**Assignment 3 – Multi-Cloud Architecture Deployment**



**Public URL of the AWS Instance Website:** [**http://34.231.224.149/cos80001/photoalbum/album.php**](http://34.231.224.149/cos80001/photoalbum/album.php)

**OCI Bastion/web server:** [**http://193.122.143.70/phpmyadmin**](http://193.122.143.70/phpmyadmin)

|  |  |  |
| --- | --- | --- |
| **Team Member** | **Student ID** | **Responsibilities** |
| **Alexander Blatchford** | 101623699 | - Set up the OCI infrastructure, including the VCN, public and private subnets, and route tables.  - Deployed the phpMyAdmin web server in the public subnet and Installed Apache, PHP, and configured phpMyAdmin to connect to the MySQL database.  - Created and deployed the MySQL database instance, including database schema setup and testing via phpMyAdmin |
| **Arun Ragavendhar Arunachalam Palaniyappan** | 104837257 | - Handled the full VPN tunnel configuration across both clouds: created DRG, Customer Gateway, and VPN Connections on OCI and AWS and deployed the photoalbum website.  - Configured all Firewalls and security components: Security Lists and Security groups on OCI, Security groups and NACL on AWS.  - Enforced the least privilege principle and tested end-to-end connectivity across cloud environments. |
| **Samsun Gulshan Sheik Dawood** | 105009251 | - Set up the AWS infrastructure, including the VPC, public and private subnets, and route tables  - Deployed and configured the AWS Bastion/Web server, including Apache, PHP, and Elastic IP assignment  - Installed and deployed the photoalbum web application under the correct directory, connecting it to the OCI database via VPN  - Created the test instance in the private subnet and performed ICMP ping tests to verify internal connectivity. |

*Table 1: Team Member Names and Contribution*

Table of Contents

[I. INTRODUCTION 2](#_Toc199024370)

[II. OCI INFRASTRUCTURE SETUP 2](#_Toc199024371)

[A. Virtual Cloud Network, Route Tabling and Subnet Configuration 2](#_Toc199024372)

[B. OCI Security Lists and NSGs 3](#_Toc199024373)

[C. OCI – OCI Bastion Webserver Instance and Heatwave Database Instance 4](#_Toc199024374)

[D. MySQL Database Deployment on OCI 6](#_Toc199024375)

[E. Object Storage Configuration on OCI 6](#_Toc199024376)

[III. AWS INFRASTRUCTURE SETUP 7](#_Toc199024377)

[A. AWS – Virtual Private Cloud, Route Tabling and Subnet Configuration 7](#_Toc199024378)

[B. AWS Security Groups and Network ACLs 8](#_Toc199024379)

[C. Web server setup on the AWS bastion web server and attaching elastic IP address 10](#_Toc199024380)

[D. ICMP PING connectivity testing from Bastion to Test Instance 10](#_Toc199024381)

[IV. SITE-TO-SITE, TUNNELING AND VPN CONFIGURATION 11](#_Toc199024382)

[A. On OCI 11](#_Toc199024383)

[B. On AWS 12](#_Toc199024384)

[V. WEB APPLICATION DEPLOYMENT AND CONNECTIVITY 13](#_Toc199024385)

[VII. COMPARISON OF CLOUD CONNECTION METHODS 14](#_Toc199024386)

[VIII. CHALLENGES AND LEARNINGS 14](#_Toc199024387)

[IX. CONCLUSION 15](#_Toc199024388)

[X. REFERENCES 15](#_Toc199024389)

# I. INTRODUCTION

This report outlines the step-by-step process followed to deploy a multi-cloud photo album web application using Amazon Web Services (AWS) and Oracle Cloud Infrastructure (OCI). The objective was to split the system architecture, where the compute resources such as the web server run on AWS, while the data storage components—MySQL database and object storage—reside in OCI. A site-to-site IPsec connection was established between the AWS VPC and OCI VCN with a singular active tunnel between gateways on both clouds to ensure safe and restricted access between clouds, the AWS bastion and the OCI database.

**Public URL of the AWS Instance Website:** [**http://34.231.224.149/cos80001/photoalbum/album.php**](http://34.231.224.149/cos80001/photoalbum/album.php)

**OCI Bastion/web server:** [**http://193.122.143.70/phpmyadmin**](http://193.122.143.70/phpmyadmin)

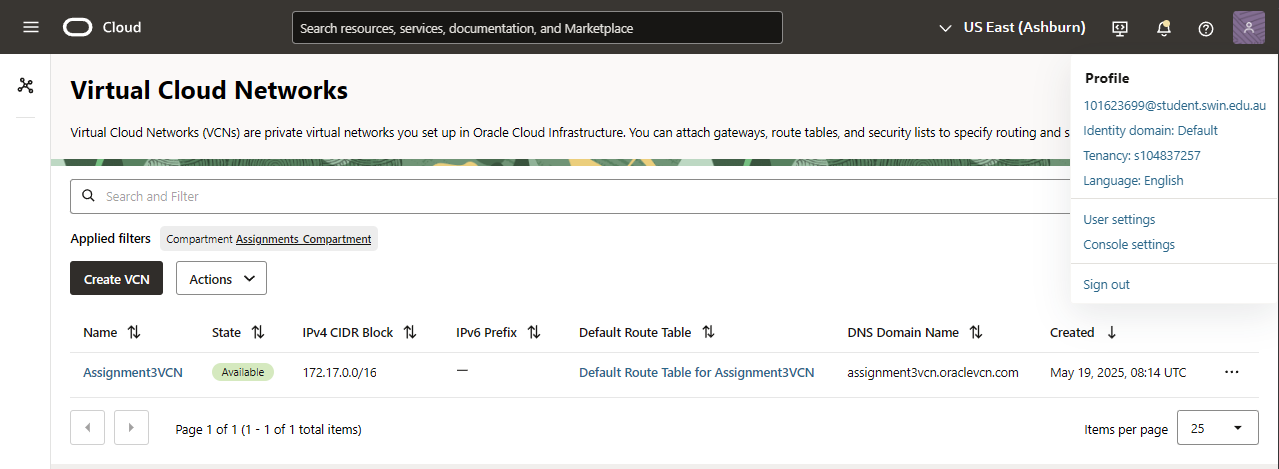
**OCI Tenancy: s104837257, Associated student Email: 104837257@student.swin.edu.au**

**AWS Account - Federated user:** [**voclabs/user3898953=104837257@student.swin.edu.au**](mailto:voclabs/user3898953=104837257@student.swin.edu.au)

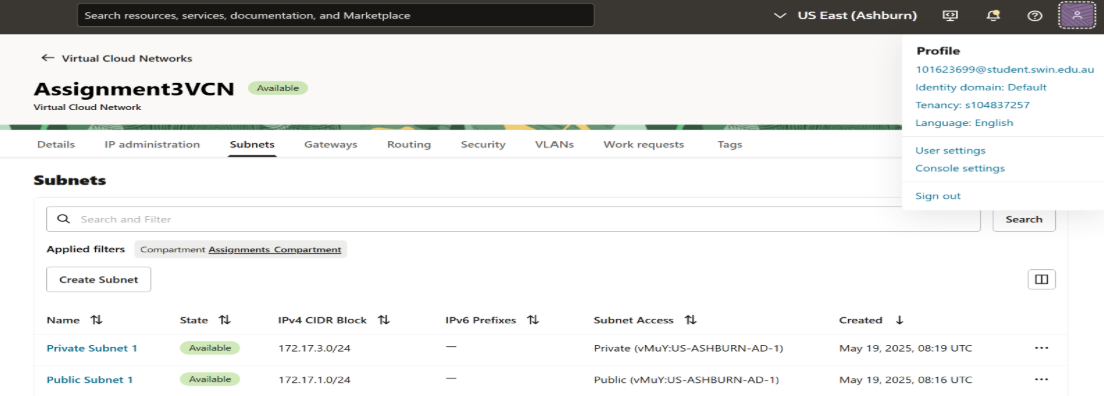
# II. OCI INFRASTRUCTURE SETUP

## A. Virtual Cloud Network, Route Tabling and Subnet Configuration

In the Oracle Cloud Infrastructure (OCI), a Virtual Cloud Network (VCN) named "Assignment-3VCN" was set up in the us-ashburn-1 region using the 172.17.0.0/16 CIDR range. Two subnets were designed within this VCN. Public Subnet 1 (172.17.1.0/24) was created to contain the phpMyAdmin instance and Private Subnet 1 (172.17.3.0/24) was used to host the MySQL database, the VCN’s CIDR is shown in fig 1 and the configuration is illustrated in fig 2.

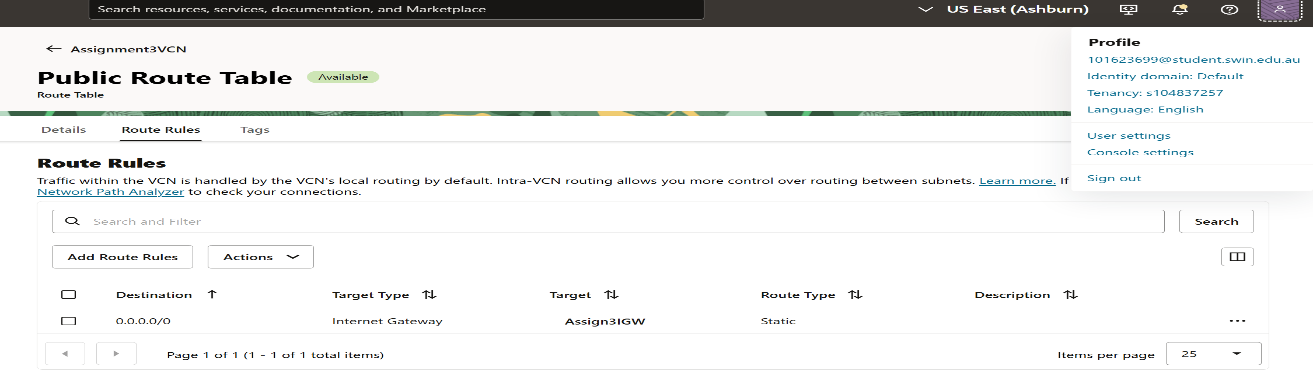


*Fig 1: OCI VCN CIDR and Name*

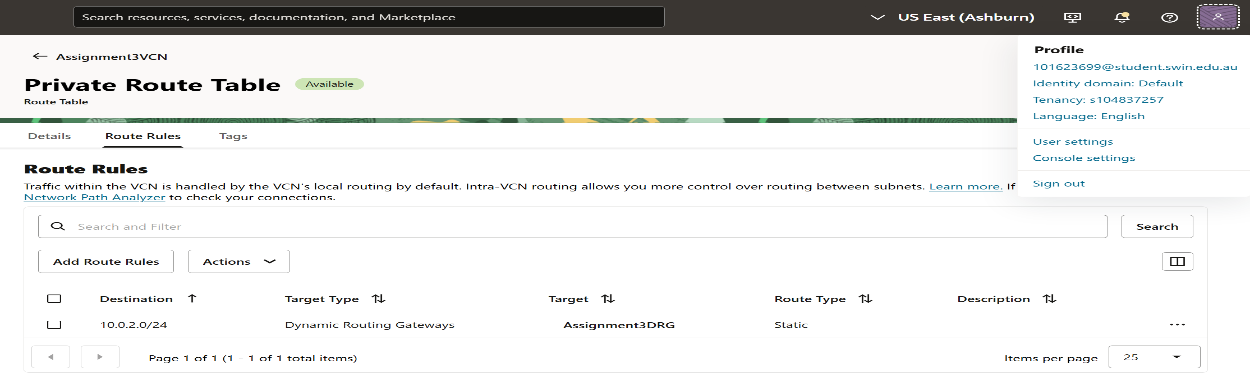


*Fig 2: OCI Subnet Configuration*

The public subnet was linked to a route table (fig 4) with an Internet Gateway (fig 5) to allow internet access. The private subnet was connected to a separate route table (fig 6) that restricted any external connectivity, enhancing internal resource security.



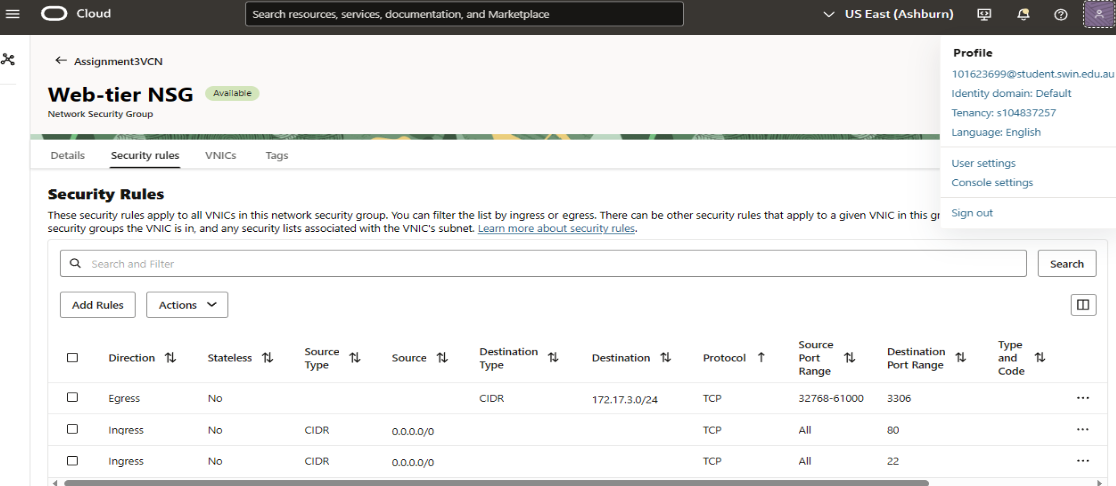
*Fig 3: Public Route Table Route Rules*



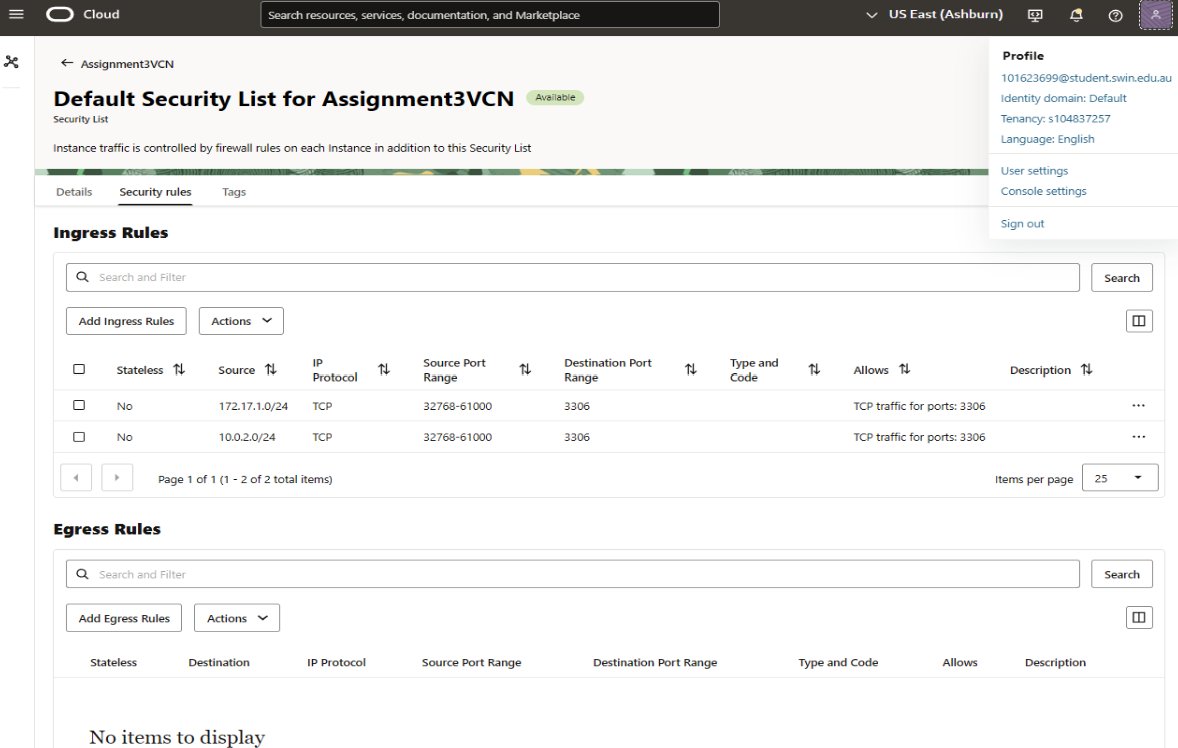
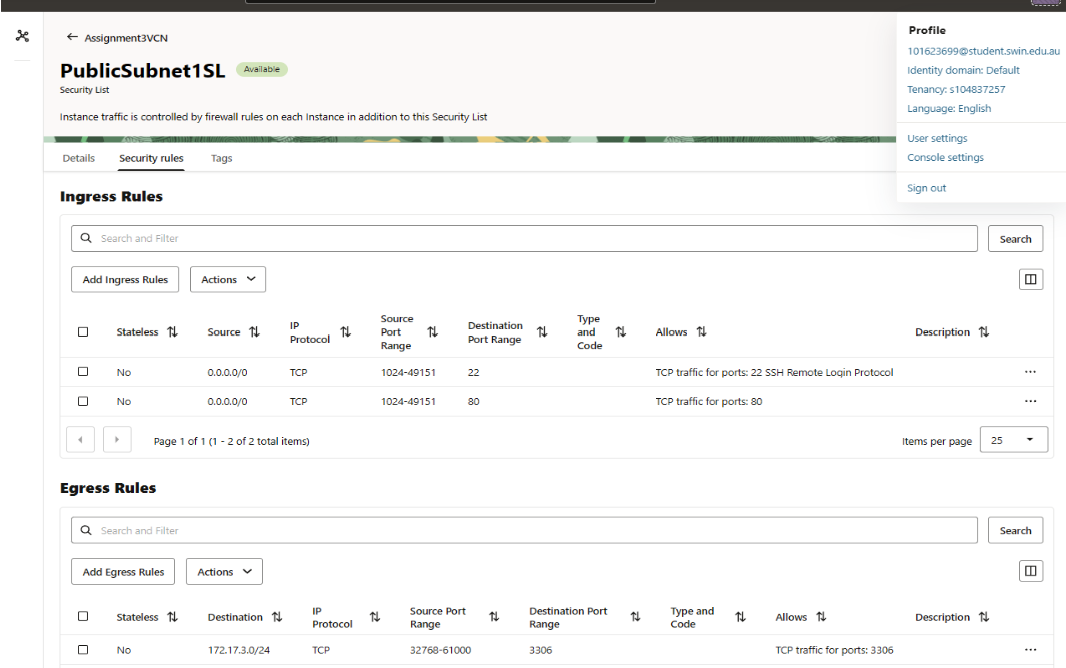
*Fig 4: Private Route Table Route Rules*

## B. Security Lists and NSGs

In OCI, security was managed using a singular security list for the public subnet called PublicSubnet1SL (fig 7) and the default security list was used to manage the private subnet’s security (fig 8). This security was utilized in tandem with a Network Security Group (NSG) called Web-tier NSG to be attached to the VNIC of the bastion instance on OCI (Fig 9). PublicSubnet1SL allows ingress for SSH and HTTP request from anywhere. The default security list allows ingress from the webserver’s restricted 32786-61000 ephemeral ports with a destination to the 3306 MySQL database port. Traffic to the database from the AWS Webserver’s public subnet CIDR (10.0.2.0/24) is also allowed when directed to port 3306 from the ephemeral range. The NSG Web-tier NSG allows egress to the AWS bastion directed to the database from the ephemeral range and ingress from all anywhere for HTTP and SSH from the internet.



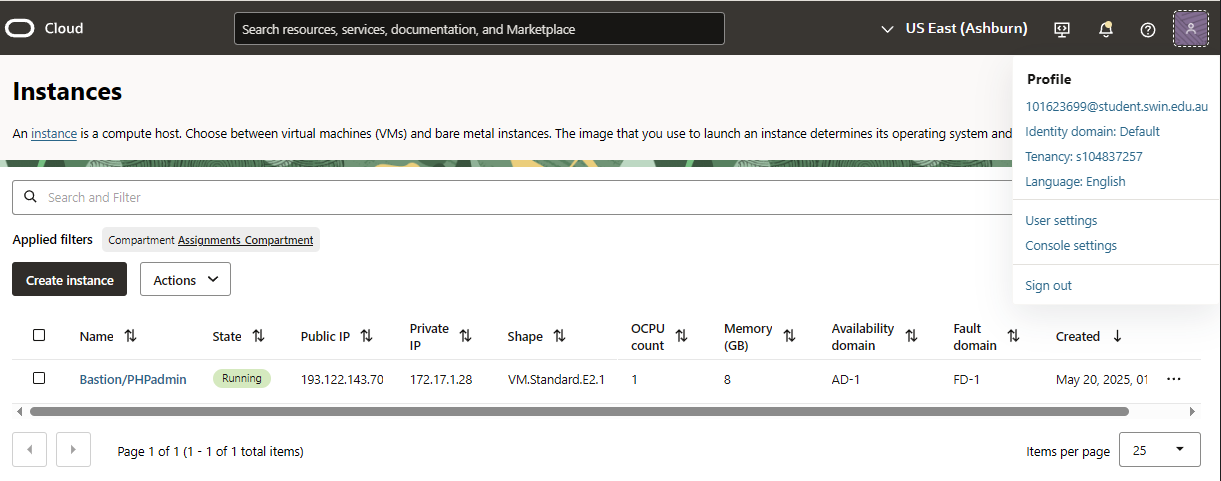
*Fig 5: Network Security Group: Web-tier NSG*

*Fig 6: Default Security List (For DB Private Subnet)* 

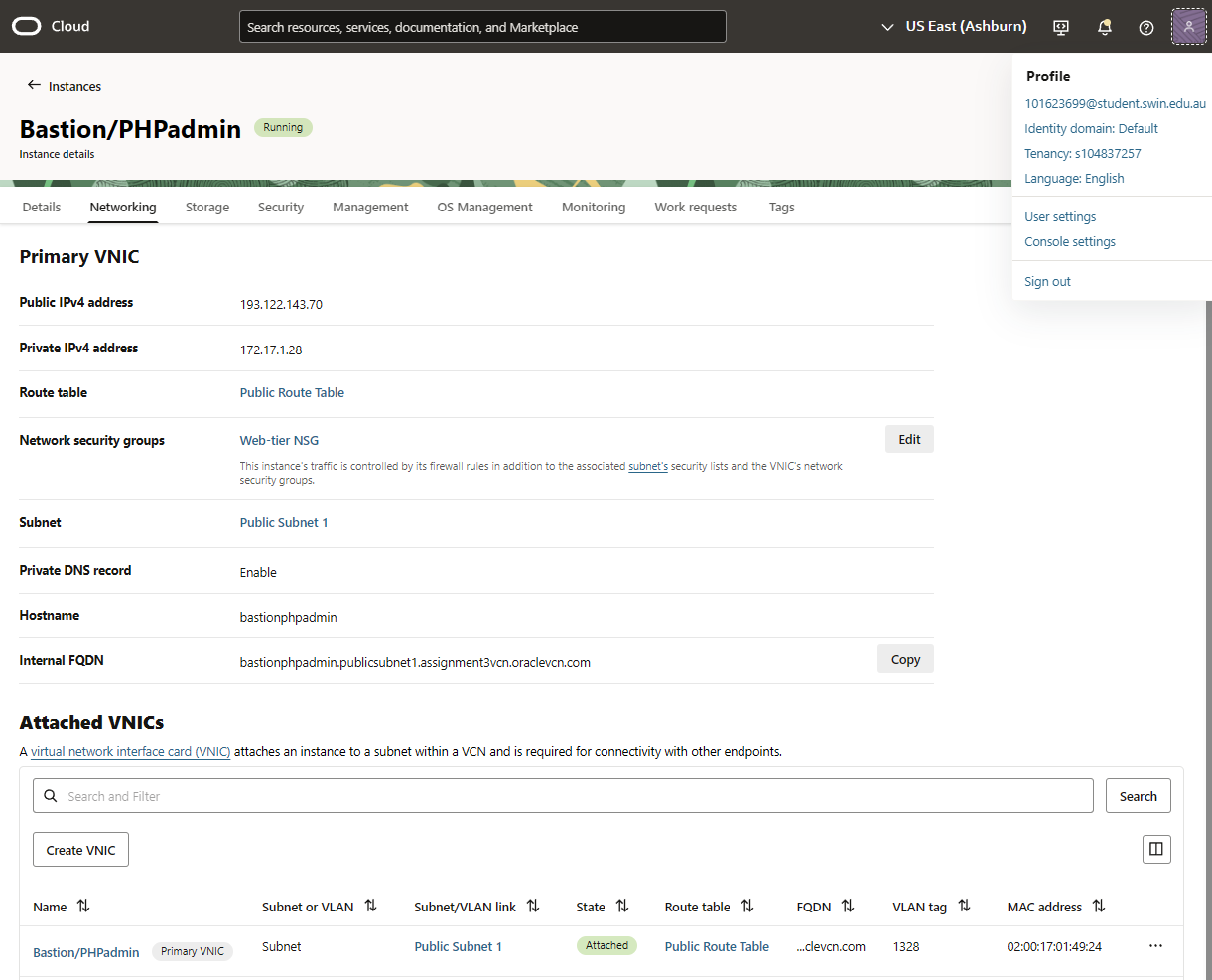
*Fig 7: Security List: PublicSubnet1SL*

## C. OCI Bastion Webserver Instance and Heatwave Database Instance

The Bastion/PHPadmin server was created in the public subnet, the name and IP addresses (public address is ephemeral) can be seen in fig 11. The security and route table association are illustrated in fig 12, where the public route table is associated.

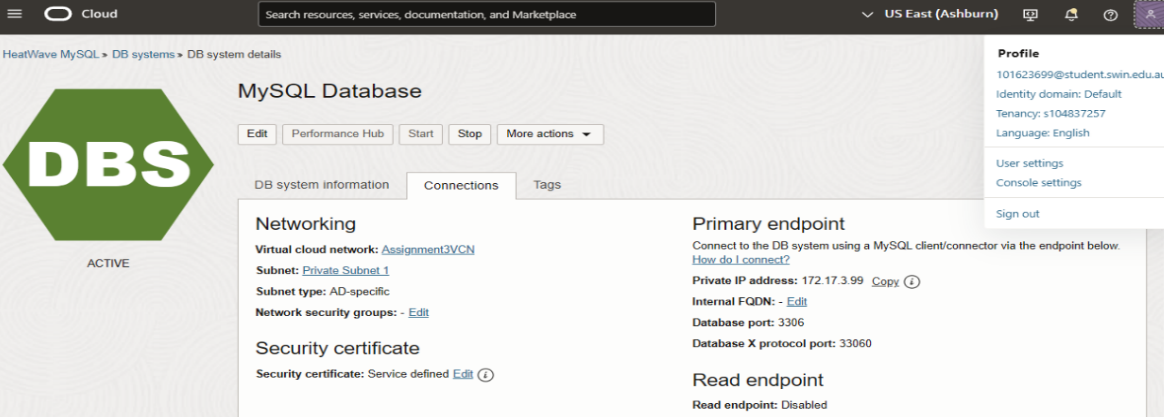


*Fig 8: Bastion/PHPadmin OCI Instance*

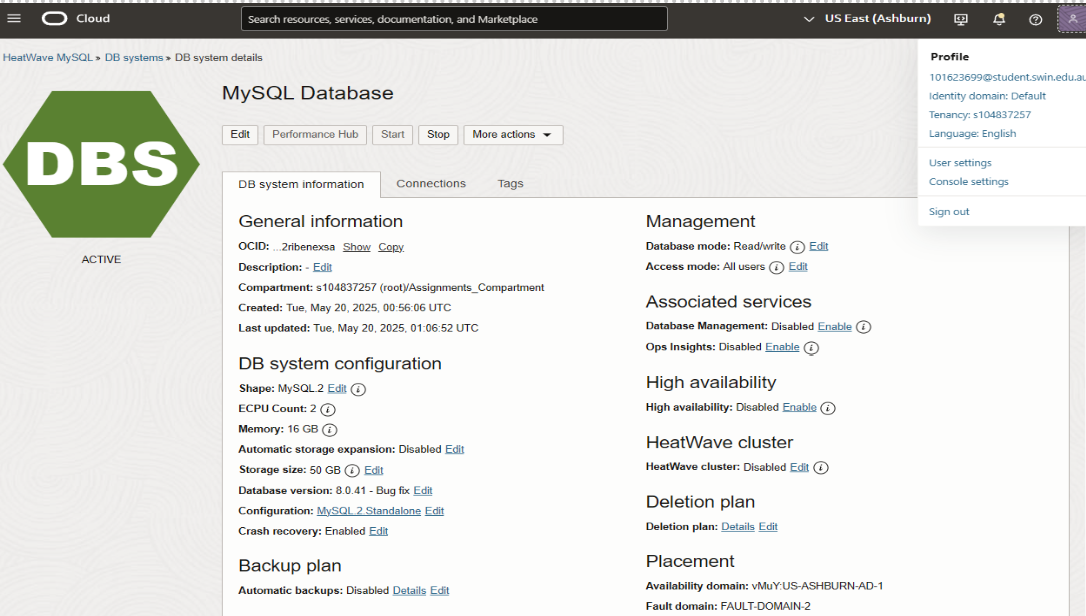


*Fig 9: Bastion/PHPadmin Route Table, Subnet and NSG Association*

The MySQL Database was deployed using OCI’s Heatwave Interface; the endpoint, subnet association and VCN association can be seen in fig 13. The database was configured to have the shape as MySQL.2, 2 ECPUs, 8.0.41 DB version, MySQL 2 Standalone config and no heatwave clustering with no backups according to specification (fig 14).



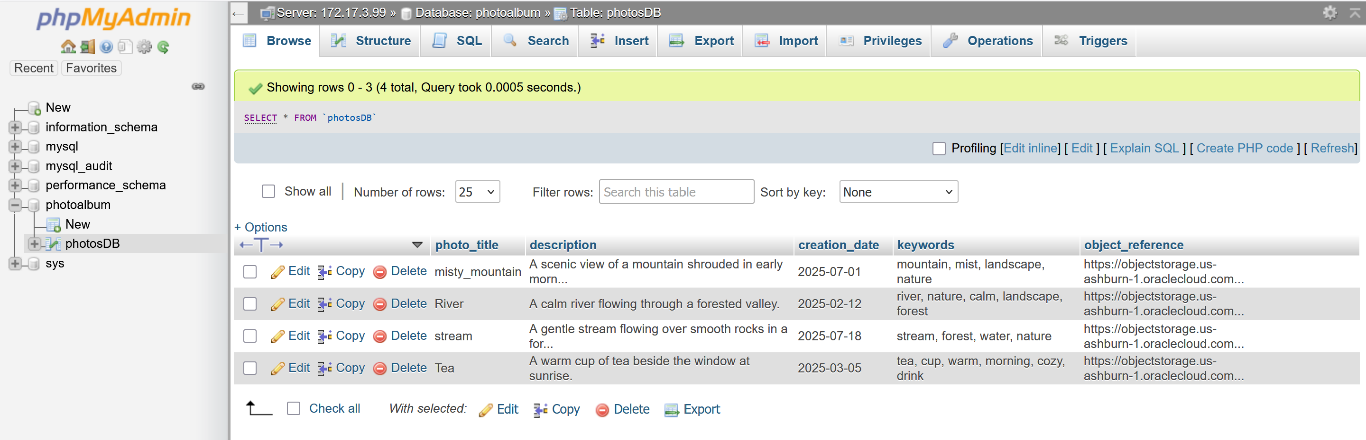
*Fig 10: MySQL Database Endpoint, VCN and Private Subnet Association*



*Fig 11: MySQL Database Deployment Details*

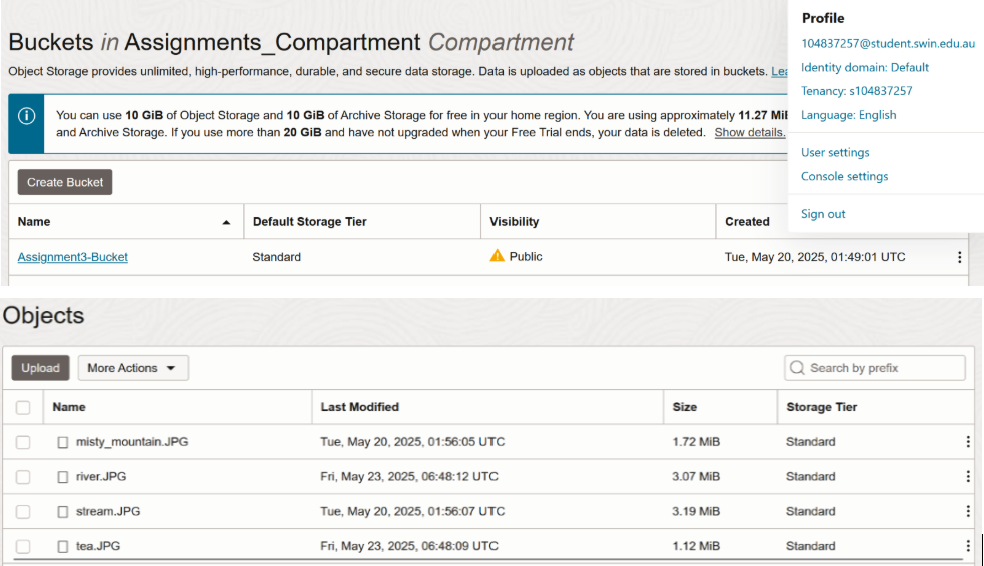
## D. MySQL Database Deployment on OCI

In OCI, a managed MySQL database (version 8.0.41, MySQL2 shape, 50GB storage) was deployed in Private Subnet 1, with public access disabled for security. It used OCI’s default security list, allowing ingress only from the OCI Bastion/Web Server and the AWS Bastion/Web Server. The database, named photoalbum, had one table called photosDB, with fields: photo\_title, description, creation\_date, keywords, and object\_reference—storing the photo's name, caption, date, tags, and image URL. It was managed using phpMyAdmin, accessed securely from the AWS server via the VPN tunnel.

*Fig 12: photosDB table schema and data records from phpMyAdmin*

## E. Object Storage Configuration on OCI

A bucket named Assignment-3\_bucket was created using OCI’s Object Storage service. Public access was granted at the bucket level, which meant all uploaded photos were accessible via direct public URLs. This removed the need to individually modify permissions for each file. Four sample images were uploaded through the OCI console and tested using an incognito browser window to confirm public access worked as intended.



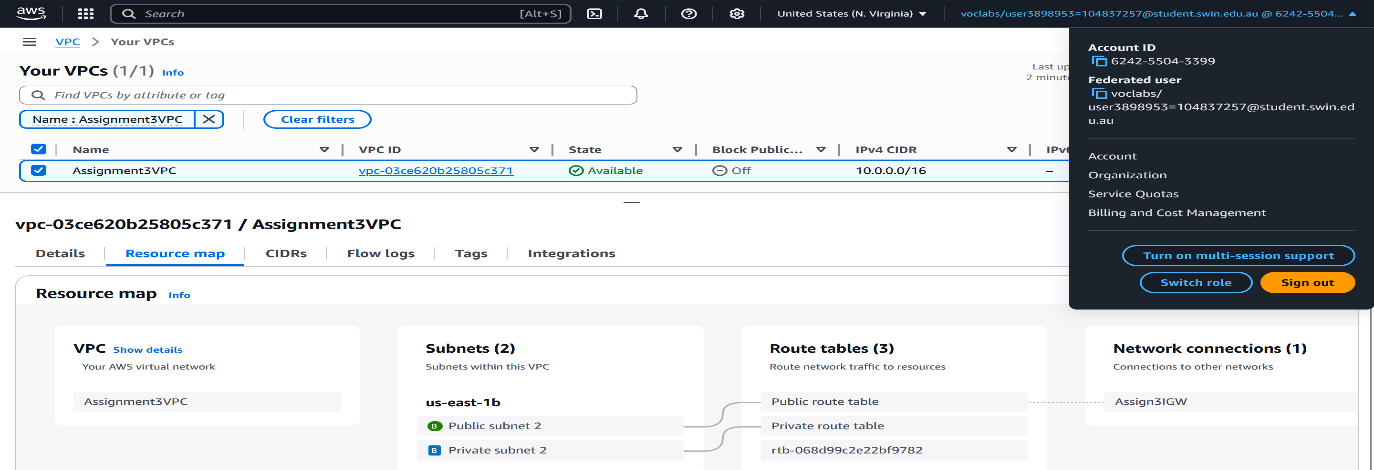
*Fig 13: Public Access Settings and Uploaded Images in Object Storage*

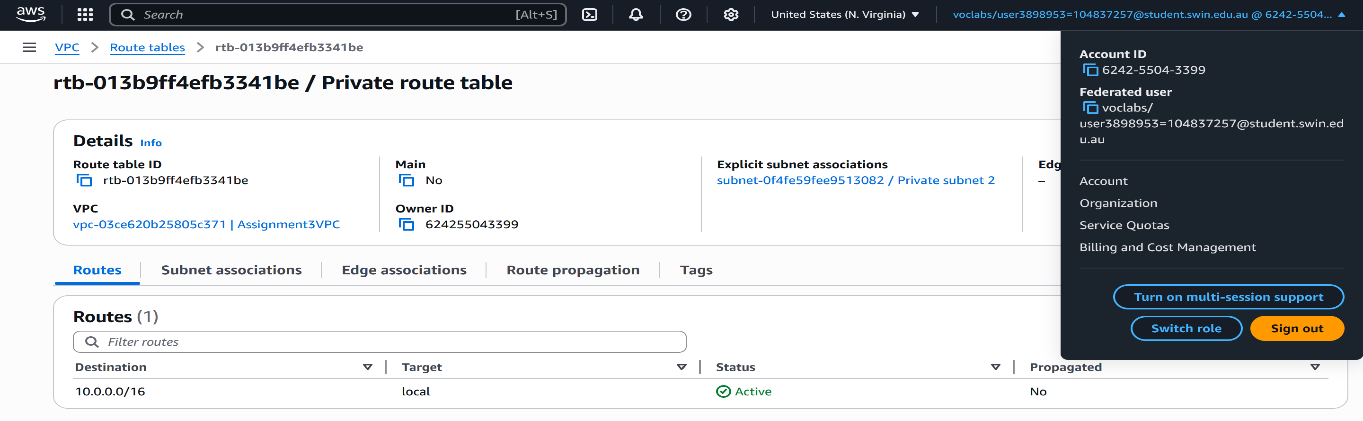
# III. AWS INFRASTRUCTURE SETUP

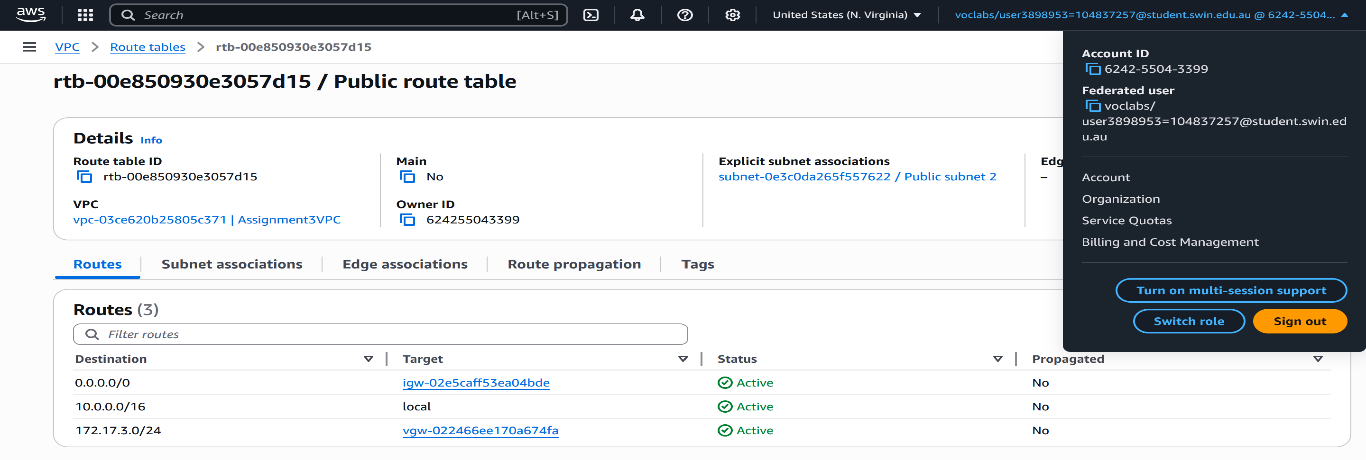
## A. Virtual Private Cloud, Route Tabling and Subnet Configuration

To begin the deployment, a custom Virtual Private Cloud (VPC) named "Assignment3VPC" was created within the us-east-1 region in AWS. The CIDR block assigned to this VPC was 10.0.0.0/16, inside this VPC two subnets were designed for a public and private subnet. Public Subnet 2 with the IP range 10.0.2.0/24 and Private Subnet 2 (10.0.4.0/24) were created for AWS, with the public for the Bastion/Web server and the private used to host a test instance.

A public route table was created and attached for the public subnet. Likewise, a private route table was created and the attached to the private subnet. Public subnets were connected to an Internet Gateway using the public route table that allowed internet access. The private subnet was connected to a separate route table that did not include internet access. The OCI database subnet CIDR (172.17.3.0/24) and the target AWS VPG to reach it, were added in the public route table, allowing the AWS bastion/web server to access the Database on the OCI Cloud through the VPN tunnel.

*Fig 14: Assignment3VPC, its CIDR, Subnet and Route Table Associations*

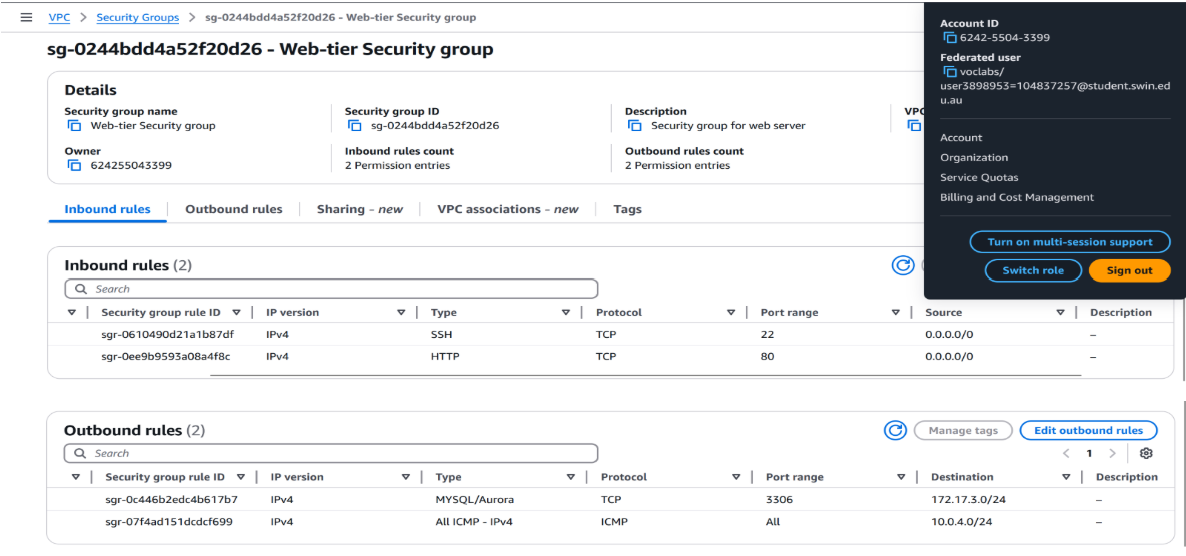
*Fig 15: Routes associated with Private Route Table*

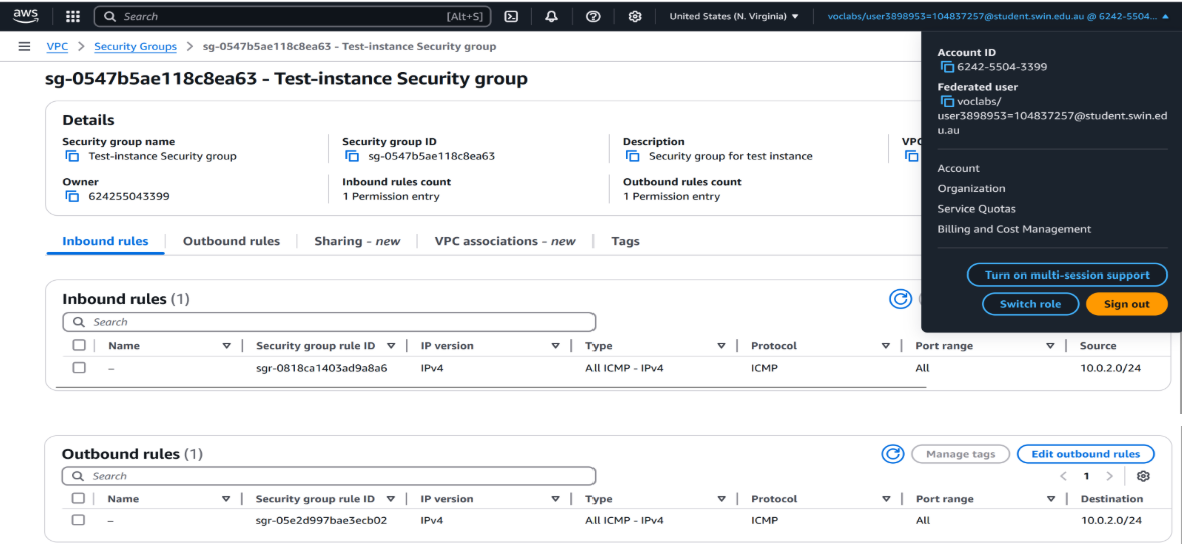
*Fig 16: Routes associated with Public Route Table*

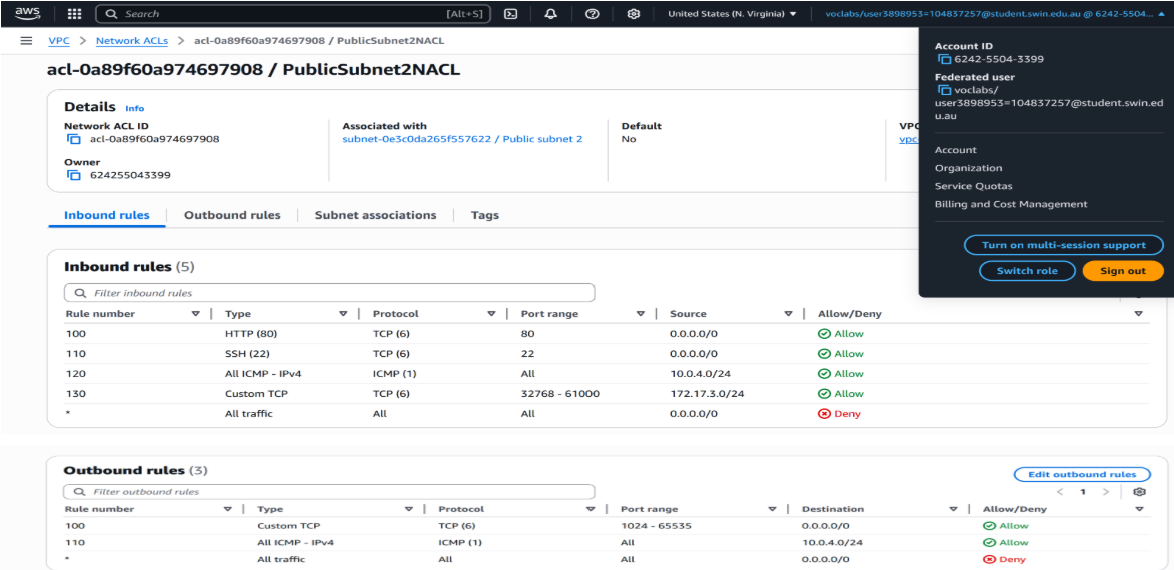
## B. Security Groups and Network ACLs

On AWS, the Web-tier Security Group controlled access to the Bastion/Web Server. It allowed SSH (port 22) and HTTP (port 80) from anywhere for remote access and public hosting. Outbound traffic was restricted to only what was needed - MySQL (port 3306) to the OCI database (172.17.3.0/24) and ICMP to the test instance (10.0.4.0/24), ensuring private, secure communication.

The PublicSubnet2NACL allowed inbound traffic on ports 22, 80, ICMP, and ephemeral ports (32768–61000), while outbound traffic was limited to ports 1024–65535 and ICMP to support return traffic and diagnostics. The Test-instance Security Group allowed only ICMP both ways, enabling ping tests while blocking everything else. All traffic was tightly controlled under the least privilege principle—only essential communication was permitted, with everything else denied maintaining strong security.

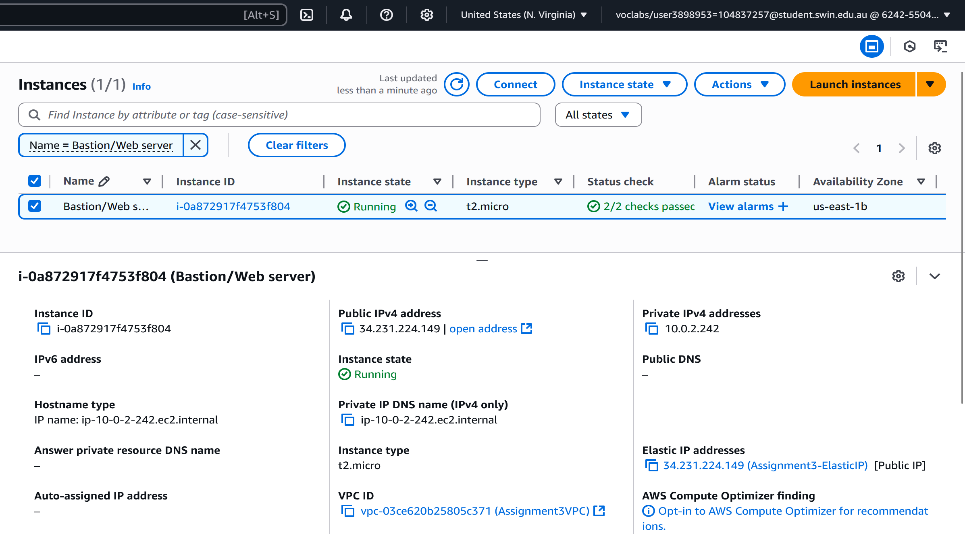
*Fig 17: Web-tier Security group with inbound and outbound rules*

*Fig 18: Test-instance Security group with inbound and outbound rules*

*Fig 19: PublicSubnet2NACL with inbound and outbound rules*

## C. Web server setup on the AWS bastion web server and attaching elastic IP address

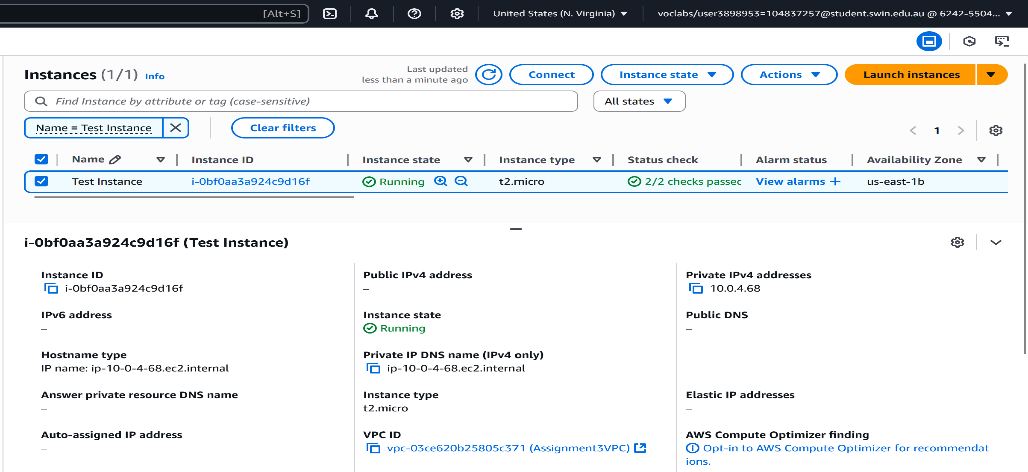
A script was run on the AWS bastion/web server to install Apache, PHP, and phpMyAdmin to deploy and host the photoalbum web site. To keep the public IP address as the same, even after restarts, an Elastic IP was created and attached.



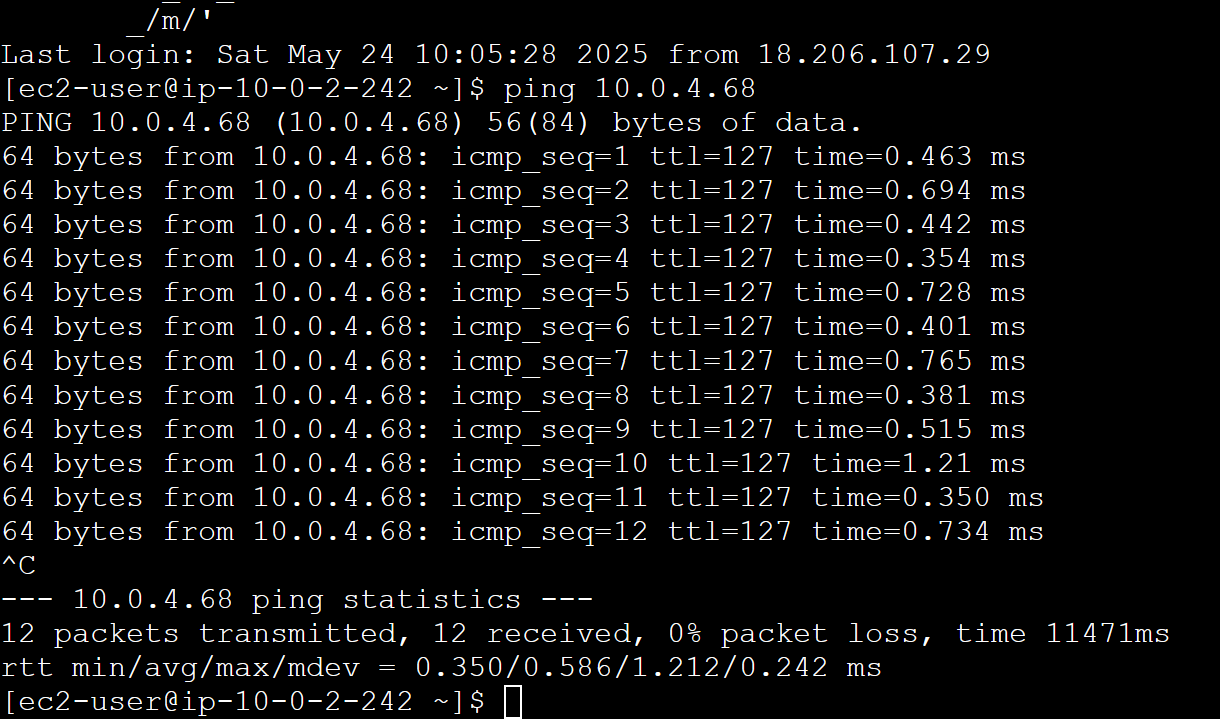
*Fig 20: Bastion/Webserver with the elastic IP address*

## D. ICMP PING connectivity testing from Bastion to Test Instance

The test instance resides in the private subnet (10.0.4.0/24) and its private IP address is 10.0.4.68. An ICMP ping was carried out from the bastion/webserver to this test instance to test connectivity and this was successful as shown in the below figure x.



*Fig 21: Test Instance with its Private IP Address*

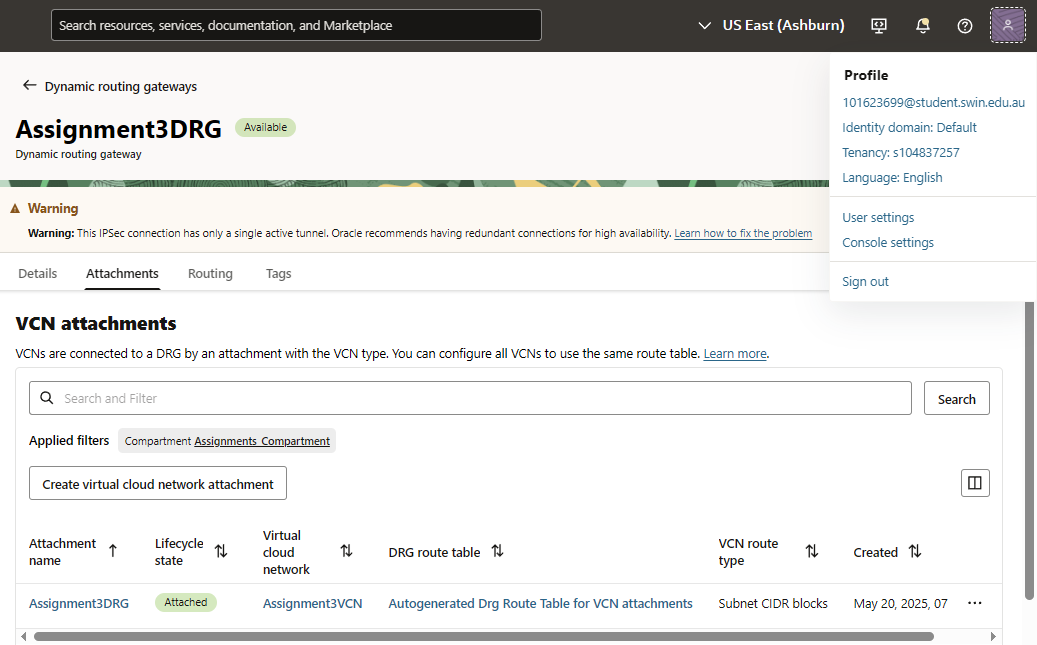


*Fig 22: Successful ICMP ping from Bastion to Test Instance*

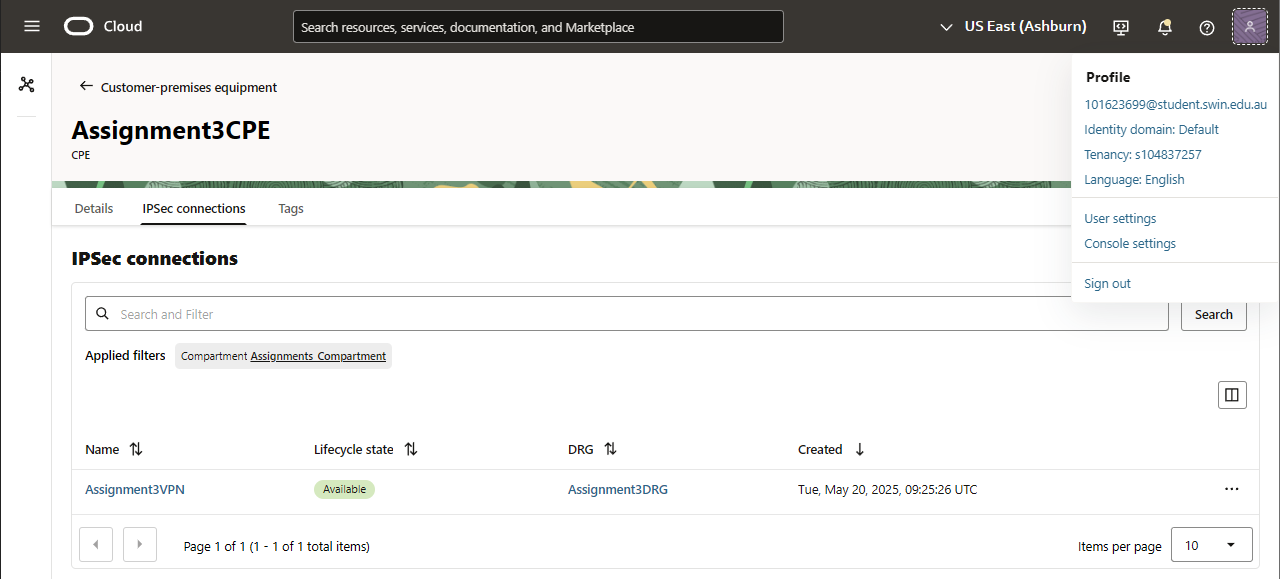
# IV. SITE-TO-SITE, TUNNELING AND VPN CONFIGURATION

## A. On OCI

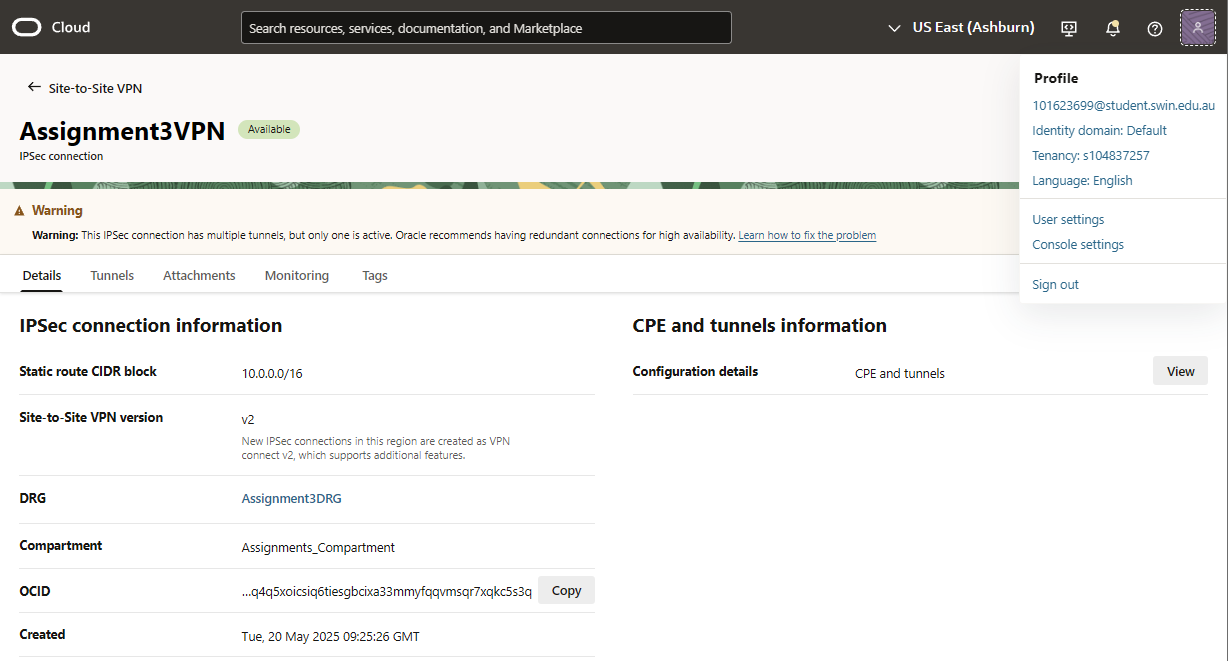
The first step to establish the VPN was to create a dummy AWS VPN and then create in an OCI Dynamic Routing Gateway (DRG) (Fig X) and attach it to the VCN. This DRG (Fig X) acts like a virtual router that can send traffic to other networks. Next, a Customer Premises Equipment (CPE) (Fig X) object was created in OCI using a dummy AWS VPN tunnel IP address. An IPsec connection was then formed between OCI and AWS using a shared secret. Finally, the private subnet route table was updated to send any traffic bound for 10.0.0.0/16 through the DRG. The tunnels that are interconnected have one that is not configured portrayed in fig X.



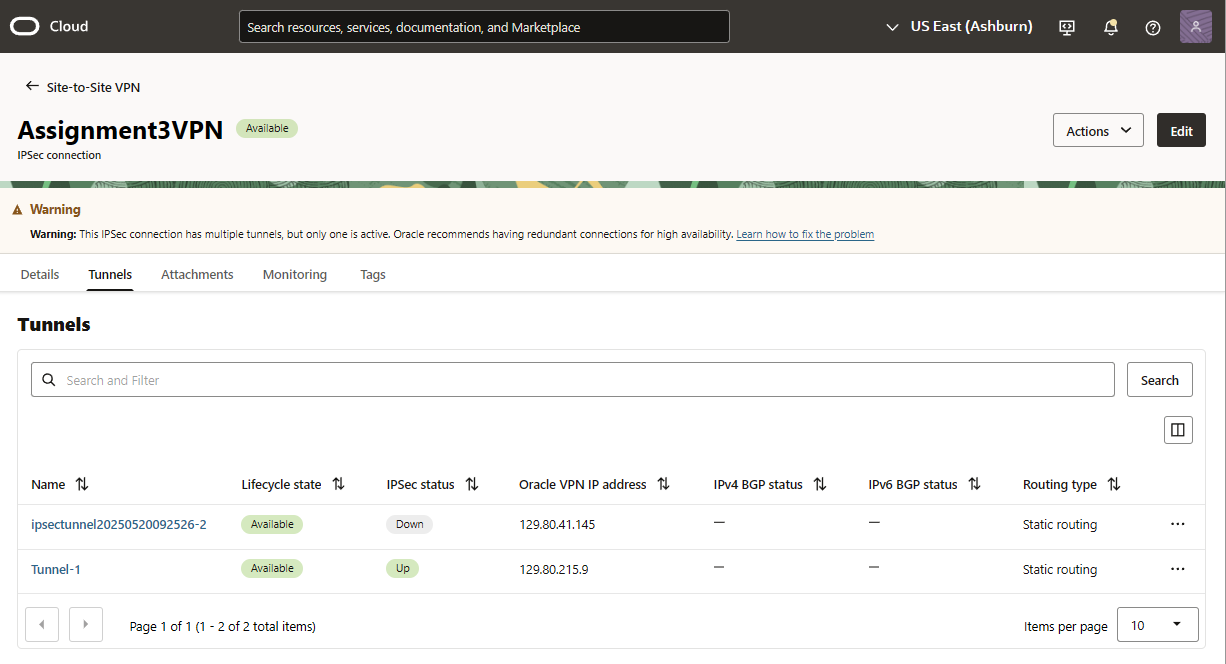
*Fig 23: Assignment3DRG OCI DRG with VCN Attachment*



*Fig 24: Assignment3CPE OCI CPE with VPN and DRG Attachments*



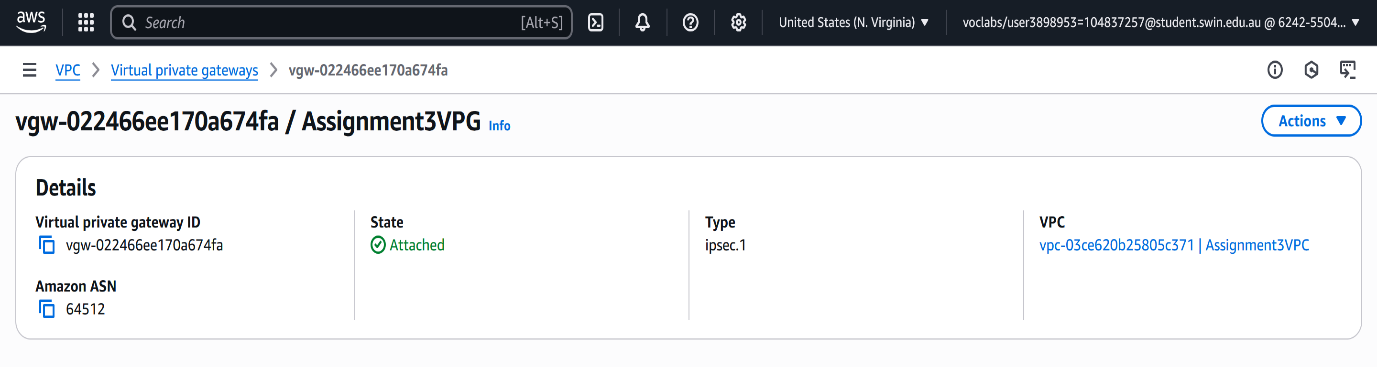
*Fig 25: Assignment3VPN OCI VPN with AWS CIDR Block*

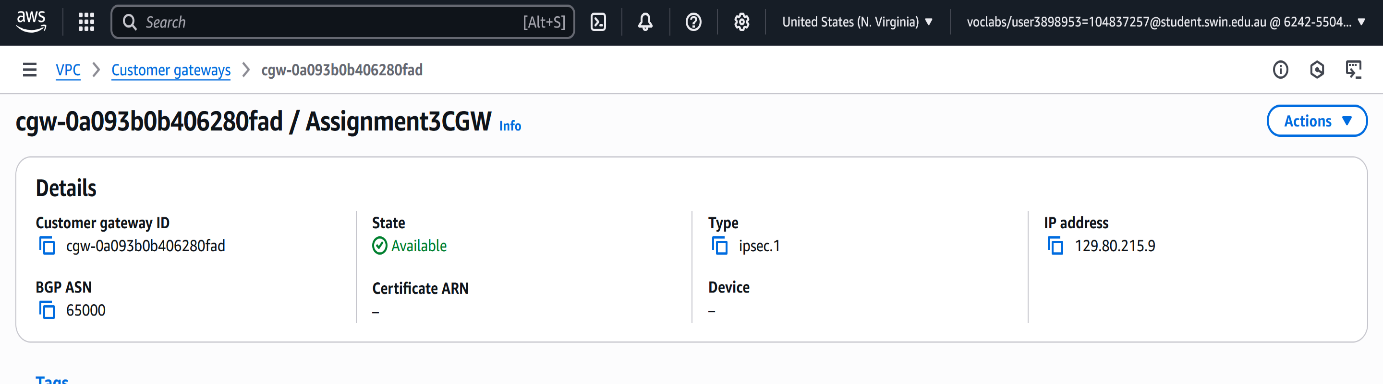


*Fig 26: Assignment3VPN OCI Tunnel Setup (1 Down and 1 Up)*

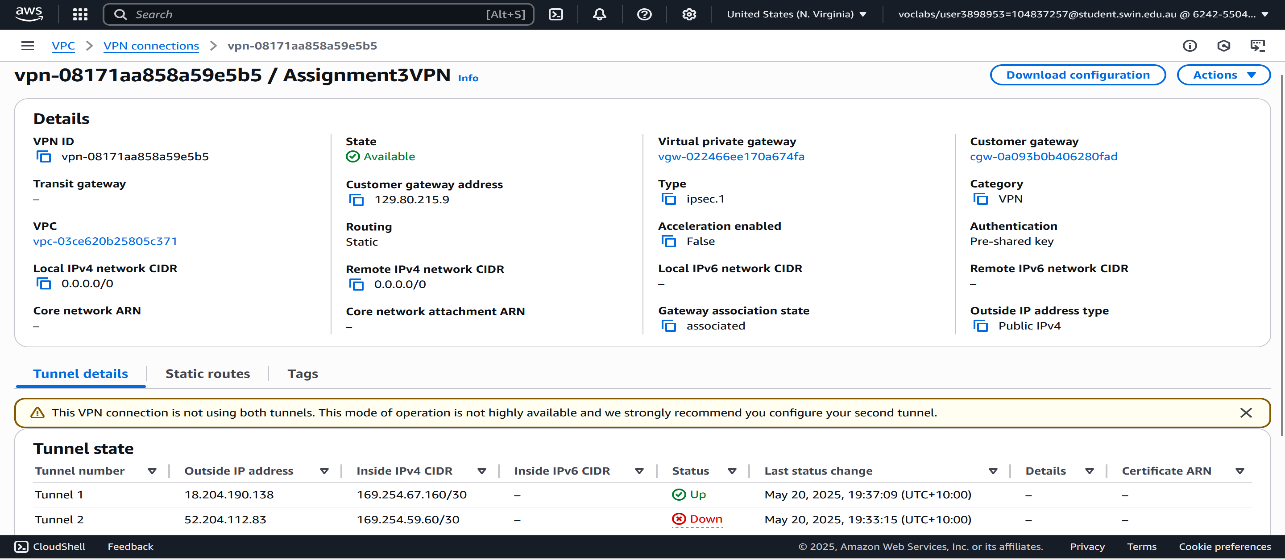
## B. On AWS

In AWS, a Virtual Private Gateway (VGW) was created and attached to the VPC. A Customer Gateway (CGW) was then defined using the public IP of the OCI VPN endpoint. A VPN connection was established by combining the VGW and CGW with the same shared secret used on the OCI side. To complete the setup, the route table for the AWS private subnets was updated to forward traffic bound for 172.17.0.0/16 through the VPN tunnel. Tunnel status was confirmed to be “UP” on both sides through the AWS and OCI dashboards.

*Fig 27: Assignment3VPG with AWS VPG to VPC attachment*



*Fig 28: Assignment3CGW with outside Public IP of the OCI side of the VPN tunnel*

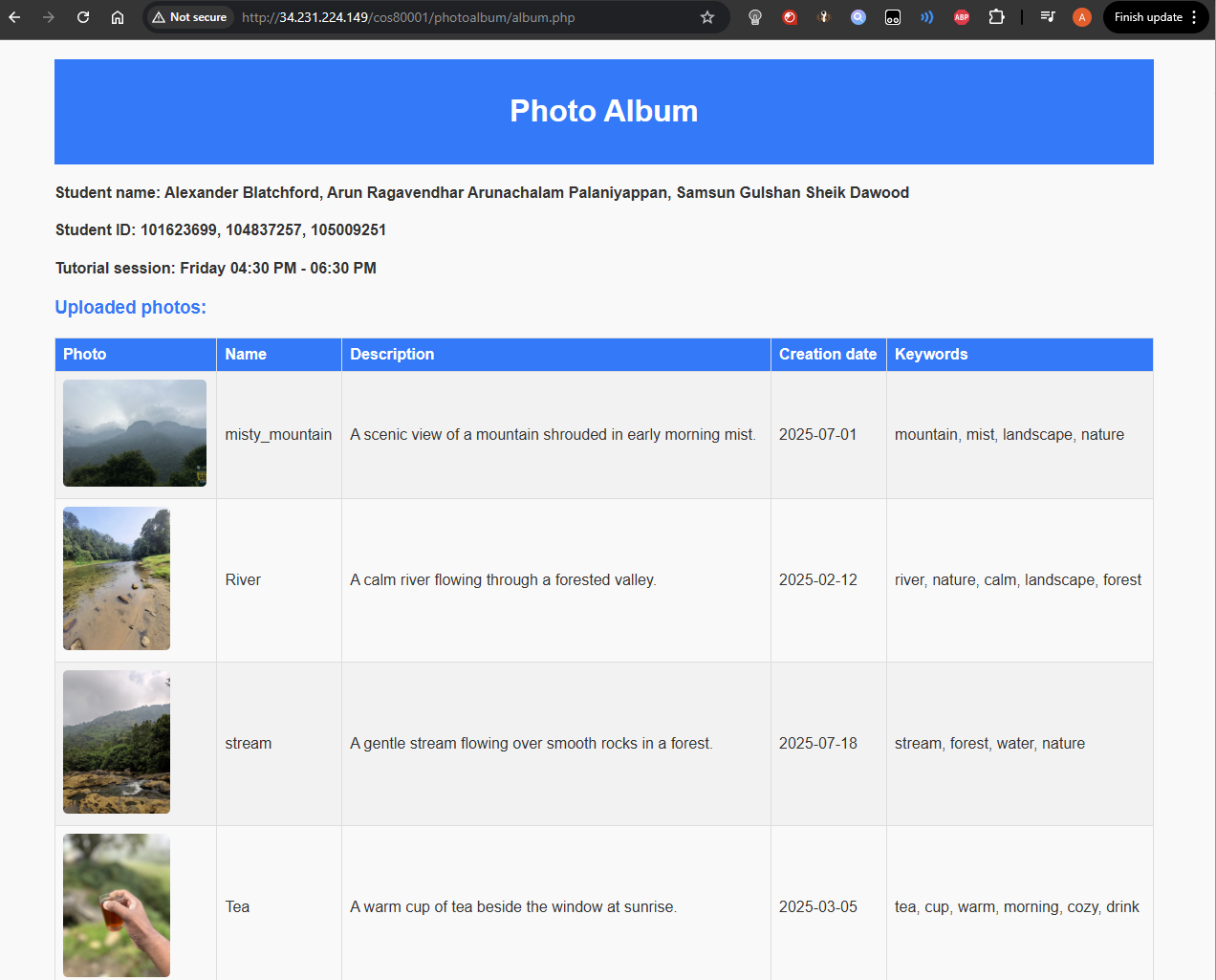


*Fig 29: Assignment3VPN AWS Tunnel Setup (1 Down and 1 Up)*

# V. WEB APPLICATION DEPLOYMENT AND CONNECTIVITY

The PHP web application named photoalbum\_v3.0 was deployed inside the EC2 instance under the /var/www/html/cos80001/photoalbum/ directory. The constants.php file was updated to use the private IP address of the OCI database, ensuring database traffic did not leave the VPN.

The application was tested by loading the album page via the public Elastic IP. It successfully displayed all images and their associated metadata from the MySQL database detailed in fig X



*Fig 30: Web Deployment of photoalbum.php*

# VII. COMPARISON OF CLOUD CONNECTION METHODS

For this setup, three ways to connect AWS and OCI were considered: AWS Direct Connect, OCI FastConnect, and IPsec VPN.

Direct Connect and FastConnect create a private connection between the two clouds. This means the data doesn’t travel over the internet—it goes through a dedicated direct line managed by the cloud provider or a network partner. These types of connections are more stable and have lower delays, which is useful for big businesses that run large systems all the time. But they are expensive and take time to set up. They often require dealing with third-party network providers and physical connection points.

IPsec VPN works differently. It uses the public internet to send data between clouds, but the data is encrypted before it’s sent and decrypted at the other end. This keeps the data secure even though it’s going over the internet. It doesn’t need any physical setup and can be done through software in just a few steps. The only downside is that internet traffic can sometimes be slower or less reliable.

For this project, IPsec VPN was chosen because it was free, quick to set up, and secure enough for the task. The goal was to connect AWS and OCI in a safe and working way for testing and learning, not for running a high-traffic system. So, IPsec VPN was the simplest and most practical option.

# VIII. CHALLENGES AND LEARNINGS

Some issues came up during the setup. The first problem was with the VPN connection. The tunnel could not be brought up at first, and traffic between AWS and OCI wasn’t working. This was because the route tables in both clouds were missing the correct CIDR entries. After adding the proper routes and pointing them to the VPN gateways, the tunnel came up and the connection started working as expected.

Next, the AWS web server couldn’t connect to the OCI MySQL database. This was due to a missing rule in the OCI Web-tier NSG, which didn’t allow traffic on port 3306. Once the rule was added to allow traffic from the AWS subnet, the issue was resolved. There was also a problem with pinging the test instance in AWS. The pings failed because the NACL didn’t have both inbound and outbound rules for ICMP. Since NACLs are stateless, both directions need to be set manually. This was fixed by updating the NACL and the test instance’s Security Group.

These problems helped the team understand how routing and security rules work differently in AWS and OCI, and the importance of allowing only the traffic that is needed as per system configuration.

# IX. CONCLUSION

This assignment successfully demonstrated how to build and deploy a working multi-cloud system using AWS and OCI. The web server and application were hosted on AWS, while the MySQL database, phpMyAdmin interface, and object storage were deployed on OCI. The two environments were connected using an IPsec VPN tunnel that allowed secure, private communication without using public access for database traffic.

Each part of the setup followed the least privilege principle, with only required ports and IPs allowed in all security components. All systems were tested for connectivity and access, including successful ICMP communication between subnets and secure data flow from the web application to the remote database. The deployment met all the project requirements and provided hands-on experience with setting up networks, routing, VPNs, firewalls, and cloud-based application hosting across two different platforms. The process also gave the team a clear understanding of how AWS and OCI differ in their network and security setups, and how to configure them to work together in a hybrid cloud setup.

This experience will be valuable in future industry roles that involve working with cloud infrastructure and setting up secure, multi-cloud systems.

# X. REFERENCES

[1] Amazon Web Services, “AWS Site-to-Site VPN,” *AWS Documentation*. [Online]. Available: <https://docs.aws.amazon.com/vpn/latest/s2svpn/>

[2] Oracle Cloud Infrastructure, “Setting Up an IPSec VPN,” *OCI Documentation*. [Online]. Available: <https://docs.oracle.com/en-us/iaas/Content/Network/Tasks/settingupIPSec.htm>

[3] Oracle Cloud Infrastructure, “Object Storage Overview,” *OCI Documentation*. [Online]. Available: <https://docs.oracle.com/en-us/iaas/Content/Object/Concepts/objectstorageoverview.htm>

[4] Oracle Cloud Infrastructure, “OCI MySQL Database Service,” *OCI Documentation*. [Online]. Available: <https://docs.oracle.com/en-us/iaas/mysql-database/doc/index.html>

[5] Amazon Web Services, “Elastic IP Addresses,” *AWS Documentation*. [Online]. Available: <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/elastic-ip-addresses-eip.html>