AWS-BASED PERSONAL PORTFOLIO WEBSITE DEPLOYMENT

"A Comprehensive Guide to Deploying a Static Website Using AWS Services."

By: Sajan Alex | Date: December 19, 2023 | Version: 1.0

Table of Contents

1. Introduction

The deployment of a personal portfolio website represents a significant milestone, combining elements of design, development, and infrastructure management. This project aims to showcase a comprehensive understanding of hosting static websites using AWS Cloud services, specifically Amazon EC2, S3, and CloudFront. As the digital landscape evolves, individuals and businesses are increasingly leveraging cloud solutions for enhanced scalability, security, and cost efficiency.

1.1 Context and Motivation

In a non-cloud environment, hosting a static website often involves overcoming challenges related to infrastructure setup, scalability, high availability, cost optimization, and security. This project delves into addressing these challenges by leveraging the power of AWS services. The motivation stems from the desire to simplify infrastructure management, provide scalability and redundancy, optimize costs, and ensure robust security—essential components for a successful static website deployment.

By embracing AWS Cloud services, we explore the potential of Amazon EC2 for hosting, Amazon S3 for efficient content storage, and Amazon CloudFront for global content delivery. These services collectively offer a seamless solution to construct and maintain a static website, providing not only a cost-efficient and scalable infrastructure but also global content delivery, streamlined deployment processes, and enhanced security measures.

This project serves as a demonstration of practical skills in deploying a personal portfolio website on AWS, contributing to a robust user experience while allowing for focused attention on core objectives. The following sections provide a detailed account of the project's objectives, methodologies, challenges encountered, solutions implemented, and insights gained throughout the development and deployment process.

2. Objectives

This project is driven by a set of clear objectives aimed at achieving a successful deployment of a personal portfolio website using AWS Cloud services. The focus encompasses understanding the nuances of static websites, their benefits, and various use cases. The primary objectives are:

1. Understand Static Websites

Gain a comprehensive understanding of static websites, exploring their benefits and diverse use cases in the context of personal portfolio websites. This involves delving into the core principles of static web content and its relevance in today's digital landscape.

2. Familiarize with AWS Cloud Services

Develop familiarity with key AWS Cloud services, including Amazon EC2, Amazon S3, and Amazon CloudFront. Acquire the knowledge required to leverage these services effectively for hosting, storage, and content delivery, respectively.

3. Set Up and Configure EC2 Instance

Create and configure an Amazon EC2 instance tailored to the specifications necessary for hosting the personal portfolio website. This involves selecting the appropriate instance type, setting up security groups, and installing and configuring web server software.

4. Manage Website Content with S3

Establish an Amazon S3 bucket for efficient storage and hosting of website assets. Upload static website files, such as HTML, CSS, JavaScript, and images, to the S3 bucket, and configure proper permissions and access controls.

5. Configure CloudFront for Content Delivery

Utilize Amazon CloudFront to configure efficient content delivery from the S3 bucket. Explore caching settings, SSL/TLS configurations, and other options to optimize content delivery and enhance the website's performance.

6. DNS Configuration with Route 53

Set up Domain Name System (DNS) records using Amazon Route 53 to route traffic to the personal portfolio website through Amazon CloudFront. Ensure proper domain resolution for a seamless user experience.

7. Test and Troubleshoot

Thoroughly test the functionality of the deployed website and troubleshoot any issues that may arise. Validate proper content delivery through CloudFront and ensure the website's responsiveness across different devices and browsers.

8. Implement Security Measures

Implement robust security measures and access controls for Amazon EC2, S3, and CloudFront. This includes enabling encryption, configuring access policies, and ensuring secure communication channels throughout the infrastructure.

9. Document the Project

Create a comprehensive document detailing the project scope, methodologies, steps taken, challenges faced, and solutions implemented. This documentation will serve as a valuable resource for future reference and knowledge sharing.

10. Deploy and Optimize

Deploy the static website on the configured AWS infrastructure, following best practices. Implement strategies for cost optimization by selecting appropriate instance types, storage options, and caching settings, contributing to an efficient and cost-effective hosting solution.

3. Technologies Used

The development and deployment of the personal portfolio website were facilitated through the utilization of a set of modern and efficient technologies. The tools and platforms employed in this project include:

1. Visual Studio Code

- **Description:** Visual Studio Code, a lightweight yet powerful code editor, served as the primary Integrated Development Environment (IDE) for coding, debugging, and version control.
- **Purpose:** Visual Studio Code provided a versatile and user-friendly environment for efficient code development, offering essential features such as syntax highlighting, debugging support, and seamless integration with version control systems.

2. AWS Cloud Services

- **Amazon EC2:**
  - **Description:** Amazon Elastic Compute Cloud (EC2) offered scalable compute capacity in the cloud, enabling the creation and configuration of virtual servers to host the personal portfolio website.
  - **Purpose:** EC2 was pivotal in providing the infrastructure needed for web hosting, allowing for flexibility in instance types and configurations.
- **Amazon S3:**
  - **Description:** Amazon Simple Storage Service (S3) functioned as scalable object storage, housing and delivering static website assets seamlessly.
  - **Purpose:** S3 was employed to store HTML, CSS, JavaScript files, images, and other web assets, ensuring efficient content hosting and retrieval.
- **Amazon CloudFront:**
  - **Description:** Amazon CloudFront, a Content Delivery Network (CDN) service, distributed website content globally, enhancing performance and reducing latency.
  - **Purpose:** CloudFront was configured to serve as the front-end for content delivery, ensuring quick and reliable access to the personal portfolio website.
- **Amazon Route 53:**
  - **Description:** Amazon Route 53, a scalable Domain Name System (DNS) web service, provided domain registration and routing capabilities.

    - **Purpose:** Route 53 facilitated the configuration of DNS records, ensuring proper domain resolution and directing traffic to the CloudFront distribution.

3. Git and GitHub

- **Git:**
    - **Description:** Git, a distributed version control system, facilitated collaborative development and version management.
    - **Purpose:** Git was instrumental in tracking changes, managing branches, and ensuring a smooth and collaborative development process.
- **GitHub:**
    - **Description:** GitHub, a web-based platform for version control and collaboration, served as a central repository for storing and managing the project's source code.
    - **Purpose:** GitHub enabled seamless collaboration, code reviews, and version control, promoting a structured and organized development workflow.

This combination of tools and services empowered the development and deployment of a robust and scalable personal portfolio website on the AWS Cloud platform. The technologies used were chosen for their efficiency, scalability, and collaborative features, contributing to a successful and streamlined project execution.

4. Project Scope

The aim of this project is to design and develop a personal portfolio website to showcase skills, projects, and experience. The website will serve as a professional online presence, allowing visitors to learn more about the individual's background, skills, and accomplishments.

4.1 Objectives

1. **Create an Informative Homepage:**
    - Design a visually appealing homepage that introduces the individual.
    - Include a professional profile picture and a brief tagline.
2. **About Me Section:**
    - Provide a detailed "About Me" section highlighting the individual's background, education, and career goals.
    - Optionally, include information about skills, interests, and personal achievements.
3. **Skills Showcase:**
    - Create a dedicated section to showcase skills and competencies.
    - Display skills using a visually appealing format.
4. **Projects Display:**
    - Showcase notable projects in a visually engaging manner.
    - Include project descriptions, technologies used, and links to project repositories.
5. **Resume Section:**
    - Display a downloadable resume or a preview with key information.
    - Include an option for users to download the complete resume.
6. **Education Information:**
    - Provide details about educational background.
    - Include information such as institutions attended, degrees earned, and graduation years.
7. **Contact Information:**
    - Include a contact section with relevant contact information.

- Optionally, include links to social media profiles.

8. **Navigation:**
    - Implement a clear and user-friendly navigation menu.
    - Ensure smooth navigation between different sections of the website.
9. **Responsive Design:**
    - Ensure the website is responsive and accessible on various devices (desktops, tablets, and mobile phones).
10. **Styling and Aesthetics:**
    - Design the website with a professional and modern aesthetic.
    - Use a consistent colour scheme and typography.

4.2 Steps Taken

1. **Planning and Research:**
    - Defined the project scope and objectives.
    - Researched best practices for personal portfolio websites.
    - Gathered necessary content such as personal information, project details, and resume.
2. **Wireframing and Design:**
    - Created wireframes to plan the layout and structure of the website.
    - Designed the website layout, choosing a clean and modern design.
3. **Setting Up Development Environment:**
    - Installed necessary tools such as a text editor and version control system (e.g., Git).
    - Set up a local development environment.
4. **HTML and CSS Development:**
    - Developed the HTML structure of the website.
    - Styled the website using CSS to achieve the desired visual design.
5. **JavaScript for Interactivity:**
    - Implemented JavaScript for interactive elements, such as smooth scrolling and dynamic content loading.

6. **Responsive Design:**
    - Ensured the website is responsive using media queries.
    - Tested the website on various devices to ensure optimal user experience.
7. **Project Showcase Section:**
    - Implemented a section to showcase projects with images, title and descriptions.
    - Used CSS to create a visually appealing layout for project cards.
8. **Resume Section:**
    - Created a resume section with downloadable options.
    - Included relevant information in a concise and organized format.
9. **Education Section:**
    - Added an education section with details about academic background.
10. **Contact Form:**
    - Implemented a contact form or provided contact details for user inquiries.
11. **Testing:**
    - Conducted thorough testing of the website for functionality and responsiveness.
    - Fixed any bugs or issues discovered during testing.
12. **Deployment:**
    - Chose a hosting solution (e.g., AWS, Netlify, GitHub Pages).
    - Deployed the website to make it accessible online.
13. **Documentation:**
    - Prepared comprehensive documentation detailing the project scope, objectives, and steps taken.
    - Included instructions for maintaining and updating the website.

5. Architecture Overview

The architecture of the personal portfolio website leverages AWS cloud services to ensure scalability, high availability, and efficient content delivery. The primary components include:

1. **EC2 Instance:** A configured EC2 instance serves as the web hosting environment, providing the foundation for website deployment.
2. **S3 Bucket:** An S3 bucket stores and manages website assets, ensuring secure and efficient content hosting.
3. **CloudFront Distribution:** CloudFront optimizes content delivery globally, enhancing performance and providing high availability.
4. **Route 53:** DNS settings are configured using Route 53 for seamless domain resolution and routing traffic to the CloudFront distribution.

The architecture prioritizes security, scalability, and cost efficiency, offering a robust foundation for the personal portfolio website's optimal functionality and user experience.

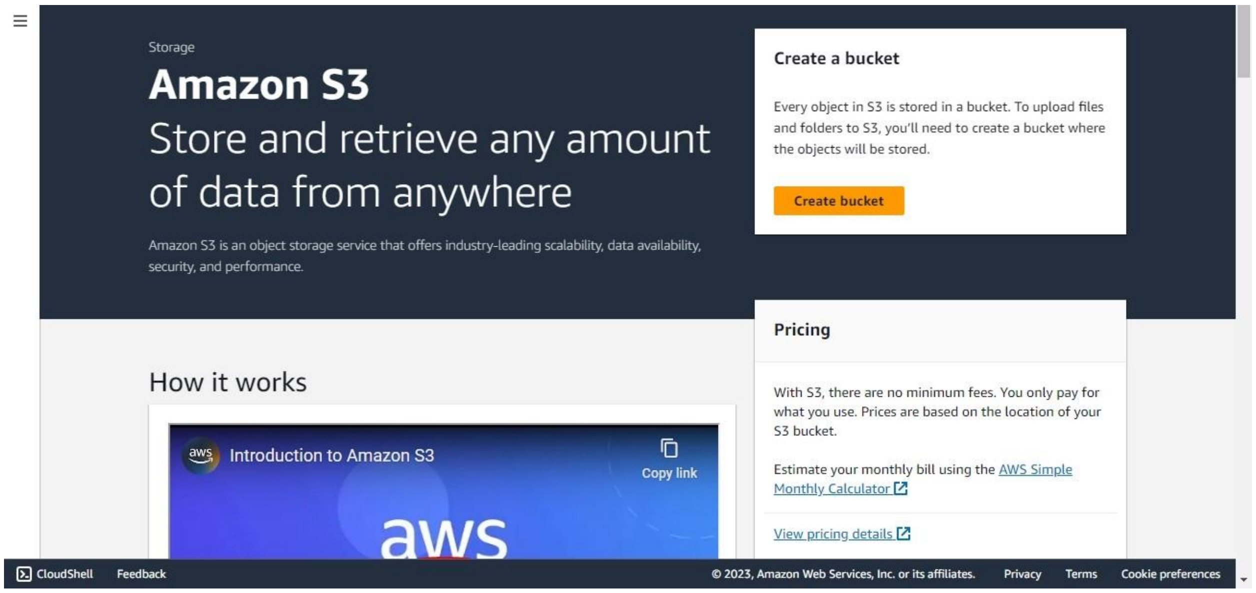
6. Implementation Steps

6.1 Research and Requirements Gathering

- Conducted thorough research to understand static website benefits and use cases.
- Gathered project requirements to inform decision-making throughout the implementation.

6.2 S3 Bucket Configuration

1. Navigate to S3:
   - Go to the S3 service from the AWS Console.
   - Click on the "Create bucket" button.



Storage

Amazon S3

Store and retrieve any amount of data from anywhere

Amazon S3 is an object storage service that offers industry-leading scalability, data availability, security, and performance.

Create a bucket

Every object in S3 is stored in a bucket. To upload files and folders to S3, you'll need to create a bucket where the objects will be stored.

Create bucket

Pricing

With S3, there are no minimum fees. You only pay for what you use. Prices are based on the location of your S3 bucket.

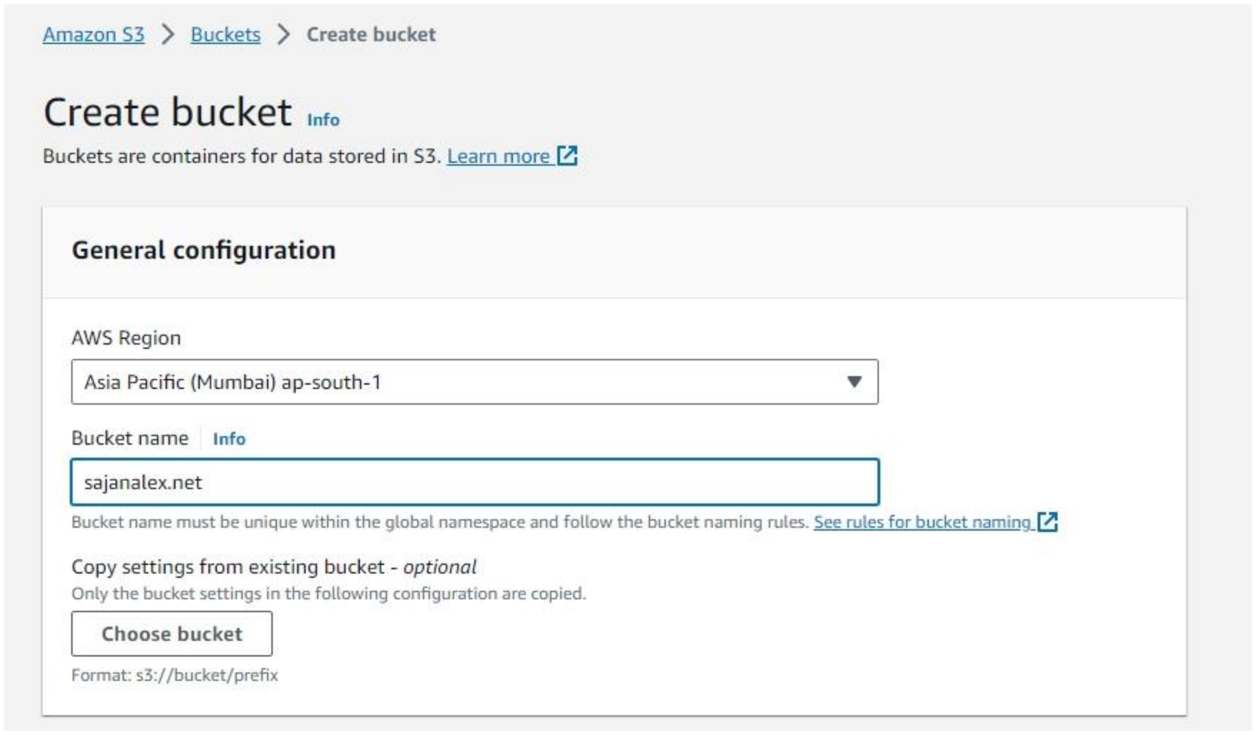
Estimate your monthly bill using the AWS Simple Monthly Calculator

View pricing details

How it works

Introduction to Amazon S3
Copy link
aws
CloudShell
Feedback
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

2. Create a New Bucket:
   - Choose a globally unique name for your bucket (**sajanalex.net**).
   - Select a region for your bucket (**ap-south-1**).



Create bucket Info

Buckets are containers for data stored in S3. Learn more

General configuration

AWS Region

Asia Pacific (Mumbai) ap-south-1

Bucket name Info

sajanalex.net

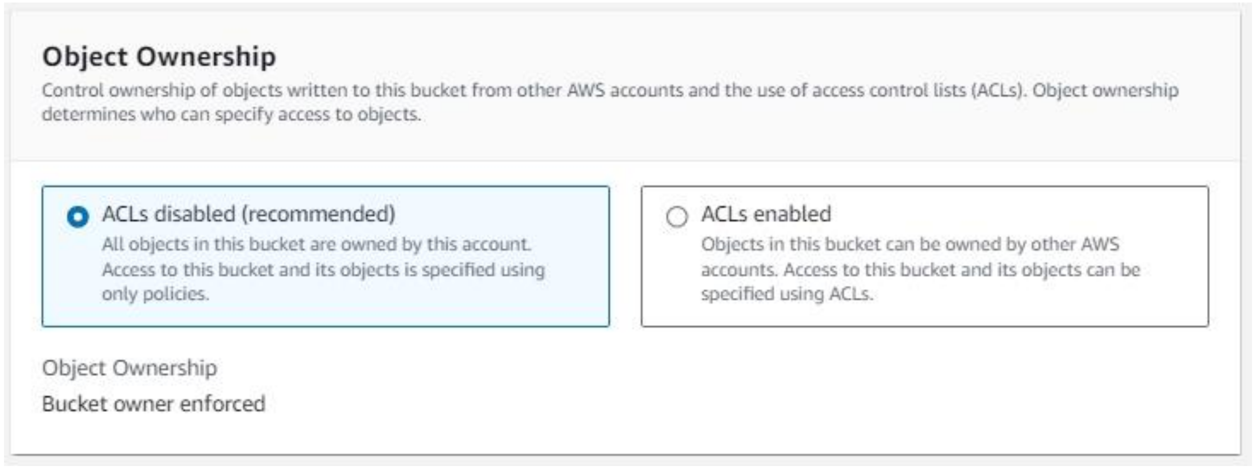
Bucket name must be unique within the global namespace and follow the bucket naming rules. See rules for bucket naming

Copy settings from existing bucket - *optional*

Only the bucket settings in the following configuration are copied.

Format: s3://bucket/prefix

- Proceed with **ACLs disabled**.



Object Ownership

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

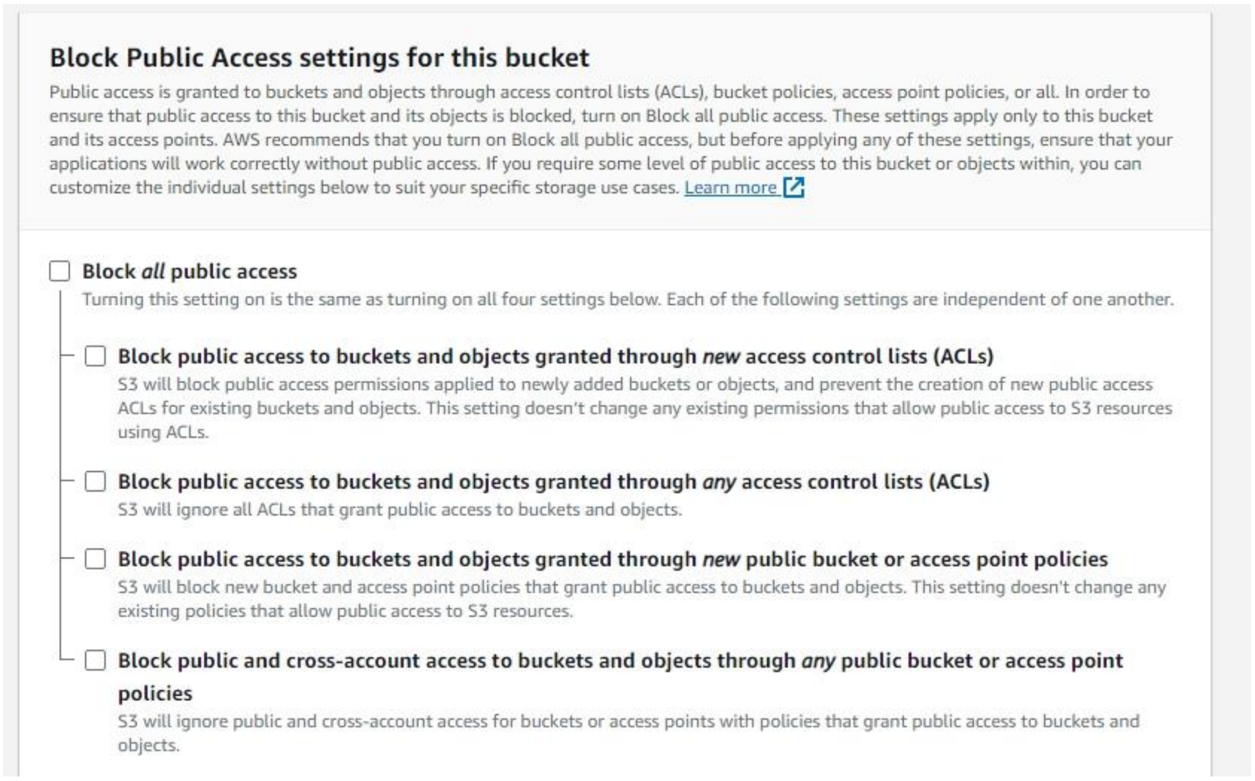
[✓ ACLs disabled (recommended)] All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

ACLs enabled: Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Object Ownership

Bucket owner enforced

- Uncheck **Block All Public Access** box to grant public access to the bucket objects.



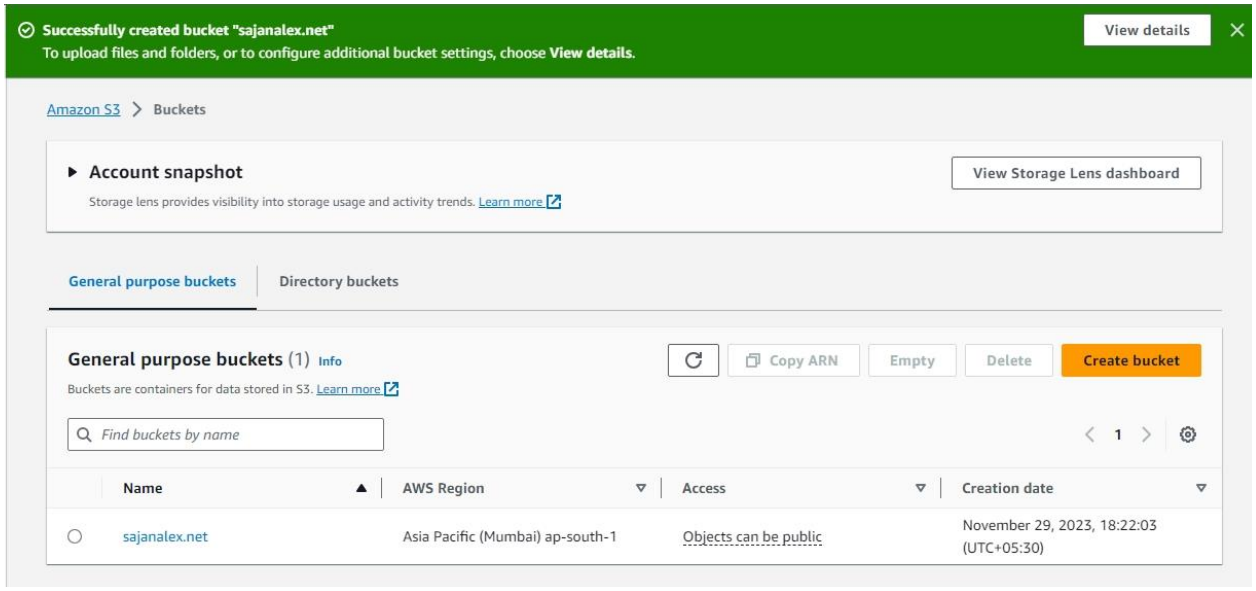
Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more

☐ **Block *all* public access**
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

- ☐ **Block public access to buckets and objects granted through *new* access control lists (ACLs)**
  S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
- ☐ **Block public access to buckets and objects granted through *any* access control lists (ACLs)**
  S3 will ignore all ACLs that grant public access to buckets and objects.
- ☐ **Block public access to buckets and objects granted through *new* public bucket or access point policies**
  S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
- ☐ **Block public and cross-account access to buckets and objects through *any* public bucket or access point policies**
  S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

- Click **Create Bucket**



Successfully created bucket "sajanalex.net"
To upload files and folders, or to configure additional bucket settings, choose **View details**.

Storage lens provides visibility into storage usage and activity trends. Learn more

General purpose buckets (1) Info

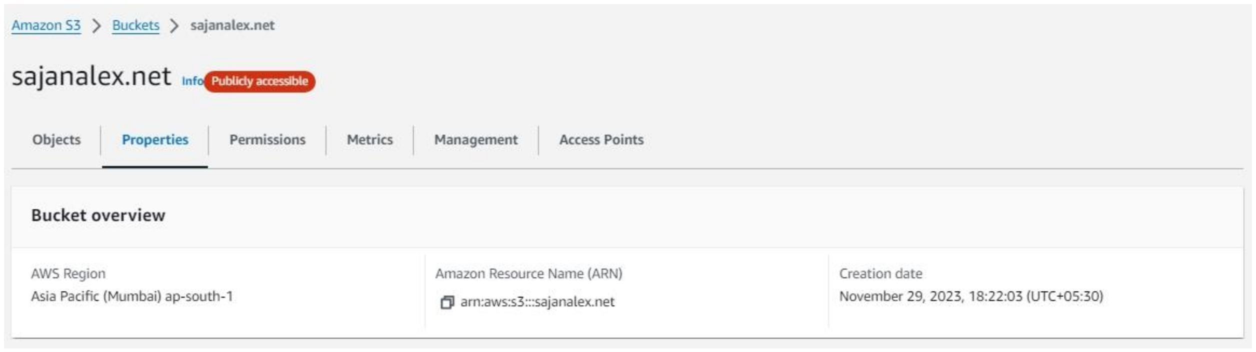
Buckets are containers for data stored in S3. Learn more

Name	AWS Region	Access	Creation date
sajanalex.net	Asia Pacific (Mumbai) ap-south-1	Objects can be public	November 29, 2023, 18:22:03 (UTC+05:30)

3. Configure Bucket Properties (Optional):
   - Choose the newly created bucket.
   - Navigate to the "Properties" tab.
   - Configure properties such as logging, versioning, and events based on your needs.

4. Set Up Static Website Hosting:

- In the "Properties" tab, find the "Static website hosting" card.

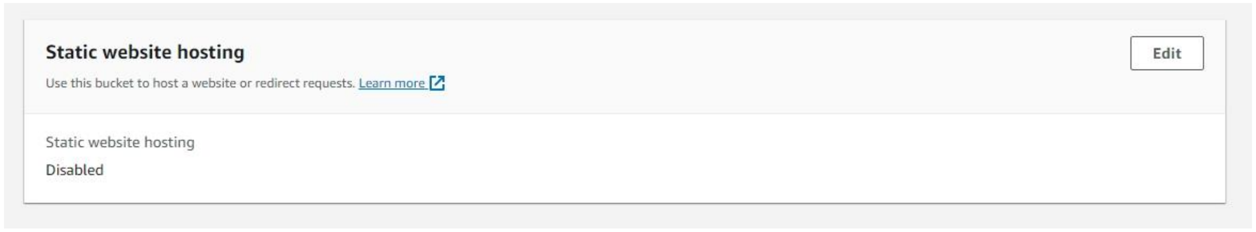


Amazon S3 > Buckets > sajanalex.net

sajanalex.net Info Publicly accessible

Bucket overview

AWS Region	Amazon Resource Name (ARN)	Creation date
Asia Pacific (Mumbai) ap-south-1	arn:aws:s3:::sajanalex.net	November 29, 2023, 18:22:03 (UTC+05:30)

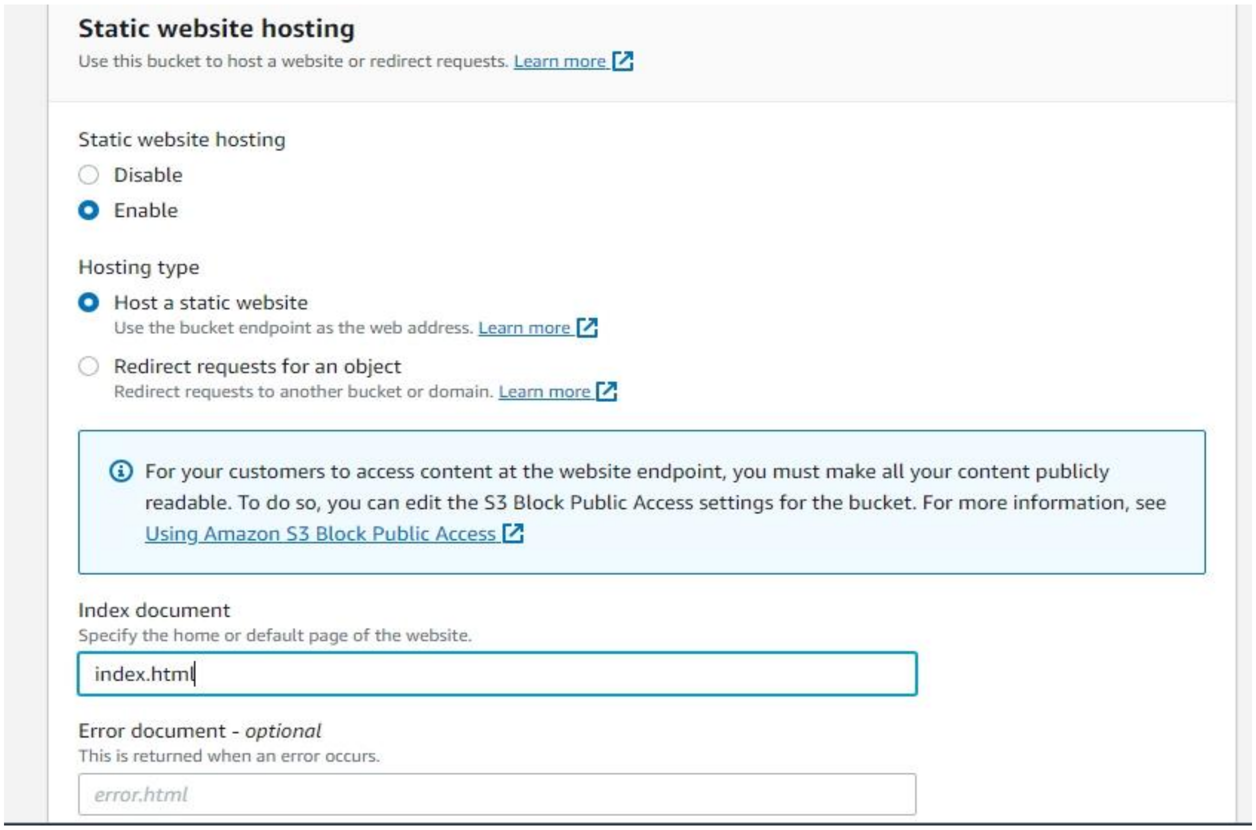


Static website hosting

Use this bucket to host a website or redirect requests. Learn more

Static website hosting
Disabled

- Click on "Edit" and enable static website hosting and choose the "Host a static website" option.
- Specify index and error documents (e.g., index.html).



Static website hosting

Use this bucket to host a website or redirect requests. Learn more

Static website hosting
Disable [✓ Enable]

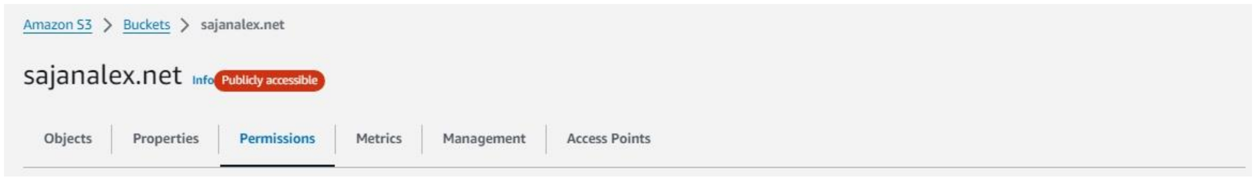
Hosting type
[✓ Host a static website]
Use the bucket endpoint as the web address. Learn more
Redirect requests for an object
Redirect requests to another bucket or domain. Learn more

For your customers to access content at the website endpoint, you must make all your content publicly readable. To do so, you can edit the S3 Block Public Access settings for the bucket. For more information, see Using Amazon S3 Block Public Access

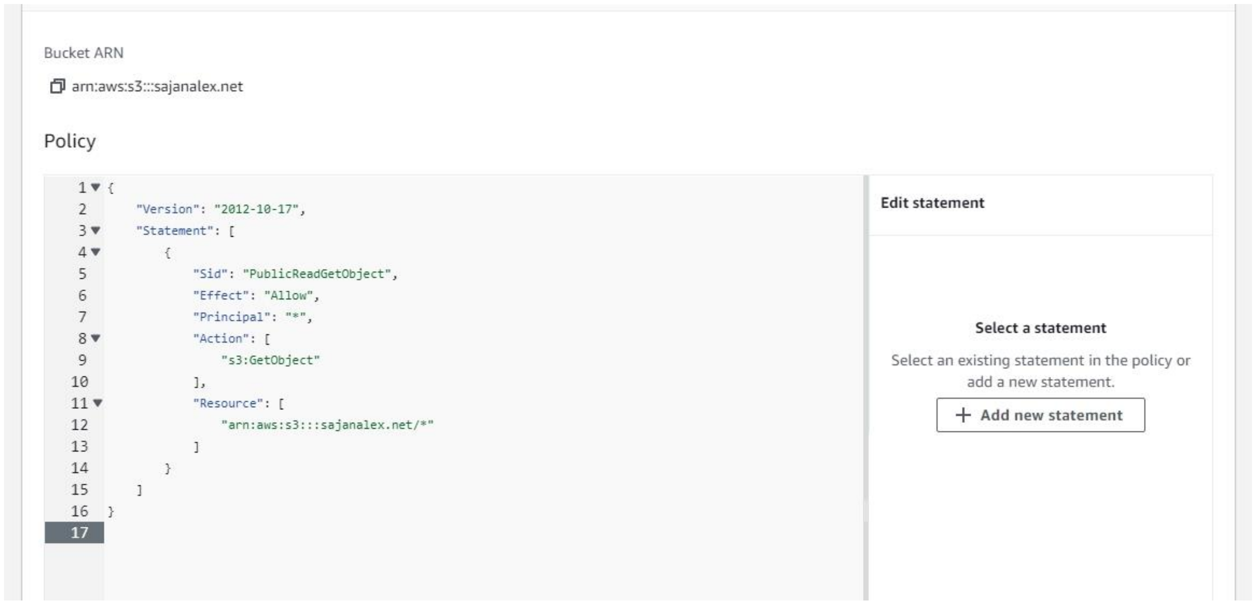
Index document
Specify the home or default page of the website.
index.html

Error document - *optional*
This is returned when an error occurs.
error.html

5. Bucket Policy to grant public access:
   - To grant public access, create a bucket policy.
   - Navigate to Permissions tab and Click on "Bucket Policy."



   - Add a policy allowing "s3:GetObject" permissions for everyone.
   - Modify the policy as per your security requirements.


Bucket ARN

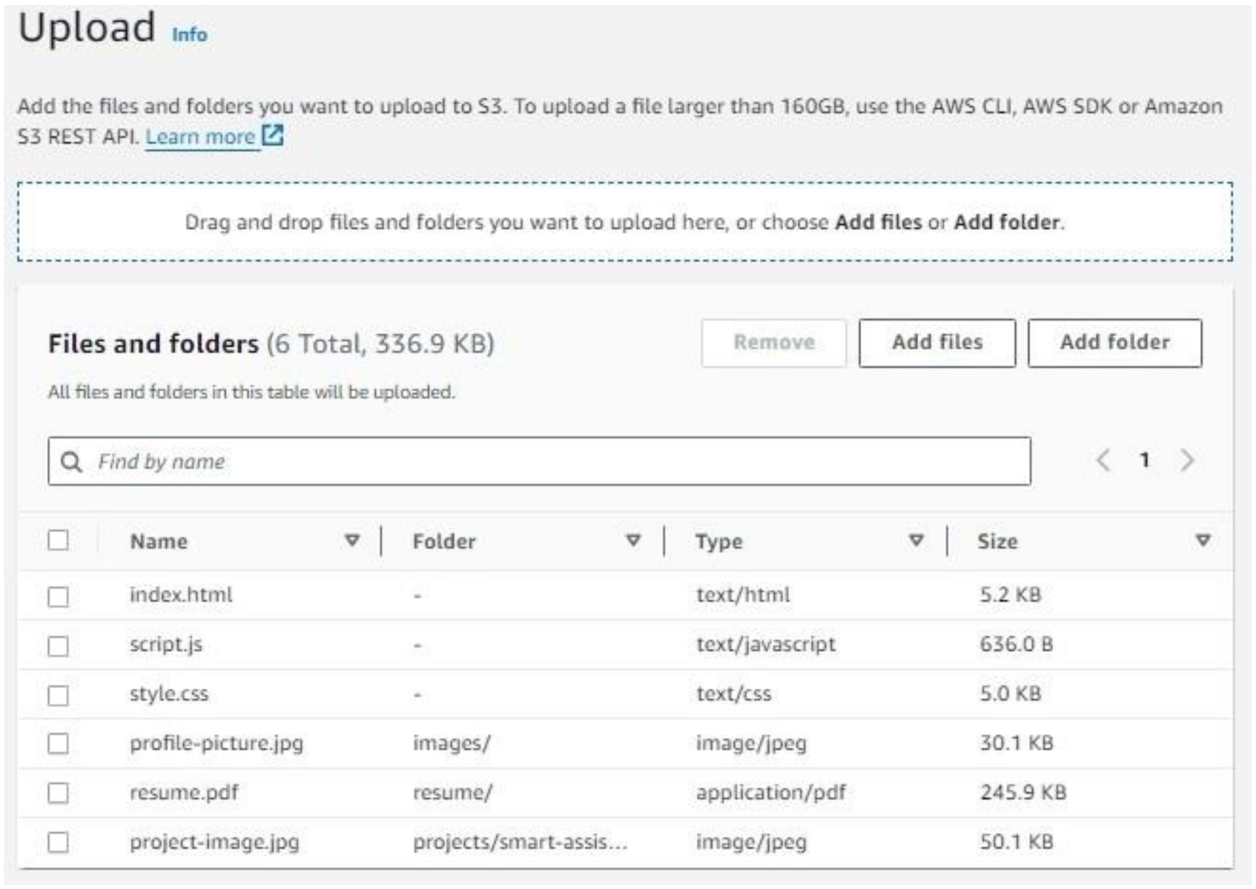
arn:aws:s3:::sajanalex.net

Policy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicReadGetObject",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::sajanalex.net/*"
            ]
        }
    ]
}
```

Edit statement
Select a statement
Select an existing statement in the policy or add a new statement.
+ Add new statement

6. Upload Website Files:
   - Return to the "Overview" tab.
   - Click on "Upload" to add your website files to the bucket.



Upload Info

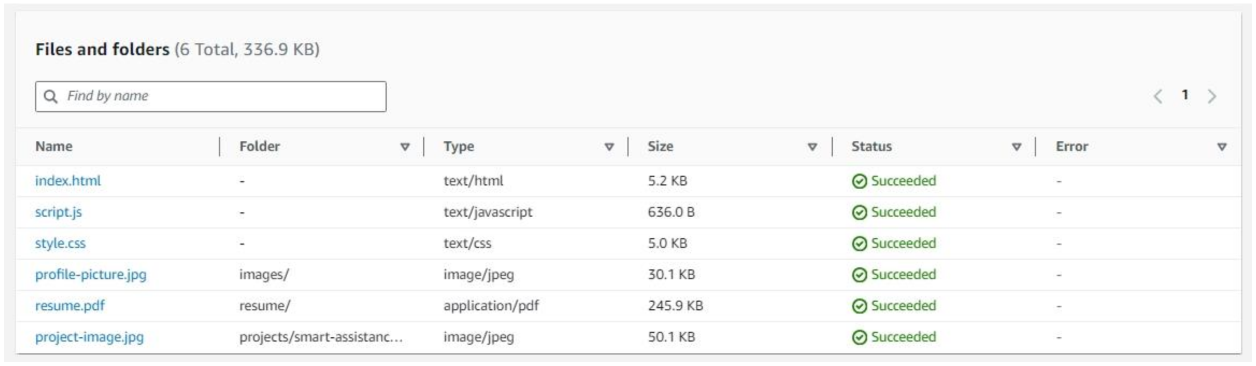
Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. Learn more

Drag and drop files and folders you want to upload here, or choose **Add files** or **Add folder**.

Files and folders (6 Total, 336.9 KB)

All files and folders in this table will be uploaded.

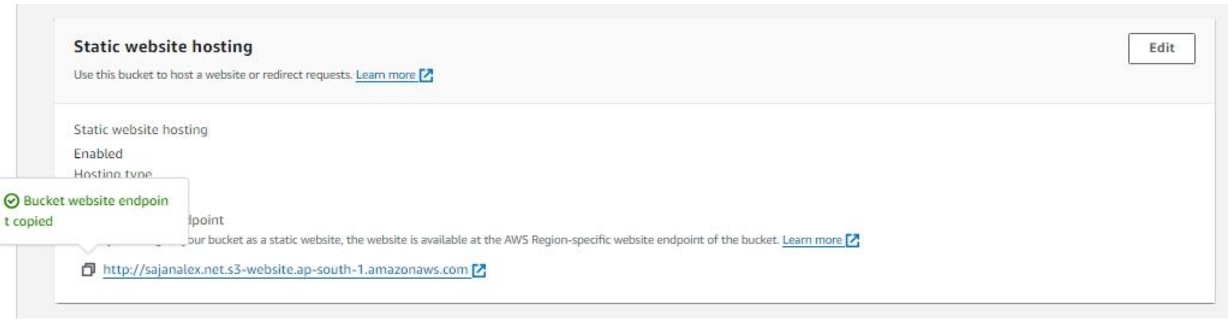
Name	Folder	Type	Size
index.html	-	text/html	5.2 KB
script.js	-	text/javascript	636.0 B
style.css	-	text/css	5.0 KB
profile-picture.jpg	images/	image/jpeg	30.1 KB
resume.pdf	resume/	application/pdf	245.9 KB
project-image.jpg	projects/smart-assis…	image/jpeg	50.1 KB



Files and folders (6 Total, 336.9 KB)

Name	Folder	Type	Size	Status	Error
index.html	-	text/html	5.2 KB	Succeeded	-
script.js	-	text/javascript	636.0 B	Succeeded	-
style.css	-	text/css	5.0 KB	Succeeded	-
profile-picture.jpg	images/	image/jpeg	30.1 KB	Succeeded	-
resume.pdf	resume/	application/pdf	245.9 KB	Succeeded	-
project-image.jpg	projects/smart-assistanc…	image/jpeg	50.1 KB	Succeeded	-

7. Retrieve Endpoint URL:
    - In the "Properties" tab of the bucket, under Static Website Hosting, find the "Bucket Website Endpoint URL”.
    - This URL will be used to access your static website.


Static website hosting

Use this bucket to host a website or redirect requests. Learn more

Static website hosting

Enabled

Hosting type

Bucket website endpoint copied

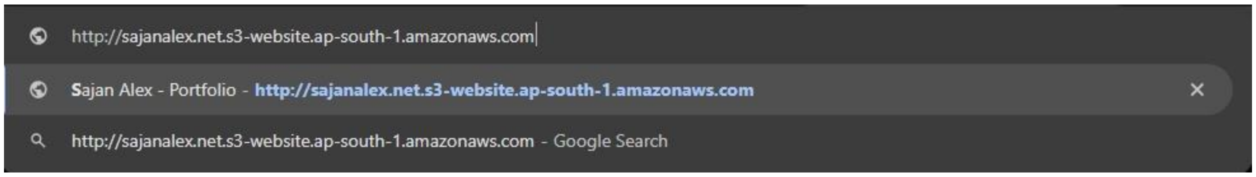
…dpoint

…our bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. Learn more

http://sajanalex.net.s3-website.ap-south-1.amazonaws.com

8. Test Your Website:

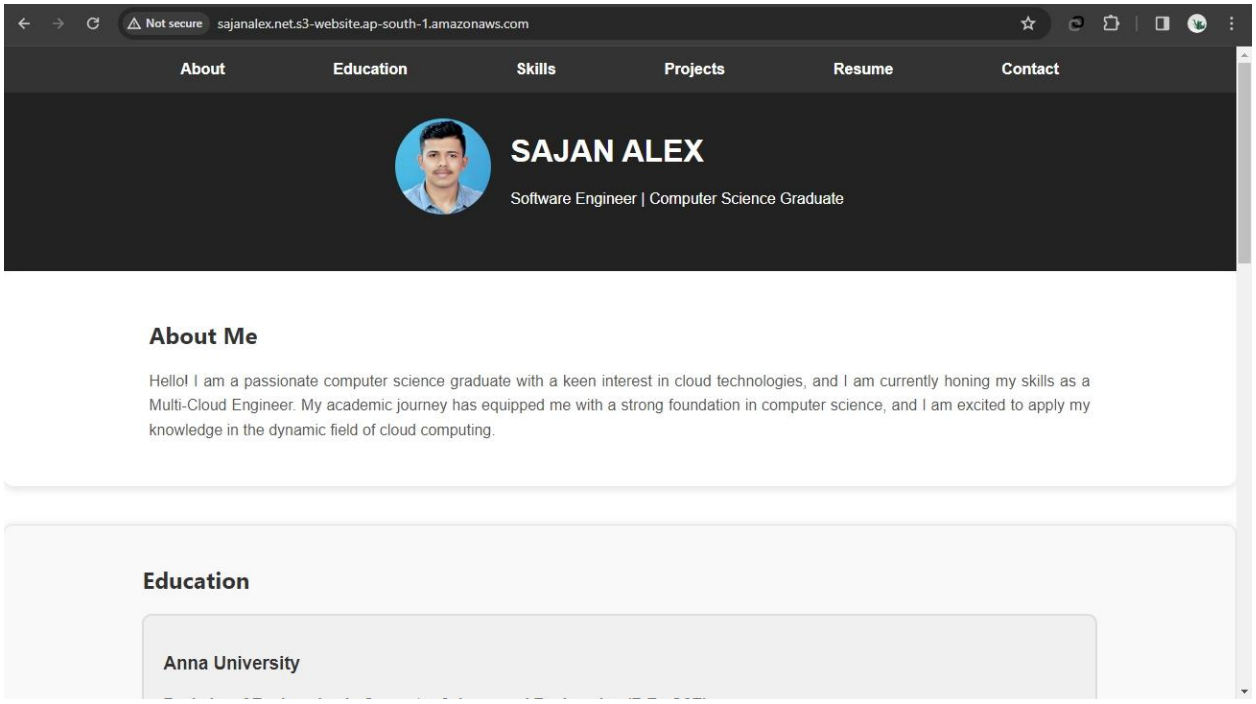
    - Open a web browser tab and enter the endpoint URL.


http://sajanalex.net.s3-website.ap-south-1.amazonaws.com

Sajan Alex - Portfolio - http://sajanalex.net.s3-website.ap-south-1.amazonaws.com

http://sajanalex.net.s3-website.ap-south-1.amazonaws.com - Google Search

    - Ensure your website loads correctly.


Not secure sajanalex.net.s3-website.ap-south-1.amazonaws.com

SAJAN ALEX

Software Engineer | Computer Science Graduate

About Me

Hello! I am a passionate computer science graduate with a keen interest in cloud technologies, and I am currently honing my skills as a Multi-Cloud Engineer. My academic journey has equipped me with a strong foundation in computer science, and I am excited to apply my knowledge in the dynamic field of cloud computing.

Education

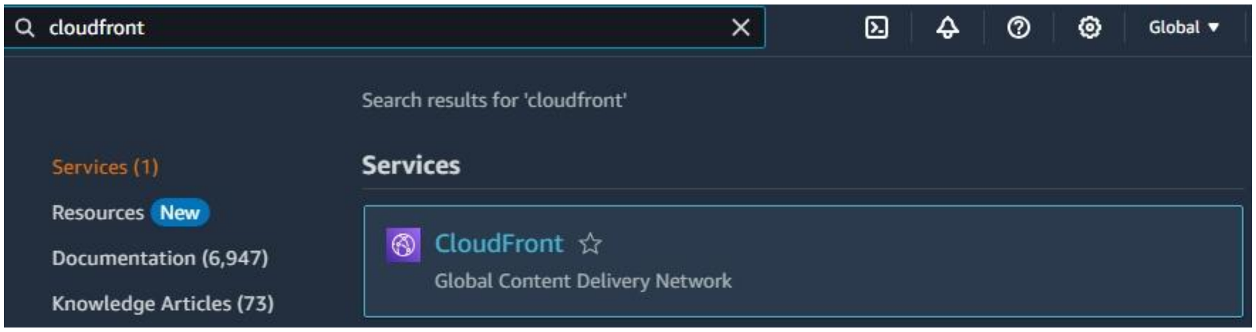
Anna University

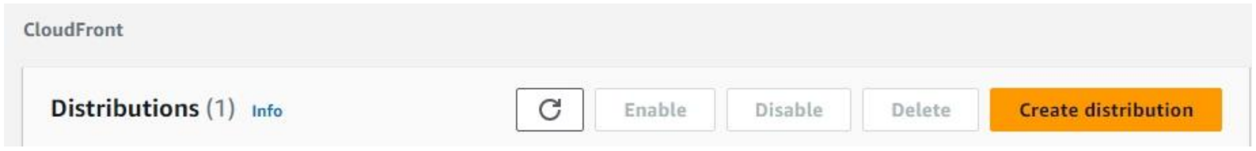
9. Optional - Configure Custom Domain:

    - If using a custom domain, configure DNS settings and set up routing to the S3 static website.

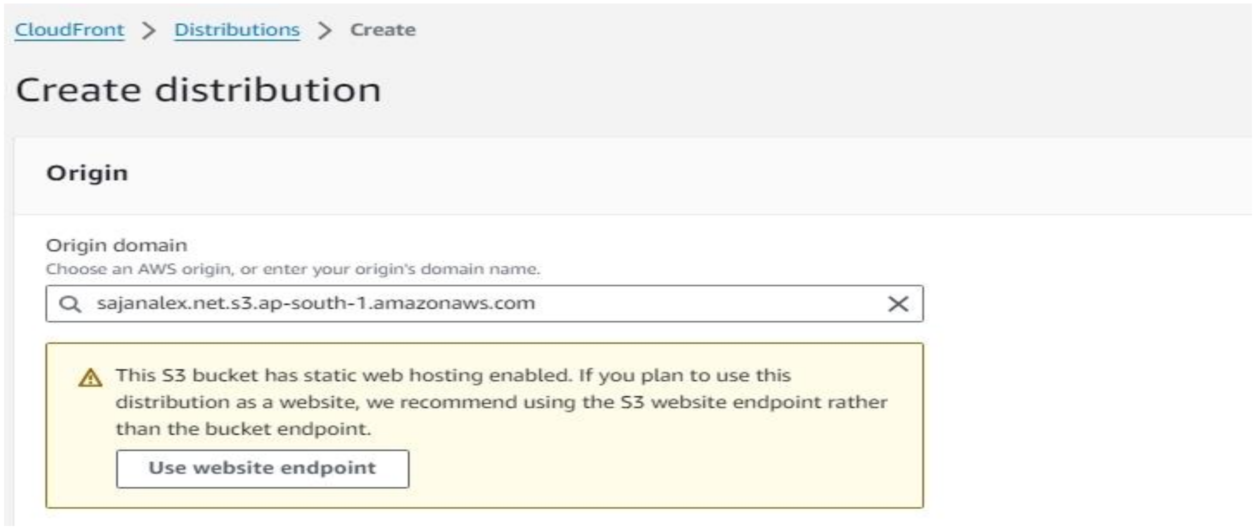
6.3 CloudFront Distribution Setup

1. Access CloudFront Console:
   - Navigate to the CloudFront service.
   - Click "Create Distribution."





2. Origin Settings:
   - In the "Origin Settings" section:
     - Choose the S3 bucket as the "Origin Domain Name."
     - When we select the S3 bucket, a warning pop-up will appear asking us to use the S3 website endpoint rather than the bucket endpoint. Click on the **use website endpoint** button.
     - Configure other settings based on your requirements or keep the default settings.


CloudFront > Distributions > Create

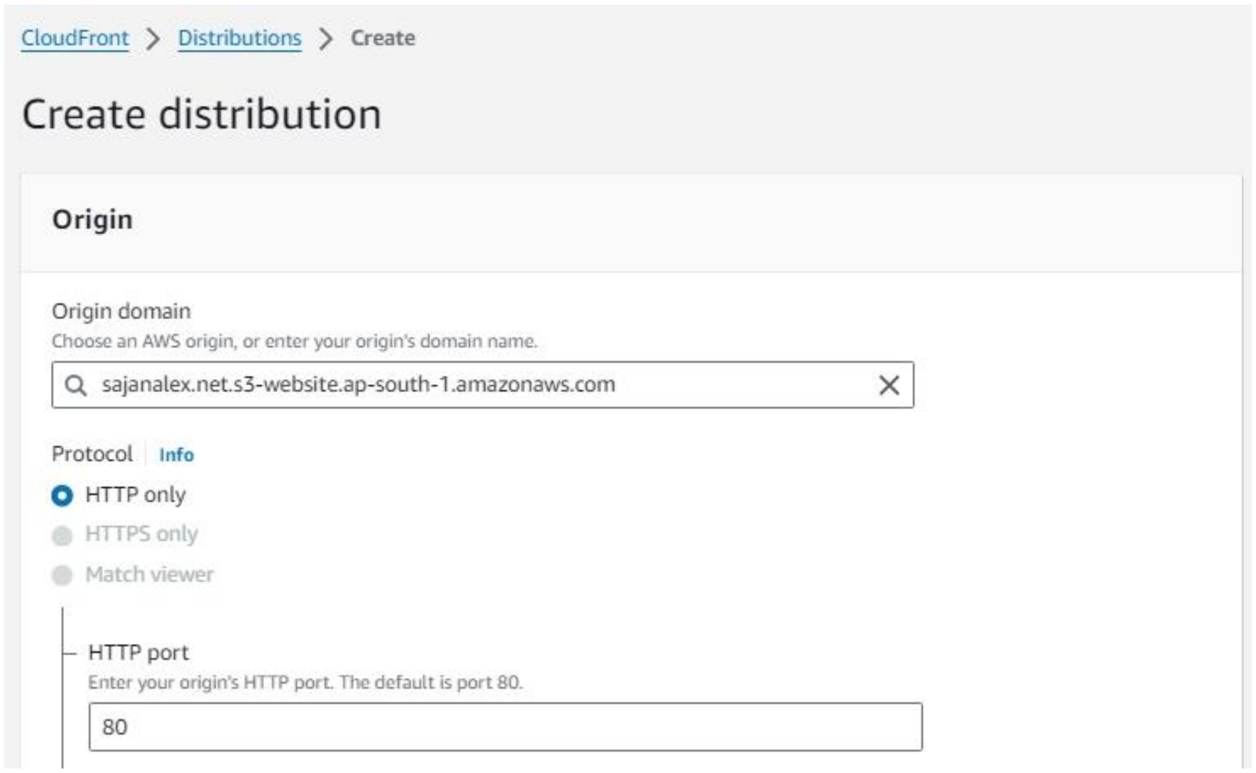
Create distribution

Origin

Origin domain
Choose an AWS origin, or enter your origin's domain name.
sajanalex.net.s3.ap-south-1.amazonaws.com

This S3 bucket has static web hosting enabled. If you plan to use this distribution as a website, we recommend using the S3 website endpoint rather than the bucket endpoint.

Use website endpoint



CloudFront > Distributions > Create

Create distribution

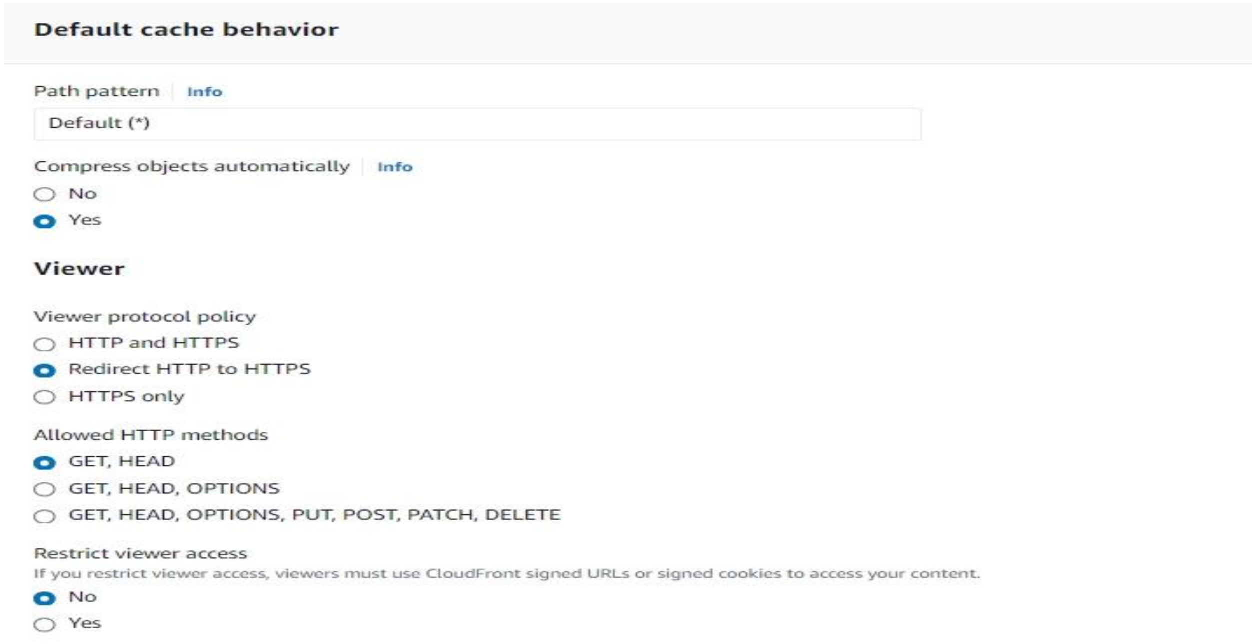
Origin

Origin domain
Choose an AWS origin, or enter your origin's domain name.
sajanalex.net.s3-website.ap-south-1.amazonaws.com

Protocol
[✓ HTTP only] HTTPS only Match viewer

HTTP port
Enter your origin's HTTP port. The default is port 80.
80

3. Default cache behavior:
    - Under viewer protocol policy:
        - Choose Redirect HTTP to HTTPS.
        - Keep the rest as default.



Default cache behavior

Path pattern
Default (*)

Compress objects automatically
No [✓ Yes]

Viewer

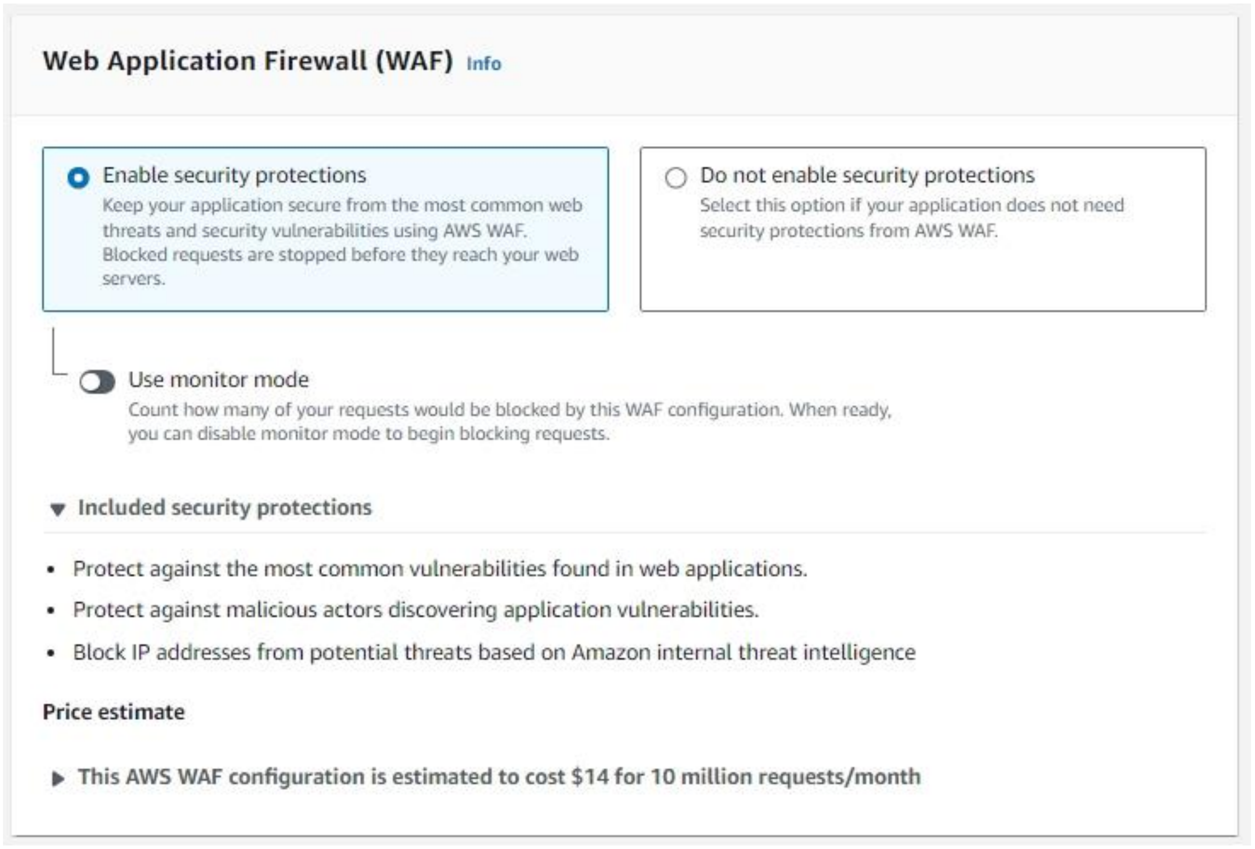
Viewer protocol policy
HTTP and HTTPS [✓ Redirect HTTP to HTTPS] HTTPS only

Allowed HTTP methods
[✓ GET, HEAD] GET, HEAD, OPTIONS GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE

Restrict viewer access
If you restrict viewer access, viewers must use CloudFront signed URLs or signed cookies to access your content.
[✓ No] Yes

4. Web Application Firewall (WAF):

- Configure AWS WAF (Web Application Firewall) if additional security is required.
- Enabling the firewall will incur supplementary fees.



Web Application Firewall (WAF) Info

[✓ Enable security protections] Keep your application secure from the most common web threats and security vulnerabilities using AWS WAF. Blocked requests are stopped before they reach your web servers.

Do not enable security protections: Select this option if your application does not need security protections from AWS WAF.

Use monitor mode: Count how many of your requests would be blocked by this WAF configuration. When ready, you can disable monitor mode to begin blocking requests.

Included security protections

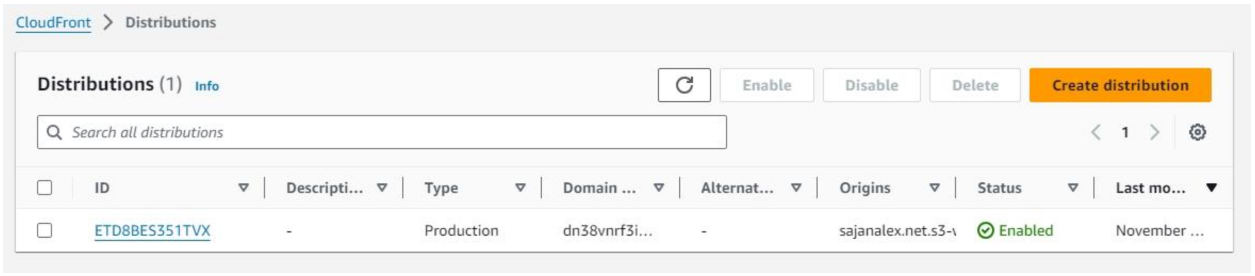
- Protect against the most common vulnerabilities found in web applications.
- Protect against malicious actors discovering application vulnerabilities.
- Block IP addresses from potential threats based on Amazon internal threat intelligence

Price estimate

This AWS WAF configuration is estimated to cost $14 for 10 million requests/month

5. Settings Section:

- Configure the settings based on your requirements or keep the default settings.
- If using a custom domain, consider configuring a custom SSL/TLS certificate for HTTPS.
- Use AWS Certificate Manager (ACM) to request and manage SSL/TLS certificates.
- Click **Create Distribution**.

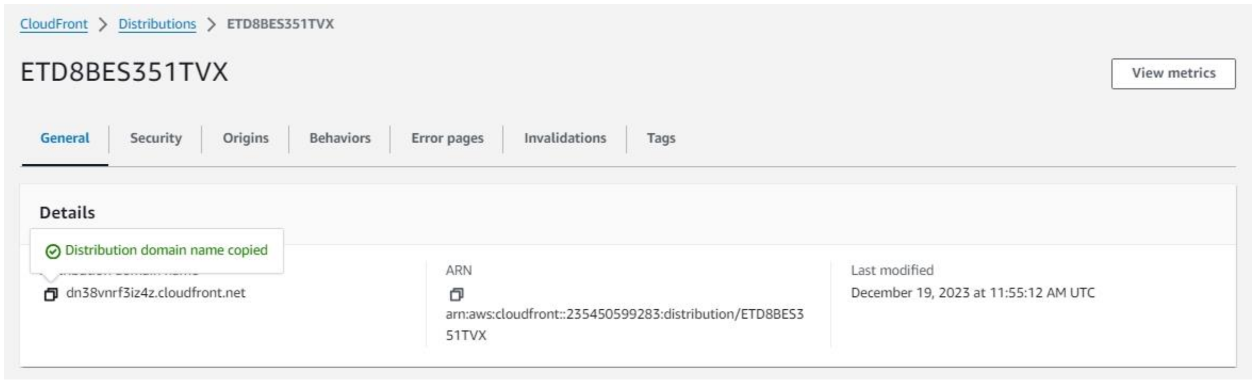


CloudFront > Distributions

Distributions (1) Info

ID	Descripti...	Type	Domain ...	Alternat...	Origins	Status	Last mo...
ETD8BES351TVX	-	Production	dn38vnrf3i...	-	sajanalex.net.s3-\	✓ Enabled	November ...

6. Distribution Domain Name:
    - Note the Distribution Domain Name assigned to your CloudFront distribution.


CloudFront > Distributions > ETD8BES351TVX

ETD8BES351TVX

Details

Distribution domain name copied

dn38vnrf3iz4z.cloudfront.net

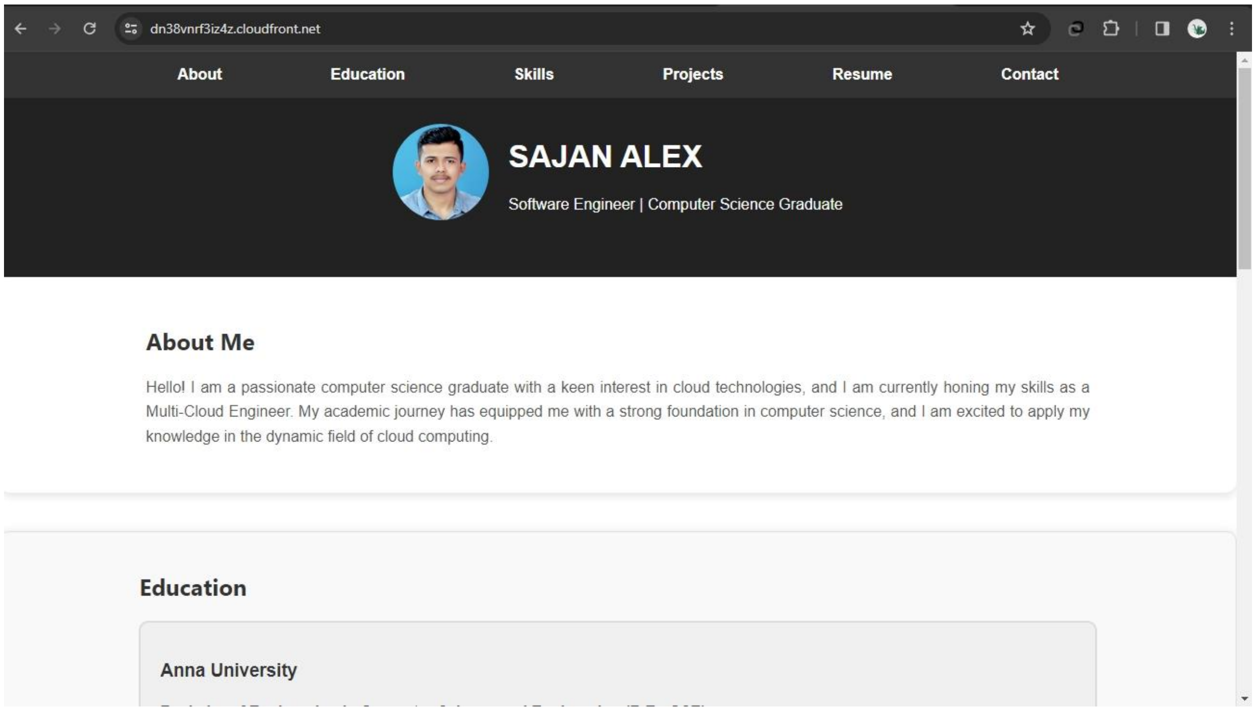
ARN

arn:aws:cloudfront::235450599283:distribution/ETD8BES351TVX

Last modified

December 19, 2023 at 11:55:12 AM UTC

7. Testing:
    - Test the distribution by accessing the CloudFront URL or custom domain if configured.
    - Ensure content is served correctly, and caching behaviour aligns with expectations.


dn38vnrf3iz4z.cloudfront.net

SAJAN ALEX

Software Engineer | Computer Science Graduate

About Me

Hello! I am a passionate computer science graduate with a keen interest in cloud technologies, and I am currently honing my skills as a Multi-Cloud Engineer. My academic journey has equipped me with a strong foundation in computer science, and I am excited to apply my knowledge in the dynamic field of cloud computing.

Education

Anna University

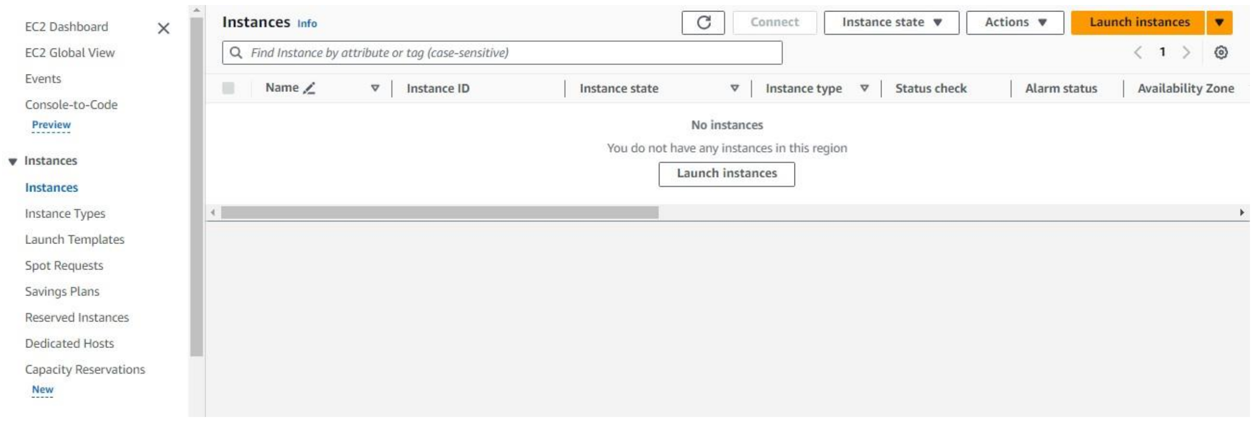
8. Documentation:
    - Document the CloudFront distribution settings and configurations for future reference.

6.4 DNS Configuration with Route 53 (Optional)

1. Access Route 53 Console:
    - Navigate to the Route 53 service.
2. Create Hosted Zone:
    - Click "Create Hosted Zone."
    - Enter your domain name (e.g., example.com).
    - Click "Create."
3. Record Sets:
    - Inside the hosted zone, create a record set for your domain.
    - Choose the record type based on your needs (e.g., A or CNAME).
    - For an A record, configure it with an alias to your CloudFront distribution.
    - For a CNAME record, point it to the CloudFront domain name.
4. Alias Configuration:
    - If using an A record, set the alias target to your CloudFront distribution.
    - If using a CNAME record, enter the CloudFront domain name.
5. TTL (Time To Live):
    - Set the TTL value based on your desired DNS caching behaviour.
6. Save Changes:
    - Save the record set changes.
7. Update Name Servers:
    - Note the name servers assigned by Route 53.
    - Update the name servers with your domain registrar if not using Route 53 as your registrar.
8. Propagation:
    - DNS changes may take time to propagate globally.
    - Monitor the status of the changes in the Route 53 console.
9. Testing:
    - Test your website using the custom domain.
    - Ensure the domain correctly resolves to your CloudFront distribution.

6.5 EC2 Instance Setup

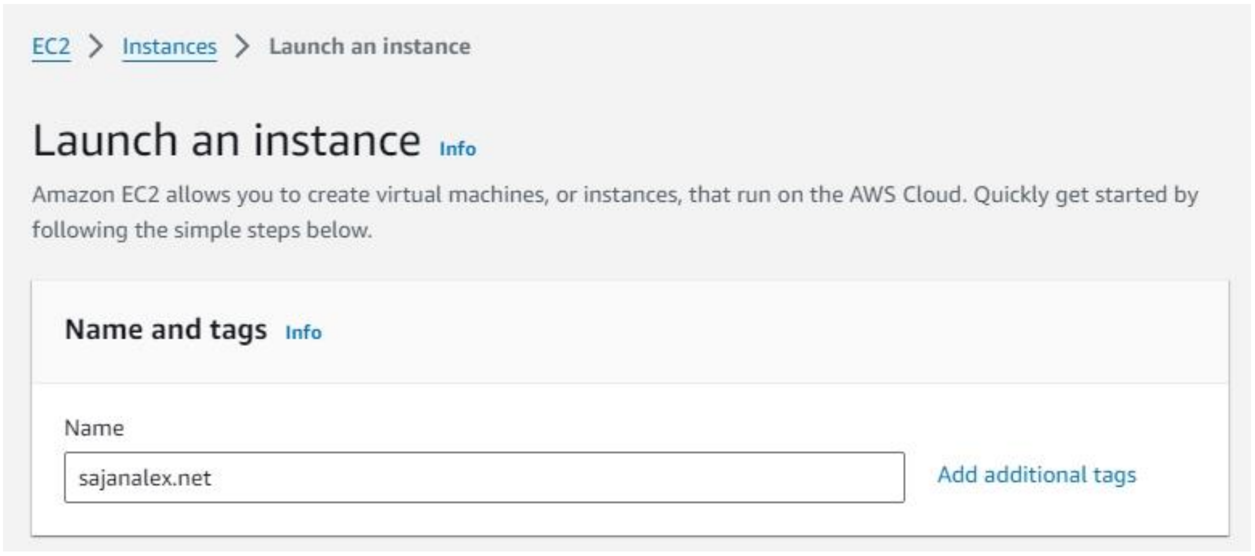
1. EC2 Instance Launch:
   - In the EC2 service dashboard.
   - Go to "Instances" and click "Launch Instances."



No instances

You do not have any instances in this region

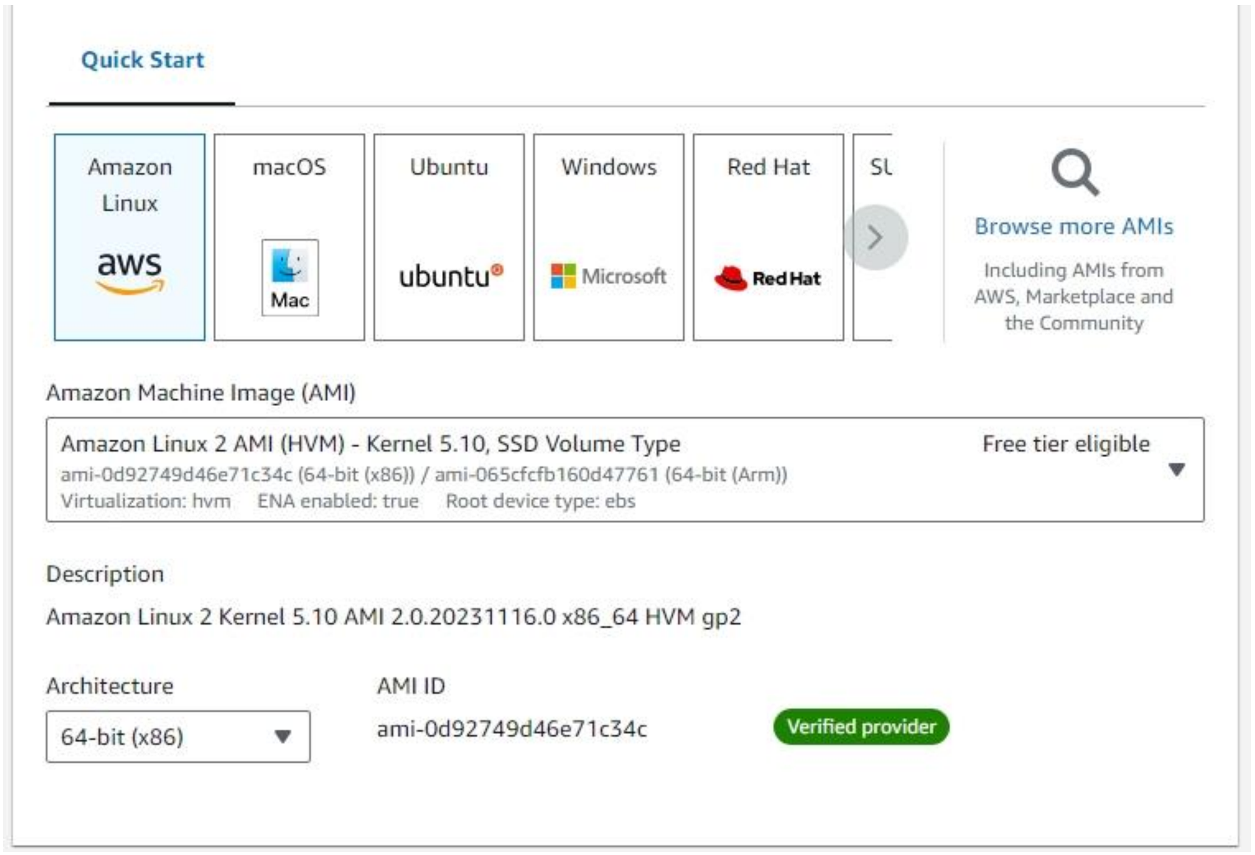
2. Provide a name for the instance:
   - Provide a name tag for the instance (**sajanalex.net**).



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: sajanalex.net

3. Select an Amazon Machine Image (AMI):
   - Choose Amazon Linux 2 AMI (Free tier eligible).



Amazon Machine Image (AMI)

Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type — Free tier eligible
ami-0d92749d46e71c34c (64-bit (x86)) / ami-065cfcfb160d47761 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Description

Amazon Linux 2 Kernel 5.10 AMI 2.0.20231116.0 x86_64 HVM gp2

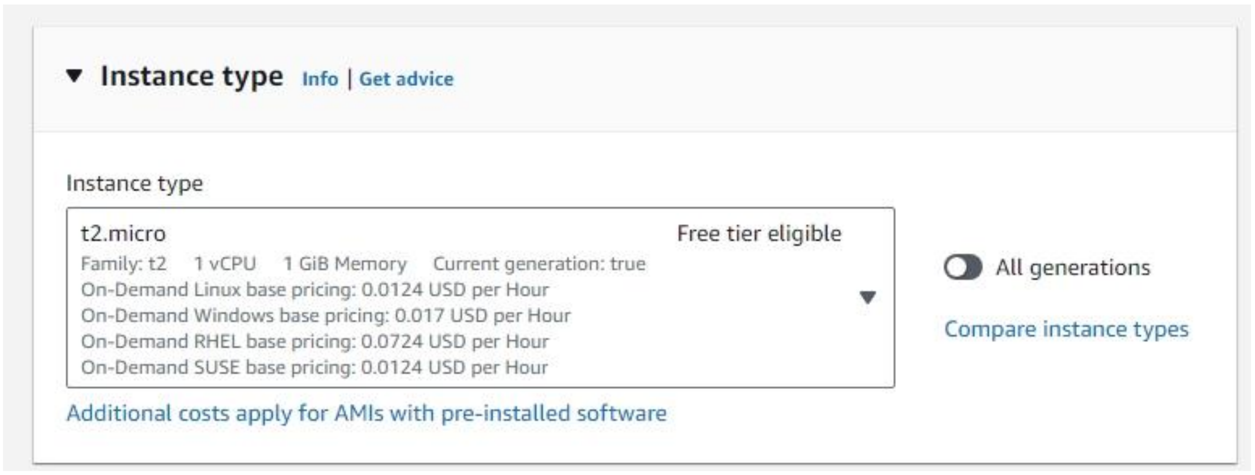
Architecture: 64-bit (x86)

AMI ID: ami-0d92749d46e71c34c

Verified provider

4. Instance Type:

- Choose the T.2 micro instance type, eligible for the free tier.



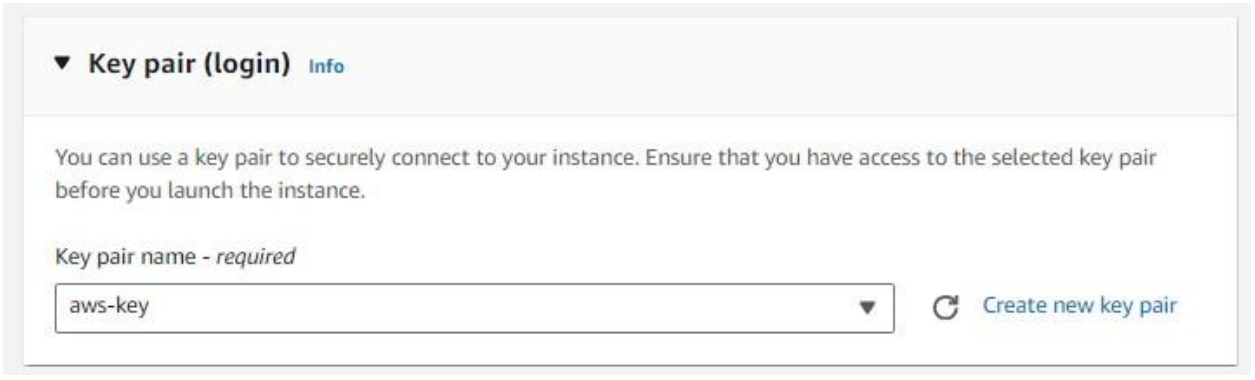
Instance type

t2.micro — Free tier eligible
Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Linux base pricing: 0.0124 USD per Hour
On-Demand Windows base pricing: 0.017 USD per Hour
On-Demand RHEL base pricing: 0.0724 USD per Hour
On-Demand SUSE base pricing: 0.0124 USD per Hour

Additional costs apply for AMIs with pre-installed software

5. Key Pair:

- Create a new key pair or proceed with an existing key pair.



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.

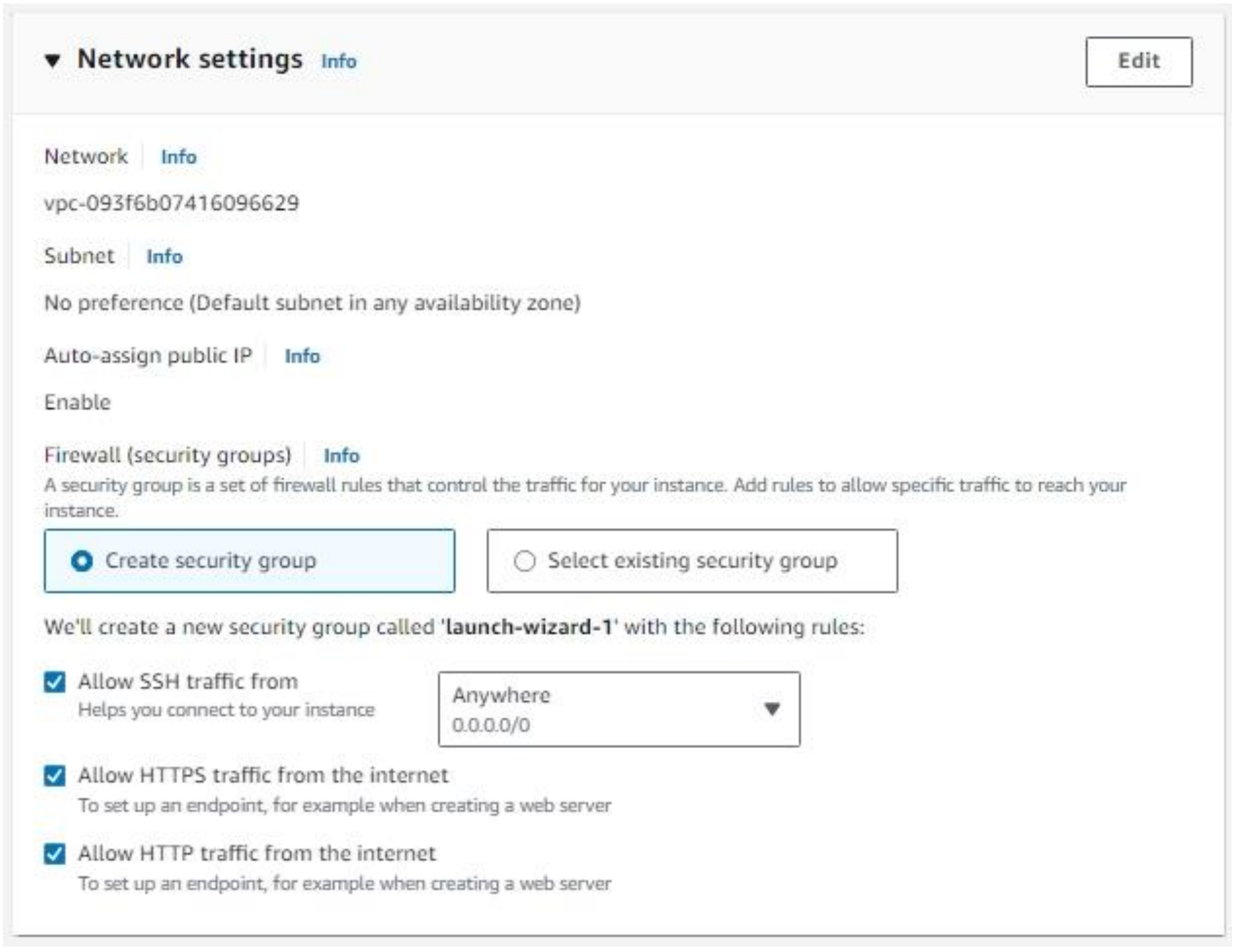
Key pair name - *required*

aws-key

Create new key pair

6. Configure Network Settings:

- Proceed with default VPC and subnet configurations.
- Enable **Auto-assign public IP**.
- Create a new security group allowing SSH, HTTP & HTTPS rules.



Network settings Info

Edit

Network Info

vpc-093f6b07416096629

Subnet Info

No preference (Default subnet in any availability zone)

Auto-assign public IP Info

Enable

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

We'll create a new security group called '**launch-wizard-1**' with the following rules:

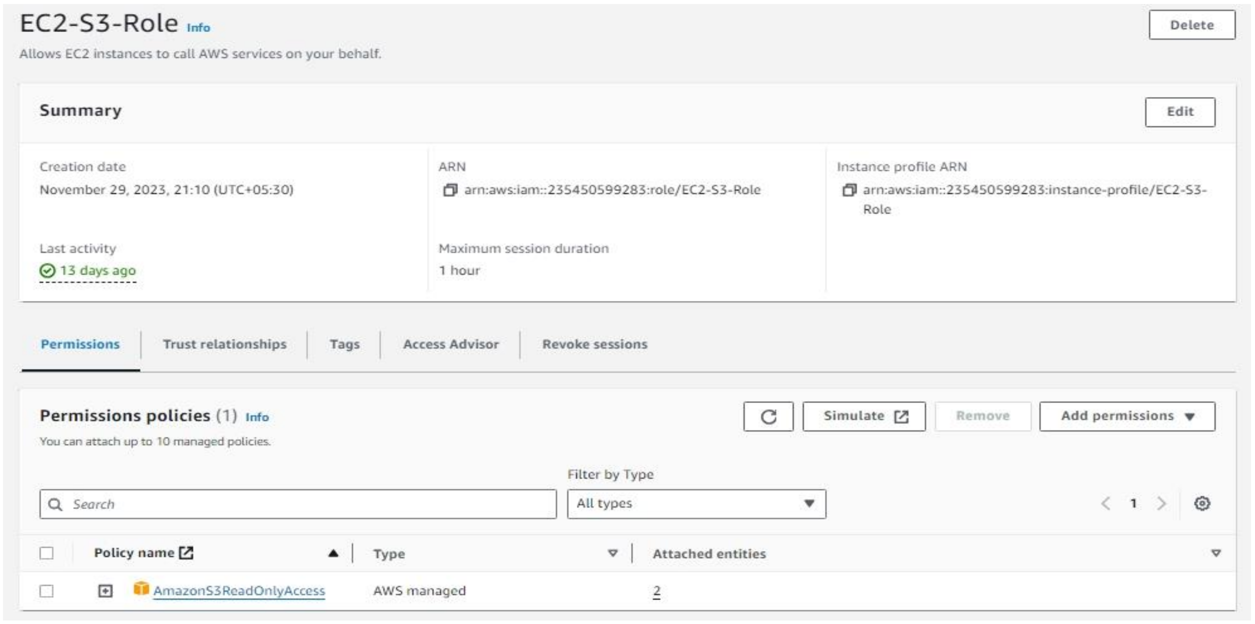
✓ Allow SSH traffic from — Anywhere 0.0.0.0/0
Helps you connect to your instance

✓ Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server

✓ Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server

7. Configure IAM Instance Profile:

- Create an IAM Role that has the necessary permissions to read from the S3 bucket. The role should have the **AmazonS3ReadOnlyAccess** policy or a custom policy with the required permissions.



EC2-S3-Role Info

Allows EC2 instances to call AWS services on your behalf.

Summary

Creation date: November 29, 2023, 21:10 (UTC+05:30)

ARN: arn:aws:iam::235450599283:role/EC2-S3-Role

Instance profile ARN: arn:aws:iam::235450599283:instance-profile/EC2-S3-Role

Last activity: 13 days ago

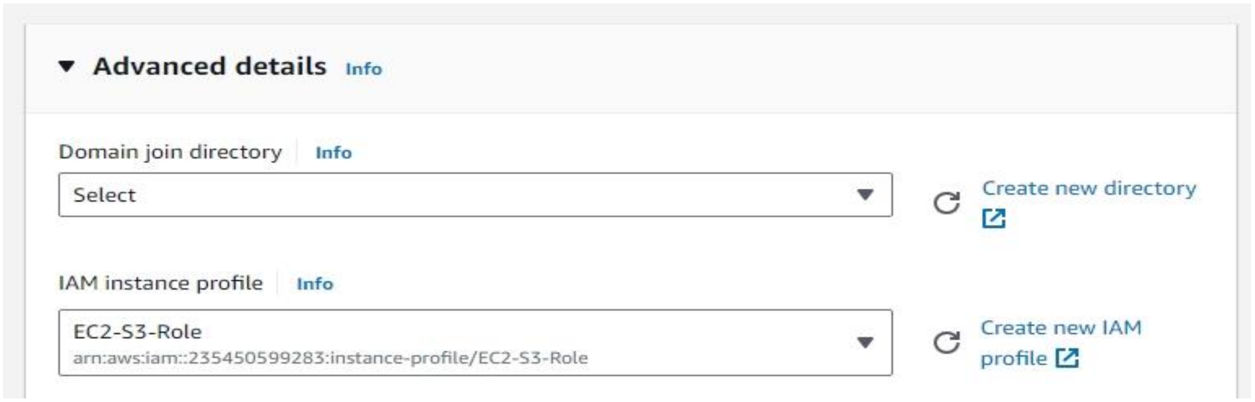
Maximum session duration: 1 hour

Permissions policies (1) Info

You can attach up to 10 managed policies.

Policy name	Type	Attached entities
AmazonS3ReadOnlyAccess	AWS managed	2

- Select the IAM EC2 Role created for EC2-S3 connection in the **IAM instance profile** under the **Advanced details** section.



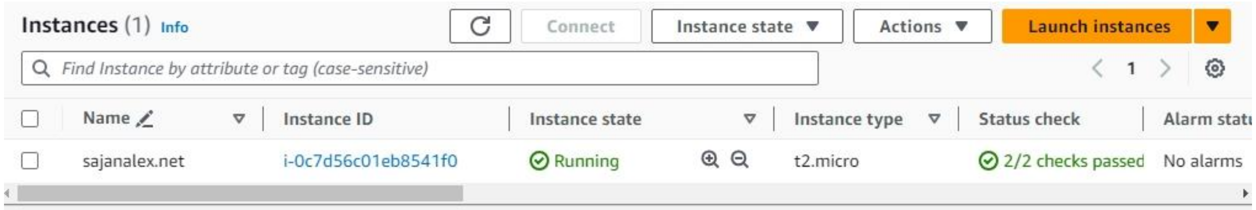
Advanced details Info

Domain join directory Info: Select

IAM instance profile Info: EC2-S3-Role arn:aws:iam::235450599283:instance-profile/EC2-S3-Role

8. Review and Launch:

- Review the configured details on the Review Instance Launch page. Click "Launch."

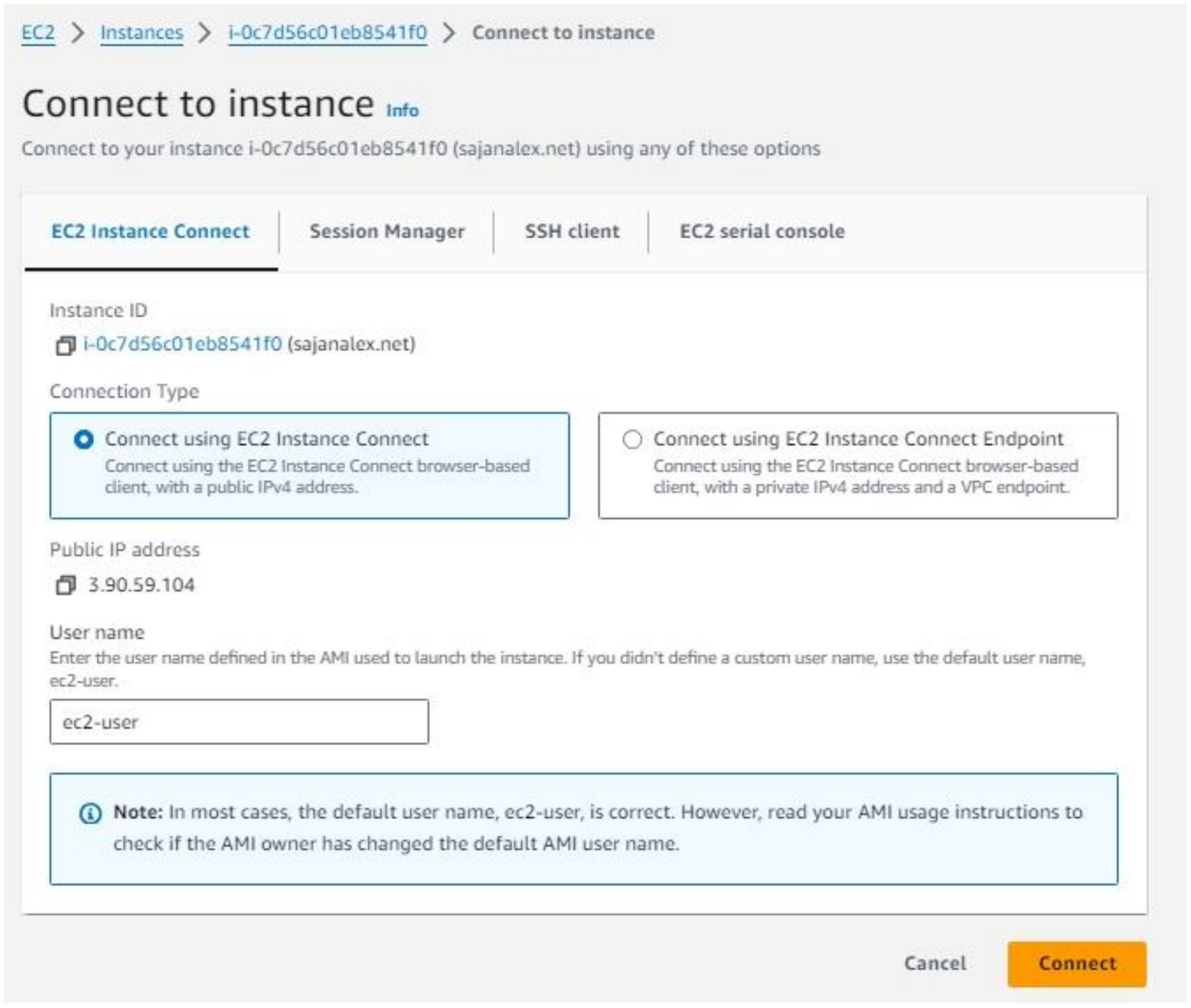


Instances (1) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm statu
sajanalex.net	i-0c7d56c01eb8541f0	Running	t2.micro	2/2 checks passed	No alarms

6.6 Web Server Software Installation

1. Connect to EC2 Instance:
   - Use SSH to connect to your Amazon Linux 2 instance.



EC2 > Instances > i-0c7d56c01eb8541f0 > Connect to instance

Connect to instance Info

Connect to your instance i-0c7d56c01eb8541f0 (sajanalex.net) using any of these options

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

Instance ID

i-0c7d56c01eb8541f0 (sajanalex.net)

Connection Type

● Connect using EC2 Instance Connect
Connect using the EC2 Instance Connect browser-based client, with a public IPv4 address.

○ Connect using EC2 Instance Connect Endpoint
Connect using the EC2 Instance Connect browser-based client, with a private IPv4 address and a VPC endpoint.

Public IP address

3.90.59.104

User name

Enter the user name defined in the AMI used to launch the instance. If you didn't define a custom user name, use the default user name, ec2-user.

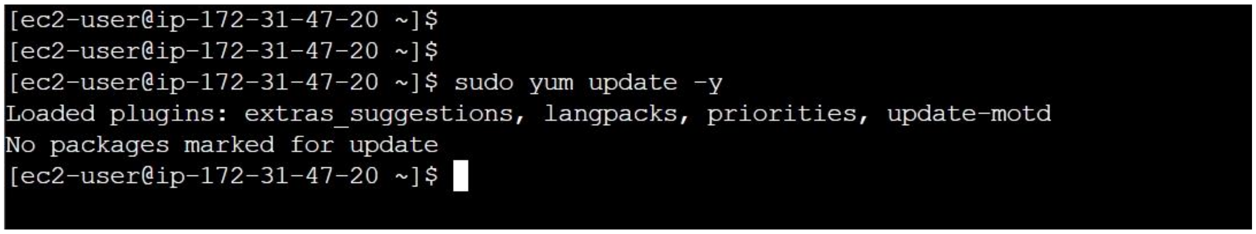
ec2-user

Note: In most cases, the default user name, ec2-user, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

Cancel | Connect

2. Update Package Manager:

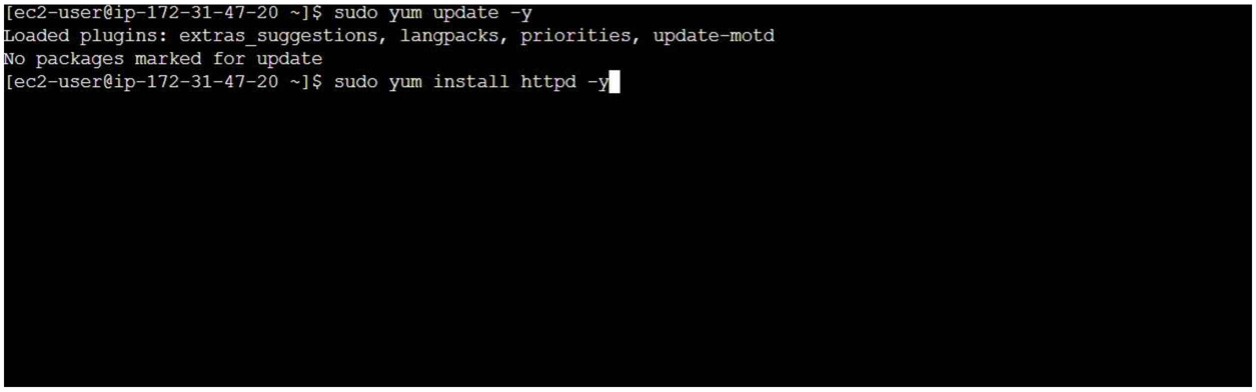
   sudo yum update -y



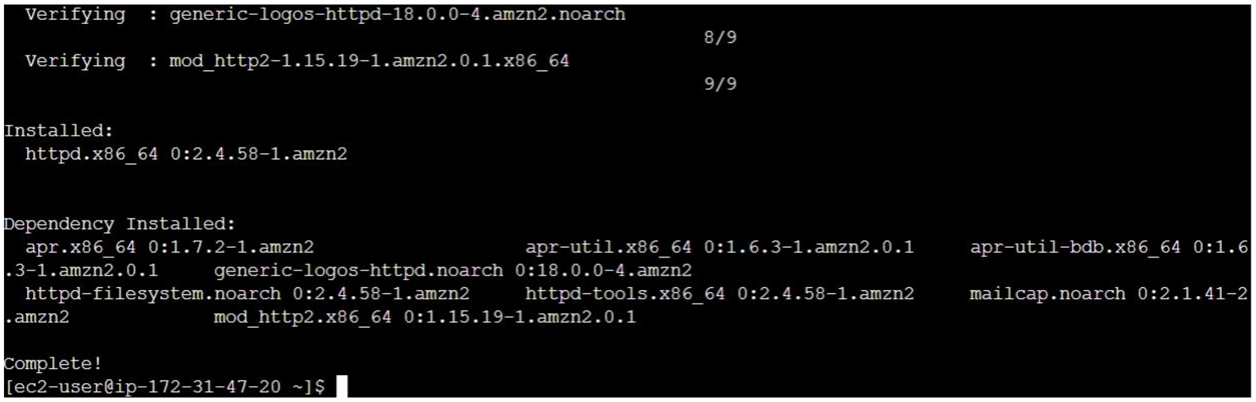
```
[ec2-user@ip-172-31-47-20 ~]$
[ec2-user@ip-172-31-47-20 ~]$
[ec2-user@ip-172-31-47-20 ~]$ sudo yum update -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
No packages marked for update
[ec2-user@ip-172-31-47-20 ~]$
```

3. Install Apache Web Server:

sudo yum install httpd -y



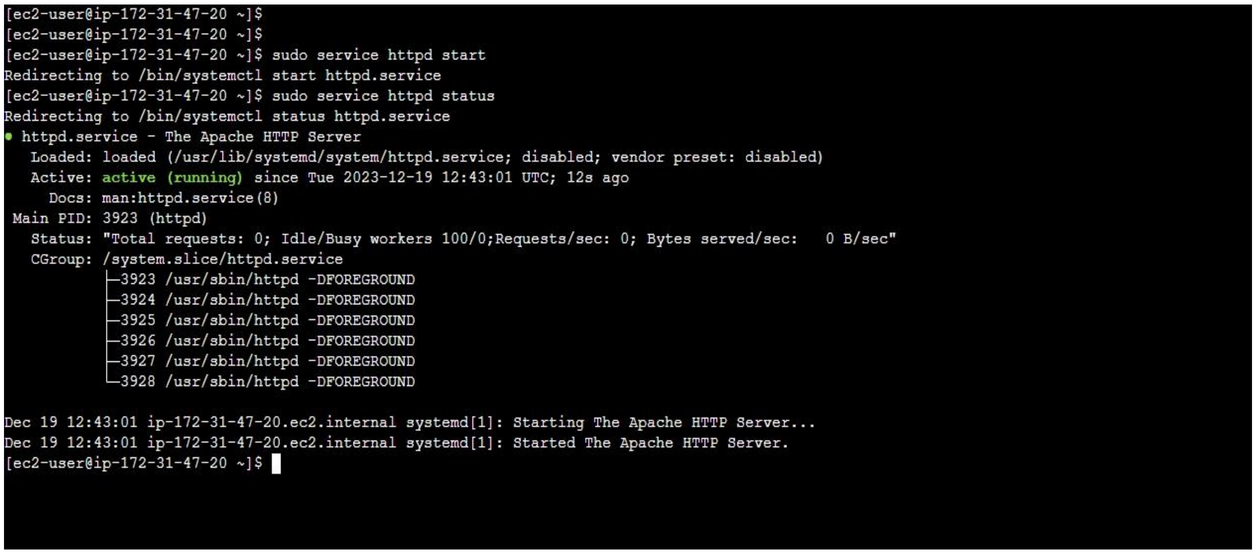
```
[ec2-user@ip-172-31-47-20 ~]$ sudo yum update -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
No packages marked for update
[ec2-user@ip-172-31-47-20 ~]$ sudo yum install httpd -y
```



```
  Verifying  : generic-logos-httpd-18.0.0-4.amzn2.noarch
                                                                   8/9
  Verifying  : mod_http2-1.15.19-1.amzn2.0.1.x86_64
                                                                   9/9

Installed:
  httpd.x86_64 0:2.4.58-1.amzn2


Dependency Installed:
  apr.x86_64 0:1.7.2-1.amzn2                  apr-util.x86_64 0:1.6.3-1.amzn2.0.1     apr-util-bdb.x86_64 0:1.6
.3-1.amzn2.0.1     generic-logos-httpd.noarch 0:18.0.0-4.amzn2
  httpd-filesystem.noarch 0:2.4.58-1.amzn2     httpd-tools.x86_64 0:2.4.58-1.amzn2     mailcap.noarch 0:2.1.41-2
.amzn2             mod_http2.x86_64 0:1.15.19-1.amzn2.0.1

Complete!
[ec2-user@ip-172-31-47-20 ~]$
```

4. Start Apache Service:

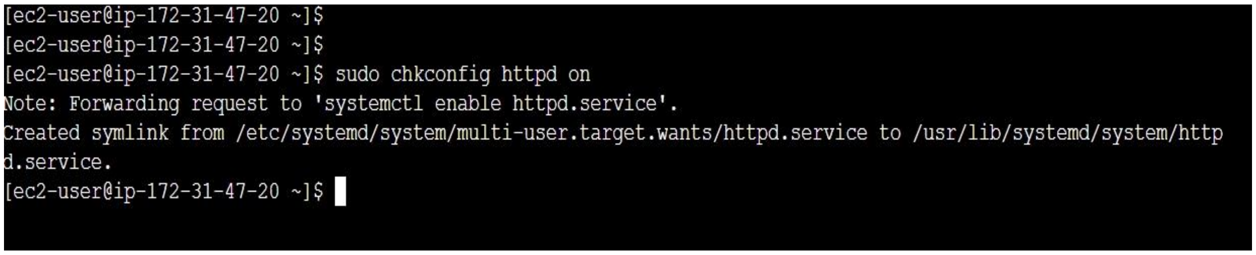
   sudo service httpd start



```
[ec2-user@ip-172-31-47-20 ~]$
[ec2-user@ip-172-31-47-20 ~]$
[ec2-user@ip-172-31-47-20 ~]$ sudo service httpd start
Redirecting to /bin/systemctl start httpd.service
[ec2-user@ip-172-31-47-20 ~]$ sudo service httpd status
Redirecting to /bin/systemctl status httpd.service
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
   Active: active (running) since Tue 2023-12-19 12:43:01 UTC; 12s ago
     Docs: man:httpd.service(8)
 Main PID: 3923 (httpd)
   Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes served/sec:   0 B/sec"
   CGroup: /system.slice/httpd.service
           ├─3923 /usr/sbin/httpd -DFOREGROUND
           ├─3924 /usr/sbin/httpd -DFOREGROUND
           ├─3925 /usr/sbin/httpd -DFOREGROUND
           ├─3926 /usr/sbin/httpd -DFOREGROUND
           ├─3927 /usr/sbin/httpd -DFOREGROUND
           └─3928 /usr/sbin/httpd -DFOREGROUND

Dec 19 12:43:01 ip-172-31-47-20.ec2.internal systemd[1]: Starting The Apache HTTP Server...
Dec 19 12:43:01 ip-172-31-47-20.ec2.internal systemd[1]: Started The Apache HTTP Server.
[ec2-user@ip-172-31-47-20 ~]$
```

5. Enable Apache to Start on Boot:

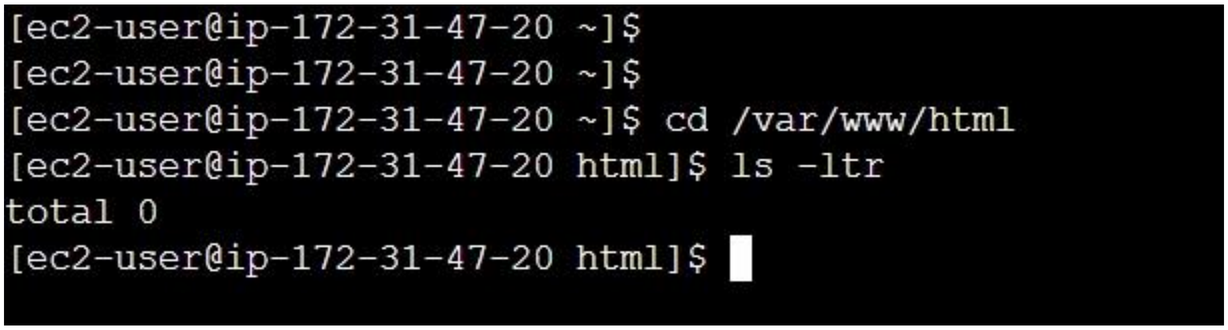
   sudo chkconfig httpd on



```
[ec2-user@ip-172-31-47-20 ~]$
[ec2-user@ip-172-31-47-20 ~]$
[ec2-user@ip-172-31-47-20 ~]$ sudo chkconfig httpd on
Note: Forwarding request to 'systemctl enable httpd.service'.
Created symlink from /etc/systemd/system/multi-user.target.wants/httpd.service to /usr/lib/systemd/system/http
d.service.
[ec2-user@ip-172-31-47-20 ~]$
```

6. Navigate to /var/www/html/ directory:

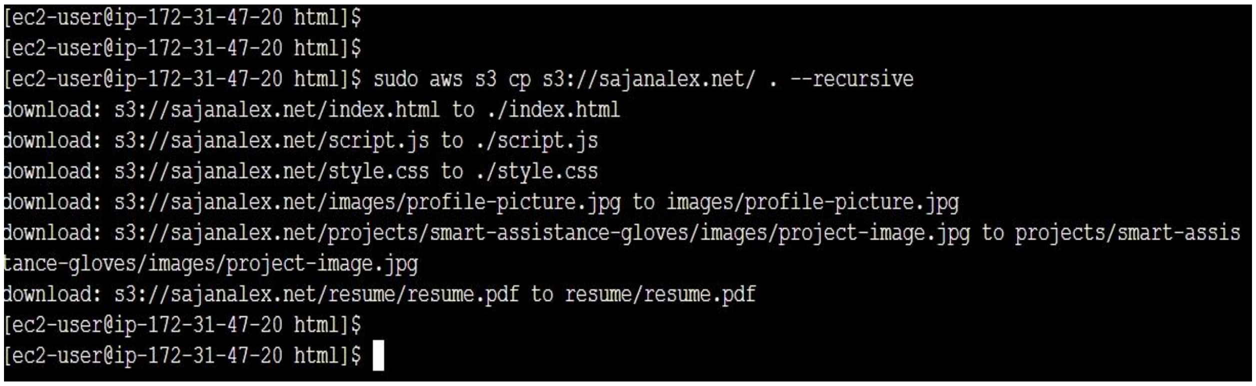
   cd /var/www/html



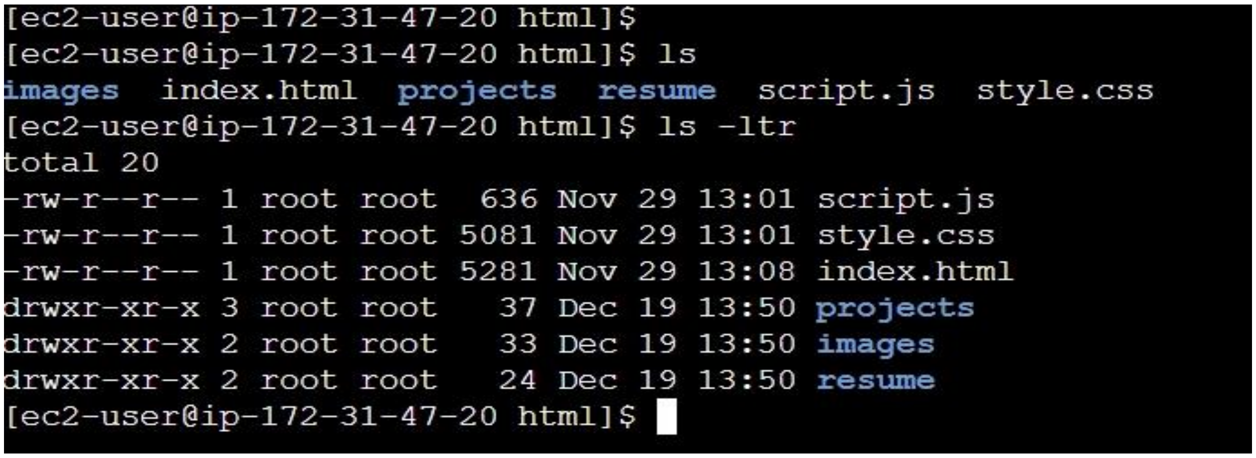
```
[ec2-user@ip-172-31-47-20 ~]$
[ec2-user@ip-172-31-47-20 ~]$
[ec2-user@ip-172-31-47-20 ~]$ cd /var/www/html
[ec2-user@ip-172-31-47-20 html]$ ls -ltr
total 0
[ec2-user@ip-172-31-47-20 html]$
```

7. Copy the buckets objects from the S3 bucket to the html directory:

   sudo aws s3 cp s3://BUCKET-NAME/ . –recursive



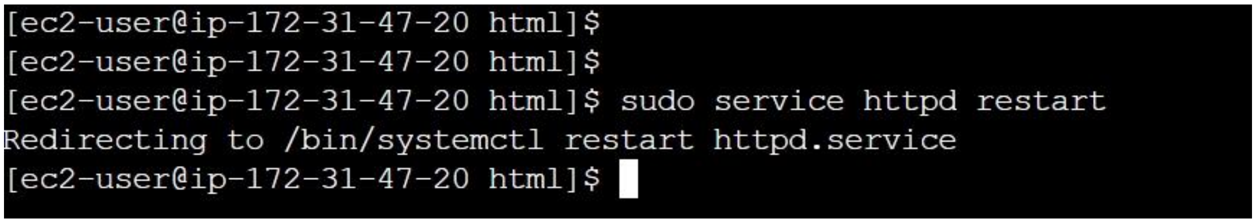
```
[ec2-user@ip-172-31-47-20 html]$
[ec2-user@ip-172-31-47-20 html]$
[ec2-user@ip-172-31-47-20 html]$ sudo aws s3 cp s3://sajanalex.net/ . --recursive
download: s3://sajanalex.net/index.html to ./index.html
download: s3://sajanalex.net/script.js to ./script.js
download: s3://sajanalex.net/style.css to ./style.css
download: s3://sajanalex.net/images/profile-picture.jpg to images/profile-picture.jpg
download: s3://sajanalex.net/projects/smart-assistance-gloves/images/project-image.jpg to projects/smart-assis
tance-gloves/images/project-image.jpg
download: s3://sajanalex.net/resume/resume.pdf to resume/resume.pdf
[ec2-user@ip-172-31-47-20 html]$
[ec2-user@ip-172-31-47-20 html]$
```



```
[ec2-user@ip-172-31-47-20 html]$
[ec2-user@ip-172-31-47-20 html]$ ls
images  index.html  projects  resume  script.js  style.css
[ec2-user@ip-172-31-47-20 html]$ ls -ltr
total 20
-rw-r--r-- 1 root root  636 Nov 29 13:01 script.js
-rw-r--r-- 1 root root 5081 Nov 29 13:01 style.css
-rw-r--r-- 1 root root 5281 Nov 29 13:08 index.html
drwxr-xr-x 3 root root   37 Dec 19 13:50 projects
drwxr-xr-x 2 root root   33 Dec 19 13:50 images
drwxr-xr-x 2 root root   24 Dec 19 13:50 resume
[ec2-user@ip-172-31-47-20 html]$
```

8. Restart the Apache Service:

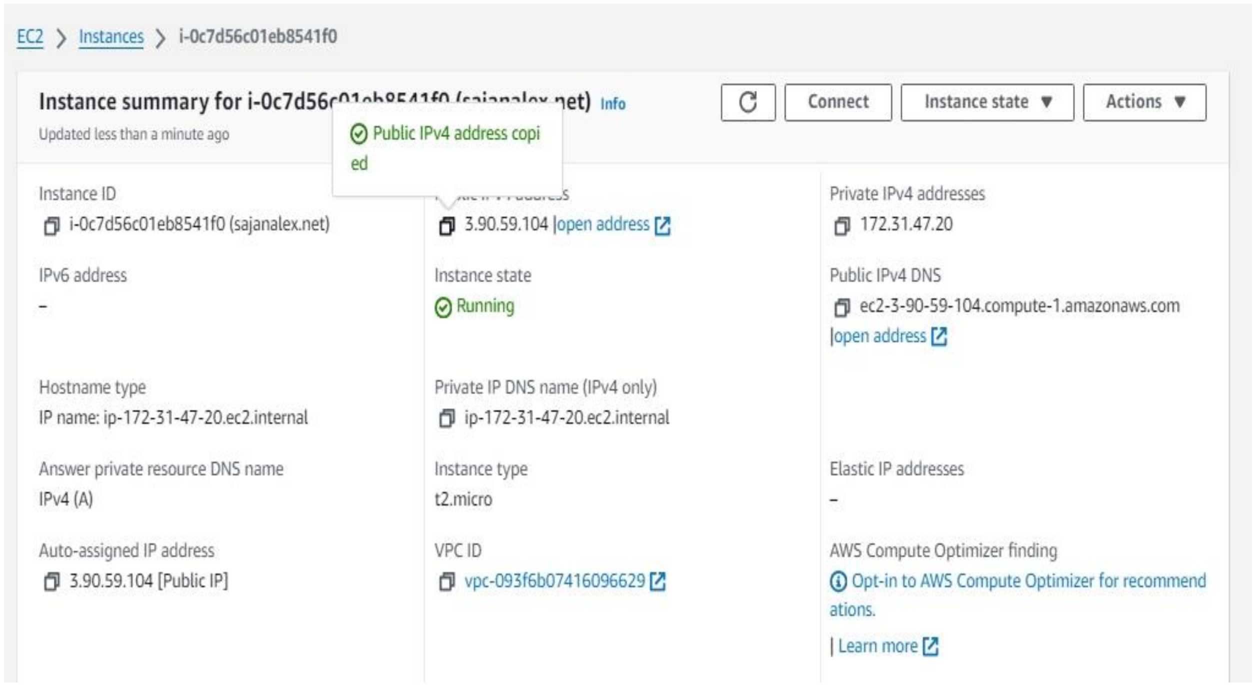
   sudo service httpd restart



```
[ec2-user@ip-172-31-47-20 html]$
[ec2-user@ip-172-31-47-20 html]$
[ec2-user@ip-172-31-47-20 html]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-47-20 html]$
```

9. Verify Installation:

   - Open a web browser and enter instance's public IP or DNS. You should see the Personal-Portfolio site.



EC2 > Instances > i-0c7d56c01eb8541f0

Instance summary for i-0c7d56c01eb8541f0 (sajanalex.net) Info

Updated less than a minute ago

Public IPv4 address copied

Instance ID
i-0c7d56c01eb8541f0 (sajanalex.net)

Public IPv4 address
3.90.59.104 |open address

Private IPv4 addresses
172.31.47.20

IPv6 address
–

Instance state
Running

Public IPv4 DNS
ec2-3-90-59-104.compute-1.amazonaws.com |open address

Hostname type
IP name: ip-172-31-47-20.ec2.internal

Private IP DNS name (IPv4 only)
ip-172-31-47-20.ec2.internal

Answer private resource DNS name
IPv4 (A)

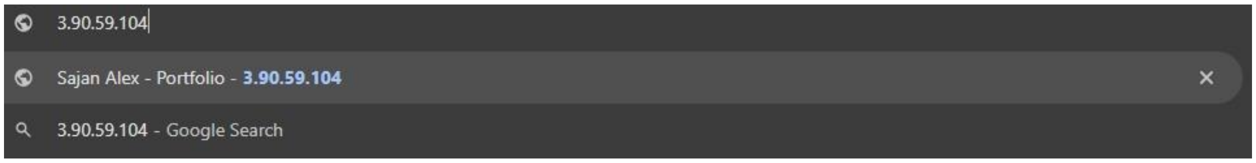
Instance type
t2.micro

Elastic IP addresses
–

Auto-assigned IP address
3.90.59.104 [Public IP]

VPC ID
vpc-093f6b07416096629

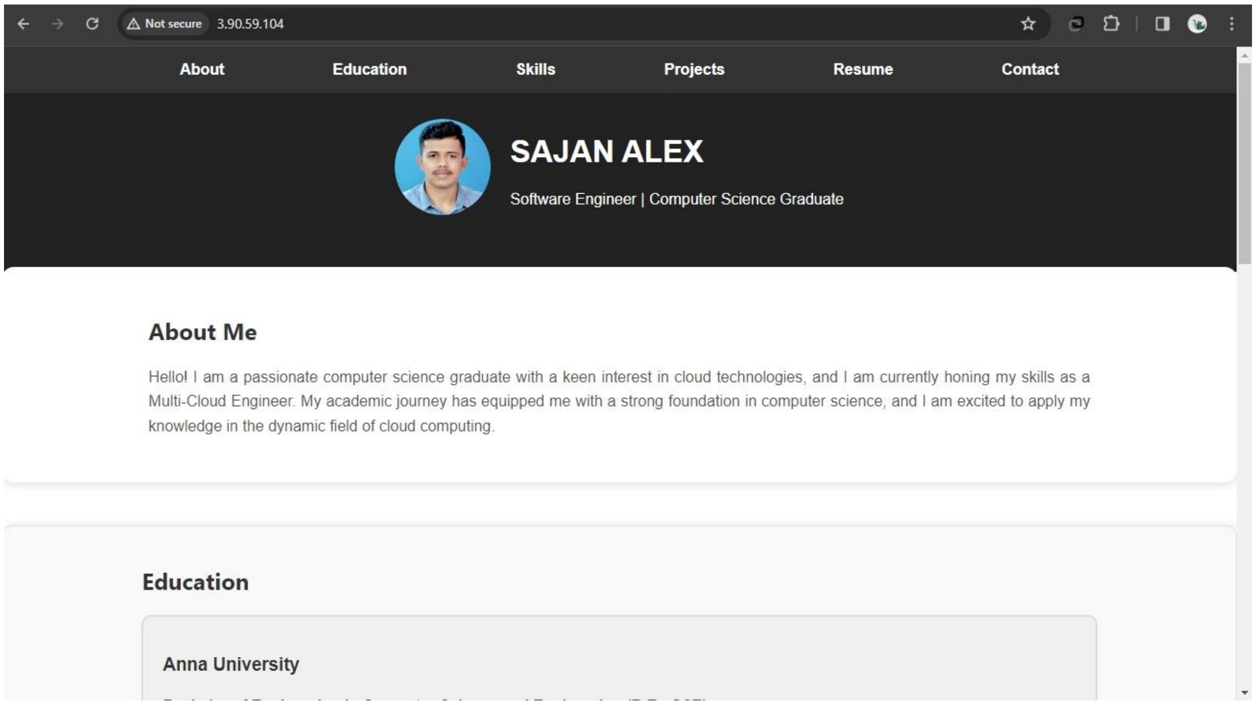
AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations.
| Learn more



3.90.59.104

Sajan Alex - Portfolio - 3.90.59.104

3.90.59.104 - Google Search


Not secure 3.90.59.104

About Education Skills Projects Resume Contact

SAJAN ALEX

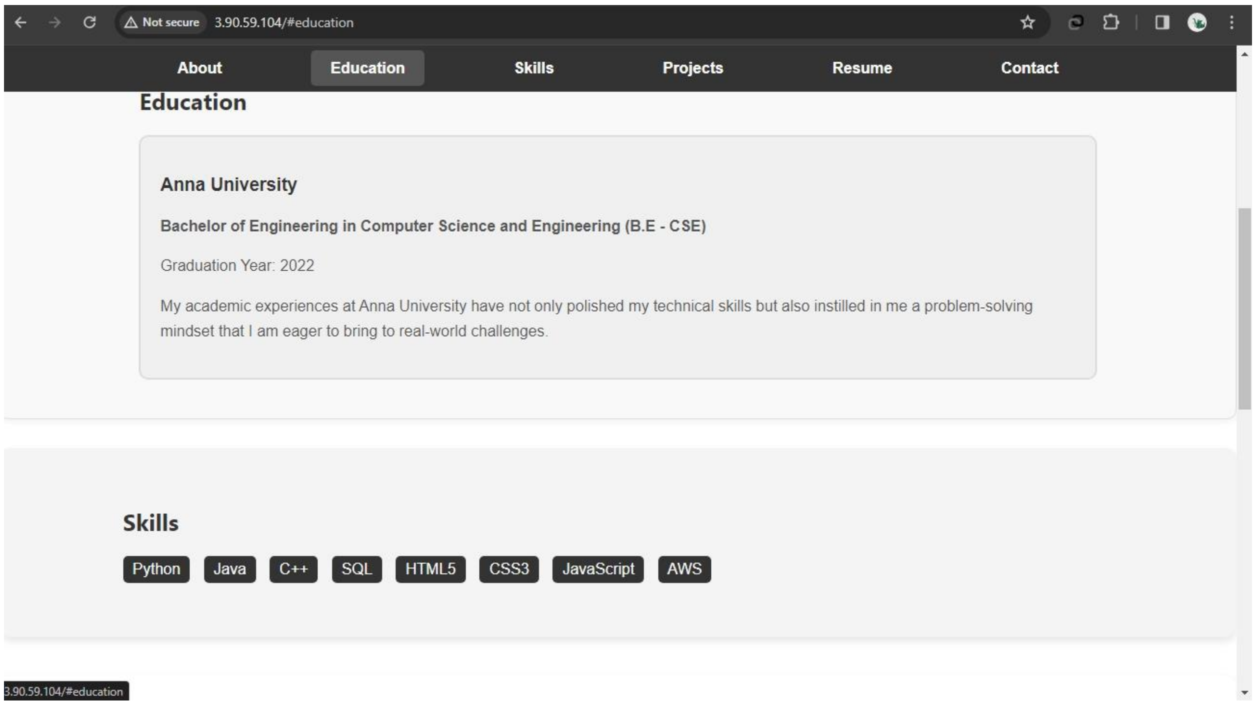
Software Engineer | Computer Science Graduate

About Me

Hello! I am a passionate computer science graduate with a keen interest in cloud technologies, and I am currently honing my skills as a Multi-Cloud Engineer. My academic journey has equipped me with a strong foundation in computer science, and I am excited to apply my knowledge in the dynamic field of cloud computing.

Education

Anna University


Not secure 3.90.59.104/#education

About Education Skills Projects Resume Contact

Education

Anna University

Bachelor of Engineering in Computer Science and Engineering (B.E - CSE)

Graduation Year: 2022

My academic experiences at Anna University have not only polished my technical skills but also instilled in me a problem-solving mindset that I am eager to bring to real-world challenges.

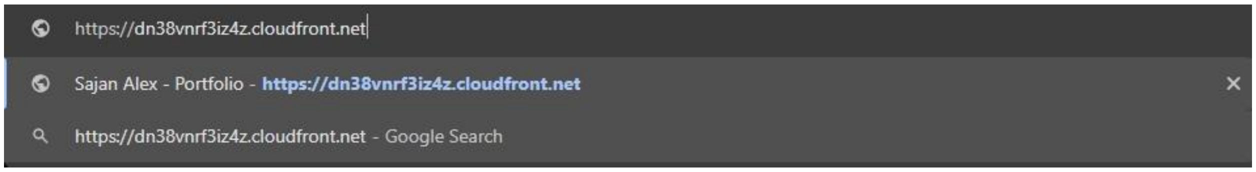
Skills

Python Java C++ SQL HTML5 CSS3 JavaScript AWS

3.90.59.104/#education

6.7 Website Functionality Testing

1. Access the Website:
   - Open a web browser.
   - Enter your custom domain or CloudFront distribution URL.

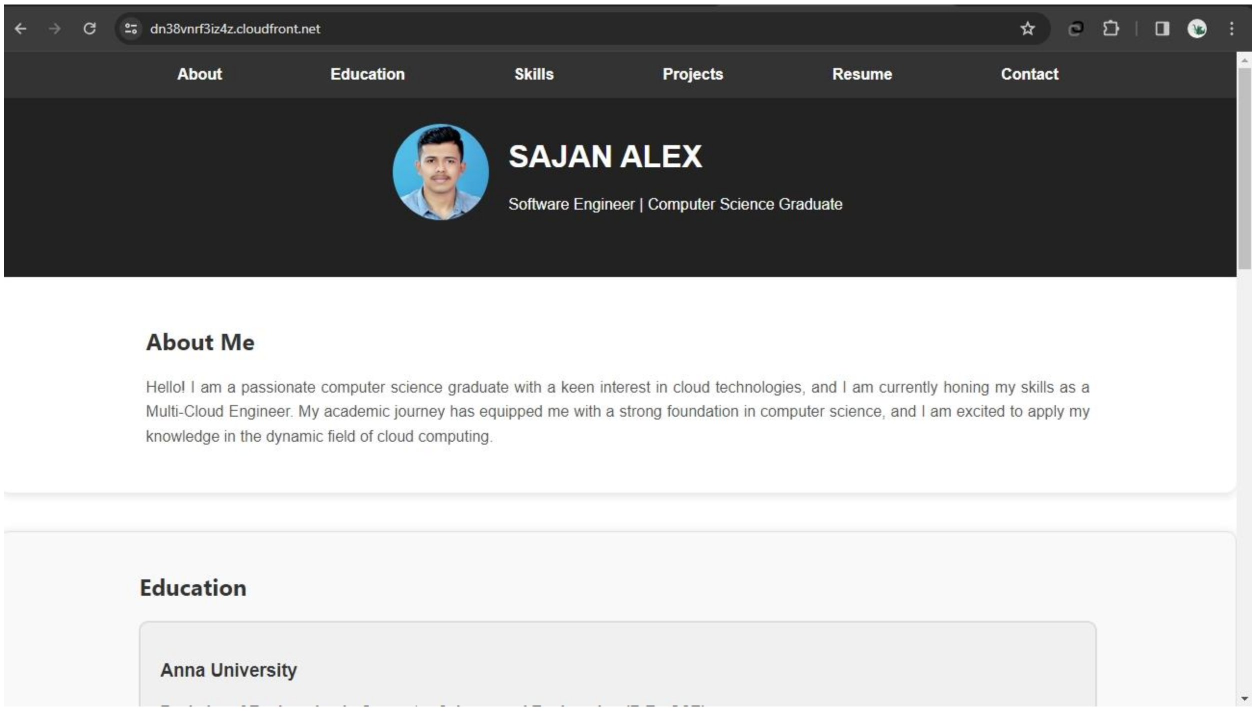


https://dn38vnrf3iz4z.cloudfront.net

Sajan Alex - Portfolio - https://dn38vnrf3iz4z.cloudfront.net

https://dn38vnrf3iz4z.cloudfront.net - Google Search

2. Homepage Verification:
   - Confirm that the homepage loads without errors.
   - Verify that all essential elements, such as navigation links and the header, are displayed correctly.



dn38vnrf3iz4z.cloudfront.net

About Education Skills Projects Resume Contact

SAJAN ALEX

Software Engineer | Computer Science Graduate

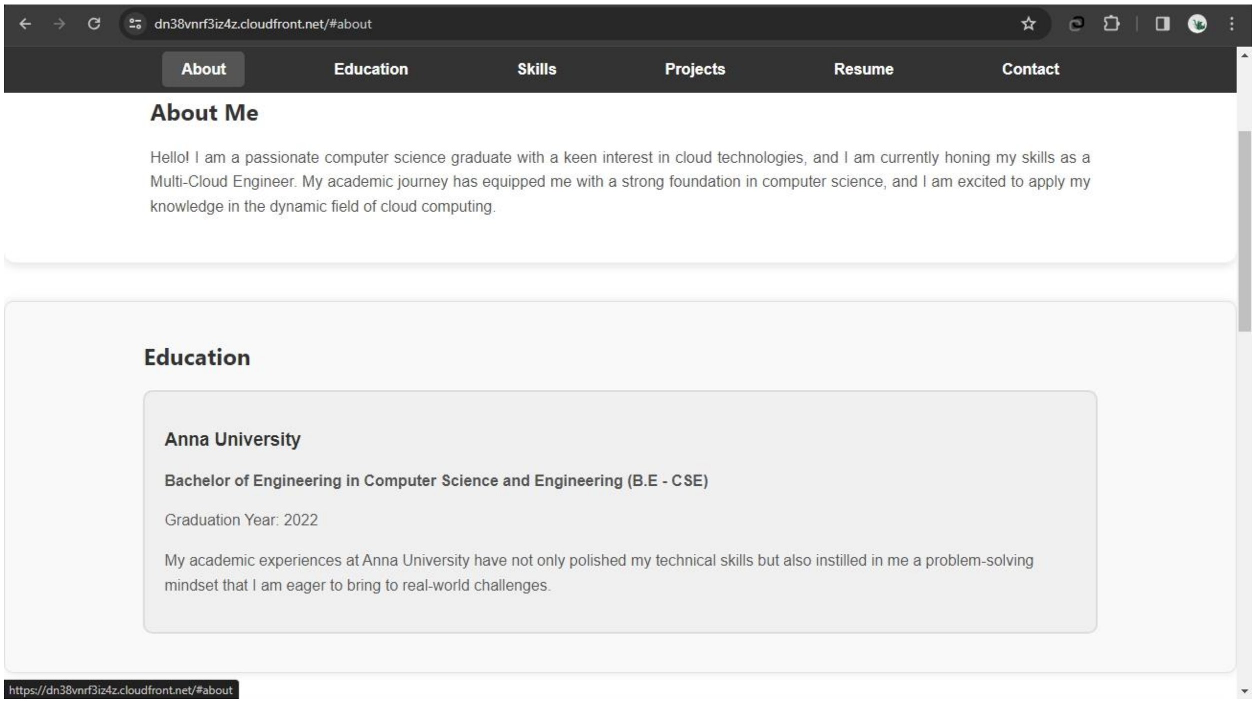
About Me

Hello! I am a passionate computer science graduate with a keen interest in cloud technologies, and I am currently honing my skills as a Multi-Cloud Engineer. My academic journey has equipped me with a strong foundation in computer science, and I am excited to apply my knowledge in the dynamic field of cloud computing.

Education

Anna University

3. Navigation Testing:
   - Click on each navigation link.
   - Ensure that each section or page loads as expected.
   - Check for smooth transitions and responsiveness.



About Me

Hello! I am a passionate computer science graduate with a keen interest in cloud technologies, and I am currently honing my skills as a Multi-Cloud Engineer. My academic journey has equipped me with a strong foundation in computer science, and I am excited to apply my knowledge in the dynamic field of cloud computing.

Education

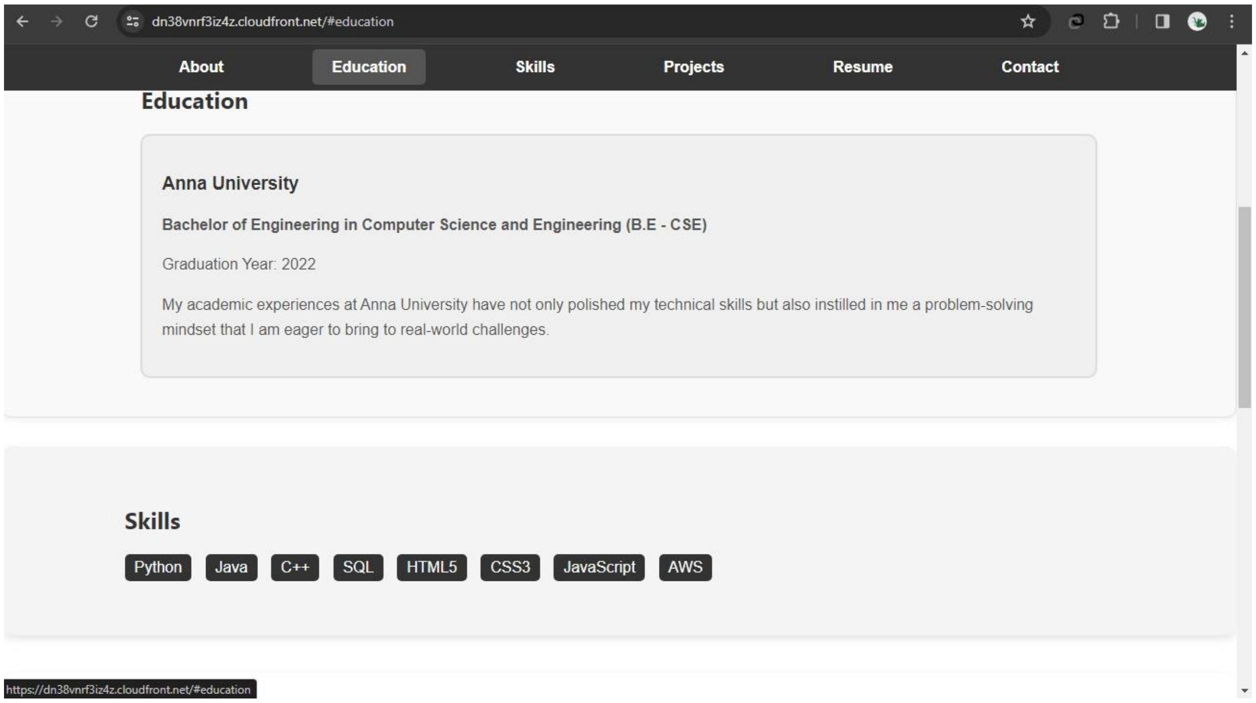
Anna University

Bachelor of Engineering in Computer Science and Engineering (B.E - CSE)

Graduation Year: 2022

My academic experiences at Anna University have not only polished my technical skills but also instilled in me a problem-solving mindset that I am eager to bring to real-world challenges.

https://dn38vnrf3iz4z.cloudfront.net/#about


dn38vnrf3iz4z.cloudfront.net/#education
About
Education
Skills
Projects
Resume
Contact

Education

Anna University

Bachelor of Engineering in Computer Science and Engineering (B.E - CSE)

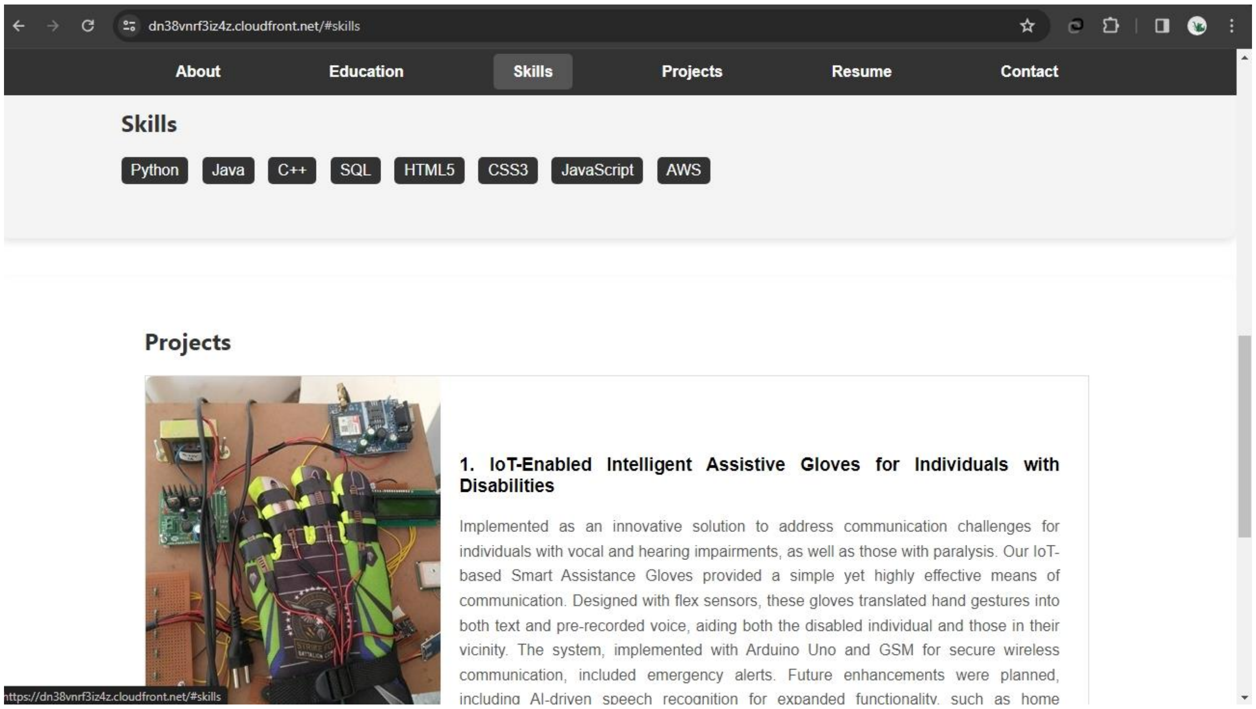
Graduation Year: 2022

My academic experiences at Anna University have not only polished my technical skills but also instilled in me a problem-solving mindset that I am eager to bring to real-world challenges.

Skills

Python Java C++ SQL HTML5 CSS3 JavaScript AWS

https://dn38vnrf3iz4z.cloudfront.net/#education



Skills

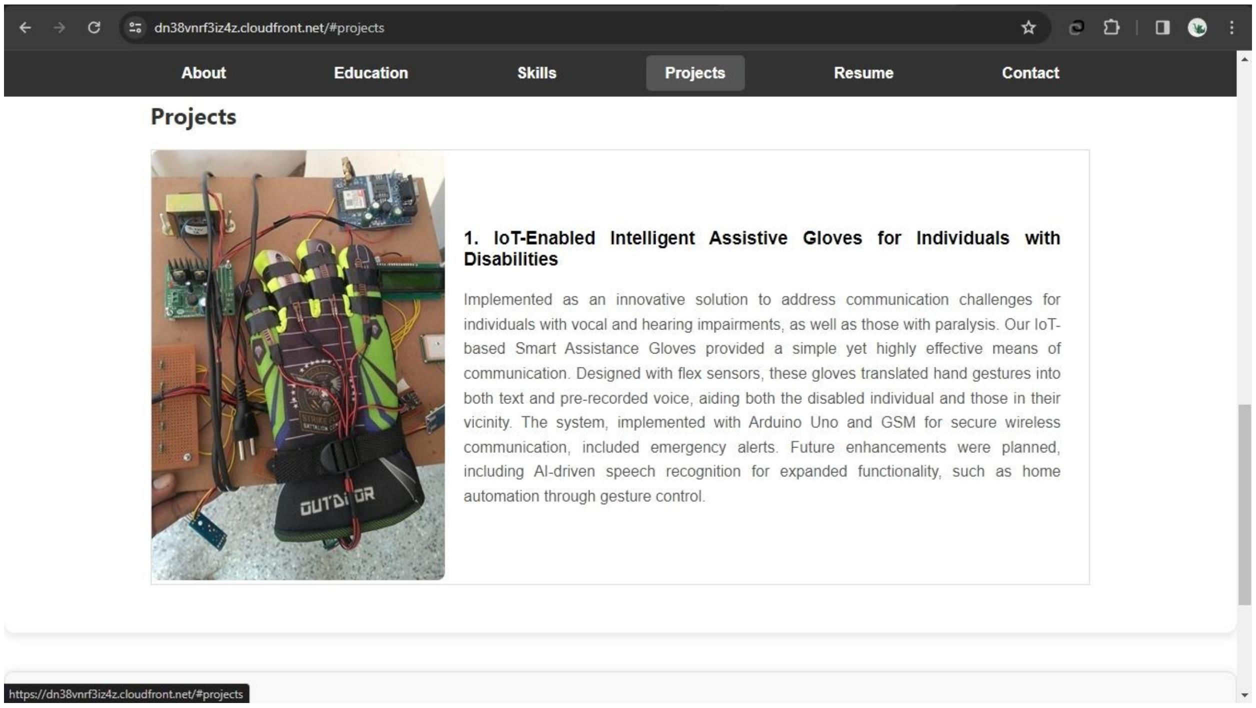
Python Java C++ SQL HTML5 CSS3 JavaScript AWS

Projects

1. IoT-Enabled Intelligent Assistive Gloves for Individuals with Disabilities

Implemented as an innovative solution to address communication challenges for individuals with vocal and hearing impairments, as well as those with paralysis. Our IoT-based Smart Assistance Gloves provided a simple yet highly effective means of communication. Designed with flex sensors, these gloves translated hand gestures into both text and pre-recorded voice, aiding both the disabled individual and those in their vicinity. The system, implemented with Arduino Uno and GSM for secure wireless communication, included emergency alerts. Future enhancements were planned, including AI-driven speech recognition for expanded functionality, such as home

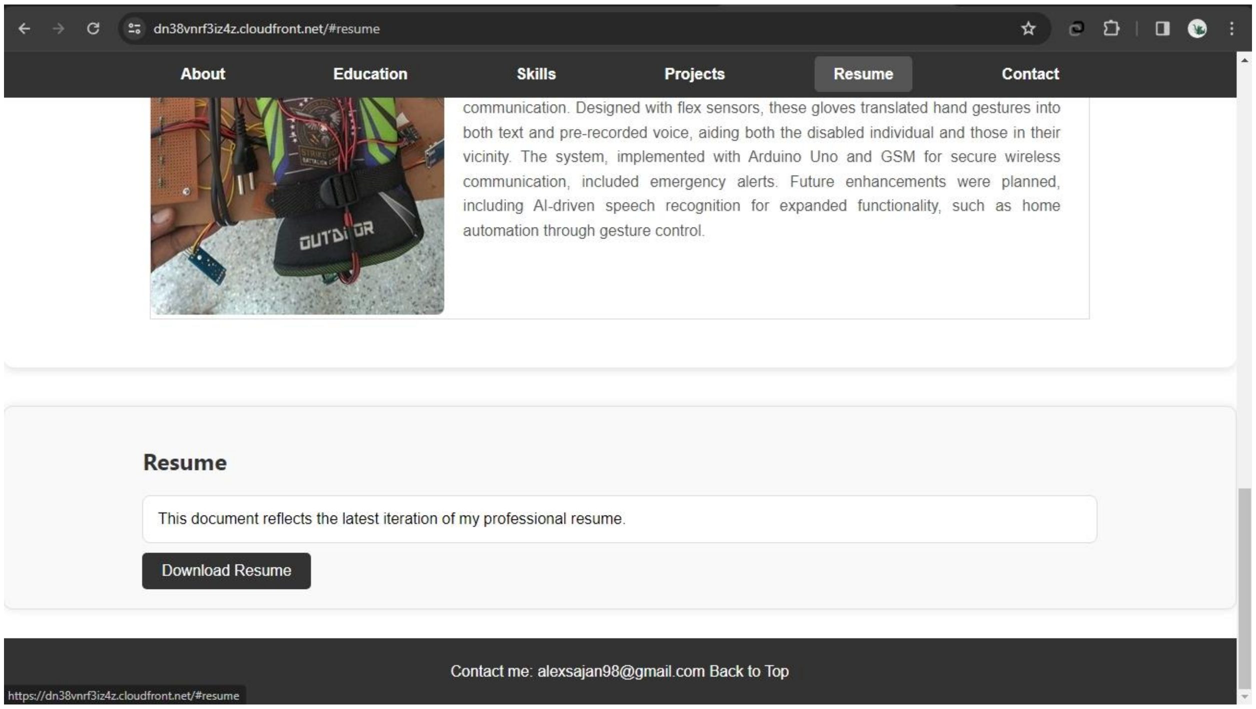
https://dn38vnrf3iz4z.cloudfront.net/#skills


dn38vnrf3iz4z.cloudfront.net/#projects
About
Education
Skills
Projects
Resume
Contact

Projects

1. IoT-Enabled Intelligent Assistive Gloves for Individuals with Disabilities

Implemented as an innovative solution to address communication challenges for individuals with vocal and hearing impairments, as well as those with paralysis. Our IoT-based Smart Assistance Gloves provided a simple yet highly effective means of communication. Designed with flex sensors, these gloves translated hand gestures into both text and pre-recorded voice, aiding both the disabled individual and those in their vicinity. The system, implemented with Arduino Uno and GSM for secure wireless communication, included emergency alerts. Future enhancements were planned, including AI-driven speech recognition for expanded functionality, such as home automation through gesture control.


dn38vnrf3iz4z.cloudfront.net/#resume

About Education Skills Projects Resume Contact

communication. Designed with flex sensors, these gloves translated hand gestures into both text and pre-recorded voice, aiding both the disabled individual and those in their vicinity. The system, implemented with Arduino Uno and GSM for secure wireless communication, included emergency alerts. Future enhancements were planned, including AI-driven speech recognition for expanded functionality, such as home automation through gesture control.

Resume

This document reflects the latest iteration of my professional resume.

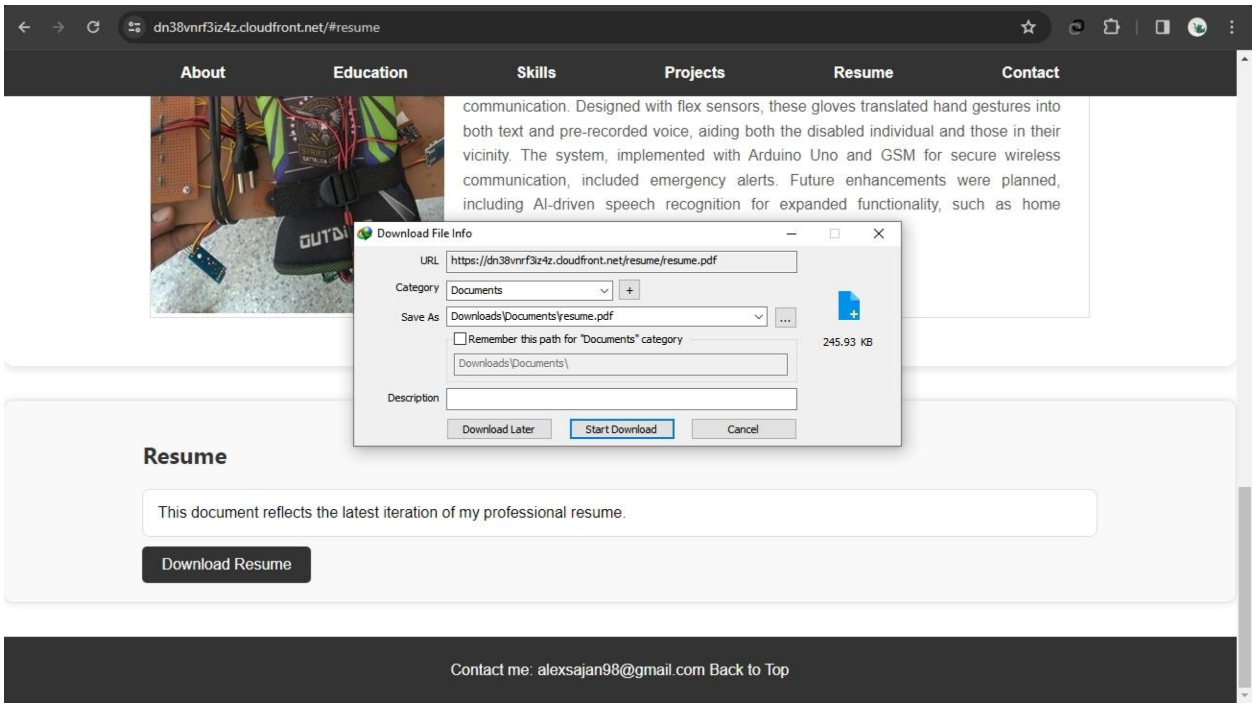
Download Resume

Contact me: alexsajan98@gmail.com Back to Top

https://dn38vnrf3iz4z.cloudfront.net/#resume

4. Content Validation:

- Review the content on each page.
- Check for accurate text, images, and multimedia elements.
- Ensure that content is up-to-date and reflects the intended information.


dn38vnrf3iz4z.cloudfront.net/#resume

About Education Skills Projects Resume Contact

communication. Designed with flex sensors, these gloves translated hand gestures into both text and pre-recorded voice, aiding both the disabled individual and those in their vicinity. The system, implemented with Arduino Uno and GSM for secure wireless communication, included emergency alerts. Future enhancements were planned, including AI-driven speech recognition for expanded functionality, such as home

Download File Info

URL: https://dn38vnrf3iz4z.cloudfront.net/resume/resume.pdf

Category: Documents

Save As: Downloads\Documents\resume.pdf

Remember this path for "Documents" category

Downloads\Documents\

245.93 KB

Description:

Download Later | Start Download | Cancel

Resume

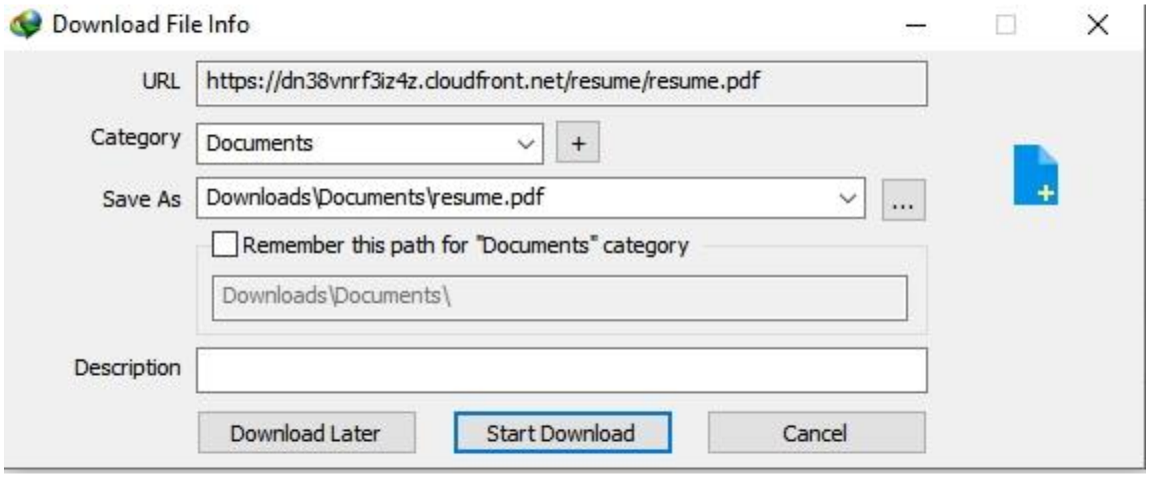
This document reflects the latest iteration of my professional resume.

Download Resume

Contact me: alexsajan98@gmail.com Back to Top

5. Interactive Elements:

    - If your website includes interactive elements (forms, buttons, etc.), test their functionality.



Download File Info

URL: https://dn38vnrf3iz4z.cloudfront.net/resume/resume.pdf

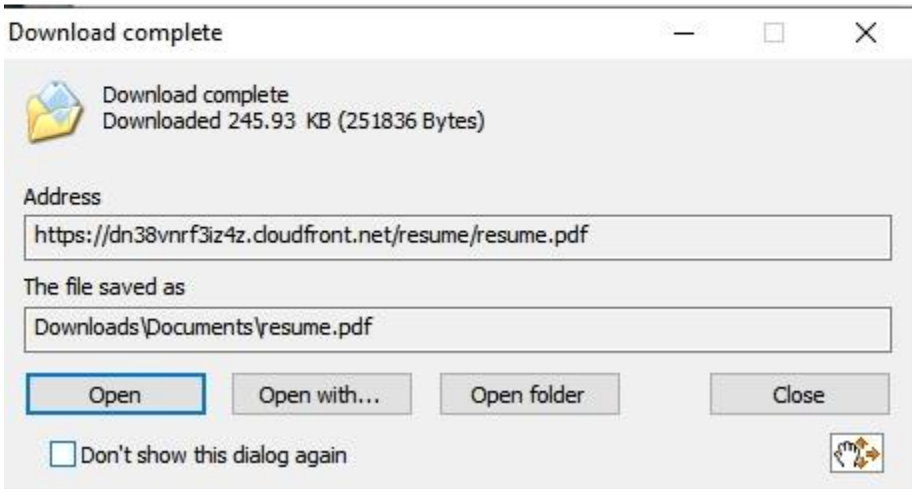
Category: Documents

Save As: Downloads\Documents\resume.pdf

[ ] Remember this path for "Documents" category

Downloads\Documents\

Description:



Download complete

Download complete
Downloaded 245.93 KB (251836 Bytes)

Address: https://dn38vnrf3iz4z.cloudfront.net/resume/resume.pdf

The file saved as: Downloads\Documents\resume.pdf

[ ] Don't show this dialog again

    - Submit forms and verify data submission if applicable.

6. Responsiveness:

    - Test the website on various devices (desktop, tablet, mobile).
    - Confirm that the layout adjusts appropriately for different screen sizes.

7. Browser Compatibility:

    - Test the website on multiple browsers (Chrome, Firefox, Safari, Edge, etc.).
    - Ensure consistent performance and appearance across browsers.

8. Error Pages:

    - Trigger intentional errors (e.g., access a non-existing page).

- Verify that the appropriate error pages (404, etc.) are displayed.

9. External Links:
   - If your website includes external links, test their functionality.
   - Confirm that external content opens in a new tab or window.
10. Forms and Interactivity:
    - Test any forms or interactive features on the website.
    - Check for validation messages and error handling.
11. Performance Testing:
    - Assess the website's loading speed.
    - Use tools like Google PageSpeed Insights to analyse and optimize performance.
12. Mobile Emulation:
    - If possible, use browser developer tools to emulate various mobile devices.
    - Ensure a smooth and visually appealing experience on mobile screens.
13. Contact Forms (if applicable):
    - If the website includes contact forms, submit test inquiries.
    - Verify that form submissions are received and processed correctly.
14. Security Checks:
    - Perform security checks to identify vulnerabilities.
    - Ensure that SSL/TLS is correctly configured for secure communication.

6.8 Security Measures Implementation

1. SSL/TLS Configuration:
    - Enable SSL/TLS for your CloudFront distribution.
    - Use AWS Certificate Manager (ACM) to provision SSL/TLS certificates for your custom domain.
2. Access Controls for S3:
    - Review and configure access controls for your S3 bucket.
    - Implement the principle of least privilege, granting minimal permissions necessary.
    - Leverage AWS Identity and Access Management (IAM) for user and role-based access controls.
3. CloudFront Security:
    - Utilize CloudFront signed URLs or cookies to control access to your content.
    - Implement Geo-Restrictions if applicable, limiting access to specific geographic locations.
4. AWS WAF (Web Application Firewall):
    - Set up AWS WAF to filter and monitor HTTP traffic to your CloudFront distribution.
    - Create rules to block common web exploits and secure against common attack patterns.
5. Logging and Monitoring:
    - Enable AWS CloudTrail to log AWS API calls.
    - Use Amazon CloudWatch for monitoring and set up alarms for security-related events.
6. Regular Software Updates:
    - Keep your EC2 instance's operating system and web server software up-to-date.
    - Schedule regular patching and updates to address security vulnerabilities.
7. Firewall Configuration:

- Configure security groups for your EC2 instance, limiting inbound and outbound traffic.
- Implement Network Access Control Lists (NACLs) if additional network-level control is needed.

8. AWS Security Best Practices:
    - Adhere to AWS security best practices and recommendations.
    - Regularly review the AWS Well-Architected Framework for security considerations.
9. DDoS Protection:
    - Leverage AWS Shield, a managed Distributed Denial of Service (DDoS) protection service.
    - Consider AWS WAF Rate-Based Rules to protect against application layer DDoS attacks.
10. Data Encryption:
    - Enable encryption for data at rest in your S3 bucket using Server-Side Encryption (SSE).
    - Implement encryption in transit by using SSL/TLS for data transfer.
11. Backup and Restore Procedures:
    - Establish backup and restore procedures for critical components.
    - Regularly backup configurations, certificates, and other essential data.
12. Incident Response Plan:
    - Develop an incident response plan outlining steps to take in case of a security incident.
    - Establish communication and coordination procedures with your team.
13. Periodic Security Audits:
    - Conduct periodic security audits and vulnerability assessments.
    - Use AWS Inspector or other security scanning tools to identify potential vulnerabilities.
14. Multi-Factor Authentication (MFA):
    - Enable MFA for AWS account users.
    - Enforce MFA for sensitive operations and access points.

6.9 Cost Optimization Strategies

1. Rightsized EC2 Instances:
    - Analyse the resource requirements of your web server.
    - Choose the right EC2 instance type based on your website's workload.
    - Utilize AWS tools like AWS Compute Optimizer to identify optimal instance types.
2. Utilize Reserved Instances (RIs) and Savings Plans:
    - Leverage Reserved Instances for steady-state workloads with predictable usage.
    - Explore AWS Savings Plans for flexibility across a variety of services.
3. S3 Storage Optimization:
    - Optimize S3 storage classes based on access patterns.
    - Utilize Amazon S3 Lifecycle policies to transition objects between storage classes.
4. Monitor and Analyse Resource Utilization:
    - Use Amazon CloudWatch to monitor resource utilization.
    - Set up CloudWatch Alarms to receive notifications on predefined thresholds.
5. Implement Auto Scaling:
    - Configure Auto Scaling groups for your EC2 instances.
    - Adjust minimum and maximum instance counts based on demand.
6. Effective Caching with CloudFront:
    - Configure CloudFront caching settings to balance performance and cost.
    - Leverage CloudFront cache behaviours and TTL settings.
7. Spot Instances for Burst Workloads:
    - Utilize EC2 Spot Instances for burst workloads and non-critical tasks.
    - Combine Spot Instances with On-Demand or Reserved Instances for cost-effectiveness.
8. Use AWS Lambda for Serverless Operations:
    - Explore AWS Lambda for serverless functions and operations.

- Optimize the size and runtime of Lambda functions.

9. Monitor and Analyse Data Transfer Costs:
    - Keep an eye on data transfer costs, especially for data transfer out.
    - Optimize content delivery with efficient use of CloudFront.
10. AWS Trusted Advisor:
    - Leverage AWS Trusted Advisor for cost optimization recommendations.
    - Regularly review the Trusted Advisor Cost Optimization checks.
11. Tagging for Cost Allocation:
    - Implement resource tagging for accurate cost allocation.
    - Use tags to categorize resources based on function, environment, or project.
12. Use AWS Cost Explorer:
    - Analyse and visualize costs with AWS Cost Explorer.
    - Identify trends, anomalies, and opportunities for optimization.
13. Review and Modify Reserved Instances:
    - Regularly review and modify Reserved Instances based on evolving requirements.
    - Consider converting Convertible RIs for increased flexibility.
14. Remove Unused Resources:
    - Identify and decommission unused or underutilized resources.
    - Review AWS billing reports for insights into resource utilization.
15. Cost-Aware Development Practices:
    - Foster a cost-aware culture within your development team.
    - Incorporate cost considerations into the development lifecycle.

6.10 Monitoring and Maintenance

6.10.1 Monitoring Steps:

1. CloudWatch Metrics Setup:
    - Utilize AWS CloudWatch to set up custom metrics for monitoring key aspects.

- Create CloudWatch Alarms to receive notifications for specific thresholds.

2. Logging and Analysis:
   - Enable logging on CloudFront and S3 to capture access logs.
   - Analyse logs regularly to gain insights into user behaviour and troubleshoot issues.
3. Real-Time Monitoring:
   - Leverage CloudWatch Real-Time Metrics (RTM) to monitor the website's performance.
   - Set up dashboards to visualize critical metrics.
4. Error Monitoring:
   - Implement error monitoring using CloudWatch Alarms.
   - Investigate and resolve any increase in error rates promptly.
5. Incident Response Plan:
   - Develop an incident response plan outlining procedures for addressing unexpected issues.
   - Assign responsibilities and escalation paths for incident resolution.

6.10.2 Maintenance Steps:

1. Regular Updates:
   - Periodically update the website content to reflect changes or additions.
   - Ensure that the latest versions of libraries, frameworks, and dependencies are used.
2. Security Patching:
   - Regularly check for security patches and updates for the web server and other software.
   - Apply patches promptly to mitigate vulnerabilities.
3. Backup and Restore:
   - Implement regular backups of S3 content, configuration settings, and any critical data.
   - Practice restoration procedures to ensure data recovery capabilities.
4. SSL/TLS Certificate Renewal:

- Monitor SSL/TLS certificate expiration dates.
- Renew certificates before they expire to prevent disruptions in HTTPS service.

5. Scale Resources Appropriately:
    - Monitor website traffic and scale resources (e.g., EC2 instances) as needed.
    - Adjust CloudFront caching settings for optimal performance.
6. Cost Optimization:
    - Continuously review AWS costs and optimize resource usage.
    - Explore reserved instances or Savings Plans for cost savings.
7. Content Delivery Optimization:
    - Optimize images and other static content for efficient delivery.
    - Consider using Content Delivery Networks (CDNs) for improved global performance.
8. Regular Testing:
    - Conduct periodic tests, including load testing and security assessments.
    - Address any identified issues and refine the website accordingly.
9. Documentation Updates:
    - Keep project documentation up-to-date with any changes or optimizations made.
    - Document any lessons learned from monitoring and maintenance activities.
10. Periodic Review of Security Measures:
    - Review and update security measures, including access controls and permissions.
    - Stay informed about emerging security threats and apply relevant measures.

7. Challenges and Solutions

7.1 Challenges Faced

The development and deployment of the static website presented several challenges that required strategic solutions for successful implementation. Key challenges encountered during the project included:

1. **Infrastructure Complexity:**
    - Configuring and managing the infrastructure in a non-cloud environment posed challenges in terms of scalability, availability, and efficient resource utilization.
2. **Manual Security and Updates:**
    - Manual handling of security measures and updates introduced potential vulnerabilities and increased the risk of overlooking critical patches.
3. **Cost and Resource Management:**
    - Efficiently managing costs and resources, including hardware and software, was a significant concern in a non-cloud setup.
4. **Device and Browser Compatibility:**
    - Ensuring consistent functionality and visual appeal across diverse devices and browsers required meticulous testing and optimization.
5. **Limited Scalability Options:**
    - The lack of scalable options in a non-cloud environment restricted the website's ability to handle increased traffic or adapt to changing demands.

7.2 Solutions Implemented

To address these challenges, the project incorporated a range of solutions aimed at enhancing infrastructure management, security, scalability, cost efficiency, and cross-device compatibility:

1. **Adoption of AWS Cloud Services:**
    - Leveraged AWS Cloud services such as EC2, S3, and CloudFront for improved infrastructure scalability, high availability, and simplified management.
2. **Automation of Security Measures:**
    - Implemented automated security measures, including regular updates and patch management, to enhance system security and reduce manual intervention.

3. **Cost Optimization Strategies:**
   - Employed cost optimization strategies by selecting appropriate AWS instance types, storage options, and caching settings to ensure efficient resource utilization.
4. **Responsive Design and Browser Testing:**
   - Applied responsive design principles and conducted thorough browser compatibility testing to ensure a seamless user experience across various devices and browsers.
5. **Scalability with CloudFront:**
   - Configured CloudFront to serve content from an S3 bucket, enabling efficient content delivery, global availability, and enhanced scalability.

By strategically implementing these solutions, the project successfully mitigated challenges associated with infrastructure, security, cost management, and cross-device compatibility, resulting in a robust and optimized static website deployment.

8. Lessons Learned

8.1 Continuous Learning and Adaptation

The project journey provided invaluable insights and lessons that contributed to a deeper understanding of cloud technologies, infrastructure management, and best practices in web development. Key lessons learned include:

1. **Agility and Adaptability:**
    - Embraced the importance of staying agile and adaptable in the face of evolving project requirements, technology advancements, and unforeseen challenges.
2. **Cloud Advantages:**
    - Acknowledged the significant advantages offered by cloud services, such as scalability, cost efficiency, and streamlined management, compared to traditional non-cloud environments.
3. **Security Automation:**
    - Recognized the necessity of incorporating automated security measures to enhance the overall security posture, reduce vulnerabilities, and ensure timely updates.
4. **Optimization Strategies:**
    - Learned to employ effective cost optimization strategies, including thoughtful selection of AWS resources and configurations, to achieve a balance between performance and cost.
5. **Cross-Device Compatibility:**
    - Emphasized the importance of prioritizing responsive design and rigorous testing across various devices and browsers to deliver a consistent and user-friendly experience.
6. **Documentation Importance:**
    - Underlined the critical role of comprehensive documentation in facilitating smooth project implementation, knowledge sharing, and troubleshooting.
7. **Global Content Delivery:**
    - Leveraged the advantages of CloudFront for global content delivery, minimizing latency, and ensuring a responsive experience for users worldwide.
8. **Monitoring and Maintenance:**

- Gained insights into the significance of continuous monitoring and proactive maintenance to identify potential issues, optimize performance, and ensure ongoing reliability.

9. **User-Centric Approach:**

    - Adopted a user-centric approach, focusing on delivering an optimized and dependable user experience by prioritizing responsiveness and usability.

These lessons have not only enhanced technical skills but also instilled a mindset of continuous learning, adaptability, and a commitment to delivering high-quality solutions in the ever-evolving landscape of web development and cloud computing.

9. Future Enhancements

9.1 Evolution Towards Excellence

The journey of developing and deploying the personal portfolio website marks a significant milestone, but the commitment to excellence and continuous improvement remains ongoing. Future enhancements are envisioned to elevate the website's capabilities, user experience, and content. Key areas for future development include:

1. **Mobile Responsiveness:**
    - Prioritizing further optimization for mobile devices to ensure a seamless and engaging experience for users across various screen sizes.
2. **Content Enrichment:**
    - Expanding the portfolio with additional projects, achievements, and case studies to provide a more comprehensive overview of skills, experiences, and contributions.
3. **Interactive Elements:**
    - Incorporating interactive elements, such as dynamic visuals, animations, or engaging user interfaces, to enhance user engagement and create a memorable browsing experience.
4. **Technology Showcase:**
    - Introducing a dedicated section to showcase proficiency in emerging technologies, tools, and frameworks, demonstrating a commitment to staying abreast of industry advancements.
5. **Blog or Insights Section:**
    - Establishing a blog or insights section to share thoughts, experiences, and expertise on relevant topics, contributing to professional visibility and industry knowledge dissemination.
6. **Personal Achievements:**
    - Highlighting personal and professional achievements, certifications, and milestones to convey continuous growth and expertise development.
7. **User Feedback Integration:**
    - Implementing a user feedback mechanism to gather insights, suggestions, and impressions, fostering a user-centric approach to refinement.

8. **Advanced Contact Options:**
    - Enhancing the contact section with additional communication channels, appointment scheduling features, or integrated messaging systems to facilitate seamless interactions.
9. **Accessibility Improvements:**
    - Conducting regular accessibility audits and implementing improvements to ensure the website is inclusive and complies with the latest accessibility standards.
10. **Performance Optimization:**
    - Continuously monitoring and optimizing website performance, incorporating best practices to ensure fast loading times and a smooth, responsive experience.
11. **Social Media Integration:**
    - Strengthening online presence by integrating social media platforms, allowing visitors to connect and engage through various channels.

The roadmap for future enhancements aligns with the commitment to delivering a website that not only reflects professional prowess but also evolves in tandem with changing requirements, industry trends, and user expectations. Through strategic development, thoughtful design, and a user-focused mindset, the website will continue to be a dynamic representation of achievements and aspirations.

10. Conclusion

In conclusion, the development and deployment of this personal portfolio website have been a journey of innovation, learning, and dedication. Through meticulous planning and strategic utilization of AWS services, including EC2, S3, and CloudFront, the project has successfully addressed challenges related to infrastructure, scalability, and security.

The achievement of a fully functioning static website, coupled with the integration of efficient content delivery mechanisms, reflects a commitment to excellence. The project not only establishes a robust online presence but also serves as a testament to the adept use of cloud technologies for web hosting.

As we move forward, the dynamic nature of technology and the ever-evolving landscape of web development inspire the anticipation of future enhancements. The commitment to continuous improvement, mobile responsiveness, content enrichment, and user engagement will drive the evolution of this portfolio into a comprehensive and cutting-edge representation of professional achievements.

In essence, this project documentation encapsulates the success of deploying a personal portfolio while laying the groundwork for ongoing advancements. It is not just a static representation but a living testament to adaptability, technological proficiency, and a commitment to excellence in the digital realm.

11. References

1. Amazon EC2 Documentation:
    - https://docs.aws.amazon.com/ec2/
2. Amazon S3 Documentation:
    - https://docs.aws.amazon.com/s3/
3. Amazon CloudFront Documentation:
    - https://docs.aws.amazon.com/cloudfront/
4. Amazon Route 53 Documentation:
    - https://docs.aws.amazon.com/route53/
5. AWS Best Practices:
    - https://aws.amazon.com/architecture/well-architected/
6. HTML5 Documentation:
    - https://html.spec.whatwg.org/
7. CSS3 Documentation:
    - https://www.w3.org/Style/CSS/specs.en.html
8. JavaScript Documentation:
    - https://developer.mozilla.org/en-US/docs/Web/JavaScript
9. Responsive Web Design Basics:
    - https://alistapart.com/article/responsive-web-design/
10. Web Content Accessibility Guidelines (WCAG):
    - https://www.w3.org/WAI/WCAG21/quickref/
11. YouTube Channels

12. Internet Blogs and Tutorials

13. Web Development Blogs

14. AWS Blogs and Case Studies