Assessment Title

Multi-Cloud Security Integration and Project Management

CCC604| Assessment-7

The Student Name | Geni Bahar
Student ID | 270850228

Group | Student_Group_01-FT
Team Members | Yusof Sharifzai (270707435), Hari Tamang (270860695)

Tutor/Assessor | Ovesh Vohra, Riddhi Joshi

Programme
Diploma in Cloud Computing and Cyber Security (120 Credits)

Course
CCC 604: Multi-Cloud Security Integration and Project Management
(Level 6, 30 Credits)

Date: 28th June 2026

Table of Contents

Task 1: Project Planning, Design, and Assignment of Group and Individual Roles

Introduction to the Project and Group

Secure Multi-Cloud Migration of Yoobee College WordPress Infrastructure (AWS + Azure + Fortinet)

In this project, our group planned and built a multi-cloud setup for Yoobee College's WordPress-based infrastructure. The main workload was hosted in AWS, while Azure was used for identity, DNS, RBAC, MFA and access testing. Fortinet FortiGate was added as the main security component for firewall control, VPN connectivity and traffic inspection.

The focus was not just to create resources, but to show how the setup can be managed securely. In AWS we used services such as VPC, EC2, RDS, ELB, Auto Scaling, S3, Lambda, CloudWatch, SNS, IAM and ACM. In Azure we used the subscription, RBAC, Azure AD tenant, MFA and Azure DNS. FortiGate was included to show how inter-cloud traffic can be controlled instead of leaving the two cloud environments separate.

We believe this design fits the assessment because it brings together the main areas we worked on in the course: cloud networking, identity, DNS, monitoring, backup and security. It also showed us that making AWS, Azure and FortiGate work together is not always straightforward.

Group Members:

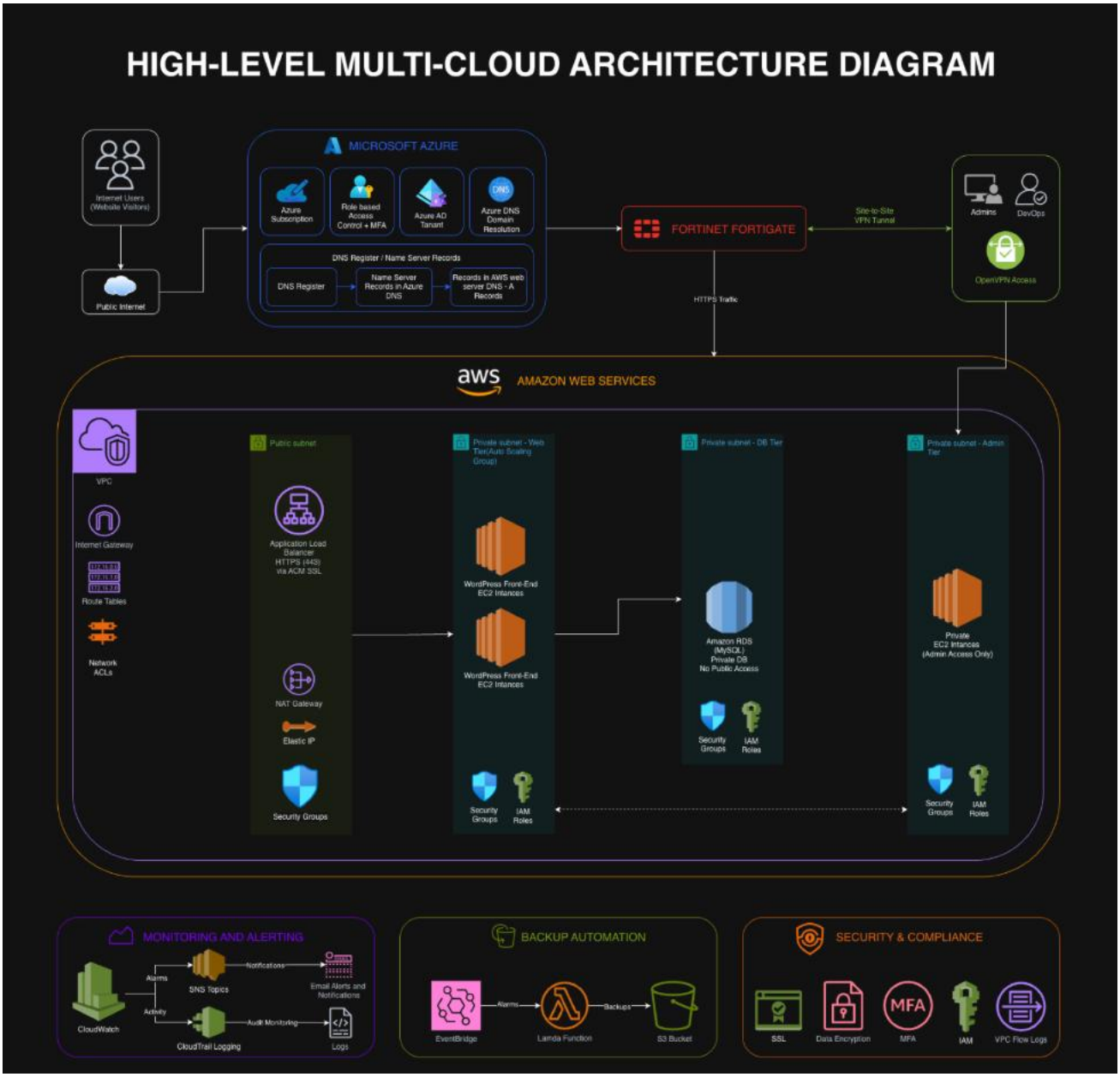
Name	Student ID	Role Summary
Yusof Sharifzai	270707435	Cloud Engineer - Azure and Co-Project Manager. Responsible for Azure subscription validation, Azure RBAC with MFA, Azure AD tenant configuration, Azure DNS setup, and VPN/HTTPS testing evidence.
Geni Bahar	270850228	Security Architect - AWS and Co-Project Manager. Responsible for AWS VPC, EC2, RDS, S3, Lambda, ELB, Auto Scaling, SNS, CloudWatch, OpenVPN, IAM, ACM, and Fortinet integration evidence.
Hari Tamang	270860695	Project Coordinator and Testing Support. Responsible for project scheduling, milestone tracking, screenshot collection, test verification support, and assisting with AWS/Azure/Fortinet configuration evidence.

Individual Student Responsibilities

Name	Role	Responsibilities
Yusof Sharifzai	Cloud Engineer (Azure) & Co-Project Manager	Create and validate the Azure subscription; configure Azure RBAC and MFA; create the Azure AD tenant using the group name; configure Azure DNS; link Azure DNS records to the AWS load balancer/DNS endpoint; test HTTPS access through Azure DNS; document Azure screenshots and support the final report.
Geni Bahar	AWS Security Architect and Co-Project Manager	Design and deploy the AWS VPC with public/private subnets, Internet Gateway, NAT Gateway, route tables, and security controls; configure EC2 WordPress hosting, RDS MySQL, IAM roles, S3, Lambda snapshot automation, CloudWatch monitoring, SNS notifications, ACM HTTPS, ELB/Auto Scaling, OpenVPN private access, and Fortinet security integration. Also responsible for AWS evidence, technical explanation, and security design alignment.
Hari Tamang	Project Coordinator and Testing Support Engineer	Maintain the project schedule and milestone tracking; support testing for HTTPS, VPN, load balancing, monitoring, and backup validation; assist with screenshot collection; support Fortinet/VPN evidence collection; help review final documentation for completeness, formatting, and submission readiness.

Project Layout and Diagram (Geni)

Figure 1 shows the final high-level design we followed for the project. The diagram helped us separate the main parts of the environment, such as public access, application hosting, database, VPN, monitoring and backup. AWS hosts the WordPress workload, Azure supports identity/DNS/testing, and FortiGate sits as the security control point between the cloud environments.



Step-by-step Checklist for Timely Completion (Yusof)

Phase 1: Project Planning - Yusof Leads, Geni Supports (11-18 May 2026)

- Finalise group roles (Yusof = Azure, Geni = AWS, Hari = Project coordination)
- Define project scope and objectives
- Create architecture diagram (with Fortinet placement)
- Complete Task 1 planning document

Phase 2: AWS Implementation - Geni Leads (18-31 May 2026)

- Create VPC with public/private subnets, IGW, NAT Gateway, route tables
- Launch EC2 Amazon Linux instance
- Configure Security Groups with required ports
- Create IAM roles (EC2 + S3 access, Lambda function role)
- Deploy RDS MySQL as backend database
- Configure Application Load Balancer + ACM SSL/TLS certificate
- Set up Auto Scaling group
- Create Lambda function for daily EC2 snapshot
- Set up S3 bucket for snapshot backups
- Configure CloudWatch to monitor EC2 start/stop events
- Set up SNS topic with Yoobee student email notification
- Deploy OpenVPN for private IP access

Phase 3: Azure Implementation - Yusof Leads (1-7 June 2026)

- Create Microsoft Azure subscription
- Verify subscription balance and validity
- Configure RBAC with MFA on the account
- Create Azure AD tenant with our group name
- Configure Azure DNS zone
- Register or update DNS name server records and validate DNS propagation
- Configure Azure DNS records to point to the AWS load balancer endpoint and verify HTTPS access
- Test VPN client access to WordPress site via HTTPS

Phase 4: Fortinet Integration - Geni Leads, Yusof Supports (8-14 June 2026)

- Deploy FortiGate (AWS Marketplace)
- Configure VPN tunnels between AWS ↔ FortiGate ↔ Azure
- Define firewall rules and inspection policies
- Document inter-cloud traffic security controls
- Update architecture diagram with Fortinet placement

Phase 5: Risk Assessment & Incident Analysis - All Members (8-14 June 2026)

- Complete risk assessment table (minimum 5 risks with mitigations)
- Document the security issue identified during testing with detection and response timeline
- Document detection, response, and recovery plan
- Include CloudWatch, CloudTrail, Azure Monitor logs

Phase 6: Testing & Verification - All Members (15-21 June 2026)

- Test HTTPS access via Azure DNS name (from any location)
- Test OpenVPN connectivity to private EC2 IPs
- Verify ELB + Auto Scaling functionality
- Validate CloudWatch alarms trigger SNS email
- Test Lambda snapshot creation in S3 bucket
- Verify IAM least privilege and RBAC MFA

Phase 7: Documentation & Submission - All Members (22-28 June 2026)

- Compile all screenshots (showing settings)
- Write executive summary
- Complete final risk assessment and incident analysis appendices
- Export the final report as a single PDF document for LMS submission
- Submit to LMS by **Sunday, 28 June 2026, 11:59 PM**

Project Submission Guidelines

Item	Requirement
File Format	Single PDF document
Cover Page	Project title, team names, student IDs, submission date
Table of Contents	Required
Diagrams	Labelled architecture diagrams (AWS, Azure, Fortinet)
Screenshots	Each practical step in sequence, showing settings/outputs
Risk Assessment	Table with likelihood, impact, mitigation strategies
Incident Analysis	Security issue identified during testing with detection and response timeline
References	APA format if external sources used
Submission Platform	LMS
Deadline	Sunday, 28 June 2026, 11:59 PM
Academic Integrity	All technical configurations, screenshots, testing evidence, and explanations must be based on the group's own implementation. External sources must be cited using APA or another suitable academic referencing format.

Project Weekly Timesheet

Work	Dates	Task	Hours	Notes
1	11-17 May	Task 1 - Planning, roles, architecture diagram	6	All members contributed
2	18-24 May	AWS VPC, EC2, IAM, RDS setup	8	Geni leads, Yusof and Hari supports
3	25-31 May	AWS ELB, Auto Scaling, Lambda, S3, SNS, CloudWatch, OpenVPN	8	Geni and Hari leads, Yusof supports
4	1-7 June	Azure subscription, RBAC+MFA, Azure AD, DNS	7	Yusof leads, Geni and Hari Supports
5	8-14 June	Fortinet deployment, VPN tunnels, firewall policies + Risk assessment + Incident analysis	8	Geni and Hari worked on Fortinet evidence and risk/incident section, with Yusof supporting where needed
6	15-21 June	Testing (HTTPS, VPN, Load Balancer, monitoring) + Screenshots	7	Hari leads, Yusof and Geni supports
7	22-28 June	Final documentation, report writing, PDF submission	8	All members contributed jointly

Meeting Minutes

Date	Attendees	Summary of Discussion	Action Item
22/05/2026	Yusof Sharifzai, Geni Bahar, Hari Tamang	Initial planning discussion. Group roles, project scope, architecture direction, and task responsibilities were agreed.	Complete Task 1 planning document and confirm AWS/Azure/Fortinet responsibilities.
05/06/2026	Yusof Sharifzai, Geni Bahar, Hari Tamang	Progress review of AWS and Azure implementation. Discussed VPC, EC2, RDS, Azure subscription, RBAC, MFA, and Azure DNS progress.	Continue screenshot collection and verify that each service has labelled evidence.
14/06/2026	Yusof Sharifzai, Geni Bahar, Hari Tamang	Reviewed Fortinet integration, VPN testing, risk assessment planning, and incident scenario selection.	Complete VPN/Fortinet evidence and prepare risk and incident analysis sections.
28/06/2026	Yusof Sharifzai, Geni Bahar, Hari Tamang	Final documentation review. Checked report structure, screenshots, captions, testing results, and PDF submission requirements.	Finalise report, proofread, update table of contents, and export as a single PDF.

Project Scope and Objectives

Area	Details
AWS Services	VPC (public/private subnets, IGW, NAT), EC2, RDS (MySQL), S3 (backups), Lambda (snapshots), ELB + Auto Scaling, ACM (SSL), CloudWatch + SNS, OpenVPN, IAM (least privilege)
Azure Services	Subscription validation, RBAC with MFA, Azure AD tenant, Azure DNS integration
Fortinet	FortiGate firewall, VPN tunnels between AWS and Azure, inspection policies
Security Controls	Encryption at rest (KMS/S3), encryption in transit (SSL/TLS), centralised logging (CloudWatch, CloudTrail, Azure Monitor)
Tagging	All resources tagged with Key = **Student_Group_01-FT**, Value = Resource identity
Testing and Verification	HTTPS access through Azure DNS, VPN client connectivity, ELB/Auto Scaling validation, CloudWatch/SNS alert testing, Lambda snapshot validation, IAM/RBAC review, and screenshot-based evidence mapping.

Objectives (Measurable goals)

1. Create a fully working website hosted by AWS EC2 with RDS as the back end.
2. Use ELB (Elastic Load Balancer) and Auto-scaling for high-availability purposes.
3. Automate daily snapshots of our EC2 instances using Lambda and store them in an Amazon S3 bucket.
4. Create SNS notification to send emails to the student email address at Yoobee when there are any state changes made to an instance running in EC2.
5. Setup Azure Active Directory (AD) tenant and Role Based Access Control (RBAC) along with Multi-Factor Authentication (MFA).
6. Configure Azure DNS records to resolve the public WordPress website through the AWS load balancer endpoint and verify secure HTTPS access from an external client.
7. Deploy Fortinet FortiGate and configure firewall policies, VPN/routing settings, and security controls for inter-cloud traffic management.
8. Document at least 5 identified Risks & Mitigations into a Risk Assessment Table.
9. Write down one documented security incident identified during project testing.
10. Finalise a single PDF report with architecture diagrams, implementation steps, screenshots, testing evidence, risk assessment, incident analysis and verification results.

Risk Management Approach

Risk ID	Owner	Description	Likelihood	Impact	Mitigation Strategy
R1	Geni	IAM misconfiguration leading to privilege escalation	Medium	High	Enforce MFA, least-privilege policies, final project review and permission validation
R2	Geni	VPC misconfiguration (NAT/IGW/routing) causing downtime	Medium	High	Peer review network design, test in stages
R3	Yusof	Azure DNS misconfiguration causing website inaccessible	Low	High	Validate DNS records before final deployment, test with temporary subdomain
R4	Geni/Yusof	VPN tunnel failure between AWS and Azure	Medium	Medium	Redundant VPN tunnels, monitoring alerts on both sides
R5	Hari	Team size reduction	High	Medium	Shared PM responsibilities, weekly check-ins, task redistribution
R6	Geni	Lambda snapshot automation failure	Low	Medium	Test Lambda function in dev environment, enable CloudWatch alerts for execution errors
R7	Geni	FortiGate policy misconfiguration exposing inter-cloud traffic	Medium	High	Peer review of firewall rules, enable logging, test in isolated VPC
R8	Geni	SNS email notification not received by Yoobee student email	Low	Low	Verify email subscription confirmation, test with alternative email

Communication Strategy

Area	Details
Meeting Frequency	Twice weekly: Mondays (planning), Thursdays (progress review) Further dates to be discussed
Primary Tools	Microsoft Teams (meetings, chat), Email (For updates and Work Hours discussion)
Documentation Sharing	Google Docs (shared planning), Teams
Screenshot Storage	If needed: Shared Google Drive folder with subfolders
Progress Tracking	Team message, updated daily and weekly
Tutor Communication	Teams and if required Video/Voice recorded during the meeting, Email
Issue Escalation	Any blocking issue escalated to tutor within 24 hours

Task 2 | Multi-Cloud Project Implementation Using AWS and Azure Cloud Services, with Security Incident Identification and Troubleshooting (Geni)

Task 2: Part-A | Project Implementation using AWS Cloud Services (Geni)

In this section, we documented the AWS side of our implementation. AWS was used as the main platform for hosting the WordPress site. The aim was to create a secure and workable environment first, and then connect it with Azure DNS and Fortinet security controls.

For the AWS setup, we created a custom VPC with public and private subnets. We also configured the main security and support services such as Security Groups, IAM roles, CloudWatch, CloudTrail, S3, Lambda backup automation, OpenVPN, Application Load Balancer, Auto Scaling and ACM for HTTPS. This made the environment closer to a real cloud hosting design rather than only a single EC2 instance.

A key point in the implementation was to avoid exposing every resource directly to the internet. Public-facing components and private components were separated, and access was controlled mainly through Security Groups, NACL review, VPN access and IAM permissions.

Requirement	**Evidence included in report**
VPC and network segmentation	VPC, public/private subnets, route tables, Internet Gateway, NAT Gateway, Elastic IP, NACLs, and Security Groups
AWS compute and application hosting	EC2 Amazon Linux instance, WordPress deployment, application access testing
Database back end	RDS MySQL deployment, subnet/security group restrictions, snapshot evidence
High availability	Load Balancer, Auto Scaling Group, HTTPS/ACM configuration
Monitoring and notification	CloudWatch monitoring, CloudTrail auditing, SNS email notification
Backup automation	Lambda snapshot automation and S3 backup storage
Secure remote access	OpenVPN client access to private resources
Azure integration	Azure subscription, RBAC/MFA, Azure AD tenant, Azure DNS, VPN/HTTPS testing
Fortinet integration	FortiGate deployment, firewall/security control point, VPN/routing evidence

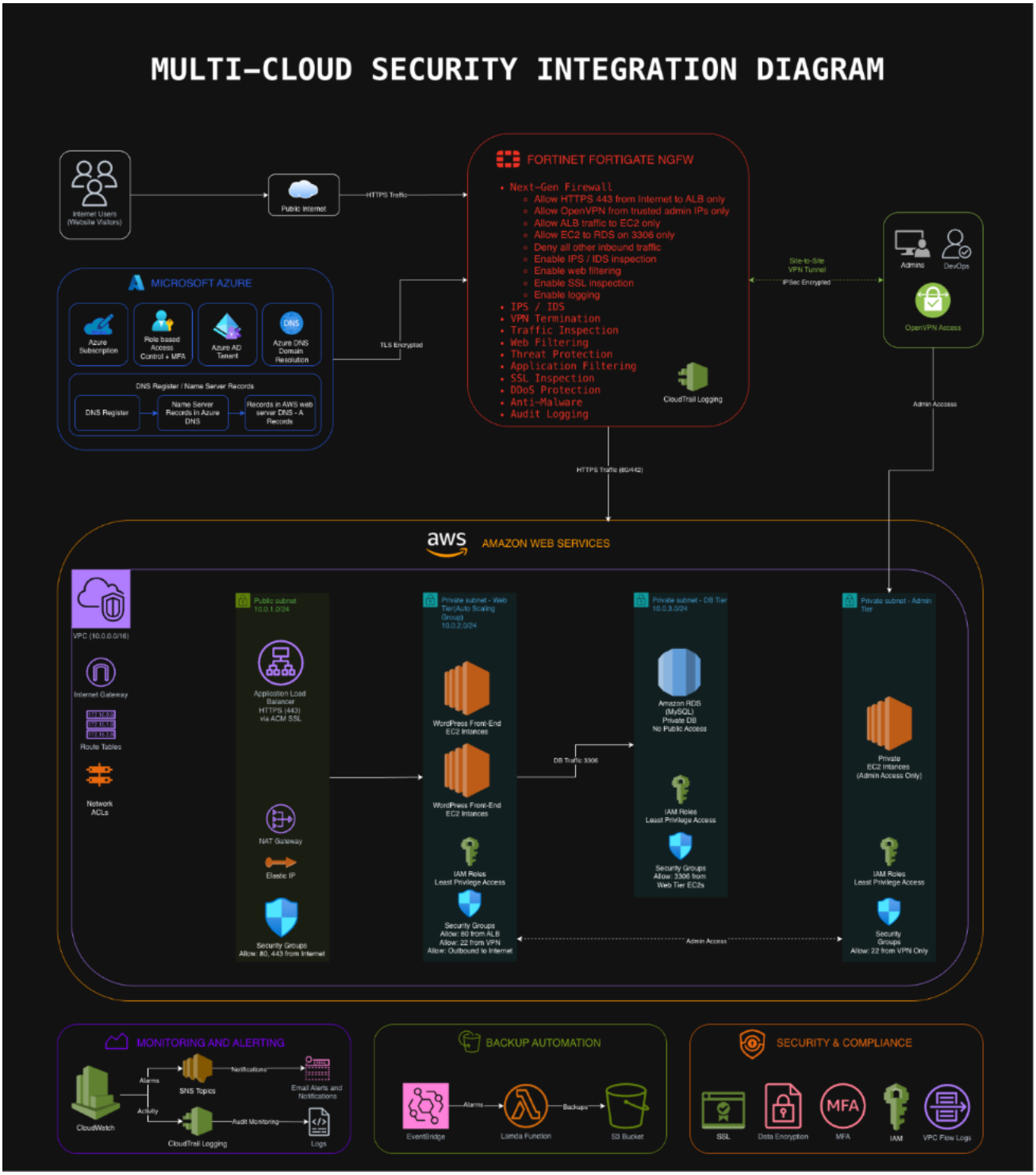
Testing and verification	HTTPS access, VPN access, monitoring alerts, backup validation, and screenshot evidence

AWS Network Architecture

For AWS, we started from the network layer and then added the application and security components. The custom VPC, public/private subnets, Internet Gateway, NAT Gateway, route tables, NACLs and Security Groups form the base of the setup. On top of that, EC2 was used for WordPress and RDS MySQL was used as the database. We also added monitoring, notification and backup services such as CloudWatch, CloudTrail, SNS, Lambda and S3.

Network Security Diagram

The diagram below shows how the AWS network was separated. Public-facing services are placed separately from internal application and database resources. This was important because we did not want the database or internal resources to be directly reachable from the internet.



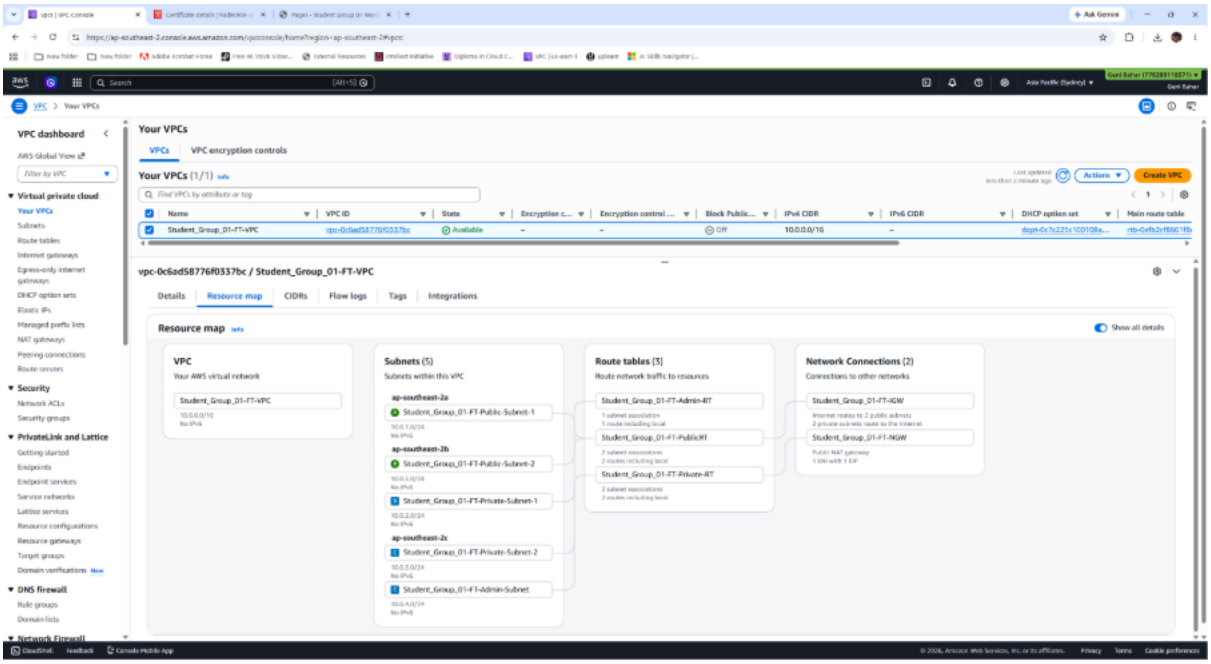
Virtual Private Cloud (VPC)

We created a dedicated VPC for this assessment so the project resources were kept inside their own network boundary. This became the base for the rest of the AWS configuration.

The VPC was then divided into public and private subnets. This made it easier to control which resources should be public and which resources should stay private.

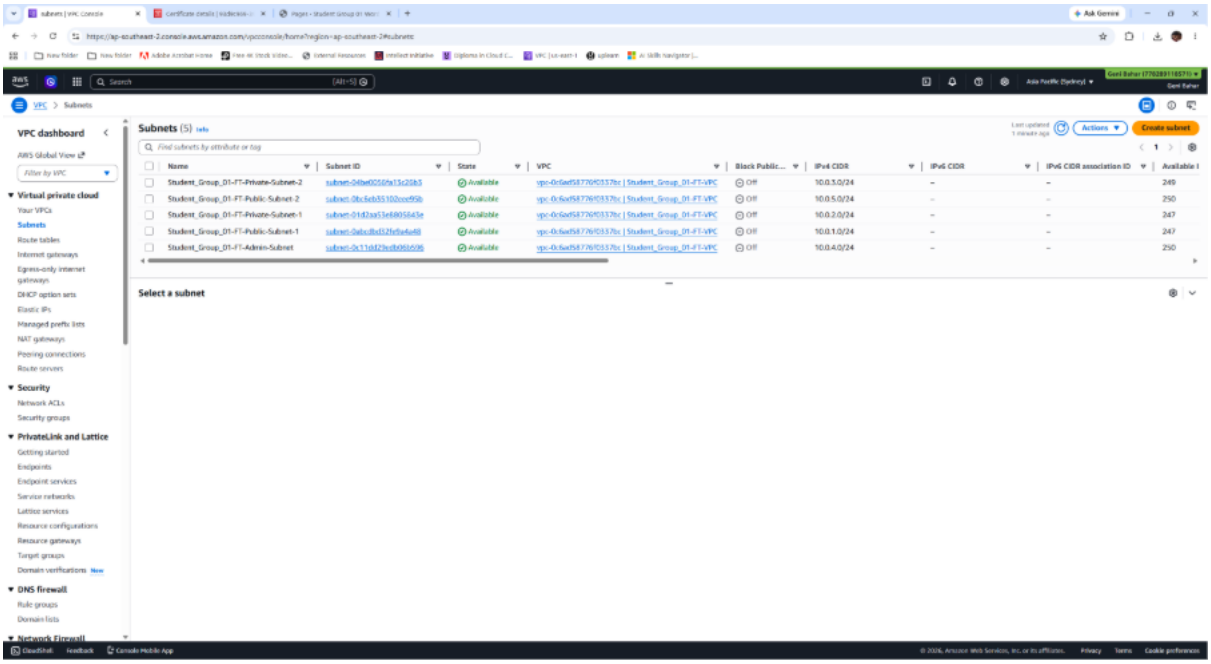
The screenshot below shows the VPC that was created for the project. All AWS services used in this part of the assessment were built around this network design.

Item	Configuration / Purpose
VPC Purpose	Isolated AWS network for WordPress and supporting services
Public Subnets	Host internet-facing resources such as load balancer/VPN access
Private Subnets	Host internal application/database components
Security Benefit	Reduces attack surface by separating public and private resources
Project Tagging	Resources tagged using group name and resource identity



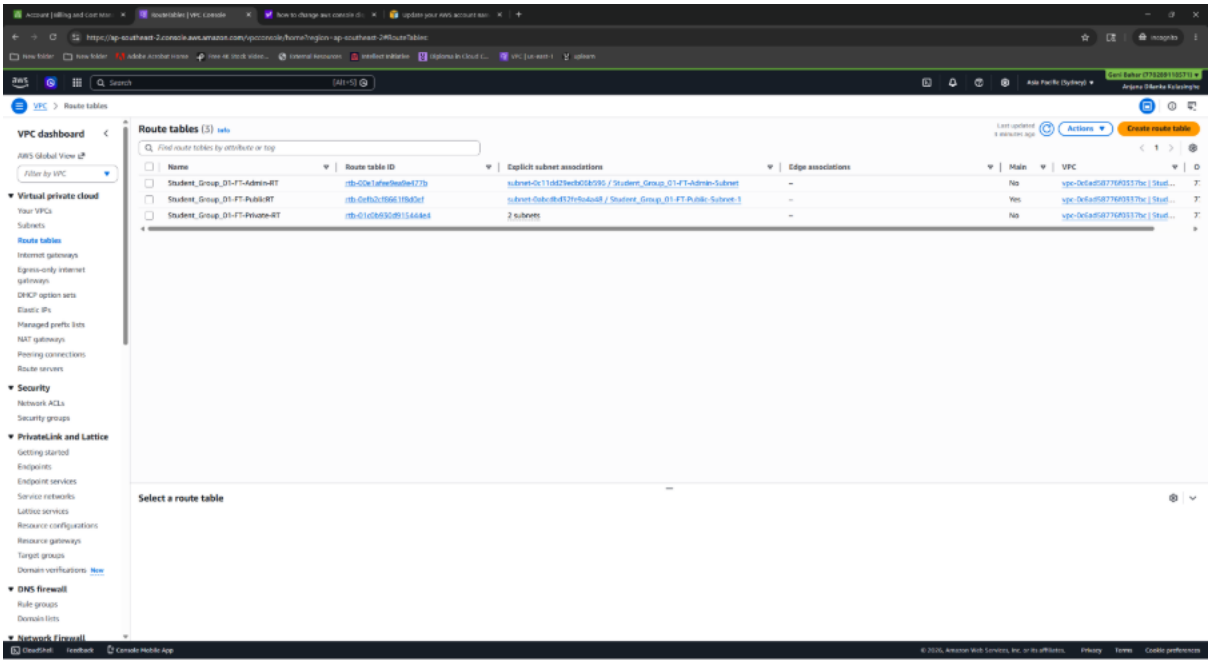
Subnets

The subnets were planned based on how the services need to be accessed. Public subnets are used for resources that need external access, and private subnets are used for internal components such as database or application-side resources. This separation is important because it reduces unnecessary exposure.



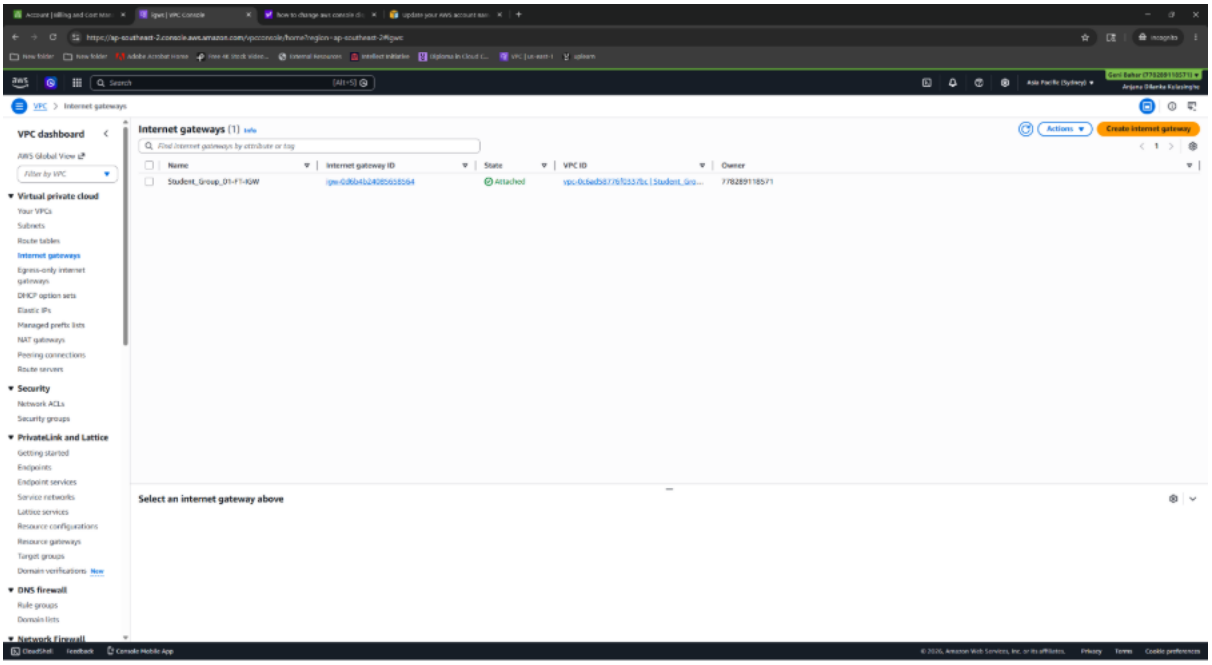
Route Tables

Route tables were configured based on the subnet type. Public subnet traffic uses the Internet Gateway, while private subnet traffic uses the NAT Gateway for outbound access. This allows private resources to download updates without making them directly public.



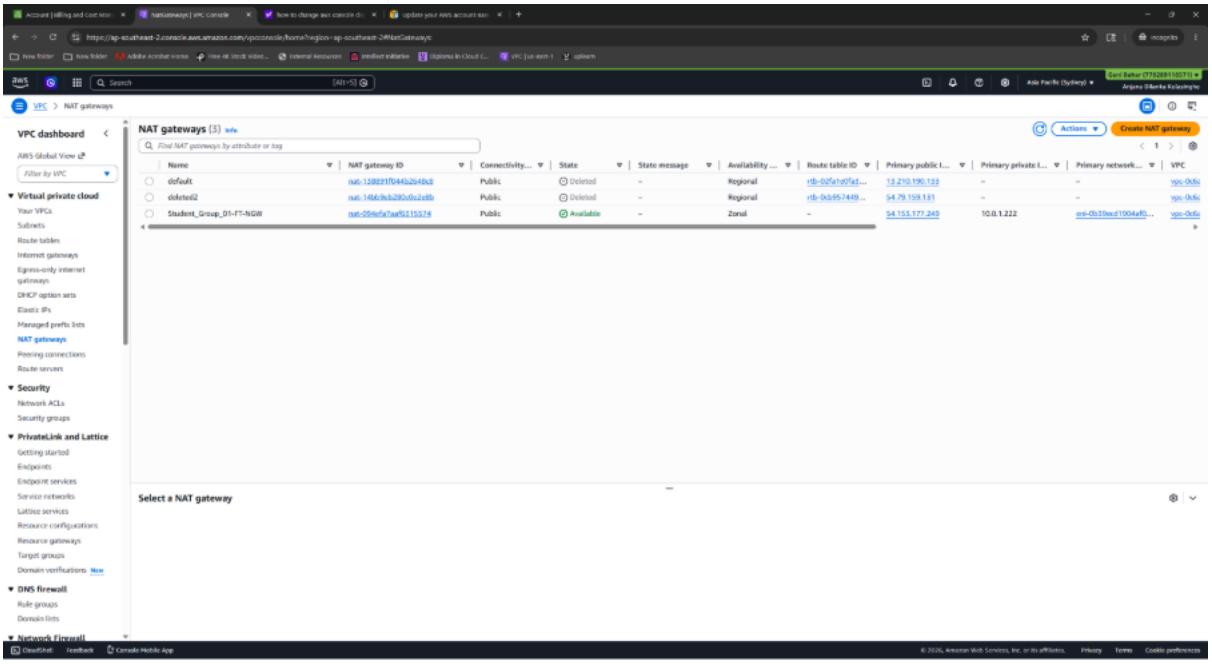
Internet Gateway

The Internet Gateway was attached to the VPC for public subnet connectivity. However, it only works for resources that also have the correct route table and Security Group configuration.



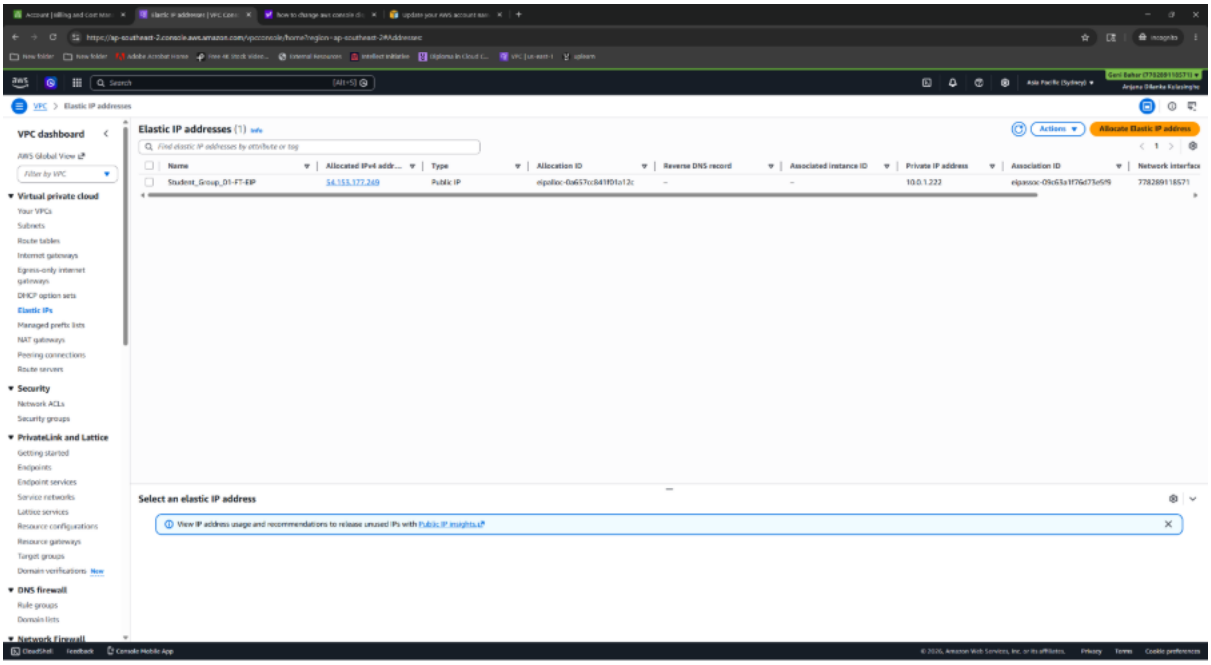
NAT Gateway

The NAT Gateway was used for private subnet outbound traffic. This was needed for updates and package installation, while still avoiding direct inbound access from the internet.



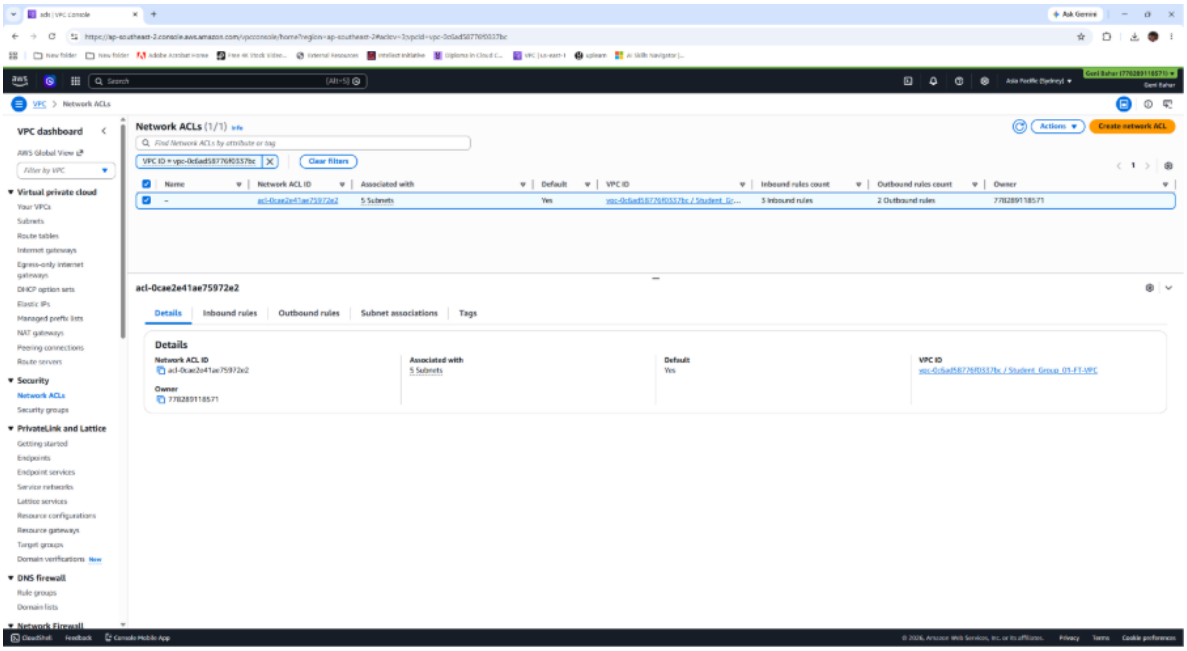
Elastic IP

We allocated an Elastic IP for the NAT Gateway so outbound traffic from private resources had a stable public address.



Network Access Control Lists (NACLs)

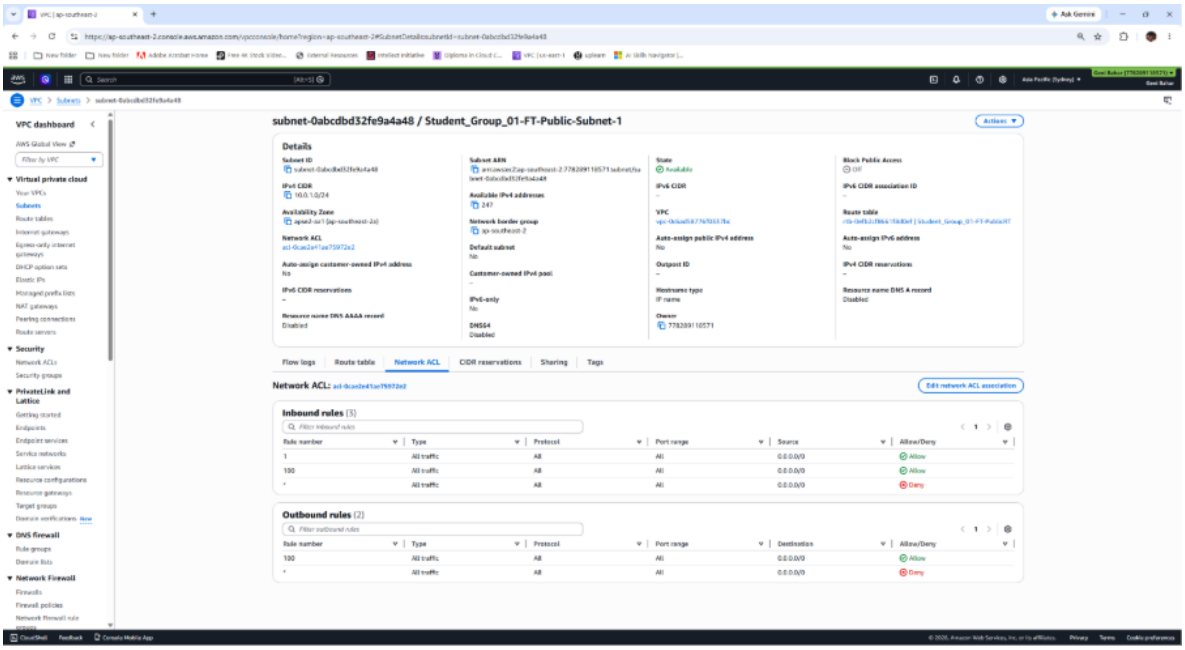
We reviewed the Network ACL configuration as part of the subnet-level security check. In this setup, the default VPC NACL was kept, and Security Groups were used as the main access control method because they are easier to apply directly to EC2, RDS, Load Balancer and VPN resources.



The project subnets use the default VPC Network ACL. This was acceptable for the assessment because Security Groups were used for the more specific access rules.

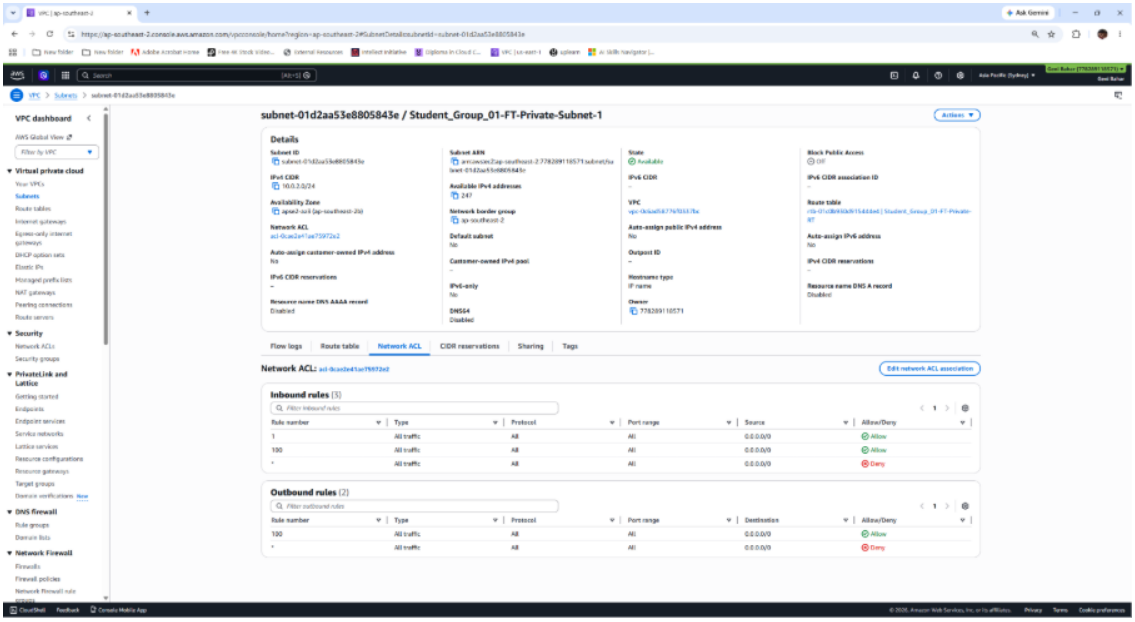
Public Subnet NACL Association

The public subnet NACL association was checked to make sure it was linked correctly to the public subnet used in the design.



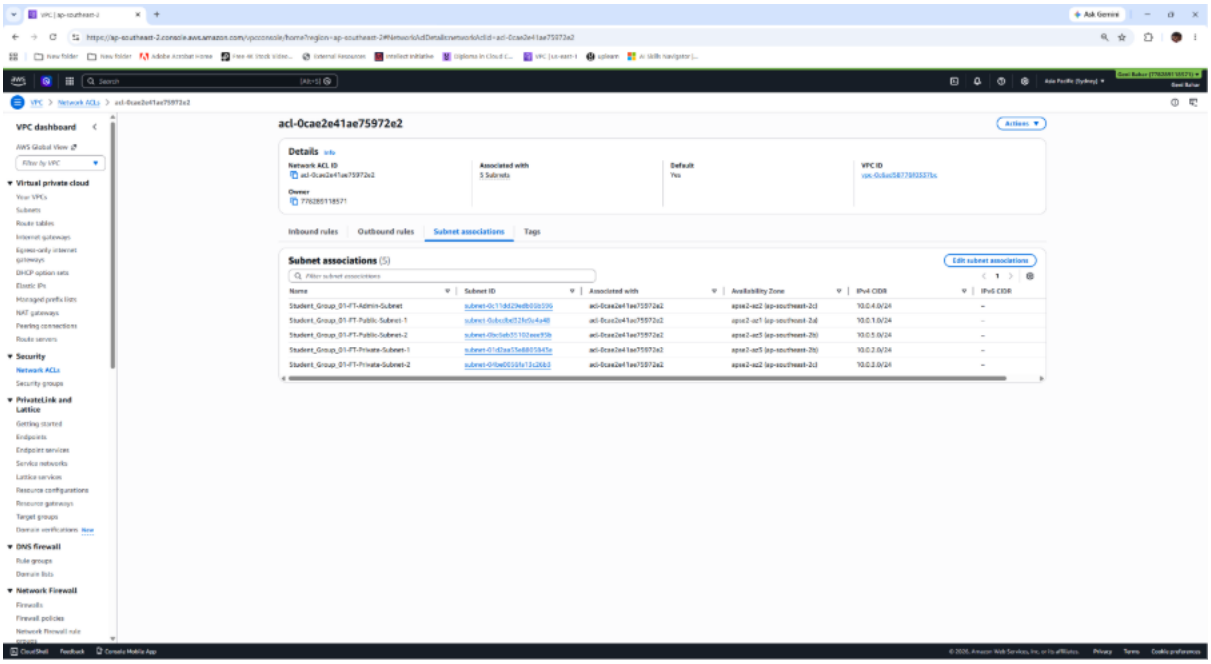
Private Subnet NACL Association

The private subnet NACL association was also checked as part of the review. Private resources were still mainly protected using route tables and Security Groups.



Associated Subnets

The screenshot confirms the NACL association with the project subnets.

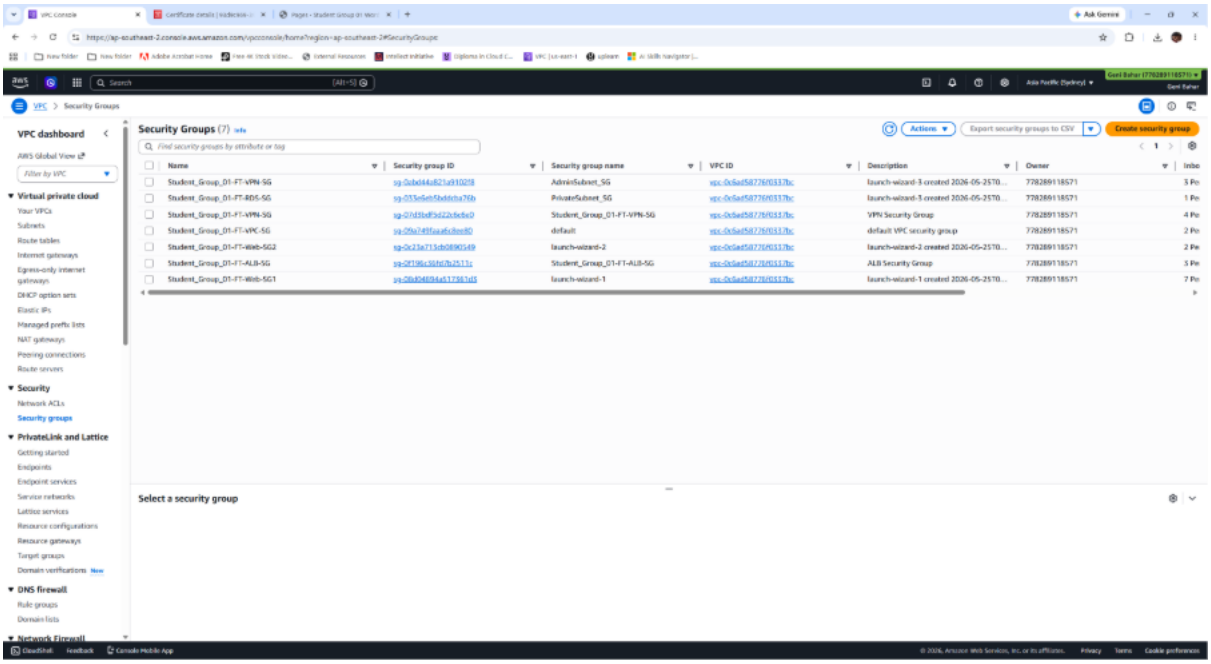


Security Groups

Security Groups were one of the most important controls in this project. We used separate Security Groups for the Load Balancer, EC2, database and VPN-related resources so each service only allowed the traffic it needed.

The inbound and outbound rules were reviewed carefully because an open rule can quickly become a security issue, especially for SSH or database access.

Security Group	**Main inbound access**	**Security purpose**
Load Balancer SG	HTTP/HTTPS from internet	Allows public web access only through the Load Balancer
EC2/Application SG	Web/app traffic from Load Balancer, admin/VPN access only where required	Prevents direct unrestricted access to application servers
RDS SG	MySQL only from approved EC2/Application SG	Prevents public database exposure
VPN/OpenVPN SG	VPN ports from authorised clients	Supports secure private administration
FortiGate SG	VPN/firewall-related traffic only	Controls inter-cloud traffic and security inspection



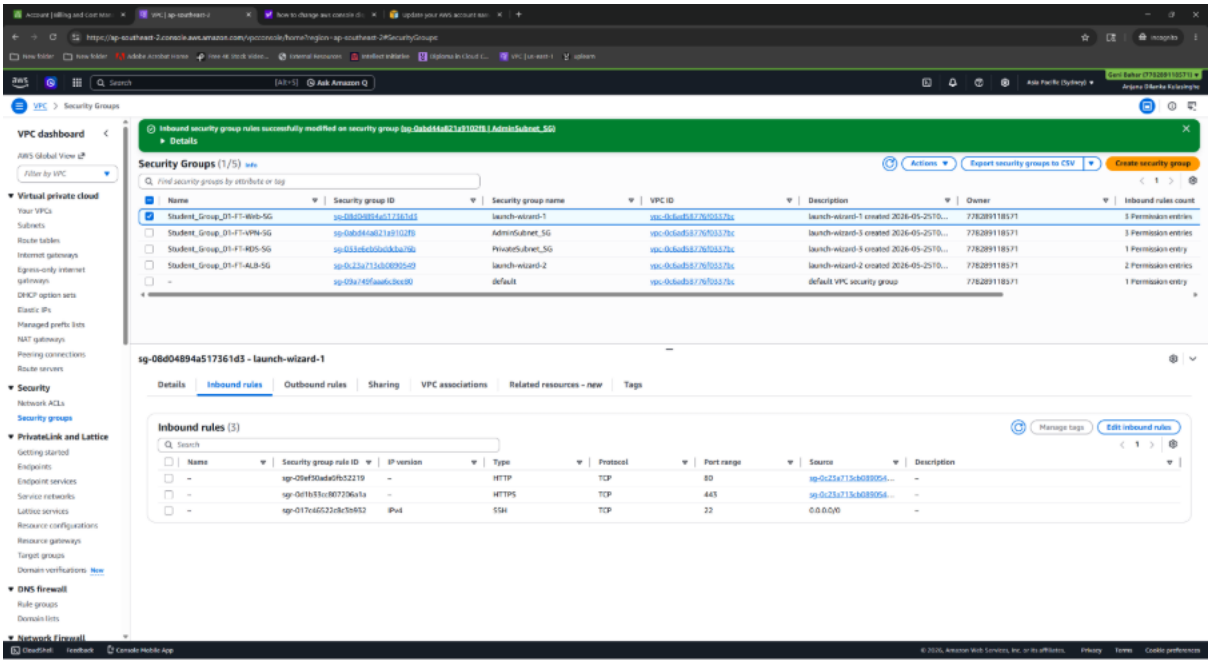
Inbound/Outbound Rules

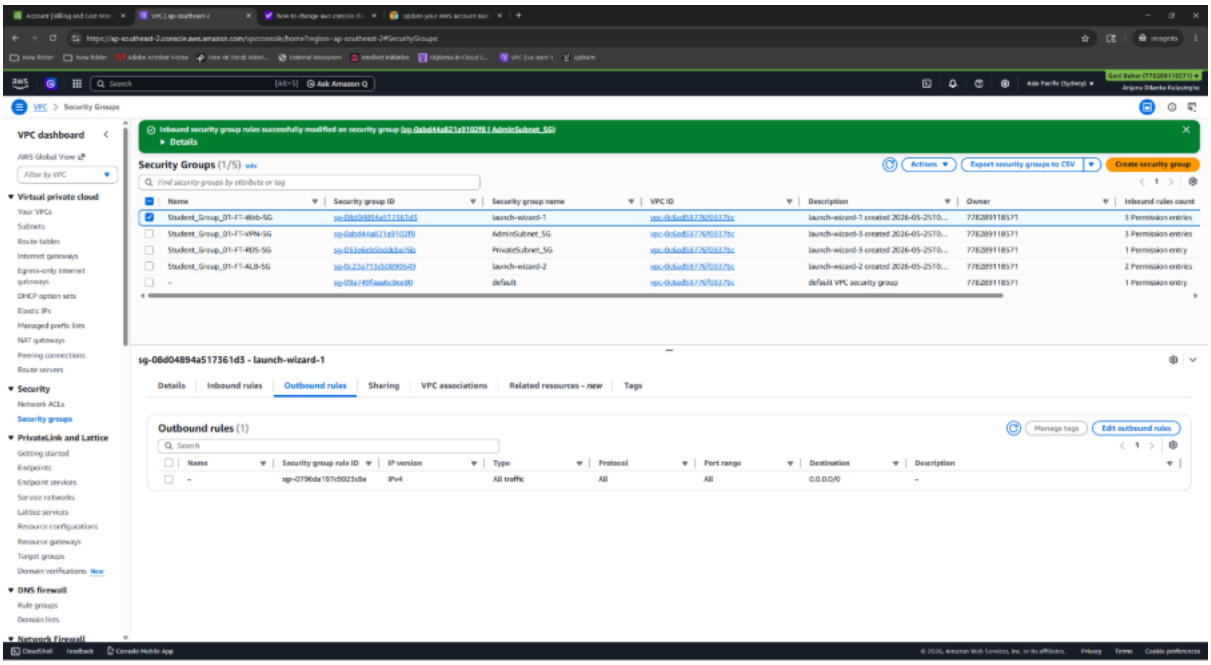
The Application Load Balancer Security Group allows HTTP and HTTPS from the internet because it is the public entry point for the WordPress site.

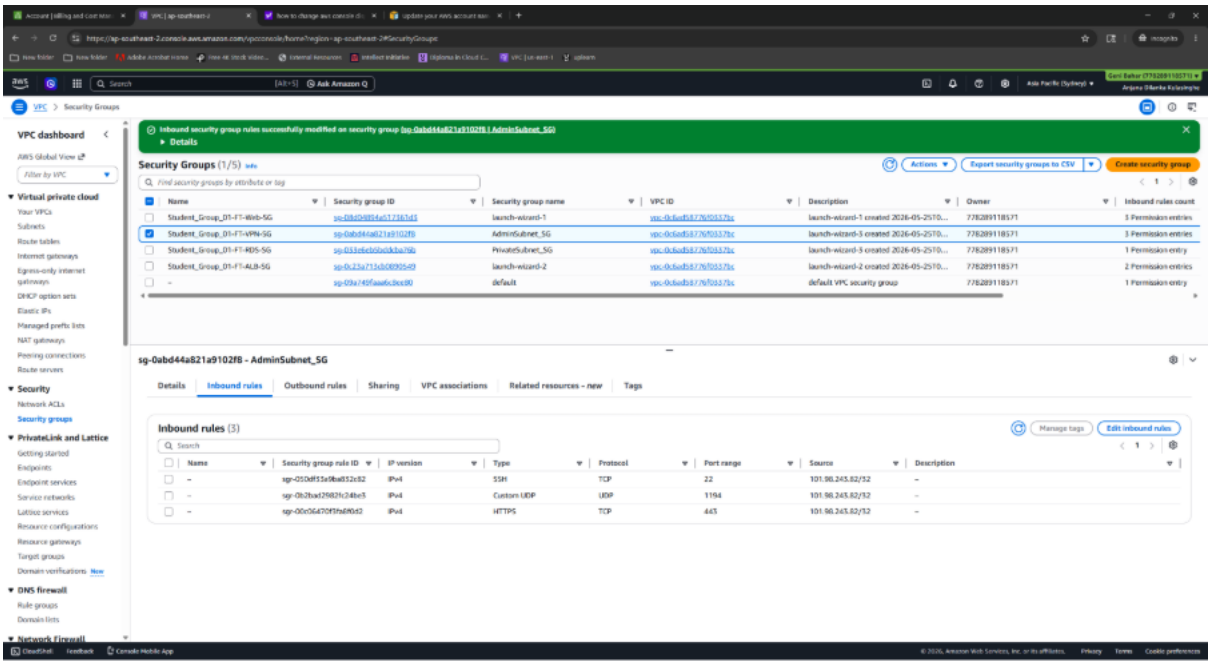
The EC2 Security Group was used to limit direct access to the application instances and keep administration access controlled.

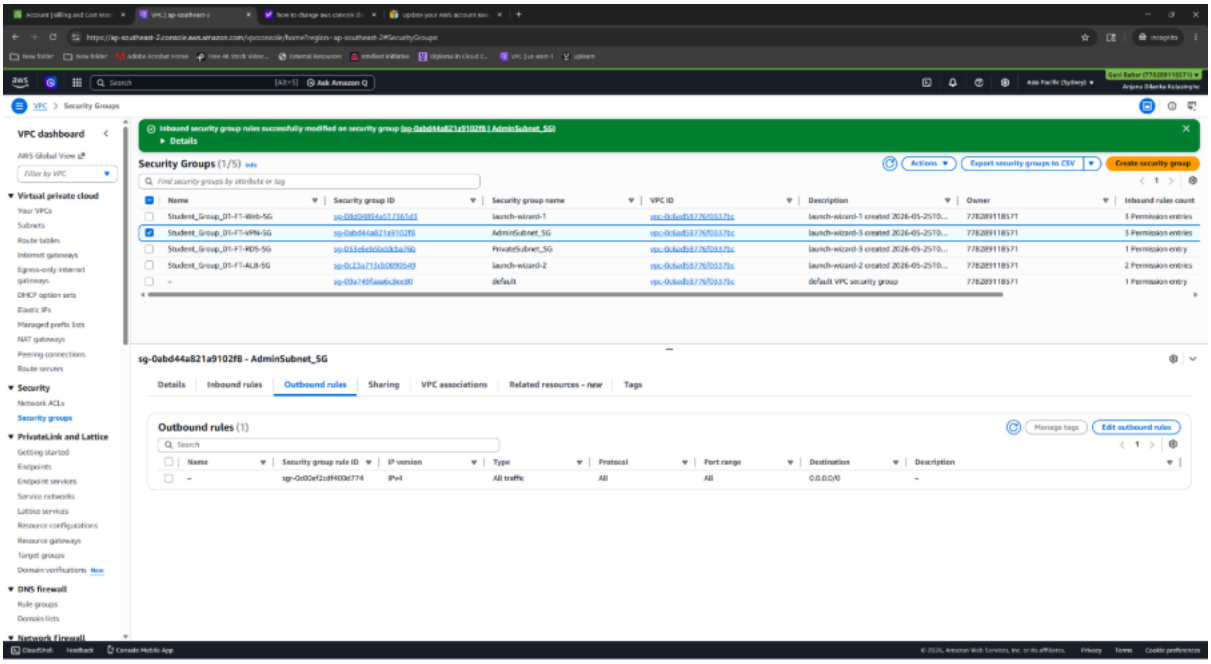
The database Security Group restricts MySQL access so the database is not open directly to the public internet.

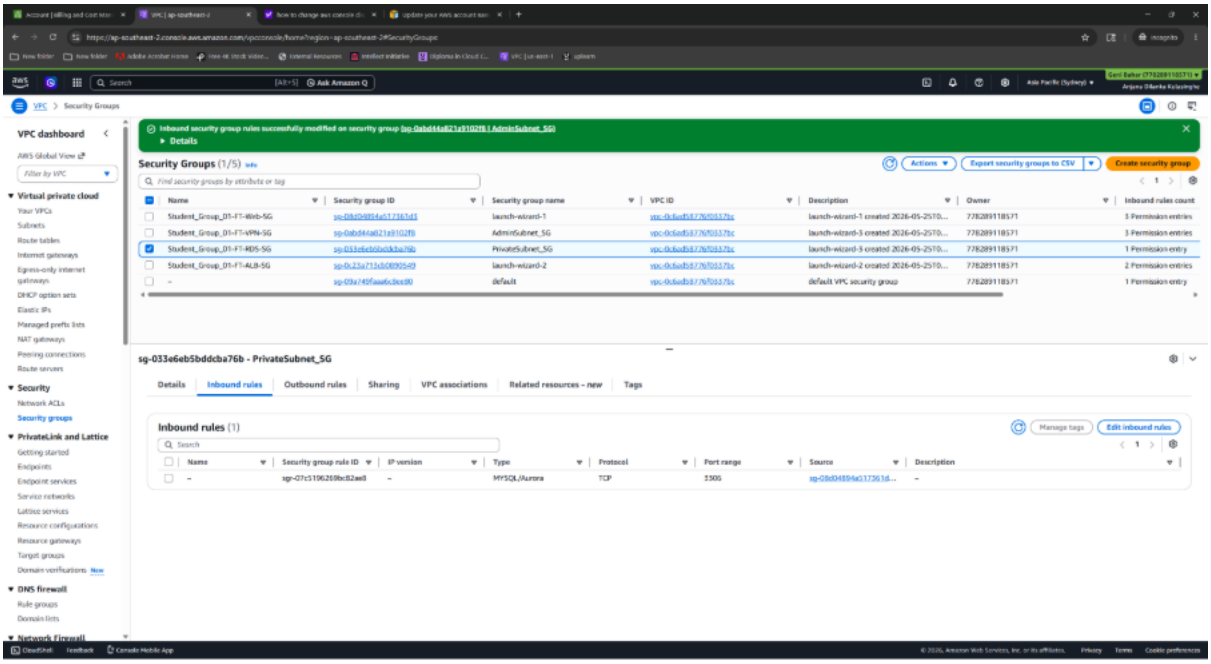
The VPN Security Group supports remote access, but only through the required VPN-related ports.

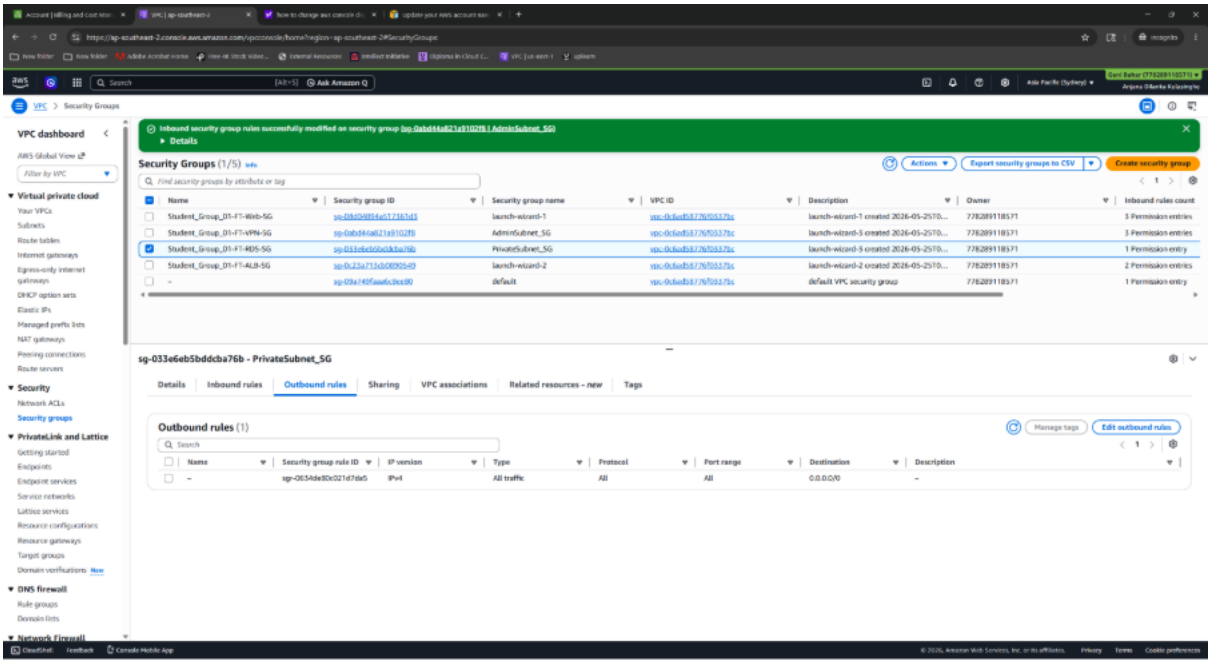


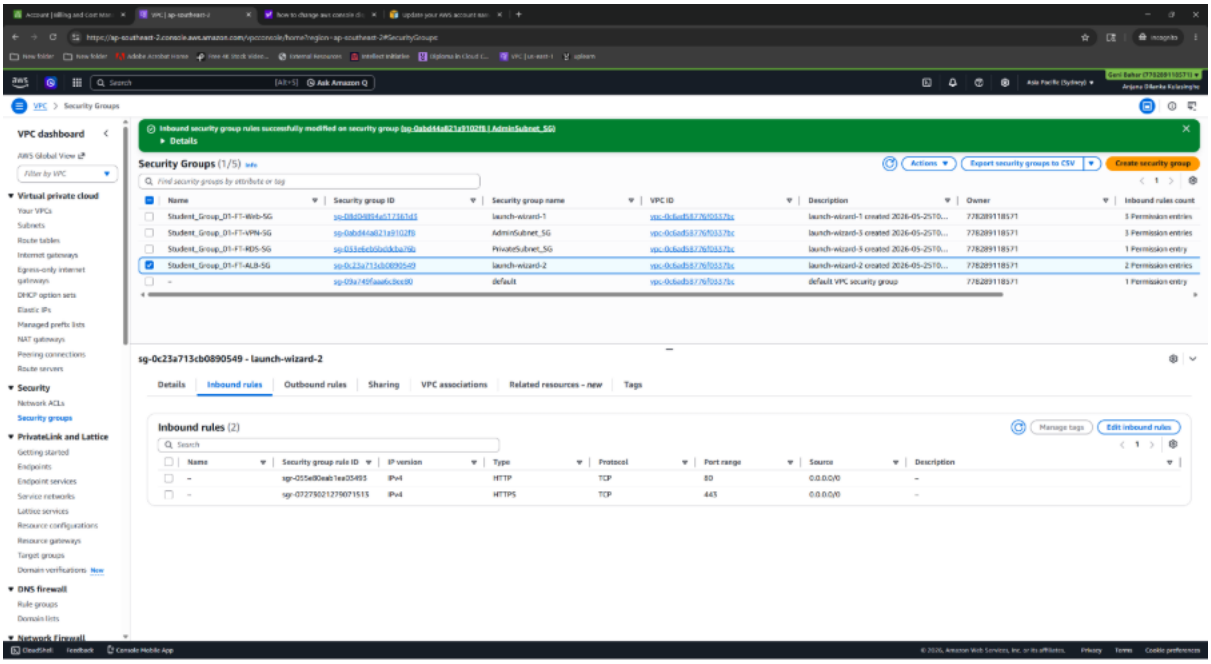

VPC dashboard
Security Groups (1/5)
sg-08d04894a517361d3 - launch-wizard-1
Details
Inbound rules
Outbound rules
Sharing
VPC associations
Related resources - new
Tags
Outbound rules (1)

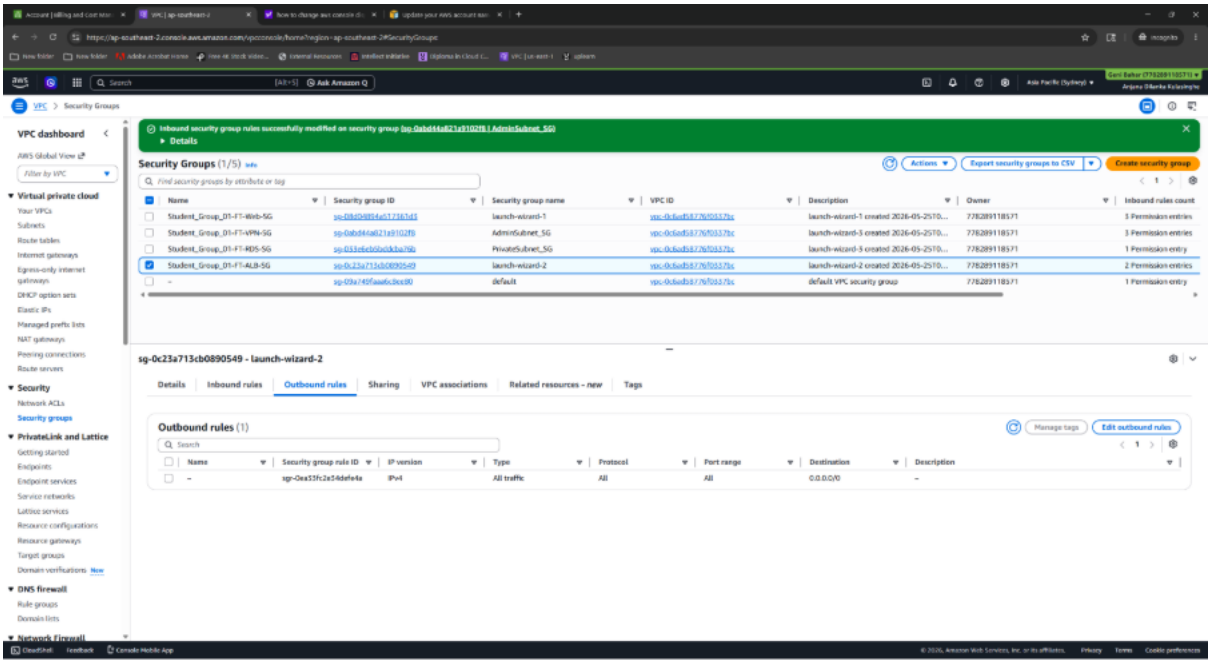

VPC dashboard
Security Groups (1/5)
sg-0abd44a821a9102ff8 - AdminSubnet_SG
Details
Inbound rules
Outbound rules
Sharing
VPC associations
Related resources - new
Tags
Inbound rules (3)


sg-0abd44a821a9102f8 - AdminSubnet_SG
Details
Inbound rules
Outbound rules
Sharing
VPC associations
Related resources - new
Tags
Outbound rules (1)


sg-033e6eb5bddcba76b - PrivateSubnet_SG
Details
Inbound rules
Outbound rules
Sharing
VPC associations
Related resources - new
Tags
Inbound rules (1)

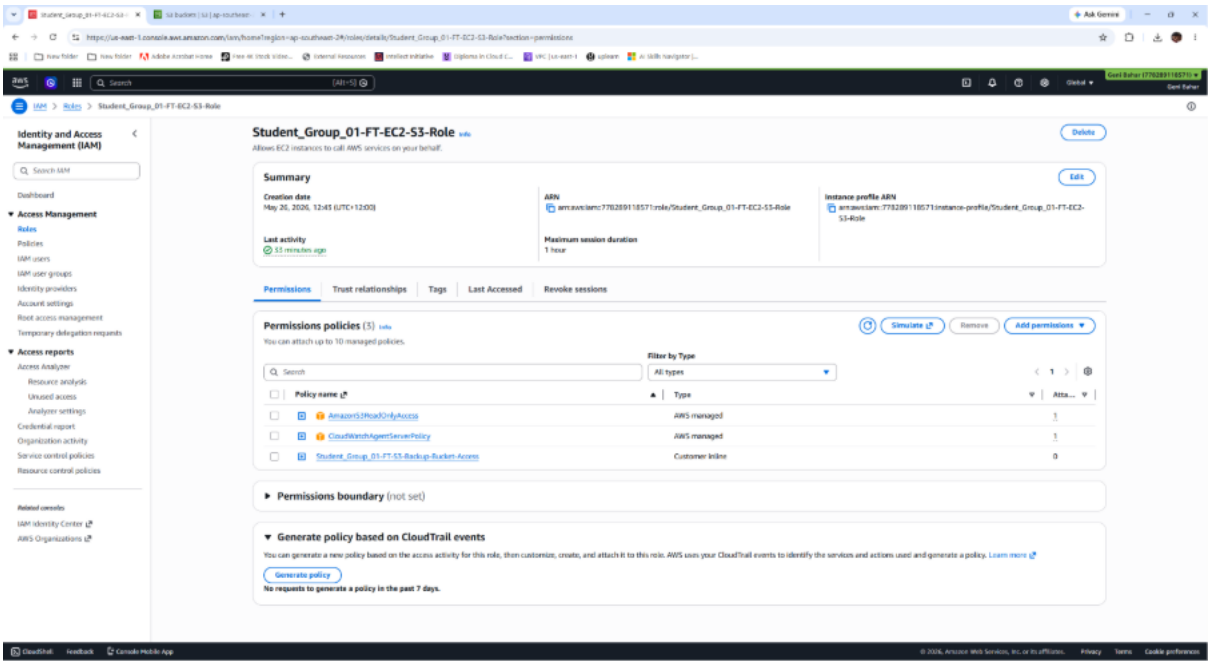

sg-033e6eb5bddcba76b - PrivateSubnet_SG
Outbound rules (1)

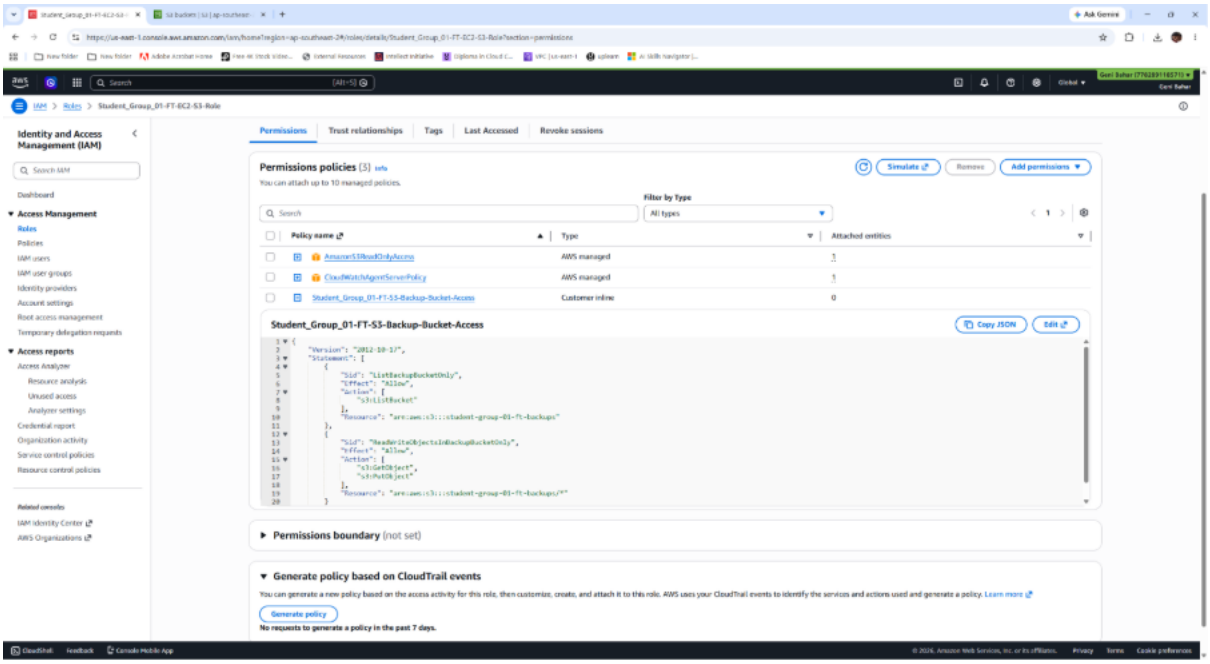

sg-0c23a713cb0890549 - launch-wizard-2
Inbound rules (2)

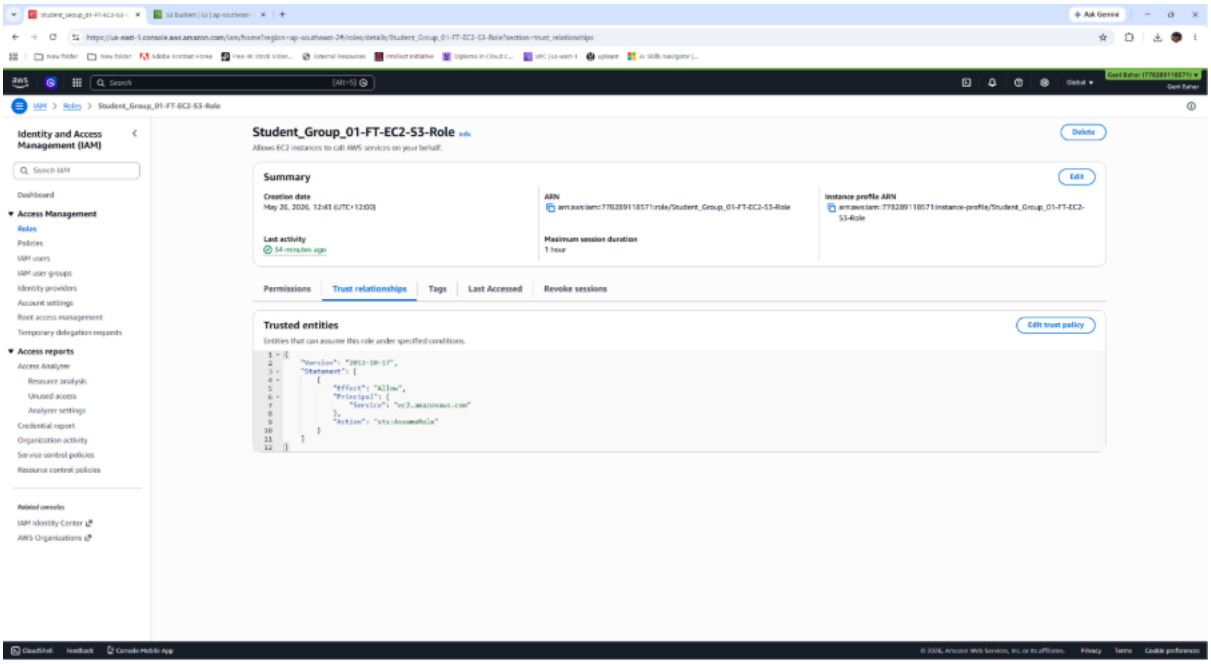

VPC dashboard
Security Groups (1/5)
sg-0c23a713cb0890549 - launch-wizard-2
Details
Inbound rules
Outbound rules
Sharing
VPC associations
Related resources - new
Tags
Outbound rules (1)

Identity and Access Management (IAM)

For IAM, we used roles instead of keeping long-term access keys inside instances or scripts. This is a safer approach because services can receive only the permissions they need. The EC2 role was used for controlled S3 access, and the Lambda role was used for snapshot automation.

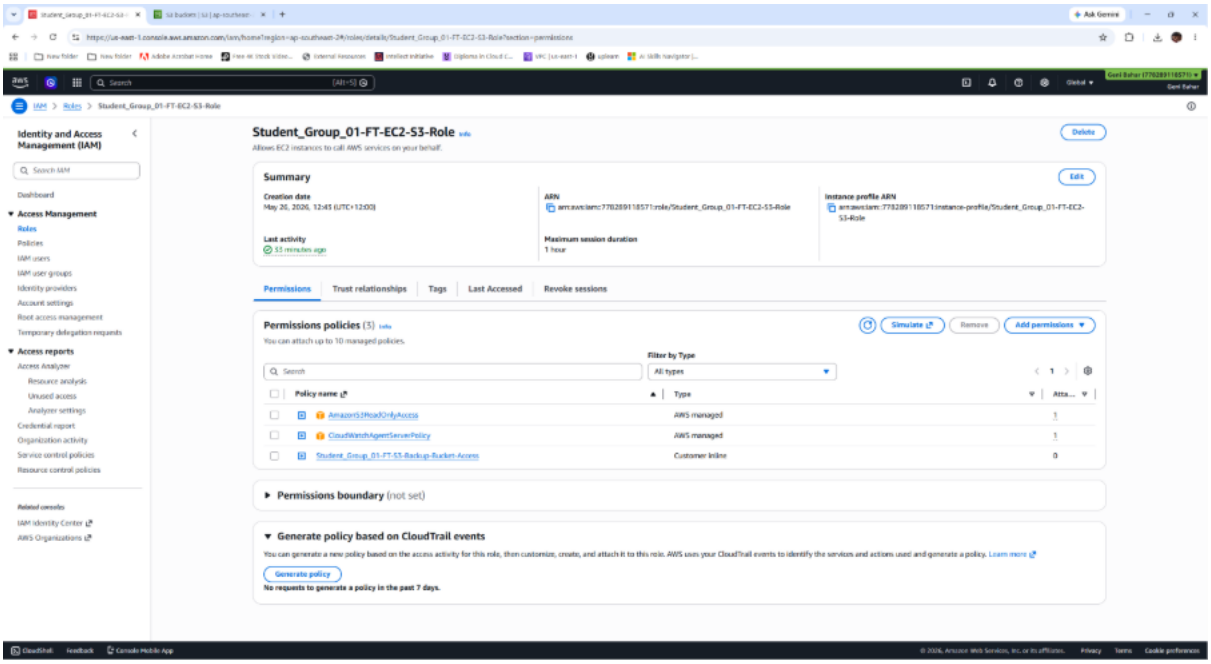


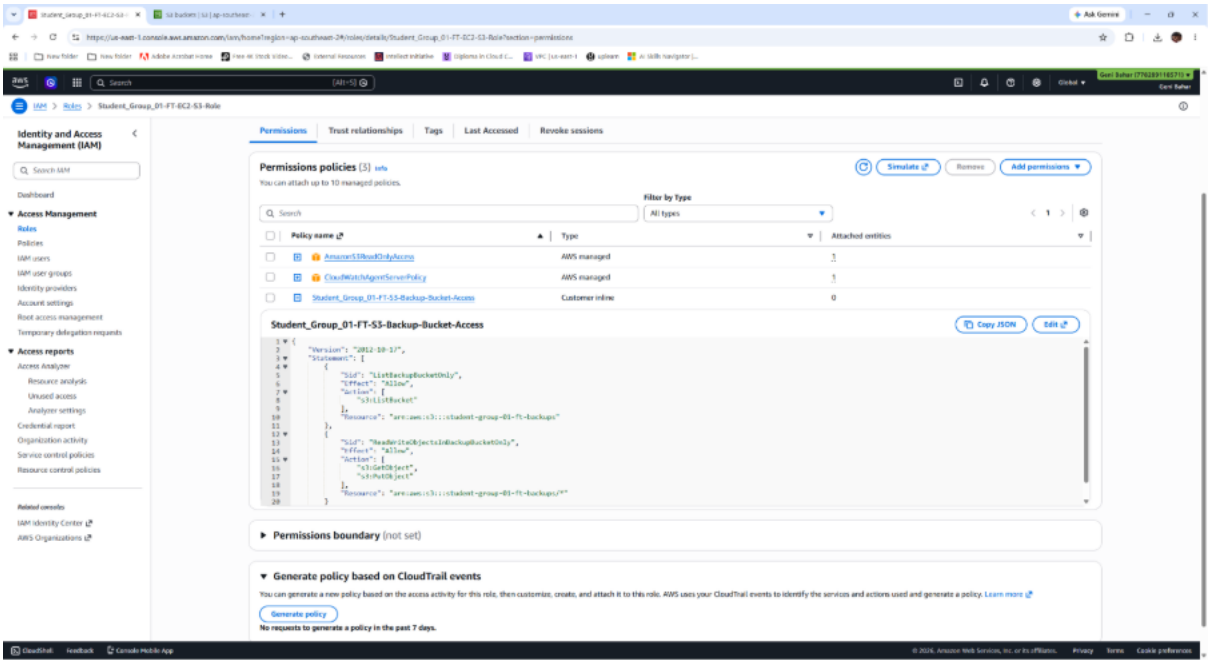


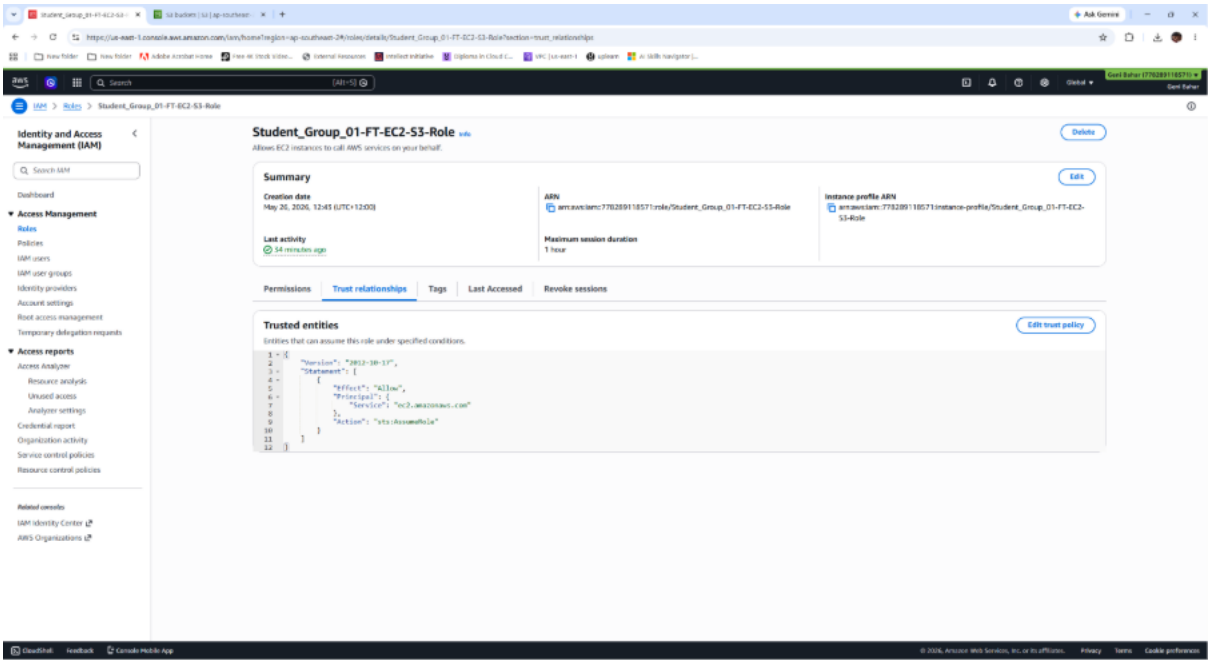


EC2 and S3

The EC2 IAM role allows the application server to access S3 without storing access keys directly on the instance.

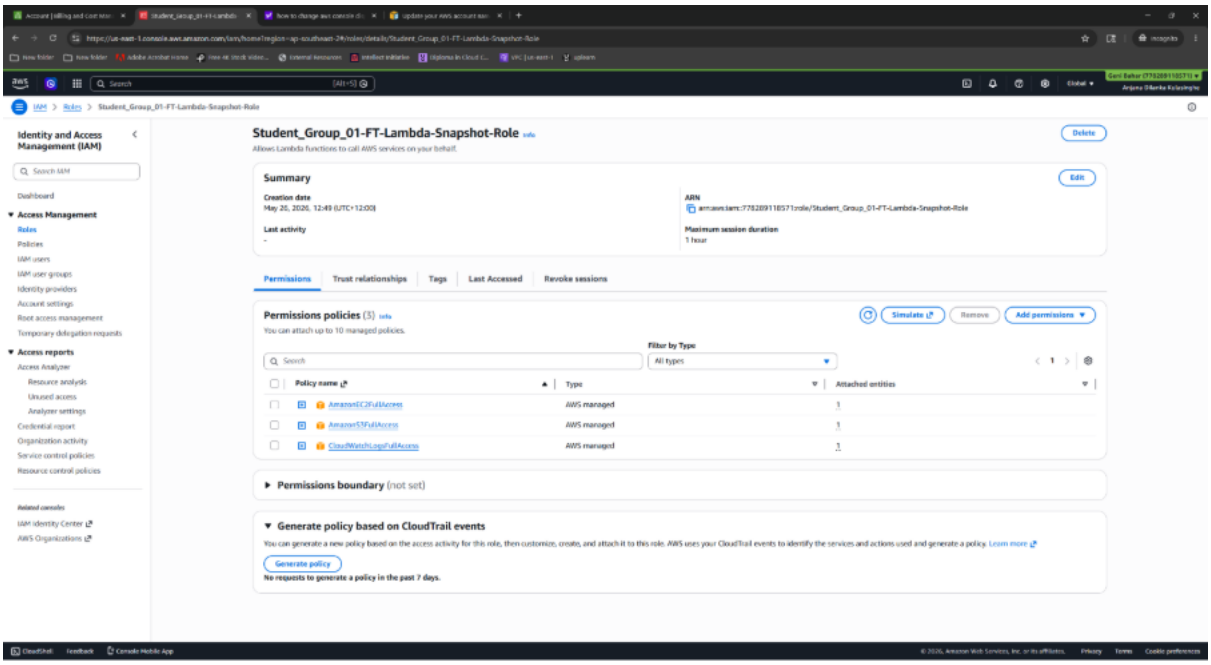


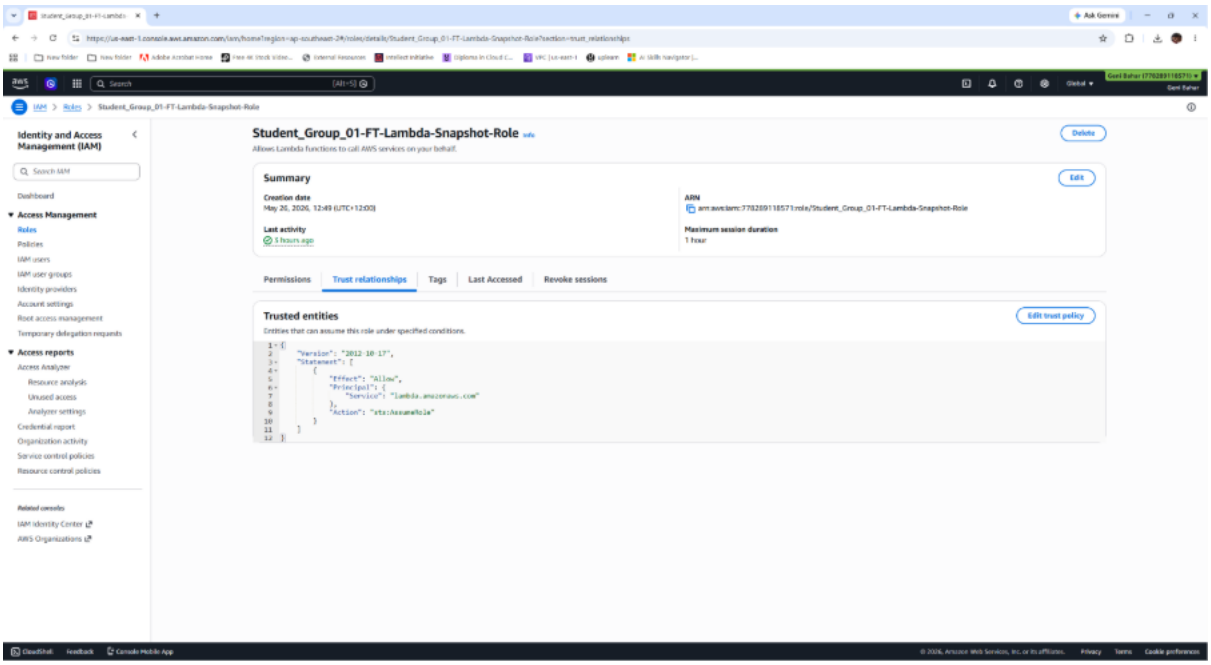

Identity and Access Management (IAM)
Permissions
Trust relationships
Tags
Last Accessed
Revoke sessions
Permissions policies (3)
Student_Group_01-FT-S3-Backup-Bucket-Access
Permissions boundary (not set)
Generate policy based on CloudTrail events


Identity and Access Management (IAM)
Student_Group_01-FT-EC2-S3-Role
Summary
Permissions
Trust relationships
Tags
Last Accessed
Revoke sessions
Trusted entities

Lambda to access EC2

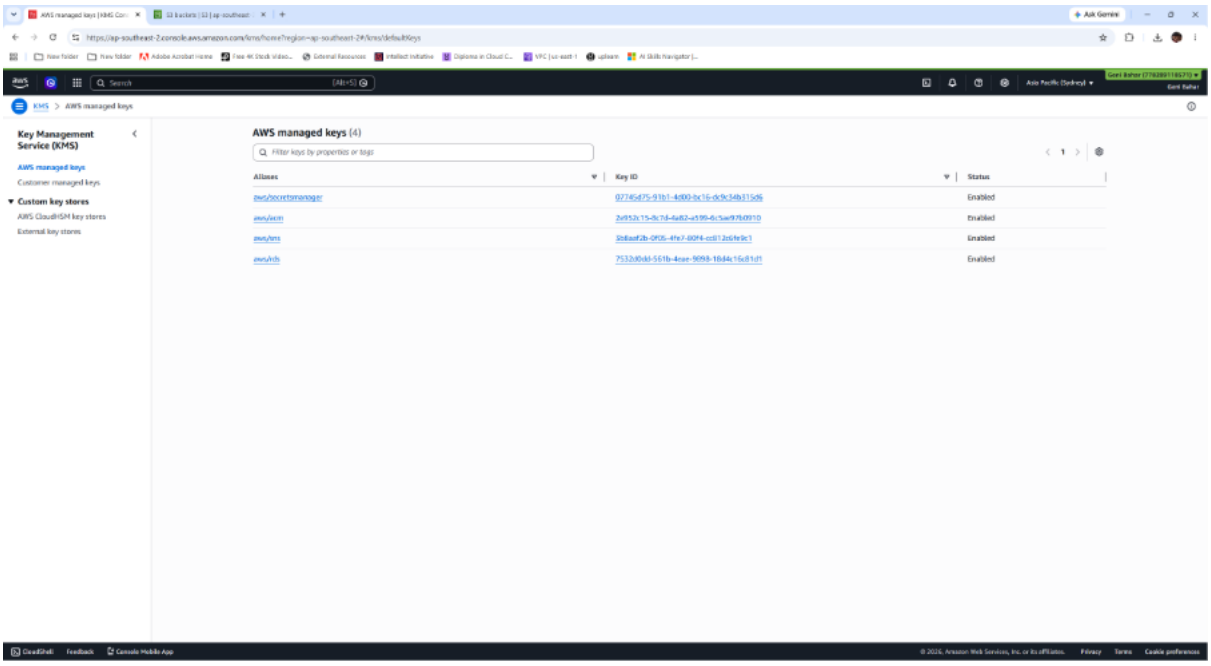
The Lambda execution role provides the permissions required for automated EBS snapshot creation.

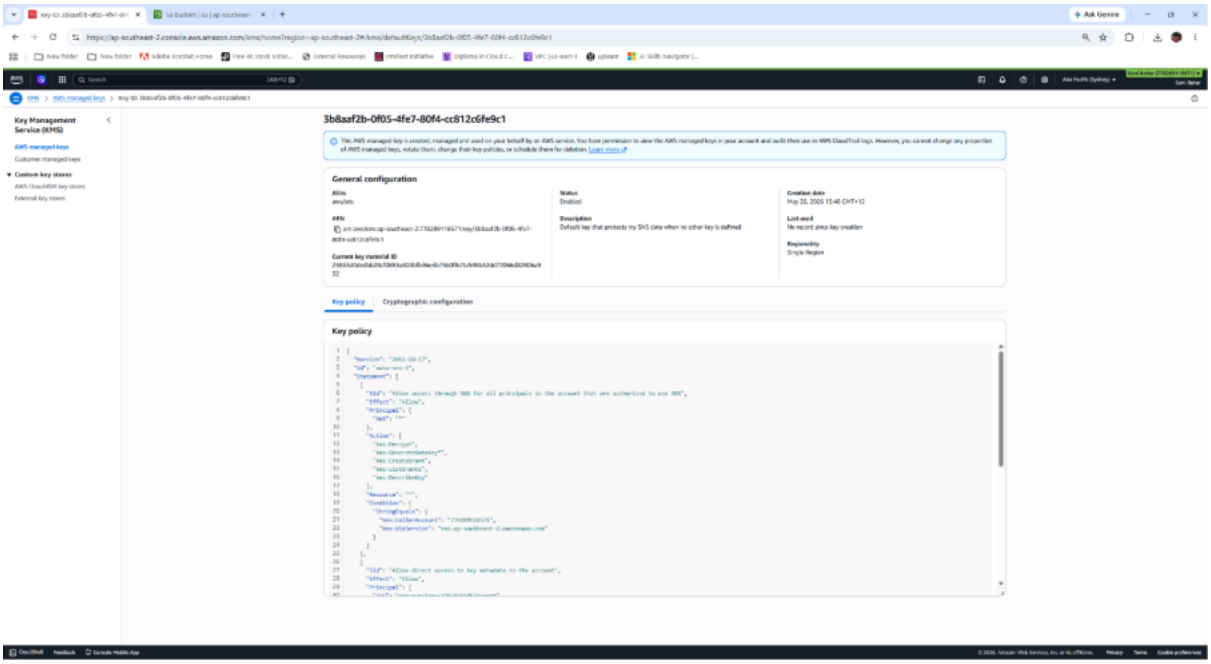


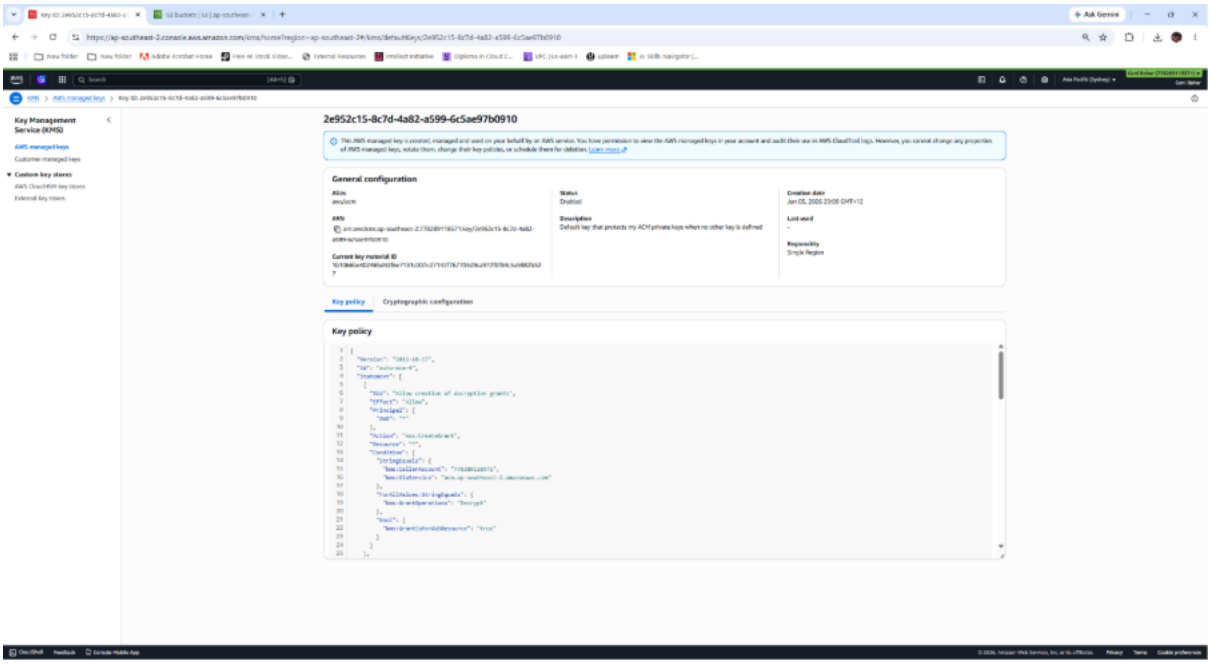


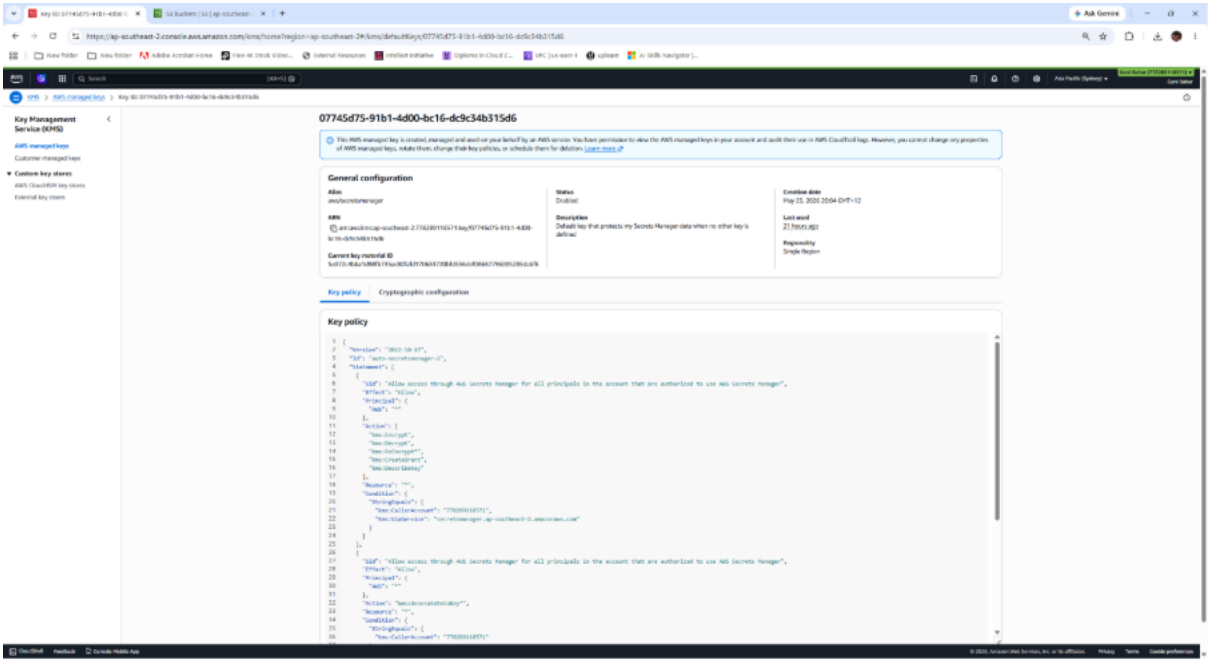
AWS KMS Encryption

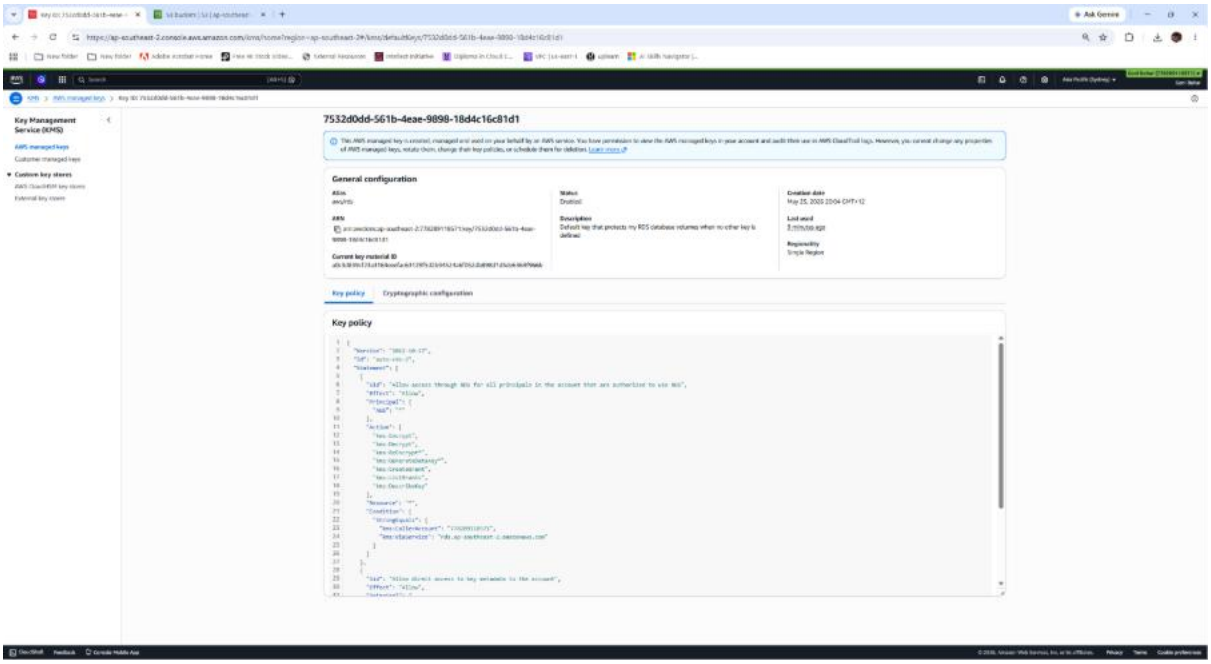
AWS KMS was reviewed and used for encryption-related protection in the AWS environment. For this assessment, we focused on making sure storage and database services were encrypted where possible.





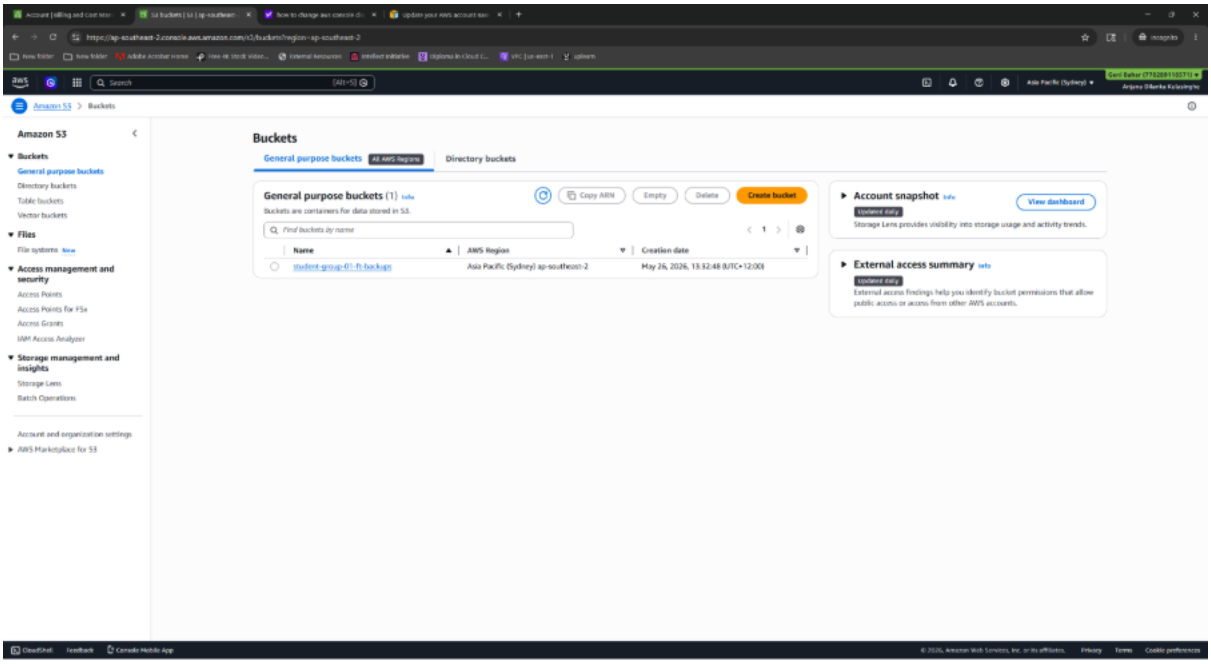

2e952c15-8c7d-4a82-a599-6c5ae97b0910
General configuration
Key policy
Cryptographic configuration


07745d75-91b1-4d00-bc16-dc9c34b315d6
General configuration
Key policy
Cryptographic configuration



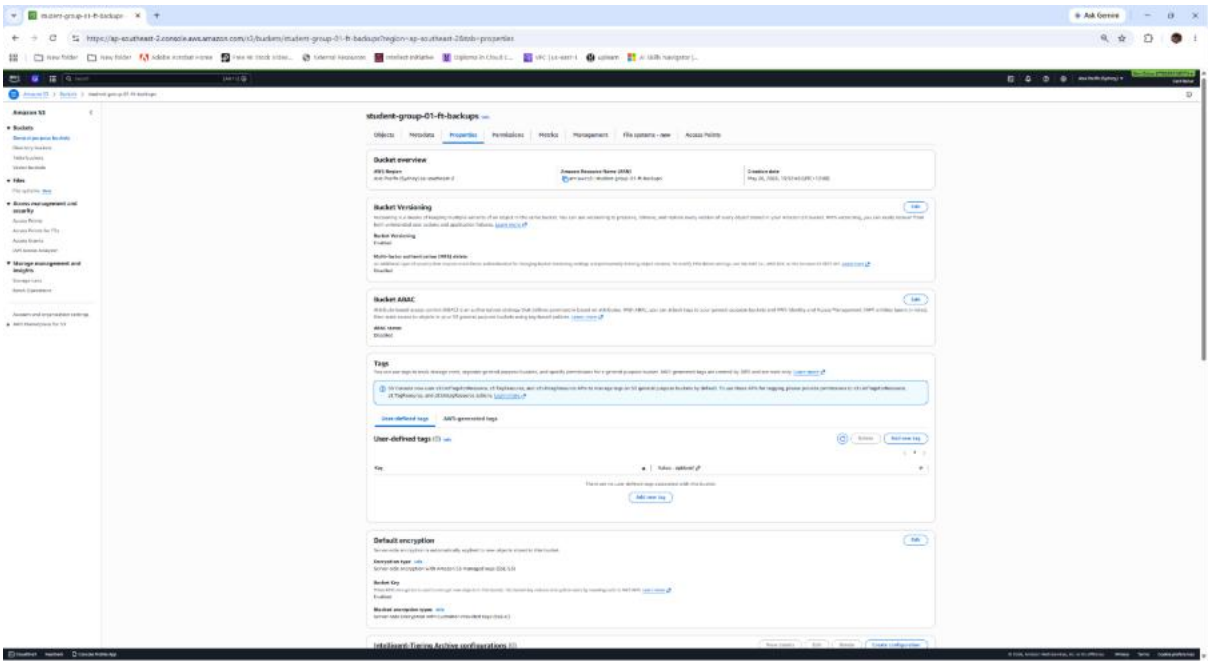
S3

The S3 bucket was used as part of the backup design. Since this bucket is used for backup data, public access is not required and should remain blocked.



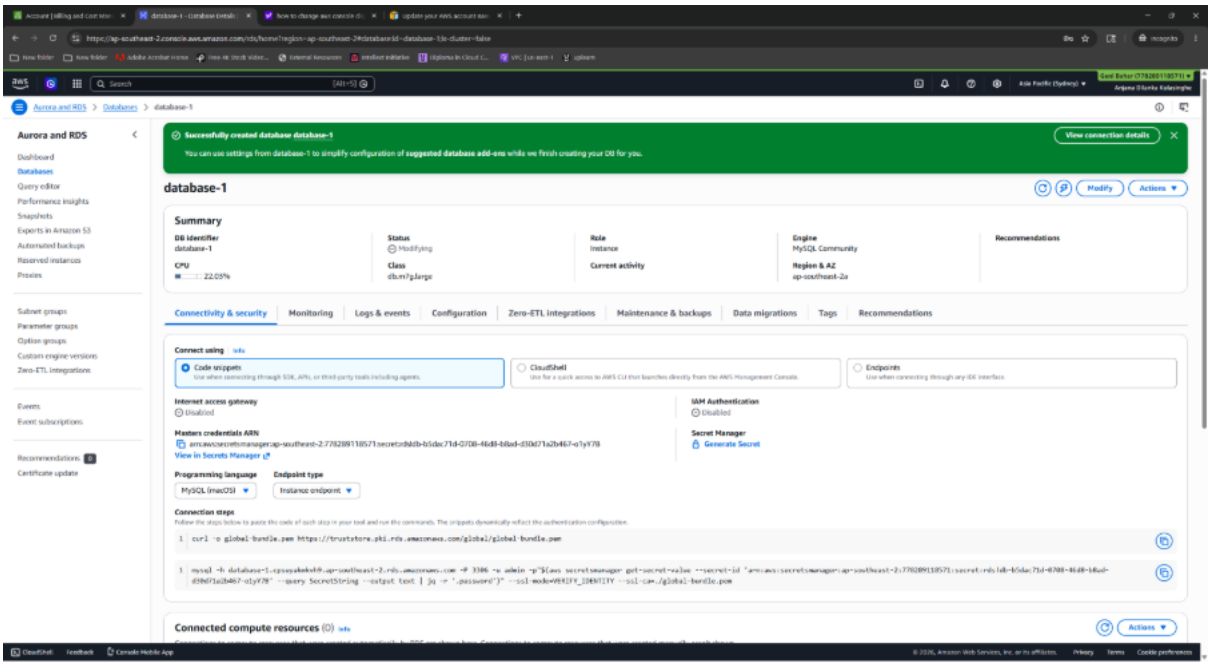
S3 Backup Encryption

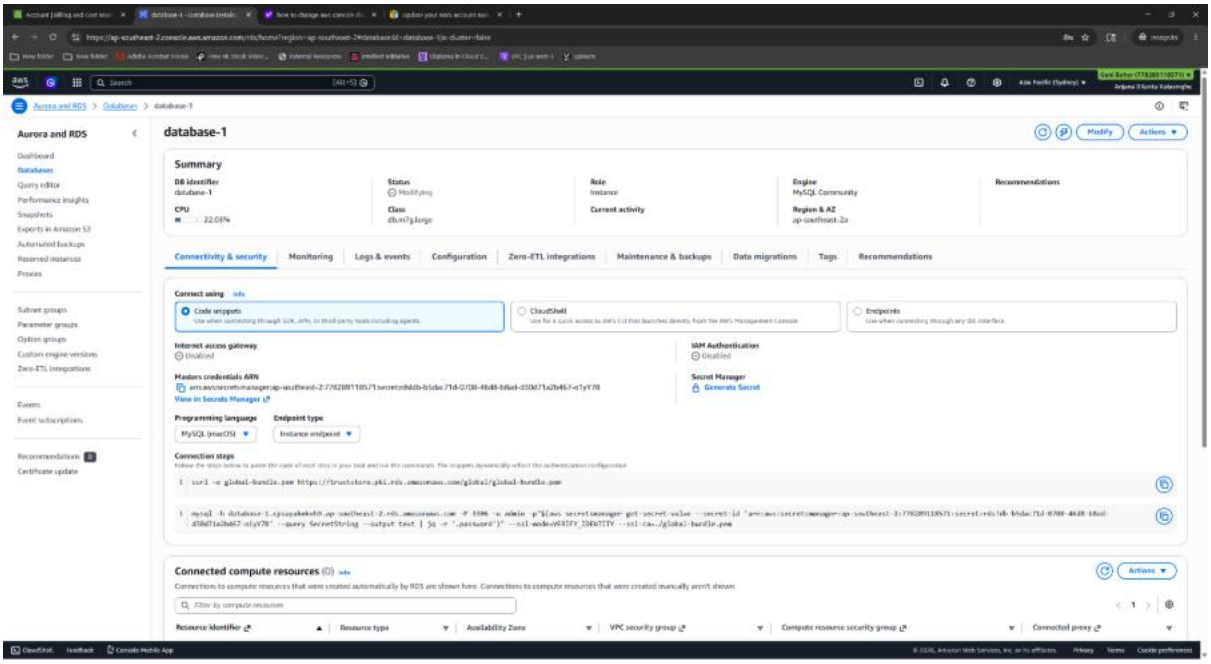
Server-side encryption was enabled for the S3 backup bucket. This helps protect backup data at rest.



RDS

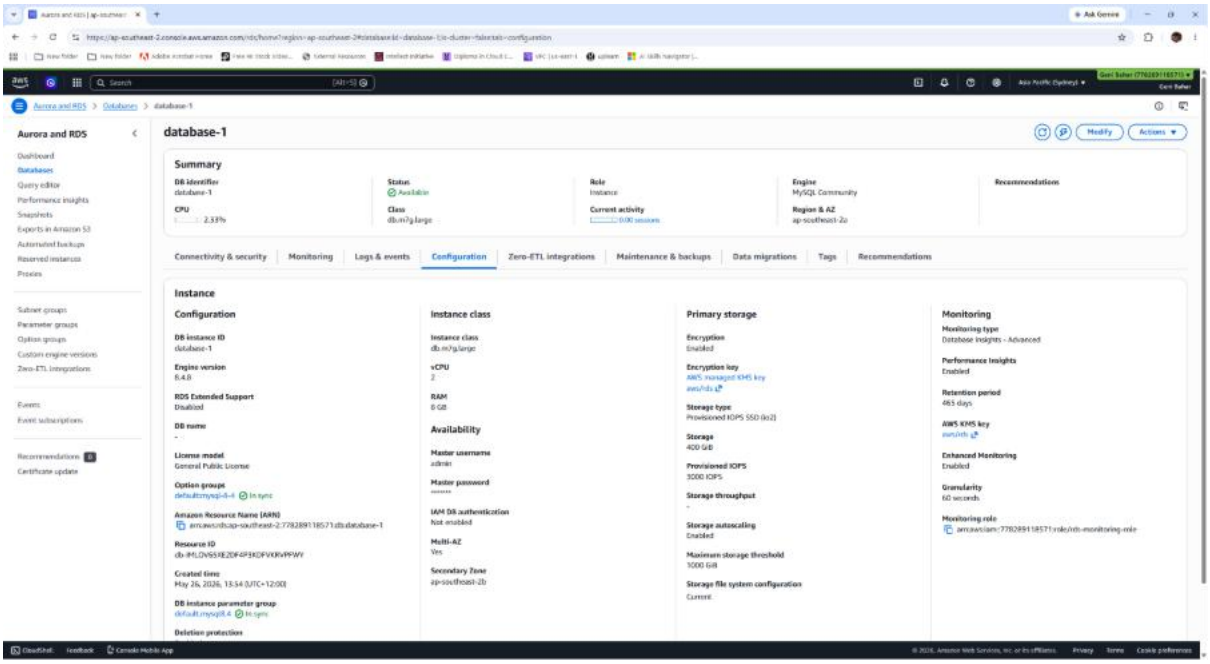
RDS was configured as the database back end for WordPress. The database should stay private and should only accept traffic from the application layer through Security Group rules.





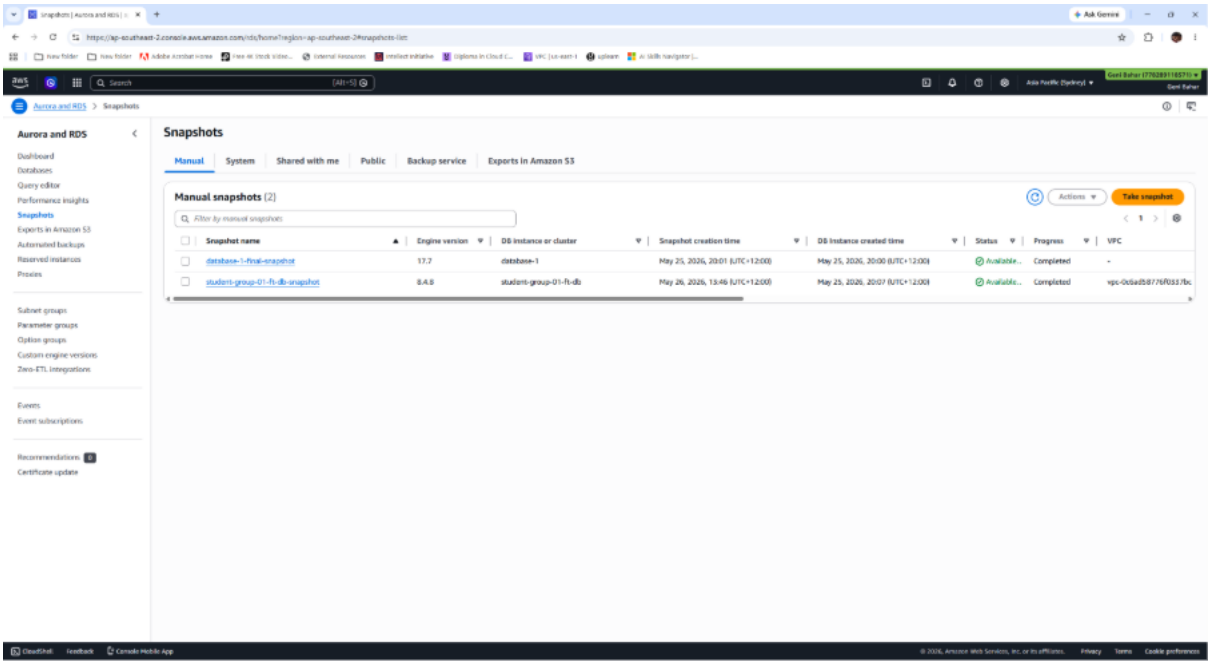
RDS Encryption

Encryption was enabled on the RDS instance using AWS managed keys. This protects the database storage, backups and snapshots.



Snapshots

Snapshots were included so the environment has a recovery point if something is deleted or misconfigured. This was an important part of the backup and recovery plan.



High Availability Architecture

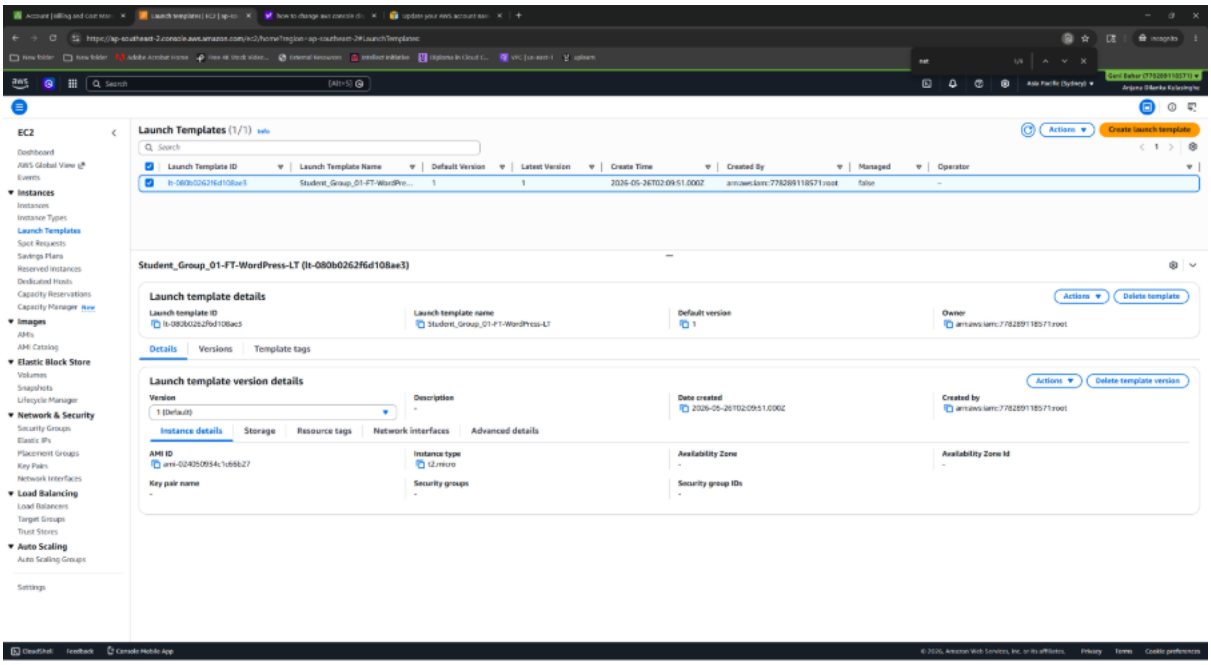
To improve availability, we configured an Application Load Balancer and Auto Scaling Group.

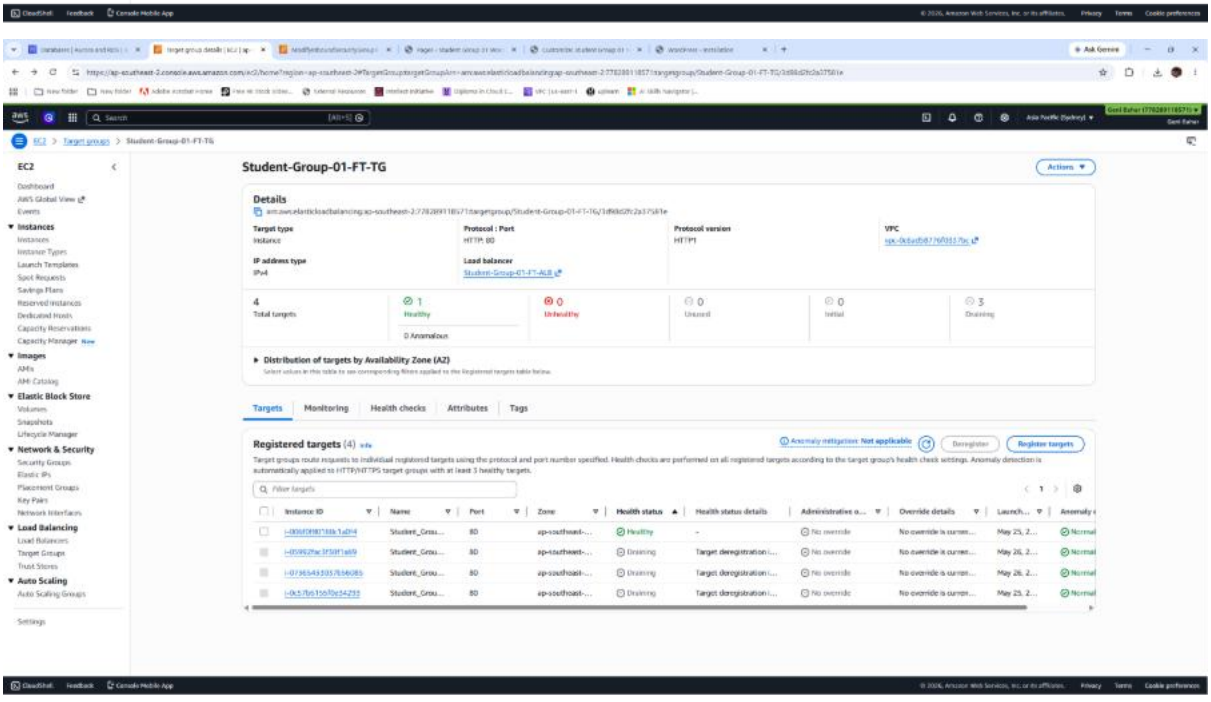
The Load Balancer distributes traffic to healthy instances, and Auto Scaling helps replace unhealthy instances or support extra capacity when needed.

Component	**High availability contribution**
Application Load Balancer	Distributes web traffic and provides a stable public entry point
Auto Scaling Group	Replaces unhealthy instances and adjusts capacity based on demand
Multiple subnets	Supports deployment across separate network segments
RDS	Provides managed database hosting and snapshot capability
DNS/HTTPS	Provides stable and secure public access

Target Group

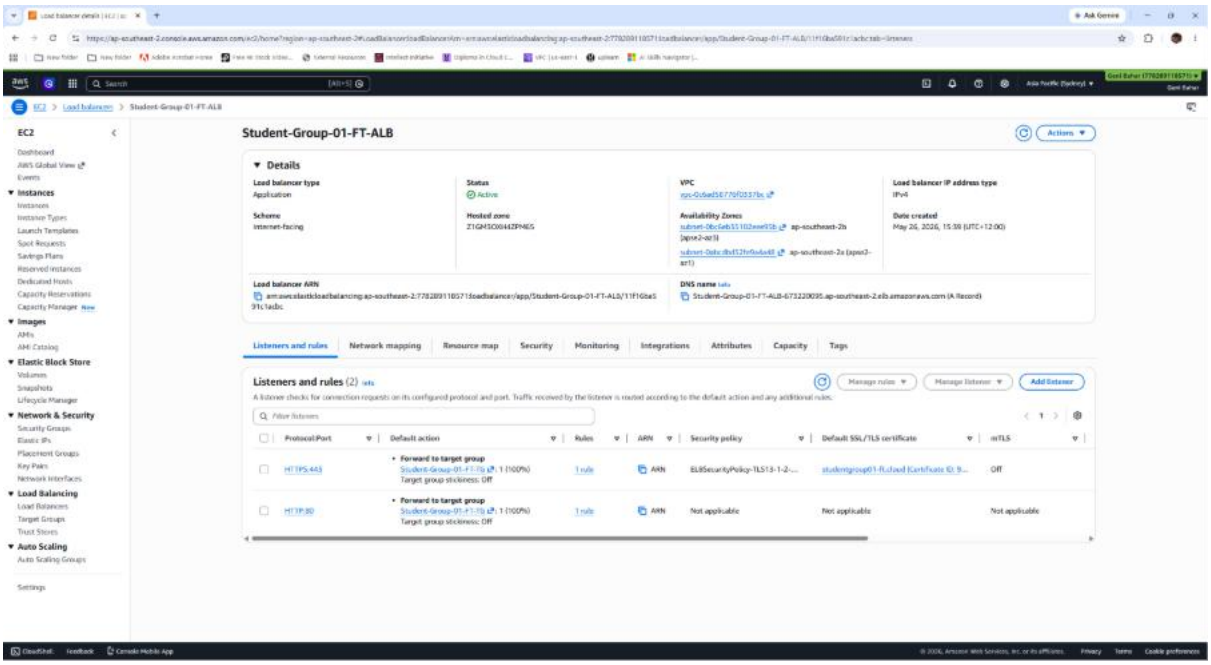
The target group was used to register the WordPress instances and check whether they were healthy before sending traffic.

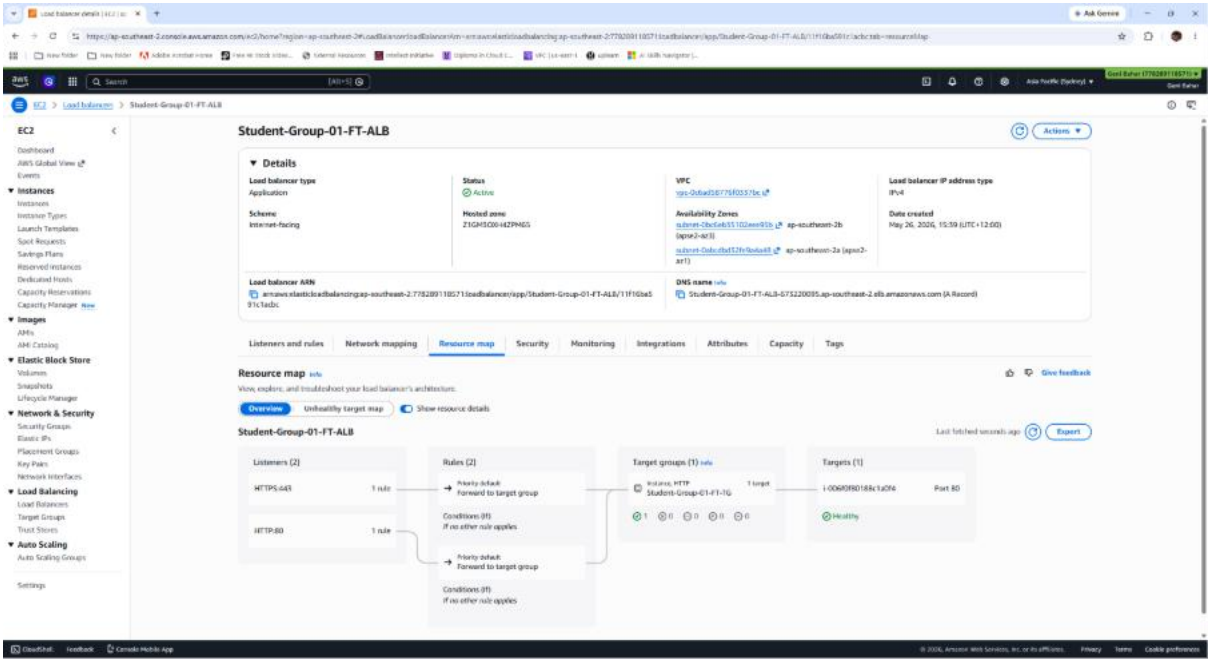




Load Balancer

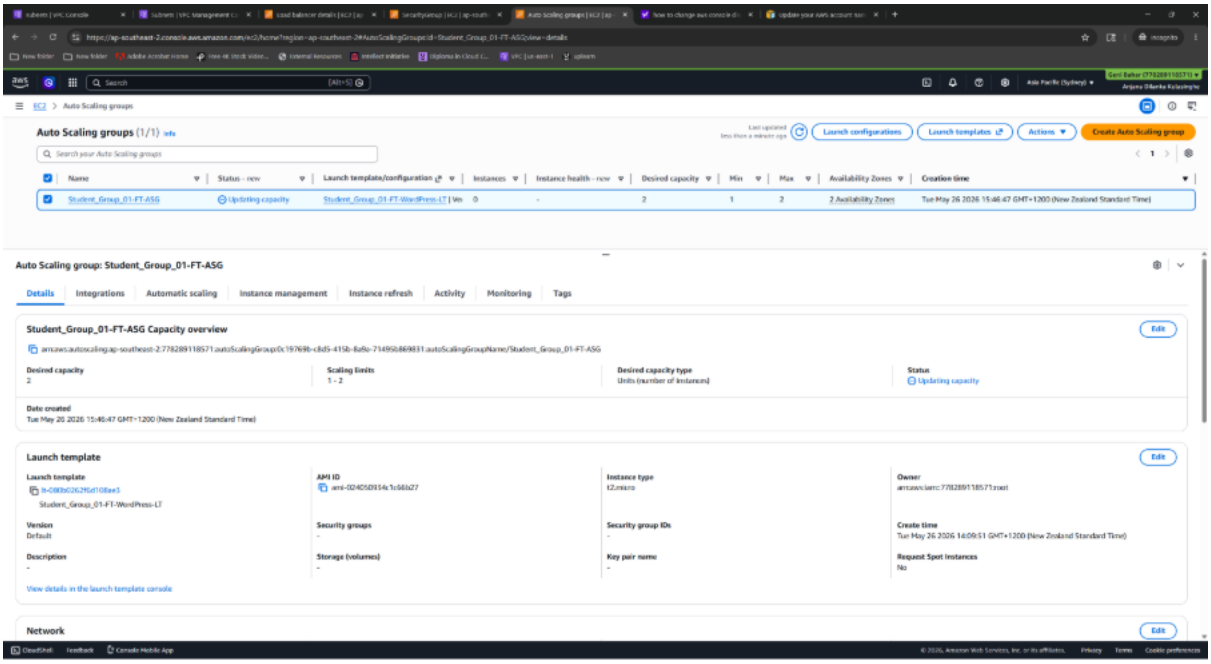
The Application Load Balancer distributes requests across available instances to improve availability.

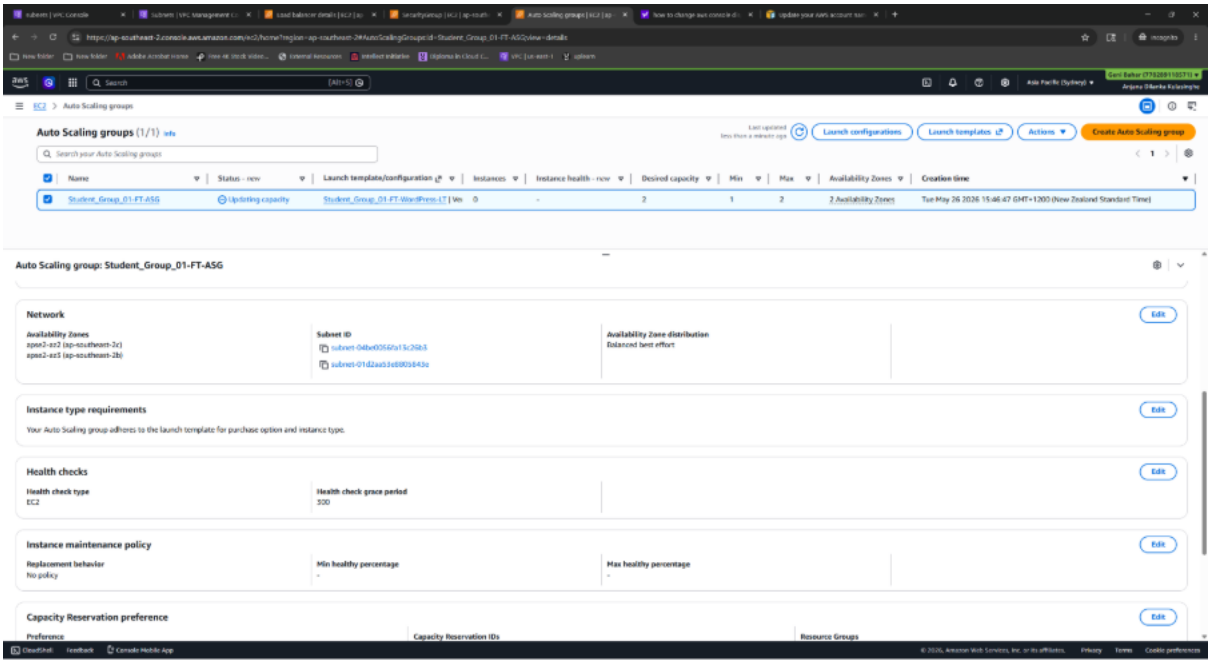




Auto Scaling Group

The Auto Scaling Group manages instance capacity and can replace unhealthy instances.



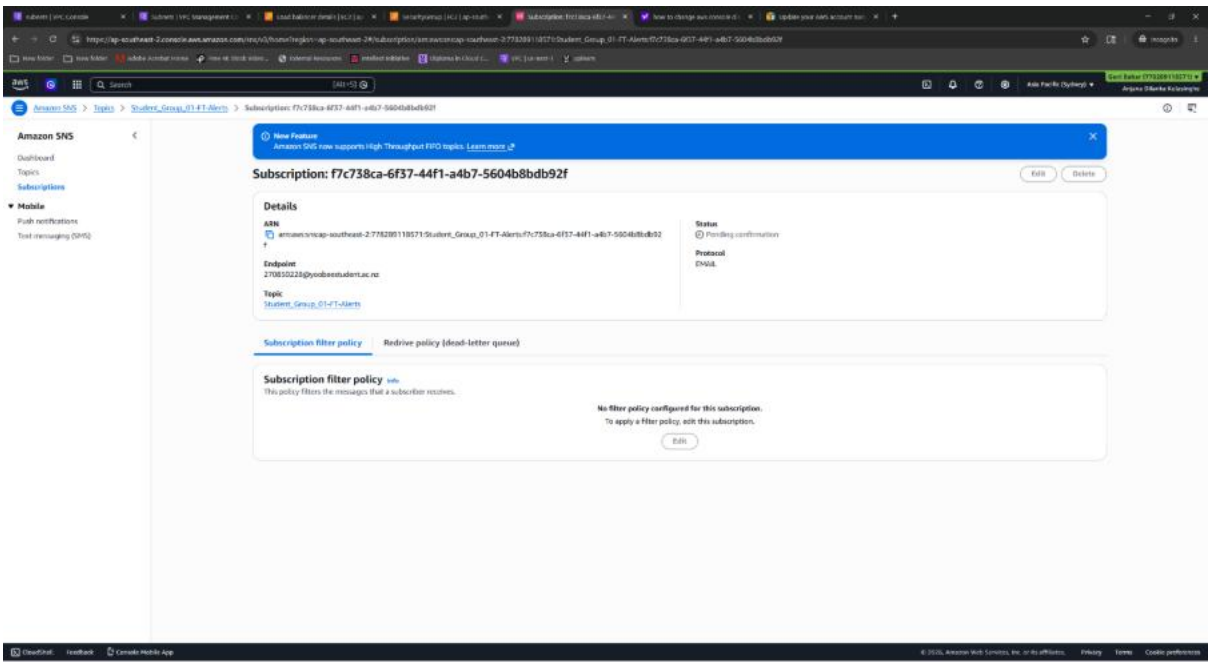


Monitoring and Logging

Monitoring and logging were implemented to support operational visibility and incident detection. CloudWatch was used to monitor EC2 state changes and trigger alerts through SNS. CloudTrail was included to support audit visibility for AWS API activity, such as configuration changes, access attempts, and administrative actions. Together, CloudWatch and CloudTrail improve the team's ability to detect operational issues and investigate potential security incidents.

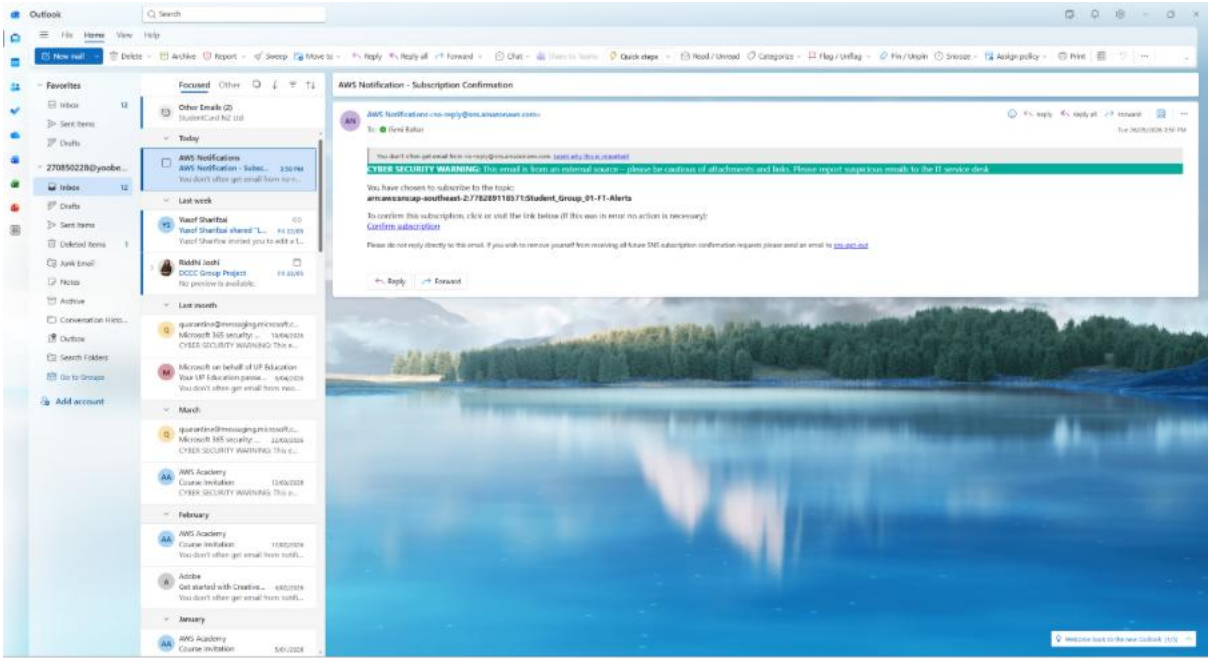
SNS

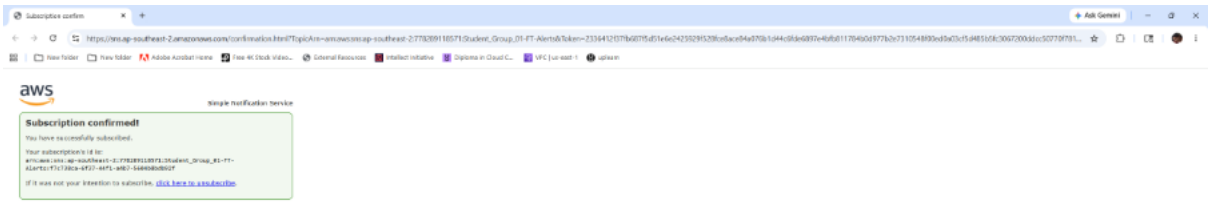
Amazon SNS was configured as the notification service for operational alerts. The service distributes monitoring notifications to subscribed recipients whenever configured events or alarms are triggered.



Alert Confirmation

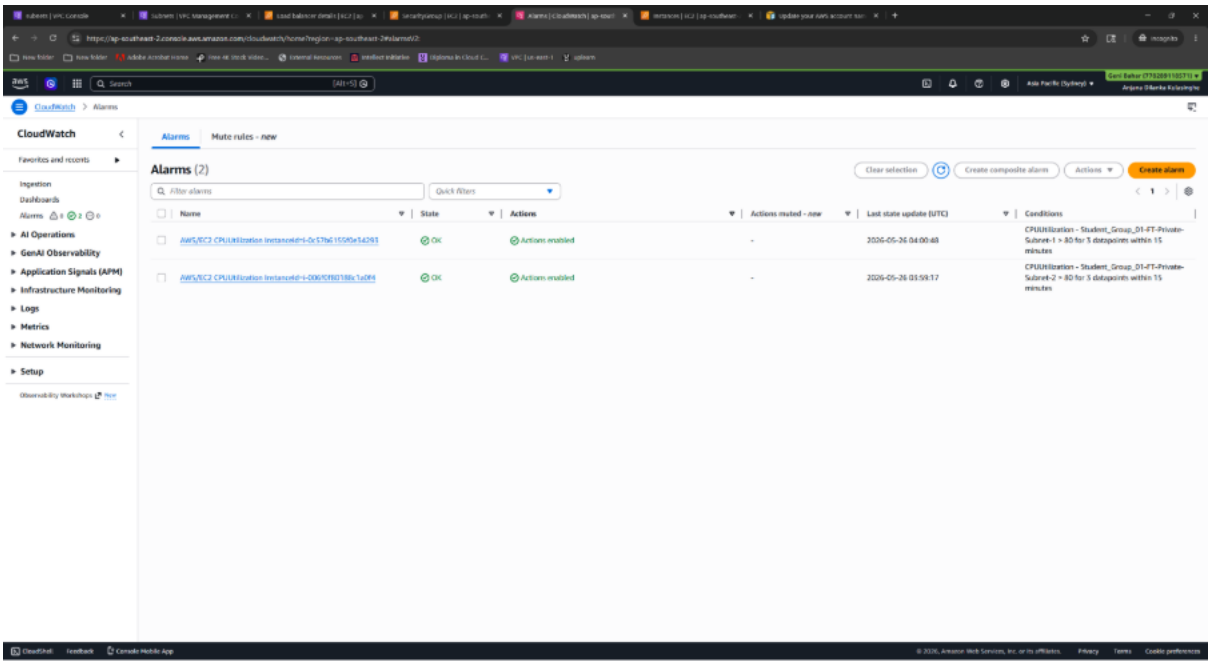
The screenshot confirms successful subscription and delivery of SNS notifications. This validates that monitoring alerts can be received by the project team when required.





CloudWatch Alerts

CloudWatch alarms were configured to monitor selected AWS resources and operational metrics. These alarms provide proactive visibility into infrastructure health and trigger notifications when predefined thresholds are exceeded.



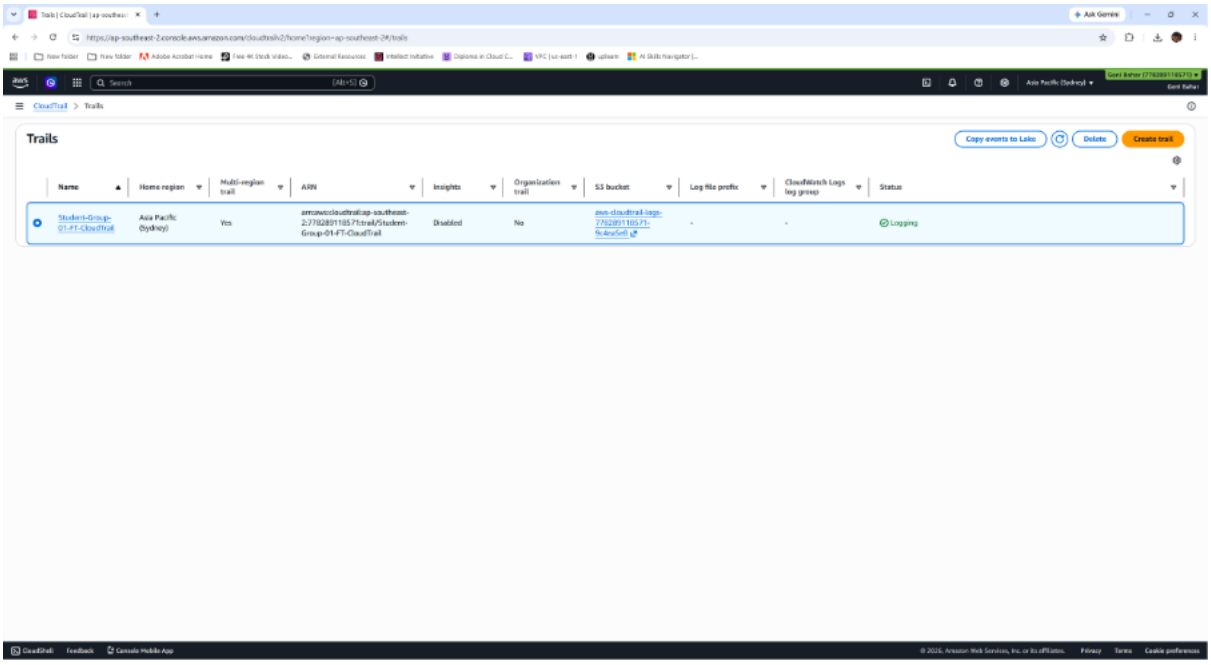
CloudTrail Auditing

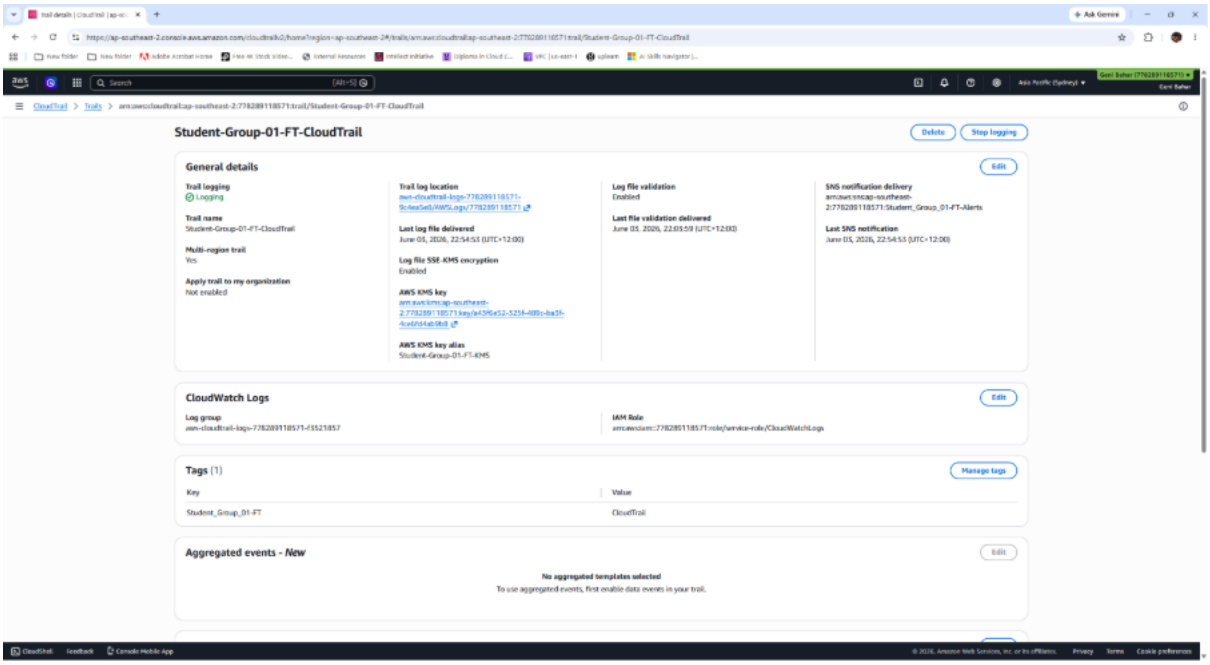
AWS CloudTrail was enabled to provide audit logging and account activity tracking. CloudTrail captures API activity across the AWS environment and stores audit records for security monitoring, compliance reporting, and troubleshooting purposes.

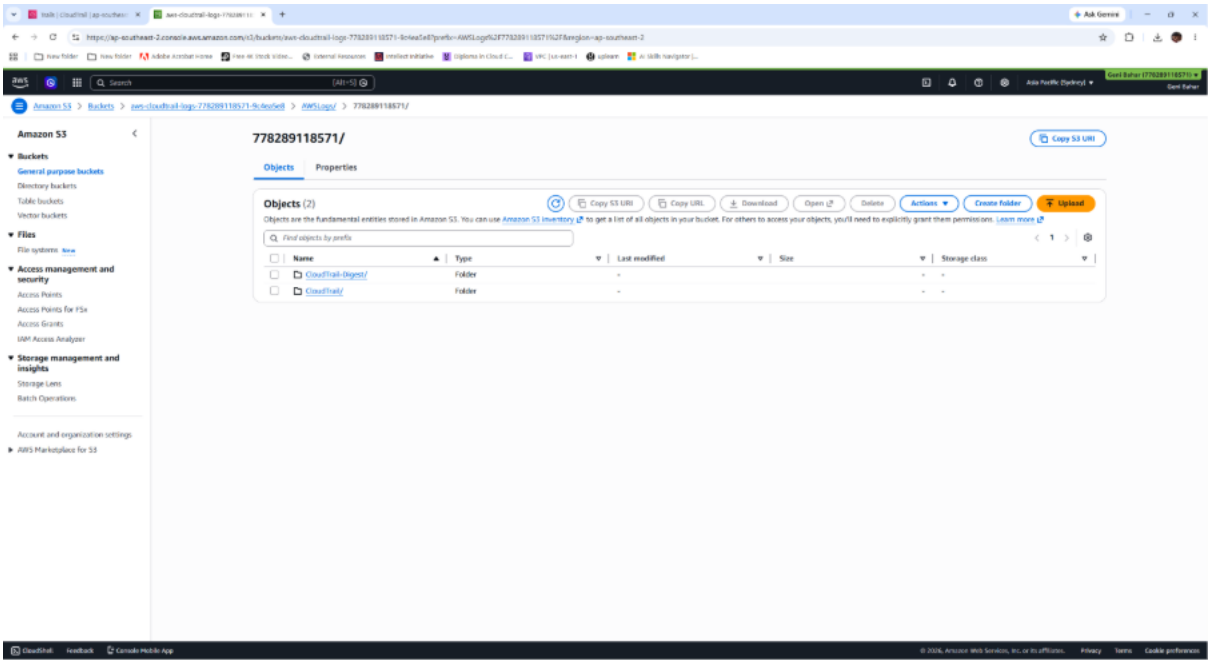
CloudTrail logging was configured to store audit data within Amazon S3 while Event History was used to validate successful activity capture.

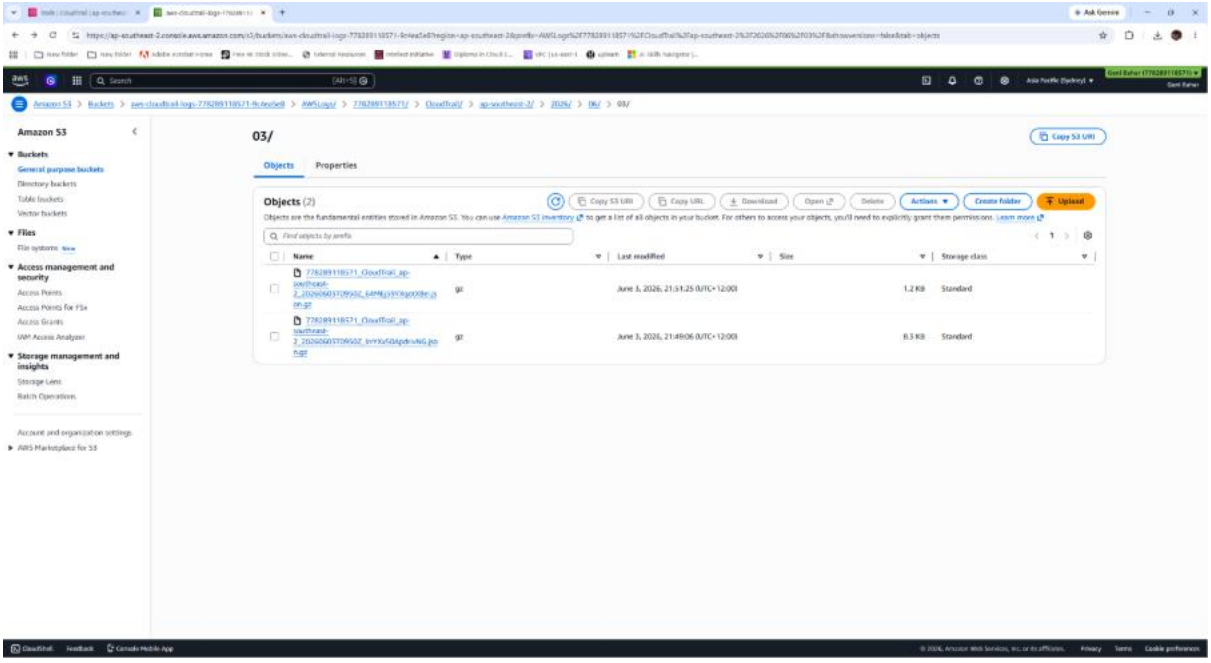
CloudTrail

CloudTrail was enabled to record API activity across the AWS environment. This provides audit visibility, supports security investigations, and assists with compliance and governance requirements.



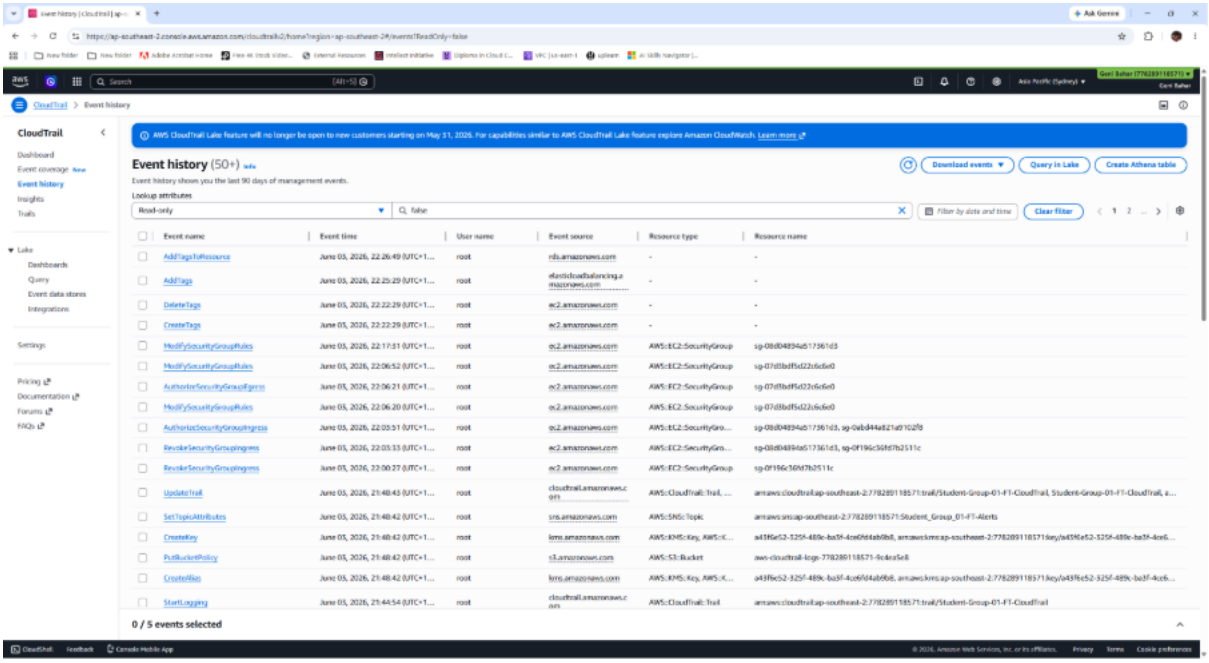



778289118571/
Objects
Properties
Objects (2)
Name
Type
Last modified
Size
Storage class
CloudTrail-Digest/
CloudTrail/


03/
Objects
Properties
Objects (2)
Name
Type
Last modified
Size
Storage class

Event History

CloudTrail Event History was reviewed to verify successful logging of AWS management activities. The recorded events demonstrate that account actions are being captured for auditing and monitoring purposes.

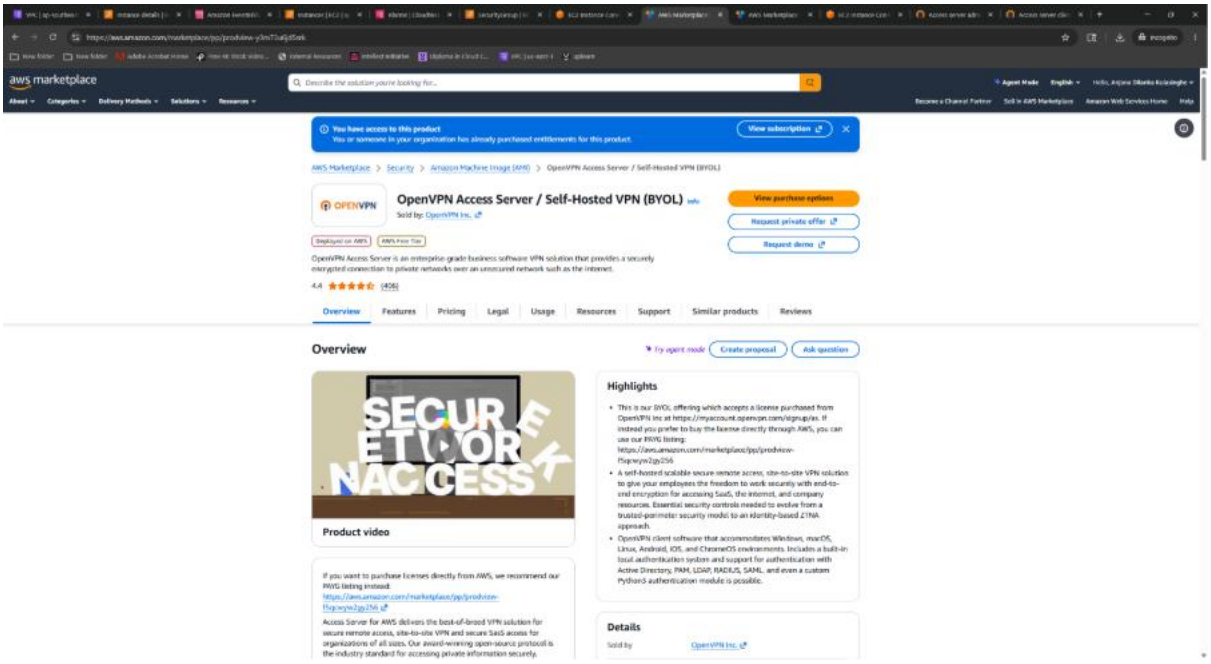


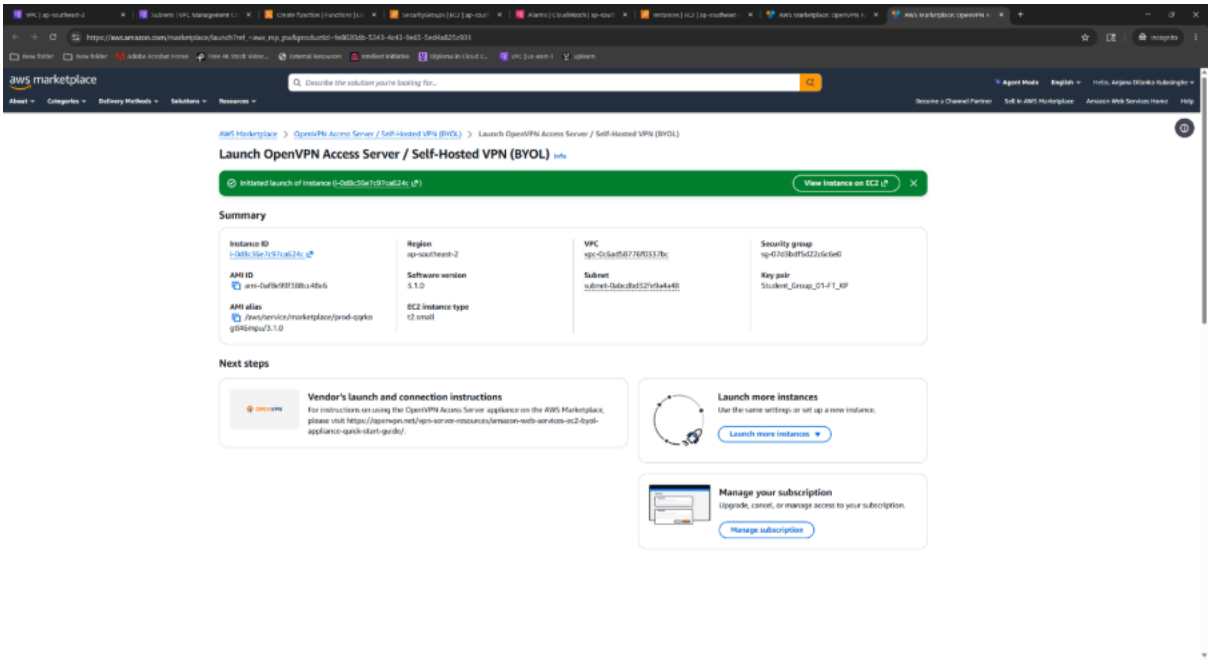
Secure Remote Access

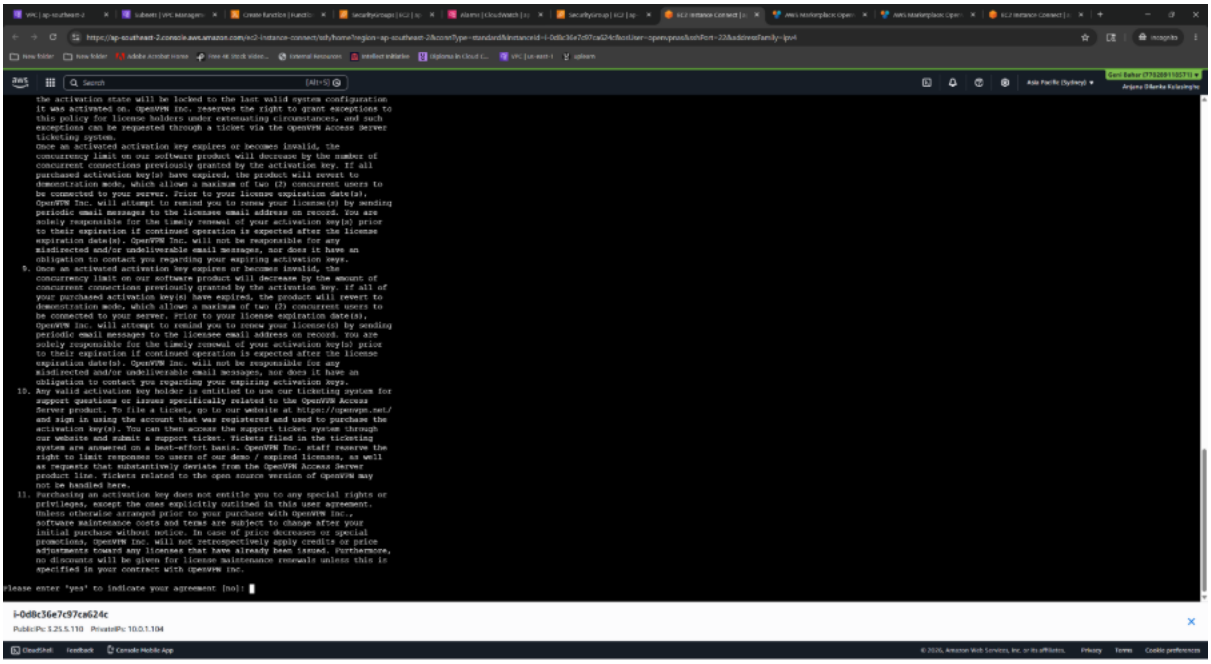
OpenVPN was used to support secure administrative access to private AWS resources without exposing management ports directly to the public internet. This approach reduces the risk of brute-force attacks and unauthorised access by requiring users to connect through a VPN client before reaching internal resources.

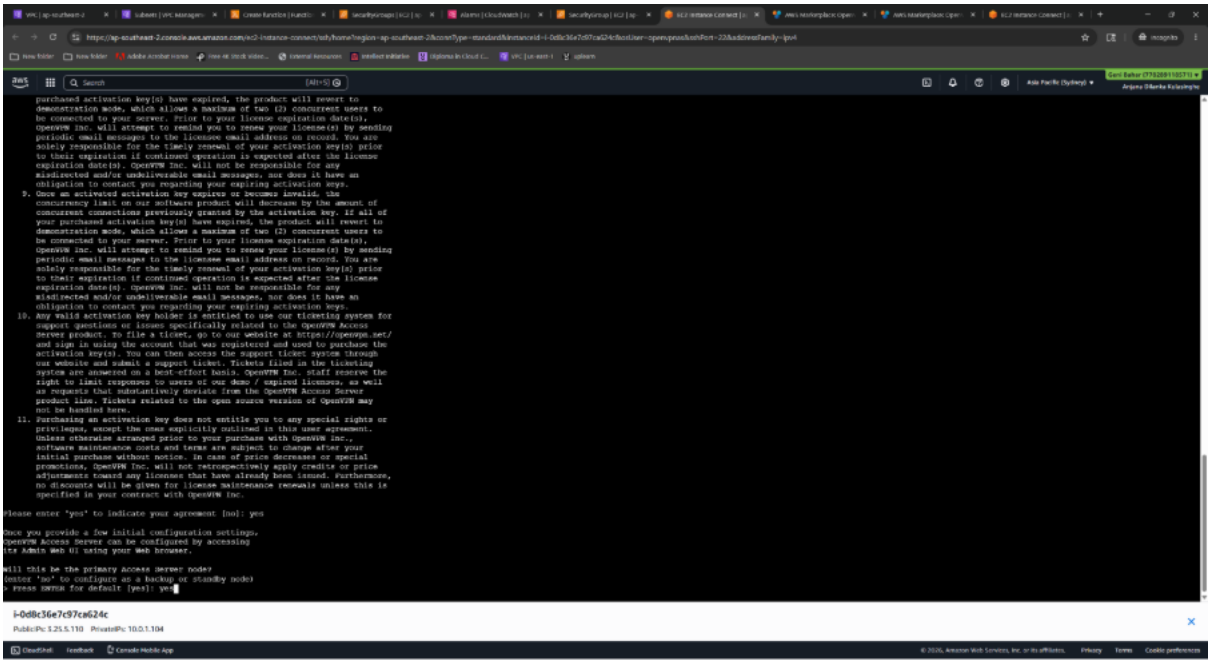
OpenVPN Setup

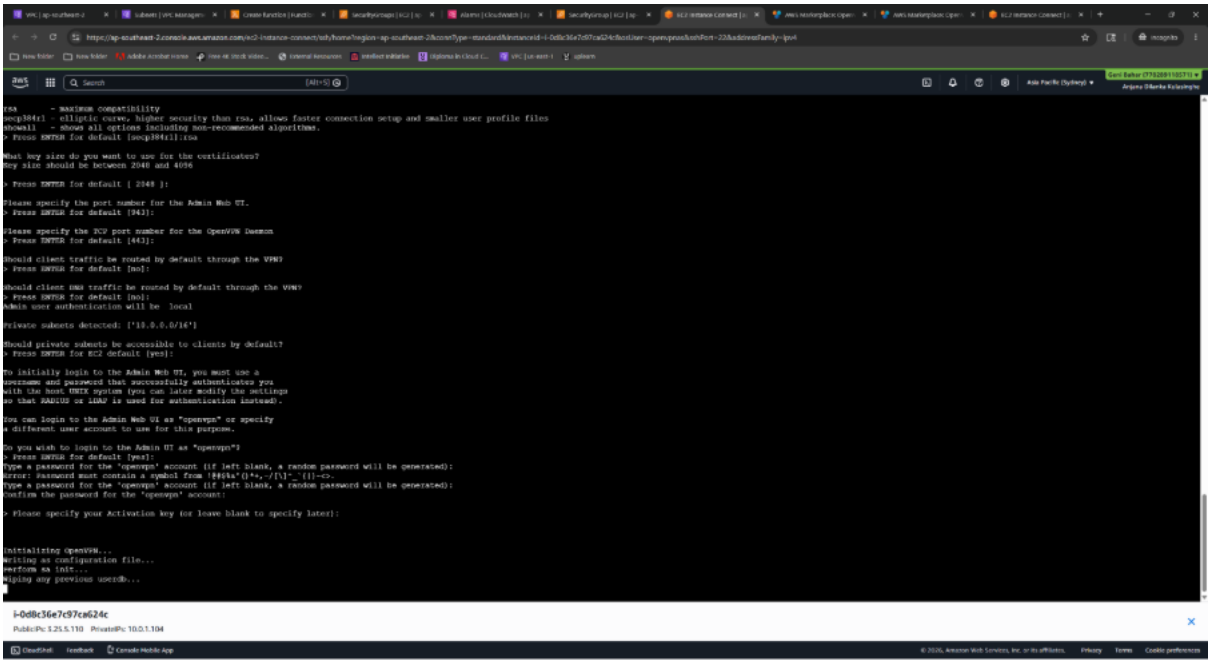
OpenVPN Access Server was deployed to provide secure remote access to AWS resources located within private subnets. The solution allows authorized users to establish encrypted VPN connections and access internal services securely.

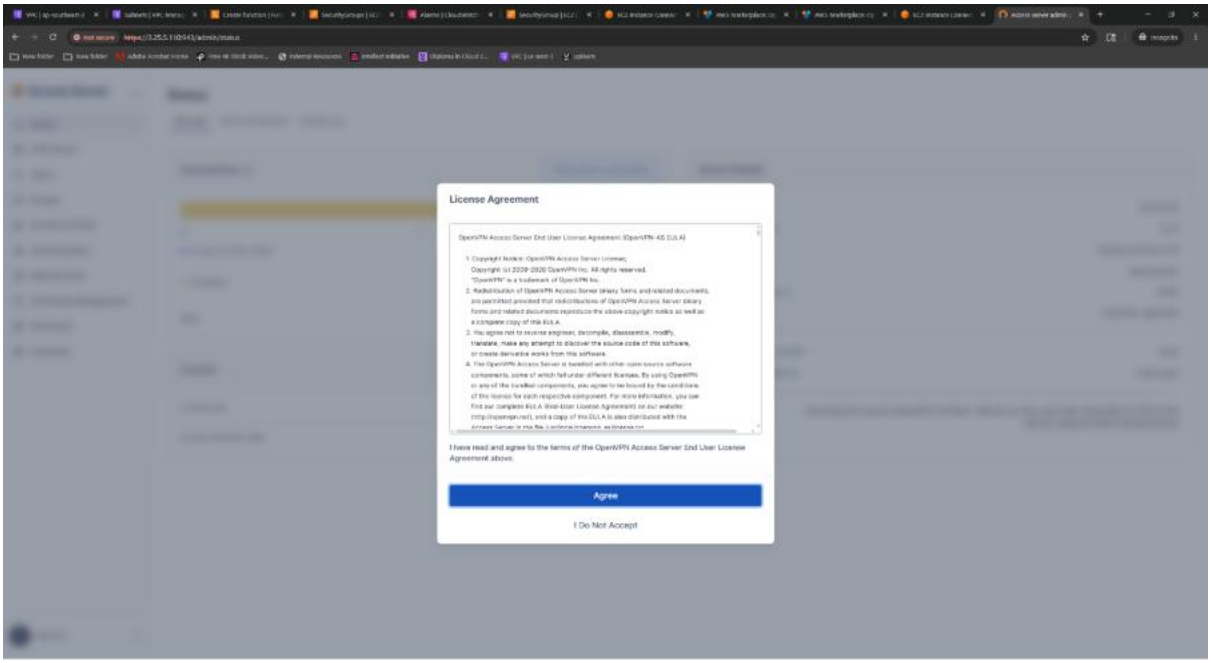


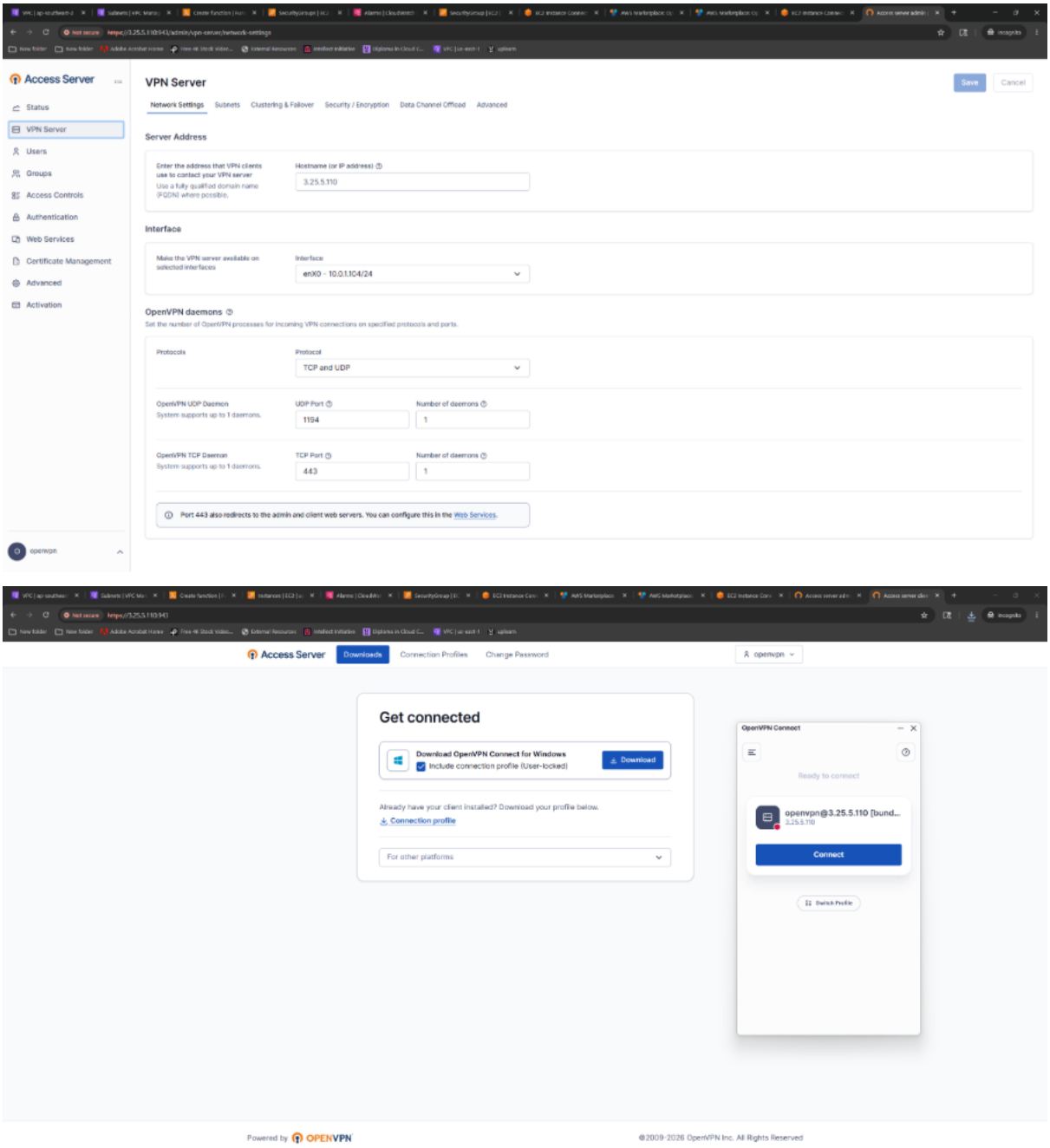



the activation state will be locked to the last valid system configuration
it was activated on. OpenVPN Inc. reserves the right to grant exceptions to
this policy for license holders under extenuating circumstances, and such
exceptions can be requested through a ticket via the OpenVPN Access Server
ticketing system.
Once an activated activation key expires or becomes invalid, the
concurrency limit on our software product will decrease by the number of
concurrent connections previously granted by the activation key. If all
purchased activation key(s) have expired, the product will revert to
demonstration mode, which allows a maximum of two (2) concurrent users to
be connected to your server. Prior to your license expiration date(s),
OpenVPN Inc. will attempt to remind you to renew your license(s) by sending
periodic email messages to the licensee email address on record. You are
solely responsible for the timely renewal of your activation key(s) prior
to their expiration if continued operation is expected after the license
expiration date(s). OpenVPN Inc. will not be responsible for any
misdirected and/or undeliverable email messages, nor does it have an
obligation to contact you regarding your expiring activation keys.
9. Once an activated activation key expires or becomes invalid, the
concurrency limit on our software product will decrease by the amount of
concurrent connections previously granted by the activation key. If all of
your purchased activation key(s) have expired, the product will revert to
demonstration mode, which allows a maximum of two (2) concurrent users to
be connected to your server. Prior to your license expiration date(s),
OpenVPN Inc. will attempt to remind you to renew your license(s) by sending
periodic email messages to the licensee email address on record. You are
solely responsible for the timely renewal of your activation key(s) prior
to their expiration if continued operation is expected after the license
expiration date(s). OpenVPN Inc. will not be responsible for any
misdirected and/or undeliverable email messages, nor does it have an
obligation to contact you regarding your expiring activation keys.
10. Any valid activation key holder is entitled to use our ticketing system for
support questions or issues specifically related to the OpenVPN Access
Server product. To file a ticket, go to our website at https://openvpn.net/
and sign in using the account that was registered and used to purchase the
activation key(s). You can then access the support ticket system through
our website and submit a support ticket. Tickets filed in the ticketing
system are answered on a best-effort basis. OpenVPN Inc. staff reserve the
right to limit responses to users of our demo / expired licenses, as well
as requests that substantively deviate from the OpenVPN Access Server
product line. Tickets related to the open source version of OpenVPN may
not be handled here.
11. Purchasing an activation key does not entitle you to any special rights or
privileges, except the ones explicitly outlined in this user agreement.
Unless otherwise arranged prior to your purchase with OpenVPN Inc.,
software maintenance costs and terms are subject to change after your
initial purchase without notice. In case of price decreases or special
promotions, OpenVPN Inc. will not retrospectively apply credits or price
adjustments toward any licenses that have already been issued. Furthermore,
no discounts will be given for license maintenance renewals unless this is
specified in your contract with OpenVPN Inc.
Please enter 'yes' to indicate your agreement [no]:
i-0d8c36e7c97ca624c

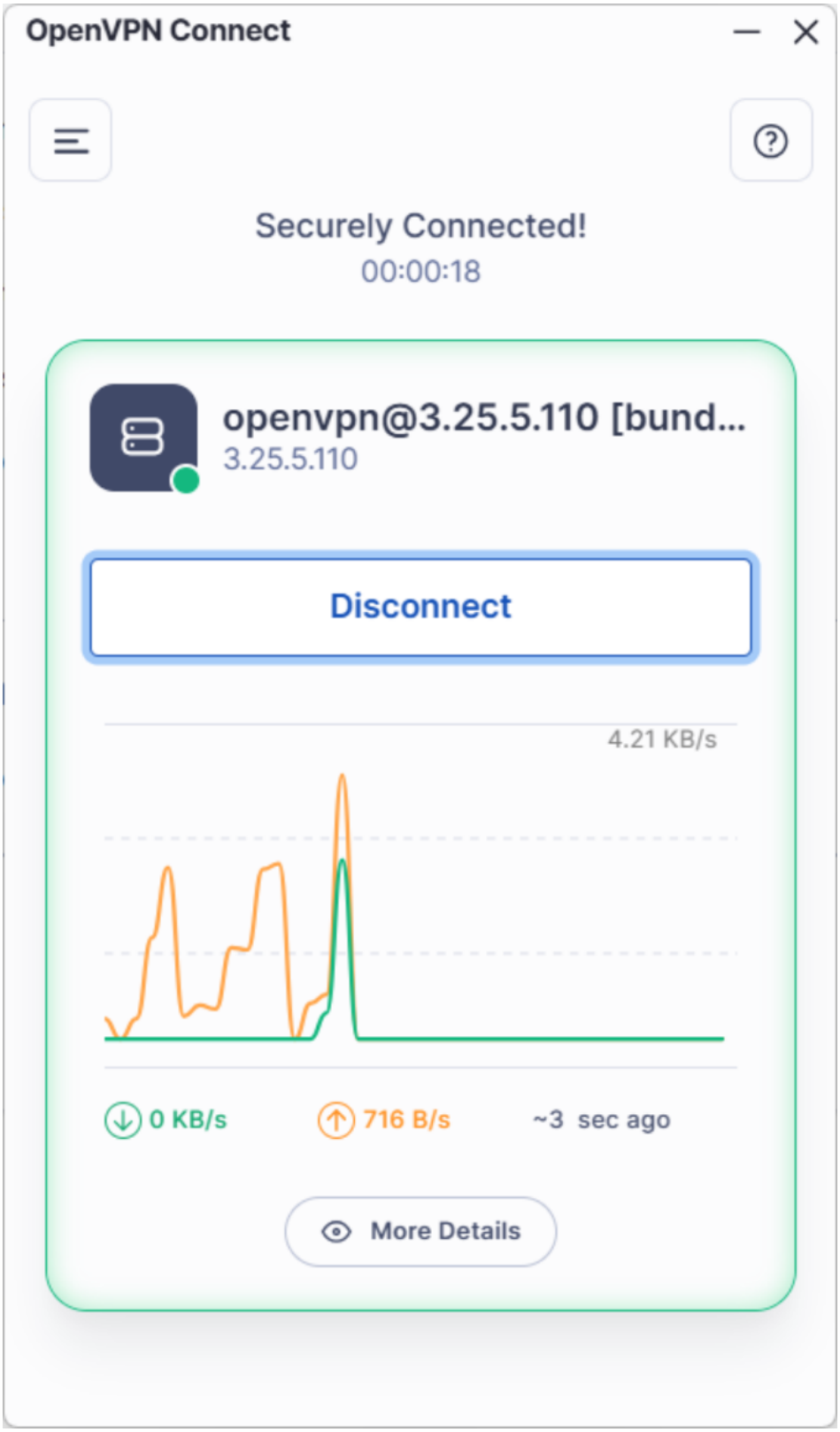

purchased activation key(s) have expired, the product will revert to
demonstration mode, which allows a maximum of two (2) concurrent users to
be connected to your server. Prior to your license expiration date(s),
OpenVPN Inc. will attempt to remind you to renew your license(s) by sending
periodic email messages to the licensee email address on record. You are
solely responsible for the timely renewal of your activation key(s) prior
to their expiration if continued operation is expected after the license
expiration date(s). OpenVPN Inc. will not be responsible for any
misdirected and/or undeliverable email messages, nor does it have an
obligation to contact you regarding your expiring activation keys.
9. Once an activated activation key expires or becomes invalid, the
concurrency limit on our software product will decrease by the amount of
concurrent connections previously granted by the activation key. If all of
your purchased activation key(s) have expired, the product will revert to
demonstration mode, which allows a maximum of two (2) concurrent users to
be connected to your server. Prior to your license expiration date(s),
OpenVPN Inc. will attempt to remind you to renew your license(s) by sending
periodic email messages to the licensee email address on record. You are
solely responsible for the timely renewal of your activation key(s) prior
to their expiration if continued operation is expected after the license
expiration date(s). OpenVPN Inc. will not be responsible for any
misdirected and/or undeliverable email messages, nor does it have an
obligation to contact you regarding your expiring activation keys.
10. Any valid activation key holder is entitled to use our ticketing system for
support questions or issues specifically related to the OpenVPN Access
Server product. To file a ticket, go to our website at https://openvpn.net/
and sign in using the account that was registered and used to purchase the
activation key(s). You can then access the support ticket system through
our website and submit a support ticket. Tickets filed in the ticketing
system are answered on a best-effort basis. OpenVPN Inc. staff reserve the
right to limit responses to users of our demo / expired licenses, as well
as requests that substantively deviate from the OpenVPN Access Server
product line. Tickets related to the open source version of OpenVPN may
not be handled here.
11. Purchasing an activation key does not entitle you to any special rights or
privileges, except the ones explicitly outlined in this user agreement.
Unless otherwise arranged prior to your purchase with OpenVPN Inc.,
software maintenance costs and terms are subject to change after your
initial purchase without notice. In case of price decreases or special
promotions, OpenVPN Inc. will not retrospectively apply credits or price
adjustments toward any licenses that have already been issued. Furthermore,
no discounts will be given for license maintenance renewals unless this is
specified in your contract with OpenVPN Inc.
Please enter 'yes' to indicate your agreement [no]: yes
Once you provide a few initial configuration settings,
OpenVPN Access Server can be configured by accessing
its Admin Web UI using your Web browser.
Will this be the primary Access Server node?
(enter 'no' to configure as a backup or standby node)
> Press ENTER for default [yes]: yes
i-0d8c36e7c97ca624c


rsa - maximum compatibility
secp384r1 - elliptic curve, higher security than rsa, allows faster connection setup and smaller user profile files
showall - shows all options including non-recommended algorithms.
> Press ENTER for default [secp384r1]:rsa
What key size do you want to use for the certificates?
Key size should be between 2048 and 4096
> Press ENTER for default [2048]:
Please specify the port number for the Admin Web UI.
> Press ENTER for default [943]:
Please specify the TCP port number for the OpenVPN Daemon
> Press ENTER for default [443]:
Should client traffic be routed by default through the VPN?
> Press ENTER for default [no]:
Should client DNS traffic be routed by default through the VPN?
> Press ENTER for default [no]:
Admin user authentication will be local
Private subnets detected: ['10.0.0.0/16']
Should private subnets be accessible to clients by default?
> Press ENTER for EC2 default [yes]:
To initially login to the Admin Web UI, you must use a
username and password that successfully authenticates you
with the host UNIX system (you can later modify the settings
so that RADIUS or LDAP is used for authentication instead).
You can login to the Admin Web UI as "openvpn" or specify
a different user account to use for this purpose.
Do you wish to login to the Admin UI as "openvpn"?
> Press ENTER for default [yes]:
Type a password for the 'openvpn' account (if left blank, a random password will be generated):
Error: Password must contain a symbol from !@#$%^&*()+=,-./[\]'_`{|}~<>.
Type a password for the 'openvpn' account (if left blank, a random password will be generated):
Confirm the password for the 'openvpn' account:
> Please specify your Activation key (or leave blank to specify later):
Initializing OpenVPN...
Writing as configuration file...
Perform sa init...
Wiping any previous userdb...
i-0d8c36e7c97ca624c


License Agreement
I have read and agree to the terms of the OpenVPN Access Server End User License Agreement above.
Agree
I Do Not Accept



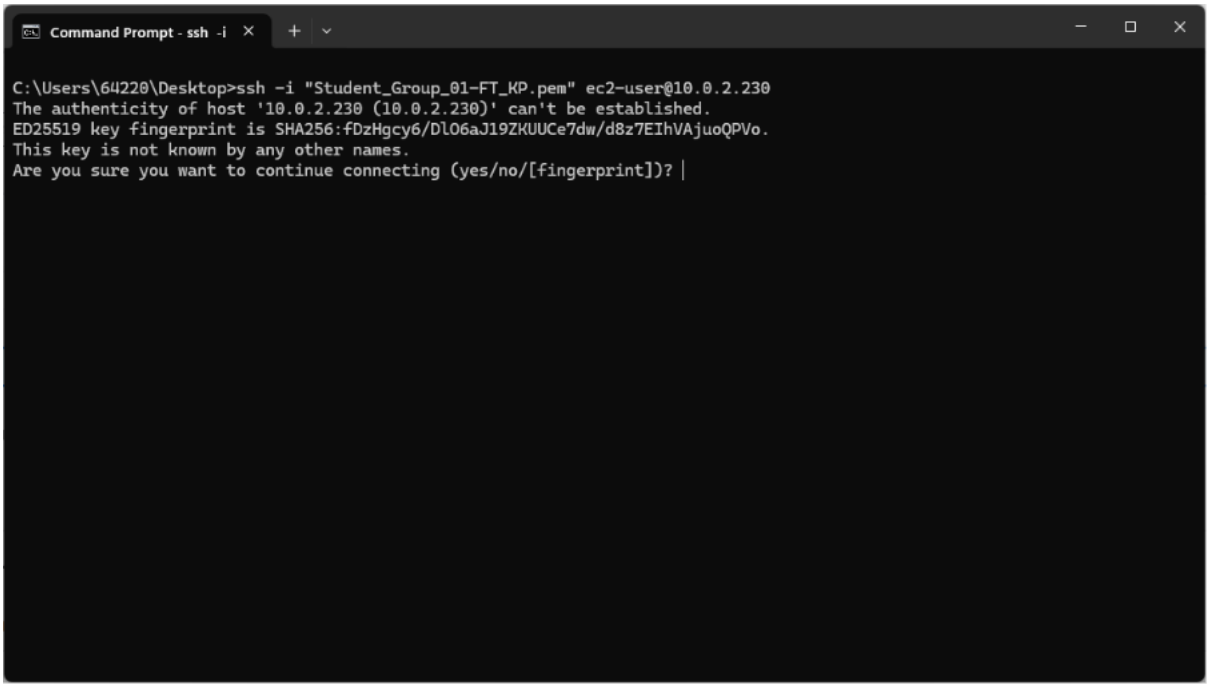
The screenshot confirms successful VPN connectivity between the client device and the AWS environment. Once connected, internal resources become accessible through private network addresses.

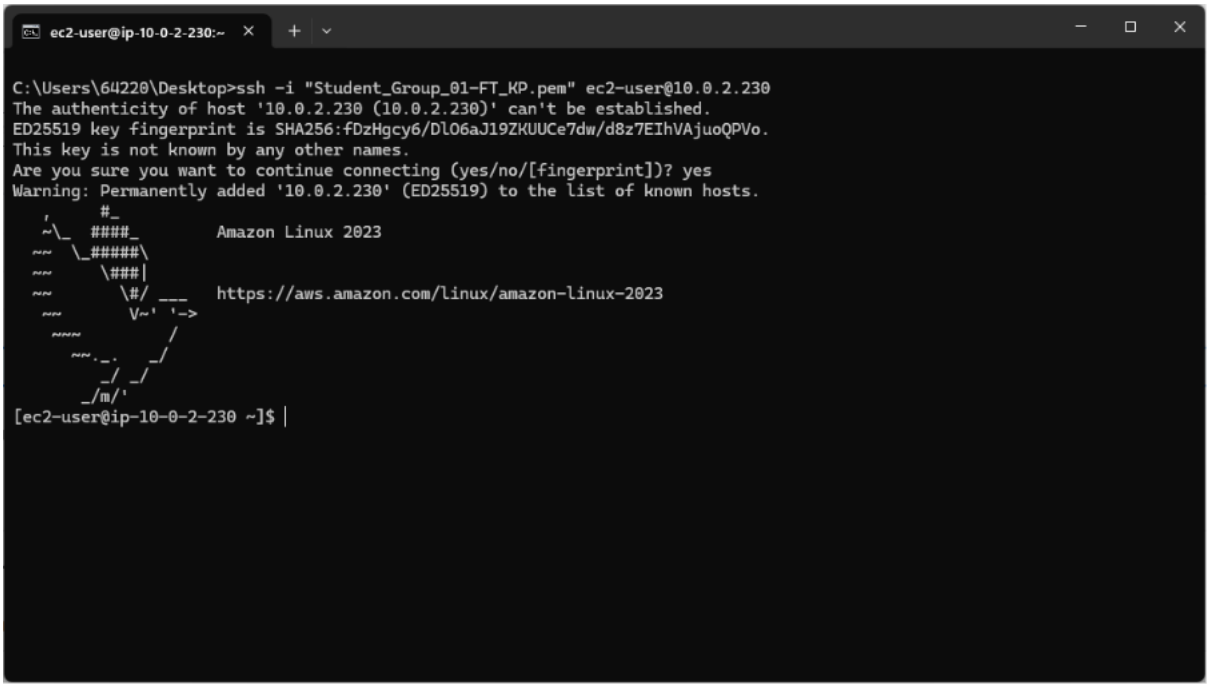

OpenVPN Connect
Securely Connected!
00:00:18
openvpn@3.25.5.110 [bund...
3.25.5.110
Disconnect
4.21 KB/s
0 KB/s
716 B/s
~3 sec ago
More Details

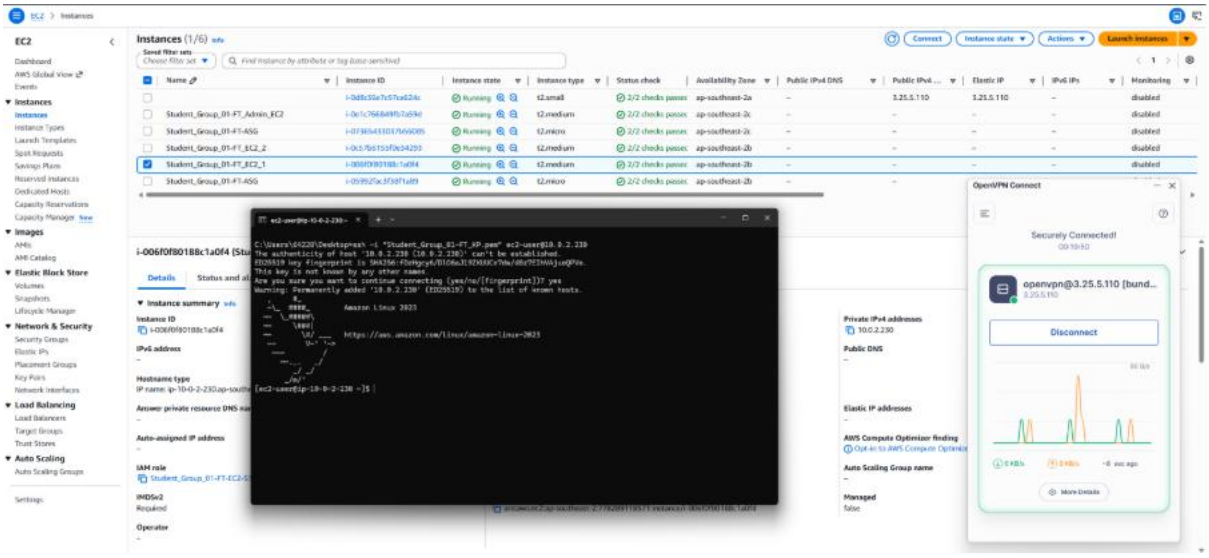
VPN Connectivity Validation

Open VPN test (Connect to Web EC2 with VPN)

Connectivity testing confirmed that authorized users can access internal AWS resources after establishing a VPN connection. This validates the effectiveness of the secure remote access implementation.



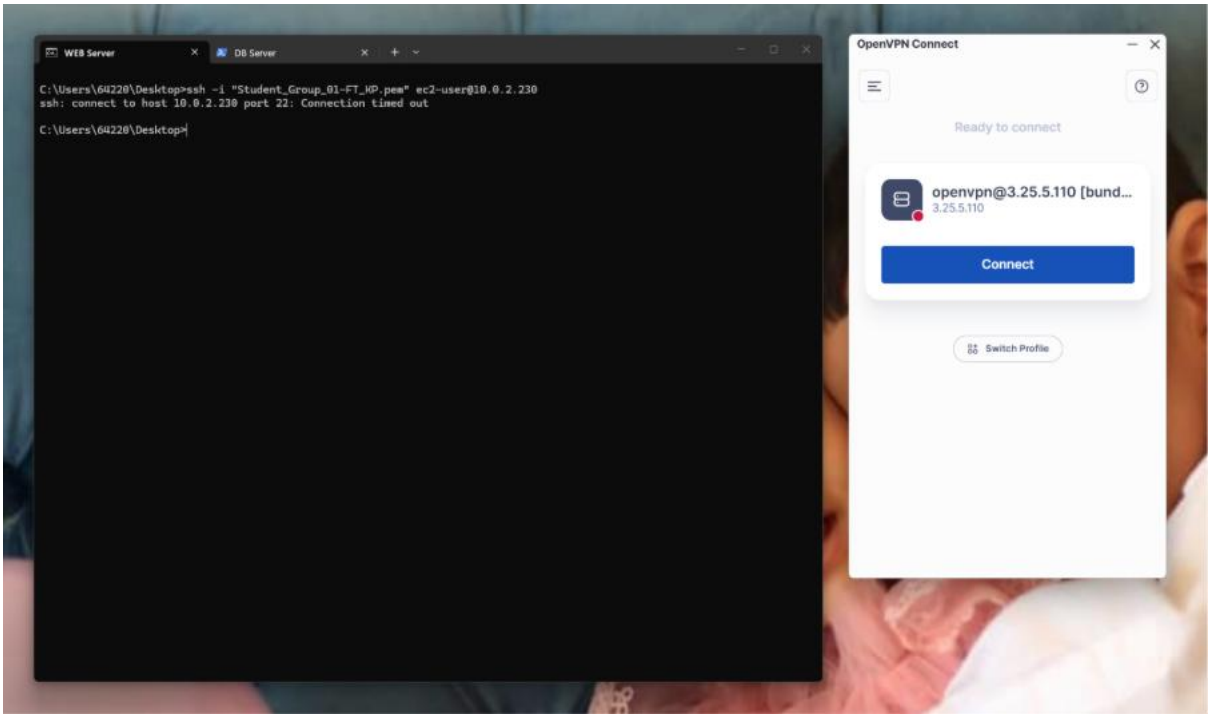




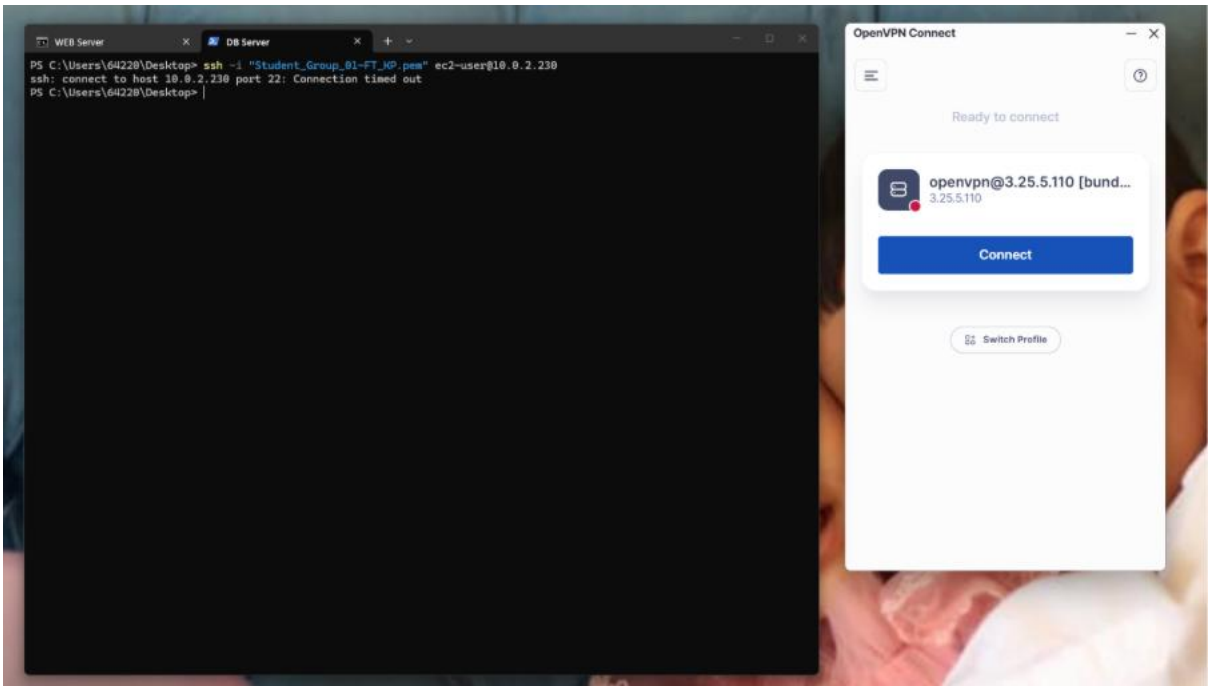
Without VPN

The tests demonstrate that private AWS resources cannot be accessed directly from the public internet. This confirms that network segmentation and security controls are functioning as intended.

Connect to Web Server



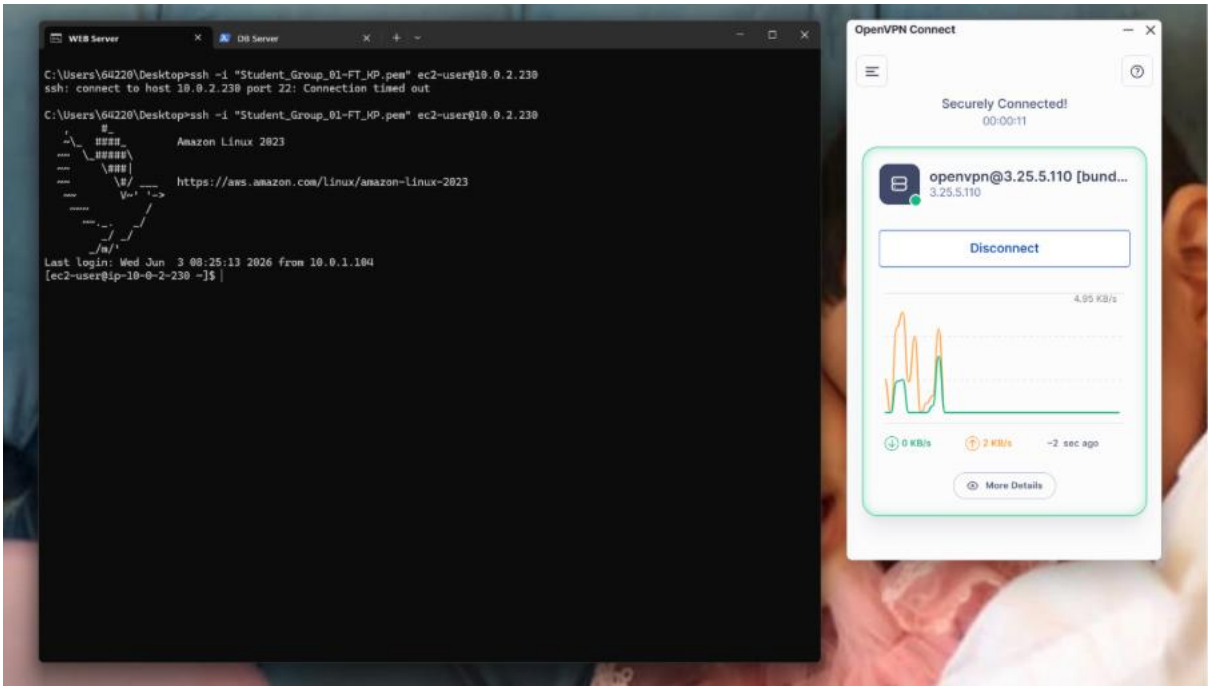
Connect to Private Application Server



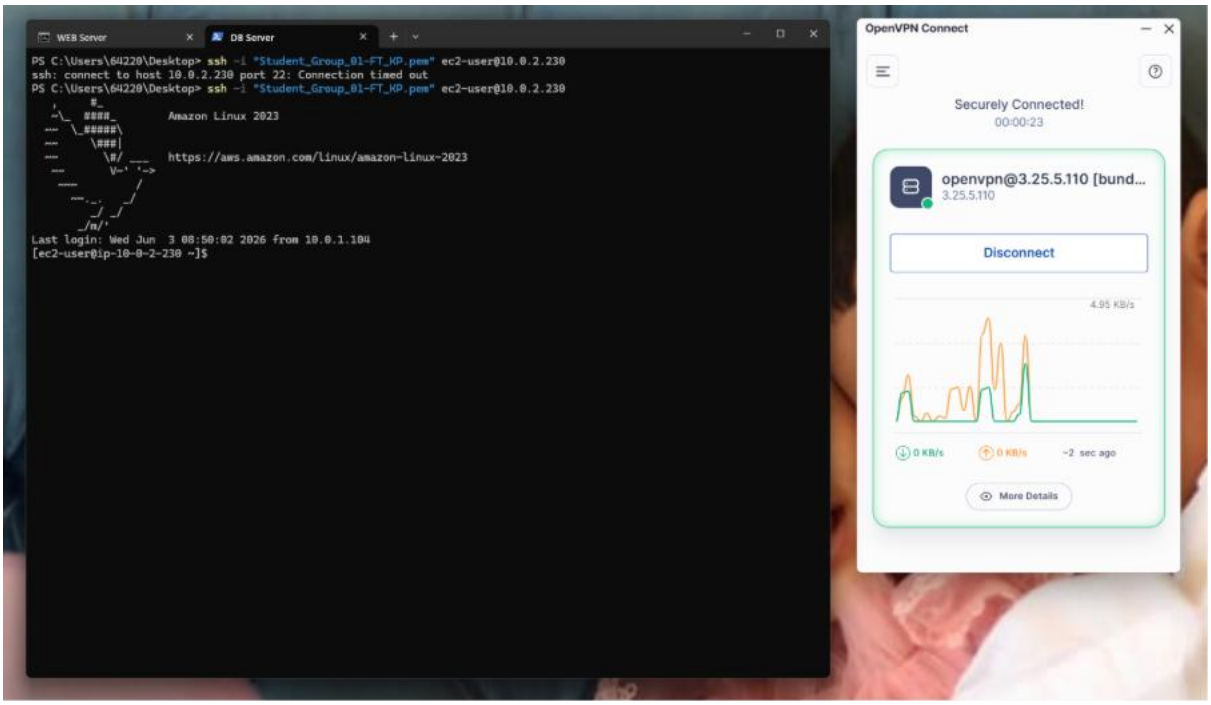
With VPN

After establishing the VPN connection, access to private resources was successfully restored. This validates that secure access is available only through authorised VPN channels.

Connect to Web Server



Connect to Private Application Server

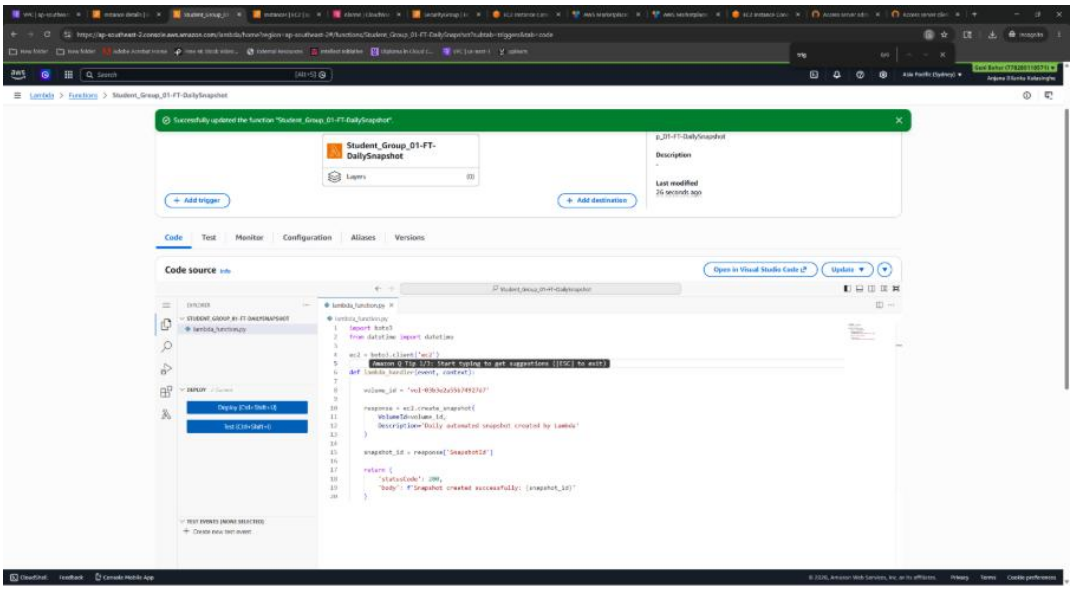


Backup Automation

Lambda was used to automate snapshot creation so that backups are not dependent on manual action. The function is triggered on a schedule and uses an IAM role with the required EC2 snapshot permissions. CloudWatch logs should be used to validate execution success or troubleshoot errors.

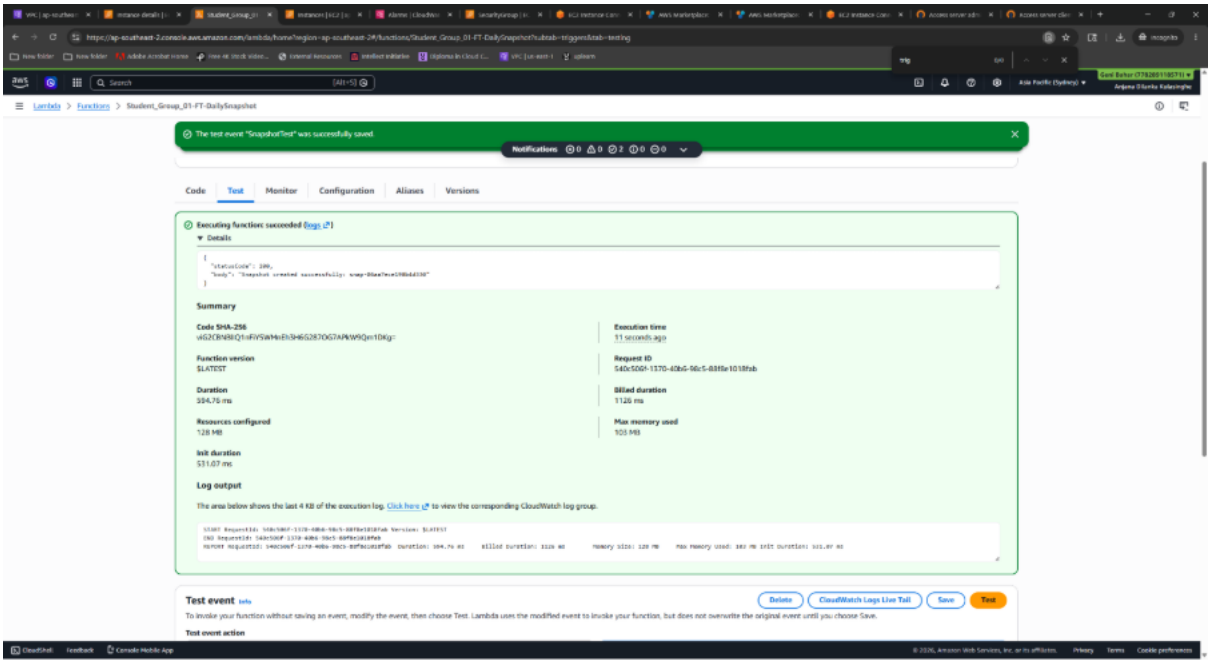
Lambda Function

The Lambda function was developed to automate snapshot creation for selected EBS volumes. Automation reduces administrative effort and improves backup consistency.



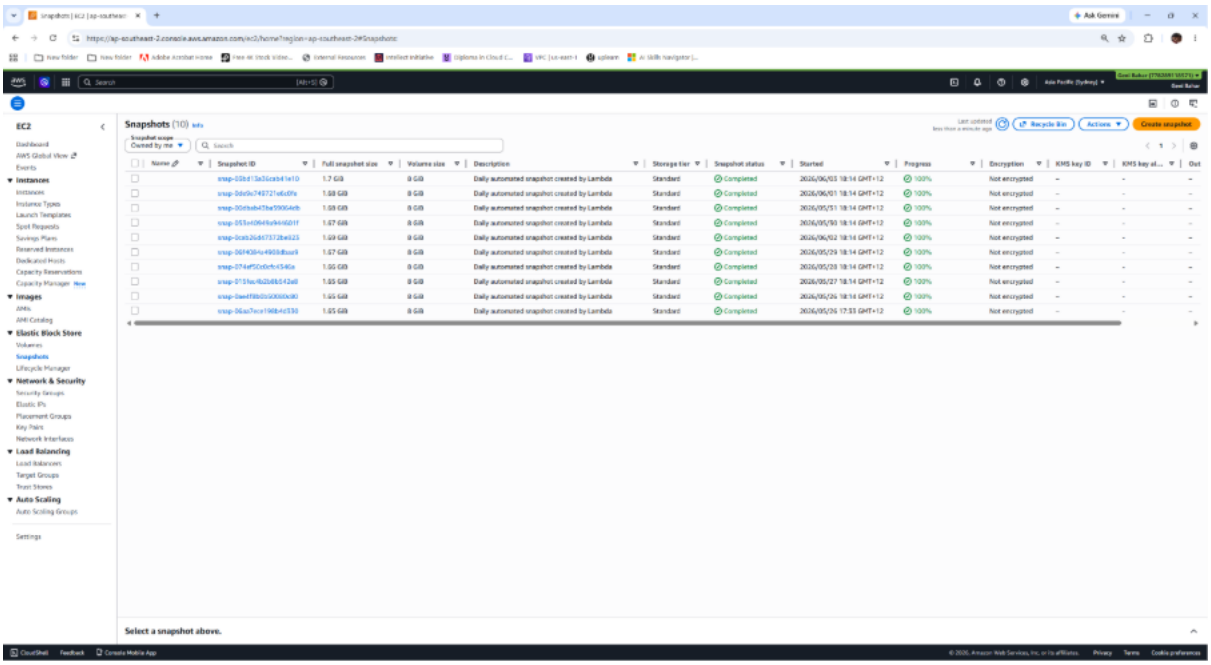
Lambda Function Test

The function was executed successfully and returned to a successful completion status, confirming that the automation workflow is operating correctly.



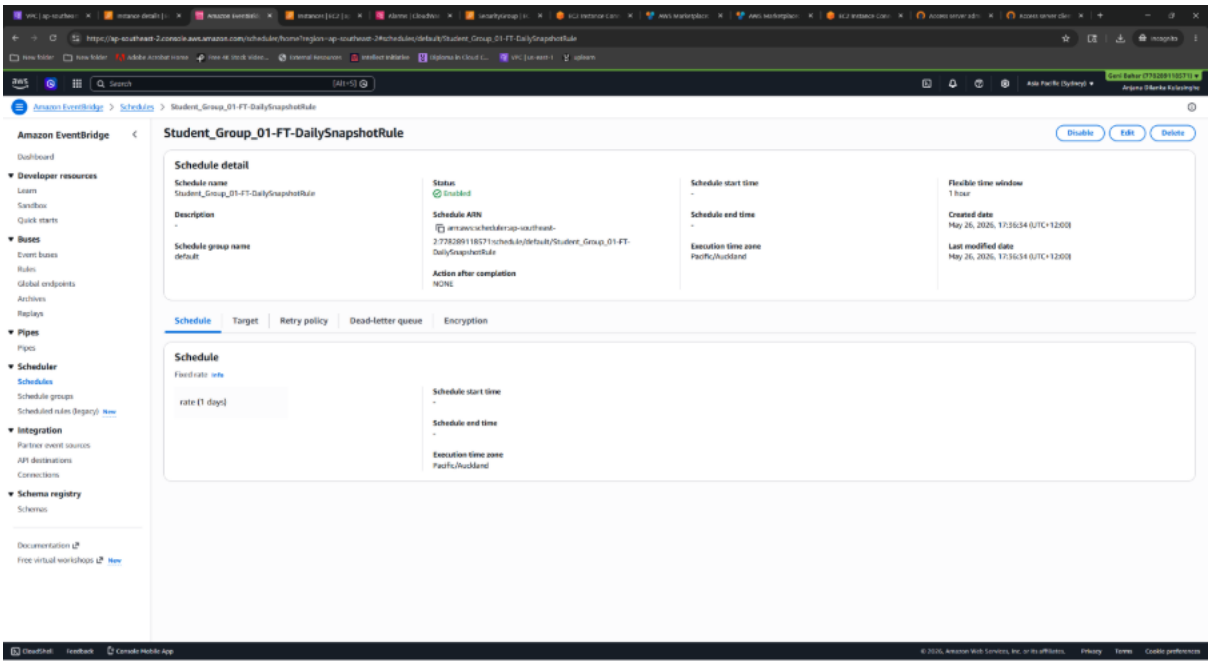
Snapshot Verification

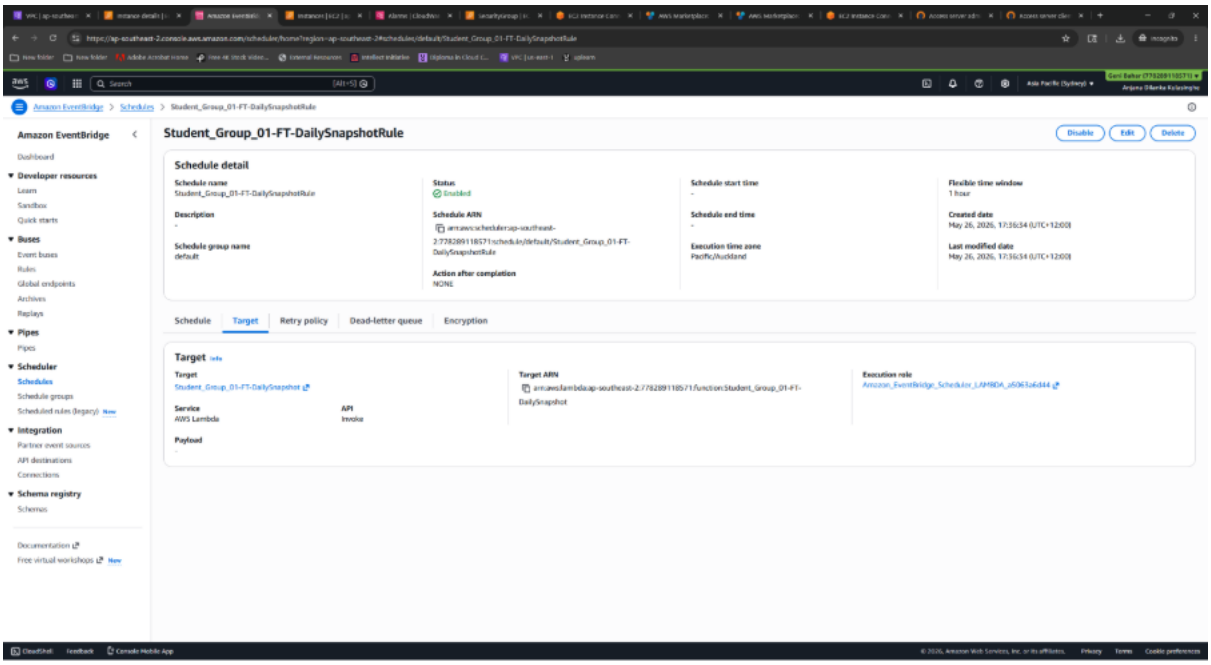
The newly created snapshot appears within the AWS console, confirming that the Lambda function successfully completed the backup operation.



Event Bridge Schedule

Amazon Event Bridge was configured to trigger the Lambda function automatically on a daily schedule. This ensures backups occur consistently without manual intervention.





Resource Tagging

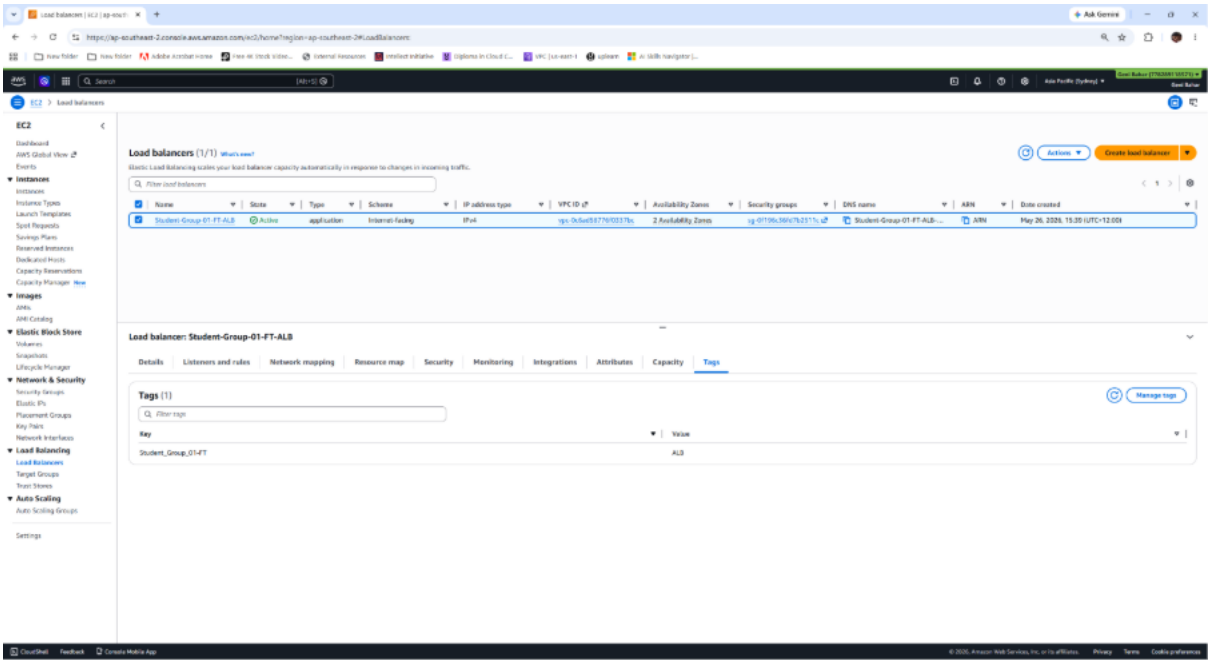
Resource tagging was implemented throughout the AWS environment to improve resource identification, management, cost tracking, and governance.

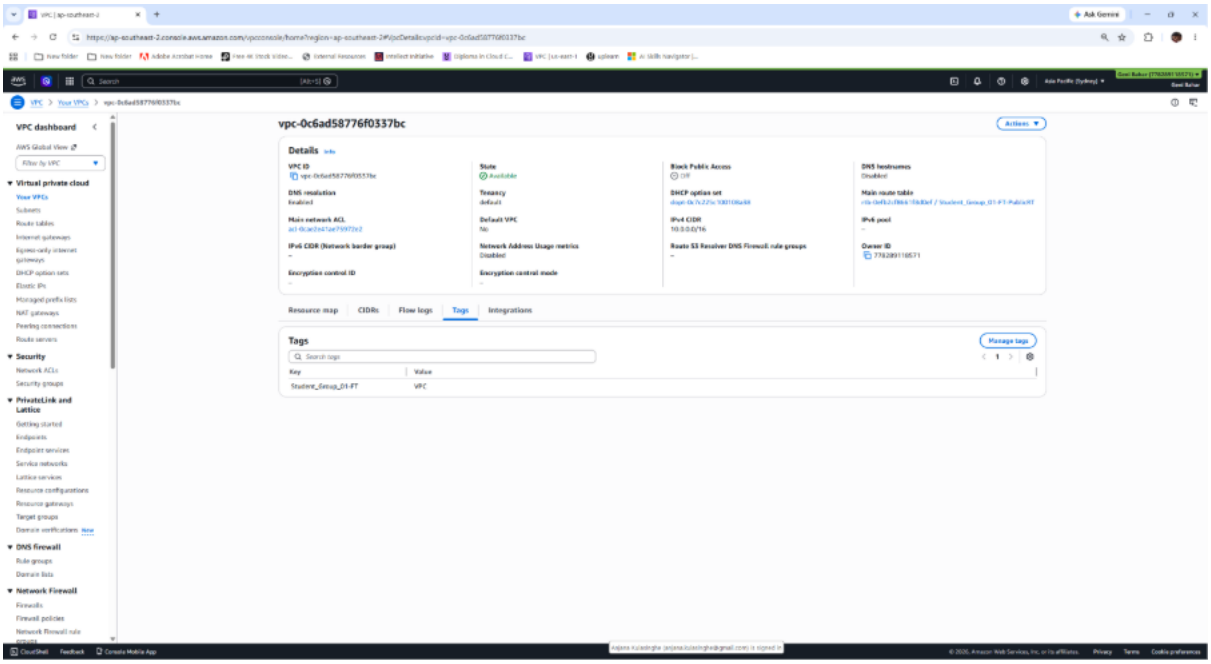
A consistent tagging strategy was used across AWS resources, following the project naming convention required by the assessment. Tags were applied to key infrastructure components including the VPC, subnets, EC2 instances, Security Groups, Load Balancer, database resources, and supporting services.

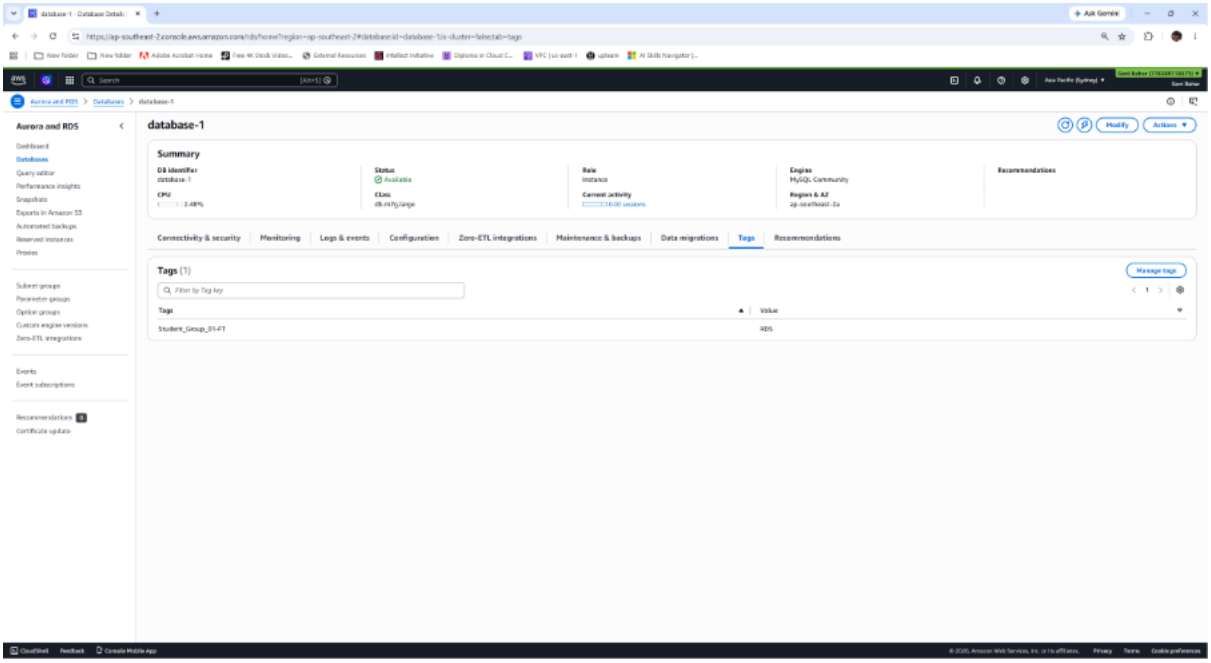
Tagging enables easier administration, simplifies reporting, and supports future operational management activities within the cloud environment.

The tagging strategy also supports future cost allocation, operational reporting, and resource lifecycle management.

All major AWS resources were tagged using the assessment naming requirement: **Key = Student_Group_01-FT** and **Value = resource identity**. This supports resource tracking, cost visibility, and assessment verification.




vpc-0c6ad58776f0337bc
Details
Resource map
CIDRs
Flow logs
Tags
Integrations
Tags
Key
Value
Student_Group_01-FT
VPC


database-1
Summary
Connectivity & security
Monitoring
Logs & events
Configuration
Zero-ETL integrations
Maintenance & backups
Data migrations
Tags
Recommendations
Tags (1)
Tag
Value
Student_Group_01-FT
RDS

WordPress Application Deployment

WordPress was deployed on the AWS application layer to demonstrate a realistic migration target for Yoobee College's web-based infrastructure. The application connects to the RDS database back end and is accessed through the load-balanced HTTPS endpoint. This validates the core functional requirement of hosting a web application in the cloud.

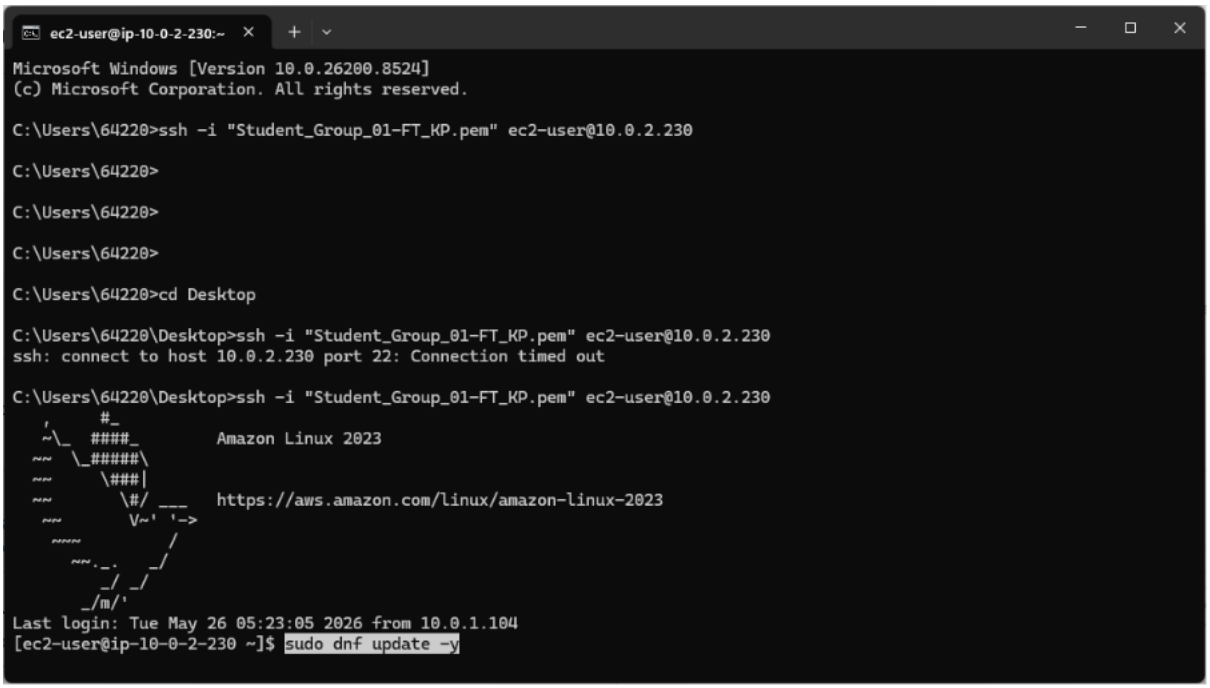
Web Application Setup

A WordPress application was deployed on an Amazon EC2 instance and integrated with Amazon RDS MySQL for database services.

The application architecture separates the web and database tiers to improve security and scalability. WordPress configuration was completed successfully and connectivity between the web server and database server was validated.

WordPress Installation

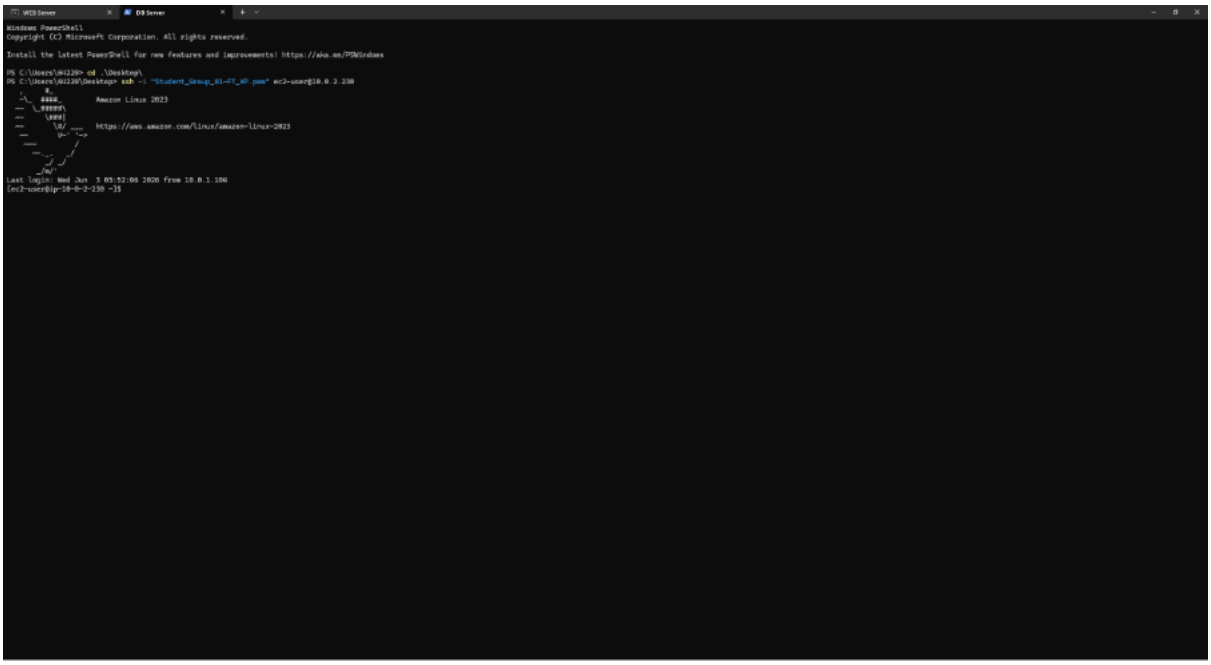
Apache, PHP, and WordPress components were installed on the EC2 instance. The web server was configured to host the WordPress application and communicate with the backend database service.

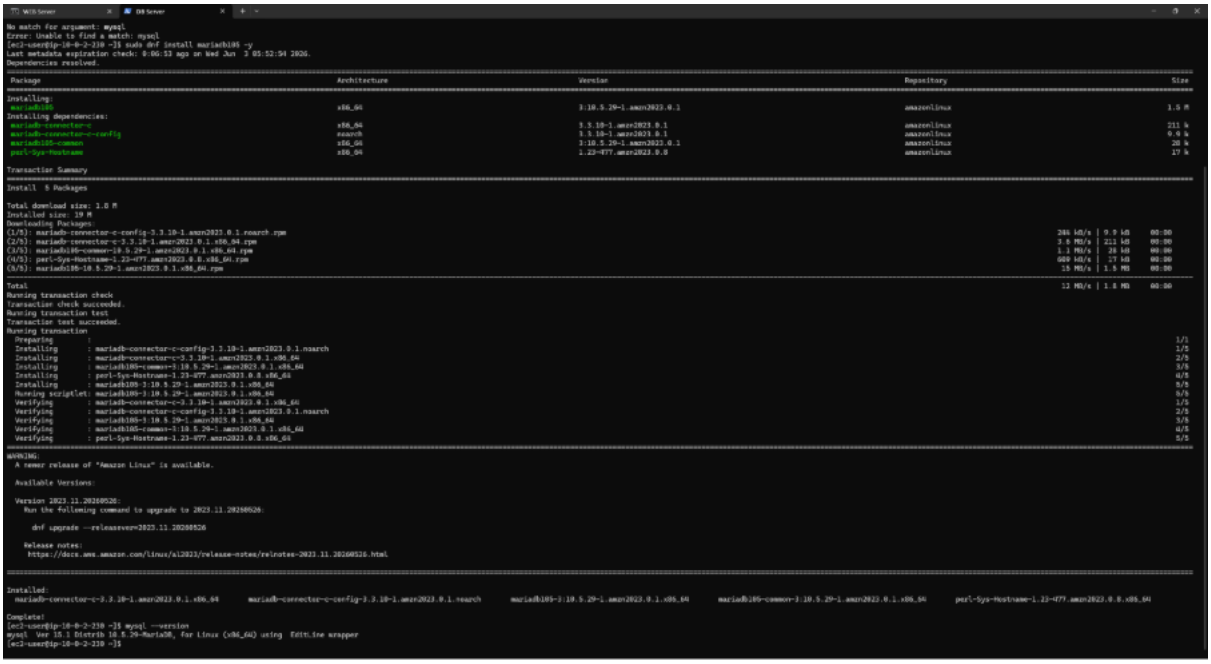


Database Setup

MySQL Installation

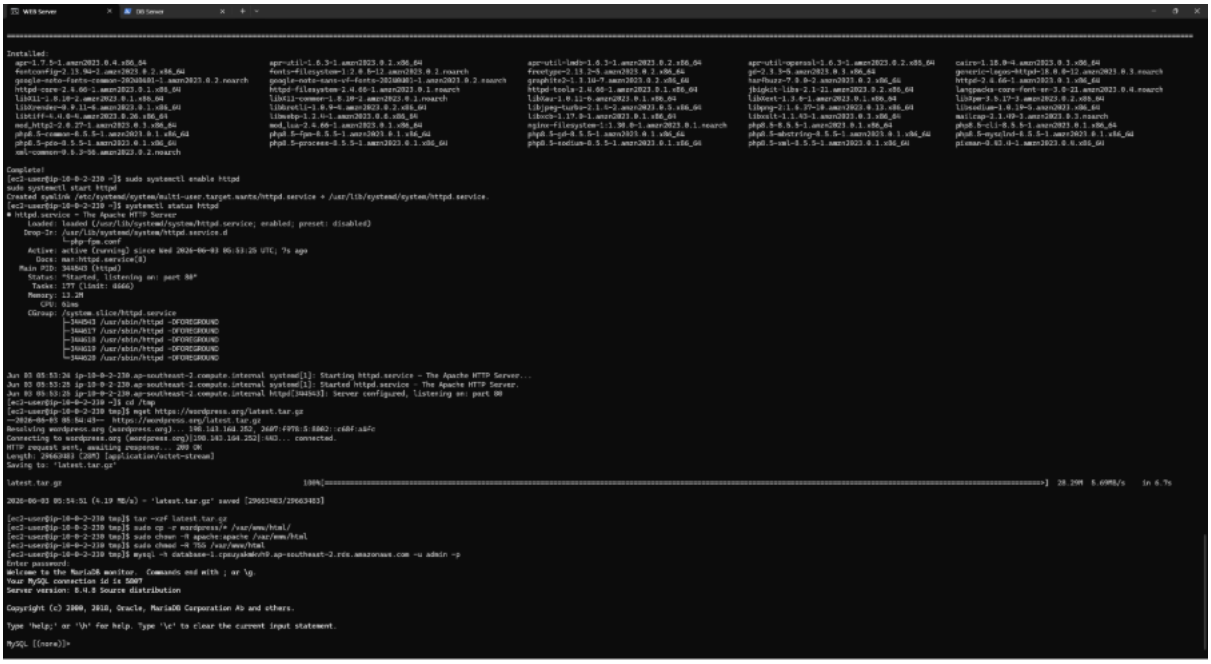
MySQL client tools were installed on the application server to facilitate connectivity testing and database administration activities.

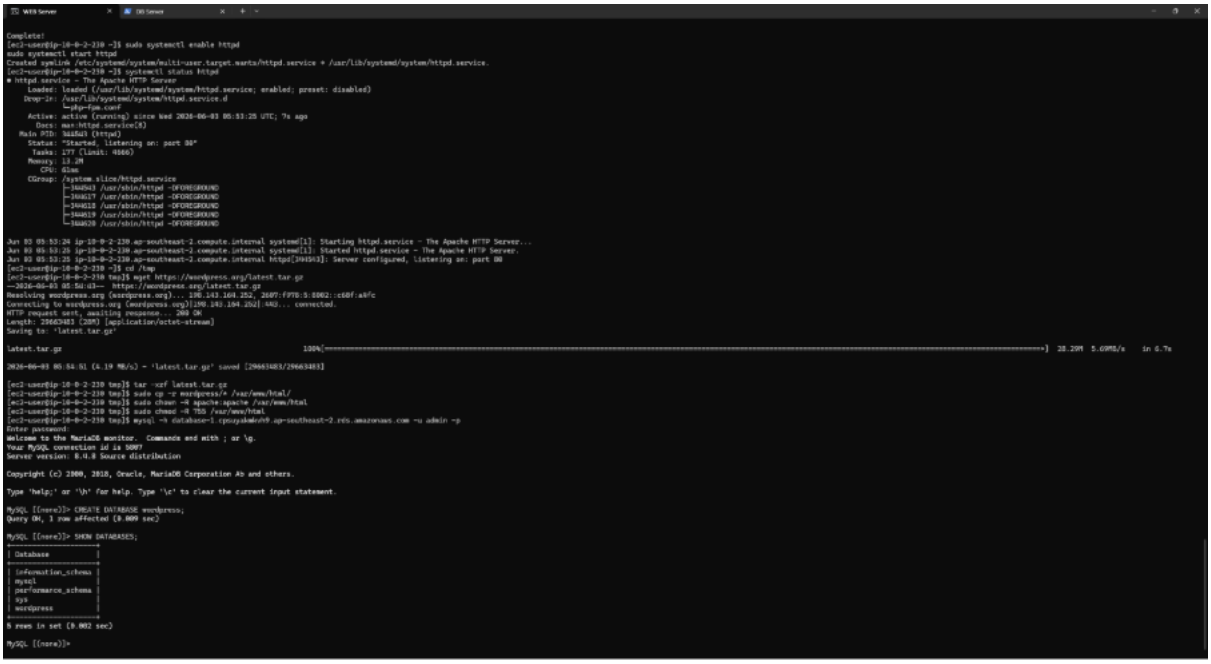


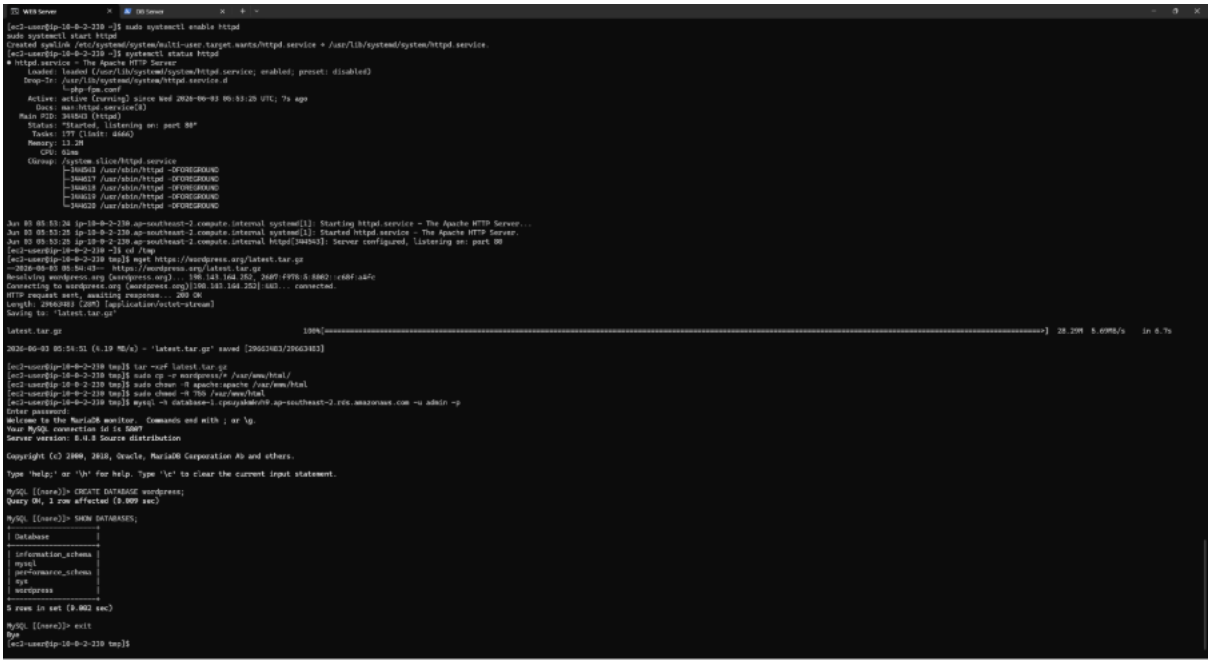


Connect DB with Webserver and create DB

Connectivity between the application server and Amazon RDS was successfully validated. The WordPress database was created and prepared for application deployment.

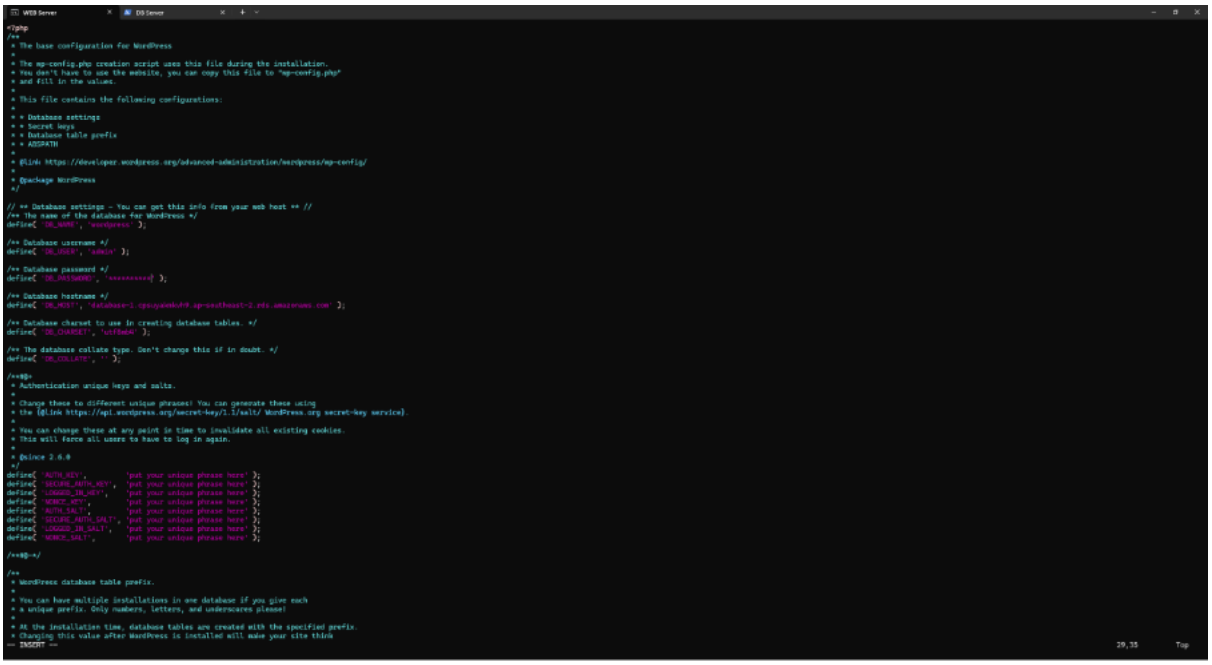


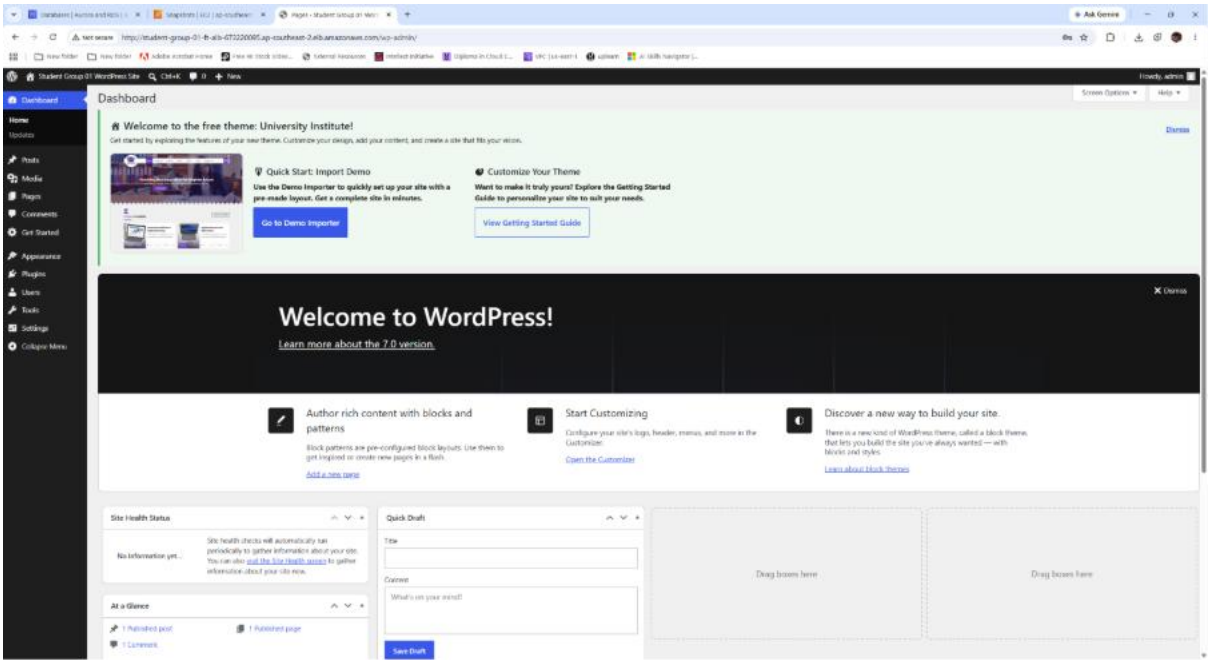

WEB Server
DB Server
Complete!
[ec2-user@ip-10-0-2-230 ~]$ sudo systemctl enable httpd
sudo systemctl start httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
[ec2-user@ip-10-0-2-230 ~]$ systemctl status httpd
● httpd.service - The Apache HTTP Server
Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)
Drop-In: /usr/lib/systemd/system/httpd.service.d
└─php-fpm.conf
Active: active (running) since Wed 2026-06-03 05:53:25 UTC; 7s ago
Docs: man:httpd.service(8)
Main PID: 3648543 (httpd)
Status: "Started, listening on: port 80"
Tasks: 177 (limit: 4666)
Memory: 13.2M
CPU: 61ms
CGroup: /system.slice/httpd.service
├─3648543 /usr/sbin/httpd -DFOREGROUND
├─3648617 /usr/sbin/httpd -DFOREGROUND
├─3648618 /usr/sbin/httpd -DFOREGROUND
├─3648619 /usr/sbin/httpd -DFOREGROUND
└─3648620 /usr/sbin/httpd -DFOREGROUND
Jun 03 05:53:24 ip-10-0-2-230.ap-southeast-2.compute.internal systemd[1]: Starting httpd.service - The Apache HTTP Server...
Jun 03 05:53:25 ip-10-0-2-230.ap-southeast-2.compute.internal systemd[1]: Started httpd.service - The Apache HTTP Server.
Jun 03 05:53:25 ip-10-0-2-230.ap-southeast-2.compute.internal httpd[3648543]: Server configured, listening on: port 80
[ec2-user@ip-10-0-2-230 ~]$ cd /tmp
[ec2-user@ip-10-0-2-230 tmp]$ wget https://wordpress.org/latest.tar.gz
--2026-06-03 05:54:03-- https://wordpress.org/latest.tar.gz
Resolving wordpress.org (wordpress.org)... 198.143.164.252, 2607:f078:5:8002::c68f:a4fc
Connecting to wordpress.org (wordpress.org)|198.143.164.252|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 29663483 (28M) [application/octet-stream]
Saving to: 'latest.tar.gz'
latest.tar.gz 100%[==========>] 28.29M 5.69MB/s in 6.7s
2026-06-03 05:54:51 (4.19 MB/s) - 'latest.tar.gz' saved [29663483/29663483]
[ec2-user@ip-10-0-2-230 tmp]$ tar -xzf latest.tar.gz
[ec2-user@ip-10-0-2-230 tmp]$ sudo cp -r wordpress/* /var/www/html/
[ec2-user@ip-10-0-2-230 tmp]$ sudo chown -R apache:apache /var/www/html
[ec2-user@ip-10-0-2-230 tmp]$ sudo chmod -R 755 /var/www/html
[ec2-user@ip-10-0-2-230 tmp]$ mysql -h database-1.cpeuyskmkvh9.ap-southeast-2.rds.amazonaws.com -u admin -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MySQL connection id is 5097
Server version: 8.4.8 Source distribution
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
MySQL [(none)]> CREATE DATABASE wordpress;
Query OK, 1 row affected (0.009 sec)
MySQL [(none)]> SHOW DATABASES;
+--------------------+
| Database |
+--------------------+
| information_schema |
| mysql |
| performance_schema |
| sys |
| wordpress |
+--------------------+
5 rows in set (0.002 sec)
MySQL [(none)]>


WEB Server
DB Server
[ec2-user@ip-10-0-2-230 ~]$ sudo systemctl enable httpd
sudo systemctl start httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
[ec2-user@ip-10-0-2-230 ~]$ systemctl status httpd
● httpd.service - The Apache HTTP Server
Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)
Drop-In: /usr/lib/systemd/system/httpd.service.d
└─php-fpm.conf
Active: active (running) since Wed 2026-06-03 05:53:25 UTC; 7s ago
Docs: man:httpd.service(8)
Main PID: 3648543 (httpd)
Status: "Started, listening on: port 80"
Tasks: 177 (limit: 4666)
Memory: 13.2M
CPU: 61ms
CGroup: /system.slice/httpd.service
├─3648543 /usr/sbin/httpd -DFOREGROUND
├─3648617 /usr/sbin/httpd -DFOREGROUND
├─3648618 /usr/sbin/httpd -DFOREGROUND
├─3648619 /usr/sbin/httpd -DFOREGROUND
└─3648620 /usr/sbin/httpd -DFOREGROUND
Jun 03 05:53:24 ip-10-0-2-230.ap-southeast-2.compute.internal systemd[1]: Starting httpd.service - The Apache HTTP Server...
Jun 03 05:53:25 ip-10-0-2-230.ap-southeast-2.compute.internal systemd[1]: Started httpd.service - The Apache HTTP Server.
Jun 03 05:53:25 ip-10-0-2-230.ap-southeast-2.compute.internal httpd[3648543]: Server configured, listening on: port 80
[ec2-user@ip-10-0-2-230 ~]$ cd /tmp
[ec2-user@ip-10-0-2-230 tmp]$ wget https://wordpress.org/latest.tar.gz
--2026-06-03 05:54:43-- https://wordpress.org/latest.tar.gz
Resolving wordpress.org (wordpress.org)... 198.143.164.252, 2607:f078:5:8002::c68f:a4fc
Connecting to wordpress.org (wordpress.org)|198.143.164.252|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 29663483 (28M) [application/octet-stream]
Saving to: 'latest.tar.gz'
latest.tar.gz 100%[==========>] 28.29M 5.69MB/s in 6.7s
2026-06-03 05:54:51 (4.19 MB/s) - 'latest.tar.gz' saved [29663483/29663483]
[ec2-user@ip-10-0-2-230 tmp]$ tar -xzf latest.tar.gz
[ec2-user@ip-10-0-2-230 tmp]$ sudo cp -r wordpress/* /var/www/html/
[ec2-user@ip-10-0-2-230 tmp]$ sudo chown -R apache:apache /var/www/html
[ec2-user@ip-10-0-2-230 tmp]$ sudo chmod -R 755 /var/www/html
[ec2-user@ip-10-0-2-230 tmp]$ mysql -h database-1.cpeuyskmkvh9.ap-southeast-2.rds.amazonaws.com -u admin -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MySQL connection id is 5097
Server version: 8.4.8 Source distribution
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
MySQL [(none)]> CREATE DATABASE wordpress;
Query OK, 1 row affected (0.009 sec)
MySQL [(none)]> SHOW DATABASES;
+--------------------+
| Database |
+--------------------+
| information_schema |
| mysql |
| performance_schema |
| sys |
| wordpress |
+--------------------+
5 rows in set (0.002 sec)
MySQL [(none)]> exit
Bye
[ec2-user@ip-10-0-2-230 tmp]$

Configure WordPress

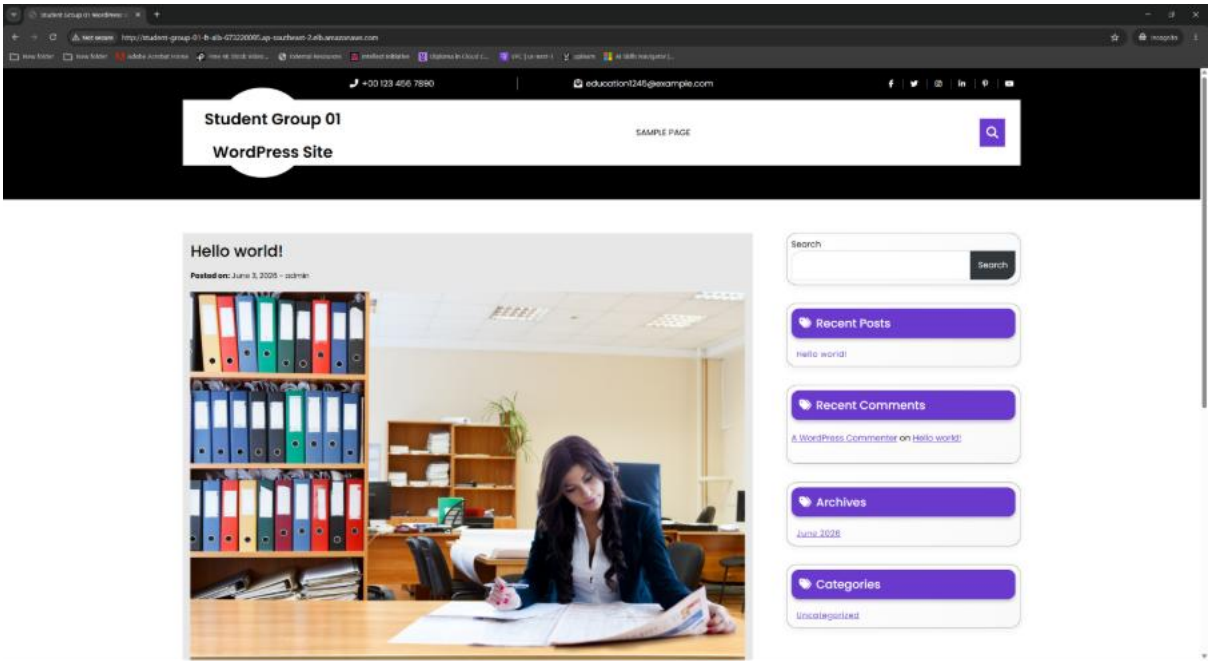
WordPress was configured to use the Amazon RDS database instance. Application setup was completed successfully, and the website became accessible through the configured Load Balancer and domain name.





Verify the website with the Load Balancer DNS

The WordPress website was tested using the Application Load Balancer DNS name to confirm that external traffic was correctly routed to the web server. The successful page load verifies that the ALB listener, target group, health checks, and EC2 web server configuration were functioning correctly.

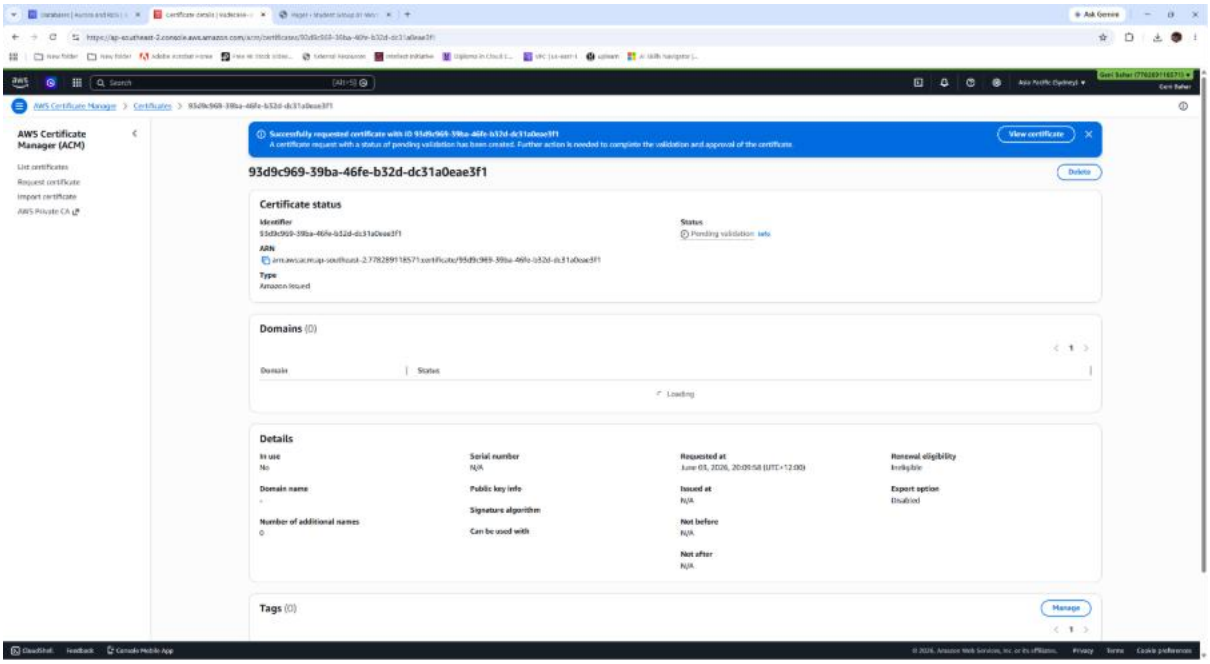


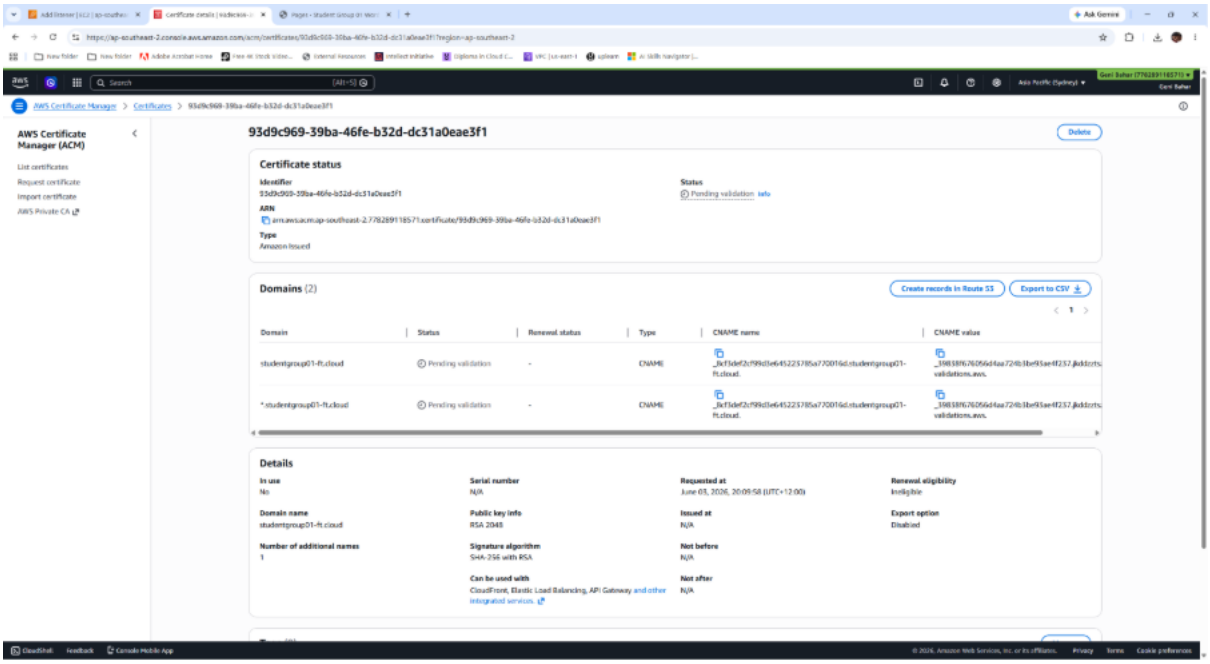
HTTPS Implementation

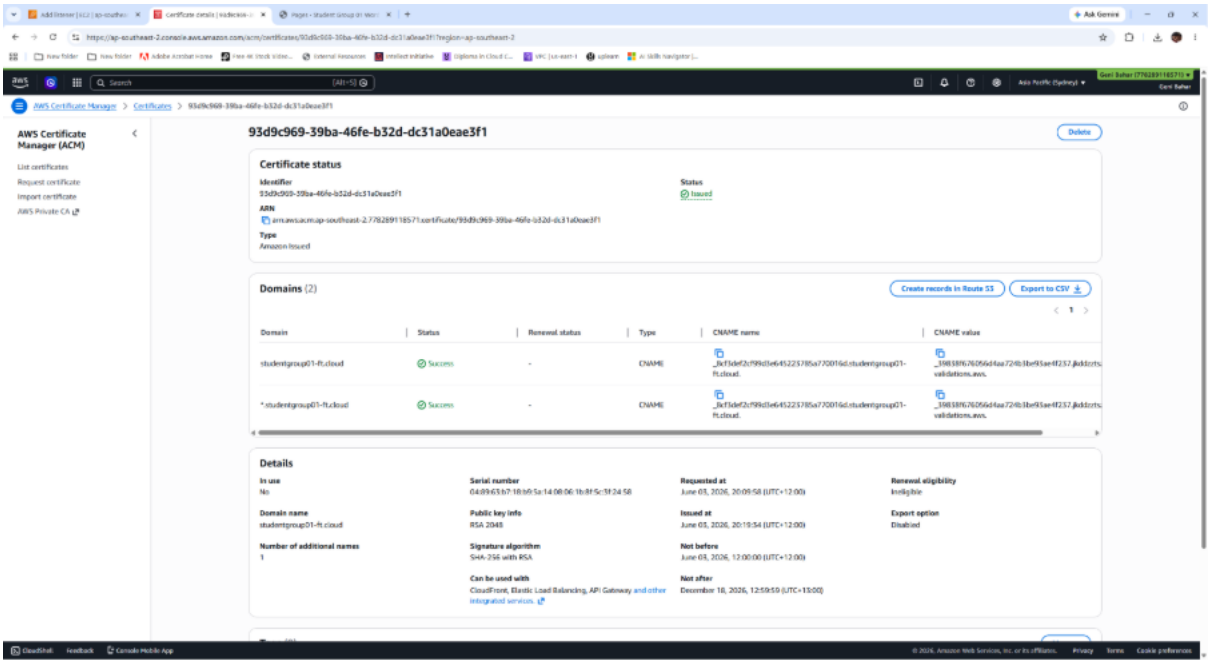
HTTPS was implemented using AWS Certificate Manager and the Load Balancer listener configuration. This protects data in transit between users and the WordPress application. HTTP access should be redirected to HTTPS where possible to avoid unencrypted traffic.

ACM Certificate Creation

An SSL/TLS certificate was issued through AWS Certificate Manager and attached to the Application Load Balancer. This enables encrypted HTTPS communication between users and the hosted application.

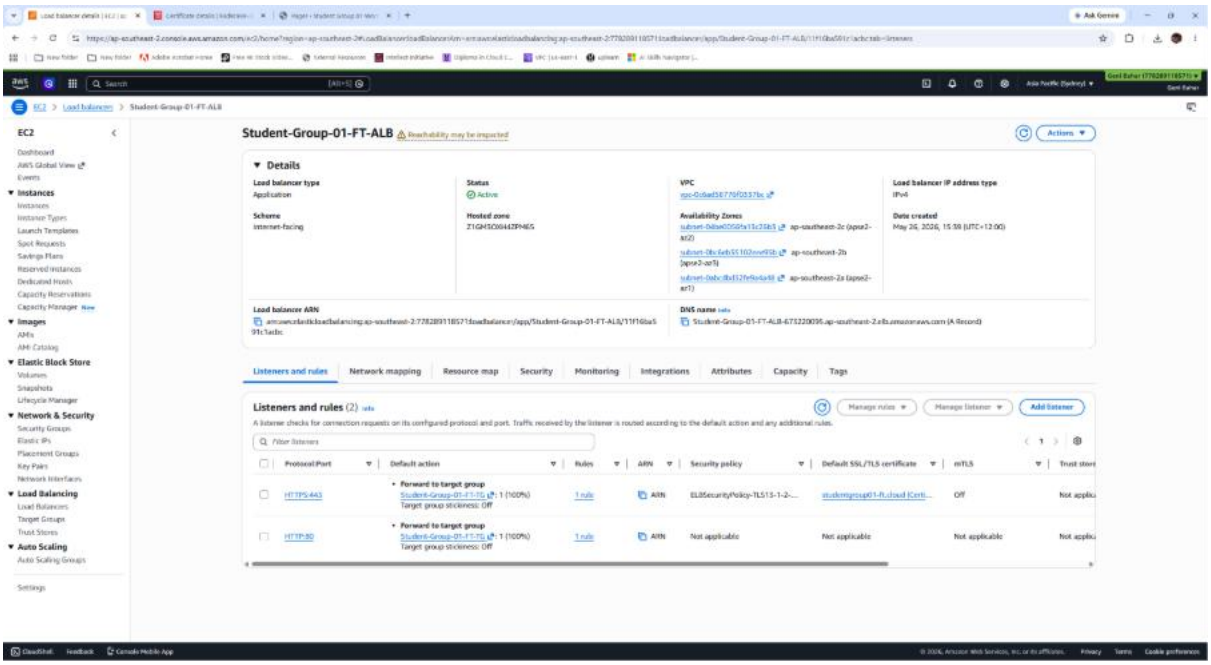


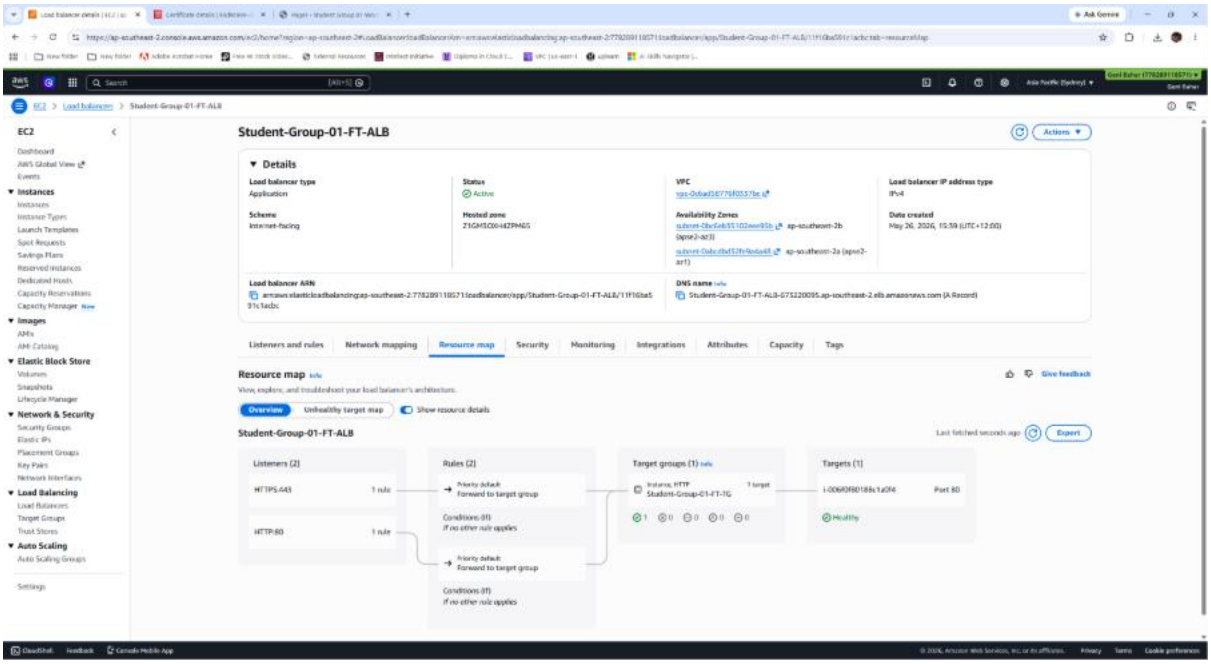




Apply Certificate to LoadBalancer

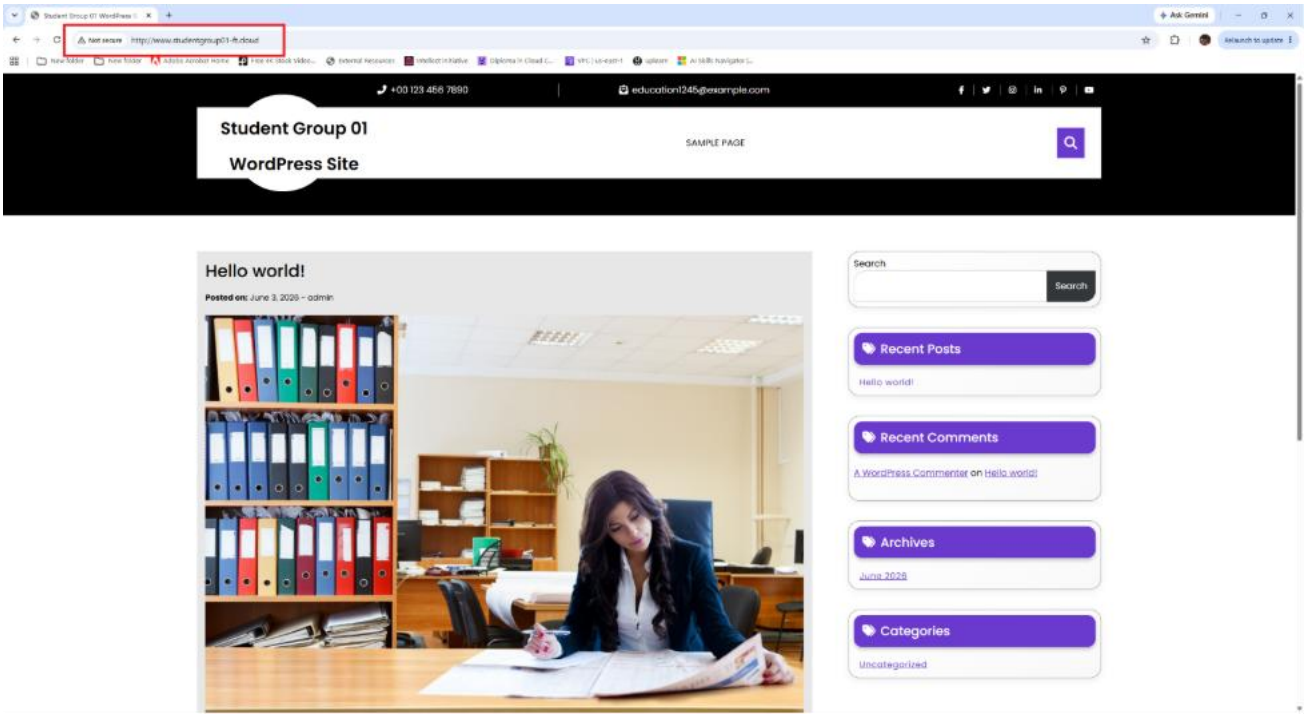
The ACM SSL/TLS certificate was attached to the HTTPS listener on the Application Load Balancer. This allows the Load Balancer to terminate HTTPS traffic securely and forward requests to the WordPress application, ensuring encrypted communication for users accessing the website.





HTTPS Check

The website was successfully accessed using HTTPS through the public domain name. The SSL/TLS certificate was issued through AWS Certificate Manager and attached to the Application Load Balancer, ensuring encrypted communication between users and the hosted application.



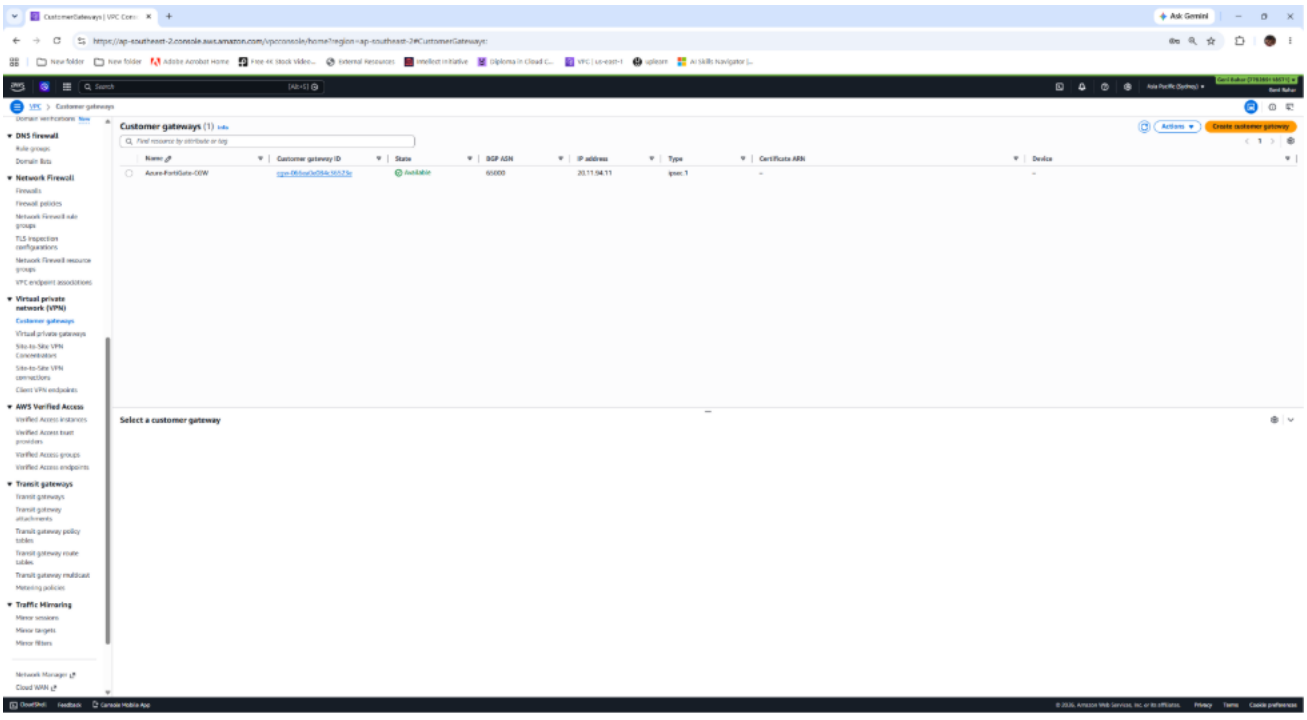
VPN Connection for Azure FortiGate Firewall

FortiGate was included as the dedicated security control point for inter-cloud traffic. The purpose of this component is to inspect, restrict, and manage traffic between AWS and Azure, while supporting secure VPN/routing configuration. FortiGate firewall policies are used to define which traffic is allowed between environments and to reduce the risk of uncontrolled cross-cloud communication.

FortiGate control	**Purpose**
VPN/routing configuration	Enables controlled connectivity between AWS and Azure components
Firewall policies	Allows only approved inter-cloud traffic
Security inspection	Supports visibility and filtering of traffic flows
Logging	Provides evidence for troubleshooting and incident analysis
Segmentation	Reduces the risk of one cloud environment exposing another

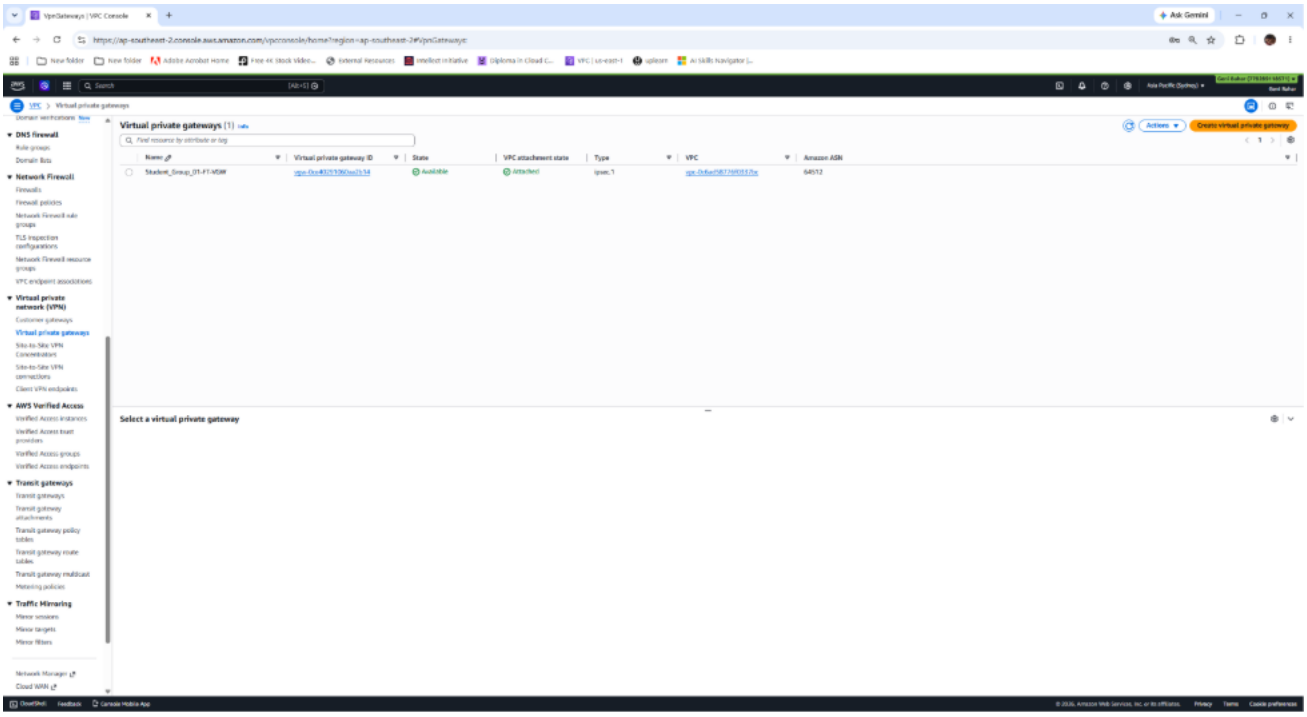
Customer Gateway Creation

An AWS Customer Gateway was created to represent the external FortiGate firewall device located within the Azure environment. The Customer Gateway stores the public IP address of the remote firewall and acts as the AWS endpoint definition for establishing VPN connectivity.



Virtual Private Gateway Creation

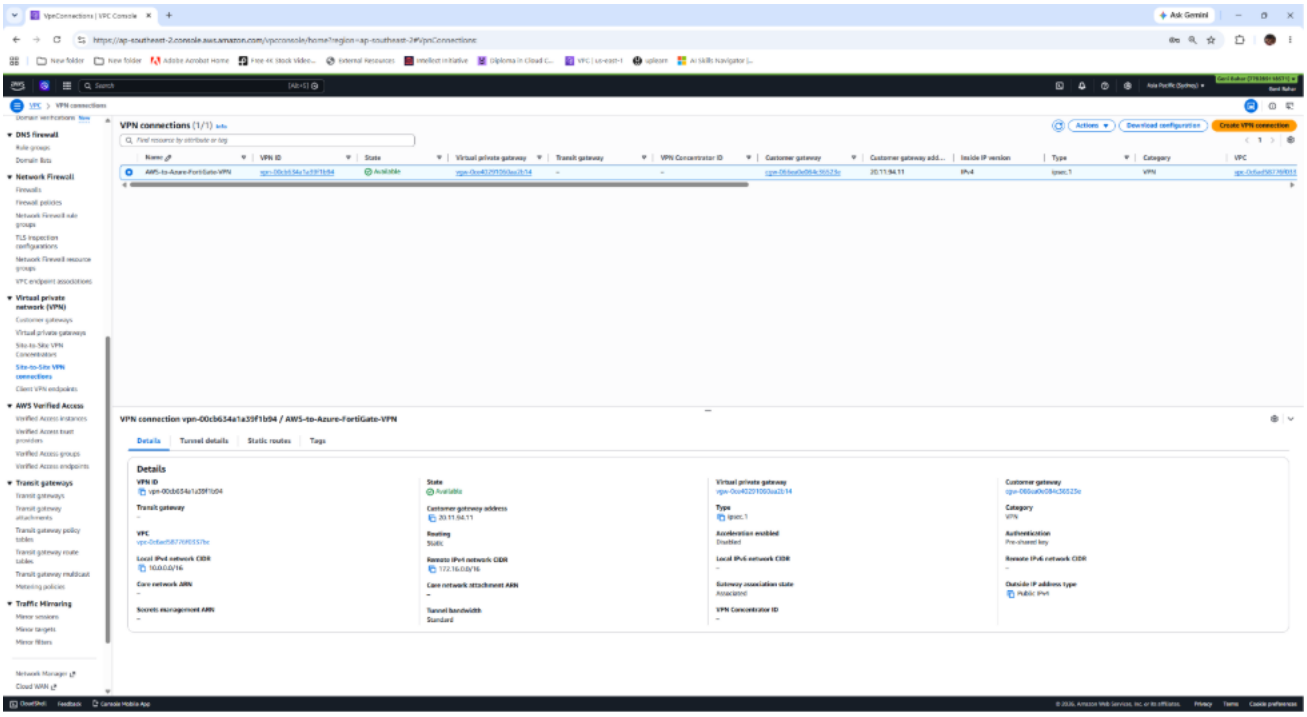
A Virtual Private Gateway (VGW) was created and attached to the AWS VPC to provide VPN capabilities for the environment. The VGW acts as the AWS-side VPN endpoint and enables encrypted communication between the VPC and remote networks.



Site-to-Site VPN Connection Creation

A Site-to-Site VPN connection was configured to establish secure communication between AWS and the external FortiGate environment.

The VPN connection uses IPsec encryption and provides secure transport for traffic exchanged between the AWS VPC and remote networks.

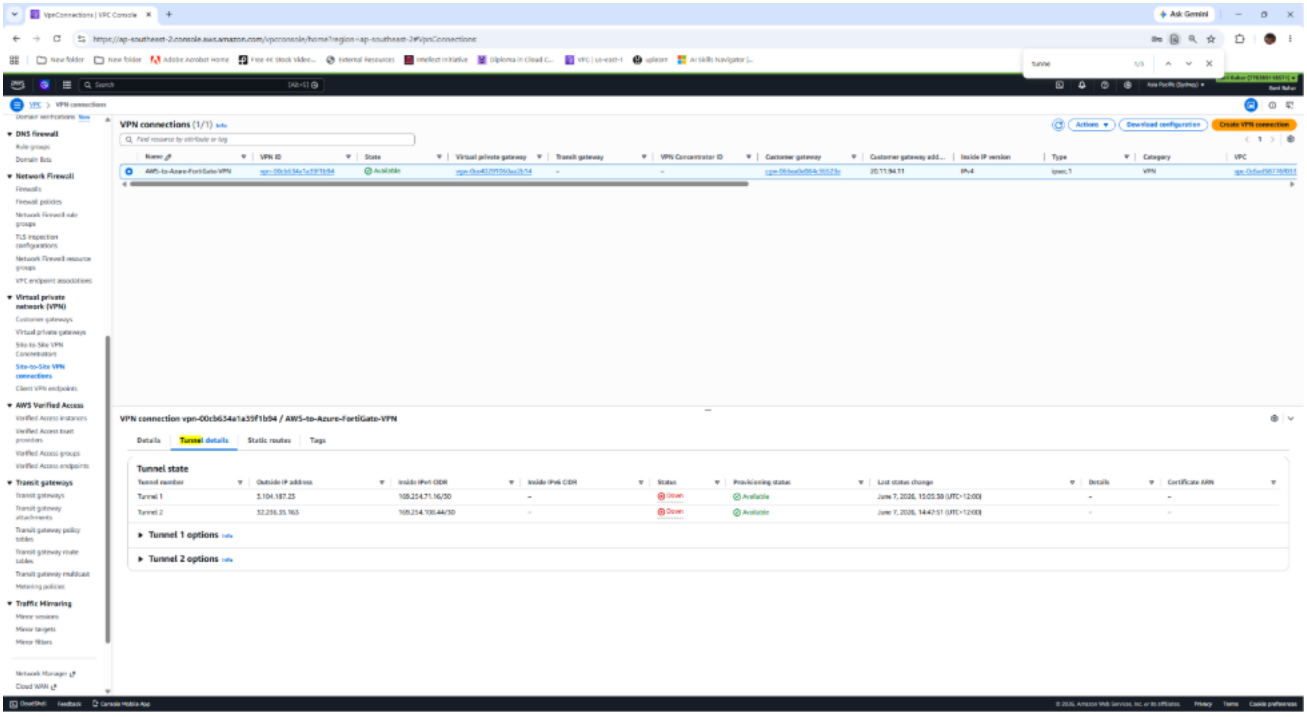


VPN Tunnel Configuration

AWS automatically generated two IPsec tunnels to provide redundancy and high availability for the Site-to-Site VPN connection.

The tunnel endpoints were successfully provisioned and made available for integration with the external FortiGate firewall environment. Since the remote FortiGate device had not yet been connected, both tunnels remained in a waiting state.

The screenshot confirms that two VPN tunnels were created successfully. The tunnel provisioning status shows "Available", indicating that the AWS VPN infrastructure was successfully deployed and ready for connection by the external firewall device.

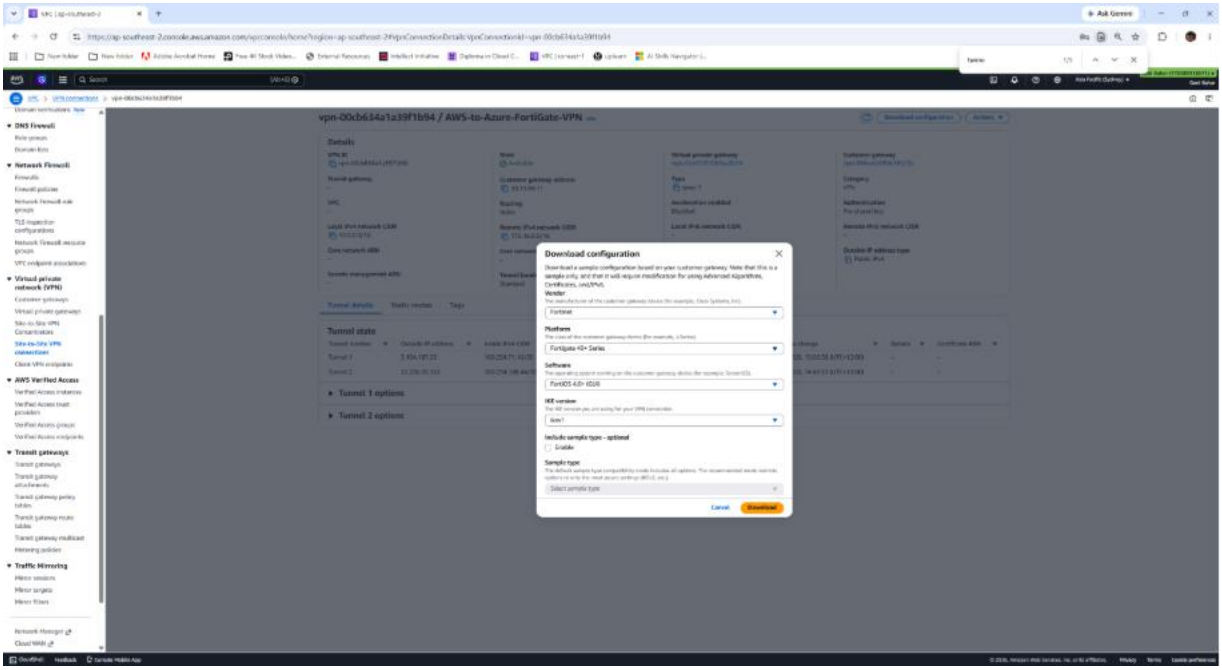


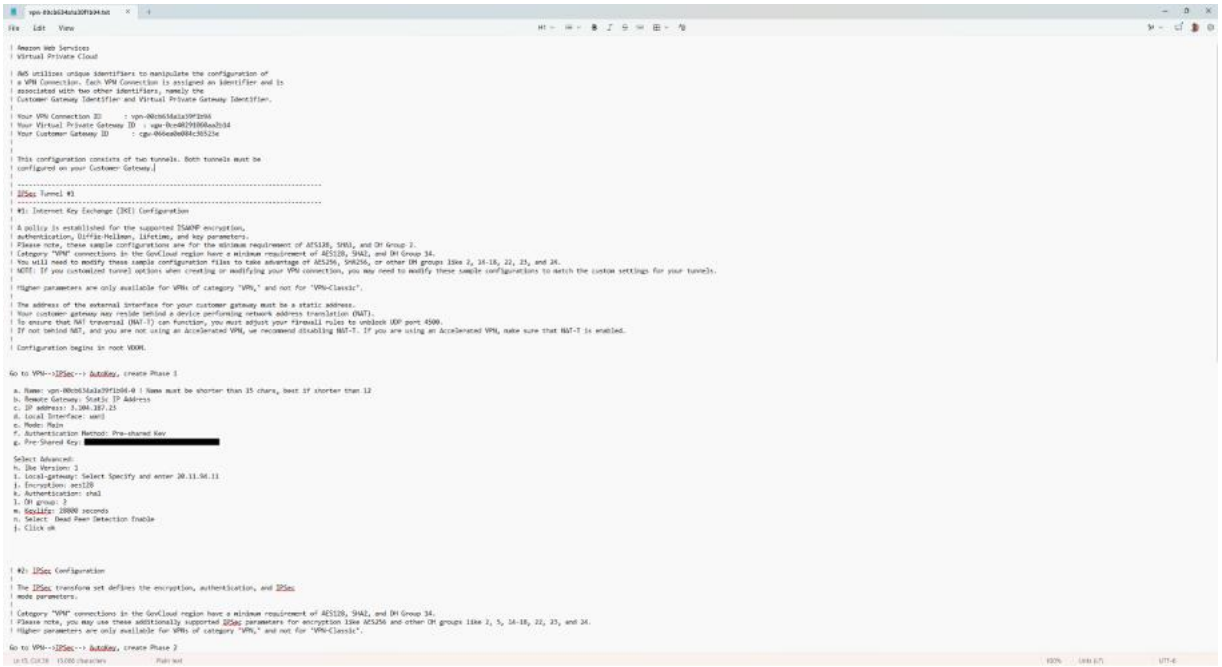
VPN Configuration File Generation

AWS provides vendor-specific VPN configuration files to simplify integration with external firewall devices. A Fortinet FortiGate configuration package was downloaded from the Site-to-Site VPN connection to support future integration with the Azure FortiGate firewall environment.

The generated configuration file contains the tunnel endpoints, Phase 1 and Phase 2 parameters, and authentication settings required to establish secure IPsec connectivity.

The downloaded configuration file provides all parameters required for configuring the remote FortiGate device and simplifies the deployment of the Site-to-Site VPN connection.





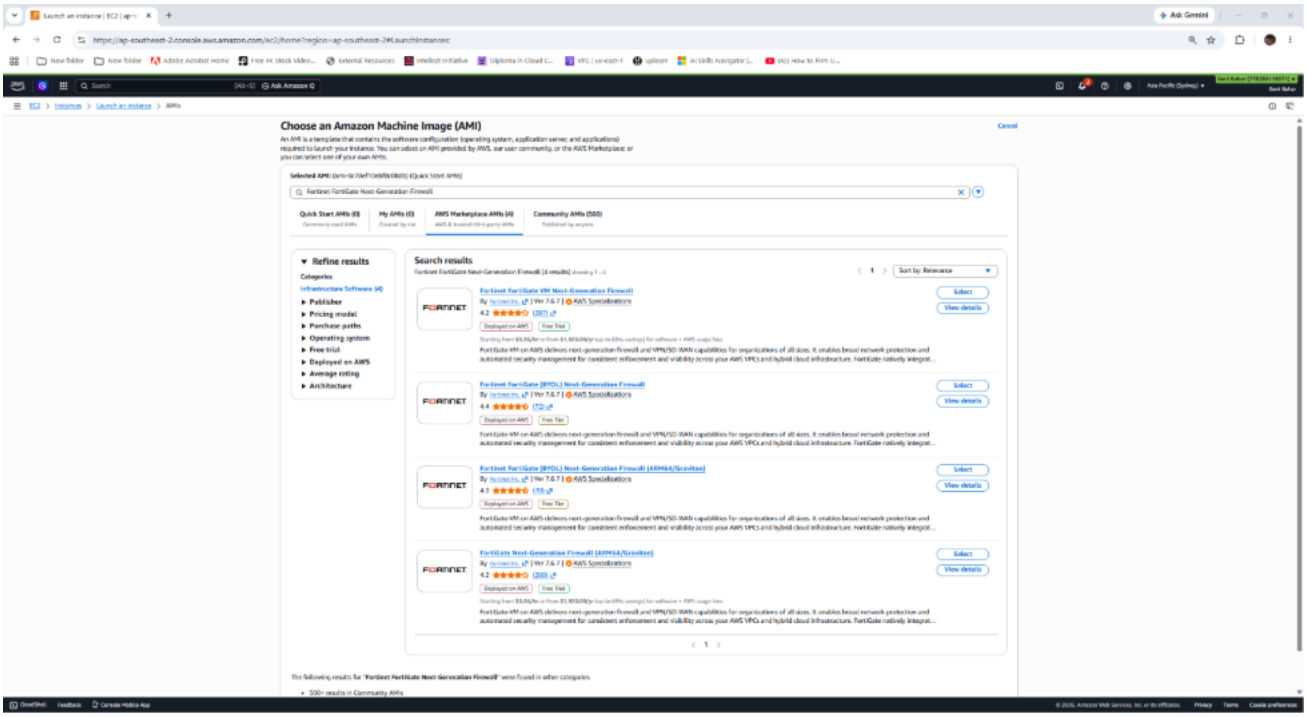
FortiGate Firewall Implementation

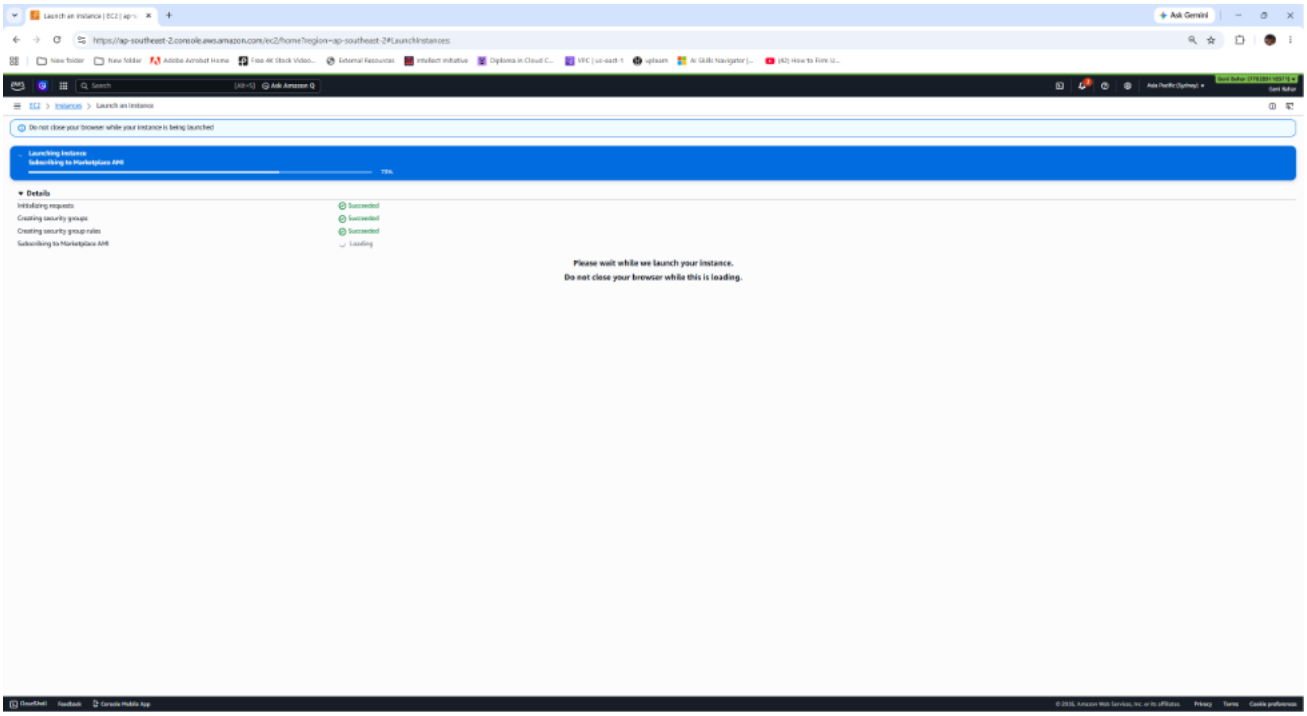
FortiGate Next-Generation Firewall was deployed in the AWS environment to demonstrate firewall-based security control for the multi-cloud architecture. The FortiGate instance was configured with public and internal network interfaces to support traffic inspection, routing, and VPN tunnel configuration.

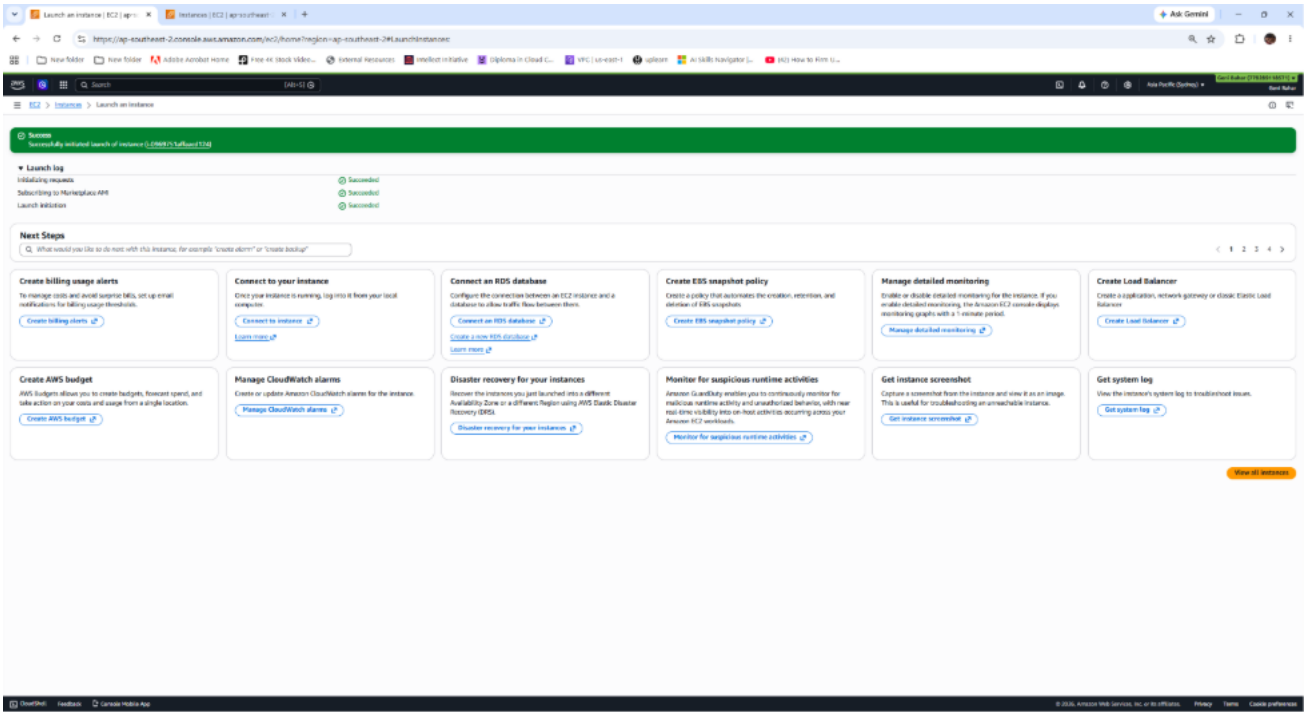
This implementation provides evidence of Fortinet security integration by demonstrating interface configuration, address objects, firewall policies, static routing, and IPsec VPN tunnel preparation.

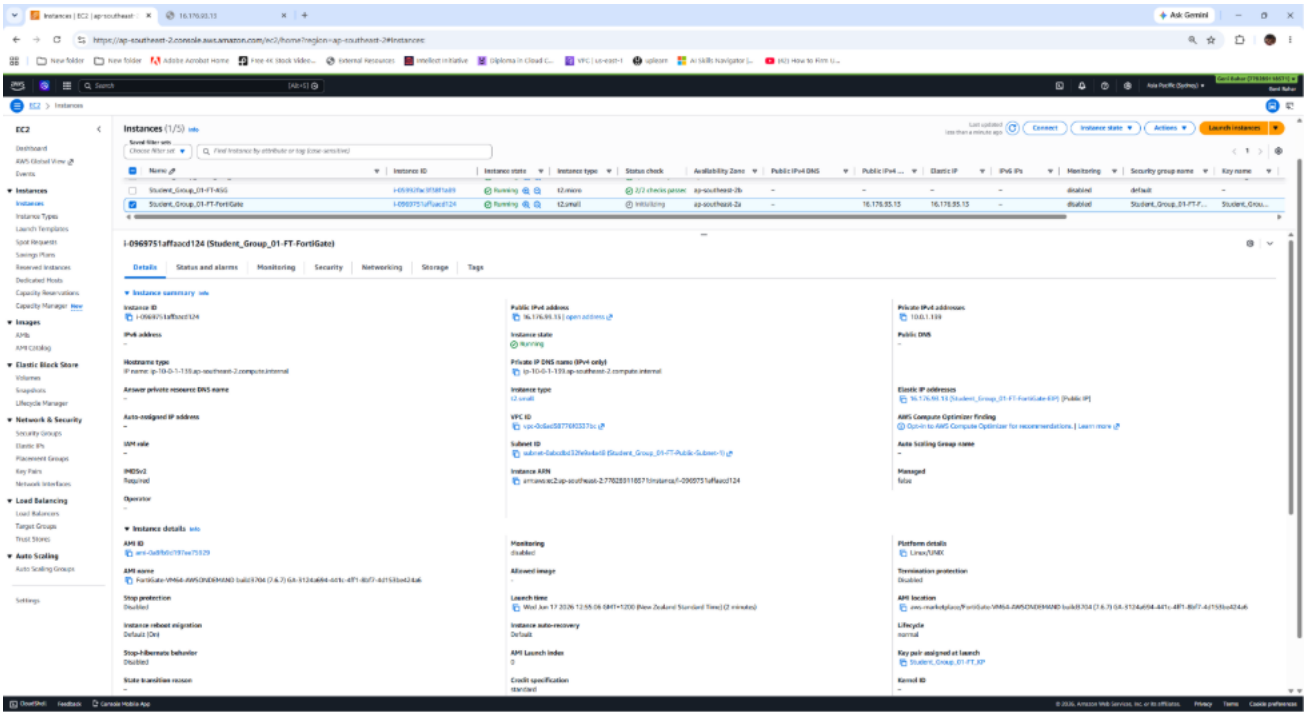
FortiGate EC2 Instance Deployment

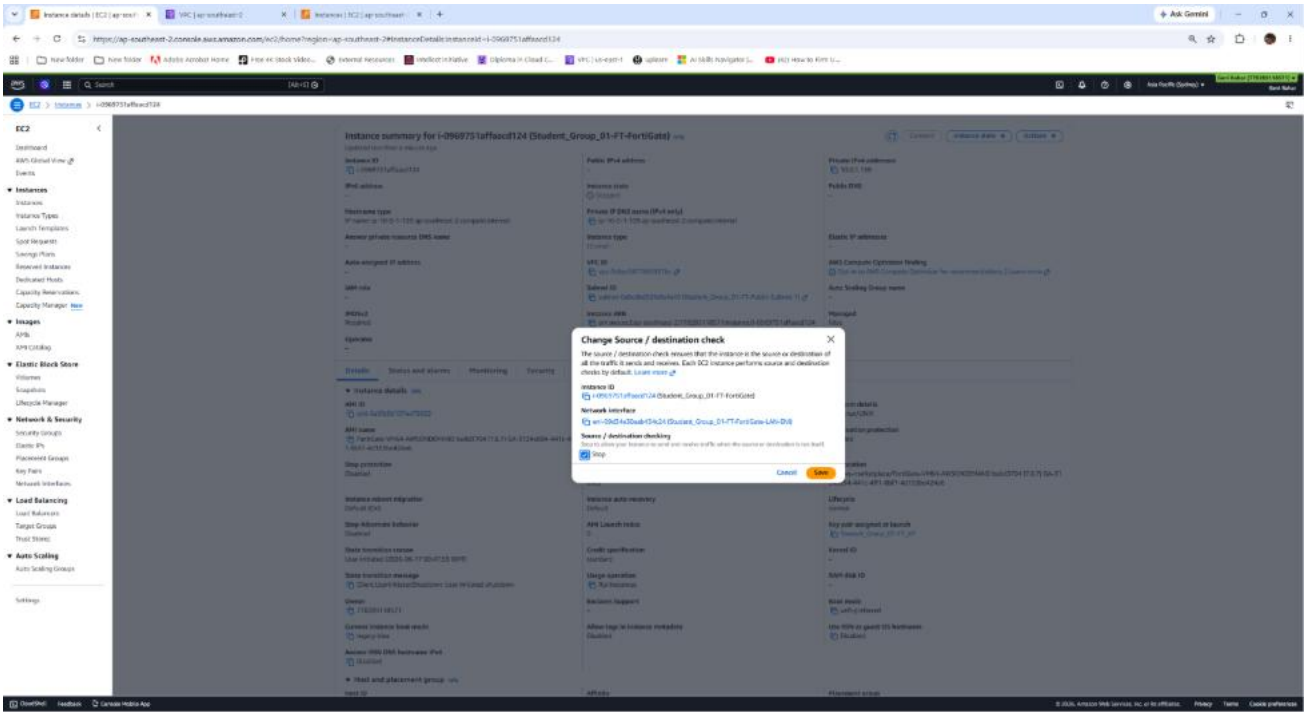
A FortiGate VM instance was deployed in AWS using the Fortinet marketplace image. The instance was placed inside the project VPC and assigned a public IP address to allow administrative access to the FortiGate management console.

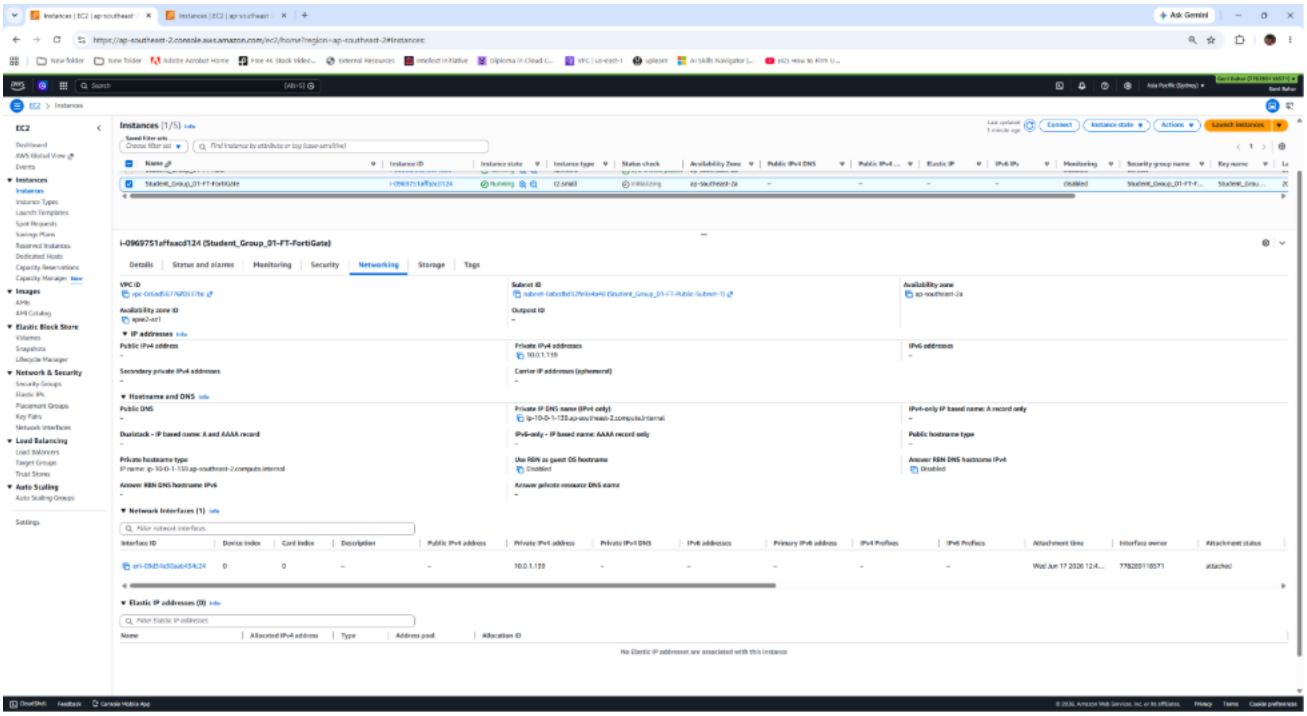


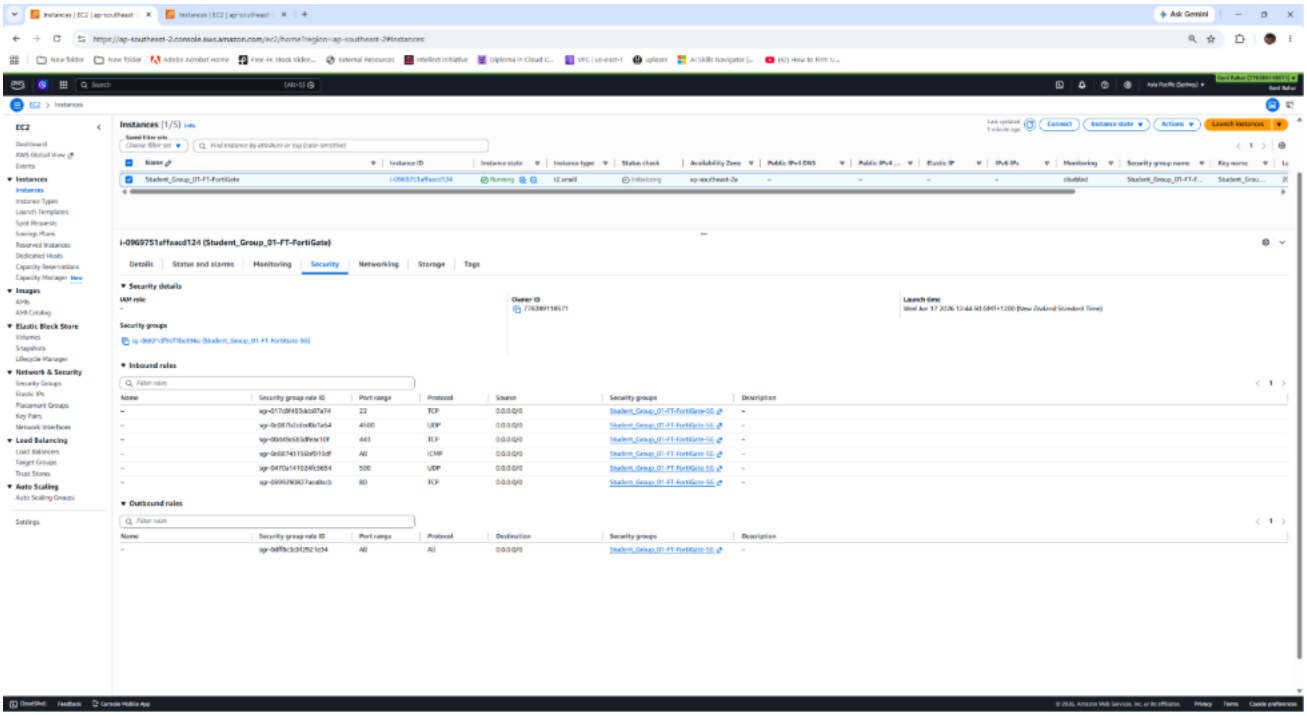

Launching instance
Subscribing to Marketplace AMI
Details
Initializing requests
Succeeded
Creating security groups
Succeeded
Creating security group rules
Succeeded
Subscribing to Marketplace AMI
Loading
Please wait while we launch your instance.
Do not close your browser while this is loading.

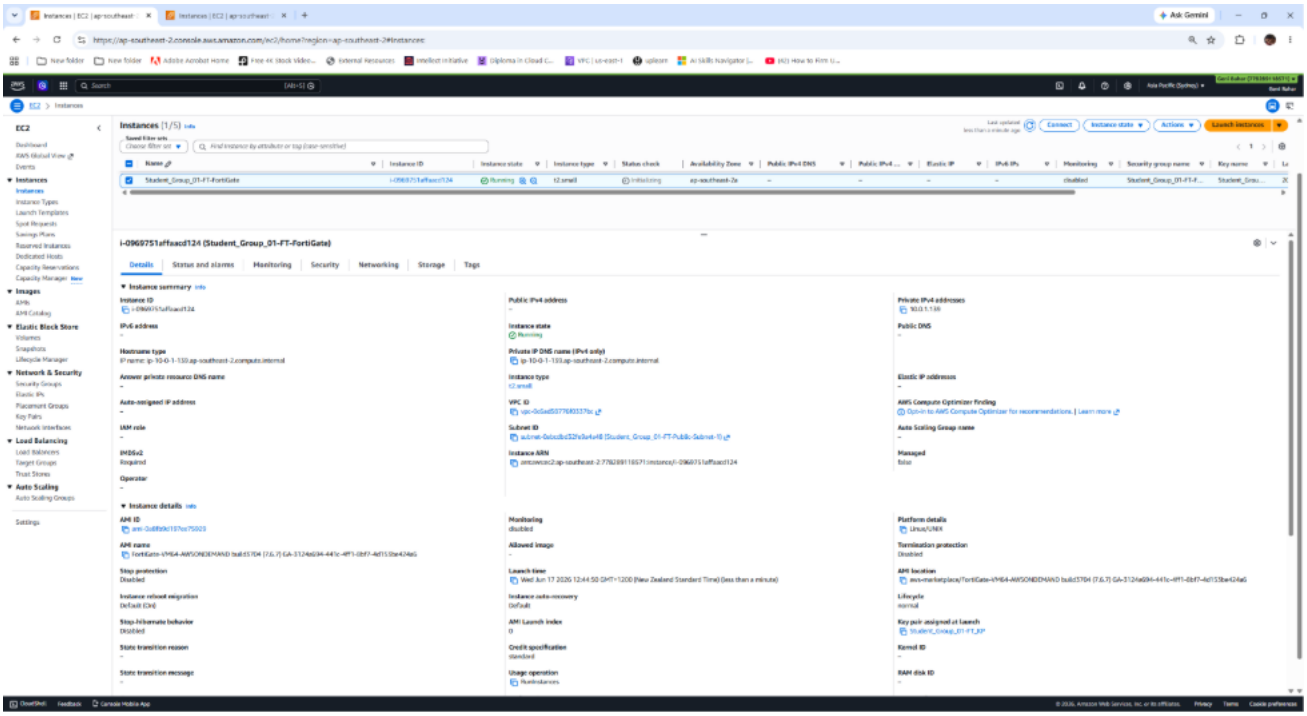

Success
Launch log
Initializing requests
Succeeded
Subscribing to Marketplace AMI
Succeeded
Launch initiation
Succeeded
Next Steps
Create billing usage alerts
Connect to your instance
Connect an RDS database
Create EBS snapshot policy
Manage detailed monitoring
Create Load Balancer
Create AWS budget
Manage CloudWatch alarms
Disaster recovery for your instances
Monitor for suspicious runtime activities
Get instance screenshot
Get system log

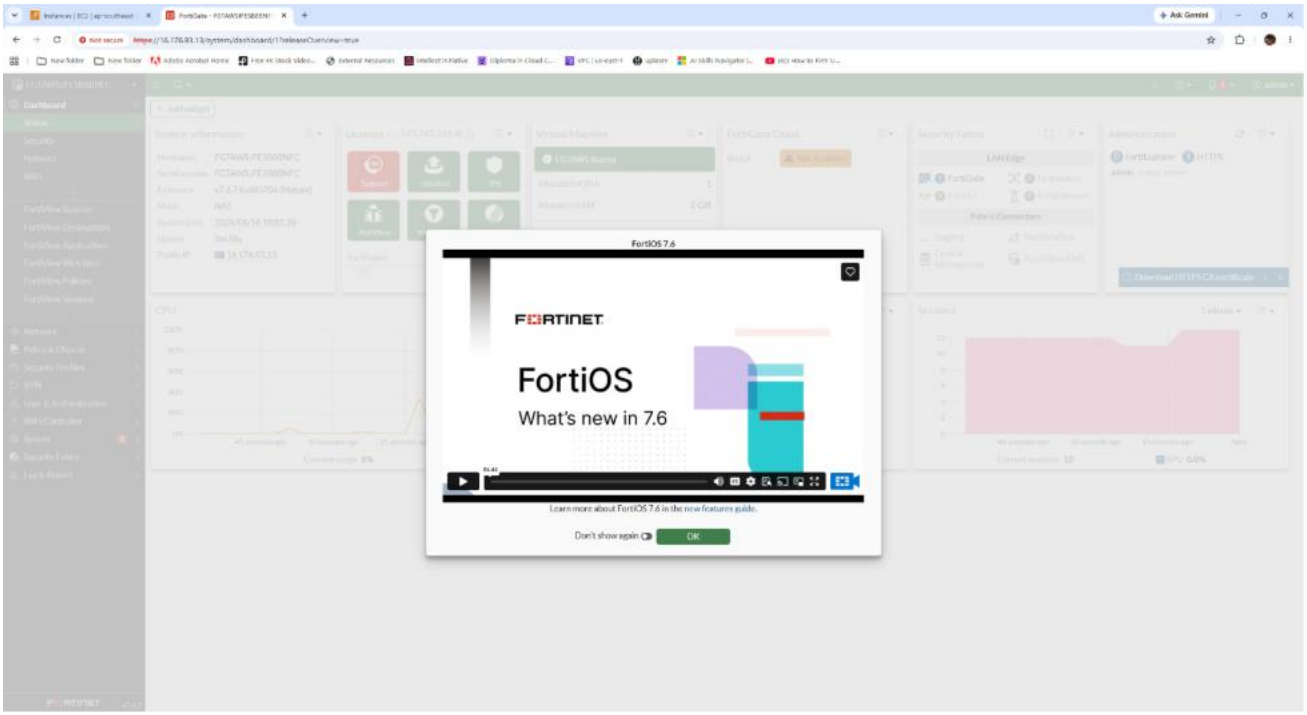


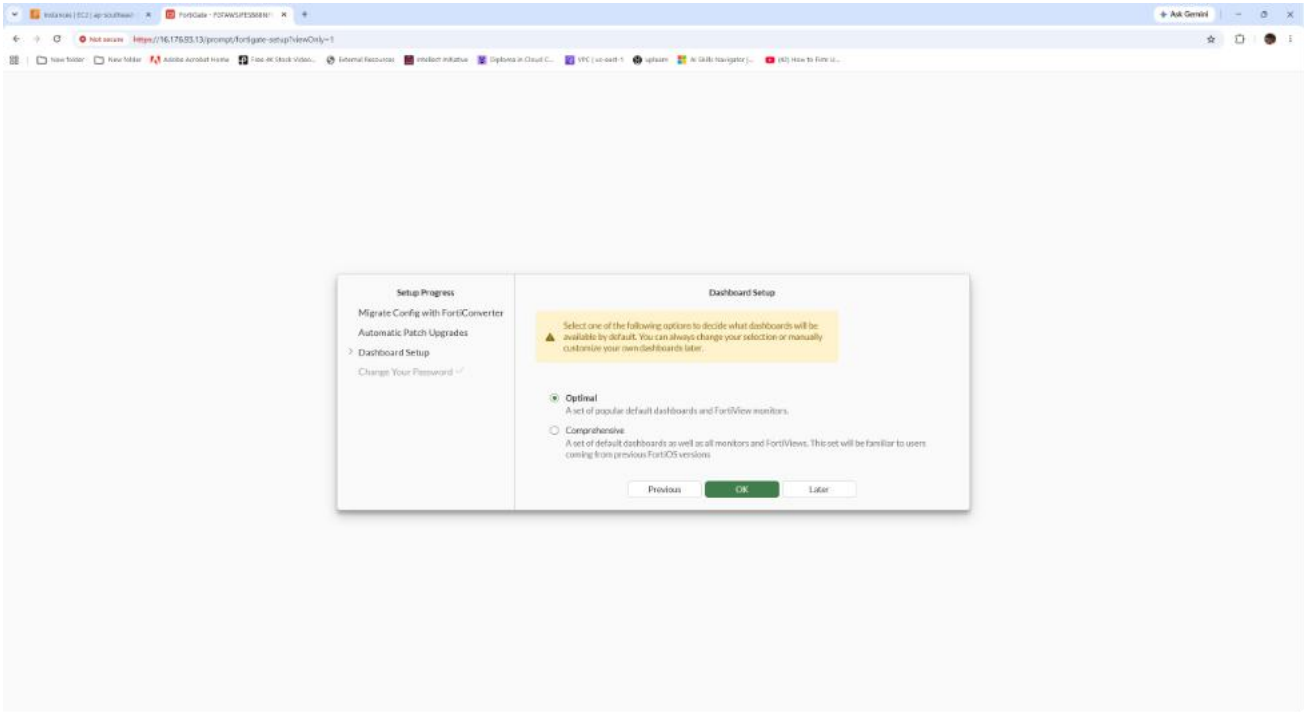

Change Source / destination check

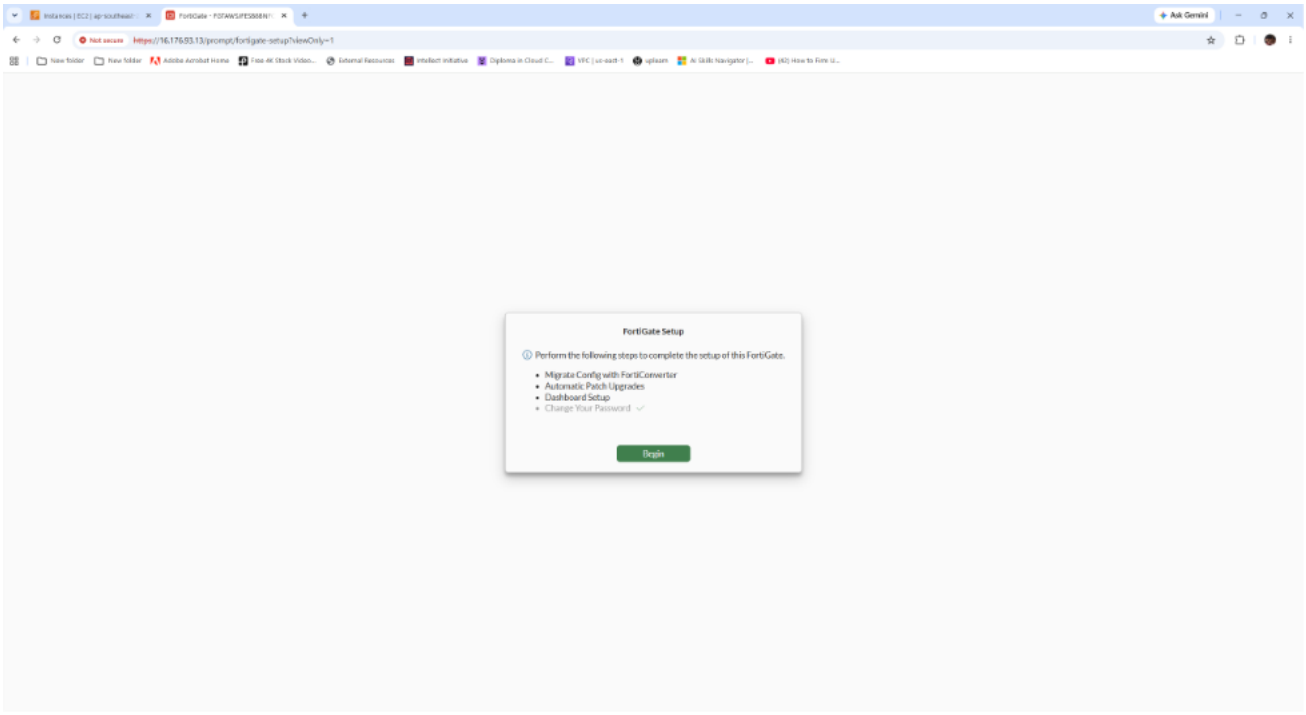


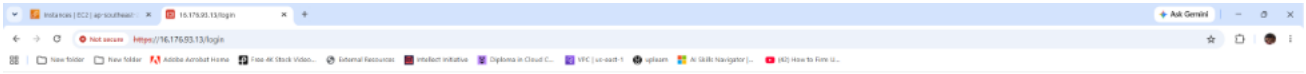


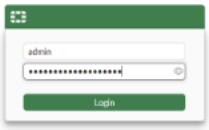


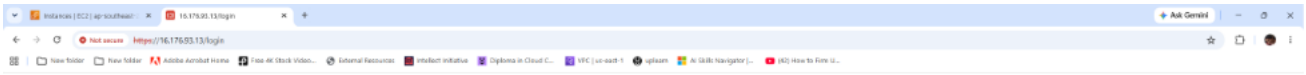

FortiOS 7.6
FORTINET
FortiOS
What's new in 7.6
Learn more about FortiOS 7.6 in the new features guide.
Don't show again
OK

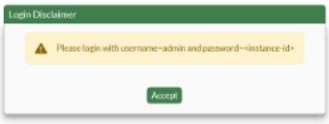

Setup Progress
Migrate Config with FortiConverter
Automatic Patch Upgrades
Dashboard Setup
Change Your Password
Dashboard Setup
Select one of the following options to decide what dashboards will be available by default. You can always change your selection or manually customize your own dashboards later.
Optimal
A set of popular default dashboards and FortiView monitors.
Comprehensive
A set of default dashboards as well as all monitors and FortiViews. This set will be familiar to users coming from previous FortiOS versions
Previous
OK
Later

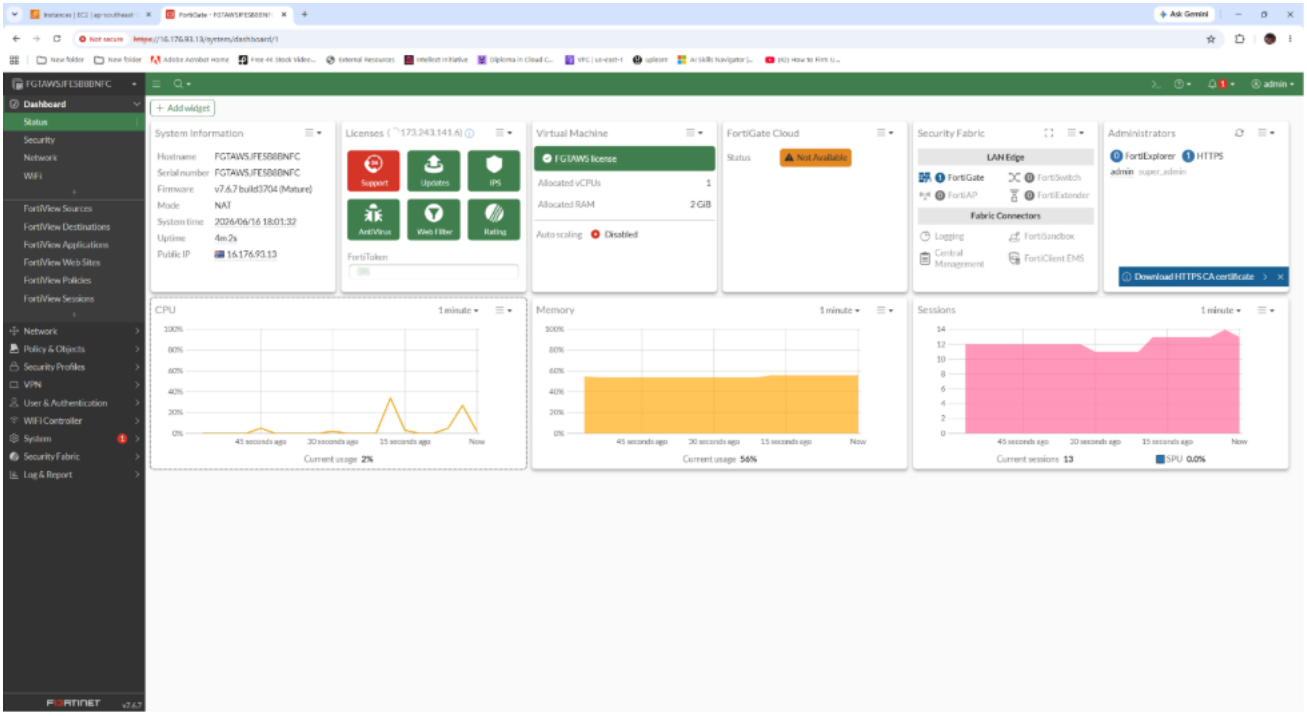

FortiGate Setup
Perform the following steps to complete the setup of this FortiGate.
Migrate Config with FortiConverter
Automatic Patch Upgrades
Dashboard Setup
Change Your Password
Begin




admin
Login

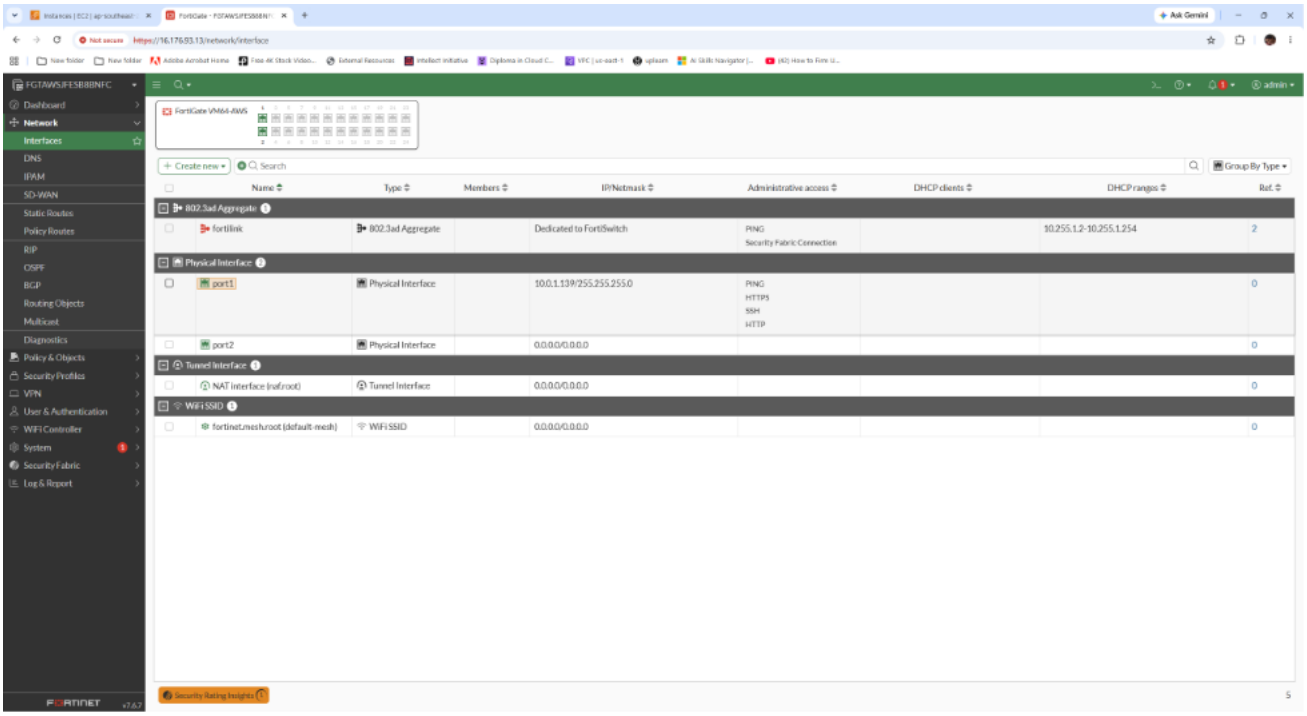


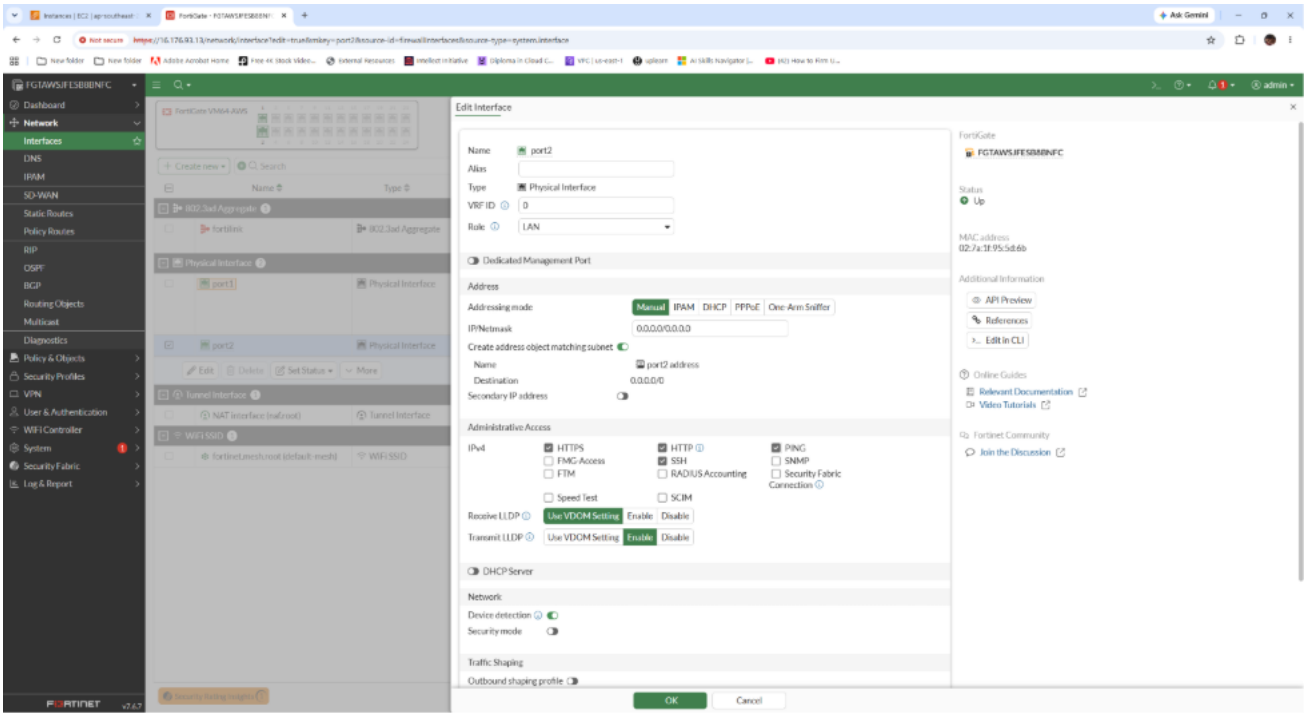

Login Disclaimer
Please login with username=admin and password=<instance id>
Accept

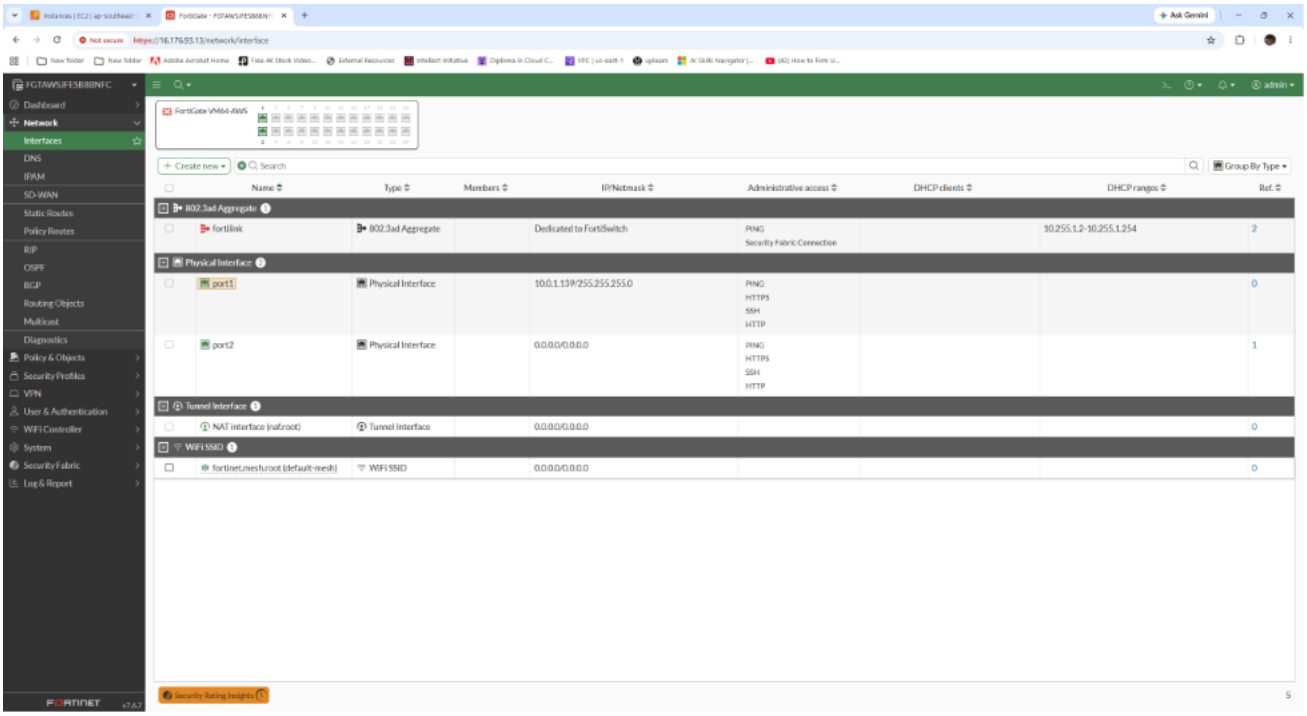

FGTAWSJFESBBNFC
Dashboard
Status
Security
Network
WiFi
FortiView Sources
FortiView Destinations
FortiView Applications
FortiView Web Sites
FortiView Policies
FortiView Sessions
Network
Policy & Objects
Security Profiles
VPN
User & Authentication
WiFi Controller
System
Security Fabric
Log & Report
Add widget
System Information
Licenses
Virtual Machine
FGTAWS license
FortiGate Cloud
Status
Not Available
Security Fabric
LAN Edge
Administrators
CPU
Memory
Sessions

FortiGate Network Interfaces

The FortiGate firewall was configured with two physical interfaces. Port1 was used as the public-facing interface, while Port2 was configured as the internal interface for private network communication. This separation allows FortiGate to inspect and control traffic between external and internal networks.

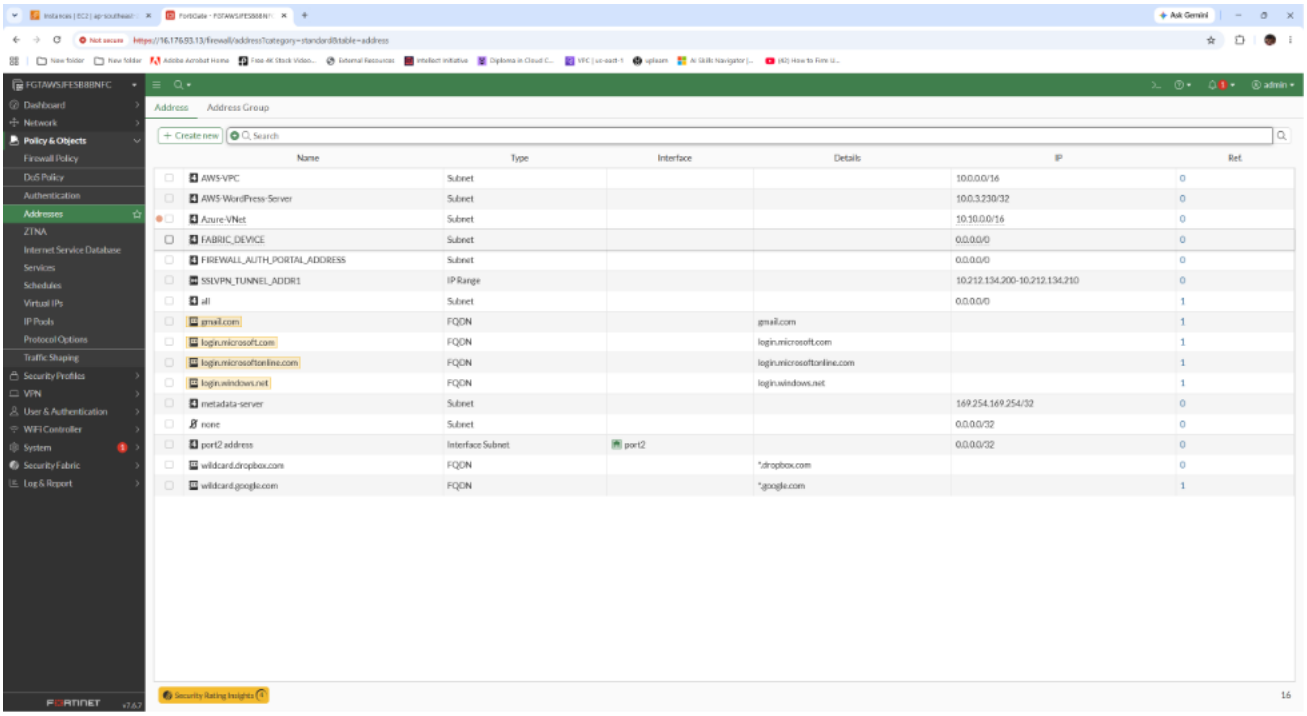






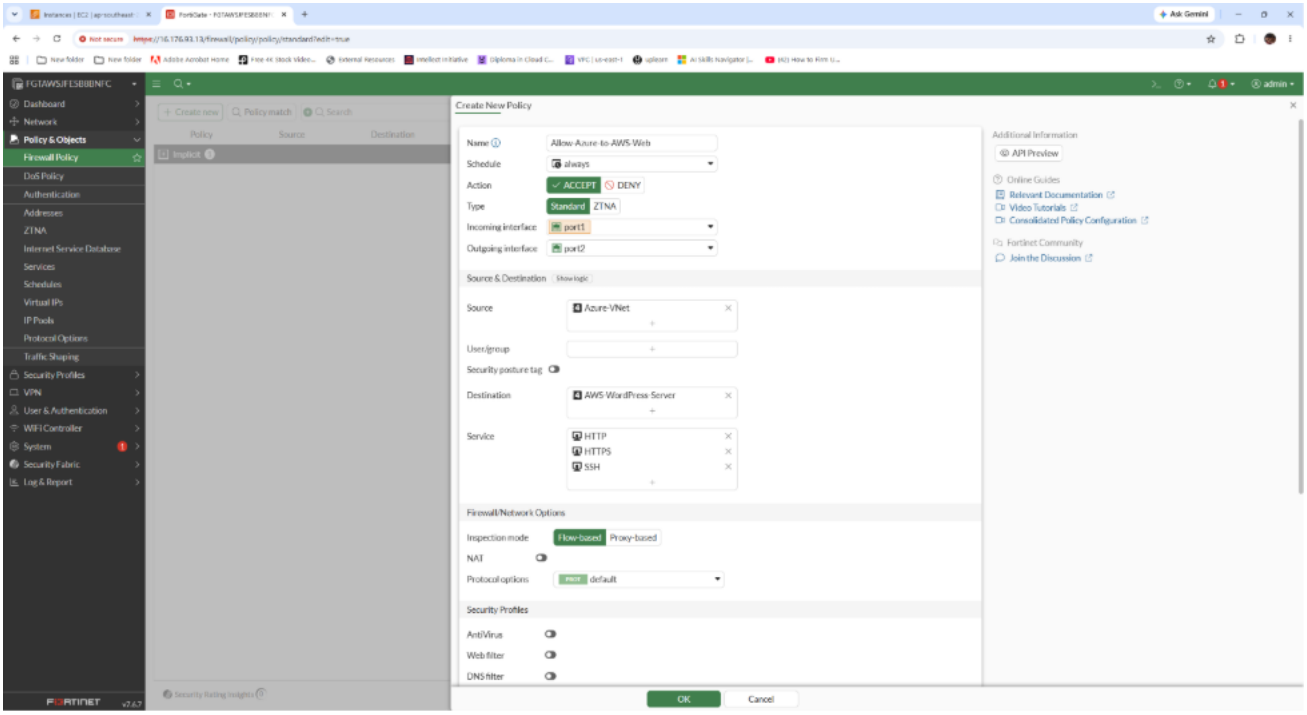
FortiGate Address Objects

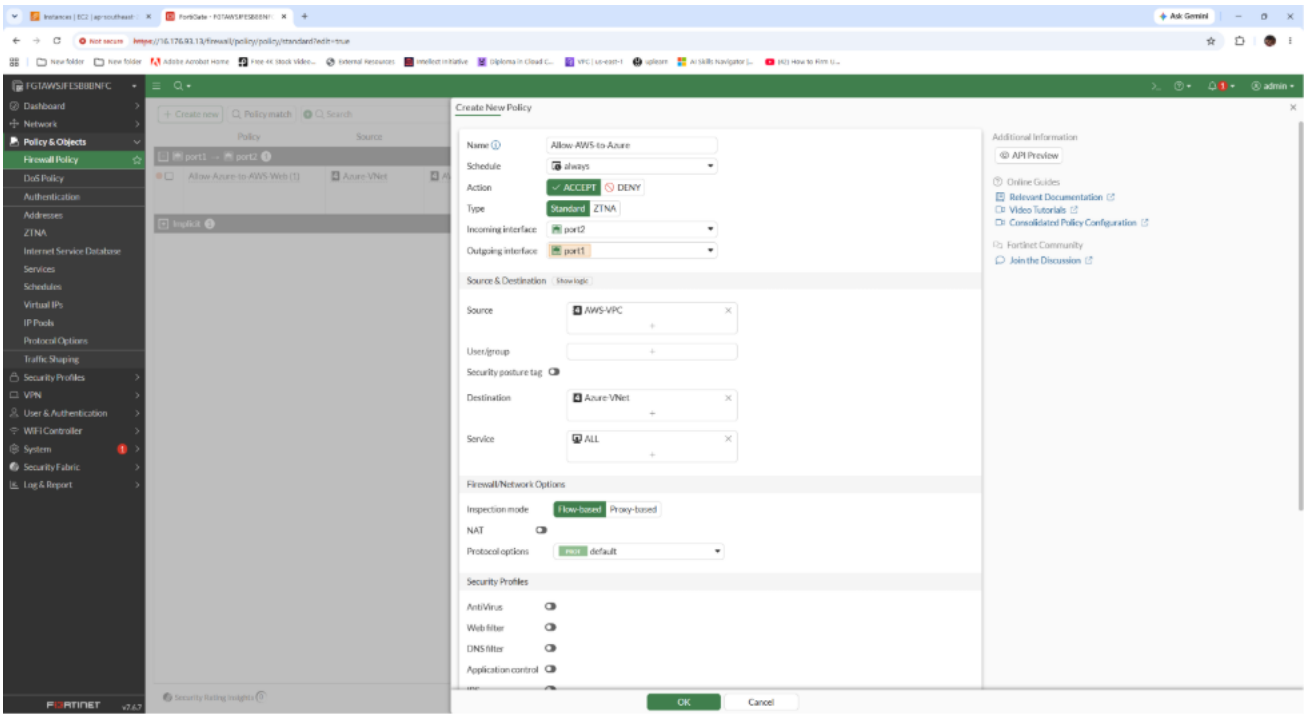
Address objects were created to represent important AWS and Azure network ranges. These objects simplify firewall policy configuration and make the rules easier to manage and understand.

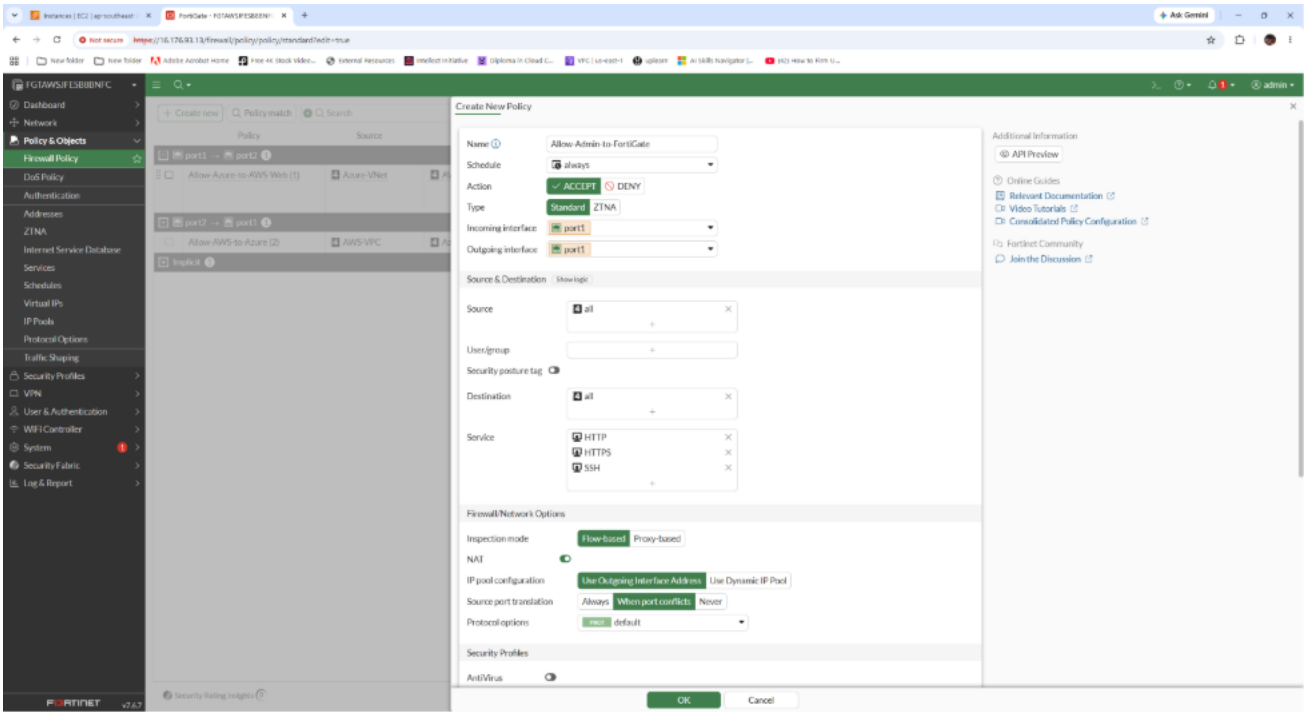


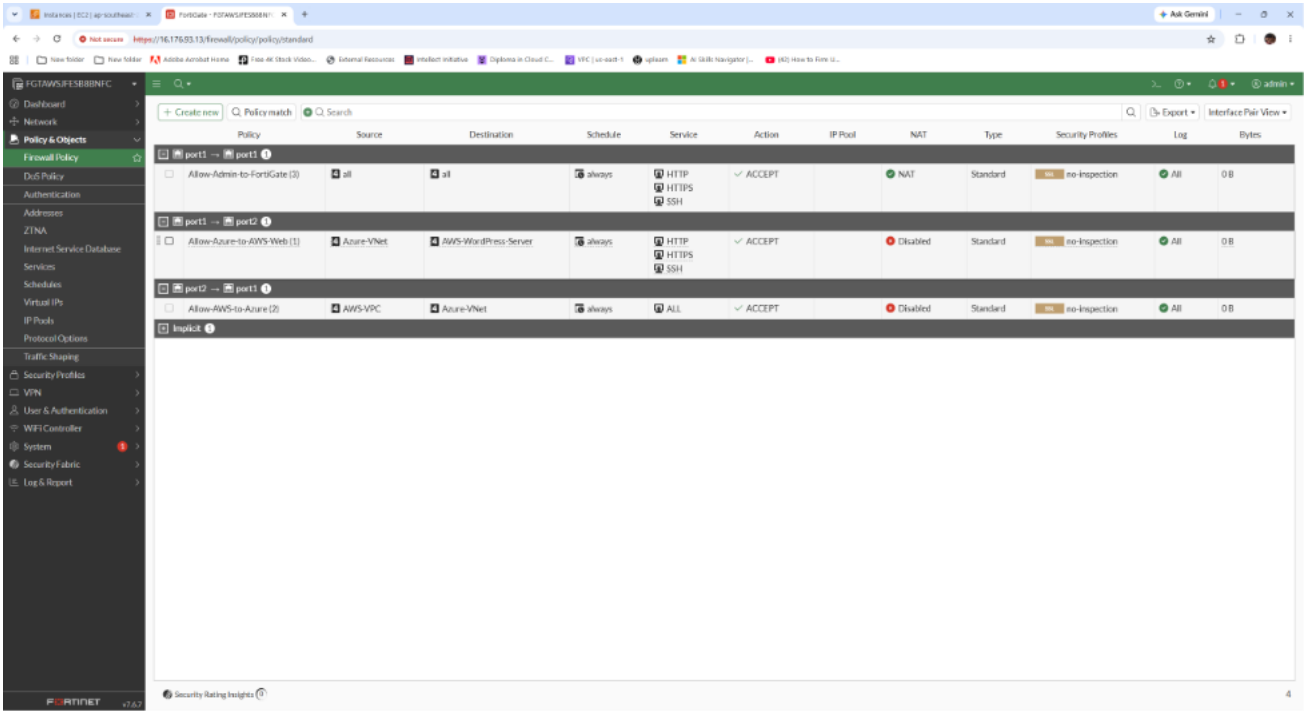
FortiGate Firewall Policies

Firewall policies were created to control traffic between the AWS environment and the planned Azure network. The policies define allowed communication paths and demonstrate how FortiGate can enforce traffic control between cloud environments.



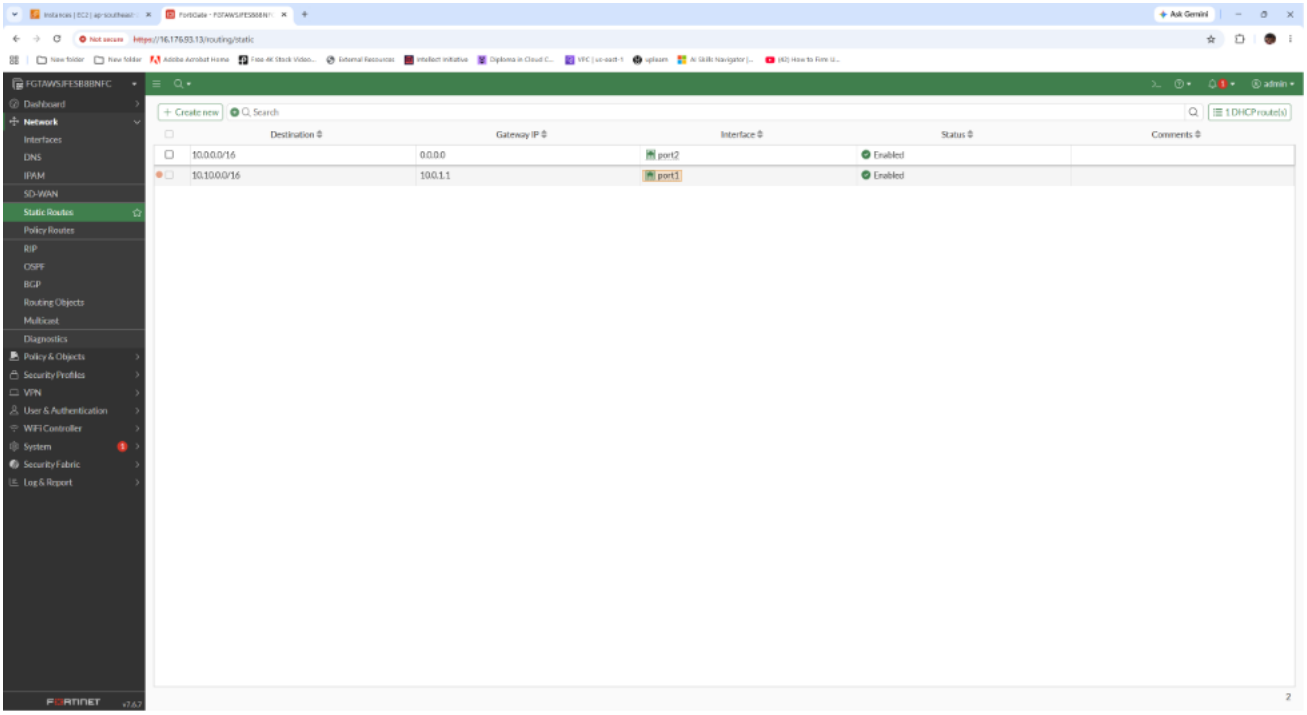



Create New Policy
Name
Allow-Admin-to-FortiGate
Schedule
always
Action
ACCEPT
DENY
Type
Standard
ZTNA
Incoming interface
port1
Outgoing interface
port1
Source & Destination
Source
all
User/group
Security posture tag
Destination
all
Service
HTTP
HTTPS
SSH
Firewall/Network Options
Inspection mode
Flow-based
Proxy-based
NAT
IP pool configuration
Use Outgoing Interface Address
Use Dynamic IP Pool
Source port translation
Always
When port conflicts
Never
Protocol options
default
Security Profiles
AntiVirus
OK
Cancel


Policy
Source
Destination
Schedule
Service
Action
IP Pool
NAT
Type
Security Profiles
Log
Bytes
port1 → port1
Allow-Admin-to-FortiGate (3)
all
all
always
HTTP
HTTPS
SSH
ACCEPT
NAT
Standard
no-inspection
All
0 B
port1 → port2
Allow-Azure-to-AWS-Web (1)
Azure-VNet
AWS-WordPress-Server
always
HTTP
HTTPS
SSH
ACCEPT
Disabled
Standard
no-inspection
All
0 B
port2 → port1
Allow-AWS-to-Azure (2)
AWS-VPC
Azure-VNet
always
ALL
ACCEPT
Disabled
Standard
no-inspection
All
0 B
Implicit

FortiGate Static Routes

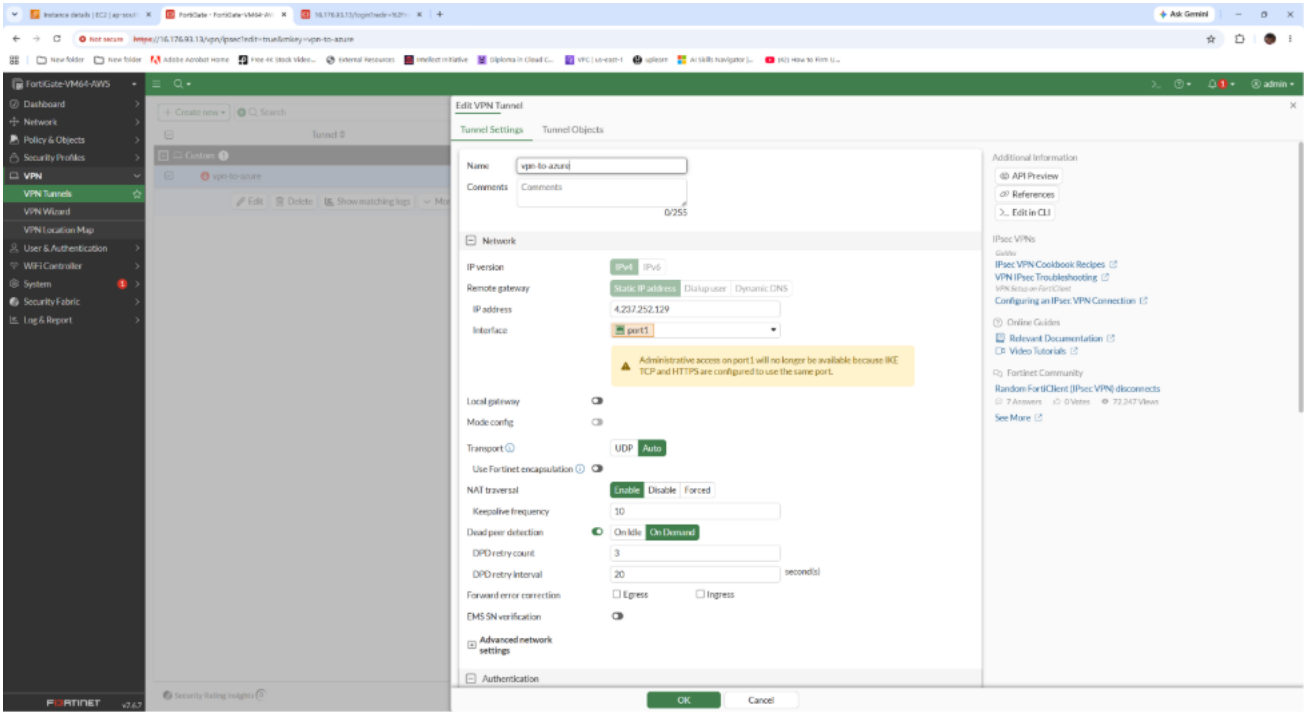
Static routes were configured to define traffic paths between the AWS network and the planned Azure network. These routes support future multi-cloud connectivity and ensure traffic is directed through the correct FortiGate interfaces.

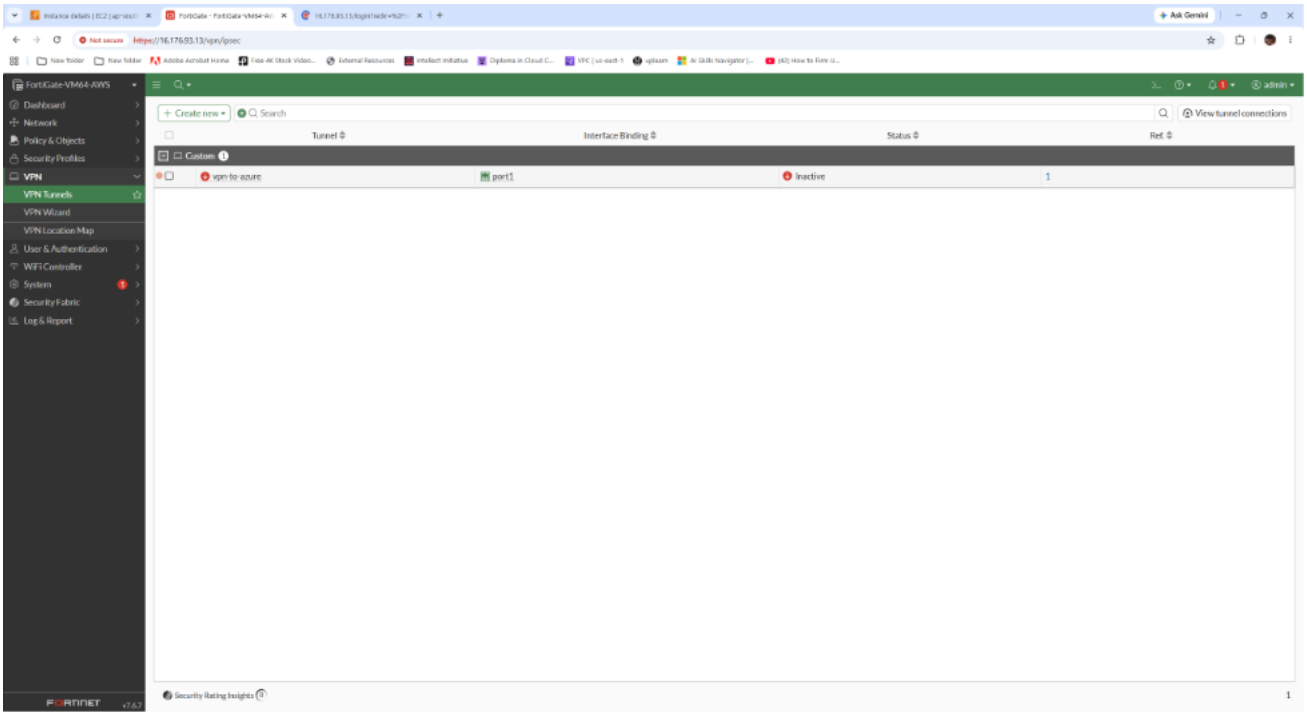


FortiGate IPsec VPN Tunnel Configuration

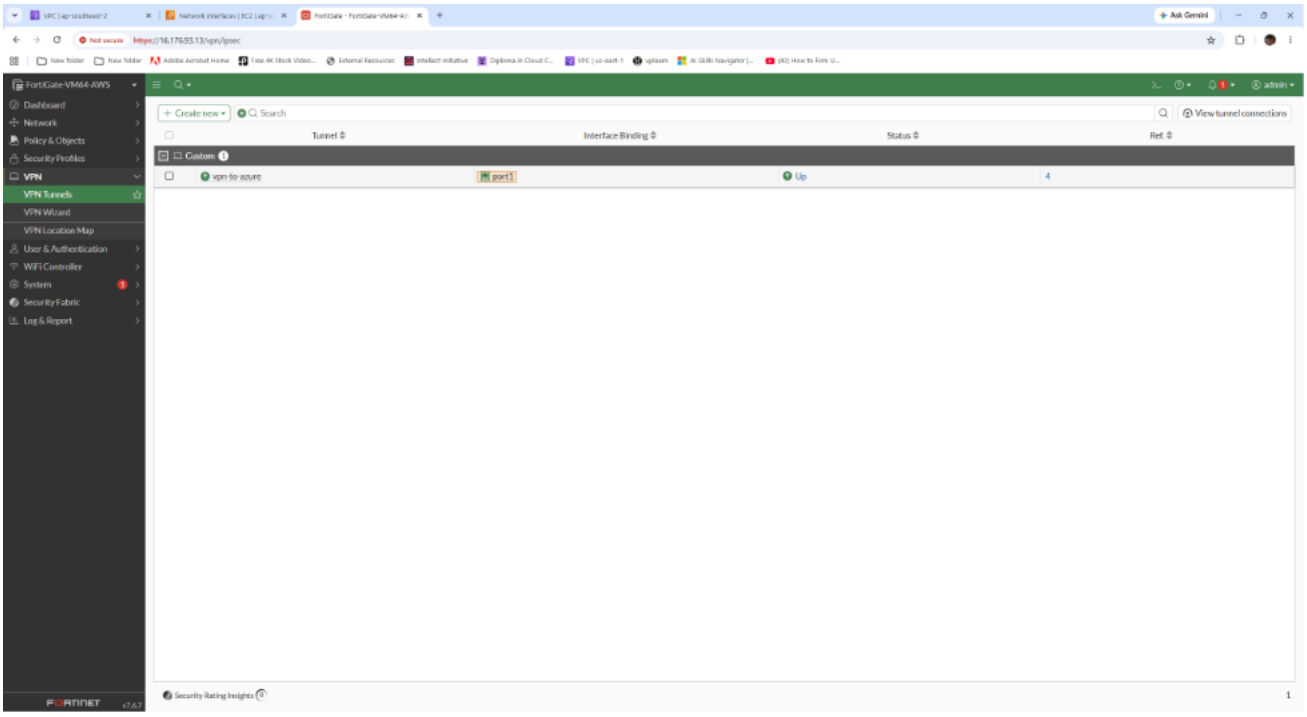
IPsec VPN tunnel settings were configured using the AWS-generated Fortinet VPN configuration file. The Phase 1 and Phase 2 parameters include IKE version, encryption, authentication, Diffie-Hellman group, key lifetime, and PFS settings.

The VPN tunnel remained inactive during testing because the remote Azure-side endpoint was not fully deployed. However, the FortiGate-side VPN configuration was prepared and ready for integration.



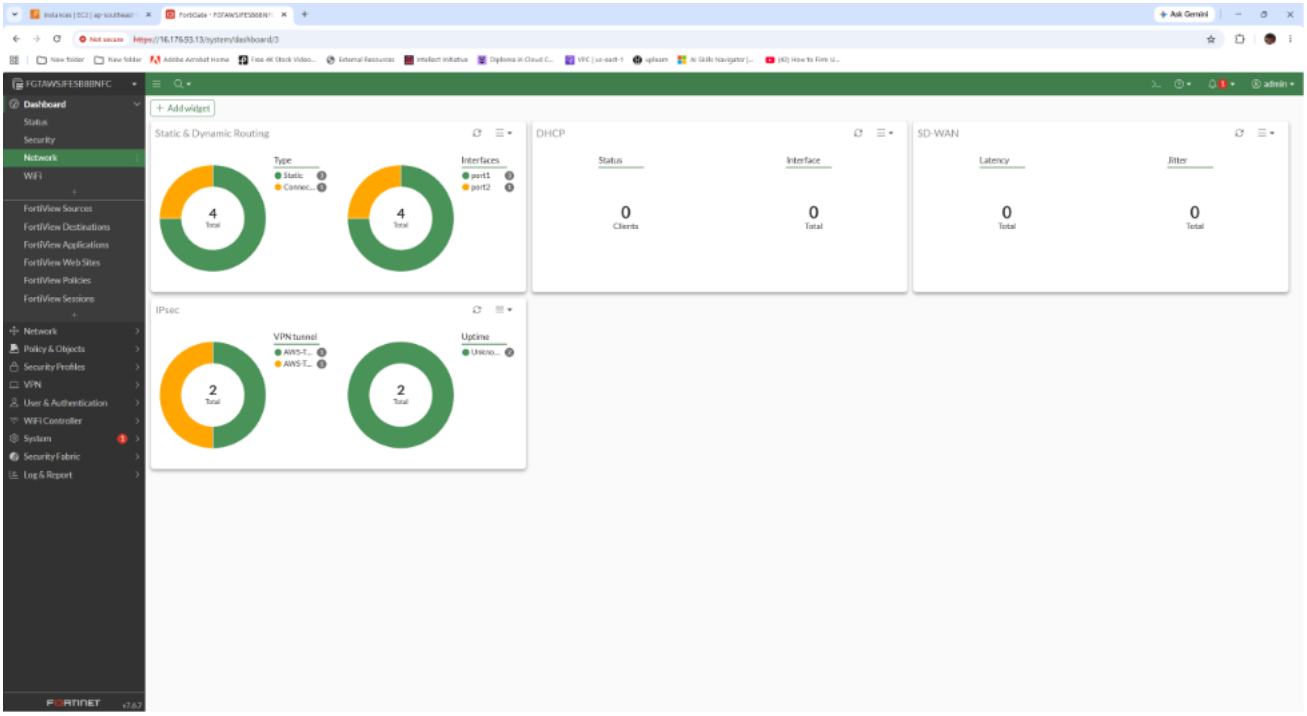


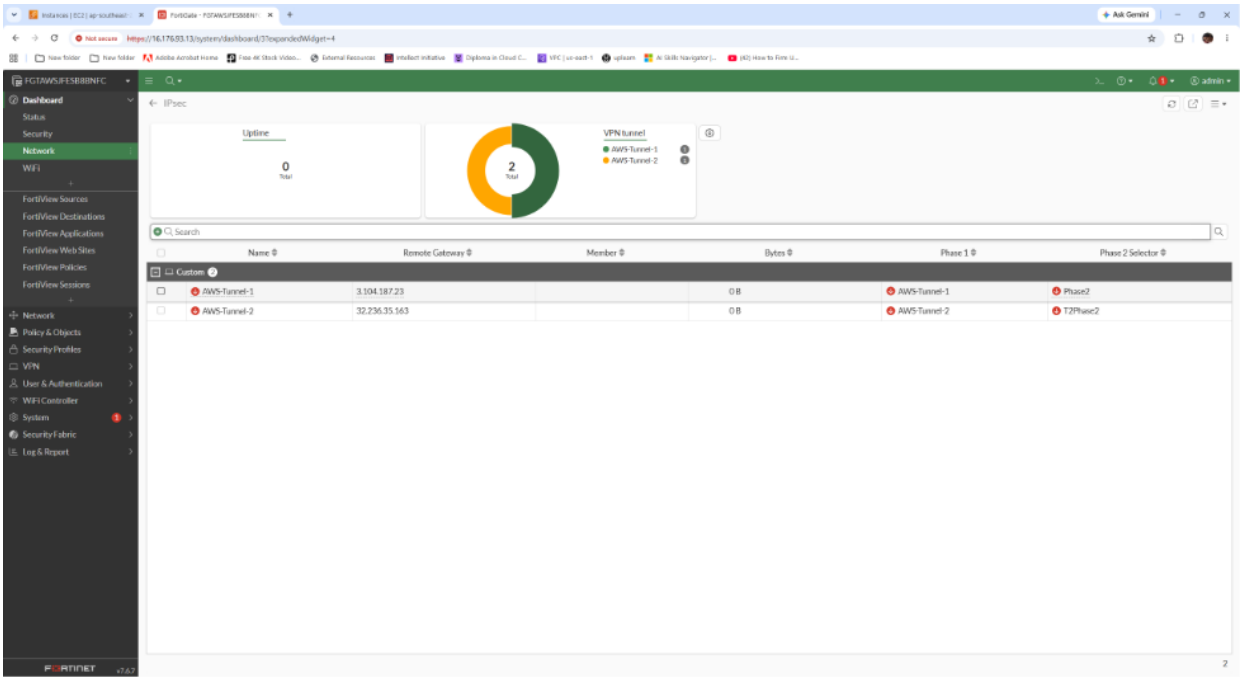
Verify Tunnel with Azure

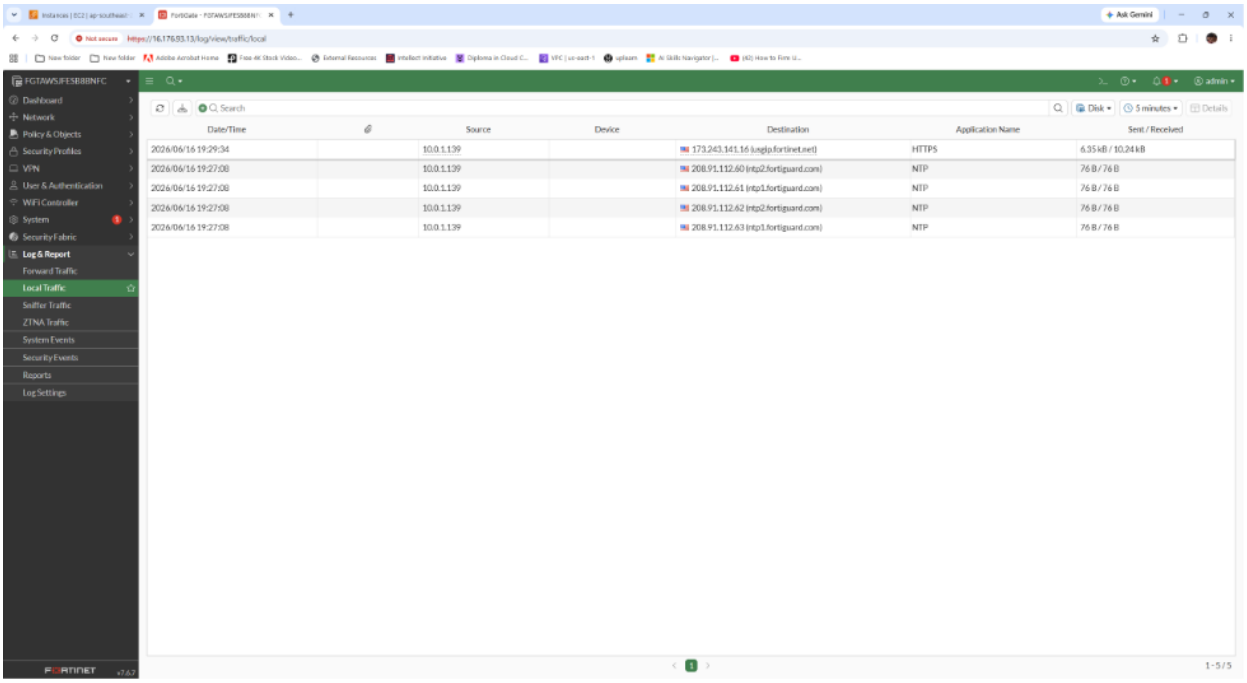


FortiGate Events

FortiGate system event logs were reviewed to verify administrative activity and configuration changes. Forward traffic logs were not populated because production traffic was not routed through the FortiGate firewall during this proof-of-concept stage.







AWS Implementation Conclusion

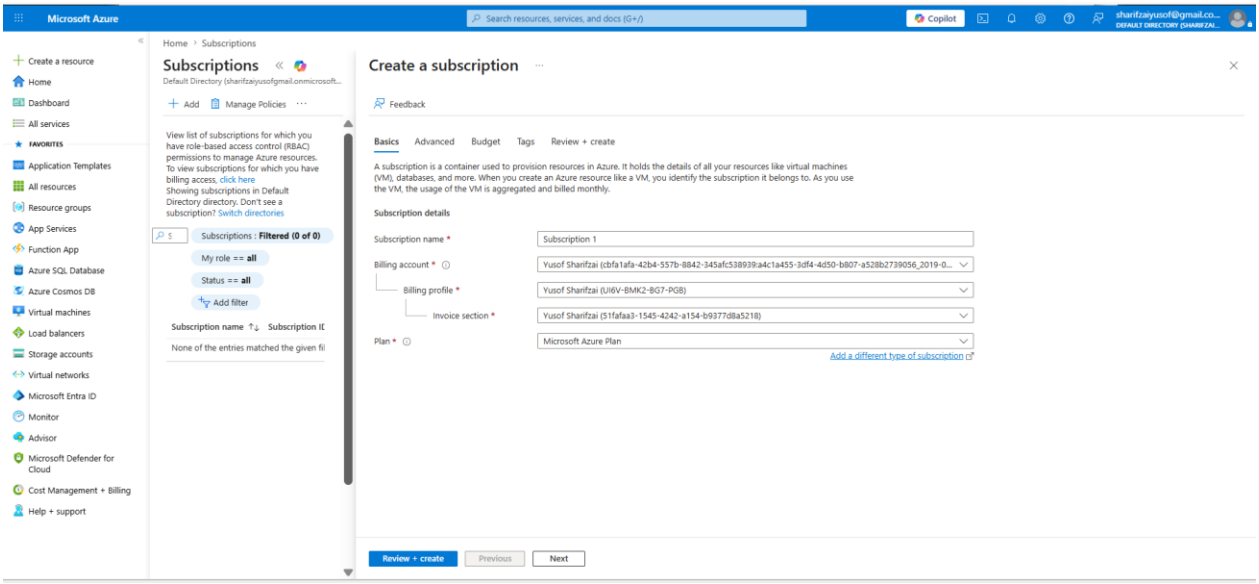
The AWS implementation successfully demonstrates a secure, scalable, and monitored hosting platform for the WordPress migration scenario. The design uses a custom VPC, public/private subnet separation, Security Groups, IAM roles, RDS, S3, Lambda, CloudWatch, SNS, ELB, Auto Scaling, ACM, and OpenVPN to address the project requirements. Security was considered through least-privilege access, restricted network paths, encryption, HTTPS, monitoring, and backup automation. The AWS environment also supports integration with Azure DNS and Fortinet FortiGate, allowing the project to meet the broader multi-cloud security integration objective.

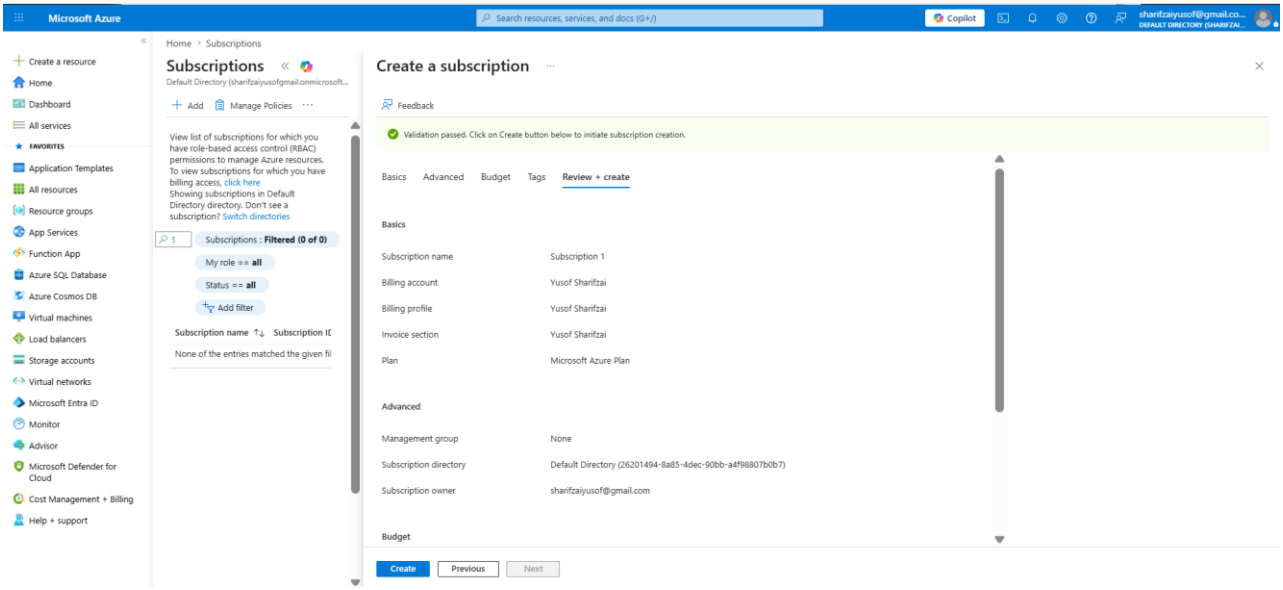
Task 2: Part B | Project Implementation using Azure Cloud Services (Yusof)

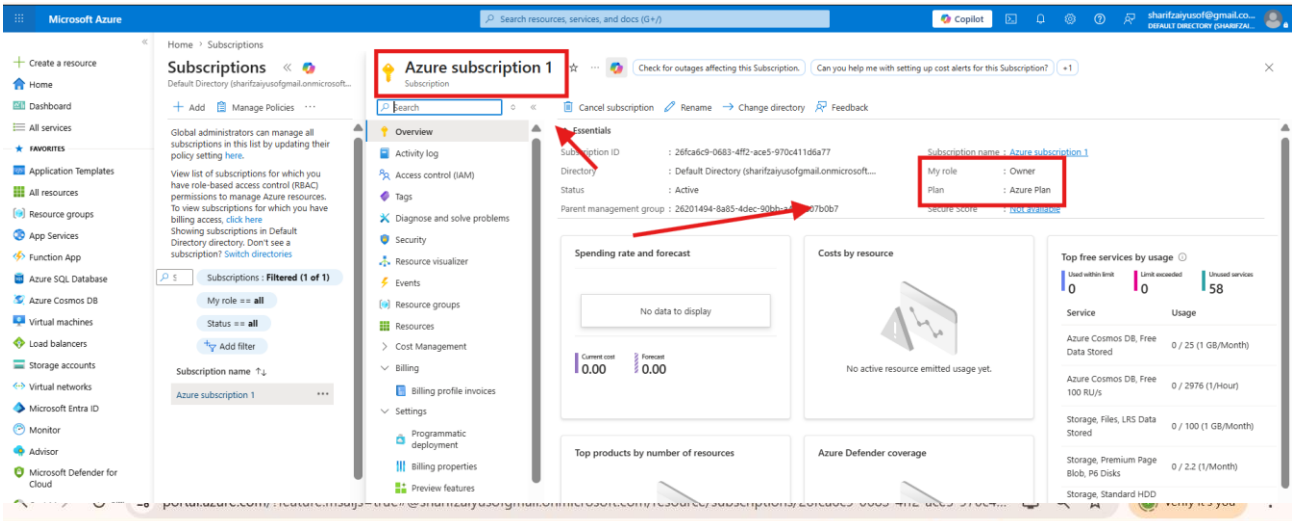
This section documents the Microsoft Azure implementation used to support the multi-cloud security project. Azure was used for subscription governance, account security, RBAC, MFA, Azure AD tenant configuration, Azure DNS, and external testing of secure access to the AWS-hosted WordPress application. The purpose of this section is to show how Azure services were integrated with AWS-hosted infrastructure to provide identity, access control, DNS name resolution, and secure connectivity validation.

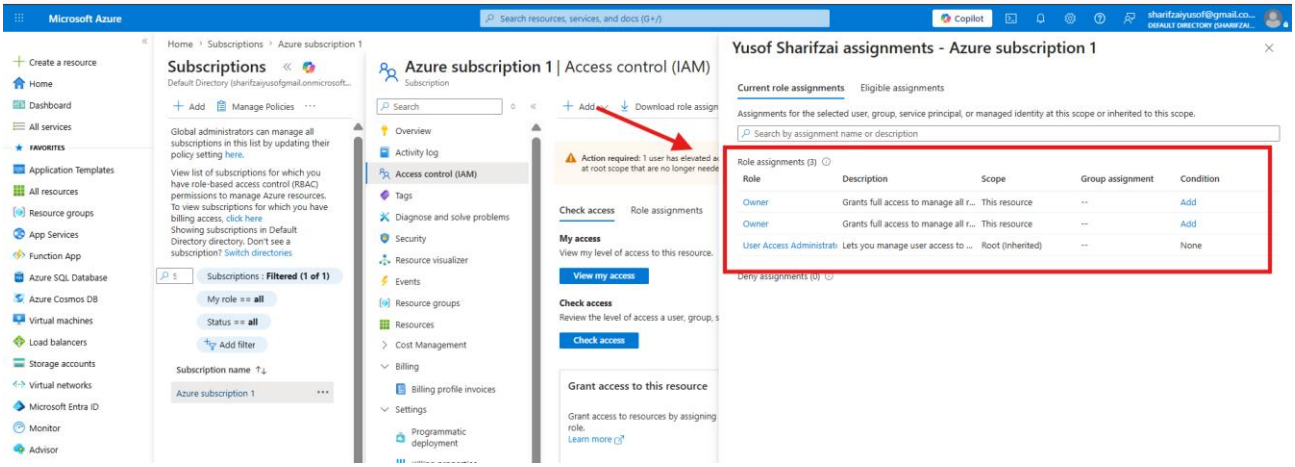
Azure Subscriptions - Configuration.

The first action We performed to implement Azure was to create an individual (dedicated) subscription. The reason for this is to provide a single place to manage our expenses and where all of our cloud resources will be contained; and therefore, can be managed as a singular entity. We set up a Pay-As-You-Go account for the subscription. We went back into the billing account section of the Azure Portal and created a new subscription using that method. Before completing the creation of the new subscription, we confirmed that our billing information was correctly populated within the billing profile/sections area. After completion of the new subscription creation, we then transferred the ownership of that new subscription over to the newly created Azure Active Directory Tenant. Transferring the ownership from our personal directory to the project specific directory enabled me to deploy all subsequent resources to the correct logical grouping. As such, once We had transferred the ownership of the subscription, we could ensure that all future deployments would remain associated with our Project-specific Directory versus being associated with our Default Personal Directory.




Microsoft Azure
Create a subscription
Basics
Advanced
Budget
Tags
Review + create
Subscription name
Subscription 1
Billing account
Billing profile
Invoice section
Plan
Microsoft Azure Plan
Management group
None
Subscription directory
Subscription owner
sharifzaiyusof@gmail.com
Create
Previous
Next


Microsoft Azure
Azure subscription 1
Subscription
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Security
Resource visualizer
Events
Resource groups
Resources
Cost Management
Billing
Billing profile invoices
Settings
Programmatic deployment
Billing properties
Preview features
Essentials
Spending rate and forecast
No data to display
Costs by resource
No active resource emitted usage yet.
Top products by number of resources
Azure Defender coverage
Top free services by usage

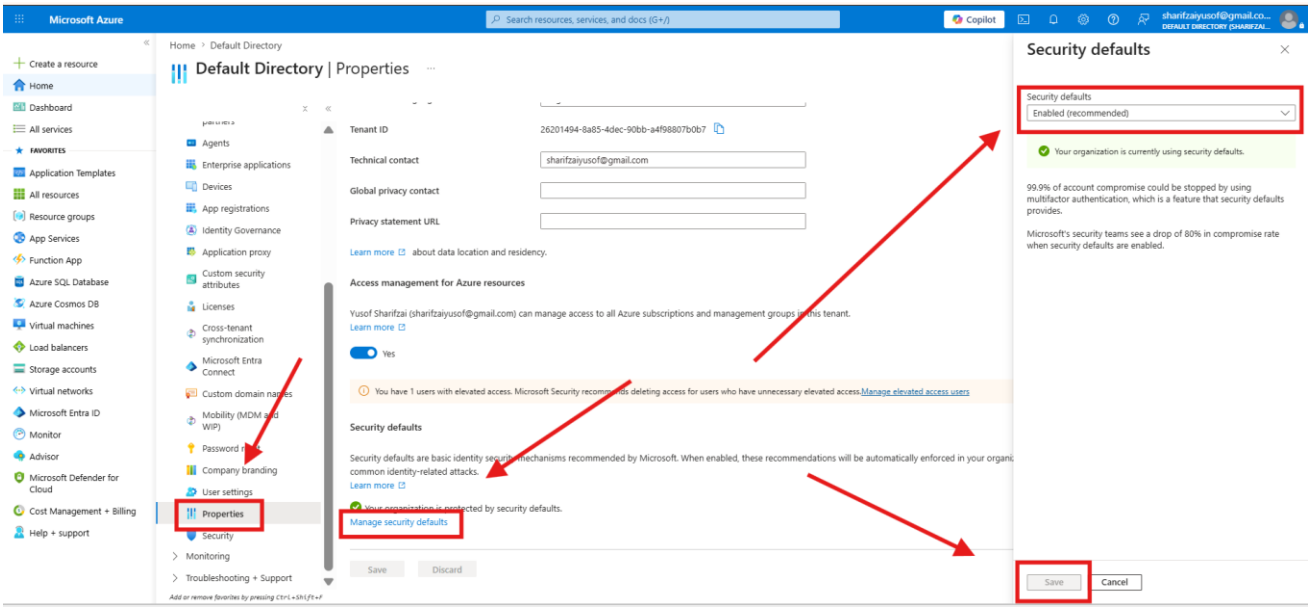

Microsoft Azure
Azure subscription 1 | Access control (IAM)
Yusof Sharifzai assignments - Azure subscription 1
Current role assignments
Eligible assignments
Role assignments (3)
Role
Description
Scope
Group assignment
Condition
Owner
Grants full access to manage all r...
This resource
Add
User Access Administrator
Lets you manage user access to ...
Root (Inherited)
None
Check access
Role assignments
My access
View my level of access to this resource.
View my access
Check access
Review the level of access a user, group,
Grant access to this resource

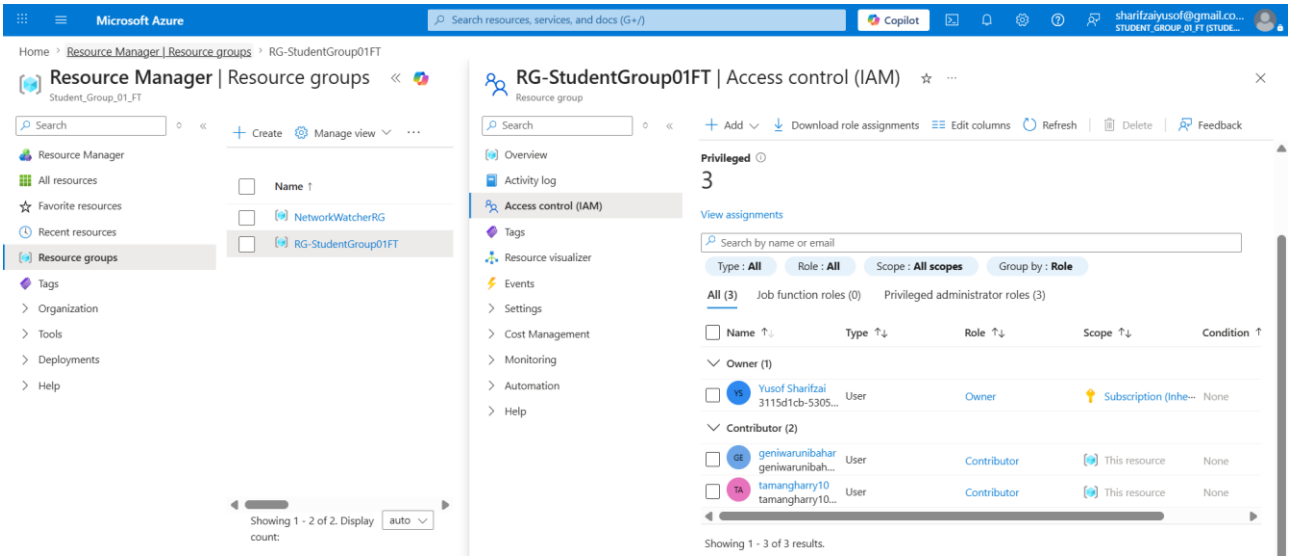
Implementing role-based access control and multi-factor authentication

To secure the Azure environment was a major concern for me, we chose to implement both Role-Based Access Control and multi-factor authentication from the beginning. In doing so, we turned on security defaults in Microsoft Entra Id, which forced all users to register for MFA (multi-factor authentication), and disabled legacy authentication protocols. The result was an immediate reduction in risk of being attacked with a reduced attack surface, since none would be able to access the tenant until the completion of another two-step form of verification.

With regards to access controls, we evaluated each of the roles assigned to our team members at both the subscription and resource group levels. As it pertained to ourself, we maintained ownership over our account to allow me to have full administrative authority to provision and configure core infrastructure. However, the remaining team members, whom required the ability to make deployments and manage resources, were given the contributor role at the resource group level. The contributor role allowed them to create and edit resources such as virtual networks and dns zone(s), however, did not grant them permission to add or remove contributors or view billable items. By implementing permissions in this manner, we granted each member what they required, while maintaining the principle of least privilege through-out the course of the project.

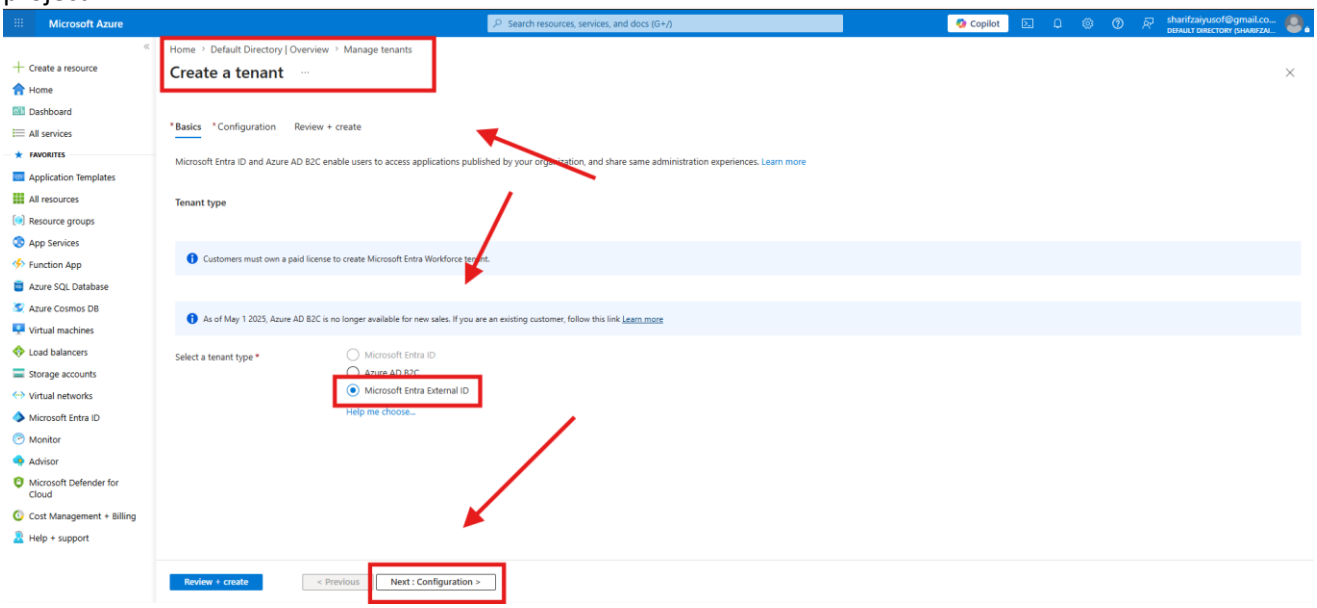
Control	Purpose
RBAC	Limits user permissions based on assigned role
MFA	Adds extra authentication protection
Azure AD tenant	Provides identity and access management boundary
Subscription access control	Ensures only approved users can manage resources

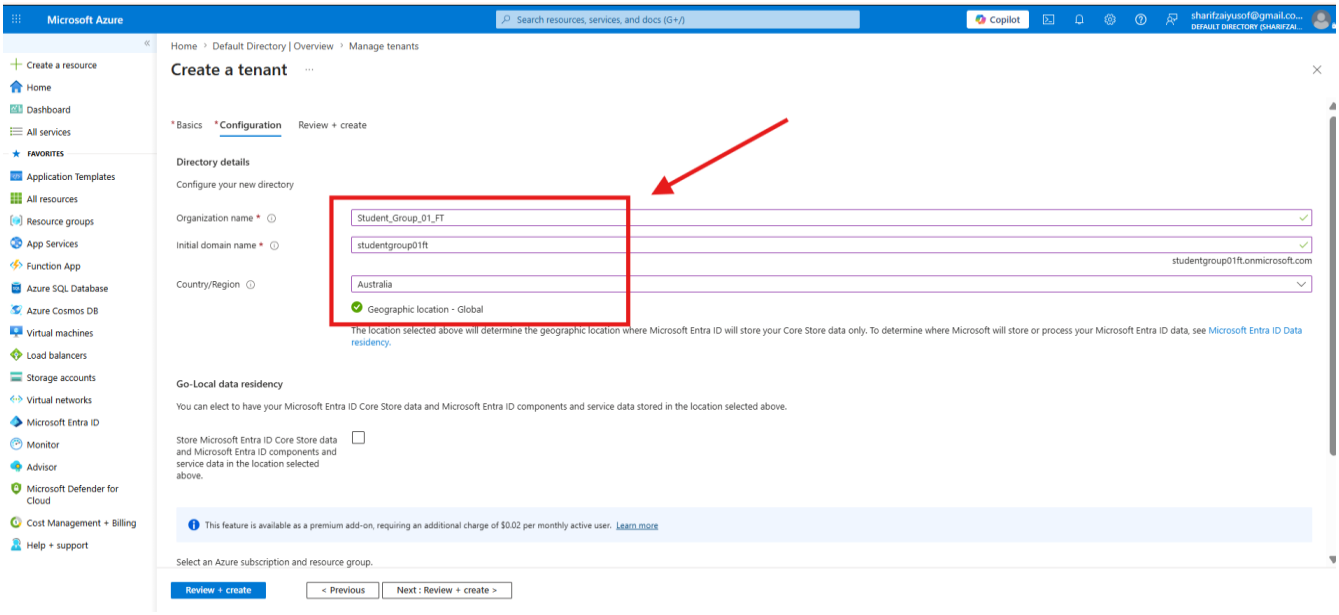


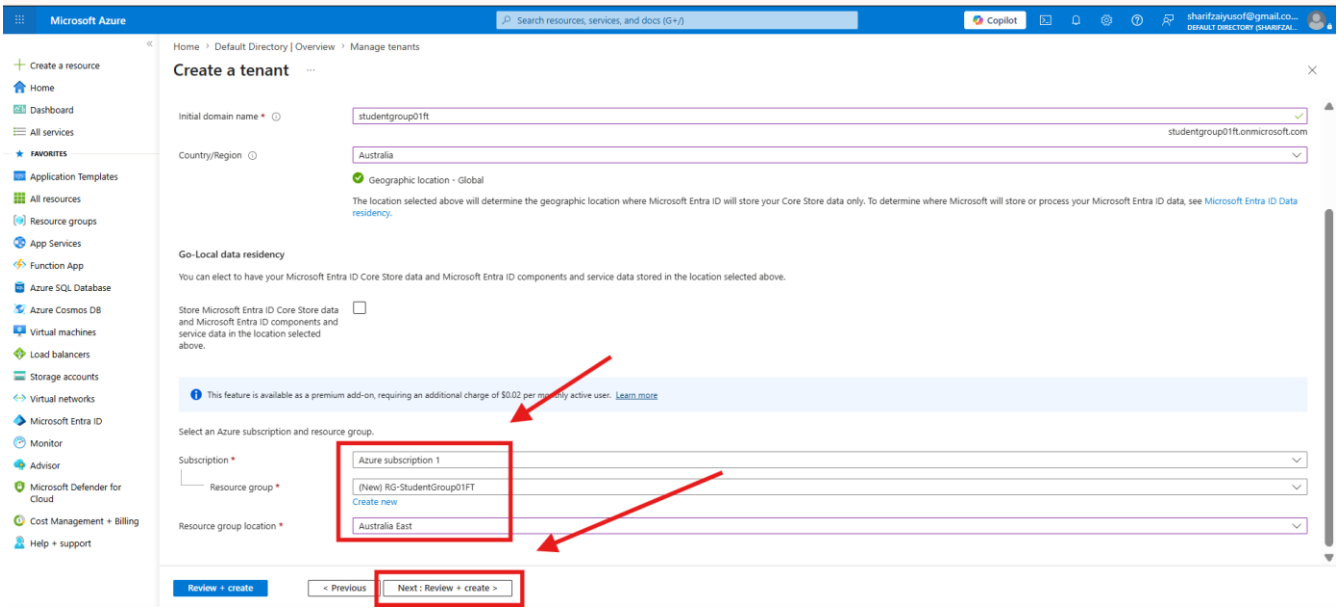


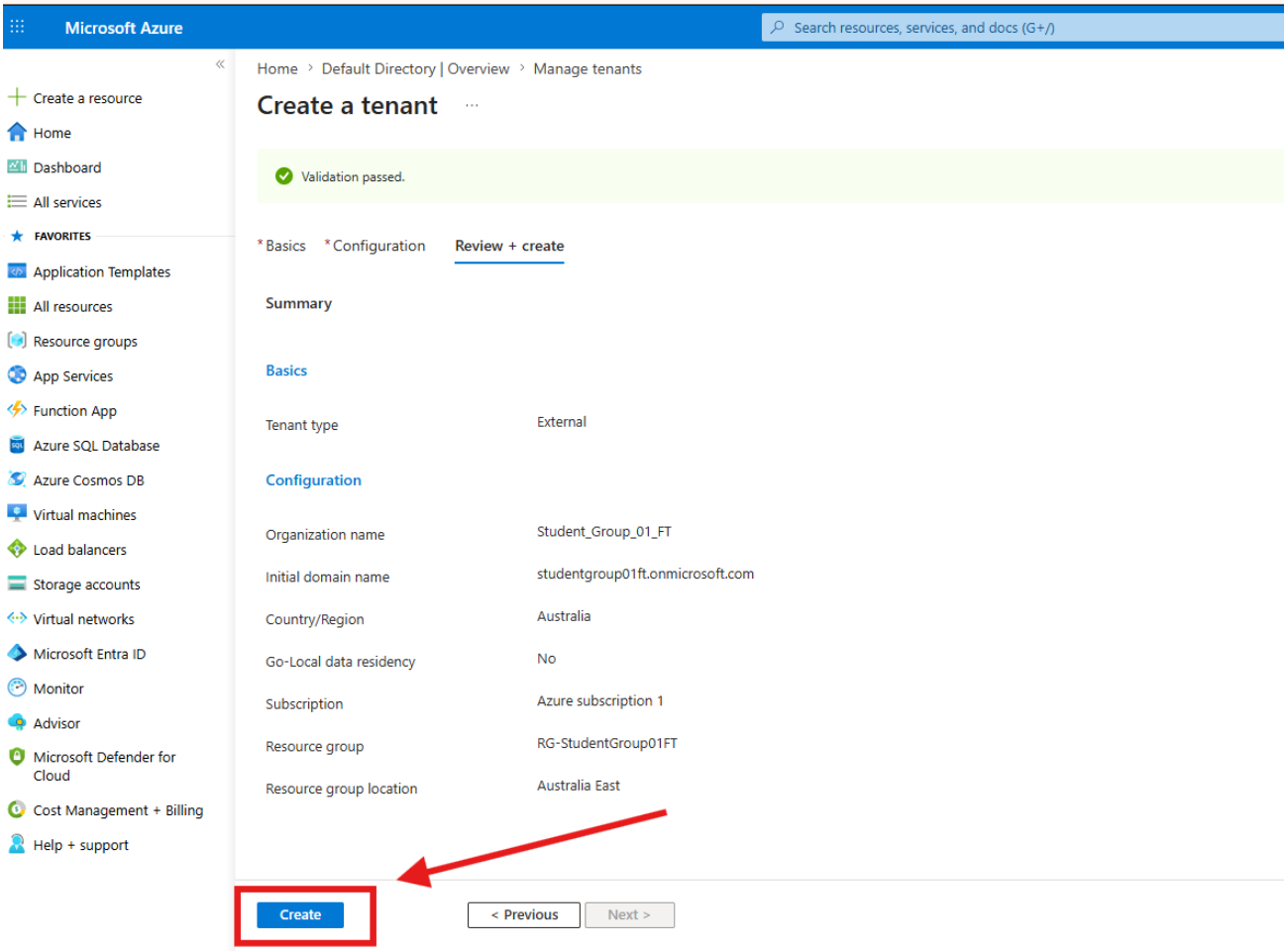
Creating an Azure Active Directory Tenant for the Project

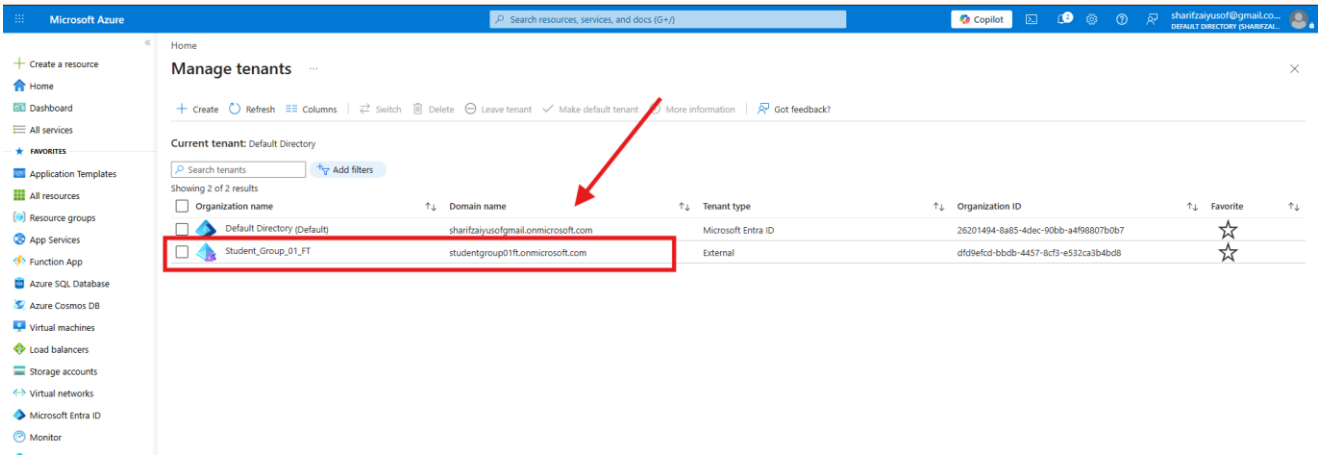
The next step in creating a defined boundary of identities for the project involved the creation of a new Microsoft Entra ID tenant called "Student_Group_01_FT". In creating the tenant, we provided an organization name; established the first domain to be studentgroup01ft.onmicrosoft.com; chose "Australia" from the list of countries/regions; associated this tenant to the active Azure subscription to which We have access; and added it to a resource group that is common to both this new tenant and existing project related resources. Upon completion of the provisioning process, the newly created tenant was visible in addition to our original directory in the AAD management portal. We then switched subscriptions to allow any additional Azure resources to be assigned to this tenant as well as ensure that they are managed under this tenant's organizational identity. The advantages of separating our resources included better auditing capabilities and improved administrative control over who can access what based on their roles in the project.

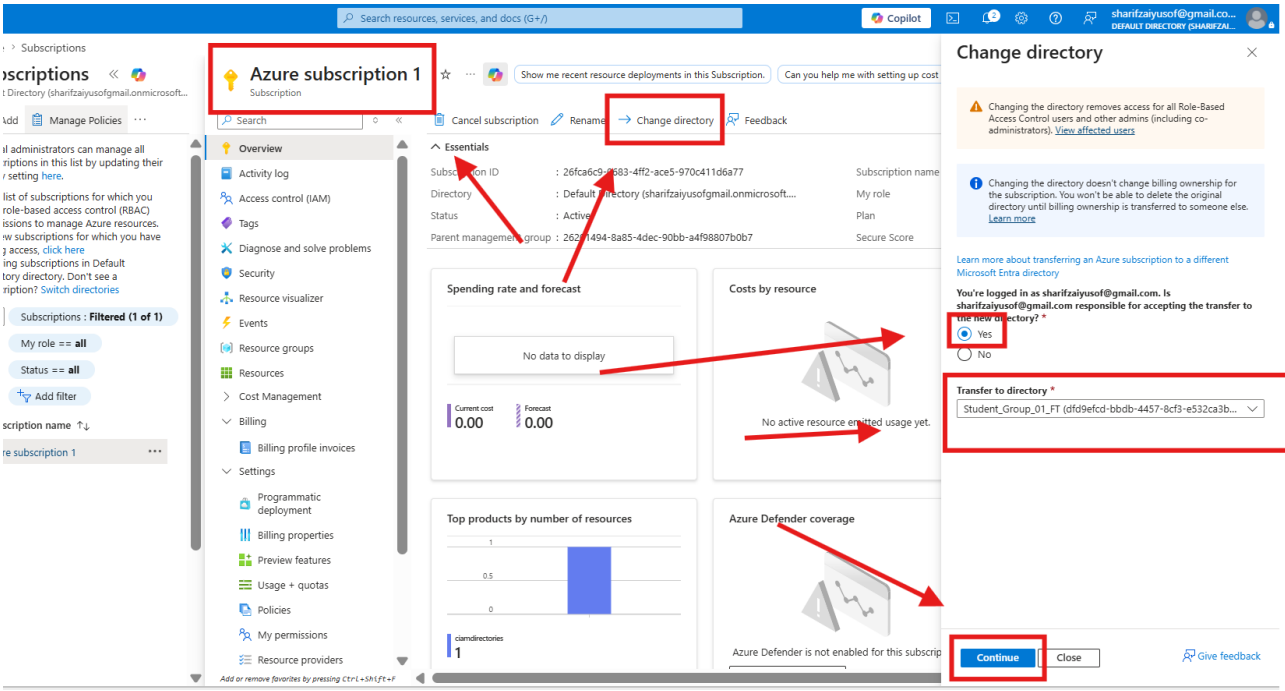


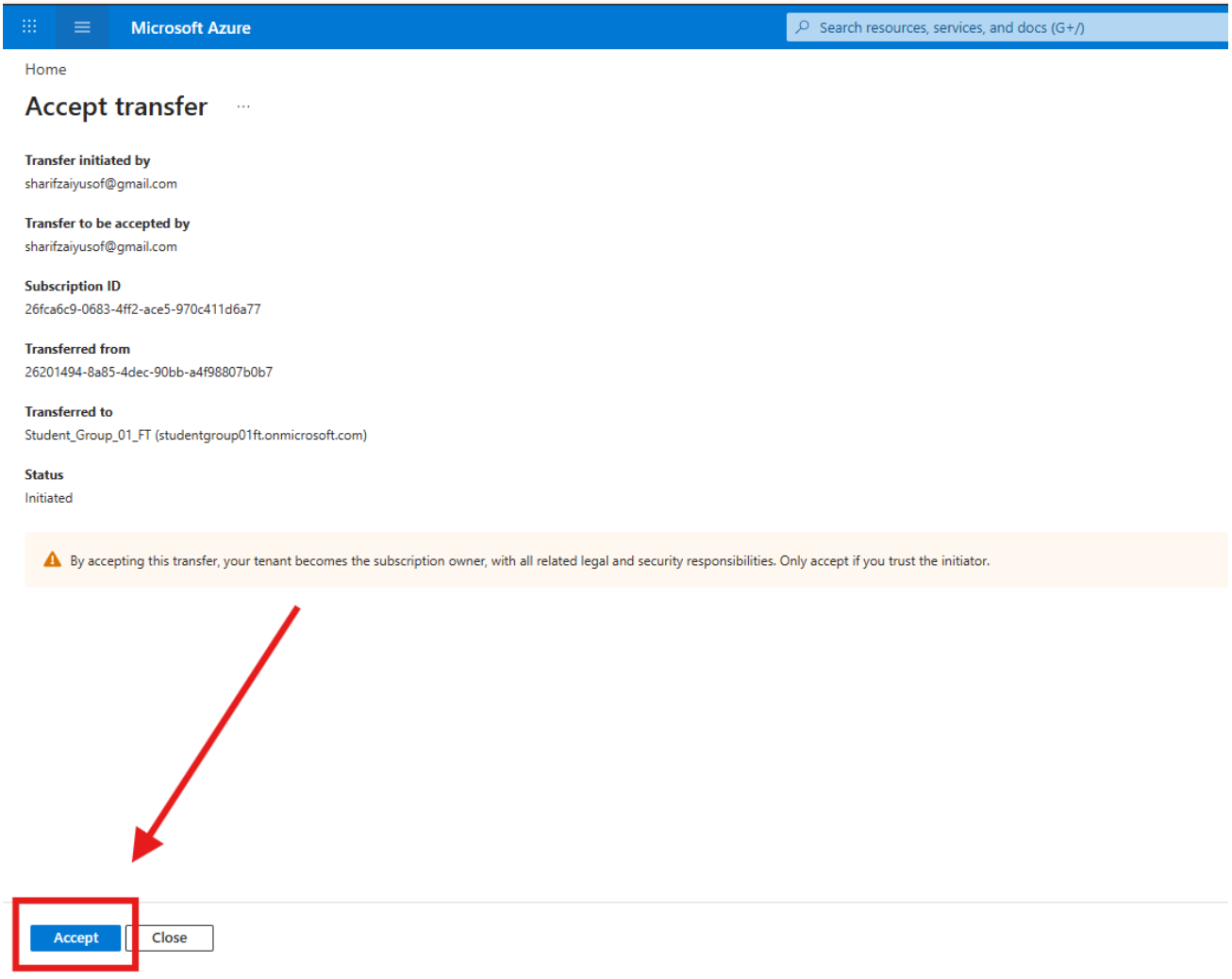

Microsoft Azure
Search resources, services, and docs (G+/)
Copilot
Home › Default Directory | Overview › Manage tenants
Create a tenant
Basics
Configuration
Review + create
Directory details
Configure your new directory
Organization name *
Student_Group_01_FT
Initial domain name *
studentgroup01ft
studentgroup01ft.onmicrosoft.com
Country/Region
Australia
Geographic location - Global
The location selected above will determine the geographic location where Microsoft Entra ID will store your Core Store data only. To determine where Microsoft will store or process your Microsoft Entra ID data, see Microsoft Entra ID Data residency.
Go-Local data residency
You can elect to have your Microsoft Entra ID Core Store data and Microsoft Entra ID components and service data stored in the location selected above.
Store Microsoft Entra ID Core Store data and Microsoft Entra ID components and service data in the location selected above.
This feature is available as a premium add-on, requiring an additional charge of $0.02 per monthly active user. Learn more
Select an Azure subscription and resource group.
Review + create
< Previous
Next : Review + create >


Microsoft Azure
Search resources, services, and docs (G+/)
Copilot
Home › Default Directory | Overview › Manage tenants
Create a tenant
Initial domain name *
studentgroup01ft
studentgroup01ft.onmicrosoft.com
Country/Region
Australia
Geographic location - Global
The location selected above will determine the geographic location where Microsoft Entra ID will store your Core Store data only. To determine where Microsoft will store or process your Microsoft Entra ID data, see Microsoft Entra ID Data residency.
Go-Local data residency
You can elect to have your Microsoft Entra ID Core Store data and Microsoft Entra ID components and service data stored in the location selected above.
Store Microsoft Entra ID Core Store data and Microsoft Entra ID components and service data in the location selected above.
Select an Azure subscription and resource group.
Subscription *
Azure subscription 1
Resource group *
(New) RG-StudentGroup01FT
Create new
Resource group location *
Australia East
Review + create
< Previous
Next : Review + create >


Microsoft Azure
Search resources, services, and docs (G+/)
Create a resource
Home
Dashboard
All services
FAVORITES
Application Templates
All resources
Resource groups
App Services
Function App
Azure SQL Database
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Virtual networks
Microsoft Entra ID
Monitor
Advisor
Microsoft Defender for Cloud
Cost Management + Billing
Help + support
Home > Default Directory | Overview > Manage tenants
Create a tenant
Validation passed.
*Basics *Configuration Review + create
Summary
Basics
Tenant type External
Configuration
Organization name Student_Group_01_FT
Initial domain name studentgroup01ft.onmicrosoft.com
Country/Region Australia
Go-Local data residency No
Subscription Azure subscription 1
Resource group RG-StudentGroup01FT
Resource group location Australia East
Create
< Previous
Next >

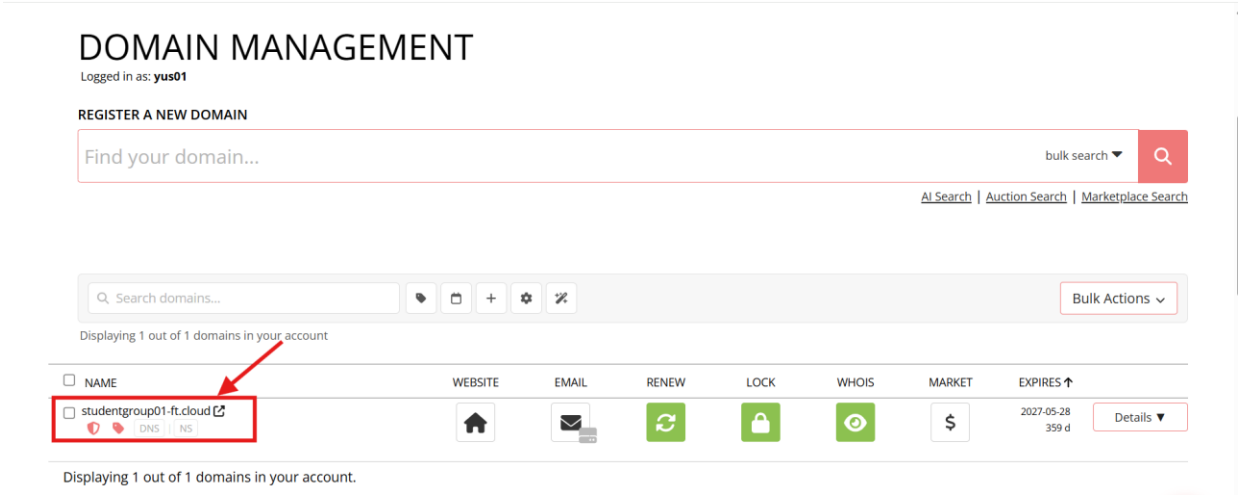

Microsoft Azure
Search resources, services, and docs (G+/)
Copilot
Home
Manage tenants
Create
Refresh
Columns
Switch
Delete
Leave tenant
Make default tenant
More information
Got feedback?
Current tenant: Default Directory
Search tenants
Add filters
Showing 2 of 2 results
Organization name
Domain name
Tenant type
Organization ID
Favorite
Default Directory (Default)
sharifzaiyusofgmail.onmicrosoft.com
Microsoft Entra ID
26201494-8a85-4dec-90bb-a4f98807b0b7
Student_Group_01_FT
studentgroup01ft.onmicrosoft.com
External
dfd9efcd-bbdb-4457-8cf3-e532ca3b4bd8

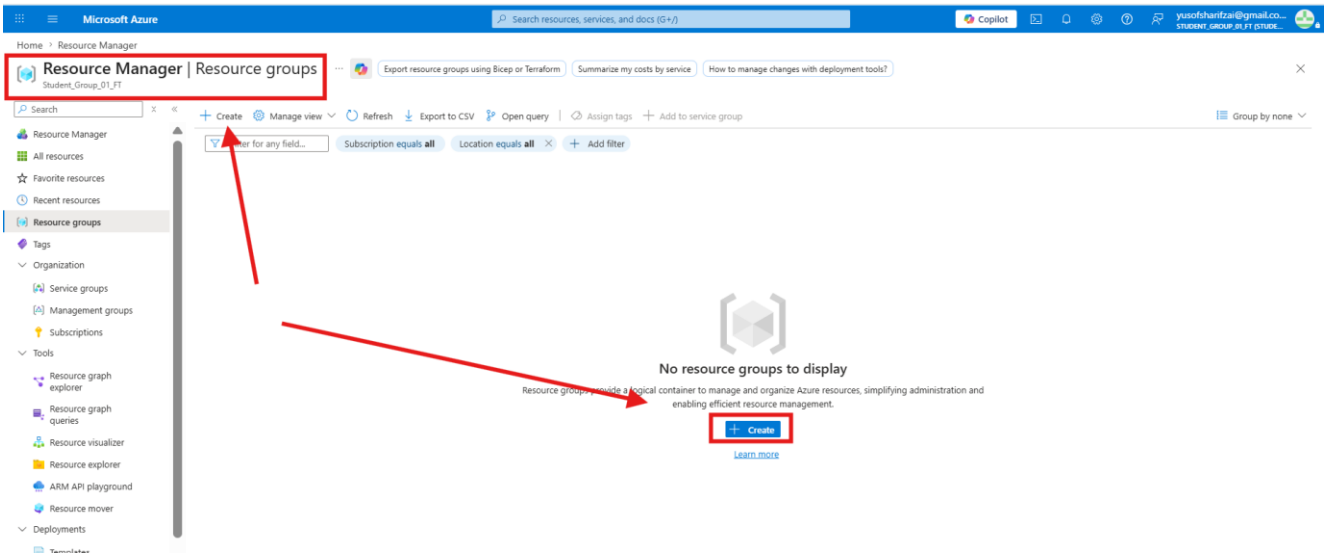

Search resources, services, and docs (G+/)
Copilot
Azure subscription 1
Subscription
Show me recent resource deployments in this Subscription.
Can you help me with setting up cost
Cancel subscription
Rename
Change directory
Feedback
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Security
Resource visualizer
Events
Resource groups
Resources
Cost Management
Billing
Billing profile invoices
Settings
Programmatic deployment
Billing properties
Preview features
Usage + quotas
Policies
My permissions
Resource providers
Essentials
Directory : Default Directory (sharifzaiyusofgmail.onmicrosoft....
Status : Active
Parent management group : 26201494-8a85-4dec-90bb-a4f98807b0b7
Subscription name
My role
Plan
Secure Score
Spending rate and forecast
No data to display
Current cost 0.00
Forecast 0.00
Costs by resource
No active resource emitted usage yet.
Top products by number of resources
Azure Defender coverage
Azure Defender is not enabled for this subscrip
Change directory
Changing the directory removes access for all Role-Based Access Control users and other admins (including co-administrators). View affected users
Changing the directory doesn't change billing ownership for the subscription. You won't be able to delete the original directory until billing ownership is transferred to someone else. Learn more
Learn more about transferring an Azure subscription to a different Microsoft Entra directory
You're logged in as sharifzaiyusof@gmail.com. Is sharifzaiyusof@gmail.com responsible for accepting the transfer to the new directory? *
Yes
No
Transfer to directory *
Student_Group_01_FT (dfd9efcd-bbdb-4457-8cf3-e532ca3b...
Continue
Close
Give feedback

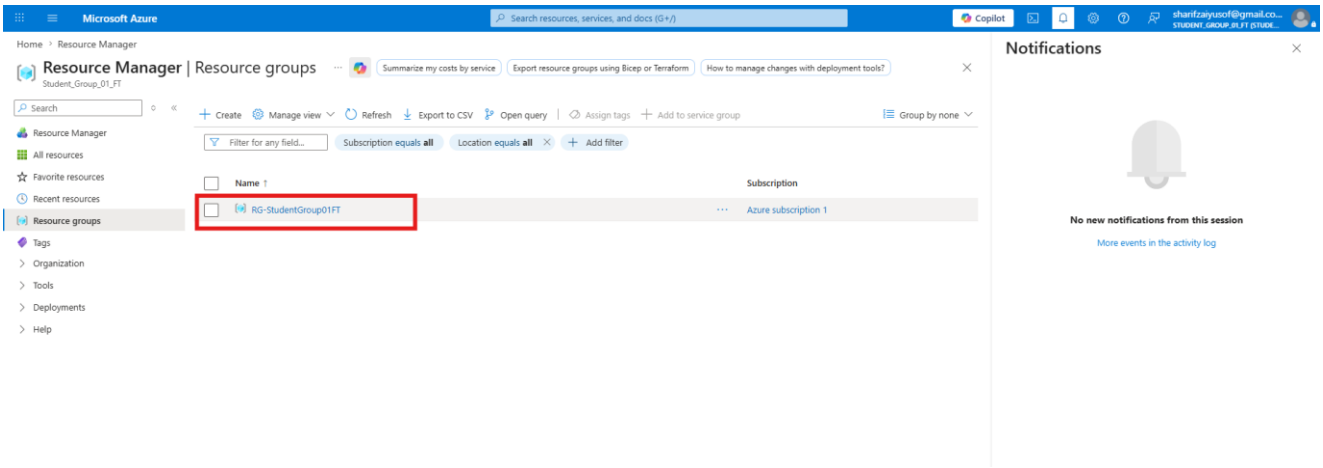

Microsoft Azure
Search resources, services, and docs (G+/)
Home
Accept transfer
Transfer initiated by
sharifzaiyusof@gmail.com
Transfer to be accepted by
sharifzaiyusof@gmail.com
Subscription ID
26fca6c9-0683-4ff2-ace5-970c411d6a77
Transferred from
26201494-8a85-4dec-90bb-a4f98807b0b7
Transferred to
Student_Group_01_FT (studentgroup01ft.onmicrosoft.com)
Status
Initiated
By accepting this transfer, your tenant becomes the subscription owner, with all related legal and security responsibilities. Only accept if you trust the initiator.
Accept
Close

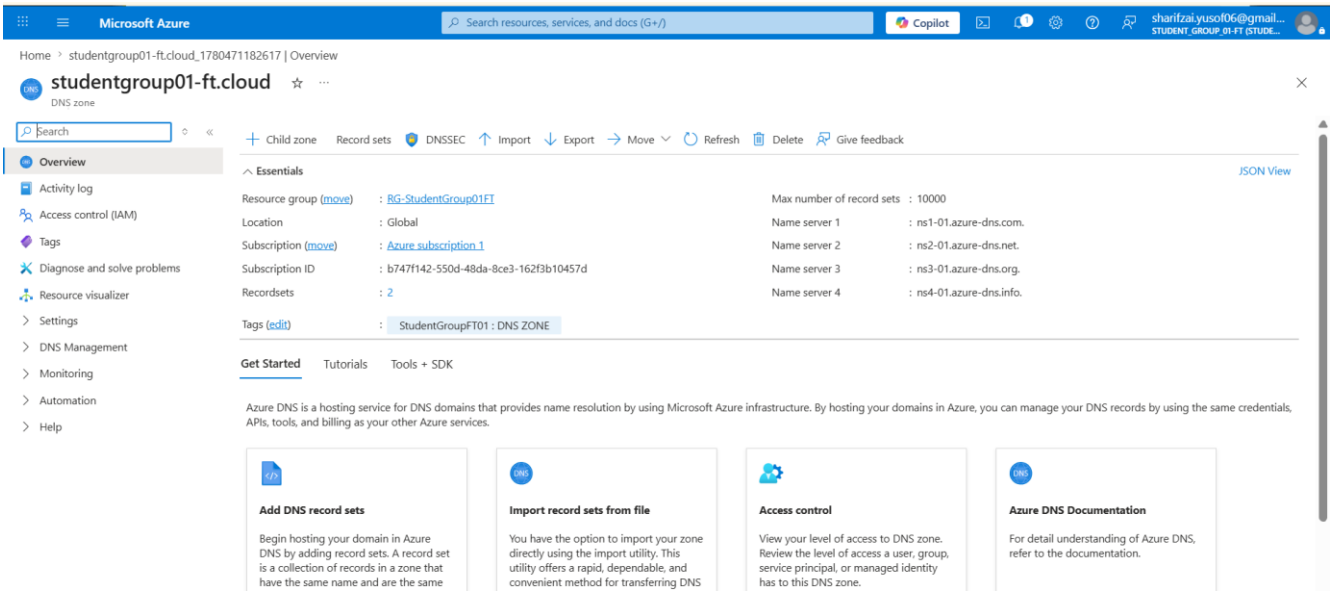
DNS Integration with AWS (Azure DNS Zone and Registrar Update)

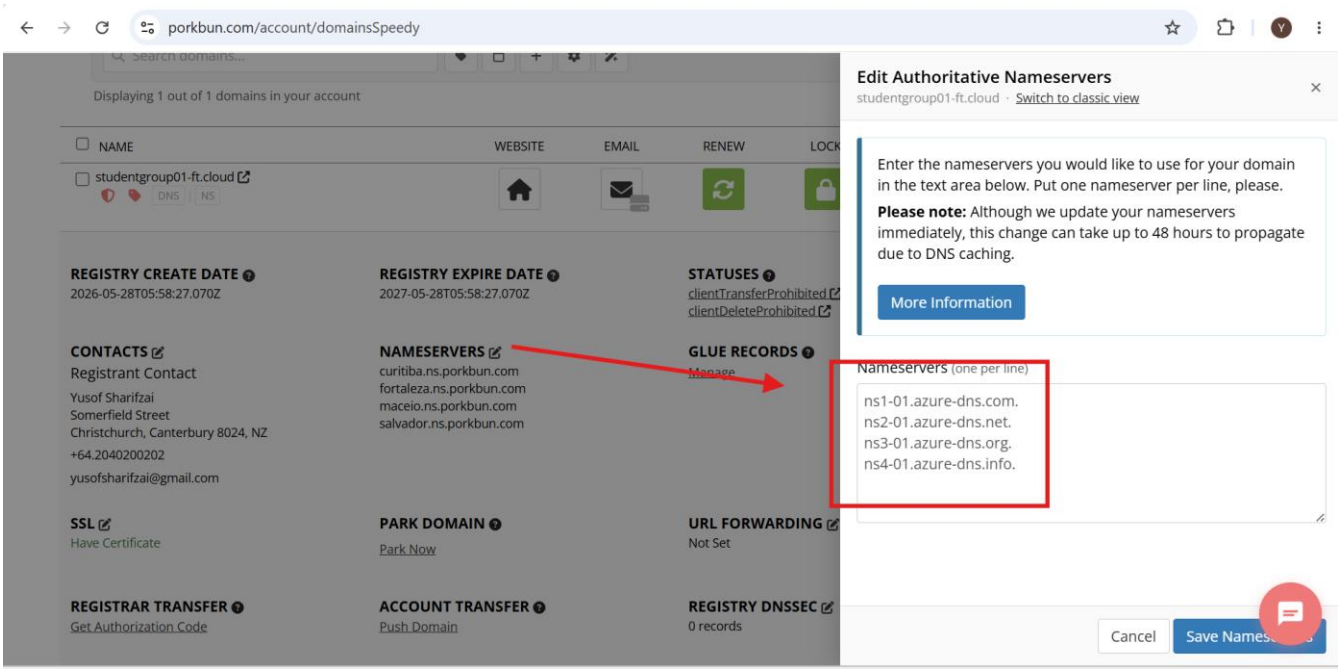
DNS integration with hosted WordPress on AWS through an update of our Azure DNS Zone and Registrar. To connect our hosted WordPress on AWS to the Azure DNS services, we had to configure the two separately. We first bought a DNS from **porkbun.com** named studentgroup01-ft.cloud and then configured our Azure DNS Services by creating a Public DNS Zone for studentgroup01-ft.cloud. The four Authoritative Name Servers were automatically generated for me when We created the zone. Then, we went back to our Domain Registrar and replaced their Default Name Servers with the newly created Azure Name Servers. The process delegated all future DNS Record Management to be done from within the Azure Portal.
Once the Zone was activated, we set up all of the DNS Records needed for public access. We created a CNAME record at the level of the www sub-domain that pointed it to the AWS Application Load Balancer that was serving WordPress Site created by team member **Geni**. With this single alias, we was able to provide visitors to our website with a clean-branded domain name vs. one of the long URLs that AWS generates. We also established the required DNS Validation Record for **Geni** so they could verify our ownership of the domain and thus generate a legitimate SSL/TLS certificate for the site. Once the DNS Propagation was completed, we verified that our domain was resolved as expected.

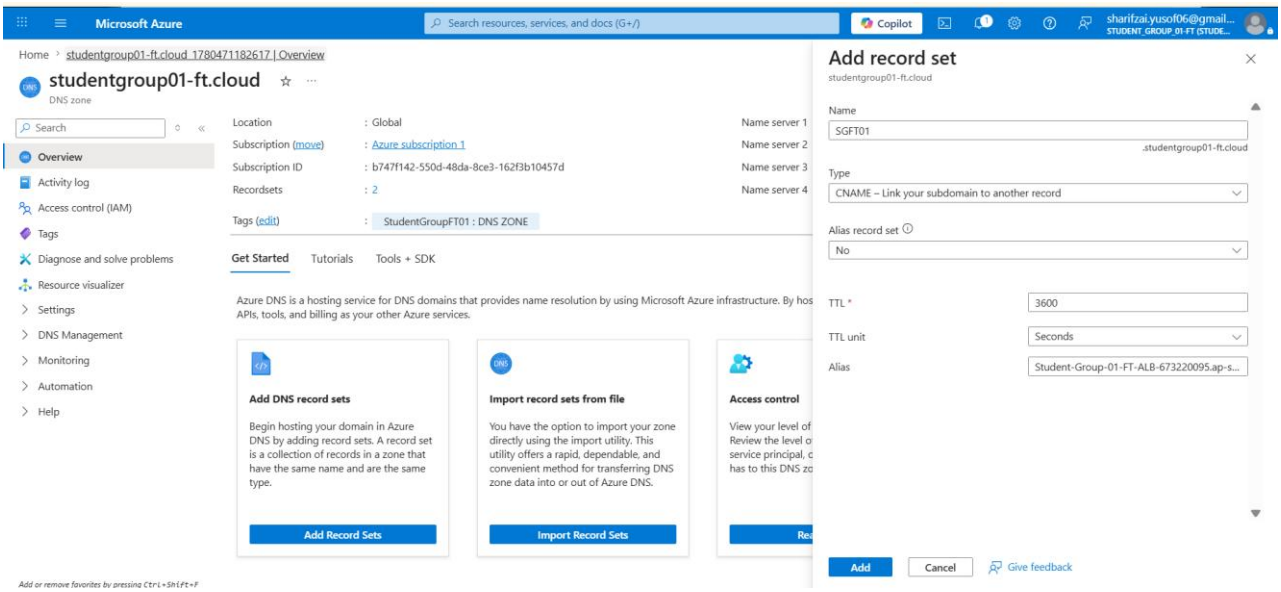


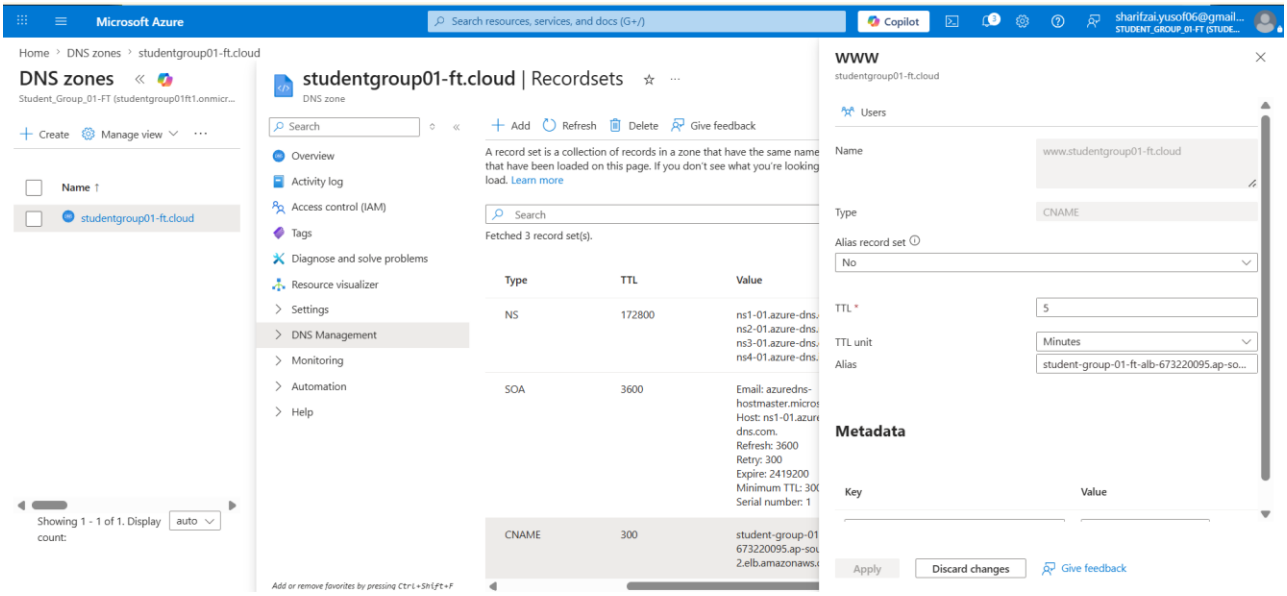



Microsoft Azure
Search resources, services, and docs (G+/)
Copilot
Home › Resource Manager
Resource Manager | Resource groups
Student_Group_01_FT
Summarize my costs by service
Export resource groups using Bicep or Terraform
How to manage changes with deployment tools?
Search
Create
Manage view
Refresh
Export to CSV
Open query
Assign tags
Add to service group
Group by none
Filter for any field...
Subscription equals all
Location equals all
Add filter
Resource Manager
All resources
Favorite resources
Recent resources
Resource groups
Tags
Organization
Tools
Deployments
Help
Name
Subscription
RG-StudentGroup01FT
Azure subscription 1
Notifications
No new notifications from this session
More events in the activity log


Microsoft Azure
Search resources, services, and docs (G+/)
Copilot
Home › studentgroup01-ft.cloud_1780471182617 | Overview
studentgroup01-ft.cloud
DNS zone
Search
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Resource visualizer
Settings
DNS Management
Monitoring
Automation
Help
Child zone
Record sets
DNSSEC
Import
Export
Move
Refresh
Delete
Give feedback
Essentials
JSON View
Resource group (move) : RG-StudentGroup01FT
Location : Global
Subscription (move) : Azure subscription 1
Subscription ID : b747f142-550d-48da-8ce3-162f3b10457d
Recordsets : 2
Tags (edit) : StudentGroupFT01 : DNS ZONE
Max number of record sets : 10000
Name server 1 : ns1-01.azure-dns.com.
Name server 2 : ns2-01.azure-dns.net.
Name server 3 : ns3-01.azure-dns.org.
Name server 4 : ns4-01.azure-dns.info.
Get Started
Tutorials
Tools + SDK
Azure DNS is a hosting service for DNS domains that provides name resolution by using Microsoft Azure infrastructure. By hosting your domains in Azure, you can manage your DNS records by using the same credentials, APIs, tools, and billing as your other Azure services.
Add DNS record sets
Begin hosting your domain in Azure DNS by adding record sets. A record set is a collection of records in a zone that have the same name and are the same
Import record sets from file
You have the option to import your zone directly using the import utility. This utility offers a rapid, dependable, and convenient method for transferring DNS
Access control
View your level of access to DNS zone. Review the level of access a user, group, service principal, or managed identity has to this DNS zone.
Azure DNS Documentation
For detail understanding of Azure DNS, refer to the documentation.


porkbun.com/account/domainsSpeedy
Displaying 1 out of 1 domains in your account
NAME
WEBSITE
EMAIL
RENEW
studentgroup01-ft.cloud
REGISTRY CREATE DATE
2026-05-28T05:58:27.070Z
REGISTRY EXPIRE DATE
2027-05-28T05:58:27.070Z
STATUSES
clientTransferProhibited
clientDeleteProhibited
CONTACTS
Registrant Contact
Yusof Sharifzai
Somerfield Street
Christchurch, Canterbury 8024, NZ
+64.2040200202
yusofsharifzai@gmail.com
NAMESERVERS
curitiba.ns.porkbun.com
fortaleza.ns.porkbun.com
maceio.ns.porkbun.com
salvador.ns.porkbun.com
GLUE RECORDS
Manage
SSL
Have Certificate
PARK DOMAIN
Park Now
URL FORWARDING
Not Set
REGISTRAR TRANSFER
Get Authorization Code
ACCOUNT TRANSFER
Push Domain
REGISTRY DNSSEC
0 records
Edit Authoritative Nameservers
studentgroup01-ft.cloud · Switch to classic view
Enter the nameservers you would like to use for your domain in the text area below. Put one nameserver per line, please.
Please note: Although we update your nameservers immediately, this change can take up to 48 hours to propagate due to DNS caching.
More Information
Nameservers (one per line)
ns1-01.azure-dns.com.
ns2-01.azure-dns.net.
ns3-01.azure-dns.org.
ns4-01.azure-dns.info.
Cancel
Save Names


Microsoft Azure
Search resources, services, and docs (G+/)
Copilot
Home
studentgroup01-ft.cloud 1780471182617 | Overview
studentgroup01-ft.cloud
DNS zone
Search
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Resource visualizer
Settings
DNS Management
Monitoring
Automation
Help
Location : Global
Subscription (move) : Azure subscription 1
Subscription ID : b747f142-550d-48da-8ce3-162f3b10457d
Recordsets : 2
Tags (edit) : StudentGroupFT01 : DNS ZONE
Name server 1
Name server 2
Name server 3
Name server 4
Get Started
Tutorials
Tools + SDK
Azure DNS is a hosting service for DNS domains that provides name resolution by using Microsoft Azure infrastructure. By hos
APIs, tools, and billing as your other Azure services.
Add DNS record sets
Begin hosting your domain in Azure DNS by adding record sets. A record set is a collection of records in a zone that have the same name and are the same type.
Add Record Sets
Import record sets from file
You have the option to import your zone directly using the import utility. This utility offers a rapid, dependable, and convenient method for transferring DNS zone data into or out of Azure DNS.
Import Record Sets
Access control
Add record set
studentgroup01-ft.cloud
Name
SGFT01
.studentgroup01-ft.cloud
Type
CNAME – Link your subdomain to another record
Alias record set
No
TTL *
3600
TTL unit
Seconds
Alias
Student-Group-01-FT-ALB-673220095.ap-s...
Add
Cancel
Give feedback


Microsoft Azure
Search resources, services, and docs (G+/)
Copilot
Home
DNS zones
studentgroup01-ft.cloud
DNS zones
Student_Group_01-FT (studentgroup01ft1.onmicr...
Create
Manage view
Name
studentgroup01-ft.cloud
Showing 1 - 1 of 1. Display count:
auto
studentgroup01-ft.cloud | Recordsets
DNS zone
Search
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Resource visualizer
Settings
DNS Management
Monitoring
Automation
Help
Add
Refresh
Delete
Give feedback
A record set is a collection of records in a zone that have the same name
that have been loaded on this page. If you don't see what you're looking
load. Learn more
Search
Fetched 3 record set(s).
Type
TTL
Value
NS
172800
ns1-01.azure-dns.
ns2-01.azure-dns.
ns3-01.azure-dns.
ns4-01.azure-dns.
SOA
3600
Email: azuredns-hostmaster.micros
Host: ns1-01.azure
dns.com.
Refresh: 3600
Retry: 300
Expire: 2419200
Minimum TTL: 30
Serial number: 1
CNAME
300
student-group-01
673220095.ap-so
2.elb.amazonaws.
WWW
studentgroup01-ft.cloud
Users
Name
www.studentgroup01-ft.cloud
Type
CNAME
Alias record set
No
TTL *
5
TTL unit
Minutes
Alias
student-group-01-ft-alb-673220095.ap-so...
Metadata
Key
Value
Apply
Discard changes
Give feedback

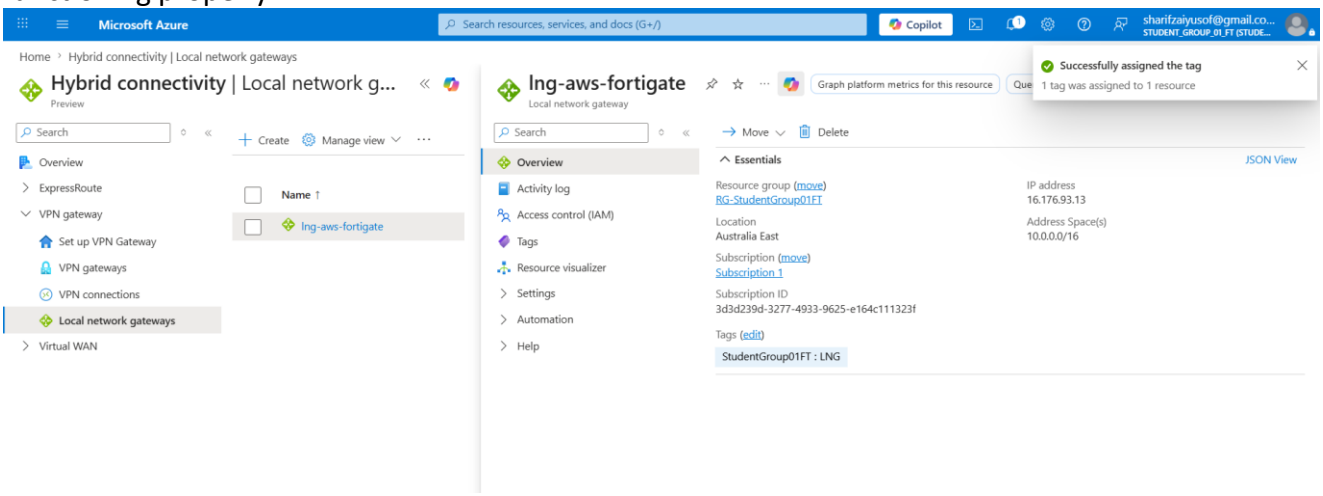

studentgroup01-ft.cloud - Micr
Student Group 01 WordPress Si
Ask Gemini
Not secure
studentgroup01-ft.cloud
+00 123 456 7890
education1245@example.com
Student Group
01 WordPress
Site
SAMPLE PAGE

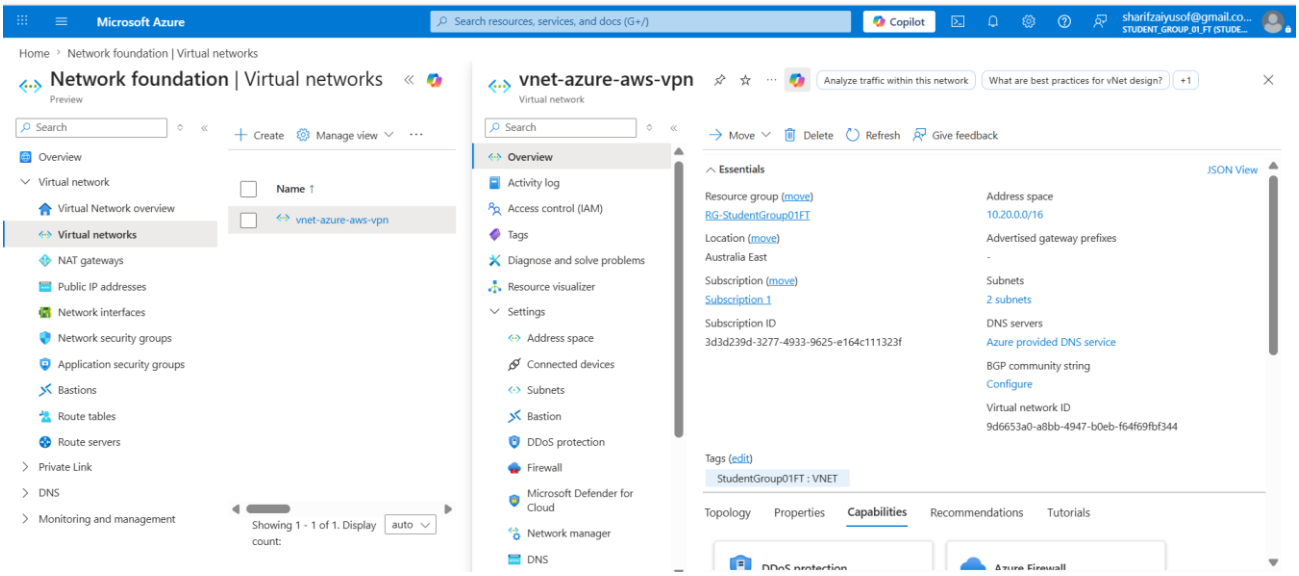

Hello world!
Posted on: June 3, 2026 - admin
Search
Search

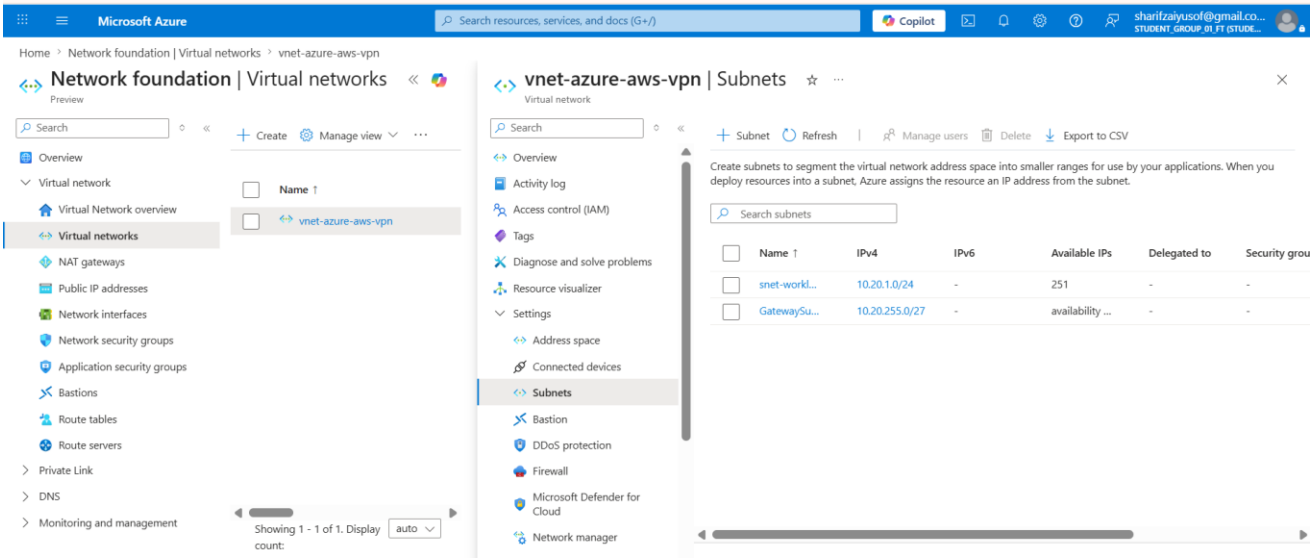
Testing Through VPN Client Services

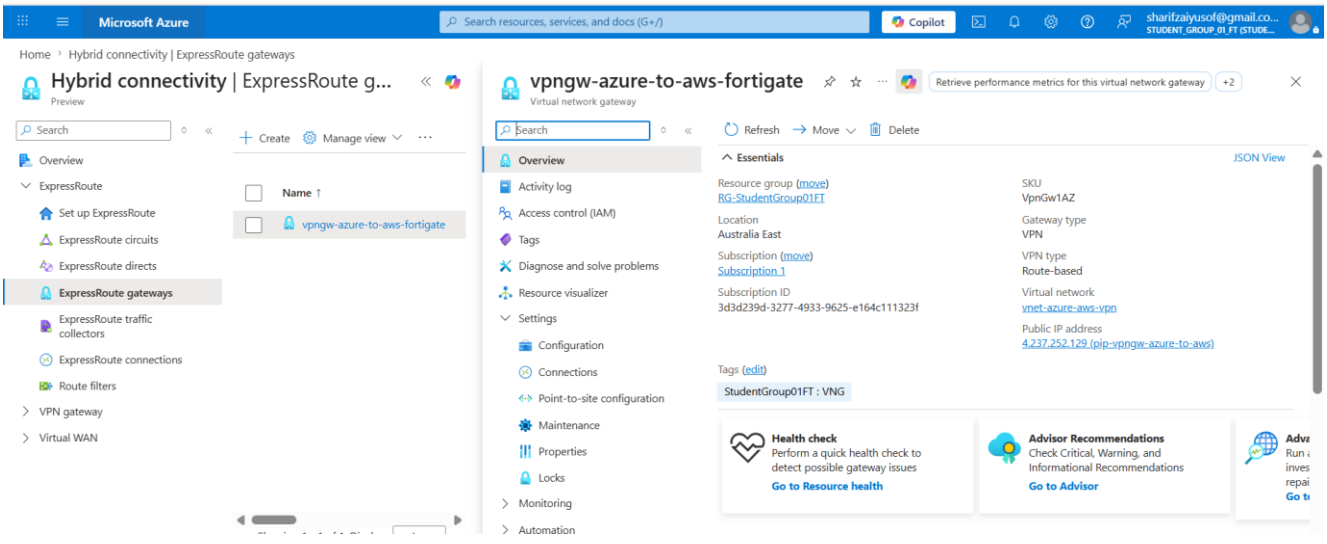
Once the complete Azure setup was done, we wanted to confirm that the hybrid connection from Azure to AWS was actually operational. The initial setup included the creation of the Virtual Network (VNet) with 2 subnets. One subnet was named s-net-workload and the second one was the required GatewaySubnet; this was reserved for the VPN gateway.
The VPN gateway itself was using the VpnGw1AZ SKU which provided zone redundant high availability. The gateway type was VPN with Route-Based Routing; this supported the IPSec config on the FortiGate appliance at AWS. A Public IP Address (4.237.252.129) was assigned to the gateway to allow it to connect to the FortiGate.
For AWS We created a Local Network Gateway (lng-aws-FortiGate) which represented the FortiGate device within Azure. The public IP address of the FortiGate (16.176.93.13) and the AWS VPC address space (10.0.0.0 /16) were added to provide to Azure what networks were available through the tunnel.
Upon configuring each end of the VPN tunnel, we created the site-to-site VPN connection "conn-Azure-to-aws-FortiGate". The Connection Type was IPsec which is the typical method of encrypting VPN tunnel traffic. Upon checking the status, it indicated that it was connected; therefore, the tunnel had been established and was transmitting data.

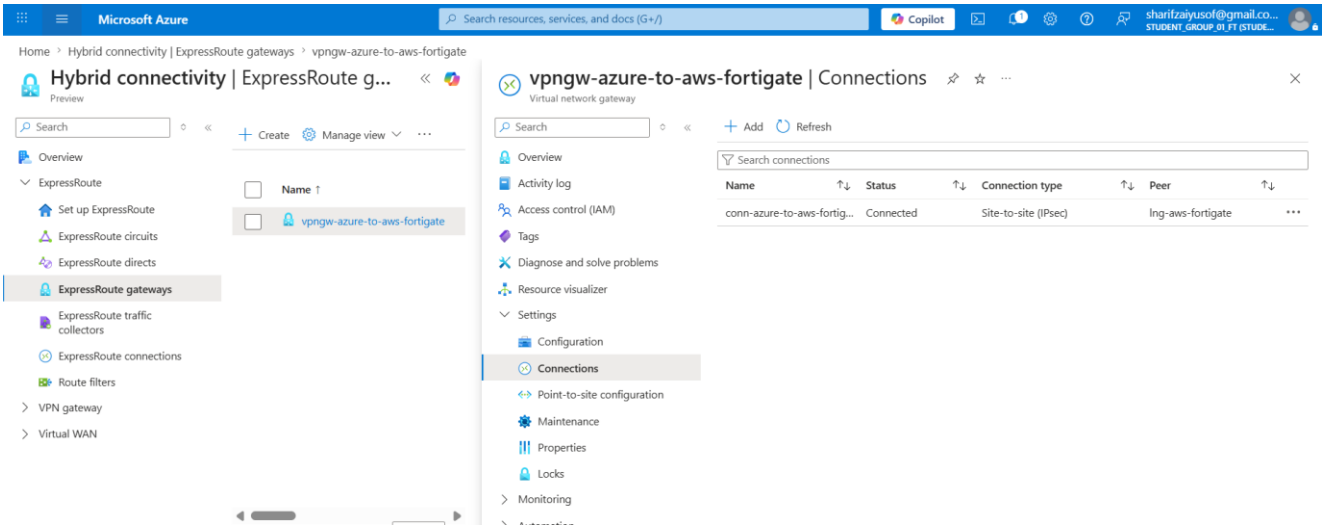
In order to test that there was full communication throughout the entire path, Geni tried pinging the private IP of the EC2 that had WordPress installed via the VPN tunnel. The ping worked which allowed me to determine that the routing had been established correctly along with both Security Groups and Network Security Groups allowing traffic to flow between them. In addition, we verified that access to WordPress was still possible via its public Domain Name using HTTPS without any issues. This allowed me to confirm that the Hybrid Network Path had been validated and functioning properly.

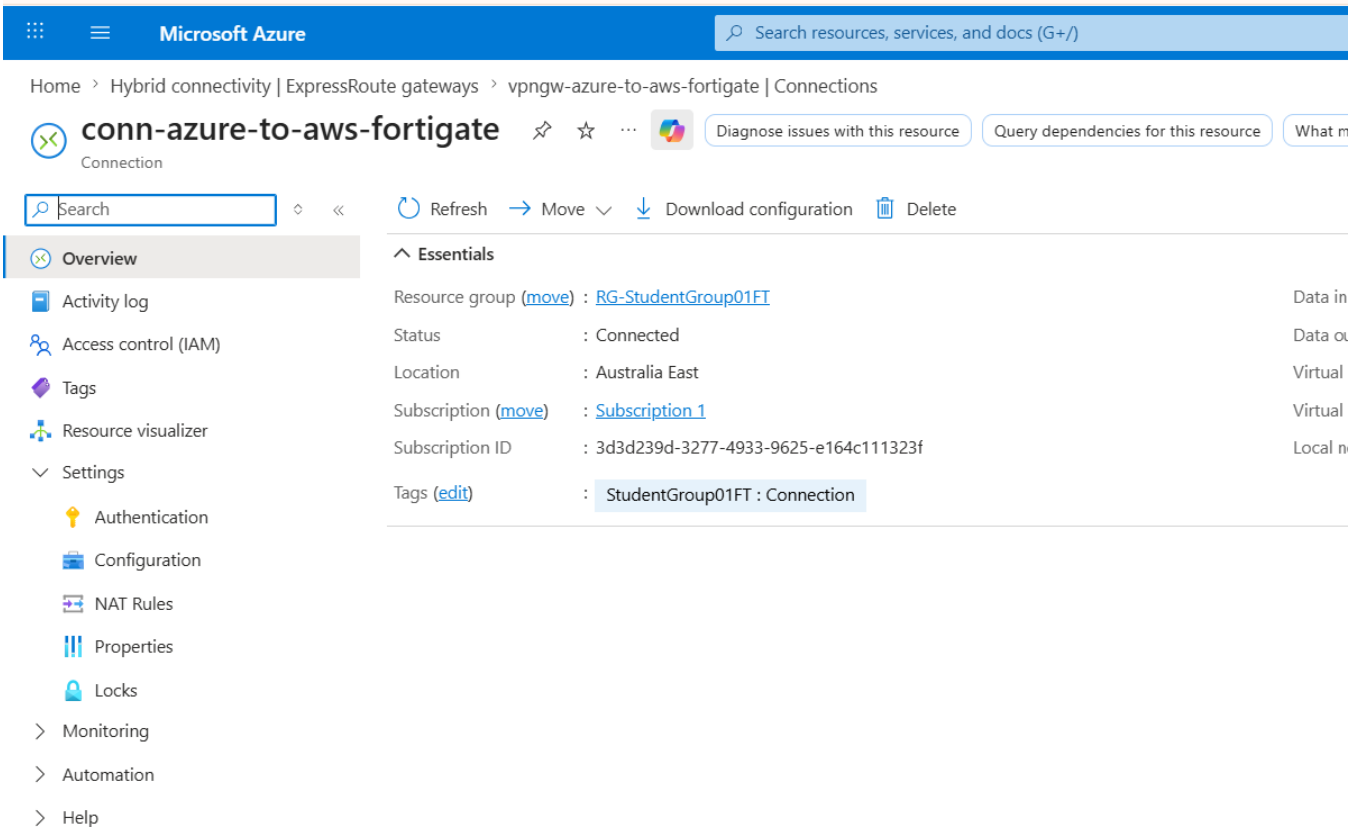



Microsoft Azure
Search resources, services, and docs (G+/)
Copilot
sharifzaiyusof@gmail.co...
STUDENT_GROUP_01_FT (STUDE...
Home › Network foundation | Virtual networks
Network foundation | Virtual networks
Preview
Search
Create
Manage view
Overview
Virtual network
Virtual Network overview
Virtual networks
NAT gateways
Public IP addresses
Network interfaces
Network security groups
Application security groups
Bastions
Route tables
Route servers
Private Link
DNS
Monitoring and management
Name
vnet-azure-aws-vpn
Showing 1 - 1 of 1. Display count:
auto
vnet-azure-aws-vpn
Virtual network
Analyze traffic within this network
What are best practices for vNet design?
+1
Move
Delete
Refresh
Give feedback
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Resource visualizer
Settings
Address space
Connected devices
Subnets
Bastion
DDoS protection
Firewall
Microsoft Defender for Cloud
Network manager
DNS
Essentials
JSON View
Resource group (move)
RG-StudentGroup01FT
Location (move)
Australia East
Subscription (move)
Subscription 1
Subscription ID
3d3d239d-3277-4933-9625-e164c111323f
Address space
10.20.0.0/16
Advertised gateway prefixes
-
Subnets
2 subnets
DNS servers
Azure provided DNS service
BGP community string
Configure
Virtual network ID
9d6653a0-a8bb-4947-b0eb-f64f69fbf344
Tags (edit)
StudentGroup01FT : VNET
Topology
Properties
Capabilities
Recommendations
Tutorials
DDoS protection
Azure Firewall


Microsoft Azure
Search resources, services, and docs (G+/)
Copilot
sharifzaiyusof@gmail.co...
STUDENT_GROUP_01_FT (STUDE...
Home › Network foundation | Virtual networks › vnet-azure-aws-vpn
Network foundation | Virtual networks
Preview
Search
Create
Manage view
Overview
Virtual network
Virtual Network overview
Virtual networks
NAT gateways
Public IP addresses
Network interfaces
Network security groups
Application security groups
Bastions
Route tables
Route servers
Private Link
DNS
Monitoring and management
Name
vnet-azure-aws-vpn
Showing 1 - 1 of 1. Display count:
auto
vnet-azure-aws-vpn | Subnets
Virtual network
Subnet
Refresh
Manage users
Delete
Export to CSV
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Resource visualizer
Settings
Address space
Connected devices
Subnets
Bastion
DDoS protection
Firewall
Microsoft Defender for Cloud
Network manager
Create subnets to segment the virtual network address space into smaller ranges for use by your applications. When you deploy resources into a subnet, Azure assigns the resource an IP address from the subnet.
Search subnets
Name
IPv4
IPv6
Available IPs
Delegated to
Security grou
snet-workl...
10.20.1.0/24
251
GatewaySu...
10.20.255.0/27
availability ...


Microsoft Azure
Search resources, services, and docs (G+/)
Copilot
sharifzaiyusof@gmail.co...
STUDENT_GROUP_01_FT (STUDE...
Home › Hybrid connectivity | ExpressRoute gateways
Hybrid connectivity | ExpressRoute g...
Preview
Search
Create
Manage view
Overview
ExpressRoute
Set up ExpressRoute
ExpressRoute circuits
ExpressRoute directs
ExpressRoute gateways
ExpressRoute traffic collectors
ExpressRoute connections
Route filters
VPN gateway
Virtual WAN
Name
vpngw-azure-to-aws-fortigate
vpngw-azure-to-aws-fortigate
Virtual network gateway
Retrieve performance metrics for this virtual network gateway
+2
Refresh
Move
Delete
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Resource visualizer
Settings
Configuration
Connections
Point-to-site configuration
Maintenance
Properties
Locks
Monitoring
Automation
Essentials
JSON View
Resource group (move)
RG-StudentGroup01FT
Location
Australia East
Subscription (move)
Subscription 1
Subscription ID
3d3d239d-3277-4933-9625-e164c111323f
SKU
VpnGw1AZ
Gateway type
VPN
VPN type
Route-based
Virtual network
vnet-azure-aws-vpn
Public IP address
4.237.252.129 (pip-vpngw-azure-to-aws)
Tags (edit)
StudentGroup01FT : VNG
Health check
Perform a quick health check to detect possible gateway issues
Go to Resource health
Advisor Recommendations
Check Critical, Warning, and Informational Recommendations
Go to Advisor





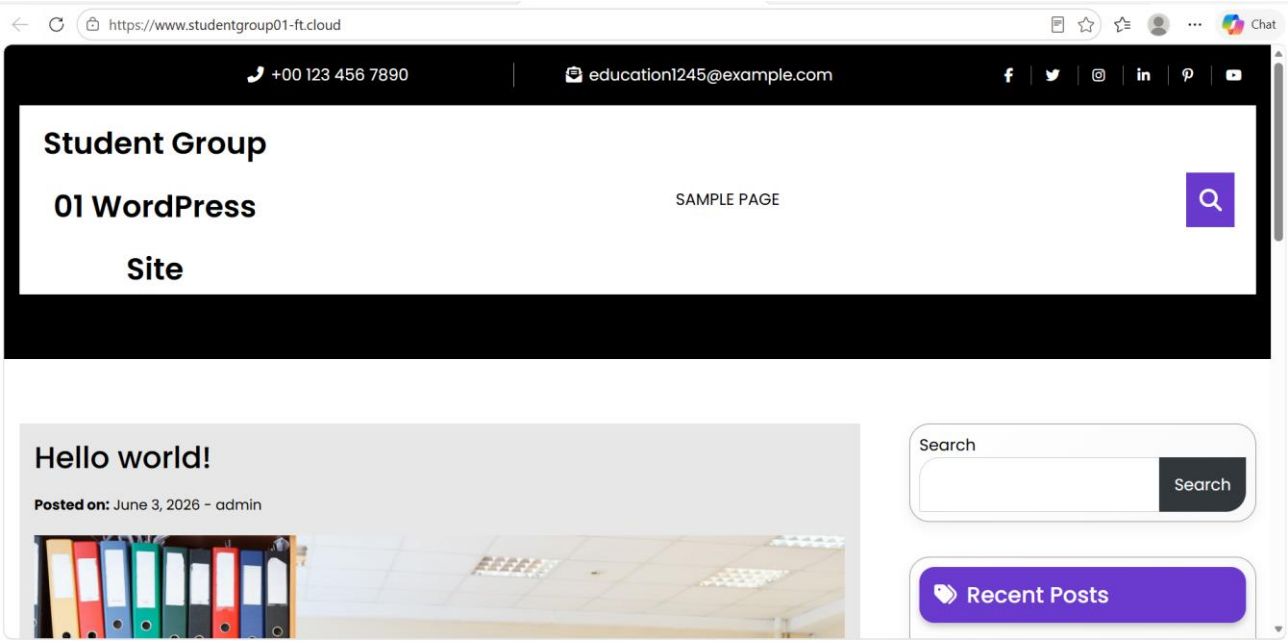
Final Outcome: Secure HTTPS Access Through Azure DNS

Ultimately, the purpose behind implementing the multi-cloud solution was for the student group's WordPress site to be available over the internet from anywhere in the world using an HTTPS connection. After waiting for the dns records to propagate through the Internet and after validation of the SSL/TLS certificate; We checked the site out by just entering the site's domain into a web browser.

the site worked properly when viewed at HTTPS://www.studentgroup01-ft.cloud. In addition, there is a padlock icon shown on the left side of the URL bar indicating that the SSL/TLS certificate is being recognized and used as expected. There are no error messages showing within the WordPress

content either. Therefore, it shows that Azure DNS was successfully resolving the domain to AWS Application Load Balancer and that the multi-cloud configuration was successful.

As planned; everything was functioning as expected. With the multi-cloud configuration, Azure will continue to manage identities and global dns resolutions and aws will continue to manage hosting of the student group's WordPress site. To users; accessing the site is transparent with a clean domain name.
The final test confirms that the WordPress website was reachable using the DNS name configured through Azure DNS and that secure HTTPS access was available. This validates the core multi-cloud requirement: Azure provides DNS integration while AWS hosts the application workload behind secure infrastructure.



Task 2: Part-C | Multi-Cloud Security Integration using AWS, Azure and Fortinet (Geni)

This section explains how the three main parts of the project connect together. In our setup, AWS hosts the WordPress workload, Azure is used for DNS, identity and access validation, and FortiGate is used as the security-focused component for firewall control and VPN/routing.

Area	What we configured	Why it mattered
AWS hosting	WordPress hosted using EC2, RDS, ELB, Auto Scaling, ACM, S3, Lambda, CloudWatch, SNS, and IAM	Provides scalable and monitored application hosting
Azure identity and governance	Azure RBAC, MFA, Azure AD tenant, subscription validation	Improves account security and controlled administration
Azure DNS integration	Azure DNS resolves the website name to the AWS load balancer endpoint	Demonstrates multi-cloud service integration
Fortinet FortiGate	Firewall/VPN/security control point between AWS and Azure	Controls and inspects inter-cloud traffic
Secure remote access	OpenVPN/VPN client testing	Reduces direct exposure of private resources
Monitoring and logs	CloudWatch, CloudTrail, Azure monitoring, FortiGate logs where available	Supports detection, troubleshooting, and incident response
Encryption	HTTPS/ACM and AWS encryption services	Protects data in transit and supports data-at-rest protection

The evidence for this integration is not shown in one single console screen because each product has its own portal. AWS evidence is shown in Task 2 Part A, Azure evidence is shown in Task 2 Part B, and FortiGate evidence is shown in the VPN/FortiGate section. Together, these screenshots show how the final multi-cloud setup was built.

The main security idea in this project was to avoid giving open access everywhere. AWS Security Groups, route tables, NACL review, VPN access and FortiGate rules were used to control network access. IAM roles were used in AWS, while RBAC and MFA were used in Azure. Monitoring was also included so issues can be found and investigated.

Further details are shown in the screenshots above. This section is kept short because the actual configuration evidence is already included in the AWS, Azure and FortiGate parts of the report.

One limitation we noticed is that multi-cloud work is not always easy to prove in one screenshot. The evidence is spread across AWS, Azure and FortiGate, but together it shows how each part contributed to the final design.

Requirement Area	Evidence Location
AWS workload hosting	Task 2 Part A: VPC, EC2, RDS, S3, Lambda, ELB, Auto Scaling, ACM
Azure integration	Task 2 Part B: Azure subscription, RBAC/MFA, Azure AD, Azure DNS
Fortinet security	Task 2 Part A FortiGate/VPN section and Task 2 Part C
Monitoring and logging	CloudWatch, CloudTrail, SNS, Azure Monitor and FortiGate log references

Backup and recovery	S3 backup, Lambda snapshot automation, RDS snapshots
Incident handling	Task 3 Part B: EC2 Security Group misconfiguration

Task 3: Risk Assessment and Security Incident Analysis

Task 3: Part A | Risk Assessment and Mitigation Strategy (Geni)

In this section, I identified the main risks we noticed in the multi-cloud design. The risks cover AWS hosting, Azure DNS and identity, FortiGate controls, VPN connectivity, monitoring and backup. The aim of this risk assessment is to show what can go wrong and how we can reduce the impact. I used Low, Medium and High ratings for likelihood and impact, and the mitigation steps are based on what we implemented or reviewed in the project.

Risk ID	Risk Area	Description of Risk	Likelihood	Impact	Mitigation Strategy
R1	AWS IAM	IAM roles or policies may be over-permissive, allowing users or services to access resources beyond their required responsibilities.	Medium	High	Apply least-privilege IAM policies, avoid shared credentials, use IAM roles instead of access keys, enable MFA for administrator access, and review permissions before final submission.
R2	AWS Security Groups	A Security Group may be misconfigured to allow SSH, database, or management ports from 0.0.0.0/0, exposing resources to the internet.	Medium	High	Restrict SSH/admin access to VPN subnet or approved administrator IPs only, review inbound rules, remove unused ports, and monitor changes using CloudTrail and CloudWatch.
R3	AWS RDS Database	The RDS database may be publicly accessible or allow MySQL access from unapproved sources.	Low	High	Place RDS in private subnet, disable public accessibility, allow MySQL only from the application Security Group, enable encryption, and take regular snapshots.
R4	Data Protection	Backup data or database storage may be exposed if encryption or access control is not configured correctly.	Medium	High	Enable S3 server-side encryption, RDS encryption, restrict S3 bucket public access, use IAM-controlled access, and validate backup permissions.
R5	Azure Identity and RBAC	Azure users may be given excessive permissions or MFA may not be enforced,	Medium	High	Enable MFA, assign RBAC roles based on job responsibility, avoid owner-level access unless

		increasing the risk of account compromise.			required, and review access assignments.
R6	Azure DNS	Incorrect Azure DNS records may point users to the wrong endpoint or make the WordPress site unavailable.	Medium	Medium	Validate DNS records, check name server configuration, test HTTPS access from an external client, and document final DNS configuration.
R7	Fortinet FortiGate Policy	Firewall policies may be too open or incorrectly ordered, allowing unapproved traffic between AWS and Azure.	Medium	High	Apply deny-by-default approach, allow only required traffic, document firewall rules, enable FortiGate logging, and test allowed/blocked traffic paths.
R8	VPN Exposure	VPN or OpenVPN access may be exposed to unauthorised users or weak authentication, allowing attackers to target private resources.	Medium	High	Restrict VPN access to approved users, use strong authentication, limit VPN Security Group rules, monitor VPN login attempts, and remove unused accounts.
R9	Monitoring Gaps	Security events may not be detected if CloudWatch, CloudTrail, Azure Monitor, or FortiGate logs are not enabled or reviewed.	Medium	High	Enable CloudWatch alarms, CloudTrail audit logging, Azure Monitor checks, FortiGate logs, and SNS notifications for important events.
R10	Backup and Recovery Failure	Lambda snapshot automation or S3 backup storage may fail, leaving the environment without a reliable recovery point.	Low	High	Test Lambda snapshot execution, review CloudWatch logs, verify S3 backup storage, and confirm snapshots are created successfully.
R11	Multi-Cloud Dependency	Failure in one cloud component, such as Azure DNS or FortiGate routing, can affect access to AWS-hosted services.	Medium	High	Document dependency paths, test AWS access directly and through Azure DNS, validate FortiGate/VPN paths, and maintain recovery steps for each component.
R12	Human Error / Change Management	Team members may accidentally modify firewall, IAM, DNS, or	Medium	High	Use peer review for configuration changes, record major changes,

		Security Group settings without review.			review screenshots before submission, and apply least privilege to reduce accidental impact.

Shared Risk and Dependency Discussion

Because the project uses AWS, Azure and Fortinet together, some risks are shared. For example, the WordPress site is hosted in AWS, but access also depends on Azure DNS. If the DNS record is wrong, users may not reach the site even if AWS is working correctly. In the same way, FortiGate or VPN routing issues can affect traffic between the cloud environments.

Access control is another shared risk. AWS IAM controls the AWS side, while Azure RBAC and MFA control the Azure side. If either side is too open, the whole design becomes weaker. Monitoring is also split across different tools such as CloudWatch, CloudTrail, Azure Monitor and FortiGate logs, so they need to be checked together.

The mitigation approach is based on having more than one layer of protection. Network access is controlled using Security Groups, NACL review, VPN and FortiGate policies. Identity is controlled using IAM roles, RBAC and MFA. Data protection is handled using encryption and restricted S3/RDS access. Backup and monitoring support recovery if something goes wrong.

Task 3: Part B | Security Incident Analysis (Hari)

Selected Incident: EC2 Security Group Misconfiguration Allowing Public SSH Access

During our implementation and testing stage, we found a security configuration issue in the AWS EC2 environment used for the WordPress deployment. One EC2 Security Group rule allowed SSH access on port 22 from 0.0.0.0/0, meaning the instance could receive SSH connection attempts from any public IP address.

I treated this as a security issue because management access should not be open to the public internet. In the intended design, SSH access should only be available through a controlled path such as VPN, FortiGate access, or a trusted IP address. The EC2 instance still required key-based authentication, but the open SSH rule still increased the attack surface.

We found this issue while reviewing AWS Security Group rules and testing evidence. The response was simple but important: identify the risky rule, correct the inbound access, review related controls, and record the lesson learned.

Incident Detection and Response Timeline

Time	Event / Activity	Evidence / Tool Used	Action Taken
05:30 PM	EC2 Security Group rule was found allowing SSH port 22 from 0.0.0.0/0.	AWS EC2 Security Group inbound rules	The rule was identified as risky because SSH was open to the public internet.
05:45 PM	Team reviewed whether SSH access was required from the internet.	AWS EC2 configuration and project design	Confirmed that public SSH access was not required for the final secure design.
06:00 PM	The Security Group rule was updated.	AWS EC2 Security Group console	Public SSH access was removed or restricted to a trusted access path.
06:15 PM	VPN and FortiGate access design was reviewed.	OpenVPN/FortiGate configuration evidence	Confirmed that administrative access should be handled through a controlled secure path.
06:30 PM	Related access controls were reviewed.	IAM, Security Groups, RDS Security Group, Azure RBAC/MFA	Checked whether similar risky access rules existed in other parts of the project.
06:45 PM	Monitoring and logging sources were reviewed for incident analysis.	CloudTrail, CloudWatch, Azure Monitor, FortiGate logs where available	Identified what evidence would support detection and troubleshooting.
07:00 PM	Findings and improvements were documented.	Project report and screenshots	Added lessons learned and prevention actions to the report.

Roles and Responsibilities

Role	Team Member	Responsibility During the Incident
AWS Implementation Lead	Geni Bahar	Reviewed AWS EC2 Security Group rules, corrected the risky inbound rule, checked IAM and EC2 exposure, and documented the AWS-side response.
Azure Implementation Lead	Yusof Sharifzai	Verified that Azure DNS, Azure AD, RBAC, and MFA were not directly affected by the AWS Security Group issue.
Project Coordinator / Testing Support	Hari Tamang	Supported screenshot collection, project documentation, and verification of testing evidence.
Fortinet / VPN Review	Geni Bahar with group support	Reviewed the intended FortiGate/VPN access path and confirmed that management access should not rely on public SSH exposure.

Detection and Investigation

We detected the issue while manually checking the EC2 Security Group rules. The inbound SSH rule from 0.0.0.0/0 was risky because any public IP address could try to connect to the EC2 instance. For the investigation, we mainly checked three things:

1. Was public SSH access required for the project?
2. Was the EC2 instance still protected by authentication?
3. Was similar risky access rules present in other services?

We confirmed that public SSH access was not needed for the final design. Access should be restricted to the VPN path, FortiGate-controlled access, or a trusted IP address. The instance still required key-based authentication, so this was mainly an exposure issue rather than confirmed unauthorised access. We also checked related areas such as IAM permissions, RDS Security Group rules, VPN access, Azure RBAC/MFA and FortiGate policy design.

Logging and Monitoring Evidence

The screenshots and configuration evidence in Task 2 support this incident analysis because they show the Security Group, CloudWatch, CloudTrail, VPN/FortiGate, and Azure control areas that were reviewed during testing.

Evidence Source	What Was Checked	Relevance to Incident
AWS EC2 Security Group	Inbound SSH rule and source range	Confirmed that port 22 was open from 0.0.0.0/0.
AWS CloudTrail	Security Group change activity	Useful for identifying when and how Security Group rules were changed.
AWS CloudWatch	EC2 monitoring and alerting capability	Useful for monitoring unusual activity or operational issues.
EC2 System Logs	Server access/authentication logs where available	Useful for checking whether login attempts occurred.
FortiGate Logs / Policies	VPN/firewall access path and traffic control	Useful for confirming that management traffic should be controlled through firewall/VPN rules.

Azure Monitor / Azure AD Review	Azure DNS, identity, and access status	Confirmed that the issue was limited to AWS EC2 access control and did not directly affect Azure identity or DNS.

Response Actions

First, the SSH rule was removed or restricted so port 22 was no longer open to 0.0.0.0/0. This immediately reduced the exposure of the EC2 instance.
Second, we confirmed the correct way to access the environment. The VPN and FortiGate design should be used for controlled access rather than direct public SSH.
Third, we reviewed related controls such as IAM roles, EC2 Security Groups, RDS Security Group rules, VPN access, Azure RBAC/MFA and FortiGate policy design to check if the same type of issue existed anywhere else.
Finally, we documented the issue as part of the report because it was a good example of how a small configuration mistake can weaken a cloud design.

Recovery Plan

Immediate Recovery:

The public SSH exposure was corrected by removing or restricting the inbound rule for port 22. SSH access was limited to a trusted access path such as VPN, FortiGate-controlled access, or an approved IP address.

Short-Term Recovery:

All Security Groups were reviewed to identify any other risky inbound rules. IAM permissions were checked to ensure that AWS services used least-privilege access. RDS access rules were reviewed to confirm that the database was not publicly exposed.

Long-Term Recovery:

The team should use a simple change review process before modifying Security Groups, firewall policies, IAM permissions, DNS records, or VPN settings. Security Group rules should be checked again before final submission. Monitoring tools such as CloudTrail, CloudWatch, Azure Monitor, and FortiGate logs should be used where possible to support future troubleshooting and incident analysis.

Lessons Learned

This incident showed me that even a small Security Group mistake can create a serious weakness. Opening SSH to the public internet may happen during testing, but it should not remain in the final setup.
We also learned that cloud security has to be checked at multiple layers. Security Groups control AWS resource access, IAM controls permissions, FortiGate and VPN control secure access paths, and Azure RBAC/MFA protects Azure administration. If one layer is too open, the whole solution becomes weaker.
The main lesson is that access rules should always be reviewed before final submission or production deployment. Management access should only be allowed through trusted and controlled paths.

Conclusion

The issue was handled by correcting the EC2 Security Group rule, restricting SSH access, and reviewing the related AWS, Azure and FortiGate controls. It was a useful project lesson about checking access rules before final submission.

Final Verification Summary

Verification Item	**Result**
AWS VPC and subnet design	Completed
EC2 WordPress deployment	Completed
RDS database setup	Completed
Load Balancer and Auto Scaling	Completed
S3/Lambda backup design	Completed
CloudWatch/SNS monitoring	Completed
Azure DNS and HTTPS testing	Completed
FortiGate/VPN security integration	Completed
Risk assessment	Completed
Security incident analysis	Completed

References

- Amazon Web Services. (n.d.). *Amazon CloudWatch*. https://aws.amazon.com/cloudwatch/
- Amazon Web Services. (n.d.). *Amazon EC2 security groups for your EC2 instances*. https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-security-groups.html
- Amazon Web Services. (n.d.). *Amazon RDS user guide*. https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Welcome.html
- Amazon Web Services. (n.d.). *Amazon S3 user guide*. https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html
- Amazon Web Services. (n.d.). *Amazon Virtual Private Cloud documentation*. https://docs.aws.amazon.com/vpc/
- Amazon Web Services. (n.d.). *AWS Auto Scaling user guide*. https://docs.aws.amazon.com/autoscaling/plans/userguide/what-is-aws-auto-scaling.html
- Amazon Web Services. (n.d.). *AWS Certificate Manager user guide*. https://docs.aws.amazon.com/acm/latest/userguide/acm-overview.html
- Amazon Web Services. (n.d.). *AWS CloudTrail user guide*. https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-user-guide.html
- Amazon Web Services. (n.d.). *AWS Identity and Access Management documentation*. https://docs.aws.amazon.com/iam/
- Amazon Web Services. (n.d.). *AWS Lambda developer guide*. https://docs.aws.amazon.com/lambda/latest/dg/welcome.html
- Amazon Web Services. (n.d.). *Elastic Load Balancing user guide*. https://docs.aws.amazon.com/elasticloadbalancing/latest/userguide/what-is-load-balancing.html
- Amazon Web Services. (n.d.). *Getting started with AWS*. https://aws.amazon.com/getting-started/
- Amazon Web Services. (n.d.). *Security pillar: AWS Well-Architected Framework*. https://docs.aws.amazon.com/wellarchitected/latest/security-pillar/welcome.html
- Fortinet. (n.d.). *FortiGate administration guide*. https://docs.fortinet.com/document/fortigate/latest/administration-guide
- Joint Task Force. (2020). *Security and privacy controls for information systems and organizations* (NIST Special Publication 800-53 Revision 5). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-53r5
- Microsoft. (n.d.). *Azure DNS documentation*. https://learn.microsoft.com/en-us/azure/dns/
- Microsoft. (n.d.). *Azure role-based access control documentation*. https://learn.microsoft.com/en-us/azure/role-based-access-control/
- Microsoft. (n.d.). *What is Azure role-based access control (Azure RBAC)?* https://learn.microsoft.com/en-us/azure/role-based-access-control/overview