OKELLO ELIJAH
https://www.linkedin.com/in/okello-elijah-5bb745201

Project: Deploying a Web Application on AWS Fargate

Overview

I worked on a cloud deployment project to host a simple web application called **"Movie Night"** a platform that allows members of a local cinema to vote on which movie should premiere each month.

The project demonstrates the use of several **AWS services** to build a secure, scalable, and containerized application deployment.

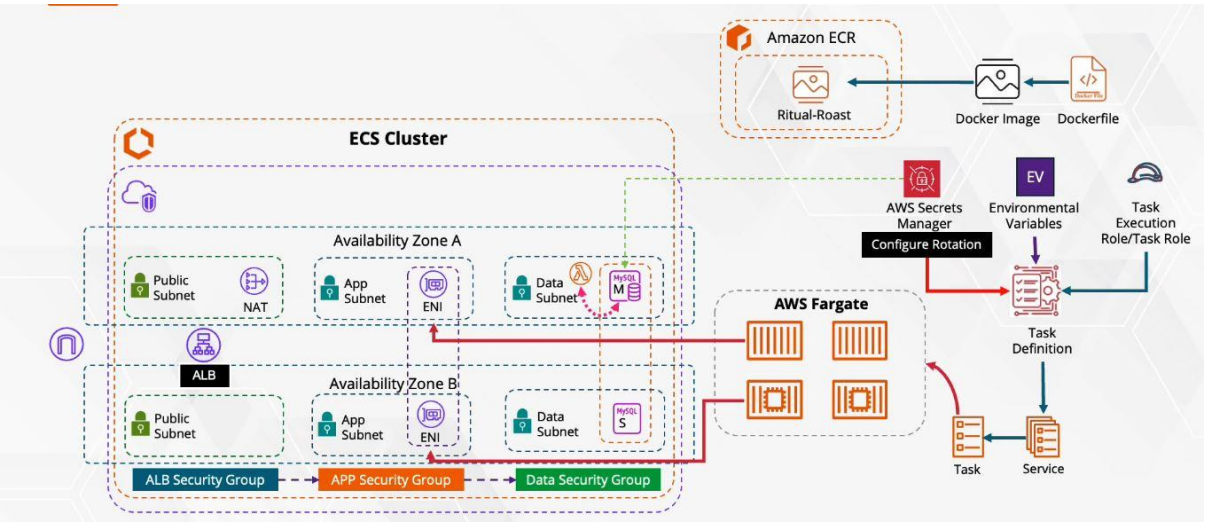
AWS Services Used

- **Amazon VPC** – Custom networking environment with public and private subnets
- **Amazon ECS (Fargate)** – Serverless container orchestration
- **Amazon ECR** – Container image repository
- **Amazon RDS (MySQL)** – Managed relational database service
- **Amazon Secrets Manager** – Secure management of application credentials
- **Amazon EC2** – Used as a Docker build server
- **AWS Lambda** – Automatic credential rotation for Secrets Manager

---

Architecture

A detailed architecture diagram illustrates the complete setup including the VPC, ALB, ECS cluster, and database tier showing secure communication flows between components

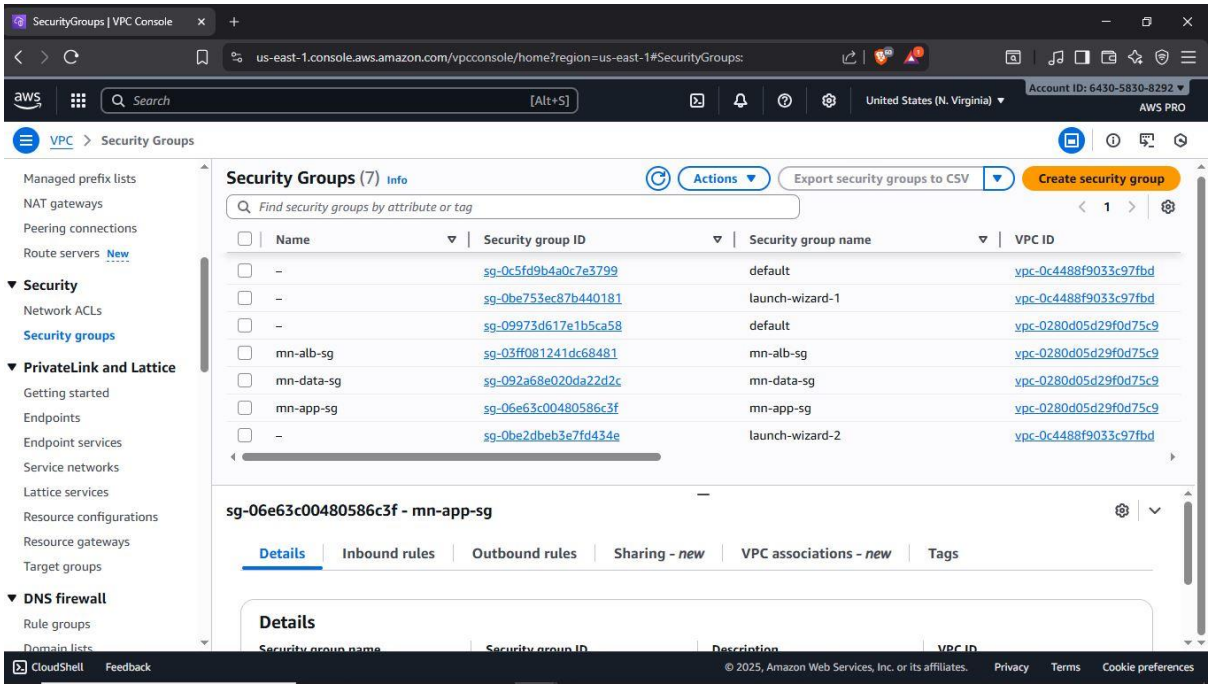


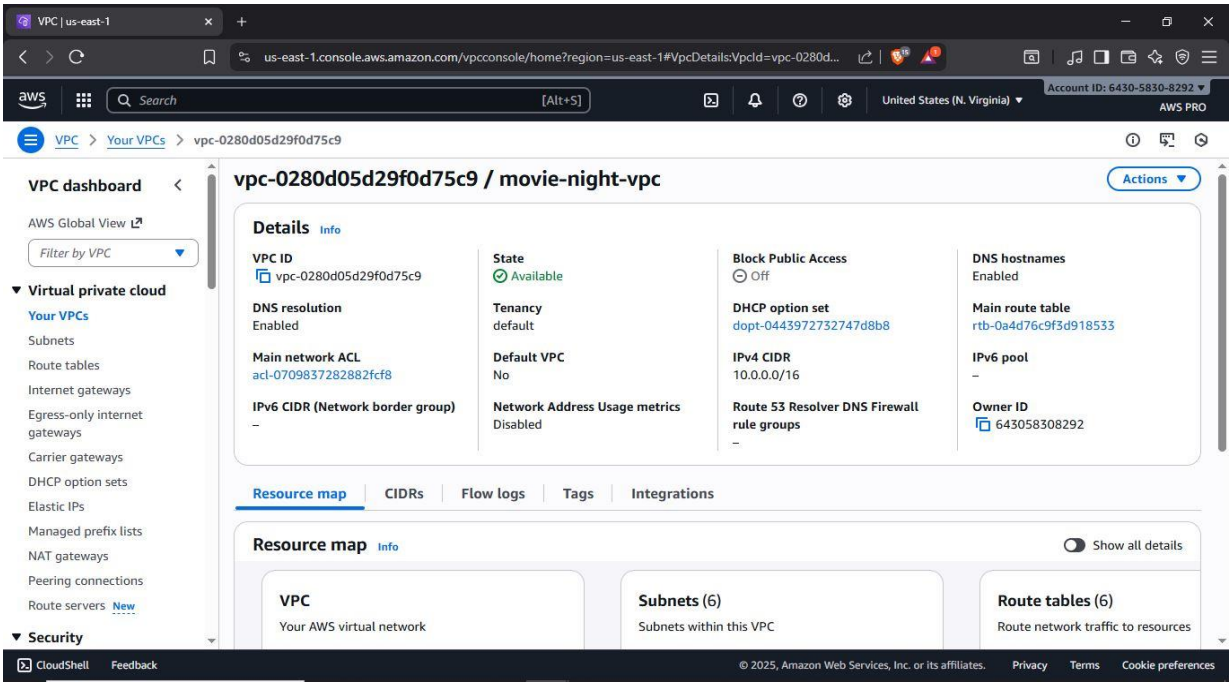
.

Implementation Steps

1. VPC Setup

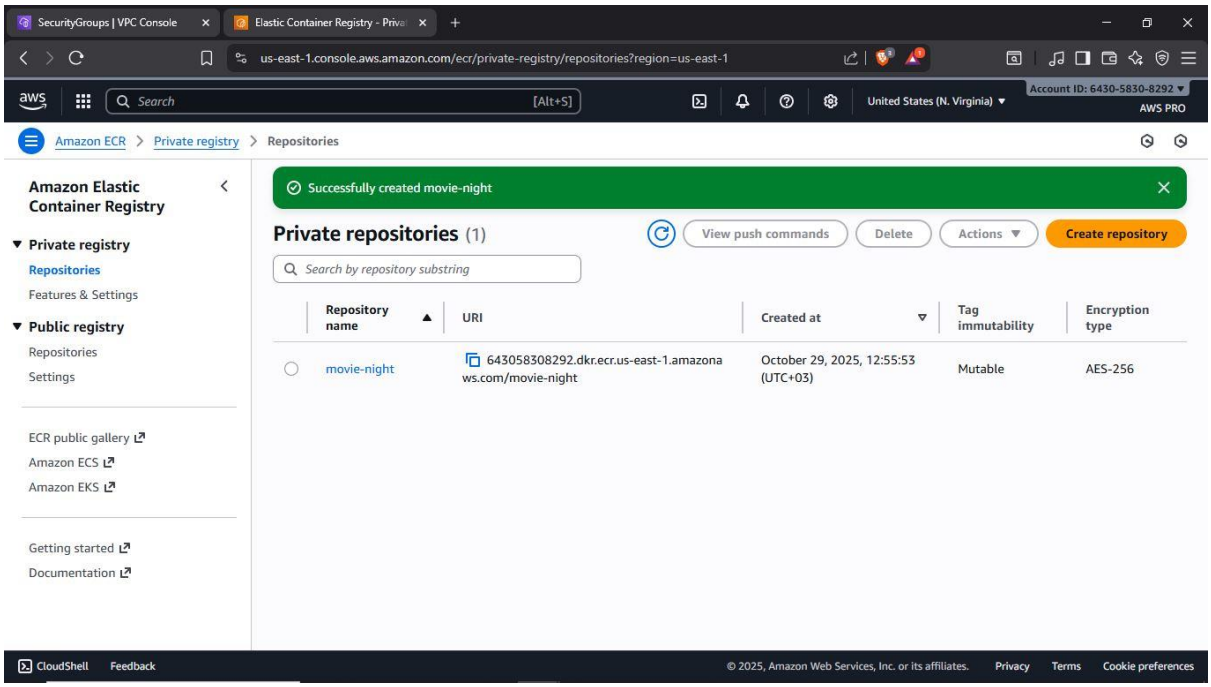
- Created a custom **VPC** with **6 subnets**: 2 public and 4 private.
- Attached an **Internet Gateway**, configured a **NAT Gateway**, and set up appropriate **route tables**.
- Defined **three security groups** to control traffic flow:
  - mn-alb-sg — Allows inbound HTTP traffic from the internet.
  - mn-app-sg — Allows inbound HTTP traffic from the ALB security group.
  - mn-data-sg — Allows inbound MySQL traffic from the app security group.


SecurityGroups | VPC Console
us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#SecurityGroups:
Search
[Alt+S]
United States (N. Virginia)
Account ID: 6430-5830-8292
AWS PRO
VPC > Security Groups
Managed prefix lists
NAT gateways
Peering connections
Route servers New
Security
Network ACLs
Security groups
PrivateLink and Lattice
Getting started
Endpoints
Endpoint services
Service networks
Lattice services
Resource configurations
Resource gateways
Target groups
DNS firewall
Rule groups
Security Groups (7) Info
Actions
Export security groups to CSV
Create security group
Find security groups by attribute or tag
Name
Security group ID
Security group name
VPC ID
– sg-0c5fd9b4a0c7e3799 default vpc-0c4488f9033c97fbd
– sg-0be753ec87b440181 launch-wizard-1 vpc-0c4488f9033c97fbd
– sg-09973d617e1b5ca58 default vpc-0280d05d29f0d75c9
mn-alb-sg sg-03ff081241dc68481 mn-alb-sg vpc-0280d05d29f0d75c9
mn-data-sg sg-092a68e020da22d2c mn-data-sg vpc-0280d05d29f0d75c9
mn-app-sg sg-06e63c00480586c3f mn-app-sg vpc-0280d05d29f0d75c9
– sg-0be2dbeb3e7fd434e launch-wizard-2 vpc-0c4488f9033c97fbd
sg-06e63c00480586c3f - mn-app-sg
Details
Inbound rules
Outbound rules
Sharing - new
VPC associations - new
Tags
Details
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


VPC | us-east-1
us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#VpcDetails:VpcId=vpc-0280d...
Search
[Alt+S]
United States (N. Virginia)
Account ID: 6430-5830-8292
AWS PRO
VPC > Your VPCs > vpc-0280d05d29f0d75c9
VPC dashboard
AWS Global View
Filter by VPC
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways
Peering connections
Route servers New
Security
vpc-0280d05d29f0d75c9 / movie-night-vpc
Actions
Details Info
VPC ID
vpc-0280d05d29f0d75c9
State
Available
Block Public Access
Off
DNS hostnames
Enabled
DNS resolution
Enabled
Tenancy
default
DHCP option set
dopt-0443972732747d8b8
Main route table
rtb-0a4d76c9f3d918533
Main network ACL
acl-0709837282882fcf8
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
643058308292
Resource map
CIDRs
Flow logs
Tags
Integrations
Resource map Info
Show all details
VPC
Your AWS virtual network
Subnets (6)
Subnets within this VPC
Route tables (6)
Route network traffic to resources
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

2. ECR Repository

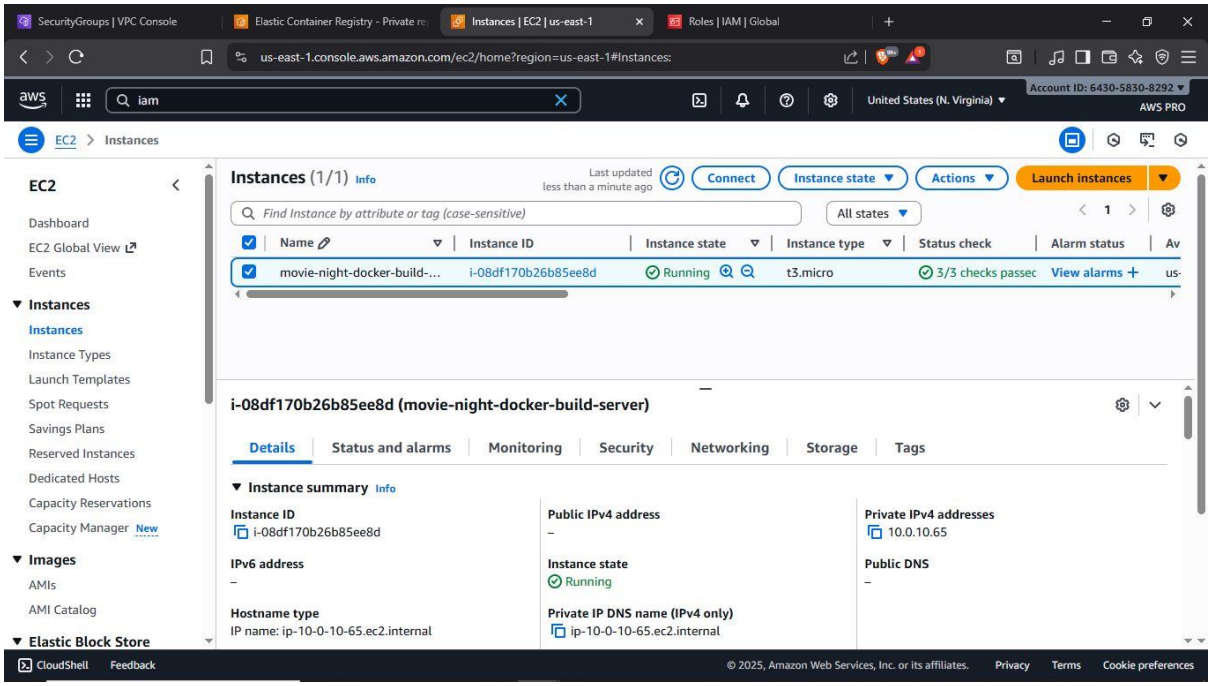
Created a **private repository** in **Amazon ECR** to store and manage container images securely.



---

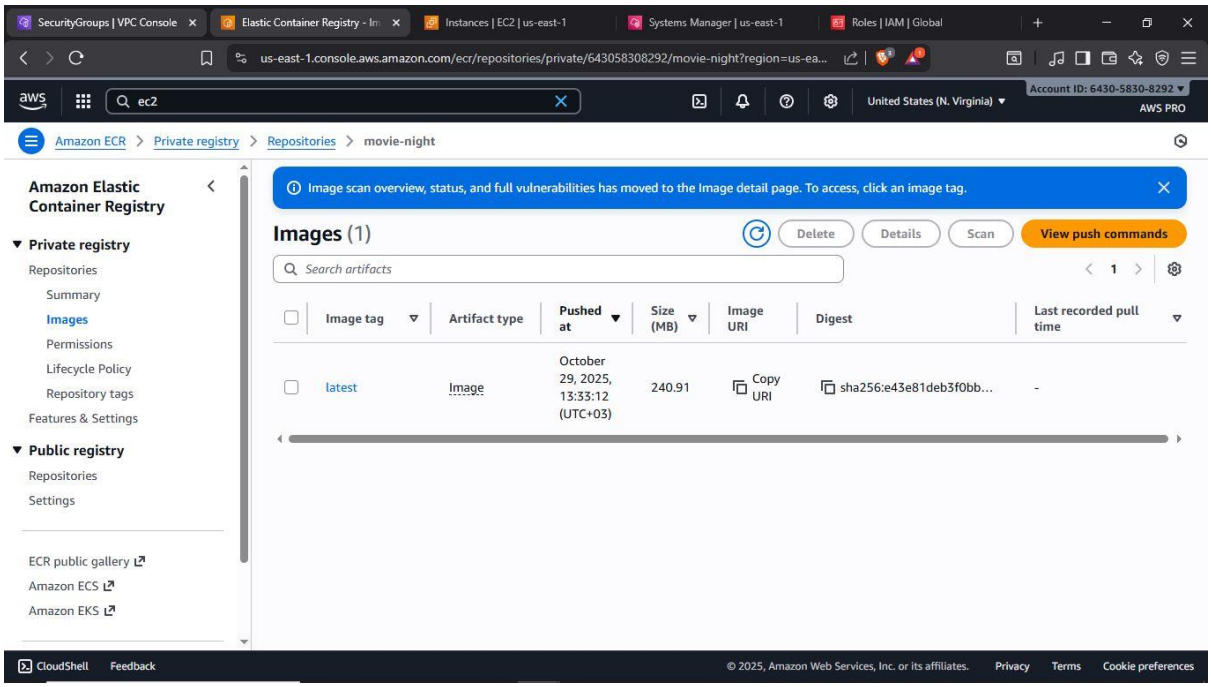
3. Docker Image Build

- Launched an **EC2 instance** as a dedicated Docker build server.
- Assigned an **IAM role** to allow interaction with ECR.
- Installed Docker, built the application image, and verified its functionality locally.



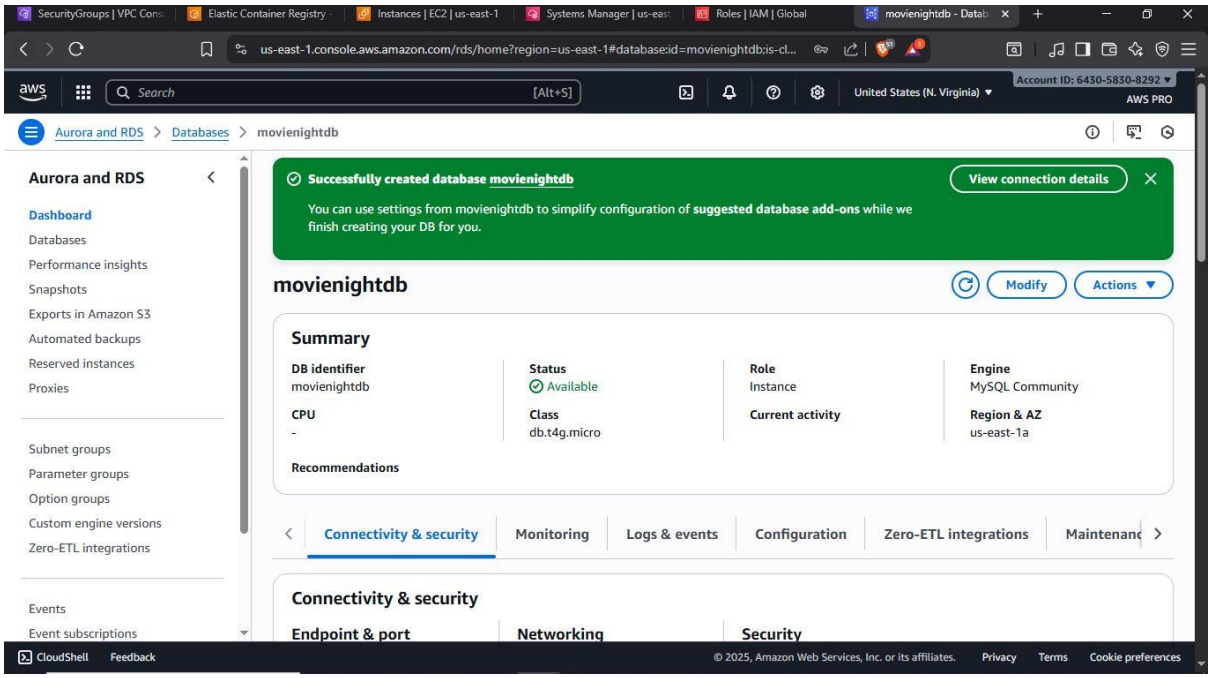
4. Push Image to ECR

Authenticated to ECR and **pushed the Docker image** to the repository for later use in ECS tasks.



5. Database Setup (RDS MySQL)

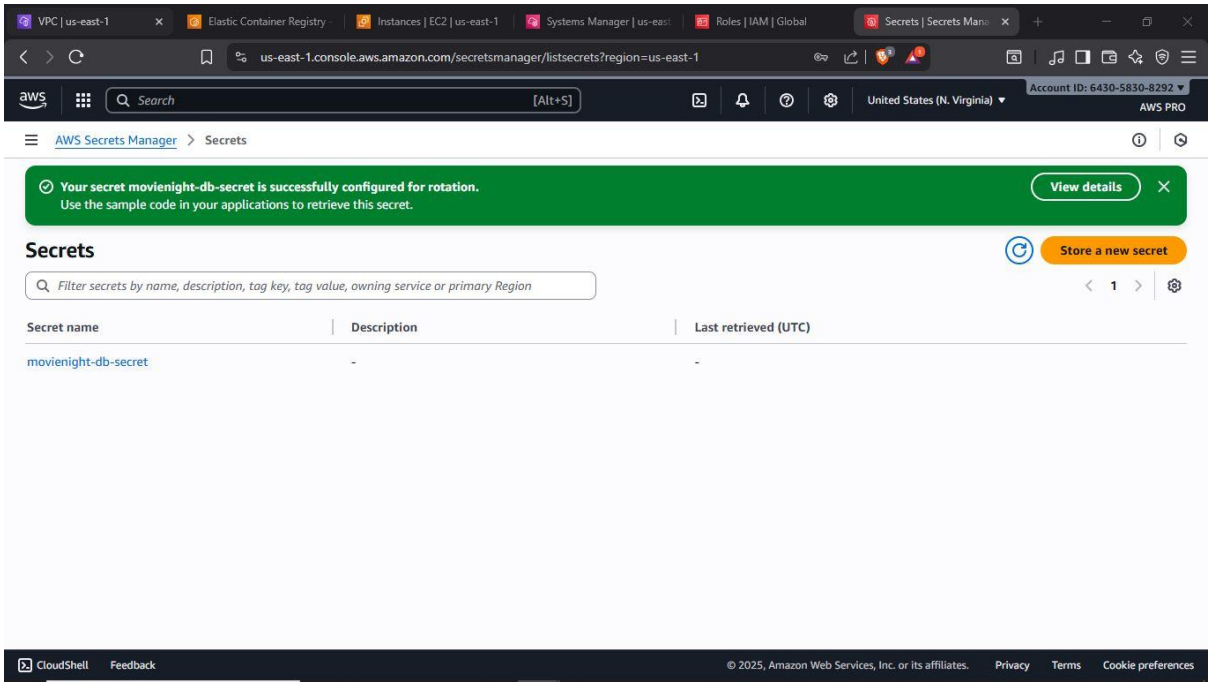
- Defined **subnet groups** for database placement within private subnets.
- Deployed an **RDS MySQL instance** with multi-AZ capability for high availability.



---

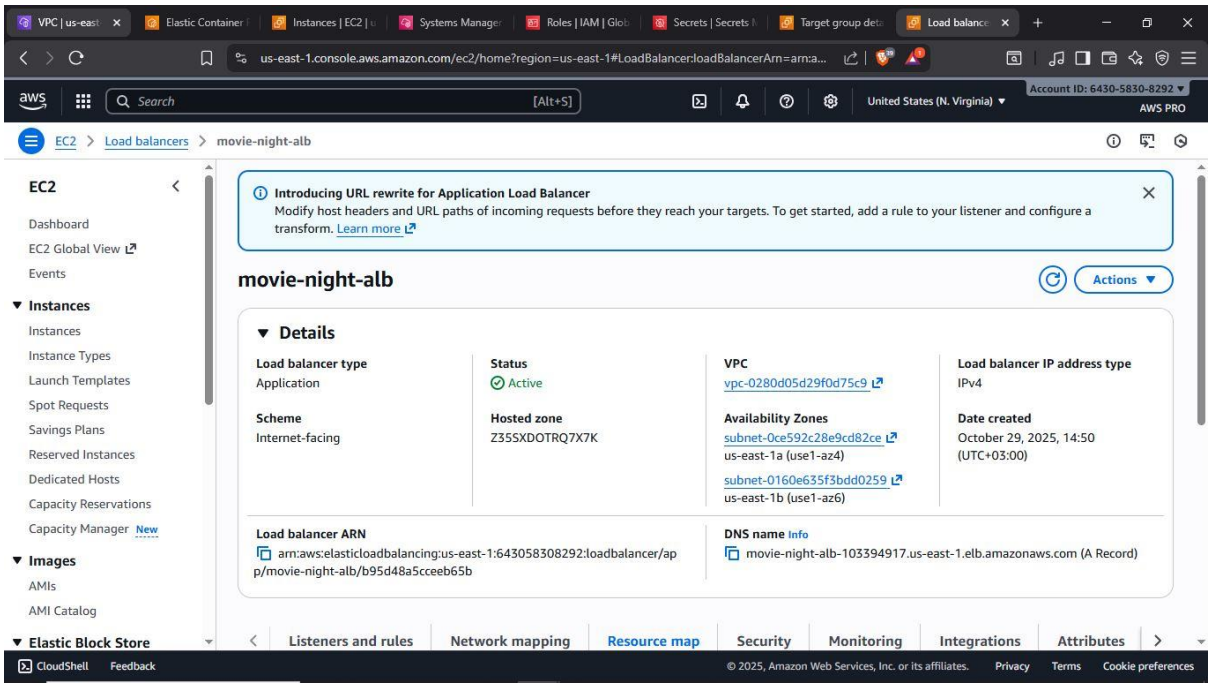
6. Secrets Management

- Configured **AWS Secrets Manager** to securely store database credentials.
- Enabled **automatic key rotation** via a **Lambda function**, reducing manual overhead and improving security posture.



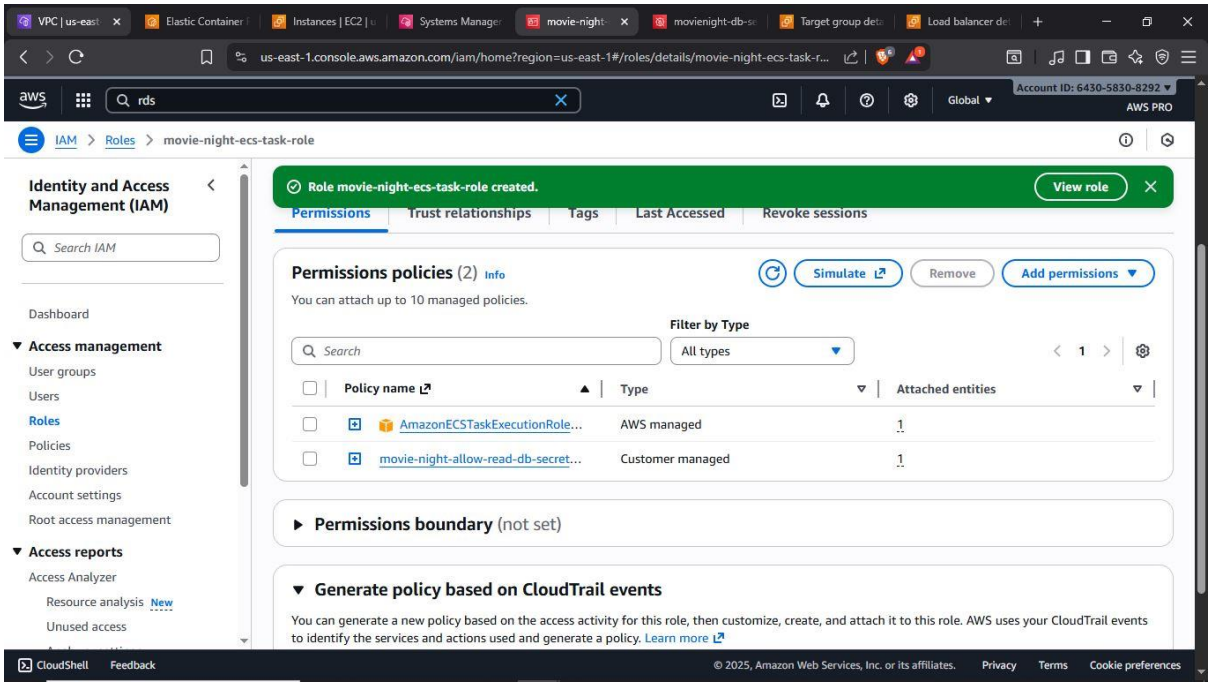
7–8. Load Balancer Configuration

- Created a **target group** for ECS tasks.
- Deployed an **Application Load Balancer (ALB)** in the public subnets to distribute inbound traffic across containers running on Fargate.



9. IAM Roles and Policies

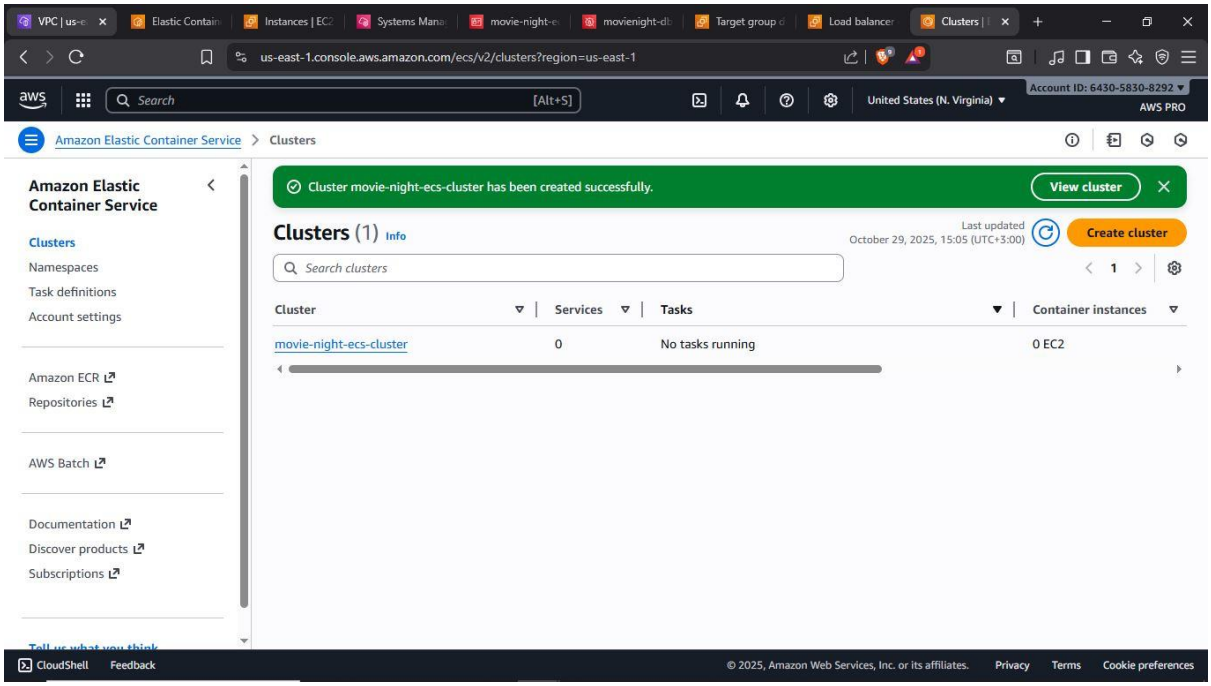
- Created a **custom IAM policy** (movie-night-allow-read-db-secrets-policy) to grant ECS tasks permission to read secrets.
- Created an **ECS Task Role** and attached:
  - AmazonECSTaskExecutionRolePolicy (managed)
  - Custom secrets access policy



---

10. ECS Cluster (Fargate)

Created an **ECS cluster** configured to use **AWS Fargate**, allowing fully managed, serverless container execution.

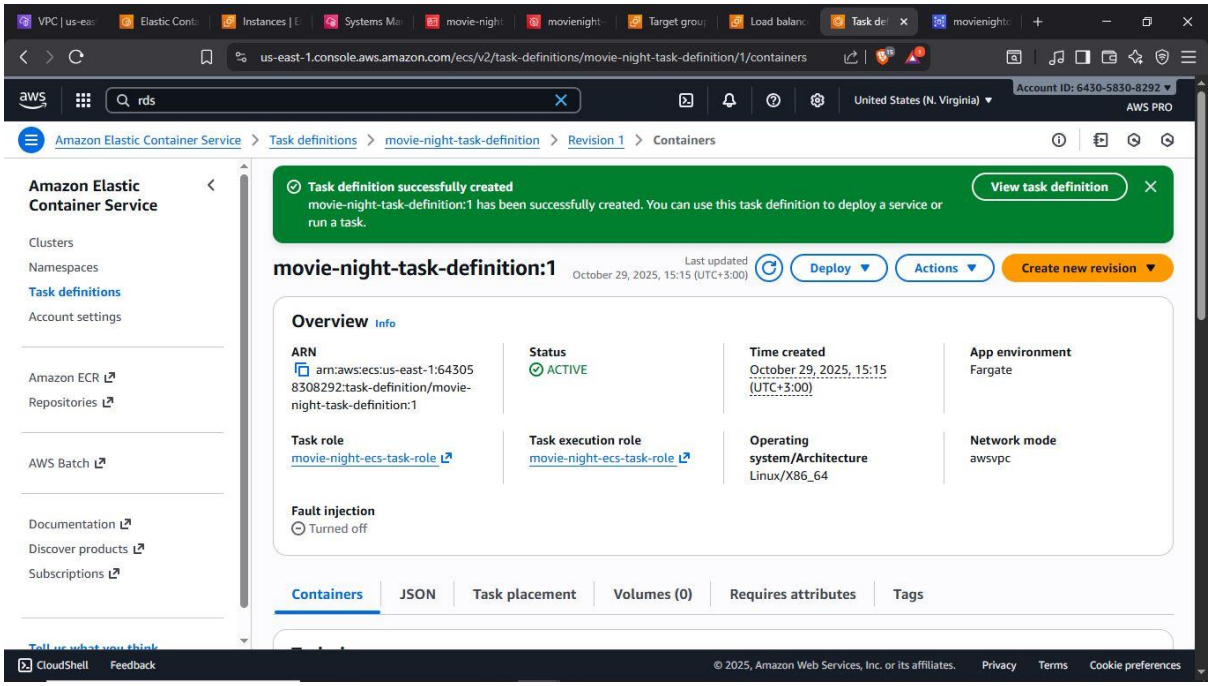


---

11. Task Definition

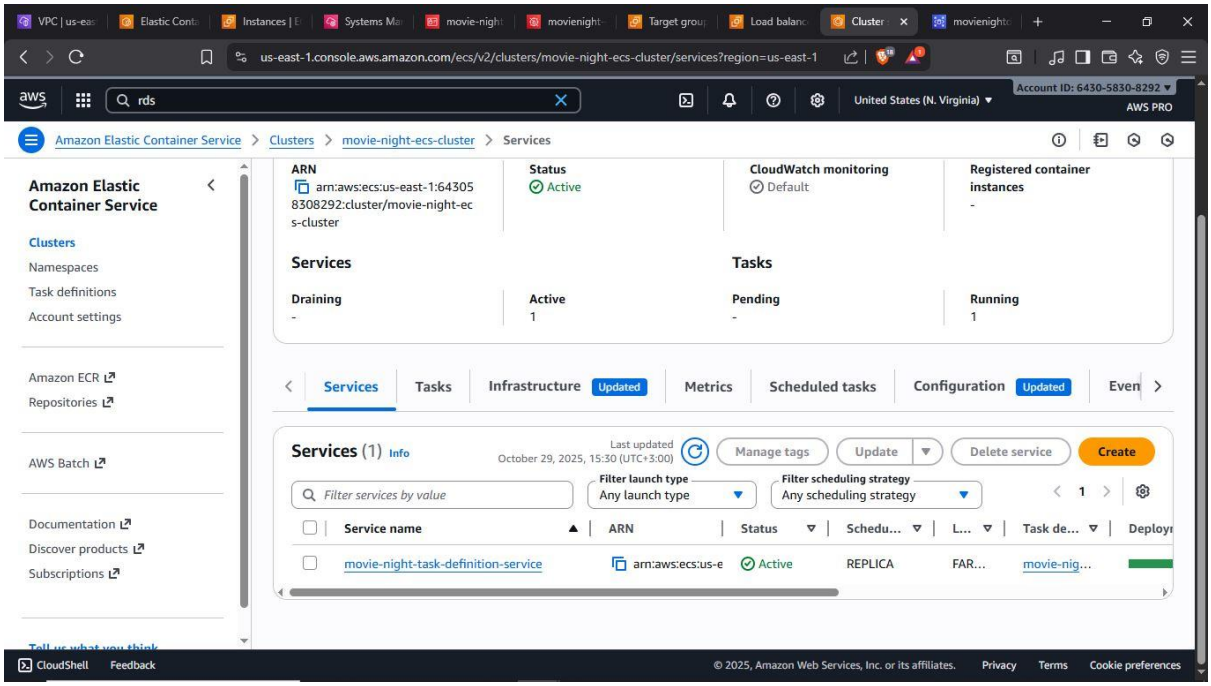
Defined a **task definition** specifying:

- Launch type (Fargate)
- Container image (from ECR)
- Port mappings
- IAM roles
- Environment variables and secrets references



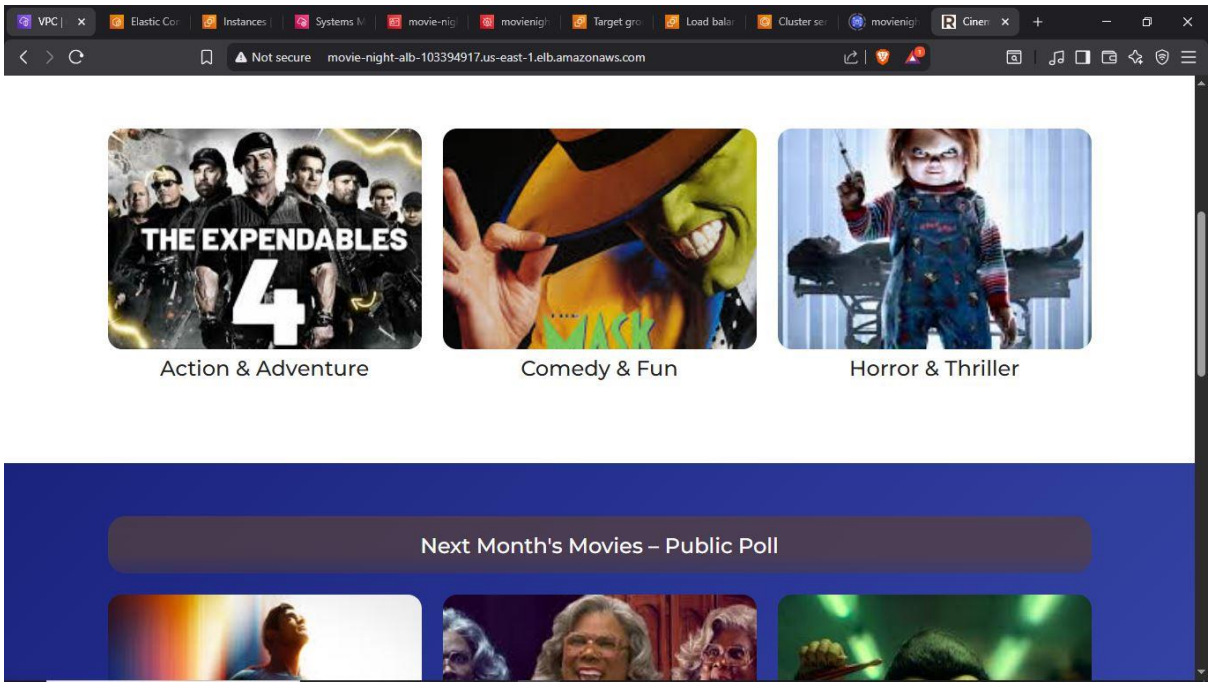
12. ECS Service Deployment

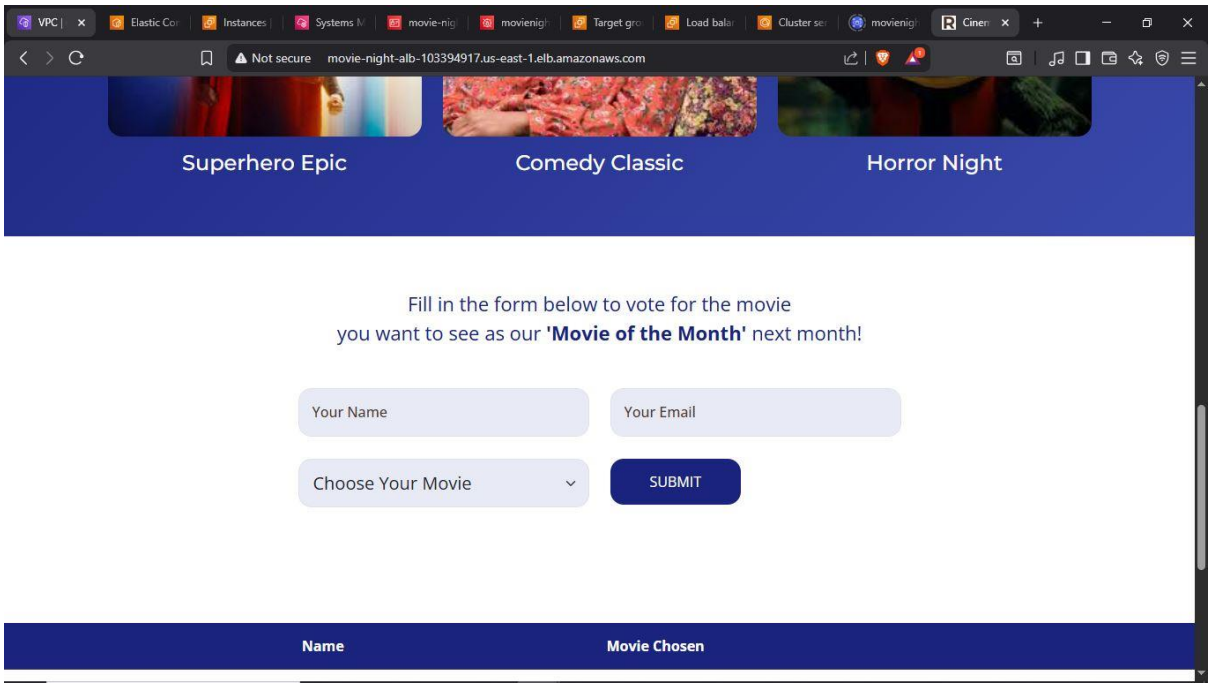
- Configured the **ECS service** for deployment, networking, load balancing, and **auto-scaling**.
- Verified that containers were successfully deployed to the cluster and accessible via the ALB endpoint.

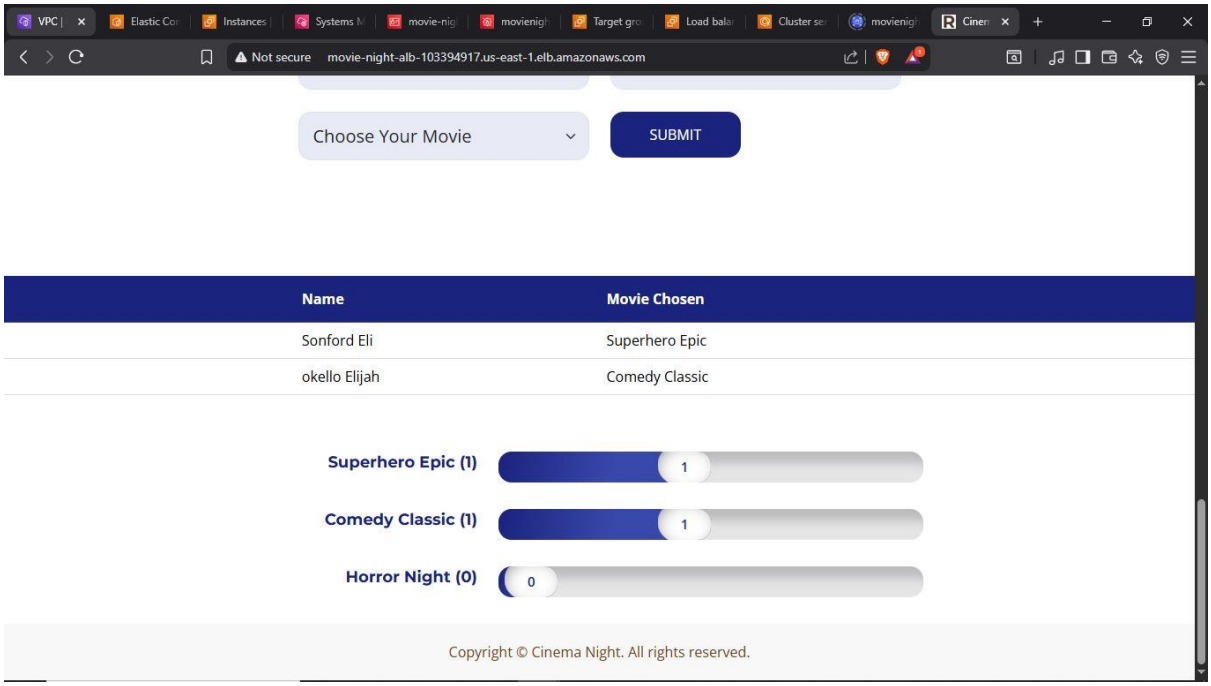


Outcome

The deployment successfully delivered a **secure, fully managed, and scalable containerized web application** on AWS Fargate, integrating best practices for networking, secrets management, and load balancing.






Not secure
movie-night-alb-103394917.us-east-1.elb.amazonaws.com
Choose Your Movie
SUBMIT
Name
Movie Chosen
Sonford Eli
Superhero Epic
okello Elijah
Comedy Classic
Superhero Epic (1)
1
Comedy Classic (1)
1
Horror Night (0)
0
Copyright © Cinema Night. All rights reserved.