The document outlines the steps to set up a CI/CD pipeline for automatically building and deploying a static website from GitHub to Amazon S3 using Jenkins on an EC2 instance.

AWS IAM Configuration for Jenkins

This section outlines the steps to create an IAM user or role for Jenkins with the necessary permissions.

- Create an IAM user named "Jenkins-EC2".
- Attach policies: AmazonS3FullAccess, AmazonEC2FullAccess, CloudWatchLogsFullAccess, and optionally AdministratorAccess.
- Generate Access Key and Secret Access Key for AWS CLI access.
- Save the Access Key and Secret Access Key securely.

Setting Up AWS S3 Bucket

This section details the creation and configuration of an S3 bucket for hosting a static website.

- Create a unique S3 bucket (e.g., my-static-website-bucket-jmo-2025) in a preferred AWS region.
- Disable "Block Public Access" to allow public access to the bucket.
- Enable static website hosting with index.html as the index document and error.html as the error document.
- Set a bucket policy to allow public access to the files.
- Upload website files (HTML, CSS, JS) to the bucket and verify accessibility via a public URL.

GitHub Repository Setup

This section describes the process of creating and configuring a GitHub repository for the static website.

- Create a new GitHub repository named "lms-static-website".
- Clone the repository locally and add static website files.
- Commit and push changes to the GitHub repository.

EC2 Instance Setup for Jenkins

This section explains how to launch and configure an EC2 instance to run Jenkins.

- Launch an EC2 instance with Ubuntu and create a key pair for SSH access.
- Configure security group rules to allow SSH, HTTP, and Jenkins access.
- Connect to the EC2 instance via SSH and install Git.
- Generate and configure an SSH key for GitHub access.

Installing and Configuring Jenkins

This section covers the installation of Jenkins and its initial setup.

- Install Java and Jenkins on the EC2 instance.
- Start Jenkins and enable it to start at boot.
- Unlock Jenkins using the initial admin password and install suggested plugins.
- Add AWS credentials to Jenkins for S3 access.

Configuring GitHub Integration with Jenkins

This section outlines the steps to integrate GitHub with Jenkins for CI/CD.

- Create a GitHub personal access token with repo and admin:repo_hook permissions.
- Add GitHub credentials in Jenkins using the token.
- Set up a webhook in GitHub to trigger Jenkins builds on repository changes.

Testing the CI/CD Pipeline

This section describes how to test the CI/CD pipeline by making changes to the GitHub repository.

- Create a Jenkins project and configure it to use the GitHub repository.
- Set up build triggers for automatic builds on GitHub changes.
- Verify the pipeline by making changes to the repository and checking Jenkins for triggered builds.

Clean Up Resources After Testing

This section emphasizes the importance of cleaning up resources to avoid incurring costs.

- Terminate the EC2 instance to stop charges.
- Delete all files in the S3 bucket and the bucket itself to prevent ongoing costs.

Author Information

This section provides details about the author of the document.

- Prepared by John Michael Oliba, with interests in DevOps, Systems Engineering, Administration, Graphics Design, and Accounts & Finance.
- Document version: 1.0, dated November 01, 2025.
- GitHub link to the project: https://github.com/johnmicky1/lms-static-website.