AWS Beginner Project 1

Problem Statement: Deploy a Static Website on AWS using S3, CloudFront and CI/CD with GitHub Actions

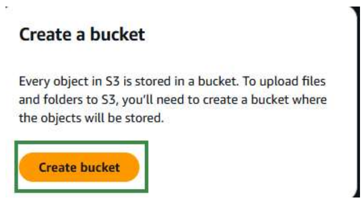
Project Overview: In the above project you will deploy a static website using Amazon S3, you will configure cloud front for content delivery and setup CI/CD using GITHUB Actions.

Pre-requisites

1. Amazon Free Tier Account.
2. Knowledge on how to create IAM users and add permissions.

Section 1 - Step by Step Guide

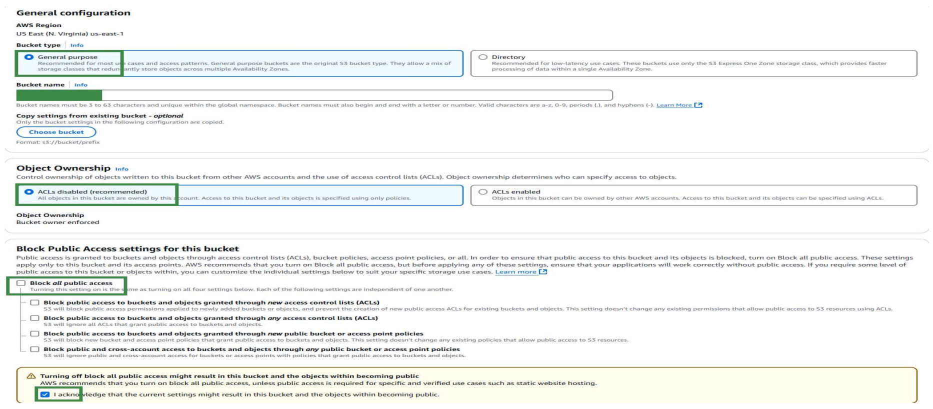
1. Login into AWS Management Console
2. Navigate to Storage – S3
3. Click on create a bucket



Create a bucket

Every object in S3 is stored in a bucket. To upload files and folders to S3, you'll need to create a bucket where the objects will be stored.

4. Check the AWS region where you are creating the bucket.
5. Select General Purpose.
6. Provide a unique bucket name.
7. Uncheck Block All public access
8. Confirm the changes
9. Click on create bucket



General configuration

AWS Region
US East (N. Virginia) us-east-1

Bucket type
[✓ General purpose] Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.
Directory: Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name
[redacted]
Bucket names must be 3 to 63 characters and unique within the global namespace. Bucket names must also begin and end with a letter or number. Valid characters are a-z, 0-9, periods (.), and hyphens (-). Learn More

Copy settings from existing bucket - *optional*
Only the bucket settings in the following configuration are copied.
Format: s3://bucket/prefix

Object Ownership
Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.
[✓ ACLs disabled (recommended)] All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.
ACLs enabled: Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

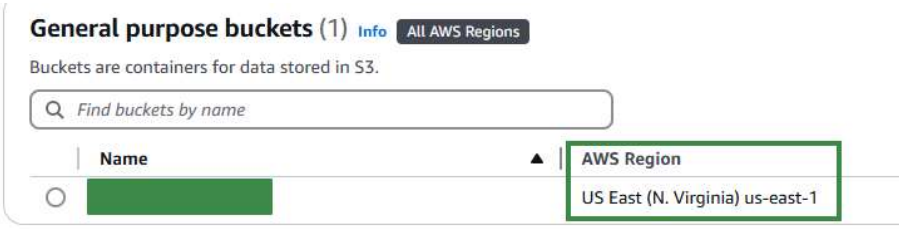
Object Ownership
Bucket owner enforced

Block Public Access settings for this bucket
Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more

☐ Block *all* public access
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.
- ☐ Block public access to buckets and objects granted through *new* access control lists (ACLs)
  S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
- ☐ Block public access to buckets and objects granted through *any* access control lists (ACLs)
  S3 will ignore all ACLs that grant public access to buckets and objects.
- ☐ Block public access to buckets and objects granted through *new* public bucket or access point policies
  S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
- ☐ Block public and cross-account access to buckets and objects through *any* public bucket or access point policies
  S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Turning off block all public access might result in this bucket and the objects within becoming public
AWS recommends that you turn on block all public access, unless public access is required for specific and verified use cases such as static website hosting.
✓ I acknowledge that the current settings might result in this bucket and the objects within becoming public.

10. The bucket has been created successfully.



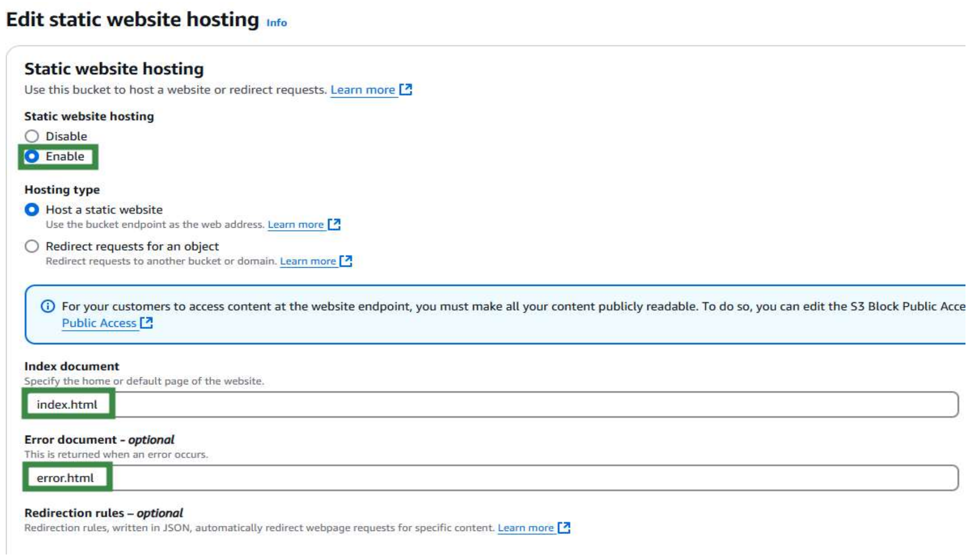
General purpose buckets (1) Info All AWS Regions

Buckets are containers for data stored in S3.

Find buckets by name

Name	AWS Region
[redacted]	US East (N. Virginia) us-east-1

11. Click on the bucket name and navigate to the Properties.
12. Enable Static Website hosting.
13. In the hosting type select – Host a Static Website
14. In the index document enter index.html and error document enter error.html
15. Click on save changes



Edit static website hosting Info

Static website hosting

Use this bucket to host a website or redirect requests. Learn more

Static website hosting
- Disable
- [✓ Enable]

Hosting type
- [✓ Host a static website] Use the bucket endpoint as the web address. Learn more
- Redirect requests for an object. Redirect requests to another bucket or domain. Learn more

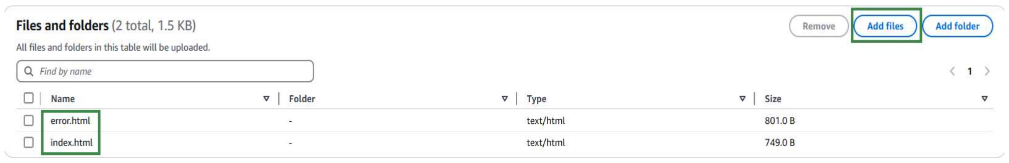
For your customers to access content at the website endpoint, you must make all your content publicly readable. To do so, you can edit the S3 Block Public Acce
Public Access

Index document
Specify the home or default page of the website.
index.html

Error document - *optional*
This is returned when an error occurs.
error.html

Redirection rules – *optional*
Redirection rules, written in JSON, automatically redirect webpage requests for specific content. Learn more

16. I will create a sample html and error file and will upload into the S3 bucket.
17. Click on add files and select the html files. Click on Upload

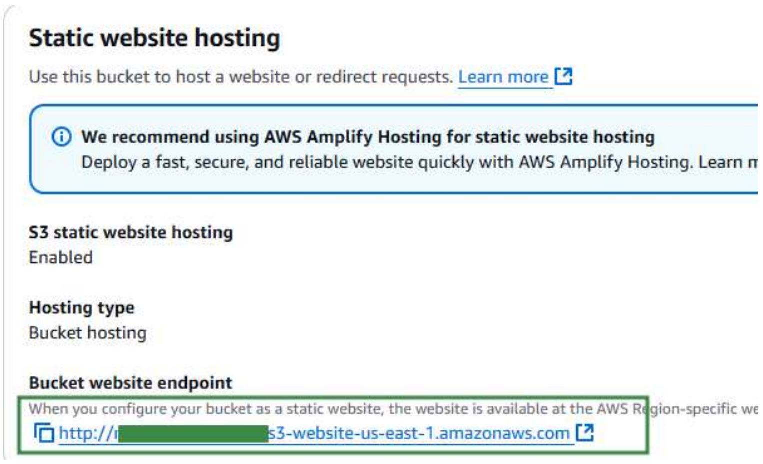


Files and folders (2 total, 1.5 KB)

All files and folders in this table will be uploaded.

Name	Folder	Type	Size
error.html	-	text/html	801.0 B
index.html	-	text/html	749.0 B

18. Let us test out if the index.html page is getting displayed using the bucket website endpoint.
19. Copy the endpoint URL and paste it in the browser. Use http://<URL>



Static website hosting

Use this bucket to host a website or redirect requests. Learn more

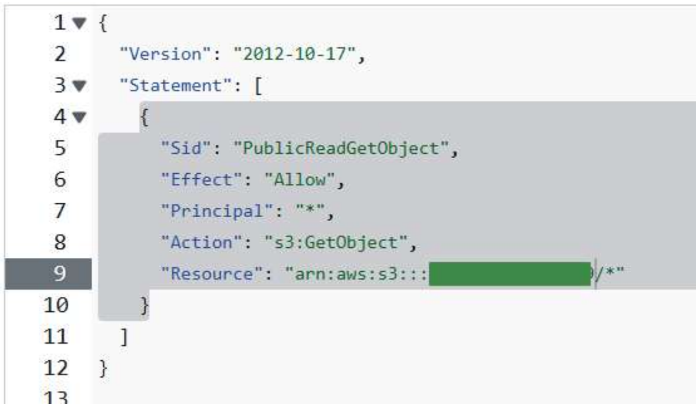
We recommend using AWS Amplify Hosting for static website hosting
Deploy a fast, secure, and reliable website quickly with AWS Amplify Hosting. Learn m

S3 static website hosting
Enabled

Hosting type
Bucket hosting

Bucket website endpoint
When you configure your bucket as a static website, the website is available at the AWS Region-specific we
http://r[redacted]s3-website-us-east-1.amazonaws.com

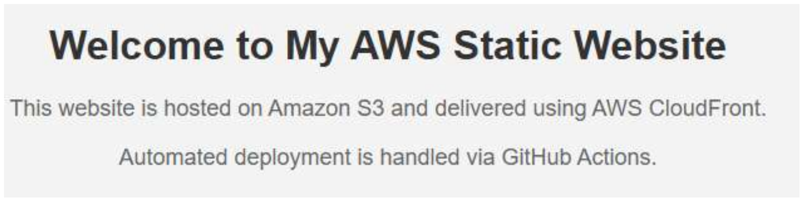
20. The message displayed will be Access Denied and need to add permission to the bucket policy.
21. Navigate to Bucket Permissions > Bucket Policy and Click on EDIT
22. I will explain the code which is written in JSON format. The users need access to view the index.html content



```
 1 {
 2   "Version": "2012-10-17",
 3   "Statement": [
 4     {
 5       "Sid": "PublicReadGetObject",
 6       "Effect": "Allow",
 7       "Principal": "*",
 8       "Action": "s3:GetObject",
 9       "Resource": "arn:aws:s3:::[redacted]/*"
10     }
11   ]
12 }
13
```

23. Line 2 represents the latest version and should always be used. It specifies the policy language version
24. The statement represents the list of permission rules and there is only one rule in the array
25. The SID is optional, and I have named it PublicReadGetObject which means the rule allows public read access
26. Effect can be allow or deny and here it means the action to read the file is permitted
27. The principal is Who can access the bucket and the '*' means anyone on the internet can access the objects in this bucket
28. You can replace with an IAM role or AWS account id if you want to limit access or services
29. The resource indicates who is affected and here it is the bucket name and /* means 'it refers all the objects inside the bucket'

30. Click on save changes
31. Refresh the end point URL and the index.html page will be displayed.

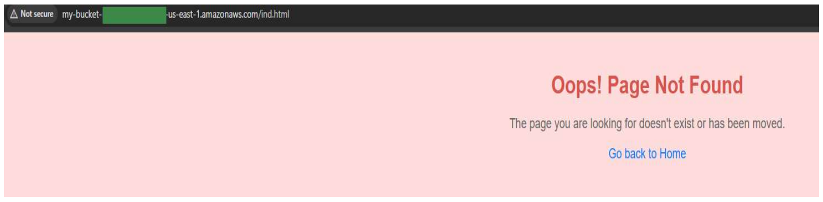


Welcome to My AWS Static Website

This website is hosted on Amazon S3 and delivered using AWS CloudFront.

Automated deployment is handled via GitHub Actions.

32. Instead of index.html type in.html and it will redirect to the error page



Not secure my-bucket-[redacted]-us-east-1.amazonaws.com/ind.html

Oops! Page Not Found

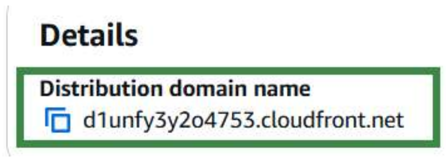
The page you are looking for doesn't exist or has been moved.

Go back to Home

Section 2 – Configure Cloud Front for Content Delivery

33. In the AWS management console, locate CloudFront Services
34. Click on Create Distribution

35. In the Set Origin Name, Select the Name of your S3 Bucket from the drop down
36. Leave all the default settings as-is
37. In the Web Application Firewall Select Do not Enable Security Permissions
38. Click on Create Distribution. It will take 5 minutes to get completed. Copy the distribution name and paste in the browser address bar.

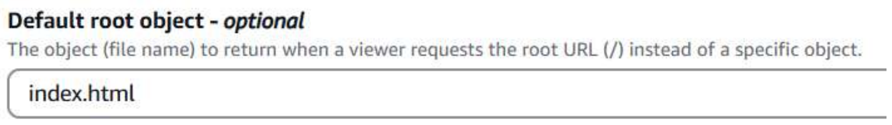


Details

Distribution domain name

d1unfy3y2o4753.cloudfront.net

39. We will get an error message Access Denied while using the cloudfront URL.
40. How do we de-bug the error message.
41. While creating the distribution, add the index.html in the default root object under the general tab of the cloud front distribution

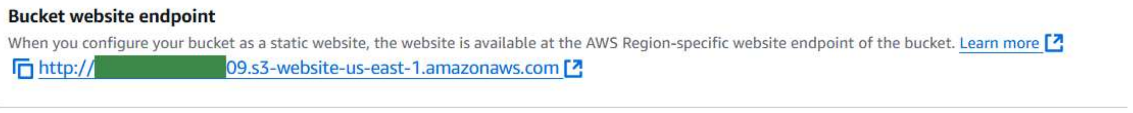

Default root object - *optional*

The object (file name) to return when a viewer requests the root URL (/) instead of a specific object.

index.html

42. While creating the distribution, add the static website hosting property URL instead of selecting the default origin name.

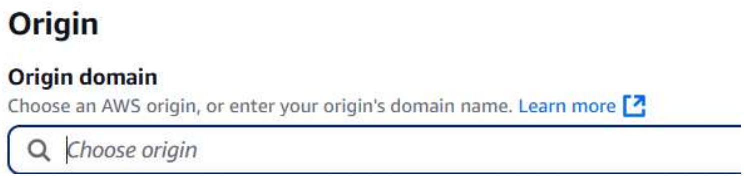
43. Copy the entire URL


Bucket website endpoint

When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. Learn more

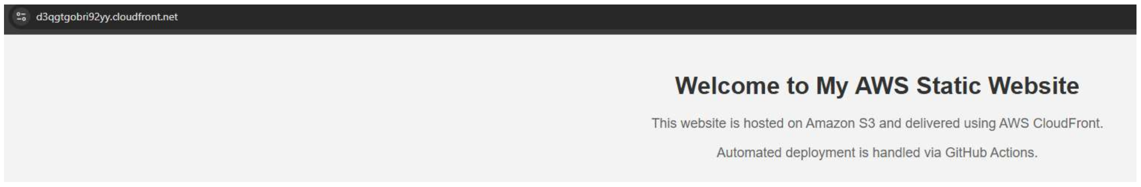
http://[redacted]09.s3-website-us-east-1.amazonaws.com

44. Paste it the Origin name


Origin

Origin domain

Choose an AWS origin, or enter your origin's domain name. Learn more


d3qgtgobri92yy.cloudfront.net

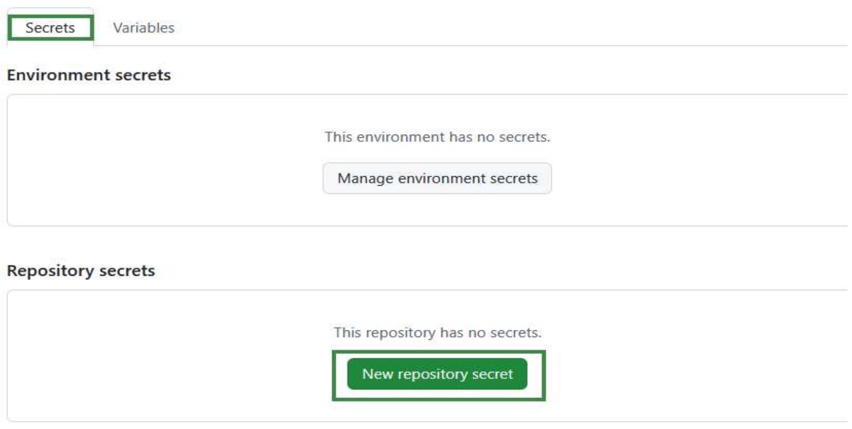
Welcome to My AWS Static Website

This website is hosted on Amazon S3 and delivered using AWS CloudFront.

Automated deployment is handled via GitHub Actions.

Section 3 – Automate Deployments using GITHUB Actions

45. You will need a GITHIB Account
46. You will need to create a repo
47. You will need the AWS Key ID and Secret Key
48. Add the AWS KEY ID and Secret Key as mentioned below
49. After creating a repo , navigate to settings
50. Locate Secrets and Variables – Click on it
51. Select Actions
52. Select Secrets Tab



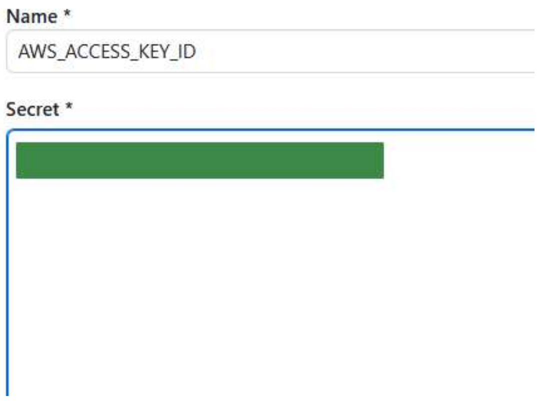
Environment secrets

This environment has no secrets.

Repository secrets

This repository has no secrets.

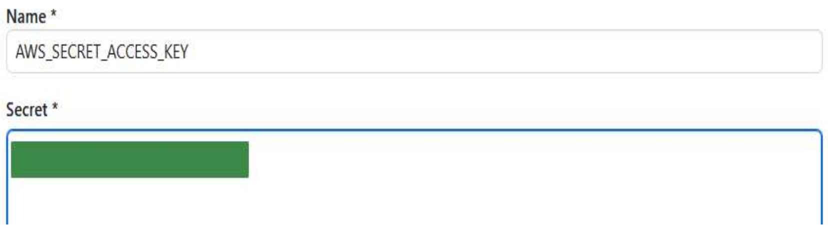
53. Click on New Repository Secret
54. Provide a name and add the secret id



Name *
AWS_ACCESS_KEY_ID

Secret *
[redacted]

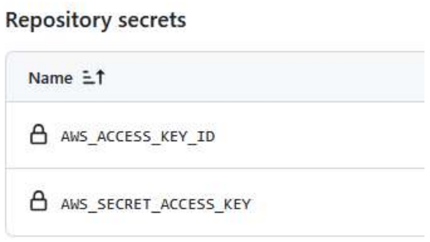
55. Click again in the New Repository Secret
56. Provide a name and add the secret key



Name *
AWS_SECRET_ACCESS_KEY

Secret *
[redacted]

57. Click on Add Secret
58. The secrets are added



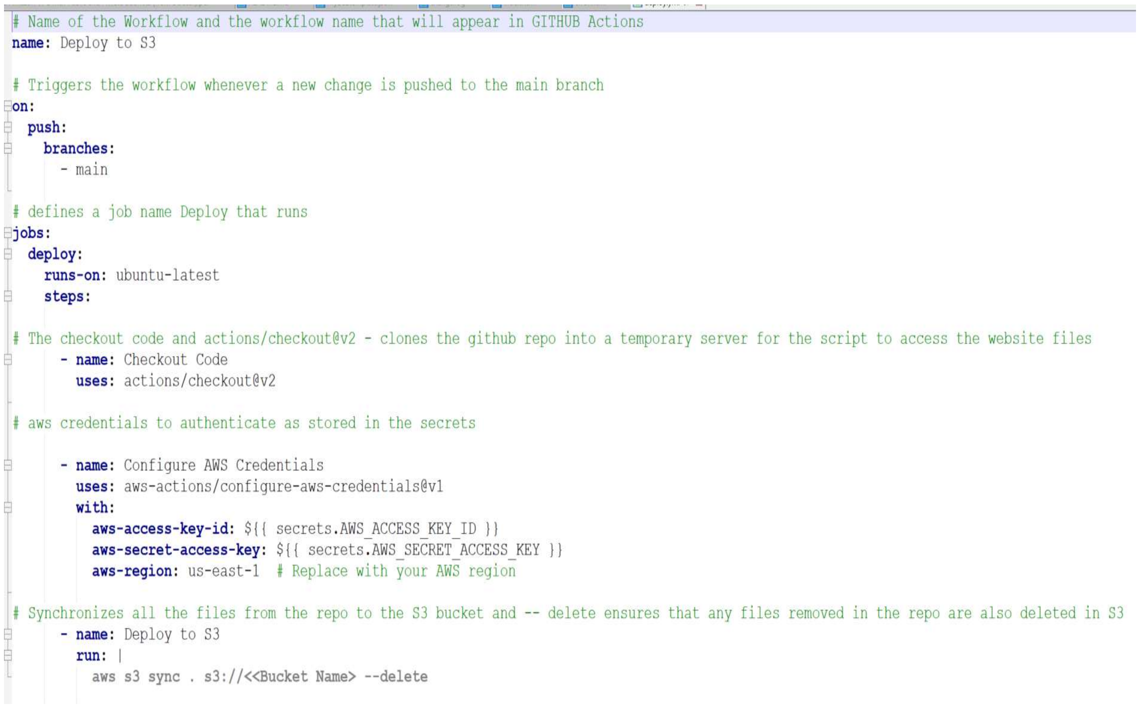
Repository secrets

Name
AWS_ACCESS_KEY_ID
AWS_SECRET_ACCESS_KEY

59. Now we will create a workflow in GITHUB and whenever a new change is pushed to the main branch and synchronized all files from the GITHUB report to the s3 bucket.

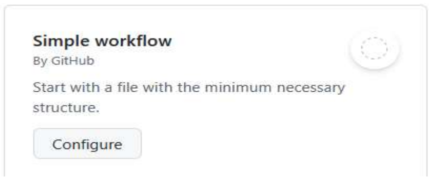
60. So, what does the workflow do is that when you push updates to the main branch of the main branch the workflow will clone the latest code, and it will authenticate using the AWS credentials and uploads the files to the S3 bucket automatically without manual intervention.

Section 4 – Writing YAM code



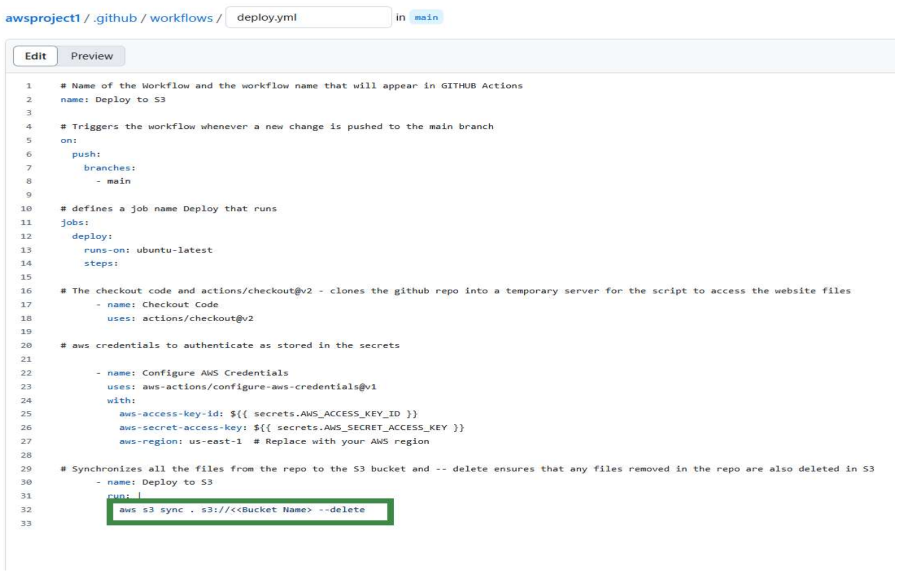
```yaml
# Name of the Workflow and the workflow name that will appear in GITHUB Actions
name: Deploy to S3

# Triggers the workflow whenever a new change is pushed to the main branch
on:
  push:
    branches:
      - main

# defines a job name Deploy that runs
jobs:
  deploy:
    runs-on: ubuntu-latest
    steps:

# The checkout code and actions/checkout@v2 - clones the github repo into a temporary server for the script to access the website files
      - name: Checkout Code
        uses: actions/checkout@v2

# aws credentials to authenticate as stored in the secrets

      - name: Configure AWS Credentials
        uses: aws-actions/configure-aws-credentials@v1
        with:
          aws-access-key-id: ${{ secrets.AWS_ACCESS_KEY_ID }}
          aws-secret-access-key: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
          aws-region: us-east-1  # Replace with your AWS region

# Synchronizes all the files from the repo to the S3 bucket and -- delete ensures that any files removed in the repo are also deleted in S3
      - name: Deploy to S3
        run: |
          aws s3 sync . s3://<<Bucket Name> --delete
```

61. Create a repo with public access and provide a description of why you are creating the repo.
62. Click on Actions and click on ‘Simple Workflow’



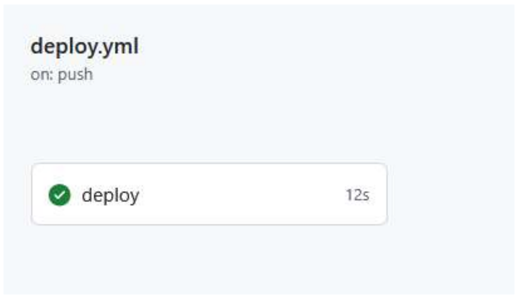
Simple workflow
By GitHub
Start with a file with the minimum necessary structure.
Configure

63. Click on configure
64. Rename it to deploy,yml and past the code
65. Do not forget to update the bucket name

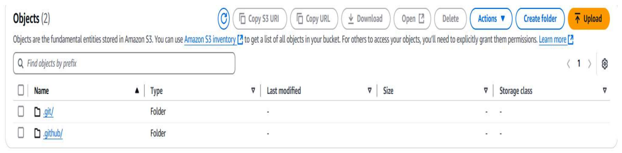


```yaml
# Name of the Workflow and the workflow name that will appear in GITHUB Actions
name: Deploy to S3

# Triggers the workflow whenever a new change is pushed to the main branch
on:
  push:
    branches:
      - main

# defines a job name Deploy that runs
jobs:
  deploy:
    runs-on: ubuntu-latest
    steps:

# The checkout code and actions/checkout@v2 - clones the github repo into a temporary server for the script to access the website files
      - name: Checkout Code
        uses: actions/checkout@v2

# aws credentials to authenticate as stored in the secrets

      - name: Configure AWS Credentials
        uses: aws-actions/configure-aws-credentials@v1
        with:
          aws-access-key-id: ${{ secrets.AWS_ACCESS_KEY_ID }}
          aws-secret-access-key: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
          aws-region: us-east-1  # Replace with your AWS region

# Synchronizes all the files from the repo to the S3 bucket and -- delete ensures that any files removed in the repo are also deleted in S3
      - name: Deploy to S3
        run: |
          aws s3 sync . s3://<<Bucket Name> --delete
```

66. Click on commit changes
67. The deploy code is available.
68. If you get an error, check credentials is correct while creating the secret key.
69. Also check permissions for the IAM user if you get an error.
70. There are no index or error files in my GitHub and the files got deleted in S3 bucket automatically.



71. Navigate to S3 and the files are not available.



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

Name	Type	Last modified	Size	Storage class
.git/	Folder	-	-	-
.github/	Folder	-	-	-