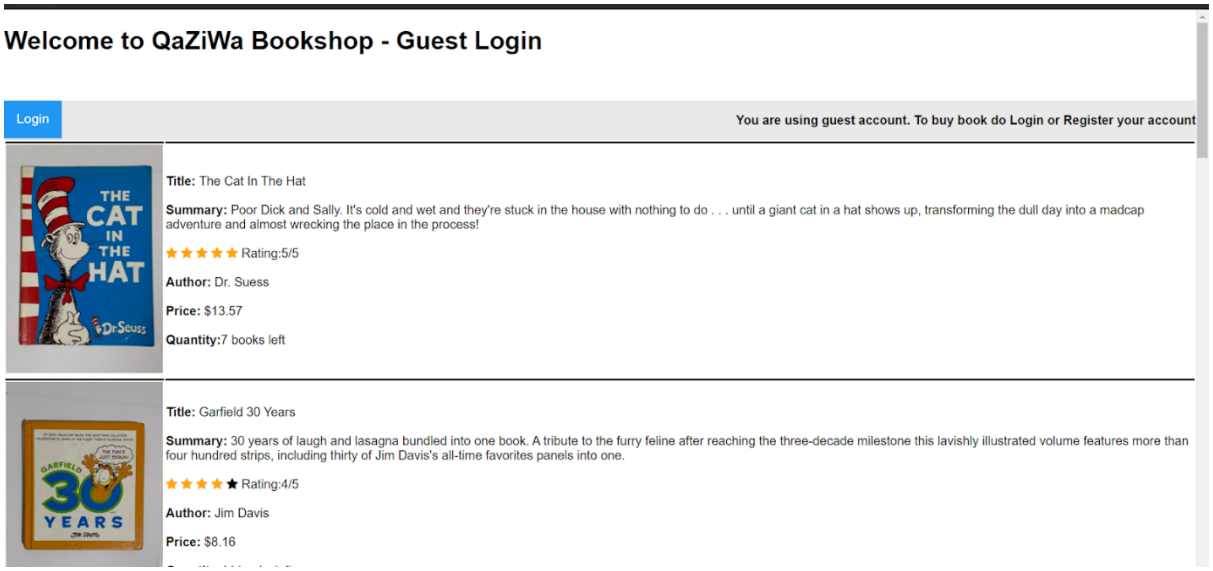
It would roughly be 6,923.52 years and assume 1ms for each attempt. Hence this is the enhanced security on the user side to ensure that their password is not being brute-forced.

Logoff page:

For the logoff page, we start by destroying the user's session by implementing session_destroy(). A string "You have logged out." is then given to a variable message to allow the code to echo it on the logout page. We then created the page to look like a logout page with a simple title called "QaZiWa Bookstore Logout".

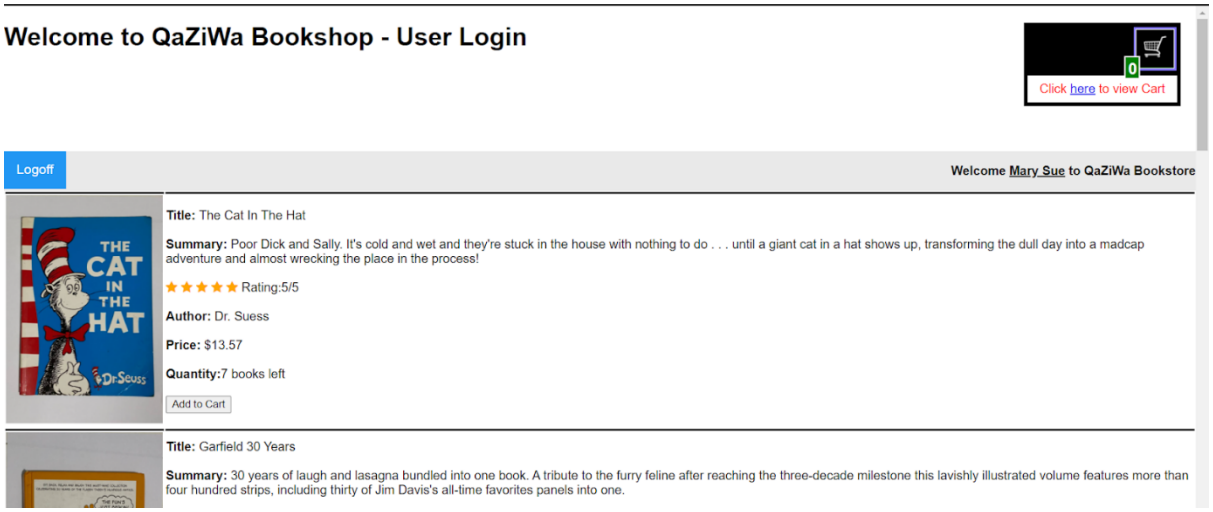
Listing Page (Guest):

The listing page for users that logs in as a guest has the least privileges. They can only view the books inside the store after creating a guest account.



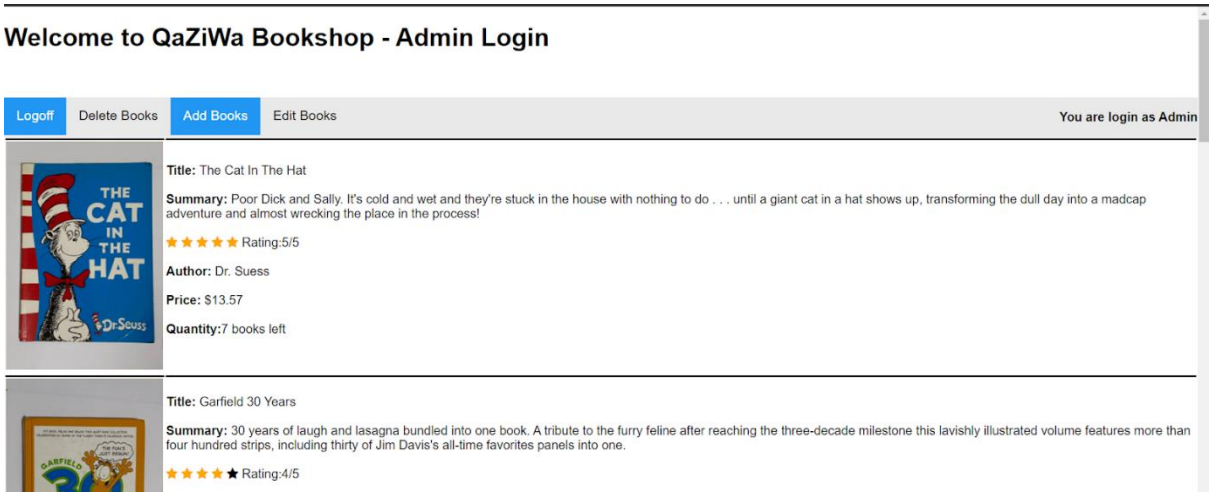
Listing Page (Registered User):

These users have more privileges as compared to the guest user. Apart from just viewing the books available inside the store, they can make purchases by adding and removing items from a shopping cart called $arrCart[]. It will be reflected inside the cart_books table inside the database. After confirming their order, they will be directed to the payment page for payment. A successful payment will bring them to the successful payment page and show the estimated time of their delivery. Users can choose to return to the listing page.

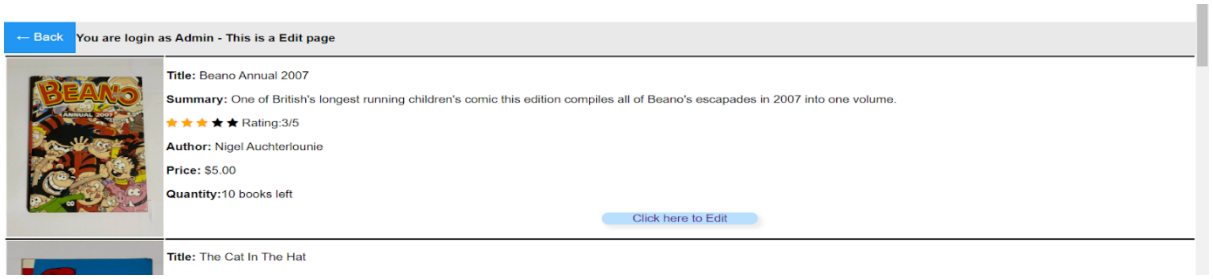


Listing Page (Administrator):

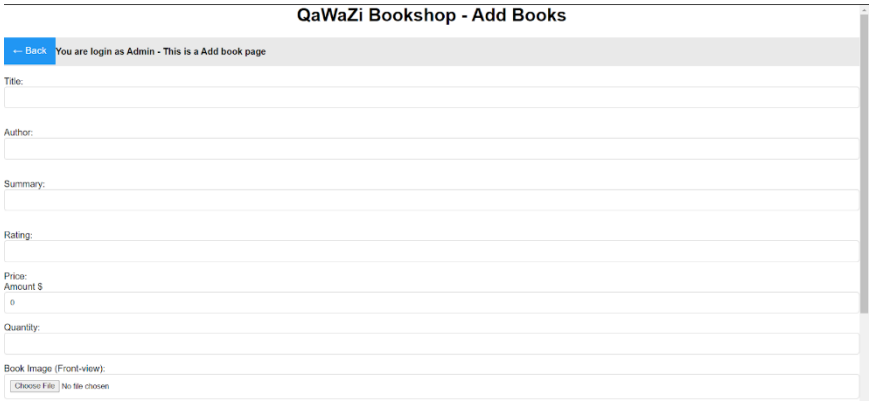
They have the most privilege among the 3 different types of users. Some basic privileges are adding, deleting, and editing books. All data will be stored inside the database QaZiWa. To add books, an INSERT statement will add them to the database where title_id gives a unique identity to each book (primary key). When editing listed books, the UPDATE statement is used to amend information about books available inside the store. When deleting listed books, a SQL query DELETE FROM statement to delete books from the database.



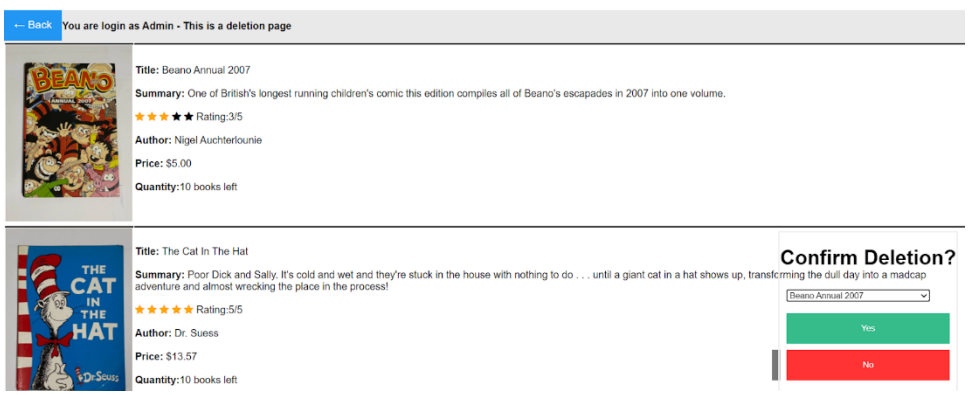
→ Edit books:



→ Add books:



→ Delete books:

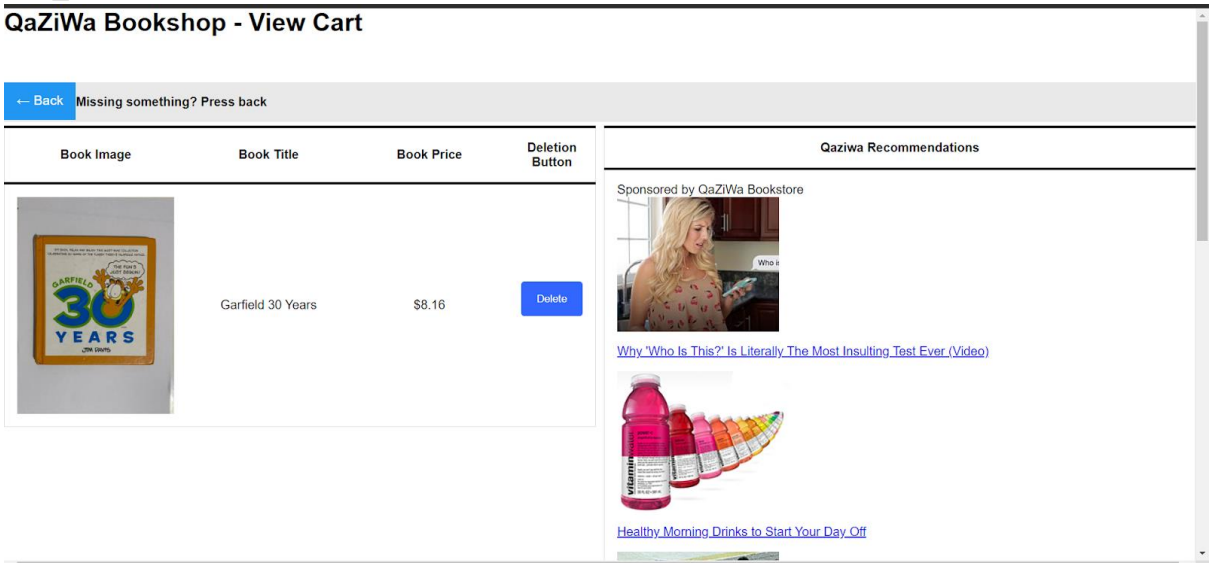


Shopping Cart Page:

Items that will be added or deleted from the cart are called $arrCart[] which is the parameter that would get the information from the database accordingly. That will then be assigned to individual parameters which will be used for the CSS styling to tidy up the display of the respective information. This is what makes the quantity, prices, title look organized. For situations where the users suddenly decide to delete the books from their cart, the code which forms the docartdelete.php would be posted for the commands there to run.

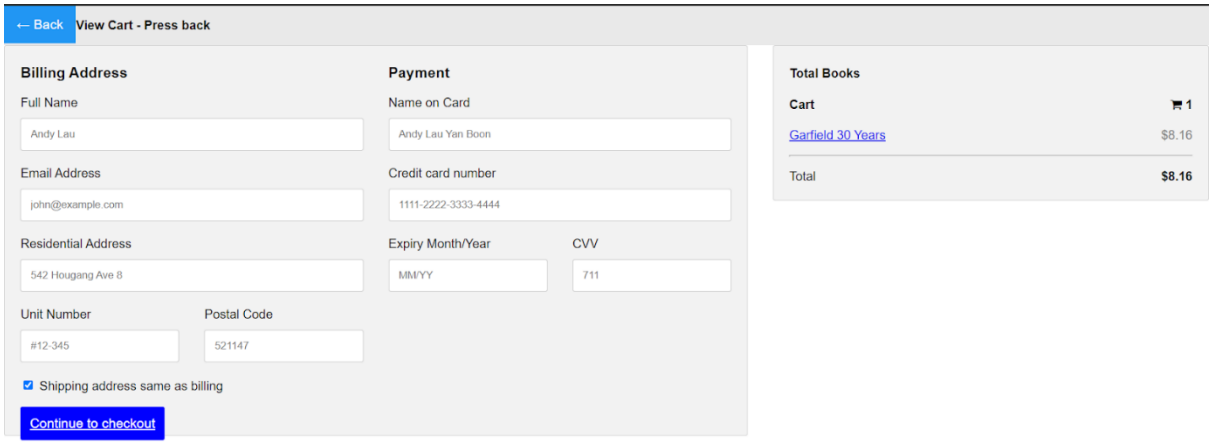
For the docartdelete.php the DELETE SQL statement is written there to delete any selected book from the cart_books table. An if-else statement is written to print an error message in the event the delete is unsuccessful.

Going back to the cart.php the cart would access the user's cart information from the database however the information has to be inserted first before any retrieval is possible therefore this is where doAddToCard comes in. The code in this PHP takes the title of the ID of books that the current user is selecting and takes in all the information of that book and inserts it in the cart_books table with the INSERT INTO statement.



Payment Page

The payment page would be page redirected after the user confirms the books that they would like to purchase at the cart. With that in mind, the code to return to the cart is implemented at the start to allow the user to return to the cart if they wished to do so. We would then add the fields for the user to enter their billing address. Each field has a placeholder to give an example of what each field should look like.



Successful Payment

For the successful payment page, it would appear after the payment page has been successfully redirected after confirming the page. A SQL query is implemented here to delete the current user's cart information from the cart_books table. After the query has been successfully executed, a message will be echoed confirming successful payment with an ETA of the order arrival. A href is also implemented at the end of the message to allow the user to be redirected back to the listing page.

Payment Successful

Thank you for shopping with us!

Your shipment will arrive around 3 weeks.

Back to Listing page

QaZiWa Copyright
© 2021

Advertisement (Cart Page)

Taken from an open-source page we took the code and its assets and inserted them into our own .js file. Afterward, we added the code to display it on the right side of the cart page. At the end of the few lines would be the code that randomizes all these advertisements and with the function, getRandomInt would return a specific number of advertisements. It would then be targeted to the cart page.

```
(function (window, undefined) {
    /* Options */

    var creatives = [
        {"title": "Is Donald Trump The Best Candidate for 2016?  Vote Here.", "creative": "/creatives/trump.jpg", "href": "http://newsmax.com"},
        {"title": "These Photos From The Past Are Bitter Sweet", "creative": "/creatives/rifle.jpg", "href": "http://buzzlamp.com"},
        {"title": "Most Satisfied People Don't Wait For What They Want, They Go Get It", "creative": "/creatives/satisfied.jpg", "href": "http://elitedaily.com/life/
        {"title": "10 Tips To Learn Any Language From The Genius Who Speaks 9", "creative": "/creatives/genius.jpg", "href": "http://elitedaily.com/life/satisfied-pe
        {"title": "A Lioness Captures A Baby Baboon And Does The LAST Thing You'd Expect", "creative": "/creatives/lioness.jpg", "href": "http://www.littlebudha.com/
        {"title": "Why 'Who Is This?' Is Literally The Most Insulting Test Ever (Video)", "creative": "/creatives/whois.jpg", "href": "http://elitedaily.com/humor/wh
        {"title": "The Six Worst Types of Coworkers: And How To Deal With Them", "creative": "/creatives/coworker.jpg", "href": "http://elitedaily.com"},
        {"title": "4 in 5 Americans Are Ignoring Buffet's Warning", "creative": "/creatives/buffet.jpg", "href": "http://www.fool.com/video-alert/stock-advisor/sa-ni
        {"title": "Warren Buffet Just Gave Americans A Big Warning", "creative": "/creatives/buffet.jpg", "href": "http://www.fool.com/video-alert/stock-advisor/sa-n
        {"title": "Americans Urged To Search Their Name On New Site", "creative": "/creatives/american.jpg", "href": "http://www.instantcheckmate.com/"},
        {"title": "The Most Addicted Shopping Site For Women", "creative": "/creatives/addicted.jpg", "href": "http://tophatter.com/blog"},
        {"title": "Power Companies Fear This Chicago Family", "creative": "/creatives/solar.jpg", "href": "http://www.thedailylife.com/new-policy-makes-power-compani
        {"title": "Six Reasons Your Wifi Is Your Most Important Relationship", "creative": "/creatives/wif.jpg", "href": "http://elitedaily.com/life/culture/6-reason
        {"title": "7 Overhyped Games That Ended Up Being Terrible", "creative": "/creatives/terrible.jpg", "href": "http://www.looper.com/2323/hyped-games-ended-terr
        {"title": "Games That Punish You Seriously For Dying", "creative": "/creatives/dying.jpg", "href": "http://www.looper.com/2117/games-seriously-punish-dying"}
        {"title": "Healthy Morning Drinks to Start Your Day Off", "creative": "/creatives/morningdrinks.png", "href": "http://epphany.com/site/post/577/"},
        {"title": "Best Ways to Scare Your Girlfriend", "creative": "/creatives/scaregirlfriend.png", "href": "http://epphany.com/site/post/116/"},
        {"title": "Movies That Ruined the Book", "creative": "/creatives/booksmovies.png", "href": "http://epphany.com/site/post/3"},
        {"title": "The 10 Most Disturbing Pokemon Facts", "creative": "/creatives/pokemon-facts.png", "href": "http://gamerant.com/pokemon-disturbing-facts"},
        {"title": "How To Impress Employers at Info Sessions", "creative": "/creatives/infosessions.png", "href": "http://ptsteadman.github.io/How-To-Impress/"},
        {"title": "So You Want to Date an Artist", "creative": "/creatives/saveartist.png", "href": "http://blog.newhive.com/so-u-want-to-date-an-artist/"},
        {"title": "So You Want to Date an Artist", "creative": "/creatives/saveartist.png", "href": "http://blog.newhive.com/so-u-want-to-date-an-artist/"},
        {"title": "Weight-lifting Kangaroo has Neighborhod on High Alert", "creative": "/creatives/buffkang.png", "href": "http://www.theweathernetwork.com/us/news/a
        {"title": "Are Lentils a Low-Calorie Food?", "creative": "/creatives/lents.png", "href": "http://healthyeating.sfgate.com/lentils-lowcalorie-food-1034.html"}
        {"title": "10 Small Dogs and What They're Like", "creative": "/creatives/pug.jpg", "href": "http://www.sheknows.com/pets-and-animals/articles/808128/top-10-s
    ];

    /* most recently parsed script */
    var scripts = document.getElementsByTagName("script");
    var index = scripts.length - 1;
    var kaboodleTag = scripts[index];
    var src = kaboodleTag.src;
    var kaboodleRoot = src.substring(0, src.lastIndexOf("/"));
    var kaboodleStyle = document.createElement("link");
    kaboodleStyle.href = kaboodleRoot + "/kaboodle.css";
    kaboodleStyle.rel = "stylesheet";
    kaboodleStyle.type = "text/css";
    document.head.appendChild(kaboodleStyle);
```

Sponsored by QaZiWa Bookstore

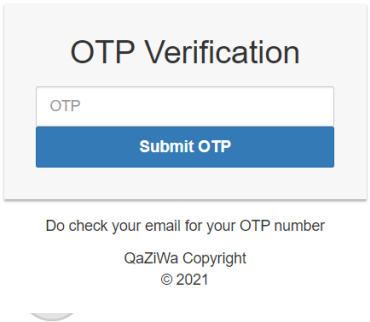
Is Donald Trump The Best Candidate for 2016? Vote Here.

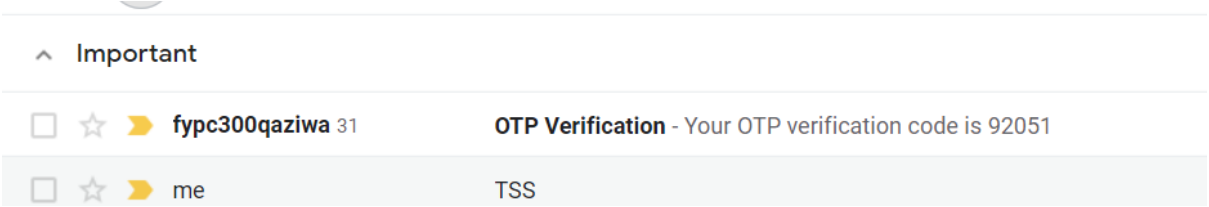
The Most Addicted Shopping Site For Women

So You Want to Date an Artist

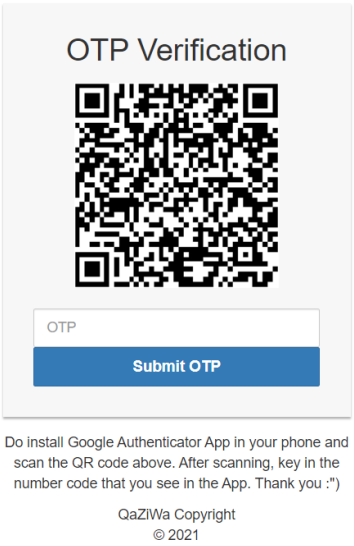
Verification Page

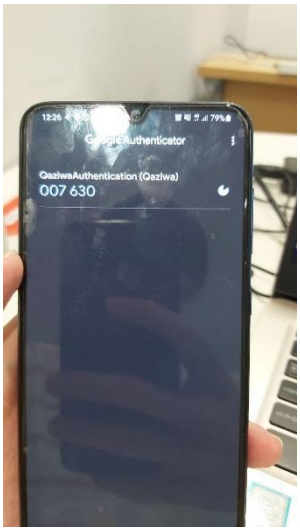
On this page, there will be an OTP that will send over to the user via email, from there the user will then log in to their own email and get the OTP number and key it into the verification page where it will then check if the number tally with the database. If it tallies, then the user is able to use the account and if not, the user will be unable to log into their account.





This is only the first security testing, but going further than this, we will be using google authenticator to authenticate our website. After intensive research and testing, we have come up with the google authenticator and integrate it into our website. This can be seen below:





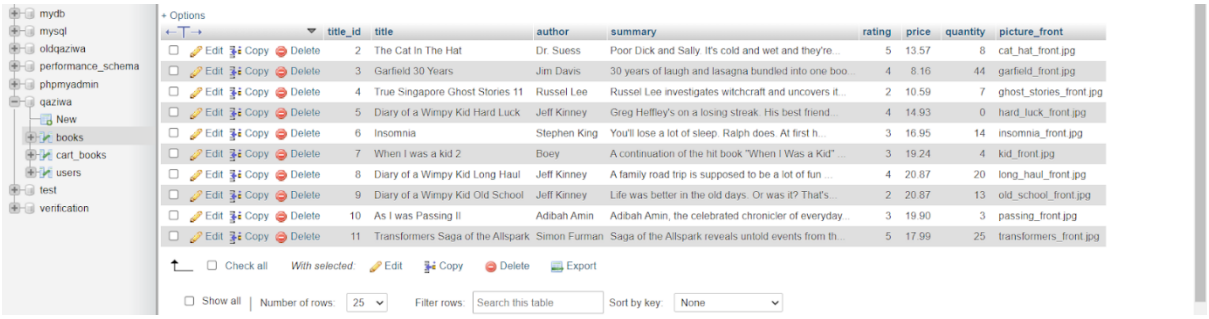
After scanning the QR code, using the google authenticator app, an OTP number is shown in which then we will key in the number in the web OTP section. This is an added security in which using a reliable source and API from Google to authenticate the OTP, as compared to the first authentication, where the attacker can brute force the OTP due to the first one being using the static OTP and the OTP having no change or time out. But when using the Google authenticator, it has a timeout session and the OTP is not static and keeps changing every few seconds, therefore, making brute force not possible. In addition, the QR code will keep changing into a new QRcode and it will knot be a static QRcode, this is an added security feature to ensure that different users have different access to the QR code and ensure that the OTP pin is kept consistently changing together with the QR code.

Creation of database tables:

Since we already know what the website consists of, now is the time for the database creation. We have planned to use phpMyAdmin as the service for our database. We created a new database from scratch with the name QaZiWa that uses the latin1_swedish_ci collation. Below show the tables that we used for our database.

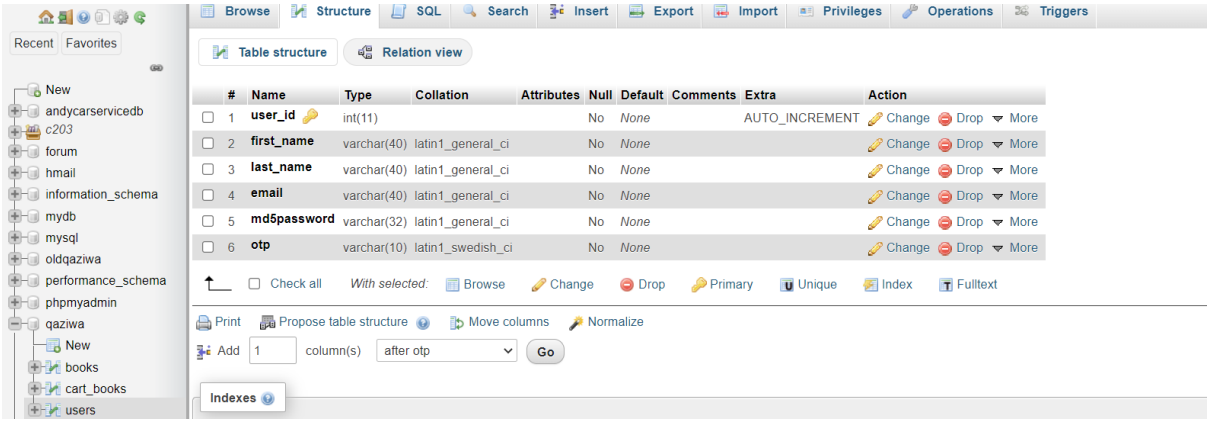
Books (table):

The table “books" would contain the information of each book that is being sold on QaZiWa. We started by creating the primary key title_id with the auto-increment setting activated for it. The type for the primary key is then set to int with a max count of 11 characters. Next would be creating a column for the title and author individually and what the name suggests would store exactly that with the type being varchar with a maximum number of characters of 40. A summary column is also created with a max character of 8000 for the admin to add in a sizable introduction to the book. Next would be the price which uses the decimal type and the range 5,2 to allow exact pricing down to the cents. The quantity is then added with the int type like the primary key. Lastly would be the picture_front column which uses the type varchar which has a max length of 70. This column keeps the filename of the image of the book which allows the website’s code to easily access the necessary picture of the targeted book. With the data, fields set we then began inserting 11 different books ranging from different genres, ages, and types.



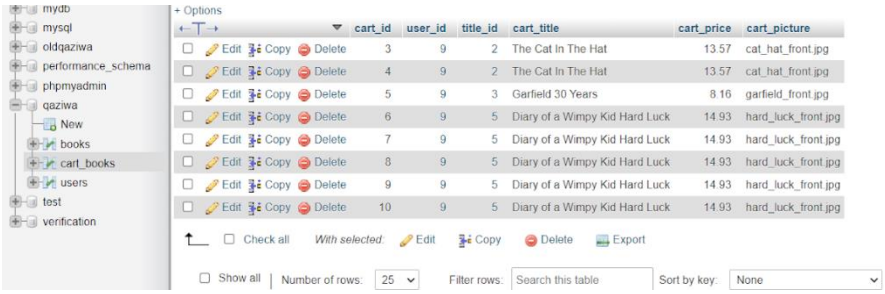
Users (table):

The “users” table stores the user’s email/username and password to allow registered users to login into their account. The primary key for this table would be user_id with the same set of auto-increment and typeset just like the books table. We then added columns for their first and last name to allow a bit of personalization when the user logs in. The email column is then added with the varchar typeset and the character count of 40. Lastly would be inserting the password with the type set to varchar and the count of 32. The max character is set to 32 as we then set the password type to be encrypted to an MD5 hashed password as an added layer of security. With all the columns created we then began inserting a few user accounts to test the functionality of the website and also created an admin account that would have the ability to add new books, delete books and edit them from the website itself. In addition, there is also an OTP section where when the user is logged into the website, an OTP will send via their email and from there it will then tally with the current database OTP, and if it tallies the user is successfully logged into the account if not then he or she cannot log into the account.



Cart_books (table):

Lastly would be the table that saves every individual user's choices of books which are called "cart_books". The primary key of this table would be called cart_id with the auto-increment option activated for it too. Unlike the other 2 tables, "cart_books" would contain a foreign key as there would be a need to access data from the other 2 tables. The foreign key would be the user_id and title_id. From there the other columns created for this table which are the cart_title, cart_price and cart_picture would all be accessed through the foreign key respectively.

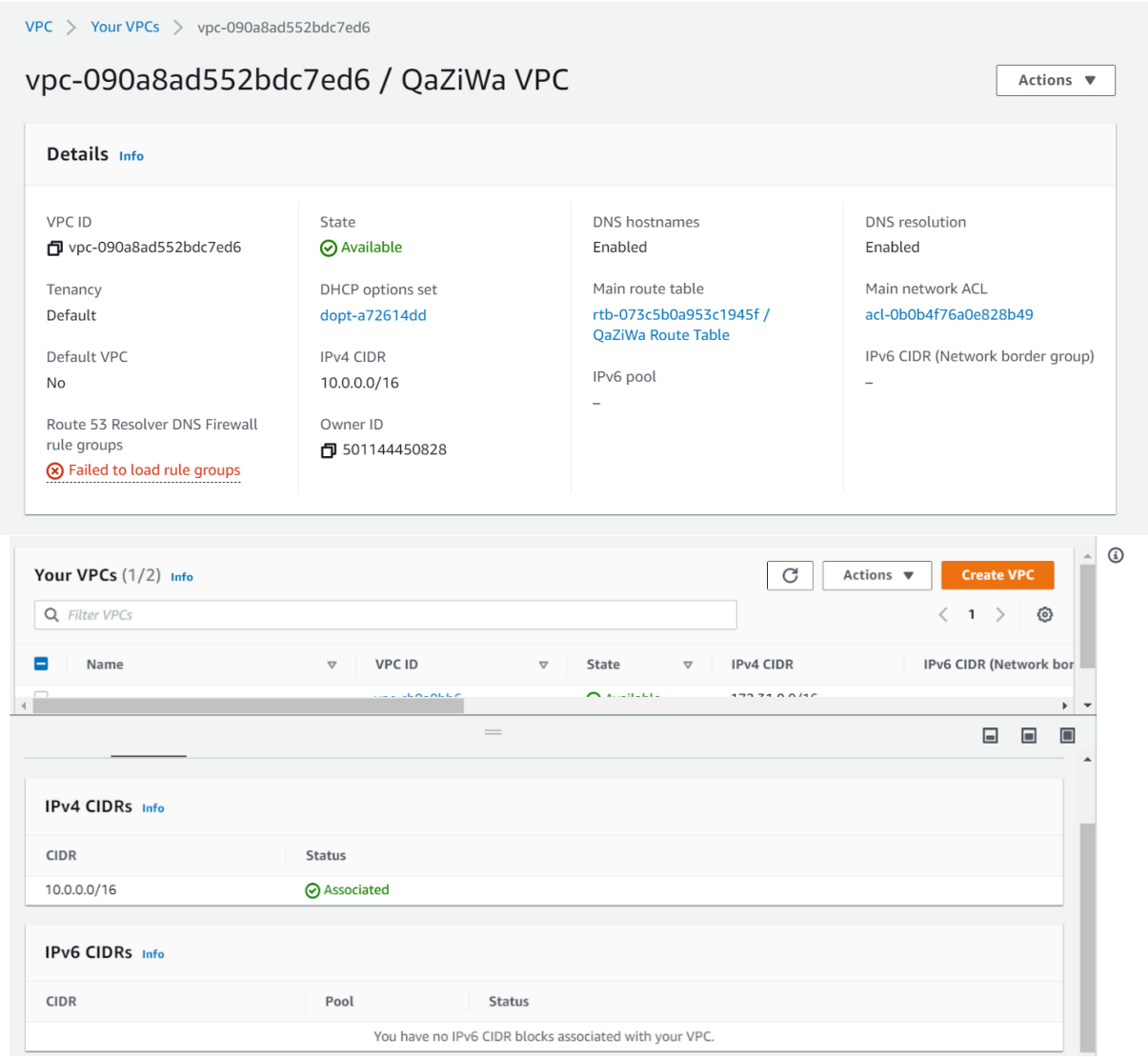


Creation of cloud networks:

Before we proceed to implement the database and web server onto the instance, we need to create our own public and private network, Virtual Private Cloud (VPC), an internet gateway, route table

Creation of VPC:

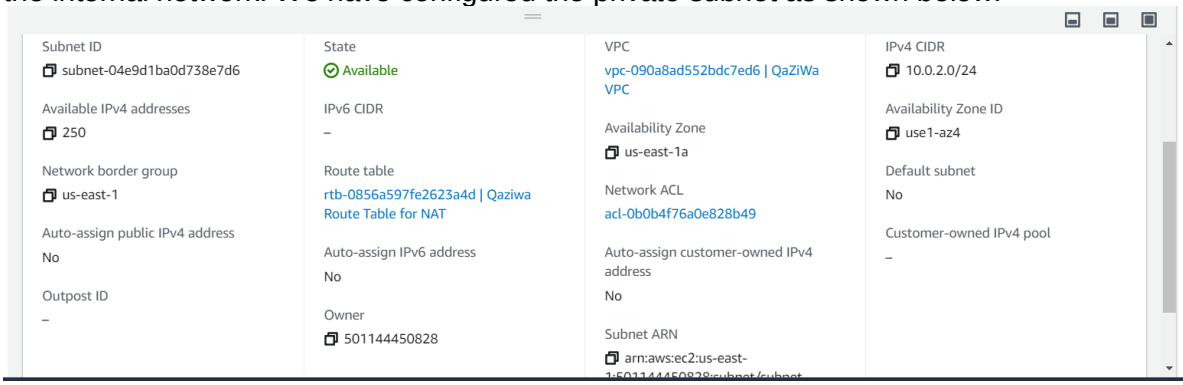
So to start off we first need to create a VPC that helps us to have a public and private facing network in which it is suitable for our web (public-facing) & database (private facing) servers.



For the VPC, we have assigned an IP address range of 10.0.0.0/16 and leave the Tenancy as default. Tenancy defines how EC2 instances are distributed across physical hardware and affect pricing. After we have set up the VPC then we will be continuing to create a subnet

Creation of Subnets:

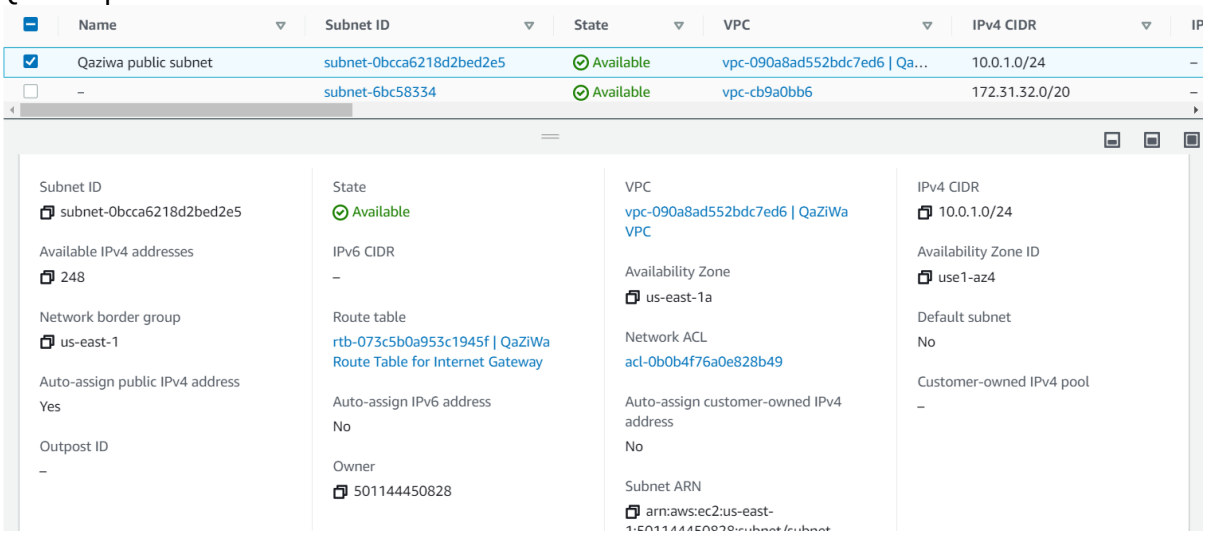
Firstly, we create a private subnet in which it is for our database server to be placed within the internal network. We have configured the private subnet as shown below.



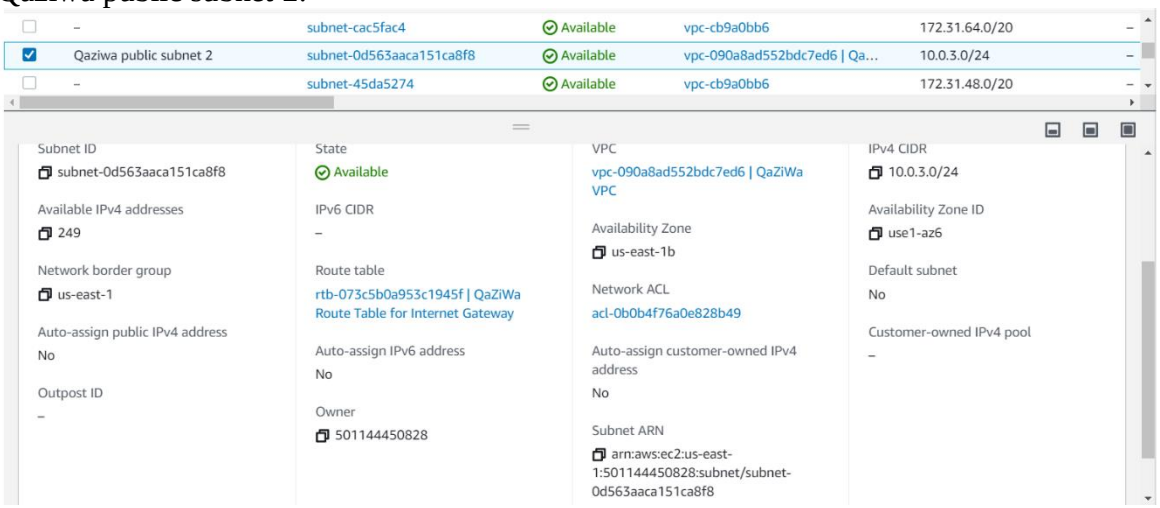
This subnet will not have an internet access/connection, therefore any instances will not be able to connect to the internet unless a NAT service is being implemented in which this configuration will be shown soon.

After creating the private subnet, now is the time to create a public subnet. We will be needing 2 public subnets which are named Qaziwa public subnet 1 & Qaziwa public subnet 2. We need 2 public subnets because later on, we need to configure the load balancer in which it needs 2 zones to ensure that we can configure the load balancer easily and provide scalability.

Qaziwa public subnet 1:



Qaziwa public subnet 2:

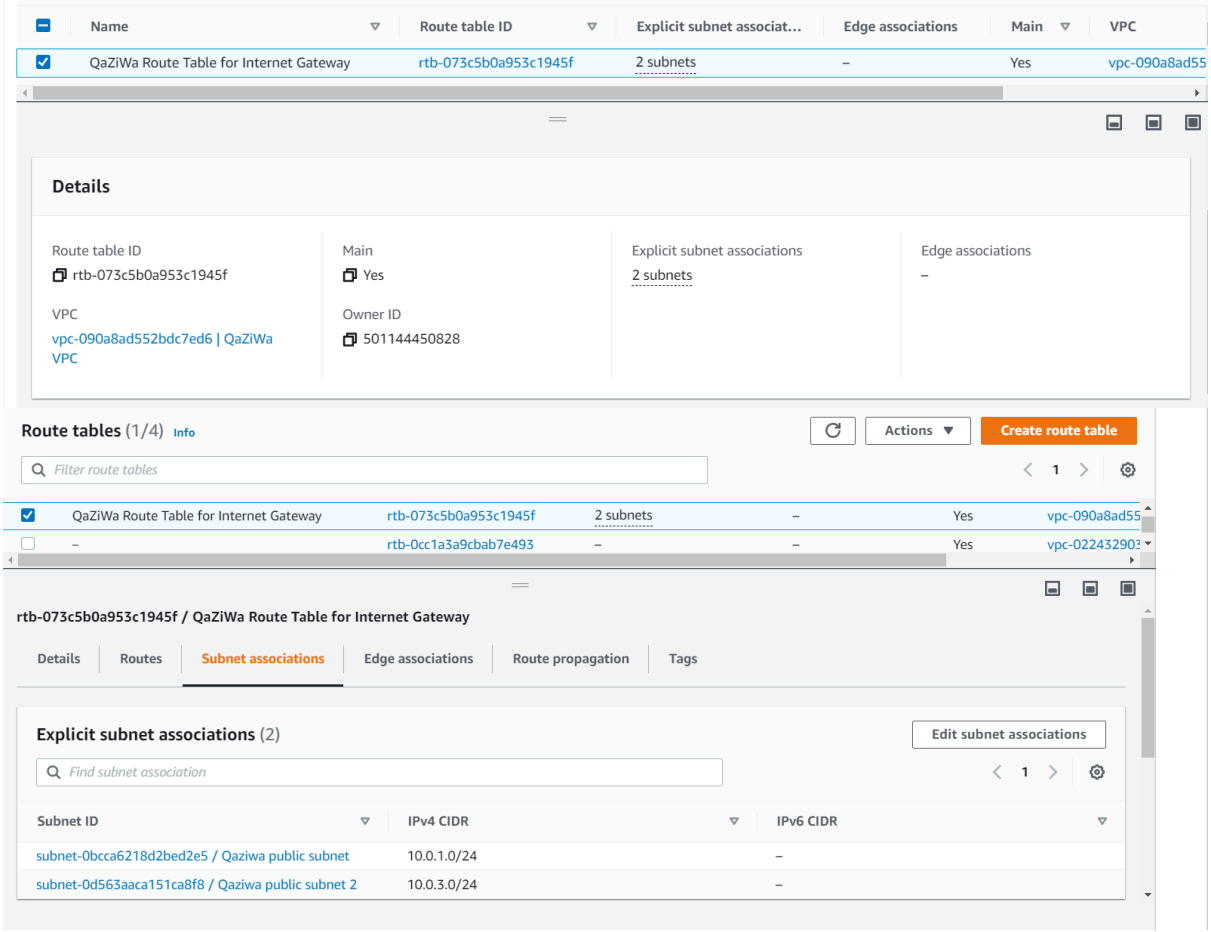


With the public subnet, we are able to connect to the internet and the internet can also connect to us. As such this will be good for our web server. Now we will continue to configure the routing table for our public subnet.

Creation of Route table:

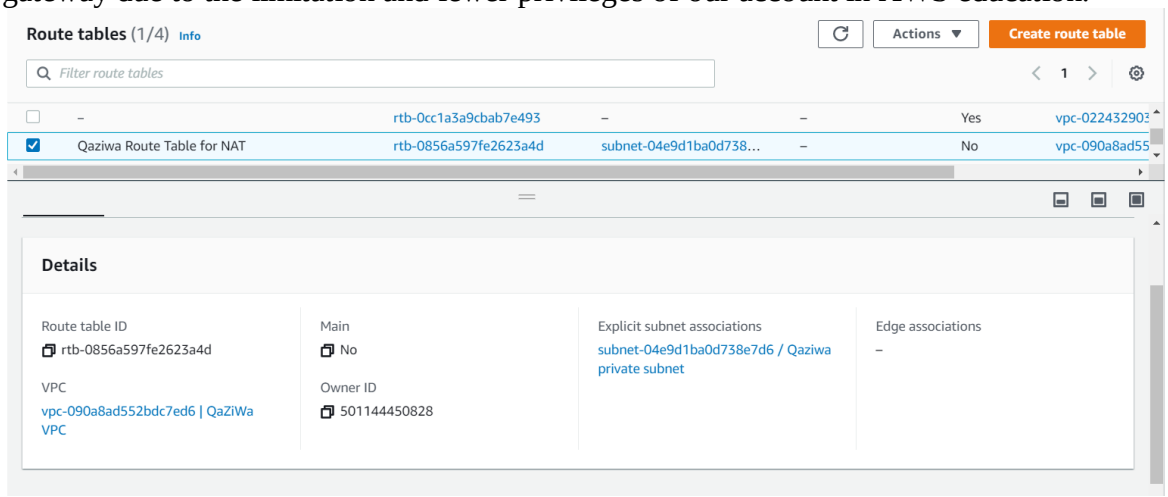
This is a routing table that connects to the public subnet and which also enables both of them to be able to ping each other and connects to one another, in addition, this route table is connected to the internet gateway which in other terms connects to the internet.

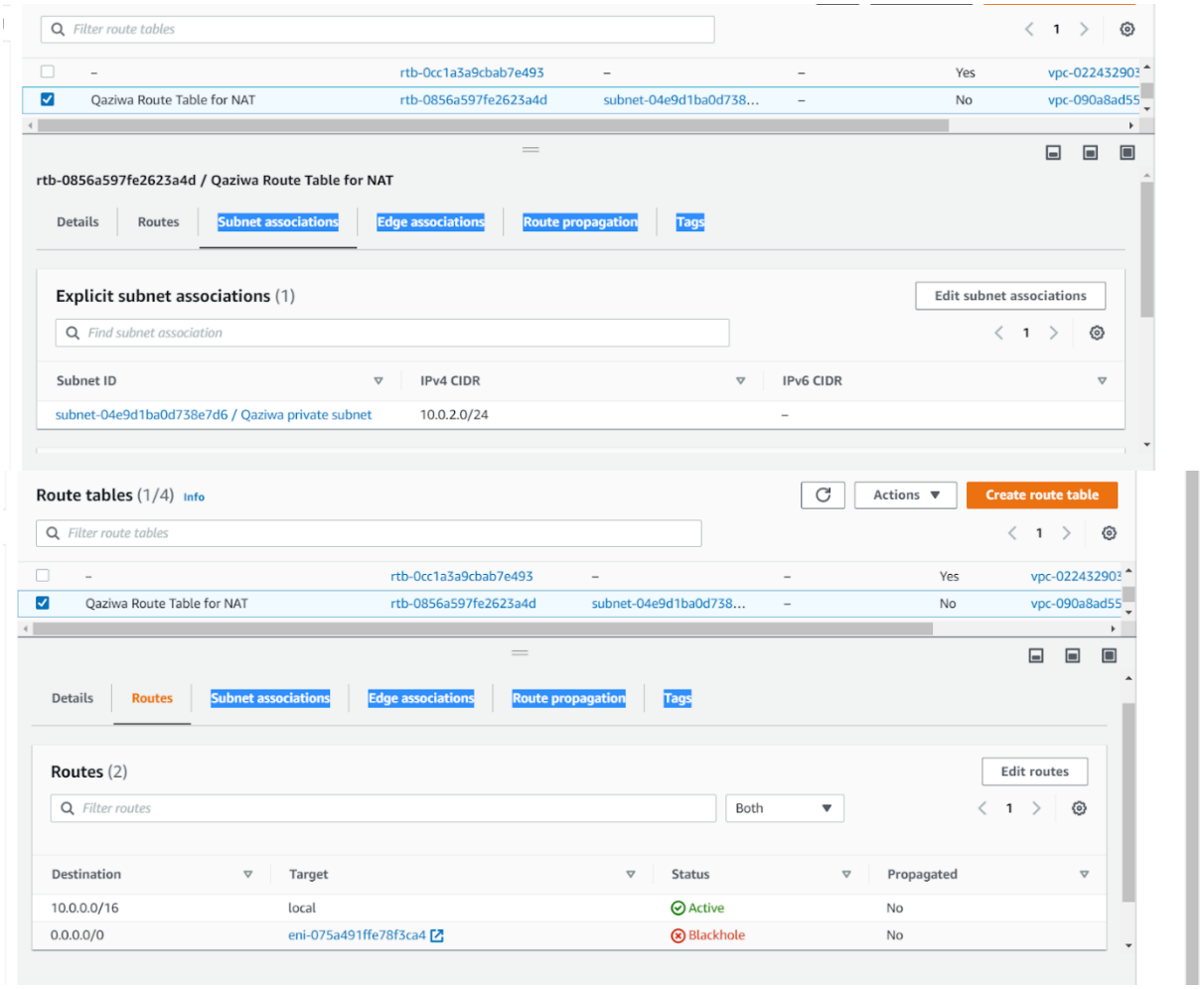
Route table to internet gateway:



Route table to NAT instance:

Due to the private subnet being unable to connect to the internet on its own, therefore, we have come to the conclusion to create a NAT instance that can help to route the traffic out to the internet. So in the configuration above we can see that the routes are looking to the NAT instance which we have already created and this NAT instance will be helping to let the traffic out to the internet but not the other way. We choose NAT instance instead of NAT gateway due to the limitation and fewer privileges of our account in AWS education.

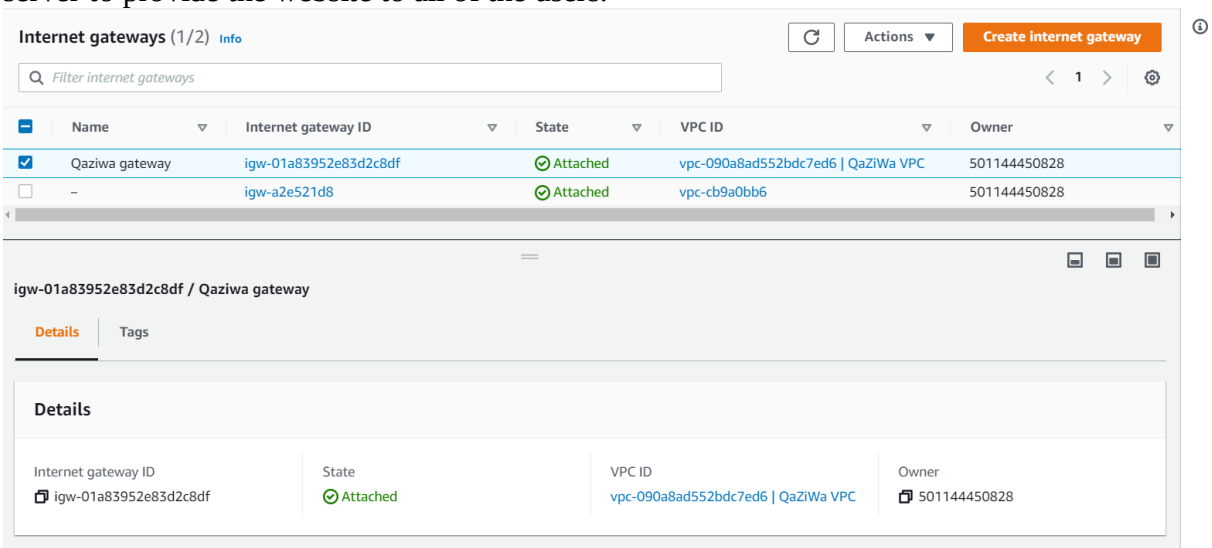




After we have successfully created our route tables, now is the time to create internet gateways.

Creation of Internet Gateways:

Internet gateway has made use to connect to the internet in which it will be useful for our web server to provide the website to all of the users.

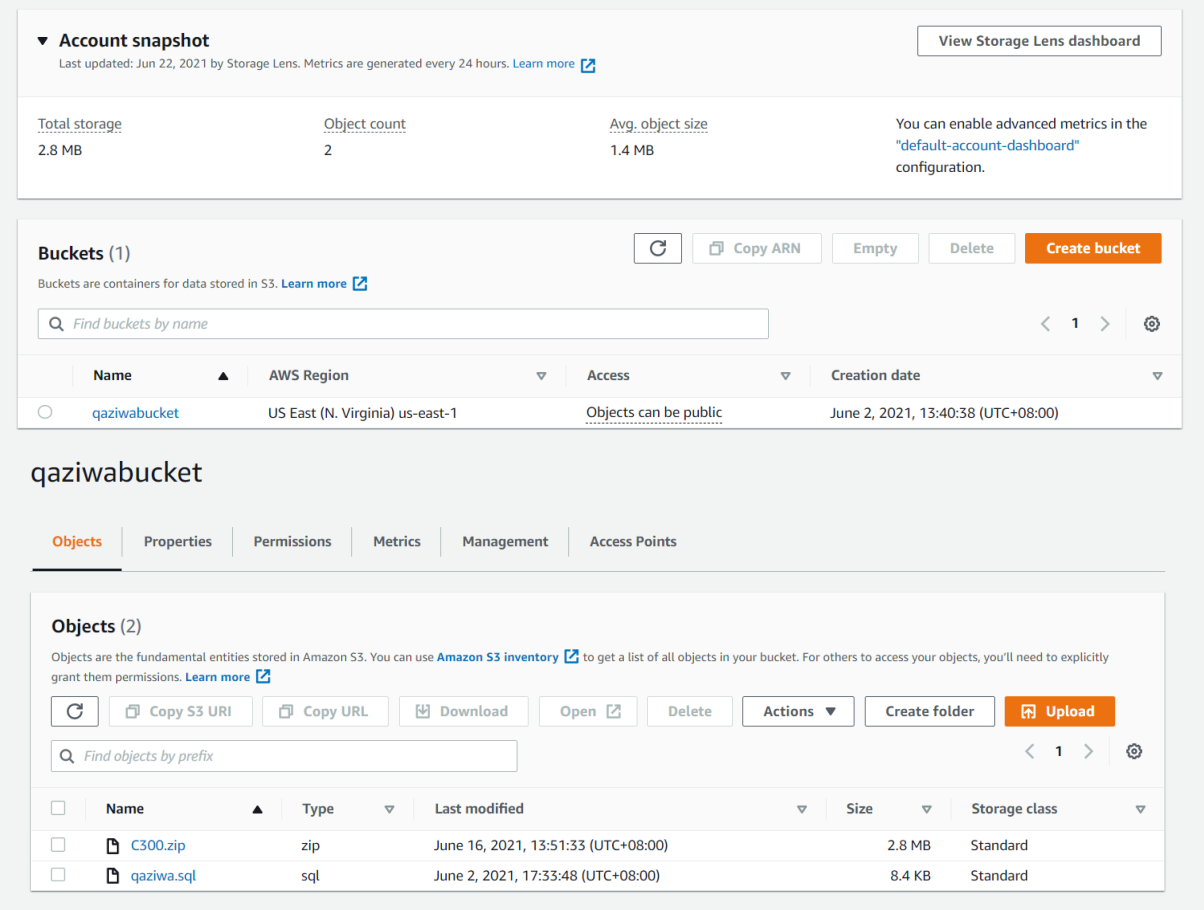


Creation of cloud and online available services:

Since we have already settled the cloud networking in which we have created the VPC, private and public subnet, route table and internet gateway, now is the time to create and implement the various services and configuration that we have planned and what AWS services have offered to us. We will be using and configuring the Instances, Application Load Balancer, Auto scaling, Snapshots, AMI, Volumes, CloudFlare, InnoDB encryption, IAM, KMS, CloudWatch, CloudTrail, Security group and many more.

Creation of S3 Bucket Storage - Website/Database:

In the AWS cloud, we started by creating a new bucket in the S3 table. We called it "qaziwabucket". Setting the initial settings for the S3 bucket we deactivated the checked setting of the bucket blocking all public access to allow us to access the file publicly. After removing the block settings we then uploaded a zip file containing all the codes for our website. After successful uploading, we then made the zip file public to allow our web server instance to access it.

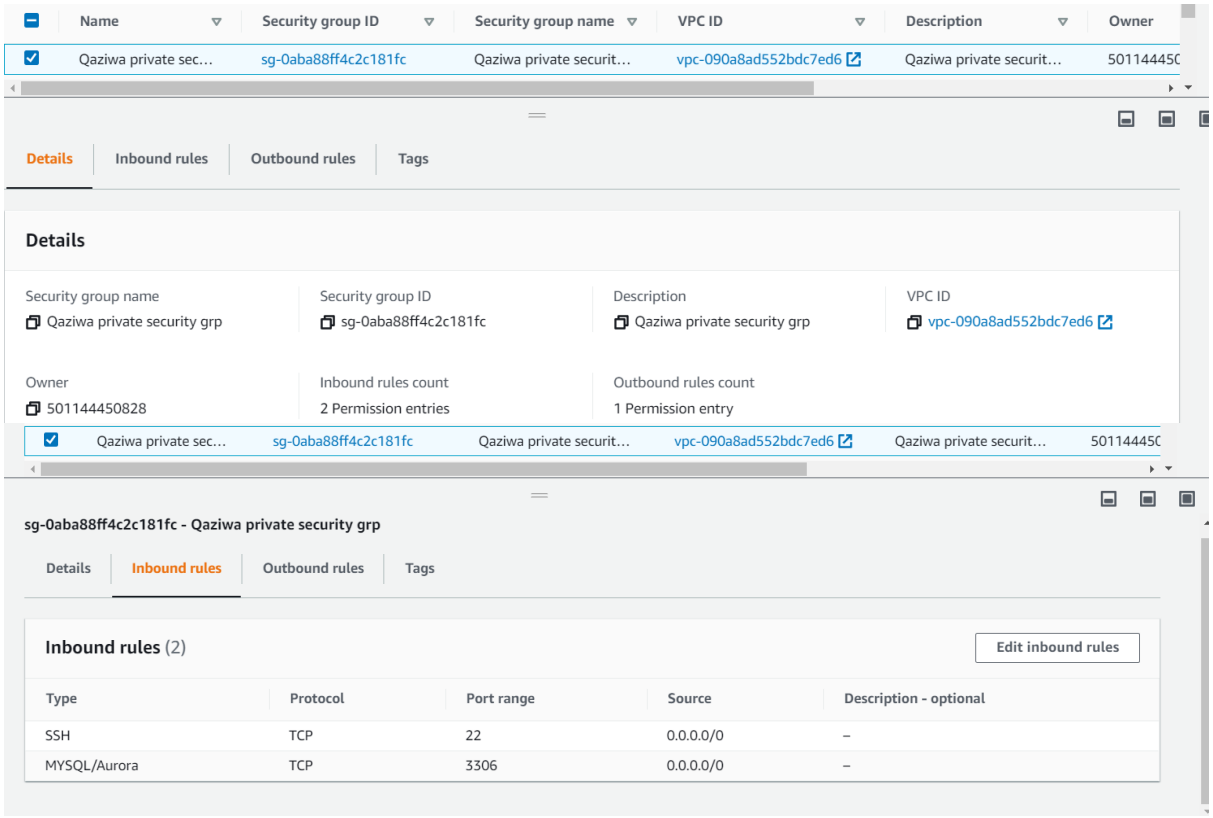


This is where we can use the wget method to fetch this zip file so that we can easily implement the website source code into the AWS instance web server and it also applies for the AWS instance database server. Once we have upload the source code into our S3 bucket, now is the time for us to create the security group.

Creation of Security group:

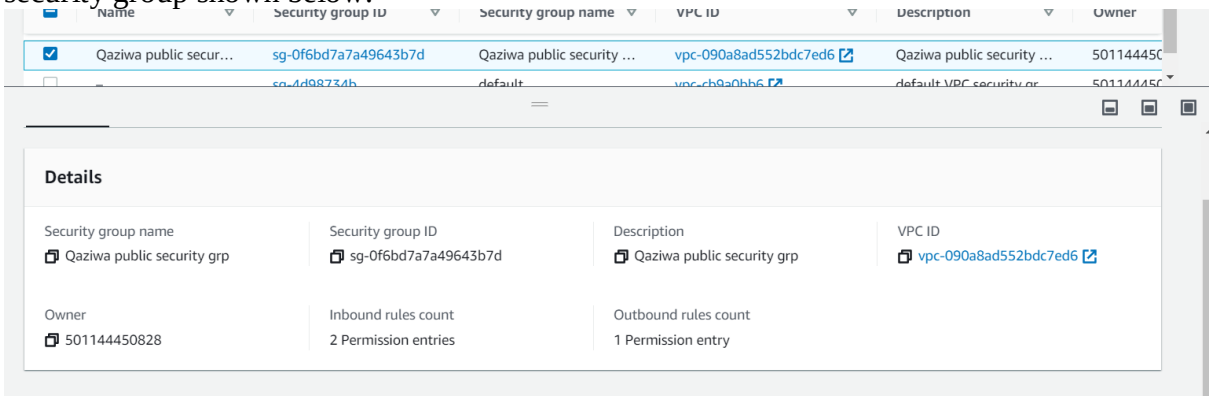
The security group is important as it will help us to manage the traffic and which traffic is allowed into the instance and out from the instance. For example, if we want only this to host at port 80, and enable ICMP, this security group will help us to do this method. This acts as traffic and control which can and cannot flow into the instance. Firstly we have configured

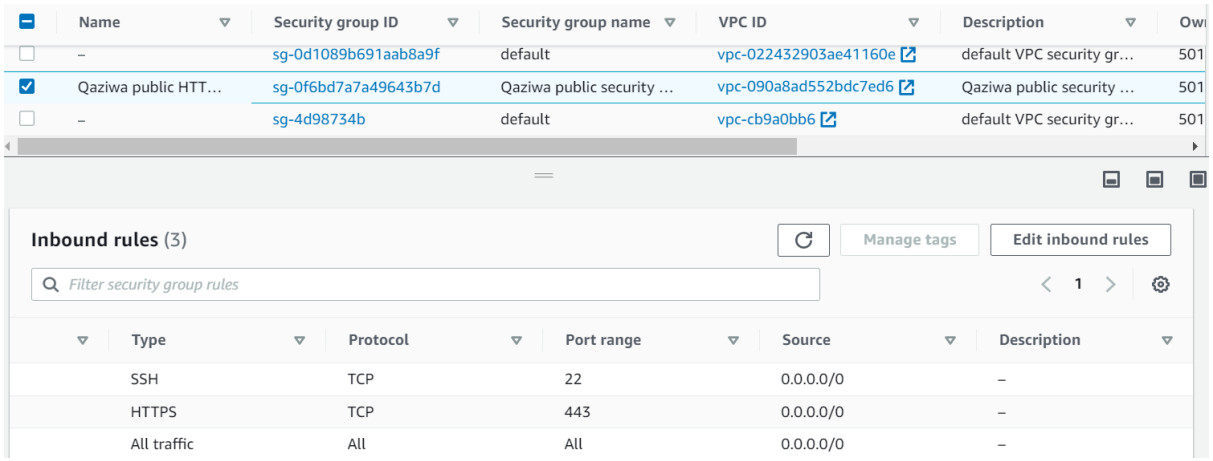
the “Qaziwa private security group”, this is where this security group will allow SSH and database default port which is 3306. This is to ensure we are able to connect to the instance via SSH and able to fetch the database/tables from the web server to the database server.



We allow from any source for SSH as we want to connect from our laptop and also for MYSQL because we have not created the instance yet, so the specific IP address has not been assigned yet.

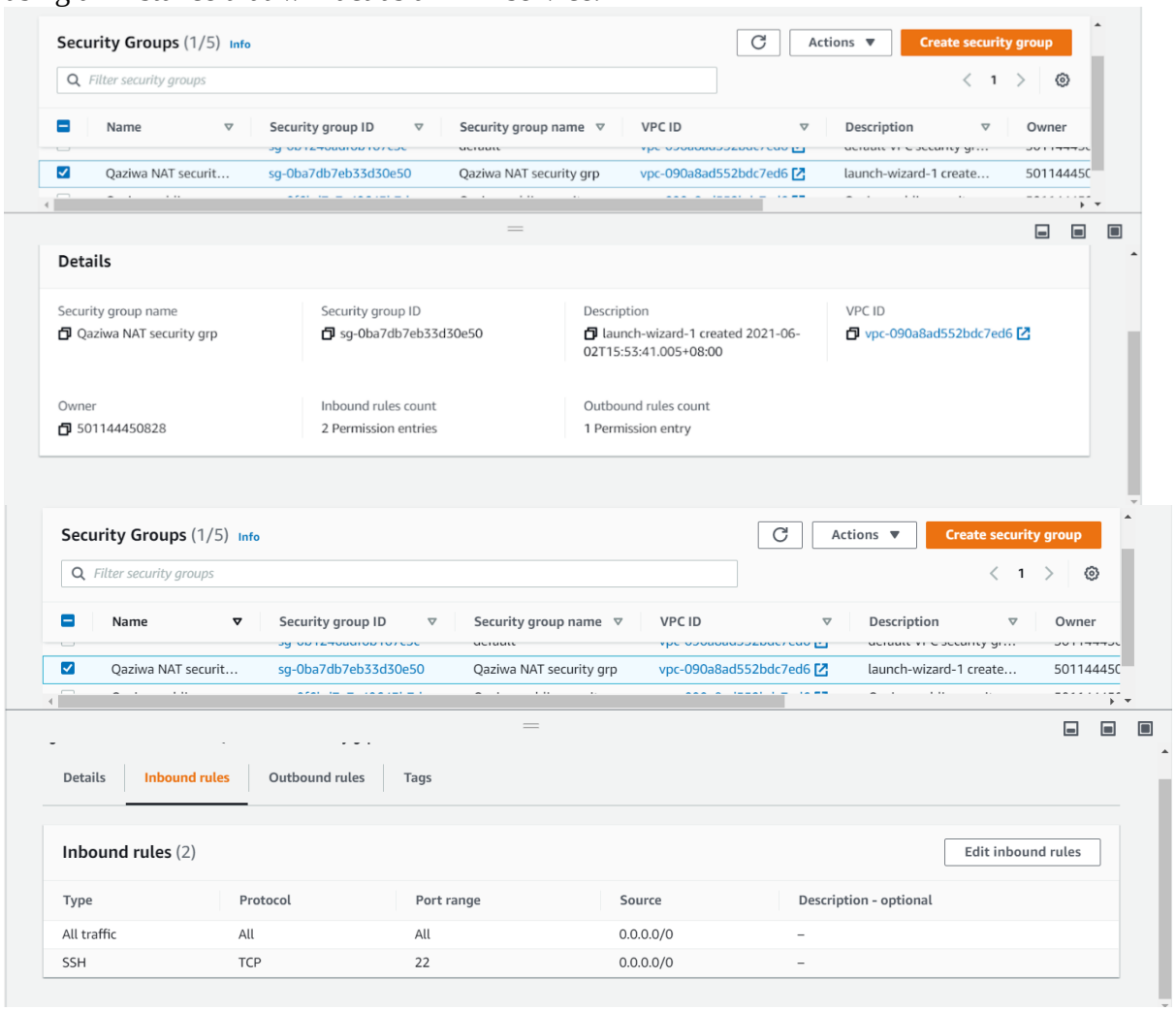
After creating the private security group, now is the time to create the public security group. This is where it is for the web server to access the internet. We have configured the public security group shown below.





For the Public security group, we have enabled SSH for our configuration and port 443 which is HTTPS due to this instance server as a web server. We are able from any source because firstly we want to try access to the web service and to the SSH and also it provides availability to the users who access the website.

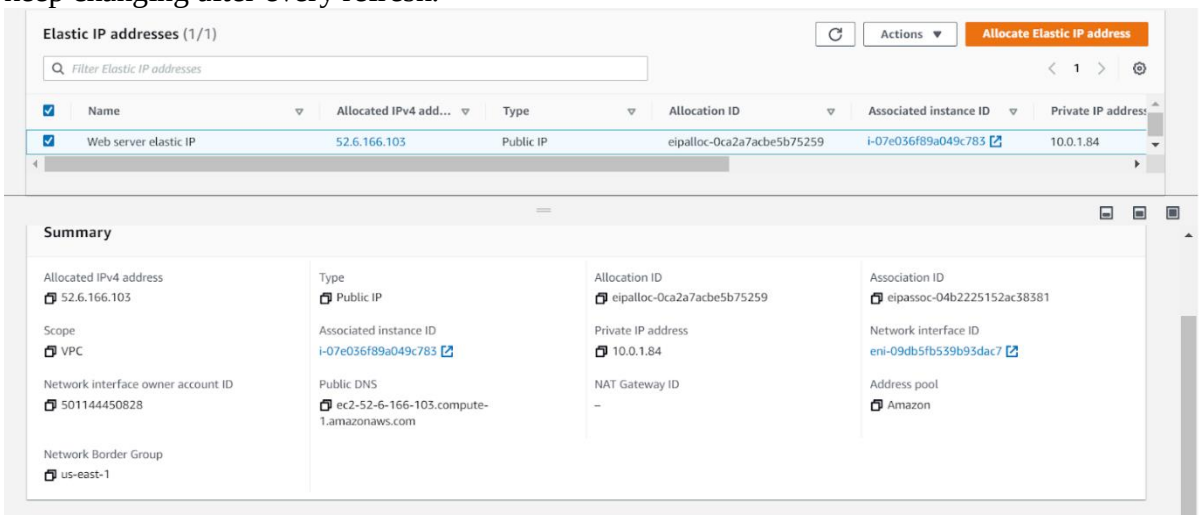
Since we know that due to the limited privileges that the AWS education has to offer and this cause we cannot use the NAT gateway. As such to improvise and mitigate this, we will be using an instance that will act as a NAT service.



NAT security groups have enabled everything due to each instance having its own firewall or security rule and therefore, implementing this is much easier.

Creation of Elastic IP for web service:

Allocate an Elastic IP address for the web server. Without allocating, the IP addresses will keep changing after every refresh.



We have created the Elastic IP in which we will be using later on for the instance implementation. This helps to provide a static public IP address in which it will not keep changing so that our DNS server will help to ensure that it will be able to locate the specific IP address. Now we have settled the configuration for the network, so next, we are ready to start recreating the instance and assign it to the load balancer and give auto-scaling, DNS, and much more to the instance so that it can provide the security and also scalability and even availability to the users.

Creation of web server instance:

To get started we created an Ubuntu instance with a micro size. The web server is then configured with a public IP to allow users to access the webserver. With the instance created we would ensure the port 22 is open for us to implement the website into the instance through there. With the SSH open, we can now connect to the instance via Putty through its public IP. Inside the command line, we would then run as root as cd into the /var/www/html folder to insert the code. Running the wget command we would insert the source code through the bucket URL of the

Putty into the IP.

-> To upgrade the Ubuntu Machine and make sure it is the newest version:

```
ubuntu@ip-10-0-1-84:~$ sudo su
root@ip-10-0-1-84:/home/ubuntu# apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ub
```

Run apt-get update

Run apt-get upgrade

```
Reading package lists... Done
root@ip-10-0-1-84:/home/ubuntu# apt-get upgrade
Reading package lists... Done
```

-> Installing the Apache2 web server with its mods:

Run apt-get install apache2

Run apt-get install php libapache2-mod-php php-mysql

```
root@ip-10-0-1-84:/home/ubuntu# apt-get install php libapache2-mod-php php-mysql
Reading package lists... Done
```

->Getting the Website source code that we have put in the AWS S3 bucket:

Run wget https://qaziwabucket.s3.amazonaws.com/C300.zip

```
root@ip-10-0-1-84:/var/www/html# wget https://qaziwabucket.s3.amazonaws.com/C300.zip
--2021-06-02 06:45:00--  https://qaziwabucket.s3.amazonaws.com/C300.zip
```

-> Installing unzips software to unzip a folder:

Run apt install unzip

-> Unzipping the C300.zip

Run unzip C300.zip

Run mv ‘C300 (new)’ C300 (To rename the folder)

```
  inflating: C300 (new)/PHPMailer/language/phpmailer.
root@ip-10-0-1-84:/var/www/html# ls
'C300 (new)'   C300.zip
root@ip-10-0-1-84:/var/www/html# mv 'C300 (new)' C300
root@ip-10-0-1-84:/var/www/html# ls
```

-> Placing the source code into the /var/www/html

Run mv C300/* (Move all of the files and folders from the C300 folder out)

```
root@ip-10-0-1-84:/var/www/html# mv C300/* .
root@ip-10-0-1-84:/var/www/html# ls
AddBooks.php      GuestList.php          UserList.php       doAddBooks.php     doRegister.php
AdminList.php     Login.php              Verification.php   doAddToCard.php    doVerification.php
C300              PHPMailer              WebDesign          doCartDelete.php   include
C300.zip          PaymentPage.php        advertisement.js   doDelete.php       index.php
Cart.php          Pictures               creatives          doEditBooks.php    logoff.php
DeleteBooks.php   Register.php           dbFunctions.php    doEditBooks2.php   nbproject
EditBooks.php     SuccessfulPayment.php  dbFunctions2.php   doLogin.php        smtp
root@ip-10-0-1-84:/var/www/html#
```

-> Restart the apache2 service

Run service apache2 restart

```
root@ip-10-0-1-84:/var/www/html# service apache2 restart
root@ip-10-0-1-84:/var/www/html#
```

-> Change the source code to match with the database server IP address

Run nano /var/www/html/dbFunctions2.php

```
 GNU nano 4.8                    dbFunctions2.php                 Modified
<?php

$db_host = "10.0.2.239";
$db_username = "qaziwa";
$db_password = "Password";
$db_name = "qaziwa";


/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
```

Our summary command line we used for our Web Server installation to the last configuration of the dbfunctions2.php

```
 1  install apt-get
 2  apt-get update
 3  apt-get upgrade
 4  apt-get install apache2
 5  apt-get install php libapache2 -mod-php php-mcrypt php-mysql
 6  apt-get install php libapache2-mod-php php-mcrypt php-mysql
 7  apt-get install php libapache2-mod-php php-mysql
 8  service apache2 restart
 9  cd /var/www/html
10  ls
11  sudo nano
12  nano test.php
13  ls
14  rm index.html && test.php
15  ls
16  rm test.php
17  ls
18  wget https://qaziwabucket.s3.amazonaws.com/C300.zip
19  ls
20  unzip
21  apt install unzip
22  ls
23  unzip C300.zip
24  ls
25  mv 'C300 (new)' C300
26  ls
27  cd /var/www/html
28  ls
29  cd /var/www/html
30  ls
31  mv C300/* .
32  ls
33  service httpd start
34  service apache2 restart
35  ls
```

-> Change of the permission of the Pictures folder to enable adding images into the folder.

```
49  ls | grep "doA"
50  nano doAddBooks.php
51  ls
52  chmod 777 Pictures
53  ls -l
54  cd /var/www/html
55  ls
56  history
root@ip-10-0-1-84:/home/ubuntu#
```

We download the apache2 and the PHP so that we can run the code, then after that, we use wget to install the website source code in which to be used in the webserver. After installing the source code, we unzip it and store it in the directory /var/www/html and move the C300/ folder inside the HTML folder. After that, we restart the apache server and edit a source code tallied with the database server IP address.

Server Logs

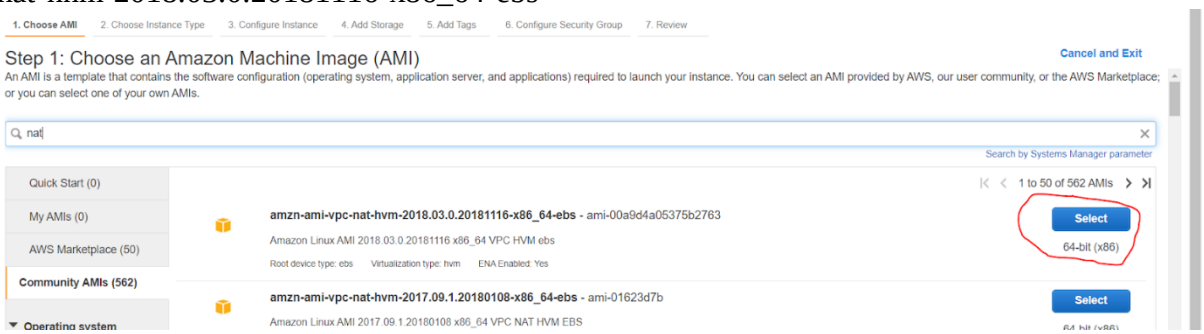
- `/var/log/apache2/access.log`: By default, every request to your web server is recorded in this log file unless Apache is configured to do otherwise.
- `/var/log/apache2/error.log`: By default, all errors are recorded in this file. The `LogLevel` directive in the Apache configuration specifies how much detail the error logs will contain.

Now from above, here is the logs directory we can check and configuration of the apache server. Once we have done setting up the web server, now is the time for us to implement the database server to fully make use of the website. But before the database is being set up we need to create the NAT instance so that we are able to install the phpmyadmin for our database.

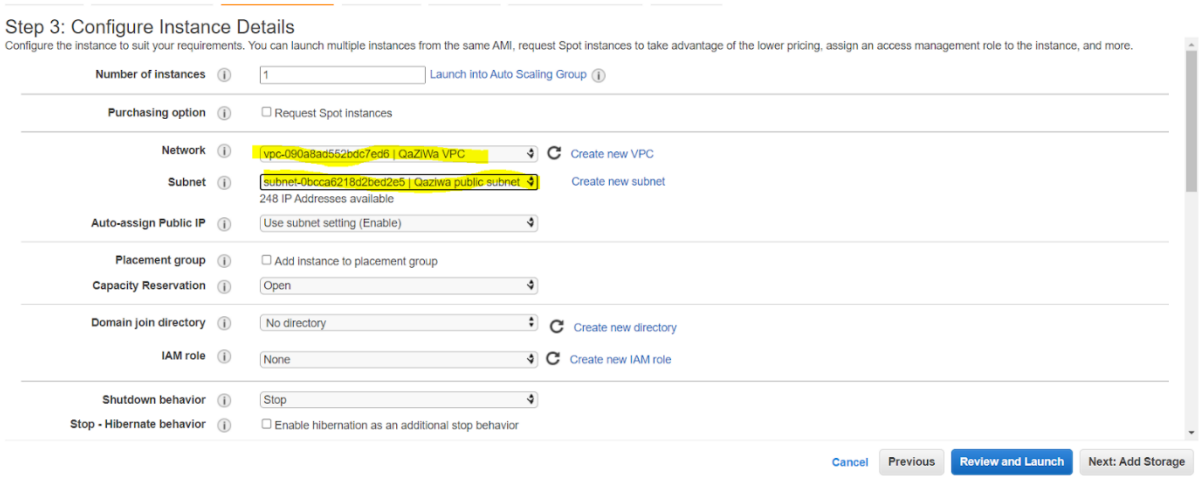
<u>Creation of NAT instance:</u>

Since AWS education, we cannot use the NAT function, therefore, we need to use the configuration of the NAT instance instead. By doing this it enables the private IP address to connect to the internet but the internet cannot connect to the private IP, this function helps to keep the database server secure and at the same time enables it to download the resources too. Below shows the configuration of the NAT instance

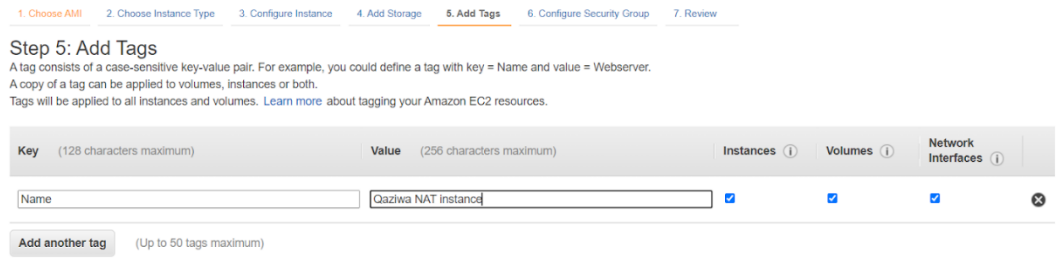
First, we create an instance by using a NAT instance.
-> Creation of the NAT instance
Goto EC2 > Launch Instance > Community AMIs > search NAT > click the amzn-ami-vpc-nat-hnm-2018.03.0.20181116-x86_64-ebs



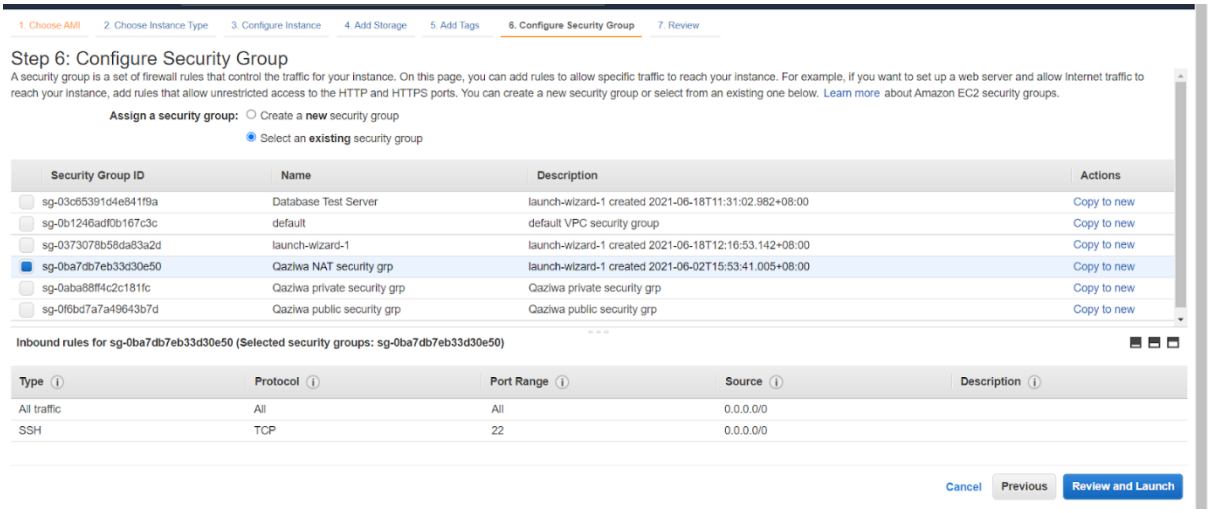
Goto Configuration Instance Details → Select our own VPC (Qaziwa VPC) → Select the public Subnet) > Click next: Add Storage



Goto to Add Tags → Key in the name and value (shown below) > Click Next: Configure Security Group

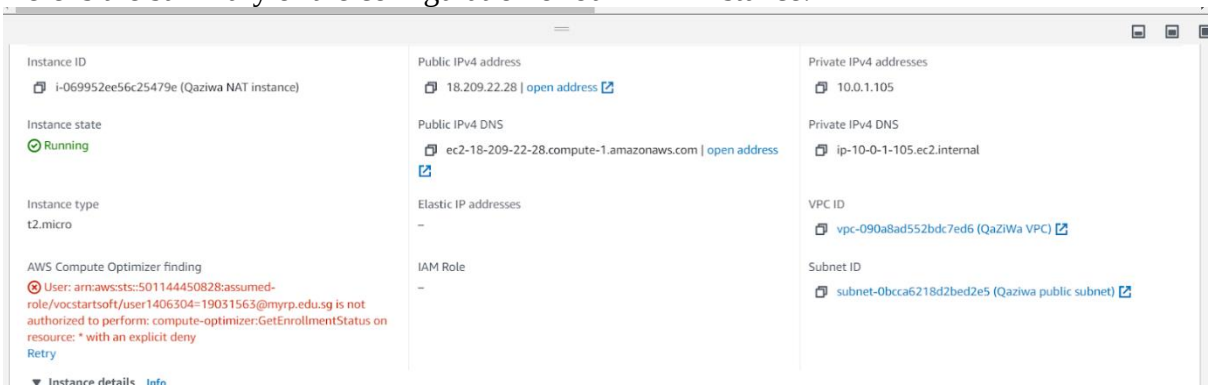


Goto Configure Security Group → Click the security group that we have already created for NAT instances.

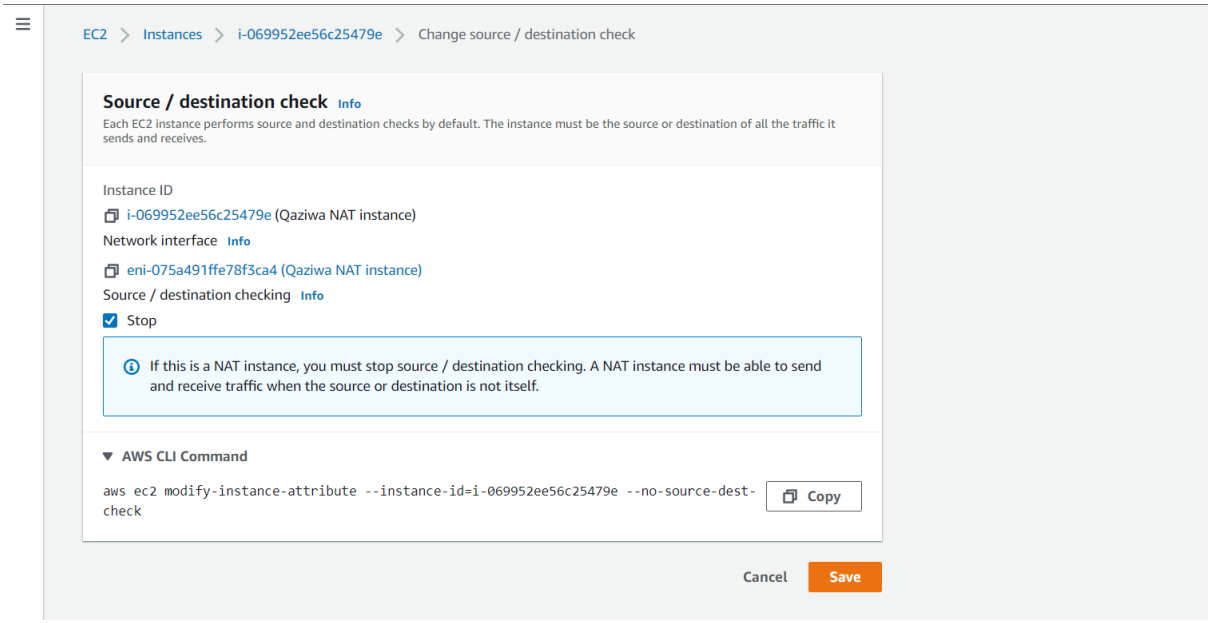


We want to install the database server inside the private group, therefore, we have enabled all traffic. After that click review and launch of this NAT instance

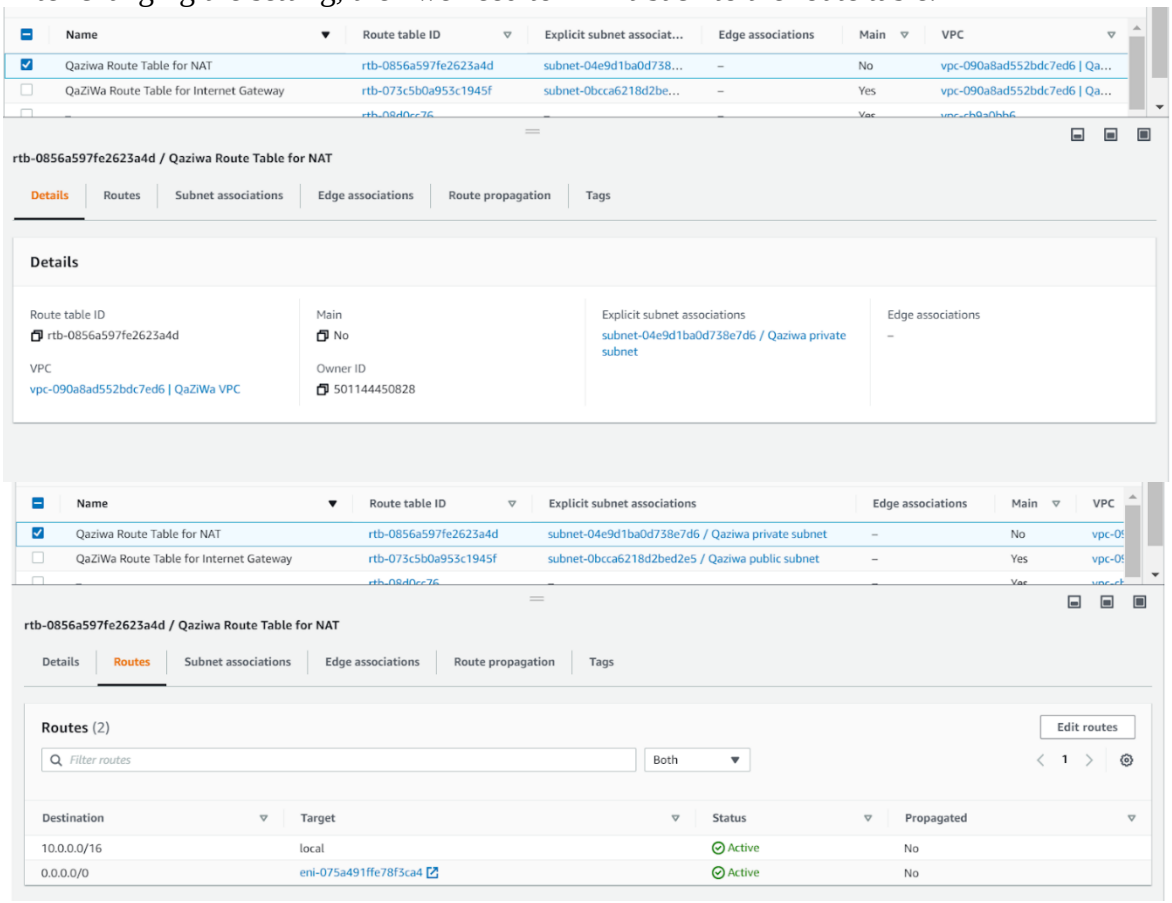
Here is the summary of the configuration of our NAT instance:



After creating an instance, we need to stop the source/destination checking and click on save



After changing the setting, then we need to link it back to the route table.



After changing the Route Table, we also need to change the private subnet so that the instance from the private subnet can connect to the NAT instance, and from the NAT instance the Traffic and flow out to the routing table and to the internet gateway and out to the internet. Since the NAT instance is fully operational now is the time to set up the database instance.

Creation of database server:

```
Last login: Wed Jun  2 10:42:59 2021 from 119.56.96.205
ubuntu@ip-10-0-1-84:~$ history
    1  ping 10.0.2.239
    2  ping google.com
    3  sudo su
    4  cd /var/www/html
    5  ls
    6  sudo su
    7  ls
    8  sudo ssh ubuntu@10.0.2.239 -i QaziwaKey.pem
    9  history
ubuntu@ip-10-0-1-84:~$
```

From the Web Server, we tried to ping into the database server to check the connection with it. After getting the response, we tried to copy the SSH private key and paste it into the webserver using sudo nano QaziwaKey.pem. Then we change the permission to 0400 and use ssh ubuntu@10.0.2.239 -i QaziwaKey.pem to get into the database.

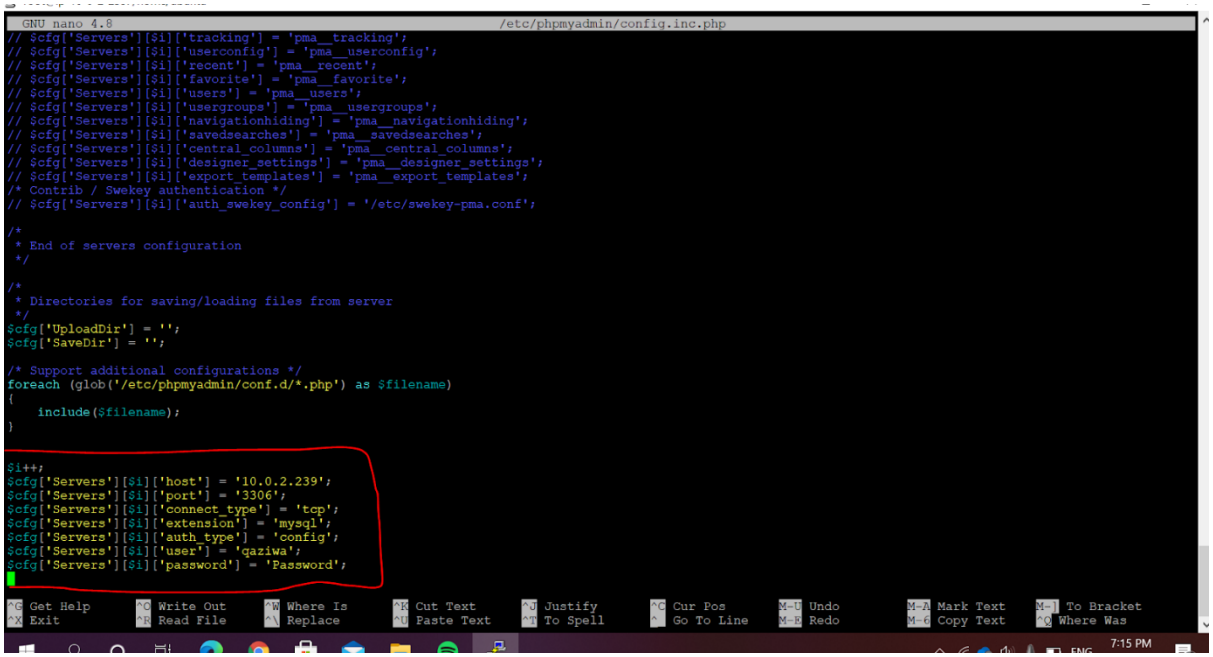
```
8 updates can be applied immediately.
7 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable


*** System restart required ***
Last login: Wed Jun  2 09:36:24 2021 from 10.0.1.84
ubuntu@ip-10-0-2-239:~$
```

After getting into the Database, we tried to ping google.com and to see if there was any response. In addition, we have already set up a NAT instance that enables us to connect to the internet without using any public IP address from the Database Server instance.

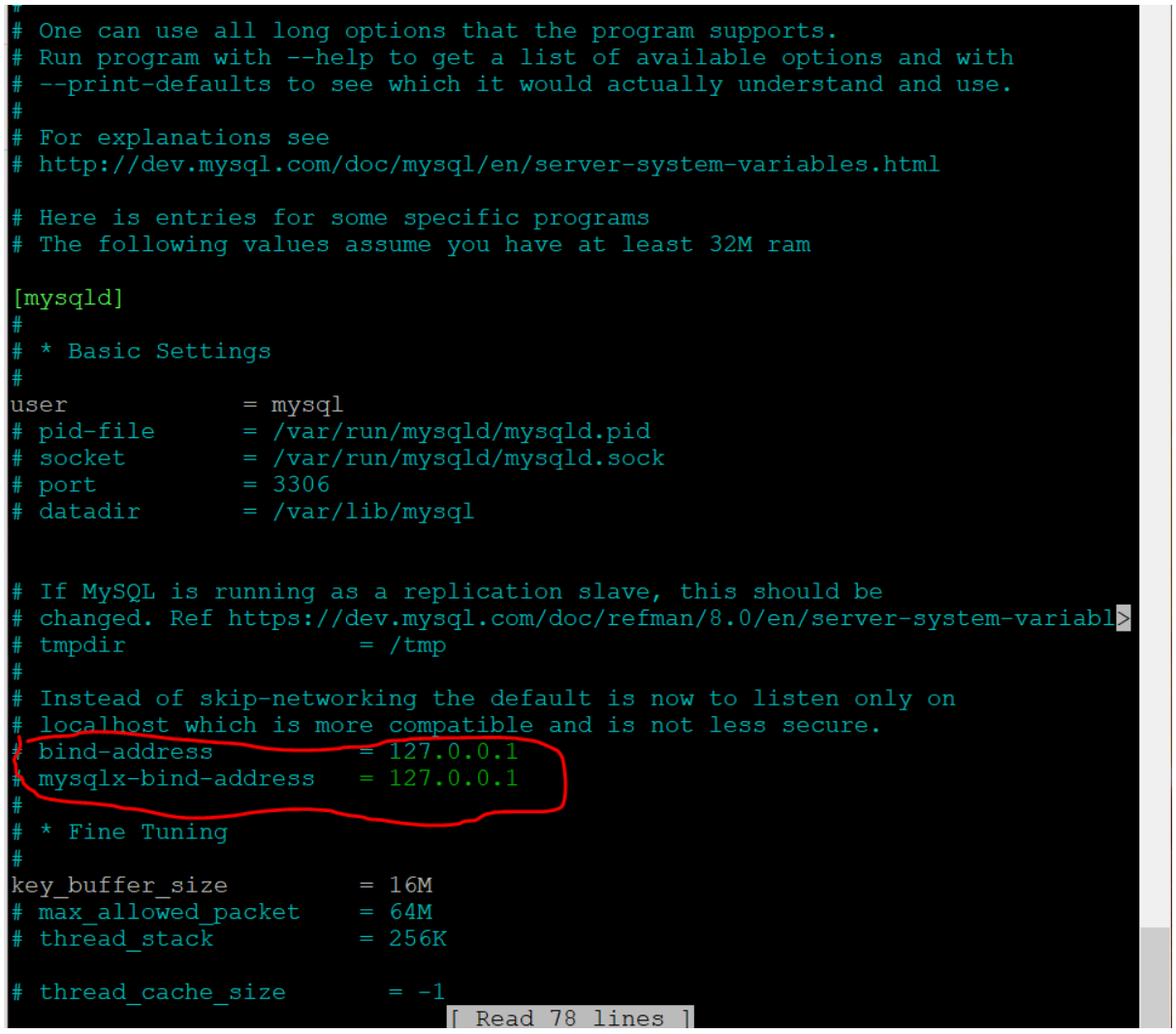
After we can see that it can connect into the database, we then use the command "apt-get install lamp-server^" → use "apt-get install PHPMyAdmin -y" (choose apache2 > yes > key in password > confirm password) after downloading the phpmyadmin, we then enter into the mysql database and create a user using "mysql -p -u root" → "CREATE USER 'qaziwa'@'%' IDENTIFIED BY '<The password>';" → "GRANT ALL PRIVILEGES ON *.* TO 'qaziwa'@'%' WITH GRANT OPTION;"

```
mysql> CREATE USER 'qaziwa'@'%' IDENTIFIED BY 'Password';
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT ALL PRIVILEGES ON *.* TO 'qaziwa'@'%' WITH GRANT OPTION;
Query OK, 0 rows affected (0.01 sec)
```

After creating the user, then I need to go to the config.inc.php to edit the file and make out my own profile inside it. Shown below:



After configuring we need to go to the nano /etc/mysql/mysql.conf.d/mysqld.cnf and comment out the bind-address. Shown below:



After save the file, and restart the mysql using “systemctl restart mysql”

After that we connect to the user using “mysql -h 10.0.2.239 -u qaziwa -p” > then now we want to create a database using “CREATE DATABASE qaziwa;” then click exit then import the qaziwa.sql from the S3 bucket using wget and the URL. Shown below

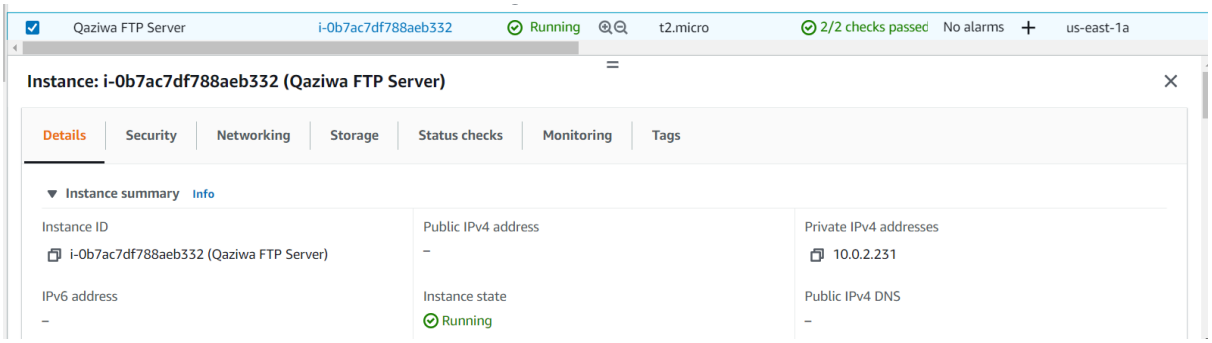
```
  8  wget https://qaziwabucket.s3.amazonaws.com/qaziwa.sql
  9  ls
 10  mysql -h 10.0.2.239 -u qaziwa -p qaziwa < qaziwa.sql
 11  sudo su
 12  history
buntu@ip-10-0-2-239:~$
```

After importing the qaziwa.sql then we will import it to the database using the command “mysql -h 10.0.2.239 -u qaziwa -p qaziwa < qaziwa.sql”. Then from there, we test the database and the website.

Since we have fully configured the web server, NAT instance and database server now are the time to create the FTP Server instance.

Creation of FTP server instance:

FTP server helps us to store and get any files content be it an image file, text file, or any other files. Since Load Balancing is needed, we have an idea that if we upload an image file to one website, the other website may not be updated. This can cause users to not have a full experience when using the website and this could lead to decrease of its functions. As such we plan and implement a FTP server in which it will help us to get and store image files whenever the web server uploads or needs the image to display to the user. Since our website has an upload function, we need to upload it into our FTP server in which when the web server needs the file, it will connect to the FTP server and take in the file that is needed which it will be depending on the database server image file name. As such now is the configuration of the FTP server. First we create an instance and ensure that the FTP server is in the private network and the same network as the database instance.



After creation of the instance, Putty inside the web server and then ssh inside the FTP server. Since we already created a NAT instance, we can use it to down a FTP resource package.

To install the FTP server we need be in root account and then run apt-get update and apt-get install vsftpd

```
sudo apt-get update
sudo apt-get install vsftpd
```

After the installation has been completed, the service is disable by default, so therefore, we need to start it automatically.

```
systemctl start vsftpd
systemctl enable vsftpd
```

Since we have already install, now is to configure the vsftpd.conf which is located at /etc/vsftpd.conf

```
sudo nano /etc/vsftpd.conf
```

After this we enable the requirements shown below

```
anonymous_enable=NO             # disable  anonymous login
local_enable=YES                # permit local logins
write_enable=YES                # enable FTP commands which change th
local_umask=022                 # value of umask for file creation fo
dirmessage_enable=YES           # enable showing of messages when use
xferlog_enable=YES              # a log file will be maintained detai
connect_from_port_20=YES        # use port 20 (ftp-data) on the serve
xferlog_std_format=YES          # keep standard log file format
listen=NO                       # prevent vsftpd from running in stan
listen_ipv6=YES                 # vsftpd will listen on an IPv6 socke
pam_service_name=vsftpd         # name of the PAM service vsftpd will
userlist_enable=YES             # enable vsftpd to load a list of use
tcp_wrappers=YES                # turn on tcp wrappers
```

```
chroot_local_user=YES
allow_writeable_chroot=YES
```

```
userlist_enable=YES
userlist_file=/etc/vsftpd.userlist
userlist_deny=NO
```

After enabling the setting shown above, we then need to restart the FTP service using the systemctl restart vsftpd.

Once it has been restarted, we need to create the username called Qaziwa with a password as a Password. So, we use the command “useradd -m -c “Qaziwa” -s /bin/bash Qaziwa” and then we need to set the password so we need to key in “passwd Qaziwa” and from there they will ask for the key in the password. The password for this will be Password.

After this, we want to add user to the vsftpd.userlist by using “echo “Qaziwa” | sudo tee -a /etc/vsftpd.userlist” and to double clarify if the Qaziwa username is being added we do a “cat /etc/vsftpd.userlist”.

Now is the time to test the FTP service. Firstly we try to login into an Anonymous account to see if we can connect or not. Since we know that the anonymous setting in the FTP server has been set to no, which means it would not let us login as anonymous.

```
ubuntu@ip-10-0-1-48:~$ ftp 10.0.2.231
Connected to 10.0.2.231.
220 (vsFTPd 3.0.3)
Name (10.0.2.231:ubuntu): Anonymous
530 Permission denied.
Login failed.
ftp> exit
221 Goodbye.
ubuntu@ip-10-0-1-48:~$ ftp 10.0.2.231
Connected to 10.0.2.231.
220 (vsFTPd 3.0.3)
Name (10.0.2.231:ubuntu): anonymous
530 Permission denied.
Login failed.
ftp>
```

As shown above, we can see that we are unable to run as an anonymous user.

So now let use with the user we have created in the previous section with the username “Qaziwa” and password “Password”

```
ubuntu@ip-10-0-1-48:~$ ftp 10.0.2.231
Connected to 10.0.2.231.
220 (vsFTPd 3.0.3)
Name (10.0.2.231:ubuntu): Qaziwa
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxrwx---    2 1001     1001         4096 Jul 29 18:14 files
226 Directory send OK.
ftp>
```

We have successfully logged in and are able to log as a Qaziwa user in which will be used to store all of the image files that the admin has uploaded.

Now is to configure the home directories for the FTP user.
Firstly we do:
mkdir /home/Qaziwa/ftp/files

chown -R Qaziwa:Qaziwa /home/Qaziwa/ftp/files
chmod -R 0770 /home/Qaziwa/ftp/files

After that, we would add the settings shown below

```
user_sub_token=$USER          # inserts the username in the local root
local_root=/home/$USER/ftp    # defines any users local root directory
```

After that do a restart of the FTP service by using the command “systemctl restart vsftpd”

Since we have restarted, now we test if we have a file folder and are able to access the file folder.

```
ubuntu@ip-10-0-1-48:~$ ftp 10.0.2.231
Connected to 10.0.2.231.
220 (vsFTPd 3.0.3)
Name (10.0.2.231:ubuntu): Qaziwa
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxrwx---    2 1001     1001         4096 Jul 29 18:14 files
226 Directory send OK.
ftp> cd files
250 Directory successfully changed.
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rw-r--r--    1 1001     1001       167908 Jul 28 20:30 beano_front.jpg
-rw-r--r--    1 1001     1001        92954 Jul 28 20:34 cat_hat_front.jpg
-rw-r--r--    1 1001     1001         2034 Jul 29 16:35 download.jpg
-rw-r--r--    1 1001     1001       105143 Jul 28 20:31 garfield_front.jpg
-rw-r--r--    1 1001     1001        95222 Jul 28 20:31 ghost_stories_front.jpg
-rw-r--r--    1 1001     1001       110324 Jul 28 20:32 hard_luck_front.jpg
-rw-r--r--    1 1001     1001         5407 Jul 29 18:14 hello.jpg
-rw-r--r--    1 1001     1001       100374 Jul 28 20:31 insomnia_front.jpg
-rw-r--r--    1 1001     1001       140113 Jul 28 20:31 kid_front.jpg
-rw-r--r--    1 1001     1001       108025 Jul 28 20:33 long_haul_front.jpg
-rw-r--r--    1 1001     1001       122984 Jul 28 20:31 old_school_front.jpg
-rw-r--r--    1 1001     1001        99563 Jul 28 20:32 passing_front.jpg
-rw-r--r--    1 1001     1001       174397 Jul 28 20:32 transformers_front.jpg
226 Directory send OK.
ftp>
```

Now we are able to get and put the image file inside the FTP server. Once that is done then we need to upload the file and download the files from the FTP server using the PHP file.

After intensive research, we manage to upload and download the file to and from the FTP server. This can be seen below.

Below is the upload from the web server to the FTP server. This is when the admin wants to upload a new book image so that the user can see the book image.

```php
$ftp_host = "10.0.2.231";
$ftp_username = "Qaziwa";
$ftp_password = "Password";

// open an FTP connection
$conn_id = ftp_connect($ftp_host) or die("Couldn't connect to $ftp_host");

// login to FTP server
$ftp_login = ftp_login($conn_id, $ftp_username, $ftp_password);

// local & server file path
$localFilePath = "/var/www/html/Pictures/$fileName";
$remoteFilePath = "files/$fileName";

// try to upload file
if (ftp_put($conn_id, $remoteFilePath, $localFilePath, FTP_BINARY)) {
    echo "File transfer successful - $localFilePath";
} else {
    echo "There was an error while uploading $localFilePath";
}

// close the connection
ftp_close($conn_id);
```

Next is to download the file from the FTP server to the web server. This is much harder as we do not want all images to be downloaded to our web server as this can cause slowness. As such we have come up with an idea to only accept the image name that tally with the database name.

+ Options

Click the drop-down arrow to toggle column's visibility.

	title_id	title	author	summary	rating	price	quantity	picture_front
Edit Copy Delete		The Cat In The Hat	Dr. Suess	Poor Dick and Sally. It's cold and wet and they're...	5	13.57	8	cat_hat_front.jpg
Edit Copy Delete	3	Garfield 30 Years	Jim Davis	30 years of laugh and lasagna bundled into one boo...	4	8.16	43	garfield_front.jpg
Edit Copy Delete	4	True Singapore Ghost Stories 11	Russel Lee	Russel Lee investigates witchcraft and uncovers it...	2	10.59	7	ghost_stories_front.jpg
Edit Copy Delete	5	Diary of a Wimpy Kid Hard Luck	Jeff Kinney	Greg Heffley's on a losing streak. His best friend ...	4	14.93	0	hard_luck_front.jpg
Edit Copy Delete	6	Insomnia	Stephen King	You'll lose a lot of sleep. Ralph does. At first h...	3	16.95	14	insomnia_front.jpg
Edit Copy Delete	7	When I was a kid 2	Boey	A continuation of the hit book "When I Was a Kid" ...	3	19.24	4	kid_front.jpg
Edit Copy Delete	8	Diary of a Wimpy Kid Long Haul	Jeff Kinney	A family road trip is supposed to be a lot of fun ...	4	20.87	20	long_haul_front.jpg
Edit Copy Delete	9	Diary of a Wimpy Kid Old School	Jeff Kinney	Life was better in the old days. Or was it? That's ...	2	20.87	13	old_school_front.jpg
Edit Copy Delete	10	As I was Passing II	Adibah Amin	Adibah Amin, the celebrated chronicler of everyday...	3	19.90	3	passing_front.jpg
Edit Copy Delete	11	Transformers Saga of the Allspark	Simon Furman	Saga of the Allspark reveals untold events from th...	5	17.99	25	transformers_front.jpg

Our database consists of the image file name in which we can download the specific books we have implemented. If the admin adds a new book, the database will be uploaded, and from the web server, which will tally the amount of file name to be downloaded and get from the FTP server.

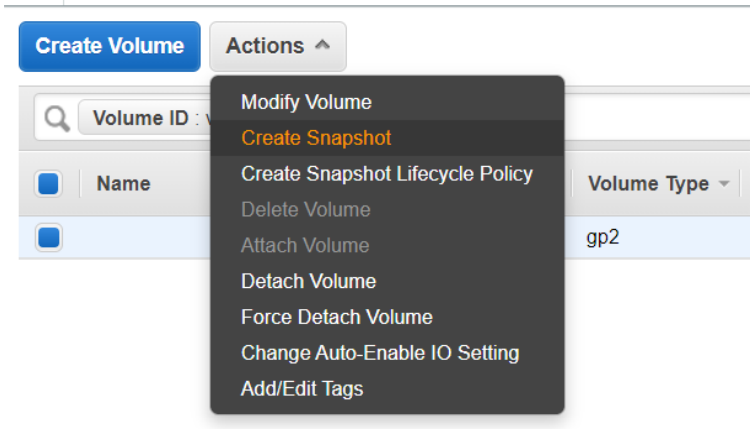
```php
$ftp_host = "10.0.2.231";
$ftp_username = "Qaziwa";
$ftp_password = "Password";

// open an FTP connection
$conn_id = ftp_connect($ftp_host) or die("Couldn't connect to $ftp_host");

// login to FTP server
$ftp_login = ftp_login($conn_id, $ftp_username, $ftp_password);

try {
    $conn99 = new PDO("mysql:host=$db_host;dbname=$db_name", $db_username, $db_password);
    $conn99->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    $stmt99 = $conn99->prepare("SELECT * FROM books");

    $stmt99->execute();
} catch (Exception $ex) {
    echo "Error: " . $ex->getMessage();
}
$conn99 = null;

//$query2 = "SELECT * FROM books";
// To secure, use mysqli_escape_string or prepared statements
//$result2 = mysqli_query($link, $query2) or die(mysqli_error($link));

while ($row99 = $stmt99->fetch()) {
    $arrProducts99[] = $row99;
}
for ($i = 0; $i < count($arrProducts99); $i++) {
    // $title = $arrProducts[$i]['title'];
    // $author = $arrProducts[$i]['author'];
    //$summary = $arrProducts[$i]['summary'];
    // $rating = $arrProducts[$i]['rating'];
    // $price = $arrProducts[$i]['price'];
    //   $quantity = $arrProducts[$i]['quantity'];
    $pictures99 = $arrProducts99[$i]['picture_front'];
    $localFilePath = "/var/www/html/Pictures/$pictures99";
    $remoteFilePath = "files/$pictures99";

// try to upload file
    if (ftp_get($conn_id, $localFilePath, $remoteFilePath, FTP_BINARY)) {
        echo "File transfer successful - $localFilePath";
    } else {
        echo "There was an error while uploading $localFilePath";
    }
}
ftp_close($conn_id);
```

As shown above, first we connect to the FTP server and then we connect to the database server by getting in the file name that is inside the database server. After it receives all of the file names, then it will send over to the web server and from the web server, it will send to the FTP server and ask for the specific file name that is needed to be downloaded. Upon successful download, a message will appear stating a “File transfer successful” and then the file that has been downloaded. As such we can use this FTP server as our Upload and download of files. In addition, we also have uploaded the file in our web server folder in which it could act as a backup if there is a need to use it. After all of this, now is the time to create the snapshot for our instance.

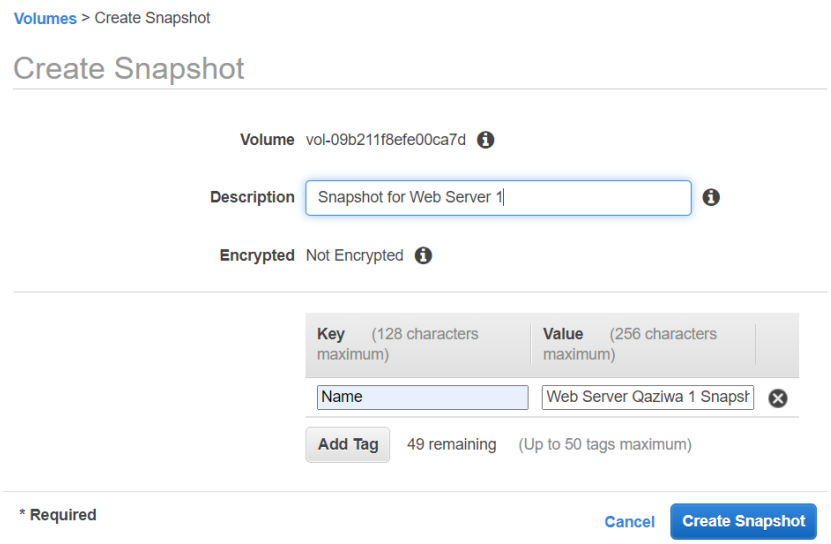
Creation of Instance Snapshot:

First, we go into our targeted instance and scroll to its root device and click on the EBS ID to be redirected to the volume interface.

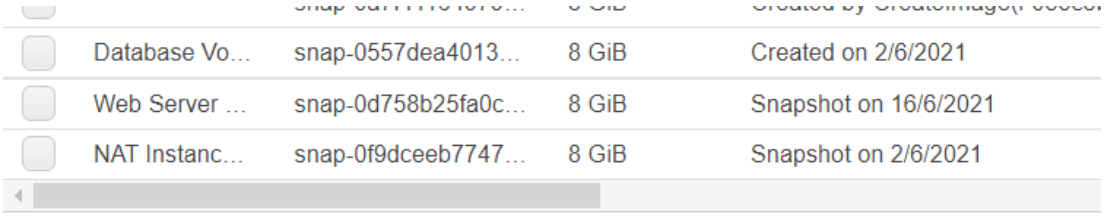
We would then go to actions and click on create a snapshot to snapshot the instance volume.



We would then label the snapshot we are creating and in this case, since we are snapshotting the web server we would label it as such.



After snapshotting all the necessary instances we can go into the snapshot tab and locate all the snapshot that has been created.

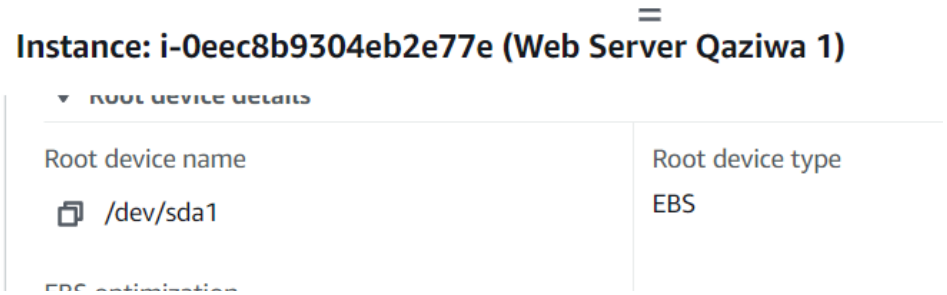


Database Vo...	snap-0557dea4013...	8 GiB	Created on 2/6/2021
Web Server ...	snap-0d758b25fa0c...	8 GiB	Snapshot on 16/6/2021
NAT Instanc...	snap-0f9dceeb7747...	8 GiB	Snapshot on 2/6/2021

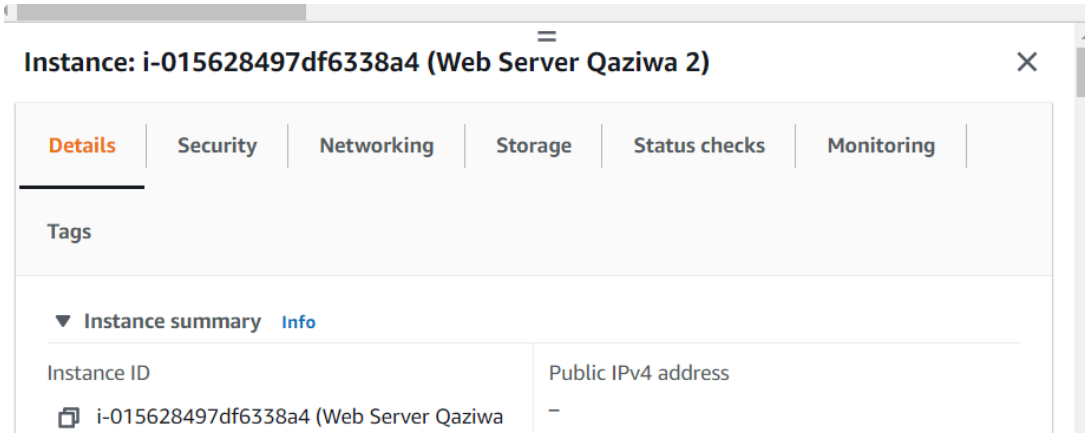
We have successfully created the instance snapshot, so all of the information will be safe inside and if there is any instance accidentally terminated, we still can use this snapshot to recover all of our saved work. Now is the time to show how we can recover the instance configuration back to the original after an instance has been accidentally terminated.

Using Instance Snapshot:

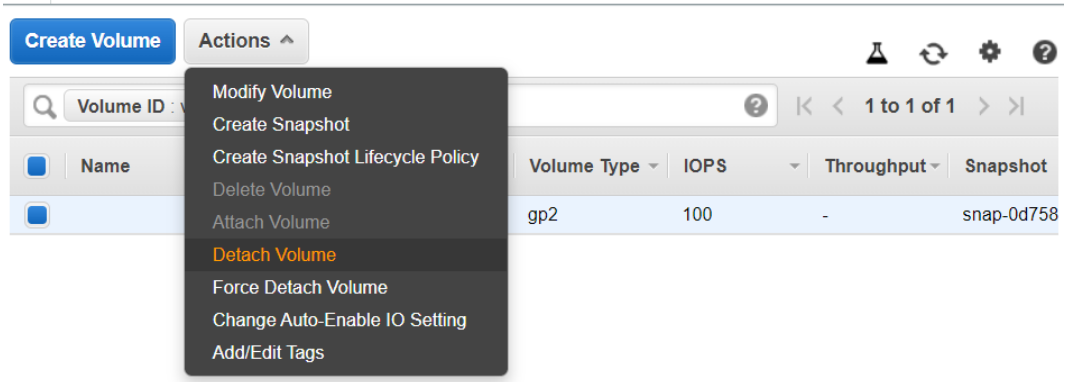
First, we would need to take note of the Root Device of the instance that we are taking the snapshot from in this case it would be /dev/sda1



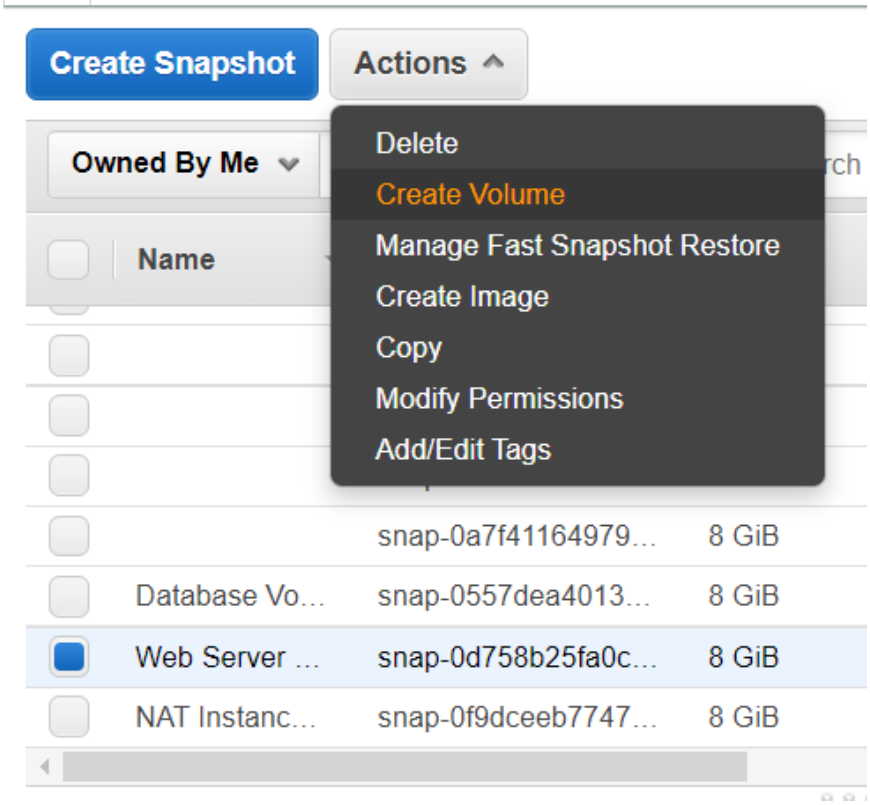
Next, we would create a new instance with all the same settings as the instance that we are taking the snapshot from ensuring that it is in the same availability zone and all that.



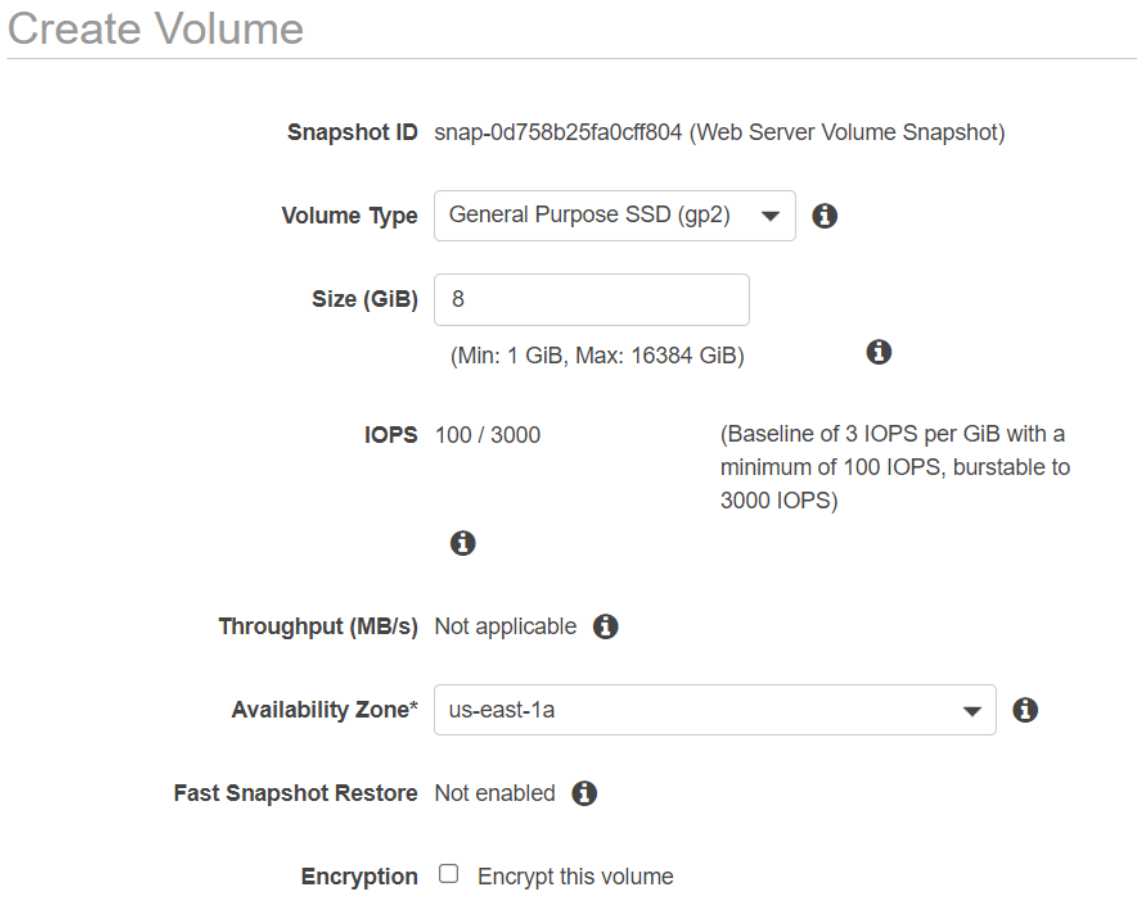
With the instance created we can now go to the volume and detach the current volume (ensure instance is turned off)



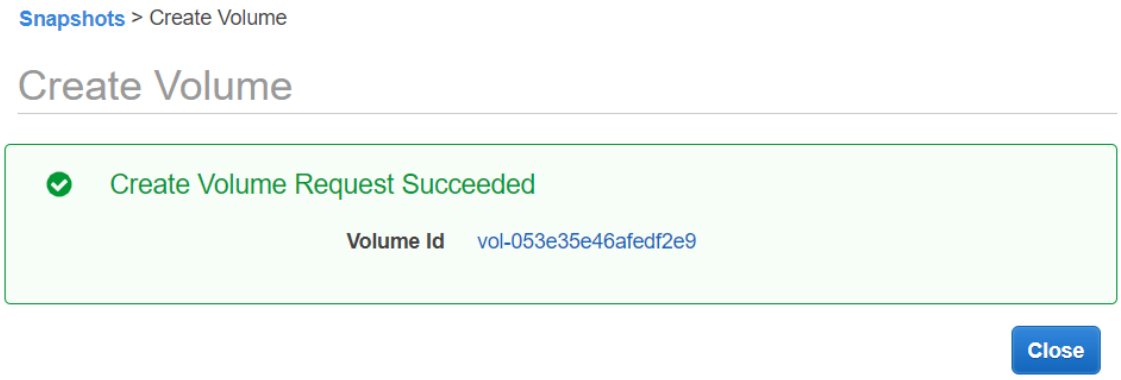
Once detach going back to the snapshot tab we would then choose the snapshot we are taking and create a new volume from actions.



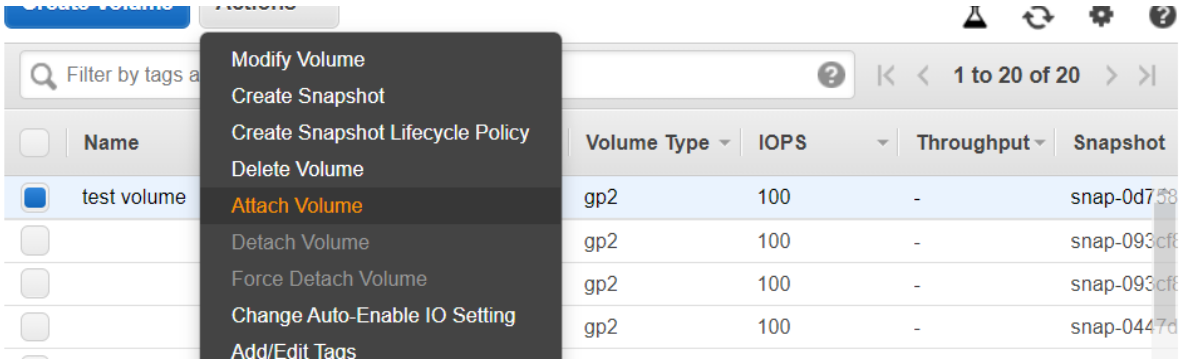
Make sure the volume is located in the same availability zone as the instance.



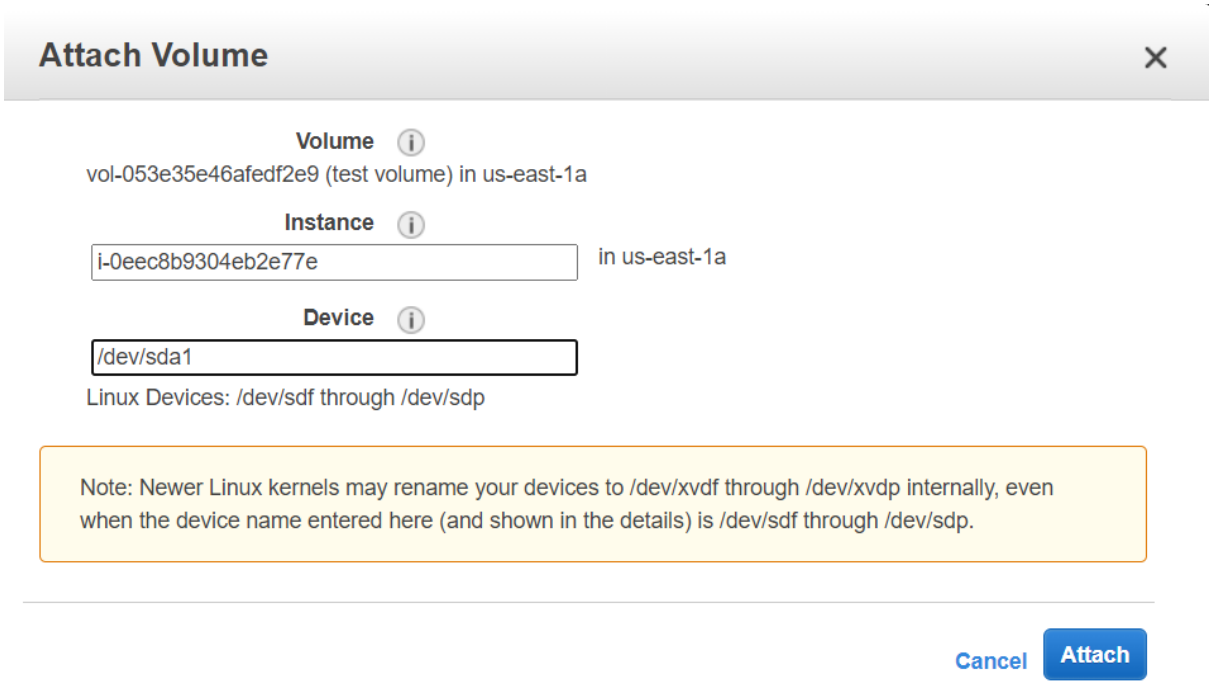
After creating the volume, we would take note of the volume ID



Going into the volumes tab we would locate our newly created volume that has the instance snapshot and attach it to our instance.



We would then need to specify the instance we would be attaching it to, in this case, it would be Web Server 2 and as such we would key in the instance ID of that instance or just search it up by name. We would also need to ensure the device is the same as the root device that was mentioned earlier for successful restoration.

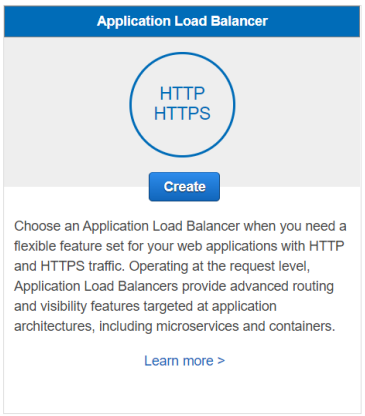


If there is any accidental termination of the instance, we can recover it using the snapshot that we have done. Since of the instance settings and configuration we have done, now is the time to create the load balancer in which to provide distribution of load evenly to the web server.

Creating of Load balancing:

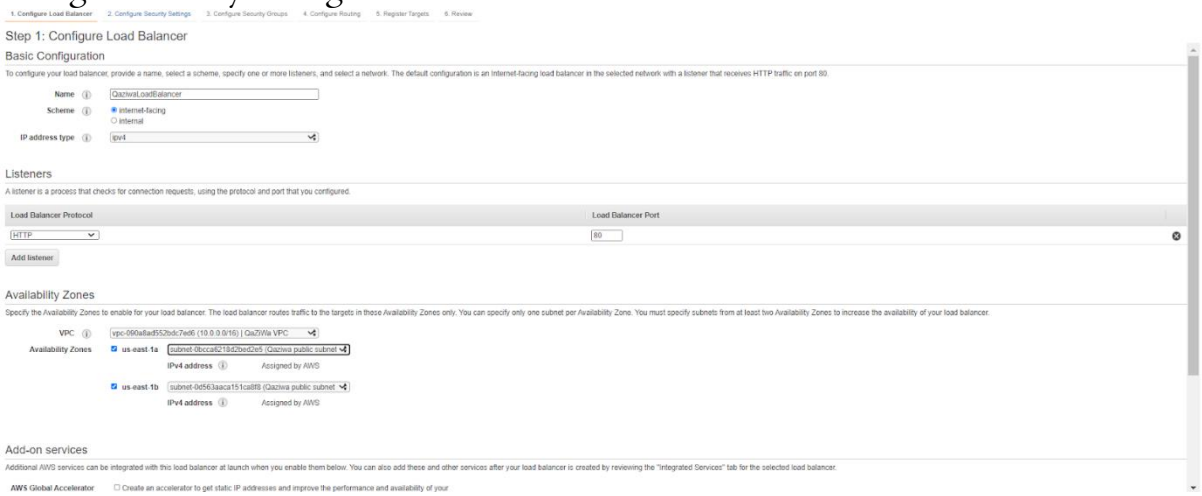
Since our server may experience heavy network traffic, a load balancer will be implemented for disturbing incoming network traffic across multiple servers/targets to prevent instances from crashing in multiple availability zones. Load balancing also monitors the health of registered targets ensuring it routes traffic to only healthy targets. The few components in a load balancer are the load balancer itself, listener and target group. The load balancer will distribute traffic as mentioned earlier. The listener will check for connection requests from the client from our configuration. For every target group, it will route traffic to registered targets which is the instance. Other than that, the load balancer needs to support at least 2 availability zones, so the public subnet 2 & 2 we have created above, we can utilize it and use the load balancer to support the two public subnets with 2 different availability zones.

In our network, we use an application load balancer (ALB) which allows configuring and routing incoming end-user traffic to applications based on the public cloud. ALB inspect packers and will create access points to HTTP and HTTPS header. The ALB in our network is connected from our instances. In addition, we also configure stickiness in our ALB so that it only sticks to 1 instance at a time.

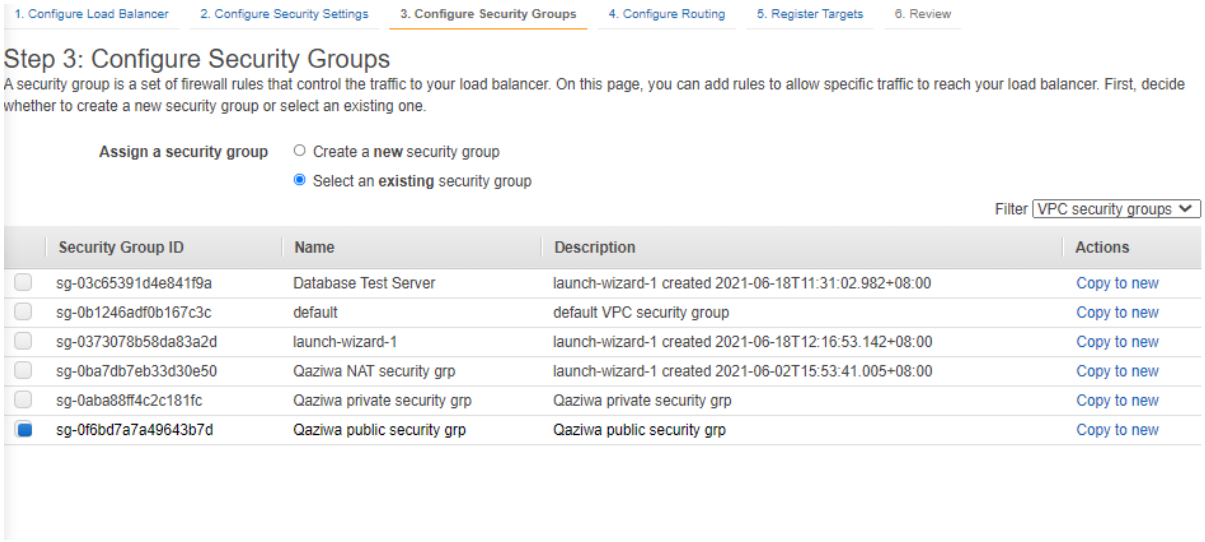


We choose ALB as our load balancer

Then go to configure load balancer → Name (QaizwaLoadBalancer) → internet-facing → IPV4 → HTTP (port 80) → Qaziwa VPC → Qaziwa Public Subnet 1 & 2 > then go to configure security settings

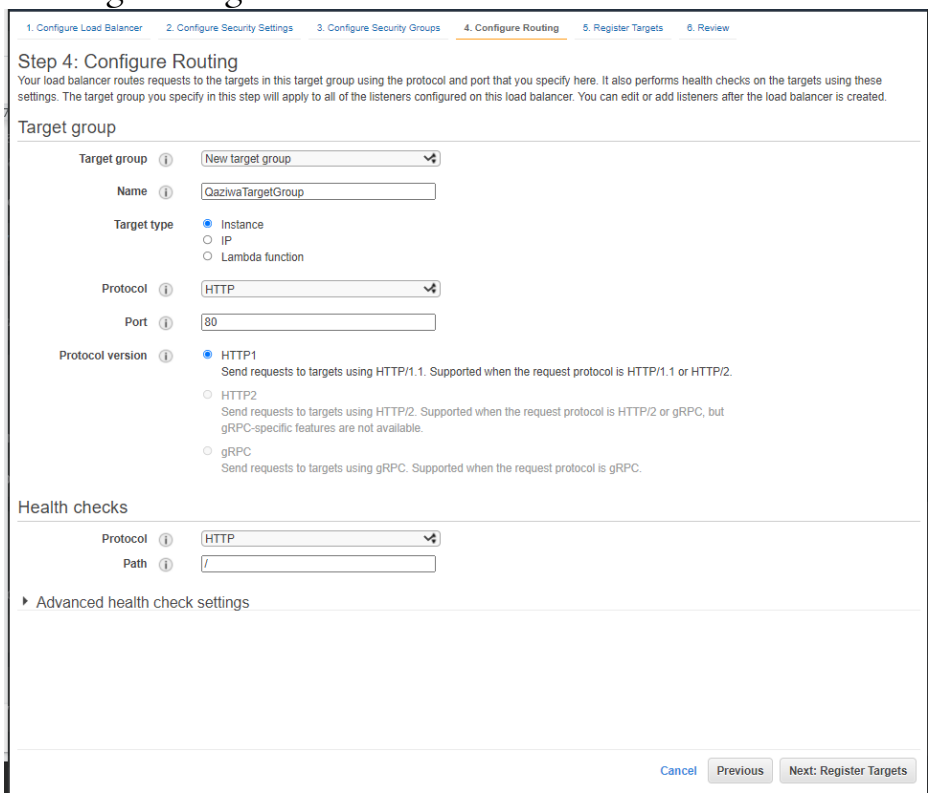


The configure security settings leave as default due to we did not enable HTTPS (due to we do not have the AWS privilege to do that) > Click Next: Configure Security Group → We choose Qaziwa Public Security Group (It enable port 80 which is HTTP)

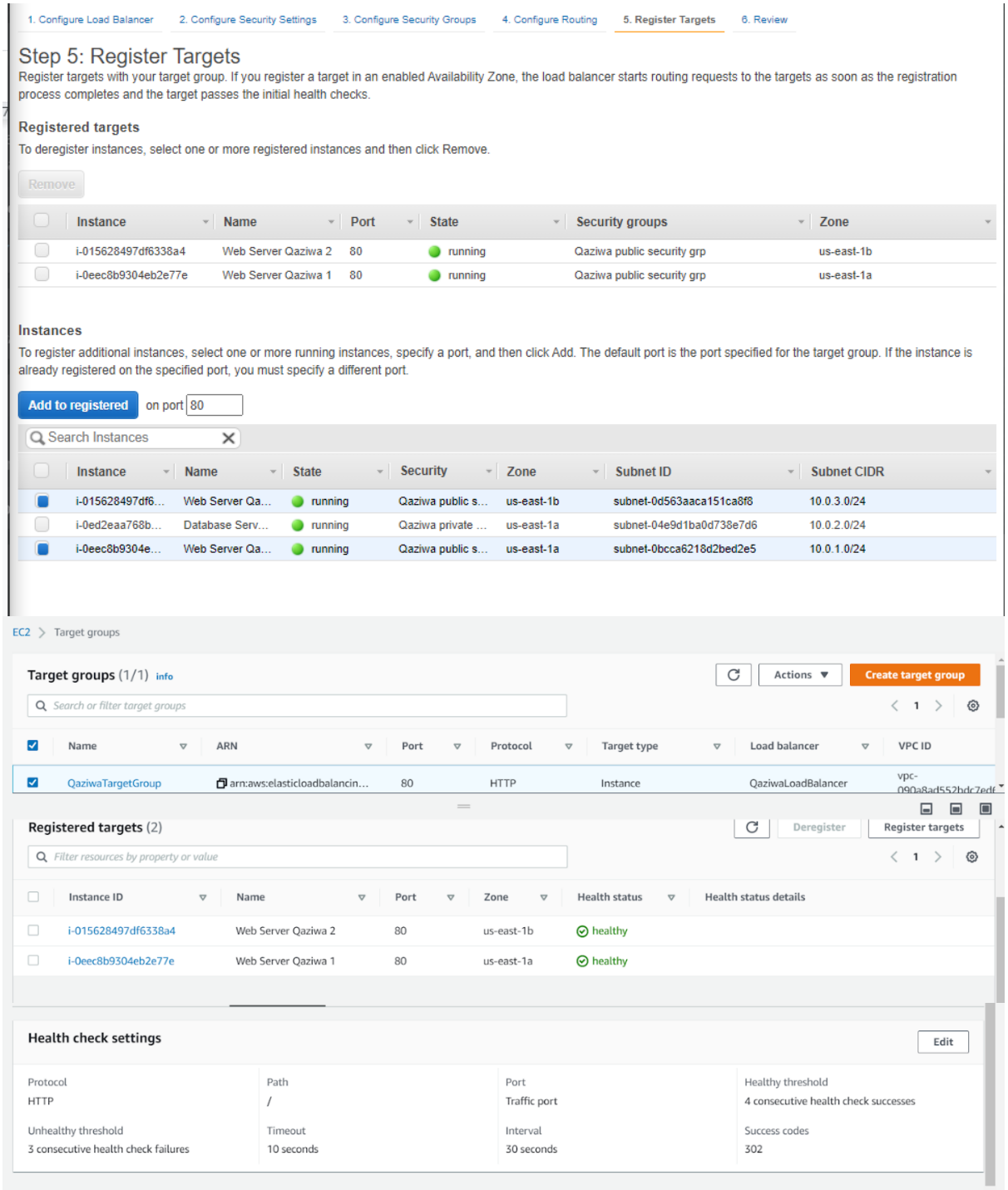


After this, create a new target group in which the instance will be inside this target group.

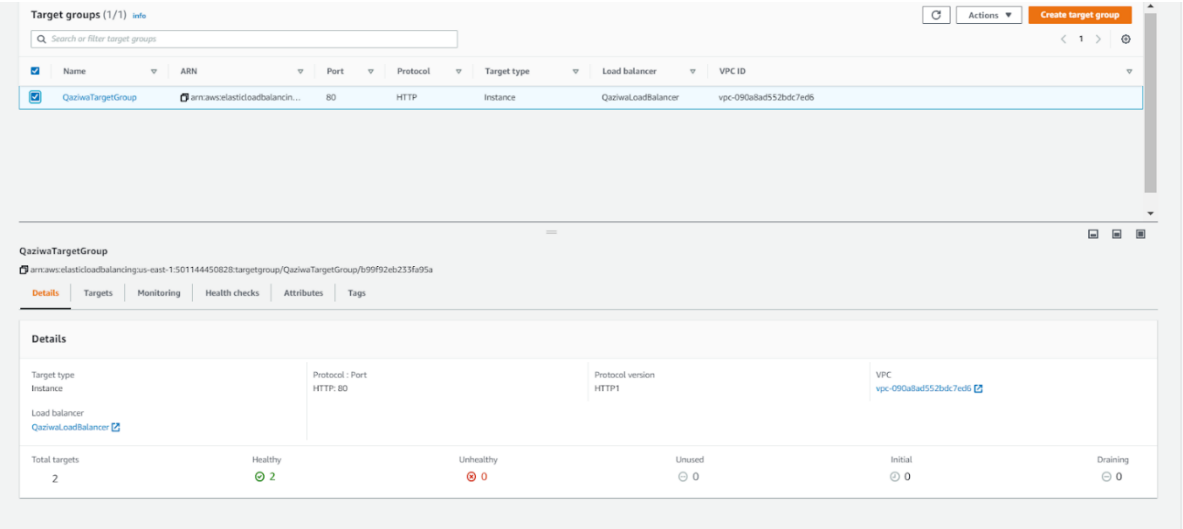
So, go to Configure Routing → Target group (new target group) → Name (QaziwaTargetGroup) → Target type (Instance) → Protocol (HTTP, port 80) then click next: register targets.



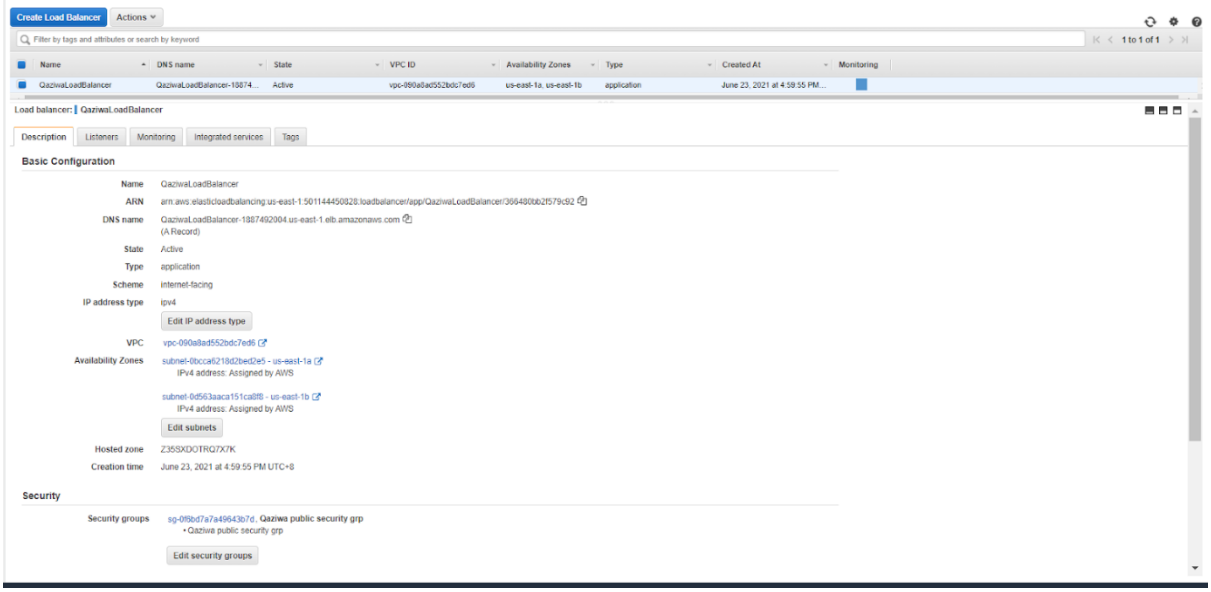
After this choose an instance, then add to registered on port 80 then next: review and then create



After you have created the target group, it should look like this:

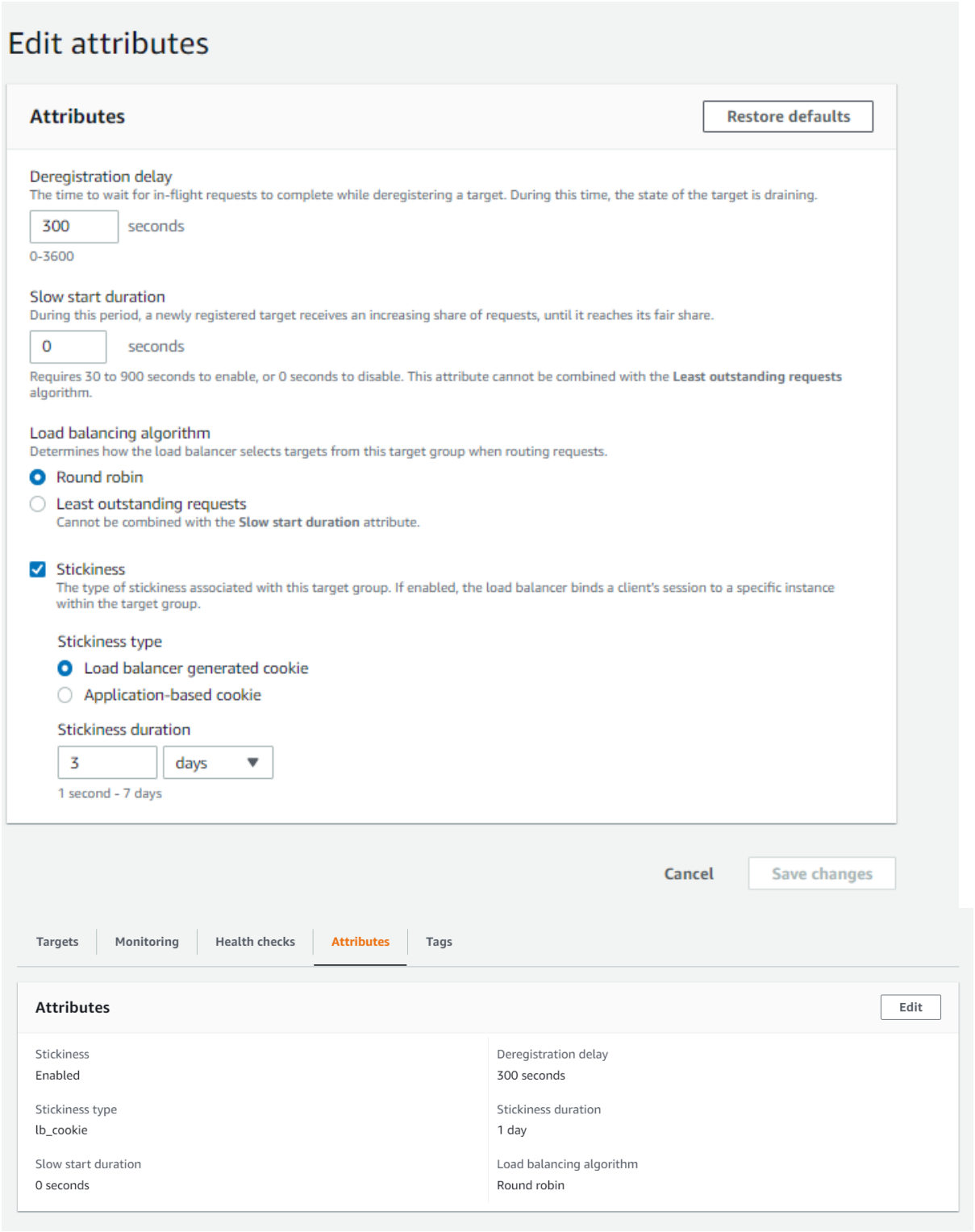


And the load balancer should look like this:



After configuring, we need to enable stickiness so that the session of the user on the instance will be consistent.

Go to target group > select the target group name > click on actions > edit attributes > check on the stickiness section



Testing of the Load Balancer DNS to see if it appear the Qaziwa Website:

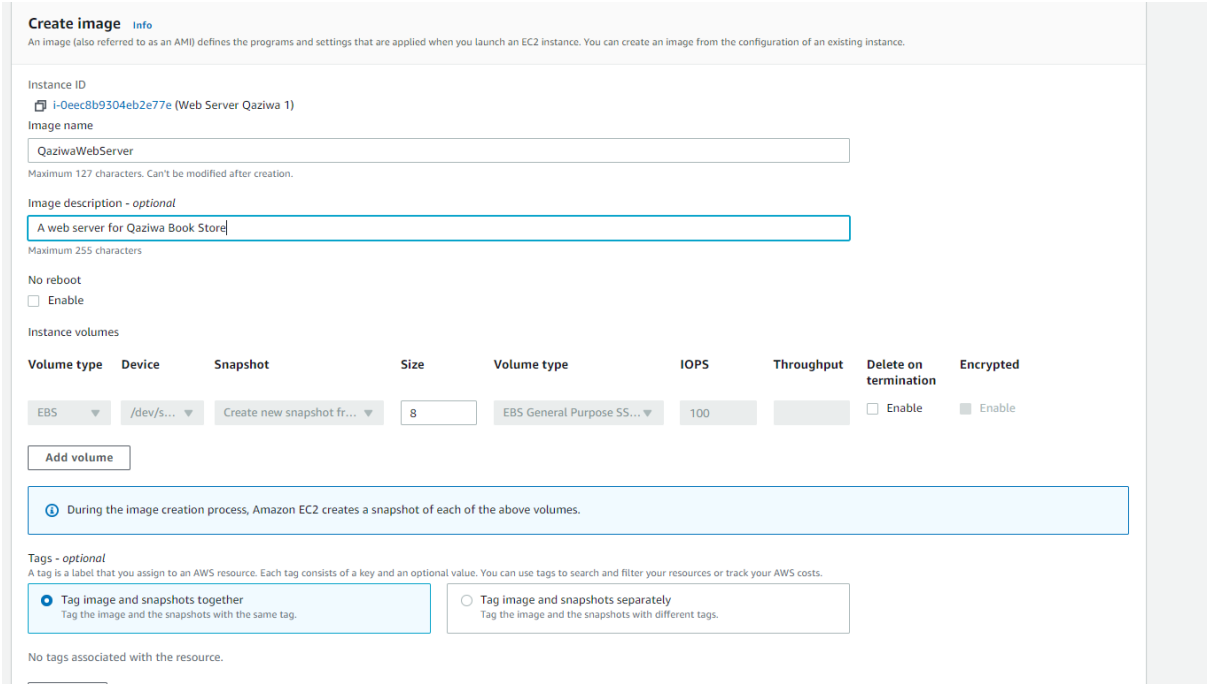
Once we have done and tested that we are able to access the website, now is time to implement auto-scaling which helps to provide scalability and availability if there 1 or more instances are down. It can also control the number of instances created based on the traffic intensity, so it will suit best based on the traffic flow during the period of time.

Creation of Auto Scaling:

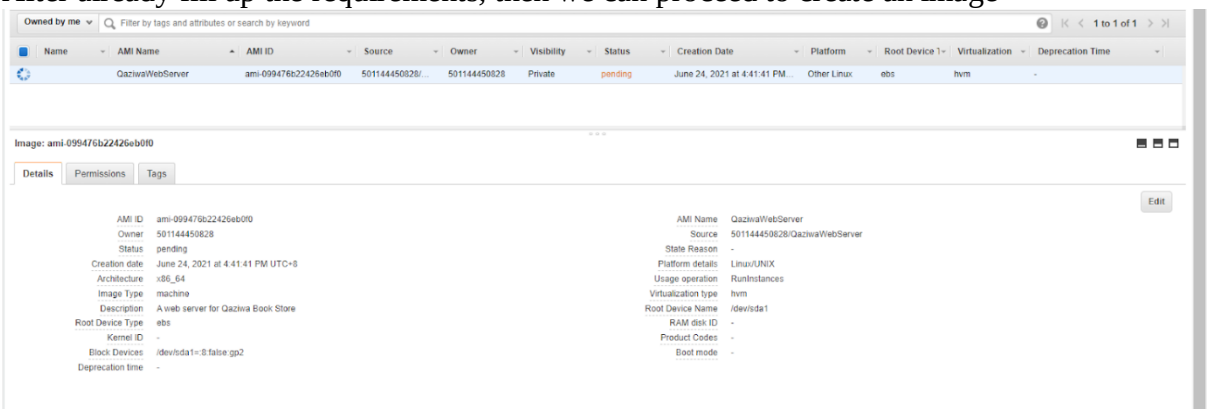
This function helps to ensure that there are steady resources available to every group at all times regardless of user demands. It works by monitoring the application and automatically scales the groups of different resources by adding or removing required capacity as the demand changes in real-time. This is part of ensuring maximum effectiveness within the application where high-demand groups have the necessary capacity to maintain the quality of service where low-demand groups will have capacity reduced for prevention of overspending.

There are various benefits of using auto-scaling in our application. A benefit is having better fault tolerance which will detect an unhealthy instance. This instance is then terminated and replaced with a new healthy instance. Knowing that auto-scaling ensures steadiness of resources, it has better availability to ensure all applications have the suitable capacity in handling the current traffic demand. In our application, auto-scaling is implemented to bridge 2 web server instances in 2 different public subnets from different availability zones. During our configuration, we will need to enable monitoring for auto-scaling to work. The main server is “Web Server Qaziwa 1” where all users will be directed to this server. Once the memory is being fully utilized (1 GiB) from the user traffic, any new users connecting to the website will be directed to the secondary web server “Web Server Qaziwa 2”. This is a technique where users will not experience server crashes due to surge of demand.

Firstly, we need to create an AMI (Amazon Machine Images) before we can launch auto-scaling. To do that we need to go to EC2 > instances > select an instance (Web Server Qaziwa 1) > Click on actions > go to Image and templates > Create Image



After already fill up the requirements, then we can proceed to create an image



After it is active, we can proceed to create a launch configuration

In launch configuration,
Goto creates launch configuration > Name (QaziwaWebServerLaunchConfig) → AMI (QaziwaWebServer) → Instance type (t2.micro) → Monitoring (Enable EC2 monitoring) → Security group (Choose Qaziwa Public Security Group) → Choose our existing key (QaziwaKey).

EC2 > Launch configurations > Create launch configuration

Create launch configuration Info

Launch configuration name

Name

QaziwaWebServerLaunchConfig

Amazon machine image (AMI) Info

AMI

QaziwaWebServer

Instance type Info

Instance type

t2.micro (1 vCPUs, 1 GiB, EBS Only) Choose instance type

Additional configuration - *optional*

Purchasing option Info

☐ Request Spot Instances

IAM instance profile Info

Select IAM role

Monitoring Info

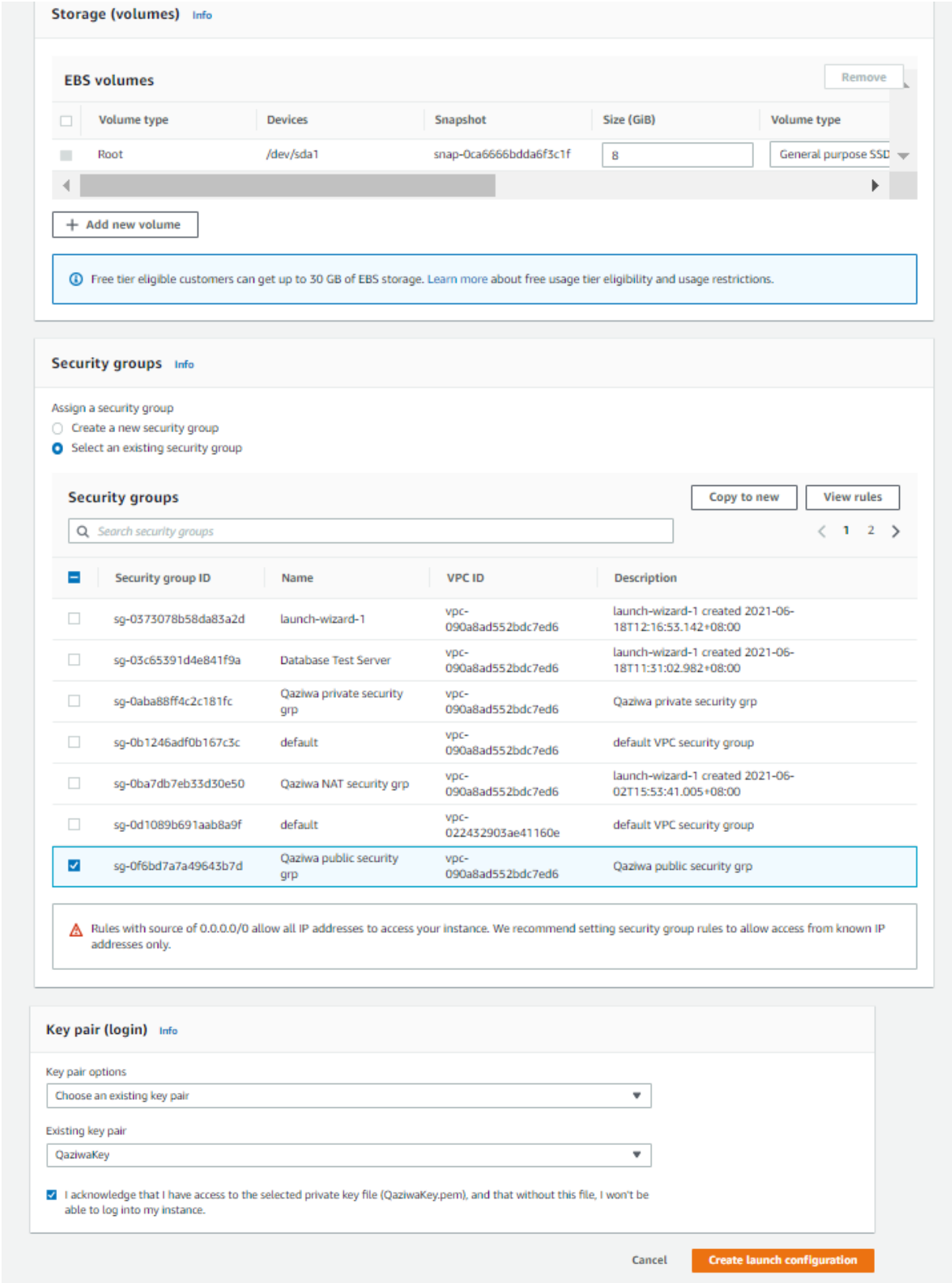
☑ Enable EC2 instance detailed monitoring within CloudWatch

EBS-optimized instance

☐ Launch as EBS-optimized instance

▶ Advanced details

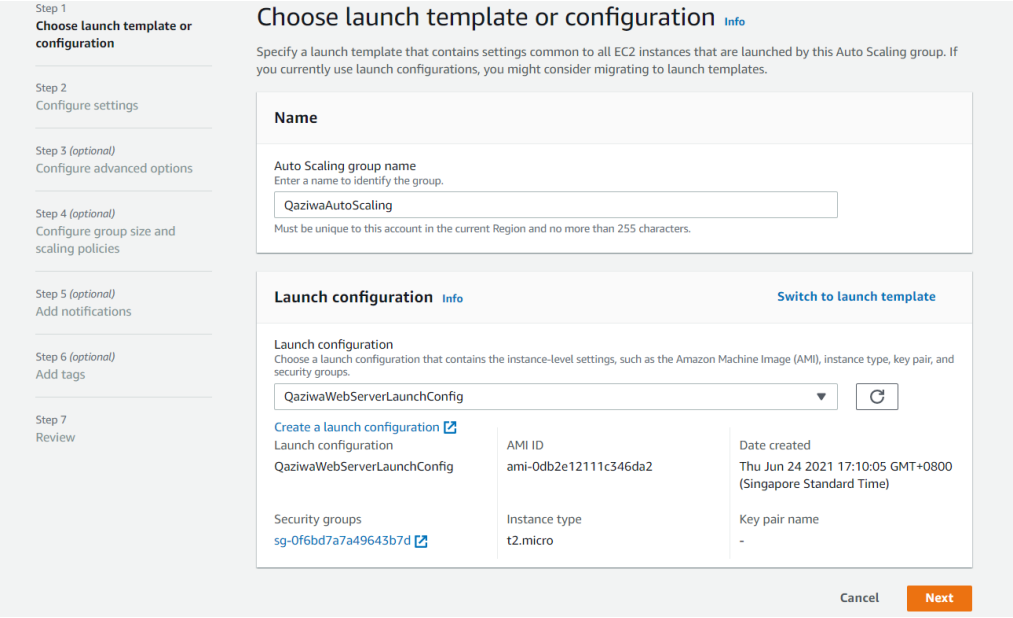
ⓘ Later, if you want to use a different launch configuration, you can create a new one and apply it to any Auto Scaling group. Existing launch configurations cannot be edited.



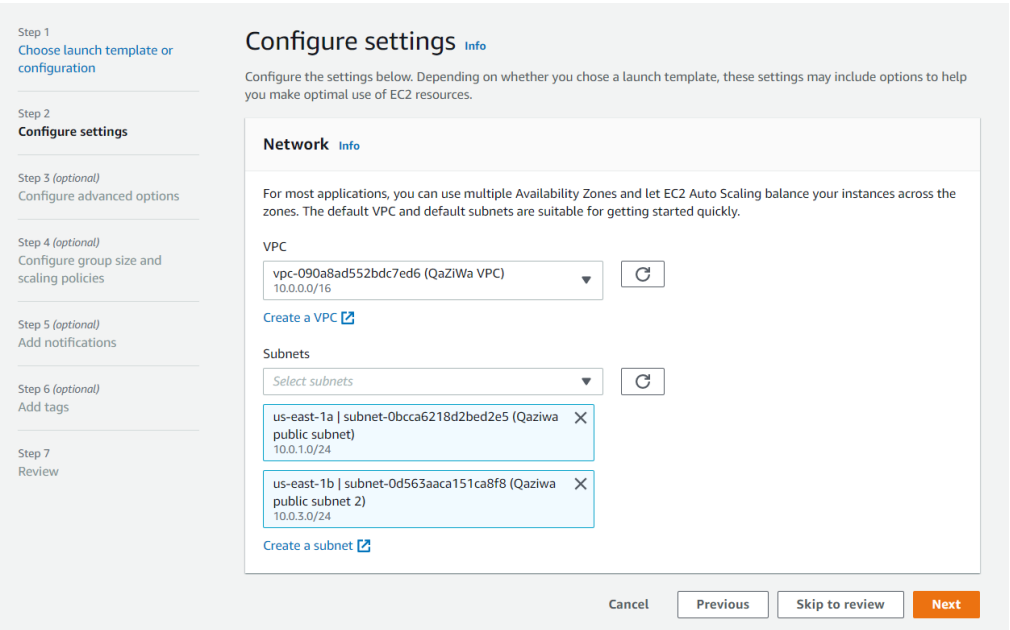
After the configuration, then click on “create launch configuration”.

Now is the time to configure autoscaling

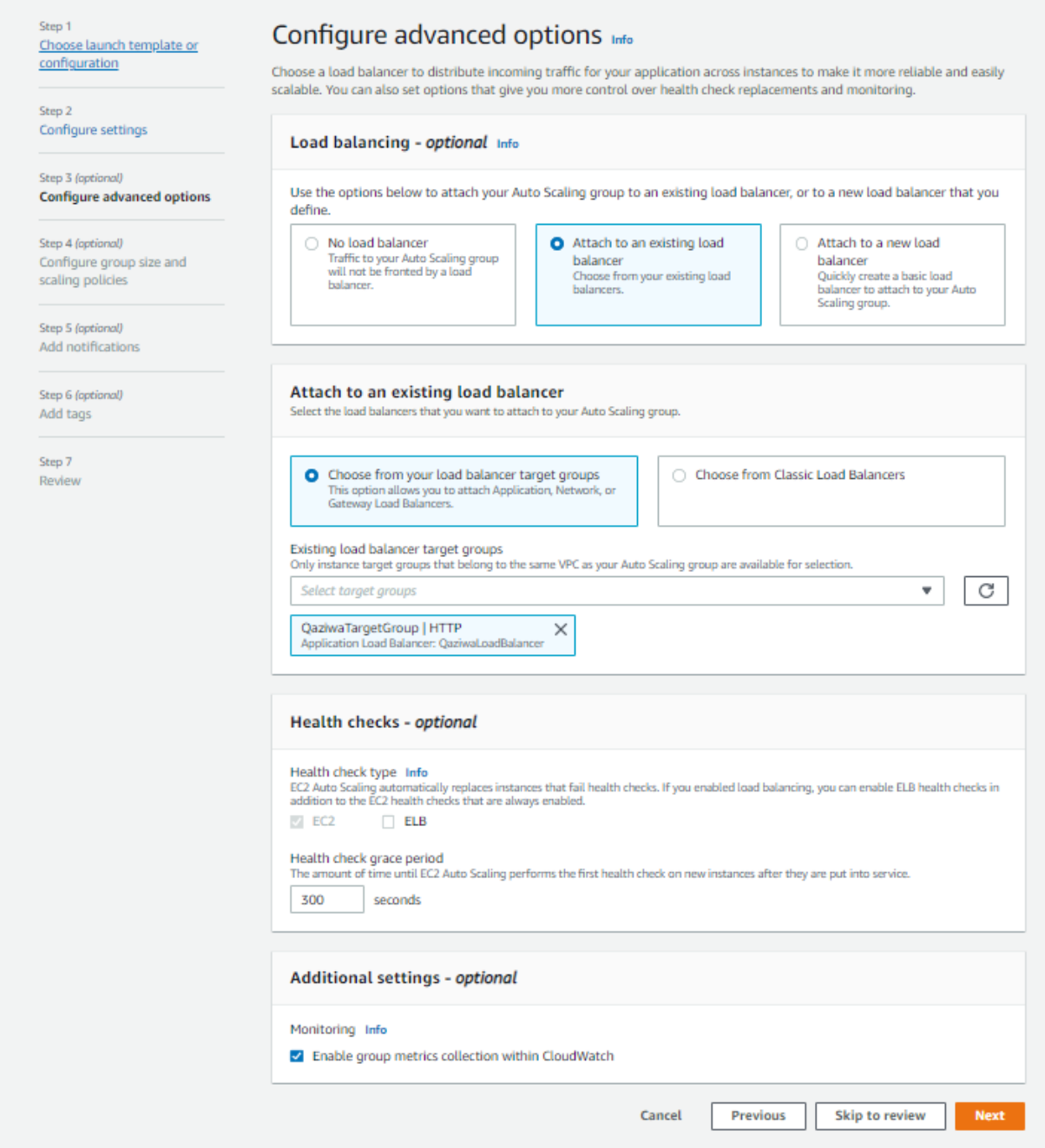
Goto Autoscaling > Create Auto Scaling Group > Auto Scaling Name (QaziwaAutoScaling) → Launch Configuration (QaziwaWebServerLaunchConfig) [taken from Launch Configuration Group > click next



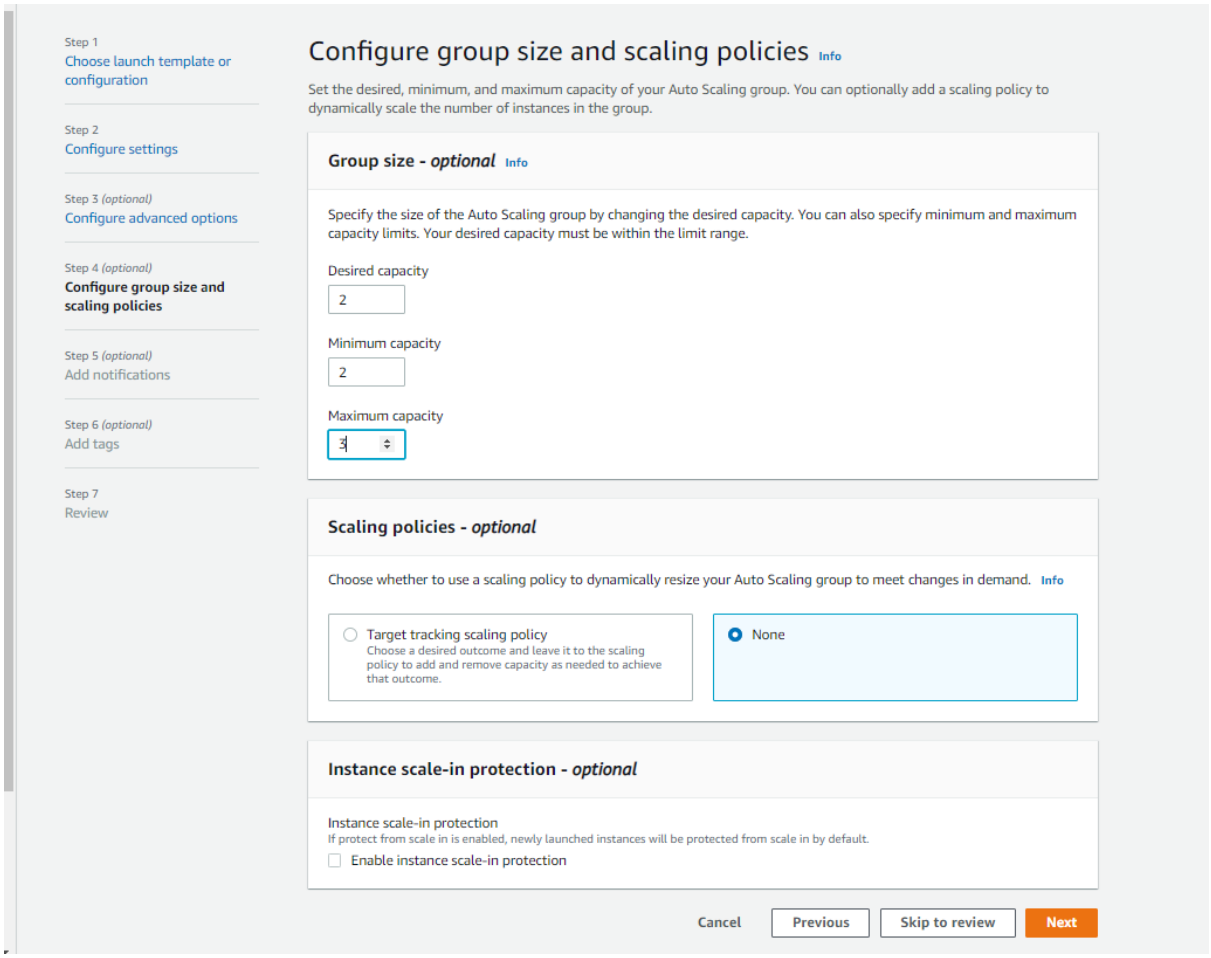
In the configure Setting → Select a VPC (Qaziwa VPC) → Subnets (Public Subnet 1 & 2) > click next



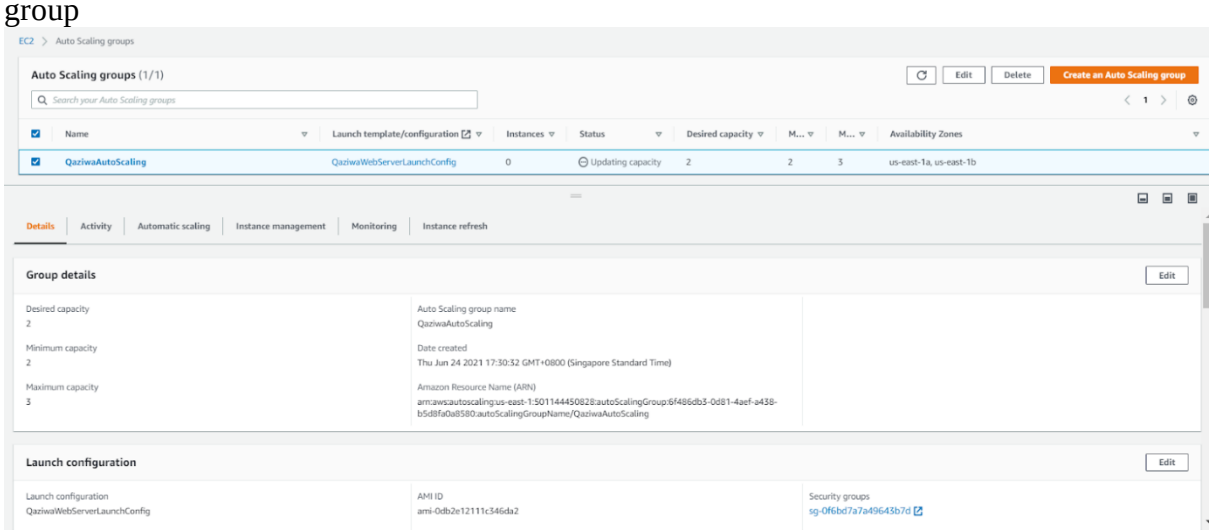
In the configure advanced options, Click on Attach to an existing load balancer → choose from your load balancer target group → choose the existing load balancer target group (QaziwaTargetGroup) → in the additional settings → Enable group metrics collection within cloudwatch > click next

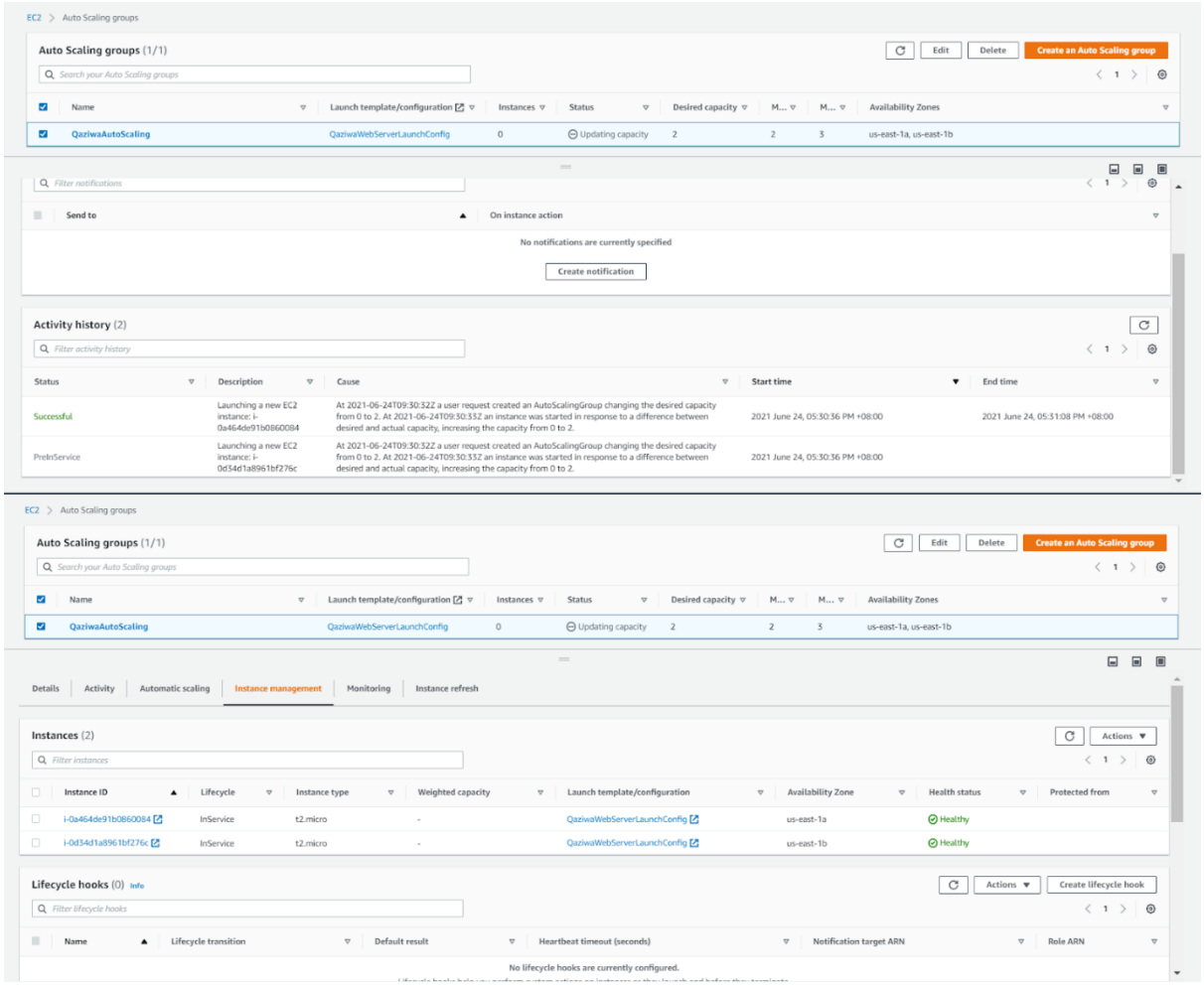


In the configure group size and scaling policies, fill it accordingly > click next



Click next for notifications and add tags and on the review page, click on create auto scaling group





The above shows that we have successfully created an Auto Scaling group.

Since all of the configurations are stable and able to access the website and provide scalability, functional, stress tests that meet our requirements, we then need to set up a domain name for our website.

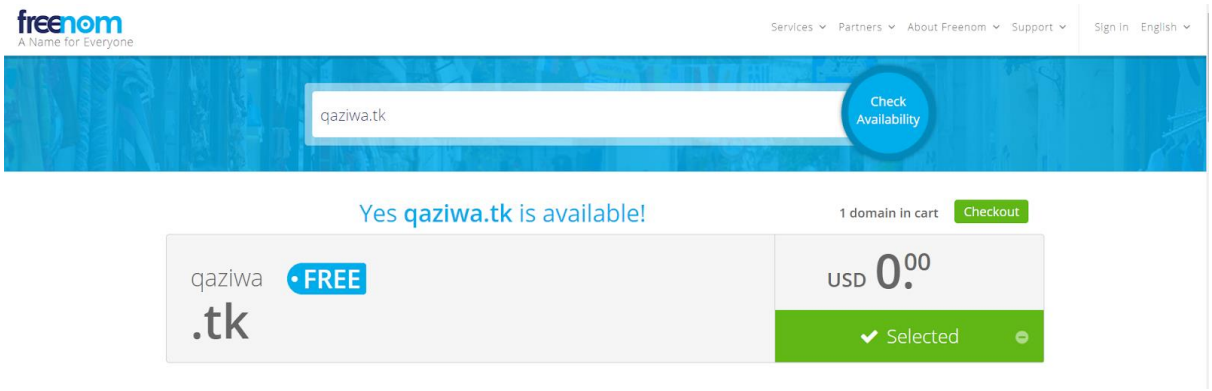
Creation of Domain Name:

Freenom Domain:

In normal real-life usage, websites have domain names for easy identification among normal users.

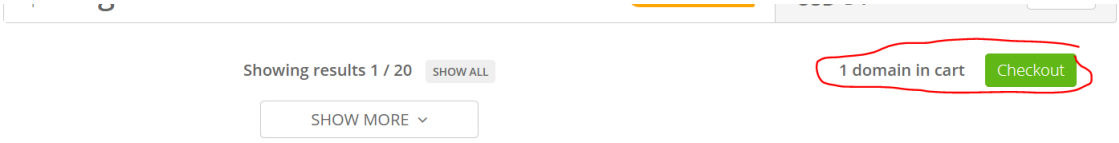
We will be using freenom which is a free DNS Website that enables us to get our own DNS free for 3 months. (https://www.freenom.com/en/index.html?lang=en)

To start off, we check if our preferred DNS (www.qaziwa.tk) is available.

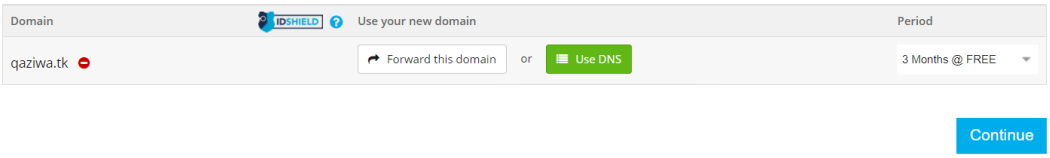


Yes, it is available and it is free, so now we are going to proceed to have this domain up and ready.

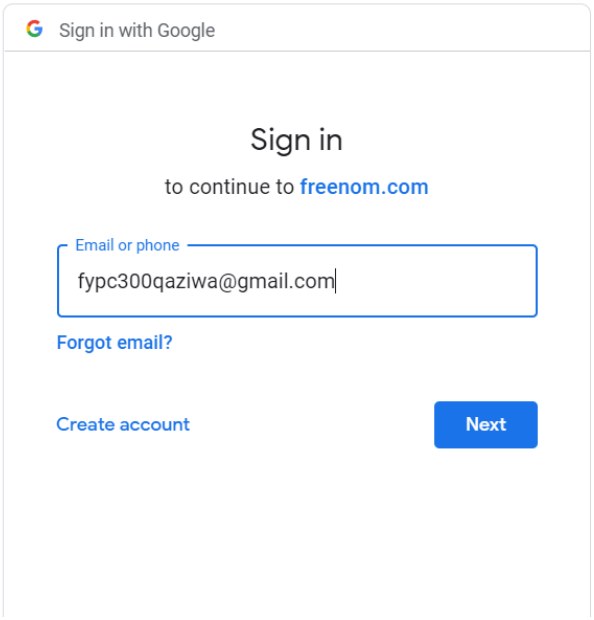
Click checkout



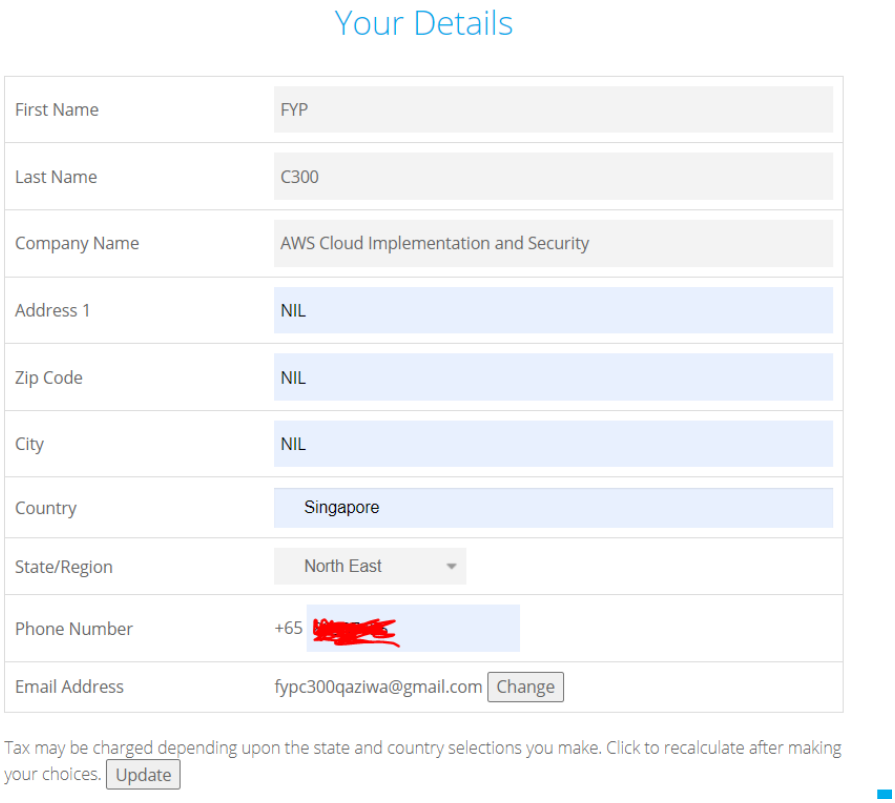
Click on Continue



Moving forward, we sign in using our FYP Gmail account and password

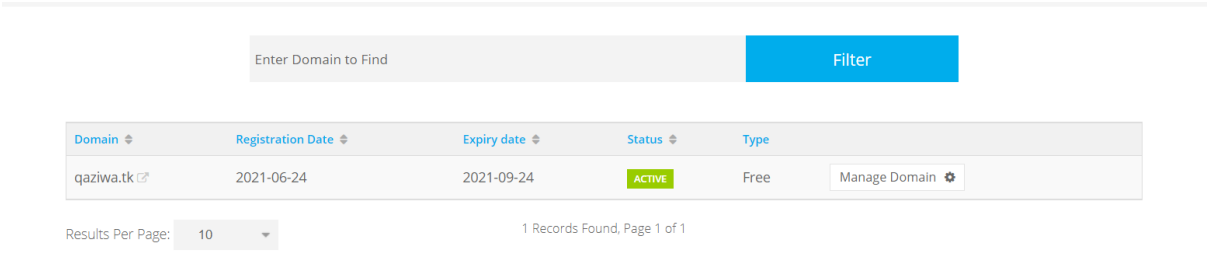


Then verify our email and then fill in our credentials



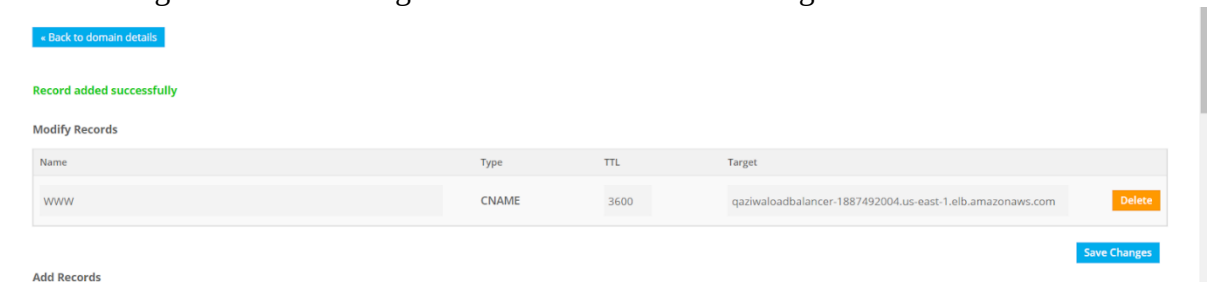
Then click checkout.

Now we have our domain up and ready

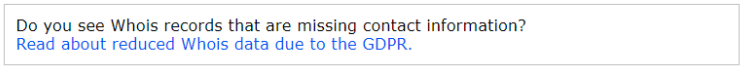


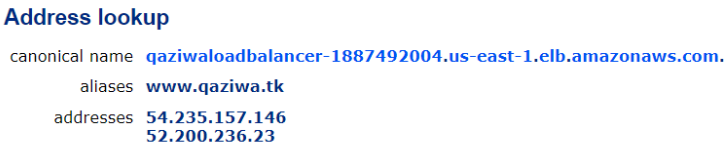
Then we need to configure our DNS record so that we are able to access the website using the load balancer DNS

Go to Manage domain > Manage freenom DNS > do the configuration shown below:



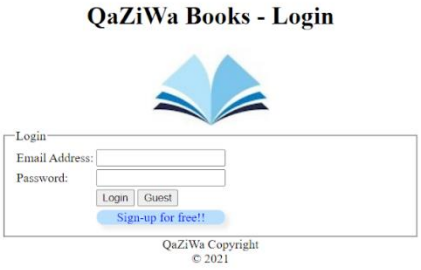
Now is to test and check





DNS records

name	class	type	data	time to live
www.qaziwa.tk	IN	CNAME	qaziwaloadbalancer-1887492004.us-east-1.elb.amazonaws.com	3600s (01:00:00)
qaziwaloadbalancer-1887492004.us-east-1.elb.amazonaws.com	IN	A	52.200.236.23	60s (00:01:00)
qaziwaloadbalancer-1887492004.us-east-1.elb.amazonaws.com	IN	A	54.235.157.146	60s (00:01:00)
us-east-1.elb.amazonaws.com	IN	NS	ns-1119.awsdns-11.org	1800s (00:30:00)
us-east-1.elb.amazonaws.com	IN	NS	ns-1793.awsdns-32.co.uk	1800s (00:30:00)
us-east-1.elb.amazonaws.com	IN	NS	ns-235.awsdns-29.com	1800s (00:30:00)
us-east-1.elb.amazonaws.com	IN	NS	ns-934.awsdns-52.net	1800s (00:30:00)
us-east-1.elb.amazonaws.com	IN	SOA	server: ns-1119.awsdns-11.org email: awsdns-hostmaster@amazon.com serial: 1 refresh: 7200 retry: 900 expire: 1209600 minimum ttl: 60	60s (00:01:00)
qaziwa.tk	IN	SOA	server: ns01.freenom.com email: soa@freenom.com serial: 1624523411 refresh: 10800 retry: 3600 expire: 604800 minimum ttl: 3600	300s (00:05:00)
qaziwa.tk	IN	NS	ns02.freenom.com	300s (00:05:00)
qaziwa.tk	IN	NS	ns03.freenom.com	300s (00:05:00)
qaziwa.tk	IN	NS	ns01.freenom.com	300s (00:05:00)
qaziwa.tk	IN	NS	ns04.freenom.com	300s (00:05:00)
146.157.235.54.in-addr.arpa	IN	PTR	ec2-54-235-157-146.compute-1.amazonaws.com	300s (00:05:00)
157.235.54.in-addr.arpa	IN	SOA	server: dns-external-master.amazon.com email: root@amazon.com serial: 14 refresh: 3600 retry: 900 expire: 604800 minimum ttl: 900	900s (00:15:00)



So now, after waiting for a few minutes, we can see that our Domain is up and running. As shown above, we can see that the connection is not secure, therefore more research is needed to ensure that we have a secure SSL Certificate and connection from the client to the load balancer.

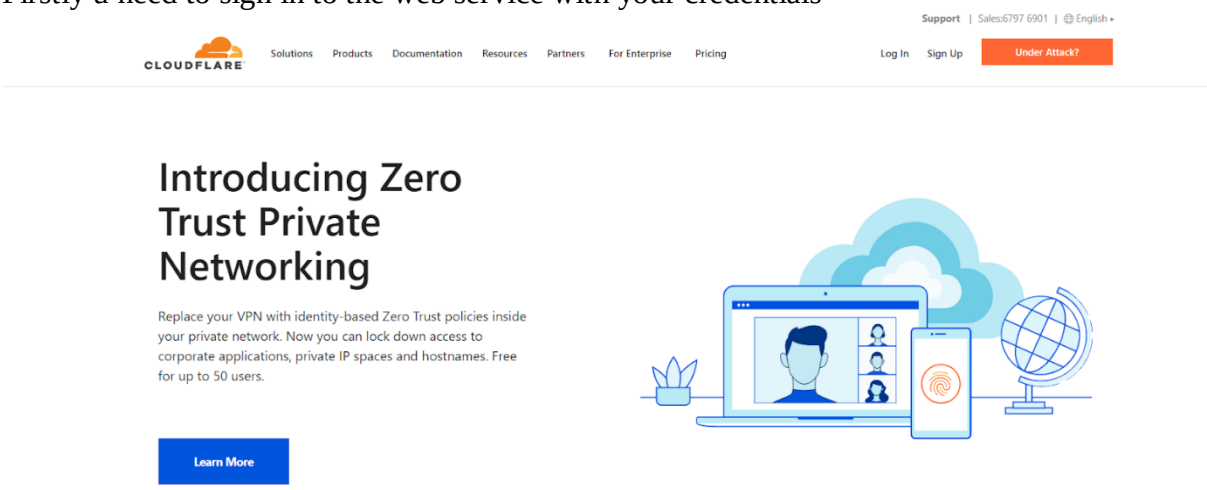
Creation of CloudFlare:

Cloudflare SSL Certificate Configuration

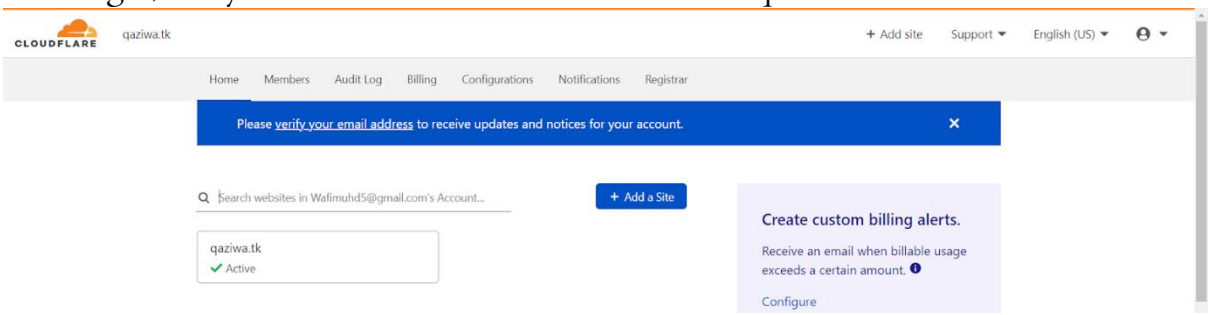
It is important to ensure that having an SSL Certificate to be implemented on the website. SSL Certificate is a type of cryptography which is also known as a digital certificate that enables electronic documents such as websites to be encrypted when transmitting its data. This means all of the private information such as user information will be encrypted and this causes the information not to be in plain text. As such, implementing an SSL Certificate will

provide good data in transmit encryption to the AWS load balancer. For the SSL Certificate, we will be using a CloudFlare website, which is a free security website that provides multiple features such as DNS, Firewall, Caching, and most importantly SSL/TLS encryption. The link to the website (https://www.cloudflare.com)

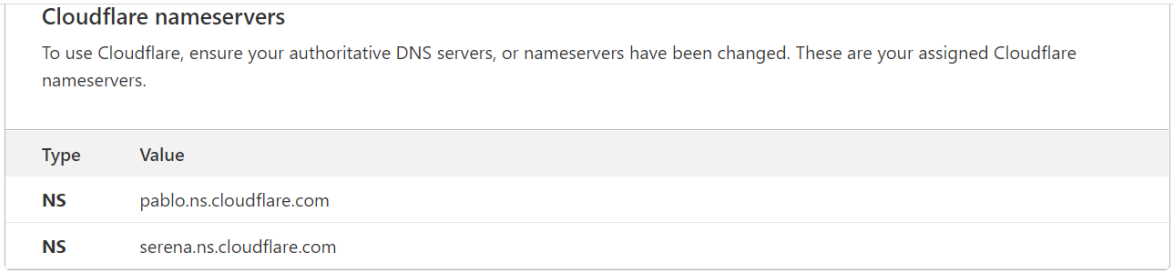
Firstly u need to sign in to the web service with your credentials



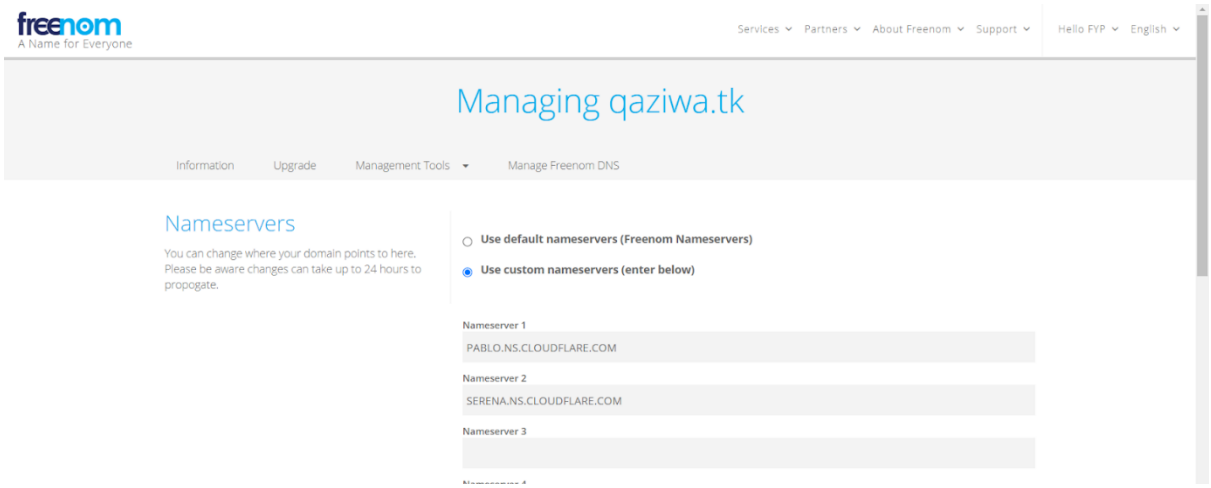
After login, add your DNS into the add a site section → qaziwa.tk



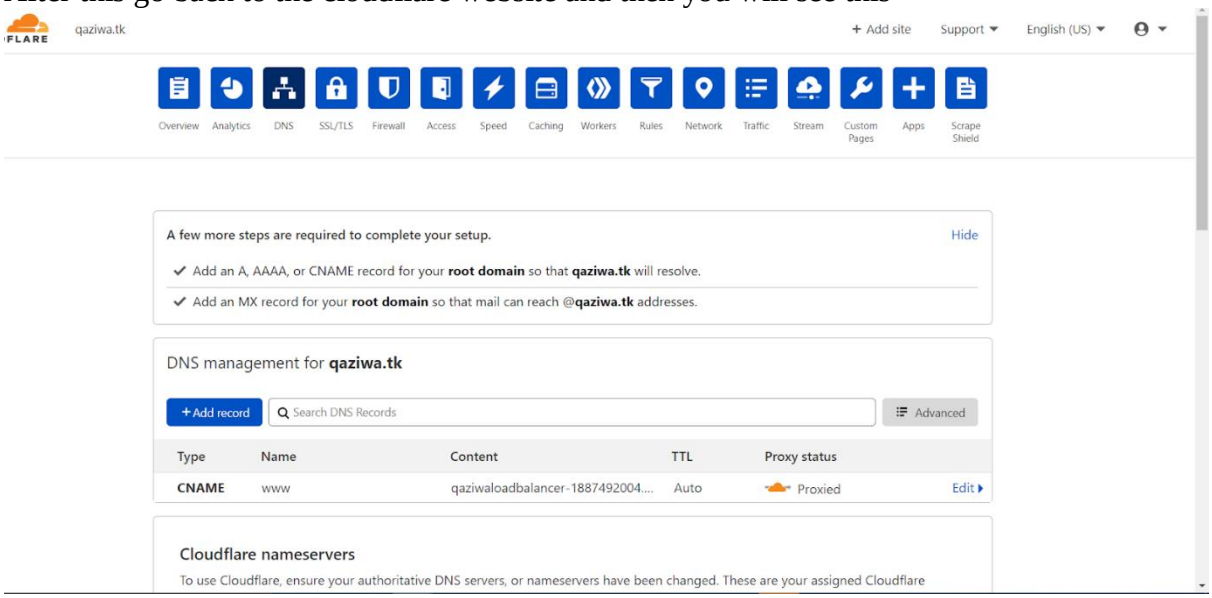
After adding our DNS to the Cloudflare website, then it will show Cloudflare nameservers



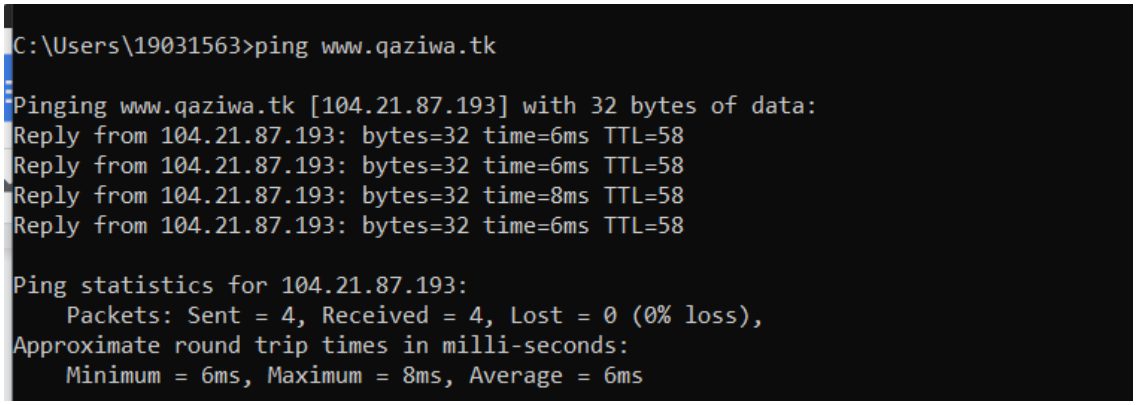
Replace the name server from the freenom to the nameserver of the Cloudflare



After this go back to the cloudflare website and then you will see this



I have enabled proxied, in which it will hide the public IP address of the load balancer and instead it will display the cloudflare IP address which hides the true IP address of the Load Balancer.



```
C:\Users\19031563>nslookup www.qaziwa.tk
Server:  router.asus.com
Address:  192.168.50.1

Non-authoritative answer:
Name:    www.qaziwa.tk
Addresses:  2606:4700:3035::6815:57c1
          2606:4700:3035::ac43:91b4
          104.21.87.193
          172.67.145.180


C:\Users\19031563>
```

Domain Dossier Investigate domains and IP addresses

domain or IP address: www.qaziwa.tk

☑ domain whois record ☑ DNS records ☐ traceroute

☑ network whois record ☑ service scan go

user: anonymous [180.129.48.35]
balance: 48 units
log in | account info

CentralOps.net

Do you see Whois records that are missing contact information?
Read about reduced Whois data due to the GDPR.

Address lookup

canonical name **www.qaziwa.tk.**

aliases

addresses **104.21.87.193**
172.67.145.180
2606:4700:3035::ac43:91b4
2606:4700:3035::6815:57c1

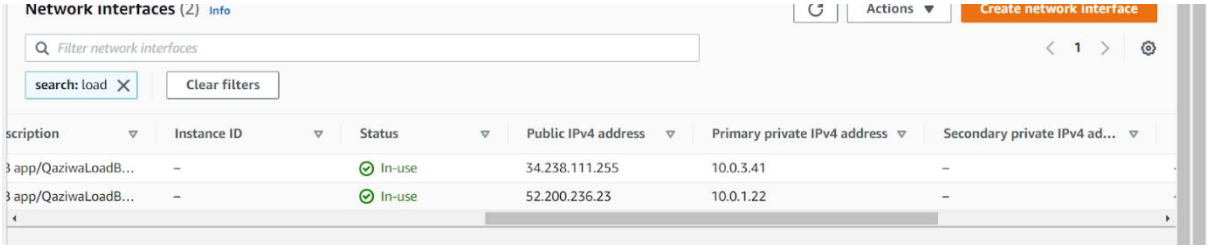
DNS records

name	class	type	data	time to live
www.qaziwa.tk	IN	HINFO	CPU: RFC8482 OS:	3789s (01:03:09)
qaziwa.tk	IN	HINFO	CPU: RFC8482 OS:	3789s (01:03:09)
qaziwa.tk	IN	NS	pablo.ns.cloudflare.com	86400s (1.00:00:00)
qaziwa.tk	IN	NS	serena.ns.cloudflare.com	86400s (1.00:00:00)
193.87.21.104.in-addr.arpa	IN	HINFO	CPU: RFC8482 OS:	3789s (01:03:09)
21.104.in-addr.arpa	IN	NS	cruz.ns.cloudflare.com	53281s (14:48:01)
21.104.in-addr.arpa	IN	NS	kevin.ns.cloudflare.com	53281s (14:48:01)
4.b.1.9.3.4.c.a.0.0.0.0.0.0.0.0.0.0.0.0.5.3.0.3.0.0.7.4.6.0.6.2.ip6.arpa	IN	HINFO	CPU: RFC8482 OS:	3789s (01:03:09)
0.0.7.4.6.0.6.2.ip6.arpa	IN	NS	leah.ns.cloudflare.com	54683s (15:11:23)
0.0.7.4.6.0.6.2.ip6.arpa	IN	NS	micah.ns.cloudflare.com	54683s (15:11:23)

Service scan

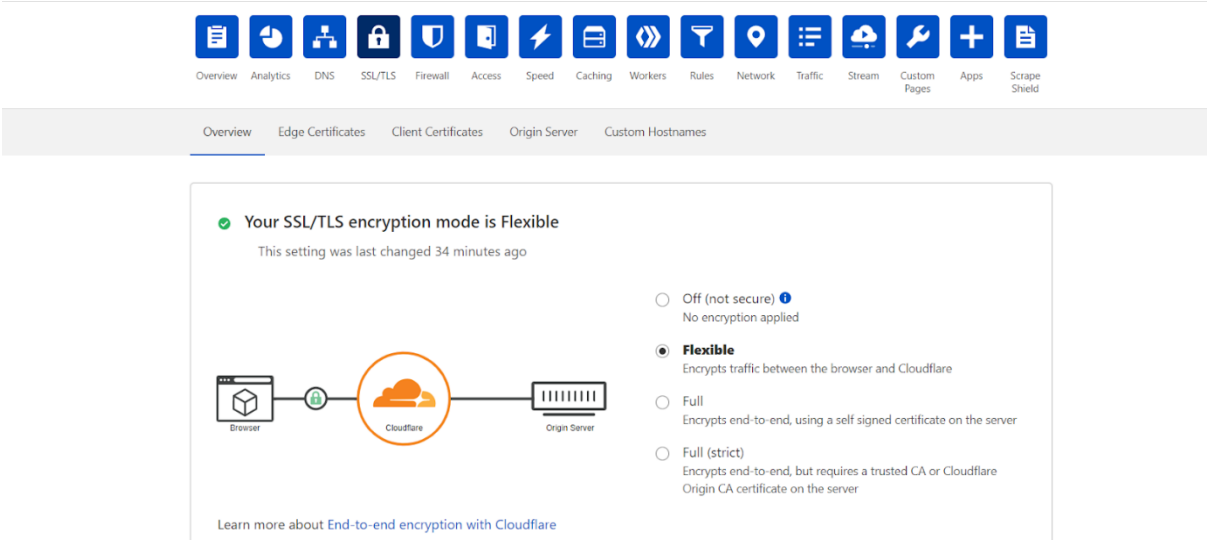
FTP - 21 Error: TimedOut
SMTP - 25 Error: TimedOut

And we tally with the AWS Load Balancer Public IP address it shows

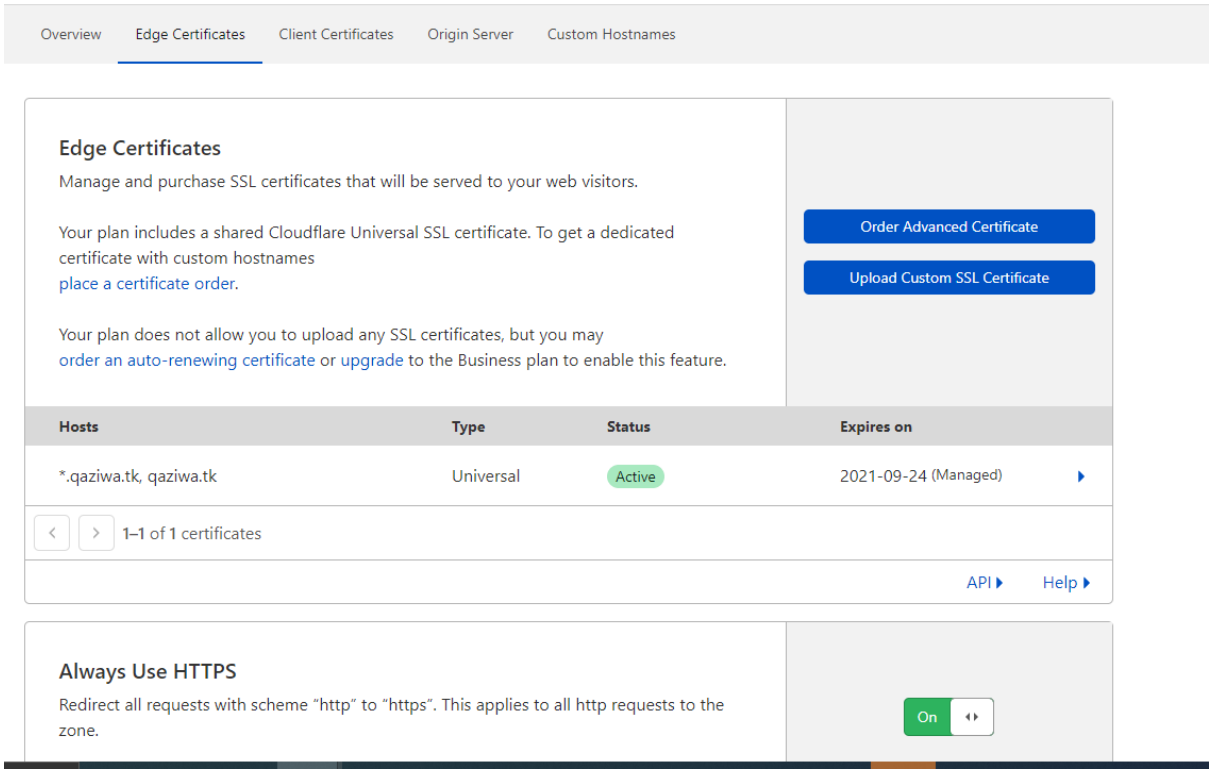


Therefore, we can conclude that it hides the IP address of the actual load balancer IP address and therefore, users cannot find the actual load balancer IP address.

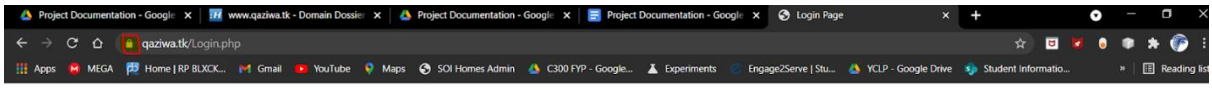
Moving forward, now we want to enable SSL Certificate, go to SSL/TLS > check on flexible → which enable it to encrypt all of the data from user or browser to cloudflare.

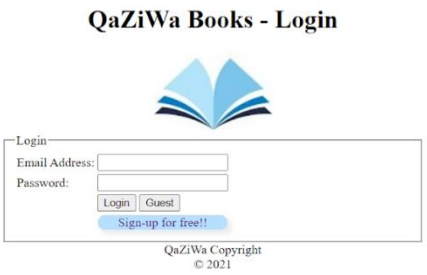


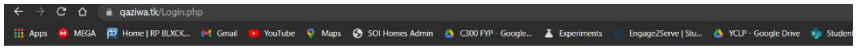
Due to load balancer limitation and less privilege/not authorised to do an SSL Certificate, therefore we cannot import a certificate that is provided by cloudflare to the load balancer. After this setting now we want to tick on the Always Use HTTPS section shown below.

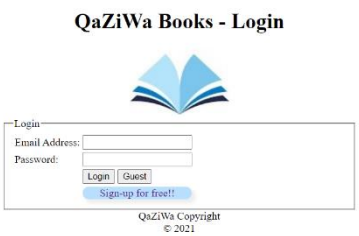


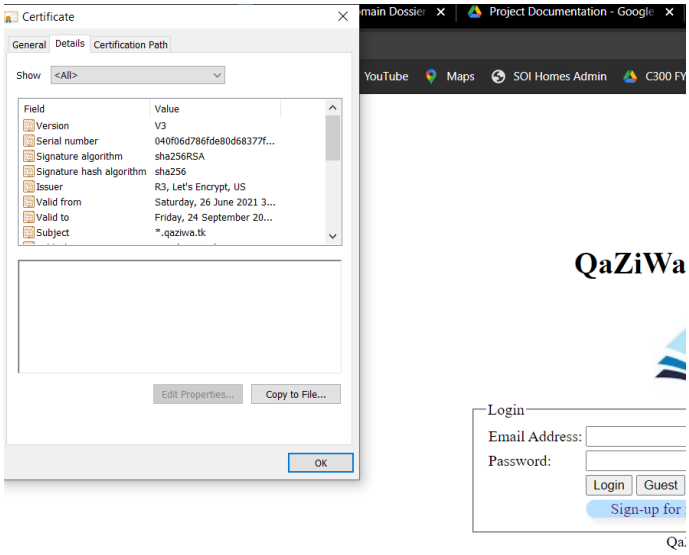
After all of this, we wait for a moment, and then we can see that our new website is already used HTTPS website and not HTTP

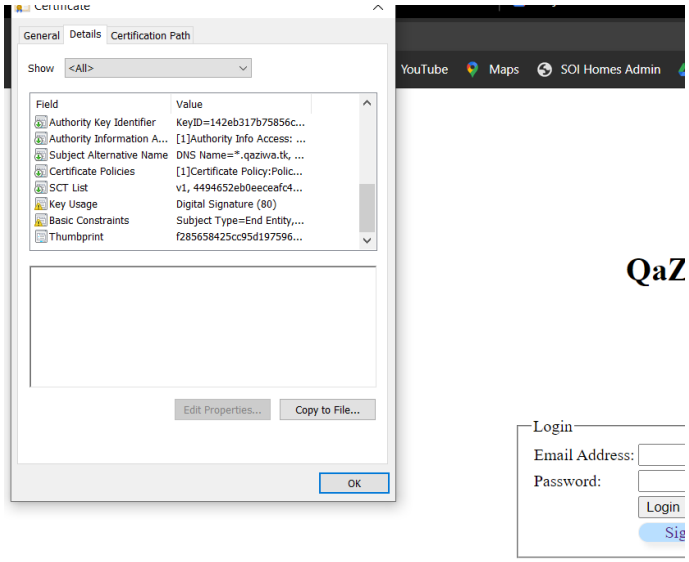










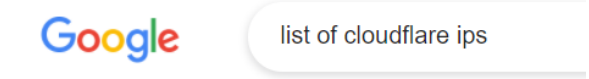


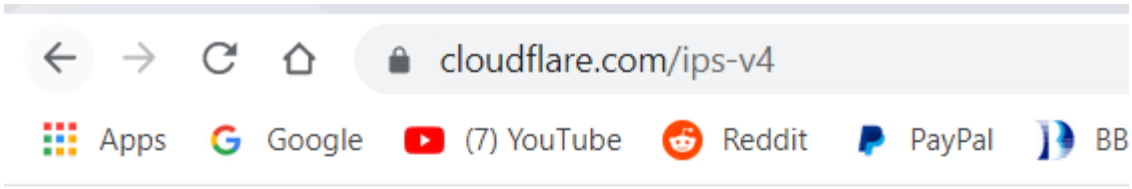
Now we have fully secured the website connection from the user to the cloudflare web service.

<u>Allowing Load Balancer to accept traffic from Cloudflare:</u>

However, there is an issue doing this. If the user finds out the ALB DNS name, he or she can access the load balancer directly in which it does not pass through cloudflare and this can cause a security risk as an attacker can do a DoS attack on our load balancer. To prevent this, we created a security group specifically on access to the Cloudflare IP address.

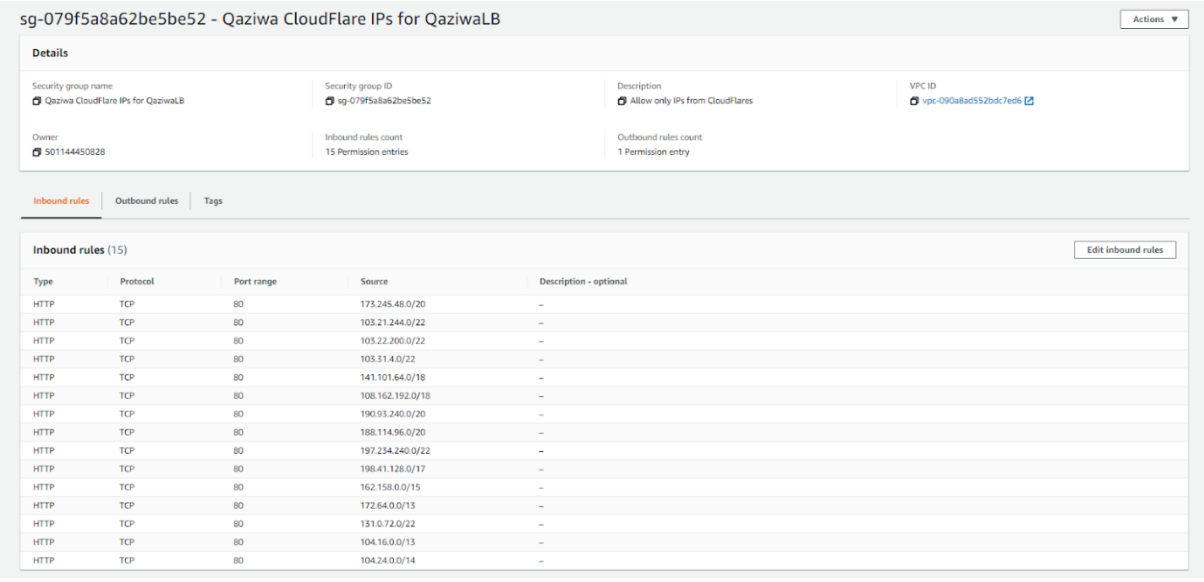
Below is a security group for the CloudFlare IP addresses.






```
173.245.48.0/20
103.21.244.0/22
103.22.200.0/22
103.31.4.0/22
141.101.64.0/18
108.162.192.0/18
190.93.240.0/20
188.114.96.0/20
197.234.240.0/22
198.41.128.0/17
162.158.0.0/15
172.64.0.0/13
131.0.72.0/22
104.16.0.0/13
104.24.0.0/14
```

The above is the IP address of the cloudflare, so this means any of this IP must be added to the Load Balancer Security Group. So since we already know the security group of the cloudflare, now we need to create a security group to allow only these IP addresses from the cloudflare.



Inbound rules (15) — Edit inbound rules

Type	Protocol	Port range	Source	Description - optional
HTTP	TCP	80	173.245.48.0/20	–
HTTP	TCP	80	103.21.244.0/22	–
HTTP	TCP	80	103.22.200.0/22	–
HTTP	TCP	80	103.31.4.0/22	–
HTTP	TCP	80	141.101.64.0/18	–
HTTP	TCP	80	108.162.192.0/18	–
HTTP	TCP	80	190.93.240.0/20	–
HTTP	TCP	80	188.114.96.0/20	–
HTTP	TCP	80	197.234.240.0/22	–
HTTP	TCP	80	198.41.128.0/17	–
HTTP	TCP	80	162.158.0.0/15	–
HTTP	TCP	80	172.64.0.0/13	–
HTTP	TCP	80	131.0.72.0/22	–
HTTP	TCP	80	104.16.0.0/13	–
HTTP	TCP	80	104.24.0.0/14	–

Above, it shows that we have already created a security group that only allows IP addresses from the selected source IP addresses with port 80 (HTTP). This is by, we want to access a website, hence port 80 is being selected. After the creation of the security group and assigned the specific IP addresses. Now we need to integrate it into our LB. To do this we need to go to Load Balancer, select Qazia LoadBalancer then click on actions, and edit the security groups that we have created.

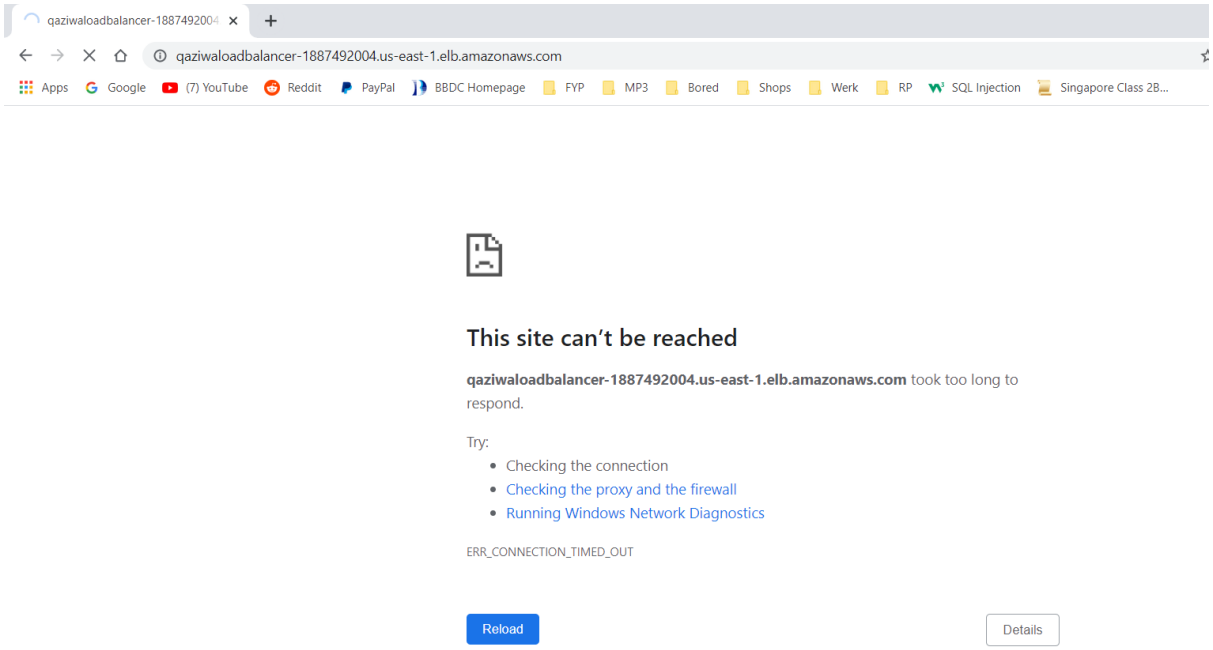
Edit security groups

Select security groups to associate with your load balancer.

	Security group ID	Name	Description
☐	sg-03c65391d4e84…	Database Test Server	launch-wizard-1 created 2021-06-18T11:31:02.982+08:00
☑	sg-079f5a8a62be5b…	Qaziwa CloudFlare …	Allow only IPs from CloudFlares
☐	sg-0ba7db7eb33d3…	Qaziwa NAT securit…	launch-wizard-1 created 2021-06-02T15:53:41.005+08:00
☐	sg-0aba88ff4c2c181fc	Qaziwa private sec…	Qaziwa private security grp
☐	sg-0f6bd7a7a49643…	Qaziwa public secur…	Qaziwa public security grp
☐	sg-0b1246adf0b167…	default	default VPC security group

Cancel Save

After we have selected, then we click Save, then wait for a few seconds for it to take effect. Now we need to test if we are still able to access the Load Balancer DNS.

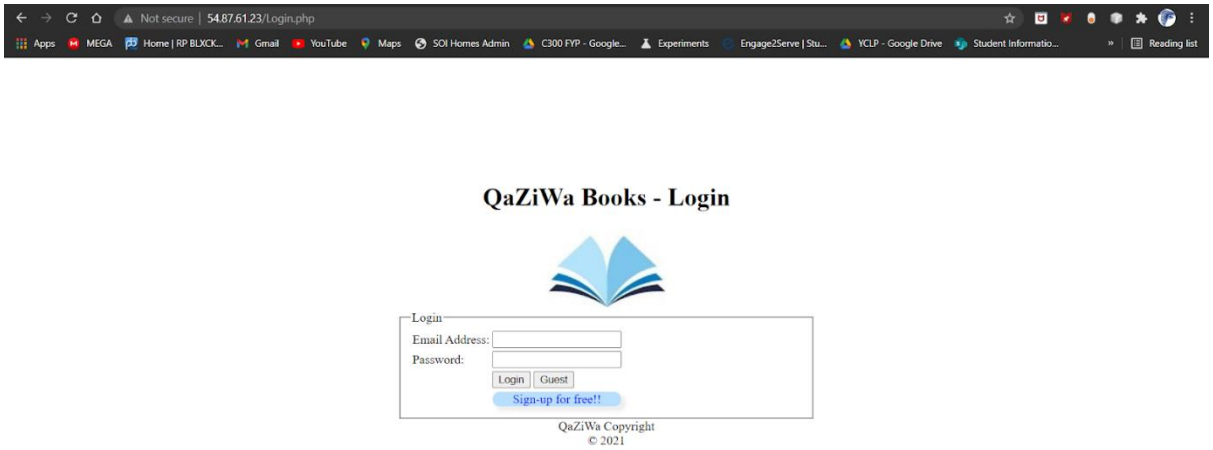


As shown above, we can see that we are unable to access the load balancer DNS, so now we test it to access the domain name that we have assigned to it which is www.qaziwa.tk

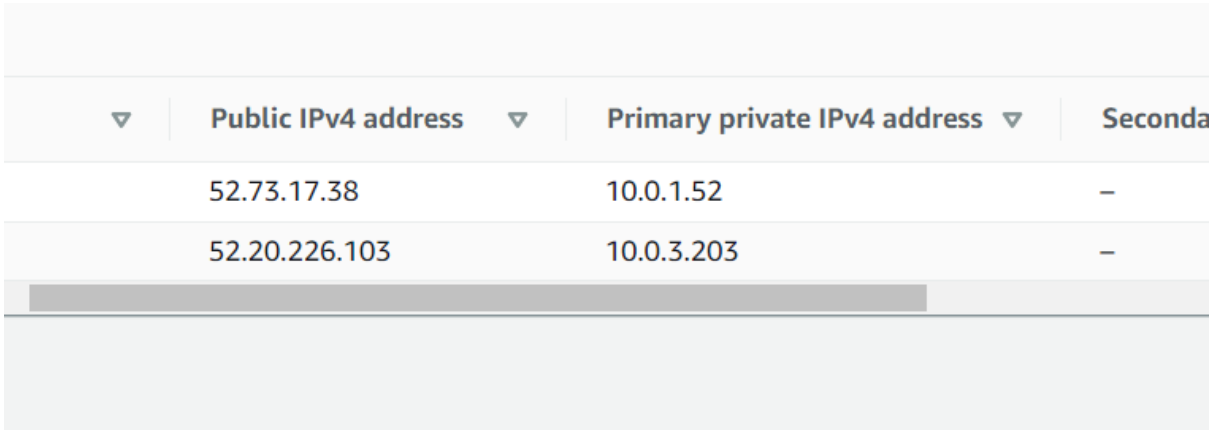
As shown above, we can see that we are able to access the website directly from the client to the cloudflare, but not from the client to the load balancer. This is an added feature to ensure that users are not able to access the load balancer directly and they must go through the cloudflare in order to access the website.

Allowing Instance to accept traffic from Load Balancer:

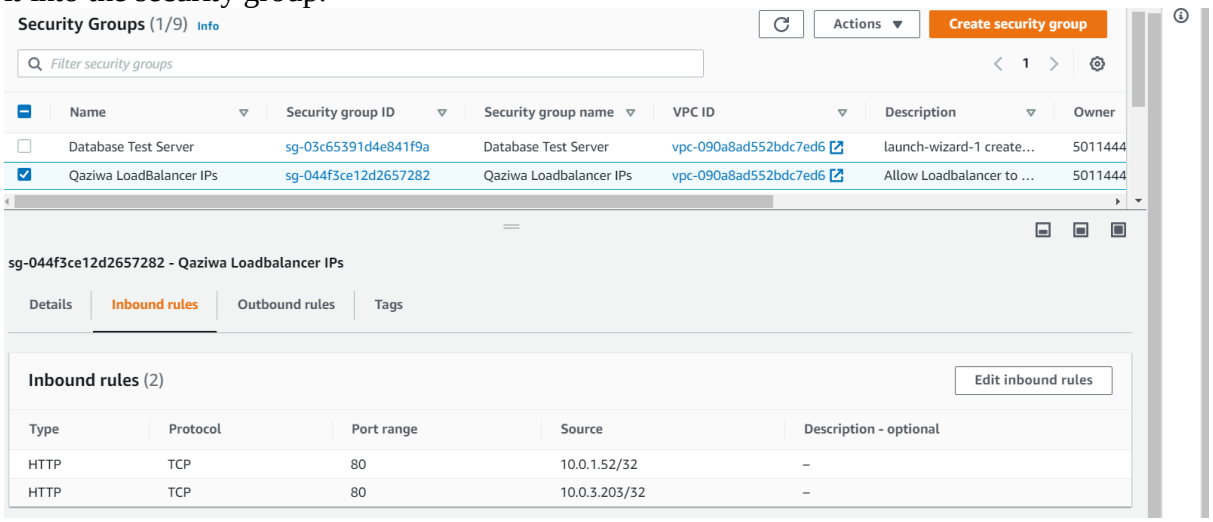
Going further, since we have secured the load balancer, now we want to secure the access of the IP address from the load balancer to the instance itself. This can be seen below.



When accessing the IP address of the instance, we can still view the Web Page, so we need to edit the security group and assign the security to the instances. But before we add the security group, we need to know the IP address of the load balancer, so we need to go to network interfaces, and from network interfaces, we search for the load balancer and then find the IP address section.

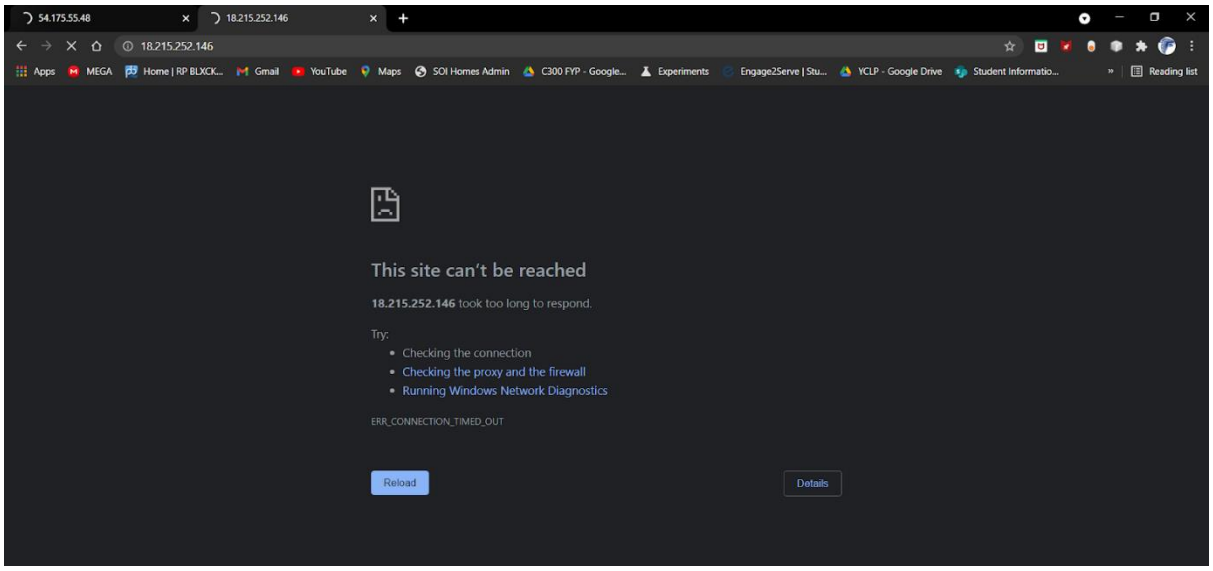


Since there are 2 IP addresses Public and Private, we need to take the private subnet and add it into the security group.



As shown above, we can see that the security group has been created and the rule is applied as HTTP due to it being web browsing and the private IP address of load balancer. 2 different IP addresses by 2 different zones. Now we need to stop the instance and then detach the old

security group to the new security group that we have created that only assigns to the IP address of the load balancer.

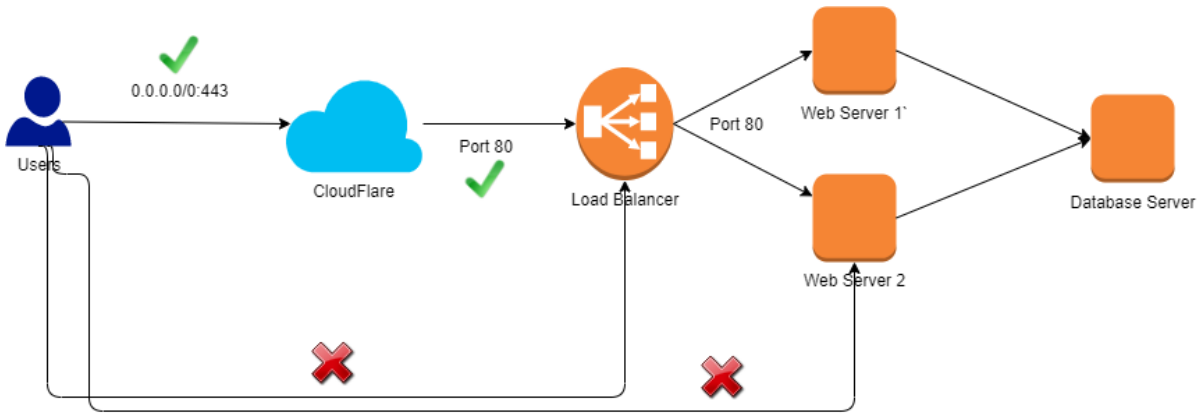


So when we access the instance directly, we can see that we are unable to access the Web Server itself, so therefore to only access this website is by using the domain name that we have created.

As shown above, you can see that when we access the domain itself, we are able to access the webpage and see its content. This is shown that the user cannot access any DNS of the load balancer and even the IP address of the Web Server. They can only access the domain name that is given. This is part of the security features so that the user needs to pass through the cloudflare which helps in detecting malicious content such as DoS, web attacks and much more and following to that, when they access the web server itself, the web server already got another level of Web application firewall and security in which it will also block any unwanted traffic or suspicious packets. Hence, we have applied defense in depth in our network.

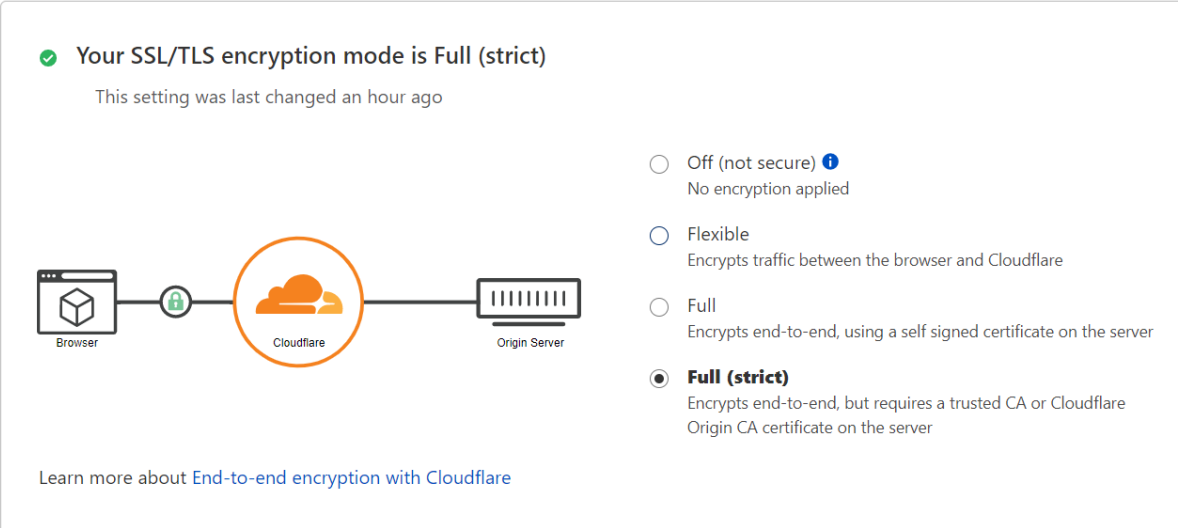
Since we know that due to the AWS limitation of importing certificates and implementing it on LB, we cannot do it. As such after the cloudflare, all of the traffic will be unencrypted.

In summary, we have configured the traffic flow in such a way that users will not have the access to the load balancer or the instance directly, but instead they need to go through the cloudflare in order to access the website.



Implementation of SSL Certificate involving instance:

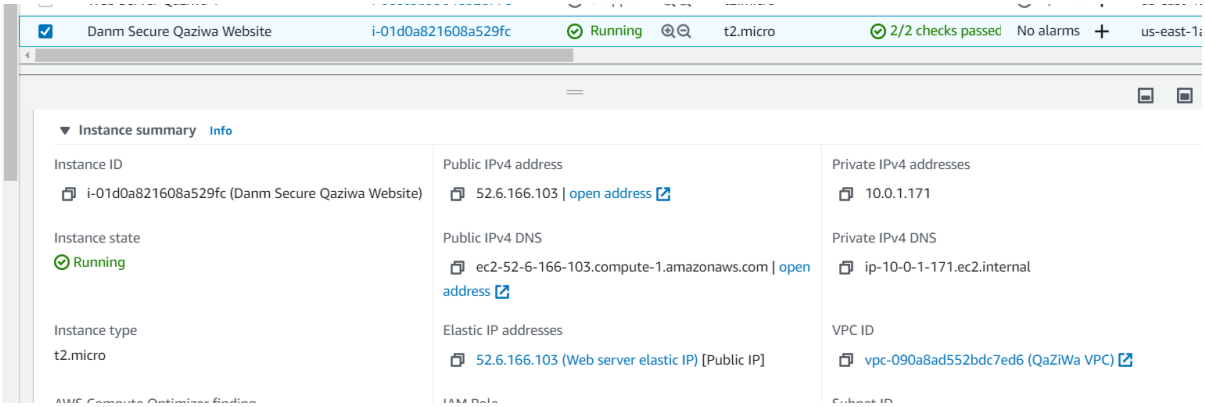
Since we have SSL encryption from the client to the cloudflare itself and not the load balancer (due to not enough privileges given in AWS educate account), we can see that there are vulnerabilities from the cloudflare to the load balancer as it is not encrypted. I pictures is shown below



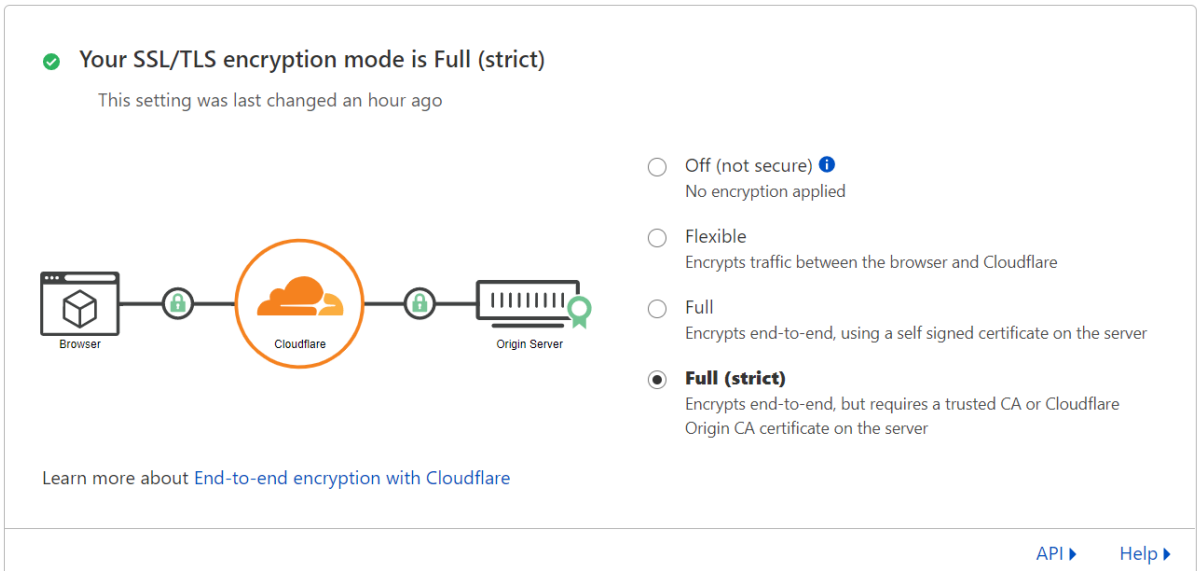
The encrypted part is the Browser to the Cloudflare but not from cloudflare to the instance, therefore, what if an attacker was somehow between the cloudflare and the EC2 instance?

To answer, we will deep down on SSL encryption and ensure that we have a full (strict) SSL encryption.

Firstly, we need to ensure that the web server is built with defense in depth by implementing several configurations and installing mods such as mod security (refer to security testing). Firstly we settle that defense in depth hardening and also even install apache server in AWS EC2 instance. After settling with the web server configuration, now we need to assign an elastic IP to ensure that this ec2 instance has a static public IP address and it would not change (used from the Elastic IP section).



After assigning to the EC2 instance, now we need to create a domain for this particular instance by going to freenom and set up the configuration (refer to Domain name section), then after that, we assign it to cloudflare to manage our DNS names. For the DNS name, we name it qaziwa1.tk. Here, we enable the SSL certificate in the cloudflare to Full (Strict) section.



After we already put the SSL/TLS encryption mode to Full (strict), now we need to go to "Origin Server > Create Certificate > Generate private key and CSR with Cloudflare (RSA 2048)> set the variable shown below> click on create" Then from there you will see a certificate key and the private key. Then we need to copy it into our notepad.

Overview | Edge Certificates | Client Certificates | Origin Server | Custom Hostnames

← Back

Origin Certificate Installation

Follow the steps below to install a certificate on your origin server.

The first step in generating a certificate for your origin is creating a private key and a Certificate Signing Request (CSR). You can provide your own CSR or we can generate a key and CSR using your web browser.

(•) Generate private key and CSR with Cloudflare

Private key type

RSA (2048)

( ) Use my private key and CSR

List the hostnames (including wildcards) on your origin that the certificate should protect. By default your origin certificate covers the apex of your domain (**example.com**) and a wildcard (***.example.com**). If there are others you wish to add, e.g., those not covered by the wildcard such as **one.two.example.com**, you can add them below.

Hostnames

*.qaziwa1.tk × qaziwa1.tk ×

Choose how long before your certificate expires. By default your certificate will be valid for fifteen (15) years. If you'd like to decrease how long your certificate will be valid make a selection below.

Certificate Validity

15 years

Cancel Create

```
-----BEGIN CERTIFICATE-----
MIIEoDCCA4igAwIBAgIUPlKd++jlMFsQB7zOEe7pvD+S
BQAwgYsxCzAJBgNVBAYTAlVTMRkwFwYDVQQKExBDbG91
MgYDVQQLEytDbG91ZEZsYXJlIE9yaWdpbiBTU0wgQ2Vy
aXR5MRYwFAYDVQQHEw1TYW4gRnJhbmNpc2NvMRMwEQYD
MB4XDTIxMDcwMzE4NDUwMFoXDTM2MDYyOTE4NDUwMFow
dWRGbGFyZSwgSW5jLjEdMBsGA1UECxMUQ2xvdWRGbGFy
BgNVBAMTHUNsb3VkRmxhcmUgT3JpZ2luIENlcnRpZmlj
9w0BAQEFAAOCAQ8AMIIBCgKCAQEAnml4ryO06D+hj8gH
Sare6rDl3UiK6Bprm+SczNLfji8yTjojjhZxPSguOE1D
GducVrIOf3ZgSv6ASl2LIZTizVkplZFewbzk8TAhB0wj
FgdidabC0QwAvz/mUWGu4WTbMuo8zzkTiP9maFXsn8tc
XNr0msejQFPPQsoA/mJCqODCYzYocTbWleneY8/N2Ruw
U62ZcG3v8An0NNvbrs4+AOgwORnCjbDnhOc4WOcdsfwI
AR4wDgYDVR0PAQH/BAQDAgWgMB0GA1UdJQQWMBQGCCsG
ATAMBgNVHRMBAf8EAjAAMB0GA1UdDgQWBBSOHR/K3tm4
BgNVHSMEGDAWgBQk6FNXXXw0QIep65TbuuEWePwppDBA
MAYIKwYBBQUHMAGGJGh0dHA6Ly9vY3NwLmNsb3VkZmxh
YTAjBgNVHREEHDAaggwqLnFheml3YTEudGuCCnFheml3
LzAtoCugKYYnaHR0cDovL2NybC5jbG91ZGZsYXJlLmNv
MA0GCSqGSIb3DQEBCwUAA4IBAQC7mGNJTEZeKvMxVRrz
SlvK+S81+3He7w0cpSJXB0zIMLxFfH7QFB6usZPYn9rs
pkJNVAg5itqKGI3m/YLTZF7TsGnX476oL8w8L2xFbsB8
4fLhePDbeQVXzclpvGoNkGnIIrnA1Wl1wxnLSSVZ99lY
qpvTkqyTPzxU+gOO6yX0cGxRnyuN2Q8O7EaBXOfQftO0
zUNxqQnDt3+cluDKp2OJd8VTc7Rlibj+JtX1jlHhHr0m
-----END CERTIFICATE-----


-----BEGIN PRIVATE KEY-----
MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgEA
yAfW6DBCdtvaSYqZknGa3W9Jqt7qsOXdSIroGmub5JzM
TUOv1jnrxsSBiKa3ER/KVVkZ25xWsg5/dmBK/oBKXYsh
TCMba6AK/UNsdLXZX6OPjDkWB2J1psLRDAC/P+ZRYa7h
y2oQmWonJPHYE7VljGd6CRlc2vSax6NAU89CygD+YkKc
G5b0gTD5kdb+u1bbaQv7b+5TrZlwbe/wCfQ029uuzj4A
/Aj/PrvrAgMBAAECggEAOfO85lmqO5LXwJl9oLXVveLs
wIni41GTD3d9k4MLrOEARtUHH9p+99AkxI0jl4bILsTL
6Sx9wsHDXjFFjnYA2P3m9CQ35al8me2xD5HfgA8KACsy
iZX6G/pOV97uyG4nmEZmU59LCUlGJJqZ0pIybhdEC2nJ
ipI9/saPFDBcFyUDacv+CW595fp3r9Ahp3b3FdOagrVL
yTKXHZ1HgUjD3aK6TZhaM7s2fAPU82xxAq1rZhZycQKE
43oTGPshrQYpBZnNoS0dKkwvyG9EABHycdwtGZexc8w3
2PP6dJQC4eIA2NTUcNnfXZmZ6hRXjysQ5NUYg5RsHm2l
gfNpR/bgCnEeNdGvtQ9AbmVp0QKBgQC95Fk8zltSI4vK
C/cRfe0mWlrkyNSGcHqsHkPcKLQsnBeUIq7ID/9MS14Z
p4EGHq8UryB8LTtJnWCOmWyFY5QJtoC7hjkkORC29FdK
KMtiGpM8+wKBgGOD56OnBeVgTy+WOvxvUvRRQLz5jpRR
Aix174z+Q9yONYZeT8IhqNEDXpYbCcc1TsOXzVzme1Uf
pFYvAlu7dibJl8TuLWT8WlOdrTjlZJNo789QfIb79qg4
EEbcZ60cJbtFUcKxaya15pE5La/daVr1CfbA3UeOTQ9y
VsriVe61cHkHRWXB1FaXoF8cZgiCJTSrSiqGYVZAxoTC
qvPuvni8o5i3O76YzERCcIVm6rIp31kcIGIjCnECgYEA
m+kowoDAamZqh8eTTUy5imuUwjejnee2GrBtDdyJaj9w
JiahCZ/NBo0/I7pXcIJMYXRkJD8cePBRHtv8SlMtReB2
unuaWGhDGm/8HFMpNCREFN4=
-----END PRIVATE KEY-----
```

Partly shown due to security issues of the keys
After we already get the key, then we need to Generate a CSR and Private Key for the web server we use “openssl req -newkey rsa:2048 -nodes -keyout qaziwa1.tk.key -out qaziwa1.tk.csr

```
root@ip-10-0-1-171:~# openssl req -newkey rsa:2048 -nodes -keyout qaziwa1.tk.
.tk.csr
Generating a RSA private key
.....................................................+++++
.....................+++++
writing new private key to 'qaziwa1.tk.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:SG
State or Province Name (full name) [Some-State]:qaziwa1
Locality Name (eg, city) []:qaziwa1
Organization Name (eg, company) [Internet Widgits Pty Ltd]:qaziwa1
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:qaziwa1.tk
Email Address []:wafimuhd5@gmail.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
root@ip-10-0-1-171:~# cat qaziwa1.tk.csr
```

Then we fill up the credentials shown above.

We then copy the private key and certificate that is given to us from the cloudflare into a file. We called it qaziwa1.tk.key (for the private key from cloudflare) and qaziwa1.tk.crt (for certificate key from cloudflare)

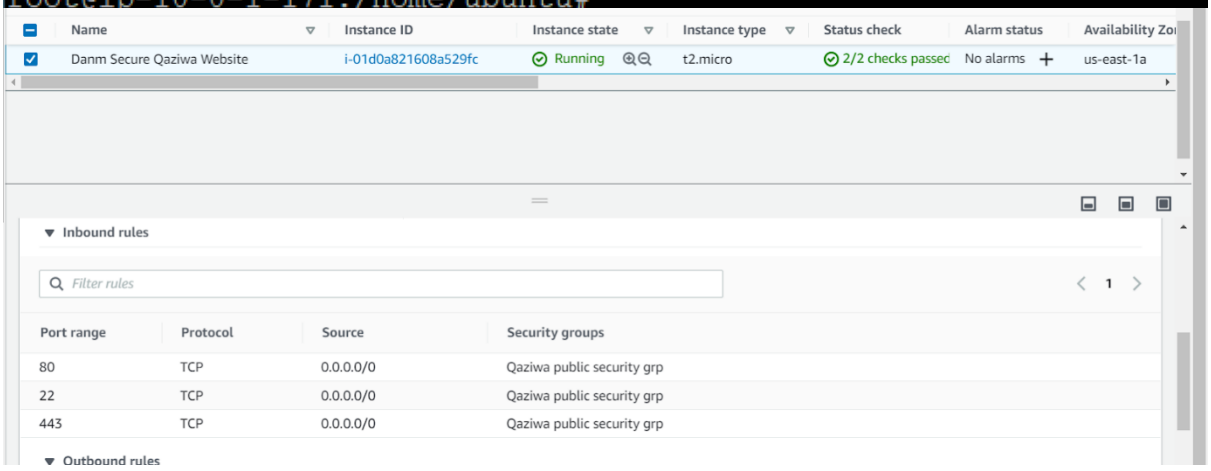
```
root@ip-10-0-1-171:/home/ubuntu# nano qaziwa1.tk.key
root@ip-10-0-1-171:/home/ubuntu# nano qaziwa1.tk.crt
```

Once we are done adding the keys into the file, then we need to go to nano /etc/apache2/sites-avaialble/000-default.conf and edit the configuration shown below

```
<VirtualHost *:443>
        # The ServerName directive sets the request scheme, hostname and port that
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
        # specifies what hostname must appear in the request's Host: header to
        # match this virtual host. For the default virtual host (this file) this
        # value is not decisive as it is used as a last resort host regardless.
        # However, you must set it for any further virtual host explicitly.
        #ServerName www.example.com

        ServerAdmin webmaster@qaziwa1.tk
ServerName qaziwa1.tk
ServerAlias www.qaziwa.tk
        DocumentRoot /var/www/html
SSLEngine on
SSLCertificateFile /home/ubuntu/qaziwa1.tk.crt
SSLCertificateKeyFile /home/ubuntu/qaziwa1.tk.key
        # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
        # error, crit, alert, emerg.
        # It is also possible to configure the loglevel for particular
        # modules, e.g.
        #LogLevel info ssl:warn

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

        # For most configuration files from conf-available/, which are
        # enabled or disabled at a global level, it is possible to
        # include a line for only one particular virtual host. For example the
        # following line enables the CGI configuration for this host only
        # after it has been globally disabled with "a2disconf".
        #Include conf-available/serve-cgi-bin.conf
    SecRuleEngine On
    SecRule ARGS:modsecparam "@contains test" "id:4321,deny,status:403,msg:'ModSecurity test rule has triggered'"
</VirtualHost>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
<VirtualHost *:80>
ServerName qaziwa1.tk
Redirect permanent / https://qaziwa1.tk/
</VirtualHost>
```

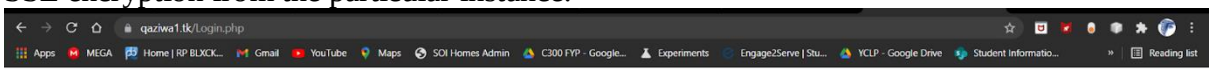
After editing it, we then do a sudo a2emod ssl, which is to enable ssl and we do a service apache2 restart, to restart the apache server.

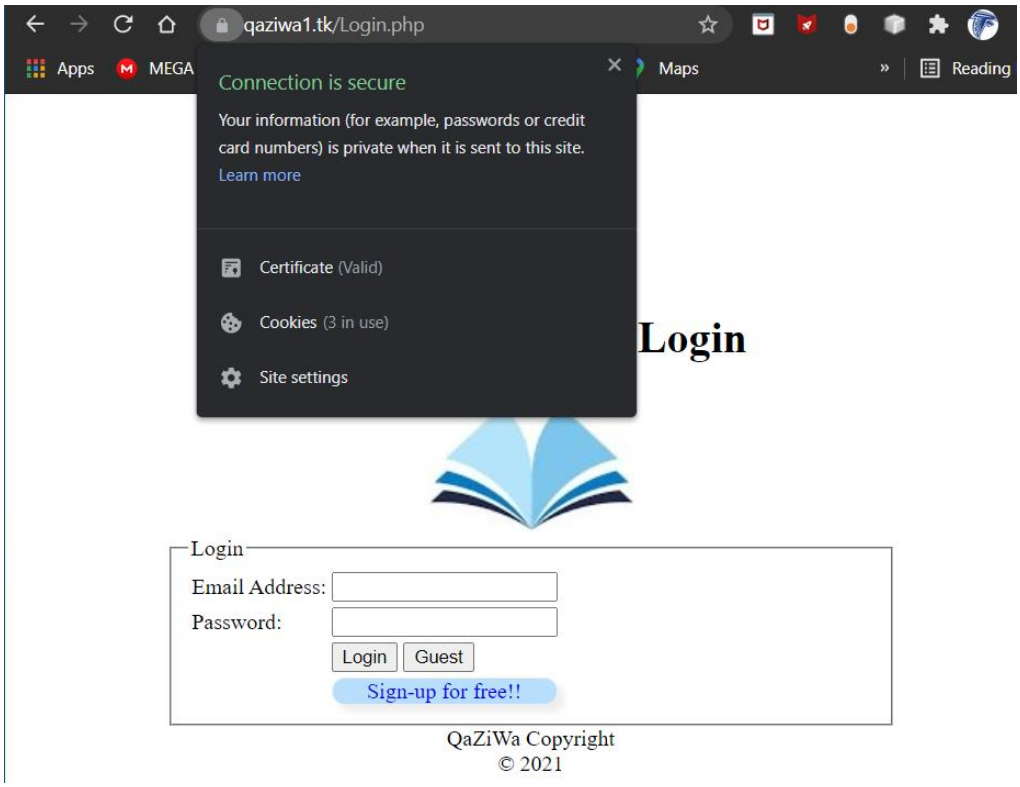
```
root@ip-10-0-1-171:/home/ubuntu# sudo a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Module socache_shmcb already enabled
Module ssl already enabled
root@ip-10-0-1-171:/home/ubuntu# service apache2 restart
root@ip-10-0-1-171:/home/ubuntu#
```

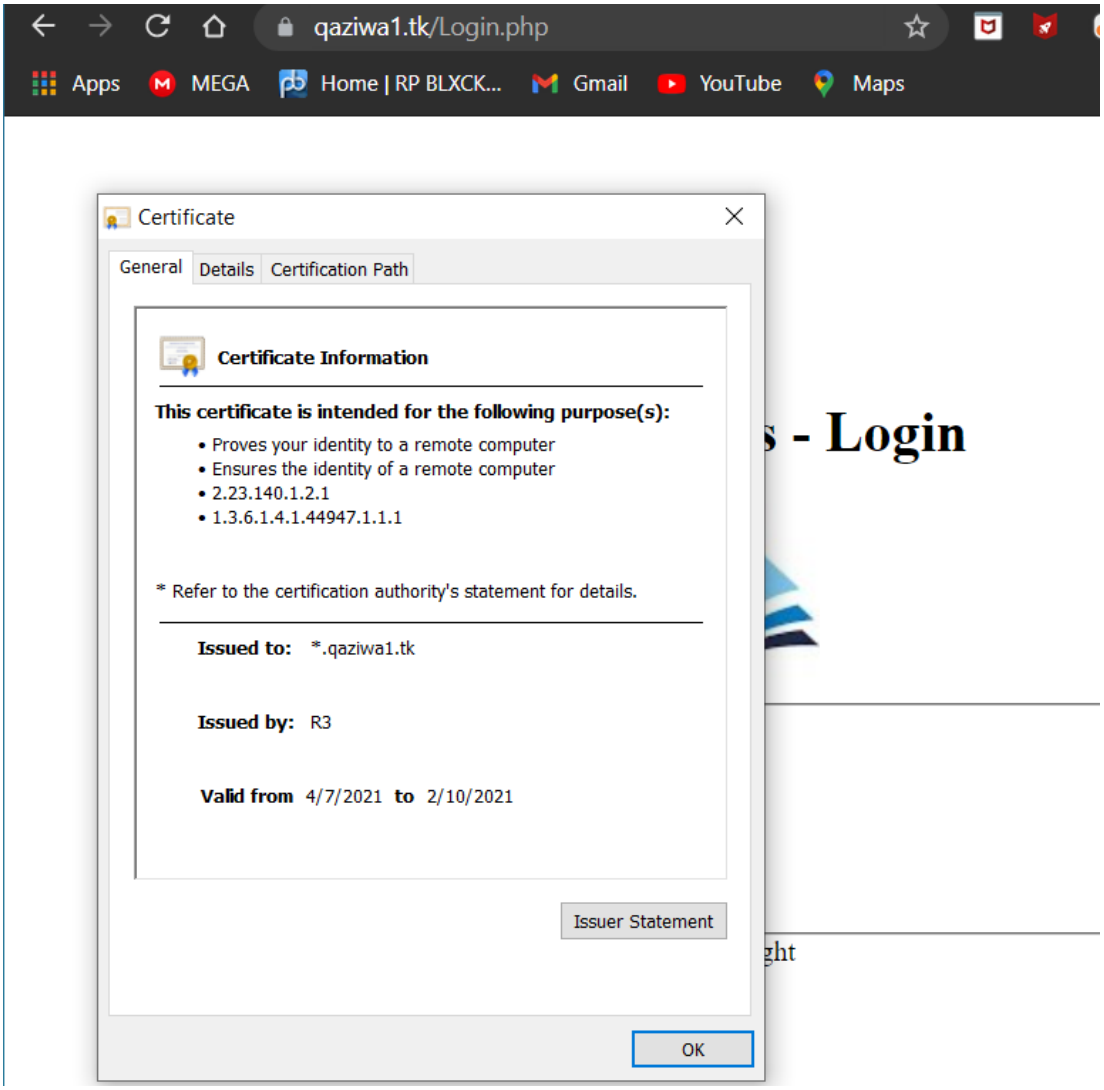


After configuring that, we then need to add another rule which enables port 443 (HTTPS) due to the fact that we are enabling HTTPS in our website.

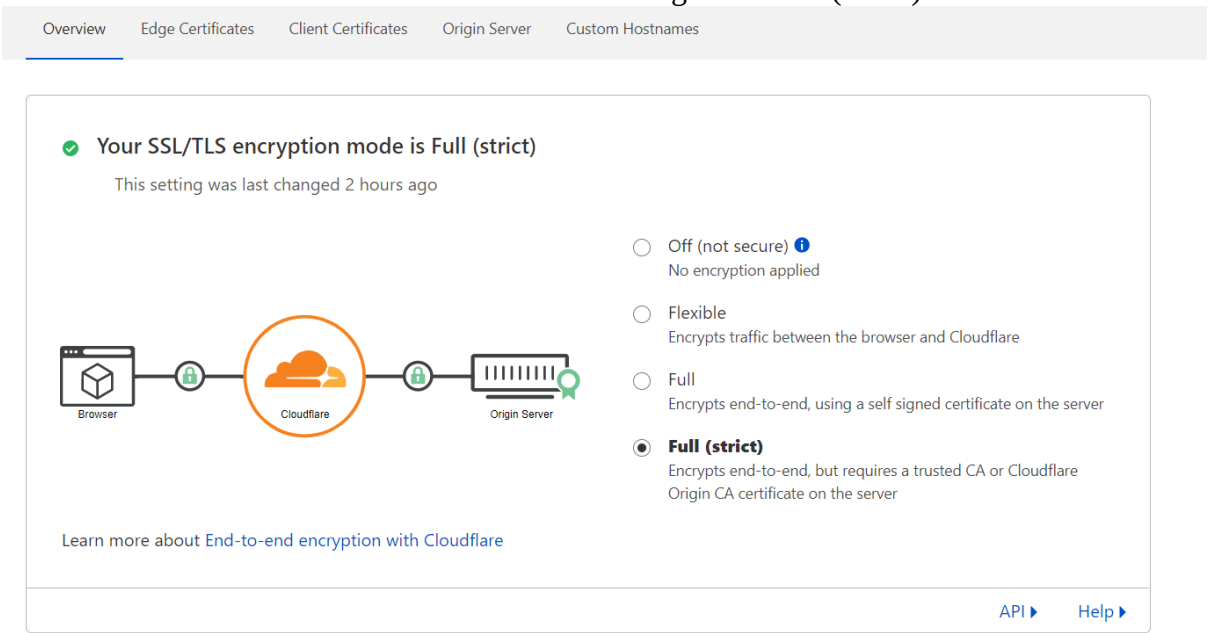
Then we wait for a while, to ensure the cert from the cloudflare is being applied to the web server. After a few minutes, we refresh the domain name (qaziwa1.tk) and see if there is an SSL encryption from the particular instance.



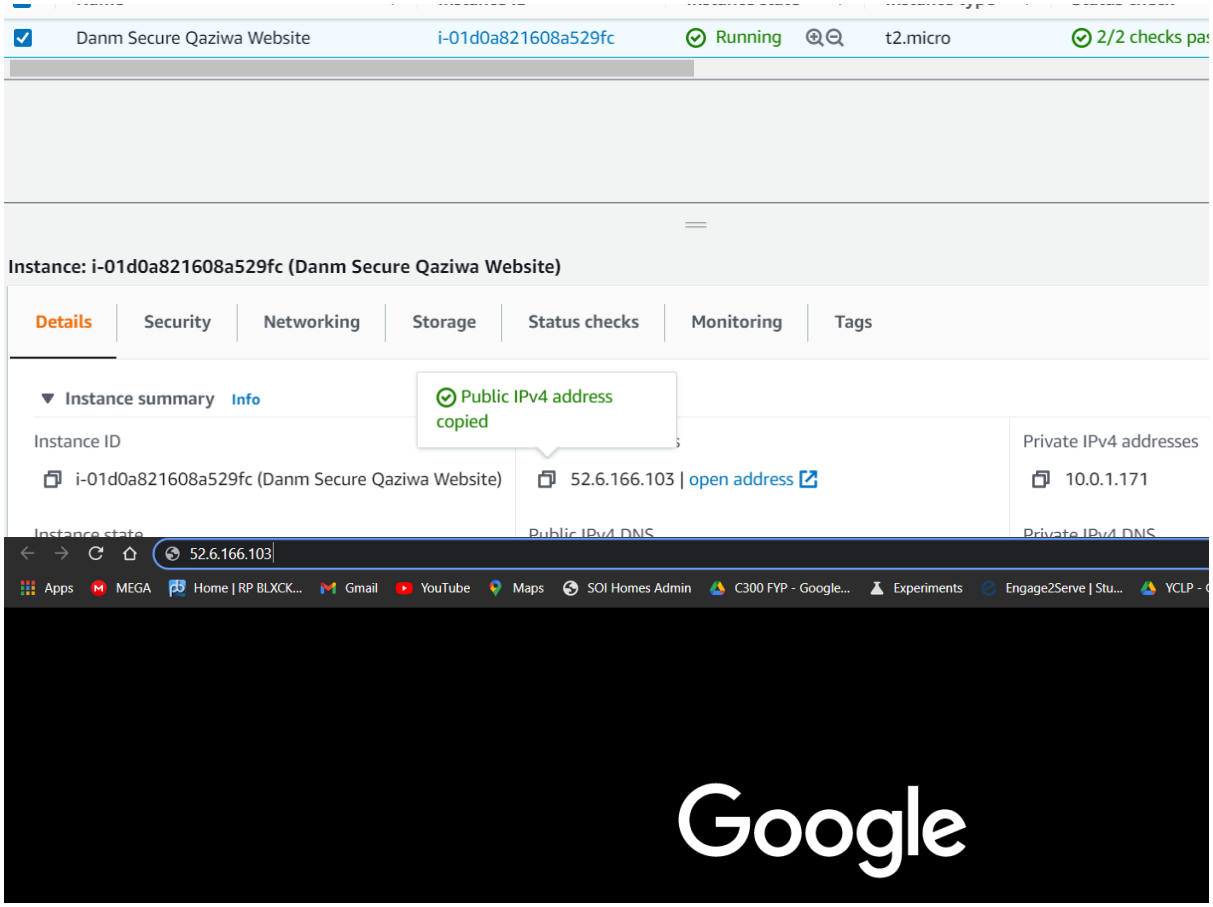




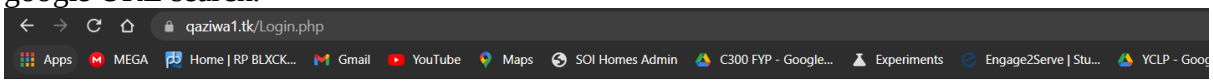
As shown above, we can see that a certificate is being applied to the web server and we want to check the cloudflare to see if the SSL/TLS is being set to Full (Strict) section.

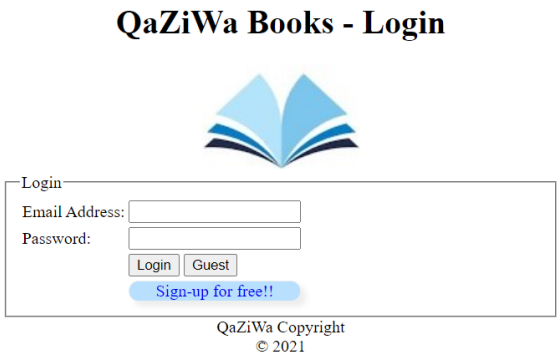


Yes, it showed that the Full (strict) section is being applied from the client to the cloudflare and to the AWS ec2 instance. As such due to AWS not having enough privileges, of inserting a certificate manager and implementing it on AWS Educate, therefore, we can just implement a full SSL certificate on AWS EC2 instance.



Now instead we try to access the IP address directly from the instance and enter it on the google URL search.



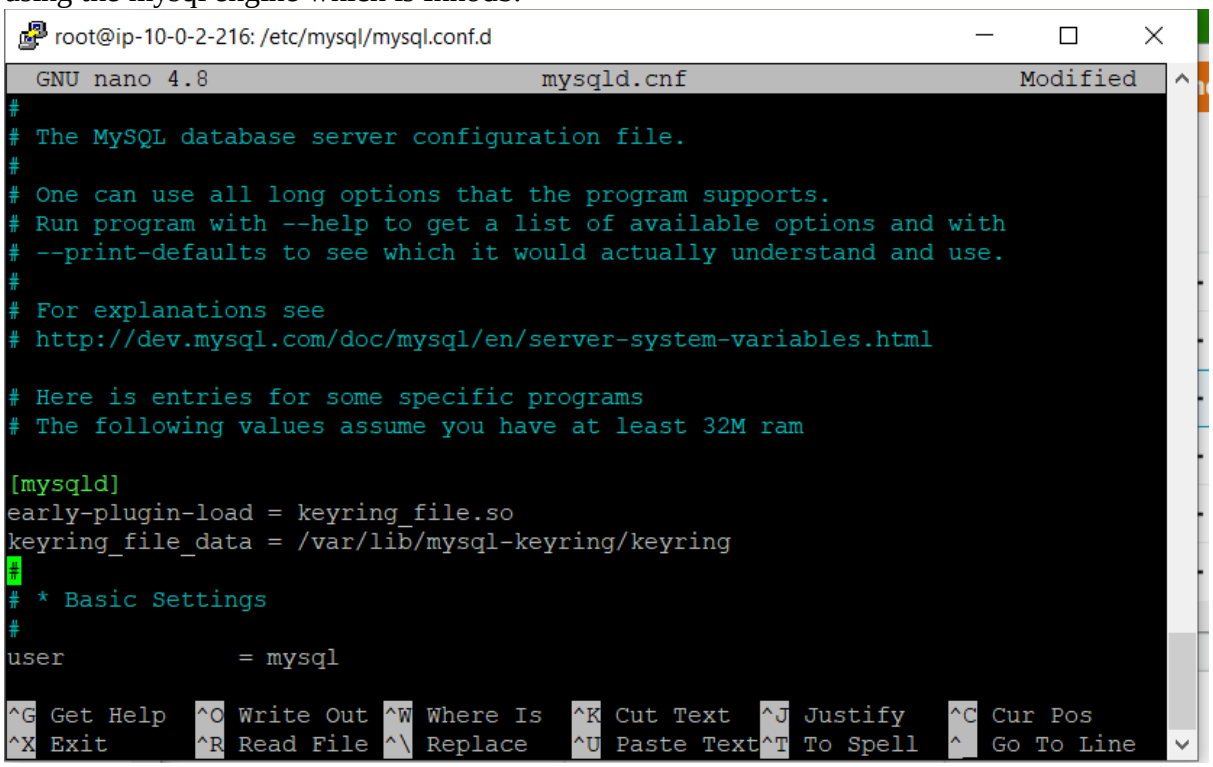


As you can see it will automatically change the IP address to qaziwa1.tk. From here we know that it is stable and the connection is secure. Since we already settled with encrypting data in transmission using HTTPs now is the time to encrypt data at rest, which is the database side.

Database Encryption:

Innodb encryption:

Encryption of the database is very important so that when the attacker wants to exfiltrate the database, it can ensure that all of the data the attacker is being encrypted and therefore, making sure that the attacker will not have the user data on his hands in plaintext. We will be using Innodb to encrypt our database and its table so that we can ensure total security in our database and its tables. As such, below is our screenshot on the encryption of the database using the mysql engine which is Innodb.



Firstly we go to the directory “cd /etc/mysql/mysql.conf.d” and “nano mysqld.cnf”. After seeing the configuration page of the mysql, we can see, edit and add “early-plugin-load = keyring_file.so” and “keyring_file_data = /var/lib/mysql-keyring/keyring”, then we save the file.

.

After saving the file, we then enter the into the mysql section shown below

```
root@ip-10-0-2-216:/etc/mysql/mysql.conf.d# ls
mysql.cnf  mysqld.cnf
root@ip-10-0-2-216:/etc/mysql/mysql.conf.d# mysql -u qaziwa -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.25-0ubuntu0.20.04.1 (Ubuntu)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Using “mysql -u qaziwa -p” to login into the database, and then after that we enter the required password to enter.

Upon entering we enter the following commands shown below to install and load a plugin into the database, configure the keyring file.

```
mysql> INSTALL PLUGIN keyring_file SONAME 'keyring_file.so';
Query OK, 0 rows affected (0.02 sec)

mysql> SET GLOBAL keyring_file_data = '/var/lib/mysql-keyring/keyring';
Query OK, 0 rows affected (0.00 sec)

mysql>
```

So now, we can create encrypted tables and even alter existing tables to encrypt them.

Below is the list of encryption that we use on the database level and the tables level.

```
mysql> ALTER SCHEMA qaziwa DEFAULT ENCRYPTION = 'Y';
Query OK, 1 row affected (0.01 sec)

mysql>
```

On the above, it is a database level encryption that we encrypt the database named ‘“qaziwa”

```
mysql> use qaziwa;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----------------+
| Tables_in_qaziwa |
+-----------------+
| books           |
| cart_books      |
| users           |
+-----------------+
3 rows in set (0.01 sec)

mysql>
```

After encrypting the database level, then we go deep to encrypt the tables in the database. So we view which tables are in the Qaziwa database.

```
mysql> ALTER TABLE users ENCRYPTION='Y';
Query OK, 7 rows affected (0.06 sec)
Records: 7  Duplicates: 0  Warnings: 0

mysql> ALTER TABLES books ENCRYPTION='Y';
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that
corresponds to your MySQL server version for the right syntax to use near 'TABLE
S books ENCRYPTION='Y'' at line 1
mysql> ALTER TABLE books ENCRYPTION='Y';
Query OK, 11 rows affected (0.06 sec)
Records: 11  Duplicates: 0  Warnings: 0

mysql> ALTER TABLW cart_books ENCRYPTION='Y';
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that
corresponds to your MySQL server version for the right syntax to use near 'TABLW
 cart_books ENCRYPTION='Y'' at line 1
mysql> ALTER TABLE cart_books ENCRYPTION='Y';
Query OK, 8 rows affected (0.07 sec)
Records: 8  Duplicates: 0  Warnings: 0

mysql>
```

After we know the tables that are inside the Qaziwa database, we now proceed to encrypt that to add a second layer of defense. As shown in a highlighted above, we encrypt the users, books, cart_books. This is to ensure that the tables cannot be seen in plaintext when the attacker manages to dump out the database.

Going further, we will be using a Master Key Rotation, this means that the master encryption key will be periodically rotated and let say the key is compromised, the attack will not have the suitable key to decrypt the tables and database, due to the key will keep changing. Everytime the master key is rotated, all tablespace keys in the MYSQL instance will be re-encrypted and saved back to their own respective tablespace headers. The rotating master encryption key will only change the master encryption key and re-encrypt the tablespace keys. It does not decrypt or re-encrypt the tablespace data.
> Table space is a storage location where the actual data in the database is stored.

```
mysql> ALTER INSTANCE ROTATE INNODB MASTER KEY;
Query OK, 0 rows affected (0.00 sec)

mysql>
```

So to enable the Master Key Rotation, we will be using the command “ALTER INSTANCE ROTATE INNODB MASTER KEY;”

After finishing the database and table encryption command using innodb engine, now we need to verify if the database and tables are encrypted or not.

```
mysql> #Now time to verify;
mysql> SELECT TABLE_SCHEMA, TABLE_NAME, CREATE_OPTIONS FROM
    ->        INFORMATION_SCHEMA.TABLES WHERE CREATE_OPTIONS LIKE '%ENCRYPTION%'
;
+--------------+------------+----------------+
| TABLE_SCHEMA | TABLE_NAME | CREATE_OPTIONS |
+--------------+------------+----------------+
| qaziwa       | books      | ENCRYPTION='Y' |
| qaziwa       | cart_books | ENCRYPTION='Y' |
| qaziwa       | users      | ENCRYPTION='Y' |
+--------------+------------+----------------+
3 rows in set (0.01 sec)

mysql>
```

In the above section, we can see that the tables in the database are encrypted.

```
mysql> SELECT SCHEMA_NAME, DEFAULT_ENCRYPTION FROM
    -> INFORMATION_SCHEMA.SCHEMATA WHERE DEFAULT_ENCRYPTION='YES';
+-------------+--------------------+
| SCHEMA_NAME | DEFAULT_ENCRYPTION |
+-------------+--------------------+
| qaziwa      | YES                |
+-------------+--------------------+
1 row in set (0.00 sec)

mysql>
```

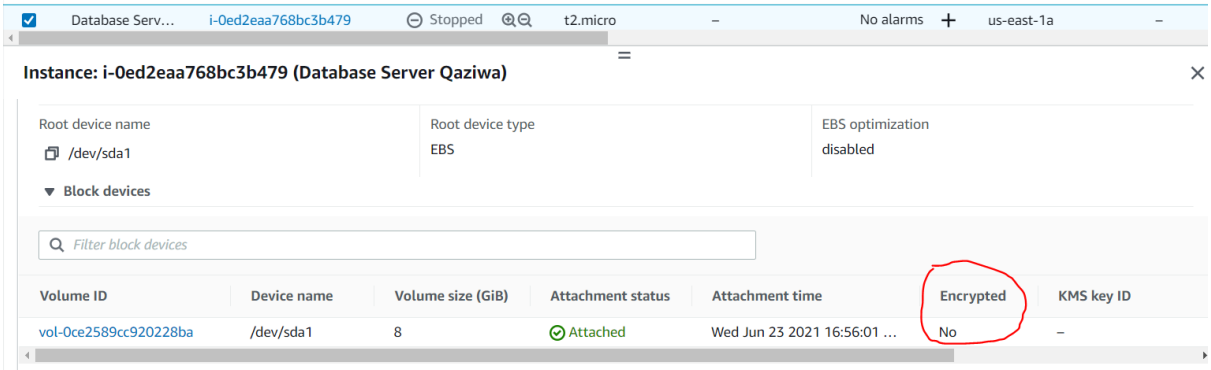
In the above section, we can see that the database is encrypted.

Therefore, encryption in the database is important to prevent any attackers from exfiltrating the database and the tables and put it into an advantage and gain the user information. Hence, the database encryption, we can certainly make sure that all of the information in the database will not be seen in plaintext.

Going further than this, it is also recommended also to encrypt the Database Instance via the EBS volume of the instance. EBS volume is durable, which is a block-level storage device that we can attach on the instance. So, as such we need to encrypt the EBS volume so that to ensure that stored data is secure. This is also part of defense in depth in which we encrypt the Database volume and ensure that when an attacker gets the instance volume, it would not be in readable plaintext but instead an encrypted volume using the AWS Key Management.

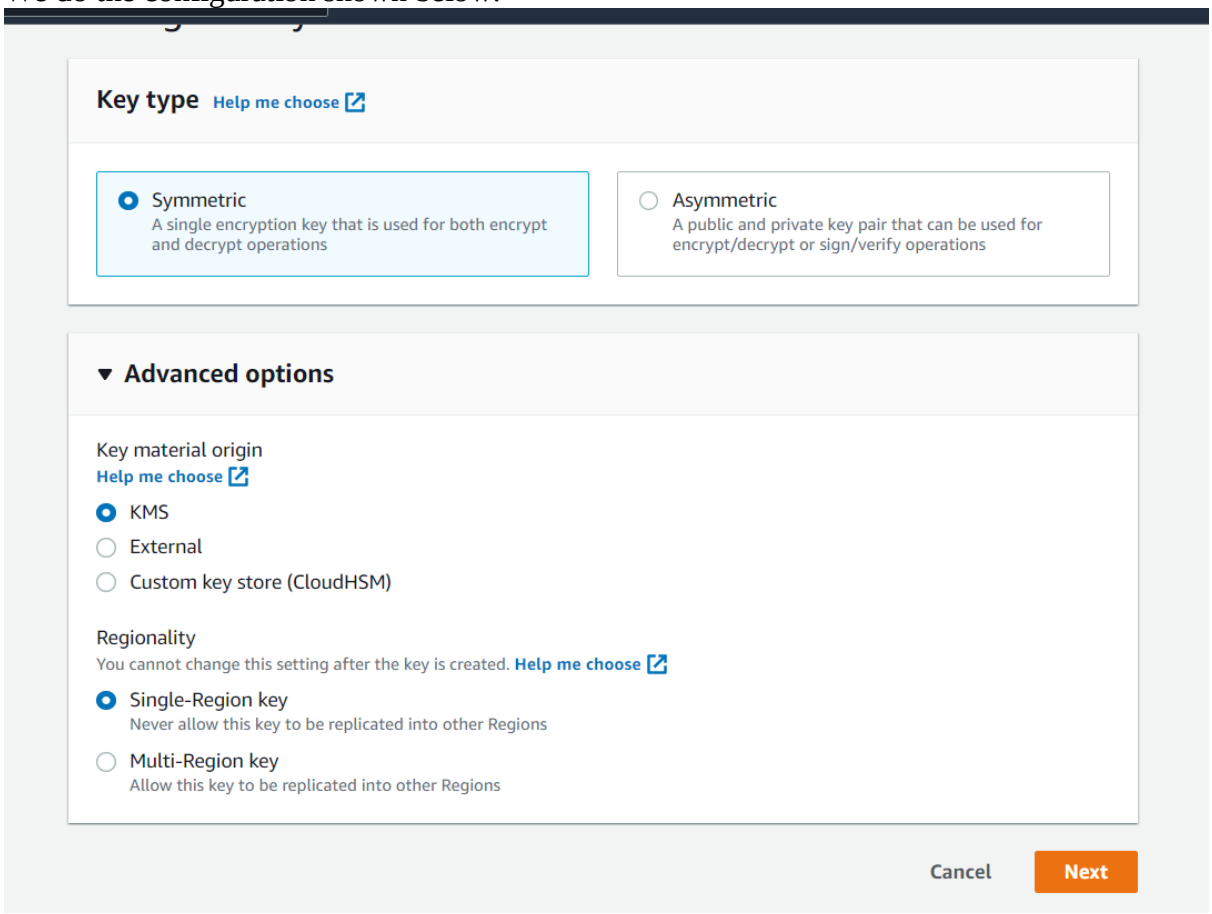
As shown below, it is shown that the Database instance volume is not encrypted.

Volume encryption:

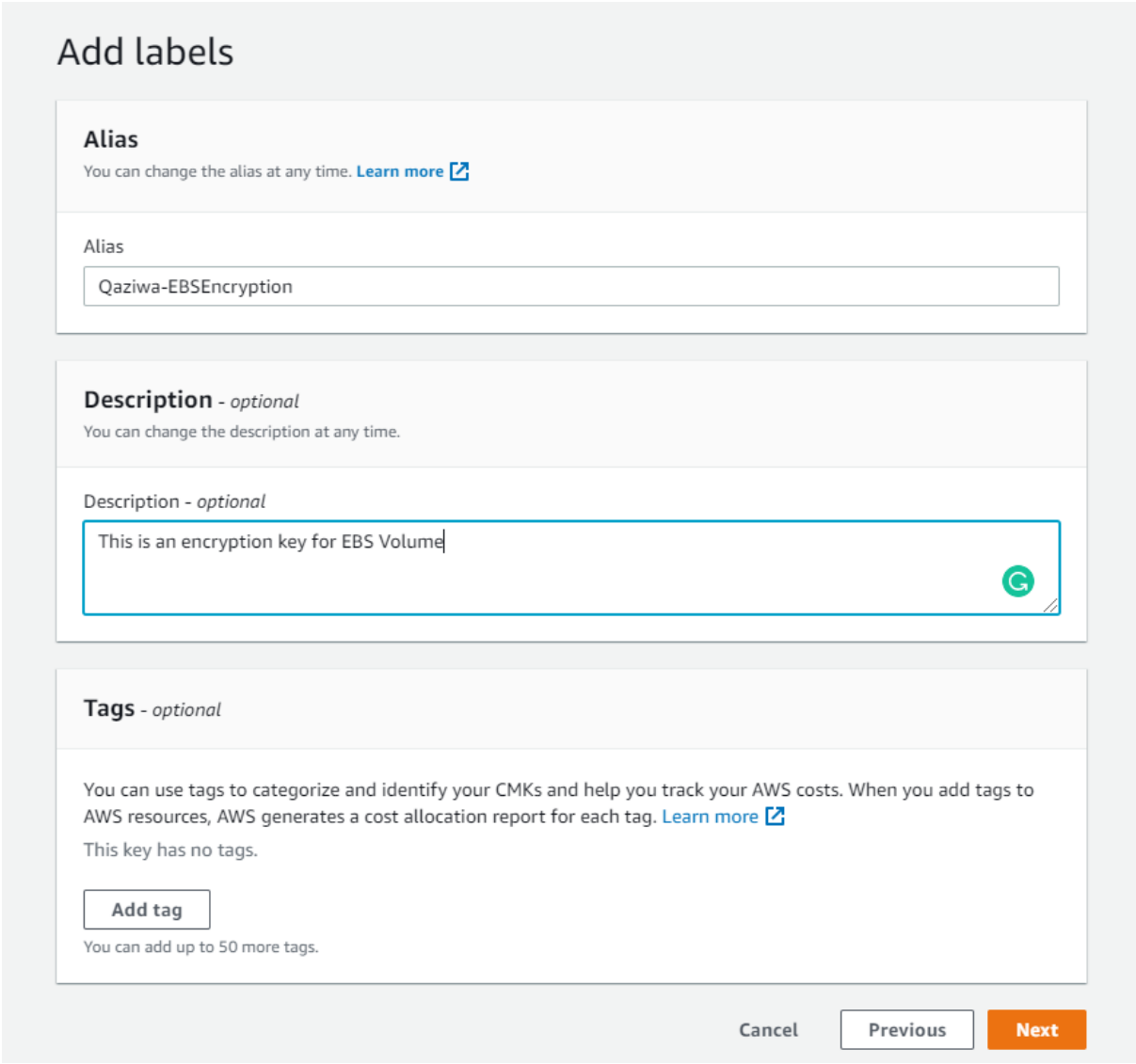


Now we know that it is not encrypted, now is the time to encrypt the volume of the database. To start off, we need to create a symmetric key from AWS Key Management Service (KMS) to ensure that we can use the key to encrypt the EBS volume.

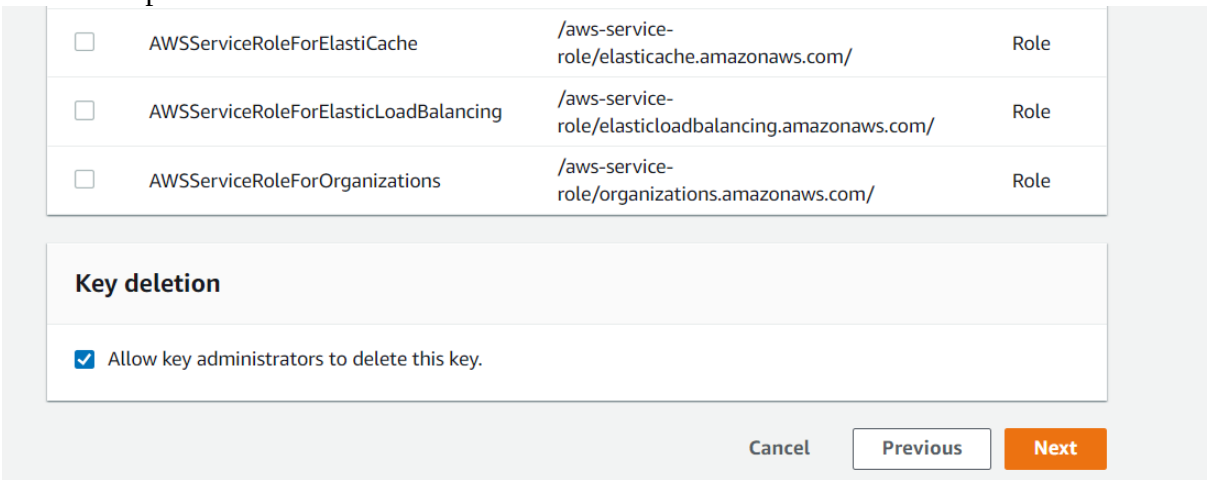
We do the configuration shown below.

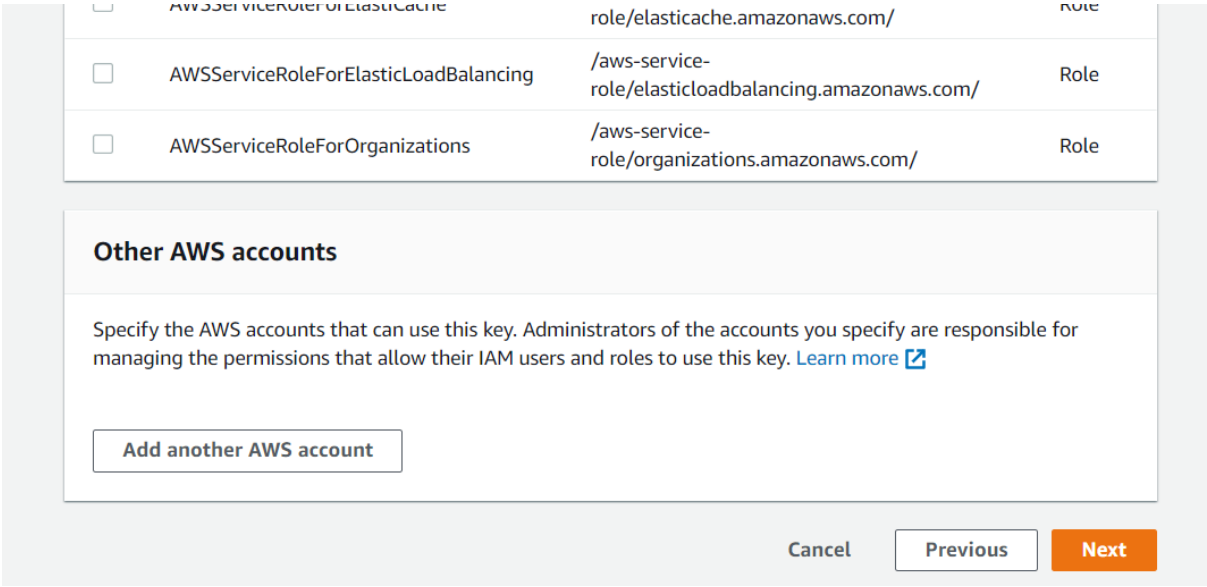


Then we click next



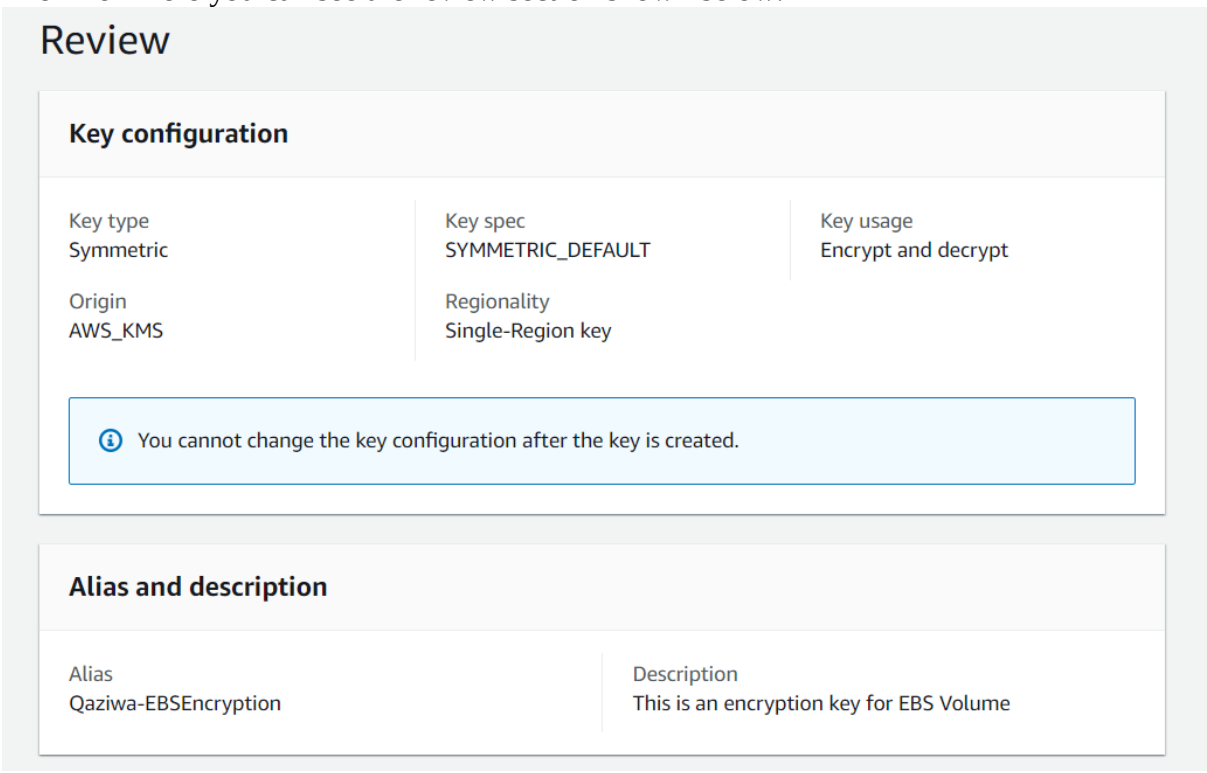
We filled up the section above and click on next

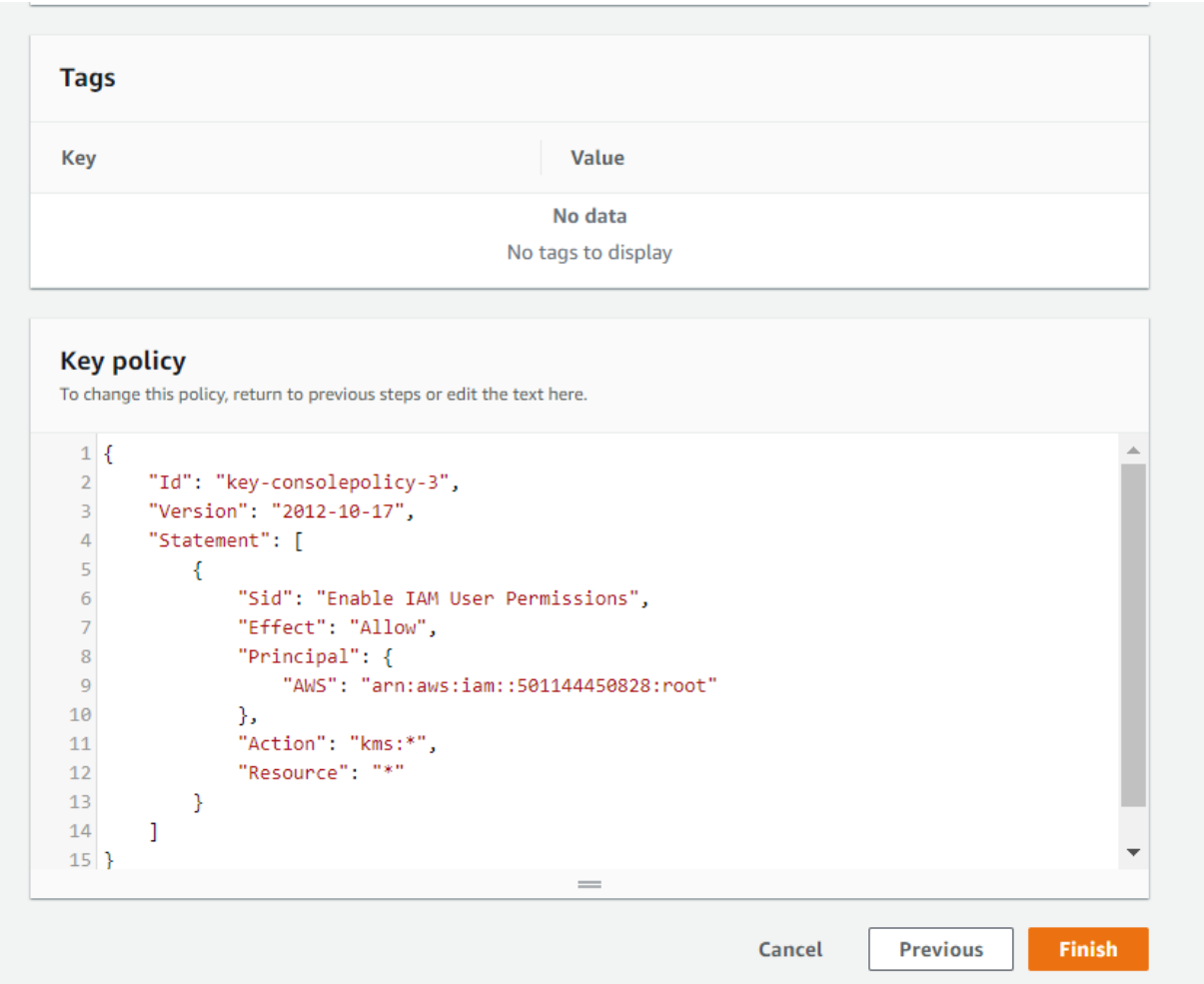




From the above, we must not select anything and click on next twice.

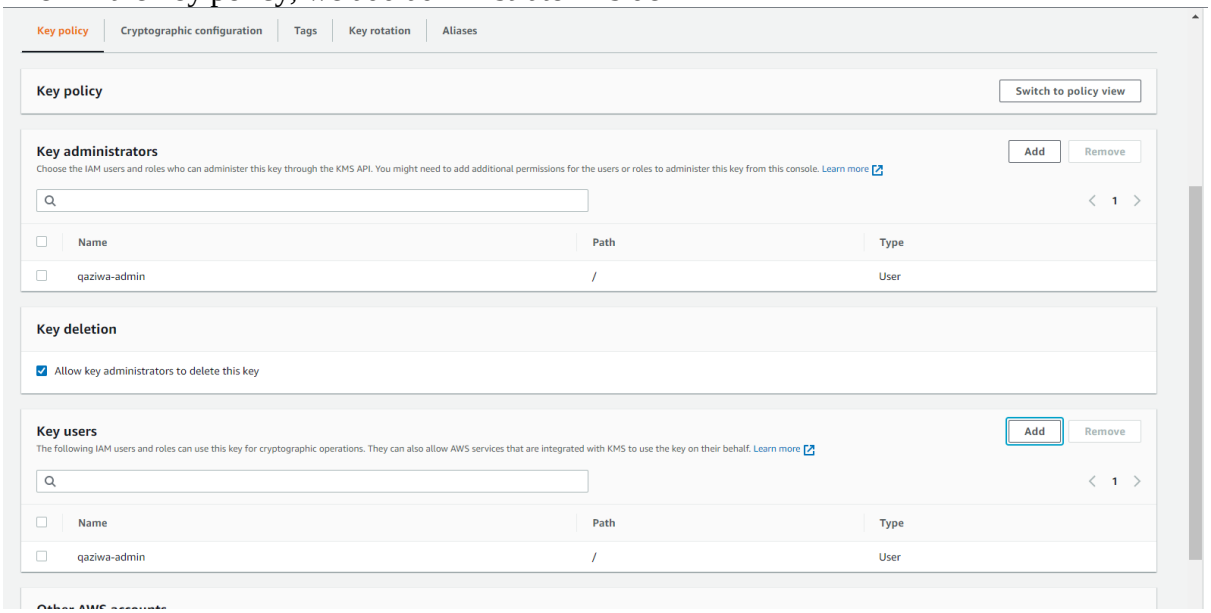
Then from here you can see the review section shown below.



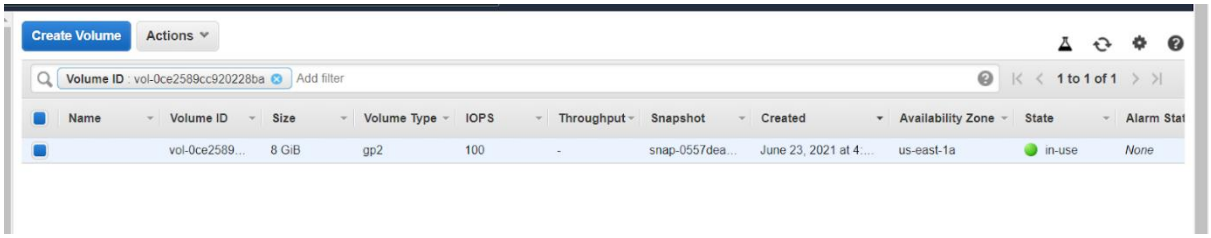


Then click on finish.

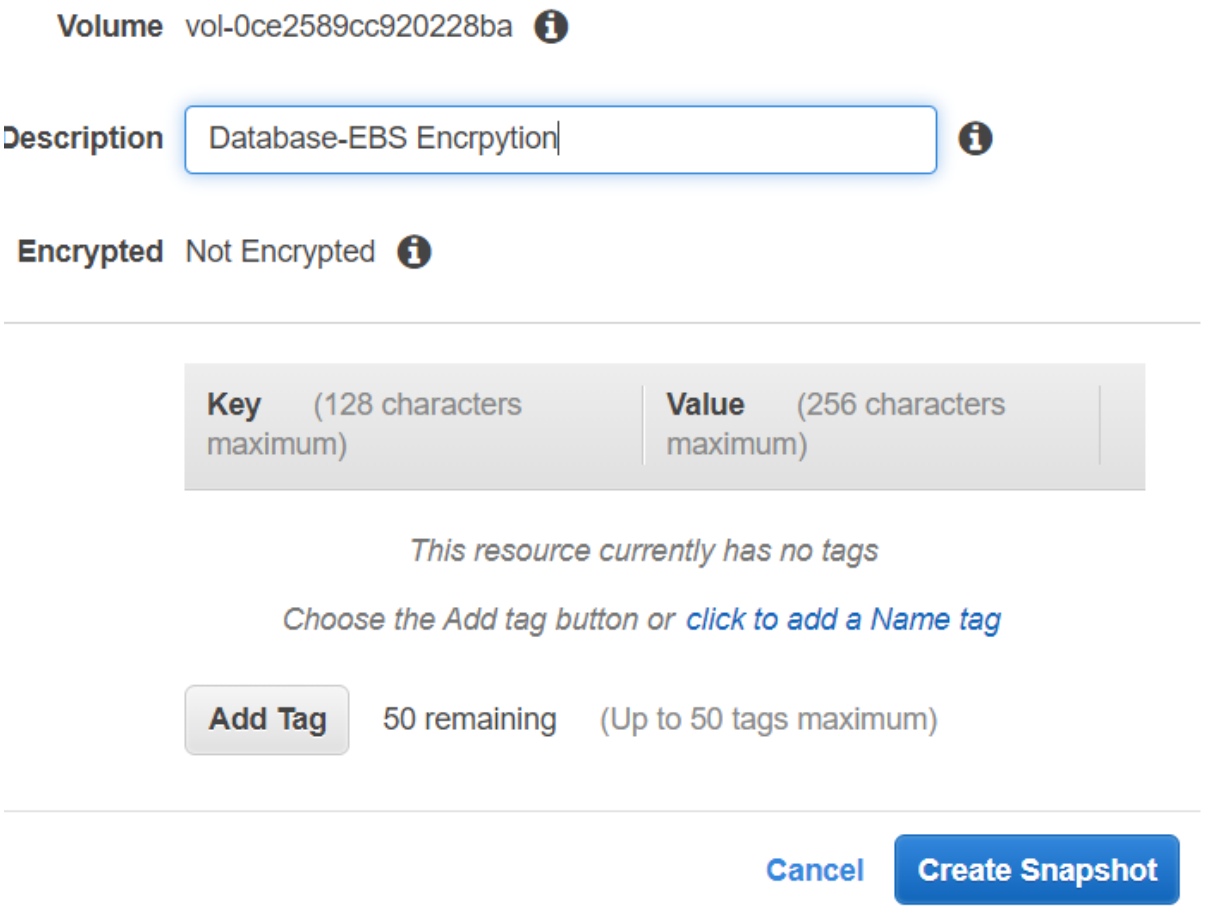
Then in the key policy, we add administrator inside



Once that is done, now we need to go to the EBS Volume section shown below

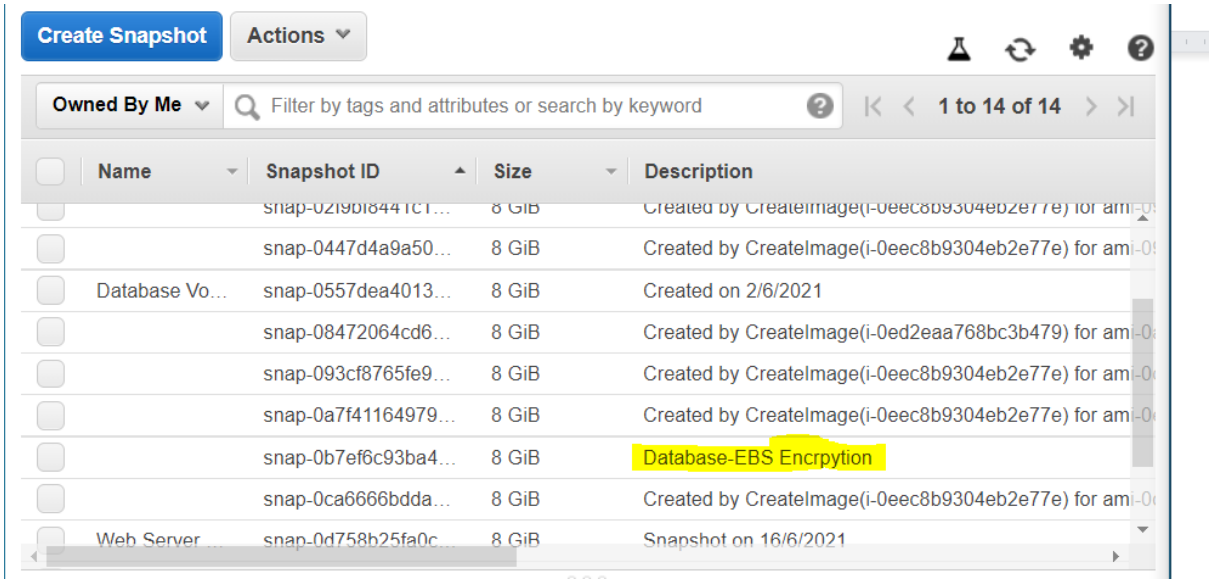


Now we need to create a snapshot for this, by doing on Action > Create Snapshot

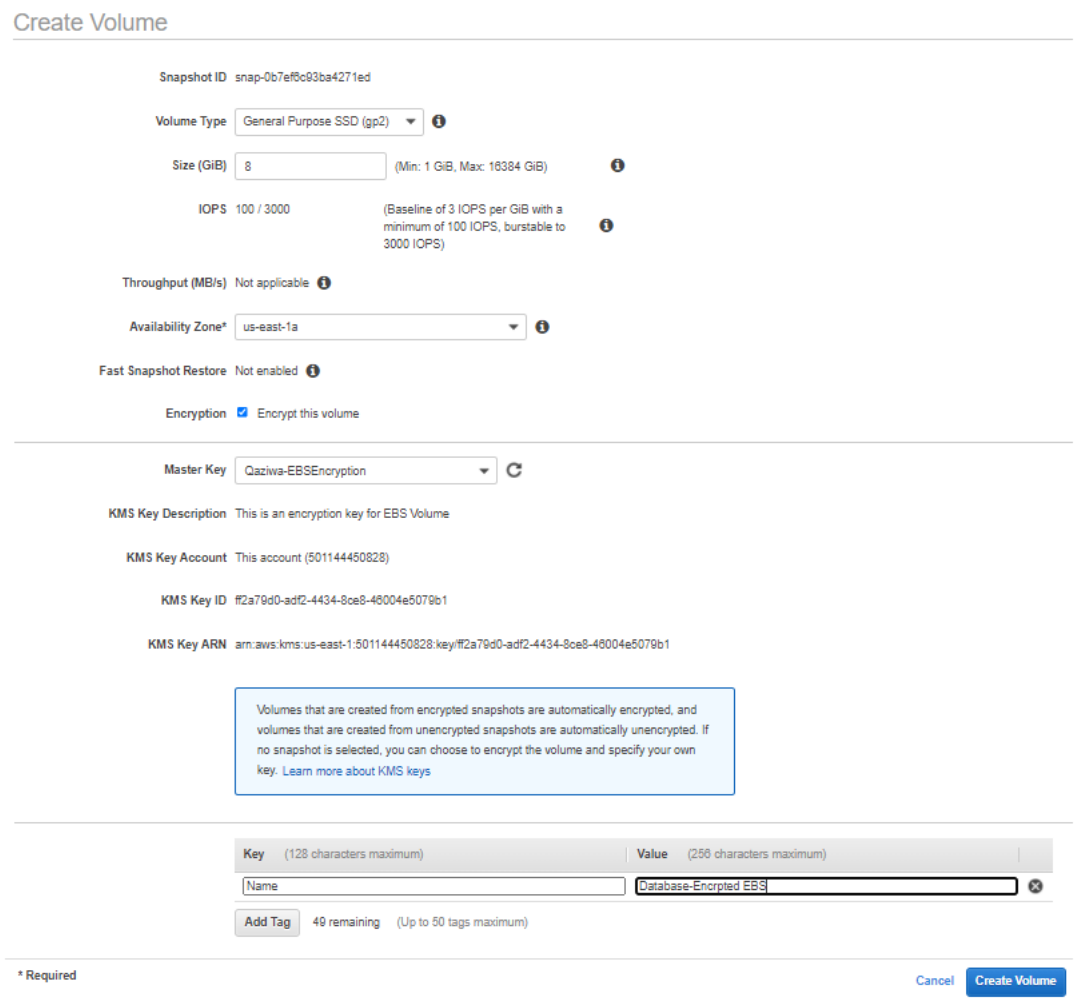


And then click on create snapshot.

After this go to the snapshot section, and locate the snapshot that we have done and wait for a while until it is fully created.

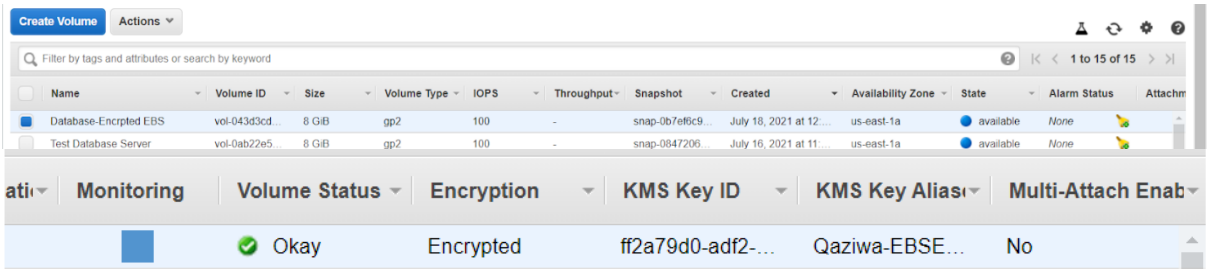


Once the creation of the Database-EBS Encryption, Now we need to select it > action > create volume. Fill up the configuration that is shown below.



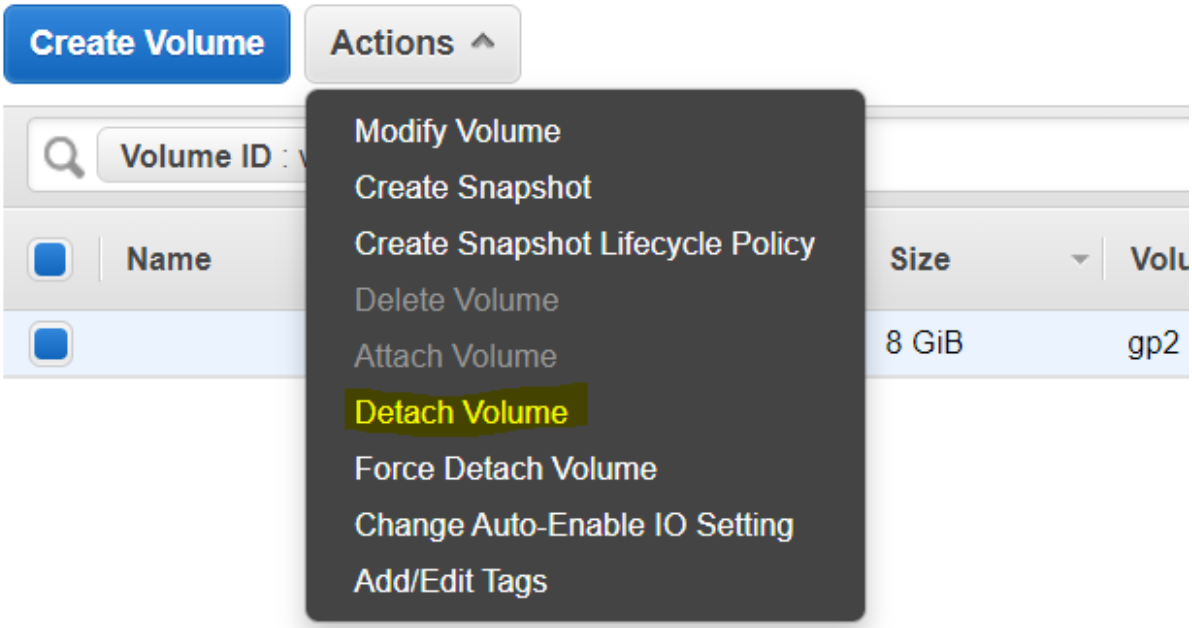
After this create volume

Moving on, we go back to the EBS volume section and locate the “Database-Encrypted EBS”



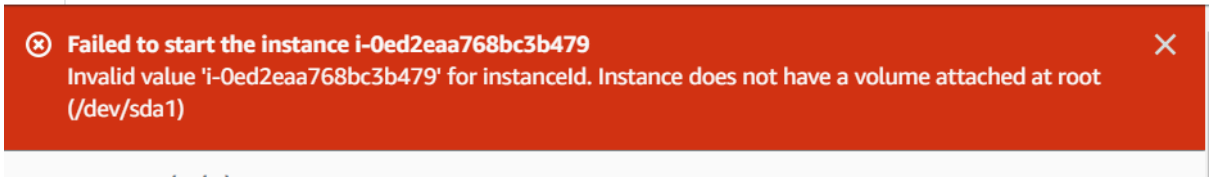
As shown above, we have successfully created the EBS volume that is encrypted.

After this, we need to detach, the current non-encrypted EBS Volume at is attached in the Qaziwa Database Server. So to do this, we go back to the instance, click on the Database server and after that go to storage and click on the volume ID to locate the volume that is attached to the database server. Then we select the Volume ID > click on actions > and click on detach volume > click on confirm detach. (Side note: Make sure that the instance is being in stopped position)

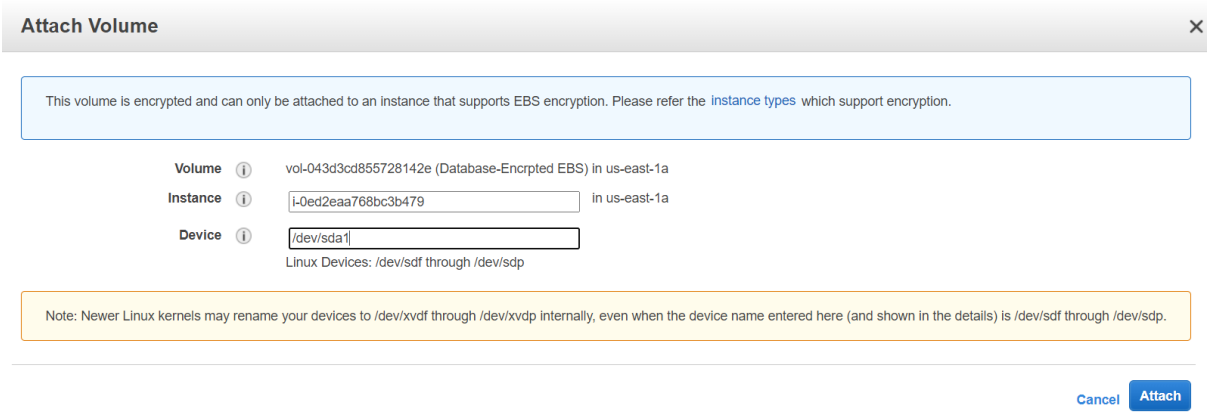


Once detach, then we need to wait for a while after it has fully detached from the database instance server.

Once done, go back to the database instance, and capture the error message,

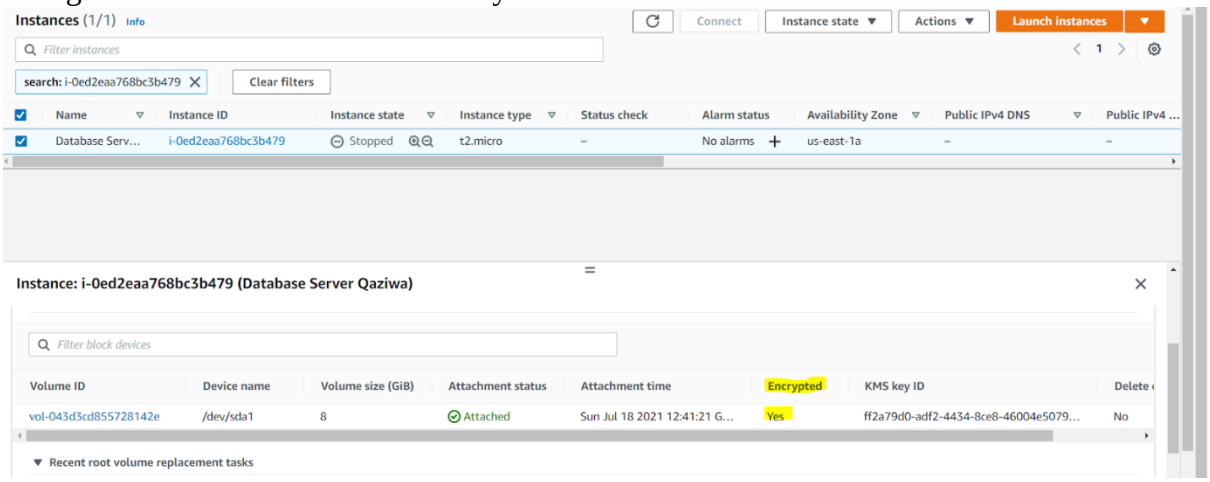


So now we need to attach the volume that we have created with the device name /dev/sda1. To do this, we go back to the volume section and locate the “Database-Encrypted EBS” volume. We click on it and click on action > click on attach volume > select the Qaziwa Database Instance and for the device, choose /dev/sda1 shown below.

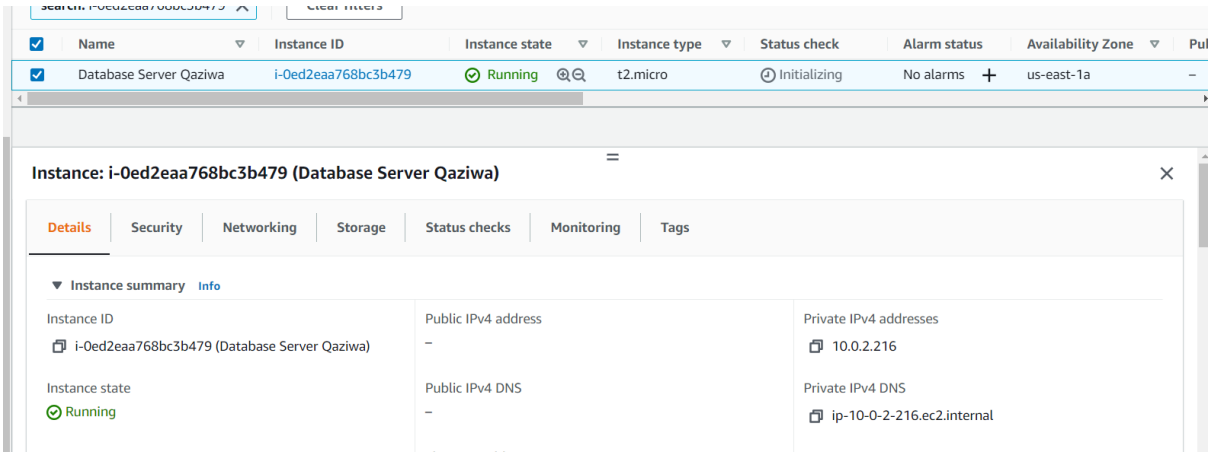


Then click on attach.

Once that is done, go back to the instance, and locate the Qaziwa Database Server and goto storage and see if the volume is already attached to the instance.



As shown above, we can see that we have successfully attached an encrypted volume into the instance and now is the time to test. We start the instance and see if it is started.



Now it is running, we have successfully detach the encrypted EBS volume and attach an encrypted EBS volume.

Creation of Monitoring system, Privileges and Encryption in AWS:

It is important to implement a monitoring system in the project and in our service as it can help to take note of the traffic, flag out any alerts with a specific rule and much more. As

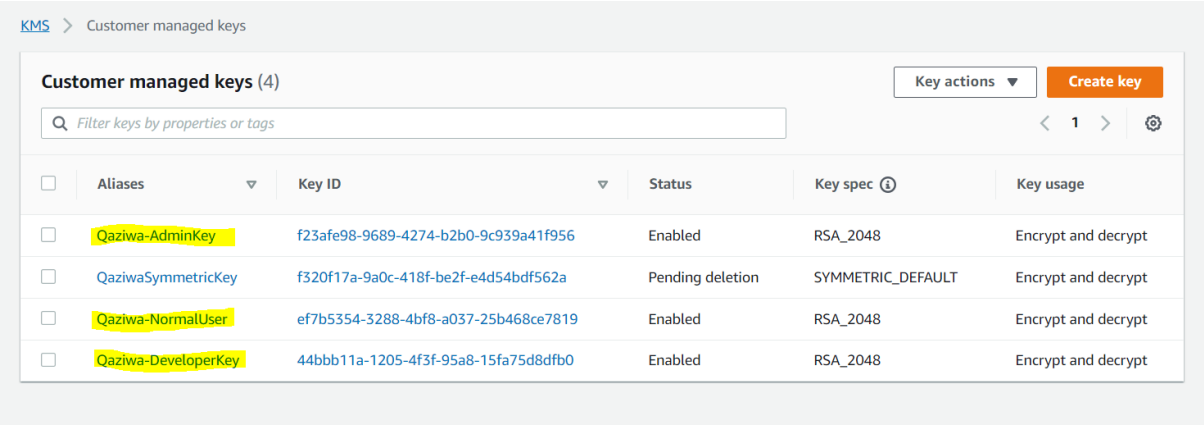
such help could help us in ensuring what time there might be heavy traffic, the CPU utilization, creation, traffic and much more.

Creation of KMS:

KMS is a centralized zone for storing all cryptographic keys to protect data. It is integrated with CloudTrail for monitoring of the key usage from the logs to allocate required resources across the network.

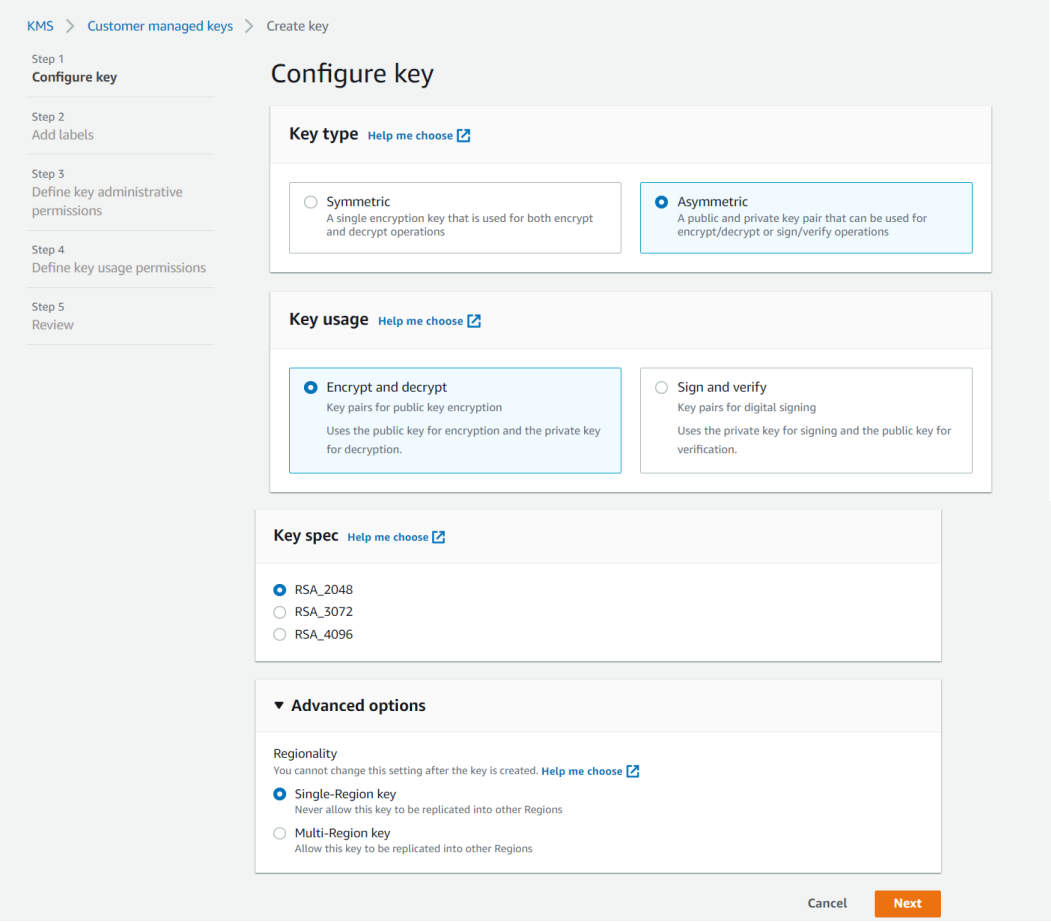
Despite having insufficient privileges, we will proceed and create KMS to show the rough idea of how the structure will be.

All Users Overview:

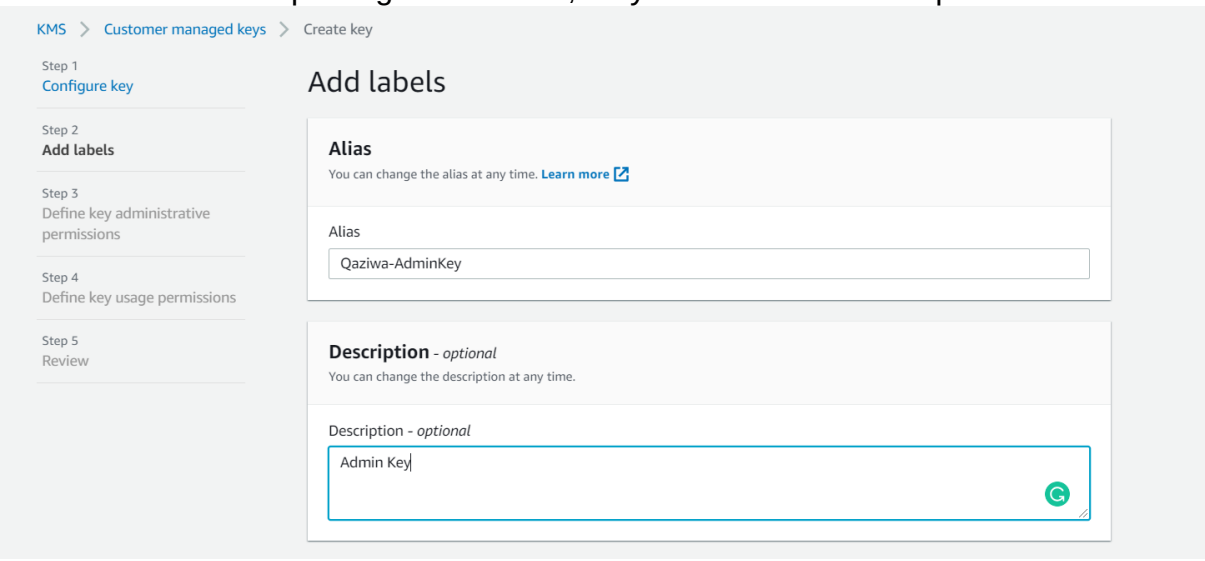


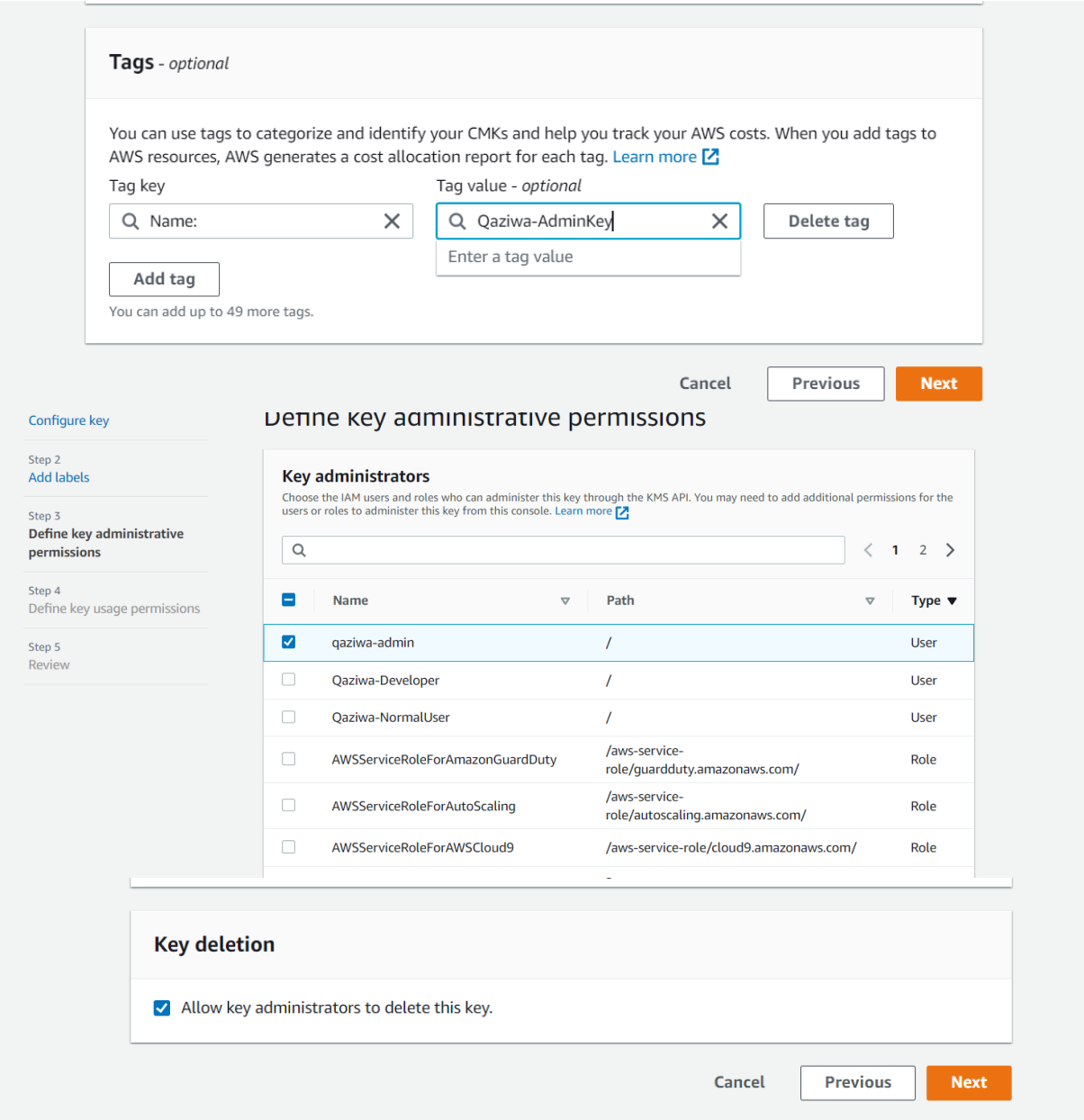
We will be creating 3 KMS keys for the different users. The first key called the “adminkey” will be allocated for the administrator account of the infrastructure. They have the most privileges for being able to control all users within the system. The second key called “normaluser” will be allocated for users accessing the website. Only a small amount of priveleges should be allocated as they should not be managing the system. The third key called the “developerkey” will be allocated for the designers and engineers. The privileges allocated should be between admin and normal users.

Admin Key:
An asymmetric key is assigned to allow only the receiver which is the admin to decrypt the message using their private key. Key usage will be solely for encrypting and decrypting with the use of the allocated keys. It will be a single-region key used within the system.

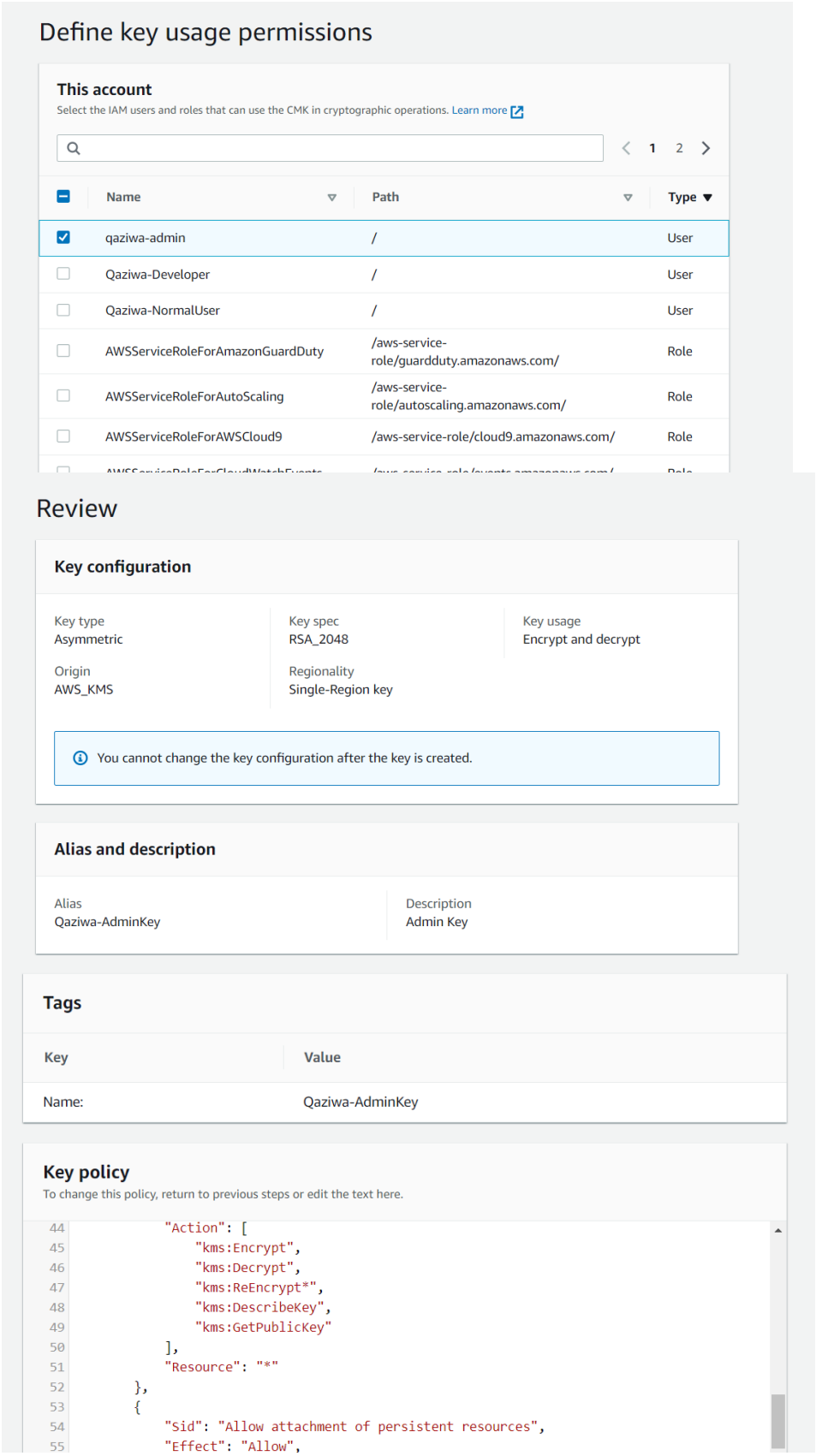


Admin has the most privilege. Therefore, they can delete when required.



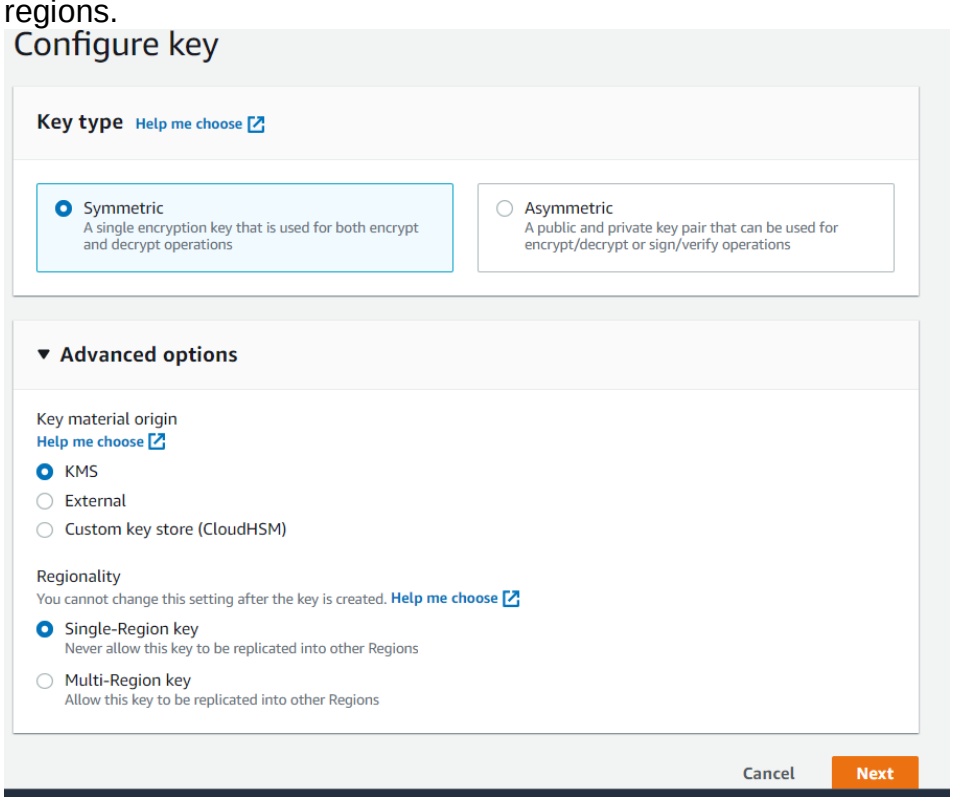


Since this key is allocated for administrator users, the admin is selected for the IAM role.

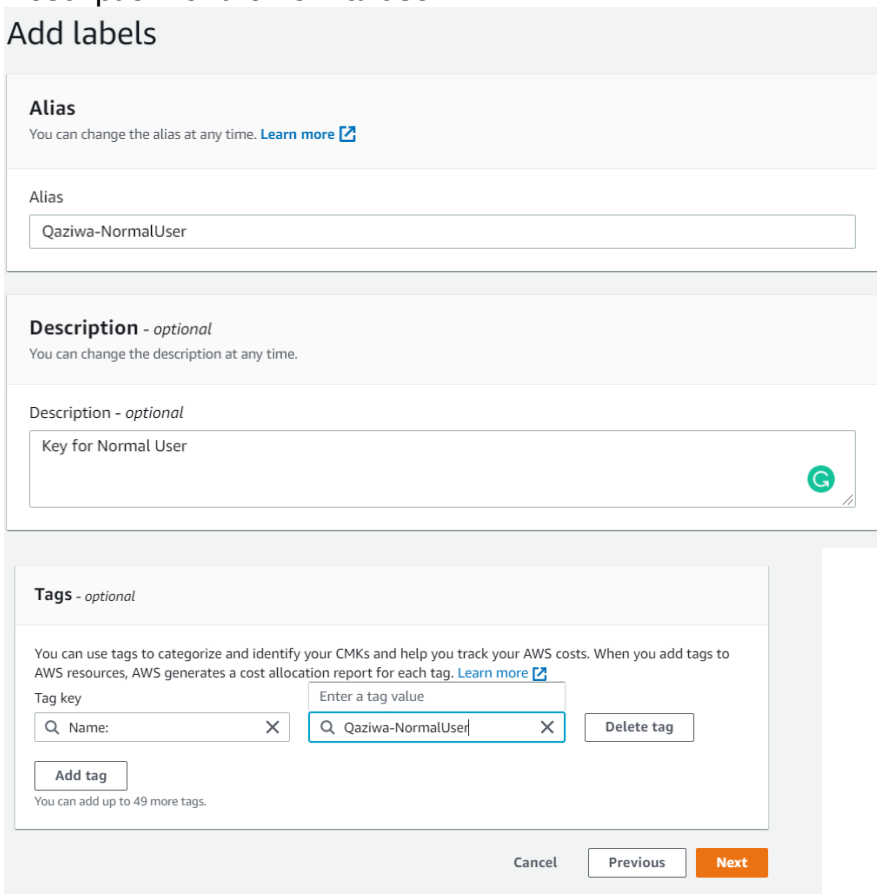


Customer Key:

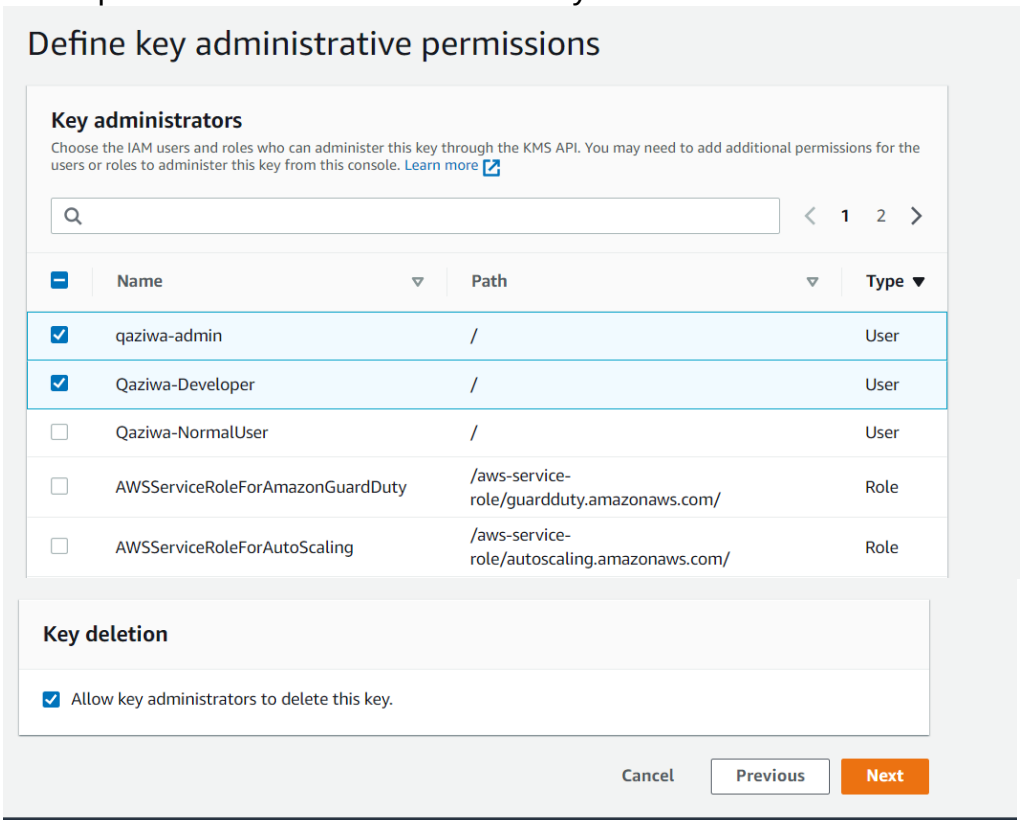
The symmetric key is used easier for encryption and decryption to the normal users. Keys are in a single region to prevent duplicated keys from lying around different regions.



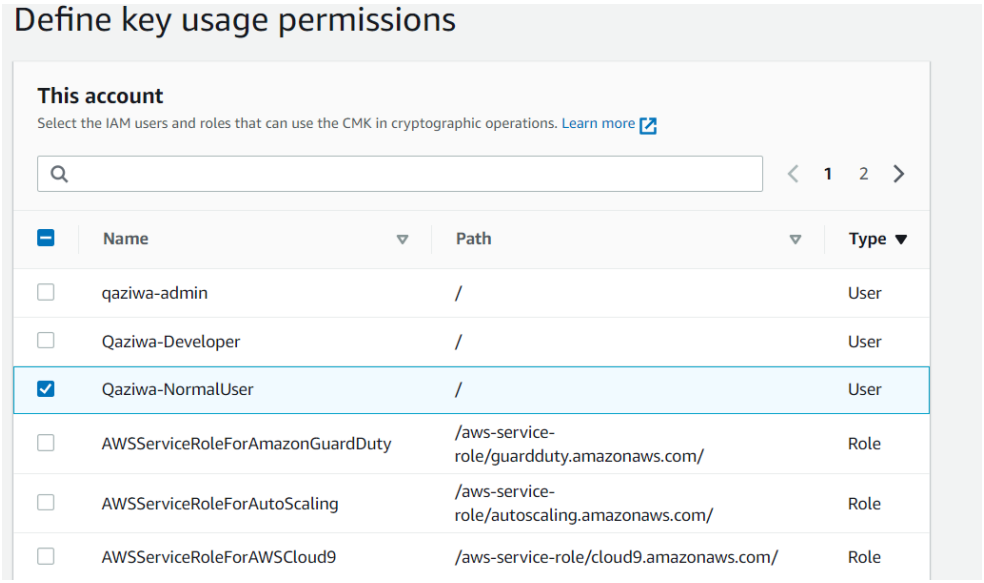
Description for the normal user.

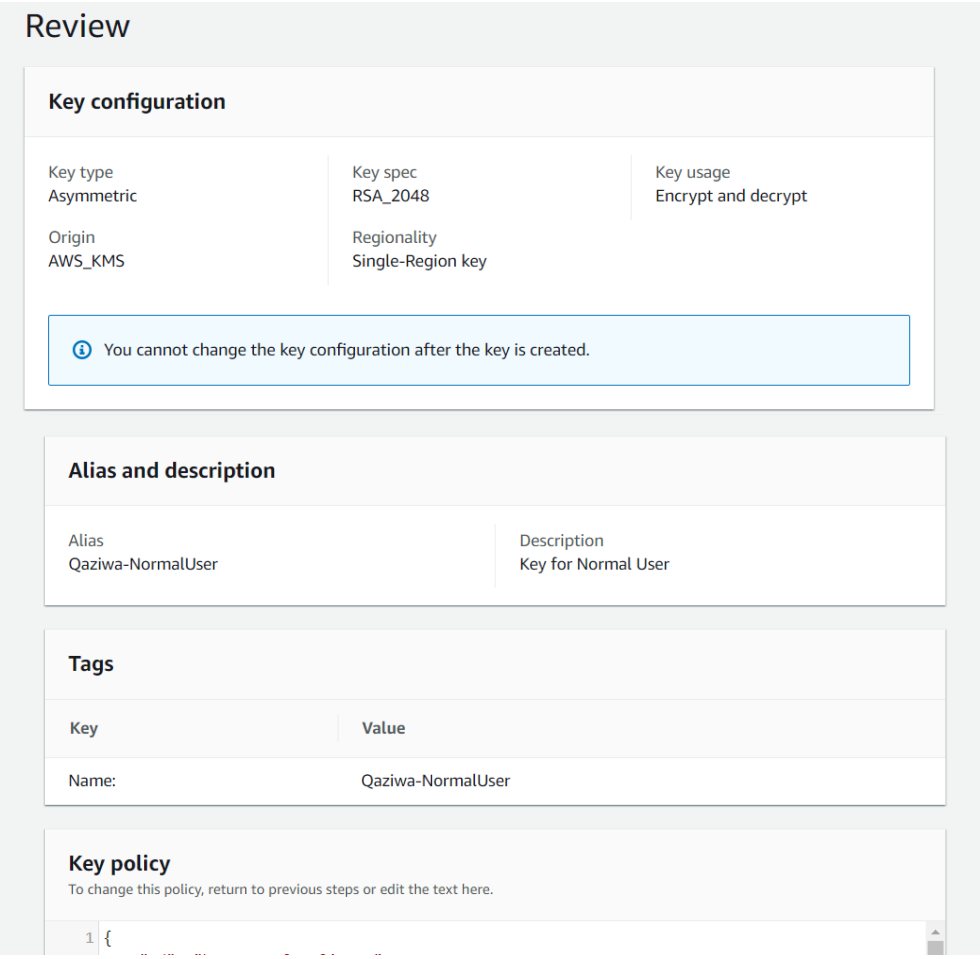


Normal users should not have the privilege to get hold of the key, hence admin and developer are allocated to control the key.

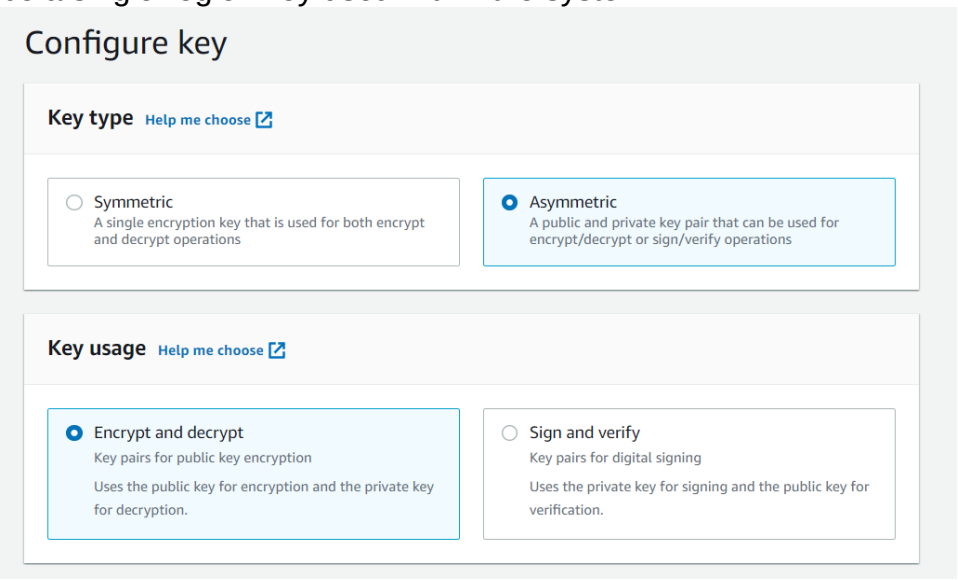


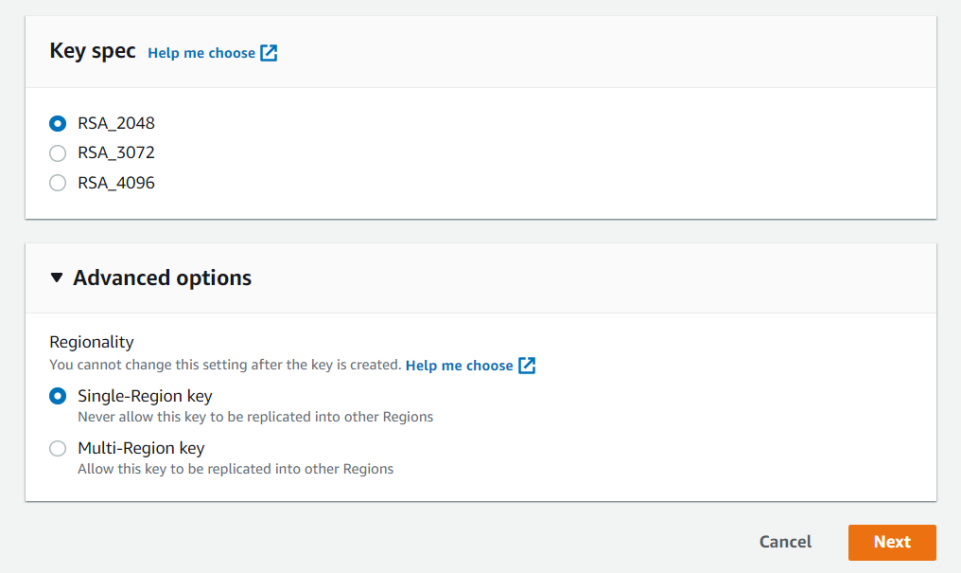
Since this key is allocated for customer/ normal users, the normal user is selected for the IAM role.



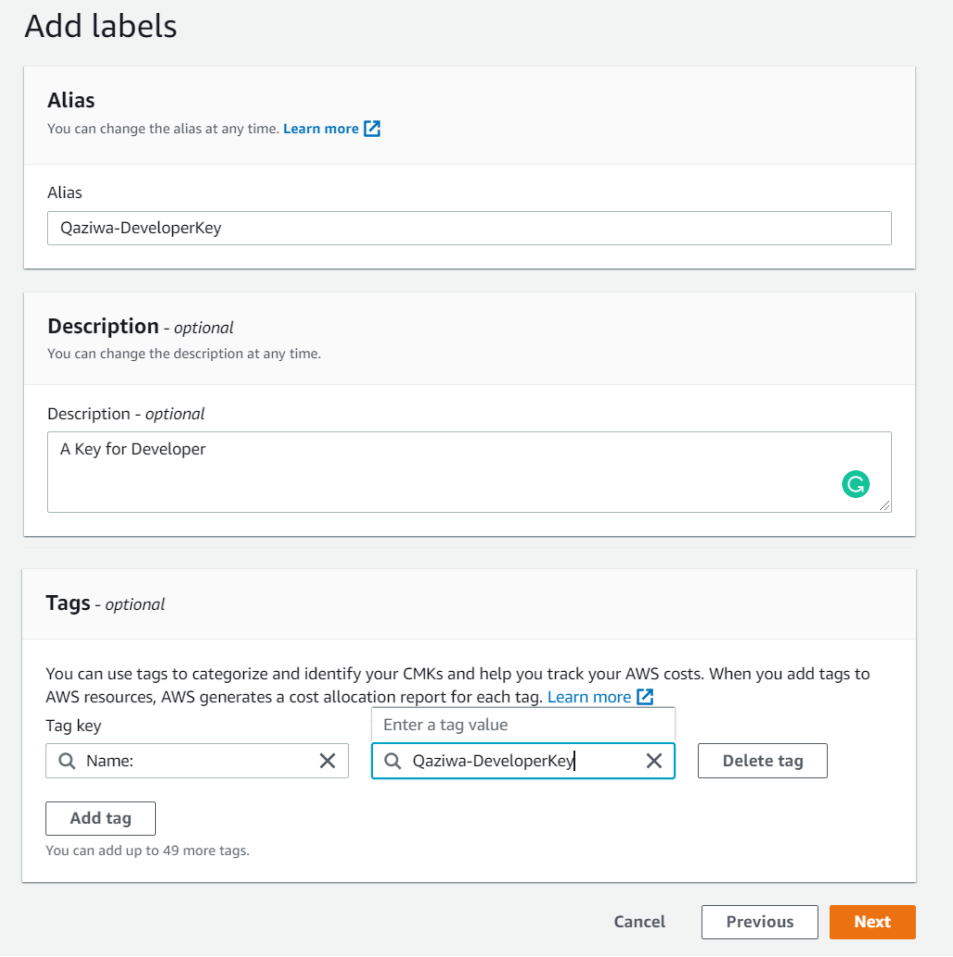


Developer Key:
An asymmetric key is assigned to allow only the receiver which is the developer to decrypt the message using their private key. Key usage will be solely for encrypting and decrypting with the use of the allocated keys. RSA2048 is enough for now. It will be a single-region key used within the system.

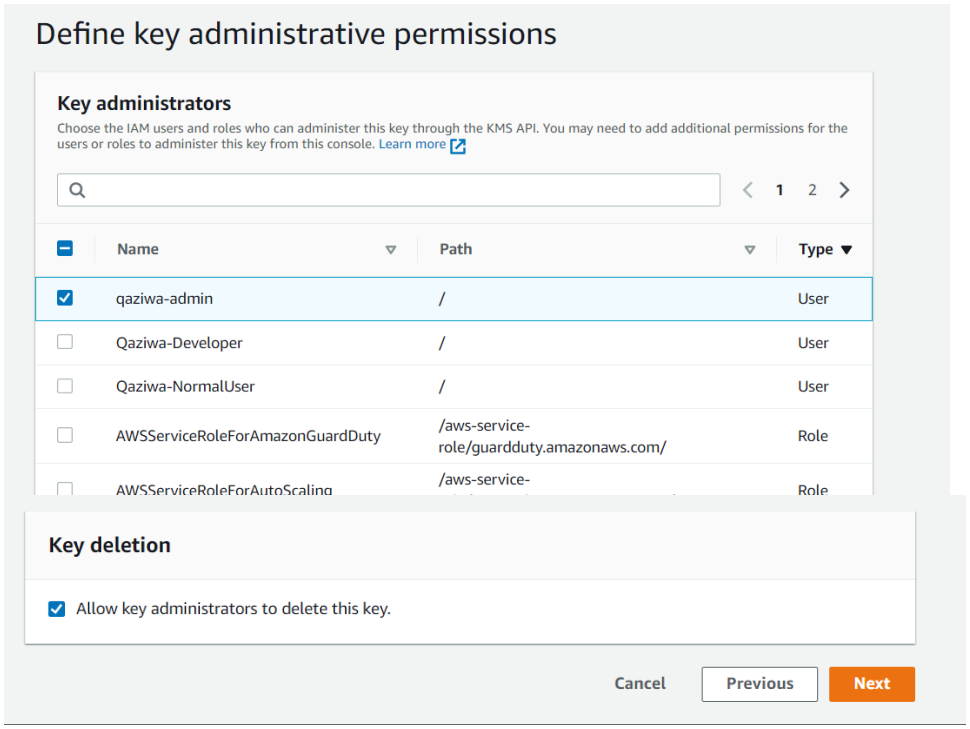




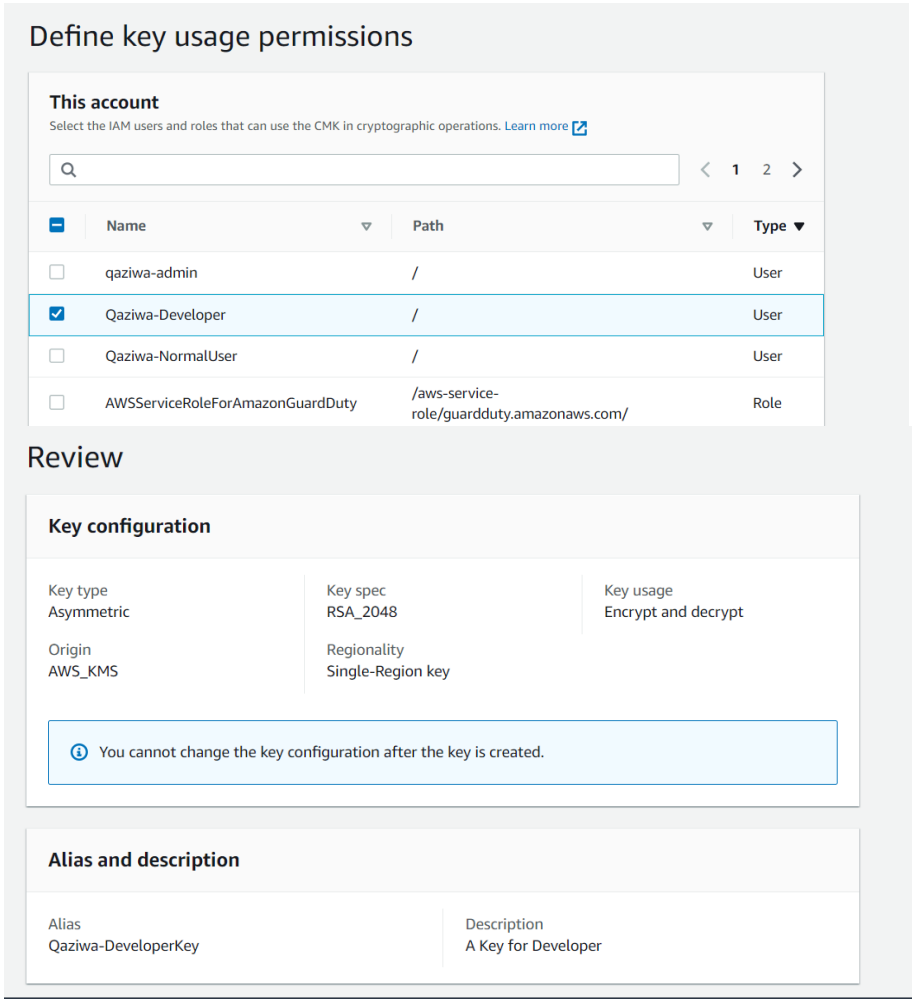
Description of developer key.

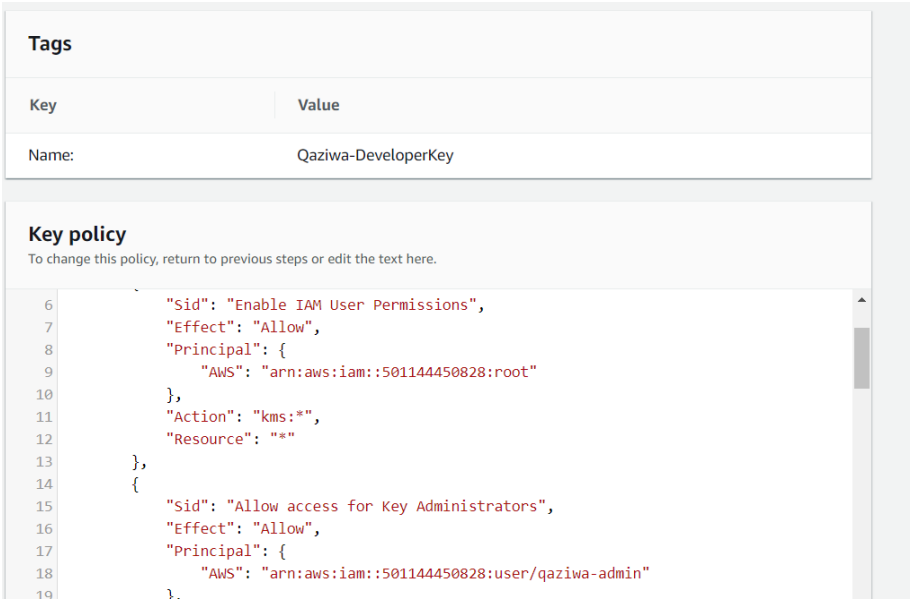


Since admin has the most privilege, the developer should not have the privilege to get hold of the key letting only the admin control the key.



Since this key is allocated for customer/ normal users, the normal user is selected for the IAM role.

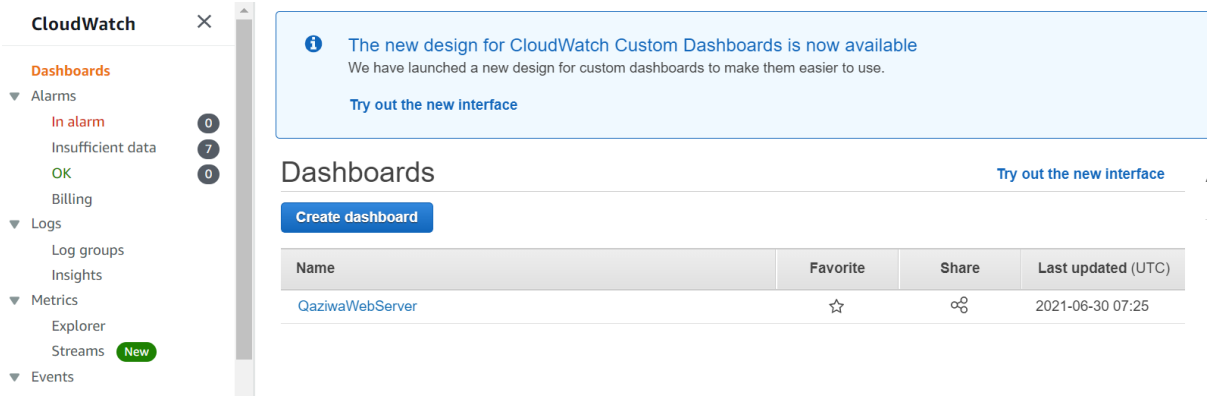




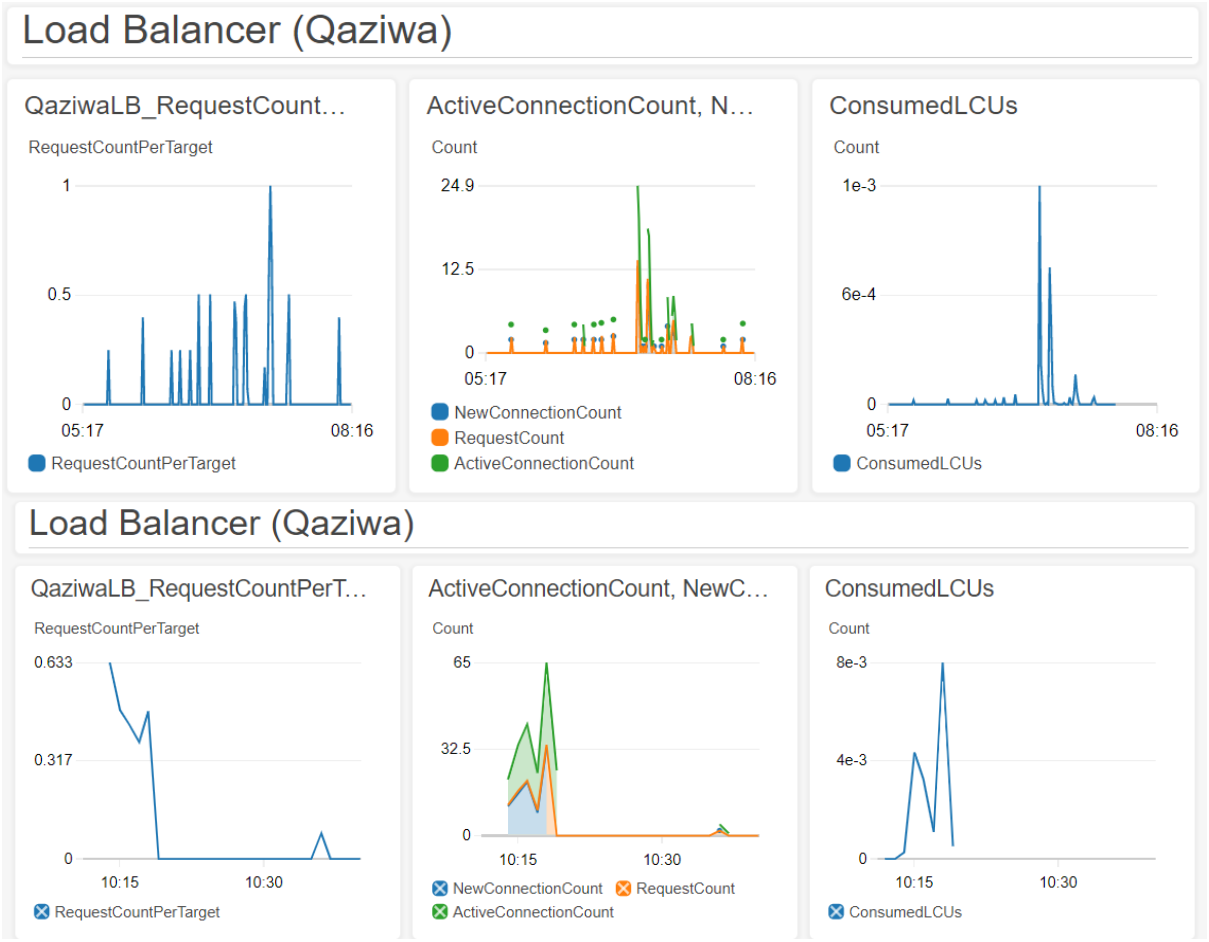
Since we finished our KMS configuration of the keys, now is to configure the CloudWatch.

Creation of CloudWatch:

AWS cloud watch will collect the monitoring and operational data by observing all the services used within the AWS classroom in the form of metrics and logs. To achieve this, we will create a dashboard called “QaziwaWebServer” to store all collected data from our cloud resource usage. The various widgets from the line graph, bar graph and even pie chart allow us to monitor the status differently. Alarms can also be set to trigger an alert when any abnormality is found, making troubleshooting easier.

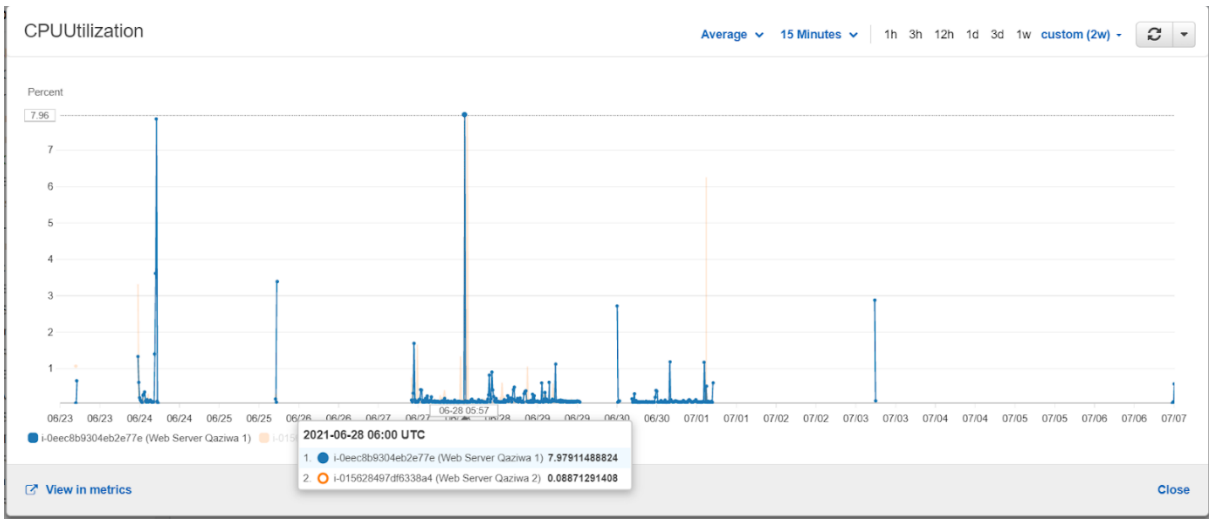


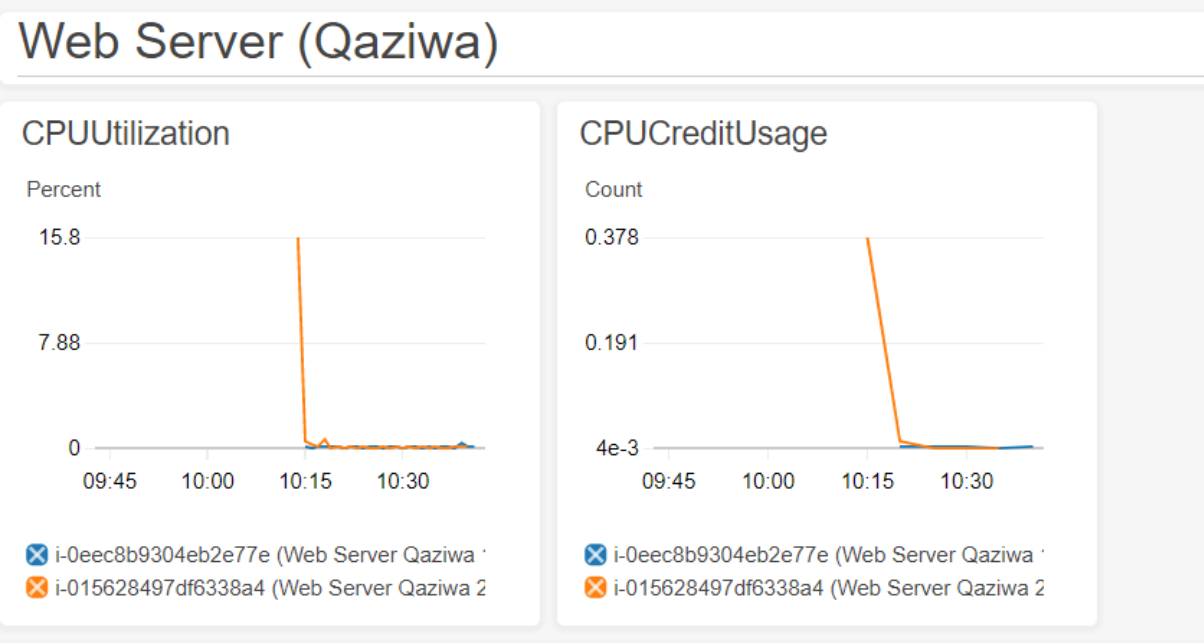
In our scenario, the cloud watch will be monitored in 4 sections - load balancer, database server, web server, and auto scaling. For this section, the 3 graphs below monitor the load balancer. The first graph represents the request count per target. When there are users using Qaziwa's website, the graph will increase accordingly to the demand as shown below. If there are no users, the graph will remain at 0. The second graph consists of new connection, request and active connection. The third graph is the consumption of the load balancing capacity unit (LCU).



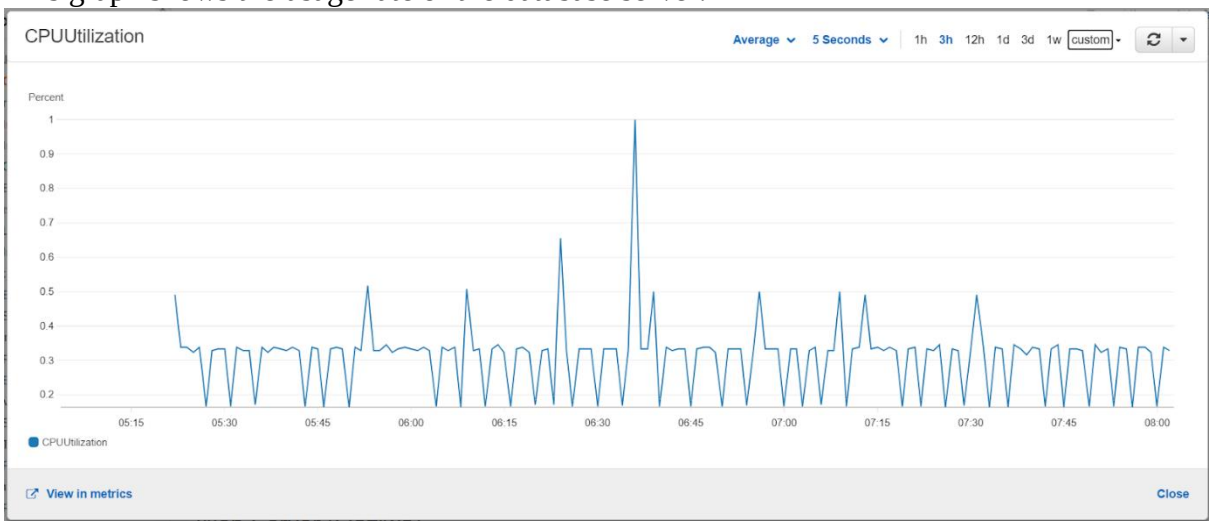
In this section, we will be showing how the monitoring works across different duration.

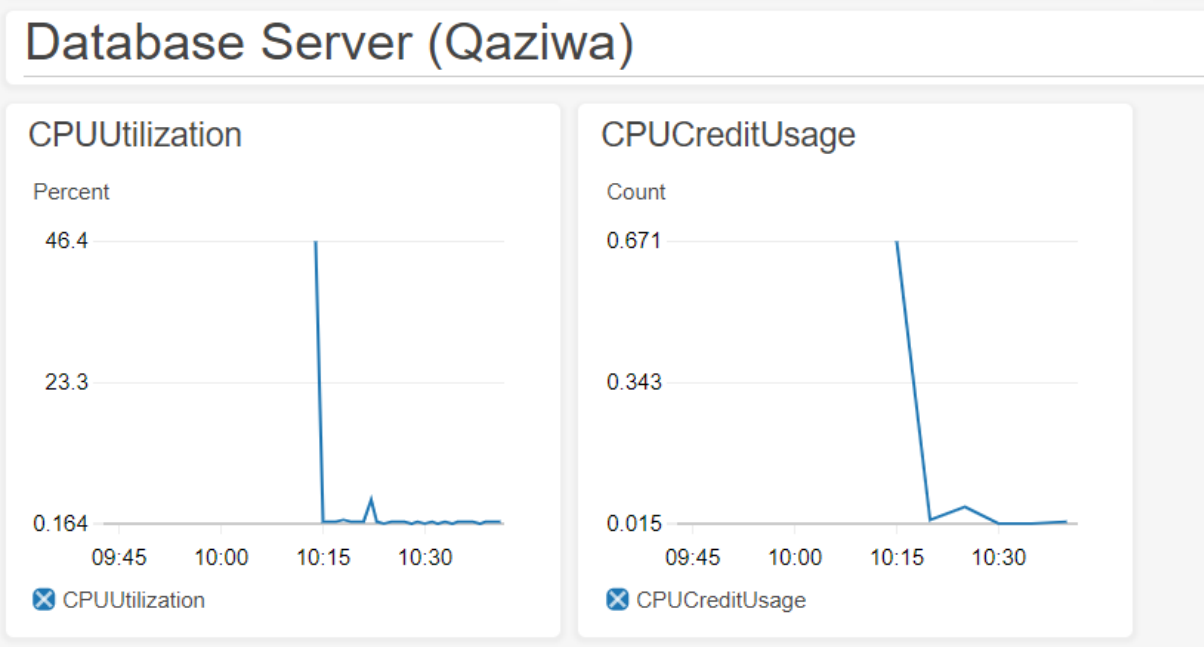
Web Server 1 CPU Data (2 weeks)
A surge will be observed when the demand is high. From this graph, it can be observed that web server 1 has a huge surge in demand whereas the demand remains low for web server 2. This is because user traffic will first be routed to the main server which is web server 1. When the CPU utilization met the limit (reach full capacity), any new users will be directed to web server 2 as part of load balancing.



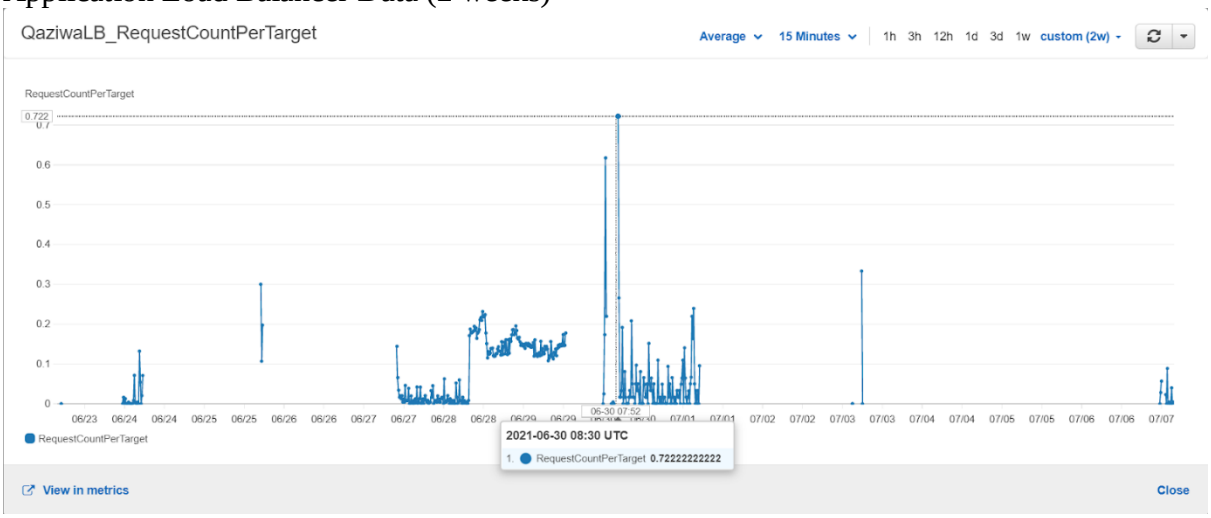


Database Server CPU Data (3 days)
This graph shows the usage rate of the database server.

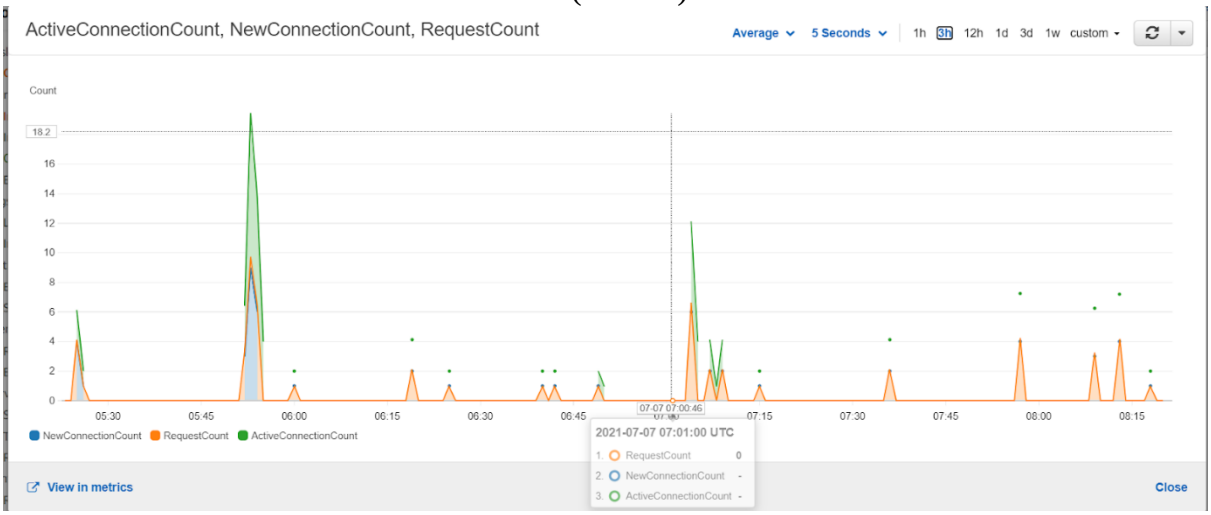




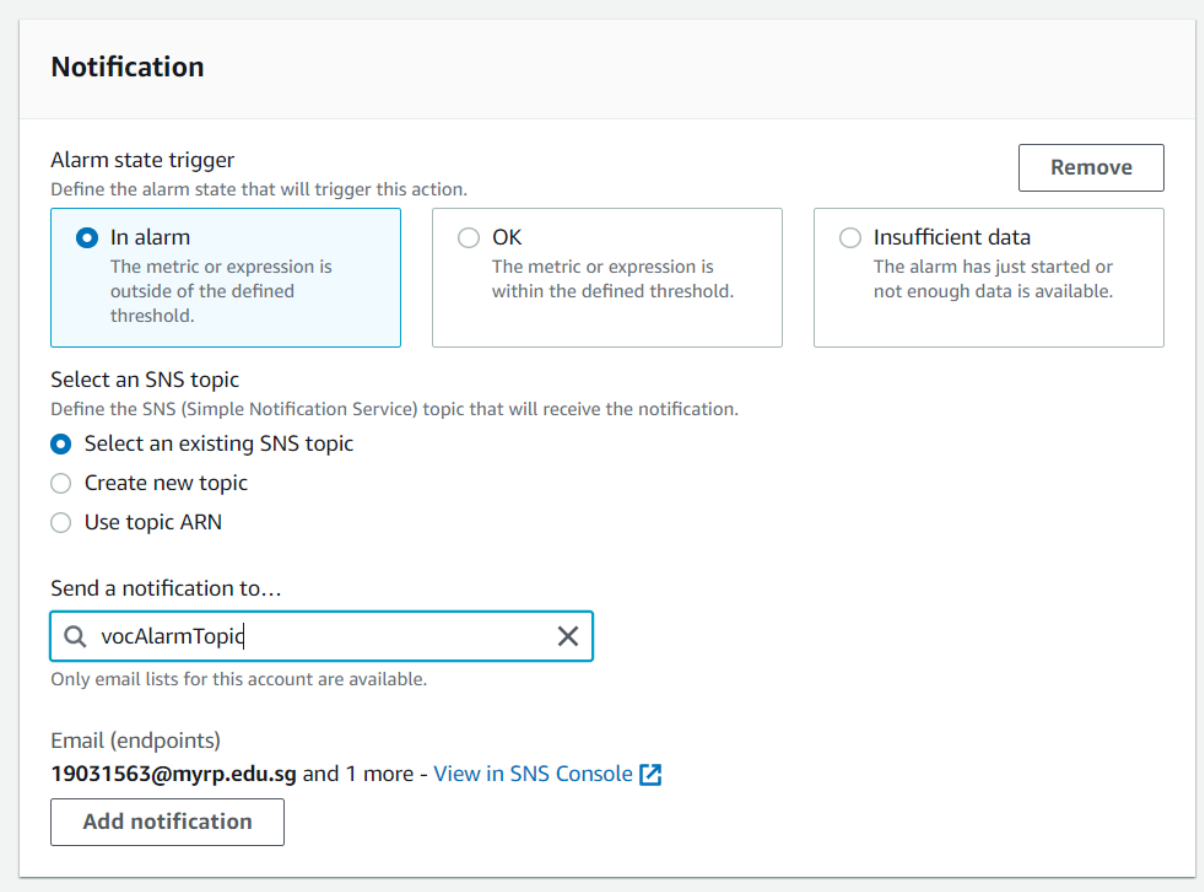
Application Load Balancer Data (2 weeks)



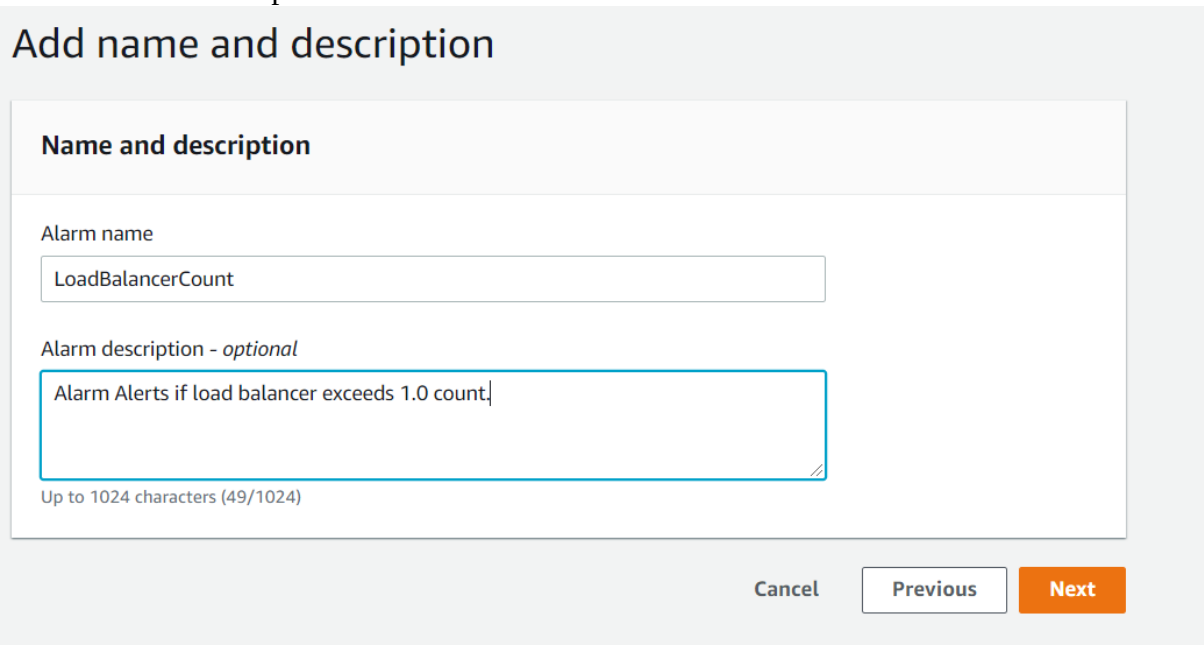
Active connection Count for Load Balancer (3 hours)



Notification sent via email (Wafi)



Load Balancer description



Alarms

Details of an actual alarm are shown below. After the alarm has triggered, it will notify the email address listed with a full summary of the alarm. With this alarm, it allows us to conduct troubleshooting accordingly.

You are receiving this email because your Amazon CloudWatch Alarm "Test" in the US East (N. Virginia) region has entered the ALARM state, because "Threshold Crossed: 1 out of the last 1 datapoints [3.0 (01/07/21 07:21:00)] was greater than or equal to the threshold (2.0) (minimum 1 datapoint for OK -> ALARM transition)." at "Thursday 01 July, 2021 07:23:15 UTC".

Alarm Details:
- Name: Test
- Description:
- State Change: OK -> ALARM
- Reason for State Change: Threshold Crossed: 1 out of the last 1 datapoints [3.0 (01/07/21 07:21:00)] was greater than or equal to the threshold (2.0) (minimum 1 datapoint for OK -> ALARM transition).
- Timestamp: Thursday 01 July, 2021 07:23:15 UTC
- AWS Account: 501144450828
- Alarm Arn: arn:aws:cloudwatch:us-east-1:501144450828:alarm:Test

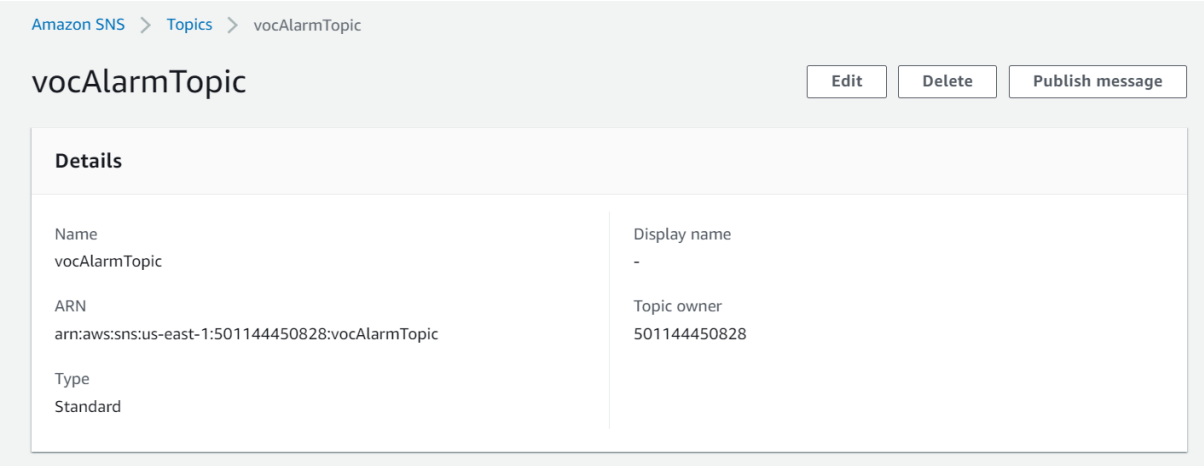
Threshold:
- The alarm is in the ALARM state when the metric is GreaterThanOrEqualToThreshold 2.0 for 60 seconds.

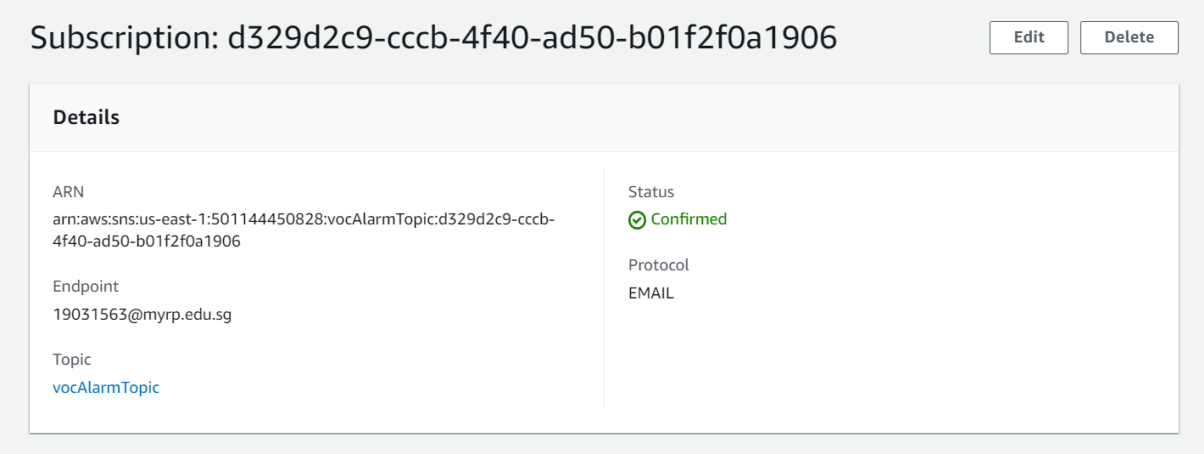
Monitored Metric:
- MetricNamespace: AWS/ApplicationELB
- MetricName: RequestCountPerTarget
- Dimensions: [TargetGroup = targetgroup/QaziwaTargetGroup/b99f92eb233fa95a]
- Period: 60 seconds
- Statistic: Sum
- Unit: not specified
- TreatMissingData: missing

State Change Actions:
- OK:
- ALARM: [arn:aws:sns:us-east-1:501144450828:vocAlarmTopic]
- INSUFFICIENT_DATA:

- Simple Notification Service (SNS)

This service is mainly based on CloudWatch. In our configuration, we will conduct a test to see whether a notification will be delivered when an abnormality is found. An endpoint needs to be specified for this to work, so we assigned Wafi's email address as a target. A notification is sent which means the alarm works successfully (example of email notification under Cloud Watch section above and Cloud Trail section below)



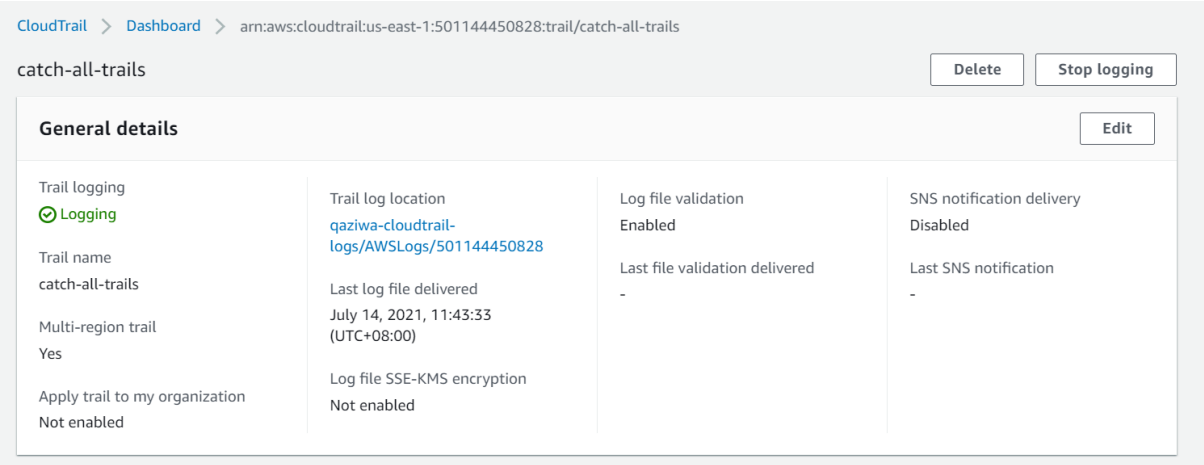


Since we already finished on our CloudTrail, now is the time to configure the CloudTrail.

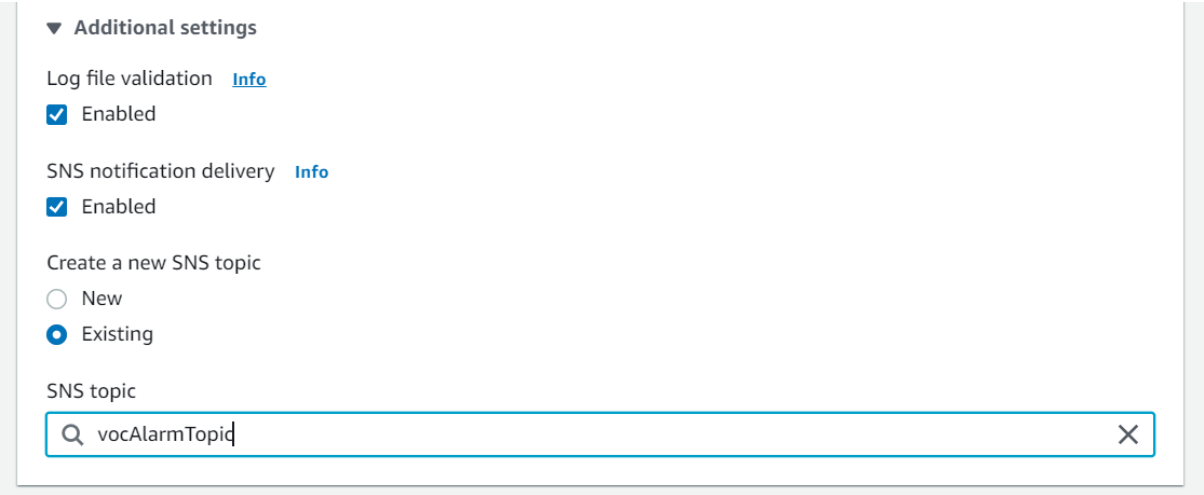
Creation of CloudTrail:

While Cloud Watch focuses on mainly the activities of AWS services and resources, Cloud Trail service will record the activities made inside the AWS infrastructure and deliver log files to the S3 bucket. Event history will reflect actions including using command-line tools and different AWS services.

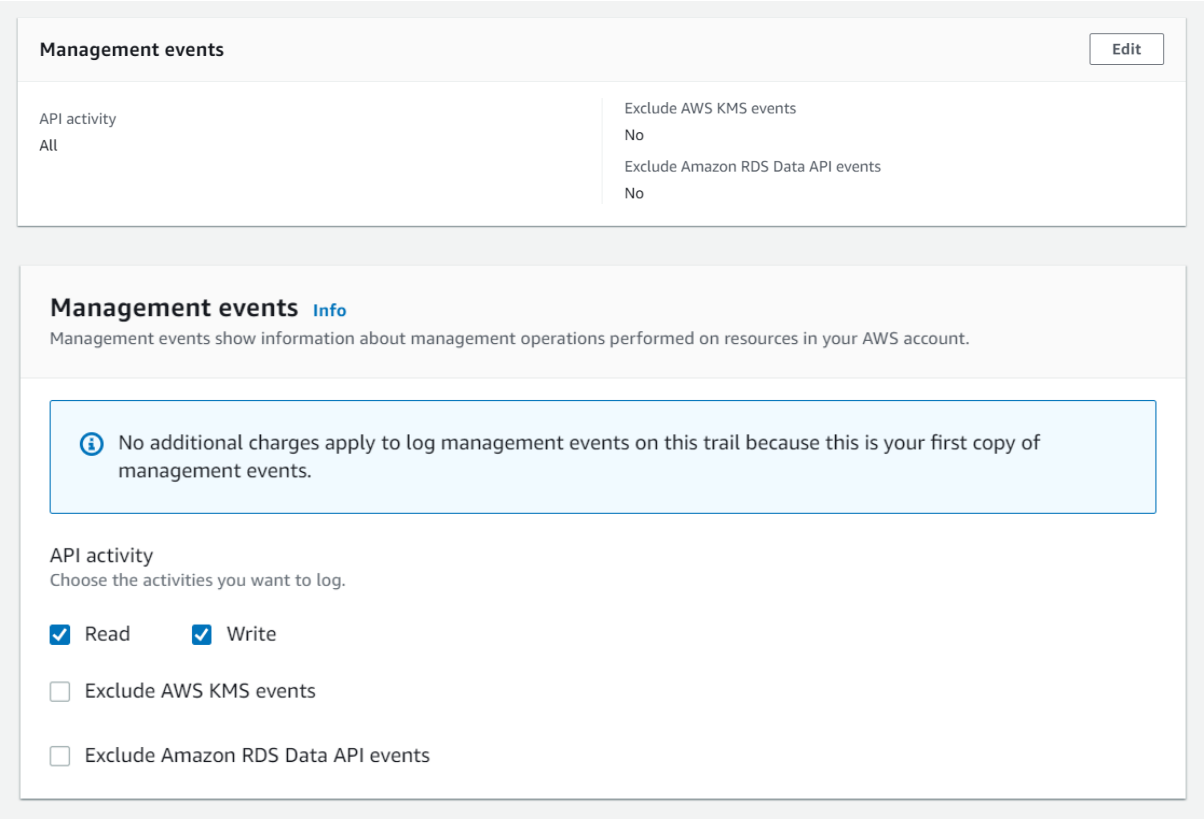
The cloud trail we created is called “catch-all-trails”. We also created a new S3 bucket “qaziwa-cloudtrail-logs” to store the logs. The log file validation is also enabled to ensure integrity where the contents inside the log file have not been changed. When enabled, a hash will be created for every log file delivered. In an interval for an hour, a digest file that contains references to the log file with hash is created. The digest file is associated with the cloud trail and will be delivered to other accounts when required.



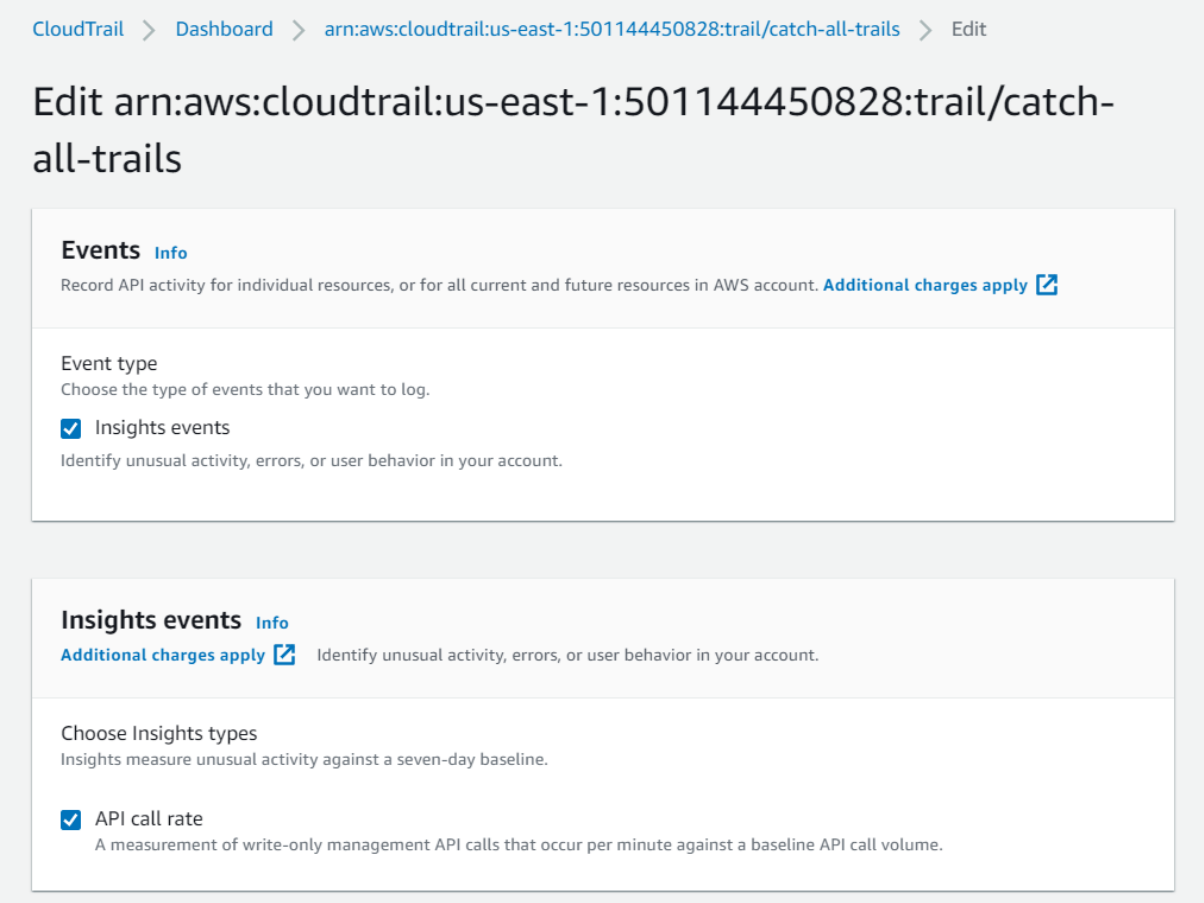
SNS will also be enabled where alarms will notify the listed email address (Wafi’s account) using the existing SNS topic “vocAlarmTopic”.



Leave management events as default



Enable insights events by enabling API call rate which will record the number of requests to the network.



Email notification (Wafi) - Example

Since we have a notification set to vocalarm topic, wafi's email is registered under that group and therefore would receive notifications regarding the created cloud trail if any logs are stored in the s3 bucket. An email would be sent for every log that is transferred to the S3 bucket therefore the notification traffic would be quite heavy as it is estimated that the logs are in the count of thousands.

From: AWS Notifications
Sent: Wednesday, 14 July 2021 11:08 AM
To: MUHD WAFIYUDDIN BIN ABDUL RAHMAN
Subject: AWS Notification Message

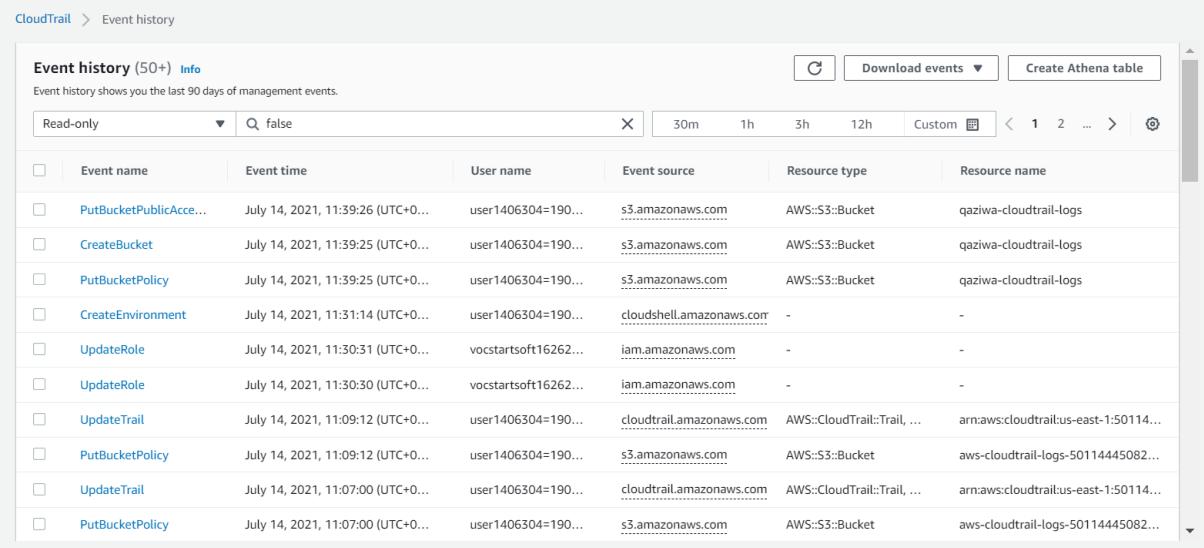
CAUTION: This email originated from outside the organization. Do not click links or open attachments unless you recognize the sender and know the content is safe.

{"s3Bucket":"aws-cloudtrail-logs-501144450828-1ec5e12a","s3ObjectKey":["AWSLogs/501144450828/CloudTrail/eu-west-1/2021/07/14/501144450828_CloudTrail_eu-west-1_20210714T0305Z_JYWK5vMLjL6mXYdx.json.gz"]}

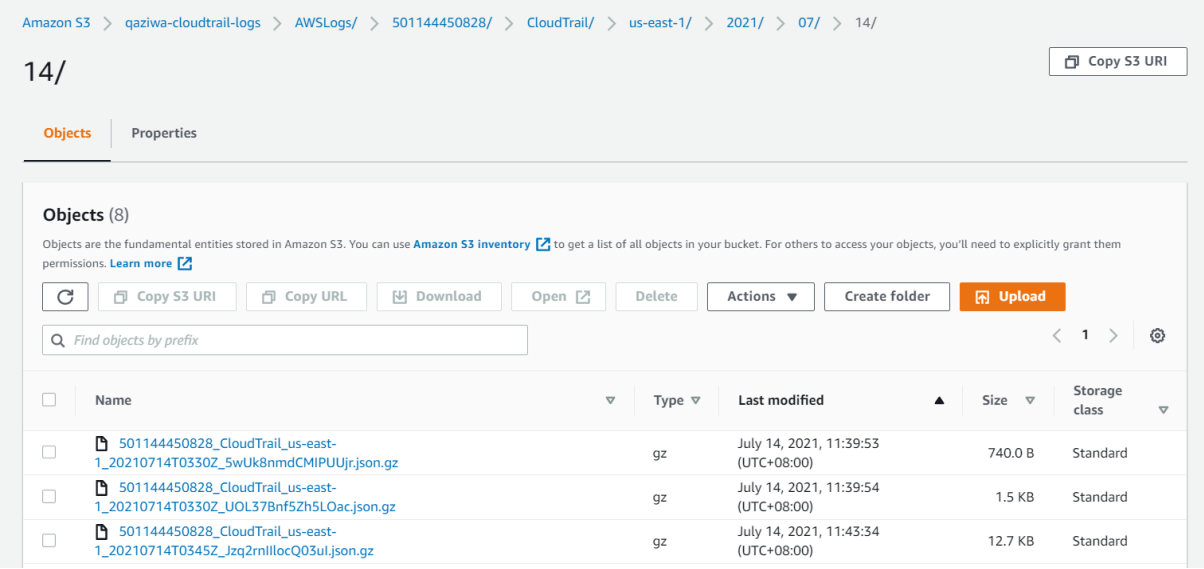
--
If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
https://apc01.safelinks.protection.outlook.com/?url=https%3A%2F%2Fsns.us-east-1.amazonaws.com%2Funsubscribe.html%3FSubscriptionArn%3Darn%3Aaws%3Asns%3Aus-east-1%3A501144450828%3AvocAlarmTopic%3Ad329d2c9-cccb-4f40-ad50-b01f2f0a1906%26Endpoint%3D19031563%40myrp.edu.sg&data=04%7C01%7C19031563%40myrp.edu.sg%7C3edd5e6956044ede9b3e08d94674b398%7Cf688b0d079f040a4864435fcdee9d0f3%7C0%7C0%7C637618289320109428%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C2000&sdata=OH%2F8dV%2FyjTu3Ayrxirf9G6O5Qe359oQ3rfo6toZVOd4%3D&reserved=0

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at
https://apc01.safelinks.protection.outlook.com/?url=https%3A%2F%2Faws.amazon.com%2Fsupport&data=04%7C01%7C19031563%40myrp.edu.sg%7C3edd5e6956044ede9b3e08d94674b398%7Cf688b0d079f040a4864435fcdee9d0f3%7C0%7C0%7C637618289320109428%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C2000&sdata=Cbbl%2BBqSG3MX7dA92jWN7U7d6KA%2Fv%2F2VH%2BcVujtrgcg%3D&reserved=0

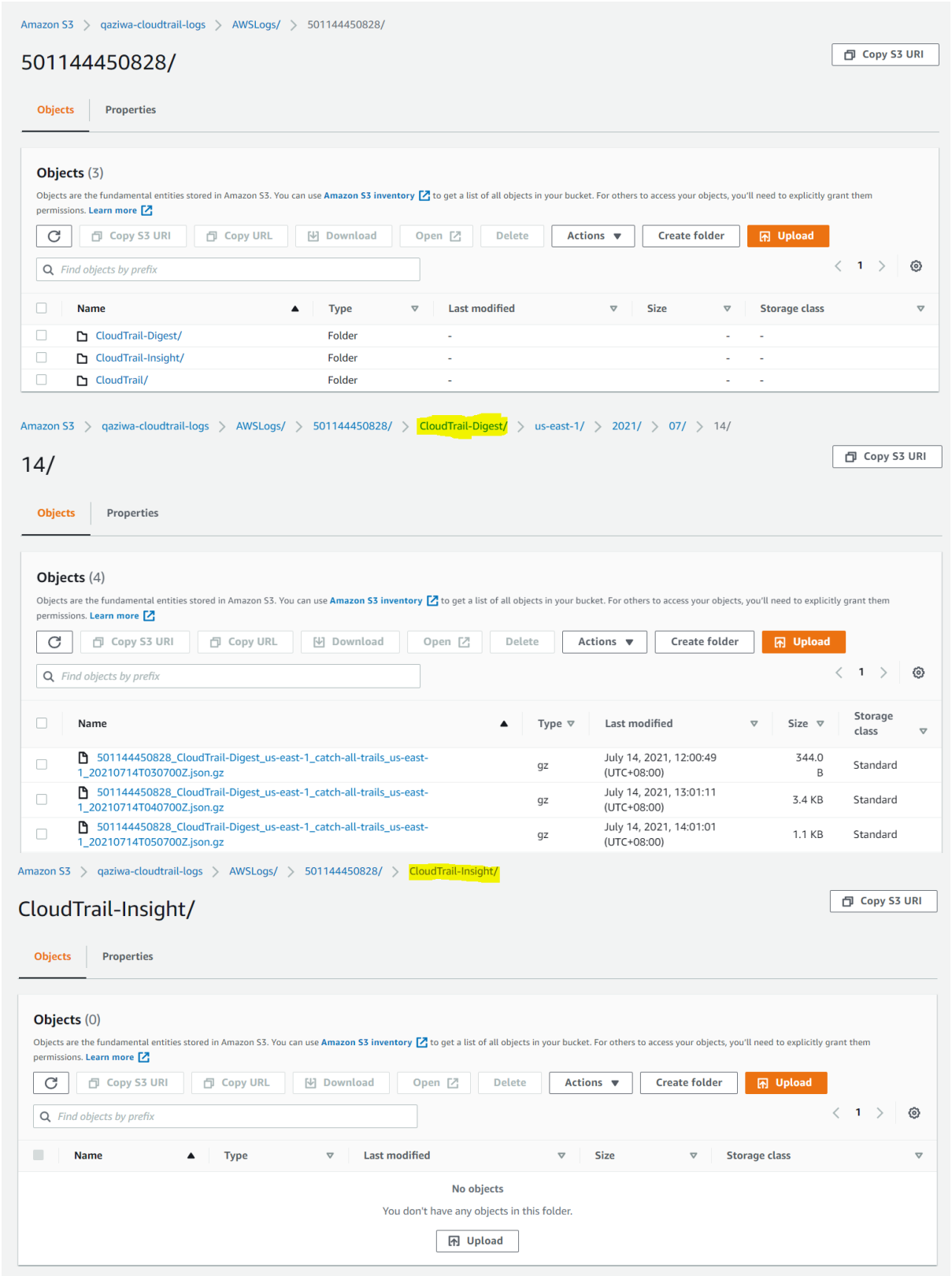
The event history shows all the activities achieved within this AWS account. It can be filtered accordingly from different actions done (e.g. create, modify) or across different time frames. The record of events will last for 90 days.



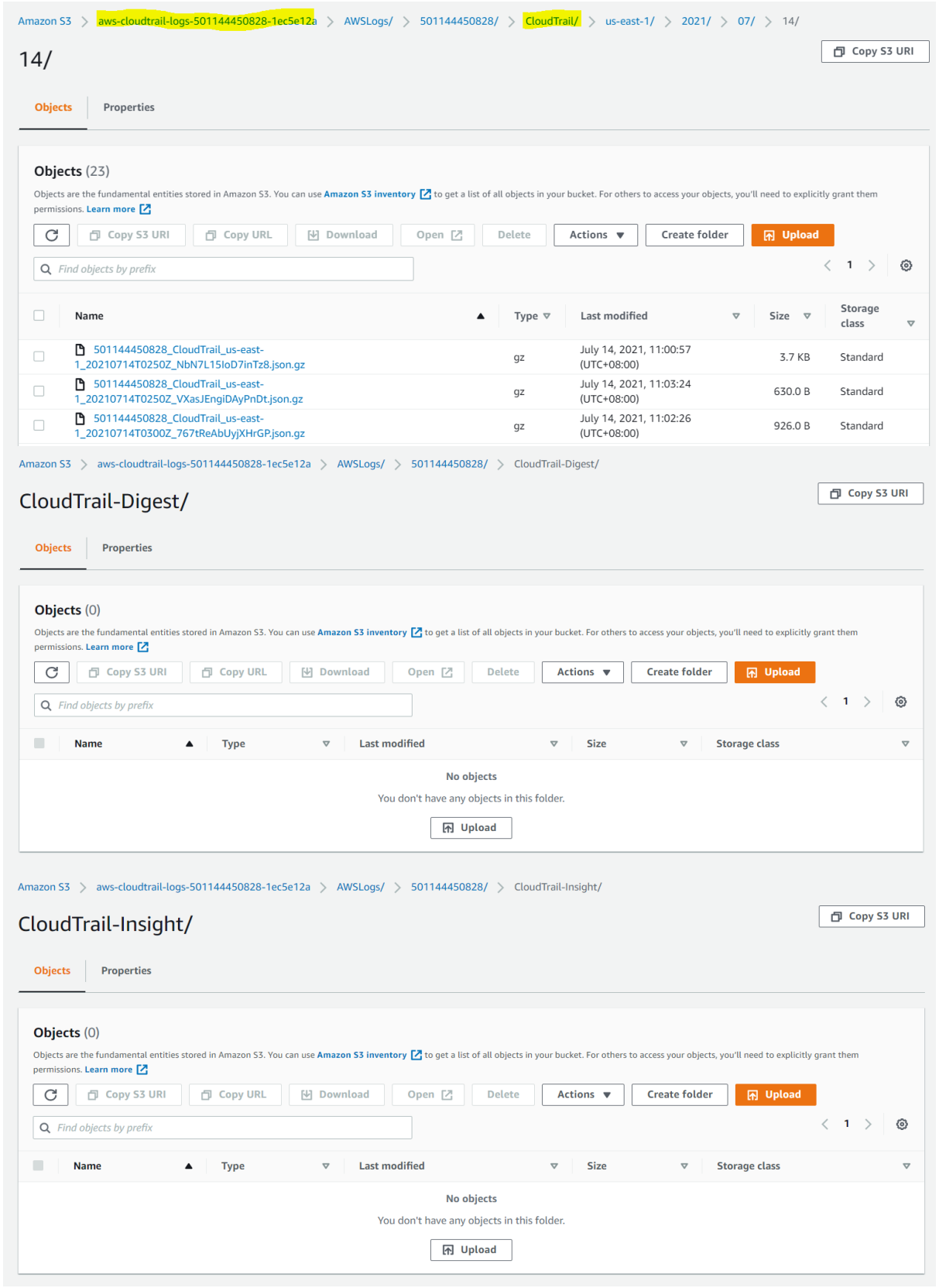
From the newly created S3 bucket, we can see that the logs are stored in the following screenshot.



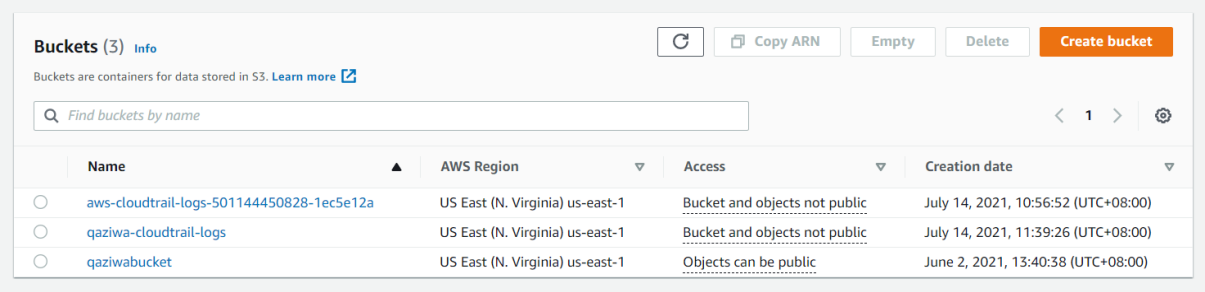
3 objects were created. All logs are stored in “CloudTrail/”.



Default bucket. All the logs would immediately be sent to the default bucket if a new bucket is not created.



Reference Summary of buckets

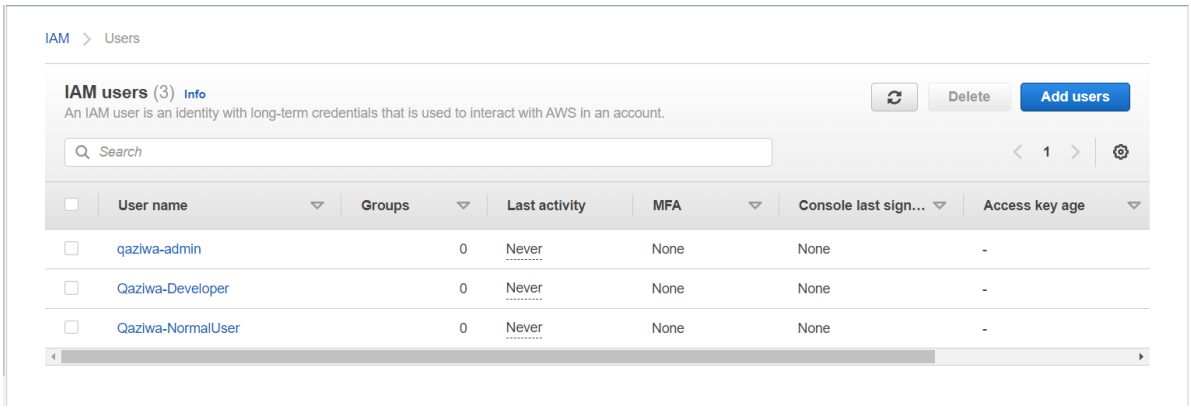


Since we have finished configuring the CloudTrail, now we can continue to create the IAM roles and groups.

Creation of IAMs:

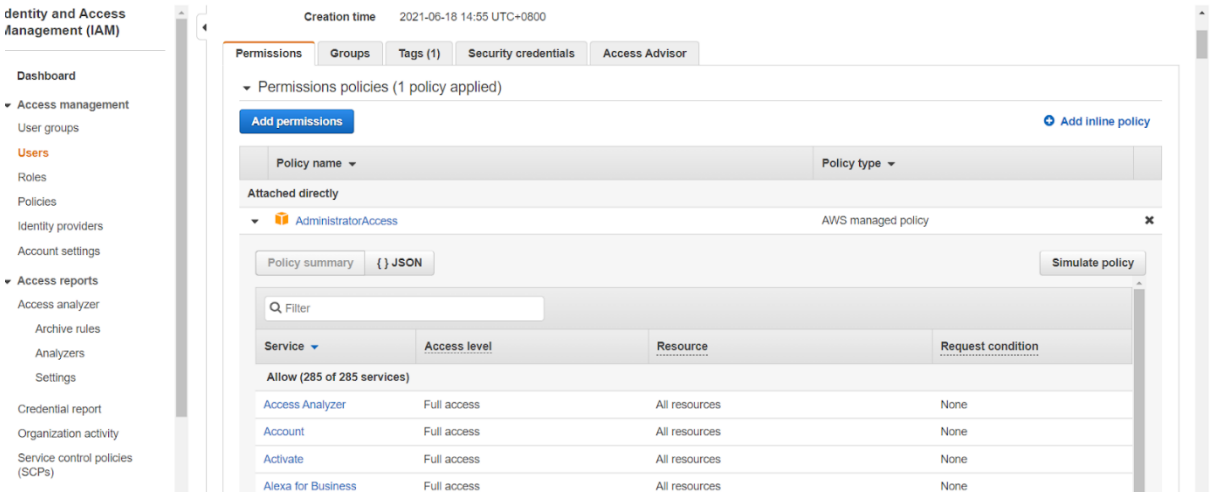
With all the instances and security configurations put into place, we would now need to create restrictions for the users that would be accessing the AWS cloud. Taking a look at the bottom screenshot the end result would be 3 created users configured for access to the cloud each with varying permissions set.

All Users:



The first user that we would configure for use would be the Admin user. This user would have the most privileges out of the three as this is essentially the placeholder for the root account. Because of this, in the permission policy, we would set admin to have AdministratorAccess. This allows the account to have access to all the AWS tools like the root account has.

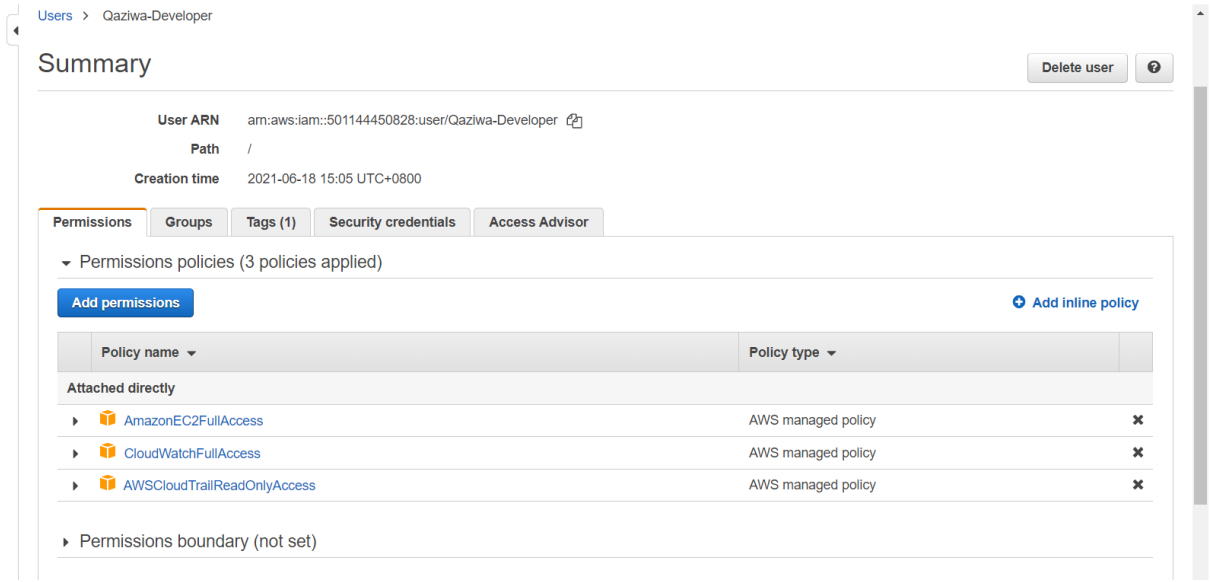
Qaziwa-Admin



This user focuses on creating and managing the instances that are on the cloud. This user is the developer and should be allowed to have read and write on services relating to the instance. We would set it for AmazonEC2FullAccess, CloudwatchFullAccess, CloudTrailReadOnly.

EC2FullAccess: Any function regarding the instance under EC2 is given read and write access.
CloudwatchFullAccess: User is able to read the montiring dashboard and add their own variables to monitor or edit the value type it returns.

Qaziwa-Developer



Qaziwa-NormalUser
Normal user would the user that uses the cloud to host their services like the clients for example. Because of this they are given the lowest privilege only allowing the ability to connect and view the instance and list the instance.

Set user details

You can add multiple users at once with the same access type and permissions. Learn more

User name* Qaziwa-NormalUser

Add another user

Select AWS access type

Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. Learn more

Access type* ☐ **Programmatic access**
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.

☑ **AWS Management Console access**
Enables a **password** that allows users to sign-in to the AWS Management Console.

Console password* ◉ Autogenerated password
○ Custom password

Select AWS access type

Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. Learn more

Access type* ☐ **Programmatic access**
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.

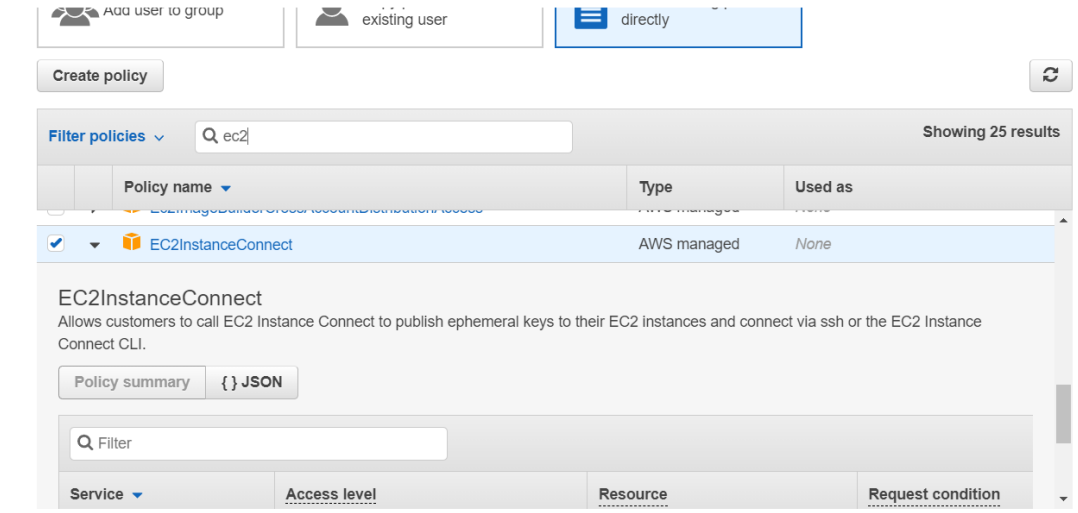
☑ **AWS Management Console access**
Enables a **password** that allows users to sign-in to the AWS Management Console.

Console password* ◉ Autogenerated password
○ Custom password

Require password reset ☑ User must create a new password at next sign-in
Users automatically get the IAMUserChangePassword policy to allow them to change their own password.

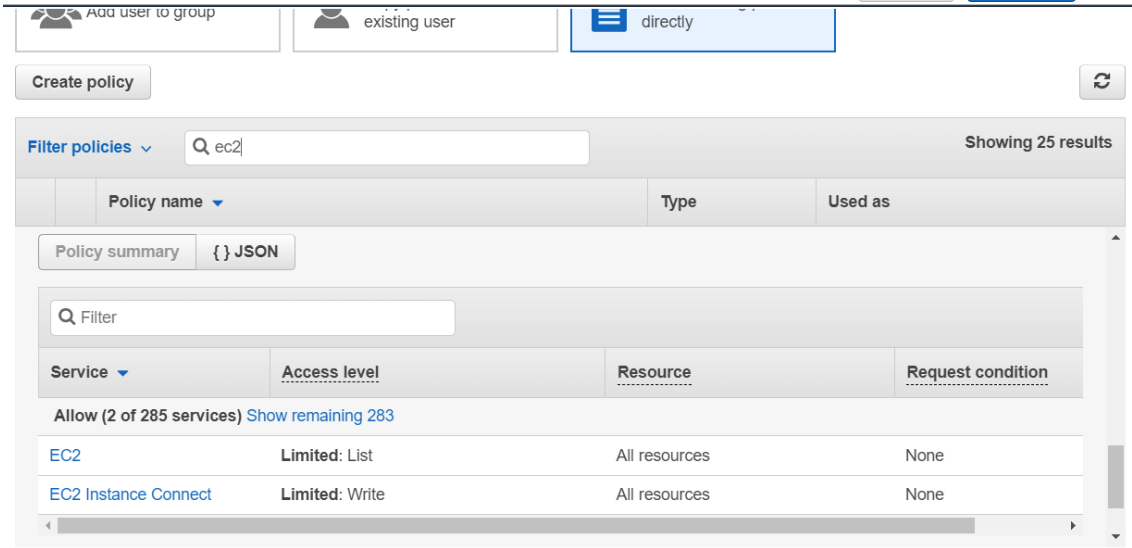
* Required

Cancel Next: Permissions



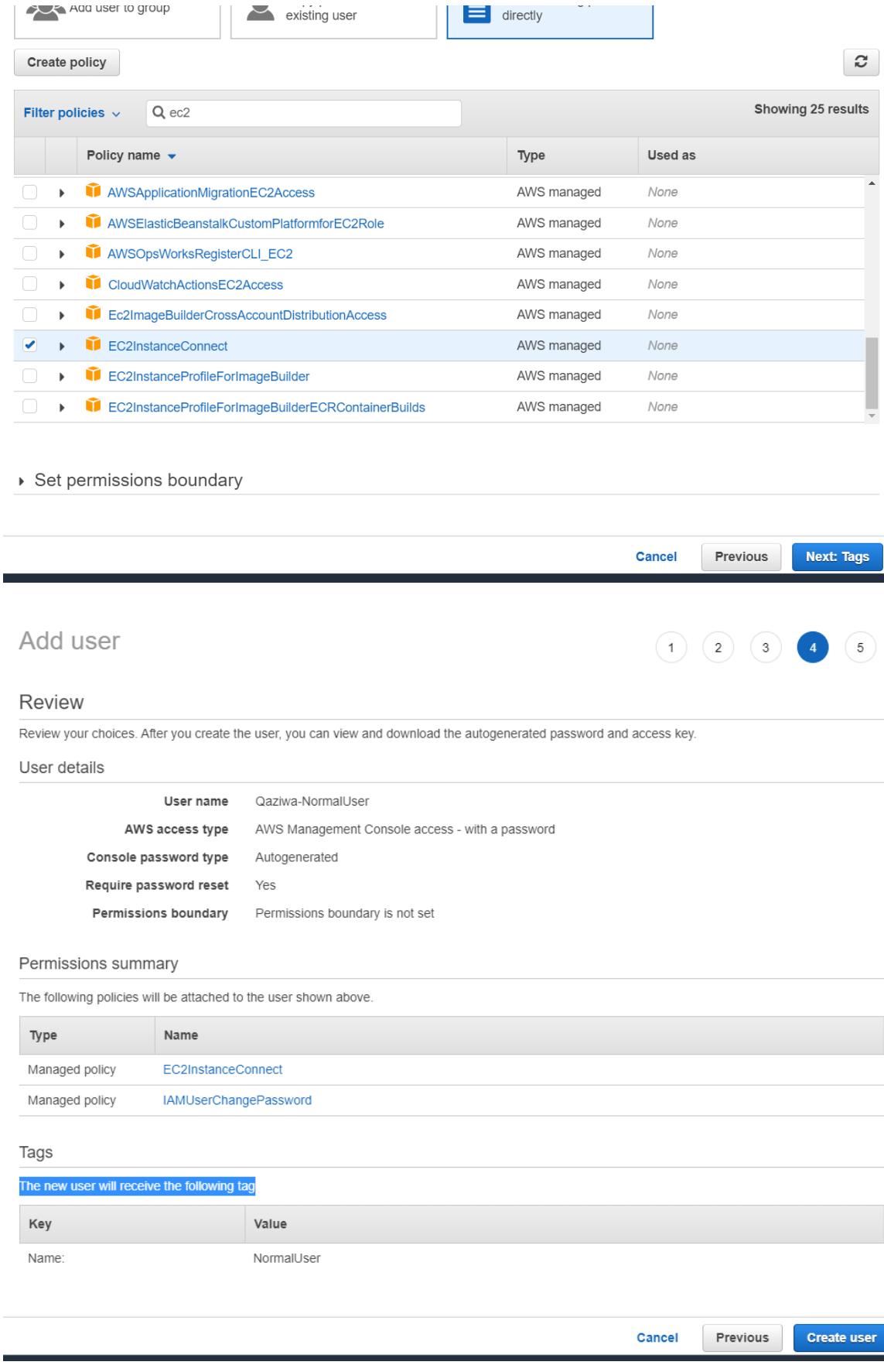
▸ Set permissions boundary

Cancel Previous Next: Tags



▸ Set permissions boundary

Cancel Previous Next: Tags



AWS educate restriction limits allocation of users.

You successfully created the following users, but some problems remain. Expand the following section for details. View and download user's security credentials as well as email users instructions to log into the AWS Management Console. This will be the last time these credentials will be available to download. However, you can manage and recreate these credentials at any time.

Users with AWS Management Console access may sign-in at: https://501144450828.signin.aws.amazon.com/console

Download .csv

		User
▾	⚠	Qaziwa-NormalUser

- Created user Qaziwa-NormalUser
- Attached policy EC2InstanceConnect to user Qaziwa-NormalUser
- Attached policy IAMUserChangePassword to user Qaziwa-NormalUser

Could not create login profile for user Qaziwa-NormalUser:

User: arn:aws:sts::501144450828:assumed-role/vocstartsoft/user1406304=19031563@myrp.edu.sg is not authorized to perform: iam:CreateLoginP

Close