CLOUD COMPUTING FINAL PROJECT REPORT

(End-to-End Deployment of a Cloud-Hosted Application on AWS)

Course: Cloud Computing

Team members:

1. Muhammad Ahtisham (22i-2690)

2. Anhar Munir (22i-2481)

3. Usman Ghani (22i-8796)

PROJECT LINKS.

GitHub Repository: https://github.com/Ahtisham992/photo_gallery.git

Frontend URL: http://photo-gallery.eu-north-1.elasticbeanstalk.com

Backend API URL: http://16.16.65.200:5000

Backend Health Check: http://16.16.65.200:5000/api/health

Contents

1. PROJECT OVERVIEW

1.1 Introduction

This project demonstrates the end-to-end deployment of a full-stack Photo Gallery application on Amazon Web Services (AWS). The application allows users to register, login, upload photos, organize them into albums, and share them publicly or keep them private.

1.2 Project Objectives

- Design and deploy a cloud-native application following AWS best practices
- Implement separate frontend and backend services
- Utilize AWS managed services for database, storage, and hosting
- Apply security best practices including IAM roles and Security Groups
- Demonstrate CRUD operations with a real-world use case
- Ensure public accessibility while maintaining security

1.3 Application Description

Photo Gallery is a full-stack web application that provides:

User Management:

- User registration with secure password hashing
- JWT-based authentication
- Session management

Photo Management:

- Upload photos with metadata (title, description, tags)
- View all public photos or personal photos

• Edit and delete owned photos

• Public/Private visibility settings

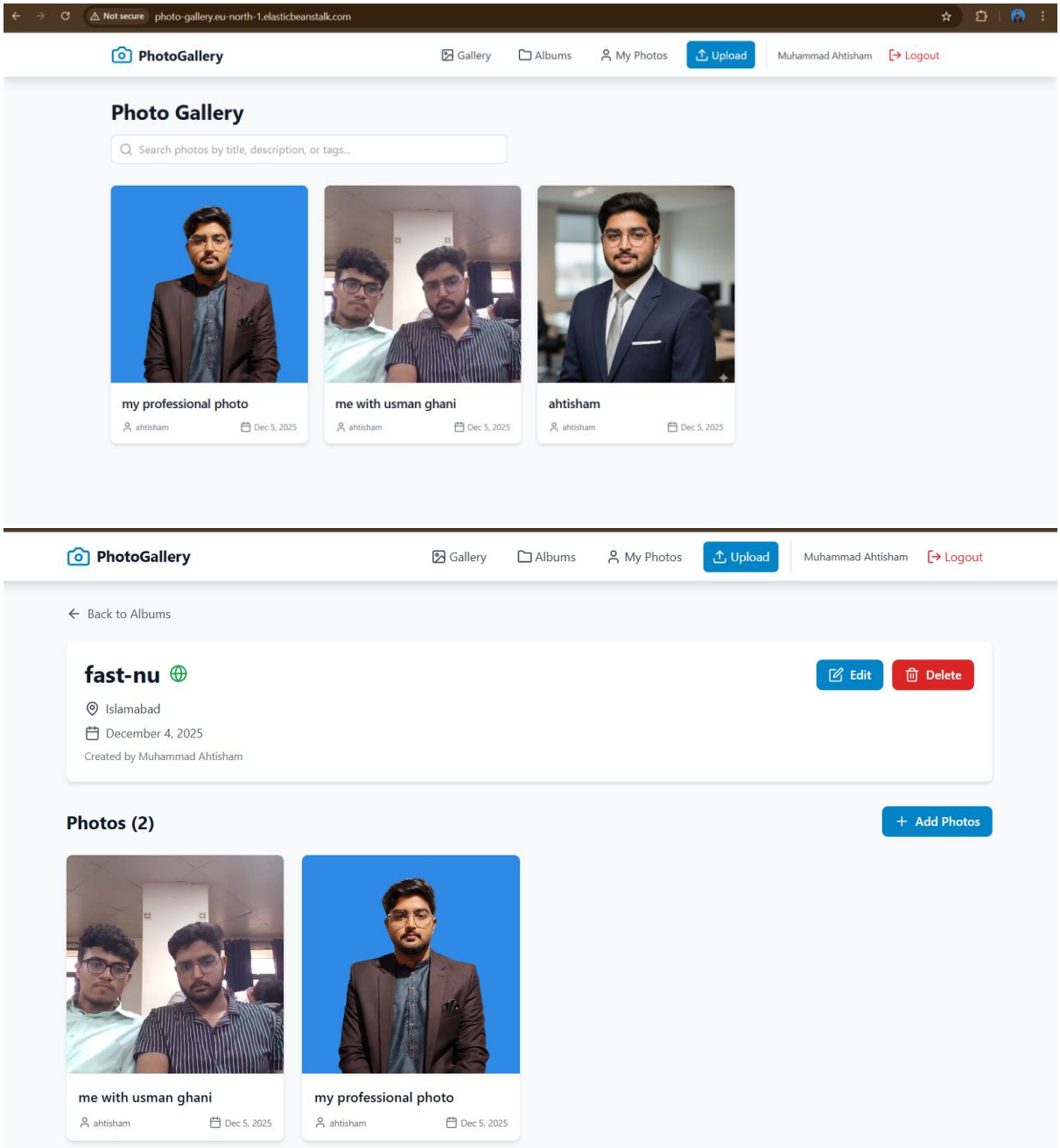
Album Management:

• Create albums with name, description, and location

• Add photos to albums

• Organize photos by trips, events, or locations

• Public/Private album settings



2. APPLICATION ARCHITECTURE

2.1 Two-Tier Architecture

The application follows a two-tier architecture with clear separation between presentation and business logic layers:

Tier 1 - Frontend (Presentation Layer):

- React-based Single Page Application (SPA)
- Deployed on AWS Elastic Beanstalk
- Handles user interface and user interactions
- Makes REST API calls to backend

Tier 2 - Backend (Business Logic Layer):

- Node.js REST API with Express framework
- Deployed on Amazon EC2 using Docker
- Handles authentication, business logic, and data operations
- Communicates with RDS database and S3 storage

2.2 Architecture Components

Component	Technology	AWS Service
Frontend	React + Vite	AWS Elastic Beanstalk
Backend	Node.js + Express	Amazon EC2
Database	MySQL	Amazon RDS
File Storage	-	Amazon S3
Networking	-	VPC, Security Groups
Access Control	-	IAM Roles & policies

2.3 Data Flow

1. User accesses the frontend via Elastic Beanstalk URL
2. Frontend makes REST API calls to the backend on EC2
3. Backend authenticates requests using JWT tokens
4. Backend queries/updates data in RDS MySQL database
5. Photos are uploaded to and retrieved from S3 bucket
6. Response is sent back to the frontend for display

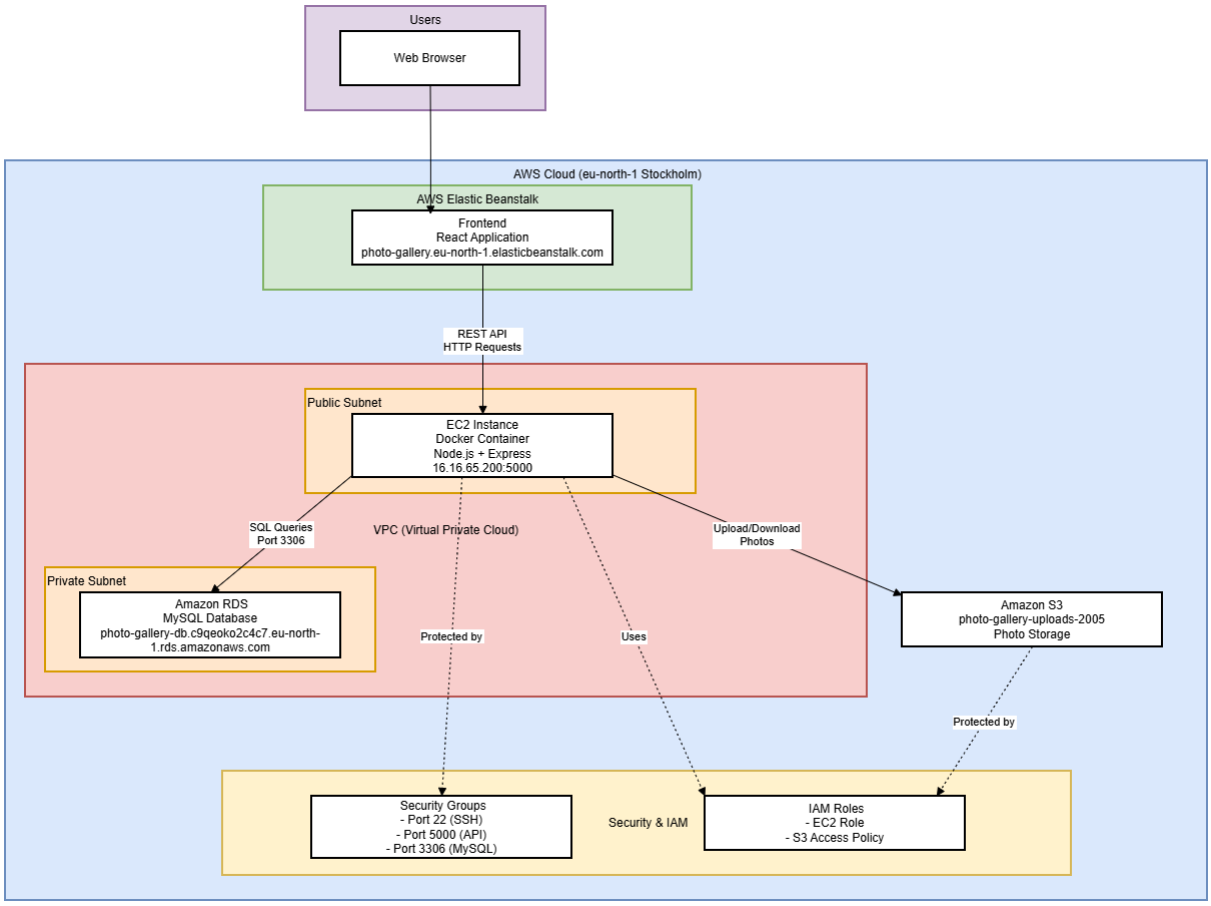


Figure 1 Architecture diagram

3. AWS SERVICES USED

Service	Configuration	Purpose
Amazon EC2 with docker	T2.micro, Amazon Linux 2	Backend Hosting
Amazon RDS service	Db.t3.micro, MySQL 8.0	Managed Database
Amazon S3	SSC	Photo file storage
Elastic Beanstalk	Node.js-20	Frontend hosting
VPC	Default	Network isolation
IAM	Roles and policies	Access management

Region: eu-north-1 (Stockholm)

All services are configured within AWS Free Tier limits to minimize costs.

4. DATABASE SETUP (AMAZON RDS)

4.1 RDS Configuration

Database Engine: MySQL 8.0.35

Instance Class: db.t3.micro (Free Tier eligible)

Storage: 20 GB General Purpose SSD (gp2)

Multi-AZ: No (Single instance for development)

Public Access: No (Private subnet)

Backup Retention: 7 days

Database Name: photo_gallery

4.2 Database Endpoint

Endpoint: photo-gallery-db.c9qeoko2c4c7.eu-north-1.rds.amazonaws.com

Port: 3306

Master Username: admin

4.3 Database Schema

The application uses three main tables:

Users Table:

- id (Primary Key)
- username
- email (Unique)
- password (Hashed with bcrypt)
- createdAt
- updatedAt

Photos Table:

- id (Primary Key)
- title
- description
- filepath
- tags
- isPublic (Boolean)
- userId (Foreign Key)
- albumId (Foreign Key, nullable)
- createdAt

• updatedAt

Albums Table:

• id (Primary Key)

• name

• description

• location

• eventDate

• isPublic (Boolean)

• userId (Foreign Key)

• coverPhotoId (Foreign Key, nullable)

• createdAt

• updatedAt

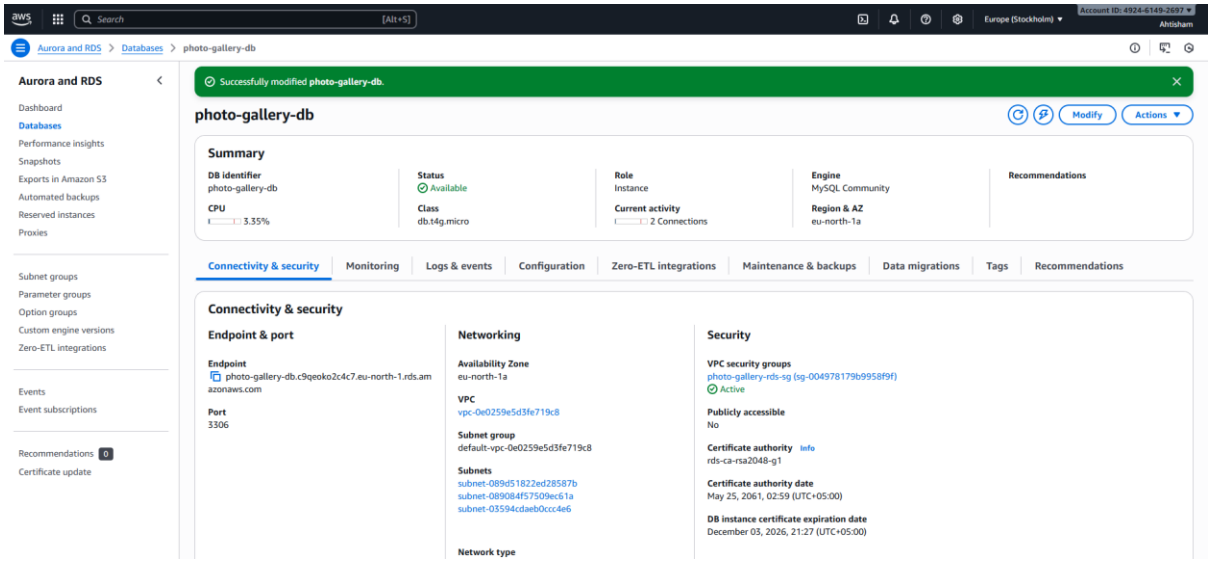
4.4 Security Configuration

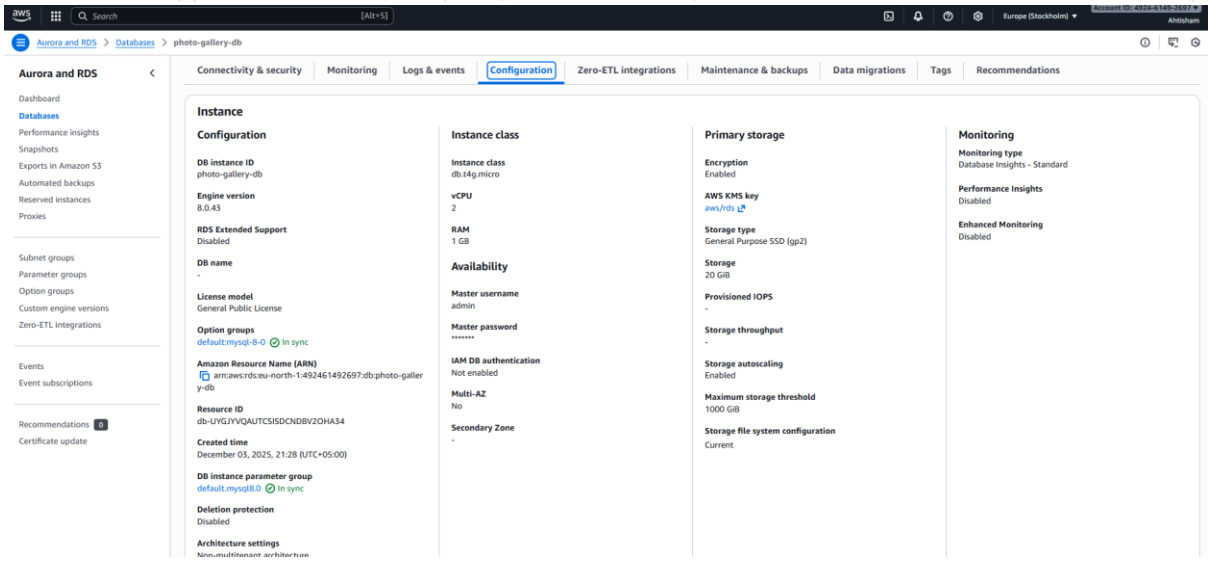
• Database is placed in a private subnet

• Security Group allows inbound traffic only from EC2 instance

• SSL/TLS encryption enabled for data in transit

• Automated backups enabled





5. STORAGE SETUP (AMAZON S3)

5.1 S3 Bucket Configuration

Bucket Name: photo-gallery-uploads-2005

Region: eu-north-1 (Stockholm)

Storage Class: S3 Standard

Versioning: Disabled

Encryption: Server-side encryption (SSE-S3)

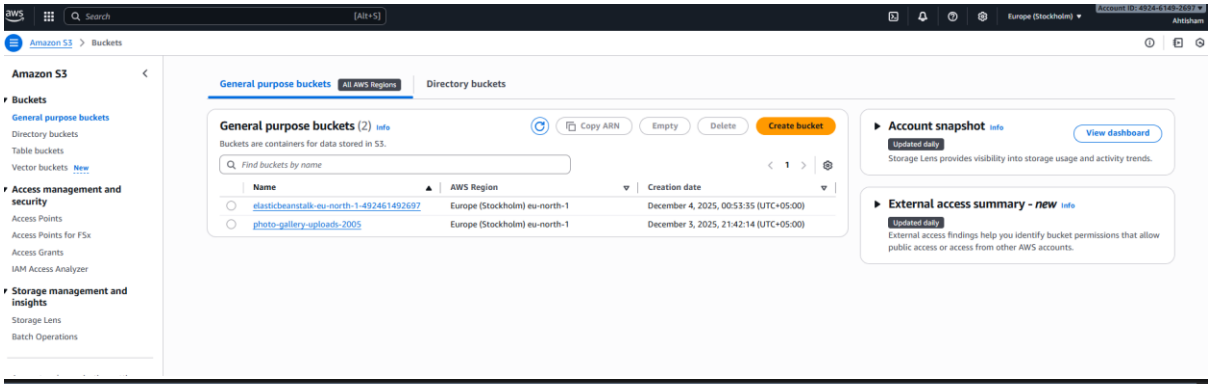
5.2 Bucket Policy

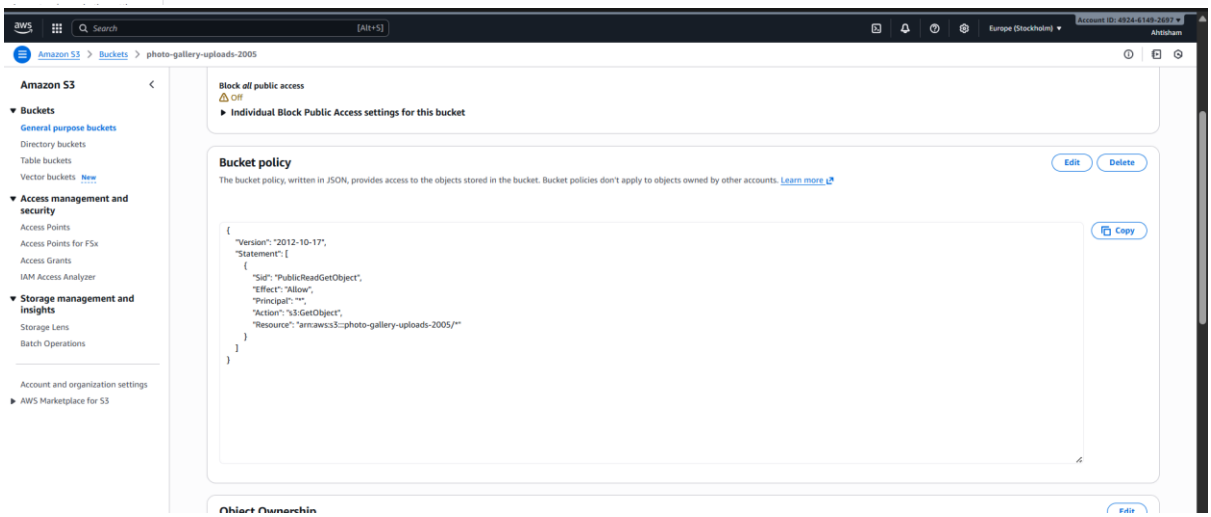
The bucket is configured with a policy that allows:

- EC2 instance to upload photos (via IAM role)
- Public read access to photo objects
- Restricted write access

5.3 Upload Process

1. User uploads photo through frontend
2. Frontend sends multipart form data to backend API
3. Backend validates file (type, size)
4. Backend generates unique filename using UUID
5. Backend uploads to S3 using AWS SDK
6. S3 returns file URL
7. Backend saves photo metadata to RDS with S3 URL
8. Frontend displays uploaded photo





6. BACKEND DEPLOYMENT (AMAZON EC2)

6.1 EC2 Instance Configuration

Instance Type: t2.micro (Free Tier eligible)

AMI: Amazon Linux 2023

vCPUs: 1

Memory: 1 GB

Storage: 8 GB EBS (gp2)

Public IP: 16.16.65.200

Region: eu-north-1 (Stockholm)

6.2 Security Group Configuration

Inbound Rules:

• SSH (Port 22) - Source: My IP (for management)

• Custom TCP (Port 5000) - Source: 0.0.0.0/0 (for API access)

Outbound Rules:

• All traffic allowed

6.3 Docker Deployment

The backend is containerized using Docker for consistency and portability.

Dockerfile Configuration:

• Base Image: node:18-alpine

• Working Directory: /app

• Dependencies: Installed via npm

• Exposed Port: 5000

• Start Command: npm start

Docker Run Command:

```
docker run -d \
  --name photo-gallery-api \
  -p 5000:5000 \
  -e NODE_ENV=production \
  -e DB_HOST=photo-gallery-db.c9qeoko2c4c7.eu-north-1.rds.amazonaws.com
\
  -e DB_USER=admin \
```

```
-e DB_PASSWORD=[password] \
-e DB_NAME=photo_gallery \
-e USE_S3=true \
-e S3_BUCKET_NAME=photo-gallery-uploads-2005 \
--restart unless-stopped \
photo-gallery-backend
```

6.4 API Endpoints

Authentication:

- POST /api/auth/register - User registration
- POST /api/auth/login - User login
- GET /api/auth/me - Get current user

Photos:

- GET /api/photos - Get all photos
- GET /api/photos/:id - Get single photo
- POST /api/photos - Upload photo
- PUT /api/photos/:id - Update photo
- DELETE /api/photos/:id - Delete photo

Albums:

- GET /api/albums - Get all albums
- GET /api/albums/:id - Get single album
- POST /api/albums - Create album
- PUT /api/albums/:id - Update album
- DELETE /api/albums/:id - Delete album

• POST /api/albums/:id/photos/:photoId - Add photo to album

• DELETE /api/albums/:id/photos/:photoId - Remove photo from album

Health Check:

• GET /api/health - Backend health status

6.5 Environment Variables

All sensitive configuration is stored in environment variables:

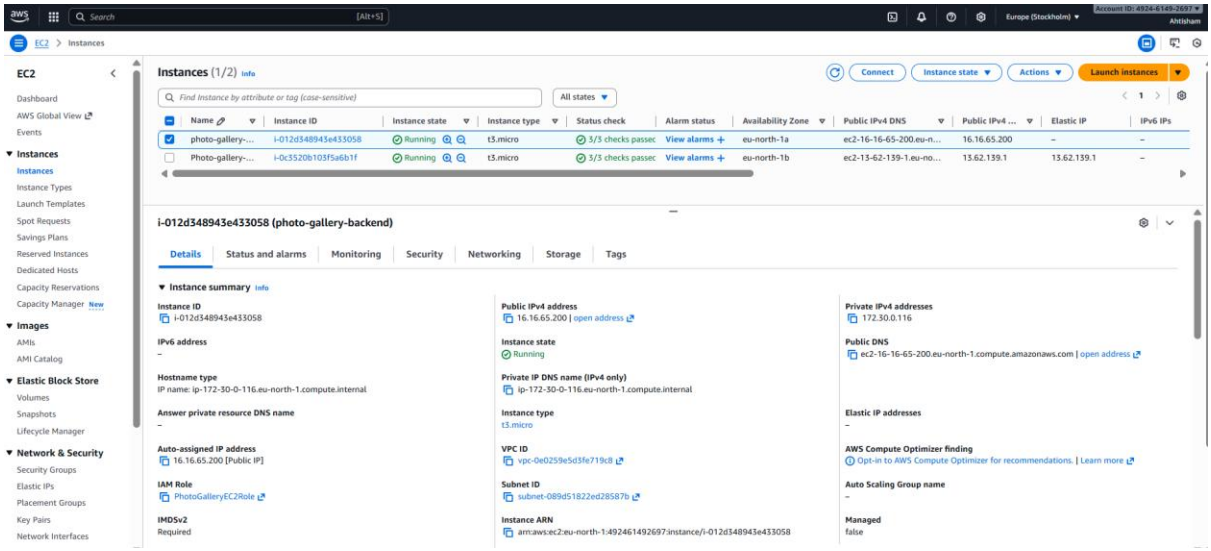
• Database credentials

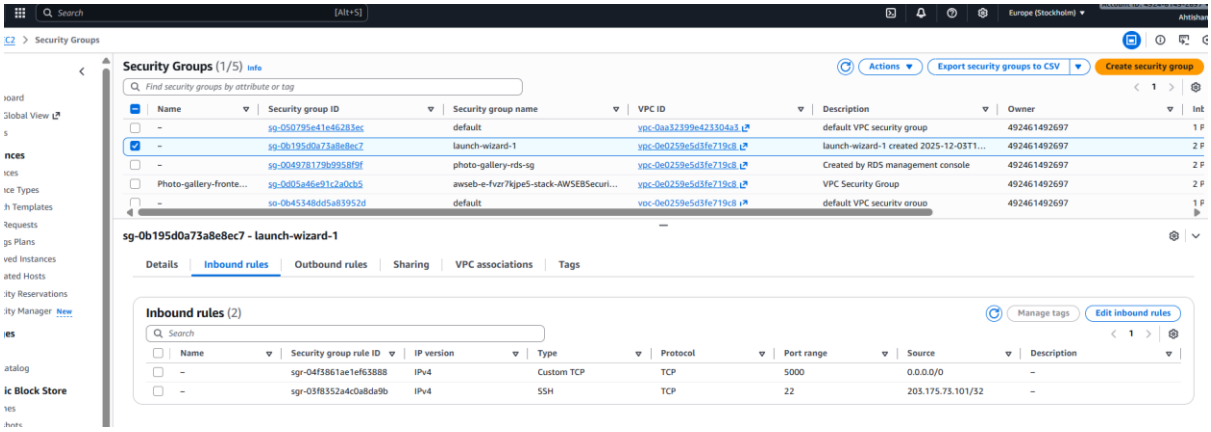
• JWT secret key

• AWS credentials (via IAM role)

• S3 bucket name

• Node environment






```
Connection to 16.16.65.200 closed.
PS C:\Users\shamii\Downloads> ssh -i "C:\Users\shamii\Downloads\photo-gallery-key.pem" ec2-user@16.16.65.200
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Thu Dec  4 20:05:09 2025 from 203.175.73.101
[ec2-user@ip-172-30-0-116 ~]$ docker logs photo-gallery-api
Database connection established successfully.
Database models synchronized.
Server is running on port 5000
Environment: production
[ec2-user@ip-172-30-0-116 ~]$
```

7. FRONTEND DEPLOYMENT (ELASTIC BEANSTALK)

7.1 Elastic Beanstalk Configuration

Application Name: photo-gallery-frontend

Environment Name: Photo-gallery-frontend-env

Platform: Node.js 20 running on 64bit Amazon Linux 2023

Platform Version: 6.7.0

Instance Type: t2.micro (Free Tier eligible)

Environment URL: photo-gallery.eu-north-1.elasticbeanstalk.com

7.2 Deployment Package

The frontend is built and packaged for deployment:

Build Process:

1. npm run build (creates optimized production build)
2. Creates build/ directory with static files
3. Adds package.json with serve dependency
4. Adds .ebextensions/ for Nginx configuration
5. Packages into deploy.zip

Package Contents:

- index.html
- assets/ (JS and CSS files)
- package.json (with serve script)
- .ebextensions/00_nginx.config (SPA routing)

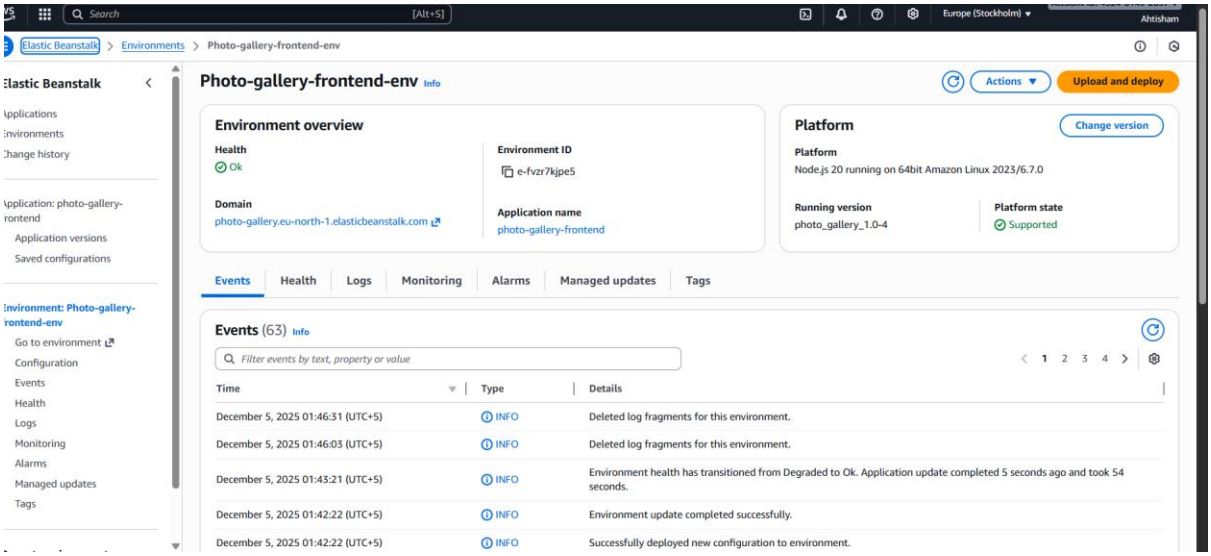
7.3 Nginx Configuration

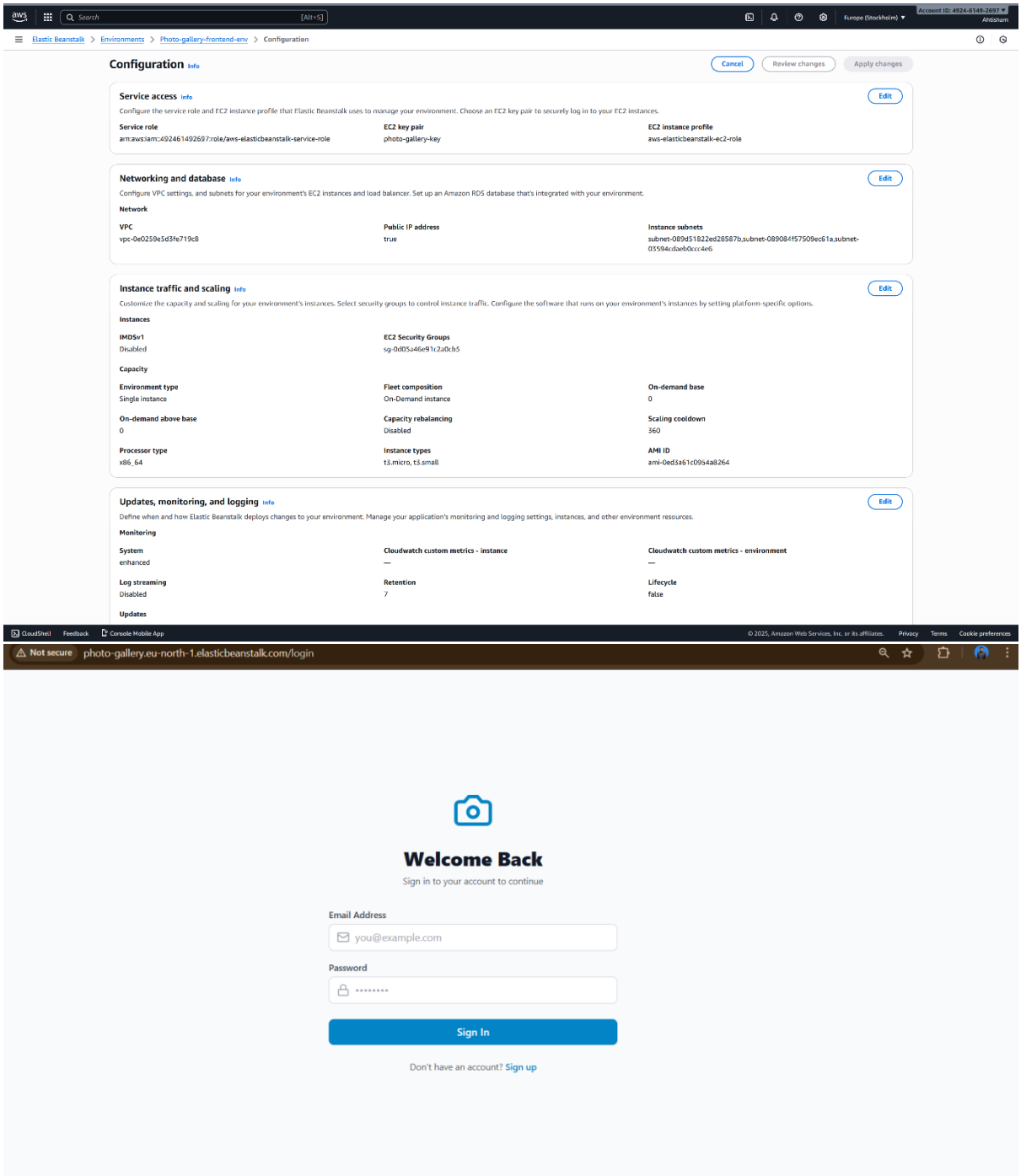
Custom Nginx configuration for React Router:

- Serves all routes through index.html
- Enables client-side routing
- Prevents 404 errors on page refresh

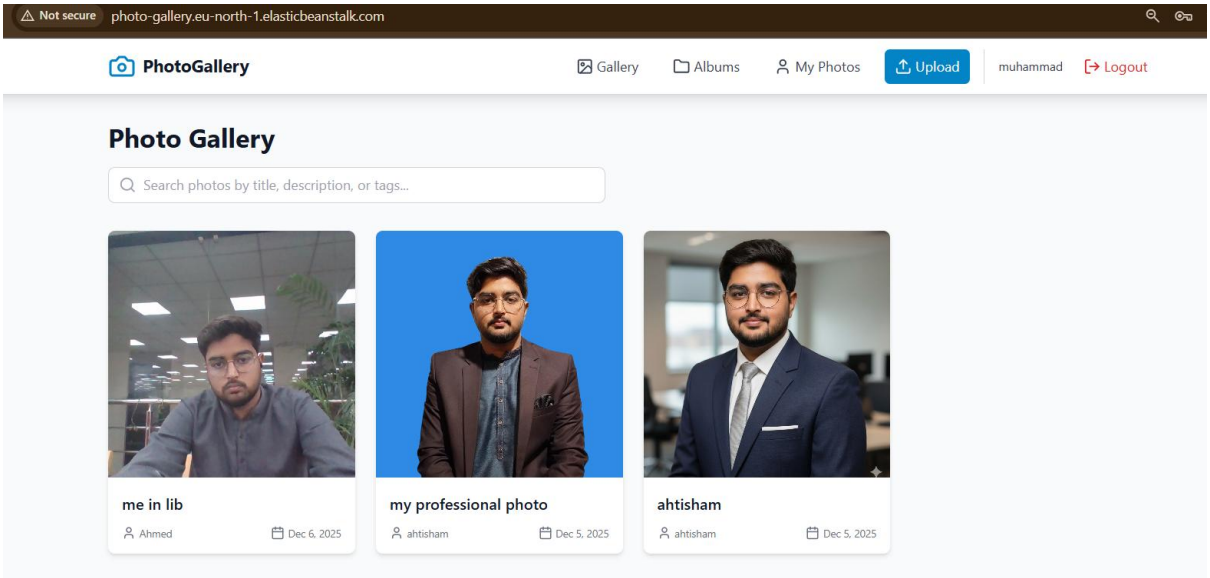
7.4 Environment Properties

- VITE_API_URL: http://16.16.65.200:5000 (Backend URL)




Elastic Beanstalk > Environments > Photo-gallery-frontend-env > Configuration
Configuration Info
Cancel
Review changes
Apply changes
Service access Info
Edit
Configure the service role and EC2 instance profile that Elastic Beanstalk uses to manage your environment. Choose an EC2 key pair to securely log in to your EC2 instances.
Service role
arn:aws:iam::492461492697:role/aws-elasticbeanstalk-service-role
EC2 key pair
photo-gallery-key
EC2 instance profile
aws-elasticbeanstalk-ec2-role
Networking and database Info
Edit
Configure VPC settings, and subnets for your environment's EC2 instances and load balancer. Set up an Amazon RDS database that's integrated with your environment.
Network
VPC
vpc-0e0259e5d3fe719c8
Public IP address
true
Instance subnets
subnet-089d51822ed28587b,subnet-089084f57509ec61a,subnet-03594cdaeb0ccc4e6
Instance traffic and scaling Info
Edit
Customize the capacity and scaling for your environment's instances. Select security groups to control instance traffic. Configure the software that runs on your environment's instances by setting platform-specific options.
Instances
IMDSv1
Disabled
EC2 Security Groups
sg-0d05a46e91c2a0cb5
Capacity
Environment type
Single instance
Fleet composition
On-Demand instance
On-demand base
0
On-demand above base
0
Capacity rebalancing
Disabled
Scaling cooldown
360
Processor type
x86_64
Instance types
t3.micro, t3.small
AMI ID
ami-0ed3a61c09s4a8264
Updates, monitoring, and logging Info
Edit
Define when and how Elastic Beanstalk deploys changes to your environment. Manage your application's monitoring and logging settings, instances, and other environment resources.
Monitoring
System
enhanced
Cloudwatch custom metrics - instance
—
Cloudwatch custom metrics - environment
—
Log streaming
Disabled
Retention
7
Lifecycle
false
Updates
CloudShell
Feedback
Console Mobile App
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Not secure
photo-gallery.eu-north-1.elasticbeanstalk.com/login
Welcome Back
Sign in to your account to continue
Email Address
you@example.com
Password
Sign In
Don't have an account? Sign up



8. SECURITY CONFIGURATION

8.1 Network Security

VPC Configuration:

- Created new VPC for the deployment
- Public subnet for EC2 and Elastic Beanstalk
- Private subnet for RDS database
- Internet Gateway for public access
- Route tables configured appropriately

Security Groups:

1. EC2 Security Group:

- Inbound: SSH (22), HTTP (5000)
- Outbound: All traffic

2. RDS Security Group:

- Inbound: MySQL (3306) from EC2 security group only
- Outbound: All traffic

3. Elastic Beanstalk Security Group:

- Inbound: HTTP (80), HTTPS (443)
- Outbound: All traffic

8.2 Application Security

Authentication:

- JWT (JSON Web Tokens) for stateless authentication
- Tokens expire after 24 hours
- Secure password hashing using bcrypt (10 salt rounds)

Authorization:

- Protected routes require valid JWT token
- Users can only modify their own photos and albums
- Public/private access control for photos and albums

Input Validation:

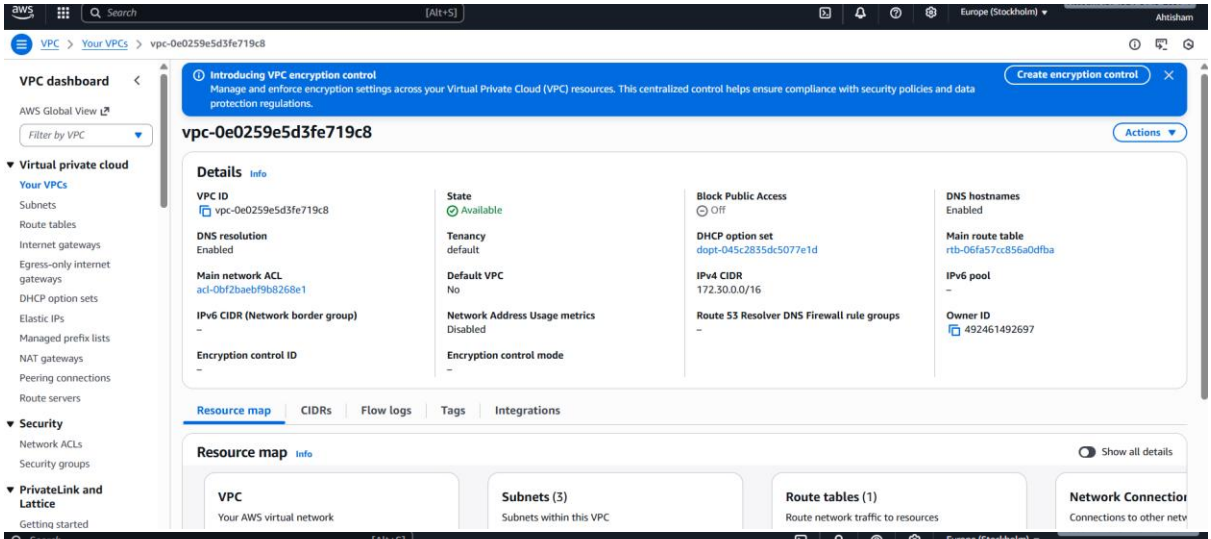
- Server-side validation for all inputs
- File type validation (only images allowed)
- File size limits (5MB maximum)
- SQL injection prevention via Sequelize ORM

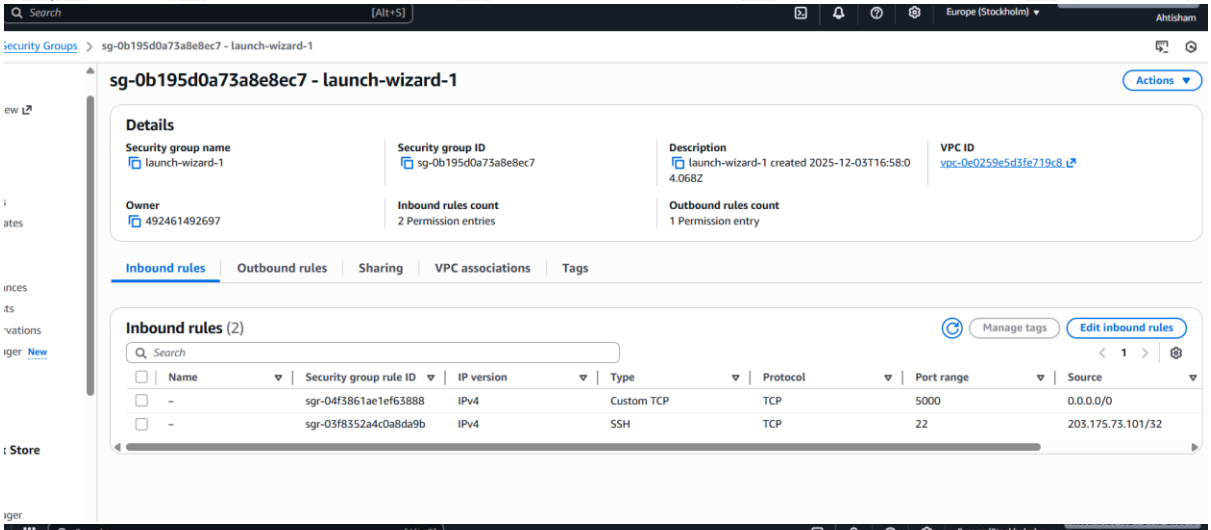
CORS Configuration:

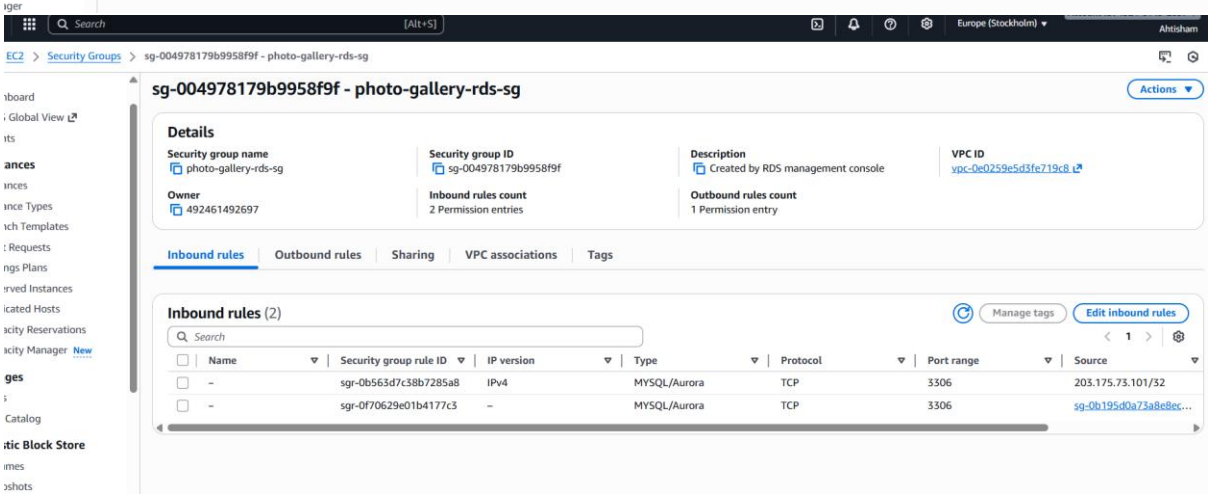
- Configured to allow requests from Elastic Beanstalk URL
- Credentials enabled for cookie-based sessions

8.3 Data Security

- Database credentials stored in environment variables
- RDS in private subnet (not publicly accessible)
- SSL/TLS encryption for data in transit
- S3 server-side encryption for data at rest
- No sensitive data in source code or Git repository


VPC dashboard
Introducing VPC encryption control
Create encryption control
vpc-0e0259e5d3fe719c8
Actions
Details
VPC ID
vpc-0e0259e5d3fe719c8
State
Available
Block Public Access
Off
DNS hostnames
Enabled
DNS resolution
Enabled
Tenancy
default
DHCP option set
dopt-045c2835dc5077e1d
Main route table
rtb-06fa57cc856a0dfba
Main network ACL
acl-0bf2baebf9b8268e1
Default VPC
No
IPv4 CIDR
172.30.0.0/16
IPv6 pool
Owner ID
492461492697
Resource map
CIDRs
Flow logs
Tags
Integrations
Resource map
Show all details
VPC
Your AWS virtual network
Subnets (3)
Subnets within this VPC
Route tables (1)
Route network traffic to resources


sg-0b195d0a73a8e8ec7 - launch-wizard-1
Actions
Details
Security group name
launch-wizard-1
Security group ID
sg-0b195d0a73a8e8ec7
Description
launch-wizard-1 created 2025-12-03T16:58:04.068Z
VPC ID
Owner
492461492697
Inbound rules count
2 Permission entries
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Sharing
VPC associations
Tags
Inbound rules (2)
Manage tags
Edit inbound rules
Name
Security group rule ID
IP version
Type
Protocol
Port range
Source
sgr-04f3861ae1ef63888
IPv4
Custom TCP
TCP
5000
0.0.0.0/0
sgr-03f8352a4c0a8da9b
SSH
22
203.175.73.101/32


sg-004978179b9958f9f - photo-gallery-rds-sg
Actions
Details
Security group name
photo-gallery-rds-sg
Security group ID
sg-004978179b9958f9f
Description
Created by RDS management console
VPC ID
Owner
492461492697
Inbound rules count
2 Permission entries
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Sharing
VPC associations
Tags
Inbound rules (2)
Manage tags
Edit inbound rules
Name
Security group rule ID
IP version
Type
Protocol
Port range
Source
sgr-0b563d7c38b7285a8
IPv4
MYSQL/Aurora
TCP
3306
203.175.73.101/32
sgr-0f70629e01b4177c3

9. IAM POLICIES AND ROLES

9.1 IAM Role for EC2

Role Name: EC2-S3-Access-Role

Purpose: Allow EC2 instance to access S3 bucket for photo uploads

Attached Policies:

1. AmazonS3FullAccess (AWS Managed Policy)

- Allows full access to S3 buckets
- Used for uploading and retrieving photos

Trust Relationship:

- Service: ec2.amazonaws.com
- Allows EC2 service to assume this role

9.2 IAM User for Development

User Name: photo-gallery-admin

Purpose: Development and deployment access

Attached Policies:

- AmazonEC2FullAccess
- AmazonRDSFullAccess
- AmazonS3FullAccess
- ElasticBeanstalkFullAccess

Access Keys:

• Access Key ID: AKIA... (stored securely)

• Secret Access Key: (stored securely, not in code)

9.3 S3 Bucket Policy

Policy allows:

• EC2 instance (via IAM role) to upload objects

• Public read access to photo objects

• Denies public write access

9.4 Security Best Practices Followed

✓ Principle of least privilege applied

✓ No hardcoded credentials in source code

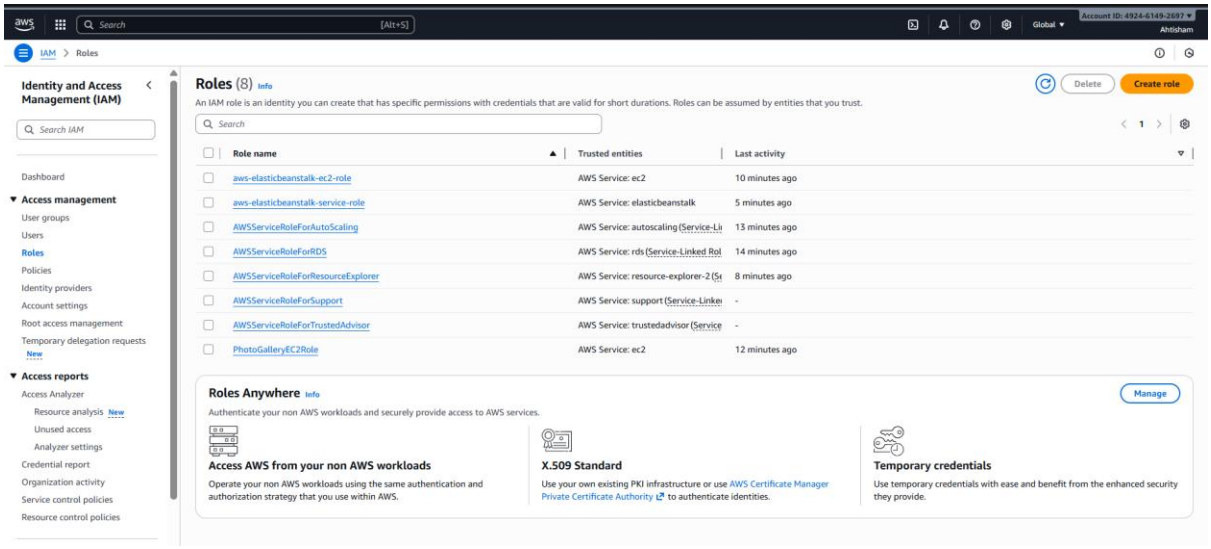
✓ Environment variables for sensitive data

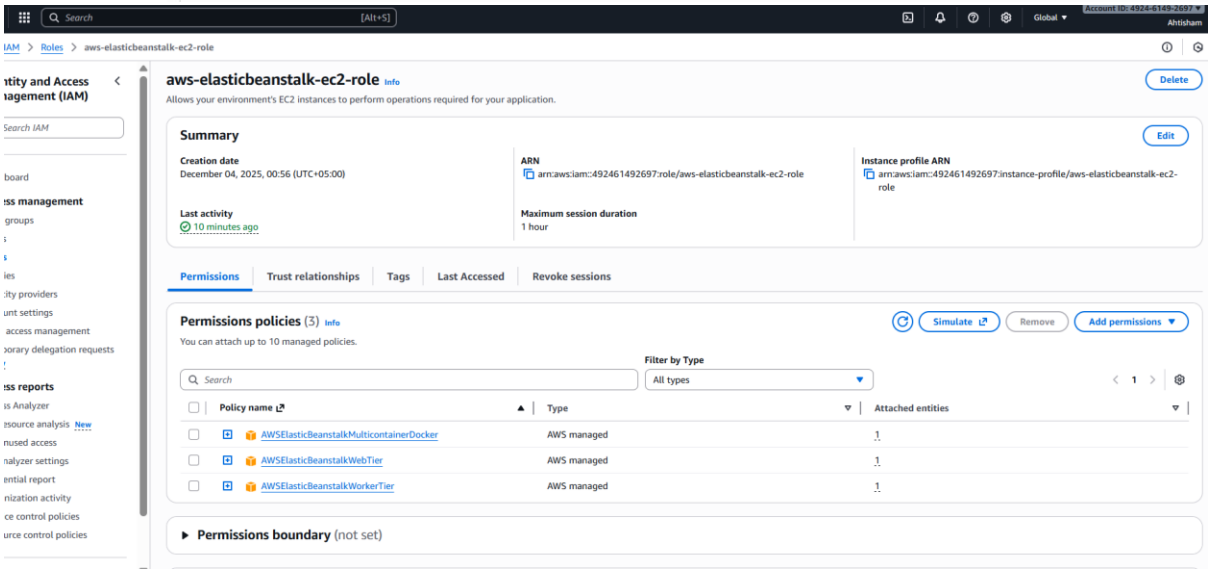
✓ IAM roles used instead of access keys where possible

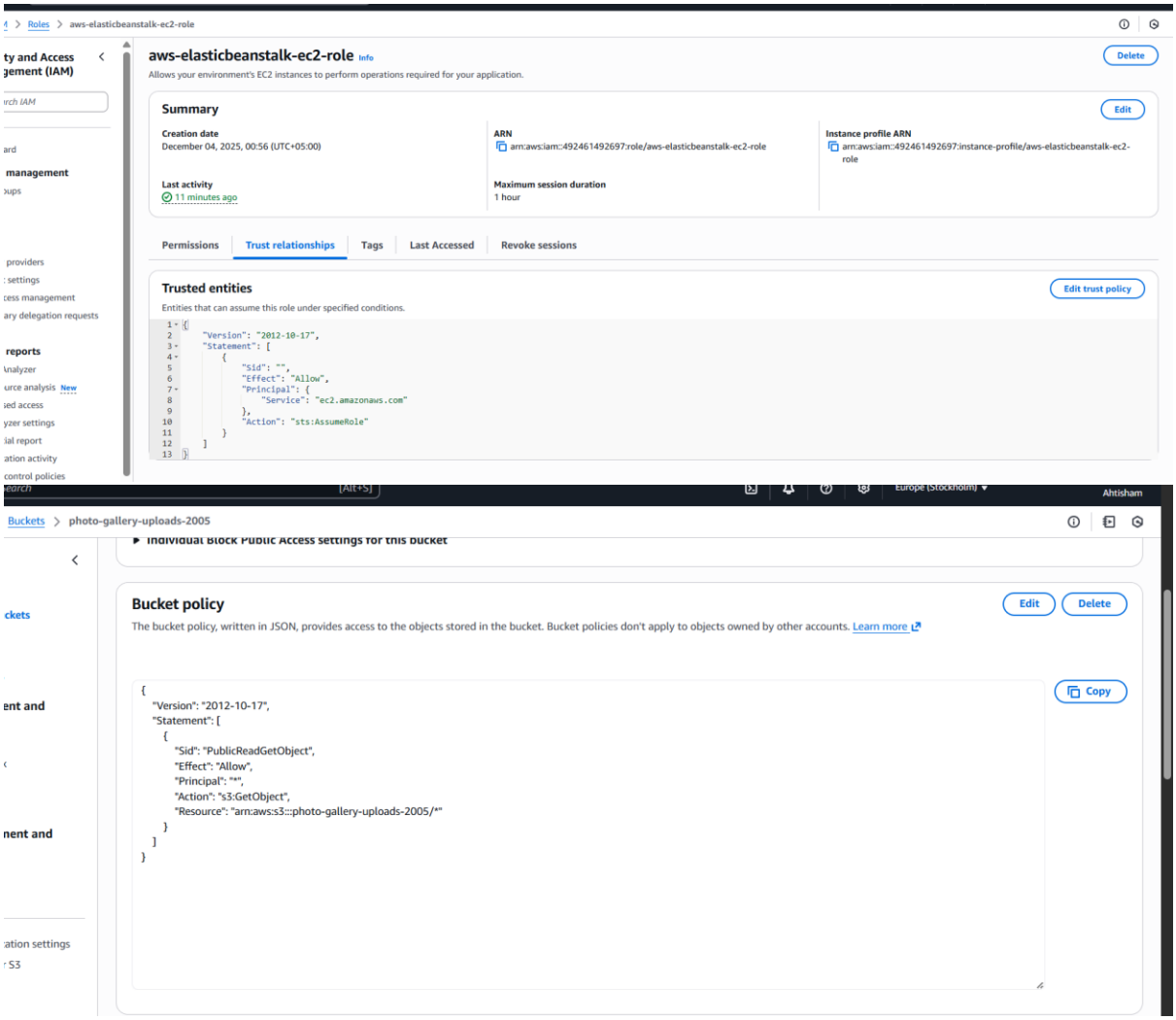
✓ Regular rotation of access keys

✓ MFA enabled on root account

✓ CloudWatch logging enabled for monitoring


Search
[Alt+S]
Global
Account ID: 4924-6149-2697
Ahtisham
IAM > Roles
Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Root access management
Temporary delegation requests
New
Access reports
Access Analyzer
Resource analysis New
Unused access
Analyzer settings
Credential report
Organization activity
Service control policies
Resource control policies
Roles (8) Info
An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.
Delete
Create role
Search
Role name
Trusted entities
Last activity
aws-elasticbeanstalk-ec2-role
AWS Service: ec2
10 minutes ago
aws-elasticbeanstalk-service-role
AWS Service: elasticbeanstalk
5 minutes ago
AWSServiceRoleForAutoScaling
AWS Service: autoscaling (Service-Li
13 minutes ago
AWSServiceRoleForRDS
AWS Service: rds (Service-Linked Rol
14 minutes ago
AWSServiceRoleForResourceExplorer
AWS Service: resource-explorer-2 (Se
8 minutes ago
AWSServiceRoleForSupport
AWS Service: support (Service-Linke
-
AWSServiceRoleForTrustedAdvisor
AWS Service: trustedadvisor (Service
-
PhotoGalleryEC2Role
AWS Service: ec2
12 minutes ago
Roles Anywhere Info
Manage
Authenticate your non AWS workloads and securely provide access to AWS services.
Access AWS from your non AWS workloads
Operate your non AWS workloads using the same authentication and authorization strategy that you use within AWS.
X.509 Standard
Use your own existing PKI infrastructure or use AWS Certificate Manager Private Certificate Authority to authenticate identities.
Temporary credentials
Use temporary credentials with ease and benefit from the enhanced security they provide.


Search
[Alt+S]
Global
Account ID: 4924-6149-2697
Ahtisham
IAM > Roles > aws-elasticbeanstalk-ec2-role
aws-elasticbeanstalk-ec2-role Info
Delete
Allows your environment's EC2 instances to perform operations required for your application.
Summary
Edit
Creation date
December 04, 2025, 00:56 (UTC+05:00)
ARN
arn:aws:iam::492461492697:role/aws-elasticbeanstalk-ec2-role
Instance profile ARN
arn:aws:iam::492461492697:instance-profile/aws-elasticbeanstalk-ec2-role
Last activity
10 minutes ago
Maximum session duration
1 hour
Permissions
Trust relationships
Tags
Last Accessed
Revoke sessions
Permissions policies (3) Info
You can attach up to 10 managed policies.
Simulate
Remove
Add permissions
Filter by Type
Search
All types
Policy name
Type
Attached entities
AWSElasticBeanstalkMulticontainerDocker
AWS managed
1
AWSElasticBeanstalkWebTier
AWS managed
1
AWSElasticBeanstalkWorkerTier
AWS managed
1
Permissions boundary (not set)



10. TESTING AND VERIFICATION

10.1 Backend API Testing

Health Check:

URL: http://16.16.65.200:5000/api/health

Expected Response:

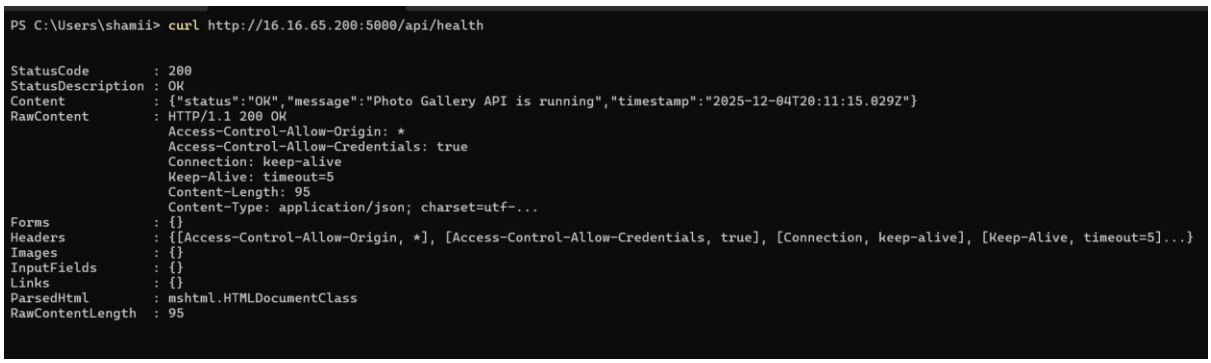
{

"status": "OK",

"message": "Photo Gallery API is running",

"timestamp": "2025-12-06T..."

}



10.2 Frontend Testing

Page Load Testing:

• Homepage loads: ✓ Success

• Login page loads: ✓ Success

• Gallery page loads: ✓ Success

• Upload page loads: ✓ Success

Functionality Testing:

• User can register: ✓ Success

• User can login: ✓ Success

• User can upload photo: ✓ Success

• Photos display in gallery: ✓ Success

• User can create album: ✓ Success

• User can add photo to album: ✓ Success

• Navigation works correctly: ✓ Success

10.3 Integration Testing

Frontend-Backend Integration:

• API calls successful: ✓ Success

• CORS configured correctly: ✓ Success

• Authentication flow works: ✓ Success

• File upload to S3 works: ✓ Success

Database Integration:

• Data persists in RDS: ✓ Success

• Queries execute correctly: ✓ Success

• Relationships maintained: ✓ Success

11. CONCLUSION

This project successfully demonstrates the end-to-end deployment of a full-stack

web application on AWS Cloud. The Photo Gallery application showcases:

✓ Independent frontend and backend architecture

✓ Use of multiple AWS managed services

✓ Proper security configuration

✓ Scalable and maintainable code structure

✓ Production-ready deployment practices

Learning Outcomes:

Through this project, we gained hands-on experience with:

1. AWS Services.
2. Cloud Architecture.
3. DevOps Practices.
4. Security Best Practices.

Challenges Faced

1. Elastic Beanstalk Deployment:

- Issue: ZIP file path format incompatibility
- Solution: Used Python script to create Linux-compatible ZIP

2. CORS Configuration:

- Issue: Frontend couldn't access backend API
- Solution: Configured CORS middleware with proper origins

3. S3 Upload Configuration:

- Issue: IAM permissions for EC2 to access S3
- Solution: Created and attached IAM role to EC2 instance

Cost Analysis

Current Monthly Cost (within Free Tier):

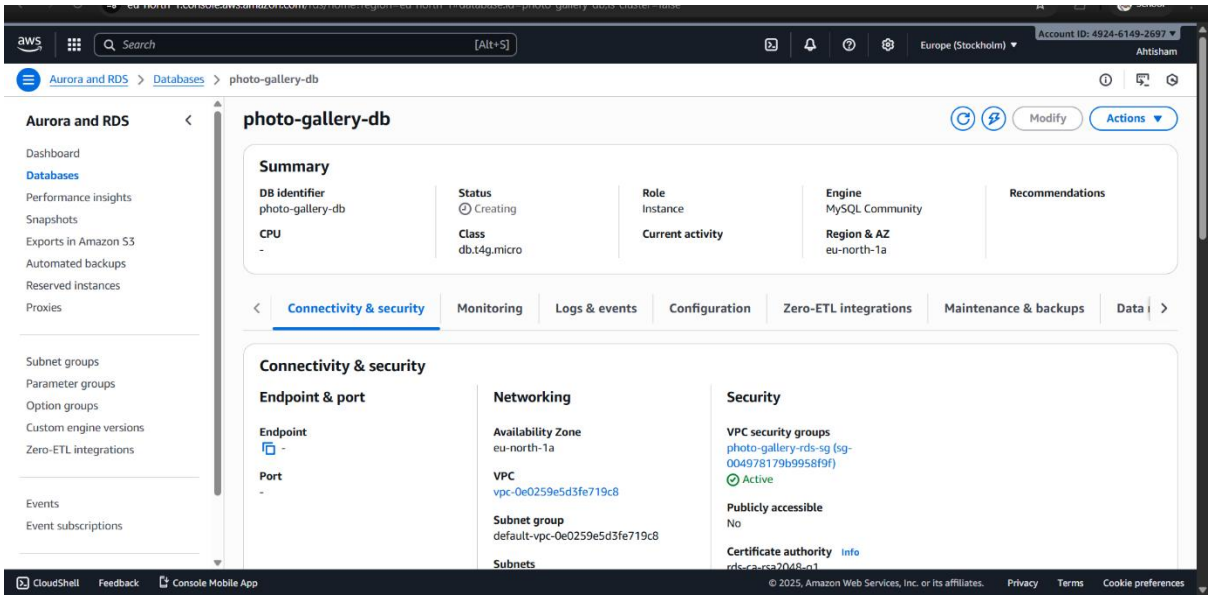
- EC2 t2.micro: $0 (750 hours free)
- RDS db.t3.micro: $0 (750 hours free)
- S3 Storage: $0 (5GB free)
- Elastic Beanstalk: $0 (only pay for resources)
- Data Transfer: $0 (100GB free)

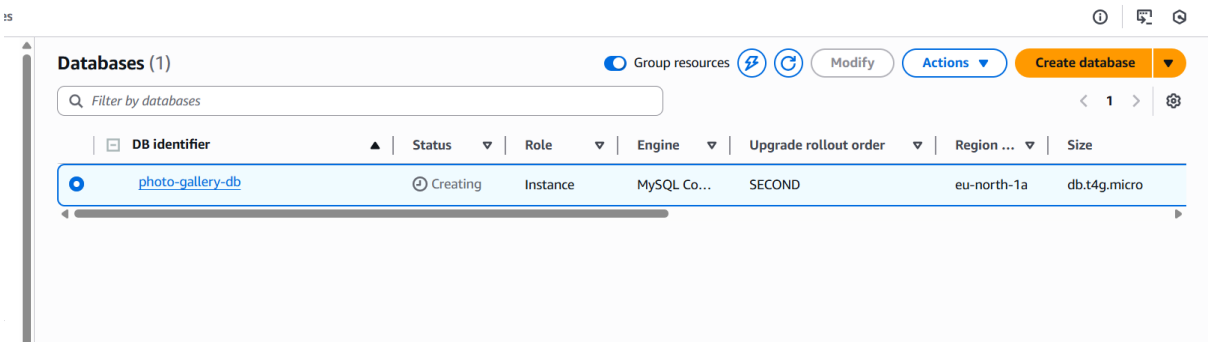
Total: $0/month (within free tier limits)

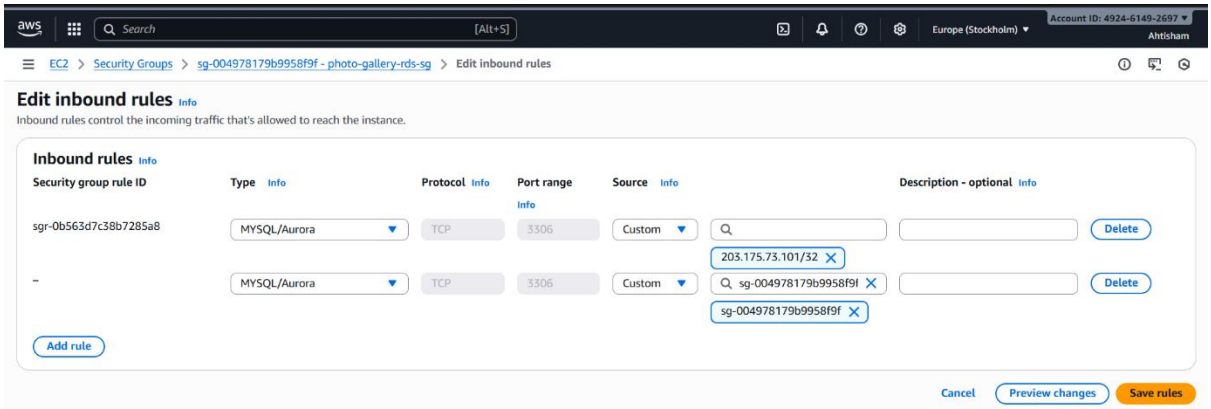
After Free Tier:

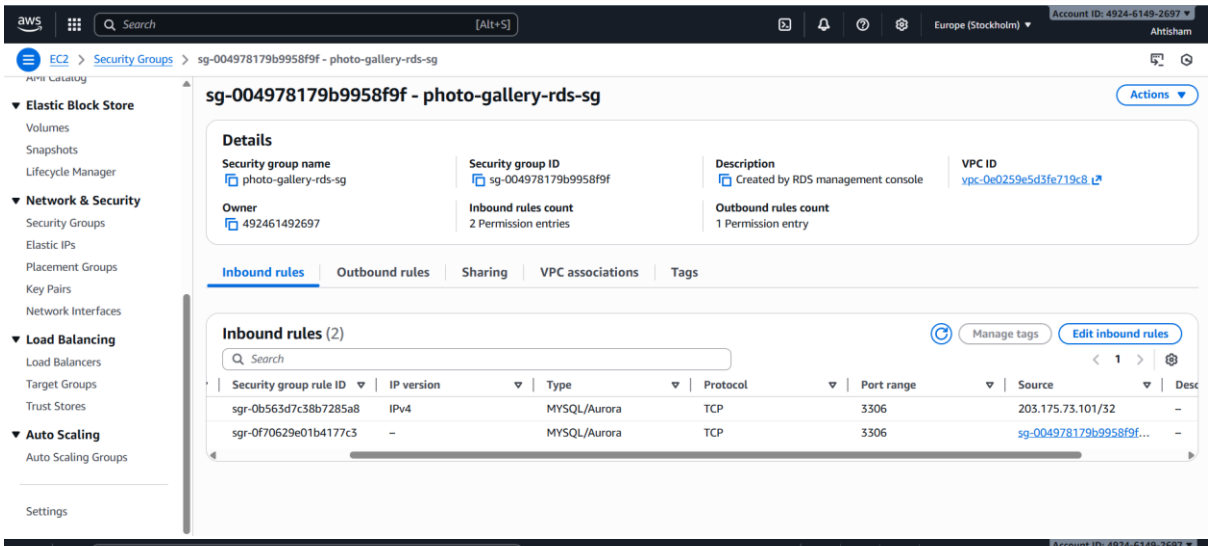
- Estimated cost: $15-30/month depending on usage

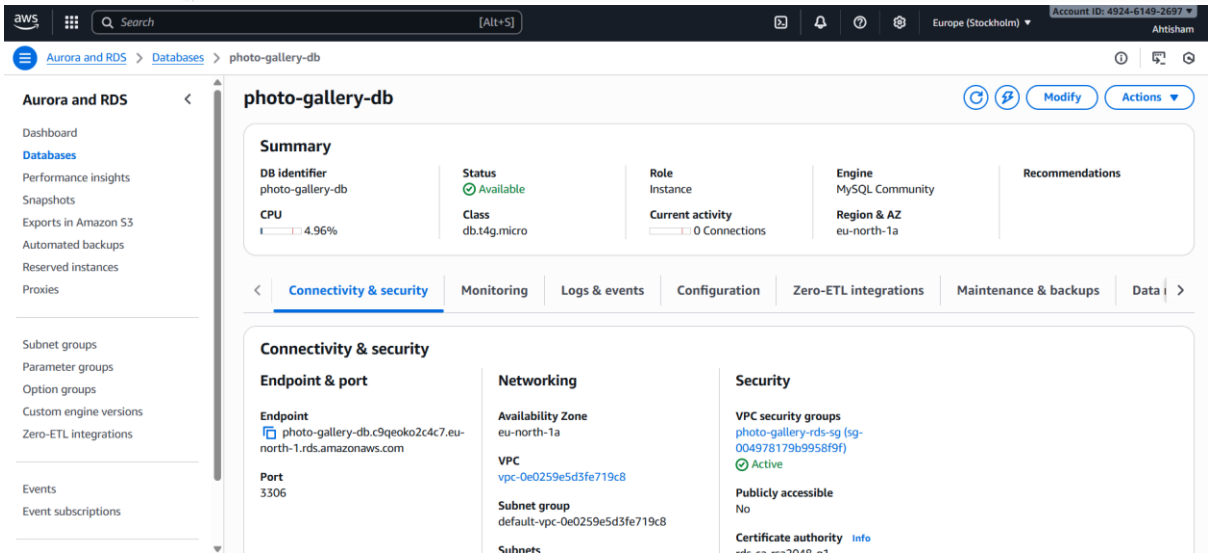
12. SCREENSHOTS:

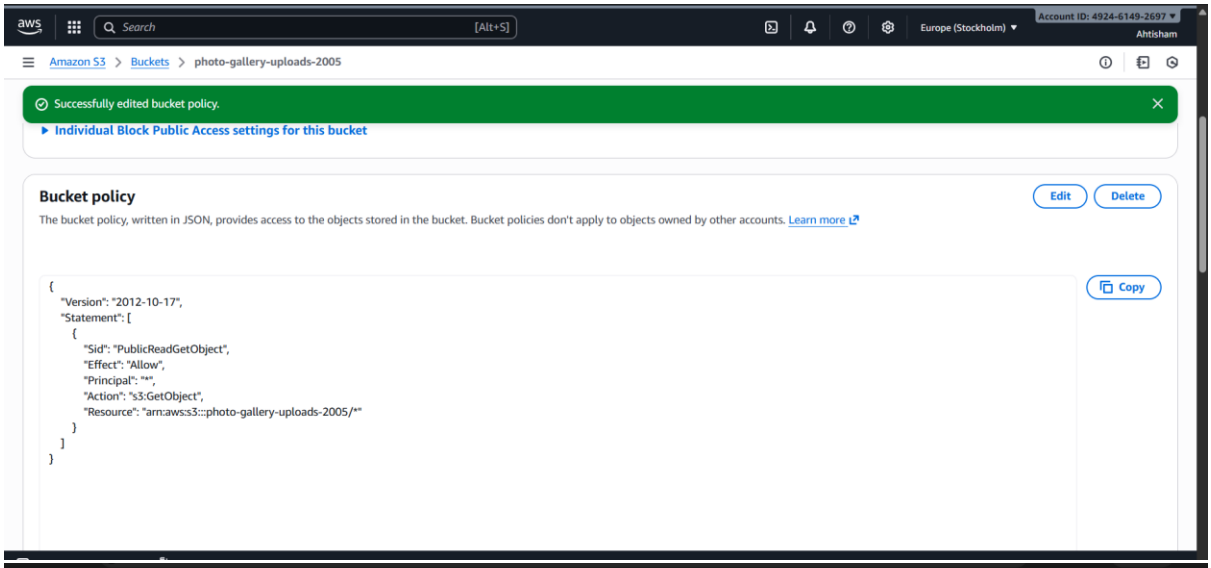


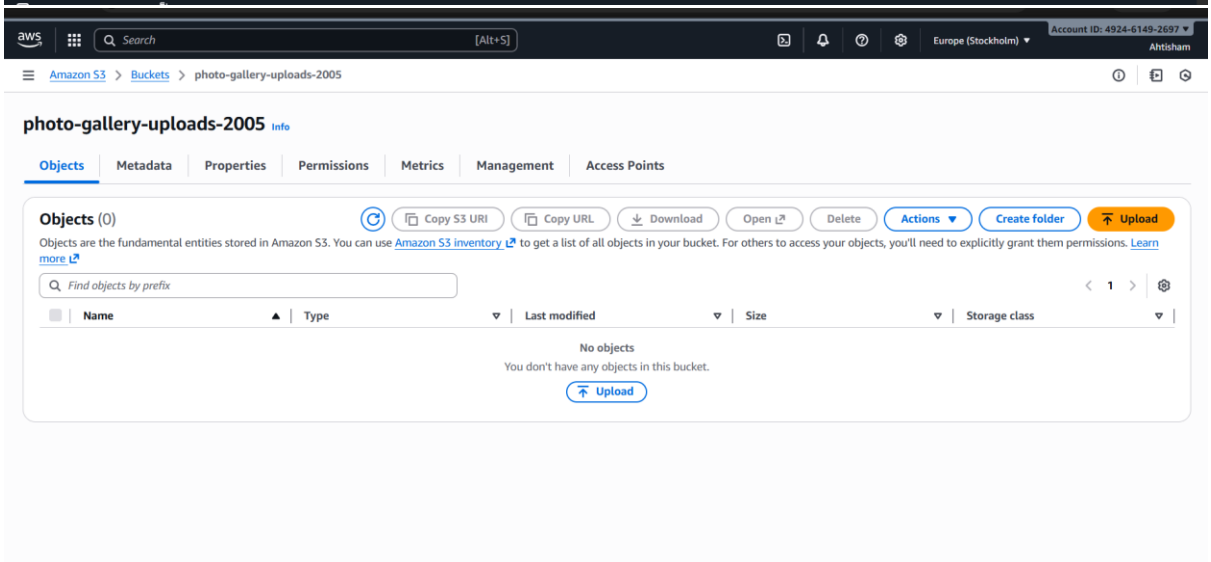


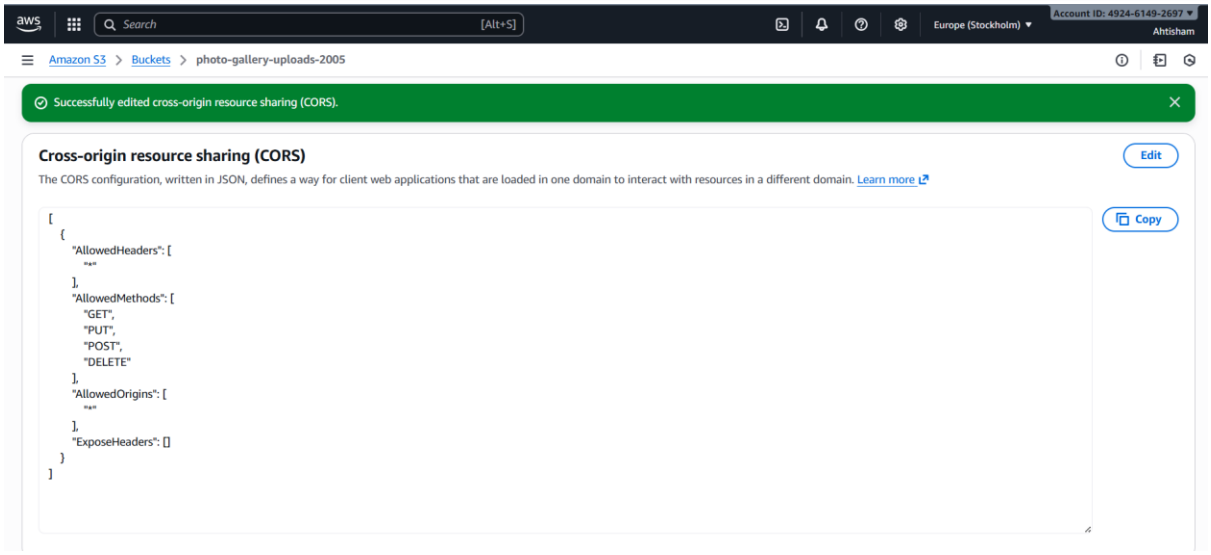

Search [Alt+S]
Europe (Stockholm)
Account ID: 4924-6149-2697
Ahtisham
EC2 > Security Groups > sg-004978179b9958f9f - photo-gallery-rds-sg > Edit inbound rules
Edit inbound rules Info
Inbound rules control the incoming traffic that's allowed to reach the instance.
Inbound rules Info
Security group rule ID
Type Info
Protocol Info
Port range Info
Source Info
Description - optional Info
sgr-0b563d7c38b7285a8
MYSQL/Aurora
TCP
3306
Custom
203.175.73.101/32
Delete
-
MYSQL/Aurora
TCP
3306
Custom
sg-004978179b9958f9f
sg-004978179b9958f9f
Delete
Add rule
Cancel
Preview changes
Save rules

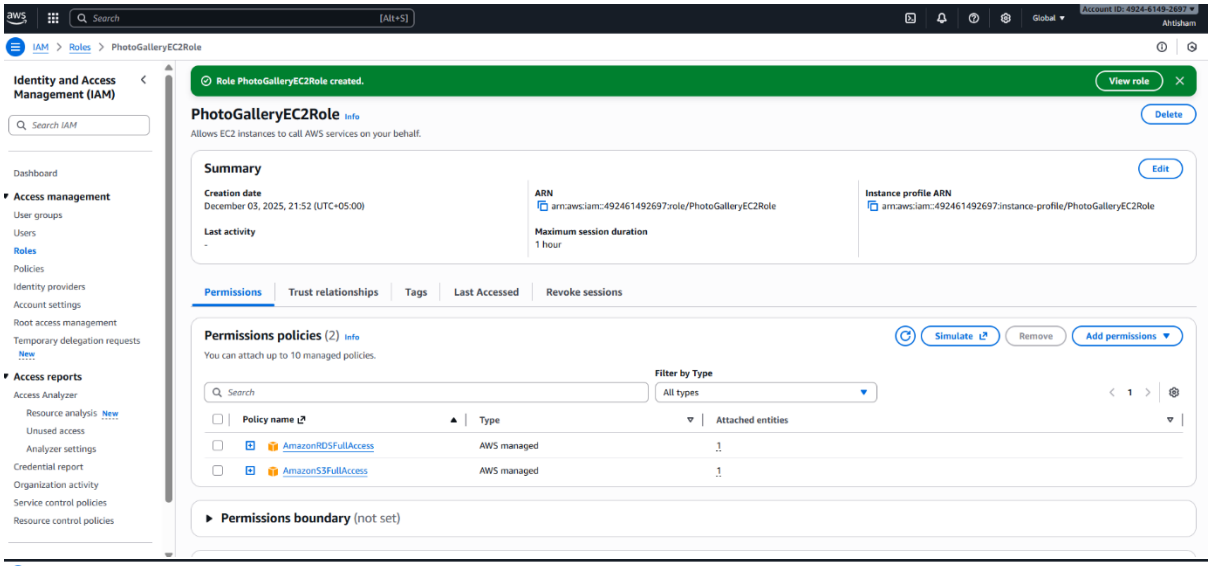

Search [Alt+S]
Europe (Stockholm)
Account ID: 4924-6149-2697
Ahtisham
EC2 > Security Groups > sg-004978179b9958f9f - photo-gallery-rds-sg
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
Load Balancing
Load Balancers
Target Groups
Trust Stores
Auto Scaling
Auto Scaling Groups
Settings
sg-004978179b9958f9f - photo-gallery-rds-sg
Actions
Details
Security group name
photo-gallery-rds-sg
Security group ID
sg-004978179b9958f9f
Description
Created by RDS management console
VPC ID
vpc-0e0259e5d3fe719c8
Owner
492461492697
Inbound rules count
2 Permission entries
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Sharing
VPC associations
Tags
Inbound rules (2)
Manage tags
Edit inbound rules
Search
Security group rule ID
IP version
Type
Protocol
Port range
Source
sgr-0b563d7c38b7285a8
IPv4
MYSQL/Aurora
TCP
3306
203.175.73.101/32
sgr-0f70629e01b4177c3
-
MYSQL/Aurora
TCP
3306
sg-004978179b9958f9f...

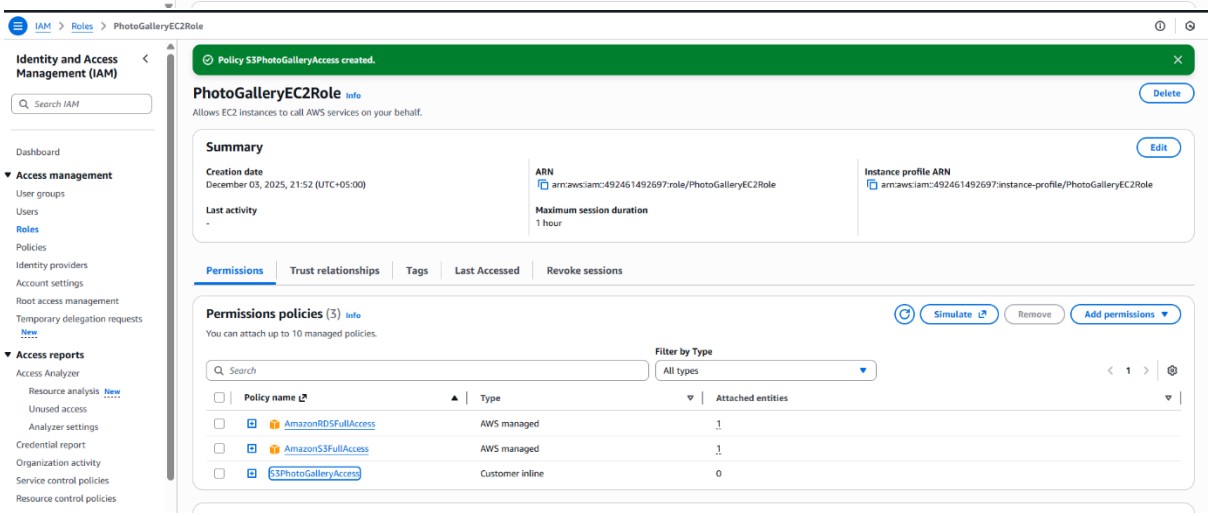

Search [Alt+S]
Europe (Stockholm)
Account ID: 4924-6149-2697
Ahtisham
Aurora and RDS > Databases > photo-gallery-db
Aurora and RDS
Dashboard
Databases
Performance insights
Snapshots
Exports in Amazon S3
Automated backups
Reserved instances
Proxies
Subnet groups
Parameter groups
Option groups
Custom engine versions
Zero-ETL integrations
Events
Event subscriptions
photo-gallery-db
Modify
Actions
Summary
DB identifier
photo-gallery-db
Status
Available
Role
Instance
Engine
MySQL Community
Recommendations
CPU
4.96%
Class
db.t4g.micro
Current activity
0 Connections
Region & AZ
eu-north-1a
Connectivity & security
Monitoring
Logs & events
Configuration
Zero-ETL integrations
Maintenance & backups
Connectivity & security
Endpoint & port
Endpoint
photo-gallery-db.c9qeoko2c4c7.eu-north-1.rds.amazonaws.com
Port
3306
Networking
Availability Zone
eu-north-1a
VPC
vpc-0e0259e5d3fe719c8
Subnet group
default-vpc-0e0259e5d3fe719c8
Security
VPC security groups
photo-gallery-rds-sg (sg-004978179b9958f9f)
Active
Publicly accessible
No
Certificate authority Info

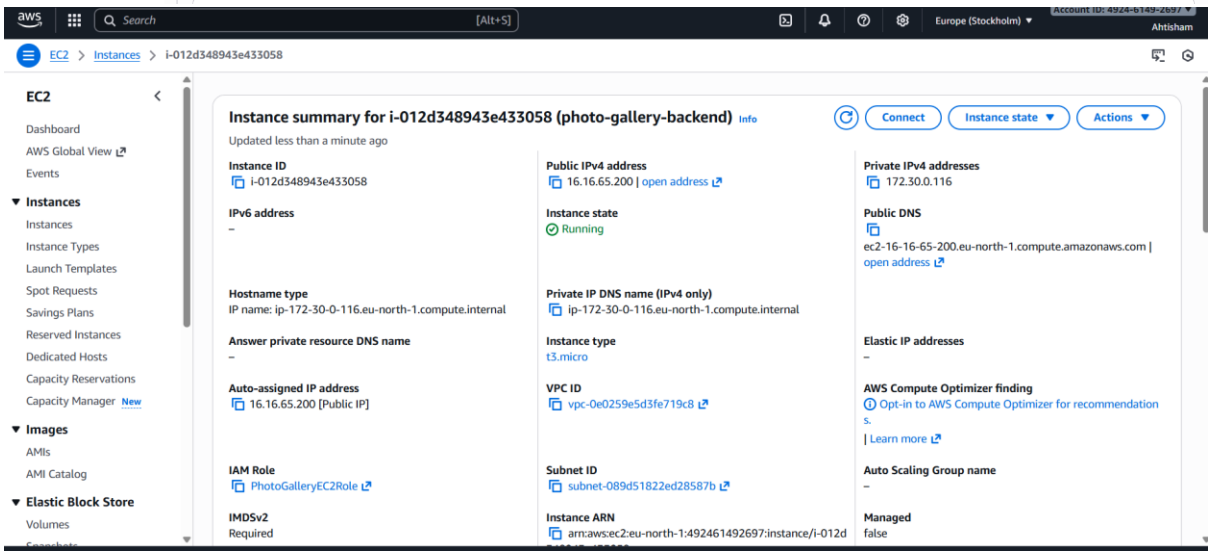

Search [Alt+S]
Europe (Stockholm)
Account ID: 4924-6149-2697
Ahtisham
Amazon S3 > Buckets > photo-gallery-uploads-2005
Successfully edited bucket policy.
Individual Block Public Access settings for this bucket
Bucket policy
Edit
Delete
The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more
Copy
{
"Version": "2012-10-17",
"Statement": [
{
"Sid": "PublicReadGetObject",
"Effect": "Allow",
"Principal": "*",
"Action": "s3:GetObject",
"Resource": "arn:aws:s3:::photo-gallery-uploads-2005/*"
}
]
}

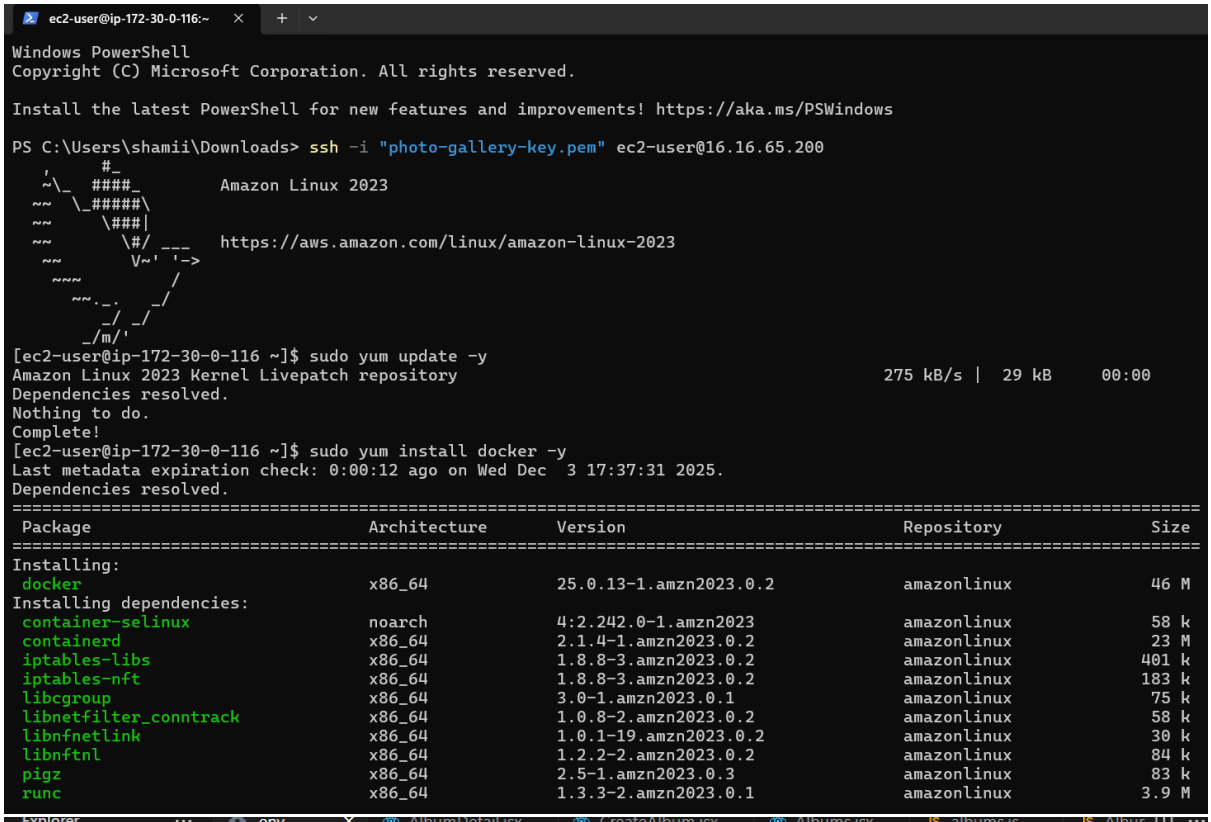

Search [Alt+S]
Europe (Stockholm)
Account ID: 4924-6149-2697
Ahtisham
Amazon S3 > Buckets > photo-gallery-uploads-2005
photo-gallery-uploads-2005 Info
Objects
Metadata
Properties
Permissions
Metrics
Management
Access Points
Objects (0)
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Find objects by prefix
Name
Type
Last modified
Size
Storage class
No objects
You don't have any objects in this bucket.
Upload

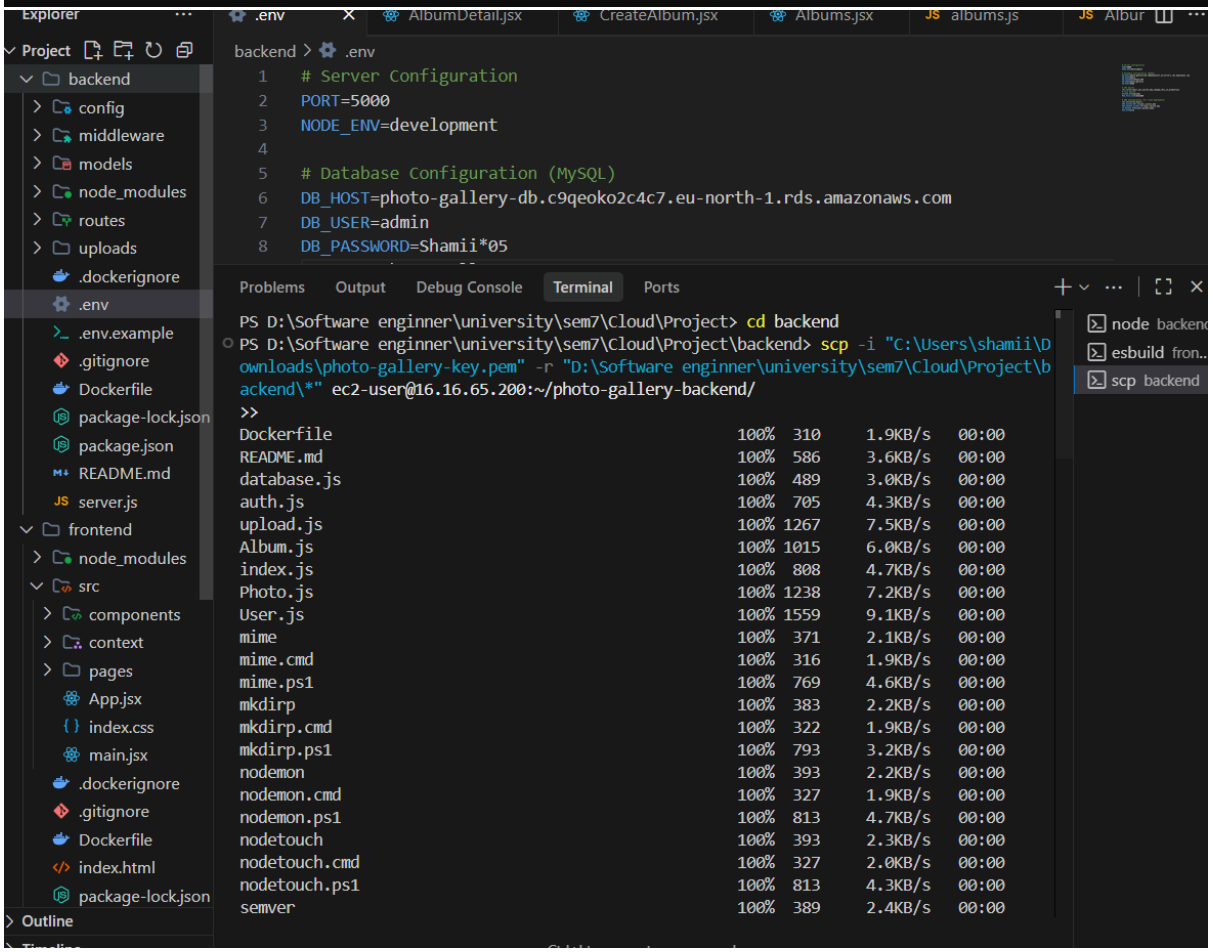

Search [Alt+S]
Europe (Stockholm)
Account ID: 4924-6149-2697
Ahtisham
Amazon S3 > Buckets > photo-gallery-uploads-2005
Successfully edited cross-origin resource sharing (CORS).
Cross-origin resource sharing (CORS)
Edit
The CORS configuration, written in JSON, defines a way for client web applications that are loaded in one domain to interact with resources in a different domain. Learn more
Copy
[
{
"AllowedHeaders": [
"*"
],
"AllowedMethods": [
"GET",
"PUT",
"POST",
"DELETE"
],
"AllowedOrigins": [
"*"
],
"ExposeHeaders": []
}
]

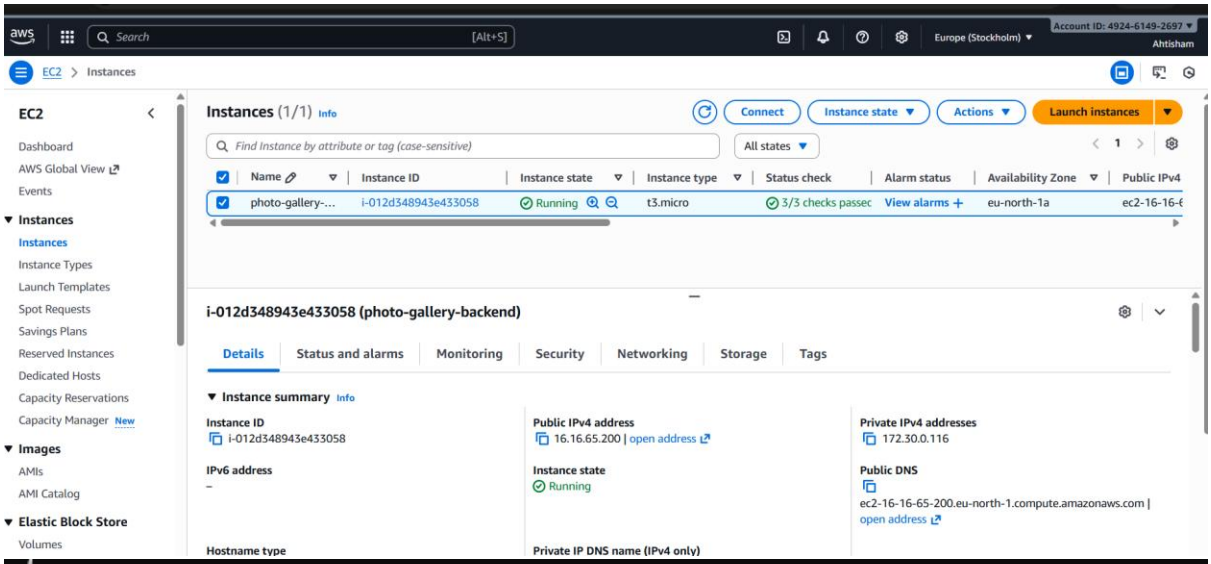

Search [Alt+S]
Global
Account ID: 4924-6149-2697
Ahtisham
IAM > Roles > PhotoGalleryEC2Role
Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Root access management
Temporary delegation requests
New
Access reports
Access Analyzer
Resource analysis New
Unused access
Analyzer settings
Credential report
Organization activity
Service control policies
Resource control policies
Role PhotoGalleryEC2Role created.
View role
PhotoGalleryEC2Role Info
Allows EC2 instances to call AWS services on your behalf.
Delete
Summary
Edit
Creation date
December 03, 2025, 21:52 (UTC+05:00)
ARN
arn:aws:iam::492461492697:role/PhotoGalleryEC2Role
Instance profile ARN
arn:aws:iam::492461492697:instance-profile/PhotoGalleryEC2Role
Last activity
-
Maximum session duration
1 hour
Permissions
Trust relationships
Tags
Last Accessed
Revoke sessions
Permissions policies (2) Info
You can attach up to 10 managed policies.
Simulate
Remove
Add permissions
Filter by Type
Search
All types
Policy name
Type
Attached entities
AmazonRDSFullAccess
AWS managed
1
AmazonS3FullAccess
AWS managed
1
Permissions boundary (not set)


IAM > Roles > PhotoGalleryEC2Role
Identity and Access Management (IAM)
Search IAM
Policy S3PhotoGalleryAccess created.
PhotoGalleryEC2Role Info
Allows EC2 instances to call AWS services on your behalf.
Delete
Summary
Edit
Creation date
December 03, 2025, 21:52 (UTC+05:00)
ARN
arn:aws:iam::492461492697:role/PhotoGalleryEC2Role
Instance profile ARN
arn:aws:iam::492461492697:instance-profile/PhotoGalleryEC2Role
Last activity
-
Maximum session duration
1 hour
Permissions
Trust relationships
Tags
Last Accessed
Revoke sessions
Permissions policies (3) Info
You can attach up to 10 managed policies.
Simulate
Remove
Add permissions
Filter by Type
Search
All types
Policy name
Type
Attached entities
AmazonRDSFullAccess
AWS managed
1
AmazonS3FullAccess
AWS managed
1
S3PhotoGalleryAccess
Customer inline
0


Search [Alt+S]
Europe (Stockholm)
Account ID: 4924-6149-2697
Ahtisham
EC2 > Instances > i-012d348943e433058
EC2
Dashboard
AWS Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Capacity Manager New
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Instance summary for i-012d348943e433058 (photo-gallery-backend) Info
Updated less than a minute ago
Connect
Instance state
Actions
Instance ID
i-012d348943e433058
Public IPv4 address
16.16.65.200 | open address
Private IPv4 addresses
172.30.0.116
IPv6 address
-
Instance state
Running
Public DNS
ec2-16-16-65-200.eu-north-1.compute.amazonaws.com | open address
Hostname type
IP name: ip-172-30-0-116.eu-north-1.compute.internal
Private IP DNS name (IPv4 only)
ip-172-30-0-116.eu-north-1.compute.internal
Answer private resource DNS name
-
Instance type
t3.micro
Elastic IP addresses
-
Auto-assigned IP address
16.16.65.200 [Public IP]
VPC ID
vpc-0e0259e5d3fe719c8
AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations.
| Learn more
IAM Role
PhotoGalleryEC2Role
Subnet ID
subnet-089d51822ed28587b
Auto Scaling Group name
-
IMDSv2
Required
Instance ARN
arn:aws:ec2:eu-north-1:492461492697:instance/i-012d
Managed
false

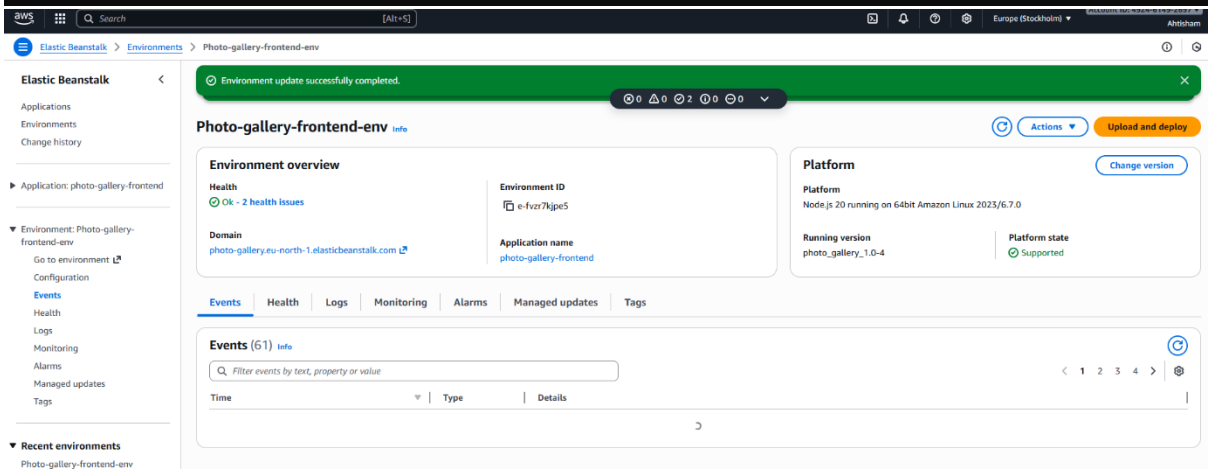

ec2-user@ip-172-30-0-116:~
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
PS C:\Users\shamii\Downloads> ssh -i "photo-gallery-key.pem" ec2-user@16.16.65.200
Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023
[ec2-user@ip-172-30-0-116 ~]$ sudo yum update -y
Amazon Linux 2023 Kernel Livepatch repository 275 kB/s | 29 kB 00:00
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-30-0-116 ~]$ sudo yum install docker -y
Last metadata expiration check: 0:00:12 ago on Wed Dec 3 17:37:31 2025.
Dependencies resolved.
Package Architecture Version Repository Size
Installing:
docker x86_64 25.0.13-1.amzn2023.0.2 amazonlinux 46 M
Installing dependencies:
container-selinux noarch 4:2.242.0-1.amzn2023 amazonlinux 58 k
containerd x86_64 2.1.4-1.amzn2023.0.2 amazonlinux 23 M
iptables-libs x86_64 1.8.8-3.amzn2023.0.2 amazonlinux 401 k
iptables-nft x86_64 1.8.8-3.amzn2023.0.2 amazonlinux 183 k
libcgroup x86_64 3.0-1.amzn2023.0.1 amazonlinux 75 k
libnetfilter_conntrack x86_64 1.0.8-2.amzn2023.0.2 amazonlinux 58 k
libnfnetlink x86_64 1.0.1-19.amzn2023.0.2 amazonlinux 30 k
libnftnl x86_64 1.2.2-2.amzn2023.0.2 amazonlinux 84 k
pigz x86_64 2.5-1.amzn2023.0.3 amazonlinux 83 k
runc x86_64 1.3.3-2.amzn2023.0.1 amazonlinux 3.9 M

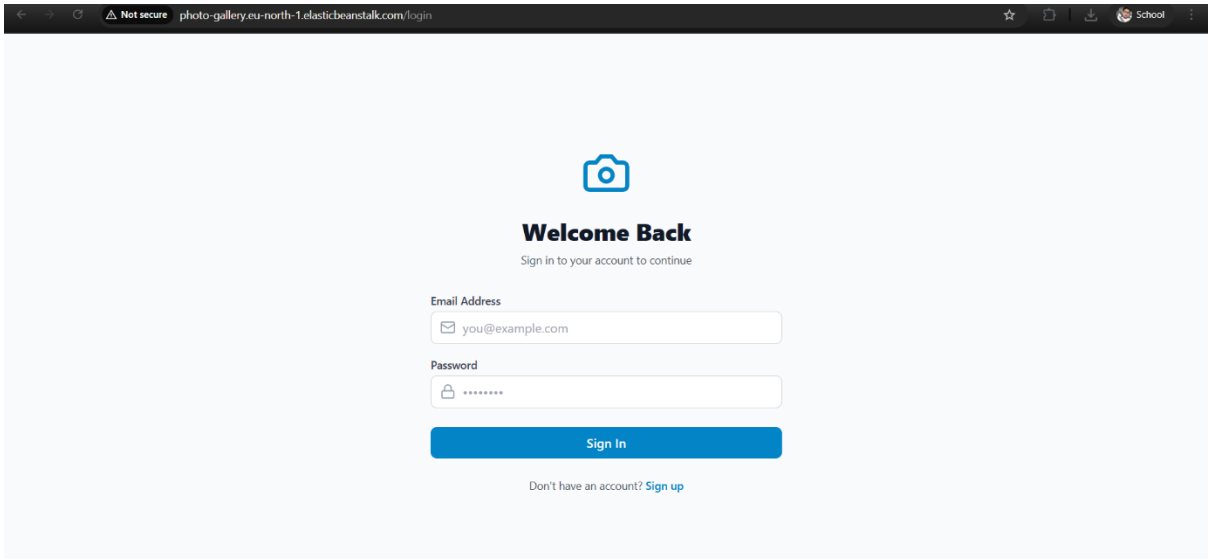

Explorer
.env
AlbumDetail.jsx
CreateAlbum.jsx
Albums.jsx
albums.js
Project
backend
config
middleware
models
node_modules
routes
uploads
.dockerignore
.env
.env.example
.gitignore
Dockerfile
package-lock.json
package.json
README.md
server.js
frontend
node_modules
src
components
context
pages
App.jsx
index.css
main.jsx
.dockerignore
.gitignore
Dockerfile
index.html
package-lock.json
Outline
backend > .env
1 # Server Configuration
2 PORT=5000
3 NODE_ENV=development
4
5 # Database Configuration (MySQL)
6 DB_HOST=photo-gallery-db.c9qeoko2c4c7.eu-north-1.rds.amazonaws.com
7 DB_USER=admin
8 DB_PASSWORD=Shamii*05
Problems Output Debug Console Terminal Ports
PS D:\Software enginner\university\sem7\Cloud\Project> cd backend
PS D:\Software enginner\university\sem7\Cloud\Project\backend> scp -i "C:\Users\shamii\Downloads\photo-gallery-key.pem" -r "D:\Software enginner\university\sem7\Cloud\Project\backend*" ec2-user@16.16.65.200:~/photo-gallery-backend/
>>
Dockerfile 100% 310 1.9KB/s 00:00
README.md 100% 586 3.6KB/s 00:00
database.js 100% 489 3.0KB/s 00:00
auth.js 100% 705 4.3KB/s 00:00
upload.js 100% 1267 7.5KB/s 00:00
Album.js 100% 1015 6.0KB/s 00:00
index.js 100% 808 4.7KB/s 00:00
Photo.js 100% 1238 7.2KB/s 00:00
User.js 100% 1559 9.1KB/s 00:00
mime 100% 371 2.1KB/s 00:00
mime.cmd 100% 316 1.9KB/s 00:00
mime.ps1 100% 769 4.6KB/s 00:00
mkdirp 100% 383 2.2KB/s 00:00
mkdirp.cmd 100% 322 1.9KB/s 00:00
mkdirp.ps1 100% 793 3.2KB/s 00:00
nodemon 100% 393 2.2KB/s 00:00
nodemon.cmd 100% 327 1.9KB/s 00:00
nodemon.ps1 100% 813 4.7KB/s 00:00
nodetouch 100% 393 2.3KB/s 00:00
nodetouch.cmd 100% 327 2.0KB/s 00:00
nodetouch.ps1 100% 813 4.3KB/s 00:00
semver 100% 389 2.4KB/s 00:00
node backend
esbuild fron...
scp backend

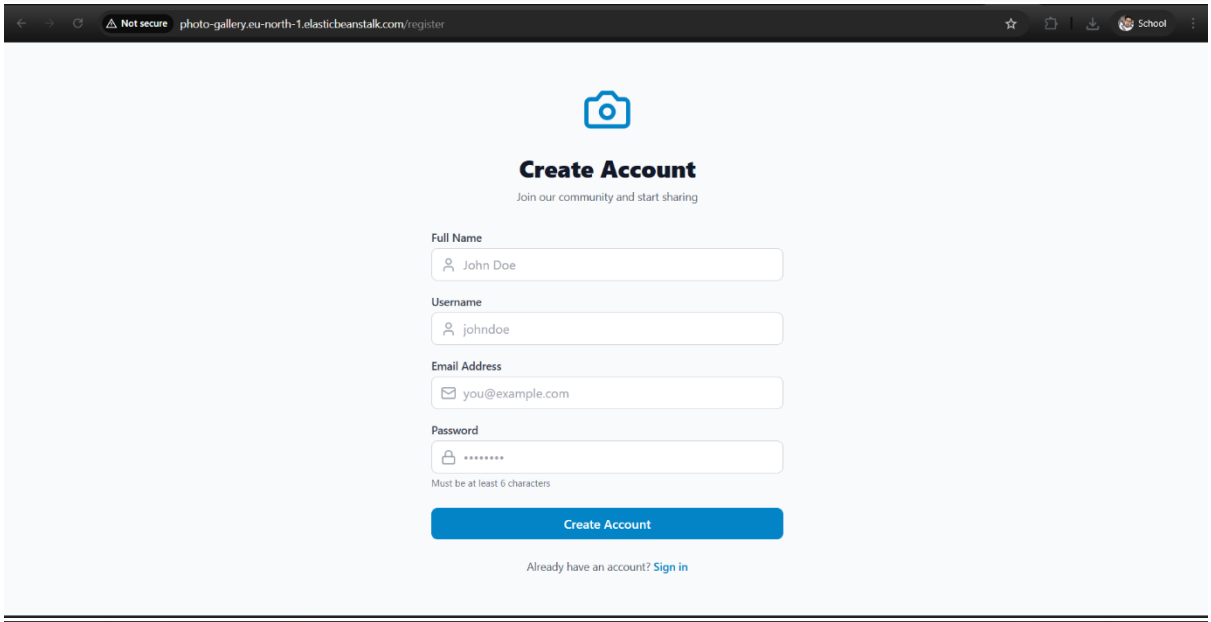


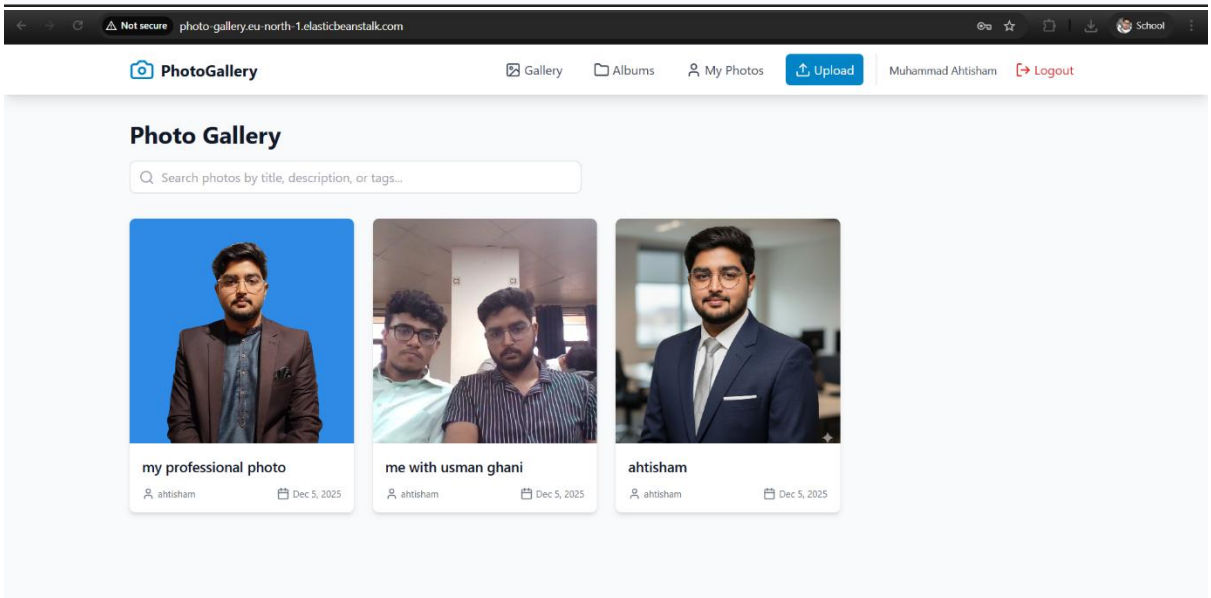

```
}
Database connection established successfully.
Database models synchronized.
Server is running on port 5000
Environment: production
```

```
PS C:\Users\shamii> curl http://16.16.65.200:5000/api/health


StatusCode        : 200
StatusDescription : OK
Content           : {"status":"OK","message":"Photo Gallery API is running","timestamp":"2025-12-03T19:04:52.966Z"}
RawContent        : HTTP/1.1 200 OK
                    Access-Control-Allow-Origin: *
                    Connection: keep-alive
                    Keep-Alive: timeout=5
                    Content-Length: 95
                    Content-Type: application/json; charset=utf-8
                    Date: Wed, 03 Dec 2025 19:04:52 GMT
                    ...
Forms             : {}
Headers           : {[Access-Control-Allow-Origin, *], [Connection, keep-alive], [Keep-Alive, timeout=5], [Content-Length, 95]...}
Images            : {}
InputFields       : {}
Links             : {}
ParsedHtml        : mshtml.HTMLDocumentClass
RawContentLength  : 95


PS C:\Users\shamii>
```




Not secure
photo-gallery.eu-north-1.elasticbeanstalk.com/login
School
Welcome Back
Sign in to your account to continue
Email Address
you@example.com
Password
Sign In
Don't have an account? Sign up


Not secure
photo-gallery.eu-north-1.elasticbeanstalk.com/register
School
Create Account
Join our community and start sharing
Full Name
John Doe
Username
johndoe
Email Address
you@example.com
Password
Must be at least 6 characters
Create Account
Already have an account? Sign in


Not secure
photo-gallery.eu-north-1.elasticbeanstalk.com
School
PhotoGallery
Gallery
Albums
My Photos
Upload
Muhammad Ahtisham
Logout
Photo Gallery
Search photos by title, description, or tags...
my professional photo
ahtisham
Dec 5, 2025
me with usman ghani
ahtisham
Dec 5, 2025
ahtisham
ahtisham
Dec 5, 2025

PhotoGallery Gallery Albums My Photos Upload Muhammad Ahtisham Logout

Albums

Organize your photos into collections

+ Create Album

nathia gali
ahdasodas
natiagali
Dec 4, 2025
0 photos usman

fyt
hunza
Jun 20, 2025
0 photos ahtisham

fast-nu
Islamabad
Dec 4, 2025
1 photo ahtisham

Not secure photo-gallery.eu-north-1.elasticbeanstalk.com/albums/1 School

PhotoGallery Gallery Albums My Photos Upload Muhammad Ahtisham Logout

fast-nu

Edit Delete

Islamabad
December 4, 2025
Created by Muhammad Ahtisham

Photos (1)

+ Add Photos

me with usman ghani
ahtisham Dec 5, 2025

Not secure photo-gallery.eu-north-1.elasticbeanstalk.com/photo/5 School

PhotoGallery Gallery Albums My Photos Upload Muhammad Ahtisham Logout

← Back

my professional photo

Public

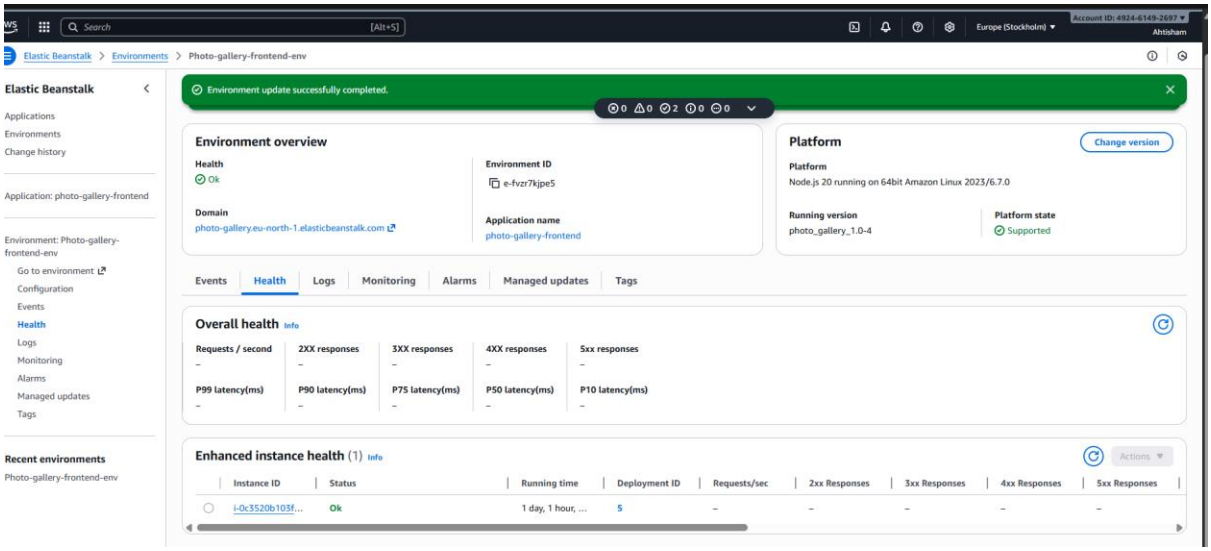
Muhammad Ahtisham
December 5, 2025 at 02:17 AM
Album: fast-nu

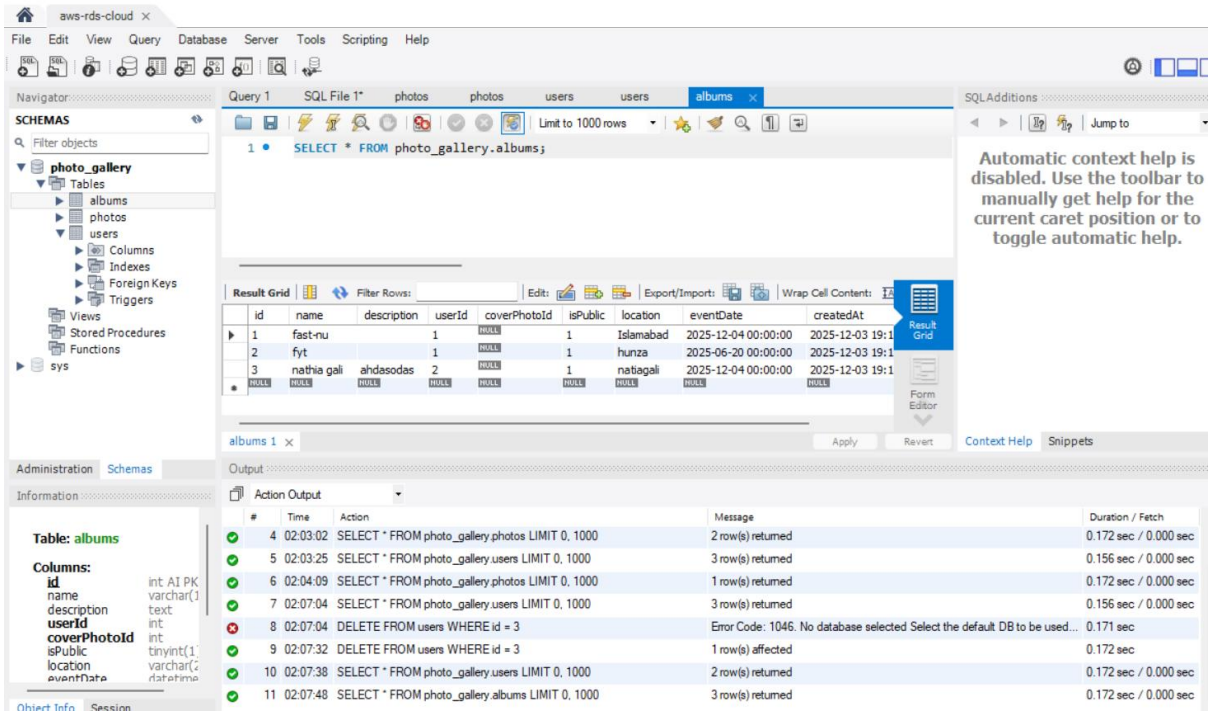
Add to Album

fast-nu

Select an album to organize this photo

Edit Delete


Elastic Beanstalk
Environments
Photo-gallery-frontend-env
Environment update successfully completed.
Environment overview
Health
Ok
Environment ID
e-fvzr7kjpe5
Domain
photo-gallery.eu-north-1.elasticbeanstalk.com
Application name
photo-gallery-frontend
Platform
Change version
Platform
Node.js 20 running on 64bit Amazon Linux 2023/6.7.0
Running version
photo_gallery_1.0-4
Platform state
Supported
Events
Health
Logs
Monitoring
Alarms
Managed updates
Tags
Overall health
Requests / second
2XX responses
3XX responses
4XX responses
5xx responses
P99 latency(ms)
P90 latency(ms)
P75 latency(ms)
P50 latency(ms)
P10 latency(ms)
Enhanced instance health (1)
Instance ID
Status
Running time
Deployment ID
Requests/sec
2xx Responses
3xx Responses
4xx Responses
5xx Responses
Ok
1 day, 1 hour, ...


aws-rds-cloud
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
photo_gallery
Tables
albums
photos
users
Columns
Indexes
Foreign Keys
Triggers
Views
Stored Procedures
Functions
sys
Query 1
SQL File 1*
photos
photos
users
users
albums
SELECT * FROM photo_gallery.albums;
Result Grid
id name description userId coverPhotoId isPublic location eventDate createdAt
1 fast-nu 1 1 Islamabad 2025-12-04 00:00:00 2025-12-03 19:1
2 fyt 1 1 hunza 2025-06-20 00:00:00 2025-12-03 19:1
3 nathia gali ahdasodas 2 1 natiagali 2025-12-04 00:00:00 2025-12-03 19:1
SQLAdditions
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Context Help
Snippets
Output
Action Output
Time Action Message Duration / Fetch
4 02:03:02 SELECT * FROM photo_gallery.photos LIMIT 0, 1000 2 row(s) returned 0.172 sec / 0.000 sec
5 02:03:25 SELECT * FROM photo_gallery.users LIMIT 0, 1000 3 row(s) returned 0.156 sec / 0.000 sec
6 02:04:09 SELECT * FROM photo_gallery.photos LIMIT 0, 1000 1 row(s) returned 0.172 sec / 0.000 sec
7 02:07:04 SELECT * FROM photo_gallery.users LIMIT 0, 1000 3 row(s) returned 0.156 sec / 0.000 sec
8 02:07:04 DELETE FROM users WHERE id = 3 Error Code: 1046. No database selected Select the default DB to be used... 0.171 sec
9 02:07:32 DELETE FROM users WHERE id = 3 1 row(s) affected 0.172 sec
10 02:07:38 SELECT * FROM photo_gallery.users LIMIT 0, 1000 2 row(s) returned 0.172 sec / 0.000 sec
11 02:07:48 SELECT * FROM photo_gallery.albums LIMIT 0, 1000 3 row(s) returned 0.172 sec / 0.000 sec
Information
Table: albums
Columns:
id int AI PK
name varchar(1
description text
userId int
coverPhotoId int
isPublic tinyint(1
location varchar(2
eventDate datetime