Project Report: Serverless Static Website Hosting with Cloudflare Workers and AWS S3

Author: ASHIQUE PK

Date: August 31, 2025

Introduction

This project's objective was to deploy a static website using a modern, serverless architecture, creating a powerful alternative to traditional hosting that avoids the complexity and cost of managing servers, domains, and SSL certificates. The implementation centers on a Continuous Integration/Continuous Deployment (CI/CD) pipeline that automates the deployment process. Code is managed on **GitHub**, deployed to **AWS S3** for storage, and served globally via the **Cloudflare** network. The key innovation is using a **Cloudflare Worker** as an intelligent reverse proxy, which provides a free https:// URL and a secure, automated hosting solution.

Abstract

This report details a fully automated, serverless pipeline for hosting a static website by integrating **GitHub**, **GitHub Actions**, **AWS S3**, and **Cloudflare Workers**. The workflow begins when a developer pushes code to a GitHub repository, which triggers a GitHub Action to sync the files to a public AWS S3 bucket configured for static website hosting. Instead of being accessed directly, traffic is routed through a Cloudflare Worker that intercepts requests to a free .workers.dev URL. The Worker fetches the corresponding content from the S3 bucket and serves it to the end-user, providing the benefits of Cloudflare's global CDN—including caching and a free SSL certificate (HTTPS)—without a custom domain. The result is a highly available and performant website with a zero-maintenance, automated deployment process.

Tools Used

- **Version Control & CI/CD:**
  - **Git:** For local version control.
  - **GitHub:** For cloud-based repository hosting and CI/CD foundation.
  - **GitHub Actions:** To automate code deployment from the repository to AWS S3.
- **Cloud Infrastructure:**
  - **AWS S3 (Simple Storage Service):** As a durable storage solution for static website files (RESUME.html).
  - **AWS IAM (Identity and Access Management):** To create a secure user with programmatic access for the GitHub Actions workflow.
- **Edge Computing & Delivery:**
  - **Cloudflare Workers:** A serverless environment running on Cloudflare's edge network to act as a reverse proxy.

- **Web Content:**
  - **HTML:** For creating the static website content.

Steps Involved in Building the Project

The project was executed in a logical sequence, from local development to a globally accessible live website.

1. Code Development and Version Control: The static website file (RESUME.html) was created locally, then committed and pushed to a new GitHub repository. This established the foundational codebase and the trigger for automated deployments.

2. Configuring the Storage Backend (AWS S3): An AWS S3 bucket was created to host the website files. It was configured with a unique name, had public access enabled, and was set to use RESUME.html as the index document. A bucket policy was applied to grant public s3:GetObject permission. The public S3 bucket website endpoint URL was saved for a later step.

3. Automating Deployment with GitHub Actions (CI/CD): A CI/CD pipeline was established using GitHub Actions to automate file deployment. An IAM user with programmatic access was created in AWS with permissions to manage objects in the target S3 bucket. The user's Access Key and Secret Key were stored as encrypted Secrets in the GitHub repository. A workflow file (.github/workflows/deploy.yml) was then created to sync the repository's contents to S3 on every push to the main branch.

4. Implementing the Proxy Layer (Cloudflare Worker): To obtain a free HTTPS URL, a Cloudflare Worker was created, which provided a unique *.workers.dev subdomain. The default boilerplate code was replaced with a custom script configured to intercept incoming browser requests, fetch the corresponding file from the S3 bucket's endpoint URL, and return its content to the visitor. The script was then deployed, making the proxy instantly active.

Conclusion

This project proves that a professional-grade, secure, and automated static site hosting solution can be built entirely with free-tier services. By integrating GitHub Actions for CI/CD, AWS S3 for storage, and a Cloudflare Worker as a proxy, all initial objectives were met. The resulting architecture is robust, scalable, and requires no manual intervention post-setup. Leveraging Cloudflare's edge network to serve content from S3 is a highly efficient pattern for securely hosting portfolios, documentation, and personal projects.