Challenge: Serving Dynamic HTML content using IaC

Author: Victor Ponce

Date: 01-28-2026

1. Requirements

A) Provision a service in AWS using Infrastructure as a Code

B) Serve an HTML page with the content “<h1>The saved string is **dynamic string**</h1>”

C) “dynamic string” can be set to any value without having to re-deploy the service

D) All users must get the same result

2. Solution Overview

The base solution lives on a two tier application, consisting of a Web tier and a Database tier.

Web Tier: runs an apache web server on EC2 instances in an Auto Scaling Group (ASG) which serves the web content to the users. This tier is spread across two private subnets in two Availability Zones for High Availability (HA).

Database Tier: a DynamoDB table stores the dynamic string which feeds the apache servers with the dynamic content.

Application Load Balancer: One internet facing ALB distributes the user requests to the instances under the ASG.

NAT Gateways: Two NAT Gateways, one in each public subnet allow the Web App to receive updates.

CloudFront (CF): A CloudFront distribution serves static content to the users.

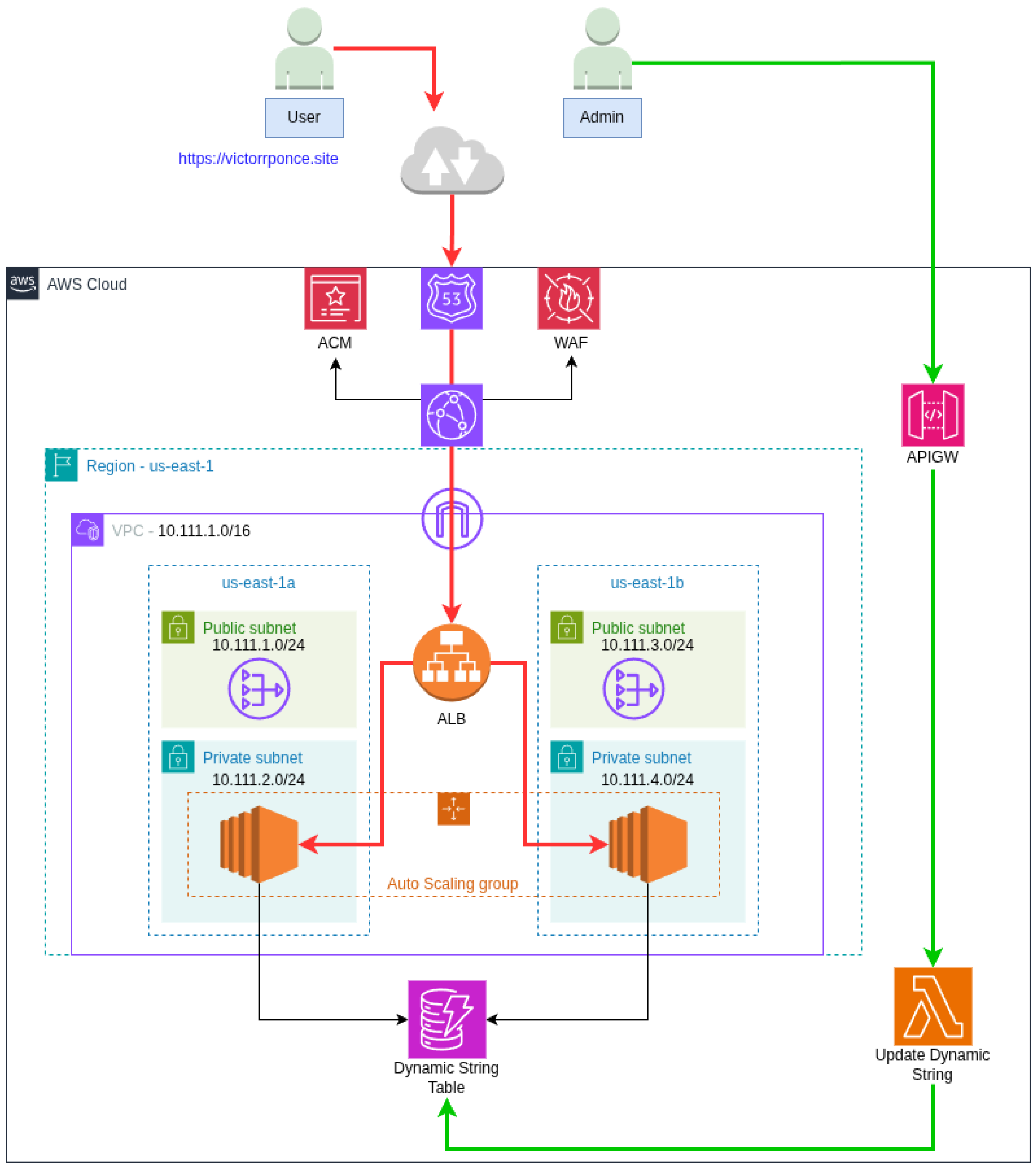
WAF: Four AWS managed rules are configured to protect the CloudFront distribution.

Lambda Function: A Lambda Function updates the dynamic string to the DynamoDB table.

API Gateway: Sets a secure public HTTP endpoint which receives requests from an Admin user and triggers the Lambda Function with these parameters. Uses HTTP API v2 which is lighter, faster and cheaper than REST APIs.

Route 53: Manages a public domain which is used to access the Web app.

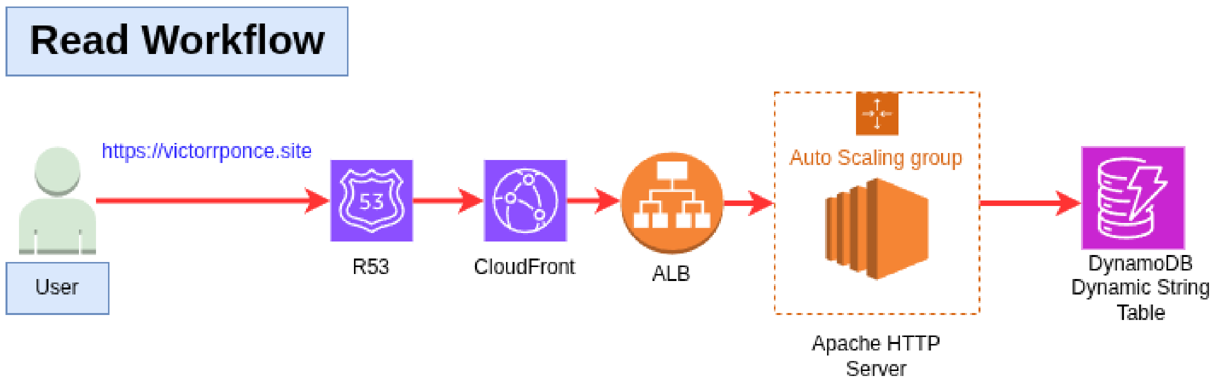
Amazon Certificate Manager (ACM): Generates the TLS certificates for CloudFront and the ALB.



(Image 1. Solution Diagram)

2.1. Read workflow

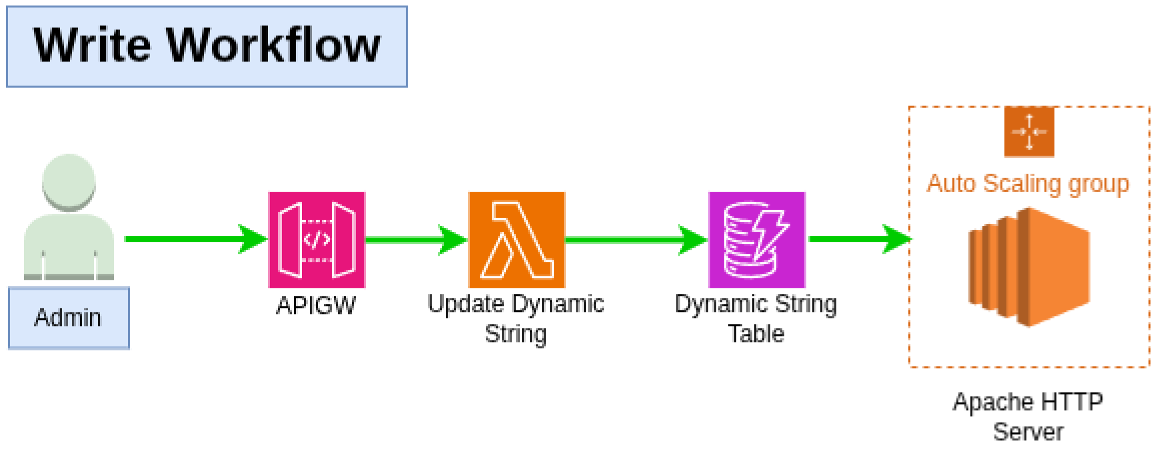
The Read Workflow is the path that allows the user to see the web content. The user access the content by invoking an URL with a custom DNS (victorponce.site) which leads to the CloudFront Distribution, the later forwards the traffic to the ALB which acts as the origin for CF and forwards the traffic to the ASG which is distributed across the respective EC2 instances. The Instances running Apache Web Server read the dynamic string from the DynamoDB table.



(Image 2. Read Workflow)

2.2. Write Workflow

An Admin user can update the dynamic string by invoking the exposed endpoint of the API Gateway by passing the new parameters (using the AWS CLI or Console). API Gateway will invoke the Lambda Function with these parameters which will update the DynamoDB table, serving this new string to the Web Servers.



(Image 3. Write Workflow)

3. Security

Tight security groups only allow user communication to our CloudFront distribution, protecting the rest of infrastructure from unauthorized access and attacks.

ACM certificates are integrated into CloudFront and the ALB, making the traffic between the users and CloudFront, and between CloudFront and the ALB encrypted, making the communication secure.

CloudFront is protected with WAF, using four AWS managed rules (table 1), protecting the solution from common attacks.

Priority	Rule Name	Type	Action	Description
1	AWSManagedRulesCommonRuleSet	Managed Rule Group	Default	Baseline protections against common threats
2	RateLimitPerIP	Rate-Based Rule	Block exceeding	Blocks IPs 800 requests in 5 minutes

				(DDoS/brute-force protection)
3	AWSManagedRulesSQLiRuleSet	Managed Rule Group	Default	Detects SQL injection attempts
4	AWSManagedRulesAmazonIpReputationList	Managed Rule Group	Default	Detects/block requests from known malicious IPs (AWS threat intelligence)

(Table 1. WAF rules)

4. Conclusion

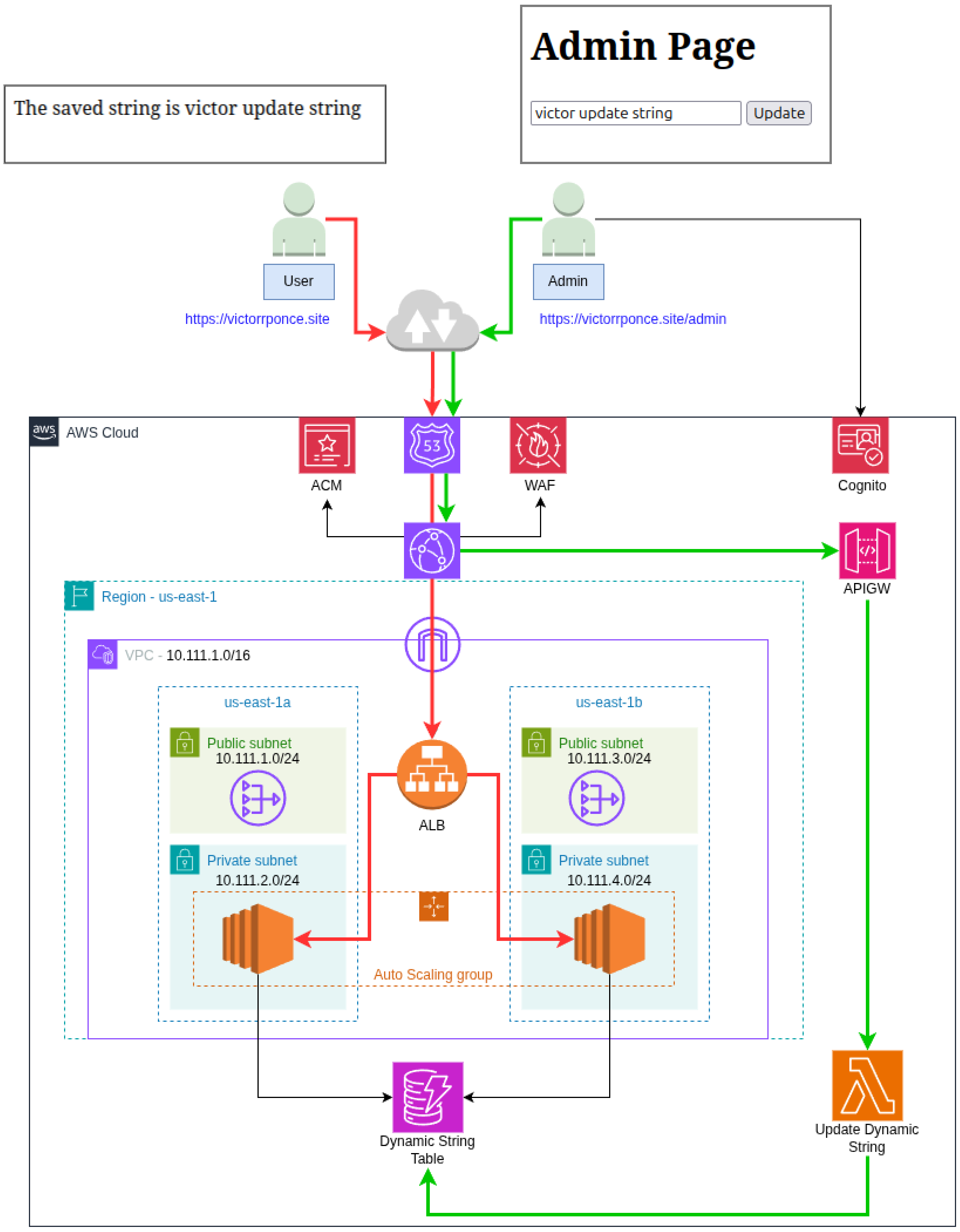
I decided to go for a more classical approach for the Read Workflow, allowing future growth, improvements and complexity to the website, this is why I used EC2 instances under an Auto Scaling Group.

I built the solution having security, scalability and high availability in mind. Users can seamlessly and securely access the web content.

The Write Workflow is serverless up to the point where the data is written to the DynamoDB table, which reduces operation and maintenance. Currently this workflow does not have security integrated, which will be discussed in the next section.

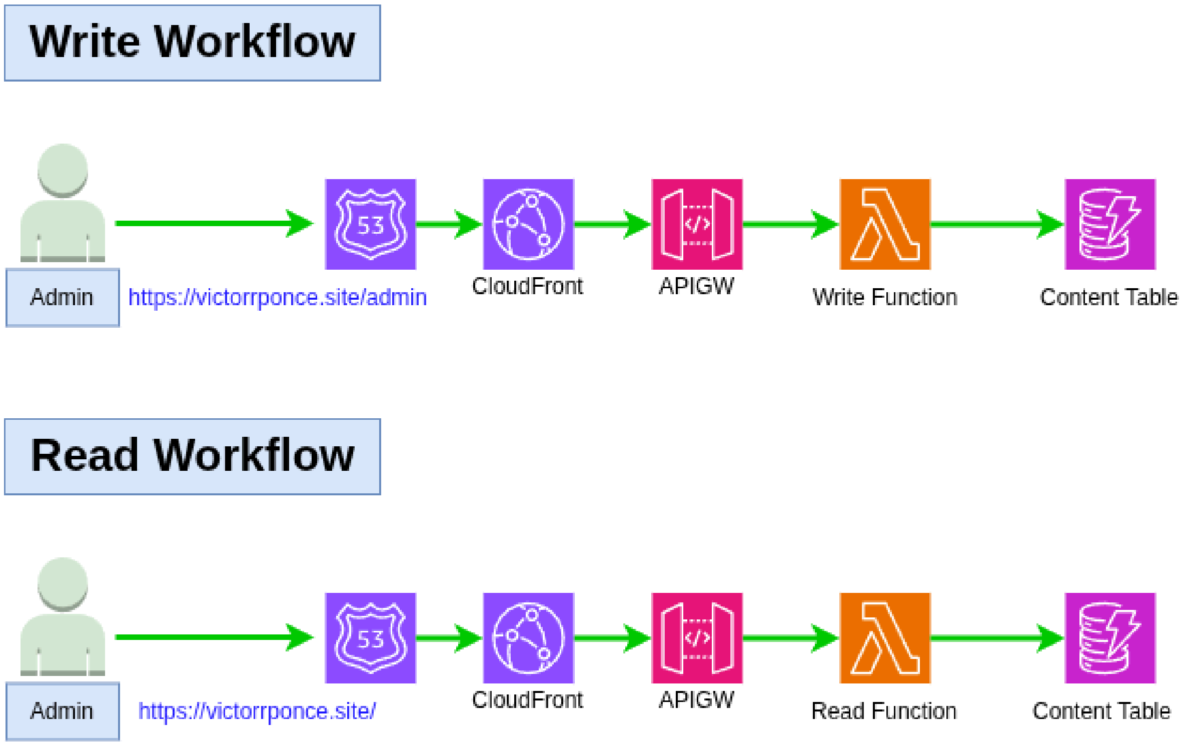
5. Future Improvements

The Write Workflow can be improved by adding a custom path to the domain, leading to a login screen which will give access to update the Dynamic String with the API endpoint (image 4). Authentication can be handled with Cognito among other options. If more complexity is needed for the Website, a relational database like RDS for MySQL or Aurora can be added, replacing the DynamoDB table.


The saved string is victor update string
Admin Page
victor update string
Update
User
Admin
https://victorrponce.site
https://victorrponce.site/admin
AWS Cloud
ACM
WAF
Cognito
APIGW
Region - us-east-1
VPC - 10.111.1.0/16
us-east-1a
us-east-1b
Public subnet 10.111.1.0/24
Public subnet 10.111.3.0/24
Private subnet 10.111.2.0/24
Private subnet 10.111.4.0/24
ALB
Auto Scaling group
Dynamic String Table
Update Dynamic String

(Image 4. Future Improvements)

There's also the option to turn this solution entirely serverless, for this we will keep the Write Workflow the same, but accessing through CloudFront and a custom domain (Route 53). The Read Workflow will have its own read API endpoint (API Gateway), triggering a read function (Lambda Function) which will retrieve the data from a DynamoDB table (image 5).



(Image 5. Serverless Option)