# The Essential Guide to Public SSL Certificates in AWS: A Recipe for Secure and Trusted Web Applications

## Public Certificates: Trust and Encryption in the Digital Age

In today's digital landscape, trust is paramount. Public certificates act as the digital passports for your website, instantly establishing legitimacy and security for your visitors. Imagine clicking onto a website and seeing a reassuring green padlock in your browser bar – that's the mark of a secure connection, achieved through the magic of public certificates.

This encryption shield protects sensitive information like passwords and credit card details, preventing malicious actors from eavesdropping or tampering with data. But it doesn't stop there. Public certificates also verify the true identity of websites, ensuring you're not interacting with imposters trying to steal your information. This builds trust and encourages visitors to linger, explore, and potentially convert into loyal customers.

Moreover, search engines like Google reward websites with valid certificates with higher rankings, boosting your online visibility and attracting even more organic traffic. So, don't wait – getting started with public certificates is easier than ever, thanks to Amazon Certificate Manager. With its user-friendly interface and even free options for basic use, there's no excuse to leave your website and your visitors vulnerable. Remember, public certificates are the cornerstone of a secure and trustworthy online presence, paving the way for a successful and flourishing website.

### How it works

**Focus**: Public certificates use cryptography and trusted authorities to lock down website communication, encrypting data and building trust.

* *Public-Key Cryptography*: This forms the foundation. You generate a key pair containing a public key (shared publicly) and a private key (kept secret). The public key encrypts data, only the private key can decrypt it.
* *Digital Signature and Trust*: A trusted Certificate Authority (CA) verifies your ownership of the domain and digitally signs a document containing your public key. This signature acts as a vouching mechanism, assuring others the public key is legitimate.
* *TLS Handshake*: When a browser visits your website, a complex "handshake" occurs. The browser initiates the connection by sending its public key. In response, your server sends its own public key along with a signed certificate. The browser verifies the signature (trusting the CA) and establishes a secure encrypted connection using both keys.

### Why it matters

**Focus:** Happy customer on secure website

* Public certificates create a smoother browsing experience by preventing security warnings and errors that can deter users. Additionally, they accelerate page loading times due to improved connection protocols, leading to higher user engagement and satisfaction.
* Search engines like Google prioritise secure websites, rewarding those with valid certificates with higher rankings. This translates to enhanced organic traffic and wider online visibility, putting your website at the forefront of potential customers' searches.
* For e-commerce businesses or websites handling sensitive information, public certificates are not just an option, they're a necessity. They build trust, provide compliance with industry regulations, and ensure a secure environment for financial transactions, fostering customer confidence and driving business success.

## Hands-on Workshop: Minting Your First Certificate with ACM

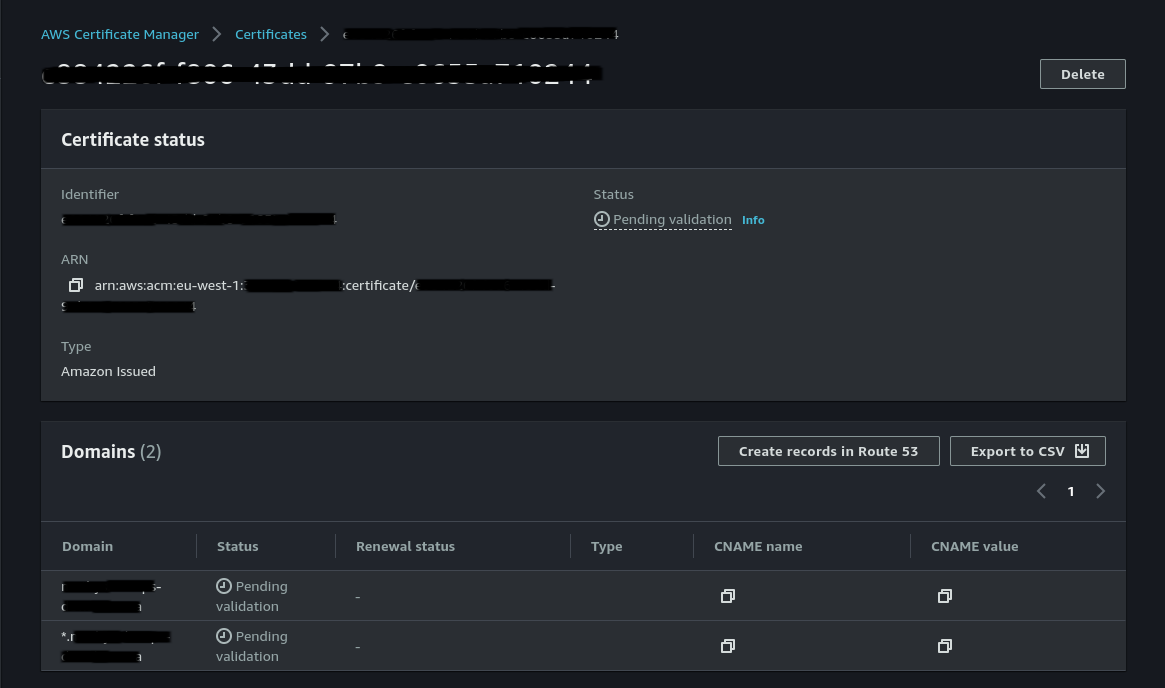
### Prerequisite

Securing your website with a public certificate starts with a few key ingredients. First, you'll need ownership of the domain you want to protect, demonstrated through either your ability to modify DNS records (for DNS validation) or access to an associated email address (for email validation). Additionally, ensure you have the necessary permissions within your domain management service to make these changes. Finally, an active AWS account with access to Amazon Certificate Manager (ACM) is your launchpad for minting your first digital security shield. With these prerequisites in place, you're ready to embark on safeguarding your online space and building trust with your visitors.

### Steps

1. Open your AWS Certificate Manager(ACM) console at <https://console.aws.amazon.com/acm/home>
2. Make sure you are in the desired region
   1. **Note:** for global services like CloudFront you need to be in US East(N.Virginia)
3. Choose the **Request certificate**, on the navigation bar.
4. For Certificate type, choose **Request a public certificate**.
5. Click **Next**.
6. Enter your domain name
   1. **Note:** Choosing the right domain names for your public certificate is crucial. You can secure a specific subdomain like "[https://www.example.com](https://www.example.com/)" or protect your entire domain with "example.com". Use a wildcard certificate with an asterisk (\**) at the beginning, like "\**.example.com", to shield all subdomains under that umbrella, including "corp.example.com" and "images.example.com". Just remember, the wildcard only works at the very beginning of the domain name. Whichever path you choose, the selected domain(s) will be clearly displayed in the certificate for transparency and security assurance.
7. (Optional) you can add your apex and bare domain by clicking **Add another name to this certificate** like example.com and \*.example.com
8. Choose a **Validation method**.
9. Choose **Request**.

Validation via DNS record:

1. You should see a message starting with Successfully requested certificate with ID. Choose View certificate.
2. Your certificate will be in a Pending state.
3. Using DNS Validation you will be met by this screen and Route 53 records can be created by clicking on **Create Route53 records**.
4. After up to 30 minutes your certificate should update to **Issued  
   Note:** After 72 hours the request will time out if not validated.
5. You are good to go! The certificate is ready to use.

## Code Your Way to Security: Automating Certificate Deployment

### Prerequisite

To deploy CloudFormation scripts you will require an account with a setup VPC with public and private subnets. Your two created certificates as mentioned in the previous steps for your CloudFront and Application Load Balancer. You will also need the following access to deploy the script:  
Read Access:

* VPC: Necessary to describe the existing VPC (if applicable) used by the template.
* EC2: Needed to describe existing security groups and instances before modification or deletion.
* ELBv2: Permission to describe existing load balancers, listeners, and target groups.
* CloudFront: Required to describe existing CloudFront distributions.
* Route 53: Allows describing the hosted zone (if it already exists) and existing record sets before modification or deletion.

Write Access:

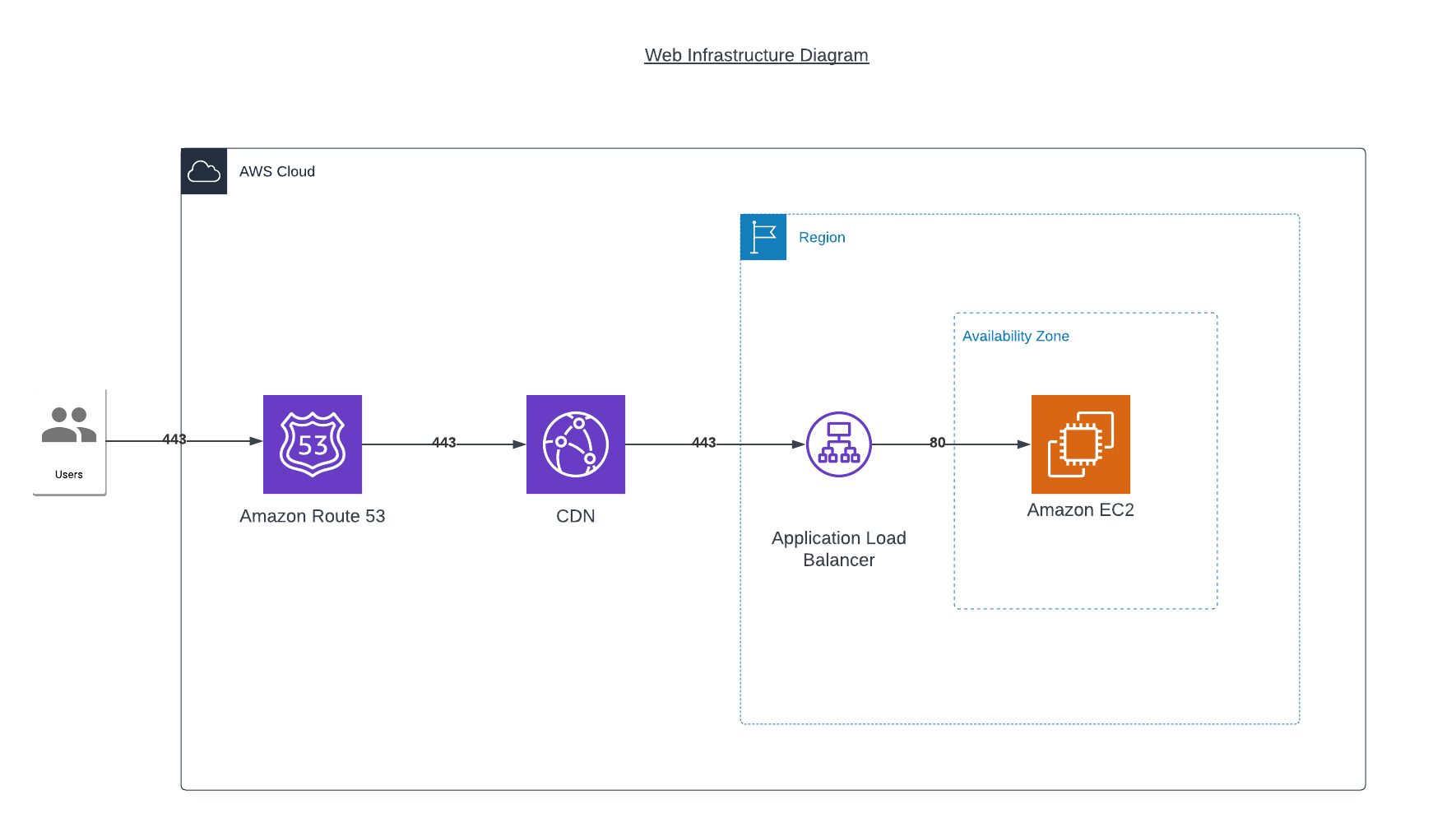
* EC2: Required to create and manage security groups, launch and manage EC2 instances, and attach/detach volumes.
* EBS: Needed to create and manage volumes used by the EC2 instances.
* ELBv2: Permissions to create and manage load balancers, listeners, and target groups.
* CloudFront: Allows creating and managing CloudFront distributions.
* Route 53: Permission to create a hosted zone (if it doesn't exist) and create/update record sets.
* CloudFormation: Grants permission to describe and delete the CloudFormation stack and its associated resources.

**Note:** Deploying this infrastructure will incur some costs, do remember to clean up afterwards.

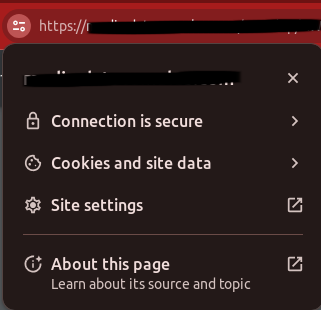
### Steps

A YAML file defining the Infrastructure as Code (IaC) is provided:

* **WebInfrastructure.yml**: defines all the resources required for this certificate test. It sets up a basic web infrastructure that we will use to secure the certificate we just created. On inspection of this file, take notice of the following parameters:
  + “*CDNCertificateARN*”: (CloudFront certificate in the N.Virginia region) used for the CloudFront resource called “CloudFrontDistribution” notice how this parameter is added to this resource and field called “AcmCertificateArn” this is what allows us to connect a certificate to a CloudFront distribution via CloudFormation(code)
  + “*ALBCertificateARN*”: connects an SSL/TLS certificate to an Application Load Balancer (ALB) in CloudFormation. This variable is used within the HttpsListener resource, specifically in the CertificateArn field. Remember, the certificate must be deployed in the same region as the ALB itself.

  
A user's encrypted request (port 443) navigates through Route 53, the DNS service, which directs them to the nearest CloudFront edge location. CloudFront serves cached content directly, but forwards uncached requests to the Application Load Balancer (ALB) within the region. The ALB acts as a secure gateway, utilising a public certificate to establish a trusted connection with the user and decrypt the incoming request. It then processes the decrypted information based on configured rules and forwards it to a healthy EC2 instance. The EC2 instance processes the request and sends its response back, which the ALB re-encrypts before returning it to the user, maintaining security throughout the journey.

Now lets deploy this infrastructure:

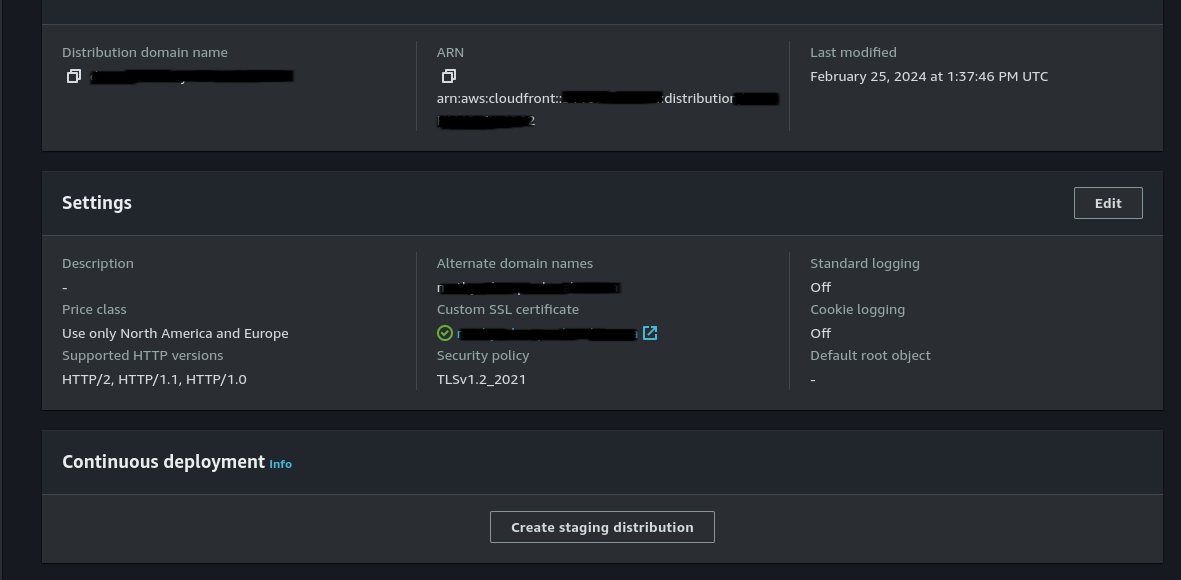
1. Log in to the AWS console, navigate to CloudFormation, and ensure "CloudFormation FullAccess" permission.
2. Click "Create stack", choose “With new resources” then “Template is ready” and "Upload a template file", select the provided template (WebInfrastructure.yml) and provide a Stack Name(e.g. web-infra-stack).
3. Fill Parameters: Review required parameters, enter values, and set optional parameters if desired. Click Next.
4. Optional: On this page it is suggested to add a tag that will tag all the resources. Suggestions are Owner with your name as the value or System with the system name as value.
5. Review & Deploy: Review all settings, click "Create stack" to deploy, and monitor progress in the "Stacks" section.
6. Once completed and all parameters entered correctly you should be able to hit your domain and see a page with the message “It works!”
7. Notice how the url directs to https and has a padlock. Further investigation can be done by clicking on it and clicking on the certificate and seeing more information.
8. You have a basic secure https:// website!

## AWS Resource Management on Console: Updating Your Certificates on the Console

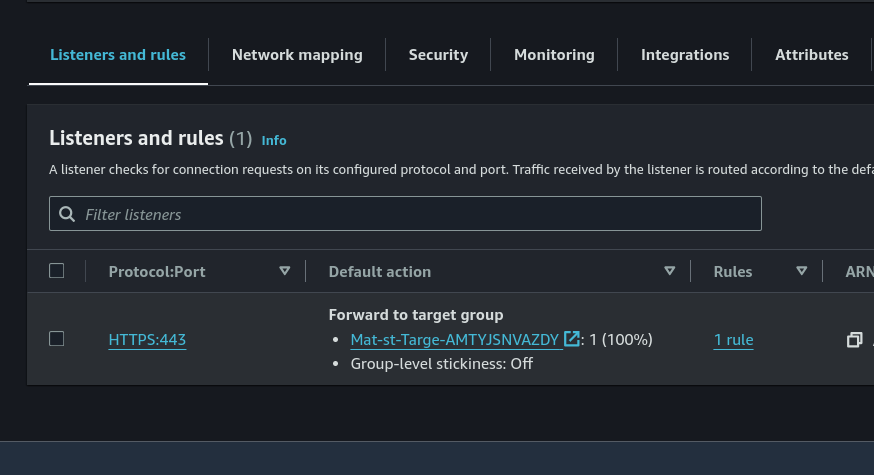
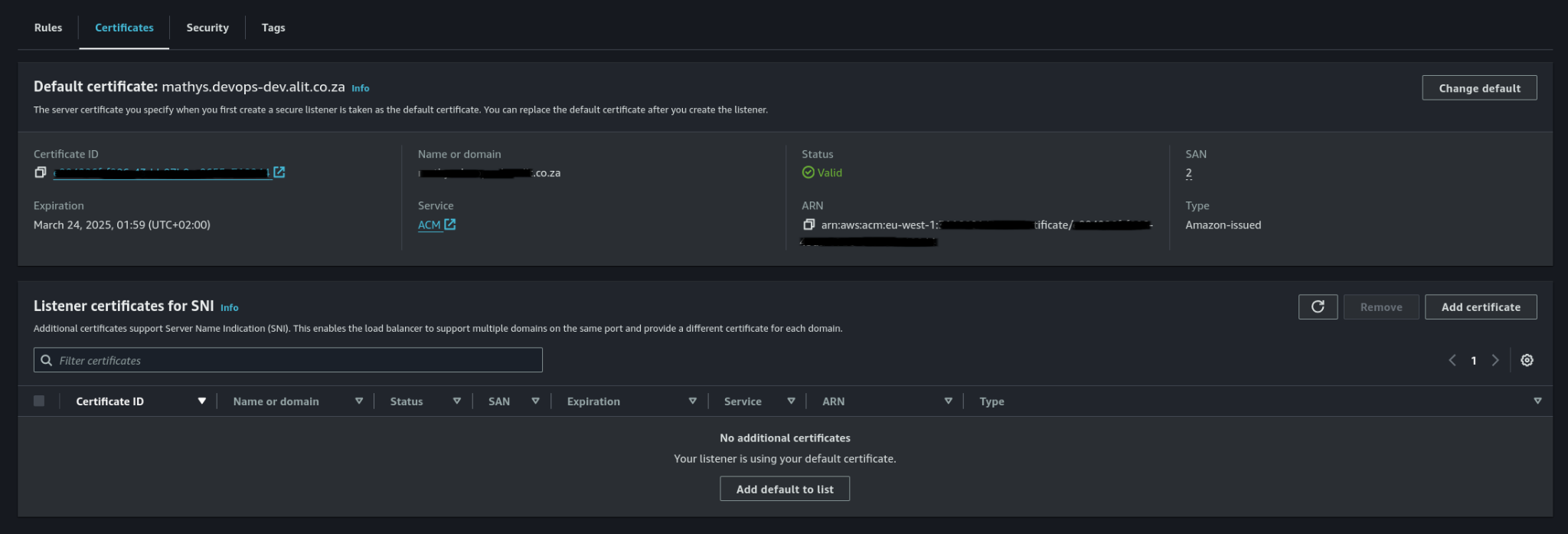
### Prerequisite

### To update CloudFront and Application Load Balancer certificates on the console, you will need to deploy the required resources, CloudFront and Loadbalancer in the CloudFormation script mentioned in the previous steps(If you are not using your own resources). You also need both the new certificate and IAM permissions ready. Steps

Updating your certificate on CloudFront:

1. Navigate to Distribution: In the CloudFront console, find the distribution you want to update the certificate for.
2. Edit Distribution Settings: Select the "Distribution Settings" tab and click "Edit" for the general settings.
3. Choose a Certificate: Under "SSL Certificate," choose "Custom SSL Certificate" and select the new certificate you want to use.
4. Specify Alternate Domain Names (Optional): If using custom domain names, add them under "Alternate Domain Names (CNAMEs)."
5. Review and Save: Double-check your changes, then click "Yes, Edit" to confirm and save the updated configuration.
6. Wait for Propagation: CloudFront will propagate the changes to its edge locations. This can take 5-15 minutes.
7. Test and Monitor: Once propagation is complete, test your website and monitor for any issues with the new certificate.

Update or add certificates to your Loadbalancer:

1. Open the Amazon EC2 console and navigate to Load Balancers. Select your specific ALB.
2. Go to the "Listeners and rules" tab and click on the HTTPS listener you want to update (HTTPS:443).
3. Under "Certificates", click "Add certificate" to upload a new certificate or select an existing one to "Edit".
4. Upload the new certificate file and private key (if adding), and configure Server Name Indication (SNI) if needed.
5. Review the changes and click "Save changes".
6. Wait for a short propagation period (usually under 5 minutes).
7. Test your website or application to ensure the update was successful.

**Tips:**A public certificate issued by ACM is eligible for automatic renewal if it meets the following criteria:

* Associated with an AWS Service: The certificate must be currently associated with an active AWS service like Elastic Load Balancing or CloudFront.
* Exported or Service-Associated: The certificate must have been either:
  + Exported after being issued or renewed by ACM.
  + Associated with an AWS service after being issued by ACM.
* Private Certificate: The certificate must be a private certificate, meaning it was issued through the acm\_request\_certificate API call or the management console/CLI.

The following certificates are not eligible for automatic renewal:

* Imported Certificates: Certificates imported into ACM are not eligible.
* Private CA Certificates: Certificates issued through the AWS Private CA IssueCertificate API are not eligible.
* Expired Certificates: Certificates that have already expired are not eligible for renewal.

**Important**:   
Once done using/testing the resources do remember to clean up as they might incur costs:

* Delete your CloudFormation stack, this will automatically delete all the resources that the stack created.
* Also remember, you need to ensure you have deleted any public certificates associated with AWS Certificate Manager (ACM) and have located both the CloudFront certificate stored in the N.Virginia region and the Application LoadBalancer (ALB) certificate in the region where you deployed your CloudFormation stack.