Assignment 1b
COS20019
Cloud Computing Architecture

Tran Thien Thao Vy

104991221

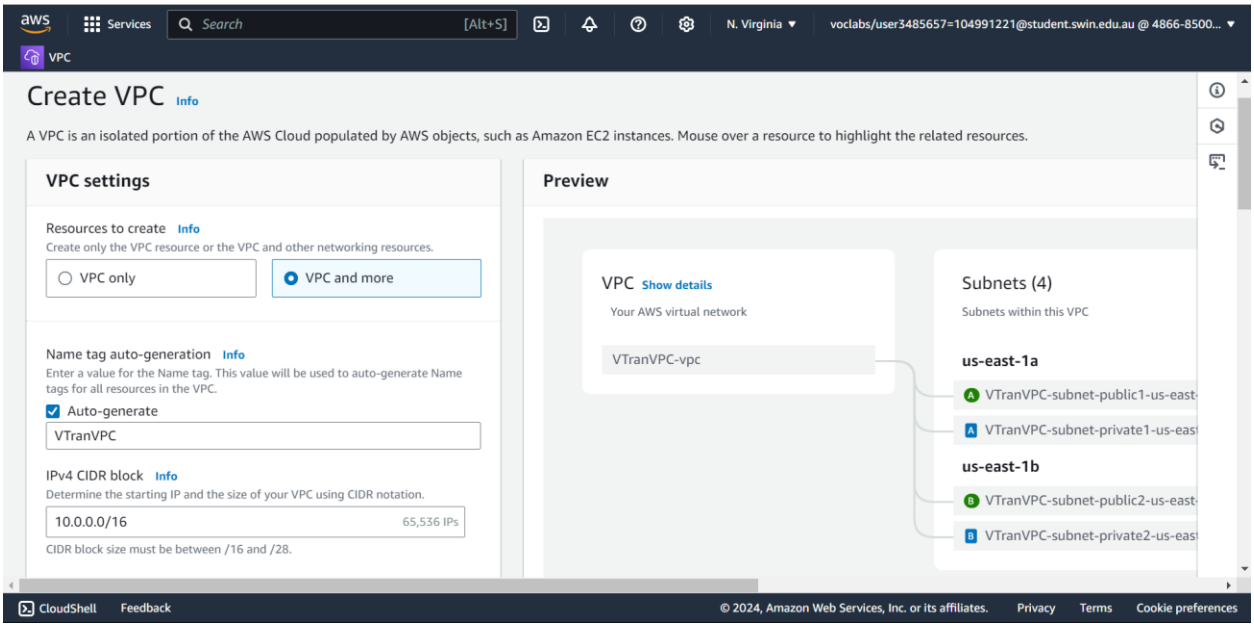
Swinburne University of Technology

School of Science, Computing and Engineering Technologies

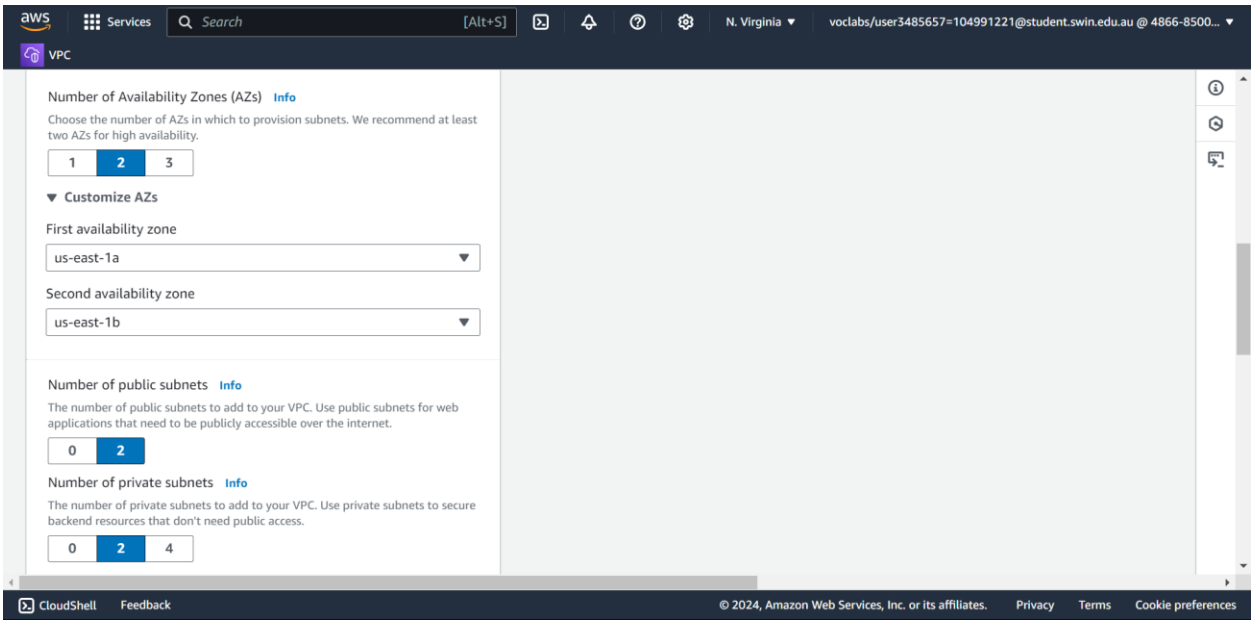
1. Infrastructure deployment

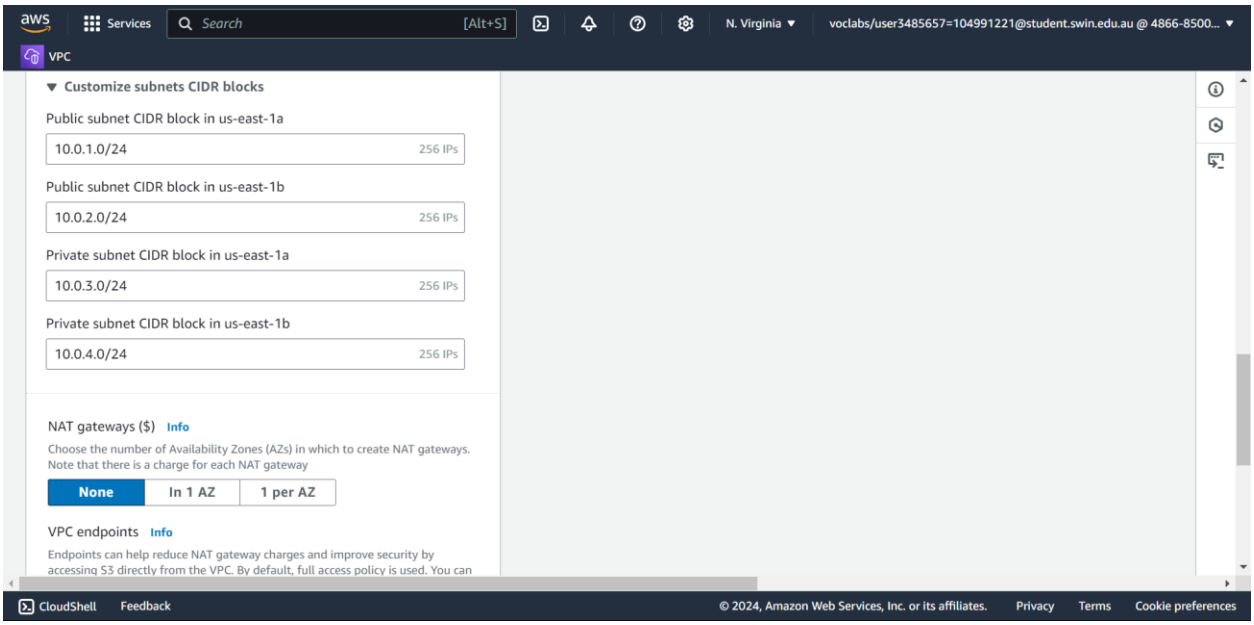
1.1 VPC

Configure the name of VPC as follows the instructions.



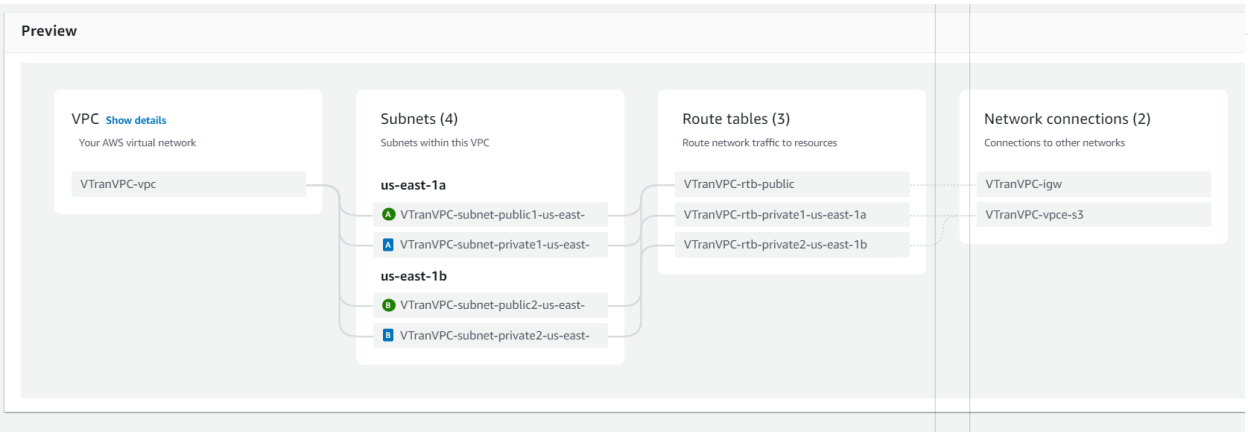
Configure 2 Availability Zones, each with public and private subnets as illustrated in the infrastructure diagram.





Preview of the Subnets connected to the right route tables. Make sure that the route table for public subnets, especially the public subnet 2, is connected to the Internet Gateway connection for further storing and reading photos.

As the private subnets are mainly used for back-end support, there is no need to be connected over the internet.

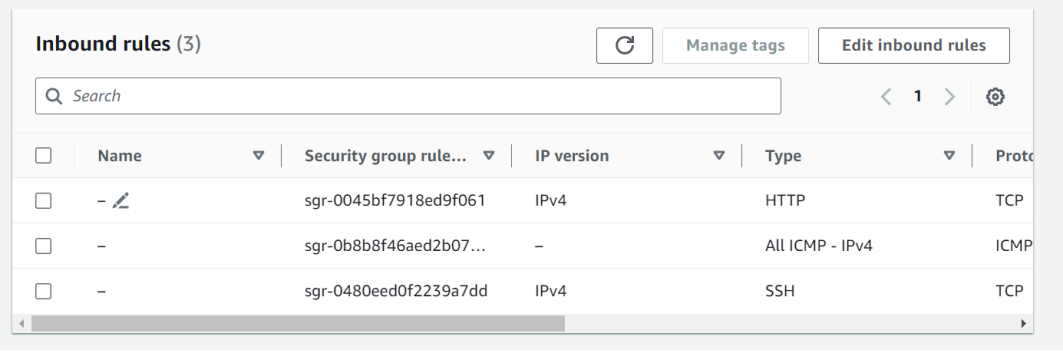


1.2 Security groups

Create 3 security groups, which are WebServerSG, TestInstanceSG, and DBServerSG with the provided Protocols and Source, through checking their Inbound Rules.

Name	Security group ID	Security group name	VPC ID
–	sg-0bf7d9ca59951cc23	WebServerSG	vpc-0c3389438ca
–	sg-0ce93f63915945e07	default	vpc-0c3389438ca
–	sg-079c0a67834989bac	DBServerSG	vpc-0c3389438ca
–	sg-017fd2aac77f46c08	default	vpc-03c1cdd6ac4
–	sg-0b9b92824e81ce26e	WebServer-SG	vpc-03c1cdd6ac4
–	sg-09175606d8e9a0050	TestInstanceSG	vpc-0c3389438ca

Inbound rules of WebServerSG:

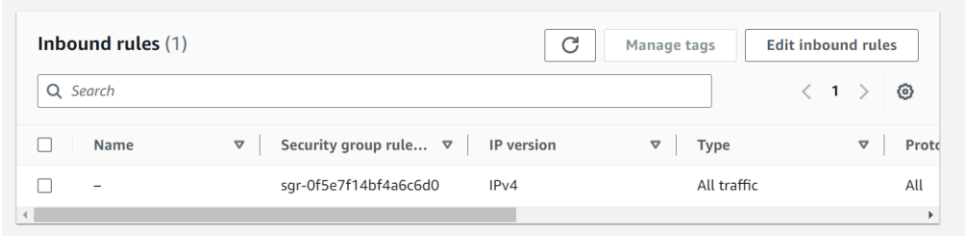


Inbound rules (3) Manage tags Edit inbound rules

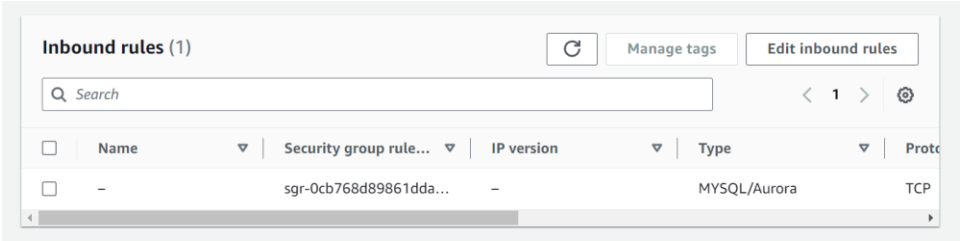
Search 1

Name	Security group rule...	IP version	Type	Proto
–	sgr-0045bf7918ed9f061	IPv4	HTTP	TCP
–	sgr-0b8b8f46aed2b07...	–	All ICMP - IPv4	ICMP
–	sgr-0480eed0f2239a7dd	IPv4	SSH	TCP

Inbound rules of TestInstanceSG:



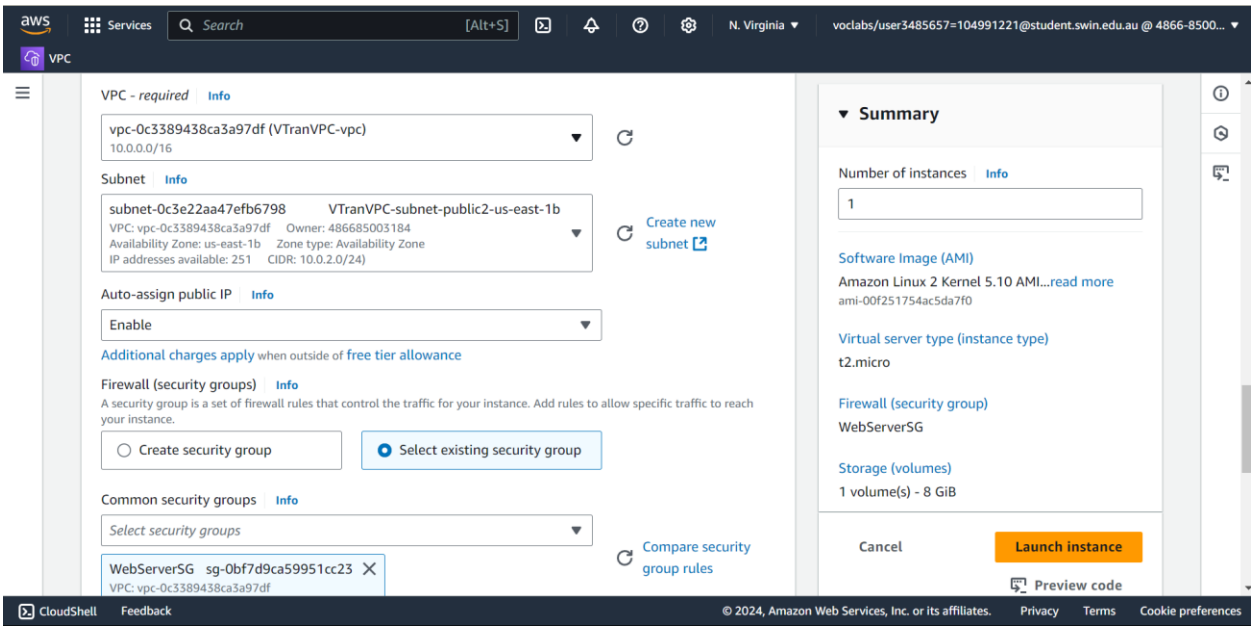
Inbound rules of DBServerSG:



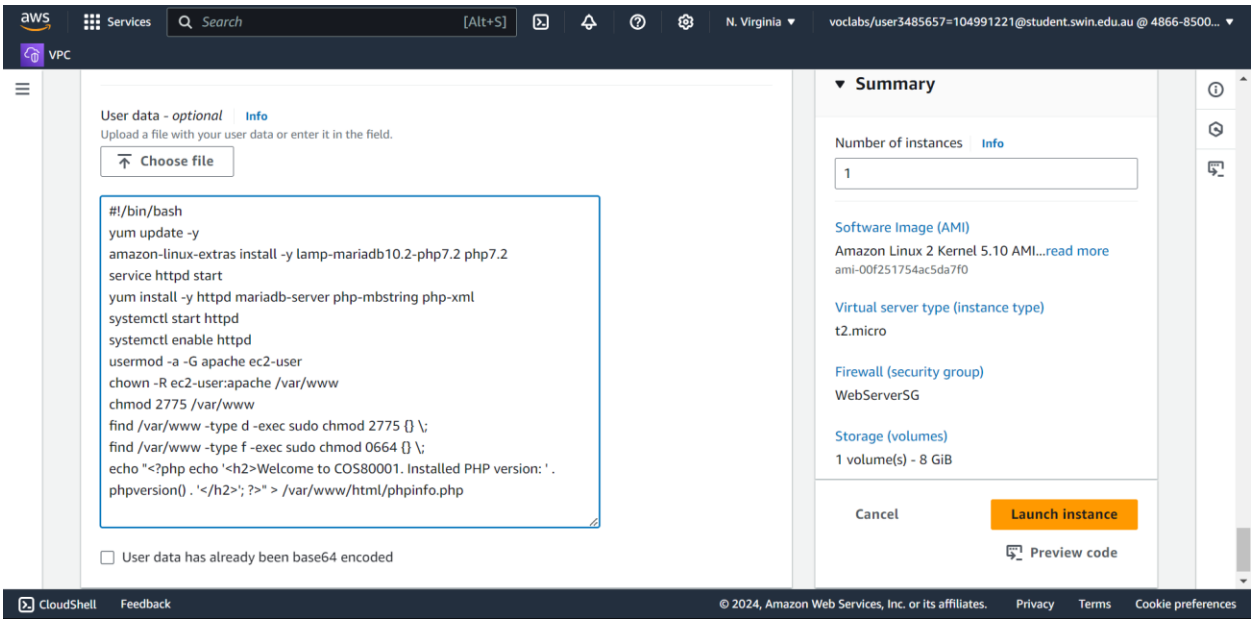
1.3 EC2 virtual machine

1.3.1 Bastion/Web server instance

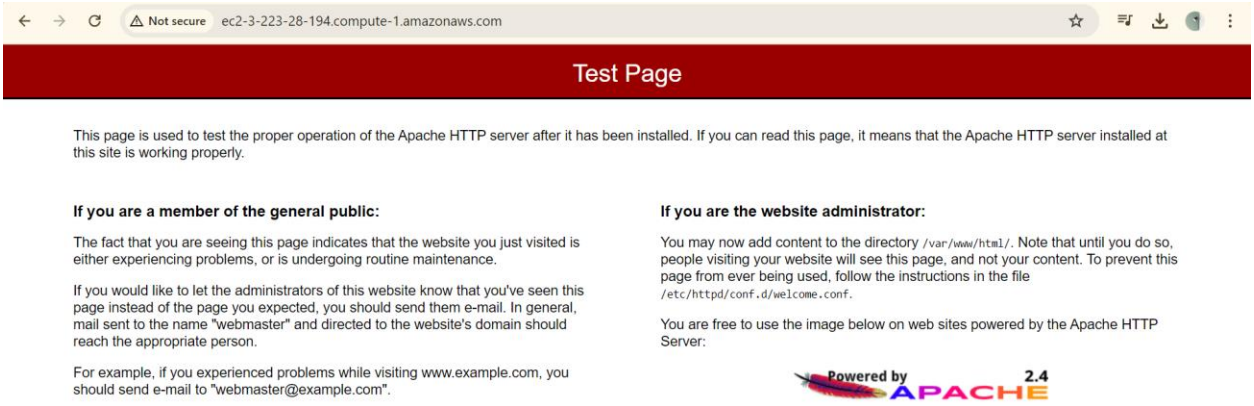
Configure the instance as the instruction requires, make sure to allocate the instance into the correct VPC, Public Subnet 2.



Reuse the same Apache text as in assignment 1a to install the Apache web server and PHP packages.

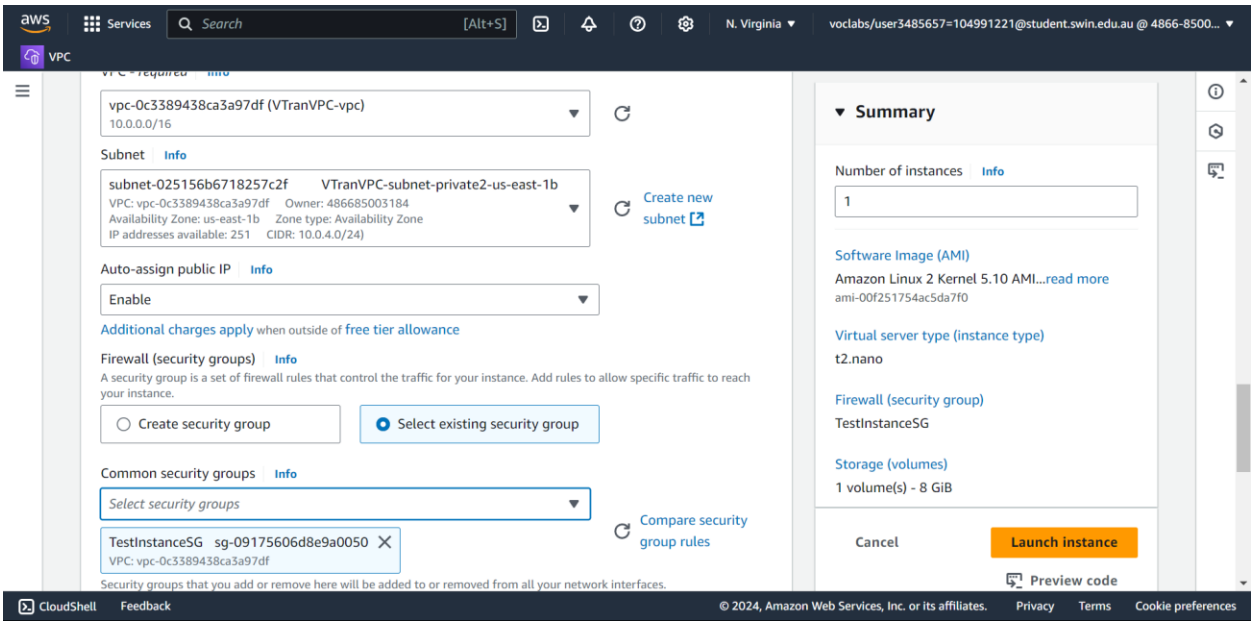


Go to the web browser, using the ec2 public DNS to check if the Apache has been installed successfully.

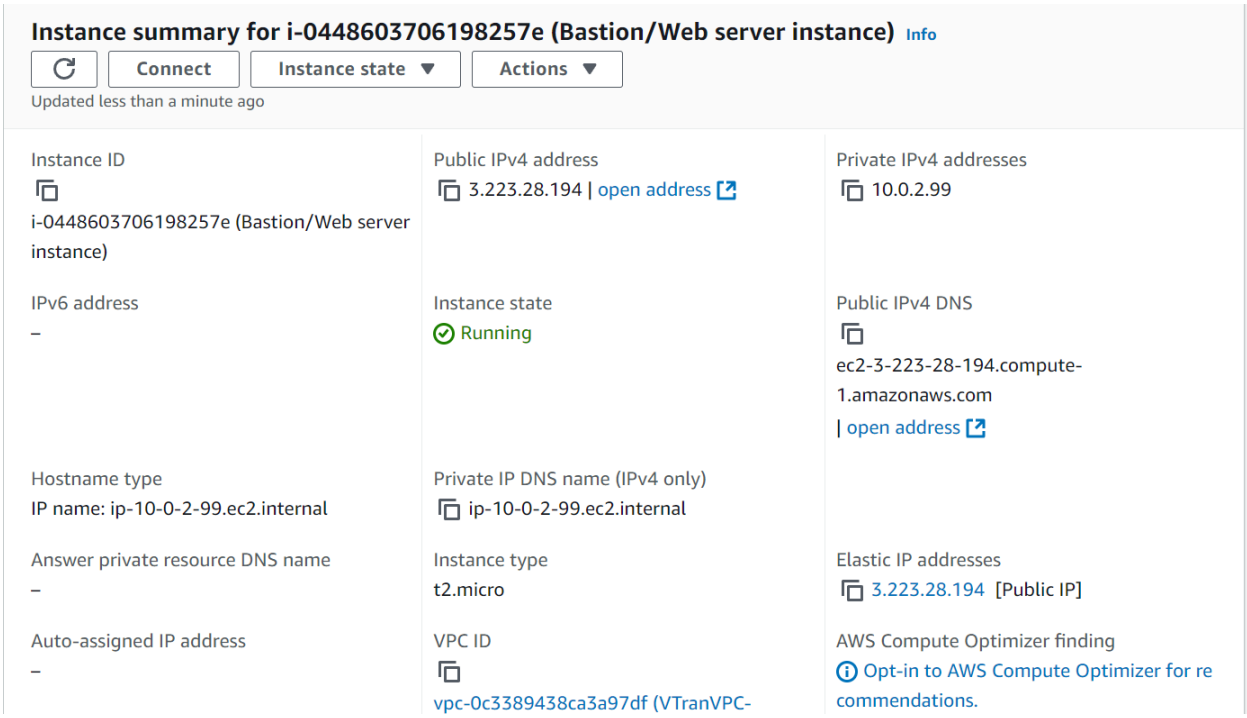


1.3.2 Test instance

Configure the test instance, make sure that it employs the TestInstanceSG security group.

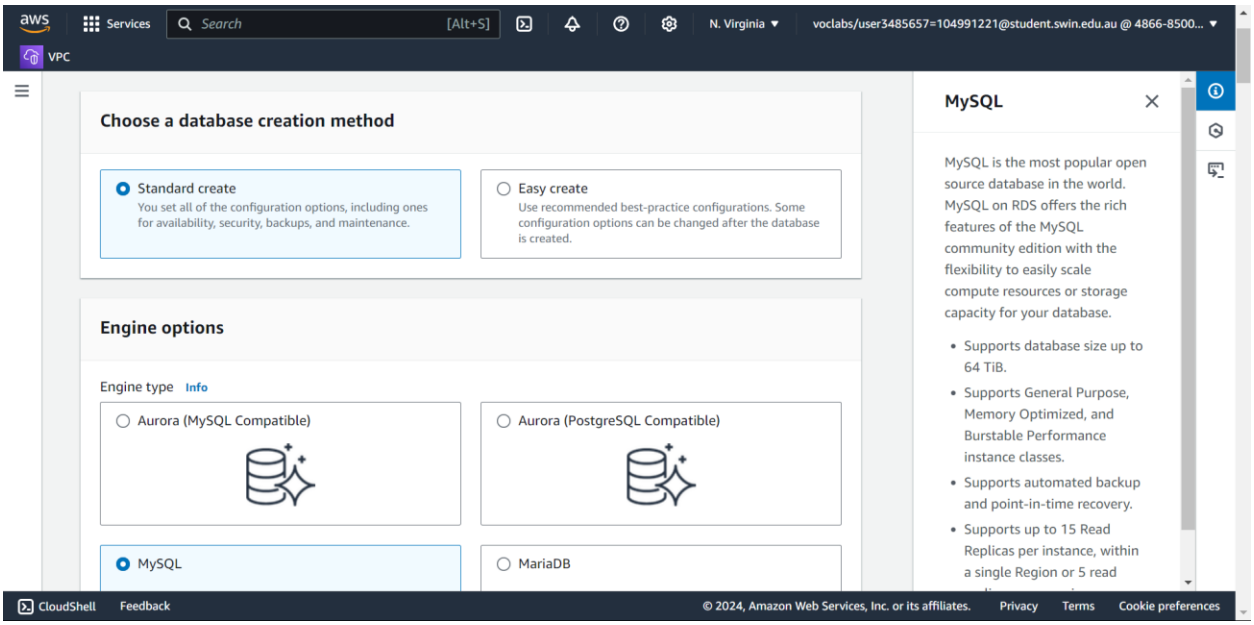


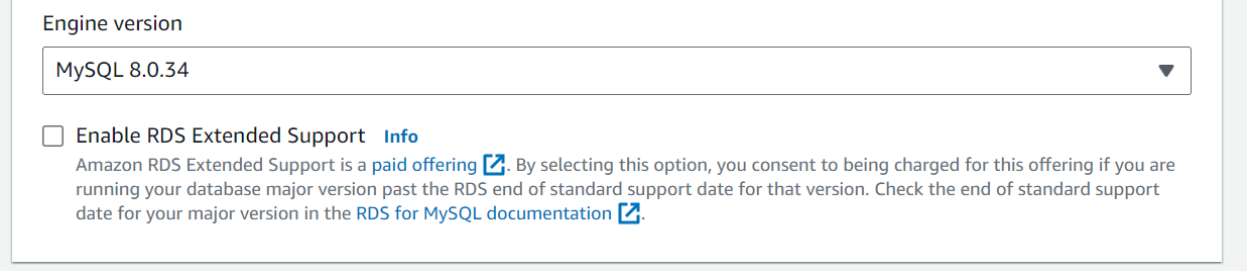
Assigning the Elastic IPs to the instances so that the next time we restart the instances, the public DNS and IPs will not be changed.



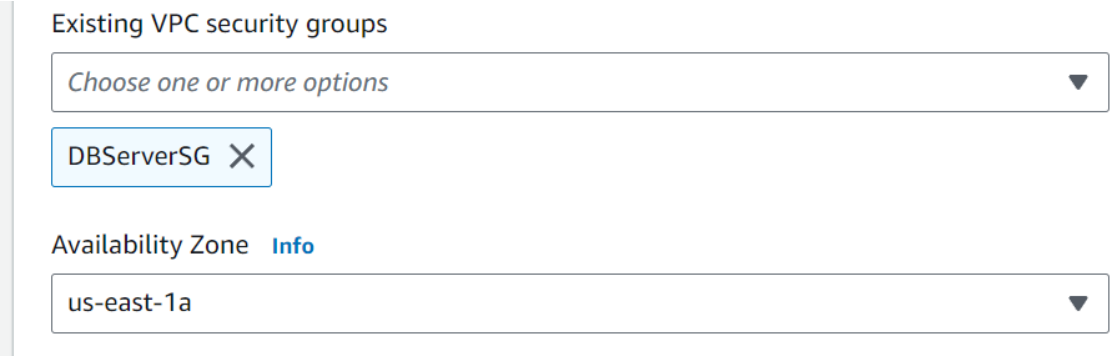
1.4 RDS database instance

Configure the RDS database instance as required.





Make sure we configure it into the DB Server security group.



To access to the database through the internet for setting up and maintenance, we can use the phpMyAdmin.

We can access the phpMyAdmin through Putty and public EC2 DNS:

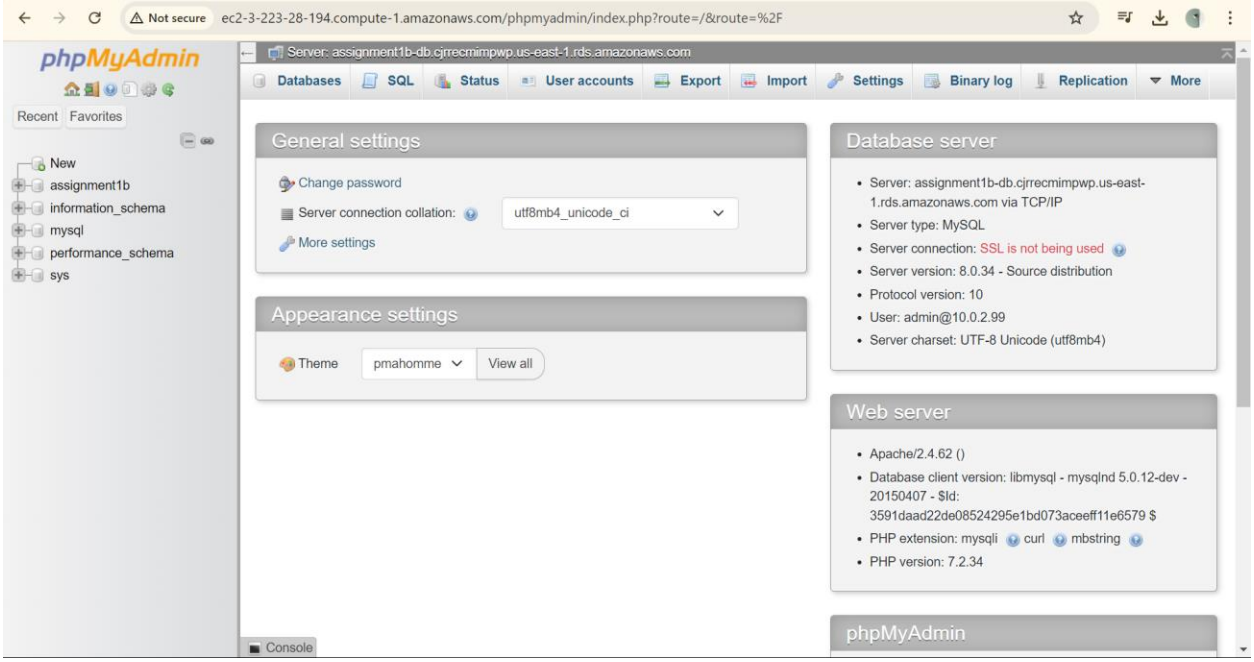
```
ec2-user@ip-10-0-2-99:/var/www/html
login as: ec2-user
Authenticating with public key "assignmentlb"
       __|  __|_  )
       _|  (     /   Amazon Linux 2
      ___|\___|___|
      https://aws.amazon.com/linux/amazon-linux-2023/
[ec2-user@ip-10-0-2-99 ~]$ cd /var/www/html
[ec2-user@ip-10-0-2-99 html]$ wget https://files.phpmyadmin.net/phpMyAdmin/5.2.1
/phpMyAdmin-5.2.1-english.zip
--2024-10-12 14:45:40--  https://files.phpmyadmin.net/phpMyAdmin/5.2.1/phpMyAdmi
n-5.2.1-english.zip
Resolving files.phpmyadmin.net (files.phpmyadmin.net)... 109.61.91.231, 2a02:6ea
0:a200::17
Connecting to files.phpmyadmin.net (files.phpmyadmin.net)|109.61.91.231|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 10448276 (10.0M) [application/zip]
Saving to: 'phpMyAdmin-5.2.1-english.zip'

100%[===================================>] 10,448,276  --.-K/s   in 0.09s

2024-10-12 14:45:40 (108 MB/s) - 'phpMyAdmin-5.2.1-english.zip' saved [10448276/
10448276]

[ec2-user@ip-10-0-2-99 html]$
```

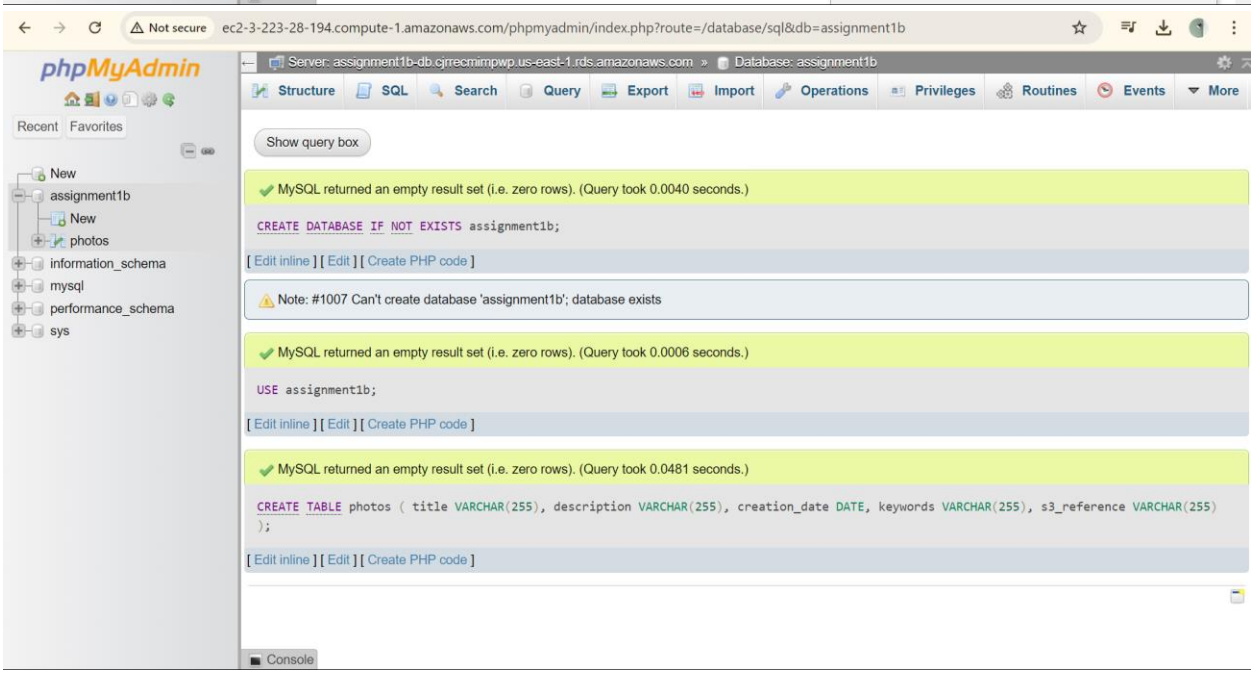
```
ec2-user@ip-10-0-2-99:/var/www/html
  inflating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/PublicKeyCredentialSource.php
  inflating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/PublicKeyCredentialSourceRepository.php
  inflating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/PublicKeyCredentialUserEntity.php
  inflating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/Server.php
  inflating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/StringStream.php
   creating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/TokenBinding/
  inflating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/TokenBinding/IgnoreTokenBindingHandler.php
  inflating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/TokenBinding/SecTokenBindingHandler.php
  inflating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/TokenBinding/TokenBinding.php
  inflating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/TokenBinding/TokenBindingHandler.php
  inflating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/TokenBinding/TokenBindingNotSupportedHandler.php
   creating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/TrustPath/
  inflating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/TrustPath/CertificateTrustPath.php
  inflating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/TrustPath/EcdaaKeyIdTrustPath.php
  inflating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/TrustPath/EmptyTrustPath.php
  inflating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/TrustPath/TrustPath.php
  inflating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/TrustPath/TrustPathLoader.php
  inflating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/U2FPublicKey.php
   creating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/Util/
  inflating: phpMyAdmin-5.2.1-english/vendor/web-auth/webauthn-lib/src/Util/CoseSignatureFixer.php
   creating: phpMyAdmin-5.2.1-english/vendor/webmozart/
   creating: phpMyAdmin-5.2.1-english/vendor/webmozart/assert/
  inflating: phpMyAdmin-5.2.1-english/vendor/webmozart/assert/CHANGELOG.md
  inflating: phpMyAdmin-5.2.1-english/vendor/webmozart/assert/LICENSE
  inflating: phpMyAdmin-5.2.1-english/vendor/webmozart/assert/README.md
  inflating: phpMyAdmin-5.2.1-english/vendor/webmozart/assert/composer.json
   creating: phpMyAdmin-5.2.1-english/vendor/webmozart/assert/src/
  inflating: phpMyAdmin-5.2.1-english/vendor/webmozart/assert/src/Assert.php
  inflating: phpMyAdmin-5.2.1-english/vendor/webmozart/assert/src/InvalidArgumentException.php
  inflating: phpMyAdmin-5.2.1-english/vendor/webmozart/assert/src/Mixin.php
   creating: phpMyAdmin-5.2.1-english/vendor/williamdes/
   creating: phpMyAdmin-5.2.1-english/vendor/williamdes/mariadb-mysql-kbs/
  inflating: phpMyAdmin-5.2.1-english/vendor/williamdes/mariadb-mysql-kbs/CHANGELOG.md
  inflating: phpMyAdmin-5.2.1-english/vendor/williamdes/mariadb-mysql-kbs/LICENSE
  inflating: phpMyAdmin-5.2.1-english/vendor/williamdes/mariadb-mysql-kbs/README.md
  inflating: phpMyAdmin-5.2.1-english/vendor/williamdes/mariadb-mysql-kbs/composer.json
   creating: phpMyAdmin-5.2.1-english/vendor/williamdes/mariadb-mysql-kbs/dist/
  inflating: phpMyAdmin-5.2.1-english/vendor/williamdes/mariadb-mysql-kbs/dist/merged-ultraslim.json
   creating: phpMyAdmin-5.2.1-english/vendor/williamdes/mariadb-mysql-kbs/src/
  inflating: phpMyAdmin-5.2.1-english/vendor/williamdes/mariadb-mysql-kbs/src/KBDocumentation.php
  inflating: phpMyAdmin-5.2.1-english/vendor/williamdes/mariadb-mysql-kbs/src/KBEntry.php
  inflating: phpMyAdmin-5.2.1-english/vendor/williamdes/mariadb-mysql-kbs/src/KBException.php
  inflating: phpMyAdmin-5.2.1-english/vendor/williamdes/mariadb-mysql-kbs/src/Search.php
  inflating: phpMyAdmin-5.2.1-english/vendor/williamdes/mariadb-mysql-kbs/src/SlimData.php
  inflating: phpMyAdmin-5.2.1-english/yarn.lock
[ec2-user@ip-10-0-2-99 html]$ mv phpMyAdmin-5.2.1-english phpmyadmin
[ec2-user@ip-10-0-2-99 html]$
```

Modify the config.inc.php, setting host to the RDS endpoint.

```
/var/www/html/phpmyadmin/config.inc.php - ec2-user@ec2-3-223-28-194.compute-1.amazonaws.com - Editor - WinSCP
declare(strict_types=1);

/**
 * This is needed for cookie based authentication to encrypt the cookie.
 * Needs to be a 32-bytes long string of random bytes. See FAQ 2.10.
 */
$cfg['blowfish_secret'] = ''; /* YOU MUST FILL IN THIS FOR COOKIE AUTH! */

/**
 * Servers configuration
 */
$i = 0;

/**
 * First server
 */
$i++;
/* Authentication type */
$cfg['Servers'][$i]['auth_type'] = 'cookie';
/* Server parameters */
$cfg['Servers'][$i]['host'] = 'assignmentlb-db.cjrrecmimpwp.us-east-1.rds.amazonaws.com';
$cfg['Servers'][$i]['compress'] = false;
$cfg['Servers'][$i]['AllowNoPassword'] = false;

/**
 * phpMyAdmin configuration storage settings.
 */

/* User used to manipulate with storage */
// $cfg['Servers'][$i]['controlhost'] = '';
// $cfg['Servers'][$i]['controlport'] = '';
// $cfg['Servers'][$i]['controluser'] = 'pma';
// $cfg['Servers'][$i]['controlpass'] = 'pmapass';

/* Storage database and tables */
// $cfg['Servers'][$i]['pmadb'] = 'phpmyadmin';
// $cfg['Servers'][$i]['bookmarktable'] = 'pma__bookmark';
Line: 31/160   Column: 41   Encoding: 1252  (ANSI - L  Modified
```

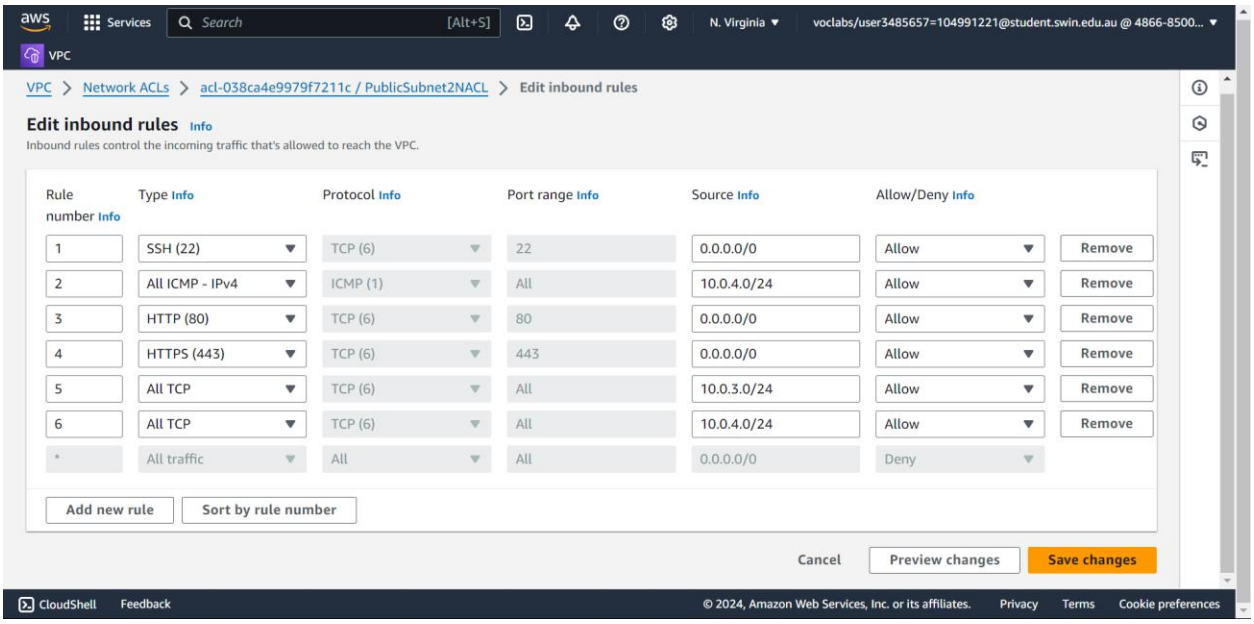
After that, we can access the phpMyAdmin console through EC2 public DNS and create a photo table with 5 columns, which are title, description, creation date, keywords and s3 references.

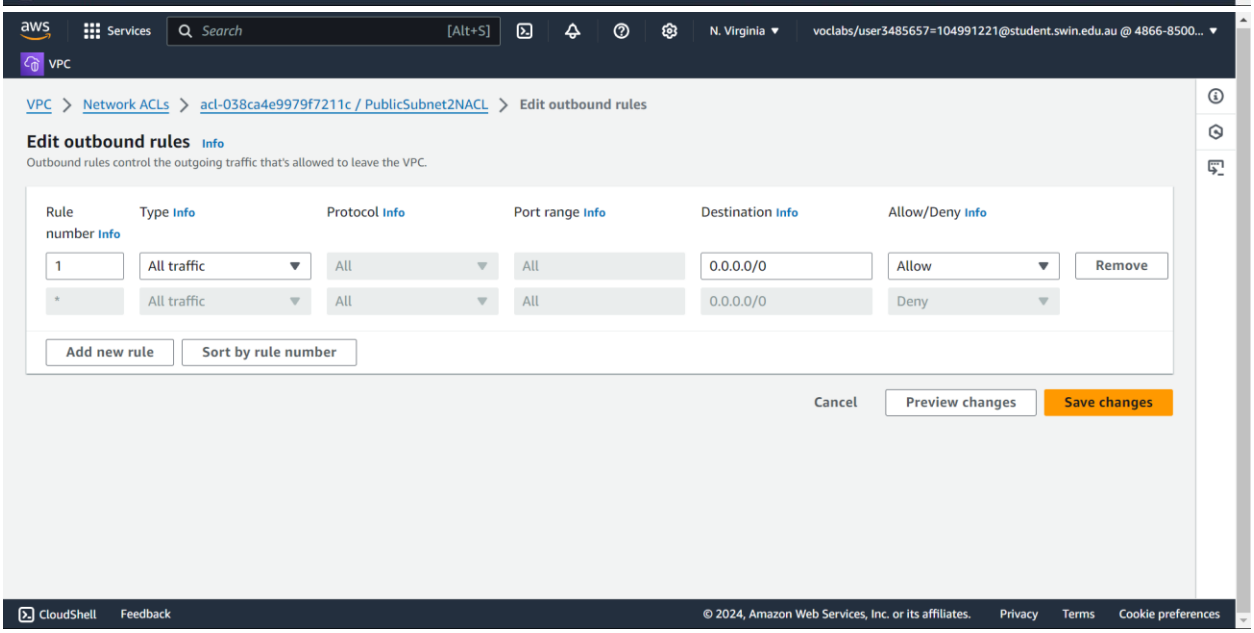




1.5 Network ACL

Network ACL is an additional layer of the security group in which we can add more inbound rules and outbound rules for the ACL named “PublicSubnet2NACL”.

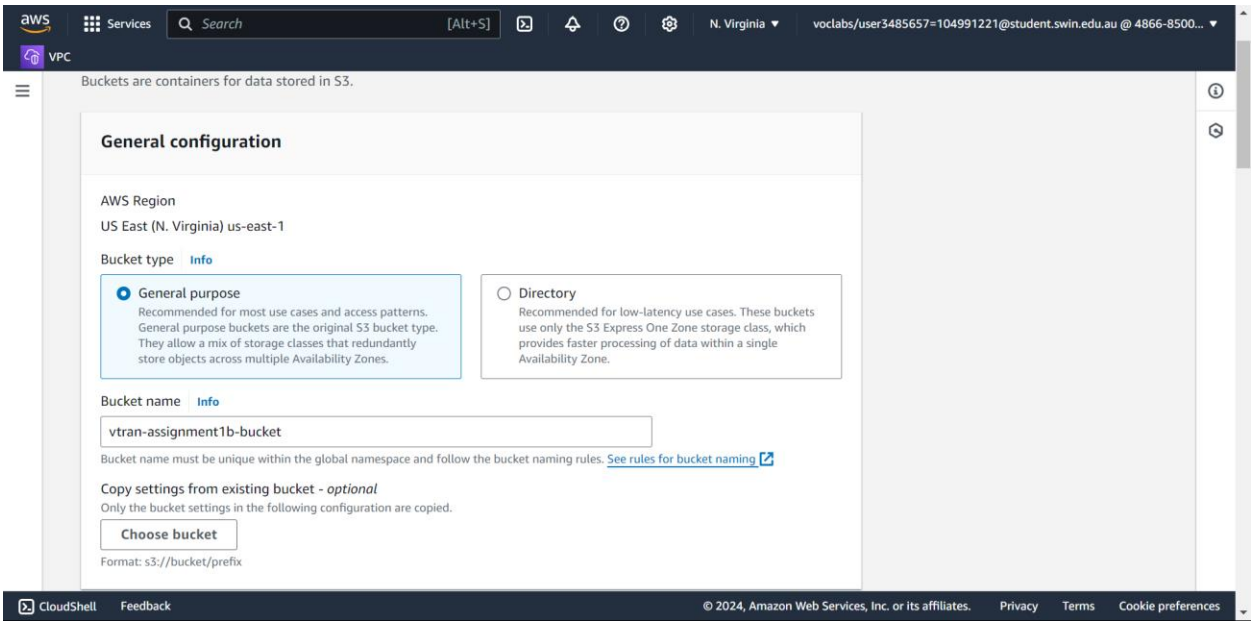




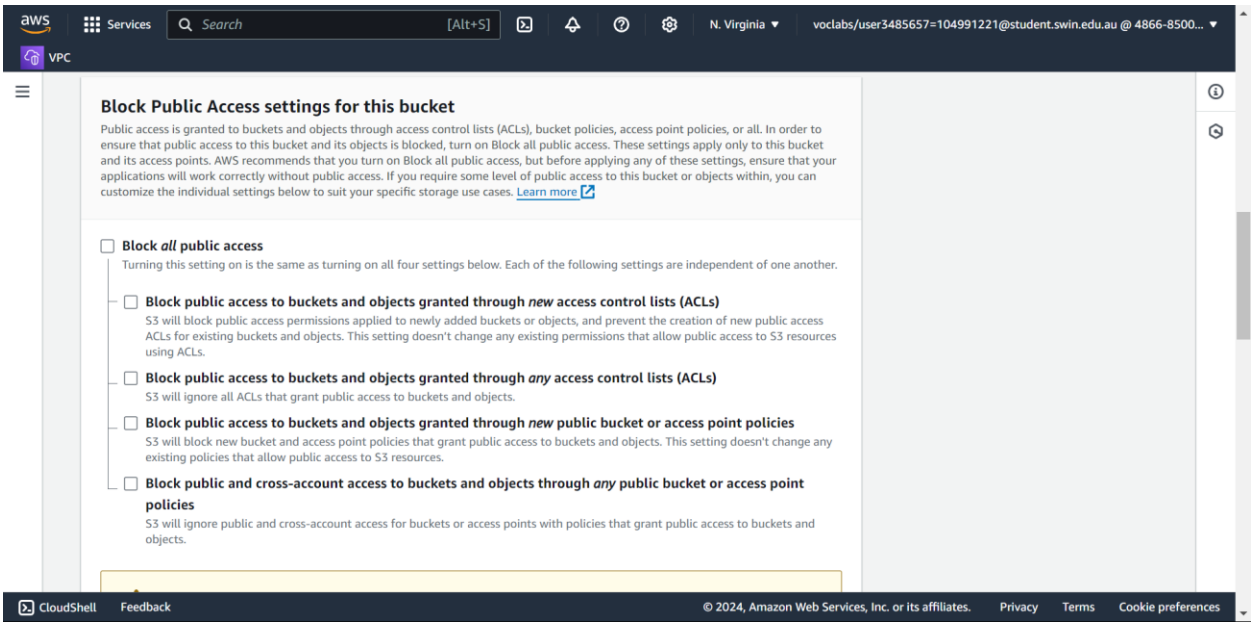
2. Functional Requirements of Photo Album website

2.1 Photo Storage

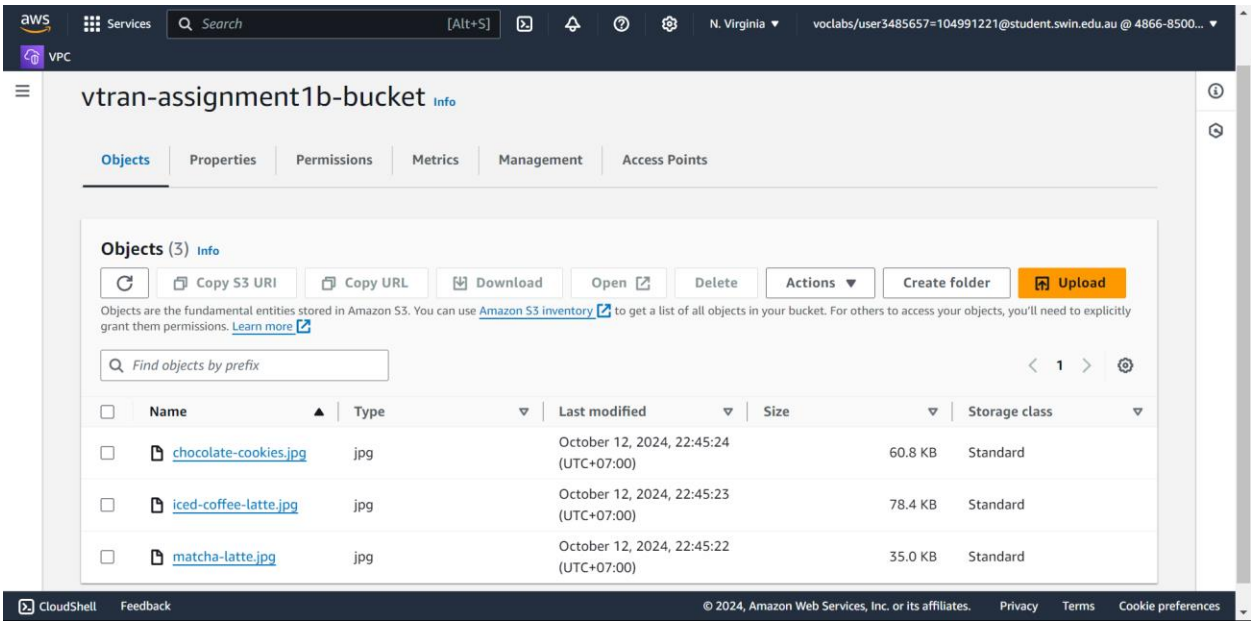
Create an S3 bucket for uploading resources.



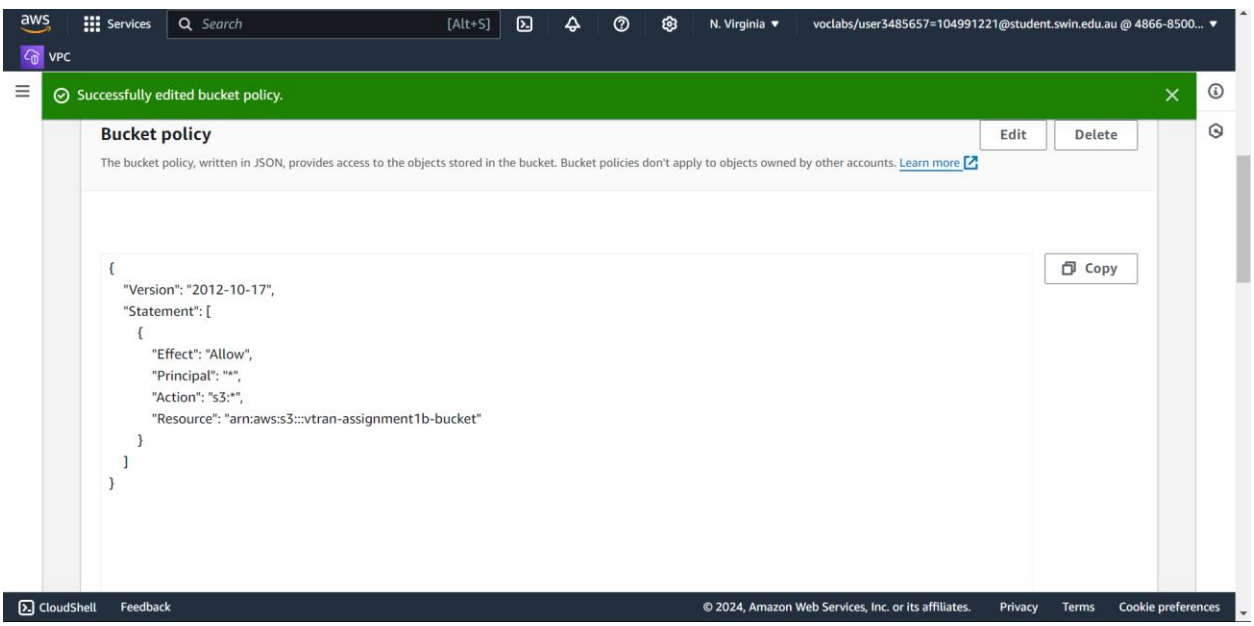
Uncheck the block all public access to the s3 bucket.



And manually upload some photos.

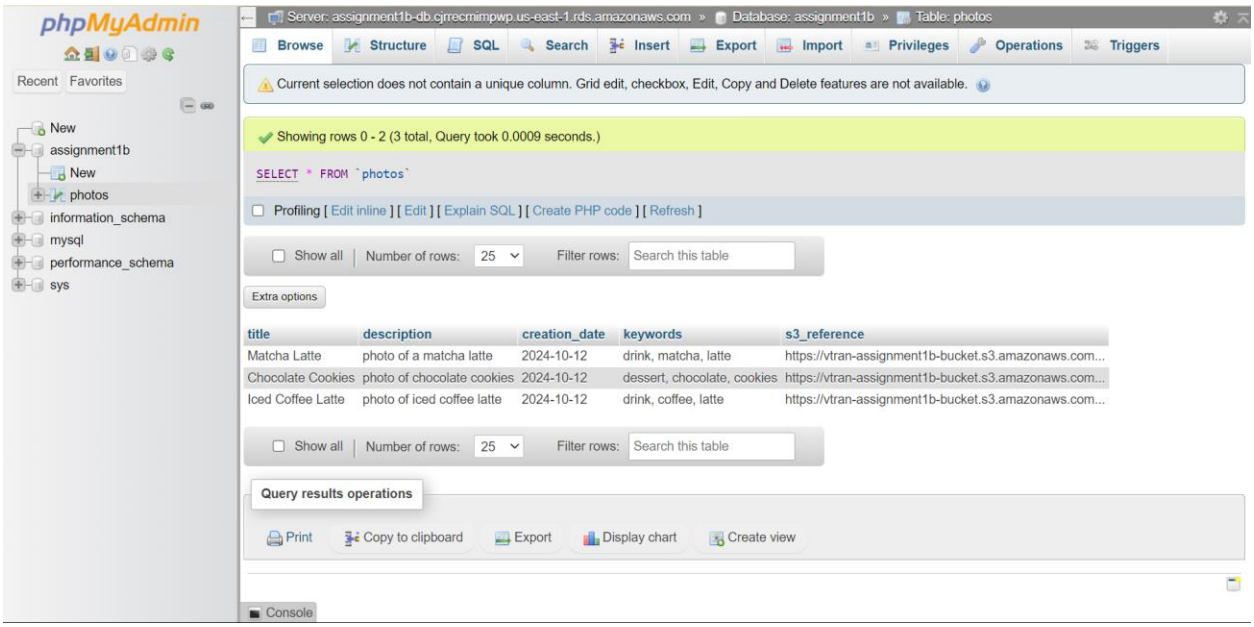


Bucket policy to enable public access.



2.2 Photo meta-data in the RDS Database

Populate the database table with some records.



2.3 Photo Album website functionality

Modify the constants.php using data from the S3 bucket and RDS database and transfer them using WinSCP.

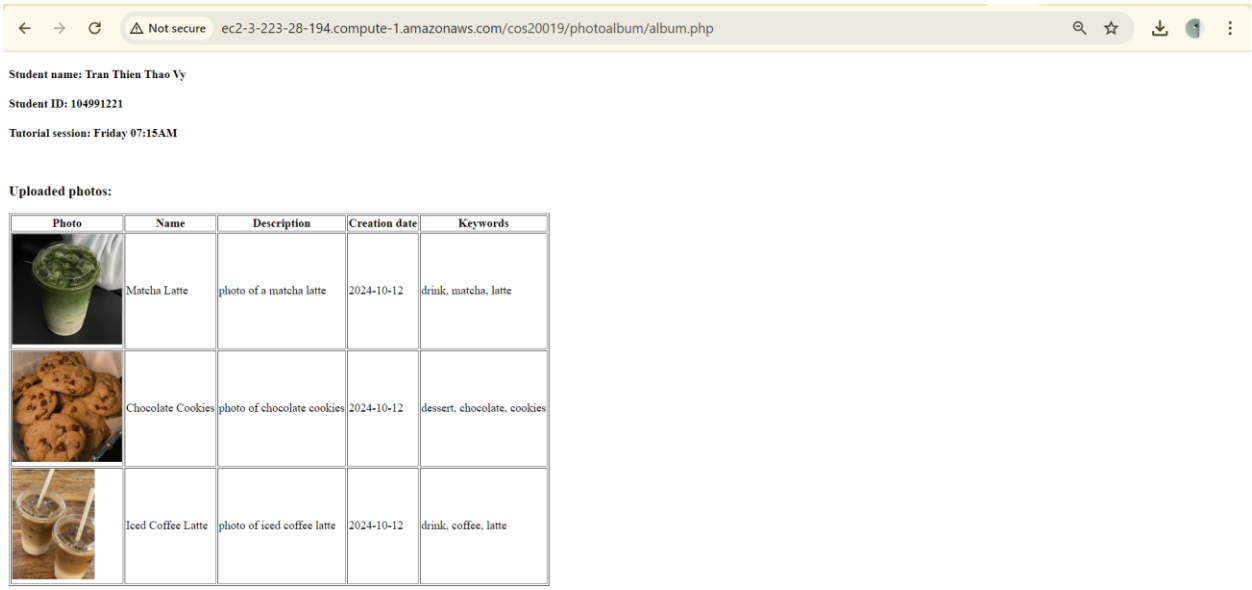
```
constants.php > ...
// [ACTION REQUIRED] your full name
define('STUDENT_NAME', 'Tran Thien Thao Vy');
// [ACTION REQUIRED] your Student ID
define('STUDENT_ID', '104991221');
// [ACTION REQUIRED] your tutorial session
define('TUTORIAL_SESSION', 'Friday 07:15AM');

// [ACTION REQUIRED] name of the S3 bucket that stores images
define('BUCKET_NAME', 'vtran-assignment1b-bucket');
// [ACTION REQUIRED] region of the above bucket
define('REGION', 'us-east-1');
// no need to update this const
define('S3_BASE_URL','https://'.BUCKET_NAME.'.s3.amazonaws.com/');

// [ACTION REQUIRED] name of the database that stores photo meta-data (note that this is not the DB identifier of the RDS instance)
define('DB_NAME', 'assignment1b');
// [ACTION REQUIRED] endpoint of RDS instance
define('DB_ENDPOINT', 'assignment1b-db.cjrrecmimpwp.us-east-1.rds.amazonaws.com');
// [ACTION REQUIRED] username of your RDS instance
define('DB_USERNAME', 'admin');
// [ACTION REQUIRED] password of your RDS instance
define('DB_PWD', 'admin123');
```

```
// [ACTION REQUIRED] name of the DB table that stores photo's meta-data
define('DB_PHOTO_TABLE_NAME', 'photos');
// The table above has 5 columns:
// [ACTION REQUIRED] name of the column in the above table that stores photo's titles
define('DB_PHOTO_TITLE_COL_NAME', 'title');
// [ACTION REQUIRED] name of the column in the above table that stores photo's descriptions
define('DB_PHOTO_DESCRIPTION_COL_NAME', 'description');
// [ACTION REQUIRED] name of the column in the above table that stores photo's creation dates
define('DB_PHOTO_CREATIONDATE_COL_NAME', 'creation_date');
// [ACTION REQUIRED] name of the column in the above table that stores photo's keywords
define('DB_PHOTO_KEYWORDS_COL_NAME', 'keywords');
// [ACTION REQUIRED] name of the column in the above table that stores photo's links in S3
define('DB_PHOTO_S3REFERENCE_COL_NAME', 's3_reference');
?>
```

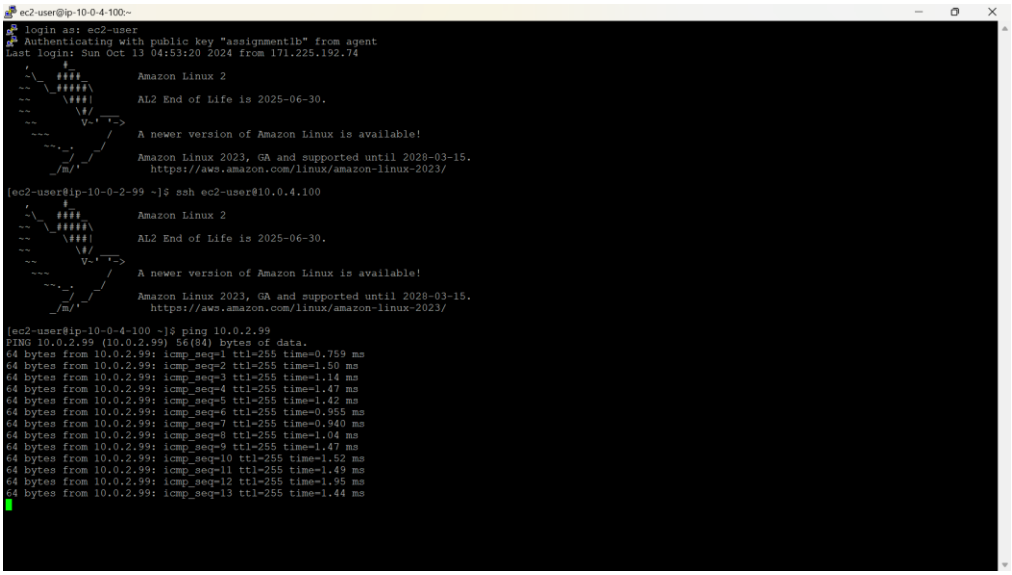
3. Testing

After the meta-data records has been populated into the phpMyAdmin table and php files has been transfer into the Web Instance.

The website is now accessible.



Test the Network ACL configuration by ssh into the Test Instance through the Bastic/Web Server host, and ping to the Web Server's IP address.



4. Additional information

EC2 link into the Web Server: http://ec2-3-223-28-194.compute-1.amazonaws.com/cos20019/photoalbum/album.php

EC2 link to the phpMyAdmin console: http://ec2-3-223-28-194.compute-1.amazonaws.com/phpmyadmin/