

Project Manual

Database Part

VPC

A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances. Mouse over a resource to highlight the related resources.

VPC settings

Resources to create [Info](#)
Create only the VPC resource or the VPC and other networking resources.
 VPC only VPC and more

Name tag auto-generation [Info](#)
Enter a value for the Name tag. This value will be used to auto-generate Name tags for all resources in the VPC.
 Auto-generate
gelgit

IPv4 CIDR block [Info](#)
Determine the starting IP and the size of your VPC using CIDR notation.
10.0.0.0/16 65,536 IPs
CIDR block size must be between /16 and /28.

IPv6 CIDR block [Info](#)
 No IPv6 CIDR block Amazon-provided IPv6 CIDR block

Tenancy [Info](#)
Default

Preview

VPC [Show details](#)
Your AWS virtual network
gelgit-vpc

Subnets (4)
Subnets within this VPC

- eu-central-1a**
 - gelgit-subnet-public1-eu-central-1a
 - gelgit-subnet-private1-eu-central-1a
- eu-central-1b**
 - gelgit-subnet-public2-eu-central-1b
 - gelgit-subnet-private2-eu-central-1b

Route table
Route networks

- gelgit-rtb-public
- gelgit-rtb-private
- gelgit-rtb-public2

aws Services Search [Option+S] Frankfurt alponzo @ 6574-2322-6272

Number of Availability Zones (AZs) [Info](#)
Choose the number of AZs in which to provision subnets. We recommend at least two AZs for high availability.
1 **2** 3

Customize AZs

First availability zone eu-central-1a

Second availability zone eu-central-1b

Number of public subnets [Info](#)
The number of public subnets to add to your VPC. Use public subnets for web applications that need to be publicly accessible over the internet.
0 **2** 4

Number of private subnets [Info](#)
The number of private subnets to add to your VPC. Use private subnets to secure backend resources that don't need public access.
0 **2** 4

Customize subnets CIDR blocks

Public subnet CIDR block in eu-central-1a 10.0.0.0/24 256 IPs

Public subnet CIDR block in eu-central-1b 10.0.16.0/24 256 IPs

Private subnet CIDR block in eu-central-1a 10.0.128.0/24 256 IPs

Private subnet CIDR block in eu-central-1b 10.0.144.0/24 256 IPs

NAT gateways (\$) [Info](#)
Choose the number of Availability Zones (AZs) in which to create NAT gateways.
Note that there is a charge for each NAT gateway.

VPC endpoints [Info](#)
Endpoints can help reduce NAT gateway charges and improve security by customizing S3 directly from the VPC. By default, full access policy is used. You can customize this policy at any time.

DNS options [Info](#)
 Enable DNS hostnames
 Enable DNS resolution

[▶ Additional tags](#)

EC2: Bastion Host / Jump Box - To connect DB

[EC2](#) > [Instances](#) > Launch an instance

Launch an instance [Info](#)

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags [Info](#)

Name
 Add additional tags

Summary

Number of instances [Info](#)

Software Image (AMI)
Amazon Linux 2023 AMI 2023.4.2...[read more](#)
ami-0f7204385566b32d0

Virtual server type (instance type)
t2.micro

[Firewall \(security group\)](#)

Currently used: gelgit-jumpbox-rds-ec2

Application and OS Images (Amazon Machine Image) [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below.

Quick Start

Amazon Linux	macOS	Ubuntu	Windows	Red Hat	SUSE Linux Enterprise Server

Browse more AMIs
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2023 AMI ami-0f7204385566b32d0 (64-bit (x86), uefi-preferred) / ami-0ea5b25e16e95714 (64-bit (Arm), uefi) Virtualization: hvm ENA enabled: true Root device type: ebs	Free tier eligible
--	--------------------

Summary

Number of instances [Info](#)

Software Image (AMI)
Amazon Linux 2023 AMI 2023.4.2...[read more](#)
ami-0f7204385566b32d0

Virtual server type (instance type)
t2.micro

[Firewall \(security group\)](#)
New security group

[Storage \(volumes\)](#)
1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is available)

Description
Amazon Linux 2023 AMI 2023.4.20240401.1 x86_64 HVM kernel-6.1

Architecture 64-bit (x86)	Boot mode uefi-preferred	AMI ID ami-0f7204385566b32d0	Verified provider
------------------------------	-----------------------------	---------------------------------	-------------------

Instance type [Info](#) | [Get advice](#)

Instance type
t2.micro Free tier eligible

Family: t2 1 vCPU 1 GiB Memory Current generation: true
 On-Demand Windows base pricing: 0.018 USD per Hour
 On-Demand SUSE base pricing: 0.0134 USD per Hour
 On-Demand Linux base pricing: 0.0134 USD per Hour
 On-Demand RHEL base pricing: 0.0734 USD per Hour

All generations Compare instance types

Additional costs apply for AMIs with pre-installed software

Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - required
jumb-box-rds

[Create new key pair](#)

Summary

Number of instances [Info](#)
1

Software Image (AMI)
Amazon Linux 2023 AMI 2023.4.2...[read more](#)
ami-0f7204385566b32d0

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

[Free tier: In your first year includes](#)

Currently used key pair: jumb-box-rds-keypair

Network settings [Info](#)

VPC - required [Info](#)
vpc-08830f054d1773126 (gelgit-vpc) [Edit](#)

Subnet [Info](#)
subnet-0221dc0cf49a02b03 gelgit-subnet-public1-eu-central-1a
VPC: vpc-08830f054d1773126 Owner: 637423226272
Availability Zone: eu-central-1a IP addresses available: 251 CIDR: 10.0.0.0/24

Create new subnet [Edit](#)

Auto-assign public IP [Info](#)

Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) [Info](#)
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group Select existing security group

Security group name - required
ec2-jump-box-rds-sg

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and _-:/@#=;&{}!\$*

Description - required [Info](#)
ec2-jump-box-rds-sg

Inbound Security Group Rules

Security group rule 1 (TCP, 22, 0.0.0.0/0)

Type Info	Protocol Info	Port range Info
ssh	TCP	22
Source type Info	Source Info	Description - optional Info
Anywhere	Add CIDR, prefix list or security group	e.g. SSH for admin desktop
0.0.0.0/0 Edit		

Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Add security group rule

Summary

Number of instances [Info](#)
1

Software Image (AMI)
Amazon Linux 2023 AMI 2023.4.2...[read more](#)
ami-0f7204385566b32d0

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

[Free tier: In your first year includes](#)

750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GiB of bandwidth to the internet

[Launch instance](#)

Current security group name = jumpbox-rds / jumpbox-rds-sg

The screenshot shows the AWS EC2 Dashboard with the 'Instances' section selected. A single instance, 'gelgit-jumpbox-rds-ec2' (ID: i-04d8bd27ceb27b795), is listed as 'Running'. The instance details page is open, showing information such as Instance ID, Public IPv4 address (54.172.200.114), Instance type (t2.micro), and VPC ID (vpc-052552b87a2cf7). The 'Details' tab is active.

The screenshot shows the AWS EC2 Connect interface for the instance 'gelgit-jumpbox-rds-ec2'. The 'SSH' tab is selected, displaying an SSH session window. The terminal output shows the user logging in via SSH and navigating to the root directory (~). The session is connected to the instance's public IP (54.172.200.114).

AWS RDS MySQL - N. Virginia

- <https://beabetterdev.com/2022/12/13/how-to-connect-to-an-rds-or-aurora-database-in-a-private-subnet/>

Step 7 – Create RDS Subnet Group



RDS > Subnet groups > Create DB subnet group

Create DB subnet group

To create a new subnet group, give it a name and a description, and choose an existing VPC. You will then be able to add subnets related to that VPC.

Subnet group details

Name

You won't be able to modify the name after your subnet group has been created.

Must contain from 1 to 255 characters. Alphanumeric characters, spaces, hyphens, underscores, and periods are allowed.

Description

VPC

Choose a VPC identifier that corresponds to the subnets you want to use for your DB subnet group. You won't be able to choose a different VPC identifier after your subnet group has been created.

Add subnets

select the two private subnets from the two availability zones

VPC

Choose a VPC identifier that corresponds to the subnets you want to use for your DB subnet group. You won't be able to choose a different VPC identifier after your subnet group has been created.

Add subnets

Availability Zones

Choose the Availability Zones that include the subnets you want to add.

Subnets

Choose the subnets that you want to add. The list includes the subnets in the selected Availability Zones.

ⓘ For Multi-AZ DB clusters, you must select 3 subnets in 3 different Availability Zones. Important note

Subnets selected (2)

Availability zone	Subnet ID	CIDR block
us-east-1a	subnet-0d78bcb11596e9768	10.0.128.0/24
us-east-1b	subnet-08f58b60ff1fccfda	10.0.144.0/24

Step 8 – Create your Aurora or RDS

To create your database, click on Databases on the left hand navigation bar and click the Create Database button.

We're going to use Standard Create.

For Engine Type, select Amazon Aurora and tick the radio box that says Amazon Aurora PostgreSQL-Compatible Edition. Note that you can use another database type here if you'd like, or even the Serverless variation of Aurora as well – all will work.

For templates, select Dev/Test.

For Master Password, input a secure password.

Change your Instance Type to something small like a db.t3.medium or better yet, use serverless.

Under the Connectivity section, make the following changes.

Select your VPC.

Change the DB Subnet Group to be private-subnet-group which is the one we created in the previous step.

For Public Access, click No.

For Security Group, click Create New and name it rds-aurora-security-group. Finally, change the availability zone to 1a (the same AZ as our jump box from earlier steps).

Below is an image of my Connectivity settings.

The screenshot shows the 'Create database' wizard in the AWS RDS console. The top navigation bar includes 'Services' and a search bar for 'vpc'. The main page title is 'Create database'. On the left, under 'Choose a database creation method', 'Standard create' is selected. In the 'Engine options' section, 'MySQL' is chosen. To the right, there's a sidebar titled 'Allocated storage' with instructions about default storage sizes and how to increase them. At the bottom, the 'Edition' is set to 'MySQL Community'.

Create database

Choose a database creation method Info

Standard create
You set all of the configuration options, including ones for availability, security, backups, and maintenance.

Easy create
Use recommended best-practice configurations. Some configuration options can be changed after the database is created.

Engine options

Engine type Info

Aurora (MySQL Compatible)

Aurora (PostgreSQL Compatible)

MySQL

MariaDB

PostgreSQL

Oracle

Microsoft SQL Server

IBM Db2

Edition

MySQL Community

Allocated storage

By default, allocated storage size for creating a read replica, restoring to a point in time, or restoring from a DB snapshot is the same as the storage size for the source DB instance or DB snapshot source.

However, you can create a read replica, restore to a point in time, or restore from a DB instance snapshot to an allocated storage size greater than the source DB instance or source DB snapshot's storage size.

To increase the allocated storage size, enter a larger value in the Allocated storage field.

Learn more

[Working with read replica](#)
[Restoring a DB instance to a specified time](#)
[Restoring from a DB snapshot](#)

Edition
 MySQL Community



Known issues/limitations

Review the [Known issues/limitations](#) to learn about potential compatibility issues with specific database versions.

Engine version [Info](#)

View the engine versions that support the following database features.

[▼ Hide filters](#)

Show versions that support the Multi-AZ DB cluster [Info](#)

Create a Multi-AZ DB cluster with one primary DB instance and two readable standby DB instances. Multi-AZ DB clusters provide up to 2x faster transaction commit latency and automatic failover in typically under 35 seconds.

Show versions that support the Amazon RDS Optimized Writes [Info](#)

Amazon RDS Optimized Writes improves write throughput by up to 2x at no additional cost.

Engine Version

MySQL 8.0.34



Templates

Choose a sample template to meet your use case.

Production

Use defaults for high availability and fast, consistent performance.

Dev/Test

This instance is intended for development use outside of a production environment.

Free tier

Use RDS Free Tier to develop new applications, test existing applications, or gain hands-on experience with Amazon RDS. [Info](#)

Availability and durability

Deployment options [Info](#)

The deployment options below are limited to those supported by the engine you selected above.

Multi-AZ DB Cluster

Creates a DB cluster with a primary DB instance and two readable standby DB instances, with each DB instance in a different Availability Zone (AZ). Provides high availability, data redundancy and increases capacity to serve read workloads.

Multi-AZ DB instance

Creates a primary DB instance and a standby DB instance in a different AZ. Provides high availability and data redundancy, but the standby DB instance doesn't support connections for read workloads.

Single DB instance

Creates a single DB instance with no standby DB instances.

Settings

DB instance identifier [Info](#)

Type a name for your DB instance. The name must be unique across all DB instances owned by your AWS account in the current AWS Region.

gelgit-database-1

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 60 alphanumeric characters or hyphens. First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

▼ Credentials Settings

Master username [Info](#)

Type a login ID for the master user of your DB instance.

admin

1 to 16 alphanumeric characters. The first character must be a letter.

Credentials management

You can use AWS Secrets Manager or manage your master user credentials.

Managed in AWS Secrets Manager - most

secure

RDS generates a password for you and manages it throughout its lifecycle using AWS Secrets Manager.

Self managed

Create your own password or have RDS create a password that you manage.

Auto generate password

Amazon RDS can generate a password for you, or you can specify your own password.

Master password [Info](#)

Minimum constraints: At least 8 printable ASCII characters. Can't contain any of the following symbols: / ' " @

Confirm master password [Info](#)

Master Username: admin

Master Password: Master2024

Instance configuration

The DB instance configuration options below are limited to those supported by the engine that you selected above.

DB instance class [Info](#)

[▼ Hide filters](#)

Show instance classes that support Amazon RDS Optimized Writes [Info](#)

Amazon RDS Optimized Writes improves write throughput by up to 2x at no additional cost.

Include previous generation classes

Standard classes (includes m classes)

Memory optimized classes (includes r and x classes)

Burstable classes (includes t classes)

db.t3.micro

2 vCPUs 1 GiB RAM Network: 2,085 Mbps

Storage

Storage type [Info](#)

Provisioned IOPS SSD (io2) storage volumes are now available.

General Purpose SSD (gp3)

Performance scales independently from storage



Allocated storage [Info](#)

20

GiB

Minimum: 20 GiB. Maximum: 6,144 GiB

- ⓘ After you modify the storage for a DB instance, the status of the DB instance will be in storage-optimization. Your instance will remain available as the storage-optimization operation completes.

[Learn more](#)

► Advanced settings

Baseline IOPS of 3,000 IOPS and storage throughput of 125 MiBps are included for allocated storage less than 400 GiB.

▼ Storage autoscaling

Storage autoscaling [Info](#)

Provides dynamic scaling support for your database's storage based on your application's needs.

Enable storage autoscaling

Enabling this feature will allow the storage to increase after the specified threshold is exceeded.

Maximum storage threshold [Info](#)

Charges will apply when your database autoscales to the specified threshold

100

GiB

The minimum value is 22 GiB and the maximum value is 6,144 GiB

Connectivity [Info](#)



Compute resource

Choose whether to set up a connection to a compute resource for this database. Setting up a connection will automatically change connectivity settings so that the compute resource can connect to this database.

Don't connect to an EC2 compute resource

Don't set up a connection to a compute resource for this database. You can manually set up a connection to a compute resource later.

Connect to an EC2 compute resource

Set up a connection to an EC2 compute resource for this database.

Network type [Info](#)

To use dual-stack mode, make sure that you associate an IPv6 CIDR block with a subnet in the VPC you specify.

IPv4

Your resources can communicate only over the IPv4 addressing protocol.

Dual-stack mode

Your resources can communicate over IPv4, IPv6, or both.

Virtual private cloud (VPC) [Info](#)

Choose the VPC. The VPC defines the virtual networking environment for this DB instance.

gelgit-vpc (vpc-0525552ba87a21cf7)

4 Subnets, 2 Availability Zones



Only VPCs with a corresponding DB subnet group are listed.

After a database is created, you can't change its VPC.

DB subnet group [Info](#)

Choose the DB subnet group. The DB subnet group defines which subnets and IP ranges the DB instance can use in the VPC that you selected.

db-private-subnet

2 Subnets, 2 Availability Zones



Public access [Info](#)

Yes

RDS assigns a public IP address to the database. Amazon EC2 instances and other resources outside of the VPC can connect to your database. Resources inside the VPC can also connect to the database. Choose one or more VPC security groups that specify which resources can connect to the database.

No

RDS doesn't assign a public IP address to the database. Only Amazon EC2 instances and other resources inside the VPC can connect to your database. Choose one or more VPC security groups that specify which resources can connect to the database.

VPC security group (firewall) [Info](#)

Choose one or more VPC security groups to allow access to your database. Make sure that the security group rules allow the appropriate incoming traffic.

 Choose existing

Choose existing VPC security groups

 Create new

Create new VPC security group

New VPC security group name

rds-mysql-sg

RDS Proxy

RDS Proxy is a fully managed, highly available database proxy that improves application scalability, resiliency, and security.

 [Create an RDS Proxy](#) [Info](#)

RDS automatically creates an IAM role and a Secrets Manager secret for the proxy. RDS Proxy has additional costs. For more information, see [Amazon RDS Proxy pricing](#).

Certificate authority - optional [Info](#)

Using a server certificate provides an extra layer of security by validating that the connection is being made to an Amazon database. It does so by checking the server certificate that is automatically installed on all databases that you provision.

rds-ca-rsa2048-g1 (default)

Expiry: May 26, 2061



If you don't select a certificate authority, RDS chooses one for you.

▼ Additional configuration**Database port** [Info](#)

TCP/IP port that the database will use for application connections.

3306

Tags

A tag consists of a case-sensitive key-value pair.

No tags associated with the resource.

[Add new tag](#)

You can add up to 50 more tags.

Database authentication

Database authentication options [Info](#)

Password authentication

Authenticates using database passwords.

Password and IAM database authentication

Authenticates using the database password and user credentials through AWS IAM users and roles.

Password and Kerberos authentication

Choose a directory in which you want to allow authorized users to authenticate with this DB instance using Kerberos Authentication.

Monitoring

Enable Enhanced Monitoring

Enabling Enhanced Monitoring metrics are useful when you want to see how different processes or threads use the CPU.

Granularity

10 seconds



Monitoring Role

default



Clicking "Create database" will authorize RDS to create the IAM role rds-monitoring-role

▼ Additional configuration

Database options, encryption turned on, backup turned on, backtrack turned off, maintenance, CloudWatch Logs, delete protection turned off.

Database options

Initial database name [Info](#)

gelgit_database

If you do not specify a database name, Amazon RDS does not create a database.

DB parameter group [Info](#)

default.mysql8.0



Option group [Info](#)

default:mysql-8-0



Backup

Backup

Enable automated backups

Creates a point-in-time snapshot of your database

⚠ Please note that automated backups are currently supported for InnoDB storage engine only. If you are using MyISAM, refer to details [here](#).

Backup retention period [Info](#)

The number of days (1-35) for which automatic backups are kept.

10



days

Backup window [Info](#)

The daily time range (in UTC) during which RDS takes automated backups.

Choose a window

No preference

Start time

Duration

00



: 00



UTC

0.5



hours

Copy tags to snapshots

Backup replication [Info](#)

Enable replication in another AWS Region

Enabling replication automatically creates backups of your DB instance in the selected Region, for disaster recovery, in addition to the current Region.

Encryption

Encryption

Enable encryption

Choose to encrypt the given instance. Master key IDs and aliases appear in the list after they have been created using the AWS Key Management Service console. [Info](#)

AWS KMS key [Info](#)

(default) aws/rds ▾

Account

637423226272

KMS key ID

alias/aws/rds

Log exports

Select the log types to publish to Amazon CloudWatch Logs

- Audit log
- Error log
- General log
- Slow query log

IAM role

The following service-linked role is used for publishing logs to CloudWatch Logs.

RDS service-linked role

Maintenance

Auto minor version upgrade [Info](#)

Enable auto minor version upgrade

Enabling auto minor version upgrade will automatically upgrade to new minor versions as they are released. The automatic upgrades occur during the maintenance window for the database.

Maintenance window [Info](#)

Select the period you want pending modifications or maintenance applied to the database by Amazon RDS.

Choose a window

No preference

Deletion protection

Enable deletion protection

Protects the database from being deleted accidentally. While this option is enabled, you can't delete the database.

Estimated Monthly costs

DB instance	24.82 USD
Storage	4.60 USD
Total	29.42 USD

This billing estimate is based on on-demand usage as described in [Amazon RDS Pricing](#). Estimate does not include costs for backup storage, IOs (if applicable), or data transfer.

Estimate your monthly costs for the DB Instance using the [AWS Simple Monthly Calculator](#).

ⓘ You are responsible for ensuring that you have all of the necessary rights for any third-party products or services that you use with AWS services.

Creating database gelgit-database-1
Your database might take a few minutes to launch. You can use settings from gelgit-database-1 to simplify configuration of [suggested database add-ons](#) while we finish creating your DB for you. [View credential details](#)

Introducing Aurora I/O-Optimized
Aurora's I/O-Optimized is a new cluster storage configuration that offers predictable pricing for all applications and improved price-performance, with up to 40% costs savings for I/O-intensive applications.

RDS > Databases

Consider creating a Blue/Green Deployment to minimize downtime during upgrades
You may want to consider using Amazon RDS Blue/Green Deployments and minimize your downtime during upgrades. A Blue/Green Deployment provides a staging environment for changes to production databases. [RDS User Guide](#) [Aurora User Guide](#)

Databases (1)	DB identifier	Status	Role	Engine	Region & AZ	Size	Recommendations	CPU	Current activity	Maintenance	VPC	Multi-AZ
gelgit-database-1	Creating	Instance	MySQL Community	us-east-1a	db.t3.micro	-	-	none	vpc-0525552ba87a21cf7	Yes		

Introducing Aurora I/O-Optimized
Aurora's I/O-Optimized is a new cluster storage configuration that offers predictable pricing for all applications and improved price-performance, with up to 40% costs savings for I/O-intensive applications.

RDS > Databases

Consider creating a Blue/Green Deployment to minimize downtime during upgrades
You may want to consider using Amazon RDS Blue/Green Deployments and minimize your downtime during upgrades. A Blue/Green Deployment provides a staging environment for changes to production databases. [RDS User Guide](#) [Aurora User Guide](#)

Databases (1)	DB identifier	Status	Role	Engine	Region & AZ	Size	Recommendations	CPU	Current activity	Maintenance	VPC	Multi-AZ
gelgit-database-1	Available	Instance	MySQL Community	us-east-1a	db.t3.micro	4.81%	0 Connections	none	vpc-0525552ba87a21cf7	Yes		

Amazon RDS

RDS > Databases > gelgit-database-1

gelgit-database-1

C Modify Actions ▾

Summary				
DB identifier gelgit-database-1	Status Available	Role Instance	Engine MySQL Community	Recommendations
CPU <div style="width: 5.27%;">5.27%</div>	Class db.t3.micro	Current activity <div style="width: 0%;">0 Connections</div>	Region & AZ us-east-1a	

Connectivity & security Monitoring Logs & events Configuration Zero-ETL integrations Maintenance & backups Tags Recommendations

Connectivity & security

Endpoint & port	Networking	Security
Endpoint gelgit-database-1.c7oma8go68g7.us-east-1.rds.amazonaws.com	Availability Zone us-east-1a	VPC security groups rds-mysql-sg (sg-0225c49f4cc1b63ad)
Port 3306	VPC gelgit-vpc (vpc-0525552ba87a21cf7)	Active
	Subnet group db-private-subnet	Publicly accessible No
	Subnets subnet-08f58b60ff1fcfd8 subnet-0d78bc11596e9768	Certificate authority Info rds-ca-rsa2048-g1
	Network type IPv4	Certificate authority date May 26, 2061, 02:34 (UTC+03:00)
		DB instance certificate expiration date April 14, 2025, 14:56 (UTC+03:00)

Connect DB to Compute Resources

Connected compute resources (1) [Info](#)

Connections to compute resources that were created automatically by RDS are shown here. Connections to compute resources that were created manually aren't shown.

Filter by compute resources

< 1 > ⌂

Resource identifier	Resource type	Availability Zone	VPC security group	Compute resource security group	Connected proxy
i-04d8bd27ceb27b795	EC2 instance	us-east-1a	rds-ec2-1	ec2-rds-1	-

Connect to a MySQL DB instance

- 1) Find the endpoint (DNS name) and port number for your DB instance.

The screenshot shows the AWS RDS console for a database named 'gelgit-database-1'. The 'Connectivity & security' tab is active. In the 'Endpoint & port' section, the endpoint is listed as 'gelgit-database-1.c7oma8go68g7.us-east-1.rds.amazonaws.com' and the port is '3306'. This endpoint URL is highlighted with a red box. Other sections visible include 'Networking' (Availability Zone: us-east-1a, VPC: gelgit-vpc, Subnet group: db-private-subnet, Subnets: subnet-08f58b60ff1fccfda, subnet-0d78bcb11596e9768, Network type: IPv4) and 'Security' (VPC security groups: rds-ec2-1 (sg-0f1f41b4e4e0fc614), rds-mysql-sg (sg-0225c49f4cc1b63ad)).

DB Endpoint:

gelgit-database-1.c7oma8go68g7.us-east-1.rds.amazonaws.com

DB Port:

3306

- 2) Connect to the EC2 instance that you created earlier by following the steps in [Connect to your Linux instance](#) in the Amazon EC2 User Guide for Linux Instances.

We recommend that you connect to your EC2 instance using SSH. If the SSH client utility is installed on Windows, Linux, or Mac, you can connect to the instance using the following command format:

```
ssh -i location_of_pem_file ec2-user@ec2-instance-public-dns-name
```

For example, assume that ec2-database-connect-key-pair.pem is stored in /dir1 on Linux, and the public IPv4 DNS for your EC2 instance is ec2-12-345-678-90.compute-1.amazonaws.com. Your SSH command would look as follows:

```
ssh -i /dir1/ec2-database-connect-key-pair.pem  
ec2-user@ec2-12-345-678-90.compute-1.amazonaws.com
```

For our specific case, you can go to the EC2 instance and select it. After that click on the "connect" button. New page shows you the way of connections. Select SSH client connection and run the given command after locating your private key file in your computer.

In our case:

```
ssh -i "jumb-box-rds-keypair.pem" ec2-user@ec2-54-172-200-114.compute-1.amazonaws.com
```

3) Get the latest bug fixes and security updates by updating the software on your EC2 instance. To do this, use the following command. (The -y option installs the updates without asking for confirmation. To examine updates before installing, omit this option.)

```
sudo dnf update -y
```

4) To install the mysql command-line client from MariaDB on Amazon Linux 2023, run the following command:

```
sudo dnf install mariadb105
```

5) Substitute the DB instance endpoint (DNS name) for endpoint, and substitute the master username that you used for admin. Provide the master password that you used when prompted for a password.

```
mysql -h <endpoint> -P 3306 -u <admin> -p
```

In our project

```
mysql -h gelgit-database-1.c7oma8go68g7.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
```

After you enter the password for the user, you should see output similar to the following.

```
Welcome to the MariaDB monitor. Commands end with ; or \g.  
Your MySQL connection id is 3082  
Server version: 8.0.28 Source distribution
```

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]>

For more information about connecting to a MySQL DB instance, see [Connecting to a DB instance running the MySQL database engine](#). If you can't connect to your DB instance, see Can't connect to Amazon RDS DB instance.

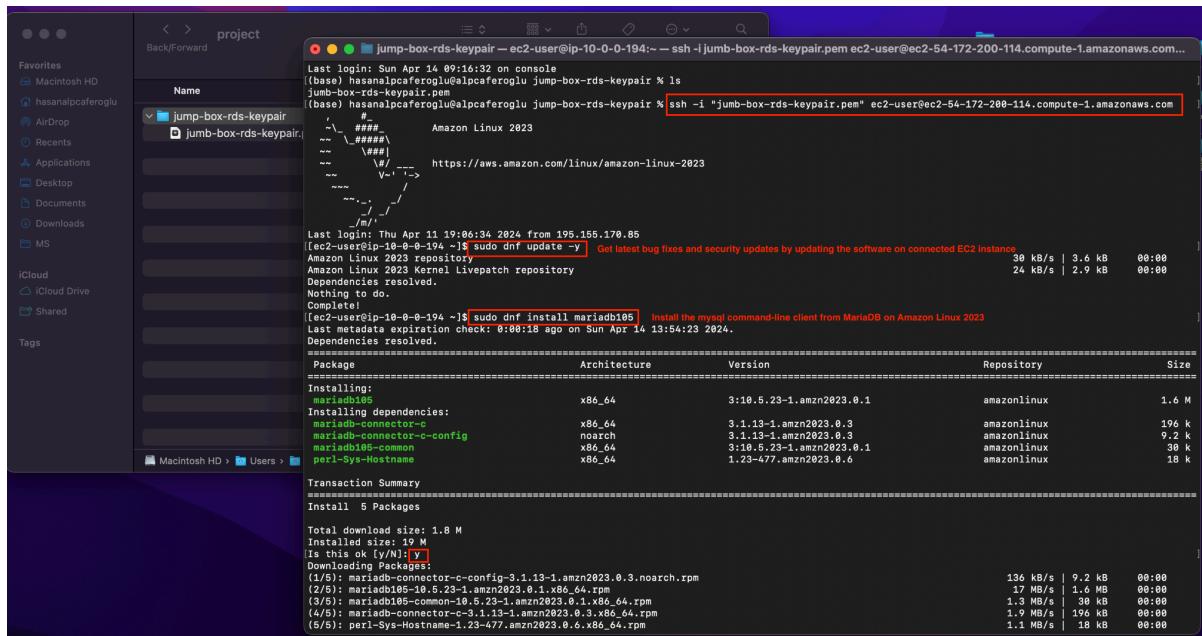
For security, it is a best practice to use encrypted connections. Only use an unencrypted MySQL connection when the client and server are in the same VPC and the network is trusted. For information about using encrypted connections, see [Connecting from the MySQL command-line client with SSL/TLS \(encrypted\)](#).

6) Run SQL commands.

For example, the following SQL command shows the current date and time:

```
SELECT CURRENT_TIMESTAMP;
```

Below you see the steps for connecting MySQL database through EC2 instance.



```

project
Back/Forward
Name
jump-box-rds-keypair
jumb-box-rds-keypair

Installing:
mariadb105                                x86_64        3:10.5.23-1.amzn2023.0.1      amazonlinux          1.6 M
Installing dependencies:
mariadb-connector-c                         x86_64        3.1.13-1.amzn2023.0.3      amazonlinux          196 k
mariadb-connector-c-config                  noarch       3.1.13-1.amzn2023.0.3      amazonlinux          9.2 k
mariadb105-common                          x86_64        3:10.5.23-1.amzn2023.0.1      amazonlinux          38 k
perl-Sys-Hostname                           x86_64        1.23-477.amzn2023.0.6      amazonlinux          18 k

Transaction Summary
=====
Install 5 Packages

Total download size: 1.8 M
Installed size: 19 M
Is this ok [y/N]: y
Downloading Packages:
(1/5): mariadb-connector-c-config-3.1.13-1.amzn2023.0.3.noarch.rpm           136 kB/s | 9.2 kB   00:00
(2/5): mariadb105-10.5.23-1.amzn2023.0.1.x86_64.rpm                         17 MB/s | 1.6 MB   00:00
(3/5): mariadb105-common-3:10.5.23-1.amzn2023.0.1.x86_64.rpm                 1.9 MB/s | 38 kB   00:00
(4/5): mariadb-connector-c-3.1.13-1.amzn2023.0.3.x86_64.rpm                   1.1 MB/s | 194 kB   00:00
(5/5): perl-Sys-Hostname-1.23-477.amzn2023.0.6.x86_64.rpm                     1.1 MB/s | 18 kB   00:00

Total                                         11 MB/s | 1.8 MB   00:00

Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
Preparing:
  Installing : mariadb-connector-c-config-3.1.13-1.amzn2023.0.3.noarch           1/5
  Installing : mariadb-connector-c-3.1.13-1.amzn2023.0.3.x86_64                  2/5
  Installing : mariadb105-3:10.5.23-1.amzn2023.0.1.x86_64                      3/5
  Installing : mariadb105-common-3:10.5.23-1.amzn2023.0.1.x86_64                4/5
  Installing : perl-Sys-Hostname-1.23-477.amzn2023.0.6.x86_64                   5/5
Running scriptlet: mariadb105-3:10.5.23-1.amzn2023.0.1.x86_64
Verifying   : mariadb-connector-c-3.1.13-1.amzn2023.0.3.noarch                   1/5
Verifying   : mariadb-connector-c-config-3.1.13-1.amzn2023.0.3.noarch            2/5
Verifying   : mariadb105-3:10.5.23-1.amzn2023.0.1.x86_64                      3/5
Verifying   : mariadb105-common-3:10.5.23-1.amzn2023.0.1.x86_64                4/5
Verifying   : perl-Sys-Hostname-1.23-477.amzn2023.0.6.x86_64                   5/5

Complete!
[ec2-user@ip-10-0-0-194 ~]$ 

```

```
mysql -h gelgit-database-1.c7oma8go68g7.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
```

Database Master Username: admin
Database Master password: Master2024

```

project
Back/Forward
Name
jump-box-rds-keypair
jumb-box-rds-keypair

(base) hasanipaferoglu@elpcaferoglu jump-box-rds-keypair
(base) hasanipaferoglu@elpcaferoglu jump-box-rds-keypair % ssh -i "jump-box-rds-keypair.pem" ec2-user@ec2-54-172-200-114.compute-1.amazonaws.com
Last login: Sun Apr 14 13:54:08 2024 from 195.155.170.85
(base) hasanipaferoglu@elpcaferoglu ~ % sudo dnf update -y
Last metadata expiration check: 0:05:27 ago on Sun Apr 14 13:54:23 2024.
Dependencies resolved.
Nothing to do.
Complete!
(base) hasanipaferoglu@elpcaferoglu ~ % sudo dnf install mariadb105
Last metadata expiration check: 0:05:27 ago on Sun Apr 14 13:54:23 2024.
Package mariadb105-3:10.5.23-1.amzn2023.0.1.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
(base) hasanipaferoglu@elpcaferoglu ~ % mysql -h gelgit-database-1.c7oma8go68g7.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MySQL connection id is 49.
Server version: 8.0.34 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

MySQL [(none)]> SELECT CURRENT_TIMESTAMP;
+-----+
| CURRENT_TIMESTAMP |
+-----+
| 2024-04-14 14:02:48 |
+-----+
1 row in set (0.002 sec)

MySQL [(none)]> 

```

```

jump-box-rds-keypair -- ec2-user@ip-10-0-0-194:~ ssh -i jump-box-rds-keypair.pem ec2-user@ec2-54-172-200-114.compute-1.amazonaws.com...

MySQL [(none)]> SELECT CURRENT_TIMESTAMP;
+-----+
| CURRENT_TIMESTAMP |
+-----+
| 2024-04-14 14:02:48 |
+-----+
1 row in set (0.002 sec)

MySQL [(none)]> CREATE TABLE User (
    ->     id INT AUTO_INCREMENT,
    ->     email VARCHAR(256) NOT NULL,
    ->     password VARCHAR(256) NOT NULL,
    ->     phone VARCHAR(256) NOT NULL,
    ->     active BOOLEAN NOT NULL DEFAULT TRUE,
    ->     PRIMARY KEY (id),
    ->     UNIQUE (email),
    ->     UNIQUE (phone)
    -> );
This didn't work because no db is selected as you see from MySQL [(none)]
ERROR 1046 (3D000): No database selected
MySQL [(none)]> CREATE DATABASE IF NOT EXISTS gelgitdb
-> :
Creating DB named as gelgitdb in the AWS RDS DB instance
Query OK, 1 row affected (0.023 sec)

MySQL [(none)]> USE gelgitdb      Selecting gelgitdb
Database changed
MySQL [gelgitdb]> CREATE TABLE User (
    ->     id INT AUTO_INCREMENT,
    ->     email VARCHAR(256) NOT NULL,
    ->     password VARCHAR(256) NOT NULL,
    ->     phone VARCHAR(256) NOT NULL,
    ->     active BOOLEAN NOT NULL DEFAULT TRUE,
    ->     PRIMARY KEY (id),
    ->     UNIQUE (email),
    ->     UNIQUE (phone)
    -> );
Creating User table
Query OK, 0 rows affected (0.108 sec)

```

```

jump-box-rds-keypair -- ec2-user@ip-10-0-0-194:~ ssh -i jump-box-rds-keypair.pem ec2-user@ec2-54-172-200-114.compute-1.amazonaws.com...

MySQL [gelgitdb]> INSERT INTO User (id, email, password, phone, active) VALUES
-> (1, 'admin1@example.com', '123456', '555 444 33 22', TRUE),
-> (2, 'admin2@example.com', '123456', '555 444 33 23', TRUE),
-> (3, 'admin3@example.com', '123456', '555 444 33 24', TRUE),
-> (4, 'company1@example.com', '123456', '555 444 33 25', TRUE),
-> (5, 'company2@example.com', '123456', '555 444 33 26', TRUE),
-> (6, 'company3@example.com', '123456', '555 444 33 27', TRUE),
-> (7, 'traveler1@example.com', '123456', '555 444 33 28', TRUE),
-> (8, 'traveler2@example.com', '123456', '555 444 33 29', TRUE),
-> (9, 'traveler3@example.com', '123456', '555 444 33 30', TRUE),
-> (10, 'traveler4@example.com', '123456', '555 444 33 31', TRUE),
-> (11, 'traveler5@example.com', '123456', '555 444 33 32', TRUE),
-> (12, 'Kamilko@example.com', '123456', '555 444 33 33', TRUE),
-> (13, 'Setur@example.com', '123456', '555 444 33 34', TRUE),
-> (14, 'Pegasus@example.com', '123456', '555 444 33 35', FALSE),
-> (15, 'Touristica@example.com', '123456', '555 444 33 36', TRUE),
-> (16, 'EtsTur@example.com', '123456', '555 444 33 37', TRUE),
-> (17, 'InterskyTravel@example.com', '123456', '555 444 33 38', FALSE),
-> (18, 'Ulusoy@example.com', '123456', '555 444 33 39', TRUE),
-> (19, 'Corendon@example.com', '123456', '555 444 33 40', TRUE);
Query OK, 19 rows affected (0.024 sec)
Records: 19  Duplicates: 0  Warnings: 0
Inserting tuples to the DB

```

```

jump-box-rds-keypair -- ec2-user@ip-10-0-0-194:~ ssh -i jump-box-rds-keypair.pem ec2-user@ec2-54-172-200-114.compute-1.amazonaws.com...

MySQL [gelgitdb]> SELECT id
-> FROM User
-> :
+---+
| id |
+---+
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 6 |
| 19 |
| 16 |
| 17 |
| 12 |
| 14 |
| 13 |
| 15 |
| 7 |
| 8 |
| 10 |
| 11 |
| 18 |
+---+
19 rows in set (0.001 sec)

```

MySQL [gelgitdb]> SELECT * FROM User;					
id	email	password	phone	active	
1	admin1@example.com	123456	555 444 33 22	1	
2	admin2@example.com	123456	555 444 33 23	1	
3	admin3@example.com	123456	555 444 33 24	1	
4	company1@example.com	123456	555 444 33 25	1	
5	company2@example.com	123456	555 444 33 26	1	
6	company3@example.com	123456	555 444 33 27	1	
7	traveler1@example.com	123456	555 444 33 28	1	
8	traveler2@example.com	123456	555 444 33 29	1	
9	traveler3@example.com	123456	555 444 33 30	1	
10	traveler4@example.com	123456	555 444 33 31	1	
11	traveler5@example.com	123456	555 444 33 32	1	
12	Kamilkoc@example.com	123456	555 444 33 33	1	
13	Setur@example.com	123456	555 444 33 34	1	
14	Pegasus@example.com	123456	555 444 33 35	0	
15	Touristica@example.com	123456	555 444 33 36	1	
16	EtsTur@example.com	123456	555 444 33 37	1	
17	InterskyTravel@example.com	123456	555 444 33 38	0	
18	Ulusoy@example.com	123456	555 444 33 39	1	
19	Corendon@example.com	123456	555 444 33 40	1	

19 rows in set (0.002 sec)

Useful Commands for MySQL Server

```
// show the list of databases available in your MySQL server
SHOW DATABASES;

// Switch to the database you want to work with
USE <database_name>;

// switch to the no database
USE ;

// delete a database
DROP DATABASE <database_name>;

// create a database
CREATE DATABASE IF NOT EXISTS <database_name>;
```

```
// view tables within the database
SHOW TABLES;

// To see the structure of a specific table. This will show you the columns and their data types.
DESCRIBE <table_name>;
DESC <table_name>;

//information about a table, including its size, number of rows
SHOW TABLE STATUS LIKE <table_name>;

// view a list of MySQL server system variables
SHOW VARIABLES;

// display information about the MySQL storage engines and their status
SHOW ENGINE STATUS;
```

```
// To see the views in your MySQL database;  
// This query retrieves the names of all views in the specified database.  
SHOW FULL TABLES IN <database_name> WHERE TABLE_TYPE LIKE 'VIEW';  
// or  
SELECT TABLE_SCHEMA, TABLE_NAME FROM information_schema.VIEWS WHERE  
TABLE_SCHEMA LIKE 'your_db_name';
```

```
SHOW FULL TABLES IN gelgitdb WHERE TABLE_TYPE LIKE 'VIEW';
```

Connecting EC2 with Session Manager

- Do the steps in homework 1. First you need to create a system manager role from IAM.
 - IAM → roles → create roles → trusted entity type: AWS service → use case: EC2 → next → add permissions: select **AmazonSSMManagedInstanceCore** → role name: SSMInstanceProfile → create role
- Attach the Systems Manager instance profile to an existing instance
 - EC2 console → choose ec2 instance → actions → security → modify IAM rule → IAM role: select SSMInstanceProfile → update IAM role
- Now it is time for connection using session manager
 - EC2 console → select ec2 → Connect → session manager

After these I couldn't connect with session manager. Messages below are taken

Connect to instance Info

Connect to your instance i-04d8bd27ceb27b795 (gelgit-jumpbox-rds-ec2) using any of these options

EC2 Instance Connect	Session Manager	SSH client	EC2 serial console
<p>⚠️ SSM Agent is not online</p> <p>The SSM Agent was unable to connect to a Systems Manager endpoint to register itself with the service.</p> <ul style="list-style-type: none"> Verify that the IAM instance profile has the correct permissions. 🔗 Verify that your instance's security group and VPC allow HTTPS (port 443) outbound traffic to the following Systems Manager endpoints: <ul style="list-style-type: none"> ssm.us-east-1.amazonaws.com ec2messages.us-east-1.amazonaws.com ssmmessages.us-east-1.amazonaws.com <p>If your VPC does not have internet access, you can use VPC endpoints 🔗 to allow outbound traffic from your instance.</p> <p>If you still can't connect to your instance, or if you receive an error, including an error about SSM Agent, see:</p> <ul style="list-style-type: none"> Troubleshooting Session Manager 🔗 Troubleshooting managed node availability using ssm-cli 🔗 Troubleshooting with Automation Runbook AWSSupport-TroubleshootSessionManager 🔗 			

Session Manager usage:

- Connect to your instance without SSH keys, a bastion host, or opening any inbound ports.
- Sessions are secured using an AWS Key Management Service key.
- You can log session commands and details in an Amazon S3 bucket or CloudWatch Logs log group.
- Configure sessions on the Session Manager [Preferences](#) [🔗](#) page.

To overcome this issue, I made changes below in security groups.

Security Groups (7) Info						
<input type="text"/> Find resources by attribute or tag C Actions Export security groups to CSV Create security group 						
	Name	Security group ID	Security group name	VPC ID	Description	Owner
<input type="checkbox"/>	-	sg-0ff1f41b4e4e0fc614	rds-ec2-1	vpc-0525552ba87a21cf7 🔗	Security group attached to gelgit-data...	63742322
<input type="checkbox"/>	-	sg-039f5f98aa0d2f68f	default	vpc-09451332da0e3923c 🔗	default VPC security group	63742322
<input type="checkbox"/>	jumpbox-rds-sg	sg-0a97b60eebccf04dd	jumpbox-rds	vpc-0525552ba87a21cf7 🔗	jumpbox-rds-sg	63742322
<input type="checkbox"/>	-	sg-06083edeef501543af	default	vpc-0525552ba87a21cf7 🔗	default VPC security group	63742322
<input type="checkbox"/>	rds-mysql-sg	sg-0225c49f4cc1b63ad	rds-mysql-sg	vpc-0525552ba87a21cf7 🔗	Created by RDS management console	63742322

Inbound rules (3)						
<input type="text"/> Search C Manage tags Edit inbound rules 						
	Name	Security group rule ID	IP version	Type	Protocol	Port range
<input type="checkbox"/>	-	sgr-09d7c91d3221c82d9	IPv4	SSH	TCP	22
<input type="checkbox"/>	-	sgr-0924b195ac136f01f	IPv4	SSH	TCP	22
<input type="checkbox"/>	-	sgr-05d51c240a6b20818	IPv4	HTTPS	TCP	443

Security Groups (1/7) [Info](#)

<input type="checkbox"/>	Security group ID	Security group name	VPC ID	Description
sg-0a97b60eefccf04dd	jumpbox-rds	vpc-0525552ba87a21cf7	jumpbox-rds-sg	
sg-06083edef501543af	default	vpc-0525552ba87a21cf7	default VPC security group	
sg-0225c49f4cc1b63ad	rds-mysql-sg	vpc-0525552ba87a21cf7	Created by RDS management console	
sg-07169aaa8f80fe8d1	aws-cloud9-AmplifyTest-e207b58629...	vpc-09451332da0e3923c	Security group for AWS Cloud9 environment aws-cloud9-AmplifyTest-e207b58629c4bf5	
sg-055d7c0c612e361d6	ec2-rds-1	vpc-0525552ba87a21cf7	Security group attached to instances to securely connect to gelgit-database-1. Modification	

Details Inbound rules **Outbound rules** Tags

Outbound rules (2)

<input type="checkbox"/>	Name	Security group rule ID	IP version	Type	Protocol	Port range	Destination
<input type="checkbox"/>	-	sgr-07805044c93df7acb	IPv4	All traffic	All	All	0.0.0.0/0 This one added
<input type="checkbox"/>	-	sgr-0738bd25fb4382de	-	MySQL/Aurora	TCP	3306	sg-0f1f41b4e4e0fc614 / rds-ec2-

After these changes,

[EC2](#) > [Instances](#) > [i-04d8bd27ceb27b795](#) > Connect to instance

Connect to instance [Info](#)

Connect to your instance i-04d8bd27ceb27b795 (gelgit-jumpbox-rds-ec2) using any of these options

EC2 Instance Connect **Session Manager** SSH client EC2 serial console

Session Manager usage:

- Connect to your instance without SSH keys, a bastion host, or opening any inbound ports.
- Sessions are secured using an AWS Key Management Service key.
- You can log session commands and details in an Amazon S3 bucket or CloudWatch Logs log group.
- Configure sessions on the Session Manager [Preferences](#) page.

Cancel **Connect**

Session ID: alponzo-047c90db267b7a5ad Instance ID: i-04d8bd27ceb27b795 [Terminate](#)

```
sh-5.2$ mysql -h gelgit-database-1.c7oma8go68g7.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
Enter password:
ERROR 1045 (28000): Access denied for user 'admin'@'10.0.0.194' (using password: YES)
sh-5.2$ mysql -h gelgit-database-1.c7oma8go68g7.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \q.
Your MySQL connection id is 127
Server version: 8.0.34 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| gelgit_database |
| gelgitdb |
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
6 rows in set (0.013 sec)

MySQL [(none)]> USE gelgitdb;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [gelgitdb]> SHOW TABLES
-> ;
+-----+
| Tables_in_gelgitdb |
+-----+
| User |
+-----+
1 row in set (0.002 sec)

MySQL [gelgitdb]>
```

```
mysql -h gelgit-database-1.c7oma8go68g7.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
```

Database Master Username: admin
Database Master password: Master2024

Creating Lambda and Connecting it with API Gateway

- 1) Create a lambda function
 - a) enter lambda function name
 - b) select runtime
 - c) set advanced settings (for example in our project, we need to enable VPC: select vpc, select subnets for lambda, choose security group)
 - d) Create lambda
 - i) Don't forget to select VPC (gelgit-vpc), two private subnet and security group named as "lambda-rds-1"
 - e) For the created lambda, you can change the code but you should deploy it.

The screenshot shows the 'Create function' wizard in the AWS Lambda console. The top navigation bar includes 'Lambda', 'Functions', and 'Create function'. Below the title 'Create function' is a note: 'Choose one of the following options to create your function.' There are three tabs: 'Author from scratch' (selected), 'Use a blueprint', and 'Container image'. The 'Basic information' section contains fields for 'Function name' (set to 'logout-lambda'), 'Runtime' (set to 'Python 3.9'), and 'Architecture' (set to 'x86_64'). The 'Permissions' section notes that Lambda will create an execution role with CloudWatch Logs permissions. At the bottom, there's a link to 'Change default execution role'.

Advanced settings

Enable Code signing [Info](#)
Use code signing configurations to ensure that the code has been signed by an approved source and has not been altered since signing.

Enable function URL [Info](#)
Use function URLs to assign HTTPS endpoints to your Lambda function.

Enable tags [Info](#)
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources, track your AWS costs, and enforce attribute-based access control.

Enable VPC [Info](#)
Connect your function to a VPC to access private resources during invocation.

VPC
Choose a VPC for your function to access.
 [Copy](#)

Allow IPv6 traffic for dual-stack subnets
You can allow outbound IPv6 traffic to subnets that have both IPv4 and IPv6 CIDR blocks.

Subnets
Select the VPC subnets for Lambda to use to set up your VPC configuration.
 [Copy](#)

subnet-08f88b60f1fcfda (10.0.144.0/24) us-east-1b X subnet-0d78bcb11596e9768 (10.0.128.0/24) us-east-1a X
Name: gelgit-subnet-private2-us-east-1b Name: gelgit-subnet-private1-us-east-1a

Security groups
Choose the VPC security groups for Lambda to use to set up your VPC configuration. The table below shows the inbound and outbound rules for the security groups that you choose.
 [Copy](#)

sg-02f8d4ba75abfaa6c (lambda-rds-1)
Security group attached to Lambda function to allow them to securely connect to gelgit-database-1. Modification could lead to connection loss.

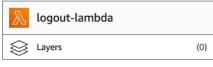
[Inbound rules](#) [Outbound rules](#)

aws Services Search [Option+S] N. Virginia alfonzo @ 6374-2322-6272

logout-lambda

Function overview [Info](#)

[Diagram](#) [Template](#)



[+ Add trigger](#) [+ Add destination](#)

Description
Last modified 29 seconds ago

Function ARN [arn:aws:lambda:us-east-1:637423226272:function:logout-lambda](#)

Function URL [Info](#)

Code [Test](#) [Monitor](#) [Configuration](#) [Aliases](#) [Versions](#)

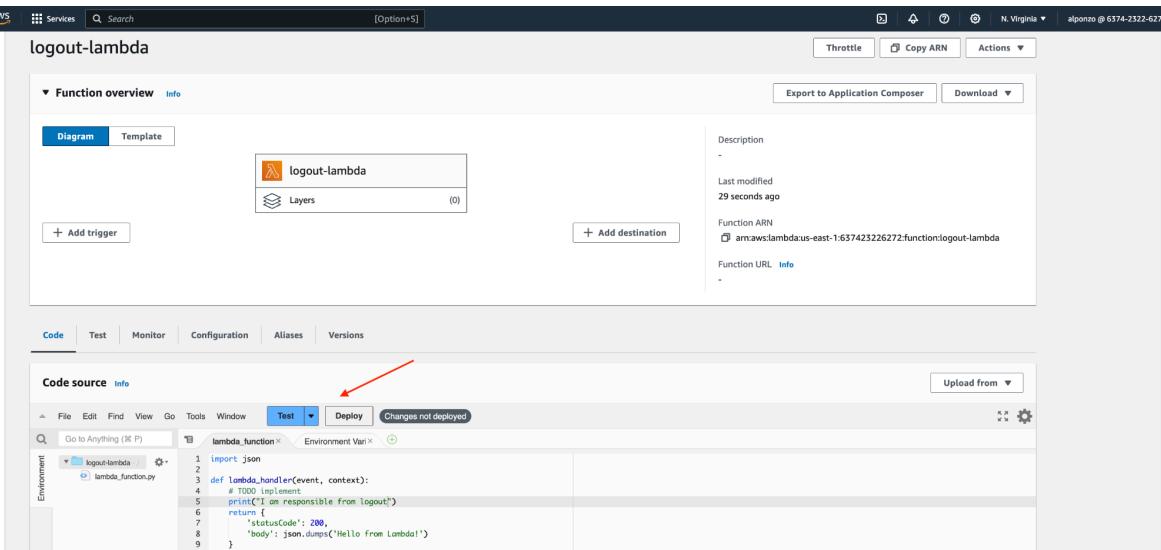
Code source [Info](#)

[File](#) [Edit](#) [Find](#) [View](#) [Go](#) [Tools](#) [Window](#) [Test](#) [Deploy](#) [Changes not deployed](#)

Upload from [...](#)

lambda_function

```
1 import json
2
3 def lambda_handler(event, context):
4     # TODO implement
5     print("From responsible from logout")
6     return {
7         'statusCode': 200,
8         'body': json.dumps('Hello from Lambda!')
9     }
```



- 2) You may need to create a model. Go to API Gateway console and go to the model bar.
 - a) Create a model
 - b) Give it a name
 - c) Write content type (application/json)
 - d) Fill Model schema

The screenshot shows the AWS API Gateway interface. On the left, the navigation bar includes 'APIs', 'Custom domain names', 'VPC links', and a collapsed 'API: login' section containing 'Resources', 'Stages', 'Authorizers', 'Gateway responses', and 'Models'. The 'Models' section is expanded, showing 'Resource policy', 'Documentation', 'Dashboard', and 'API settings'. Below these are 'Usage plans', 'API keys', 'Client certificates', and 'Settings'. The main content area is titled 'Model details' for a model named 'loginRequestBody'. It includes fields for 'Content type' (set to 'application/json') and 'Description - optional'. The 'Model schema' field contains the following JSON schema:

```

1 v {
2   "$schema": "http://json-schema.org/draft-04/schema#",
3   "title": "loginRequestBody",
4   "type": "object",
5   "properties": {
6     "email": {
7       "type": "string"
8     },
9     "password": {
10       "type": "string"
11     }
12   },
13   "required": ["email", "password"]
14 }

```

- 3) Navigate to the Resources bar in API Gateway.
 - a) Click on Create resource
 - b) Select resource path and write resource name, then create it.
- 4) Now, we need to create a method for that resource.
 - a) Click on the created resource
 - b) Click on create method
 - c) Select method type
 - d) Select lambda function
 - e) Select Method request settings
 - i) You can select AWS Cognito if you have it or you can edit the method after creating AWS Cognito
 - f) Select Request body if needed from the models you created.

The screenshot shows the AWS API Gateway interface. The left sidebar is identical to the previous screenshot. The main content area is titled 'Stages' and shows a single stage named 'dev'. A green banner at the top says 'Successfully edited method request for POST.'. The 'Stage details' section includes fields for 'Stage name' (set to 'dev'), 'Rate info' (empty), 'Web ACL' (empty), 'Cache cluster info' (set to 'Inactive'), 'Burst info' (empty), 'Client certificate' (empty), 'Default method-level caching' (set to 'Inactive'), 'Invoke URL' (set to 'https://3yyw8u4a4m.execute-api.us-east-1.amazonaws.com/dev'), and 'Active deployment' (set to 'qklql on April 17, 2024, 15:18 (UTC+03:00)'). The 'Logs and tracing' section includes fields for 'CloudWatch logs' (set to 'Inactive'), 'Detailed metrics' (set to 'Inactive'), and 'X-Ray tracing' (set to 'Inactive')).

- 5) Deploy API to a selected stage
 - a) You can select a *New Stage* and name it as you wish
- 6) Navigate to Stages bar from the API Gateway
 - a) Select the stage
 - b) Select the resource
 - c) Select the method you want to invoke
 - d) Copy the Invoke URL

- 7) Create a request from postman using method's invoke url and observe

```

1 {
2   ...."email": "traveler1@example.com",
3   ...."password": "123456"
4 }

```

```

1 {
2   "statusCode": 200,
3   "body": "\\"Hello from Lambda Login!\\""
4 }

```

- 8) You can monitor Lambda Function Logs using CloudWatch logs. If you have print in your lambda function, you can see these prints using CloudWath

The screenshot shows the AWS Lambda console. In the top navigation bar, 'Services' is selected. Under 'Lambda > Functions > login-lambda', the 'Function overview' tab is active. On the left, there's a diagram showing 'login-lambda' connected to an 'API Gateway' trigger. On the right, there's a 'Description' section with details like 'Last modified 3 hours ago', 'Function ARN arn:aws:lambda:us-east-1:637423226272:function:login-lambda', and a 'Function URL'. Below the diagram, tabs for 'Code', 'Test', 'Monitor', 'Configuration', 'Aliases', and 'Versions' are shown. The 'Monitor' tab is highlighted with a red arrow pointing to the 'View CloudWatch logs' button.

This screenshot shows the CloudWatch Log Group details for the '/aws/lambda/login-lambda' group. The left sidebar has 'CloudWatch' selected under 'Logs'. The main area shows 'Log group details' for the standard log group. It lists ARN (arn:aws:logs:us-east-1:637423226272:log-group:/aws/lambda/login-lambda), Creation time (5 hours ago), Retention (Never expire), and Stored bytes (-). Below this, the 'Log streams' section is shown, containing three log entries. The first entry, dated 2024-04-17 at 20:28:38, is highlighted with a red arrow.

This screenshot shows the CloudWatch Log Events page for the '/aws/lambda/login-lambda' log stream. The left sidebar has 'CloudWatch' selected under 'Logs'. The main area shows a table of log events. A search bar at the top contains the text 'I am lambda responsible from login o ye'. One of the log entries in the table matches this search term and is highlighted with a red box. The table includes columns for 'Timestamp' and 'Message', with additional filters and actions buttons at the bottom.

Example Lambda Function:

```
import json
import sys
import logging
import pymysql
import os

"""
The code creates the connection to your database outside of the handler function.
Creating the connection in the initialization code allows the connection to be
re-used by subsequent invocations of your function and improves performance.
"""

# rds settings
user_name = os.environ['USER_NAME']
password = os.environ['PASSWORD']
rds_proxy_host = os.environ['RDS_PROXY_HOST']
db_name = os.environ['DB_NAME']

logger = logging.getLogger()
logger.setLevel(logging.INFO)

# create the database connection outside of the handler to allow connections to be
# re-used by subsequent function invocations.
try:
    conn = pymysql.connect(host=rds_proxy_host, user=user_name, passwd=password,
                           db=db_name, connect_timeout=5)
except pymysql.MySQLError as e:
    logger.error("ERROR: Unexpected error: Could not connect to MySQL instance.")
    logger.error(e)
    sys.exit(1)

logger.info("SUCCESS: Connection to RDS for MySQL instance succeeded")

def lambda_handler(event, context):
    """
    This function inserts a new tuple to Traveler and User Tables
    """
    # message = event['Records'][0]['body']
    # data = json.loads(message)
    data = event
    email = data['email']
    password = data['password']
    passwordRepeat = data['passwordRepeat']
    phone = data['phone']
    TCK = data['TCK']
    name = data['name']
    surname = data['surname']
    age = data['age']
```

```

item_count = 0
sql_string = f"insert into Traveler (id, TCK, name, surname, age, balance) values(%s, %s, %s, %s, %s, %s)"

with conn.cursor() as cursor:

    cursor.execute('SELECT * FROM User WHERE email = %s ', (email, ))
    accountExistWithSameEmail = cursor.fetchone()

    cursor.execute('SELECT * FROM User WHERE phone = %s ', (phone, ))
    accountExistWithSamePhone = cursor.fetchone()

    cursor.execute('SELECT * FROM Traveler WHERE TCK = %s ', (TCK, ))
    accountExistWithSameTCK = cursor.fetchone()

if not email or not password or not phone or not TCK or not name or not surname or not age:
    message = 'Please fill out the form!'
elif accountExistWithSameEmail:
    message = 'There is already an account with that email!'
elif accountExistWithSameTCK:
    message = 'There is already an account with this TCK!'
elif accountExistWithSamePhone:
    message = 'There is already an account with this phone number!'
elif password != passwordRepeat:
    message = 'Password mismatch. Please enter same password!'
elif int(age) < 18:
    message = 'To register, you should be older than 18!'
else:
    cursor.execute('INSERT INTO User (id, email, password, phone, active) VALUES (NULL, %s, %s, %s, TRUE)', (email, password, phone, ))
    conn.commit()
    # find newly added traveler
    cursor.execute('SELECT * FROM User WHERE email = %s ', (email, ))
    newUser = cursor.fetchone()
    # get the user id and insert this user into Traveler
    newUserId = newUser[0] # index 0 corresponds to the id
    cursor.execute('INSERT INTO Traveler (id, TCK, name, surname, age, balance) VALUES (%s, %s, %s, %s, %s, 0)', (newUserId, TCK, name, surname, age, ))
    conn.commit()
    message = 'User successfully created! Please log in.'

    logger.info(message)

    conn.commit()

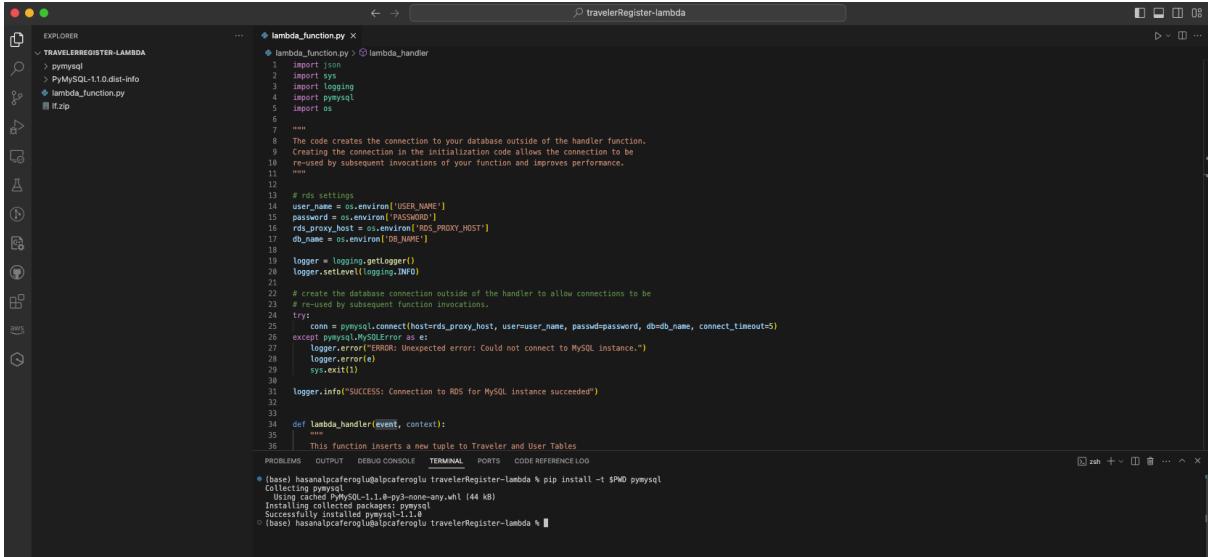
return {
    'statusCode': 200,
    'body': message
}

```

Using a Lambda function to access an Amazon RDS database

Youtube Link: <https://www.youtube.com/watch?v=vyLvmPkQZkI&t=612s>

Tutorial link: <https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/rds-lambda-tutorial.html>



The screenshot shows the AWS Lambda Function Editor interface. The left sidebar displays a file tree for a project named 'TRAVELERREGISTER-LAMBDA'. Inside the project, there are three files: 'lambda_function.py', 'pymysql', and 'PyMySQL-1.1.0.dist-info'. The main editor area contains the code for 'lambda_function.py'.

```
#!/usr/bin/python
# This code creates the connection to your database outside of the handler function.
# Creating the connection in the initialization code allows the connection to be
# re-used by subsequent invocations of your function and improves performance.
#
# rds settings
user_name = os.environ['USER_NAME']
password = os.environ['PASSWORD']
rds_proxy_host = os.environ['RDS_PROXY_HOST']
db_name = os.environ['DB_NAME']

logger = logging.getLogger()
logger.setLevel(logging.INFO)

# create the database connection outside of the handler to allow connections to be
# re-used by subsequent function invocations.
try:
    conn = pymysql.connect(host=rds_proxy_host, user=user_name, passwd=password, db=db_name, connect_timeout=5)
except pymysql.MySQLError as e:
    logger.error("ERROR: Unexpected error: Could not connect to MySQL instance.")
    logger.error(e)
    sys.exit(1)

logger.info("SUCCESS: Connection to RDS for MySQL instance succeeded")

def lambda_handler(event, context):
    """
    This function inserts a new tuple to Traveler and User Tables
    """
    pass
```

The terminal tab at the bottom shows the command used to install the 'pymysql' package:

```
(base) hasan@laptop:~/aws/travelerRegister-lambda % pip install -t $PWD pymysql
Collecting pymysql
  Downloading pymysql-1.1.0-py3-none-any.whl (44 kB)
Installing collected packages: pymysql
Successfully installed pymysql-1.1.0
(base) hasan@laptop:~/aws/travelerRegister-lambda %
```

- Since we use pymysql package in our lambda, we need to instal required package
- So, in local, I open a folder and I write a lambda function. Also, I installed pymysql which creates package files. These are pymysql and PyMySQL-1.1.0.dist-info
- pip install -t \$PWD pymysql
- After lambda_function.py file written and necessary package installed, zip these 3 files.

After that, click on "Up From" button and upload the zip file

Note that you have to set the environment variables from the configuration - environment variables.

Key	Value
DB_NAME	gelgitdb
PASSWORD	Master2024
RDS_PROXY_HOST	gelgit-database-1.c7oma8go68g7.us-east-1.rds.amazonaws.com
USER_NAME	admin

The screenshot shows the AWS Lambda console. The top navigation bar includes 'Services', 'Search', and 'Option+S'. The main title is 'travelerRegister-lambda'. Below it, there's a 'Function overview' section with tabs for 'Diagram' (selected) and 'Template'. It shows a diagram with a Lambda icon labeled 'travelerRegister-lambda' and an API Gateway icon. To the right, there are buttons for 'Throttle', 'Copy ARN', and 'Actions'. Below the diagram, there are sections for 'Description', 'Last modified' (11 minutes ago), 'Function ARN' (arn:aws:lambda:us-east-1:657423226272:function:travelerRegister-lambda), and 'Function URL' (info). The 'Configuration' tab is active, showing the 'Environment variables' section with four entries: DB_NAME (gelgitdb), PASSWORD (Master2024), RDS_PROXY_HOST (gelgit-database-1.c7oma8go68g7.us-east-1.rds.amazonaws.com), and USER_NAME (admin). A sidebar on the left lists 'General configuration', 'Triggers', 'Permissions', 'Destinations', 'Function URL', 'Environment variables' (selected), 'Tags', and 'VPC'.

Also be sure that Lambda is connected with RDS. If not, create connection.

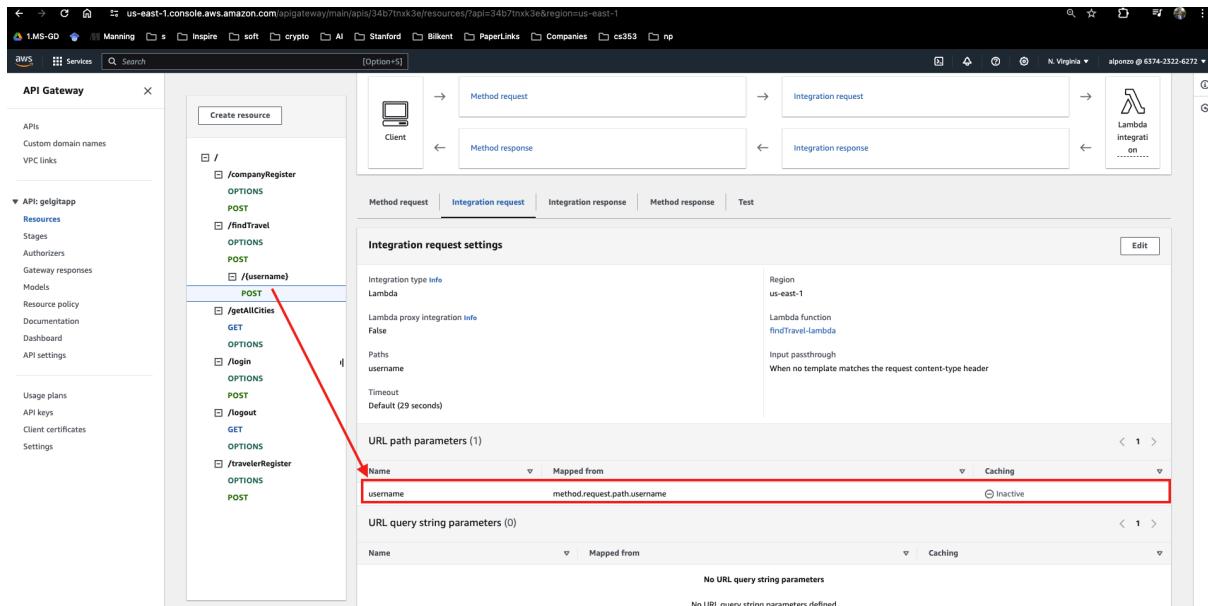
The screenshot shows the AWS Lambda console with the 'travelerRegister-lambda' function. The 'RDS databases' tab is selected. It displays a table titled 'RDS database connections (1)' with one entry: 'gelgit-database-1'. The table columns are 'DB identifier', 'Proxy identifier', 'Database status', 'Lambda security group', and 'DB security group'. The status is 'Available'. The Lambda security group is 'lambda-rds-1 sg-02f6d4ba75a...' and the DB security group is 'rds-lambda-1 sg-0935e58f6a7...'. There are buttons for 'Add Proxy' and 'Connect to RDS database'.

Enable CORS on a resource using the API Gateway console

<https://docs.aws.amazon.com/apigateway/latest/developerguide/how-to-cors-console.html>

The screenshot shows the AWS API Gateway 'Enable CORS' configuration page. At the top, there's a navigation bar with the AWS logo, 'Services' dropdown, search bar, and 'Option+S' keyboard shortcut. Below the navigation, the path is shown as 'API Gateway > APIs > Resources - gelgitapp (34b7tnxk3e) > Enable CORS'. The main title is 'Enable CORS'. Under 'CORS settings', there's an 'Info' section with a note: 'To allow requests from scripts running in the browser, configure cross-origin resource sharing (CORS) for your API.' The 'Gateway responses' section has two checked options: 'Default 4XX' and 'Default 5XX'. The 'Access-Control-Allow-Methods' section has two checked methods: 'OPTIONS' and 'POST'. The 'Access-Control-Allow-Headers' section contains a list box with 'Content-Type,X-Amz-Date,Authorization,X-Api-Key,X-Amz-Security-Token'. The 'Access-Control-Allow-Origin' section has a text input field containing a single asterisk (*). Under 'Additional settings', there's an 'Access-Control-Expose-Headers' section with an empty text input field. The 'Access-Control-Max-Age' section has an empty text input field. A checkbox for 'Access-Control-Allow-Credentials' is present with the note: 'Select to allow requests that include credentials.' At the bottom right, there are 'Cancel' and 'Save' buttons.

Path Variable



<https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-create-api-step-by-step.html>

Created API Gateway connected to Lambda Functions are as follows

- login
- logout
- companyRegister
- travelerRegister
- travels
(@app.route('/travel/<string:vehicle_type>/from:<string:departure_city>/to:<string:arrival_city>/date:<string:departure_date>/extra_date:<string:extra_date>', methods=['GET']))
- findTravel (@app.route('/findTravel', methods=['GET', 'POST']))
- myTravels (@app.route('/myTravels', methods=['GET', 'POST']))
- makeComment(travel_id) (@app.route('/makeComment/<int:travel_id>', methods = ['GET', 'POST']))
- deleteComment(travel_id, traveler_id, company_id)
(@app.route('/deleteComment/<int:travel_id>/<int:traveler_id>/<int:company_id>', methods = ['GET', 'POST']))
- buy_travel(travel_id)(@app.route('/travel/buy/<int:travel_id>', methods=['GET', 'POST']))

- coupons() @app.route('/coupons', methods=['GET', 'POST'])
- userProfile() @app.route('/userProfile', methods=['GET', 'POST'])
- updateTravelerProfile() @app.route('/updateTravelerProfile/', methods=['GET', 'POST'])
- balance() @app.route('/balance', methods = ['GET', 'POST'])
- journeys() @app.route('/journeys', methods = ['GET', 'POST'])
- buy_all() app.route('/buy_all', methods=['POST'])
- companysAllTravels(upcomingOrPast)
 - @app.route('/companysAllTravels/<string:upcomingOrPast>', methods = ['GET', 'POST'])
- addCompanyTravel(travelVehicleType)
 - @app.route('/addCompanyTravel/<string:travelVehicleType>', methods = ['GET', 'POST'])
- aTravelDetails(travelId) @app.route('/aTravelDetails/<int:travelId>', methods = ['GET', 'POST'])
- commentsOnATravel(travelId) @app.route('/commentsOnATravel/<int:travelId>', methods = ['GET', 'POST'])
- editUpcomingTravel(travelId) @app.route('/editUpcomingTravel/<int:travelId>', methods = ['GET', 'POST'])
- deleteATravel(travelId) @app.route('/deleteATravel/<int:travelId>', methods = ['GET', 'POST'])
- deleteAndRefundAPurchase(PNRToBeDeleted)
 - @app.route('/deleteAndRefundAPurchase/<string:PNRToBeDeleted>', methods=['GET', 'POST'])
- deleteWORefundAPurchase(PNRToBeDeleted)
 - @app.route('/deleteWORefundAPurchase/<string:PNRToBeDeleted>', methods=['GET', 'POST'])
- deleteAndGiveFreeTravel(PNRToBeDeleted)
 - @app.route('/deleteAndGiveFreeTravel/<string:PNRToBeDeleted>', methods=['GET', 'POST'])
- deleteAReservation(PNRToBeDeleted)
 - @app.route('/deleteAReservation/<string:PNRToBeDeleted>', methods=['GET', 'POST'])
- companyProfile(companyId) @app.route('/companyProfile/<int:companyId>', methods=['GET', 'POST'])
- editCompanyProfile(companyId) @app.route('/editCompanyProfile/<int:companyId>', methods=['GET', 'POST'])
- BELOW ARE ADMIN RELATED ROUTES
- makePurchaseOnBehalfOfTraveler(travelId)
 - @app.route('/makePurchaseOnBehalfOfTraveler/<int:travelId>', methods = ['GET', 'POST'])
- companies() @app.route('/companies', methods=['GET', 'POST'])
- deactivateCompany(companyId) @app.route('/deactivateCompany/<int:companyId>', methods = ['GET', 'POST'])
- deleteCompany(companyId) @app.route('/deleteCompany/<int:companyId>', methods = ['GET', 'POST'])
- activateCompany(companyId) @app.route('/activateCompany/<int:companyId>', methods = ['GET', 'POST'])
- validateCompany(companyId) @app.route('/validateCompany/<int:companyId>', methods = ['GET', 'POST'])

- `couponManagement() @app.route('/couponManagement', methods = ['GET', 'POST'])`
- `createCoupon() @app.route('/createCoupon', methods = ['GET', 'POST'])`
- `deleteACoupon(couponId) @app.route('/deleteACoupon/<int:couponId>', methods = ['GET', 'POST'])`
- `vehicleManagement() @app.route('/vehicleManagement', methods = ['GET', 'POST'])`
- `createVehicleType() @app.route('/createVehicleType', methods = ['GET', 'POST'])`
- `deleteAVehicleType(vehicleTypeId) @app.route('/deleteAVehicleType/<int:vehicleTypeId>', methods = ['GET', 'POST'])`
- `terminalManagement() @app.route('/terminalManagement', methods = ['GET', 'POST'])`
- `deleteTerminal(terminalId) @app.route('/deleteTerminal/<int:terminalId>', methods = ['GET', 'POST'])`
- `createTerminal() @app.route('/createTerminal', methods = ['GET', 'POST'])`
- `reportManagement() @app.route('/reportManagement', methods = ['GET', 'POST'])`
- `reportDetails(report_id) @app.route('/reportDetails/<int:report_id>', methods = ['GET', 'POST'])`
- `printReportDetails(report_id) @app.route('/printReportDetails/<int:report_id>', methods=['GET', 'POST'])`
- `createReport() @app.route('/createReport', methods = ['GET', 'POST'])`
- `companysAllTravelsByAdmin(upcomingOrPast, companyId)
@app.route('/companysAllTravelsByAdmin/<int:companyId>/<string:upcomingOrPast>', methods = ['GET', 'POST'])`

Cognito

What is Amazon Cognito?

<https://docs.aws.amazon.com/cognito/latest/developerguide/what-is-amazon-cognito.html>

Common Amazon Cognito Scenarios:

<https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-scenarios.html>

Using tokens with user pools:

<https://docs.aws.amazon.com/cognito/latest/developerguide/amazon-cognito-user-pools-using-tokens-with-identity-providers.html>

Authentication with a user pool:

<https://docs.aws.amazon.com/cognito/latest/developerguide/authentication.html>

Integrating Amazon Cognito authentication and authorization with web and mobile apps:

<https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-integrate-apps.html>

Amplify Auth Concepts:

<https://docs.amplify.aws/react/build-a-backend/auth/concepts/>

Token And Credentials:

<https://docs.amplify.aws/react/build-a-backend/auth/concepts/tokens-and-credentials/>

Set up Amplify Auth:

<https://docs.amplify.aws/react/build-a-backend/auth/set-up-auth/>

- Instead of Authenticator, you can bring your own UI and leverage the library from [aws-amplify](#) to handle authentication flows manually. Since we have custom ui, we will probably use this library.

Enable sign-up, sign-in, and sign-out:

<https://docs.amplify.aws/gen1/react/prev/build-a-backend/auth/enable-sign-up/>

<https://docs.amplify.aws/gen1/react/build-a-backend/auth/enable-sign-up/>

Configure Amplify:

<https://docs.amplify.aws/gen1/react/build-a-backend/auth/set-up-auth/>

<https://docs.amplify.aws/gen1/react/tools/libraries/configure-categories/>

Manage user sessions

<https://docs.amplify.aws/react/build-a-backend/auth/connect-your-frontend/manage-user-sessions/>

Manage User Profiles

<https://docs.amplify.aws/gen1/react/build-a-backend/auth/manage-user-profile/>

Using the ID Token

<https://docs.aws.amazon.com/cognito/latest/developerguide/amazon-cognito-user-pools-using-the-id-token.html>

Using the access token:

<https://docs.aws.amazon.com/cognito/latest/developerguide/amazon-cognito-user-pools-using-the-access-token.html>

Configure cross-account Amazon Cognito authorizer for a REST API using the API Gateway console:
<https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-cross-account-cognito-authorizer.html>

Integrate a REST API with an Amazon Cognito user pool: (THIS IS IMPORTANT !!!)

<https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-enable-cognito-user-pool.html>

Create an Amazon Cognito user pool for a REST API:

<https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-create-cognito-user-pool.html>

IMPORTANT - how-to-change-user-status-force-change-password

<https://stackoverflow.com/questions/40287012/how-to-change-user-status-force-change-password>

Some Youtube Videos:

- <https://www.youtube.com/watch?v=ma1FA2be8Ac>
- <https://www.youtube.com/watch?v=8a0vtkWJIA4>
- <https://www.youtube.com/watch?v=LI31QxfAgho>

Creating Cognito User Pool

Select user pool sign-in options.AWS requires username for signup (later I only select username)

The screenshot shows the AWS Cognito 'Create user pool' wizard. The top navigation bar includes the AWS logo, 'Services' dropdown, a search bar with placeholder 'Search' and keyboard shortcut '[Option+S]', and a sidebar menu with six steps: Step 1 (Configure sign-in experience), Step 2 (Configure security requirements), Step 3 (Configure sign-up experience), Step 4 (Configure message delivery), Step 5 (Integrate your app), and Step 6 (Review and create). The main content area is titled 'Configure sign-in experience' with a 'Info' link. It states: 'Your app users can sign in to your user pool with a user name and password, or sign in with a third-party identity provider.' A section titled 'Authentication providers' with a 'Info' link says: 'Configure the providers that are available to users when they sign in.' Below it, 'Provider types' are listed: 'Cognito user pool' (selected, checked) and 'Federated identity providers'. The 'Cognito user pool' description notes: 'Choose whether users will sign in to your Cognito user pool, a federated identity provider, or both. Amazon Cognito has different pricing for federated users and user pool users. Learn more about pricing' with a link. Under 'Cognito user pool sign-in options' (with an 'Info' link), it says: 'Choose the attributes in your user pool that are used to sign in. If you select only one attribute, or you select a user name and at least one other attribute, your user can sign in with all of the selected options. If you select only phone number and email, your user will be prompted to select one of the two sign-in options when they sign up.' Options include 'User name' (checked), 'Email' (checked), and 'Phone number' (unchecked). 'User name requirements' include 'Allow users to sign in with a preferred user name' (unchecked) and 'Make user name case sensitive' (unchecked). A note in a yellow box states: '⚠️ Cognito user pool sign-in options can't be changed after the user pool has been created.' At the bottom right are 'Cancel' and 'Next' buttons.

Now, it is time for security requirements configuration. Since our users passwords are all 123456 for the demos. I select custom password policy and remove all the password requirements like min number of special character etc. No multi-factor authentication is selected since in our app we don't use it. In real life, it would be nice to use it. For the user account recovery, "email only" is. Note that SMS messages are charged separately by Amazon SNS. Email messages are charged separately by Amazon SES.

Amazon Cognito > User pools > Create user pool

Step 1 Configure sign-in experience

Step 2 **Configure security requirements**

Step 3 Configure sign-up experience

Step 4 Configure message delivery

Step 5 Integrate your app

Step 6 Review and create

Configure security requirements

Configure security requirements Info

Set up a strong password requirement in addition to multi-factor authentication to protect your app users from accidentally compromising their credentials.

Password policy

Create a password policy to define the length and complexity of the passwords your users can set.

Password policy mode Info

Cognito defaults Use default password requirements.

Custom Use password requirements that you define.

Password minimum length 6 character(s)

Must be a number between 6 and 99. We strongly recommend that you require passwords to be at least 8 characters in length.

Password requirements

Contains at least 1 number

Contains at least 1 special character

Contains at least 1 uppercase letter

Contains at least 1 lowercase letter

Temporary passwords set by administrators expire in 7 day(s)

Must be a number between 0 and 365.

Multi-factor authentication

Configure secure access to your app by enforcing multi-factor authentication (MFA) during the user sign-in process. MFA settings are applied to all app clients.

MFA enforcement Info

Require MFA - Recommended Users must provide an additional authentication factor when signing in.

Optional MFA Users can sign in with a single authentication factor, and can choose to add additional authentication factors.

No MFA Users can only sign in with a single authentication factor. This is the least secure option.

User account recovery

Configure how users will recover their account when they forget their password. Recipient message and data rates apply.

Self-service account recovery Info

Enable self-service account recovery - Recommended Allow forgot-password operations in your user pool. In the hosted UI sign-in page, a "Forgot your password?" link is displayed. When this feature is not enabled, administrators reset passwords with the Cognito API.

Delivery method for user account recovery messages Info

Select how your user pool will deliver messages when users request an account recovery code. SMS messages are charged separately by Amazon SNS. Email messages are charged separately by Amazon SES. Learn more about pricing. [?](#)

Email only

SMS only

For the sign-up experience, the following configuration is set. I allow automatic send messages to verify and confirm with email. And then, I set phone_number as the required attribute. (Update, I remove the phone_number from the required attributes) Also, I add custom attribute named as "active" since our user table in the database contains a column named "active".

AWS Services Search [Option+S]

Amazon Cognito > User pools > Create user pool

Step 1 Configure sign-in experience

Step 2 Configure security requirements

Step 3 Configure sign-up experience

Step 4 Configure message delivery

Step 5 Integrate your app

Step 6 Review and create

Configure sign-up experience Info

Determine how new users will verify their identities when signing up and which attributes should be required or optional during the user sign-up flow.

Self-service sign-up Info

Choose whether new users of your app can register for an account themselves.

Enable self-registration
Display a "Sign up" link on the sign-in page in the hosted UI, and allow the use of public APIs to create new user accounts. When this feature is not enabled, federation and administrative API operations create user profiles.

ⓘ If you activate user sign-up in your user pool, anyone on the internet can sign up for an account and sign in to your apps. Don't enable self-registration in your user pool until you want to open your app to public sign-up. [Learn more](#)

Attribute verification and user account confirmation

Choose between Cognito-assisted and self-managed user attribute verification and account confirmation. Only verified attributes can be used for sign-in, account recovery, and MFA. A user account must be confirmed either by attribute verification, or user pool administrator confirmation, before a user is allowed to sign in.

Cognito-assisted verification and confirmation Info

Automatically send

Allow Cognito to automatically send messages to verify and confirm - Recommended
Amazon Cognito sends a verification message with a code that the user must enter. For new users, this will verify the attribute and confirm their account.

Don't automatically send messages
Amazon Cognito doesn't send messages to users who add or change an attribute. Update attributes and confirm users with administrative API operations or Lambda triggers.

Attributes to verify Info
Choose the user contact attribute that Cognito will send a verification message to. Recipient message and data rates apply when you use SMS.

Send SMS message, verify phone number
Verify with SMS to allow users to use their phone number for sign-in, MFA, and account recovery. SMS messages are charged separately by Amazon SNS.

Send email message, verify email address
Verify with email to allow users to use their email address for sign-in, MFA, and account recovery. Email messages are charged separately by Amazon SES.

Send SMS message if phone number is available, otherwise send email message
You must build custom code when you want to verify both email and phone numbers at user account creation.

Verifying attribute changes Info

Keep original attribute value active when an update is pending - Recommended
When you update the value of an email or phone number attribute, your user must verify the new value. Until they verify the new value, they can receive messages and sign in with the original value. If you don't turn on this feature, your user can't sign in with that attribute before they verify the new value.

Active attribute values when an update is pending Info
Choose the attributes that you want to keep active when an update to their value is pending. Your users can receive messages and sign in with the original attribute value until they verify the new value.

Phone number

Email address

The screenshot shows the AWS Cognito User Pool creation wizard, Step 3: Set User Pool Attributes. The configuration includes:

- Attribute Verification:** Selected "Send email message, verify email address".

Verify with email to allow users to use their email address for sign-in, MFA, and account recovery. Email messages are charged separately by Amazon SES.
- Active attribute values when an update is pending:** Selected "Email address".

Choose the attributes that you want to keep active when an update to their value is pending. Your users can receive messages and sign in with the original attribute value until they verify the new value.

 - Phone number
 - Email address**
 - Phone number and email address
- Required attributes:** email, phone_number.
- Custom attributes - optional:** active (Number type, Min: 0, Max: 1, Mutable: checked).

Name must be 20 characters or fewer.
Value must be 2^{1023} or less.
Value must be 2^{1023} or less.

For the message delivery configuration, I selected "send email with Cognito" although SES (Simple Email Service) is recommended. This setting can be changed.

The screenshot shows the 'Create user pool' wizard at Step 4: 'Configure message delivery'. The left sidebar lists steps from 1 to 6. The main area is titled 'Configure message delivery' with a sub-section 'Email'. It explains that Cognito uses Amazon SES and SNS for email and SMS messages. Two options are shown: 'Send email with Amazon SES - Recommended' (selected) and 'Send email with Cognito' (disabled). A note states: 'You must have configured a verified sender with [Amazon SES](#) to use the SES feature.' Below this are fields for 'SES Region' (set to 'US East (N. Virginia)'), 'FROM email address' (set to 'gelgitapp@gmail.com'), and 'REPLY-TO email address - optional' (with a placeholder 'Enter an email address'). A note about permission to send email is present. Navigation buttons 'Cancel', 'Previous', and 'Next' are at the bottom.

Now, it is time to configure the integration part.

We didn't use the Cognito Hosted UI since we have a ui. Then, I created an app client and selected not to generate a client secret.

The screenshot shows the 'Create user pool' wizard at Step 5: 'Integrate your app'. The left sidebar lists steps from 1 to 6. The main area is titled 'Integrate your app' with a sub-section 'User pool name'. It says: 'Set up app integration for your user pool with Cognito's built-in authentication and authorization flows.' A field 'User pool name' contains 'gelgitapp_userpool'. A note states: 'Your user pool name can't be changed once this user pool is created.' Below this is a section 'Hosted authentication pages' with a note: 'Choose whether to use Cognito's Hosted UI and OAuth 2.0 service endpoints in Amazon Cognito. When this feature is not enabled, use Cognito API operations to perform sign-up and sign-in.' A checkbox 'Use the Cognito Hosted UI' is unchecked. Navigation buttons 'Cancel', 'Previous', and 'Next' are at the bottom.

AWS Services cognito

Choose whether to use Cognito's Hosted UI and OAuth 2.0 server for user sign-up and sign-in flows.

Use the Cognito Hosted UI
Build hosted sign-up, sign-in, and OAuth 2.0 service endpoints in Amazon Cognito. When this feature is not enabled, use Cognito API operations to perform sign-up and sign-in.

Initial app client

Configure an app client. App clients are single-app platforms in your user pool that have permissions to call unauthenticated API operations. A user pool can have multiple app clients.

App type | [Info](#)
Select an app type and we will automatically populate common default settings. You can add additional app clients after the user pool is created.

Public client
A native, browser or mobile-device app. Cognito API requests are made from user systems that are not trusted with a client secret.

Confidential client
A server-side application that can securely store a client secret. Cognito API requests are made from a central server.

Other
A custom app. Choose your own grant, auth flow, and client-secret settings.

App client name | [Info](#)
Enter a friendly name for your app client.
gelgitapp_publicAppClient

App client names are limited to 128 characters or less. Names may only contain alphanumeric characters, spaces, and the following special characters: + = , . @ -

Client secret | [Info](#)
Choose whether your app client will have a client secret. Client secrets are used by the server-side component of an app to authorize API requests. Using a client secret can prevent a third party from impersonating your client.

Generate a client secret

Don't generate a client secret

⚠ You cannot change or remove a client secret after you allow Amazon Cognito to generate it for your app client.

► Advanced app client settings
We have populated suggested authentication flows, OAuth 2.0 grant types, and OIDC scopes based on the selections you made earlier.

► Attribute read and write permissions [Info](#)
Choose the standard and custom attributes this app can read and write. Required attributes are locked as writable. We recommend that you set immutable custom attributes as writable to allow the app client to set initial values during sign-up.

Tags (0) - *optional*
You can add tags to your user pool for cost management and access control.

No tags associated with the resource.

[Add new tag](#)
You can add up to 50 tags.

[Cancel](#) [Previous](#) **Next**

Reviewing the user pool settings

AWS Services cognito

Amazon Cognito > User pools > Create user pool

Step 1
[Configure sign-in experience](#)

Step 2
[Configure security requirements](#)

Step 3
[Configure sign-up experience](#)

Step 4
[Configure message delivery](#)

Step 5
[Integrate your app](#)

Step 6
Review and create

Review and create Info
Review your selections and when satisfied, choose Create to confirm.

Step 1: Configure sign-in experience Edit

Authentication providers

Provider types Cognito user pool	Cognito user pool sign-in options Email
Federated sign-in options -	

⚠️ Cognito user pool sign-in options can't be changed after the user pool has been created.

Step 2: Configure security requirements Edit

Password policy Info

Password minimum length 6 character(s)	Password requirements -
Temporary passwords set by administrators expire in 7 day(s)	

Multi-factor authentication Info

MFA enforcement No MFA

User account recovery Info

Self-service account recovery Enabled	Recovery message delivery method Email only
--	--

Step 3: Configure sign-up experience Edit

Self-service sign-up Info

Self-registration Enabled

AWS Services cognito X

Attribute verification and user account confirmation [Info](#)

Cognito-assisted verification and confirmation Allow Cognito to automatically send messages to verify and confirm Yes Attributes to verify Send email message, verify email address	Verifying attribute changes Keep original attribute value active when an update is pending Enabled Active attribute values when an update is pending Email address
---	--

Required attributes [Info](#)

Required attributes email

Custom attributes (1) [Info](#)

Name	Type	Min value/le...	Max value/le...	Mutable
custom:active	Number	0	1	true

Step 4: Configure message delivery [Edit](#)

Email [Info](#)

Email provider Send email with Cognito SES Region US East (N. Virginia)	FROM email address gelgitapp@gmail.com REPLY-TO email address -
--	---

Step 5: Integrate your app [Edit](#)

User pool name

User pool name gelgitapp_userpool

⚠ Your user pool name can't be changed once this user pool is created.

Main application client settings

Main application client settings

App type Public client	App client name gelgitapp_publicAppClient1
Client secret -	

Advanced app client settings

Authentication flows ALLOW_REFRESH_TOKEN_AUTH ALLOW_USER_SRP_AUTH	Advanced security configurations Enable token revocation Enable prevent user existence errors
Authentication flow session duration 3 minutes	
Refresh token expiration 30 day(s) and 0 minute(s)	
Access token expiration 0 day(s) and 60 minute(s)	
ID token expiration 0 day(s) and 60 minute(s)	

Attribute read and write permissions for this app

Choose the standard and custom attributes that this app can read and write. Required attributes are locked as writable. We recommend that you set immutable custom attributes as writable to allow the app client to set initial values during sign-up.

Attribute	▲	Read	Write	
address	<input checked="" type="checkbox"/>	Read	<input checked="" type="checkbox"/>	Write
birthdate	<input checked="" type="checkbox"/>	Read	<input checked="" type="checkbox"/>	Write
custom:active (mutable)	<input checked="" type="checkbox"/>	Read	<input checked="" type="checkbox"/>	Write
email	<input checked="" type="checkbox"/>	Read	<input checked="" type="checkbox"/>	Write
email_verified	<input checked="" type="checkbox"/>	Read	<input type="checkbox"/>	Not writable
family_name	<input checked="" type="checkbox"/>	Read	<input checked="" type="checkbox"/>	Write
gender	<input checked="" type="checkbox"/>	Read	<input checked="" type="checkbox"/>	Write
given_name	<input checked="" type="checkbox"/>	Read	<input checked="" type="checkbox"/>	Write
locale	<input checked="" type="checkbox"/>	Read	<input checked="" type="checkbox"/>	Write
middle_name	<input checked="" type="checkbox"/>	Read	<input checked="" type="checkbox"/>	Write
name	<input checked="" type="checkbox"/>	Read	<input checked="" type="checkbox"/>	Write
nickname	<input checked="" type="checkbox"/>	Read	<input checked="" type="checkbox"/>	Write
phone_number	<input checked="" type="checkbox"/>	Read	<input checked="" type="checkbox"/>	Write

In order to change user verification code to a link do the following:

"User confirmed their account with a verification link sent to their phone or email. In order to enable this option you need to go to the Amazon Cognito console, look for your userpool, then go to the Messaging tab and enable link mode inside the Verification message option. Finally you need to define the signUpVerificationMethod to link inside the Cognito option of your Auth config"

Adding custom attribute.

The screenshot shows the 'gelgitapp_userpool' user pool configuration page. In the 'Custom attributes' section, a new attribute 'customUserType' is being added. It is defined as a string type with a length limit of 10 characters, is mutable, and has a default value of 'true'. The 'Add custom attributes' button is visible at the top right of this section.

When a custom attribute is added, then the client app should have read and write permission to store/read/write.

The screenshot shows the 'gelgitapp_publicAppClient1' app client configuration page. In the 'Attribute read and write permissions' section, the 'customUserType' attribute is listed with both 'Read' and 'Write' permissions selected. A note at the top of this section states: 'When a new custom attribute is added, then the client app should have read and write permission to change it.'

Edit attribute read and write permissions Info

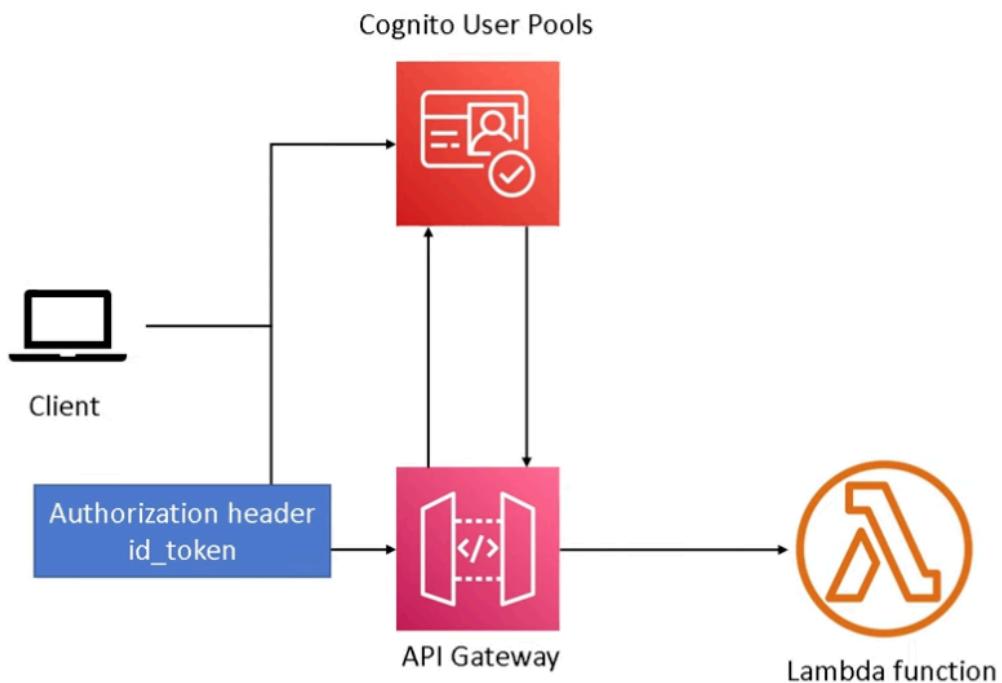
Attribute read and write permissions Info

Choose the standard and custom attributes this app can read and write. Required attributes are locked as writable. We recommend that you set immutable custom attributes as writable to allow the app client to set initial values during sign-up.

Attribute	▲	<input checked="" type="checkbox"/> Read	<input checked="" type="checkbox"/> Write
address		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
birthdate		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
custom:UserType (mutable)		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
email		<input checked="" type="checkbox"/>	<input type="checkbox"/>
email_verified		<input checked="" type="checkbox"/>	<input type="checkbox"/>
family_name		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
gender		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
given_name		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
locale		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
middle_name		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
name		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
nickname		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
phone_number		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
phone_number_verified		<input checked="" type="checkbox"/>	<input type="checkbox"/>
picture		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
preferred_username		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
profile		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
updated_at		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
website		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
zoneinfo		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

[Cancel](#)[Save changes](#)

Using Cognito for Authorization for the API Gateway



Adding authorization to an API Gateway resource

Read the article below

Integrate a REST API with an Amazon Cognito user pool: (THIS IS IMPORTANT !!!)

<https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-enable-cognito-user-pool.html>

Edit method request

Method request settings

Authorization

gelgitapp_userpool_authorizer

Authorization Scopes

Add a scope Add

Request validator

None

API key required

Operation name - optional

GetPets

▶ URL query string parameters

▶ HTTP request headers

▶ Request body

Cancel

Save

API Gateway > APIs > Resources - gelgitapp (34b7tnxk3e)

Resources

Create resource

/balance - POST - Method execution

ARN: arn:aws:execute-api:us-east-1:61742328272:3ab7m85e/POST/balance

Resource ID: jsw7

Actions Deploy API

Update documentation Delete

Method request Integration request Integration response Method response Text

Method request settings

Authorization: **gelgitapp_userpool_authorizer** (highlighted with a red arrow)

Request validator: None

SDK operation name: Generated based on method and path

Request paths (0)

Name: Coding
No request paths defined

CI/CD

ci/cd with amplify:

<https://aws.amazon.com/blogs/mobile/complete-guide-to-full-stack-ci-cd-workflows-with-aws-amplify/>

CI/CD pipelines with AWS Amplify

<https://dev.to/this-is-learning/cicd-pipelines-with-aws-amplify-5a1b>

References:

- AWS official post: Create an Amazon RDS MySQL Database
 - <https://docs.aws.amazon.com/opsworks/latest/userguide/customizing-rds-connect-create.html>
- AWS official post: Creating and connecting to a MySQL DB instance
 - https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_GettingStarted_CreatingConnecting.MySQL.html
- AWS official post: Connecting to a DB instance running the MySQL database engine
 - https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_ConnectToInstance.html
- Unofficial post:
 - <https://beabetterdev.com/2022/12/13/how-to-connect-to-an-rds-or-aurora-database-in-a-private-subnet/>
- AWS official post: Automatically connecting a Lambda function and a DB instance
 - <https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/lambda-rds-connect.html>
- AWS official post: Tutorial: Using a Lambda function to access an Amazon RDS database
 - <https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/rds-lambda-tutorial.html>
- <https://stackoverflow.com/questions/20343259/how-to-create-table-in-rds-database-on-amazon-web-service>
- Understanding data models
 - <https://docs.aws.amazon.com/apigateway/latest/developerguide/models-mappings-models.html>
- Enabling CORS in AWS API Gateway
 - <https://docs.aws.amazon.com/apigateway/latest/developerguide/how-to-cors-console.html>
- AWS Secret Manager:
 - <https://docs.aws.amazon.com/secretsmanager/latest/userguide/intro.html>
- PyMySQL
 - <https://pymysql.readthedocs.io/en/latest/modules/index.html>
 - <https://pymysql.readthedocs.io/en/latest/user/examples.html>
 - <https://pymysql.readthedocs.io/en/latest/modules/cursors.html#pymysql.cursors.Cursor.fetchone>