Development and Reflection Phase: Cloud Architecture for a Simple Webpage on AWS

Hosting a High-Availability, Low-Latency, and Scalable Restaurant Menu Webpage

Student: Mohamed Elzeini

Matriculation Number: 102303120

Date: 30.10.2024

Project Overview

- **Objective:**
  - To create a highly available webpage that is accessible globally and can scale to handle more visitors.
- **Project Goals:**
  - **Reliability:** Ensure the page is always available.
  - **Scalability:** Prepare the infrastructure to handle growth.
  - **Accessibility:** Global reach with low latency.
- **Visual Element:**
  - User Request → AWS Cloud → Webpage Response

Selected AWS Services

- **Amazon S3** :
  - Hosts static content, including HTML and images (AWS, 2024a).
  - Chosen for high availability, durability, and built-in scalability (AWS, 2024a).
- **Amazon CloudFront** :
  - Distributes content globally, leveraging edge locations (AWS, 2024b).
  - Improves latency and performance by caching content close to users (AWS, 2024b).
- **Note**: Amazon Route 53 was initially planned but removed for simplicity and cost efficiency, as Amazon CloudFront already handles the DNS functionality.

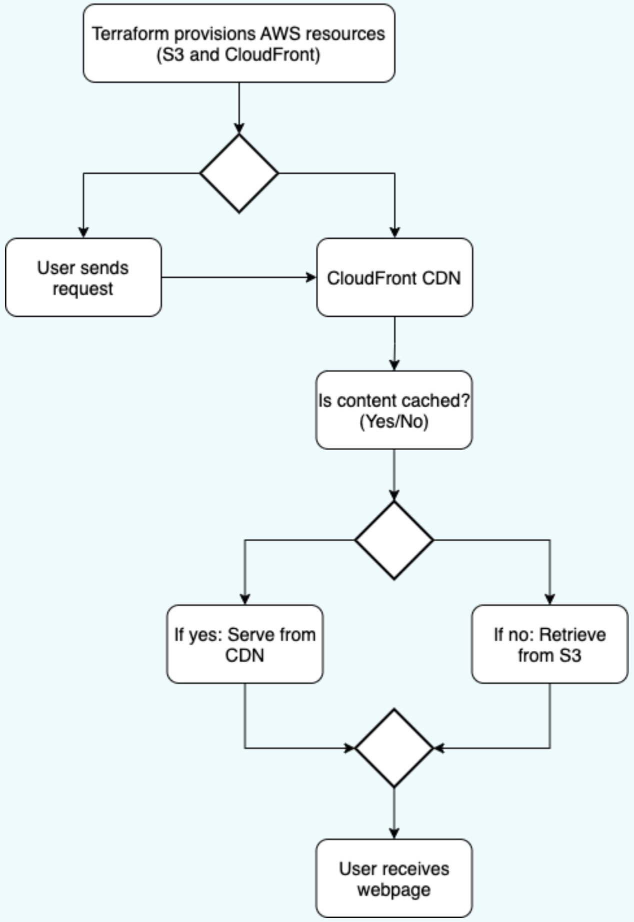
Infrastructure as Code (IaC) with Terraform

- **Terraform** :
  - Automates AWS resource configuration for consistency and scalability (HashiCorp Cloud Platform, n.d.).
  - Helps ensure **reproducible deployments** and reduces manual errors (HashiCorp Cloud Platform, n.d.).
- **Hyperlink to Terraform Documentation**:
  - Terraform Intro (HashiCorp Cloud Platform, n.d.).
- **Terraform Code for Infrastructure Setup:**
  - **Access the main.tf File**: GitHub - Mohamed Elzeini
- **Visual Element:**
  - Plan → Apply → Provisioned Resources.

Architecture Diagram

- **Diagram Structure**:
  - User → CloudFront (Global Distribution) → S3 Bucket (Static Content Storage).
- **Explanation**:
  - User requests content from CloudFront.
  - CloudFront serves cached content or fetches from S3 if it’s not cached (AWS, 2024b).

Architecture Diagram (Visual)



Terraform Configuration Explanation

- **Key Code Sections**:
  - **S3 Bucket**: Configured for hosting static website files, including index.html and images (AWS, 2024a).
  - **CloudFront Distribution**: Distributes content globally, configured with OAI to restrict direct access to S3 (AWS, 2024b).
  - **Outputs**: CloudFront URL is output for quick access to the webpage.
  - **Link to my webpage CloudFront URL**: CloudFront Webpage Domain
- **Code Examples**:
  - **S3 Configuration Snippet**:

    ```
    resource "aws_s3_bucket" "restaurant_menu_webpage" { ... }
    ```

  - **CloudFront Configuration Snippet**:

    ```
    resource "aws_cloudfront_distribution" "restaurant_menu_distribution" { ... }
    ```

Security Considerations

- **Origin Access Identity (OAI)**:
  - Restricts access to S3 bucket, allowing only CloudFront to access content (AWS, 2024c).
- **S3 Bucket Policy**:
  - Configured to deny public access and allow CloudFront only via OAI (AWS, 2024c).
- **HTTPS with CloudFront**:
  - Enforces secure data transmission using the default HTTPS certificate (AWS, 2024d).

Advantages and Improvements

- **Advantages**:
  - **High Availability**: S3 and CloudFront work together to ensure uptime (AWS, 2024a).
  - **Scalability**: Easily handles increased traffic without manual adjustments (AWS, 2024a).
- **Reflection and Improvement**:
  - **Removed Route 53**: Simplified architecture and reduced costs by using the CloudFront-provided URL directly.

Deployment and Testing

- **Deployment Steps**:
  - Use terraform apply to provision AWS resources.
  - Verify webpage accessibility through CloudFront URL.
- **Testing**:
  - Access the page from different geographic locations to ensure low-latency performance.
- **Link to CloudFront URL**:
  - CloudFront Webpage Domain
- **Link to the project on GitHub:**
  - Hosting a Simple Webpage on AWS

Conclusion and Future Scope

- **Summary**:
  - Successfully deployed a scalable, globally accessible webpage using S3 and CloudFront.
  - Terraform configuration ensures consistent, automated infrastructure setup.
- **Future Considerations**:
  - Potential to add a custom domain with Route 53 if desired.
  - Additional monitoring and logging could be added for enhanced observability.

References

- AWS. (2024a). What is Amazon S3? Amazon Web Services. Retrieved from https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html.
- AWS. (2024b). What is Amazon CloudFront? Amazon Web Services. Retrieved from https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/Introduction.html.
- AWS. (2024c). Restrict access to an Amazon Simple Storage Service origin. Amazon Web Services. Retrieved from https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/private-content-restricting-access-to-s3.html.
- AWS. (2024d). Require HTTPS for communication between CloudFront and your Amazon S3 origin. Amazon Web Services. Retrieved from https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/using-https-cloudfront-to-s3-origin.html.
- HashiCorp Cloud Platform. (n.d.). What is Terraform? HashiCorp Cloud Platform. Retrieved from https://developer.hashicorp.com/terraform/intro.