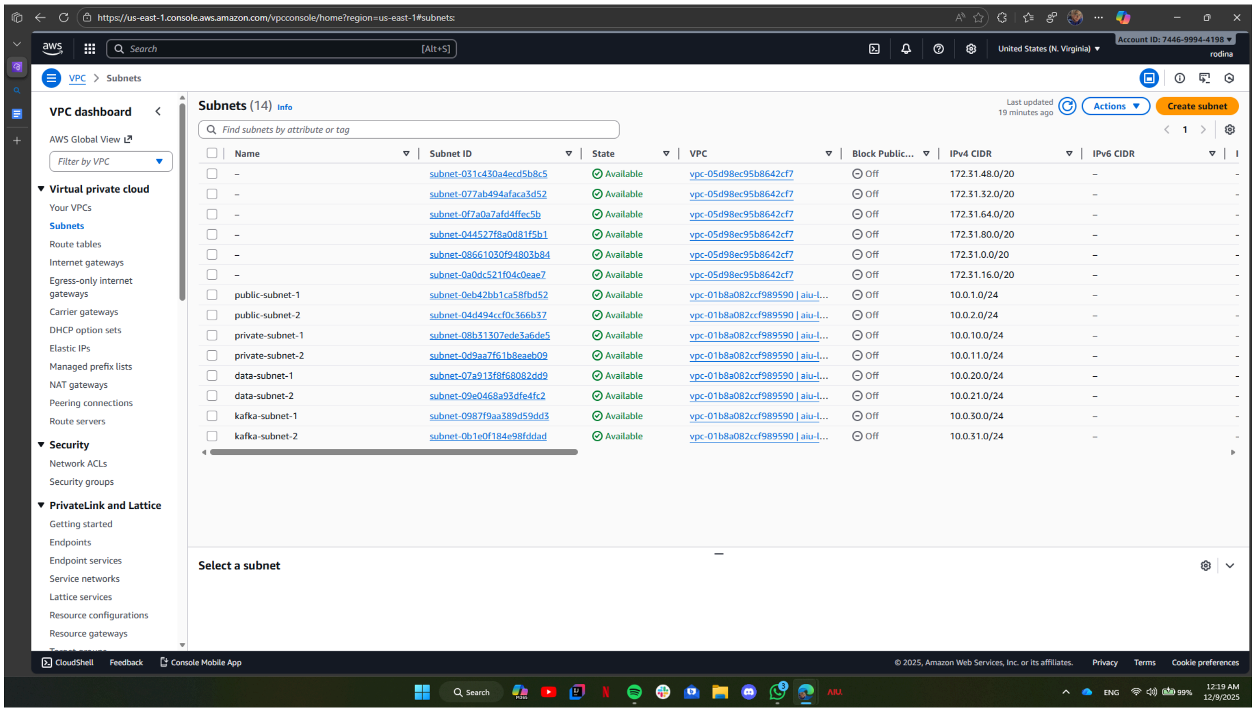
Aya Mamdouh Ahmed Elhabashy
22101087

VPC Creation:
Name: aiu-learning-platform-vpc

IPv4 CIDR: `10.0.0.0/16`

Tier	Subnets (2 AZs)	Purpose
Public	10.0.1.0/24, 10.0.2.0/24	Internet-facing ALB, NAT
Private	10.0.10.0/24,10.0.11.0/24	Container hosts
Data	10.0.20.0/24,10.0.21.0/24	RDS
Kafka	10.0.30.0/24,10.0.31.0/24	Kafka brokers & Zookeeper

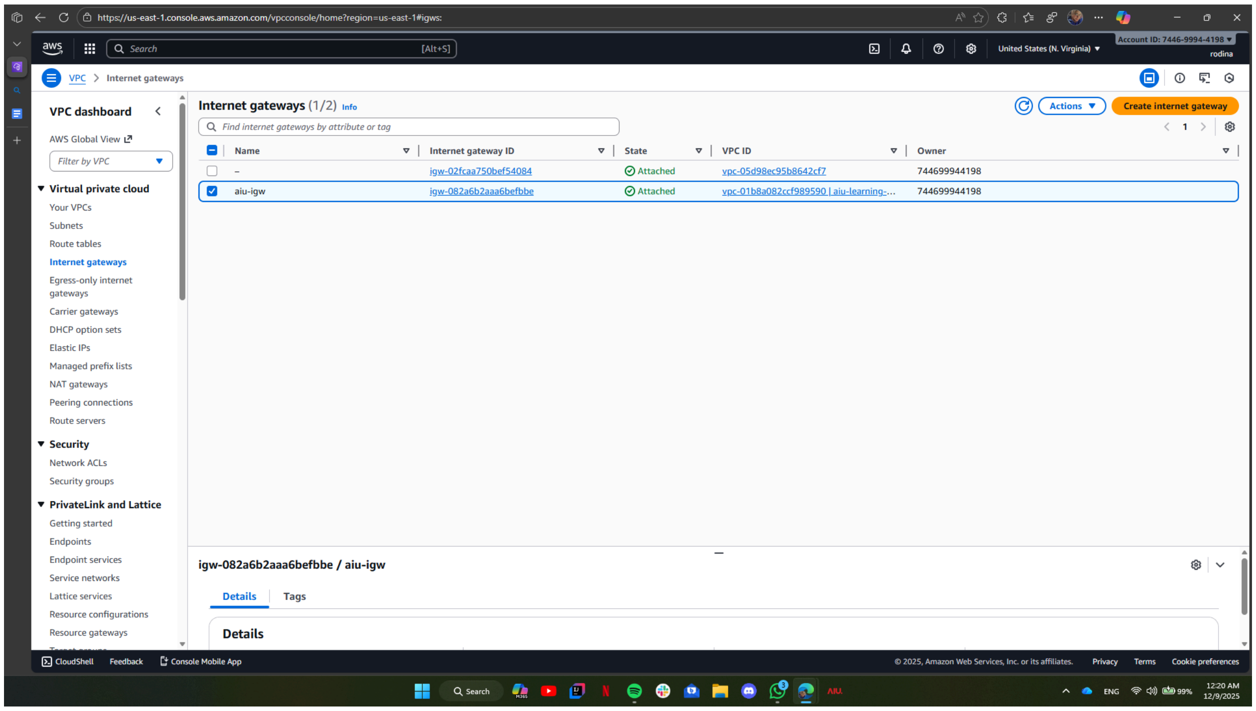
Subnets:



Subnets (14)

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR	IPv6 CIDR
–	subnet-031c430a4ecd5b8c5	Available	vpc-05d98ec95b8642cf7	Off	172.31.48.0/20	–
–	subnet-077ab494afaca3d52	Available	vpc-05d98ec95b8642cf7	Off	172.31.32.0/20	–
–	subnet-0f7a0a7afd4ffec5b	Available	vpc-05d98ec95b8642cf7	Off	172.31.64.0/20	–
–	subnet-044527f8a0d81f5b1	Available	vpc-05d98ec95b8642cf7	Off	172.31.80.0/20	–
–	subnet-08661030f94803b84	Available	vpc-05d98ec95b8642cf7	Off	172.31.0.0/20	–
–	subnet-0a0dc521f04c0eae7	Available	vpc-05d98ec95b8642cf7	Off	172.31.16.0/20	–
public-subnet-1	subnet-0eb42bb1ca58fbd52	Available	vpc-01b8a082ccf989590 | aiu-l...	Off	10.0.1.0/24	–
public-subnet-2	subnet-04d494ccf0c366b37	Available	vpc-01b8a082ccf989590 | aiu-l...	Off	10.0.2.0/24	–
private-subnet-1	subnet-08b31307ede3a6de5	Available	vpc-01b8a082ccf989590 | aiu-l...	Off	10.0.10.0/24	–
private-subnet-2	subnet-0d9aa7f61b8eaeb09	Available	vpc-01b8a082ccf989590 | aiu-l...	Off	10.0.11.0/24	–
data-subnet-1	subnet-07a913f8f68082dd9	Available	vpc-01b8a082ccf989590 | aiu-l...	Off	10.0.20.0/24	–
data-subnet-2	subnet-09e0468a93dfe4fc2	Available	vpc-01b8a082ccf989590 | aiu-l...	Off	10.0.21.0/24	–
kafka-subnet-1	subnet-0987f9aa389d59dd3	Available	vpc-01b8a082ccf989590 | aiu-l...	Off	10.0.30.0/24	–
kafka-subnet-2	subnet-0b1e0f184e98fddad	Available	vpc-01b8a082ccf989590 | aiu-l...	Off	10.0.31.0/24	–

Internet Gateway:



https://us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#igws:

VPC > Internet gateways

Internet gateways (1/2) Info

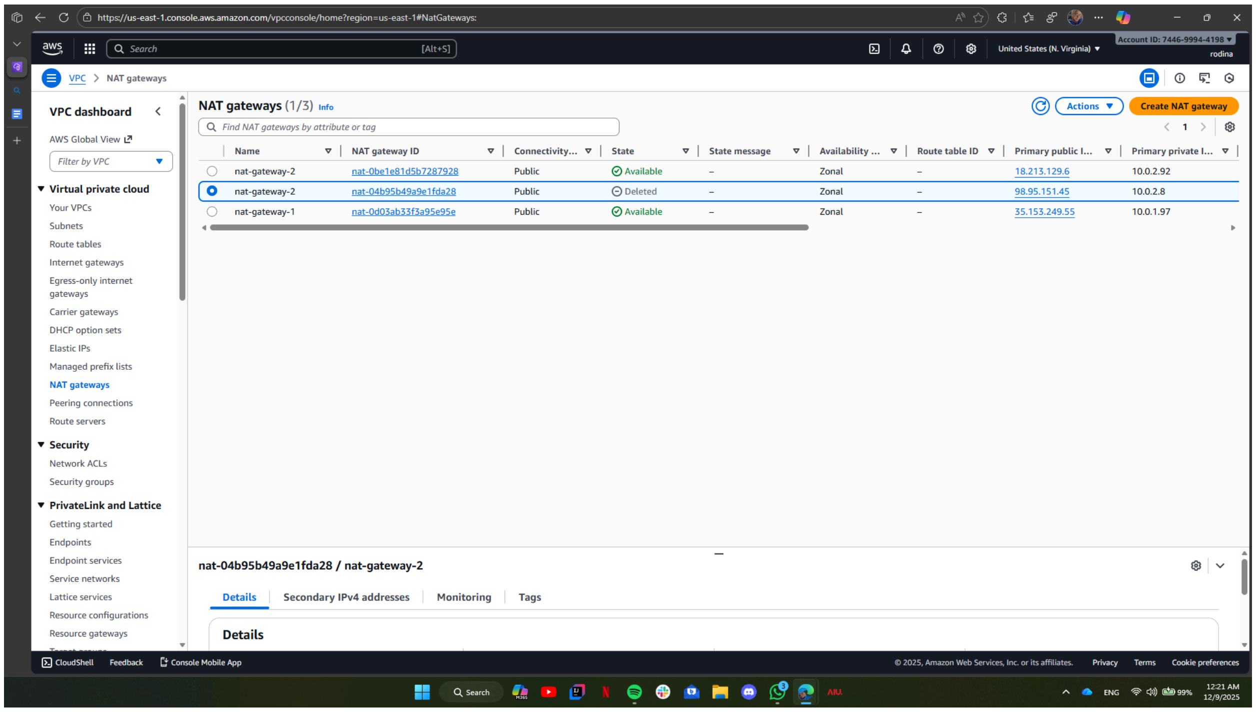
	Name	Internet gateway ID	State	VPC ID	Owner
☐	–	igw-02fcaa750bef54084	Attached	vpc-05d98ec95b8642cf7	744699944198
☑	aiu-igw	igw-082a6b2aaa6befbbe	Attached	vpc-01b8a082ccf989590 | aiu-learning-...	744699944198

igw-082a6b2aaa6befbbe / aiu-igw

Details | Tags

Details

NAT gateways:



https://us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#NatGateways:

VPC > NAT gateways

NAT gateways (1/3) Info

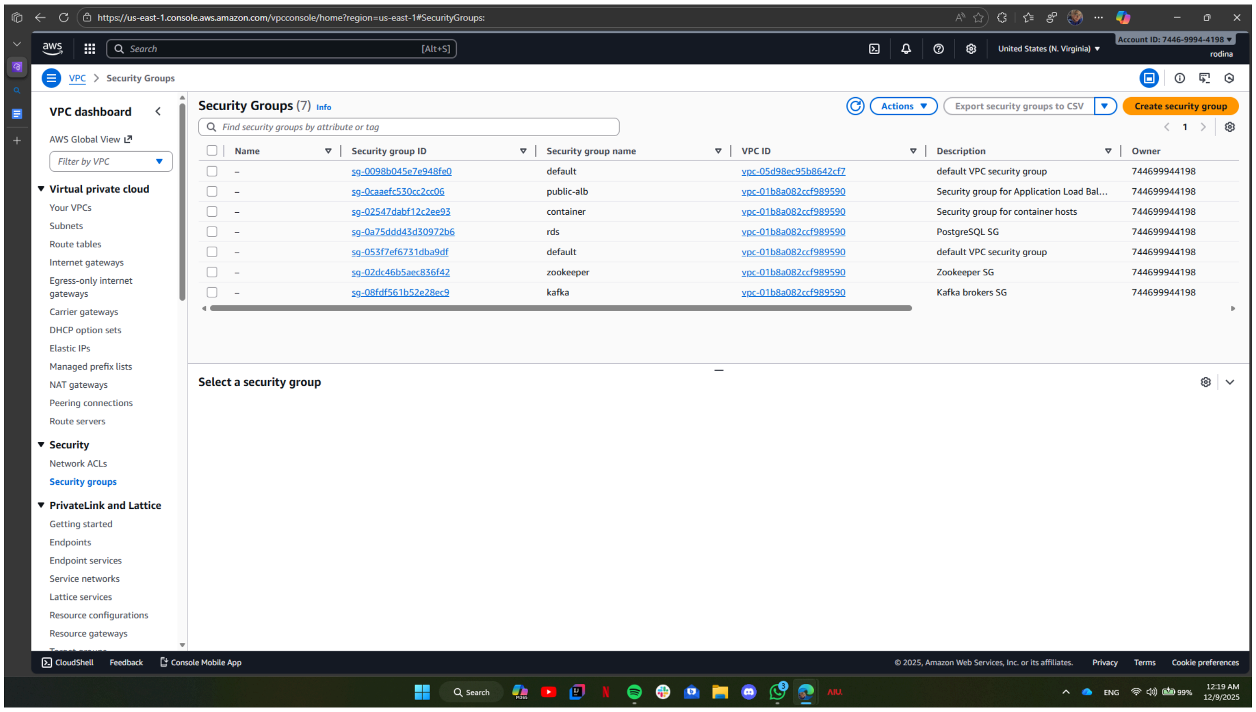
	Name	NAT gateway ID	Connectivity...	State	State message	Availability ...	Route table ID	Primary public I...	Primary private I...
○	nat-gateway-2	nat-0be1e81d5b7287928	Public	Available	–	Zonal	–	18.213.129.6	10.0.2.92
●	nat-gateway-2	nat-04b95b49a9e1fda28	Public	Deleted	–	Zonal	–	98.95.151.45	10.0.2.8
○	nat-gateway-1	nat-0d03ab33f3a95e95e	Public	Available	–	Zonal	–	35.153.249.55	10.0.1.97

nat-04b95b49a9e1fda28 / nat-gateway-2

Details | Secondary IPv4 addresses | Monitoring | Tags

Details

Security Groups:


https://us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#SecurityGroups:

Account ID: 7446-9994-4198 · United States (N. Virginia) · rodina

VPC > Security Groups

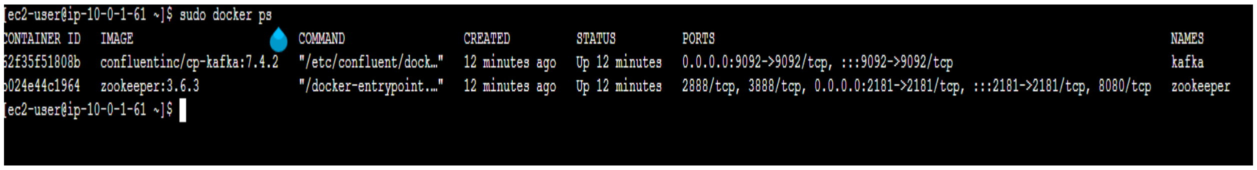
Security Groups (7) Info

Name	Security group ID	Security group name	VPC ID	Description	Owner
–	sg-0098b045e7e948fe0	default	vpc-05d98ec95b8642cf7	default VPC security group	744699944198
–	sg-0caaefc530cc2cc06	public-alb	vpc-01b8a082ccf989590	Security group for Application Load Bal...	744699944198
–	sg-02547dabf12c2ee93	container	vpc-01b8a082ccf989590	Security group for container hosts	744699944198
–	sg-0a75ddd43d30972b6	rds	vpc-01b8a082ccf989590	PostgreSQL SG	744699944198
–	sg-053f7ef6731dba9df	default	vpc-01b8a082ccf989590	default VPC security group	744699944198
–	sg-02dc46b5aec836f42	zookeeper	vpc-01b8a082ccf989590	Zookeeper SG	744699944198
–	sg-08fdf561b52e28ec9	kafka	vpc-01b8a082ccf989590	Kafka brokers SG	744699944198

Select a security group

For Phase2 i worked on **AWS Apache Kafka**

I created the Kafka & Zookeeper


```
[ec2-user@ip-10-0-1-61 ~]$ sudo docker ps
CONTAINER ID   IMAGE                        COMMAND                  CREATED          STATUS          PORTS                                                                      NAMES
52f35f51808b   confluentinc/cp-kafka:7.4.2  "/etc/confluent/dock…"   12 minutes ago   Up 12 minutes   0.0.0.0:9092->9092/tcp, :::9092->9092/tcp                                  kafka
b024e44c1964   zookeeper:3.6.3              "/docker-entrypoint.…"   12 minutes ago   Up 12 minutes   2888/tcp, 3888/tcp, 0.0.0.0:2181->2181/tcp, :::2181->2181/tcp, 8080/tcp    zookeeper
[ec2-user@ip-10-0-1-61 ~]$
```

Using this:

For kafka :

```
sudo docker run -d --name kafka \
  --network kafka-net \
  -e KAFKA_BROKER_ID=1 \
  -e KAFKA_ZOOKEEPER_CONNECT=zookeeper:2181 \
  -e KAFKA_LISTENERS=PLAINTEXT://0.0.0.0:9092 \
  -e KAFKA_ADVERTISED_LISTENERS=PLAINTEXT://10.0.1.61:9092 \
  -e KAFKA_OFFSETS_TOPIC_REPLICATION_FACTOR=1 \
  -p 9092:9092 \
  confluentinc/cp-kafka:7.4.2
```

For zookeeper:

```
sudo docker run -d --name zookeeper \
  --network kafka-net \
  -p 2181:2181 \
  -v /home/ec2-user/zookeeper-data:/data \
  -e ZOO_MY_ID=1 \
  -e ZOO_PORT=2181 \
  -e ZOO_TICK_TIME=2000 \
  -e ZOO_INIT_LIMIT=5 \
  -e ZOO_SYNC_LIMIT=2 \
  zookeeper:3.6.3
```

And for creating the topics:

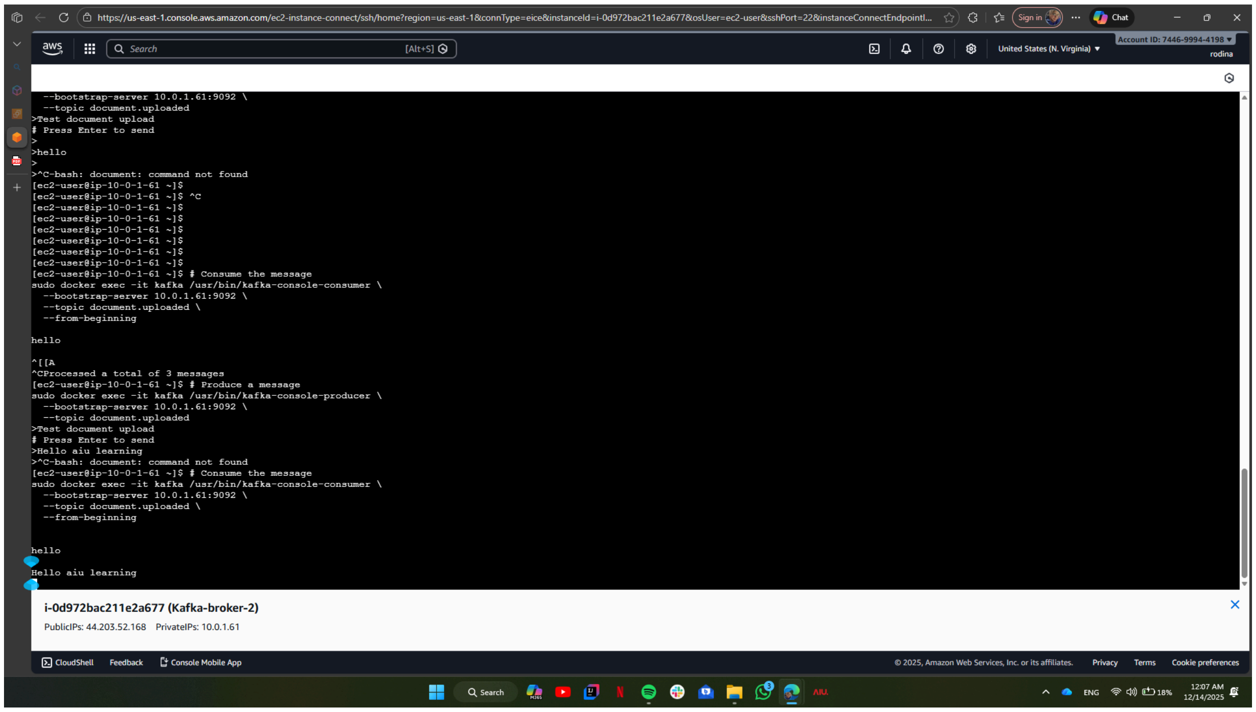
```
TOPICS=(
document.uploaded
document.processed
notes.generated
quiz.requested
quiz.generated
audio.transcription.requested
audio.transcription.completed
audio.generation.requested
audio.generation.completed
chat.message
)

for topic in "${TOPICS[@]}"; do
  sudo docker exec -it kafka /usr/bin/kafka-topics \
    --create \
    --bootstrap-server 10.0.1.61:9092 \
    --replication-factor 1 \
    --partitions 3 \
    --topic "$topic"
done
```

I did it on kafka broker1 but suddenly kafka & zookepeer stopped working so i used kafka broker 2 and started it all again

I created the producer & consumer using this:

And finally the producer sends a message to the consumer

```
# Produce a message
sudo docker exec -it kafka /usr/bin/kafka-console-producer \
  --bootstrap-server 10.0.1.61:9092 \
  --topic document.uploaded
>Test document upload
# Press Enter to send
# Consume the message
sudo docker exec -it kafka /usr/bin/kafka-console-consumer \
  --bootstrap-server 10.0.1.61:9092 \
  --topic document.uploaded \
  --from-beginning
```



```
  --bootstrap-server 10.0.1.61:9092 \
  --topic document.uploaded
>Test document upload
# Press Enter to send
>
>hello
>
>^C-bash: document: command not found
[ec2-user@ip-10-0-1-61 ~]$
[ec2-user@ip-10-0-1-61 ~]$ ^C
[ec2-user@ip-10-0-1-61 ~]$
[ec2-user@ip-10-0-1-61 ~]$
[ec2-user@ip-10-0-1-61 ~]$
[ec2-user@ip-10-0-1-61 ~]$
[ec2-user@ip-10-0-1-61 ~]$
[ec2-user@ip-10-0-1-61 ~]$
[ec2-user@ip-10-0-1-61 ~]$ # Consume the message
sudo docker exec -it kafka /usr/bin/kafka-console-consumer \
  --bootstrap-server 10.0.1.61:9092 \
  --topic document.uploaded \
  --from-beginning

hello

^[[A
^CProcessed a total of 3 messages
[ec2-user@ip-10-0-1-61 ~]$ # Produce a message
sudo docker exec -it kafka /usr/bin/kafka-console-producer \
  --bootstrap-server 10.0.1.61:9092 \
  --topic document.uploaded
>Test document upload
# Press Enter to send
>Hello aiu learning
>^C-bash: document: command not found
[ec2-user@ip-10-0-1-61 ~]$ # Consume the message
sudo docker exec -it kafka /usr/bin/kafka-console-consumer \
  --bootstrap-server 10.0.1.61:9092 \
  --topic document.uploaded \
  --from-beginning


hello

Hello aiu learning
```

i-0d972bac211e2a677 (Kafka-broker-2)

PublicIPs: 44.203.52.168 PrivateIPs: 10.0.1.61

I created the infrastructure as code "IAC"

WHY?

Infrastructure as Code Implementation

In this phase, AWS CloudFormation was used to implement the infrastructure using Infrastructure as Code (IaC). This approach allows the entire AWS environment to be defined declaratively in a YAML template, ensuring consistency, repeatability, and automation across deployments.

The infrastructure was designed in a modular manner, starting with a Virtual Private Cloud (VPC) that serves as an isolated network boundary for all application resources. Public and private subnets were created across multiple Availability Zones to support high availability and fault tolerance. Public subnets host internet-facing components

such as the Application Load Balancer, while private subnets are reserved for backend services like ECS tasks and databases, improving security.

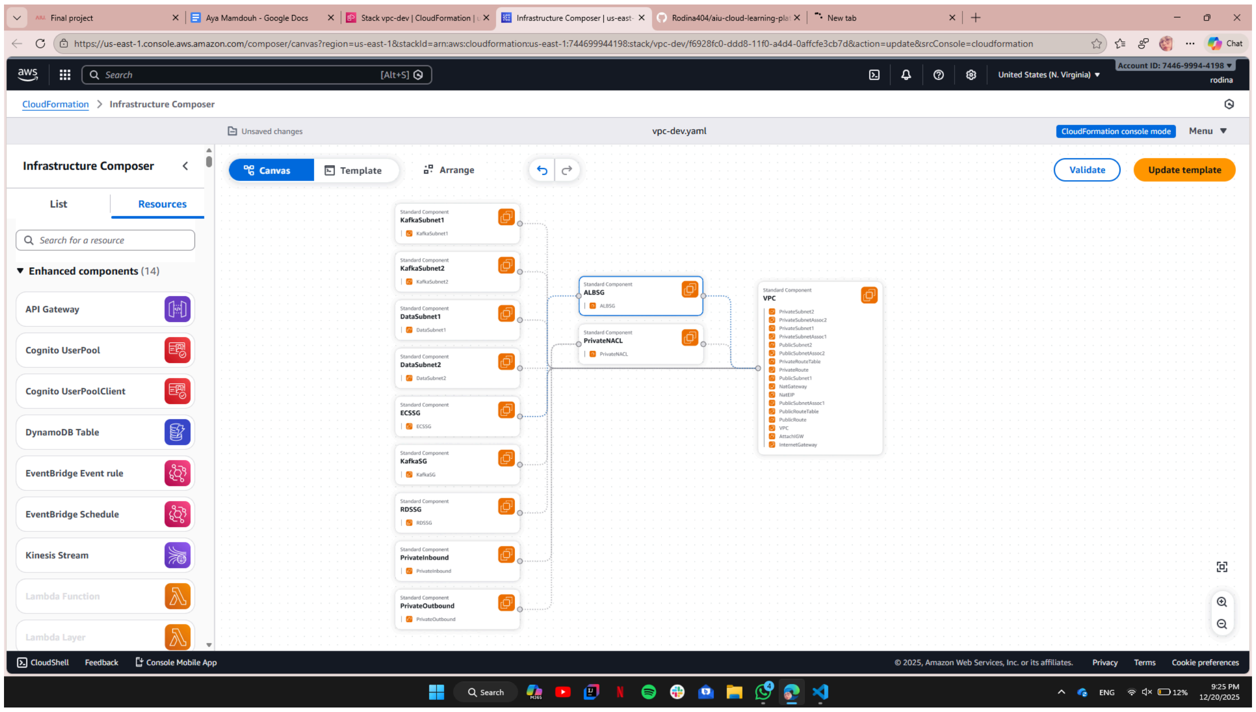
To enable outbound internet access for private resources without exposing them directly, a NAT Gateway was deployed in the public subnet and connected to private route tables. This design follows AWS best practices for secure networking.

Environment separation was achieved using CloudFormation parameters, allowing the same template to be reused for development, staging, and production environments. Resource naming and tagging dynamically adapt based on the selected environment, reducing duplication and simplifying management.

Security was enforced through the use of dedicated security groups. The Application Load Balancer security group allows HTTP and HTTPS traffic from the internet, while the ECS security group only permits traffic originating from the load balancer, implementing the principle of least privilege.

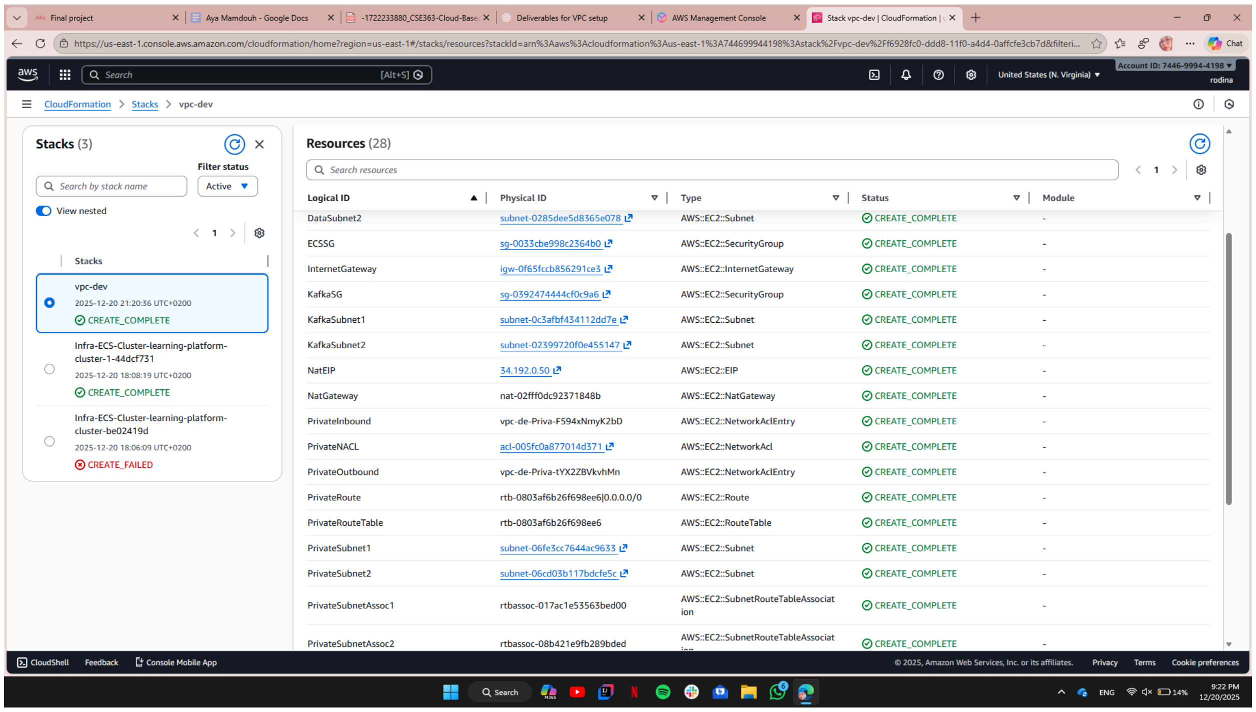
Finally, essential infrastructure identifiers such as VPC ID, subnet IDs, and security group IDs were exposed using CloudFormation outputs. These outputs enable seamless integration with CI/CD pipelines, allowing automated deployments to reference the infrastructure reliably.

That was the canvas:



It said completed Elhamdullah

And that the resources:



CloudFormation > Stacks > vpc-dev

Stacks (3)

Filter status: Active

View nested

Stacks
vpc-dev 2025-12-20 21:20:36 UTC+0200 CREATE_COMPLETE
Infra-ECS-Cluster-learning-platform-cluster-1-44dcf731 2025-12-20 18:08:19 UTC+0200 CREATE_COMPLETE
Infra-ECS-Cluster-learning-platform-cluster-be02419d 2025-12-20 18:06:09 UTC+0200 CREATE_FAILED

Resources (28)

Logical ID	Physical ID	Type	Status	Module
DataSubnet2	subnet-0285dee5d8365e078	AWS::EC2::Subnet	CREATE_COMPLETE	-
ECSSG	sg-0033cbe998c2364b0	AWS::EC2::SecurityGroup	CREATE_COMPLETE	-
InternetGateway	igw-0f65fccb856291ce3	AWS::EC2::InternetGateway	CREATE_COMPLETE	-
KafkaSG	sg-0392474444cf0c9a6	AWS::EC2::SecurityGroup	CREATE_COMPLETE	-
KafkaSubnet1	subnet-0c3afbf434112dd7e	AWS::EC2::Subnet	CREATE_COMPLETE	-
KafkaSubnet2	subnet-02399720f0e455147	AWS::EC2::Subnet	CREATE_COMPLETE	-
NatEIP	34.192.0.50	AWS::EC2::EIP	CREATE_COMPLETE	-
NatGateway	nat-02fff0dc92371848b	AWS::EC2::NatGateway	CREATE_COMPLETE	-
PrivateInbound	vpc-de-Priva-F594xNmyK2bD	AWS::EC2::NetworkAclEntry	CREATE_COMPLETE	-
PrivateNACL	acl-005fc0a877014d371	AWS::EC2::NetworkAcl	CREATE_COMPLETE	-
PrivateOutbound	vpc-de-Priva-tYX2ZBVkvhMn	AWS::EC2::NetworkAclEntry	CREATE_COMPLETE	-
PrivateRoute	rtb-0803af6b26f698ee6|0.0.0.0/0	AWS::EC2::Route	CREATE_COMPLETE	-
PrivateRouteTable	rtb-0803af6b26f698ee6	AWS::EC2::RouteTable	CREATE_COMPLETE	-
PrivateSubnet1	subnet-06fe3cc7644ac9633	AWS::EC2::Subnet	CREATE_COMPLETE	-
PrivateSubnet2	subnet-06cd03b117bdcfe5c	AWS::EC2::Subnet	CREATE_COMPLETE	-
PrivateSubnetAssoc1	rtbassoc-017ac1e53563bed00	AWS::EC2::SubnetRouteTableAssociation	CREATE_COMPLETE	-
PrivateSubnetAssoc2	rtbassoc-08b421e9fb289bded	AWS::EC2::SubnetRouteTableAssociat	CREATE_COMPLETE	-