

## AWS Command Line Interface (CLI)

The AWS Command Line Interface (CLI) is a powerful tool provided by Amazon Web Services (AWS) that allows you to interact with AWS services from the command line.

### 1. Visit the AWS CLI Official Installation Page:

- Open your web browser, preferably Google Chrome, and search for aws cli install for windows
- Click on link <https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>
- Scroll down and click on windows
- Click on link <https://awscli.amazonaws.com/AWSCLIV2.msi> to download the file

### 2. Run the Installer:

- Once the installer is downloaded, locate the downloaded MSI file (it's typically in your Downloads folder).
- Double-click on the MSI file to run the installer.

### 3. Follow the Installation Wizard:

- The AWS CLI installation wizard will guide you through the installation process.
- Follow the on-screen instructions, and make sure to select options like adding the AWS CLI to the system PATH if prompted.

### 4. Verify the Installation:

- After the installation is complete, open a git bash terminal.
- Type the following command to verify that the AWS CLI is installed:

**aws --version**

### 5. Configure AWS CLI:

- If this is your first time using the AWS CLI on your machine, run the following command to configure it:

**aws configure**

- Enter your AWS Access Key ID, Secret Access Key, default region, and output format when prompted.

aws\_access\_key\_id = AKIA5UPT7X4MIDUE4NTX

aws\_secret\_access\_key = AK9qqLF/R8dSJV2LiGoFGM8OTFKTy61c5KPzWyOY

```
DELL@DESKTOP-TG07D38 MINGW64 ~/Desktop
$ aws configure
AWS Access Key ID [None]: AKIA5UPT7X4MBG3KCDGT
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]:
```

- Command to check the configuration

**aws sts get-caller-identity**

```
DELL@DESKTOP-TG07D38 MINGW64 ~/Desktop
$ aws sts get-caller-identity
{
  "UserId": "937351429912",
  "Account": "937351429912",
  "Arn": "arn:aws:iam::937351429912:root"
}
```

- Now you can create instances ,autoscalling groups,elastic load balancer,vpc etc using aws cli commands
- Some reference for aws cli commands

<https://www.thegeekstuff.com/2016/04/aws-ec2-cli-examples/>

<https://aws.amazon.com/cli/>

## IAM POLICIES

IAM (Identity and Access Management) policies in AWS define permissions for actions that can be performed on AWS resources. Policies are JSON documents that specify the permissions, and they are attached to IAM users, groups, or roles. Here are some key points about IAM policies:

### 1. Policy Structure:

- IAM policies are JSON documents with a specific structure.
- A policy consists of one or more statements, each with a "Effect" (Allow/Deny), "Action" (list of permissions), and "Resource" (the AWS resources the permissions apply to).

## 2. Managed Policies:

- AWS provides a set of managed policies that cover common use cases. These policies are predefined by AWS and can be attached to IAM users, groups, or roles.
- Examples include AWS managed policies like **AmazonS3ReadOnlyAccess**, **AdministratorAccess**, etc.

## 3. Inline Policies:

- In addition to managed policies, you can create inline policies directly attached to an IAM user, group, or role.
- Inline policies are defined and managed directly within the IAM entity to which they are attached.

## 4. Policy Versions:

- IAM policies support versioning. When you update a policy, a new version is created.
- You can specify a specific version of a policy when attaching it to an IAM entity.

## 5. IAM Roles:

- IAM roles are entities that define a set of permissions for making AWS service requests. Roles are not associated with a specific user or group but are assumed by trusted entities (such as AWS services, applications, or federated users).

## 6. Policy Variables:

- Policies can include variables such as **aws:username** or **aws:userid** to reference specific user information dynamically.

## 7. Conditions:

- Policies can include conditions, which allow you to specify when a policy should be applied based on certain criteria (e.g., time, source IP address).

## 8. Least Privilege Principle:

- It's a best practice to follow the principle of least privilege, granting only the permissions necessary to perform a task.

### Example IAM Policy:

Here's a simple example of an IAM policy that allows read-only access to an S3 bucket:

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": "s3:GetObject",  
      "Resource": "arn:aws:s3:::example-bucket/*"  
    }  
  ]  
}
```

In this policy:

- **Effect** is set to "Allow."
- **Action** is set to "s3:GetObject," allowing the user to retrieve objects.
- **Resource** specifies the ARN (Amazon Resource Name) of the S3 bucket and objects.

IAM policies can become more complex as you define more permissions for various resources and actions in AWS. Always review and test policies thoroughly to ensure they provide the necessary access without granting excessive privileges.

### Creating IAM (Identity and Access Management) roles in AWS

Creating IAM (Identity and Access Management) roles in AWS involves defining a set of permissions that determine what other AWS services and resources the role can access. Below are general steps to create an IAM role using the AWS Management Console :

#### 1. Sign in to the AWS Management Console:

- Navigate to the [AWS Management Console](#).
- Sign in with your AWS account credentials.

**2. Open the IAM Console:**

- In the AWS Management Console, search for and select "IAM" under the "Security, Identity, & Compliance" section.

**3. Navigate to "Roles" Section:**

- In the IAM dashboard, select "Roles" from the left navigation pane.

**4. Create a New Role:**

- Click on the "Create role" button.

**5. Select Type of Trusted Entity:**

- Choose the type of trusted entity that will assume the role. This could be an AWS service, a service in another AWS account, or an SSO identity provider.

**6. Choose Use Case or Service:**

- Depending on the selected type, you'll need to choose a use case or service that will assume the role. For example, if you are creating a role for an EC2 instance, choose "EC2".

**7. Attach Permissions Policies:**

- Attach policies that define the permissions for the role. You can attach AWS managed policies, customer-managed policies, or inline policies.

**8. Set Role Name and Tags:**

- Provide a meaningful name for the role and optionally add tags for better organization.

**9. Review and Create:**

- Review the configured settings and click "Create role" to finish the process.

Once the role is created, you can use its ARN (Amazon Resource Name) to grant permissions to other AWS resources or entities.

**Example CLI Command for Creating an IAM Role:**

Alternatively, you can use the AWS CLI to create an IAM role. Here's an example command:

bashCopy code

```
aws iam create-role --role-name MyRole --assume-role-policy-document file:///trust-policy.json
```

In this example, replace "MyRole" with your desired role name and provide a JSON file (**trust-policy.json**) containing the trust policy that defines who can assume the role.

Remember to adjust the commands and settings based on your specific requirements and use cases.

## **Creating an IAM role to access an S3 bucket in AWS**

Creating an IAM role to access an S3 bucket in AWS and attaching it to running instances

### **Step 1: Create an IAM Role**

#### **1. Sign in to the AWS Management Console:**

- Navigate to the [AWS Management Console](#).
- Sign in with your AWS account credentials.

#### **2. Open the IAM Console:**

- In the AWS Management Console, search for and select "IAM" under the "Security, Identity, & Compliance" section.

#### **3. Navigate to "Roles" Section:**

- In the IAM dashboard, select "Roles" from the left navigation pane.

#### **4. Create a New Role:**

- Click on the "Create role" button.

#### **5. Choose the Type of Trusted Entity:**

- For the type of trusted entity, select "AWS service" because you want EC2 instances to assume this role.

#### **6. Select Use Case:**

- For the use case, choose "EC2" because you want EC2 instances to use this role.

#### **7. Attach Permissions Policies:**

- Attach policies that provide the necessary permissions to access the S3 bucket. You might want to attach the "AmazonS3FullAccess" policy or create a custom policy with specific S3 permissions.

## 8. Review and Create:

- Provide a name for the role, review the settings, and click "Create role."

### Step 2: Attach the IAM Role to Running EC2 Instances

#### 1. Navigate to "Instances" in EC2 Dashboard:

- In the AWS Management Console, go to the EC2 dashboard.

#### 2. Select Running Instances:

- In the Instances view, select the EC2 instances to which you want to attach the IAM role.

#### 3. Choose Actions > Security > Modify IAM Role:

- Select "Actions" > "Security" > "Modify IAM role" from the menu.

#### 4. Choose the IAM Role:

- In the "Modify IAM role" window, choose the IAM role you created earlier.

#### 5. Save Changes:

- Save the changes.

### Step 3: Verify Access to S3

#### 1. SSH into the EC2 Instances:

- If you haven't already, SSH into the EC2 instances where you attached the IAM role.

#### 2. Install AWS CLI (if not already installed):

- Install the AWS CLI on your EC2 instance if it's not already installed.

#### 3. Test S3 Access:

- Run AWS CLI commands to test access to your S3 bucket. For example:

**aws s3 ls s3://your-bucket-name**

Ensure that the IAM role has the necessary permissions, and the EC2 instances have the correct IAM role attached. It may take a few moments for IAM role changes to propagate.

## Amazon RDS (Relational Database Service)

Amazon RDS (Relational Database Service) in AWS is a managed database service that makes it easier to set up, operate, and scale a relational database. Here are some key points about Amazon RDS:

### 1. Database Engines:

- Amazon RDS supports various database engines, including MySQL, PostgreSQL, MariaDB, Oracle, and Microsoft SQL Server.

### 2. Managed Service:

- RDS is a fully managed service, meaning AWS takes care of routine database tasks such as backups, software patching, monitoring, and scaling, allowing you to focus on your application.

### 3. Multi-AZ Deployments:

- RDS allows you to create high-availability deployments using Multi-AZ (Availability Zone) configurations for failover support.

### 4. Read Replicas:

- You can create read replicas to offload read traffic from the primary database, improving performance.

### 5. Automatic Backups:

- RDS provides automated daily backups and allows you to retain backup data for a specified period. You can also manually create snapshots.

### 6. Security:

- RDS provides security features such as encryption at rest and in transit, IAM database authentication, and the option to use Virtual Private Cloud (VPC) for network isolation.

### 7. Scalability:

- You can easily scale your database instance vertically (by changing the instance type) or horizontally (by adding read replicas).



#### 8. **Parameter Groups:**

- RDS allows you to customize database engine parameters using parameter groups to fine-tune your database configuration.

#### 9. **Database Monitoring:**

- Amazon RDS integrates with AWS CloudWatch for monitoring, allowing you to set up alarms and gain insights into database performance.

#### 10. **Maintenance Windows:**

- You can specify a maintenance window during which Amazon RDS can apply updates and perform maintenance tasks on your database instance.

#### 11. **Performance Insights:**

- RDS provides a feature called Performance Insights, which helps you analyze the performance of your database.

#### 12. **Global Database (for Aurora):**

- Amazon Aurora, a MySQL and PostgreSQL-compatible database engine, supports global databases, enabling cross-region replication for read scalability and disaster recovery.

### **Creating an Amazon RDS**

Creating an Amazon RDS (Relational Database Service) instance in AWS involves several steps. Here is a step-by-step guide to creating an RDS instance using the AWS Management Console:

#### 1. **Sign in to the AWS Management Console:**

- Navigate to the [AWS Management Console](#).
- Sign in with your AWS account credentials.

#### 2. **Open the RDS Console:**

- In the AWS Management Console, search for and select "RDS" under the "Database" section.

#### 3. **Click "Create Database":**

- In the RDS dashboard, click on the "Create database" button.

#### 4. **Select the Database Engine:**

- Choose the database engine you want to use (e.g., MySQL, PostgreSQL, etc.).

**5. Choose a Use Case:**

- Depending on your use case, select the appropriate option. For example, if you want a production database, choose "Production." As of now click on free tier.

**6. Specify Settings:**

- Fill in the necessary details, including DB instance identifier, master username, and master password.
- Choose a DB instance class, storage type, allocated storage, and other settings based on your requirements.

**7. Configure Advanced Settings:**

- For public access click on yes to access
- Configure additional settings such as VPC, subnet group, database name, port, backup retention period, and more.
- You can also configure options for high availability, monitoring, and maintenance.

**8. Configure IAM Authentication (Optional):**

- If needed, you can enable IAM database authentication for additional security.

**9. Review and Launch:**

- Review all the configured settings to ensure they are correct.
- Click "Create database" to launch the RDS instance.

**10. Monitor the Creation Process:**

- Once the creation process begins, you can monitor the progress on the RDS dashboard.
- It may take several minutes for the RDS instance to be provisioned.

**11. Access and Connect to the Database:**

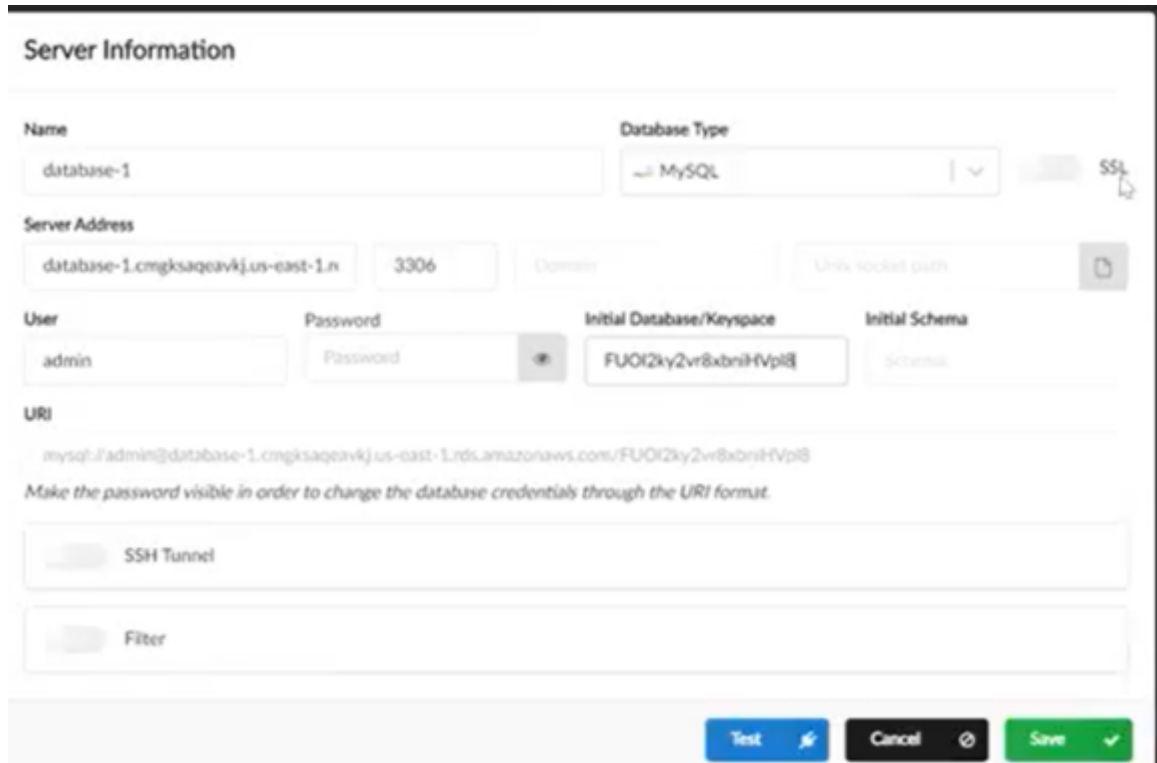
- After the RDS instance is created, you can find the endpoint in the RDS dashboard.

- Use the provided endpoint, username, and password to connect to your database.

Remember to manage your security groups, parameter groups, and other configurations according to your specific requirements. Additionally, always consider best practices for security and performance when setting up your RDS instance.

## 12. For connecting database :

- Install sqlectron: <https://sqlectron.github.io/>
- Click on Download gui
- Click on <https://github.com/sqlectron/sqlectron-gui/releases/download/v1.38.0/sqlectron-1.38.0-ia32-win.7z>
- Extract the file and access the file
- Click on sqlectron to connect database
- Click on add and give your database name n username,password..etc
- Click on test to check connection and save
- Now you can create your data using sql



The screenshot shows the 'Server Information' form in the sqlectron GUI. The form contains the following fields and controls:

- Name:** A text input field containing 'database-1'.
- Database Type:** A dropdown menu with 'MySQL' selected.
- SSL:** A toggle switch that is currently turned off.
- Server Address:** A section containing:
  - Host:** A text input field containing 'database-1.cmgksaqvkvj.us-east-1.rds.amazonaws.com'.
  - Port:** A text input field containing '3306'.
  - Domain:** A text input field that is empty.
  - Unix socket path:** A text input field that is empty.
- User:** A text input field containing 'admin'.
- Password:** A text input field containing 'Password' with a toggle switch to show/hide the password.
- Initial Database/Keyspace:** A text input field containing 'FUOI2ky2vr8xbniHVpl8'.
- Initial Schema:** A text input field containing 'schema'.
- URI:** A text area showing the generated URI: 'mysql://admin@database-1.cmgksaqvkvj.us-east-1.rds.amazonaws.com/FUOI2ky2vr8xbniHVpl8'. Below this, a note says: 'Make the password visible in order to change the database credentials through the URI format.'
- SSH Tunnel:** A toggle switch that is currently turned off.
- Filter:** A text input field that is empty.
- Buttons:** At the bottom right, there are three buttons: 'Test' (blue), 'Cancel' (black), and 'Save' (green).

## Amazon Route 53

Amazon Route 53 is a scalable and highly available Domain Name System (DNS) web service provided by AWS. It is designed to route end-user requests to endpoints globally, providing a reliable and cost-effective way to route end users to internet applications.

Here are key features and concepts related to Amazon Route 53:

### 1. DNS Management:

- Route 53 allows you to register domain names, manage DNS records, and perform domain transfers.

### 2. Domain Registration:

- You can register new domain names directly through Route 53 or transfer existing domains from other registrars.

### 3. DNS Hosting:

- Route 53 provides DNS hosting services, allowing you to manage the DNS records for your domains.

### 4. Routing Policies:

- Route 53 supports various routing policies to control how traffic is distributed to your resources. Common routing policies include Simple, Weighted, Latency-Based, Geolocation, and Failover.

### 5. Health Checks:

- You can configure health checks for your resources, and Route 53 can automatically route traffic away from unhealthy resources.

### 6. Alias Records:

- Alias records are a Route 53-specific feature that allows you to map your domain to AWS resources directly, such as an S3 bucket, CloudFront distribution, or an Elastic Load Balancer (ELB).

### 7. Traffic Flow:

- Route 53 Traffic Flow is a visual policy editor that helps you create complex routing configurations using a simple drag-and-drop interface.

### 8. Domain Name System Security Extensions (DNSSEC):

- Route 53 supports DNSSEC, which adds an additional layer of security to DNS by signing DNS data with cryptographic signatures.

#### 9. Private DNS for Amazon VPC:

- You can use Route 53 to create private hosted zones, enabling DNS resolution between resources in your Amazon Virtual Private Cloud (Amazon VPC).

#### 10. Query Logging:

- Route 53 provides query logging, allowing you to log DNS queries for your domain.

#### 11. Billing and Cost Management:

- Route 53 has a pay-as-you-go pricing model, and you pay based on the number of hosted zones, the number of queries, and the number of health checks.

❖ As route53 is costlier so we can purchase domain in godaddy website for less cost.

### GoDaddy website

GoDaddy is a popular domain registrar and web hosting company that provides domain registration services, website hosting, and additional services, including DNS management. If you have a domain registered with GoDaddy, you can manage your DNS settings through their platform.

### create domain in godaddy

Creating a domain in GoDaddy involves registering a new domain name through their platform.

#### 1. Visit the GoDaddy Website:

- Go to the [GoDaddy website](https://godaddy.com).

#### 2. Sign In or Create an Account:

- If you already have a GoDaddy account, sign in with your credentials.
- If you don't have an account, you'll need to create one. Click on "Sign In" and then choose "Create an Account."

#### 3. Search for Your Desired Domain:

- In the search bar on the GoDaddy homepage, enter the domain name you want to register.

- GoDaddy will check the availability of the domain. If your desired domain is taken, it may suggest alternative options.

#### **4. Choose Your Domain:**

- Once you find an available domain you want, click the "Select and Continue" button.

#### **5. Review and Customize Options:**

- Review the selected domain and choose additional options such as domain privacy protection, email services, and other add-ons.
- Adjust the registration period (the number of years you want to register the domain).

#### **6. Add to Cart:**

- After customizing your options, click the "Add to Cart" button.

#### **7. Continue to Cart:**

- Click the "Continue to Cart" button to proceed with the registration process.

#### **8. Review Your Order:**

- Review your order summary, ensuring that the domain name, registration period, and selected options are correct.

#### **9. Create an Account (if not already signed in):**

- If you didn't sign in earlier, you'll be prompted to create an account or sign in at this stage.

#### **10. Checkout:**

- Click the "Proceed to Checkout" button.

#### **11. Provide Contact Information:**

- Enter the required contact information for the domain registration.

#### **12. Select Payment Method:**

- Choose your preferred payment method and enter the necessary details.

#### **13. Complete the Purchase:**

- Review your order one last time and click the "Complete Purchase" or "Place Your Order" button to finalize the registration.

#### **14. Domain Registration Confirmation:**

- Once the payment is processed, you'll receive a confirmation email with details about your domain registration.

**Note:**

- Keep in mind that domain registrations are typically billed annually, and you'll need to renew your registration to maintain ownership of the domain.
- GoDaddy often provides additional services like website hosting, email services, and more. You can explore these options during the registration process or later in your account dashboard.

#### **Creating hosted zones in route53 using GoDaddy domain**

While GoDaddy is commonly used for domain registration, Amazon Route 53 is typically used for DNS hosting and management. If you have a domain registered with GoDaddy and want to use Amazon Route 53 for DNS management, you'll need to perform the following general steps:

##### **1. Sign in to Amazon Route 53:**

- Open the [Amazon Route 53 Console](#).
- Sign in with your AWS account credentials.

##### **2. Create a Hosted Zone:**

- In the Route 53 dashboard, click on "Hosted zones" in the left navigation pane.
- Click the "Create Hosted Zone" button.

##### **3. Enter Domain Information:**

- Enter your domain name (e.g., example.com) in the "Domain Name" field.
- Choose a region for your hosted zone.

##### **4. Create Hosted Zone:**

- Click the "Create hosted zone" button.

##### **5. Record Sets:**

- Inside your newly created hosted zone, you'll need to add record sets to define how your domain is resolved.

#### 6. Get Route 53 Nameservers:

- In your hosted zone details, note the nameservers provided by Route 53.

#### 7. Update Nameservers at GoDaddy:

- Log in to your GoDaddy account.
- Navigate to the DNS management section.
- Update the nameservers to the ones provided by Route 53.

#### Important Considerations:

- When updating nameservers at GoDaddy, keep in mind that DNS changes may take some time to propagate globally. It's normal for changes to take anywhere from a few minutes to 48 hours to become effective.
- Make sure to enter the correct nameserver information provided by Route 53. This information is crucial for the correct functioning of your DNS.
- It's a good practice to take note of your existing DNS settings at GoDaddy before making changes so that you can revert them if needed.

### Creating ELB and target groups

1. Launch two instances with user data

```
#!/bin/bash/
```

```
yum install httpd
```

```
systemctl start httpd
```

```
systemctl enable httpd
```

```
echo "welcome to aws tutorial from $(hostname -f)" > var/www/html/index.html
```

2. Now create target group and elastic load balancer with above two instances



### 3. Create a Record in Route 53

➤ **Go to Route 53:**

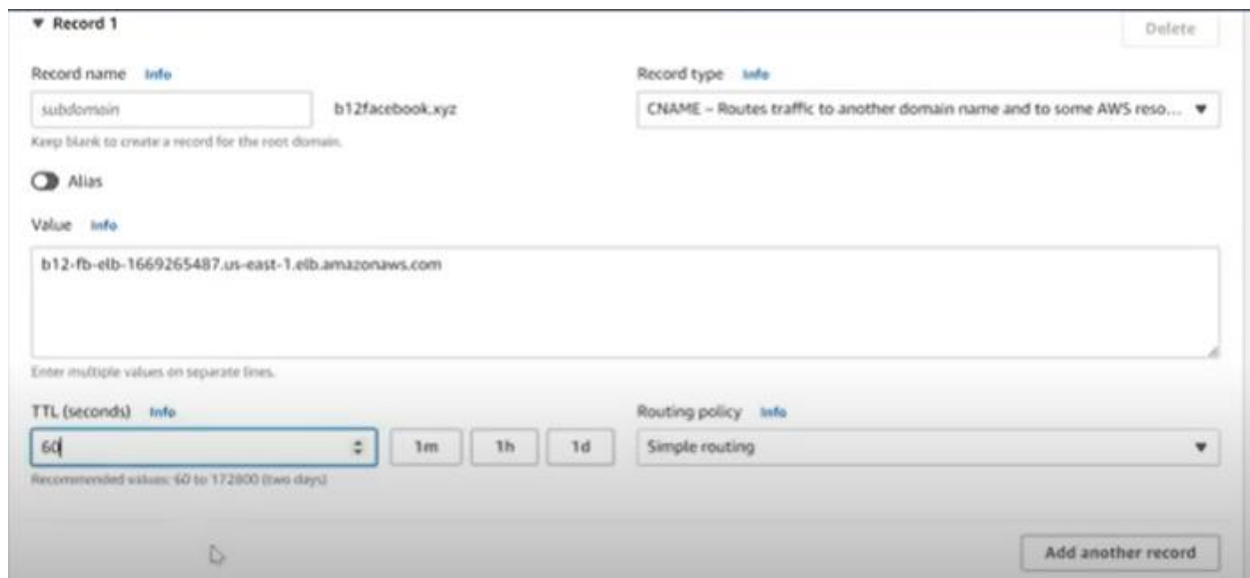
- In the AWS Management Console, go to the Route 53 Dashboard.

➤ **Create a Hosted Zone:**

- If you don't have a hosted zone for your domain, create one.

➤ **Create a Record Set:**

- Inside your hosted zone, create a record set.
- Choose the record type (e.g., A for IPv4 address, CNAME for elb), and configure it to point to your ELB's DNS name.



Record 1

Record name: subdomain b12facebook.xyz

Record type: CNAME -- Routes traffic to another domain name and to some AWS reso...

Value: b12-fb-elb-1669265487.us-east-1.elb.amazonaws.com

TTL (seconds): 60

Routing policy: Simple routing

- ❖ Now you can access your application with your own domain name
- ❖ Use your domain name in a web browser to see if the requests are being served by your instances through the ELB.
- ❖ For checking Your DNS information in gitbash command:  
nslookup: naming server lookup  
**nslookup loginb12facebook.xyz(your dns)**

```
Madhu Kiran@Madhukiran MINGW64 ~/Desktop
$ nslookup b12facebook.xyz
Non-authoritative answer:
Server: UnKnown
Address: 192.168.0.1

Name: b12facebook.xyz
Address: 18.209.24.132
```

## AWS Certificate Manager (ACM)

AWS Certificate Manager (ACM) is a service provided by Amazon Web Services that allows you to provision, manage, and deploy SSL/TLS certificates for use with AWS services and your internal resources. ACM takes care of the complexity of certificate management tasks, such as renewal, and integrates seamlessly with other AWS services. Here are key points about AWS Certificate Manager:

### 1. Certificate Types:

- ACM supports the issuance and management of public and private SSL/TLS X.509 certificates.

### 2. Integration with AWS Services:

- ACM integrates with several AWS services, including Elastic Load Balancing (ELB), CloudFront, API Gateway, and Elastic Beanstalk.

### 3. Automatic Renewal:

- ACM automatically handles the renewal of your SSL/TLS certificates, reducing the operational overhead associated with certificate management.

### 4. Managed Private Certificate Authority (CA):

- ACM Private CA is a separate service that allows you to create and manage a private CA. You can integrate ACM Private CA with ACM to issue private certificates for internal resources.

### 5. No Additional Cost for Certificates Issued by ACM:

- There is no additional cost for the SSL/TLS certificates issued by ACM. You only pay for the AWS resources you use.

### 6. Public and Private Certificates:

- ACM can issue public certificates that can be used for websites and applications accessible over the internet. Additionally, ACM can issue private certificates for internal resources within your Virtual Private Cloud (VPC).

### 7. Certificate Request Process:

- To get a certificate, you can request one directly through the ACM console or use the ACM API. ACM validates domain ownership before issuing the certificate.

### 8. Multi-Domain Certificates (SAN):

- ACM supports Subject Alternative Names (SAN), allowing you to include multiple domain names in a single certificate.

### **9. Wildcard Certificates:**

- ACM supports wildcard certificates, allowing you to secure multiple subdomains with a single certificate.

### **Example Steps to Request a Certificate in ACM:**

- 1. Navigate to ACM Console:**
  - Open the [ACM console](#).
- 2. Request a Certificate:**
  - Click on the "Request a certificate" button.
- 3. Specify Domain Names:**
  - Enter the domain names for which you want to request a certificate.
- 4. Choose Validation Method:**
  - Choose a validation method (DNS validation or email validation).
- 5. Add Tags (Optional):**
  - Optionally, add tags to your certificate.
- 6. Review and Confirm:**
  - Review your settings and click "Confirm and request."
- 7. Validation:**
  - Complete the validation process based on the method you chose.

### **Validation Record:**

- ACM will provide you with a DNS CNAME record that you need to add to the DNS configuration of the domain.

### **1. Add DNS Record in Route 53:**

- Open the [Route 53 console](#).
- Navigate to the hosted zone associated with the domain for which you are requesting the certificate.

## 2. Add CNAME Record:

- Add a new CNAME record with the information provided by ACM. The CNAME record typically has a name like **\_acme-challenge.<your-domain>** and points to a specific value provided by ACM.

## 3. Wait for DNS Propagation:

- DNS changes may take some time to propagate. You may need to wait for a few minutes to an hour for the DNS record to be recognized globally.

## 4. Verify Certificate Status in ACM:

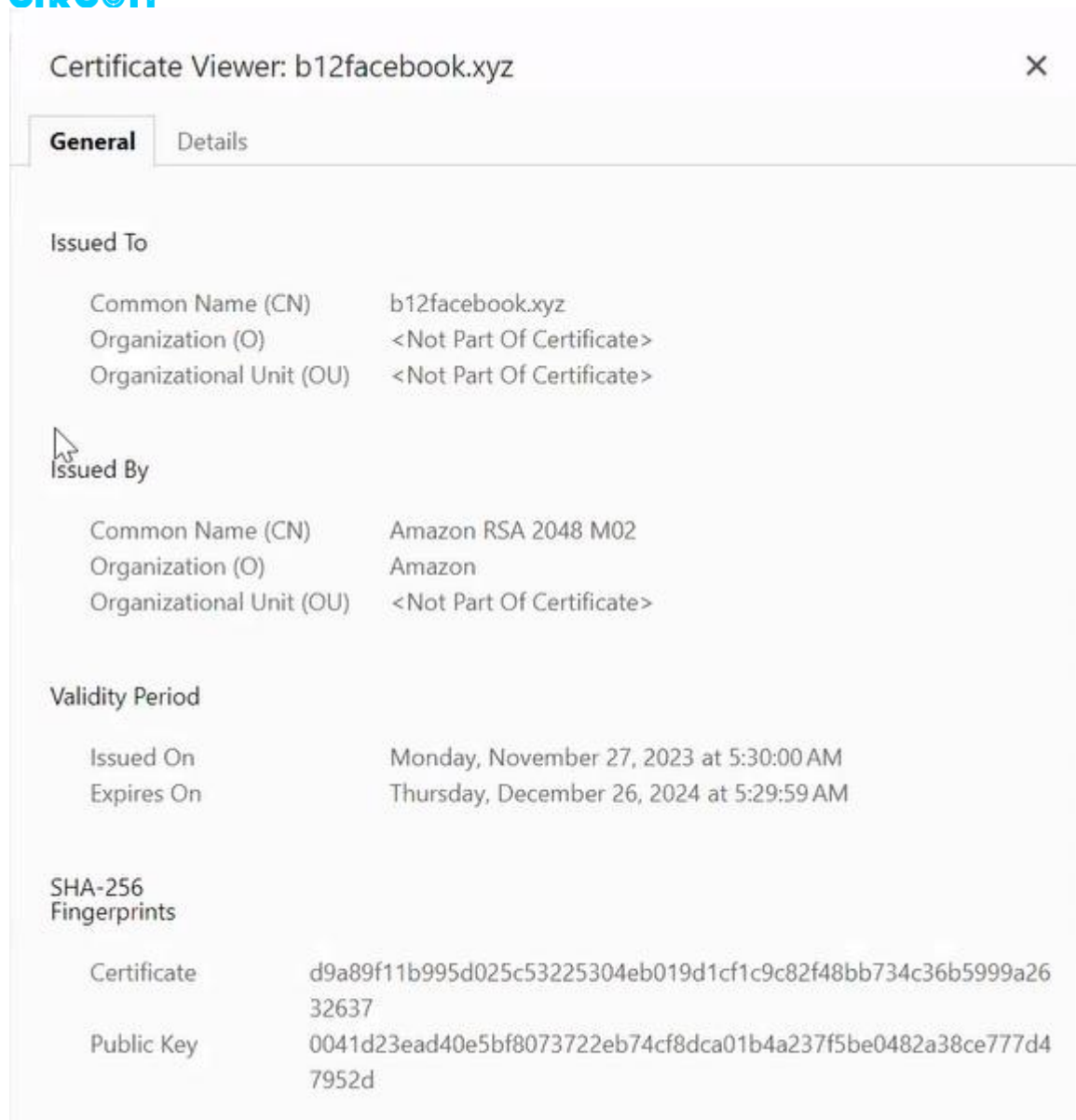
- Go back to the ACM console.
- Once the DNS record is added and propagated, ACM will automatically detect the validation and mark the certificate as "Issued."

## Important Considerations:

- **Propagation Time:** DNS changes, including the addition of new records, can take time to propagate. Be patient and allow some time for the DNS record to be recognized globally.
- **Correct Record Information:** Ensure that you enter the correct CNAME record information in Route 53. Any typos or errors can lead to the validation process failing.
- **Remove Validation Record:** After the certificate is issued, you can remove the DNS validation record from your Route 53 hosted zone.

## 8. Certificate Issued:

- Once validated, the certificate is issued and ready for use.



When you have obtained an SSL/TLS certificate from AWS Certificate Manager (ACM) or any other Certificate Authority and you want to use it with an HTTPS listener on an Elastic Load Balancer (ELB) in AWS, you need to follow these general steps:

### 1. Obtain an SSL/TLS Certificate:

- First, ensure you have obtained an SSL/TLS certificate. You can use ACM to request and manage certificates or import a certificate from another Certificate Authority.

## 2. Create an HTTPS Listener on the ELB:

- Go to the EC2 Dashboard in the AWS Management Console.
- Navigate to "Load Balancers" and select the load balancer you want to configure.
- In the "Listeners" tab, add a new listener for HTTPS (port 443).

## 3. Configure SSL/TLS for the Listener:

- In the HTTPS listener settings, configure the SSL/TLS settings:
  - **SSL Certificate:** Choose the certificate you obtained in the previous step from the drop-down list.
  - **Security Policy:** Choose an appropriate security policy based on your requirements. AWS provides predefined security policies with different encryption and protocol versions.

## 4. Adjust Security Group Settings:

- Ensure that the security group associated with your ELB allows traffic on port 443 (HTTPS). Update the security group rules if needed.

## 5. Update Instance Security Groups:

- If the ELB forwards traffic to EC2 instances, make sure that the security groups of the EC2 instances allow traffic from the ELB on the necessary ports (e.g., port 80 or 443).

## 7. Test the Configuration:

- After making the changes, test the configuration by accessing your application using the HTTPS endpoint of the ELB.