🛡️ Project 1: Secure Application Deployment on AWS (Security Pillar)

📌 Executive Summary

In this project, I acted as a Solutions Architect for **SecureCart**, an e-commerce platform that was initially launched with a focus on speed over security. The original infrastructure left critical resources like **EC2, RDS, and S3** directly exposed to the public internet.

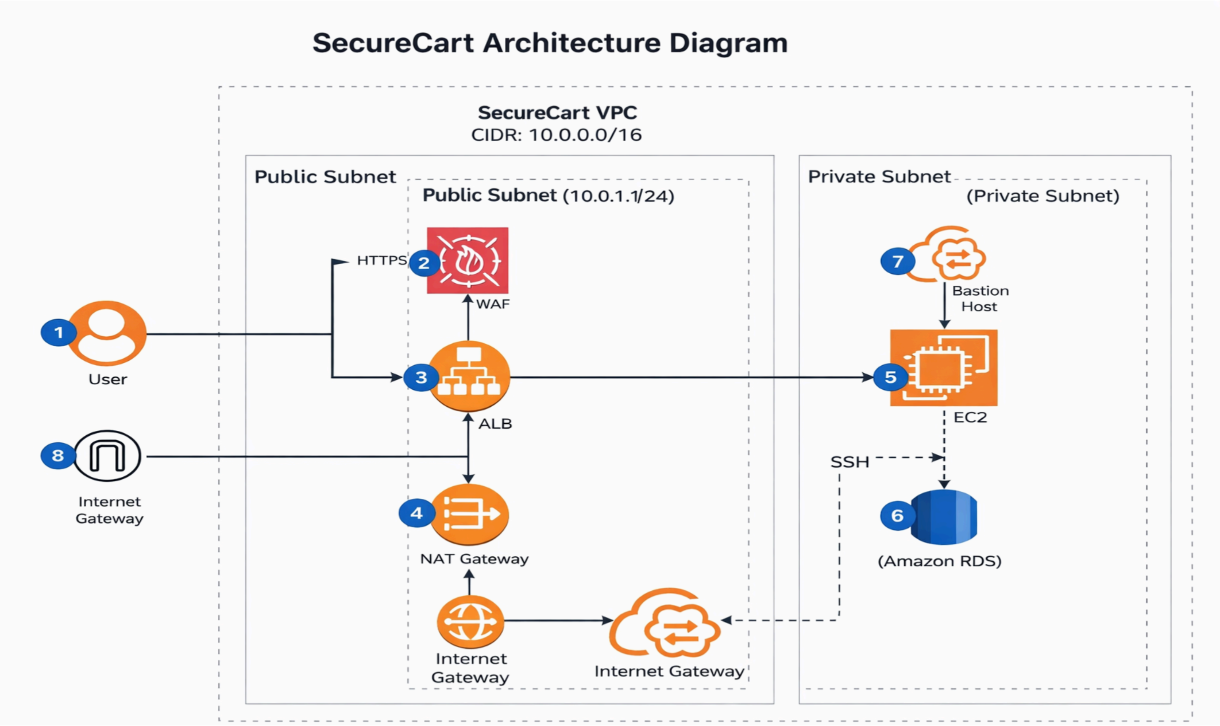
My objective was to completely redesign this vulnerable environment into a **production-ready, enterprise-grade architecture** that follows the AWS Well-Architected Framework's Security Pillar.

Key Transformations

- **Network Isolation:** Moved application and database layers from public to private subnets.
- **Controlled Access:** Implemented a Bastion Host and "chained" Security Groups to enforce the Principle of Least Privilege.
- **Perimeter Defense:** Added an Application Load Balancer (ALB) and AWS WAF to filter malicious web traffic.

🏗️ Architecture Diagram

This diagram represents the final state of the SecureCart environment. Traffic flows from the Internet Gateway through the WAF and ALB into the private application layer.

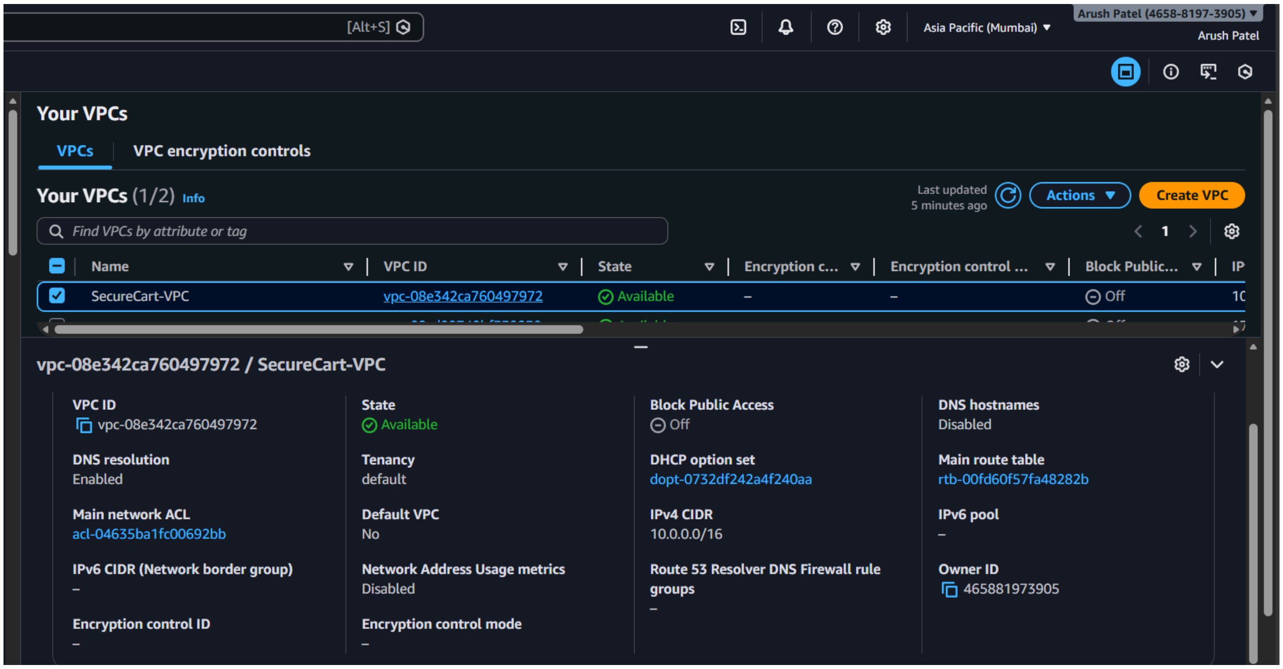


🛠️ Step-by-Step Implementation Guide

Phase 1: Building the Secure Network (The Moat)

What I did: Created a custom VPC (`SecureCart-VPC`) with segmented public and private subnets.

- **Step 1: VPC Creation**
    - **Action:** Deployed a VPC with CIDR `10.0.0.0/16`.
    - **Why:** Default VPCs are often too open for production; a custom VPC allows for granular control over IP ranges and routing.



Your VPCs

Your VPCs (1/2) Info

☑	Name	VPC ID	State	Encryption c...	Encryption control ...	Block Public...	IP
☑	SecureCart-VPC	vpc-08e342ca760497972	✓ Available	–	–	⊖ Off	10

vpc-08e342ca760497972 / SecureCart-VPC

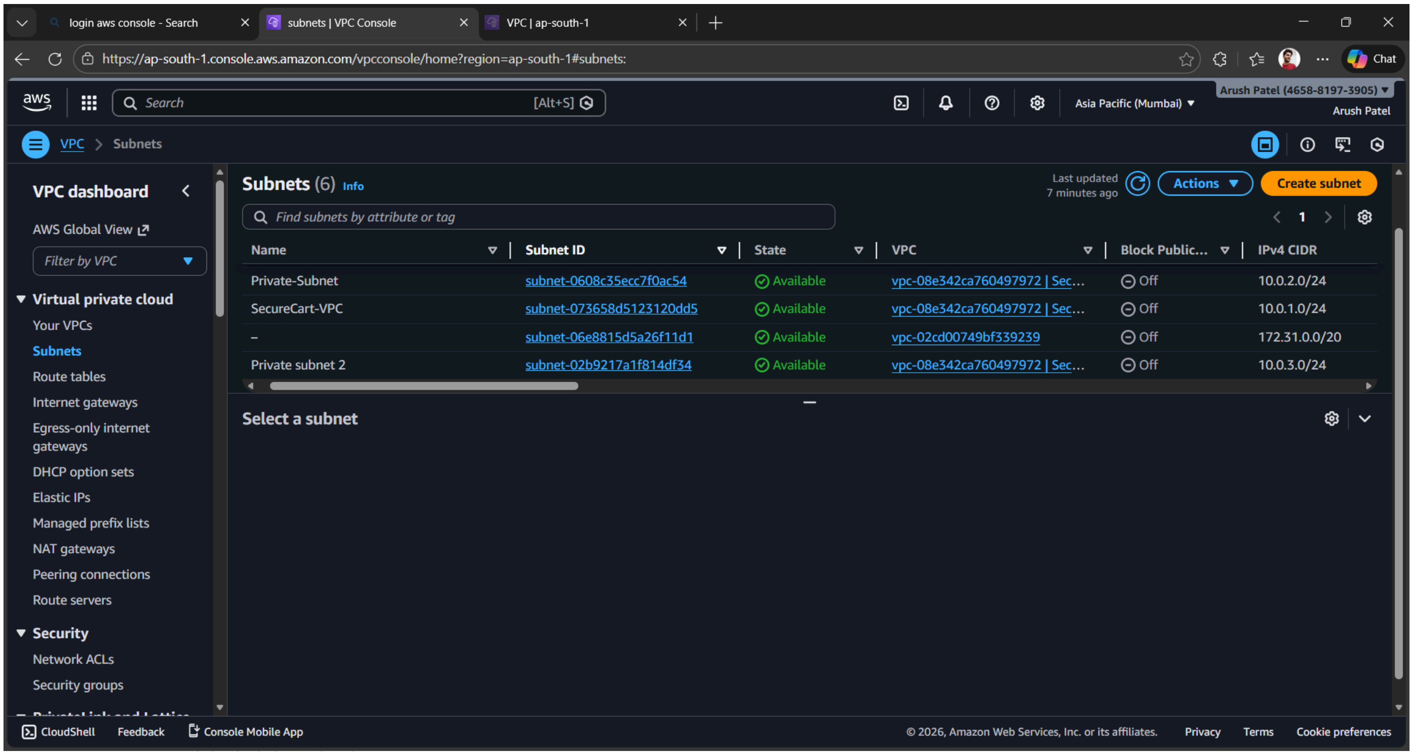
VPC ID vpc-08e342ca760497972	**State** ✓ Available	**Block Public Access** ⊖ Off	**DNS hostnames** Disabled
DNS resolution Enabled	**Tenancy** default	**DHCP option set** dopt-0732df242a4f240aa	**Main route table** rtb-00fd60f57fa48282b
Main network ACL acl-04635ba1fc00692bb	**Default VPC** No	**IPv4 CIDR** 10.0.0.0/16	**IPv6 pool** –
IPv6 CIDR (Network border group) –	**Network Address Usage metrics** Disabled	**Route 53 Resolver DNS Firewall rule groups** –	**Owner ID** 465881973905
Encryption control ID –	**Encryption control mode** –		

Services → VPC → Your VPCs → Create VPC

- Name tag: `SecureCart-VPC`
- IPv4 CIDR block: `10.0.0.0/16`
- Tenancy: Default

- **Step 2: Subnet Segmentation**
    - **Action:** Created a **Public Subnet** (`10.0.1.0/24`) for internet-facing resources and a **Private Subnet** (`10.0.2.0/24`) for sensitive data.
    - **Why:** To ensure that if a public-facing component is compromised, the attacker still doesn't have a direct path to the data layer. This follows the **Principle of**

Least Privilege at the network layer, ensuring frontend and backend assets are separated.



Subnets (6) Info

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
Private-Subnet	subnet-0608c35ecc7f0ac54	Available	vpc-08e342ca760497972 | Sec...	Off	10.0.2.0/24
SecureCart-VPC	subnet-073658d5123120dd5	Available	vpc-08e342ca760497972 | Sec...	Off	10.0.1.0/24
–	subnet-06e8815d5a26f11d1	Available	vpc-02cd00749bf339239	Off	172.31.0.0/20
Private subnet 2	subnet-02b9217a1f814df34	Available	vpc-08e342ca760497972 | Sec...	Off	10.0.3.0/24

VPC → Subnets → Create subnet

Subnet name: Public-Subnet
Availability Zone:
CIDR block: 10.0.2.0/24

VPC: SecureCart-VPC

Subnet name: Private-Subnet
Availability Zone:
CIDR block: 10.0.1.0/24

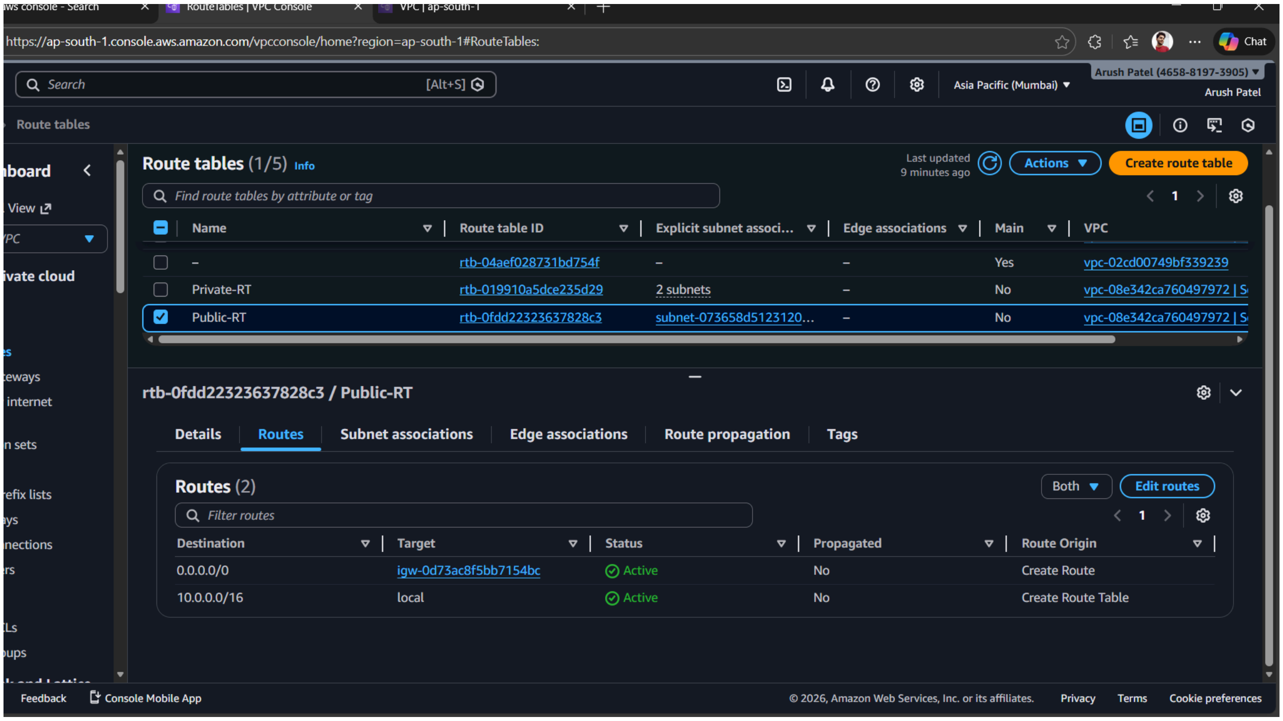
Step 3: Create & Attach Internet Gateway (IGW)

- **How:** Created `SecureCart-IGW` and used the "Attach to VPC" action to link it to `SecureCart-VPC`.
- **Why:** An Internet Gateway is the "door" that allows communication between the VPC and the public internet. Without this, even your public resources would be unreachable.
- **The "Why" (Security Logic):** By attaching it at the VPC level but only routing the *Public Subnet* to it, we control exactly which instances are "allowed" to see the outside world.

VPC → Internet Gateways → Create internet gateway
Name: SecureCart-IGW
👉 Create
Attach it : Select IGW
Actions → Attach to VPC
Select SecureCart-VPC 👉 Attach

Step 4: Configure Public Route Tables

- **How:** Created `Public-RT` and added a route for `0.0.0.0/0` targeting the `SecureCart-IGW`. Then, I performed a "Subnet Association" to link it specifically to the `Public-Subnet`.
- **Why:** A route table acts as a GPS for data packets. By pointing `0.0.0.0/0` (all internet traffic) to the IGW, we officially make the Public Subnet "Public".
- **The "Why" (Security Logic):** We intentionally leave the Private Subnet **off** this route table. This ensures that even if a hacker knows the IP of your database, there is no physical "road" for them to reach it from the internet.



https://ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#RouteTables:

Route tables (1/5)

Last updated 9 minutes ago

Name	Route table ID	Explicit subnet associ...	Edge associations	Main	VPC
–	rtb-04aef028731bd754f	–	–	Yes	vpc-02cd00749bf339239
Private-RT	rtb-019910a5dce235d29	2 subnets	–	No	vpc-08e342ca760497972 | S
Public-RT	rtb-0fdd22323637828c3	subnet-073658d5123120...	–	No	vpc-08e342ca760497972 | S

rtb-0fdd22323637828c3 / Public-RT

Routes (2)

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	igw-0d73ac8f5bb7154bc	✓ Active	No	Create Route
10.0.0.0/16	local	✓ Active	No	Create Route Table

© 2026, Amazon Web Services, Inc. or its affiliates.

👉 VPC → Route Tables → Create route table
Name: Public-RT
VPC: SecureCart-VPC
👉 Create
Add Internet Route
Select Public-RT
Routes tab → Edit routes → Add route
Destination: 0.0.0.0/0
Target: Internet Gateway → SecureCart-IGW
👉 Save changes
Associate Public Subnet
Subnet associations → Edit
Select Public-Subnet
👉 Save

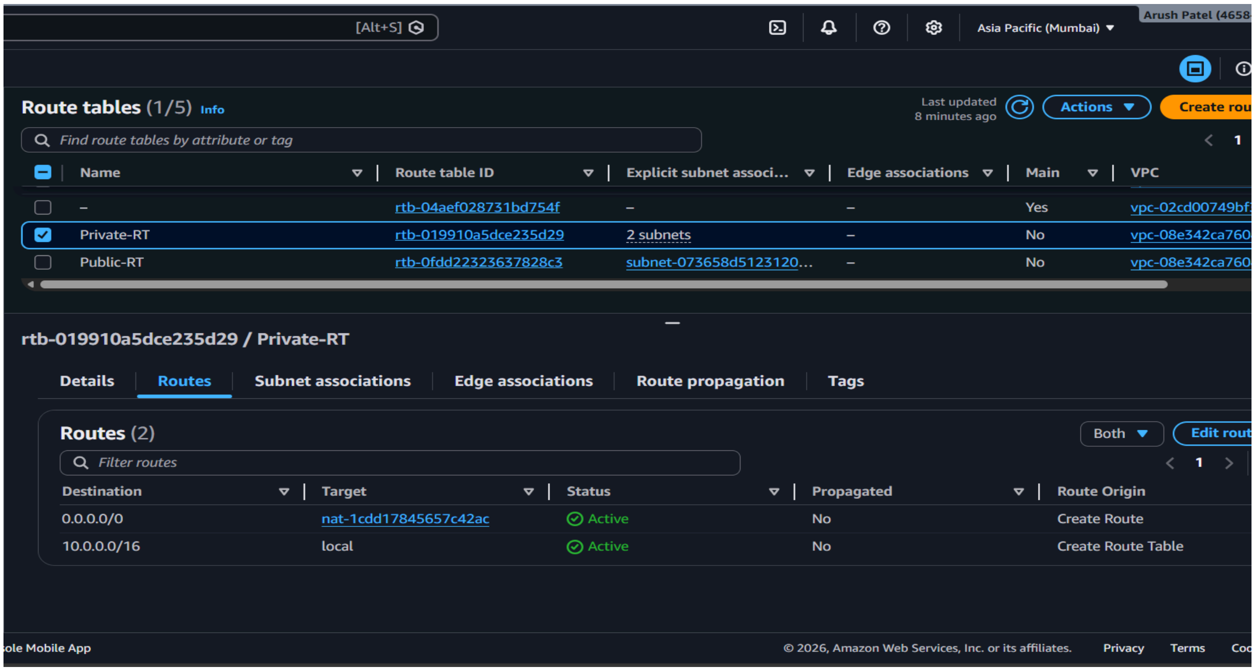
Step 5: Create NAT Gateway (The One-Way Valve)

- **How:** Navigated to **VPC > NAT Gateways**, selected the **Public Subnet**, and allocated an **Elastic IP**.
- **Why:** Instances in a private subnet cannot be reached *from* the internet, but they often need to reach *out* to the internet to download security patches, OS updates, or software dependencies.
- **The "Why" (Security Logic):** A NAT (Network Address Translation) Gateway acts like a one-way valve. It allows outbound traffic from your private instances but prevents the public internet from initiating a connection back into them. Placing it in the **Public Subnet** is mandatory because it needs its own path to the Internet Gateway to fulfill those requests.

  👉 VPC → NAT Gateways → Create NAT Gateway
  Subnet: Public-Subnet
  Connectivity type: Public
  Elastic IP → Allocate Elastic IP
  👉 Create NAT Gateway
  ⏳ Wait until status = Available

Step 6: Configure Private Route Table

- **How:** Created a new route table named `Private-RT`. I added a route for `0.0.0.0/0` (all traffic) and set the **Target** as the **NAT Gateway** created in Step 5. Finally, I associated this route table with the `Private-Subnet`.
- **Why:** Without this, the private subnet has no "map" for internet-bound traffic. By pointing the route to the NAT Gateway instead of an Internet Gateway, we maintain the privacy of the subnet.
- **The "Why" (Security Logic):** This creates a "Protected Egress" environment. Unlike the Public Subnet (which points to the IGW), the Private Subnet has no direct entry point for hackers. It is a "stub" network where traffic can only leave, not enter uninvited.



Route tables (1/5)

	Name	Route table ID	Explicit subnet associ...	Edge associations	Main	VPC
□	–	rtb-04aef028731bd754f	–	–	Yes	vpc-02cd00749bf
☑	Private-RT	rtb-019910a5dce235d29	2 subnets	–	No	vpc-08e342ca760
□	Public-RT	rtb-0fdd22323637828c3	subnet-073658d5123120...	–	No	vpc-08e342ca760

rtb-019910a5dce235d29 / Private-RT

Details | Routes | Subnet associations | Edge associations | Route propagation | Tags

Routes (2)

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	nat-1cdd17845657c42ac	✓ Active	No	Create Route
10.0.0.0/16	local	✓ Active	No	Create Route Table

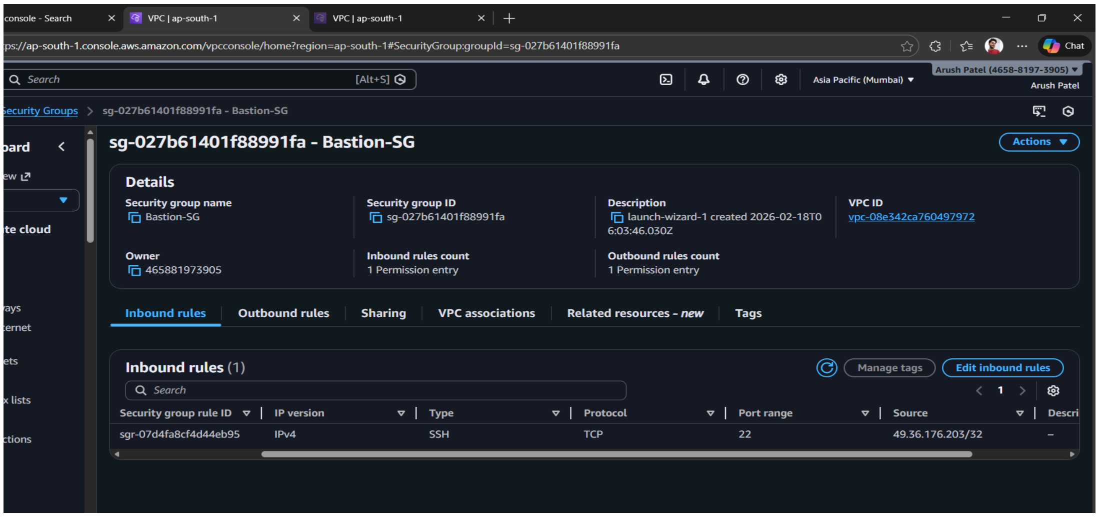
© 2026, Amazon Web Services, Inc. or its affiliates.

Name: Private-RT
VPC: SecureCart-VPC
👉 Create
Add NAT Route
Routes → Edit routes → Add route
Destination: 0.0.0.0/0
Target: NAT Gateway → Select created NAT
👉 Save
Associate Private Subnet
Subnet associations → Edit
Select Private-Subnet
👉 Save

Step 7: Launch Bastion Host

- **Action:** Launched an EC2 in the Public Subnet with a Public IP.
- **Why:** To manage private servers without exposing them. The Bastion Host acts as the single, hardened entry point for administrators.

👉 Services → EC2 → Instances → Launch instance
Name: Bastion-Host
AMI: Amazon Linux 2
Instance type: t2.micro
Key pair: Create/select key
Network Settings
VPC: SecureCart-VPC
Subnet: Public-Subnet
Auto-assign public IP: Enable
Security Group
Name: Bastion-SG
SSH (22) → My IP only
👉 Launch instance



sg-027b61401f88991fa - Bastion-SG

Details

Security group name: Bastion-SG
Security group ID: sg-027b61401f88991fa
Description: launch-wizard-1 created 2026-02-18T06:03:46.030Z
VPC ID: vpc-08e342ca760497972
Owner: 465881973905
Inbound rules count: 1 Permission entry
Outbound rules count: 1 Permission entry

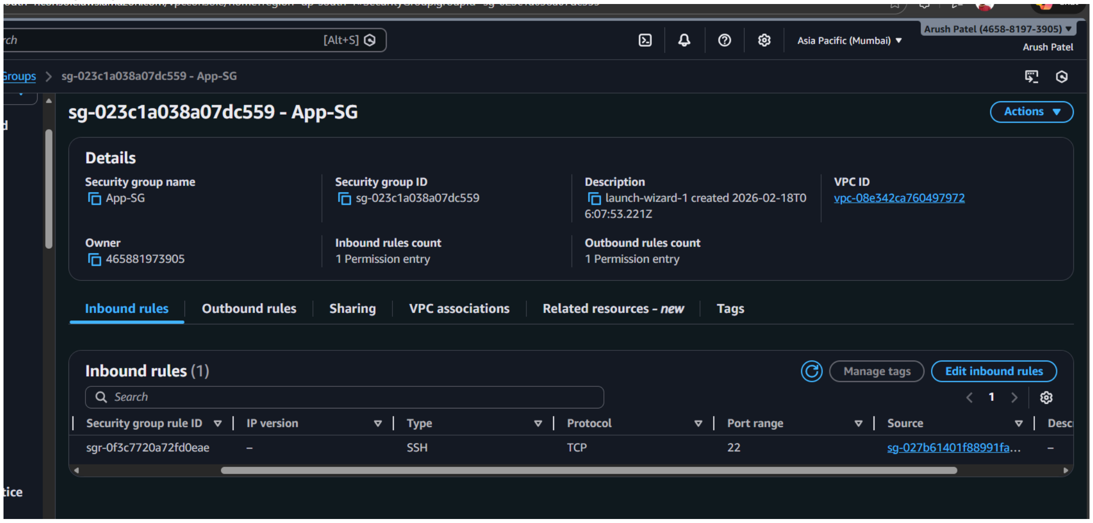
Inbound rules (1)

Security group rule ID	IP version	Type	Protocol	Port range	Source	Descri
sgr-07d4fa8cf4d44eb95	IPv4	SSH	TCP	22	49.36.176.203/32	–

Step 8: Deploy Private App Server

- **Action:** Launched `App-Server` in the Private Subnet with **no public IP**.
- **Why:** This prevents 100% of direct internet-based attacks (like brute-force SSH) on the application server.

👉 EC2 → Launch instance
Name: App-Server
AMI: Amazon Linux 2
Instance type: t2.micro
Key pair: Same as bastion
VPC: SecureCart-VPC
Subnet: Private-Subnet
Auto-assign public IP: Disable
Security Group (App-SG)
HTTP (80) → Source: ALB SG (add later)
SSH (22) → Source: Bastion-SG 👉 Launch



[Alt+S]
Asia Pacific (Mumbai) ▼
Arush Patel (4658-8197-3905) ▼
Arush Patel
Groups > sg-023c1a038a07dc559 - App-SG
Actions ▼

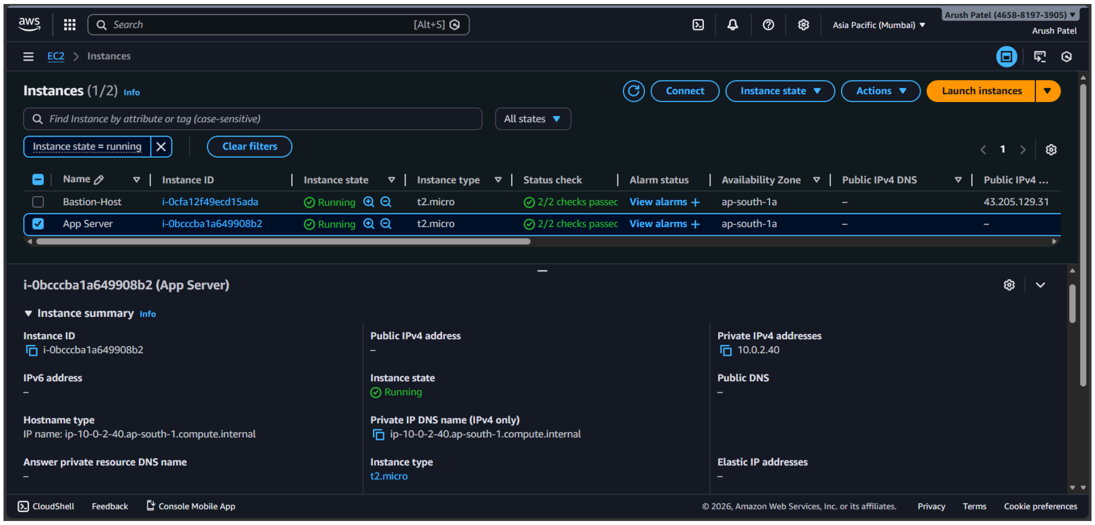
sg-023c1a038a07dc559 - App-SG

Details

Security group name App-SG	Security group ID sg-023c1a038a07dc559	Description launch-wizard-1 created 2026-02-18T06:07:53.221Z	VPC ID vpc-08e342ca760497972
Owner 465881973905	Inbound rules count 1 Permission entry	Outbound rules count 1 Permission entry	

Inbound rules (1)

Security group rule ID	IP version	Type	Protocol	Port range	Source	Desc
sgr-0f3c7720a72fd0eae	–	SSH	TCP	22	sg-027b61401f88991fa...	–



Instances (1/2) Info

Find Instance by attribute or tag (case-sensitive)
All states ▼
Instance state = running ✕
Clear filters
1

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...
□	Bastion-Host	i-0cfa12f49ecd15ada	Running	t2.micro	2/2 checks passec	View alarms +	ap-south-1a	–	43.205.129.31
☑	App Server	i-0bcccba1a649908b2	Running	t2.micro	2/2 checks passec	View alarms +	ap-south-1a	–	–

i-0bcccba1a649908b2 (App Server)

▼ Instance summary Info

Instance ID i-0bcccba1a649908b2	Public IPv4 address –	Private IPv4 addresses 10.0.2.40
IPv6 address –	Instance state Running	Public DNS –
Hostname type IP name: ip-10-0-2-40.ap-south-1.compute.internal	Private IP DNS name (IPv4 only) ip-10-0-2-40.ap-south-1.compute.internal	
Answer private resource DNS name –	Instance type t2.micro	Elastic IP addresses –

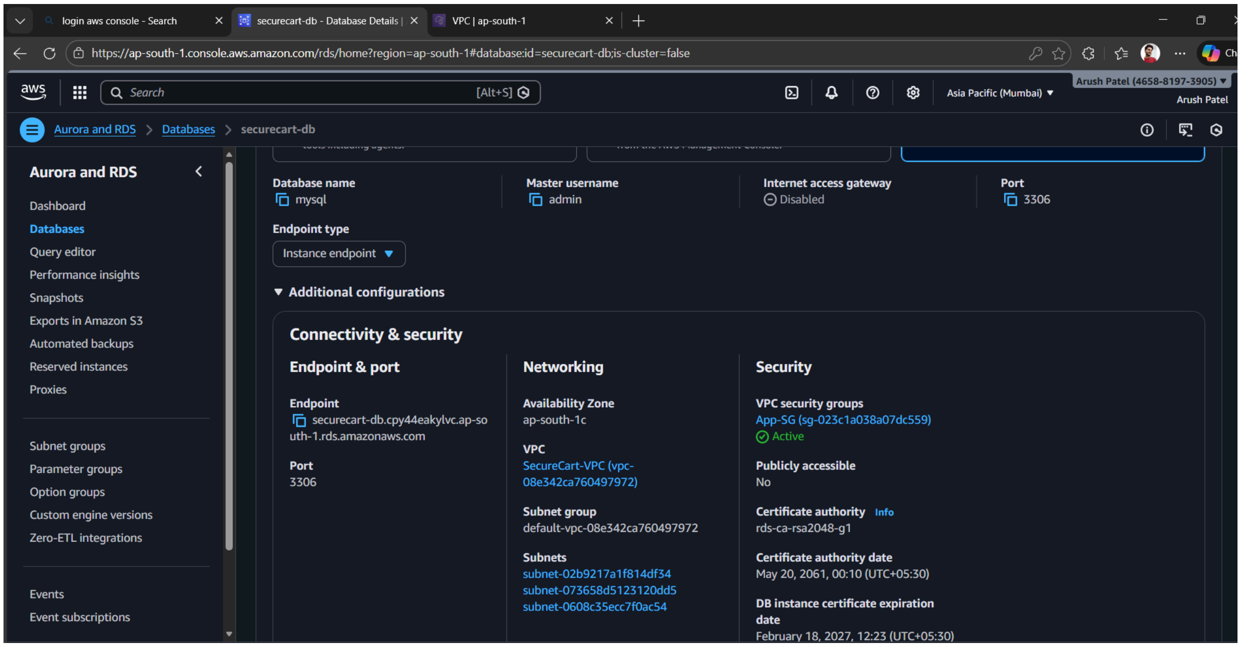
Step 9: Application Load Balancer (ALB)

- **Action:** Created an internet-facing ALB deployed across multiple public subnets in different Availability Zones.
- **Why:** Users should never connect directly to an EC2 instance. The ALB provides a buffer and handles SSL termination.

👉 EC2 → Load Balancers → Create load balancer
Application Load Balancer
Name: SecureCart-ALB
Scheme: Internet-facing
IP type: IPv4
Network Mapping
VPC: SecureCart-VPC
Subnet: Public-Subnet
Security Group
Allow HTTP (80) from 0.0.0.0/0
Target Group
Create new target group
Type: Instances
Protocol: HTTP
Port: 80
Register App-Server👉 Create ALB

Step 10: Private RDS Database

- **Action:** Launched a MySQL RDS instance in the Private Subnet. Set `Publicly Accessible: No`.
- **Why:** Database security is paramount. By making it private, we ensure it only speaks to our `App-Server`.



https://ap-south-1.console.aws.amazon.com/rds/home?region=ap-south-1#database:id=securecart-db;is-cluster=false

Database name: mysql
Master username: admin
Internet access gateway: Disabled
Port: 3306

Endpoint type: Instance endpoint

Connectivity & security

Endpoint & port
Endpoint: securecart-db.cpy44eakylvc.ap-south-1.rds.amazonaws.com
Port: 3306

Networking
Availability Zone: ap-south-1c
VPC: SecureCart-VPC (vpc-08e342ca760497972)
Subnet group: default-vpc-08e342ca760497972
Subnets: subnet-02b9217a1f814df34, subnet-073658d5123120dd5, subnet-0608c35ecc7f0ac54

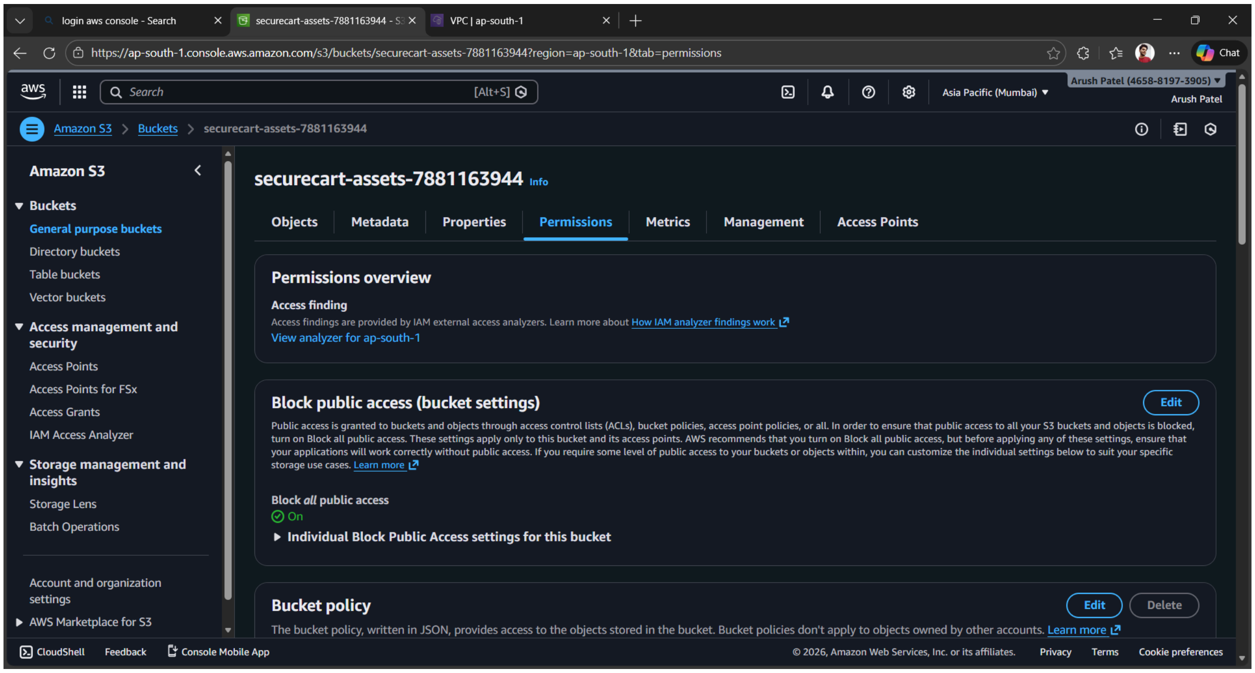
Security
VPC security groups: App-SG (sg-023c1a038a07dc559) Active
Publicly accessible: No
Certificate authority Info: rds-ca-rsa2048-g1
Certificate authority date: May 20, 2061, 00:10 (UTC+05:30)
DB instance certificate expiration date: February 18, 2027, 12:23 (UTC+05:30)

👉 Services → RDS → Create database

Engine: MySQL
Template: Free Tier
Settings
DB identifier: securecart-db
Username/password
Connectivity
VPC: SecureCart-VPC
Public access: ❌ No
Subnet group: Custom DB subnet group containing multiple private subnets.
Security Group:
Allow DB port ONLY from App-SG
Additional
Enable encryption
Enable backups
👉 Create database

Step 11: S3 Bucket Security

- **Action:** Enabled **Block all public access** and Server-Side Encryption (SSE-S3).
- **Why:** To prevent accidental data leaks (like the common "open S3 bucket" vulnerability).



https://ap-south-1.console.aws.amazon.com/s3/buckets/securecart-assets-7881163944?region=ap-south-1&tab=permissions

securecart-assets-7881163944 Info

Permissions overview

Access finding

Access findings are provided by IAM external access analyzers. Learn more about How IAM analyzer findings work

View analyzer for ap-south-1

Block public access (bucket settings)

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more

Block *all* public access
✓ On

▸ Individual Block Public Access settings for this bucket

Bucket policy

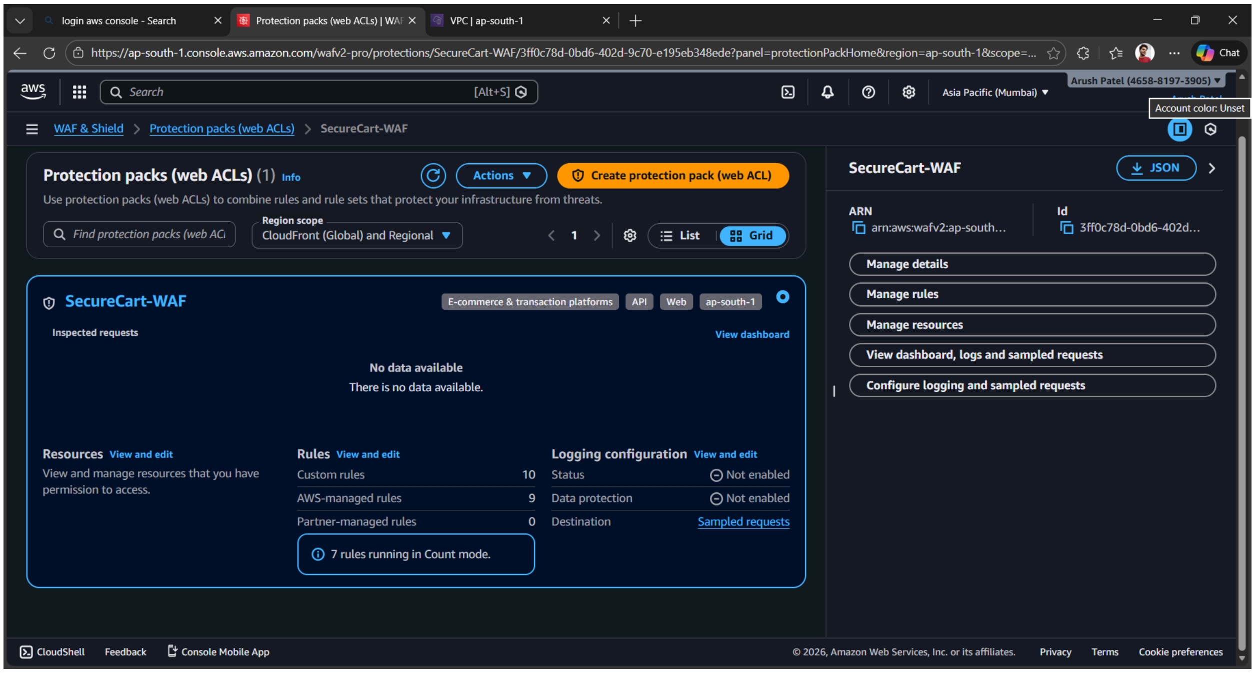
The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more

👉 Services → S3 → Create bucket
Name: securecart-assets-<unique>
Region:
Permissions
Block ALL public access ✅

Encryption
Enable Server-side encryption (SSE-S3)
👉 Create bucket

Step 12: AWS WAF Integration

- **Action:** Created a Web ACL (`SecureCart-WAF`) and associated it with the ALB. Attached managed rule sets to the ALB.
- **Why:** To protect the app from Layer 7 attacks like **SQL Injection** and **Cross-Site Scripting (XSS)**.



login aws console - Search
Protection packs (web ACLs) | WAF
VPC | ap-south-1
Chat
Search [Alt+S]
Asia Pacific (Mumbai)
Arush Patel (4658-8197-3905)
Account color: Unset
WAF & Shield > Protection packs (web ACLs) > SecureCart-WAF
Actions
Create protection pack (web ACL)
List
Grid
JSON
Manage details
Manage rules
Manage resources
View dashboard, logs and sampled requests
Configure logging and sampled requests
CloudShell
Feedback
Console Mobile App
Privacy
Terms
Cookie preferences

https://ap-south-1.console.aws.amazon.com/wafv2-pro/protections/SecureCart-WAF/3ff0c78d-0bd6-402d-9c70-e195eb348ede?panel=protectionPackHome&region=ap-south-1&scope=...

Protection packs (web ACLs) (1) Info

Use protection packs (web ACLs) to combine rules and rule sets that protect your infrastructure from threats.

Find protection packs (web ACL

Region scope: CloudFront (Global) and Regional

SecureCart-WAF — E-commerce & transaction platforms | API | Web | ap-south-1

Inspected requests — View dashboard

No data available
There is no data available.

Resources View and edit
View and manage resources that you have permission to access.

Rules View and edit	
Custom rules	10
AWS-managed rules	9
Partner-managed rules	0

ⓘ 7 rules running in Count mode.

Logging configuration View and edit	
Status	⊖ Not enabled
Data protection	⊖ Not enabled
Destination	Sampled requests

SecureCart-WAF

ARN: arn:aws:wafv2:ap-south...
Id: 3ff0c78d-0bd6-402d...

© 2026, Amazon Web Services, Inc. or its affiliates.

👉 Services → WAF & Shield → Create Web ACL
Name: SecureCart-WAF
Resource type: Regional
Region:
Associate with: SecureCart-ALB
Rules
Add managed rule groups:
AWS Core Rule Set
SQL Injection Rule Set
👉 Create Web ACL

✅ FINAL VERIFICATION CHECKLIST — HOW TO CONFIRM EACH POINT

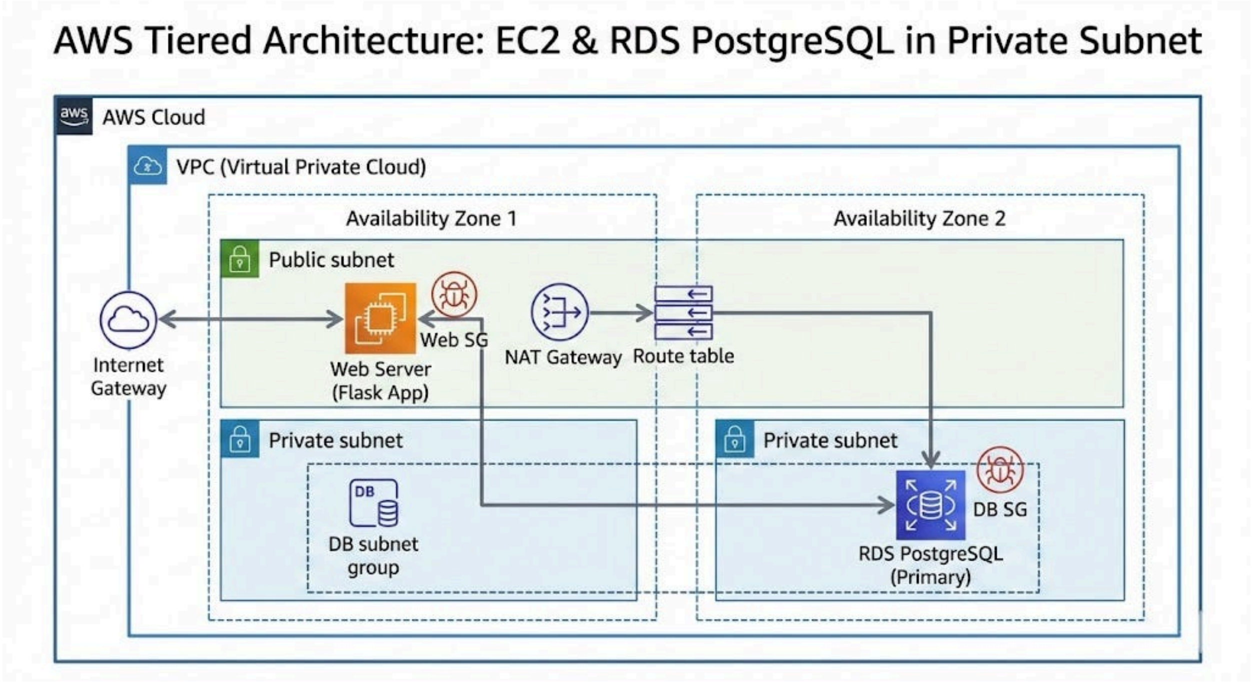
✔ EC2 instances are deployed in private subnets with no public IPs, preventing direct internet access.
✔The RDS database is isolated in private subnets with public access disabled.
✔All S3 buckets enforce block public access and encryption at rest.
✔Traffic is routed securely through AWS WAF and ALB to private EC2 instances, which access RDS over private networking.

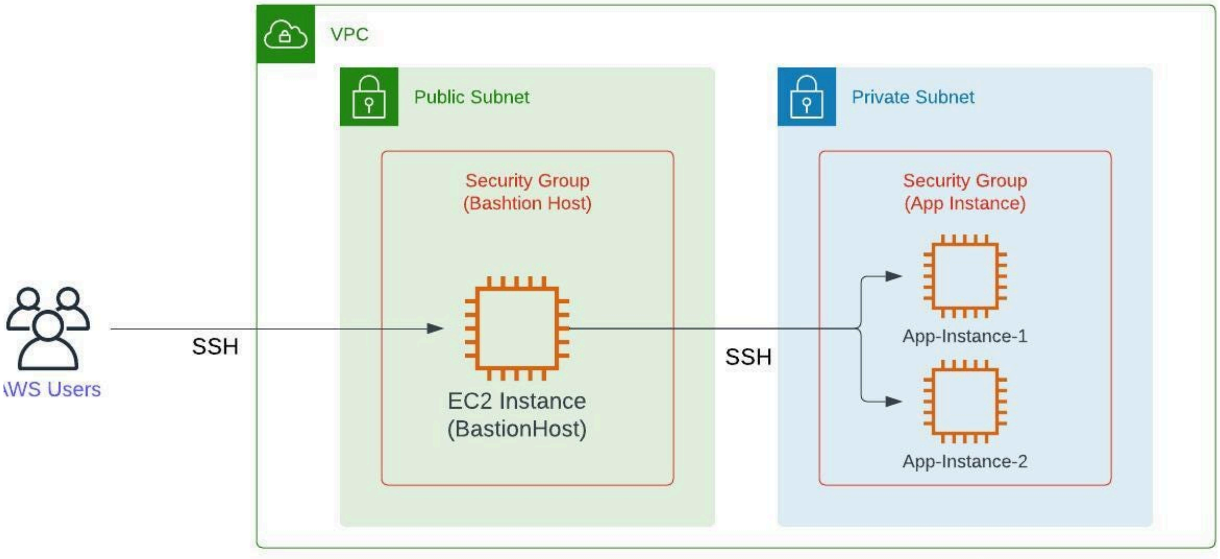
✔Administrative access is restricted using a bastion host, enforcing controlled SSH access.
✔Outbound internet access from private subnets is controlled using a NAT Gateway.

Conclusion:

I designed and implemented a secure, real-world AWS cloud architecture by using a custom VPC with public and private subnets, a bastion host for controlled access, private EC2 and RDS resources, AWS WAF for application protection, NAT Gateway for safe outbound access, and a private S3 bucket for storage. Through this project, I learned how to build security-first cloud systems, control traffic flow, apply least-privilege access, and follow AWS best practices, while also documenting the complete architecture with diagrams and screenshots for future reference and knowledge sharing.

For Reference:




VPC
Public Subnet
Private Subnet
Security Group (Bashtion Host)
Security Group (App Instance)
SSH
EC2 Instance (BastionHost)
SSH
App-Instance-1
App-Instance-2
WS Users