# **Portfolio Website Deployment Documentation**

## **Introduction**

Creating and deploying a personal portfolio website is an essential step to showcase your skills, achievements, and projects. Leveraging AWS services like Amazon S3 and CloudFront, I successfully built and deployed my portfolio site to ensure scalability, reliability, and global accessibility. This document details the process and the benefits of the deployment.

## **Development and Hosting Workflow**

### **1. Frontend Development**

To create a visually appealing and responsive portfolio:

* **Tools Used**: HTML, CSS, and JavaScript.
* **Features**:
  + A clean, responsive design for desktop and mobile users.
  + Interactive elements to engage visitors.

### **2. Amazon S3: Static Website Hosting (** [**http://surl.li/nupejz**](http://surl.li/nupejz) **)**

Amazon S3 (Simple Storage Service) was used to host the static files of the website.

#### **Steps:**

1. **Bucket Creation**:
   * Created a new S3 bucket with a unique name, e.g., sakshi-portfolio-bucket.
   * Enabled public access settings for hosting a static website.
2. **File Upload**:
   * Uploaded all static website files (HTML, CSS, JS, images, etc.) to the S3 bucket.
3. **Static Website Hosting Configuration**:
   * Enabled the "Static Website Hosting" feature.
   * Set the **index document** (e.g., index.html) and error document (e.g., error.html).
4. **Bucket Policy**:
   * Configured a bucket policy to allow public read access for website files.

#### **Why S3?**

* Scalable, cost-effective storage.
* High availability and durability.
* Easy setup for static website hosting.

### **3. Amazon CloudFront: Content Delivery Network (CDN)**

To ensure faster load times and secure global delivery, CloudFront was configured to serve the content from the S3 bucket.

#### **Steps:**

1. **Distribution Creation**:
   * Created a CloudFront distribution and set the S3 bucket as the origin.
   * Enabled caching to improve performance.
2. **Custom Domain Configuration**:
   * Added a custom domain to the distribution.
   * Linked the domain using an **AWS Certificate Manager (ACM)** certificate for HTTPS.
3. **Edge Locations**:
   * Leveraged CloudFront’s global edge network to reduce latency.
4. **Security**:
   * Configured HTTPS for secure connections.
   * Restricted bucket access to only allow requests from CloudFront.

#### **Why CloudFront?**

* Reduces latency with global edge locations.
* Ensures secure, reliable, and fast delivery.
* Scales automatically to handle traffic spikes.

## **Deployment Summary**

### **Steps Recap**

1. Built a responsive website using HTML, CSS, and JavaScript.
2. Uploaded the static files to an Amazon S3 bucket configured for static website hosting.
3. Created a CloudFront distribution for faster and secure content delivery.
4. Configured a custom domain with HTTPS for professional presentation.

## **Benefits of the Deployment**

1. **Scalability**: Amazon S3 automatically scales storage to handle large traffic volumes.
2. **Global Reach**: CloudFront ensures low latency and fast load times worldwide.
3. **Cost-Effectiveness**: Pay-as-you-go pricing for both S3 and CloudFront makes this setup affordable.
4. **Reliability**: AWS services provide high availability and redundancy.
5. **Security**: HTTPS and restricted access via CloudFront improve security.

## **Conclusion**

Deploying my portfolio website with AWS S3 and CloudFront has allowed me to showcase my work on a platform that is fast, reliable, and secure. This approach ensures an excellent user experience for visitors worldwide. I encourage others to explore AWS’s services for their deployment needs.