Serverless Scalable E-Commerce Website with S3, Cloudflare & GitHub Actions

Introduction:

A scalable static e-commerce website inspired by Amazon, built using:

- Amazon S3 for reliable, cost-effective static file hosting.
- Cloudflare for global CDN, HTTPS, and security.
- GitHub Actions for automated CI/CD deployment.
- This serverless setup ensures high availability, fast performance, and automated deployments without backend servers.

Abstract:

- A serverless e-commerce website is built using Amazon S3 for static hosting, Cloudflare for global CDN, security, and performance, and GitHub Actions for automated CI/CD. This architecture ensures high scalability, low cost, enhanced security, and no server management, making it ideal for modern online stores.

Tools :

- **Amazon S3**
    - Static website hosting.
    - Scalable, durable, and cost-effective storage.
- **Cloudflare**
    - CDN (Content Delivery Network).
    - SSL/TLS encryption.
    - DDoS protection and caching.
    - Performance optimization.
- **GitHub Actions**
    - CI/CD automation.
    - Build, test, and deploy workflows.
    - Automatic deployment to S3.
- **GitHub**
    - Version control.
    - Repository for code collaboration.

Steps Involved in Building the Project:

- Create a basic static website using HTML, CSS, and JavaScript.
- Structure files in a way compatible with S3 (e.g., index.html and 404.html at the root).

  Configure S3 Bucket for Website Hosting
- Create an S3 bucket with the name matching your domain (e.g., example.com).

  Enable static website hosting in the S3 bucket properties.
- Set up index.html and 404.html as the default documents.
- Make the bucket publicly accessible (or use CloudFront for private buckets if needed).

- Upload Static Files to S3

  Manually upload files or automate using the AWS CLI or CI/CD.
- Set Up Cloudflare for CDN and DNS
- Add your domain to Cloudflare.
- Update your domain's nameservers to point to Cloudflare’s.
- Create DNS records (usually a CNAME or A record) that point to the S3 website endpoint.
- Enable features like caching, HTTPS (via Flexible or Full SSL), and performance settings.

- Implement GitHub Actions for CI/CD
- Create a GitHub Actions workflow YAML file (e.g., .github/workflows/deploy.yml).
- Configure the workflow to trigger on pushes to main or deploy branches.
  Use AWS CLI or a GitHub Action to sync changes to the S3 bucket securely using IAM credentials.
- Optionally, add a Cloudflare cache purge step post-deployment.
- Test the Full Deployment Pipeline
- Push changes to GitHub and verify:

- Code builds.
- Files sync to S3.
- CDN is caching properly, and SSL is functioning.
- Changes are reflected on the live site via Cloudflare.

Conclusion

- The serverless e-commerce website leverages S3 for scalable hosting, Cloudflare for global performance & security, and GitHub Actions for automated CI/CD, delivering a secure, resilient, and cost-efficient solution that is easy to maintain and scale.