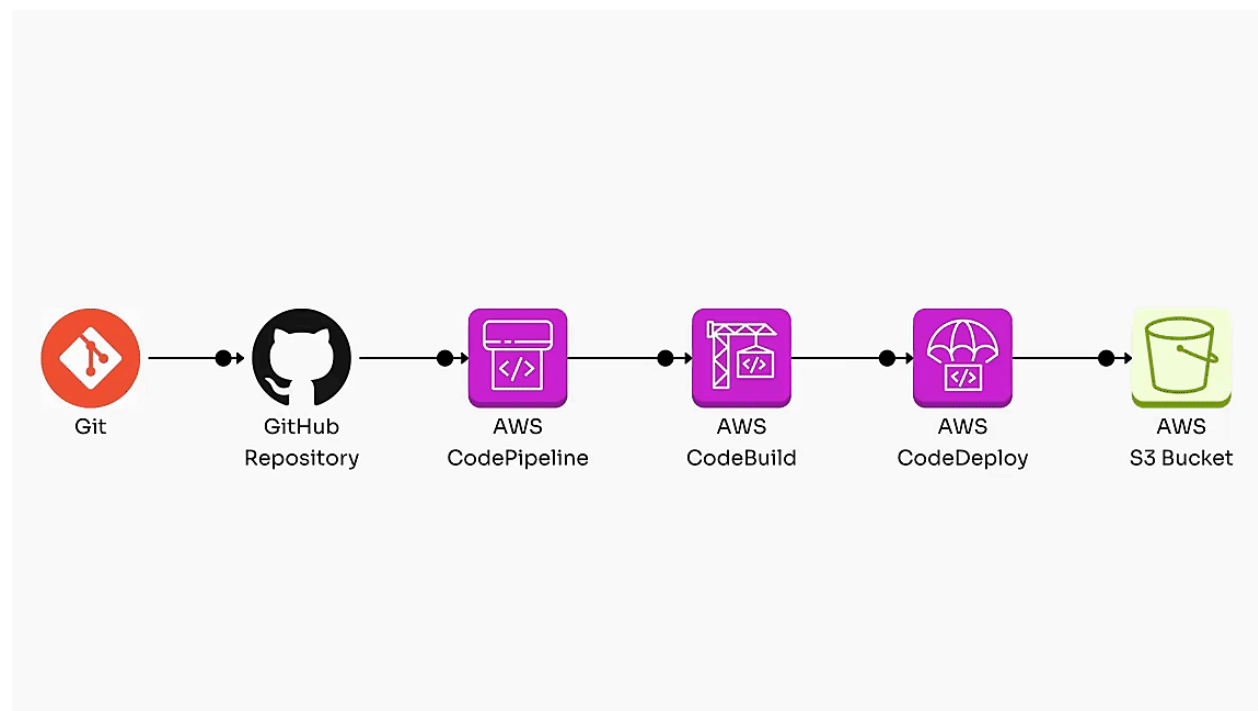


Automated CI/CD Pipeline for Static Website Hosting on AWS

Project Overview

This project demonstrates the creation of a fully automated **CI/CD pipeline** to deploy a static website using **Amazon Web Services (AWS)**. The website is hosted on **Amazon S3** with automated builds and deployments triggered by code changes in a **GitHub repository**, managed through **AWS CodePipeline** and **AWS CodeBuild**.

Architecture Diagram



Technologies Used

1. **Amazon S3**: Hosts the static website and serves it over the internet.
2. **AWS CodePipeline**: Automates the build, test, and deploy process.
3. **AWS CodeBuild**: Executes the build process and packages the website files.
4. **AWS CodeDeploy**: Deploys the website files to S3 bucket.
5. **GitHub**: Stores the website's source code and triggers pipeline executions on new commits.

Workflow

1. Clone a repository and push files into GitHub.

- Installed git and cloned a repository named “Serverless-CICD”.

```
hardikrathod@DA:F8:CA:C3:DD:3C ~ % git --version
git version 2.47.0
hardikrathod@DA:F8:CA:C3:DD:3C ~ % git clone https://github.com/Hardik9791/Serverless-CICD.git
Cloning into 'Serverless-CICD'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
hardikrathod@DA:F8:CA:C3:DD:3C ~ %
```

- Added static website files to cloned repository and verified it.

```
hardikrathod@DA:F8:CA:C3:DD:3C Serverless-CICD % git status
On branch main
Your branch is up to date with 'origin/main'.
```

Untracked files:

(use "git add <file>..." to include in what will be committed)

```
app.js
buildspec.yaml
index.html
styles.css
```

```
nothing added to commit but untracked files present (use "git add" to track)
hardikrathod@DA:F8:CA:C3:DD:3C Serverless-CICD %
```

- Pushed all the files to **GitHub** using “git” commands.

```
hardikrathod@DA:F8:CA:C3:DD:3C Serverless-CICD % git push origin main
Username for 'https://github.com/Hardik9791/Serverless-CICD.git': hardik9791
Password for 'https://hardik9791@github.com/Hardik9791/Serverless-CICD.git':
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 1.08 KiB | 1.08 MiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Hardik9791/Serverless-CICD.git
    db9cf8f..d47e4b2  main -> main
hardikrathod@DA:F8:CA:C3:DD:3C Serverless-CICD %
```

2. Set Up S3 for Static Website Hosting

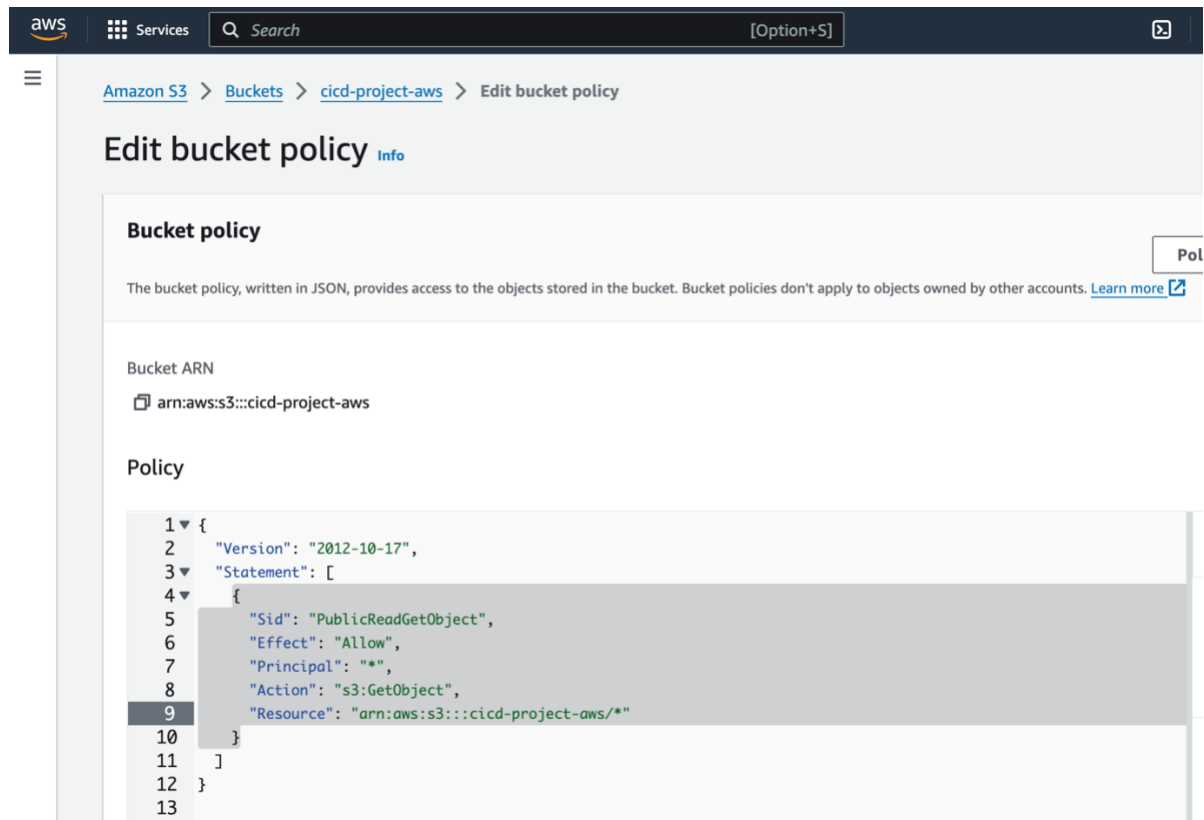
- I created an **S3 bucket** named `cicd-project-aws` and ensured it was publicly accessible by disabling the "Block all public access" setting.

The screenshot shows the AWS Management Console interface for creating a new S3 bucket. The breadcrumb navigation at the top indicates the path: [Amazon S3](#) > [Buckets](#) > [Create bucket](#). The main heading is "Create bucket" with an "Info" link. Below this, a subheading states "Buckets are containers for data stored in S3." The "General configuration" section is active, showing the "AWS Region" as "Canada (Central) ca-central-1". The "Bucket name" field is populated with "cicd-project-aws" and has an "Info" link. A note below the field states: "Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)". There is a section for "Copy settings from existing bucket - optional" with a "Choose bucket" button and a format example: "Format: s3://bucket/prefix". The "Object Ownership" section is also visible, with two options: "ACLs disabled (recommended)" (selected) and "ACLs enabled". The footer of the console shows "CloudShell", "Feedback", and a copyright notice "© 2024, Am".

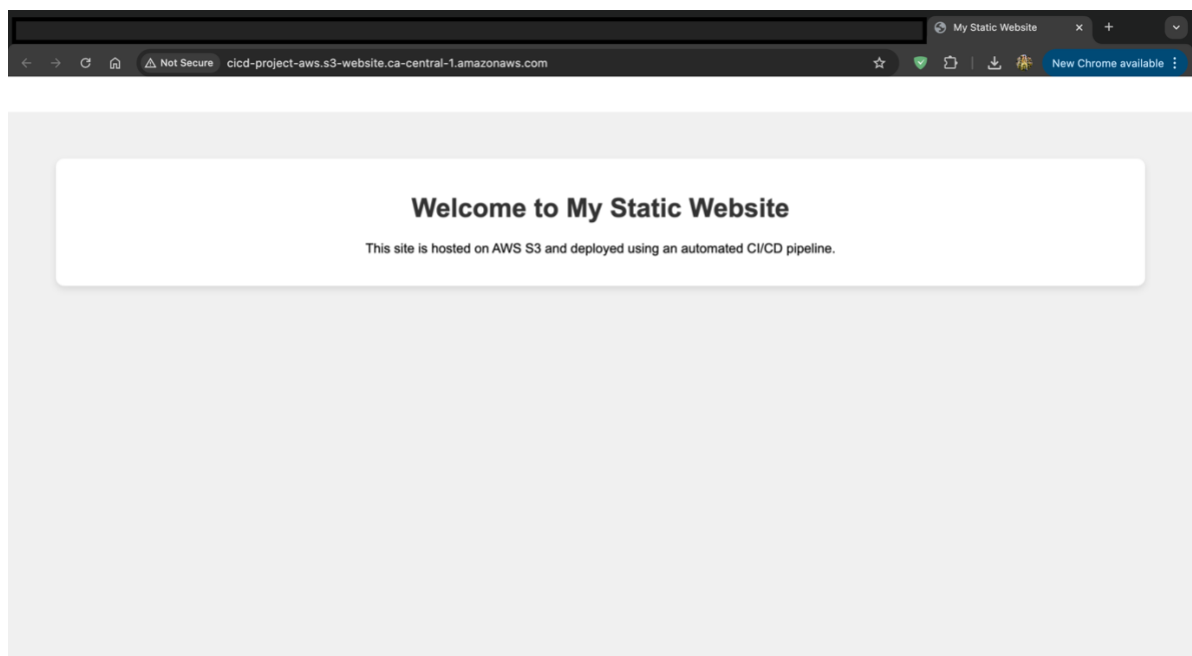
- I enabled **Static Website Hosting** for the bucket and specified `index.html` as the **Index document**.

The screenshot shows the "Static website hosting" configuration page for the bucket. The heading is "Static website hosting" with an "Edit" button in the top right. A subheading states: "Use this bucket to host a website or redirect requests. [Learn more](#)". The "S3 static website hosting" section shows "Enabled". The "Hosting type" is set to "Bucket hosting". The "Bucket website endpoint" section shows the endpoint URL: <http://cicd-project-aws.s3-website-ca-central-1.amazonaws.com>. The footer of the console shows "CloudShell", "Feedback", and a copyright notice "© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences".

- To ensure public access, I added a **Bucket Policy** that grants read permissions to all users, allowing my static files to be accessed over the web.



- Accessed a website using static website URL.



3. Set Up AWS CodePipeline

- I created a new **AWS CodePipeline** to automate the deployment process.

The screenshot shows the AWS CodePipeline console interface. On the left, a sidebar lists steps from Step 3 to Step 6. The main area is titled 'Pipeline settings' and contains the following fields and options:

- Pipeline name:** A text input field containing 'cicd-project-pipeline'. A note below states: 'Enter the pipeline name. You cannot edit the pipeline name after it is created. No more than 100 characters.'
- Pipeline type:** A section with a blue information box stating: 'You can no longer create V1 pipelines through the console. We recommend you use the V2 pipeline type with improved release safety, pipeline triggers, parameterized pipelines, and a new billing model.'
- Execution mode:** Radio buttons for 'Superseded', 'Queued (Pipeline type V2 required)', and 'Parallel (Pipeline type V2 required)'. The 'Queued' option is selected. Descriptions for each mode are provided.
- Service role:** Two options: 'New service role' (selected) and 'Existing service role'. The 'New service role' option includes a sub-option 'Create a service role in your account'.
- Role name:** A text input field containing 'AWSCodePipelineServiceRole-ca-central-1-cicd-project-pipeline'. A note below states: 'Type your service role name.'
- Checkboxes:** A checkbox labeled 'Allow AWS CodePipeline to create a service role so it can be used with this new' is checked.

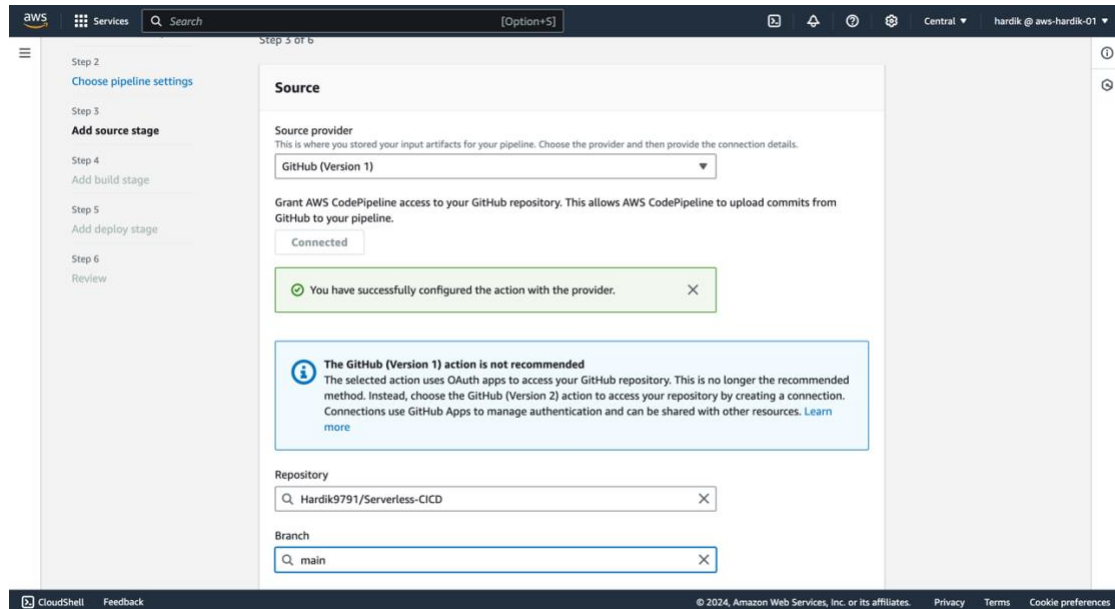
The bottom of the console shows the AWS logo, 'CloudShell', 'Feedback', and copyright information for 2024.

- Source Stage:** I connected my GitHub repository to the pipeline, so any new commits to the main branch would trigger the pipeline.

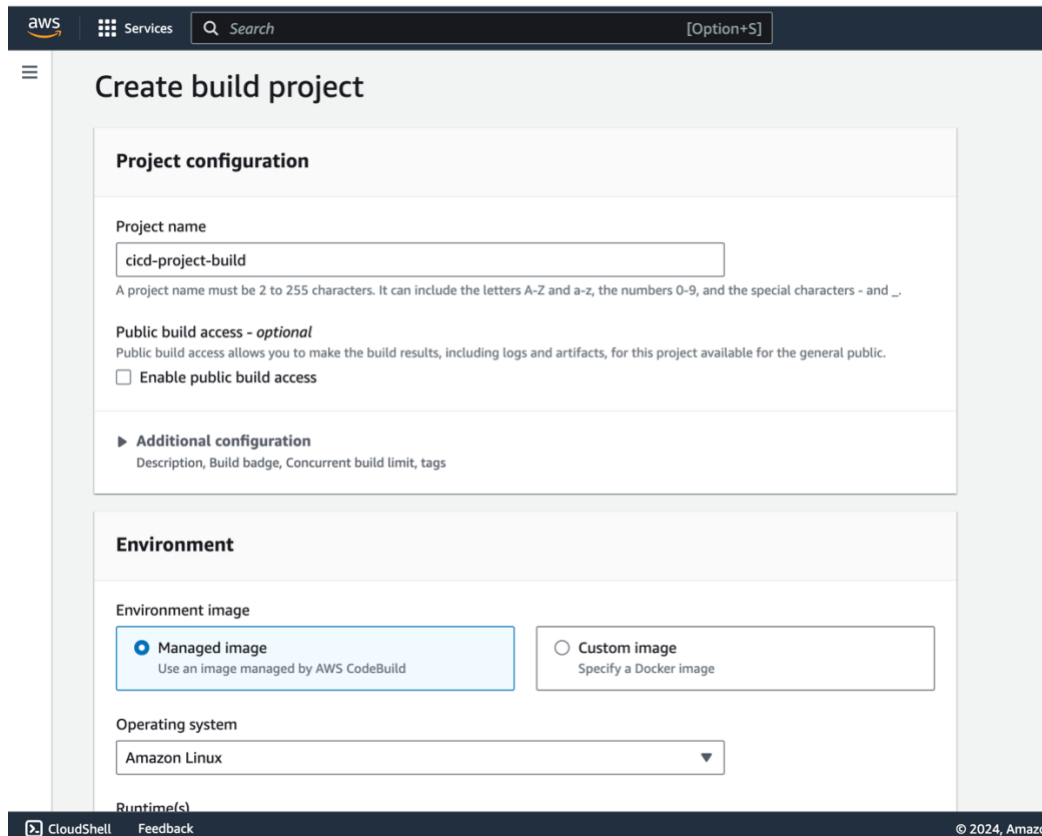
The screenshot shows a GitHub authorization dialog titled 'Authorize AWS CodePipeline (Canada Central)'. At the top, there are icons for AWS CodePipeline, a green checkmark, and the GitHub logo. The dialog contains the following information:

- Requester:** 'AWS CodePipeline (Canada Central) by aws-codesuite' wants to access your 'Hardik9791' account.
- Repository webhooks and services:** A section with a dropdown arrow, showing 'Admin' access.
- Repositories:** A section with a dropdown arrow, showing 'Public and private' access.
- Buttons:** 'Cancel' and 'Authorize aws-codesuite'.
- Footer:** 'Authorizing will redirect to https://ca-central-1.console.aws.amazon.com'

- After connecting, selected GitHub as a source provider.



- **Build Stage: I set up AWS CodeBuild to package my static website files. In my GitHub repository, I added a “buildspec.yml” file to define how the files are built and prepared for deployment.**



aws

Services

Search

[Option+S]

image version

Always use the latest image for this runtime version

☐ Use GPU-enhanced compute

Service role

☒ New service role
Create a service role in your account

☐ Existing service role
Choose an existing service role from your account

Role name

codebuild-cicd-project-build-service-role

Type your service role name

► Additional configuration

Timeout, privileged, certificate, VPC, compute type, environment variables, file systems

Buildspec

Build specifications

☐ Insert build commands
Store build commands as build project configuration

☒ Use a buildspec file
Store build commands in a YAML-formatted buildspec file

Buildspec name - optional

By default, CodeBuild looks for a file named buildspec.yml in the source code root directory. If your buildspec file uses a different name or location, enter its path from the source root here (for example, buildspec-two.yml or configuration/buildspec.yml).

buildspec.yml

CloudShell

Feedback

© 2024, Amazon V

[Option+S]

Build - optional

Build provider

Choose the tool you want to use to run build commands and specify artifacts for your build action.

☐ Commands

☒ Other build providers

AWS CodeBuild

Project name

Choose a build project that you have already created in the AWS CodeBuild console. Or create a build project in the AWS CodeBuild console and then return to this task.

Q ccd-project-build

X

or

Create project

✓ Successfully created ccd-project-build in CodeBuild.

X

Environment variables - optional

Choose the key, value, and type for your CodeBuild environment variables. In the value field, you can reference variables generated by CodePipeline. [Learn more](#)

Add environment variable

Build type

☒ Single build
Triggers a single build.

☐ Batch build
Triggers multiple builds as a single execution.

Region

Canada (Central)

© 2024, Amazon Web Services, Inc. or its affiliates.

- **Deploy Stage:** I configured the pipeline to deploy the built files to my S3 bucket using the **S3 deploy action**, ensuring that the latest version of the website would be served.

[Option+S]

Deploy - optional

Deploy provider

Choose how you deploy to instances. Choose the provider, and then provide the configuration details for that provider.

Amazon S3

Region

Canada (Central)

Input artifacts

Choose an input artifact for this action. [Learn more](#)

BuildArtifact

Defined by: Build

No more than 100 characters

Bucket

cicd-project-aws

S3 object key

index.html

Enter the object key. You can include a file path without the delimiter character (/) at the beginning. Include the file extension. Example: SampleApp.zip

☐ Extract file before deploy

The deployed artifact will be unzipped before deployment.

Additional configuration

© 2024, Amazon Web Services, Inc. or its affiliates.

Objects | Properties | Permissions | Metrics | Management | Access Points

Objects (6) Info

Copy S3 URI

Copy URL

Download

Open

Delete

Actions

Create folder

Upload

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Find objects by prefix

Show versions

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	.DS_Store	DS_Store	October 18, 2024, 02:48:04 (UTC-04:00)	6.0 KB	Standard
<input type="checkbox"/>	app.js	js	October 18, 2024, 02:48:04 (UTC-04:00)	82.0 B	Standard
<input type="checkbox"/>	buildspec.yml	yml	October 18, 2024, 02:48:04 (UTC-04:00)	248.0 B	Standard
<input type="checkbox"/>	index.html	html	October 18, 2024, 02:48:04 (UTC-04:00)	498.0 B	Standard
<input type="checkbox"/>	README.md	md	October 18, 2024, 02:48:04 (UTC-04:00)	17.0 B	Standard
<input type="checkbox"/>	styles.css	css	October 18, 2024, 02:48:04 (UTC-04:00)	279.0 B	Standard

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

4. Tested the CI/CD Pipeline

- I made changes to my website's source code in the GitHub repository, including updates to index.html.

```
<div class="container">
  <h1>Welcome to My Static Website</h1>
  <p>This site is hosted on AWS S3 and deployed using an automated CI/CD pipeline.<
  <p>CICD Working!</p>
```

- After pushing the changes to GitHub, the **CodePipeline** automatically triggered the process:

```
hardikrathod@DA:F8:CA:C3:DD:3C Serverless-CICD % git add .
hardikrathod@DA:F8:CA:C3:DD:3C Serverless-CICD % git commit -m "updated index.html file"
[main 100d4e0] updated index.html file
 1 file changed, 1 insertion(+), 1 deletion(-)
hardikrathod@DA:F8:CA:C3:DD:3C Serverless-CICD % git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 308 bytes | 308.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/Hardik9791/Serverless-CICD.git
 767bc70..100d4e0  main -> main
hardikrathod@DA:F8:CA:C3:DD:3C Serverless-CICD %
```

- The pipeline fetched the new code from GitHub.

Developer Tools > CodePipeline > Pipelines > cicd-project-pipeline

cicd-project-pipeline

Pipeline type: V2 Execution mode: QUEUED

Buttons: Notify, Edit, Stop execution, Clone pipeline, Release change

Source Succeeded

Pipeline execution ID: ff19c769-599f-4abf-9d38-65df8287baec

GitHub-version-2

[GitHub \(Version 1\)](#)

Succeeded - Just now

[767bc707](#)

View details

[767bc707](#) [GitHub-version-2: updated index.html file](#)

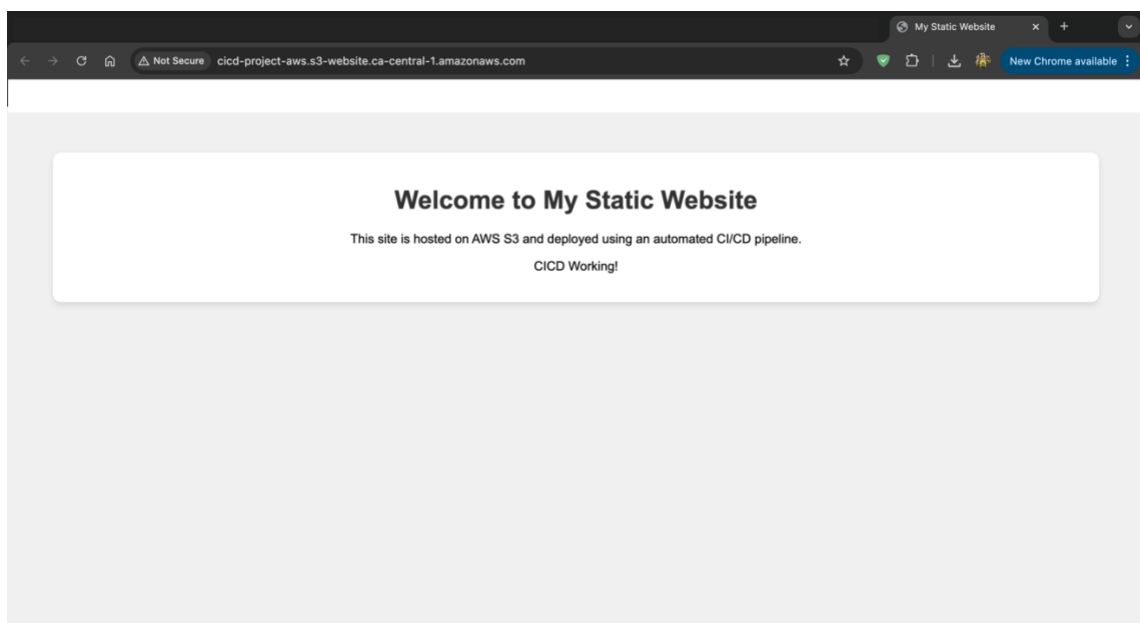
- CodeBuild **packaged the updated files according to my buildspec.yml configuration.**



- The new version of the site was deployed to my **S3 bucket.**



- Finally, I accessed my static website through the **S3 website endpoint** to verify that the changes were deployed successfully.



Project Features

- **Automated Deployments:** The pipeline automatically builds and deploys the website whenever there is a new commit to the GitHub repository.
- **No Servers to Manage:** Since the website is hosted on S3 and deployed using CodePipeline, you don't need to manage any infrastructure.
- **Scalable and Cost-Effective:** S3 provides virtually unlimited scalability, and you only pay for the storage and bandwidth you use.

How the Project Works

1. **Source Code:** The static website's source code (HTML, CSS, JavaScript) is stored in a **GitHub repository**.
2. **Pipeline Trigger:** When changes are pushed to the GitHub repository, the **CodePipeline** is automatically triggered to start the deployment process.
3. **Build Process:**
 - **AWS CodeBuild** packages the static website files based on the instructions defined in the `buildspec.yml` file.
 - The files are prepared for deployment, ensuring that the static assets (HTML, CSS, JS) are correctly bundled.
4. **Deployment:**
 - The packaged files are deployed to **Amazon S3**, where they are made publicly accessible.
 - Proper deployment configuration ensures that all files are extracted and placed in the root of the S3 bucket, making the website accessible.
5. **Accessing the Website:** Users can access the website via the **S3 website endpoint**, which serves the static files directly over the web.

Challenges Faced

- **Deployment Configuration Issue:** I initially set the S3 key object as `index.html` in the deploy stage, which caused all files to be bundled into a single `index.html` file and compressed into a zip. This made the static website inaccessible.
- **Solution:** I corrected the issue by removing the `index.html` from the S3 key object and choosing the option to **extract all files before deployment**. This allowed the files to be deployed properly, and I was able to access my static website successfully.

Skills Highlighted

- **AWS Services Expertise:** Demonstrated proficiency in AWS S3, CodePipeline, and CodeBuild, effectively automating the deployment process for a static website.
- **Problem Solving:** Identified and resolved issues with file extraction and deployment paths, ensuring the static website was accessible post-deployment.
- **Automation:** Set up a fully automated CI/CD pipeline, showcasing knowledge of continuous integration and deployment processes using AWS services.
- **GitHub Integration:** Successfully integrated GitHub with CodePipeline to trigger automatic deployments based on new commits, highlighting proficiency in version control and source management.

Conclusion

This project provides a fully automated, serverless CI/CD pipeline for deploying static websites using AWS services. By leveraging **S3**, **CodePipeline**, and **CodeBuild**, we achieved continuous deployment with minimal setup and no manual intervention required after code changes.